# Text Mining and Natural Language Processing

2023-2024

**Yusuf Mehmet COLAK and associated id: 513221**

*[TC-YMC]*

## Introduction

The goal of this project is to develop a sentiment analysis model to classify tweets into different sentiment categories (Positive, Negative, Neutral, Irrelevant). The model utilizes machine learning techniques to process and analyze textual data from Twitter. Sentiment analysis can be valuable for understanding public opinion, customer feedback, and various other applications.

## Data

I used a dataset from Twitter, consisting of training and validation sets. The dataset had unusable-nonsense columns, lots of missing values and duplicated values. The preprocessing of the data has been done by some steps which are:

1. Drop Unnecessary columns

2. Rearrange Names

3. Dropping missing values

4. Lowercasing

5. Removing URLs-Links- HTMLS

6. Removing numeric digits

7. Removing Punctioations

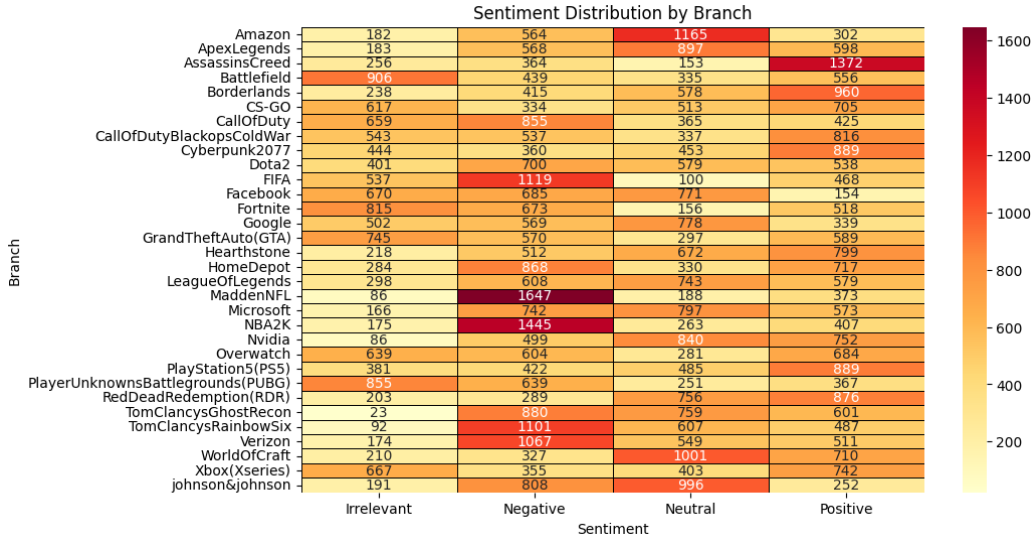
8. Removing Stopwords

9. Removing Emojies

Followed by word tokenization. In my case it was the tokanization of the remaining textual data after pre-processing from tweets.

## Data Visualization

- **Word Cloud**: The most frequent words in the dataset were visualized using a word cloud to understand common terms and potential noise.
- **Distribution Graphs**: Graphs showing the distribution the data from different aspects.




By this chart we can see the distributions of the sentiments among the all branches. The elements are the sum of the negative, positive etc. comments about the regarding branch.

**The EXTREMES are:**

Sentiment 'Irrelevant':

Minimum value '23' at Branch 'TomClancysGhostRecon'

Maximum value '906' at Branch 'Battlefield'

Sentiment 'Negative':

Minimum value '289' at Branch 'RedDeadRedemption(RDR)'

Maximum value '1647' at Branch 'MaddenNFL'

Sentiment 'Neutral':

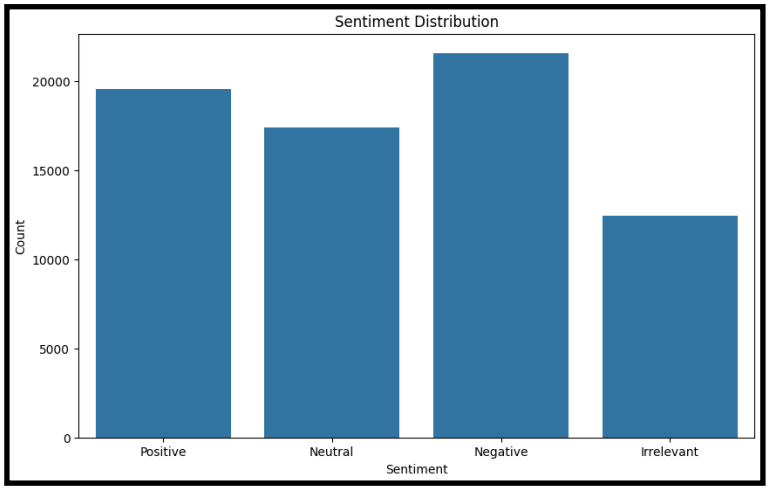
Minimum value '100' at Branch 'FIFA'

Maximum value '1165' at Branch 'Amazon'

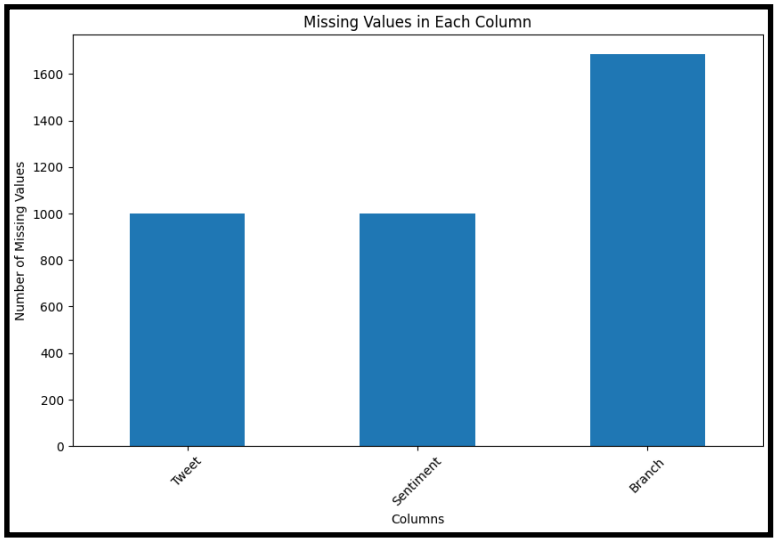
Sentiment 'Positive':

Minimum value '154' at Branch 'Facebook'

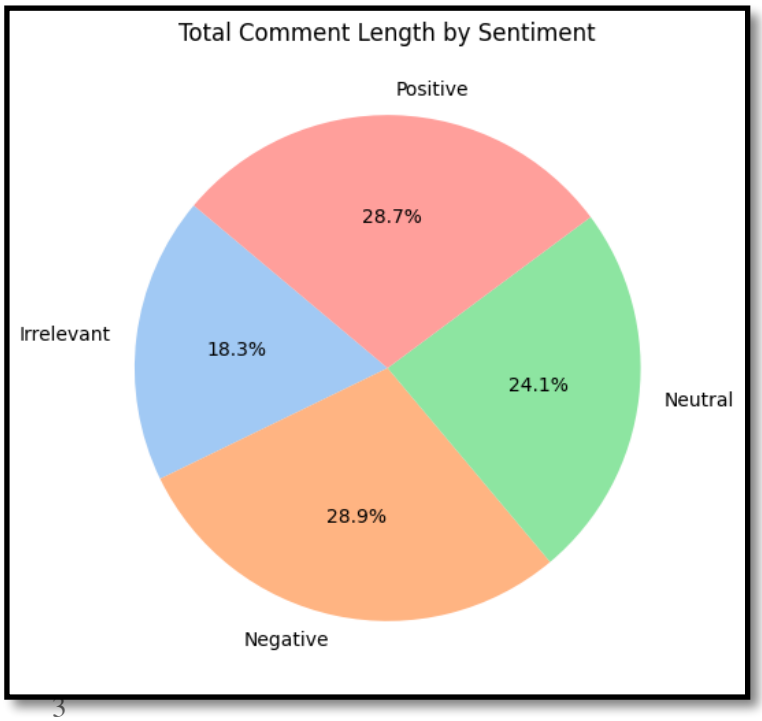
Maximum value '1372' at Branch 'AssassinsCreed'




