# Final Report
for
# Machine Learning Based Spam Detection


**Prepared by:**
**Benny Berba, Syam Jason Bonela,**
**Leonard Garcia, Saiyang Liu, Mary Semerdjian**

**California State University, Los Angeles**

**Computer Science Faculty Advisor:**
**Dr. Jung Soo Lim**

# Table of Contents

# 1. Introduction

## 1.1. Background

The Spam Identifier project was created to help the consumer identify spam from legitimate text messages and emails. The intended users of our product include anyone who regularly uses text messages and/or emails as a form of communication, and who could potentially be adversely affected by receiving spam messages. Because the intended audience of our product would be everyday users, we wanted the Spam Identifier to be easy to use and have the machine learning back-end portion hidden from the users. As a result, we created an easy-to-use graphical user interface (GUI) that any consumer can use to analyze their messages for potential spam.

## 1.2 Problem

The problem our project intended to solve was successfully identifying whether a given text message or email is spam or not spam. We thought this problem was important because more people every day are inundated with seemingly legitimate messages that turn out to be spam messages. Spam messages, while presumedly harmless at first glance, have the potential to expose the user to harmful programs or software.

## 1.3 Contributions

Our group had five members who were assigned roles and tasks that remained consistent throughout the project. Our Computer Science Professor and Advisor was Dr. Jung Soo Lim. We initially split the project group into three sections: machine learning, GUI / design, and documentation. Leonard and Mary worked on the machine learning portion of the project, Saiyang and Benny worked on the GUI / design portion, and finally, Mary, Syam and Benny worked on the documentation. Although our group was split into three different sections, each member helped out wherever another team member needed support.
With help from Mary, Leonard took the lead with the training and testing of the different machine learning models. Benny created the UI mockup of the application, designing the potential screens and interactions between them, as well as contributing to the documentation. As acting team lead, Benny arranged group meeting updates with the TA and worked to resolve any conflicts between team members. Saiyang took the lead on developing the graphical user interface based on Benny's UI mockup. He worked with Leonard to get the machine learning model's results to work with the GUI. Mary took the lead with the documentation, making sure everything was thoroughly and accurately documented. Syam helped with the documentation. Every team member helped with the preparation of the slides.

## 1.4 Solutions

Our project stands out from previous solutions in that our solution has a graphical user interface. While prior solutions also use machine learning models, their solutions can only be found on Jupyter Notebooks. The Spam Identifier project has a lightweight GUI that can be easily learned and used to analyze text messages and emails and does not require the user to have prior knowledge of machine learning models to use.

The key design principles of the Spam Identifier GUI include balance, contrast, emphasis, and proportion. In terms of balance, the GUI uses a symmetrical design layout where elements of an equal weight are on

both sides of an imaginary center line. For contrast, the GUI uses different colors to make various elements, such as buttons, stand out. Another key design principle that the GUI uses is emphasis, which is used to make the important information stand out, as well as also reducing the impact of other information. Lastly, the Spam Identifier GUI uses proportion, which involves the size of elements in relation to one another, to convey what is important – with larger elements having greater importance than smaller elements.

The benefits of our design include using the key design principles to make the GUI intuitive and easy to use.

In the experiment and testing results, we trained and tested different learning models such as decision tree, random forest, naïve bayes, K-Nearest Neighbors (KNN), support vector machine (SVM), AdaBoost, bagging, extra trees, gradient boosting, and XGBoost on two Kaggle datasets (text messages and emails). When comparing the results, we noticed that the random forest classifier achieved the best results, having not only the highest accuracy, but also the highest precision rate.

# 2. Related Technologies

## 2.1. Existing Solutions

One of the existing solutions we found was a spam classifier that used Bayes' Theorem to classify whether a given message is spam or not spam. Bayes' Theorem describes the probability of an event based on prior knowledge of related conditions. In this solution, the authors used the Natural Language Toolkit (NLTK) for processing the messages, WordCloud and Matplotlib for visualization, Pandas for data loading, and NumPy for generating the random probabilities for the train-test split. The authors first preprocessed the data using tokenization and stemming, and then used Bag of Words, Term Frequency-Inverse Document Frequency, and additive smoothing to train the model. This solution was able to achieve a precision of 89.1%, recall of 59.7%, F-score of 71.5%, and accuracy of 93.4%.

## 2.2. Reused Products

No reused products were used.

# 3. System Architecture

## 3.1 Level 0 DFD


Figure 1: The Level 0 DFD is shown above.

The graph starts with a user, where the user begins with an input. A user represents the individual using the application (i.e., the user that uses this application). The user can type in an input, evaluate it, view the evaluation history, change the app's settings, and export all the history evaluations. Last, the user ends with an output.

