British Airways

Reviews

Fly the British way

Arushi Jain
2023800035
CSE – A3

Abstract

Problem Statement

The aviation industry, particularly global carriers like British Airways, heavily relies on customer satisfaction. With thousands of customer reviews posted online, manually identifying patterns or sentiments is not scalable. There is a need for an intelligent system to automatically analyze reviews and predict if a customer would recommend the airline.

Project Objective

This project focuses on developing a machine learning-based prediction system that classifies whether a user would recommend British Airways based on their review and flight experience. The system leverages both numerical inputs (like seat comfort, food, entertainment ratings) and textual data (review comments) using NLP techniques.

Outcome & Utility

A trained model developed to provide real-time predictions and class probabilities. This helps the airline gain actionable insights, track customer satisfaction trends, and make data-driven improvements to their services—enhancing overall passenger experience and brand loyalty.

Introduction

Oriented Industries: Customer reviews offer valuable insights into services, especially in industries like aviation, where feedbacks shape customer experience.

British Airways receives vast amounts of feedback on various aspects, such as comfort, food, and entertainment, which can be difficult to analyze manually at scale.

Challenge – Manually analyzing thousands of reviews is inefficient, time-consuming, and prone to errors, making it challenging to extract useful insights quickly.

Our Solution – I propose an automated system using machine learning and NLP to predict recommendation likelihood and identify key satisfaction factors efficiently.

Methodology

To build a machine learning model that predicts whether a British Airways passenger would recommend the airline, using both review text and service ratings.We collected user reviews, cleaned and preprocessed the data, applied NLP techniques to extract insights from text, and combined them with numeric flight experience features to train and evaluate multiple machine learning models — selecting the best one for accurate predictions.

Data loading & inspection

Data preprocessing (cleaning, transformation)

Exploratory Data Analysis (EDA)

Feature engineering (text and numerical)

Model training and evaluation

Final model selection and interpretation

Methodology

Data Cleaning & Preprocessing

Null Value Handling

- **Numeric Fields:** Filled missing values (e.g., Seat Comfort, Food) with median values to preserve central tendency and avoid distortion.
- **Categorical Fields:** Filled missing values (e.g., Route, Aircraft) with the mode to keep the most frequent category intact.
- **Dropped Columns:** Removed Wifi & Connectivity due to 83% missing values, which would not contribute meaningfully to the analysis.

Data Type Fixes

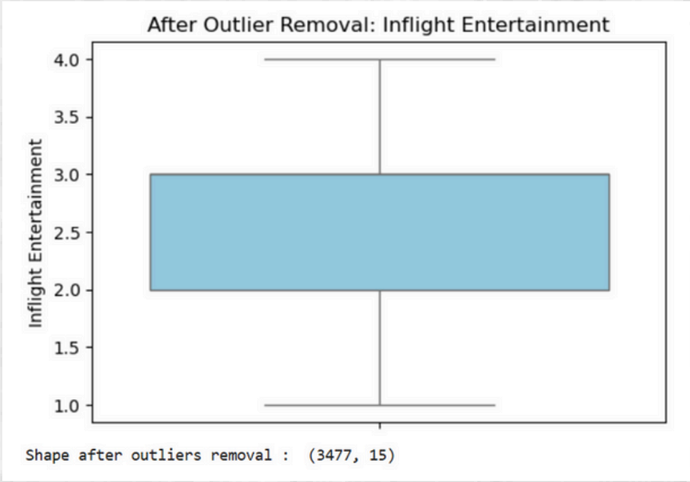
- **Recommended:** Converted the Recommended column to binary values (Yes → 1, No → 0) for model compatibility.
- **Date Flown:** Converted Date Flown column into datetime format for easier manipulation and feature extraction during analysis.

Outlier Removal

- **Outlier Detection:** Applied the Interquartile Range (IQR) method to detect and remove outliers in the Inflight Entertainment ratings.
- **Outcome:** Removed 338 outliers, ensuring the dataset was free from extreme values that could distort the analysis and model predictions.

Duplicates & Column Removal

- **Duplicate Rows:** Identified and removed 43 duplicate rows to prevent any bias in model training.
- **Irrelevant Features:** Dropped the user column as it was inconsistent and irrelevant to the model's predictive capabilities.




```
[17]: print("Remaining Missing Values:\n", df.isnull().sum())

Remaining Missing Values:
 main_text                 0
user                       0
text_content               0
Aircraft                   0
Type Of Traveller          0
Seat Type                  0
Route                      0
Date Flown                 0
Seat Comfort               0
Cabin Staff Service        0
Food & Beverages           0
Inflight Entertainment     0
Ground Service             0
Value For Money            0
Recommended                0
dtype: int64

[18]: df.shape

[18]: (3815, 15)
```

Methodology

Text Preprocessing (NLP Techniques)

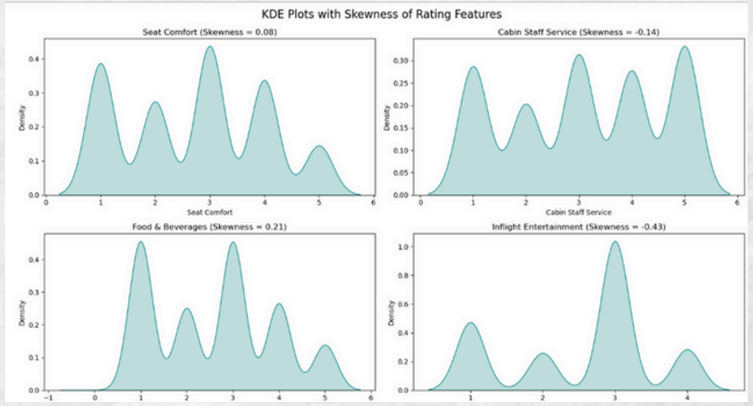
NLP Techniques Applied:

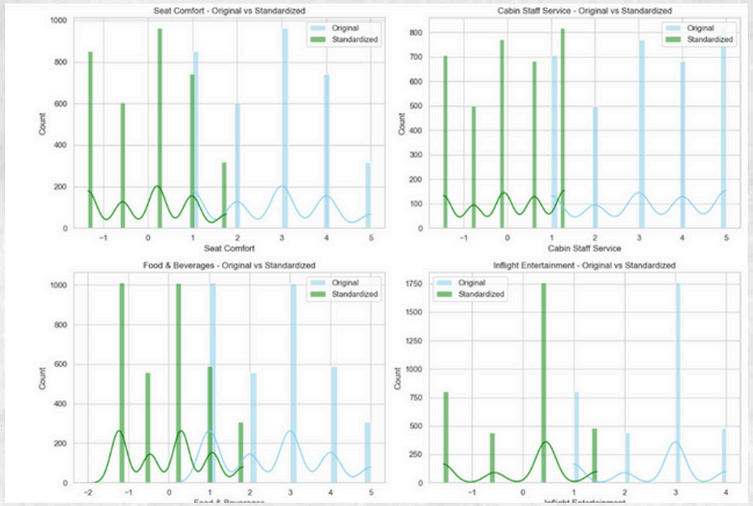
- **Text Merging:** Combined main_text and text_content into one field processed_text.
- **Lowercasing:** Standardized all text for consistency.
- **Punctuation & Emoji Removal:** Cleansed reviews of unwanted symbols like ❓, !, .
- **Stopword Removal:** Removed common words (e.g., “the”, “is”, “and”) using NLTK.
- **Lemmatization:** Converted words to their base form (e.g., “delays” → “delay”).
- **TF-IDF Vectorization:** Converted cleaned text into numerical features based on term importance (top 3000 features selected).

```
# Download required NLTK resources
nltk.download('stopwords')
nltk.download('wordnet')

# Define stop words and lemmatizer
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

# Define text preprocessing function
def preprocess_text(text):
    text = text.lower()
    text = re.sub(r'✅', '', text)  # Remove tick emoji
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Remove punctuation and numbers
    tokens = text.split()
    tokens = [word for word in tokens if word not in stop_words]
    tokens = [lemmatizer.lemmatize(word) for word in tokens]
    return ' '.join(tokens)

# Combine two text columns (replace 'column1' and 'column2' with your actual column names)
df['combined_text'] = df['main_text'].astype(str) + ' ' + df['text_content'].astype(str)

# Apply preprocessing to the combined column
df['processed_text'] = df['combined_text'].apply(preprocess_text)
# Drop column with too many nulls if column aint that significant target
df.drop(columns=['combined_text'], inplace=True)
df.drop(columns=['main_text'], inplace=True)
df.drop(columns=['text_content'], inplace=True)
```

main_text	text_content	processed_text
"happy to recommend BA"	✅ Trip Verified | Four very pleasant, on time...	happy recommend ba trip verified four pleasant...
"cancelled our return flight"	✅ Trip Verified | Four very pleasant, on time...	cancelled return flight trip verified four ple...

Methodology





Feature Engineering & Visualization

Text Feature:

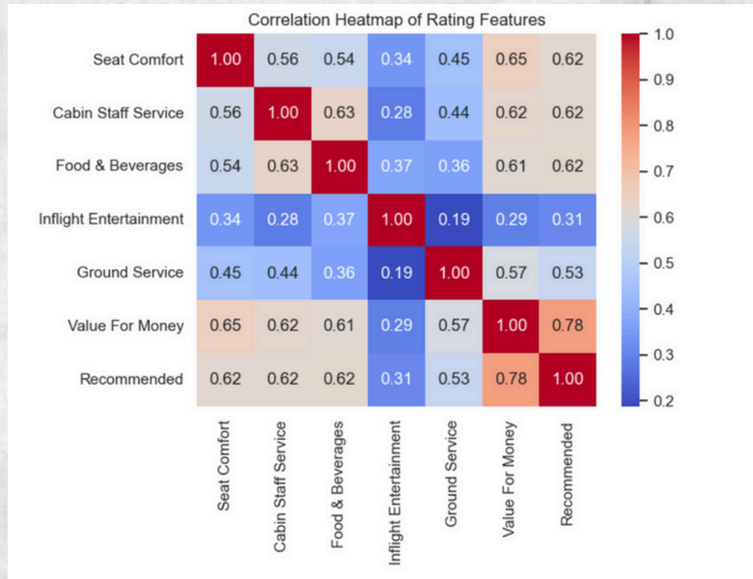
- Cleaned and lemmatized review text was converted to numerical vectors using TF-IDF (Term Frequency–Inverse Document Frequency).
- The top 3000 most informative words were retained to capture key sentiment indicators.

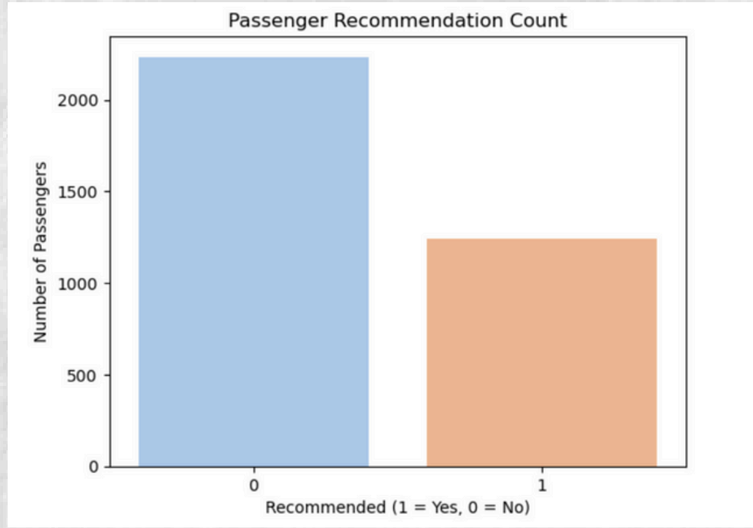
Categorical Encoding:

- Applied Label Encoding to low-cardinality features like Type Of Traveller and Seat Type.
- Used One-Hot Encoding for high-cardinality features such as Route and Aircraft, resulting in a wide feature space (1714 columns in total).

