American Academic Scientific Research Journal for Engineering, Technology, and Sciences
ISSN (Print) 2313-4410, ISSN (Online) 2313-4402
http://asrjetsjournal.org/

Text Categorization Model Based on Linear Support Vector Machine

Linda Uchenna Oghenekaro[a]*, Augustus Tammy Benson[b]

[a,b]*Department of Computer Science, University of Port Harcourt, Rivers State, 5323, Nigeria.*
[a]*Email: linda.oghenekaro@uniport.edu.ng* , [b]*Email: bensonaugustus@gmail.com*

Abstract

Spam mails constitute a lot of nuisances in our electronic mail boxes, as they occupy huge spaces which could rather be used for storing relevant data. They also slow down network connection speed and make communication over a network slow. Attackers have often employed spam mails as a means of sending phishing mails to their targets in order to perpetrate data breach attacks and other forms of cybercrimes. Researchers have developed models using machine learning algorithms and other techniques to filter spam mails from relevant mails, however, some algorithms and classifiers are weak, not robust, and lack visualization models which would make the results interpretable by even non-tech savvy people. In this work, Linear Support Vector Machine (LSVM) was used to develop a text categorization model for email texts based on two categories: Ham and Spam. The processes involved were dataset import, preprocessing (removal of stop words, vectorization), feature selection (weighing and selection), development of classification model (splitting data into train (80%) and test sets (20%), importing classifier, training classifier), evaluation of model, deployment of model and spam filtering application on a server (Heroku) using Flask framework. The Agile methodology was adopted for the system design; the Python programming language was implemented for model development. HTML and CSS was used for the development of the web application. The results from the system testing showed that the system had an overall accuracy of 98.56%, recall: 96.5%, F1-score: 97% and F-beta score of 96.23%. This study therefore could be beneficial to e-mail users, to data analysts, and to researchers in the field of NLP.

Keywords: Support vector machine; spam; email; ham; model; feature extraction.

1. Introduction

Electronic mail refers to messages transmitted and received by digital computers through a network. An e-mail system allows computer users on a network to send text, graphics, sounds, and animated images to other users. On most networks, data can be simultaneously sent to a universe of users or to a select group or individual. Network users typically have an electronic mailbox that receives, stores, and manages their correspondence.

---

* Corresponding author.

Recipients can elect to view, print, save, edit, answer, forward, or otherwise react to communications. Many e-mail systems have advanced features that alert users to incoming messages or permit them to employ special privacy features. Large corporations and institutions use e-mail systems as an important communication link between employees and other people allowed on their networks. Although the email system has being a convenient means of communication in recent years, however users have had to deal with managing unsolicited emails, which are also referred to as junk mails or spam. Text categorization is a key technique in text analysis for handling structured data organization problems [24]. It is a very important field in Natural Language Processing (NLP) [23]. They are important in retrieving specific information from a document and providing a guide for a user's search through a large data pool. This is achieved by assigning class labels to different data in a data pool. A major application area of text categorization is the filtering of spam email from a large pool of mails. Email spam is a serious challenge all over the world. According to Kaspersky Laboratory (2014), "almost 65.7% of all emails were considered as spam, with Asia as the highest source of spam mail with 49.1% out of the seven regions in the world". Test classifiers are developed using machine learning algorithms such as Linear Support Vector Machines (SVMs), K-NNs, Recurrent Neural Networks (RNNs), amongst others.

Linear support vector machine (LSVM) is a very efficient text classification algorithm. It is a supervised learning algorithm that is employed for several classification problems [23]. Apart from text categorization, SVM can be applied in credit risk analysis, medical diagnosis and information retrieval. SVM can also handle sparse data while retaining the same accuracy level and offer better generalization abilities. This algorithm will be used to develop our proposed text categorization model for spam mail classification.

2. Related Works

Reference [7] carried out a systematic review of some of the popular machine learning based email spam filtering approaches. Important concepts, attempts, efficiency, and the research trend in spam filtering were covered in the survey. Initially, the applications of machine learning techniques to the email spam filtering process of Gmail, Yahoo and Outlook emails spam filters were discussed and then general email spam filtering process and the various efforts by different researchers in combating spam through the use machine learning techniques were reviewed. They recommended deep learning and deep adversarial learning as the future techniques that can effectively handle the menace of spam emails. However, no spam mail filtering model was developed in this study.

Reference [4] proposed an efficient email classification approach based on semantic methods. They employed the WordNet ontology and applied different semantic based methods and similarity measures for reducing the huge number of extracted textual features, and hence the space and time complexities were reduced. However, to get the minimal optimal features' set, feature dimensionality reduction has been integrated using feature selection techniques such as the Principal Component Analysis (PCA) and the Correlation Feature Selection (CFS). Experimental results on the standard benchmark Enron Dataset showed that the proposed semantic filtering approach combined with the feature selection achieves high computational performance at high space and time reduction rates. However, this model has limitations such as poor preprocessing techniques, weak classifiers and lack of a user-friendly interface.

Semantic implementation of information retrieval system by integrating semantic knowledge in the indexing phase was designed and applied. Reference [21] presented an Arabic search engine based on different levels of terminologies of the language. The Okapi measure gave the best results for a mixture rate that equated 0, 1. The introduction of semantics into the indexing phase considerably reduced the search time compared to the old method by 75%. However, their model was not robust and could not be used to categorize other texts such as English texts.

Reference [8] documented classification based on support vector machine using a concept vector model. The proposed model was novel. The model was for categorizing document(s) based on support vector machine with a concept vector model in order to solve the drawbacks of classifying document using traditional approach which does not take into account the semantic relations among the documents/keywords. The results showed that traditional term-based vector space model yields lower accuracy compared to using the concept vector model. However, this model was not tested using real world datasets.

Reference [9] presented an algorithm for automatically learning a function from related classification problems. Their algorithm's parameter function was used to define a new learning algorithm for text classification, which could be applied to novel classification tasks. They discovered that their learned classifier outperformed existing methods on a variety of multiclass text classification tasks. However, they could not implement this algorithm using a programming language.

Reference [21] implemented text categorization and information retrieval using WordNet senses. They used kNN for text categorization. They investigated three techniques: the direct use of a weighted matrix, the Singular Value Decomposition (SVD) technique in the Latent Semantic Indexing (LSI) model, and the bisecting spherical k-means clustering technique. These techniques were used to experiment information retrieval (IR). The performance analysis revealed error rates for the techniques to be within 30% to 60%. However, they did not clearly state the method of implementation used to derive the results.

Reference [14] proposed k-NN model-based approach for automatic text categorization. They presented a text-based categorization system prototype. The algorithms used for the implementation included k-NN model, kNN, Rocchio and Support Vector Machine (SVM). The system was tested using the 20-newsgroup collection and the ModApte version of the Reuters-21578 collection of news stories in order to evaluate its performance. The result revealed that the kNN model outperformed the other algorithms (SVM and Rocchio) which was used for the text categorization. However, their model could not handle marginal data that falls outside the regions of representations.

