Advance Data Science

Missing Data Analysis - Edgar Data

Assignment – 1 Part - 2

Report by:

Ankit Bhayani

Rajat Agrawal

Vishakha Sawant

**OBJECTIVE**

The report summarizes the design and implementation of the analysis performed on the Edger log files. This report is divided into two section.

**Section 1**: Fetching and Analysis of Edgar logs file.

**Section 2**: Handling Missing Values and compute summary metric for Edgar Log files

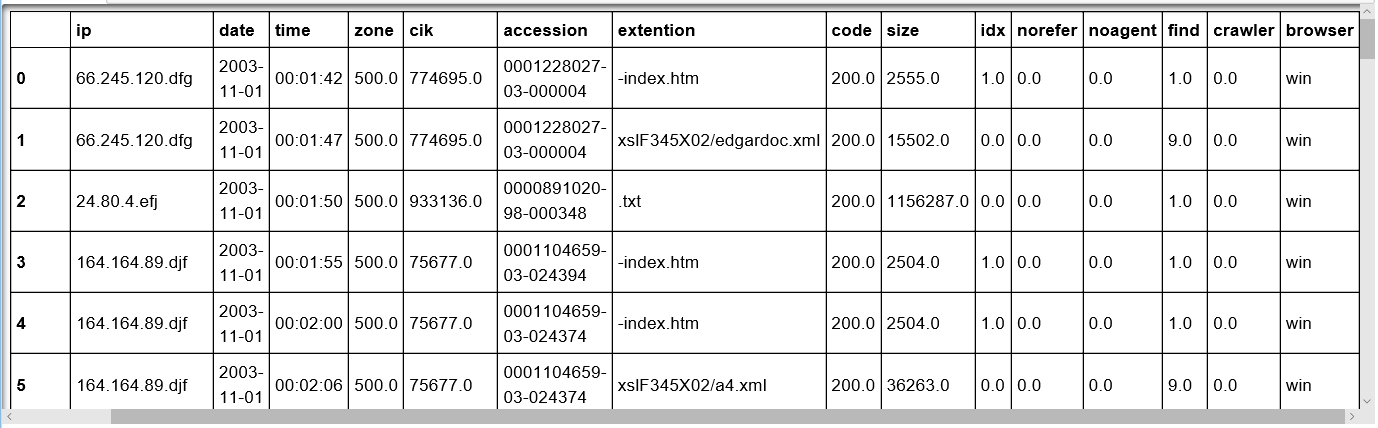
**Section 3**: Dockerize the process using Docker file and Docker Hub

**Section 4**: Create the Tableau representation of the analysis performed on the log

**SECTION - 1 ANALYSIS OF EDGAR LOG FILES**

The EDGAR Log File Data Set contains information in CSV format extracted from Apache log files that record and store user access statistics for the SEC.gov website. These log are captured on a daily basis and are stored in a zip format on Edgar website.

Edgar log files consist of the following columns:



Each column in this log file store various information which compromise of the:

**CIK**: Edgar company CIK for filing purpose

**Accession**: Accession number to access the particular file

**Extension**: consist of the file that is requested by supplying CIK and accession number

**Code**: implies the response code from the server

**Date:** log file creation file

Analyzing these log based on the description present on the variables we found that:

* There were many missing values for the column name browser and size.
* Many extensions in the files were incomplete (Without the file name).
* CIK length is greater than 10. Maximum length mentioned on the site for CIK is 10.
* For Code 304, we have all the file size mentioned a 0.

**SECTION - 2 HANDLING MISSING VALUE**

This section discusses the working of the code where for a given year, the program get data for the first day of the month and programmatically generate the url for every month in the year and process the file.

**STEP 1:**

**GET THE YEAR FROM THE USER**

The program takes ‘Year’ as the parameter from the user. As there are only logs file presented for the year 2003 to 2016, if the user provides the parameter outside the given range, the program will show error to provide the valid year.

If the year is valid, then we will fetch the url from the Edgar website which consist of all the log files associated for a particular year.

For error handling, we will check if the response of the request is 200 (URL is valid) and go to the next step to fetch the request file form the next URL



**STEP 2:**

**Download the Extracted Log file for all the 1st date of the Month**

Since we are using python 3.2 for accessing the URL, we have to import the following python library to access the Edgar website.

import urllib.request :- To open a requested URL

In the next step, we will be hitting the actual URL which consist of the log files to be analyzed achieve this task, we will be using BeautifulSoup, a python library which is very efficient to work with the HTML files.

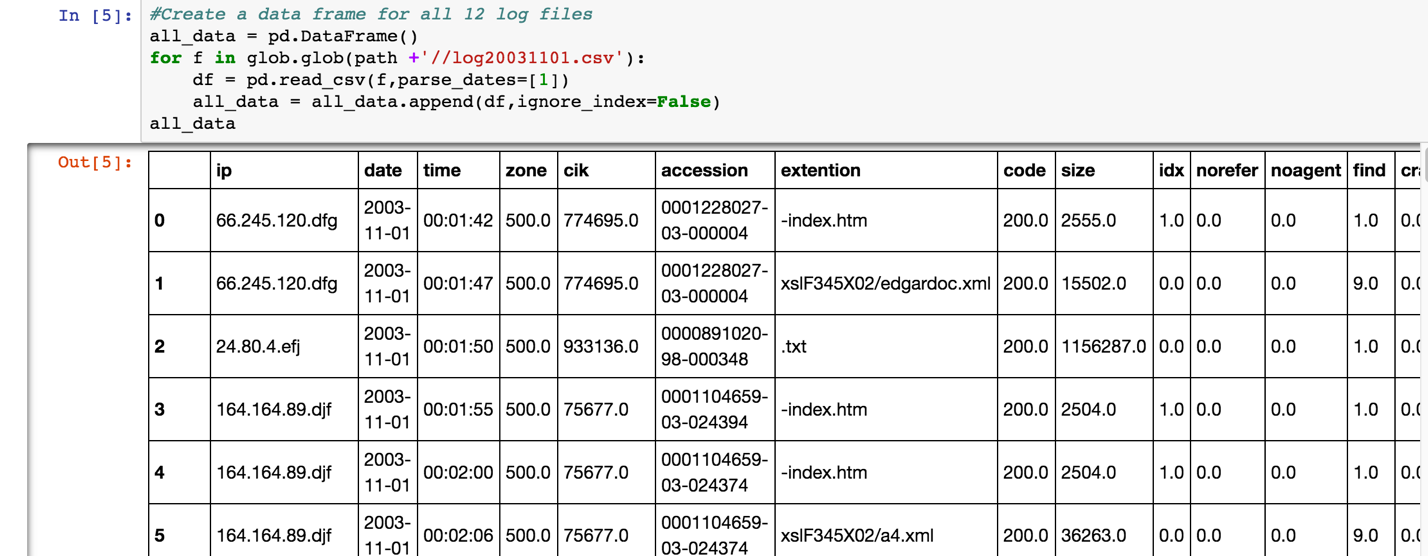
from bs4 import BeautifulSoup : - To fetch the entire HTML content from the requested file.



**Step 3:**

**Load the Log File in the Data Frame using Pandas**

After getting all the log file for a year, we will use the Python Library **Pandas** for creating a dataframe and load all the log file data into it for the analysis purpose.

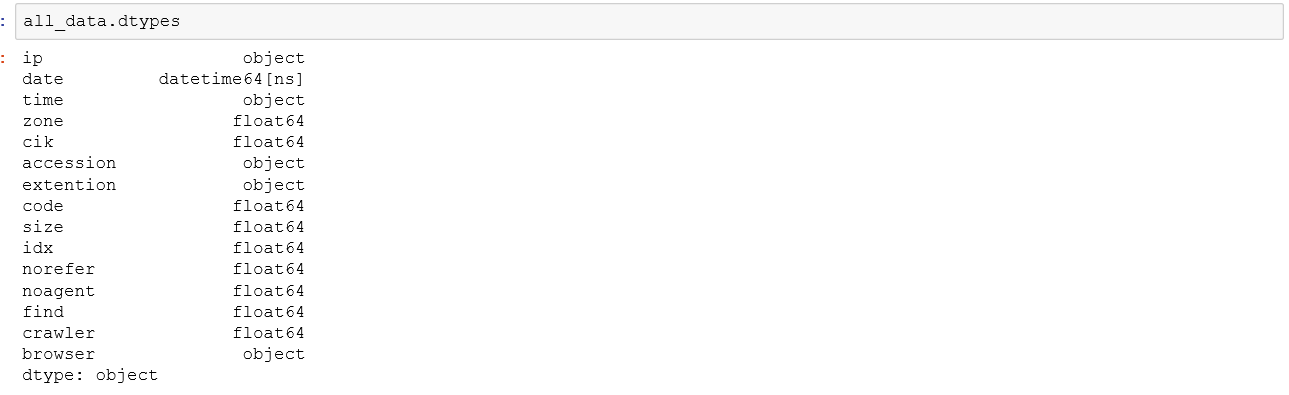


**Step 4:**

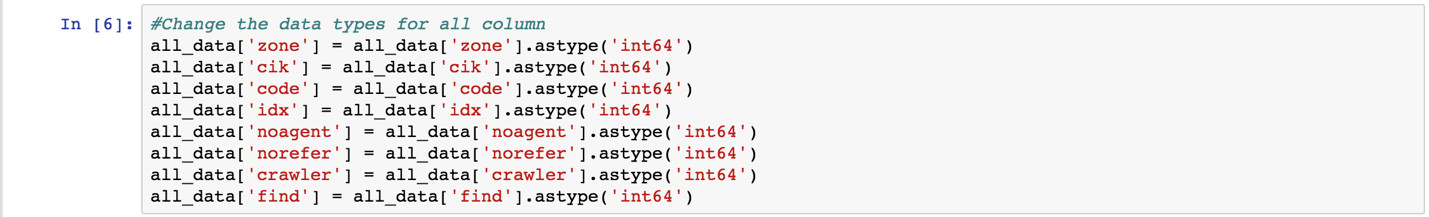
**Change the data type of the column present in the Dataframe**

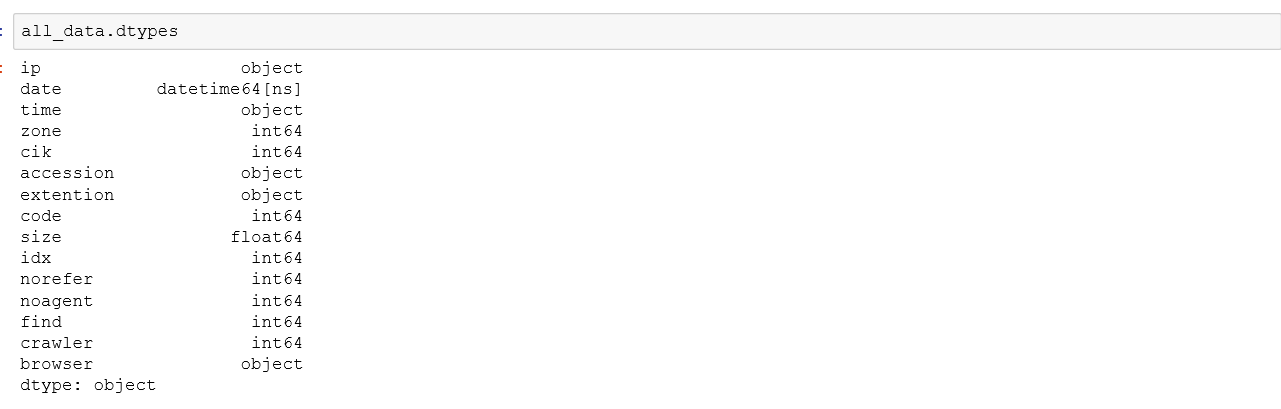
After getting all the log file for a year, we will use the Python Library **Pandas** for creating a dataframe and load all the log file data into it for the analysis purpose.

**Check the Data types:**



Now Convert the data type before manipulation:



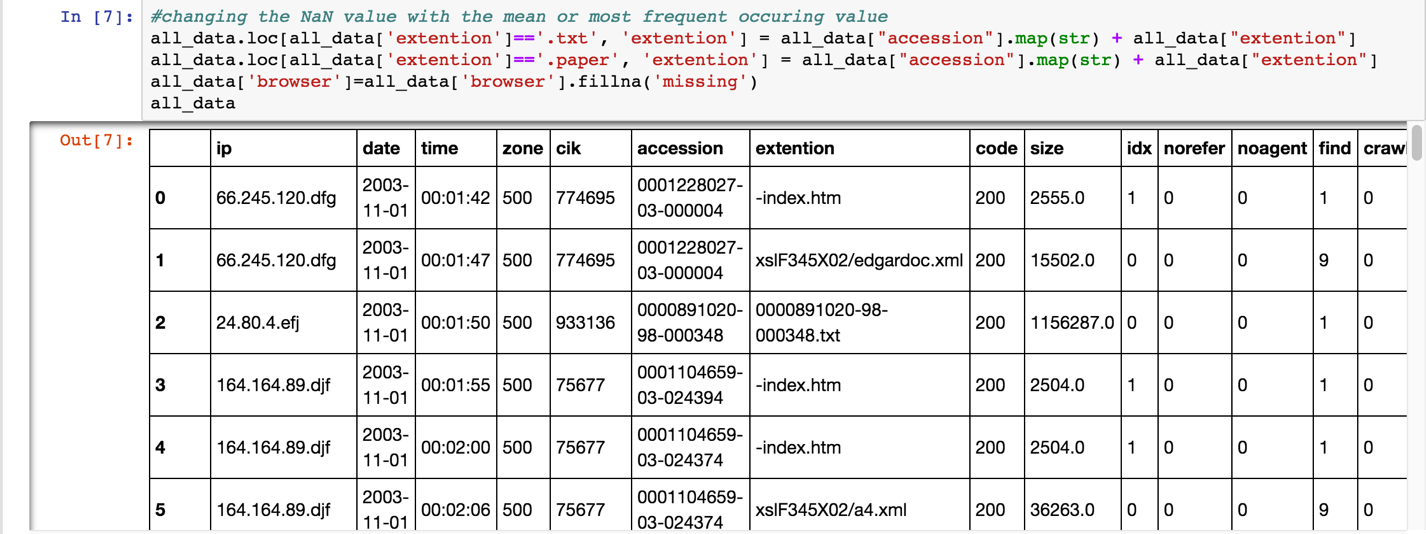


**Step 5:**

**CHANGE THE EXTENSION COLUMN WITH ACCESSION NUMBER**

As mentioned on the Edgar site *“This variable provides the filename of the file requested including the document extension. If the filename is missing and only the file extension is present, then the filename is the document accession number.”*

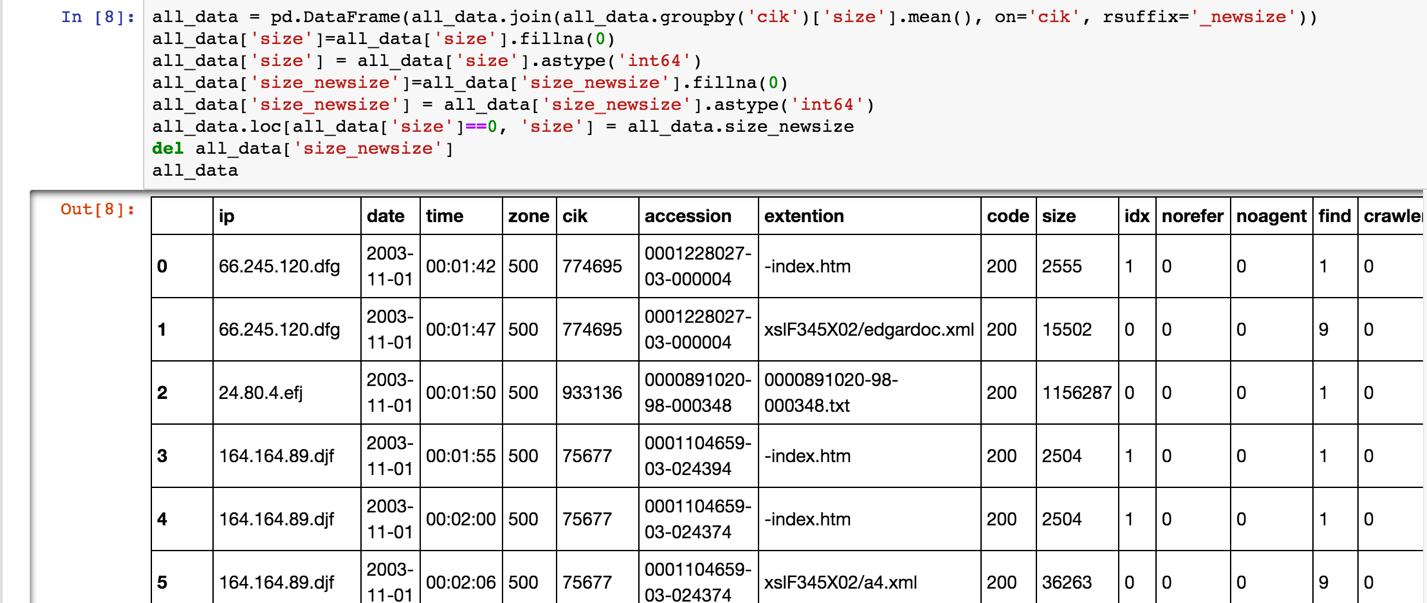
We will replace all the extension with the valid accession number by combing the accession number and the extension for the missing extension values.



