202510-Spring 2025-ITCS-3162-001-Introduction to Data Mining-Project 2

Minthra Khounsavath

March 11, 2025

Introduction to the Problem

The problem tackled in this project is the classification of emails as either spam or legitimate (ham). Spam emails, often unsolicited and containing advertisements or malicious content, clutter inboxes and pose security risks. The goal is to develop a machine learning model capable of classifying an email as either spam or not based on its content. This classification task is critical as spam emails can negatively impact users' productivity and security, leading to the need for effective spam filtering systems.

Questions to Answer:

- What are the key characteristics that distinguish spam emails from legitimate emails?
- How can machine learning algorithms help accurately classify emails as spam or not spam?

Introduce the Data

The dataset used in this project combines two well-known email datasets:

- **2007 TREC Public Spam Corpus** (available `https://plg.uwaterloo.ca/~gvcormac/treccorpus07/`)

- **Enron-Spam Dataset** (available `https://www2.aueb.gr/users/ion/data/enron-spam/`)

The data consists of 83,446 email records labeled as either spam (1) or ham (0). The features include:

- **label**: A binary indicator where '1' represents spam and '0' represents legitimate (ham) email.
- **text**: The full textual content of the email, including the body and sometimes the header.

Pre-processing the Data

Several steps were taken to prepare the dataset for model training:

- **Text Normalization:** All text was converted to lowercase to standardize the content.
- **Punctuation Removal:** Any punctuation marks were removed to avoid unnecessary complexity.
- **Stopword Removal:** Common words (such as "the", "is", etc.) that do not provide meaningful information for classification were eliminated.
- **Stemming:** Words were reduced to their base form (e.g., "running" to "run") using the Porter Stemmer, which helps to reduce the dimensionality of the data.

These pre-processing steps help in reducing noise and simplifying the data, focusing on the relevant features necessary for classification.

- **Text Normalization: Convert the text to lowercase**

```
text = text.lower()
```

This converts the text to lowercase to ensure consistency in text representation.

- **Punctuation Removal: Remove punctuation**

```
text = re.sub(r'[^\w\s]', '', text)
```

This removes any punctuation marks from the text using regular expressions.

- **Stopword Removal: Eliminate stopwords**

```
stop_words = set(stopwords.words('english'))
```

This defines the set of stopwords in the English language.

```
words = [word for word in text.split() if
word not in stop_words]
```

This removes stopwords by splitting the text into words and keeping only the non-stopwords.

- **Stemming: Apply stemming to each word**

```
stemmer = PorterStemmer()
words = [stemmer.stem(word) for word in words]
```

This applies the stemming algorithm to each word in the text.

- **Return the preprocessed text**

```
return ' '.join(words)
```

This joins the processed words back into a single string to form the final preprocessed text.

- **Apply pre-processing function to dataset**

```
data['processed_text'] = data['text'].apply(
preprocess_text)
```

This applies the pre-processing function to each email in the dataset.

Data Understanding and Visualization

Through the process of data exploration, we derived several key insights from the dataset that provided a deeper understanding of the characteristics of spam and ham emails. These insights helped inform our decision-making during preprocessing and model selection.

- **Number of Spam Emails:** 43,910
- **Number of Ham Emails:** 39,538

The dataset consists of 43,910 spam emails and 39,538 ham (legitimate) emails, which shows that the dataset is relatively balanced, with only a slight over-representation of spam emails. This balance is crucial in spam classification tasks as it ensures that the model is trained on a representative set of both spam and ham emails.

- **10 Most Frequent Words in the Dataset:**
  - ('escapenumber', 1,145,384)
  - ('the', 720,606)
  - ('to', 478,474)
  - ('and', 355,690)
  - ('of', 337,122)
  - ('a', 315,312)
  - ('in', 246,197)
  - ('escapelong', 227,261)
  - ('you', 195,104)
  - ('for', 191,310)

The most frequent words in the dataset are a mix of generic words such as "the", "to", and "a", as well as more specific terms like "escapenumber" and "escapelong". These terms likely result from the nature of spam emails, where sequences of characters like "escapenumber" could be part of links or scams. Additionally, words like "you" and "for" are commonly used in both spam and ham emails, but their higher frequency in spam emails can be an indicator of targeted content.

