In [1]:
```python
import numpy as np
import pandas as pd

#using graphical representation visualization

import matplotlib.pyplot as plt
```

In [2]:
```python
df = pd.read_csv('quikr_car.csv')
```

In [3]:
```python
df.shape
```

Out[3]:
```
(892, 6)
```

In [4]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 892 entries, 0 to 891
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   name        892 non-null    object
 1   company     892 non-null    object
 2   year        892 non-null    object
 3   Price       892 non-null    object
 4   kms_driven  840 non-null    object
 5   fuel_type   837 non-null    object
dtypes: object(6)
memory usage: 41.9+ KB
```

Data cleaning started

In [5]:
```python
df.head(3)
```

Out[5]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
2	Maruti Suzuki Alto 800 Vxi	Maruti	2018	Ask For Price	22,000 kms	Petrol

In [6]:
```python
# isnumeric function to check numeric values

df = df[df['year'].str.isnumeric()]
```

In [7]:
```python
# converting into integer (year)

df['year'] = df['year'].astype(int)
```

In [8]:
```python
# removing Ask for price

df = df[df['Price']!='Ask For Price']
```

In [9]:
```python
df['Price'].unique()
```

Out[9]:
```
array(['80,000', '4,25,000', '3,25,000', '5,75,000', '1,75,000',
       '1,90,000', '8,30,000', '2,50,000', '1,82,000', '3,15,000',
       '4,15,000', '3,20,000', '10,00,000', '5,00,000', '3,50,000',
       '1,60,000', '3,10,000', '75,000', '1,00,000', '2,90,000', '95,000',
       '1,80,000', '3,85,000', '1,05,000', '6,50,000', '6,89,999',
       '4,48,000', '5,49,000', '5,01,000', '4,89,999', '2,80,000',
       '3,49,999', '2,84,999', '3,45,000', '4,99,999', '2,35,000',
       '2,49,999', '14,75,000', '3,95,000', '2,20,000', '1,70,000',
       '85,000', '2,00,000', '5,70,000', '1,10,000', '4,48,999',
       '18,91,111', '1,59,500', '3,44,999', '4,49,999', '8,65,000',
       '6,99,000', '3,75,000', '2,24,999', '12,00,000', '1,95,000',
       '3,51,000', '2,40,000', '90,000', '1,55,000', '6,00,000',
       '1,89,500', '2,10,000', '3,90,000', '1,35,000', '16,00,000',
       '7,01,000', '2,65,000', '5,25,000', '3,72,000', '6,35,000',
       '5,50,000', '4,85,000', '3,29,500', '2,51,111', '5,69,999',
       '69,999', '2,99,999', '3,99,999', '4,50,000', '2,70,000',
       '1,58,400', '1,79,000', '1,25,000', '2,99,000', '1,50,000',
       '2,75,000', '2,85,000', '3,40,000', '70,000', '2,89,999',
       '8,49,999', '7,49,999', '2,74,999', '9,84,999', '5,99,999',
       '2,44,999', '4,74,999', '2,45,000', '1,69,500', '3,70,000',
       '1,68,000', '1,45,000', '98,500', '2,09,000', '1,85,000',
       '9,00,000', '6,99,999', '1,99,999', '5,44,999', '1,99,000',
       '5,40,000', '49,000', '7,00,000', '55,000', '8,95,000', '3,55,000',
       '5,65,000', '3,65,000', '40,000', '4,00,000', '3,30,000',
       '5,80,000', '3,79,000', '2,19,000', '5,19,000', '7,30,000',
       '20,00,000', '21,00,000', '14,00,000', '3,11,000', '8,55,000',
       '5,35,000', '1,78,000', '3,00,000', '2,55,000', '5,49,999',
       '3,80,000', '57,000', '4,10,000', '2,25,000', '1,20,000', '59,000',
       '5,99,000', '6,75,000', '72,500', '6,10,000', '2,30,000',
       '5,20,000', '5,24,999', '4,24,999', '6,44,999', '5,84,999',
       '7,99,999', '4,44,999', '6,49,999', '9,44,999', '5,74,999',
       '3,74,999', '1,30,000', '4,01,000', '13,50,000', '1,74,999',
       '2,39,999', '99,999', '3,24,999', '10,74,999', '11,30,000',
       '1,49,000', '7,70,000', '30,000', '3,35,000', '3,99,000', '65,000',
       '1,69,999', '1,65,000', '5,60,000', '9,50,000', '7,15,000',
       '45,000', '9,40,000', '1,55,555', '15,00,000', '4,95,000',
       '8,00,000', '12,99,000', '5,30,000', '14,99,000', '32,000',
       '4,05,000', '7,60,000', '7,50,000', '4,19,000', '1,40,000',
       '15,40,000', '1,23,000', '4,98,000', '4,80,000', '4,88,000',
       '15,25,000', '5,48,900', '7,25,000', '99,000', '52,000',
       '28,00,000', '4,99,000', '3,81,000', '2,78,000', '6,90,000',
       '2,60,000', '90,001', '1,15,000', '15,99,000', '1,59,000',
       '51,999', '2,15,000', '35,000', '11,50,000', '2,69,000', '60,000',
       '4,30,000', '85,00,003', '4,01,919', '4,90,000', '4,24,000',
       '2,05,000', '5,49,900', '4,35,000', '1,89,700', '3,89,700',
       '3,60,000', '2,95,000', '1,14,990', '10,65,000', '4,70,000',
       '48,000', '1,88,000', '4,65,000', '1,79,999', '21,90,000',
       '23,90,000', '10,75,000', '4,75,000', '10,25,000', '6,15,000',
       '19,00,000', '14,90,000', '15,10,000', '18,50,000', '7,90,000',
       '17,25,000', '12,25,000', '68,000', '9,70,000', '31,00,000',
       '8,99,000', '88,000', '53,000', '5,68,500', '71,000', '5,90,000',
       '7,95,000', '42,000', '1,89,000', '1,62,000', '35,999',
       '29,00,000', '39,999', '50,500', '5,10,000', '8,60,000',
       '5,00,001'], dtype=object)
```

In [10]:
```python
# removing commas and add astype function

df['Price'] = df["Price"].str.replace(',','').astype(int)
```

kms_driven column cleaning

In [11]:
```python
df.head(3)
```

Out[11]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40 kms	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28,000 kms	Petrol

In [12]:
```python
# removing kms in kms_driven column

df['kms_driven'] = df['kms_driven'].str.split(' ').str.get(0)
```

In [13]:
```python
df.head(5)
```

Out[13]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45,000	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28,000	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	575000	36,000	Diesel
6	Ford Figo	Ford	2012	175000	41,000	Diesel

In [14]:
```python
# removing comma from kms_driven values

df['kms_driven'] =df['kms_driven'].str.replace(',','')
```

In [15]:
```python
df.head(5)
```

Out[15]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28000	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	575000	36000	Diesel
6	Ford Figo	Ford	2012	175000	41000	Diesel

In [16]:
```python
# let see fuel_type unique values

df['fuel_type'].unique()
```

Out[16]:
```
array(['Petrol', 'Diesel', nan, 'LPG'], dtype=object)
```

In [17]:
```python
#  checking and removing null values.

df = df[~df['fuel_type'].isnull()]
```

In [18]:
```python
# checking unique values

df['fuel_type'].unique()
```

Out[18]:
```
array(['Petrol', 'Diesel', 'LPG'], dtype=object)
```

In [19]:
```python
# After removing null values shape will

df.shape
```

Out[19]:
```
(816, 6)
```

In [20]:
```python
df.head(3)
```

Out[20]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	425000	40	Diesel
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	325000	28000	Petrol

In [21]:
```python
# checking name

df['name']
```

Out[21]:
```
0        Hyundai Santro Xing XO eRLX Euro III
1                     Mahindra Jeep CL550 MDI
3      Hyundai Grand i10 Magna 1.2 Kappa VTVT
4            Ford EcoSport Titanium 1.5L TDCi
6                                   Ford Figo
                        ...
883                Maruti Suzuki Ritz VXI ABS
885                 Tata Indica V2 DLE BS III
886                      Toyota Corolla Altis
888                       Tata Zest XM Diesel
889                        Mahindra Quanto C8
Name: name, Length: 816, dtype: object
```

In [22]:
```python
# checking  null values in company

df['company'].isnull().sum()
```

Out[22]:
```
0
```

In [23]:
```python
#checking unique values

