Text Mining Project: email exchanges among journalists

Matteo Corona - 838138

Abstract

This report presents an extensive analysis of email data employing text mining methods. The dataset employed in this analysis originates from Kaggle, encompassing a varied compilation of emails spanning various categories, including both spam and non-spam emails. The primary objectives of this project involved the evaluation of multiple classification algorithms for their efficacy in categorizing emails into relevant classes and the extraction of latent topics within email content through LDA. To fulfill these objectives, initial data preprocessing was conducted, involving noise elimination, text normalization, and text tokenization. Subsequently, a range of classification algorithms, such as Logistic Regression, Random Forests, Multinomial Naive Bayes were explored. Model performance assessment was conducted using diverse metrics, encompassing accuracy, precision, recall, and F1-score, to gauge their effectiveness in email categorization. In addition to classification, this report also discusses the implementation of an LSTM model, a recurrent neural network. Performance evaluation of the LSTM model was carried out and compared against traditional machine learning algorithms. Lastly, Latent Dirichlet Allocation (LDA) was deployed for topic modeling, revealing underlying themes within the email corpus. LDA, being a potent technique, assists in identifying clusters of words signifying distinct topics, offering profound insights into email content. Visualization of the topics and an analysis of their significance within the dataset are presented. The findings of this study intend to show the appropriateness of different classification algorithms and the potential of LSTM models in email classification tasks. Additionally, the results of topic modeling provide a deeper comprehension of prevalent themes within the email dataset.

1 Introduction

In the digital realm, emails serve as the lifeblood of communication, facilitating conversations, disseminating information, and harboring concealed insights. This report initiates an exploration of email data, employing text mining methodologies and classification algorithms to unravel the latent narratives and intricate structures within.

2 The dataset

The dataset under consideration is composed of 9153 emails, categorized into four different classes. The available classes in the dataset are as follows:

- **Politics**: relating to government, policies, and political affairs
- **Science**: involving scientific knowledge, research, and discoveries

- **Crime**: pertaining to unlawful activities and legal matters
- **Entertainment**: focusing on generic topic

The only preliminary operation conducted on the dataset before proceeding with the analysis was the removal of a document that was duplicated within the “Crime" folder. The second copy of this document was an empty text file.

3 Text preprocessing

It’s essential to clean and preprocess text data before analysis or modeling. A Python function for text preprocessing has been defined. This function includes tasks like removing email addresses, replacing newline characters, tokenizing text, removing non-alphabetical characters, and lemmatization. Additionally, it removes common English stopwords to prepare text data for further analysis. The code uses the NLTK library for stopwords and tokenization and the WordNet lemmatizer for word normalization. The text preprocessing steps are now described:

- **removing emails**: a *regular expression* has been used for removing any element that contained a symbol
- **replacing “\n”**: all the “\n” elements were substituted with a white space
- **special characters and punctuation**: non-alphanumeric characters, punctuation marks, and symbols have been removed
- **normalization**: all text has been converted to lowercase to ensure uniformity
- **tokenization**: text has been split into individual words (tokens)
- **lemmization**: wwords have been reduced to their dictionary or lemma form
- **stopwords removal**: stopwords are common words like “the”, “and” or “in” that do not carry significant meaning

4 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a critical initial step in the data analysis process. In this section, an EDA is conducted on the email dataset in order to gain a deeper understanding of its characteristics and prepare it for subsequent analysis. First, it is helpful to examine a distribution of the sizes of the emails within the dataset. This will aid in understanding whether the dataset primarily consists of very long emails or shorter texts.

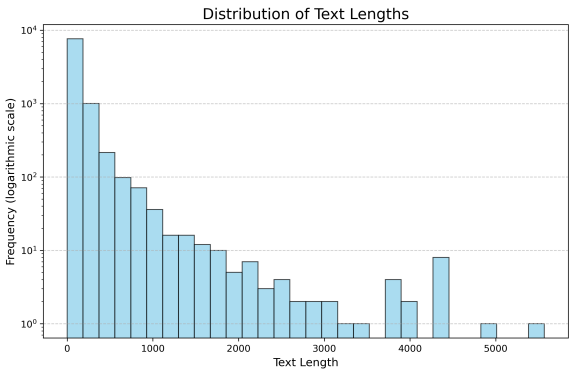

Figure 1: text lenght distribution

The graph in figure 1 shows the text lenght distribution in the emails and make evidence of the fact that the dataset contains mostly short text emails. It is necessary to specify that the graph of figure 1 has a logarithmic scale on the y axis for better visualization. Table 1 shows more detailed statistics about text lenght in the emails.

