IC 152

Computing & Data Science

Lab 7

Mohit Verma

B20215

Instructor- Dr.Padmanabhan Rajan

Questions & Answers

Q1. List all the manufacturers mentioned in the Cars93 data set in alphabetical order.

Ans 1) Manufacturers in Alphabetical order is shown-

```
0          Acura
1          Acura
2           Audi
3           Audi
4            BMW
         ...
88    Volkswagen
89    Volkswagen
90    Volkswagen
91         Volvo
92         Volvo
Name: Manufacturer, Length: 93, dtype: object

In [65]:
```

Code for Output:

```
# -*- coding: utf-8 -*-
"""
Created on Fri Apr 16 09:07:32 2021

@author: verma
"""

import pandas as pd

cars=pd.read_csv("cars93.csv")
df=pd.DataFrame(cars)

k=cars.sort_values("Manufacturer")
print(k.Manufacturer)
```

Q 2) Print all details of the costliest car of each of the 'Types'.

Ans 2) The details of costliest cars of each type are shown below-

```
  Manufacturer    Model   Type  ...  Weight   Origin          Make
0        Acura  Integra  Small  ...    2705  non-USA  Acura Integra

[1 rows x 27 columns]
     Manufacturer Model     Type  ...  Weight   Origin               Make
58  Mercedes-Benz  300E  Midsize  ...    3525  non-USA  Mercedes-Benz 300E

[1 rows x 27 columns]
     Manufacturer Model     Type  ...  Weight   Origin               Make
57  Mercedes-Benz  190E  Compact  ...    2920  non-USA  Mercedes-Benz 190E

[1 rows x 27 columns]
   Manufacturer     Model   Type  ...  Weight  Origin             Make
51      Lincoln  Town_Car  Large  ...    4055     USA  Lincoln Town_Car

[1 rows x 27 columns]
   Manufacturer     Model    Type  ...  Weight  Origin               Make
18    Chevrolet  Corvette  Sporty  ...    3380     USA  Chevrolet Corvette

[1 rows x 27 columns]
   Manufacturer   Model Type  ...  Weight   Origin          Make
86       Toyota  Previa  Van  ...    3785  non-USA  Toyota Previa

[1 rows x 27 columns]
```

Code for output –

```
 7
 8    import pandas as pd
 9    import numpy as np
10    import matplotlib.pyplot as plt
11
12    cars=pd.read_csv("cars93.csv")
13    df=pd.DataFrame(cars)
14    df=df.drop(['Unnamed: 0'],axis=1)
15
16    for i in df['Type'].unique():
17     k = df.groupby('Type').get_group(i)
18     print(k[k['Price']==k['Price'].max()])
19
```

Q3) Write a function that asks you to enter the name of a manufacturer and print all the models produced by that manufacturer. Print an appropriate error message if the input is invalid.

Ans 3) Here is the function –

```
Enter the name of Manufacturer below -
Volkswagen
   Manufacturer    Model     Type  ...  Weight   Origin               Make
87   Volkswagen      Fox    Small  ...    2240  non-USA      Volkswagen Fox
88   Volkswagen  Eurovan      Van  ...    3960  non-USA  Volkswagen Eurovan
89   Volkswagen   Passat  Compact  ...    2985  non-USA   Volkswagen Passat
90   Volkswagen  Corrado   Sporty  ...    2810  non-USA  Volkswagen Corrado

[4 rows x 27 columns]

In [35]: runfile('C:/Users/verma/OneDrive/Desktop/LAB 7/untitled0.py', wdir='C:/Users/verma/
OneDrive/Desktop/LAB 7')
Reloaded modules: jupyter_client.session, zmq.eventloop, zmq.eventloop.ioloop,
tornado.platform, tornado.platform.asyncio, tornado.gen, zmq.eventloop.zmqstream,
jupyter_client.jsonutil, jupyter_client.adapter, pandas.io.formats.string,
IPython.lib.guisupport, IPython.external, IPython.external.qt_for_kernel,
IPython.utils.version, IPython.external.qt_loaders, spyder, spyder.pil_patch, PIL,
PIL._version, PIL.Image, PIL.ImageMode, PIL.TiffTags, PIL._binary, PIL._util, PIL._imaging,
cffi, cffi.api, cffi.lock, cffi.error, cffi.model

Enter the name of Manufacturer below -
abcd
Following Manufacturer doesn't exist.

In [36]:
```

Here is the code for the function-

```
7
8   import pandas as pd
9
10  cars=pd.read_csv("cars93.csv")
11  df=pd.DataFrame(cars)
12  df=df.drop(['Unnamed: 0'],axis=1)
13
14  o=input("Enter the name of Manufacturer below - \n")
15  Manufacturer=list(df.Manufacturer)
16  if o in Manufacturer:
17   print(df[df["Manufacturer"]==o])
18  else:
19   print("Following Manufacturer doesn't exist.")
```

Q4) Print the total count of cars per manufacturer. Print it in the following format:

Manufacturer Count

Mfr1 N1

Mfr2 N2

Mfr3 N3

Ans 4) The output is-

```
In [39]: runfile('C:/Users/verma/OneDrive/Desktop/LAB 7/untitled0.py', wdir='C:/Users/verma/OneDrive/Desktop/LAB 7')
Reloaded modules: jupyter_client.session, zmq.eventloop, zmq.eventloop.ioloop, tornado.platform, tornado.platform.asyn
pandas.io.formats.string, spyder, spyder.pil_patch, PIL, PIL._version, PIL.Image, PIL.ImageMode, PIL.TiffTags, PIL._bi
IPython.lib.guisupport, IPython.external, IPython.external.qt_for_kernel, IPython.utils.version, IPython.external.qt_

------------------------------
 Manufacturer Count
------------------------------

    Chevrolet 8
         Ford 8
        Dodge 6
      Pontiac 5
        Mazda 5
   Volkswagen 4
        Buick 4
       Toyota 4
   Oldsmobile 4
       Nissan 4
      Hyundai 4
       Subaru 3
        Honda 3
      Mercury 2
        Acura 2
   Mitsubishi 2
Mercedes-Benz 2
      Lincoln 2
         Audi 2
        Lexus 2
          Geo 2
        Eagle 2
     Chrysler 2
     Cadillac 2
        Volvo 2
     Infiniti 1
     Plymouth 1
         Saab 1
       Saturn 1
     Chrylser 1
       Suzuki 1
          BMW 1

In [40]:
```

Code for output –

```
4
5   @author: verma
6   """
7
8   import pandas as pd
9
10  cars=pd.read_csv("cars93.csv")
11  df=pd.DataFrame(cars)
12  df=df.drop(['Unnamed: 0'],axis=1)
13
14  k=df.value_counts("Manufacturer")
15  df_value_counts=pd.DataFrame(k)
16  df_value_counts=df_value_counts.reset_index()
17  df_value_counts.columns=["",""]
18  print("""
19  ------------------------------
20   Manufacturer Count
21  ------------------------------""")
22  print(df_value_counts.to_string(index=False))
23
```

Q5) List all the non-USA “Small” (‘Type’) cars (‘Make’) in alphabetical order.

Ans 5) Output data “Make” in alphabetical order.

```
0         Acura Integra
1             Geo Metro
2           Honda Civic
4       Hyundai Elantra
3         Hyundai Excel
5             Mazda 323
6         Mazda Protege
7     Mitsubishi Mirage
8         Nissan Sentra
9          Subaru Justy
10        Subaru Loyale
11         Suzuki Swift
12        Toyota Tercel
13      Volkswagen Fox
Name: Make, dtype: object
```

Code for output -

```
7
8   import pandas as pd
9
10  cars=pd.read_csv("cars93.csv")
11  df=pd.DataFrame(cars)
12  df=df.drop(['Unnamed: 0'],axis=1)
13
14  nusa=df[(df['Type']=="Small") & (df['Origin']=="non-USA")].reset_index(drop=True).sort_values("Make")
15  print(nusa.Make)
16
```

Q6) Group the cars by 'Manufacturer' and find the average price for each of the manufacturers by averaging the 'Price' of all the models produced by a particular manufacturer and then print all the 'Manufacturers' sorted (in ascending order) by average price.

Ans 6) The output is shown below-

```
    Manufacturer  Mean Price
0         Suzuki    8.600000
1            Geo   10.450000
2        Hyundai   10.475000
3         Saturn   11.100000
4         Subaru   12.933333
5       Plymouth   14.400000
6        Mercury   14.500000
7           Ford   14.962500
8          Dodge   15.700000
9          Eagle   15.750000
10       Pontiac   16.140000
11         Honda   16.466667
12        Nissan   17.025000
13        Toyota   17.275000
14    Oldsmobile   17.500000
15         Mazda   17.600000
16    Volkswagen   18.025000
17     Chevrolet   18.187500
18    Mitsubishi   18.200000
19      Chrylser   18.400000
20         Buick   21.625000
21      Chrysler   22.650000
22         Volvo   24.700000
23         Acura   24.900000
24          Saab   28.700000
25           BMW   30.000000
26         Lexus   31.600000
27          Audi   33.400000
28       Lincoln   35.200000
29      Cadillac   37.400000
30  Mercedes-Benz   46.900000
31       Infiniti   47.900000

In [54]:
```

Code for output-

```
7
8   import pandas as pd
9
10  cars=pd.read_csv("cars93.csv")
11  df=pd.DataFrame(cars)
12  df=df.drop(['Unnamed: 0'],axis=1)
13
14  dfgrouped=df.groupby("Manufacturer",as_index=False)["Price"].agg({"Mean Price":"mean"}).sort_values("Mean Price")
15  j=dfgrouped.reset_index(drop=True)
16  print(j)
17
```

Q7) Find the car models that are “Midsize” (‘Type’) and cost less than the average cost of all the cars (of all the types).

Ans7) The output is shown below-

```
          Model     Type
0        Legend  Midsize
1           100  Midsize
2          535i  Midsize
3       Riviera  Midsize
4       Seville  Midsize
5        Taurus  Midsize
6           Q45  Midsize
7         ES300  Midsize
8         SC300  Midsize
9   Continental  Midsize
10         300E  Midsize
11     Diamante  Midsize
12       Maxima  Midsize
13          850  Midsize

In [58]:
```

Code for output is shown below-

```
7
8   import pandas as pd
9
10  cars=pd.read_csv("cars93.csv")
11  df=pd.DataFrame(cars)
12  df=df.drop(['Unnamed: 0'],axis=1)
13
14  avgprice=df["Price"].mean()
15  data=df[(df["Type"]=="Midsize") & (df["Price"]>avgprice)]
16  print(data[["Model","Type"]].reset_index(drop=True))
17
```

Q8) Find the car models that have MPG.city and MPG.highway higher than the average of MPG.city and average of MPG.highway, respectively. Here, average should be calculated over all the cars.

Ans 8) The code for output is shown below-

```
7
8   import pandas as pd
9
10  cars=pd.read_csv("cars93.csv")
11  df=pd.DataFrame(cars)
12  df=df.drop(['Unnamed: 0'],axis=1)
13
14  avgMPGcit=df["MPG.city"].mean()
15  avgMPGhig=df["MPG.highway"].mean()
16  data1=df[(df["MPG.city"]>avgMPGcit)&(df["MPG.highway"]>avgMPGhig)]
17  print(data1["Model"].reset_index(drop=True))
18
```

The output is shown below-

```
0           Integra
1         Cavalier
2          Corsica
3             Colt
4           Summit
5          Festiva
6           Escort
7            Probe
8            Metro
9            Storm
10         Prelude
11           Civic
12          Accord
13           Excel
14          Scoupe
15             323
16         Protege
17             626
18          Mirage
19          Sentra
20          Altima
21         Achieva
22    Cutlass_Ciera
23            Laser
24           LeMans
25         Sunbird
26              SL
27           Justy
28          Loyale
29          Legacy
30           Swift
31          Tercel
32          Celica
33             Fox
Name: Model, dtype: object

In [60]:
```

Q9) Calculate the ratio of Max.Price and Min.Price for each of the car models and add a new column called 'Max_min_ratio' to the imported dataframe.
a) Print the cars for whom the ratio is equal to one.
b) Find the manufacturer whom the gap between largest and smallest max_min_ratio value is highest. Also, print the corresponding ratio values.
c) Print the car models (manufacturer and model) for whom the max_min_ratio is within ±1% of the mean of Max_min_ratio.

