CS410 Group Project

Sentiment Analysis on Disneyland Reviews

Team Members：

Jingxin Deng (jdeng19): jdeng19@illinois.edu
Yujia Qiu(yujiaq5): yujiaq5@illinois.edu
Luxia Yin(luxiay2): luxiay2@illinois.edu

Introduction
Customer satisfaction reflects consumer's perception of products, services, and organizations. Extracting meaningful insights from customer's feedback can be challenging due to the unstructured nature of textual data. Sentiment Analysis is a powerful technique that helps identify and classify the polarity of opinions as positive, negative, or neutral. By leveraging sentiment analysis, decision-makers can track changes in customer sentiment related to products and services, enabling organizations to enhance their offerings and effectively improve customer's experience.

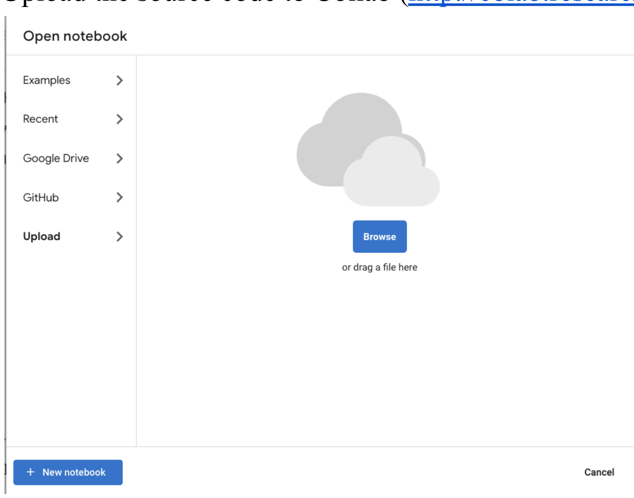
In this project, we conducted sentiment analysis on Disneyland Reviews data to develop a model capable of predicting customer's rating based on review content. This analysis offers valuable insights by identifying frequently used words in positive and negative feedback, uncovering factors commonly highlighted by visitors. These insights empower Disneyland to prioritize improvements that align with visitor's needs, address recurring issues, and enhance overall satisfaction. Additionally, the findings support more effective marketing strategies and customer service enhancements, reinforcing Disneyland's reputation as a premier destination for creating magical experiences.

Data

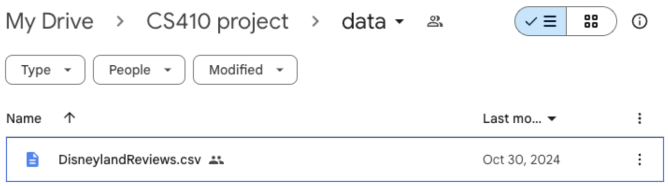
The Disneyland Reviews dataset, publicly available on Kaggle (https://www.kaggle.com/datasets/arushchillar/disneyland-reviews/data), was used for this project.

Software Usage
For this project, we have used Google Collab.

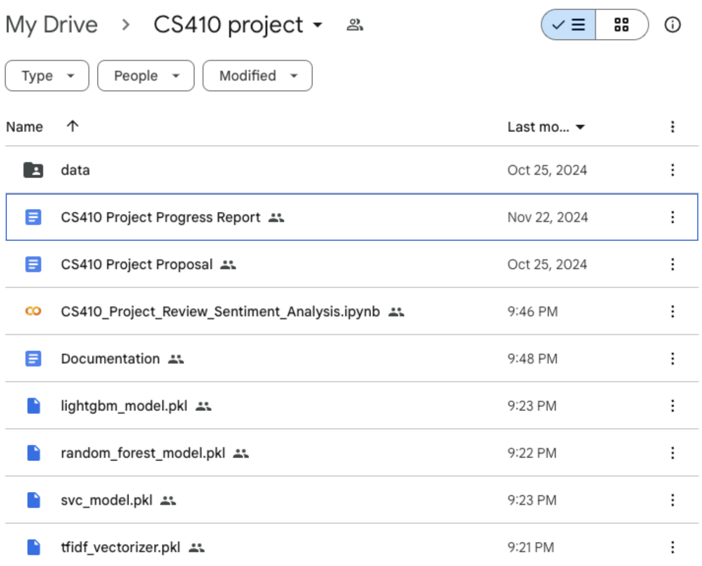
1. First login to the google account. Create a new account if you don't have one already.
1. Download the source code from GitHub
2. https://github.com/jdeng19/CS410-Fall24-Project/blob/main/CS410_Project_Review_Sentiment_Analysis.ipynb
3. Upload the source code to Collab (http://colab.research.google.com)



