GOLDSMITHS, UNIVERSITY OF LONDON

DOCTORAL THESIS

Handling Class Imbalance Using Swarm Intelligence Techniques, Hybrid Data and Algorithmic Level Solutions

Author:
Haya ALHAKBANI

Supervisor:
Dr. Mohammad AL-RIFAIE

A thesis submitted in fulfilment of the requirements
for the degree of Doctor of Philosophy
in the

Department of Computing
Goldsmiths, University of London

March 14, 2019

Declaration of Authorship

I, Haya ALHAKBANI, declare that this thesis titled, “Handling Class Imbalance Using Swarm Intelligence Techniques, Hybrid Data and Algorithmic Level Solutions” and the work presented in it are my own. I confirm that this thesis has not been previously submitted for a degree or other qualifications at any other university.

Signed:

Date:

Acknowledgements

After months of study, perseverance, and many laborious hours thinking about my research, the day of completion of this research has finally arrived. This research tested my patience, and my willpower. It wasn't easy, but it was always a labour of love. I didn't go on this journey alone. In these next few paragraphs I'd like to thank all of those who made this achievement possible. I am so indebted to my supervisors, friends, and most importantly my family for making this dream a possibility.

First and foremost, I would like to thank God for all the strength, knowledge, opportunity, and for the chance to conduct this research study. Through him, all things are possible and it's through his grace that I have reached this milestone.

Next, I would like to thank my first supervisor, Dr. Mohammad al-Rifaie, for his generous advice and support that helped me in every step of the way, from the early stages of my research until the writing of this thesis. I'm thankful for your involvement in this project. Believe me when I say that I don't think I could have gotten through this without your assistance and encouragement.

I would also like to thank my second supervisor, Prof. Robert Zimmer, for all of his insights into this project. I am appreciative of all the brilliant minds who surrounded me during this project; I was so glad to have your perspective and ideas to think about my research from different angles. Thank you so much for working with me.

To the computing department at the university: I don't know where I would be without you. Thank you for every kind word, conversation, and even every smile that I shared with each and every one of you. You have no idea what even the smallest exchanges meant to me. They're what kept me going when I didn't think I could get this research done.

Thank you also to my family. Thank you, mother and father, for instilling in me a confidence and work ethic that helped me overcome challenges and have the drive to get this project done. I am so fortunate to have the love of wonderful parents to keep me humble, and to always remind me of what's most important in this life. Thank you to all my brothers and sisters who were a sounding board for me when I was frustrated and for all of the lunches, cups of coffee, and most of all, for the support. Thank you Maha and Alhanouf, for the love, the help and the smiles. I am so blessed to have you.

Thank you to my loving husband. You have been my pillar of support throughout this whole journey. Thank you for always listening to me talk about my experiments, for letting me bounce ideas off you, and for sharing your life with me. I am truly lucky to have a friend and partner like you in my life and I'm very grateful every day for you.

Lastly, I would like to thank my son. I had the privilege of delivering my boy during the course of my studies. My sweet Abdulaziz, it's you who I wake up for every single day. Your existence on this planet is what pushes me forward. Juggling motherhood and my studies isn't easy, but you bring to my life more joy and happiness than I ever thought was possible. This accomplishment is ours together, and I am so proud to be your mother. Finishing this thesis is a great achievement, but nothing could make me more honoured in this life than to be your mother. You have always been the centre of why I was determined to finish this research, and I am so glad you came into my life...

Abstract

Goldsmiths, University of London

Doctor of Philosophy

Handling Class Imbalance Using Swarm Intelligence Techniques, Hybrid Data and Algorithmic Level Solutions

by Haya ALHAKBANI

This research focuses mainly on the binary class imbalance problem in data mining. It investigates the use of combined approaches of data and algorithmic level solutions. Moreover, it examines the use of swarm intelligence and population-based techniques to combat the class imbalance problem at all levels, including at the data, algorithmic, and feature level. It also introduces various solutions to the class imbalance problem, in which swarm intelligence techniques like Stochastic Diffusion Search (SDS) and Dispersive Flies Optimisation (DFO) are used. The algorithms were evaluated using experiments on imbalanced datasets, in which the Support Vector Machine (SVM) was used as a classifier. SDS was used to perform informed undersampling of the majority class to balance the dataset. The results indicate that this algorithm improves the classifier performance and can be used on imbalanced datasets. Moreover, SDS was extended further to perform feature selection on high dimensional datasets. Experimental results show that SDS can be used to perform feature selection and improve the classifier performance on imbalanced datasets. Further experiments evaluated DFO as an algorithmic level solution to optimise the SVM kernel parameters when learning from imbalanced datasets. Based on the promising results of DFO in these experiments, the novel approach was extended further to provide a hybrid algorithm that simultaneously optimises the kernel parameters and performs feature selection.

Contents

List of Figures

List of Tables

List of Abbreviations

ACO	Ant Colony Optimisation
ANN	Artificial Neural Network
C	Misclassification cost
DE	Differential Evolution
DFO	Dispersive Flies Optimisation
ED	Euclidean Distance
GA	Genetic Algorithm
GP	Genetic Programming
K-NN	K-Nearest Neighbours
LDA	Linear Discriminant Analysis
LR	Logistic Regression
NB	Naive Bayes
PSO	Particle Swarm Optimisation
SDS	Stochastic Diffusion Search
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machine

This thesis and my academic achievement is
dedicated to my parents and my husband for their
endless motivation and support...

Chapter 1

Introduction

Over the last decade, there has been a rapid increase in the worldwide volume and velocity of data due to the unparalleled growth of data gathering via internet-connected devices. Data volume of data is defined as the amount of data and velocity is defined as the speed of data generation (Gandomi and Haider, 2015). The proliferation of the Internet of Things (IoT) provides a massive data lake from which increasingly large datasets are derived fast. For example, Twitter has 330 million monthly active users with an average of 500 million tweets being sent per day [1]. This is a large amount of data containing important information. Thus, it is important to speed up data processing to quickly generate information. However, datasets are rendered and useless unless meaning can be accurately derived from structured, unstructured, and semi-structured data. Data mining and analytic approaches are therefore crucial. Data mining technologies have been adopted by various sectors, including banking, retail and telecommunications, and are viewed as the foremost technologies for processing data that is now measured in tera-, peta-, and exabytes. Layers of analysis are required to both funnel the data into a database and then provide pipelines for further analytical access by organisational analysts and data scientists, who then use it to make predictive and inferential forecasts for a target population. Embedded within the analytical framework are the challenges attributed to the problem of *class imbalance*. In imbalanced datasets, the majority class is the class that has a higher number of instances, while the minority class is the one that has a comparatively lower number of instances. In terms of practical applications, class imbalance is a problematic feature in fraud detection, medical diagnosis, direct marketing campaigns, and other applications in which the class of interest is the minority class. Given the importance of cybersecurity and medical diagnoses, among other data heavy disciplines that can directly and

[1] Information on the number of tweets available from https://www.omnicoreagency.com/twitter-statistics/.

drastically impact human lives, finding optimal solutions to the class imbalance problem is imperative.

Numerous researchers have used data mining preprocessing techniques as a method of solving the issue of imbalanced data (Chawla et al. (2002), Han, Wang, and Mao (2005), Chawla (2009), and Liu et al. (2010)). However, this solution is problematic as it can involve loss of important information and over-fitting to balance the dataset. These problems have not stopped researchers from using preprocessing techniques to overcome class imbalance, though. Swarm intelligence techniques and population-based algorithms have also been used to solve data mining issues, either by applying them alone or by combining them with other preprocessing techniques. For example, swarm intelligence techniques have been used to tune the learning algorithm parameters or search for optimal features in high dimensional datasets (Unler, Murat, and Chinnam (2011) and Tiwari (2014)).

This work seeks to explore data mining preprocessing techniques to identify one that will offer the best performance in applications related to real-world problems such as direct marketing that suffer from class imbalance. It has also been found that high dimensional datasets like the microarray datasets usually suffer from class imbalance. Thus, feature selection is introduced and the possibility of using it as a solution to the class imbalance problem is investigated. Moreover, the use of swarm intelligence techniques as a solution for overcoming the class imbalance issue is explored. The next section covers this study's main objectives and explains the research methodology used in this thesis to tackle the issue of class imbalance in data mining.

1.1 Objectives and methodology

The research explores how data mining preprocessing techniques such as sampling, cost sensitive learning, and feature selection are used to solve cases where the class of interest is the minority class. It also examines the role that swarm intelligence plays in solving this issue. This will be achieved by using experimental methodology to evaluate solutions for the class imbalance problem. To accomplish this, real-world datasets will be used from resources such as the University of California, Irvine (UCI) machine learning repository [2] and then compared with other results in the literature. When setting up the experiments it is important to consider the following aspects: the definition

[2] An archive of databases for evaluating machine learning models: http://archive.ics.uci.edu/ml/index.php.

of the questions that drive the experiment, good record keeping and close examination of the utility of the knowledge gained from the experiment. The key research questions raised in this work are:

1. What are the available methods for handling the class imbalance problem?
2. How do swarm intelligence techniques help in dealing with imbalanced datasets at the data level?
3. How do swarm intelligence techniques help in dealing with imbalanced datasets at the algorithmic level?
4. How could swarm intelligence techniques, like SDS and DFO, provide a way to address feature selection?

The first research topic investigates the current solutions for handling the class imbalance problem and provides examples of how various solutions are applied. This is followed by a number of experiments whose outcomes demonstrate that a combination of various existing data mining preprocessing techniques can improve the model performance on the direct marketing dataset.

In order to answer the second research question, various swarm intelligence and population-based algorithms are presented. SDS and DFO are the algorithms discussed in most detail, and examples of how these algorithms work are provided. In experimental work, the use of SDS as a data level solution was first investigated. SDS was used to undersample the majority class. Following the promising results, SDS was extended further to undersample the majority class at the feature level using a feature level threshold. This was combined with the Synthetic Minority Oversampling Technique (SMOTE) at the data level. At the algorithmic level, grid search was used to tune the SVM parameters.

Moreover, to address the third question, DFO, another swarm intelligence algorithm, is presented and used as an optimiser for the SVM parameters to solve the class imbalance problem at the algorithmic level.

Finally, to answer the fourth question, the use of SDS to perform feature selection is investigated and the results are compared with those of other techniques from the literature on real-world imbalanced datasets. DFO is developed further to perform parameters tuning and feature selection simultaneously. The proposed approach is evaluated using two microarray datasets.

1.2 Contributions

This thesis presents research work on novel methods of data mining and swarm intelligence techniques for handling the class imbalance issue. It demonstrates the advantages of using swarm intelligence search techniques in balancing the dataset, finding a feature subset, and tuning the SVM parameters. The original contributions are made in the area of handling class imbalance and the use of swarm-based techniques. In general, the contributions of this thesis are as follows:

- Class imbalance in the direct marketing dataset: In this experiment, a model is proposed to improve the classifier performance on a direct marketing imbalanced dataset using a combination of existing techniques. At the data level, oversampling and undersampling are used. At the algorithmic level, grid search is used to optimise the C, γ and the kernel type of SVM. The model is used to overcome the class imbalance problem in a real-world dataset that was collected from a Portuguese marketing campaign for bank-deposit subscriptions (available from the UCI machine learning repository). The results indicate that the proposed approach performed competitively when compared to other models on the same dataset. The outcome of this experiment was published in the proceedings of the SAI Computing Conference, London 2016 [3].

  Alhakbani, H. A., al-Rifaie, M. M. (2016). *"Handling Class Imbalance In Direct Marketing Dataset Using A Hybrid Data and Algorithmic Level Solutions."* In SAI Computing Conference, pp 446-451, 2016. IEEE, London, UK.

- SDS and undersampling : This study investigates a new approach for balancing the direct marketing dataset using a swarm intelligence technique, SDS, to undersample the majority class on the direct marketing dataset. The outcome of the novel application of this swarm intelligence algorithm demonstrates promising results which encourage the possibility of undersampling a majority class by removing redundant data whilst protecting the useful data in the dataset. This new approach

[3] The Conference Website: http://saiconference.com/Conferences/Computing2016.

for undersampling the dataset using SDS was published in the proceedings of the 10th International Conference on Swarm Intelligence, ANTS 2016, Brussels, Belgium [4].

Alhakbani, H. A., al-Rifaie, M. M. (2016). *"A Swarm Intelligence Approach in Undersampling Majority Class."* In ANTS 2016 Tenth International Conference on Swarm Intelligence, Lecture Notes in Computer Science (LNCS). Volume 9882, pp 225-232, Brussels.

- Exploring feature-level duplications on imbalanced datasets using SDS: In this experiment, a hybrid approach is proposed to deal with real-world imbalanced datasets. The proposed model involves oversampling the minority class, undersampling the majority class as well as optimising the parameters of the classifier. The proposed model uses SMOTE to perform the oversampling on the minority class and the agents of SDS to perform an *informed* undersampling of the majority class. The use of this swarm-based technique in conducting the undersampling tasks is investigated and its impact on improving the classification results is demonstrated. Not only was the agents-led undersampling compared with random undersampling, but the results were also contrasted with other well-known techniques on nine real-world datasets. Additionally, further experiments are designed to explore the behaviour of the SDS agents during the undersampling process. The results of this work were published at the 14th European Conference on Multi-Agent Systems, EUMAS 2016, Valencia Spain [5].

  Alhakbani, H. A., al-Rifaie, M. M. (2016). *"Exploring Feature-Level Duplications on Imbalanced Data Using Stochastic Diffusion Search."* In 14th European Conference on Multi-Agent Systems (EUMAS 2016), Springer LNCS/LNAI. Valencia, Spain.

- Feature selection using SDS: In this experiment, a new approach for feature selection is introduced in which SDS is used to find a feature subset. The proposed approach is called SDS-based feature selection (SDS-FS). The results indicate that SDS-FS outperforms the classifier performance without SDS feature selection. When compared with other methods from the literature used on datasets with a feature size greater

[4] The Conference Website: http://iridia.ulb.ac.be/ants2016/.

[5] The Conference Website: http://eumas-at2016.webs.upv.es/

than 10, SDS-FS offers a competitive performance. The outcome of this novel approach for feature selection was published in the proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 2017, Berlin, Germany [6].

Alhakbani, H. A., al-Rifaie, M. M. (2017), *"Feature Selection using Stochastic Diffusion Search,"* GECCO-2017: Genetic and Evolutionary Computation Conference, pp. 385-392, Association for Computing Machinery (ACM).

- Optimising SVM parameters using DFO: Although substantial work has been done on tuning SVM parameters using various swarm intelligence techniques such as Particle Swarm Optimisation (PSO), the use of other swarm-based techniques remains unexplored. In this experiment, a novel approach that uses DFO to optimise the SVM kernel parameters, C and γ, is proposed. This is to perform cost sensitive learning to improve the classifier's performance on imbalanced datasets. The use of the swarming behaviour of the flies in DFO has been evaluated on eight real-world datasets. The proposed approach has been compared with other techniques for parameters tuning, such as PSO and grid search. The results demonstrate that the proposed approach outperforms other techniques on the same datasets. This proposed, novel approach was published in the proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS), 2017, Prague, Czech Republic [7].

  Alhakbani, H. A., al-Rifaie, M. M. (2017), *"Optimising SVM to classify imbalanced data using Dispersive Flies Optimisation,"* Proceedings of the 2017 Federated Conference on Computer Science and Information Systems. IEEE.

- Feature selection and parameters tuning using DFO: After the use of DFO for SVM parameters tuning, the algorithm was further extended to perform feature selection. In this experiment, another novel approach is proposed to simultaneously tune the kernel's parameters while finding a subset of features. The model has been evaluated using two

[6] The Conference Website: http://gecco-2017.sigevo.org/index.html/HomePage.

[7] The Conference Website: https://fedcsis.org/2017/.

well-known microarray datasets: small round blue cell tumours (SRBCT) and Leukemia MLL datasets. The performance of the proposed approach was compared with other models from the literature on the same datasets. The results indicate that the proposed DFO-based approach outperforms a variety of techniques from the literature.

1.3 Chapters outline

The rest of the thesis is structured as follows

- Chapter 2 begins by defining data mining and investigating why it is important. It then provides a review of current literature on the major issues in data mining, mainly the problem of class imbalance. It also presents the suggested solutions from the literature to overcome this problem. In particular, three solutions are outlined: 1) data level solutions that make use of various approaches, such as sampling, to balance the datasets; 2) algorithmic level solutions that work by adjusting the misclassification cost; 3) hybrid solutions that work by combining undersampling and oversampling until the dataset is balanced or by combining data level solutions and algorithmic level solutions for a better performance of the classification models. This chapter also highlights the importance of feature selection in improving the model's overall accuracy and outlines how it can be used to solve the class imbalance problem. Finally, the metrics used to evaluate models on imbalanced datasets are explained.

- Chapter 3 begins with a discussion of the basic concept of swarm intelligence and evolutionary computation. It then goes on to define main swarm intelligence algorithms. The chapter highlights the use of swarm intelligence in data mining by providing examples from the literature. It also provides an in-depth definition of two swarm intelligence techniques - SDS and DFO - which can be used to solve search and optimisation problems.

- Chapter 4 presents the set of experiments and obtained results for each experiment. These experiments have been the main source of practice to solve the class imbalance issue. The practice has been achieved using real-world datasets.

- Chapter 5 provides an in-depth discussion of the results based on the experiments.
- The report concludes with Chapter 6, which includes a summary of contributions thus far, along with the main achievements of this thesis. The chapter also provides plans for future work, including experiments to be conducted.

Chapter 2

Class Imbalance

This chapter presents a brief overview of data mining. The definition of data mining is followed by a description of the main approaches used to extract knowledge from data. Major issues in data mining are identified before the chapter goes on to offer an in-depth investigation of the class imbalance problem and a discussion of the proposed solutions by researchers. This includes a brief introduction to the theory of SVM and the recent approach of optimising the SVM kernel parameters to deal with the class imbalance problem. Subsequently, an introduction to feature selection is given, along with an explanation of how it can be used as a solution to class imbalance. Finally, the evaluation metrics used to measure the classification model's performance on imbalanced datasets are presented.

2.1 Introduction

In recent years, technological and scientific breakthroughs have significantly increased the availability of raw data, thereby enabling knowledge discovery and data science research to make a number of contributions to various applications, from day-to-day tasks to decision-making support systems. Data mining techniques have the functional capability to process structured data and subsequently extract meaningful patterns (Mazurowski et al., 2008). Data mining is an important field as it helps in extracting information from large raw data. It has been defined as *"the non-trivial extraction of implicit, previously unknown, and potentially useful information from data"* (Frawley, Piatetsky-Shapiro, and Matheus, 1992). Another definition for data mining is *"the process of discovering interesting knowledge from large amounts of data stored either in databases, data warehouses, or other information repositories"* (Han, Pei, and Kamber, 2011). It involves the construction of data models using available data to represent and interpret the current understanding of reality. Moreover, it provides insights and facilitates decision making. A data model describes patterns and relationships that are apparent from the analysis and slicing of the data sources (Berson, Smith, and Thearling, 2000).

Data mining algorithms analyse the data to look for patterns, so the data to be mined must be presented as a collection of examples or instances . Each example, *an instance*, is described by a number of attributes, or features values that can be either numerical or categorical. Table 2.1 shows fourteen examples of ideal and non-ideal days to play golf (Witten and Frank, 2005). Each example (row) represents a day that is described in terms of features or attributes (columns), including: outlook, the outside temperature, humidity, wind and the class label which specifies if the weather is suitable for playing golf or not (Hall, 1999).

From a set of structured examples of a target concept, a data mining algorithm can find patterns or trends in three different ways: Clustering, association rule and classification.

TABLE 2.1: Golf dataset

	Attributes				**Class label**
Example #	Outlook	Temperature	Humidity	Windy	Play
1	Sunny	85.0	85.0	False	No
2	Sunny	80.0	90.0	True	No
3	Overcast	83.0	86.0	False	Yes
4	Rainy	70.0	96.0	False	Yes
5	Rainy	68.0	80.0	False	Yes
6	Rainy	65.0	70.0	True	No
7	Overcast	64.0	65.0	True	Yes
8	Sunny	72.0	95.0	False	No
9	Sunny	69.0	70.0	False	Yes
10	Rainy	75.0	80.0	False	Yes
11	Sunny	75.0	70.0	True	Yes
12	Overcast	72.0	90.0	True	Yes
13	Overcast	81.0	75.0	False	Yes
14	Rainy	71.0	91.0	True	No

Clustering and association rule mining are *"unsupervised learning"* processes as they do not involve the prediction of values for a specific attribute (Tang et al., 2009). Clustering involves dividing the set of examples or instances into groups based on some measure of similarity. The labels do not exist in the datasets so they are not available to use at the beginning. Instead, clustering is used to generate the class labels. Association rule mining looks for useful predictive patterns or trends between any combination of features or attributes in the data. This approach is different from "*supervised learning*". In supervised learning, one of the attributes, called the *class attribute* or *dependent attribute*, can be predicted according to the values for those attributes called *independent attributes*. The class attribute can be categorical or numerical. If the class attribute is categorical, the process is called classification. However, if the class attribute is numeric, the process is called regression (Witten and Frank, 2005). Classification is defined as "*the process of finding a set of models (or functions) which describe and distinguish data classes or concepts, for the purposes of being able to use the model to predict the class of objects whose class label is unknown*" (Han, Pei, and Kamber, 2011). The objective of classification is to successfully predict the values of the class attributes of an example or instance based on the values for its independent attributes. The classifier success is usually measured using the overall predictive accuracy (more details about evaluation metrics can be found in section 2.5). Going back to the dataset example in Table 2.1, the class attribute is *play* and the independent attributes are outlook, outside temperature, humidity, and wind, with the objective being to create a model that successfully predicts if the day's weather is suitable for playing golf or not.

The goal of the classification algorithms is to achieve higher accuracy by having the maximum number of correct predictions. Thus, to explore patterns from a dataset so that future examples can be classified correctly, the datasets should be valid and well structured. Unfortunately, in many real-world classification applications, datasets are not perfect. A number of issues such as class imbalance, outliers, and missing values can affect the quality of the dataset used to train a classifier. This often leads to reduced classification accuracy. This thesis focuses on classification and its main issue, namely class imbalance. Section 2.2 briefly describes the major issues found in datasets and their effects on the data mining process. It also reviews the literature on the suggested solutions to these issues.

2.2 Major issues in data mining

There has been a rapid increase in the number of datasets all over the globe due to the unparalleled growth of globalisation and global markets (Chen and Zhang (2014) and Hashem et al. (2015)). However, these datasets suffer from issues such as missing values, outliers, or imbalance in the class distribution. Therefore, data cleaning and preprocessing techniques are important steps in data mining. These are especially used for filling in missing values, overcoming the class imbalance problem and identifying outliers. This section will briefly list the major data issues in data mining and the suggested solutions for each issue.

2.2.1 Outliers

Many data contain instances that do not conform to general behaviour. These instances are called *outliers*, and they are usually treated as noise or errors in the data that may cause a loss of important information. An outlier can be defined as "*an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism*" (Hawkins, 2013). There are various types of outliers, including global outliers, conditional outliers, and collective outliers. A global outlier occurs if the instance value is far outside the whole dataset to which it belongs. A conditional outlier, also called a contextual outlier, is when the instance is an exception or abnormality in a specific context, but not all. A collective outlier is a collection of instances that are abnormal with respect to the whole dataset (Ajitha and Chandra, 2015).

The process of finding outliers is called *outlier analysis* or *outlier detection*. At the preprocessing stage graphical presentation of the data can help in finding outliers. Outlier detection is useful in applications like credit card fraud detection, as a transaction that does not comply with the credit card behaviour is classified as an outlier and should be reported for security reasons (Bolton and Hand, 2002). Also, in medical diagnosis, patients' test results may indicate potential health problems that can be discovered by comparing with the characteristics of other patients (Hauskrecht et al., 2013). There are different outlier detection methods that use statistical measurement to find outliers. Distance-based techniques calculate distances between examples, attributes, or objects to detect any nonconformities in the dataset. Examples of the distance-based detection techniques is the density-based method, where local regions are used for identification, and the deviation-based approach, where the outliers are the instances that do not conform to any of the group characteristics (Kriegel, Kröger, and Zimek, 2010). Another approach is clustering-based outlier detection, in which similar data are grouped as clusters. In this approach, the normal instances belong to a cluster while the outliers or abnormal instances do not belong to any clusters (Ajitha and Chandra, 2015). Various algorithms based on clustering are used to detect outliers. These include Balanced Iterative Reducing and Clustering using

Hierarchies (BIRCH) (Zhang, Ramakrishnan, and Livny, 1996), Clustering Using Representatives (CURE) (Guha, Rastogi, and Shim, 1998) and Clustering Large Applications based on Randomised Search (CLARANS) (Liu and Özsu, 2009).

The choice of outlier detection technique differs according to the data type, the outlier type, and the chosen model. Some models consider outliers at different levels, for example, local level or global level. It is an important step at the preprocessing stage especially when a classifier such as *k*-NN is highly affected by the outliers and noisy data. Various outlier detection techniques have been proposed in the literature to deal with outliers in certain types of data. For example, Aggarwal and Yu, 2001 proposed a new technique for outlier detection for high dimensional datasets. This technique differs from distance-based techniques in that it overcomes the effect of dimensionality. Janakiram, Reddy, and Kumar, 2006 proposed an outlier detection technique for wireless sensor networks using Bayesian Belief Networks. The proposed approach is capable of detecting the outliers in sensor streamed data. He, Deng, and Xu, 2005 proposed a model for outlier detection in categorical data. The proposed approach performed well on real and synthetic datasets. More details on various outlier detection techniques can be found in (Hodge and Austin (2004), Patcha and Park (2007), Kriegel, Kröger, and Zimek (2010), Zhang, Meratnia, and Havinga (2010), and Divya and Babu (2016)).

2.2.2 Missing values

Another issue in data mining is missing values. Data cleaning is a stage at which some work is required to handle missing values in the dataset. If the dataset has missing values, the performance of the algorithm will be poor (Acuna and Rodriguez, 2004). Datasets can have missing values for various reasons, such as failure to record some observations, data warehouse corruption, and errors in the data entry stage.

There are classifiers such as Naïve Bayes (NB) that can handle missing values by removing the attributes with missing values when calculating the probabilities. However, when the classifiers cannot handle missing values, as happens with Linear Discriminant Analysis (LDA), solutions for dealing with missing values are required at the data cleaning stage or the preprocessing stage (Aisha, Adam, and Shohaimi (2013) and Acuna and Rodriguez

(2004)). According to Han, Pei, and Kamber, 2011, there are six different ways to handle missing values:

1. Ignore the tuple by deleting the case which contains attributes with missing values. The case deletion method is one of the easiest methods, and works by deleting all cases that have missing values. However, this approach can cause loss of important information in the case.

2. Manually fill in missing values, which is time consuming, especially when the dataset is large.

3. Use a specified constant to fill in missing values, for example, "unknown", which may let the classifier think that the instance is an example of an interesting group.

4. Use measures like mean and median to fill in the missing values. This is known as mean and median imputation.

5. At the class level, measures like mean or median can be used to fill in the missing values within the class. This is called class mean or median imputation.

6. Fill in the missing value with the most probable value. This value can be specified using inference-based tools, such as using decision tree induction to predict a missing value in a dataset. The process is known as imputation using decision tree algorithms.

It is important to explore the dataset well and understand the data mining model goals and objectives in order to choose the optimal method for dealing with the missing values, as some applications do not treat missing values as an error. For example, when applying for a job, candidates may be asked to supply a previous employer's fax number. Candidates who do not know the fax number may leave this blank. Therefore, forms should allow for answers like "none" or "not applicable" or even analyse other values like "do not have" to simplify the process and not waste time on cleaning missing values that are not considered as errors.

Numerous research studies have been conducted to explore various solutions for handling the missing values in specific cases at the data preparation stage (Harel and Zhou (2007) and Sessa and Syed (2016)). For example, Poolsawad et al., 2012 designed methodologies based on feature selection to handle many issues, including missing values in clinical datasets. Another example is the work done by Acuna and Rodriguez, 2004, in which the authors evaluated four different techniques for dealing with the problem of missing values: median imputation, mean imputation, *k*-NN imputation, and the case deletion method. In this experiment, the evaluation was carried out on twelve datasets using two classifiers: LDA and *k*-NN. The experimental results indicate that there is not a big difference in the misclassification error rate between the case deletion and the imputation methods for both classifiers when dealing with datasets that have a small amount of instances that include missing values. However, there is a great difference when dealing with datasets with a high percentage of missing values such as the Hepatitis dataset [1].

In summary, there are many techniques for handling missing values. However, different cases require different solutions. Moreover, each technique has its own advantages and disadvantages. The simplest approach is to delete the examples with missing values, but this may cause loss of data. Therefore, it is important to fully understand the goals of the prediction model and its way of dealing with missing values before choosing the technique that is suitable for solving the issue. More details on other methods for dealing with missing values can be found in (Brown and Kros (2003), Pelckmans et al. (2005), Li, Stuart, and Allison (2015), Magnani (2004), and Lee and Simpson (2014)).

2.2.3 Imbalanced datasets

Another major issue in data mining is dealing with imbalanced datasets. These are datasets in which the number of the majority class instances significantly outnumbers that of the minority class instances. This problem is called *class imbalance* and most data contains this problem. It often occurs in customer-related data, churn prediction, medical diagnosis, text categorisation, and fraud detection, where the class of interest is the minority class. The imbalance in datasets is a major challenge in data mining because in many applications, the cost of misclassifying the minority class is high, such as in direct marketing, where businesses are interested in identifying potential buyers, and in charities when identifying potential donors.

The class imbalance issue has received attention in the literature (Ling and Li (1998), Chawla et al. (2002), Chawla (2005), Chawla (2009), and Batuwita and Palade (2013)). This is because data mining models usually tend to be

[1] The dataset can be found in the UCI machine learning repository https://archive.ics.uci.edu/ml/datasets/hepatitis.

influenced by the majority class. Therefore, the minority class is usually misclassified, leading to poor performance and low predictive accuracy. In an even worse scenario, minority examples are considered as outliers of the majority class. Thus, they are ignored in the learning process. Moreover, the algorithm goal is to maximise accuracy, so it assumes that the class distribution is equal and the misclassification cost for all classes is the same. However, this is not always the case (Thai-Nghe, Gantner, and Schmidt-Thieme, 2010).

As previously mentioned, the imbalance between the classes is a major challenge when learning from imbalanced datasets. However, in some cases when learning from an imbalanced dataset, the classification algorithms are able to learn from the datasets and provide good accuracy. This indicates that the class imbalance ratio is not the only issue that affects the performance of the classifier when learning from imbalanced datasets. Therefore, more work is needed to analyse other factors when learning from imbalanced datasets (Japkowicz and Shah, 2011). One factor is the sparsity of the minority class, which is the distribution of the data within the minority class itself. In research by Jo and Japkowicz, 2004, the authors found that the main issue that affects classifier performance on imbalanced datasets is the small disjuncts within the dataset. The authors found that by focusing on the small disjuncts problem, performance was improved when compared to the performance of the classifier if the focus was only on the imbalance ratio. Another important factor is class overlapping. Stefanowski, 2013 studied the effect of overlapping, along with other factors such as the sparsity of the minority class. The experimental results indicate that both class decomposition and class overlapping cause difficulty when learning from imbalanced datasets. In a study by Napierala and Stefanowski, 2016, the authors suggested that analysing the local characteristics of the minority examples and defining their types are important steps when learning from imbalanced datasets. The authors defined four types of minority class example: safe, borderline, rare examples and outliers, in which the last three are classified as unsafe examples.

Various solutions have been suggested to overcome the class imbalance problem (Chawla et al. (2002), Han, Wang, and Mao (2005), Kubat and Matwin (1997), Berson, Smith, and Thearling (2000), and Chawla (2005)). They follow three main approaches, being applied at the data, algorithmic, or hybrid level. At the data level, the solutions work by applying various sampling techniques to balance the dataset. At the algorithmic level, the solutions work by modifying existing learning algorithms to overcome the bias toward the majority class and adapting them so that they learn from imbalanced datasets with a skewed distribution. Hybrid algorithms combine both approaches: data and algorithmic level. Although there is increased awareness of the importance of imbalanced data and available solutions, many of the key issues are still open and occur more often, especially when dealing with big data. More work is needed to analyse the dataset factors when learning from imbalanced datasets and to investigate the possibility of combining various techniques to overcome the class imbalance problem.

This research will specifically address the problem of *class imbalance* in classification. The remainder of this chapter is organised as follows: Section 2.3 explains the problem and investigates the suggested solutions from the literature for this major issue. It also explores the possibility of applying variations of those solutions to imbalanced datasets and highlights different perspectives of the class imbalance problem and how it is likely to be solved based on various approaches further predicated on domain specificity. Moreover, there exists a variety of combinatorial solutions that merge sampling techniques that are used together, or sampling techniques are joined with algorithmic solutions to increase the chances of accurate classification; these are presented in brief. Section 2.4 explains feature selection as a solution to the class imbalance problem and provides a description of the main methods for feature selection. Section 2.5 defines the performance metrics used to evaluate the classification model on imbalanced datasets.

2.3 Background on the class imbalance problem

In a number of real-world classification applications, such as software prediction, oil spill detection from satellite images, detection of online credit card fraud, and diagnosis of rare diseases, the training data might be imbalanced (Lesperance et al. (2001), Weiss (2004), and Chawla (2009)). The class imbalance problem occurs if the number of instances or examples in some classes is much smaller than in other classes. An example of an imbalanced dataset is shown in Fig 2.1, in which the majority class counts for 70% of the dataset and the minority class counts for only 30%. When the learning model is applied to an imbalanced dataset, it can be biased or completely out of context, as the algorithms being applied to classify the dataset can potentially ignore the important minority.

There are many reasons for the class imbalance problem. One main reason is a limitation on the data collection process; e.g., high cost or privacy problems. For example, biomedical data that are derived from a rare disease and an abnormal condition or data that are obtained via expensive experiments can often limit the size of the dataset. Moreover, most demographic information, such as age, gender, and income, is private and should never be disclosed to third parties (Kadiyala and Srivastava, 2011). Given the recent European Union (EU) adoption of the General Data Protection Regulation (GDPR) [2], this will likely further limit the amount of data available for analysis.

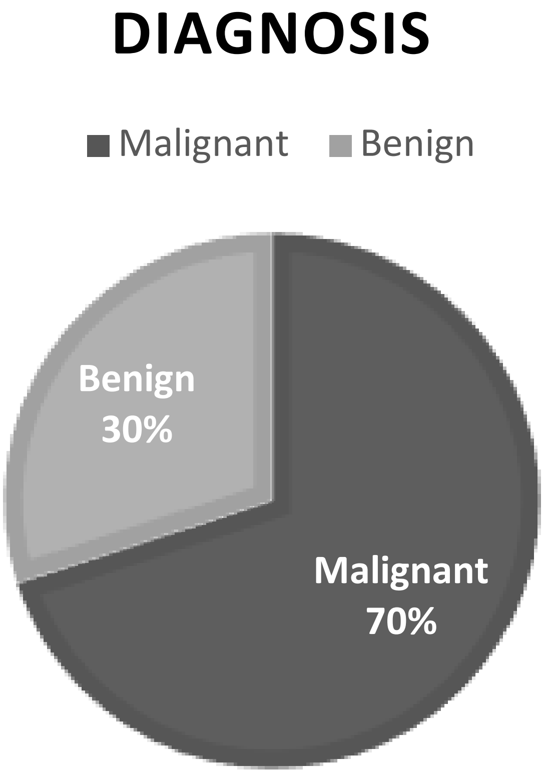

FIGURE 2.1: Breast cancer imbalanced dataset

The key issue regarding an imbalanced learning problem is that its dataset restricts the performance of most standard learning algorithms, and such algorithms often assume balanced class distributions. Problems like these lead to biased machine learning algorithms, which are biased towards the majority class since these types of algorithms attempt to maximise the overall accuracy. Conversely, in the event of complex imbalanced datasets, these algorithms will not reveal the distributive characteristics of the dataset, thus

[2] EU data protection and privacy regulator, more details can be found in their website: http://www.eugdpr.org/.

leading to unfavourable accuracies within the dataset classes. In the biomedical field, this issue is particularly crucial, since learning from imbalanced data can help us gain useful knowledge from which we can then make important medical decisions.

An example of a biased SVM between two classes (red and black marks) is shown in Fig 2.2. The SVM gives a biased separate line because its main objective is to have maximum classification accuracy that is dominated by the majority class. As shown in Fig 2.2, the SVM line is shifted toward the minority class that is away from the majority class.

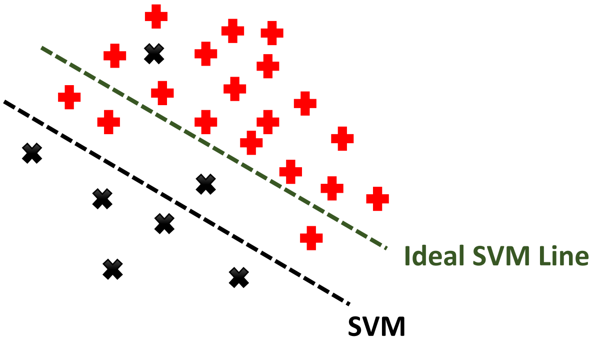

FIGURE 2.2: An example of a biased SVM

The problem of class imbalance can occur in binary, or two-class classification problems and in multi-class classification problems. The next section will briefly look into the multi-class imbalance problem and the available solutions.

Multi-class imbalance

In an imbalanced dataset, one class immensely outnumbers the other and this represents a traditional binary class imbalance problem. Yet, real-world problems frequently require classification between more than two classes (Sahare and Gupta, 2012). The multi-class imbalanced datasets problem expands on the traditional binary class imbalanced dataset problem, in which a dataset has N classes rather than two. Given the existing binary class imbalance problem, the impact of this imbalance presents a much greater risk when dealing with multi-class imbalanced datasets.

In recent years, various researchers have explored imbalanced data classification problems and worked on improving the performance of classification models (Japkowicz and Stephen (2002), Zhou and Liu (2006), and Seiffert et al. (2010)). However, the majority of these studies only address binary classification problems and only a few have been expanded to deal with multi-class imbalanced datasets. One solution is to decompose multi-class problems into binary class problems; in other words, classifying each individual class over other classes (Dietterich and Bakiri, 1991). However, there are a few significant limitations to this approach: one is expensive training, which is required to learn about the identification model for each class; the second is the difficulty of comparing classes, as each has different properties; and the third is that imbalanced distribution can be further worsened in small classes (Dua and Du, 2011). Data level solutions, such as sampling, can be directly applied to a multi-class problem. However, this is not the case for many algorithmic level techniques (Tang et al., 2009).

Despite the significant amount of research on dealing with binary classification, there remain many open challenges and issues that need to be addressed. The remainder of this thesis will mainly assume a binary or two-class classification in which datasets suffer from class imbalance. In such cases, the classes are well identified: one class counts for the majority while the other counts for the minority. This is because class imbalance in binary classifications exists in many real-world applications such as fraud detection, medical diagnosis, and direct marketing. Section 2.3.1 briefly explains the various solutions to overcome the class imbalance problem.

2.3.1 Solutions to the class imbalance problem

Numerous solutions to the class imbalance issue have been proposed at the data and algorithmic levels. Some solutions combine both levels, and are known as hybrid methods. At the data level, different solutions have been recommended to resolve the class imbalance problem (Wang, 2011). These solutions work through the application of sampling techniques to resolve the imbalance of data, including undersampling of the majority class and oversampling of the minority class (Kubat and Matwin (1997) and Chawla, Japkowicz, and Kotcz (2004)). The algorithmic solutions take into consideration adjustment to the costs of various classes. This section explores the various solutions to class imbalance, including data level solutions, algorithmic level solutions and hybrid solutions.

Data level solutions

At the data level, the solutions work by applying various sampling techniques to balance the dataset. Sampling is the simplest and most basic approach to address a class imbalance problem. Here, the original class frequencies or the number of instances in each class are changed at the preprocessing stage to balance the datasets. There is no need to change the learning algorithm as the balanced dataset is suitable for a standard learning algorithm.

Technically speaking, sampling methods involve modifying the distribution of an imbalanced dataset until it balances. This is because a number of studies have found that the classifier performance improves when applied to a balanced dataset (Weiss and Provost (2001), Estabrooks, Jo, and Japkowicz (2004), and García, Luengo, and Herrera (2015)). Furthermore, undersampling, oversampling, or both could be used. The sampling amount in each class is empirically chosen (Chawla, 2009).

Random oversampling balances the dataset by creating duplicate instances of the minority class, as shown in Fig 2.3. While this process has performed well in some studies, given that it only duplicates existing examples and does not add new information about the class, it has been argued that this method results in making the decision region for the minority class very specific, which can ultimately lead to an over-fit (Kotsiantis, Kanellopoulos, and Pintelas, 2006). Over-fitting means the model is perfectly modelling the training data and lacks generalisation ability. In addition to over-fitting, some researchers have highlighted that computational costs are higher when oversampling is applied, as more examples are added for processing (Provost, 2000).

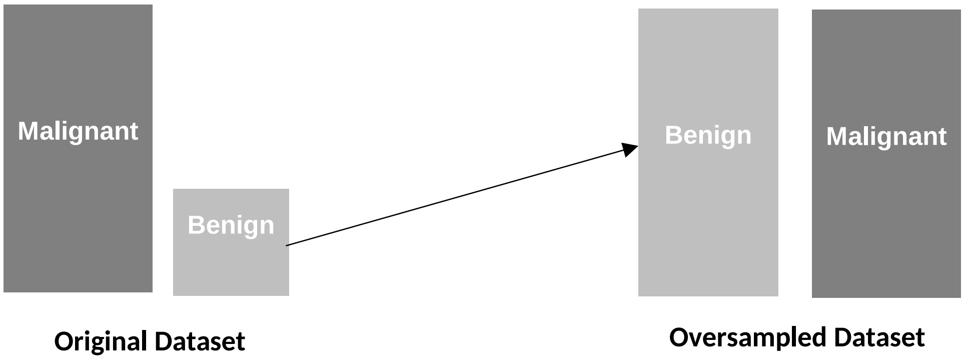

FIGURE 2.3: Oversampling the breast cancer dataset

Another solution is to randomly undersample the dataset by decreasing the instances in the majority class, see Fig 2.4. As with random oversampling, the process has performed well in some cases, but it has exhibited the disadvantage of removing useful information from the training dataset (García et al., 2016). Therefore, many algorithms combine both of these sampling types in order to benefit from the best of both (Estabrooks, Jo, and Japkowicz (2004) and Liu, Wu, and Zhou (2009)).

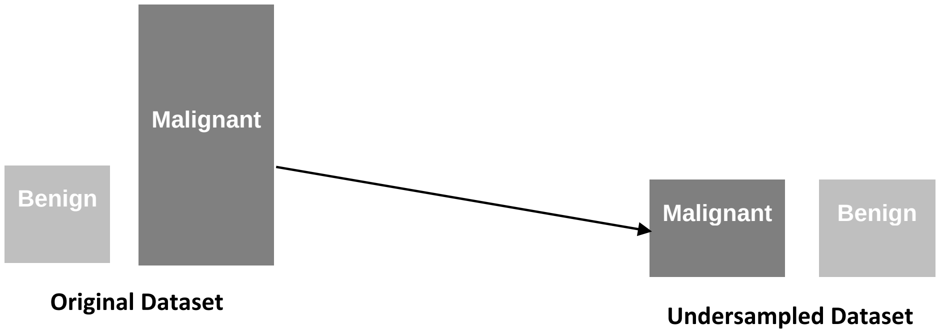

FIGURE 2.4: Undersampling the breast cancer dataset

Despite the issues mentioned above, sampling is still used to address class imbalance. Researchers have made valuable progress in finding better ways to undersample the majority class and oversample the minority class to overcome issues like over-fitting in random oversampling and the loss of information in random undersampling. For example, a proposed model uses an approach in which the minority class is oversampled by creating synthetic examples until the dataset is balanced (Chawla, 2005). The approach is called SMOTE. It is a state-of-the-art informed oversampling process where for each minority class, it finds the *k*-NN, randomly selects a number of these neighbours and generates synthetic examples, as shown in Fig 2.5. Therefore, SMOTE creates synthetic examples of the minority class based on the feature space, not on the data space (Gao, 2015). It has been widely used as a solution to the class imbalance problem (Wang (2008), Jeatrakul, Wong, and Fung (2010), and Lusa (2013)).