From the chart we Cleary understand that the labels are distributed mostly equally. Which is very good for a text-classification task. Only irrelevant sentiments are less compared to others.

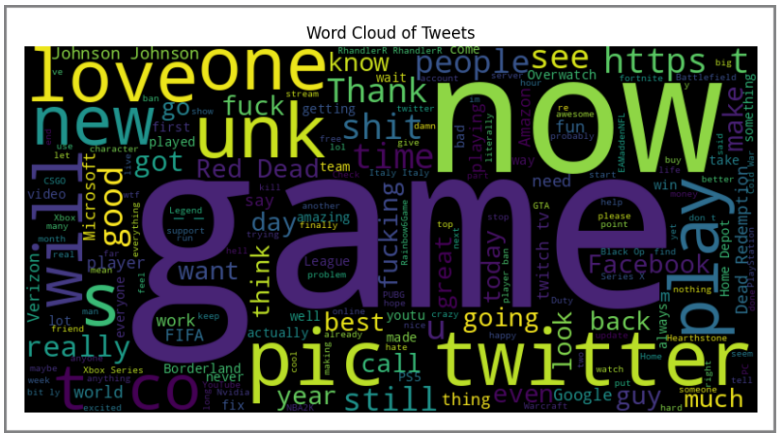



From the chart we can see that we have great amount of missing values especially for the Branches. Since they would create problem for the next task we are dropping them. Thus, We will have more clear dataset to work on.




This pie chart shows the average comment length for each sentiment category. By analyzing this data, we can determine whether people tend to write longer or shorter comments based on their feelings.

Beside social perspective, it highlights the volume of information available for each sentiment. For instance, we might expect a bias towards negative sentiments if we have more comments categorized as negative compared to those labeled as irrelevant. This insight is crucial for ensuring balanced sentiment analysis and understanding public opinion accurately.




From this Word Cloud we gain the information about the most repeated words, which are represented bigger than the others.

## Methodology

## Pre-Processing

| | 2401 | Borderlands | Positive | im getting on borderlands and i will murder you all , | 3364 | Facebook | Irrelevant | I mentioned on Facebook that I was struggling for motivation to go for a run the other day, which has been translated by Tom’s great auntie as ‘Hayley can’t get out of bed’ and told to his grandma, who now thinks I’m a lazy, terrible person 😭 |
|---|---|---|---|---|---|---|---|---|
| 0 | 2401.0 | Borderlands | Positive | I am coming to the borders and I will kill you... | NaN | NaN | NaN | NaN |
| 1 | 2401.0 | Borderlands | Positive | im getting on borderlands and i will kill you ... | NaN | NaN | NaN | NaN |
| 2 | 2401.0 | Borderlands | Positive | im coming on borderlands and i will murder you... | NaN | NaN | NaN | NaN |
| 3 | 2401.0 | Borderlands | Positive | im getting on borderlands 2 and i will murder ... | NaN | NaN | NaN | NaN |
| 4 | 2401.0 | Borderlands | Positive | im getting into borderlands and i can murder y... | NaN | NaN | NaN | NaN |

First of all I started pre-processing of the data. As you can see form the image there was a dumb column with irrelevant text and NaN values. And some other columns that will not contribute the learning process. I dropped them. Lastly I renamed:

**-Postive -> Sentiment**

**-‘im getting on borderlands and i will murder you all ‘ -> ‘Tweet'**

**-Borderlands -> Branch**

| | Tweet | Sentiment | Branch | Comment_Length |
|---|---|---|---|---|
| 0 | Borderlands | Positive | I am coming to the borders and I will kill you... | 11 |
| 1 | Borderlands | Positive | im getting on borderlands and i will kill you ... | 11 |
| 2 | Borderlands | Positive | im coming on borderlands and i will murder you... | 11 |
| 3 | Borderlands | Positive | im getting on borderlands 2 and i will murder ... | 11 |
| 4 | Borderlands | Positive | im getting into borderlands and i can murder y... | 11 |

Then I created a detailed analize on the data for a broader insight of the material. Afterwards, I started to Pre-Processing of the **Tweet** column. THE STEPS OF PRE-PROCESSING:

1. Drop Unnecessary columns
2. Rearrange Names
3. Dropping missing values
4. Lowercasing
5. Removing URLs-Links- HTMLS
6. Removing numeric digits
7. Removing Punctioations
8. Removing Stopwords
9. Removing Emojies

---

## Tokenization

Before Pre-Processing and tokenization : '' just like the windows partition of my mac is like 6 years behind on its drivers so you have no idea how i didn't notice''

After Pre-Processing and tokenization :

```
'like',
 'windows',
 'partition',
 'mac',
 'like',
 'years',
 'behind',
 'drivers',
 'idea',
```

'didn',
''',
't',
'notice']

---

## Word2Vec

After tokenization I have started the vectorization (numerical representation ) of the text data (corpus). I have split the data as train and test.  Then I convert the textual data to vectors using TD-IDF.

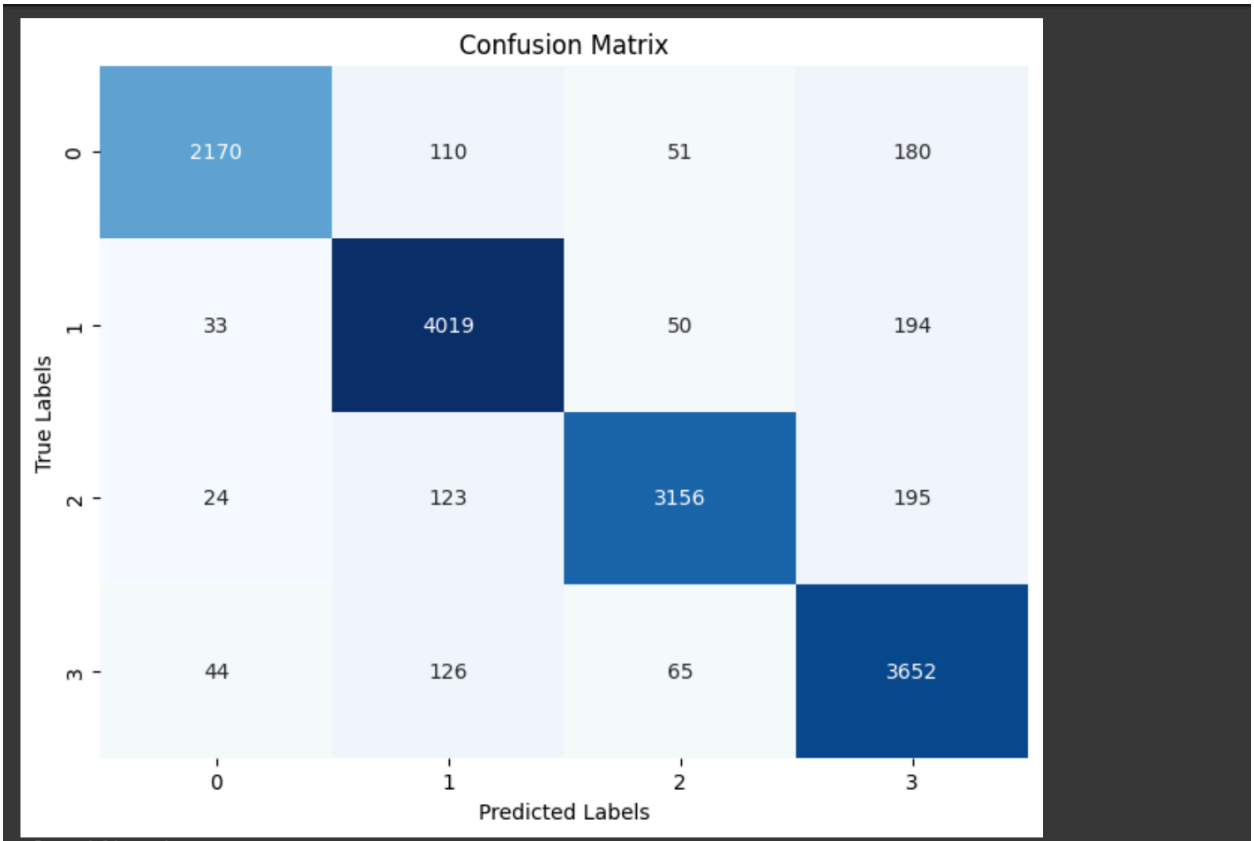
---

## RandomForestClassifier

I used random forest classifier as ML algorithm for multiple class classification, and obtain:

Test Accuracy: 0.9157976324689966

Which was a very sufficient result. There is the confusion matrix below




```
Classification Report:
              precision    recall  f1-score   support

  Irrelevant       0.96      0.86      0.91      2511
    Negative       0.92      0.94      0.93      4296
     Neutral       0.95      0.90      0.93      3498
    Positive       0.87      0.94      0.90      3887

    accuracy                           0.92     14192
   macro avg       0.92      0.91      0.92     14192
weighted avg       0.92      0.92      0.92     14192
```

## Confusion Matrix Analysis:

1. Class 0 (Irrelevant)

- True Positives (TP): 2170

- False Positives (FP): 51 (Predicted as Irrelevant but true class is different)

- False Negatives (FN): 341 (Predicted as other classes but true class is Irrelevant)

- Observation: Majority of irrelevant instances are correctly classified, but there are some confusions with other classes, especially class 3 (Positive).

2. Class 1 (Negative):

- True Positives (TP): 4019

- False Positives (FP): 50

- False Negatives (FN): 277

- Observation: Negative sentiments are very well classified with minimal errors, indicating strong model performance for this class.

3. Class 2 (Neutral):

- True Positives (TP): 3156

- False Positives (FP): 195

- False Negatives (FN): 342

- Observation: Neutral sentiments also show good classification but with some confusion with both irrelevant and positive classes.

4. Class 3 (Positive):

- True Positives (TP): 3652

- False Positives (FP): 65

- False Negatives (FN): 235

- Observation: Positive sentiments are accurately classified but some instances are confused with irrelevant and neutral classes.

Classification Report Analysis:

1. Precision : Measures the accuracy of the positive predictions.

- Irrelevant: 0.96

- Negative: 0.92

- Neutral: 0.95

- Positive: 0.87

- Observation: The classifier is highly precise for irrelevant and neutral classes, reasonably precise for negative, and somewhat lower for positive.

2. Recall: Measures the ability to find all relevant instances.

- Irrelevant: 0.86

- Negative: 0.94

- Neutral: 0.90

- Positive: 0.94

- Observation: The recall is high for negative and positive classes, indicating the model can identify most of these sentiments correctly.

3. F1-Score: Harmonic mean of precision and recall.

- Irrelevant: 0.91

- Negative: 0.93

- Neutral: 0.93

- Positive: 0.90

- Observation: The F1-scores are quite balanced across all classes, showing that the classifier maintains a good balance between precision and recall.