4. Download the data/DisneylandReviews.csv from GitHub and upload the data folder to Google Drive.



5. Download all of the models and vectorizer (.pkl files) from GitHub and upload to Google Drive.



6. Copy the file path from the Google Drive and replace the PATH and url where you saved the dataset and models (at the top of the notebook).

```
[15] from google.colab import drive
     drive.mount('/content/drive')

     Drive already mounted at /content/drive; to attempt to forcibly remount, ca

[16] path = '/content/drive/MyDrive/CS410 project/'

[17] url = path+'/data/DisneylandReviews.csv'
     df = pd.read_csv(url, encoding="cp1252")
```

7. Import required libraries (at the top of the notebook)

```python
import numpy as np
import pandas as pd
import os
import matplotlib.pyplot as plt
plt.style.use('ggplot')
import seaborn as sns

from wordcloud import WordCloud
import re

import random

import nltk
nltk.download('punkt')
nltk.download('punkt_tab')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('omw-1.4')
nltk.download('vader_lexicon')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer, PorterStemmer

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from scipy.sparse import hstack

from imblearn.over_sampling import RandomOverSampler
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.svm import LinearSVC
from sklearn.ensemble import RandomForestClassifier
from lightgbm import LGBMClassifier, early_stopping, log_evaluation
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from nltk.sentiment import SentimentIntensityAnalyzer
import joblib
from textblob import TextBlob

import warnings
warnings.filterwarnings("ignore")
```

8. Run following cells (at the end of the notebook):

```python
[143] analyzer = SentimentIntensityAnalyzer()

def get_sentiment_scores(text):
    sentiment = analyzer.polarity_scores(text)
    return pd.Series([sentiment['neg'], sentiment['neu'], sentiment['pos'], sentiment['compound']])
```

```python
def preprocess_review(review_text):
    # Calculate the number of characters and sentences
    num_of_characters = len(review_text)
    num_of_sentences = len(nltk.sent_tokenize(review_text))

    # Get sentiment scores
    neg, neu, pos, compound = get_sentiment_scores(review_text)
    vectorizer = joblib.load(path+'tfidf_vectorizer.pkl')
    tfidf_vector = vectorizer.transform([review_text])

    additional_features = pd.DataFrame({
        'Negative': [neg],
        'Neutral': [neu],
        'Positive': [pos],
        'Compound': [compound],
        'num_of_characters': [num_of_characters],
        'num_of_sentences': [num_of_sentences]
    })

    print(additional_features)

      # Concatenate TF-IDF features and additional features
    combined_features = pd.concat(
        [pd.DataFrame(tfidf_vector.toarray()), additional_features.reset_index(drop=True)],
        axis=1
    )
    return combined_features
```

```
def predict_review_rating(review_text):
  model = joblib.load(path+'random_forest_model.pkl')

  processed_review = preprocess_review(review_text)
  processed_review.columns = processed_review.columns.astype(str)


  labels = ['neutral', 'satisfied', 'unsatisfied']
  label_encoder = LabelEncoder()
  label_encoder.fit(labels)

  predicted_encoded_label = model.predict(processed_review)[0]
  print(predicted_encoded_label)

  predicted_category = label_encoder.inverse_transform([predicted_encoded_label])[0]
  print(f"Predicted Category: {predicted_category}")



# Example review text
review_text = "omg it's tired"
predict_review_rating(review_text)
```

```
   Negative  Neutral  Positive  Compound  num_of_characters  num_of_sentences
0     0.592    0.408       0.0   -0.4404                 14                 1
2
Predicted Category: unsatisfied
```

9. You can always edit ‘model’ with corresponding model .pkl file, ‘review_text’ to predict different reviews with different models.

Exploratory Data Analysis

```
print(df.head())
```

```
   Review_ID  Rating Year_Month     Reviewer_Location  \
0  670772142       4     2019-4             Australia
1  670682799       4     2019-5           Philippines
2  670623270       4     2019-4  United Arab Emirates
3  670607911       4     2019-4             Australia
4  670607296       4     2019-4        United Kingdom

                                         Review_Text               Branch
0  If you've ever been to Disneyland anywhere you...  Disneyland_HongKong
1  Its been a while since d last time we visit HK...  Disneyland_HongKong
2  Thanks God it wasn   t too hot or too humid wh...  Disneyland_HongKong
3  HK Disneyland is a great compact park. Unfortu...  Disneyland_HongKong
4  the location is not in the city, took around 1...  Disneyland_HongKong
```

The Disneyland Reviews dataset consists of 42,656 rows and 6 features, capturing feedback from visitors across different branches of Disneyland. The features include a unique Review_ID (42,636 unique values), a Rating (5 unique values, ranging from 1 to 5), the Year_Month of the review (112 unique values), the Reviewer_Location (162 unique values), the Review_Text (42,632 unique values), and the Branch (3 unique values, representing different Disneyland locations).

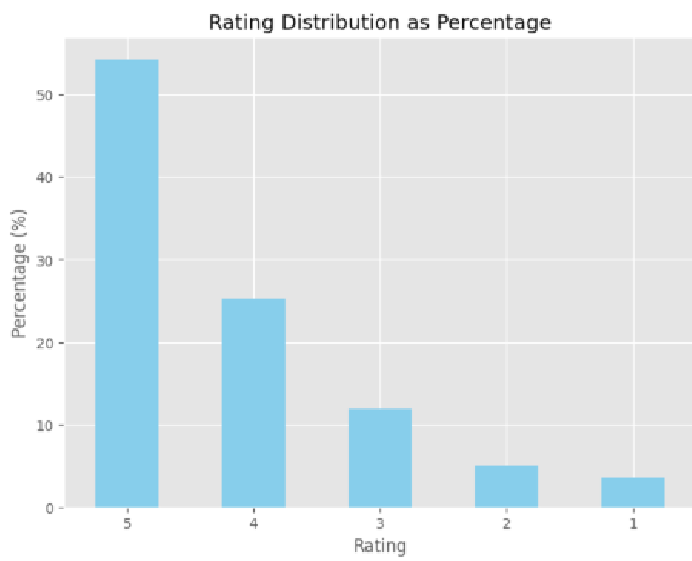
```
Rows     :  42656
Columns  :  6

Features :
 ['Review_ID', 'Rating', 'Year_Month', 'Reviewer_Location', 'Review_Text', 'Branch']

Missing values :   0

Unique values :
 Review_ID           42636
Rating                  5
Year_Month            112
Reviewer_Location     162
Review_Text         42632
Branch                  3
dtype: int64
```

We focused on extracting the Review_Text and Rating columns, which served as the target variable for sentiment analysis. The ratings range from 1 to 5, where 5 represents the highest level of customer satisfaction. To better understand the data, we created a bar chart to visualize the Rating Distribution as a Percentage, offering insights into customer satisfaction levels. The chart reveals that approximately 80% of customers rated their experience as 5 or 4, indicating high satisfaction. However, 12% of customers gave a rating of 3, and 8% rated their experience as 2 or 1, highlighting areas that may require improvement.



Text Preprocessing

To prepare the data for analysis, we implemented the following preprocessing steps:

1. **Lowercasing**: All reviews were converted to lowercase to ensure uniformity.
2. **Removal of Unwanted Patterns**: URLs, handles, punctuations, and special characters were removed from the processed_text column.
3. **Stemming and Lemmatization**: Words were reduced to their base or root forms to simplify and standardize text analysis.
4. **Stop Word Removal**: Common stop words (e.g., "the," "and") were removed to focus on meaningful terms. Additionally, frequent but less informative words such as "thi," "ride," "park," "one," "disney," and "wa" were excluded to enhance text clarity.

