CHAPTER 5

DESIGN OF ENHANCED SVM ENSEMBLE CLASSIFICATION SYSTEM

Phase II of the research methodology is dedicated to build an enhanced classification system that can effectively find ham and spam online reviews. This enhanced classifier is then used to build the hybrid classifier in Phase III. The classifier that is enhanced is the Support Vector Machine (SVM), which has been used successfully in several applications Dinh *et al.*, 2021, Gaye *et al.*, 2021; Lumbanraja *et al.*, 2021). The SVM classifier is improved in two manners. The first is to include optimization procedures that would enhance its performance in terms of accuracy and speed. The, second method uses this enhance version to design an ensemble model. This chapter describes the methods used to enhance the single SVM classification model and the ensemble model using enhanced SVM classifier as the base system.

5.1. SUPPORT VECTOR MACHINE

Support Vector Machines (SVMs) were presented by Vapnik (1995) (Basu et al., 2002) and have end up being quick compelling classifiers and works adequately with high dimensional datasets too (Eirinaki, 2009). A Support Vector Machine (SVM) is an idea in software engineering for a bunch of related regulated learning techniques that investigate information and perceive designs that are basically utilized for order and relapse patterns. The standard SVM takes a bunch of input information and predicts, for each given information, which of two potential classes the information is an individual from, which makes the SVM a non-probabilistic binary linear classifier.

Given a bunch of training models, each set apart as having a place with one of two classifications, an SVM training algorithm, constructs a model that relegates new models into one classification or the other. An SVM model is a portrayal of the models as focuses in space, planned so the instances of the different classes are partitioned by an unmistakable hole that is pretty much as wide as could be expected. New models are then planned into that equivalent space and anticipated to have a place with a class dependent on which side of the hole they fall on.

In spite of the fact that SVMs were initially planned as binary classifiers, moves toward that address a multi-class issue as a solitary "all-together" advancement issue exist (Weston and Watkins, 1999). A multi-class characterization task generally includes isolating information into preparing and testing sets. Each example in the preparation set contains one 'target value" (for example class labels) and a few "attributes" (for example features). The objective of SVM is to create a model (in view of the preparation information) which predicts the objective upsides of the test information given just the test information ascribes. Numerically SVM can be portrayed as follows (Boser et al., 1992; Cortes and Vapnik, 1995).

Considering the binary classification case, let $((x_1, y_1) \ldots (x_n, y_n))$ be the training dataset where xi are the component vectors that address the perceptions and $y_i \in (-1, +1)$ be the two marks that every perception can be appointed to. From these perceptions, SVM assembles an ideal hyperplane (a linear discriminant in the part changed higher dimensional component space) that maximally isolates the two classes by the vastest edge by limiting the goal work. For a directly divisible arrangement of 2D-focuses which have a place with one of two classes, discover an isolating straight line is displayed in Figure 5.1a. In this model, there exist different straight lines that different the information focuses into two gatherings. Choosing the optimal divider is a natural basis.

By and large, a line is viewed as terrible on the off chance that it passes excessively near the point in light of the fact that it will be commotion delicate and it won't sum up effectively. Along these lines, the objective here is to discover the line passing beyond what many would consider possible from all points. Along these lines, the activity of the SVM calculation depends on finding the hyperplane that gives the biggest least distance to the training models. Twice, this distance gets the significant name of edge inside SVM's hypothesis. Accordingly, the ideal isolating hyperplane expands the edge of the preparation information. Illustration of an ideal hyperplane is displayed in Figure 5.1b.

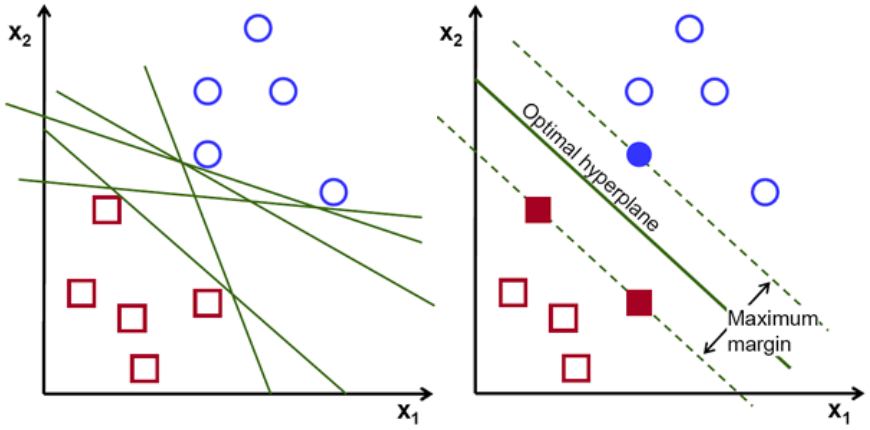

Figure 5.1 : Support Vector Machine Hyperplane

The hyperplane computation method of SVM is described below. Let the hyperplane be defined as below (Equation 5.1).

$$f(x) = \beta_0 + \beta^T x \qquad (5.1)$$

where β is for representing the weight factor and β_0 is the inclination. The optimal hyperplane is addressed in an endless number of various ways by scaling of β and β_0.

As an issue of show, among every one of the potential portrayals of the hyperplane, the one picked is given by Equation (5.2).

$$| \beta_0 + \beta^T x| = 1 \qquad (5.2)$$

where x represents the training models nearest to the hyperplane. By and large, the training models that are nearest to the hyperplane are called **support vectors** and this portrayal is known as the **canonical hyperplane**. Presently, the after effect of calculation that gives the distance between a point x and a hyperplane $\{\beta, \beta_0\}$ is assessed utilizing Equation (5.3).

$$\text{distance} = \frac{|\beta_0 + \beta^T x|}{||\beta||} \qquad (5.3)$$

Specifically, for the canonical hyperplane, the numerator is equivalent to one and the distance to the support vectors is Equation (5.4).

$$\text{distance}_{\text{support\_vectors}} = \frac{|\beta_0 + \beta^T x|}{||\beta||} = \frac{1}{||\beta||} \quad (5.4)$$

Let M to indicate the edge which is double the distance to the nearest models (Equation 5.5).

$$M = \frac{2}{||\beta||} \quad (5.5)$$

Presently, the issue of expanding M is identical to the issue of limiting a capacity L(β) subject to certain limitations. The limitations model the prerequisite for the hyperplane to characterize accurately all the preparation models x_i. Officially, it very well may be characterized as Equation (5.6).

$$\min_{\beta,\beta_0} L(\beta) = \frac{1}{2}||\beta||^2 \qquad \text{subject to } y_i(\beta^T x_i + \beta_0) \geq 1 \ \forall i \quad (5.6)$$

where y_I explains every label of the training examples.

The SVM classifier uses different kernel functions, whose generalized form is given in Equation (5.7).

$$K(x_i, xj) \equiv (x_i)^T \phi(x_j) \quad (5.7)$$

The above kernel can belong to any one of the following four types available (Equations 5.8-5.11).

1. Linear Kernel : $K(x, x_j) = x_i^T x_j$ (5.8)
2. Polynomial Kernel : $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$ (5.9)
3. Radial Basis Function (RBF) : $K(x_i, x_j) = \exp(-\gamma||x_i - x_j||^2), \gamma > 0$ (5.10)
4. Sigmoid Kernel : $K(x_i , x_j) = \tanh(x_i^T x_j + r)$ (5.11)

Here, γ, r and d are kernel parameters. This research work uses the radial basis function.

5.2. ENSEMBLE CLASSIFICATION SYSTEM

A vast range of classifiers are available nowadays to categorise data and picking the one that perfectly meets an application is a hard issue. It is important that the features and the classification algorithm match, so as to provide the desired results for a specific application. Ensemble classifiers are one method for accomplishing this. The ensemble classification model, ensures best match, by pooling results from multiple classification results. The ensemble classifier is also called as fusion classifier, multiple classifier or Ensemble of Classifiers (EoC).

The EoC enables classifiers tailored to individual specifications to deal with the numerous requirements of a difficult problem. Numerically, EoC adds another degree of flexibility to the traditional inclination/difference tradeoff, allowing for combinations that would be difficult (if not impossible) to achieve with just a single classifier. In light of these benefits, EoC is used to design the ham/spam detection system.

The important component of the EoC is the set of independent classifiers whose outputs are combined or aggregated to obtain the classification results. It is vital to select these independent classifiers, also called as base classifiers, carefully and should be diverse in order to make it more powerful. A conventional EoC system is shown in Figure 5.2.

