SMS Spam Detection using NLP with Ensemble Learning Techniques

Aditya Kumar[1], Adarsh Kumar Kar Mahapatra[2], Fancy C.[3]

[1,2,3]Department of Networking and Communication, School of Computing, College of Engineering and Technology, SRM Institute of Science and Technology, Potheri, Kattankulathur 603203, Chengalpattu District, Tamil Nadu, India

Corresponding author mail id: [3]fancyc@srmist.edu.in

***Abstract*— Spam SMS messages are a prevalent problem in today's world and have become a source of annoyance for users. This research paper proposes a novel approach to detect SMS spam using Natural Language Processing (NLP) and creating multiple models and apply ensemble learning on them. The proposed method includes pre-processing the data with NLP techniques such as stopwords removal, stemming, and lemmatizing, extracting relevant features from the text data, and transforming it into numerical representations. The performance of the proposed method was evaluated on a real-world dataset and compared to traditional machine learning algorithms such as Naïve Bayes, SVM.The multiple classifiers are trained on the transformed data, and an Ensemble Learning algorithm is applied to combine their predictions to obtain a more accurate result. The resulting model gave higher accuracy than traditional models. The findings of this research can be useful for developing an effective spam SMS detector to reduce the amount of spam messages that users receive. In addition, the model was hosted on the cloud-based platform, Railway, using the streamlit web app framework, making it easily accessible to users without requiring any software installation.**

Keywords*— *Natural Language Processing, Ensemble Learning, Spam Detection

I. Introduction

The proliferation of mobile phones has brought about a new form of communication, Short Message Service (SMS), which has become an essential aspect of daily life. However, with the growth of SMS as a medium of communication has come the rise of spam messages, which are unsolicited and often unwanted. These spam messages not only cause inconvenience for users but can also lead to privacy concerns and financial loss .To address this problem, various machine learning algorithms have been proposed for SMS spam detection, including multiple types of Naïve Bayes. However, these methods have limitations and do not provide adequate performance in terms of accuracy and speed. To overcome these limitations, this research paper proposes a new method for SMS spam detection using NLP and creating a new model using Ensemble methods. NLP techniques will be used to pre-process the data and extract relevant features, while Ensemble Learning will be used to combine the predictions of multiple classifiers to produce a more accurate result. The results of this research will contribute to the development of a more accurate and efficient method for detecting SMS spam messages, which will benefit users and help reduce the amount of spam messages received. The model will also be hosted on a cloud-based platform, making it easily accessible to users without requiring any software installation.

II. Related Work

The authors in [1] talked about implementing feature selection in the model building so that the model would classify the dataset on those features which would lead the model to classify the messages between spam and ham more accurately. Feature selection would give our models more datapoints to work on and so when we proceed to choosing the different models to work on, the authors of [2] stated that using Naïve Bayes would give us the best accuracy on the dataset. We would perform feature extraction using Tf-Idf Transformer based on [3]. The authors in [4] used lemmatization on the dataset using OpenNLP and [5] talked about how we could implement transformer efficiently. The models that were obtained/talked about in the papers were not able to build an efficient model such as the results in [5] and to increase the accuracy of the model we use Ensemble Learning.

III. Technologies Used

A. Natural Language Toolkit (nltk)

A python library that contains functions and programs that lets us process natural languages such as English and helps us in processing the text messages using stopwords and stemmers such as Porter Stemmer. We use this library to process our text data for our model.

B. Scikit-learn (sklearn)

Scikit-learn is another python library which contains various models/algorithms that we will be using to build our models and then use the library functions to build our ensemble model using the functions provided by the library.

C. Streamlit

Streamlit is a web application framework which helps us in creating a web interface for our model. We will use it to create the webpage which will take the text input from the user and the model will take it as input for our model.

D. Railway

It is an IaaS platform where we will host the website using streamlit which will host our ensemble model and take in text data and predict if the text is spam or legitimate.

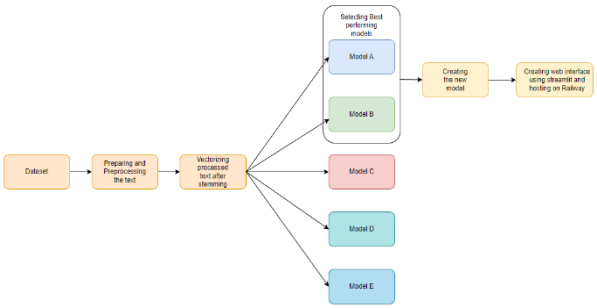

Figure 1. Architecture Diagram

IV. METHODOLOGY

A. Dataset

The dataset was taken from UCI Machine Learning from Kaggle.com and it consists of 5574 text messages which are marked as spam or ham.

B. EDA

When performing Exploratory Data Analysis on the dataset, we find that there are 418 duplicate entries and after removing them from the dataset we get 5156 unique text messages.

