Analysis and Processing of Spam Messages By Using Natural Language Processing Methods (NLP)

N.Bandžović*, A.Pašić*, Dž.Mehanović*

*International Burch University, Faculty of Engineering, Natural and Medical Sciences, Sarajevo, Bosnia and Herzegovina

nedim.bandzovic@stu.ibu.edu.ba, ajdin.pasic@stu.ibu.edu.ba, dzelila.mehanovic@ibu.edu.ba

Abstract - The Internet is one of the most important factors in our everyday lives. The evolution of the Internet has brought many benefits to us, starting from the fact that we can access valuable and important information by just clicking on something, to buying different products and communicating with our friends and family who may live in many different parts of the world. The Internet is a great tool and it's evolution is not a finished process, and by that, we can include the development of tools which include artificial intelligence, that is also representing an important part of internet usage in the 21st century. However, besides benefits, unfortunately our world has many individuals whose goal is to harm or just simply annoy people on the Internet, in order to cause some damage or to disrupt normal operations in our reality. The Internet can be used as a tool for provocation, causing mental harm, financial losses and for all other activities which can change the typical workflow of a human or an organization. One of these methods include spamming, which simply can be explained by using the definition from Google Dictionary which states that spamming is sending the same message to a large number of internet users. These messages can just be those annoying messages which cannot cause harm, but can frustrate the user and they can also include malicious links, redirects which when empowered, may cause some privacy and financial breaches. Thanks to the community, most of these messages were collected in many different datasets and databases, which can be used for research purposes in order to detect the most common patterns that are followed in terms of generation of spam messages. The results of these researches have helped internet users to easily recognise spam messages and

as well to protect themselves from potential harm which these messages can cause. Some of these researches include usage of Python programming language with its natural language processing tools, which can help in producing some useful results that can be empowered in future spam protection solutions. This paper will concentrate on the analysis of spam messages as well as processing them by using many different machine learning models as well as collecting valuable results from research of literature found on the Internet related to natural language processing in one place.

Keywords: spam, Python, Internet, natural language processing, machine learning models

I.Introduction

Every Internet user, especially those who intensively use social media platforms as well as email engines, will meet the phenomena of spam messages. There are many definitions of what spam messages really are. Kaspersky, in their article [1], explains spam messages in the term of being unwanted messages which are very easy to send. The reason why they are so easy to send is because the only thing that the person who wants to send these messages needs is the computer as well as the social media account or an email address. Kaspersky also mentions that sending spam messages is not so often in terms of mobile phone usage, but it is important to state that it can also occur in this way of working. For example, if you give your phone number to a certain newsletter or an organization, that organization can forward your mobile phone number to another third-party organization which can later send spam messages to the user. This can also occur if a certain person or an organization which contains data related to email addresses or phone numbers get breached, the malicious party may steal those credentials and use them to send spam messages. The person who receives spam messages can be deliberately picked for spamming, but in majority of cases, as [1] states, people who send spam messages often pick their victims randomly, or they just collect all addresses in a bulk, and then they send spam messages. Spam messages do not always have to be harmful, but in the majority of cases, they can be used for malicious purposes which can result in identity theft or in placing malicious software onto the device where the message was opened. In order to detect a spam message, which this paper will also deal

with in the later sections, [1] proposed several patterns which can be used for detection of spam messages. The most common patterns of spam messages include those where the recipient is informed that he / she won a certain prize or a gift card, that the recipient received a certain bonus from the government, suspicious activity being detected on the recipient's account which can be solved by paying in order to protect the account, etc. Besides receiving bonuses and paying for protection, recipients can also receive messages of urgent nature, which include immediate reaction by the recipient. For example, many email users from Bosnia and Herzegovina had received a message in which the attacker introduced himself as an employee of the Bosnian embassy in Cyprus. The "employee" had mentioned that he was attacked by a Cypriot gang and that the gang members had stolen his wallet and passport and is asking for the recipient to immediately pay him an amount of 5000 convertible marks (KM), in order for the so-called embassy employee to return to Sarajevo. The majority of users had ignored the mail, but most probably there were cases where recipients had believed that this was a valid case, and had sent money. One more scam which is also common in terms of spam messages is that many email users around the world, had been contacted by members of the Nigerian royal family who want to transfer large amounts of money to the recipient, due to the fact that the "king" is expected to die soon and he wants to give money to someone who is reliable and trustable. [1] also that for an unknown reason, most of these spam messages have many grammatical and spelling errors, which can automatically signalize that something is wrong, because many organizations, government institutions and others pay attention to these details, because they can also affect the reputation of the party which is contacting the individual. The second common factor mentioned by [1] is that these messages come out all of the sudden. Many organizations and companies will inform the user that something is not wrong and then will send an email message with details. The third common factor is that the message has no common point with the user, for example, the recipient can receive an email address in which he is informed that his delivery will be ready in two or three days, even though the recipient had not ordered anything. We can see that spam messages can have serious consequences, but the most important thing related to them is that there are common patterns which can

