Email Sender Classification

Group P29
Rohan Vitthal Dhamale, Dinesh Pasupuleti, Srivardhan Vura, Shanmukha Sreenivas Deety
rdhamal, dpasupu, svvura, sdeety

1 Background

1.1 Problem

Machine learning models are being used more and more in a variety of fields, one of which is email communication analysis. Our project's goal is to predict email senders using a variety of machine learning algorithms and thorough pre-processing methods on the Enron email dataset. The Enron email dataset is a large corpus made up of a huge number of emails that were exchanged between Enron Corporation employees. This dataset is a useful resource for analyzing sender identity, language usage, and email communication trends.

The identification of email senders within a large dataset poses a significant challenge, primarily due to the sheer volume of diverse emails exchanged within an organization like Enron. Our objective is to develop a robust predictive model that accurately attributes emails to their respective senders. This entails the utilization of various machine learning algorithms, coupled with meticulous pre-processing steps, to extract meaningful features and patterns from the emails.

The complexity of language, varied writing styles among individuals, potential noise within the dataset, and the absence of labeled data pose specific challenges in accurately predicting the sender of each email. Addressing these challenges requires a systematic approach involving data cleaning, feature engineering, and the selection of appropriate machine learning models.
Our report will delve into the methodology employed, the pre-processing steps undertaken, the machine learning models utilized, and the evaluation of these models' performance in predicting email senders within the Enron dataset. By addressing this challenge, we aim to contribute to the advancement of email analysis techniques, potentially aiding in diverse fields including cybersecurity, forensic investigations, and authorship attribution studies.

1.2 Literature Survey

The paper "The Enron Corpus: A New Dataset for Email Classification Research" talks about the dataset and outlines the classification results using Support Vector Classifiers (SVC) and various representations of email data fields. It discusses the importance of the body of the email and how combining scores from multiple fields can improve classification accuracy. It also highlights various features used in email classification, such as unstructured text (subject and body), categorical text (e.g., "to" and "from"), and numeric data (e.g., message size).

The paper titled "A Comparative Study for Email Classification" discusses the problem of spam emails and presents an experiment in which four different classifiers (Neural Network, SVC, Naïve Bayesian, and J48) were used to classify email data as either spam ('1') or not spam ('0'). The study explores the impact of data size and feature size on the classification results.

The article titled "Sentiment Analysis through Recurrent Neural Network" discusses the significance of sentiment analysis in identifying an individual's writing style. It explores various models such as SVM, Naive Bayes, and LSTM employed for conducting sentiment analysis. This

paper contributes to our comprehension of essential text pre-processing procedures necessary for obtaining accurate sentiment analysis. Additionally, it evaluates the performance of each model employed in the analysis.

2 Methodology

2.1 Approach

In our pursuit of machine learning techniques for authorship and email mining tasks, we selected five algorithms: Support Vector Classifier (SVC), Random Forests, AdaBoost, Naive Bayes, and Long Short-Term Memory (LSTM) Neural Network. These methods were chosen to evaluate their performance using attributes from the Enron Corpus.

The SVC model has remarkable adaptability in text classification, skillfully managing binary and multiclass issues by maximizing margins in high-dimensional spaces after text preprocessing. Naive Bayes relies on feature independence assumptions to perform independently and quickly in text classification problems. AdaBoost uses weighted misclassified cases to iteratively improve classifiers, Random Forests use numerous decision trees to manage high-dimensional data. The sophisticated recurrent neural network known as LSTM is excellent at comprehending long sequences, which is useful for language modeling and sentiment analysis, but it requires careful tweaking.

2.2 Rationale

In our pursuit of performing classification tasks to identify email senders from a list of potential candidates, we outlined the methods experimented with thus far. Our selection of these particular approaches for the classification problem is underpinned by several strategic reasons.