Conclusion:

The Random Forest classifier performs well on this sentiment analysis task with a high overall accuracy of 92%. It shows strong performance across all classes, particularly for negative and neutral sentiments. The precision and recall are well-balanced, and the model is effective in identifying irrelevant, negative, and neutral sentiments, with slightly less but still good performance for positive sentiments.

---

## Saving the model and using random tweet inputs

I decided to save the model and try to make sentiment analize with random tweets written by me. And I observed that the model doesn't capture sarcasm or jokes or mixed statements. The CUPHEAD tweet can be an example:

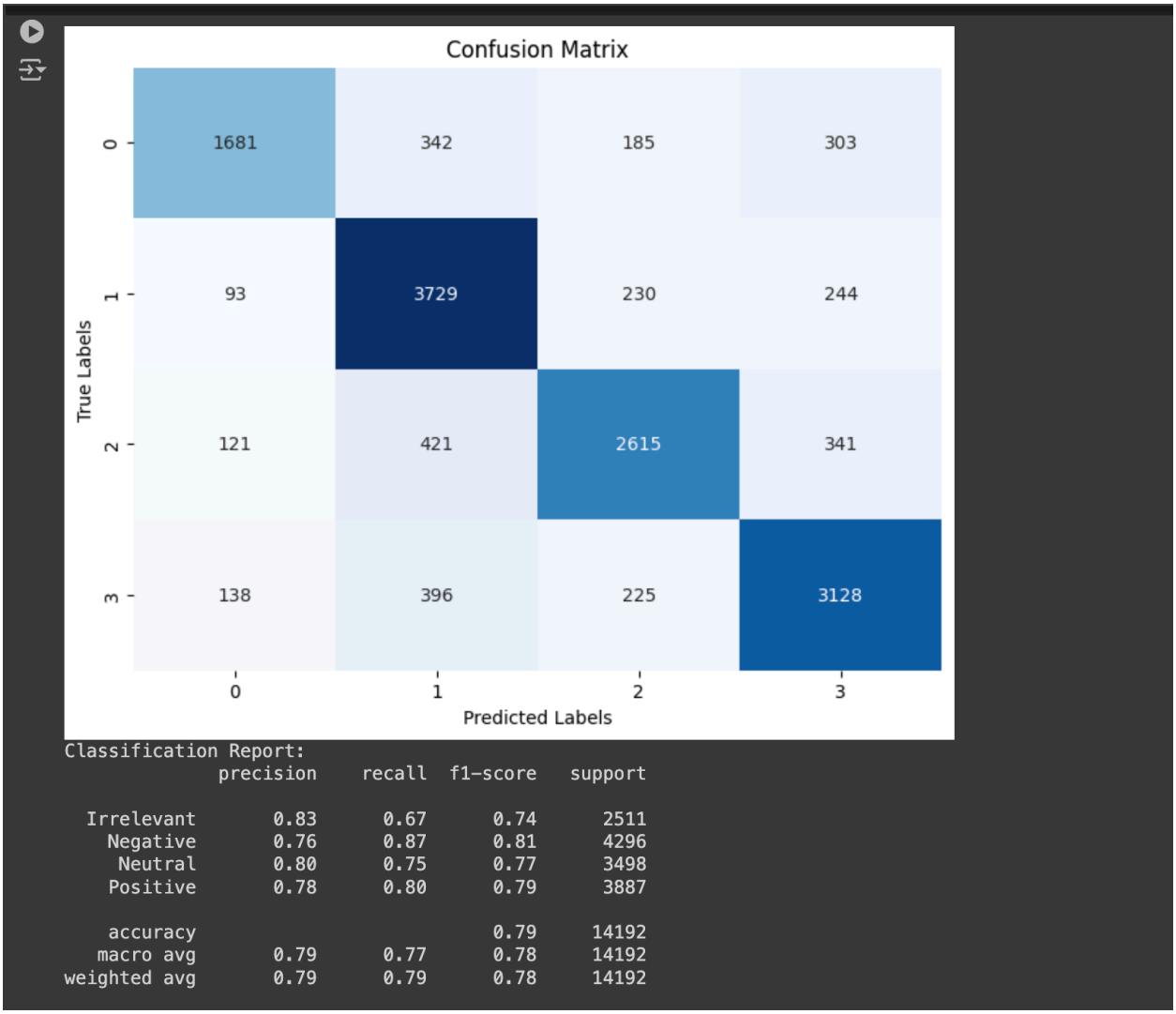

---

## Linear Regression Model

I created a Linear regression model but this time I have used a count vectorizer. From the model I get this results:

Logistic Regression Accuracy: 0.7858652762119503
Logistic Regression Precision: 0.7930023618130748
Logistic Regression Recall: 0.7724437320618135

Which was clearly lower than Random Forest model. Here is the Confusion matrix :




Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Irrelevant | 0.83 | 0.67 | 0.74 | 2511 |
| Negative | 0.76 | 0.87 | 0.81 | 4296 |
| Neutral | 0.80 | 0.75 | 0.77 | 3498 |
| Positive | 0.78 | 0.80 | 0.79 | 3887 |
| accuracy | | | 0.79 | 14192 |
| macro avg | 0.79 | 0.77 | 0.78 | 14192 |
| weighted avg | 0.79 | 0.79 | 0.78 | 14192 |

## Confusion Matrix Analysis:

1. **Class 0 (Irrelevant)**:
    - **True Positives (TP)**: 1681
    - **False Positives (FP)**: 185
    - **False Negatives (FN)**: 830
    - **Observation**: Many irrelevant instances are misclassified, with significant confusion, especially with classes 1 (Negative) and 3 (Positive).
2. **Class 1 (Negative)**:
    - **True Positives (TP)**: 3729
    - **False Positives (FP)**: 230
    - **False Negatives (FN)**: 567