Ans 9) (a) The cars with ratio equal to one is shown below-

```
        Model Manufacturer
0     Riviera        Buick
1     Corsica    Chevrolet
2    Concorde     Chrylser
3    Imperial     Chrysler
4        RX-7        Mazda
5      Cougar      Mercury
6  Silhouette   Oldsmobile
```

(b) The manufacturer for whom the gap between largest and smallest max_min_ratio value is highest is shown below-

```
Manufacturer
Chevrolet      1.152941
```

(c) Car models with max_min_ratio within ±1% of the mean of Max_min_ratio is shown below-

```
   Manufacturer   Model
0         Acura  Legend
1         Mazda     626
2         Mazda     MPV
```

Code for output-

```
7
8   import pandas as pd
9   import numpy as np
10
11  cars=pd.read_csv("cars93.csv")
12  df=pd.DataFrame(cars)
13  df=df.drop(['Unnamed: 0'],axis=1)
14
15  Max=np.array((df["Max.Price"]))
16  Min=np.array((df["Min.Price"]))
17  ratio=Max/Min
18  df["Max_min_ratio"]=ratio
19  # (a)
20  data=df[df["Max_min_ratio"]==1]
21  print(data[["Model","Manufacturer"]].reset_index(drop=True))
22  # (b)
23  data1=df.groupby(["Manufacturer"]).agg(diff=("Max_min_ratio",lambda x:(max(x)-min(x))))
24  l=data1[data1["diff"]==data1["diff"].max()]
25  print(l)
26  # (c)
27  mean=df["Max_min_ratio"].mean()
28  data2=df[(df["Max_min_ratio"]>=(0.99*mean)) & (df["Max_min_ratio"]<=(1.01*mean))]
29  print("\n\n",data2[["Manufacturer","Model"]].reset_index(drop=True))
30
```

Q10) Calculate the average MPG ((MPG.city + MPG.highway)/2) for each of the car models and add a new column called 'AverageMPG' to the updated dataframe.

a) List the top 5 cars based on 'AverageMPG'.

b) Sort the "Midsize" cars by 'AverageMPG'.

c) Print all the non-USA cars (manufacturer and model) in alphabetical order whose 'AverageMPG' is higher than the average of the 'AverageMPG' column.

Ans 10) (a) The top 5 cars is shown below-

```
   AverageMPG Manufacturer    Model
0        28.0        Acura  Integra
1        21.5        Acura   Legend
2        23.0         Audi       90
3        22.5         Audi      100
4        26.0          BMW     535i
```

(b) Sorted Midsize cars is shown-

	AverageMPG	Manufacturer	Model
0	19.5	Infiniti	Q45
1	20.5	Lexus	SC300
2	20.5	Cadillac	Seville
3	21.0	Mitsubishi	Diamante
4	21.0	Lexus	ES300
5	21.5	Lincoln	Continental
6	21.5	Acura	Legend
7	22.0	Mercedes-Benz	300E
8	22.5	Mercury	Cougar
9	22.5	Audi	100
10	23.0	Buick	Riviera
11	23.0	Pontiac	Grand_Prix
12	23.5	Hyundai	Sonata
13	23.5	Nissan	Maxima
14	24.0	Dodge	Dynasty
15	24.0	Volvo	850
16	25.0	Chevrolet	Lumina
17	25.5	Ford	Taurus
18	25.5	Toyota	Camry
19	26.0	BMW	535i
20	26.5	Buick	Century
21	27.0	Oldsmobile	Cutlass_Ciera

(c) The non-USA cars (manufacturer and model) in alphabetical order whose ‘AverageMPG’ is higher than the average of the ‘AverageMPG’ column is shown below-

	Manufacturer	Model	AverageMPG
0	Acura	Integra	28.0
1	BMW	535i	26.0
2	Geo	Metro	48.0
3	Geo	Storm	33.0
4	Honda	Prelude	27.5
5	Honda	Civic	44.0
6	Honda	Accord	27.5
7	Hyundai	Excel	31.0
8	Hyundai	Scoupe	30.0
9	Mazda	626	30.0
10	Mazda	Protege	32.0
11	Mazda	323	33.0
12	Mitsubishi	Mirage	31.0
13	Nissan	Sentra	31.0
14	Nissan	Altima	27.0
15	Subaru	Justy	35.0
16	Subaru	Loyale	27.5
17	Subaru	Legacy	26.5
18	Suzuki	Swift	41.0
19	Toyota	Tercel	34.5
20	Toyota	Celica	28.5
21	Volkswagen	Fox	29.0

Code for all parts is shown-

```
7
8   import pandas as pd
9
10  cars=pd.read_csv("cars93.csv")
11  df=pd.DataFrame(cars)
12  df=df.drop(['Unnamed: 0'],axis=1)
13  # (a)
14  df["AverageMPG"]=(df["MPG.city"]+df["MPG.highway"])/2
15  print(df[["AverageMPG","Manufacturer","Model"]].head())
16  # (b)
17  o=df[(df["Type"]=="Midsize")].sort_values("AverageMPG")
18  print(o[["AverageMPG","Manufacturer","Model"]].reset_index(drop=True))
19  # (c)
20  l=df["AverageMPG"].mean()
21  h=df[(df["Origin"]=="non-USA")&(df["AverageMPG"]>l)].sort_values("Manufacturer")
22  print(h[["Manufacturer","Model","AverageMPG"]].reset_index(drop=True))
23
```

Q11) Create a new identifier for the cars listed in the dataset by concatenating the first three alphabets of 'Manufacturer' with the first three alphabets of 'Model' followed by the horsepower. Use hyphens to indicate the concatenation. For example, the identifier for the first row is "Acu-Int-140".
a) Add a new column called 'Identifier' to the updated dataframe and generate identifiers for each of the car models.
b) Save the updated dataframe as a new csv file by suffixing your roll number to the filename. For example, if your roll number is b20500, then the file name should be "Cars93b20500.csv". Also, note that the updated dataframe has now three new columns, namely, 'Max_min_ratio', 'AverageMPG', and 'Identifier'. To verify that load both the csv files and list the columns that are not present in the original csv file, i.e., "Cars93.csv".

