Python Project: Used-Car Data

Team Members

Bhumika Suvagia

Ankita Savaliya

MSIS, California State University, Los Angeles

5270: Business Intelligence, Spring 2023

Professor Dr. Shilpa Balan

May 16th, 2023

Table of Content

A. Introduction

The topic that we have chosen from Kaggle for our project is Used-car Data as we know that the used car industry in the United States is a thriving market that continues to grow. According to the "Used Car Dealers Industry in the US - Market Research Report (March 23, 2023)."[1] the industry is driven by increasing consumer demand for affordable and reliable vehicles. As consumers seek alternatives to high-priced new cars, the used car market presents an attractive option. This market research report provides valuable insights into the industry, including market trends, competitive analysis, and growth projections. In addition, the article titled "USED CAR SALES DATA: WHERE HAVE WE BEEN AND WHERE ARE WE GOING? (June 24, 2022)."[2] sheds light on the trajectory of used car sales in the United States. It highlights the patterns and shifts observed in the market over time, offering valuable information for understanding the past and potential future developments in the industry. This data-driven analysis helps us comprehend the dynamics and trends that shape the used car market.

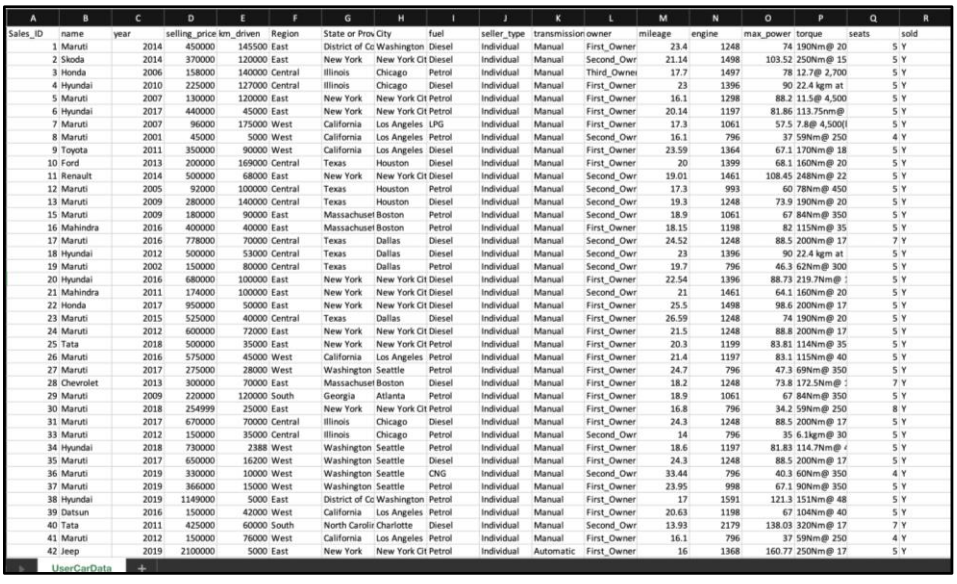
Moreover, with consumers increasingly looking for affordable options, the used car market is becoming more popular. However, for many buyers new to the used car market, there can be confusion about what to expect. Our analysis of the "Used-Car Data" dataset aims to provide valuable insights into the used car market in the US between 1994 and 2020. By answering questions such as the top 10 cars available for sale, the number of cars listed for sale in different cities, the total number of used cars sold, and the variation in average maximum power across top cities based on fuel type and transmission, we aim to gain a comprehensive understanding of the market dynamics. Additionally, we will explore the percentage of used cars sold by individuals versus dealers. This analysis will provide valuable information for individuals, businesses, and researchers interested in the used car market, enabling them to make informed decisions, identify market trends, and uncover opportunities in this thriving industry.

The records contain a wide variety of information about various used vehicles, including name of the used car, year of the car purchased, selling price, mileage, and fuel type. This rich data set can be used for a variety of purposes, including market analysis, trend detection, and price forecasting. Researchers, analysts, and companies can use this data to gain insight into the used car market and make informed decisions.

One of the main advantages of this dataset is its diversity. It contains information on used cars of various makes, models, and years, making it an ideal resource for understanding the dynamics of the used car market. The dataset can also be used to build machine learning models that can predict used car sales prices based on various factors such as model, year, and mileage.

Overall, the Kaggle used car dataset is a valuable resource for anyone interested in the used car market. Its comprehensive and diverse nature makes it an ideal tool for analysis and modeling, and its potential uses are numerous.

The below screenshot shows our dataset.



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
Sales_ID	name	year	selling_price	km_driven	Region	State or Prov	City	fuel	seller_type	transmission	owner	mileage	engine	max_power	torque	seats	sold
1	Maruti	2014	450000	145500	East	District of Co	Washington	Diesel	Individual	Manual	First_Owner	23.4	1248	74	190Nm@ 20	5	Y
2	Skoda	2014	370000	120000	East	New York	New York Cit	Diesel	Individual	Manual	Second_Owr	21.14	1498	103.52	250Nm@ 15	5	Y
3	Honda	2006	158000	140000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Owner	17.7	1497	78	12.7@ 2,700	5	Y
4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	23	1396	90	22.4 kgm at	5	Y
5	Maruti	2007	130000	120000	East	New York	New York Cit	Petrol	Individual	Manual	First_Owner	16.1	1298	88.2	11.5@ 4,500	5	Y
6	Hyundai	2017	440000	45000	East	New York	New York Cit	Petrol	Individual	Manual	First_Owner	20.14	1197	81.86	113.75nm@	5	Y
7	Maruti	2007	96000	175000	West	California	Los Angeles	LPG	Individual	Manual	First_Owner	17.3	1061	57.5	7.8@ 4,500(I	5	Y
8	Maruti	2001	45000	5000	West	California	Los Angeles	Petrol	Individual	Manual	Second_Owr	16.1	796	37	59Nm@ 250	4	Y
9	Toyota	2011	350000	90000	West	California	Los Angeles	Diesel	Individual	Manual	First_Owner	23.59	1364	67.1	170Nm@ 18	5	Y
10	Ford	2013	200000	169000	Central	Texas	Houston	Diesel	Individual	Manual	First_Owner	20	1399	68.1	160Nm@ 20	5	Y
11	Renault	2014	500000	68000	East	New York	New York Cit	Diesel	Individual	Manual	Second_Owr	19.01	1461	108.45	248Nm@ 22	5	Y
12	Maruti	2005	92000	100000	Central	Texas	Houston	Petrol	Individual	Manual	Second_Owr	17.3	993	60	78Nm@ 450	5	Y
13	Maruti	2009	280000	140000	Central	Texas	Houston	Diesel	Individual	Manual	Second_Owr	19.3	1248	73.9	190Nm@ 20	5	Y
15	Maruti	2009	180000	90000	East	Massachuset	Boston	Petrol	Individual	Manual	Second_Owr	18.9	1061	67	84Nm@ 350	5	Y
16	Mahindra	2016	400000	40000	East	Massachuset	Boston	Petrol	Individual	Manual	First_Owner	18.15	1198	82	115Nm@ 35	5	Y
17	Maruti	2016	778000	70000	Central	Texas	Dallas	Diesel	Individual	Manual	Second_Owr	24.52	1248	88.5	200Nm@ 17	7	Y
18	Hyundai	2012	500000	53000	Central	Texas	Dallas	Diesel	Individual	Manual	Second_Owr	23	1396	90	22.4 kgm at	5	Y
19	Maruti	2002	150000	80000	Central	Texas	Dallas	Petrol	Individual	Manual	Second_Owr	19.7	796	46.3	62Nm@ 300	5	Y
20	Hyundai	2016	680000	100000	East	New York	New York Cit	Diesel	Individual	Manual	First_Owner	22.54	1396	88.73	219.7Nm@ 1	5	Y
21	Mahindra	2011	174000	100000	East	New York	New York Cit	Diesel	Individual	Manual	Second_Owr	21	1461	64.1	160Nm@ 20	5	Y
22	Honda	2017	950000	50000	East	New York	New York Cit	Diesel	Individual	Manual	First_Owner	25.5	1498	98.6	200Nm@ 17	5	Y
23	Maruti	2015	525000	40000	Central	Texas	Dallas	Diesel	Individual	Manual	First_Owner	26.59	1248	74	190Nm@ 20	5	Y
24	Maruti	2012	600000	72000	East	New York	New York Cit	Diesel	Individual	Manual	First_Owner	21.5	1248	88.8	200Nm@ 17	5	Y
25	Tata	2018	500000	35000	East	New York	New York Cit	Petrol	Individual	Manual	First_Owner	20.3	1199	83.81	114Nm@ 35	5	Y
26	Maruti	2016	575000	45000	West	California	Los Angeles	Petrol	Individual	Manual	First_Owner	21.4	1197	83.1	115Nm@ 40	5	Y
27	Maruti	2017	275000	28000	West	Washington	Seattle	Petrol	Individual	Manual	First_Owner	24.7	796	47.3	69Nm@ 350	5	Y
28	Chevrolet	2013	300000	70000	East	Massachuset	Boston	Diesel	Individual	Manual	First_Owner	18.2	1248	73.8	172.5Nm@ 1	7	Y
29	Maruti	2009	220000	120000	South	Georgia	Atlanta	Petrol	Individual	Manual	First_Owner	18.9	1061	67	84Nm@ 350	5	Y
30	Maruti	2018	254999	25000	East	New York	New York Cit	Petrol	Individual	Manual	First_Owner	16.8	796	34.2	59Nm@ 250	8	Y
31	Maruti	2017	670000	70000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	24.3	1248	88.5	200Nm@ 17	5	Y
33	Maruti	2012	150000	35000	Central	Illinois	Chicago	Petrol	Individual	Manual	Second_Owr	14	796	35	6.1kgm@ 30	5	Y
34	Hyundai	2018	730000	2388	West	Washington	Seattle	Petrol	Individual	Manual	First_Owner	18.6	1197	81.83	114.7Nm@ 4	5	Y
35	Maruti	2017	650000	16200	West	Washington	Seattle	Diesel	Individual	Manual	First_Owner	24.3	1248	88.5	200Nm@ 17	5	Y
36	Maruti	2019	330000	10000	West	Washington	Seattle	CNG	Individual	Manual	Second_Owr	33.44	796	40.3	60Nm@ 350	4	Y
37	Maruti	2019	366000	15000	West	Washington	Seattle	Petrol	Individual	Manual	First_Owner	23.95	998	67.1	90Nm@ 350	5	Y
38	Hyundai	2019	1149000	5000	East	District of Co	Washington	Petrol	Individual	Manual	First_Owner	17	1591	121.3	151Nm@ 48	5	Y
39	Datsun	2016	150000	42000	West	California	Los Angeles	Petrol	Individual	Manual	First_Owner	20.63	1198	67	104Nm@ 40	5	Y
40	Tata	2011	425000	60000	South	North Caroli	Charlotte	Diesel	Individual	Manual	Second_Owr	13.93	2179	138.03	320Nm@ 17	7	Y
41	Maruti	2012	150000	76000	West	California	Los Angeles	Petrol	Individual	Manual	First_Owner	16.1	796	37	59Nm@ 250	4	Y
42	Jeep	2019	2100000	5000	East	New York	New York Cit	Petrol	Individual	Automatic	First_Owner	16	1368	160.77	250Nm@ 17	5	Y

UserCarData

B. Data set URL's and Data set Description:

Dataset URL: https://www.kaggle.com/datasets/shubham1kumar/usedcar-data?select=UserCarData.csv/

Dataset Description:

The Used-Car Data Dataset is a comprehensive collection of data related to the used car market in the USA between 1994 to 2020. The dataset consists of 18 columns, including the name of the used car, year of purchase, selling price, total km driven, region, state or province, city, fuel type, seller type, transmission type, owner type, mileage, engine power, max power, number of seats, and whether the used car is sold or not.

This dataset is an ideal resource for understanding the dynamics of the used car market, and can be used for various purposes, including market analysis, trend detection, and price forecasting. With its diverse range of information, it is an excellent tool for researchers, analysts, and companies to gain insight into the used car market and make informed decisions.

This dataset contains 18 columns, and only 17 columns will be useful for our python project.

The below table describes the data set in detail.

Data Field	Description	Example Values
Sales_ID	Unique identifier for each sale. Data type: Integer	145500,120000,140000, etc.
Name	The name of the used car. Data type: String	Maruti, Hyundai, Skoda, Honda, etc.
Year	The year in which the car was purchased. Data type: Integer	From 1994 to 2020

Selling_Price	The current selling price of the used car. Data type: Float	From 30K to 10 million
Km_Driven	The total distance driven by the car is in kilometers. Data type: Integer	Between 1 and 1500000 mi
Region	The region where the car is used. Data type: String	Central, West and others
State or Province	The state or province where the car is used. Data type: String	California, Texas, and others
City	The city where the car is used. Data type: String	New York city, Los Angeles, and others
Fuel	The type of fuel used by the car. Data type: String.	Diesel, Petrol, and others
Seller_Type	The type of seller Data type: String	Individual, Dealer, and others
Transmission	The type of transmission. Data type: String	Manual and Automatic
Owner	The type of ownership. Data type: String	First_Owner, Second Owner, and others
Mileage	The mileage of the car in kilometers per liter or miles per gallon. Data type: float.	From 0 to 34

Engine	The engine displacement is cubic centimeters. Data type: integer.	Between 624 to 3604
Max_Power	The maximum power output of the car's engine is in horsepower or kilowatts. Data type: float.	From 32.8 to 400
Seats	The number of seats in the car. Data type: integer.	Between 2 to 14
Sold	It indicates whether the used car is sold or not. Data type: string	True and False

C. Data Cleaning:

1) Removing the duplicate rows:

[Tool: Jupyter Notebook]

In our data cleaning process, the initial step involved removing duplicate rows. We observed that a few rows in our dataset were repeated, and it is crucial to remove duplicate values for the true analysis. And another reason was that this action ensures data consistency and enables us to obtain precise insights, which in turn supports informed decision-making based on the data.

Before Cleaning:

	Sales_ID	name	year	selling_price	km_driven	Region	State or Province	City	fuel	seller_type	transmission	owner	mileage	engine	max_power	torque	seats	sold
0	1	Maruti	2014	450000	145500	East	District of Columbia	Washington	Diesel	Individual	Manual	First_Owner	23.40	1248	74.00	190Nm@ 2000rpm	5	Y
1	2	Skoda	2014	370000	120000	East	New York	New York City	Diesel	Individual	Manual	Second_Owner	21.14	1498	103.52	250Nm@ 1500-2500rpm	5	Y
2	3	Honda	2006	158000	140000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Owner	17.70	1497	78.00	12.7@ 2,700(kgm@ rpm)	5	Y
3	4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	23.00	1396	90.00	22.4 kgm at 1750-2750rpm	5	Y
4	5	Maruti	2007	130000	120000	East	New York	New York City	Petrol	Individual	Manual	First_Owner	16.10	1298	88.20	11.5@ 4,500(kgm@ rpm)	5	Y

```
new_dataset.shape

new_dataset.Sales_ID.duplicated()

new_dataset.Sales_ID.duplicated().sum()

new_dataset.loc[new_dataset.duplicated(keep=False), :]

new_dataset = new_dataset.drop_duplicates(keep='last')

new_dataset.head()

new_dataset.shape
```

In order to check how many rows and columns in our dataset, we run the following command. After running this command, we came to know that we have 7912 rows and 17 columns in our dataset.

```
new_dataset.shape
```

```
(7912, 17)
```

To check how many duplicates rows in our dataset, we used sum () function.

```
new_dataset.Sales_ID.duplicated().sum()
```

```
6
```

Also, we wanted to see that how many duplicates rows are there in our dataset, and when we used the duplicated() method, it provided true and false values where true indicates the duplicate rows whereas false indicates no duplicates.

```
new_dataset.duplicated()
```

```
0       False
1       False
2       False
3       False
4       False
        ...
7907     True
7908     True
7909     True
7910     True
7911     True
Length: 7912, dtype: bool
```

In order to see which rows are duplicated, we use the loc [] method that only shows the duplicates rows.