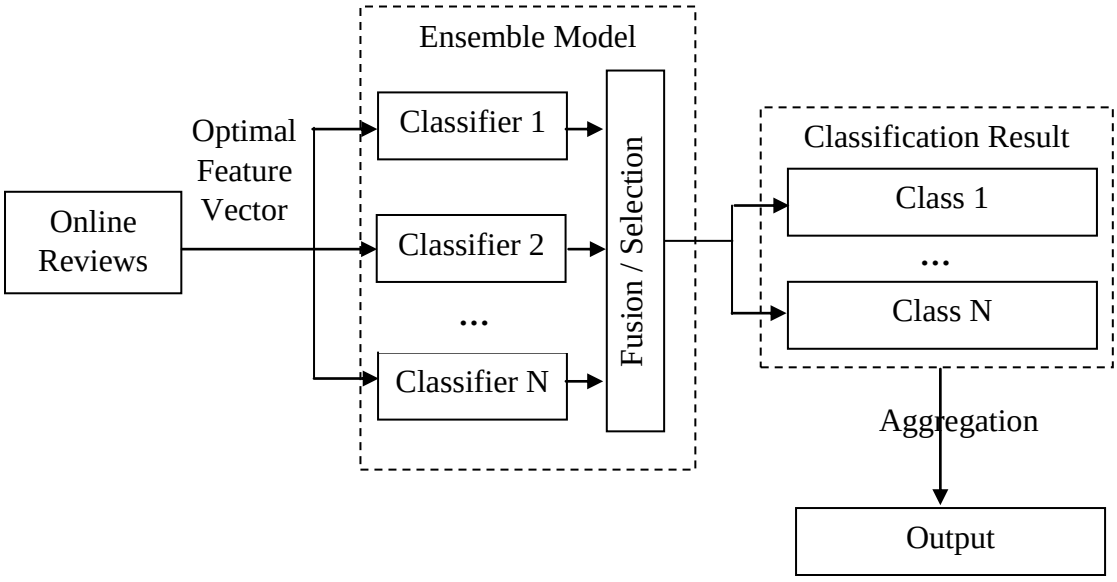

Figure 5.2: The EoC System

The success of the EoC system depends on two important two steps. The first step is the training the component step or base classifier creation step and the second step is the accumulation point, where the results are integrated from the various base classifiers. The EoC system consists of four main steps, as listed below.

(i) Construction of Base Classifiers
(ii) Ensemble Creation
(iii) Aggregation
(iv) Evaluation

5.2.1. Construction of Base Classifier

This step requires information regarding the number of base classifiers to create and type of classifier. There is no universal method available to determine both these factors and in most of the cases depends on the application and dataset used. The main point to ensure that the selected base classifier's performance is high and there exist maximum diversity between the selected classifiers. The performance of a classifier is generally determined using its classification accuracy. The diversity between base classifiers can be estimated using Yule's Q method (Kuncheva and Whitaker, 2003). During the design of EoC, all the classifiers that satisfy the accuracy and diversity criteria are selected as base classifiers during the construction of Ensemble of Classifier model.

5.2.2. Ensemble Creation

According to Yildırım *et al.* (2019), several methods for creating ensembles have been proposed. Examples include knowledge-based methods, randomization methods, training data manipulation methods, manipulation of input data/features and manipulation of target labels (Banfield *et al.*, 2004).

5.2.3. Aggregation

After creating the ensemble system using the base classifiers, the next step is to use a method that integrates the findings of the base classifiers. For this, two methods, namely, integration or fusion methods and selection methods have been used (Ahmad, 2019). In the integration approach, all classifiers contribute towards the final decision, while in

the selection method, one classifier is used to give the final decision. The integration method requires the classifiers to be competitive while with the selection methods, it has to be complementation. The second type, selection, combines results in three levels (Large *et al.*, 2019). They are,

- Abstract - Each classifier results with a class label for each test data
- Rank - Each classifier results with a ranking list of all possible classes for each test data
- Measurement - Each classifier results with a score (or probability or confidence level) for each test data

Techniques that are used for this purpose include minimum, majority voting, maximum, sum, average, product, Bayes, decision template and behavior knowledge space.

5.2.4. Evaluation

The last component is used to assess the categorization output's performance or quality. Several metrics can be used for this purpose. However, it should be noted that the method of evaluation depends on the application under consideration and it should also be noted that there are no universal method or one common method for measuring the performance of all EoC models. In general, the metrics that are used for evaluating a single classifier prediction system can be used to evaluate the EoC.

5.3. PROPOSED ENHANCED ENSEMBLE CLASSIFICATION SYSTEM

The proposed enhanced EoC system consists of the above four steps, namely, base classifier construction, ensemble creation, aggregation and evaluation. Details on how these steps are performed in the proposed EoC system is presented in this section.

5.3.1. Construction of Base Classifiers

As mentioned earlier, the base classifier used in the proposed EoS is the SVM classifier. The important feature of the above described SVM classifier is that it assures to reduce error rates (or increaser accuracy) during classification, which is enabled through the generalization capability contributed through the implementation of SRM (Structural

Risk Minimization) principal that helps to construct an optimal hyperplane separating the various classes (Haykin, 1999; Isa *et al.*, 2008; Lee, 2008). However, as mentioned in Methodology, the classifier also suffers from high computing time and cost (Simone, 2014). Several research works have focused on providing solutions to solve these issues (Julia *et al.*, 2020; Xuancang *et al.*, 2020). In this research, it is solved in two manners.

1. Remove irrelevant support vectors which do not have any role during classification.
2. Replace the Euclidean distance used by a more powerful Mahalanobis distance.

The design of the enhanced SVM classifier, incorporating the above solutions, is shown in Figure 5.3.

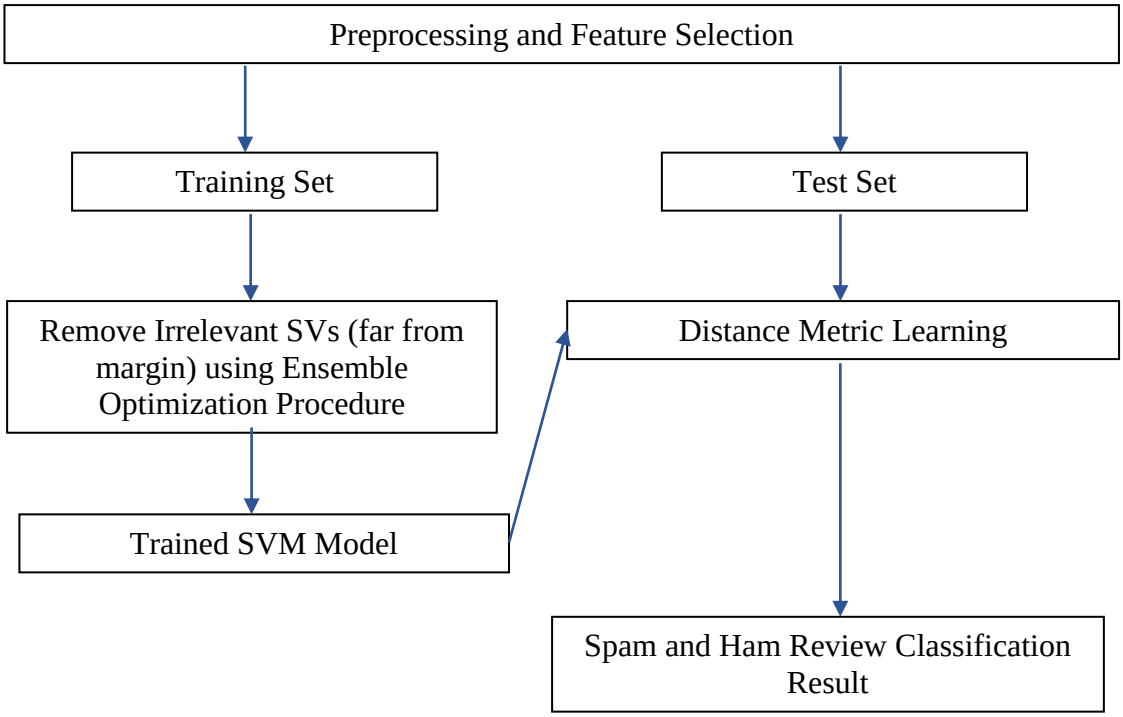

Figure 5.3 : Enhanced SVM Classifier

The optimization procedure focuses on the training set of the review feature set. From the discussion of SVM classifier, it is understood that the classifier uses a hyperplane to differentiate the feature space into two categories, namely, ham reviews and

spam reviews. This step uses an iterative process to identify the Support Vectors (SVs) for the training set (Ray, 2017). It was found that, even after the usage of feature selection algorithm, the resultant feature set is still large. The number of SVs generated by the SVM classifier is inversely proportional to the quality of the feature vector, implying that the SVM classifier generates a large number of SVs. This high number increases the time complexity and has a negative impact on classification performance. These issues can be solved by reducing the number of SVs by selecting only those SVs which can improve the classification performance. This also results in a smaller set of SVs to be analysed.

A popular solution to the above issue is to reduce the size of training dataset. For example, Joachims (1999) proposed a method that created a number of chunks, each having a set of features. Each chunk, whose size is less than the training set, was processed using an iterative technique that converted non-redundant features to SVs. Each chuck, whose size is less than the training set, was processed using an iterative technique that converted non-redundant features to SVs. All such identified SVs were used to train SVM and the rest were removed. This method while memory efficient, still incurred time complexity overhead due to the usage of the iterative technique. As an alternate solution, several researchers used subspace sampling methods to select appropriate subset as training set (Khaire and Dhanalakshmi, 2019). Examples of subspace sampling methods include random subspace selection method, sequential subspace selection method, Adaboost and bagging (Pearson*et al.*, 2019). These methods while successful in minimizing the feature quantity examined by the classifier, did not stop the creation of irrelevant SVs.

