```
import pandas as pd
```

```
data=pd.read_csv('https://www.dropbox.com/s/5saczr8wmjketen/DiamondPrices.csv?dl=1')
```

```
data
```

	sn	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...	...	...
53935	53936	0.72	Ideal	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	53937	0.72	Good	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	53938	0.70	Very Good	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	53939	0.86	Premium	H	SI2	61.0	58.0	2757	6.15	6.12	3.74
53939	53940	0.75	Ideal	D	SI2	62.2	55.0	2757	5.83	5.87	3.64

53940 rows × 11 columns

```
data['cut'].unique()
```

```
array(['Ideal', 'Premium', 'Good', 'Very Good', 'Fair'], dtype=object)
```

```
data['cut'].value_counts()
```

```
Ideal        21551
Premium      13791
Very Good    12082
Good          4906
Fair          1610
Name: cut, dtype: int64
```

```
data['cut']=data['cut'].map({'Ideal':0,'Premium':1,'Very Good':2,'Good':3,"Fair":4})
```

```
data
```

	sn	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	0	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	1	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	3	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	1	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	3	J	SI2	63.3	58.0	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...	...	...
53935	53936	0.72	0	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	53937	0.72	3	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	53938	0.70	2	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	53939	0.86	1	H	SI2	61.0	58.0	2757	6.15	6.12	3.74

```
data["color"].unique()
```

```
array(['E', 'I', 'J', 'H', 'F', 'G', 'D'], dtype=object)
```

```
data["color"].value_counts()
```

```
G    11292
E     9797
F     9542
H     8304
D     6775
I     5422
J     2808
Name: color, dtype: int64
```

```
data["color"]=data["color"].map({"G":0,"E":1,"F":2,"H":3,"D":4,"I":5,"J":6})
```

```
data
```

	sn	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	0	1	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	1	1	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	3	1	VS1	56.9	65.0	327	4.05	4.07	2.31

```
data["clarity"].unique()
```

```
array(['SI2', 'SI1', 'VS1', 'VS2', 'VVS2', 'VVS1', 'I1', 'IF'],
      dtype=object)
```

```
data["clarity"].value_counts()
```

```
SI1     13065
VS2     12258
SI2      9194
VS1      8171
VVS2     5066
VVS1     3655
IF       1790
I1        741
Name: clarity, dtype: int64
```

```
data["clarity"]=data["clarity"].map({"SI1":0,"VS2":1,"SI2":2,"VS1":3,"VVS2":4,"VVS1":5,"I
```

```
data
```

	sn	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	0	1	2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	1	1	0	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	3	1	3	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	1	5	1	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	3	6	2	63.3	58.0	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...	...	...
53935	53936	0.72	0	4	0	60.8	57.0	2757	5.75	5.76	3.50
53936	53937	0.72	3	4	0	63.1	55.0	2757	5.69	5.75	3.61
53937	53938	0.70	2	4	0	62.8	60.0	2757	5.66	5.68	3.56
53938	53939	0.86	1	3	2	61.0	58.0	2757	6.15	6.12	3.74
53939	53940	0.75	0	4	2	62.2	55.0	2757	5.83	5.87	3.64

53940 rows × 11 columns

```
x=data.drop("price",axis=1)
```

```
y=data[["price"]]
```

```
x.shape,y.shape
```

```
((53940, 10), (53940, 1))
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,shuffle=True)
```

```
x_train.shape,y_train.shape,x_test.shape,y_test.shape
```

```
((37758, 10), (37758, 1), (16182, 10), (16182, 1))
```

```
from sklearn.linear_model import LinearRegression
```

```
model=LinearRegression()
```

```
model.fit(x_train,y_train)
```

```
LinearRegression()
```

```
predictions=model.predict(x_test)
```

```
from sklearn.metrics import r2_score
```

```
r2_score(y_test,predictions)
```

```
0.861767732227381
```

```
import numpy as np
```

```
p=pd.DataFrame(np.c_[predictions,y_test],columns=["Predicted prices","Actual prices"])
```

```
  p
```

	Predicted prices	Actual prices
0	9820.052551	12047.0
1	821.781205	1026.0
2	1599.871156	1720.0
3	6767.826533	5120.0
4	7225.653787	7042.0
...	...	...
16177	15986.392945	16369.0
16178	-543.272311	662.0
16179	409.453959	556.0
16180	5872.815877	5287.0
16181	5456.885059	5920.0

16182 rows × 2 columns

✓ 0s completed at 10:09 PM