**BEGQ Data Analytics Interview – ANSWERS BY**

**AISHWARYA RAVICHANDRAN**

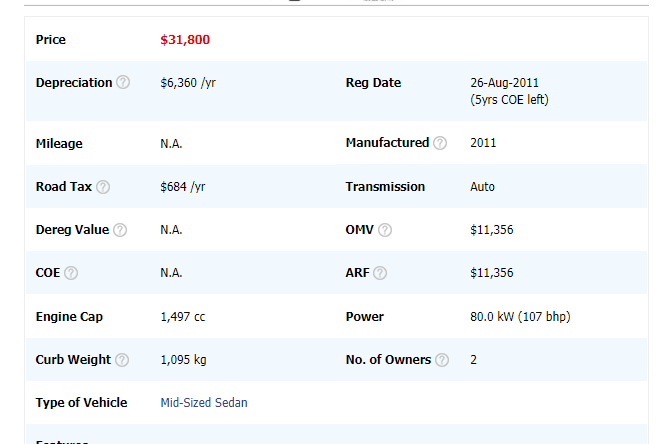
**EMAIL ID:** [**aishwaryaravichandran5@gmail.com**](mailto:aishwaryaravichandran5@gmail.com)

**Phone : +6582165226**

**[ ENTIRE CODE IN DOWNLOADABLE FORMAT CAN BE FOUND IN THE ZIP FOLDER ‘**AISHWARYA\_INTERVIEW\_ANSWERS.zip **‘.**

**AND ATTACHED WITH THE MAIL AND MY GITHUB LINK:** <https://github.com/aishwaryar5/INTERVIEW_ANSWERS_AISHWARYA.git> **]**

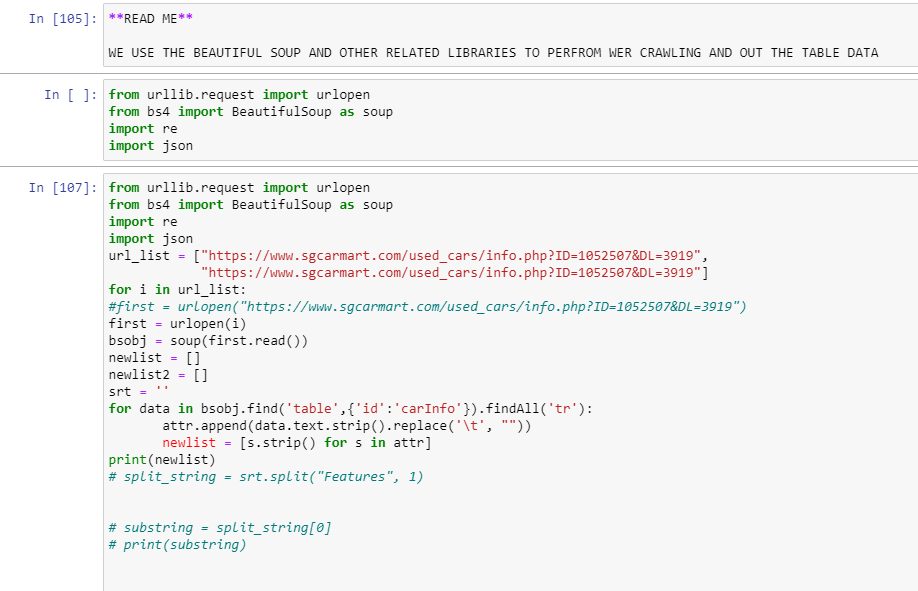
1. Typical reports in micron, are generated by different teams, and are often seen as HTML webapps. We would like to learn more on your approach on how you would crawl webpages. For this question, please develop a script (use Python3 to develop the script, and you may use any publicly available module as you find fit) that can crawl the following data from the attached HTML webpages (Found in folder *Question1*) and output as a dictionary.



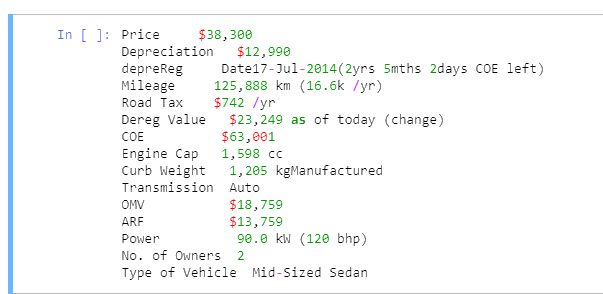
ANSWER: I HAVE USED THE WEB CRAWLING TECHNIQUE USING BEAUTIFUL SOAP AND OTHER LIBRARIES TO CRAWL THE GIVEN HTML TO OBTAIN THE TABLE OUTPUT:

STEPS INVOLVED:

* IMPORT ALL THE REQUIRED LIBRARIES BEAUTIFUL SOUP AND OTHER LIBRARIES
* CREATE A LIST OF WEBSITES TO CRAWL PASS THEM AS INPUT CREATE A OBJECT TO STORE THE OUTPUT DICTIONARY DATA OF THE TABLE



OUTPUT: RESULT WE GET THE OUTPUT DATA IN THE TABLE TD TAG AS BELOW USING WEB CRAWLING BEAUTIFUL SOAP TECHNIQUE ON HTML.



1. In the folder *Question2*, you will find two csv file:
   1. NewData.csv
   2. MasterDB.csv

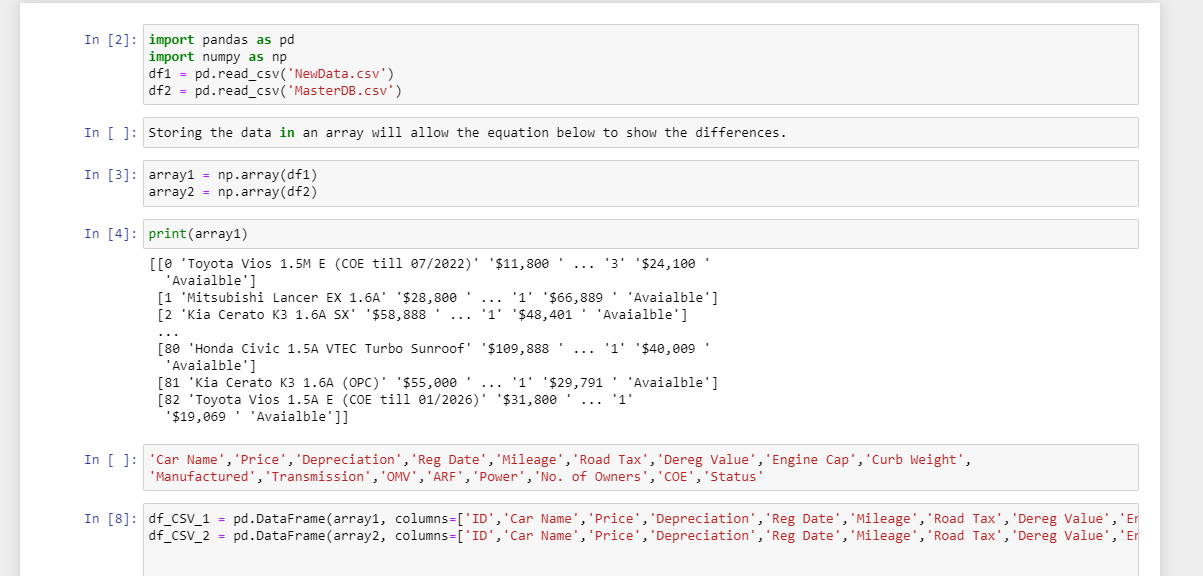
You need to read both the *‘NewData.csv’*, and the ‘*MasterDB.csv’*, and do the following:

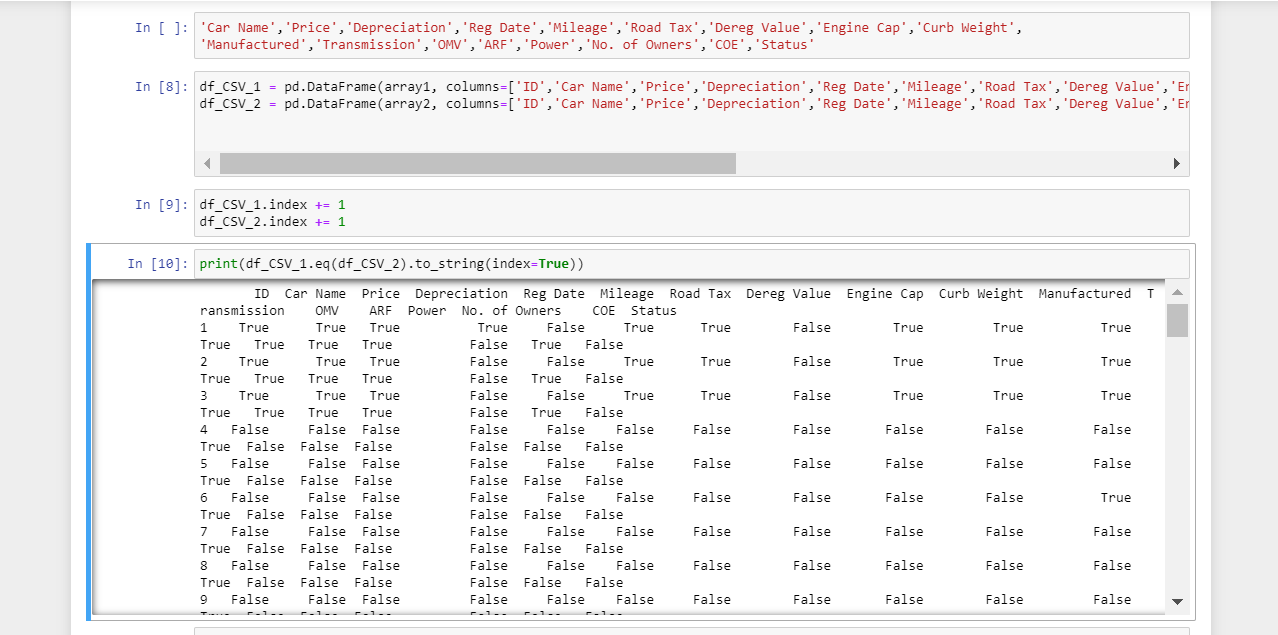
1. Check if there are new lines in the ‘NewData.csv’, and append them to the existing *‘MasterDB.csv’*, as long as the ‘Status’ in the row is ‘Available’, and the ‘Price’ and ‘COE’ columns are not ‘N.A’ (has value in $).
2. For the existing lines, see if the NewData.csv, contains any changes. If yes, update the changes in the *‘MasterDB.csv’.*
3. If the column ‘Status’ in the NewData.csv is ‘Sold’, then remove those lines from the *‘MasterDB.csv’*

Extra points, for making the code modular and developing a strong architecture. Imagine developing for a production environment where the csv files are typically SQL tables where your connection time to the tables are limited.

**ANSWER:**

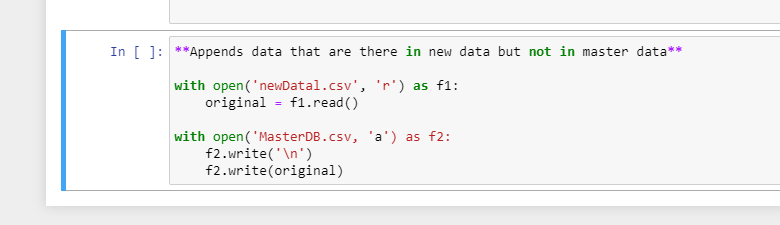
Check if there are new lines in the ‘NewData.csv’, and append them to the existing *‘MasterDB.csv’*, as long as the ‘Status’ in the row is ‘Available’, and the ‘Price’ and ‘COE’ columns are not ‘N.A’ (has value in $).



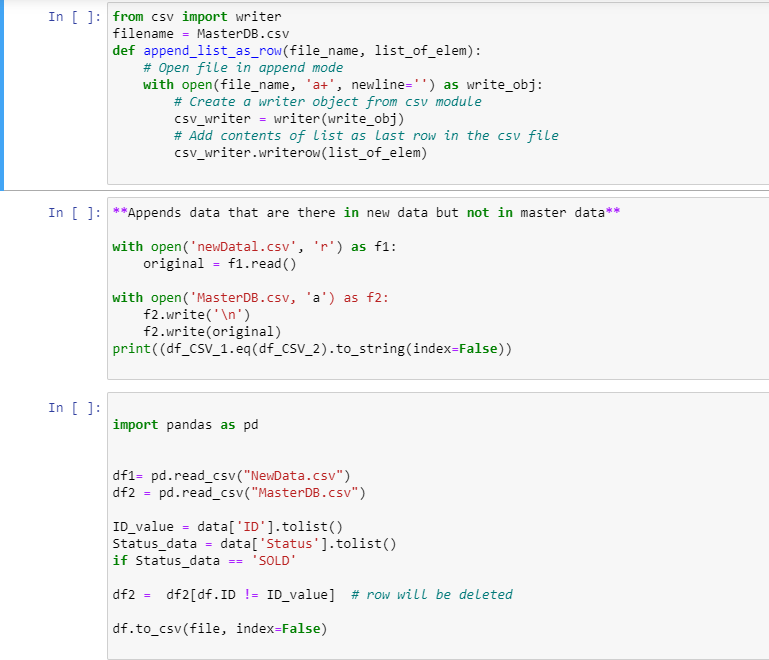


HERE USING THE ABOVE CODE WE COMPARE THE UNIQUE IDS IN MASTER DATA WITH NEW DATA FILE IF COMPARISON OUTPUT IS FALSE IT MEANSE THERE IS A NEW LINE IN THE NEW DATA FILE AND WE NARROW DOWN THE ID AND APPEND THE SAME USING THE APPEND FUNCTION INTO THE MASTER DB DATA.

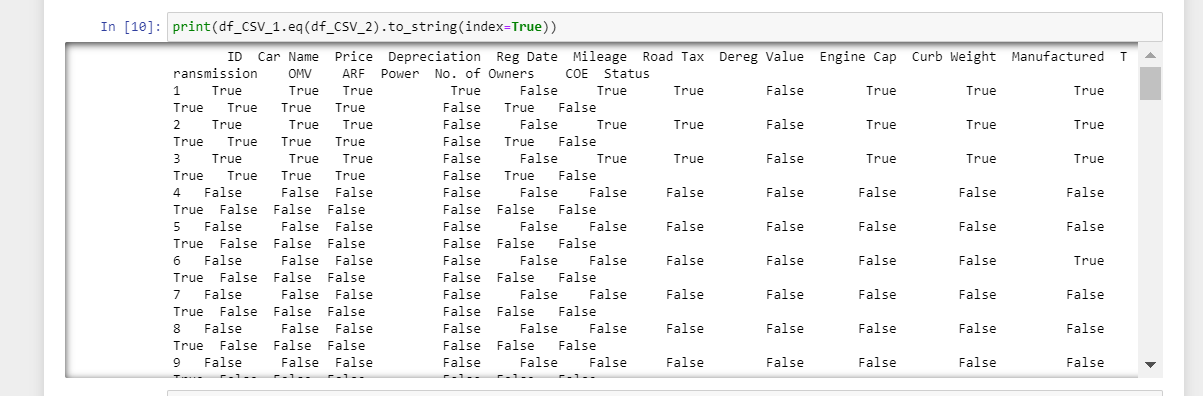
WE APPEND DATA THAT ARE IN NEW DATA BUT NOT IN MASTER DB:

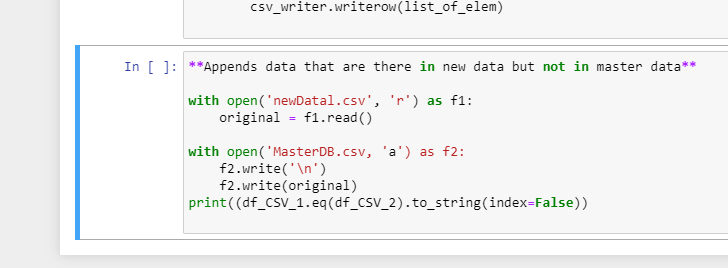


FOR ALL THE VALUES IN NEW DATA WHICH IS DIFFERENT FROM MASTER DATA(OUTPUT WILL BE REPRESENTED AS FALSE) WE UPDATE THE MASTER DB:



RESULT : WE DELETE ROW FROM MASTER DB WHERE NEW DB STATUS IS SOLD



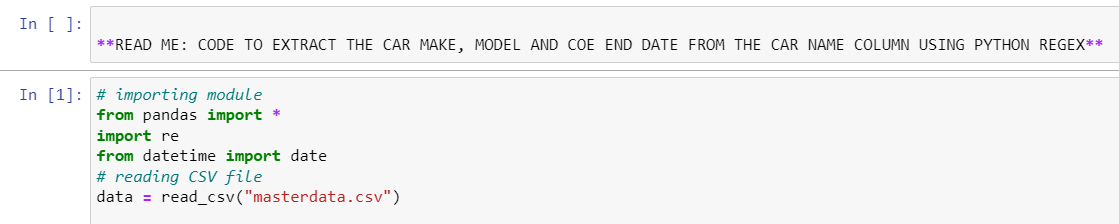


1. Find the CSV in folder Question4 perform the following actions:
   1. Develop a script that can split Column ‘Car Name’ to get the following attributes
      1. Car Make
      2. Car Model Name
      3. COE End Date
   2. Build statistical model for every car make
      1. Mean, Median, mode
      2. +- 3 Sigma Value
   3. What kind of useful metrics/graphs that you can get out of this data for a potential customer?
      1. Support your answer with graphs and explanations on why and how you chose the metrics

**ANSWER FOR QUESTION 3:**

1. **Using the below python script that I had developed, I have split the column car name into the 3 attributes: car make, car model, COE end date.**

**STEP 1: CREATED A READ ME TO BRIEF ABOUT THE STEPS USED, IMPORTED THE CSV FILE INTO MY PYTHON 3 JUPYTER DIRECTORY. AND USED REGEX ON THE CSV CAR NAME COLUMN TO OBTAIN THE 3 ATTRIBUTES**



**STEP 2: I USED THE REGEX EXPRESSION:**

Car\_Make = [re. match(r'[^\d]+', x).group(0) for x in Car\_Name] AND OBTAINED THE CAR MAKE DATA AND STORED IT IN A LIST

SIMILARLY ,I USED THE REGULAR EXPRESSION TO OBTAIN THE CAR MODEL DATA AS SHOWN BELOW

timeline = re.compile(r'^[0-9]\*[-][yr]')

coe = re.compile(r'(?<![/\d])(?:0\d|[1][012])/(?:19|20)?\d{2}(?![/\d])')

r = re.compile(r'^[0-9]\*[.,]{0,1}[0-9][a-zA-Z]\*$')

if r.match(j):

Car\_Model.append(j)

if coe.match(j):

p = (j[:-1])

COE\_END\_DATE.append(p)

if timeline.match(j):

timeline2.append(j)

for z in timeline2:

y = z.split('-')

k =y[0]

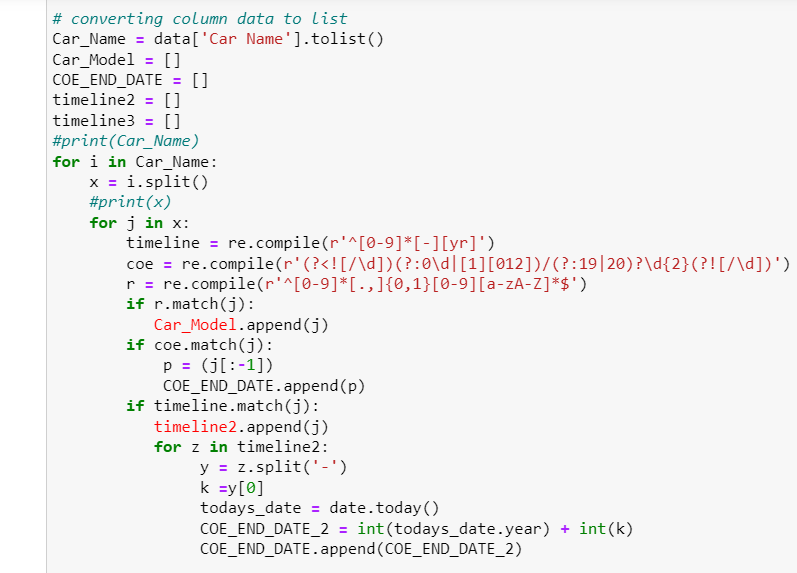
todays\_date = date.today()

COE\_END\_DATE\_2 = int(todays\_date.year) + int(k)

COE\_END\_DATE.append(COE\_END\_DATE\_2)

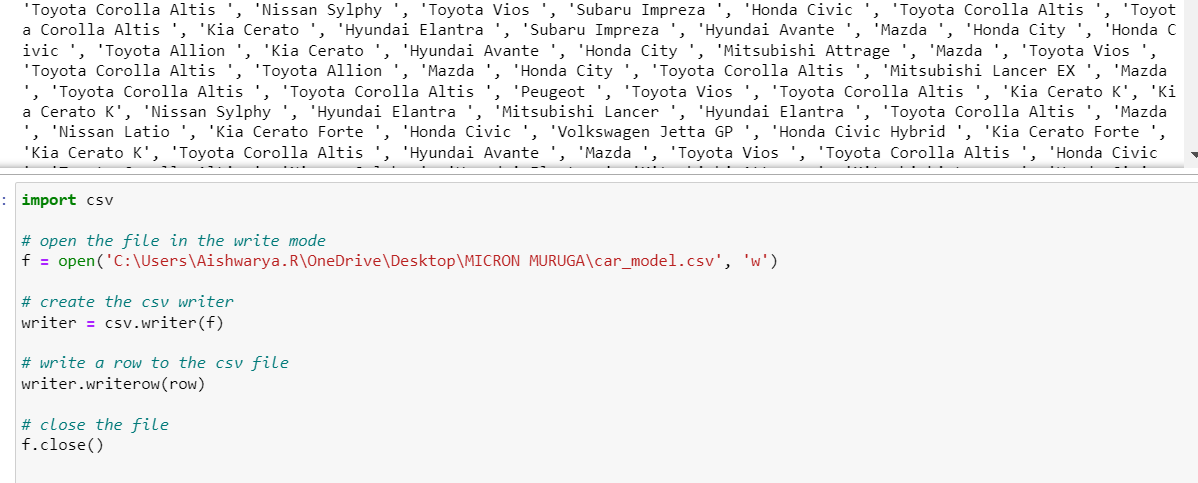
**IN ORDER TO OBTAIN THE COE END DATE, I USED 2 METHODOLOGIES:**

* **TO FIND THE VALUE WITH DATE FORMAT MM/YYYY**
* **TO FIND THE NUMBER OF YEARS QUOTED IN CERTAIN CAR DETAILS EG. 5-YR , 10-YR ETC AND PERFORMED ADDITION TO OBTAIN THE TOTAL NUMBER OF YEARS FOR COE TO ED BY ADDING THE TIME LINE[ 5 OR 10 YEAR ETC] TO THE CURRENT DATE -YEAR(YYYY) AND THEREBY FINDING THE END COE DATE FOR BOTH CASES.**



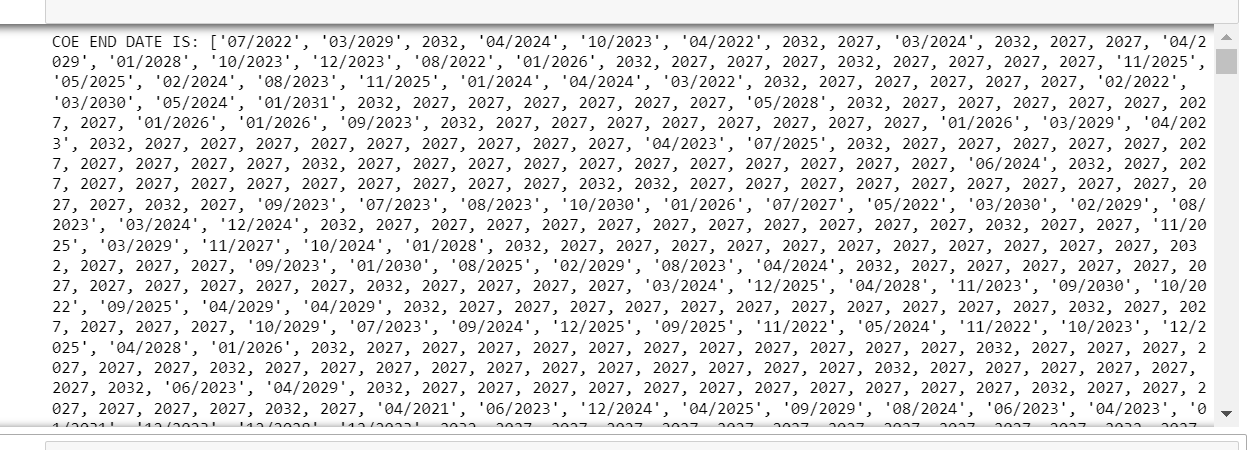


**I USED THE BELOW CODE TO OUTPUT THE CAR MAKE DATA TO A CSV FILE TO PERFORM STATISTICAL ANALYSIS FOR PART b.**

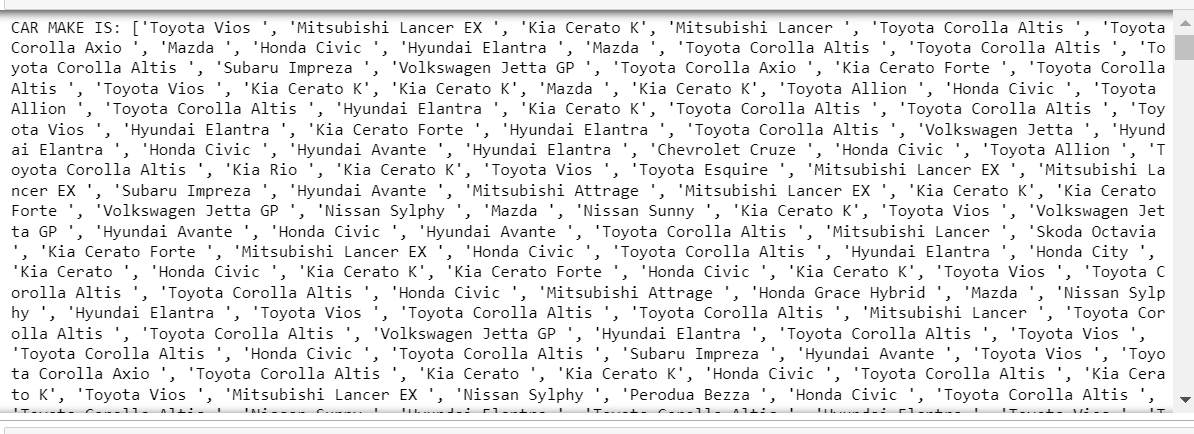


**OUTPUT:**

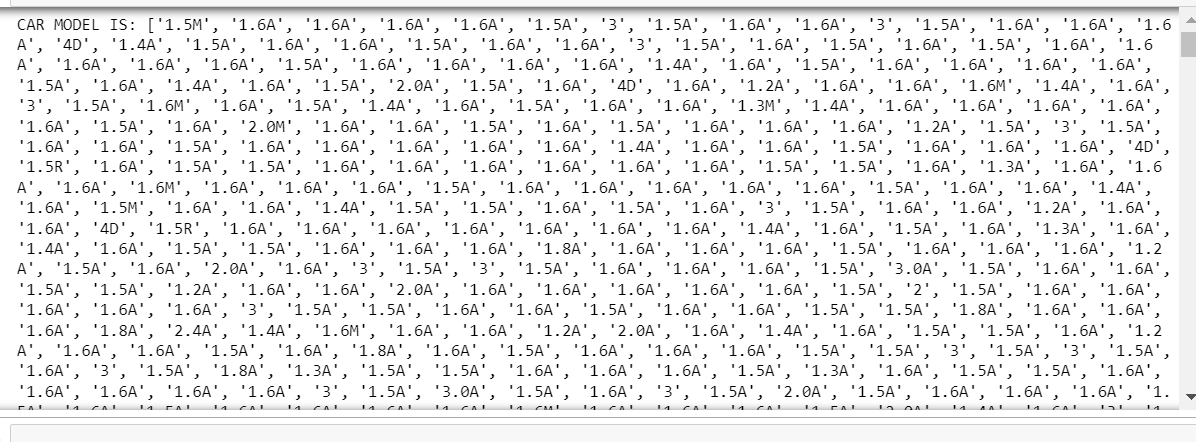
**RESULTS: COE END DATE OUTPUT CONSISTS OF 2 TYPES OF OUTPUT : MM/YYYY FORMAT AND YYYY FORMAT BASED ON LOGIC APPLIED BASED ON DATA AND CODE AS EXPLAINED ABOVE.**



**OUTPUT CAR MAKE DATA**

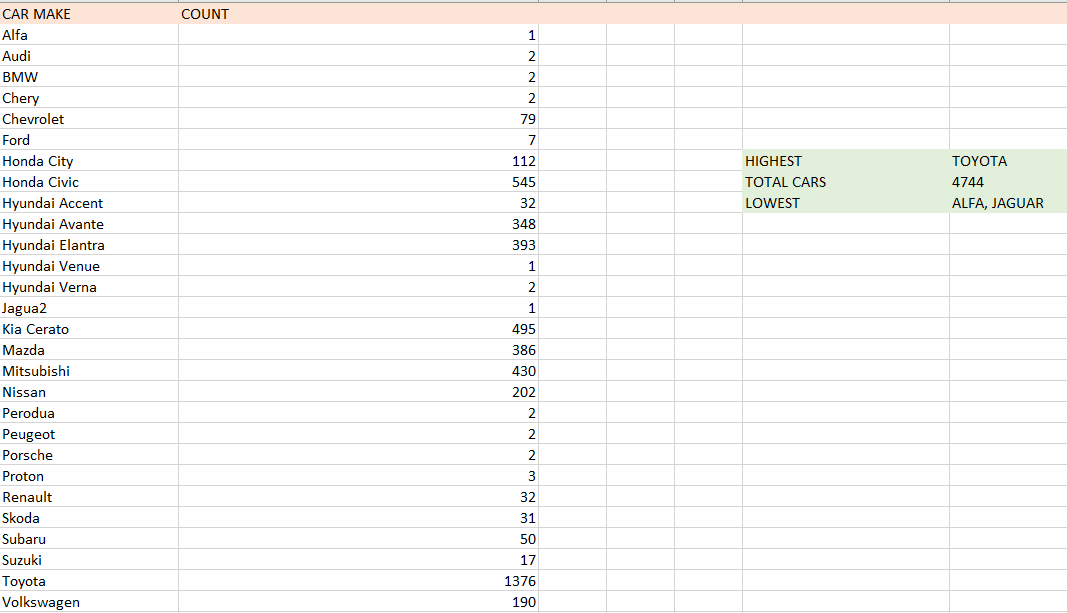


**OUTPUT CAR MODEL DATA**

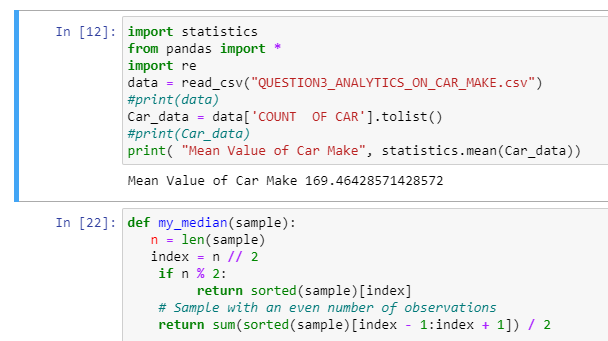


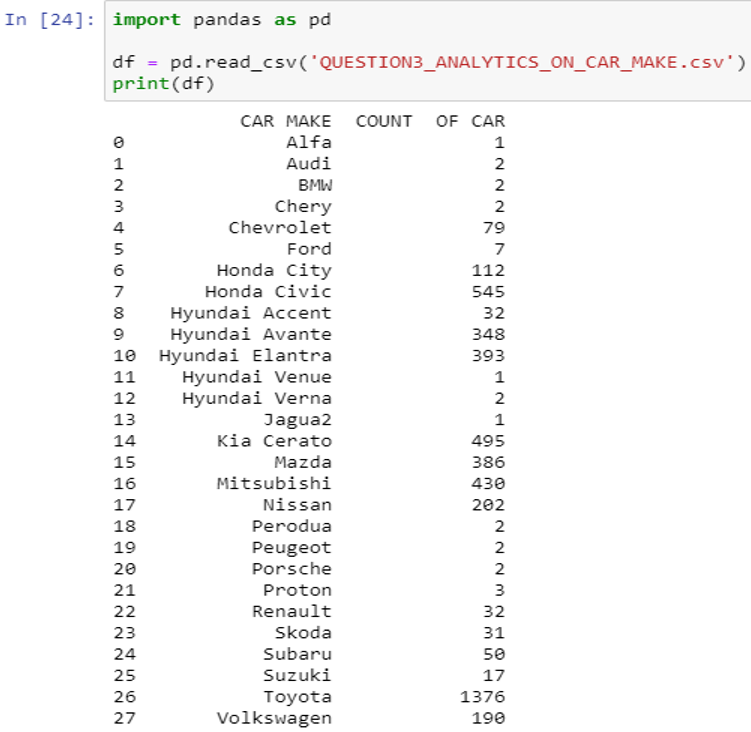
1. **BUILD STATISTICAL MODEL TO FIND MEAN MEDIAN MODE AND +-3 SIGMA VALUE FOR CAR MAKES**

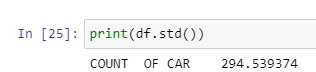
* **I EXPORTED THE CAR MAKE DATA FROM PART A AND SAVED IT IN AN EXCEL WORKBOOK.**
* **I CREATED NEW ATTRIBUTES AND USED RELEVANT FEATURE ENGINEERING TECHNIQUES TO OBTAIN THE NEW COLUMNS RELEVANT FOR MY ANALYSIS.**



**FINDING MEAN , MEDIAN , MODE AND STD DEVIATION USING PYTHON FOR CAR MAKE.CSV DATA CREATED :**





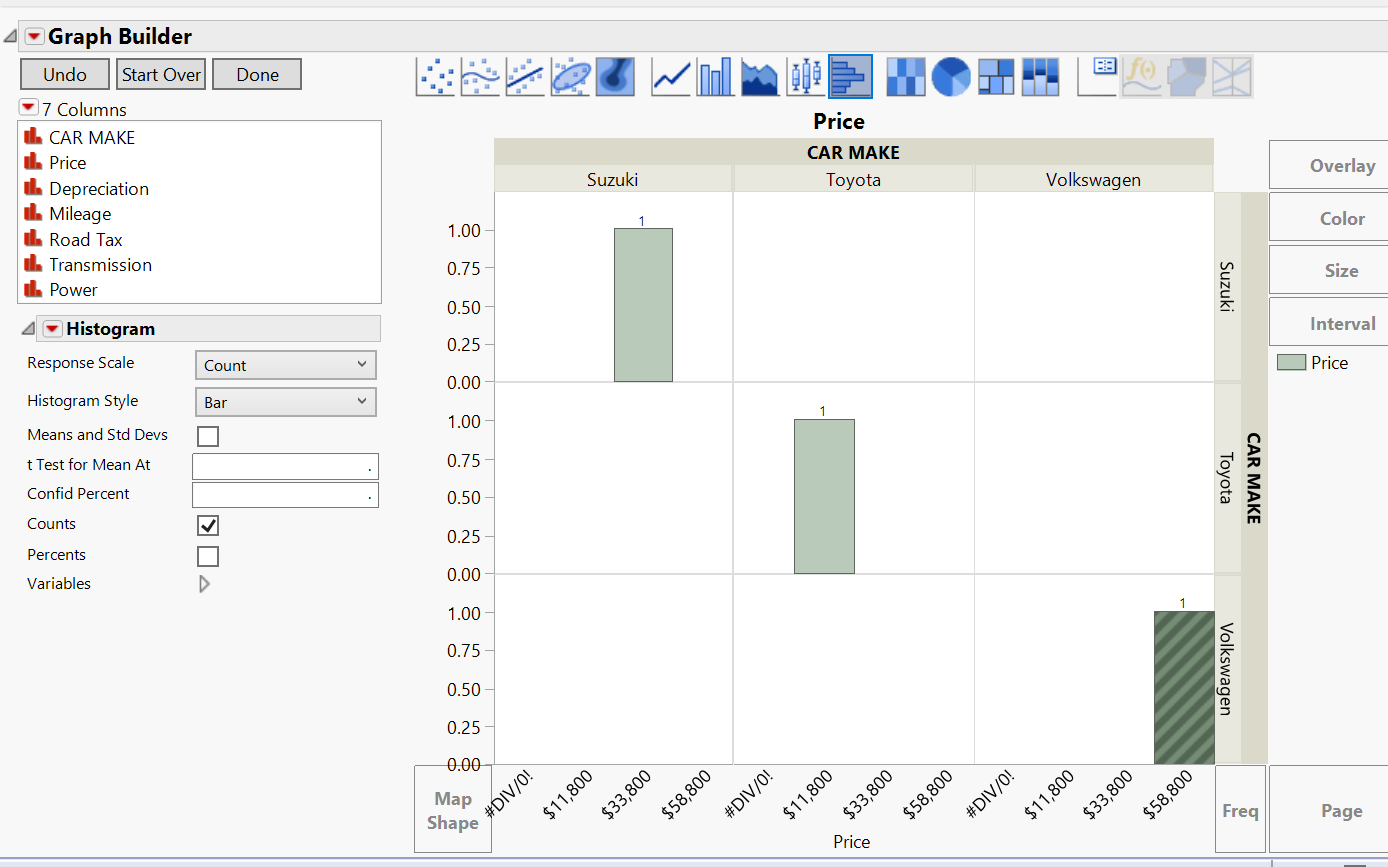


**C ] What kind of useful metrics/graphs that you can get out of this data for a potential customer?**

* + 1. **Support your answer with graphs and explanations on why and how you chose the metrics**

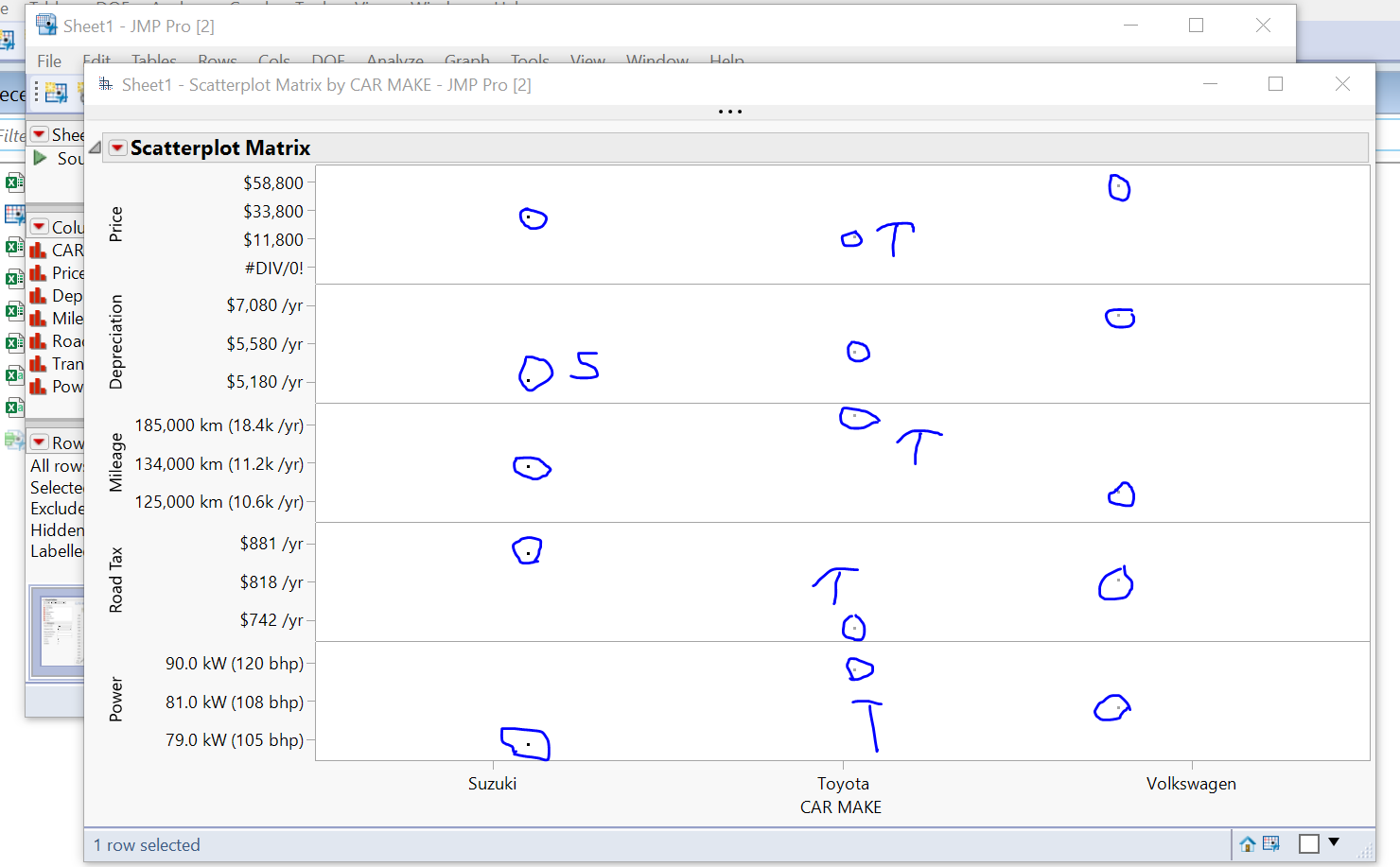
ANSWER : I WILL CONSIDER THE FACTORS THAT PLAY A MAJOR ROLE FOR CUSTOMER TO DECIDE ON BUYING THAT CAR : PRICE, MILEAGE, COMFORT, POER, AUTOMATIC OR MANUAL ALLTHESE FACTORS DETERMINE THE LIKEABILITY OF POTENTIAL CUSTOMER TOWARDS A PRODUCT.

IN THE BELOW ANALYSIS I COMPARE PRICE VS MAKE USING JMP:

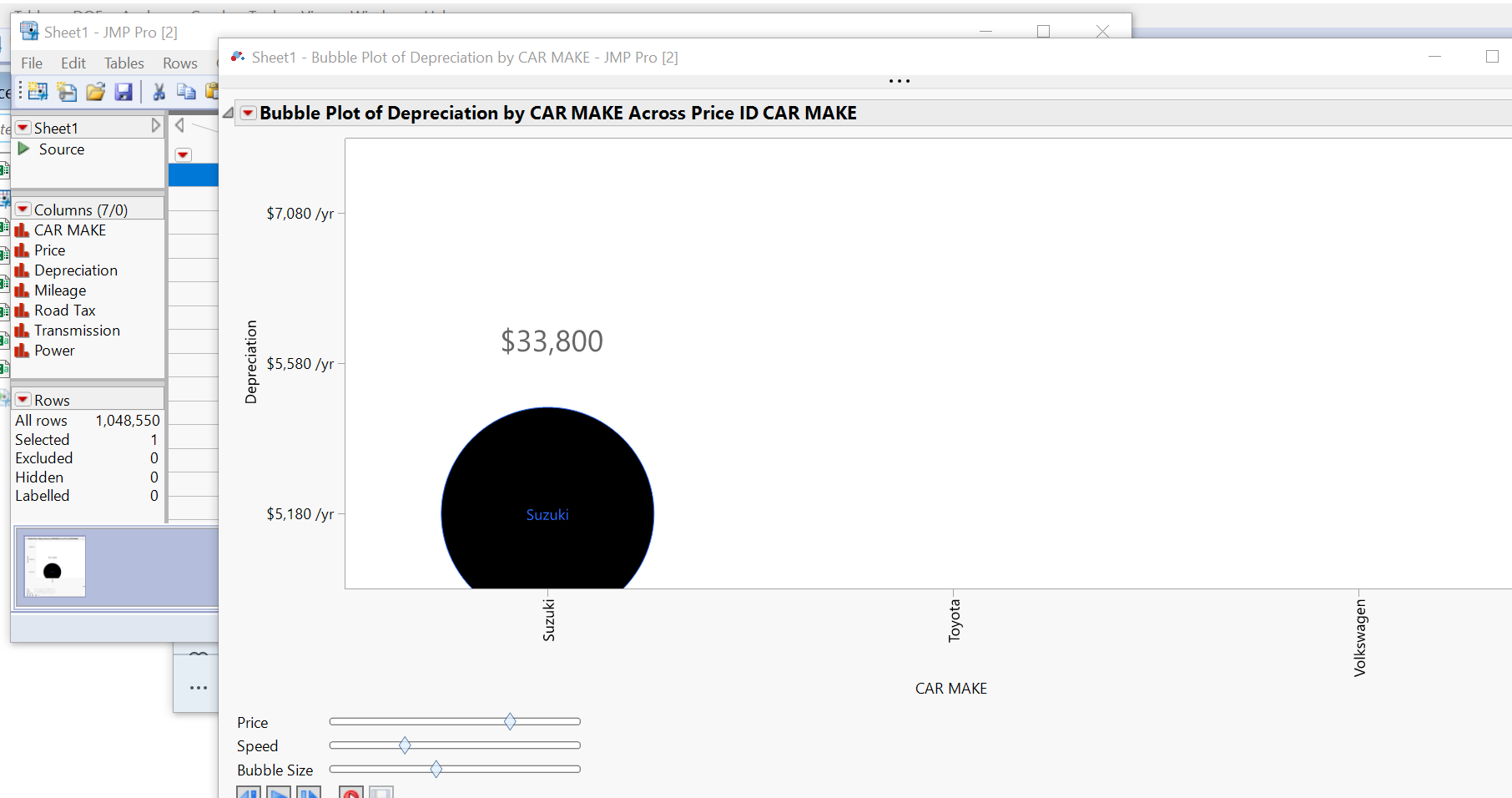


WE SEE THAT SUZUKI IS THE LEAST EXPENSIVE HENCE HAS HIGHEST LIKEABILITY AMONG CUSTOMERS.

SCATTER PLOT COMPARISON: TOYOTA VS SUZUKI VS VOLKSWAGEN:



**WE SEE THAT MAXIMUM ADVANTAGE IS ASSOCIATED WITH TOYOTA CAR HENCE IT IS THE MOST PREFERABLE.**



**WE USE THE BUBBLE PLOT TO PERFORM VALUE FOR MONEY ANALYSIS WE FIND THAT SUZUKI HAS LEAST DEPRECIATION VALUE AND LONGER LIFETIME HENCE WINS THIS CATEGORY.**

1. Build a script that can extract number plate values as a string from the below image
   1. Split the string as [XXX]-[0000]-[X] (starting 2/3 characters, middle numbers and trailing character)
   2. Additional points if you can relatively accurately extract the color of the cars you see



**ANSWER FOR QUESTION 4:**

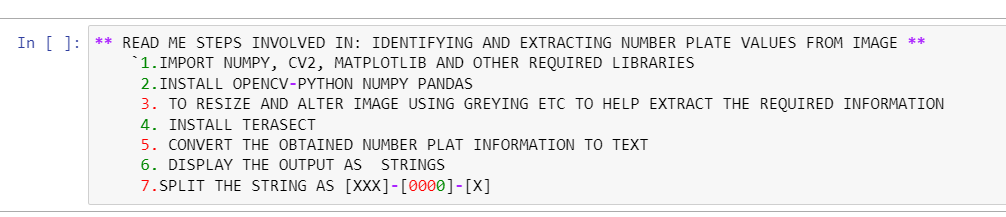
1. **IDENTIFYING AND EXTRACTING NUMBER PLATE VALUES FROM IMAGE:**

**[ ENTIRE CODE IN DOWNLOADABLE FORMAT CAN BE FOUND IN THE ZIP FOLDER ‘**AISHWARYA\_INTERVIEW\_ANSWERS.zip **‘.**

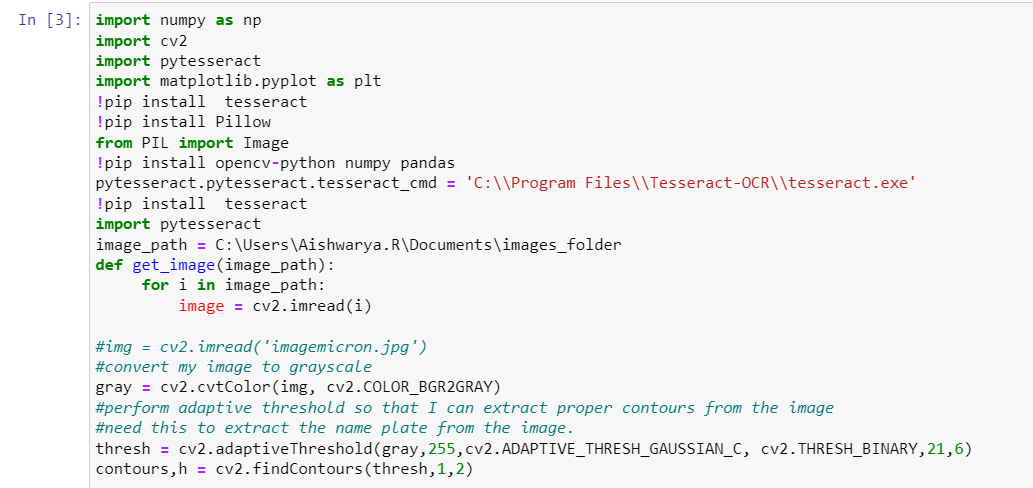
**AND ATTACHED WITH THE MAIL AND MY GITHUB LINK:** <https://github.com/aishwaryar5/INTERVIEW_ANSWERS_AISHWARYA.git> **]**

* **STEPS / PROCESS INVOLVED AND THE PYTHON CODES:**

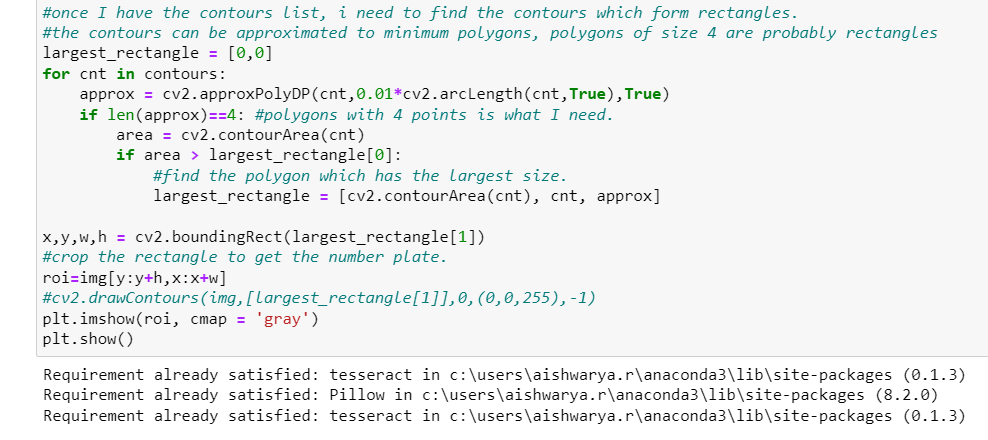
**STEP 1: CREATE A READ ME FOR EASIER UNDERSTANDING OF CODE AND IMPORT THE NECESSARY LIBRARIES:**



**STEP 2: INSTALL TESSERACT, PILLOW, IMPORT CV2, MATPLLOTLIB etc. WHICH ARE REQUIRED TO EXTRACT IMAGE AS A TEXT CONVERT IMAGE TO GRAY SCALE FOR PROCESSING:**

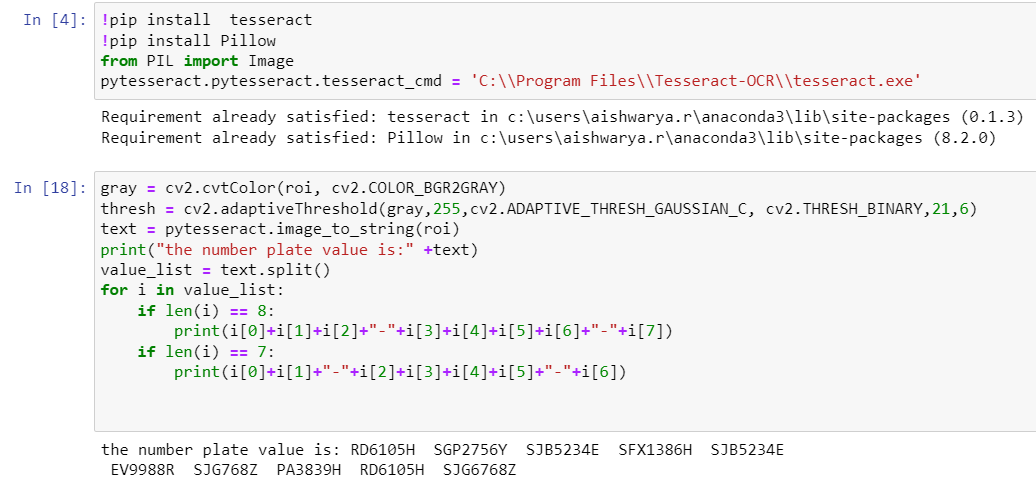


**STEP 3: DETERMINE THE CONTOUR BOUNDING BOX USING COMPUTER VISION CV2 AND USE IT TO OBTAIN THE IMAGE ROI, THEN PRINT IMAGE AFTER THE BOUNDING BOX DETERMINATION AND APPLYING GREY SCALE, OBTAIN THE IMAGE OUTPUT USING IMSHOW MATPLOT FUNCTION**

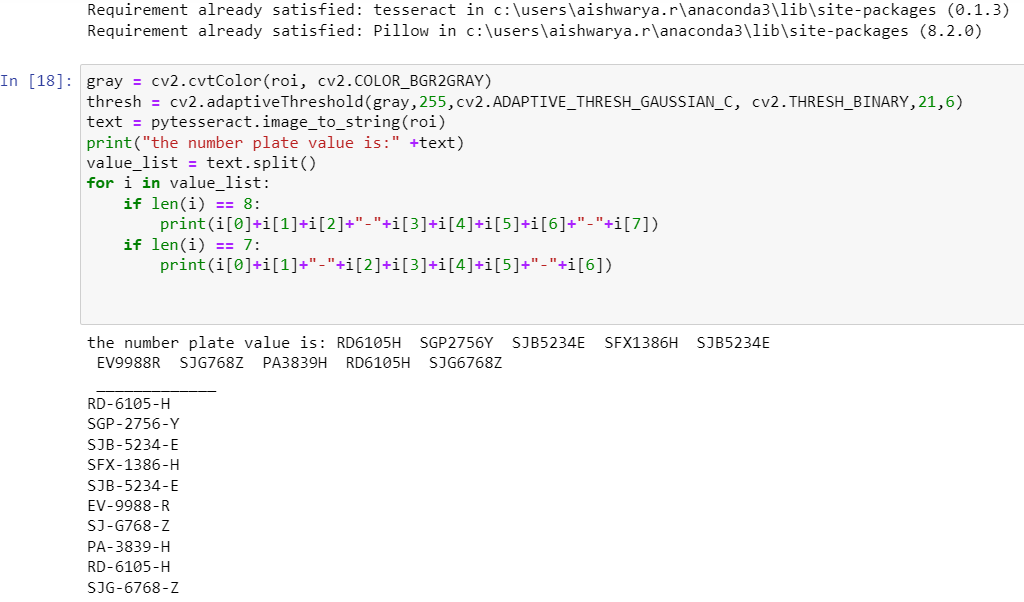




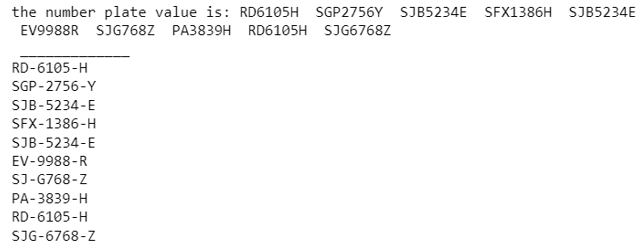
**STEP 4: INSTALL** **TESSERACT AND ALL THE RELATED TESSERACT LIBRARIES IN THE BASE ENVIRONMENT AND USE PY-TESSERACT ALONG WITH IMAGE ROI TO CONVERT THE IMAGE TO STRING.**



**STEP 5: OBSERVE THE PATTERNS IN THE NUMBER PLATE AND USE A BASIC LOOP FUNCTION TO SPLIT IT AS: as [XXX]-[0000]-[X] (I USED A VERY SIMPLE LOOP FUNCTION TO SPLIT STRING AS ABOVE, IF WE HAVE DETAILED PATTERN INFORMATION OF ALL POSSIBLE NUMBER PLATE PATTERN, I CAN CREATE A SIMPLE REGEX TO FIT ANY NUMBER PLATE VALUE)**



