**Report- “Developing a Machine Learning Model for Sentiment Analysis of Positive and Negative Tweets Using the ‘Twitter Sentiment Analysis’ Dataset”**

**Team members-**

**Dhiraj Jethani- F415122**

**Kavinka Fernando - F416208**

**Assan Begimov - F433704**

**Michel Benedict Sanga - F432039**

**Habiba Abdelaziz - F433950**

**1. Introduction**

The following report presents a comprehensive summary of the executed sentiment analysis, which employed textual social media data from Twitter and aimed to assess and classify them into positive and negative user sentiments. The dataset used in the following project is the “sentiment140”, widely used for sentiment analysis tasks, containing 1,600,000 tweets.

**Structure of the Report**

1. Introduction to the dataset and preprocessing performed.
2. EDA and explanation of the Model Algorithms.
3. Standard classification metrics utilised to evaluate the machine learning models.
4. Analysis accompanied with chart illustrations, heatmaps and ROC curves.
5. Summary of the key outputs of the analysis.
6. Possible advancements proposed for future framework enhancement.

**2. Dataset description**

Overall, the dataset consists of 1,600,000 user-generated entries, specifically “tweets”, from one of the largest social microblog platforms popular for public opinion expression and real-time trend tracking, Twitter, or as it called now, X (Ripamonti, 2018). Apart from the text, the dataset is structured with other details and features for broader analysis of inputted UGC and correlation building, including target, timestamps, user data, flag and tweet’s unique ID. Prelabelled sentiment targets facilitate training and enable the model to understand patterns, detect similarity and correlations between text features of the tweets and their sentiment classification, also outlining the benchmarks for accurate predictions based on new, unseen inputs. Before using the data, the text inputs were simplified and cleaned from all the unnecessary noise elements that might confuse the trained model, like special characters, URLs or text gaps, preserving tweets’ genuine emotional nature, as well as filtered out all the irrelevant tweets and spam. As a result, the preprocessing of the outsourced dataset not only improved its quality but also enhanced the accuracy and interpretability of the employed machine learning models at the feature engineering stage of the project.

**3. Exploratory Data Analysis**

The dataset contained varied sentiment value ranging from 0-4 depicting:

**0= Negative**

**2= Neutral**

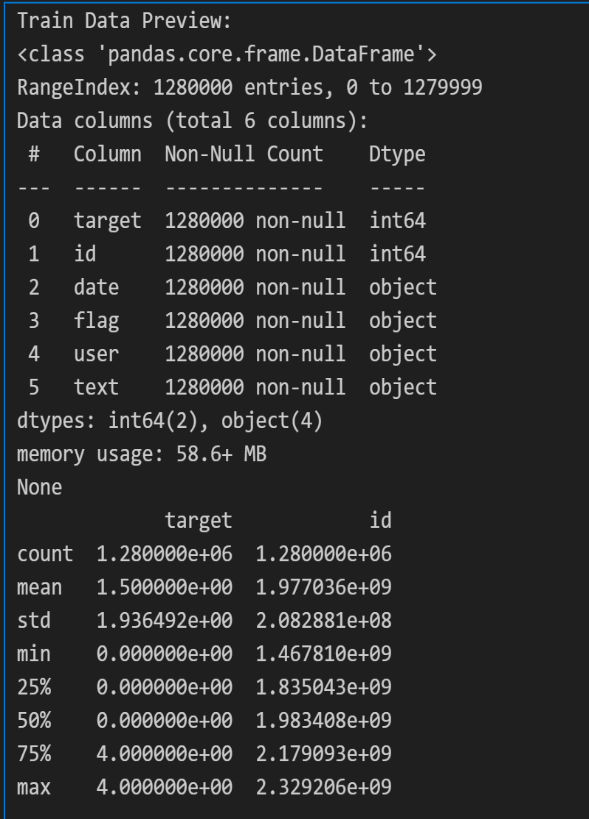
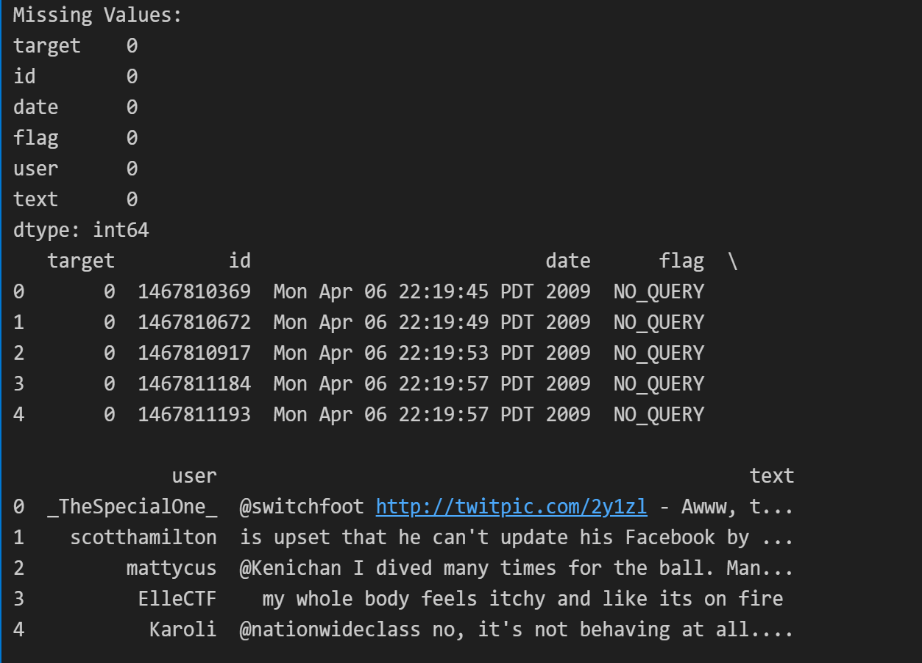
**4= Positive**

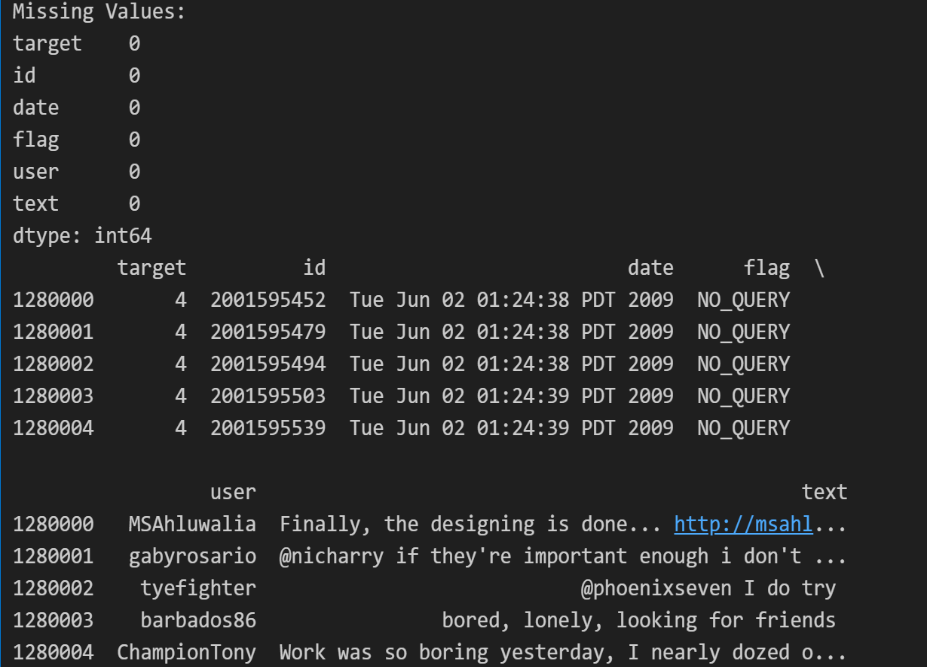
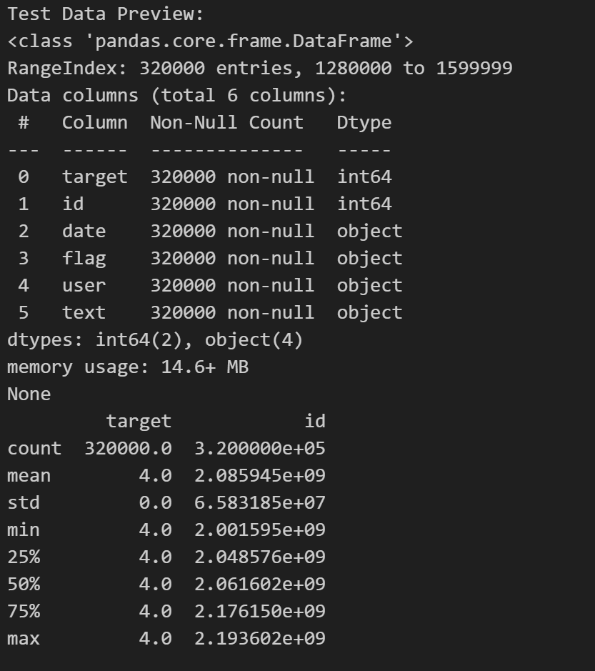
However, for our program we are only using the negative and positive polarities of the data.   
The dataset consists of 1.6 million entries with 6 columns: Target, Id, Date, Flag, User and Text. After splitting the dataset, the training set consists of 1.28 million entries and the test set has 320,000 entries, with no missing vales across any column.

**Statistical Observations**

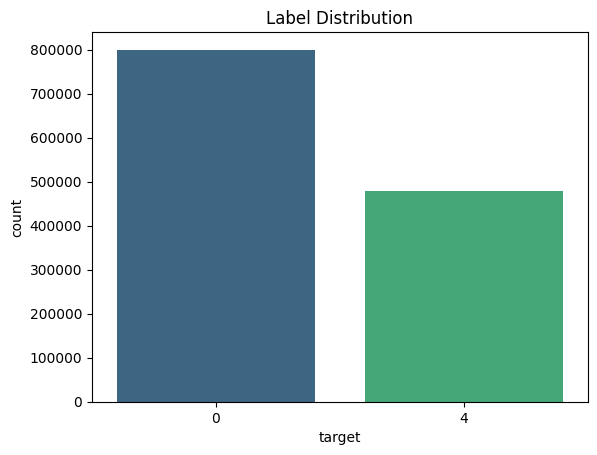
* **Target Column**: The target variable in the training data ranges from 0 to 4, with a mean of 1.5 and a standard deviation of 1.94, suggesting a skew towards lower values (0). The test data is more uniform, with all target values equal to 4, indicating a significant change or filter applied in the test set.
* **ID Column**: The id values are large integers ranging from approximately 1.47 billion to 2.33 billion in the training set. The average value is around 1.98 billion, with a standard deviation of 208 million. This suggests that the id values are unique identifiers with a considerable range across the data.
* **Date Column**: The date column, although non-null, is represented as an object type (string), requiring conversion to a datetime format to allow temporal analysis.
* **Flag Column**: The flag column consists of categorical values (e.g., NO\_QUERY), indicating specific data conditions, potentially useful for filtering or grouping.
* **Text Column**: The text column contains user-generated content, which would need text preprocessing (e.g., tokenization, stop-word removal) for any natural language processing tasks.

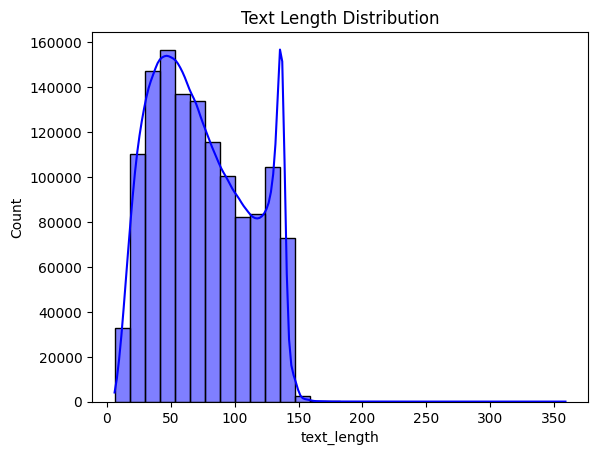
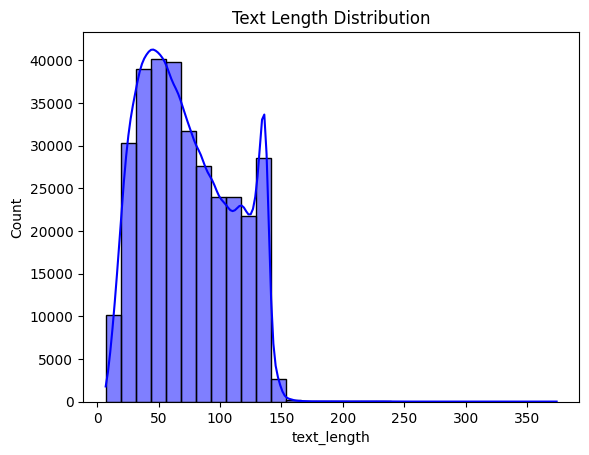
The dataset's target column ranges from 0 to 4, with a mean of 1.5 and a standard deviation of 1.94. The id column ranges from 1.468 to 2.329 billion, with a mean of 1.977 billion and a standard deviation of 208 million, showing considerable variation in identifier values.



