Spam Email Detection System using Machine Learning & NLP

Jayanth Mayannanakoppalu Shankara

Abstract

The goal of this project is to use machine learning and natural language processing techniques to create an effective spam email detection system. With the increasing volume of uninvited and damaging emails, it becomes vital to construct algorithms that can reliably classify communications as spam or ham. The project uses the Word2Vec embedding approach to capture the semantic context of words in email content after processing raw email data and extracting significant features.

The dataset is bootstrapped and tagged with spam indicators to increase model generalization and accuracy. Several machine learning models are trained and assessed using different performance metrics, including Random Forest, Support Vector Machine (SVM), and Logistic Regression. A user-friendly interface is also designed to demonstrate real-time email classification using the most effective model.

The finished solution exhibits the practical application of machine learning in the area of cybersecurity and intelligent filtering. Its modular pipeline—from feature extraction and preprocessing to model selection and UI implementation makes it scalable to larger datasets and future additions, such as phishing detection or multilingual spam filtering.

Introduction

Email is still the main way that people communicate in both personal and professional contexts. However, safeguarding inboxes has become a major worry due to the increase in spam emails, which frequently contain viruses, phishing attempts, or pointless adverts. More sophisticated, learning-based methods are required since traditional rule-based spam filters frequently fall short of modern spam tactics.

Our project's goal is to create a machine learning-based spam detection system that examines email content and structure. To cut down on noise, important information including the sender, subject, and message body are retrieved and pre-processed. Word2Vec is utilized for text representation because it helps the model detect spam in spite of linguistic variances by capturing the semantic similarities between words.

Using performance criteria including accuracy, recall, precision, and F1 score, the project compares several models, including Random Forest, SVM, and Logistic Regression. To enable users to test and categorize emails directly, a basic graphical user interface has also been developed. In the end, the project shows how well NLP and vector-based embeddings work to create a more intelligent, scalable spam detection system.

Novel Contribution and Distinction from Existing Approaches

The use of Word2Vec-based semantic embedding for feature representation is what sets this research apart from traditional spam detection algorithms. Our model makes use of Word2Vec to produce dense vector representations that capture the semantic similarity and contextual meaning of words within the email content, in contrast to conventional methods that mainly rely on TF-IDF or bag-of-words techniques—which treat words as independent tokens and lack contextual understanding.

The model's capacity to identify spam emails that circumvent keyword-based filters by using rephrased or synonym-rich language is much improved by this method. Phrases like "Congratulations, claim your reward" and "You've won a prize," for instance, may have different lexicons yet express the same idea. Because Word2Vec can recognize these semantic linkages, the classifier can effectively generalize to a wide range of spam utterances.

As a result, the model performed better on unseen data and the total prediction accuracy significantly increased. Furthermore, we offer a workable and scalable approach for real-time spam classification by including this semantically-aware model into an interface that is accessible to users.

One of our work's main contributions is the innovative incorporation of contextual word embeddings into the spam filtering pipeline, which distinguishes it from other implementations and shows how useful semantic understanding is for email threat identification.

Literature Survey

Numerous studies have proposed innovative approaches, structures, and learning strategies to increase accuracy and efficiency in the important field of spam email detection. Key contributions from current and pertinent works in this field are compiled in the literature review that follows:

The problem of changing spam behavior on sites like Twitter was examined by Yang et al. [1], who also suggested an adaptable architecture to counteract spam strategies. Their research highlights the necessity of ongoing model modifications to address the evolving form of spam. Similarly, for spam categorization, Kumar and Arumugam [2] used a Probabilistic Neural Network (PNN) optimized by Particle Swarm Optimization (PSO), showing enhanced performance as a result of the best feature selection.

SDAI, an integrated evaluation framework for content-based spam filtering algorithms, was first presented by Díaz et al. [3]. Their study emphasizes how crucial it is to assess filters based on a variety of factors for dependability in the real world, not simply correctness. Sharma et al. [4] conducted a comparative study between Naïve Bayes and Neural Networks and came to the conclusion that, when trained on enough data, neural models provide superior learning for spam identification.

Inspired by immune systems, Ma et al. [5] developed a spam detection model based on the Negative Selection Algorithm that exhibits encouraging anomaly-based detection results. In their thorough analysis of machine learning techniques for spam filtering, Guzella and Caminhas [6] highlighted the advantages and disadvantages of several algorithms, such as SVM, ANN, and Bayesian classifiers.

Using benchmark datasets, Kumar et al.'s recent work [7] assessed many machine learning methods, including Random Forest, SVM, and Decision Trees, for spam identification and demonstrated their relative efficacy. An LSTM-based semantic model was proposed by Jain et al. [8], showing how deep learning can identify intricate spam patterns that conventional techniques are unable to detect.

Fake user identification, which is frequently connected to email spam campaigns, was highlighted by Masood et al. [9], who concentrated on social network spammer detection. In their thorough feature selection survey, Chandrashekar and Sahin [10] emphasized the importance of dimensionality reduction in enhancing spam filtering effectiveness.

In order to improve classification efficiency, Mohamad and Selamat [11] evaluated hybrid feature selection strategies that combined filter and wrapper techniques. In order to capture multimodal spam content, Harisinghaney et al. [12] suggested a hybrid spam detection system that uses text and picture analysis utilizing KNN, Naïve Bayes, and Reverse DBSCAN.

Methodology

The methodology adopted in this project follows a structured six-step approach, beginning with the collection and bootstrapping of raw email data, followed by data preprocessing to clean and normalize the content. Key features such as sender, subject, and body are extracted and converted into semantic vectors using Word2Vec. The labelled data is then split for training and testing using classifiers like SVM, Logistic Regression, and Random Forest. Model performance is evaluated using standard metrics, and a simple UI is developed to enable real-time spam classification.

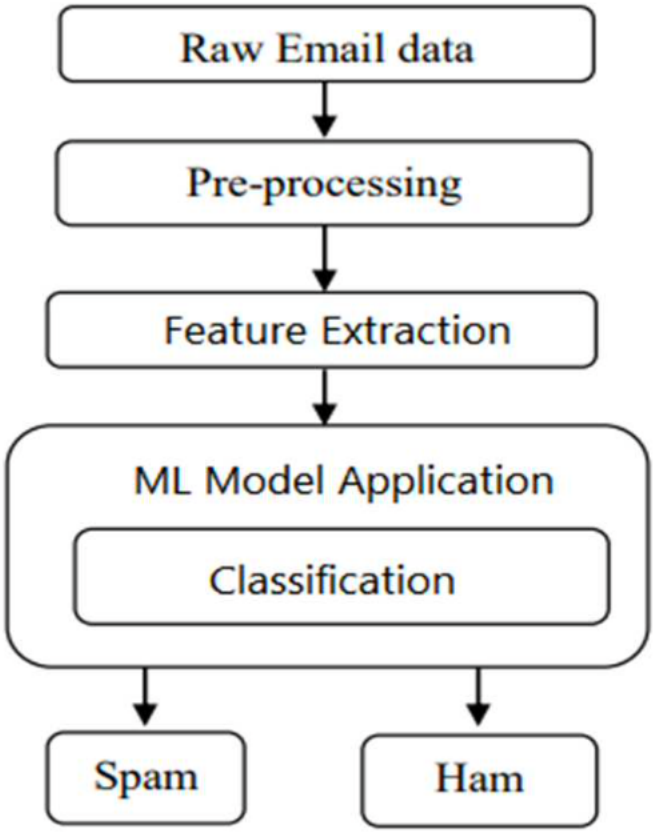

Figure: Methodology Adopted

1. Platform/System Setup Procedures

To ensure smooth execution of the notebook, the development environment is set up using Google Colab follow below mentioned setup steps:

i. Environment Setup in Google Colab

Installing the necessary dependencies for the different phases of the spam email filtering project is the first step. Important Python libraries are imported, including matplotlib (used

for data visualization), pandas (used for data manipulation and analysis), and scikit-learn (used for creating machine learning models). Preprocessing, cleaning, word embedding, visualization, UI integration, and prediction tasks all depend on these libraries. The configuration guarantees that the notebook is prepared for the next steps of data loading and model development.

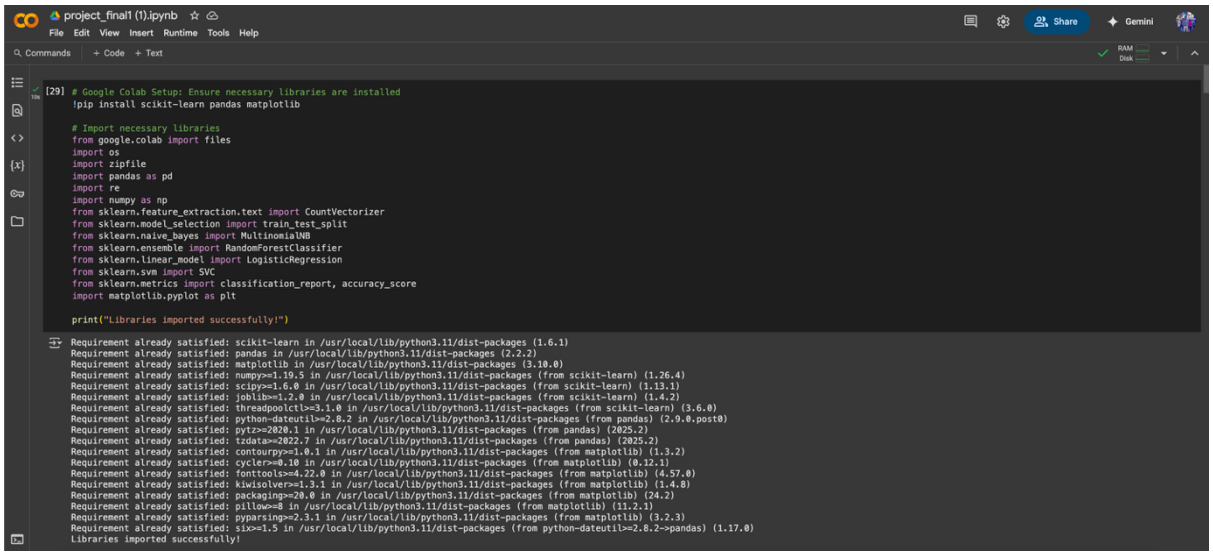

Figure: Screenshot of importing libraries

ii. File Preparation and Verification

First, the dataset file Combined_Spam_Ham.zip is uploaded to Google Colab's working directory. The contents are extracted into a folder called Combined_Spam_Ham when it has been uploaded and unzipped. Because the emails are encoded and not saved in the common.txt format, it is crucial to confirm that the files in the folder have been extracted correctly. In order to verify successful extraction, a sample file is loaded into the environment as part of the verification procedure. The extracted folder is shown in Colab's left panel, and the output window verifies that a raw email file was read, signifying that the setup and file preparation processes are finished.

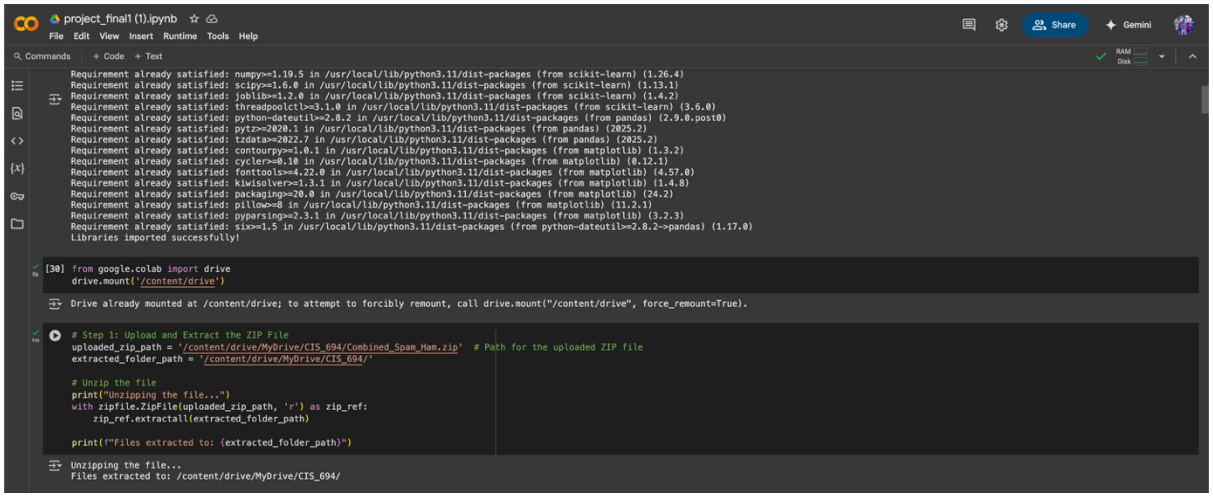

Figure: Screenshot of extraction of Raw files

2. Load Data and extract features

The raw email files in the Combined_Spam_Ham folder are put into a single DataFrame for analysis following the successful extraction of the dataset. In order to streamline the dataset for spam identification, a number of superfluous elements, including HTML tags, CSS content, and message IDs, were eliminated, as was illustrated in the class presentation. Extracting the most pertinent information from each email—the file name, sender, recipient, email subject, and body—was the main goal. The following phases of preprocessing and model creation are built upon these features that have been cleaned and organized.