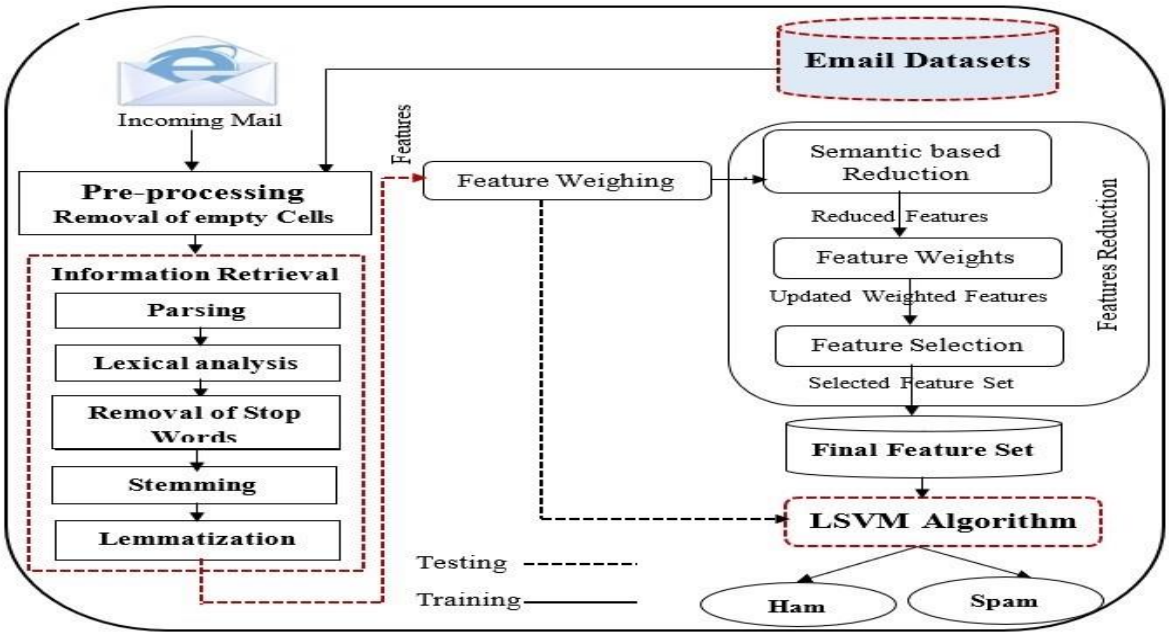

Figure 1: The Proposed System Architecture.

3. Materials and Methods

The proposed approach includes several components for reducing the feature dimensionality to filter the E-mails into two classes: Ham and Spam. The proposed architecture as seen in figure 1 has two phases which are training and testing. The training phase consists of four main modules: Pre-processing; Feature Weighting; Feature Reduction and Classification. The testing phase consists of Pre-processing, Feature Weighting and Classification. The most significant part in the architecture is the reduction module.

3.1 Dataset

The dataset was gotten from Kaggle spam mail repository. 5574 records were found in the dataset, the dataset was split into training set, which contained 80% of the dataset (4459 records) and the test set which contained 20% of the data (1115 records). This splitting was carried out after preprocessing.

3.2 Preprocessing

In this stage, empty records were searched for using he excel special function “DELETE Empty Spaces”. Redundant or repeated records were also removed from the datasets. This guaranteed a clean data which was then passed for information retrieval. Information Retrieval is the first process under which was the breaking of the document into discrete components forming unit documents, this is called parsing. Then tokenization was performed by breaking the word streams into phrases, symbols and other meaningful symbols called tokens. Stemming involve cutting down the affixes in order to discover the root word. Stemming was carried out using the SS stemmer library in Python.

3.3 Lemmatization

Lemmatization involved performing Vocabulary and Morphological analysis of words. It aimed to remove inflectional endings only. The proposed system architectural phases and their outputs used a sample email text as illustrated in this section:

Email text: I'm gonna be home soon and I don't want to talk about this stuff anymore tonight, k? I've cried enough today.

3.4 Feature Weighing

Here the weight of the extracted feature was calculated, where each was weighted by a weight using term frequency/ inverse document frequency method. The frequency calculated the number of times the word appeared in the email document.

3.5 Features Reduction

This is core process of the proposed approach. This module is responsible for reducing the email extracted features in the previous steps by undergoing several different reduction techniques. It consists of three processes: Semantic-based reduction, Features weights updating and Feature selection. In this processing module, the synonyms of each feature are extracted, and then the extracted features are replaced with their synonym set concepts. Besides extracting the synonyms of each term, the hypernym/hyponym relations are considered by consulting the email Datasets as seen in table 1. Next, the similarity between words is measured using different semantic similarity measures. The Semantic similarity measures are calculated based on Email Datasets as well. It begins by generating a set of reduced weighted features as an input for the next process ‘‘feature weights updating”. A semantic weight is calculated for each term in the reduced feature set, and the outputs of this step is a set of updated feature weights. Finally, feature selection process was applied on the current feature set to select the most discriminating and important features that assist the classification accuracy. The main objective of this process is to reduce the feature size in each Email by employing different semantic techniques. After getting the weighted features from feature weighting module in the previous section, Email Datasets is used as a lexical database to link English nouns, adjectives, verbs and adverbs to the sets of synonyms. Such synonyms are called synsets, which are linked together through semantic relations that determine word definitions. In this process, synonyms set of each term in the email are used to group the terms that have common synonyms. Furthermore, the hypernym/hyponym relationships among the noun synsets are considered. The hyponymy relation refers to ‘‘is a kind of” or ‘‘is a”, where it links more general synsets to specific ones, while the hypernym relation represents the inverse of the hyponymy relation. This can help in merging the terms that have common parent or common children. After applying semantic relations, different semantic similarity measures are applied in order to increase the reduction rate for the features.

