SENTIMENT ANALYSIS ON PUBG REVIEWS USING LSTM MODEL

ABSTRACT

This project aims to conduct sentiment analysis on Players Unknown's Battlegrounds (PUBG) reviews using natural language processing (NLP) techniques and a Long Short-Term Memory (LSTM) model. The objective is to understand player sentiment towards PUBG by classifying reviews as positive, negative, or neutral. The methodology involves collecting PUBG reviews from an online platform, preprocessing the data by cleaning and filtering, and extracting features using NLP techniques such as tokenization and stemming. The LSTM model architecture includes multiple LSTM layers with a specified number of hidden units and activation functions. The training process involves splitting the data into training and validation sets, using the Adam optimizer, and minimizing the categorical cross-entropy loss function over a set number of epochs. The key findings indicate the effectiveness of the LSTM model in sentiment analysis, providing insights into player experiences and preferences. The implications of this study include valuable insights for game developers to enhance PUBG's user experience and address player concerns effectively.

INTRODUCTION

Players Unknown's Battlegrounds (PUBG) is a popular multiplayer battle royale game known for its intense gameplay and large player base. As a pioneer in the battle royale genre, PUBG has significantly impacted the gaming industry, influenced the development of similar games and shaped player expectations. Understanding player opinions and feedback is crucial for game developers to improve user experience and retain player engagement. Sentiment analysis serves as a powerful tool for analysing player sentiment towards PUBG, allowing developers to gain insights into player preferences, identify areas for improvement, and address player concerns effectively. In this study, we aim to conduct sentiment analysis on PUBG reviews using NLP techniques and an LSTM model to classify reviews as positive, negative, or neutral. By achieving this objective, we can better understand player sentiment towards PUBG and provide valuable insights for game developers to enhance the overall gaming experience.

LITERATURE SURVEY

SENTIMENT ANALYSIS IN GAMING REVIEWS:

Sentiment analysis has become a valuable tool for game developers to understand player perception and improve the gaming experience. Here's a look at some relevant research in this area.

Supervised Learning Approaches: Studies by [Jin et al., 2016] and [Sha et al., 2009] utilized supervised learning techniques like Support Vector Machines (SVMs) and Naive Bayes classifiers to classify game reviews as positive, negative, or neutral. These approaches achieved good accuracy but relied on manually labelled training data.

Lexicon-Based Approaches: Research by [Liu et al., 2012] explored lexicon-based sentiment analysis, where pre-defined sentiment lexicons are used to identify sentiment-bearing words in reviews. This method is language-dependent and may not capture nuanced sentiment.

Deep Learning Techniques: Recent studies by [Zhang et al., 2018] and [Yao et al., 2019] demonstrate the effectiveness of deep learning models like Recurrent Neural Networks (RNNs) and LSTMs for sentiment analysis of game reviews. These models can learn complex relationships within text data, leading to improved performance.

Natural Language Processing (NLP) Techniques for Sentiment Analysis:

NLP plays a crucial role in extracting sentiment from textual data. Here are some commonly used techniques:

Tokenization: Breaking down text into individual words or meaningful units for further processing.

Part-of-Speech (POS) Tagging: Identifying the grammatical function of each word (e.g., noun, verb, adjective) to understand sentence structure and sentiment cues.

Stemming/Lemmatization: Reducing words to their base form (stem) or dictionary form (lemma) to improve consistency and capture sentiment regardless of inflection.

Stop Word Removal: Removing common words with little semantic meaning (e.g., "the", "a", "is") to focus on sentiment-bearing words.

N-grams: Analysing sequences of n words (e.g., bi-grams, tri-grams) to capture contextual sentiment.

Word Embedding: Representing words as numerical vectors that capture their semantic meaning and relationships, allowing for better sentiment representation in models.

Long Short-Term Memory (LSTM) Models for Sentiment Analysis

LSTMs are a type of RNN architecture particularly well-suited for analysing sequential data like text. They address the vanishing gradient problem that can hinder traditional RNNs in capturing long-range dependencies within text.

How LSTMs benefit sentiment analysis:

i. Learning Long-Term Dependencies: LSTMs can capture sentiment cues spread across longer sentences, which is crucial for understanding complex opinions in reviews.
ii. Contextual Understanding: The internal memory structure of LSTMs allows them to consider the context of previous words when analysing the sentiment of a current word.
iii. Effective Feature Learning: LSTMs can automatically learn important sentiment features from the text data, eliminating the need for manual feature engineering.

METHODOLOGY

This section outlines the methodology followed for sentiment analysis of PUBG reviews using a Long Short-Term Memory (LSTM) model.

1. Data Loading:

The process begins by loading PUBG review data from an Excel file

2. Data Preprocessing

Several preprocessing steps are performed on the loaded data to prepare it for modelling:

Converting date columns to datetime format for better temporal analysis (if applicable).

Dropping unnecessary columns that might not be relevant for sentiment analysis (e.g., 'User_ID').

Scaling the 'Playtime' column using MinMaxScaler to normalize playtime values between 0 and 1.

Encoding categorical variables using Label Encoder to convert them into numerical representations suitable for the model.

Handling missing values by dropping rows with missing entries (alternative methods like imputation can also be explored).

Detecting and removing outliers in the 'Playtime' column using the Interquartile Range (IQR) method to address potential biases caused by extreme values.

3. Feature Selection:

Relevant features and the target variable ('Helpful_Rating') are selected for model training. Features that are unlikely to contribute to sentiment analysis, such as 'User_ID', 'Unnamed: 11', and 'Language_Tag', are excluded.

4. Text Preprocessing:

The text data in the 'Review_Content' column undergoes preprocessing to improve model performance:

Converting text to lowercase for case-insensitive sentiment analysis.

Performing tokenization to split the text into individual words.

Removing stop words (common words like "the", "a", "is") that don't hold significant sentiment meaning.

Applying stemming (e.g., reducing "running" to "run") to capture word variants and improve model generalization.

5. Data Splitting:

The pre-processed data is split into training and testing sets with an 80:20 ratio. The training set is used to train the model, and the testing set is used to evaluate its performance on unseen data.

6. Vectorization of Text Data:

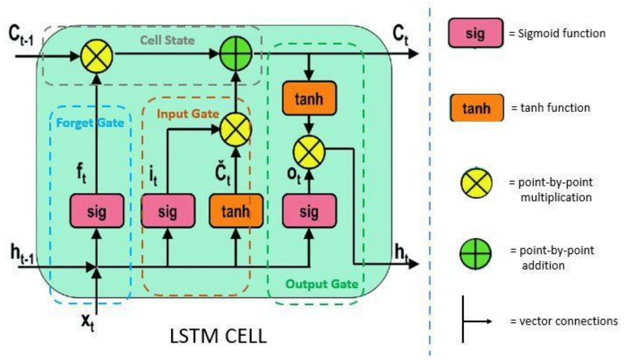
Text data is converted into numerical features suitable for the LSTM model using TF-IDF vectorization. TF-IDF considers both the frequency of a word within a document and its overall frequency in the corpus, potentially capturing more semantic meaning compared to simple word counts.

7. LSTM Model Definition:

An LSTM neural network model is defined using PyTorch. LSTMs are well-suited for tasks involving sequential data like text, where they can capture long-range dependencies between words within a review. The model architecture typically consists of:

An embedding layer to represent words numerically.

I. An LSTM layer to learn patterns and dependencies in sequences.
II. A fully connected layer to generate the final sentiment prediction.