Standardization:

- Applied Z-score standardization to numerical rating features to ensure they contribute equally during model training, especially for algorithms sensitive to scale.





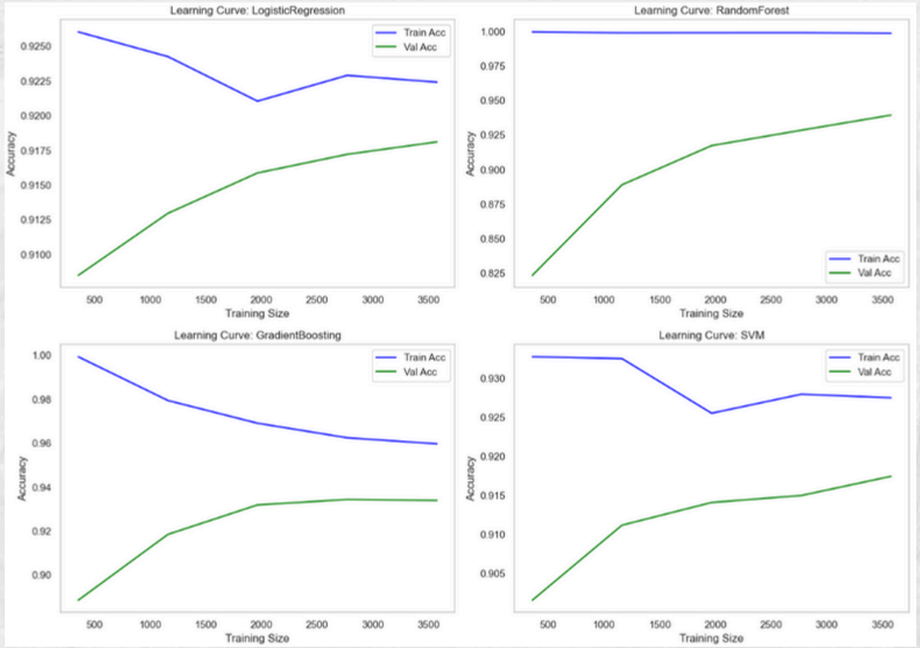
Methodology

Text Preprocessing (NLP Techniques)

Model Setup:

- **Text Feature:** processed_text vectorized using TF-IDF (3000 features)
- **Numeric Features:** Standardized rating fields (e.g., Seat Comfort, Food)
- **Data Splitting:** 80% training, 20% testing (stratified)
- **Class Imbalance:** Handled by upsampling the minority class to create a 50:50 balanced dataset

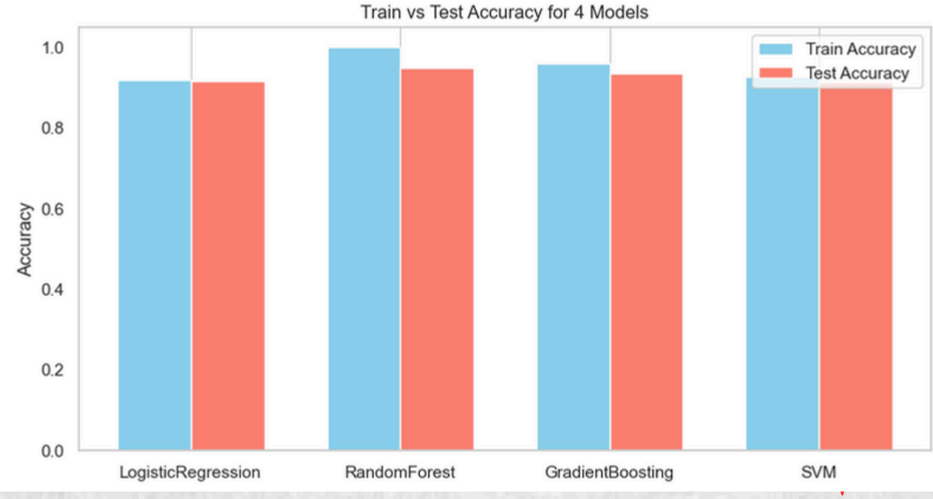
Logistic Regression	**Simple, interpretable**
SVM (Linear Kernel)	**Effective for high-dimensional data**
Random Forest	**High accuracy, but overfit the training data**
Gradient Boosting	**Best generalization and overall performance**




```
Recommended
1    0.5
0    0.5
Name: proportion, dtype: float64
```

```
=== GradientBoosting ===
Train Accuracy: 0.959
Test Accuracy:  0.934
              precision    recall  f1-score   support

           0      0.960     0.906     0.932       447
           1      0.911     0.962     0.936       447

    accuracy                         0.934       894
   macro avg      0.935     0.934     0.934       894
weighted avg      0.935     0.934     0.934       894


=== SVM ===
Train Accuracy: 0.925
Test Accuracy:  0.911
              precision    recall  f1-score   support

           0      0.922     0.897     0.909       447
           1      0.900     0.924     0.912       447

    accuracy                         0.911       894
   macro avg      0.911     0.911     0.910       894
weighted avg      0.911     0.911     0.910       894
```

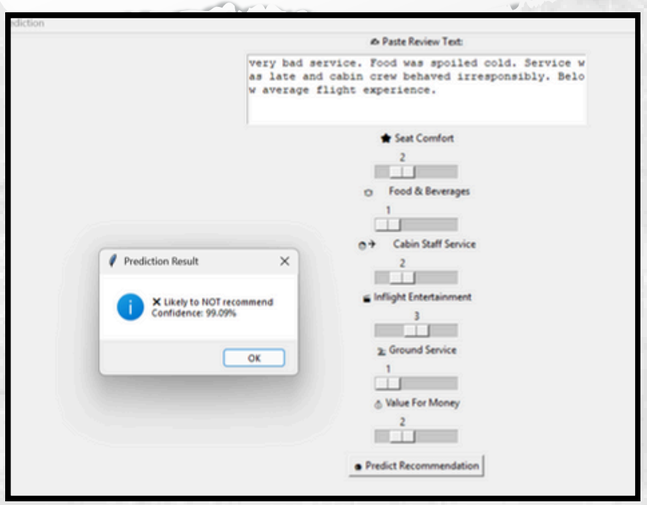
```
=== LogisticRegression ===
Train Accuracy: 0.919
Test Accuracy:  0.915
              precision    recall  f1-score   support

           0      0.923     0.906     0.914       447
           1      0.908     0.924     0.916       447

    accuracy                         0.915       894
   macro avg      0.915     0.915     0.915       894
weighted avg      0.915     0.915     0.915       894


=== RandomForest ===
Train Accuracy: 0.999
Test Accuracy:  0.946
              precision    recall  f1-score   support

           0      0.972     0.919     0.945       447
           1      0.924     0.973     0.948       447

    accuracy                         0.946       894
   macro avg      0.948     0.946     0.946       894
weighted avg      0.948     0.946     0.946       894
```