**Visualization for label distribution**

****

**Feature analysis  
** ****

Training dataset Testing dataset

**4. Feauture Engineering And Model Implementation**

**4.1 Feature Engineering Techniques**

1. **Text Vectorization:** The dataset includes text data. So to convert text into numerical data “CountVectorizer” is used. Also sparse matrix representation (csr\_matrix) is used to prevent memory errors .
2. **Data cleaning:** Removal of irrelevant elements such as links, mentions and special characters from tweets using regular expressions(re.sub). Text is converted to lowercase and stemmed using nltk.PorterStemmer, and stopwords are removed.
3. **Train-Test Split:** The Dataset is split into an 80% training and 20% test set .
4. **Scaling:** “StandardScalar” is used “with\_mean=False” to standardize the sparse matrix so as to ensure all features contribute equally.

**4.2 Machine Learning Models Used**

1. **Logistic regression**

* Used for the simplicity and efficiency of the model in binary classification tasks such as possitive or negative sentiment analysis.
* It is also ideal as a baseline model due to its simple interpretablity.

1. **Random forrest Classifier**

* It is an ensemble model that bulds multiple decision trees and combines their prediction.
* It is also robust to overfitting and also works well with large datasets. Very beneficial to handle non linear data.

1. **Decision tree Classifier**

* It is a tree based model that splits data into decision nodes.
* Very easy to interpret and also provides insight into feature importance, making it great for feature analysis.

1. **SGD Classifier**

* It is a linear classifier that uses Stochastic Gradient Descent for training.
* It is particularly effective for large datasets and scalable to high-dimensional data. The model is suitable for classification tasks with a large number of features, like text data.

1. **XGBoost Classifier**

* It is a powerful, gradient-boosting algorithm known for its speed and accuracy.
* It is well-suited for large datasets and iteratively improves model performance by focusing on minimizing errors from previous iterations.

**Model Implementation Process:** Models are trained on the training set, validated with performance metrics such as **accuracy**, **F1-score**, and the **confusion matrix** to assess model precision and recall. Hyperparameter tuning is performed to optimize the models, ensuring the best possible performance.

1. **Model Evaluation and Performance**

Several classification methods, such as Random Forest, Logistic Regression, Decision Tree, Stochastic Gradient Descent (SGD) Classifier, and XGBoost, were used in the model evaluation for the Twitter Sentiment Analysis project.  
  
**Validation and Metrics Used**  
An 80/20 ratio was used to divide the dataset into training and validation sets. Accuracy, weighted F1 Score, and a Confusion Matrix were standard evaluation criteria. While the F1 Score balanced precision and recall for a more complete performance metric, accuracy measured the percentage of accurate predictions. Additional information on imbalances and classification problems was revealed by the confusion matrix.  
  
**Performance Results**

* Overfitting was evident when the **Random Forest** Classifier's validation accuracy fell to 76% from its high training accuracy of 97%.
* With an F1 Score of 0.77 and training and validation accuracy of 78% and 77%, respectively, **Logistic Regression** demonstrated balanced performance.
* With a 97% training accuracy and a 71% validation accuracy, the **Decision Tree** was significantly overfit
* With an F1 Score of 0.75 and an accuracy of 76%, **Stochastic gradient descent (SGD)** demonstrated a moderate level of performance.
* **XGBoost** handled complicated data well while having a lower accuracy (74%).