	Sales_ID	name	year	selling_price	km_driven	Region	State or Province	City	fuel	seller_type	transmission	owner	mileage	engine	max_power	seats	sold
7900	8123	Hyundai	2014	475000	80000	Central	Texas	Plano	Diesel	Individual	Manual	Second_Owner	22.54	1396	88.73	5	N
7901	8124	Hyundai	2013	320000	110000	Central	Texas	Plano	Petrol	Individual	Manual	First_Owner	18.50	1197	82.85	5	N
7902	8125	Hyundai	2007	135000	119000	Central	Texas	Port Arthur	Diesel	Individual	Manual	Fourth_Above_Owner	16.80	1493	110.00	5	N
7903	8126	Maruti	2009	382000	120000	Central	Texas	Port Arthur	Diesel	Individual	Manual	First_Owner	19.30	1248	73.90	5	N
7904	8127	Tata	2013	290000	25000	East	Massachusetts	Lunenburg	Diesel	Individual	Manual	First_Owner	23.57	1396	70.00	5	N
7905	8128	Tata	2013	290000	25000	East	Maine	Augusta	Diesel	Individual	Manual	First_Owner	23.57	1396	70.00	5	N
7906	8123	Hyundai	2014	475000	80000	Central	Texas	Plano	Diesel	Individual	Manual	Second_Owner	22.54	1396	88.73	5	N
7907	8124	Hyundai	2013	320000	110000	Central	Texas	Plano	Petrol	Individual	Manual	First_Owner	18.50	1197	82.85	5	N
7908	8125	Hyundai	2007	135000	119000	Central	Texas	Port Arthur	Diesel	Individual	Manual	Fourth_Above_Owner	16.80	1493	110.00	5	N
7909	8126	Maruti	2009	382000	120000	Central	Texas	Port Arthur	Diesel	Individual	Manual	First_Owner	19.30	1248	73.90	5	N

After Cleaning:

We successfully utilized the drop_duplicates() function to remove duplicate rows from the dataset. Only 7906 rows were left after removing the duplicates.

```
new_dataset = new_dataset.drop_duplicates(keep='last')
```

```
new_dataset.head()
```

	SALES_ID	NAME	YEAR	SELLING_PRICE	KM_DRIVEN	REGION	STATE OR PROVINCE	CITY	FUEL	SELLER_TYPE	TRANSMISSION	OWNER	MILEAGE	ENGINE	MAX_POWER	SEATS	SOLD
0	1	Maruti	2014	450000	145500	East	District of Columbia	Washington	Diesel	Individual	Manual	First_Owner	23.40	1248	74.00	5	Y
1	2	Skoda	2014	370000	120000	East	New York	New York City	Diesel	Individual	Manual	Second_Owner	21.14	1498	103.52	5	Y
2	3	Honda	2006	158000	140000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Owner	17.70	1497	78.00	5	Y
3	4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	23.00	1396	90.00	5	Y
4	5	Maruti	2007	130000	120000	East	New York	New York City	Petrol	Individual	Manual	First_Owner	16.10	1298	88.20	5	Y

```
new_dataset.shape
```

```
(7906, 17)
```

2) Removing extra space in the text:

During the data cleaning process, we identified the second category of tasks, which involved removing extra spaces in the text. Specifically, we noticed that there were extra spaces in the city names within the "City" column. The purpose of removing these extra spaces was to ensure consistency, maintain data integrity, improve formatting, and enhance efficiency. By eliminating the extra spaces, we aimed to achieve more accurate and reliable analysis results.

Before cleaning:

Firstly, we checked which column had the extra spaces between the text using the for loop which provided the column name that was “City” column as you can see from the below code output.

After identifying the column, we replaced it with single space using replace function.

```
string_cols = new_dataset.select_dtypes(include='object').columns

extra_space_cols = [col for col in string_cols if new_dataset[col].str.contains(r'\s\s+').any()]

print(extra_space_cols)
['City']

# Replace extra spaces with single space in 'City' column
new_dataset['City'] = new_dataset['City'].str.replace(r'\s\s+', ' ', regex=True)

new_dataset.to_csv('UserCarDataset.csv', index=False)
```

```
string_cols = new_dataset.select_dtypes(include='object').columns
extra_space_cols = [col for col in string_cols if new_dataset[col].str.contains(r'\s\s+').any()]
```

```
print(extra_space_cols)
# Replace extra spaces with single space in 'City' column
new_dataset['City'] = new_dataset['City'].str.replace(r'\s\s+', ' ', regex=True)
new_dataset.to_csv('UserCarDataset.csv', index=False)
new_dataset.head()
print(extra_space_cols)
```

After cleaning:

```
new_dataset.head()
```

	Sales_ID	name	year	selling_price	km_driven	Region	State or Province	City	fuel	seller_type	transmission	owner	mileage	engine	max_power	seats	sold
0	1	Maruti	2014	450000	145500	East	District of Columbia	Washington	Diesel	Individual	Manual	First_Owner	23.40	1248	74.00	5	Y
1	2	Skoda	2014	370000	120000	East	New York	New York City	Diesel	Individual	Manual	Second_Owner	21.14	1498	103.52	5	Y
2	3	Honda	2006	158000	140000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Owner	17.70	1497	78.00	5	Y
3	4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	23.00	1396	90.00	5	Y
4	5	Maruti	2007	130000	120000	East	New York	New York City	Petrol	Individual	Manual	First_Owner	16.10	1298	88.20	5	Y

```
print(extra_space_cols)
[]
```

3) Renaming the columns name:

During the data cleaning process, we encountered the third category, which involved renaming columns name in our dataset. We noticed inconsistencies in the columns name, with some being in uppercase and others in lowercase. To prevent errors when creating graphs in the spider, it was crucial to ensure that the column names were referenced correctly. By renaming the columns, we aimed to standardize the naming conventions, making it easier to work with the data and avoid potential mistakes.

Before Cleaning:

In order to rename the column names in our dataset, we used “str.upper()” method.

	Sales_ID	name	year	selling_price	km_driven	Region	State or Province	City	fuel	seller_type	transmission	owner	mileage	engine	max_power	seats	sold
0	1	Maruti	2014	450000	145500	East	District of Columbia	Washington	Diesel	Individual	Manual	First_Owner	23.40	1248	74.00	5	Y
1	2	Skoda	2014	370000	120000	East	New York	New York City	Diesel	Individual	Manual	Second_Owner	21.14	1498	103.52	5	Y
2	3	Honda	2006	158000	140000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Owner	17.70	1497	78.00	5	Y
3	4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	23.00	1396	90.00	5	Y
4	5	Maruti	2007	130000	120000	East	New York	New York City	Petrol	Individual	Manual	First_Owner	16.10	1298	88.20	5	Y

```
new_dataset.columns = new_dataset.columns.str.upper()
new_dataset.to_csv('UserCarDataset.csv', index=False)
# Display the updated dataset
new_dataset.head()
```

```
new_dataset.columns = new_dataset.columns.str.upper()

new_dataset.to_csv('UserCarDataset.csv', index=False)

# Display the updated dataset
new_dataset.head()
```

After cleaning:

	SALES_ID	NAME	YEAR	SELLING_PRICE	KM_DRIVEN	REGION	STATE OR PROVINCE	CITY	FUEL	SELLER_TYPE	TRANSMISSION	OWNER	MILEAGE	ENGINE	MAX_POWER	SEATS	SOLD
0	1	Maruti	2014	450000	145500	East	District of Columbia	Washington	Diesel	Individual	Manual	First_Owner	23.40	1248	74.00	5	Y
1	2	Skoda	2014	370000	120000	East	New York	New York City	Diesel	Individual	Manual	Second_Owner	21.14	1498	103.52	5	Y
2	3	Honda	2006	158000	140000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Owner	17.70	1497	78.00	5	Y
3	4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	23.00	1396	90.00	5	Y
4	5	Maruti	2007	130000	120000	East	New York	New York City	Petrol	Individual	Manual	First_Owner	16.10	1298	88.20	5	Y

4) Replacing 0 with mean value:

As part of the fourth category of data cleaning, our initial approach was to create a scatter plot to examine the correlation between the selling price and the mileage of used cars. However, during the analysis, we observed that the "Mileage" column contained numerous 0 values. To ensure accurate and meaningful insights from the scatter plot, we decided to replace these 0 values with the mean value of the "Mileage" column.

Before cleaning:

```
zero_count = (used_car_dataset['MILEAGE'] == 0).sum()
print("Number of 0 values in the mileage column: ", zero_count)
```

```
Number of 0 values in the mileage column:  17
```

```
zero_count = (used_car_dataset['MILEAGE'] == 0).sum()
print("Number of 0 values in the mileage column: ", zero_count)
mileage_mean = used_car_dataset['MILEAGE'].mean()
print(mileage_mean)
new_mileage_mean = round(mileage_mean,2)
print(new_mileage_mean)
new_dataset['MILEAGE'] = new_dataset['MILEAGE'].replace(0, new_mileage_mean)
new_dataset['MILEAGE']
#count of number of mean values
mean_count = new_dataset['MILEAGE'].value_counts()[19.42]
print("Number of mean values in 'mileage'column is:", mean_count)
new_dataset.to_csv('UserCarDataset1.csv', index=False)
```

First, we calculated the mean value of “Mileage” column and round it seen as below.

```
mileage_mean = used_car_dataset['MILEAGE'].mean()

print(mileage_mean)
19.419860865165674

new_mileage_mean = round(mileage_mean,2)

print(new_mileage_mean)
19.42
```

Then we replaced 0 with mean value that we calculated earlier using replace method.

```
new_dataset['MILEAGE'] = new_dataset['MILEAGE'].replace(0, new_mileage_mean)
new_dataset['MILEAGE']
```

After cleaning:

```
#count of number of mean values
mean_count = new_dataset['MILEAGE'].value_counts()[19.42]
print("Number of mean values in 'mileage'column is:", mean_count)
Number of mean values in 'mileage'column is: 17
```

```
new_dataset.to_csv('UserCarDataset.csv', index=False)
```

5)Dropping unnecessary column:

As part of the data cleaning process, we identified the fifth category which involved dropping unnecessary columns from the dataset. These columns were deemed to be irrelevant and not useful for any analysis.

Note: This is not a part of the data cleaning category, but we performed this step, so we have added it here.

```
import pandas as pd

path_to_csv_file = r'/Users/bhumisuvagia/Documents/MIS/CIS5270/Project/UserCarData.csv'

df = pd.read_csv(path_to_csv_file)

df.head()
```

Before Cleaning:

	Sales_ID	name	year	selling_price	km_driven	Region	State or Province	City	fuel	seller_type	transmission	owner	mileage	engine	max_power	torque	seats	sold
0	1	Maruti	2014	450000	145500	East	District of Columbia	Washington	Diesel	Individual	Manual	First_Owner	23.40	1248	74.00	190Nm@ 2000rpm	5	Y
1	2	Skoda	2014	370000	120000	East	New York	New York City	Diesel	Individual	Manual	Second_Owner	21.14	1498	103.52	250Nm@ 1500-2500rpm	5	Y
2	3	Honda	2006	158000	140000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Owner	17.70	1497	78.00	12.7@ 2,700(kgm@ rpm)	5	Y
3	4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	23.00	1396	90.00	22.4 kgm at 1750-2750rpm	5	Y
4	5	Maruti	2007	130000	120000	East	New York	New York City	Petrol	Individual	Manual	First_Owner	16.10	1298	88.20	11.5@ 4,500(kgm@ rpm)	5	Y

```
new_dataset = df.drop(['torque'], axis=1)

new_dataset.head()
```

After Cleaning:

```
new_dataset.head()
```

	Sales_ID	name	year	selling_price	km_driven	Region	State or Province	City	fuel	seller_type	transmission	owner	mileage	engine	max_power	seats	sold
0	1	Maruti	2014	450000	145500	East	District of Columbia	Washington	Diesel	Individual	Manual	First_Owner	23.40	1248	74.00	5	Y
1	2	Skoda	2014	370000	120000	East	New York	New York City	Diesel	Individual	Manual	Second_Owner	21.14	1498	103.52	5	Y
2	3	Honda	2006	158000	140000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Owner	17.70	1497	78.00	5	Y
3	4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Owner	23.00	1396	90.00	5	Y
4	5	Maruti	2007	130000	120000	East	New York	New York City	Petrol	Individual	Manual	First_Owner	16.10	1298	88.20	5	Y

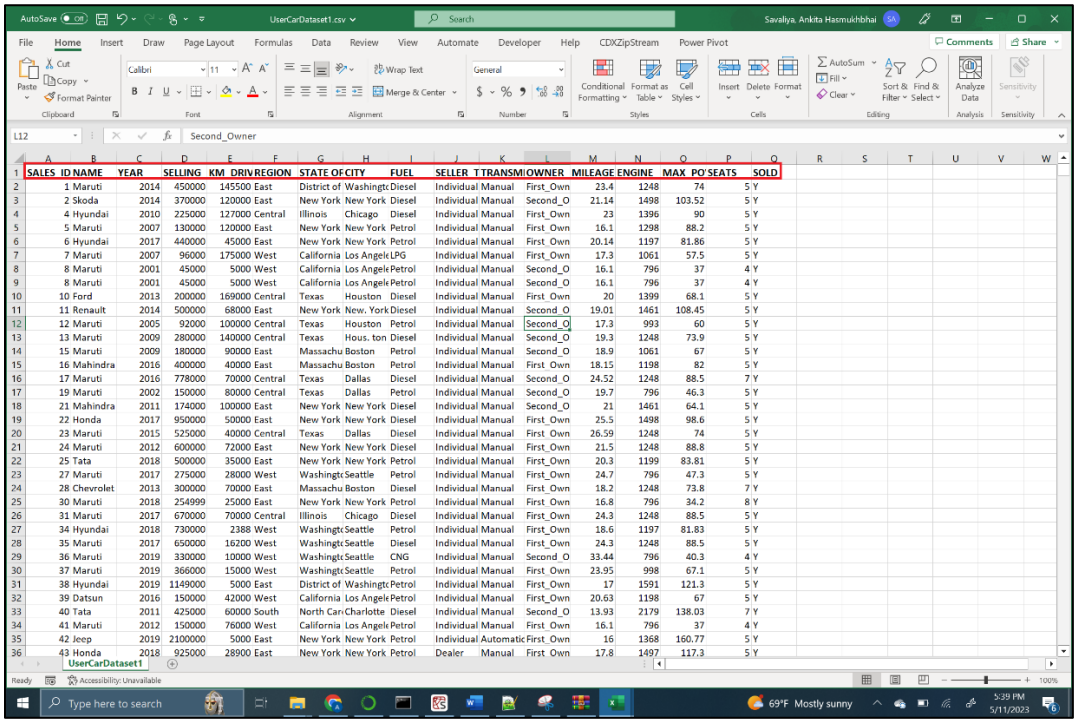
6)Mapping Latitude and Longitude file to the Dataset:

[Tool: Spider]

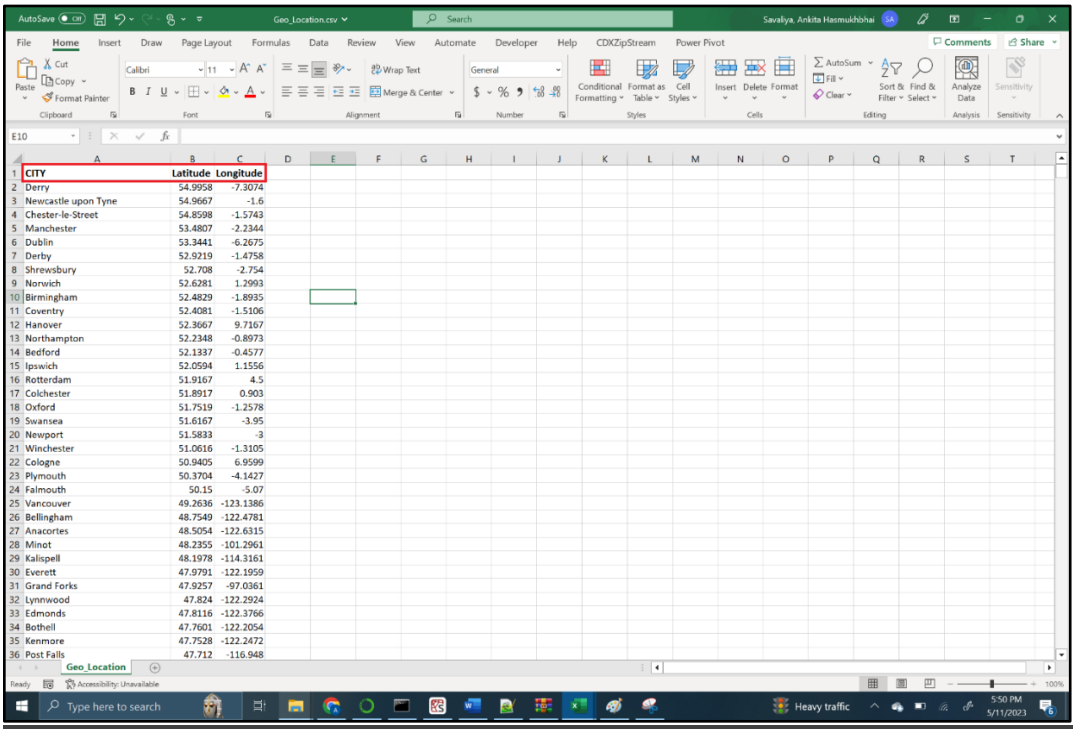
As part of the data cleaning process, the sixth step involved merging another dataset containing longitude and latitude information with the existing dataset. The primary objective of this step was to do the analysis of cars listed for sale in different cities. The purpose of this step was to enable the analysis of cars listed for sale in different cities. To accomplish this, a file named "Geo_Location.csv" was used to obtain the latitude and longitude values. This file contains information about cities, states or province, and their corresponding latitude and longitude coordinates. The join operation was performed between the two datasets using the common columns "City" and "State or Province" to ensure that the merging process avoids any duplication. By combining the datasets based on these columns, the resulting dataset would have incorporated the longitude and latitude information from the "Geo_Location.csv" file into the original dataset.

This merged dataset would have provided the necessary geographical coordinates for each car listing, enabling further analysis and exploration of the data based on location.

Before Cleaning:



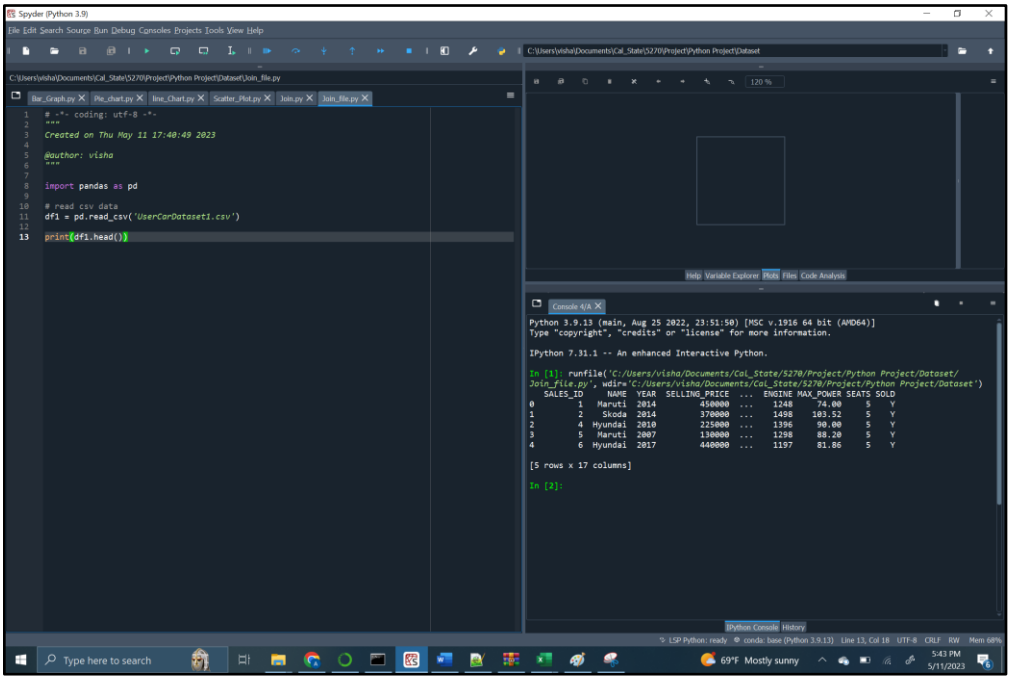
SALES_ID	NAME	YEAR	SELLING_	KM_DRIV	REGION	STATE OF	CITY	FUEL	SELLER_T	TRANSMI	OWNER	MILEAGE	ENGINE	MAX_PO	SEATS	SOLD
1	Maruti	2014	450000	145500	East	District of	Washingtc	Diesel	Individual	Manual	First_Own	23.4	1248	74	5	Y
2	Skoda	2014	370000	120000	East	New York	New York	Diesel	Individual	Manual	Second_O	21.14	1498	103.52	5	Y
4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	23	1396	90	5	Y
5	Maruti	2007	130000	120000	East	New York	New York	Petrol	Individual	Manual	First_Own	16.1	1298	88.2	5	Y
6	Hyundai	2017	440000	45000	East	New York	New York	Petrol	Individual	Manual	First_Own	20.14	1197	81.86	5	Y
7	Maruti	2007	96000	175000	West	California	Los Angele	LPG	Individual	Manual	First_Own	17.3	1061	57.5	5	Y
8	Maruti	2001	45000	5000	West	California	Los Angele	Petrol	Individual	Manual	Second_O	16.1	796	37	4	Y
8	Maruti	2001	45000	5000	West	California	Los Angele	Petrol	Individual	Manual	Second_O	16.1	796	37	4	Y
10	Ford	2013	200000	169000	Central	Texas	Houston	Diesel	Individual	Manual	First_Own	20	1399	68.1	5	Y
11	Renault	2014	500000	68000	East	New York	New. York	Diesel	Individual	Manual	Second_O	19.01	1461	108.45	5	Y
12	Maruti	2005	92000	100000	Central	Texas	Houston	Petrol	Individual	Manual	Second_O	17.3	993	60	5	Y
13	Maruti	2009	280000	140000	Central	Texas	Hous. ton	Diesel	Individual	Manual	Second_O	19.3	1248	73.9	5	Y
15	Maruti	2009	180000	90000	East	Massachu	Boston	Petrol	Individual	Manual	Second_O	18.9	1061	67	5	Y
16	Mahindra	2016	400000	40000	East	Massachu	Boston	Petrol	Individual	Manual	First_Own	18.15	1198	82	5	Y
17	Maruti	2016	778000	70000	Central	Texas	Dallas	Diesel	Individual	Manual	Second_O	24.52	1248	88.5	7	Y
19	Maruti	2002	150000	80000	Central	Texas	Dallas	Petrol	Individual	Manual	Second_O	19.7	796	46.3	5	Y
21	Mahindra	2011	174000	100000	East	New York	New York	Diesel	Individual	Manual	Second_O	21	1461	64.1	5	Y
22	Honda	2017	950000	50000	East	New York	New York	Diesel	Individual	Manual	First_Own	25.5	1498	98.6	5	Y
23	Maruti	2015	525000	40000	Central	Texas	Dallas	Diesel	Individual	Manual	First_Own	26.59	1248	74	5	Y
24	Maruti	2012	600000	72000	East	New York	New York	Diesel	Individual	Manual	First_Own	21.5	1248	88.8	5	Y
25	Tata	2018	500000	35000	East	New York	New York	Petrol	Individual	Manual	First_Own	20.3	1199	83.81	5	Y
27	Maruti	2017	275000	28000	West	Washingtc	Seattle	Petrol	Individual	Manual	First_Own	24.7	796	47.3	5	Y
28	Chevrolet	2013	300000	70000	East	Massachu	Boston	Diesel	Individual	Manual	First_Own	18.2	1248	73.8	7	Y
30	Maruti	2018	254999	25000	East	New York	New York	Petrol	Individual	Manual	First_Own	16.8	796	34.2	8	Y
31	Maruti	2017	670000	70000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	24.3	1248	88.5	5	Y
34	Hyundai	2018	730000	2388	West	Washingtc	Seattle	Petrol	Individual	Manual	First_Own	18.6	1197	81.83	5	Y
35	Maruti	2017	650000	16200	West	Washingtc	Seattle	Diesel	Individual	Manual	First_Own	24.3	1248	88.5	5	Y
36	Maruti	2019	330000	10000	West	Washingtc	Seattle	CNG	Individual	Manual	Second_O	33.44	796	40.3	4	Y
37	Maruti	2019	366000	15000	West	Washingtc	Seattle	Petrol	Individual	Manual	First_Own	23.95	998	67.1	5	Y
38	Hyundai	2019	1149000	5000	East	District of	Washingtc	Petrol	Individual	Manual	First_Own	17	1591	121.3	5	Y
39	Datsun	2016	150000	42000	West	California	Los Angele	Petrol	Individual	Manual	First_Own	20.63	1198	67	5	Y
40	Tata	2011	425000	60000	South	North Cari	Charlotte	Diesel	Individual	Manual	Second_O	13.93	2179	138.03	7	Y
41	Maruti	2012	150000	76000	West	California	Los Angele	Petrol	Individual	Manual	First_Own	16.1	796	37	4	Y
42	Jeep	2019	2100000	5000	East	New York	New York	Petrol	Individual	Automatic	First_Own	16	1368	160.77	5	Y
43	Honda	2018	925000	28900	East	New York	New York	Petrol	Dealer	Manual	First_Own	17.8	1497	117.3	5	Y



CITY	Latitude	Longitude
Derry	54.9958	-7.3074
Newcastle upon Tyne	54.9667	-1.6
Chester-le-Street	54.8598	-1.5743
Manchester	53.4807	-2.2344
Dublin	53.3441	-6.2675
Derby	52.9219	-1.4758
Shrewsbury	52.708	-2.754
Norwich	52.6281	1.2993
Birmingham	52.4829	-1.8935
Coventry	52.4081	-1.5106
Hanover	52.3667	9.7167
Northampton	52.2348	-0.8973
Bedford	52.1337	-0.4577
Ipswich	52.0594	1.1556
Rotterdam	51.9167	4.5
Colchester	51.8917	0.903
Oxford	51.7519	-1.2578
Swansea	51.6167	-3.95
Newport	51.5833	-3
Winchester	51.0616	-1.3105
Cologne	50.9405	6.9599
Plymouth	50.3704	-4.1427
Falmouth	50.15	-5.07
Vancouver	49.2636	-123.1386
Bellingham	48.7549	-122.4781
Anacortes	48.5054	-122.6315
Minot	48.2355	-101.2961
Kalispell	48.1978	-114.3161
Everett	47.9791	-122.1959
Grand Forks	47.9257	-97.0361
Lynnwood	47.824	-122.2924
Edmonds	47.8116	-122.3766
Bothell	47.7601	-122.2054
Kenmore	47.7528	-122.2472
Post Falls	47.712	-116.948

```
import pandas as pd
# read csv data
```

```
df1 = pd.read_csv('UserCarDataset1.csv')
print(df1.head())
```

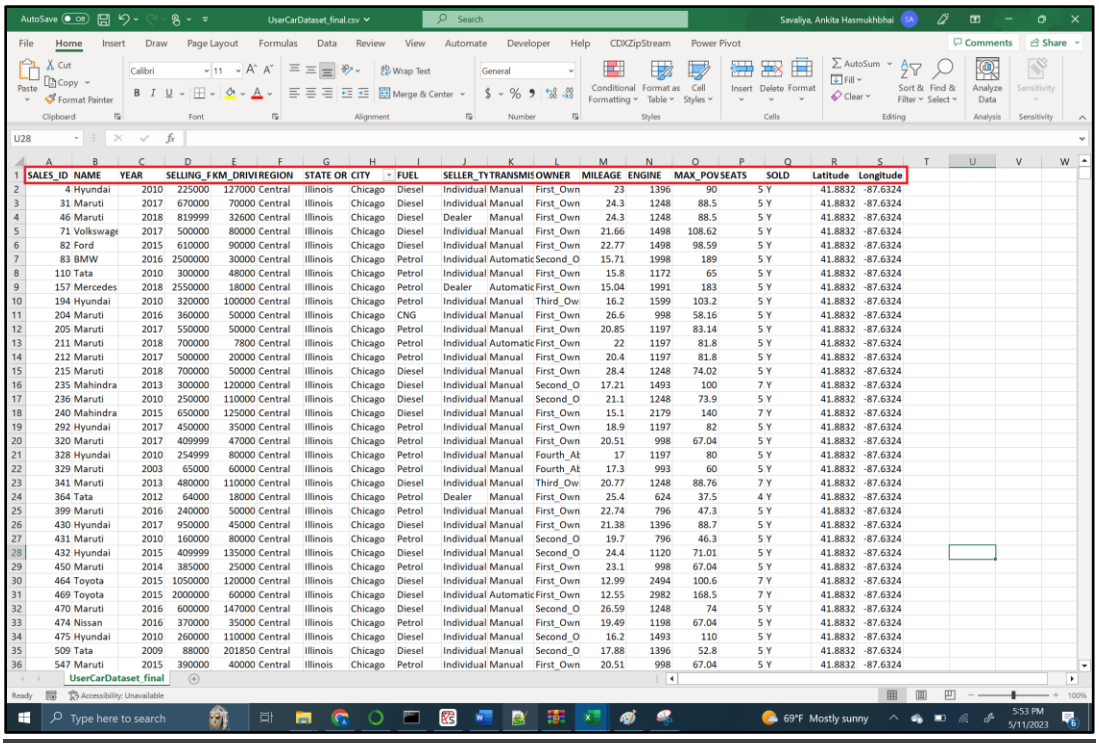


```
   SALES_ID     NAME  YEAR  SELLING_PRICE  ...  ENGINE MAX_POWER SEATS SOLD
0         1   Maruti  2014         450000  ...    1248     74.00     5    Y
1         2    Skoda  2014         370000  ...    1498    103.52     5    Y
2         4  Hyundai  2010         225000  ...    1396     90.00     5    Y
3         5   Maruti  2007         130000  ...    1298     88.20     5    Y
4         6  Hyundai  2017         440000  ...    1197     81.86     5    Y

[5 rows x 17 columns]
```

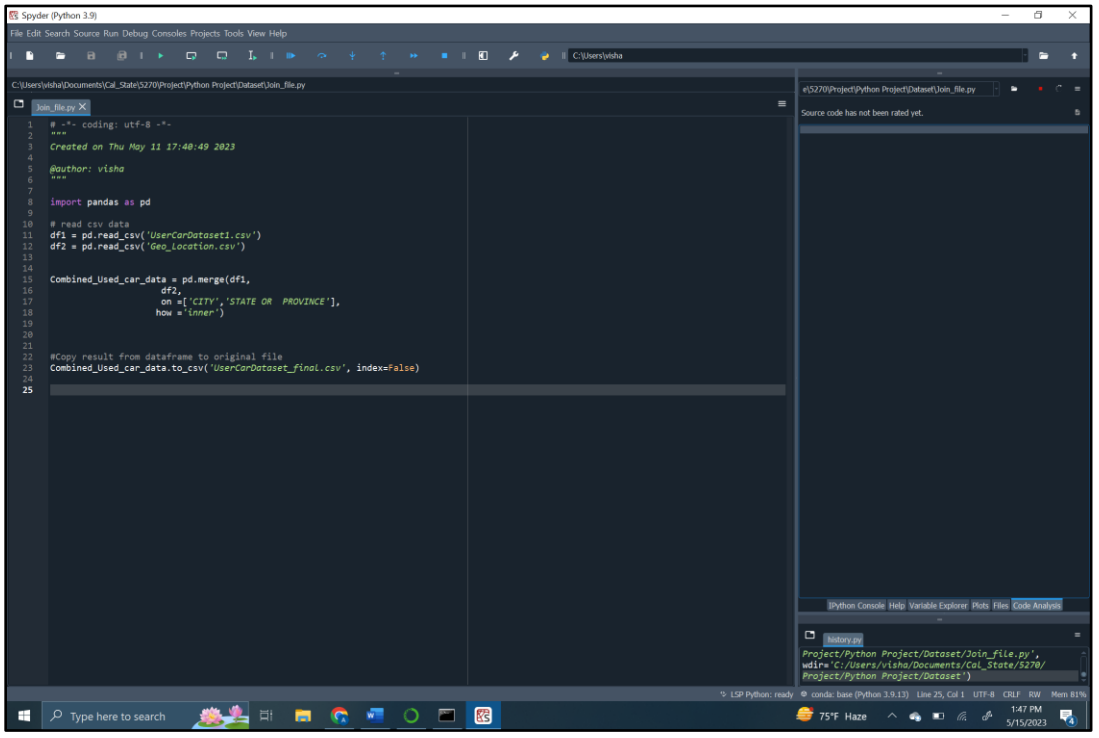
After Cleaning:

```
import pandas as pd
# read csv data
df1 = pd.read_csv('UserCarDataset1.csv')
df2 = pd.read_csv('Geo_Location.csv')
Combined_Used_car_data = pd.merge(df1,
                 df2,
                 on =['CITY','STATE OR  PROVINCE'],
                how ='inner')
```

```
#Copy result from dataframe to original file
Combined_Used_car_data.to_csv('UserCarDataset_final.csv', index=False)
```



SALES_ID	NAME	YEAR	SELLING_P	KM_DRIVI	REGION	STATE OR	CITY	FUEL	SELLER_TY	TRANSMIS	OWNER	MILEAGE	ENGINE	MAX_POV	SEATS	SOLD	Latitude	Longitude
4	Hyundai	2010	225000	127000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	23	1396	90	5	Y	41.8832	-87.6324
31	Maruti	2017	670000	70000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	24.3	1248	88.5	5	Y	41.8832	-87.6324
46	Maruti	2018	819999	32600	Central	Illinois	Chicago	Diesel	Dealer	Manual	First_Own	24.3	1248	88.5	5	Y	41.8832	-87.6324
71	Volkswage	2017	500000	80000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	21.66	1498	108.62	5	Y	41.8832	-87.6324
82	Ford	2015	610000	90000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	22.77	1498	98.59	5	Y	41.8832	-87.6324
83	BMW	2016	2500000	30000	Central	Illinois	Chicago	Petrol	Individual	Automatic	Second_O	15.71	1998	189	5	Y	41.8832	-87.6324
110	Tata	2010	300000	48000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	15.8	1172	65	5	Y	41.8832	-87.6324
157	Mercedes	2018	2550000	18000	Central	Illinois	Chicago	Petrol	Dealer	Automatic	First_Own	15.04	1991	183	5	Y	41.8832	-87.6324
194	Hyundai	2010	320000	100000	Central	Illinois	Chicago	Petrol	Individual	Manual	Third_Ow	16.2	1599	103.2	5	Y	41.8832	-87.6324
204	Maruti	2016	360000	50000	Central	Illinois	Chicago	CNG	Individual	Manual	First_Own	26.6	998	58.16	5	Y	41.8832	-87.6324
205	Maruti	2017	550000	50000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	20.85	1197	83.14	5	Y	41.8832	-87.6324
211	Maruti	2018	700000	7800	Central	Illinois	Chicago	Petrol	Individual	Automatic	First_Own	22	1197	81.8	5	Y	41.8832	-87.6324
212	Maruti	2017	500000	20000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	20.4	1197	81.8	5	Y	41.8832	-87.6324
215	Maruti	2018	700000	50000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	28.4	1248	74.02	5	Y	41.8832	-87.6324
235	Mahindra	2013	300000	120000	Central	Illinois	Chicago	Diesel	Individual	Manual	Second_O	17.21	1493	100	7	Y	41.8832	-87.6324
236	Maruti	2010	250000	110000	Central	Illinois	Chicago	Diesel	Individual	Manual	Second_O	21.1	1248	73.9	5	Y	41.8832	-87.6324
240	Mahindra	2015	650000	125000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	15.1	2179	140	7	Y	41.8832	-87.6324
292	Hyundai	2017	450000	35000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	18.9	1197	82	5	Y	41.8832	-87.6324
320	Maruti	2017	409999	47000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	20.51	998	67.04	5	Y	41.8832	-87.6324
328	Hyundai	2010	254999	80000	Central	Illinois	Chicago	Petrol	Individual	Manual	Fourth_At	17	1197	80	5	Y	41.8832	-87.6324
329	Maruti	2003	65000	60000	Central	Illinois	Chicago	Petrol	Individual	Manual	Fourth_At	17.3	993	60	5	Y	41.8832	-87.6324
341	Maruti	2013	480000	110000	Central	Illinois	Chicago	Diesel	Individual	Manual	Third_Ow	20.77	1248	88.76	7	Y	41.8832	-87.6324
364	Tata	2012	64000	18000	Central	Illinois	Chicago	Petrol	Dealer	Manual	First_Own	25.4	624	37.5	4	Y	41.8832	-87.6324
399	Maruti	2016	240000	50000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	22.74	796	47.3	5	Y	41.8832	-87.6324
430	Hyundai	2017	950000	45000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	21.38	1396	88.7	5	Y	41.8832	-87.6324
431	Maruti	2010	160000	80000	Central	Illinois	Chicago	Petrol	Individual	Manual	Second_O	19.7	796	46.3	5	Y	41.8832	-87.6324
432	Hyundai	2015	409999	135000	Central	Illinois	Chicago	Diesel	Individual	Manual	Second_O	24.4	1120	71.01	5	Y	41.8832	-87.6324
450	Maruti	2014	385000	25000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	23.1	998	67.04	5	Y	41.8832	-87.6324
464	Toyota	2015	1050000	120000	Central	Illinois	Chicago	Diesel	Individual	Manual	First_Own	12.99	2494	100.6	7	Y	41.8832	-87.6324
469	Toyota	2015	2000000	60000	Central	Illinois	Chicago	Diesel	Individual	Automatic	First_Own	12.55	2982	168.5	7	Y	41.8832	-87.6324
470	Maruti	2016	600000	147000	Central	Illinois	Chicago	Diesel	Individual	Manual	Second_O	26.59	1248	74	5	Y	41.8832	-87.6324
474	Nissan	2016	370000	35000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	19.49	1198	67.04	5	Y	41.8832	-87.6324
475	Hyundai	2010	260000	110000	Central	Illinois	Chicago	Diesel	Individual	Manual	Second_O	16.2	1493	110	5	Y	41.8832	-87.6324
509	Tata	2009	88000	201850	Central	Illinois	Chicago	Diesel	Individual	Manual	Second_O	17.88	1396	52.8	5	Y	41.8832	-87.6324
547	Maruti	2015	390000	40000	Central	Illinois	Chicago	Petrol	Individual	Manual	First_Own	20.51	998	67.04	5	Y	41.8832	-87.6324



```
# -*- coding: utf-8 -*-
"""
Created on Thu May 11 17:40:49 2023

@author: visha
"""

import pandas as pd

# read csv data
df1 = pd.read_csv('UserCarDataset1.csv')
df2 = pd.read_csv('Geo_Location.csv')


Combined_Used_car_data = pd.merge(df1,
                 df2,
                 on =['CITY','STATE OR  PROVINCE'],
                how ='inner')



#Copy result from dataframe to original file
Combined_Used_car_data.to_csv('UserCarDataset_final.csv', index=False)
```

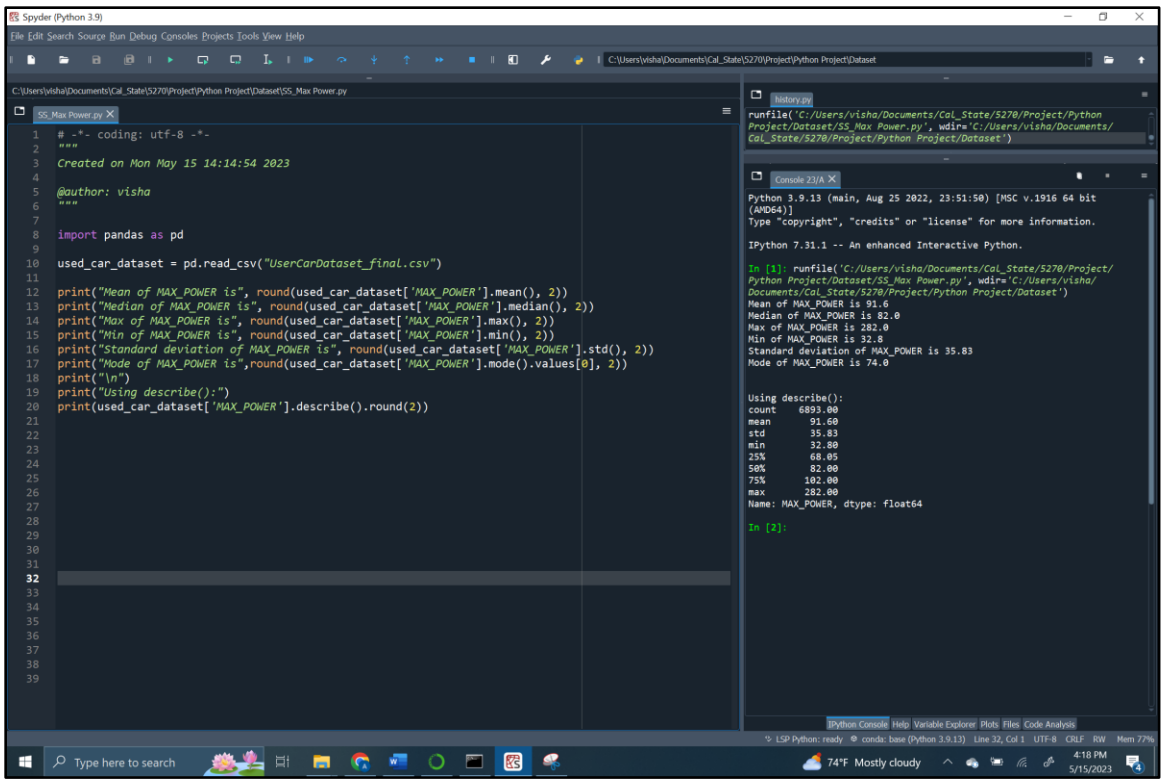
```
   SALES_ID        NAME  YEAR  SELLING_PRICE  ...  SEATS SOLD Latitude Longitude
0         4     Hyundai  2010         225000  ...      5    Y  41.8832  -87.6324
1        31      Maruti  2017         670000  ...      5    Y  41.8832  -87.6324
2        46      Maruti  2018         819999  ...      5    Y  41.8832  -87.6324
3        71  Volkswagen  2017         500000  ...      5    Y  41.8832  -87.6324
4        82        Ford  2015         610000  ...      5    Y  41.8832  -87.6324

[5 rows x 19 columns]
```

D. Summary Statistics:

After data cleaning, we performed summary statistics based on 3 numeric columns ("MAX_POWER", "KM_DRIVEN", and "ENGINE") where we used mean, median, max, min, standard deviation, mode, and describe functions.

1)MAX_POWER:

```
import pandas as pd
used_car_dataset = pd.read_csv("UserCarDataset_final.csv")
print("Mean of MAX_POWER is", round(used_car_dataset['MAX_POWER'].mean(), 2))
print("Median of MAX_POWER is", round(used_car_dataset['MAX_POWER'].median(), 2))
print("Max of MAX_POWER is", round(used_car_dataset['MAX_POWER'].max(), 2))
print("Min of MAX_POWER is", round(used_car_dataset['MAX_POWER'].min(), 2))
print("Standard deviation of MAX_POWER is", round(used_car_dataset['MAX_POWER'].std(), 2))
print("Mode of MAX_POWER is",round(used_car_dataset['MAX_POWER'].mode().values[0], 2))
print("\n")
print("Using describe():")
print(used_car_dataset['MAX_POWER'].describe().round(2))
```




```python
import pandas as pd

used_car_dataset = pd.read_csv("UserCarDataset_final.csv")

print("Mean of MAX_POWER is", round(used_car_dataset['MAX_POWER'].mean(), 2))
print("Median of MAX_POWER is", round(used_car_dataset['MAX_POWER'].median(), 2))
print("Max of MAX_POWER is", round(used_car_dataset['MAX_POWER'].max(), 2))
print("Min of MAX_POWER is", round(used_car_dataset['MAX_POWER'].min(), 2))
print("Standard deviation of MAX_POWER is", round(used_car_dataset['MAX_POWER'].std(), 2))
print("Mode of MAX_POWER is",round(used_car_dataset['MAX_POWER'].mode().values[0], 2))
print("\n")
print("Using describe():")
print(used_car_dataset['MAX_POWER'].describe().round(2))
```

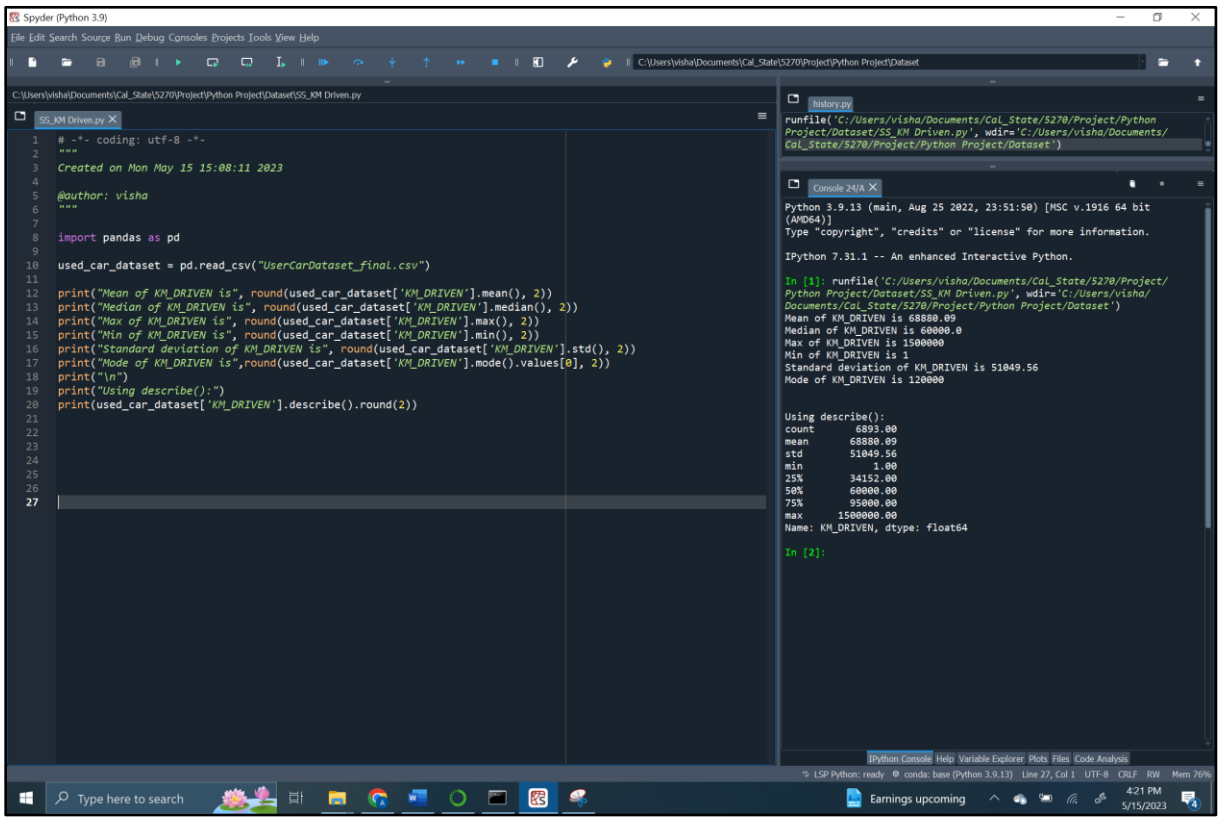
```
Mean of MAX_POWER is 91.6
Median of MAX_POWER is 82.0
Max of MAX_POWER is 282.0
Min of MAX_POWER is 32.8
Standard deviation of MAX_POWER is 35.83
Mode of MAX_POWER is 74.0


Using describe():
count    6893.00
mean       91.60
std        35.83
min        32.80
25%        68.05
50%        82.00
75%       102.00
max       282.00
Name: MAX_POWER, dtype: float64
```

Interpretation of the statistical results:

Based on the statistical summary, the mean of MAX_POWER is 91.6 which means the average value of the maximum power of used cars in the dataset while median value is 82 which indicates that half of the MAX_POWER values fall below this point. The difference between the mean and median suggests a potential asymmetry or presence of outliers in the distribution. Besides, the maximum value of MAX_POWER is 282 which means highest recorded maximum power among the used cars in the dataset, while the minimum value of 32.8 represents the lowest recorded maximum power. The standard deviation is 35.83 which indicates that a relatively wide dispersion of values around the mean, indicating variability in the maximum power of the used cars. The mode of 74 indicates that the most common maximum power value among the cars is 74.

2) KM_DRIVEN:

```
import pandas as pd
used_car_dataset = pd.read_csv("UserCarDataset_final.csv")
print("Mean of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].mean(), 2))
print("Median of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].median(), 2))
print("Max of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].max(), 2))
print("Min of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].min(), 2))
print("Standard deviation of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].std(),
2))
print("Mode of KM_DRIVEN is",round(used_car_dataset['KM_DRIVEN'].mode().values[0], 2))
print("\n")
print("Using describe():")
print(used_car_dataset['KM_DRIVEN'].describe().round(2))
```




```python
import pandas as pd

used_car_dataset = pd.read_csv("UserCarDataset_final.csv")

print("Mean of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].mean(), 2))
print("Median of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].median(), 2))
print("Max of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].max(), 2))
print("Min of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].min(), 2))
print("Standard deviation of KM_DRIVEN is", round(used_car_dataset['KM_DRIVEN'].std(), 2))
print("Mode of KM_DRIVEN is",round(used_car_dataset['KM_DRIVEN'].mode().values[0], 2))
print("\n")
print("Using describe():")
print(used_car_dataset['KM_DRIVEN'].describe().round(2))
```

```
Mean of KM_DRIVEN is 68880.09
Median of KM_DRIVEN is 60000.0
Max of KM_DRIVEN is 1500000
Min of KM_DRIVEN is 1
Standard deviation of KM_DRIVEN is 51049.56
Mode of KM_DRIVEN is 120000


Using describe():
count       6893.00
mean       68880.09
std        51049.56
min            1.00
25%        34152.00
50%        60000.00
75%        95000.00
max      1500000.00
Name: KM_DRIVEN, dtype: float64
```

Interpretation of the statistical results:

The mean of 'KM_DRIVEN' is 68880.09, indicating that, on average, the used cars in the dataset have been driven approximately 68,880 kilometers. The median value, which represents the midpoint of the data, is 60,000.0 kilometers. This indicates that half of the used cars have a distance driven less than or equal to 60,000.0 kilometers. The maximum value of 'KM_DRIVEN' is 1500000, indicating that the highest recorded value for kilometers driven among the used cars in the dataset is 1,500,000 kilometers. The minimum value of 'KM_DRIVEN' is 1, indicating that there is at least one used car in the dataset with only 1 kilometer driven. The standard deviation of 'KM_DRIVEN' is 51049.56. This indicates the dispersion or variability of the 'KM_DRIVEN' values around the mean. A higher standard deviation suggests a wider range of values. The mode of 'KM_DRIVEN' is 120000, which is the most frequently occurring value in the column. This indicates that a significant number of used cars in the dataset have been driven approximately 120,000 kilometers.

Using the describe() function provides additional descriptive statistics:

The count of records for KM_DRIVEN is 6,893.

The 25th percentile (1st quartile) is 34,152, meaning that 25% of the data points have a value less than or equal to this value.

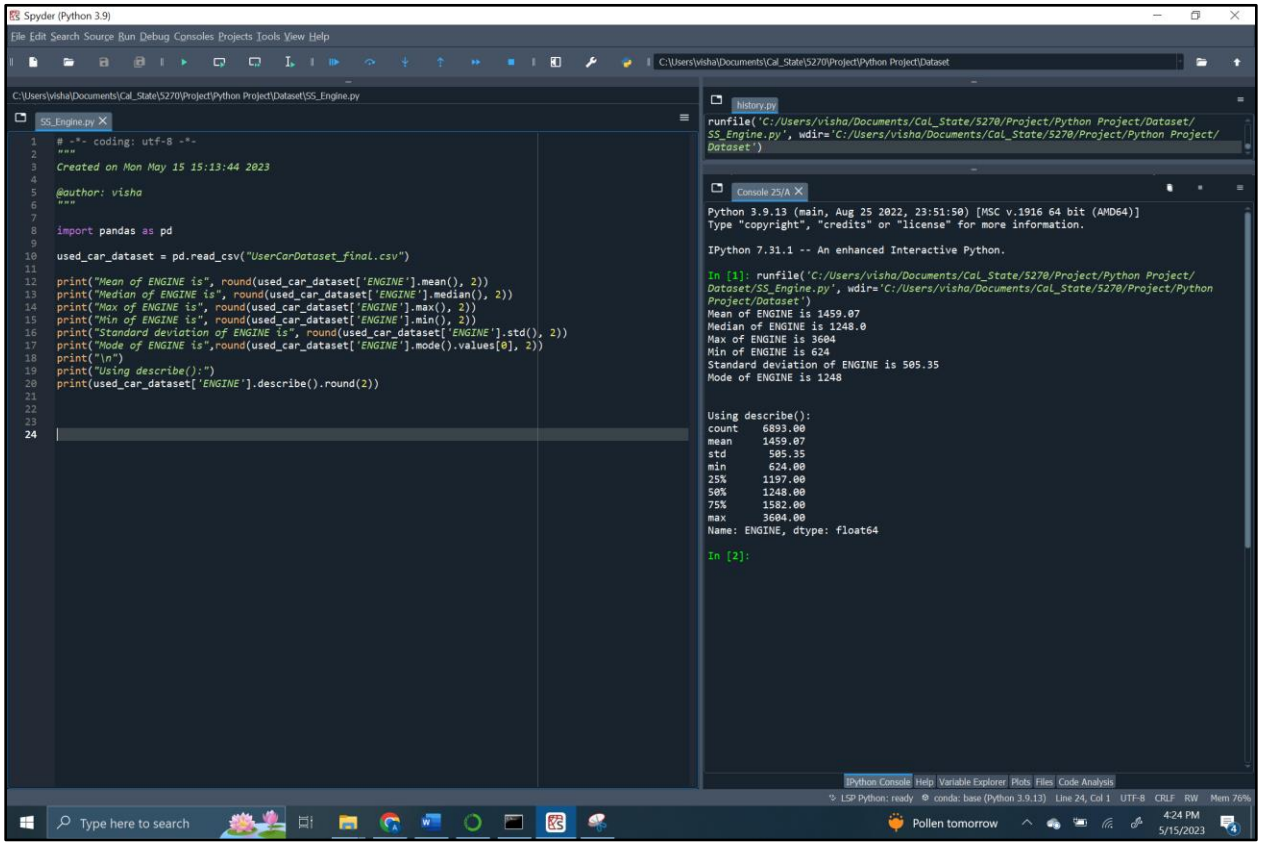
The 50th percentile (2nd quartile) or median is 60,000, indicating that 50% of the data points have a value less than or equal to this value.

The 75th percentile (3rd quartile) is 95,000, implying that 75% of the data points have a value less than or equal to this value.

This analysis provides a comprehensive understanding of the 'KM_DRIVEN' column, including central tendency measures (mean, median, mode), dispersion (standard deviation), and additional quartile values obtained through the describe() function.

3) ENGINE:

```
import pandas as pd
used_car_dataset = pd.read_csv("UserCarDataset_final.csv")
print("Mean of ENGINE is", round(used_car_dataset['ENGINE'].mean(), 2))
print("Median of ENGINE is", round(used_car_dataset['ENGINE'].median(), 2))
print("Max of ENGINE is", round(used_car_dataset['ENGINE'].max(), 2))
print("Min of ENGINE is", round(used_car_dataset['ENGINE'].min(), 2))
print("Standard deviation of ENGINE is", round(used_car_dataset['ENGINE'].std(), 2))
print("Mode of ENGINE is",round(used_car_dataset['ENGINE'].mode().values[0], 2))
print("\n")
print("Using describe():")
print(used_car_dataset['ENGINE'].describe().round(2))
```




```python
import pandas as pd

used_car_dataset = pd.read_csv("UserCarDataset_final.csv")

print("Mean of ENGINE is", round(used_car_dataset['ENGINE'].mean(), 2))
print("Median of ENGINE is", round(used_car_dataset['ENGINE'].median(), 2))
print("Max of ENGINE is", round(used_car_dataset['ENGINE'].max(), 2))
print("Min of ENGINE is", round(used_car_dataset['ENGINE'].min(), 2))
print("Standard deviation of ENGINE is", round(used_car_dataset['ENGINE'].std(), 2))
print("Mode of ENGINE is",round(used_car_dataset['ENGINE'].mode().values[0], 2))
print("\n")
print("Using describe():")
print(used_car_dataset['ENGINE'].describe().round(2))
```

```
Mean of ENGINE is 1459.07
Median of ENGINE is 1248.0
Max of ENGINE is 3604
Min of ENGINE is 624
Standard deviation of ENGINE is 505.35
Mode of ENGINE is 1248


Using describe():
count    6893.00
mean     1459.07
std       505.35
min       624.00
25%      1197.00
50%      1248.00
75%      1582.00
max      3604.00
Name: ENGINE, dtype: float64
```

Interpretation of the statistical results:

The mean engine displacement of used cars is approximately 1459.07 cc (cubic centimeters).

The median value, which represents the midpoint of the data, is 1248.0 cc. This indicates that half of the used cars have an engine displacement less than or equal to 1248.0 cc. The maximum value in the 'ENGINE' column is 3604 cc, indicating that some cars have larger engine displacements. The minimum value is 624 cc, suggesting that there are entries for cars with smaller engine displacements. The 'ENGINE' column has a standard deviation of approximately 505.35 cc. This value measures the spread or dispersion of the data points from the mean, indicating a significant variation in engine sizes. The mode of the 'ENGINE' column is 1248 cc. This value represents the most frequently occurring engine displacement among the used cars.

Using the describe() function provides additional descriptive statistics:

Count: The 'ENGINE' column consists of 6,893 non-null entries, indicating the number of available data points.

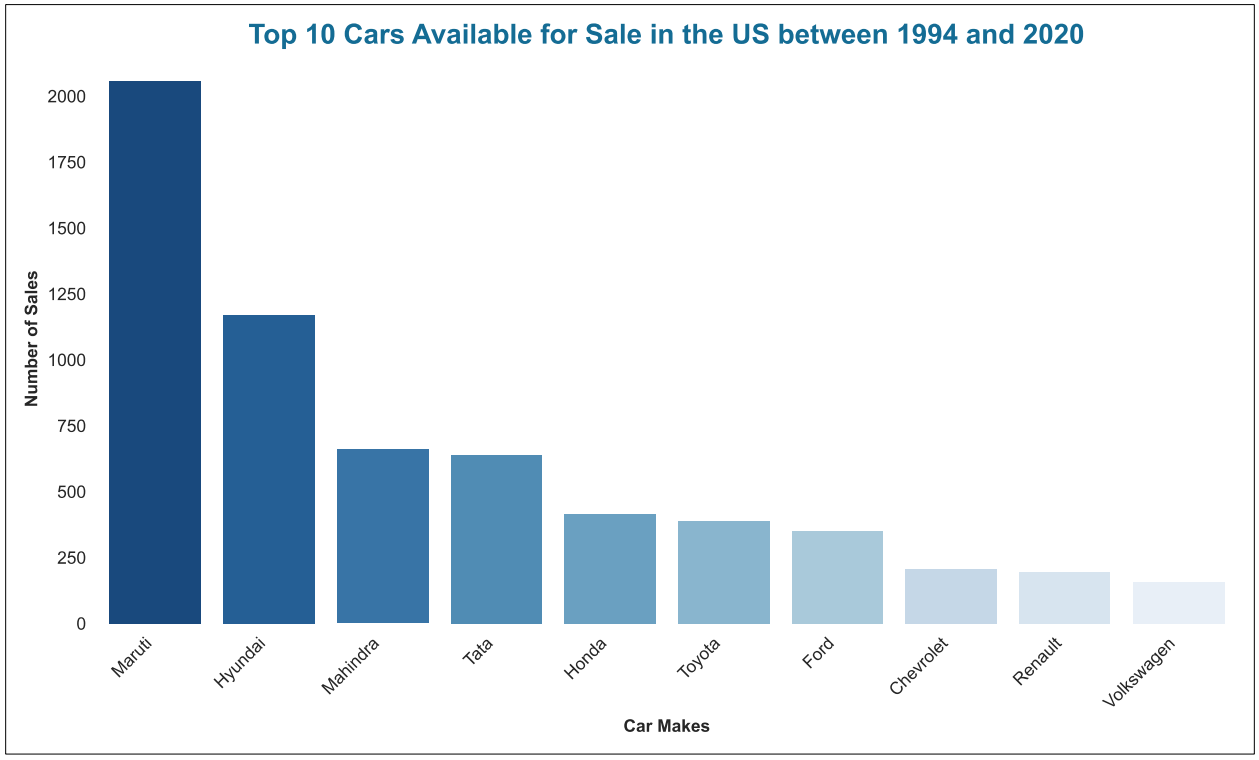
25%, 50%, and 75%: The 25th percentile (Q1) is 1197 cc, the 50th percentile (Q2 or median) is 1248 cc, and the 75th percentile (Q3) is 1582 cc. These values provide insights into the distribution of the engine displacements across different quartiles.

Maximum: The maximum value remains unchanged at 3604 cc.

This detailed analysis offers a comprehensive understanding of the 'ENGINE' column, including measures of central tendency (mean, median, mode), dispersion (standard deviation), and additional quartile values obtained through the describe () function.

E. Analysis & Visualizations:

1) Which were the top 10 cars available for sale in the US between 1994 and 2020?



