ASSIGNMENT 1
PRACTICAL DATA SCIENCE COSC2670

SUBMITTED BY-

VAMIKA PARDESHI
s3701024

INTRODUCTION-

The given data encapsulates three types of entities: the specification of an auto in terms of various characteristics, its assigned insurance risk rating, its normalized losses in use as compared to other cars. The second rating corresponds to the degree to which the auto is riskier than its price indicates. Cars are initially assigned a risk factor symbol associated with its price. Then, if it is riskier (or less), this symbol is adjusted by moving it up (or down) the scale. The third factor is the relative average loss payment per insured vehicle year.

This report will scrutinize and cover all the methods used, in order to prepare the data for the analysis, such as, cleaning of the data and dealing with all the potential errors/issues in the data. Also, the exploration of the data using respective visualizations.

METHODS-

1.DATA PREPARATION-

1.1 Addressing the task-

Firstly, the data has been loaded to python by downloading the data and moving it to the home directory and then using the respective CSV command. Secondly, in order to check the loaded data to the source CSV file, head() command is used, that displays the first few rows of the dataset.

In [2]:
```
#Loading the CSV file and printing the first few rows to check the loaded data to the source CSV file-
auto_mobile = 'Automobile.csv'
Data_AM = pan.read_csv(auto_mobile,sep='#',decimal='.',header = None,
                       names = ['Symboling','Normalized Losses','Make','Fuel Type','Aspiration','No.of Doors',
                                'Body Style','Drive Wheels','Engine Location','Wheel Base','Length','Width','Height',
                                 'Curb Weight','Engine Type','No.of Cylinders','Engine Size','Fuel System','Bore','Stroke',
                                'Compression Ratio','Horsepower','Peak RPM','City MPG','Highway MPG','Price'])
Data_AM.head()
```

Out[2]:

	Symboling	Normalized Losses	Make	Fuel Type	Aspiration	No.of Doors	Body Style	Drive Wheels	Engine Location	Wheel Base	Length	Width	Height	Curb Weight	Engine Type	No.of Cylinders	Eng
0	3	NaN	alfa-romero	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four	
1	3	NaN	alfa-romero	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four	
2	1	NaN	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv	six	
3	2	164.0	audi	gas	std	four	sedan	fwd	front	99.8	176.6	66.2	54.3	2337	ohc	four	
4	2	164.0	audi	gas	std	four	sedan	4wd	front	99.4	176.6	66.4	54.3	2824	ohc	five	

Then, before further dealing with the data, the types of the loaded columns and size of the dataset have been checked.

In [3]:
```
#Checking the types of the loaded columns-
Data_AM.dtypes
```

In [4]:
```
#Checking the size of the dataset
Data_AM.shape

#So, the dataset contains 238 rows(observations) and 26 columns.
```

1.2 Dealing with the potential issues/errors

After printing the data and having an overview of it, some of potential issues have been found like redundant whitespaces, incorrect names and formats.

1.2.1 Removing the redundant whitespace

Redundant whitespaces were found in few of the observations of No. of cylinders, Make, Drive wheels, Fuel types and Body style. Hence, those have been stripped using the following code-

In [6]:
```
#Removing the redundant whitespaces-
Data_AM['No.of Cylinders']=Data_AM['No.of Cylinders'].str.strip()
Data_AM['Make']=Data_AM['Make'].str.strip()
Data_AM['Drive Wheels']=Data_AM['Drive Wheels'].str.strip()
Data_AM['Fuel Type']=Data_AM['Fuel Type'].str.strip()
Data_AM['Body Style']=Data_AM['Body Style'].str.strip()
print(Data_AM)
```

1.2.2 Dealing with the typos/incorrect spellings

There were few typos in the 'Make' column like 'alfa-romero' and 'vol00112ov', and 'turrrrbo' in 'Aspiration' column that have been replaced with the correct spellings of 'alfa-romeo', 'volvo', and 'turbo' respectively. Also, there were font errors in the data, that has also been dealt with, using the following codes.

In [7]:
```
#Replacing the incorrect names and typos-
Data_AM['Make'].replace('alfa-romero','alfa-romeo',inplace=True)
Data_AM['Make'].replace('Nissan','nissan',inplace=True)
Data_AM['Make'].replace('Peugot','peugot',inplace=True)
Data_AM['Make'].replace('vol00112ov','volvo',inplace=True)
Data_AM['No.of Doors'].replace('fourR','four',inplace=True)
Data_AM['Aspiration'].replace('turrrrbo','turbo',inplace=True)
Data_AM['Engine Location'].replace('FRONT','front',inplace=True)
Data_AM['Engine Location'].replace('Front','front',inplace=True)
Data_AM['Engine Location'].replace('Rear','rear',inplace=True)
Data_AM['Engine Location'].replace('REAR','rear',inplace=True)
Data_AM['Fuel Type'].replace('diesel','Diesel',inplace=True)
Data_AM['Fuel Type'].replace('gas','Gas',inplace=True)
Data_AM['No.of Cylinders'].replace('Four','four',inplace=True)
Data_AM['Body Style'].replace('Sedan','sedan',inplace=True)
Data_AM['Body Style'].replace('Wagon','wagon',inplace=True)
print(Data_AM)
```

1.2.3 Dealing with the missing values

Since, there should be no missing values in the data for better exploration, so, missing values have been checked for each of the columns, and have got the following output-

In [8]:
```
#Checking the number of missing values for each column-
Data_AM.isnull().sum()
```

Out[8]:
```
Symboling            0
Normalized Losses    47
Make                 0
Fuel Type            0
Aspiration           0
No.of Doors          2
Body Style           0
Drive Wheels         0
Engine Location      0
Wheel Base           0
Length               0
Width                0
Height               0
Curb Weight          0
Engine Type          0
No.of Cylinders      0
Engine Size          0
Fuel System          0
Bore                 4
Stroke               4
Compression Ratio    0
Horsepower           2
Peak RPM             2
City MPG             0
Highway MPG          0
Price                4
dtype: int64
```

Its clear, that 'Normalized Losses' has the maximum number of missing values, i.e., 47. Whereas, bore, stroke and price have 4 and No. of doors, Horsepower and Peak RPM have 2. These missing values have been removed/reduced with the following code.

```
In [9]:
#Dealing with the missing values for the respective columns containing missing values using the mean values, grouped by the
Data_AM['Normalized Losses'] = Data_AM['Normalized Losses'].fillna(Data_AM.groupby('Make')['Normalized Losses'].transform('me
Data_AM['Price'] = Data_AM['Price'].fillna(Data_AM.groupby('Make')['Price'].transform('mean'))
Data_AM['Bore'] = Data_AM['Bore'].fillna(Data_AM.groupby('Make')['Bore'].transform('mean'))
Data_AM['Stroke'] = Data_AM['Stroke'].fillna(Data_AM.groupby('Make')['Stroke'].transform('mean'))
Data_AM['Horsepower'] = Data_AM['Horsepower'].fillna(Data_AM.groupby('Make')['Horsepower'].transform('mean'))
Data_AM['Peak RPM'] = Data_AM['Peak RPM'].fillna(Data_AM.groupby('Make')['Peak RPM'].transform('mean'))
Data_AM
```

```
In [10]:
#Rechecking for the missing value for Normalized losses-
Data_AM.isnull().sum()
#Still there are 10 more missing values even after the first treatement.

#Dealing with the missing values of Normalized Losses grouped by 'Body Style'-
Data_AM['Normalized Losses'] = Data_AM['Normalized Losses'].fillna(Data_AM.groupby('Body Style')['Normalized Losses'].transfo
Data_AM
```

After applying the above code, the number of missing values has been re-checked and have got the following output-

```
In [11]:
#Rechecking after the final treatment-
Data_AM.isnull().sum()
```

```
Out[11]: Symboling           0
         Normalized Losses   0
         Make                0
         Fuel Type           0
         Aspiration          0
         No.of Doors         2
         Body Style          0
         Drive Wheels        0
         Engine Location     0
         Wheel Base          0
         Length              0
         Width               0
         Height              0
         Curb Weight         0
         Engine Type         0
         No.of Cylinders     0
         Engine Size         0
         Fuel System         0
         Bore                0
         Stroke              0
         Compression Ratio   0
         Horsepower          2
         Peak RPM            2
         City MPG            0
         Highway MPG         0
         Price               0
         dtype: int64
```

Hence, the missing values has been removed successfully, and the data is ready for the further tasks.

RESULTS-

2.DATA EXPLORATION-

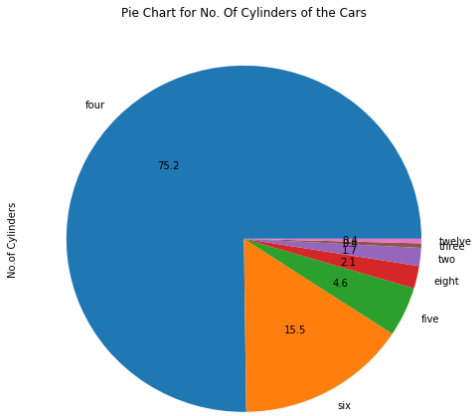
2.1 Exploring the data on the basis of nominal, ordinal and numerical values-

2.1.1 On the basis of Nominal values-

'No. Of Cylinders' has been chosen for exploring the data on the basis of Nominal values. There are 7 categories in this, i.e., two, three, four, five, six, eight and twelve, and 'Pie chart' has been used to visualize it.