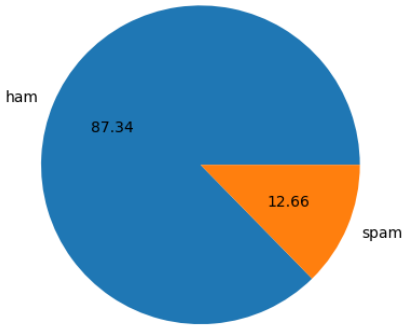

Figure 2. Dataset distribution

The dataset has 12.66% spam messages whereas 87.34% ham messages as shown in Figure 2.

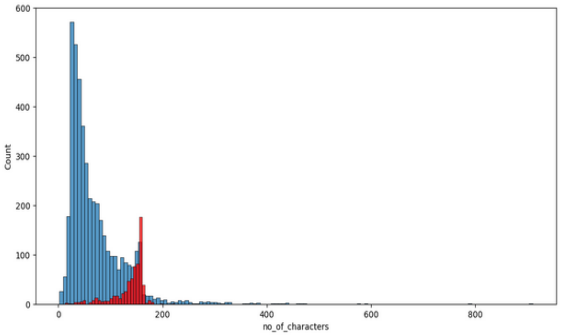

Figure 3.1. Number of Characters

Here we can see that the number of characters in a ham message (blue) is less where as the number of characters in a spam message is more.

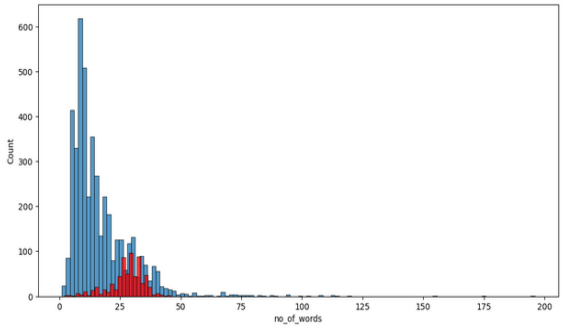

Figure 3.2. Number of Words

Again, we see the number of words in a ham message are very less than compares to a spam message.

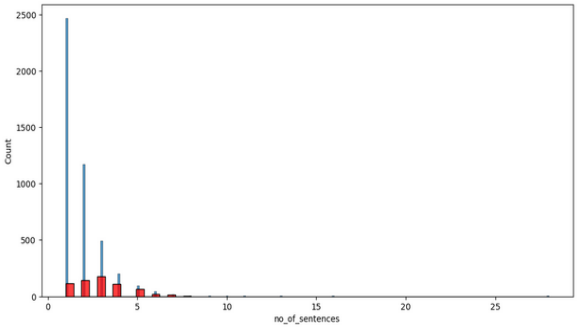

Figure 3.3. Number of Sentences

If the number of words and characters are less in ham than spam messages, the number of sentences in ham message will also be less, as shown in Figure 2.3

C. Data Preprocessing

First we convert all the text to lowercase and tokenize using the tokenize function from nltk library and remove all special characters. From the nltk library we import stopwords and remove any stopwords from the texts and we also use porter stemmer and get the root words from the processed text to get the final processed text.

D. Model Building

From sklearn library we import Tf-Idf Vectorizer to transform the processed text to array. We use 20% of the dataset for test and 80% for training. As stated in Related Words, we use Naïve Bayes as our model first. We choose three types of NB models, Gaussian Multinomial and Bernoulli.

```
gnb = GaussianNB()
mnb = MultinomialNB()
bnb = BernoulliNB()
```

```
gnb.fit(X_train,y_train)
y_pred1 = gnb.predict(X_test)
print(accuracy_score(y_test,y_pred1))
print(confusion_matrix(y_test,y_pred1))
print(precision_score(y_test,y_pred1))
```

```
0.8507751937984496
[[762 134]
 [ 20 116]]
0.464
```

```
mnb.fit(X_train,y_train)
y_pred2 = mnb.predict(X_test)
print(accuracy_score(y_test,y_pred2))
print(confusion_matrix(y_test,y_pred2))
print(precision_score(y_test,y_pred2))
```

```
0.9563953488372093
[[896   0]
 [ 45  91]]
1.0
```

```
bnb.fit(X_train,y_train)
y_pred3 = bnb.predict(X_test)
print(accuracy_score(y_test,y_pred3))
print(confusion_matrix(y_test,y_pred3))
print(precision_score(y_test,y_pred3))
```

```
0.9670542635658915
[[894   2]
 [ 32 104]]
0.9811320754716981
```

Figure 4.1. Naïve Bayes Variations

We see that we get the highest accuracy while using Bernoulli NB but we get more precision in Multinomial Naïve Bayes. We choose the model with the highest precision as our NB model because of the skewedness of the dataset.

