Roll No : 2 | Darshit Asalaliya | Msc.ICT : 3 | Introduction to Python and Data Science Assignment - 2

Here, I have used pandas, numpy and matplotlib library to perform operation (read data, chart etc..) on .csv file.

Program :

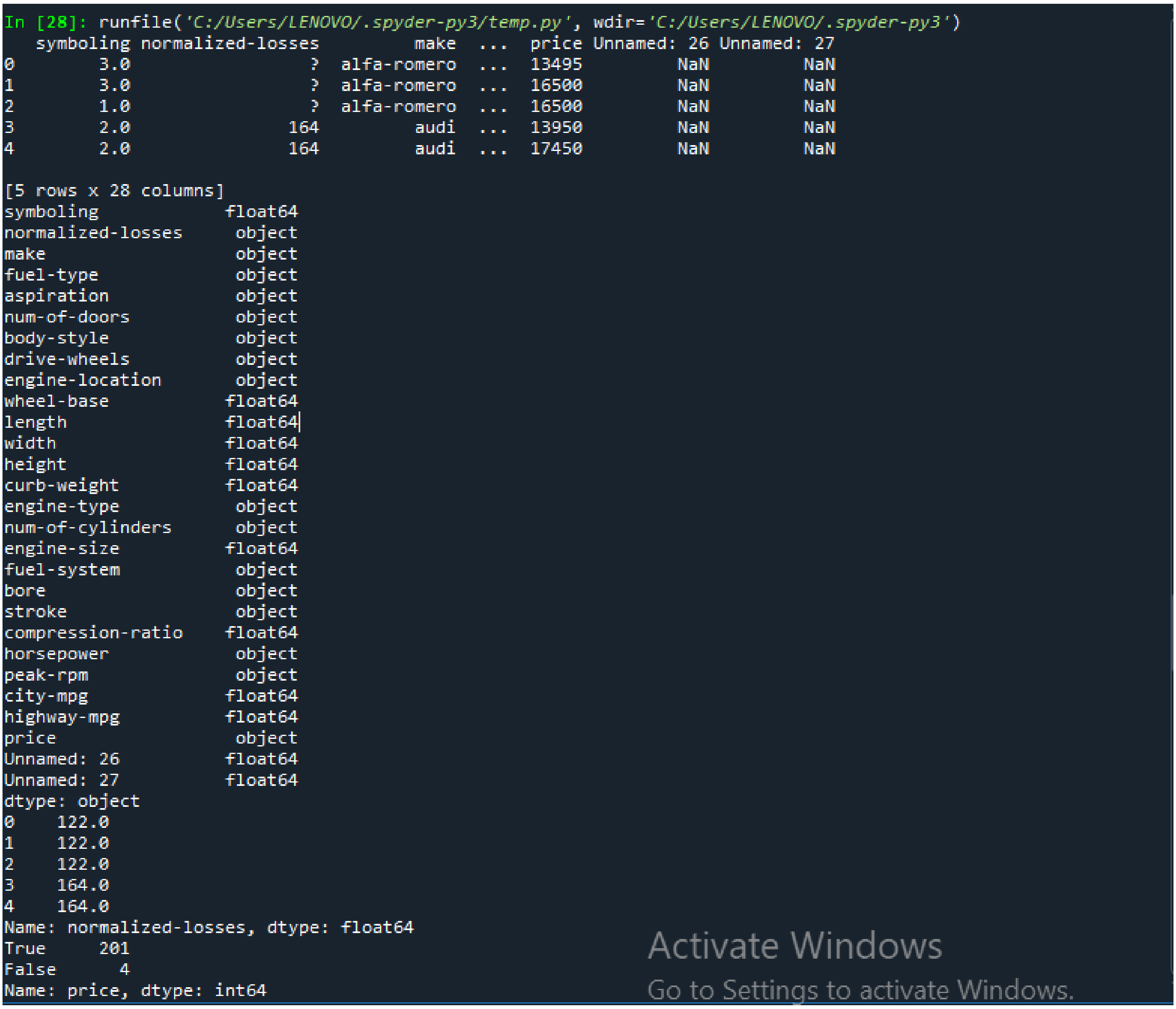
```
# import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# reading data
automobile = pd.read_csv('Automobile_data.csv')
print(automobile.head())

# # # getting datatypes
print(automobile.dtypes)



# # #setting missing to mean
# # # Setting the missing value to mean of normalized losses and conver the datatype to integer
nl = automobile['normalized-losses'].loc[automobile['normalized-losses'] != '?']
nlmean = nl.astype(str).astype(float).mean()
automobile['normalized-losses'] = automobile['normalized-losses'].replace('?',nlmean).astype(float)
print(automobile['normalized-losses'].head())

# #cleaning price data
# # Find out the number of values which are not numeric
print(automobile['price'].str.isnumeric().value_counts())



# #List out the values which are not numeric
print(automobile['price'].loc[automobile['price'].str.isnumeric() == False])

# #Setting the missing value to mean of price and convert the datatype to integer
price = automobile['price'].loc[automobile['price'] != '?']
pmean = price.astype(str).astype(float).mean()
automobile['price'] = automobile['price'].replace('?',pmean).astype(float)
```

```
print(automobile['price'].head())



# #Checking the numberic and replacing with mean value and conver the datatype to integer

automobile['horsepower'].str.isnumeric().value_counts()

horsepower = automobile['horsepower'].loc[automobile['horsepower'] != '?']

hpmean = horsepower.astype(str).astype(float).mean()

automobile['horsepower'] = automobile['horsepower'].replace('?',pmean).astype(float)



# #Checking the outlier of horsepower


print(automobile.loc[automobile['horsepower'] > 10000])


# #Excluding the outlier data for horsepower

print(automobile[np.abs(automobile.horsepower - automobile.horsepower.mean()) <= (3*automobile.horsepower.std())])


# #Find out the number of invalid value

print(automobile['bore'].loc[automobile['bore'] == '?'])


# #Replace the non-numeric value to null and conver the datatype

automobile['bore'] = pd.to_numeric(automobile['bore'],errors='coerce')

print(automobile.dtypes)



# #Replace the non-number value to null and convert the datatype

automobile['stroke'] = pd.to_numeric(automobile['stroke'],errors='coerce')

print(automobile.dtypes)



# #Convert the non-numeric data to null and convert the datatype

automobile['peak-rpm'] = pd.to_numeric(automobile['peak-rpm'],errors='coerce')

print(automobile.dtypes)


# #remove the records which are having the value '?'

automobile['num-of-doors'].loc[automobile['num-of-doors'] == '?']
```

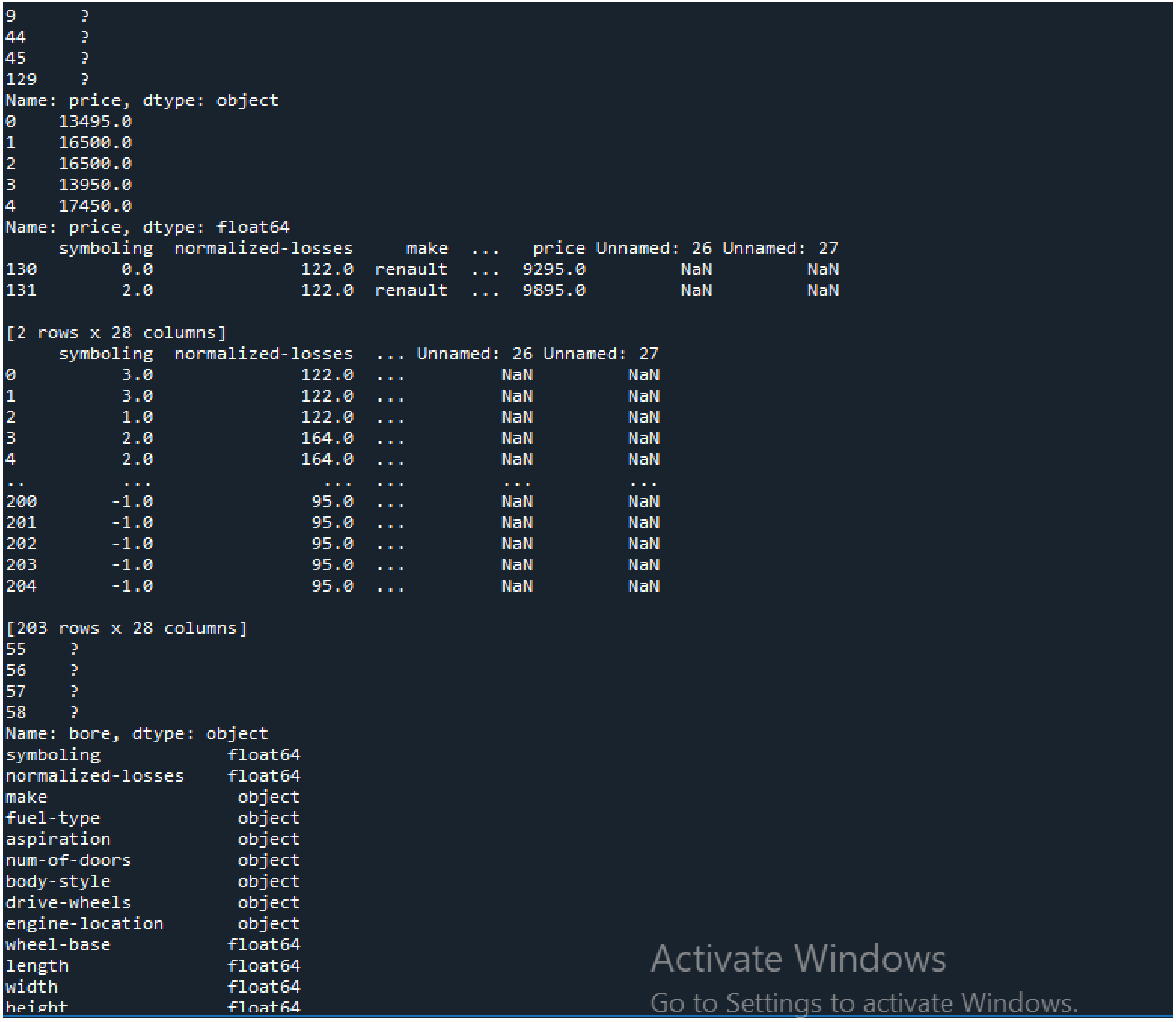
```
automobile = automobile[automobile['num-of-doors'] != '?']

automobile['num-of-doors'].loc[automobile['num-of-doors'] == '?']
```

```
automobile.symboling.hist(bins=6,color='green');

plt.title("Insurance risk ratings of vehicles")

plt.ylabel('Number of vehicles')

plt.xlabel('Risk rating');


automobile['normalized-losses'].hist(bins=5,color='orange');

plt.title("Normalized losses of vehicles")

plt.ylabel('Number of vehicles')

plt.xlabel('Normalized losses');


automobile['fuel-type'].value_counts().plot(kind='bar',color='purple')

plt.title("Fuel type frequence diagram")

plt.ylabel('Number of vehicles')

plt.xlabel('Fuel type');


automobile['aspiration'].value_counts().plot.pie(figsize=(6, 6), autopct='%.2f')

plt.title("Fuel type pie diagram")

plt.ylabel('Number of vehicles')

plt.xlabel('Fuel type');


automobile.horsepower[np.abs(automobile.horsepower-automobile.horsepower.mean())<=(3*automobile.horsepower.std())].hist(bins=5,color='red');

plt.title("Horse power histogram")

plt.ylabel('Number of vehicles')

plt.xlabel('Horse power');


automobile['num-of-doors'].value_counts().plot(kind='bar',color='purple')

plt.title("Number of doors frequency diagram")

plt.ylabel('Number of vehicles')

plt.xlabel('Number of doors');
```

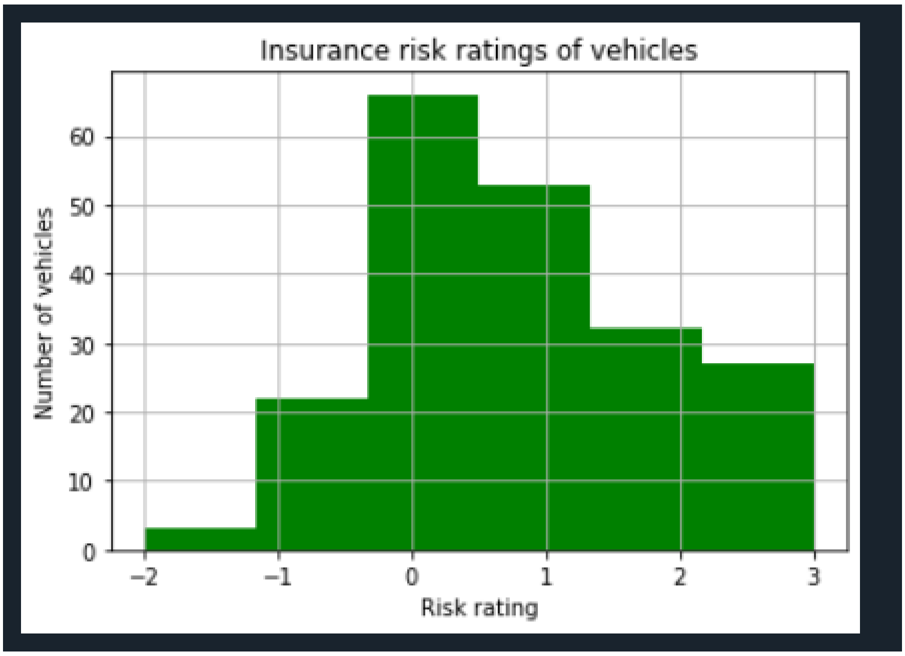
Output :

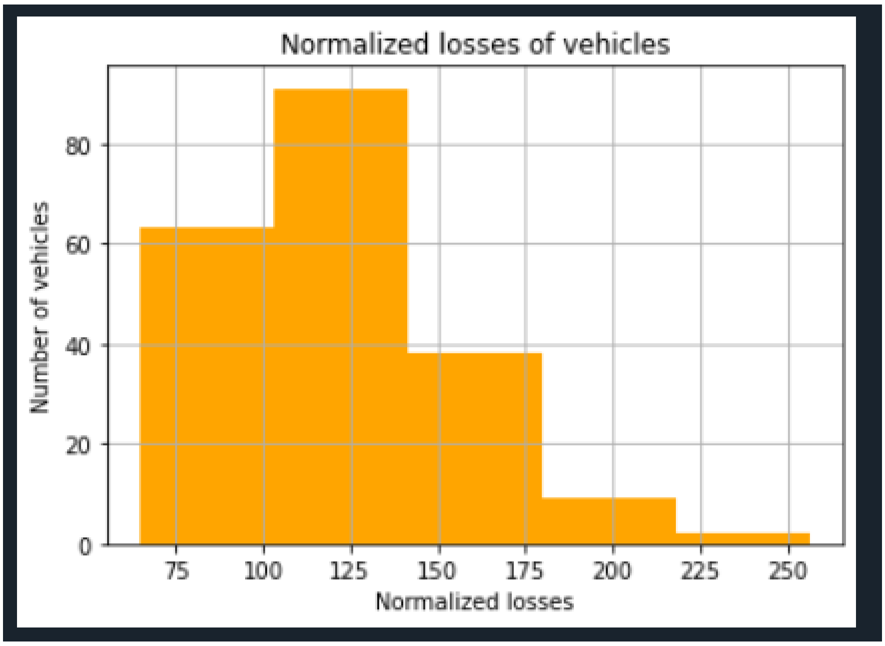


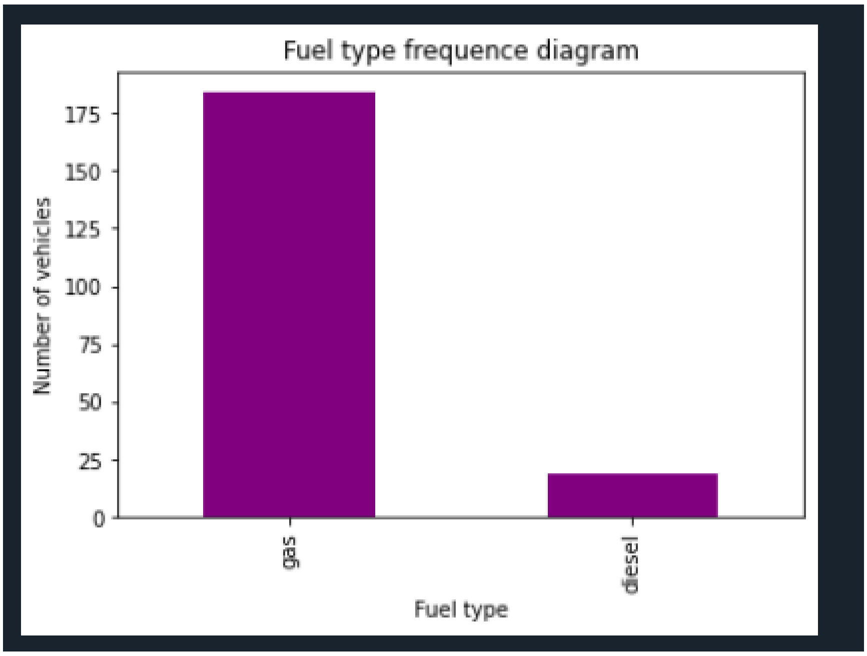
```
In [28]: runfile('C:/Users/LENOVO/.spyder-py3/temp.py', wdir='C:/Users/LENOVO/.spyder-py3')
   symboling normalized-losses         make  ...  price Unnamed: 26 Unnamed: 27
0        3.0                 ?  alfa-romero  ...  13495         NaN         NaN
1        3.0                 ?  alfa-romero  ...  16500         NaN         NaN
2        1.0                 ?  alfa-romero  ...  16500         NaN         NaN
3        2.0               164         audi  ...  13950         NaN         NaN
4        2.0               164         audi  ...  17450         NaN         NaN

[5 rows x 28 columns]
symboling            float64
normalized-losses     object
make                  object
fuel-type             object
aspiration            object
num-of-doors          object
body-style            object
drive-wheels          object
engine-location       object
wheel-base           float64
length               float64
width                float64
height               float64
curb-weight          float64
engine-type           object
num-of-cylinders      object
engine-size          float64
fuel-system           object
bore                  object
stroke                object
compression-ratio    float64
horsepower            object
peak-rpm              object
city-mpg             float64
highway-mpg          float64
price                 object
Unnamed: 26          float64
Unnamed: 27          float64
dtype: object
0    122.0
1    122.0
2    122.0
3    164.0
4    164.0
Name: normalized-losses, dtype: float64
True     201
False      4
Name: price, dtype: int64
```