With 75.2%, It is clear from the below pie chart that cars with 'four' cylinders are most commonly used, among all the 7 categories and then comes the cars with six cylinders. The least used car is the one with three cylinders.

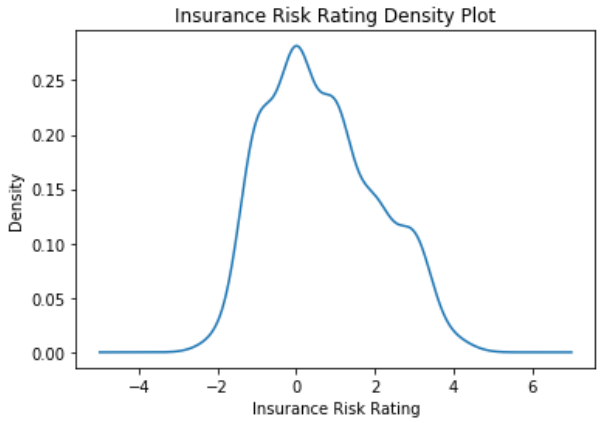
```
Data_AM['No.of Cylinders'].value_counts().plot(kind='pie',autopct='%.1f')
plt.title('Pie Chart for No. Of Cylinders of the Cars')
plt.show()
rcParams['figure.figsize'] = 5,5
```



2.1.2 On the basis of Ordinal values-

'Symboling' being the only ordinal value in the given dataset, it has been chosen for this task. Here, the value +3 indicates the auto being risky, and -3 indicates it to be pretty safe. Density plot has been used for visualizing the ordinal value. From the output, it can be seen that the risk factor for auto is moderate, being highest at '0'.

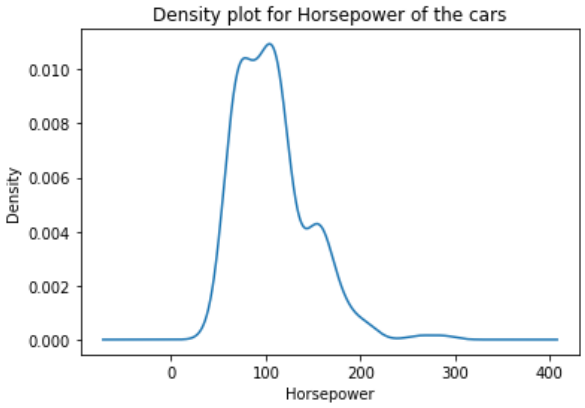
```
Data_AM['Symboling'].plot(kind='density')
plt.title('Insurance Risk Rating Density Plot')
plt.xlabel('Insurance Risk Rating')
plt.show()
```



2.1.3 On the basis of Numerical values-

Data consists of many numerical values, but since the ask is to choose just one of the numerical values, 'Horsepower' has been chosen for this task. From the output, it is clear that auto with horsepower around 100 is the highest.

```
Data_AM['Horsepower'].plot(kind='density')
plt.title('Density plot for Horsepower of the cars')
plt.xlabel('Horsepower')
plt.show()
```

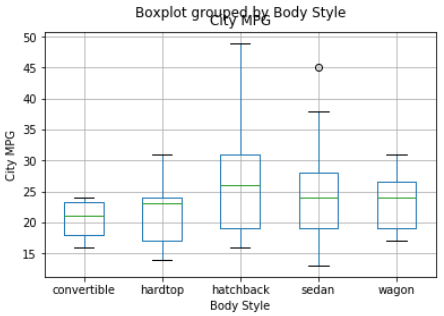


2.2 Exploring the relationships between the columns-

2.2.1 First pair of columns-

For the first pair of columns, 'City MPG' and 'Body Style' have been chosen. A boxplot has been plotted to show the relationship between the two.

```
Data_AM.dropna().boxplot(column='City MPG',by='Body Style')
plt.xlabel('Body Style')
plt.ylabel('City MPG')
plt.show()
rcParams['figure.figsize'] = 8,8
```

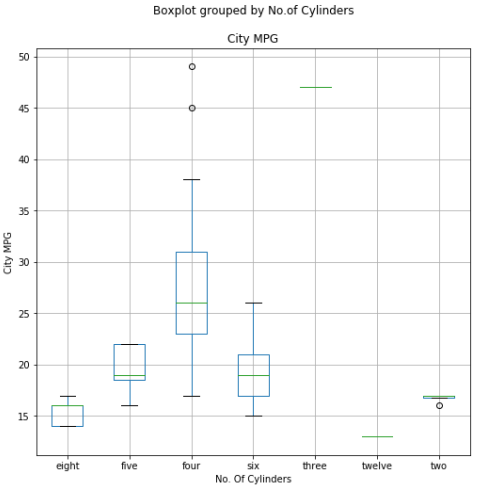


From the above plot, it is clear that, hatchback has the highest City MPG of around 25, while the lowest is observed in convertibles. Hardtop, sedan and wagon displays moderate City MPG.