**Step 6:**

**Change the Size of the accessed file having NA with Mean value**

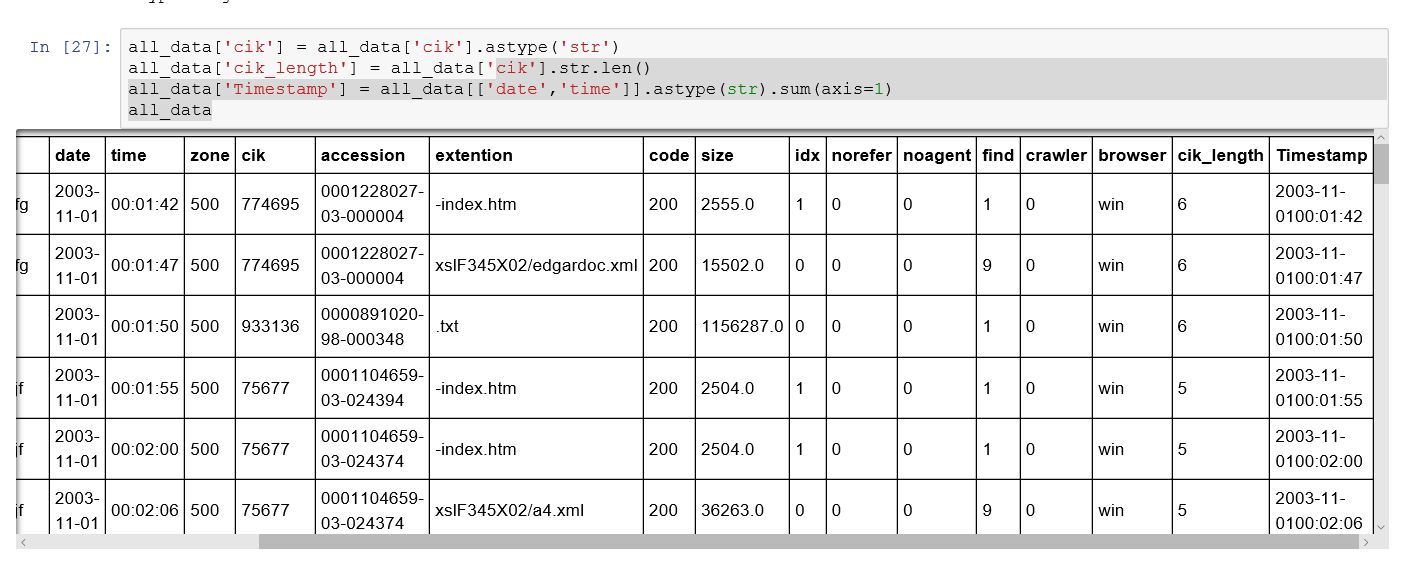
To replace the size of all the NA fields with the valid numeric value, we will group by size on the CIK and find the mean to replace the NA values.



**Step 7:**

**Create a derived column for the Timestamp and CIK length for Analysis purpose**

We will create a derived column for the timestamp which will consist of the concatenation of date and time field. Also we will calculate a CIK length for the analysis purpose.

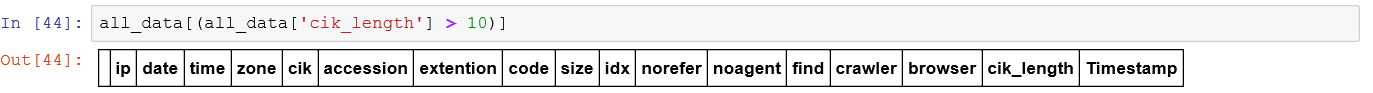


**Step 8:**

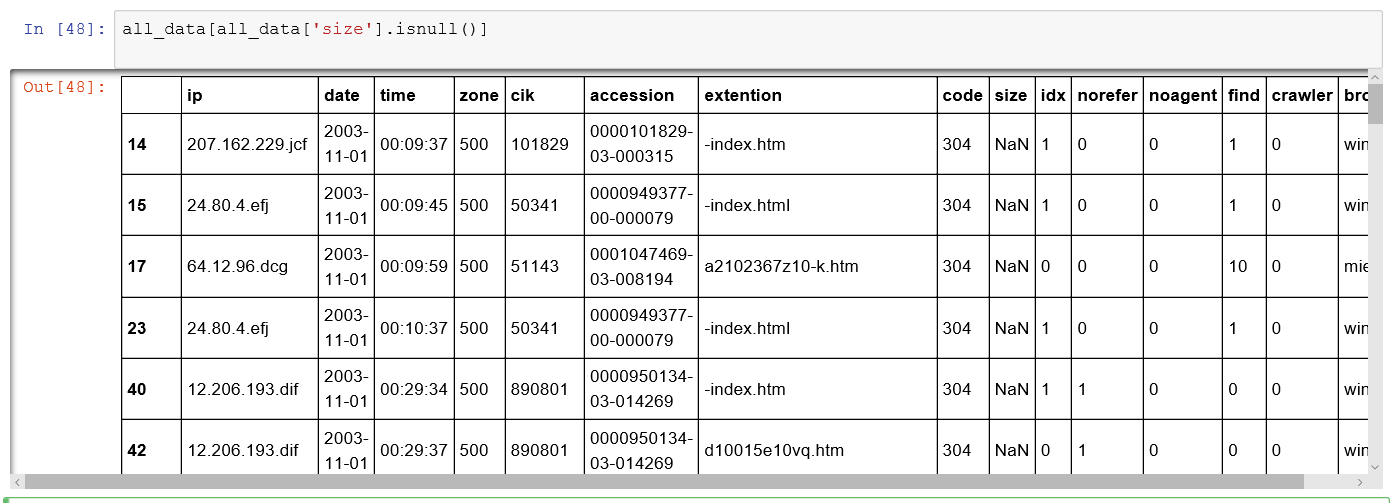
**Find the Anomalies**

We have observed following anomalies in the log files.

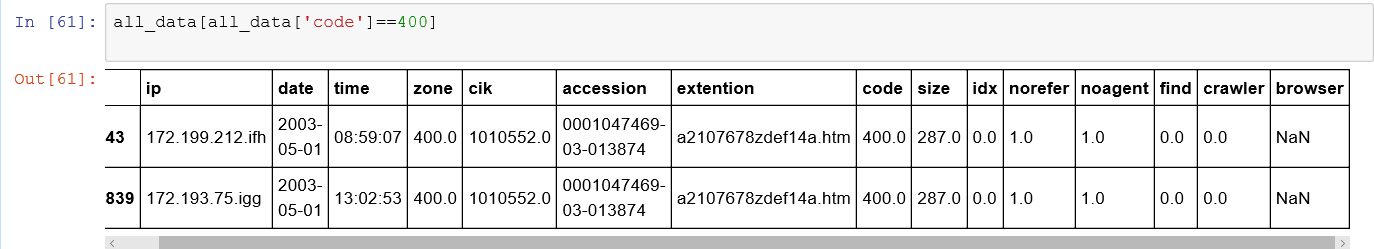
1. CIK having length more than 10.



1. For code 304, we have all the size filed as 0.



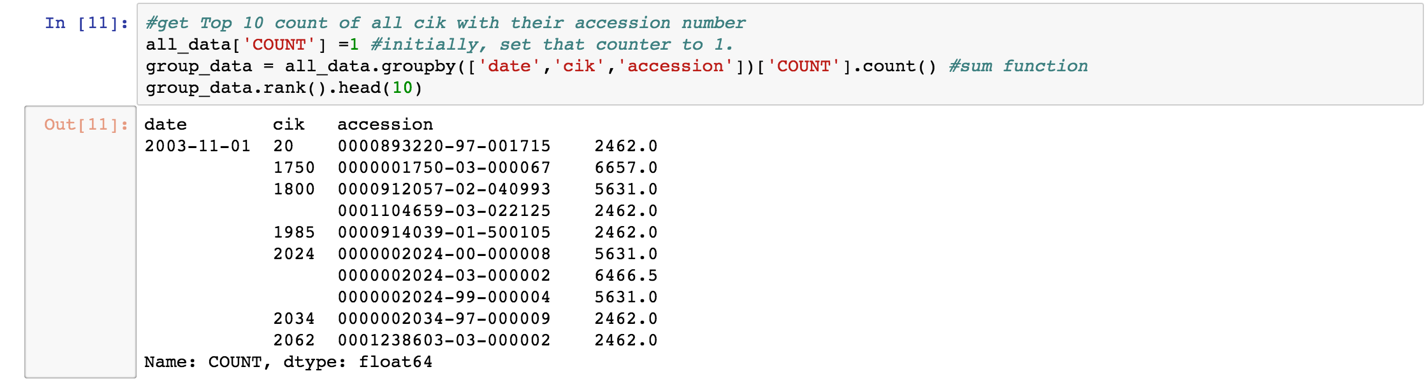
1. For Code 404, we have crawler value should be one.

