Palestine Polytechnic University

Deanship of Graduate Studies and Scientific Research

Master of informatics

Using PSO and Weighted SVM to Enhance Imbalanced Data Classification

Submitted by:

Ghadeer Waheed Hassouneh

Supervisor:

Dr. Mohammed Aldasht

Thesis submitted in partial fulfillment of requirements of the degree

Master of Science in Informatics

August, 2019

The undersigned hereby certify that they have read, examined and recommended to the Deanship of Graduate Studies and Scientific Research at Palestine Polytechnic University the approval of a thesis entitled "Using WSVM and PSO to Enhance Imbalanced Data Classification ", submitted by Ghadeer Waheed Hassouneh in partial fulfillment of the requirements for the degree of Master in Informatics.

Graduate Advisory Committee

Dr. Mohammed Aldasht (Supervisor), Palestine Polytechnic University.

Signature:_____ Date:_____

Dr. Hashem Tamimi (Internal Examiner), Palestine Polytechnic University.

Signature:_____ Date:_____

Dr. Mohammed Awad (External Examiner), Arab American University, Palestine.

Signature:_____ Date:_____

Thesis approved

Dr. Murad Abusubaih
Dean of Graduate Studies and Scientifc Research
Palestine Polytechnic University

Signature:_____ Date:_____

DECLARATION

I declare that the master thesis entitled "Using PSO and WSVM to Enhance Imbalanced Data Classification" is my own original work, and hereby certify that unless stated, all work contained within this thesis is my own independent research and has not been submitted for the award of any other degree at any institution, except where due acknowledgement is made in the text.

Ghadeer Waheed Hassouneh

Signature:_____ Date:_____

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for the master degree in Informatics at Palestine Polytechnic University, I agree that the library shall make it available to borrowers under rules of the library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgement of the source is made.

Permission for extensive quotation from, reproduction, or publication of this thesis may be granted by my main supervisor, or in his absence, by the Dean of Graduate Studies and Scientific Research when, in the opinion of either, the proposed use of the material is for scholarly purposes.

Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Ghadeer Waheed Hassouneh

Signature:_____ Date:_____

Abstract

Different machine learning classifiers work well only when the training data is balanced. That is, when the number of positive examples is similar to the number of negative examples. If the training data is not balanced, the performance of the classifiers will degrade.

This thesis presents an approach for handling the two-class classification problem in classifying imbalanced datasets using weighted support vector machine (WSVM) with particle swarm optimization algorithm (PSO). This study implemented other algorithms such as Random Forest (RF), support vector machine (SVM) and WSVM to compare their results with our implemented enhanced weighted SVM (EWSVM) results. Five benchmarks datasets, with different imbalance ratios, were used to evaluate our approach. The classification performance was evaluated using four measurements which are; Area under curve (AUC), Sensitivity, Specificity and Accuracy.

This approach was compared with other implemented algorithms; Random Forest, SVM and weighted SVM and enhanced the classification performance for Ecoli4 and Poker datasets in terms of AUC.

Our results was compared with other researches in the literature that uses SVM, weighted SVM, SVM with PSO and ensemble AdaBoost. When classifying Pima dataset, it has a better performance of 0.71 than using SVM classifier without weights with 0.6768 in terms of AUC and better than 0.704 that uses Adaboost and a result of 0.95 closed to 0.96 that uses Adaboost when classifying Ecoli4 dataset and good results for the other datasets in terms of sensitivity, specificity and accuracy.

DEDICATION

To my mother and father .....

ACKNOWLEDGMENT

I greatly appreciate the thesis's supervisor for his support throughout the entire process.

I would like to thank the thesis's examiners Dr. Hashem Tamimi and Dr. Mohammed Awad for their valuable information.

Also, I would like to thank my family and my husband for helping me to complete this thesis.

Contents

List of Figures

List of Tables

List of Abbreviations

Acc	Accuracy
Acc+	Sensitivity
Acc-	Specificity
ACO	Ant Colony Optimization
AI	Artificial Intelligence
AUC	Area Under Curve
EAs	Evolutionary Algorithms
EWSVM	Enhanced Weighted Support Vector Machine
FN	False Negative
FP	False Positive
IR	Imbalance Ratio
ML	Machine Learning
OSS	One Side Selection
PSO	Particle Swarm Optimization
RF	Random Forest
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TPR	True Positive Rate
WSVM	Weighted Support Vector Machine

Chapter 1

Introduction

Classifier's training data is called imbalanced when the number of examples in at least one class (the positive class) is very small with respect to the examples in other classes [34]. This case causes a degradation in the performance of the classifier as classifiers concern in improving the overall accuracy of the classification [34]. When the training data is imbalanced, the classifier is biased to the majority class so we need to improve the classifier performance for the minority class by minimizing the number of miss-classified examples. In the real world, many datasets are highly skewed, in which most of the training data samples belong to a larger class and fewer of the training data samples belong to a smaller class, for example, telephone fraud detection and credit card fraud detection [23].

In recent years, different methods have been proposed to handle the class imbalance problem at two levels. These are the data level and the algorithm level [15].
Data level methods include data re-sampling methods for balancing the training data, while algorithm level includes methods that modify the used classifier such as cost sensitive learning. Re-sampling methods for handling imbalanced data are under-sampling and over-sampling. In the random under-sampling method, the majority class examples are randomly selected with the size equal to the size of

minority class examples to form a balanced training set and this may cause loss of useful information from the majority class examples.

On the other hand, random over-sampling repeats the examples of the minority class in order to balance the training set and this may lead to over-fitting and it is time consuming [15]. Synthetic minority over-sampling technique () is an over-sampling technique that modifies the examples of the minority class rather than using them as they are [15].

Ensemble-based methods also have been used to handle class imbalance problem as they are better in performance from using the previously mentioned ones [18]. These methods first balance the training dataset using the data re-sampling methods [5, 11, 23], or by including an algorithm modification such as cost sensitive learning [11] [36]in the learning process to minimize the number of misclassified examples in the minority class bu56t this method may cause over-fitting.

As stated previously, cost-sensitive learning was used to improve the performance when classifying imbalanced datasets. This approach considers higher costs for the miss-classification of samples of the positive class with respect to the negative class. For this purpose, weighted support vector machines are powerful classifiers and can be used to assign each training sample a different penalty of miss-classification so that the classification accuracy for the class with smaller training samples is improved [20]. Each training sample should have a different weighting factor in imbalanced data according to its importance for classification, and this can be achieved in imbalanced datasets by sitting different weighting factors for training samples of the different class which is the positive class.

Several methods have been used for selecting the weights for training samples such as calculating fixed weights and optimizing weights using Particle Swarm Optimization Algorithm (PSO) [19, 16].

In this research, we used PSO to optimize the weights of the training data samples with a support vector machine (SVM) classifier.

1.1 Thesis objective

- The basic objective of this thesis is to enhance the classification performance of imbalanced datasets with different imbalance ratios using weighted support vector machine algorithm by optimizing the weights of the training samples using the PSO optimization algorithm.
- We have another objective which is to apply other classification methods on the same datasets and to compare their classification performance with our proposed method.

1.2 Contributions

In this thesis, we proposed to enhance weighted SVM by optimizing the used weights using PSO and we made the following contributions:

- Build a model that enhances the classification performance of imbalanced datasets.
- Optimize the weights of the weighted SVM algorithm using PSO.
- Conduct comparisons with previous work to verify the effectiveness of our approach.
- Apply other classification algorithms to compare their performance with our proposed model using well known datasets benchmarks.

1.3 Thesis organization

The remaining parts of the thesis are organized as follow; Chapter 2 describes background and theories on basic concepts that are needed to understand the rest of the thesis and contains a summary of some previous works related to our work. Chapter 3 covers the methodology used to enhance the classification performance when classifying imbalanced datasets. Chapter 4 demonstrates experiments and the results achieved by the work, and results discussion. Finally, Chapter 5 concludes the work and propose some new direction for the future work.

Chapter 2

Background and Literature Review

This chapter presents a theoretical background and concepts that are related to this thesis. In addition it summaries related works in handling imbalanced datasets.

2.1 Background

This section gives a brief introduction on machine learning concept. Then, it describes data classification and classifiers implemented for this thesis. After that, it explains some aspects of bio inspired optimization algorithms, the PSO optimization algorithm and radial basis function neural network (RBF-NN).

2.1.1 Machine learning

Machine learning (ML) "is a core sub-area of artificial intelligence (AI) ; it enables computers to get into a mode of self-learning without being explicitly programmed"[2]. There are four types of machine learning [2]:

- Supervised learning: in this type, training data samples include the labels (outputs) and this type is used by most machine learning algorithms.

  Supervised learning consists of three types, which are:

  - Classification: in this type, the function being learned is discrete.
  - Regression: in this type, the function being learned is continuous.
  - Probability estimation: in this type, the output of the function is a probability.

- Unsupervised learning: in this type, training data samples does not include the desired labels(outputs), for example, clustering method.
- Semi-supervised learning: in this type, training data samples include a small number of labeled data with a large number of unlabeled data.
- Reinforcement learning : this type rewards from a sequence of actions, for example, Q-learning algorithm learns a policy that tells an agent what actions to take under what circumstances.

In general, the process of implementing ML algorithms can be summarized as follows [2]:

1. Start loop
   (a) Understand the domain and goals.
   (b) Data selection, cleaning and pre-processing:this aims to have high quality data
   (c) Apply learning models
   (d) Interpreting results: this depends on what domain experts want the results to be.
   (e) Deploying discovered knowledge: the resulted project used in practice.
2. End loop

2.1.2 Data classification

Classification is assigning a given input, called a case, to one of predefined classes. A case is described by a collection of features [32]. Classification has two phases [25] ; training phase and testing phase. In the training phase, the model learns to classify new test examples by choosing parameters of the classifier to get the highest results of the classifier. In the testing phase, the model classifies the test examples. Sections 2.1.3, 2.1.4 and 2.1.5 explain the applied classifiers in this thesis which are Random Forest (RF), Support Vector Machine (SVM) and weighted Support Vector Machine(WSVM).

2.1.3 Random Forest classifier

Random Forest (RF) is an ensemble classifier that trains a number of decision trees M in order to classify an instance z . The target class is the class that has the majority votes from all the trees. Algorithm 1 clarifies the steps of implementing RF [6].

Algorithm 1 Random Forest

1. For $m = 1$ to M
   (a) Select N instances at random but with replacement
   (b) For each N, construct T_m as follows:
      i. Select w variables at random from N
      ii. Choose the best variable to split the node
      iii. Repeat 1 and 2 to maximum possible size without pruning
2. Print the majority vote of the instance z from the votes of all the trees ${T_m(z)}^M_{m=1}$

where T_m is the $m-th$ tree, w is the number of variables that will be selected from the N instances to construct the tree T_m. RF can run efficiently on large datasets. It gives estimates of what variables are important in the classification and it has a method for estimating missing data and maintains accuracy when data is missing [11].

2.1.4 SVM classifier

SVM [39] realizes a hyper plane surface in the input space. The separation function can be represented as a linear combination of kernals associated with the support vector as shown in equation 2.1:

$$f(x) = \sum_{x_j \in S} \alpha_j y_j K(x_j, x) + b \tag{2.1}$$

Where $\alpha \geq 0$, $y_j = 1$, x_j is an example with $y_j = 1$, $K(x_j, x)$ is the kernal function that maps the input example to the high dimensional feature space and b is the offset.
If we have l data vectors $(x_i, y_i), i = 1, 2, \ldots, l$, $y_i \in \{-1.1\}$,$\mathrm{x}_i \in \mathrm{R}^2$ where x_i is the $i-th$ vector that belongs to a binary class y_i. A class label of any input x can be identified using the following decision function $Y(x) = sgn[w.\phi(x) + b]$.
Where $\phi(x)$ is a nonlinear mapping of the input data x into the high dimensional feature space and sgn is the sign of $Y(x)$. The optimal separating hyper–plane for svm is $w.\phi(x) + b = 0$ is obtained by solving the following optimization problem:

$$min_{w,b,\xi} \frac{1}{2} \|w\| + C \sum_{i=1}^{l} \xi_i \tag{2.2}$$

where $y_i(w.\phi(x_i) + b) \geq 1 - \xi_i$, $\xi_i \geq 0, i = 1, 2, \ldots, l$,$\xi_i (i = 1, 2, \ldots, l)$ are the slack variables and C is a penalty parameter chosen by the user, a larger C means assigning a higher penalty to mis-classification [20].

2.1.5 WSVM classifer

In the classification using standard SVM, parameter C is empirically selected and may not consider the importance of the training samples that are important to be classified to the correct class and some samples that may tolerate misclassification. So, training samples should have different weighting factor for classification. For this reason, weighted SVM is proposed in which it gives a different C for each training sample such that the optimization problem will be as follows:

$$min_{w,b,\xi} \frac{1}{2} \|w\| + C \sum_{i=1}^{l} S_i \xi_i \tag{2.3}$$

where $y_i(w.\phi(x_i) + b) \geq 1 - \xi_i$, $\xi_i \geq 0$, $i = 1, 2, \ldots, l$, and S_i is a weighting factor for the $i-th$ training sample.

2.1.6 Bio Inspired Optimization Algorithms

Bio inspired stochastic algorithms are called meta-heuristic methods and has ability to solve complex problems with computational efficiency compared to deterministic techniques [10]. The main purpose of these algorithms is to find the best solution for a problem. Bio inspired optimization algorithms are categorized as evolutionary algorithms (EAs) and swarm-based algorithms. Swarm-based algorithms are inspired by the behavior of animals and evolutionary algorithms are mimicking the natural evolution base [10].

Evolutionary Algorithms

Evolutionary Algorithms (EAs) are stochastic search and optimization heuristics derived from the evolution theory [42]. These algorithms can be used to solve hard problems and they only need little problem specific knowledge and can be applied to wide range of problems. EA methods only need the target (fitness) function for a given problem which is to be optimized [42]. The main idea for EA is that when individuals of a population are reproduced and met a certain selection

criteria (fitness value) and the other individuals of the population die, the new population will be the individuals that best meet the selection criteria [42]. The general scheme for EA is stated in Algorithm 2 [42].

Algorithm 2 General EA Algorithm

1. Initiate a random population (pop)
2. Evaluate all individuals from pop
3. Choose the best individuals from pop to generate the next generation
4. Create the next generation

Swarm-Based Algorithms

These algorithms were designed according to swarm intelligence that has a collective behavior of decentralized, self-organized systems interacting with each other and with their environments.
Swarm intelligence deals with systems consist of many individuals that coordinate using self- organization. The most popular examples of swarm intelligence are Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) [10].

Algorithm 3 PSO Algorithm

1. Initialize random position (P_i) and velocity (V_i) for each particle
2. For $it = 1$ to max iteration
   (a) Calculate the fitness function for each P_i
   (b) **if** fitness (P_i) $>$ pbest **then**
   Update pbest(u_p)
   (c) **end if**
   (d) Set best of all u_p of particles as gbest(u_g)
   (e) Update P_i and V_i using equations (2.4) and (2.5)
3. Give the optimal solution $gbest(u_g)$

2.1.7 Particle swarm optimization algorithm (PSO)

PSO is an optimization technique designed to mimic the flocking behavior of birds searching for food randomly in a particular area and they don't know the location of food but they find out how far it is available [10] [9]. Each solution in PSO is known as a particle. Every particle has a fitness value, a velocity (V_i) and a position (P_i). PSO algorithm starts with a group of random particles then finds an optimal solution at each iteration. The first solution is P_i and the second-best value is known as personal best u_p. The global best value u_g is the best optimal solution obtained by all particles so far in the swarm. The velocity and the position are updated at each iteration in the PSO algorithm until the last criteria of optimization or maximum iteration (max iteration) according to equation 2.4 and equation 2.5 respectively. The steps of implementing PSO are shown in Algorithm 3 [10].

$$V_i(t+1) = wV_i(t) + c_1.r_1(t)(P_b(t) - P_i(t)) + c_2.r_2(t)(P_g(t) - P_i(t)) \qquad (2.4)$$

$$P_i(t+1) = P_i(t) + V_i(t) \tag{2.5}$$

where $i = 1,2,...m$ is the number of particles, $t+1$ is the current iteration number, t is the previous iteration number, w is the inertia weight, c_1 and c_2 are the acceleration constant factors usually between (0,2), P_i is the best previous position of particle i known as personal best position (*pbest*), P_g is the position of the best particle among all the particles in the swarm known as the global best position (*gbest*), $r1$ and $r2$ are random numbers distributed uniformly in [0,1].

2.1.8 Radial Basis Function Neural Network

Radial Basis Function Neural Network (RBF-NN) is a type of neural network that is mainly applied to supervised learning problems such as classification, regression and time series prediction [29]. The RBF-NN has three feed-forward layers, which are the input layer, the hidden layer and the output layer.

As shown in Figure 1, each of n examples feed forward to N basis functions. Then their outputs are combined with M weights to form the network output y_m. The examples that feed the hidden layer have to be from the same class at one time. So, y will be the resulted function from this particular class [29] [13]. The activation function in the hidden layer (ϕ) will be the Gaussian function as shown in equation 1 [17].

$$\phi = exp\left[-\frac{\|x - c_i\|^2}{\sigma_i^2}\right] \tag{2.6}$$

Where $x = (x_1, x_2, \ldots x_n)$ is a vector of input data, $i = 1,2,...,N, N$ is the number of nodes in the hidden layer, c_i is the center of $i-th$ hidden neuron, and σ_i is the width of $i-th$ hidden neuron.

