In [1]:
```
import pandas as pd
```

In [2]:
```
df=pd.read_csv("auto-mpg.csv")
df=pd.DataFrame(df)
```

In [4]:
```
df
```

Out[4]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
393	27.0	4	140.0	86	2790	15.6	82	1	ford mustang gl
394	44.0	4	97.0	52	2130	24.6	82	2	vw pickup
395	32.0	4	135.0	84	2295	11.6	82	1	dodge rampage
396	28.0	4	120.0	79	2625	18.6	82	1	ford ranger
397	31.0	4	119.0	82	2720	19.4	82	1	chevy s-10

398 rows × 9 columns

In [3]:
```
df.drop(['car name'],inplace=True,axis=1)
```

In [12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 392 entries, 0 to 397
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   mpg           392 non-null    float64
 1   cylinders     392 non-null    int64  
 2   displacement  392 non-null    float64
 3   horsepower    392 non-null    object 
 4   weight        392 non-null    int64  
 5   acceleration  392 non-null    float64
 6   model year    392 non-null    int64  
 7   origin        392 non-null    int64  
dtypes: float64(3), int64(4), object(1)
memory usage: 27.6+ KB
```

In [17]:
```
y=df['mpg']
x=df.drop("mpg",axis=1)
# df[df["horsepower"]=="?"]
df=df[df['horsepower']!="?"]
```

In [18]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
lm=LinearRegression()
```

In [19]:
```
lm.fit(x_train,y_train)
```

Out[19]:
```
LinearRegression()
```

In [22]:
```
y_pred=lm.predict(x_test)
# diff=pd.DataFrame({'actual':y_test,"predict":y_pred})
# print(diff)
print("mse",metrics.mean_squared_error(y_test,y_pred))
```

```
     actual    predict
116    16.0  12.240739
100    18.0  21.234543
327    36.4  26.696668
101    23.0  20.575697
198    33.0  32.684178
..      ...        ...
113    21.0  22.020824
341    23.5  27.064982
294    34.1  33.541372
66     17.0  14.781277
392    27.0  27.820956

[79 rows x 2 columns]
mse 9.412211778154111
```

In [25]:
```
df=pd.read_csv("winequalityN.csv")
df=pd.DataFrame(df)
```

In [27]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6497 entries, 0 to 6496
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   type                  6497 non-null   object 
 1   fixed acidity         6487 non-null   float64
 2   volatile acidity      6489 non-null   float64
 3   citric acid           6494 non-null   float64
 4   residual sugar        6495 non-null   float64
 5   chlorides             6495 non-null   float64
 6   free sulfur dioxide   6497 non-null   float64
 7   total sulfur dioxide  6497 non-null   float64
 8   density               6497 non-null   float64
 9   pH                    6488 non-null   float64
 10  sulphates             6493 non-null   float64
 11  alcohol               6497 non-null   float64
 12  quality               6497 non-null   int64  
dtypes: float64(11), int64(1), object(1)
memory usage: 660.0+ KB
```

In [28]:
```
df.isna().sum()
```

Out[28]:
```
type                   0
fixed acidity         10
volatile acidity       8
citric acid            3
residual sugar         2
chlorides              2
free sulfur dioxide    0
total sulfur dioxide   0
density                0
pH                     9
sulphates              4
alcohol                0
quality                0
dtype: int64
```

In [23]:
```
# df.describe()
```

In [30]:
```
df['fixed acidity'].fillna(df['fixed acidity'].mean(),inplace=True)
df['volatile acidity'].fillna(df['volatile acidity'].mean(),inplace=True)
df['citric acid'].fillna(df['citric acid'].mean(),inplace=True)
df['residual sugar'].fillna(df['residual sugar'].mean(),inplace=True)
df['chlorides'].fillna(df['chlorides'].mean(),inplace=True)
df['pH'].fillna(df['pH'].mean(),inplace=True)
df['sulphates'].fillna(df['sulphates'].mean(),inplace=True)

df.isna().sum()
```

Out[30]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
type_white              0
dtype: int64
```

In [38]:
```
df
```

Out[38]:

	type	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	white	7.0	0.270	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	0.450000	8.8	6
1	white	6.3	0.300	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	0.490000	9.5	6
2	white	8.1	0.280	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	0.440000	10.1	6
3	white	7.2	0.230	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.400000	9.9	6
4	white	7.2	0.230	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.400000	9.9	6
...	...	...	...	...	...	...	...	...	...	...	...	...	...
6492	red	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.580000	10.5	5
6493	red	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.531215	11.2	6
6494	red	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.750000	11.0	6
6495	red	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.710000	10.2	5
6496	red	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.660000	11.0	6