- **Support Vector Classification (SVC):** SVC offers robustness in handling high-dimensional feature spaces and noise, maximizing margins for diverse sender writing styles. Its versatility in both binary and multiclass scenarios, interpretability, and potential for high performance with proper tuning make it a strong choice.
- **Naive Bayes:** Chosen for its simplicity and efficiency in handling text data, Naive Bayes consistently delivers reliable results despite its assumption of feature independence. With suitable feature preprocessing and tuning, it demonstrates commendable performance in sender classification tasks
- **AdaBoost:** AdaBoost's ensemble learning effectively combines weak classifiers into a robust model, improving misclassified instances iteratively. Its adaptability, emphasis on challenging instances, and resilience against overfitting suit both binary and multiclass sender attribution scenarios.
- **Random Forests:** Renowned for handling high-dimensional data and mitigating overfitting, Random Forests combine predictions from multiple trees for dependable results in sender attribution tasks, particularly in handling noisy data while maintaining accuracy.
- **Long Short-Term Memory (LSTM):** Leveraging LSTM's adeptness in handling complex sequential data, it efficiently recognizes diverse sender writing styles across binary and multiclass sender attribution tasks. Its resilience to outliers and interpretability make it promising, requiring meticulous feature engineering and hyperparameter tuning for optimal performance.

3 Plan & Experiment

3.1 Dataset

The Enron dataset, linked to the corporation's bankruptcy in 2001 due to corporate fraud, is widely used by researchers and data scientists for fraud detection, email classification, network analysis, and natural language processing. Comprising over 500,000 emails exchanged between Enron employees from 1999 to 2002, totaling around 1.7 gigabytes, this diverse dataset covers various topics, including business strategies, finances, personal relationships, and more. Its significance lies in its extensive

metadata, encompassing timestamps, sender and recipient addresses, and email folders, which prove valuable for linguistic analysis and sender prediction research

3.1.1 Exploratory Data Analysis

In this phase, the dataset's folder structure was transformed into a CSV format to facilitate efficient data parsing. Specifically, folders such as 'sent,' '_sent_mail,' and 'sent_items' as shown in Figure 1 were selected to streamline the dataset, mitigating redundant and duplicate emails. This conversion process structured the complex folders into a tabular format, incorporating columns for file paths and email content.

Moreover, an initial exploration focused on understanding the distribution of emails across various users or senders within the dataset. This analysis quantified and visualized the email volumes attributed to each user, shedding light on individual email activity patterns. The identification of prolific senders, along with those exhibiting lower email volumes, provided valuable insights into communication dynamics, highlighted significant contributors, and potentially identified anomalies or outliers. Typically, visual representations such as bar charts or histograms, exemplified in Figure 2, were employed to illustrate this sender distribution within the Enron corpus.

3.1.2 Data Pre Processing and Cleansing

The file path served as a label, containing the sender's name. Concurrently, we extracted critical elements such as the email body, subject, and timestamp from the content for detailed analysis.
Regarding preprocessing steps applied to the email content:

- **Removal of Unnecessary Prefixes:** Strings like 'Forwarded by' and 'Original Message' were eliminated from the email content, ensuring the exclusion of irrelevant or redundant information.
- **Filtering of Rows:** Any rows containing 'NA' values or empty strings in the final processed text were removed, refining the dataset and enhancing its integrity.
- **Text Standardization:** To ensure uniformity in the dataset, all text was converted to lowercase, and punctuations were systematically removed, simplifying subsequent analysis and processing.

3.1.3 Data Sampling

Additionally, to ensure a balanced and unbiased dataset for the model's training, users who had sent a minimum of 2000 emails were selectively sampled. This approach aimed to prevent skewed input into the model, ensuring fair representation and robustness in handling diverse email patterns. This meticulous sampling strategy contributes to the model's overall reliability and impartiality in its predictions.

3.1.4 Feature Extraction

In the process of feature extraction from the Enron dataset, we focused on deriving essential insights:.

- **Recipient Information:** Extracted email addresses or names of recipients to analyze communication networks within Enron.
- **Word Counts and Average Word Lengths:** Insights into email lengths and complexity, providing an overview of communication depth.
- **Sentiment Scores:** Calculated polarity scores to gauge emotional tone and sentiment within emails.
- **Parts of Speech (POS) Insights:** Tagged words with grammatical categories to understand language constructs and styles.
- **Greeting and Most Common Words (Body and Subject):** Identification of common greetings and frequently used words, offering insights into prevalent email opening styles and common topics.