2.2.2 Second pair of columns-

Second pair is of 'City MPG' and 'No. of cylinders'. Again, a boxplot is used to show the relationship.

```
Data_AM.dropna().boxplot(column='City MPG',by='No.of Cylinders')
plt.xlabel('No. Of Cylinders')
plt.ylabel('City MPG')
plt.show()
```



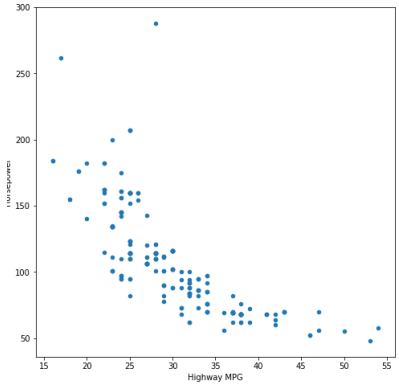
From the above plot, it can be seen that, cars with 'four' cylinders provide the highest City MPG, close to 25, whereas the lowest City MPG is for the cars with 'eight' cylinders.

2.2.3 Third pair of columns-

For the third pair, 'Highway MPG' and "Horsepower' have been chosen. This time, a Scatter plot has been plotted.

From the below output, it can be seen that the Scatter plot follows a strong negative or downward trend. In other words, it can be said that cars with higher horsepower will give lesser mileage on highways, whereas, cars with lower horsepower will provide higher mileage on highways. Here, the cars with around 50 horsepower is giving the highest mileage of around 55.

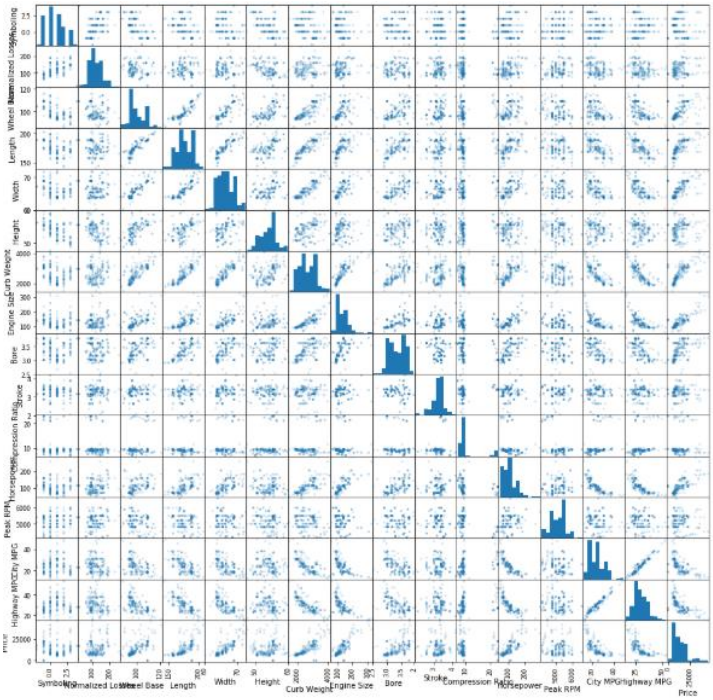
```
#Third Pair
Data_AM.plot(kind='scatter',x='Highway MPG',y='Horsepower')
plt.show()
```



2.3 Scatter matrix for all numerical values-

In order to check the correlation and the trends among all the numerical values, a scatter matrix has been created with the following code.

```
from pandas.tools.plotting import scatter_matrix
scatter_matrix(Data_AM,alpha=0.2,figsize=(16,16),diagonal='hist')
plt.show()
```



From the above scatter matrix, it can be observed that, the strongest positive correlation is found between Highway MPG and City MPG. Whereas, the strongest negative correlation is found that of horsepower with both ‘City MPG’ and “Highway MPG’. ‘Peal RPM’ has no correlation with any of the values.

CONCLUSION-

Checking and analyzing the data carefully is the most vital task for any of the data-driven decision making. For which, Data cleaning is the first and by far the most important step to be followed. Also, dealing with the appropriate issue/errors present in the data is equally important. Visualization of the data always provides a better detailed understanding for the developer or for any other reader. Hence, any data-driven decision is a series of important steps to be followed in order to reach the end objective/target.