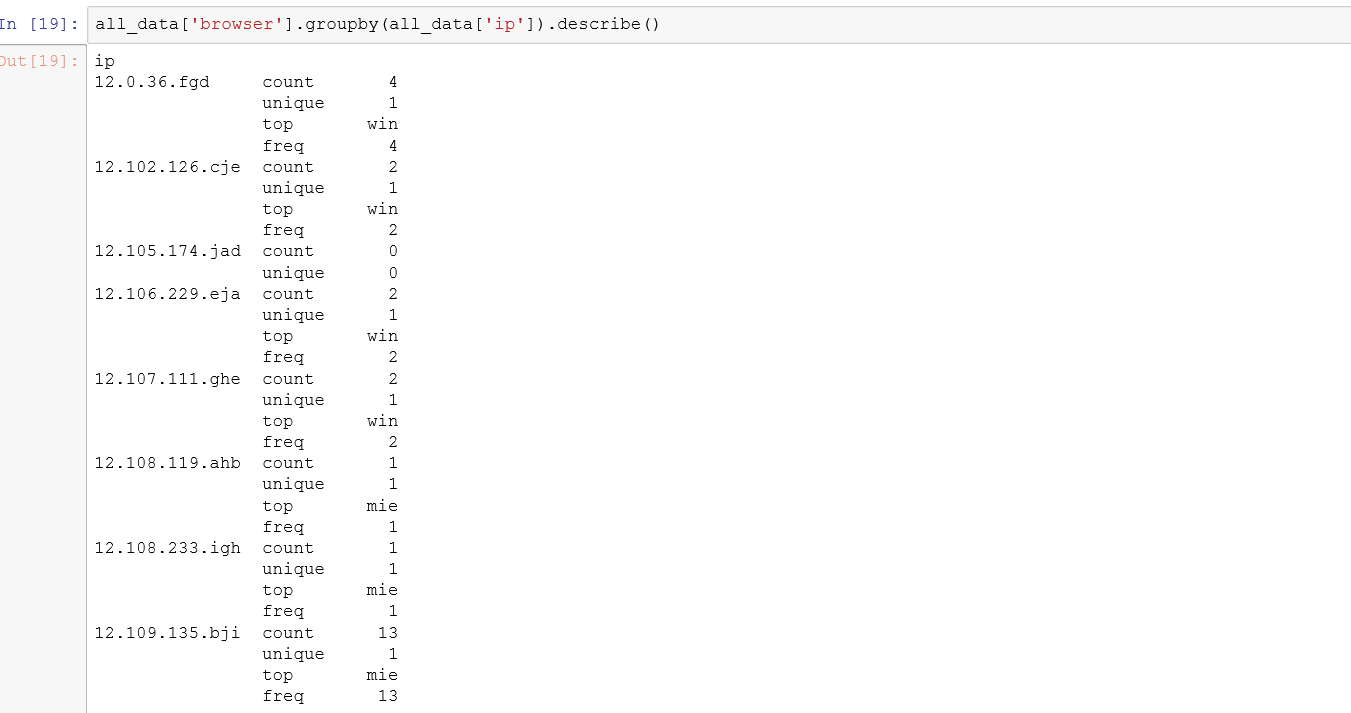


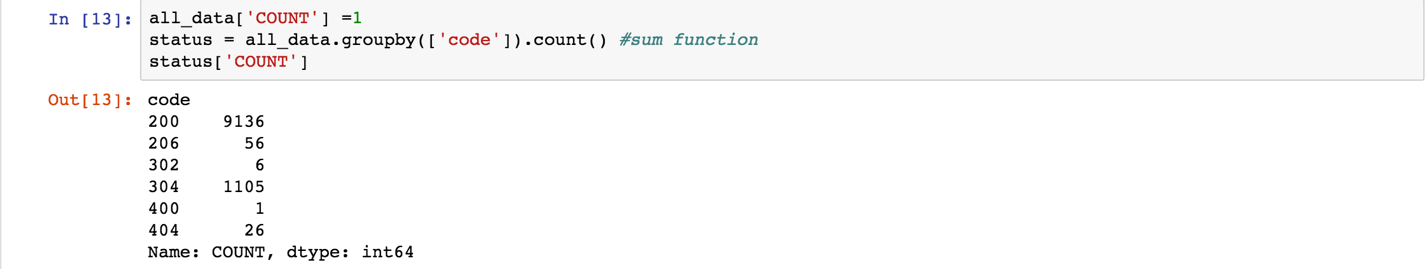
**Step 9:**

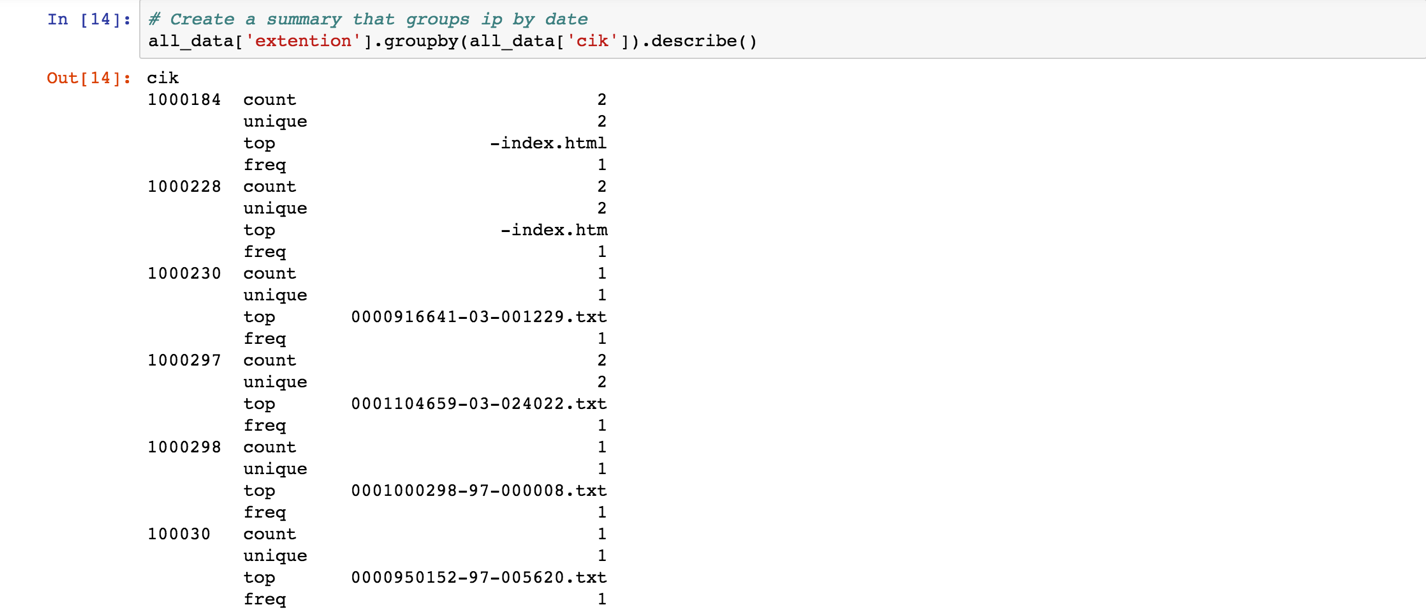
**Find the summaries matrix**











**Step 10:**

**CREATE CSV FOR Final Dataframe**

After getting all the data tables, we have to export data of each of these tables and create a csv file for each table. We will create a folder name with the name of the file and then create output csv inside that folder.

**Step 11:**

**LOGGING EACH STEP IN A LOG FILE**

After the final result, we also created and captured the various processing of the task in the log file named, **Edgar\_log** which will consist of minute details on which process is running and also capture the last point of failure based on the input parameters given.