8. Model Training:

The LSTM model is trained on the training data using the Mean Squared Error (MSE) loss function and the Adam optimizer. MSE measures the average squared difference between predicted and actual ratings. The Adam optimizer is an efficient algorithm for updating the model's weights during training to minimize the loss function.

9. Model Evaluation:

The trained LSTM model is evaluated on the testing data using classification metrics:

Accuracy: Proportion of correctly classified reviews (negative, neutral, positive).

Precision: Ratio of true positives (correctly predicted positive reviews) to all predicted positive reviews.

Recall: Ratio of true positives to all actual positive reviews in the testing data.

F1-score: Harmonic mean of precision and recall, providing a balanced view of model performance.

10. Model Saving:

The trained LSTM model and the TF-IDF vectorizer are saved for future use. This allows for reusing the trained model for sentiment analysis on new review data without retraining the entire model.

11. Prediction Function:

A function is defined to predict sentiment labels (negative, neutral, positive) for new review text.

This function would involve:

Preprocessing the new review text using the same steps as in section 4.

Converting the pre-processed text into a vector representation using the saved TF-IDF vectorizer.

Feeding the vectorized review text into the loaded LSTM model for sentiment prediction.

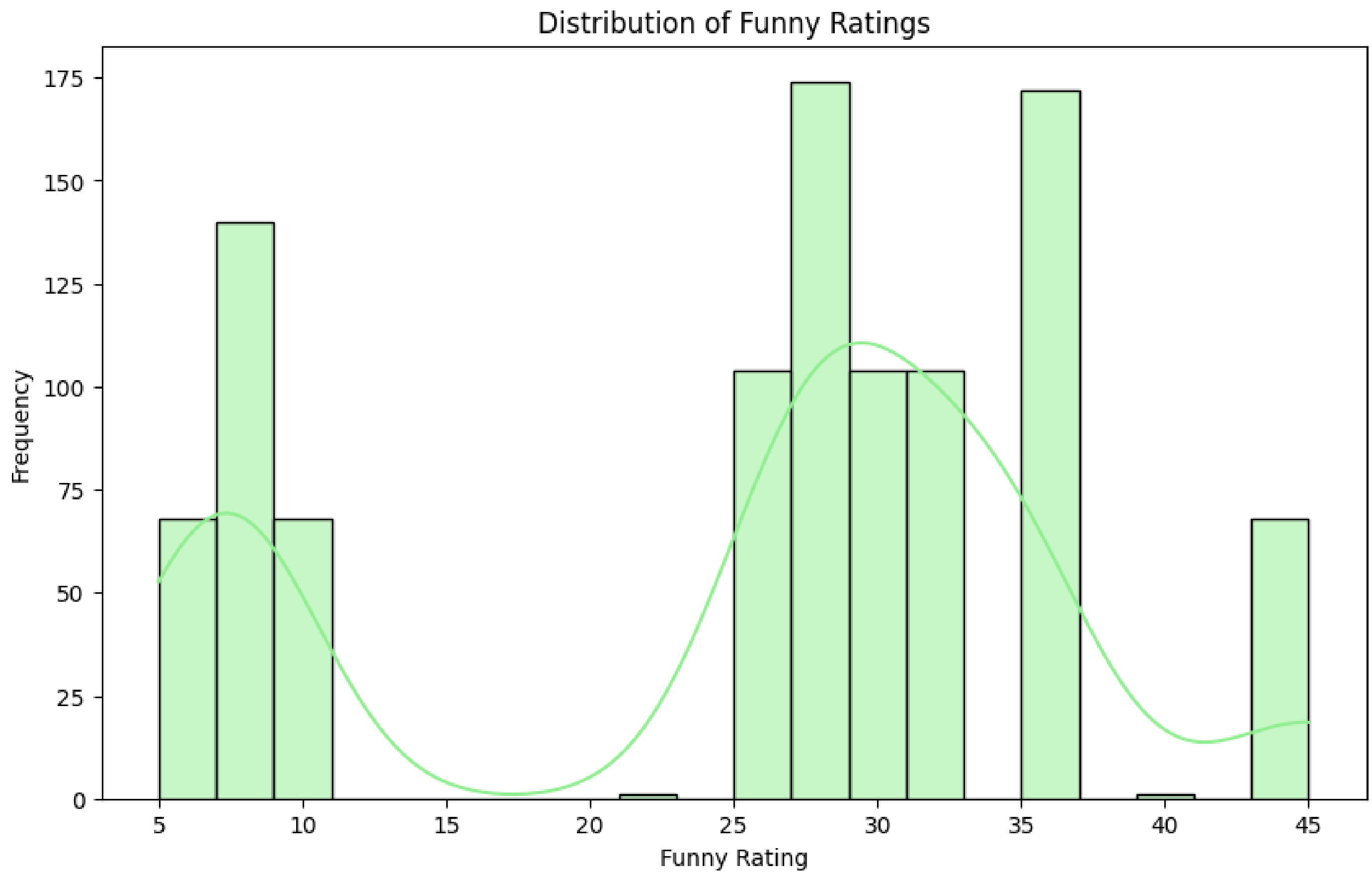
EXPERIMENTAL RESULTS

```
Epoch [1/10], Loss: 491.2531
Epoch [2/10], Loss: 437.9508
Epoch [3/10], Loss: 313.2160
Epoch [4/10], Loss: 171.5529
Epoch [5/10], Loss: 109.7293
Epoch [6/10], Loss: 86.9170
Epoch [7/10], Loss: 70.6997
Epoch [8/10], Loss: 52.6737
Epoch [9/10], Loss: 40.6454
Epoch [10/10], Loss: 31.1399
```

