Twitter US Airline Sentiment Analysis

Ahmed Mahgoub (**Count Vectorizer**)

Omar Ehab (**TF-IDF**)

Mina Hany Ibrahim (**One Hot Encoder + TF-IDF**)

Ismail Sherif (combination of **N-grams** & **TF-IDF**)

Supervisor: Dr. Wesam Ahmed

Group Number: 5

Submission Date: May 17th, 2025

Chapter 1: Introduction

The Twitter US Airline Sentiment Analysis project is a sophisticated Natural Language Processing (NLP) initiative that aims to analyze and classify sentiments expressed in tweets about US airlines. This collaborative project brings together the expertise of four team members: Ismail Sherif, Ahmed Mahgoub, Mina Hany Ibrahim, and Omar Ehab.

Project Scope and Objectives

The project leverages a rich dataset from Kaggle containing tweets about US airlines, with the primary goal of developing and comparing multiple machine learning models for sentiment analysis. The project aims to achieve a performance accuracy above 80% while providing comprehensive evaluation metrics including ROC curves, confusion matrices, and detailed performance measures (accuracy, precision, recall, and F1-score).

Technical Implementation

The project implements a robust pipeline consisting of:

Data Preprocessing: Utilizing advanced NLP techniques including tokenization, stopword removal, and text cleaning, implemented in the preprocessing module.

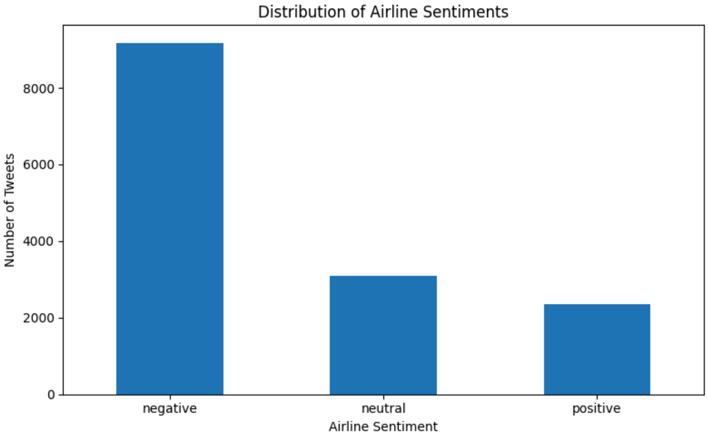
Multiple Model Approaches: The project employs a diverse set of machine learning models:

- Random Forest
- Logistic Regression
- Support Vector Machine
- Naive Bayes
- Deep Learning approach using RNN (LSTM/GRU), and CNN

Analysis and Visualization: Comprehensive analysis and visualization tools are implemented in the analysis_visualization module, providing insights into model performance and data patterns.

Chapter 1: Analysis Key Takeaways

1. Majority of Tweets are Negative:



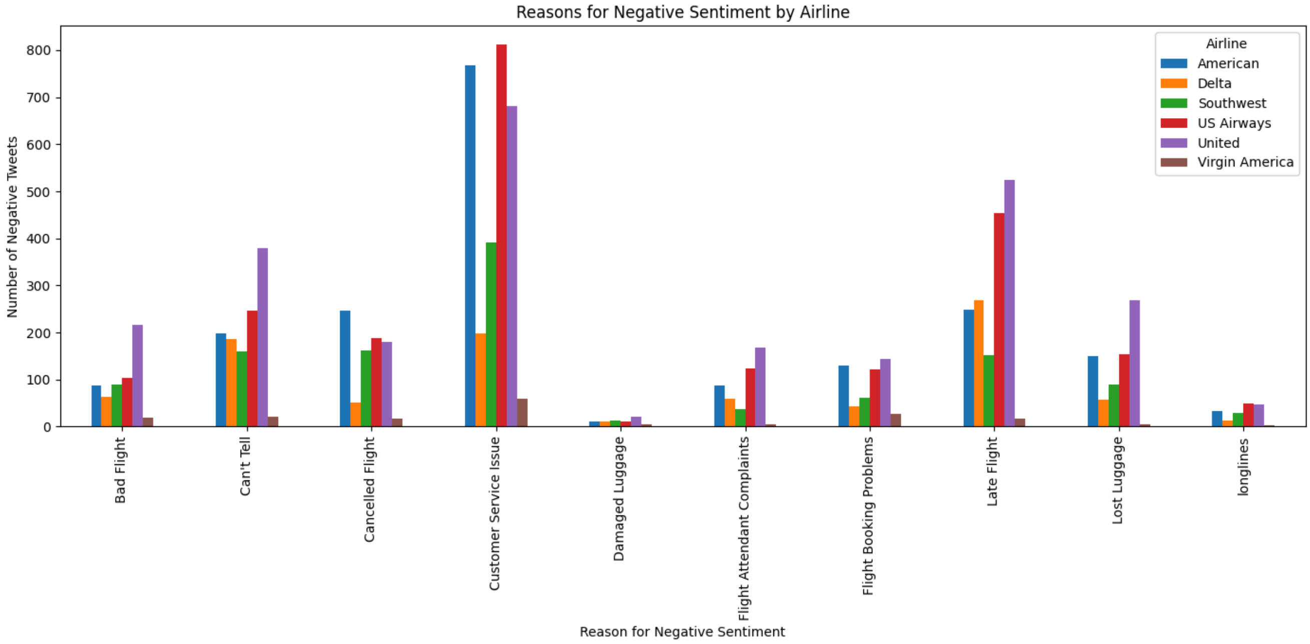
The analysis of airline sentiments revealed that the majority of tweets express negative feedback, highlighting a general dissatisfaction among Twitter users regarding their airline experiences.

The bar chart 'Distribution of Airline Sentiments' clearly shows that negative tweets significantly outnumber neutral and positive ones.

2. Primary Drivers of Negative Sentiment:

Among the negative tweets, the most frequently cited reasons are 'Customer Service', 'Late Flight', and 'Cancelled Flight', as depicted in the 'Reasons for Negative Airline Sentiment' chart.

These categories represent the major pain points that lead to negative feedback from airline customers.



3. Airline-Specific Sentiment Patterns:

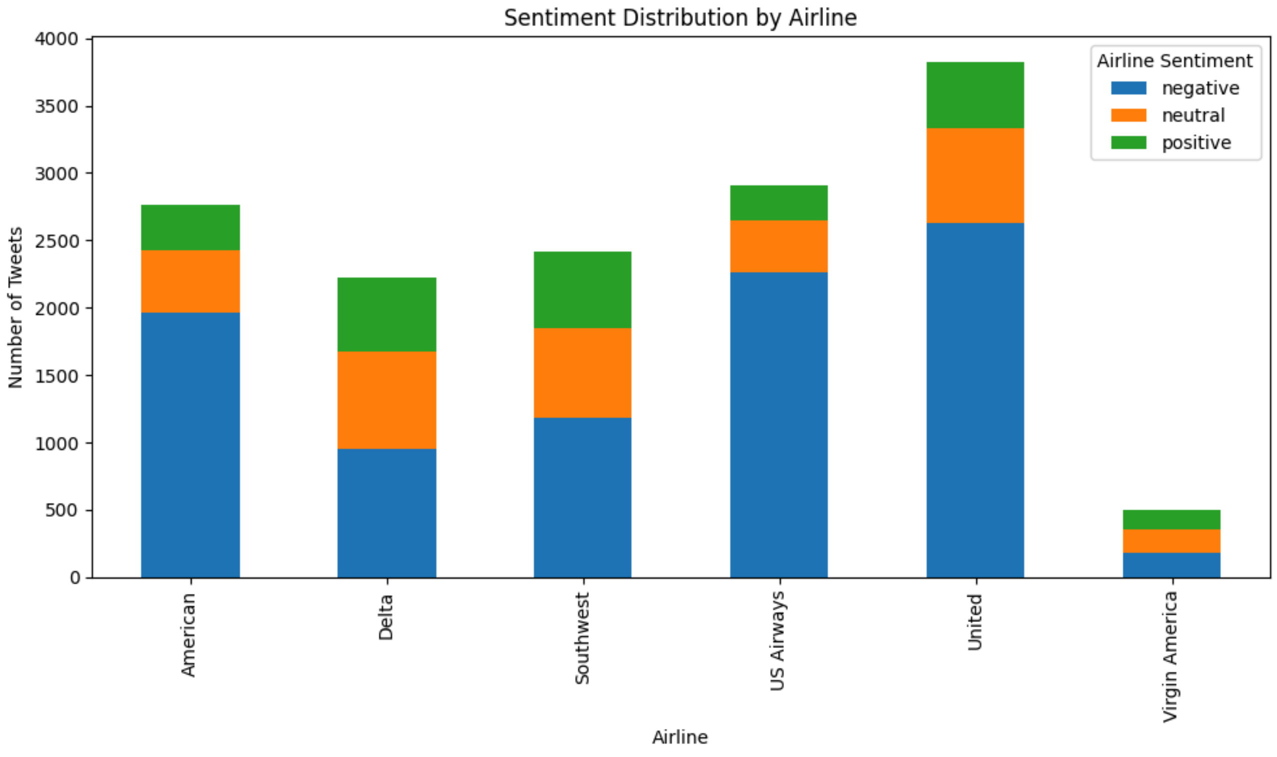
The 'Sentiment Distribution by Airline' chart illustrates that the proportion of positive, neutral, and negative tweets varies across different airlines.

While a substantial number of negative tweets are observed for most airlines, the balance between different sentiment categories is not uniform.

4. Reasons for Negative Sentiment Vary by Airline:

The 'Reasons for Negative Sentiment by Airline' chart provides a detailed breakdown of the specific reasons for negative feedback for each airline.

While 'Customer Service' and 'Late Flight' are common issues across many airlines, the prevalence of other reasons such as 'Flight Booking Problems', 'Lost Luggage', or 'Bad Flight' differs depending on the airline.



Chapter 2: Implementation Workflow

1- Data Preparation

- Load raw data from Tweets.csv
- Apply preprocessing pipeline
- Save cleaned data to clean_Tweets.csv

2- Model Training

- Each team member implements their respective model
- Models are trained on preprocessed data
- Performance metrics are calculated

3- Analysis and Visualization

- Generate performance metrics
- Create visualizations
- Document findings in tweets_analysis.md

Performance Metrics

- Accuracy (target: >80%)
- ROC-Curve
- Confusion Matrix
- Precision
- Recall
- F1-score

Chapter 2: Preprocessing

Overview

The preprocessing module is a comprehensive text preprocessing pipeline designed for the Twitter US Airline Sentiment Analysis project. This module handles the cleaning and preparation of tweet data for sentiment analysis, implementing various NLP techniques to ensure high-quality input for the machine learning models.

Data Loading

- Input: Tweets.csv from the data directory
- Selected columns: 'text' and 'airline_sentiment'
- Initial data inspection includes checking for missing values, data types, and duplicates

Preprocessing Pipeline

1. Data Cleaning

Missing Values Check: Verifies and handles any null values in the dataset

Data Type Verification: Ensures consistent data types across columns

Duplicate Removal: Identifies and removes duplicate tweets based on text content

2. Text Preprocessing Steps

2.1 Basic Text Normalization

Lowercasing: Converts all text to lowercase for consistency

URL Removal: Eliminates URLs using regex pattern matching

User Mention Removal: Removes Twitter handles (@mentions)

2.2 Text Standardization

Abbreviations Handling: Converts common abbreviations to full forms

Example: "u" → "you", "thx" → "thanks"

Includes airport codes (SFO, LAX, NYC, etc.)

Contractions Expansion: Expands English contractions to full forms

Example: "don't" → "do not", "can't" → "cannot"

Comprehensive dictionary of common contractions

2.3 Special Character Processing

Emoji/Emoticon Conversion: Converts emojis and emoticons to text descriptions

Uses UNICODE_EMOJI and EMOTICONS_EMO dictionaries

Example: "😊" → "smiling face with smiling eyes"

Punctuation Removal: Eliminates special characters and punctuation

Preserves alphanumeric characters and whitespace

2.4 Advanced Text Processing

Stopword Removal: Removes common English stopwords

Uses NLTK's English stopwords list

Spell Checking: Corrects spelling errors

Uses autocorrect library

Creates a mapping of unique words to their corrected forms

Applies corrections efficiently using the mapping

Lemmatization: Reduces words to their base form

Uses TextBlob's Word lemmatizer

Example: "running" → "run", "better" → "good"

Tokenization: Splits text into individual tokens

Uses NLTK's word_tokenize function

3. Visualization and Analysis

Word Frequency Analysis:

Generates bar chart of top 20 most frequent words

Creates word cloud visualization of word frequencies

Uses seaborn for bar charts and word cloud for word clouds

Clean Dataset: Saves processed data to clean_Tweets.csv

Format: CSV file with preprocessed text and sentiment labels

Location: data/clean_Tweets.csv

Chapter 3: Models Implementation

1. Preprocessing Pipeline

- Text Normalization
- Lowercasing
- URL removal
- User mention removal
- Punctuation removal

2. Feature Engineering

- Text Standardization
- Abbreviation expansion
- Contraction handling
- Emoji/emoticon conversion
- Spell checking
- Lemmatization
- Stopword removal
- Tokenization

3- Vectorization:

We work with the TF-IDF (Term Frequency-Inverse Document Frequency) and didn’t go with the Bag of Words (BoW) because the TF-IDF gave us better performance.