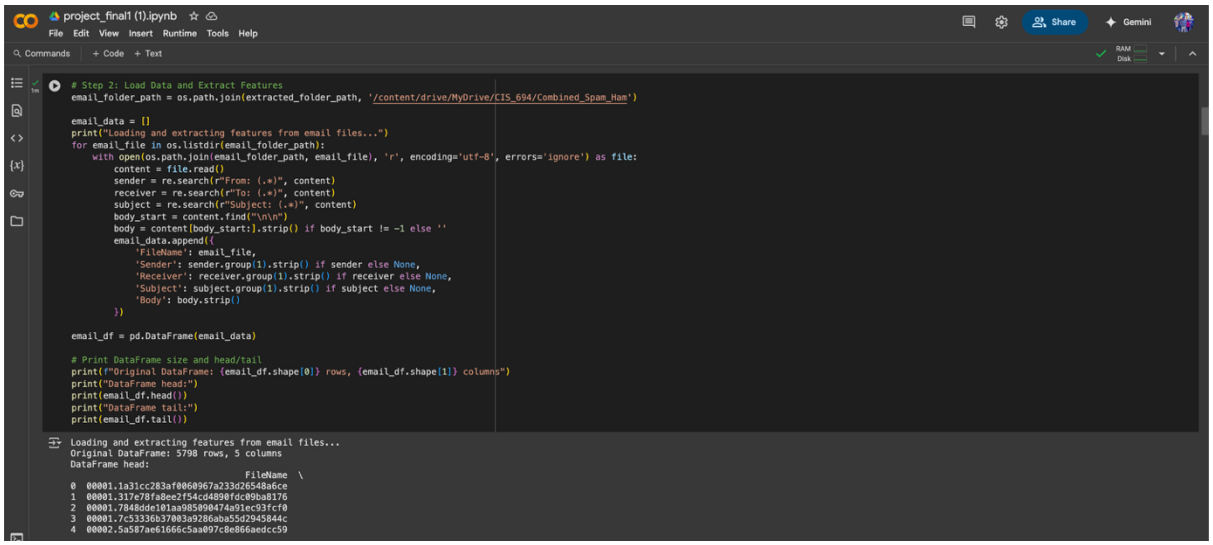

Figure: Load and extract features

3. Putting all the data into csv format for next process of data cleaning

The data was saved to a CSV file called email_features.csv for later usage once the important features were extracted and organized into a DataFrame. To ensure portability and accessibility, this CSV file is part of the project's zip folder. In order to verify that the structure is consistent and prepared for additional preprocessing and analysis, a screenshot was also taken to show the data types of each feature in the DataFrame.

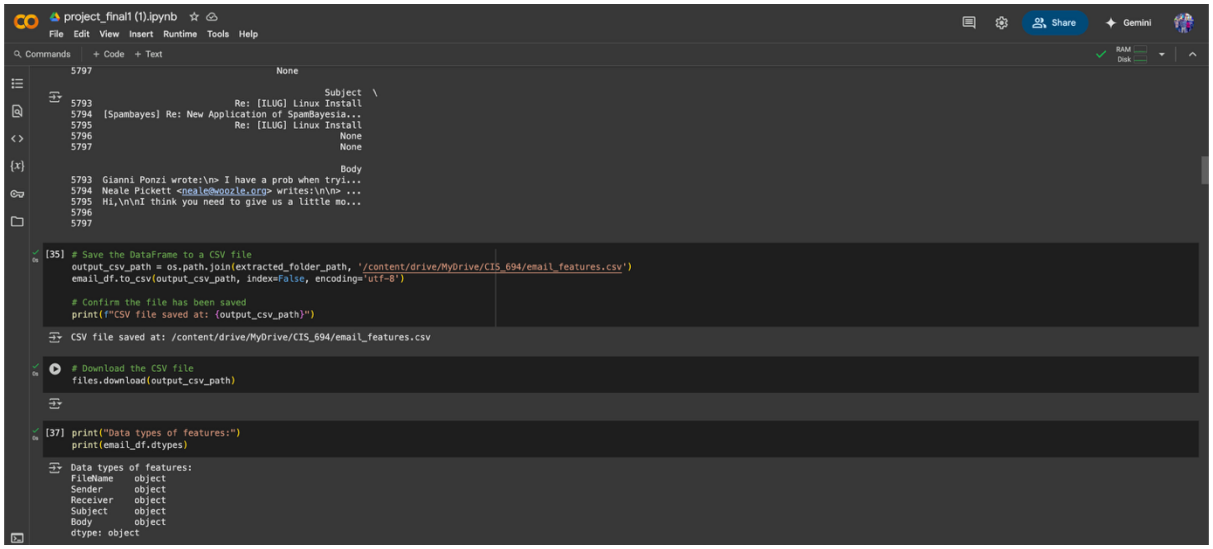

Figure: Converting data frame to CSV file

4. Removing all the unnecessary tags and creating cleaned body

A number of cleaning procedures were carried out in order to get the email body content ready for analysis. These included eliminating superfluous spaces, digits, URLs, HTML tags, and non-alphanumeric characters from the text. Additionally, the email body was cleared of superfluous or extraneous terms by employing the STOPWORDS function to remove common English stop words. Following cleaning, the refined text data was stored as preprocessed_email.csv, a new CSV file that is also included in the project's zip folder for future processing and reference.

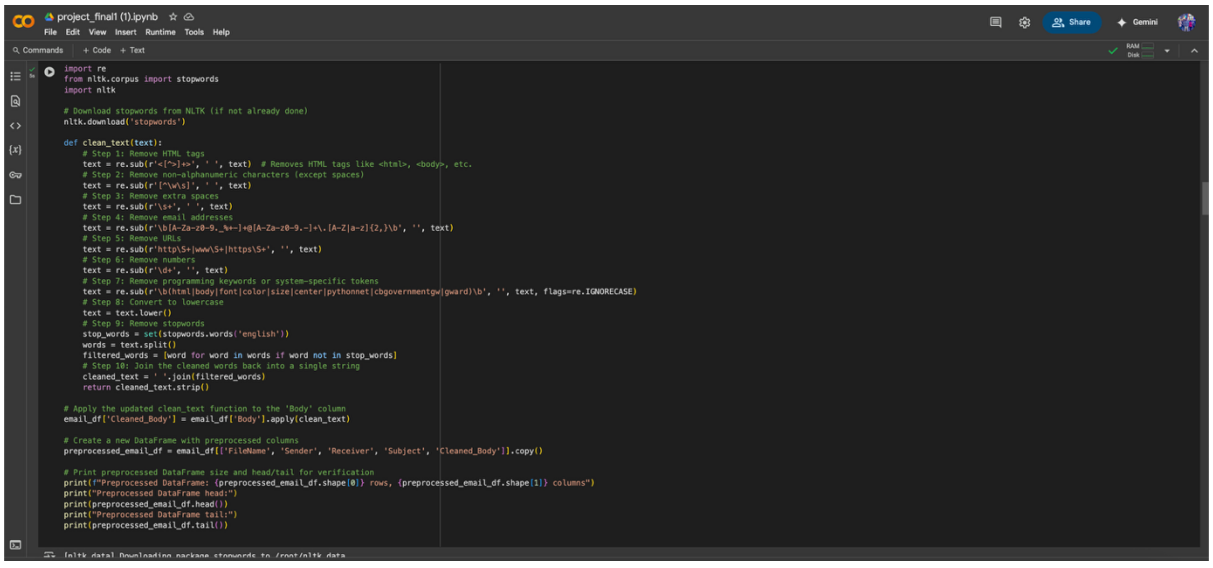

Figure: Cleaned body with features

5. Bootstrapping to create a larger dataset

The dataset's initial size of 6,000 emails was insufficient to create a reliable spam detection algorithm. The bootstrapping technique, which uses replacement sampling and random sampling from the available data to artificially expand the dataset size, was used to overcome this constraint. Consequently, the dataset was increased to 15,000 rows, which gave machine learning algorithms a bigger and more balanced input, improving model training efficacy and generalization.