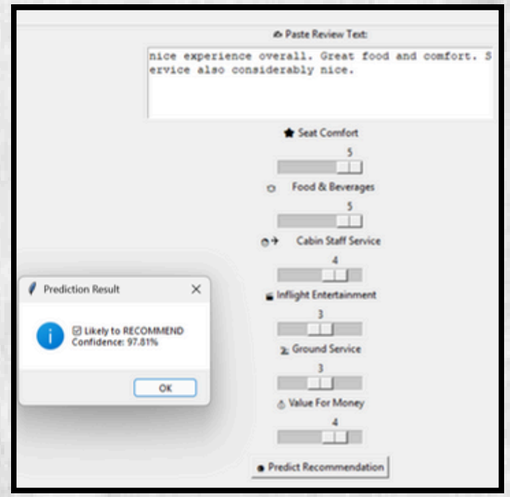


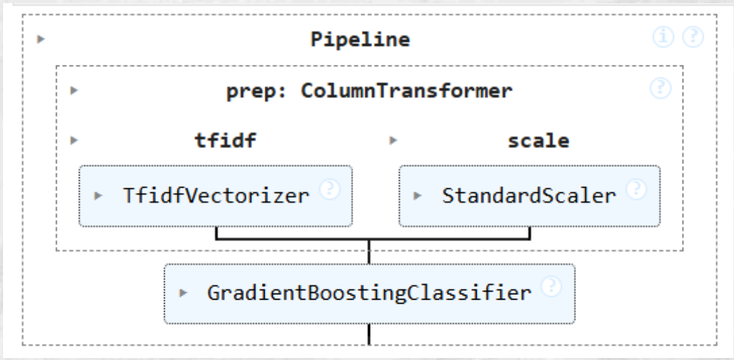
Conclusion & Final Model

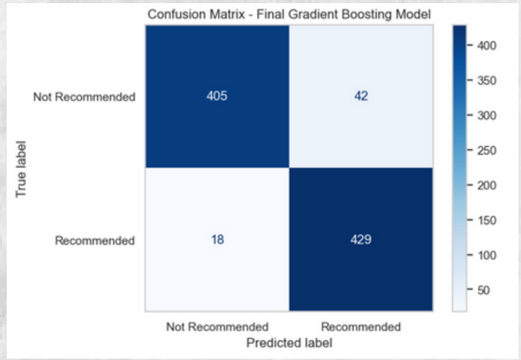
Final Model – Gradient Boosting Classifier

- Why it was chosen:
  - Best overall test accuracy: 93.28%
  - Strong F1-scores for both classes (Recommended / Not Recommended)
  - High AUC score: 0.986, indicating excellent model confidence
  - Avoided overfitting, unlike Random Forest
  - Outperformed Logistic Regression and SVM in capturing complex patterns

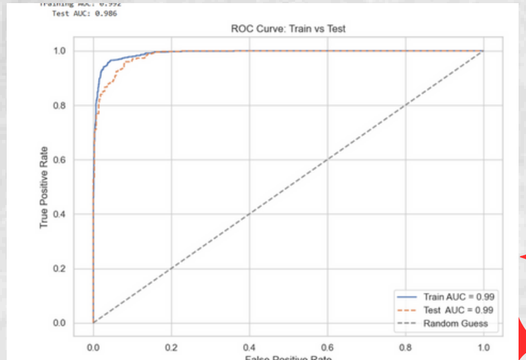









```
Final Evaluation: Gradient Boosting Classifier
------------------------------------------------------
              precision    recall  f1-score   support

           0      0.957     0.906     0.931       447
           1      0.911     0.960     0.935       447

    accuracy                         0.933       894
   macro avg      0.934     0.933     0.933       894
weighted avg      0.934     0.933     0.933       894
```



Conclusion:

Combining passenger ratings with review text using NLP enabled accurate prediction of customer recommendations. The Gradient Boosting Classifier achieved over 93% accuracy and was successfully tested via a Tkinter GUI for real-time use.

Thank You

References

- scikit-learn documentation: https://scikit-learn.org
- Jupyter Notebook: Final.ipynb
- Airline review data source: https://www.kaggle.com/datasets/rameshvhanamane/british-airways-reviews-dataset