**OUTPUT OBTAINED:**



* **RESULTS:** THE OUTPUT DISPLAYED SHOWS THE NUMBER PLATE VALUES OF THE CAR CONVERTED FROM IMAGE TO TEXT USING TESSERACT
* **INSIGHTS /CONCLUSIONS:** WE USE TESSERACT AND ROI VALUE USING MATPLOTLIB AND CV2 TO EXTRACT TEXT FROM IMAGE
* **FURTHER SCOPE:** I JUST USED A SIMPLE FUNCTION TO SPLIT STRING; I CAN ALSO APPLY A REGEX TO MAKE CODE MORE COHERENT (NEED BASIC PATTERN OF NUMBER PLATE VALUE)

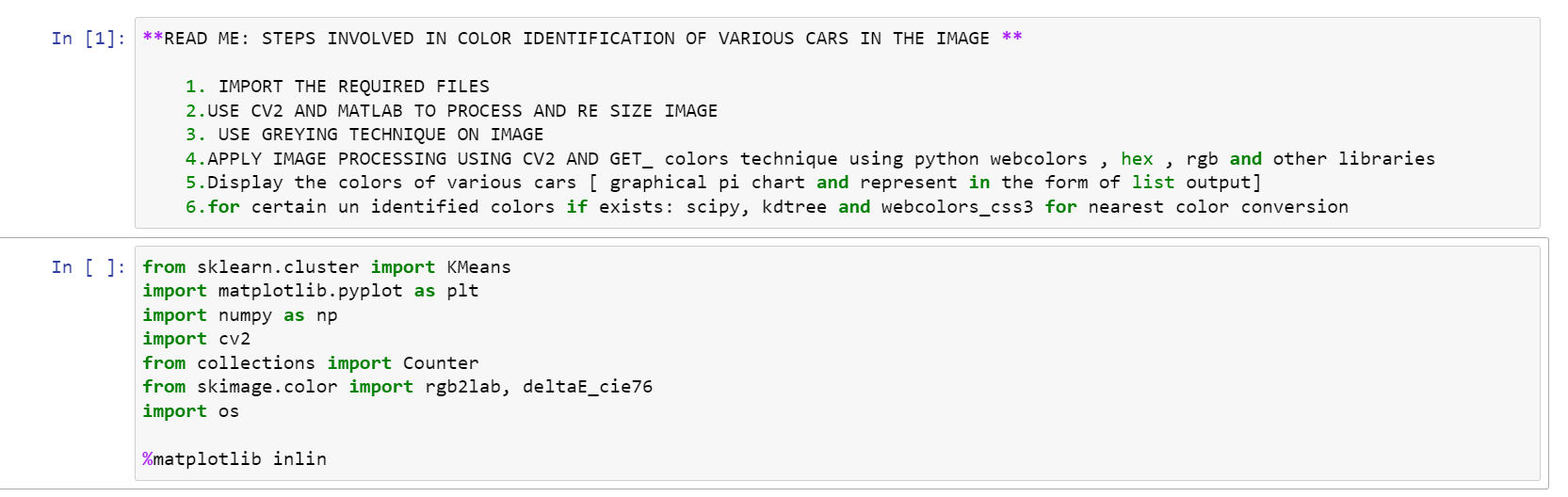
1. **IDENTIFYING AND EXRACTING THE CAR COLORS FROM IMAGE:**

**[ ENTIRE CODE IN DOWNLOADABLE FORMAT CAN BE FOUND IN THE ZIP FOLDER ‘**AISHWARYA\_INTERVIEW\_ANSWERS.zip **‘.**

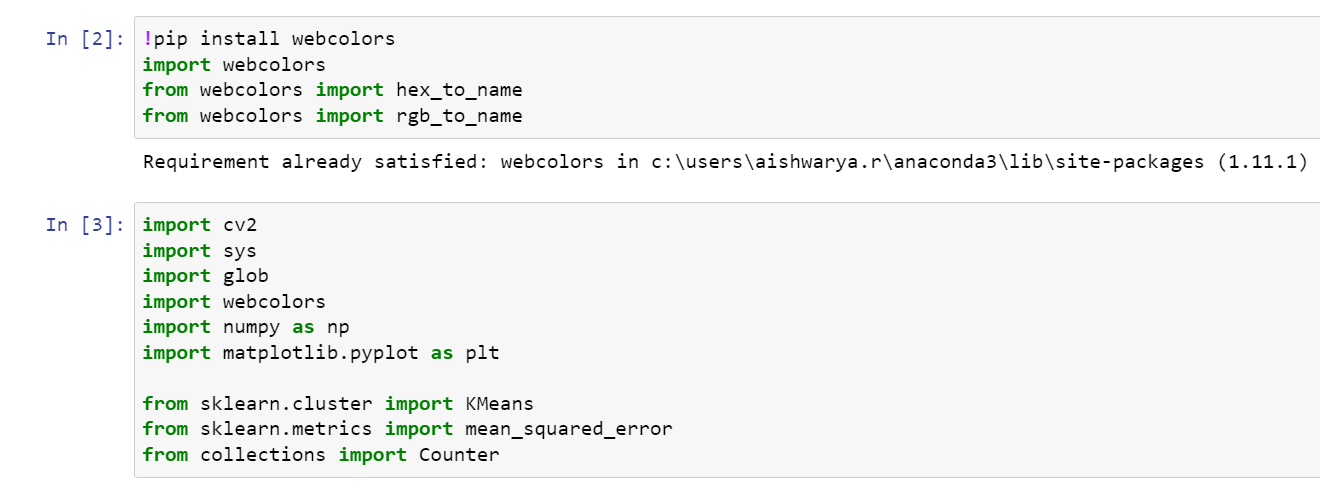
**AND ATTACHED WITH THE MAIL AND MY GITHUB LINK:** <https://github.com/aishwaryar5/INTERVIEW_ANSWERS_AISHWARYA.git> **]**

* **STEPS / PROCESS INVOLVED AND THE PYTHON CODES:**

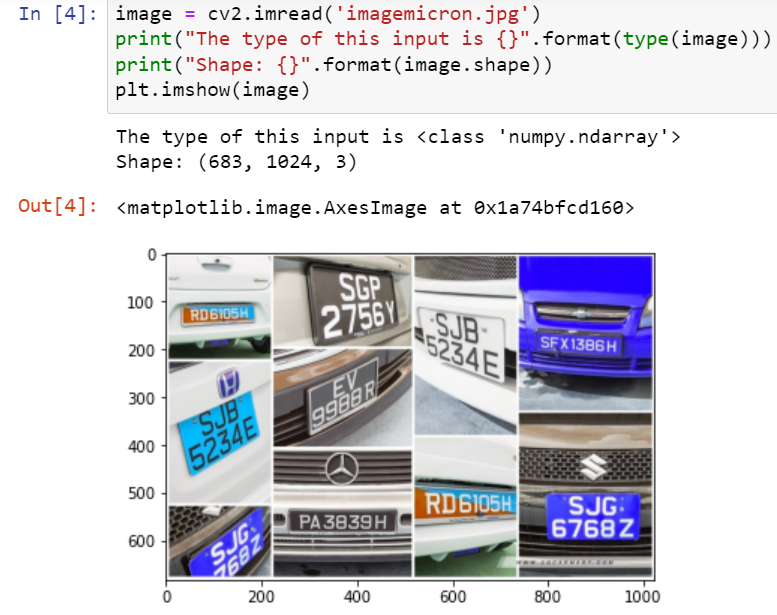
**STEP 1: IMPORT THE NECESSARY LIBRARIES:**



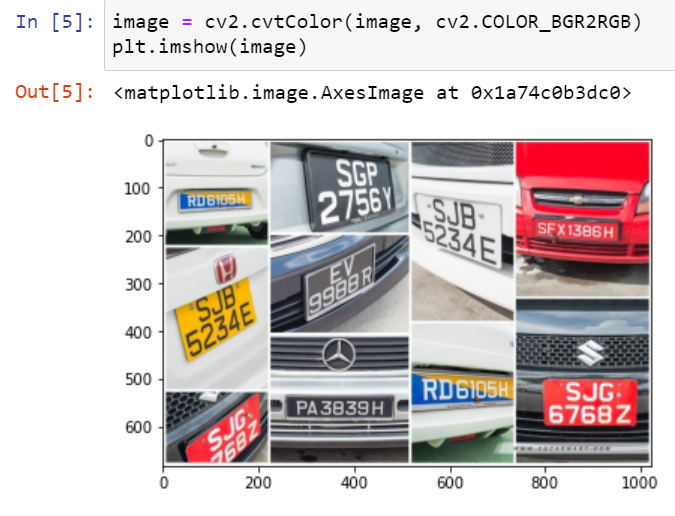
**STEP 2: INSTALL NECESSARY PACKAGES LIKE WEBCOLORS:**



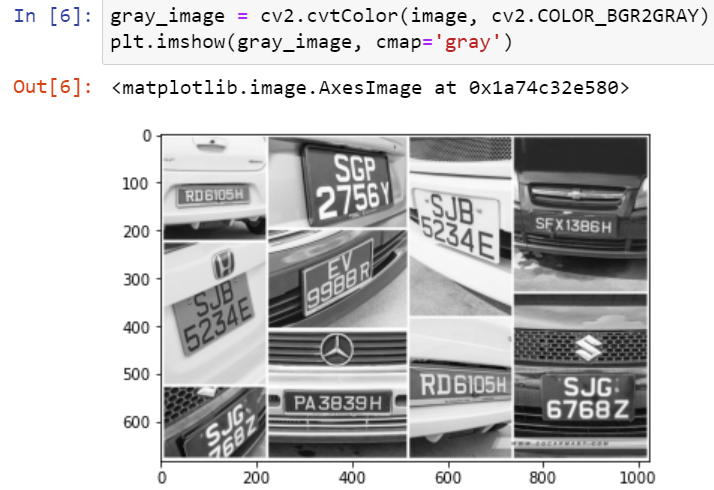
**STEP 3: USE CV2 AND MATPLOTLIB TO EXTRACT AND FORMAT THE IMAGE:**



**STEP 4: DISPLAY IMAGE AND CV2 AXIS AFTER FORMATTING THE IMAGE:**



**STEP 5: DISPLAY IMAGE AFTER CONVERTING THE IMAGE TO CV2.GREY FOR INITIAL IMAGE PROCESSING FOR COLOR IDENTIFICATION:**



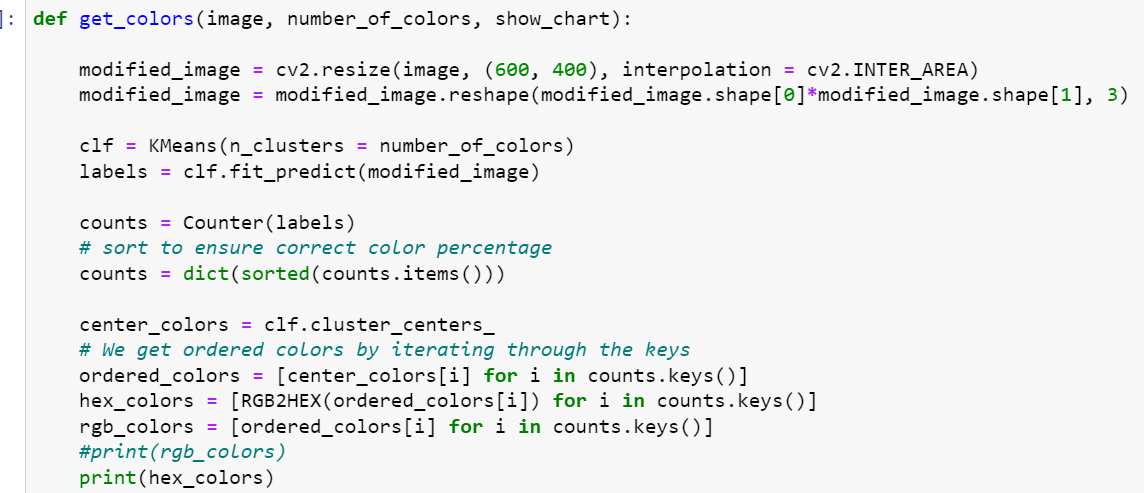
**STEP 6: DISPLAY IMAGE AFTER CONVERTING THE IMAGE AND RESIZING IMAGE AXES TO USE WEBCOLORS AND EXTRACT COLOR INFORMATION:**

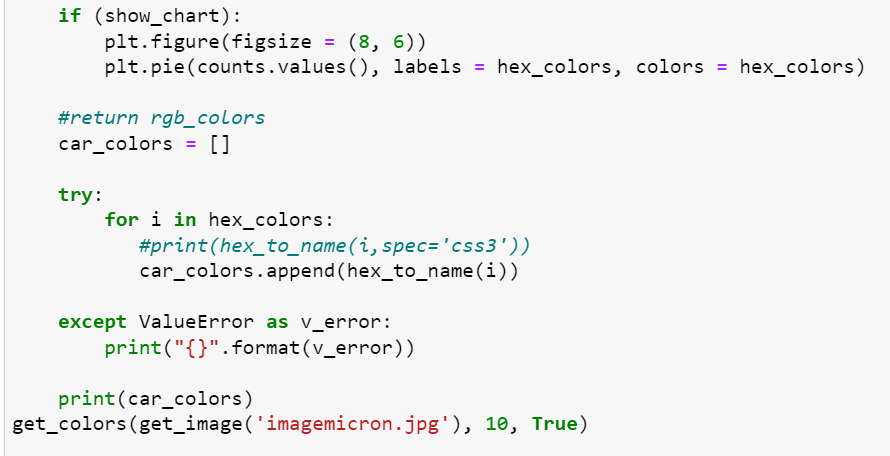


**STEP 7: DEFINING FUNCTION TO RETRIEVE AND CONVERT IMAGE COLOR VALUE IN ‘RGB’ AND ‘HEX’ COLOR CODE FORMAT AND LATER CONVERT IT TO ENGLISH COLOR NAMES:**

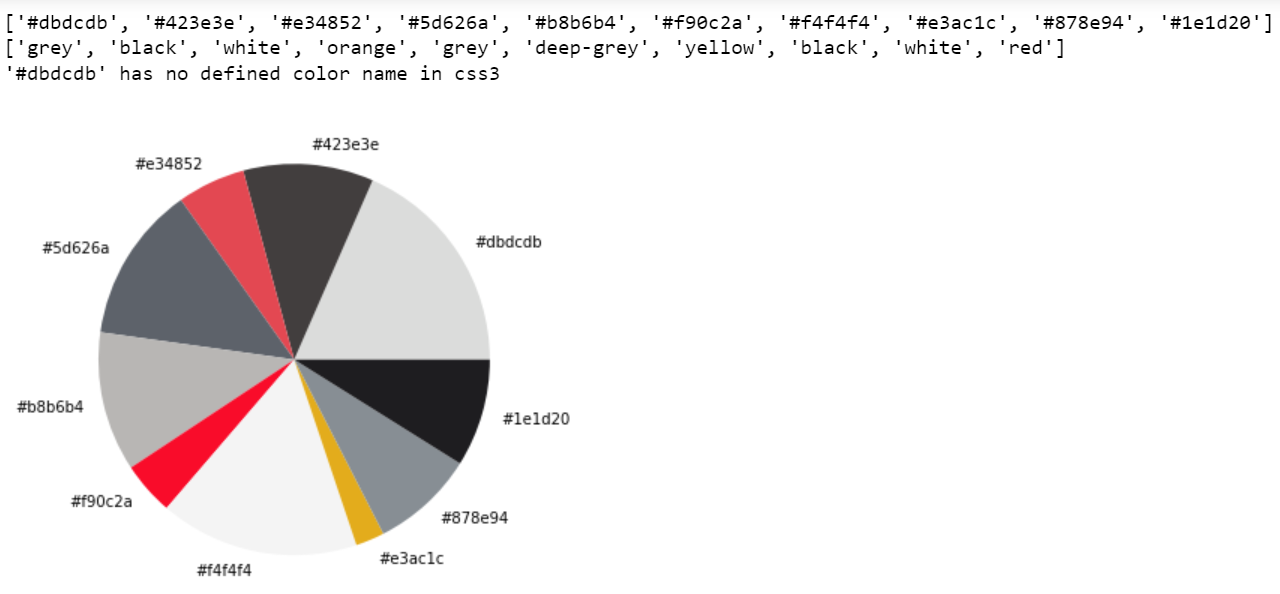


**STEP 8: DEFINING FUNCTION TO EXTRACT THE COLOR INFORMATION AND STORING THEM IN A LIST AND DISPLAYING THEM IN THE FORM OF A PIE CHART FOR VISUAL PURPOSES, USING WEBCOLORS, KMEANS AND CV2:**



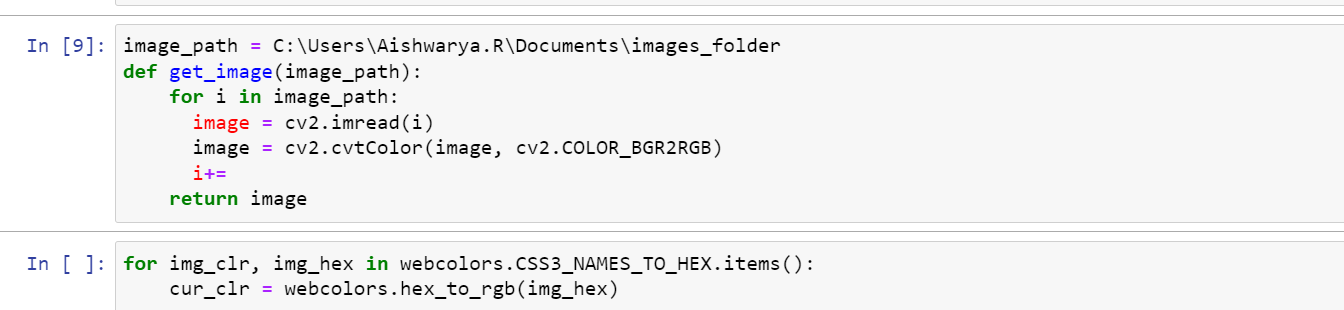


**OUTPUT OBTAINED:**



* **RESULTS:** THE OUTPUT DISPLAYED [USING STEP 8] SHOWS THE COLORS OF DIFFERENT CARS PRESENT IN THE IMAGE IN HEX AND WORD-CONVERTED FORMAT ALONG WITH THE PIE CHART FOR GRAPHICAL\VISUAL DISPLAY OF COLORS.
* **INSIGHTS /CONCLUSIONS:** THECOLOURS ARE EXTRACTED WITH AROUND 90% ACCURACY FOR ALL 10 CARS IN THE IMAGE. THE RESULTS OBTAINED WERE PRETTY ACCURATE WHEN COMPARED WITH THE ACTUAL IMAGE INPUT.
* **FURTHER SCOPE:** SINCE THE IMAGE OF ONE CAR WAS TOO SALL AND NUMBER PLATE WAS MUCH BIGGER THE COLOUR OF THE NUMBER PLATE IS DISPLAYED AS COLOUR OF CAR, NEED TO WORK ON EXCEPTIONAL CASES WHERE SIZE OF CAR IS SMALLER THAN NUMBER PLATE

**STEP 9: CODE OPTIMISATION 🡪 [FORMATTING FUNCTION TO AVOID HARDCODING AND CALLING MULTIPLE IMAGES IN CASE OF FUTURE USE]**



**ALTERNATIVE METHODS\CODES THAT WERE TRIED:**

[Complete codes in GitHub link provided above]

