**import** **pandas** **as** **pd**

**import** **matplotlib.pyplot** **as** **plt**

**import** **seaborn** **as** **sns**

**from** **sklearn.model\_selection** **import** train\_test\_split

**from** **sklearn.linear\_model** **import** LinearRegression

**from** **sklearn.linear\_model** **import** Lasso

**from** **sklearn** **import** metrics

In [3]:

car\_data\_set = pd.read\_csv("car data.csv")

In [4]:

print(car\_data\_set)

Car\_Name Year Selling\_Price Present\_Price Kms\_Driven Fuel\_Type \

0 ritz 2014 3.35 5.59 27000 Petrol

1 sx4 2013 4.75 9.54 43000 Diesel

2 ciaz 2017 7.25 9.85 6900 Petrol

3 wagon r 2011 2.85 4.15 5200 Petrol

4 swift 2014 4.60 6.87 42450 Diesel

.. ... ... ... ... ... ...

296 city 2016 9.50 11.60 33988 Diesel

297 brio 2015 4.00 5.90 60000 Petrol

298 city 2009 3.35 11.00 87934 Petrol

299 city 2017 11.50 12.50 9000 Diesel

300 brio 2016 5.30 5.90 5464 Petrol

Seller\_Type Transmission Owner

0 Dealer Manual 0

1 Dealer Manual 0

2 Dealer Manual 0

3 Dealer Manual 0

4 Dealer Manual 0

.. ... ... ...

296 Dealer Manual 0

297 Dealer Manual 0

298 Dealer Manual 0

299 Dealer Manual 0

300 Dealer Manual 0

[301 rows x 9 columns]

In [5]:

car\_data\_set.head()

Out[5]:

|  | **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 |

In [6]:

car\_data\_set.tail()

Out[6]:

|  | **Car\_Name** | **Year** | **Selling\_Price** | **Present\_Price** | **Kms\_Driven** | **Fuel\_Type** | **Seller\_Type** | **Transmission** | **Owner** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **296** | city | 2016 | 9.50 | 11.6 | 33988 | Diesel | Dealer | Manual | 0 |
| **297** | brio | 2015 | 4.00 | 5.9 | 60000 | Petrol | Dealer | Manual | 0 |
| **298** | city | 2009 | 3.35 | 11.0 | 87934 | Petrol | Dealer | Manual | 0 |
| **299** | city | 2017 | 11.50 | 12.5 | 9000 | Diesel | Dealer | Manual | 0 |
| **300** | brio | 2016 | 5.30 | 5.9 | 5464 | Petrol | Dealer | Manual | 0 |

In [7]:

car\_data\_set.shape

Out[7]:

(301, 9)

In [8]:

car\_data\_set.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 [9]:

car\_data\_set.isnull().sum()

Out[9]:

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 [10]:

print(car\_data\_set.Fuel\_Type.value\_counts())

print(car\_data\_set.Seller\_Type.value\_counts())

print(car\_data\_set.Transmission.value\_counts())

Petrol 239

Diesel 60

CNG 2

Name: Fuel\_Type, dtype: int64

Dealer 195

Individual 106

Name: Seller\_Type, dtype: int64

Manual 261

Automatic 40

Name: Transmission, dtype: int64

In [11]:

car\_data\_set.replace({"Fuel\_Type":{"Petrol":0, "Diesel":1, "CNG":2}}, inplace=**True**)

car\_data\_set.replace({"Seller\_Type": {"Dealer":0, "Individual":1}}, inplace=**True**)

car\_data\_set.replace({"Transmission":{"Manual":0, "Automatic":1}}, inplace=**True**)

In [12]:

print(car\_data\_set)

Car\_Name Year Selling\_Price Present\_Price Kms\_Driven Fuel\_Type \

0 ritz 2014 3.35 5.59 27000 0

1 sx4 2013 4.75 9.54 43000 1

2 ciaz 2017 7.25 9.85 6900 0

3 wagon r 2011 2.85 4.15 5200 0

4 swift 2014 4.60 6.87 42450 1

.. ... ... ... ... ... ...

296 city 2016 9.50 11.60 33988 1

297 brio 2015 4.00 5.90 60000 0

298 city 2009 3.35 11.00 87934 0

299 city 2017 11.50 12.50 9000 1

300 brio 2016 5.30 5.90 5464 0

Seller\_Type Transmission Owner

0 0 0 0

1 0 0 0

2 0 0 0

3 0 0 0

4 0 0 0

.. ... ... ...

296 0 0 0

297 0 0 0

298 0 0 0

299 0 0 0

300 0 0 0

[301 rows x 9 columns]

In [13]:

x=car\_data\_set.drop(["Car\_Name","Selling\_Price"], axis= 1)

y=car\_data\_set["Selling\_Price"]

In [14]:

print(x)

print(y)

Year Present\_Price Kms\_Driven Fuel\_Type Seller\_Type Transmission \

0 2014 5.59 27000 0 0 0

1 2013 9.54 43000 1 0 0

2 2017 9.85 6900 0 0 0

3 2011 4.15 5200 0 0 0

4 2014 6.87 42450 1 0 0

.. ... ... ... ... ... ...

296 2016 11.60 33988 1 0 0

297 2015 5.90 60000 0 0 0

298 2009 11.00 87934 0 0 0

299 2017 12.50 9000 1 0 0

300 2016 5.90 5464 0 0 0

Owner

0 0

1 0

2 0

3 0

4 0

.. ...

296 0

297 0

298 0

299 0

300 0

[301 rows x 7 columns]

0 3.35

1 4.75

2 7.25

3 2.85

4 4.60

...

296 9.50

297 4.00

298 3.35

299 11.50

300 5.30

Name: Selling\_Price, Length: 301, dtype: float64

In [15]:

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y, test\_size=0.1, random\_state=2)

In [16]:

lin\_reg\_model = LinearRegression()

In [17]:

lin\_reg\_model.fit(x\_train, y\_train)

Out[17]:

LinearRegression()

In [18]:

train\_data\_prediction = lin\_reg\_model.predict(x\_train)

In [19]:

error\_score = metrics.r2\_score(y\_train, train\_data\_prediction)

In [20]:

print(error\_score)

0.8799451660493708

In [21]:

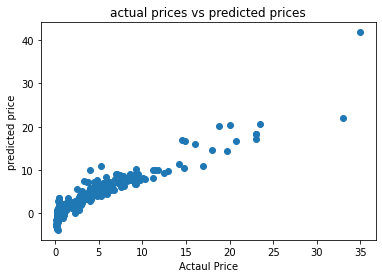
plt.scatter(y\_train, train\_data\_prediction)

plt.xlabel("Actaul Price")

plt.ylabel("predicted price")

plt.title("actual prices vs predicted prices")

plt.show()



In [24]:

test\_data\_prediction =lin\_reg\_model.predict(x\_test)

In [28]:

error\_score = metrics.r2\_score(y\_test, test\_data\_prediction)

In [26]:

print(error\_score)

0.8799451660493708

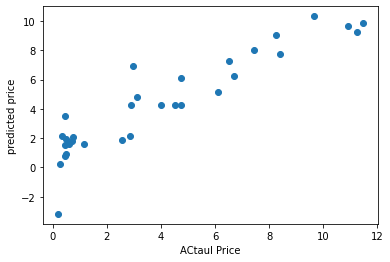
In [30]:

plt.scatter(y\_test, test\_data\_prediction)

plt.xlabel("ACtaul Price")

plt.ylabel("predicted price")

plt.show()



In [50]:

lin\_reg\_model = Lasso()

In [51]:

lin\_reg\_model.fit(x\_train,y\_train)

Out[51]:

Lasso()

In [52]:

train\_data\_prediction = lin\_reg\_model.predict(x\_train)

In [53]:

error\_score=metrics.r2\_score(y\_train, train\_data\_prediction)

In [54]:

print(error\_score)

0.8427856123435794

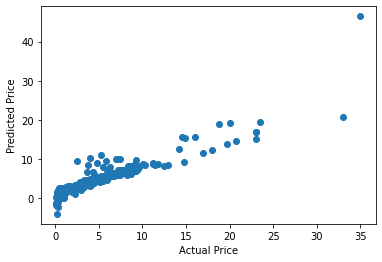
In [55]:

plt.scatter(y\_train,train\_data\_prediction)

plt.xlabel("Actual Price")

plt.ylabel("Predicted Price")

plt.show()



test\_data\_prediction = lin\_reg\_model.predict(x\_test)

In [32]:

error\_score = metrics.r2\_score(y\_test,test\_data\_prediction)

In [57]:

print(error\_score)

0.8427856123435794

In [35]:

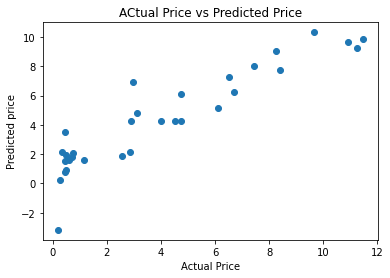
plt.scatter(y\_test,test\_data\_prediction)

plt.xlabel("Actual Price")

plt.ylabel("Predicted price")

plt.title("ACtual Price vs Predicted Price")

plt.show()



In [ ]: