Text Analysis of Airline Twitter

1 Introduction

This report presents the analysis and classification of airline-related tweets into three sentiment categories: negative, neutral, and positive. The objective was to preprocess the raw text data and build models to classify the sentiment of these tweets. Multiple machine learning models were evaluated, including Logistic Regression, Naive Bayes (Complement and Multinomial), and the final performance metrics of each model were compared. The steps, analysis, and insights are detailed below.

2 Text Preprocessing

Preprocessing text data is essential for transforming raw tweets into a structured form that machine learning models can process. The following steps were applied to the text data:

- **Removing URLs**: All URLs within the tweets were removed to reduce noise.
- **Lowercasing**: Text was converted to lowercase to maintain uniformity.
- **Tokenization**: The tweets were split into individual words using the NLTK `word_tokenize()` function.
- **Removing Non-alphabetic Characters**: Special characters and numbers were removed from the text.
- **Stopword Removal**: Stopwords (common words like 'the', 'and', etc.) were removed to keep only meaningful words.
- **Stemming**: The Porter Stemmer was applied to reduce words to their base forms (e.g., 'running' to 'run').

Text Preprocessing Code Snippet:

```
def preprocess_text(text):
    text = re.sub(r'http\S+|www\S+|https\S+', '', text, flags=re.MULTILINE)
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    text = text.lower()
    tokens = word_tokenize(text)
    tokens = [stemmer.stem(word) for word in tokens if word not in stop_words]
    return ' '.join(tokens)
```

After preprocessing, frequent words and irrelevant stopwords were removed. This cleaned the dataset, preparing it for vectorization.

3 Exploratory Data Analysis (EDA)

Several visualizations were created to explore the distribution and characteristics of the text data.

- **Tweet Length Distribution by Sentiment**: The distribution of tweet lengths across different sentiment categories was plotted. It was observed that negative tweets tended to be longer, while positive and neutral tweets were relatively shorter.

- **Word Clouds**: Word clouds were generated to visually highlight the most frequent words used in positive, neutral, and negative tweets. This helped identify key words that may influence sentiment detection.

Figure 1: Word Cloud for Positive Tweets

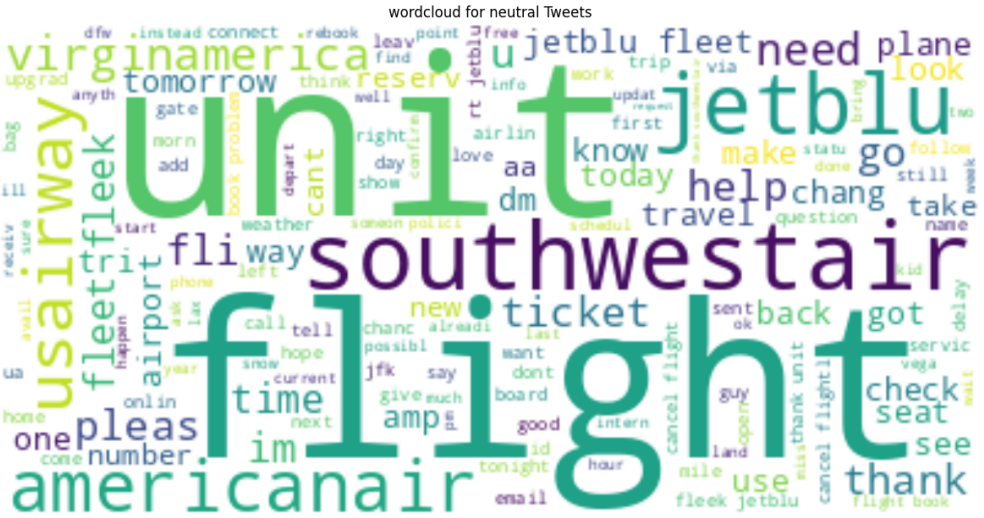

Figure 2: Word Cloud for Negative Tweets

4 Model Training

Three different models were trained using the preprocessed data:

- **Multinomial Naive Bayes**: A probabilistic model ideal for text classification. The model was trained on the TF-IDF vectorized data.
- **Complement Naive Bayes**: This variant of Naive Bayes is particularly suited for imbalanced datasets and often performs better on text classification tasks with class imbalance.
- **Logistic Regression**: A linear model that can classify text into multiple categories.

The dataset was split into training and testing sets using an 80-20 split. Text data was transformed into numerical form using TF-IDF (Term Frequency-Inverse Document Frequency).

TF-IDF Vectorization Code:

```
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(data['text'])
y = data['sentiment']
```

5 Model Evaluation

The models were evaluated based on the following metrics:

- **Accuracy**: The percentage of correct predictions.
- **Precision and Recall**: These metrics evaluate how well the model identifies positive and negative instances.
- **F1-Score**: The harmonic mean of precision and recall, providing a balanced performance measure.
- **Confusion Matrix**: A matrix that shows the actual vs predicted sentiment classifications for the test data.

Performance Summary:

Model	Accuracy	Precision (Weighted Avg)	Recall (Weighted Avg)	F1-Score (Weighted Avg)
Logistic Regression	76.43%	0.76	0.76	0.76
Multinomial Naive Bayes	75.24%	0.74	0.75	0.73
Complement Naive Bayes	74.79%	0.73	0.75	0.73

Table 1: Model Performance Summary

Confusion Matrix (Logistic Regression):

```
Confusion Matrix:
      0    1    2
0  1766   77   46
1   312  224   44
2   204   42  213
```

6 Insights and Conclusion

- **Model Performance**: Logistic Regression performed slightly better than Naive Bayes models, achieving an accuracy of 76.43%. The Complement Naive Bayes model was close behind, indicating that simpler models like Naive Bayes can be effective for text classification.
- **Challenges**: Classifying neutral tweets was particularly challenging, as observed from the lower recall scores for neutral sentiments.
- **Next Steps**: The model could be improved by experimenting with deep learning models such as LSTMs or BERT-based transformers, which can capture more complex relationships within the text.