Spam Detection

Hitesh Bhandari, Samarth Raina, Kushiluv Jangu, Rohit Aggarwal
Deparment of Computer Science
Indraprastha Institute of Information Technology, Delhi
(hitesh20436, samarth20465, kushiluv20076, rohit19474)@iiitd.ac.in

Abstract

It is estimated that around 68 % (predicted to increase even more) of all SMSs sent to Indian mobile subscribers are unwanted. The reason for such a vast number is that spammers have become more sophisticated and creative with their ideas to get their messages into people's inboxes. This upsurge in the number of unsolicited, unwanted and virus-infected SMS called spam has created an intense need to develop a robust and dependable program. We thus aim to use advanced machine learning algorithms that would help us analyse these messages and create a classifier which would make the whole experience more user-friendly. The dataset used has 5 features and 5572 entries/rows. Different machine-learning models have been employed to classify messages into spam or ham(good mail). Among these models, multinomial naive Bayes and Random Forests have produced the best results, with an accuracy of 97.09% and 97.48%, respectively. Lastly, to further improve the accuracy and efficacy of the whole model, we've also implemented stemming and vectorization. The code and trained model is available on the github repo which you can access by clicking here.

1. Introduction

Spam messages are becoming more prevalent across all electronic media due to increased internet and mobile phone usage. These messages have been a significant issue for mobile service communication services. The simplicity of sending spam messages to numerous individuals has contributed to the growth of spam communications. The number of SMS spam is increasing due to several factors. Since neither the SMS application (access layer) nor the telecom operator (service provider layer) has properly installed SMS spam filters, spam messages can be routed to a user's mobile phone unfiltered.[1] Also, SMS does not require the user to be connected to the internet, therefore spam can reach the user's mobile phone and is thus different from e-mail and other online communication services. Lastly, the ease of sending bulk SMS to several users at once is another factor. Spam messages can distract a mobile phone user from reading critical messages by piling up their inbox and pushing them to the top of the inbox. Users may find it annoying to receive an SMS spam alert while hoping for other important messages. Because of this, sending spam messages can potentially reduce the effectiveness of SMS as a communication tool. Spam messages are often delivered to many users and are irrelevant or uninvited. Spam detection has been developed to address SMS spam issues. Typically, binary classification is employed as a spam detection method. A spam detector classifies every message into two types: spam and ham(or good mail). This classification helps in creating a user-friendly environment free of unwanted messages, thereby saving a lot of time and hassle. Spam could be of various types; some major ones are unnecessary advertisements and different types of fraud. Our dataset contains 5 features and 5574 data entries . We have analysed different types of machine learning classifiers and used metrics such as accuracy and precision to program our spam detector.

2. Literature Survey

Dewi et al. [3] lay a solid foundation for ground-up spam detection. Working on SMS in Malay, there aren't any pretrained models accurate enough for industrial use. Past researches on SMS spam detection only classified SMS into two categories, spam and not spam. The binary classification of SMS spam prevents the user from seeing the spam messages that they do not really hate, e.g. an advertisement from their favorite product. They propose a multiclass classification of SMS into: regular, info, ads, and fraud. They use content-based (top-N unigram) as well as non-content-based features.

The result shows that the best accuracy is achieved by logistic regression that is 97.5 % accuracy with configuration of normalization preprocess and 4096 top-N unigram features.

Kawintiranon et al [1] in contrast use the state-of-the-art techniques to explore nuances in detecting spam on Twitter, which is a dynamic public messaging platform. The pa-

per makes distinction between traditional spam and context-specific spam. What qualifies as spam varies across domains. For example if you're viewing a Parenting tweet thread, ads on diapers aren't unwelcome by the users. But an ad for a music speaker is contextually a spam in the Parenting thread. Again, the same ad isn't if you're viewing Top 100 Billboard.

Different domains/themes of conversation on Twitter have different levels of spam, leading to different class balances in training data sets. This affects the cross-domain generalizability and transferability of spam detection models

3. Dataset

3.1. Dataset Details

We have used the SMS Spam Collection Dataset from the UCI Machine Learning Repository available at https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset. The SMS Spam Collection is a public set of SMS labeled messages that have been collected for mobile phone spam research. It contains one set of SMS messages in English of 5,574 messages, tagged as being ham (legitimate) or spam, which are the two types of classification of our dataset. Out of these, 87% messages are labelled as ham and around 13% as spam. This shows that data is highly imbalanced. The dataset consist of 5572 rows and 5 columns. These 5 columns being v1 (has the labels : ham or spam), v2 (has the text) and three unnamed columns and these unnamed columns have a very low count of non null values.

3.2. Dataset Visualisation

To depict the relation of the columns among each other we plotted pair plots (Figure 1). The graph between the number of characters and number of sentences is roughly linear, with some outliers. We plotted the correlation heat map (Figure 2) of the columns and selected the highly correlated ones. It can be seen that with the increase in number of characters the tendency of a message to be a spam increases. For our modelling we'll be taking the number of characters feature as it holds the highest correlation (0.38).

3.3. Dataset Cleaning

The practise of repairing or deleting incorrect, corrupted, improperly formatted, duplicate, or incomplete data from a dataset is known as data cleaning. There are numerous ways for data to be duplicated or mislabeled when merging multiple data sources. We also identified the need for data cleaning for our dataset. 3 out of the 5 columns had a very low count of non null values (50, 12, 6 out of 5572 values), its better to get rid of these so that it doesn't impose hinderance in our analysis. For data cleaning, we dropped columns having majority null values and changed some columns names in order to provide more clarity. We renamed v1 (which has the labels hams and spam) and v2 (which has the sms text) columns as 'target' and 'text'. Next, we replaced string values in the 'target' column with integer values: ham as 0 and spam as 1. At last, we removed the duplicates in our dataset.