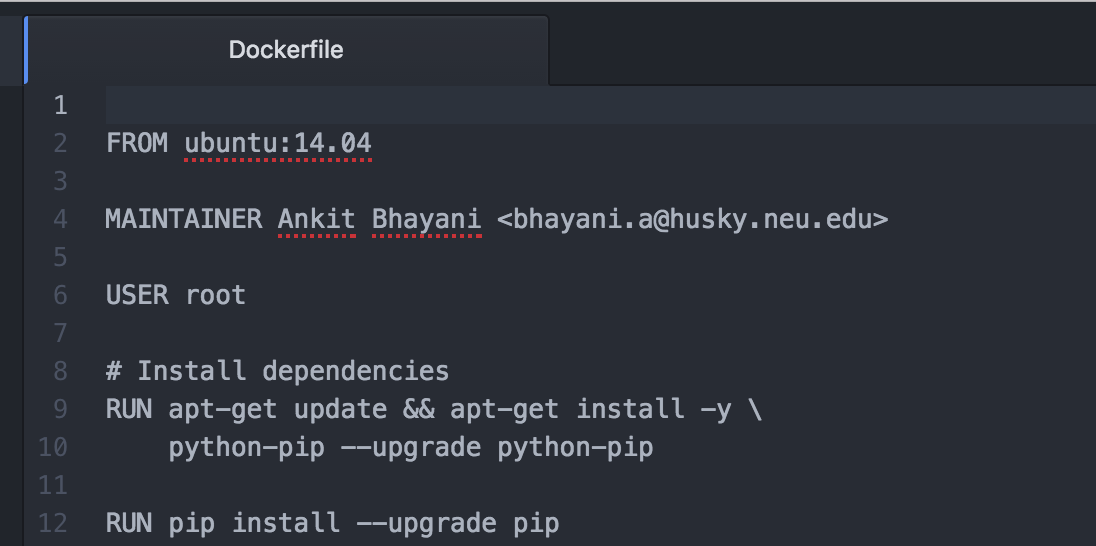
**SECTION - 3 DOCKERIZE THE PROCESS**

**CTIVE**

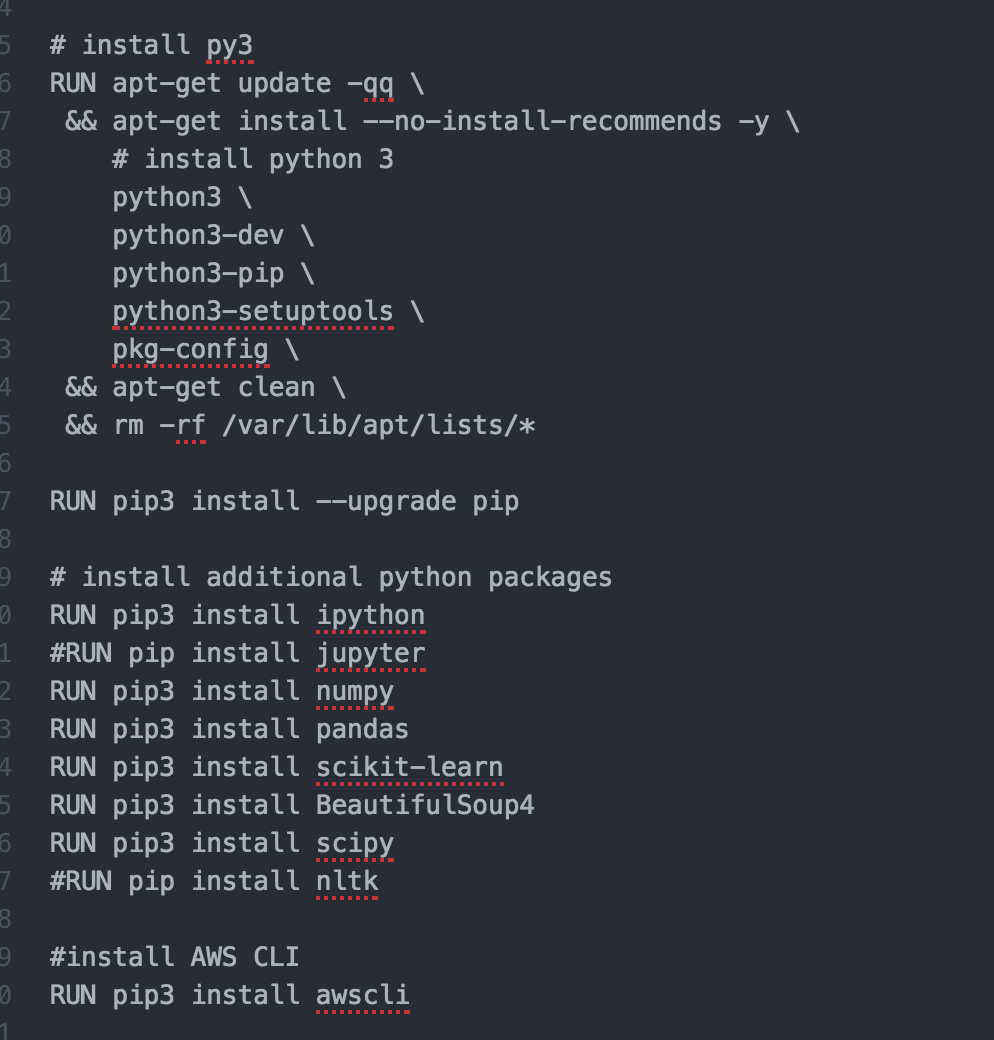
**STEP 1:**

**CREATE THE DOCKER IMAGE FROM DOCKERFILE**

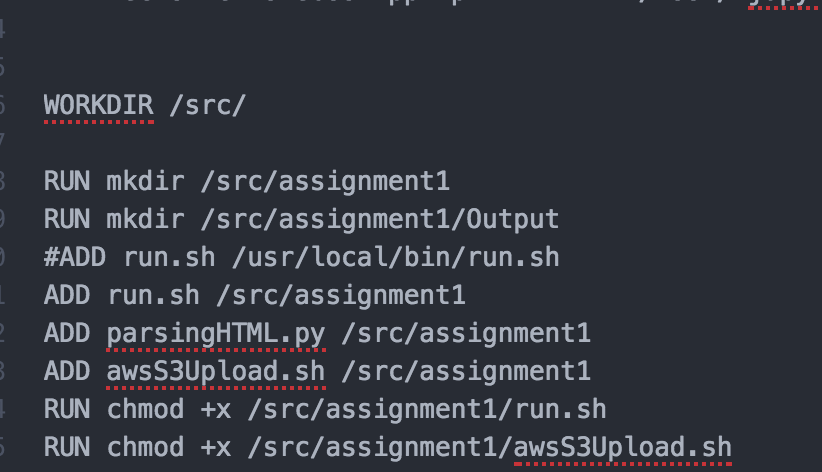
* 1. After taking the base UBUNTU 14.04 image.

****

* 1. Install Python and all required packages which is used in source code

****

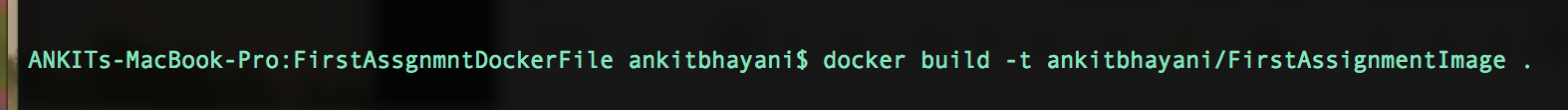
* 1. While building DockerFile; First creating the work directory and then copying the source python and shell files from local machine to DOCKER image.

****

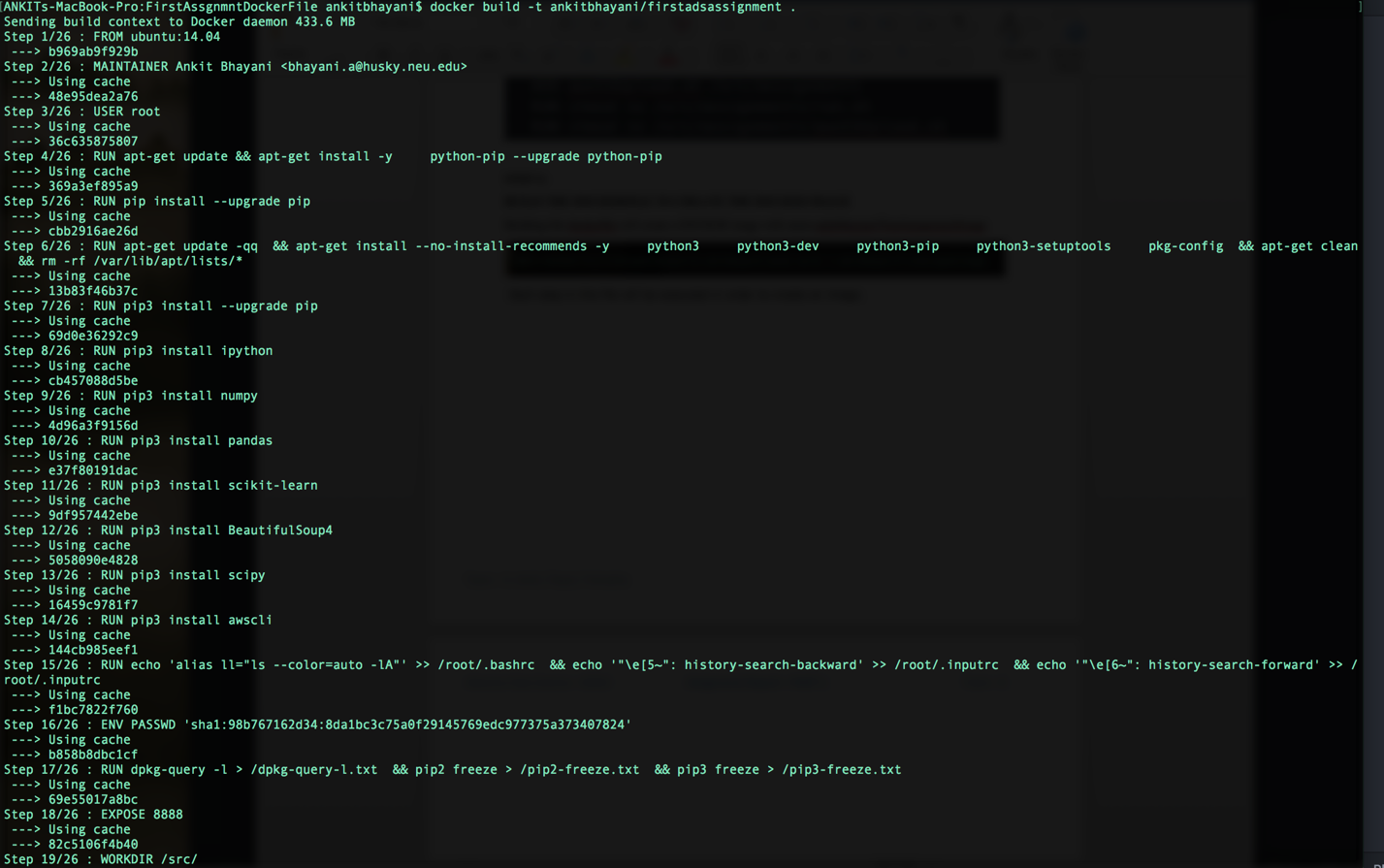
**STEP 2:**

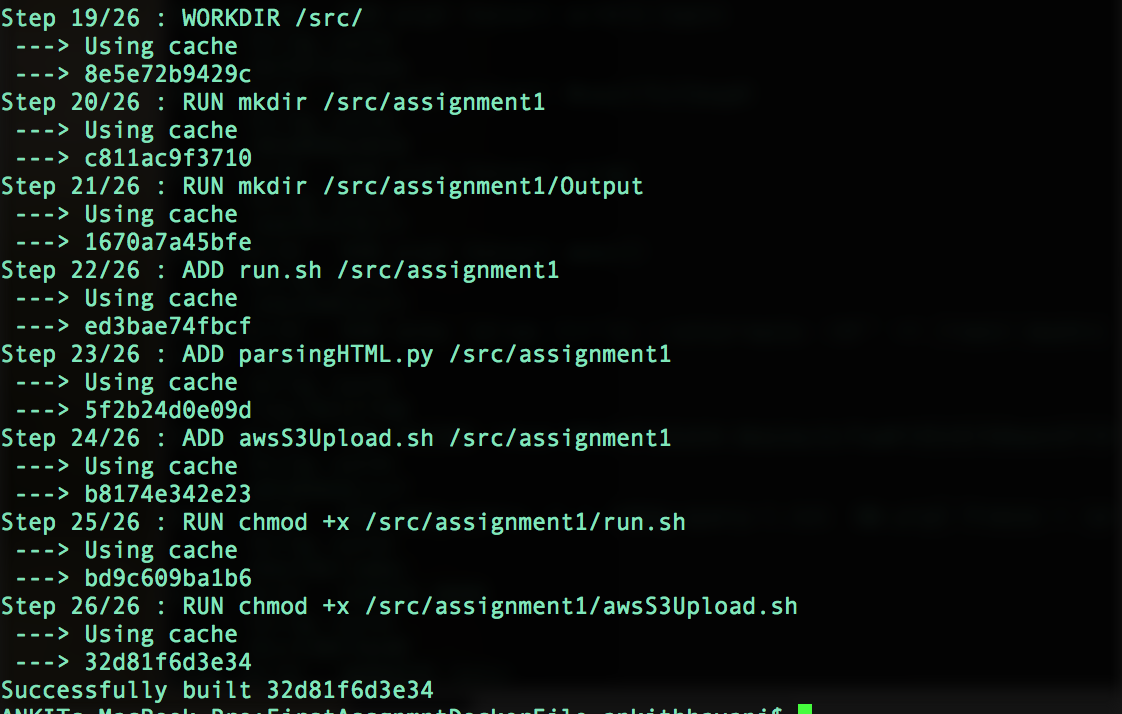
**BUILD THE DOCKERFILE TO CREATE THE DOCKER IMAGE**

Building this dockerfile will create a DOCKER image with name ankitbhayani/FirstAssignmentImage

****

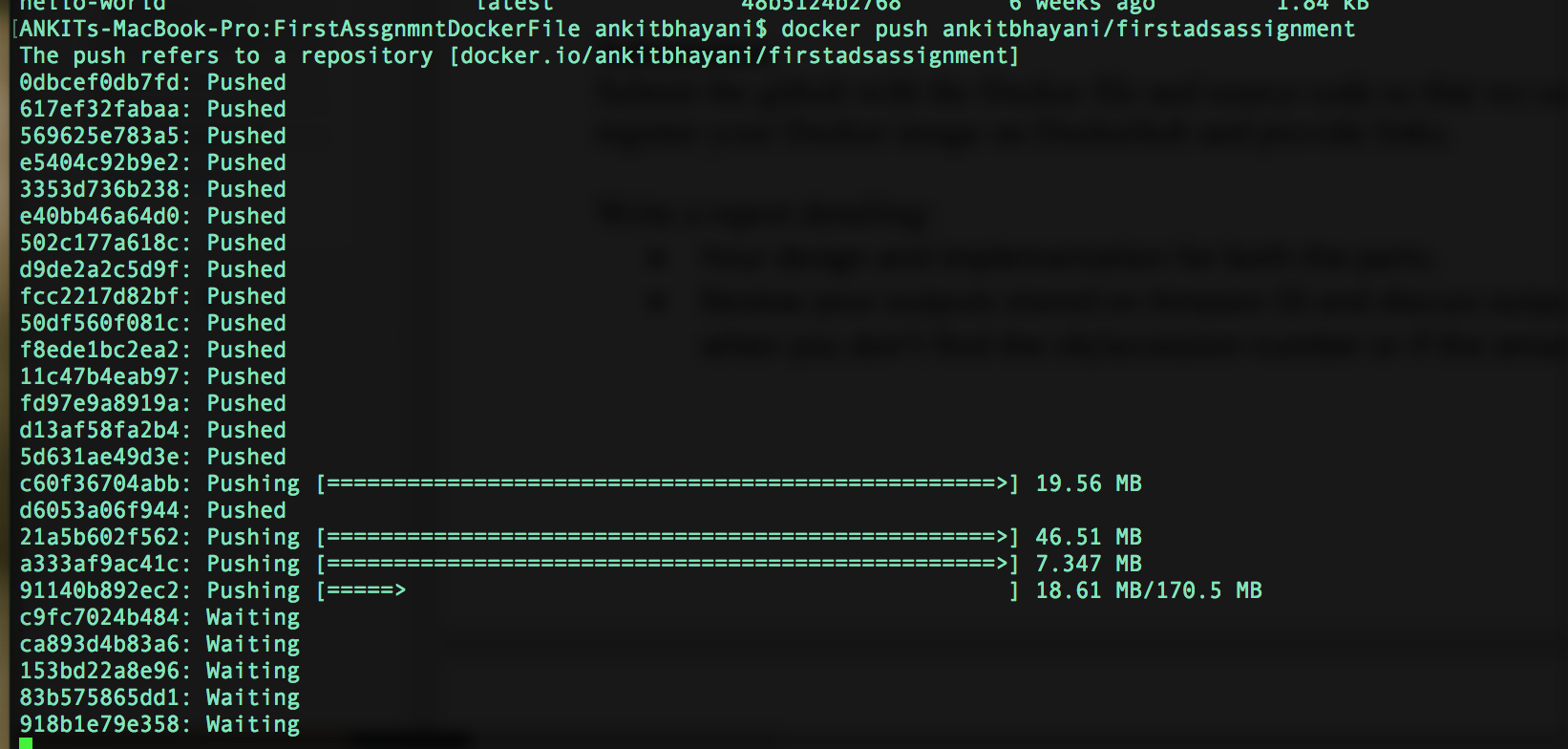
Each step in this file will be executed in order to create an image. AWS CLI is also integrated inside the dockerfile

