CHAPTER 6

DESIGN OF HYBRID CLASSIFICATION SYSTEMS

Hybridization is described as the process of blending various or more algorithms to solve a single problem by transitioning between them dynamically and aim to improve the performance of the classification systems. In general, hybrid systems are created by combining algorithms from the same domain. That is, hybrid systems combine either two or more classification algorithms or two or more clustering algorithms. In a different path, hybridization can also combine algorithms from different domains, like, combining classification and clustering algorithms. The main focus of Phase III of the research methodology is to design hybrid classification systems that are formed through the systematic combination of classification, clustering and ensembling algorithms and are designed with the aim of reducing the error rate of ham and spam review identification systems. This chapter presents details regarding the hybrid system that combines these three algorithms.

6.1. PROPOSED HYBRID SYSTEMS

As outlined in Chapter 3 (Methodology), the hybrid system designed has two main steps. The first step uses a single classifier/clustering algorithm, while the second uses the ensemble classification proposed in Phase II of the research methodology. All the hybrid systems proposed uses the optimal feature vector constructed in Phase I. Again, as given in Chapter 3 (Methodology), three types of hybrid systems are proposed, each with a different way of combining clustering and classification methods.They are re-listed for ease of discussion.

- Type 1 : Hybrid System using clustering algorithm in step 1 and ensemble classification algorithm in step 2
- Type 2 : Hybrid System using classification algorithm in step 1 and ensemble classification algorithm in step 2
- Type 3 : Hybrid System using classification followed by clustering in step 1 and ensemble classification algorithm in step 2

The aim here is to identify the best type that improves the ham/spam detection accuracy rate. For this purpose, three clustering algorithms and four classification algorithms were selected. These algorithms were chosen because it is very popular in the research field and has high accuracy rates. The selected algorithms are listed below.

- Clustering algorithms
  - K-Means (KM) Clustering Algorithm
  - Mean Shift (MS) Clustering Algorithm
  - Expectation-Maximization (EM) Clustering Algorithm
- Classification algorithms
  - Support Vector Machine (SVM)
  - K-Nearest Neighbour (KNN)
  - Naïve Bayes (NB)
  - Enhanced SVM

Using the above algorithms, seven hybrid systems are proposed (Table6.1), which differ in the combination in which these algorithms are applied. Irrespective of the algorithm used in Step 1, Step 2 always uses the enhanced ensemble system that used enhanced SVM as base classifier.

TABLE 6.1

LIST OF PROPOSED HYBRID SYSTEMS

Type	System	Step 1	Step 2
Hybrid System 1			
1	KM_ESVM	KM	Enhanced Ensemble Based on Enhanced SVM
	MS_ESVM	MS	
	EM_ESVM	EM	
Hybrid System 2			
2	SVM-ESVM	SVM	Enhanced Ensemble Based on Enhanced SVM
	KNN-ESVM	KNN	
	NB-ESVM	NB	
Hybrid System 3			
3	SVM_KM_ESVM	SVM + KM	Enhanced Ensemble Based on Enhanced SVM

In all the hybrid systems, Step 1 is treated as a preprocessing step and is applied only on the training set. This step removes unrepresentative features that negatively impact the performance of the ensemble classifier in Step 2. It is treated as a feature space reduction step, as removal of unrepresentative features present in training data reduces its size and produces a quality improved set. Table 6.2 shows the details of the hybrid ensemble systems used in Step 2.

TABLE 6.2

DETAILS OF THE HYBRID ENSEMBLE SYSTEMS

Factors	Details
Number of Classifiers	4 classifiers
Step 1 Clustering Algorithms	KM, MS and EM
Step 1 Classification Algorithms	SVM, KNN and NB
Step 2 Classifier	Enhanced Ensemble System Based on Enhanced SVM classifier
Feature Vectors	Optimal Feature Vector from Phase I
Ensemble Creation Methods	Four kernels of SVM
Partitioning Method Used	Hold-out method
Aggregation Method	Weighted Majority Voting Algorithm
Type of Training	Training of the individual classifiers and applying aggregation that does not require further training

6.2. CLASSIFICATION ALGORITHMS

This section describes the classification algorithms used in the design of the proposed hybrid system. As the description of the SVM classifier was already provided in Chapter 4, the rest of the two selected classifiers, namely, KNN and NB are provided in this section.

6.2.1. KNN Classifier

The K-Nearest Neighbour Classifier (Cover and Hart, 2006) is a well-known statistical pattern identification technique that has been thoroughly investigated by a number of academics. In their investigations on various data sets, these research have shown that the KNN algorithm generates very good results (Ayyad *et al.*, 2019; Reeve and Kaban, 2019; Gou *et al.*, 2019).

The KNN classifier assigns the sign of the majority to the k closest points of a data point. To break ties, it's normal to choose k tiny and odd numbers (typically 1, 3 or 5). Larger K factors guide to prevent the occurrence of noisy points in the training data set, and cross-validation is frequently used to choose K. It's a non-parametric categorization system, which means it's not based on a set of rules, where each member of a "target" dataset is classified using the training dataset. Figure 6.1 depicts the algorithm (Purohit et al., 2011). The ideal value for k is determined by the data; in general, bigger values of k lessen the impact of noise on classification while making class boundaries less apparent. In tests, the value of 'K' was set to 3 (K=3).

1. For each row (case) in the target dataset (the set to be classified), locate the k closest members (the k nearest neighbors) of the training dataset.
2. A Euclidean Distance measure is used to calculate how close each member of the training set is to the target row that is being examined.
3. Examine the k nearest neighbors to find the class that is very near to the category and assign this category to the row being examined.
4. Repeat this procedure for the remaining rows (cases) in the target set.

Figure 6.1 : KNN Classification Algorithm

In summary, the input is a class membership and the output is the k closest training instances in the feature space. An item is classed here by a majority vote of its neighbours, with the object being allocated to the class that has the most members among its k nearest neighbours. If K = 1, the item is simply ascribed to that single nearest neighbor's class. If K = 1, the item is simply assigned to that single nearest neighbor's class. The KNN classifier is an example of instance-based or lazy learning, in which the function is only approximated locally and full computation is postponed until classification. The KNN algorithm is one of the most basic of all machine learning algorithms, with the advantage of consistently great performance without making any assumptions about the distributions from which the training examples are derived.

6.2.2. NB Classifier

A Naive Bayes classifier is a simple probabilistic classifier that uses Bayes' theorem (from Bayesian statistics) and strong (Naive) independence assumptions to classify data. A Naive Bayes classifier, in simple terms, posits that the presence (or lack) of one feature of a class is independent of the values (or absence) of any such feature. Naive Bayes classifiers can be learned very efficiently in a supervised learning scenario, depending on the particular nature of the probability system. In many practical applications, the maximum likelihood technique is used to estimate parameters for Naive Bayes systems; in other words, one can work with the Naive Bayes (NB) probabilistic system without believing in Bayesian probability or utilising any Bayesian methods.

The Naive Bayes technique works by estimating the probabilities of categories presented in a review using the combined probabilities of words and categories. The Naive Bayes Theorem, which may be stated as Equation (6.1), underpins this strategy.

$$P(c_j \mid d) = \frac{P(c_j)P(d \mid c_j)}{P(d)} \qquad (6.1)$$

where, $P(c_j| d)$ is the posterior probability of observing class c_j given feature vector 'd,' $P(d \mid c_j)$ is the prior probability of exploring feature vector d given event that occurs of class c_j, $P(c_j)$ is the proportion of training examples in category c_j (Choi, and Yao, 2005), and P(d) is the probability that a randomly selected review has vector d as its representation (Sahlgren and Swanberg, 2000). Only $P(d \mid c_j)$ is needed to be computed because P (d) is constant for all classes.

It would be exceedingly computationally expensive to compute $P(d \mid c_j)$ for data sets with many features. To solve this problem, it is assumed that any two coordinates of the feature vector are statistically independent of each other when viewed as random variables; this assumption is conveyed by the Equation (6.2) (Sebastiani, 2002).

$$P(d \mid c_j) = \prod_{k=1}^{m} P(w_k \mid c_j) \qquad (6.2)$$

where, w is the number of different features in the review collection, and m is the number of distinct features in the review collection. Naive Bayes classifiers are probabilistic classifiers that make this assumption. Training time scales linearly with the number of all training documents since training consists of counting the features in training documents. Because classification grows linearly with the number of categories, it may be done by searching up the pre-computed probabilities for each feature vector. The Naive Bayes algorithm classifies reviews by first calculating the likelihood of each feature in the training set that belongs to a certain class. The probabilities associated with attributes are then combined to assess the likelihood that this review belongs to distinct classes. Finally, it assigns the review to the most likely classification.

The Naive Bayes classifier has the advantage of using only a little amount of training data to estimate the classification parameters (variable means and variances). Since factors are assumed, only the deviations of the variables in each class must be estimated, not the entire covariance matrix.

6.3. CLUSTERING ALGORITHMS

This section presents details regarding the working of the three clustering algorithms, namely, K-Means, Mean Shift and Expectation Maximization algorithms.

6.3.1. K-Means Clustering Algorithm

The K-Means clustering algorithm (Hugo, 1957; Edward, 1965; MacQueen, 1967) produces a set of disconnected, flat (non-hierarchical) groups. It works well for forming globular clusters. Figure 6.2 depicts the classic K-Means algorithm (https://en.wikipedia.org/wiki/K-means_clustering).

.

Step 1 : Begin with a decision on the value of k = number of clusters

Step 2 : Put any initial partition that classifies the data into k clusters.

1. Take the first k data as single-element clusters
2. Assign each of the remaining (N-k) data to the cluster with the nearest centroid. After each assignment, recompute the centroid of the gaining cluster.

Step 3 : Take each data in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the cluster gaining the new sample and the cluster losing the sample. The distance metric used is Euclidean distance.

Step 4 : Repeat step 3 until convergence is achieved, that is, until a pass through the training sample causes no new assignments.

Figure 6.2: K-Means Algorithm

The SSE (Sum of Squared Error) is the most often used convergence criteria (objective function) for the K-Means method (Equation 6.3).

$$SSE = \sum_{j=1}^{k} \sum_{x_i \in c_j} \| x_i - \mu_j \|^2 \qquad (6.3)$$

where, n_j denotes the number of instances in cluster c_j, and $\mu_j = \frac{1}{n_j} \sum_{x_i \in c_j} x_i$ denotes the mean of cluster c_j. The K-Means algorithm converges to a local minimum in all cases. The specific local minimum that is discovered is determined by the beginning cluster centroids. The cluster centroids are updated using the K-Means algorithm until a local minimum is established. Because cluster proximity does not always require the 'central,' every member of a cluster is closer to its cluster than any other cluster.

The K-Means technique is an iterative, unsupervised, non-deterministic method that has the following qualities (Caoa et al., 2009).

- K clusters are always present,
- They are non-hierarchical and do not overlap.

6.3.2. Mean Shift Algorithm

A process for determining the maxima (modes) of a density function given discrete data collected from that function (*Yizong, 1995*) is known as mean shift. This is an iterative approach that begins with an estimate of x. Let's choose a kernel function, $K(x_i - x)$. For re-estimation of the mean, this function calculates the weight of close points. On the distance to the current estimate, a Gaussian kernel is typically utilised (Equation 6.4).

$$K(x_i - x) = e - c \| xi - x \|^2 \quad (6.4)$$

Equation (6.5) gives the weighted mean of the density in the window determined by K.

$$m(x) = \frac{\sum_{x_i \in N(x)} K(x_i - x) x_i}{\sum_{x_i \in N(x)} K(x_i - x)} \quad (6.5)$$

where N(x) is a set of points in the vicinity of x for which $K(x_i) \neq 0$. The difference $m(x) - x$ is referred to as the mean shift (Fukunaga and Hostetler, 1975). The mean-shift algorithm now sets $x \leftarrow m(x)$ and continues to estimate until m (x) converges.

Assume that data is a finite set S embedded in an n-dimensional Euclidean space, X. Let K be a flat kernel that represents the λ-ball's characteristic function in X (Equation 6.6).

$$K(x) = \begin{cases} 1 & \text{if } \| x \| \leq \lambda \\ 0 & \text{if } \| x \| > \lambda \end{cases} \quad (6.6)$$

$s \leftarrow m(s)$ is performed for all $s \in S$ simultaneously in each iteration of the algorithm. Given a sparse set of data, the task is to estimate the density function. Smoothing the data, for example, by convolving it with a fixed kernel of width h, is one of the easiest ways (Equation 6.7).

$$f(x) = \sum_i K(x - x_i) = \sum_i k\left(\frac{\| x - x_i \|^2}{h^2}\right) \quad (6.7)$$

Where, the input features are x_i, and the kernel function is k(r). The bandwidth is the only parameter in the method, and it is called h. Kernel density estimation, often known as the Parzen window technique, is a method for estimating density. Once the above equation has been used to compute f (x), the local maxima can be identified using gradient ascent or another optimization approach. The problem with this "brute force" strategy is that evaluating f(x) over the entire search space becomes computationally expensive for greater dimensions. Instead, mean shift employs a form of multiple restart gradient descent, which is well-known in the optimization literature. Starting with a hunch for a local maximum, y_k, which might be a random data point x_1.The gradient of the density estimate f(x) at y_k is computed via mean shift, and an uphill step is taken in that direction (Szeliski, 2011).

The mean shift algorithm can use different types of kernels. A kernel is defined as follows:

Let X be the n-dimensional Euclidean space, R^n. The norm of x is a non-negative number, $||x||^2 = x^T x \geq 0$. A function $K : X \rightarrow R$ is said to be a kernel if there exists a profile, $k : [0, \infty] \rightarrow R$, such that, $K(x) = k(||x||^2)$ and k is non-negative, increasing and piecewise continuous.

Flat and Gaussian kernel profiles are the most commonly utilised kernel profiles for mean shift. This research work uses the Guassian kernel (Equation 6.8).

$$k(x) = e^{-\frac{x^2}{2\sigma^2}} \quad (6.8)$$

where h is the bandwidth parameter and s is the standard deviation parameter.

Consider a set of points in two-dimensional space while applying mean shift for clustering. Assume the kernel is a circular window with a radius of r and a centre of C. Mean shift is a hill climbing algorithm that shifts this kernel to a higher density region iteratively until convergence. A mean shift vector is used to define each shift. The mean shift vector always points in the direction of the greatest density increase. The kernel is

relocated to the centroid or mean of the points within it at each iteration. The mechanism for determining this mean is determined on the kernel selected. If a Gaussian kernel is used instead of a flat kernel in this example, each point will be given a weight that will decline exponentially as the distance from the kernel's centre grows. There will be no direction in which a shift can accommodate more points inside the kernel at convergence.

6.3.3. EM Clustering Algorithm

Dempster et al. (1977) first proposed the Expectation-Maximization (EM) technique, which was designed to solve problems with Maximum Likelihood methods. Expectation Maximization blends statistical approach with algorithmic execution and has recently attracted a lot of attention in the field of missing data. Because it assumes incomplete situations have data missing at random rather than missing totally at random, Expectation Maximization has been shown to work better than methods such as list-wise, pair-wise data deletion, and mean substitution (Allison, 2002).

EM is an iterative process that converges to the marginal a posteriori probability function's local maximum (Equation 6.9).

$$p(\theta|x) = p(x|\theta)\, p(\theta) \tag{6.9}$$

Here,$\theta$is the original data's dataset unknown data (Y_2) of the data (X). It is a general method for determining the parameters' greatest likelihood estimate.

The proposed algorithm assumes 10 clusters (C_j=1...10) and let x_k (k=1..N) be an instance belonging to a cluster C_j with probability $P(C_j|x_k)$ and let N be the number of records. It undertakes successive parameter estimates with the goal of approximation$\theta'(t)$ of X when t =0, 1...10 (Equation 6.10). It operates iteratively by applying two phases, namely E Step (Expectation step) and M Step (Maximization step).

$$\theta'(t) = \{\mu_j(t), \sum_j(t)\} \text{ where } j = 1...10 \tag{6.10}$$

- **E-Step :** Estimate the probability of each cluster C_j for the population attribute (Equation 6.11)

$$p(c_J \mid X) = \frac{|\Sigma_J(T)|^{-1/2} \exp^{\eta_j} P_j(t)}{\sum_{k=1}^{M} |\Sigma_J(T)|^{-1/2} \exp^{\eta_k} P_k(t)} \tag{6.11}$$

where

$$\eta_j = -\frac{1}{2}(x - \mu_j(t))^T \sum_j^{-1}(t)(x - \mu_j(t)) \text{ and } \delta_k = -\frac{1}{2}(x - \mu_k(t))^T \sum_k^{-1}(t)(x - \mu_k(t)).$$

Each cluster has its mean (μj) and standard deviation (∑j). At t = 0, the implementation randomly initializes starting values for mean and standard deviation.

- **M-Step:** The M-step updates the parameter estimation for t + 1 using Equations (6.12) to (6.14).

$$\mu_j(t+1) = \frac{\sum_{k=1}^{N} P(C_j \mid x_k) x_k}{P(C_j \mid x_k)} \tag{6.12}$$

$$\sum_j (t+1) = \frac{\sum_{k=1}^{N} P(C_j \mid x_k)(x_k - \mu_j(t))(x_k - \mu_j(t))^T}{\sum_{k=1}^{N} P(C_j \mid x_k)} \tag{6.13}$$

$$P_j(t+1) = \frac{1}{N}\sum_{k=1}^{N} P(C_j \mid x_k) \tag{6.14}$$

The iteration stops when Equation (6.15) is reached.

$$||\theta(t+1) - \theta(t)|| < \varepsilon \tag{6.15}$$

where ||.|| is the mean of the differences between μj(t+1) and μ(t).

In general the concept of missing value handling using EM algorithm can be denoted as Z = (X, Y) where Z is the complete dataset (augmented data), X is the observed data (incomplete data) and Y is the hidden data (missing data). In simple terms, the EM algorithm considers a set of starting parameters and then uses these parameters to estimate the missing data. The complete data is then used to update the parameters. The steps are repeated until convergence.

The EM method is a broad approach to fitting systems to missing data. The link between missing data and unknown parameters in a data system is exploited by EM. It will

be simple to estimate the system parameters if the missing values are known. Similarly, if the parameters of the data system are understood, unbiased predictions for the data can be made for the missing values. Because of the dependency between system parameters and missing values, an iterative strategy is suggested, in which the missing values are predicted first using assumed values for the parameters. The parameter estimations are then updated using these predictions, and the process is repeated. The sequence of parameters leads to maximum-likelihood estimates that implicitly average over the missing value distribution. Figure 6.3 summarises the steps involved in EM in general.

1. Start with initial guesses for the parameters.
2. Expectation Step
    - Calculate the conditional expected value.
3. Maximization Step
    - fill missing data by the conditional expected values.
    - maximize the simplified likelihood.
4. Iterate expectation and maximization steps until convergence.

Figure 6.3 : EM Algorithm

In this work, the EM algorithm is built to guess the average value of Gauss function. Let's say the data set includes k different classes, which means it's made up of a probability distribution that's a combination of k different normal distributions. Two steps are used to create each sample. A random normal distribution is picked from among k normal distributions in the first phase. In the second stage, a sample data set is created based on this distribution. Each datum in the set is processed in the same way.

6.4. TYPE 1 HYBRID SYSTEMS

Hybrid systems belonging to Type 1 category applies clustering to pre-process the training set and uses the enhanced ensemble system for classifying the reviews in ham and spam. Three hybrid systems are proposed, they are KM_ESVM, MS_ESVM and EM_ESVM. Let F be the optimal feature vector obtained from Phase I. The design of the hybrid system starts with the partitioning of F into training (FTr) and testing (FTe) sets.

The enhanced ensemble system using enhanced SVM as base classifier is designed as a binary classification system, as it is required to classify a review into ham or spam review.

6.4.1. Step 1 : Clustering

Step 1 starts by clustering the training vector, FTr to obtain clusters and its centroids. Let the set of centroids obtained be termed as CC = $\{cc_{1,} \ldots, cc_n\}$. The next step considers each cluster in iteration and determines the distance between each feature point f_i and all its nearest neighbours in the same cluster. The results are stored in a vector, named, d. Using d, the shortest distance is determined. Thus, this identifies new distance based features, d_i and its nearest neighbours.

Apart from the above, two more values (V_{Dost1} and V_{Dist2}) are estimated.V_{dist1} is the distance between all feature points and cluster centres, while V_{dist2} represents the distance between each feature point and its closest cluster neighbour. The sum of V_{Dist1} and V_{Dist2} is calculated and stored in a feature vector, FTr_{New}. This vector replaces Ftr, as has several merits over FTr. FTr_{new} is a one-dimensional vector and consists of features that represent the training set in a more efficient manner. Moreover, dimension_size (FTr_{Nnew}) < dimension_size (FTr), which implies that FTr_{New} is a sized reduced version of FTr.

In the above process, the distance measure used is the Euclidean distance. The Euclidean distance, a widely used metric in clustering methods, is a standard metric for geometrical issues. It is calculated as the distance between two locations in two-dimensional space and may be simply measured using a ruler. The Euclidean distance of the two feature points is defined as in Equation (6.16).

$$D_E(t_a, t_b) = \left(\sum_{t=1}^{m} \left| w_{t,a} - w_{t,b} \right|^2 \right)^{1/2} \qquad (6.16)$$

6.4.2. Step 2 :Classification of New Data (Testing)

The FTr_{New} is paired with the original feature set F to test or forecast the step of a new review feature. The procedure of extracting cluster centres and their nearest neighbours is then repeated to create a new set. At this point, just the features of FTe are

taken into account.As a result of this phase, the new distance-based feature set is created (FTe_{New}). T_s and T_{snew} are used to train and test the ensemble classifier in a Type 1 hybrid system. During testing, the cluster whose centroid is closest to the new review features is determined after getting the clusters.The Euclidean distance metric (Equation 6.16) is used once more for this purpose. The cluster with the shortest distance is chosen, and fresh data is assigned to it. The anticipated class is then provided as the resultant cluster.

Figure 6.4 shows the steps involved in Type 1 hybrid systems.

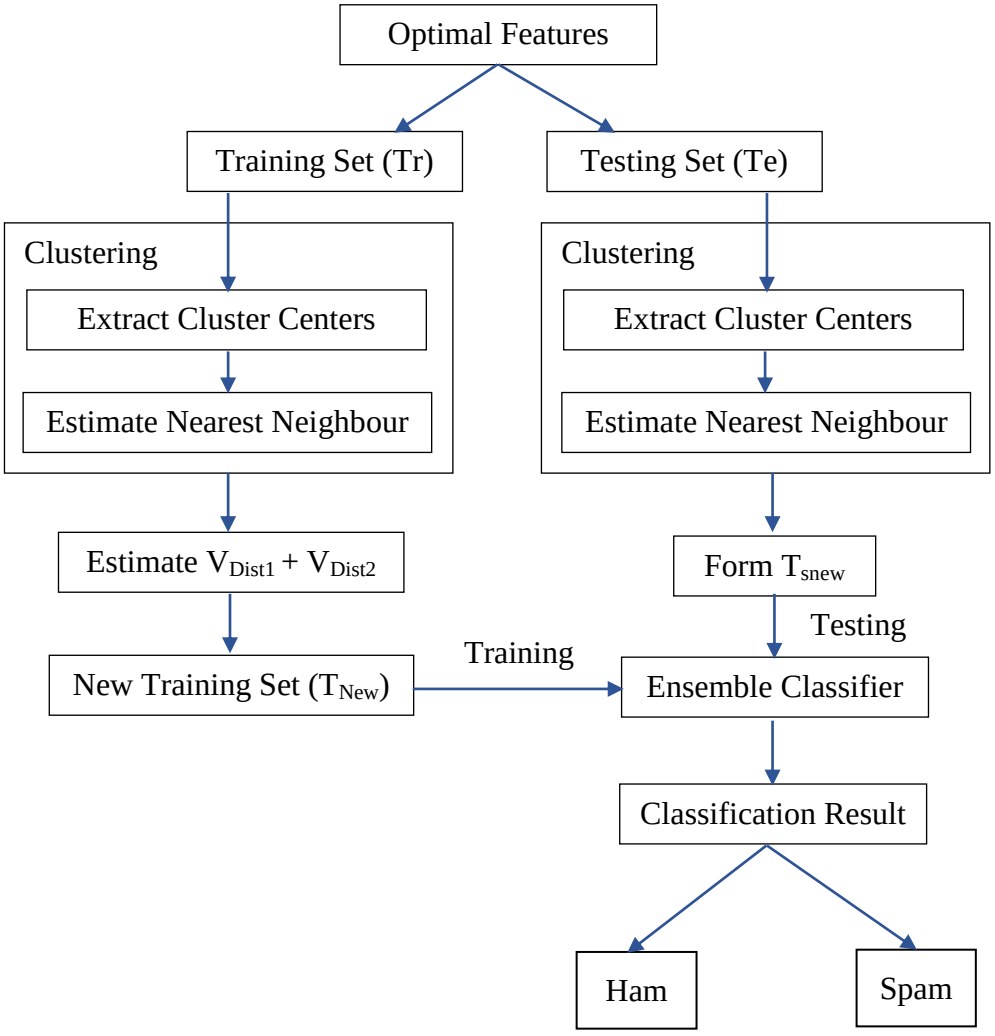

Figure 6.4 : Steps in Type 1 Hybrid Systems

6.5. TYPE 2 HYBRID SYSTEMS

Type 2 hybrid systems uses classifier in step 1 (for preprocessing) and the enhanced ensemble classifier in step 2 for detecting ham and spam image emails. Three type 2 hybrid models, namely, SVM-ESVM, KNN-ESVM and NB-ESVM, are designed.

6.5.1. Step 1 : Classification (Preprocessing)

Step 1 starts by classifying FTr using any one of the selected three classifiers (SVM, KNN and NB). After the analysis of confusion matrix, only the true positives and true negatives, are collected. The false positives and false negatives are treated as noise and outliers and the corresponding features are removed from FTr. This condensed version of FTr (FT_{rNew}) contains optimised features with higher distinguishing capacity, which can improve ham/spam detection accuracy. The reason behind this the new FTr has only positive results. The FTr_{New} is used to train the enhanced ensemble classification system, which is used in step 2.

6.5.2. Step 2 : Classification of New Data (Testing)

All the proposed Type 2 hybrid systems use FTr_{New} to train the enhanced ensemble system. This trained system is then tested using FTe, in order to identify the class to which the test data belongs. Figure 6.5 shows the steps involved in Type 2 hybrid systems.

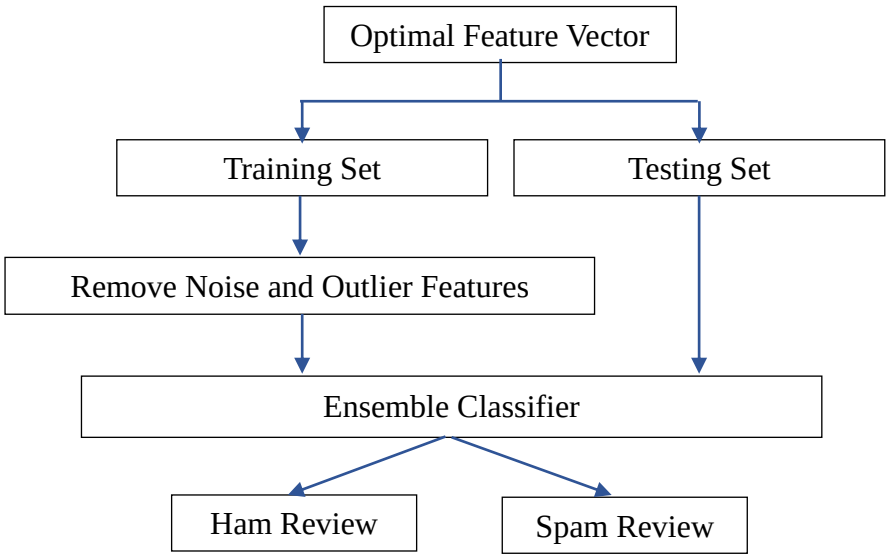

Figure 6.5 : Steps in Type 2 Hybrid Systems

6.6. TYPE 3 HYBRID SYSTEM

Type 3 hybrid system is an amalgamation of type 1 and type 2 systems. This system first uses a classifier to remove noise and outliers, followed by the use of type 1 hybrid system that uses clustering for preprocessing and performs detection using ensemble classification system. Figure 6.6 shows the steps involved.

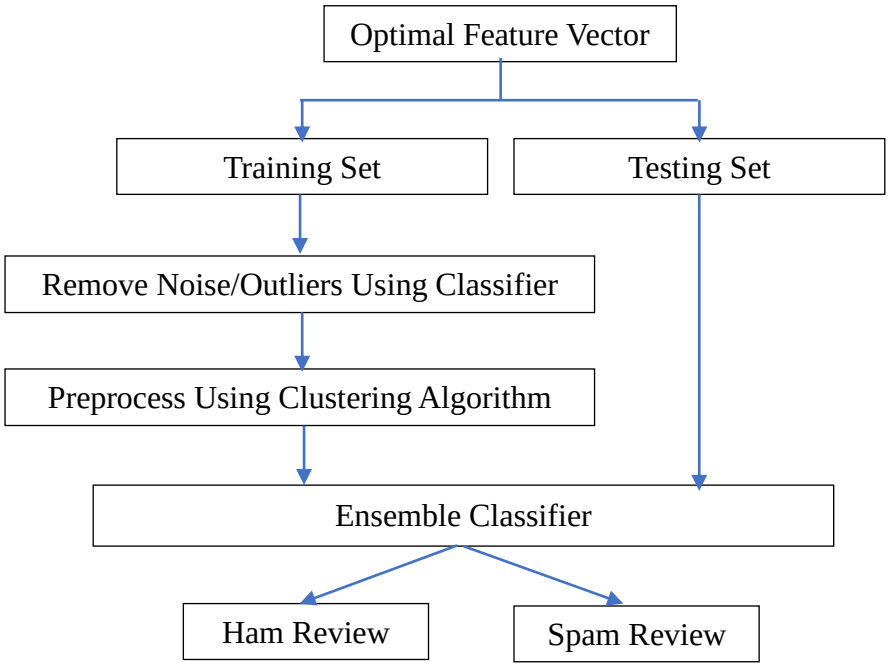

Figure 6.6 : Steps in Type 3 Hybrid Systems

6.7. CHAPTER SUMMARY

The final phase of the research work used clustering and classification algorithms to design hybrid system to identify spam and ham reviews. The proposed hybrid systems were enhanced through the use of ensemble classification system. A total of seven hybrid systems were proposed. One major question while proposing multiple system, is to identify the one which gives maximum performance. This is done using a series of experiments whose results are presented and discussed in the following chapter, Chapter 7, Results and Discussion. This chapter also analyses the proposed feature selection algorithm and studies the effect of the optimization methods incorporated in the enhanced ensemble system.