- **Observation**: Negative sentiments are relatively well classified but still have considerable misclassifications compared to the Random Forest model.

3. **Class 2 (Neutral)**:
    - **True Positives (TP)**: 2615
    - **False Positives (FP)**: 341
    - **False Negatives (FN)**: 883
    - **Observation**: Neutral sentiments have a higher rate of misclassification, indicating the model struggles with distinguishing these instances.
4. **Class 3 (Positive)**:
    - **True Positives (TP)**: 3128
    - **False Positives (FP)**: 225
    - **False Negatives (FN)**: 759
    - **Observation**: Positive sentiments are not as accurately classified as in the Random Forest model, with a significant number of instances misclassified as other classes.

## Classification Report Analysis:

1. **Precision**: Measures the accuracy of the positive predictions.
    - Irrelevant: 0.83
    - Negative: 0.76
    - Neutral: 0.80
    - Positive: 0.78
    - **Observation**: Precision is notably lower for negative sentiments, indicating a higher rate of false positives.
2. **Recall**: Measures the ability to find all relevant instances.
    - Irrelevant: 0.67
    - Negative: 0.87
    - Neutral: 0.75
    - Positive: 0.80
    - **Observation**: Recall is lower for irrelevant and neutral classes, indicating many instances are missed.
3. **F1-Score**: Harmonic mean of precision and recall.
    - Irrelevant: 0.74
    - Negative: 0.81
    - Neutral: 0.77
    - Positive: 0.79
    - **Observation**: F1-scores are generally lower across all classes compared to the Random Forest model.

4. **Support**: Number of actual occurrences of the class in the dataset.
    - Irrelevant: 2511
    - Negative: 4296
    - Neutral: 3498
    - Positive: 3887
    - **Observation**: The distribution of classes in the dataset remains the same, which allows for a direct comparison of model performance.