SMOTE works by taking each instance of minority class, x_i, in the dataset and find its *k*-NN. It then randomly picks one of the nearest neighbours, $x_{\mathrm{i},1}$, calculates the *d*-dimensional feature difference vector x_i - $x_{\mathrm{i},1}$ and assumes that there are *p* predictors in the dataset. The new synthetic example, x_new, is calculated by multiplying a random number σ between [0, 1], and then adding x_i: $x_\mathrm{new} = x_\mathrm{i} + (x_\mathrm{i} - x_{\mathrm{i},1}) \times \sigma$. As a result, the new synthetic example,

x_{new}, is an instance that is along the line between x_{i} and $x_{\text{i,1}}$. The process is repeated until the dataset is balanced (Gao, 2015).

SMOTE generates instances only within or between the available examples and never creates instances outside the border. Therefore, SMOTE never creates new regions of minority instances. Moreover, it can be applied in binary classification with continuous features type only. Despite the limitations, SMOTE overcomes random oversampling by generalising the decision region for the minority class as it does not necessarily cause over-fitting (Chawla et al., 2002). The success of SMOTE has led to new variants, such as borderline-SMOTE1 and borderline-SMOTE2, in which only samples close to the borderline or the decision boundary are chosen for oversampling (Han, Wang, and Mao, 2005). This is because samples or examples in this area are very important in determining the optimal decision boundary. Various studies have found that sampling the entire minority class is not always necessary, and by focusing on the borderline area, the classification algorithm achieves better performance (Wang, 2008).

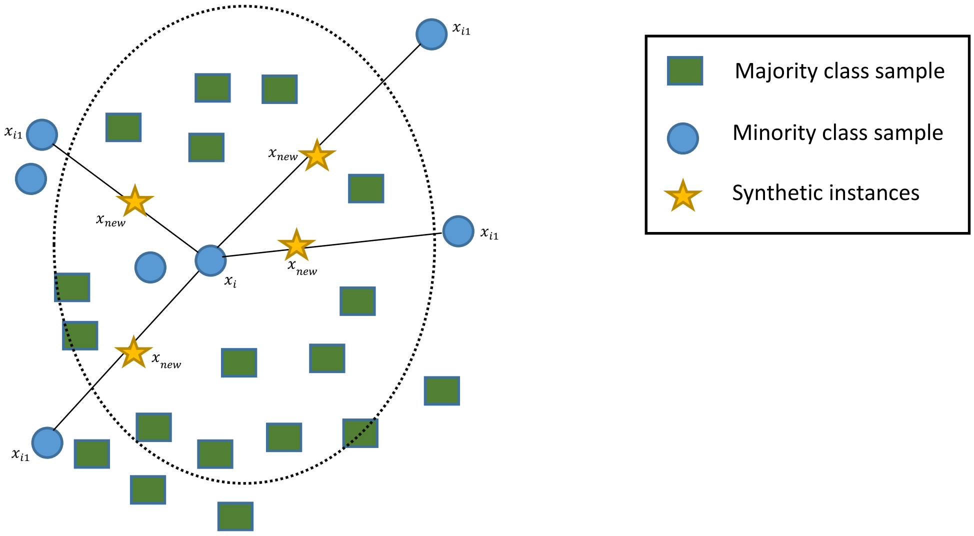

FIGURE 2.5: SMOTE oversampling in two-dimensional feature space

When it comes to undersampling, researchers have found a way to avoid losing useful information by performing random undersampling such as Focused Undersampling (FUS), in which only samples located at the borderline are removed and clustering techniques are used in selecting a subset of training data (Yen and Lee (2006) and Dubey et al. (2014)). This is because removing the instances on the border of the majority class will clarify the decision boundary between the majority and minority class. Moreover, Kubat and Matwin, 1997 proposed a one-sided selection undersampling technique to find a representative of the majority class instances and remove noisy or borderline majority instances to balance the dataset in a more informative method. The study has been applied to various datasets; it has performed well in cases where either the accuracy of the positive instances or the negative instances is low. Sowah et al., 2016 proposed an undersampling technique known as the Cluster Undersampling Technique (CUST), which is capable of improving the model performance when learning from imbalanced datasets. The performance of the proposed undersampling approach has been compared with other sampling methods on sixteen real-world imbalanced datasets using C4.5 decision tree and One Rule (OneR). The results indicate that the proposed undersampling approach provides higher performance when using the geometric mean (G-mean) and the area under receiver operating characteristic curve (AUC) as evaluation metrics. Yen and Lee, 2009 proposed another cluster based undersampling technique to select training data. The experimental results show that the cluster based undersampling approach gives better results as opposed to other undersampling techniques. Ofek et al., 2017 proposed a model for overcoming the class imbalance in binary classification which is known as Fast Clustering-Based Undersampling (Fast-CBUS). Another informed undersampling approach is known as the Nearest Cleaning Route (NCR) (Laurikkala, 2001). It removes examples from the majority classes that are misclassified when using *k*-NN. This approach is based on the Edited Nearest Neighbour Rule (ENNR) (Wilson, 1972). When comparing the proposed approach with random undersampling and random oversampling, results show that the proposed approach gives higher sensitivity .

Tomek link algorithm is another informed undersampling approach, in which Tomek links are removed from the majority class (Tomek, 1976). Tomek links are pairs of different instances (one is from the majority class and the other from the minority class) that are close together. In other words, two points x and y are Tomek linked if x is the closest point to y, y is the closest point to x and they are from different classes (He and Ma, 2013). The goal is to clarify the border between the classes, distinguish the majority and minority regions and decrease the class overlap. However, borderline undersampling removes examples close to the decision boundary. This is done only to clear the boundary between the classes and not to remove majority class instances that exist in the minority cluster. Tomek link has also been used as a cleaning method. Usually, classes are not well defined as some majority class instances might be in the minority class space and vice versa, which creates a class overlap. Thus, oversampling the minority class and introducing more samples in the majority class space can increase the overlap. Tomek link therefore can be used to create better-defined majority and minority clusters by not only removing majority classes instances but also minority class instances (Batista, Prati, and Monard, 2004). This approach has been widely used in undersampling the majority class instances and has been combined with other sampling techniques (Batista, Prati, and Monard (2004) and Elhassan et al. (2016)). For example, Batista, Bazzan, and Monard, 2003 combined Tomek link for undersampling and SMOTE for oversampling to combat the class imbalance problem in Bioinformatics.

Another solution to address class imbalance is at the algorithmic level, in which the learning algorithm is modified to tackle the problem. The following section introduces the algorithmic level solution to the class imbalance problem.

Algorithmic level solutions

At the algorithmic level, valuable progress has been made in finding better ways to overcome the disadvantages of sampling by investigating possible algorithmic solutions (Elkan (2001) and Thai-Nghe, Gantner, and Schmidt-Thieme (2010)). In any machine learning process, various types of costs are incurred, such as misclassification costs, costs of training, and costs of testing. In classification tasks, the goal is to reduce the number of misclassified examples and increase the number of correctly classified examples, which are known as *misclassification costs*. Algorithmic level solutions, or costs sensitive learning, aim to reduce misclassification costs when conducting a learning process. This is because misclassification costs are the most important type of costs when it comes to classification tasks (Weiss, McCarthy, and Zabar, 2007), especially in fields like medical diagnosis datasets, which suffer from class imbalance because the cost of classifying a cancer patient as a non-cancer patient is very high.

When learning from imbalanced data and building a classifier, it is important to consider misclassification costs. Some learning algorithms assume that misclassification costs are the same, but that is not true in all cases; a stark example would be in medical diagnosis and tumour detection, where the cost of misclassifying cancerous cells could potentially lead to risking patients' health. Moreover, the class imbalance ratios vary as they are not always the same and can be as low as 1:100 or as extreme as 1:100000. This problem can be solved at the algorithmic level by assigning different costs to the samples of the classes or by adjusting the probabilistic estimate at the tree leaf when using a decision tree (Chawla, Japkowicz, and Kotcz, 2004). This means that the majority class becomes less important relative to the minority class when engaging in the training phase. Assigning different penalty constants for the negative and positive samples has been proved highly effective, as concluded by Thai-Nghe, Gantner, and Schmidt-Thieme, 2010 in their research on cost sensitive learning methods for imbalanced datasets.

Applying an algorithmic solution requires an awareness of the appropriate learning algorithms and the reasons for their failure when mining skewed class distributions, in addition to an awareness of the application domain. It also necessitates an understanding of why the learning algorithm does not perform in the event of an uneven class distribution of available datasets. Numerous classification methods have been proven to perform well with imbalanced datasets. These include adjusted *k*-NN, SVM and Genetic Programming (GP) (Ganganwar, 2012). SVM has advantages that gives it unique features and makes it popular, including speed and high accuracy (Zhu, Liu, and Yu, 2002). For example, real-world applications such as bioinformatics (Schölkopf, Tsuda, and Vert, 2004), churn prediction (Rodan et al. (2014) and Rodan and Faris (2015)) and bankruptcy prediction (Wu et al., 2007) have reported using SVMs. However, when a SVM classifier is trained on an imbalanced dataset, it usually produces a model that is biased toward the majority class, as shown in Fig 2.2. There are various approaches to effectively applying SVM on imbalanced datasets. Batuwita and Palade, 2013 defined two approaches for SVM to handle the class imbalance: re-sampling methods which are known as external solutions and algorithmic modifications to SVMs which are known as internal methods. At the algorithmic level, solutions to this issue include, but are not limited to, assigning different weight or penalties to the majority and minority classes, in which a high penalty is assigned to the minority class. By assigning different costs, the model has been found to produce good results (Krishnapuram, Yu, and Rao, 2011).

In terms of classification algorithms, the focus of this thesis is on SVMs. The next section provides a brief literature review for the use of SVM when learning from imbalanced datasets and explains the main algorithmic solution for overcoming the problem.

SVM and class imbalance

SVM is a widely used machine learning algorithm that has been applied to solve classification problems in many real-world applications (Chang and Lin, 2011). This is due to its advantages from both a theoretical and practical perspective, such as the high generalisation ability, the mathematical background and the ability to perform non-linear classification solutions (Batuwita and Palade, 2013). SVMs work well when applied to balanced datasets, but they could also produce a biased model with imbalanced datasets. Various data preprocessing techniques and algorithmic level solutions have been proposed to overcome the class imbalance problem for SVMs.

This section briefly defines the algorithm and its main properties. It also investigates how SVM can be applied to imbalanced datasets.

Support vector machines

SVMs are forms of classification algorithms, which are supervised learning methods that analyse data and discover patterns; they can be used for classification and regression in data mining. They are a blend of linear modelling and instance-based learning, where a small number of critical boundary instances called support vectors are selected from each class to build a linear discriminant function that separates data as widely as possible (Witten and Frank, 2005). Moreover, SVMs perform non-linear classification, in which the method maps data into a higher dimensional space non-linearly in order for it to be classified by a kernel function.

The model makes predictions using kernel basis methods, in which the best choice will be the hyper plane that leaves the maximum margin from both classes (Wang et al. (2003) and Witten and Frank (2005)). The accuracy of the model relies on the kernel's parameters. There are three widely used types of kernels: polynomial, sigmoid, and radial basis kernel function (RBF) (Huang and Wang (2006) and Lin et al. (2008)), Eqs. 2.1, 2.2 and 2.3. The RBF kernel, Eq 2.3, has been proved to be a good choice for learning from imbalanced datasets as it can be applied to high dimensional datasets and has only two parameters to tune (Hsu, Chang, and Lin (2003), Ding and Chen (2010), and Cao, Zhao, and Zaiane (2013)). The first parameter is the misclassification cost or the penalty parameter, C. The choice of the value of C affects the classification model. A high value of C will lead to a low bias and high variance model (over-fitting model) while too small a value of C will create a high bias and low variance model. The second important parameter within the kernel function is the gamma, γ that defines the spread of the kernel. If the value of γ is too small, this will produce a low bias and high variance model and vice versa (Sudheer et al., 2014).

This approach of optimising the kernel's parameters has been shown to perform well when dealing with imbalanced datasets (Hsu, Chang, and Lin (2003), Lin and Lin (2003), and Lin et al. (2008)). One way to find the best values for these parameters when using the RBF kernel for SVM is *grid search*, which is a widely used method whereby the search is performed by trying all possible combinations within a list of values for each parameter. In this interval or list of values, the best values for C and γ can be found, which gives the highest classification accuracy. However, this search is time consuming. It also requires setting feasible range within which to search and a suitable sampling step (Hsu and Lin (2002) and Ding and Chen (2010)).

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i.\mathbf{x}_j)^d \tag{2.1}$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(k\mathbf{x}_i.\mathbf{x}_j\delta) \tag{2.2}$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}} \tag{2.3}$$

Another way to optimise the kernel's parameters is to use meta-heuristic approaches such as swarm intelligence or evolutionary computation-based approaches. These meta-heuristic models have garnered an increased level of interest for solving common search and optimisation problems (Nasuto and Bishop (1999) and Huang and Wang (2006)). These techniques often mimic the behaviours of social insects like bees, wasps and ants to offer powerful tools to solve problems. The emergence of collective intelligence, which is the core of such techniques, lies in a network of interactions among social insects and their environment; this process is described as a self-organised and decentralised system of collective behaviour (Beni and Wang, 1993).

Various swarm intelligence techniques and population-based techniques have been used to optimise the SVM kernel parameters, among them Genetic Algorithm (GA), PSO, and Ant Colony Optimisation (ACO). GA is a meta-heuristic algorithm based on Darwin's evolution of biological systems and developed in 1970s by John Holland (Goldberg and Holland, 1988). The algorithm uses three main operators: crossover, mutation and selection. It has been widely used to generate solutions to various search and optimisation problems. In terms of data mining, GA has been widely used to search for the optimal SVM configurations for best prediction performance when learning from imbalanced datasets. Di Martino et al., 2011 proposed a model for predicting fault prone components in which GA was used to configure the SVM parameters for optimal performance on an imbalanced dataset. Experimental results indicate that the proposed approach gives better performance, increasing the model's sensitivity without decreasing the precision rate. Li and Kong, 2014 proposed a GA-based model called GA-SVM for landslide displacement rate prediction. GA is used to optimise the kernel parameters, C and γ. The proposed model gives the smallest root mean square error equal to 0.0009. In other research undertaken by Wei and Hui-Mei, 2014, an improved version of GA was used to optimise the kernel parameters, C, γ as well as the loss function parameter, ϵ, was proposed. Results indicate the superior performance of the proposed approach.

PSO is another meta-heuristic based search optimisation algorithm, in which optimisation is carried out using simple agents called particles (Kennedy and Mendes, 2002). PSO has been widely applied in model selection in machine learning. An example of a PSO-based approach is seen in Lin et al., 2008, where the authors proposed a parameters optimisation and feature selection approach in which the RBF kernel parameters, C and γ are optimised and the optimal feature subset is selected. The proposed approach performed well on several datasets when compared with grid search. Another approach by Guo et al., 2008, in which a parameter's selection method was proposed for Least-Squares Support Vector Machines (LS-SVMs) using PSO. The proposed method performed well on benchmark datasets. The results also show that the best performance was obtained by using the Scaling Radial Basis kernel Function (SRBF) and RBF kernel functions. Wang et al., 2014 combined PSO and SMOTE on three classifiers: decision tree, logistic regression (LR), and k-NN. To evaluate the proposed approach, G-mean and predictive accuracy were used. Experimental results show that SMOTE + PSO improves the classifier performance in breast cancer prediction. Subbulakshmi and Deepa, 2015 proposed a PSO-SVM model to improve the classification accuracy on the Electromyography (EMG) signal. The results show that PSO-SVM accuracy is equal to 97.41% as opposed to 96.75% for SVM. Therefore, the proposed approach PSO-SVM can be used for diagnosis of neuromuscular disorders.

In another investigation, Melgani and Bazi, 2008 proposed a novel model that uses PSO to improve the classification accuracy and the generalisation ability of SVM on the electrocardiogram (ECG) [3] beats dataset. The experimental results indicate the superiority of the proposed approach against

[3] Electrocardiogram is a test to detect heart rhythm abnormalities.

three different classifiers: SVM, *k*-NN, and the RBF neural network. Pan and Luo, 2010 proposed a parameter selection model for SVM based on PSO. The results indicate that the proposed approach improves the SVM model prediction and generalisation capabilities.

In the literature, it has been found that PSO has been widely used as a way to tune SVM parameters to overcome the class imbalance problem and avoid over-fitting by improving the model's generalisation capabilities. The algorithm gives good results, as seen in previous studies on SVM parameters tuning (Bao, Hu, and Xiong, 2013). Due to its search and optimisation capabilities, PSO has also been used to simultaneously optimise the SVM kernel parameters and perform feature selection (Huang and Dun (2008) and Lin et al. (2008)).

ACO, which mimics the behaviour of ants to search for solutions in a optimisation problem (Dorigo, Birattari, and Stutzle, 2006) has also been applied to optimise the SVM parameters. Zhang, Chen, and He, 2010 proposed a model that is based on ACO to tune SVM parameters. The proposed model provided promising results when applied to real-world benchmark datasets. In another study by Alwan and Ku-Mahamud, 2012a, a model to tune the SVM parameters using incremental continuous ACO was proposed. The proposed approach performed well on eight datasets from the UCI machine learning repository. In another study conducted by Alwan and Ku-Mahamud, 2012b, the authors proposed an algorithm for optimising the SVM using continuous ACO. This was applied without discretising continuous values for the SVM parameters as it has been found that the discretisation process has been shown to causes loss of some values, which leads to lower classification accuracy. The experimental results indicate that the proposed approach outperforms grid search on seven datasets from the UCI machine learning repository. Fang and Bai, 2009 proposed a model to predict share prices. The proposed approach was based on ACO to optimise SVM parameters. Here ACO was integrated with wavelet transform to remove any fluctuant from the dataset. The hybrid approach performed well in predicting the price of Huaneng Guoji shares. Another ACO based search approach was introduced by Zhang et al., 2008 to optimise the SVM kernel's parameters. The results indicate that the proposed approach provides an easy and efficient solution for parameters tuning.

In the literature, the use of ACO to optimise the SVM has produced promising results. Like PSO, ACO has also been used to simultaneously optimise

SVM parameters and find a suitable feature subset (Huang, 2009) (more details can be found in Section 2.4). Despite, the recent development in the use of meta-heuristic techniques to optimise SVM parameter, GA and PSO are the most commonly used, while other methods that can be used to optimise SVM are still at the early stage of experiment. As a result, there are other population-based algorithms with efficient optimisation capabilities, such as SDS and DFO, which can be used to optimise the SVM kernel parameters. Thus, further work is needed to develop the use of other meta-heuristic algorithms to improve SVM performance on imbalanced datasets. For example, one could combine various meta-heuristic based parameters tuning approaches with different sampling methods available from the literature to improve the model performance and take the best of both when learning from imbalanced datasets. The next section discusses how a hybrid approach, namely, a combination of both data level and algorithmic level solutions, can perform well on imbalanced datasets.

Hybrid approach

The process of customising a machine learning algorithm or data sampling technique to address the class imbalance problem varies from case to case and further depends on the difference of the data ratio between the classes. Another solution to address this problem is a hybrid approach that either comprises both oversampling and undersampling techniques (Singh, 2016), or a combination of data level and algorithm level solutions to minimise the effect of class imbalance (Burez and Poel (2009) and Kotsiantis, Kanellopoulos, and Pintelas (2006)). The compendium of research described herein suggests that an iterative way of applying this combination can improve classification by deploying the algorithm after every undersampling and oversampling of the dataset. Some of the earliest work on the utilisation of hybrid approaches has been documented in (Ling and Li, 1998), whereby a *combined approach* was suggested. This applies the undersampling to classes consisting of a larger number of instances and the oversampling to classes consisting of a smaller number of instances. Depending on the importance of the minority sampled data, oversampling of the minority class and undersampling of the majority class can be used, but only if it is effective. The reason is that if the minority class data can be ignored because it presents anomalies, then there is no need to oversample it or undersample the majority class.

SMOTE is one useful approach to oversample the minority class. This is to balance the dataset (Chawla et al., 2002). A study by Batista, Prati, and Monard, 2004 suggests that a hybrid solution that is a combination of SMOTE with other techniques like Extended Nearest Neighbour (ENN) has performed well with highly imbalanced datasets that have a small number of minority class instances. Moreover, in applications like fraud detection, Padmaja et al., 2007 proposed a hybrid approach that uses extreme outlier elimination and a hybrid sampling approach in which a combination of SMOTE to oversample the minority class and random undersampling to decrease the number of instances in the majority class is used. The results obtained from the hybrid approach improved classification accuracy. Rahman and Davis, 2013 proposed a model that uses both SMOTE and a new cluster-based undersampling approach to improve the training sets for a better classifier performance on the cardiovascular dataset. The experimental results indicate that the proposed approach gives a better performance when compared with other undersampling approaches. In terms of SVM, Tang and Zhang, 2006 proposed a model for imbalanced datasets that is a combination of Granular SVM and Random Undersampling (GSVM-RU). The proposed approach was one of the best models in the ACM KDD cup 2004 competition [4] on the protein homology imbalanced dataset. Gao et al., 2011 combined SMOTE and PSO to determine the RBF kernel parameters for SVM, in which the aim was to minimise the leave-one-out misclassification rate. The proposed approach has been applied to one simulated imbalanced dataset and three real-world imbalanced datasets. The results indicate that the proposed approach performs well when compared with other state-of-the-art models such as SMOTE+ *k*-NN.

Similarly, the application of algorithms to adjust the cost for each class or for the class attributes in training data after undersampling or oversampling can improve the performance on imbalanced datasets (Tang et al., 2009). This can be clarified with an example of a dataset containing 1300 samples in the training set where the majority class has 1000 samples, while the minority class has 300 samples. In the training set, it is important to keep the balance of the cost assigned to each class because applying more cost to the minority class can also result in inaccurate classification when applied to the real data. If we oversample the minority class using techniques like SMOTE, and if, after each oversampling, we adjust the cost and evaluate the classification, this will give us a robust balance point where the minority class is given its due inclusion in the final results. Another hybrid solution which can be applied to address the problem of class imbalance is that once the data has

[4] An annual knowledge discovery competition: http://www.kdd.org/kdd-cup.

been classified and has an imbalance, then the algorithm can be applied separately to the majority and minority classes. When these two classes are run through the algorithm separately, there will be further classification of the data embedded within each class. Subsequently, the data samples from each class that should be considered are automatically filtered. This filtering can then be input into the undersampling and oversampling techniques so that redundant data is discarded based on the feedback of the algorithm. The technique can be improved by selecting random datasets for training the algorithm and running the iterations to determine if the data being discarded or added is aligned with the complete dataset. It will remove the close dependency on the selected training set and explore the data in different randomly selected training sets to increase the chances of accurate classification, which is based on using inbound and outbound loops of algorithm and sampling techniques.

In summary, there is no unified and general answer regarding the best method to apply to imbalanced datasets. Weiss, McCarthy, and Zabar, 2007 compared cost sensitive learning and sampling to find which is better at handling class imbalance. To deal with this issue, the authors used three models on fourteen datasets with different minority class percentages. The first model adjusted the misclassification cost in the learning algorithm and the other two models incorporated oversampling and undersampling to balance the dataset. It was found that the choice of method depended on the dataset in terms of factors, such as the number of instances and the degree of class imbalance; for example, in datasets with $10,000$ instances, cost sensitive learning outperformed sampling. Maloof, 2003 also compared cost-sensitive learning with sampling on one dataset and found that they all performed nearly the same, but this assumption cannot be generalised because only cases of class distribution were compared. In another study, where the class imbalance problem was investigated from a probabilistic point of view, it was found that undersampling works better in high dimensional datasets (Wallace et al., 2011). An investigation of the impact of the class imbalance ratio on the classifier in adverse drug events prediction was conducted by Taft et al., 2009. The authors compared various sampling methods, including random undersampling, SMOTE, Gabriel graph [5] based SMOTE (gg-SMOTE)

[5] Gabriel graph is an proximity graph introduced in 1969 (Gabriel and Sokal, 1969). It has been used in various real-world applications such as geographical variation analysis and cluster analysis (more details about the proximity graph can be found in (Sánchez, Pla, and Ferri (1997) and Hossain et al. (2015))).

and Wilson's editing [6] with Modified Selective Subset (MSS) condensing algorithm over the negative instances as undersampling approaches. The results show that oversampling was better than undersampling in highly imbalanced datasets. This is because undersampling causes loss of information. However, when the imbalance ratio is low, there is no difference in the performance between undersampling and oversampling. In another study by Batista, Prati, and Monard, 2004, the authors found that the combination of SMOTE with an informed undersampling method such as Tomek links, gives better results for smaller datasets. However, when dealing with datasets that contain a higher number of minority examples, random oversampling gives better results.

As a result, it is still unclear which method works better for a given imbalanced dataset with unequal misclassification cost. Researchers have proposed various models to overcome the class imbalance problem from different angles. However, there is still no clear final winner out of undersampling, oversampling, and cost sensitive learning, as they all have advantages and disadvantages. Thus, further research is needed to identify what works better for a given dataset and analyse the influence of factors like dataset size on the choice of solution for the class imbalance problem. The next section explores feature selection, defines its main approaches, and discusses how it can be used to address the class imbalance problem.

2.4 Feature selection

In addition to the data and algorithmic level solutions, feature selection has been used as a solution to the class imbalance problem at the feature level (Wasikowski and Chen (2010) and Chawla, Japkowicz, and Kotcz (2004)). This is because most high dimensional datasets are imbalanced when it comes to class distribution (Maldonado, Weber, and Famili, 2014). Examples for high dimensional and imbalanced datasets include: microarray dataset (Xing, Jordan, and Karp, 2001), text categorisation (Forman, 2003) and bioinformatics (Saeys, Inza, and Larrañaga, 2007). When learning from these types of datasets, data level and algorithmic level solutions are not always sufficient to improve the classifier performance. Therefore, more work is required at the feature level. Numerous investigations have used feature selection

[6] Wilson editing is based on applying k-NN to predict the class label of all instances in the training sample and remove all samples whose class label is not the same as the class label of the largest number of the k neighbours.

and combined it with other data mining techniques to increase the classifier's generalisation ability and improve overall performance on high dimensional and imbalanced datasets (Chen and Wasikowski (2008), Chawla (2009), Wasikowski and Chen (2010), and Al-Shahib, Breitling, and Gilbert (2005)). Al-Shahib, Breitling, and Gilbert, 2005 proposed a hybrid approach that uses feature selection and undersampling to improve SVM performance in predicting protein function from sequence. The results indicate that the combined approach provides promising solutions. Maldonado, Weber, and Famili, 2014 proposed a backward elimination approach based on successive holdout steps. Six highly imbalanced microarray datasets are used to evaluate the proposed approach. A results comparison has shown that the proposed method outperforms other well-known methods with smaller feature subset size. In another approach by Han et al., 2016, two online feature selection algorithms are proposed using the Passive-Aggressive (PA) algorithm and a Truncated Gradient (TG) method to improve the classifier performance on several high dimensional imbalanced datasets. In a study by Zheng, Wu, and Srihari, 2004, a feature selection method is proposed for text categorisation imbalanced datasets using multinomial NB and regularised LR as classifiers. More details on other methods for feature selection on imbalanced dataset can be found in (Mladenic and Grobelnik (1999), Forman (2003), and Shang et al. (2007)).

The next section provides a definition for feature selection and briefly describes its main approaches and challenges.

Feature selection definition

Today's datasets are massive, and they usually possess the characteristics of high dimensionality. However, not all features or attributes are of equal value. Indeed, a certain percentage of features are either repetitive, irrelevant, or both. Taking this into account, feature selection is defined as "*a process that selects a small subset of the relevant features*" (Tran et al., 2016). It is an important technique for removing the redundant or irrelevant features to increase the speed of data mining algorithms, reduce data storage space, improve predictive accuracy, ease interpretation for researchers, and reduce the problem of over-fitting. Feature selection differs from feature extraction in that it requires the selection of the optimal feature subset through the deletion or removal of redundant features. However, feature extraction involves constructing new features out of the original features to reduce the dataset's dimensionality. Essentially, feature selection searches for the most relevant features for classifications within the dataset as opposed to deriving new features. Examples for feature selection techniques include chi-square, information gain, and mutual information.

Nowadays, feature selection methods are used in various fields such as information retrieval and filtering, text classification, risk management, web categorisation, medical diagnosis, and detection of credit card fraud, all of which are subject to high dimensionality and class imbalance problems (Jamali, Bazmara, and Jafari (2012) and Chandrashekar and Sahin (2014)). They are a vital step for a number of learning algorithms, particularly for high dimensional datasets such as microarray based classification and text classification (Carvajal et al., 2004). Feature selection strategies include three main approaches: filter, wrapper and hybrid. In the filter approach, feature subset selection is dependent on selected criteria to evaluate the feature's importance for class label classification. However, the wrapper approach is dependent on the classification algorithm to evaluate the feature subset that gives the best prediction performance. It is more computationally expensive when compared to the filter approach. The hybrid approach combines the advantages of both the filter and wrapper models. The approach applies an evaluation criterion and a data mining algorithm to search for a feature subset.

Section 2.4.1 sets out feature selection's main challenges. Section 2.4.2 describes the three main feature selection approaches.

2.4.1 Feature selection challenges

In real-world situations, the relevant features of the data are often not known in advance. This increases the complexity of feature selection. Additional issues or challenges exist with regard to the following:

- Complex interactions among features.
- Deciding which data and features are redundant versus relevant.
- An important feature may become irrelevant when combined with other features in the dataset (and vice versa).
- The large search space makes it nearly impossible to perform an exhaustive search within the exponential search space.

Thus, feature selection is not an easy preprocessing technique. As such, a growing number of advanced feature selection methods using swarm intelligence techniques are being tested. Currently, they combine traditionally used data mining preprocessing techniques with meta-heuristic techniques such as PSO. Accordingly, feature selection methods fall into three broad categories: filter, wrapper, and hybrid methods (Raymer et al. (2000) and Saeys, Inza, and Larrañaga (2007)).

2.4.2 Feature selection approaches

There are two primary objectives when assessing a feature selection algorithm: optimal classification performance and minimisation of the number of features when dealing with high dimensional datasets. In addition to the different sub-goals of each method, the filter and wrapper techniques have their own evaluation criteria (Liu and Yu, 2005). There are clear advantages and disadvantages to both, which are discussed further below. In the next section, the filter method will be presented, followed by the wrapper method and then the hybrid method.

Filter Approach

Filter methods are traditionally used at the preprocessing step of feature selection. They rely on the general characteristics of the training data's selected features, as shown in Fig 2.6. The feature subsets are weighted – or scored – and then ranked using a statistical measure. Those with top scores (high rank) are kept and those with lower scores are removed or discarded. Scoring measures include Z-score, Bayesian test, information gain, and mutual information. The ranking method is used to evaluate the feature relevance. This is to evaluate how the feature is useful in discriminating the various classes in the dataset.

The benefit of filtering is that it is fast, computationally simple to perform and easily scalable for high dimensional datasets (Wu et al., 2017). Filtering is also reported to often ignore the interaction with the classifier and therefore tends to ignore feature dependencies, which when contrasted with other methods can lead to worse classification performance (Liu and Yu, 2005). One major drawback in filtering is that the selected feature is not always optimal as it may include redundant and highly correlated features. Many filter techniques have been proposed to overcome the correlation problem, among them Pearson correlation.

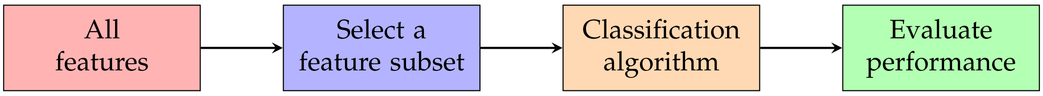

FIGURE 2.6: Filter method

Many feature selection methods are proposed in the literature. An example is the use of mutual information to measure relevant features and whether there is any redundancy. In Mutual Information based Feature Selection (MIFS), features are ranked or evaluated based on their mutual information with regards to the class label. Battiti, 1994 introduced a mutual information based feature selection approach that considers both the mutual information with regards to the class label and with the already-selected features in the feature subset, known as MIFS-U. The method was proposed to overcome a limitation of MIFS, which is that it does not show the relationship between the feature and the class label in its redundancy term. Another variant of MIFS is Normalised Mutual Information Feature election (NMIFS). The method calculates the normalised mutual information to measure the redundancy (Estevez et al., 2009).

Another filter method is information gain (IG). IG measures the total bits of information gained for class prediction by knowing the value of a feature. In particular, it calculates the expected decrease in entropy whenever a certain feature is available, whereby important features cause greater decrease when calculating the entropy (Mitchell, 1997).

$$IG(F) = E(S) - \sum_{n} \frac{S_n}{S} E(S_n) \tag{2.4}$$

where E(S) is the entropy of the given dataset and E(S_n) is the entropy of the n^{th} subset generated by partitioning the dataset based on feature F.

Researchers have developed other ways for feature selection to overcome the shortcomings of standard feature selection methods. Kira and Rendell, 1992 proposed a feature selection approach for binary classification known as Relief feature selection. The approach is efficient, noise tolerant and sensitive to feature interactions, making it applicable to real-world applications such as protein folding. It is a statistical method that calculates each feature score to rank the features. Based on the ranking the top scoring features are chosen for feature selection. Later, Kononenko, 1994 proposed an extension of the Relief algorithm that handles multi-classification and datasets with missing values. The method is called ReliefF. Another approach was developed by Xie et al., 2018, who proposed a feature selection method based on Relief that can be extended and used on unbalanced multi-label datasets. The method is called a UBML-ReliefF algorithm. Experimental results show that the new UBML-ReliefF outperformed the original ReliefF algorithm. Tiwari, 2014 proposed a modified version of Relief which gives higher weight to attributes when dealing with minority classes. The results indicate that the proposed method gives better results when compared to the original Relief algorithm. Peng, Long, and Ding, 2005 proposed a correlation-based feature selection approach called minimum-redundancy-maximum-relevance (mRMR). The algorithm provides a balance between relevancy and redundancy (Nguyen, Franke, and Petrovic, 2010). Another approach for feature selection is called the Maximum Relevance–minimum Multi-Collinearity (MRmMC) method, in which both correlation and redundancy are evaluated and eliminated. The proposed approach performed well on eight datasets from the UCI machine learning repository (Senawi, Wei, and Billings, 2017). Another example is the FOCUS algorithm, in which all possible feature subsets are examined. One major drawback of this algorithm is that it is computationally expensive (Almuallim and Dietterich, 1994).

The next section looks at the second approach for feature selection, which is the wrapper approach. Unlike the filter approach which uses feature ranking criteria, wrapper methods rely on the classification algorithm to find the optimal feature subset.

Wrapper Approach

In terms of the wrapper approach, feature selection makes use of the result of the classification algorithm to determine the strength and usefulness of any given subset. The method involves initiating a search procedure, whereby various dataset attributes are defined, generated and evaluated. In the wrapper approach, the process of feature selection can be generalised into four steps, the first two of which are repeated until the third stopping criterion (3) is achieved:

1. Subset generation: a search strategy that derives the potential feature subsets.

2. Subset evaluation: comparative analysis of the potential subsets.

3. Stopping criterion: determines when the search is to be stopped, of which there are several methods; the search self-completes, a given

bound is reached, additional feature deletions or additions do not improve results, a sufficiently good subset is selected (e.g., based on a below threshold error rate).

4. Result validation: this largely depends on the scenario; if there is a predetermined measurement of best fit, such as having knowledge of relevant features beforehand, then results can be validated in comparison to this already known set of features. However, if such knowledge is not available, then the error rate method can be applied.

Although the approach takes feature dependency into account, one common problem is that it has a higher risk of over-fitting than the filter techniques. In addition, its performance tends to be slow and computationally expensive (Jun-shan, Wei, and Yan, 2009). Additionally, given the dual computation occurring, larger datasets slow the algorithm's performance. However, when coupled with classification algorithms such as SVM or *k*-NN and swarm intelligence algorithms (e.g., GA), increased classification accuracy has been reported within the realm of gene classification research (Alba et al., 2007).

Numerous wrapper approaches have been proposed in the literature. An example of two widely used methods are Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS). In SFS, the process starts with no features at all, then features are added. This is repeated until there is no increase in the classifier performance. However, in SBS the process starts with all features, and then features are removed in sequence. The removal is stopped when the increase in classifier performance stops. One major drawback of these two approaches is the nesting effect (Kabir, Islam, and Murase, 2010). The nesting effect problem can be solved by combining SFS and SBS into one method, such as the Sequential Forward Floating Selection (SFFS) and Sequential Backward Floating Selection (SBFS) (Reunanen, 2003).

Various search algorithms can be used to find the feature subset and maximise the classification performance most use meta-heuristic techniques to speed up the search (Blanco et al. (2004) and Jirapech-Umpai and Aitken (2005)) and few use sequential search approaches (Xiong, Fang, and Zhao, 2001). Many others have used a hybrid filter-wrapper approach to take advantage of both (Sebban and Nock (2002) and Bermejo, Gámez, and Puerta (2011)).

The following section provides examples from the literature on various hybrid approaches.

Hybrid Approach

Various search techniques have been proposed to speed up the generation and evaluation of feature subsets, such as combining a filter and wrapper approach and using meta-heuristic methods to find the solution in a cost-effective way (Xue et al., 2016). Many researchers have proposed a hybrid filter-wrapper approach to perform feature selection. More details about various hybrid approaches can be found in (Sebban and Nock (2002) and Jashki et al. (2009)).

Swarm intelligence and population-based algorithms have been widely used as a solution for the feature subset selection problem. An example of a population-based technique which has been used as a solution for the feature subset selection problem is GA (Punch III et al., 1993). Leardi, 2000 proposed a GA feature selection in the spectral datasets. In another study by Oh, Lee, and Moon, 2004, a novel hybrid GA for feature selection was proposed. The results indicate that the proposed hybrid GA gives better results than the classic GA and the sequential search algorithms.

In terms of PSO, a model was proposed by Zahran and Kanan, 2009, which uses PSO for text feature selection. It reduces the dimensionality of the dataset in order to improve text categorisation efficiency. The proposed model has been evaluated against others including chi-square and document frequency. The results illustrate the superiority of the proposed model on the Arabic dataset. Two multi-objective PSO algorithms for feature selection were proposed by Xue, Zhang, and Browne, 2013. In the first algorithm, the concept of nondominated sorting was used with PSO to solve the feature selection problem. The second algorithm applies crowding, mutation, and dominance to search for the optimal pareto front solution using PSO. Both algorithms outperform three other well known multi-objective algorithms. In another study, a hybrid approach that uses both PSO and SVM to improve the classification task (PSO-SVM) was proposed (Huang and Dun, 2008). The algorithm simultaneously optimises the features subset and SVM kernel parameters by using a distributed architecture to reduce the computational cost. The results show that the proposed method can find the optimal feature subset and improve the classification accuracy. Moreover, a gene selection method was developed using two hybrid techniques: PSO and GA (Alba et al., 2007). Both methods performed well when compared with other meta-heuristic algorithms from the literature on six datasets.

In terms of ACO, Aghdam and Kabiri, 2016 introduced an ACO based feature selection approach to reduce dimensionality in the intrusion detection system. The proposed approach is simple and computationally low. Moreover, it outperforms other techniques on the KDD Cup 99 and NSL-KDD intrusion detection benchmark datasets [7]. Huang, 2009 proposed a hybrid ACO-SVM approach to simultaneously find a feature subset and optimise the SVM kernel parameters. The experimental results indicate that the proposed approach gives higher classification accuracy and reduces the feature size. Another model proposed by Kanan and Faez, 2008, uses ACO to perform feature selection for a face recognition system. The proposed method can easily be implemented and does not require prior information about the features, as opposed to standard ACO based methods. Ding, 2009 proposed an ACO based model to perform feature selection and parameter optimisation. The proposed approach uses F-measure to simultaneously perform feature selection by removing irrelevant features and optimise the C and γ. Experimental results show both the feasibility and efficiency of the ACO based model.

In short, feature selection comes in many different forms, each resulting in a measurably different outcome. Research regarding the variety of feature selection methods increases accuracy in establishing which metrics perform best on different datasets. Currently, there is a lack of expert systems that can successfully reduce both training and testing times. Therefore, it is important for researchers to dedicate their time to testing and identifying the most effective feature selection methods that can be applied to each dataset.

The next section illustrates the various metrics available for evaluating the model when learning from imbalanced datasets.

2.5 Metrics to evaluate models on imbalanced datasets

The performance of the learning model is typically evaluated using predictive accuracy (Acc), see Eq 2.5. However, this is not suitable when the dataset suffers from class imbalance or the costs of errors vary. An example is the classification of microscopic biopsy images (Kumar, Srivastava, and Srivastava, 2015). Acc, as shown in Eq 2.5, is not an effective performance metric for models on imbalanced datasets as it does not show how the model correctly classified the minority class instances.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FN} + \text{FP}} \tag{2.5}$$

[7] The NSL-KDD is a dataset available at http://www.unb.ca/cic/datasets/nsl.html.

TABLE 2.2: Confusion matrix

	Predicted Positive Class	**Predicted Negative Class**
Actual Positive Class	True Positives (TP)	False Negatives (FN)
Actual Negative Class	False Positives (FP)	True Negatives (TN)

Alternative performance metrics are sensitivity and specificity (Han, Pei, and Kamber (2011) and Tang et al. (2009)). Sensitivity, as shown in Eq 2.6, also known as the True Positive Rate (TPR) or *recall rate*, refers to the percentage of positive examples correctly predicted as positive class examples.

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.6}$$

Specificity, as shown in Eq 2.7, also known as the True Negative Rate (TNR), refers to the percentage of negative examples that are correctly predicted negative class examples.

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{2.7}$$

A combination of sensitivity and specificity is another metric which can also be used for imbalanced dataset. The metrics are called F-measure and the geometric mean (G-mean) of sensitivity and specificity. The F-measure is calculated using the harmonic mean of precision and sensitivity as shown in Eq 2.8. It is a commonly used metric on multi-class imbalanced datasets (Guo and Viktor, 2004). G-mean shows the balance between the classifier performance on both the majority class and the minority class, and it considers both the sensitivity and the specificity, as shown in Eq 2.10. Both metrics have been widely used by researchers to evaluate the classifier performance over imbalanced datasets, more details can be found in (Su and Hsiao (2007) and Zhang and Wang (2013)).

$$\text{F-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.8}$$

where precision is the positive predictive value which indicates the accuracy of the model's recall. Precision can be calculated using the following equation:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.9}$$

$$\text{G-mean} = \sqrt{\text{Sensitivity} \times \text{Specificity}} \tag{2.10}$$

Additionally, for models comparison, Receiver Operating Characteristics (ROC) rate and Area Under the ROC Curve (AUC) can be used where the False Positive (FP) rate on the x-axis and True Positive (TP) rate on the y-axis and each classifier is represented by a point on the curve. Both metrics are widely used to evaluate classification models performance (Weiss, 2004). The ROC is a visual tool to evaluate the classifier performance over a range of trade-offs between true positive and false positive error rates, in which the perfect point on the ROC curve would be (0,100). This point indicates that all the positive examples have been classified correctly and there are no negative examples classified as positives (Chawla et al. (2002), Alpaydin (2014), and Maimon and Rokach (2010)). AUC is a single metric derived from the ROC curve to summarise classifier performance, and the larger the AUC, the better the classifier performance when learning from the dataset.

Therefore, the choice of performance measurements depends on the problem and the class distribution. It has been suggested that a combination of measurements gives a balanced evaluation of the model's performance (Sokolova, Japkowicz, and Szpakowicz, 2006).

2.6 Summary

Through the analysis of all available results, it is evident that data mining is a valuable tool for extracting useful information hidden within data. It can be implemented in numerous real-world applications to support various decision making processes. There are a variety of classification methods with demonstrable evidence regarding how well they perform in certain cases where datasets suffer from class imbalance such as customer response prediction, medical diagnosis and fraud detection. The background research and the literature review presented in this chapter aimed to summarise related work and provided a critical discussion of the topic. This investigation helped to engender a deeper understanding about the results of research conducted within the field of data mining and the problem of class imbalance.

The review also highlighted an important problem with data mining algorithms when classifying imbalanced datasets: that these algorithms can be unintentionally biased towards the majority class, which can increase the likelihood of inaccurate classification. Also discussed were in-depth solutions related to the issue of class imbalance that used SVM as a learning algorithm, which assists in constructing the thesis framework and methodology. The solutions include sampling, cost sensitive learning and feature selection.

Included within this examination was the description of a link between the problem of class imbalance, the type of data, and its dependence on the instances included in a minority samples class. The importance of the instances in a minority samples class and its identification before selecting a technique to solve the problem should not be underestimated as it varies from problem to problem. In other words, the criticality of the minority class in one dataset cannot be compared to another dataset and from one application to another, therefore a sampling technique can be expected to vary from one classification model to another with a similar number of instances in the majority and minority class. Moreover, assigning more cost or weight to the minority class does not always solve the problem because it cannot be generically applied to different datasets or even different training sets of the same dataset. This is due to each classification attempt on a different training set resulting in a different number of instances in each class, even if the training sets are taken from the same dataset. Therefore, cost sensitive learning can be combined with other techniques to tackle various aspects of the class imbalance problem.

Another link was the use of feature selection as a tool to overcome the class imbalance problem, as most high dimensional datasets have skewed distribution. The review provided an insight into the use of swarm intelligence techniques and population-based algorithms such as GA, PSO, and ACO to resolve issues related to class imbalance. This includes the use of swarm based techniques to balance the datasets, applying a cost sensitive solution by optimising the kernel parameters, C and γ, searching for a feature subset when dealing with high dimensional imbalanced datasets.

The next chapter of this thesis, Chapter 3, addresses significant aspect for data mining: the role of swarm intelligence and population-based algorithms in solving the class imbalance issue. It provides an overview of swarm intelligence and evolutionary computation, and their advantages and disadvantages. This is followed by a definition of well-known algorithms and a brief introduction to the role of swarm intelligence in data mining through a review of how swarm intelligence algorithms have been used to overcome the class imbalance problem. Chapter 4 presents the set of experiments performed on real-world datasets to evaluate the performance of various models, the objective of the experiments being to answer the research questions raised in this thesis.

Chapter 3

Swarm Intelligence

This chapter presents a background for swarm intelligence and evolutionary computations and provides definitions of a number of terms, followed by a discussion of the main advantages and limitations of swarm intelligence. Several population-based algorithms are then introduced, and their main components and behaviours are highlighted. These algorithms include GA, SDS, ACO, PSO, DE, and DFO. This is followed by a review of the work done by earlier researchers on the use of various population-based algorithms in data mining. The aim of this chapter is to shed light on how these algorithms can be used to solve various search and optimisation issues in data mining, mainly the class imbalance problem.

3.1 Background

Swarm intelligence and evolutionary computation are both artificial intelligence based approaches and have been introduced as optimisation methodologies. They can produce solutions to various search and optimisation problems using a population-based approach. Swarm intelligence is inspired by the collective behaviour of swarms such as ants and bird flocks. However, evolutionary computation is inspired by the biological evolution in nature and is based on the processes of Darwinian evolution (Fogel, 1995). Generally speaking, evolutionary computation algorithms include but are not limited to GA, GP, and DE. Swarm intelligence algorithms include but are not limited to ACO and PSO.

Although the algorithms have different sources of inspiration and use different names for their operations, they still share many characteristics (Eberhart and Shi, 1998). For example, both use fitness functions and population to initialise. A study by Yang, 2014, found that the random route generation in the ACO is analogous to mutation in evolutionary computation. However, there is no specific crossover in the ACO algorithm. In this thesis, swarm intelligence is considered to include both swarm intelligence algorithms and evolutionary computation algorithms. According to Grosan, Abraham, and Chis, 2006, a swarm is defined as *"a large number of homogeneous, simple agents interacting locally among themselves, and their environment, with no central control to allow a global interesting behaviour to emerge"*. In particular, swarm is applied to entities or animals that exhibit a collective behaviour. The swarming behaviour was first simulated by Reynolds, 1987 using the Boids program. The model was first used to simulate the behaviour of flocks of birds. However, it can be applied to simulate other swarming behaviours like schools of fish. Swarm intelligence is based on boids or simple agents who interact locally with each other and with the environment, leading to intelligent collective behaviour. Moreover, it refers to *"a relatively new branch of Artificial Intelligence that is used to model the collective behaviour of social swarms in nature, such as ant colonies, honey bees, and bird flocks"* (Grosan, Abraham, and Chis, 2006). Another definition of swarm intelligence by Bonabeau, Dorigo, and Theraulaz, 1999 further define swarm intelligence as *"the emergent collective intelligence of groups of simple agents"*. Even though agents, such as swarm individuals or insects, are with very little intelligence by themselves and with very few individual capabilities, they do interact with each other through particular behaviours to achieve tasks, which are vital for their survival. Recently, swarm-based algorithms have emerged. They are both nature and population driven algorithms and have the ability to produce a series of robust, fast and low-cost solutions to a number of complex issues (Abdelbar, Ragab, and Mitri (2003) and Abonyi, Feil, and Abraham (2005)). Furthermore, research by Abraham and Ramos, 2003 has demonstrated that a swarm individual's social activity (interactions) can be either direct or indirect. For example, visual or audio contact can be considered direct interaction (e.g., honey bees' waggle dance), whereas indirect interactions arise from the changes that an individual makes to their surrounding environment. This creates a new environment to which other individuals react (e.g., pheromone trails that ants deposit when searching for food); this is known as *stigmergy*. Stigmergy is defined by Grassé as *"indirect communication via interaction with the environment"* (Marsh and Onof, 2008). It simply creates a means of communication through the environment, which affects the behaviour of other members of the population or swarm.

Within the past twenty years, both natural scientists and biologists have investigated insects' social behaviours due to the incredible efficiency these natural swarm systems exhibit. Their efficiency stems from their information sharing and decentralised behaviour in identifying solutions. Computer scientists wanted to introduce the scientific phenomenon of natural swarm systems to artificial intelligence. Thus, there has been a rapid increase in the development of new or novel, nature-inspired algorithms, to which important attention has been given. It has been found that the tsunami of meta-heuristic algorithms which is based on natural or man made processes, is causing a shift of meta-heuristics away from scientific rigor (Sörensen, 2015), and indeed some newly emerged methods do still require clearer presentation. Thus, in order to evaluate the contribution of the main concept behind each proposed method, theoretical and practical formalisation is required along with an explanation of how the methods can be applied to various optimisation problems. Moreover, it is necessary to clarify how the proposed methods differ from already existing methods (Nesmachnow, 2014). In general, all swarm intelligence based algorithms are naturally inspired and are population-based. They differ in the inspiration and how the agents are exploring and exploiting the search space (Mavrovouniotis, Li, and Yang, 2017). In 1975, GA, which is inspired by natural evolution, was proposed to find a solution to search and optimisation problems (Holland, 1975). In 1989, SDS was introduced to perform evaluation of search and optimisation hypotheses (Bishop, 1989b) . Thereafter, in 1992, Marco Dorigo introduced an innovative and nature-driven meta-heuristic known as ACO to overcome hard Combinatorial Optimisation (CO) problems (Blum (2005a) and Chen et al. (2005)). Soon after, Kennedy and Eberhart developed an algorithm known as PSO to simulate bird flocking social behaviour (Chen, Abraham, and Yang (2005), Chen et al. (2005), and Kennedy (2011)). In 2005, the Artificial Bee Colony (ABC) algorithm was proposed, which is inspired by the behaviour of a honey bee swarm (Karaboga, 2005). Another method was proposed in 2005, known as Glowworm Swarm Optimisation (GSO). It is based on agents called glowworm for multi-modal functions optimisation (Krishnanand and Ghose, 2009). In 2009, a Cuckoo Species Algorithm (CSA) was proposed by Yang and Deb, 2009 . Later on in 2011 a new swarm-based algorithm was introduced, based on modifications to the original CSA. It is known as Modified Cuckoo Search (Walton et al., 2011). Another swarm intelligence algorithm is the Quantum Inspired Cuckoo Search Algorithm (QICSA), aimed to improve CSA diversity. Subsequently, Al-Rifaie, 2014 proposed a new swarm intelligence algorithm derived from the behaviour of flies hovering over food. The algorithm is called DFO and has become a new addition to the existing set of swarm intelligence algorithms. There are many more swarm-based algorithms other than those listed above, among them GP. More details can be found in (Ab Wahab, Nefti-Meziani, and Atyabi (2015) and Parpinelli and Lopes (2011)).

Ever since the introduction of swarm intelligence and evolutionary computation, the number of studies documenting the success of meta-heuristic algorithms in various optimisation tasks and research problems has steadily increased (Parpinelli, Lopes, and Freitas (2002) and Bonabeau, Dorigo, and Theraulaz (1999)). The successful application of swarm intelligence algorithms to a number of problematic situations, such as optimal routes discovery, function optimisation problems, image and data analysis, structural optimisation and scheduling, have been demonstrated in (Dall'Asta et al. (2006) and Tan, Shi, and Niu (2016)). Further applications of the meta-heuristic algorithms have been applied in several other domains, including: bioinformatics, dynamical systems, medical informatics, operations research, machine learning, and even finance and business (Bhattacharyya, 2015). Despite this increase in the use of swarm intelligence algorithms, there is the no-free-lunch theory proposed by Wolpert and Macready, 1997. This theory states that any algorithm will perform equally on average to another algorithm. For example, A and B are two algorithms which have an averaged equal performance; this means if algorithm A performs better than B in some problems, then algorithm B will perform better than A in other problems. Thus, there is no algorithm that performs well in all types of problems, and the algorithm performance should be measured for the given problem (Yang, 2012).

This chapter is organised as follows: Section 3.2 briefly illustrates the main advantages and limitations of population-based techniques. Section 3.3 describes the main population-based algorithms. Section 3.4 then provides an overview of the use of these algorithms in data mining.

3.2 Advantages and disadvantages of swarm intelligence

As with any system, there are several advantages and disadvantages in swarm intelligence. This section discusses the main advantages and disadvantages of swarm intelligence algorithms.

3.2.1 Advantages of swarm intelligence

One of the significant advantages of swarm intelligence algorithms is their highly scalable nature. The outstanding abilities of swarm intelligence algorithms are generally sustained in the event of using group sizes, whether it be sufficiently few or millions of individuals. For example, swarm intelligence systems' control mechanisms do not always rely on swarm size, provided that the size is not too small (Belal et al. (2006) and Tan and Zheng (2013)). Swarm intelligence based algorithms are also known to adapt well to quickly changing environments, as well as making the most of using their self-organisation and auto-configuration capabilities, thereby enabling them to freely and flexibly adapt individual behaviour to the outside environment (Belal et al. (2006), Tan and Zheng (2013), and Kennedy et al. (2001)). Moreover, they can collectively function without any central control, and the swarm is not dependent on a single individual; this is why swarm intelligence is a highly robust system. In other words, in swarm intelligence systems, there is a high degree of fault tolerance capability since the swarm-based system has no single point of failure. In contrast, systems with a single point of failure can potentially jeopardise the system, leading to complete system failure (Dorigo (2007) and Shi (2012)). The swarm is also made of simple components with little capabilities on their own as individuals, but the simple rules they use create a collective sophisticated behaviour for the group.

3.2.2 Disadvantages of swarm intelligence

It is clear that swarm intelligence is growing at a rapid pace and is becoming a far reaching phenomenon. It is an innovative way of designing complex systems where no centralised control is needed (Parpinelli and Lopes, 2011). However, there are some disadvantages or limitations to swarm intelligence based systems. Given the emergent solutions that swarm intelligence systems provide, these systems cannot be used for time-critical applications, such as elevator controllers and nuclear reactor temperature controllers. Nevertheless, swarm intelligence can be applied to non-time critical applications involving various repetitions of a desired activity (Belal et al., 2006). One of the most significant limitations of swarm intelligence optimisation techniques is parameters tuning. Given that various swarm intelligence algorithms' parameters are problematic, they are usually empirically pre-selected based on a problematic trial-and-error scenario (Bhattacharyya, 2015). Another limitation is *stagnation*. Systems may experience some form of stagnation due to limited central coordination or an untimely convergence to a local optimum. For instance, a common form of stagnation in ACO is that ants may eventually follow the same suboptimal path. However, this issue can easily be resolved by carefully setting algorithmic parameters (Yang and Karamanoglu, 2013). For example, fine tuning the swarm-based algorithm's parameters by hybridising with other swarm intelligence algorithms. This can be used as a way to overcome problems like stagnation and resources utilisation.

Despite their limitations, swarm intelligence based optimisation techniques are still popular because of their robustness and flexibility. They are robust because they are made up of simple agents and the failure of one agent has little or no impact on the whole system. For example, ACO was first introduced to solve the Travel Salesman Problem (TSP). After that, it was extended further to solve other issues such as scheduling problems and bioinformatics problems. In terms of data mining, an improvement in performance is found by incorporating swarm-based techniques with data mining techniques. This motivates the use of swarm intelligence techniques in data mining (Martens, Baesens, and Fawcett, 2011).

This section defined swarm intelligence and discussed its main advantages and disadvantages, the next section delves into a number of other aspects associated with swarm intelligence. It includes a brief introduction to major swarm-based algorithms and reviews the recent applications of swarm intelligence in knowledge discovery and data mining.

3.3 Swarm intelligence algorithms

Swarm intelligence has two main approaches. The first category consists of a search technique in which individuals of a swarm move through the solution space in search for a solution to a task. The second category is a data organising approach, which occurs when swarms move data away from low dimensional feature space so that they can achieve a clustering solution of the data. These two categories occur through various meta-heuristic algorithms that are utilised to evaluate solutions to single and multi-objectives fitness functions (Kennedy et al. (2001) and Rao and Patel (2013)). This section introduces some swarm-based techniques. First is the GA algorithm, which is based on natural selection and genetics, and next is SDS, a multi-agent optimisation algorithm that has a strong mathematical framework to describe its behaviour. This is then followed by a brief introduction to ACO, which is based on ant behaviour when searching for food. PSO is introduced after, which mimics the behaviour of birds flocking birds in order to provide guidance for the particles that are searching for the global solution in the search space. Following this, a brief introduction to DE is provided. Finally, DFO is explained and its behaviour is illustrated.

3.3.1 Genetic algorithm

GA is the most popular approach in the research area of evolutionary computation. It is a search technique for global optimisation in the complex search space. As suggested by the name, natural selection, as well as genetic concepts such as mutation, crossover and inheritance are employed in it. John Holland developed the concept of GA in 1975 at the University of Michigan. He worked on the initial version of GA, which involved the particular simulation of the Darwinian principle known as "*survival of the fittest*" (Holland, 1975). Based on this principle, GA reaches the optimal solution after a series of generations or iterative computations.

GA has received increasing popularity in the research area of computer science, business, engineering, operations research, and social sciences (Kumar et al., 2010). Challenging and complex problems in these research areas have benefited from GA, and various problems still require the use of GAs to solve them. GA's main components are chromosome representation, fitness function, initial population, generations' termination criteria, selection function, and the genetic operation for reproduction. The initial population in GA is usually a randomly generated solution. When it comes to termination criteria for the reproduction to stop, the most commonly used approach is to set a maximum number of generations. GA usually works by representing the chromosome (solution) in a string of 0's and 1's (Agrawal and Bala, 2007). In the evaluation step, GA uses the fitness function to evaluate the quality of the solution. The fitness function represents the main requirements for the intended solution of a problem. For example, shortest route, minimum cost, and most compact arrangements are computed in this feature of the algorithm. Based on the chromosomes' fitness value, the selection process starts. There are various ways to select chromosomes for reproduction, such as roulette wheel selection and ranking methods. In the ranking method, the fittest chromosomes have a higher chance (higher probability) of being recombined with other chromosomes to produce improved solutions (Blickle and Thiele, 1996). However, roulette wheel selection is the simplest selection method, in which a random number is generated and the chromosome is selected if its segment spans the selected random number.

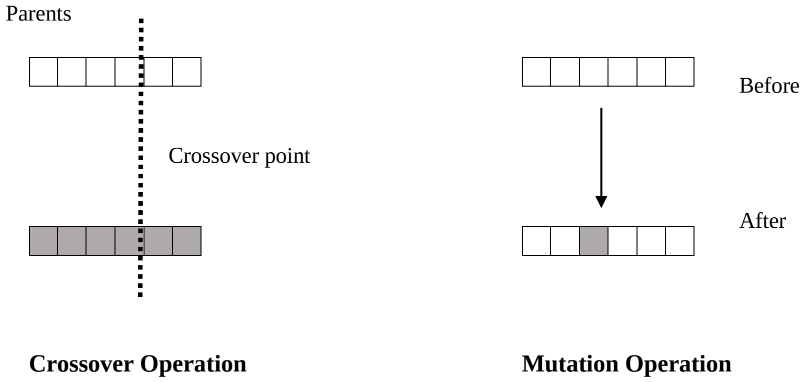

FIGURE 3.1: GA algorithm operations: Crossover (left) and Mutation (right)

Genetic operation for reproduction is the process by which genetic operators such as crossover and mutation are used to create new solutions based on the existing ones. Fig 3.1 shows the GA operation of crossover and mutation. Crossover is the operation in which new solutions, known as Offspring, are created (from the parents) in the search space for exploration. However, mutation is the operation by which the gene is altered. For example, in the string representation of 0s and 1s, the 1 is changed to 0 or vice versa.

GA is capable of finding a good solution. However, this solution is not necessarily the optimal solution. Another limitation is that solving problems with longer chromosomes takes time. Despite its limitations, GA has been widely used in solving complex issues such as scheduling and gene selection. More details on GA's different applications can be found in (Man, Tang, and Kwong (1996), Bandyopadhyay and Pal (2007), and Kumar et al. (2010)).

3.3.2 Stochastic diffusion search

SDS was first described by Bishop, 1989b as a population-based matching algorithm that uses direct communication patterns such as cooperative transport as found among social insects, to perform evaluations of the search and optimisation hypothesis. In SDS, the direct communication between agents is analogous to the "tandem calling" mechanism employed by a particular species of ants called *Leptothorax Acervorum*. De Meyer, 2000 described SDS as a framework that fits within processes governed by mechanisms of variation, replication, and selection. The method employs a probabilistic approach to solve matching and pattern recognition problems (Bishop, 1989b).

Unlike other nature-inspired search methods, SDS, which is rooted in mathematical frameworks, describes the behaviour of the algorithm by investigating convergence to the global optimum in a linear time complexity (Bishop (1989b) and Nasuto (1999)). SDS presents itself as a more effective solution to persistent problems encountered in search and optimisation, in addition to being reliable and fast. The main components of SDS are as follows:

- The model, which is the target pattern.
- The search space, which consists of micro features that define the model.
- Agents performing independent searches with internal operational states containing location-mapping pointing to the search space.

SDS is most applicable to optimisation and search problems where component functions can be evaluated independently using swarm agents who maintain a hypothesis about the optima (Nasuto and Bishop, 1999). In SDS, the agents' population has a "*hypothesis*" about the possible solutions; these hypotheses are partially evaluated in the *test phase*. This is to provide feedback that ensures the agents' convergence on promising solutions. Using SDS, agents' communication and the "*partial*" evaluation of hypotheses play a critical role in the performance of the agents and the emergence of "*intelligence*" (Bishop, 1989b). Unlike other swarm intelligence algorithms, SDS is not based on applying non-linear transfer and does not calculate based on standard learning rules (Nasuto, 1999). Given its approach, agents' communication appears as a collective and emergent property (Nasuto and Bishop, 1999). Unlike ACO, which is based on stigmergetic communication, SDS uses direct communication between agents.

SDS then functions by finding the best match for a given model in any search space using a *diffusion process*, whereby each agent operates independently and only communicates to provide information about their findings. During SDS, every agent is able to examine the entire search space and process information about the target. For example, each agent can access an entire document containing texts and when the selected hypothesis test function returns a positive result, the agent becomes more active and able to recruit others to increase the agent population. Once an individual agent has individually evaluated its hypothesis, individuals in the population come together through various methods to achieve diffusion (exchanging information and communicating with each other). Once information is communicated about the positive hypotheses, hypotheses that do not yield positive results are discarded and the number of agents multiply on the positive result through the information exchanged. In principle, the partial function evaluation feature in SDS makes it applicable to a dynamically changing problem space. Thus it is a dynamic, not static, process. This feature of dynamic changes causes SDS to be widely used in various applications across different fields, among them, eye tracking in facial images, where a stochastic search network is combined with an n-tuple network (Bishop and Torr, 1992), site decision for transmission equipment for wireless networks (Hurley and Whitaker, 2002), and mouth locating in human faces images (Grech-Cini and McKee, 1993). More details on SDS applications can be found in Section 3.3.2.

SDS Architecture

SDS starts the search with the initialisation phase. This is then followed by iterations of the test and diffusion phases. The structure of SDS is shown in Algorithm 1.

Algorithm 1 SDS Algorithm

```
Initialising phase
while stopping condition is not met do
    Test phase
    Diffusion phase
```

Initialisation phase: The initialisation phase is where each agent randomly selects a hypothesis (i.e. an element’s index or, in the case of a dataset, the instance number) from the search space. These *“pointers”* are later used to lead the search process of the SDS population.

Test phase: In the test phase, which follows the initialisation phase, each agent is assigned to a random hypothesis in the search space, and each agent’s hypothesis is partially and individually evaluated based on the objective function; if the hypothesis evaluation is successful, the agent is set to be active (boolean value) and if not, it is inactive. Therefore, at the end of the test phase, each agent adopts either of the two possible boolean outcomes: active or inactive.

Diffusion phase: During the diffusion phase, information about the hypotheses is exchanged among agents, depending on the recruitment strategy employed. For example, the standard SDS algorithm employs the passive recruitment strategy in which each inactive agent selects another agent randomly; if the randomly selected agent is active, the hypothesis of the active agent is *diffused* to the inactive agent; otherwise, the inactive agent randomly selects a hypothesis from the search space. There are other types of recruitment strategies, and the next section describes variations in recruitment strategies to balance between global exploration and local exploitation.

Recruitment strategies

Various types of recruitment strategies are employed for SDS to converge. The choice of recruitment strategy depends on the desired level of greediness and robustness. The methods include active, passive, dual, context sensitive, and context free strategies, as shown in Fig 3.2. The agent in SDS can be in one of three different status: *Active status*: if the agent was successful in the hypothesis evaluation at the test phase, *inactive status*: if the agent was not successful at the test phase and *engaged status*: if the agent is engaged in a communication with another agent (Al-Rifaie and Bishop, 2013a).

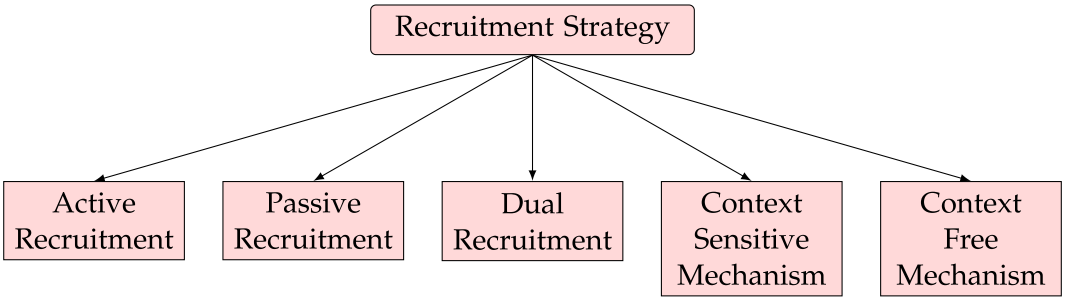

FIGURE 3.2: Recruitment strategies

Passive recruitment strategy: A passive recruitment strategy is a process in which inactive agents randomly select other agents to adopt their hypotheses if those hypotheses are active. Otherwise, the inactive agent is assigned to a new random hypothesis, as shown in Algorithm 2. In passive recruitment strategy, it is possible but unlikely that the agents' population would converge to any hypothesis in a single attempt.

Algorithm 2 Passive recruitment strategy

```
for Ag = 1 → Total_No_of_agents do
    if Ag.active() == false then
        rand_Ag = pick a random agent()
        if rand_Ag.active() == true then
            Ag.setHypo(rand_Ag.getHypo())
        else
            Ag.setHypo(random_Hypo())
```

Active recruitment strategy: Active recruitment takes place when active agents randomly select other agents. If the selected agent is inactive and not engaged in a communication with another agent, then the active agent's hypothesis is passed to the inactive agent and the agent is set as engaged. This is repeated for all the active agents. Otherwise, if the agent is neither active nor engaged, it will be assigned a new random hypothesis. See Algorithm 3.

Algorithm 3 Active recruitment strategy

```
for Ag = 1 → Total_No_of_agents do
    if Ag.active() == true then
        rand_Ag = pick a random agent()
        if rand_Ag.active() == false and rand_Ag.Engaged() == false then
            rand_Ag.setHypo(Ag.getHypo())
            rand_Ag.Engaged(true)
for Ag = 1 → Total_No_of_agents do
    if Ag.active() == false and Ag.Engaged() == false then
        Ag.setHypo(random_Hypo())
```

Dual recruitment strategy: This is a combination recruitment method in which both active and inactive agents randomly select other agents. If an active agent randomly selects another agent that is inactive and not engaged in another communication, then the active agent shares its hypothesis with the inactive one and the inactive agent is set as engaged. Moreover, if an agent that is inactive and not engaged in a communication, it selects a random agent that is active, the active agent will share its hypothesis with the inactive agent, which will be set as engaged. Finally, if there is still an agent that is inactive and not engaged in a communication, it will be assigned a new random hypothesis. See Algorithm 4.

Algorithm 4 Dual recruitment strategy

```
for Ag = 1 → Total_No_of_agents do
    if Ag.active() == true then
        rand_Ag = pick a random agent()
        if rand_Ag.active() == false and rand_Ag.Engaged() == false then
            rand_Ag.setHypo(Ag.getHypo())
            rand_Ag.Engaged(true)
    else
        rand_Ag = pick a random agent()
        if rand_Ag.active() == true and rand_Ag.Engaged() == false then
            Ag.setHypo(rand_Ag.getHypo())
            Ag.Engaged(true)
for Ag = 1 → Total_No_of_agents do
    if Ag.active() == false and Ag.Engaged() == false then
        Ag.setHypo(random_Hypo())
```

Context sensitive recruitment strategy: In contrast with dual recruitment strategy, in active recruitment strategy the robustness and greediness decrease. However, in dual recruitment, the two features are increased. Even though an increase in greediness in dual recruitment causes a decrease in the robustness of the SDS algorithm, context sensitive recruitment strategy controls this decrease (Myatt, Nasuto, and Bishop, 2006). Therefore, the context sensitive recruitment strategy will increase diversity and improve global exploration. In context sensitive recruitment strategy, if the randomly selected agent is active and shares the same hypothesis, the selecting agent changes its status to inactive and picks a new random hypothesis, as shown in Algorithm 5. This recruitment strategy facilitates global search by freeing up some of the resources (Al-Rifaie and Bishop, 2013a).

Context free strategy: In context free recruitment strategy, if the randomly selected agent is active, the selecting agent changes its status to inactive and picks a new random hypothesis regardless of whether it shares

Algorithm 5 Context sensitive recruitment strategy

```
for Ag = 1 → Total_No_of_agents do
    if Ag.active() == true then
        rand_Ag = pick a random agent()
        if rand_Ag.active == true and Ag.getHypo() == rand_Ag.getHypo() then
            Ag.setActive(false)
            Ag.setHypo(random_Hypo())
```

the same hypothesis or not, as shown in Algorithm 6. This recruitment strategy ensures that other possible solutions are investigated and about 50% of the agents explore the search space.

Algorithm 6 Context free strategy

```
for Ag = 1 → Total_No_of_agents do
    if Ag.active() == true then
        rand_Ag = pick a random agent()
        if rand_Ag.active == true then
            Ag.setActivity(false)
            Ag.setHypo(random_Hypo())
```

More details on recruitment strategies can be found in (Bishop (1989a), Bishop (1989b), and Al-Rifaie and Bishop (2013a)).

Types of SDS

A number of variants of SDS have been proposed, including methods without standard parameters for selecting hypothesis to solve the random selection problem (Al-Rifaie and Bishop, 2013a). One of the forms of SDS algorithms is unlabelled SDS, which makes no agent accessible to other agents' internal state (De Meyer, 2000). Apart from the diffusion phase, the operation of unlabelled SDS is similar to the standard process. When compared to standard SDS, it has the same equilibrium population, but unlabelled SDS has a slower convergence speed. Another type of SDS is data-driven SDS (DDSDS), which has similar components and processes to SDS other than a few dissimilarities in some stages of the algorithm (Myatt and Bishop, 2003). DDSDS is used to solve parameter estimation problems such as locating spoken word in an audio file that has some noise. The algorithm contains a composite hypothesis known as the manifold hypothesis and datum hypothesis. The manifold hypothesis contains the minimum description of the hypothesis while the datum hypothesis contains the smallest building block of the hypothesis. Bishop, 2003 has also proposed a Coupled SDS (CSDS) based on the modification of other SDS algorithms such as DDSDS. Like DDSD, CSDS has a composite hypothesis that includes the manifold hypotheses and datum hypotheses. However, in CSDS there are two different populations of agents and two different types of hypothesis are formed. In CSDS, the datum hypotheses are chosen randomly from the whole search space. These types of SDS have been proposed to improve the performance of standard SDS to increase the convergence speed and tolerate the noise in the search space.

Applications

Existing studies have tested the efficiency of different SDS algorithms and their fitness for solving various search problems (Al-Rifaie and Bishop, 2013a). In 1989, SDS was first introduced using a simple text search example to illustrate the use of partial function evaluation and partly to evaluate the text to find the best match to the model (Bishop, 1989a). Later in 1992, SDS was used in applications like eye tracking in grey scale facial images by using a combination of a stochastic search network and an n-tuple network (Bishop and Torr, 1992). In a similar project, SDS was used to locate mouths on images of human faces (Grech-Cini and McKee, 1993). SDS was also applied to locate an autonomous wheelchair in a busy environment. The results of this work indicate that Focused Stochastic Diffusion Network (FSDN), along with laser range sensors, provides a valuable solution to the Automated Guided Vehicles (AGV) localisation problem (Beattie and Bishop, 1998). Later, SDS was used in site decisions for transmission equipment for wireless networks (Hurley and Whitaker, 2002). Another version of SDS, known as Constrained Stochastic Diffusion Search (CSDS), was introduced to solve the best fit sequence matching problem. It has been applied in computational molecular biology (Jones, 2002).

In another visual tracking application, Group Stochastic Search (GSS) was used to track objects like heads in cluttered environments. In this proposed approach, each agent uses SDS, histogram interaction methods, and an n-tuple neural network to evaluate the location (Evans and Ferryman, 2005). In 2006, Nicran used standard SDS in voting methods (Nircan, 2006). Two years later in 2008, SDS was applied in feature tracking as other methods like Euclidean distance (ED) were computationally expensive (Hernandez-Carrascal and Nasuto, 2008).

In another study, SDS was used in room design and the creation of virtual rooms. The placement of objects like books on a bookshelf were solved using SDS (Cant and Langensiepen, 2009). SDS has been further used in social networks, where a new version of the algorithm was created and called Stochastic Diffusion Market Research (SDMS). In SDMS, a new advertisement method between participating businesses was proposed. The results indicate that SDMS convergences to a stable state in which the distribution of market prices changes to power-law properties (Salamanos et al., 2010). Medical imaging is another application in which SDS has been used, where it was employed for the first time on bone scans (Al-Rifaie, Aber, and Raisys, 2011). This work was later extended to mammography (Al-Rifaie, Aber, and Oudah, 2012). Moreover, SDS has been used in various artistic applications. SDS-PSO is a hybrid algorithm used to sketch drawings received from input images (Al-Rifaie (2011) and Al-Rifaie, Bishop, and Caines (2012)). A further application of SDS is machine learning. SDS has been used in reinforcement learning to discover instances of strong correlations. The results of this study show that SDS is able to discover the majority of instances at different time indexes (Hughes, 2011).

This part of the thesis has reviewed the concept of SDS and its excellent approach to solving complex search and data optimisation problems in various applications. Due to its partial evaluation of objective function and its communication pattern, which allows good information to be disseminated to agents quickly, SDS has proven to be an effective technique in solving various search and optimisation problems. Previous experiments have also shown that SDS can be applied not only to static but also dynamic problems (Al-Rifaie and Bishop, 2013a). However, due to its nature, SDS is found only to perform well on certain problems. Thus, further research is required to make SDS more dynamic and have better auto-tuning to solve more problems such as class imbalance in data mining. Although more sophisticated versions of SDS and other models have been produced, hybridisation strategies that combine population based algorithms have been argued to be valuable, especially as the type of multi-population concept found in these algorithms can be applied. Such an approach can also solve larger scale optimisation and search problems. Expansion of SDS is also likely to enable it to develop scope for solving wider problems such as molecular optimisation and water distribution system optimisation.

The next section explores another swarm-based technique known as ACO.

3.3.3 Ant colony optimisation

The ACO algorithm, inspired by the social behaviour of ant colonies, was introduced in 1992 (Dorigo, 2007). Ants are social insects whose main interest is the colony in which they reside and the survival of their colony, as opposed to individual survival. Furthermore, ants have the ability to discover the shortest route to their closest food source, and this scenario inspired the proposed ACO algorithm (Merkle and Middendorf, 2014).

When ants search for their food, they first randomly explore the closest area to their nest. Ants will leave behind a chemical pheromone trail during their trip, and the smell helps them to keep track of the areas they have already explored and prevents dwindling away from the colony. Ants will often select the most suitable paths, indicated by the strongest pheromone concentration (Zhang et al., 2014). Upon discovering a food source, the ant will first evaluate the quantity and quality, and will then bring it back to its nest. Upon returning to the nest, the potency of the ant's pheromone reflects the food quality and quantity, and the trail directs other ants to the food source. ACO uses an indirect communication mode in which there is no exchange of information between the ants directly. In short, pheromone trails are forms of indirect communication that allow ants to locate the food source nearest to their nest. The ACO algorithm consists of five key steps (Dorigo and Blum, 2005). The steps are as follows:

- Establish a pheromone trail.
- Use the pheromone trail to construct a solution.
- Each ant comes up with a solution to the problem based on a probabilistic approach.
- Define the state transition rule based on the pheromone conditions.
- Update the pheromone trail.

There are two key stages in a global pheromone updating rule. The first is the evaporation stage, in which a tiny amount of pheromone evaporates, and the second is a reinforcement stage, in which each ant excretes a quantity of pheromone based on the current state of its solution The process is repeated until the initiation of a termination condition (Toksari, 2006).

In terms of ACO convergence, the time is uncertain but granted. Moreover, the positive feedback leads to discovering good solutions. ACO has been widely used in various applications such as transportation engineering (Teodorović, 2008) and data mining (Parpinelli, Lopes, and Freitas, 2002). Due to its popularity, there have been several variants of algorithms since the 1990s including: Beam-ACO (Blum, 2005b), Max-Min Ant System (MMAS) and Ant Colony System (ACS) (Chaparro and Valdez, 2013). More details on ACO applications can be found in (Dorigo and Stützle (2003) and Blum (2005a)).

The next section looks into another population-based algorithm, called PSO.

3.3.4 Particle swarm optimisation

Grosan, Abraham, and Chis, 2006 state that PSO is "*a population-based search algorithm and is initialised with a population of random solutions, called particles*". The algorithm was inspired by the Reynolds boids model. Boids is an artificial simulation program that simulates the flocking behaviour of birds. The goal of this simulation is to reproduce the flocking behaviour of birds. There are three basic rules to explain collective behaviour: cohesion, separation and alignment (Reynolds, 1987). PSO was initially designed to simulate food searches from a bird's perspective; this is referred to as a "*cornfield vector*" (Kennedy and Mendes, 2002). PSO consists of a swarm of birds, and each bird is called a particle. All PSO particles are related to a velocity and flies through a search space consisting of other velocities; these are adjusted based on their historic behaviour (Cheng, Shi, and Qin, 2015). Consequently, these particles often fly toward a much improved search area throughout the process.

For example, there may be a number of food searching processes going on among a flock of birds in a particular area. Only one piece of food is left within the search space, and the birds are unaware of its location. The birds, however, have an idea of how far away the food is and where their neighbouring flock members are. Therefore, one must ask which strategy a bird could employ to locate the food: simply find and follow the bird closest to where the food is located. PSO learns from this kind of situation, and becomes used to overcoming existing optimisation issues. The solution in a PSO-type scenario can be represented as a bird in the search space or the "*particle*" in the d-dimensional space, which carry a fitness value. These values are assessed according to the function to be optimised, known as the fitness function, and comprise velocities which give direction to the flying particles.

Furthermore, PSO starts with a solution (a point in the d-dimensional search space), and after that seeks out optima by informing each generation. Per generation, a particle is updated via two key *"best"* values; the most ideal position of the particle, which is known as the personal best *"pBest"*, and the best position of all other particles in the swarm, known as the global best *"gBest"* (Cheng, Shi, and Qin, 2015). The particle is updated by adding a velocity to the current positions. The particle's velocity is updated using the equation below:

$$v_{id}^{t} = wv_{id}^{t-1} + c_1 r_1 \left(p_{id} - x_{id}^{t-1}\right) + c_2 r_2 \left(g_{id} - x_{id}^{t-1}\right) \tag{3.1}$$

$$x_{id}^{t} = v_{id}^{t} + x_{id}^{t-1} \tag{3.2}$$

where w is the inertia weight whose; $\vec{v}_{id}^{t-1}$ is the velocity of particle i in dimension d at time step $t-1$; $c_{1,2}$ are the acceleration constants for pBest and neighbourhood best respectively; $r_{1,2}$ are random numbers generated from a uniform distribution on the unit interval $U(0,1)$ and used to add stochasticity to the algorithm; p_{id} is the pBest position of particle $\vec{x}_i$ in dimension d; and g_{id} is neighbourhood best at dimension d (Al-Rifaie and Bishop, 2013b).

After the update, particles move on to a new location on their new velocities. Algorithm 7 summarises the PSO algorithm.

Algorithm 7 PSO

```
Initialise a population of particles with random values positions and velocities from $D$ dimensions in the search space
while Termination condition not reached do
    for Each particle $i$ do
        Calculate the fitness function
        if the fitness value > the stored fitness value (pBest) then
            Set the fitness value as the new (pBest)
    Find the particle with best fitness value and set as (gBest)
    for Each particle $i$ do
        Calculate Velocity
        Update the position of the particle using Equation 3.1 and 3.2
```

In PSO, the inertia weight is applied to provide a balance between the global exploration and local exploitation. Basically, it controls the effect of the previous velocity and the current particle's velocity. Therefore, setting a large value for the inertia weight will increase the algorithm global exploration while a small value will increase the local exploitation. A suitable setting for the inertia weight will provide a balance between the global exploration and local exploitation thus requiring fewer iterations to search for the optimal value.

PSO is easy to implement and has a memory of the particle's previous best location. However, it suffers from premature convergence and is mainly used for continuous optimisation. The original PSO has gone through a number of changes since its introduction to overcome various limitations, such as premature convergence and being trapped in a local optimum. Moreover, PSO was initially introduced as an optimiser in the continuous search space. The first version of PSO for discrete optimisation was proposed by Kennedy and Eberhart, 1997. For more details on other variants of PSO, see (Kennedy (2003) and Poli (2008)).

3.3.5 Differential evolution

DE is another population-based algorithm introduced by (Storn, 1996). It is a multi-dimensional real value optimiser that has been widely used in optimisation tasks that are noisy and change over time. Similar to GA, DE uses crossover, mutation and selection. However, DE relies on mutation when constructing a new solution, as opposed to GA, which relies on crossover. DE starts by having a population of candidate solutions known as agents. Agents move in the search space to create new solutions by combining existing ones using a mathematical formula. If the new solution or the position of an agent is better, then it forms a part of the population; otherwise it is discarded. This is repeated until a satisfactory solution is found.

Algorithm 8 DE Algorithm

```
Initialising phase
Evaluation phase
do
    Mutation phase
    Recombination phase
    Evaluation phase
    Selection phase
while stopping condition is not met
```

DE has performed well in various search and optimisation problems. Storn and Price, 1997 found that DE is better than genetic algorithms and simulated annealing. In a study by Ali and Törn, 2004, the authors evaluated various population-based algorithms in terms of their efficiency and robustness. The results indicate that DE outperforms GA and controlled random search. More details about other applications of DE can be found in (Das, Abraham, and Konar (2008), Chakraborty (2008), and Chen, Rangaiah, and Srinivas (2017)).

3.3.6 Dispersive flies optimisation

Inspiration was taken from nature to solve the optimisation problems of a search space by using the swarming behaviour of the objects in the search space to find optimal solutions. DFO, first introduced in (Al-Rifaie, 2014), is an algorithm inspired by the swarming behaviour of flies hovering over food sources. Several factors affect the swarming behaviour of flies, including the presence of threat, which disturbs the flies' convergence to the optimal value. As a result, both the formation and breaking of the swarms are considered in the proposed algorithm (Al-Rifaie, 2014). The position vectors of the flies are defined as:

$$\vec{x}_i^t = \left[x_{i1}^t, x_{i2}^t, \ldots, x_{iD}^t\right], \qquad i = 1, 2, \ldots, N \tag{3.3}$$

Where t is the current time step, D is the dimension of the problem space and N is the number of flies (population size).

In the first generation, when $t = 0$, the i^{th} vector's d^{th} component is initialised as:

$$x_{id}^0 = x_{\min,d} + r\left(x_{\max,d} - x_{\min,d}\right) \tag{3.4}$$

Where r is a random number drawn from a uniform distribution on the unit interval $U\left(0,1\right)$; $x_{\min}$ and $x_{\max}$ are the lower and upper initialisation bounds of the d^{th} dimension, respectively. As a result, each fly in the population is randomly initialised with a position in the search space (Al-Rifaie and Aber, 2016).

At each iteration, the position vectors' components are independently updated. This is achieved by considering three different aspects: the component's value, the corresponding value in the best neighbouring fly's vector that has the best fitness, and the corresponding value in the best fly in the whole swarm. The updated equation is described below:

$$x_{id}^t \;=\; x_{i_{nd}}^{t-1} + \mathrm{U}\left(0,1\right) \times \left(x_{sd}^{t-1} - x_{id}^{t-1}\right) \tag{3.5}$$

Where $x_{i_{nd}}^{t-1}$ is the position value of $\vec{x}_i^{t-1}$'s best *n*eighbouring fly in the d^{th} dimension at time step $t-1$; x_{sd}^{t-1} is the swarm's best fly value in the d^{th} dimension at time step $t-1$; and U $(0,1)$ is the uniform distribution between 0 and 1 (Al-Rifaie (2014) and Al-Rifaie and Aber (2016)).

Technically speaking, there are two main components of DFO: the use of social neighbouring network in updating the flies' position and the flies' communication with each other about the best results found. Moreover, the disturbance of the swarm plays an important role in DFO as it displaces the flies, leading to the discovery of better positions. Thus, at the update phase a stochastic element is introduced that is known as the *disturbance threshold* or Δ. As a result, if a random number, r, generated from a uniform distribution on the unit interval $U(0,1)$ is less than Δ, each component of the flies' position vectors are reset. The *disturbance threshold* provides a disturbance and helps in avoiding stagnation over a local minima (Al-Rifaie, 2014).

Algorithm 9 summarises the DFO algorithm.

Algorithm 9 Dispersive Flies Optimisation

```
while Function Evalutions < Evaluations Allowed do
    for $i = 1 \rightarrow N$ do
        $\vec{x}_i$.fitness $\leftarrow f(\vec{x}_i)$
    $\vec{x}_s$ = arg min $[f(\vec{x}_i)], \quad i \in \{1, \dots, N\}$
    for $i = 1 \rightarrow N$ and $i \neq s$ do
        $\vec{x}_{i_n}$ = arg min $[f(\vec{x}_{i-1}), f(\vec{x}_{i+1})]$
        for $d = 1 \rightarrow$ D do
            if $(u < \Delta)$ then
                $x_{id}^{t+1} \leftarrow x_{\min,d} + u\,(x_{\max,d} - x_{\min,d})$
            else
                $x_{id}^{t+1} \leftarrow x_{i_{nd}}^{t} + u(x_{sd}^{t} - x_{id}^{t})$
```

In summary, DFO is a population-based algorithm, originally proposed to search for the optimum value over the continuous search space. Although the algorithm is simple, it has been found that DFO outperforms the standard versions of the well-known PSO, GA, and DE algorithms on an extended set of benchmarks over three performance measures of error, efficiency and reliability (Al-Rifaie, 2014). It is shown that DFO is more efficient in 84.62% and more reliable in 90% of the 28 standard optimisation benchmarks used (Al-Rifaie and Aber, 2016). Not only has theoretical research been carried out on the DFO algorithm, but it has also recently been applied to medical imaging

(Al-Rifaie and Aber, 2016); furthermore, ongoing and current research is being conducted in the fields of image analysis, simulation and gaming (King and Al-Rifaie, 2017), computational aesthetic measurements (Al-Rifaie et al., 2017a), digital arts (Al-Rifaie et al., 2017b), protein folding and more.

A comparison between PSO and DFO

PSO is based on the social behaviour observed in a bird flock. It has a population called particles that moves around in the search space based on two vectors: the particle position and the velocity. In addition to the two vectors, each particle has a memory of its personal best position. Each particle movement in the search space is a factor of its personal best particle and the global best particle. Originally, PSO was applied to continuous optimisation problems. However, it has been extended further to solve discrete optimisation problems and even multi-objective optimisation problems (Kennedy, 2011).

DFO is based on the swarming behaviour of flies hovering over food sources. The formation of the swarm is affected by the presence of threats which can disturb the flies. Thus, in DFO both the formation of the swarms and the breaking of the swarms is considered in the algorithm. In DFO, the movement of the flies is affected by the best neighbouring fly and the best fly in the whole swarm. The flies will be dispersed from their location depending on the disturbance threshold. This is to explore new locations in the search space (Al-Rifaie, 2014).

PSO and DFO have many similarities in that both are continuous, population-based optimisers and a fly or particle's fitness is evaluated using a fitness function. Also, both have a number of parameters to tune that affect algorithm performance when dealing with a certain problem. PSO has four parameters: population size, the acceleration constants, $c_{1,2}$, and the weight w. Moreover, in terms of memory, PSO requires more memory to store information on personal previous best particle vector. Unlike PSO, DFO has only two tunable parameters: the population size and the disturbance threshold. Despite the introduction of the other variants of PSO in an attempt to simplify the algorithm such as the bare bone PSO in which the velocity vector is removed, DFO still has less components, and it exhibits better performance when using the three measures of error, efficiency and reliability (see Eq 1, 2 and 3) in (Al-Rifaie, 2014) . The simplicity of DFO in terms of parameters and components allows for further analysis.

There are many more swarm intelligence and population-based algorithms available, but they are not discussed in this section as only GA, ACO, PSO, and DE as well as a more in-depth introduction to SDS and DFO are within the scope of this thesis. The reason is to explore the possibility of using these two population-based algorithms for solving the class imbalance issue in data mining.

The next section reviews the most commonly used population-based algorithms in applications related to class imbalance in data mining.

3.4 Swarm intelligence and data mining

Although swarm optimisation and data mining may appear to have very few properties in common, these properties can be utilised to develop alternative methods to those that are very difficult to implement. The motivation for using swarm intelligence and population-based algorithms to solve various data mining issues is that these are fast, adaptable, and able to perform a global search to find the optimal solution. However, data mining techniques perform a local grid search in the solution space. As a result, various studies have been carried out to benefit from meta-heuristics algorithms' speed and search capabilities to solve various data mining issues like feature selection, parameters tuning, and class imbalance.

GA is a meta-heuristic search technique inspired by natural evolution that has been widely used in data mining. In terms of parameter optimisation, research has been carried out to tune the SVM parameters using GA (Chunhong and Licheng (2004), Liu, Jia, and Ma (2005), and Pourbasheer et al. (2009)). For example, Wu et al., 2007 proposed a real-valued GA to optimise the kernel parameters: C and γ. This is to improve SVM classification accuracy and generalisation ability in predicting bankruptcy. Liu et al., 2014 proposed a hybrid GA approach to forecast short-term wind speed, in which GA was used to optimise the SVM parameters while improving its generalisation ability. In another approach by Samadzadegan, Soleymani, and Abbaspour, 2010, GA was used to optimise the SVM parameters in multi-class problems. The results indicate that GA outperforms grid search when conducting experiments on various benchmark datasets.

In terms of feature selection, various approaches have been carried out using GA in gene selection. Silva, Souza Ribeiro, and Amaral, 2013 used GA to perform feature selection for a cancer dataset. Experimental results show the outperformance of the proposed approach when compared with other classifiers on the same datasets. In another novel approach by Huerta, Duval, and Hao, 2010, LDA is combined with GA to find the most informative genes. The GA based approach was applied on seven real-world datasets. The results show that the proposed approach gives higher accuracy with a small number of genes. Ahmad et al., 2013 proposed a GA based approach for the automatic and simultaneous parameters tuning and feature selection of multi-layer perception networks. Karnan and Thangavel, 2007 introduced a GA based method for cancer diagnosis. This proposed approach was used to detect microcalcifications in mammograms that can be characterised as breast cancer. In this method, AUC was used as an evaluation metric on 114 abnormal digitised mammograms available from Mammogram Image Analysis Society database [1]. Diaz, Pinon, and Solano, 2014 proposed a GA based approach for feature selection for SVM and Artificial Neural Network (ANN) in lung cancer diagnosis. Yu and Cho, 2003 introduced a GA-SVM wrapper approach for feature selection in keystroke dynamics identity verification. Wang, Yu, and Liu, 2005 proposed a GA based feature selection approach for SVM known as GA-SVM. The results indicate that the proposed approach outperforms the original SVM when applied to the UCI spam datasets. Another hybrid approach using a combination of feature ranking and wrapper method for feature selection using multi-class SVM in microarray datasets has been proposed by Agrawal and Bala, 2007. First, the features were ranked using Rankgene Su et al., 2003. Then, GA was used on the top ranked genes to select a smaller feature subset. In this proposed approach, GA is used to implement the wrapper method, and the fitness function objectives maximise accuracy while minimising the number of features. The proposed approach performed well on four microarray datasets. More details on other GA based wrapper methods using various fitness functions with different weightings for classifier performance and feature subset size can be found in (Yang and Honavar (1998) and Huang, Cai, and Xu (2007)).

In terms of PSO as an instrument for data mining, various approaches have been proposed in the literature. Sousa, Silva, and Neves, 2004 proposed a model to evaluate how useful PSO is for data mining. The performance of three key PSO variants and the GA and Tree Induction Algorithm (J48) were compared. The results conclude that PSO algorithms can be used to provide solutions to various classification tasks. Moreover, it was found that PSO produces competitive results, not only when compared with other evolutionary algorithms but also with standard algorithms in the field such as J48.

Researchers have also applied feature selection techniques using PSO (Alba et al. (2007) and Chuang et al. (2008)). An example of a PSO based feature

[1] Dataset is a available at: http://www.mammoimage.org/databases/.

selection approach was proposed by Chuang, Ke, and Yang, 2016. The proposed approach is a hybrid filter and wrapper method. At the first stage, information gain was used to filter the most informative genes. Then, an improved binary PSO is implemented as a wrapper approach to select a gene subset that improves classification accuracy. The results indicate that the proposed approach is capable of reducing dimensionality and improving the classification accuracy for microarray classification. Another model proposed by Zahran and Kanan, 2009 uses PSO for text feature selection. It reduces the dimensionality of the dataset in order to improve text categorisation efficiency. The proposed model has been evaluated against others including; chi-square and document frequency. The results illustrate the superiority of the proposed model. Two multi-objective PSO algorithms for feature selection were proposed by Xue, Zhang, and Browne, 2013. In the first algorithm, the concept of non-dominated sorting has been used with PSO to solve the feature selection problem. The second algorithm applies crowding, mutation and dominance to search for the optimal pareto front solution using PSO. Both algorithms outperform three other well-known multi-objective algorithms.

Another important application of PSO is in SVM parameter optimisation to solve the class imbalance problem at the algorithmic level. PSO has been used to improve SVM performance accuracy by selecting the best classifier. The results show that PSO is able to optimise SVM parameters, which will lead to better classification accuracy (Garšva and Danenas (2014) and Melgani and Bazi (2008)). In another study, a hybrid approach that uses PSO and SVM to improve image classification was proposed. The proposed approach used the images in the Corel image datasets and the results indicate that the classification accuracy of the PSO based approach outperforms SVM, Back Propagation Neural Network (BPNN), and RBF neural network (Zhang, Xie, and Cheng, 2010).

Microarray data classification is one of the major challenging issues in data mining. PSO has been applied to improve the classifier performance on medical datasets. For example, Subbulakshmi and Deepa, 2015 proposed a model that integrates PSO with the Extreme Learning Machine (ELM) classifier. Here, PSO was used to determine ELM optimum parameters to improve the model generalisation ability. The proposed model performed well when compared with other classifiers on five medical datasets available at the UCI machine learning repository. Valdés, 2004 developed a hybrid model, which was inspired by PSO and combined it with traditional optimisation approaches.

This method is used in a number of experiments involving high dimensional datasets as a way of understanding the structure of processed and raw data types. Moreover, experiments involving datasets based on common diseases, such as Alzheimer's disease, suggest that combining PSO with traditional optimisation methods can help with achieving high quality visual representation. Studies have also been conducted on the behavior of certain parameters when controlling swarm evolution. PSO has also been used for both image processing and pattern recognition (Omran, Engelbrecht, and Salman, 2005a). A novel clustering technique based on PSO was introduced and applied to image segmentation and unsupervised classification. This PSO-driven approach was introduced to overcome a number of image-based problems, such as spectral un-mixing problems and colour image quantisation.

DE has also been used in data mining. Omran, Engelbrecht, and Salman, 2005b proposed a DE based model for image classification. In another model by Prabusankarlal, Thirumoorthy, and Manavalan, 2017, DE was used with ELM and rough set feature selection for the classification of breast masses in ultrasound images. Garcia-Nieto, Alba, and Apolloni, 2009 proposed a DE based feature selection using SVM called DESVM. The proposed DESVM approach has been evaluated using two widely used microarray datasets: Diffuse Large B-cell Lymphoma (DLBCL) and Colon Tumour gene expression datasets. Experimental results indicate that DESVM gives better results when compared with other methods from the literature. In another approach by Wang et al., 2012, DE was used to tune the Support Vector Regression (SVR) parameters in load forecasting. The proposed approach outperformed the standard SVR model, BPNN, as well as regression forecasting models in annual load forecasting. In a study by Li and Yin, 2012, DE was used for quantitative interpretation of self-potential data in which six parameters are optimised. The self-potential method is an optimisation problem in geophysics mainly used for mineral exploration. In this experiment, self-potential data for geometrical body approximation such as a sphere was used. DE was applied on three different kinds of data from Turkey: noise-free data, contaminated synthetic data, and field example. Experimental results show that DE is applicable in solving the quantitative interpretation of self-potential data when compared with other methods from the literature. Chauhan, Ravi, and Chandra, 2009 proposed a model that used DE for bankruptcy prediction in the bank industry. The model is called the DE trained Wavelet Neural Network (DEWNN). Results indicate that the proposed approach outperforms the standard Wavelet Neural Network (WNN).

In terms of ACO, Deneubourg et al., 1991 first introduced ant colony-based clustering algorithms by imitating different kinds of natural phenomena. Data clustering is essentially based on the idea that secluded things or items must be dropped and picked up in other locations where similar items can be found. The ACLUSTER algorithm proposed by Ramos, Muge, and Pina, 2002, helps to interpret ant-like behaviours. Applying this strategy supports ants to adaptively discover clusters of objects (Yang, 2014). Another clustering algorithm known as AntClust-Miner, which is based on ACO and was proposed by Salama and Abdelbar, 2016, uses two different clustering approaches: instance-based and medoid-based clustering. This is done to create cluster-based classification systems. The performance of the proposed approach has been evaluated on 30 UCI machine learning repository benchmark datasets and three different classifiers: C4.5, *k*-NN and the Ripper algorithm. The results indicate that the proposed algorithm, AntClust-Miner, gives statistically significant results when compared to *k*-NN and the Ripper algorithm. AntPart, presented by Admane et al., 2006 is another unsupervised classification technique inspired by how specific classes of ants, such as *Pachycondyla apicalis*, behave. Moreover, the performance of both AntPart and three others (AntClass, AntTree and AntClust) were compared, all of which were inspired by ants' social behaviour. A time-series segmentation algorithm inspired by ACO was proposed by Weng and Liu, 2006, to demonstrate the variability of time series data. A bottom-up method was used, as it was reported to provide better results for time-series segmentation. The findings suggest that time-series segmentation based on ACO identifies numerous segments and has a low segmentation cost. In a study by Shelokar, Jayaraman, and Kulkarni, 2004, an ACO meta-heuristic model was developed as a rule-based machine learning method known as the ant colony classifier system. It was discovered that this algorithm helps to address knowledge acquisition problems in terms of developing and maintaining the knowledge base by employing a simple mechanism. Another study proposed an ACO based classification rule mining approach called Ant-Miner. The performance of the proposed approach was compared with another classification algorithm on six datasets. The results indicate that the proposed approach is competitive with CN2 in terms of classification accuracy and the size of the rule list (Parpinelli, Lopes, and Freitas, 2002). Yu, Ni, and Zhao, 2013 introduced an ACO sampling (ACOSampling) approach in which feature selection is first applied to reduce the noise in the data. Then, the original training set is randomly divided into training and validation sets, where ACO is applied to filter the less informative majority samples. It was demonstrated that the proposed model outperforms other sampling approaches when applied on four skewed DNA microarray datasets using the SVM classifier.

The number of studies hybridising two population-based algorithms to take advantage of both and address their main shortcomings in solving various data mining issues is increasing. Nazir, Majid-Mirza, and Ali-Khan, 2014 proposed a hybrid PSO-GA model for gender classification using facial and clothing information. The model has been evaluated using real-world face image datasets and SVM. Results show that the proposed approach produced high classification accuracy equal to 98.3%. Ghamisi and Benediktsson, 2015 proposed a model for feature selection based on the hybridisation of GA and PSO to overcome PSO premature convergence and GA difficulty in finding the optimal solution. The results indicate that the proposed hybrid model is able to find the best feature efficiently. Holden and Freitas, 2007 proposed a hybrid ACO-PSO algorithm for classification, in which there is no need to convert the nominal attributes to numerical attributes at the preprocessing stage. The proposed algorithm has been evaluated on sixteen real-world datasets. Experimental results show that the proposed approach competitively outperforms other techniques. Another hybrid ACO-GA approach has been proposed for feature selection in protein function prediction (Nemati et al., 2009). More details on other hybrid models can be found in (Li, Wu, and Tan (2008), Shi et al. (2003), Kuo and Lin (2010), Liu et al. (2013), and Holden and Freitas (2005)).

In this section, a review of the applications of evolutionary computation and swarm intelligence techniques for their speed and global search and ability to solve data mining issues has been provided. The results show that GA and PSO continue to be the most popular algorithms in the field of computer science (Corne, Reynolds, and Bonabeau, 2012). It has been also found that there is a striking trend in the latest work on swarm intelligence techniques toward the creation of solutions to various data mining issues using hybrid algorithms. These hybrids involve either combining two meta-heuristic approaches or one meta-heuristic approach and a data mining preprocessing technique to solve issues like feature selection and parameter optimisation. This enables them to use the advantages of both while also addressing their main limitations. Moreover, a hybrid of two meta-heuristic techniques can be used to utilise each technique to optimise a different aspect of the solution in the optimisation problem in an attempt to create a more powerful swarm algorithm that can be used to solve a wider range of issues (Bergh and Engelbrecht (2004) and Muthiah, Rajkumar, and Rajkumar (2016)).

3.5 Summary

In summary, this chapter has presented the main concepts of swarm intelligence, highlighted its main advantages and disadvantages, and defined the main swarm-based algorithms. In addition, a review of previous studies of swarm intelligence in data mining has been provided. It has been found that swarm algorithms and population-based algorithms have many similarities, such as the initialisation of the swarm and the fitness function that is used to evaluate the solution found by each member in the swarm. Moreover, swarm intelligence techniques provide a fast and robust solution for search and optimisation problems that can be potentially used to address various data mining issues, such as the problem of class imbalance. This can be achieved by applying swarm-based algorithms by themselves or by combining these algorithms with non-swarm techniques like feature selection algorithms to increase the chances of even more accurate classification. Data mining optimisation problems are complex and require a large amount of computation to find the optimal solution. This is because datasets can be large and suffer from issues like class imbalance and high dimensionality, all of which require time and effort to improve classifier performance. There are standard methods for optimisation, such as grid search. However, these methods are impractical and time consuming, especially when dealing with large datasets. Instead, swarm intelligence techniques can efficiently solve the optimisation problem.

The review also indicates that GA and PSO remain the most widely used swarm intelligence algorithms in solving data mining issues. More work is needed to investigate the possibility of implementing other swarm-based techniques like SDS and DFO in solving data mining issues like class imbalance and feature selection. SDS is applicable for solving search and optimisation problems due to various characteristics, among them partial function evaluation, which provides a cost effective solution when conducting a search. Another important characteristic of SDS is the different recruitment strategies that can be employed as an alternative mechanism for achieving a balance between exploration and exploitation. Moreover, it can be used as a continuous and discrete optimiser. In terms of DFO, the simplicity of this swarm intelligence algorithm adds to its appeal when applied to complex search and optimisation problems with only one parameter to tune, which is Δ, as opposed to the presence of more parameters in several other swarm techniques. Moreover, it has less components and is easier to analyse, as DFO only uses the current positions of the population to determine future movements (i.e. it does not hold memory or velocity vectors as in PSO). Finally, neither SDS nor DFO have ever been used for solving the class imbalance problem. This is therefore the first time these algorithms are being used to improve classifier performance when learning from imbalanced datasets. SVM is used as a classification algorithm. It is a popular machine learning algorithm that has been widely used in real-world applications from different domains. Moreover, it has various advantages such as non-linear classification using kernels and high generalisation capabilities (Batuwita and Palade, 2013). Swarm intelligence based techniques can be used to optimise the kernel parameters efficiently and to achieve better performance on imbalanced datasets.

The next chapter presents a set of experiments that use existing techniques from the literature and combine them with swarm intelligence techniques like SDS or DFO to take the best of both in solving the class imbalance problem from different aspects while using SVM as classification algorithm.

Chapter 4

Experiments

This chapter presents the set of experiments that aimed to explore various solutions to the class imbalance problem using swarm intelligence techniques like SDS and DFO.

4.1 Experiment I: Class imbalance in a direct marketing dataset

Some studies have found that a combination of data level and algorithmic level solutions can lead to better model performance when dealing with imbalanced datasets (Burez and Poel (2009) and Longadge and Dongre (2013)). In this experiment, a model is proposed to solve imbalanced data using a Hybrid of Data-level and Algorithmic-level solutions (HybridDA). It involves oversampling the minority class using SMOTE to create a synthetic example of the minority class, random undersampling of the majority class , and optimising the C, γ and kernel type of SVM using a grid search. The proposed model performed competitively compared with other models on the same dataset. The dataset used in this experiment is real-world data collected from a Portuguese marketing campaign for bank-deposit subscriptions and is available from the UCI machine learning repository [1]. This dataset was collected in an attempt to convince clients to subscribe to a term deposit using phone calls as a means of reaching potential clients and improving the quality of Customer Relationship Management (CRM) in banks (Moro, Cortez, and Rita, 2014). It contains 21 attributes and 4119 examples (see Table 4.1 for the list of the attributes).

The next section describes the experiment setup.

[1] The dataset is available at http://mlr.cs.umass.edu/ml/datasets/Bank+Marketing.

TABLE 4.1: Attributes list

Attributes	Name	Description
1	Age	In integer
2	Job	Type of job
3	Marital status	Married, single, divorced or unknown
4	Default	Has credit
5	Balance	Average yearly balance
6	Housing	If there is a housing loan
7	Loan	Has a personal loan
8	Contact	Type of communication
9	Day	Last contact day of the month
10	Month	Last contact month of the year
11	Duration	Duration of last contact in second
12	Campaign	Number of contacts performed during this campaign and for this client
13	Pdays	Number of days passed since last campaign contact
14	Previous	Number of contacts performed for this client from previous campaigns
15	P-outcome	Previous outcome results
16	Emp-Var-rate	Employment-variation rate
17	Con-Price-index	Consumer price index
18	Cons-Conf-index	Consumer confidence index
19	Euribor3m	Euribor 3-month rate
20	No Of employed	Number of employees
21	Subscription	Label class Target: has the client subscribed

4.1.1 Experiments setup

In the direct marketing dataset, the target variable is ***"subscription"***, a binomial type of attribute that is set as the ***"label class"*** in the model, as shown in Table 4.1. From the meta data, it can be seen that the number of ***"subscribers"*** is dramatically below that of the ***"non-subscribers"***, which causes a class imbalance, as shown in Table 4.2.

The next section describes the data level and algorithmic level solutions

TABLE 4.2: The label class before balancing the instances

Label	**Number of instances**
Subscribers	451
Non-subscribers	3668

used to overcome the class imbalance issue found in the direct marketing dataset.

Data Level Solution

At the data level, there are several methods to deal with the imbalance in the dataset. The proposed model focused on balancing the data by oversampling the minority class and undersampling the majority class, using SMOTE and random undersampling respectively. To oversample the minority class, the model implemented SMOTE, with the parameters initialised as follows:

- Class was set to zero to detect the minority class automatically.
- Nearest neighbours was set to 5, which created synthetic instances from the 5 nearest neighbours [2].
- The percentage of instances to create was set to 458%. This was to oversample the number of subscribers to 2516 subscribers and balance the dataset. The majority class was then undersampled using random undersampling to 2550.
- The number of seeds used for sampling was set to 0 [3].

After the SMOTE algorithm was applied, it was necessary to randomise the instances, especially in some cases (e.g., when applying 10-folds cross validation where some folds will have too many positive instances or too many negative instances). After the randomisation process was applied, the majority class had to be randomly undersampled to balance the dataset further. When this was done, the dataset was balanced and the minority class was then only slightly smaller than the majority class (see Table 4.3).

[2] Based on recommendations from the literature, the number of nearest neighbours is set to five (Chawla et al. (2002) and Blagus and Lusa (2013)).

[3] Random number generator seed was set to 0. In other words, no seed was used by the random number generator.

TABLE 4.3: The label class after balancing the instances

Label	Number of instances
Subscribers	2516
Non-subscribers	2550

Algorithmic Level Solution

At the algorithmic level, solutions to learn from imbalanced datasets include adjusting the cost measurement to consider the class imbalance, or adjusting the probabilistic estimate at the tree leaf when using a decision tree. The HybridDA model uses SVM and a grid search to optimise the C, γ, and kernel type. As suggested by Hsu, Chang, and Lin, 2003, the range for C has been defined as [2^{-5},2^{15}] and the range for γ as [2^{-15},2^{3}]. Moreover, k-fold cross validation was used to calculate the classification accuracy. In this study, the value of k was equal to 10 and the dataset was partitioned equally into 10 sub samples, 9 of which were used for training and 1 for testing. The process was then repeated 10 times and the classification accuracies were averaged. The advantage of k-fold cross validation is that all instances are used for both training and validation, and each instance is used for validation exactly once. All possible combinations were evaluated and the optimised results evaluated by the grid search are shown in Table 4.4. A flowchart of the proposed system is shown in Fig 4.1.

TABLE 4.4: The optimised parameters combination using grid search

Parameter	Optimised value
Kernel	Poly
Gamma	4.800
C	3276.828

The next section presents the obtained experimental results and provides a comparison with other techniques from the literature on the same dataset.

4.1.2 Results

As previously mentioned, in cases of imbalanced datasets and the class of interest being the minority class, predictive accuracy is not the best performance indicator. In this experiment, the TPR, also known as the recall rate, was used as a performance measurement as the higher the recall rate, the

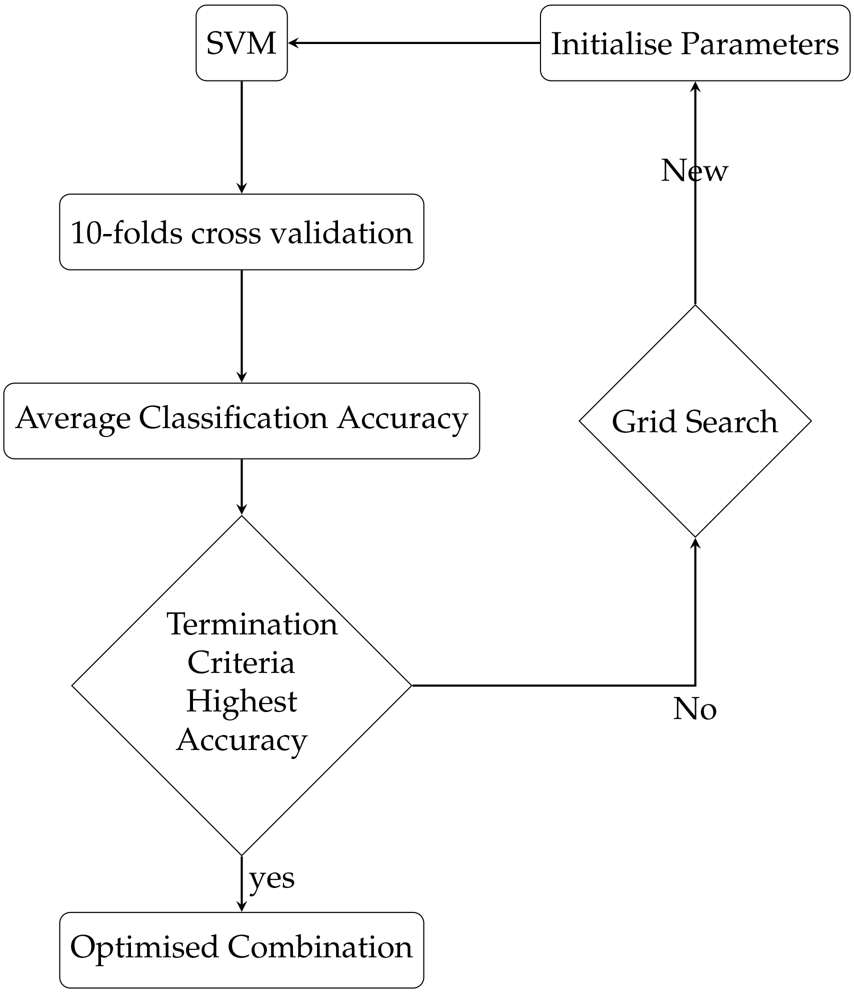
SVM
Initialise Parameters
New
10-folds cross validation
Average Classification Accuracy
Grid Search
Termination Criteria Highest Accuracy
No
yes
Optimised Combination

FIGURE 4.1: The proposed model flowchart

better the class of interest is classified. To complement TPR, TNR, the accuracy and AUC were also used (more details on evaluating the performance on imbalanced datasets can be found in Section 2.5). The proposed model presents an improved predictive model, as shown in Table 4.5.

TABLE 4.5: HybridDA results

Performance metric	**Value**
Acc	96.73%
TPR	97.93%
TNR	94.82%
AUC	0.98

As discussed in the next section, the proposed system outperforms previously reported results conducted on the dataset. In summary, this has been achieved by applying SMOTE, which creates more synthetic examples to learn from, undersampling the majority class to balance the dataset, and then combining this with the optimised values for C, γ and the kernel type in SVM.

Comparison with other techniques

There have been some relevant studies on various versions of the dataset used in this experiment. For example, Moro, Laureano, and Cortez, 2011 implemented the Cross Industry Standard Process for Data Mining (CRISP-DM) methodology on previous versions of the dataset. The CRISP-DM methodology allows the building of a data mining model in six non-grid phases that can be used in a business environment. The phases are business understanding, data understanding, data preparation, modelling, evaluation and deployment (Azevedo and Santos, 2008). The study implemented three classification methods using an R package for data mining: DT, NB and SVM. The algorithms have been evaluated using both the ROC and lift curve. The ROC curve and lift curve are visualisation tools used to show how good the classification model is. The ROC curve is a plot between the TPR on the y-axis and the FPR on the x-axis, in which the higher the curve the better the model. However, the lift curve shows the results gained with or without using the classification model. In the lift curve, the x-axis represents a sorted list of the population and the y-axis represents the TPR. The lift curve is helpful in business campaigns (Vuk and Curk (2006) and Zadrozny and Elkan (2001)). For example, in direct marketing it shows how likely it is to receive potential subscribers from the first 10% group of customers. According to the study by Moro, Laureano, and Cortez, 2011, SVM gives the best results with a AUC value and Area Under The Lift Curve (ALIFT) value equal to 0.9 and 0.8 respectively.

In a 2014 study, (Moro, Cortez, and Rita, 2014) used four classification methods: DT, Neural Networks (NN), SVM and LR. Two evaluation metrics were also used: AUC and ALIFT. The models were tested on the most recent contacts and NN performed the best with AUC=0.8 and ALIFT=0.7. In terms of simplicity, LR and DT produce a more understandable model while giving good results. However, SVM and NN are more flexible and have better learning capabilities, thus yielding better results. At the beginning of the above-mentioned study, a semi-automated approach was used to reduce the attributes from 150 to 22. This was done in two steps. In the first step, the attributes were analysed from a business perspective, and in the second step the forward selection method was implemented.

Vajiramedhin and Suebsing, 2014 proposed a model on the same dataset, focusing on using correlation-based feature subset selection algorithm and a dataset balancing technique. The balancing technique was used to make the dataset label equivalent by randomly selecting the dataset of each label equally. For the correlation-based feature subset selection algorithm for feature correlation measurements, the authors proposed a model that implements a C4.5 algorithm. The proposed model scored a high TPR of 92% and an ROC rate of 95%, when compared with other methods where balancing or feature selection was excluded.

Another model proposed by Feng, Zhang, and Liao, 2014 on the direct marketing dataset combines Bayes networks (BNs). The experiment started by combining two BNs, then three, then more than three. All average accuracies were compared and led to results improvement with average accuracies of 83%.

A study by Elsalamony, 2014 focused on increasing the effectiveness of the marketing campaign by finding the major attributes that affected the success of the phone call. The author compared four classification methods-Multilayer Perception Neural Network (MLPNN), Bayesian networks, LR, and C5.0- on the direct marketing dataset and found that C5.0 gave the best results, with the testing part scoring:

TABLE 4.6: The C5.0 classification results

Performance metric	Value
Acc	90.09%
TPR	59.06%
TNR	93.23%

The author also found that *"duration"* was the most important attribute from C5.0, MLPNN and LR, and that (age) was the most effective attribute from BN.

Another model that has been proposed by Bahnsen, Aouada, and Ottersten, 2015 is an example-dependent cost-sensitive decision-tree algorithm where the direct marketing dataset was used with two other datasets. In this model, the Cost-Sensitive Decision Tree (CSDT) was evaluated against the standard DT with three different cases: without pruning, with error-based pruning and with cost-sensitive pruning training. The three different tree algorithms were trained using the training, undersampling, cost-proportionate rejection sampling and cost-proportionate sampling dataset.

The results show that the proposed learning algorithm is the best performing for all three databases, with direct marketing dataset accuracy of 88.28% and an F-measure of 0.35. This model has been used further to measure cost savings. The summary of the comparison with other techniques is shown in Table 4.7.

TABLE 4.7: Summary of the results for previous models on the direct marketing dataset

Models	AUC	ALIFT	Acc	TPR	TNR	ROC
Moro, Laureano, and Cortez, 2011	0.938	0.887	NA	NA	NA	NA
Moro, Cortez, and Rita, 2014	0.8	0.7	NA	NA	NA	NA
Vajiramedhin and Suebsing, 2014	NA	NA	NA	NA	92.14%	95.6%
Feng, Zhang, and Liao, 2014	NA	NA	83%	NA	NA	NA
Elsalamony, 2014	NA	NA	90.09%	59.06%	93.23%	NA
Bahnsen, Aouada, and Ottersten, 2015	NA	NA	88.28%	NA	NA	NA
HybridDA	0.98	NA	96.73%	97.93%	94.82%	NA

4.1.3 Summary

This experiment proposed a new approach to address class imbalance by using a combination of both data-level and algorithmic-level solutions. One such imbalanced dataset was used in the experiment to evaluate the proposed method using this dataset, and different researchers have taken different approaches to building a classification model that predicts potential subscribers to a bank term deposit. While most research projects have compared different classification algorithms on the dataset, some focused on finding the most effective attribute by applying different classification algorithms. The experiment described in this section showed promising results when evaluated against previous work on the same dataset. This is likely to be caused by focusing on minimising the imbalance effect on the learning algorithm without affecting identification of the class of interest of customers who subscribed to the term deposit. SMOTE was chosen to oversample the minority class in order to create more synthetic examples, which is better than random oversampling (which causes over-fitting). Random undersampling was applied only to the majority class to balance the data further. To complement the sampling process, an algorithmic solution was applied by adjusting the misclassification cost, the kernel type, and the γ. The adjustment was made by applying the standard grid search to optimise the parameter combination. Therefore, the proposed model, HybridDA, uses a combination of data and algorithmic level solutions to handle the existing class imbalance in the dataset. The result of this hybridisation demonstrates a competitive performance. It is suggested that balancing a dataset using a more powerful random oversampling technique and combining it with random undersampling to address the class imbalance, then applying a cost-sensitive learning algorithm where the parameters are optimised, works well with imbalanced datasets. Given that the proposed model is not generalised, caution should be exercised when applying the system to other datasets. Thus, future work will focus on testing the suggested model on the large Portuguese marketing campaign dataset as well as on other datasets from different fields, while using additional techniques at the data level. Another future project will involve the use of meta- heuristic approaches to optimise the SVM kernel parameters. The purpose is to speed up the search and lower computational expenses.

4.2 Experiment II: SDS and undersampling

In this investigation, a set of experiments at the data level was conducted to compare a number of undersampling approaches that were applied to the same direct marketing campaigns of the Portuguese bank dataset used in Experiment 4.1. This was done to investigate a new approach for balancing a marketing dataset using a swarm intelligence technique, SDS, to undersample the majority class on the direct marketing dataset and apply a data level solution using a meta-heuristic search technique. SDS is a simple discrete optimiser in which partial function evaluation is used. Partial function evaluation allows agents to quickly have an opinion or a decision about the hypothesis (the investigated solution from the search space) without exhaustive search. The outcome of the novel application of this swarm based algorithm demonstrates promising results which encourage the possibility of undersampling a majority class by removing redundant data whilst protecting the useful data in the dataset. The next section outlines the experiment setup and presents the findings.

4.2.1 Experiment setup

As shown in Table 4.2, there is a difference between the number of subscribers, which is equal to 451 and that of non-subscribers which is equal to 3668. To balance the dataset, the model used SDS for undersampling the majority class and SMOTE to oversample the minority class. At the algorithmic level, the model made predictions using the SVM with the RBF kernel in which the γ set to 1.00 and the *C* set to 0.00. To prepare the dataset for the SVM, the following pre-processing steps were taken: all nominal values are converted to numerical; and all values are normalised to avoid the value scale difference among all attributes. For all the experiments, 10-folds cross validation was applied.

In order to oversample the minority class to 2000, SMOTE algorithm was used with the following configurations:

- Class was set to zero to detect the minority class automatically.
- Nearest neighbours was set to 5, which created synthetic instances from the 5 nearest neighbours [4].
- The percentage of instances to create was set to 345%. This was to oversample the minority class to 2006 subscribers. The majority class was then undersampled using SDS to 2000 to balance the dataset.
- The number of seeds used for sampling was set to 0.

The aim was to oversample the minority class in order to reach a comparable size with the undersampled majority class (see Table 4.8 for the size of the dataset before and after balancing). The next section illustrates the balancing techniques used in this work.

TABLE 4.8: The label class before and after balancing the dataset

	Number of instances in the dataset	
Label	**Imbalanced**	**Balanced**
Subscribers	451	2006
Non-subscribers	3668	2000

Balancing the Dataset

There are several methods to deal with the class imbalance problem in the dataset; the proposed model investigated balancing the dataset by applying two different approaches: undersampling the majority class and over-

[4] As mentioned in Experiment 4.1, the number of nearest neighbours was set to five based on the literature recommendations (Chawla et al. (2002) and Blagus and Lusa (2013)).

sampling the minority class, which was conducted using SMOTE. The undersampling process was performed by SDS, whose performance was then contrasted against random undersampling as well as undersampling with ED.

Applying SDS for undersampling

The initial experiment used SDS to undersample the majority class from 3668 to 2000 non-subscribers. In this experiment, 100 agents were used [5]. Initially, the model was selected from the search space (all non-subscribers) and the agents were set to find the closest match from the remaining items of the search space. Once a match or the most similar item was found, it was removed from the majority class with the aim of removing redundant data. Given that this process aims at reducing the size of the search space without removing useful data, removing the closest item to a randomly selected model discourages the deletion of useful data. This hypothesis was later validated (in section 4.2.3) when the spread and the central tendency of the data were investigated before and after the undersampling process (McCluskey and Lalkhen, 2007).

Following the initialisation phase where each agent was allocated to a hypothesis from the search space (a random non-subscriber), in the *test phase*, a randomly selected micro-feature (attribute) from the hypothesis was compared with the corresponding micro-feature of the model; if the randomly selected micro-feature of the hypothesis lay within a specific threshold (which will be discussed later) of the model's micro-feature, the agent was set to active, otherwise to inactive. This process is repeated for all agents.

In the next phase, the *diffusion phase*, a *passive recruitment* mode was applied where each inactive agent chose another agent and adopted the same hypothesis if the randomly selected agent was active. If the randomly selected agent was inactive, the selecting agent picked a random hypothesis (i.e. a random non-subscriber from the search space). This process was repeated for all inactive agents.

The cycle of test-diffusion was repeated 10 times, which is the best empirically chosen value, at the end of which a non-subscriber with the maximum number of active agents was removed from the search space and the model was moved to another list (e.g., model list). This guaranteed that while the

[5] The best empirically found value.

most similar item was removed from the search space, the model, which represents the deleted item, was kept and used later during the classification process. This process was repeated until the dataset was undersampled.

In brief, the process of picking a random non-subscriber as a model and deleting the most similar item was repeated until the size of the search space plus the model list was equal to the number required (i.e., 2000, which is close to the number of the oversampled minority class).

In the experiments reported in this work, three different thresholds, including 1.00, 0.50, and 0.00 were used and thus three different datasets of non-subscribers were generated, all sized 2000. As the input dataset was normalised and the range of values was between 0.00 and 1.00, the SDS algorithm with threshold of 1.00 randomly undersampled the data; threshold 0.00 looked for an exact micro-feature match from the model; and threshold 0.50 was a state between random and exact-match undersampling.

Applying euclidean distance for undersampling

Euclidean distance is a metric used to measure distances between n points in the space. Over the past years, this measure has been widely used for database dimensionality reductions (Keogh et al. (2001) and Beckmann, Ebecken, and Lima (2015)). Although it is a comprehensive metric and there is a high computational expense involved in it to the undersampling problem, it was used in this experiment as the mean to contrast with the proposed computationally cheaper swarm intelligence technique. ED was used to undersample the majority class; in each iteration, a model was picked randomly, then the ED of the model with each element in the search space was calculated; once all the distances had been calculated, the closest element to the model was removed. This process was repeated until the size of the search space was reduced to the number required (i.e. 2000 entries).

4.2.2 Results

In this experiment, various performance measurements were used: accuracy, sensitivity, specificity, Area Under the Curve (AUC), F-measure, and precision. The experimental results show that the new approach (i.e., a combination of SDS at threshold 0.00 to undersample the majority class, and SMOTE to oversample the minority class) achieved the best performance in terms of accuracy, specificity, F-measure, and precision, as shown in Table 4.9.

TABLE 4.9: Performance measurements comparison

Threshold	0.00	0.50	1.00	Euclidean Distance
Accuracy	**90.46**%	88.56%	88.56%	89.47%
Sensitivity	95.46%	96.06%	96.06%	**96.76**%
Specificity	**85.45**%	81.04%	81.04%	82.15%
AUC	0.959	0.96	0.96	**0.965**
F-measure	**90.93**%	89.41%	89.41%	90.91%
Precision	**86.82%**	83.67%	83.67%	84.48%

As shown in Table 4.9, the proposed model achieved higher accuracy because of the higher specificity. On the other hand, obtaining higher F-measure is attributable to the higher precision rate as opposed to the ED undersampling. However when using ED for undersampling, the results exhibited higher sensitivity and AUC which can be justified given the much higher computational expense; this claim is explored further in the next section along with a more in-depth discussion about SDS and the impact of the varying thresholds on the results. From the results, the proposed SDS undersampling process performed the undersampling and removed redundant data using the threshold 0.00 as a distance measurement to the model, with a better specificity rate as opposed to the ED. Threshold 0.00 indicates that the difference between each corresponding attribute is minimised as the dataset is normalised between 0 and 1. Thus, threshold 0.00 finds the closest corresponding attribute in the majority class. Also, SDS undersampling outperformed the ED undersampling in terms of speed and computation time. Moreover, the results reported in this experiment show that the proposed method can offer promising results when compared with previous work on the same dataset, as shown in Table 4.10.

TABLE 4.10: Results for previous models on the direct marketing dataset

Models	AUC	Accuracy	Sensitivity	Specificity
Moro, Laureano, and Cortez, 2011	0.938	NA	NA	NA
Moro, Cortez, and Rita, 2014	0.8	NA	NA	NA
Feng, Zhang, and Liao, 2014	NA	83%	NA	NA
Elsalamony, 2014	NA	90.09%	59.06%	**93.23**%
Bahnsen, Aouada, and Ottersten, 2015	NA	88.28%	NA	NA
Proposed Model	**0.959**	**90.46%**	**95.46%**	85.45%

4.2.3 Discussion

This section discusses the results reported in section 4.2.1 and section 4.2.2, and further investigates the use of SDS for undersampling and the benefit of the partial function evaluation in decreasing computational complexity. It also analyses the spread of data to explore whether there were changes and whether the proposed approach caused a removal of useful data from the majority class. Finally, it explores computational cost of the proposed approach.

Applying SDS to Imbalance Data

SDS presents itself as an effective tool to persistent problems encountered in search and optimisation. This is due to SDS's strong partial function evaluation feature which assists agents to explore the existing large search space and gather global knowledge without having to evaluate all the existing dimensions; the in-depth analysis of the dimensions occurs only once a viable solution is found, at which stage, agents explore further dimensions of plausible solutions. In this experiment, the agent points to a randomly selected micro-feature and compares it with the corresponding micro-feature of the model. Therefore, partial evaluation enables an agent to form a quick "opinion" about the quality of the investigated solution without exhaustive testing (i.e. complete attributes comparison) which potentially leads to increased computational complexity. However, using ED, the search for a close match to the model is more computationally expensive because of the complete (vs. partial) function evaluation that accompanies each evaluation.

In the experiments reported earlier, three SDS threshold values were evaluated. The activity of the agents in each of the three presented thresholds are illustrated in Fig 4.2. As expected, when the threshold is set to 1.00, all agents become active at the end of the first iteration, and thus stop communicating with other agents in the diffusion phase; when the threshold is set to 0.00, the algorithm only "settles" on an exact match, thus as shown in the figure, while around half of the agents are active, the other half searches for the closer match. The middle ground status of SDS when the threshold is set to 0.50, is also illustrated in the figure.

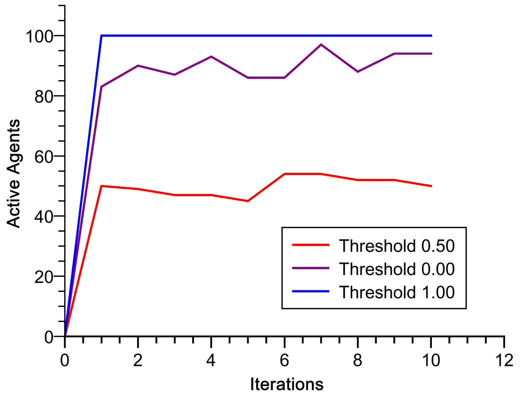

FIGURE 4.2: SDS agent's activity

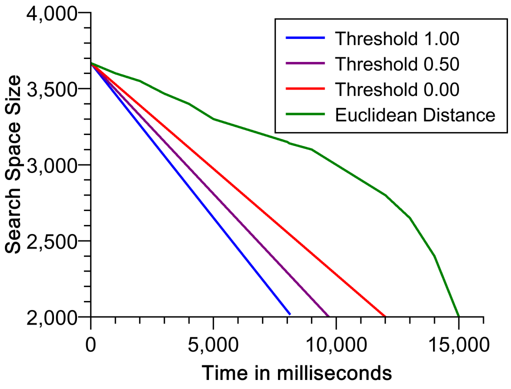

FIGURE 4.3: SDS & ED time comparison

SDS and the Spread of Data Analysis

Several metrics were used in the experiment to evaluate the quality of the undersampled data during the classification phase. However, the spread of data and the central tendency are two other important metrics for verifying that useful data was not removed during undersampling. These measurements were used to calculate the mean and the standard deviation of all data points. The difference between each undersampled dataset and the original dataset (before undersampling) is 0.01095 ± 0.00925 for SDS with threshold 0.00 and 0.00945 ± 0.00775 when using ED. The results highlight the lack of any significant change in the spread of data with the threshold set to 0.00 and when ED is used. This shows not only the success of the algorithm in keeping the useful data, but also the presence of redundant data in the dataset.

SDS and Computational Complexity

As stated before, due to the partial function evaluation feature of SDS, the computational cost of running SDS on a huge dataset is only dependant on the number of agents and the number of iterations. In the presented work, where the initial size of the majority class was 3668, having 100 agents performing 10 iterations, means that the agent population partially evaluates more than a quarter of the search space (i.e. 100 agents $\times$ 10 iterations $=$ 1000 micro-features evaluated). Over time, with the shrinkage of the search space – thanks to the removal of the redundant data – the algorithm's coverage increased to half. One possible approach, which is the subject of ongoing research, is to reduce the computational expense further by keeping the coverage of the swarm at a constant rate throughout the undersampling process.

Fig 4.3 shows the time taken for SDS in all three thresholds to undersample the data as well as the time taken when ED is used. As can be seen in the figure, the undersampling process with SDS demonstrates a linear time complexity throughout the undersampling process (where the search space shrinks but the number of agents and the iterations allowed are constant), thus exhibiting the time complexity of (n).

4.2.4 Summary

Various researchers have proposed advanced undersampling techniques to reduce the majority class samples without removing useful information. This work has proposed a swarm based undersampling approach that reduces the sizes of the majority class in a reliable yet cheap computational way, using the agents and partial evaluation of the majority instance, in which the individuals of the swarm move through the solution space in search of a solution that is close to the model. In the proposed method, the capability of SDS to perform majority class undersampling has been investigated on a real-world Portuguese bank dataset. The obtained results imply that SDS can be used as a good undersampling tool for class imbalance.

Future work includes the investigation of SDS on other imbalanced datasets, as well as comparison with other swarm intelligence techniques that have been applied to overcome the class imbalance issue. Another topic of ongoing research is the relationship between (and the impact of) the population size of the SDS and the coverage percentage of the dynamically shrinking search space of the dataset being undersampled.

4.3 Experiment III: Feature level duplication

Following the promising results of the use of SDS to perform undersampling of the majority class, the work was extended further to overcome the class imbalance problem on nine real-world datasets. This was combined with SMOTE to oversample the minority class. Moreover, to solve the problem at the algorithmic level SVM values *C* and γ were optimised using a grid search and 5-folds cross validations to train and test the classifier. This experiment initially aimed at investigating the impact of the duplication at dataset's feature-level (the role of duplications on each individual feature) on the undersampling process, the use of SDS as a way to address the feature-level duplications, and the possibility of making a recommendation on when to use the pro- posed approach. The next section describes the experiment setup.

4.3.1 Experiment setup

In the experiments conducted for this work, the first task is the application of SDS to undersample the majority class where the aim is to reduce the size of majority class (SDS's search space). The proposed model uses SDS to undersample the majority class to around 50%; in cases where the minority class instances need to be oversampled for balancing the distribution (to reach a comparable size with undersampled majority class) SMOTE is applied (this is to all datasets with minority class size less than 30%), with the following configurations:

- Class was set to zero to detect the minority class automatically.
- Nearest neighbours was set to 5, as this would create synthetic instances from the 5 nearest neighbours [6].
- The percentage of instances to create depended on the size of the undersampled majority class.
- The number of seeds used for the sampling was set to 0.

Table 4.11 shows in which of the datasets SMOTE was used to oversample the minority class. At the algorithmic level, the model used the SVM algorithm, where parameters such as C and γ for the RBF kernel were optimised using a grid search, in which the range for C was defined as $[2^{-5}, 2^{15}]$ and the range for γ as $[2^{-15}, 2^{3}]$ (Hsu, Chang, and Lin, 2003). The search was performed using 5-folds cross validation and multi-threading to run multiple processes at a time. The hybrid approach was then contrasted against random undersampling (along with SVM optimisation). To evaluate the proposed model, nine imbalanced datasets were used in this experiment. The datasets are available from the UCI machine learning repository, plus the Oil Spills dataset (Kubat, Holte, and Matwin, 1998). The datasets, all collected from real-world cases, vary greatly in their class distributions, sizes and features characteristics (continuous and discrete features). The full list of datasets used is shown in Table 4.11. In the experiments reported here, for the Abalone dataset, the model is applied on classes "9" versus "18"; for the Yeast dataset, the model

[6] Based on recommendations from the literature, the number of nearest neighbours is set to five (Chawla et al. (2002) and Blagus and Lusa (2013)).

was applied on the classes “CYT” versus “POX”; and for the Vehicle dataset, the class “Van” vs the others is used, in order to have a highly imbalanced dataset. Additionally, all datasets values were normalised; this is to avoid the value scale difference among all attributes, while instances with missing values are removed.

Applying SDS to undersample the majority class instances:

In this experiment, the number of SDS agents was empirically set to be half of the search spaces (half the number of instances) and a quarter of the SDS population size was set for the number of iterations to undersample the majority class. Initially, a model (an instance from the majority class) was randomly selected from the search space (the entire majority class instances) and the agents were set to find the closest match (an instance) from the remaining items of the search space. Once a match or the most similar item was found, it was removed from the majority class with the aim of removing redundant data. Given that this process aimed at undersampling the majority class without removing useful information, removing the closest match to a randomly selected model prevented the deletion of useful information from the search space. The initialisation, test, and diffusion phases of SDS were expanded in order to shed more light on how SDS is adapted for the purpose of undersampling. In the *initialisation phase* each agent was assigned to a hypothesis from the search space (i.e., a random instance number from the majority class). Subsequently, in the *test phase*, a randomly selected micro-feature (*one of the attributes of the instance*) was compared against the corresponding micro-feature of the model (i.e., the corresponding attribute of the model); if the randomly selected micro-feature of the hypothesis was within the threshold of the model’s micro-feature, the agent was set to active, otherwise to inactive (threshold vector calculation is described in the next section). This process was repeated for all the agents, after which all agents were either active or inactive. Once the status of all the agents was determined in the test phase, the next phase started. In the *diffusion phase*, each inactive agent randomly picked another agent; if the randomly selected agent was active, its hypothesis (i.e., instance number) was shared with the inactive agent; otherwise, the selecting agent picked a random hypothesis (a random instance number) from the search space. The cycle of test-diffusion phases was repeated equal to the number of iterations allowed. The instance which attracted the largest number of agents was labelled as the “*closest match*” and thus removed from the search space. The model was then transferred to another “models list”.

TABLE 4.11: Summary of datasets used in this experiment

Dataset	No. of instances	Missing Values	Minority Class	Majority Class	Distribution	Continuous Features	Discrete Features	SMOTE
Oil Spills	937	No	41	896	0.04 : 0.96	49	0	Yes
Yeast	483	No	20	463	0.04 : 0.96	8	0	Yes
Abalone	731	No	42	689	0.06 : 0.94	7	1	Yes
Vehicle	846	No	199	647	0.23 : 0.77	18	0	Yes
Breast Cancer	699	Yes	241	458	0.34 : 0.66	9	0	No
Bank Marketing	4119	No	451	3668	0.11 : 0.89	10	10	Yes
Thoracic Surgery	470	No	70	400	0.15 : 0.85	3	13	Yes
Ionosphere	351	No	126	225	0.35 : 0.65	34	0	No
Hepatitis	155	Yes	32	132	0.21 : 0.79	6	13	Yes

In the next step, another model was randomly chosen from the remaining instances, its closest match was found and removed from the search space, and the new model was then added to the "models list". Once the sum of the size of the models list and the remaining search space reached the desired number (i.e., when the majority class was downsized), the undersampling process was terminated..

Feature Dependent Threshold Vector

There are two types of features or attributes in the datasets (i.e., continuous and discrete). Depending on their types, the feature's threshold was calculated accordingly and separately (for each feature). For continuous features, the thresholds were found by calculating the median values (excluding the zeros) of the difference between the values of the features. Following the same analogy for the discrete features, the threshold was calculated using the following equation:

$$\tau_i = \frac{1}{n-1} \tag{4.1}$$

where τ_i is the threshold of feature i, and n is the number of discrete values. Therefore, τ returns the value of the 'gap' between each neighbouring discrete value.

Using the method described for calculating the threshold vector, $\vec{\tau}$, the algorithm can perform an evaluation as to whether any two selected values from the same feature can be considered "adjacent" values. Therefore, using $\vec{\tau}$ during the test phase for each agent, the proximity of the instances can be partially evaluated (through each individual feature comparison). It has been shown in many other applications that, after several iterations, SDS is capable of finding the closest match, which can then be removed as part of the undersampling process. By applying this method, the undersampling is performed at feature level to find the closest match for the randomly picked model from the remaining majority instances. The next section presents the obtained results.

4.3.2 Results

This section summarises the results of using SDS-SVM to undersample the majority classes of all the datasets used. In order to fairly evaluate the performance of the proposed model, various performance measurements have been used: G-mean, F-measure, AUC, accuracy, sensitivity, and specificity (more details on performance metrics can be found in section 2.5). The results achieved in this experiment have been compared with the hybrid approach where random undersampling and optimised SVM are used (RND-SVM), as well as some of the best previous models on the same datasets.

In Table 4.12, the results of SVM classification after random undersampling (RND-SVM) and SDS undersampling (SDS-SVM) are reported and contrasted against other methods from the existing literature. The results in Table 4.12 show that, in most cases, SDS-SVM outperformed RND-SVM. In order to investigate the reason behind this difference of performance, the redundancy at instance and feature levels are discussed in section 4.3.3.

The current literature demonstrates that there have been other relevant experiments on the same datasets (see Table 4.12); for example, Zhang and Li, 2013 implemented a Positive biased Nearest Neighbour algorithm (PNN) on real-world datasets including the Oil Spills and Vehicle datasets. The proposed model gave the best results when compared against the *k*-NN algorithm, other sampling methods such as SMOTE, and the general method for making a classifier cost sensitive, known as MetaCost. The model was evaluated using AUC, and PNN gave the best results, with AUC equal to 0.847 for the Oil Spills dataset and 0.983 for the Vehicle dataset. In both instances, SDS-SVM outperformed PNN. In another study, Guo and Viktor, 2004 proposed a new model called DataBoost-IM. The model was evaluated using F-measure, G-mean, and Accuracy on seventeen imbalanced datasets, including datasets used in this experiment. The proposed model scored well on highly imbalanced datasets in terms of the F-measure, and was comparable with (in some instances higher than) other models with regards to G-mean and Accuracy. This algorithm was outperformed by the model proposed in this experiment.

The next section explores the behaviour of the algorithm proposed for undersampling in this experiment. The aim is to provide some insight into where it would be recommended to use SDS-SVM.

4.3.3 Discussion

In this section, the proposed approach is analysed and the agent behaviour is investigated.

TABLE 4.12: Results for the datasets

	G-mean	**AUC**	**F-measure**	**Accuracy**	**Sensitivity**	**Specificity**
Oil Spills						
RND-SVM	35.27%	0.648	69.61%	56.27%	**100.00%**	12.44%
SDS-SVM	**98.74%**	**0.999**	**98.74%**	**98.74%**	99.58%	97.92%
DataBoost-IM [1]	67.70%	NA	55.0%	96.60%	46.30%	**98.90%**
PNN [2]	NA	0.847	NA	NA	NA	NA
Yeast						
RND-SVM	**91.43%**	**0.969**	**90.86%**	91.26%	**94.00%**	88.94%
SDS-SVM	90.33%	0.965	89.74%	90.11%	94.00%	86.81%
DataBoost-IM [1]	66.9%	NA	58.0%	**97.3%**	45.00%	**99.90%**
GSVM-RU [3]	NA	0.845	68.8%	NA	NA	NA
Abalone						
RND-SVM	88.69%	0.951	89.29%	88.62%	**91.43%**	88.00%
SDS-SVM	**89.83%**	**0.957**	**89.39%**	**89.77%**	91.11%	88.57%
GSVM-RU [3]	86.5%	NA	60.4%	NA	NA	NA
DataBoost-IM [1]	61.1%	NA	45.0%	94.6%	38.0%	**98.1%**
Vehicle						
RND-SVM	**98.45%**	0.995	**98.46%**	**98.45%**	99.06%	97.85%
SDS-SVM	**98.45%**	**0.999**	**98.46%**	**98.45%**	**99.37%**	97.54%
DataBoost-IM [1]	95.7%	NA	93.7%	97.0%	93.4%	**98.1%**
PNN [2]	NA	0.983	NA	NA	NA	NA
Breast Cancer						
RND-SVM	**97.70%**	**0.996**	**97.71%**	**97.70%**	**98.33%**	97.08%
SDS-SVM	95.81%	0.972	95.77%	95.83%	97.07%	94.58%
DataBoost-IM [1]	96.40%	NA	95.2%	96.70%	95.40%	**97.3%**
Bank Marketing						
RND-SVM	**92.91%**	0.972	**93.43%**	93.06%	97.05%	88.95%
SDS-SVM	90.96%	0.966	91.46%	91.07%	94.04%	88.00%
HybridDA [4]	NA	**0.98**	NA	**96.73%**	**97.93%**	**94.82%**
Thoracic Surgery						
RND-SVM	71.69%	0.755	70.51%	71.82%	68.88%	74.63%
SDS-SVM	**73.51%**	**0.767**	**72.78%**	**73.59%**	**71.94%**	**75.12%**
Boosted SVM [5]	65.7%	NA	NA	NA	60.00%	72.00%
Ionosphere						
RND-SVM	94.01%	0.979	93.77%	94.15%	**97.69%**	90.48%
SDS-SVM	**95.32%**	**0.986**	**95.27%**	**95.31%**	96.03	94.62%
CSB2 [6]	93.00%	NA	89.7%	82.90%	96.5%	89.7%
DataBoost-IM [1]	92.3%	NA	91.2%	94.0%	87.3%	**97.7%**
Hepatitis						
RND-SVM	91.02%	0.960	90.62%	91.21%	**93.55%**	88.57%
SDS-SVM	**91.98%**	**0.963**	**91.47%**	**92.42%**	87.10%	**97.14%**
CSB2 [6]	80.9%	NA	63.4%	80.6%	81.3%	80.5%
DataBoost-IM [1]	76.2%	NA	62.6%	83.8%	65.6%	88.6%

1. Guo and Viktor, 2004
2. Zhang and Li, 2013
3. Tang et al., 2009
4. Alhakbani and Al-Rifaie, 2016b
5. Zięba et al., 2014
6. Ting, 2000

Analysing instance and feature levels redundancy:

It is intuitive that in the case of a dataset with a high level of duplication, picking a randomly selected instance and removing it as part of the undersampling process is less likely to cause the removal of important information. This hypothesis was clearly demonstrated with two of the datasets used (i.e. Yeast, and Breast Cancer) where there were duplications at instance level, making all the features of some samples identical to some others. Table 4.13 (No. of Duplicates) shows the duplications (percentage of duplicate instances) for both of these datasets.

While this justifies the outperformance of RND-SVM over SDS-SVM, it neither justifies the outperformance of RND-SVM in Bank Marketing nor offers a strong reason for the outperformance of SDS-SVM in all the remaining datasets. For this reason, redundancy at the feature level was explored, as shown in Table 4.13 where the number of repetitions in each feature was calculated and then the median, mean, and standard deviation of all the feature repetitions were taken into account. Considering these figures, a link can be established between a high level of similarity (duplications) between the features (e.g., combination of median (or average) and standard deviation) and the performance of SDS-SVM. For instance, in the case of the Oil Spills dataset, the median repetition of 81.47% and the standard deviation of 32.61% indicate a varying level of duplications across various features, leading to the superior performance of SDS-SVM, which partially evaluates the instances. In terms of Bank Marketing, where there is no duplication of instances, there is a high level of duplication at feature level with a median of 99.72% and standard deviation of only 4.14% which justifies the good performance of RND-SVM. In all other cases (Oil Spills, Abalone, Vehicle, Thoracic Surgery, Ionosphere and Hepatitis), where feature-level duplication is not high, and there are no large standard deviations (causing a larger level of oscillations), SDS-SVM is a recommended method to use. As can be seen, feature-level duplication analysis also caters for instance-level duplication analysis, thus providing a better insight into which of the two algorithms to use.

Investigating agents behaviour:

The SDS algorithm adopted for the purpose of undersampling is responsive towards feature-level duplications, and when there are many duplications at feature level, the number of active agents is higher; this is illustrated in

TABLE 4.13: Instance and feature duplication rates

	Instance Level Figures	Features Level Figures			
Datasets	**No. of Duplicates**	**Median**	**Average**	**Deviation**	**Best Model**
Oil Spills	0	81.47%	68.65%	32.61	**SDS-SVM**
Yeast	25 (5.39%)	91.25%	92.54%	4.61	RND-SVM
Abalone	0	61.62%	58.01%	33.57	**SDS-SVM**
Vehicle	0	94.12%	88.46%	13.43	**SDS-SVM**
Breast Cancer	231 (50.43%)	95.19%	95.26%	0.2	RND-SVM
Bank Marketing	0	99.72%	98.48%	4.18	RND-SVM
Thoracic Surgery	0	99.50%	94.90%	10.63	**SDS-SVM**
Ionosphere	0	4.88%	7.67%	6.3	**SDS-SVM**
Hepatitis	0	97.01%	81.30%	26.15	**SDS-SVM**

the graphs of Fig 4.4, where the Bank dataset with the high feature-level duplications is shown (on the left) as opposed to the Ionosphere dataset (on the right) where the feature-level duplication is much lower (with a median of 4.88% and the standard deviation of 6.3). The oscillating behaviour of the population's activity is attributed to the micro-feature evaluation of each of the agents. In other words, if an agent picks a certain micro-feature and becomes active, it is likely that other agents are attracted to the hypothesis of that agent, thus adopting the same hypothesis (but a randomly selected and likely different micro-feature); if the newly selected micro-feature is not within the threshold, this would lead to agent inactivity for the next cycle. This mechanism assists the agents to maintain their activities only when a hypothesis is (in most of its micro-feature selections) within the calculated threshold. One interesting feature of the algorithm is that a high activity level of the population does not always correspond to convergence to a single instance. While this in itself is a useful feature for identifying more (than one) similar instances, work still needs to be done in the future on this characteristic. Also, it would be worthwhile to explore whether each trial (i.e., removal of one similar instance from the search space) could be terminated depending on the activity level of the populations.

Search Space Coverage:

In the case of the SDS algorithm and its partial function evaluation, while SDS might visit each instance "briefly" (i.e. checking one of a few features), it does not run a greedy comparison on all of the instance's features. Therefore, the agent aims to "form an idea" before spending further computational

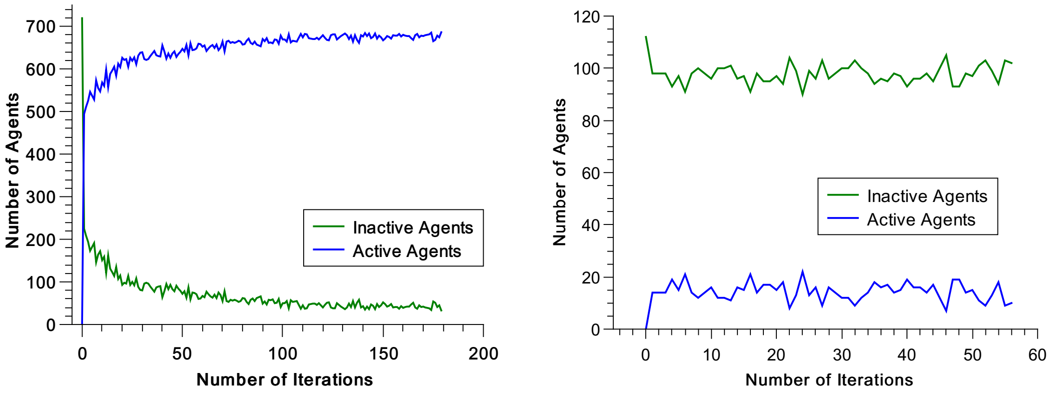

FIGURE 4.4: Convergence of agents over the iterations allowed for the Bank (left) and Ionosphere (right) datasets

time (by itself or by attracting other agents). This behaviour of the agents can be summarised in two sets of experiments: the first would be to explore the percentage of the instances visited by the SDS agents, and the second to calculate the percentage of all features visited from the whole of the dataset (i.e. all the features of all the instances). The results of these two experiments are shown in the graphs of Fig 4.5. It is shown that while the empirically chosen values for the number of agents and the iterations suffice in visiting the instances at least once, not all the features are (or need to be) visited.

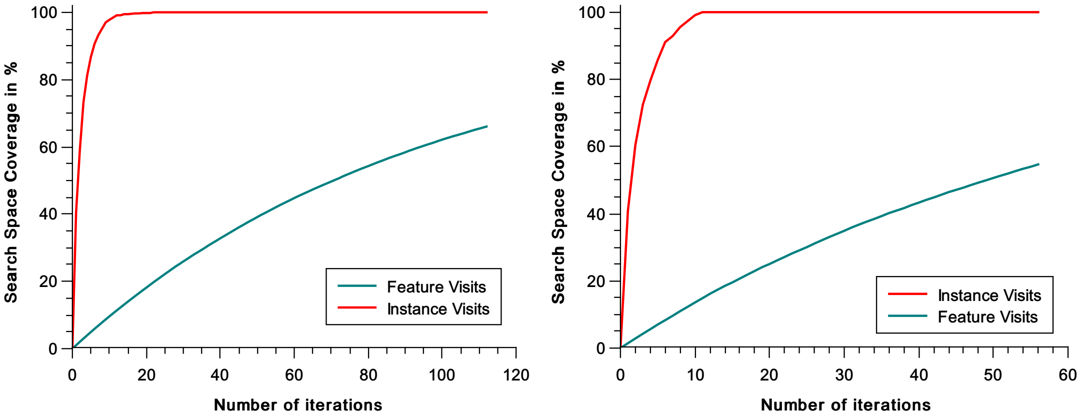

FIGURE 4.5: Search space coverage for the Oil Spills (left) and Ionosphere (right) datasets

In another experiment, the frequency of agents visiting each feature was explored and the distribution of agents' exploration capability in the search space was investigated, with the ultimate goal of finding the closest match. For this purpose, three datasets with varying degrees of feature-level duplication were chosen, and the results are illustrated in Fig 4.6. For instance, in the case of the Bank Marketing dataset, where duplication is very high, with a median of 99.72% and standard deviation of 4.14%, it was shown that agents converged to the closest matches (showing themselves as stripe of white lines) while in the case of the Oil Spills and Ionosphere, the agents presence was distributed across the search space. Also, in terms of the Oil Spills dataset, instances in position 450 to 550 attracted more agent visits, which is attributable to their similarity to the model that is located at position 471.

4.3.4 Summary

This experiment proposed a model which uses a swarm intelligence based algorithm, SDS, which is assigned to perform the undersampling of the majority classes in imbalanced datasets. This work presents an analysis of both instance- and feature-level redundancies and establishes a link between the feature-level duplications and the role of the feature-level undersampling mechanism. This analysis is accompanied by an investigation of the behaviour of the agents through their activity level during the undersampling process. It is shown that the agents' activity is directly proportional to the level of redundancy in the datasets (not only at the instance level, but more importantly, at the feature level). Another investigation carried out in this project regards the ability of the algorithm to comprehensively explore the search space without having to greedily investigate all features of all instances in the dataset. As part of future research, various coverage percentages could be explored, thus associating the coverage percentage with the termination criteria. This might shed light on the "bare essential" coverage needed before removing an instance. There is ongoing study being conducted on the link between the agents' activity level and the termination criteria as well as the possibility of removing more than one instance from the dataset when the agents share a "similar interest" in multiple instances.

4.4 Experiment IV: Feature selection using SDS

As shown in Experiments 4.2 and 4.3, SDS has performed well in solving the class imbalance issue. In this experiment, the use of SDS was extended further to tackle the class imbalance problem by performing feature selection. Feature selection is a technique applicable to datasets that experience high dimensionality problems. It aims to select a subset of features, which enables a classifier to maximise its performance. It is a vital step for a number

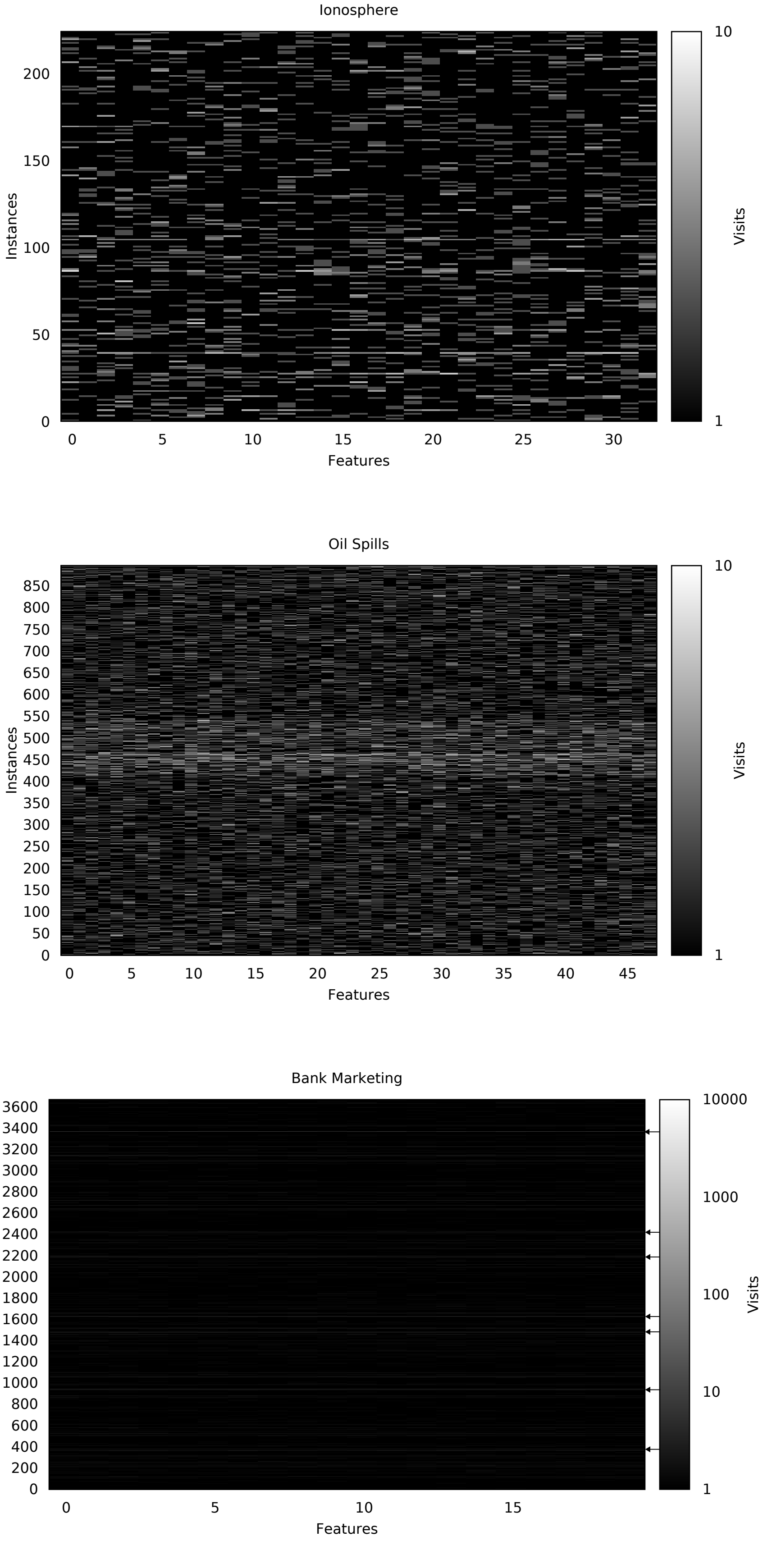

FIGURE 4.6: Frequency of visiting individual features in three datasets

of classification tasks like micro array-based classification and text classification. As a result, effective feature selection methods are needed to speed up the search and improve the predictive accuracy. SDS has been applied to various areas of search and optimisation but has not been applied to feature selection problems in imbalanced datasets classification yet.

In terms of feature reductions, there are some similarities between feature selection algorithms and swarm algorithms because both have factors or parameters that influence the solution being applied. This is the case especially for the wrapper variation of the feature selection algorithms because the algorithms determine the validity or strength of the subset based on the performance of the algorithm, which is very similar to dynamically updating the swarming behaviour of the swarm intelligence algorithms. The method introduced in this experiment uses SDS to select the most relevant feature subset for the classification task. In this algorithm, SDS is adapted to find a suitable feature subset. Moreover, SVM is used as a classifier to evaluate the predictive accuracy of the agent. The proposed method exhibits a statistically significant superior performance when compared with the performance of the classifier without the SDS-powered feature selection. Additionally, the results have been compared with other methods from the literature over nine datasets. It is shown that the proposed SDS based feature selection (SDS-FS) offers a competitive performance compared to other methods on datasets with a feature size greater than 10. The behaviour of the proposed algorithm has been investigated in the context of global exploration and local exploitation. The next section describes the experiment setup.

4.4.1 Experiment setup

In this experiment, eight real-world datasets were used from the UCI machine learning repository as well as the Small Round Blood Cells dataset (SRBCT), which contains information on 83 samples and 2308 genes (Khan et al., 2001). These datasets vary in terms of the number of features, and they have been widely used as benchmarks to compare the performance of different feature selection methods in the literature. Table 4.14 describes the datasets used to evaluate the proposed method in terms of number of features, number of instances, and number of classes. Feature scaling was applied to the dataset to improve the classification accuracy of the learning algorithm and prevent distance calculation difficulties. In general, the range of each feature value was scaled to [0,1] range.

TABLE 4.14: Summary of datasets used in these experiments

Dataset	No of features	No of instances	Classes
Ionosphere	34	351	2
Sonar	60	208	2
Wine	13	175	3
Bupa	6	345	2
Breast Cancer (WDBC)	32	569	2
Vowel	10	990	11
Glass	10	214	6
Lung Cancer	56	32	3
SRBCT	2308	83	4

SDS for feature selection

This section provides a description of the proposed method that employs SDS for feature selection. The motivation for this method is the use of agents' direct communication to perform feature subset evaluation in a way that effectively improves the classification model's overall performance and decreases the training time.

In these experiments, 100 agents were set to perform the search for a feature subset of size d (where d = half of the original number of features in the dataset), and the number of iterations allowed was 50. These values were the suggested empirical values. The aim of this study was to find the best d features out of N, the total number of features. The accuracy of the classifier was taken as a fitness function.

Initially, each agent was assigned to a combination of feature subset (i.e., hypothesis) from the search space (i.e., all possible combinations of features). It should be noted that each agent used an *independent* random split in the dataset to form the two subsets of training and testing, 80% and 20% of the dataset respectively. The hypothesis was a binary string that represents a features subset within the subset size. In this string if a bit was 1, the corresponding feature was included and if 0, it was not.

In the *test phase*, agents' activities were determined based on the accuracy of the classifier that was calculated in the fitness function. In this phase, an agent selected another random agent and compared the predictive accuracy of them both. If the selecting agent's accuracy was higher than the random agent's, the selecting agent was set to active; otherwise, the selecting agent was set to inactive. This process was repeated for all agents to determine their status. Following this, the next phase, called the diffusion phase, commenced.

In the *diffusion phase*, both inactive agents and active agents randomly picked another agent. In the case of an inactive agent randomly picking another agent, if the randomly selected agent was active, its offset hypothesis (feature subset) was shared with the inactive agent; if inactive, the selecting agent picked a random hypothesis (feature subset) from the search space (all possible combinations of features within the subset size). In the offsetting, one feature was removed randomly (by changing the 1 to 0) and another was added randomly (by changing 0 to 1), thus preserving the subset size. Moreover, when an active agent picked another active agent that maintains the same hypothesis (feature subset), the selecting agent was set to inactive and was assigned to a random hypothesis. This freed up the agents, improved their diversity and increased the algorithm's ability to search widely throughout the search space. This cycle of test and diffusion was repeated equal to the iterations allowed. See Algorithm 10.

Fitness function

The purpose of feature selection is to find a particular subset of feature by applying certain fitness evaluation criteria to find a subset which has a better classification accuracy. SVM has been applied to several classification tasks due to its speed and high accuracy (Zhu, Liu, and Yu, 2002). To evaluate the feature subset, this model used the RBF kernel for SVM, which is able to deal with high dimensional datasets (Hsu, Chang, and Lin (2003) and Lin and Lin (2003)). Moreover, predictive accuracy was used as a performance metric as shown in Eq 2.5, in section 2.5, where the goal was to find a subset with the maximum classification accuracy. In this model, a train/test split (80% train, 20% test) was used to assess the performance of the feature subset on unseen dataset using SVM (each agent splits that data randomly for two sets: one for training and the other for testing). This is a common approach in splitting the datasets (Chang et al., 2010).

Algorithm 10 SDS algorithm for feature selection

```
Initialisation phase
Assign agents to random hypotheses with inactive states
while less than iterations allowed do
    #Evaluation Phase
    for all agents do
        Evaluate the fitness value
        Find the maximum fitness value
    #Test Phase
    for all agents do
        if Agent's fitness >random agent's fitness then
            Set agent as active
        else
            Set agent as inactive
    #Diffusion Phase
    for all agents do
        if agent is inactive then
            Select a random agent
            if selected agent is active then
                Copy its hypothesis & offset it
                Calculate the fitness value
            else
                Pick a random hypothesis
                Calculate the fitness value
        else agent is active
            Select a random agent
            if rand agent is active & shares hypothesis then
                Set the selecting agent to inactive
                Assign a new hypothesis
                Calculate the fitness value
```

4.4.2 Results

Results I: Presence and lack of feature selection

Table 4.15 summarises the best results of applying SDS as a feature selection method. It shows the accuracy of SVM without feature selection and the accuracy of SVM with SDS-FS on the datasets. As shown in the table, the use of SDS-FS was found to improve predictive accuracy for all datasets. For example, for the Sonar dataset, which has 60 features, the classification accuracy increased from 69.04% to 97.61%. Moreover, the model performed well on a larger number of features. For example, for the SRBCT dataset, which has 2308 features, the classification accuracy increased from 29.41% to 94.11%. Similar improvements in the classification accuracy can be observed in the rest of the datasets.

SVM, when learning from an imbalanced dataset produces a model that is biased toward the majority class. Therefore, it gives poor results when trained on an imbalanced dataset and without applying any solution to combat the class imbalance as seen in Table 4.15. The use of feature selection on imbalanced datasets that suffers from high dimensionality was found to improve the SVM classification accuracy for all the datasets by decreasing the degree of class overlap. Thus, the selected subset of features, which discriminates between the classes, allows SVM to perform well on imbalanced datasets using the default parameters (without tuning the kernel parameters C and γ). These results also indicate the possibility of improving the classifier performance further through the use of algorithmic level solutions and data level solutions while selecting a feature subset. As a result, the proposed model which uses SDS to perform feature selection, demonstrates the ability to reduce the dimensionality and improve classification accuracy.

In order to conduct the statistical analysis measuring the presence of any significant difference in the performance of the algorithms with and without SDS-powered feature selection, a t-test was deployed. This test was applied using the outcome of all the trials (30 runs) on each experiment. Through this statistical test, it was shown that the proposed feature selection technique offers a statistically significant superior performance over all the datasets used.

TABLE 4.15: Accuracy of SVM with and without feature selection

Dataset	Original No of features	SVM without feature selection	SVM with SDS-FS No of features	 Accuracy
Ionosphere	34	87.32%	17	**100%**
Sonar	60	69.04%	30	**97.61%**
Wine	13	94%	6	**100%**
Bupa	6	50.72%	3	**79.71%**
WDBC	32	92.98%	16	**100%**
Vowel	10	68.18%	5	**75.25%**
Glass	10	69.67%	5	**100%**
Lung Cancer	56	57.14%	28	**71.42%**
SRBCT	2308	29.41%	1154	**94.11%**

Results II: Comparison with other techniques

This part aims to compare the performance of the proposed method with other techniques from the literature.

Table 4.16 compares the results of the proposed model with other methods on the same datasets from the literature. This includes PSO-SVM (Tu et al., 2007), binary bare bones PSO (BPSO) (Zhang et al., 2015), PSO+SVM with feature selection (Lin et al., 2008), mr^2PSO (Unler, Murat, and Chinnam, 2011), IFS with feature selection and GA+SVM (Liu et al., 2011). As seen in Table 4.16, the proposed method excels in comparison to other techniques when applied to datasets with a dimensionality more than 10. For example, in the Ionosphere dataset, SDS-FS is outperformed other methods in terms of both maximising accuracy and reducing the feature size. However, in datasets with a dimensionality of less than 10, the SDS-FS method was outperformed by other methods. For example, in the Vowel dataset PSO+SVM with feature selection outperformed the rest with an accuracy equal to 100%. Moreover, for the Glass dataset BPSO (Zhang et al., 2015), it scored the same accuracy with lower number of features (feature subset size = 3) as opposed to SDS-FS, in which the size of the feature subset was 5.

The results indicate that SDS-FS is applicable to feature selection on high dimensional data. It achieves dimensionality reduction (to half of the feature size) and improves classification accuracy in a more effective way when applied to datasets with a feature size larger than or equal to 10. More experiments on other large datasets are needed before any generalisations can be made on this claim.

Moreover, for completeness, Table 4.17 compares the results of the proposed model with other non-evolutionary computation techniques on the same datasets from the literature. This includes sequential forward search (SFS) (Tu et al., 2007), maximum relevance-minimum multicollinearity (MR-mMC), minimal redundancy maximum relevance (mRMR) and mutual information based feature selection (MIFS) (Senawi, Wei, and Billings, 2017). As shown in Table 4.17, other than the Vowel dataset when processed with the SFS algorithm, the proposed SDS based model excels all other non-evolutionary computation techniques. When comparing to non-evolutionary techniques, it is also clear that SDS-FS is applicable to feature selection on high dimensional data.

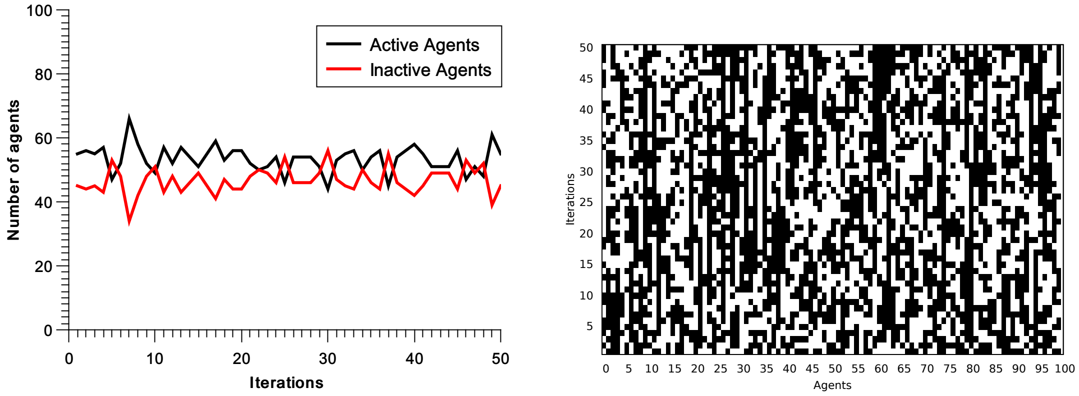

FIGURE 4.7: Ionosphere dataset. Left: agents' activity over the iterations; right: detailed view of the individual agent's activities over the iterations.

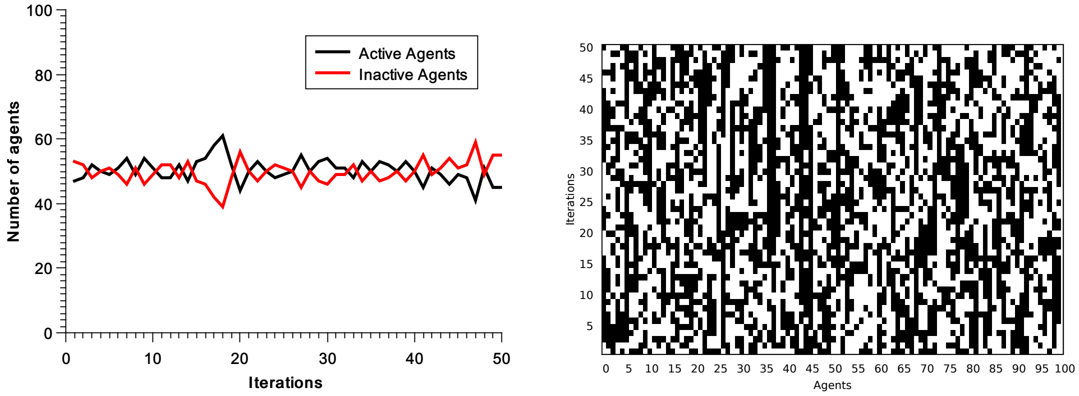

FIGURE 4.8: Vowel dataset. Left: agents' activity over the iterations; right: detailed view of the individual agent's activities over the iterations.

TABLE 4.16: Classification accuracy comparison of SDS-FS and other evolutionary computation techniques

	Algorithm	PSO-SVM [1]		BPSO [2]		PSO+SVM+FS [3]		mr^2PSO [4]		IFS+FS [5]		GA+SVM [5]		SDS-FS	
Dataset	**Dimension**	d	Acc	d	Acc	d	Acc	d	Acc	d	Acc	d	Acc	d	Acc
Ionosphere	34	15	97.33%	11.4	96.21%	21±3	99.01%	6	94.92%	-	-	-	-	17	**100%**
Sonar	60	34	96.15%	28.2	96.08%	37±4.75	96.26%	15	88.15%	25 ±0.97	93.70%	31.0 ±1.212	91.60%	30	**97.61%**
Wine	13	8	**100%**	7.5	99.45%	8 ±1.56	**100%**	6	99.72%	-	-	-	-	6	**100%**
Bupa	6	-	-	-	-	4±0.88	**82.05%**	-	-	-	-	-	-	3	79.71%
WDBC	32	13	95.61%	12.9	98.36%	-	-	3	96.64%	13 ±1.3	99.40%	17.3 ±0.991	98.90%	16	**100%**
Vowel	10	7	99.49%	8	99.7%	7 ±0.95	**100%**	-	-	-	-	-	-	5	75.25%
Glass	10	-	-	3	**100%**	5±1	85.26%	5	79.7%	-	-	-	-	5	**100%**
Lung Cancer	56	-	-	-	-	-	-	-	-	-	-	-	-	28	**71.42%**

1. Tu et al., 2007
2. Zhang et al., 2015
3. Lin et al., 2008
4. Unler, Murat, and Chinnam, 2011
5. Liu et al., 2011

TABLE 4.17: Classification accuracy comparison of SDS-FS and other non-evolutionary computation techniques

	Algorithm	**mRMR** [1]		**MIFS** [1]		**MRmMC** [1]		**SFS** [2]		**FS-SFS** [3]		**DFPA** [4]		**Filter** [3]		**REF** [3]		**SDS-FS**	
Dataset	**Dimension**	*d*	Acc	*d*	Acc	*d*	Acc	*d*	Acc	*d*	Acc	*d*	Acc	*d*	Acc	*d*	Acc	*d*	Acc
Ionosphere	34	-	-	-	-	-	-	14	90.88%	10	92.0%	10	92.6%	10	92.0%	10	92.7%	17	**100%**
Sonar	60	30	76.85%	30	73.86%	30	77.48%	24	89.90%									30	**97.61%**
Wine	13	-	-	-	-	-	-	8	95.51%	-	-	-	-	-	-	-	-	6	**100%**
Bupa	6	-	-	-	-	-	-	-	-	4-5	70.2%	5	67.1%	5	61.7%	5	69.7	3	**79.71%**
WDBC	32	-	-	-	-	-	-	15	92.00%	15	92.9%	15	95.4%	15	92.9%	15	95.4%	16	**100%**
Vowel	10	5	61.83%	5	60.53%	5	59.34%	8	**99.70%**	-	-	-	-	-	-	-	-	5	78.82%
Glass	10	5	60.79%	5	54.22%	5	59.13%	-	-	4	93.8%	5	91.9%	5	90.5%	5	87.1%	5	**100%**
Lung Cancer	56	-		- -		- -		- -		- -		- -		- -		- -		-28	**71.42%**

[1] Senawi, Wei, and Billings, 2017
[2] Tu et al., 2007
[3] Liu and Zheng, 2006
[4] Mao, 2004

4.4.3 Discussion

The purpose of global exploration is to discover a wider segment of the search space in the hope of discovering other solutions. This requires that the search be diversified in order to circumvent being restricted within a local optimum. On the other hand, local exploitation explores a limited area of the search space (which surrounds the agent) in the hope of improving the solution that already exists.

As of yet, there is no generally accepted agreement about which of the two has greater advantages. For a search to be successful, there needs to be a balance between global exploration and local exploitation (Črepinšek, Liu, and Mernik, 2013). For example, a hybrid Chaos PSO (CPSO) model has been proposed. The proposed algorithm balances between global exploration and local exploitation by introducing an Adaptive Inertia Weight Factor (AIWF). After comparison with other results, it was found that the proposed CPSO improved the search quality and diversity (Liu et al., 2005). In this experiment, a combination of an *offset* and *context sensitive recruitment strategy* was used to balance between local exploitation and global exploration.

In the offset, an inactive agent copies an active agent's hypothesis, in which a randomly selected feature is removed from the subset and a randomly selected one is added instead (i.e. changing 1 to 0 and 0 to 1, in the string representing the feature subsets). This focuses more on the search space surrounding the current solution which will improve the algorithm's local exploitation. Moreover, a context sensitive recruitment strategy is used to free up active agents (those sharing the same hypothesis with another selected agent) to increase their ability to explore other hypotheses far from the current one, a process designed to improve the algorithm's diversity and global exploration. By applying both mechanisms, a balance between global exploration and local exploitation is achieved.

This balance is reflected in the agents' activity level in each iteration, as shown in Figs 4.7 and 4.8. As described above, given the agents' communication strategy and the rule determining their activity status (in the test phase), at any given point during the optimisation process, approximately half of the agent population is active and the other half inactive. The plots in Figs 4.7-left and 4.8-left illustrate the number of active and inactive agents oscillating around the middle. The heatmaps shown on Figs 4.7-right and 4.8-right provide a detailed account of each agent's activity in each iteration. Using these heatmaps, it can be shown that, on average, over the lifetime of the agents (i.e., 50 iterations), each individual agent is active $54.50 \pm 13.94\%$

and $51.24 \pm 15.16\%$ of the time when processing the Ionosphere and Vowel datasets respectively.

Further investigation of the figures representing the heatmaps shows that, despite the balanced activity of the agents, they exhibit resistance in changing their activity status. In other words, an active agent strives to stay active more often than not, paving the way for further "exploitation". On the other hand, once an agent "loses" its activity in one of the iterations, it starts its "exploration" of the search space before it settles on returning to exploitation. This switching behaviour is shown to be less than 50% in the Ionosphere and Vowels datasets. The switching tendencies in agents' activity are $32.64 \pm 10.21\%$ and $32.08 \pm 10.61\%$ respectively, demonstrating that in approximately 60% of cases, agents "dedicate" their time to perform further exploration (or exploitation), before they switch their status to exploitation (or exploration).

Another illustration has been created to shed light on the behaviour of inactive agents during the diffusion phase. Fig 4.9 illustrates the behaviour of inactive agents in the Vowel dataset as to the proportion of each inactive agent picking an active agent randomly (and sharing the hypothesis) and the instances where the inactive agents which fail to pick active agents (and picking a random hypothesis from the search space); the first boosts exploitation and the latter furthers the exploration of the search space.

Fig 4.10 shows the improvements made on the two sample datasets over the iterations, and two heatmaps in Fig 4.11 visualise the identification of alternative feature combination as the optimisation progresses across the relevant datasets, thus enhancing the accuracy with the newly identified feature subsets. As shown in the figures, there are instances where a difference in the identification of a different feature subset does not contribute to an increased accuracy.

Feature selection bias

Feature selection is very important since it plays a very crucial role in reducing the dimensionality of a dataset. It increases the speed of the classification process, simplifies the learned classifier, and improves and maintains the performance of the classification. However, this preprocessing step faces the challenge of feature selection bias, where the whole set of data is used

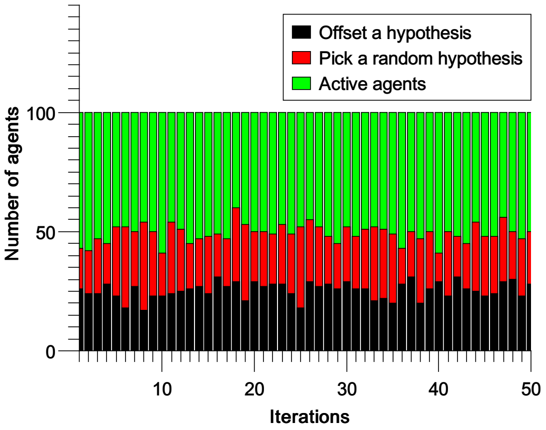

FIGURE 4.9: Behaviour of inactive agents during the diffusion phase in the Vowel dataset

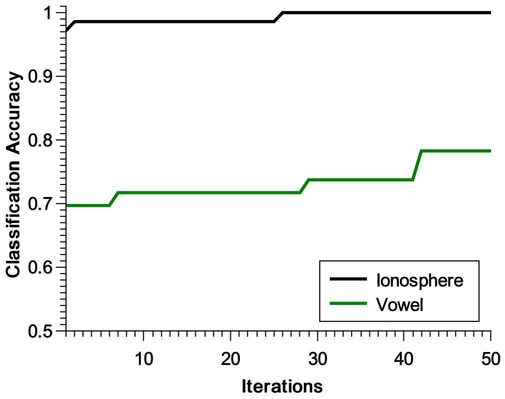

FIGURE 4.10: Classification accuracies over the iterations in the Ionosphere and Vowel datasets

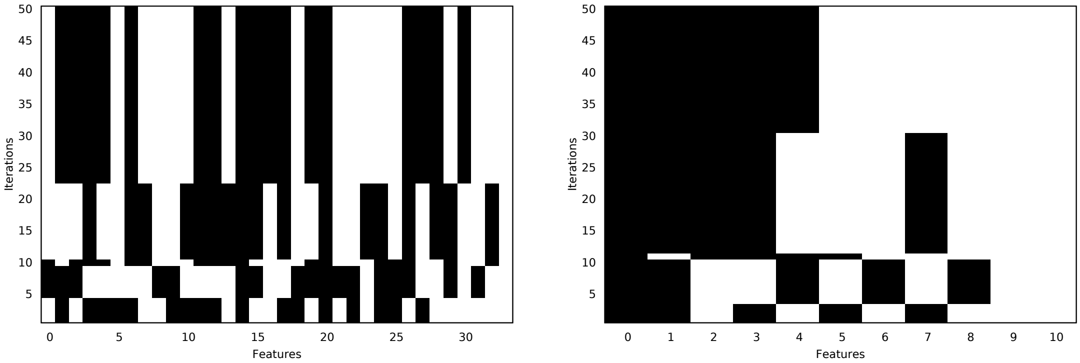

FIGURE 4.11: Features selected from the dataset over the iterations (black represents the inclusion of the features in the feature list, and white represents the removal of the features). Left: Ionosphere dataset; right: Vowel dataset.

within the feature selection process and no test data is unseen (Singhi and Liu (2006) and Krawczuk and Łukaszuk (2016)). This bias can negatively affect the model's ability to predict on new datasets. It has been also found that feature selection bias has less negative effects in classification when compared to regression because of the dissimilarity of the two tasks (Singhi and Liu, 2006).

Several studies have investigated the feature selection bias and suggested solutions to eliminate the bias (Krawczuk and Łukaszuk, 2016). Ambroise and McLachlan, 2002 proposed two of these: external cross validation and the bootstrap approach. Tran et al., 2016 proposed a model that uses a PSO algorithm known as PSO-LSRG, which contains a fast *local search* together with a *gbest* mechanism that resets automatically to improve the performance of the PSO in conducting feature selection. This is because resetting the *gbest* leads to the swarming search for solutions with smaller sizes, and the *local search* increases the classification performance further. Ultimately, PSO-LSRG can put into utilisation the two mechanisms in such a manner that the numbers of features are reduced while increasing performance. When compared to PSO, PSO with *gbest* resetting or local search only, the proposed model selects the smallest number of features when there is no feature selection bias.

Since swarm intelligence techniques like PSO are highly stochastic approaches which require the performance of multiple runs aimed at testing overall accuracy and finding the optimal feature subset, cross validation is computationally expensive and time consuming. Therefore, designing an efficient cross validation for evolutionary computations multiple runs is difficult. One way is to use two loops for cross validation, in which an inner cross validation loop is used to perform feature selection and training and an outer loop to calculate the classifier's performance (Tran et al., 2016). In this work, the datasets were uniquely divided, for each agent with a candidate solution, into two datasets for training and testing (i.e. each agent has its 80/20 train-test split) . This was to evaluate the accuracy of each feature subset in an attempt to eliminate the feature selection bias by directing the test to be independent of the training process for each agent.

4.4.4 Summary

This experiment proposes an algorithm that performs feature selection to improve the classification accuracy over nine datasets. The introduced approach implements a swarm intelligence technique, SDS, which benefits from agents' direct communication to find the optimal feature subset. The aim is to navigate a very large search space, which contains all the possible combinations of features $\binom{n}{m}$ where n is the number of original features in the dataset and m is the reduced number of features.

The proposed method provides a balance between local exploitation and global exploration by iterating through the test and diffusion phases as well as combining the *offsetting* and *context sensitive recruitment strategy*. The accuracy of the proposed method has been compared with the accuracy of the classifier when all features are used. The results show that the proposed method offers *statistically significant* improvements in the classification accuracy over the experiments without feature reductions. Moreover, the method has been compared with other hybrid algorithms on the same datasets. It has been demonstrated that the proposed SDS-FS works well with high dimensional datasets where the number of features is more than 10.

The current study does not aim at finding the optimal algorithm-related parameter (population and the number of iterations) and provides proof of principle of the performance of the proposed algorithm with an empirically suggested parameter set as a starting point for researchers. Finding the optimal parameter values is the topic of ongoing research. Another topic for future study is to find the optimal size for the final feature subset. Moreover, coupling the presented technique with simultaneous optimisation of the classifier's RBF kernel parameters, C and γ is currently being investigated.

4.5 Experiment V: DFO and parameter optimisation

This experiment investigates the possibility of implementing DFO in optimising SVM's parameters like C and γ for higher accuracy classification models. This is to perform cost sensitive learning to improve the classifier's performance on imbalanced data. The use of the swarming behaviour of the flies and their diversity in the search space in conducting cost sensitive learning are investigated on eight real-world datasets. The proposed algorithm has been compared with other techniques to optimise the classifier's parameters, including PSO, grid search, and random search, which is used as a control algorithm. The results demonstrate a statistically significant outperformance of the proposed optimisation technique over other techniques on the same datasets. The next section outlines the experiment setup.

4.5.1 Experiment setup

In this experiment, DFO was used to search for the optimal kernel parameters: C and γ. In this model, F-measure was deployed as an evaluation metric and the performance of DFO was compared with that of other parameter optimisation techniques to find the optimal kernel values over a set of benchmark datasets (more details on the performance metric in Chapter 2, Section 2.5). In order to evaluate the performance of the proposed technique, eight real-world datasets are used and available from the UCI machine learning repository [7]. These datasets are imbalanced and they vary in size and class distribution. Moreover, they have been widely used as benchmarks to compare the performance of various methods in the literature. Table 4.18 provides a description of the datasets used. In this experiment, the Abalone datasets for the class "9" versus "18" and for the Vehicle dataset, the model is applied on the class "Van" vs the others. Moreover, normalisation was applied to the datasets to scale each feature value to a [0,1] range, and instances with missing values were removed.

TABLE 4.18: Dataset list

Dataset	Minority Class	Majority Class	Attributes
Vehicle	199	647	18
Sonar	97	111	60
Ionosphere	34	126	34
WDBC	212	357	32
Abalone	42	689	8
Hepatitis	32	123	19
German credit	300	700	20
Breast Cancer	241	458	9

Furthermore, to make predictions on new data valid, a train/test split was used, in which 80% of the dataset was used for training and 20% was used for testing. The advantages of train/test split are that the optimised C and γ are evaluated on unseen dataset. As the datasets are imbalanced, F-measure was used as a fitness value for SVM, in which the goal was to find the C and γ which would give the maximum F-measure.

[7] The datasets are available at: http://archive.ics.uci.edu/ml/.

DFO for parameters optimisation

In this experiment, 50 flies were set to optimise the SVM's parameters, in which the range for C was defined as [2^{-5},2^{15}] and the range for γ was defined as [2^{-15},2^{3}] based on Hsu, Chang, and Lin, 2003. The iterations allowed were equal to 10. The number of the flies and the total iterations allowed are the best empirically chosen values. At the *initialisation phase*, each fly was assigned randomly to two values, with the first value being for C and the second for γ; by using these values the fly's fitness, the F-measure, is calculated. The fitness value was stored for each fly to find the best neighbouring fly and the best fly in the whole swarm. At every iteration, the components of the position vector were independently updated at the *update phase*, considering the components vector for the best neighbouring fly and the components vector for the best fly in the whole swarm. It also considered if the random number, r, that was generated from the uniform distribution on the range [0,1], is less than the disturbance threshold, Δ. In the experiment, the Δ was empirically equal to 0.5, which meant that 50% of the flies' components were randomly initialised to new positions in the search space. This enhanced the diversity of the algorithm and provided a balance between exploration and exploitation. In order to ensure that the performance of the algorithm was not solely due to the disturbance mechanism, a control algorithm (random algorithm) was also applied to the problem and the results are reported for comparison.

4.5.2 Results

Table 4.19 summarises the results of applying DFO as an optimisation algorithm and compares them with other methods on the same datasets. This includes PSO, grid search, and random search. As shown in the table, the use of DFO was found to improve the F-measure for all datasets and the proposed model outperformed other techniques on the same datasets. For example, for the Ionosphere dataset, the F-measure increased from 94.52%, as obtained by the PSO, to 98.59%. Similar improvements in the F-measure can be seen in the rest of the datasets. As a result, the proposed model which uses DFO to optimise the SVM kernel's parameters C and γ, demonstrates the ability to improve the classifier performance on imbalanced datasets.

As Fig 4.12 illustrates, while the other techniques exhibit varying performance over different datasets, DFO is shown to provide a consistent outperformance over all datasets. Given the importance of conducting a statistical analysis to measure the presence of any significant difference in the performance of the proposed model and the other techniques including PSO, grid search and random search, the t-test was applied. This statistical significance test was applied using the outcome of all the trials (30 runs) on each experiment. The results show that, the F-measure difference is significant at 5% level. The result of this test indicates that the proposed optimisation technique offers a statistically significant improvement in the classifier's performance on the imbalanced datasets when compared to the other techniques.

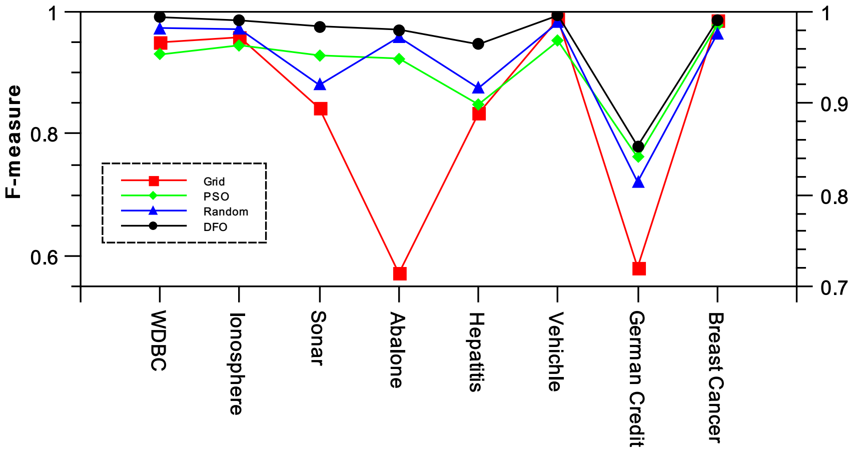

FIGURE 4.12: Comparison of F-measure on all datasets

Impact of Disturbance Threshold

The disturbance mechanism in DFO provides a stable independent convergence throughout the optimisation process. It also maintains a balance between exploration and exploitation. At the update phase, the Δ is the only adjustable parameter controlling the diversity of the algorithm that needs to be set. A suitable value for this parameter depends on the size of the swarm, the number of iterations and the size of the search space. Therefore, further work needs to be done to find a theoretically suitable value for this parameter. In this experiment, Δ was empirically set to 0.5, which allowed for an enhanced diversity of the population in covering the search space, as well as the ability to escape local optima.

TABLE 4.19: Performance measurements comparison of DFO-SVM and other techniques

Dataset	Method	Accuracy	Sensitivity	Specificity	F-measure	AUC
WDBC	PSO	92.98%	93.33%	92.75%	93.00%	0.93
	Grid	94.73%	90.69%	97.18%	95.00%	0.93
	Random	97.36%	93.87%	100%	97.35%	0.96
	DFO	99.12%	98%	100%	**99.12%**	0.99
Sonar	PSO	92.85%	90.90%	100%	92.86%	0.92
	Grid	87.71%	76.19%	95.14%	84.21%	0.823
	Random	88.90%	92.30%	81.25%	88.00%	0.86
	DFO	97.61%	96.42%	100%	**97.63%**	0.98
Ionosphere	PSO	94.36%	92.30%	100%	94.52%	0.96
	Grid	97.14%	95.83%	97.83%	95.83%	0.95
	Random	97.18%	100%	91.30%	97.15%	0.95
	DFO	98.59%	97.87%	100%	**98.59%**	0.98
Abalone	PSO	93.87%	30.76%	100%	92.35%	0.65
	Grid	97.95%	40.00%	100%	57.14%	0.88
	Random	96.59%	37.50%	100%	95.85%	0.68
	DFO	97.27%	62.50%	99.28%	**97.09%**	0.80
Hepatitis	PSO	87.50%	33.33%	100%	84.82%	0.66
	Grid	87.50%	83.33%	90.00%	83.33%	0.83
	Random	87.50%	50.00%	92.85%	87.50%	0.71
	DFO	93.75%	100%	93.33%	**94.68%**	0.96
Vehicle	PSO	95.29%	86.36%	98.41%	95.21%	0.92
	Grid	98.22%	98.43%	97.62%	98.81%	0.99
	Random	98.24%	95.35%	99.21%	98.23%	0.97
	DFO	99.41%	97.50%	100%	**99.40%**	0.98
German Credit	PSO	76.00%	54.90%	83.22%	76.14%	0.69
	Grid	79.33%	45.74%	94.66%	58.11%	0.83
	Random	73.50%	48.28%	82.39%	72.11%	0.65
	DFO	78.00%	65.07%	83.94%	**78.00%**	0.74
Breast Cancer	PSO	97.81%	97.77%	97.82%	97.81%	0.97
	Grid	99.25%	97.94%	100%	98.56%	0.96
	Random	96.34%	94.00%	97.70%	96.34%	0.95
	DFO	98.54%	98.70%	98.88%	**98.59%**	0.98

As stated previously, a random algorithm was included in the comparison as a control algorithm to ensure the DFO's performance was not solely attributable to its disturbance mechanism and that the coupled mechanisms of forming and breaking the swarm, together, gave rise to the performance of the algorithm. Equally, in order to demonstrate the impact of the absence or reduction of diversity (induced through the disturbance mechanism), another control algorithm with a small disturbance threshold ($\Delta = 0.001$) is proposed. Fig 4.13 illustrates that the sole presence of diversity or the lack of it,

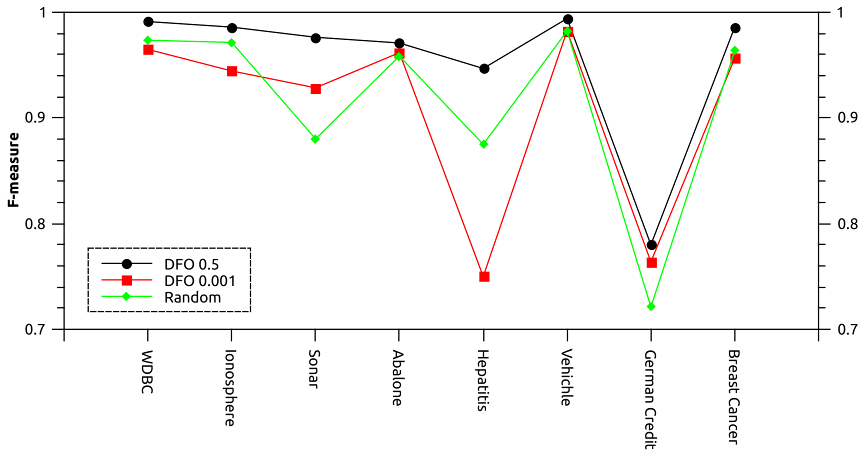

FIGURE 4.13: Negative impact of reducing the disturbance threshold to $\Delta = 0.001$

negatively impacts the performance of the algorithm.

4.5.3 Summary

Class imbalance is a major problem in machine learning. This experiment investigated the use of DFO to optimise the RBF kernel's parameters to improve the classifier performance without changing the distribution of the dataset by applying data level solutions such as oversampling or undersampling the dataset. The proposed method had a better, *statistically significant* performance when compared to other techniques on all datasets. Moreover, the simplicity of this swarm intelligence algorithm adds to its appeal when applied to complex search and optimisation problems as it has only one parameter to tune in contrast with several other swarm and evolutionary computation techniques, which have more. Future work is needed to compare the performance of DFO with that of other swarm and evolutionary computation techniques in larger datasets.

4.6 Experiment VI: Feature selection and parameters tuning using DFO

In Experiment 4.5, DFO was successfully applied to tune SVM parameters. Due to this, it was extended further to perform simultaneous feature selection and parameters tuning. The purpose was to reduce dimensionality and overcome the class imbalance problem in various applications such as medical diagnosis. In this experiment, a model using DFO was proposed to tune the kernel's parameters whilst finding a subset of features. To evaluate the proposed model, two microarray datasets were used: Small Round Blue Cell Tumours (SRBCT) and Leukemia MLL datasets. The classification accuracy of the developed model was compared with that of other models from the literature. The results indicate that the proposed approach outperforms various techniques from the literature. Moreover, the proposed model was evaluated using F-macro, an evaluation metric for imbalanced datasets, and the results indicate that the model is applicable to imbalanced datasets.

4.6.1 Experiment setup

This work developed a DFO based approach, DFO-SVM, for feature subset selection and kernel parameter optimisation. To evaluate the proposed model, two multi-class microarray high dimensional datasets were used, as shown in Table 4.20. The two datasets were the SRBCT dataset (Khan et al., 2001) and the MLL dataset (Armstrong et al., 2002). These multi-class high dimensional datasets are widely used as benchmarks to compare the performance of various methods in the literature (Li and Yin (2013) and Sharma and Paliwal (2008)). The relative frequencies of each class in both datasets are:

- SRBCT datasets: contains 83 instances 29 of which are instances of Ewing's sarcoma (ES) (34.9%), 25 are instances of rhabdomyosarcoma (RMS) (30.1%), 18 are instances of neuroblastoma (NB) (21.7%) and 11 are instances of Burkitt's lymphoma (BL) (13.3%).
- MLL dataset: contains 72 instances 24 of which are instances of acute lymphoblastic leukemia (ALL) (33.33%), 28 are acute myeloid leukemia (AML) (38.88%) and 20 are instances of mixed-lineage leukemia gene (MLL) (27.77%).

The model used multi-class SVM, One-vs-the-Rest (OvR), in which one classifier is fitted per class and each classifier is fitted against other classes. As the datasets suffer from high dimensionality the RBF kernel is used as shown in Eq 2.3.

At the preprocessing stage, feature scaling was applied to the datasets to standardise the range of values to [0,1]. This is also known as normalisation and is an important preprocessing step to avoid numerical difficulties in the calculation. In this experiment, 10-folds cross validation was used.

The approach consisted of two phases: filter and wrapper. In the first phase, a filtering algorithm was used to rank the genes. In this experiment, *information gain* was used to find the top 50 genes (see Chapter 2, section 2.4). As these genes are usually highly correlated, they were further reduced in the second phase in which DFO was used to find a subset of features whilst optimising the RBF kernel parameters, C and γ. This phase applied a wrapper approach using a meta-heuristic search technique in which the goal was to find a subset of features and optimise the kernel's parameters while improving the classifier performance. The next section presents the two steps that were used to apply the filter approach, optimise the kernel's parameters and perform feature selection.

Step I: Information gain

In this experiment, information gain was used first to rank the top 50 genes. Numeric features were discretised using the Mean- Entropy based discretisation method because it has been found to be more effective when classifying gene expression data (Li, Liu, and Wong (2003), Liu, Li, and Wong (2002), and Li, Zhang, and Ogihara (2004)).

TABLE 4.20: Dataset list

Dataset	Features	Instances	Classes
SRBCT	2308	83	4
MLL	12582	72	3

Step II: DFO for feature selection and parameters optimisation

In this experiment, 50 flies were used to find a subset of features while optimising the RBF kernel's parameters, in which the range for C and γ were defined as [2^{-5},2^{15}] and [2^{-15},2^{3}] respectively (Hsu, Chang, and Lin, 2003) and the value of each bit for feature selection ranged between 0 and 1. At the initialisation phase, each fly was assigned randomly to a vector of size D, (where $D = 52$), in which the first two values , $\mathrm{P_C}$ and P_γ, were for C and γ and the remaining 50, $\mathrm{F_2}$ to $\mathrm{F_{D\text{-}2}}$, were for feature selection as shown in Fig 4.14. If the value of $\mathrm{F_d}$ was more than 0.5 then its corresponding feature is chosen when building the classification model. Conversely, if the value of $\mathrm{F_d}$ was less than 0.5 then its corresponding feature was not chosen when building the classification model. Using these values, the fitness function was calculated.

TABLE 4.21: DFO parameters

DFO parameters	**Value**
Number of flies	50
Number of iterations	30
Disturbance threshold	0.05

P_C	P_γ	F_2	...	F_d	...	$F_{D\text{-}2}$	$F_{D\text{-}1}$

FIGURE 4.14: Vector representatives

After all flies were assigned a position in the D-dimensional space, each fly's fitness was calculated and the maximum fitness was set as the global best fitness and labelled as the global best fly. At the update phase, the components of the vector were independently updated based on the global best fitness (the global best fly) and the best neighbour (the neighbour with a higher fitness). Note that if the position of the fly updated to beyond its lower or upper bound, the fly's component was assigned to the value of its corresponding lower or upper bound. This update happened whenever a random number, r, that is generated from the uniform distribution on the range [0,1], was less than Δ.

However, if the value of r was more than the value of Δ, the flies were randomly initialised to new positions in the search space (resetting the position). In this experiment the value of Δ was equal to 0.05. This meant that 5% of each of the fly's components would explore a new location and would increase the algorithm's overall diversity.

The number of iterations allowed in this experiment was set to 30 [8], as shown in Table 4.21. After the final iteration, the final fitness value and size of the feature subset were given a score based on Eq 4.2. The obtained feature subset was further reduced in the next generation, and the results (fitness value and size of feature subset) were given a score again. The goal was to minimise the score. If the new score was higher than the old score, then the optimisation process stopped and the final results obtained. Otherwise, DFO was run again to further reduce the feature subset.

[8] The number of the flies and the total iterations allowed are the best empirically chosen values.

$$f(a,b) = \lambda(1-a) + (1-\lambda)\frac{b}{50} \tag{4.2}$$

where a is the optimised classification evaluation metric, b is the number of the selected feature and λ is a weight parameter. In this experiment, λ was set to 0.25, giving the reduction of the number of features more weight equal to 0.75. Allowing domain experts to extract knowledge regarding relevant features and conduct further microarray studies such as statistical pattern recognition.

The experiment was conducted using two different fitness functions. The first used the classification accuracy as an evaluation metric. The other used the macro averaging of F-measure, *(F-macro)*, as it gives equal weight to each class and so would show the effectiveness of the model on the small classes with lower frequencies (Özgür, Özgür, and Güngör, 2005). The F-measure is calculated using the harmonic mean of precision (*p*) and recall (*r*) as shown in Eq 2.8 (see section 2.5). It is a commonly used metric on multi-class imbalanced datasets (Ye et al. (2012) and Guo and Viktor (2004)). The F-macro is computed by first calculating the F-measure of each class and then the average over all classes is taken (This approach allows each class to have equal weight) (Zhang, Wang, and Zhao, 2015).

The next section summarises the results of the experiments.

Experiment I:

This section presents the results of applying the hybrid approach, DFO- SVM, while using the model classification accuracy as a fitness value.

Table 4.22 summarises the results of applying the hybrid approach, DFO-SVM, to perform parameters optimisation and feature selection. As shown in Table 4.22, the use of DFO reduces the number of features while improving the classification accuracy. For example, the average accuracy for the SRBCT dataset for 30 runs, is 96% and the average size of the features subset is 7.66 and the best obtained accuracy is 100% with a feature subset size equal to 7. Furthermore, the results were statistically analysed to measure the presence of any significant difference in the performance of the hybrid approach (combination of feature selection and parameters optimisation) and the performance of the classifier without parameter optimisation (classifier performance on the selected features using the default SVM kernel parameters: C=1.0, γ=1/Number of features). The t-test , which is a statistical significance test, was applied using the results of all the trials (30 runs) on each experiment: the hybrid DFO-SVM and the classifier performance on the selected features with the default RBF kernel's parameters. Based on the results, the classification accuracy difference is significant at 5% level. The result of this test indicates that the proposed hybrid approach of parameters optimisation and feature subset selection offers a statistically significant improvement on the datasets when compared to the classifier performance on the selected features without parameters setting. Thus, the developed approach can simultaneously optimise the parameters values and find a subset of features without lowering SVM classification accuracy. Figs 4.15 and 4.16 illustrate the difference of the classification accuracy on the SRBCT dataset and the MLL dataset over the 30 trials.

TABLE 4.22: Classification accuracies

Dataset	Average accuracy	Average feature selected	Best Accuracy (Features)
SRBCT	96.00%±0.04	7.66	100.00%(7)
MLL	93.00%±0.03	5.1	98.57%(10)

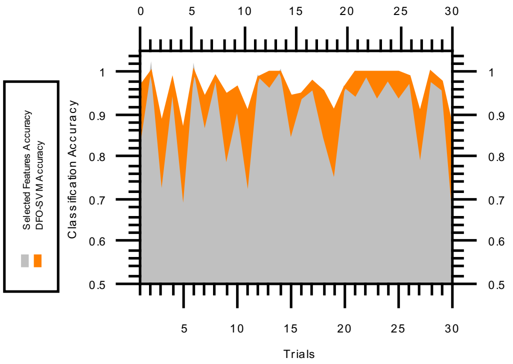

FIGURE 4.15: Comparison of the classification accuracy on the SRBCT dataset

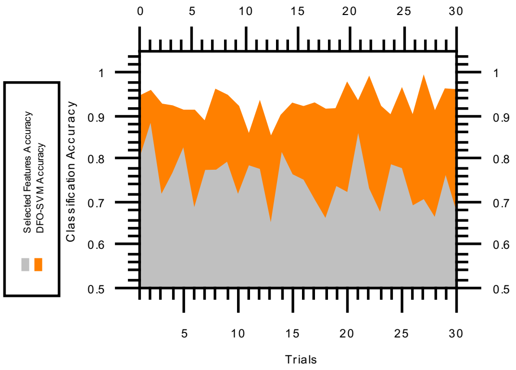

FIGURE 4.16: Comparison of the classification accuracy on the MLL dataset

Experiment II:

The SRBCT dataset does not only suffer from high dimensionality but also from class imbalance, with ES=34.9%, RMS=30.1%, NB=21.7% and BL=13.3%. Therefore, classification accuracy rate, as shown in Eq 2.5, in section 2.5, is not an effective performance metric for models on imbalanced datasets because it does not show how the model correctly classified the minority class instances (Zhu and Davidson, 2007). For example, in a case where the minority class accounts for only 2% of the training dataset, a 98% accuracy does not mean that the classifier is perfect as it can mean that it is classifying one class perfectly. Therefore, caution is advised when evaluating a model on imbalanced data, as accuracy is only suitable if the cost of FP is equal to the cost of FN, which is not always the case as in medical diagnoses. In cases like this, more suitable evaluation metrics should be used, especially when dealing with multi-class imbalanced datasets (Sun, Wong, and Kamel, 2009).

An alternative performance metric for multi-class datasets is the F-macro, which is a metric used to give equal weight to all classes regardless of their relative frequency and is usually affected by the classifier performance on rare classes. This section summarises DFO-SVM performance when the fitness function for the SVM is the F-macro, with the goal being to search for a feature subset while improving the F-macro.

Table 4.23 summarises the results of optimising the F-macro for the SRBCT dataset and the MLL dataset. The results indicate that the proposed model is capable of improving the model F-macro while reducing the number of features and optimising the kernel's parameters. As shown in Table 4.23, for the SRBCT dataset, the best F-macro is equal to 100% with feature subset size equal to 8 and, for the MLL dataset, it is equal to 98.66% with feature subset size equal to 5.

TABLE 4.23: F measure on SRBCT

	Average F-macro	**Average feature selected**	**Best F-macro (Features)**
SRBCT	94.72% ± 0.04	7.13	100.00%(8)
MLL	93.46% ± 0.03	5.22	98.66%(5)

Moreover, to determine if the results are statistically significant, a t-test was applied to measure if there was a difference in the performance of the hybrid approach (combination of feature selection and parameters optimisation) and the performance of the classifier without parameter optimisation (classifier performance on the selected features using the default SVM kernel parameters) while using F-macro as a fitness value. The results indicate that the proposed approach is statistically significant at 5% level with p-values less than 0.05. Thus the proposed approach, DFO-SVM, performs better on imbalanced datasets while using the F-macro as an evaluation metric.

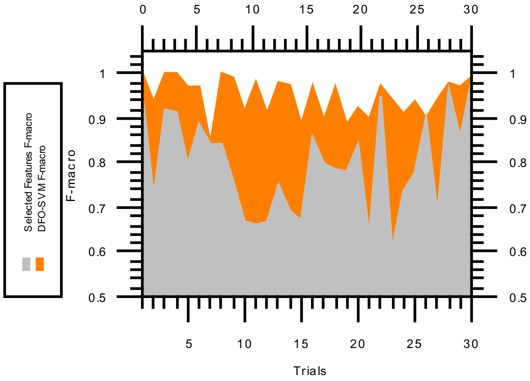

FIGURE 4.17: Comparison of F-macro on the SRBCT dataset

4.6.2 Results

Tables 4.24 and 4.25 summarise a comparison of the obtained results in terms of classification accuracy and the number of features with other methods from the literature for the SRBCT and MLL datasets respectively . For DFO-SVM, the classification accuracy presented is the best results obtained from 30 independent runs. From Tables 4.24 and 4.25, it can be observed for the SRBCT dataset and MLL datasets the proposed model outperforms in terms of both classification accuracy and the number of selected features when compared with methods reported in (Chuang et al. (2008), Khan et al. (2001), and Liu, Krishnan, and Mondry (2005)). The same classification accuracy was obtained by Zhao and Wu (2016) and Agrawal and Bala (2007) with fewer numbers of selected features. In Zhao and Wu (2016), the average classification accuracy and number of selected features was reported. However, in (Agrawal and Bala, 2007) the optimisation was applied using a train/test split which causes a feature selection bias, as using the same training dataset to conduct both feature selection and classification can lead to feature selection bias. This bias will cause over-fitting and will make it difficult to evaluate new data (Krawczuk and Łukaszuk, 2016).

In terms of F-macro, which is one of the appropriate measures for imbalanced datasets, the proposed approach gives higher F-macro values for the SRBCT dataset and the MLL datasets and fewer numbers of features when compared with the results reported in (You and Li, 2011) as shown in Tables 4.26 and 4.27. Thus, the proposed approach is applicable to multi-class imbalanced datasets.

TABLE 4.24: DFO-SVM performance comparisons for the SRBCT dataset (Accuracy)

Dataset	Evaluation metric	Validation	Results	Features	Classifier	Reference
SRBCT	Accuracy	LOOC	100%	431	*k*-NN	IBPSO [1]
SRBCT	Accuracy	10-folds	100%	4	*k*-NN	WLMGS [2]
SRBCT	Accuracy	10-folds	100%	5	SVM	WLMGS [2]
SRBCT	Accuracy	Train/Test	100%	3	SVM	GA+SVM [3]
SRBCT	Accuracy	3-folds	100%	96	ANN	ANN [4]
SRBCT	Accuracy	LOOC	100%	58	K means Clustering	Mrmr [5]
SRBCT	Accuracy	10-folds	72.75%	45	K-means Clustering	K-means [6]
SRBCT	Accuracy	10-folds	74.37%	45	K-means Clustering	K-means [6]
SRBCT	Accuracy	10-folds	100%	7	SVM	DFO-SVM

1. Chuang et al., 2008
2. Zhao and Wu, 2016
3. Agrawal and Bala, 2007
4. Khan et al., 2001
5. Liu, Krishnan, and Mondry, 2005
6. Remli et al., 2017

TABLE 4.25: DFO-SVM performance comparisons for the MLL dataset (Accuracy)

Dataset	Evaluation metric	Validation	Results	Features	Classifier	Reference
MLL	Accuracy	LOOC	100%	1292	*k*-NN	IBPSO [1]
MLL	Accuracy	10-folds	100%	4	*k*-NN	WLMGS [2]
MLL	Accuracy	10-folds	100%	4	SVM	WLMGS [2]
MLL	Accuracy	10-folds	98.57%	10	SVM	DFO-SVM

1. Chuang et al., 2008
2. Zhao and Wu, 2016

TABLE 4.26: DFO-SVM performance comparisons for the SRBCT dataset (F-macro)

Dataset	Evaluation metric	Validation	Results	Features	Classifier	Reference
SRBCT	F-macro	5-folds	100%	99	FGS1-SVM	FrPA [1]
SRBCT	**F-macro**	**10-folds**	**100%**	**8**	**SVM**	**DFO-SVM**

1. You and Li, 2011

TABLE 4.27: DFO-SVM performance comparisons for the MLL dataset (F-macro)

Dataset	Evaluation metric	Validation	Results	Features	Classifier	Reference
MLL	F macro	5-folds	97.00%	100	FGS2-SVM	FrPA [1]
MLL	**F macro**	**10-folds**	**98.66%**	**5**	**SVM**	**DFO-SVM**

1. You and Li, 2011

4.6.3 Discussion

As shown in the graphs of Fig 4.19, the proposed approach converges to the optimum solution in less than or equal to 10 generations (in all 30 trials). Moreover, Fig 4.18 is an example of one run that shows the reduction of the number of features over six generations in the SRBCT dataset. The proposed approach reduces the number of features from 50 to 6, with classification accuracy equal to 98.14 %.

The flow of information between flies creates similar flies while maintaining diversity by applying the disturbance mechanism. In DFO, the disturbance mechanism, which breaks part of the swarm to discover new locations in the search space, promotes diversity, decreases the risk of pre-mature convergence, and provides a balance between global exploration and local exploitation. In this experiment, the Δ is empirically set to 0.05. This guarantees that a percentage of the flies are randomly initialised to a new position in the search space, increasing the global exploration and lowering the risk of being trapped at the local optimum.

A balance between global exploration and local exploitation is important in any swarm intelligence algorithm to achieve good performance. If there is a high rate of exploitation and less exploration, the algorithm may converge quicker but probably to the local optimum. On the other hand, if there is a high rate of exploration and less exploitation, the flies will wander around the search space (increase randomness) which will slow the convergence. Therefore, setting a balance between global exploration and local exploitation ultimately leads to the best algorithm performance. However, it has been found that achieving a good balance is still a problem as no algorithm has claimed to achieve this balance (Yang, 2014). In other words, the balance in any algorithm is an optimisation problem, as it is an optimisation of optimisation problem. Moreover, this balance has many factors such as the population size, search space and the number of iterations. Therefore, tuning these parameters is a challenge in any optimisation problem.

4.6.4 Summary

This experiment investigated DFO performance to simultaneously optimise the kernel's parameter values for C and γ whilst selecting the optimal feature subset on two widely used microarray datasets. The results indicate

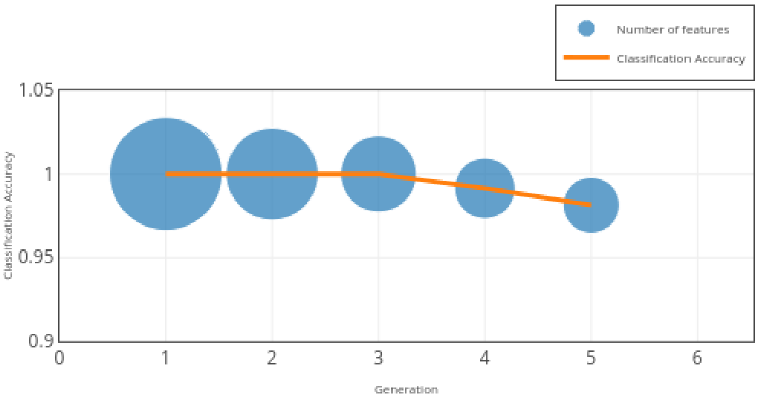

FIGURE 4.18: Reducing the number of features over one generation in the SRBCT dataset (Accuracy)

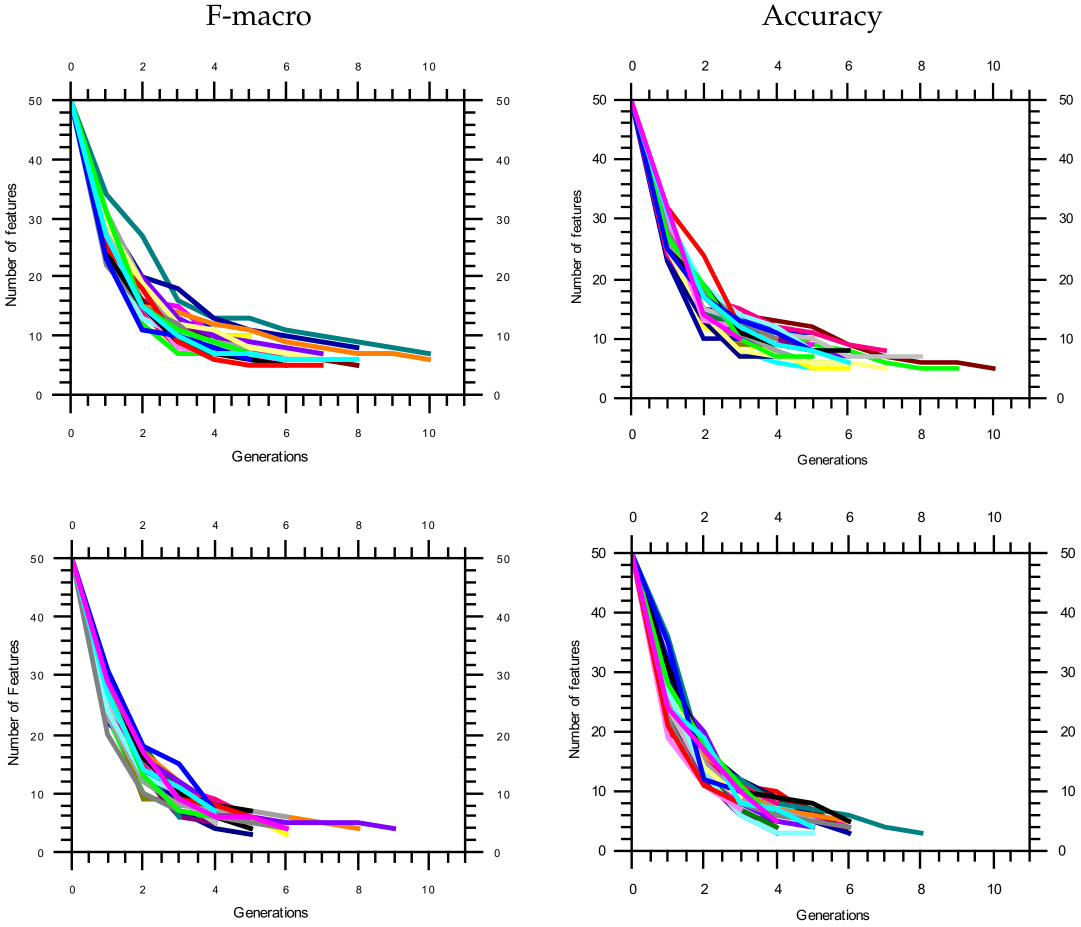

FIGURE 4.19: DFO-SVM feature selection over generations on SRBCT (top) and MLL (bottom) datasets over 30 trials.

that the proposed approach is capable of finding the optimal kernel parameter values for SVM and performing feature selection while improving the overall model performance. Moreover, the use of a combination of feature selection and parameter optimisation was found to improve the model performance when compared with feature selection only. In this experiment, results were obtained using the RBF kernel of the SVM. The simplicity of DFO makes it applicable in solving complex search and optimisation problems. Moreover, when compared to other swarm intelligence algorithms, DFO has fewer parameters to tune, apart from the population size; the only parameter to tune is the disturbance threshold. Future work includes experiments using other kernel types and other datasets with different ranges of dimensionality. Moreover, finding a theoretically optimal value for Δ, considering dimension dependent value for Δ and an adaptable and dynamically changing Δ value depending on the problem.

Chapter 5

Discussion

This chapter discusses the conducted experiments in the context of the results found and their main achievements. It also discusses the limitations of the experiments.

Class imbalance occurs when the number of instances in the majority class is significantly higher than in the minority class. Thus, the dataset is not evenly distributed. Various data mining preprocessing techniques have been applied to solve the issue of imbalanced data. However, these solutions face problems such as over-fitting. Despite these problems, researchers are still implementing various data mining preprocessing techniques to overcome the class imbalance issue. The literature review suggests three main approaches to deal with class imbalance: sampling, algorithmic solutions, and feature selection. There is no clear general answer yet as to which solution to choose. Batuwita and Palade, 2013 concluded that solutions to the class imbalance problem are dependent on the datasets. Therefore, there is no solution that fits all, and applying various data level and algorithmic level solutions and evaluating their performance on imbalanced datasets is highly recommended. This has also been suggested by Longadge and Dongre, 2013 in their review of the class imbalance problem. The authors found that applying two or more techniques gives better results when dealing with imbalanced datasets. It has also been found that a larger number of studies suggest solutions to class imbalance for binary classification, while only a few tackle the problem in multi-class classification. This is possibly due to two reasons, as stated by Ali, Shamsuddin, and Ralescu, 2013. One reason is that the class imbalance problem occurs more often in applications with binary classifications, such as medical diagnoses (patients versus controls). Another reason is that it is more complicated and computationally expensive to deal with multi-class problems. Moreover, an increasing number of models are using various state-of-the art swarm intelligence techniques such as PSO as a solution to the class imbalance problem. There is also an increase in the number of hybrid techniques that combine swarm intelligence algorithms with various data mining techniques to overcome the class imbalance problem. GA and PSO remain the most commonly used methods, along with various classification algorithms such as SVM (Deep Singh and Chug, 2017). However, the existing techniques have limitations, and more research is needed to investigate other swarm based algorithms and their capabilities in handling the class imbalance problem in order to overcome these limitations. For example, it has been found that PSO suffers from premature convergence. This is likely because particles might converge to a point that is between the personal best and the global best and that point might be a local optimum. Another limitation is the loss of diversity found in PSO because of the fast flow of information between the particles (Trelea, 2003).

In this thesis, various experiments were carried out to investigate available solutions for overcoming the class imbalance problem by combining two or more solutions to improve the classifier performance on imbalanced datasets. In experiment I, a model using a hybrid of data level and algorithmic level solutions was proposed to tackle imbalanced data. At the data level, random undersampling and SMOTE were combined to balance the dataset. Moreover, grid search was used to optimise the SVM parameters: the kernel type, the misclassification cost C, and γ. The dataset used in this work was the direct marketing dataset. For better evaluation, the proposed model used various performance metrics to evaluate the model performance: AUC, accuracy, sensitivity, and specificity. The proposed model performed competitively when compared with other models on the same dataset. Despite the promising results of the combined approach, the model still cannot be generalised, as only one dataset was used. Another limitation is the use of grid search to optimise the kernel parameters. Grid search is a time consuming and computationally expensive method to optimise the values (see section 2.3.1). Future work can evaluate the performance of the suggested model on other datasets from different fields with different class distribution. Moreover, it could investigate the use of meta-heuristic search algorithms to optimise the SVM kernel parameters and overcome the grid search limitations (for more details on the experiment, see section 4.1).

As previously mentioned, there has been an increase in the use of swarm intelligence techniques to solve the class imbalance problem. This is because swarm intelligence algorithms have performed well in tackling the class imbalance problem in various applications (for more details, see section 3.4).

This research explored the possibility of using SDS to handle the class imbalance problem, taking advantage of its speed (thanks to the partial function evaluation) and the various recruitment strategies that can be used to balance the global exploration and local exploitation in the search problem. Experiment II was a first attempt to use SDS in undersampling and investigated a new approach for balancing the marketing dataset using SDS. In the context of the results reported in Experiment II, the proposed model, which used SDS, could potentially serve as a computationally cheap yet competitive swarm intelligence approach that could be applied in the growing field of data mining to re-sample frequently imbalanced classes in real-world applications. The results of this experiment have shown that while comparable to the comprehensive yet impractical Euclidean distance tool, SDS offers a promising alternative. It can be used to perform an "informed" undersampling of the majority class, as opposed to random undersampling, which might cause loss of information. However, there are some limitations in experiment II. Firstly, only one dataset was used to evaluate the proposed model, so it should be evaluated on other imbalanced datasets. Moreover, the performance of the proposed model has not been compared with that of other swarm based techniques. Finally, the population size is constant over the iterations allowed. Thus, a dynamically changing population size should be considered to reduce computational time and memory (more details about the experiment can be found in section 4.2).

Based on the promising results of experiment II, SDS's behaviour in undersampling was further explored and combined with other methods on various datasets in Experiment III. In this experiment, a hybrid approach was proposed to deal with real-world imbalanced datasets. The proposed model used SMOTE to perform the oversampling and the agents of SDS to perform an "informed" undersampling of the majority classes. Moreover, grid search was used to tune the C and γ SVM parameters. Acknowledging the role of SDS in removing exact duplicates (redundancies) of instances from the majority on the outcome of the classifiers, as seen in Experiment II, the proposed SDS algorithm in this experiment, Experiment III, conducted its process at the feature-level (attribute-level) of the datasets using a feature-level threshold (more details on the threshold calculation in section 4.3.1). In other words, the undersampling was carried out in the feature space, not in the instance space. This algorithm demonstrated effectiveness in identifying feature-level duplications, which were shown to also play an important role in the performance of the classifier. This experiment presented an analysis of both instance- and feature-level redundancies and established a link between the feature-level duplications and the role of the feature-level undersampling mechanism. This analysis was accompanied by an investigation of the behaviour of the agents through their activity level during the undersampling process. It was shown that agents' activity is directly proportional to the level of redundancy in the datasets. In other words, the higher the level of feature-level duplications, the higher the level of agent activity, which also resulted in faster convergence of the population. This experiment also investigated the ability of the algorithm to comprehensively explore the search space by forming a quick decision using the partial function evaluation and without having to visit all the features of each instance in the datasets. This was done using two methods: calculating the percentage of the instances visited by the SDS agent and the percentage of features visited (see Fig 4.5 in section 4.3). In this experiment, SDS proved to be an effective technique for undersampling imbalanced datasets with a high feature duplication rate because it limited the effect of removing useful information from the majority class while improving the model's overall performance. In spite of the promising results, there are some limitations to the proposed model. First, it was noted that during the undersampling process, there was a gradual decrease in the size of the search space. By establishing a more dynamic population size and iteration numbers, the coverage of the search space could be made homogeneous throughout the undersampling process. Moreover, the proposed model removes one instance at time. Therefore, more work is needed to improve the proposed approach to make it able to remove more than one instance at a time, thus speeding up the undersampling process, and make it applicable for large datasets (more details about the experiment can be found in section 4.3).

Experiments II and III demonstrated that SDS can be used in performing an informed undersampling of the majority class to remove redundant data while protecting useful information. Based on the promising results, SDS was extended further to tackle the class imbalance problem at the feature level by performing feature selection to find the most relevant feature subset for the classification task. In Experiment IV, SDS was adapted to find a suitable feature subset. Moreover, SVM was used as a classifier to evaluate the predictive accuracy of the agent. In other words, the fitness function of SDS was the SVM classification accuracy. The proposed method exhibited a statistically significant outperformance when compared with the performance of the classifier without SDS-powered feature selection. The results were also compared with other methods from the literature over nine real-world datasets. It was shown that the proposed SDS based feature selection (SDS-FS) offered a competitive performance compared to other methods on datasets with a feature size greater than 10. In this experiment, the behaviour of the proposed algorithm was investigated in the context of global exploration and local exploitation. SDS-FS uses a *context sensitive recruitment* strategy. In this, if an active agent finds another active agent with whom it shares the same hypothesis, the active agent is set to inactive. This frees the agent, improves the agent's diversity and increases its global exploration. Moreover, to improve the local exploitation the proposed approach used an *offset* mechanism. When an inactive agent picks an active agent it copies its hypothesis with one random feature offsetted or changed (for example, 1 is changed to 0 or 0 is changed to 1). By combining both approaches, *context sensitive strategy* and *offset* mechanism, a balance between the global exploration and local exploitation was achieved. Despite the promising results of the novel approach, there are a few limitations. One is that the size of the feature subset is empirically set to half of the original feature size, so the size of the feature subset needs to be set beforehand by the user. Therefore, more work needs to be carried out to find the optimal and adaptive feature subset size.

In terms of the algorithmic level solution, Experiment v explored the use of DFO as a continuous optimiser to optimise the SVM kernel's parameters (C and γ) and perform cost sensitive learning to improve the classifier's performance on imbalanced datasets. In this experiment, the use of the swarming behaviour of the flies and their diversity (attributed to the Δ at the update phase) in the search space in conducting cost sensitive learning was investigated on eight real-world datasets. The proposed model was compared with other techniques for optimising the classifier's parameters, which include the well-known PSO, the frequently used grid search, and random search, which is used as a control algorithm. The results demonstrate the statistically significant outperformance of the proposed optimisation technique over the other techniques on the same datasets. In this experiment, Δ, which was the only parameter to set, was set to 0.5 to improve the algorithm's diversity (50% of the flies would commence the search in another random location within the search space). The purpose was to explore new locations in the search space and avoid being trapped at a local optimum. Moreover, to investigate the effect of Δ, DFO was applied to optimise the kernel's parameters with a Δ equal to 0.001. As shown in Fig 4.13, the performance of DFO with Δ

equal to 0.5 was better than the DFO with a Δ equal to 0.001 and the complete random search. Moreover, the performance of DFO with Δ equal to 0.5 was better than a random search for optimising the SVM kernel's parameters. The motivation to use DFO is its simplicity as an SVM kernel optimiser. DFO has fewer parameters to adjust or tune, as opposed to the widely used GA and PSO (more details can be found in Chapter 3). In DFO, the only parameter is Δ at the update phase. The simplicity of this algorithm, with its fewer components and fewer tunable parameters, makes it applicable in solving search and optimisation problems. In this experiment, one limitation is that the value of Δ was empirically chosen. Thus, more work is needed to theoretically set the value of Δ to control stochasticity in the algorithm and provide a balance between global exploration and local exploitation.

After the promising results of using DFO to perform cost sensitive learning, the use of DFO was extended to provide a model that optimises the kernel parameters and performs feature selection simultaneously, as shown in Experiment VI. The model was evaluated on two multi-class microarray datasets, in which the experiment was conducted twice using two evaluation metrics: accuracy and the F-macro. Accuracy was used for comparison purposes as most work in the literature uses accuracy as an evaluation metric in the fitness function. However, accuracy as an evaluation metric is irrelevant when the classifier is trained on imbalanced datasets. Therefore, it is recommended to employ other evaluation metrics when training a model on skewed datasets. The F-macro was thus used to optimise performance. The proposed model was compared with other models from the literature. Results show that the model is capable of optimising the kernel parameters, C and γ, and perform feature selection. In terms of the F-macro, the model outperformed the other technique that uses the F-macro in the fitness function, known as the FrPA technique (You and Li, 2011). DFO has therefore been demonstrated to be applicable in overcoming the class imbalance problem by tuning the SVM and removing irrelevant features. However, the results cannot be generalised as only two datasets were used. The proposed approach thus needs to be conducted with other datasets that have different imbalance ratios. As with Experiment V, more work is needed to theoretically set the Δ value in DFO. Moreover, like other nature-inspired population-based algorithms consisting of both stochastic and guided parts, the features generated in this experiment varied from one trial to the next. Thus, more work is needed to improve the stability in the feature selection, especially in applications like bioinformatics and medical diagnoses.

In this thesis, the problem of class imbalance was studied. Various solutions including novel approaches were examined using six different experiments. The investigation used six different approaches, each with its own way of handling the class imbalance problem. There were a number of goals: The first was to understand how different techniques can be used to solve the class imbalance issue, what preprocessing steps are required to prepare the data for the classification algorithm, and how swarm intelligence can be used to improve the performance of the learning algorithm on imbalanced datasets. The second advantage is dealing with datasets that have different dimensions, different distributions, and different feature types. The third advantage is that they provide an opportunity to explore the possible solutions for the class imbalance problem and how this problem can be solved at the data, algorithmic, and feature level using swarm intelligence techniques. The final goal was to introduce various novel swarm-based approaches to handle the class imbalance issue in data mining. Based on the results, there is no systemic approach or general framework to follow when dealing with the problem of class imbalance, and research is still needed to identify which method or combination of methods works better. For example, when it comes to choosing between oversampling or undersampling the dataset, more work needs to be done so that the techniques or algorithms can automatically identify whether data should be removed from the majority class or more instances should be added to the minority class. More work also needs to be done to combine different techniques and evaluate these combinations to find out which works better in which problem domain. Rather than trying only a combination of algorithm level and data level techniques, it could be interesting to find a combination of different swarm algorithms and a combination of swarm algorithms with non-swarm algorithms such as feature selection methods that can supplement each other and reduce the customisation effort of sorting out the class imbalance problem. The scenario in which there is more than one majority or minority class (multi-classification) is another future dimension that needs more research, as different classes may advocate for different combinations of attributes that may conflict with each other. This could make undersampling or oversampling challenging. The next chapter will outline directions for future research.

Chapter 6

Conclusion and Future Work

This chapter provides a summary and conclusion. Recommendations for future research are also given.

6.1 Summary

The aim of this thesis was to provide a review of current work on the class imbalance problem in data mining. It offered a definition of key swarm intelligence algorithms and highlighted the applications of these algorithms in solving various aspects of the class imbalance issue. The knowledge gained from the research activities was then used to implement applications and evaluate various solutions for imbalanced datasets. First, the possibility of combining existing techniques to improve the classifier performance on a direct marketing imbalanced dataset was investigated. The performance of the hybrid data and algorithmic level solution, HybridDA, was compared with other methods from the literature on the same dataset, and the results show the outperformance of the proposed model. This highlights the benefits of combining existing techniques when tackling the class imbalance problem. Following that, a novel approach that used SDS to undersample the majority class in the direct marketing datasets was introduced. SDS was combined with SMOTE to combat the class imbalance problem at the data level. The outcome of this novel approach was the introduction of an informed undersampling technique that reduces the number of instances in the majority class without removing useful information.

Another novel undersampling approach based on SDS was proposed in which SDS was combined with SMOTE at the data level and the SVM parameters were tuned using a grid search at the algorithmic level. The purpose was to improve the classifier performance on imbalanced datasets. The proposed approach was compared with random undersampling and other techniques on nine imbalanced datasets. The results show the outperformance of the SDS based model when dealing with the class imbalance problem. Moreover, an analysis of the agent-led approach in undersampling was provided, in which a link between the feature-level duplications and the role feature-level undersampling mechanism was found. The analysis found that the agents activity is directly proportional to the level of redundancy in the datasets at both instance level and feature level.

Following the promising results of SDS in performing undersampling at the data level (in both instance space and feature space), the use of SDS was extended to perform feature selection. SDS was used to reduce the size of the feature to half of the feature size and improve the SVM performance on nine imbalanced datasets with different dimensions. The outcome of the investigation shows that SDS based feature selection performs competitively with other methods on datasets when the original feature size is greater than 10.

In this thesis, DFO was used to investigate the problem of class imbalance and its behaviour was analysed. First, DFO was applied as a continuous optimiser for the SVM kernel parameters. The outcome of this model is a novel approach for cost sensitive learning from imbalanced datasets using DFO. The proposed approach gives a higher F-measure, when compared with other techniques from the literature on all eight datasets. Following these promising results, the model was extended further to deal with high dimensionality and to perform feature selection and parameters tuning on two microarray datasets. Optimising the feature subset and kernel parameters was conducted using two different fitness functions with two different evaluation metrics including classification accuracy and *F-macro*. The results indicate that the proposed DFO based approach can be used to perform feature selection and tune the kernel parameters. Moreover, the combined approach gives better results when compared with feature selection only.

Class imbalance is a challenging issue in data mining that occurs in many real-world applications, and it is not always sufficient to solve the imbalance problem by adjusting the class distribution. Therefore, research was carried out to investigate the problem and explore various solutions. Although numerous approaches have been proposed in the literature, the class imbalance problem remains a challenge in the development of the learning model. This challenge will become more significant due to recent developments in big data . Despite the amount of research conducted thus far, there is still no unified framework to deal with class imbalance. This is because there are various challenges when dealing with imbalanced datasets, such as class distribution and the training samples size. Moreover, it is a dataset dependent problem as each dataset has its own characteristics, which might affect the classifier performance. Thus, no solution fits all.

This research has mainly focused on the class imbalance problem. It investigated the use of combined approaches and swarm intelligence based techniques, SDS and DFO, to combat the class imbalance problem at all levels: data, algorithmic, and feature level. The results of the various approaches in this research leads to the same conclusion that no solution fits all. Depending on the characteristics of the dataset, such as the number of features, the imbalance rate, and the size, solutions may vary. One should try two or more techniques because hybrid approaches give better results when dealing with imbalanced datasets. When employing a hybrid approach, the use of two or more evaluation metrics to compare the model performance on the imbalanced dataset is recommended for a better understanding. The findings also suggest that more work is needed to analyse the nature of the majority and minority classes in order to gain a better understanding of the reasons for the difficulties that arise when learning from the imbalanced dataset. There are many more directions where further developments to the solutions for handling imbalanced datasets can be considered. The next section lists the most important future research directions.

6.2 Future work

Research and experiments conducted in this study suggest that many more directions can be identified for investigating further solutions to the class imbalance problem. This section lists the most important potential directions for future work.

- Investigate the use of SDS to oversample the minority class by picking an instance as a model, finding the closest instance, and then generating a new one. This will create an informed oversampling technique using SDS. The performance of SDS can be evaluated using various benchmarks, and its performance can be compared with other oversampling techniques from the literature.

- Another suggestion is to investigate the use of SDS to oversample the minority class. This can be achieved by using feature similarities where values like the median of value difference of each feature is calculated, then a threshold is used to create instances to increase the number of minority class examples until the minority class balances with the majority class. This approach performs *informed* oversampling to balance the dataset.

- The hybridisation of the two swarm intelligence algorithms, DFO and SDS, in which DFO is used to optimise the best values for the SDS's agent size and iterations are allowed when undersampling the dataset and in order to study the relationship between both values and the coverage percentage of the shrinking search space of the dataset being undersampled. The simplicity of DFO, which has only one parameter to tune, that is the Δ the update phase, makes it applicable to solve complex search and optimisation problems. DFO is a numerical optimiser over a continuous search space. In this future work, the possibility of using DFO as a discrete optimiser for integer values will be investigated and the quality of the solution found will be evaluated using various benchmarks.

- The hybridisation of SDS and DFO on high dimensional and imbalanced datasets, in which SDS is used for feature selection and DFO is used for parameters tuning. The purpose is to take the best of both. SDS is a discrete optimiser which can be used for feature selection, while DFO is a continuous optimiser that can be used to optimise the kernel parameters, C and γ. The hybridisation of the two population-based meta-heuristic techniques provides a hybrid to simultaneously perform feature selection and parameters tuning. The performance of the proposed approach can be evaluated against other hybrid models from the literature.

Bibliography

Ab Wahab, M. N., S. Nefti-Meziani, and A. Atyabi (2015). "A comprehensive review of swarm optimization algorithms". In: *PloS one* 10.5, e0122827.

Abdelbar, A. M., S. Ragab, and S. Mitri (2003). "Applying co-evolutionary particle swam optimization to the egyptian board game seega". In: *Proceedings of the first Asian-Pacific workshop on genetic programming*. LIS-CONF-2006–020, pp. 9–15.

Abonyi, J., B. Feil, and A. Abraham (2005). "Computational intelligence in data mining". In: *Informatica* 29.1.

Abraham, A. and V. Ramos (2003). "Web usage mining using artificial ant colony clustering and linear genetic programming". In: *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on*. Vol. 2, pp. 1384–1391.

Acuna, E. and C. Rodriguez (2004). "The treatment of missing values and its effect on classifier accuracy". In: *Classification, Clustering, and Data Mining Applications*, pp. 639–647.

Admane, L. et al. (2006). "AntPart: an algorithm for the unsupervised classification problem using ants". In: *Applied Mathematics and Computation* 180.1, pp. 16–28.

Aggarwal, C. C. and P. S. Yu (2001). "Outlier detection for high dimensional data". In: *ACM Sigmod Record*. Vol. 30. 2. ACM, pp. 37–46.

Aghdam, M. H. and P. Kabiri (2016). "Feature Selection for Intrusion Detection System Using Ant Colony Optimization." In: *IJ Network Security* 18.3, pp. 420–432.

Agrawal, R. K. and R. Bala (2007). "A hybrid approach for selection of relevant features for microarray datasets". In: *International Journal of Computer and Information Engineering* 1.2.

Ahmad, F. et al. (2013). "Intelligent Medical Disease Diagnosis Using Improved Hybrid Genetic Algorithm - Multilayer Perceptron Network". In: *Journal of Medical Systems* 37.2, p. 9934.

Aisha, Nazziwa, Mohd Bakri Adam, and Shamarina Shohaimi (2013). "Effect of missing value methods on Bayesian network classification of hepatitis data". In: *International Journal of Computer Science and Telecommunications* 4.6, pp. 8–12.

Ajitha, P. and E. Chandra (2015). "A Survey on Outliers Detection in Distributed Data Mining for Big Data". In: *Journal of Basic and Applied Scientific Research* 5.2, pp. 31–38.

Al-Rifaie, M. M. (2011). "When birds and ants set off to draw". In: *15th International Conference Information Visualisation*.

Al-Rifaie, M. M. and A. Aber (2016). "Dispersive flies optimisation and medical imaging". In: *Recent advances in computational optimization*. Springer, pp. 183–203.

Al-Rifaie, M. M., A. Aber, and A. M. Oudah (2012). "Utilising Stochastic Diffusion Search to identify metastasis in bone scans and microcalcifications on mammographs". In: *2012 IEEE International Conference on Bioinformatics and Biomedicine Workshops*, pp. 280–287.

Al-Rifaie, M. M., A. Aber, and R. Raisys (2011). "Swarming robots and possible medical applications". In: *International Society for the Electronic Arts (ISEA 2011)*.

Al-Rifaie, M. M. and J. M. Bishop (2013a). "Stochastic diffusion search review". In: *Paladyn, Journal of Behavioral Robotics* 4.3, pp. 155–173.

— (2013b). "Swarmic Sketches and Attention Mechanism". In: *Evolutionary and Biologically Inspired Music, Sound, Art and Design*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 85–96.

Al-Rifaie, M. M., J. M. Bishop, and S. Caines (2012). "Creativity and autonomy in swarm intelligence systems". In: *Cognitive Computation* 4.3, pp. 320–331.

Al-Rifaie, M. M. et al. (2017a). "On Symmetry, Aesthetics and Quantifying Symmetrical Complexity". In: *Computational Intelligence in Music, Sound, Art and Design: 6th International Conference, EvoMUSART 2017, Amsterdam, The Netherlands, April 19–21, 2017, Proceedings*. Springer International Publishing, pp. 17–32.

Al-Rifaie, M. M. et al. (2017b). "Swarmic autopoiesis and computational creativity". In: *Connection Science*, pp. 1–19.

Al-Rifaie, M. M. (2014). "Dispersive Flies Optimisation". In: *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*. Vol. 2. Annals of Computer Science and Information Systems. IEEE, pp. 529–538.

Al-Shahib, A., R. Breitling, and D. Gilbert (2005). "Feature selection and the class imbalance problem in predicting protein function from sequence". In: *Applied Bioinformatics* 4.3, pp. 195–203.

Alba, E. et al. (2007). “Gene selection in cancer classification using PSO/SVM and GA/SVM hybrid algorithms”. In: *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*. IEEE, pp. 284–290.

Alhakbani, H. A. and M. M. Al-Rifaie (2016a). “A Swarm Intelligence Approach in Undersampling Majority Class”. In: *Swarm Intelligence*. Cham: Springer International Publishing, pp. 225–232.

— (2016b). “Handling class imbalance in direct marketing dataset using a hybrid data and algorithmic level solutions”. In: *2016 SAI Computing Conference (SAI)*, pp. 446–451.

— (2017a). “Exploring Feature-Level Duplications on Imbalanced Data Using Stochastic Diffusion Search”. In: *Multi-Agent Systems and Agreement Technologies*. Cham: Springer International Publishing, pp. 305–313.

— (2017b). “Feature Selection Using Stochastic Diffusion Search”. In: *Proceedings of the Genetic and Evolutionary Computation Conference*. GECCO ’17. Berlin, Germany: ACM, pp. 385–392.

— (2017c). “Optimising SVM to classify imbalanced data using Dispersive Flies Optimisation”. In: *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 399–402.

Ali, A., S. M. Shamsuddin, and A. L. Ralescu (2013). “Classification with class imbalance problem”. In: *Int. J. Advance Soft Compu. Appl* 5.3.

Ali, M. M. and A. Törn (2004). “Population set-based global optimization algorithms: some modifications and numerical studies”. In: *Computers & Operations Research* 31.10, pp. 1703–1725.

Almuallim, H. and T. G. Dietterich (1994). “Learning boolean concepts in the presence of many irrelevant features”. In: *Artificial Intelligence* 69.1–2, pp. 279–305.

Alpaydin, E. (2014). *Introduction to machine learning*. MIT press.

Alwan, H. B. and K. R. Ku-Mahamud (2012a). “Incremental continuous ant colony optimization technique for support vector machine model selection problem”. In: *Proceedings the 17th WSEAS International Conference on Applied Mathematics (AMATH ’12) 2012*.

Alwan, H. B. and K. R. Ku-Mahamud (2012b). “Optimizing Support Vector Machine parameters using continuous Ant Colony Optimization”. In: *2012 7th International Conference on Computing and Convergence Technology (ICCCT)*, pp. 164–169.

Ambroise, C. and G. J. McLachlan (2002). “Selection bias in gene extraction on the basis of microarray gene-expression data”. In: *Proceedings of the National Academy of Sciences* 99.10, pp. 6562–6566.

Armstrong, S. A. et al. (2002). “MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia.” In: *Nature Genetics* 30.1.

Azevedo, A. I. R. L. and M. F. Santos (2008). “KDD, SEMMA and CRISP-DM: a parallel overview”. In: *IADS-DM*.

Bahnsen, A. C., D. Aouada, and B. E. Ottersten (2015). “Ensemble of Example-Dependent Cost-Sensitive Decision Trees”. In: *CoRR* abs/1505.04637.

Bandyopadhyay, S. and S. K. Pal (2007). *Classification and learning using genetic algorithms: applications in bioinformatics and web intelligence*. Springer Science & Business Media.

Bao, Y., Z. Hu, and T. Xiong (2013). “A PSO and pattern search based memetic algorithm for SVMs parameters optimization”. In: *Neurocomputing* 117, pp. 98–106.

Batista, G. E. A. P. A., R. C. Prati, and M. C. Monard (2004). “A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data”. In: *SIGKDD Explor. Newsl.* 6.1, pp. 20–29.

Batista, Gustavo EAPA, Ana LC Bazzan, and Maria Carolina Monard (2003). “Balancing Training Data for Automated Annotation of Keywords: a Case Study.” In: *WOB*, pp. 10–18.

Battiti, R. (1994). “Using mutual information for selecting features in supervised neural net learning”. In: *IEEE Transactions on Neural Networks* 5.4, pp. 537–550.

Batuwita, R. and V. Palade (2013). “Class imbalance learning methods for support vector machines”. In: *Imbalanced learning: Foundations, Algorithms, and Applications* 83.

Beattie, P. D. and J. M. Bishop (1998). “Self-Localisation in the “Senario” Autonomous Wheelchair”. In: *J. Intell. Robotics Syst.* 22.3–4, pp. 255–267.

Beckmann, M., N. F. F. Ebecken, and B. S. L. P. de Lima (2015). “A KNN Undersampling Approach for Data Balancing”. In: *Journal of Intelligent Learning Systems and Applications* 7.04, p. 104.

Belal, M. et al. (2006). “Swarm intelligence”. In: *Handbook of Bioinspired Algorithms and Applications. Series: CRC Computer & Information Science* 7.

Beni, G. and J. Wang (1993). “Swarm intelligence in cellular robotic systems”. In: *Robots and Biological Systems: Towards a New Bionics?* Springer, pp. 703–712.

Bergh, F. van den and A. P. Engelbrecht (2004). “A Cooperative approach to particle swarm optimization”. In: *IEEE Transactions on Evolutionary Computation* 8.3, pp. 225–239.

Bermejo, P., J. A. Gámez, and J. M. Puerta (2011). "A GRASP algorithm for fast hybrid (filter-wrapper) feature subset selection in high-dimensional datasets". In: *Pattern Recognition Letters* 32.5, pp. 701–711.

Berson, A., S. Smith, and K. Thearling (2000). *Building Data Mining Applications for CRM*. McGraw-Hill Companies.

Bhattacharyya, S. (2015). *Handbook of Research on Swarm Intelligence in Engineering*. IGI Global.

Bishop, J. M. (1989a). "Anarchic techniques for pattern classification". PhD thesis. University of Reading.

— (1989b). "Stochastic searching networks". In: *Artificial Neural Networks, 1989., First IEE International Conference on (Conf. Publ. No. 313)*. IET, pp. 329–331.

— (2003). "Coupled stochastic diffusion processes". In: *Proc. School Conference for Annual Research Projects (SCARP), Reading, UK*, pp. 185–187.

Bishop, J. M. and P. Torr (1992). "The Stochastic Search Network". In: *Neural Networks for Vision, Speech and Natural Language*. Springer Netherlands, pp. 370–387.

Blagus, R. and L. Lusa (2013). "SMOTE for high-dimensional class-imbalanced data". In: *BMC Bioinformatics* 14.1, p. 106.

Blanco, R. et al. (2004). "Gene selection for cancer classification using wrapper approaches". In: *International Journal of Pattern Recognition and Artificial Intelligence* 18.08, pp. 1373–1390.

Blickle, T. and L. Thiele (1996). "A Comparison of Selection Schemes Used in Evolutionary Algorithms". In: *Evolutionary Computation* 4.4, pp. 361–394.

Blum, C. (2005a). "Ant colony optimization: Introduction and recent trends". In: *Physics of Life Reviews* 2.4, pp. 353–373.

Blum, C. (2005b). "Beam-ACO–Hybridizing ant colony optimization with beam search: An application to open shop scheduling". In: *Computers & Operations Research* 32.6, pp. 1565–1591.

Bolton, R. J. and D. J. Hand (2002). "Statistical fraud detection: A review". In: *Statistical Science*, pp. 235–249.

Bonabeau, E., M. Dorigo, and G. Theraulaz (1999). *From Natural to Artificial Swarm Intelligence*. Oxford University Press.

Brown, M. L. and J. F. Kros (2003). "Data mining and the impact of missing data". In: *Industrial Management & Data Systems* 103.8, pp. 611–621.

Burez, J. and D. Van den Poel (2009). "Handling class imbalance in customer churn prediction". In: *Expert Systems with Applications* 36.3, Part 1, pp. 4626–4636.

Cant, R. J. and C. S. Langensiepen (2009). "Methods for automated object placement in virtual scenes". In: *Computer Modelling and Simulation, 2009. UKSIM'09. 11th International Conference on*. IEEE, pp. 431–436.

Cao, P., D. Zhao, and O. Zaiane (2013). "An optimized cost-sensitive SVM for imbalanced data learning". In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, pp. 280–292.

Carvajal, K. et al. (2004). "Neural network method for failure detection with skewed class distribution". In: *Insight-Non-Destructive Testing and Condition Monitoring* 46.7, pp. 399–402.

Chakraborty, U. K. (2008). *Advances in differential evolution*. Vol. 143. Springer.

Chandrashekar, G. and F. Sahin (2014). "A Survey on Feature Selection Methods". In: *Comput. Electr. Eng.* 40.1, pp. 16–28.

Chang, C.-C. and C.-J. Lin (2011). "LIBSVM: a library for support vector machines". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.3, p. 27.

Chang, Y.-W. et al. (2010). "Training and testing low-degree polynomial data mappings via linear SVM". In: *Journal of Machine Learning Research* 11.Apr, pp. 1471–1490.

Chaparro, I. and F. Valdez (2013). "Variants of Ant Colony Optimization: A Metaheuristic for Solving the Traveling Salesman Problem". In: *Recent Advances on Hybrid Intelligent Systems*. Springer Berlin Heidelberg, pp. 323–331.

Chauhan, N., V. Ravi, and D. K. Chandra (2009). "Differential evolution trained wavelet neural networks: Application to bankruptcy prediction in banks". In: *Expert Systems with Applications* 36.4, pp. 7659–7665.

Chawla, N. V. (2005). "Data Mining for Imbalanced Datasets: An Overview". In: *Data Mining and Knowledge Discovery Handbook*. Springer US, pp. 853–867.

Chawla, N. V., N. Japkowicz, and A. Kotcz (2004). "Editorial: Special Issue on Learning from Imbalanced Data Sets". In: *SIGKDD Explor. Newsl.* 6.1, pp. 1–6.

Chawla, N. V. et al. (2002). "SMOTE: synthetic minority over-sampling technique". In: *Journal of Artificial Intelligence Research* 16, pp. 321–357.

Chawla, N. V. (2009). "Data mining for imbalanced datasets: An overview". In: *Data mining and knowledge discovery handbook*. Springer, pp. 875–886.

Chen, CL Philip and Chun-Yang Zhang (2014). "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data". In: *Information Sciences* 275, pp. 314–347.

Chen, S., G. P. Rangaiah, and M. Srinivas (2017). "Differential evolution: method, developments and chemical engineering applications". In: *Differential Evolution In Chemical Engineering: Developments And Applications* 6, p. 35.

Chen, X.-w. and M. Wasikowski (2008). "FAST: A Roc-based Feature Selection Metric for Small Samples and Imbalanced Data Classification Problems". In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '08. Las Vegas, Nevada, USA: ACM, pp. 124–132.

Chen, Y., A. Abraham, and B. Yang (2005). "Hybrid neurocomputing for breast cancer detection". In: *Soft Computing as Transdisciplinary Science and Technology*. Springer, pp. 884–892.

Chen, Y. et al. (2005). "Hybrid Methods for Stock Index Modeling". In: *Fuzzy Systems and Knowledge Discovery: Second International Conference, FSKD 2005, Changsha, China, August 27–29, 2005, Proceedings, Part II*. Springer Berlin Heidelberg, pp. 1067–1070.

Cheng, S., Y. Shi, and Q. Qin (2015). "Population Diversity of Particle Swarm Optimizer Solving Single-and Multi-Objective Problems". In: *Emerging Research on Swarm Intelligence and Algorithm Optimization*, pp. 71–98.

Chuang, L.-Y., C.-H. Ke, and C.-H. Yang (2016). "A Hybrid Both Filter and Wrapper Feature Selection Method for Microarray Classification". In: *CoRR* abs/1612.08669.

Chuang, L.-Y. et al. (2008). "Improved binary PSO for feature selection using gene expression data". In: *Computational Biology and Chemistry* 32.1, pp. 29–38.

Chunhong, Z. and J. Licheng (2004). "Automatic parameters selection for SVM based on GA". In: *Intelligent Control and Automation, 2004. WCICA 2004. Fifth World Congress on*. Vol. 2. IEEE, pp. 1869–1872.

Corne, D. W., A. Reynolds, and E. Bonabeau (2012). "Swarm intelligence". In: *Handbook of Natural Computing*. Springer, pp. 1599–1622.

Črepinšek, M., S.-H. Liu, and M. Mernik (2013). "Exploration and exploitation in evolutionary algorithms: A survey". In: *ACM Computing Surveys (CSUR)* 45.3, p. 35.

Dall'Asta, L. et al. (2006). "Agreement dynamics on small-world networks". In: *EPL (Europhysics Letters)* 73.6, p. 969.

Das, S., A. Abraham, and A. Konar (2008). "Particle swarm optimization and differential evolution algorithms: technical analysis, applications and hybridization perspectives". In: *Advances of Computational Intelligence in Industrial Systems*. Springer, pp. 1–38.

De Meyer, K. (2000). *Explorations in stochastic diffusion search: Soft-and hardware implementations of biologically inspired spiking neuron stochastic diffusion networks*. Tech. rep. Technical Report KDM/JMB/2000.

Deep Singh, P. and A. Chug (2017). "Software defect prediction analysis using machine learning algorithms". In: *2017 7th International Conference on Cloud Computing, Data Science Engineering - Confluence*, pp. 775–781.

Deneubourg, J.-L. et al. (1991). "The dynamics of collective sorting robot-like ants and ant-like robots". In: *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*, pp. 356–363.

Di Martino, S. et al. (2011). "A genetic algorithm to configure support vector machines for predicting fault-prone components". In: *International Conference on Product Focused Software Process Improvement*. Springer, pp. 247–261.

Diaz, J. M., R. C. Pinon, and G. Solano (2014). "Lung cancer classification using genetic algorithm to optimize prediction models". In: *IISA 2014, The 5th International Conference on Information, Intelligence, Systems and Applications*, pp. 1–6.

Dietterich, T. G. and G. Bakiri (1991). "Error-correcting output codes: A general method for improving multiclass inductive learning programs". In: *AAAI*, pp. 572–577.

Ding, S. (2009). "Feature Selection Based F-Score and ACO Algorithm in Support Vector Machine". In: *2009 Second International Symposium on Knowledge Acquisition and Modeling*. Vol. 1, pp. 19–23.

Ding, S. and L. Chen (2010). "Intelligent optimization methods for high-dimensional data classification for support vector machines". In: *Intelligent Information Management* 2.06, p. 354.

Divya, D. and S. S. Babu (2016). "Methods to detect different types of outliers". In: *2016 International Conference on Data Mining and Advanced Computing (SAPIENCE)*, pp. 23–28.

Dorigo, M. (2007). "In The Editorial of the First Issue of: Swarm Intelligence Journal". In: Springer Science + Business Media, pp. 1–2.

Dorigo, M., M. Birattari, and T. Stutzle (2006). "Ant colony optimization". In: *IEEE Computational Intelligence Magazine* 1.4, pp. 28–39.

Dorigo, M. and C. Blum (2005). "Ant colony optimization theory: A survey". In: *Theoretical Computer Science* 344.2–3, pp. 243–278.

Dorigo, M. and T. Stützle (2003). "The ant colony optimization metaheuristic: Algorithms, applications, and advances". In: *Handbook of metaheuristics*. Springer, pp. 250–285.

Dua, S. and X. Du (2011). *Data Mining and Machine Learning in Cybersecurity*. 1st. Auerbach Publications.

Dubey, R. et al. (2014). "Analysis of sampling techniques for imbalanced data: An n= 648 ADNI study". In: *NeuroImage* 87, pp. 220–241.

Eberhart, R. C. and Y. Shi (1998). "Comparison between genetic algorithms and particle swarm optimization". In: *International Conference on Evolutionary Programming*. Springer, pp. 611–616.

Elhassan, T. et al. (2016). "Classification of Imbalance Data using Tomek Link (T-Link) Combined with Random Under-sampling (RUS) as a Data Reduction Method". In: *Journal of Informatics and Data Mining* 1.2.

Elkan, C. (2001). "The foundations of cost-sensitive learning". In: *International joint conference on artificial intelligence*. Vol. 17. 1, pp. 973–978.

Elsalamony, H. A. (2014). "Bank direct marketing analysis of data mining techniques". In: *International Journal of Computer Applications* 85.7.

Estabrooks, A., T. Jo, and N. Japkowicz (2004). "A Multiple Resampling Method for Learning from Imbalanced Data Sets". In: *Computational Intelligence* 20.1, pp. 18–36.

Estevez, P. A. et al. (2009). "Normalized Mutual Information Feature Selection". In: *IEEE Transactions on Neural Networks* 20.2, pp. 189–201.

Evans, M. and J. Ferryman (2005). "Group stochastic search for object detection and tracking". In: *IEEE Conference on Advanced Video and Signal Based Surveillance, 2005.* Pp. 153–158.

Fang, X. and T. Bai (2009). "Share Price Prediction Using Wavelet Transform and Ant Colony Algorithm for Parameters Optimization in SVM". In: *2009 WRI Global Congress on Intelligent Systems*. Vol. 3, pp. 288–292.

Feng, G., J.-D. Zhang, and S. S. Liao (2014). "A novel method for combining Bayesian networks, theoretical analysis, and its applications". In: *Pattern Recognition* 47.5, pp. 2057–2069.

Fogel, David B. (1995). *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. Piscataway, NJ, USA: IEEE Press.

Forman, G. (2003). "An extensive empirical study of feature selection metrics for text classification". In: *Journal of Machine Learning Research* 3.Mar, pp. 1289–1305.

Frawley, W. J., G. Piatetsky-Shapiro, and C. J. Matheus (1992). "Knowledge discovery in databases: An overview". In: *AI magazine* 13.3, p. 57.

Gabriel, K R. and R. R. Sokal (1969). "A new statistical approach to geographic variation analysis". In: *Systematic Zoology* 18.3, pp. 259–278.

Gandomi, A. and M. Haider (2015). "Beyond the hype: Big data concepts, methods, and analytics". In: *International Journal of Information Management* 35.2, pp. 137–144.

Ganganwar, V. (2012). "An overview of classification algorithms for imbalanced datasets". In: *International Journal of Emerging Technology and Advanced Engineering* 2.4, pp. 42–47.

Gao, M. et al. (2011). "A combined SMOTE and PSO based RBF classifier for two-class imbalanced problems". In: *Neurocomputing* 74.17, pp. 3456–3466.

Gao, T. (2015). "Hybrid classification approach for imbalanced datasets". MA thesis. IOWA: IOWA State University.

García, S., J. Luengo, and F. Herrera (2015). *Data preprocessing in data mining*. Springer.

García, S. et al. (2016). "Big data preprocessing: methods and prospects". In: *Big Data Analytics* 1.1, p. 9.

Garcia-Nieto, J., E. Alba, and J. Apolloni (2009). "Hybrid DE-SVM Approach for Feature Selection: Application to Gene Expression Datasets". In: *2009 2nd International Symposium on Logistics and Industrial Informatics*, pp. 1–6.

Garšva, G. and P. Danenas (2014). "Particle swarm optimization for linear support vector machines based classifier selection". In: *Nonlinear Anal: Model Control* 19.1, pp. 26–42.

Ghamisi, P. and J. A. Benediktsson (2015). "Feature selection based on hybridization of genetic algorithm and particle swarm optimization". In: *IEEE Geoscience and Remote Sensing Letters* 12.2, pp. 309–313.

Goldberg, D. E. and J. H. Holland (1988). "Genetic algorithms and machine learning". In: *Machine Learning* 3.2, pp. 95–99.

Grech-Cini, H. J. and G. T. McKee (1993). "Locating the mouth region in images of human faces". In: *Optical Tools for Manufacturing and Advanced Automation*. International Society for Optics and Photonics, pp. 458–465.

Grosan, C., A. Abraham, and M. Chis (2006). "Swarm intelligence in data mining". In: *Swarm Intelligence in Data Mining*. Springer, pp. 1–20.

Guha, S., R. Rastogi, and K. Shim (1998). "CURE: an efficient clustering algorithm for large databases". In: *ACM Sigmod Record*. Vol. 27. 2. ACM, pp. 73–84.