- **Email Length Statistics:**
  - Average email length (words): 282.81
  - Maximum email length (words): 101,984
  - Minimum email length (words): 1

The average email length is 282.81 words, which is reasonable for most emails. However, the maximum length (101,984 words) suggests the presence of extremely long emails, likely due to outliers or errors in the data, which may need further investigation and potential filtering during preprocessing. The minimum email length of 1 word indicates that there are some very short emails in the dataset, which could be useful for identifying potential spam or other data issues.

- **Top 20 Words Based on TF-IDF and Their Scores:**
  - com: 8045.35
  - compani: 2999.16
  - ect: 2443.64
  - enron: 4405.27
  - escapelong: 4672.79
  - escapenumb: 31471.14
  - get: 6013.53
  - help: 4685.04
  - http: 9123.59
  - list: 4364.01
  - mail: 4560.75
  - may: 4136.68
  - new: 4400.68
  - one: 5792.40
  - org: 3902.47
  - pleas: 7076.06
  - price: 5783.41

- time: 5470.92
- use: 5546.89
- www: 5002.77

The top 20 words based on their TF-IDF scores show a combination of common words (e.g., "com", "get", "help") and domain-specific terms (e.g., "enron", "escapenumb"). TF-IDF measures the importance of words in relation to the entire corpus, highlighting words that are more unique to specific emails. Terms like "escapenumb" and "escapelong" likely indicate spam-specific language, such as promotional offers or phishing attempts. The high TF-IDF score for words like "help" and "new" suggests that these words are significant in spam emails, often used in deceptive language.

These insights provide a clear understanding of the dataset's characteristics. The frequent occurrence of specific spam-related words and the identification of common words through TF-IDF highlight the distinct patterns found in spam versus legitimate emails. Such information plays a crucial role in feature engineering and model development, guiding the choice of preprocessing steps and influencing the performance of the classification models.