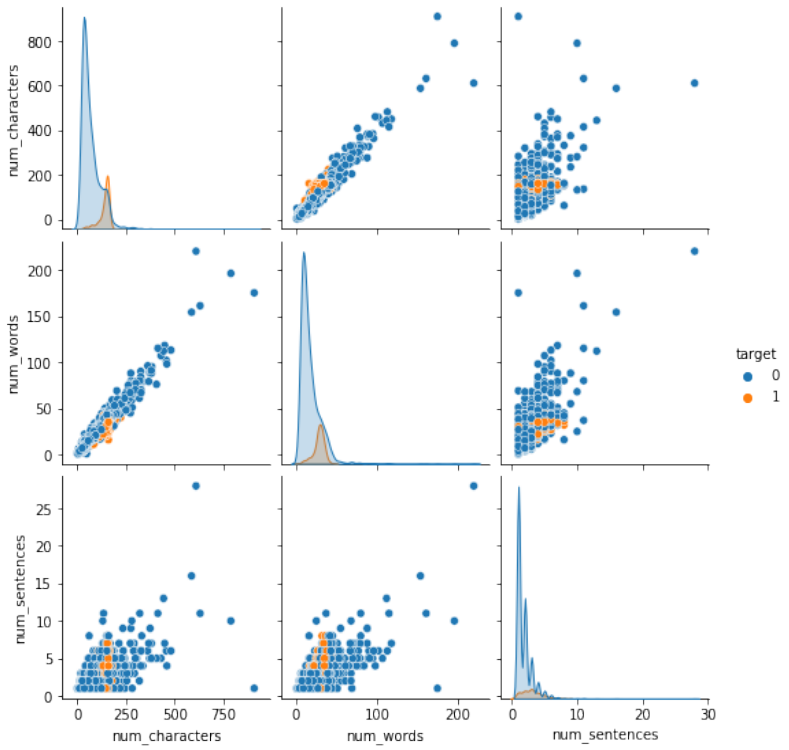


Figure 1. Plotting of the different types of variables of the dataset

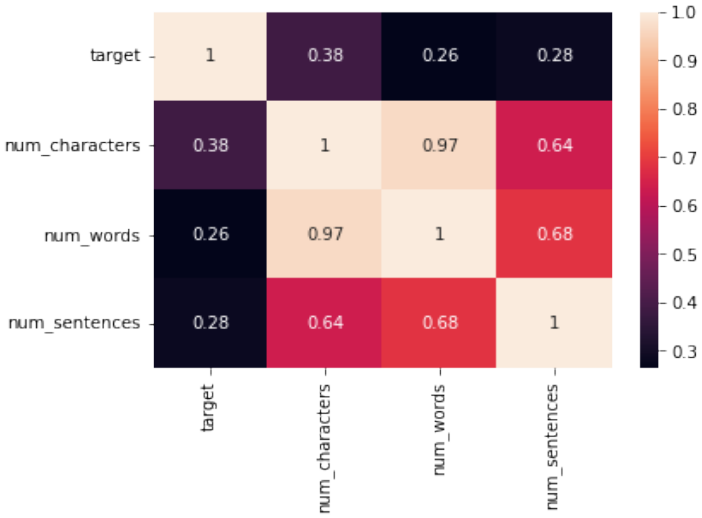


Figure 2. Heatmap of correlation between features in provided dataset

3.4. Dataset Preprocessing

Data preprocessing is an important step in machine learning which involves the manipulation or removal of data before it is utilised to ensure or improve performance.We divided the preprocessing task into the 5 major steps (in order):

1. Converting the sms text into ***lower case*** letters to ensure uniformity in the letter case.
2. ***Tokenization:*** breaking of large sms texts into smaller units/tokens (words)
3. Removing ***special characters*** like '@%()' etc.
4. Removing ***stop words*** (words which have insignificant contribution in the meaning of the sentence) and ***punctuation***.
5. ***Stemming*** (reducing a word to its word stem)

This helped us filter out the relevant and significant words in a sms that will be useful for our model making and its analysis. In order to illustrate the most common words used in spam and ham messages, we used WordCloud. The term "Word Cloud" refers to a data visualisation technique in python for visualising text data in which the size of each word represents its frequency or relevance. A word cloud can be used to emphasise important textual data points.

Figure 3. Word Cloud for Spam texts

4. Methodology: Model Building and it's Analysis

4.1. Model Details

For our model building we started with Naive Bayes, as it's known to perform well on textual data. Many machine

Figure 4. Word Cloud for Ham texts

learning models such as Naive Bayes takes numerical as inputs. Therefore, the transformed text that we obtaied after data preprocessing has to be converted into a numerical data. This is done using vectorisation. Vectorisation is the process of converting input data from text into vectors of real numbers which is the format that ML models support. There are several vectorization methods:

1. ***Bag of Words/Count Vector:*** An algorithm for converting text into fixed-length vectors. This is accomplished by counting the number of times the word appears in a document.
2. ***Word2Vec and GloVe!:*** A group of related models used to produce word embeddings. Uses a novel idea of dense distributed representation of each word. Now 'context' is embedded into each word. Word2Vec will be used later together with LSTM
3. ***TF-IDF:*** TF-IDF or Term Frequency-Inverse Document Frequency, is a numerical statistic that's intended to reflect how important a word is to a document. Sequential/contextual information is still absent.

$$TF = \frac{Frequency\ of\ word\ in\ a\ document}{Total\ number\ of\ words\ in\ that\ document}$$

$$IDF = \log(\frac{Total\ number\ of\ documents}{Documents\ containing\ word\ W})$$

$$TF - IDF = TF * IDF$$

4.2. Performance metrics

For testing our model we used various performance metrics such as accuracy, precision, recall and F1-score.

1. ***Accuracy:*** Accuracy measures the overall efficiency of a classifier. For our model, we require that most of the spam and ham messages are classified correctly for good performance. Accuracy is calculated using the following:

   Accuracy = TP / (TN +FP+ FN +TP)

2. ***Precision:*** The ratio of the true positives to the sum of the true positives and false positives is defined as precision. In our model, its the measure of the number of messages in the dataset classified correctly out of all the positive samples.

   Precision = TP / (TP + FP)

3. ***Recall:*** It is the ability of a classifier to categorize positively labeled data. Hence, for all messages in our data set, it conveys to us the number of correctly classified ones. Mathematically it's represented as:

   Recall = TP / (TP + FN)

4. ***F1 score:*** The harmonic mean between precision and recall is defined as the F1 score. A high F1 score indicates a good trade-off between precision and recall.

