Sentiment Analysis Model EDA Report

Dataset Statistics

Total samples: (1599999, 6)

Sentiment distribution: {4: 800000, 0: 799999}

Average text length: 74.09 characters

Number of unique users: 659775

Model Performance

Accuracy: 0.3395

Classification Report:

	precision	recall	f1-score	support
0	0.75	0.68	0.71	799999
1	0.00	0.00	0.00	0
4	0.00	0.00	0.00	800000
accuracy			0.34	1599999
macro avg	0.25	0.23	0.24	1599999
weighted avg	0.38	0.34	0.36	1599999

Expert Analysis and Recommendations

1. **Dataset Characteristics Analysis**:

- The dataset consists of 1,599,999 samples with 6 features.

- Sentiment distribution shows a binary classification task with sentiment labels 0 and 4, where 0 represents negative sentiment and 4 represents positive sentiment.

- The average text length is 74.09 characters, indicating relatively short text inputs.

- There are 659,775 unique users in the dataset, suggesting a diverse set of contributors.

- The data was collected between April 17, 2009, and May 27, 2009, providing a specific timeframe for the dataset.

2. **Model Performance Insights**:

- The model's accuracy of 0.3395 indicates poor performance.

- The classification report shows low precision, recall, and F1-scores for all classes. The model fails to predict any samples for class 1, which indicates a severe issue.

- The macro-average F1-score of 0.24 suggests poor overall model performance across classes.

3. **Recommendations for Improving Model Accuracy**:

- **Data Preprocessing**:

- Text preprocessing techniques like tokenization, lowercasing, and removing stopwords can help clean the text data.

- Handling of special characters, emojis, and URLs might improve text normalization.

- **Advanced Text Representations**:

- Using word embeddings like Word2Vec, GloVe, or fastText can capture semantic relationships in the text data better.

- **Model Selection and Tuning**:

- Trying different models like LSTM, GRU, or transformer-based models may improve performance.

- Hyperparameter tuning using techniques like grid search or random search can optimize model performance.

- **Ensemble Methods**:

- Implementing ensemble methods like bagging or boosting can combine multiple models to improve predictive performance.

4. **Potential Areas for Feature Engineering**:

- **N-grams**:

  - Including bi-grams or tri-grams in the text representation can capture more contextual information.

- **Sentiment Lexicons**:

  - Using sentiment lexicons like SentiWordNet or AFINN can provide additional features for sentiment analysis.

- **POS Tags**:

  - Incorporating Part-of-Speech (POS) tags as features can enhance the model's understanding of text structure.

- **Emotion Analysis**:

  - Extracting emotional features from text, such as joy, sadness, anger, etc., can enrich the sentiment analysis process.

5. **Handling Class Imbalance**:

- Since there is a class imbalance with an equal split between negative (0) and positive (4) sentiments, techniques to address this issue include:

- **Resampling Methods**:

  - Using techniques like oversampling (SMOTE) or undersampling can balance the class distribution.

- **Class Weights**:

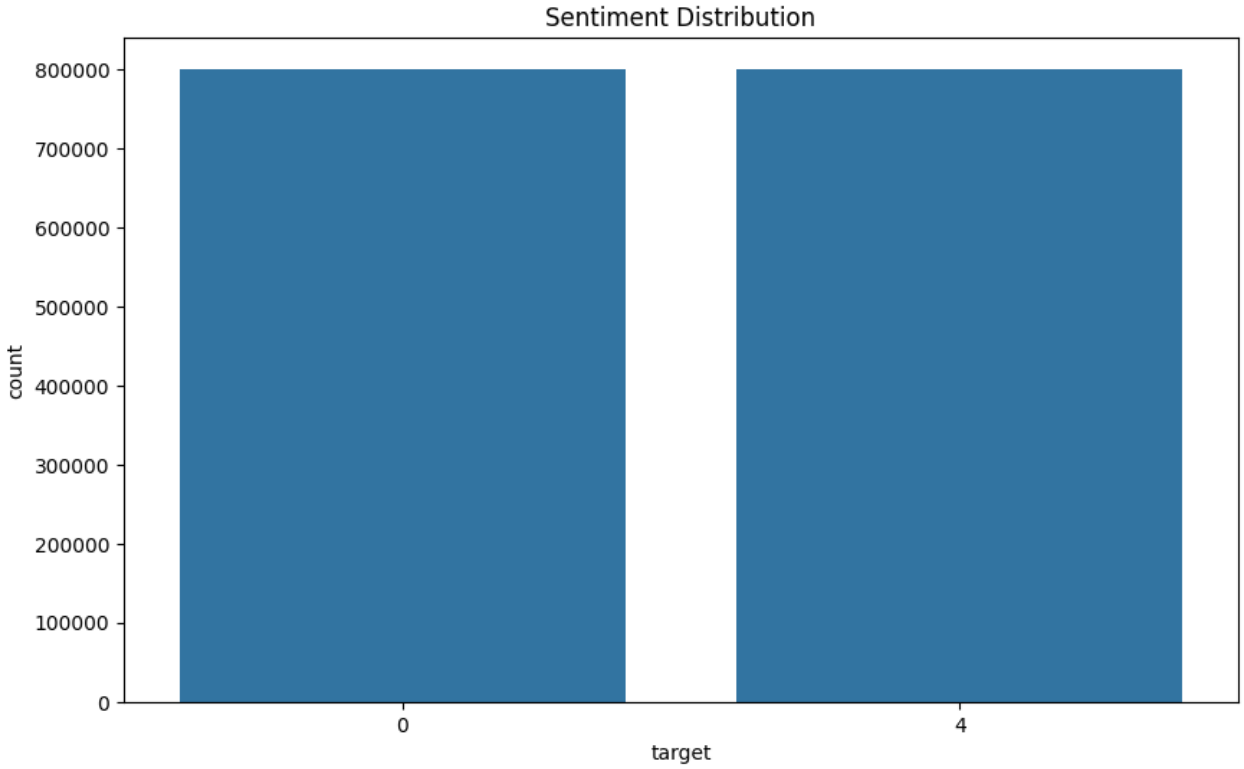
  - Assigning class weights during model training can penalize misclassifications in the minority class.

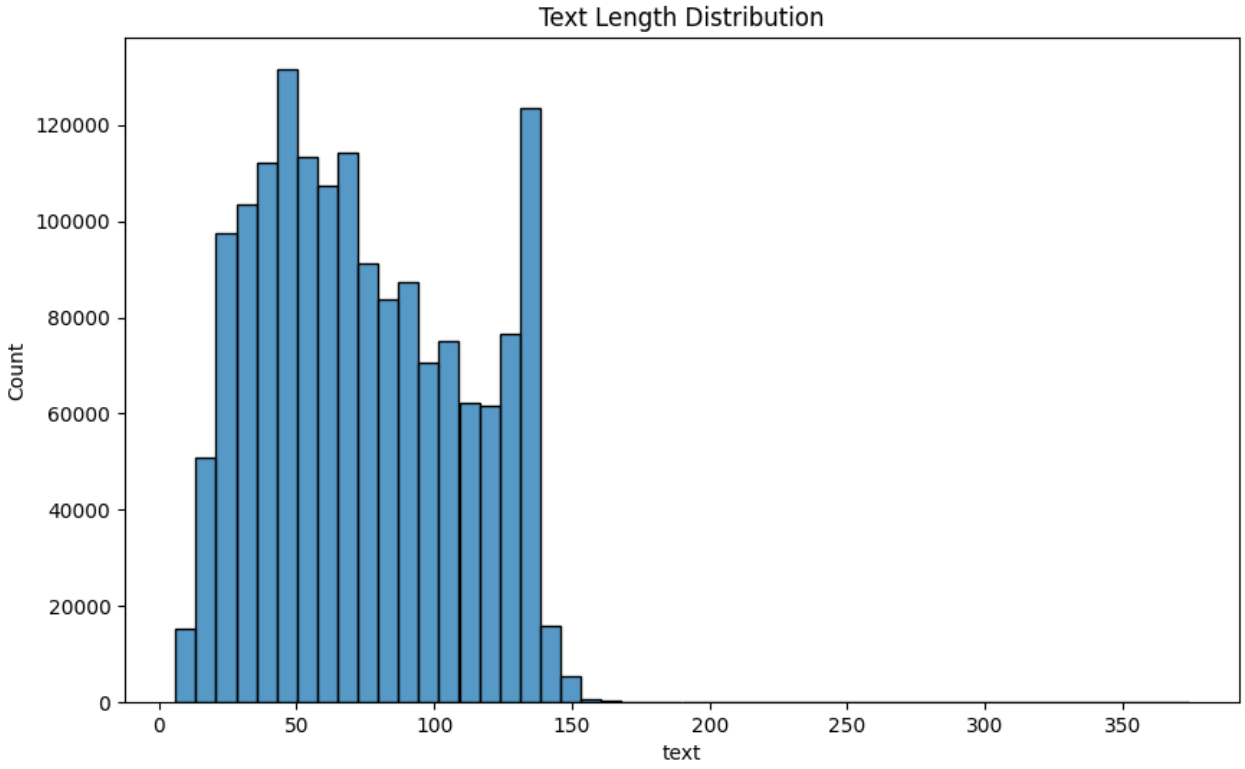
- **Anomaly Detection**:

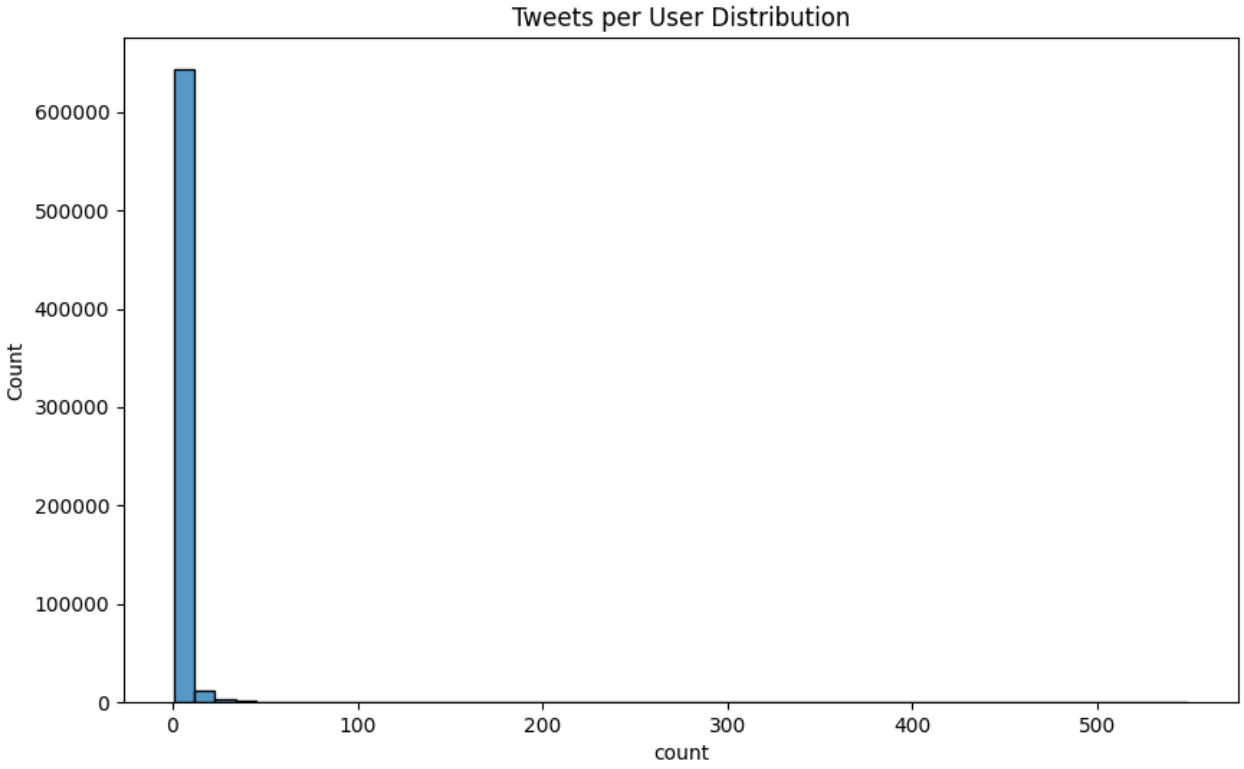
  - Treating the imbalance as an anomaly detection problem and using techniques like One-Class

SVM or Isolation Forest.

Overall, addressing data preprocessing, enhancing text representations, optimizing model selection, utilizing feature engineering, and handling class imbalance can significantly improve the sentiment analysis model's accuracy and performance.


Sentiment Distribution
800000
700000
600000
500000
400000
300000
200000
100000
0
count
0
4
target


Text Length Distribution
Count
text
0
20000
40000
60000
80000
100000
120000
0
50
100
150
200
250
300
350


Tweets per User Distribution
600000
500000
400000
300000
200000
100000
0
Count
0
100
200
300
400
500
count