Guo, H. and H. L. Viktor (2004). "Learning from imbalanced data sets with boosting and data generation: the databoost-im approach". In: *ACM SIGKDD Explorations Newsletter* 6.1, pp. 30–39.

Guo, X. C. et al. (2008). “A novel LS-SVMs hyper-parameter selection based on particle swarm optimization”. In: *Neurocomputing* 71.16. Advances in Neural Information Processing (ICONIP 2006) / Brazilian Symposium on Neural Networks (SBRN 2006), pp. 3211–3215.

Hall, Mark Andrew (1999). *Correlation-based feature selection for machine learning.*

Han, C. et al. (2016). “Online Feature Selection of Class Imbalance via PA Algorithm”. In: *Journal of Computer Science and Technology* 31.4, pp. 673–682.

Han, H., W.-Y. Wang, and B.-H. Mao (2005). “Borderline-SMOTE: A New Over-sampling Method in Imbalanced Data Sets Learning”. In: *Proceedings of the 2005 International Conference on Advances in Intelligent Computing - Volume Part I*. ICIC’05. Hefei, China: Springer-Verlag, pp. 878–887.

Han, J., J. Pei, and M. Kamber (2011). *Data mining: concepts and techniques*. Elsevier.

Harel, O. and X.-H. Zhou (2007). “Multiple imputation: review of theory, implementation and software”. In: *Statistics in Medicine* 26.16, pp. 3057–3077.

Hashem, I. A. T. et al. (2015). “The rise of “big data” on cloud computing: Review and open research issues”. In: *Information Systems* 47, pp. 98–115.

Hauskrecht, M. et al. (2013). “Outlier detection for patient monitoring and alerting”. In: *Journal of Biomedical Informatics* 46.1, pp. 47–55.

Hawkins, D. (2013). *Identification of Outliers*. Springer Science & Business Media.

He, Haibo and Yunqian Ma (2013). *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons.

He, Z., S. Deng, and X. Xu (2005). “An Optimization Model for Outlier Detection in Categorical Data”. In: *Advances in Intelligent Computing*. Springer Berlin Heidelberg, pp. 400–409.

Hernandez-Carrascal, A and S Nasuto (2008). “A swarm intelligence method for feature tracking in amv derivation”. In: *Ninth International Wind Workshop*.

Hodge, V. and J. Austin (2004). “A survey of outlier detection methodologies”. In: *Artificial Intelligence Review* 22.2, pp. 85–126.

Holden, N. and A. A. Freitas (2005). “A hybrid particle swarm/ant colony algorithm for the classification of hierarchical biological data”. In: *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.* Pp. 100–107.

Holden, N. P. and A. A. Freitas (2007). "A hybrid PSO/ACO algorithm for classification". In: *Proceedings of the 9th annual conference companion on Genetic and evolutionary computation*. ACM, pp. 2745–2750.

Holland, J. H. (1975). "Adaptation in natural and artificial systems. An introductory analysis with application to biology, control, and artificial intelligence". In: *Ann Arbor, MI: University of Michigan Press*, pp. 439–444.

Hossain, M. Z. et al. (2015). "Spatial Subdivision of Gabriel Graph". In: *Advances in Swarm and Computational Intelligence*. Springer International Publishing, pp. 321–328.

Hsu, C.-W., C.-C. Chang, C.-J. Lin, et al. (2003). *A practical guide to support vector classification*.

Hsu, C.-W. and C.-J. Lin (2002). "A Simple Decomposition Method for Support Vector Machines". In: *Machine Learning* 46.1, pp. 291–314.

Huang, C.-L. (2009). "ACO-based hybrid classification system with feature subset selection and model parameters optimization". In: *Neurocomputing* 73.1, pp. 438–448.

Huang, C.-L. and J.-F. Dun (2008). "A distributed PSO–SVM hybrid system with feature selection and parameter optimization". In: *Applied Soft Computing* 8.4, pp. 1381–1391.

Huang, C.-L. and C.-J. Wang (2006). "A GA-based feature selection and parameters optimization for support vector machines". In: *Expert Systems with Applications* 31.2, pp. 231–240.

Huang, J., Y. Cai, and X. Xu (2007). "A hybrid genetic algorithm for feature selection wrapper based on mutual information". In: *Pattern Recognition Letters* 28.13, pp. 1825–1844.

Huerta, E. B., B. Duval, and J.-K. Hao (2010). "A hybrid LDA and genetic algorithm for gene selection and classification of microarray data". In: *Neurocomputing* 73.13–15, pp. 2375–2383.

Hughes, R. (2011). "Stochastic diffusion search with reinforcement learning". In: *Proc School Conference for Annual Research Projects (SCARP)*.

Hurley, S. and R. M. Whitaker (2002). "An Agent Based Approach to Site Selection for Wireless Networks". In: *Proceedings of the 2002 ACM Symposium on Applied Computing*. SAC '02. Madrid, Spain: ACM, pp. 574–577.

Jamali, I., M. Bazmara, and S. Jafari (2012). "Feature Selection in Imbalance data sets". In: *International Journal of Computer Science Issues* 9.3, pp. 42–45.

Janakiram, D., V. A. Reddy, and A. V. U. P. Kumar (2006). "Outlier detection in wireless sensor networks using Bayesian belief networks". In: *Communication System Software and Middleware, 2006. Comsware 2006. First International Conference on*. IEEE, pp. 1–6.

Japkowicz, N. and M. Shah (2011). *Evaluating learning algorithms: a classification perspective*. Cambridge University Press.

Japkowicz, N. and S. Stephen (2002). "The Class Imbalance Problem: A Systematic Study". In: *Intell. Data Anal.* 6.5, pp. 429–449.

Jashki, M.-A. et al. (2009). "An Iterative Hybrid Filter-Wrapper Approach to Feature Selection for Document Clustering". In: *Advances in Artificial Intelligence*. Springer Berlin Heidelberg, pp. 74–85.

Jeatrakul, P., K. W. Wong, and C. C. Fung (2010). "Classification of imbalanced data by combining the complementary neural network and SMOTE algorithm". In: *International Conference on Neural Information Processing*. Springer, pp. 152–159.

Jirapech-Umpai, T. and S. Aitken (2005). "Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes". In: *BMC Bioinformatics* 6.1, p. 148.

Jo, T. and N. Japkowicz (2004). "Class Imbalances Versus Small Disjuncts". In: *SIGKDD Explor. Newsl.* 6.1, pp. 40–49.

Jones, D (2002). "Constrained stochastic diffusion search". In: *Proc School Conference for Annual Research Projects (SCARP)*.

Jun-shan, T., H. Wei, and Q. Yan (2009). "Application of Genetic Algorithm in Data Mining". In: *2009 First International Workshop on Education Technology and Computer Science*. Vol. 2, pp. 353–356.

Kabir, M., M. Islam, and K. Murase (2010). "A new wrapper feature selection approach using neural network". In: *Neurocomputing* 73.16–18, pp. 3273–3283.

Kadiyala, S. S. and A. Srivastava (2011). "Data Mining For Customer Relationship Management". In: *International Business & Economics Research Journal (IBER)* 1.6.

Kanan, H. R. and K. Faez (2008). "An improved feature selection method based on ant colony optimization (ACO) evaluated on face recognition system". In: *Applied Mathematics and Computation* 205.2, pp. 716–725.

Karaboga, D (2005). *An idea based on honey bee swarm for numerical optimization*. Tech. rep. Technical report-tr06 Erciyes University Engineering Faculty Computer Engineering Department.

Karnan, M. and K. Thangavel (2007). “Automatic detection of the breast border and nipple position on digital mammograms using genetic algorithm for asymmetry approach to detection of microcalcifications”. In: *Computer Methods and Programs in Biomedicine* 87.1, pp. 12–20.

Kennedy, J. (2003). “Bare bones particle swarms”. In: *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE*, pp. 80–87.

Kennedy, J. and R. C. Eberhart (1997). “A discrete binary version of the particle swarm algorithm”. In: *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*. Vol. 5, pp. 4104–4108.

Kennedy, J. and R. Mendes (2002). “Population structure and particle swarm performance”. In: *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*. Vol. 2. IEEE, pp. 1671–1676.

Kennedy, J. et al. (2001). *Swarm intelligence*. Morgan Kaufmann.

Kennedy, J. (2011). “Particle swarm optimization”. In: *Encyclopedia of machine learning*. Springer, pp. 760–766.

Keogh, E. et al. (2001). “Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases”. In: *Knowledge and Information Systems* 3.3, pp. 263–286.

Khan, J. et al. (2001). “Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks”. In: *Nature Medicine* 7.6, p. 673.

King, M. and M. M. Al-Rifaie (2017). “Building Simple Non-identical Organic Structures with Dispersive Flies Optimisation and A* Path-finding”. In: *AISB 2017: Games and AI*, pp. 336–340.

Kira, K. and L. A. Rendell (1992). “The feature selection problem: Traditional methods and a new algorithm”. In: *Aaai*. Vol. 2, pp. 129–134.

Kononenko, Igor (1994). “Estimating attributes: Analysis and extensions of RELIEF”. In: *Machine Learning: ECML-94*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 171–182.

Kotsiantis, S., D. Kanellopoulos, P. Pintelas, et al. (2006). “Handling imbalanced datasets: A review”. In: *GESTS International Transactions on Computer Science and Engineering* 30.1, pp. 25–36.

Krawczuk, Jerzy and Tomasz Łukaszuk (2016). “The feature selection bias problem in relation to high-dimensional gene data”. In: *Artificial Intelligence in Medicine* 66, pp. 63–71.

Kriegel, H.-P., P. Kröger, and A. Zimek (2010). “Outlier detection techniques”. In: *Tutorial at KDD* 10.

Krishnanand, K. N. and D. Ghose (2009). "Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions". In: *Swarm Intelligence* 3.2, pp. 87–124.

Krishnapuram, B., S. Yu, and R. B. Rao (2011). *Cost-sensitive Machine Learning*. CRC Press.

Kubat, M., R. C. Holte, and S. Matwin (1998). "Machine Learning for the Detection of Oil Spills in Satellite Radar Images". In: *Machine Learning* 30.2, pp. 195–215.

Kubat, M. and S. Matwin (1997). "Addressing the Curse of Imbalanced Training Sets: One-Sided Selection". In: *Proceedings of the Fourteenth International Conference on Machine Learning*. Morgan Kaufmann, pp. 179–186.

Kumar, M. et al. (2010). "Genetic algorithm: Review and application". In: *International Journal of Information Technology and Knowledge Management* 2.2, pp. 451–454.

Kumar, R, R Srivastava, and S Srivastava (2015). "Detection and classification of cancer from microscopic biopsy images using clinically significant and biologically interpretable features". In: *Journal of Medical Engineering* 2015.

Kuo, R. J. and L. M. Lin (2010). "Application of a hybrid of genetic algorithm and particle swarm optimization algorithm for order clustering". In: *Decision Support Systems* 49.4, pp. 451–462.

Laurikkala, J. (2001). "Improving identification of difficult small classes by balancing class distribution". In: *Conference on Artificial Intelligence in Medicine in Europe*. Springer, pp. 63–66.

Leardi, R. (2000). "Application of genetic algorithm-PLS for feature selection in spectral data sets". In: *Journal of Chemometrics* 14.5–6, pp. 643–655.

Lee, K. J. and J. A. Simpson (2014). "Introduction to multiple imputation for dealing with missing data". In: *Respirology* 19.2, pp. 162–167.

Lesperance, Y. et al. (2001). "AAAI 2000 Workshop Reports". In: *AI Magazine* 22.1, p. 127.

Li, J., H. Liu, and L. Wong (2003). "Mean-entropy discretized features are effective for classifying high-dimensional bio-medical data". In: *Proceedings of the 3rd International Conference on Data Mining in Bioinformatics*. Springer-Verlag, pp. 17–24.

Li, P., E. A. Stuart, and D. B. Allison (2015). "Multiple imputation: a flexible tool for handling missing data". In: *Jama* 314.18, pp. 1966–1967.

Li, S., X. Wu, and M. Tan (2008). "Gene selection using hybrid particle swarm optimization and genetic algorithm". In: *Soft Computing* 12.11, pp. 1039–1048.

Li, T., C. Zhang, and M. Ogihara (2004). "A Comparative Study of Feature Selection and Multiclass Classification Methods for Tissue Classification Based on Gene Expression". In: *Bioinformatics* 20.15, pp. 2429–2437.

Li, X. and M. Yin (2012). "Application of differential evolution algorithm on self-potential data". In: *PloS one* 7.12, e51199.

Li, X. Z. and J. M. Kong (2014). "Application of GA–SVM method with parameter optimization for landslide development prediction". In: *Natural Hazards and Earth System Sciences* 14.3, pp. 525–533.

Li, Xiangtao and Minghao Yin (2013). "Multiobjective binary biogeography based optimization for feature selection using gene expression data". In: *IEEE Transactions on NanoBioscience* 12.4, pp. 343–353.

Lin, H.-T. and C.-J. Lin (2003). "A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods". In: *Neural Computation*, pp. 1–32.

Lin, S.-W. et al. (2008). "Particle swarm optimization for parameter determination and feature selection of support vector machines". In: *Expert Systems with Applications* 35.4, pp. 1817–1824.

Ling, C. X. and C. Li (1998). "Data Mining for Direct Marketing: Problems and Solutions". In: *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*. KDD'98. New York, NY: AAAI Press, pp. 73–79.

Liu, Bo et al. (2005). "Improved particle swarm optimization combined with chaos". In: *Chaos, Solitons & Fractals* 25.5, pp. 1261–1271.

Liu, D. et al. (2014). "Short-term wind speed forecasting using wavelet transform and support vector machines optimized by genetic algorithm". In: *Renewable Energy* 62, pp. 592–597.

Liu, H., J. Li, and L. Wong (2002). "A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns". In: *Genome Informatics* 13, pp. 51–60.

Liu, H. and L. Yu (2005). "Toward integrating feature selection algorithms for classification and clustering". In: *IEEE Transactions on Knowledge and Data Engineering* 17.4, pp. 491–502.

Liu, H. et al. (2013). "An experimental investigation of two Wavelet-MLP hybrid frameworks for wind speed prediction using GA and PSO optimization". In: *International Journal of Electrical Power & Energy Systems* 52, pp. 161–173.

Liu, L. and M. T. Özsu (2009). "CLARANS (Clustering Large Applications Based Upon Randomized Search)". In: *Encyclopedia of Database Systems*. Springer US, pp. 330–330.

Liu, P. et al. (2010). "Classifying skewed data streams based on reusing data". In: *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*. Vol. 4, pp. V4–90–V4–93.

Liu, S., C.-Y. Jia, and H. Ma (2005). "A new weighted support vector machine with GA-based parameter selection". In: *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*. Vol. 7. IEEE, pp. 4351–4355.

Liu, X., A. Krishnan, and A. Mondry (2005). "An Entropy-based gene selection method for cancer classification using microarray data". In: *BMC Bioinformatics* 6.1, p. 76.

Liu, X. Y., J. Wu, and Z. H. Zhou (2009). "Exploratory Undersampling for Class-Imbalance Learning". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.2, pp. 539–550.

Liu, Y. and Y. F. Zheng (2006). "FS_SFS: A novel feature selection method for support vector machines". In: *Pattern recognition* 39.7, pp. 1333–1345.

Liu, Y. et al. (2011). "An improved particle swarm optimization for feature selection". In: *Journal of Bionic Engineering* 8.2, pp. 191–200.

Longadge, R. and S. Dongre (2013). "Class imbalance problem in data mining review". In: *arXiv preprint arXiv:1305.1707*.

Lusa, L. et al. (2013). "SMOTE for high-dimensional class-imbalanced data". In: *BMC Bioinformatics* 14.1, p. 106.

Magnani, M. (2004). "Techniques for dealing with missing data in knowledge discovery tasks". In: *Obtido http://magnanim. web. cs. unibo. it/index. html* 15.01, p. 2007.

Maimon, O and L Rokach (2010). *Data Mining and Knowledge Discovery Handbook*. 2nd. Springer Publishing Company, Incorporated.

Maldonado, S., R. Weber, and F. Famili (2014). "Feature selection for high-dimensional class-imbalanced data sets using Support Vector Machines". In: *Information Sciences* 286, pp. 228–246.

Maloof, M. A. (2003). "Learning when data sets are imbalanced and when costs are unequal and unknown". In: *ICML-2003 workshop on learning from imbalanced data sets II*. Vol. 2, pp. 2–1.

Man, K.-F., K.-S. Tang, and S. Kwong (1996). "Genetic algorithms: concepts and applications [in engineering design]". In: *IEEE Transactions on Industrial Electronics* 43.5, pp. 519–534.

Mao, K. Z. (2004). "Feature subset selection for support vector machines through discriminative function pruning analysis". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 34.1, pp. 60–67.

Marsh, L. and C. Onof (2008). "Stigmergic epistemology, stigmergic cognition". In: *Cognitive Systems Research* 9.1, pp. 136–149.

Martens, D, B Baesens, and T Fawcett (2011). "Editorial survey: swarm intelligence for data mining". In: *Machine Learning* 82.1, pp. 1–42.

Mavrovouniotis, M, C Li, and S Yang (2017). "A survey of swarm intelligence for dynamic optimization: Algorithms and applications". In: *Swarm and Evolutionary Computation* 33, pp. 1–17.

Mazurowski, M. A. et al. (2008). "Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance". In: *Neural Networks* 21.2. Advances in Neural Networks Research: IJCNN '07, pp. 427–436.

McCluskey, A. and A. G. Lalkhen (2007). "Statistics II: Central tendency and spread of data". In: *Continuing Education in Anaesthesia, Critical Care & Pain* 7.4, pp. 127–130.

Melgani, F. and Y. Bazi (2008). "Classification of electrocardiogram signals with support vector machines and particle swarm optimization". In: *IEEE Transactions on Information Technology in Biomedicine* 12.5, pp. 667–677.

Merkle, D. and M. Middendorf (2014). "Swarm intelligence". In: *Search Methodologies*. Springer, pp. 213–242.

Mitchell, T. M. (1997). *Machine Learning*. 1st ed. McGraw-Hill, Inc.

Mladenic, D. and M. Grobelnik (1999). "Feature selection for unbalanced class distribution and Naive Bayes". In: *Proceedings of the 16th International Conference on Machine Learning (ICML*. Morgan Kaufmann Publishers, pp. 258–267.

Moro, S., P. Cortez, and P. Rita (2014). "A data-driven approach to predict the success of bank telemarketing". In: *Decision Support Systems* 62, pp. 22–31.

Moro, S., R. Laureano, and P. Cortez (2011). "Using data mining for bank direct marketing: An application of the crisp-dm methodology". In: *Proceedings of European Simulation and Modelling Conference-ESM'2011*. Eurosis, pp. 117–121.

Muthiah, A., A. Rajkumar, and R. Rajkumar (2016). "Hybridization of Artificial Bee Colony algorithm with Particle Swarm Optimization algorithm for flexible Job Shop Scheduling". In: *2016 International Conference on Energy Efficient Technologies for Sustainability (ICEETS)*, pp. 896–903.

Myatt, D. and J. M. Bishop (2003). "Data driven stochastic diffusion networks for robust high-dimensionality manifold estimation - more fun than you can shake a hyperplane at". In: *Proc. School Conference for Annual Research Projects (SCARP)*.

Myatt, D. R., S. Nasuto, and J. M. Bishop (2006). *Alternative recruitment strategies for stochastic diffusion search*. Artificial Life X, Bloomington USA.

Napierala, K. and J. Stefanowski (2016). "Types of minority class examples and their influence on learning classifiers from imbalanced data". In: *Journal of Intelligent Information Systems* 46.3, pp. 563–597.

Nasuto, S. (1999). "Resource allocation analysis of the stochastic diffusion search". PhD thesis. University of Reading.

Nasuto, S. and J. M. Bishop (1999). "Convergence analysis of stochastic diffusion search". In: *Parallel Algorithms and Application* 14.2, pp. 89–107.

Nazir, M., A. Majid-Mirza, and S. Ali-Khan (2014). "PSO-GA based optimized feature selection using facial and clothing information for gender classification". In: *Journal of Applied Research and Technology* 12.1, pp. 145–152.

Nemati, S. et al. (2009). "A novel ACO–GA hybrid algorithm for feature selection in protein function prediction". In: *Expert Systems with Applications* 36.10, pp. 12086–12094.

Nesmachnow, Sergio (2014). "An overview of metaheuristics: accurate and efficient methods for optimisation". In: *International Journal of Metaheuristics* 3.4, pp. 320–347.

Nguyen, H. T., K. Franke, and S. Petrovic (2010). "Towards a generic feature-selection measure for intrusion detection". In: *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, pp. 1529–1532.

Nircan, A. (2006). "Stochastic diffusion search and voting methods". PhD thesis. Bogaziki University.

Ofek, N. et al. (2017). "Fast-CBUS: A fast clustering-based undersampling method for addressing the class imbalance problem". In: *Neurocomputing* 243, pp. 88–102.

Oh, I.-S., J.-S. Lee, and B.-R. Moon (2004). "Hybrid genetic algorithms for feature selection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.11, pp. 1424–1437.

Omran, M., A. P. Engelbrecht, and A. Salman (2005a). "Particle swarm optimization method for image clustering". In: *International Journal of Pattern Recognition and Artificial Intelligence* 19.03, pp. 297–321.

Omran, M. G. H., A. P. Engelbrecht, and A. Salman (2005b). "Differential evolution methods for unsupervised image classification". In: *2005 IEEE Congress on Evolutionary Computation*. Vol. 2, pp. 966–973.

Özgür, Arzucan, Levent Özgür, and Tunga Güngör (2005). "Text Categorization with Class-Based and Corpus-Based Keyword Selection". In: *Computer and Information Sciences - ISCIS 2005: 20th International Symposium, Istanbul, Turkey, October 26-28, 2005. Proceedings*. Springer Berlin Heidelberg, pp. 606–615.

Padmaja, T. M. et al. (2007). "Unbalanced data classification using extreme outlier elimination and sampling techniques for fraud detection". In: *15th International Conference on Advanced Computing and Communications (ADCOM 2007)*, pp. 511–516.

Pan, L. and Y. Luo (2010). "Parameters Selection of Support Vector Machine Using an Improved PSO Algorithm". In: *2010 Second International Conference on Intelligent Human-Machine Systems and Cybernetics*. Vol. 2, pp. 196–199.

Parpinelli, R. S. and H. S. Lopes (2011). "New inspirations in swarm intelligence: a survey". In: *International Journal of Bio-Inspired Computation* 3.1, pp. 1–16.

Parpinelli, R. S., H. S. Lopes, and A. A. Freitas (2002). "Data mining with an ant colony optimization algorithm". In: *IEEE Transactions on Evolutionary Computation* 6.4, pp. 321–332.

Patcha, A. and J.-M. Park (2007). "An overview of anomaly detection techniques: Existing solutions and latest technological trends". In: *Computer Networks* 51.12, pp. 3448–3470.

Pelckmans, K. et al. (2005). "Handling missing values in support vector machine classifiers". In: *Neural Networks* 18.5–6, pp. 684–692.

Peng, H., F. Long, and C. Ding (2005). "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.8, pp. 1226–1238.

Poli, R. (2008). "Analysis of the Publications on the Applications of Particle Swarm Optimisation". In: *J. Artif. Evol. App.* 2008, 4:1–4:10.

Poolsawad, N. et al. (2012). "Handling missing values in data mining - A case study of heart failure dataset". In: *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 2934–2938.

Pourbasheer, E. et al. (2009). "Application of genetic algorithm-support vector machine (GA-SVM) for prediction of BK-channels activity". In: *European journal of medicinal chemistry* 44.12, pp. 5023–5028.

Prabusankarlal, K. M., P. Thirumoorthy, and R. Manavalan (2017). "Classification of breast masses in ultrasound images using self-adaptive differential evolution extreme learning machine and rough set feature selection". In: *Journal of Medical Imaging* 4.2, p. 024507.

Provost, F. (2000). "Machine learning from imbalanced data sets 101". In: *Proceedings of the AAAI'2000 workshop on imbalanced data sets*, pp. 1–3.

Punch III, W. F. et al. (1993). "Further Research on Feature Selection and Classification Using Genetic Algorithms." In: *ICGA*, pp. 557–564.

Rahman, M. M. and D. N. Davis (2013). "Addressing the class imbalance problem in medical datasets". In: *International Journal of Machine Learning and Computing* 3.2, p. 224.

Ramos, V., F. Muge, and P. Pina (2002). "Self-Organized Data and Image Retrieval as a Consequence of Inter-Dynamic Synergistic Relationships in Artificial Ant Colonies." In: *HIS* 87, pp. 500–512.

Rao, R. V. and V. Patel (2013). "Multi-objective optimization of heat exchangers using a modified teaching-learning-based optimization algorithm". In: *Applied Mathematical Modelling* 37.3, pp. 1147–1162.

Raymer, M. L. et al. (2000). "Dimensionality reduction using genetic algorithms". In: *IEEE Transactions on Evolutionary Computation* 4.2, pp. 164–171.

Remli, M. A. et al. (2017). "K-Means Clustering with Infinite Feature Selection for Classification Tasks in Gene Expression Data". In: *11th International Conference on Practical Applications of Computational Biology & Bioinformatics*. Springer International Publishing, pp. 50–57.

Reunanen, J (2003). "Overfitting in making comparisons between variable selection methods". In: *Journal of Machine Learning Research* 3.Mar, pp. 1371–1382.

Reynolds, C. W. (1987). "Flocks, Herds and Schools: A Distributed Behavioral Model". In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '87. New York, NY, USA: ACM, pp. 25–34.

Rodan, A. and H. Faris (2015). "Echo State Network with SVM-readout for customer churn prediction". In: *2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, pp. 1–5.

Rodan, A. et al. (2014). "A support vector machine approach for churn prediction in telecom industry". In: *International Information Institute (Tokyo). Information* 17.8, p. 3961.

Saeys, Y., I. Inza, and P. Larrañaga (2007). "A review of feature selection techniques in bioinformatics". In: *Bioinformatics* 23.19, pp. 2507–2517.

Sahare, M. and H. Gupta (2012). "A review of multi-class classification for imbalanced data". In: *International Journal of Advanced Computer Research* 2.5, pp. 160–164.

Salama, K. M. and A. M. Abdelbar (2016). "Using Ant Colony Optimization to Build Cluster-Based Classification Systems". In: *Swarm Intelligence: 10th International Conference, ANTS 2016, Brussels, Belgium, September 7–9, 2016, Proceedings*. Springer International Publishing, pp. 210–222.

Salamanos, N. et al. (2010). "Advertising Network Formation Based on Stochastic Diffusion Search and Market Equilibria". In: *Proceedings of the 28th ACM International Conference on Design of Communication*. SIGDOC '10. Sao Carlos, Sao Paulo, Brazil: ACM, pp. 81–87.

Samadzadegan, F., A. Soleymani, and R. A. Abbaspour (2010). "Evaluation of Genetic Algorithms for tuning SVM parameters in multi-class problems". In: *2010 11th International Symposium on Computational Intelligence and Informatics (CINTI)*, pp. 323–328.

Sánchez, J. S.r, F. Pla, and F. J. Ferri (1997). "Prototype selection for the nearest neighbour rule through proximity graphs". In: *Pattern Recognition Letters* 18.6, pp. 507–513.

Schölkopf, B., K. Tsuda, and J.-P. Vert (2004). *Kernel methods in computational biology*. MIT press.

Sebban, M. and R. Nock (2002). "A hybrid filter/wrapper approach of feature selection using information theory". In: *Pattern Recognition* 35.4, pp. 835–846.

Seiffert, C. et al. (2010). "RUSBoost: A Hybrid Approach to Alleviating Class Imbalance". In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 40.1, pp. 185–197.

Senawi, A., H.-L. Wei, and S. A. Billings (2017). "A new maximum relevance-minimum multicollinearity (MRmMC) method for feature selection and ranking". In: *Pattern Recognition* 67, pp. 47–61.

Sessa, J. and D. Syed (2016). "Techniques to deal with missing data". In: *2016 5th International Conference on Electronic Devices, Systems and Applications (ICEDSA)*, pp. 1–4.

Shang, W. et al. (2007). “A novel feature selection algorithm for text categorization”. In: *Expert Systems with Applications* 33.1, pp. 1–5.

Sharma, A. and K. K. Paliwal (2008). “Cancer classification by gradient LDA technique using microarray gene expression data”. In: *Data & Knowledge Engineering* 66.2, pp. 338–347.

Shelokar, P. S., V. K. Jayaraman, and B. D. Kulkarni (2004). “An ant colony classifier system: application to some process engineering problems”. In: *Computers & Chemical Engineering* 28.9, pp. 1577–1584.

Shi, X. H. et al. (2003). “Hybrid evolutionary algorithms based on PSO and GA”. In: *Evolutionary Computation, 2003. CEC’03. The 2003 Congress on*. Vol. 4. IEEE, pp. 2393–2399.

Shi, Y. (2012). *Innovations and Developments of Swarm Intelligence Applications*. IGI Global.

Silva, R. G. O., M. W. de Souza Ribeiro, and L. R. do Amaral (2013). “Building high level knowledge from high dimensionality biological dataset (NCI60) using Genetic Algorithms and feature selection strategies”. In: *2013 IEEE Congress on Evolutionary Computation*, pp. 578–583.

Singh, Manoj Kumar et al. (2016). *Effective Big Data Management and Opportunities for Implementation*. IGI Global.

Singhi, S. K. and H. Liu (2006). “Feature subset selection bias for classification learning”. In: *Proceedings of the 23rd International Conference on Machine learning*. ACM, pp. 849–856.

Sokolova, M., N. Japkowicz, and S. Szpakowicz (2006). “Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation”. In: *AI 2006: Advances in Artificial Intelligence*. Springer, pp. 1015–1021.

Sörensen, K. (2015). “Metaheuristics—the metaphor exposed”. In: *International Transactions in Operational Research* 22.1, pp. 3–18.

Sousa, T., A. Silva, and A. Neves (2004). “Particle swarm based data mining algorithms for classification tasks”. In: *Parallel Computing* 30.5, pp. 767–783.

Sowah, R. A. et al. (2016). “New Cluster Undersampling Technique for Class Imbalance Learning”. In: *International Journal of Machine Learning and Computing* 6.3, p. 205.

Stefanowski, J. (2013). “Overlapping, Rare Examples and Class Decomposition in Learning Classifiers from Imbalanced Data”. In: *Emerging Paradigms in Machine Learning*. Springer Berlin Heidelberg, pp. 277–306.

Storn, R. (1996). "On the usage of differential evolution for function optimization". In: *Fuzzy Information Processing Society, 1996. NAFIPS., 1996 Biennial Conference of the North American*. IEEE, pp. 519–523.

Storn, R. and K. Price (1997). "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces". In: *Journal of Global Optimization* 11.4, pp. 341–359.

Su, C.-T. and Y.-H. Hsiao (2007). "An evaluation of the robustness of MTS for imbalanced data". In: *IEEE Transactions on Knowledge and Data Engineering* 19.10.

Su, Y. et al. (2003). "RankGene: identification of diagnostic genes based on expression data". In: *Bioinformatics* 19.12, pp. 1578–1579.

Subbulakshmi, C. V. and S. N. Deepa (2015). "Medical dataset classification: a machine learning paradigm integrating particle swarm optimization with extreme learning machine classifier". In: *The Scientific World Journal* 2015.

Sudheer, C. et al. (2014). "A hybrid SVM-PSO model for forecasting monthly streamflow". In: *Neural Computing and Applications* 24.6, pp. 1381–1389.

Sun, Y., A. K. C. Wong, and M. S. Kamel (2009). "Classification Of Imbalanced Data: A Review". In: *International Journal of Pattern Recognition and Artificial Intelligence* 23.04, pp. 687–719.

Taft, L. M. et al. (2009). "Countering imbalanced datasets to improve adverse drug event predictive models in labor and delivery". In: *Journal of Biomedical Informatics* 42.2, pp. 356–364.

Tan, Y. and Z.-y. Zheng (2013). "Research Advance in Swarm Robotics". In: *Defence Technology* 9.1, pp. 18–39.

Tan, Ying, Yuhui Shi, and Ben Niu (2016). *Advances in swarm intelligence*. Springer.

Tang, Y. and Y.-Q. Zhang (2006). "Granular SVM with Repetitive Undersampling for Highly Imbalanced Protein Homology Prediction". In: *2006 IEEE International Conference on Granular Computing*, pp. 457–460.

Tang, Y. et al. (2009). "SVMs Modeling for Highly Imbalanced Classification". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.1, pp. 281–288.

Teodorović, D. (2008). "Swarm intelligence systems for transportation engineering: Principles and applications". In: *Transportation Research Part C: Emerging Technologies* 16.6, pp. 651–667.

Thai-Nghe, N., Z. Gantner, and L. Schmidt-Thieme (2010). "Cost-sensitive learning methods for imbalanced data". In: *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8.

Ting, K. M. (2000). “A Comparative Study of Cost-Sensitive Boosting Algorithms”. In: *Proceedings of the Seventeenth International Conference on Machine Learning*. ICML ’00. Morgan Kaufmann Publishers Inc., pp. 983–990.

Tiwari, D. (2014). “Handling Class Imbalance Problem Using Feature Selection”. In: *International Journal of Advanced Research in Computer Science & Technology* 2.2, pp. 516–520.

Toksari, M. D. (2006). “Ant colony optimization for finding the global minimum”. In: *Applied Mathematics and Computation* 176.1, pp. 308–316.

Tomek, I. (1976). “Two Modifications of CNN”. In: *IEEE Transactions on Systems, Man, and Cybernetics* SMC-6.11, pp. 769–772.

Tran, B. et al. (2016). “Investigation on particle swarm optimisation for feature selection on high-dimensional data: Local search and selection bias”. In: *Connection Science* 28.3, pp. 270–294.

Trelea, I. C. (2003). “The particle swarm optimization algorithm: convergence analysis and parameter selection”. In: *Information Processing Letters* 85.6, pp. 317–325.

Tu, C.-J. et al. (2007). “Feature selection using PSO-SVM”. In: *International Journal of Computer Science*.

Unler, A., A. Murat, and R. B. Chinnam (2011). “mr 2 PSO: a maximum relevance minimum redundancy feature selection method based on swarm intelligence for support vector machine classification”. In: *Information Sciences* 181.20, pp. 4625–4641.

Vajiramedhin, C. and A. Suebsing (2014). “Feature selection with data balancing for prediction of bank telemarketing”. In: *Applied Mathematical Sciences* 8.114, pp. 5667–5672.

Valdés, J. (2004). “Building virtual reality spaces for visual data mining with hybrid evolutionary-classical optimization: Application to microarray gene expression data”. In: *2004 IASTED International Joint Conference on Artificial Intelligence and Soft Computing, ASC*, pp. 161–166.

Vuk, Miha and Tomaz Curk (2006). “ROC curve, lift chart and calibration plot”. In: *Metodoloski zvezki* 3.1, p. 89.

Wallace, B. C. et al. (2011). “Class Imbalance, Redux”. In: *2011 IEEE 11th International Conference on Data Mining*, pp. 754–763.

Walton, S. et al. (2011). “Modified cuckoo search: a new gradient free optimisation algorithm”. In: *Chaos, Solitons & Fractals* 44.9, pp. 710–718.

Wang, H.-b., Y. Yu, and Z. Liu (2005). “SVM classifier incorporating feature selection using GA for spam detection”. In: *Embedded and Ubiquitous Computing – EUC 2005*, pp. 1147–1154.

Wang, H. Y. (2008). "Combination approach of SMOTE and biased-SVM for imbalanced datasets". In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp. 228–231.

Wang, J. et al. (2012). "An annual load forecasting model based on support vector regression with differential evolution algorithm". In: *Applied Energy* 94, pp. 65–70.

Wang, K.-J. et al. (2014). "A hybrid classifier combining SMOTE with PSO to estimate 5-year survivability of breast cancer patients". In: *Applied Soft Computing* 20. Hybrid intelligent methods for health technologies, pp. 15–24.

Wang, S. (2011). "Ensemble diversity for class imbalance learning". PhD thesis. University of Birmingham.

Wang, W. et al. (2003). "Determination of the spread parameter in the Gaussian kernel for classification and regression". In: *Neurocomputing* 55.3, pp. 643–663.

Wasikowski, M. and X.-w. Chen (2010). "Combating the small sample class imbalance problem using feature selection". In: *IEEE Transactions on Knowledge and Data Engineering* 22.10, pp. 1388–1400.

Wei, C. and Y. Hui-Mei (2014). "An improved GA-SVM algorithm". In: *2014 9th IEEE Conference on Industrial Electronics and Applications*, pp. 2137–2141.

Weiss, G., K. McCarthy, and B. Zabar (2007). "Cost-Sensitive Learning vs. Sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs?" In: *DMIN*. CSREA Press, pp. 35–41.

Weiss, G. M. (2004). "Mining with Rarity: A Unifying Framework". In: *SIGKDD Explor. Newsl.* 6.1, pp. 7–19.

Weiss, G. M. and F. Provost (2001). *The effect of class distribution on classifier learning: an empirical study*. Tech. rep. Rutgers Univ.

Weng, S.-S. and Y.-H. Liu (2006). "Mining time series data for segmentation by using Ant Colony Optimization". In: *European Journal of Operational Research* 173.3, pp. 921–937.

Wilson, D. L. (1972). "Asymptotic properties of nearest neighbor rules using edited data". In: *IEEE Transactions on Systems, Man, and Cybernetics* 3, pp. 408–421.

Witten, I. H. and E. Frank (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Wolpert, D. H. and W. G. Macready (1997). "No free lunch theorems for optimization". In: *IEEE Transactions on Evolutionary Computation* 1.1, pp. 67–82.

Wu, C.-H. et al. (2007). "A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy". In: *Expert Systems with Applications* 32.2, pp. 397–408.

Wu, Y. et al. (2017). "Large-Scale Online Feature Selection for Ultra-High Dimensional Sparse Data". In: *ACM Trans. Knowl. Discov. Data* 11.4, pp. 1–22.

Xie, Y. et al. (2018). "An Improved Multi-label Relief Feature Selection Algorithm for Unbalanced Datasets". In: *Advances in Intelligent Systems and Interactive Applications*. Springer International Publishing, pp. 141–151.

Xing, E. P., M. I. Jordan, R. M. Karp, et al. (2001). "Feature selection for high-dimensional genomic microarray data". In: *ICML*. Vol. 1. Citeseer, pp. 601–608.

Xiong, M., X. Fang, and J. Zhao (2001). "Biomarker identification by feature wrappers". In: *Genome Research* 11.11, pp. 1878–1887.

Xue, B., M. Zhang, and W. N. Browne (2013). "Particle swarm optimization for feature selection in classification: A multi-objective approach". In: *IEEE Transactions on Cybernetics* 43.6, pp. 1656–1671.

Xue, B. et al. (2016). "A Survey on Evolutionary Computation Approaches to Feature Selection". In: *IEEE Transactions on Evolutionary Computation* 20.4, pp. 606–626.

Yang, J. and V. Honavar (1998). "Feature subset selection using a genetic algorithm". In: *Feature Extraction, Construction and Selection*. Springer, pp. 117–136.

Yang, X.-S. (2014). "Swarm intelligence based algorithms: a critical analysis". In: *Evolutionary Intelligence* 7.1, pp. 17–28.

Yang, X.-S. and S. Deb (2009). "Cuckoo search via Lévy flights". In: *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. IEEE, pp. 210–214.

Yang, Xin-She (2012). "Swarm-based metaheuristic algorithms and no-free-lunch theorems". In: *Theory and New Applications of Swarm Intelligence*. InTech.

Yang, Xin-She and Mehmet Karamanoglu (2013). "Swarm intelligence and bio-inspired computation: an overview". In: *Swarm Intelligence and Bio-Inspired Computation*. Elsevier, pp. 3–23.

Ye, N. et al. (2012). "Optimizing F-measure: A Tale of Two Approaches". In: *CoRR* abs/1206.4625.

Yen, S.-J. and Y.-S. Lee (2009). "Cluster-based under-sampling approaches for imbalanced data distributions". In: *Expert Systems with Applications* 36.3, Part 1, pp. 5718–5727.

Yen, S.-J. and Y.-S. Lee (2006). "Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset". In: *Intelligent Control and Automation*. Springer, pp. 731–740.

You, M. and G.-Z. Li (2011). "Feature selection for multi-class problems by using pairwise-class and all-class techniques". In: *International Journal of General Systems* 40.04, pp. 381–394.

Yu, E. and S. Cho (2003). "GA-SVM wrapper approach for feature subset selection in keystroke dynamics identity verification". In: *Proceedings of the International Joint Conference on Neural Networks, 2003.* Vol. 3. IEEE, pp. 2253–2257.

Yu, H., J. Ni, and J. Zhao (2013). "ACOSampling: An Ant Colony Optimization-based Undersampling Method for Classifying Imbalanced DNA Microarray Data". In: *Neurocomput.* 101, pp. 309–318.

Zadrozny, Bianca and Charles Elkan (2001). "Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers". In: *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*. Vol. 1. Citeseer, pp. 609–616.

Zahran, Bilal and Ghassan Kanan (2009). "Text Feature Selection using Particle Swarm Optimization Algorithm". In: *World Applied Sciences Journal* 7, pp. 69–74.

Zhang, D., J. Wang, and X. Zhao (2015). "Estimating the Uncertainty of Average F1 Scores". In: *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*. ICTIR '15. Northampton, Massachusetts, USA: ACM, pp. 317–320.

Zhang, T., R. Ramakrishnan, and M. Livny (1996). "BIRCH: an efficient data clustering method for very large databases". In: *ACM Sigmod Record*. Vol. 25. 2. ACM, pp. 103–114.

Zhang, X. and Y. Li (2013). "A Positive-biased Nearest Neighbour Algorithm for Imbalanced Classification". In: *Advances in Knowledge Discovery and Data Mining*. Springer Berlin Heidelberg, pp. 293–304.

Zhang, X. L., X. F. Chen, and Z. J. He (2010). "An ACO-based algorithm for parameter optimization of support vector machines". In: *Expert Systems with Applications* 37.9, pp. 6618–6628.

Zhang, X.-L. et al. (2008). "A grid-based ACO algorithm for parameters optimization in support vector machines". In: *2008 IEEE International Conference on Granular Computing*, pp. 805–808.

Zhang, Y., N. Meratnia, and P. Havinga (2010). "Outlier detection techniques for wireless sensor networks: A survey". In: *IEEE Communications Surveys & Tutorials* 12.2, pp. 159–170.

Zhang, Y. and D. Wang (2013). "A cost-sensitive ensemble method for class-imbalanced datasets". In: *Abstract and Applied Analysis*. Vol. 2013. Hindawi Publishing Corporation.

Zhang, Y. et al. (2015). "Feature selection algorithm based on bare bones particle swarm optimization". In: *Neurocomputing* 148, pp. 150–157.

Zhang, Yu, Xiaopeng Xie, and Taobo Cheng (2010). "Application of PSO and SVM in image classification". In: *2010 3rd International Conference on Computer Science and Information Technology*. Vol. 6, pp. 629–631.

Zhang, Z. et al. (2014). "On swarm intelligence inspired self-organized networking: its bionic mechanisms, designing principles and optimization approaches". In: *IEEE Communications Surveys & Tutorials* 16.1, pp. 513–537.

Zhao, G. and Y. Wu (2016). "Feature Subset Selection for Cancer Classification Using Weight Local Modularity". In: *Scientific Reports* 6.

Zheng, Z., X. Wu, and R. Srihari (2004). "Feature Selection for Text Categorization on Imbalanced Data". In: *SIGKDD Explor. Newsl.* 6.1, pp. 80–89.

Zhou, Z.-H. and X.-Y. Liu (2006). "Training cost-sensitive neural networks with methods addressing the class imbalance problem". In: *IEEE Transactions on Knowledge and Data Engineering* 18.1, pp. 63–77.

Zhu, G.-q., S.-r. Liu, and J.-s. Yu (2002). "Support vector machine and its applications to function approximation". In: *Journal-East China University Of Science And Technology* 28.5, pp. 555–559.

Zhu, X. and I. Davidson (2007). *Knowledge Discovery and Data Mining: Challenges and Realities*. IGI Global.

Zięba, M. et al. (2014). "Boosted SVM for extracting rules from imbalanced data in application to prediction of the post-operative life expectancy in the lung cancer patients". In: *Applied Soft Computing* 14. Special issue on hybrid intelligent methods for health technologies, pp. 99–108.