Table 1: Sample of Kaggle Spam mail.

Ham/Spam	Message Details
Ham	Nah I don't think he goes to us f or he lives around here though
Spam	Free Msg Hey there darling it's been 3 weeks now and no word back! I'd like some fun you up for it still ? Tb ok XxXstdchans to send, a£1.50 to rev
Ham	Even my brother is not like to speak with me. They treat me like aids patent.
Ham	As per request? MelleMelle (OruMiunnnaminunginteNurunguVettam) has been set as your callertune
Spam	WINNER!! As a valued network customer you have been selected to receive a £900 price reward! To claim call 09061701461. Claim code kl341. Valid 12 hours only.
Spam	Had your mobile or more? U R entitled to Update to the latest colour mobiles with camera for Free Call the Mobile Update Co FREE on 08002986030
Ham	I'm gonna be home soon and I don't want to talk about this stuff anymore tonight, k? I've cried enough today.
Spam	SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsanndCs apply Reply HL 4 info
Spam	URGENT! You have won a 1 week FREE membership in our a £100,000 Prize Jackpot! Txt the word. CLAIM to No:8010 T&C www.dbuk.net LCCL TD POBOX 4403LDNW1A7RW18
Ham	I've been searching for the right words to thank you for this breather. I promise I wont take your help for granted and will fulfill my promise. You have been wonderful and a blessing at all times.

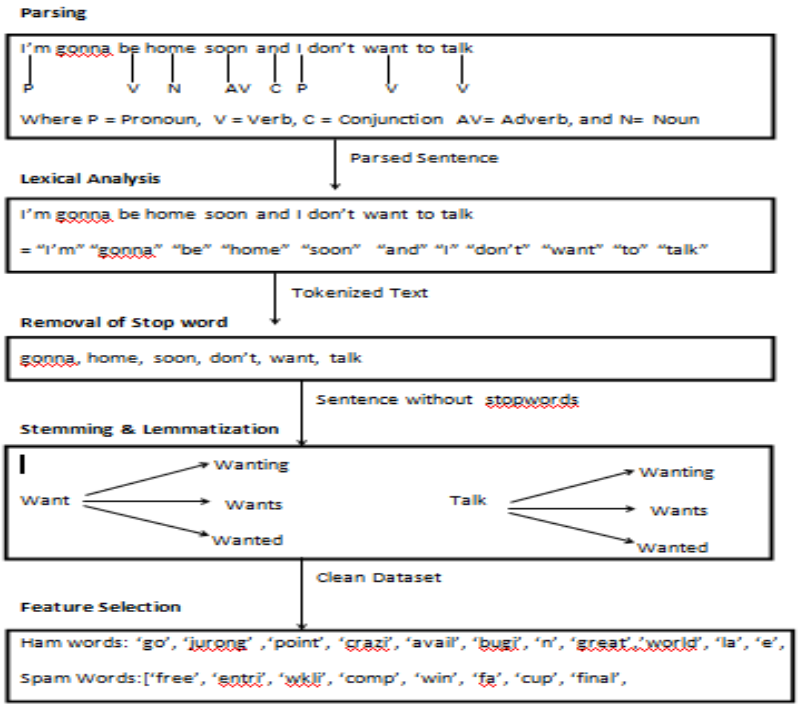

Figure 2: Information retrieval phase.

From Figure 2, a sentence is used to illustrate the phases with which mail is translated into Ham and Spam form.

"I'm gonna be home soon and I don't want to talk."

Parsing: I'm gonna be home soon and I don't want to talk.

P V N AV C P V V

Where P= pronoun, V = Verb, C= Conjunction, AV= Adverb, and N= Noun

Lexical analysis I'm gonna be home soon and i don't want to talk.