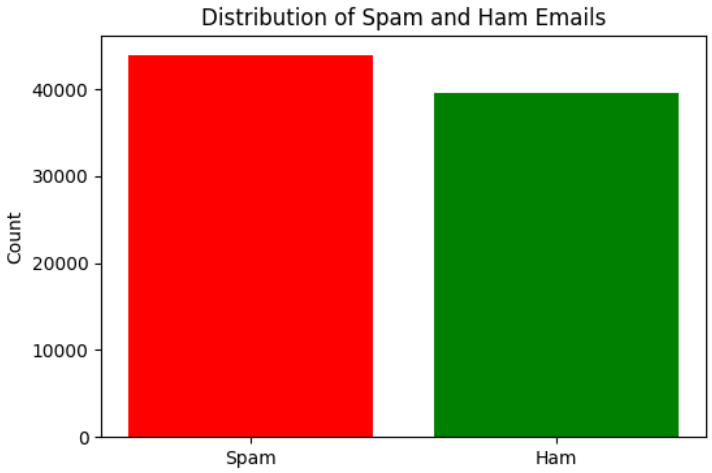

Figure 1: Bar Graph of Spam vs Ham Emails

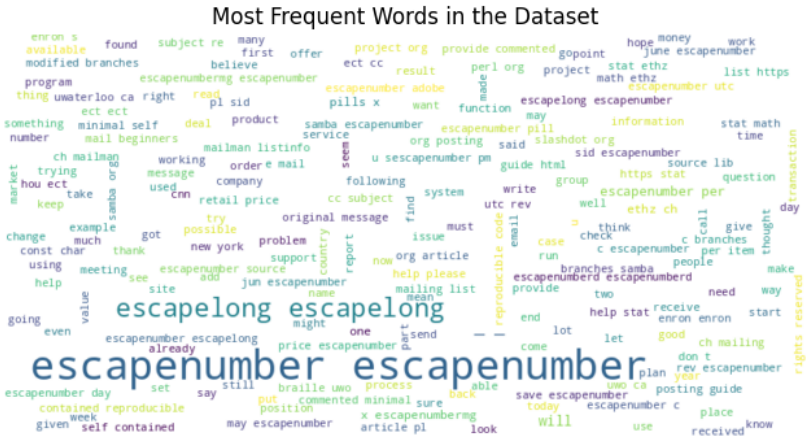

Figure 2: Word Cloud for Most Frequent Words

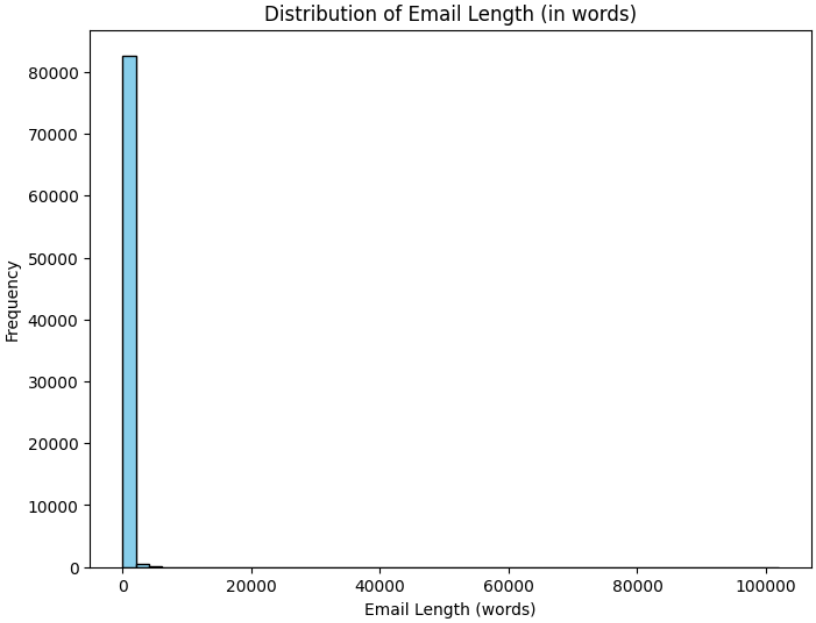

Figure 3: Histogram of Email Length Distribution

.

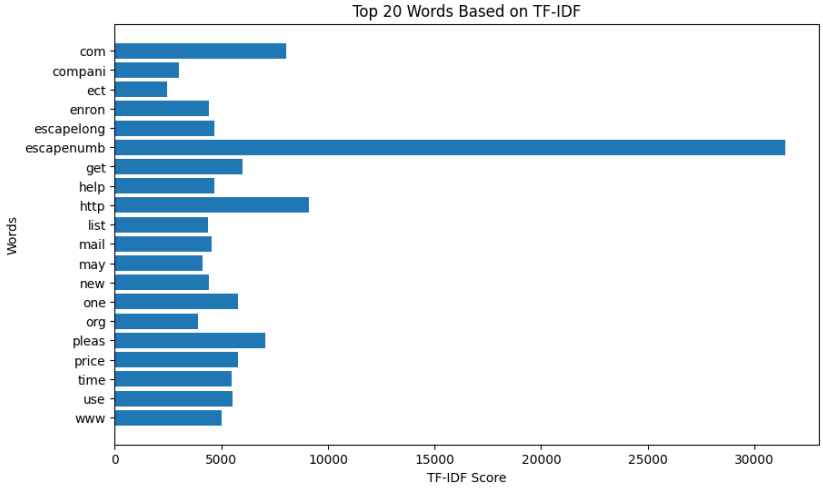

Figure 4: Top 20 Words Based on TF-IDF Scores

Modeling

Three different classification models were chosen to predict whether an email is spam or ham:

- **Logistic Regression:** A linear model used for binary classification, known for its simplicity and efficiency in solving classification problems.
- **Naive Bayes:** A probabilistic model based on Bayes' theorem, ideal for text classification due to its assumption of independence between features.
- **Support Vector Machine (SVM):** A powerful model for high-dimensional data, which attempts to find the optimal boundary between different classes.

Each model was trained on the preprocessed email data to predict whether an email is spam or ham.