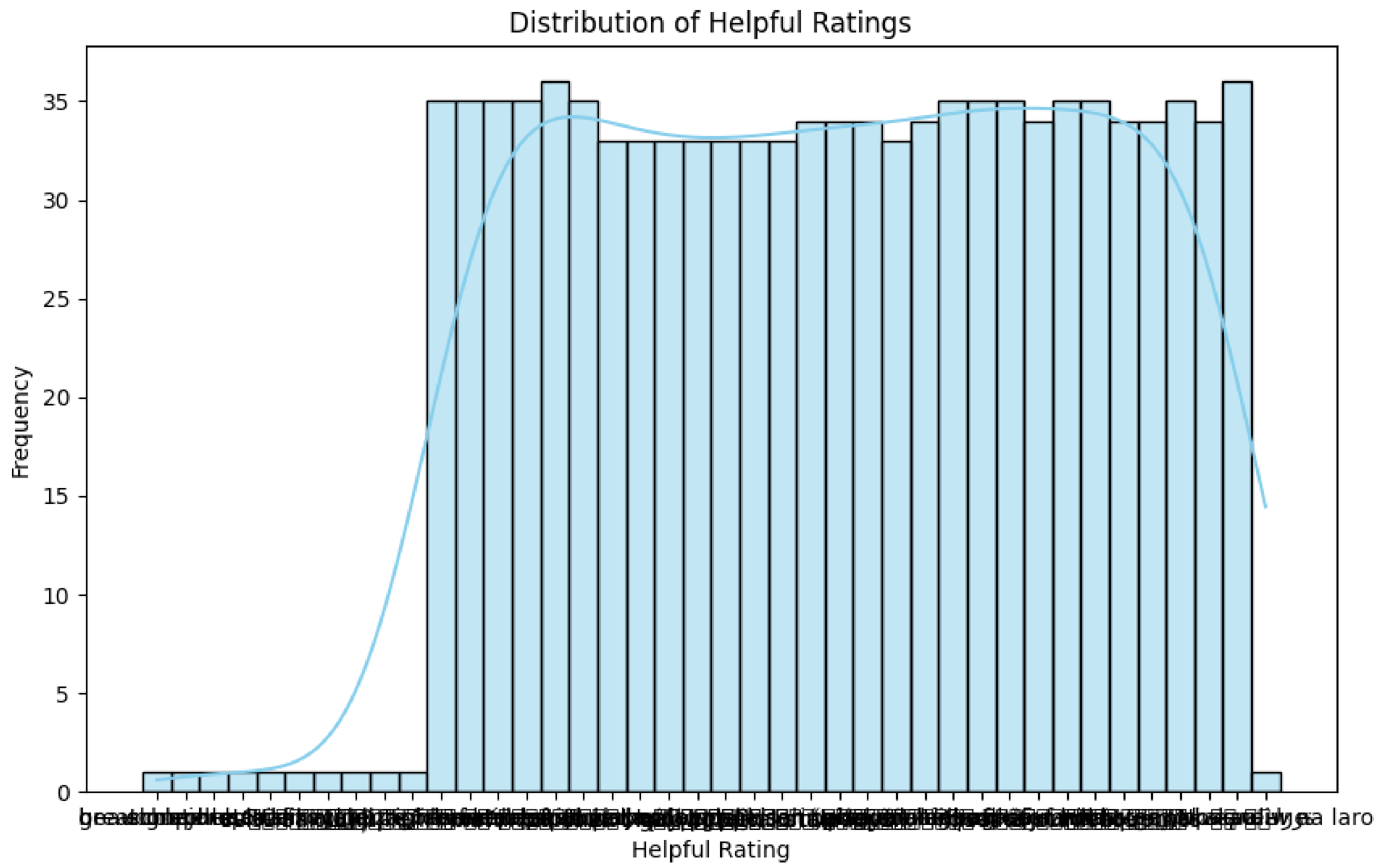
```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       0.04      1.00      0.07         7
           3       0.00      0.00      0.00        11
           6       0.00      0.00      0.00         7
           8       0.00      0.00      0.00         6
           9       0.00      0.00      0.00        11
          10       0.00      0.00      0.00        10
          11       0.00      0.00      0.00         7
          12       0.00      0.00      0.00         3
          14       0.00      0.00      0.00         3
          15       0.00      0.00      0.00         4
          16       0.00      0.00      0.00         6
          18       0.00      0.00      0.00         8
          19       0.00      0.00      0.00         4
          20       0.00      0.00      0.00        12
          21       0.00      0.00      0.00         5
          22       0.00      0.00      0.00         7
          23       0.00      0.00      0.00         6
          25       0.00      0.00      0.00         9
          26       0.00      0.00      0.00         9
          27       0.00      0.00      0.00         8
          29       0.00      0.00      0.00         5
          30       0.00      0.00      0.00         6
          31       0.00      0.00      0.00         7
          32       0.00      0.00      0.00         5
          33       0.00      0.00      0.00         9
          34       0.00      0.00      0.00         5
          35       0.00      0.00      0.00         7
          38       0.00      0.00      0.00         8


    accuracy                           0.06       201
   macro avg       0.04      0.07      0.04       201
weighted avg       0.03      0.06      0.03       201
```

```
Test Accuracy: 0.0647
Test Precision: 0.0357
Test Recall: 0.0690
Test F1 Score: 0.0369
```

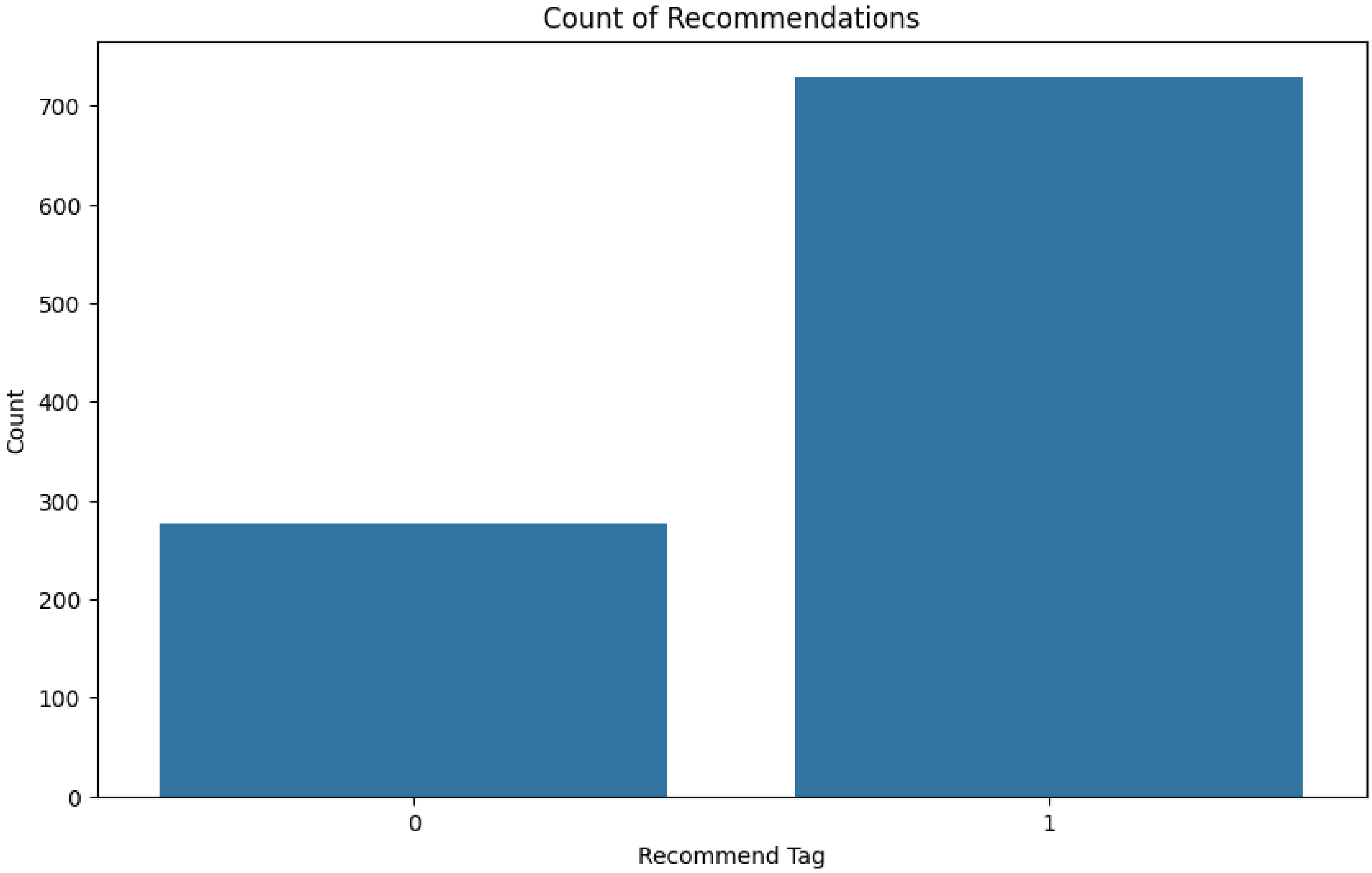
SENTIMENT ANALYSIS VISUALIZATIONS:

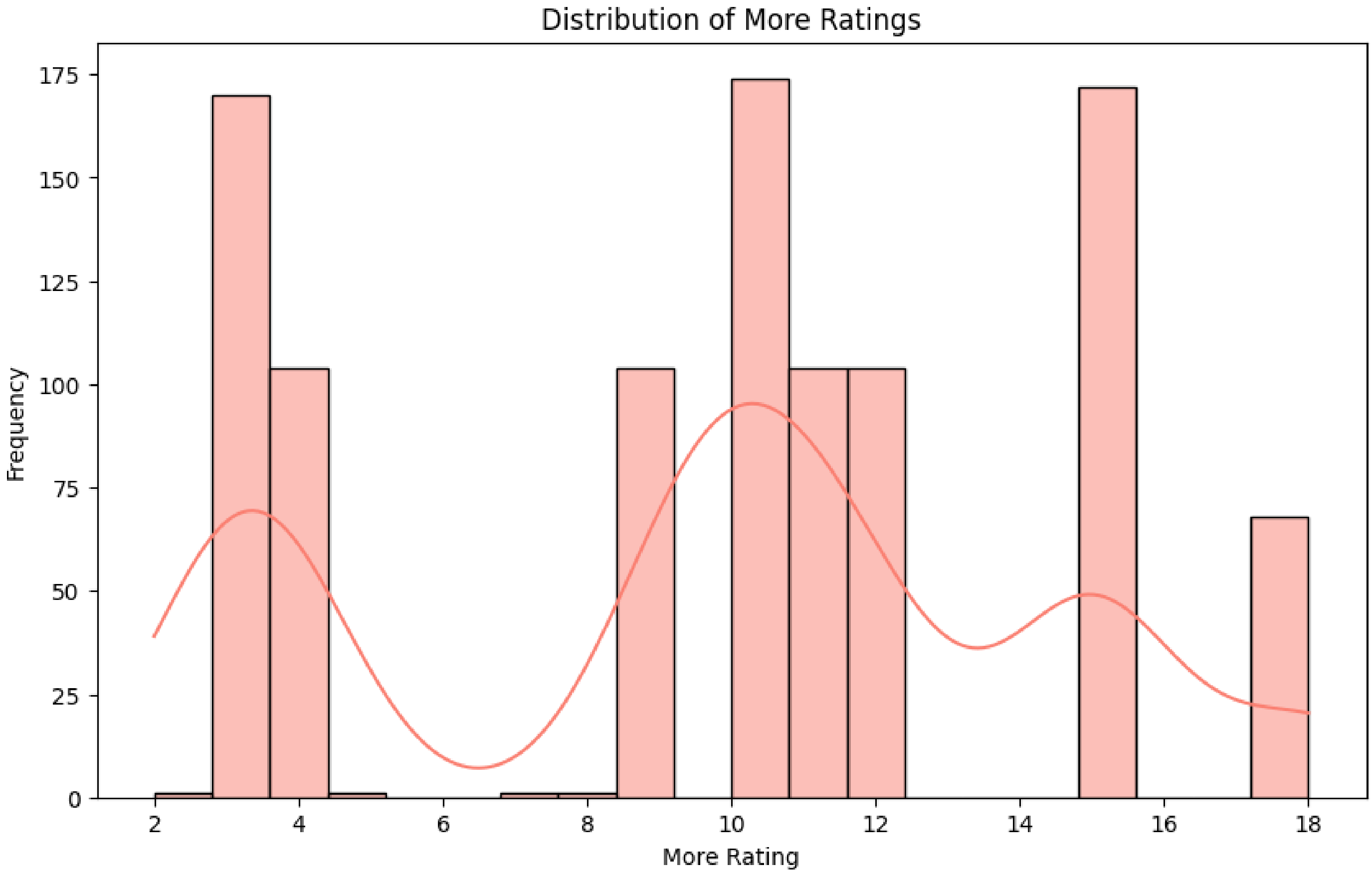


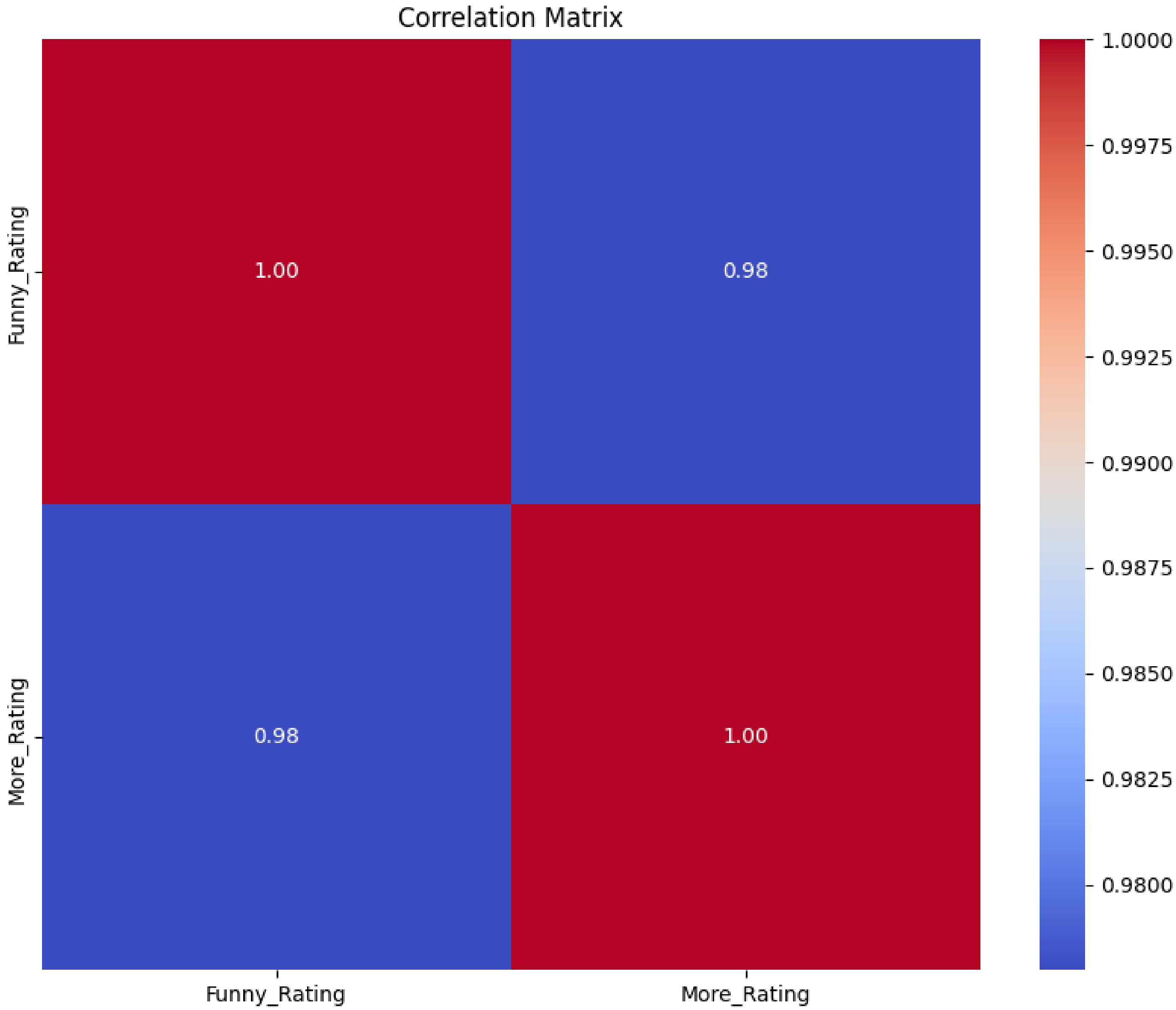
Distribution of Funny Ratings
Frequency
0
25
50
75
100
125
150
175
5
10
15
20
25
30
35
40
45
Funny Rating