Strengths: The accuracy and generalisation of logistic regression were very good. Strong learning capacity was indicated by the high training accuracy of Random Forest and Decision Tree. Despite its reduced accuracy, XGBoost handled complex data well.

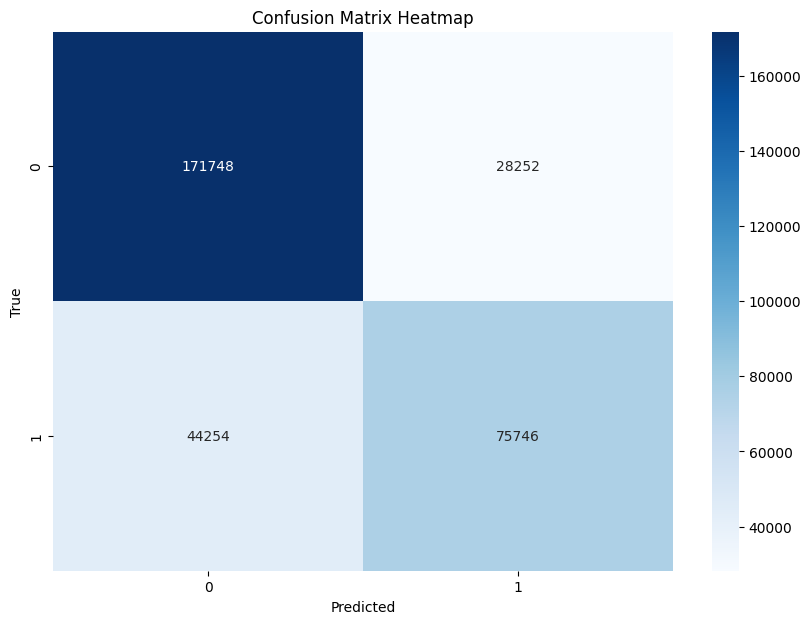
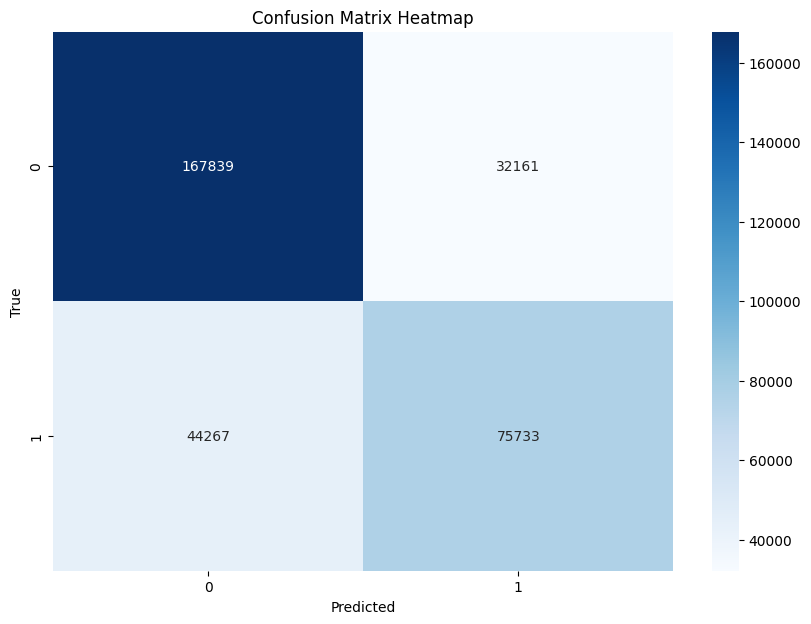
Limitations: Decision trees and random forests overfitted and had poorer validation accuracy. Despite being effective on big datasets, SGD lacked the stability of Logistic Regression, whereas XGBoost had a significant false negative rate.