The output y_m is calculated as follow [17] :

$$y_m = \sum_{m=1}^{M} w_{im}\phi_i(x) \tag{2.7}$$

Where $m = 1,2,3...,M$ and M is the number of nodes in the output layer. W_{im} is the connection weight value between the $i-th$ hidden layer node and the $m-th$ output node.

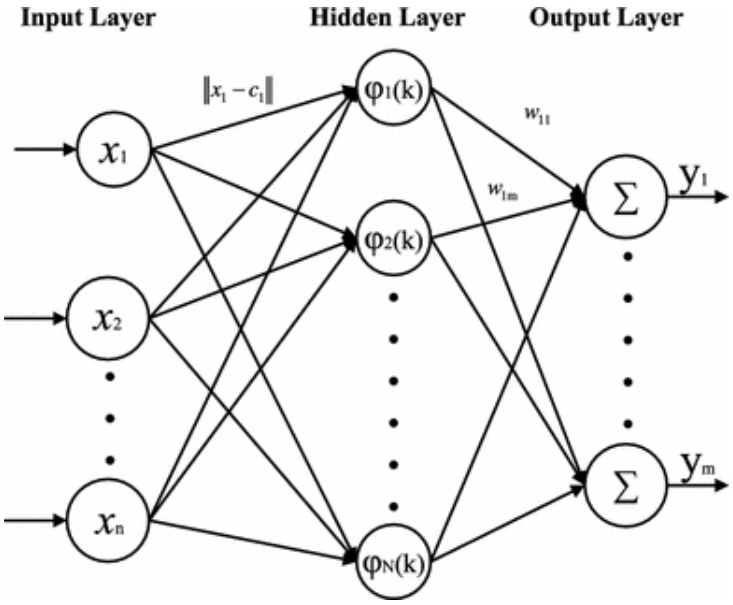

Figure 2.1: RBF Neural Network Representation [17]

Classification decision is taken by assigning the input to the class with the highest score. The score for a specific class is obtained from each output node according to equation 2.

2.2 Literature review

This section shows related works in handling imbalanced datasets using ensemble methods, support vector machine [24], [35] and weighted SVM classifiers at different levels; data level and algorithm level or as ensemble of classifiers to en-

hance the performance of classifying the imbalanced datasets. The next sections will introduce these works.

In [3], the authors used AdaBoost algorithm to classify the imbalanced dataset. First, they proposed to under-sample the dataset then at each iteration they proposed to modify AdaBoost weight update equation to include cost term. This cost term is updated at each iteration at AdaBoost such that it will be increased for mis-classified instances and decreased for correctly classified instances

In [14], they proposed an ensemble algorithm based on automatic clustering and under-sampling. This approach modifies the subset of samples of the AdaBoost classifier. The samples in each subset have a different weight according to its importance. First, the algorithm clusters the majority class samples after a sign weight to them. Then it takes the top-weighted samples in each cluster and combines them with minority class samples to consider them as training samples.

2.2.1 Literature proposed approaches with SVM

The authors in [38] combined resampling and ensemble techniques. They have balanced the dataset by over-sampling the support vector instances of minority class using Synthetic Minority Oversampling Technique (SMOTE) and one class support vector model to estimate the support vector instances. After that, they decompose the majority class samples into clusters using their own clustering method. The next step is to build an ensemble of SVMs to train each cluster with the oversampled minority class samples. Their results were good compared with other ensemble-based methods such as bagging and adaboost.

In 2011, Liu et .al [23] has also proposed to handle imbalanced data using ensemble classifier integrated with SMOTE data level re-sampling algorithm. This algorithm filters out the impure instances that may exist in the dataset, then uses SMOTE to over sample the minority class samples. After that, the bootstrap sampling used to under-sample the majority class so that each bootstrap sample has the same or similar size as the over-sampled positive instances. Then, it combines

each bootstrap sample (of the majority class) with the over-sampled positive instances to form a training set to train an SVM. The class prediction is determined by majority voting of the ensemble of classifiers.

The authors in [33] proposed to use combined sampling methods at the data level to first balance the datasets then applying SVM for classification. They used SMOTE oversampling method to increase minority class samples and used Tomek links method to under-sample the majority class samples and eliminate the noisy samples. The results were compared with experiments using SMOTE and Tomek links separately then classifying using SVM.

Their results were better than the other applied experiments using accuracy, AUC, F measure metrics. In this work, they didn't compare their results with other studies and prove the effectiveness of their approach. The authors in [22] proposed to first cluster the original imbalanced dataset using k-means clustering algorithm then to over-sample the minority class examples in each cluster using SMOTE then training the resulting clusters using ensemble SVM.

In this method, the authors did not state how to determine the number of clusters for the clustering method. The author in [8] proposed an over-sampling method enhances SMOTE method by sensitizing new samples closer to the actual distribution of the datasets to over-sample the minority class samples. They also used SVM with different error coasts for the positive and negative classes. In [9], this method uses SMOTE to generate artificial instances to over-sample the minority class instances. It also uses PSO algorithm to optimize the hyper-plane of the SVM classifier by choosing the artificial instances that were generated by SMOTE to consider them as the new support vectors for SVM hyper-plane as they improve SVM classification performance. A problem with this method is that it has computational cost due to the use of the over-sampling method. The authors in [40] used under-sampling to delete samples from majority class samples by first dividing the imbalanced dataset to m subsets then use one of the subsets to train SVM classifier and generate the hyper-plan and delete samples that far from the hyper-plan. After that, they use the SMOTE algorithm to over-sample the positive

samples that exist in the subset. The next step is to train new SVM classifier and to repeat the same mentioned process for the rest of the subsets.

The main drawback of this method is that the final resulted classifier is trained using a subset of the data and this may lead to loss of important information from other subsets of samples.

Another approach [26] proposes to over-sample minority class samples by generating synthetic instances for each instance in the minority class. they modified the SMOTE algorithm when using the original instance and one of the randomly selected neighbour instance to generate the synthetic instance. After that, they propose to modify the kernel matrix of SVM to include the generated synthetic instances. In [7] they used hybrid re-sampling for balancing the dataset. They used SMOTE over-sampling algorithm to increase the number of samples in the minority class and use one side selection (OSS) under-sampling method to reduce the number of samples in the majority class then merge the resulted datasets to train the SVM classifier.

. In [37], they modified the weight of AdaBoost classifier using cost parameter when using SVM as base learner.

2.2.2 Literature proposed approaches with WSVM

In [41] they proposed to assign different weight values of the support vector machine for each instance in the dataset. This weight value is updated using a boosting algorithm. The authors in [21] proposed a weight factor to be included in SVM with quadratic cost function (lagrangian SVM). This weight is calculated for each instance in the dataset using equation 3.1 in section 3.2.1.

In [16], they used weighted SVM to classify the imbalanced datasets using soft margin SVM (C-SVM). They choose to use fixed weight value for each sample in each class. For majority class samples, the weight is 1 and for minority class samples the weight is m_+/m_-. where m_+ is the majority class samples and m_- is the minority class samples. The authors in [12] proposed to assign weights for each feature of the samples in the dataset and this weight depends on the average

value of the feature values in the positive class samples and the average value of this feature values in the negative class samples.

According to the literature, few studies have used PSO for optimizing weights of weighted SVM method. Some algorithms used re-sampling methods with other classifiers such as SVM, other algorithms used ensemble methods with SVM, and there are algorithms used weighted SVM with fixed weights for samples. So, in our thesis we propose to use PSO optimization algorithm in order to optimize weights of the weighted SVM algorithm.

Chapter 3

Data and Applied Models

This chapter discusses the data benchmarks and the applied models that were used in order to classify the imbalanced datasets. The first section 3.1 introduces the selected benchmarks datasets. Then section 3.2 discusses the three classifiers that were applied in order to compare their performance with our enhanced classifier and will also clarify our proposed classifier that uses PSO for optimal weights for weighted SVM. Section 3.3 describes the PSO representation for our approach. Finally, section 3.4 will show the performance evaluation technique that used to evaluate our proposed approach.

Four classifiers are applied so that we can compare three of them with our proposed model that enhances the classification accuracy using PSO for optimal weights of weighted support vector machine (WSVM). Figure 3.1 shows the steps were used to implement our proposed model. Section 3.2 will clarify our proposed algorithm for the classification of the selected imbalanced datasets.

