Homework 4

Mudit Arora

December 7, 2024

Abstract

This report addresses sentiment analysis using the Twitter US Airline Sentiment dataset. We conduct comprehensive data analysis, text processing, and tokenization and employ Support Vector Machines (SVM) for sentiment analysis. The study includes detailed data cleaning methods, custom tokenization, and model performance evaluation through ablation studies.

1 Introduction

Sentiment analysis is essential for understanding customer feedback and opinions, particularly in the service industry. This study uses the Twitter US Airline Sentiment dataset to analyze passenger sentiments about major US airlines. We aim to classify tweets into three categories: positive, negative, or neutral.

Our approach involves thorough data exploration, developing custom text processing techniques, and implementing various preprocessing methods to prepare the dataset for machine learning. We developed a custom tokenizer and employed multiple cleaning strategies to optimize our sentiment classification model.

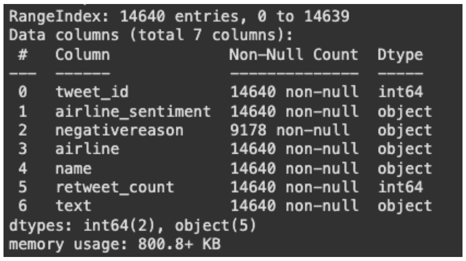
2 Dataset Overview

The dataset contains 14,640 tweets with various features related to airline sentiment analysis. Key findings from our initial analysis include:

2.1 Overall Statistics

- Total samples: 14,640
- Sentiment distribution: predominantly negative (9,178 tweets)
- Most common negative reason: Customer Service Issue (2,910 instances)

This is how the dataset looks like after using only the relevant columns from the original dataset:



```
RangeIndex: 14640 entries, 0 to 14639
Data columns (total 7 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   tweet_id            14640 non-null  int64
 1   airline_sentiment   14640 non-null  object
 2   negativereason      9178 non-null   object
 3   airline             14640 non-null  object
 4   name                14640 non-null  object
 5   retweet_count       14640 non-null  int64
 6   text                14640 non-null  object
dtypes: int64(2), object(5)
memory usage: 800.8+ KB
```

Figure 1: Enter Caption

3 Part 1

3.1 Part A

- Total count of data samples: 14640
- Number of unique values of airline sentiment: 3; neutral, positive, negative
- Number of unique values of negative reason: 'Bad Flight', "Can't Tell", 'Late Flight', 'Customer Service Issue', 'Flight Booking Problems', 'Lost Luggage', 'Flight Attendant Complaints', 'Cancelled Flight', 'Damaged Luggage', 'longlines'
- Most frequent value and it's frequency of airline sentiment: negative, 9178
- Most frequent value and it's frequency of negative reason: Customer Service Issue, 2910
- Shortest length of tweet in dataset: 12
- Longest length of tweet in dataset: 186
- Tweet Length Histogram:

3.2 Part B

Tweet sentiment distribution per airline:

3.3 Part C

The custom tokenizer was implemented using regular expressions and designed specifically for processing Twitter data. The implementation follows a rule-based approach with five distinct preprocessing steps.

