**Question 2:**

**Intent:**

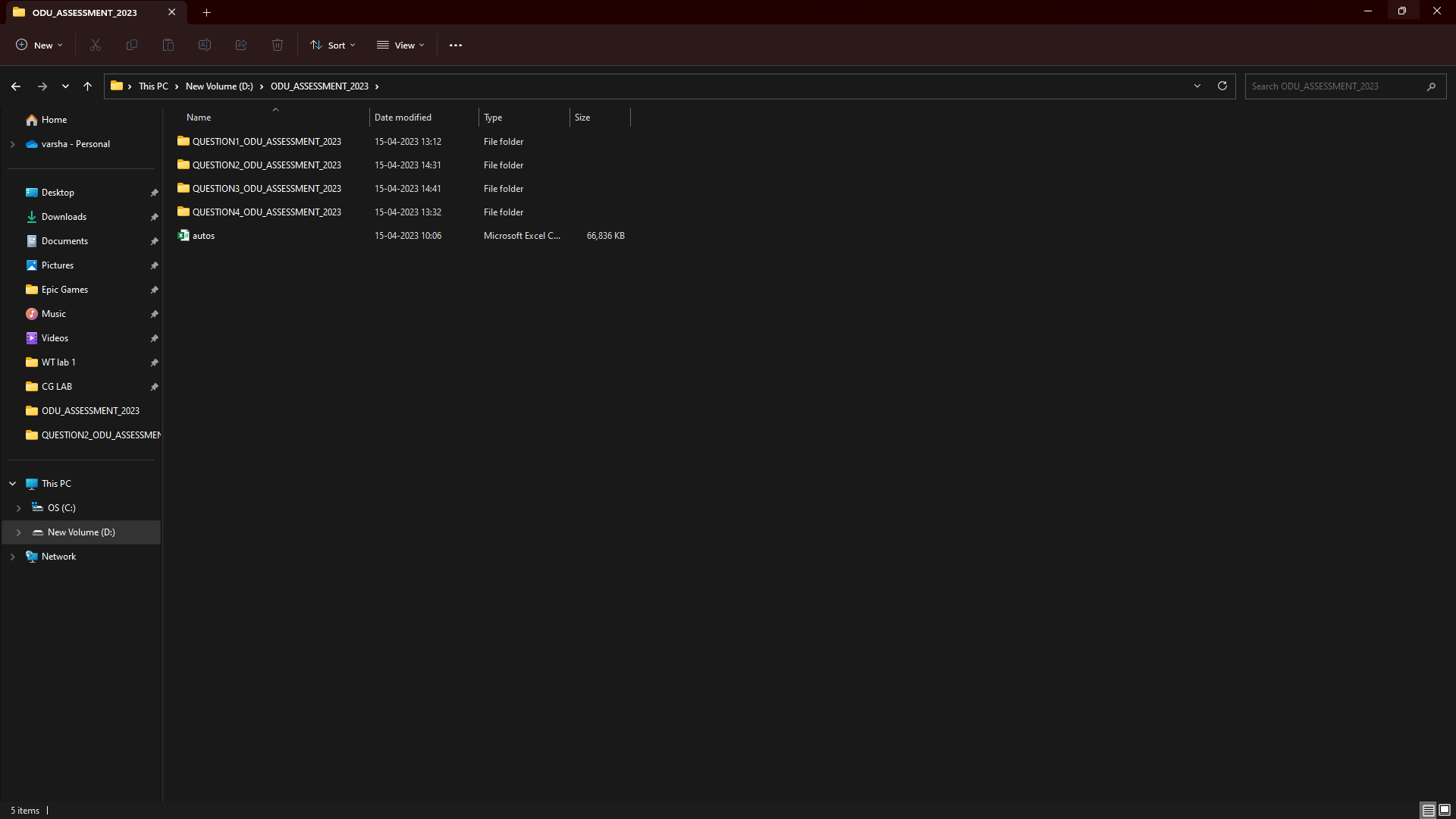
We would like you to learn the basics of python and data science to load a dataset, read it and perform some operations to find multiple mathematical metrics such as average, maximum, minimum and such. Here is a dataset for autos.   
<https://drive.google.com/file/d/1QP21K5tiJAjt5NA7W2FxSe9Wam9-tIcQ/view?usp=sharing>   
  
Flow:

1. Download this dataset.
2. Write basic python script to load csv and read it  as dataframe
3. Use the dataframe to perform following:
   1. Find Average price of autos ( using **price** column of dataset)
   2. Print the list of different possible types of **VehicleType** found in dataset
   3. Calculate and print lowest **yearOfRegistration**  and highest **yearOfRegistration**
   4. Find and print standard deviation of column **kilometer**
   5. Draw a bar graph to represent count of different type of column **brand**
   6. Find out which **VehicleType** is sold minimum and maximum
   7. Create a pie chart to represent different types of  **gearbox** count

SOLUTION:

1. Download this dataset.

Simple download the data set to your any drives.