3.1 Data selection

Five benchmark datasets were selected from the public KEEL Imbalanced Data repository in order to test our proposed approach [4]. These datasets are tow-class

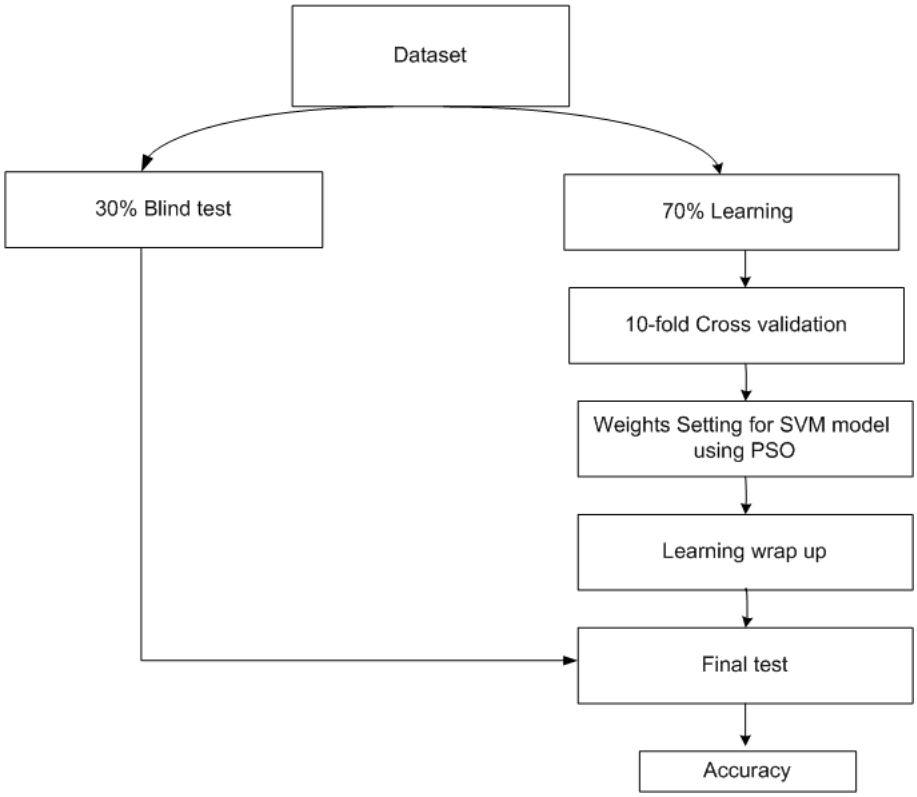

Figure 3.1: Our proposed algorithm for the classification of imbalanced data

problem related datasets to use with 10-Fold-cross validation. The selection of the datasets was according to their imbalance ratios. Table 3.1 shows the names of the five datasets sorted by the value of the imbalance ratio with the number of features, training data samples, testing data samples, hidden data samples and all samples. Imbalance ratio (IR) is often used to indicate that at least one class has a very small number of examples with respect to the examples in other classes [34]. This ratio can be calculated when dividing the number of examples in the majority class (the negative class) by the number of examples in the minority class (the positive class). All the datasets were split into training dataset with 70% of the data samples, with the same imbalance ratio as the original dataset, and testing dataset with 30% of the data samples. The steps of splitting the datasets

into imbalance training dataset and hidden dataset are as follow:

1. Randomly select 70%, as training dataset, from the original dataset.
2. Open the resulted training dataset manually and ensure the imbalance ratio.
3. If the imbalance ratio is not correct then manually modify the ratio by replacing samples from positive and negative classes to get the required imbalance ratio.
4. The remaining 30% of the data samples will be the hidden dataset.

Table 3.1: Summary for the imbalanced datasets used in the experiments

Imbalanced dataset	Imbalance ratio	Features	Training samples	Hidden data samples	All Samples
Pima	1: 1.86	8	535	230	768
Ecoli4	1: 15.8	7	233	100	336
Glass5	1: 22	9	147	64	214
Poker-$9_{vs}7$	1: 29.5	10	168	73	244
Yeast6	1: 41.4	8	1035	445	1484

In this research, we will test our proposed approach with datasets with different imbalance ratios. Starting from datasets low IR (PIMA data set (IR= 1: 1.86), Ecoli4 (IR = 1: 15.8)) then datasets with moderate IR (glass5 (IR=1: 22.78), Poker-9_vs_7 (IR=1:29.5)) and finally a dataset with severe IR (yeast6 (IR= 1:40)). Section 3.1.1 will show a brief description of the selected five imbalanced datasets.

The selected imbalanced datasets can be categorized as follow:

1. Low imbalance ratio datasets
   This category includes two datasets; Pima dataset with 1: 1.86 IR and Ecoli4 dataset with 1: 15.8 IR.
   - Pima dataset
     Pima dataset (Pima Indian's Diabetes Dataset) has diagnostic measures as attributes which can be used to predict the onset of diabetes.

This dataset has two classes; A positive class and a negative class. The total number of instances is 768 in which 268 (34.9%) belong to the positive class and the remaining 500 (65.1%) belong to the negative class. Pima IR can be calculated when dividing the number of instances belongs to the negative class by the number of instances belongs to the positive class. 500/268 = 1.86 which can be written as 1: 1.86.

- Ecoli4 dataset

  Ecoli4 dataset contains information of Escherichia coli. It is a bacterium of the genus Escherichia that is commonly found in the lower intestine of worm blooded organisms. This dataset has two classes; A positive class and a negative class.

  The total number of instances is 336 in which 20 (6%) belong to the positive class and the remaining 316 (94%) belong to the negative class.

  Ecoli4 IR can be calculated when dividing the number of instances belongs to the negative class by the number of instances belongs to the positive class. 316/20 = 15.8 which can be written as 1: 15.8.

2. Moderate imbalance ratio datasets

   This category includes two datasets; Glass5 dataset with 1: 22.78 IR and Poker-9_vs_7 dataset with 1:29.5 IR.

   - Glass5 dataset

     Glass5 dataset identifies 7 types of glasses using 9 features. This dataset has two classes; A positive class and a negative class. The positive class represents class 5 of glass and the remaining 6 classes represents the negative class.

     The total number of instances is 214 in which 9 (4.2%) belong to the positive class and the remaining 205 (95.8%) belong to the negative

class.

Glass5 IR can be calculated when dividing the number of instances belongs to the negative class by the number of instances belongs to the positive class $205/9 = 22.78$ which can be written as $1 : 22.78$.

- Poker-9_vs_7 dataset
  Poker-9_vs_7 dataset is a description of a hand consisting of 5 playing cards. Each card has two attributes; Rank and suit, So that it has 10 features. This dataset has two classes; A positive class and a negative class. The positive class represents class 9 poker hand and the negative class represents class 7 poker hand.
  The total number of instances is 244 in which $8(0.03\%)$ belong to the positive class and the remaining $236(0.97\%)$ belong to the negative class.

  Poker-9_vs_7 IR can be calculated when dividing the number of instances belongs to the negative class by the number of instances belongs to the positive class $236/8 = 29.5$ which can be written as $1 : 29.5$.

3. Severe imbalanced ratio dataset
   This category includes Yeast6 dataset with 1:41.4 IR.

- Yeast6
  Yeast6 dataset contains information on yeast in order to predict the localization sites of proteins. This dataset has two classes; A positive class and a negative class. The positive class represents class 6 which is extracellular (Exc) and the remaining classes are represented by the negative class. The total number of instances is 1484 in which $35(0.02\%)$ belong to the positive class and 1449 (0.98%) belong to the negative class.

Yeast6 IR can be calculated when dividing the number of instances belongs to the negative class by the number of instances belongs to the positive class $1449/35 = 41.4$ which can be written as $1 : 41.4$.

All of the datasets were divided into training data and testing data used with 10-fold cross-validation with 70% of the original dataset and the remaining 30% of the dataset was used as hidden data for the blind test of the model.

These datasets will also be tested using another three classifiers which are Random Forest classifier, SVM classifier and weighted support vector machine (WSVM) classifier in order to compare their results with our proposed enhanced weighted SVM results.

The next section 3.2 will clarify the steps of implementing the three classifiers; Random forest SVM and weighted SVM.

3.2 Learning algorithms description

We have applied three classifiers which are Random Forest (RF), Support Vector Machine (SVM) and weighted Support Vector Machine (WSVM) in order to compare their performance with our proposed model. This comparison will show that our proposed enhanced weighted SVM (EWSVM) using PSO has a better classification performance than the other three classifiers in classifying datasets with different levels of imbalanced ratios. In this section we will present the algorithm of each of them.

3.2.1 Random forest, SVM and weighted SVM algorithms description

This section discusses the steps that we have used to implement the three classifiers in order to compare their performance with our proposed model when classifying imbalanced datasets with different imbalance ratios.