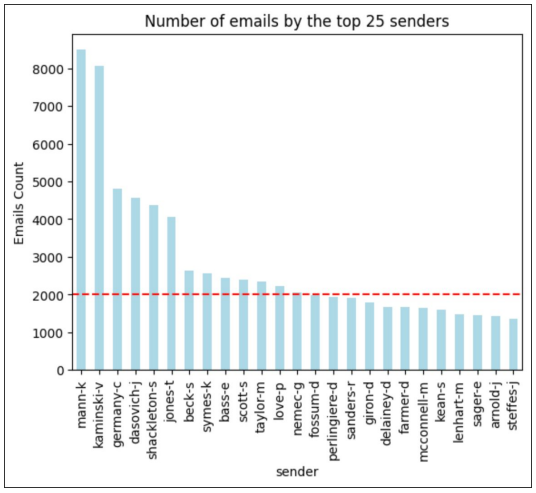


Figure 1: Top Senders

These extracted features were pivotal in creating a robust understanding of the dataset's contents, providing a foundation for subsequent analyses and model development.

3.1.5 Data Transformation

- **Tokenization:** Break the email text into individual words or tokens, making it easier to analyze. For example, "The quick brown fox jumps" becomes ["The", "quick", "brown", "fox", "jumps"].
- **Removing Stop Words:** Eliminate common, less informative words like "the," "is," and "in" from ["quick", "brown", "fox", "jumps"].
- **Lemmatization:** Reduce words to their base form, ensuring consistency. For example, "jumps" is lemmatized to "jump."

These steps were integral in transforming the raw email text into a structured and refined format, laying the groundwork for modeling.

3.2 Hypotheses

- **Linguistic Patterns and Sender Identification:** Investigate whether distinctive linguistic patterns, writing styles, and communication tendencies within emails can predict the probable sender reliably, regardless of the model's performance metrics.
- **Impact of Metadata on Sender Prediction:** Explore how crucial metadata elements such as word count and sender-recipient associations augment the precision of identifying email senders.

3.3 Experimental Setup

The dataset, comprising 10 features and 60,000 rows, underwent a standardized model training process across multiple classifiers, including Support Vector Classifier (SVC), Naive Bayes, AdaBoost, and Random Forest models. The collective steps for model training and evaluation are detailed below:
To train diverse models including Naïve Bayes, SVM, Random Forest, and AdaBoost, we employed various preprocessing techniques tailored to different data types:

- **Numeric Columns Preprocessing:**
  Applied MinMax scaling to ensure consistent scales across numeric columns.
- **Categorical Columns Preprocessing:**
  Utilized one-hot encoding to represent categorical data for effective model training.

- **Email Body Processing:**
  Employed both CountVectorizer and TF-IDF Vectorizer to process and extract features from the email body, allowing different models to interpret textual information effectively.

LSTM Model Training:

- Encoded the email body text into sequences using integer encoding for the LSTM model, enabling the model to understand and analyze the sequential nature of the textual data.
- Applied one-hot encoding to represent sender information, ensuring compatibility and effective training of the LSTM model on this information-rich dataset.

This comprehensive preprocessing setup aimed to optimize the input data for the various models employed, allowing them to interpret and learn from the distinctive features present in the dataset.
In addition to the preprocessing steps outlined earlier, we conducted comprehensive model training by employing both 5-fold and 10-fold cross-validations. This allowed us to analyze the performance across four trained models:

Cross Validation:
Trained models on diverse cross-validation techniques: 5-fold and 10-fold. Utilized four distinct approaches for model training:

- 5-Fold using Count Vectorizer
- 10-Fold using Count Vectorizer
- 5-Fold using TF-IDF Vectorizer
- 10-Fold using TF-IDF Vectorizer

The adoption of varied cross-validation methods and vectorization techniques enabled a meticulous examination of model performance, ensuring a robust assessment of each model's effectiveness in handling the dataset's complexities and nuances.

We've set up a series of experiments utilizing diverse models and preprocessing techniques to address the hypotheses, aiming to overcome challenges associated with feature extraction and model interpretation for robust, accurate sender identification.

4 Results

The model's performance was assessed using various cross-validation techniques and vectorization methods.

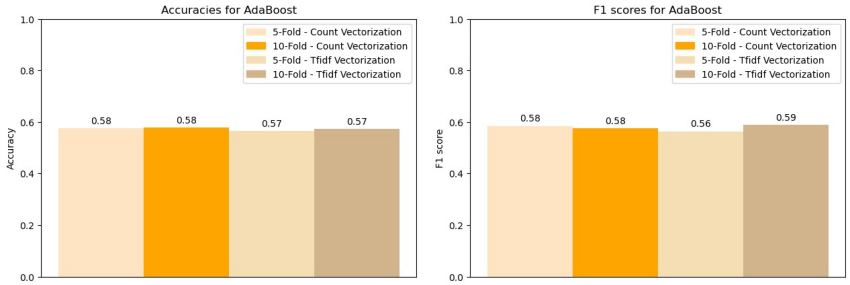
4.1 AdaBoost

The graphs in Figure 2 summarize the accuracies and F1 scores obtained for each configuration. The AdaBoost model's performance across these configurations appears relatively consistent in terms of accuracies and F1 scores. Notably, the 10-fold cross-validation using TfidfVectorizer yielded the highest F1 score of 0.59, indicating slightly better predictive ability compared to other setups. However, the differences in performance metrics among the configurations were marginal.

These results suggest that the AdaBoost model, while stable across various setups, did not exhibit substantial variations in performance concerning the cross-validation techniques and vectorization methods applied.

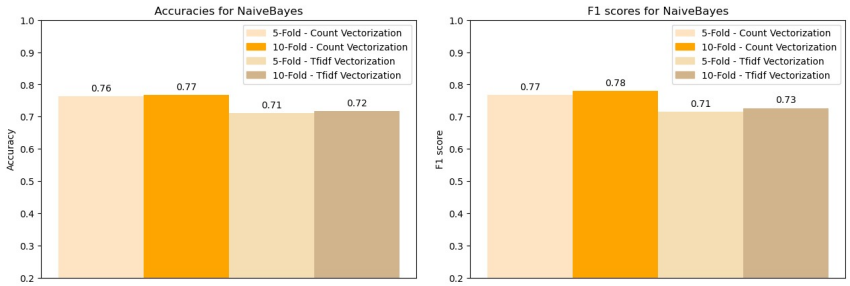
4.2 NaiveBayes

Figure 3 summarizes the accuracies and F1 scores obtained for each configuration. The Naive Bayes model performance differed notably based on the vectorization techniques used for text preprocessing.



(a) AdaBoost Accuracy (b) AdaBoost F1 Score

Figure 2: AdaBoost Results
https://www.overleaf.com/project/656d484265b548e4e25d4b25



(a) Naive Bayes Accuracy (b) Naive Bayes F1 Score

Figure 3: Naive Bayes Results

When considering the CountVectorizer, the 10-fold cross-validation demonstrated slightly higher accuracy (0.77) and F1 score (0.78) compared to the 5-fold cross-validation, indicating improved model robustness. Conversely, with TfidfVectorizer, both accuracy and F1 scores were marginally lower for both 5-fold and 10-fold cross-validation setups.

Considering these outcomes, the Naive Bayes model exhibited better performance with the CountVectorizer, especially evident in the 10-fold cross-validation, suggesting that this configuration led to a more accurate and precise model.

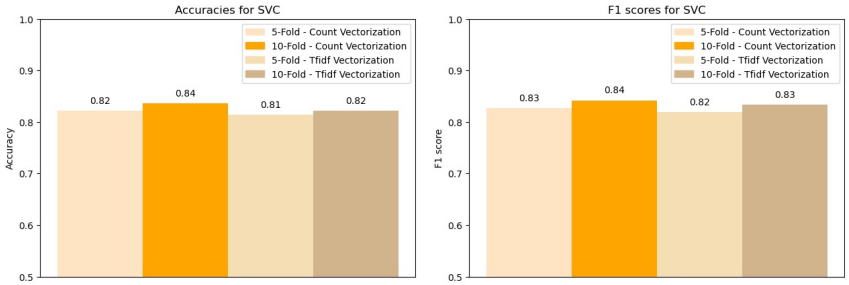
4.3 Support Vector Classifier (SVC)

The graphs in Figure 4 summarize the accuracies and F1 scores obtained for each configuration. The Support Vector Classifier (SVC) model's performance varied significantly based on the vectorization technique applied during text preprocessing. Notably, the model achieved higher accuracy and F1 scores with the CountVectorizer across both 5-fold and 10-fold cross-validations compared to the TfidfVectorizer. The 10-fold cross-validation using CountVectorizer exhibited the best performance, yielding an accuracy of 0.84 and an F1 score of 0.84, indicating robust and precise classification in identifying email senders.

These results emphasize the importance of the chosen vectorization method in optimizing the SVC model's performance for sender identification tasks.

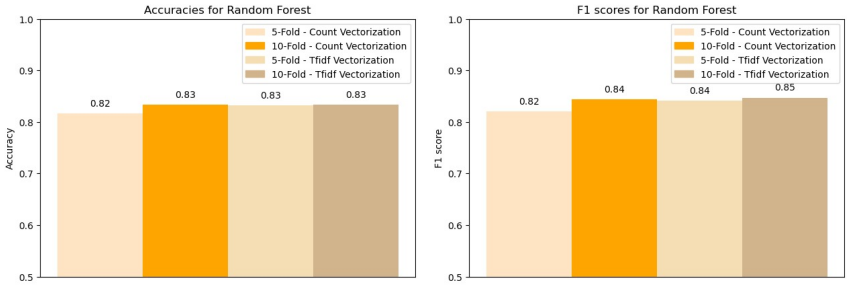
4.4 Random Forest Model

Figure 5 summarizes the accuracies and F1 scores obtained for each configuration. The Random Forest model demonstrated consistent and promising performance across both CountVectorizer and TfidfVectorizer in terms of accuracy and F1 scores. In the case of the 10-fold cross-validation, both



(a) SVC Accuracy (b) SVC F1 Score

Figure 4: SVC Results



(a) Random Forest Accuracy (b) Random Forest F1 Score

Figure 5: Random Forest Results

vectorization techniques yielded similar accuracies of 0.83. However, the TfidfVectorizer showcased a slightly higher F1 score of 0.85 compared to 0.84 with CountVectorizer.

This indicates that the Random Forest model, particularly when using TfidfVectorizer, excels in classifying email senders based on textual features. The marginal improvement in F1 score suggests that the TfidfVectorizer might capture the textual nuances more effectively, contributing to enhanced performance.

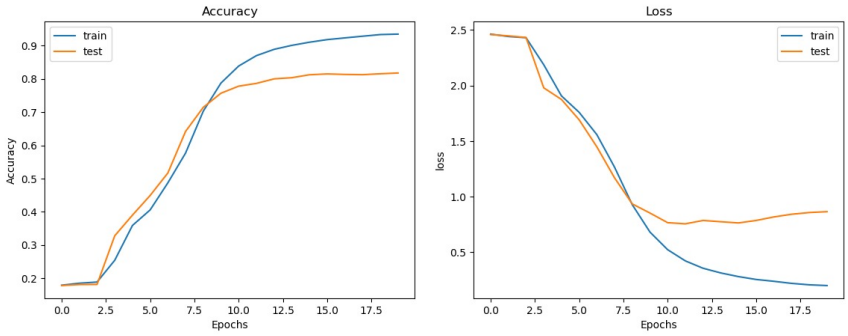
The results affirm the robustness of the Random Forest model in handling the sender classification task, particularly when leveraging the TfidfVectorizer for feature extraction.

4.5 LSTM Model

The LSTM (Long Short-Term Memory) model architecture consisted of an Embedding layer for text vectorization, followed by an LSTM layer for sequential pattern learning, a Flatten layer for reshaping data, and a Dense layer for multi-class classification with 13 output classes.

Loss Function
During the validation phase, the model exhibited a slight increase in the loss function towards the end of the training process. Despite this, the model demonstrated promising accuracy, showcasing its ability to accurately predict sender identities based on the email content and associated features.

The LSTM model achieved an accuracy of 82.30% in classifying email senders, indicating its proficiency in understanding and leveraging sequential patterns within the textual data. While there was a slight increase in the loss function on the validation data towards the end of training, the model still maintained a promising accuracy level, showcasing its robustness in classifying email sender identities.



(a) LSTM Accuracy (b) LSTM Loss

Figure 6: LSTM Results

5 Conclusion

This project provided significant insights into email sender identification using machine learning models with the Enron email dataset. Each model—SVC, Naive Bayes, AdaBoost, Random Forest, and LSTM Neural Network—exhibited distinct strengths and limitations in predicting email senders. Key Findings:

1. **Preprocessing Impact:** Text preprocessing methods heavily influenced model performance, demonstrating varied efficacy in handling textual features.
2. **Model Variability:** Model performance differed based on text vectorization methods, with Naive Bayes excelling with CountVectorizer, while SVC performed better with TfidfVectorizer.
3. **Model Robustness:** Random Forests maintained reliable performance across techniques, displaying resilience in handling textual data. AdaBoost demonstrated stability in predictive ability across various configurations.
4. **LSTM Versatility:** The LSTM model achieved an accuracy of 82.30% in identifying senders, showcasing its aptitude in understanding sequential data patterns.

Overall, this project highlighted the pivotal role of feature extraction, preprocessing, and model selection in accurately attributing email senders. Further exploration into ensemble methods, optimization techniques, and deeper LSTM architectures could enhance future efforts in this domain.

References

[1] A Comparative Study for Email Classification.

[2] Sentiment Analysis by using Recurrent Neural Network.

[3] The Enron Corpus: A New Dataset for Email Classification Research.

[4] Anirudh Harisinghaney, Aman Dixit, Saurabh Gupta, and Anuja Arora. 2014. Text and image-based spam email classification using KNN, Naïve Bayes and Reverse DBSCAN algorithm. In *2014 International Conference on Reliability Optimization and Information Technology (ICROIT)*. 153-155.

Github Repository

CSC522 Final Project Report- Cover Page
Team Member Contributions

Member 1: Rohan Vitthal Dhamale

- **Data Cleaning:** Removal of emails in folders other than sent, sent_items, sent_mail.
- **Methods Development:** Worked on Naive Bayes model.
- **Exploration:** Extracted features like greetings, POS and most common words from the body and subject.
- **Results Analysis:** Performed a comprehensive analysis, calculating essential metrics.
- **Conclusion Drawing:** Concluded that preprocessing significantly affects model performance; Naive Bayes effectively handles extracted features.
- **Presentation Preparation:** Introduction, dataset and data pre-processing
- **Final Report Creation:** Results, Conclusion

Member 2: Dinesh Pasupuleti

- **Data Cleaning:** Text standardization.
- **Methods Development:** Worked on Random Forest and LSTM models.
- **Exploration:** Extracted features like recipient information and sentiment scores.
- **Results Analysis:** Analyze the performance of different cross-validation techniques.
- **Conclusion Drawing:** Concluded on text standardization's importance; identified Random Forest and LSTM robustness through cross-validation.
- **Presentation Preparation:** Models, conclusion.
- **Final Report Creation:** Dataset, Data Sampling

Member 3: Srivardhan Vura

- **Data Cleaning:** Removal of text with unnecessary prefixes.
- **Methods Development:** Worked on SVC model.
- **Exploration:** Extracted features like sentences count, average word count, and removed stop words.
- **Results Analysis:** Compare and evaluate the performance of CountVectorizer and TF-IDF.
- **Conclusion Drawing:** Concluded on the impact of text vectorization methods on SVC model; highlighted specific relevant text features.
- **Presentation Preparation:** Models and data visualisation of results.
- **Final Report Creation:** Problem Statement, Literature Survey

Member 4: Shanmukha Sreenivas Deety

- **Data Cleaning:** Derived the email body and recognized and handled null values.
- **Methods Development:** Worked on AdaBoost model and implemented column transformers.
- **Exploration:** Performed tokenization and lemmatization.
- **Results Analysis:** Analyzed accuracies and F1 scores for all the models.
- **Conclusion Drawing:** Concluded on AdaBoost's efficacy; emphasized data cleaning and tokenization/ lemmatization impact on accuracy.
- **Presentation Preparation:** Data sampling and feature extraction.
- **Final Report Creation:** Approach, Rationale