User classification using Enron email dataset

P21: Krishna Chaithanya Marripati, Mery Harika Gaddam, Sai Vikas Reddy Yeddulamala
Department of Computer Science, North Carolina State University
Raleigh, NC 27695
kmarrip, mgaddam, syeddul@ncsu.edu

1 Background:

1.1 Enron email classification

The Enron email dataset contains 500,000 emails from 150 users, the dataset contains emails sent to and from other employees of Enron. Email metadata contains data related to attachments, deleted emails, junk emails, calendar items, and other discussion threads. Enron is an American corporation involved in accounting fraud, which resulted in billions of dollars of loss to its investors. Since then the data set containing the confidential emails has been open-sourced for the data scientists to find patterns among the emails sent by Enron executives.

1.2 Problem Statement:

The foundation of email's safety and trustworthiness rests on its unique ability to easily verify the authenticity of an email address, making it a formidable challenge for malicious attempts at spoofing. This verification process adds a layer of security, ensuring that users can have confidence in the legitimacy of their communication channels. (1) Moreover, email's versatility extends beyond individual interactions to encompass mass communication. Organizations and entities leverage this medium to efficiently relay information to targeted audiences, making it a highly effective tool for making announcements, updates, and crucial messages.

Analyzing the Enron Email Dataset involves identifying unique patterns in writing styles, consistent grammatical nuances, and overall sentiment. The project aims to accurately predict email authors by leveraging these distinct patterns. By focusing on individualized features, such as writing habits and emotional tone, the goal is to build a precise predictive model for authorship attribution within the dataset. This project contributes to the broader field of authorship attribution and pattern recognition in textual data.

1.3 Literature Survey:

Based on an extensive review of existing literature, prominent techniques emerge for addressing the challenges in user classification from the Enron Email Dataset. One key approach involves the utilization of (2) TF-IDF (Term Frequency-Inverse Document Frequency) for robust feature extraction. Support Vector Machines (SVM) and Random Forests have proven to be effective tools for unveiling hidden data patterns in the emails.SVM is a supervised learning algorithm that excels in classification tasks.Random Forests, on the other hand, leverage the strength of ensemble learning by combining multiple decision trees to enhance accuracy. (3) The computational complexity scales well with the desired expressiveness, and the underlying principles can be applied flexibly to various problems. Overall, it provides a versatile and efficient means of interpreting neural network behavior.

2 Proposed method:

2.1 Approach and Intuition:

The primary intuition behind choosing the machine learning models is that, there is a definitive one-to-one mapping with the person who writes the email, with their grammatical structure, lexicon,

vocabulary, and length of the email body. We exploit these features to predict the sender. We convert these abstract concepts to machine readable values using below mentioned feature extraction techniques.

2.2 Description of its algorithms and Rationale:

The rationale for choosing the below algorithms is:

Random forest is an ensemble learning algorithm that builds on multiple decision tress, which makes it highly accurate and handles high dimensional data. For MNB, while the assumption of feature independence might not hold in reality, it often works well in practice for text classification. This makes MNB a robust choice for tasks where the presence of certain words contributes to the classification independently of others. In scenarios where the relationship between features and classes is approximately linear, SVM with a linear kernel is a solid choice for building accurate and interpretable text classification models. LSTMs are recurrent neural networks which are really good at finding data patterns in input data, while it might be computationally expensive, with enough data and hyper parameter tuning, they yield exceptional results in text classification.

2.2.1 Multinomial Naïve Bayes:

Multinomial Naïve Bayes assumes that the features used to describe an instance are conditionally independent, given the class. In the context of text classification, these features often represent word counts or frequencies. The algorithm calculates the probability of a document belonging to a particular class by considering the frequency of each word in the document and the overall probability of each class. It is widely used for tasks such as spam filtering and document categorization.

We have used Min-Max scalar on the selected columns, as it helps transform data by scaling features in the range 0 to 1. Then we have performed label encoding on the employees column to convert categorical data into binary numeric data. We split the data into features and target to train the model. We have performed stratified K-fold by selecting the number of splits into 10. Then we have trained the model using both the feature sets by splitting the data set into training and testing data and selecting the best performing feature set with the help of the accuracy score obtained.

2.2.2 Random Forest trees:

Random forest are really good for data with bias, it's difficult to over-fit them. Random Forest, when applied to text classification, functions as a collaborative ensemble of decision trees. This method involves the transformation of textual information into numerical representations, allowing for comprehensive analysis. Multiple decision trees are then constructed, each focused on distinct aspects of the text data, collectively contributing to the classification process. The combination of individual tree predictions yields the final decision. Advantages of using Random Forest includes its adept handling of extensive vocabulary, resilience against data intricacies, and the capacity to discern the significance of specific words in the classification task.

2.2.3 Support Vector Machine:

SVM with a linear kernel uses a straight line to separate different groups based on their features. It's like drawing a line on a graph to classify things. The goal is to find the best line (hyperplane) that separates the groups as much as possible. This works well when the relationship between features and groups is simple and straight.

2.2.4 Long-short term memory [LSTM]:

LSTM model is a type of recurrent neural network that can learn and remember pattern over long sequences of data, making it useful for NLP tasks. Here we selected the maximum number of words as 10,000 and maximum length of a sequence as 500. Since machine learning algorithms accepts input in numerical data, we performed word embedding using tokenizer on each email's body. After tokenizing the text, we have trained the model by first splitting the data into 80% training data and 20% testing data. While training the model, we have again split the data into 80% as training and 20% as validation data. Then selected the number of epochs as 10 and the batch size as 128. Later, we have evaluated using the test data and calculated the accuracy.

3 Plan and Experiment:

3.1 Dataset(s):

3.1.1 Description of the data set:

The Enron email dataset contains 500,000 emails from 150 users, the dataset contains emails sent to and from other employees of Enron. Email metadata contains data related to attachments, deleted emails, junk emails, calendar items, and other discussion threads.Here are the top 10 email senders from the dataset.

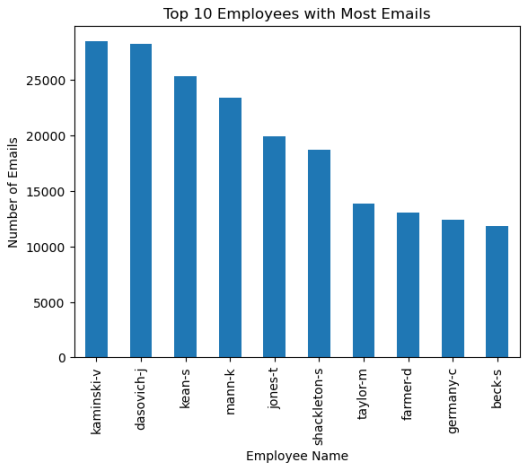


Figure 1: Top 10 Email senders

3.1.2 Email Parser:

Since the data set consists of emails, we first wrote a utility.py file, after that, this file is built using the email parser library, which helps us to extract the features like "from", "to", "subject", "body of the email", etc. Then we convert the whole data set file consisting of all emails from different users into a CSV file.

3.1.3 Data cleaning:

We've streamlined the dataset for enhanced accessibility and clarity. First, we converted the text file to a CSV format, simplifying data access. Additionally, we addressed missing values for data completeness. To refine the dataset further, we identified and removed duplicate forwarded messages, focusing on preserving significant content. In terms of encoding consistency, all non UTF-8 files were either removed or converted to the closest UTF-8 characters, ensuring uniform encoding throughout.

Furthermore, to optimize the training data, we eliminated duplicate emails originating from internal communications among users. This step enhances the significance of the dataset by retaining essential information while discarding redundant content, particularly in scenarios where both sender and receiver convey identical messages. These preprocessing steps collectively contribute to a more refined and meaningful dataset for subsequent analysis and modeling.

3.1.4 Data pre-processing:

Now starting lowercasing the text data present in all the columns so that it becomes easier for the model to understand. Third, remove all the punctuation marks present in the columns using the 'regex' library for regular expressions.

We remove the stop words from all the columns by using the NLTK library, which is mainly used to remove all the most commonly occurring words in a text, which will not have any valuable information. This NLTK library approximately includes 180 stop-words, which helps us remove all the unnecessary information.

Stemming is used to reduce the word to its root stem, for example, 'jog', 'jogging', 'jogged', 'jogger', and 'to jog', are derived from the same root word 'jog'

3.1.5 Feature Extraction:

We employ various data pre-processing techniques to feed the data model with processed data. Using feature extraction techniques like TF-IDF, count vectorizers emails are converted to machine-readable numbers. Different data models that can be used here are SVM, Random forests, and other NLP models. User classification using Enron email involves finding patterns among the emails sent by a user. One user might use a set of words more frequently than others, there could be a way one user might use a set of words grammatically. These machine-learning models help us retrieve these hidden patterns and can be used to extrapolate and predict the user who might have drafted this email.
Feature extraction needs to be done, because the content and structure in the emails are different from that of normal documents. So there needs to be a lot of data that should be analyzed for capturing unique writing styles and differentiating authors.
Feature selection is done by using TF-IDF vectorizer which compares the number of times a word appears in document with the number of documents the word appears in. And Count vectorizer vectorizes the textual data into numeric data based on number of times a word is appeared in a sentence. Stratified sampling is done to divide the email data into different subgroups. Semantic features describe the tone of the same message that needs to be conveyed by different authors. It can be extracted from the emails using polarity score. The tone of the message is analyzed by using text blob and Vader sentiment analysis methods to determine the polarity score and to categorize the message into positive, negative or neutral. Text blob classifies the text using NLTK based predefined lexicon and gives polarity associated with a word. VADER sentiment analysis uses rule-based approach combining a predefined lexicon and grammatical rules to assess the sentiment

3.1.6 Sentiment analysis:

After text pre-processing, we performed sentiment analysis on the cleaned data set. For sentiment analysis, we used two methods, mainly text blob and VADER (Valence Aware Dictionary and sentiment Reasoner). Text blob uses a polarity-based system ranging values from -1 to +1 for each word, for example, 'scary' can have a value of -0.4, and 'happy' can have a value of 0.5, the overall sentiment of the text block is calculated as the sum of individual words and then they are classified as positive, negative and neutral. VADER uses sentiment lexicon, grammar rules, and exclamation points which help to identify the sentiment of the text block. VADER has four metrics positive, negative, neutral, and compound score. Here, in our data set, we will be using a compound score. The scores range from -1 to +1. We performed tokenization on every "body of the email" in the data set to find out the number of words in the email. This feature is used by passing it to the model for better performance.

3.1.7 TF-IDF vectorization:

We have performed TF-IDF vectorization to understand the significance and number of occurrences a word has across all the emails which makes the emails easily distinguishable based on the value. The less the number, the easier it is to classify the emails based on the relevance of the word searched by the user.

3.1.8 Count vectorization:

We have performed count vectorization to represent a document as a numerical vector based on the frequency of each word present in the document. In this approach, each document is transformed into a vector where each element corresponds to the count of a specific word in the document.

3.2 Hypothesis:

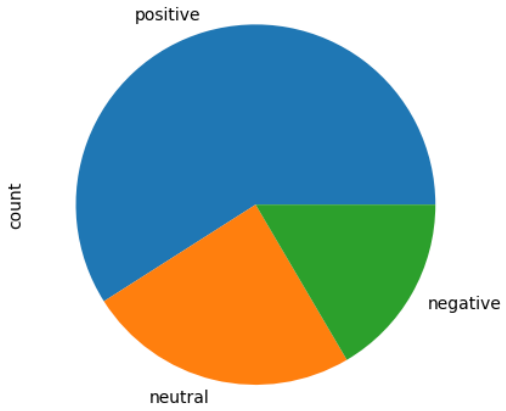
The central hypothesis of this project posits a robust connection between the writing style, grammatical usage, and vocabulary found in emails and the individual responsible for drafting them. The contention is that these linguistic patterns exhibit a distinct and singular mapping onto each person, suggesting a definitive one-to-one correspondence between the identified data patterns and the respective authors.

3.3 Experimental Design:

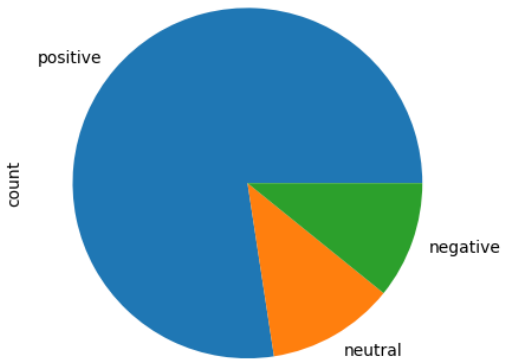
To test the hypothesis of a one-to-one mapping between writing style in emails and individuals, follow these steps: collect diverse email data, extract key writing features, preprocess and engineer features, choose a suitable model, train and test it, validate performance, use appropriate evaluation metrics, analyze results for feature importance, and iterate for improvement.The Enron email data set is to be collected, considering the user's privacy. Enron email dataset consists of 500,000 emails from 250 different users, each user's email inbox contains emails from various folders like sent email, primary inbox, junk folder and spam folder. The data once converted in python pandas dataframe is then used in in various machine learning models employed here like Multinomial Naive bayes, Linear SVC, LSTM and Random forests.

4 Results:

Performed text pre-processing and made sure that data is readily available for sentiment analysis. Performed EDA to get insights on different data patterns, like analyzing top 10 recipients. Performed sentiment analysis using text blob and VADER sentiment and categorized them into positive, negative, and neutral. The polarity values for the text blob are 60% positive, 18% negative and 22% neutral. The results of sentiment analysis using VADER sentiment are 75% positive, 12% negative, and 13% neutral.



(a) Results of sentimental analysis using Textblob

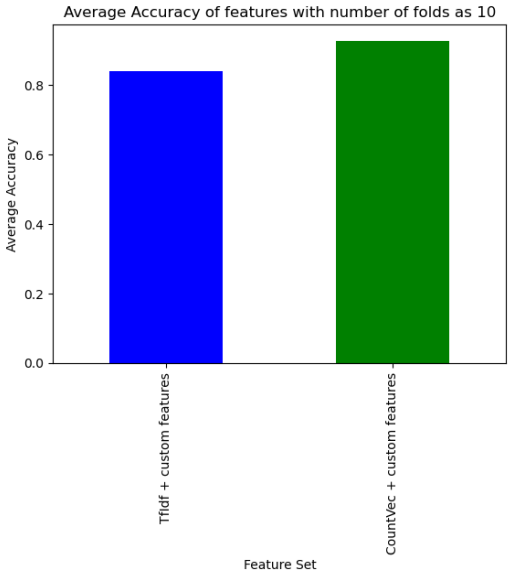


(b) Results of sentimental analysis using Vader

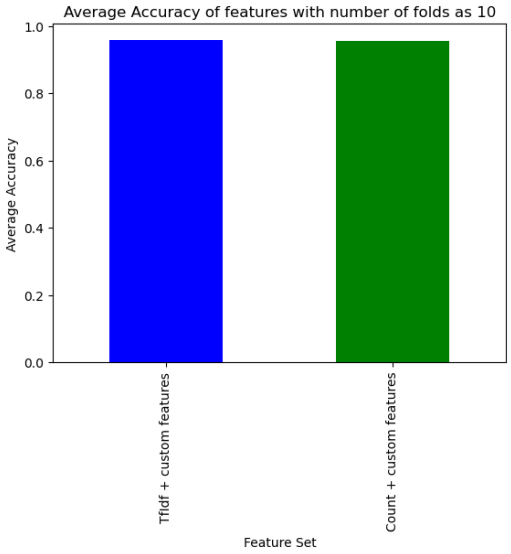
We have selected 3 baseline models and 1 neural network model to build the sender classifier. We have selected the number of splits as 10 to perform cross validation. Below are the results obtained for all the models that we have used.

Multinomial Naive Bayes model

The TFIDF-based feature set achieved an accuracy of 84.012%, whereas the Count Vectorization-based feature set outperformed with an accuracy of 92.70%. This difference was visually evident in the accompanying plot. The results show the MNB model's preference for Count Vectorization over TFIDF, especially compared to the Random Forest classifier. Count Vectorization's higher accuracy suggests its effectiveness in this text classification scenario. Overall, Count Vectorization aligned better with the MNB model, underscoring the need to tailor feature engineering techniques to specific models and datasets.



(a) Accuracy of Multinomial Naive Bayes Model



(b) Accuracy of Random Forest Classifier

Random Forest Classifier

The first feature set using TFIDF vectorization achieved an average accuracy of about 95.78%, while the second set with Count Vectorization had a slightly lower accuracy of 95.61%. The plot shows a negligible difference in performance between the two methods. Both techniques, combined with custom features, resulted in robust feature sets for the Random Forest classifier, with TFIDF having a slight advantage. This highlights the effectiveness of ensemble methods like Random Forest in text analysis and the importance of feature engineering. The similar performance metrics suggest that the choice between TFIDF and Count Vectorization may depend on factors like model complexity, computational efficiency, or dataset specifics.

Linear SVC

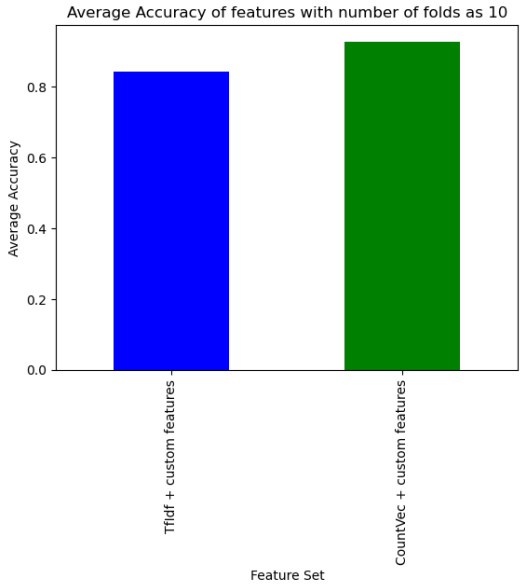


Figure 4: Accuracy of LinearSVC model

The comparison between two feature sets for the Linear SVC model showed distinct results. The TFIDF vectorized feature set achieved an accuracy of 84.36%, while the Count Vectorized set outperformed it with an accuracy of 92.891%. A plot visually highlighted these differences. The higher performance of the Count Vectorization indicates that term frequency is more crucial for classification in Linear SVC than term specificity and inverse document frequency, as suggested by the lower accuracy with the TFIDF set. This highlights the significance of choosing the right feature set in machine learning, with Count Vectorization aligning better with the Linear SVC model for more accurate results.

LSTM Model

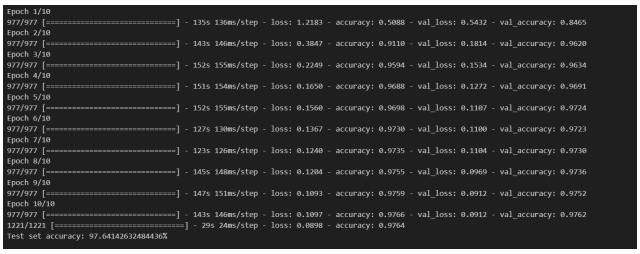


Figure 5: Training of LSTM model

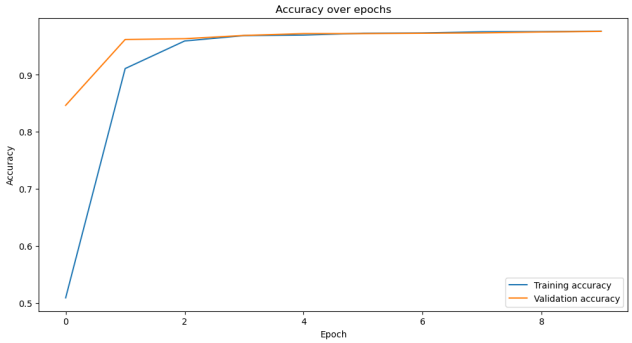


Figure 6: Training and Validation accuracies while training LSTM model

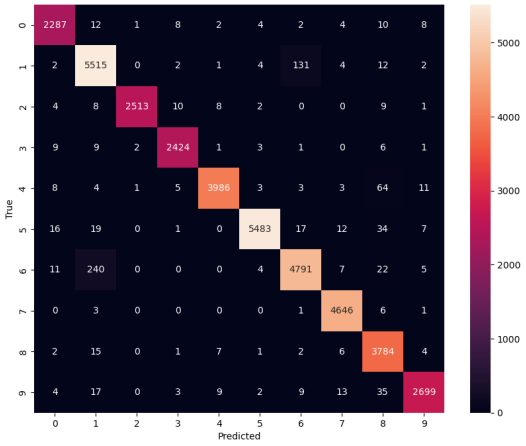


Figure 7: Confusion Matrix of LSTM model

An LSTM classifier was constructed and trained over 10 epochs, utilizing 32 LSTM units and a neural network of the same size, with a batch size set at 128. The training logs reveal a substantial improvement from the initial epoch to the last, with the model's accuracy starting at approximately 50.88% and finishing at 97.64%. The validation accuracy also saw significant enhancement, initiating at 84.65% and culminating at 97.62%. The accuracy curve graph indicates that the model achieved a stable convergence, with the validation accuracy closely tracking the training accuracy, suggesting minimal overfitting. The test set accuracy of the model stood at a remarkable 97.64%. Examining the confusion matrix, it is evident that the classifier performed exceptionally well across multiple classes, with a high concentration of correct predictions along the diagonal. The lighter shades in the off-diagonal cells point to a few misclassifications, but these are considerably outnumbered by the correct predictions, underscoring the model's strong discriminative ability across the classes. This robust performance attests to the LSTM's capacity to capture sequential patterns in the data, making it a powerful tool for classification tasks in complex datasets.

Model	**Accuracies**
Random Forest Classifier	95.8%
Multinomial Naïve Bayes	92.9%
Support Vector Machine (LinearSVC)	92.89%
LSTM	97.64%

Table 1: Accuracy of Different Models

5 Conclusions:

In this project, we endeavoured to identify the authorship of emails using a blend of stylometric features alongside TF-IDF and Count Vectorization techniques. Our methodology was put to the test using four distinct classifiers: Random Forest, LinearSVC, Multinomial Naive Bayes, and LSTM. The Random Forest Classifier, known for its robustness, showed promising results with high accuracy, indicating its capability to handle the complexity of feature-rich data. LinearSVC, typically favoured for its efficiency with high-dimensional data, performed comparably, highlighting its appropriateness for text classification tasks. The Multinomial Naive Bayes classifier, although slightly less accurate, provided valuable insights given its probabilistic approach and computational efficiency. Lastly, the LSTM model demonstrated the power of neural networks to capture sequential dependencies within text data, achieving a remarkable accuracy that underscores the potential of deep learning in natural language processing tasks.
Limited computational power led us to focus on the top 10 email senders initially. To enhance the model, we need more computational resources for a broader dataset. Leveraging advanced models like transformers in NLP tasks could improve accuracy. Diverse classifiers, fine-tuned through strategic feature selection, can capture writing nuances. Despite constraints, ongoing ML and computational advancements offer opportunities for scaling and refining our approach in authorship identification.

6 Github Link of the project:

https://github.ncsu.edu/kmarrip/engr-ALDA-project-Fall2023-P15

References

[1] ifile: An application of machine learning to e-mail filtering, 2000. URL: https://www.researchgate.net/publication/2636083_ifile_An_Application_of_Machine_Learning_to_E-Mail_Filtering.

[2] William W. Cohen. Learning rules that classify e-mail. 1996. URL: https://api.semanticscholar.org/CorpusID:12621391.

[3] Template-based algorithms for connectionist rule extraction, 1994. URL: https://proceedings.neurips.cc/paper/1994/hash/24896ee4c6526356cc127852413ea3b4-Abstract.html.

Team P-21 Cover Page: Project Contributions

Member 1: Krishna Chaithanya Marripati (kmarrip)

- **Data Cleaning:** Efficiently converted extensive text data from Kaggle into a manageable CSV format.
- **Methods Development:** Innovatively applied Random Forest algorithms, optimizing them for text classification.
- **Exploration:** Investigated key NLP features like word count and average word length, while eliminating stop words to refine data quality.
- **Result Analysis:** Conducted in-depth evaluations of various models, leading to the selection of LSTM based on performance metrics.
- **Conclusion:** Emphasized the critical role of text preprocessing in enhancing model accuracy.
- **Presentation Preparation:** Focused on elucidating data and text preprocessing techniques, alongside exploratory data analysis.
- **Final Report:** Detailed the dataset characteristics and sampling strategies.

Member 2: Mery Harika Gaddam (mgaddam)

- **Data Cleaning:** Transformed the CSV data into a functional pandas DataFrame, resolving encoding issues by standardizing to UTF-8.
- **Methods Development:** Pioneered the use of Linear SVC, tailoring these approaches for our specific dataset.
- **Exploration:** Implemented stemming to distill text data to its linguistic roots.
- **Result Analysis:** Applied multiple cross-validation techniques to compare various machine learning methods.
- **Conclusion:** Highlighted the significant impact of feature engineering and the effectiveness of TF-IDF over simple count vectorization.
- **Presentation Preparation:** Concentrated on the intricacies of feature engineering, TF-IDF, and count vectorization processes.
- **Final Report:** Crafted the introductory and concluding sections, framing the project's scope and summarizing key findings.

Member 3: Sai Vikas Reddy Yeddulamala (syeddul)

- **Data Cleaning:** Converting text data into lowercase, removing punctuation marks, removing stop words.
- **Methods Development:** Focused on deploying Naive Bayes techniques, fine-tuning them for enhanced predictive accuracy.Build an LSTM model for classifiying the sender of the email.
- **Exploration:** Investigated sentiment analysis using polarity scores, employing tools like TextBlob and VADER.
- **Result Analysis:** Drew insightful comparisons between the effectiveness of count vectorizers and TF-IDF in model performance.
- **Conclusion:** Underlined the reason behind by LSTM model has high accuracies over baseline models.
- **Presentation Preparation:** Presented a comprehensive overview of various machine learning techniques including Naive Bayes, SVC, Random Forests, and LSTM.
- **Final Report:** Authored the literature review and detailed the approach and rationale behind our methodological choices.