6497 rows × 13 columns

In [31]:
```
df=pd.get_dummies(data=df,drop_first=True)
df
```

Out[31]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	type_white
0	7.0	0.270	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	0.450000	8.8	6	1
1	6.3	0.300	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	0.490000	9.5	6	1
2	8.1	0.280	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	0.440000	10.1	6	1
3	7.2	0.230	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.400000	9.9	6	1
4	7.2	0.230	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.400000	9.9	6	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
6492	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.580000	10.5	5	0
6493	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.531215	11.2	6	0
6494	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.750000	11.0	6	0
6495	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.710000	10.2	5	0
6496	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.660000	11.0	6	0

6497 rows × 13 columns

In [32]:
```
y=df['quality']
x=df.drop("quality",axis=1)
```

In [34]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
lm=LinearRegression()
```

In [35]:
```
lm.fit(x_train,y_train)
```

Out[35]:
```
LinearRegression()
```

In [38]:
```
y_pred=lm.predict(x_test)
print("mSe",metrics.mean_squared_error(y_test,y_pred))
```

```
mSe 0.5775498959781392
```

In [39]:
```
print("mAe",metrics.mean_absolute_error(y_test,y_pred))
```

```
mAe 0.584147072519033
```

In [41]:
```
df=pd.read_csv("car data.csv")
df=pd.DataFrame(df)
```

In [42]:
```
df
```

Out[42]:

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0
...	...	...	...	...	...	...	...	...	...
296	city	2016	9.50	11.60	33988	Diesel	Dealer	Manual	0
297	brio	2015	4.00	5.90	60000	Petrol	Dealer	Manual	0
298	city	2009	3.35	11.00	87934	Petrol	Dealer	Manual	0
299	city	2017	11.50	12.50	9000	Diesel	Dealer	Manual	0
300	brio	2016	5.30	5.90	5464	Petrol	Dealer	Manual	0

301 rows × 9 columns

In [43]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Car_Name       301 non-null    object 
 1   Year           301 non-null    int64  
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Kms_Driven     301 non-null    int64  
 5   Fuel_Type      301 non-null    object 
 6   Seller_Type    301 non-null    object 
 7   Transmission   301 non-null    object 
 8   Owner          301 non-null    int64  
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

In [44]:
```
df.isna().sum()
```

Out[44]:
```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Kms_Driven       0
Fuel_Type        0
Seller_Type      0
Transmission     0
Owner            0
dtype: int64
```

In [45]:
```
df
```

Out[45]:

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0
...	...	...	...	...	...	...	...	...	...
296	city	2016	9.50	11.60	33988	Diesel	Dealer	Manual	0
297	brio	2015	4.00	5.90	60000	Petrol	Dealer	Manual	0
298	city	2009	3.35	11.00	87934	Petrol	Dealer	Manual	0
299	city	2017	11.50	12.50	9000	Diesel	Dealer	Manual	0
300	brio	2016	5.30	5.90	5464	Petrol	Dealer	Manual	0

301 rows × 9 columns

In [47]:
```
df.drop(["Car_Name"],axis=1,inplace=True)
```

In [48]:
```
df
```

Out[48]:

	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0
...	...	...	...	...	...	...	...	...
296	2016	9.50	11.60	33988	Diesel	Dealer	Manual	0
297	2015	4.00	5.90	60000	Petrol	Dealer	Manual	0
298	2009	3.35	11.00	87934	Petrol	Dealer	Manual	0
299	2017	11.50	12.50	9000	Diesel	Dealer	Manual	0
300	2016	5.30	5.90	5464	Petrol	Dealer	Manual	0

301 rows × 8 columns

In [51]:
```
df=pd.get_dummies(data=df,drop_first=True)
df
```

Out[51]:

	Year	Selling_Price	Present_Price	Kms_Driven	Owner	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_I
0	2014	3.35	5.59	27000	0	0	1	0	
1	2013	4.75	9.54	43000	0	1	0	0	
2	2017	7.25	9.85	6900	0	0	1	0	
3	2011	2.85	4.15	5200	0	0	1	0	
4	2014	4.60	6.87	42450	0	1	0	0	
...	...	...	...	...	...	...	...	...	
296	2016	9.50	11.60	33988	0	1	0	0	
297	2015	4.00	5.90	60000	0	0	1	0	
298	2009	3.35	11.00	87934	0	0	1	0	
299	2017	11.50	12.50	9000	0	1	0	0	
300	2016	5.30	5.90	5464	0	0	1	0	

301 rows × 9 columns

In [53]:
```
y=df['Selling_Price']
x=df.drop("Selling_Price",axis=1)
```

In [54]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
lm=LinearRegression()
```

In [55]:
```
lm.fit(x_train,y_train)
```

Out[55]:
```
LinearRegression()
```

In [56]:
```
y_pred=lm.predict(x_test)
```

In [57]:
```
print("mSe",metrics.mean_squared_error(y_test,y_pred))
```

```
mSe 12.052297978660043
```

In [ ]:
```

```