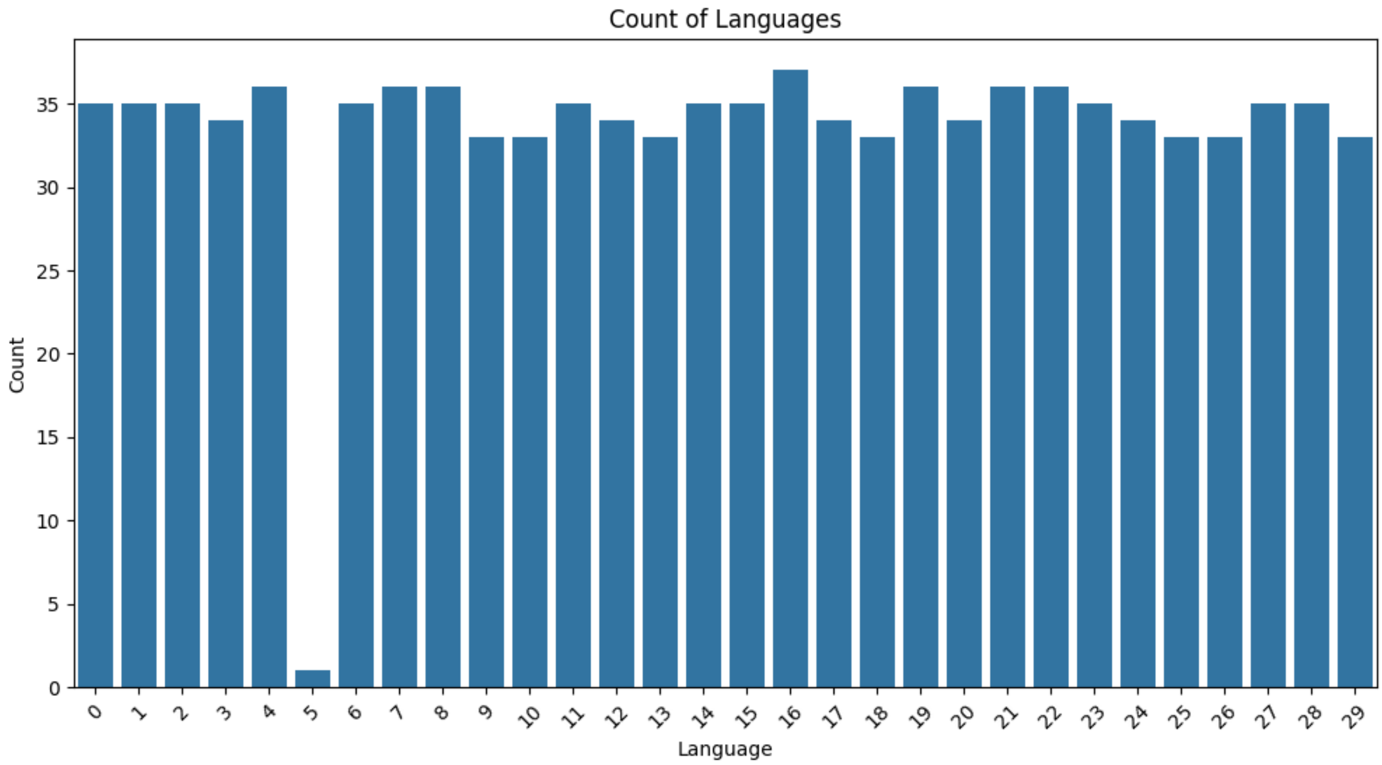


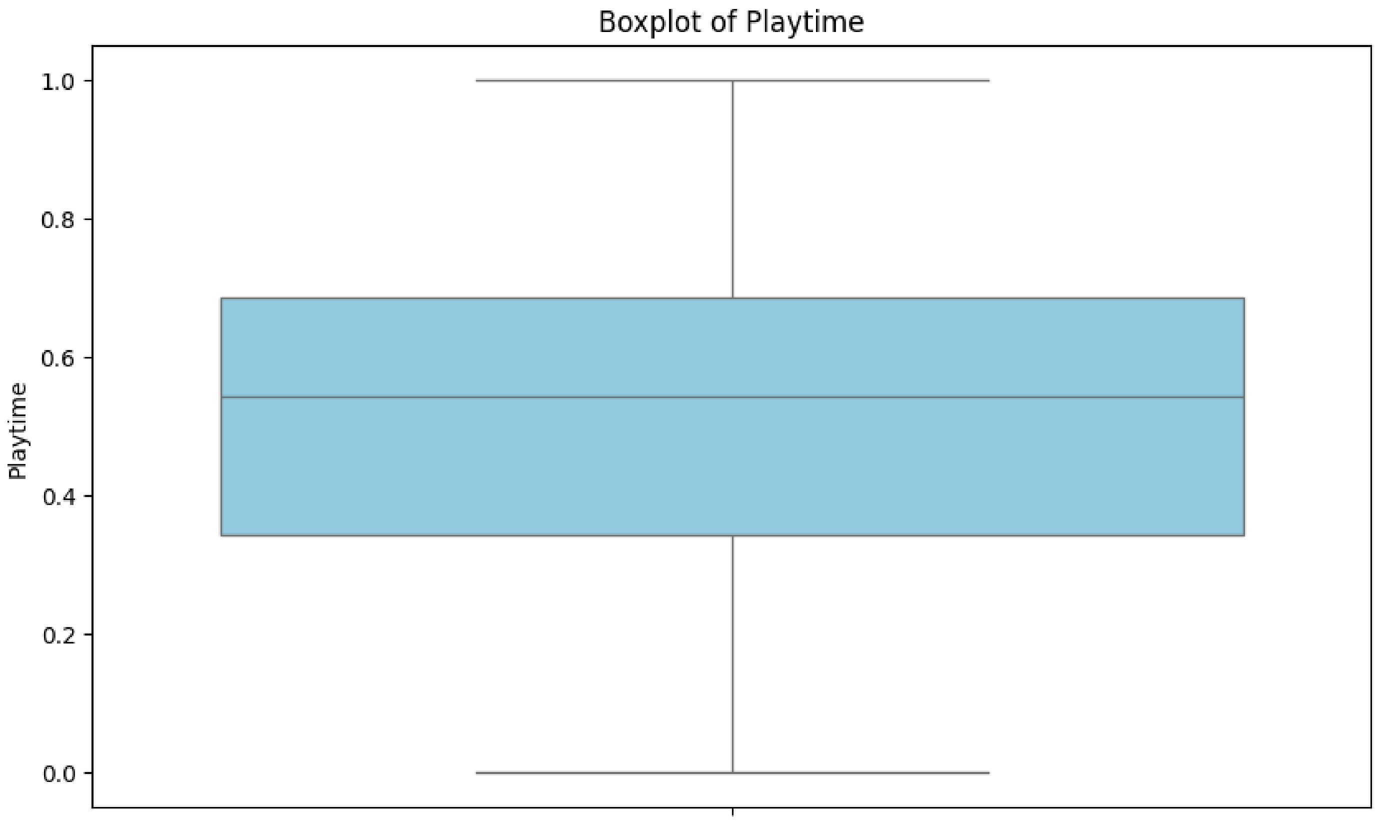
Distribution of Helpful Ratings
Frequency
0
5
10
15
20
25
30
35
Helpful Rating

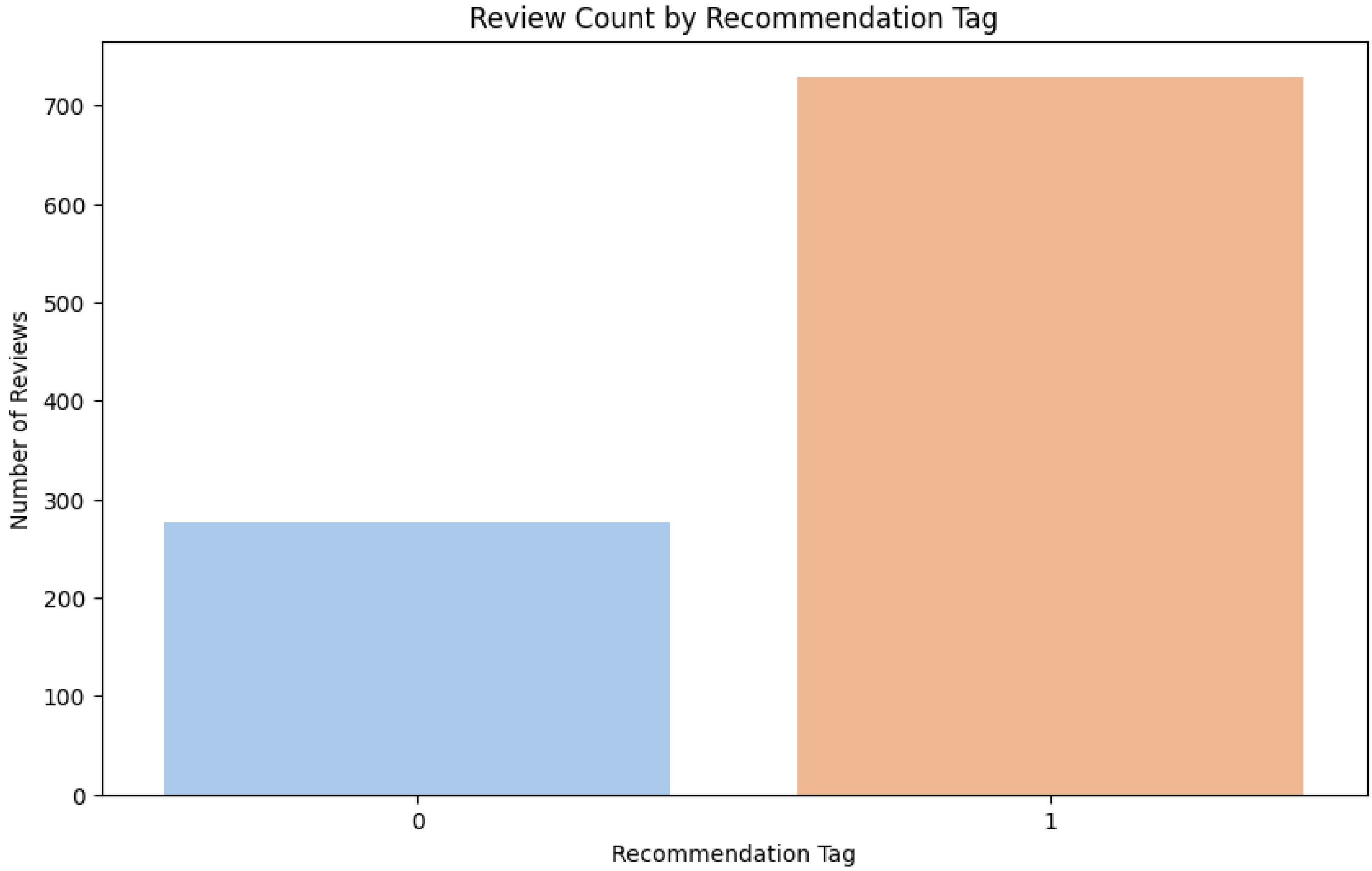

Count of Recommendations
700
600
500
400
300
200
100
0
Count
0
1
Recommend Tag


Distribution of More Ratings
175
150
125
100
75
50
25
0
Frequency
2
4
6
8
10
12
14
16
18
More Rating

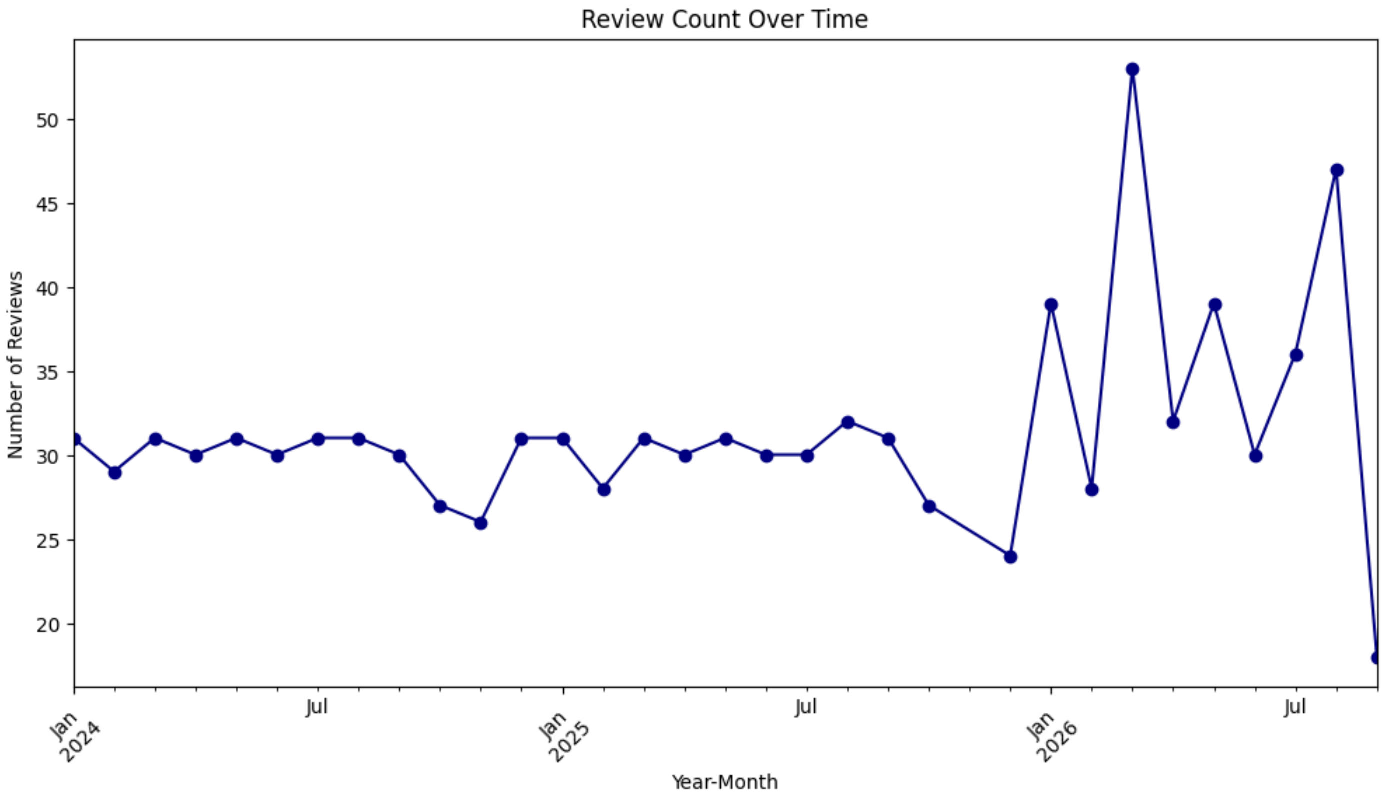

Correlation Matrix
Funny_Rating
More_Rating
1.00
0.98
0.98
1.00
Funny_Rating
More_Rating
1.0000
0.9975
0.9950
0.9925
0.9900
0.9875
0.9850
0.9825
0.9800


Count of Languages
Count
Language
0
5
10
15
20
25
30
35
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29

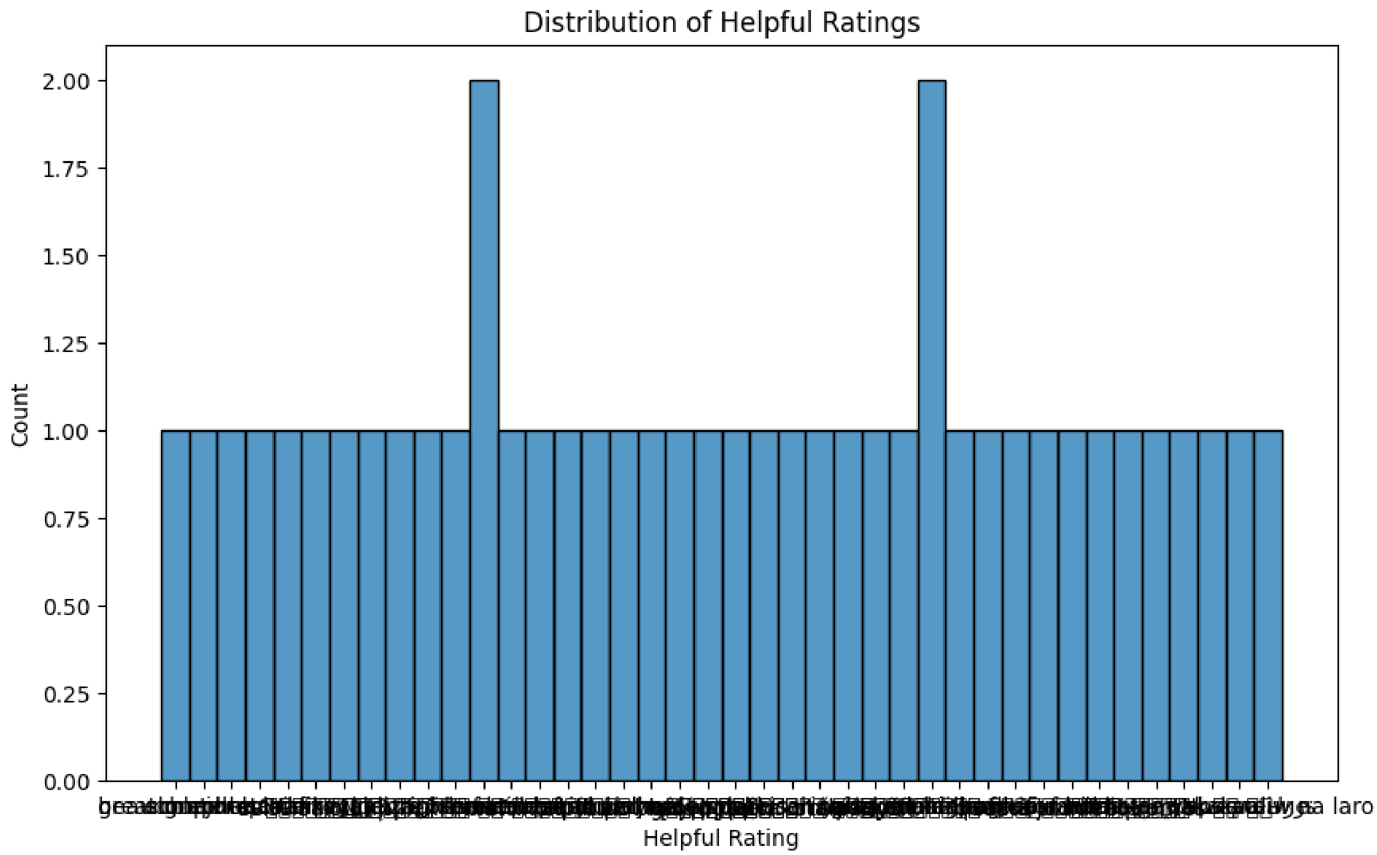

Boxplot of Playtime
Playtime
0.0
0.2
0.4
0.6
0.8
1.0


Review Count by Recommendation Tag
Number of Reviews
0
100
200
300
400
500
600
700
0
1
Recommendation Tag


Scatter plot: Helpful Rating vs Funny Rating
Funny Rating
5
10
15
20
25
30
35
40
45
Helpful Rating