	Review_ID	Rating	Year_Month	Reviewer_Location	Review_Text	Branch	processed_text	stemmed_text	cleaned_text
0	670772142	4	2019-4	Australia	If you've ever been to Disneyland anywhere you...	Disneyland_HongKong	if youve ever been to disneyland anywhere youl...	if youv ever been to disneyland anywher youll ...	youv ever anywher youll find hong kong veri si...
1	670682799	4	2019-5	Philippines	Its been a while since d last time we visit HK...	Disneyland_HongKong	its been a while since d last time we visit hk...	it been a while sinc d last time we visit hk d...	sinc last visit hk yet stay tomorrowland aka m...
2	670623270	4	2019-4	United Arab Emirates	Thanks God it wasn t too hot or too humid wh...	Disneyland_HongKong	thanks god it wasn t too hot or too humid wh...	thank god it wasn t too hot or too humid when ...	thank god hot humid visit otherwis would big i...
3	670607911	4	2019-4	Australia	HK Disneyland is a great compact park. Unfortu...	Disneyland_HongKong	hk disneyland is a great compact park unfortun...	hk disneyland is a great compact park unfortun...	hk great compact unfortun quit bit mainten wor...
4	670607296	4	2019-4	United Kingdom	the location is not in the city, took around 1...	Disneyland_HongKong	the location is not in the city took around 1 ...	the locat is not in the citi took around 1 hou...	locat citi took around 1 hour kowlon like much...

We then added new features to calculate the number of characters and sentences in each review.

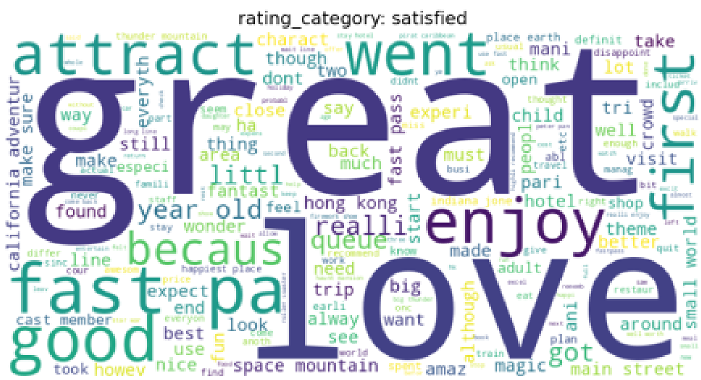
	Review_ID	Rating	Year_Month	Reviewer_Location	Review_Text	Branch	processed_text	stemmed_text	cleaned_text	num_of_characters	num_of_sentences
0	670772142	4	2019-4	Australia	If you've ever been to Disneyland anywhere you...	Disneyland_HongKong	if youve ever been to disneyland anywhere youl...	if youv ever been to disneyland anywher youll ...	youv ever anywher youll find hong kong veri si...	329	4
1	670682799	4	2019-5	Philippines	Its been a while since d last time we visit HK...	Disneyland_HongKong	its been a while since d last time we visit hk...	it been a while sinc d last time we visit hk d...	sinc last visit hk yet stay tomorrowland aka m...	970	19
2	670623270	4	2019-4	United Arab Emirates	Thanks God it wasn t too hot or too humid wh...	Disneyland_HongKong	thanks god it wasn t too hot or too humid wh...	thank god it wasn t too hot or too humid when ...	thank god hot humid visit otherwis would big i...	938	4
3	670607911	4	2019-4	Australia	HK Disneyland is a great compact park. Unfortu...	Disneyland_HongKong	hk disneyland is a great compact park unfortun...	hk disneyland is a great compact park unfortun...	hk great compact unfortun quit bit mainten wor...	485	3
4	670607296	4	2019-4	United Kingdom	the location is not in the city, took around 1...	Disneyland_HongKong	the location is not in the city took around 1 ...	the locat is not in the citi took around 1 hou...	locat citi took around 1 hour kowlon like much...	163	2

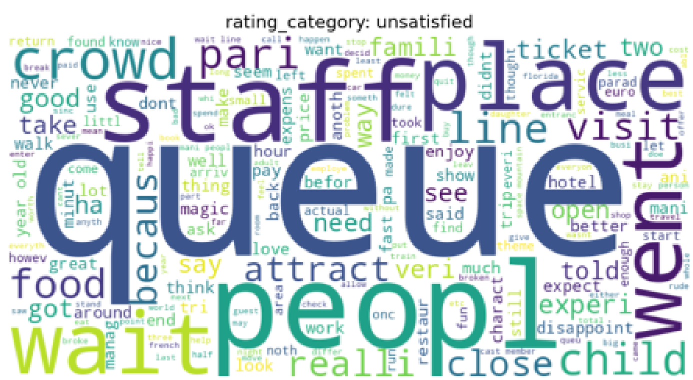
Next, we re-labeled the review text based on customer ratings into three categories for sentiment analysis:

- Unsatisfied: Ratings of 1 and 2.
- Neutral: Rating of 3.
- Satisfied: Ratings of 4 and 5.

	Review_ID	Rating	Year_Month	Reviewer_Location	Review_Text	Branch	processed_text	stemmed_text	cleaned_text	num_of_characters	num_of_sentences	rating_category
0	670772142	4	2019-4	Australia	If you've ever been to Disneyland anywhere you...	Disneyland_HongKong	if youve ever been to disneyland anywhere youl...	if youv ever been to disneyland anywher youll ...	youv ever anywher youll find hong kong veri si...	329	4	satisfied
1	670682799	4	2019-5	Philippines	Its been a while since d last time we visit HK...	Disneyland_HongKong	its been a while since d last time we visit hk...	it been a while sinc d last time we visit hk d...	sinc last visit hk yet stay tomorrowland aka m...	970	19	satisfied
2	670623270	4	2019-4	United Arab Emirates	Thanks God it wasn t too hot or too humid wh...	Disneyland_HongKong	thanks god it wasn t too hot or too humid wh...	thank god it wasn t too hot or too humid when ...	thank god hot humid visit otherwis would big i...	938	4	satisfied
3	670607911	4	2019-4	Australia	HK Disneyland is a great compact park. Unfortu...	Disneyland_HongKong	hk disneyland is a great compact park unfortun...	hk disneyland is a great compact park unfortun...	hk great compact unfortun quit bit mainten wor...	485	3	satisfied
4	670607296	4	2019-4	United Kingdom	the location is not in the city, took around 1...	Disneyland_HongKong	the location is not in the city took around 1 ...	the locat is not in the citi took around 1 hou...	locat citi took around 1 hour kowlon like much...	163	2	satisfied

Finally, we generated Word Clouds to visualize the most frequently used words in each rating category, offering a clearer picture of customer sentiment across different satisfaction levels.





Data Processing

The primary goal of this stage is to preprocess the Disneyland review dataset by extracting meaningful features, combining numerical and textual data, and preparing the dataset for model training and evaluation.

1. **Separate Features and Labels**

   To analyze the sentiment of reviews, we appled two sentiment analysis toos:

   - Using VADER to extract sentiment features such as Negative, Neutral, Positive, and Compound sentiment scores.
   - Adjusted sentiment scores (Positive and Neutral) for low ratings (ratings 1 and 2) by scaling them down (multiplied by 0.75) to better reflect the review tone.

- Using TextBlob to generate additional sentiment features, including Polarity and Subjectivity.
- The processed sentiment features were then combined with other numerical features to enrich the dataset.

Review_ID	Rating	Year_Month	Reviewer_Location	Review_Text	Branch	processed_text	stemmed_text	cleaned_text	num_of_characters	num_of_sentences	rating_category	Negative	Neutral	Positive	Compound
670772142	4	2019-4	Australia	If you've ever been to Disneyland anywhere you...	Disneyland_HongKong	if youve ever been to disneyland anywhere youl...	if youv ever been to disneyland anywher youll ...	youv ever anywher youll find hong kong veri si...	329	4	satisfied	0.000	0.113	0.113	0.7069
670682799	4	2019-5	Philippines	Its been a while since d last time we visit HK...	Disneyland_HongKong	its been a while since d last time we visit hk...	it been a while sinc d last time we visit hk d...	sinc last visit hk yet stay tomorrowland aka m...	970	19	satisfied	0.040	0.231	0.231	0.9901
670623270	4	2019-4	United Arab Emirates	Thanks God it wasn t too hot or too humid wh...	Disneyland_HongKong	thanks god it wasn t too hot or too humid wh...	thank god it wasn t too hot or too humid when ...	thank god hot humid visit otherwis would big i...	938	4	satisfied	0.024	0.235	0.235	0.9920
670607911	4	2019-4	Australia	HK Disneyland is a great compact park. Unfortu...	Disneyland_HongKong	hk disneyland is a great compact park unfortun...	hk disneyland is a great compact park unfortun...	hk great compact unfortun quit bit mainten wor...	485	3	satisfied	0.080	0.160	0.160	0.8489
670607296	4	2019-4	United Kingdom	the location is not in the city, took around 1...	Disneyland_HongKong	the location is not in the city took around 1 ...	the locat is not in the citi took around 1 hou...	locat citi took around 1 hour kowlon like much...	163	2	satisfied	0.000	0.101	0.101	0.2846

- Use labelencoder to convert categorical labels in y into numerical values for model compatibility

```
                                        cleaned_text  Negative  Neutral  \
0  youv ever anywher youll find hong kong veri si...     0.000    0.113   
1  sinc last visit hk yet stay tomorrowland aka m...     0.040    0.231   
2  thank god hot humid visit otherwis would big i...     0.024    0.235   
3  hk great compact unfortun quit bit mainten wor...     0.080    0.160   
4  locat citi took around 1 hour kowlon like much...     0.000    0.101   

   Positive  Compound  num_of_characters  num_of_sentences  
0     0.113    0.7069                329                 4  
1     0.231    0.9901                970                19  
2     0.235    0.9920                938                 4  
3     0.160    0.8489                485                 3  
4     0.101    0.2846                163                 2  
```

2. Data Splitting

The preprocessed data was split into training and testing sets using an 80/20 split:

- Ensured stratification to maintain label distribution across training and testing sets.
- Verified feature and label shapes to ensure proper splitting.

```
# Split the dataset into training and testing sets (e.g., 80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=101, stratify=y)

print("Training set size:", X_train.shape)
print("Testing set size:", X_test.shape)
```

3. Text Feature Transformation with TF-IDF

TF-IDF Vectorization:

The cleaned review text was converted into numerical features using TF-IDF with following parameters:

- N-gram range: (2, 3)

- Maximum features: 5000

The TF-IDF vectorizer was trained on the training set and applied to both training and testing datasets.

```
# Initialize the TF-IDF Vectorizer
vectorizer = TfidfVectorizer(ngram_range=(2, 3), max_features=5000)

# Fit the vectorizer on the training data and transform both training and testing sets

X_train_tfidf = vectorizer.fit_transform(X_train['cleaned_text'])
X_test_tfidf = vectorizer.transform(X_test['cleaned_text'])

# Check the shape to ensure the transformation is successful
print("TF-IDF training set shape:", X_train_tfidf.shape)
print("TF-IDF testing set shape:", X_test_tfidf.shape)

joblib.dump(vectorizer, path+'tfidf_vectorizer.pkl')
```

Feature Combination:

- Extracted additional numerical features and combined them with TF-IDF features.
- Used hstack to merge sparse matrices from textual and numerical features.
- Combined the numerical features with the TF-IDF features using hstack to form a single sparse matrix for both training and testing sets.

```
# Extract and split the variables besides text context for train and test sets
X_train_sentiments = X_train[['Negative', 'Neutral', 'Positive', 'Compound','num_of_characters', 'num_of_sentences']]
X_test_sentiments = X_test[['Negative', 'Neutral', 'Positive', 'Compound','num_of_characters', 'num_of_sentences']]

# Convert the sentiment score features to sparse matrices
X_train_sentiments_sparse = X_train_sentiments.values
X_test_sentiments_sparse = X_test_sentiments.values

# # Combined the numerical features with the TF-IDF features using hstack
X_train_combined = hstack([X_train_tfidf, X_train_sentiments_sparse])
X_test_combined = hstack([X_test_tfidf, X_test_sentiments_sparse])

# Display the shapes of the combined features
print("X_train_tfidf shape:", X_train_tfidf.shape)
print("X_train_branch shape:", X_train_sentiments_sparse.shape)
print("X_train_combined shape:", X_train_combined.shape)

print("X_test_tfidf shape:", X_test_tfidf.shape)
print("X_test_branch shape:", X_train_sentiments_sparse.shape)
print("X_test_combined shape:", X_test_combined.shape)
```

Handle Imbalanced Dataset

- We used SMOTE (Synthetic Minority Oversampling Technique) to handle imbalanced datasets and improve the classifier's performance. It creates synthetic samples of the minority class to balance the class distribution.

```
[138] # ros = RandomOverSampler(random_state=42)
      # X_train_resampled, y_train_resampled = ros.fit_resample(X_train_combined, y_train)

      # handle imbalanced dataset
      smote = SMOTE(random_state=42)
      X_train_resampled, y_train_resampled = smote.fit_resample(X_train_combined, y_train)
```