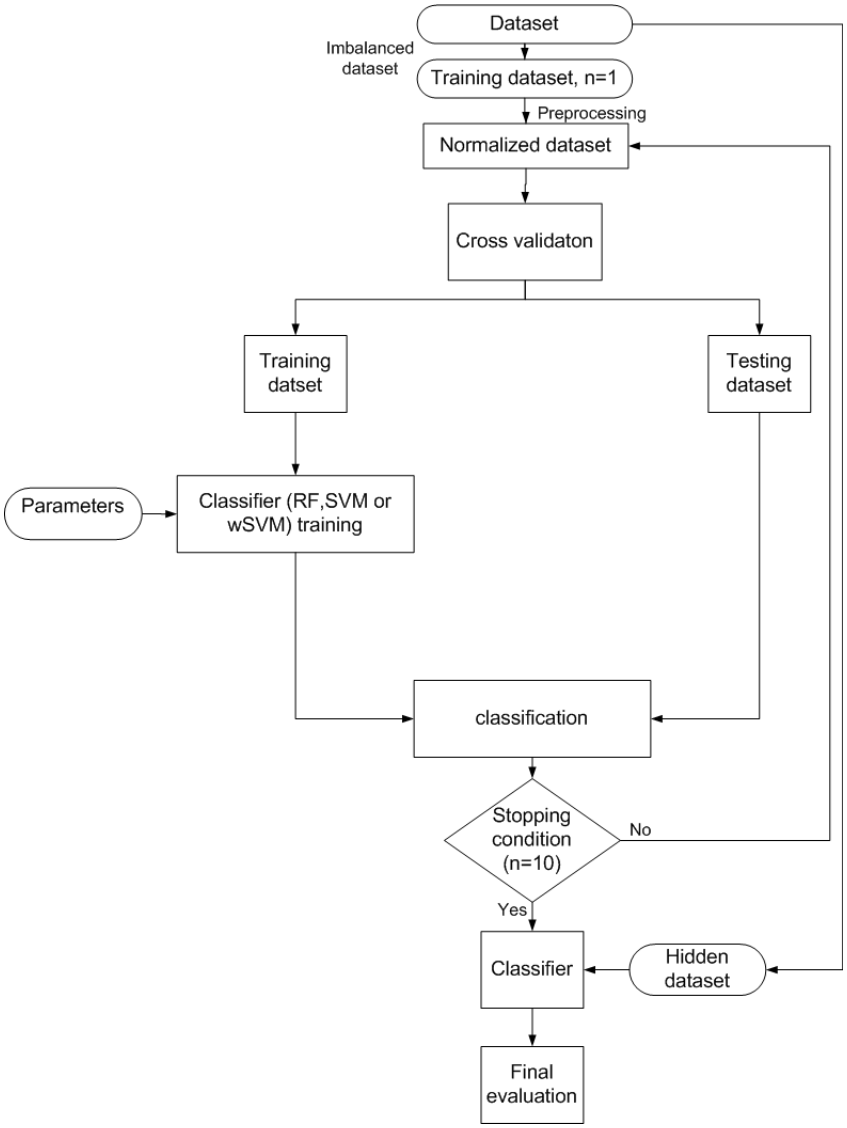

Figure 3.2: The steps of implementing the three classifiers (RF, SVM and weighted SVM)

As shown in figure 3.2, the imbalanced dataset has pre-processed using normalization then split into training dataset and hidden dataset. The training dataset was used with cross validation and split into two parts; training and testing dataset. The training dataset was used to train the classifier and the testing dataset was used

to evaluate the performance of the classifier at the training process. The hidden dataset was used in the final evaluation of the classifier.

Random forest classifier has a parameter which is the number of trees used in the classifier.

Weighted SVM classifier has a parameter which is weight. Weights are used for every sample in the training dataset and these weights are fixed and calculated using this formula [21]:

$$w_i = \begin{cases} 1 & if \quad y_i = 1 \quad and \quad N_{pos} N_{neg}, \\ \frac{N_{neg}}{N_{pos}} & if \quad y_i = 1 \quad and \quad N_{pos} < N_{neg}, \\ \frac{N_{pos}}{N_{neg}} & if \quad y_i = -1 \quad and \quad N_{pos} N_{neg}, \\ -1 & if \quad y_i = 1 \quad and \quad N_{pos} < N_{neg} \end{cases} \tag{3.1}$$

Where, N_{pos} is the number of the positive class samples and N_{neg} is the number of the negative class samples.

3.2.2 Enhanced weighted SVM model description

This section clarifies our enhancement on the weighted SVM to improve the performance when classifying imbalanced datasets with different imbalance ratios. Our proposed approach (EWSVM) uses PSO algorithm with SVM classifier. PSO is used to optimize the weight for each individual data sample in the training dataset. We used PSO because it is an optimization technique and has many advantages such as; it is simple and easy to implement and it has a high convergence rate to get the best solution [10]. PSO also has few parameters to adjust and it provides optimal solution to the applications with limited computational resources such as memory [10]. As shown in figure 3.3, the dataset is pre-processed using normalization then split into training dataset and hidden dataset. The steps for implementing enhanced weighted SVM are as follow:

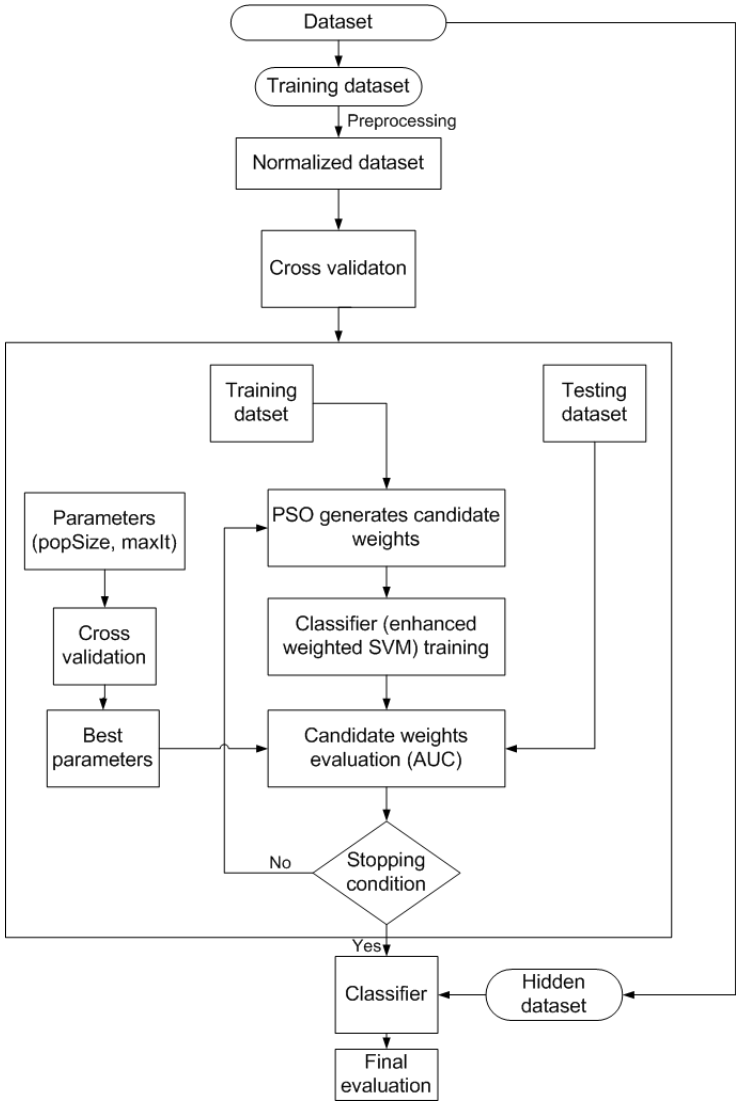

Figure 3.3: The steps of implementing our proposed EWSVM model

1. Pre-processing the dataset using normalization in order to change the values if the features to a common scale.

2. Splitting the dataset into two parts; training dataset and hidden dataset.

3. Selecting the best parameters (popSize, maxIt) for PSO.

4. Applying cross-validation on the training dataset so that training dataset and testing dataset will be generated.

5. Applying PSO in order to generate candidate weights that will be used in training the weighted SVM classifier.

6. Evaluating the generated candidate weights using the testing dataset.

   Steps for 4,5 and 6 will be repeated until reaching the stopping condition. There are two types of stopping conditions to select one of them [27]:

   - Stopping condition based on generation procedure; in this type the number of iterations is predefined.
   - Stopping condition based on an evaluation function; in this type, the operation will be repeated until the optimal solution is reached according to some evaluation function.

