Literature Review of SMS Spam Detection using various Machine Learning Classifiers

Abstract:

Under short messaging service (SMS) spam is understood the unsolicited or undesired messages received on mobile phones. With technological advancements and increment in content based advertisement, the use of Short Message Service (SMS) on phones has increased to such a significant level that devices are sometimes flooded with a number of spam SMS. These spam messages can lead to loss of private data as well. There are many content-based machine learning techniques which have proven to be effective in filtering spam emails. Modern day researchers have used some stylistic features of text messages to classify them to be ham or spam. SMS spam detection can be greatly influenced by the presence of known words, phrases, abbreviations and idioms. This paper aims to compare different classifying techniques on different datasets collected from previous research works, and evaluate them on the basis of their accuracy scores. The comparison has been performed between various machine learning techniques.

Introduction:

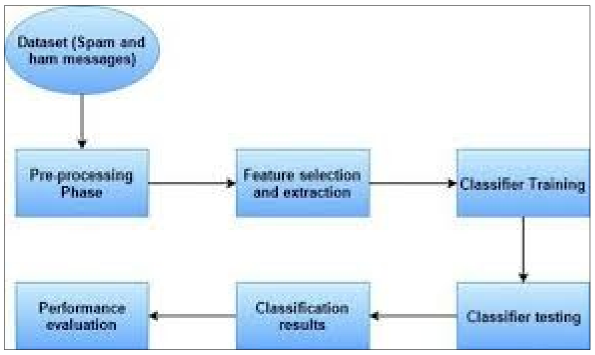
The increasing mobile phones become one of the attached companions for many individuals. Short Message Service (SMS) is a technique of sending short text messages from one device to another. Spam refers to the word which is generally used for message or information that is junk or unsolicited. So a spam SMS can be defined as any junk message delivered to a mobile device through text messaging. Whether spam is in the form of email or SMS, danger is equal. Spam may result in leaking personal information, invasion of privacy or accessing unauthorized data from mobile devices. In this era of Smartphone devices, users now have personal and confidential information such as contact lists, credit card numbers, photographs, passwords and much more stored in their smartphones, making them prone to cyber attacks through spam SMS. This enables hackers involved in unethical activities to access smartphone data without the knowledge of end-user, thus compromising the user's privacy. This could also result in pecuniary as well as functional loss. Spam messages seem to be increasing and result not only in annoyance for users but critical data loss for some users. Apart from this, SMS spam can also act as a driving force for malwares and key-loggers. The unanimous spreading of this kind of problem and less effective security control measures has inspired many researchers in the development of a set of techniques to help prevent it in a varied and efficient way. The main motive is to handle security issues in terms of protection of privacy, solidarity and accessibility. Many users are still unaware about protection mechanisms, thereby, making their mobiles prone to cyber attacks. The Government of India has set up NCPR (National Customer Preference Register) registry, which to some extent has reduced junk calls, but does not filter spam SMS. Although various datasets are available to test email spam detection algorithms, the datasets to train and test techniques for SMS spam detection are still limited and small sized. Moreover, unlike emails, the length of text messages is short, that is, less statistically-differentiable information, due to which the availability of number of features required to detect spam SMS are less. Text messages are highly influenced by the presence of informal languages like regional words, idioms, phrases and abbreviations due to which email spam filtering methods fail in the case of SMS. There are various approaches proposed in previous works for detecting email and SMS spam. In this paper, our aim is to train, test and compare different traditional machine learning classifiers and deep learning techniques on different datasets. On the basis of their accuracies, precision and recall, various considered classifiers are evaluated. A comparison and analysis of all the techniques has been then carried out.