= “I’m” “gonna” “be” “home” “soon” “and” “i” “dont” “want” “to” “talk”

Removal of Stop word gonna, home, soon, don’t, want, talk.

Stemming & Lemmatization Want= Wanting, Wants, Wanted.

Talk= Talking, Talked, Talks

Feature Selection =Hamwords: 'go', 'jurong', 'point', 'crazi', 'avail', 'bugi', 'n',

'great', 'world', 'la', 'e',

=Spam Words: ['free', 'entri', 'wkli', 'comp', 'win', 'fa', 'cup', 'final',

LSVM Algorithm = from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df, y, test_size=0.20,random_state=0) # 80% training and 20% test

#Import svm model

from sklearn import svm

spam_detect_model = svm.SVC(kernel='linear') # Linear Kernel

spam_detect_model.fit(X_train, y_train)

y_pred = spam_detect_model.predict(X_test)

4. Analysis of the Proposed System

The proposed system is an improvement of the existing system developed by [4]. The major enhancement in this work is the adoption of an improved text classification algorithm called Linear Support Vector Machine (LSVM). Another enhancement is the integration of information retrieval (IR) processes after the preprocessing stage to normalize the data in the dataset before feature selection is carried out. The dataset was gotten from Kaggle Spam email repository. The dataset consists of 5574 email records classified as Ham and Spam. Table1.

shows a sample of the dataset containing 10 rows. The IR processes are made of modules such as dataset parsing, lexical analysis, removal of stop words, stemming, and lemmatization. These steps are necessary to retrieve the necessary information from the dataset that will be fed to the classifier in order to train it for spam classification and identification.

5. Implementation

For the effective implementation and performance of the proposed system, the following system configuration in terms of hardware and software are required;

a Software requirements

Anaconda IDE, Pycharm, Notepad++, Jupyter Notebook, Python libraries, Python Programming Language, Operating System: Windows 7, 8 and 10.

b Hardware Requirements

RAM: at least 2 GB for best performance, Hard Disk: at least 1 GB, Processor: 450-MHZ core(i_3) and above is recommended, Battery life of 2 Hours and above is highly recommended.

5.1 Choice of Programming Language Used

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

The model was implemented in Python as stated in the section above. The Anaconda platform was used to develop the model. First of all, the anaconda prompt is used to launch the IDE called Jupyter Notebook where the codes will be written. On the IDE, we imported the libraries which will enable us to perform our task. The libraries include Pandas (this will be used to import the.CSV file where our dataset is stored), Numpy, NLTK, Sklearn, Matplotlib etc. if these libraries are not already installed on the IDE, they will need to be preinstalled so that they can be invoked. After importing the libraries, we imported our dataset stored in a .CSV format which is made up of 2 columns and 5574 rows. The columns are “Labels” and “Messages”, the labels include the classification of messages as either ham or spam, while the messages are email messages that correspond to the labels. From the wordcloud library, we downloaded a stop word corpus which will be used to set criteria for the data cleaning phase. The corpus was made up of about 300 stop words. The dataset was saved in a message data frame (msg_df) which was used to refer to the data from then. Using the “value_count” function, we printed the number of the two unique labels in the dataset (ham=4825, Spam=747).

Additional variable called “length” was created in the data frame which return the number of characters in a message. Using a histogram, we demonstrated the relationship between the length of a message and its label (that is. ham or spam), the results demonstrated that the longer the message, the greater the chances of it being a spam.

The next phase was stemming. All the messages were converted to lower case, and then the words were broken down using the “message. Split( )” function. Using the Snowball stemmer (ss) the words were broken down to discover the root word and searched the stop word corpus for the word, if the word was found in the corpus, it was deleted from the message and if not the messages were joined together using the “return " ".join(words)” keyword.

6. Result

Feature weighing was performed by extracting all the words that form the spam words and assigning them to a spam data frame, the same was also done for the ham words. Using the tokenization technique, the sentences were broken into phrase and then into words. All the words found in the spam data frame represented the features of spam mail. They were arranged in order of their frequency of occurrence in the spam messages. The Nltk library was used for this feature extraction and weighing task. From the extracted features a word cloud was formedas shown in figure 3, with the highest occurring words in large letters and the least occurring words in smaller letters. The two categories in our dataset were encoded into numbers (ham=0, spam=1) for proper representation in machine learning.