## 3.2 Level 1 DFD

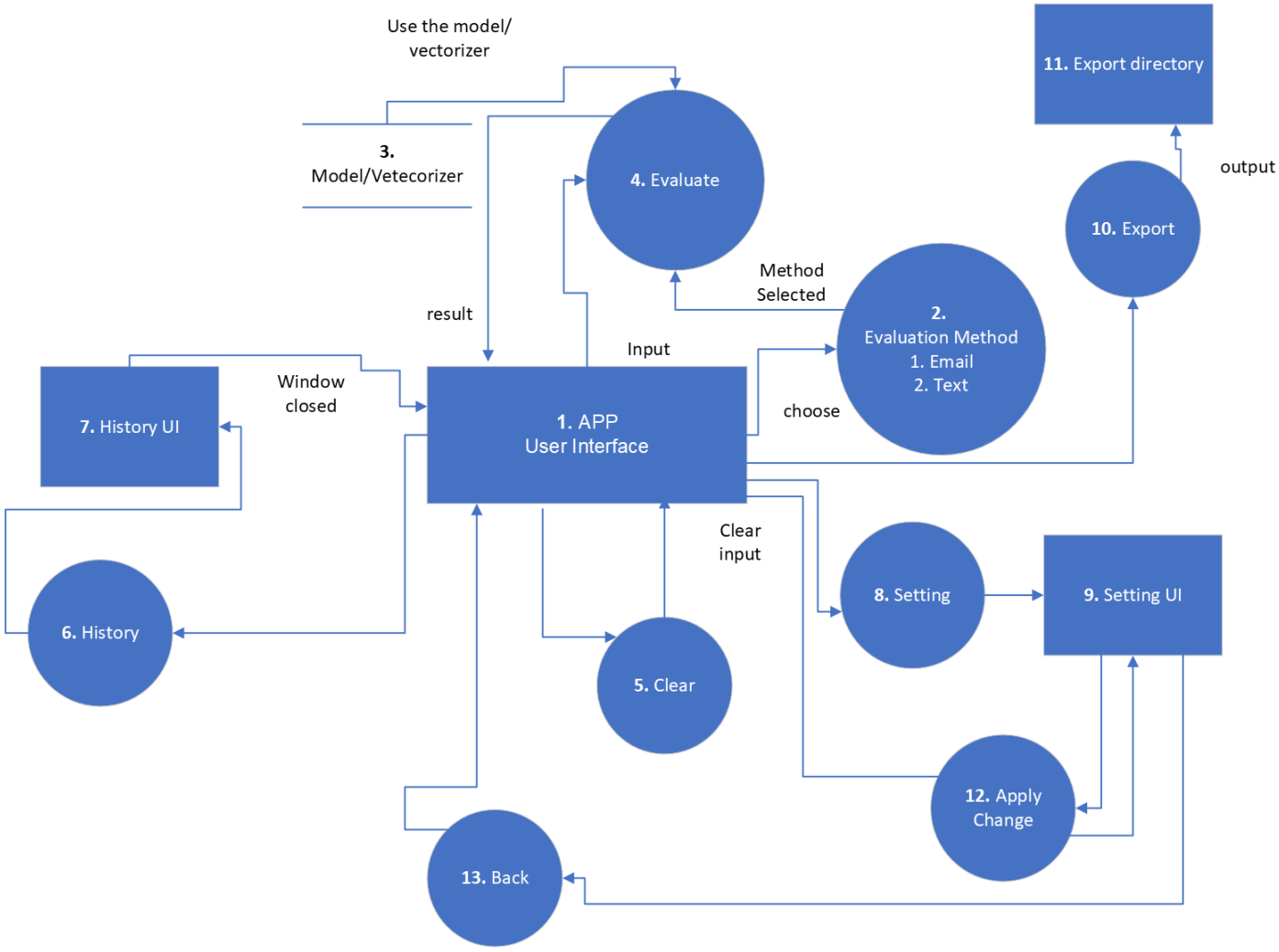


Figure 2: The Level 1 DFD is shown above.

The graph starts with an App/User Interface. The application itself, by default, shows the main user interface page. Next, the Evaluation Method is displayed. Note that there are two methods: Email and Text. In the evaluation method, radio buttons were used to allow the user to choose based on the input type the user wants to use as an evaluation tool. After the Evaluation Method, the Model-Vectorizer was used. Next, the exported machine model and vectorizer files were then loaded into the application for input evaluations. After the model and vectorizer have been exported, the Evaluation method takes the input and the corresponding evaluation method (3) to evaluate the input.

Once the process is done, return the result to the user. The clear cleared the input that the user typed. The history led the user to the History UI page (7). The History UI in a new pop-up window shows all the prior input evaluation results, including input message, result, and types of evaluation. When closed, return to the main page. The setting page leads the user to the setting page (9).

The settings UI page allows the user to adjust different settings, including window resolution and export directory. The exporting page exports all the prior evaluations that can be viewed through the History UI (7) at the target location (11).

The export directory, which is the file export location, runs after running Export (10), and the file will be in a .csv extension. Export Directory can be changed in the Setting UI (9). Then apply changes, after selecting the new setting for the application, the user can apply those changes, and it will affect the whole application (Main Interface (1) and Setting Interface (9)) except History UI (7).

The user will continue to stay at the Setting Interface after the change is applied. Last, the back will lead the user back to the Main Interface (1). If any unsaved change exists, discard it and return it to the Main Interface (1).

## 3.2 Level 2 DFD

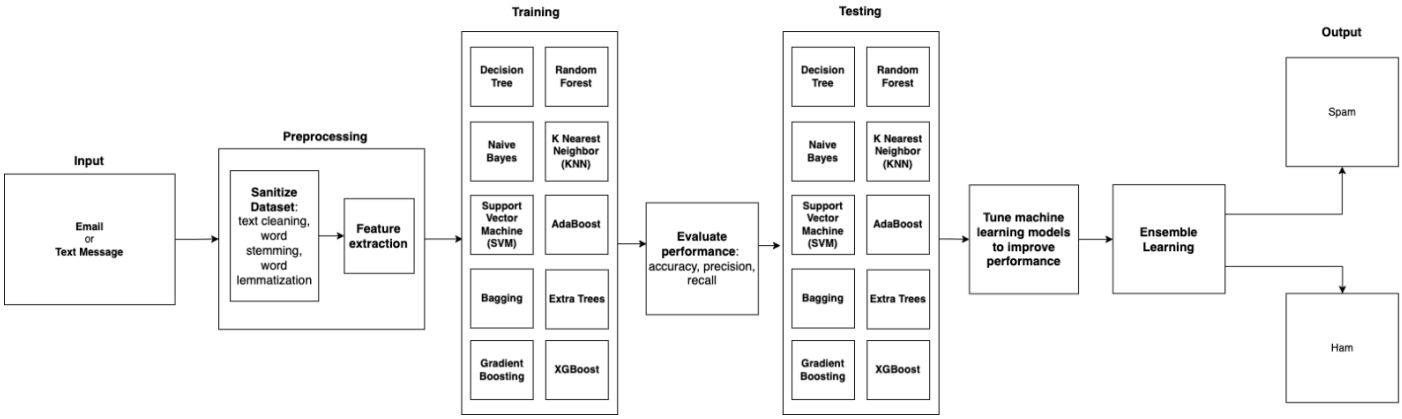


Figure 3: Level 2 DFD is shown above.

Start with an input. An input takes Email or Text Messages as inputs. Next, preprocess the data. Note that there are two preprocessing methods: sanitizing the dataset and feature extraction. Data sanitization does text cleaning, word stemming, and word lemmatizations. Then train the model in the training set. A total of ten Machine Learning classifiers were trained. Evaluate the performance of accuracy, precision, and recall. Test the testing set. Again, a total of ten Machine Learning classifiers were tested. Tune the machine-learning models. The tune machine learning model tunes the machine learning models to improve performance. Then perform Ensemble Learning. Ensemble Learning is used for better prediction results. Finally, output the model. Output the model, such as Spam or Ham, to see if the machine learning models were predicted accurately.

The algorithms we used include decision tree, random forest, naïve bayes, K-Nearest Neighbors (KNN), support vector machine (SVM), AdaBoost, bagging, extra trees, gradient boosting, and XGBoost.

# 4. Results and Conclusions

## 4.1 Spam Text Results

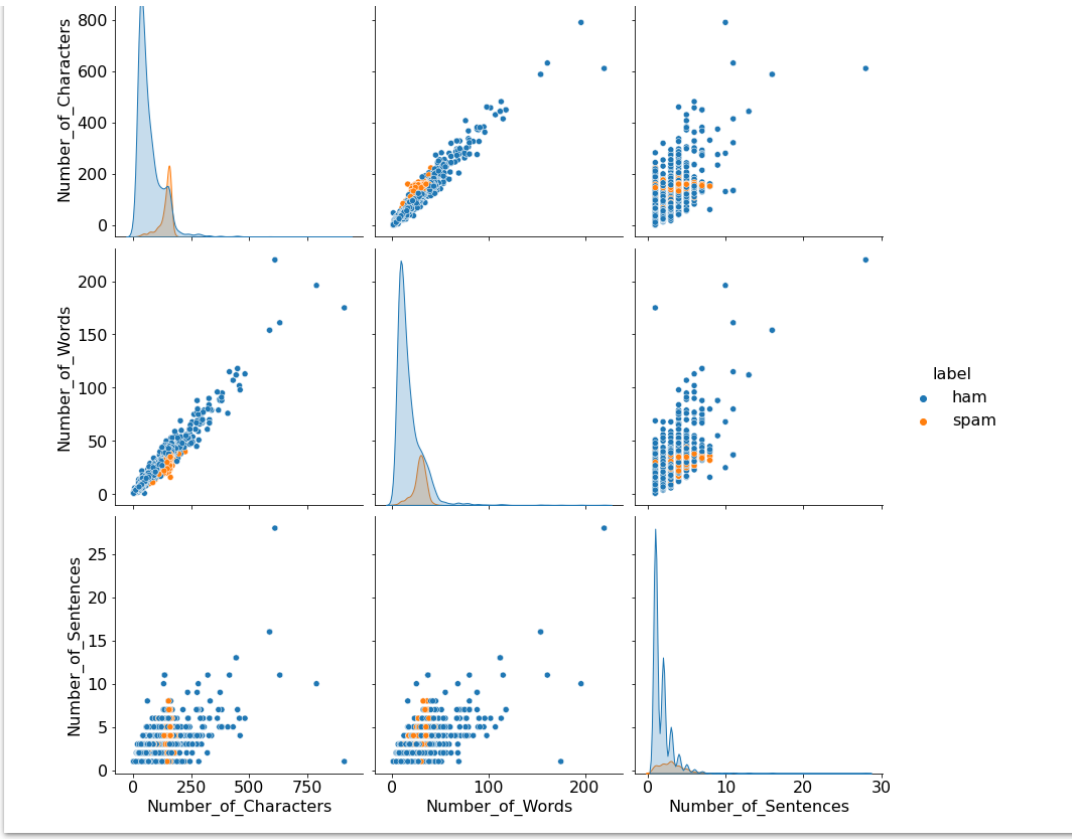


Figure 4: Plot pairwise relationships.

In Figure 4, we plotted the pairwise relationships in the dataset. There are three rows, and the x axis of the row specifies the number of characters, the number of words, and the number of sentences. The y-axis specifies the number of sentences, the number of words, and the number of characters. So, each x-y axis plots the relationships between the sentences, words, and characters. We have the height as four which defines four levels of the height in each axis. The labels are spam and ham with orange being spam and blue being ham.

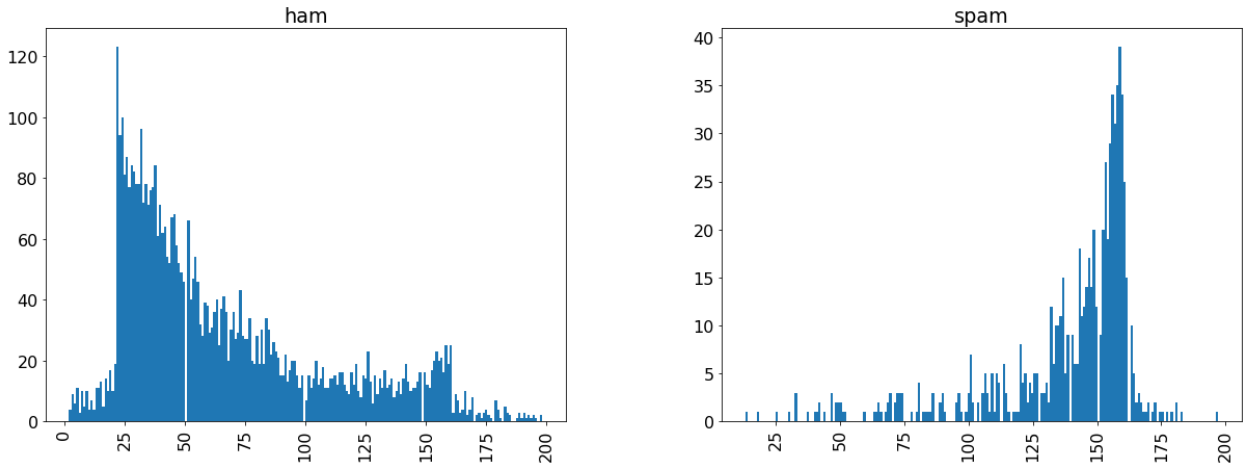


Figure 5: Exploratory Data Analysis (EDA) of Spam Texts using a histogram to plot frequency of message length by the number of characters.

In Figure 5, we conducted Exploratory Data Analysis (EDA) of the spam texts database using a histogram to plot the frequency of message length by the number of characters. During the EDA, we noticed that the legitimate text messages (ham) had fewer characters, while spam text messages had a greater number of characters.

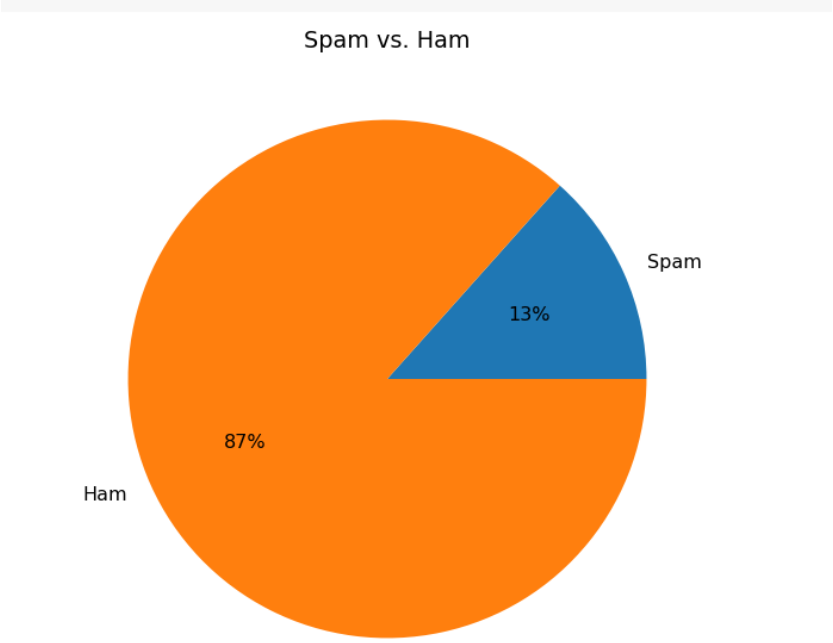


Figure 6: EDA of Spam Texts using a pie chart.

Figure 6 shows a pie chart of Spam Texts. We show the initial distribution of the dataset. Note that, in the pie chart, Ham makes up eighty-seven percent of the pie chart, whereas Spam only makes up thirteen percent of the pie chart, in order to total the pie chart to one hundred percent.

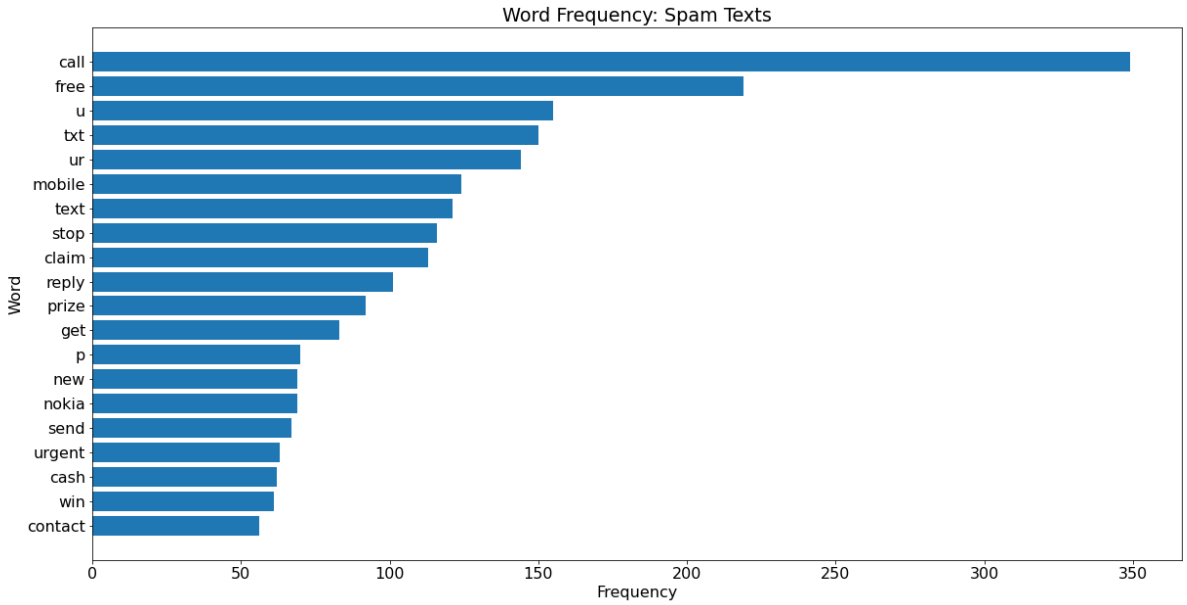


Figure 7: Word Frequency of Spam Texts

Figure 7 shows a chart of the Word Frequency of Spam Texts. The y-axis shows the words, and the x-axis shows the frequency of the words. In the word frequency of the Spam Texts, the call word is used the most, exceeding to three hundred and fifty frequencies. Contact is the least used word, which exceeds over fifty frequencies. The rest of the following words exceed between fifty and three-hundred and fifty.