Thus, in Phase II, the time complexity is solved by analysing the hyperplane details and retains only relevant SVs. This reduces the number of SVs to be analysed and the classifier is able to use only optimal SVs to train the classifier, thus increasing the accuracy also. The removal or irrelevant SVs optimization procedure is performed using four steps (Figure 5.4).

Step 1 :	Partition the training set into testing and training sets.
Step 2 :	Construct hyperplane
Step 3 :	Identify and remove SVs that make the hyperplane highly complex from the training set.
Step 4 :	Train the classifier using the optimal SVs from Step 3.

Figure 5.4 : Optimization Procedure to Remove Irrelevant SVs

(i) Step 1 : Partitioning

The first step is to partition the optimal feature set into training and testing tests. Let F be the optimal feature set size Z having n-dimensional feature vectors obtained from N features. In general, the training set has to be larger than the testing set. This method partitions Z into two groups, with first group (training) using $2/3^{rd}$ of the feature set as training data and the rest grouped as testing set. The strategy is negative on the grounds that solitary a part of the underlying information is utilized to determine the model.

(ii) Step 2 : Hyperplane Construction

It is well-known fact that the SVM builds the classifier by constructing a hyperplane (constructed using Equation 5.1) that partition the features into different categories, in the feature vector space. Using the hyperplane construction Equation, different number of hyperplanes can be constructed to divide the feature points into two categories (Figure 5.1a), however, only one will be considered optimal (Figure 5.1b). The optimal hyperplane is somewhere between the maximal margin, where the margin is characterized as the amount of distances of the hyperplanes to the support vector. In Figure 5.1b, this margin is denoted as maximum margin and is the summation of two distances d1 + d2, as shown in Figure 5.5. As already mentioned, the optimal separating hyperplane is determined by the closest features (or SVs) from each target class.

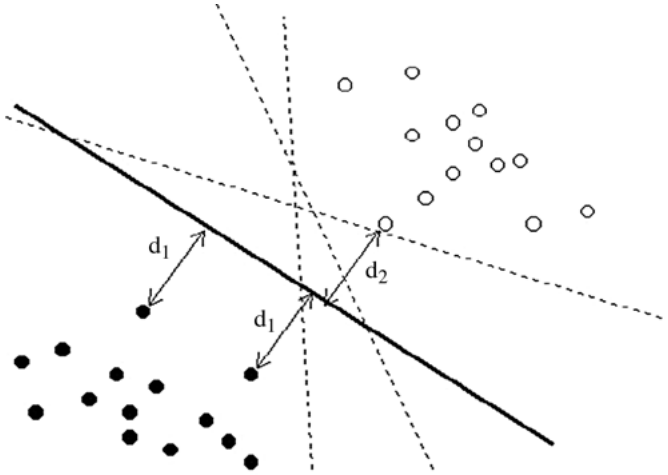

Figure 5.5 : Hyper Planes and Margins

In the non-linear case, kernel functions are used during partitioning. By using kernel functions, the non-linearly separable features can be delineated from the original input space to animmense spatial feature space using non-linear transformation (Figure 5.6). To improve this process, the kernel function in conventional SVM, can be translated to a generalized version (Equations5.6 and 5.7).

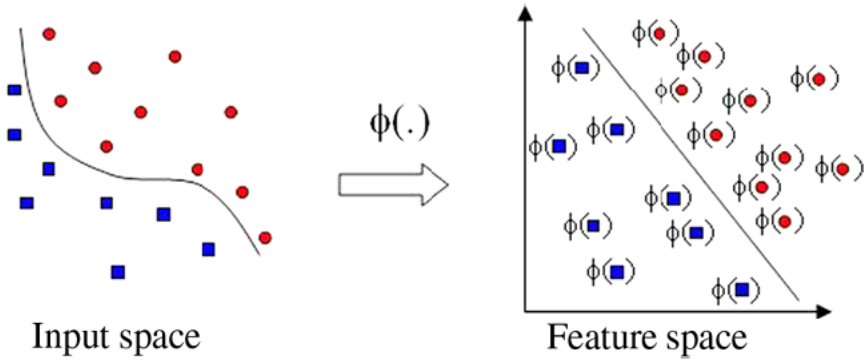

Figure 5.6: Mapping Feature Points into Feature Space Using Kernel Function

Non-linear classification is identical to linear classification, with the exception that a non-linear kernel function replaces every dot product. By mapping feature points into high dimensional space using kernel function, the performance of classifier is increased as it permits SVM to carry out features separation even with complex margins. Again, the optimal kernel, from the various available functions has to be determined.

There are different kernel functions available as shown in Equations (5.8 - 5.11). Each of these kernels have their own properties and manner of responses while handling the feature set (Guo *et al.*, 2010). For example SVM with sigmoid kernel function working is similar to 2-layer perceptron neural network (Burges, 1998). On the other hand, the RBF kernel SVM works similar to RBF neural network. Thus, the classification accuracy of the SVM classifier is directly dependent on the kernel function implemented. This makes the selection and appropriate implement of kernel function is important. Several researches have been conducted to identify the correct kernel function that would produce high classification accuracy for all types of applications (Lee *et al.*, 2012). But, no final solution has been found. In this research work, the problem of identifying the correct kernel is solved through the use of ensemble system. The proposed ensemble system is constructed as a homogeneous system, where the base classifiers are constructed using the four different kernels.

As previously stated, the margin is calculated using Euclidean distance and is defined as the sum of hyperplane to SV distances. When testing, the new review feature vector is projected into the same vector space as the other target classes' SVs, and the average distance between adjacent target classes' SVs is calculated using Euclidean distance. A category is used to make the classification choice, whose average distance is small when compared with the new feature points. In this research work, the conventional usage of Euclidean distance is replaced by Mahalanobis measure. This decision was made to obtain the various advantages of Mahalanobis distance over Euclidean distance (Bhavsar and Ganatra, 2015). The advantages are

(i) the coordinate axes are automatically scaled
(ii) the correlation between the features are corrected automatically
(iii) provides a curved and linear decision boundaries and
(iv) handle efficiently both small and very high dimensional datasets.

The Mahalanobis distance is defined using Equation (5.12)

$$M_{dist} = sqrt((x - m)C_x^{-1}(x-m)^T) \quad (5.12)$$

where x is the new feature point, m is the mean of different classes, C_x is the covariance matrix. Let $m_1 \ldots m_c$ represent the means for the classes and $C_1 \ldots C_c$ be its corresponding co-variances. A feature for testing The Mahalanobis distance between x and each of the means is used to classify vector x, and x is allocated to the class with the shortest Mahalanobis distance. Figure 5.7 shows how this works.

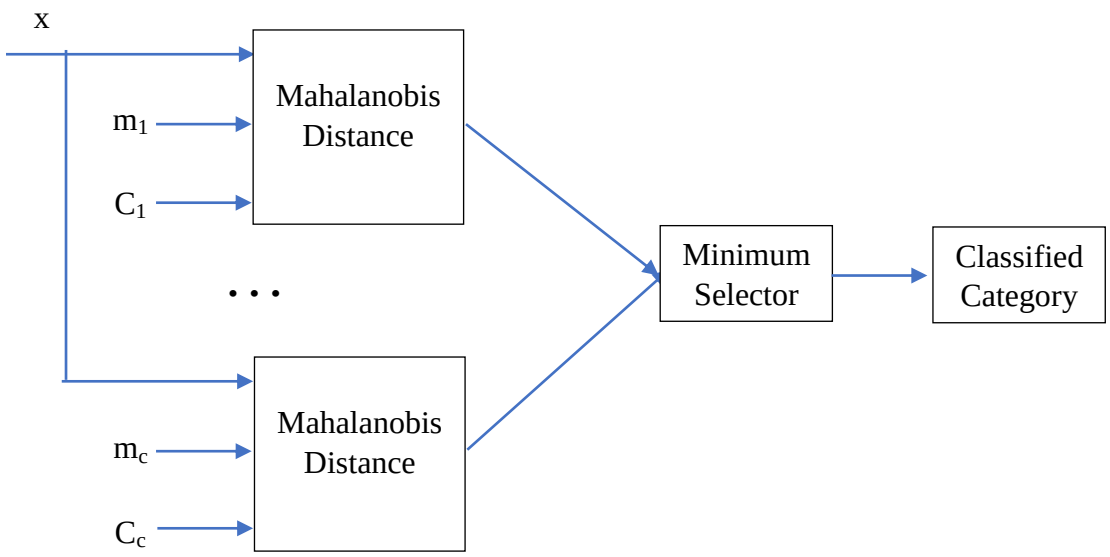

Figure 5.7 : Target Class Assignment While Using Mahalanobis Distance

(iii) Step 3 : Removal of Irrelevant SVs

From Section 5.1, it is understood that the result of the training or learning process are the set of SVs and the separable hyperplane is constructed based on these SVs. Therefore, it is important to identify all those features which can be possible SVs. If features that can be probable SV are eliminated, then the performance of the SVM classifier will degrade. From experiments, it was understood that all features near to the margins of the hyperplane have the strongest possible of being a SV and almost all features far away from the decision boundary do not get to be nominated as an SV.

It can be understood that with SVM, training features that are closer to the hyperplane have maximum likelihood of being a SV. Thus, all features far from hyperplane do not play an important role during SV identification. This observation is used and all features that are at a distance from hyperplane are removed, thus reducing the

size of training set without compromising on the training quality. Now, the major task is to determine which of the training dataset features are placed far from the hyperplane.

The elimination of irrelevant SVs is done using a margin of ensemble approaches in this study. A margin of a feature x is determined through Equation (5.13), according to Schapire's definition.

$$\text{margin}(x, y) = \frac{v_y - \max_{c=1..,L \cap c \neq y}(v_c)}{\sum_{C=1}^{L} v_c} \tag{5.13}$$

where v_y is the vote count for ham class y and v_c is the amount of votes for spam class c. The limit of the margin is [-1 +1]. A positive feature x margin value indicates that it has been appropriately classified, whereas a negative value indicates that it has been incorrectly classified. The bigger the margin, the more confident you can be in your classification. Furthermore, a significant positive score suggests that many of the other ensemble model's basic classifiers effectively handled the target class. This indicates that this characteristic is at or near the centre of the distribution of all features in the linked class. All of these samples represent the associated class's general characteristics.

A big negative number, and from the other hand, suggests that just a handful of the ensemble system's base classifiers are capable of accurately identifying the target class. This may indicate that the features involved are outliers. A value zero indicates that the number of base classifier which correctly identifies the ham and number of classifiers that classified wrongly are equal. This indicates that the feature would be most obviously on the class boundary. These characteristics contain information about both classes. Because they have more knowledge of the target classes, the suggested optimization technique is more interested in features that are on the frontiers of classes. Thus, the margin definition (Equation 5.13) is modified to Equation (5.14).

$$\text{margin}(x) = \frac{vc_1 - vc_2}{\sum_{c=1}^{L} v_c} = \frac{\max_{c=1...L}(v_c) - \max_{c=1..L \cap c \neq c_1(v_c)}}{\sum_{c=1}^{L} v_c} \tag{5.14}$$

Here c_1 and vc_1 are the maximum voted class for sample x and the number of corresponding votes respectively. Similarly, c_2 and vc_2 are the second most voted class along with its related votes. The modified margin has a range of 0 to +1. The lesser the margin, the closer the linked feature is near the class boundary and hence significantly better. Furthermore, the true class label of sample x is not required by the updated margin.

According to the conventional SVM classifier, to develop the classifier, only certain training data near the boundaries is required. Moreover, in order to reduce the training time, all the SVs that are far away from the boundary are deleted prior to training. This will aid in increasing training speed. The proposed ensemble system consists of the following steps.

1. Construct the ensemble classifier using the full training set
2. Compute the modified margin for each training instance
3. Order all training instances based on their margin values
4. Select smallest SVs as optimal SVs
5. Use these optimal SVs to train the classifier.

5.3.2. Aggregation Component

The output from the base classifiers are combined using majority voting scheme. Because of its simplicity and quickness, the majority voting scheme is one of the oldest strategies. The procedure is outlined below.

Let $d_{t,\ j} \in \{0, 1\}$, t = 1,..T and j = 1,..., C be the i^{th} classifier's decision, where T is the number of classifiers and C is the number of classes. If the i^{th} classifier selects class j, $d_{t,j}$= 1; otherwise, $d_{t,j}$= 0. A class is a type of voting system that uses a majority vote according to Equation (5.15).

$$\sum_{t=1}^{T} d_{t,J} = \max_{j=1}^{c} \sum_{t=1}^{T} d_{t,j} \qquad (5.15)$$

Under the following reasonable hypotheses, majority voting is an optimal combination rule:

• An arbitrary amount of classifiers for a two-class problem, the majority voting is a makes getting rule.

• For any occurrence x, the probability of each classifier selecting the proper class is p; and

• The outputs of the classifier are self-contained.

If at least $\lfloor T/2 \rfloor + 1$ classifiers chose the correct label, the fusion classifier reached a crucial conclusion with a majority vote, where the floor leader takes the highest integer less than or equal to its parameter. The binomial distribution can be used to express the accuracy of the fusion classifier as the overall chance of selecting k T/2 + 1 intelligent ones out of T classifiers, where p is the chance of success for every classifier As a result, P_{ens}, the probability of successful fusion classification is provided in Equation (5.16).

$$P_{ens} = \sum_{k=(T/1)+1}^{T} \binom{T}{k} p^{k} (1-p)^{T-k} \qquad (5.16)$$

Each classifier seems to have the same weight in majority voting, and the ultimate outcome is determined by ensemble members voting. Irrespective of each classifier's generalizability's diversity and quality, it usually takes over than half of the ensemble members to converge on a conclusion before it can be recognised as the ensemble's final output.