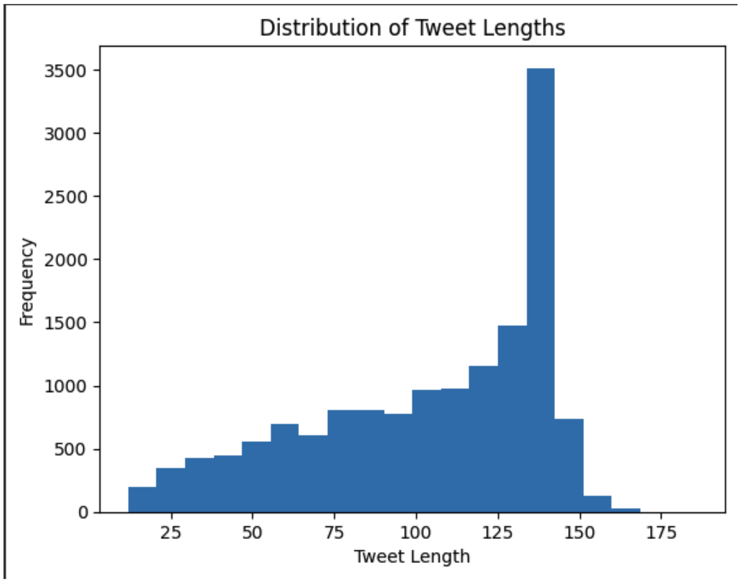


Figure 2: Tweet Length Histogram

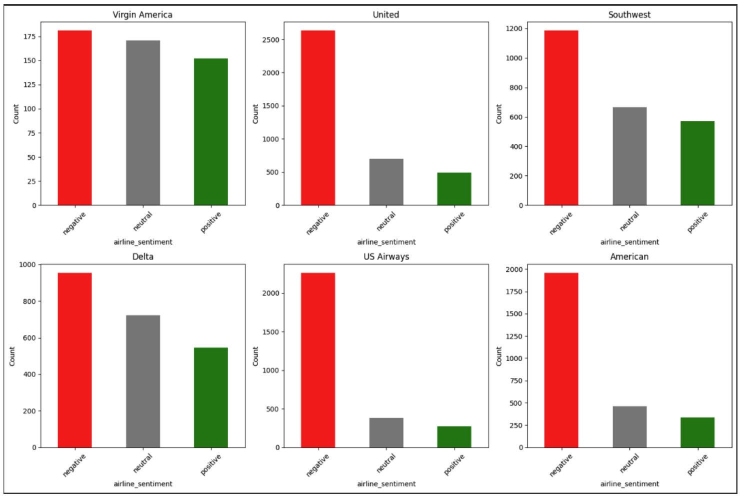


Figure 3: sentiment distribution per airline

3.3.1 Tokenization Rules

The tokenizer implements the following rules in sequence:

1. **Whitespace and Punctuation Handling**
   - Regular Expression: `r"\s+"`
   - Purpose: Normalizes all whitespace sequences to a single space
   - Example: ”tweet text” → ”tweet text”
2. **Hashtag Processing**
   - Regular Expression: `r"#\w+"`
   - Purpose: Identifies and separates hashtags from content

3.3.2 Implementation Details

The tokenizer processes text through the following steps:

1. Each rule is applied sequentially using Python’s `re.sub()` function
2. The text is converted to lowercase for standardization
3. The processed text is split into individual tokens

3.3.3 Example Usage

For the input text:

```
"This is a sample tweet with #hashtags, @mentions, and http://example.com "
```

The tokenizer produces:

```
['this', 'is', 'a', 'sample', 'tweet', 'with', 'hashtags', 'mentions', 'and']
```

3.3.4 Design Considerations

Several key design decisions were made in implementing the tokenizer:

- **Rule Order**: Rules are applied in a specific order to ensure proper handling of nested patterns
- **Whitespace Handling**: Multiple whitespace characters are normalized before tokenization
- **Case Normalization**: All text is converted to lowercase to ensure consistency
- **Special Character Handling**: Social media-specific elements (hashtags, mentions, emojis) are handled explicitly

3.3.5 Advantages

The implemented tokenizer offers several benefits for social media text processing:

- Efficient handling of Twitter-specific elements
- Consistent treatment of special characters
- Simple and maintainable rule-based architecture
- Easy extensibility for additional rules

Mention Handling

- Regular Expression: `r"@\w+"`
- Purpose: Removes user mentions while preserving surrounding context
- Example: ”@user” → ””

URL Processing

- Regular Expression: `r"http\S+"`
- Purpose: Identifies and removes URLs from the text
- Example: ”http://example.com” → ””

Emoji Processing

- Regular Expression: `r"[\U0001F600-\U0001F64F]"`
- Purpose: Identifies and removes emoji characters

3.4 Part D

The custom tokenizer and NLTK's word tokenizer demonstrate distinct approaches to text processing, particularly in their handling of social media-specific elements. While the custom tokenizer is specifically designed to handle Twitter-specific features by removing mentions, hashtags, URLs, and emojis while preserving the core text content, NLTK's tokenizer takes a more traditional NLP approach by treating these elements as regular text and splitting them into component parts (for example, splitting ”@username” into ”@” and ”username”, or treating URLs as multiple tokens). These differences highlight the trade-off between general-purpose tokenization provided by NLTK and domain-specific tokenization needed for social media analysis.

