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

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

In [3]:
```
df
```

Out[3]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	engir
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	
...	...	...	...	...	...	...	...	...	
200	201	-1	volvo 145e (sw)	gas	std	four	sedan	rwd	
201	202	-1	volvo 144ea	gas	turbo	four	sedan	rwd	
202	203	-1	volvo 244dl	gas	std	four	sedan	rwd	
203	204	-1	volvo 246	diesel	turbo	four	sedan	rwd	
204	205	-1	volvo 264gl	gas	turbo	four	sedan	rwd	

205 rows × 26 columns

In [4]:
```
## DATA PREPROCESSING ##
```

In [4]:
```
df.head()
```

Out[4]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginel
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	

5 rows × 26 columns

In [5]:

```
df.tail()
```

Out[5]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	engineloc
200	201	-1	volvo 145e (sw)	gas	std	four	sedan	rwd	
201	202	-1	volvo 144ea	gas	turbo	four	sedan	rwd	
202	203	-1	volvo 244dl	gas	std	four	sedan	rwd	
203	204	-1	volvo 246	diesel	turbo	four	sedan	rwd	
204	205	-1	volvo 264gl	gas	turbo	four	sedan	rwd	

5 rows × 26 columns

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   car_ID            205 non-null    int64  
 1   symboling         205 non-null    int64  
 2   CarName           205 non-null    object 
 3   fueltype          205 non-null    object 
 4   aspiration        205 non-null    object 
 5   doornumber        205 non-null    object 
 6   carbody           205 non-null    object 
 7   drivewheel        205 non-null    object 
 8   enginelocation    205 non-null    object 
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64  
 14  enginetype        205 non-null    object 
 15  cylindernumber    205 non-null    object 
 16  enginesize        205 non-null    int64  
 17  fuelsystem        205 non-null    object 
 18  boreratio         205 non-null    float64
 19  stroke            205 non-null    float64
 20  compressionratio  205 non-null    float64
 21  horsepower        205 non-null    int64  
 22  peakrpm           205 non-null    int64  
 23  citympg           205 non-null    int64  
 24  highwaympg        205 non-null    int64  
 25  price             205 non-null    float64
dtypes: float64(8), int64(8), object(10)
memory usage: 41.8+ KB
```

In [7]:

```
df.shape
```

Out[7]:

```
(205, 26)
```

In [8]:

```
df.columns
```

Out[8]:

```
Index(['car_ID', 'symboling', 'CarName', 'fueltype', 'aspiration',
       'doornumber', 'carbody', 'drivewheel', 'enginelocation', 'wheelbase',
       'carlength', 'carwidth', 'carheight', 'curbweight', 'enginetype',
       'cylindernumber', 'enginesize', 'fuelsystem', 'boreratio', 'stroke',
       'compressionratio', 'horsepower', 'peakrpm', 'citympg', 'highwaympg',
       'price'],
      dtype='object')
```

In [9]:

```
df.duplicated().sum()
```

Out[9]:

```
0
```

In [10]:

```
df.isnull().sum()
```

Out[10]:

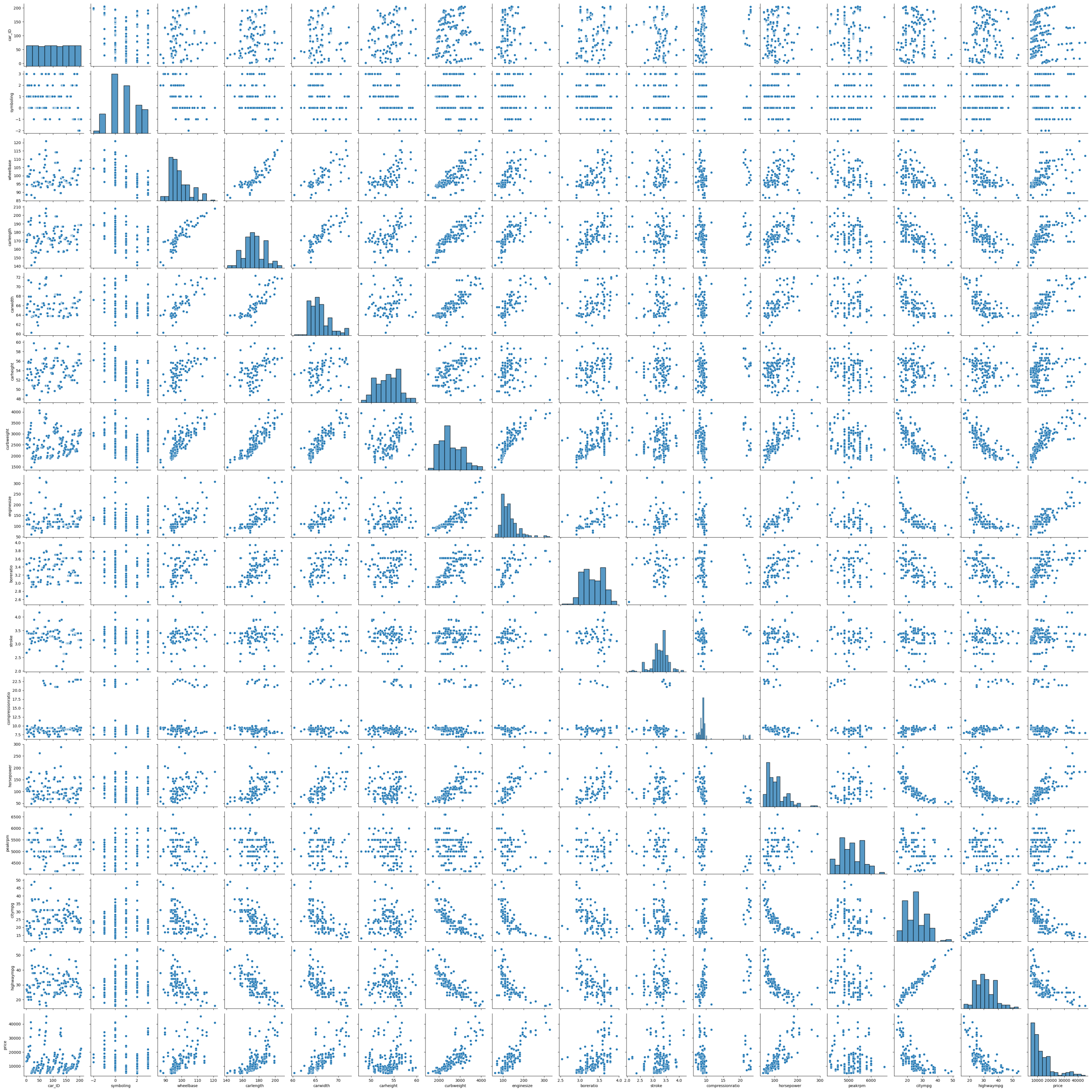
```
car_ID              0
symboling           0
CarName             0
fueltype            0
aspiration          0
doornumber          0
carbody             0
drivewheel          0
enginelocation      0
wheelbase           0
carlength           0
carwidth            0
carheight           0
curbweight          0
enginetype          0
cylindernumber      0
enginesize          0
fuelsystem          0
boreratio           0
stroke              0
compressionratio    0
horsepower          0
peakrpm             0
citympg             0
highwaympg          0
price               0
dtype: int64
```

In [11]:

```python
import matplotlib.pyplot as plt
import seaborn as sns

sns.pairplot(df)
```

Out[11]:

```
<seaborn.axisgrid.PairGrid at 0x1f5eb6eef70>
```



In [12]:

```
## CLASSIFICATION  OF CATEGORICAL DATA ##

print(df.fueltype.value_counts())
print(df.aspiration.value_counts())
print(df.doornumber.value_counts())
print(df.carbody.value_counts())
print(df.drivewheel.value_counts())
print(df.fuelsystem.value_counts())
print(df.cylindernumber.value_counts())
```