7. Validating the final selected weights if they are valid or not by using the hidden dataset.

3.3 PSO Representation used in our approach

Particle swarm is denoted by $P = [p_1, p_2, \ldots . p_m]$. Each particle is a vector with $(1 * (N_{neg} + N_{pos}))$ dimensionality.

Where N_{neg} is the number of negative samples in the training dataset and N_{pos} is the number of positive samples in the training dataset and m is the size of the initial population.

The problem is to determine the optimal samples weights that improve performance. $(m * (N_{neg} + N_{pos}))$ is the dimensional search space.

The search space of each individual $W = [w_1, w_2, \ldots . w_{N_{pos}+N_{neg}}]$ is

$$W_{min} = 1$$

$$W_{max} = N_{neg}/N_{pos} \tag{3.2}$$

Each particle $p(i)$ has $(1*(N_{neg}+N_{pos}))$ dimensional velocity $V = [v_1, v_2, \ldots, v_{(N_{neg}+N_{pos})}]$, and this velocity is computed according to equation 3.3.

$$V_i(t+1) = wV_i(t) + c_1.r_1(t)(P_b(t) - w_i(t)) + c_2.r_2(t)(P_g(t) - w_i(t)) \tag{3.3}$$

For each particle $P(i)$, position is value of weight and this value is updated according to the following equation.

$$w_i(t+1) = w_i(t) + V_i(t) \tag{3.4}$$

The swarm of the candidate particles $\{P_i^L\}_{i=1}^m$ is moved in the search space to find a solution, where m is the population size, $l \in \{0, 1, \ldots, L\}$ denotes the $l-th$ movement of the swarm.

3.4 Performance evalution

In our proposed approach we used the area under the receiving operating characteristic curve (AUC) as a fitness function for PSO. We also used AUC , sensitivity, specificity and classification accuracy for validation and performance evaluation. AUC is calculated using 3.5 equation [31].

$$AUC = \frac{TPR + TNR}{2} \tag{3.5}$$

where TPR is the true positive rate (sensitivity) and is the true negative rate (specificity).

Sensitivity can be calculated using the following formula:

$$Sensitivity(Acc+) = \frac{TP}{TP + FN} \tag{3.6}$$

Specificity can be calculated using the following formula:

$$Specificity(Acc-) = \frac{TN}{TN+FP} \tag{3.7}$$

Accuracy can be calculated as shown in the following formula:

$$Accuracy(Acc) = \frac{TP+TN}{TP+TN+FP+FN} \tag{3.8}$$

Where TP (True Positive) is the number of positive samples that are correctly classified, FP (False Positive) is the number of negative samples that are incorrectly classified as positive, TN (True Negative) is the number of negative samples that are correctly classified as negative, FN (False Negative) is the number of positive samples that are incorrectly classified as negative.

Chapter 4

Experiments and Results

This chapter will show the experimental results of our proposed enhanced weighted SVM approach. First section 4.1 describes the computing environment and experimental settings that were used to conduct this work. Then, section 4.2 describes the parameters selection of PSO. Section 4.3 presents the pre-processing phase used. Section 4.4 presents the classification results of the datasets using Random Forest, SVM and weighted SVM classifiers. Finally, section 4.5 shows the classification results of our proposed approach and compares it with traditional three applied classifiers and with the results obtained by other researches using the selected benchmarks.

First of all, we have used Radial Basis Function () nueral network with genetic algorithm to enhance the performance of classifying imbalanced datasets. RBF centers were selected using k-means on each class [28]. RBF radii were computed according to the following equation

$$\delta^2 = \frac{\sum |x_i - c_k|^2}{n} \tag{4.1}$$

where n is the number of the class instances, $i : [1,2,3,...,n]$, and c_k is the $k-th$ center of the class, where $k : [1,2,3,...,$ number of the class centers]. The

weights were optimized using genetic algorithm. The results were not enhanced when increasing the number of centers and the best AUC when classifying Pima dataset was 72%.

Another experiment we have applied using RBF neural network with PSO. RBF parameters were computed using the same methods as that used with genetic, but weights were optimized using PSO. The best AUC we got when classifying Pima dataset was 73% and the results were not enhanced when increasing the number of centers.

According to these results, we conclude that RBF is not good for imbalanced datasets.

4.1 Computing environment

All the machine learning algorithms, that were used in our experiments, were implemented using MATLAB® 2013 under windows 10 with Core i7-7500U CPU 2.70 GHz, 8GB RAM memory. MATLAB is a high-performance language for technical computing. It integrates visualization, programming and computation. MATLAB is a matrix-based language and has built-in graphics to visualize data [1]. In addition, it contains a library of pre-built toolboxes that can be used to implement algorithms in different domains. In our experiments, we have used libSVM- weights library in order to implement weighted SVM classifier. libSVm-weights: libSVM is a simple software for SVM classification and regression. It can solve C-SVM classification, nu-SVM classification, one class SVM and others. libSVM-wheigts is a tool provides a simple interface to libSVM with instance weights support.

4.2 Parameters selection for PSO

This section presents the parameters that used for PSO optimization algorithm for our proposed EWSVM approach. In order to choose the best value of the number of iterations *maxIt* and the number of particles *popSize*, we have applied many experiments for PSO.

Table 4.1 shows the selected parameters for PSO. Figure 4.1 illustrates the behaviour of the PSO fitness function (AUC) according to the number of particles when the number iterations are fixed = 40. The number of particles that gives the best fitness value (AUC) at the fixed number of iterations is 40. After choosing the number of particles, we have applied another experiments with the selected number of particles to determine the best value of iterations that gives the best fitness value and this value is 60. Figure 4.2 illustrates the behaviour of the PSO fitness function (AUC) for each number of iteration using 60 particles.

Table 4.1: Selected parameters for PSO used in EWSVM

Parameter	Value
popSize	40
maxIt(number of iterations)	60
c1	1.4
c2	1.4
wmax	$(\frac{N_{neg}}{N_{pos}})$
wmin	1
w	1
wdamp	0.99

4.3 Pre-processing

The attributes of each of the four previously mentioned datasets are numerical values and each attribute is measured using a different scale and has a different range

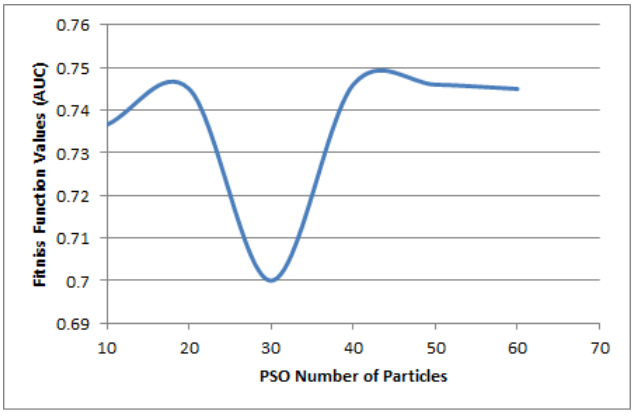

Figure 4.1: PSO fitness function (AUC) behavior according to the number of particles

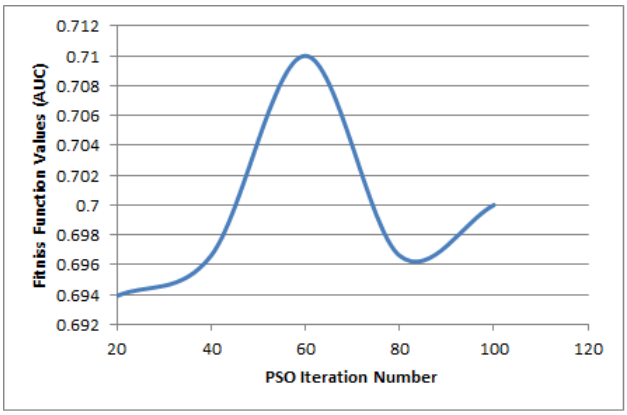

Figure 4.2: PSO fitness function (AUC) behavior for each iteration number

of possible values. Normalization is a method performed for data pre-processing and used to standardize the range of independent features (attributes of data).

According to [30], normalization improves the classification accuracy and speeds up the classification and the convergence of the model. We have used it to normalize the data into the range [0,1] using equation 4.2.

$$Z = \frac{X_i - X_{min}}{X_{max} - X_{min}} \tag{4.2}$$

Where Z is the new normalized value, X_i is the original attribute value, X_{min} is the minimum value of all the attribute values and X_{max} is the maximum value of all the attribute values.

4.4 Applied classifiers results

This section presents the results of applying Random forest, SVM and weighted SVM algorithms of training and testing phases.

Each value of the results is the average value obtained after repeating the training and testing phases three times, each time with different training and testing datasets, for each classifier.