be checked in order to determine whether a message is spam or not. As mentioned earlier in the abstract of this paper, many IT professionals and generally the community, had collected various types of spam messages which they had received, and had collected them in the form of databases and datasets, which were later used for research purposes with the use of different tools. This paper, will use Python programming language as well as natural language processing tools in the form of NLTK, Pandas, Numpy and others in order to extract the information related to most common words / tokens that these messages contain as well as usage of spam messages and words in order to build different types of machine learning models. This project is also specific due to the fact that it will collect results of different researches which had been conducted in the segment of spam messages and will embody them in the form of a single project.

II.Literature review

During the research process for this project, we had noticed that there is a large amount of natural language processing projects which are related to spam messages. We can mention [2], where the author had built a Naive Bayes Support Vector Machine model which will be used for classification of spam messages. The project includes usage of different Python libraries, starting from NumPy and Pandas, which were used for graphical representation and depiction of data. The data used was taken from a dataset which contains a variety of spam messages, and the dataset can be found on Kaggle. The author also measured the successfulness of his model, and after getting the parameters which were needed, with the use of API, the model was integrated into Twitter which allowed the model to follow live tweets.

[3] is a project which is similar to the previous [2], but in this scenario, the authors is building a more detailed model which includes manual determination of the type of the message (whether it is spam or not), as well as building different models and algorithms (not only one like in the previous example), where one includes the usage of Recurrent Neural Networks. However, this project was only used for explanation and educational purposes, unlike [2] which was integrated with a social media platform.

[4] is a paper where the authors attempt to prove whether semantic analysis can be used in the terms of spam filtering. Based on the research done, the authors conclude that with the use of semantic analysis and also by

implementing different models, they were able to conclude that over time, semantic analysis models may have the accuracy of 99.21% which confirms the importance of natural language processing and machine learning in terms of spam filtering.

[5] is related to a project where besides using semantic analysis, the authors have also empowered the usage of personality recognition techniques, where with analysis of features which depict and describe a certain personality, the model can determine whether the message that was sent was sent with malicious intent or not. The analysis was done separately. Firstly, the authors had analyzed the personalities of email senders and after this first stage, analysis of messages that were sent was done. After both of these stages were concluded, the authors had collected the results from both stages, and created certain conclusions.

[6] is a project where the author is trying to detect spam email messages by creating a combined NLP-RF model (Natural Language Processing - Random Forest) with using multiple decision trees to cover the natural language processing segment of the project, while the spam filtering is covered with the use of a random node.

III.Methods and Materials

A.Dataset Overview

The dataset that will be used in this project is the “SMS Spam Collection Dataset” from Kaggle. The dataset consisted of 5574 messages that were written in the English language. The authors of the dataset have divided the messages into two categories, one being ham and the other one being spam. The ham category includes all the messages which are considered to be legit and safe for the users, while on the other hand, messages that are placed under the spam category, are as the category’s name is describing, considered to be harmful and as the name suggests - spam. The dataset includes messages from several British social media and SMS platforms. As the official documentation of the dataset states, 425 SMS messages in the dataset are recovered from a famous British forum named “Grumbletext”. 3375 SMS messages which can be found in the dataset were taken from the NUS SMS Corpus, which contains more than 10000 messages that were collected by the students of the Department of Computer Science at the University of Singapore and most of the messages that can be found in the dataset were placed by the

teacher and students of that University. 1002 SMS messages and 322 email messages were also taken from the SMS Spam Corpus v.0.1 Big, which is one of the most known corpuses used for research of spam messages. The remaining 450 SMS messages which can be found in the dataset were taken from one PhD Thesis, that was done by Caroline Tag which was written in 2009 for the University of Birmingham. The size of the dataset is around 492 KB, and it contains 5572 rows with five columns. The columns in the dataset are as follows:

Column name	Description
v1	Category of the message, which can be either ham or spam
v2	Content of the message
Unnamed:2	Contains null values
Unnamed:3	Contains null values
Unnamed:4	Contains null values

Table 1: Columns with their respective description from the dataset used in this project

B.Technologies used

This project will be concluded with the use of the Python programming language. However, in order to get the needed results, this project will also demand the usage of certain libraries. The libraries which were included in this project are NLTK, NumPy, Pandas, Seaborn, Matplotlib, Re as well as the ones which are needed to create the pipeline in which we will run the models that will be built.

NLTK is a library which will allow our Python project to work with human language data. As [7] which is the official documentation for NLTK explains, this library provides many different corpora and lexical resources, which allow the user to tokenize, stem, tag, parse, wrap and perform many other operations which are related to natural language processing. NLTK also offers a large forum which is meeting users from all over the world, where they can exchange experiences and perform many experiments and work on projects together. NumPy is a library which contains many different mathematical functions which allow us to perform various operations of our data when we convert it into the form of an array. NumPy was very useful during the data preprocessing stage as well as adding

additional values which had been calculated from the existing parameters in the dataset, which will be explained in another section of this paper.

The second library which was very important is Pandas.

Pandas provides us many different tools for data cleaning, analysis and preparation. The original state of the dataset was not prepared for any data analysis and operations so Pandas came in as a very helpful tool in the data preprocessing stage.

Matplotlib and Seaborn are libraries which are mostly used for representing data in a graphical manner. Matplotlib allows us to work in a way like if we had used Matlab by having functions which allow us to draw graphs, create figures, add labels to the graph while Seaborn provides Python users a high-level user interface which can be used for plotting and visualization of data.

C.Scenario

The main scenario of this project is to use this dataset in order to gain certain conclusions. As mentioned previously in this paper, in order to detect a spam message, we have some certain patterns which we can follow in order to determine the character and the nature of the message. By performing preprocessing and analysis of the data, we should be able to detect certain patterns in the form of common words which spam and ham messages contain and also to be able to differentiate between spam messages and ham messages, which are, as mentioned earlier in this paper, all messages which are not of harmful, malicious and annoying character and nature. Besides differentiation, clean and preprocessed data may allow us to extract certain results in the form of finding the most common tokens in both categories as well as performing certain analytical operations, whose best representative may be embodied in the form of finding the frequency distribution. After performing the preprocessing stage, we will move to the next segment, and that is model development. For the purpose of this project, we will use the preprocessed data and create models where we will test their accuracies and training scores, as well as generate their own confusion matrices. The models that will be built include Logistic Regression, Support Vector Classification (SVC), Gradient Boosting, Random Forest Classifier and Naive Bayesian Classifier.

D.Preprocessing Stage

In order to work with this dataset, a certain amount of analysis and preprocessing must be done in order to get the needed and correct results. The first thing that had to be done was to determine what features from the dataset will we need in order to extract information and build the models that we need. As this dataset is straightforward, it was clear that the only two features that we will use are the "v1" column that contains the information related to the category of the message and "v2" which contains the actual content of the message. Due to the fact that in the official documentation related to this dataset, there is no information related to the "Unnamed:2", "Unnamed:3" and "Unnamed:4" columns, we had determined that these were the candidates for dropping from the dataset. However, in order to perform full analysis, we also tested whether these columns contain certain information, but however, we had noticed that all of these three columns contain null values, which automatically meant that these columns were either created due to a processing error or were simply left there without any purpose. The conclusion was that these columns will be dropped in the preprocessing stage.

Column name	**Amount of null values**
v1	0
v2	0
Unnamed:2	5522
Unnamed:3	5560
Unnamed:4	5566

Table 2: The amount of null values in each column of the dataset

After dropping the unnecessary columns, we analyzed the overall nature of the dataset, in order to see which category has taken the majority of the dataset so we can compare and check whether our models and our conclusions had been correct at the end. This part of the preprocessing stage has brought us to a conclusion that the dataset with which we are working with is dominated by ham messages, with the percentage of 87% while the remaining 13% are related to spam messages.

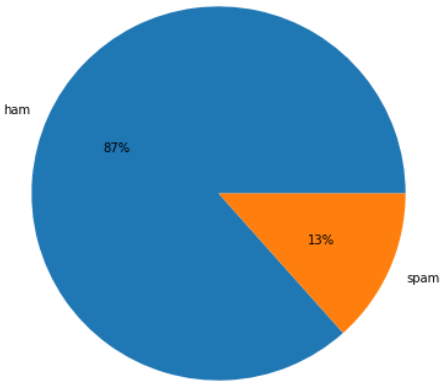

Figure 1:: Presence of both categories in the dataset, expressed in percentages

After cleaning and learning more about the dataset, we were ready to move to the next, and maybe the most important stage of data preprocessing and that is direct manipulation with the data inside the dataset. Due to the fact that this is a natural language processing project, we will have to enter the core of the text in order to prepare it for models as well as for analysis. The first thing that was done is known as tokenization, which is one of the crucial parts when it comes to the NLTK library. Tokenization simply divides a text into smaller parts where each word, symbol inside a text becomes a separate part. This can be easily done by using the built-in functions that are the composite part of the NLTK library. After performing the tokenization, we will need to move to further text processing. The first thing that had to be done was to remove unnecessary symbols from certain tokens. For example, many tokens had been in a form where they contain special characters like '@', '%' , '.' which is understandable due to the fact that the dataset consists of SMS and email messages, where many senders do not pay too much significant attention to grammar rules. To fix this, we had to use regular expressions where we had removed all unnecessary symbols from tokens, but had kept the white spaces in order to preserve the sentence-like nature of the text. After the basic preparation, what was recommended by much literature which was taken into consideration during the project development was to pay attention to stopwords. A stopword is a word which is often used in a language, where for example, in English language we can mention the words "the", "is" and "and" as the most common words. Due to the fact that stop words are not so important because they do not have a significant context which can contribute to the end results, the most common approach which is used in dealing with stopwords is to eliminate them from the text. The NLTK library already contains a detailed dictionary which is full of stopwords, which allows us to compare the newly generated tokens with the stopwords, and in case any occurrences have been

