CS_410_Group_ZZZRQ project documentation

Overview of the functions

This tool serves to extract prediction scores from yelp reviews by feeding in text reviews published by yelp along with other relevant fields such as location, category and time. The prediction score will be normalized to a similar format to the number of stars of an actual yelp review. The relative accuracy of predicted scores may be drawn from the plot that plots the predicted scores against their original yelp review score, which shows a good level of correlation. The data collection scripts can be used to collect a sample set of data based on certain criteria within the yelp official reviews dataset. Using the prediction scores, we can construct a word cloud that provides insight on the overall sentiment distribution within the sample set, and can be used to plot visuals from which restaurant owners, psychological researchers and other potential users may potentially use to discover useful information and trends.

Implementation Documentation:

1. Data Collector:

    a. Date related data collection:
    The main function of date data collection is find_specific_date_data(reviewdatas, start, end, res). This function is in order to find the data with a specific range date(from start date to end date). The script can also randomly choose several data by using random_choose_data(num, input_path, output_path). The num parameter in this function is the number of data we need to choose. The function combine_business_information(business_path, input_path, out_path) is ordered to collect the information from a business data file by using business_id.

    b. Restaurant related data collection:
    Combine the restaurant and business json together and then filter 6 json
    Every json represents a restaurant and contains 50 data,the data schema is attached at the end

    c. Location related data collection:
    Combine the restaurant and business json together and then filter 6 json
    Every json represents a location and contains 50 data ,the data schema is attached at the end

`

2. Data schema:

    user_id: yelp user's id
    business_id: yelp restaurants' or companies' id
    review_id: review's id
    date: the year/month/day date of review created
    time: the time of review created
    categories: the yelp restaurants' or companies' type
    city: the restaurant's or company's city
    state: the restaurant's or company's state
    stars: the restaurant's or company's score
    business_review_count: the restaurant's or company's reviews number
    predit_yelp_score: the predicted sentiment analysis score of review
    actual_yelp_score: the actual score of review

3. Azure sentiment analysis implementation:
   Using Text Analytics client library to implement the sentiment analysis. The function of calculate_score(input_path, out_path) is used to calculate sentiment analysis score. The output of the model is positive score, neutral score and negative score. Then we use the ***positive score * 5 + neutral score *2.5 + negative score * 0*** to calculate the predicted sentiment analysis score, which normalizes the sentiment analysis score to a predicted review score that resembles an actual yelp review score that is out of 5. add_score() can then add the predicted review score to the data dictionary.

4. Azure custom text classification model training and deployment:
   a. Collected 300 Yelp reviews with a uniform distribution of 100 five-star, 100 three-star, and 100 one-star reviews for the custom text classification machine learning model training using Data_extraction.py.
   b. The five-star reviews are labeled as positive, three-star as neutral, and one-star as negative. The model uses 80% of the labeled reviews as the training set and 20% as the testing set.
   c. The custom-trained model can be deployed and used for inference by using the Azure API with the corresponding user key, endpoint, project name, and model name.
   d. The software for classification uses python with Azure API. The software can open all the review txt files in the folder and classify all of them into three classes: positive, neutral, and negative.

5. Create the word cloud:
   The main function of this script is create_word_cloud(input_path). It will show the restaurants/companies word cloud based on the predicted sentiment analysis score.

Usage Documentation:

Restaurant/location Data Collection, located in filter.ipynb or filter.py:

getData():

The input is the path of the business json file and the path of the review json file, the output is the two lists of dict together for further use.

construct_filter():

6 parameters are used in this function; the first one is the business data, the second one is the reviews data. These two are obtained from the previous function. The third parameter is choice which will be used to justify getting location data or restaurant data. If we set the choice to be True, we will get the location json orelse we will get the restaurant jsons. The fourth and fifth parameters are num_reviews which is the number of reviews in final json files and num_output which is the number of json files. The last parameter is download, which is used to decide whether to download the output; if set to True, they will be downloaded and stored in the directory where the script is located as a json file, orelse we will get a list of dict within the return value.

Date Data Collection, located in data_collector.ipynb:

For this script, it is required to choose the specific range of date data specified by start date and end date of datetime type. In this project we chose a month during the covid-19 period: August 2020 specified by 2020-08-01 to 2020-08-31 and a month outside the covid-19 period: August 2018 specified by 2018-08-01 to 2020-08-31. Moreover, we also collect the thanksgiving holiday months data from 2018 to 2021. For using this script, we need to use two different json data from the yelp dataset. One is review_data.json and another is business_data.json. Because we need three steps to get the final results, the user needs to create three empty folders: timesplitData, choose_data and final_data. The users then need to set up the input path and output path for each function.