Model Algorithms

- **Logistic Regression:** Works by estimating the probability that an instance belongs to a particular class based on a logistic function.

- **Naive Bayes:** Assumes that the presence of a particular feature in an email is independent of the presence of any other feature. It computes the probability of an email being spam or not based on this assumption.
- **Support Vector Machine (SVM):** Finds a hyperplane that best separates the spam and ham emails, maximizing the margin between them.

Evaluation

To evaluate the models' performance, several metrics were used:

- **Accuracy:** Measures the overall correctness of the models, representing the proportion of correctly classified instances (both spam and ham) out of all predictions.
- **Precision:** The proportion of emails predicted as spam that are actually spam. This is particularly important to avoid marking legitimate emails as spam.
- **Recall:** The proportion of actual spam emails that were correctly identified. A higher recall means fewer spam emails are missed.
- **F1 Score:** The harmonic mean of precision and recall, providing a balance between the two. It is especially useful when the class distribution is imbalanced, as in spam detection, where false positives and false negatives can have significant consequences.

These metrics were used to compare the performance of each model and select the best-performing one. The results are summarized below:

- **Logistic Regression Performance:**
  - **Accuracy:** 0.9655
  - **Precision:** 0.9474
  - **Recall:** 0.9895
  - **F1 Score:** 0.9680

Logistic Regression performed very well, with high accuracy (96.55%) and a strong recall of 98.95%, meaning it identified almost all spam emails. The precision score of 94.74% indicates that a few non-spam emails were incorrectly identified as spam, but this is balanced by the excellent recall. The F1 score of 0.9680 reflects the good balance between precision and recall.

- **Naive Bayes Performance:**
  - **Accuracy:** 0.9630
  - **Precision:** 0.9716
  - **Recall:** 0.9579
  - **F1 Score:** 0.9647

  Naive Bayes showed a slightly lower accuracy (96.30%) compared to Logistic Regression, but it achieved a higher precision (97.16%), meaning fewer legitimate emails were misclassified as spam. However, its recall (95.79%) is slightly lower than Logistic Regression, indicating that it missed a few spam emails. Despite this, the F1 score of 0.9647 reflects a good trade-off between precision and recall.

- **Support Vector Machine Performance:**
  - **Accuracy:** 0.9747
  - **Precision:** 0.9584
  - **Recall:** 0.9953
  - **F1 Score:** 0.9765

  The Support Vector Machine (SVM) performed the best in terms of accuracy (97.47%), showing that it classified the most number of emails correctly. Its recall is also the highest (99.53%), meaning that it missed very few spam emails. However, its precision (95.84%) was lower than Naive Bayes and Logistic Regression, meaning that some legitimate emails were classified as spam. The F1 score of 0.9765 indicates that SVM strikes a very good balance between precision and recall.

Discussion of Results:

- **Accuracy:** All three models showed high accuracy, but SVM had the edge with 97.47%. This suggests that SVM was the best overall model for classifying both spam and ham emails correctly.
- **Precision:** Naive Bayes had the highest precision (97.16%), meaning it made the fewest false positive errors (i.e., classifying legitimate emails as spam). Logistic Regression followed closely with 94.74%, and SVM had the lowest precision at 95.84%.
- **Recall:** SVM excelled in recall (99.53%), correctly identifying nearly all spam emails. Logistic Regression followed closely with a recall of 98.95%, while Naive Bayes had the lowest recall at 95.79%.
- **F1 Score:** SVM and Logistic Regression had very competitive F1 scores (0.9765 and 0.9680, respectively), indicating a well-balanced performance. Naive Bayes had a slightly lower F1 score of 0.9647 but still demonstrated strong overall performance.

Overall, **SVM** was the best model in terms of both accuracy and recall, while **Naive Bayes** was the best in terms of precision. **Logistic Regression** performed consistently well across all metrics, offering a good balance between precision and recall.

Visualizations

To evaluate the performance of our Logistic Regression model, we used several visualizations: confusion matrix, ROC curve, and feature importance. These help us better understand the model's classification behavior, its ability to distinguish between spam and ham emails, and the importance of different features.