Model Training and Save Model

- Random Forest Classifier

```
# RandomForest Classifier
rf = RandomForestClassifier(
    n_estimators=200,
    max_depth=10,
    random_state=42
)
rf.fit(X_train_resampled, y_train_resampled)
joblib.dump(rf, path+'random_forest_model.pkl')
```

- LightGBM Classifier

```
# LightGBM Classifier
lgbm = LGBMClassifier(
    n_estimators=200,
    min_child_samples=20,
    class_weight='balanced',
    max_depth=10,
    learning_rate=0.3,
    random_state=42
)
lgbm.fit(X_train_resampled, y_train_resampled)
joblib.dump(lgbm, path+'lightgbm_model.pkl')
```

- SVC Classifier

```
# SVC
svc = LinearSVC(
    C=10,
    max_iter=200,
    loss='hinge',
    random_state=42
)
svc.fit(X_train_resampled, y_train_resampled)
joblib.dump(svc, path+'svc_model.pkl')
```

Load Model and Model Evaluation

```
[141] # Load the saved models
rf_loaded = joblib.load(path+'random_forest_model.pkl')
lgbm_loaded = joblib.load(path+'lightgbm_model.pkl')
svc_loaded = joblib.load(path+'svc_model.pkl')
vectorizer = joblib.load(path+'tfidf_vectorizer.pkl')
```

```
models = {
    'RandomForest Classifier': rf_loaded,
    'LightGBM Classifier': lgbm_loaded,
    'SVC': svc_loaded
}

accuracy_scores = []

for name, model in models.items():
    y_pred = model.predict(X_test_combined)
    accuracy = accuracy_score(y_test, y_pred)
    print("\n")
    print(f"For {name}:")
    print(f"Accuracy: {accuracy:.4f}")

    labels = [str(label) for label in label_encoder.classes_]
    conf_matrix = confusion_matrix(y_test, y_pred)
    print("Confusion Matrix:")
    print(conf_matrix)
    print("Classification Report:")
    print(classification_report(y_test, y_pred, target_names=labels))

    accuracy_scores.append(accuracy)
```

- Random Forest Classifier

```
For RandomForest Classifier:
Accuracy: 0.8259
Confusion Matrix:
[[ 467  514   41]
 [ 867 5891   27]
 [   1   35  689]]
Classification Report:
              precision    recall  f1-score   support

     neutral       0.35      0.46      0.40      1022
   satisfied       0.91      0.87      0.89      6785
 unsatisfied       0.91      0.95      0.93       725

    accuracy                           0.83      8532
   macro avg       0.72      0.76      0.74      8532
weighted avg       0.85      0.83      0.83      8532
```

- LightGBM Classifier

```
For LightGBM Classifier:
Accuracy: 0.8801
Confusion Matrix:
[[ 413  609    0]
 [ 414 6371    0]
 [   0    0  725]]
Classification Report:
              precision    recall  f1-score   support

     neutral       0.50      0.40      0.45      1022
   satisfied       0.91      0.94      0.93      6785
 unsatisfied       1.00      1.00      1.00       725

    accuracy                           0.88      8532
   macro avg       0.80      0.78      0.79      8532
weighted avg       0.87      0.88      0.87      8532
```

- SVC Classifier

```
For SVC:
Accuracy: 0.6301
Confusion Matrix:
[[ 279  318  425]
 [ 556 4376 1853]
 [   0    4  721]]
Classification Report:
              precision    recall  f1-score   support

     neutral       0.33      0.27      0.30      1022
   satisfied       0.93      0.64      0.76      6785
 unsatisfied       0.24      0.99      0.39       725

    accuracy                           0.63      8532
   macro avg       0.50      0.64      0.48      8532
weighted avg       0.80      0.63      0.68      8532
```

- Summary: Based on accuracy, the LightGBM Classifier performed the best among the three models. However, the precision and recall were the weakest for the "neutral" class across all models. This is likely due to the nature of neutral review texts, which often contain both positive and negative sentiments, making it challenging for the models to extract meaningful features and predict the correct label. Notably, the LightGBM Classifier achieved a precision of 1.0, indicating potential overfitting. Therefore, the Random Forest Classifier may be a more balanced alternative for prediction.