Assignment 1 Report

Working with Edgar datasets: Wrangling, Pre-processing and exploratory data analysis

INFO 7390
Advanced Data Science & Architecture

Professor:
Srikanth Krishnamurthy

Stduents:
Team 2
Chenlian Xu
Qianli Ma

Problem 1: Data wrangling Edgar data from text files
The goal of this exercise is to extract tables from 10Q filings using Python.

Part 1: Parse files
1. Initializing the log file.

2. The first step is to get the parameters from the user end like CIK, Accession Number, S3 Access Key, etc. Program simply prints requirements like “Please input the CIK” in the terminal to take the values one by one by users’ inputting.

3. Program assigns the values to local variables as the parameters which will be used later. If CIK or Accession Number is not provided, program will assign default values of IBM's quarterly financial filing (form 10-Q) in October 2013 to local variables. Location will be “Default” if the provided value is not in the AWS S3 Location list.

4. Program validates Amazon AWS credentials by connecting to AWS using the access and secret keys provided by the user. However, the only way to verify AWS credentials is to actually use them to sign a request and see if it works. Otherwise, program creates the ec2 connection object anyway and will not catch an exception. Thus, program needs to actually send a request which will not return a huge amount of data. “get_all_regions()” was used here for validating. Program will exit if the Amazon keys are invalid.

5. After validating the Amazon keys, program will handle the URL of 10Q fillings with provided parameters. Program simply joint the domain ("http://www.sec.gov/Archives/edgar/data/"), CIK, Accession Number and postfix ("-index.html") together to form the FTP file path according to the instruction of EDGAR Database.

6. Once the URL of FTP file is open, program will search every <a> tags and their href attributes with BeatifulSoup lib, and check if “10q.htm” is in the tag. Once the program find out the link of 10Q fillings, it will end the loop and keep the link.

7. Once the URL of 10Q fillings is open, program will fetch all the table with statistic data. These tables are all in the <div><table> tags according to the page elements.

8. Program stores all the tables it finds in the last table in a container (which is a list). Then it goes through the list to extract the tables into csv files.

9. There are some unreadable symbols in the output csv files. Thus, the program need to encode the tables it parsed from the website to utf-8 and replace some of the symbols like dash and white space.

10. Program zips the csv files it outputted in the last step and the log files together then uploads it to Amazon AWS S3. The bucket is named as AWS access key plus date and time to ensure it is unique.

Part 2: Build Docker

1. Create Dockerfile

```
FROM continuumio/anaconda3
ADD Problem1.py /
CMD [ "python", "./Problem1.py" ]
```

2. Docker image build command

```
Docker build –t team2/problem1
```

3. Running the docker image

```
Docker run team2/problem1 python Problem1.py
```

Problem 2: Missing Data Analysis
The goal of this exercise is to analyze the EDGAR Log File Data Set.

Part 1: Data Analysis
1. initialize the log file.

2. Program takes the parameter from the user. The implementation is pretty much the same as the step 2 in Problem 1. Here the year of the data would be needed and it must be ranged from 2003 to 2017. If an invalid year is provided, the program will exit.

3. Validate the Amazon account. The implementation is same as the part in Problem 1.

4. Cleaned up required directory

5. Generate the URL for download the zip file. The URL is formed with domain and key-value pairs in month quarter dict.

6. Download the zip file and unzip file. If the file does not exist, the program will log a warning.

7. Read all unzipped csv files and put them in a DataFrame structure.

8. Detect the anomalies and clean the data set. By doing this, the program counts the null values, check if there is any incorrect value instead of 0 and 1 in idx, norefer and noagent fields.

9. handle the missing data. The program will delete the rows which have missing value in CIK, accession number, ip, date and time columns. For the columns of idx, browser, code, find, extension, zone, the program will fill the missing value with the most used data of that column; For the columns of norefer, noagent, replace missing values

with 1; For the column of crawler, replace the missing values with 0; For the column of size, replace the missing value with average value.

10. for each month log file, compute the summary matrix

11. combine all the individual DataFrame into one DataFrame and export a csv file.

12. zip the csv file

13. create a new amazon bucket and upload all files

Part 2: Build Docker

1. Create Dockerfile

```
FROM continuumio/anaconda3
ADD Problem2.py /
CMD [ "python", "./Problem2.py" ]
```

2. Docker image build command

```
Docker build –t team2/problem2
```

3. Running the docker image

```
Docker run team2/problem2 python Problem2.py
```

Handle exception

If the user input wrong Amazon access key and accession secret key, the program will exit at the point when create the amazon s3 bucket leaving the logging information ’Amazon keys are invalid’.

If the user input the year excess the range year, the program will exit with a logging warning ’Invalid year, please enter a valid year between 2003 and 2017’

If the data set in a month’s first day is empty, the program will give a log warning ‘Log file is empty ’.