## Overall Model Performance:

- **Accuracy**: 0.79
- **Macro Avg**: Average performance metric across all classes.
    - Precision: 0.79
    - Recall: 0.77
    - F1-Score: 0.78
- **Weighted Avg**: Average performance metric considering the support of each class.
    - Precision: 0.79
    - Recall: 0.79
    - F1-Score: 0.78

## Possible Reasons for Worse Performance Compared to Random Forest:

1. **Model Complexity**: Linear regression is a simpler model compared to Random Forest. Random Forest can capture more complex patterns and interactions in the data because it is an ensemble of decision trees.

2. **Feature Interactions**: Random Forest can handle non-linear relationships and interactions between features better than linear regression, which assumes a linear relationship.

3. **Overfitting vs. Underfitting**: Random Forest is less prone to overfitting due to averaging over many trees, whereas linear regression might underfit if the true relationship between features and the target is non-linear or complex.

4. **Noise Handling**: Random Forest is generally better at handling noise in the data due to its ensemble nature, whereas linear regression can be more sensitive to outliers and noise.

5. **Feature Importance**: Random Forest can effectively determine feature importance and reduce the impact of irrelevant features, improving performance. Linear regression treats all features equally unless regularized.

## Conclusion:

The Random Forest classifier outperforms linear regression in this sentiment analysis task due to its ability to capture complex patterns, handle feature interactions, and manage noise more effectively. Linear regression, being a simpler model, struggles with these aspects, leading to higher misclassification rates and overall lower performance metrics.

## Conclusion

In summary, this project successfully developed a sentiment analysis model to classify tweets into various sentiment categories: Positive, Negative, Neutral, and Irrelevant. The model utilized a comprehensive preprocessing pipeline, data visualization techniques, and multiple machine learning algorithms to achieve high classification accuracy.

The Random Forest classifier demonstrated superior performance with a test accuracy of approximately 92%, demonstrating its ability to handle complex patterns and interactions within the data. The model's precision and recall were well-balanced across all classes, indicating its superiority in identifying and correctly classifying different sentiments. On the other hand, the Linear Regression model, despite being simpler, showed lower performance metrics, highlighting the importance of choosing the right model for complex text classification tasks.

Despite these achievements, the project identified areas for improvement. Incorporating additional techniques such as hyperparameter tuning, cross-validation, and exploring more advanced models like deep learning architectures could further enhance performance.

Additionally, addressing the model's limitations in capturing nuances like sarcasm or mixed sentiments remains a crucial challenge for future work.

---

## RESOURCES

This project has been made by me but I get help from a lots of resources and I am going to share the links of the materials I used:

Google dev: https://developers.google.com/machine-learning/guides/text-classification/step-1

**Lab1 - An Introduction to Textual Data.ipynb**
https://drive.google.com/file/d/1YKomhhH8KgBMZBMcwIuG-yRBELbdHEl7/view?usp=sharing

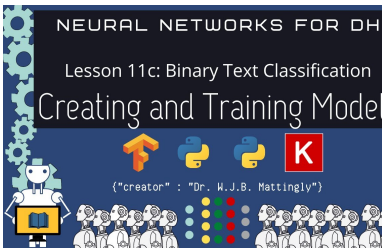
Lab3 - SVD - Word2Vec - Text classification.ipynb :https://drive.google.com/file/d/15JxvOJG8PhowmCQZg3HNnZLTjpXZH51y/view?usp=sharing

**Logistic Regression**
https://web.stanford.edu/~jurafsky/slp3/5.pdf

Github: https://github.com/pik1989/Sentiment-Analysis/blob/main/Data Exploration & Modelling.ipynb

YouTube:




Tensorflow: https://www.tensorflow.org/tutorials/keras/text_classification

Blog: https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-language-model-nlp-python-code/

Hugging-face: https://huggingface.co/docs/datasets/tutorial

## AI policies

I get assistance from Chat-GPT-4o model of OpenAI. Generally I tried to get advices and comments on the code. Also I used it to get creative ideas for the project and to enhance some parts of the report . Beside that I get advantage of the materials that I have provided in the conclusion part.