We choose more models and see their accuracy and precision on this dataset.

```
clfs = {
    'SVC' : svc,
    'KN' : knc,
    'NB': mnb,
    'DT': dtc,
    'LR': lrc,
    'RF': rfc,
    'AdaBoost': abc,
    'BgC': bc,
    'ETC': etc,
    'GBDT':gbdt,
    'xgb':xgb
}
```

Figure 4.2. Classifiers chosen

SVC - Support Vector Machine (sigmoid kernel)

KN - K Nearest neighbors

MN – Multinomial Naïve Bayes

DTC – Decision Tree Classifier

LR – Logistic Regression

RF- Random Forest

AdaBoost – Ada Boost Classifier

BgC – Bagging Classifier

ETC – Extra Trees Classifier

GBDT – Gradient Boosting Classifier

XGB - eXtreme Gradient Boosting Classifier

	Algorithm	Accuracy	Precision
1	KN	0.896318	1.000000
2	NB	0.956395	1.000000
5	RF	0.962209	1.000000
8	ETC	0.968992	1.000000
0	SVC	0.974806	0.982456
6	AdaBoost	0.968023	0.963964
4	LR	0.947674	0.955556
10	xgb	0.967054	0.939655
9	GBDT	0.946705	0.909091
7	BgC	0.953488	0.866667
3	DT	0.929264	0.831579

Figure 4.3 Accuracy and Precision of models

We see that along with Naïve Bayes, K nearest neighbors, Random Forest and Extra Trees Classifiers give us precision of 1.

E. Model Optimization

To optimize the models for the highest accuracy with precision as 1, we run the model multiple times by setting and changing the number of maximum features of our transformer.

	0	1	2	3	4	5
Algorithm	KN	NB	RF	ETC	SVC	AdaBoost
Accuracy	0.896318	0.956395	0.962209	0.968992	0.974806	0.968023
Precision	1.0	1.0	1.0	1.0	0.982456	0.963964
Accuracy_max_features_1000	0.91376	0.975775	0.972868	0.974806	0.974806	0.959302
Precision_max_features_1000	0.979592	0.966387	0.990909	0.958333	0.982456	0.927273
Accuracy_max_features_2000	0.911822	0.974806	0.971899	0.974806	0.975775	0.963178
Precision_max_features_2000	1.0	0.982456	1.0	0.991071	0.974359	0.9375
Accuracy_max_features_3000	0.906008	0.973837	0.968992	0.973837	0.978682	0.963178
Precision_max_features_3000	1.0	0.982301	1.0	0.982301	0.983051	0.929825

Figure 4.4. Results with different max_features limit

We see that we get high accuracy with 100% precision from K Nearest Neighbors, Naïve Bayes and Random Forest Classifier models.

F. Implementing Ensemble Learning

We choose the base Naïve Bayes model, the K Nearest Neighbors model when max features is set at 2000 and Random Forest model when max features of transformer is set at 3000.

We choose Stacking Classifier for ensemble learning model and chose Random Forest Classifier as the final estimator and obtained our final model.

```
kn = KNeighborsClassifier()
mnb = MultinomialNB()
rfc = RandomForestClassifier(n_estimators=50, random_state=2,)

estimators=[('kn', kn), ('nb', mnb), ('rfc', rfc)]
final_estimator=RandomForestClassifier()
```

```
clf = StackingClassifier(estimators=estimators, final_estimator=final_estimat
```

```
clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)
print("Accuracy",accuracy_score(y_test,y_pred))
print("Precision",precision_score(y_test,y_pred))
```

```
Accuracy 0.9796511627906976
Precision 0.9457364341085271
```

Figure 4.5. Accuracy and Precision of the new model

We get higher accuracy when using the ensemble model that using the Multinomial Naïve Bayes model by 2.3% but precision goes down by 5.5%.

We choose the ensemble model because high accuracy would mean that the model will be able to correctly classify for many text messages in the dataset whereas high precision in ensemble model can fall down when more data is added in the dataset or if some other dataset is used and because false negatives could lead to a user missing important texts which is more severe problem than having some false positives.

V. Conclusion

In this research we were able to preprocess the text messages and stem them and perform feature extraction and use it in out model. The dataset was given to multiple algorithms to give multiple models and we combined the models with high accuracy and precision using Stacking method to get a model that had even greater accuracy.

References

[1] Aakanksha Sharaff, Spam Detection in SMS Based on Feature Selection Techniques. Emerging Technologies in Data Mining and Information Security pp 555–563,September 2018.

[2] O. Abayomi-Alli, A. Abayomi-Alli, Sanjay Misra. A deep learning method for automatic SMS spam classification: Performance of learning algorithms on indigenous dataset, Concurrency and Computation: Practice and ExperienceVolume 34, Issue 17 e6989, April 2022.

[3] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, et al.. 2020. Transformers: State-of-the-Art Natural Language Processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp 38–45,2020.

[4] Mariam Khader, Arafat Awajan, Ghazi Al-Naymat, The Effects of Natural Language Processing on Big Data Analysis: Sentiment Analysis Case Study, 2018 International Arab Conference on Information Technology (ACIT), March 2019.

[5] Xiaoxu Liu, Haoye Lu, Amiya Nayak, A Spam Transformer Model for SMS Spam Detection, 2020.

[6] Aliaksandr Barushka, Petr Hajek, Spam detection on social networks using cost-sensitive feature selection and ensemble-based regularized deep neural networks, Neural Computing and Applications volume 32, pp 4239–4257 2020.

[7] Isra'a AbdulNabi, Qussai Yaseen, Spam Email Detection Using Deep Learning Techniques, Procedia Computer Science,Volume 184, 2021, pp 853-858,2021.