In [5]:

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

In [6]:

```
#reading dataset
df = pd.read_csv('Automobile price data _Raw_.csv')
```

In [7]:

```
df.head()
```

Out[7]:

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wh b
0	3	?	alfa-romero	gas	std	two	convertible	rwd	front	
1	3	?	alfa-romero	gas	std	two	convertible	rwd	front	
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	front	
3	2	164	audi	gas	std	four	sedan	fwd	front	
4	2	164	audi	gas	std	four	sedan	4wd	front	

5 rows × 26 columns

In [8]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   symboling          205 non-null    int64  
 1   normalized-losses  205 non-null    object 
 2   make               205 non-null    object 
 3   fuel-type          205 non-null    object 
 4   aspiration         205 non-null    object 
 5   num-of-doors       205 non-null    object 
 6   body-style         205 non-null    object 
 7   drive-wheels       205 non-null    object 
 8   engine-location    205 non-null    object 
 9   wheel-base         205 non-null    float64
 10  length             205 non-null    float64
 11  width              205 non-null    float64
 12  height             205 non-null    float64
 13  curb-weight        205 non-null    int64  
 14  engine-type        205 non-null    object 
 15  num-of-cylinders   205 non-null    object 
 16  engine-size        205 non-null    int64  
 17  fuel-system        205 non-null    object 
 18  bore               205 non-null    object 
 19  stroke             205 non-null    object 
 20  compression-ratio  205 non-null    float64
 21  horsepower         205 non-null    object 
 22  peak-rpm           205 non-null    object 
 23  city-mpg           205 non-null    int64  
 24  highway-mpg        205 non-null    int64  
 25  price              205 non-null    object 
dtypes: float64(5), int64(5), object(16)
memory usage: 41.8+ KB
```

1.symboling

- +3 automobile is risky, -3 automobile is preety safe.

2.normalizes losses

- It is relative avg loss payment per insured vehical in a year.

3.price

- predict price using other variables in dataset

cleaning the data

In [9]:

```
#filling missing values with NaN
df = df.replace('?',np.nan)
```

price

not a categorical data

datatype is object,convert to float64

In [10]:

```
#predicting price is goal so drop rows with missing values
df = df.dropna(subset=['price'])
```

In [11]:

```
#converting from dtype object to float
df['price'] = df['price'].astype('float64')
df
```

Out[11]:

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location
0	3	NaN	alfa-romero	gas	std	two	convertible	rwd	front
1	3	NaN	alfa-romero	gas	std	two	convertible	rwd	front
2	1	NaN	alfa-romero	gas	std	two	hatchback	rwd	front
3	2	164	audi	gas	std	four	sedan	fwd	front
4	2	164	audi	gas	std	four	sedan	4wd	front
...	...	...	...	...	...	...	...	...	...
200	-1	95	volvo	gas	std	four	sedan	rwd	front
201	-1	95	volvo	gas	turbo	four	sedan	rwd	front
202	-1	95	volvo	gas	std	four	sedan	rwd	front
203	-1	95	volvo	diesel	turbo	four	sedan	rwd	front
204	-1	95	volvo	gas	turbo	four	sedan	rwd	front

201 rows × 26 columns

In [12]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 201 entries, 0 to 204
Data columns (total 26 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   symboling          201 non-null    int64
 1   normalized-losses  164 non-null    object
 2   make               201 non-null    object
 3   fuel-type          201 non-null    object
 4   aspiration         201 non-null    object
 5   num-of-doors       199 non-null    object
 6   body-style         201 non-null    object
 7   drive-wheels       201 non-null    object
 8   engine-location    201 non-null    object
 9   wheel-base         201 non-null    float64
 10  length             201 non-null    float64
 11  width              201 non-null    float64
 12  height             201 non-null    float64
 13  curb-weight        201 non-null    int64
 14  engine-type        201 non-null    object
 15  num-of-cylinders   201 non-null    object
 16  engine-size        201 non-null    int64
 17  fuel-system        201 non-null    object
 18  bore               197 non-null    object
 19  stroke             197 non-null    object
 20  compression-ratio  201 non-null    float64
 21  horsepower         199 non-null    object
 22  peak-rpm           199 non-null    object
 23  city-mpg           201 non-null    int64
 24  highway-mpg        201 non-null    int64
 25  price              201 non-null    float64
dtypes: float64(6), int64(5), object(15)
memory usage: 42.4+ KB
```

subtrating it from 201(price) ,we can get number of non-null values

symboling

In [13]:

```
#unique values in symboling
df['symboling'].unique()
```

Out[13]:

```
array([ 3,  1,  2,  0, -1, -2], dtype=int64)
```

In [14]:

```
df['symboling'].isnull().any()
```

Out[14]:

```
False
```

no null values.

datatype is int,no changes required here

Normalized losses

In [15]:

```
df['normalized-losses'].value_counts()
```

Out[15]:

```
161    11
91      8
150     7
104     6
128     6
134     6
85      5
102     5
94      5
65      5
95      5
74      5
103     5
168     5
122     4
148     4
106     4
93      4
118     4
115     3
137     3
83      3
125     3
101     3
154     3
145     2
119     2
188     2
153     2
164     2
158     2
89      2
81      2
108     2
194     2
110     2
113     2
129     2
192     2
197     2
87      2
78      1
186     1
231     1
256     1
90      1
142     1
77      1
121     1
107     1
98      1
Name: normalized-losses, dtype: int64
```

In [16]:

```
#check for null values
df['normalized-losses'].isnull().any()
```

Out[16]:

```
True
```

In [17]:

```
#convertinf to float as deimal data can also be considered
df['normalized-losses'] = df['normalized-losses'].astype('float64')
```

In [18]:

```
#filling null values by mean because it is not categorical data
mean_n1 = df['normalized-losses'].mean()
mean_n1
df['normalized-losses'] = df['normalized-losses'].fillna(mean_n1)
```

no. of doors

In [19]:

```
df['num-of-doors'].unique()
```

Out[19]:

```
array(['two', 'four', nan], dtype=object)
```

In [20]:

```
#convert to no. by replace
df['num-of-doors'] = df['num-of-doors'].replace(['two', 'four'], [2,4])

#count how many times occuring
df['num-of-doors'].value_counts()
```

Out[20]:

```
4.0    113
2.0     86
Name: num-of-doors, dtype: int64
```

In [21]:

```
#find if null values present
df['num-of-doors'].isnull().any()
```

Out[21]:

```
True
```

In [22]:

```
#4 is mode (frequent no. in the col).so filling rest null values with four  + categoric
al data
df['num-of-doors'] = df['num-of-doors'].fillna(4)
df['num-of-doors'].unique()
```

Out[22]:

```
array([2., 4.])
```

In [23]:

```
#its floating data,have to convert it to integer types :)
df['num-of-doors'] = df['num-of-doors'].astype('int64')
```

wheel base,length,width,height

this is no categorical data.

float type,which is good.

null values to be filled with median

In [24]:

```
#check null values for wheel base
df['wheel-base'].isnull().any()
```

Out[24]:

```
False
```

In [25]:

```
#check null values for wheel length
df['length'].isnull().any()
```

Out[25]:

```
False
```

In [26]:

```
#check null values for wheel width
df['width'].isnull().any()
```

Out[26]:

```
False
```

In [27]:

```
#check null values for wheel height
df['height'].isnull().any()
```

Out[27]:

```
False
```

no null values found,no changes required

curb-weight

not a categorical data

its dtype is int64 ,to be converted to float64

if null values found,replace with mean

In [28]:

```
#check for null values
df['curb-weight'].isnull().any()
```

Out[28]:

```
False
```

In [29]:

```
#convert to float64
df['curb-weight'] = df['curb-weight'].astype('float64')
```

No. of cylinders

categorical data

dtype = object to be converted to int64

if null values found, they replaced by mode

In [30]:

```
#checking for unique values
df['num-of-cylinders'].unique()
```

Out[30]:

```
array(['four', 'six', 'five', 'three', 'twelve', 'two', 'eight'],
      dtype=object)
```

In [31]:

```
#find values for each unique value
df['num-of-cylinders'].value_counts()
```

Out[31]:

```
four      157
six        24
five       10
two         4
eight       4
three       1
twelve      1
Name: num-of-cylinders, dtype: int64
```

In [32]:

```
#replace words with nos
df['num-of-cylinders'] = df['num-of-cylinders'].replace(['four', 'six', 'five', 'three'
, 'twelve', 'two', 'eight'],[4,6,5,3,12,2,8])
```

In [33]:

```
#check null values
df['num-of-cylinders'].isnull().any()
```

Out[33]:

```
False
```

In [34]:

```
#check dataype and values.impt convert to int
df['num-of-cylinders'].unique()
```

Out[34]:

```
array([ 4,  6,  5,  3, 12,  2,  8], dtype=int64)
```

engine size

not categorical data.

dtype is int64,convert to float64 to get decimal value prediction also

if null values fnd,replace with mean

In [35]:

```
df['engine-size'].unique()
```

Out[35]:

```
array([130, 152, 109, 136, 131, 108, 164, 209,  61,  90,  98, 122, 156,
        92,  79, 110, 111, 119, 258, 326,  91,  70,  80, 140, 134, 183,
       234, 308, 304,  97, 103, 120, 181, 151, 194, 132, 121, 146, 171,
       161, 141, 173, 145], dtype=int64)
```

In [36]:

```
#int64 indicates absence of null values, then too, i will again check for time pass
df['engine-size'].isnull().any()
```

Out[36]:

```
False
```

In [37]:

```
#converting datatype into float
df['engine-size'] = df['engine-size'].astype('float64')
```

bore and stroke

not categorical data

shd be converted to float for decimal value.if,

if,null values ,then,filled by mean

In [38]:

```
#check for null values
df['bore'].isnull().any()
```

Out[38]:

```
True
```

In [39]:

```
#check unique values
df['bore'].unique()
```

Out[39]:

```
array(['3.47', '2.68', '3.19', '3.13', '3.5', '3.31', '3.62', '2.91',
       '3.03', '2.97', '3.34', '3.6', '2.92', '3.15', '3.43', '3.63',
       '3.54', '3.08', nan, '3.39', '3.76', '3.58', '3.46', '3.8', '3.78',
       '3.17', '3.35', '3.59', '2.99', '3.33', '3.7', '3.61', '3.94',
       '3.74', '2.54', '3.05', '3.27', '3.24', '3.01'], dtype=object)
```

In [40]:

```
#convert to float
df['bore'] = df['bore'].astype('float64')
```

In [41]:

```
#check for null values
df['stroke'].isnull().any()
```

Out[41]:

```
True
```

In [42]:

```
#check unique values
df['stroke'].unique()
```

Out[42]:

```
array(['2.68', '3.47', '3.4', '2.8', '3.19', '3.39', '3.03', '3.11',
       '3.23', '3.46', '3.9', '3.41', '3.07', '3.58', '4.17', '2.76',
       '3.15', nan, '3.16', '3.64', '3.1', '3.35', '3.12', '3.86', '3.29',
       '3.27', '3.52', '2.19', '3.21', '2.9', '2.07', '2.36', '2.64',
       '3.08', '3.5', '3.54', '2.87'], dtype=object)
```

In [43]:

```
#convert to float
df['stroke'] = df['stroke'].astype('float64')
```

In [44]:

```
#replace null values with mean
mean_b = df['bore'].mean()
mean_s = df['stroke'].mean()

df['bore'] = df['bore'].fillna(mean_b)
df['stroke'] = df['stroke'].fillna(mean_s)
```

compression ratio

check for- not categorical data

float64 present,good

no null values present

In [45]:

```
df['compression-ratio'].unique()
```

Out[45]:

```
array([ 9.  , 10.  ,  8.  ,  8.5 ,  8.3 ,  8.8 ,  9.5 ,  9.6 ,  9.41,
        9.4 ,  7.6 ,  7.  ,  9.2 , 10.1 ,  9.1 ,  8.1 , 11.5 ,  8.6 ,
       22.7 , 22.  , 21.5 ,  7.5 , 21.9 ,  7.8 ,  8.4 , 21.  ,  8.7 ,
        9.31,  9.3 ,  7.7 , 22.5 , 23.  ])
```