Out[7]:

	label	message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

Figure 3: Spam email dataset.

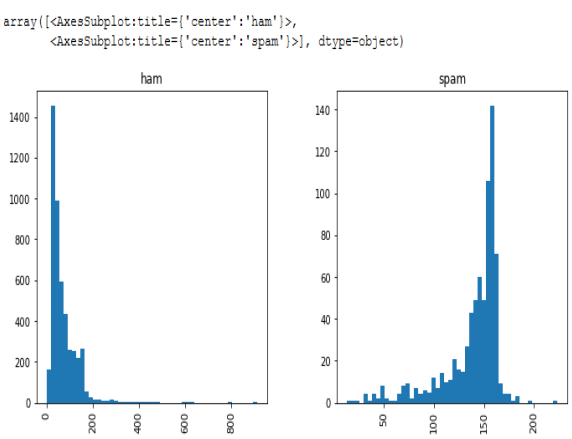

Figure 4: Relationship chart of length of Messages and Labels.

Vectorization is a method of converting categories into numerical data which can be understood by a classifier. Two Vectorization methods were used to achieve this: Count_vectorizer and Tfidf_Vectorizer. Using the fit_transform function, the features were converted into arrays of numbers which represent the finally under Vectorization, the extracted feature were also vectorized.

Classifier training carried out by importing train_test_split from Sklearn library. SVM was also imported form Sklearn and the data was split into 80% for training and 20% for testing. The values used for the training were in

numeric forms and represented as an array hence using the train set from the X-axis and the y-axis, the data was trained. The classification accuracy was generated to be 98.564%. The classification was successfully carried out based on ham and spam messages in the train set.

In order to create an application which can be used to test the model, we developed a web application running on a local cloud server called Heroku. A pickle file was created which and dumped a compressed version of the classification model in it using the joblib.dump function. We also created transform pickle file which will convert the user's input into numeric form for classification. This pickle file serves as a script that will run the user interface. The file has a .pkl file extension. The pickle file takes the input of the user and classifies it as either ham or spam based on previous training stored in it. Also, a "Procfile" was created to specify that the application is web based, also name of application and command that it executes are defined in the Procfile. The Procfile is the brain of the application and is stored without a file extension. A requirements.txt file was created containing all the libraries that are required for the application to run on any system. An html file was created as a user interface shown in figure 4 and CSS was used to style it.

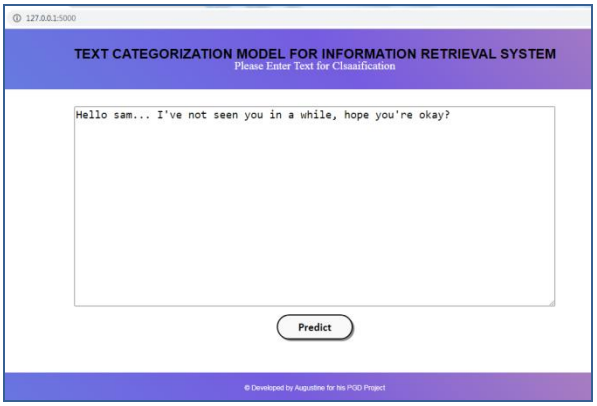

Figure 5: User Interface of Spam Detection Application.

Finally, an app.py file was created to invoke the pickle file and define relationships between the various files in the project folder and the command flow from input to classification result. Flask was used to deploy this app to the web framework. The route of the interacting files in the project from index page to the result page is also defined in the route. Then the requirements.txt file was frozen so that it can be run on the anaconda terminal or prompt. An account was opened on Herokusite and signed up for free. Their after clicked on the deploy button and entered application name and deployed the files.

The app runs on the host address http://127.0.0.1:5000. It is started by creating a new environment on anaconda prompt, navigating to the application directory and running the app.py file. The results from the spam filtering or categorization are shown in figure 5, 6 and 7.