However this method is simple to use and implement, there are a number of fundamental issues with majority voting (Shahzad and Lavesson, 2013; Yu, 2012). To begin with, it neglects the fact that some classifiers in the minorities do occasionally produce right results. Second, if an excessive number of ineffective and uncorrelated classifiers are used, the conclusions made by using a single classifier may be worse than those obtained by utilising a majority vote. Third, when employed in particular contexts, such as the likelihood of outliers, it ignores disparities in projected performance. It ignores the existence of diversity, which is the ensemble's goal, at the stage of combining (Yang and Browne, 2004; Yu *et al.*, 2008; Yu *et al.*, 2010). In order to solve these challenges, a weighting mechanism can be used.

The study team utilised a combo of majority vote and a weighting procedure to pool the results of the classifiers. Below is an explanation of the modified majority vote scheme, which includes a weighting mechanism (Equation 5.17).

Let d_t, j $0 \in$ 1, 1, t = 1,..T and j = 1,..., C be the i^{th} classifier's choice, where T is the number of classifiers and C is the number of classes. If the i^{th} classifier selects class ω, then $d_{t,j}$= 1 and 0, respectively. If a majority voting scheme is used, a class j is selected.

$$\sum_{t=1}^{T} d_{t,J} = \max_{j=1}^{c} \sum_{t=1}^{T} d_{t,j} * w^t \qquad (5.17)$$

Here, the weight allocated to the classifier t is w_t, which is determined using Kuncheva's (2004) approach (Equation 5.18).

$$w^t = \log \frac{p^t}{1 - p^t} \qquad (5.18)$$

5.3.3. Evaluation

The main task of evaluation component is to analyze the performance of the EoC system using performance metrics. Examples of performance metrics include accuracy, F-measure (related to precision and recall), specificity and sensitivity of predictions. The method of evaluation and results obtained from these experiments are presented and discussed in Chapter 7, Results and Discussion.

5.3.4. Type of Training

A multiple classifier system can be trained in a number of different methods.

- Individual and aggregate classifier training that does not necessitate extra teaching (e.g., aggregation techniques like average, minimum, product, maximum, etc.)
- Individual classifiers are trained first, then the aggregation is trained.
- The entire scheme is being trained at the same time.

After the individual classifier has been trained, the current study strategy applies the first procedure, in which no further categorising is required. This strategy was chosen since the fusion classification is dependent on the outcomes of the individual classifier.

Figure 5.8 shows the various steps used to construct the proposed enhanced EoC system, which is merely the consolidation of the steps presented in the previous section.

Input

Base Classifier – SVM, Training set – S with size N_T, No. of Base Classifiers - 4, Sampling Ratio (SR) : 0.632, Size M of optimal SVs

Procedure

Step 1 : Create training and testing sets

Step 2 : Training Stage

For each kernel do (Radial Basis function, sigmoid, linear and polynomial kernel)

Map feature points to feature space using selected kernel function

Use ensemble margin algorithm to select optimal SVs

Compute margin of each training feature for each category

Order training features in ascending order of margin values

Select first M smallest margin as optimal support vectors (SVs)

Calculate mean of SVs for each category

Step 3 : Testing Stage

For each feature point

Calculate distance between test data and mean of SVs using Mahalanobis distance m

Classify test data into the category whose distance is small

Step 4 :Use Majority Voting Algorithm to aggregate the results

Figure 5.8 : Steps in the Design of the Proposed Enhanced EoC System

5.4. CHAPTER SUMMARY

This chapter presented an enhanced ensemble classification system that used an enhanced SVM classifier as base classifier. The main motivation of this research work is to design hybrid models. In order to design an efficient hybrid model, the research proposes the combination of classification, clustering and ensembling system proposed in

this chapter. Phase II of the research work thus proposes an ensemble algorithm that is enhanced by incorporating the following enhancement methods.

1. Ensemble creation using different kernel functions
2. Usage of enhanced SVM classifier (through the removal of irrelevant SVs) as base classifier
3. Usage of an efficient distance measure during SVM classification

All the three steps are combined into a single ensemble creation algorithm. Detailed description on the design of these hybrid systems are presented in the next chapter, Chapter 6, Design of Hybrid Classification Systems.