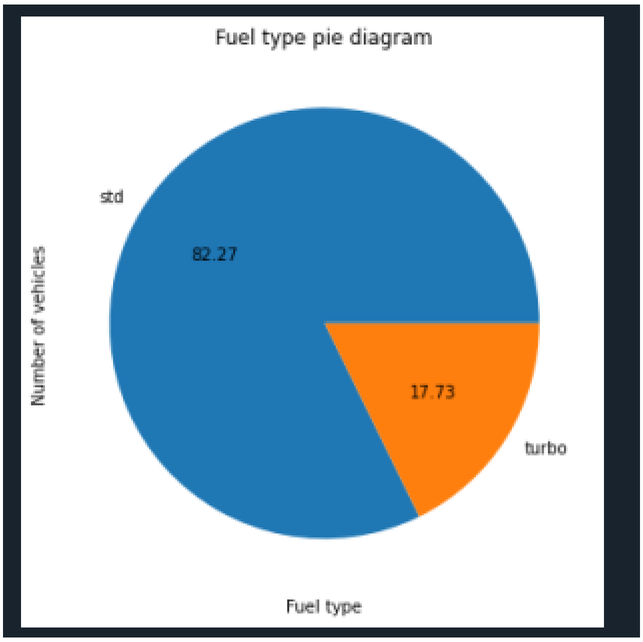


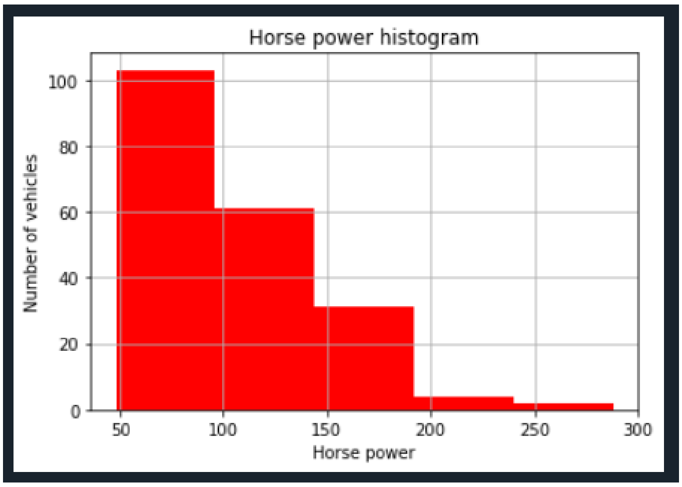
```
9      ?
44     ?
45     ?
129    ?
Name: price, dtype: object
0    13495.0
1    16500.0
2    16500.0
3    13950.0
4    17450.0
Name: price, dtype: float64
     symboling  normalized-losses     make  ...   price Unnamed: 26 Unnamed: 27
130        0.0              122.0  renault  ...  9295.0         NaN         NaN
131        2.0              122.0  renault  ...  9895.0         NaN         NaN

[2 rows x 28 columns]
     symboling  normalized-losses  ... Unnamed: 26 Unnamed: 27
0          3.0              122.0  ...         NaN         NaN
1          3.0              122.0  ...         NaN         NaN
2          1.0              122.0  ...         NaN         NaN
3          2.0              164.0  ...         NaN         NaN
4          2.0              164.0  ...         NaN         NaN
..         ...                ...  ...         ...         ...
200       -1.0               95.0  ...         NaN         NaN
201       -1.0               95.0  ...         NaN         NaN
202       -1.0               95.0  ...         NaN         NaN
203       -1.0               95.0  ...         NaN         NaN
204       -1.0               95.0  ...         NaN         NaN

[203 rows x 28 columns]
55    ?
56    ?
57    ?
58    ?
Name: bore, dtype: object
symboling            float64
normalized-losses    float64
make                  object
fuel-type             object
aspiration            object
num-of-doors          object
body-style            object
drive-wheels          object
engine-location       object
wheel-base           float64
length               float64
width                float64
height               float64
```

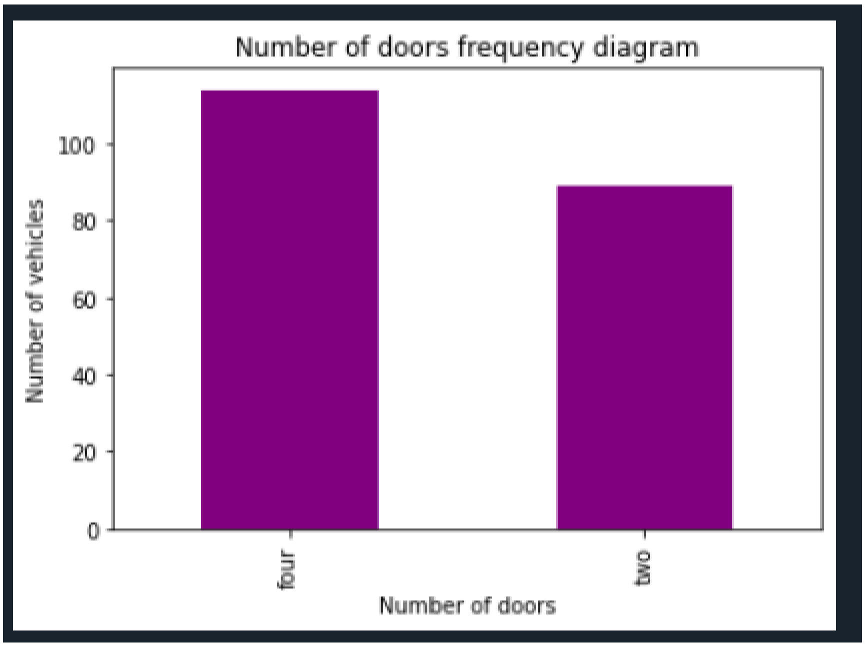



Normalized losses of vehicles
Number of vehicles
0
20
40
60
80
75
100
125
150
175
200
225
250
Normalized losses


Fuel type frequence diagram
Number of vehicles
0
25
50
75
100
125
150
175
gas
diesel
Fuel type


Fuel type pie diagram
std
82.27
17.73
turbo
Number of vehicles
Fuel type


Horse power histogram
100
80
60
40
20
0
50
100
150
200
250
300
Horse power
Number of vehicles


Number of doors frequency diagram
100
80
60
40
20
0
four
two
Number of doors
Number of vehicles