1. **Discussions and Interpretation**

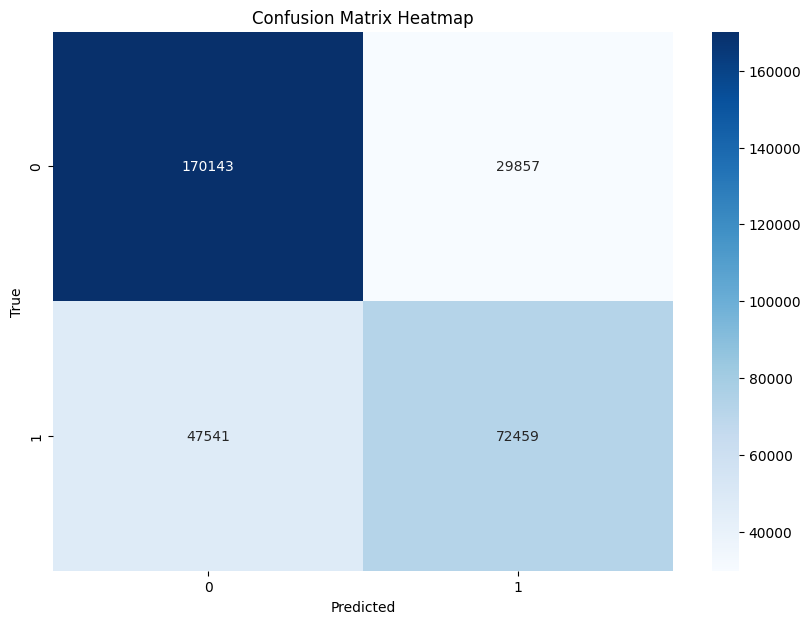
The primary objective of this project was to classify tweets into positive or negative sentiment categories, with a focus on detecting hate speech as an extreme form of negative sentiment. EDA and model evaluation revealed several key insights:

1. Balanced Dataset: The class distribution was balanced, reducing the need for resampling techniques and enabling direct focus on model optimization.
2. Text Characteristics: Tweets were concise, typically between 150–200 characters. Shorter tweets often reflected extreme sentiments, aligning with hate speech characteristics.
3. Data Preprocessing: The absence of missing values simplified preparation, allowing emphasis on cleaning text (e.g., removing punctuation and stop words) and eliminating irrelevant metadata like user and date.

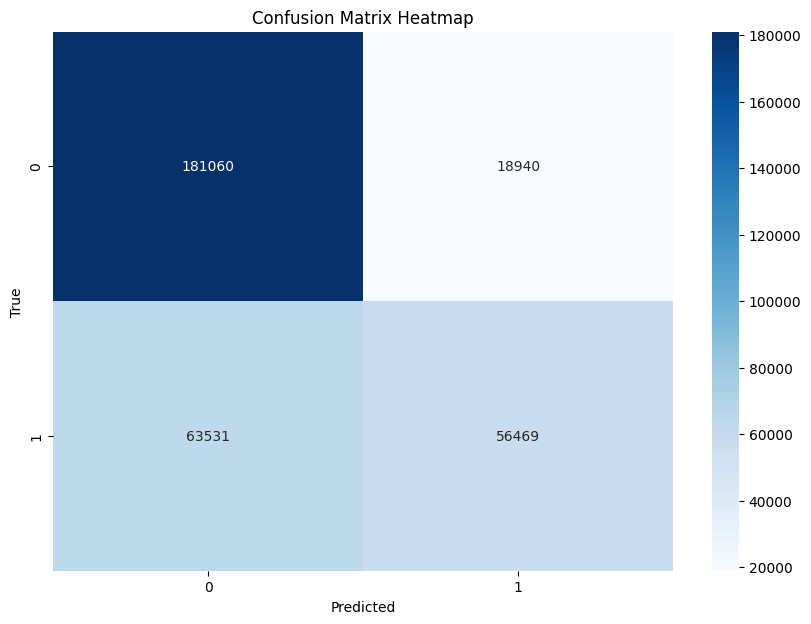
Broader Implications: Logistic Regression achieved the most balanced performance, while other models like Random Forest showed overfitting. The results highlight practical applications in real-world sentiment analysis and emphasize the need for advanced context-aware models, like transformers, to improve classification accuracy for ambiguous tweets.



Random forest classifier Logistic regression



Decision tree classifier Stochastic Gradient Descent Classifier

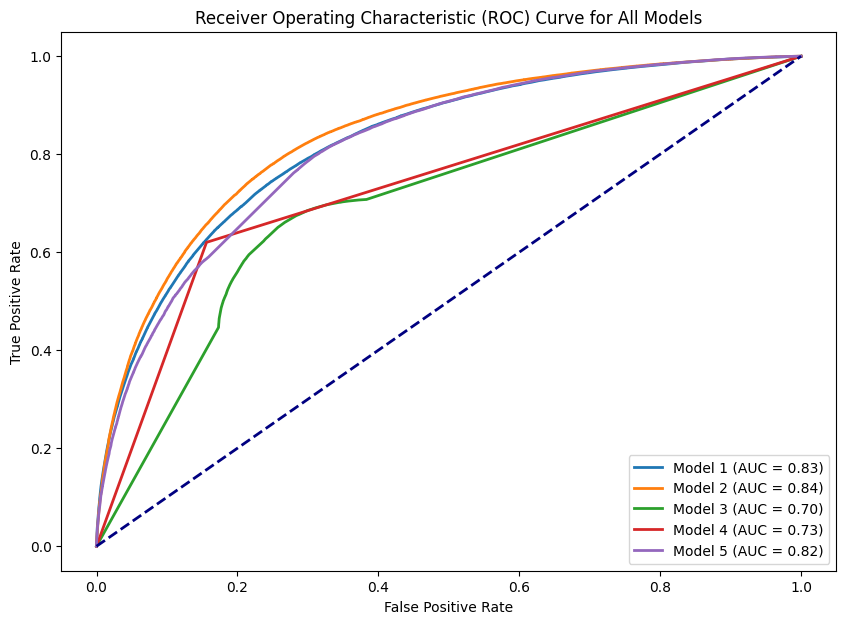


XGBoost classifier

1. **Conclusion and future directions**

The project successfully classified tweets into positive and negative sentiments, with a focus on detecting hate speech. Logistic Regression emerged as the most balanced model, achieving a strong trade-off between accuracy and generalizability. Random Forest and Decision Tree models demonstrated high training accuracy but suffered from overfitting, while XGBoost and SGD offered moderate performance, indicating their suitability for complex and large datasets.

Future improvements include implementing advanced context-aware models like transformers (e.g., BERT) to better handle ambiguous sentiments and longer texts. Additionally, addressing real-world challenges such as imbalanced datasets or missing data can further enhance the model's robustness and adaptability.

****

**8. Refrences**

**1. NumPy**: NumPy. (n.d.). *NumPy Documentation*. Retrieved from <https://numpy.org/doc/>

**2. Pandas:** The pandas development team. (2023). *Pandas Documentation*. Retrieved from <https://pandas.pydata.org/pandas-docs/stable/>

**3. Matplotlib:** Matplotlib. (n.d.). *Matplotlib Documentation*. Retrieved from <https://matplotlib.org/stable/contents.html>

4. **Seaborn**: Seaborn. (n.d.). *Seaborn Documentation*. Retrieved from <https://seaborn.pydata.org/>

5. **SciPy**: SciPy. (n.d.). *SciPy Documentation*. Retrieved from <https://docs.scipy.org/doc/scipy/>

6. **NLTK**: Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media.

7. **Scikit-learn**: Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830. Retrieved from <https://scikit-learn.org/>

8. **XGBoost**: Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794). ACM. https://doi.org/10.1145/2939672.2939785

9. **WordCloud**: Andreas Mueller. (2018). *WordCloud Documentation*. Retrieved from <https://github.com/amueller/word_cloud>

10. **Python Libraries**: Python Software Foundation. (n.d.). *Python Documentation*. Retrieved from <https://docs.python.org/>

11. **Kaggle Dataset:** Kazanova, S. (2017). *Sentiment140 dataset with 1.6 million tweets*. Kaggle. Retrieved from <https://www.kaggle.com/datasets/kazanova/sentiment140>

12**. Github:** Sharma, R. (2020). *Twitter Sentiment Analysis*. GitHub. Retrieved from <https://github.com/sharmaroshan/Twitter-Sentiment-Analysis>. (for understanding on how design a sentiment analysis model using twitter datasets )

**Source code**