Table 4.2 shows the results of our classification experiments for Random forest algorithm with the five imbalanced datasets; each dataset was classified using the applied Random forest algorithm with 50 trees. Training accuracy was computed for the training phase then AUC, testing accuracy, Acc+ and Acc- were computed for the testing phase using the hidden data samples of the imbalanced datasets.

Table 4.2: Classification results for Random forest algorithm

Imbalanced dataset	AUC	Training acuuracy	Testing accuracy	Acc+	Acc-
Pima	0.73	0.76	0.77	0.6156	0.8461
Ecoli4	0.80	0.98	0.97	0.6048	1
Glass5	0.71	0.98	0.97	0.4444	0.9844
Poker-9_vs_7	0.50	0.97	0.98	0	1
Yeast6	0.67	0.99	0.97	0.3459	0.9985

4.4.1 SVM algorithm results

Table 4.3 shows the results of our classification experiments for the SVM algorithm. Training accuracy was computed for the training phase then AUC, testing accuracy, Acc+ and Acc- were computed for the testing phase using the hidden data samples of the imbalanced datasets.

Table 4.3: Classification results for SVM algorithm

Imbalanced dataset	AUC	Training accuracy	Testing accuracy	Acc+	Acc-
Pima	0.71	0.77	0.74	0.6359	0.7925
Ecoli4	0.92	0.96	0.95	0.8968	0.9579
Glass5	0.92	0.96	0.95	1	0.8429
Poker-9_vs_7	0.59	0.92	0.88	0.2778	0.9098
Yeast6	0.86	0.88	0.86	0.8633	0.8551

4.4.2 Weighted SVM (WSVM) algorithm results

Table 4.4 shows the results of our classification experiments for weighted SVM algorithm; Training accuracy was computed for the training phase then AUC, testing accuracy, Acc+ and Acc- were computed for the testing phase using the hidden data samples of the imbalanced datasets.

Table 4.4: Classification results for WSVM algorithm

Imbalanced dataset	AUC	Training acuuracy	Testing accuracy	Acc+	Acc-
Pima	0.73	0.74	0.75	0.71	0.7562
Ecoli4	0.94	0.96	0.94	0.9444	0.9417
Glass5	0.82	0.95	0.96	0.6667	0.9728
Poker-9_vs_7	0.60	0.93	0.91	0.2778	0.9335
Yeast6	0.88	0.87	0.89	0.9111	0.8594

4.5 Enhanced weighted SVM (EWSVM) results

This algorithm enhances weighted SVM algorithm by selecting optimal weights using PSO optimization algorithm. The fitness function of PSO is AUC used to select the best weight value for each sample in the training dataset. Table 4.5 shows the results of our classification experiments for the EWSVM algorithm. Training accuracy was calculated using the confusion matrix of the training phase and testing accuracy, Acc+ and Acc- were calculated using the confusion matrix of the testing phase (with blind test dataset). The confusion matrix allows visualization of the performance of classification algorithms and allows detailed analysis of

the performance results. Table 4.6 shows the components of the Pima confusion matrix.

Table 4.5: Classification results for EWSVM algorithm

Imbalanced dataset	AUC	Training accuracy	Testing accuracy	Acc+	Acc-
Pima	0.71	0.77	0.77	0.5336	0.8855
Ecoli4	0.95	0.99	0.99	0.91	1
Glass5	0.75	0.99	0.97	0.50	1
Poker-9_vs_7	0.66	0.99	0.97	0.33	1
Yeast6	0.55	0.98	0.98	0.1	0.9954

Table 4.6: Confusion matrix components

	Positive class	Negative class
Positive class	TP	FN
Negative class	FP	TN

Table 4.7: Confusion matrix for Pima dataset for training phase

	Positive class	Negative class
Positive class	103	79
Negative class	39	314

Table 4.8: Confusion matrix for pima dataset for testing phase

	Positive class	Negative class
Positive class	47	38
Negative class	16	129

4.6 Discussion

In this section, we discuss the results of the proposed approach EWSVM and compare them with traditional algorithms implemented to prove the efficiency of our approach. It also compares our results with previous works when classifying imbalanced datasets with different imbalance ratios.

Tables (4.2, 4.3 and 4.4) show the results of our experiments using Random forest, SVM and WSVM algorithms. Our approach has good performance in classifying all the imbalanced datasets. For Pima dataset, all of the algorithms show good results in terms of AUC. However, our approach outperforms them in terms of training and testing accuracies and outperforms them in terms of AUC for Ecoli4 and Poker datasets. Accuracy in classifying negative samples (Acc-) was better than the other traditional algorithms and EWSVM has correctly classifies all the negative samples for Ecoli4, Glass5, Poker and Yeast6 imbalanced datasets.

In addition, the other algorithms didn't maintain a good performance in classifying negative samples, whereas our approach did for all the datasets.

According to the experiments results, all the classifiers (RF, SVM, WSVM and EWSVM) have better performance in classifying Ecoli4 dataset (1:16 IR) than Pima (1:2 IR). SVM and WSVM classifiers have better performance, in terms of AUC, when classifying Yeast6 (1:41 IR) than poker (1:29 IR) and have not affected, in terms of accuracy, in classifying positive samples when the imbalance ratio increased.

SVM and WSVM classification performance, in terms of AUC, has not affected by increasing the imbalance ratio of the datasets but EWSVM has affected except for Ecoli4 dataset.

All the classifiers have the best performance in classifying Ecoli4 dataset. EWSVM classification performance, in terms of accuracy and accuracy in classifying negative samples (Acc-), has not affected by the increasing of the imbalance ratio of the datasets.

Table 4.9 shows the results of other works in classifying Pima dataset using different performance measurements (AUC, Acc+, Acc- and accuracy).

Our approach has good results in terms of AUC (0.71) for Pima dataset compared with [7] that uses SVM classifier, and [3] that uses Adaboost classifier. Accuracy was high compared to other works [21] and [7] and similar to [12]. In addition, our proposed approach maintains the performance of classifying negative samples and performs good in classifying positive samples while the other approaches do not maintains good performance in classifying negative samples when they have good performance in classifying positive samples.

AUC for Ecoli4 dataset is 0.95 and it is good compared with [14] that has $AUC = 0.96$ using Adaboost ensemble classifier with under-sampling that assigns weights for the training samples. Acc+ (0.91) and Acc- (100%) for Ecoli4 dataset are the average values calculated from the confusion matrix of the blind test phase. These results show that the algorithm did not over-fit for the training dataset.

Table 4.9: Related works results for classifying Pima dataset

Approach	Performance measurement	The result	EWSVM result
hwang[21] - WSVM	Acc+	0.8160	0.5336
	Acc-	0.7370	0.8855
	Accuracy	0.7490	0.77
liu[23]-SVM	Acc+	0.6866	0.5336
	Acc-	0.8201	0.8855
cao[7]-SVM	AUC	0.6768	0.71
	Accuracy	0.6513	0.77
cervantes[9]-(SVM+PSO)	AUC	0.7420	0.71
cheng[12]-(weighted-features SVM)	Accuracy	0.77	0.77
ahmed[3]-adaboost	AUC	0.704	0.71

Chapter 5

Conclusion and Future Work

This chapter concludes this thesis and discuss future work perspectives.

5.1 Conclusion

In this thesis, we proposed an algorithm to enhance the performance of classifying imbalanced datasets. This algorithm depends on optimizing the weights of weighted SVM using PSO. The results show the enhancement in performance of our algorithm compared with Random Forest, SVM and WSVM implemented algorithms. The proposed model was tested using five datasets with different imbalance ratios from 1:2 IR to 1:41 IR. These datasets are Pima, Ecoli4, Glass5, Poker-9_vs_7, and Yeast6. The obtained results were compared with traditional results and with other works using different performance measurements; AUC, Sensitivity, Specificity, and Accuracy.

For Pima dataset with imbalance ratio 1:2, this data was hard to be classified and the results show that our approach gives 0.71 for AUC and good results for sensitivity 0.5336 and specificity 0.8855 and it is better in AUC compared with other works[7] and [3] and close to [9].

For Ecoli dataset, AUC is good 0.95 and it is close to the result in [14], AUC =

0.96. The results 0.91 and 1 are in terms of sensitivity and specificity respectively. Sensitivity is high and our approach also maintains specificity in classifying negative samples.

For glass5 training and testing accuracies are very good 0.99 and 0.97 and our approach has 0.50 for sensitivity for blind test and it maintains a very good result for specificity (Acc- = 1).

For Poker-9_vs_7 dataset, AUC = 0.66, Acc+ = 0.33 and Acc- = 1 are better compared to the other implemented classifiers.

For yeast6 dataset, the results were not very good compared to other implemented classifiers, AUC = 0.55 and Acc+ = 0.1. However, the performance was better in terms of training and testing accuracies; 0.98 for both tests and in terms of Acc- was 0.9954 for the blind test.

The rate of imbalance affected the classification performance of EWSVM classifier in terms of AUC and Acc+, but did not affect it in terms of accuracy and Acc-. SVM and WSVM classification performance was not affected by the rate of imbalance in terms of AUC and Acc+, but they have a bad performance in classifying Poker dataset. So, the classification performance of a classifier depends on the dataset features and the imbalance ratio.

As a future work we can use a feature selection method to select important features for the minority class samples to enhance the learner. Or we can use this algorithm with a re-sampling method to first re-sample and balance the training dataset to enhance the classification performance.

Bibliography

[1] https://ch.mathworks.com/helpmatlablearn_matlabproduct-description.html. 2062019.

[2] https://machinelearningmastery.com/basic-concepts-in-machine-learning/. 20/6/2019.

[3] Sajid Ahmed, Farshid Rayhan, Asif Mahbub, Md Rafsan Jani, Swakkhar Shatabda, and Dewan Md Farid. Liuboost: Locality informed underboosting for imbalanced data classification. In *Emerging Technologies in Data Mining and Information Security*, pages 133–144. Springer, 2019.

[4] Jesús Alcalá-Fdez, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez, and Francisco Herrera. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17, 2011.

[5] Jerzy Błaszczyński and Jerzy Stefanowski. Neighbourhood sampling in bagging for imbalanced data. *Neurocomputing*, 150:529–542, 2015.

[6] Leo Breiman and Adele Cutler. Random forests-classification description. *Department of Statistics, Berkeley*, 2007.

[7] Lu Cao and Yikui Zhai. Imbalanced data classification based on a hybrid resampling svm method. In *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and*

Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom), pages 1533–1536. IEEE, 2015.

[8] Qinghua Cao and Senzhang Wang. Applying over-sampling technique based on data density and cost-sensitive svm to imbalanced learning. In *2011 International Conference on Information Management, Innovation Management and Industrial Engineering*, volume 2, pages 543–548. IEEE, 2011.

[9] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodriguez, Asdrúbal López, José Ruiz Castilla, and Adrian Trueba. Pso-based method for svm classification on skewed data sets. *Neurocomputing*, 228:187–197, 2017.

[10] SD Chavan and Nisha P Adgokar. An overview on particle swarm optimization: Basic concepts and modified variants. *IJSR Journal*, 4(5), 2015.

[11] Chao Chen, Andy Liaw, and Leo Breiman. Using random forest to learn imbalanced data. *University of California, Berkeley*, 2004.

[12] Ding Cheng and Min Wu. A novel classifier-weighted features cost-sensitive svm. In *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 598–603. IEEE, 2016.

[13] Feng Chu and Lipo Wang. Applying rbf neural networks to cancer classification based on gene expressions. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pages 1930–1934. IEEE, 2006.

[14] Xiaoheng Deng, Weijian Zhong, Ju Ren, Detian Zeng, and Honggang Zhang. An imbalanced data classification method based on automatic clustering under-sampling. In *2016 IEEE 35th international performance computing and communications conference (IPCCC)*, pages 1–8. IEEE, 2016.

[15] Shaza M Abd Elrahman and Ajith Abraham. A review of class imbalance problem. *Journal of Network and Innovative Computing*, 1(2013):332–340, 2013.

[16] Belkacem Fergani, Laurent Clavier, et al. Importance-weighted the imbalanced data for c-svm classifier to human activity recognition. In *2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA)*, pages 330–335. IEEE, 2013.

[17] Honggui Han, Qili Chen, and Junfei Qiao. Research on an online self-organizing radial basis function neural network. *Neural Computing and Applications*, 19(5):667–676, 2010.

[18] Hsiao-Yun Huang, Yi-Jhen Lin, Youg-Siang Chen, and Hung-Yi Lu. Imbalanced data classification using random subspace method and smote. In *Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS), 2012 Joint 6th International Conference on*, pages 817–820. IEEE, 2012.

[19] Wenhao Huang, Guojie Song, Man Li, Weisong Hu, and Kunqing Xie. Adaptive weight optimization for classification of imbalanced data. In *International Conference on Intelligent Science and Big Data Engineering*, pages 546–553. Springer, 2013.

[20] Yi-Min Huang and Shu-Xin Du. Weighted support vector machine for classification with uneven training class sizes. In *2005 International Conference on Machine Learning and Cybernetics*, volume 7, pages 4365–4369. IEEE, 2005.

[21] Jae Pil Hwang, Seongkeun Park, and Euntai Kim. A new weighted approach to imbalanced data classification problem via support vector machine with quadratic cost function. *Expert Systems with Applications*, 38(7):8580–8585, 2011.

[22] Jaedong Lee and Jee-Hyong Lee. K-means clustering based svm ensemble methods for imbalanced data problem. In *2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS)*, pages 614–617. IEEE, 2014.

[23] Yang Liu, Xiaohui Yu, Jimmy Xiangji Huang, and Aijun An. Combining integrated sampling with svm ensembles for learning from imbalanced datasets. *Information Processing & Management*, 47(4):617–631, 2011.

[24] Sebastián Maldonado and Julio López. Dealing with high-dimensional class-imbalanced datasets: Embedded feature selection for svm classification. *Applied Soft Computing*, 67:94–105, 2018.

[25] Tomasz Markiewicz, Stanislaw Osowski, Bonenza Marianska, and Leszek Moszczynski. Automatic recognition of the blood cells of myelogenous leukemia using svm. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 4, pages 2496–2501. IEEE, 2005.

[26] Josey Mathew, Chee Khiang Pang, Ming Luo, and Weng Hoe Leong. Classification of imbalanced data by oversampling in kernel space of support vector machines. *IEEE transactions on neural networks and learning systems*, 29(9):4065–4076, 2017.

[27] Mohammed Aldasht Murad Alkubabji. One class genetic-based feature selection for classification in large datasets. *Master Thesis,Palestine Polytechnic University*, 2015.

[28] Yi L Murphey, Hong Guo, and Lee A Feldkamp. Neural learning from unbalanced data. *Applied Intelligence*, 21(2):117–128, 2004.

[29] Mark JL Orr et al. Introduction to radial basis function networks, 1996.

[30] Priti Sudhir Patki and V Kelkar. Classification using different normalization techniques in support vector machine. *International Journal of Computer Applications*, 975:8887, 2013.

[31] Daniel Peralta, Sara del Río, Sergio Ramírez-Gallego, Isaac Triguero, Jose M Benitez, and Francisco Herrera. Evolutionary feature selection for big data classification: A mapreduce approach. *Mathematical Problems in Engineering*, 2015, 2015.

[32] Bruce W Porter, Ray Bareiss, and Robert C Holte. Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence*, 45(1-2):229–263, 1990.

[33] Hartayuni Sain and Santi Wulan Purnami. Combine sampling support vector machine for imbalanced data classification. *Procedia Computer Science*, 72:59–66, 2015.

[34] Arpit Singh and Anuradha Purohit. A survey on methods for solving data imbalance problem for classification. *International Journal of Computer Applications*, 127(15):37–41, 2015.

[35] Sergey Sukhanov, Andreas Merentitis, Christian Debes, J Hahn, and Abdelhak M Zoubir. Combining svms for classification on class imbalanced data. In *2018 IEEE Statistical Signal Processing Workshop (SSP)*, pages 90–94. IEEE, 2018.

[36] R Sundar and M Punniyamoorthy. Performance enhanced boosted svm for imbalanced datasets. *Applied Soft Computing*, page 105601, 2019.

[37] Xinmin Tao, Qing Li, Wenjie Guo, Chao Ren, Chenxi Li, Rui Liu, and Junrong Zou. Self-adaptive cost weights-based support vector machine cost-sensitive ensemble for imbalanced data classification. *Information Sciences*, 487:31–56, 2019.

[38] Jiang Tian, Hong Gu, and Wenqi Liu. Imbalanced classification using support vector machine ensemble. *Neural computing and applications*, 20(2):203–209, 2011.

[39] S Vishwanathan and M Narasimha Murty. Ssvm: a simple svm algorithm. In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, volume 3, pages 2393–2398. IEEE, 2002.

[40] Qiang Wang. A hybrid sampling svm approach to imbalanced data classification. In *Abstract and Applied Analysis*, volume 2014. Hindawi, 2014.

[41] Xiaoguang Wang, Xuan Liu, and Stan Matwin. A distributed instance-weighted svm algorithm on large-scale imbalanced datasets. In *2014 IEEE International Conference on Big Data (Big Data)*, pages 45–51. IEEE, 2014.

[42] Xinjie Yu and Mitsuo Gen. *Introduction to evolutionary algorithms*. Springer Science & Business Media, 2010.