Azure sentiment analysis, located in implement_model.ipynb:

The user needs to set up the correct name of the dictionary key according to the data schema and the API credential (Azure API key) and the endpoint. The user will also need to set up the output path and input path. The Azure Language Studio is used for data labeling and training.

Azure.core.credentials API is needed for identification in the Python script before using the custom-trained model. Azure.ai.textanalytics API needs to be initialized with the correct project and model name.

Use Azure.ai.textanalytics.begin_single_label_classify() function for deploying and using the model for review classification. The function returns the class with a confidence score. Use calculate_score() to calculate the predicted review score and add_score() to the corresponding data file.

Word cloud implementation, located in word_cloud_implementation.ipynb:

The user needs to set up the input path. If the input has the predicted sentiment analysis score. The user can directly use the WordCloud() to create the word cloud. This script will create a tool, which can create the restaurants' or companies' names word cloud based on the predit_yelp_score. Users can use the data inside the test_data folder to test this tool.

Test Processes:

Word Cloud tool:

For word cloud tool tests, the user can use the data in the test_data folder. Use the path to change the word_cloud_impementation.ipynb file input path. Then running the code you will get the word cloud picture. Word_cloud_impementation.ipynb combines the steps to predict the score or combine the business information.

Date data collection:

For the **data_collector.ipynb** test, the user needs to download the data from yelp. (https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset). The user then needs to choose the review file and the business file and then input their paths into the **data_collector.ipynb**. Because we need three steps to get the final results, you need to create three empty folders. The first one is **timesplitData**. The second one is **choose_data**. The third one is **final_data**. Then users need to set up the input path and output path for each function. you can choose the start date and end date which is datetime type. Running the code in the **data_collector.ipynb** with a specified data storage path then will output the filtered result. Then, use this result path as the input path to the **implement_model.ipynb** will get the predicted scores. The date-related data result will be placed in the **final_data_add_predit_score** folder.

Restaurant/location related data collection:

There are two versions of filter: filter.py which is based on function calls and filter.ipyn based on jupyter cells.

For testing purposes, we will use **filter.ipynb:**

For Restaurant/location tool tests, the user will need to download the data from yelp. (https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset). The user will need to put the review file and the business file in the same folder with the ipynb file. The user will then import json and collections files and then run the cells line by line. Once the entire file finishes running, the output results will be located in the same directory.

The user can change the choice parameter to get the location json (choice = True) or the restaurant json (choice = False) and set the num_reviews to change the number of reviews or the num_output change the num of output json. Besides, the download parameter may be used to determine if the output will be in the form of return variable or output json file.

Data analysis and visualization application:

The output json files from data collection may be fed into Tableau or any other data visualization software for further analysis. A combined json file that is the merged version of all sample test data collected, called "combined.json" can be used as input to Tableau. Using Tableau, we can validate the effectiveness of the predicted review scores by plotting the predicted review scores against the actual yelp review scores to visualize the strength of correlation. Other plots such as average review scores per month during Covid-19, average review scores per hour of the day and average review scores disaggregated according to restaurant types can be used to visualize and discover various interesting trends among the data.

The demo-version of Tableau visualization may be found in this Tableau Public link:
https://public.tableau.com/app/profile/zetian.li/viz/CS410_Project_Final/PredictedVSActualScores

Team members’ contribution:

Xinhe Xu: Data collection, Data Cleaning, Azure sentiment analysis implementation to make prediction, and word cloud tool implementation

Zhichao Fan: Date Collection on location and restaurant , Data Cleaning ,Data Combination on review json and business json

Rick Wang: Training data labeling, Azure Text Classification Model training, and Python script for batch text classification using Azure AI API.

Zetian Li: Training data collection, Sentiment Analysis review score design, Data Curation before feeding into Tableau, Analysis and Visualization using Tableau and overall logistics.

Qiwei He: Review data visualization and analysis, Azure sentiment analysis implementation and classification Model training.

Citation:

https://learn.microsoft.com/en-us/azure/cognitive-services/language-service/custom-text-classification/overview

https://github.com/Azure/azure-sdk-for-python/tree/main/sdk/textanalytics/azure-ai-textanalytics/samples

https://learn.microsoft.com/en-us/python/api/overview/azure/ai-textanalytics-readme?view=azure-python-preview&preserve-view=true

https://towardsdatascience.com/how-to-make-word-clouds-in-python-that-dont-suck-86518cdcb61f

https://towardsdatascience.com/text-mining-and-sentiment-analysis-for-yelp-reviews-of-a-burger-chain-6d3bcfcab17b

Data source: https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset