Twitter Sentiment Analysis Project

Project Overview

This project implements a machine learning pipeline to classify tweets as positive (non-offensive) or negative (racist/sexist). The analysis follows a complete data science workflow from data loading and preprocessing to model training and evaluation.

Data Exploration

The dataset consists of approximately 32,000 tweets labeled as either offensive (1) or non-offensive (0). Each tweet is represented with three columns:

- `id`: Unique identifier
- `label`: Binary classification (0 = non-offensive, 1 = offensive)
- `tweet`: Raw tweet text

Data Preprocessing Pipeline

The preprocessing workflow includes several text cleaning techniques:

1. **Username Removal**: Removed @user mentions using regex pattern matching
   python
   Copy
   ```
   df['clean_tweet'] = df['tweet'].str.replace('@[\w]+', '')
   ```
2. **Punctuation and Special Character Removal**: Stripped non-alphabetic characters
   python
   Copy
   ```
   df['clean_tweet'] =
   df['clean_tweet'].str.replace("[^a-zA-Z\s]", "")
   ```
3. **Tokenization**: Split text into individual words
   python
   Copy
   ```
   tokenized_tweet = df['clean_tweet'].apply(lambda x: x.split())
   ```

Stemming: Applied Porter Stemmer to reduce words to their root form
python
Copy
```
stemmer = PorterStemmer()
```

4. `tokenized_tweet = tokenized_tweet.apply(lambda sentence: [stemmer.stem(word) for word in sentence])`

Exploratory Data Analysis (EDA)

Performed visual analysis to understand word frequency distribution across sentiment classes:

1. **Overall Word Cloud**: Generated visualization of most frequent words across all tweets
2. **Positive Sentiment Word Cloud**: Most common words in non-offensive tweets showing themes of positivity
3. **Negative Sentiment Word Cloud**: Most common words in offensive tweets revealing problematic language

Feature Extraction

Used the Bag-of-Words approach with scikit-learn's CountVectorizer:

```
bow_vectorizer = CountVectorizer(
    max_df=0.90,
    min_df=2,
    max_features=1000,
    stop_words='english'
)
bow = bow_vectorizer.fit_transform(df['clean_tweet'])
```

Hashtag Analysis

Implemented custom function to extract and analyze hashtags from tweets:

```
def hashtag_extract(tweets):
    hashtags = []
    for tweet in tweets:
        ht = re.findall(r"#\w+", tweet)
        hashtags.append(ht)
    return hashtags
```

Model Training and Evaluation

Used Logistic Regression as the classification model:

python

Copy

```
model = LogisticRegression()
model.fit(X_train, y_train)
```

Model Performance

Model evaluation using standard classification metrics:

Default Threshold (0.5):

- F1 Score: 0.4911
- Accuracy: 0.9466

Custom Threshold (0.3):

- F1 Score: 0.5656
- Accuracy: 0.9441

Key Findings

1. The model achieves high accuracy (94%) but moderate F1 score due to class imbalance
2. Adjusting the probability threshold from 0.5 to 0.3 improved F1 score
3. Word clouds reveal distinct linguistic patterns between offensive and non-offensive content
4. Hashtag analysis shows topic clustering by sentiment category

Based on the Twitter Sentiment Analysis Project information, the class imbalance appears to be a significant disparity between the number of non-offensive (0) and offensive (1) tweets in the dataset.

The key indicators of this imbalance are:

1. The high accuracy (94.66%) but moderate F1 score (0.4911) with the default threshold - this pattern typically occurs when a model performs well on the majority class but struggles with the minority class
2. The improvement in F1 score (from 0.4911 to 0.5656) when adjusting the classification threshold from 0.5 to 0.3 - lowering the threshold is a common technique to address class imbalance by making the model more sensitive to the minority class
3. The need for specialized evaluation metrics beyond accuracy - the project specifically uses F1 score, which is more informative than accuracy when classes are imbalanced

This suggests that non-offensive tweets (label 0) significantly outnumber offensive tweets (label 1) in the dataset. This imbalance is common in sentiment analysis tasks, particularly when

dealing with toxic content detection, as most social media content tends to be non-offensive under normal circumstances.

The imbalance explains why the model achieves high accuracy (by mostly predicting the majority class) but struggles with the F1 score (which balances precision and recall, especially important for the minority class).