Email Spam Filtering using Supervised Machine Learning techniques

Pallavi S. Joshi, *Graduate Student, Dept. Of Electrical and Computer Engineering, University of Florida*

***Abstract*— The ever -increasing volume of unsolicited bulk email or spam email has led to the need of more efficient spam filtering techniques. Emails can be classified as spam or ham depending on their length, words or from its domain. In this project, I aim to evaluate and implement some machine learning methods such as Naïve Bayesian classification, k-NN, SVMs for email spam filtering. I aim to comparatively study the algorithms measuring the accuracy, simulation time, pros and cons of various filtering techniques on the Spambase dataset using Python.**

Keywords—Naïve Bayes, Support Vector Machines, K-nearest neighbor, Spam, ham, machine learning

I. INTRODUCTION

The Internet or the World Wide Web has become a standard and the most frequent way of communication in our day to day lives. Emails are the most common way of communicating over the internet. In this new era, people spend most of their time with their electronic gadgets and almost all communication and deals are done virtually instead of being physically present. This is one of the reasons why unsolicited bulk email or commercial is used by companies to advertise their product, or to reach a large amount of people in a short span of time. However, such unsolicited email or spam may not be useful to the users but be rather harmful as it may extract users' information over the internet or it could be a means for spreading viruses. It is very necessary that the users are exposed to only relevant or trustworthy users. In recent statistics, 40% of all emails are spam of which about 15.4 billion email per day and that cost internet users about $355 million per year [2]. Thus, using a knowledge based approach where the email is scanned for good or bad words, its length or domain according to a set of regulations might seem to be a good solution. However, a machine learning approach to spam filtering would not need to follow a set of regulations but would learn from a group of pre-classified spam or ham messages. By ham messages, I refer to intended or safe messages while spam refers to unsolicited bulk or commercial email.This paper aims to classify spam and ham accurately by using Supervised Machine learning techniques on pre-processed data from emails as derived from the Spambase dataset.

II. PROCESSES INVOLVED

Pattern classification is carried out in two basic steps namely training and filtering which is commonly expressed by terms such as fitting to the model and predicting. Depending on the amount of samples used in training and testing, we can predict or classify the data better. Ideally, the more we train the data, the better our prediction should become. In machine learning, the order of sequences in training set can be a deciding parameter in determining the prediction efficiency, which can be called the accuracy.

For practical purposes and experimentation, the dataset must be divided into training and testing set and also must be preprocessed to in order to be defined in terms of probabilistic measures. In other words, the data must be parsed so as to have the frequency of occurrence of each word in the email and must also record the domain from which it was sent. The dataset must also include information for the missing words in the email (the probability of occurrence for the word must be zero for such cases).

Supervised learning/training is a process in machine learning of inferring a function (or a target value) from labeled training data [14]. A supervised learning algorithm analyzes the training data and produces an inferred function, which is further used for mapping the target value for new examples or data [14].

The last step for spam classification would be filtering, that is splitting the dataset according to the predicted value. Different kernel functions such as rbf, linear, polynomial etc is used in practice.

Lastly, the performance of each the spam classifiers needs to be measured which can be determined by the spam precision, spam recall, spam accuracy and simulation time as described in [2].

III. PROBLEMS ASSOCIATED WITH CLASSIFICATION

A. Noise

Everyday data is greatly affected by noise. Images are affected by shadows, poor lighting conditions, blurs due to shaky hands while clicking the image[16]. All this reduces the reliability with the feature values would be measured. In the case of emails, sometimes typing mistakes can lead to presence of words which carry negative meaning which may accidently lead to the filter classifying ham as spam. In a similar way, spam can be mistaken for ham as spammers add noise to the email by intentionally misspelling words to pass through email filters.

B. Overfitting

Overfitting occurs when there are too many attributes and relatively less observations[14,16]. The given dataset may be such that the data may have only two or three labels. But with several attributes which may or may not be present, we increase of chances of misclassifying data. This is because the classifier

becomes so complex so as to identify the trained values perfectly but would not be able to classify simple and novel data patterns.

This condition of a filter being too complex to identify complex or trained data correctly but not being able to adjust to new patterns is called as overfitting.

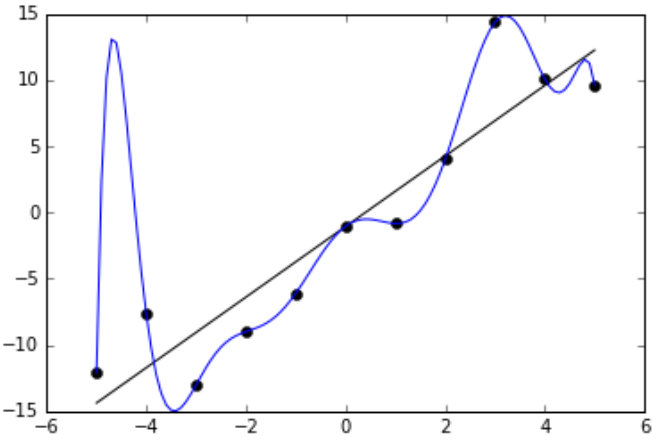

Figure 1. Overfitted data[14]

In the figure above, we see that the polynomial function fits the data perfectly. However, it might not be able to adapt to a new pattern. A linear function can generalize better for new data which has not been fitted yet.

Avoiding overfitting data and finding the correct complexity of the classifier is one of the most important challenges in statistical research. Cross Validation techniques can also be used to address overfitting.

C. Missing values

In pattern classification techniques, we sometimes do not have information about all the features. This gives rise to zero probabilities which might greatly affect the ability of a classifier to discriminate between classes . For the case of email spam filtering, the classifier is trained for only words contained in the emails belonging to the training set. However, test instance may not have all the words that belong to the training set. This will lead to zero probabilities for those word attributes.

IV. K –NN ALGORITHM

K-Nearest Neighbors is a lazy learning algorithm. It is an instance based process which procrastinates induction based process till the classification is performed. In other words, the testing phase of the KNN algorithm is more computationally expensive than the training phase. K-NN aims to exploit spatial redundancy. In that it assumes that with a given dataset, instances with similar properties exist close to each other. So, to obtain the label of a testing instance, the label of the nearest classified instances must be taken into consideration [5]. The test instance class label is assigned as the most frequent label of the nearest neighbor. For the case involving only two classes, a majority vote is sufficient to make a decision.

The training phase fits a set of data attributes to a class label. In the testing phase, the distance of test instance to each instance in the training set is computed .The distance between any two instances can computed based on different distance metrics [5]. This makes the testing phase computationally expensive.

Ideally, the distance metric must minimize the distance between two similarly classified instances, while maximizing the distance between instances of different classes. Different distance metrics such as Minskowsky, Manhattan, Chebyschev, Euclidean Distance etc. can be used to compute the distance between two instances. The formulae for the distance metrics are represented in Table 1 as per [5].

TABLE 1: Distance Metrics and their formulae

Distance Metric	$\boldsymbol{D(x,y)}$
Minkowsky	$\left(\sum_{i=1}^{m}\left\|\|x_i - y_i\|\right\|^r\right)^{1/r}$
Manhattan	$\sum_{i=1}^{m}\|x_i - y_i\|$
Chebychev	$Max\|x_i - y_i\|$,*i=1,2,…m*
Euclidean	$\left(\sum_{i=1}^{m}\|x_i - y_i\|^2\right)^{1/2}$
Canberra	$\sum_{i=1}^{m}\frac{\|x_i - y_i\|}{\|x_i + y_i\|}$
Kendall's Rank correlation	$1 - \frac{2}{m(m-1)}\sum_{i=j}^{m}\sum_{j=1}^{i-1} sgn(x_i - x_j)sgn\,(y_i - y_j)$

For the purpose of this project, Euclidean distance metric has been taken into consideration[5].

$$D(x,y) = (\sum_{i=1}^{m}|x_i - y_i|^2)^{1/2}$$

Consider the figure 2 shown above. The query for the k nearest neighbors starts at the test instance and grows a spherical region or computes distance until encapsulates k training instances, and classifies the test instance by a majority vote of the instances in the nearest neighbors set. The figure depicts a case where k =5 .There are three black points and two red points in 5 nearest neighborhood of X. Thus, X will be labelled as a black point [16].

KNN algorithm has some serious limitations. It is computationally intensive, has large storage requirements and there is no easy way to determine most appropriate value of k for a given dataset. Cross validation can be used to determine a value of k suitable `for the dataset. However, cross –validation is a tedious process [5].

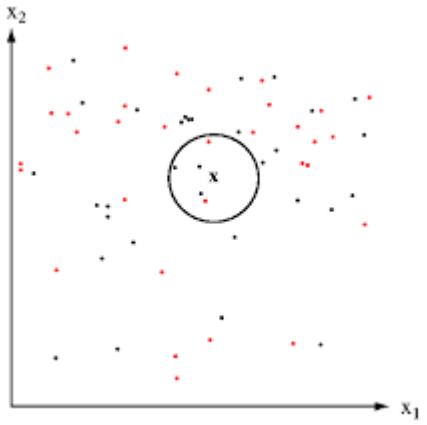

Figure 2.K-NN with k=5 classifying two classes

Wettschereck (1997) [18]performed research on the noisy cases of KNN and found that the performance of KNN was indifferent to large values of K [5]. One nearest neighbor would make more sense when working on a smaller dataset. However, for most large datasets, small values of K provide more robust performance than larger values. This behavior has been verified by my implementation over the spambase dataset.

V. NAÏVE BAYES

Naïve Bayes Classifier is a popular statistical approach to classifying data, especially in document classification and email spam filtering. The linear computational complexity, simplicity and high accuracy makes Naïve Bayes classifier an ideal candidate for classifying large datasets [1]. Its performance is almost comparable to more complex supervised learning algorithms like Support Vector Machines but with the added advantage of less computation time.

Cestnik et al[17,5]first used the Naive Bayes Algorithm for Machine Learning. Naive is a class of algorithms for training classifiers depending on the kind of data distribution they follow. It is based on the assumption that the value of a particular feature is independent of the other features, given the class [14]. However, it rarely is the case that the features are not correlated. Rather, such correlations would help in discriminating classes. Due to this, accuracy of Naïve Bayes is not very high. However, it provides better accuracy for new data patterns as it uses a more general approach by not considering the correlations.

Naïve Bayes algorithm is based on the Bayes theorem for conditional probability. The probability that a given set of n features($x_1, x_2, \dots, x_n$) enclosed in a vector $\vec{x}$ belong to a category or class C is given by,

$$p(c\ |\vec{x}) = \frac{p(c).p(\vec{x}|c)}{p(\vec{x})}$$

For the case that we have more than two categories, if the conditional probability that the n feature set belongs to a class is greater than a threshold T, it is labelled as the corresponding class. For the purpose of email spam filtering, we have only two categories, so we choose the class label as the one whose conditional probability is higher among the two. In other words, we choose the category as the class label for a particular test instance if its conditional probability is greater than 0.5. By setting the threshold to an appropriate value we can vary the accuracy of the naïve Bayes classifier.

$$p(S\ |\vec{x}) = \frac{p(S).p(\vec{x}|S)}{p(S).p(\vec{x}|S) + p(H).p(\vec{x}|H)}$$

$$p(H\ |\vec{x}) = \frac{p(H).p(\vec{x}|H)}{p(S).p(\vec{x}|S) + p(H).p(\vec{x}|H)}$$

Since the spam filtering involves two categories, computing for only one of these quantities is sufficient. Since

$$p(H\ |\vec{x}) + p(S|\vec{x}) = 1$$

Hence, if $p(S|\vec{x}) > T$, the given test instance can be classified as Spam (1) else it is classified as Ham (0)

Since, Bayes classification uses product of terms to compute the probability, it is greatly affected by zero probabilities. This is one of the reasons why the accuracy of Bayes classifier is less as compared to the SVM techniques. Depending on the nature of the data used in the training set, different types of Naïve Bayes(NB) Classifier techniques such as Gaussian NB, Bernoulli NB, Multinomial NB, Flexible NB etc.

It is a general assumption that continuous values usually follow a Gaussian distribution. Thus we can calculate the mean and variance of continuous values. Since, the spambase dataset that has been used for this project has 57 continuous attributes, I have used the Gaussian Naïve Bayes classifier. Assuming that the training set contains a continuous attribute x. In the training phase of the Gaussian Naïve Bayes classifier, we segment the data by the class and compute the mean $\mu_c$and variance σ_c^2 of the values in for each class. For a test instance with attribute value as v .Then the probability of v given that it belongs to class c can be calculated by the equation for the normal distribution characterized by the mean $\mu_c$and variance σ_c^2 by substituting the value of x as v.The formula for computing the probability of v given c is given as,

$$p(x = v|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}$$

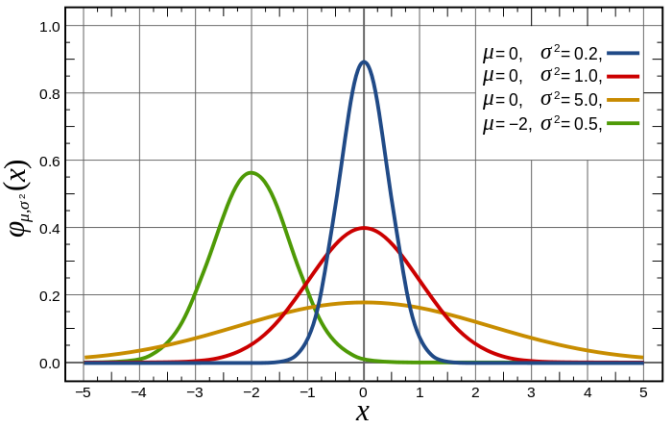

Figure 3.Normal distribution

Figure 3 shows the normal distribution for different values of mean and variance.

VI. SUPPORT VECTOR MACHINES(SVMS)

SVM is a powerful supervised machine learning technique based on the principle of structured risk minimization. SVM can handle high dimensional data when most other techniques fail due to the "curse of dimensionality"[4]. SVM is quick and uses less memory as it uses only a subset of training set instances called support vectors in its decision function [4]. SVM is versatile as kernel functions can be varied as per requirement in the decision function[15]. Radial Basis Functions(rbf) and linear functions have been used for this project.

From [4],we have,

For the problem of classifying data into two separate classes.Let an example set of training vector be:$(x_1,y_1), (x_2,y_2), \dots, (x_i,y_i)$ where $x_i \in R_n$, $y_i \in \{-1,1\}$

Let a hyperplane separating the vectors be represented by, $(w.x) + b = 1$

Such that $y_i[(w.x) + b] \geq 1$ where $i = (1,2,3, \dots, l)$

The optimal hyperplane is the one with the largest margin. The distance between the closest vectors is maximum from the hyperplane and it separates classes perfectly.

Distance between two points from the separating plane is given as,

$$d(w,b) = \frac{2}{\|w\|}$$

So in order to maximize the distance between the two points, we must minimize $\frac{1}{2}\|w\|^2$

Using lagrange multipliers ($\propto_i$) where i=(1,2,3,…,l) we get,

$$w(\alpha_i) = \sum_{i=1}^{l} \alpha_i x_i y_i$$

By wolfe's theory, the problem can be transformed to,

Max$\sum_i \alpha_i - \frac{1}{2} w(\alpha).w(\alpha)$, $\alpha_i \geq 0$,

$$\sum_i \alpha_i y_i = 0$$

The decision function with optimal hyperplane can be written as,

$$f(x) = (w_0.x) + b_0$$

The testing instances can then be classified as,

$$label(x) = sgn(f(x)) = sgn((w_0.x) + b_0)$$

The training instances satisfying $y_i[(w.x) + b] = 1$ are the support vectors. These define the boundaries of the class.

Margin band represents the region between the hyperplane through the support vectors which form the boundary of classes. For linearly non-separable training data, slack variables can be used to solve the problem.

$$Minimize \left(\frac{1}{2}\|w\|^2 + C\sum_i \xi_i\right)$$

Where $y_i[(w.x) + b] \geq 1 - \zeta_i, \zeta_i \geq 0$

The problem can be reduced to equation( ) by Wolfe's theory. Different kernel functions can be used to calculate the inner product. They are of the form $K(x_i , x_j) = (\varphi(x_i),\varphi(x_j))$ where high dimensional training vectors are mapped to φ .

The Radial Basis function is given by,

$$K(x_i, x_j) = \exp(-\gamma\|x_i - x_j\|^2)$$

SVC and NuSVC use a multiclass one to one approach for classification. If there are n classes for classification, it makes n (n-1)/2 classifiers and each classifier trains data from the two classes. For Linear SVM, which uses one vs rest approach, for data having two classes, only a single model is trained. Thus, we observe short simulation time for Linear SVM as compared to other kernel functions. C and γ are important parameters when training SVM with RBF [15]. C deals with the smoothness of the decision function. Lower the C smoother the decision surface .Higher C aims at classifying the training instances correctly and does not fit well for more generalized cases[15]. γ determines the influence of each instance in the training set. Larger the value of γ , more closer samples will be affected[15].Nu SVC is similar to SVC with C, which is often called the regularization parameter, but instead it uses value parameter v instead of C. v determines the total number of errors while training and the number of support vectors. It is mathematically equivalent to standard SVC but provides an upper bound on the fraction of errors while training and limits the fraction of support vectors for the given training set.

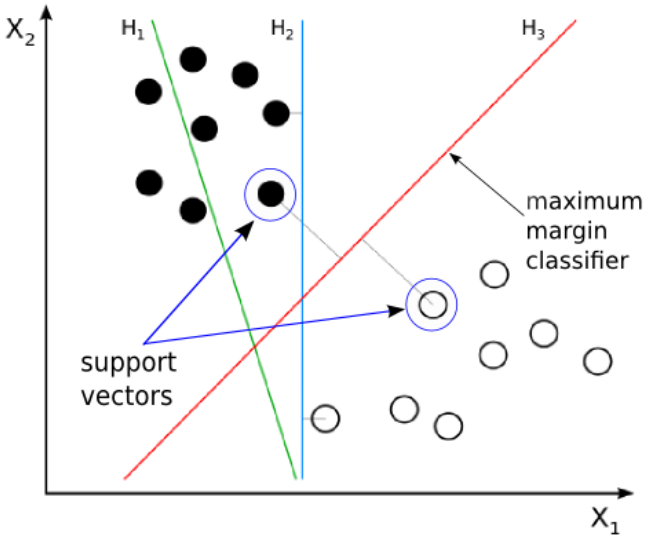

Figure 4.Hyperplanes for Linear SVM

Figure 4 shows three hyperplanes H1,H2 and H3.H1 is not a valid hyperplane as it does not divide the data points of the two classes in different regions.H2 and H3 are both valid. H3 is the maximum margin classifier and it is placed such that the highlighted such that the closest points from the plane are at the

farthest distance from each other. The highlighted points which lie on the margin band are the support vectors which are used for classification. This diagram depicts a Linear SVM classifier. Figure 5 was generated by using two attributes –word frequencies of 'make' and 'address' to classify the testing instances. The figure shows the contours corresponding to different kernel functions used for SVM classification.

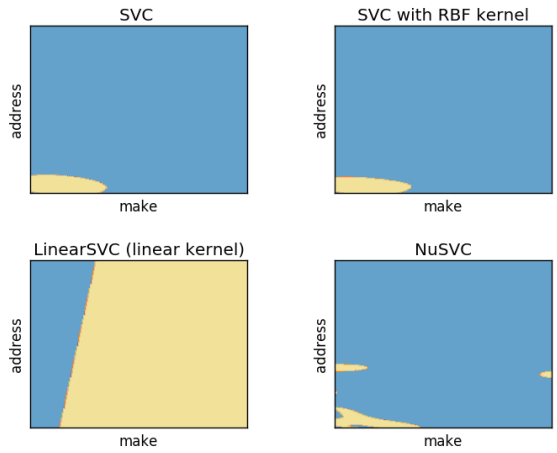

Figure 5.Contours for SVC techniques

Figure 6 shows the training points on previously plotted contour regions.

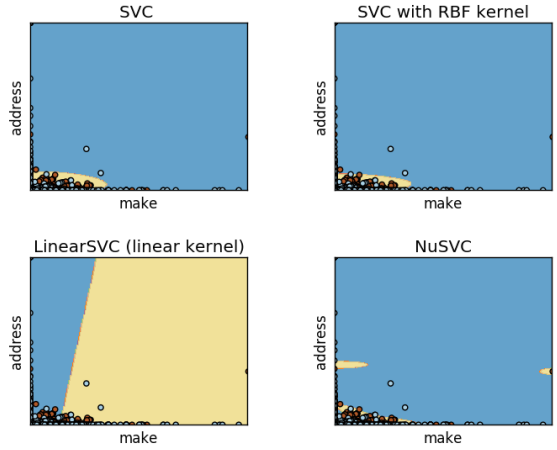

Figure 6.Training points along with the contour plots for SVM techniques

VII. IMPLEMENTATION

A. *Dataset*

The Spambase dataset was taken from the UCI Machine Learning Repository. The spambase dataset is a pre-processed dataset of 4601 emails .It has 57 attributes and the last attribute class attribute which is '1' for spam and '0'. Most of the attributes include the percentage of word frequencies of the words appearing in the email, and some run length attributes. spambase.names contains information about each of the attributes. spambase.data contains data in software independent format. The dataset contains 39.4% spam instances. Further information about the attributes can be found from [13].

TABLE 2: Characteristics of Spambase Dataset

Data Set Characteristics:	Multivariate	**Number of Instances:**	4601
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	57
Associated Tasks:	Classification	**Missing Values?**	Yes

B. *Libraries and Code Description*

The K-NN algorithm has been implemented without any library packages in Python. The Naïve Bayes classifier and SVM techniques have been implemented using the sklearn (scikit-learn) package for Python[15]. Implementation of each technique involves the following steps:

1) Loading the dataset.
2) Splitting the dataset into training and test datasets.
3) Record the start time.
4) Fitting the training instances to the label according to the Machine learning technique.
5) Finding the prediction using the testing instances.
6) Accumulating all predictions for test instances.
7) Record the end time.
8) Calculate the simulation time=end time-start time.
9) Finding the accuracy by comparing the predictions with the labels available to us from the test set derived from the dataset.

For experiments using k –fold cross validation:

1) Loading the dataset
2) Record the start time
3) Dividing the dataset into k sets using k-fold cross validation method which removes one subset each time as the testing set and uses the other subsets as the training set [2].
4) Conduct steps (4) to (6) k times with each subset as the test set once while the others act as training sets.
5) Find the average value of each of the probabilities obtained for predictions to form a final prediction list.
6) Record the end time.
7) Compute the accuracy and simulation time.

TABLE 3: Parameters for SVC

SVC	SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape=None, degree=3, gamma='auto', kernel='rbf', max_iter=-1, probability=False, random_state=None, shrinking=True, tol=0.001, verbose=False)
Linear SVC	LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True, intercept_scaling=1, loss='squared_hinge', max_iter=1000, multi_class='ovr',

	penalty='l2', random_state=None, tol=0.0001, verbose=0)
Nu SVC	NuSVC(cache_size=200, class_weight=None, coef0=0.0,decision_function_shape=None, degree=3, gamma='auto', kernel='rbf', max_iter=-1, nu=0.5, probability=False, random_state=None, shrinking=True, tol=0.001, verbose=False)

For K-NN, Euclidean distance is used as the distance metric. The Naïve Bayes classification has been performed using GaussianNB model from the sklearn package. Table 3 shown above states the parameters used for SVC, Linear SVC and NuSVC.

VIII. Analysis Of Experimental Results

The results obtained are depicted in charts as per the experiments performed in Python on the spambase dataset.

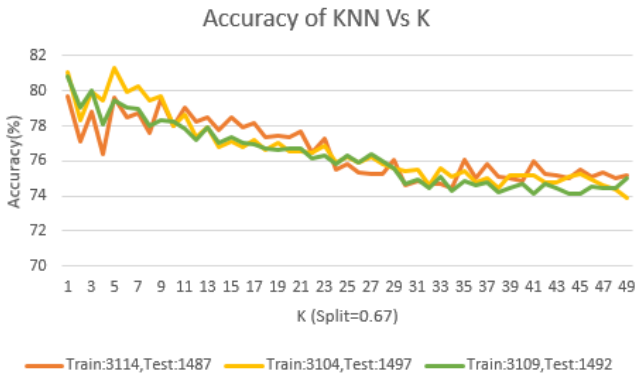

Figure 7.Accuracy of KNN Vs K

Figure 7 shows the accuracy of KNN for different values of k for three runs of the algorithm with a split ratio of 0.67. We observe that in all three cases, that the accuracy of the algorithm is the high for small values of k with a peak accuracy of 81.29% attained at k=5. The accuracy is not affected much by larger values of k or rather remains the same.

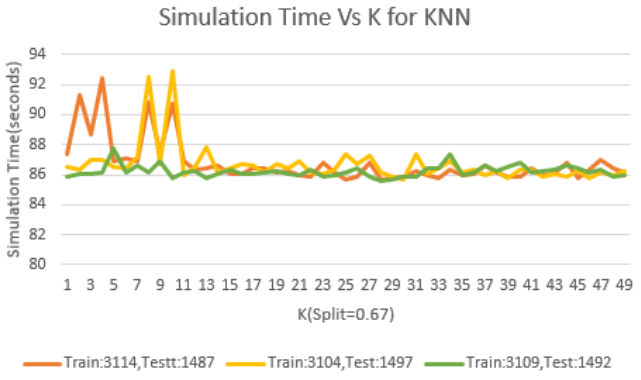

Figure 8.Simulation Time Vs K

Figure 8 depicts the simulation time of KNN. The simulation time does not vary much with the value of k as the algorithm calculates the distance of testing instance from all instances in the training set. Depending on k, k values would be extracted from the distances list .Since accessing a value at a position in a list is an O(1) operation, for all almost all values of K, the simulation time nearly the same spanning 86 to 87 seconds. The computation time of KNN is high as a lot of distance calculations need to be performed in the testing phase for each individual instance.

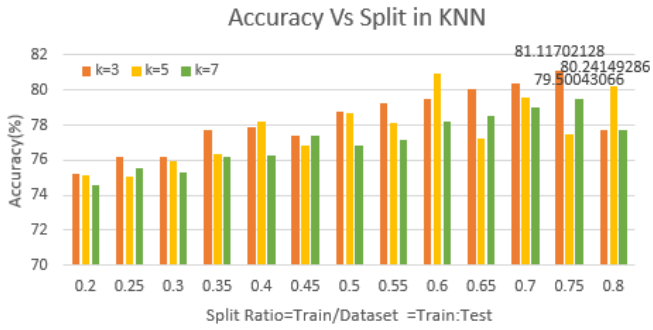

Figure 9.Accuracy Vs Split for KNN

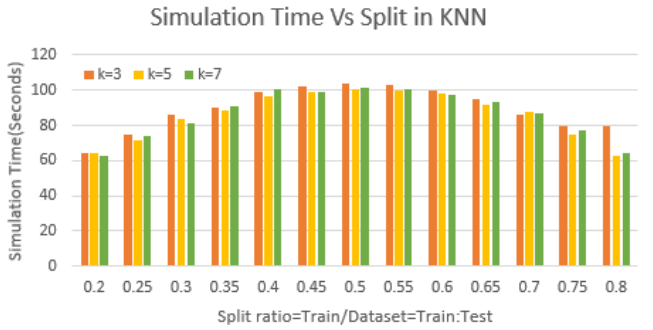

Figure 10.Simulation Time Vs Split in KNN

Figure 9 and figure 10 show the plots of accuracy and simulation time verses split ratio for different values of k. Split ratio is the ratio of the training set to the total dataset. We, observe that the accuracy obtained is the highest when the split ratio is in the range of 0.6 to 0.75 with the highest accuracy of 81.117 %. It has been observed that the simulation times follow a Gaussian distribution. We observe that the simulation times for split ratio of 0.7 is nearly the same for all three values of k and is decreasing with increase in the split ratio. Since, 0.65 to 0.75 fall under the criteria of reaching a tradeoff between accuracy and simulation time. This justifies choosing the split ratio as 0.67 for the experiments performed.

Figure 11 shows the distribution of accuracy of Gaussian Naïve Bayes classifier vs the number of sets used for cross validation. The measure of cross validation starts from 3 as having 2 equal sets implies having one set as the training set and other as the testing set which would definitely not yield greater accuracy. The accuracy of Naïve Bayes for our dataset ranges from 81.5% to 82.5%.The simulation time for Gaussian Naïve Bayes

increases linearly with the increase in the number of sets used for cross validation as shown in figure 12. However, the time taken for simulation of Naïve Bayes is only a fraction of a second which is very small as compared to on average 80 seconds required for the KNN algorithm (without any cross validation).

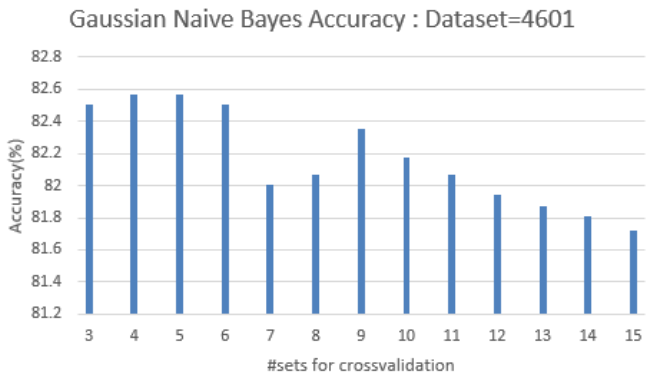

Figure 11.Accuracy of Gaussian Naive Bayes Classifier

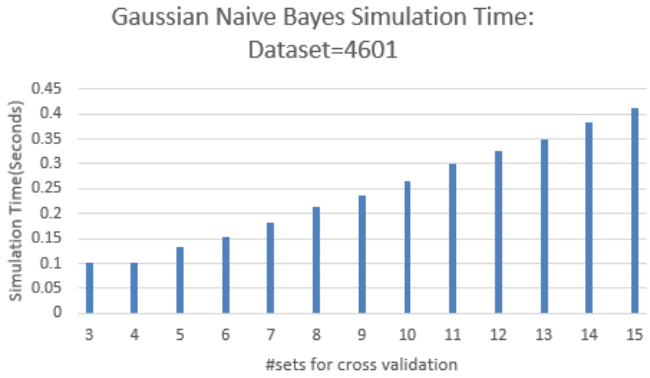

Figure 12.Simulation Time for Gaussian Naive Bayes Classifier

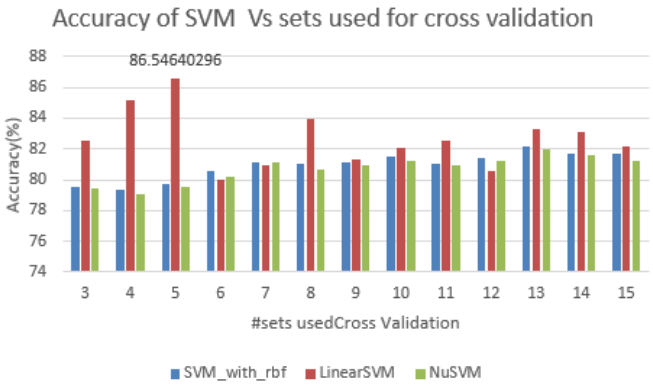

Figure 13.Accuracy of SVM techniques

The figure 13 above depicts the plot of accuracy obtained using various SVM techniques verses the number of sets used for cross validation. As a general observation, for the spambase dataset, linear SVM tends to perform better in most cases regardless of the number of sets used for cross validation. An accuracy high of 86.54% was obtained from linear SVM with 5 sets used for cross validation. Cross validation gives us a better idea of choosing the filtering technique as it takes into consideration different instances in the training set every time. However, large number of sets used for cross validation would increase the simulation time significantly. The performance of the filtering techniques with respect to simulation time and cross validation sets is depicted in figure 14.

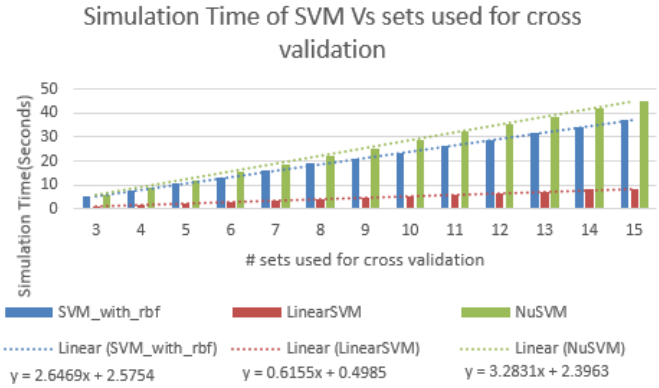

Figure 14.Simulation Time for SVM techniques

The slope for linear SVM is small ie. The simulation times do not increase very rapidly for increase in the number of sets. The simulation time of linear SVM is very less as compared to SVM with rbf filter and Nu SVM. However, from further experiments I have observed that SVM techniques are still slower as compared to Naïve Bayes algorithm but have slightly better accuracy.

After running repeated tests on the dataset, to split them into training and test sets according to a split ratio of 0.67. Several runs of the algorithm give us a better idea of the accuracy and running times of the classification algorithms.

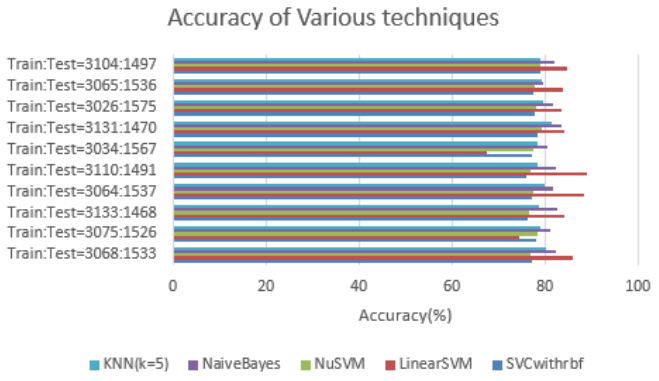

Figure 15.Comparison of Accuracy of ML techniques on different training and test sets

All 5 algorithms-Naïve Bayes, K-NN, Linear SVM, Nu SVM and SVM with the RBF were run on the same training and test sets to calculate the accuracy of predicting spam and ham correctly and their simulation time was also computed. K-NN is the slowest algorithm converging in about 90 seconds among the

other 4 implementations. Since the performance of KNN in terms of simulation time is not comparable to the other implementations, it is not included in the figure 16. From figure 16, it can be observed that the Naïve Bayes Algorithm is the fastest among the 5 implementations for the spambase dataset which has continuous attributes.

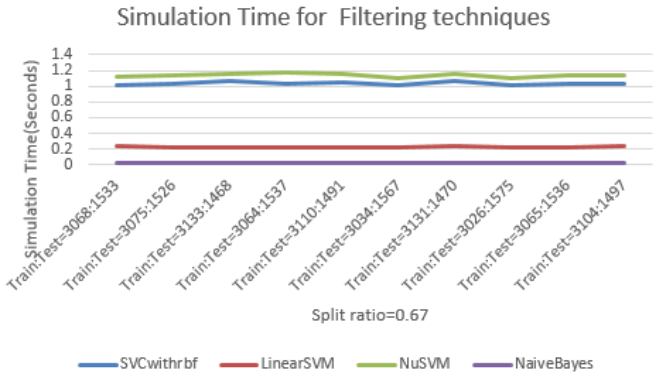

Figure 16.Comparison of Simulation Time for ML techniques

Table 4 shows the average accuracy and the average simulation time for each of the algorithms obtained after conducting the experiment 10 times with a split ratio of 0.67. Since the dataset is divided into training sets and testing sets randomly using a factor of 0.67, the elements in the training and test sets are not always the same in each run and the size of train and test set pairs is different in every run. This ensures that the tests performed are not redundant and helps predict the performance parameters better.

TABLE 4: Average Accuracy of Spam Filtering Techniques

ML Techniques	**Average Accuracy (%)**	**Average Simulation Time (seconds)**
SVC with RBF	77.49152	1.038186
Linear SVM	82.62054	0.228514
Nu SVM	77.82272	1.135457
Naïve Bayes	81.79272	0.0184
KNN(k=5)	79.41065	91.5043

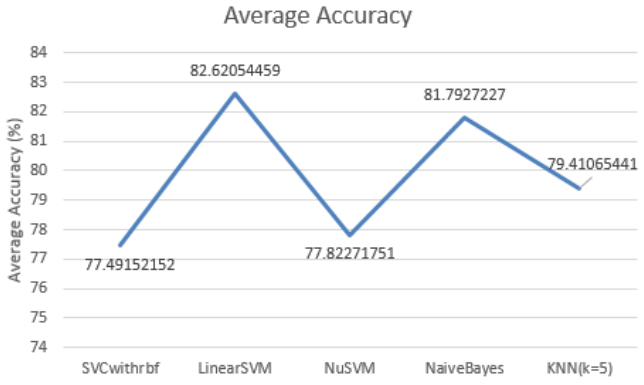

Figure 17.Average Accuracy of Email Spam Filtering Techniques

As can be seen from the accuracies and simulation times from table 4 and figures 18 and 19, Naïve Bayes algorithm filters spam or classifies data in a fraction of a second and is much faster than K-NN algorithm which takes a few minutes to perform the same classification. The highest accuracy is obtained by linear SVM for the spambase dataset while the accuracies obtained by Naïve Bayes are also comparable to Linear SVM.

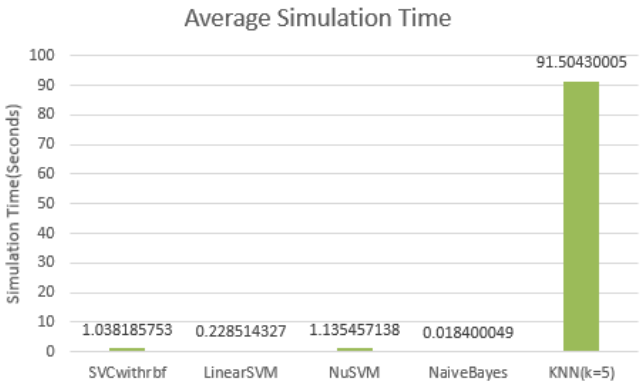

Figure 18.Average Simulation Time for Email Spam Filtering Techniques

IX. Conclusion

Among the supervised machine learning tasks that were implemented and evaluated, the Naïve Bayes classifier and Linear SVM offer superior performance in terms of both accuracy and simulation for the spambase dataset.

The Naïve Bayes classifier must be preferred over linear SVM where speed is more than the accuracy and vice versa. Naive Bayes classifier is nearly 12 times faster than Linear SVM and 60 times faster than Nu SVM and SVC with RBF. The performance of Naïve Bayes is justified with respect to continuous data attributes as per the dataset used. Naive Bayes classifier uses a simplistic statistical approach which yields results with great accuracy in less simulation time. Naïve Bayes is appropriate fit for email spam filtering.

KNN being a lazy approach which requires a lot of time should only be used for small datasets as the computational complexity goes on increasing with more data points and thus increases processing time. K-NN performs efficiently for smaller values of k than at larger values. Its simulation time is nearly 5000 times greater than the fastest Naïve Bayes algorithm for the spambase dataset.

Cross validation technique allows us to better decide the pattern classification technique used for a particular application.

X. Related Work

Email Spam filtering techniques have been explored since a long time. Paul Graham came up with the idea for a plan for Spam and believed in the idea that spam could be tackled if we could write software that could recognize spam filtering in 2002 [11].

He introduced better Bayesian Filtering in [12].Several papers were published with regards to Machine Learning techniques. Naïve Bayesian filters became popular for text categorization and different Naïve Bayes algorithms were studied and their performance was evaluated on different datasets such as Spambase and corpus in [1, 6, 7, 9].KNN and improvised faster KNN were explored in [3].An in depth study of SVM with kernel parameters can be found in [4].A comparative study of supervised machine learning algorithm like Naïve Bayes, KNN, Artificial Neural Networks(ANN)-Multilayer Perceptron, Support Vector Machines (SVM) can be found in [2,5,9,10]. As a future scope to this project, unsupervised machine learning techniques and techniques using neural networks could be used to identify spam email as has been implemented and analyzed in [8]. Spammers have become smart over the years and thus spam emails are not limited to text alone. A lot of spam email today does not contain information in the form of text but instead have images and several emoticons. A better spam email classifier should be versatile enough to deal with different forms of data and be able to classify email perfectly. Classification techniques can be used in various applications such as recommendation systems and object recognition techniques.

REFERENCES

[1] Vangelis Metsis, "*Spam Filtering with Naive Bayes -- Which Naive Bayes?"*, Vangelis Metsis Telecommunications (2006).

[2] W.A. Awad1 and S.M. ELseuofi, *"Machine Learning Methods for Spam Email Classification*", International Journal of Computer Science & Information Technology (IJCSIT), Vol 3, No 1 pp. 173 – 183, [Feb 2011].

[3] La Bouli, Yu Shiven, Lu Qin , *"An improved k Nearest Neighbors algorithm for text Categorization"* , International Conference on Neural Information Processing,Vol 2 ,pp.731-735(2002).

[4] Hsu Wei-Chih and Tsan-Ying Yu,*"E-mail Spam Filtering Using Support Vector Machines with Selection of Kernel Function Parameters"*, 2009 Fourth International Conference on Innovative Computing, Information and Control,pp.764-767.

[5] S. B. Kotsiantis, *"Supervised Machine Learning: A Review of Classification Techniques",* Informatica 31 (2007),pp. 249-268.

[6] Haiyi Zhang, Di Li*,"Naïve Bayes Text Classifier*", 2007 IEEE International Conference on Granular Computing,pp.708-711.

[7] Chuanliang Chen , Yingjie Tian ,Chunhua Zhang *, "Spam Filtering with Several Novel Bayesian Classifiers*", 19 th International Conference on Pattern Recognition,pp.1-4(2008)

[8] Tiny du Toit , Hennie Kruger *, "Filtering Spam E-Mail with Generalized Additive Neural Networks"* ,2012 Information Security for South Africa,pp.1-8.

[9] Md. Saiful Islam, Shah Mostafa Khaled, Khalid Farhan, Md. Abdur Rahman and Joy Rahman,"*Modeling Spammer Behavior: Naïve Bayes vs. Artificial Neural Networks*", 2009 International Conference on Information and Multimedia Technology,pp.52-55.

[10] Prabin Kumar Panigrahi *, "A Comparative Study of Supervised Machine Learning Techniques for Spam E-Mail Filtering*", 2012 Fourth International Conference on Computational Intelligence and Communication Networks,pp.506-512.

[11] http://www.paulgraham.com/spam.html

[12] http://www.paulgraham.com/better.html

[13] https://archive.ics.uci.edu/ml/datasets/Spambase

[14] https://www.wikiwand.com/en/

[15] http://scikit-learn.org/stable/modules/svm.html

[16] Richard O. Duda,Peter E.Hart,David G.Stork,*"Pattern Classification, Second Edition"*.

[17] Cestnik, B., Kononenko, I., Bratko, I., *(1987)"A knowledge elicitation tool forsophisticated users",*Proceedings of the Second European Working Session on Learning, pp. 31-45.

[18] Wettschereck, D., Aha, D. W. & Mohri, T.(1997). *"A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms"* ,Artificial Intelligence Review10:1–37.

[19] Cestnik, B. (1990), "*Estimating probabilities: A crucial task in machine learning*" , Proceedings of the European Conference on Artificial Intelligence,pp. 147-149.