|  |
| --- |
|  |
| Assignment 1 |
|  |
| Working with Edgar datasets: Wrangling, Pre-processing and exploratory data analysis |

**Northeastern University: ProfEssor Krishnamurthy**

October 6, 2017

Authored by: Madhumathi Prakash and Sonali

Problem 1

Parsing Data

# The Data Set

In this problem we are using the EDGAR (Electronic Data Gathering, Analysis, and Retrieval) system. It contains company’s performs automated collection, validation, indexing, acceptance, and forwarding of submissions.

# The Problem

We are accessing data from Edgar site. Given a CIK number and the accession number, we are creating a url to get the data but replacing the X, Y, and Z in the following url: [http://www.sec.gov/Archives/edgar/data/XXX/ZZZ/YYY- index.html](http://www.sec.gov/Archives/edgar/data/XXX/ZZZ/YYY-%20index.html) . From this url we are locating the 10Q file html link and then extracting all the tables. Lastly, we are loading all the tables into a CSV file. When looking for the tables in the 10Q files, we should also deal with logging all activities. For instance, we should deal with error if there is an invalid CIK or accession number or if the amazon keys aren’t valid.

We are using Docker for this problem and building a docker image which will automate the task. Put all zip file on Amazon S3.

# Our Approach

1. We first took a specific use case of the IBM url and worked with it: ([http://www.sec.gov/Archives/edgar/data/51143/000005114313000007/0000051143-13-000007- index.html](http://www.sec.gov/Archives/edgar/data/51143/000005114313000007/0000051143-13-000007-%20index.html)) and parsed the url and got the specific 10-Q file.
2. To make the url general: [http://www.sec.gov/Archives/edgar/data/CIK/ZZZ/access number- index.html](http://www.sec.gov/Archives/edgar/data/CIK/ZZZ/access%20number-%20index.html)
   1. Through console, user will input a CIK and Accession number
   2. To deal with “ZZZ” we realized that it was the access number without the “-“ so we removed it.
   3. We then concatenated URL with these items.
3. In terms of error logging, we are dealing with the use case of if user does not provide CIK or access number, then we will provide an logging.info message in the error log saying that, because CIK or access number was not given, we will be assigning a default CIK and access number of IBM.
   1. We also deal with the cease if there is invalid url or 10-Q url and these cases will show Warning message in log file and exit.
   2. We also clear the log file every time the code is run.
   3. We print out warning messages to user incase something goes wrong from their side.
4. After looking at the specific 10-Q file and generating url, we generalized by using the cik and access number, similar to the main url.
   1. We have also dealt with the case where there might be multiple 10-Q files and created a loop. This way we will be reading each 10-Q file.
5. We used BeautifulSoup in dealing with the 10-Q file to get the tables and format it.
   1. We first grabbed all tables in the 10-Q file by finding the div table tags in the html.
   2. We then looked at the html page and noticed all the tables which have financial data (numerical data) had tables with % and $ signs. When looking at the htm markup, we noticed that there are tables that are really just text and those are ones we do not want.
   3. So we looped through the table and for each table we are looking at the row and certain cell to see if the table has $ or % and if and only if a table had either, we added the table to table.append.
   4. Next step was to clean up and retrieve text from the cells and make a table from it. Then put the table in a CSV. So each table had its own CSV.
6. Using Amazon S3
   1. At the beginning we use secret key and access key to connect to the amazon s3.
      1. Make sure to enter your specific access key and secret key for your Amazon S3 into the console.
      2. We are using boto.s3 and setting the location to default for the bucket.

# Steps to Run the code

Problem 2

Missing Data

# The Dataset/ Our Problem

The EDGAR Log File Data Set [https://www.sec.gov/data/edgar-log-filedata-set.html ], which has listed log files with data month wise for every year. Given a year, we will be grabbing data for 1st day of every month by generating a url.

From this csv, we should deal with missing data and analyze the data with data summaries.

# Our Approach

1. We got the year from console and used that to generate a url.
   1. We first looped through and broke it by Quarter, as each quarter has a different url. Then appended the month and day 01 to the end right before ,zip.
   2. Then because the files come in zipped, we unzipped it and got rid of the zipped folder so only the csv remains.
2. We read each csv into a data frame.
3. We then go through each data frame and deal with missing data for each column.
   1. Since we cannot replace CIK, IP, or Access number by most used or the mean as that would make data invalid, we dropped the rows that have NAN for these columns.
   2. For the file size column, we replace the cells that have NAN with the mean files size.

**if ip address is nan or missing then just replace with unavailable - done**

**for NAN is size replace with mean - done**

**If date is missing then replace with date from any of the other row**

**For time if 0 then replace with the end of day time**

**delete rows with missing cik and accession number - done**

**will do all the logging items for part 1 and 2 so far**

**browser column replace with most used browser of chrome - done**

**if extension like .txt is missing then append it with the access name.**

**if extension is missing itself then replace it by counting the most used extension and replace it by adding the access number.most used extension**

**zone can be replaced with most used zone?**

norefer or noagent is empty/NAN replace with 1 as it says in the <https://www.sec.gov/files/EDGAR_variables_FINAL.pdf>

crawler

how to zip files together

no need to regard outliers since this is usage/visiting data and

date with 00:00:00 is valid since that is start of day

General make everything float an int