df['company'].unique()
```

Out[23]:
```
array(['Hyundai', 'Mahindra', 'Ford', 'Maruti', 'Skoda', 'Audi', 'Toyota',
       'Renault', 'Honda', 'Datsun', 'Mitsubishi', 'Tata', 'Volkswagen',
       'Chevrolet', 'Mini', 'BMW', 'Nissan', 'Hindustan', 'Fiat', 'Force',
       'Mercedes', 'Land', 'Jaguar', 'Jeep', 'Volvo'], dtype=object)
```

In [24]:
```python
#excepting only three words and removing other words

df['name'] = df['name'].str.split().str.slice(start = 0, stop = 3).str.join(' ')
```

In [25]:
```python
# comming three words

df['name']
```

Out[25]:
```
0          Hyundai Santro Xing
1          Mahindra Jeep CL550
3            Hyundai Grand i10
4       Ford EcoSport Titanium
6                    Ford Figo
                ...
883         Maruti Suzuki Ritz
885             Tata Indica V2
886       Toyota Corolla Altis
888               Tata Zest XM
889         Mahindra Quanto C8
Name: name, Length: 816, dtype: object
```

In [26]:
```python
df
```

Out[26]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
3	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
4	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
6	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
883	Maruti Suzuki Ritz	Maruti	2011	270000	50000	Petrol
885	Tata Indica V2	Tata	2009	110000	30000	Diesel
886	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
888	Tata Zest XM	Tata	2018	260000	27000	Diesel
889	Mahindra Quanto C8	Mahindra	2013	390000	40000	Diesel

816 rows × 6 columns

In [27]:
```python
# now change index

df = df.reset_index(drop=True)
```

In [28]:
```python
df
```

Out[28]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
811	Maruti Suzuki Ritz	Maruti	2011	270000	50000	Petrol
812	Tata Indica V2	Tata	2009	110000	30000	Diesel
813	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
814	Tata Zest XM	Tata	2018	260000	27000	Diesel
815	Mahindra Quanto C8	Mahindra	2013	390000	40000	Diesel

816 rows × 6 columns

In [29]:
```python
#cleaned data csv file.

df.to_csv('Cleaned_car_data.csv')
```

In [30]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 816 entries, 0 to 815
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   name        816 non-null    object
 1   company     816 non-null    object
 2   year        816 non-null    int32
 3   Price       816 non-null    int32
 4   kms_driven  816 non-null    object
 5   fuel_type   816 non-null    object
dtypes: int32(2), object(4)
memory usage: 32.0+ KB
```

In [31]:
```python
# convert object into integer.

df['kms_driven'] = df['kms_driven'].astype(int)
```

In [32]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 816 entries, 0 to 815
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   name        816 non-null    object
 1   company     816 non-null    object
 2   year        816 non-null    int32
 3   Price       816 non-null    int32
 4   kms_driven  816 non-null    int32
 5   fuel_type   816 non-null    object
dtypes: int32(3), object(3)
memory usage: 28.8+ KB
```

In [33]:
```python
df.head(5)
```

Out[33]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel

In [34]:
```python
df.describe()
```

Out[34]:

	year	Price	kms_driven
count	816.000000	8.160000e+02	816.000000
mean	2012.444853	4.117176e+05	46275.531863
std	4.002992	4.751844e+05	34297.428044
min	1995.000000	3.000000e+04	0.000000
25%	2010.000000	1.750000e+05	27000.000000
50%	2013.000000	2.999990e+05	41000.000000
75%	2015.000000	4.912500e+05	56818.500000
max	2019.000000	8.500003e+06	400000.000000

In [35]:
```python
# maximum value

df['Price'].max()
```

Out[35]:
```
8500003
```

In [36]:
```python
df
```

Out[36]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
811	Maruti Suzuki Ritz	Maruti	2011	270000	50000	Petrol
812	Tata Indica V2	Tata	2009	110000	30000	Diesel
813	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
814	Tata Zest XM	Tata	2018	260000	27000	Diesel
815	Mahindra Quanto C8	Mahindra	2013	390000	40000	Diesel

816 rows × 6 columns

In [37]:
```python
# Extracting training data