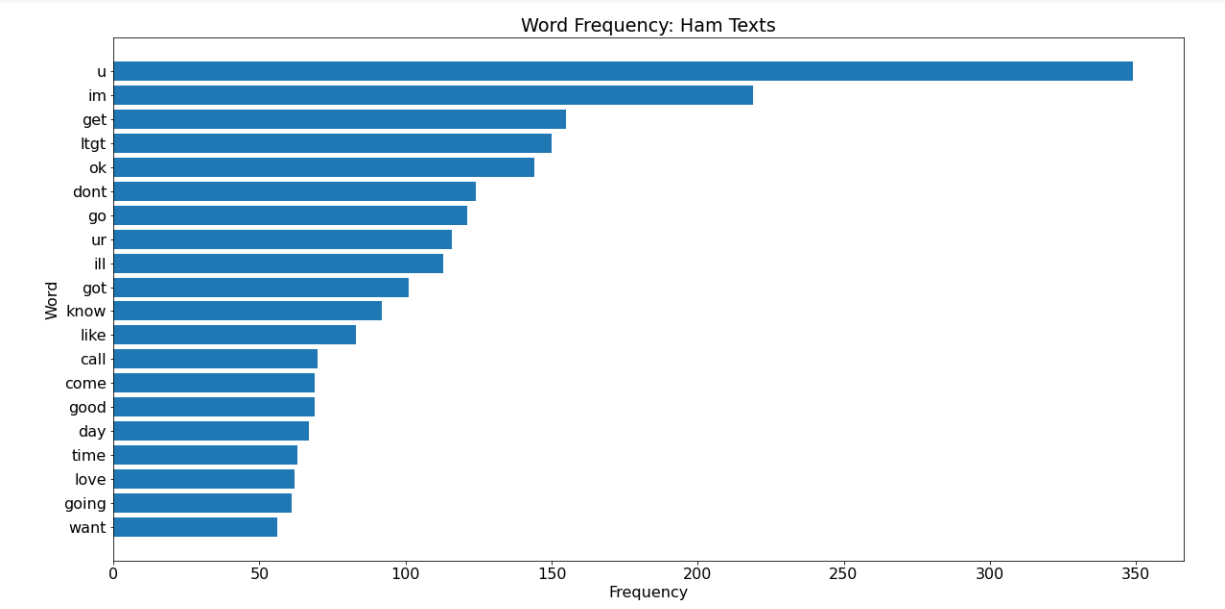


Figure 8: Word Frequency of Ham Texts

Figure 8 shows a chart of the Word Frequency of Ham Texts. The y-axis shows the words, and the x-axis shows the frequency of the words. In the word frequency of the Ham Texts, the u word is used the most, exceeding three hundred and fifty frequencies. Want is the least used word, which exceeds over fifty frequencies. The rest of the following words exceed between fifty and three-hundred and fifty.

| | Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|---|
| 1 | RandomForest Classifier | 0.974888 | 1.000000 | 0.813333 | 0.897059 |
| 6 | Bagging Classifier | 0.974888 | 0.969231 | 0.840000 | 0.900000 |
| 7 | Extra Trees Classifier | 0.973991 | 0.984000 | 0.820000 | 0.894545 |
| 4 | Support Vector Machine Classifier | 0.973094 | 0.991803 | 0.806667 | 0.889706 |
| 2 | NaiveBayes Classifier | 0.968610 | 0.870968 | 0.900000 | 0.885246 |
| 5 | AdaBoost Classifier | 0.962332 | 0.957627 | 0.753333 | 0.843284 |
| 8 | Gradient Boosting Classifier | 0.947085 | 0.978947 | 0.620000 | 0.759184 |
| 9 | XGBoost Classifier | 0.943498 | 0.978022 | 0.593333 | 0.738589 |
| 0 | Decision Tree Classifier | 0.923767 | 1.000000 | 0.433333 | 0.604651 |
| 3 | KNNeighbours Classifier | 0.918386 | 0.983607 | 0.400000 | 0.568720 |

Figure 9: Machine Learning Model Classifiers

In Figure 9, the Machine Learning Model Classifiers are displayed. There are a total of ten classifiers ranging from zero to nine. The Machine Learning Models are the Random Forest, Bagging, Extra Trees, SVM, Naïve Bayes, AdaBoost, Gradient Boosting, XGBoost, Decision Tree, and the KNN. The Accuracy score is given in Figure 10 along with the Precision, Recall and F1 Score. The scores are ranging from 0.0 to 1.0. The highest accuracy is the Random Forest Classifer and the Bagging Classifer.

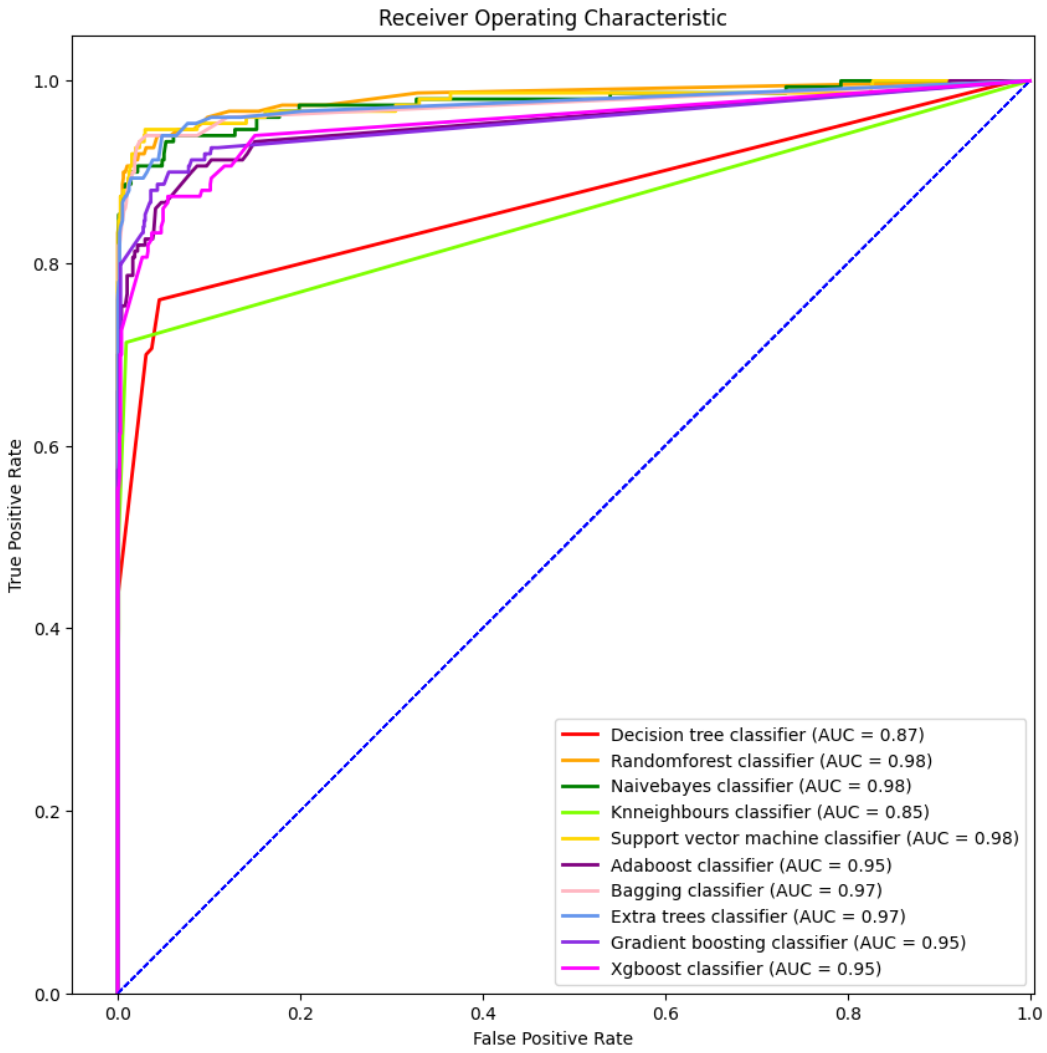


Figure 10: ROC-AUC Curve

Figure 10 shows the ROC-AUC Curve. We see the ROC-AUC Curve ranging from 0.0 to 1.0. The colors specify the different Machine Learning Classifiers that were used to obtain results with the highest accuracy. On the y-axis we have the True Positive Rate whereas in the x-axis we have the False-Positive Rate. The outcomes of the AUC are really great, as most of the classifiers have reached over 80% accuracy.

## 4.2 Spam Email Results

Exploratory Data Analysis

```
[ ]  df
```

| | label | text |
|---|---|---|
| **0** | spam | naturally irresistible your corporate identity... |
| **1** | spam | the stock trading gunslinger fanny is merrill... |
| **2** | spam | unbelievable new homes made easy im wanting t... |
| **3** | spam | 4 color printing special request additional i... |
| **4** | spam | do not have money , get software cds from here... |
| **...** | ... | ... |
| **5723** | ham | search and development charges to gpg here it... |
| **5724** | ham | ceipts from visit jim , thanks again for the... |
| **5725** | ham | nron case study update wow ! all on the same ... |
| **5726** | ham | interest david , please , call shirley crens... |
| **5727** | ham | news : aurora 5 . 2 update aurora version 5 .... |

5728 rows × 2 columns

Figure 11: EDA of the Spam Emails

In Figure 11, we have the Exploratory Data Analysis of Spam Emails. Our data contains five thousand and twenty-eight rows by two columns. The left-hand side shows the rows ranging from zero to thousand and twenty-eight, the labels are spam or ham, and the text is the sentences.

| | label | text | Number_of_Characters | Number_of_Words | Number_of_Sentences |
|---|---|---|---|---|---|
| **2650** | ham | from the enron india newsdesk - april 27 th ne... | 43943 | 8477 | 303 |
| **1380** | ham | from the enron india newsdesk - april 27 th ne... | 43928 | 8364 | 284 |
| **2338** | ham | from the enron india newsdesk - april 23 rd ne... | 31046 | 6348 | 191 |
| **536** | spam | make thousands just sending emails . it ' s ea... | 28423 | 6129 | 329 |
| **2560** | ham | from the enron india newsdesk - may 5 - 7 news... | 27956 | 5689 | 149 |
| **151** | spam | industry giants can ' t match this opportunity... | 22467 | 4046 | 138 |
| **126** | spam | investment op in proven nasa technology hey ,... | 19165 | 3366 | 118 |
| **49** | spam | breaking biotech news hey , i thought you mig... | 18990 | 3361 | 117 |
| **461** | spam | a better investment than the stock market . a... | 18767 | 4112 | 184 |
| **1594** | ham | from the enron india newsdesk - may 4 th newsc... | 18366 | 3756 | 99 |

Figure 12: EDA with Number of Characters, Words, and Sentences

In Figure 12, we have the Exploratory Data Analysis of Spam Emails, but with a more methodological approach. In the EDA we have the labels as spam ham and the text in sentences. In addition, we also have the number of characters, the number of words and the number of sentences.

## 4.3 Spam Identifier GUI Screens

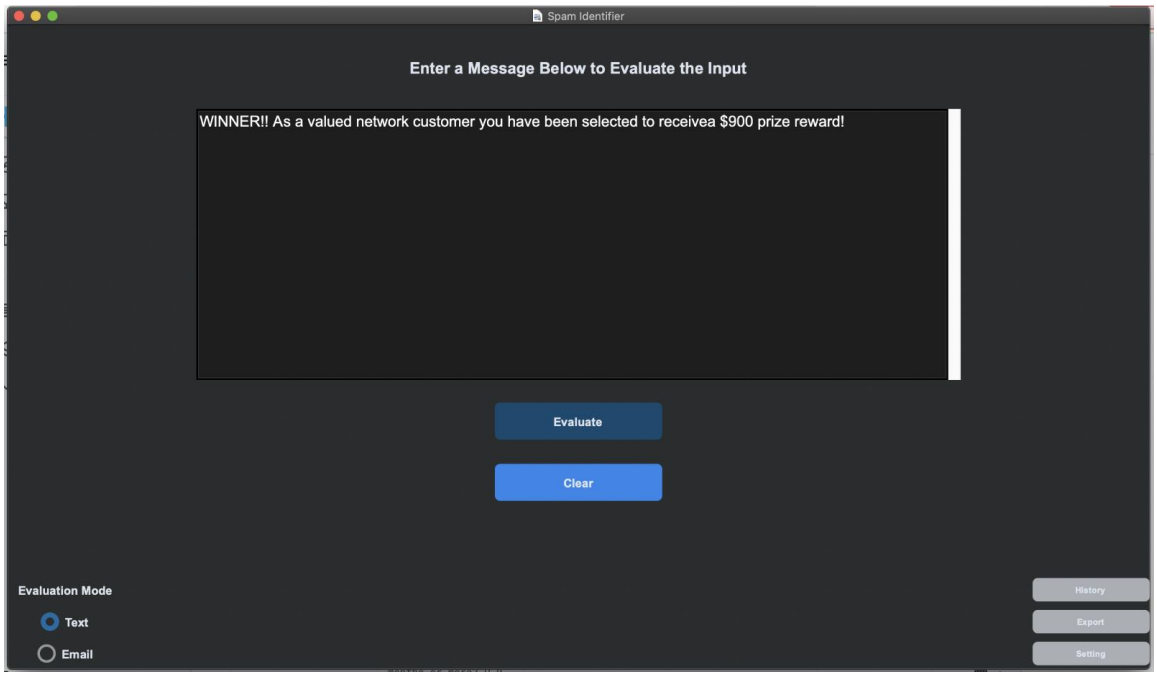


Figure 13: Shows the main interface screen where the user has added input to be analyzed.

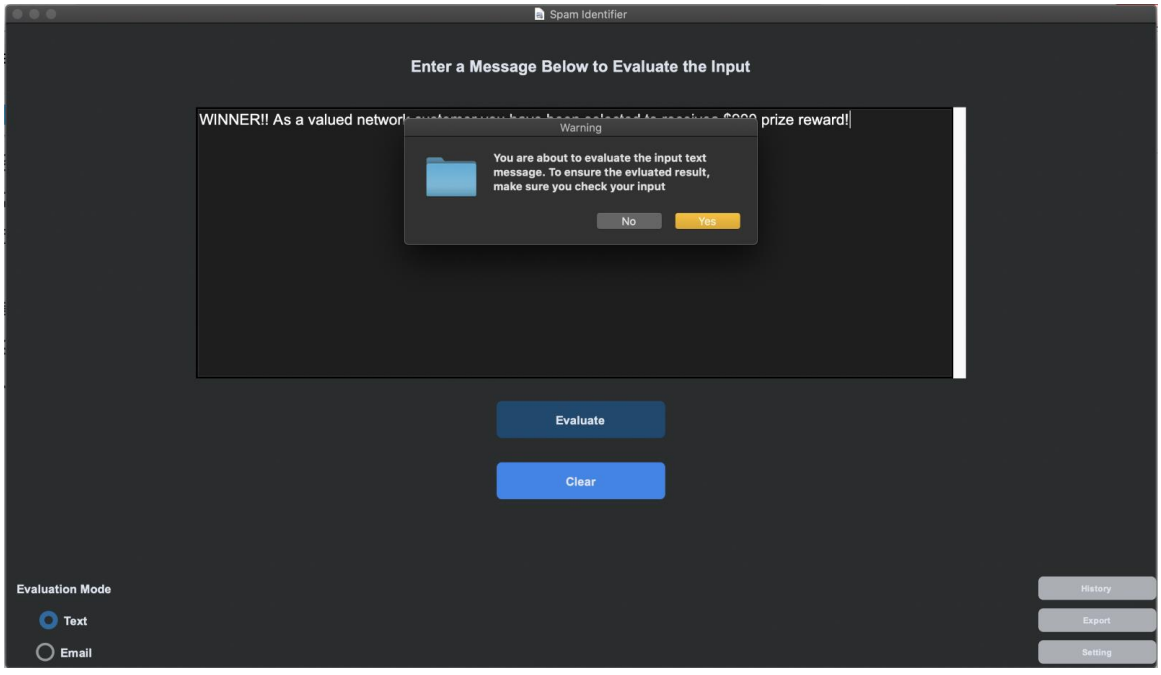


Figure 14: The application prompts the user with a dialog box asking if the input to be evaluated is correct.

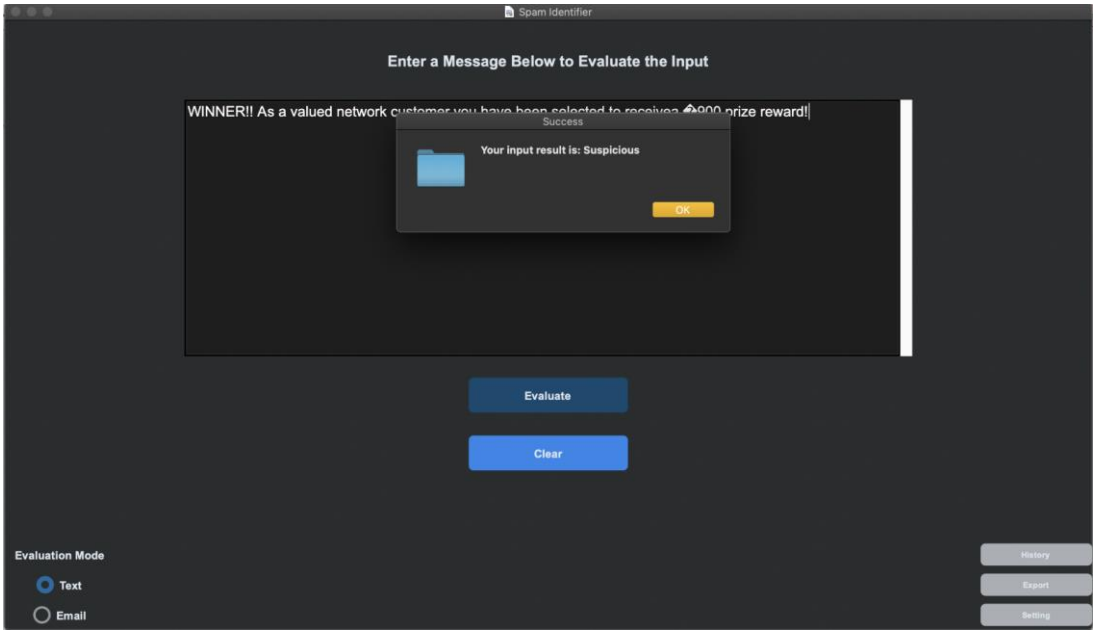


Figure 15: The result is output to the user as suspicious.

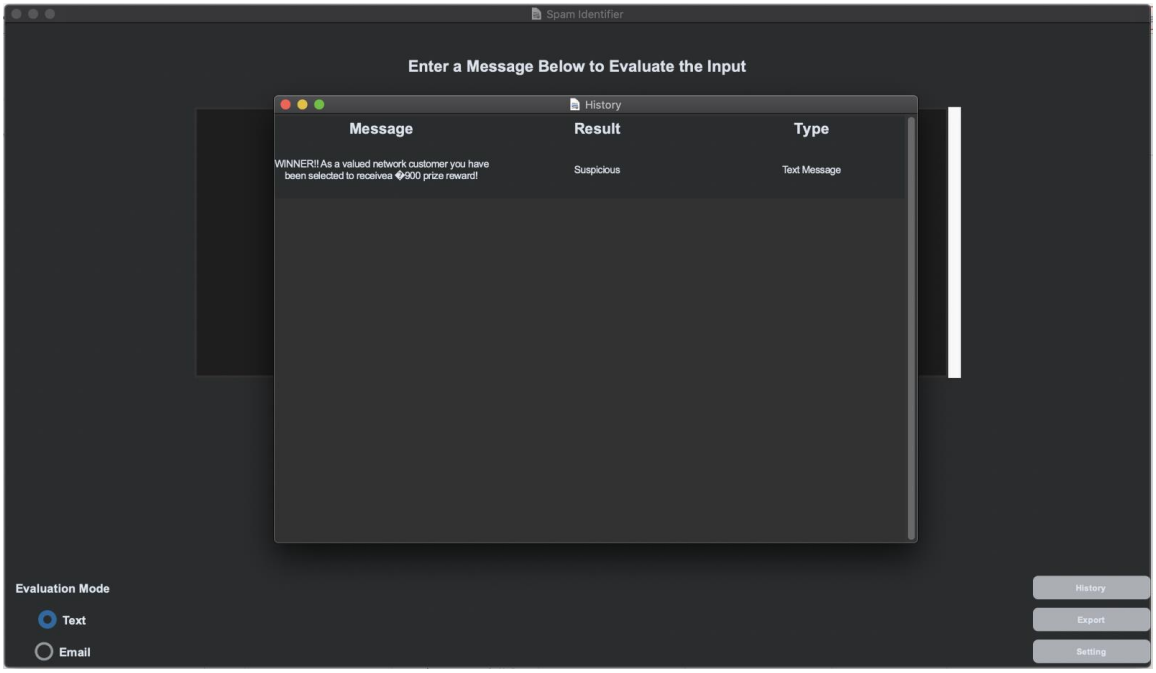


Figure 16: The History screen shows prior input evaluations, which include the message, result, and type (text message or email).

In our project, we successfully created a graphical user interface that uses the random forest classifier to classify text messages and emails as either spam or ham.

The issues and limitations within the graphical user interface were the limited amount of training data and testing data, only to be able to identify old fashion spam messages. Also limited features were part of the issues.

In the ROC-AUC Curve, our graph showed an almost professional graph, however this is bad because it will lead us to overfitting the graph. But, because our metrics in the graph are increasing and starts to touch the graph, this causes us to be too overfit.

## 4.4 Conclusion

Overall, significant results were obtained in the Machine Learning Spam Based Detection.
For future work, the group would like to find a more robust dataset to train the model to give it a more modern result. With more additional features obtained, the group will support exporting more file types, cache the configuration, and Export it to a .cfg file, so the user will no longer need to re-configure the app every time they start the application. Furthermore, we will support more font types (and maybe font sizes) for better visualization for different groups. Lastly, we will manually switch between the light and dark modes of the application to make the app more productive for further recommendations.

# 5. References


1. UCI Machine Learning Repository: Spambase Data Set
2. https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset
3. https://www.kaggle.com/datasets/karthickveerakumar/spam-filter
4. https://www.investopedia.com/terms/n/neuralnetwork.asp
5. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
6. https://github.com/TomSchimansky/CustomTkinter
7. https://www.synopsys.com/blogs/software-security/python-pickling/
8. https://developers.google.com/machine-learning/crash-course/classification/true-false-positive-negative
9. https://www.jeremyjordan.me/imbalanced-data/
10. https://www.toptal.com/designers/ui/principles-of-design
11. https://towardsdatascience.com/spam-classifier-in-python-from-scratch-27a98ddd8e73