1. Write basic python script to load csv and read it as dataframe.

import pandas as pd

import matplotlib.pyplot as plt

*# A basic python script to load and read it as dataframe.*

auto\_var = pd.read\_csv('autos.csv', encoding='latin-1')

import pandas as pd

import matplotlib.pyplot as plt

(to install pandas and matplotlib use-

Pip install pandas

Pip install mtplotlib

\*for windows)

auto\_var = pd.read\_csv('autos.csv', encoding='latin-1')

The dataframe is assigned to a variable named autos,it loads the ‘autos.csv’file as a pandas dataframe usinf read\_csv() function.

The encoding='latin-1' parameter is used to specify the character encoding, which is necessary because the file may contain non-ASCII characters ( ex:é,ñ,ç,ö,田,時,月,π,©,€).

3a. Find Average price of autos ( using **price** column of dataset)

*# Find the average price of autos using price column of database*

avg\_price = auto\_var['price'].mean()

print("Average price of autos:", round(avg\_price, 3))

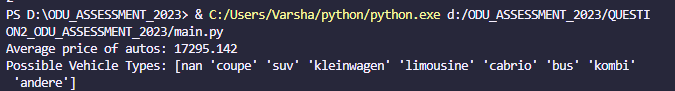
the code calculates and prints the average price of autos using the mean() method from Pandas library. It first selects the 'price' column of the auto\_var DataFrame and then applies the mean() method to it. The round() function is used to round off the result to 3 decimal places.

3b. Print the list of different possible types of **VehicleType** found in dataset

*# Print the list of different possible types of VehicleType in dataset*

vehicle\_type = auto\_var['vehicleType'].unique()

print("Possible Vehicle Types:", vehicle\_type)

solution:

3c. Calculate and print lowest **yearOfRegistration**  and highest **yearOfRegistration**

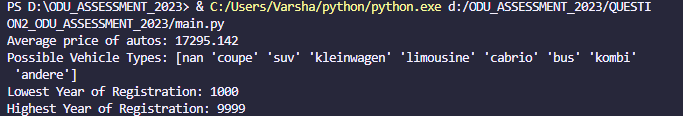
*#Calculate and print lowest yearOfRegistration and highest yearOfRegistration*

lowest\_reg\_year = auto\_var['yearOfRegistration'].min()

highest\_reg\_year = auto\_var['yearOfRegistration'].max()

print("Lowest Year of Registration:", lowest\_reg\_year)

print("Highest Year of Registration:", highest\_reg\_year)

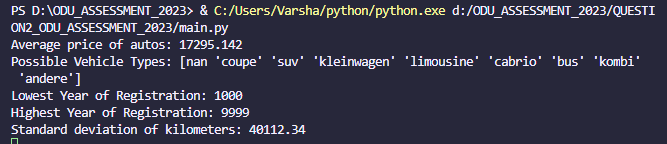
solution:

3d. Find and print standard deviation of column **kilometer**

*# Find and print standard deviation of column kilometer*

kilometer\_std\_dev = auto\_var['kilometer'].std()

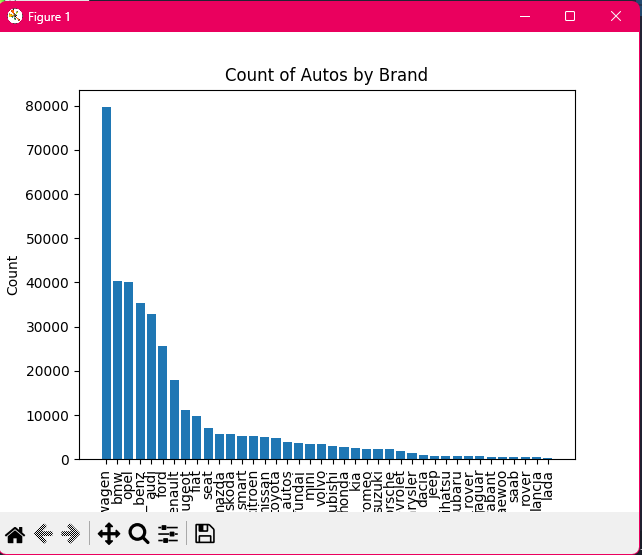
print("Standard deviation of kilometers:", round(kilometer\_std\_dev, 2))



3e. Draw a bar graph to represent count of different type of column **brand**

1. *# Draw a bar graph to represent count of different type of column brand*
2. brand\_counts = auto\_var['brand'].value\_counts()
3. plt.bar(brand\_counts.index, brand\_counts.values)
4. plt.xticks(rotation=90)
5. plt.xlabel('Brand')
6. plt.ylabel('Count')
7. plt.title('Count of Autos by Brand')
8. plt.show()

solution:



3f. Find out which **VehicleType** is sold minimum and maximum

*# Find out which VehicleType is sold minimum and maximum*

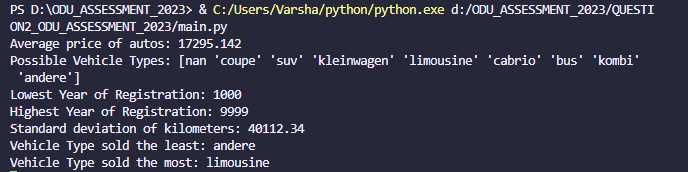
min\_sold\_vehicle\_type = auto\_var.groupby('vehicleType')['dateCreated'].count().idxmin()

max\_sold\_vehicle\_type = auto\_var.groupby('vehicleType')['dateCreated'].count().idxmax()

print("Vehicle Type sold the least:", min\_sold\_vehicle\_type)

print("Vehicle Type sold the most:", max\_sold\_vehicle\_type)

solution:



3g. Create a pie chart to represent different types of  **gearbox** count .

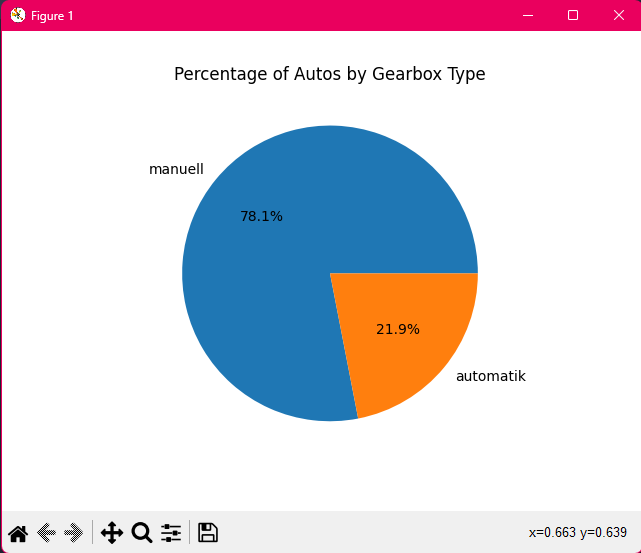
*# Create a pie chart to represent different types of gearbox count*

gearbox\_counts = auto\_var['gearbox'].value\_counts()

plt.pie(gearbox\_counts.values, labels=gearbox\_counts.index, autopct='%1.1f%%')

plt.title('Percentage of Autos by Gearbox Type')

plt.show()



CODE:

import pandas as pd

import matplotlib.pyplot as plt

*# A basic python script to load and read it as dataframe.*

auto\_var = pd.read\_csv('autos.csv', encoding='latin-1')

*# Find the average price of autos using price column of database*

avg\_price = auto\_var['price'].mean()

print("Average price of autos:", round(avg\_price, 3))

*# Print the list of different possible types of VehicleType found in dataset*

vehicle\_type = auto\_var['vehicleType'].unique()

print("Possible Vehicle Types:", vehicle\_type)

*#Calculate and print lowest yearOfRegistration and highest yearOfRegistration*

lowest\_reg\_year = auto\_var['yearOfRegistration'].min()

highest\_reg\_year = auto\_var['yearOfRegistration'].max()

print("Lowest Year of Registration:", lowest\_reg\_year)

print("Highest Year of Registration:", highest\_reg\_year)

*# Find and print standard deviation of column kilometer*

kilometer\_std\_dev = auto\_var['kilometer'].std()

print("Standard deviation of kilometers:", round(kilometer\_std\_dev, 2))

*# Draw a bar graph to represent count of different type of column brand*

brand\_counts = auto\_var['brand'].value\_counts()

plt.bar(brand\_counts.index, brand\_counts.values)

plt.xticks(rotation=90)

plt.xlabel('Brand')

plt.ylabel('Count')

plt.title('Count of Autos by Brand')

plt.show()

*#  Find out which VehicleType is sold minimum and maximum*

min\_sold\_vehicle\_type = auto\_var.groupby('vehicleType')['dateCreated'].count().idxmin()

max\_sold\_vehicle\_type = auto\_var.groupby('vehicleType')['dateCreated'].count().idxmax()

print("Vehicle Type sold the least:", min\_sold\_vehicle\_type)

print("Vehicle Type sold the most:", max\_sold\_vehicle\_type)

*# Create a pie chart to represent different types of gearbox count*

gearbox\_counts = auto\_var['gearbox'].value\_counts()

plt.pie(gearbox\_counts.values, labels=gearbox\_counts.index, autopct='%1.1f%%')

plt.title('Percentage of Autos by Gearbox Type')

plt.show()