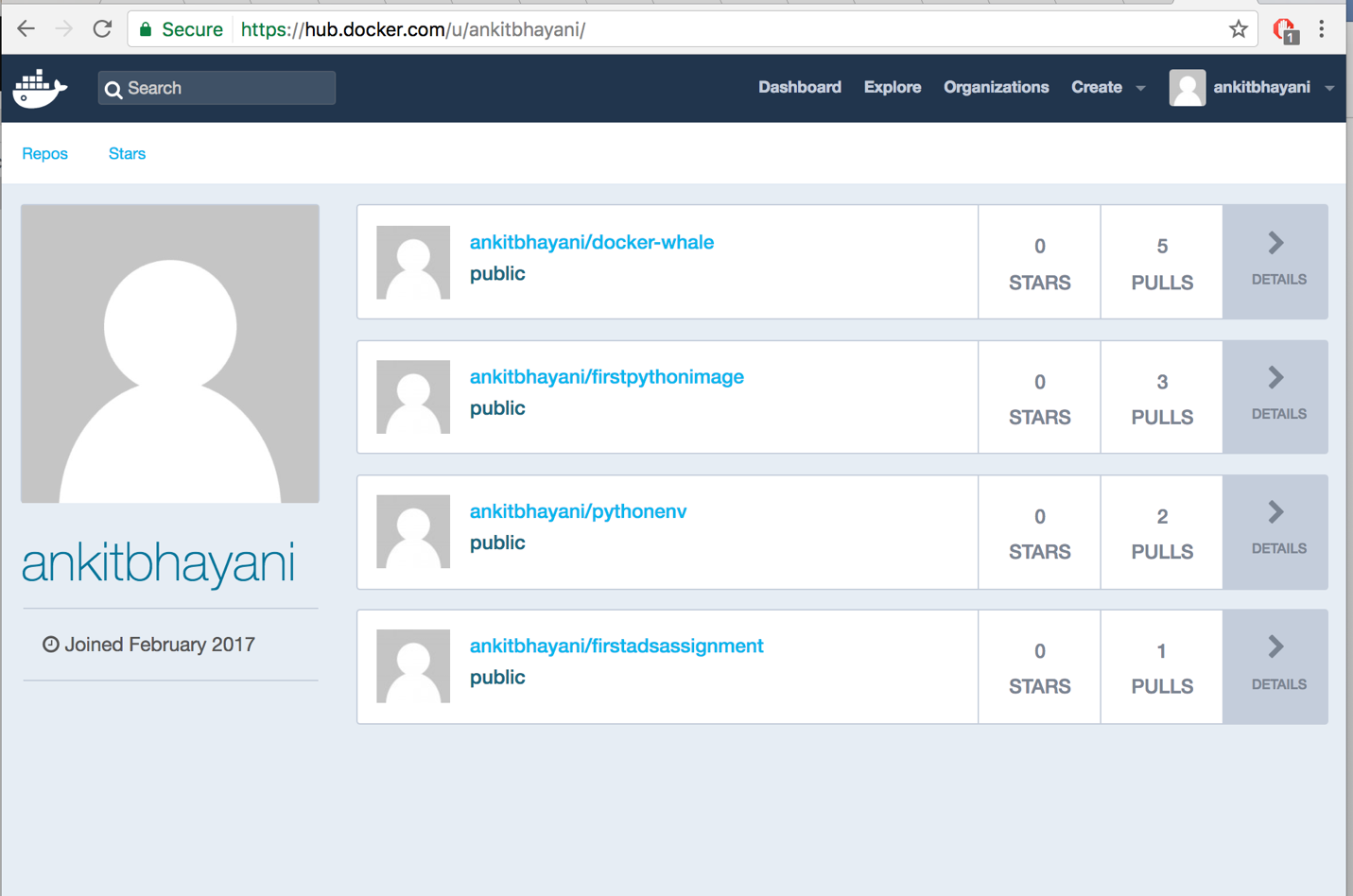


****

**STEP 3:**

**PUSH THIS DOCKER IMAGE to DOCKERHUB**

****



**STEP 4:**

**RUN THE DOCKER IMAGE**

**Command:**

docker run -e ACCESS\_KEY=**<YOUR\_ACCESS\_KEY>**

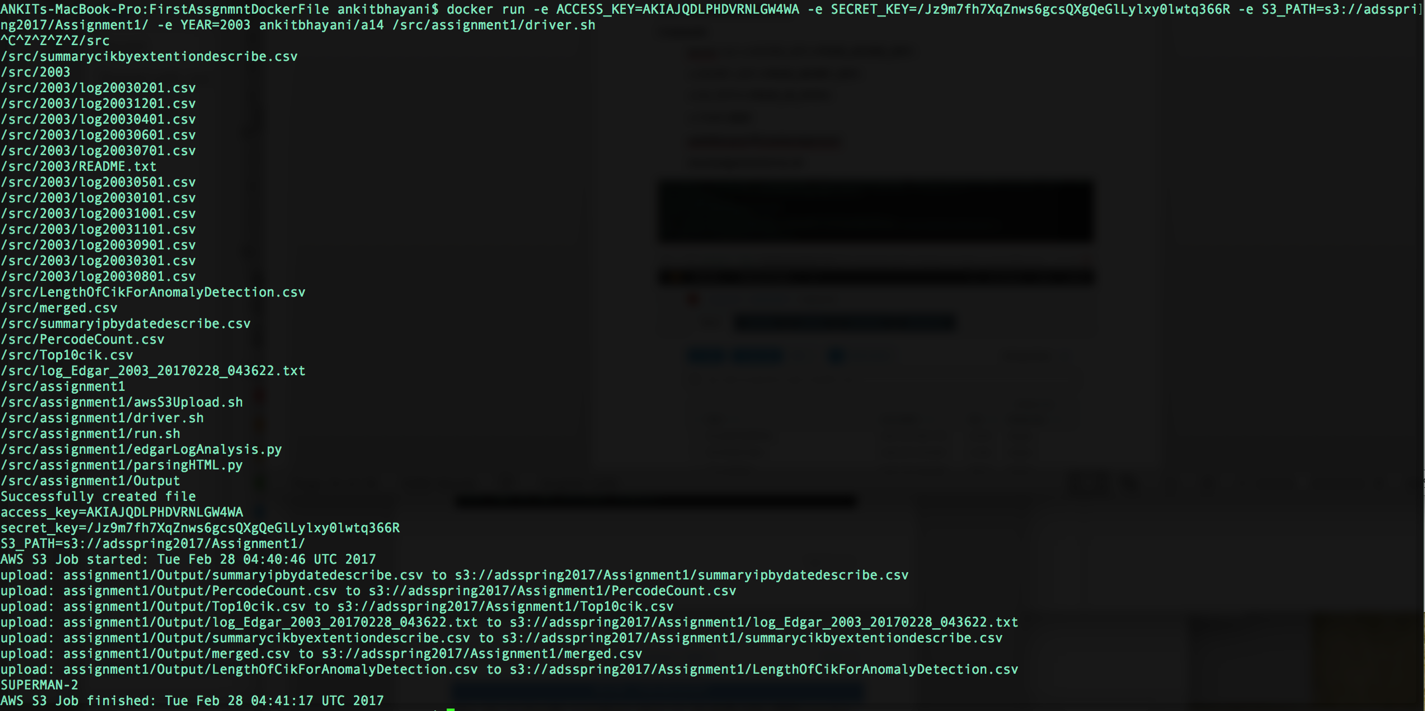
-e SECRET\_KEY=**<YOUR\_SECRET\_KEY>**

-e S3\_PATH=**<YOUR\_S3\_PATH>**

-e YEAR=**2003**

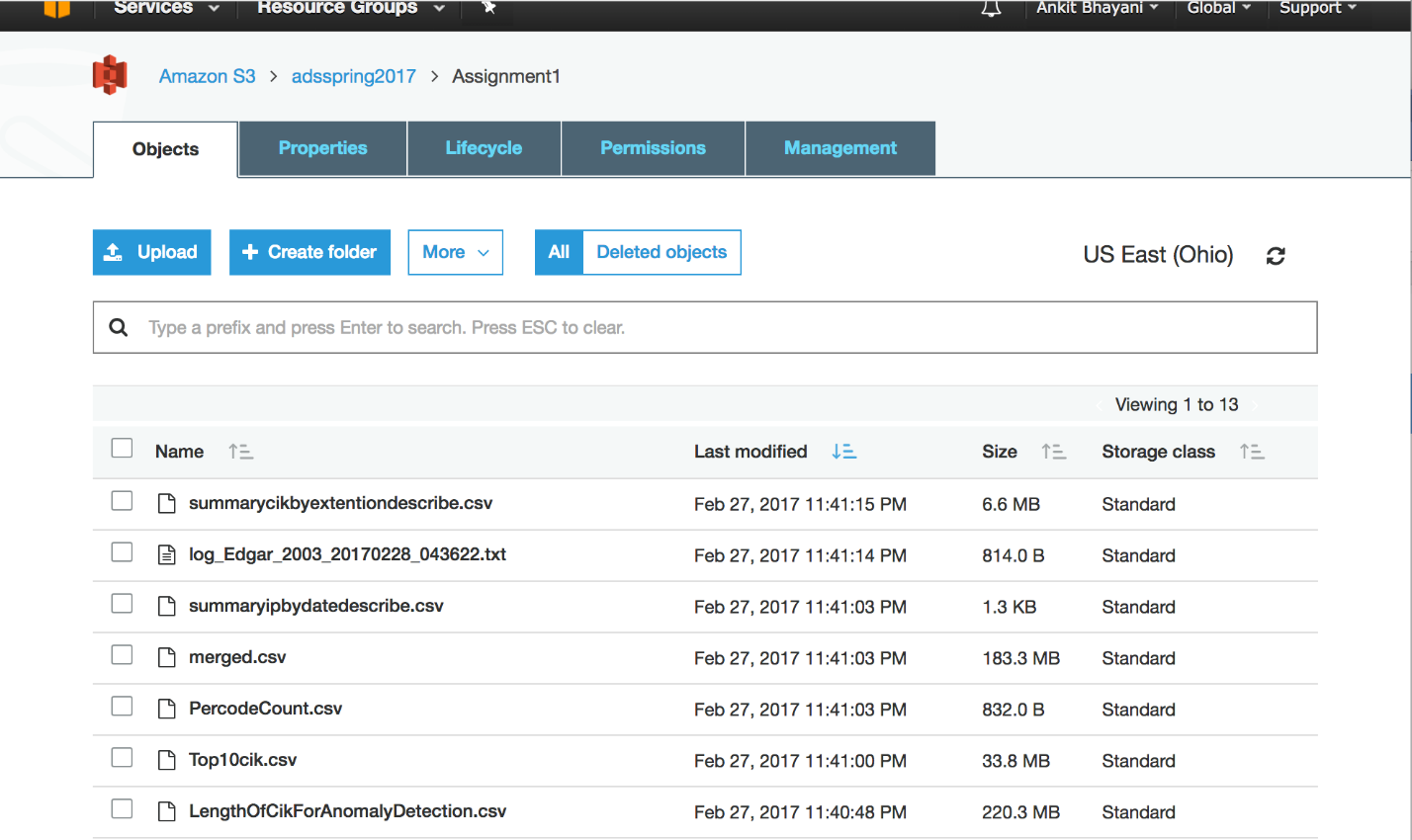
**ankitbhayani/firstadsassignmentpart2**

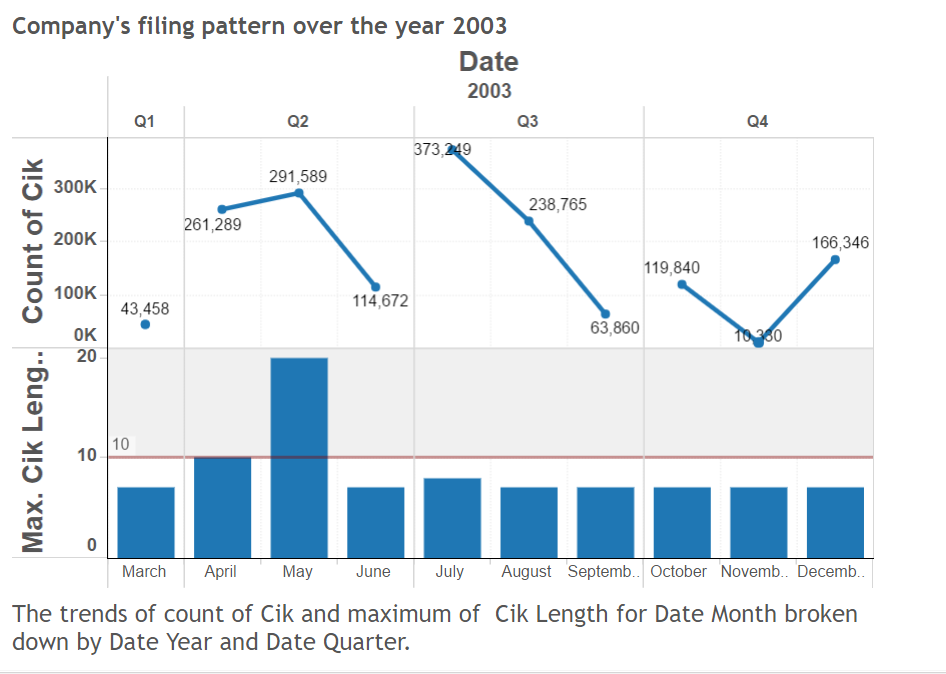
**/src/assignment1/driver.sh**



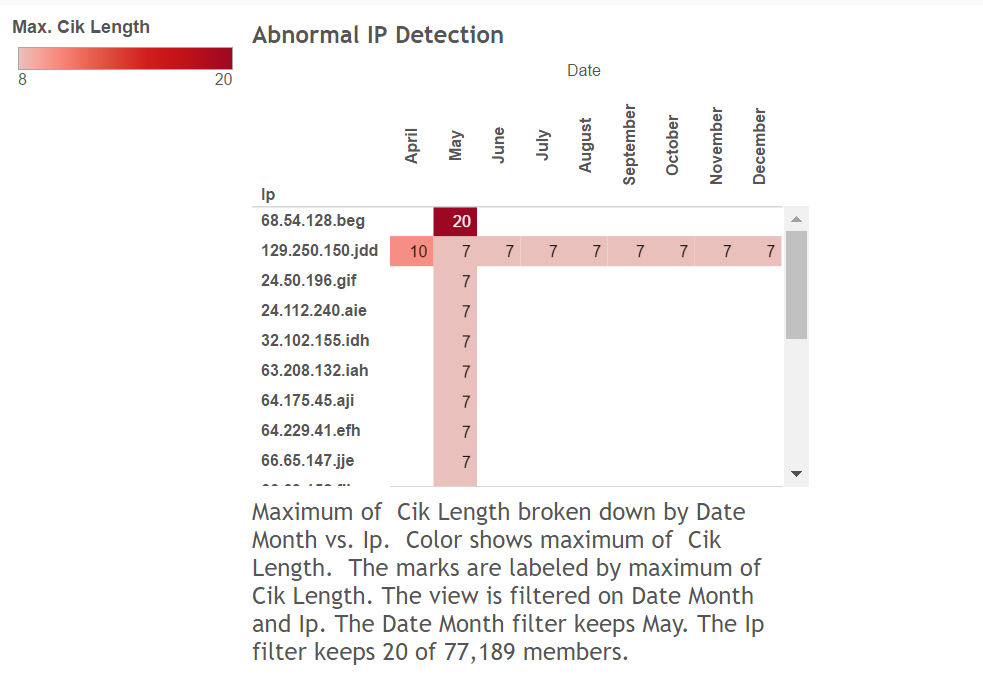
**AWS OUTPUT:**

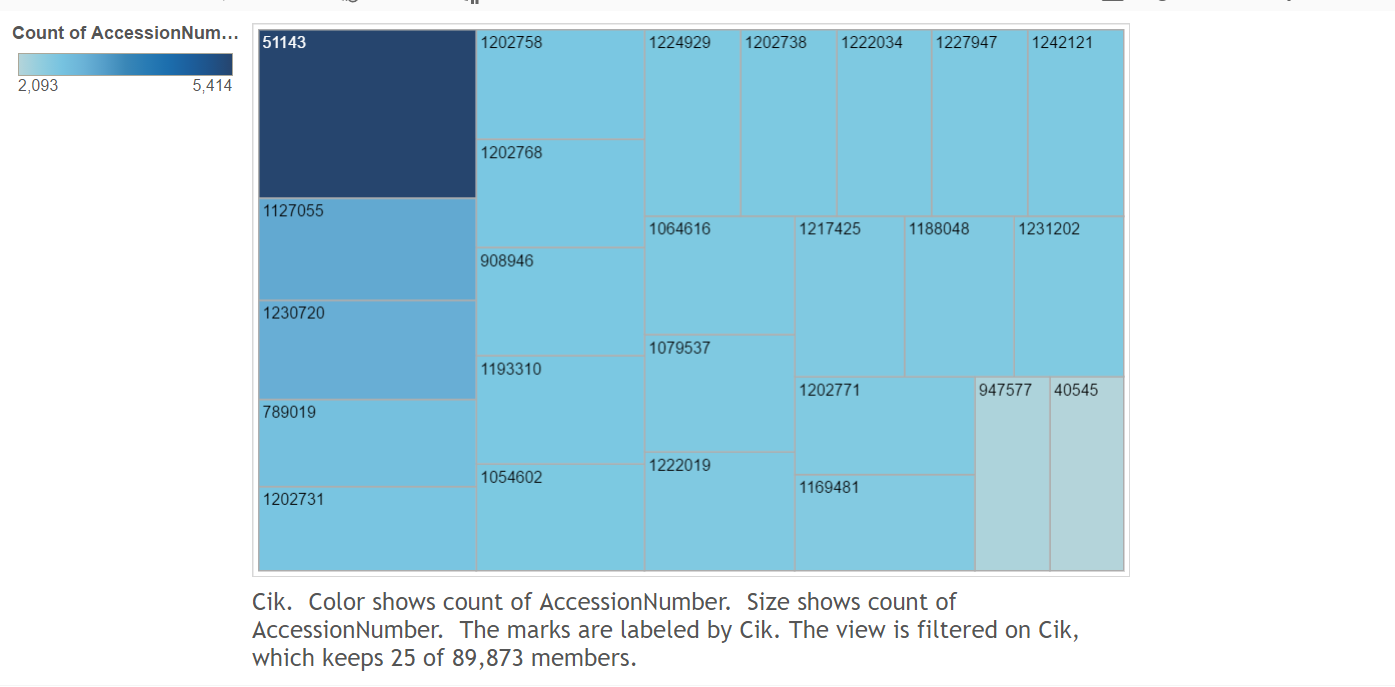
[https://s3.us-east-2.amazonaws.com/adsspring2017/Assignment1/](https://s3.us-east-2.amazonaws.com/adsspring2017/Assignment1/LengthOfCikForAnomalyDetection.csv)

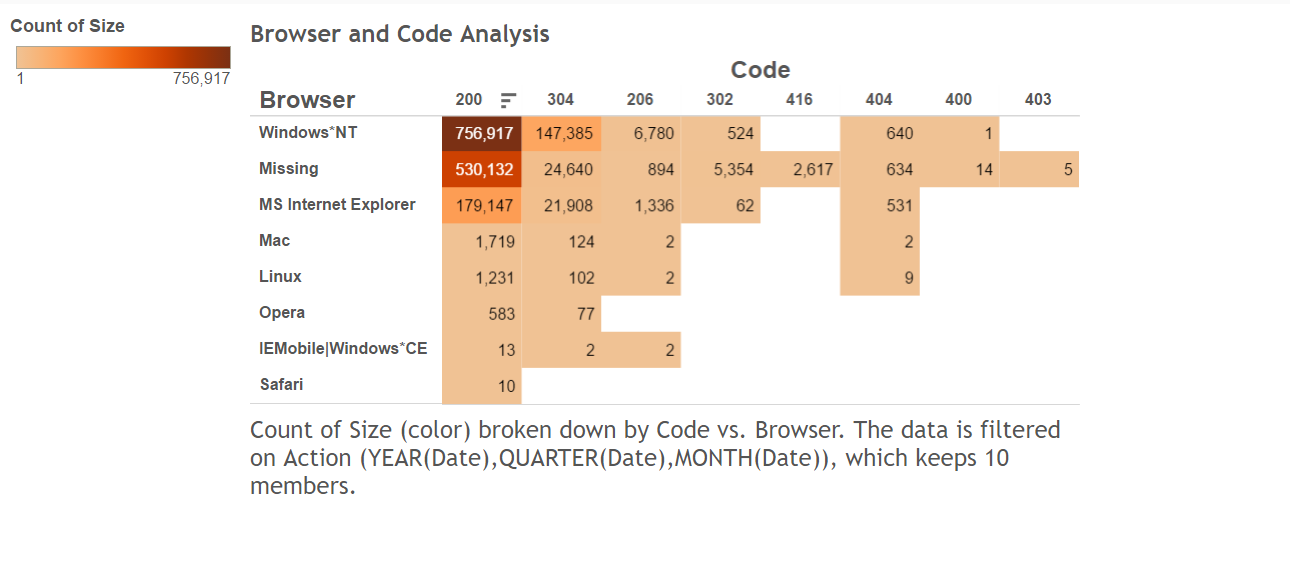




**SECTION – 4 TABLEAU ANALYSIS**







**Dashboard Link:**

[**https://10az.online.tableau.com/#/site/edgarlogfileanalysis/views/PROBLEM-2\_TABLEAU\_REPORT/EdgarLogDashboard?:iid=2**](https://10az.online.tableau.com/#/site/edgarlogfileanalysis/views/PROBLEM-2_TABLEAU_REPORT/EdgarLogDashboard?:iid=2)