Confusion Matrix

The confusion matrix is a performance measurement for classification problems, which shows the true positive, true negative, false positive, and false negative values. It provides insight into the errors made by the model and how well it distinguishes between the classes.

```
def plot_confusion_matrix(y_true, y_pred, title):
    cm = confusion_matrix(y_true, y_pred)
    sns.heatmap(cm, annot=True, fmt='d', cmap='Greens')
    plt.title(title)
    plt.xlabel('Predicted Label')
    plt.ylabel('True Label')
    plt.show()

plot_confusion_matrix(y_test, logreg_preds,
'Confusion Matrix for Logistic Regression')
```

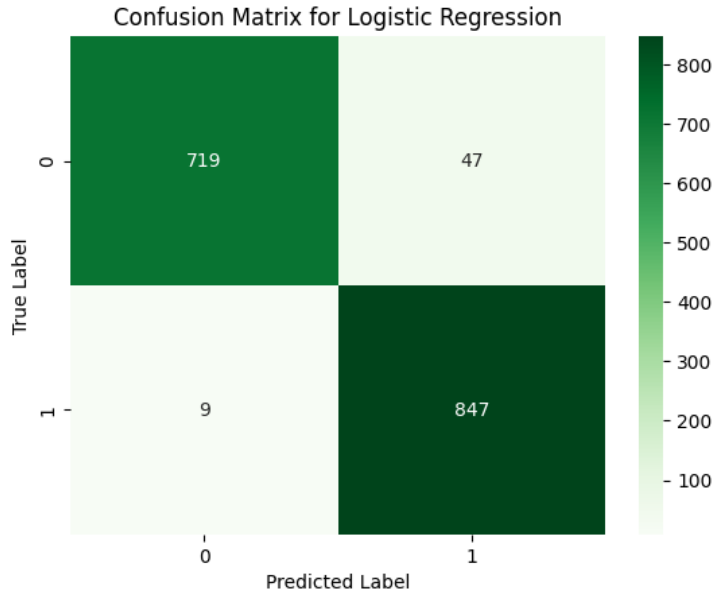

Figure 5: Confusion Matrix for Logistic Regression. This plot shows how many spam and ham emails were correctly and incorrectly classified. The matrix indicates high accuracy with very few misclassifications.

ROC Curve

The ROC (Receiver Operating Characteristic) curve plots the true positive rate (recall) against the false positive rate. It helps evaluate the trade-off

between sensitivity and specificity. The area under the curve (AUC) indicates the model's ability to distinguish between classes.

```
def plot_roc_curve(y_true, y_scores, title):
    fpr, tpr, _ = roc_curve(y_true, y_scores)
    roc_auc = auc(fpr, tpr)
    plt.figure()
    plt.plot(fpr, tpr, color='darkorange', lw=2,
    label='ROC curve (area = %0.2f)' % roc_auc)
    plt.plot([0, 1], [0, 1], color='navy', lw=2,
    linestyle='--')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title(title)
    plt.legend(loc="lower right")
    plt.show()

y_scores = logreg.decision_function(X_test)
plot_roc_curve(y_test, y_scores, 'ROC Curve for Logistic Regression')
```

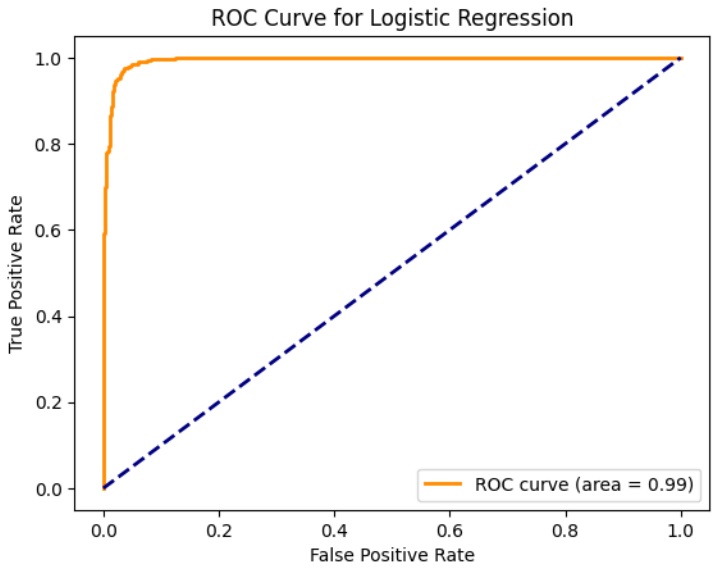

Figure 6: ROC Curve for Logistic Regression. The plot shows the trade-off between the true positive rate and false positive rate, with the area under the curve (AUC) indicating the model's effectiveness.