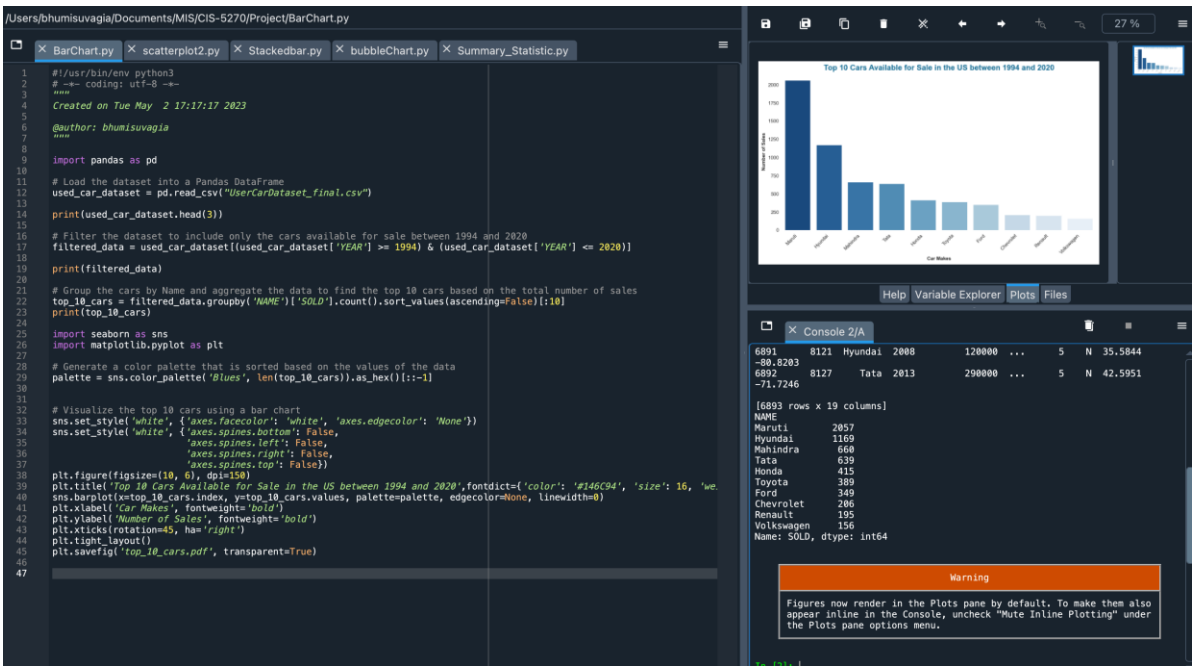
Insight:

The above bar chart illustrates the top 10 cars available for sale in the US between 1994 and 2020 based on the total number of sales. It also highlights the number of sales for each car model and

ranks them in descending order. From the above chart, it can be observed that the Maruti was the most popular car model, with over 2,000 sales. This is followed by the Hyundai which sales were above 1,000 between 1994 and 2020. Besides, Mahindra and Tata had the same sales range which were approximately 700 and 600 respectively. The other car models that make up the top 10 include Toyota, Ford, Chevrolet, Renault, and Volkswagen. From this visualization we can say that it provides valuable insights into the popularity of different car models in the US market, which could be useful for car manufacturers and dealerships in making informed decisions about their products and marketing strategies.

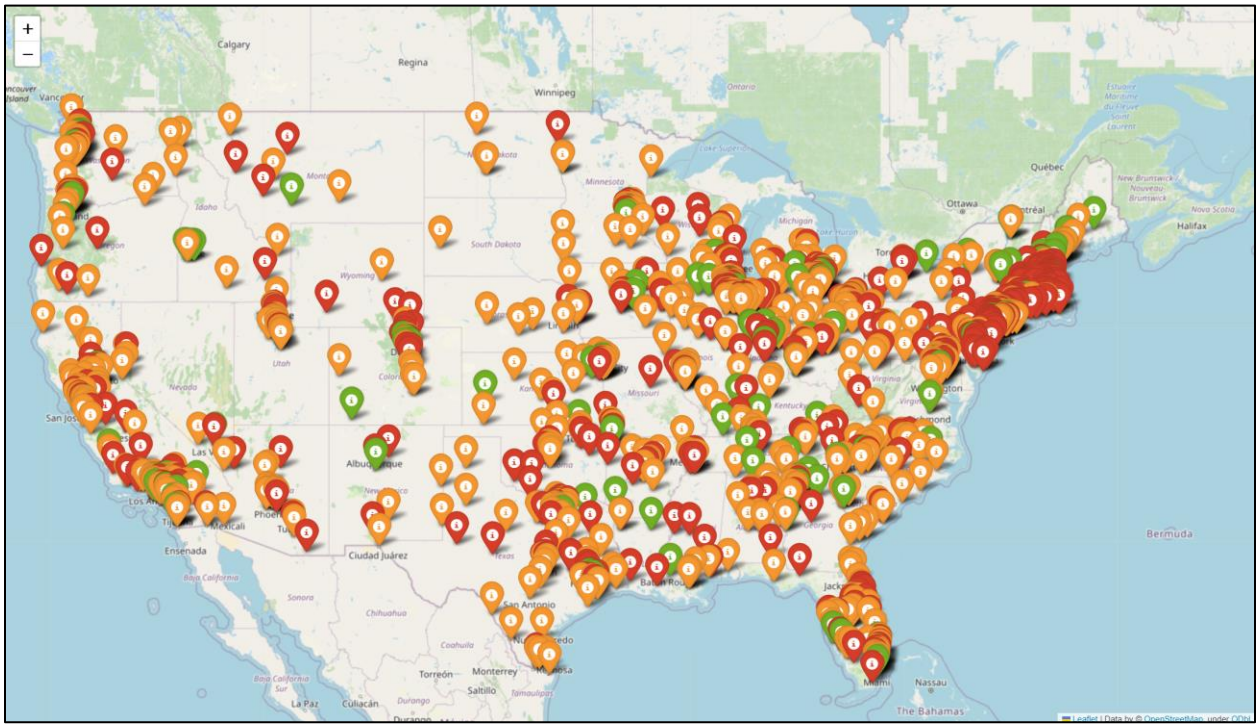
```
import pandas as pd
# Load the dataset into a Pandas DataFrame
used_car_dataset = pd.read_csv("UserCarDataset_final.csv")
print(used_car_dataset.head(3))

# Filter the dataset to include only the cars available for sale between 1994 and 2020
filtered_data = used_car_dataset[(used_car_dataset['YEAR'] >= 1994) &
(used_car_dataset['YEAR'] <= 2020)]
print(filtered_data)

# Group the cars by Name and aggregate the data to find the top 10 cars based on the total number of sales
top_10_cars =
filtered_data.groupby('NAME')['SOLD'].count().sort_values(ascending=False)[:10]
print(top_10_cars)

import seaborn as sns
```

```
import matplotlib.pyplot as plt


# Generate a color palette that is sorted based on the values of the data
palette = sns.color_palette('Blues', len(top_10_cars)).as_hex()[::-1]


# Visualize the top 10 cars using a bar chart
sns.set_style('white', {'axes.facecolor': 'white', 'axes.edgecolor': 'None'})
sns.set_style('white', {'axes.spines.bottom': False,
                        'axes.spines.left': False,
                        'axes.spines.right': False,
                        'axes.spines.top': False})
plt.figure(figsize=(10, 6), dpi=150)
plt.title('Top 10 Cars Available for Sale in the US between 1994 and 2020',fontdict={'color':
'#146C94', 'size': 16, 'weight': 'bold'})
sns.barplot(x=top_10_cars.index, y=top_10_cars.values, palette=palette, edgecolor=None,
linewidth=0)
plt.xlabel('Car Mokes', fontweight='bold')
plt.ylabel('Number of Sales', fontweight='bold')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.savefig('top_10_cars.pdf', transparent=True)
```




```python
import pandas as pd

# Load the dataset into a Pandas DataFrame
used_car_dataset = pd.read_csv("UserCarDataset_final.csv")

print(used_car_dataset.head(3))

# Filter the dataset to include only the cars available for sale between 1994 and 2020
filtered_data = used_car_dataset[(used_car_dataset['YEAR'] >= 1994) & (used_car_dataset['YEAR'] <= 2020)]

print(filtered_data)

# Group the cars by Name and aggregate the data to find the top 10 cars based on the total number of sales
top_10_cars = filtered_data.groupby('NAME')['SOLD'].count().sort_values(ascending=False)[:10]
print(top_10_cars)

import seaborn as sns
import matplotlib.pyplot as plt

# Generate a color palette that is sorted based on the values of the data
palette = sns.color_palette('Blues', len(top_10_cars)).as_hex()[::-1]


# Visualize the top 10 cars using a bar chart
sns.set_style('white', {'axes.facecolor': 'white', 'axes.edgecolor': 'None'})
sns.set_style('white', {'axes.spines.bottom': False,
                        'axes.spines.left': False,
                        'axes.spines.right': False,
                        'axes.spines.top': False})
plt.figure(figsize=(10, 6), dpi=150)
plt.title('Top 10 Cars Available for Sale in the US between 1994 and 2020',fontdict={'color': '#146C94', 'size': 16, 'weight': 'bold'})
sns.barplot(x=top_10_cars.index, y=top_10_cars.values, palette=palette, edgecolor=None, linewidth=0)
plt.xlabel('Car Makes', fontweight='bold')
plt.ylabel('Number of Sales', fontweight='bold')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.savefig('top_10_cars.pdf', transparent=True)
```

2) How many cars are listed for sale in different cities?



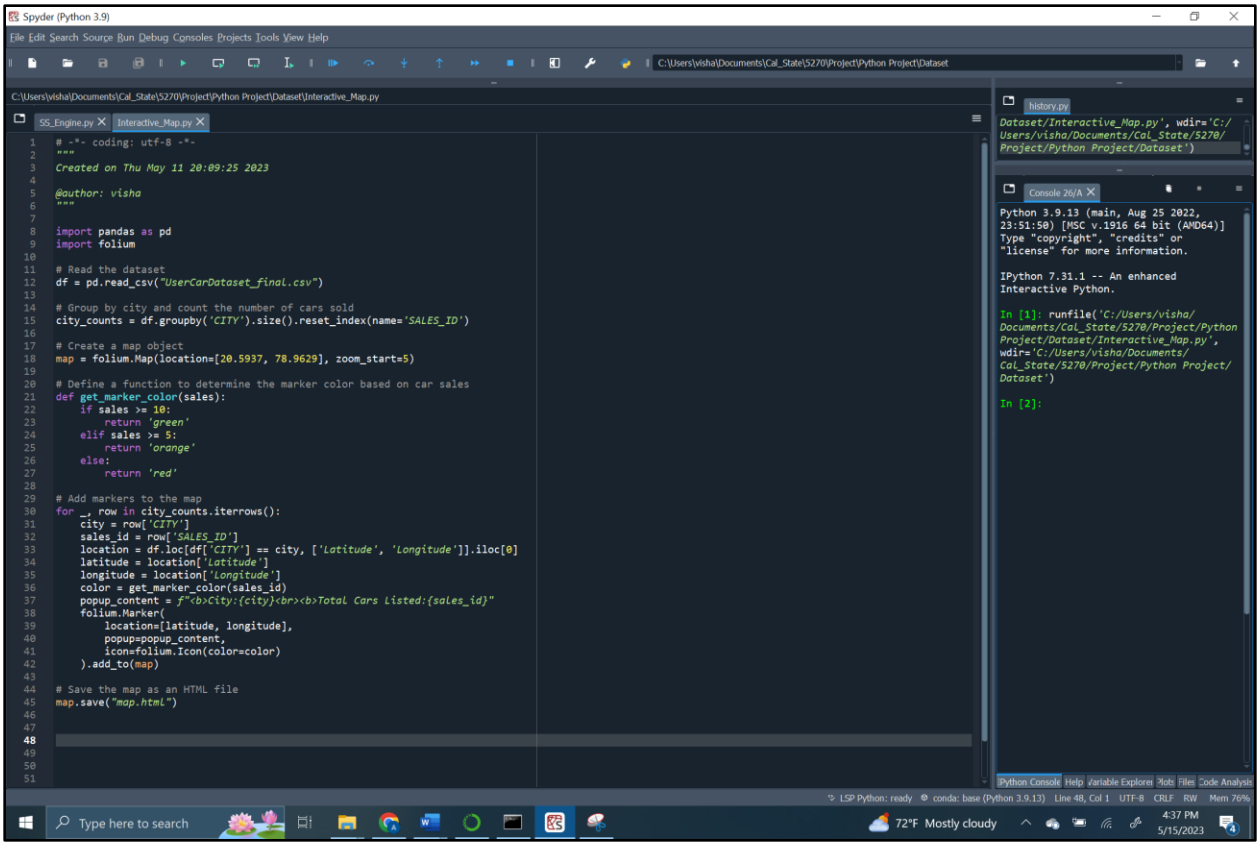
Insight:

The interactive map shows the total number of cars listed for sale in various cities across the United States from 1994 to 2020.The map utilizes different colors to represent the number of cars listed in each city. Cities with a significant number of car listings, 10 or more, are displayed in green, while cities with 5 or more listings are shown in orange. Cities with fewer than 5 listings are marked in red. By opening the HTML file in the Chrome browser, users will be able to see the map with markers representing different cities. When a user clicks on a marker on the map, a popup will appear displaying the city name and the total number of cars listed for sale in that city. This interactive feature allows users to explore the map, view specific city information, and analyze the distribution of car listings across different cities in the United States.

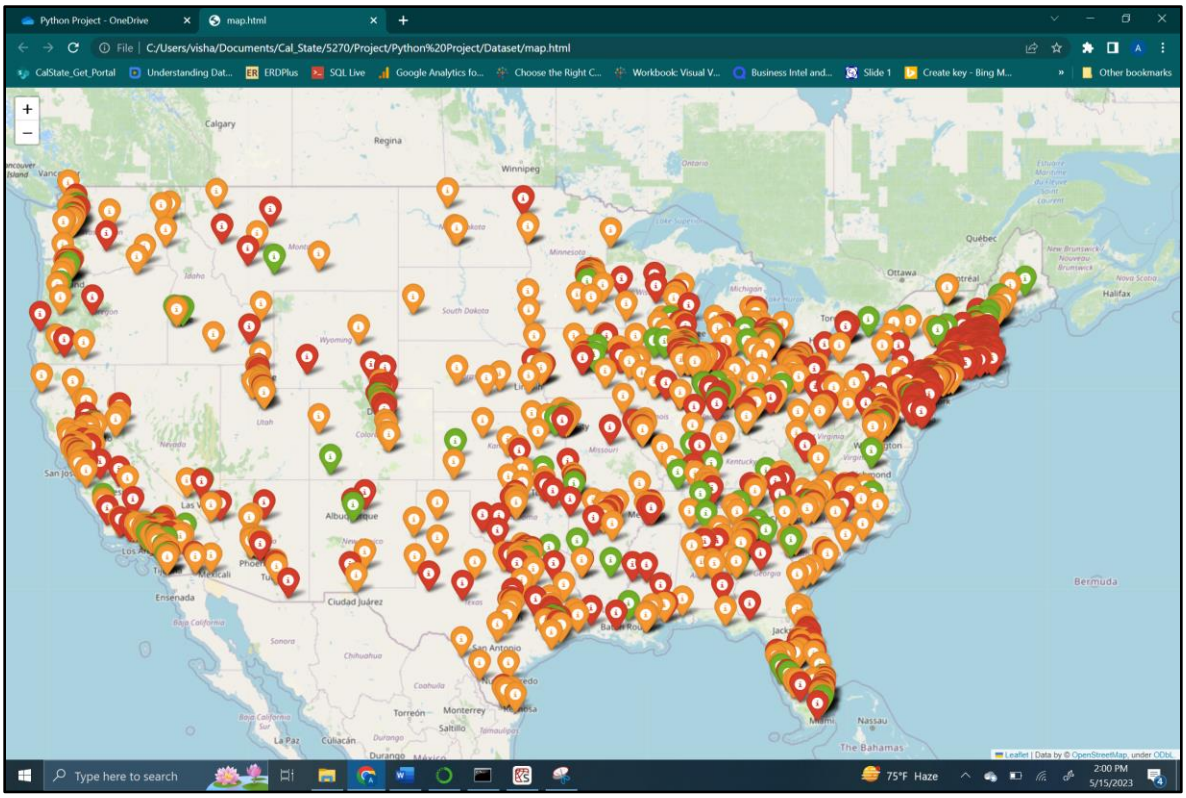
```
import pandas as pd
import folium
```

```
# Read the dataset
df = pd.read_csv("UserCarDataset_final.csv")

# Group by city and count the number of cars sold
city_counts = df.groupby('CITY').size().reset_index(name='SALES_ID')

# Create a map object
map = folium.Map(location=[20.5937, 78.9629], zoom_start=5)

# Define a function to determine the marker color based on car sales
def get_marker_color(sales):
    if sales >= 10:
        return 'green'
    elif sales >= 5:
        return 'orange'
    else:
        return 'red'

# Add markers to the map
for _, row in city_counts.iterrows():
    city = row['CITY']
    sales_id = row['SALES_ID']
    location = df.loc[df['CITY'] == city, ['Latitude', 'Longitude']].iloc[0]
    latitude = location['Latitude']
    longitude = location['Longitude']
    color = get_marker_color(sales_id)
    popup_content = f"<b>City:{city}<br><b>Total Cars Listed:{sales_id}"
    folium.Marker(
        location=[latitude, longitude],
        popup=popup_content,
        icon=folium.Icon(color=color)
```

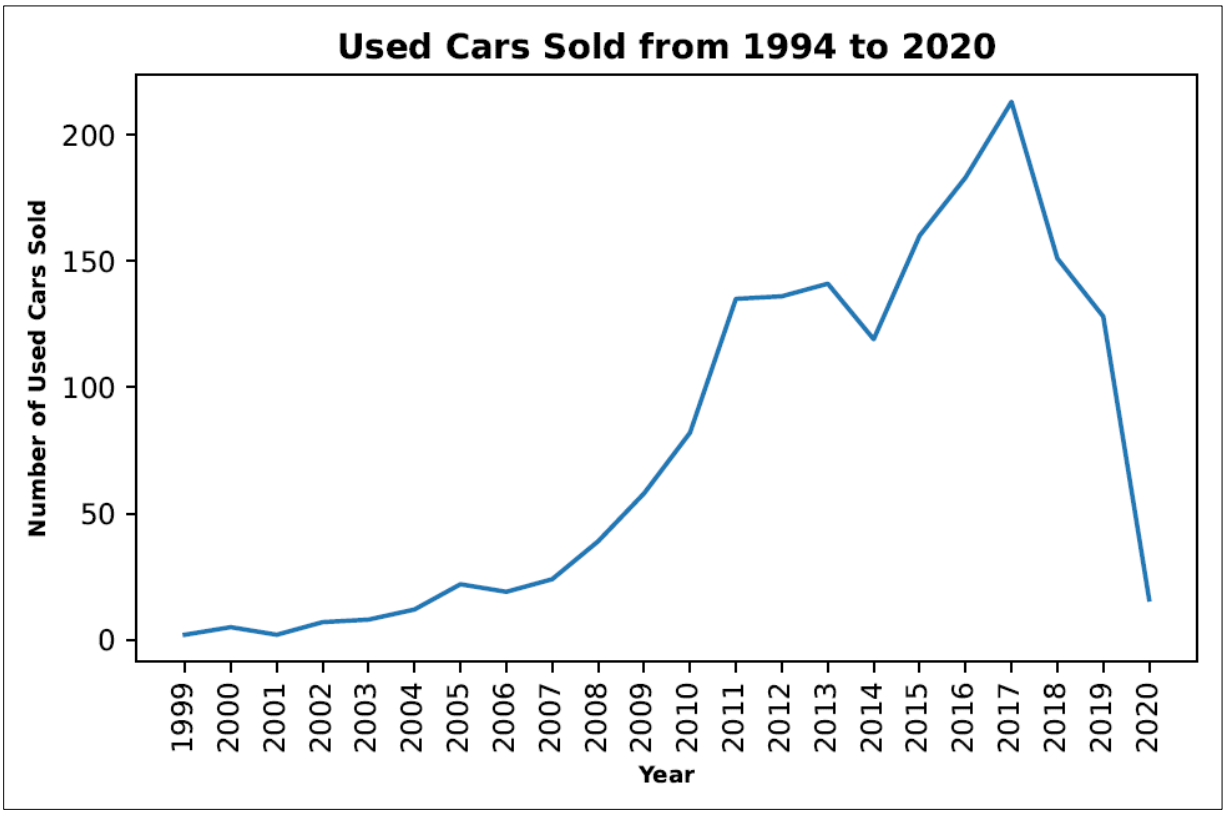
```
    ).add_to(map)


# Save the map as an HTML file
map.save("Map.html")
```



```python
import pandas as pd
import folium

# Read the dataset
df = pd.read_csv("UserCarDataset_final.csv")

# Group by city and count the number of cars sold
city_counts = df.groupby('CITY').size().reset_index(name='SALES_ID')

# Create a map object
map = folium.Map(location=[20.5937, 78.9629], zoom_start=5)

# Define a function to determine the marker color based on car sales
def get_marker_color(sales):
    if sales >= 10:
        return 'green'
    elif sales >= 5:
        return 'orange'
    else:
        return 'red'

# Add markers to the map
for _, row in city_counts.iterrows():
    city = row['CITY']
    sales_id = row['SALES_ID']
    location = df.loc[df['CITY'] == city, ['Latitude', 'Longitude']].iloc[0]
    latitude = location['Latitude']
    longitude = location['Longitude']
    color = get_marker_color(sales_id)
    popup_content = f"<b>City:{city}<br><b>Total Cars Listed:{sales_id}"
    folium.Marker(
        location=[latitude, longitude],
        popup=popup_content,
        icon=folium.Icon(color=color)
    ).add_to(map)

# Save the map as an HTML file
map.save("map.html")
```



3) How many used cars were sold from 1994 to 2020, categorized by year?

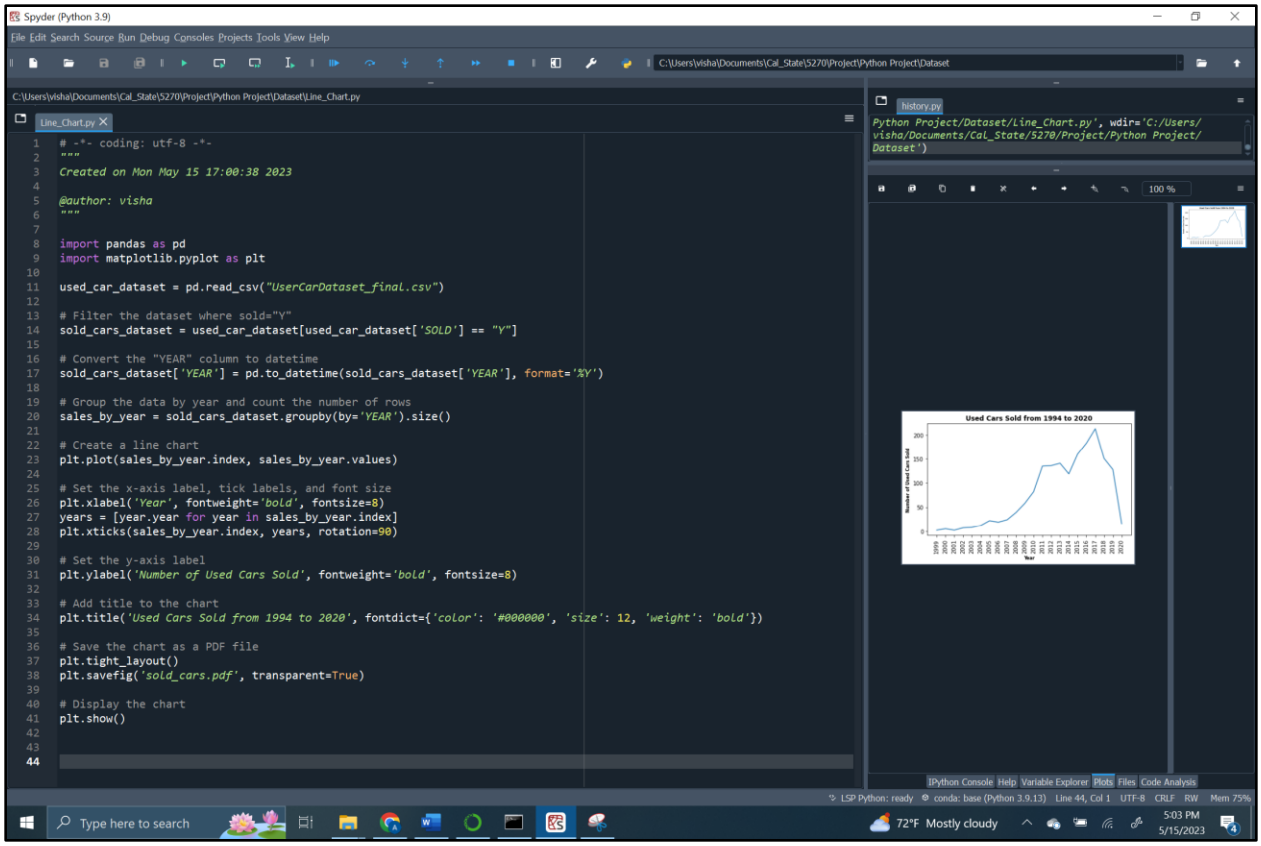


Insights:

The line chart depicting used car sales in the US from 1994 to 2020 presents a comprehensive overview of the trends observed in the used car market. The data reveals no car has been sold from 1994 t0 1998. From 1999, the number of cars has been listed as a sold and numbers keep rising throughout the years till 2013. After a slight decline in the following year, the number of sold cars rose significantly, surpassing the combined sales of previous years. The peak was observed in 2017, but from that point onwards, there was a drastic decline in sales, continuing until the end of 2020 as it is mostly likely due to the covid-19 pandemic.

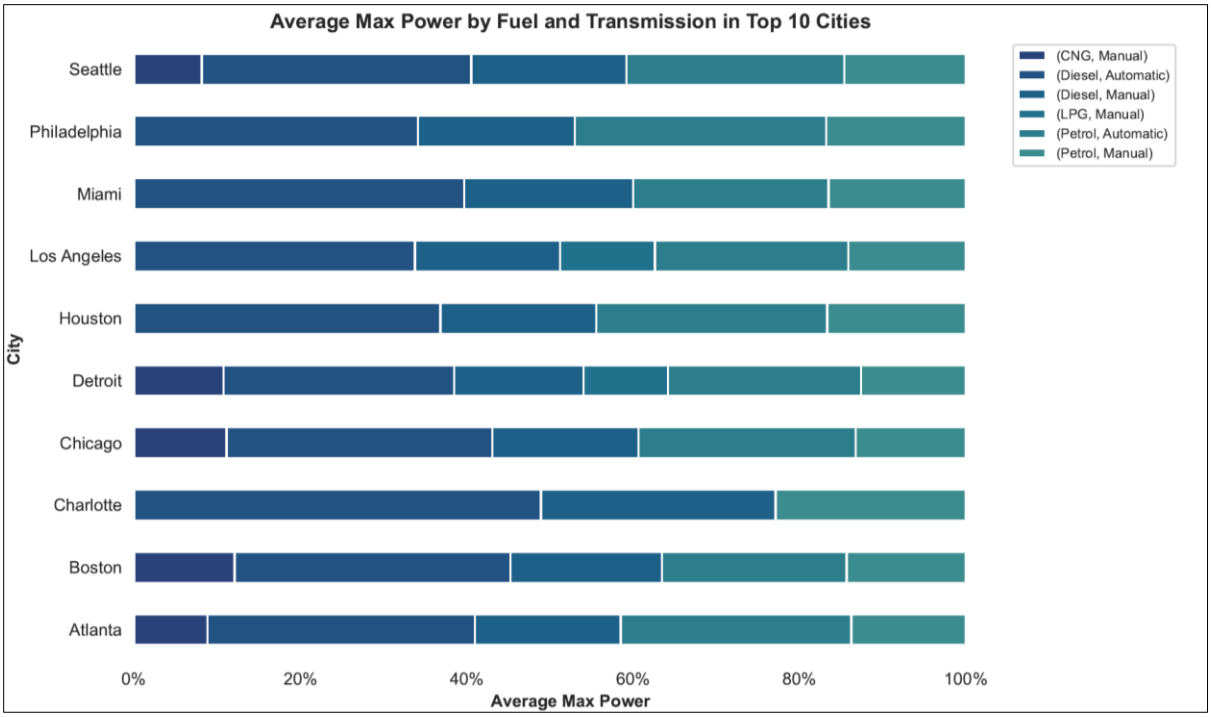
```
import pandas as pd
import matplotlib.pyplot as plt

used_car_dataset = pd.read_csv("UserCarDataset_final.csv")

# Filter the dataset where sold="Y"
sold_cars_dataset = used_car_dataset[used_car_dataset['SOLD'] == "Y"]

# Convert the "YEAR" column to datetime
sold_cars_dataset['YEAR'] = pd.to_datetime(sold_cars_dataset['YEAR'], format='%Y')

# Group the data by year and count the number of rows
sales_by_year = sold_cars_dataset.groupby(by='YEAR').size()

# Create a line chart
plt.plot(sales_by_year.index, sales_by_year.values)

# Set the x-axis label, tick labels, and font size
plt.xlabel('Year', fontweight='bold', fontsize=8)
years = [year.year for year in sales_by_year.index]
plt.xticks(sales_by_year.index, years, rotation=90)

# Set the y-axis label
plt.ylabel('Number of Used Cars Sold', fontweight='bold', fontsize=8)

# Add title to the chart
plt.title('Used Cars Sold from 1994 to 2020', fontdict={'color': '#000000', 'size': 12, 'weight':
'bold'})

# Save the chart as a PDF file
plt.tight_layout()
```

```
plt.savefig('sold_cars.pdf', transparent=True)


# Display the chart
plt.show()
```



```python
import pandas as pd
import matplotlib.pyplot as plt

used_car_dataset = pd.read_csv("UserCarDataset_final.csv")

# Filter the dataset where sold="Y"
sold_cars_dataset = used_car_dataset[used_car_dataset['SOLD'] == "Y"]

# Convert the "YEAR" column to datetime
sold_cars_dataset['YEAR'] = pd.to_datetime(sold_cars_dataset['YEAR'], format='%Y')

# Group the data by year and count the number of rows
sales_by_year = sold_cars_dataset.groupby(by='YEAR').size()

# Create a line chart
plt.plot(sales_by_year.index, sales_by_year.values)

# Set the x-axis label, tick labels, and font size
plt.xlabel('Year', fontweight='bold', fontsize=8)
years = [year.year for year in sales_by_year.index]
plt.xticks(sales_by_year.index, years, rotation=90)

# Set the y-axis label
plt.ylabel('Number of Used Cars Sold', fontweight='bold', fontsize=8)

# Add title to the chart
plt.title('Used Cars Sold from 1994 to 2020', fontdict={'color': '#000000', 'size': 12, 'weight': 'bold'})

# Save the chart as a PDF file
plt.tight_layout()
plt.savefig('sold_cars.pdf', transparent=True)

# Display the chart
plt.show()
```

4)How does the average maximum power of used cars vary across the top 10 cities in terms of fuel type and transmission?



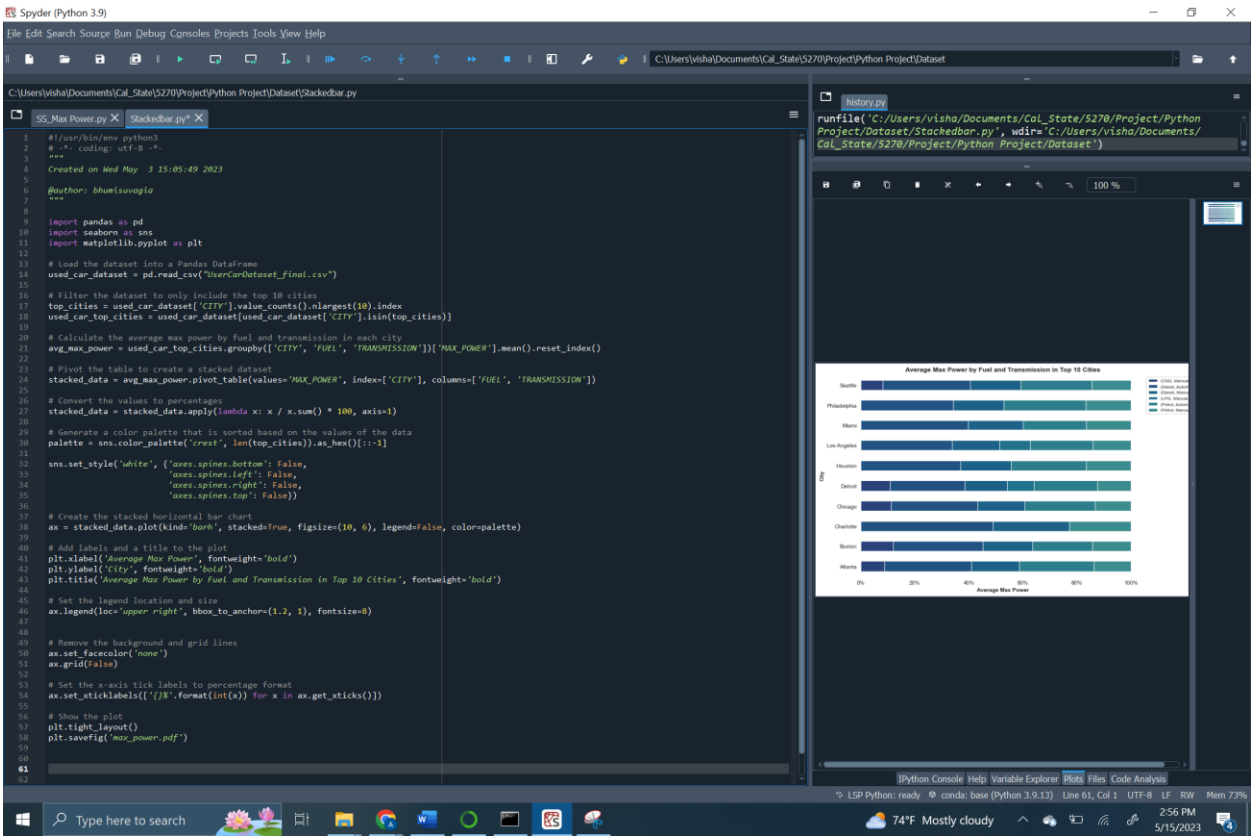
Insight:

The above 100% stacked bar graph illustrates the average maximum power of used cars across the top 10 cities in the United States. It can be clearly observed that in Boston, CNG with manual transmission stood out, accounting for approximately 15% of used cars. Charlotte showed a clear preference for diesel-powered vehicles, with around 50% having automatic transmission and 25% equipped with manual transmission. LPG with manual transmission was observed exclusively in Los Angeles and Detroit, making up approximately 8% and 7% of used cars, respectively. Philadelphia exhibited a higher proportion of petrol-powered cars with automatic transmission, around 20%. Petrol with manual transmission was found in all cities, indicating its widespread availability and popularity. Notably, the combination of CNG and manual transmission was not found in Philadelphia, Miami, Houston, or Charlotte. Overall, the analysis provides valuable insights into the fuel type and transmission preferences of used car buyers in these cities, highlighting regional variations and indicating the dominant choices in each market.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset into a Pandas DataFrame
used_car_dataset = pd.read_csv("UserCarDataset_final.csv")

# Filter the dataset to only include the top 10 cities
top_cities = used_car_dataset['CITY'].value_counts().nlargest(10).index
used_car_top_cities = used_car_dataset[used_car_dataset['CITY'].isin(top_cities)]

# Calculate the average max power by fuel and transmission in each city
```

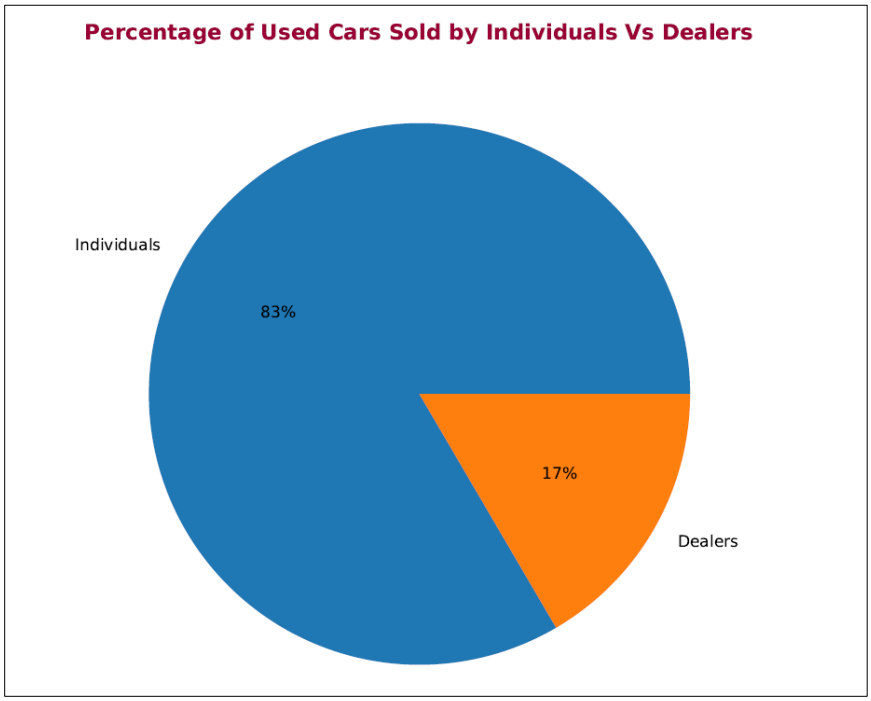
```
avg_max_power = used_car_top_cities.groupby(['CITY', 'FUEL',
'TRANSMISSION'])['MAX_POWER'].mean().reset_index()

# Pivot the table to create a stacked dataset
stacked_data = avg_max_power.pivot_table(values='MAX_POWER', index=['CITY'],
columns=['FUEL', 'TRANSMISSION'])

# Convert the values to percentages
stacked_data = stacked_data.apply(lambda x: x / x.sum() * 100, axis=1)

# Generate a color palette that is sorted based on the values of the data
palette = sns.color_palette('crest', len(top_cities)).as_hex()[::-1]
sns.set_style('white', {'axes.spines.bottom': False,
                        'axes.spines.left': False,
                        'axes.spines.right': False,
                        'axes.spines.top': False})

# Create the stacked horizontal bar chart
ax = stacked_data.plot(kind='barh', stacked=True, figsize=(10, 6), legend=False, color=palette)

# Add labels and a title to the plot
plt.xlabel('Average Max Power', fontweight='bold')
plt.ylabel('City', fontweight='bold')
plt.title('Average Max Power by Fuel and Transmission in Top 10 Cities', fontweight='bold')

# Set the legend location and size
ax.legend(loc='upper right', bbox_to_anchor=(1.2, 1), fontsize=8)

# Remove the background and grid lines
ax.set_facecolor('none')
ax.grid(False)
```

```
# Set the x-axis tick labels to percentage format
ax.set_xticklabels(['{}%'.format(int(x)) for x in ax.get_xticks()])


# Show the plot
plt.tight_layout()
plt.savefig('max_power.pdf')
```



```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset into a Pandas DataFrame
used_car_dataset = pd.read_csv("UserCarDataset_final.csv")

# Filter the dataset to only include the top 10 cities
top_cities = used_car_dataset['CITY'].value_counts().nlargest(10).index
used_car_top_cities = used_car_dataset[used_car_dataset['CITY'].isin(top_cities)]

# Calculate the average max power by fuel and transmission in each city
avg_max_power = used_car_top_cities.groupby(['CITY', 'FUEL', 'TRANSMISSION'])['MAX_POWER'].mean().reset_index()

# Pivot the table to create a stacked dataset
stacked_data = avg_max_power.pivot_table(values='MAX_POWER', index=['CITY'], columns=['FUEL', 'TRANSMISSION'])

# Convert the values to percentages
stacked_data = stacked_data.apply(lambda x: x / x.sum() * 100, axis=1)

# Generate a color palette that is sorted based on the values of the data
palette = sns.color_palette('crest', len(top_cities)).as_hex()[::-1]

sns.set_style('white', {'axes.spines.bottom': False,
                        'axes.spines.left': False,
                        'axes.spines.right': False,
                        'axes.spines.top': False})

# Create the stacked horizontal bar chart
ax = stacked_data.plot(kind='barh', stacked=True, figsize=(10, 6), legend=False, color=palette)

# Add labels and a title to the plot
plt.xlabel('Average Max Power', fontweight='bold')
plt.ylabel('City', fontweight='bold')
plt.title('Average Max Power by Fuel and Transmission in Top 10 Cities', fontweight='bold')

# Set the legend location and size
ax.legend(loc='upper right', bbox_to_anchor=(1.2, 1), fontsize=8)


# Remove the background and grid lines
ax.set_facecolor('none')
ax.grid(False)

# Set the x-axis tick labels to percentage format
ax.set_xticklabels(['{}%'.format(int(x)) for x in ax.get_xticks()])

# Show the plot
plt.tight_layout()
plt.savefig('max_power.pdf')
```

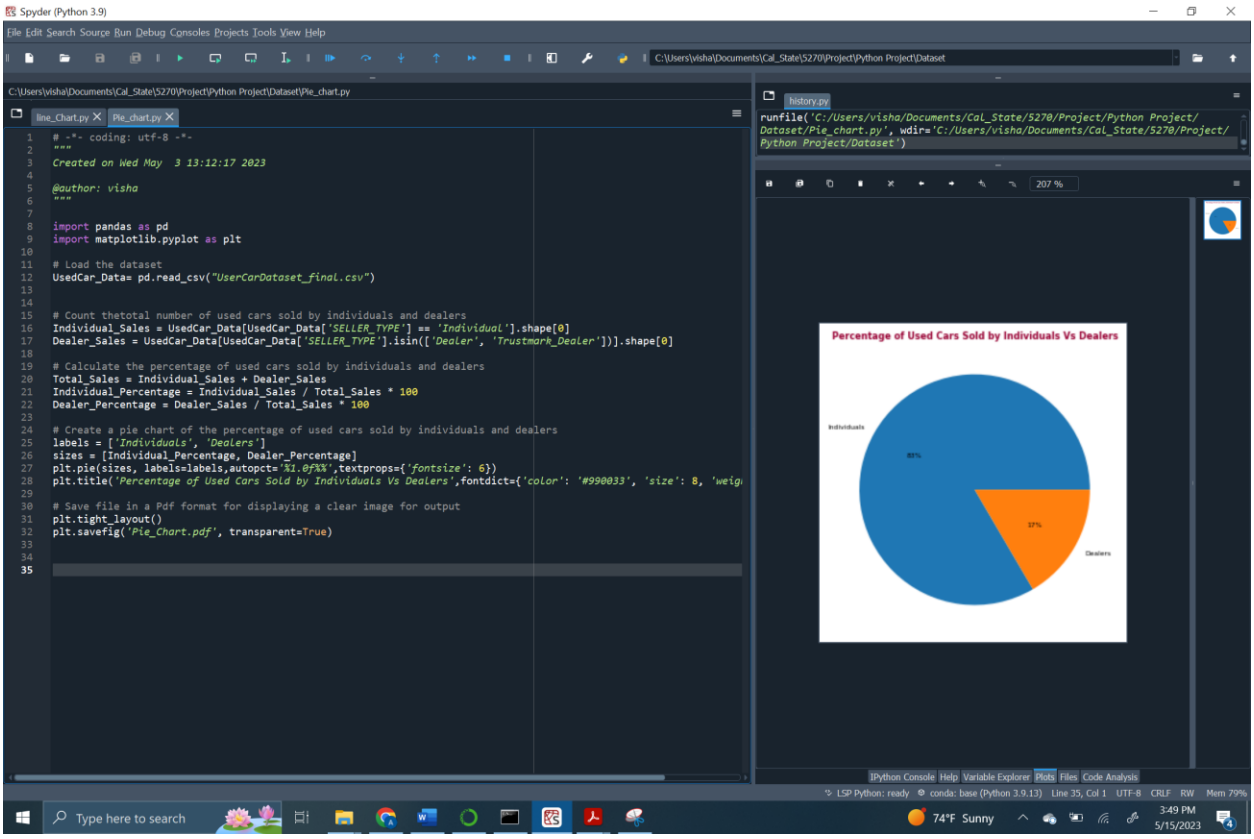
5) What percentage of used cars are sold by individuals versus dealers?



Insights:

The above pie chart shows the distribution of used car sales by individuals and dealers. The graph shows that Majority of used cars in the dataset were sold by individuals, accounting for 83% of all sales, and only 17% by dealers. Based on our analysis, we can conclude that individuals have a larger share of the used car market compared to dealers. This trend can be attributed to the fact that individuals may sell their own cars or sell cars on behalf of family and friends, and typically offer lower prices than dealers. In contrast, dealers typically purchase cars from individuals or other dealers, and add their own commission, resulting in higher prices for customers. Overall, the pie chart provides a clear visual representation of the distribution of used car sales by seller type, highlighting the dominance of individual sales in the dataset.

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
UsedCar_Data= pd.read_csv("UserCarDataset_final.csv")

# Count thetotal number of used cars sold by individuals and dealers
Individual_Sales = UsedCar_Data[UsedCar_Data['SELLER_TYPE'] == 'Individual'].shape[0]
Dealer_Sales = UsedCar_Data[UsedCar_Data['SELLER_TYPE'].isin(['Dealer',
'Trustmark_Dealer'])].shape[0]

# Calculate the percentage of used cars sold by individuals and dealers
Total_Sales = Individual_Sales + Dealer_Sales
Individual_Percentage = Individual_Sales / Total_Sales * 100
Dealer_Percentage = Dealer_Sales / Total_Sales * 100

# Create a pie chart of the percentage of used cars sold by individuals and dealers
labels = ['Individuals', 'Dealers']
sizes = [Individual_Percentage, Dealer_Percentage]
plt.pie(sizes, labels=labels,autopct='%1.0f%%',textprops={'fontsize': 6})
plt.title('Percentage of Used Cars Sold by Individuals Vs Dealers',fontdict={'color': '#990033',
'size': 8, 'weight': 'bold'})

# Save file in a Pdf format for displaying a clear image for output
plt.tight_layout()
plt.savefig('Pie_Chart.pdf', transparent=True)
```




```python
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
UsedCar_Data= pd.read_csv("UserCarDataset_final.csv")


# Count thetotal number of used cars sold by individuals and dealers
Individual_Sales = UsedCar_Data[UsedCar_Data['SELLER_TYPE'] == 'Individual'].shape[0]
Dealer_Sales = UsedCar_Data[UsedCar_Data['SELLER_TYPE'].isin(['Dealer', 'Trustmark_Dealer'])].shape[0]

# Calculate the percentage of used cars sold by individuals and dealers
Total_Sales = Individual_Sales + Dealer_Sales
Individual_Percentage = Individual_Sales / Total_Sales * 100
Dealer_Percentage = Dealer_Sales / Total_Sales * 100

# Create a pie chart of the percentage of used cars sold by individuals and dealers
labels = ['Individuals', 'Dealers']
sizes = [Individual_Percentage, Dealer_Percentage]
plt.pie(sizes, labels=labels,autopct='%1.0f%%',textprops={'fontsize': 6})
plt.title('Percentage of Used Cars Sold by Individuals Vs Dealers',fontdict={'color': '#990033', 'size': 8, 'weight': 'bold'})

# Save file in a Pdf format for displaying a clear image for output
plt.tight_layout()
plt.savefig('Pie_Chart.pdf', transparent=True)
```

Conclusion

In conclusion, our analysis of the "Used-Car Data" dataset has revealed interesting insights into the US used car market. Maruti emerged as the top-selling car between 1994 and 2020, while Volkswagen had the lowest presence. We noticed limited listings of cars for sale in various cities, indicating potential regional variations in market demand. The year 2017 stood out with the highest number of used car sales, and we observed differences in maximum power across the top 10 cities based on fuel type and transmission. Notably, the majority (83%) of used cars were sold by individuals, with dealers accounting for only 17% of sales. These findings offer valuable information for consumers, dealers, researchers, and industry professionals, helping them make informed decisions and understand market dynamics in the used car industry.

References

[1] ACV Auctions. (2022, June 24). *Used car sales data: Used car market statistics: ACV auctions*. ACV Auction. https://www.acvauctions.com/blog/used-car-sales-data/

[2] *Industry market research, reports, and Statistics*. IBISWorld. (n.d.). https://www.ibisworld.com/united-states/market-research-reports/used-car-dealers-industry/