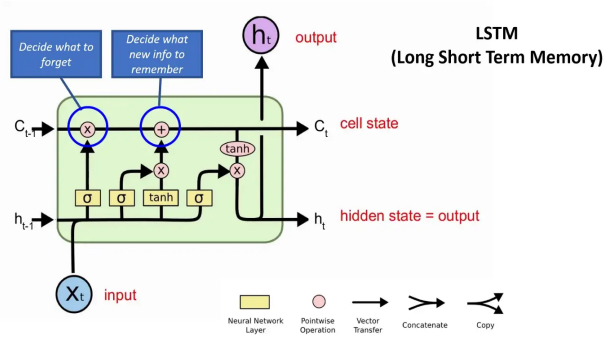
   F1-Score = TP / (TP + 0.5 (FP + FN))

5. GridSearchCV - Hyperparameter Tuning

In order to find the most accurate predictions for our model, we used various methods, one of them being the Grid-Search which is used to find the optimal hyperparameters of a model.

Classifier Algorithm	Optimal Parameters
Decision Tree	'ccp alpha':0.001, 'criterion':entropy, 'max depth': 12, 'max features':auto
Random Forest Classifier	'max depth': 10, 'max features': 7, 'n estimators': 10
Naive Bayes	'estimator':MultinomialNB(), 'var smoothing':0.129
Logistic Regression	'max iter': 5000, 'multi class': 'ovr', 'penalty': 'none', 'solver': 'lbfgs'
KNeighborsClassifier	'algorithm': d tree, 'n neighbors': 1, 'weights': uniform

6. Long Short-Term Memory (LSTM)



LSTM: For Neural Network architecture, we decided to use LSTM. Long Short Term Memory (LSTM) is a variant of the Recursive Neural Network (RNN), which is used for sequential data processing tasks. It allows our hidden state to learn long term dependency due to architectural changes, whereas previously RNNs and ANNs faced problems of vanishing gradient. Various literature reviews suggest that LSTM outperforms ANNs for Natural Language Processing tasks [2] like spam detection.

7. Results and Analysis

Since the data is very imbalanced, accuracy isn't a good metric for our problem. Precision gives us more relevant results in case of an imbalanced dataset.

After going through a lot of models, as mentioned in Figure 6, the best model was Multinomial Naive Bayes and Random Forest due to their comparatively really good accuracy and precision score ratio.

$$Precision = \frac{TP}{TP + FP}$$

8. Conclusion

Through this midsem report it is shown that preprocessing can play an especially extensive role in getting the data into model inputable form. Machine learning model techniques, and tools used play a vital role in classification scores analysis. The machine learning model techniques used are Multinomial Naive Bayes, Logistic regression, Gaussian Naive Bayes, Decision trees, Random forests, etc. with 3-fold cross validation. Random Forests and Multinomial Naive Bayes give the best accuracies. Our methodology not just includes data cleaning and normalization but also incorporates various Natural Language Processing techniques to remove extraneous parts from the text such that only the core, semantically and sequentially important tokens remain. Finally, in such language intensive task EDA