```
gas       185
diesel     20
Name: fueltype, dtype: int64
std      168
turbo     37
Name: aspiration, dtype: int64
four    115
two      90
Name: doornumber, dtype: int64
sedan          96
hatchback      70
wagon          25
hardtop         8
convertible     6
Name: carbody, dtype: int64
fwd    120
rwd     76
4wd      9
Name: drivewheel, dtype: int64
mpfi    94
2bbl    66
idi     20
1bbl    11
spdi     9
4bbl     3
mfi      1
spfi     1
Name: fuelsystem, dtype: int64
four      159
six        24
five       11
eight       5
two         4
three       1
twelve      1
Name: cylindernumber, dtype: int64
```

In [13]:

```
df['horsepower'].value_counts()
```

Out[13]:

```
68     19
70     11
69     10
116     9
110     8
95      7
114     6
160     6
101     6
62      6
88      6
145     5
76      5
97      5
84      5
90      5
82      5
102     5
92      4
111     4
123     4
86      4
207     3
73      3
182     3
121     3
85      3
152     3
176     2
94      2
56      2
112     2
161     2
184     2
155     2
156     2
52      2
100     2
162     2
140     1
115     1
134     1
78      1
142     1
288     1
143     1
48      1
200     1
58      1
55      1
60      1
175     1
154     1
72      1
120     1
64      1
135     1
262     1
106     1
Name: horsepower, dtype: int64
```

In [14]:

```
df['stroke'].value_counts()
```

Out[14]:

```
3.400    20
3.230    14
3.150    14
3.030    14
3.390    13
2.640    11
3.290     9
3.350     9
3.460     8
3.110     6
3.270     6
3.410     6
3.070     6
3.580     6
3.190     6
3.500     6
3.640     5
3.520     5
3.860     4
3.540     4
3.470     4
3.255     4
3.900     3
2.900     3
3.100     2
4.170     2
2.800     2
2.190     2
3.080     2
2.680     2
2.360     1
3.160     1
2.070     1
3.210     1
3.120     1
2.760     1
2.870     1
Name: stroke, dtype: int64
```

In [15]:

```
df['compressionratio'].value_counts()
```

Out[15]:

```
9.00     46
9.40     26
8.50     14
9.50     13
9.30     11
8.70      9
8.00      8
9.20      8
7.00      7
8.60      5
21.00     5
8.40      5
7.50      5
23.00     5
9.60      5
21.50     4
7.60      4
10.00     3
22.50     3
8.30      3
8.80      3
7.70      2
8.10      2
9.10      1
9.31      1
7.80      1
9.41      1
21.90     1
22.00     1
22.70     1
10.10     1
11.50     1
Name: compressionratio, dtype: int64
```

In [16]:

```
df['citympg'].value_counts()
```

Out[16]:

```
31    28
19    27
24    22
27    14
17    13
26    12
23    12
21     8
25     8
30     8
38     7
28     7
16     6
37     6
22     4
29     3
15     3
20     3
18     3
14     2
34     1
35     1
32     1
36     1
45     1
13     1
49     1
47     1
33     1
Name: citympg, dtype: int64
```

In [17]:

```
df['highwaympg'].value_counts()
```

Out[17]:

```
25    19
38    17
24    17
30    16
32    16
34    14
37    13
28    13
29    10
33     9
22     8
31     8
23     7
27     5
43     4
42     3
26     3
41     3
19     2
39     2
18     2
16     2
20     2
36     2
47     2
46     2
54     1
17     1
53     1
50     1
Name: highwaympg, dtype: int64
```

In [18]:

```
## CHANGING THE CATEGORICAL ATTRIBUTES INTO NUMERIC DATA FOR BETTER ANALYSIS ##

df.replace({'fueltype':{'gas':0,'diesel':1}},inplace=True)
df.replace({'aspiration':{'std':0,'turbo':1}},inplace=True)
df.replace({'doornumber':{'two':0,'four':1}},inplace=True)
df.replace({'carbody':{'convertible':0,'hatchback':1,'sedan':2, 'wagon':3}},inplace=Tr
df.replace({'drivewheel':{'rwd':0,'fwd':1,'4wd':2}},inplace=True)
df.replace({'fuelsystem':{'mpfi':0,'2bbl':1,'1bbl':2,'mfi':3, 'spf1':4, 'idi':5}},inpl
```

In [19]:

```
## hot encoding ##
df = pd.get_dummies(df,drop_first=True)
df
```

Out[19]:

	car_ID	symboling	fueltype	aspiration	doornumber	drivewheel	wheelbase	carlength	carwidt
0	1	3	0	0	0	0	88.6	168.8	64
1	2	3	0	0	0	0	88.6	168.8	64
2	3	1	0	0	0	0	94.5	171.2	65
3	4	2	0	0	1	1	99.8	176.6	66
4	5	2	0	0	1	2	99.4	176.6	66
...	...	...	...	...	...	...	...	...	
200	201	-1	0	0	1	0	109.1	188.8	68
201	202	-1	0	1	1	0	109.1	188.8	68
202	203	-1	0	0	1	0	109.1	188.8	68
203	204	-1	1	1	1	0	109.1	188.8	68
204	205	-1	0	1	1	0	109.1	188.8	68

205 rows × 190 columns

In [20]:

```
## Splitting the data ##

x = df.drop(['price'], axis=1)
y = df['price']
print(len(x), len(y))
print(x)
print(y)
print(x.shape)
print(y.shape)
```

```
205 205
     car_ID  symboling  fueltype  aspiration  doornumber  drivewheel  \
0         1          3         0           0           0           0   
1         2          3         0           0           0           0   
2         3          1         0           0           0           0   
3         4          2         0           0           1           1   
4         5          2         0           0           1           2   
..      ...        ...       ...         ...         ...         ...   
200     201         -1         0           0           1           0   
201     202         -1         0           1           1           0   
202     203         -1         0           0           1           0   
203     204         -1         1           1           1           0   
204     205         -1         0           1           1           0   

     wheelbase  carlength  carwidth  carheight  ...  cylindernumber_three  \
0         88.6      168.8      64.1       48.8  ...                     0   
1         88.6      168.8      64.1       48.8  ...                     0   
2         94.5      171.2      65.5       52.4  ...                     0   
3         99.8      176.6      66.2       54.3  ...                     0   
4         99.4      176.6      66.4       54.3  ...                     0   
..         ...        ...       ...        ...  ...                   ...   
200      109.1      188.8      68.9       55.5  ...                     0   
201      109.1      188.8      68.8       55.5  ...                     0   
202      109.1      188.8      68.9       55.5  ...                     0   
203      109.1      188.8      68.9       55.5  ...                     0   
204      109.1      188.8      68.9       55.5  ...                     0   

     cylindernumber_twelve  cylindernumber_two  fuelsystem_1  fuelsystem_2  \
0                        0                   0             0             0   
1                        0                   0             0             0   
2                        0                   0             0             0   
3                        0                   0             0             0   
4                        0                   0             0             0   
..                     ...                 ...           ...           ...   
200                      0                   0             0             0   
201                      0                   0             0             0   
202                      0                   0             0             0   
203                      0                   0             0             0   
204                      0                   0             0             0   

     fuelsystem_3  fuelsystem_5  fuelsystem_4bbl  fuelsystem_spdi  \
0               0             0                0                0   
1               0             0                0                0   
2               0             0                0                0   
3               0             0                0                0   
4               0             0                0                0   
..            ...           ...              ...              ...   
200             0             0                0                0   
201             0             0                0                0   
202             0             0                0                0   
203             0             1                0                0   
204             0             0                0                0   

     fuelsystem_spfi  
0                  0  
1                  0  
2                  0  
3                  0  
4                  0  
..               ...  
```

```
200                0
201                0
202                0
203                0
204                0

[205 rows x 189 columns]
0      13495.0
1      16500.0
2      16500.0
3      13950.0
4      17450.0
        ...
200    16845.0
201    19045.0
202    21485.0
203    22470.0
204    22625.0
Name: price, Length: 205, dtype: float64
(205, 189)
(205,)
```

In [21]:

```
## Training and Test Data ##
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=1/10,random_state=0)
```

In [22]:

```
## Linear Regression ##

from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train,y_train)
model.score(x_test,y_test)
```

Out[22]:

```
-2.140565648522962
```

In [23]:

```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(x_train, y_train)
```

Out[23]:

```
LinearRegression()
```

In [24]:

```
t_data_predic = regressor.predict(x_train)
```

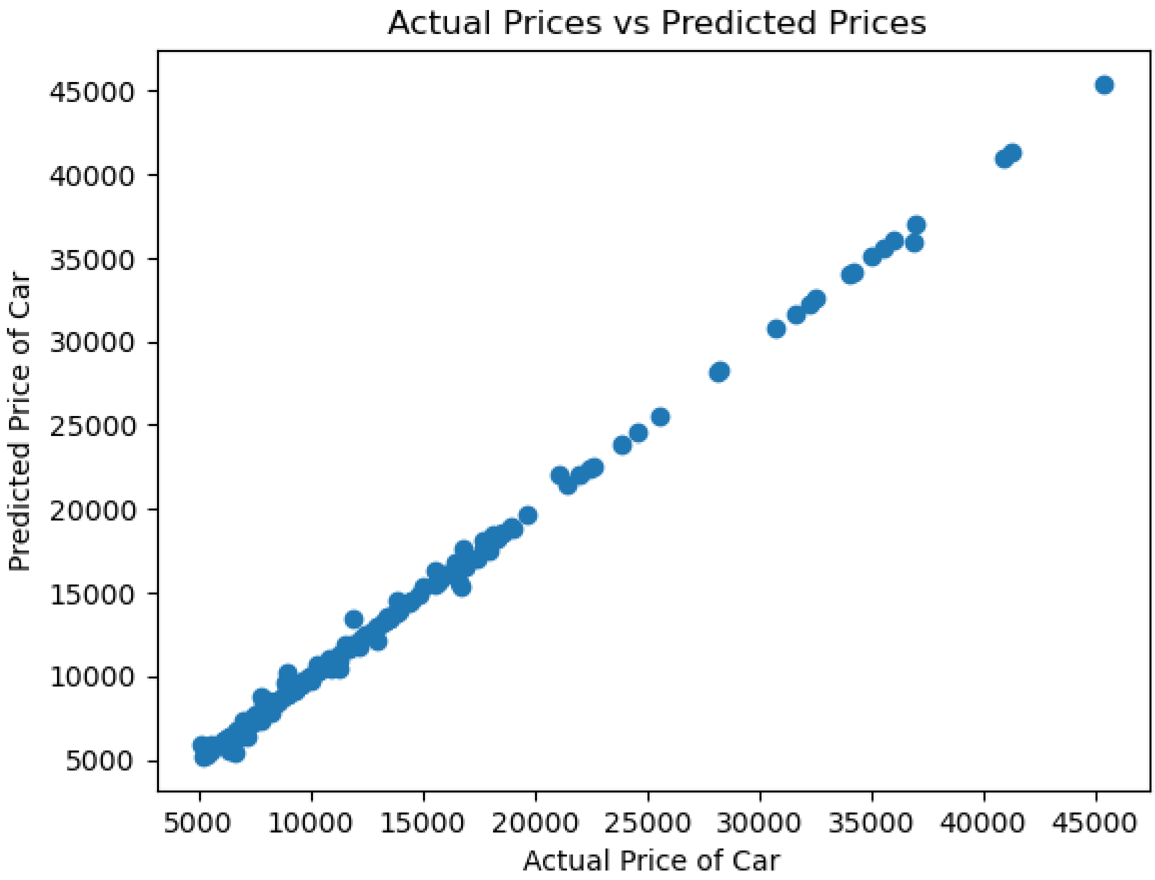
In [25]:

```
## Error Calculation ##

from sklearn import metrics
error_score = metrics.r2_score(y_train, t_data_predic)
print("R squared Error : ", error_score)
```

```
R squared Error :  0.9982334777472928
```

In [26]:

```
## Plotting THE data ##

plt.scatter(y_train, t_data_predic)
plt.xlabel("Actual Price of Car")
plt.ylabel("Predicted Price of Car")
plt.title(" Actual Prices vs Predicted Prices")
plt.show()
```



In [31]:

```
 ## prediction on Training data ##
t_data_predic = regressor.predict(x_test)
```

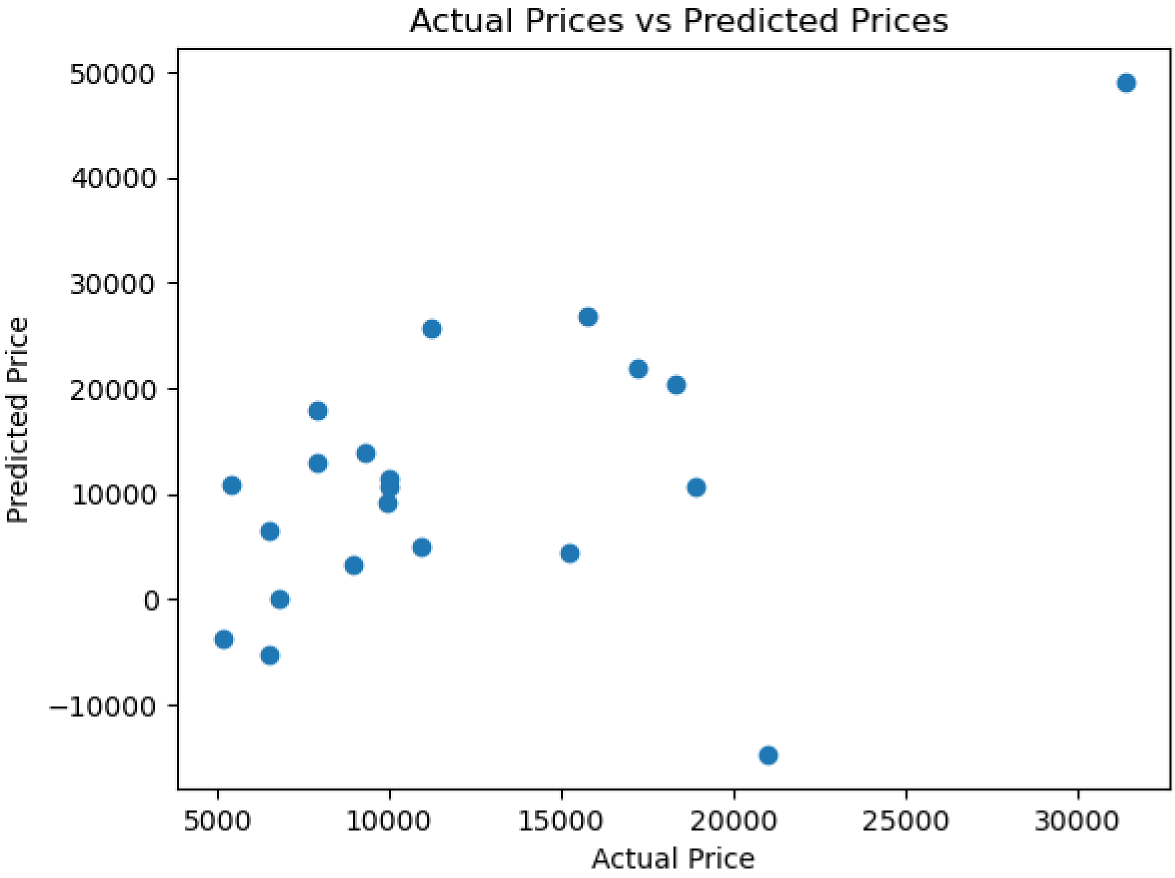
In [32]:

```
# R squared Error ##
error_score = metrics.r2_score(y_test, t_data_predic)
print("R squared Error : ", error_score)
```

```
R squared Error :  -2.140565648522962
```

In [33]:

```
plt.scatter(y_test, t_data_predic)
plt.xlabel("Actual Price")
plt.ylabel("Predicted Price")
plt.title(" Actual Prices vs Predicted Prices")
plt.show()
```


Actual Prices vs Predicted Prices
Predicted Price
50000
40000
30000
20000
10000
0
−10000
Actual Price
5000
10000
15000
20000
25000
30000

```
In [ ]:
```

```
In [ ]:
```