Ans 11)(a) The identifier column with model is shown below-

```
     Identifier    Model
0   Acu-Int-140  Integra
1   Acu-Leg-200   Legend
2    Aud-90-172       90
3   Aud-100-172      100
4   BMW-535-208     535i
..          ...      ...
88  Vol-Eur-109  Eurovan
89  Vol-Pas-134   Passat
90  Vol-Cor-178  Corrado
91  Vol-240-114      240
92  Vol-850-168      850

[93 rows x 2 columns]

In [84]:
```

(b) New data frame along with old data frame. We can clearly observe three columns i.e. , ‘Max_min_ratio’, ‘AverageMPG’, and ‘Identifier’ are not present in the old dataframe.

```
Updated dataframe cars93b20215

    Unnamed: 0 Manufacturer    Model  ... Max_min_ratio  AverageMPG   Identifier
0            0        Acura  Integra  ...      1.457364        28.0  Acu-Int-140
1            1        Acura   Legend  ...      1.325342        21.5  Acu-Leg-200
2            2         Audi       90  ...      1.247104        23.0   Aud-90-172
3            3         Audi      100  ...      1.448052        22.5  Aud-100-172
4            4          BMW     535i  ...      1.527426        26.0  BMW-535-208
..         ...          ...      ...  ...           ...         ...          ...
88          88   Volkswagen  Eurovan  ...      1.367470        19.0  Vol-Eur-109
89          89   Volkswagen   Passat  ...      1.272727        25.5  Vol-Pas-134
90          90   Volkswagen  Corrado  ...      1.034934        21.5  Vol-Cor-178
91          91        Volvo      240  ...      1.077982        24.5  Vol-240-114
92          92        Volvo      850  ...      1.149194        24.0  Vol-850-168

[93 rows x 31 columns]

 Old dataframe cars93

    Unnamed: 0 Manufacturer    Model  ... Weight   Origin               Make
0            1        Acura  Integra  ...   2705  non-USA       Acura Integra
1            2        Acura   Legend  ...   3560  non-USA        Acura Legend
2            3         Audi       90  ...   3375  non-USA             Audi 90
3            4         Audi      100  ...   3405  non-USA            Audi 100
4            5          BMW     535i  ...   3640  non-USA            BMW 535i
..         ...          ...      ...  ...    ...      ...                 ...
88          89   Volkswagen  Eurovan  ...   3960  non-USA  Volkswagen Eurovan
89          90   Volkswagen   Passat  ...   2985  non-USA   Volkswagen Passat
90          91   Volkswagen  Corrado  ...   2810  non-USA  Volkswagen Corrado
91          92        Volvo      240  ...   2985  non-USA           Volvo 240
92          93        Volvo      850  ...   3245  non-USA           Volvo 850

[93 rows x 28 columns]

In [90]:
```

Code for output in part (a) and (b)-

```
 7
 8   import pandas as pd
 9   import numpy as np
10
11   cars=pd.read_csv("cars93.csv")
12   df=pd.DataFrame(cars)
13   df=df.drop(['Unnamed: 0'],axis=1)
14
15   Max=np.array((df["Max.Price"]))
16   Min=np.array((df["Min.Price"]))
17   ratio=Max/Min
18   df["Max_min_ratio"]=ratio
19
20   df["AverageMPG"]=(df["MPG.city"]+df["MPG.highway"])/2
21   #(a)
22   k=list(df.Manufacturer)
23   l=list(df.Model)
24   m=list(df.Horsepower)
25   f=[]
26   for b,i,h,g in zip(range(0,93),k,l,m):
27       f.append((k[b][0:3]+'-'+l[b][0:3]+'-'+str(m[b])))
28   df["Identifier"]=f
29   print(df[["Identifier","Model"]].reset_index(drop=True))
30   #(b)
31   df.to_csv("cars93b20215.csv")
32
33   cars_1=pd.read_csv("cars93b20215.csv")
34   df1=pd.DataFrame(cars_1)
35   print("Updated dataframe cars93b20215 \n\n",df1,"\n\n","Old dataframe cars93 \n\n",cars)
36
```

Q12) Compare the average prices (average price is calculated after grouping them based on the 'Type') of different types ("Small", "Midsize" etc.) of non-USA cars with that of the USA. Which one is cheaper in each category?

Ans 12) The cheaper car in each category is shown below-

```
       Type   Origin
73  Compact      USA
19    Large      USA
46  Midsize  non-USA
30    Small      USA
45   Sporty  non-USA
15      Van      USA

In [2]:
```

Code for output-

```
import pandas as pd

cars=pd.read_csv("cars93.csv")
df=pd.DataFrame(cars)
df=df.drop(['Unnamed: 0'],axis=1)

grouped= df.groupby("Type")
idx=[]
for name,group in grouped:
 idx.append(group["Price"].idxmin())
print(df.loc[idx][["Type","Origin"]])
```

Q13) Find the correlation of the area ('Length' x 'Width') with (a) 'Passengers', (b) 'Luggage.room', (c) 'Rear. Seat. room', and (d) 'Weight'. For each of the cases, did the sign of the correlation coefficient match your expectations?

Ans 13) The correlation between required column is shown below-

```
(a) Correlation of Area and Passengers  0.5080324162001505
(b)Correlation of Area and Luggage Room  0.723031757841467
(c)Correlation of Area and Rear Seat Room  0.536567343068455
(d)Correlation of Area and Weight  0.8668538434279643

In [3]:
```

Yes, the sign of correlation coefficient matches my expectation since in (a) with increase in Area there will be increase in no. of passenger's so positive sign.

Similarly, in (b), (c) & (d) with increase in area there will be increase in luggage room, rear seat room and weight will increase hence sign should be positive.

The code for output is shown below-

```
import pandas as pd

cars=pd.read_csv("cars93.csv")
df=pd.DataFrame(cars)
df=df.drop(['Unnamed: 0'],axis=1)

length=list(df["Length"])
width=list(df["Width"])
Area=[]
for l,w in zip(length,width):
    Area.append(l*w)
df["Area"]=Area
print(" (a) Correlation of Area and Passengers ",df["Area"].corr(df["Passengers"]))
print(" (b)Correlation of Area and Luggage Room ",(df["Area"].corr(df["Luggage.room"])))
print(" (c)Correlation of Area and Rear Seat Room ",df["Area"].corr(df["Rear.seat.room"]))
print(" (d)Correlation of Area and Weight ",df["Area"].corr(df["Weight"]))
```

Q14) Find the correlation between (a) 'EngineSize' and 'Horsepower', (b) 'Cylinders' and 'EngineSize', (c) 'Horsepower' and 'RPM', (d) 'EngineSize' and 'Fuel. Tank. capacity'. For each of the cases, did the sign of the correlation coefficient match your expectations?

Ans 14) The correlation between required column is shown below-

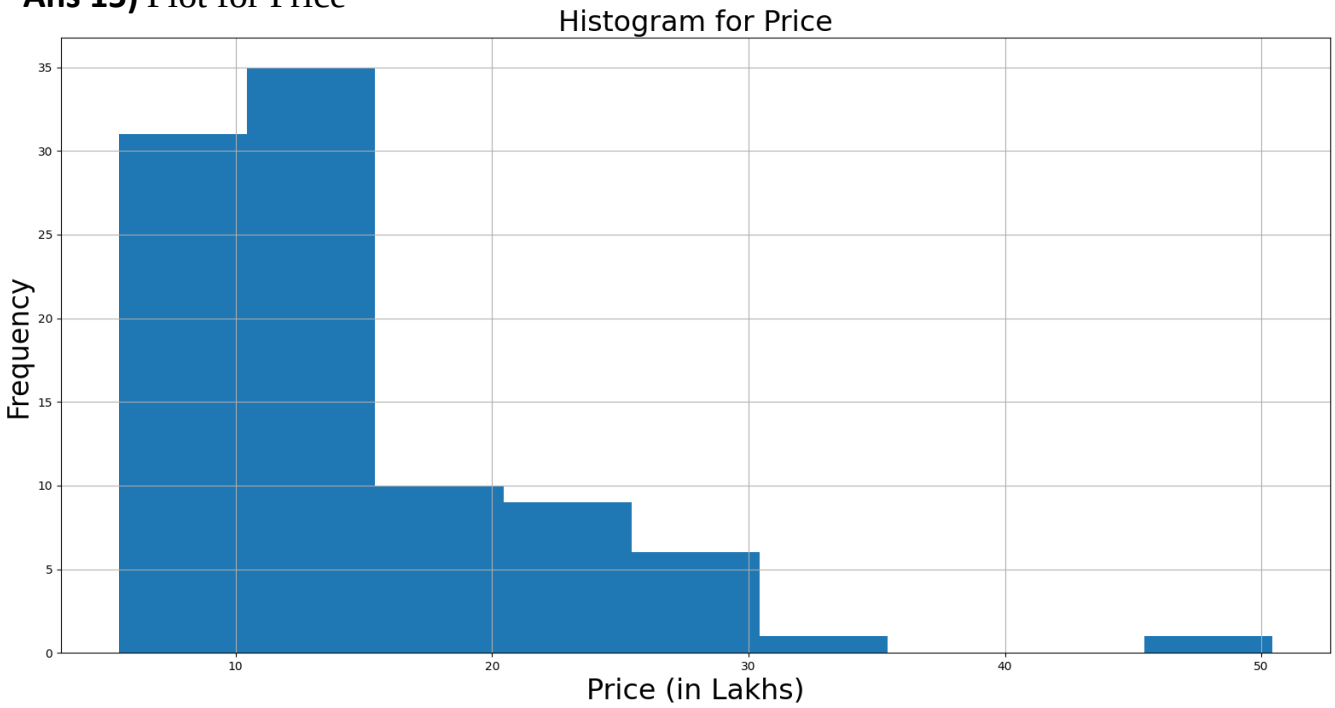
```
(a) Correlation of Engine Size and Horsepower  0.7900672754055842
(b) Correlation of Engine Size and Cylinders  0.890732316759097
(c) Correlation of RPM and Horsepower  -0.011519061905013327
(d) Correlation of Engine Size and Fuel tank capacity  0.7860213518366649

In [9]:
```

The code for output is shown below-

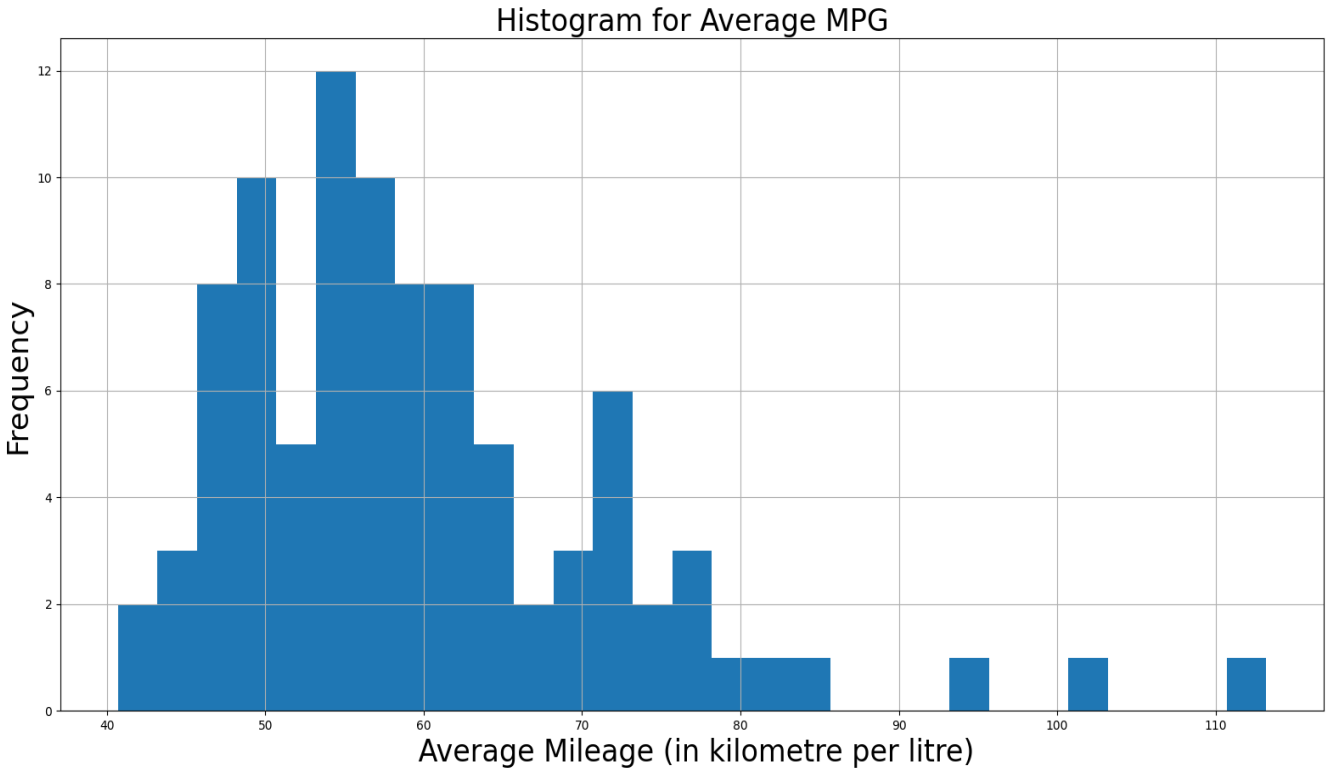
```
import pandas as pd

cars=pd.read_csv("cars93.csv")
df=pd.DataFrame(cars)
df=df.drop(['Unnamed: 0'],axis=1)

#Removing non Integer values
dfn=df.copy()
for col in["EngineSize","Horsepower","Cylinders","RPM","Fuel.tank.capacity"]:
    if (dfn[col].dtype!="int64") or (dfn[col].dtype!="float64") :
        for x in dfn.index:
            try:
                int(dfn.loc[x,col])
            except ValueError:
                dfn.drop(x,inplace=True)
        dfn[col]=dfn[col].apply(float)

print(" (a) Correlation of Engine Size and Horsepower ",(dfn["EngineSize"].corr(dfn["Horsepower"])))
print(" (b) Correlation of Engine Size and Cylinders ",(dfn["EngineSize"].corr(dfn["Cylinders"])))
print(" (c) Correlation of RPM and Horsepower ",dfn["RPM"].corr(dfn["Horsepower"]))
print(" (d) Correlation of Engine Size and Fuel tank capacity ",dfn["EngineSize"].corr(dfn["Fuel.tank.capacity"]))
```

Q15) Plot histograms of the following columns with a bin size (or width) mentioned next to the column name after converting the values into the unit mentioned inside the brackets.
a) Price, 5 (unit is Lakhs) [Assume $1 = INR 73.5]
b) AverageMPG, 2.5 (unit is Kilometres per litre)
c) Horsepower, 20 (unit is Kilowatt)
d) Area, 1 (unit is sq. metres)
e) Weight, 250 (unit is kg)

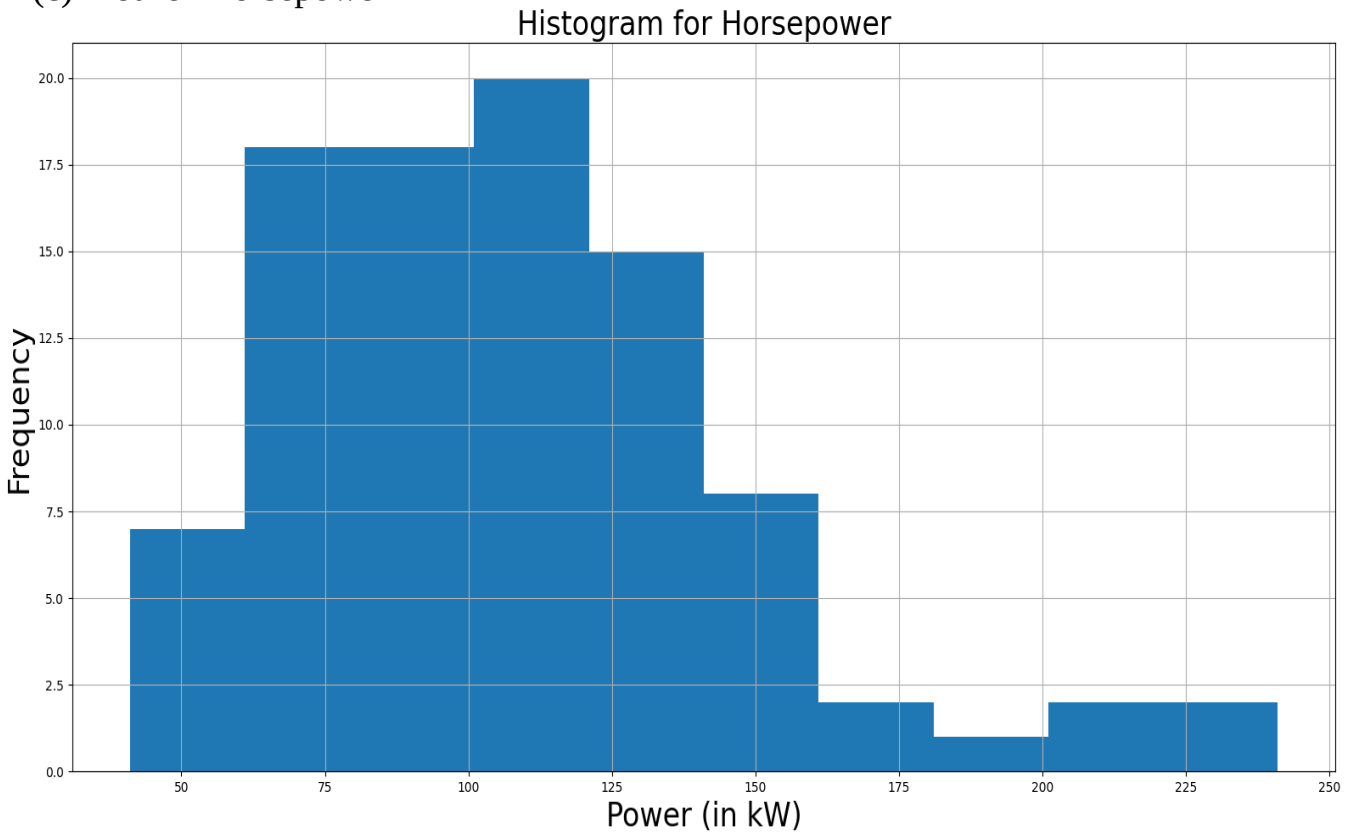
Ans 15) Plot for Price-



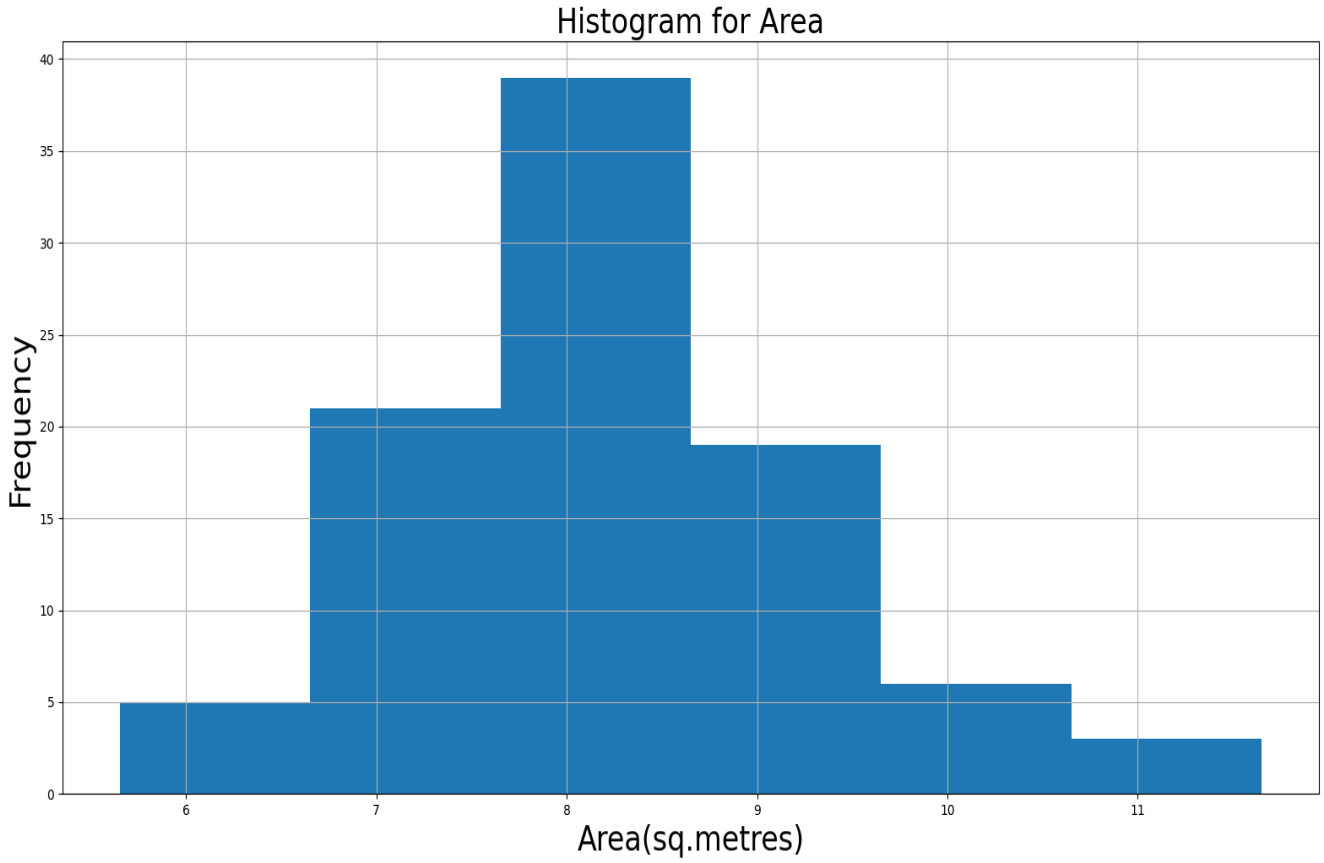
(b) Plot for AverageMPG –



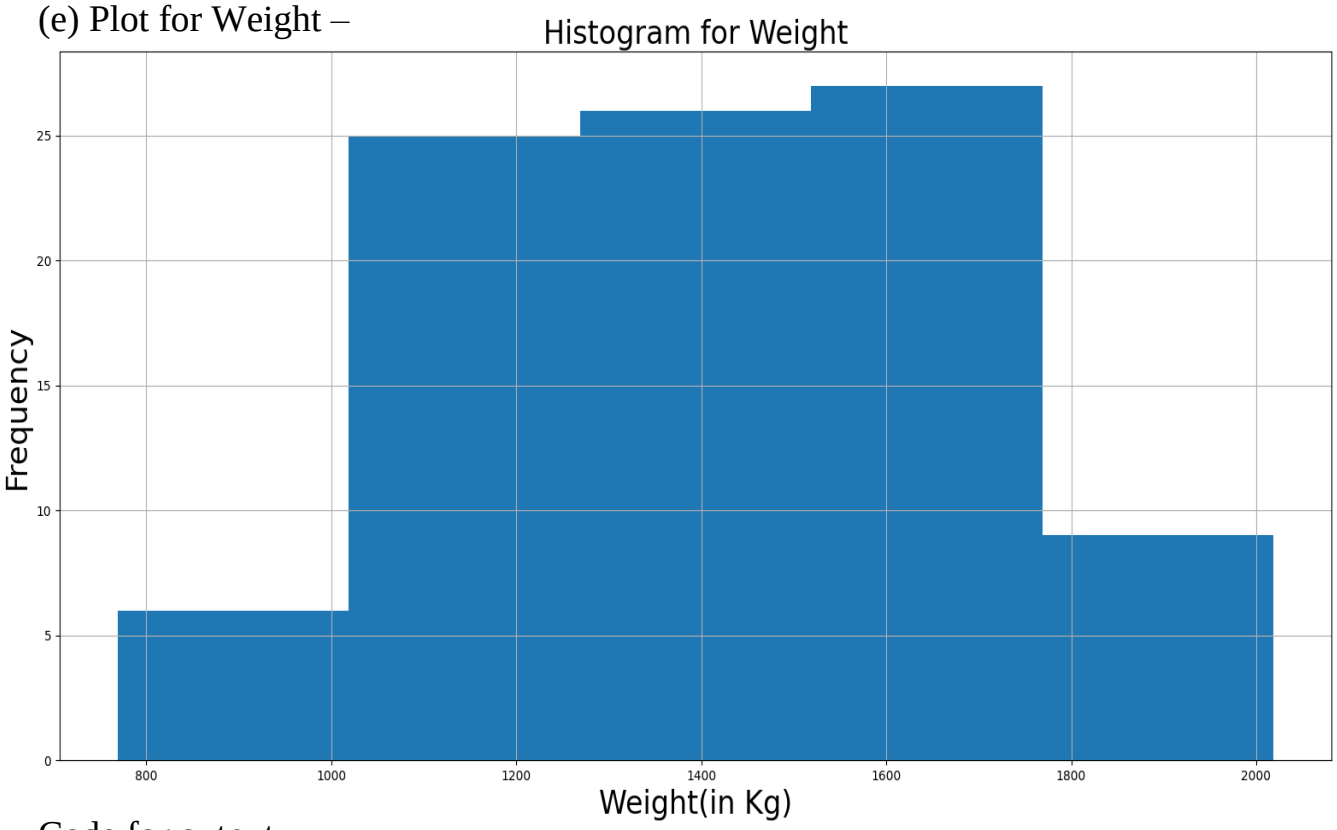
(c) Plot for Horsepower-



(d) Plot for Area-



(e) Plot for Weight –



Code for output-

```
7
8   import pandas as pd
9   import numpy as np
10  import matplotlib.pyplot as plt
11
12  cars=pd.read_csv("cars93.csv")
13  df=pd.DataFrame(cars)
14  df=df.drop(['Unnamed: 0'],axis=1)
15
16  dfn=df.copy()
17  dfn["Price"]=(73.5/100)*dfn["Price"]
18  dfn["Price"].hist(bins=np.arange(dfn["Price"].min(),dfn["Price"].max()+5,5))
19  plt.title("Histogram for Price",fontsize=24)
20  plt.xlabel("Price (in Lakhs)",fontsize=24)
21  plt.ylabel("Frequency",fontsize=24)
22
23  plt.figure()
24  dfn["AverageMPG"]=(dfn["MPG.city"]+dfn["MPG.highway"])/2
25  dfn["AverageMPG"]=dfn["AverageMPG"]/(0.43)
26  dfn["AverageMPG"].hist(bins=np.arange(dfn["AverageMPG"].min(),dfn["AverageMPG"].max()+2.5,2.5))
27  plt.title("Histogram for Average MPG",fontsize=24)
28  plt.xlabel(" Average Mileage (in kilometre per litre)",fontsize=24)
29  plt.ylabel("Frequency",fontsize=24)
30
31  plt.figure()
32  dfn["Horsepower"]=dfn["Horsepower"]*(0.7457)
33  dfn["Horsepower"].hist(bins=np.arange(dfn["Horsepower"].min(),dfn["Horsepower"].max()+20,20))
34  plt.title("Histogram for Horsepower",fontsize=24)
35  plt.xlabel("Power (in kW)",fontsize=24)
36  plt.ylabel("Frequency",fontsize=24)
37
38  plt.figure()
39  dfn["Area"]=(dfn["Length"]*dfn["Width"])*(0.00064516)
40  plt.hist(dfn["Area"],bins=np.arange(dfn["Area"].min(),dfn["Area"].max()+1,1))
41  plt.xlabel("Area(sq.metres)",fontsize=24)
42  plt.title("Histogram for Area",fontsize=24)
43  plt.grid()
44  plt.ylabel("Frequency",fontsize=24)
45
46  plt.figure()
47  dfn["Weight"]=dfn["Weight"]*(0.453592)
48  dfn["Weight"].hist(bins=np.arange(dfn["Weight"].min(),dfn["Weight"].max()+250,250))
49  plt.title("Histogram for Weight",fontsize=24)
50  plt.xlabel("Weight(in Kg)",fontsize=24)
51  plt.ylabel("Frequency",fontsize=24)
52
```