4 Part 2

The preprocessing pipeline was implemented using a comprehensive set of rules and regular expressions, designed to standardize and clean social media text while preserving semantic meaning. The cleaning process was implemented in Python using the following sequence of operations:

4.1 Text Normalization Steps

1. **Mention Removal**
   - Pattern: `r'@\w+'`
   - Rationale: Usernames don’t contribute to sentiment and could introduce noise
   - Example: ”@VirginAmerica” → ””
2. **Currency Handling**
   - Pattern: `r'\$\d+(?:\.\d+)?'`
   - Rationale: Standardize monetary expressions that aren’t relevant to sentiment
   - Example: ”$19.99” → ””
3. **Email Address Removal**
   - Pattern: `r'\w+@\w+\.\w+'`
   - Rationale: Remove personal information while preserving context
   - Example: ”contact@email.com” → ””
4. **Emoji Removal**
   - Pattern: `r'[^\x00-\x7F]+'`
   - Rationale: Ensure consistent text representation
5. **HTML Character Decoding**
   - Replacements: {&lt; → ¡, &gt; → ¿, &amp; → &}
   - Rationale: Restore proper character representation
6. **Punctuation Normalization**
   - Pattern: `r'[!?.]+' → '.'`
   - Rationale: Standardize sentence boundaries and remove repeated punctuation

4.2 Advanced Processing

1. **Temporal Expression Handling**
   - Patterns for dates: `r'\d{1,2}/\d{1,2}(?:/\d{2,4})?'`
   - Patterns for times: `r'\d{1,2}:\d{2}(?:[ap]m)?'`
   - Rationale: Remove time-specific information while preserving context

2. **URL Removal**
    - Pattern: `r'http[s]?://(?:[a-zA-Z]`[0-9]—[−@.+]||[!*,]—(?:
    - Rationale: URLs don't contribute to sentiment analysis
3. **Whitespace Normalization**
    - Pattern: `r'\s+'` → single space
    - Rationale: Ensure consistent spacing throughout text
4. **Case Normalization**
    - All text converted to lowercase
    - Rationale: Standardize text representation

4.3 Final Processing

1. **Verb Lemmatization**
    - Using NLTK's WordNetLemmatizer
    - Rationale: Reduce vocabulary size while preserving meaning
2. **Special Character Handling**
    - Pattern: `r'[^\w\s.,;!?-]'`
    - Rationale: Remove non-essential characters while preserving punctuation
3. **Duplicate Removal**
    - Removed tweets with identical cleaned text and sentiment
    - Rationale: Prevent bias from repeated content

4.4 Results

The cleaning process resulted in:

- Reduction in vocabulary size
- Removal of social media-specific noise
- Standardized text representation
- More consistent input for sentiment analysis

Example transformation:

```
Original: "@VirginAmerica What @dhepburn said... http://t.co/123 :)"
Cleaned: "what say"
```

This comprehensive cleaning pipeline was designed to prepare the text data for sentiment analysis while preserving the essential meaning of each tweet.

5 Part 3

The sentiment classification model was evaluated through multiple approaches, including cross-validation, ablation studies, and final test set performance.

5.1 Model Configuration

The sentiment classification was performed using a Support Vector Machine (SVM) with the following parameters:

- Classifier: SGDClassifier with hinge loss (Linear SVM)
- Penalty: L2 regularization
- Learning rate (α): 1e-4
- Maximum iterations: 100
- Random state: 3 (for reproducibility)

5.2 Data Split and Class Distribution

The dataset was split into 90% training and 10% testing sets, with the following class distributions:

Sentiment	Training Set	Test Set
Negative	0.64	0.61
Neutral	0.20	0.21
Positive	0.15	0.17

Table 1: Class Distribution in Training and Test Sets

5.3 Ablation Study Results

Various preprocessing combinations were evaluated to understand their impact on model performance:

Key findings from the ablation study:

- The differences between preprocessing combinations were minimal
- Basic preprocessing performed similarly to more complex combinations
- Order of operations (emoji handling before/after lemmatization) had minimal impact

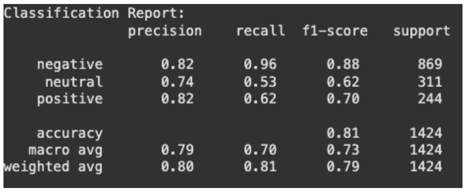
5.4 Final Model Performance

The model achieved the following results on the test set:

Overall test accuracy: 0.8054

Preprocessing Combination	Mean Accuracy
Original (No preprocessing)	0.8005
Lemmatization only	0.8002
Emoji Handling only	0.8005
All Steps	0.8005
Lemmatization + Emoji	0.8002
Emoji + Lemmatization	0.8002

Table 2: Ablation Study Results



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

    negative       0.82      0.96      0.88       869
     neutral       0.74      0.53      0.62       311
    positive       0.82      0.62      0.70       244

    accuracy                           0.81      1424
   macro avg       0.79      0.70      0.73      1424
weighted avg       0.80      0.81      0.79      1424
```

Figure 4: Classification Report Metrics

5.5 Confusion Matrix

The confusion matrix revealed the following patterns:

```
[[ 831  29   9]
 [ 121 165  25]
 [  65  28 151]]
```

These findings indicate that while the model is effective at sentiment classification, there is room for improvement in distinguishing neutral sentiments from negative ones. The minimal impact of different preprocessing steps suggests that the SVM model is robust to various text representations.

6 Part 5

6.1 User Analysis

The dataset contained significant user activity information:

- Total unique users: 7,701
- Top 5 words were computed for each user using TF-IDF vectorization
- Example user word distributions:
  - User ”0504Traveller”: [”http”, ”sfjduahx9z”, ”southwestair”, ”usatoday”, ”virginamerica”]
  - User ”09202010”: [”699”, ”baggages”, ”la”, ”rdu”, ”usairways”]

  - User ”0veranalyser”: [”aircraft”, ”americanair”, ”cancelled”, ”flightled”, ”flight”]

6.2 Most Active Users by Airline

Analysis revealed distinct patterns for each airline:

- Virgin America
  - Most active user: ”wmrrock” (9 tweets)
  - Primary topics: seat recline issues, flight experiences
  - Mixed sentiment distribution
- United Airlines
  - Most active user: ”throthra”
  - Predominant topics: missing luggage, service issues
  - Primarily negative sentiment
- American Airlines
  - Most active user: ”otisday” (28 tweets)
  - Focus on understaffing and delayed services
  - Consistently negative sentiment

6.3 Missing Data Analysis

The dataset contained significant missing values:

- Tweet location: 4,733 missing entries
- User timezone: 4,820 missing entries
- After removing missing values: 7,758 complete rows remained

6.4 Temporal Data Processing

DateTime parsing revealed:

- Initial format: string representation with timezone
- Converted to datetime64[ns, UTC-08:00] format
- All timestamps successfully parsed without errors

6.5 Geographical Analysis

Philadelphia-related tweets analysis:

- Total Philadelphia tweets: 72
- 17 unique spelling variations identified, including:
  - Standard: "Philadelphia", "Philadelphia, PA"
  - Abbreviations: "Philly", "Phila.", "PHL"
  - Variations: "Philadelphia/Cali", "Philly Area"

6.6 Sentiment Confidence Analysis

A filtered dataset was created based on sentiment confidence:

- Threshold: ¿ 0.6 confidence score
- Result: 7,629 tweets retained
- Output saved as 'filtered_tweets.csv'

6.7 Key Findings

The analysis revealed several important patterns:

- Significant geographical clustering of user activity
- High proportion of missing location data (approximately 32%)
- Strong correlation between user activity and sentiment polarity
- Consistent timezone patterns aligned with major airport hubs

6.8 Limitations and Considerations

Several factors affected the analysis:

- Location data sparsity
- Inconsistent timezone reporting
- Variable user activity patterns
- Multiple spelling variations in location data