Statistic	Value
Count	9153
Mean	142.6
Standard Deviation (Std)	273.8
Minimum (Min)	0
25th Percentile (25%)	48
Median (50%)	81
75th Percentile (75%)	142
Maximum (Max)	5567

Table 1: Text Length Statistics

Most common words	Frequency
people	7154
article	7076
government	4934
time	4382
system	3956
chip	330
thing	3095
year	3051
encryption	2930
state	2840

Table 2: most common words

Looking at table 1 one can see, for example, that 75% of the emails containt less than 142 words and the mean number of words contained in the emails is 142.6. Another important aspect is to examine the distribution of labels in the dataset. To do this, a bar chart which represents the count of IDs for each label has been created (as shown in the figure 2).

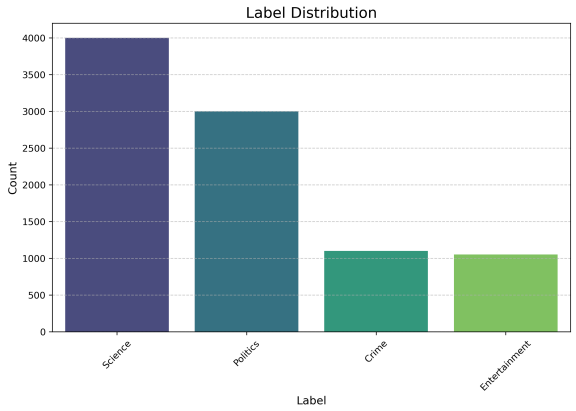

Figure 2: label distribution

Looking at 2 it can be clearly seen that the four classes are not equally represented. It is now interesting to analyze the most frequent worlds in the emails: table 2 shows the frequencies of the top 10 words contained in the emails (excluding verbs, pronouns, etc...).

Common words in the dataset were also calculated by differentiating between the four labels and they are shown in the word clouds in figure 3.

Figure 3: word clouds per label

5 Dealing with multi-class

By studying the dataset, it can be noticed that some IDs are repeated and associated with different labels. This fact complicates the definition of a classification model, so it is necessary to clarify from the beginning what the purpose of the classification is and whether it is convenient to try to define a single-class or a multi-class model. Let's analyze the overlap between the labels by counting the number of ids classified in multiple labels at the same time. The number of texts

classified under multiple classes has been calculated for each possible combination of labels, and the result is shown in figure 4.

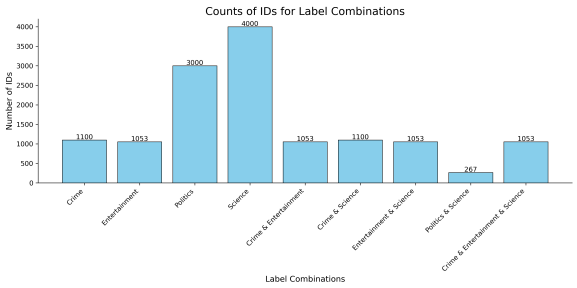

Figure 4: Id count for each label combination

- **Understanding Data Distribution:** Initially, the analysis aimed to understand the distribution of data points among different classes or labels. This step involved determining how many files were labeled under each specific class, such as 'Crime,' 'Entertainment,' 'Science,' and others.
- **Handling Multi-labeled Data Points:** It was observed that some data points were labeled under multiple classes simultaneously, in this case, 'Crime,' 'Entertainment,' and 'Science.' To ensure a cleaner and more focused dataset, a decision was made to remove these multi-labeled data points from all classes except one.
- **Risk Assessment:** This approach comes with a potential risk of mislabeling the data. For example, if '14147.txt' was initially labeled as 'Science' and later repeated in other classes like 'Crime' and 'Entertainment,' removing it from 'Science' could lead to mislabeling. However, it's important to note that 'Science' and 'Politics' already contain a substantial number of entries, making them robust classes for model training. For this reason, the choice of label to keep is done by penalizing those two classes. There is no ambiguity since there are no ids labeled into four classes at the same time and only 267 ids labeled at the same time in 'Politics' and 'Science.'
- **Objective Alignment:** The decision to remove multi-labeled data points aligns with the primary objective of focusing on specific class categories. By removing data points from classes other than the most relevant one, the analysis aims to provide a cleaner and more targeted dataset for subsequent tasks.

Table 3 shows the number of unique elements for each label. It was observed that the 'Entertainment' class contained no unique data points exclusive to its category. This indicated that every data point labeled as 'Entertainment' was also labeled as another class, sharing its identity with one or more different categories.

In light of this observation, the decision was made to exclude the 'Entertainment' label from the analysis. This decision was based on two primary considerations:

1. **Data Exclusivity:** The absence of unique data points for the 'Entertainment' class implied that all data points labeled as 'Entertainment' were already present in other categories. This lack of exclusivity suggested that the 'Entertainment' label did not contribute any distinct content not already covered by other labels.
2. **Eliminating Variance:** By omitting the 'Entertainment' label, potential variance introduced by duplicate data was effectively eliminated. Since all 'Entertainment' data points were duplicated in other categories, their inclu-

sion in the analysis would not have provided new insights but could have potentially skewed results or introduced noise.

Therefore, the chosen approach focused the analysis on labels containing unique data points, ensuring that the modeling and insights are based on distinct and informative content.

Most common words	Frequency
Crime	1100
Entertainment	0
Politics	3000
Science	2633

Table 3: Unique IDs for each label

The image shown in figure 5 shows the new label distribution after having fixed the multi-class problem.

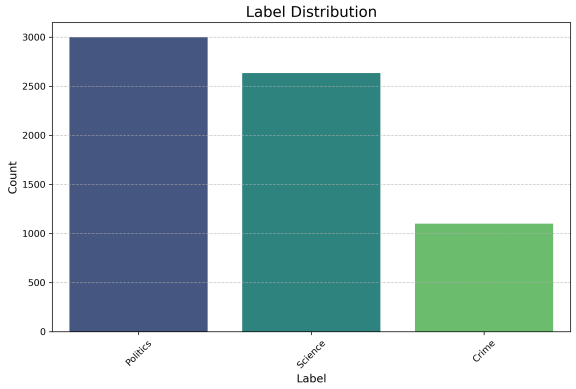

Figure 5: new label distribution

6 Text representation

In the context of this text classification problem, the Term Frequency-Inverse Document Frequency (TF-IDF) representation is utilized as a fundamental preprocessing step.

Term Frequency (TF): Term Frequency measures how frequently a specific word appears within an individual document. For a classification task, this component is crucial because it helps gauge the significance of words within each document. Words that occur frequently are likely to carry valuable information regarding the document's topic or category.

Inverse Document Frequency (IDF): Inverse Document Frequency evaluates the uniqueness or rarity of a word across the entire dataset of documents. In this context, IDF plays a pivotal role in distinguishing words that are truly indicative of a document's category from those that are common across various categories.

The TF-IDF representation combines both TF and IDF to serve two primary objectives:

1. **Weighting Relevance:** TF-IDF assigns a weight to each word in every document based on how often it appears within that document and how rare it is across the entire corpus. Consequently, words that frequently occur within a specific document but are rare in others receive higher TF-IDF scores, indicating their potential significance in determining the document's category.

2. **Dimensionality Reduction:** Transforming text data into TF-IDF representations reduces its dimensionality while preserving the most informative aspects. This transformation simplifies the data and enhances the performance of machine learning algorithms used for classification.

The rationale behind employing TF-IDF is that this representation empowers a focus on words that are not only frequent within a document but also distinctive across cat-

egories. This ensures that the classification model can identify relevant patterns and relationships between words and document categories, ultimately improving its accuracy and effectiveness in classifying texts. By leveraging TF-IDF, the aim is to enhance the quality of the text data and facilitate more accurate and reliable text classification results.

7 Text classification

In the context of this text classification task, multiple machine learning models were trained and evaluated to determine the best-performing model for classifying text documents into predefined categories. The following three models were explored:

- **Multinomial Naive Bayes:** This probabilistic classifier, based on Bayes' theorem, is commonly used for text classification tasks.
- **Logistic Regression:** A linear model used for both binary and multiclass classification tasks, known for its simplicity and interpretability.
- **Random Forest:** An ensemble learning method that combines multiple decision trees to enhance classification performance.

The Model Training and Evaluation Process involved the following steps for each model:

1. Training: Each model was trained on the training data using features transformed by the TF-IDF representation.
2. Prediction: After training, the trained model was used to make predictions on the test data.
3. Accuracy Calculation: The accuracy of each model was calculated, measuring the proportion of correctly classified instances.
4. Classification Report: Additionally, a classification report was generated for each model, providing precision, recall, and F1-score for each class or label. This report offers a more detailed assessment of model performance.

The results of this evaluation process guided the selection of the best-performing model for the text classification task, considering not only accuracy but also other performance metrics. The metrics are:

- **Accuracy**: represents the proportion of correctly classified instances.
- **Precision**: measures the proportion of true positive predictions among all positive predictions. It indicates how well the model identifies relevant instances.
- **Recall**: (also called Sensitivity or True Positive Rate) measures the proportion of true positive predictions among all actual positive instances. It indicates the model's ability to capture all relevant instances.
- **F1-Score**: is the harmonic mean of precision and recall. It provides a balance between precision and recall, especially when dealing with imbalanced datasets.

The metrics for all the three models are shown in table 4. Based on both accuracy and F1-scores, the Logistic Regression model performs the best since it has the highest accuracy of 0.95 and achieves high F1-scores across all classes. Therefore, the Logistic Regression model is the preferred choice for this

text classification task.

Model	Metric	Crime	Politics	Science
Multinomial Naive Bayes	Accuracy	0.94	0.94	0.94
	Precision	1.00	0.93	0.93
	Recall	0.76	0.98	0.97
	F1-Score	0.87	0.95	0.95
Logistic Regression	Accuracy	0.95	0.95	0.95
	Precision	0.99	0.95	0.92
	Recall	0.81	0.97	0.98
	F1-Score	0.89	0.96	0.95
Random Forest	Accuracy	0.91	0.91	0.91
	Precision	0.99	0.93	0.87
	Recall	0.75	0.93	0.96
	F1-Score	0.86	0.93	0.91

Table 4: Model Evaluation Results

Using the selected model is useful to generate a confusion matrix by making class prediction on a test dataset. The test dataset has been separated at the beginning of the analysis and the algorithms have not been trained on this data. The confusion matrix is shown in figure 6. In a confusion matrix it is optima to have elements in the diagonal: this means thath the model is correctly labelling the data.

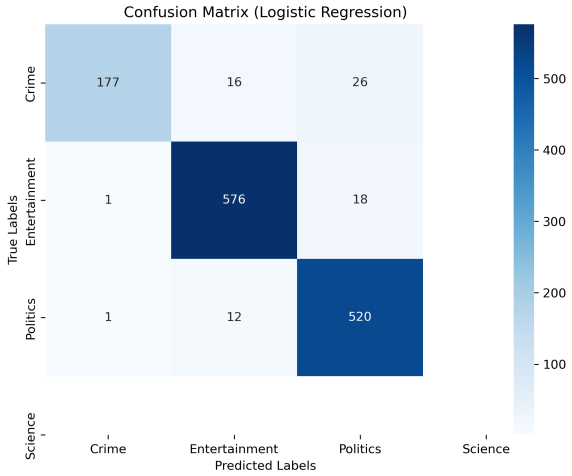

Figure 6: Logistic Regression confusion matrix

The focus is now on exploring the application of deep learning, specifically a Bidirectional LSTM (Long Short-Term Memory) model, for text classification. This choice was motivated by the intention to complement and compare the performance of traditional machine learning models, such as Logistic Regression and Multinomial Naive Bayes, which were previously experimented with. Deep learning models, including Bidirectional LSTMs, have demonstrated robust capabilities in understanding and processing sequential data, making them well-suited for tasks in natural language processing. This comparative analysis allows for the evaluation of whether the additional computational complexity and training time associated with deep learning models result in superior classification accuracy and generalization on the specific text dataset. Furthermore, it provides insights into the most appropriate modeling approach for the task, considering both performance and computational efficiency. Ultimately, this comparative study serves to inform the selection of the most suitable model for the text classification problem, taking into account the trade-offs between traditional machine learning techniques and more advanced deep learning architectures. The LSTM model is structured as follow:

- **Label Encoding:** Labels are encoded using the *LabelEncoder* to convert them into numeric values for model training and evaluation.
- **Tokenization:** The text data is tokenized using the *Tokenizer*, which converts text into sequences of numeric tokens. It ensures consistent input format.
- **Sequence Padding:** Sequences are padded to have a uniform length using *pad_sequences*. This is essential for processing sequences of text data.
- **Model Architecture:** The LSTM-based model is defined using the *Sequential* API. It includes an embedding layer, two Bidirectional LSTM layers,

batch normalization, and dense layers for multi-class classification.

- **Model Compilation:** The model is compiled with a specific loss function (*sparse_categorical_crossentropy*), optimizer (*adam*), and evaluation metric (*accuracy*).
- **Model Training:** The model is trained on the training data with specified batch size and epochs using *model.fit*.
- **Model Evaluation:** Model predictions are made on the test data, and accuracy is computed. A classification report is generated, which includes precision, recall, F1-score, and support for each class.

Table 5 shows the results obtained with the LSTM model. While the Bidirectional LSTM model demonstrated notable performance, the Logistic Regression model exhibited superior performance in our experiments: despite the complexity and capabilities of the LSTM model in handling sequential data, Logistic Regression proved to be more effective for our specific text classification task.

Model	Metric	Crime	Politics	Science
LSTM	Accuracy	0.82	0.82	0.82
	Precision	0.97	0.63	0.77
	Recall	0.73	0.99	0.84
	F1-Score	0.95	0.72	0.82

Table 5: LSTM Evaluation Results

8 Topic modeling

In this section, Latent Dirichlet Allocation (LDA) topic modeling is employed to extract meaningful topics from text data within each label category. LDA is a probabilistic model utilized to reveal hidden topics in a collection of documents, making it a valuable tool for text analysis.

1. **Model Selection**: To determine the optimal number of topics for each label category, model selection is performed. This involves training LDA models with varying numbers of topics, ranging from a minimum to a maximum value for each label category present in the dataset.
2. **Metrics and Interpretation**: Each LDA model is evaluated based on several metrics, including perplexity, coherence score, and topic overlap. Perplexity measures how effectively the model predicts the data, while the coherence score assesses the interpretability of the topics. Topic overlap provides insights into the distinctiveness of the identified topics. For each label category, the LDA model with the best combination of these metrics is selected. Finally, the top terms associated with the identified topics in the best model for each label are presented.

This approach uncovers the most relevant and coherent topics within each label category, offering valuable insights into the content of the text data. The LDA models are trained with the following parameters:

- **passes**: passes refer to the number of times the LDA model goes through the entire corpus during training. This variable is set to 15
- **chunksize**: It determines the number of documents to load into memory at once for training. This variable is set to 10000
- **random_state**: It sets the random seed for reproducibility. Using the

same random seed ensures that the results are consistent across different runs. This variable is set to 100

- **iterations**: The number of iterations controls the maximum number of iterations for the LDA training algorithm. This variable is set to 500
- **alpha**: alpha is a hyperparameter that controls the sparsity of document-topic distributions. Setting it to "auto" allows the model to estimate an optimal alpha based on the data.
- **eta**: eta is another hyperparameter that controls the sparsity of topic-word distributions. Like alpha, setting it to "auto" allows the model to estimate an optimal value based on the data.

The metrics used for LDA model selection are the following:

- **Perplexity**: lower perplexity values indicate a better fit of the model to the data
- **Coherence**: higher coherence scores suggest more interpretable and semantically meaningful topics
- **Overlap**: the overlap score measures the degree of similarity or overlap between topics. A lower overlap score indicates less redundancy among topics.

The results are shown in table 6.

Number of topics	5 topics	6 topics	7 topics	8 topics
Perplexity	-8.467	-8.470	-8.465	-8.464
Coherence	0.478	0.470	0.434	0.445
Overlap [%]	7.66	14.603	15.645	1.080
Number of topics	9 topics	10 topics	11 topics	12 topics
Perplexity	-8.476	-8.476	-8.487	-8.479
Coherence	0.440	0.445	0.431	0.438
Overlap [%]	7.955	9.779	9.997	5.297

Table 6: LDA metrics

It is difficult to select a single model that maximize all of the three metrics but a reasonable choice could be between 5 and 7 topics, as these configurations have relatively high coherence scores and moderate overlap scores.

Below are the results of the LDA model with 6 topics related to the 'Science' label, along with a possible interpretation of the topics.

1. **Space and Celestial Bodies**: space, earth, orbit, moon, system, planet, solar
2. **Health and Medicine**: article, patient, disease, doctor, food
3. **Time**: time, much, article
4. **Space Research**: space, program, center, year, health, nasa , launch, national
5. **Electrical Circuits**: grounds, circuits, current, need
6. *unknown*: article, water, henry

Figure 7 shows a representation of the six topics usign word clouds. It is also interesting to see weather the topics are related. This can be done visually by creating a Network Graph whihc is based on the *Jaccard Similarity* between the topics. The results is shown in figure 8. As one can see from figure 8, topics do not have high siminarity and also topic 1 and 4 have zero similarity with the others. This confirms that the six topic are well distinct.

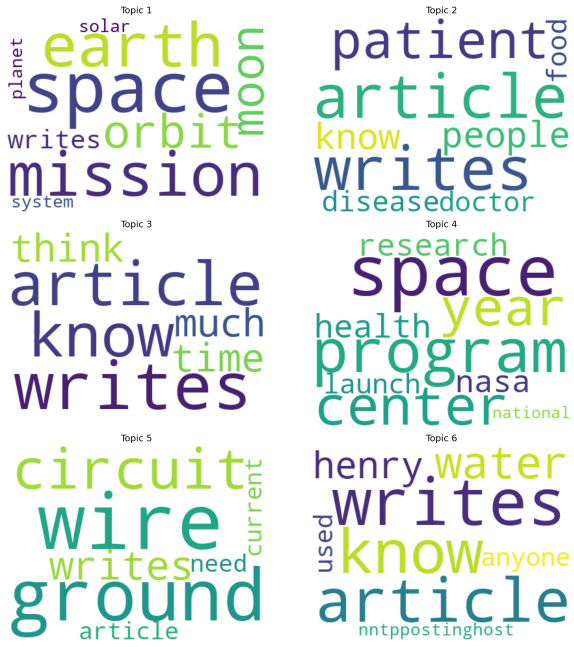

Figure 7: Science topics word clouds

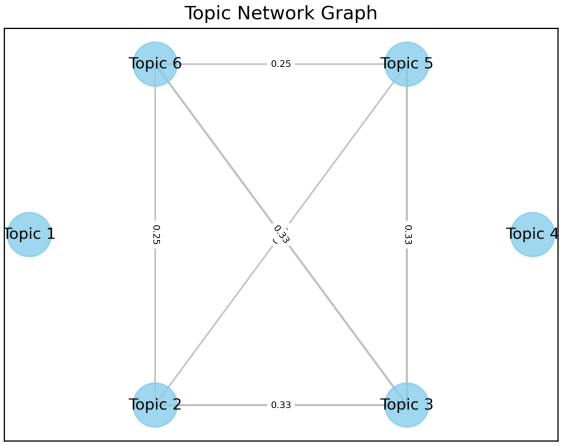

Figure 8: Science topic Network Graph

9 Conclusions

Throughout this study, various machine learning and deep learning models were evaluated for the task of text classification. Among these models, Logistic Regression emerged as the top performer, achieving an accuracy rate of 95%. This result highlights the effectiveness of traditional machine learning techniques in handling text classification tasks. In addition to text classification, topic modeling was explored to extract meaningful topics from the text data. After experimenting with different combinations, the selected model for topic modeling was Latent Dirichlet Allocation (LDA) with six topics. These topics were well-defined, providing valuable insights into the content of the text data. However, it's worth noting that one of the topics presented challenges in terms of clear identification. Overall, the results of this study are promising, with Logistic Regression demonstrating exceptional performance in text classification. The LDA-based topic modeling approach also proved effective in uncovering relevant themes within the text data. While there were minor challenges in topic interpretation, the overall outcomes of this analysis are encouraging and lay the foundation for further exploration and refinement of text analysis techniques.