In [46]:

```
df['compression-ratio'].isnull().any()
```

Out[46]:

```
False
```

horse power and peak-rpm

- not categorical data

convert to float64

null values to be replaced with mean col

In [47]:

```
#null val for horse power
df['horsepower'].isnull().any()
```

Out[47]:

```
True
```

In [48]:

```
#null val for peakrmp
df['peak-rpm'].isnull().any()
```

Out[48]:

```
True
```

In [49]:

```
#convert to float for hp
df['horsepower'] = df['horsepower'].astype('float64')
```

In [50]:

```
#convert to float
df['peak-rpm'] = df['peak-rpm'].astype('float64')
```

In [51]:

```
#fill null for the hp with mean
mean_h = df['horsepower'].mean()
df['horsepower'] = df['horsepower'].fillna(mean_h)
```

In [52]:

```
#fill null for the peak with mean
mean_p = df['peak-rpm'].mean()
df['peak-rpm'] = df['peak-rpm'].fillna(mean_p)
```

city mpg and highway mpg

- not categorical
- convert from int to float for decimal value prediction also
- no null values

In [53]:

```
df['city-mpg'].isnull().any()
```

Out[53]:

```
False
```

In [54]:

```
df['highway-mpg'].isnull().any()
```

Out[54]:

```
False
```

In [55]:

```
df['city-mpg'] = df['city-mpg'].astype('float64')
```

In [56]:

```
df['highway-mpg'] = df['highway-mpg'].astype('float64')
```

In [57]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 201 entries, 0 to 204
Data columns (total 26 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   symboling          201 non-null    int64
 1   normalized-losses  201 non-null    float64
 2   make               201 non-null    object
 3   fuel-type          201 non-null    object
 4   aspiration         201 non-null    object
 5   num-of-doors       201 non-null    int64
 6   body-style         201 non-null    object
 7   drive-wheels       201 non-null    object
 8   engine-location    201 non-null    object
 9   wheel-base         201 non-null    float64
 10  length             201 non-null    float64
 11  width              201 non-null    float64
 12  height             201 non-null    float64
 13  curb-weight        201 non-null    float64
 14  engine-type        201 non-null    object
 15  num-of-cylinders   201 non-null    int64
 16  engine-size        201 non-null    float64
 17  fuel-system        201 non-null    object
 18  bore               201 non-null    float64
 19  stroke             201 non-null    float64
 20  compression-ratio  201 non-null    float64
 21  horsepower         201 non-null    float64
 22  peak-rpm           201 non-null    float64
 23  city-mpg           201 non-null    float64
 24  highway-mpg        201 non-null    float64
 25  price              201 non-null    float64
dtypes: float64(15), int64(3), object(8)
memory usage: 42.4+ KB
```

In [58]:

```
df
```

Out[58]:

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location
0	3	122.0	alfa-romero	gas	std	2	convertible	rwd	front
1	3	122.0	alfa-romero	gas	std	2	convertible	rwd	front
2	1	122.0	alfa-romero	gas	std	2	hatchback	rwd	front
3	2	164.0	audi	gas	std	4	sedan	fwd	front
4	2	164.0	audi	gas	std	4	sedan	4wd	front
...	...	...	...	...	...	...	...	...	...
200	-1	95.0	volvo	gas	std	4	sedan	rwd	front
201	-1	95.0	volvo	gas	turbo	4	sedan	rwd	front
202	-1	95.0	volvo	gas	std	4	sedan	rwd	front
203	-1	95.0	volvo	diesel	turbo	4	sedan	rwd	front
204	-1	95.0	volvo	gas	turbo	4	sedan	rwd	front

201 rows × 26 columns

In [ ]:

```
df.to_csv('x_automobile.csv',index=False)
```