Project Name: Restaurant Sentiment Analysis

Author: Hiranmayi Palanki (palanki2@illinois.edu)

Team: Solo, worked independently on the course project

The course project report provides insight into the project overview & purpose, the software functionality, software implementation, code structure, usage, future improvements, and conclusion.

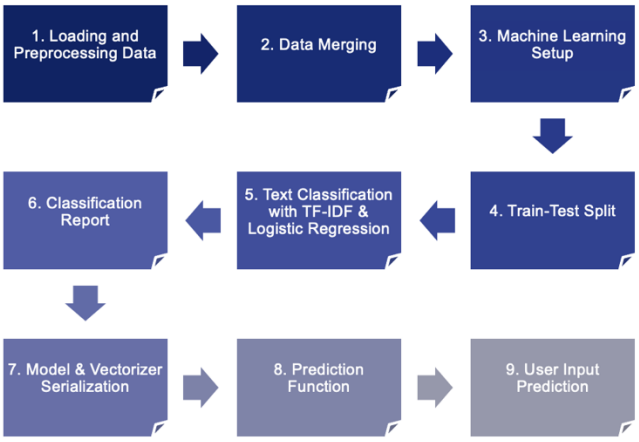
I. **Project Overview**: The objective of this project is to build a Restaurant Review Sentiment Analysis system using a Machine Learning Model. It involves sentiment analysis on restaurant reviews from two different sources (Google and TripAdvisor). The goal is to predict the sentiment of a given restaurant review as either Negative, Neutral, or Positive, based on the text of the review.

II. **Project Purpose**: The project's main purpose is to complement the traditional rating system used for restaurant reviews with sentiment analysis. Here are the key reasons for integrating sentiment analysis alongside a rating system:

- Deeper Insights: Sentiment analysis provides a more detailed understanding of customer feedback by categorizing reviews as positive, negative, or neutral. This allows for deeper insights into the reasons behind the ratings.

- Filtering Ambiguity: Rating systems can be misleading when a customer gives a low score due to disliking one aspect of the service but liking other aspects. Sentiment analysis helps filter such cases more effectively.

III. Overview of the Software Functionality: The overall functionality of the software is summarized via a nine-step process as depicted in the diagram here.



1. Loading and Preprocessing data

Two datasets are used in this project:

- TripAdvisor Reviews
- Google Reviews

The datasets are loaded using Pandas for further analysis. Data preprocessing includes handling missing values. Any rows with missing data are dropped from both datasets to ensure the quality of the data used for analysis. Summary statistics are generated for both the TripAdvisor and Google Review datasets. This helps provide a quick overview of key metrics such as count, mean, standard deviation, minimum, and maximum for numerical columns in the datasets. Data visualization is performed to gain a better understanding of the datasets and extract meaningful insights. Histograms with KDE (Kernel Density Estimation) plots are used to visualize the distribution of review lengths for both datasets. This helps understand the distribution of review text lengths.

2. Data Merging

The TripAdvisor and Google Review datasets are merged into a single dataset to create a comprehensive dataset for sentiment analysis.

3. Machine learning Setup

The machine learning setup involves selection of a subset of columns and defining the rating categories for sentiment analysis.

4. Train-test Split

The combined dataset is split into training and testing sets. The training set contains 80% of the data, while the testing set contains 20%.

5. Text Classification with TF-IDF and Logistic Regression

Sentiment analysis is performed using a Multinomial Logistic Regression model. The following steps are taken:

- TF-IDF Vectorization: Text data is transformed into numerical vectors using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. A TF-IDF vectorizer is initialized and fitted on the training data.

- Model Training: A Multinomial Logistic Regression model is trained using the TF-IDF vectors of the training data and the corresponding sentiment labels (Negative, Neutral, Positive).

- Model Evaluation: The model is evaluated on the testing data, and accuracy metrics are computed to assess its performance.

6. Classification Report

The model's performance is evaluated using accuracy metrics and a classification report. The classification report includes precision, recall, F1-score, and support for each sentiment category (Negative, Neutral, Positive).

