**Spotify Sentiment Analysis**

****

**Conducted By:**

Y. Govardhan Reddy

**Contents**

* About Project
* Business Understanding
* Exploratory Data Analysis
* Data Preprocessing
* Data Preparation
* Model Building
* Model Evaluation
* Conclusion

**About Project**

This project analyzes over **52,702** Spotify user reviews from the **Google Play Store** to understand user sentiment and feedback themes. By identifying positive and negative trends, we can highlight specific issues that impact user satisfaction, such as technical difficulties and feature requests.

The analysis reveals common words and phrases in user feedback, helping Spotify's product development team prioritize updates that enhance user experience. By focusing on user needs, Spotify can create a more enjoyable app and build customer loyalty. Ultimately, these insights will guide Spotify in improving user satisfaction and engagement.

**Business Understanding**

Spotify, as a leading music streaming platform, aims to continually improve user experience by addressing key areas of concern and enhancing its app features. To achieve this, understanding user sentiment through analysis of customer reviews is essential. By examining feedback from over 52,702 user reviews on the Google Play Store, Spotify can identify positive and negative trends, as well as common issues that users face.

These insights provide a foundation for actionable improvements, allowing Spotify’s product development team to address technical challenges, introduce desired features, and optimize the app's functionality. In turn, this approach supports Spotify’s overarching business goal: to increase user satisfaction and engagement, retain loyal customers, and stay competitive in the rapidly evolving music streaming market.

**Exploratory Data Analysis :**

In the exploratory phase, we focused on understanding the dataset's structure and quality. Key activities included:

**1.Missing Values Removal :** Ensured data integrity by identifying and removing any entries with missing values, which could distort our analysis.

**2.** **Duplicate Removal :** Eliminated duplicate reviews to prevent skewed results and ensure that our sentiment analysis reflects unique user opinions.

**3. Word Cloud Visualization :**

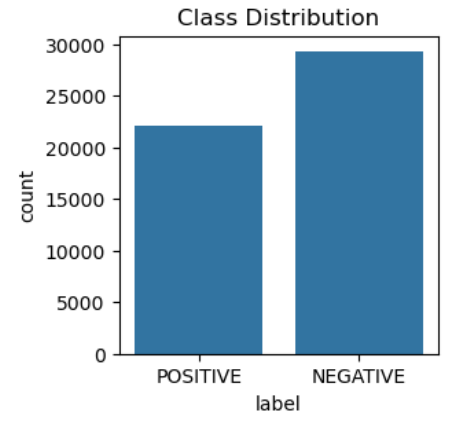
To visualize the most common words used in both positive and negative reviews, we created word clouds. This helped us quickly identify prevalent sentiments and themes.

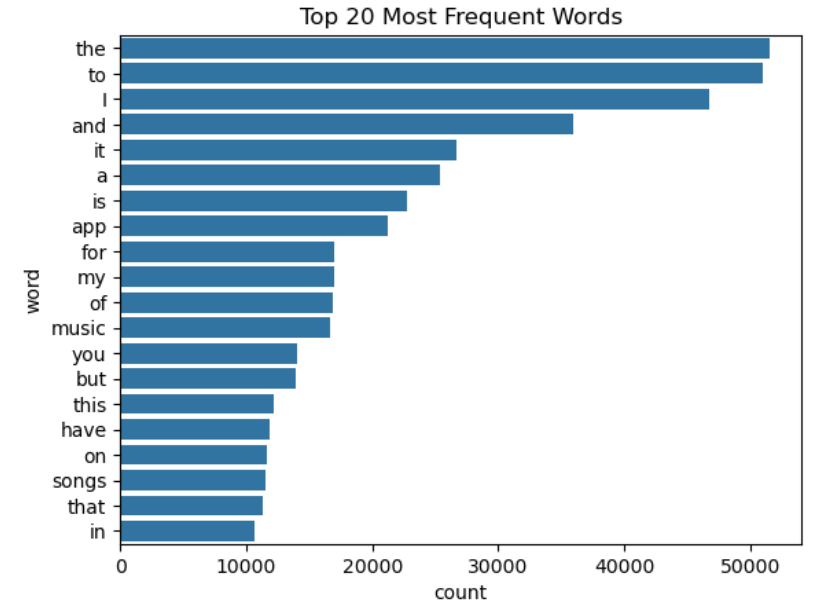
Words like **Song, app, spotify, music, love, listen, playlist** are most frequent words in both negative and positive class.

In Negative class words like ad, issue, premium, fix clearly representing problems faced by users while using application

it leads to bad experience for user.

**4. Class Label Distribution:** We examined the distribution of class labels to understand the balance between positive and negative reviews, which is crucial for model training.



**5. Top 20 Frequent Words:** By visualizing the top 20 most frequent words in user reviews, we gained insights into the words commonly used, highlighting their sentiments towards the app. 

**Data Preprocessing :**

Data preprocessing is a crucial step in preparing our dataset for sentiment analysis. In this project, we utilized the **Natural Language Toolkit** (NLTK) to clean and refine the user reviews effectively. The preprocessing steps included in our analysis:

**1. Tokenization:** We split the text into individual words or tokens, allowing us to analyze the frequency and significance of each word in the context of sentiment.

**2. Stopword Removal:**

* To enhance the quality of our analysis, we removed common stopwords that do not contribute significant meaning to the sentiment, such as "is," "the," and "and".
* In addition to the standard stopwords, we included a custom list of frequently occurring words specific to our dataset, such as "app, music, sound, play, spotify, song, listen, and playlist." This helps in eliminating noise and focusing on more meaningful terms that reflect user sentiments.

**3. Lemmatization:** We employed lemmatization to reduce words to their base or root forms. This step ensures that different forms of a word (e.g., "playing," "played," "plays") are treated as a single entity, improving the model’s understanding of user sentiments.

**4. Special Character Removal:** We cleaned the text by removing special characters, which could confuse the sentiment analysis model. This ensures that our analysis focuses solely on meaningful words.

**5. Whitespace Removal:** Unwanted white spaces were eliminated to maintain a clean dataset, further enhancing the quality of the text data used for analysis.

**Data Preparation :**

In the data preparation phase, aimed to create a well-structured dataset suitable for training our sentiment analysis models. We split our dataset into two parts: 80% for training and 20% for testing, ensuring a robust evaluation of our models.

**1.Bag of Words and TF-IDF:**

* Implemented both unigrams and bigrams in our **BoW** and **TF-IDF** models, resulting in a total of **243,786** features. This high-dimensional feature space captures a wealth of information but may also lead to challenges like overfitting.
* The use of unigrams allowed us to consider individual words, while bigrams enabled us to capture common word pairs, providing richer context and enhancing our model's ability to understand user sentiments.

**2. Word2Vec:**

* For the Word2Vec embeddings, we utilized the **Gensim** library, which provided a pre-trained model trained by Google. Each token in our corpus was transformed into an embedding, allowing us to capture semantic meanings and relationships between words.
* By taking the **average** of these embeddings across sentences, we created a more compact representation of the text data that retains meaningful information while reducing dimensionality.

**Model Building :**

In the model building phase, evaluated several classification algorithms to identify the best-performing model for sentiment analysis based on the features extracted from our dataset. We utilized Bag of Words (BoW), TF-IDF, and Word2Vec for feature representation and assessed the following classifiers: Logistic Regression, Multinomial Naive Bayes (NB), and Random Forest Classifier.

**Approach 1:**

**1.Bag of Words (BoW) Results:**

* **Logistic Regression:** 
  + Accuracy: 88%
  + Prediction Time: 0.0000 seconds
* **Multinomial Naive Bayes:** 
  + Accuracy: 86.25%
  + Prediction Time: 0.0160 seconds
* **Random Forest Classifier:** 
  + Accuracy: 85.51%
  + Prediction Time: 0.5474 seconds

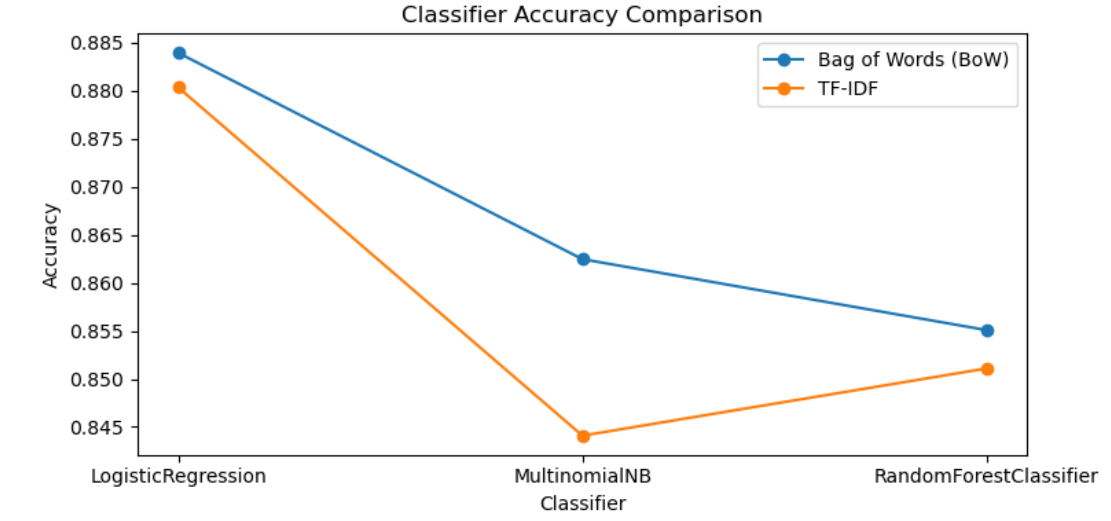
**2.TF-IDF Results:**

* **Logistic Regression:** 
  + Accuracy: 87%
  + Prediction Time: 0.000 seconds
* **Multinomial Naive Bayes:** 
  + Accuracy: 84.41%
  + Prediction Time: 0.0104 seconds
* **Random Forest Classifier:** 
  + Accuracy: 85.11%
  + Prediction Time: 0.5383 seconds

**3.Word2Vec Results:**

* **Logistic Regression:** 
  + Accuracy: 88.39%

The results indicate that Logistic Regression consistently achieved the highest accuracy across both the BoW and Word2Vec feature representations, making it a strong candidate for our final model. The efficiency in prediction time also suggests that Logistic Regression may be preferred in real-time applications, where speed is essential.



**Approach 2:**

**Stacked LSTM Model :**

In addition to traditional machine learning models, I explored advanced deep learning techniques by implementing a **stacked LSTM** model using **Keras**. This approach aimed to leverage the capabilities of LSTM layers for better performance in sentiment analysis.

Key Features of the Stacked LSTM Model:

**1.Tokenization and Preprocessing:** Utilized the Keras tokenizer for text preprocessing, handling out-of-vocabulary (OOV) tokens efficiently. Implemented padding to ensure uniform input length across all samples.

**2.Model Architecture:** The model consists of three stacked LSTM layers, enabling it to capture complex temporal dependencies within the data.

**3.Batch Normalization:** Incorporated batch normalization layers to stabilize learning and improve convergence speed.

**4.Activation Function:** The activation function used was **tanh**, which is well-suited for LSTM networks to manage the gradients effectively.

**5.Optimizer:** Employed the **RMSprop** optimizer, known for its efficiency in training recurrent networks.

**6.Callbacks:** Integrated essential callbacks:

* **EarlyStopping:** To halt training when validation performance starts to degrade, preventing overfitting.
* **ModelCheckpoint:** To save the best model during training for future use.
* **ReduceLROnPlateau:** To adjust the learning rate when a plateau in validation loss is detected, facilitating better convergence.

**7.Model Saving:** Saved the trained LSTM model in Keras format for later use, alongside the tokenizer in pickle format for consistency in text preprocessing during predictions.

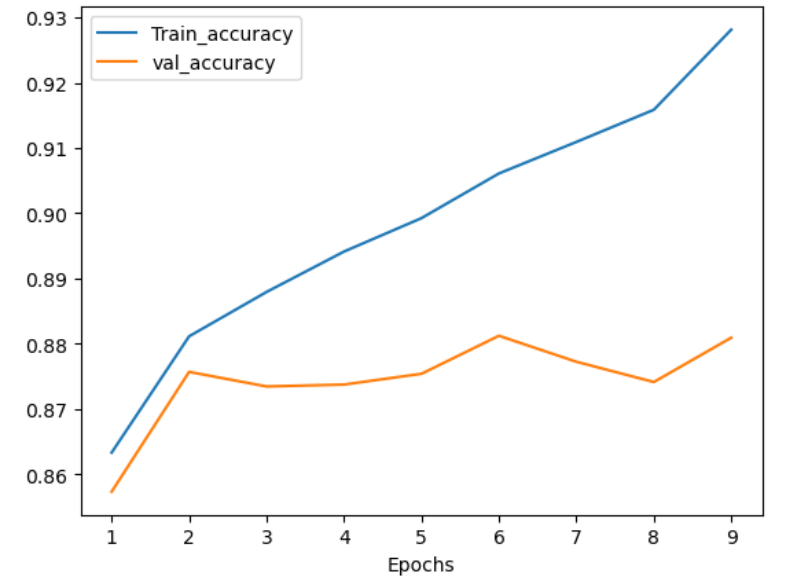
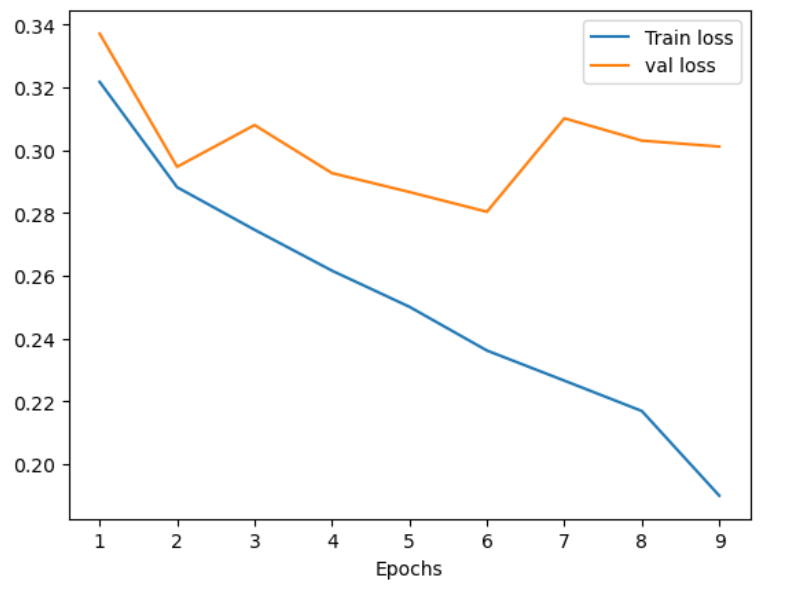
**Performance Visualization:**

The model's performance was monitored throughout the training process, and I generated plots for:

* **Loss and Validation Loss:** To visualize how the model's performance improved over epochs and to monitor for signs of overfitting.
* **Accuracy and Validation Accuracy:** To assess the model's ability to generalize on unseen data.

**Observations on LSTM Model:**

* **Training Accuracy:** The LSTM model achieved an accuracy of **93%** on the training set.
* **Validation Accuracy:** The model's accuracy on the validation set was **88%**.
* **Overfitting:** There is a slight overfitting observed in the model, indicated by the higher training accuracy compared to the validation accuracy.



**Model Evaluation :**

* The models were evaluated based on accuracy, with Logistic Regression achieving 88.39% accuracy using Bag of Words, while the LSTM model reached a training accuracy of 93% and validation accuracy of 88%.

* A classification report was generated, highlighting precision, recall, and F1-score, providing a comprehensive view of model performance across different classes.
* Overall, these evaluations ensured a thorough understanding of each model's strengths and weaknesses in sentiment analysis.

**Conclusion**

The analysis of over **52,702** user reviews reveals key areas for improvement within the **Spotify app**. Here are concise recommendations:

**1. Address Technical Issues:** Focus on resolving recurring bugs and optimizing app performance to enhance user experience.

**2. Enhance User Features**: Develop features that users have requested, such as better **playlist management** and music discovery options, while also **improving the ad experience**.

**3. Leverage User Feedback:** Establish continuous feedback mechanisms to guide product development and ensure updates align with user needs.

**4. Communicate Effectively:** Keep users informed about updates and fixes based on their feedback to build trust and loyalty.

By acting on these recommendations, Spotify can enhance user satisfaction and strengthen its position in the competitive music streaming market.