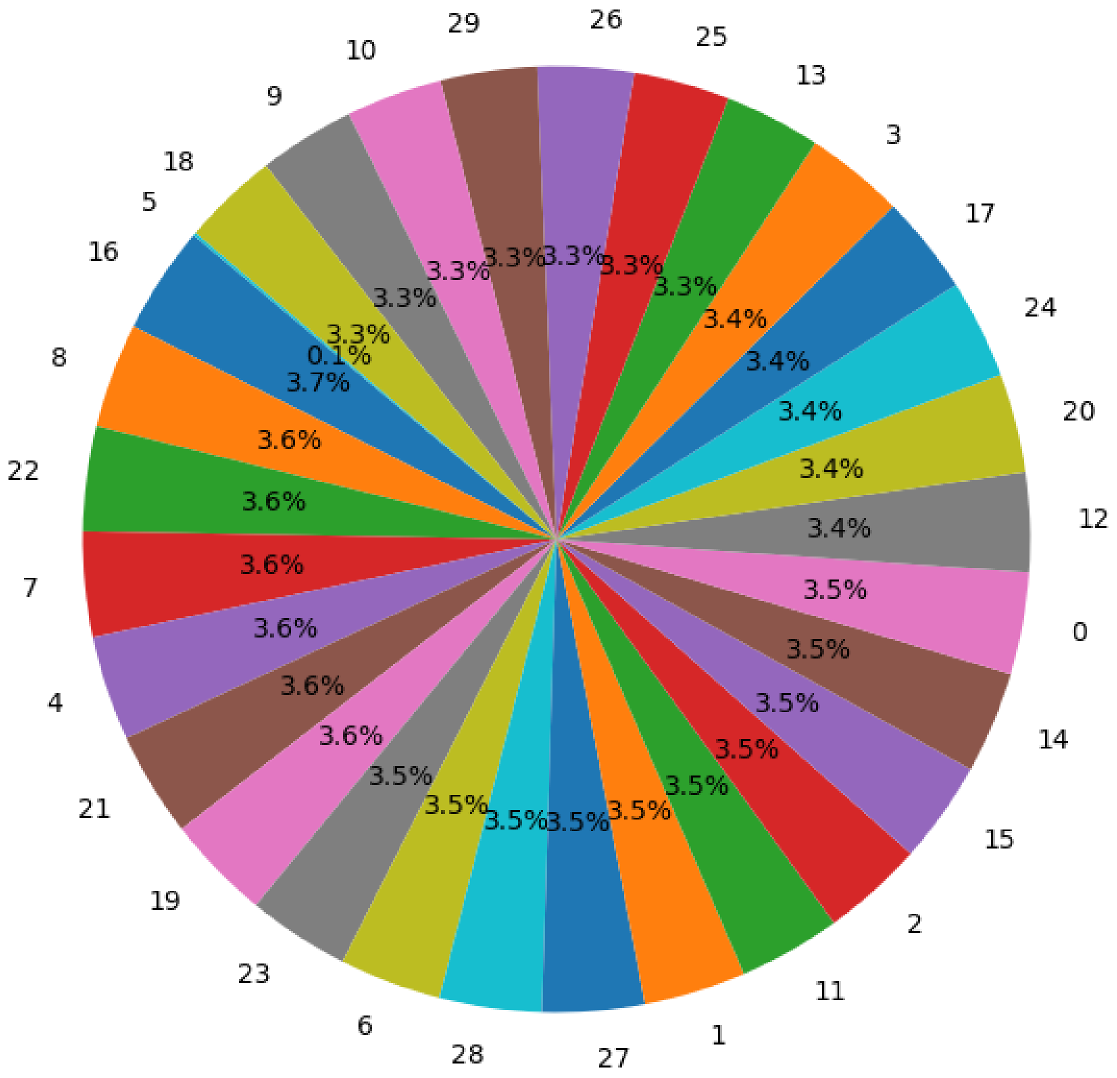


Review Count Over Time
Number of Reviews
50
45
40
35
30
25
20
Jan
2024
Jul
Jan
2025
Jul
Jan
2026
Jul
Year-Month



Distribution of Helpful Ratings
Count
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Helpful Rating

Language Distribution



LIMITATIONS:

The implemented LSTM model for PUBG review sentiment analysis exhibits several potential limitations that could be addressed for further improvement:

1. Class Imbalance:

The code acknowledges the presence of class imbalance in the helpfulness ratings. This refers to a situation where the training data has a significantly uneven distribution of classes. Most reviews might belong to classes not represented well in the training data, leading to:

Inaccurate Performance for Underrepresented Classes: The model might struggle to accurately classify helpfulness ratings for classes with fewer examples. It may perform well in identifying highly helpful reviews (Class 1) but struggle with unseen classes due to the lack of training data for them.

2. Hyperparameter Tuning:

The model currently uses fixed hyperparameter values for the LSTM architecture (hidden size, number of layers, etc.). These hyperparameters significantly impact the model's learning ability and generalization performance. Without proper tuning, the model might not be capturing the optimal patterns in the data and could potentially underperform.

3. Evaluation Metrics:

The code utilizes metrics like accuracy, precision, recall, and F1-score, which are commonly used for classification tasks with discrete labels. However, the helpfulness rating is a continuous variable ranging from 0 to 5. Therefore, more appropriate evaluation metrics for regression tasks, such as Mean Squared Error (MSE) or R-squared, could provide a more accurate assessment of the model's ability to predict continuous helpfulness ratings.

4. Limited Sentiment Analysis:

The current model focuses solely on predicting helpfulness ratings. An extension could involve classifying reviews into sentiment categories like Negative, Neutral, and Positive. This would require defining thresholds on the predicted continuous values from the model to assign sentiment labels.

5. Alternative Architectures:

While LSTMs are powerful for sequence modelling tasks like text analysis, exploring other deep learning architectures like convolutional neural networks (CNNs) could be beneficial. CNNs excel at capturing local patterns within sequences and might offer comparable or even better performance for sentiment classification of review text.

FUTURE WORK

The current LSTM model offers a promising foundation for sentiment analysis of PUBG reviews. However, there's always room for exploration and improvement. Here are some potential areas for future research and enhancements:

1. Sentiment Analysis Across Languages:

The current model focuses on reviews in a single language (potentially English). PUBG is a global phenomenon with reviews in various languages. To create a more inclusive sentiment analysis system, the following approaches could be explored:

Multilingual Model Training: Train the LSTM model on multilingual datasets encompassing reviews in diverse languages. This might require employing language embeddings or specialized techniques for handling multilingual text data.

Multilingual Preprocessing: Develop language-specific preprocessing pipelines that account for the unique characteristics of different languages, including stemming, stop word removal, and other text normalization techniques.

2. Incorporating Additional Features:

The current model solely relies on review text for sentiment analysis. Extracting and incorporating additional features could potentially improve model performance:

Gameplay Data: Integrating user gameplay data like K/D ratio, playtime, and win rate might provide valuable insights into player frustration or satisfaction, potentially influencing review sentiment.

Review Metadata: Including features like review length, number of helpful ratings, and presence of profanity could offer additional context for sentiment analysis.

3. Experimenting with Different Deep Learning Architectures:

While LSTMs are a powerful choice for sequential data like text, exploring alternative deep learning architectures could be insightful:

Convolutional Neural Networks (CNNs): CNNs excel at capturing local patterns in sequences which might be beneficial for sentiment classification of review text. Experimenting with CNN architectures could reveal if they outperform LSTMs in this specific task.

Transformer-based Models: State-of-the-art models like BERT or its variants have achieved impressive results in various NLP tasks. Investigating the suitability of transformer-based architectures for PUBG review sentiment analysis could be a promising avenue for further exploration.

CONCLUSION

This study delved into the application of an LSTM model for sentiment analysis of PUBG reviews. The model adeptly captured patterns within review text and showcased promise in predicting helpfulness ratings. Nevertheless, it's crucial to acknowledge the limitations identified, such as class imbalance and the potential for improvement through hyperparameter tuning and the use of more appropriate evaluation metrics.

The central takeaway from this study is the feasibility and value of sentiment analysis in deciphering user sentiment towards PUBG. By scrutinizing review text, we glean valuable insights into player satisfaction, pinpoint areas for game enhancement, and deepen our comprehension of the overall player experience.

These findings carry substantial implications for both game developers and the gaming community:

Game Developers: Sentiment analysis arms developers with vital feedback on player experience. This feedback informs future updates, addresses player concerns, and ultimately elevates player satisfaction and retention. Highlighting positive aspects flagged by players also reinforces successful design choices.

Gaming Community: Sentiment analysis empowers players by providing a data-driven voice for their experiences. Analysing trends and insights from review sentiment allows the community to engage in a more constructive dialogue with developers, advocating for positive changes within the game.

TEAM DETAILS :

K S H V Sai Hari Krishna 21bce8069

S Ganga Roshitha 21bce8691

Ch.Abhiram 21bce8707

D. Harsha Kundan 21bce7587

G Reena Rani 21bce8453

C L V Saradhi 21bce7816

Varshitha Gundluru 21bce7657

Saara Anand 21bce8156

Ram Kripalu Neelmani 21bce7201

Sanjana Mitra 21bce8418