The accuracy of the model is found to be approximately 86%.

7. Model & Vectorizer Serialization

The trained TF-IDF vectorizer and Multinomial Logistic Regression model are saved to files using the pickle library for future use.

8. Prediction Function

A prediction function is defined to predict the sentiment category of a user's input restaurant review. This function uses the saved model and TF-IDF vectorizer to preprocess the input text and make predictions. It returns the predicted sentiment category (Negative, Neutral, Positive) along with probabilities for each category.

9. User Input Prediction

Sample predictions are provided to demonstrate the functionality of the prediction function. The predicted sentiment category and probabilities are shown for three different user input restaurant reviews.

Review 1:

Text: The food was terrible, and the service was slow.
Predicted Category: Negative
Probabilities:
Negative: 0.9807
Neutral: 0.0186
Positive: 0.0008

Review 2:

Text: The food was very good, and the service was excellent. Worth going to that restaurant.
Predicted Category: Positive
Probabilities:
Negative: 0.0019
Neutral: 0.0049

Positive: 0.9932

Review 3:

Text: The food was ok.
Predicted Category: Neutral
Probabilities:
Negative: 0.0713
Neutral: 0.7035
Positive: 0.2252

IV. Functionality of Code: The overall functionality of the code is elaborated in the section below.

1. **Loading and Preprocessing Data**

    - The code reads two datasets (`**google_df**` and `**tripadvisor_df**`) related to restaurant reviews.
    - Missing values are removed from both datasets.
    - Summary statistics and visualizations are created to understand the distribution of ratings in both datasets.
    - Word clouds are generated to visualize common words in the reviews.

2. **Data Merging**

    - The code merges the two datasets (`**google_df**` and `**tripadvisor_df**`) into a single DataFrame named **combined_df.** This combines reviews from both sources.

3. **Machine Learning Setup**

    - A subset of columns ('**Review**' and '**Rating**') is selected to create a new DataFrame named `**ml_data**`.
    - Rating categories ('Negative', 'Neutral', 'Positive') are defined based on specified bins and labels.

4. **Train-Test Split**

    - The `**train_test_split**` function is used to split the `**ml_data**` DataFrame into training and testing sets (80% training, 20% testing).

5. **Text Classification with TF-IDF and Logistic Regression**

    - TF-IDF vectorization is applied to the review texts.
    - A Multinomial Logistic Regression model is trained on the training set.
    - The model is evaluated on the testing set, and accuracy is printed.

6. **Classification Report**

    - The `**classification_report**` function is used to generate a detailed classification report, including precision, recall, and F1-score for each class.

7. **Model and Vectorizer Serialization**

    - The trained TF-IDF vectorizer and Logistic Regression model are saved to files using the `**pickle**` module.

8. **Prediction Function**

    - A function named `**predict_rating_category**` is defined for making predictions on new user input.
    - The function loads the pre-trained model and vectorizer (if not provided) and predicts the rating category for a given user review.

9. **User Input Prediction**

    - An example user review is provided, and the `**predict_rating_category**` function is used to predict the rating category and display probabilities.

V. **Software Implementation:** The section below provides comprehensive information about the implementation, functionality, and usage of the code.

Code Structure:

1. **Data Loading and Preprocessing**

    - Libraries Used:
        - Pandas (for reading files and handling data)
        - NumPy (for data manipulation and related operations)
        - Matplotlib (for plotting graphs and data visualization)
        - Seaborn (for enhanced data visualization)

- Functionality:
    - Reads and cleans restaurant review datasets from Google and TripAdvisor.
    - Checks for missing values and removes them.
    - Generates summary statistics and visualizations of the review ratings.
    - Creates word clouds to visualize common words in the reviews.

2. **Data Merging**

- Functionality:
    - Merges Google and TripAdvisor datasets into a single DataFrame (`**combined_df**`).
    - Combines reviews from both sources for a unified analysis.