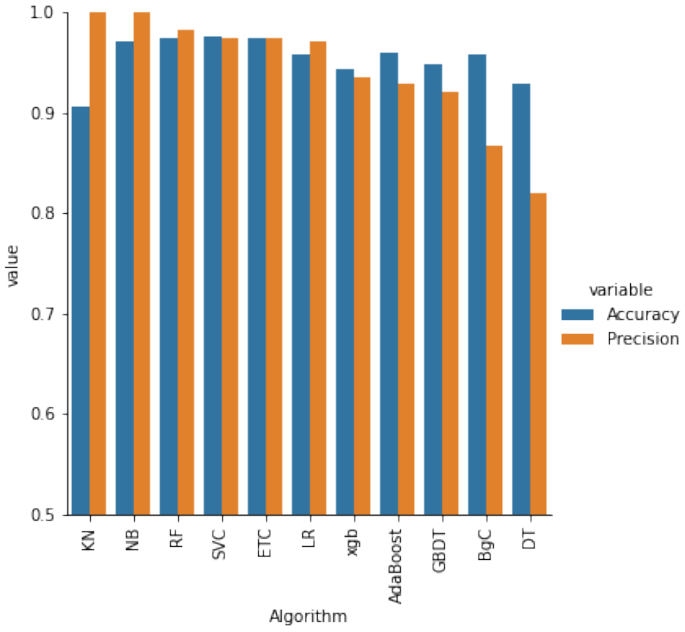


Figure 5. Accuracy and Precision Scores for different models

Model	Accuracy	Precision	Recall	F1 Score
Support Vector Classifier	**0.97582**	0.97479	0.97479	**0.90273**
Decision Tree	0.93037	0.81731	0.81731	0.70248
Logistic Regression	0.95841	0.97030	0.97030	0.82008
Random Forest	0.97486	**0.98276**	**0.98276**	0.89764
Gradient Boosted Decision Trees	0.94778	0.92	0.92	0.77311
Long short-term Memory	0.985	-	-	-

Figure 6. Accuracy, Precision, Recall and F1 Scores for different models

using histograms and word clouds gave us vital feedback on how informal or formal, homogenous or heterogenous our dataset is. Depending on this we again move to preprocessing our data. We have achieved almost perfect accuracy and precision score metrics of around 95-98% in 3 out of 5 models, reiterating that the feature engineering performed by us was adequate.

9. Future Work

This work has a high potential for further research. The spam classification done as of now does binary ham-spam classification. However, as shown in the literature survey Dewi et al (2017) suggest multi-class classification of sms into: regular, info, ads and fraud. This in-depth classification, however, requires more data and metadata. Such dataset hasn't been curated for sms. Multiclass classification is very prevalent and developed for email classification, due to its more economic importance. As email data is more easily available for in-depth analysis. This can be a line of investigation.

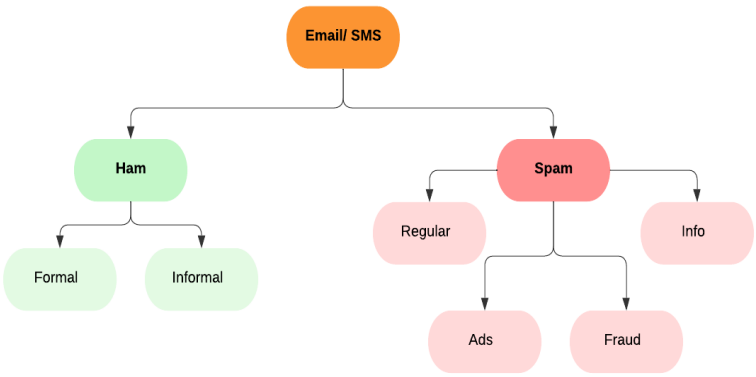


Figure 7. Future Work

10. Individual Contribution

- ***Hitesh Bhandari*** → Literature Survey, Dataset Preprocessing and visualization, Training models, Report Writing, Presentation preparation
- ***Kushiluv Jangu*** → Introduction & Abstract, Dataset preprocessing visualization, Training models, Conclusion, Report Writing, Presentation preparation
- ***Rohit Aggarwal*** → Literature Survey, Dataset analysis, Conclusion, Report Writing, Presentation preparation
- ***Samarth Raina*** → Introduction & Abstract, Dataset analysis, preprocessing & visualization, Results and Analysis, Report Writing, Presentation preparation

References

[1] Mohamad Rahmanianto Rahmad Mahendra Fatia Dewi, Mgs Fadhlurrahman. Multiclass sms message categorization: Beyond spam binary classification, 2018. Spam-ham.

[2] Manisha Sharma Basant Agarwal Gauri Jain. Optimizing semantic lstm for spam detection, 2019. NN Architecture.

[3] L. Budak C. Kawintiranon K., Singh. Traditional and context-specific spam detection in low resource settings, 2022. Mach Learn 111, 2515–2536 (2022).