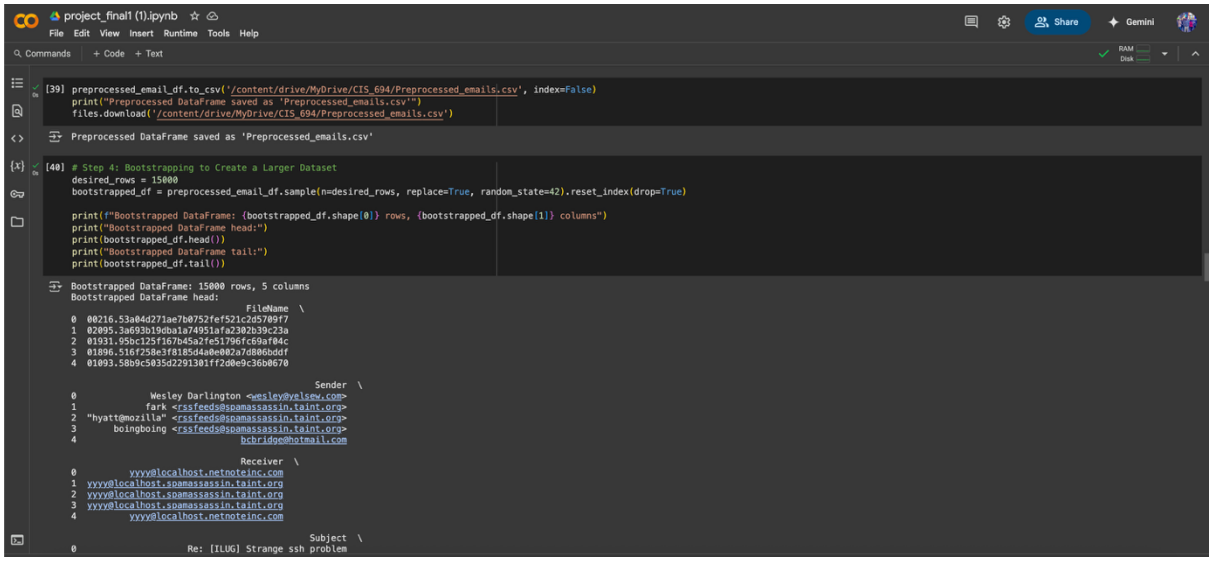

Figure: Bootstrapping to create larger dataset

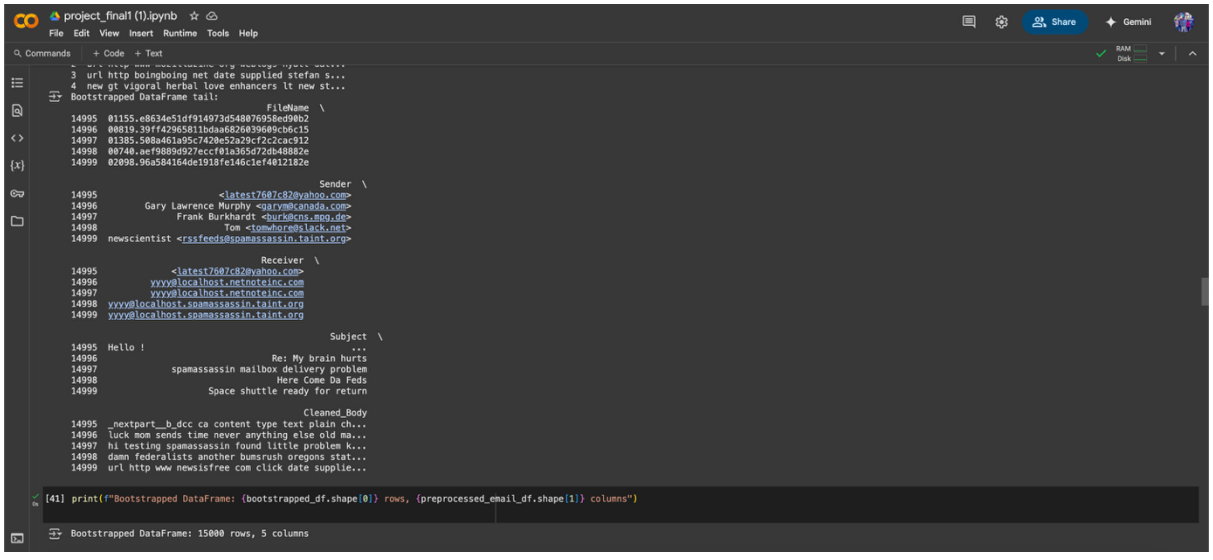

Figure: 15000 rows of data

6. Adding class label (spam/ham) based on spam keywords

Each email was given a binary class label, with 0 denoting ham (non-spam) and 1 denoting spam, in order to prepare the dataset for training a spam detection algorithm. In addition to a comprehensive manual assessment of spam and ham emails, search engine research was done to find frequently used keywords and indicators linked to spam content. These keywords formed the basis for creating a more precise and trustworthy model and assisted in informing the categorization process. The recognized spam-related phrases that help differentiate between spam and ham emails are displayed in a screenshot.