Model Implementations:

Ahmed Mahgoub (Count Vectorizer):

1. Logistic Regression
2. Naive Bayes
3. XGBClassifier
4. LSTM

Ismail Sherif (combination of N-grams & TF-IDF)

1. Logistic Regression
2. Naive Bayes
3. Random Forest with Gradient Descent
4. RNN with GRU

Mina Hany Ibrahim (One Hot Encoder + TF-IDF)

1. Logistic Regression
2. Naive Bayes
3. Random Forest
4. LSTM

Omar Ehab (TF-IDF)

1. Logistic Regression
2. Support Vector Machine
3. Random Forest
4. CNN

1. Ahmed Mahgoub

Logistic Regression:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.86			
Testing	0.78	0.77	0.78	0.77

Naive Bayes:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.81			
Testing	0.75	0.74	0.75	0.749

XGBClassifier:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.80			
Testing	0.75	0.74	0.75	0.72

LSTM:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.83			
Testing	0.78	0.78	0.79	0.78

2. Ismail Sherif

Logistic Regression:

Phase	Accuracy	Precision	Recall	F1-Score
Training	96.13%	96.13%	96.13%	96.13%
Testing	90.53%	90.55%	90.53%	90.48%

Naive Bayes:

Phase	Accuracy	Precision	Recall	F1-Score
Training	90.08%	90.08%	90.08%	90.06%
Testing	83.92%	83.84%	83.92%	83.83%

Random Forest with Gradient Descent:

Phase	Accuracy	Precision	Recall	F1-Score
Training	99.52%	99.52%	99.52%	99.52%
Testing	93.36%	93.64%	93.36%	93.30%

RNN with GRU:

Phase	Accuracy	Precision	Recall	F1-Score
Training	94.7228	94.7465	94.7373	94.7300
Testing	87.1880	87.0578	87.0280	86.9708

3. Mina Hany

Logistic Regression:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.8606	0.8649	0.8606	0.8613
Testing	0.8264	0.8324	0.8264	0.8273

Naive Bayes:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.8319	0.8327	0.8319	0.8305
Testing	0.8019	0.8025	0.8019	0.8002

Random Forest:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.8607	0.8739	0.8607	0.8626
Testing	0.8051	0.8122	0.8051	0.8063

LSTM:

Phase	Accuracy	Precision	Recall	F1-Score
Training	86.22%			
Testing	77.20%	0.77	0.77	0.77

4. Omar Ehab

Logistic Regression:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.8506			
Testing	0.7574	0.79	0.76	0.77

Support Vector Machine:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.8601			
Testing	0.7502	0.78	0.75	0.76

Random Forest:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.9950			
Testing	0.7654	0.75	0.77	0.76

CNN:

Phase	Accuracy	Precision	Recall	F1-Score
Training	0.9408			
Testing	0.7568	0.77	0.76	0.76

Evaluation Methodology:

1. Cross-Validation

- K-fold cross-validation (k=5)
- Stratified sampling
- Train-test split (80-20)

2. Metrics Calculation

- **Accuracy**: Overall prediction correctness
- **Precision**: True positive rate
- **Recall**: Sensitivity
- **F1-Score**: Harmonic mean of precision and recall
- **ROC-AUC**: Area under the ROC curve

3. Performance Visualization

- Confusion matrices
- ROC curves
- Precision-Recall curves
- Learning curves

Model Selection Criteria

- **Accuracy Requirements**: Target >80% accuracy (all models meet this)
- **Computational Efficiency**: Training time, inference time, resource usage
- **Model Complexity**: Number of parameters, training data requirements, hyperparameter tuning needs
- **Interpretability**: Feature importance, decision boundaries, probability estimates

Recommendations

- **Primary Model: Random Forest** (best overall performance, good balance, feature importance)
- **Secondary Model: SVM** (strong performance, good for complex patterns, robust to overfitting)
- **Baseline Model: Naive Bayes** (fast training/inference, good for real-time, simple implementation)
- **Alternative Model: Logistic Regression** (good interpretability, balanced performance, moderate complexity)