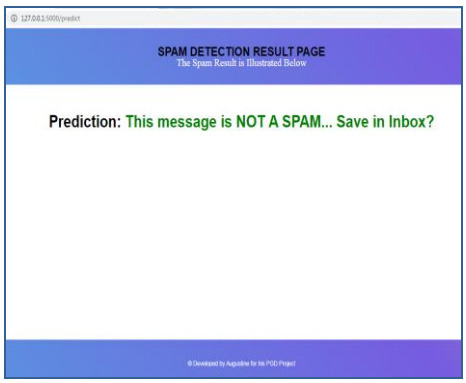

Figure 6: First Classification Result: Ham.

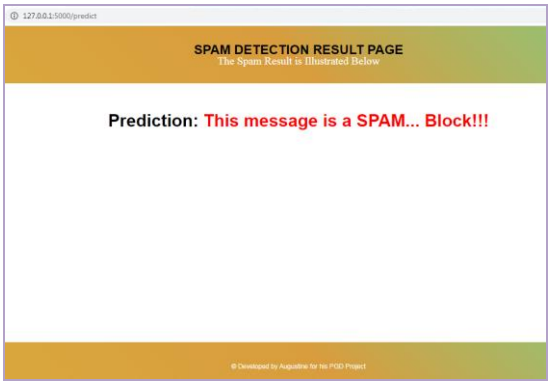

Figure 7: Spam Detection Result 2: Spam!!

6.1 Performance Evaluation

Certain parameters were used to evaluate the efficiency of the model. Some of them include: Precision, Recall (True positives, True Negatives, False Positive and False Negative), F1-Score and Support. The overall accuracy was calculated using the sklearn.metrics function. The other parameters were derived in this way also. From the recall, the confusion matrix for the model was derived.

True positives (TP) is the number of times the model accurately classified a true class as Positive. True negative (TN) is the number of times the model classfied a negative class as Negative. False positive (FP) is a number of times the model classified a negative class as positive. False negative (FN) is the number of times a positive class was calculated as Negative. The confusion matrix generated is shown in Table 2 shows the comparision between the evaluation parameters of the proposed model to the existing [4] model. The proposed model recorded a higher percentage across all parameters, irrespective of its short execution time.

Table 2: Performance Evaluation.

Parameters	Existing System (%)	Proposed System (%)	Parameters
Average Precision	91	96.7	Average Precision
Average Support	83	83.6	Average Support
Average F1-Score	91	97	Average F1-Score
Average Recall	90	96	Average Recall
Overall Accuracy	94	98.56	Overall Accuracy
F-beta	89	96.23	F-beta
Execution time	10 sec	4 Sec	Execution time

7. Conclusion

The existing email filtering or classification systems were limited by weak model applications, lack of user interface and visualization model for spam classification results. In this work, these limitations were addressed by developing enhanced model using LSVM classification algorithm to develop a classification model for email texts. Python was used to implement the model and Flask was used to deploy the application on a cloud server called Heroku. Email datasets containing over 5,000 email messages and labels were used to train and test the

classifier and the classification model. The model was then pickled and used as a script to power a web application for handling spam classification. The classification accuracy was gotten to be 98.56%, with 1424 correct predictions and 10 wrong classifications. The model was executed in 4 seconds which demonstrates no time complexity issues.

References

[1]. Abebaw, T. "Applying thesaurus based semantic compression for improving the performance of amharic text retrieval". M.Sc Thesis, University of Gondar. pp. 1-80, 2014.

[2]. Aseervatham, S., Antoniadis, A., Gaussier, E., Burlet, M. & Denneulin, Yves. "A sparse version of the ridge logistic regression for large-scale text categorization". Pattern Recognition Letters. 32. 101-106. 10.1016/j.patrec.2010.09.023, 2011.

[3]. Aski, A. S. & Sourati, N. K. "Proposed efficient algorithm to filter spam using machine learning techniques." Pacific Science Review A: Natural Science and Engineering, 18(2), 145-149, 2016.

[4]. Bahgat, E. M., Rady, S., Gad, W. & Moawad, I. F. "Efficient email classification approach based on semantic methods." Ain Shams Engineering Journal.9, 3259-3269, 2018.

[5]. Bhaskar, M., Fernando, D. & Nick, C. "Learning to match using local and distributed representation of text for web search. " International World Wide Web Conference, Committee. Austrialia: ACM, 2017.

[6]. Christina, V., Karpagavalli, S., & Suganya, G. "Email spam filtering using supervised machine learning techniques." International Journal of Computing. Science and Engineering,2(9), 3126–3129, 2010.

[7]. Dada, E. C., Bassi, J.S., Chiroma, H., Abdulhamid, S.M., Adetunmbi, A.O. & Ajibuwa, O.E. "Machine learning for e-mail spam filtering: review, approaches and open research problem." Elsevier, 5(1), 1-24, 2019.

[8]. Deng, S. & Peng, H. "Document classification based on support vector machine using a concept vector model.:" Proceeding of the 2006 IEEE/WIC/ACM International Conference on Web Intelligenc. 2006.

[9]. Do, C. B. & Andrew, Y. N. Transfer learning for text classification. IEEE Open Access. 1-8, 2005.

[10]. Drias, H., Khennak, I. & Boukhedra, A. A.. "Hybrid genetic algorithm for large scale information retrieval." IEEE, 1-10. 2009.

[11]. Fonseca, D. M., Fazzion, O. H., Cunha, E., Las-Casas, I., Guedes, P. D., Meira, W. & Chaves, M. "Measuring characterizing, and avoiding spam traffic costs." IEEE Int. Comp.99. 121-112, 2016.

[12]. Gaurav, D. Tiwari, S.M. Goyal, A., Gandhi, N. & Abraham, A. "Machine intelligence-based algorithms for spam filtering on document labeling". Soft Computing, 24(13). 9625- 9638, 2020.

[13]. Goudjil, M., Koudil, M., Bedda, M. & Ghoggali, N. "A Novel Active Learning Method Using SVM for Text Classification". International Journal of Automation and Computing, 15, 1110-1121. 10.1007/s11633-015-0912-z., 2016.

[14]. Guo, G., Wang, H., Bell, D., Bi, Y. & Greer, K.. Using kNN model-based approach for automatic text categorization. 1-15, 2001.

[15]. Joachims, T. "Text categorization with support vector machines: Learning with many Relevant features". 1-7, 1998.

[16]. Karabadji, N., Seridi-Bouchelaghem, H., Bousetouane, F., Dhifli, W. & Aridhi, S. "An evolutionary

scheme for decision tree construction." Knowledge-Based Systems, 119, 166-177. 10.1016/j.knosys.2016.12.011, 2017.

[17]. Khaoula, T., Salma, B., Ksantini, R. & Lachiri, Z. "RBF kernel based SVM classification for landmine detection and discrimination". Doi: 10.1109/IPAS.2016.7880146, 2016.

[18]. Klabbankoh, B. & Pinngern, O. "Applied genetic algorithms in information retrieval." IJCIM, 2, 1-10, 1999.

[19]. Kolluni, J., Razia, S. & Ranjan, S. N. "Text classification using machine learning and deep learning models." International Conference on Artificial Intelligence in Manufacturing & Renewable Energy, 1-7, 2019.

[20]. Korde, V. "Text classification and classifiers: A survey." International Journal of Artificial Proceedings of the 2016 Federated Conference on Computer Science and Information 2016.

[21]. Rosso, P., Ferretti, E., Jimenez, D. & Vidal, V. (2003). Text categorization and information retrieval using WordNet senses. Proceedings from GWC 2004, 299-304.

[22]. Tazzite, N., Yousfi, A., & Bouyakhf, E. "Design and implementation of IR system by integrating semantic knowledge in the indexing phase". ICGST-AIML Journal, 9(1), 49-56, 2009.

[23]. Thangaraj, M., & Sivakami, M. "Text classification techniques: A literature review". Interdisciplinary Journal of Information, Knowledge, and Management, 13, 117-135, 2018.

[24]. Thorsten, J. Text categorization with Support Vector Machines: Machine Learning: ECML-98. 1398. Springer, Berlin, Heidelberg. https://doi.org/10.1007/BFb0026683, 1998.