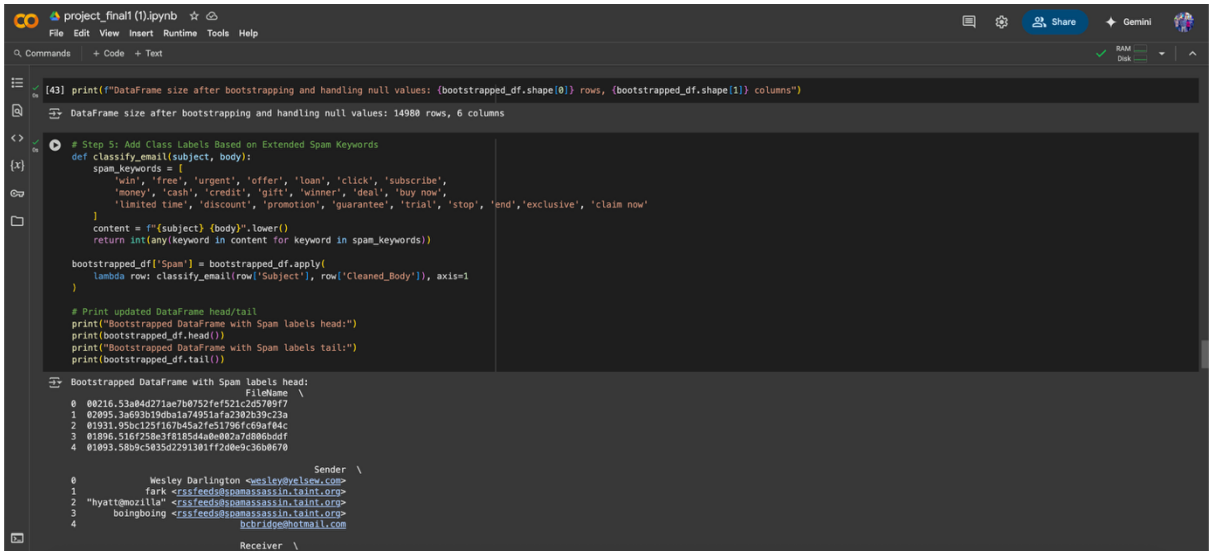

Figure: Adding class label on spam indicators

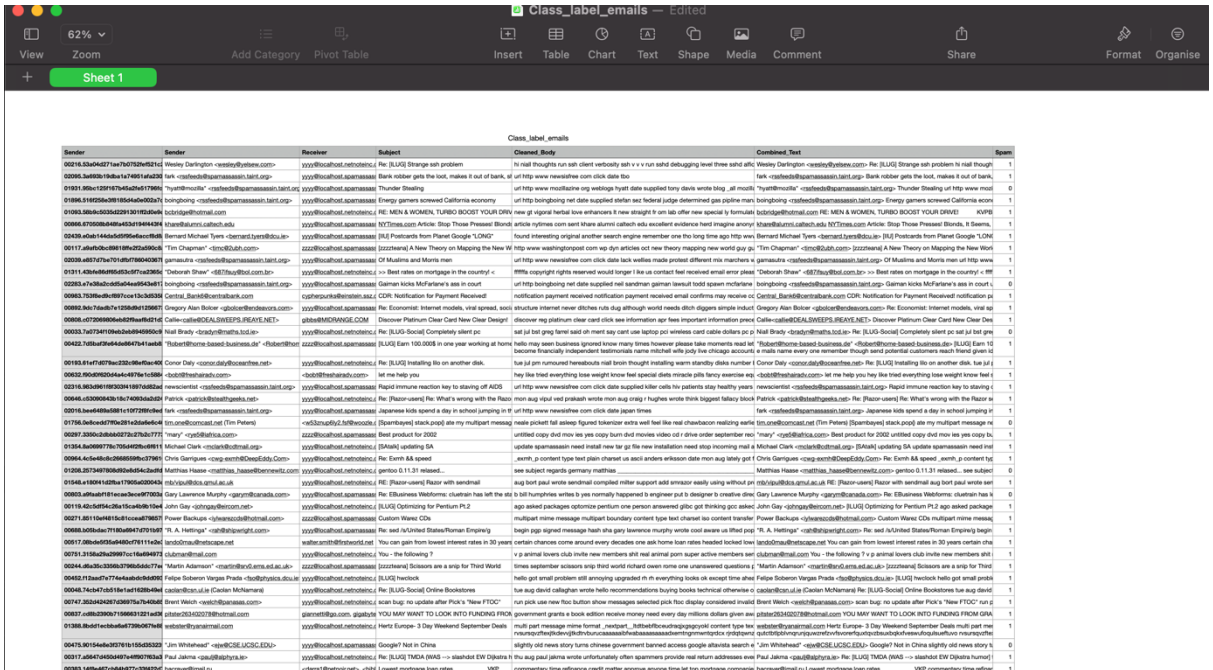

Figure: CSV file with class label on spam indicators

7. Word2Vec Model for Spam Email Detection

By capturing the semantic and contextual meaning of words, the Word2Vec model was used in this experiment to improve spam email identification. Three main benefits set Word2Vec apart from conventional keyword-based or bag-of-words strategies.

- **Semantic Understanding:** By rewording typical spam terms, spam emails frequently evade detection. The semantic similarity between these variations is captured by Word2Vec, which recognizes, for instance, that "Congratulations! You're the winner!" and "Well done! "You've won!" expresses the same idea.
- **Contextual Features:** Word2Vec creates word embeddings that comprehend the context in which concepts arise by taking into account the surrounding words. In sentences like "Limited time offer," for instance, the model recognizes that the word "offer" is spam-indicative when combined with specific nouns.
- **Dimensionality Reduction:** By transforming words into low-dimensional dense vectors, Word2Vec preserves substantial linguistic patterns while drastically lowering memory consumption and processing costs as compared to sparse representations like TF-IDF.

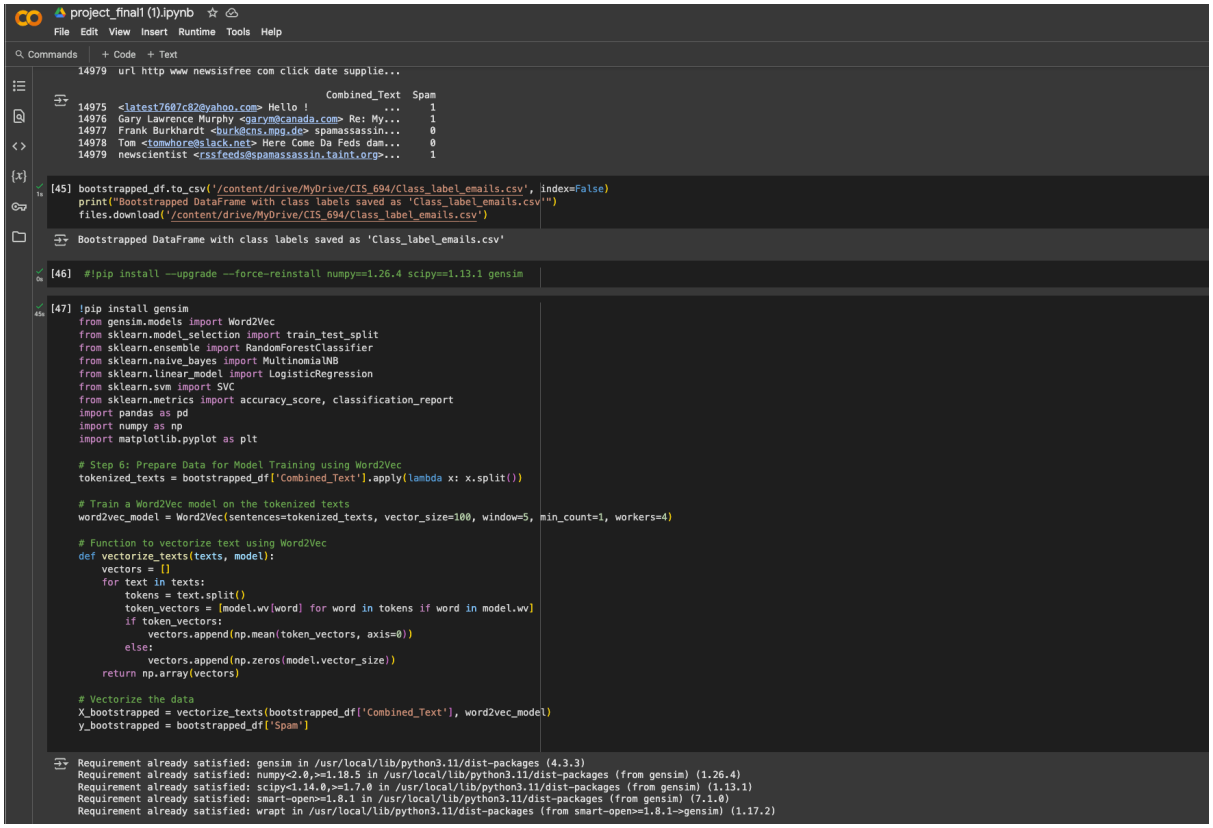

Figure: Word2Vec model for word embedding to predict spam

8. Training and Testing

To build and evaluate the spam detection model effectively, the dataset was split into two subsets: 70% for training and 30% for testing. The training set is used to teach the model underlying patterns and associations in the labelled data, while the testing set is used to measure how well the model performs on previously unseen emails. This split maintains the overall distribution of the original dataset to ensure fairness and accuracy in evaluation. Additionally, a random state parameter was applied during splitting to ensure reproducibility—without it, each run would produce a different data split, potentially affecting the consistency of results.

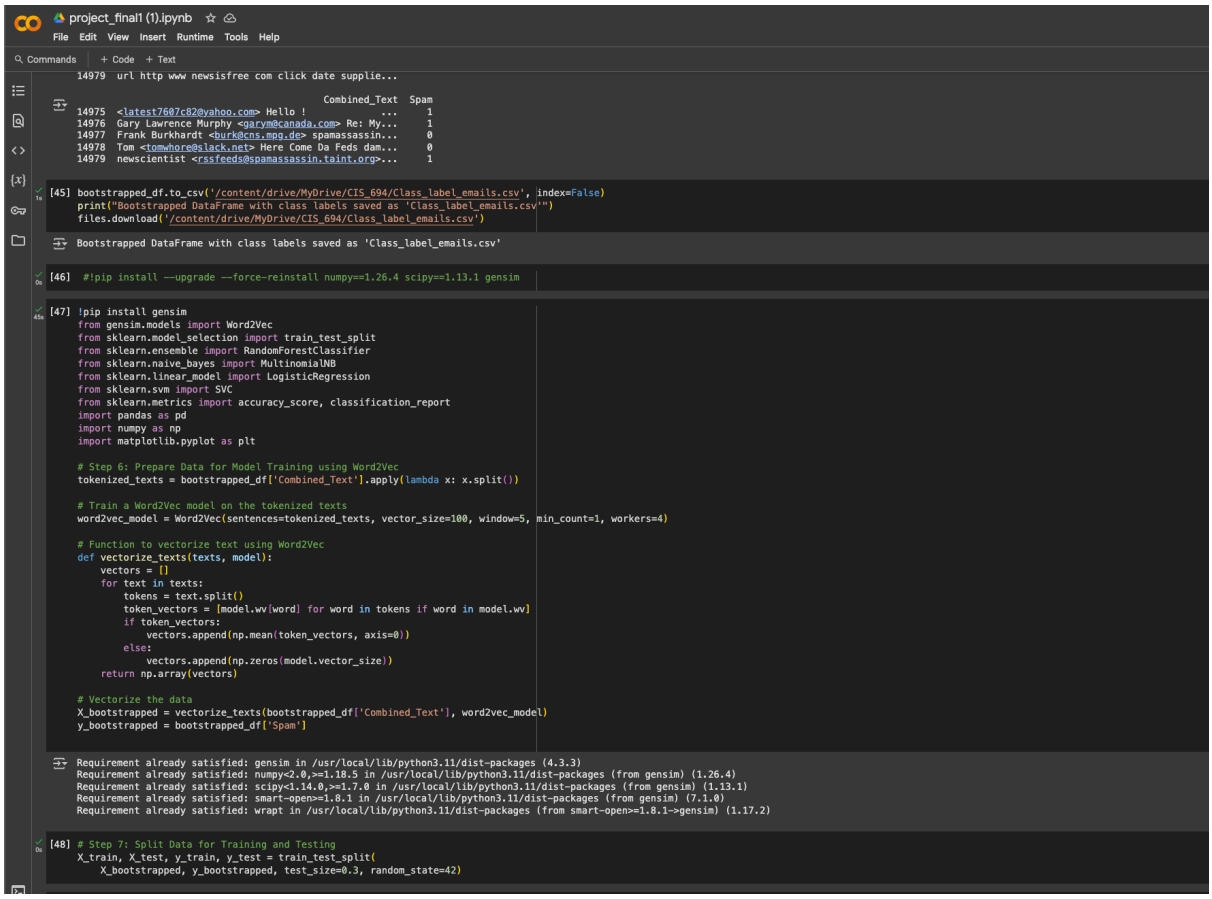

Figure: Split data for training and testing

Result and Discussion

1. Model Training and Evaluation

Three machine learning models—Support Vector Machine (SVM), Logistic Regression, and Random Forest—were used to classify spam emails. These models were chosen because they work well with text data and are compatible with Word2Vec and other word embeddings. To identify the patterns that differentiate spam from non-spam emails, each model was trained using the processed dataset. Then, criteria including recall, accuracy, and precision were used to assess their performance. A screenshot demonstrating the use of all three models and demonstrating how they were used to successfully forecast email categories is given.

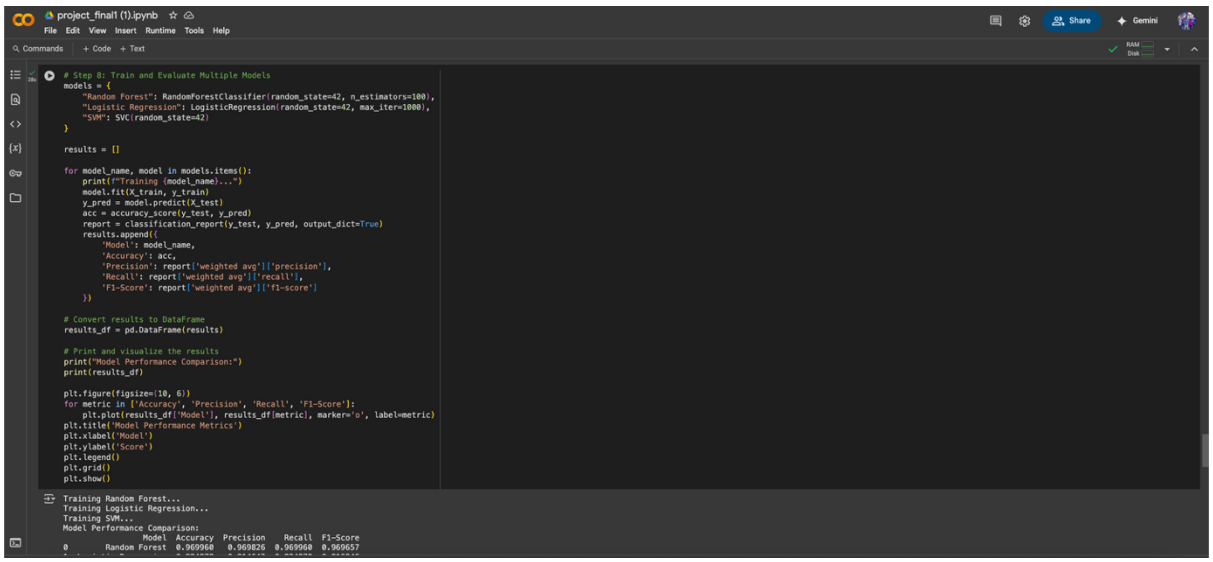

Figure: Evaluation major for spam email detection

2. Model Performance Visualization

Visualizations of the accuracy, precision, recall, and F1 score of the three trained models—SVM, Logistic Regression, and Random Forest—were made in order to compare their efficacy. The capacity of each model to accurately distinguish between spam and non-spam emails was thoroughly assessed by these performance criteria. The best-performing model was chosen for deployment with the aid of the visual comparison. The final spam detection model was selected based on its balanced metric scores and overall accuracy. A clean and succinct screenshot shows how well each of the three models performs in comparison.

Based on the outcomes displayed in the screenshot:

- With roughly 96.96% accuracy, precision, recall, and F1-score, Random Forest performed best across all measures, demonstrating its consistent and accurate email classification.
- With an accuracy of about 82.47%, logistic regression performed moderately well and was less dependable than random forest.
- With an accuracy of 84.69%, SVM outperformed Logistic Regression by a small margin, but Random Forest was still far superior.

Each model's performance across the four metrics is shown in detail in the line graph. The models are shown on the x-axis, and the metric score (which ranges from 0.80 to 0.97) is shown on the y-axis. Every coloured line represents a distinct metric:

- Blue for Precision
- Orange for Accuracy
- Green to Help You Remember
- F1-Score in red

The peak at the leftmost position of the graph indicates that Random Forest routinely receives the highest scores across all metrics. SVM performs marginally better than Random Forest but is still not competitive, while Logistic Regression has the lowest performance. The best model for this spam detection task is Random Forest, as can be easily seen from this visualization.

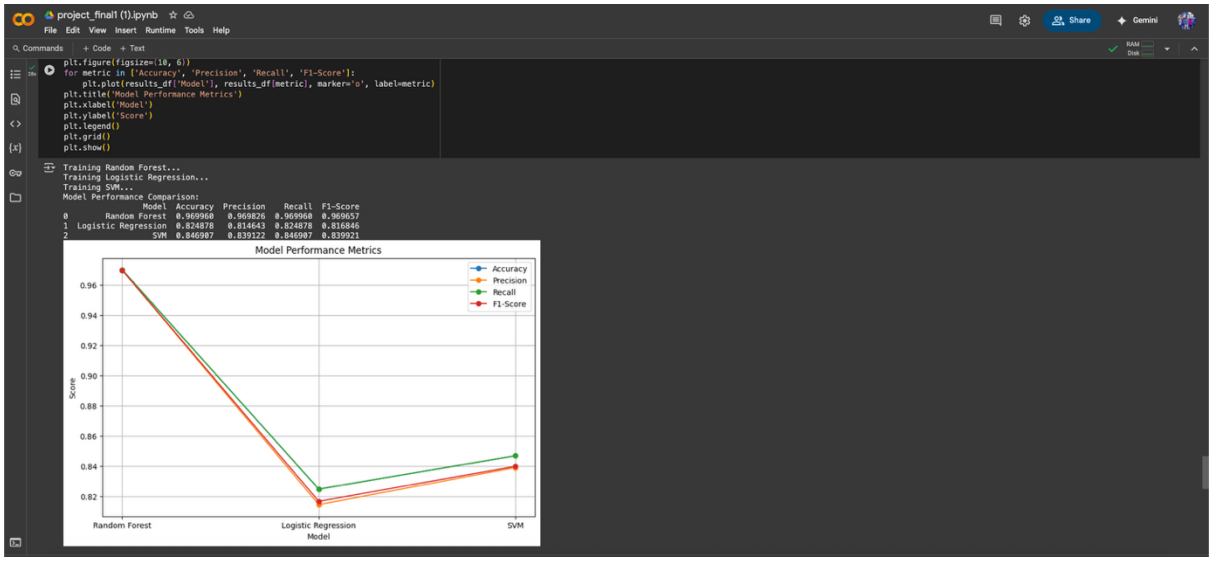

Figure: Visualization of all three models with accuracy, precision, F1 and recall

3. Future Predictions and Basic User Interface

A simple User Interface (UI) for determining if an email is spam or not was created to show practical application. Users can enter new email text into this interface and get immediate classification results using the trained model. To generate future predictions on unknown data, it incorporates the model's prediction logic. Designed for testing and display, the design is simple and practical. The underlying user interface programming that drives this interactive spam classification capability is displayed in two screenshots.

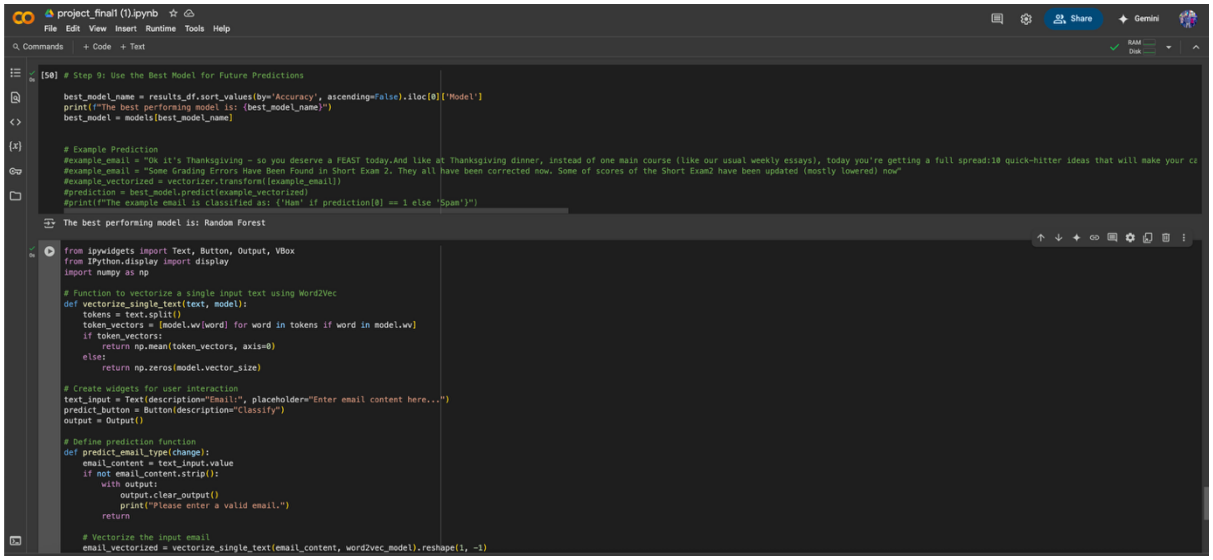

Figure: UI code for spam classification

As part of the project, a basic user interface was created to show off the spam detection system's usefulness. Users can input a sample email into this interface's input box, and the trained model will use the content to determine whether the email is categorized as "spam" or "ham" (non-spam). Users can evaluate the efficacy of the spam filter on custom email text using this interactive feature, which demonstrates how the model can be used in real-time applications.

The Input Email: Your refund is on its way Hi Jayanth, Sorry things didn’t work out. We’ve initiated a refund and wanted to provide a quick overview for you. If you have any follow-up questions, please visit our Help Center. Refund overview Refund Initiated card image AMEX ending in 1006$19.91 Credited within 10 business days Thanks for shopping with us, Team Walmart Order number 2000130-07764860 Refund initiated Sold and shipped by Walmart item image Jif Creamy Peanut Butter, 16-Ounce Jar Qty: 1 $3.12 item image Great Value 100% Whole Wheat Round Top Bread, 20 oz Qty: 1 $1.97 item image Fresh Roma Tomato, Each $1.65/LB 1.693LB $1.65 item image Fresh Whole Red Onion, Each $3.69/LB 2.078LB $3.69 item image Cetaphil Daily Facial Cleanser for Sensitive, Combination to Oily Skin, 8 oz Qty: 1 $8.78 Refund summary Subtotal $19.21 Taxes $0.70 Refund total $19.91

Output: Ham

Figure: Email Classification using best model

4. Problems Encountered and Resolutions

A number of difficulties were encountered and successfully overcome throughout the spam detection system's development:

Error 1: Problems with File Encoding
Issue: When reading some email files, encoding issues occurred.
Resolution: To get around troublesome characters, use errors='ignore' in the open() method.

Error 2: Difficulties with Data Parsing
Issue: Structured headers like From, To, and Subject were missing from a lot of email files.
Resolution: To extract data even in cases when headers were missing, regular expressions with fallback techniques were used.

Error 3: Problems with CSV Files Following Bootstrapping: Several records had null values added by bootstrapped data.
Resolution: Prior to additional processing, null values were managed programmatically using data cleaning techniques.

```
✓ # Handle missing values by removing rows with null values
✓ print("Handling missing values by removing rows with null values...")
✓
✓ # Check for any null values in required columns
✓ print("Before dropping null values:")
✓ print(bootstrapped_df.isnull().sum())
✓
```

Error 4: TF-IDF and Other Vectorizers' Limitations
Issue: Inconsistencies were produced by TF-IDF and related techniques because of improper word weighting, which resulted in incorrect classifications.
Resolution: To improve performance and interpretability, manually recognized spam indicators were used in place of TF-IDF.

Error 5: Class Imbalance in Dataset
Issue: There were disproportionately high and low amounts of spam and ham emails in the dataset.
Resolution: To preserve class distribution in both training and testing datasets, stratified sampling was used during the train_test_split.

Conclusion

The goal of this research was to create a reliable spam email detection system by combining machine learning models, semantic word embedding, and data preparation. After extracting significant features from unprocessed, encoded email files, the dataset was cleaned to eliminate extraneous components such as HTML tags, special characters, and stop words. As a result, just the most crucial elements—such as the sender, subject, and body—were kept for examination.

Stratified sampling was utilized to guarantee a balanced class distribution during model training and testing, and bootstrapping techniques were employed to increase the data from 6,000 to 15,000 samples in order to overcome dataset restrictions. In order to capture the semantic relationships between words and their context within the email content, Word2Vec was selected above more conventional techniques like TF-IDF.

Key performance parameters like accuracy, precision, recall, and F1 score were used to train and assess three machine learning models: Random Forest, SVM, and Logistic Regression. Finding the best model for deployment was made easier by the visualization of these findings. To enable users to test new email content for spam prediction, a basic user interface was also put into place.

Multiple technical problems, such as inconsistent email formats, file encoding mistakes, and data sparsity issues, were encountered and fixed during the project. By overcoming these obstacles, the study not only succeeded in detecting spam but also showed how well machine learning and natural language processing can be combined for practical uses.

References

[1] C. Yang, R. Harkreader, and G. Gu, "Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers", In: Recent Advances in Intrusion Detection, Springer Berlin/Heidelberg, pp.318-337, 2011.

[2] S. Kumar, and S. Arumugam, "A Probabilistic Neural Network Based Classification of Spam Mails Using Particle Swarm Optimization Feature Selection", Middle-East Journal of Scientific Research, Vol.23, No.5, pp.874-879, 2015.

[3] N. P. DíAz, D. R. OrdáS, F. F. Riverola, and J. R. MéNdez, "SDAI: An integral evaluation methodology for content-based spam filtering models", Expert Systems with Applications, Vol.39, No.16, pp.12487-12500, 2012.

[4] A. K. Sharma, S. K. Prajapat, and M. Aslam, "A Comparative Study Between Naive Bayes and Neural Network (MLP) Classifier for Spam Email Detection", In IJCA Proceedings on National Seminar on Recent Advances in Wireless Networks and Communications. Foundation of Computer Science (FCS), pp.12- 16, 2014.

[5] W. Ma, D. Tran, and D. Sharma, "A novel spam email detection system based on negative selection", In: Proc. of Fourth International Conference on Computer Sciences and Convergence Information Technology, ICCIT'09, Seoul, Korea, pp.987-992, 2009.

[6] T. S. Guzella, and W. M. Caminhas, "A review of machine learning approaches to spam filtering", Expert Systems with Applications, Vol.36, No.7, pp.10206-10222, 2009.

[7] N. Kumar, S. Sonowal, and Nishant, “Email spam detection using machine learning algorithms,” in Proceedings of the 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 108–113, IEEE, Coimbatore, India, July 2020.

[8] G. Jain, M. Sharma, and B. Agarwal, “Optimizing semantic lstm for spam detection,” International Journal of Information Technology, vol. 11, no. 2, pp. 239–250, 2019.

[9] F Masood, G. Ammad, A. Almogren et al., “Spammer detection and fake user identification on social networks,” IEEE Access, vol. 7, pp. 68140–68152, 2019.

[10] G. Chandrashekar, and F. Sahin, "A survey on feature selection methods", Computers & Electrical Engineering, Vol.40, No.1, pp.16-28, 2014.

[11] M. Mohamad, and A. Selamat, "An evaluation on the efficiency of hybrid feature selection in spam email classification", In: Proc. of 2015 International Conference on Computer, Communications, and Control Technology (I4CT), Kuching, Sarawak, Malaysia, pp.227- 231, 2015.

[12] M. Sahami, S. Dumais, D. Heckerman, E. Horvitz: A Bayesian approach to filtering junk e-mail. In: AAAI-98 Workshop on Learning for Text Categorization. 1998. Volume 460.