X = df[['name','company','year','kms_driven','fuel_type']]
y = df['Price']
```

In [38]:
```python
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2)
```

In [39]:
```python
from sklearn.linear_model import LinearRegression
```

In [40]:
```python
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
from sklearn.pipeline import make_pipeline
from sklearn.metrics import r2_score
```

In [41]:
```python
df
```

Out[41]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
811	Maruti Suzuki Ritz	Maruti	2011	270000	50000	Petrol
812	Tata Indica V2	Tata	2009	110000	30000	Diesel
813	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
814	Tata Zest XM	Tata	2018	260000	27000	Diesel
815	Mahindra Quanto C8	Mahindra	2013	390000	40000	Diesel

816 rows × 6 columns

In [42]:
```python
# use and create object(ohe)

ohe = OneHotEncoder()
```

In [43]:
```python
#fit values in ohe

ohe.fit(X[['name','company','fuel_type']])
```

Out[43]:
```
OneHotEncoder()
```

In [44]:
```python
# categorical column transform

column_trans = make_column_transformer((OneHotEncoder(categories=ohe.categories_), ['name','company','fuel_type']),remainder='passthrough')
```

In [45]:
```python
column_trans
```

Out[45]:
```
ColumnTransformer(remainder='passthrough',
                  transformers=[('onehotencoder',
                                 OneHotEncoder(categories=[array(['Audi A3 Cabriolet', 'Audi A4 1.8', 'Audi A4 2.0', 'Audi A6 2.0',
       'Audi A8', 'Audi Q3 2.0', 'Audi Q5 2.0', 'Audi Q7', 'BMW 3 Series',
       'BMW 5 Series', 'BMW 7 Series', 'BMW X1', 'BMW X1 sDrive20d',
       'BMW X1 xDrive20d', 'Chevrolet Beat', 'Chevrolet Beat Diesel',
       'Chevrolet Beat LS', 'Chevrolet B...
       'Volkswagen Vento Konekt', 'Volvo S80 Summum'], dtype=object),
                                                           array(['Audi', 'BMW', 'Chevrolet', 'Datsun', 'Fiat', 'Force', 'Ford',
       'Hindustan', 'Honda', 'Hyundai', 'Jaguar', 'Jeep', 'Land',
       'Mahindra', 'Maruti', 'Mercedes', 'Mini', 'Mitsubishi', 'Nissan',
       'Renault', 'Skoda', 'Tata', 'Toyota', 'Volkswagen', 'Volvo'],
      dtype=object),
                                                           array(['Diesel', 'LPG', 'Petrol'], dtype=object)]),
                                 ['name', 'company', 'fuel_type'])])
```

In [46]:
```python
# using linaearregression algorithm

model = LinearRegression()
```

In [47]:
```python
# construct a pipeline

pipe = make_pipeline(column_trans,model)
```

In [48]:
```python
# using simple fit

pipe.fit(X_train,y_train)
```

Out[48]:
```
Pipeline(steps=[('columntransformer',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('onehotencoder',
                                                  OneHotEncoder(categories=[array(['Audi A3 Cabriolet', 'Audi A4 1.8', 'Audi A4 2.0', 'Audi A6 2.0',
       'Audi A8', 'Audi Q3 2.0', 'Audi Q5 2.0', 'Audi Q7', 'BMW 3 Series',
       'BMW 5 Series', 'BMW 7 Series', 'BMW X1', 'BMW X1 sDrive20d',
       'BMW X1 xDrive20d', 'Chevrolet Beat', 'Chevrolet Beat...
                                                                            array(['Audi', 'BMW', 'Chevrolet', 'Datsun', 'Fiat', 'Force', 'Ford',
       'Hindustan', 'Honda', 'Hyundai', 'Jaguar', 'Jeep', 'Land',
       'Mahindra', 'Maruti', 'Mercedes', 'Mini', 'Mitsubishi', 'Nissan',
       'Renault', 'Skoda', 'Tata', 'Toyota', 'Volkswagen', 'Volvo'],
      dtype=object),
                                                                            array(['Diesel', 'LPG', 'Petrol'], dtype=object)]),
                                                  ['name', 'company',
                                                   'fuel_type'])])),
                ('linearregression', LinearRegression())])
```

In [49]:
```python
y_pred = pipe.predict(X_test)
```

In [50]:
```python
y_pred
```

Out[50]:
```
array([ 5.28099773e+05,  2.32220189e+05,  2.34173713e+05,  2.32972000e+05,
        3.63964050e+04,  5.84084761e+05,  3.07402753e+05,  3.64084210e+05,
        1.26407959e+05,  2.98251178e+05,  2.92586854e+06,  9.37959158e+05,
        1.98928544e+05,  1.12164056e+05,  1.23821880e+06,  3.64365018e+05,
        4.13617526e+05,  2.48730945e+05,  2.00852752e+05,  4.05515728e+04,
        2.98655633e+05,  2.82142512e+05,  5.21018098e+05,  2.83034881e+05,
        3.83394688e+05,  1.20114498e+05,  2.11604324e+05,  2.86628318e+05,
        2.36265902e+05,  5.85239187e+05,  4.04676901e+05,  2.31384103e+05,
        1.47681092e+05, -2.31043242e+05,  1.08901023e+05, -9.68368398e+04,
        3.80552963e+05,  5.28099773e+05,  5.16358195e+05,  3.13294159e+05,
        2.83630424e+05,  1.24913905e+05,  7.86048694e+05, -6.86580652e+04,
        3.39240972e+05,  1.35149125e+06,  5.21018098e+05,  1.21070829e+06,
        7.16352037e+05,  1.71264832e+05,  1.76959935e+05,  8.19209827e+05,
        1.83272050e+05,  3.53741704e+05,  5.46661882e+05,  3.13347327e+05,
        4.01059648e+05,  5.16746720e+05,  1.42257438e+05,  5.23040352e+05,
        4.53101650e+05,  6.28579080e+05,  2.72766352e+04,  6.50516410e+04,
        4.99641277e+05, -1.57395065e+05,  5.26613520e+05,  5.06456325e+05,
        3.85894241e+05,  3.71072468e+05,  2.60863366e+05,  6.62450742e+05,
        5.86893002e+05,  4.32363016e+05,  3.17393036e+05,  7.90089169e+05,
        2.32033117e+06,  7.22286744e+05,  1.78244161e+05,  1.39545108e+05,
        1.49596233e+06,  5.00891769e+05,  3.72453906e+05,  3.54477288e+05,
        4.33986441e+05, -8.94491740e+04,  4.40061649e+05,  1.99664379e+05,
        4.76341990e+05,  4.01401676e+05,  9.51058675e+05,  2.36307119e+05,
        6.28671637e+05,  3.00972028e+05,  3.49333436e+05,  9.51058675e+05,
        1.49522675e+06,  4.67882781e+05,  6.16907861e+05,  7.53607476e+04,
        4.20697319e+05,  9.05932069e+04,  1.36575896e+05, -3.69268038e+04,
        1.26961008e+06,  5.22278649e+05,  2.48312575e+05,  3.85837167e+05,
        3.88751967e+05,  4.04535671e+04,  7.50775293e+05,  1.47308471e+05,
        4.32721238e+05,  9.37959158e+05,  4.46799675e+05,  1.11366865e+06,
        3.45664218e+05,  1.06510242e+05,  2.02250361e+05,  2.84868155e+05,
        4.27954810e+04,  2.32033117e+06,  3.86495225e+05,  4.52486094e+05,
        3.51375202e+05,  2.08344988e+05,  3.59249745e+05,  2.29424617e+05,
        1.98023486e+05,  2.67192670e+05,  6.68753261e+04,  3.16657453e+05,
        2.53216752e+05, -7.29555601e+04,  4.05840940e+05,  5.45424435e+05,
        3.05434870e+05,  4.31056382e+05,  3.93757797e+05,  2.46817149e+05,
        3.48013586e+05,  2.99321975e+05,  3.15774753e+05,  1.05986425e+05,
        2.32036795e+06,  4.08658000e+05,  2.40107999e+05,  4.19000969e+05,
        5.39647217e+05,  1.93969024e+05, -1.35113406e+05,  1.49596233e+06,
        2.76798244e+03,  2.45408369e+05, -1.97851686e+04,  1.46090530e+05,
        8.17643518e+05,  2.72610637e+05,  1.65398066e+05,  2.47333337e+05,
        6.17858344e+05,  4.90391579e+05,  3.53183121e+05,  8.78507624e+04])
```

In [51]:
```python
r2_score(y_test,y_pred)
```

Out[51]:
```
0.48253481800219367
```

Conclusion : So the model is ready