3. **Machine Learning Setup**

- Libraries Used:
    - Scikit-learn
- Functionality:
    - Selects relevant columns to create a DataFrame for machine learning (`**ml_data**`).
    - Defines rating categories ('Negative', 'Neutral', 'Positive') based on specified bins and labels.

4. **Train-Test Split**

- Libraries Used:
    - Scikit-learn
- Functionality:
    - Splits the `**ml_data**` DataFrame into training and testing sets (80% training, 20% testing).

5. **Text Classification with TF-IDF and Logistic Regression**

- Libraries Used:
    - Scikit-learn
- Functionality:

- Applies TF-IDF vectorization to the review texts.
- Trains a Multinomial Logistic Regression model on the training set.
- Evaluates the model on the testing set and prints accuracy.

6. **Model Serialization**

   - Libraries Used:
     - Pickle
   - Functionality:
     - Saves the trained TF-IDF vectorizer and Logistic Regression model to files.

7. **Prediction Function**

   - Libraries Used:
     - Pickle
   - Functionality:
     - Defines a function (`**predict_rating_category**`) for predicting sentiment category based on user input.
     - Loads pre-trained model and vectorizer (if not provided).
     - Applies TF-IDF to user input and predicts the rating category.
     - Returns the predicted category and probabilities.

8. **User Input Prediction**

   - Functionality:
     - Provides an example user review and uses the `**predict_rating_category**` function to make predictions.
     - Prints the predicted rating category and probabilities.

Code Usage

1. Clone the repository.

2. Run the Jupyter Notebook (`**Sentiment_Analysis.ipynb**`) or integrate the provided code into your own Python environment.

3. Follow the steps in the notebook to understand the code execution and visualize the results.

4. For future predictions, use the `**predict_rating_category**` function with new user input.

Future Improvements

I'll consider using more advanced models or fine-tuning hyperparameters for better performance. I'll explore additional features or sentiment analysis techniques for improvement.

VI. **Documentation of the usage of the software**: The section below provides instructions on how to run the provided code. This documentation provides users with clear instructions on installing the required dependencies, running the software using either Jupyter Notebook or a Python script, and utilizing the prediction function.

Requirements:

- Python 3.x
- Jupyter Notebook (optional, if running the provided notebook)
- Required Python packages (install using `**pip install package_name**`):
  - pandas
  - numpy
  - matplotlib
  - seaborn
  - scikit-learn
  - wordcloud
  - pickle

Installation:

1. Clone the repository:

```
git clone https://github.com/palankihiran/CourseProject.git
cd CourseProject
```

2. Install required packages

```
pip install pandas numpy matplotlib seaborn scikit-learn wordcloud
```

Running the Software:

Jupyter Notebook:

1. Open Jupyter Notebook
2. Navigate to the `**Sentiment_Analysis.ipynb**` notebook.

3. Run the notebook cell by cell to observe the step-by-step execution.

Python Script (for direct execution):

1. Open a terminal or command prompt.

2. Navigate to the project directory:

```
cd /path/to/CourseProject
```

3. Run the script:

```
python sentiment_analysis_script.py
```

Using the Prediction Function:

The software provides a function (`**predict_rating_category**`) for predicting sentiment categories based on user input.

1. Open a Python interpreter or a script.

2. Import the function:

```
from sentiment_analysis_script import predict_rating_category
```

3. Use the function with a user input:

```
user_input = "The food was excellent, and the service was prompt."
predicted_category, probabilities = predict_rating_category(user_input)
print(f"Predicted Rating Category: {predicted_category}")
print("Probabilities:")
for category, prob in probabilities.items():
    print(f"{category}: {prob:.4f}")
```

VII. **Conclusion**: The Restaurant Review Sentiment Analysis project successfully demonstrates the implementation of a machine learning model to predict the sentiment of restaurant reviews. By combining data from TripAdvisor and Google Reviews, preprocessing, and training a Multinomial Logistic Regression model, the project provides a valuable tool for restaurant owners and stakeholders to gain deeper insights into customer feedback. The model can help identify areas for improvement and enhance the overall customer experience. Additional features such as sentiment trends over time can be incorporated by using more advanced natural language processing techniques.