detected, we remove them. This means that we will only keep the most valuable words which have a significant impact and context in our dataset. After removing the unnecessary symbols and unimportant tokens from the dataset, the time has come to concentrate on the nature of each token, and this is where the processes of stemming and lemmatization had taken place. Lemmatization is a natural language processing method where we group different forms of a certain word. An example which was given in [8], mentions the words: “studying”, “studies”, “studied” and “studier” which can be grouped into one word and that is “study”. On the other hand, we had also applied the stemming process, which is as explained in [9] producing the different morphological variants of a root word. It can also be viewed as a process in which a word is reduced to its root word. For example, let’s mention the word “programmer” where with the use of the stemming process, we can reduce it to the root word of “program”. After performing these operations, we had successfully concluded the preprocessing stage, which will allow us to extract certain conclusions in the form of finding the frequency distribution that will help us in finding the most common words / tokens in the dataset as well as finding the most common tokens in both of the categories, which will help us to extract the most common patterns that can be used for spam detection. Also, the conclusion of the preprocessing stage will also allow us to build our models due to the fact that now we have cleaned and formatted data.

E.Logistic Regression

In order to build our models, the first thing that needs to be done is to explain the method in which those models will be empowered. In this project, we will use something called a model pipeline. A model pipeline is as [10] explains, a sequence of different data modeling and preparation steps. For all models, we will use a pipeline with a common construction. Our pipeline will consist of the model which will be used (for example, linear regression in this case), CountVectorizer which is responsible for conversion of text into numerical data and TfidfTransformer, which is responsible for converting a count matrix to a tf (times frequency) representation, which is very useful if we want to retrieve the accurate amount of a token occurring in a text corpus. After defining the pipeline, now it is time to

explain each model that will be implemented.

Logistic regression is one of the most used models in terms of classification. As stated in [11], it is also known as a logit model. This model is used when we want to find a probability that a certain event has occurred. There are many examples starting from checking whether a person does have a tumor or not or whether a student had passed a certain course or not. In logistic regression models, the most common output is either a 0 or 1 and is in a form of probability. This means if the value that our logistic model has returned is closer to 0, this means that the chances that a certain event will occur are small or it is even impossible. If the value is closer to 1, this means that the chances a certain value occurred are very high. Logistic regression also pays massive attention to odds which are calculated when the probability that a successful event will occur is divided with the probability of a failed event. In order to get the probabilities and predictions, the logistic regression is using a sigmoid function. A sigmoid function contains a curve in the form of the letter 'S' where it maps the value to point somewhere between 0 and 1. The activation sigmoid function which is used in different aspects of mathematics is stated as the following in [11]:

$$f(x) = 1/1 + e^{-x}$$

The logistic regression equation is using the activation sigmoid function in the form of:

$$y = e^{(b0+b1x)} /1 + e^{(b0+b1x)}$$

In this equation, y is the predicted output while x represents the input value which is sent to the equation (for example, any value from the dataset for which we want to create a prediction). The parameter b0 is related to an intercept term which plays a huge role in bias, which is one of the known phenomena which occur in logistic regression while b1 is the coefficient for the x value which has been inputted.

F.Support Vector Classification (SVC)

The other model which we will implement is known as Support Vector Classification. As explained in [12], Support Vector Classification may be used in both classification and in regression problems, even though they are most commonly used in classification problems, where spam analysis is mentioned as one of the most frequently used scenarios. This model uses the principle of a hyperplane. A hyperplane can be simply explained as a line which divides a dataset into two distinct classes.

When the dataset is divided, we then have the occurrence of support vectors, which can be defined as the data points which are located very close to the hyperplane and in case any alteration with support vectors occurs, this can change the crucial properties of the hyperplane starting from its position and etc. In order to determine whether a model was classified properly, we need to check something which is known as the margin. Due to the fact that with the existence of a hyperplane, our dataset has two classes, the most important question is related to determining which class should be taken in order to have our classification performed in a correct manner. In order to do that, we need to check the distance of the nearest data points from both classes from the hyperplane. The data point from a class whose margin is larger means that that hyperplane can be used in order to obtain the most accurate classification.

G.Gradient Boosting

The third model that was chosen for this project is known as gradient boosting. Gradient boosting as referred to in [13] is an algorithm where we have the occurrence of combining weak models in order to create a stronger one. The main principle of how it works is that we combine multiple weak models, and then perform the training process in the form where every new model is trained in a way where it will minimize the loss function of its previous model. All models are placed into an ensemble, and in every iteration, the gradient boosting algorithm will take several inputs from the models inside the ensemble, starting from the loss function where it will extract its gradient. Then, the algorithm will train a new model with the goal of minimizing the previously calculated gradient of the ensemble. The boosting process will continue until a certain stopping point has been met.

H.Random Forest Classifier

The fourth model which will be built is the Random Forest Classifier model. As mentioned in [14], it is one of the most popular machine learning algorithms and it has a similar behavior like gradient boosting, due to the fact that this algorithm is also taking into account certain parameters from previous models. The way in which Random Forest Classification works is that here we have the occurrence of decision trees, where each three is created by applying different parts of data and

features in order to get a certain prediction. Every tree, based on its analysis of the data and features it had received, creates a certain prediction which can be later used to classify data only if its popularity is also high in other trees. [14] mentions the example of having five trees which need to predict either "Dog" or "Cat". Due to the fact that out of five trees, four of them had returned "Cat" as the predicted value which meant that the final predicted value would be of "Cat".

I.Naive Bayes Classifier

Naive Bayes Classifier is, as explained in [15] an algorithm represented in a field of supervised learning. Naive Bayes Classifier is used in classification problems, it is based on Bayes' theorem and it is capable of building robust and powerful machine learning models for making accurate and fast predictions.

Naive Bayes is treated as a probabilistic classifier because it predicts on the basis of probability of an occurrence. Naive Bayes comprises two words: Naive (meaning that it assumes the occurrence of a certain characteristic is independent of the occurrence of other characteristics) and Bayes (meaning it relies on Bayes' theorem). Due to its probabilistic nature, Naive Bayes Classifier is an excellent candidate for our spam message project. Bayes' Theorem a.k.a. Bayes' Rule can best be described as a way to determine probability of hypothesis with prior acquired knowledge. Formula used for calculating Bayesian probability is the following:

$$P(A|B) = P(B|A)P(A)/P(B)$$

In the formula, P(A|B) is the probability of hypothesis A on the observed event B, P(B|A) is the probability of proof given the probability of a hypothesis is true, P(A) is the probability prior to observing the results while P(B) is the probability of evidence. Some of the advantages of Naive Bayes Classifier are that it does not require too much training, is fast and easy to implement and is highly scalable.

IV. Results

In the terms of the results, we can say that we had managed to extract interesting information. The first scenario which we wanted to satisfy is to find the most common words which occur in the dataset, regardlessly of their category. In order to do this, we used frequency distribution which is contained in the NLTK library which in a matter of seconds, allowed us to find the

most common words in this dataset. The most common words in the dataset are “to” which was used 2148 times and was followed by “you”, “I”, “a”, “the”, “and”, “in”, “is”, “u” and “i”.

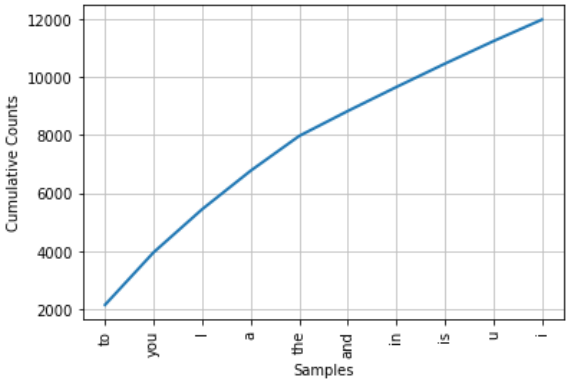

Figure 2::Frequency distribution which shows the top 10 most common words.

Word	**Occurrence**
to	2148
you	1809
I	1487
a	1329
the	1199
and	859
in	827
is	805
u	778
i	743

Table 3: Tabular representation of results obtained by frequency distribution

After extracting the most common words in both categories, we can now move on to finding the most common words in the ham and spam messages separately. After performing preprocessing and data preparation, we were able to extract this information and easily detect the patterns which are common in both categories of messages. This will help us to build a pattern in which we can easily differentiate which message belongs to the ham category as well as to the spam category. In terms of analysis of ham messages, we have obtained the following results which confirm the non malicious nature of messages which can be found under the ham category.

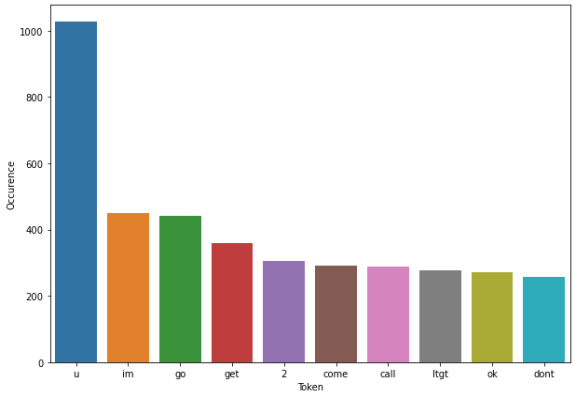

Figure 3::Graphical representation of the most common words in the ham category

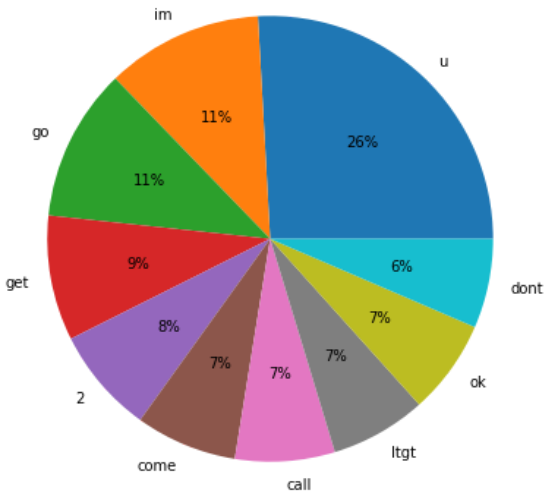

Figure 4::Graphical / pie representation of the most common words in the ham category

The most common words which can be found in the ham category are “u” which is a well known chat slang for the word “you” and was mentioned 1026 times in the ham category and is followed by the words “im”, “go”, “get”, “2”, “come”, “call”, “ltgt”, “ok” and “dont”.

Word	**Occurrence**
u	1026
im	450
go	440
get	360
2	305
come	292
call	287
ltgt	276
ok	272
dont	257

Table 4: Tabular representation of the most common words in the ham category

After analysis of messages in the ham category, we can confirm that the hypothesis which was placed by the authors of the dataset in which they imply that the messages which are sorted under this category contain no harmful or malicious intent, due to the fact that most of them contain words which are commonly used in the form of chat slang, where the best representatives are the words “2” which is a chat slang for the word “too” or “u” which was previously explained in this section. However, when we had turned towards the results of the most common words which can be found in the spam category, we got useful evidence which confirms the hypothesis that was placed by Kaspersky in its article [1].

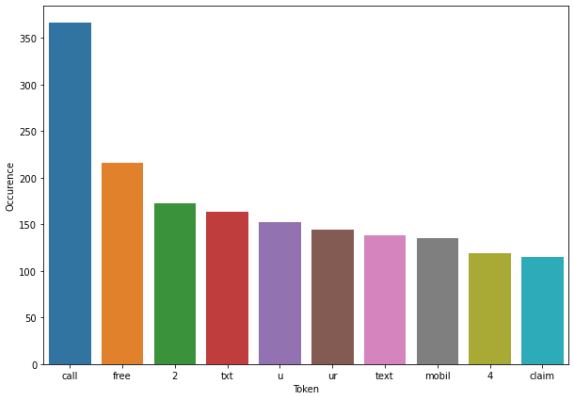

Figure 5:Graphical representation of the most common words in the spam category

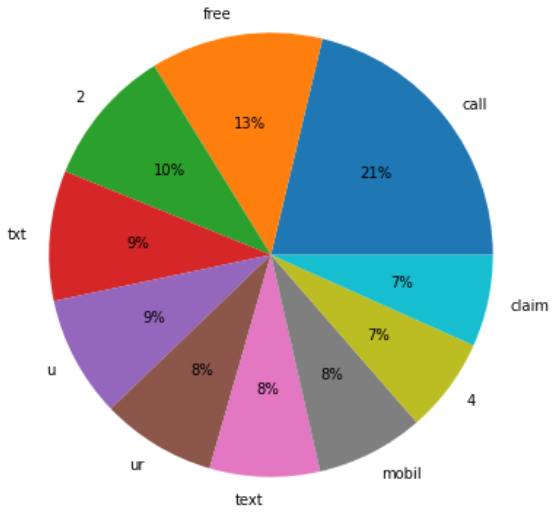

Figure 6 :Graphical / pie representation of the most common words in the spam category

The most common words which can be found in the spam category are “call” which was used 366 times and is followed by the words “free”, “2”, “txt”, “u”, ”ur”, “text”, “mobil”, “4” and “claim”.

Word	**Occurrence**
call	366
free	216
2	173
txt	163
u	152
ur	144
text	138
mobil	135
4	119
claim	115

Table 5: Tabular representation of the most common words in the spam category

Based on the results that we had obtained from this part of the analysis, we can see that the majority of words that was used has a context in which the recipient is invited to call, claim, write a text which can be sent over mobile devices or media platforms in order to trick the recipient to perform a certain action, which can result in either a psychological discomfort or in a more severe scenario, which can be demonstrated in a form of a financial loss or psychological abuse and harm, which is one of the most brutal methods for which spam messages can be used as a help tool.

After processing the text in order to gain conclusions, we can now proceed to the models which we had built. The goal of this segment of the project is to launch many different models and compare their successes after they have processed our data. The first model that was built was Logistic Regression. After we had tested the model, we had seen that our logistic regression model had an accuracy of 96.12% while its training score was 96.88%.

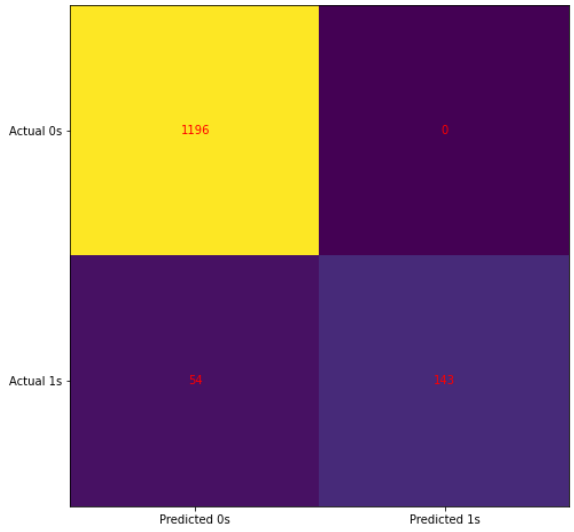

Figure 7 :Confusion matrix for the Logistic Regression model

The second model that was tested was the Support Vector Classifier, whose accuracy was 97.70% and its training score was 99.68%.

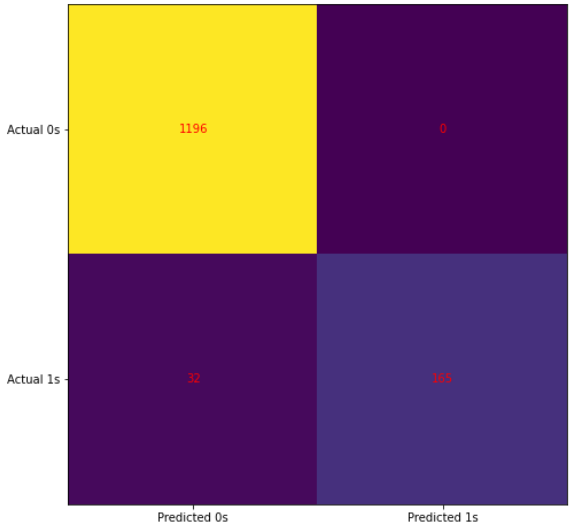

Figure 8 :Confusion matrix for the Support Vector Classifier model

The third model which was part of this research was the Gradient Boosting model. After using the model with our data, the feedback which this model gave us is that its accuracy was around 95.33% while its training score is also around 98.22%.

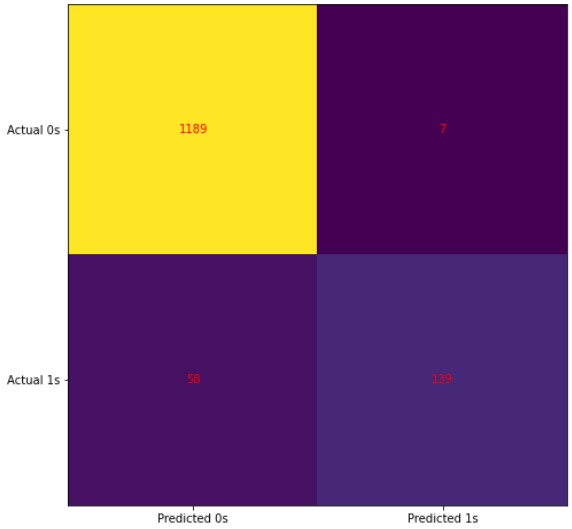

Figure 9 : Confusion matrix for the Support Vector Classifier model

The fourth model which was included is the Random Forest Classifier, which gave us the accuracy of 96.55% and most surprisingly, a training score of 100%.

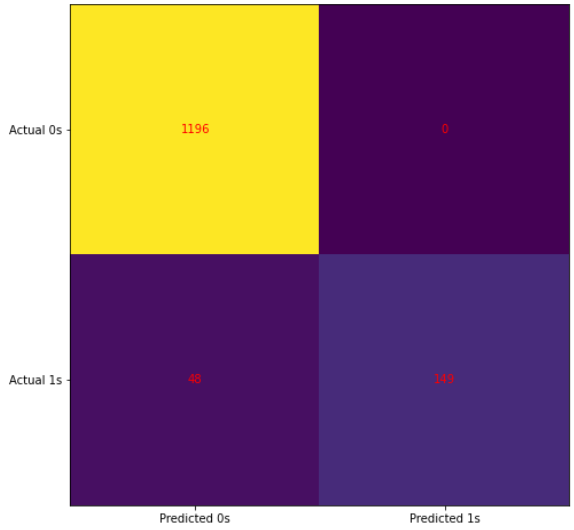

Figure 10 : Confusion matrix for the Random Forest Classifier model

The last model which was tested was the Naive Bayes Classifier, which gave us an accuracy of 96.48% and a training score of 98.63%.

After collecting all the necessary information, we can create certain conclusions related to which model performed the best.

Model name	**Accuracy**	**Training score**
Logistic Regression	96.12347451543431	96.88920794448433
Support Vector Classifier	97.70279971284997	99.68892079444844
Gradient Boosting	95.33381191672649	98.22924144532185
Random Forest Classifier	96.55419956927494	100.0
Naive Bayes Classifier	96.48241206030150	98.63603732950466

Table 6: Tabular representation of performances of all models which were created for the purpose of this project

We can say that we have built very successful models where each one has a large percentage of accuracy and training scores, with Support Vector Classifiers having the highest accuracy rate of 97.70% while the model with the highest training score is the Random Forest Classifier is 100%. The least accurate model in our case is the Gradient Boosting one, which has the accuracy of 95.33% while the model with the least training score is the Logistic Regression model with the score of 96.88%.

V. Conclusion

This project has proved that we can use the Python programming language with its

available tools in order to prove and conclude something from the data that has been fiercely collected by many different participants of this project, starting from students and teachers at one university in Singapore to platform users from all around the world. With the use of exploratory data analysis, natural language processing and machine learning, we have managed to prove hypotheses which were imposed by different sources when we talk about the phenomena of spam messages, which even though they look benign, can cause major damages if enough attention is not paid towards them. Thanks to the usage of some of the most default libraries which are taught to high school and university students, we have been able to create certain assumptions, extract patterns from text, preprocess the text and build machine learning models where we want to check the behavior of the models with the data which we had been working and processing on. For the future, we may plan to work on the same project, but now with usage of additional frameworks which may include Hadoop or PySpark.

REFERENCES

[1] *What are spam texts and how to get rid of spam texts*. (2023, April 19). www.kaspersky.com. https://www.kaspersky.com/resource-center/preemptive-safety/how-to-stop-spam-texts

[2] Garg, A. (2022, November 8). *Natural language processing to detect spam messages*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/10/natural-language-processing-to-detect-spam-messages/

[3] AI, L. (2020, September 8). *Using natural language processing for spam detection in emails*. Medium. https://medium.datadriveninvestor.com/using-natural-language-processing-for-spam-detection-in-emails-281a7c22ddbc

[4] Ezpeleta, E., Zurutuza, U., & Gómez Hidalgo, J. M. (2016). Does sentiment analysis help in Bayesian spam filtering? *Lecture Notes in Computer Science*, 79-90. https://doi.org/10.1007/978-3-319-32034-2_7

[5] Ezpeleta, E., Velez de Mendizabal, I., Hidalgo, J. M., & Zurutuza, U. (2020). Novel email spam detection method using sentiment analysis and personality recognition. *Logic Journal of the IGPL*,

28(1), 83-94. https://doi.org/10.1093/jigpal/jzz073

[6] Rayan, A., & Taloba, A. I. (2021). Detection of email spam using natural language processing based random forest approach. https://doi.org/10.21203/rs.3.rs-921426/v1

[7] (n.d.). NLTK :: Natural Language Toolkit. https://www.nltk.org/

[8] *What is Lemmatization? | Definition from TechTarget*. (2023, March 13). Enterprise AI. https://www.techtarget.com/searchenterpriseai/definition/lemmatization

[9] *Python | Stemming words with NLTK*. (2023, April 15). GeeksforGeeks. https://www.geeksforgeeks.org/python-stemming-words-with-nltk/

[10] Jason Brownlee. (2021, January 16). *A Gentle Introduction to Machine Learning Modeling Pipelines*. Machine Learning Mastery. https://machinelearningmastery.com/machine-learning-modeling-pipelines/

[11] *What is logistic regression? Equation, assumptions, types, and best practices*. (2022, April 18). Spiceworks. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/

[12] *Support vector machines: A simple explanation*. (n.d.). KDnuggets. https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html

[13] *Gradient boosting in ML*. (2023, March 31). GeeksforGeeks. https://www.geeksforgeeks.org/ml-gradient-boosting/

[14] (n.d.). Learn R, Python & Data Science Online | DataCamp. https://www.datacamp.com/tutorial/random-forests-classifier-python

[15] Tom M. Mitchell. (n.d.). GENERATIVE AND DISCRIMINATIVE CLASSIFIERS: NAIVE BAYES AND LOGISTIC REGRESSION. In *Machine Learning*.