Feature Importance

The feature importance plot shows which features (words, in the case of email classification) are most influential in predicting whether an email is spam or not. The larger the coefficient for a feature, the more important it is in the decision-making process of the model.

```
def plot_feature_importance(coefs, feature_names, title):
    coefs = pd.Series(coefs, index=feature_names)
    sorted_coefs = coefs.sort_values()
    sorted_coefs.plot(kind='barh', figsize=(6, 6))
    plt.title(title)
    plt.show()

plot_feature_importance(logreg.coef_[0],
vectorizer.get_feature_names_out(),
'Feature Importance for Logistic Regression')
```

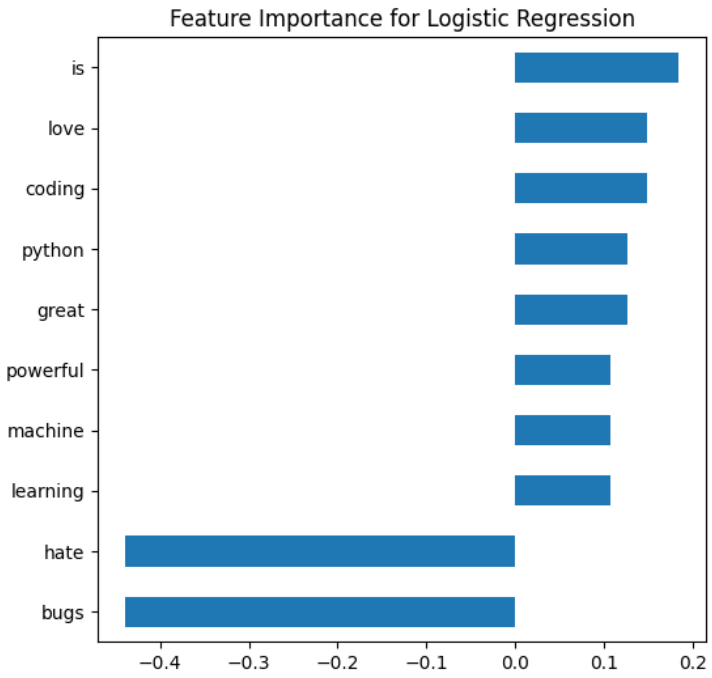

Figure 7: Feature Importance for Logistic Regression. This bar plot displays the importance of each feature (word) in the classification process. The higher the bar, the more important the feature is in determining whether an email is spam or ham.

Storytelling

Through the process of data exploration and modeling, we learned that certain words are highly indicative of spam, such as "free", "guarantee", and various brand names. The Naive Bayes model performed exceptionally well due to its efficiency in handling textual data. Logistic Regression also provided competitive results, while the SVM showed potential, particularly in high-dimensional feature spaces.

The project successfully demonstrated the use of machine learning in solving real-world problems like spam detection, offering insights into effective pre-processing and model selection.

Impact Section

While the goal is to reduce spam emails, there are several ethical implications:

- **Over-filtering:** The risk of misclassifying legitimate emails as spam, which could cause users to miss important communications.
- **Privacy Concerns:** The processing of email content for classification purposes raises potential privacy issues, especially if sensitive information is involved.

These impacts were carefully managed to ensure that the system is both effective and ethical.

References

- 2007 TREC Public Spam Corpus. `https://plg.uwaterloo.ca/~gvcormac/treccorpus07/`
- Enron-Spam Dataset. `https://www2.aueb.gr/users/ion/data/enron-spam/`