Related Works:

The most common formulation of spam filtering is the task of classification. For a piece of text, the aim is to predict whether or not it is spam. However, past work varies in terms of what kinds of datasets are used for training and testing. For example, Delany et al. created their own dataset consisting of ham messages collected from websites like GrumbleText, Who Calls Me and spam messages from SMS Spam Collection. They analysed the spam messages using content based clustering and nearly identified nine to ten clearlydefined clusters. The approach used by Kim et al. proposed a method that was based on calculation of frequency which measures lightness and quickness of filtering methods. They considered Naive Bayes, J-48 and Logistic algorithms for their research. They proved that their proposed technique had a similar capability as those of others, even though it used a simple formula, giving them an advantage over other techniques in particular sense. Mahmoud and Mahfouz developed an Artificial Immune System (AIS) for classification of SMS. The system used a set of features as an input spam filter. It was then used to categorize the text messages with the help of a trained dataset which included spam words, phone numbers etc. The results of this experiment showed better accuracy and convergence speed than the Naive Bayesian algorithm in classifying messages either as spam or non-spam. Researchers like Chan et al. and Najadat et al. used SVM classifier in their research, and further recruited more accurate results compared to other techniques. Although neural networks have not been widely used for classification of SMS, they have been significantly used to detect spam emails by different researchers. The paper by Yang and Elfayoumy compares feed-forward backpropagation neural network Bayesian classifiers to evaluate their effectiveness in classifying spam email. High accuracy and sensitivity was shown by the neural network algorithm making it a good competition for traditional classifiers, but it took a considerably longer training time. On the other hand, it was observed that the task to construct Bayesian classifiers was very easy, but they were not as accurate. Clark et al. showed in terms of classification performance how their back propagation based system outperformed many other algorithms. The effects of various different techniques like feature selection, weighting and normalization were also explored. The best results in email spam filtering were produced by normalization at mailbox level with Frequency and tf-idf weighting. Convolutional Neural Networks have traditionally been used for image classification problems. The paper by Zhang et al. goes against this notion by comparing character-level convolutional networks with a number of traditional and deep learning models using very big datasets in their empirical study. The results so obtain encourage the application of such neural networks for problems of text classification and a broader range of natural language processing. Researchers like Murynets and Jover have gone even further to analyse the SMS traffic. Messaging behaviour similar to a spammer was observed for some networked applications because few Machine to Machine systems transmit a huge count of texts per day. It was suggested to consider such systems when designing SMS spam detection models.

Methodology:

We begin with conversion, preprocessing and splitting of datasets as per the requirement of considered algorithms. The various models are then trained and tested, followed by evaluation and comparison.

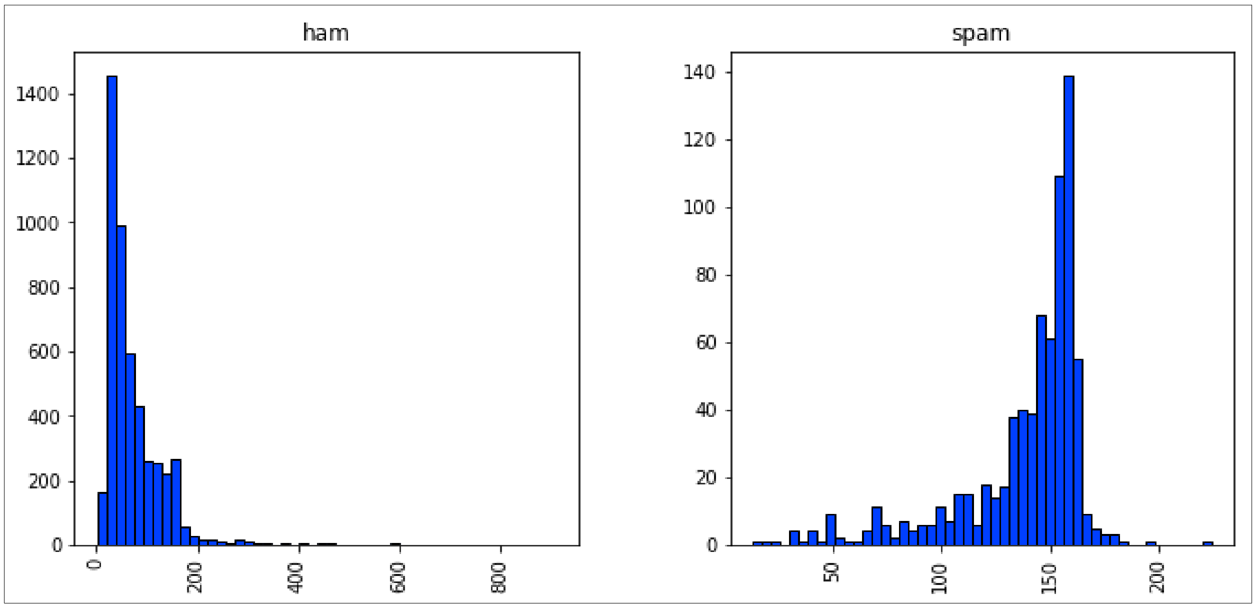


Flow Diagram

A. Dataset Description:The SMS Spam Collection is a set of SMS tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged acording being ham (legitimate) or spam. The files contain one message per line. Each line is composed by two columns: v1 contains the label (ham or spam) and v2 contains the raw text. This corpus has been collected from free or free for research sources at the Internet:

- A collection of 425 SMS spam messages was manually extracted from the Grumbletext Web site. This is a UK forum in which cell phone users make public claims about SMS spam messages, most of them without reporting the very spam message received. The identification of the text of spam messages in the claims is a very hard and time-consuming task, and it involved carefully scanning hundreds of web pages.

- A subset of 3,375 SMS randomly chosen ham messages of the NUS SMS Corpus (NSC), which is a dataset of about 10,000 legitimate messages collected for research at the Department of Computer Science at the National University of Singapore. The messages largely originate from Singaporeans and mostly from students attending the University. These messages were collected from volunteers who were made aware that their contributions were going to be made publicly available.
- A list of 450 SMS ham messages collected from Caroline Tag's PhD Thesis.
- Finally, we have incorporated the SMS Spam Corpus v.0.1 Big. It has 1,002 SMS ham messages and 322 spam messages and it is public available.



B. Preparing Readable Datasets: The datasets have been prepared as comma-separated values (CSV) files. These files contain one text message per line. Each line has two columns - v1 is the label (ham or spam) and v2 is the raw text. The Spam SMS Dataset 2011-12 was procured as a zipped file with numerous text files containing a message each. The name of the file indicated whether the message it contained was legitimate or spam. A script was prepared and executed to accurately label the messages and unify them into a single CSV file. For any classifier to be able to use this data, we need to do some preprocessing.

C. Data Preprocessing: Different preprocessing approaches have been applied to different classifiers based on their requirement of input data. Following is a brief description of these approaches.

1. Using CountVectorizer— CountVectorizeris a great tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. This is

helpful when we have multiple such texts, and we wish to convert each word in each text into vectors (for using in further text analysis). CountVectorizer creates a matrix in which each unique word is represented by a column of the matrix, and each text sample from the document is a row in the matrix. The value of each cell is nothing but the count of the word in that particular text sample.

2. Using Tokenizer: When working with text, it is always good to start with splitting the text into words. Words are known as tokens. Tokenization is the process of splitting text into words or tokens. It is used to turn texts into sequences. A sequence is a list of word indexes where the word of rank i in the dataset (starting at 1) has index i. Text can be split into a list of words. This function by default splits words by space, converts text to lowercase and filters out punctuation. Tokenization is restricted to the top most common words in the corpus. After applying either of the approaches on our datasets for different classifiers, the representation of first column was changed. Ham and spam were transformed to values 1 and 0 using sklearn's LabelEncoder.

D. Different Classifiers: Different preprocessing approaches have been applied to different classifiers based on their requirement of input data.Following is a brief description of these approaches.

1) Support Vector Machine (SVM): SVM is a discriminative classifier which is widely used for classification task. The algorithm plots each data item as a point in n-dimensional space assuming the value of each feature as the value of a particular coordinate. It then forms a line that splits the whole data into two differently classified groups of data. The closest points in the two groups will be the farthest away from this line.
2) Naive Bayes (NB): This classification technique is based on Bayes’ theorem, assuming independence between predictors. The Bayesian classifier assumes that a particular feature in a class is not related to the presence of any other feature. Even if they do depend upon each other or upon the existence of the feature, the Bayesian classifier will consider all of the desired properties to independently contribute to the probability. The classifier holds well when the desired input’s dimensionality is high. It is regarded as simple and sturdy. An advanced version of NB is Multinomial Naive Bayes (MNB). The main improvement is the independence between document length and class. It involves multinomial distribution which works well for countable type of data such as the words in a document or text. In simple terms, NB classifier involves conditional independence of each of the features in the model, whereas MNB classifier is a special case of a NB classifier which uses a multinomial distribution for each feature. an input layer, some hidden layers and an output layer, but it is not fully connected. Some layers are convolutional, that use a mathematical model to pass on the results to layers ahead in the network.
3) Decision Tree (DT): DT is a supervised learning algorithm which is normally preferred for classification tasks. The algorithm works well for both types of variables i.e., categorical and continuous. It starts with splitting the population into multiple homogeneous sets which is done on the basis of most significant attributes or independent variables. DT is non-parametric and hence the need for checking outlier existence or data linearity separation is not required.

4) Logistic Regression (LR): It is considered as the go-to method for classification involving binary results. It is mainly used in estimating discrete values which are based on set of variables which are independent. In more relative terms, LR outputs the probability of an event by fitting it into a logistic function which helps in prediction. The logistic function which is mostly used is sigmoid.
5) Random Forest (RF): It is a term used for an ensemble of decision trees. The Random Forest classifier is a ensemble learning method which involves collection of decision trees. Voting is done to classify a new object which is performed by each tree i.e., the trees mark their votes for that class. The class having most number of votes decides the classification label.
6) AdaBoost: AdaBoost or Adaptive Boosting is a meta- machine learning algorithm, used to increase the performance of a classifier by simply using the weak classifiers to combine them into a strong one. The final output of the boosted classifier depends upon the weighted sum of the output of all the weak classifiers. A drawback of this technique is that although it predicts more accurately, it takes more time for building the boosted model.
7) k-nearest neighbors: K-nearest neighbors (KNN) algorithm is a type of supervised ML algorithm which can be used for both classification as well as regression predictive problems. However, it is mainly used for classification predictive problems in industry. The following two properties would define KNN well :− Lazy learning algorithm − KNN is a lazy learning algorithm because it does not have a specialized training phase and uses all the data for training while classification and Non-parametric learning algorithm − KNN is also a non-parametric learning algorithm because it doesn't assume anything about the underlying data.
8) Bagging Classifier: A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. Such a meta-estimator can typically be used as a way to reduce the variance of a black-box estimator (e.g., a decision tree), by introducing randomization into its construction procedure and then making an ensemble out of it.Each base classifier is trained in parallel with a training set which is generated by randomly drawing, with replacement, N examples(or data) from the original training dataset – where N is the size of the original training set. Training set for each of the base classifiers is independent of each other. Many of the original data may be repeated in the resulting training set while others may be left out.
9) Extra Tree Classifier: Extremely Randomized Trees Classifier(Extra Trees Classifier) is a type of ensemble learning technique which aggregates the results of multiple de-correlated decision trees collected in a "forest" to output it's classification result. In concept, it is very similar to a Random Forest Classifier and only differs from it in the manner of construction of the decision trees in the forest. Each Decision Tree in the Extra Trees Forest is constructed from the original training sample. Then, at each test node, Each tree is provided with a random sample of k features from the feature-set from which each decision tree must select the best feature to split the data based on some mathematical criteria

(typically the Gini Index). This random sample of features leads to the creation of multiple de-correlated decision trees.To perform feature selection using the above forest structure, during the construction of the forest, for each feature, the normalized total reduction in the mathematical criteria used in the decision of feature of split (Gini Index if the Gini Index is used in the construction of the forest) is computed. This value is called the Gini Importance of the feature. To perform feature selection, each feature is ordered in descending order according to the Gini Importance of each feature and the user selects the top k features according to his/her choice.

E. Evaluating our model:

Accuracy score measures how often the classifier makes the correct prediction. It's the ratio of the number of correct predictions to the total number of predictions (the number of test data points).

Precision score tells us what proportion of messages we classified as spam, actually were spam. It is a ratio of true positives(words classified as spam, and which are actually spam) to all positives(all words classified as spam, irrespective of whether that was the correct classification), in other words it is the ratio of [True Positives/(True Positives + False Positives)]

Recall score(sensitivity) tells us what proportion of messages that actually were spam were classified by us as spam. It is a ratio of true positives(words classified as spam, and which are actually spam) to all the words that were actually spam, in other words it is the ratio of
[True Positives/(True Positives + False Negatives)]

For classification problems that are skewed in their classification distributions like in our case, for example if we had a 100 text messages and only 2 were spam and the rest 98 weren't, accuracy by itself is not a very good metric. We could classify 90 messages as not spam(including the 2 that were spam but we classify them as not spam, hence they would be false negatives) and 10 as spam(all 10 false positives) and still get a reasonably good accuracy score. For such cases, precision and recall come in very handy. These two metrics can be combined to get the F1 score or F-measure, which is weighted average of the precision and recall scores. This score can range from 0 to 1, with 1 being the best possible F1 score.

We will be using all 4 metrics to make sure our model does well. For all 4 metrics whose values can range from 0 to 1, having a score as close to 1 as possible is a good indicator of how well our model is doing.

Experimental Results:

A. For Testing Dataset 1:

Classifiers	**Accuracy Score**	**Precision Score**	**Recall Score**	**F1 Score**
SVM	0.839196	0.913808	0.900247	0.906977
K Neighbors	0.870782	0.870782	1.0	0.930929
Decision Tree	0.962670	0.981743	0.975268	0.978495
Bagging	0.970567	0.986711	0.979390	0.983037
AdaBoost	0.982053	0.984502	0.995054	0.989750
Random Forest	0.985642	0.983779	1.0	0.991823
Naive Bayes	0.982771	0.990917	0.989283	0.990099
Extra Trees	0.978464	0.977401	0.998351	0.987765
Logistic Regression	**0.986360**	0.986156	**0.998351**	**0.992216**

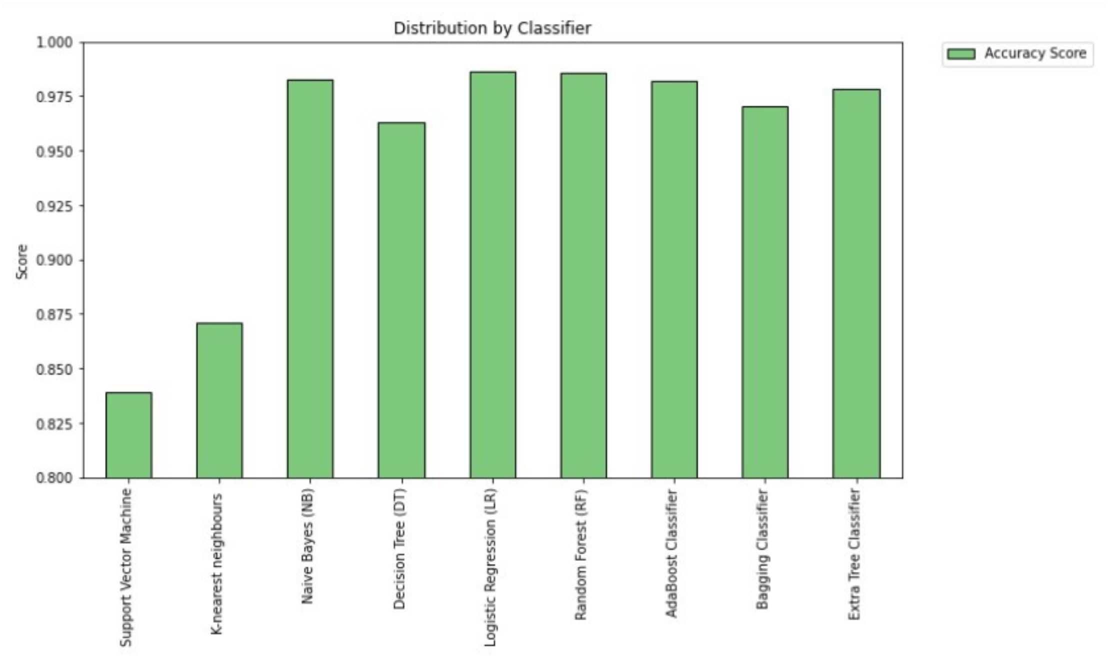
B. For Testing Dataset 2:

Classifiers	**Accuracy Score**	**Precision Score**	**Recall Score**	**F1 Score**
SVM	0.870782	0.870782	1.0	0.930928
K Neighbors	0.921751	0.917549	1.0	0.957001
Decision Tree	0.961952	0.978547	0.97774	0.978144
Bagging	0.971284	0.989157	0.977741	0.983416
AdaBoost	0.976310	0.979674	0.993404	0.986492
Random Forest	0.982770	0.981376	0.999175	0.990196
Naive Bayes	0.985642	0.990139	0.993404	0.991769
Extra Trees	0.984206	0.986917	0.995053	0.990968
Logistic Regression	**0.987796**	0.987765	**0.998351**	**0.993029**

To compare the performance of algorithms in the experiment, this paper employs the performance evaluation measures such as accuracy score, precision score, recall score and f1 score. Both table shows the performance of algorithms on two different testing datasets.

It is notable that CountVectorizer+LR(logistic regression) achieves the best performance and outperforms the other evaluated algorithms in this experiment in terms of accuracy score, precision score, recall score and f-measure. It accurately classifies 98.77% while it achieves 0.9877 in precision while f1 score is 0.9930. The second-best algorithm in terms of accuracy percentage is CountVectorizer+Naive Bayes's with 98.56%, and the result of precision is 0.99 with the propose method. The lowest performance is determined using the CountVectorizer+SVM with the result accuracy is 87.07%, precision 0.87, recall score is 1 and f1 score is 0.93. The reason behind this result might be due to the fact that SVM cannot handle imbalanced dataset very well.
The imbalanced data might cause the performance loss in several ways such as the imbalances ratio of positive and negative support vectors or the position of positive points being far from the ideal boundary. Nonetheless, SVM still manages to get more than 75% for overall performance.

CONCLUSION AND FUTURE SCOPE:



The SMS spam message problem is plaguing almost every country and keeps increasing without a sign of slowing down as the number of mobile users increase in addition to cheap rates of SMS services. Therefore, this paper presents the spam filtering technique using various machine learning algorithms. Based on the experiment, CountVectorizer with Multinomial Naïve Bayes is good but CountVectorizer with Logistic Regression classification algorithm outperforms good compare to other algorithms in terms of accuracy percentage. However, it is not enough to evaluate the performance based on the accuracy alone since the dataset is imbalanced; therefore, the precision, recall and f-measure of the algorithms must also be observed. After some examinations, Logistic Regression algorithm still manages to provide good precision and f-measure with 0.98 of precision while 0.99 for f-measure. Different algorithms will provide different performances and results based on the features used. Significant results have been obtained from this work, corresponding to which this research can be taken to real world application level for detection of spam SMS.