In [4]:
```
# importing required libraries
import pandas as pd
```

In [5]:
```
# Reading dataframe from source
df = pd.read_csv(r"C:\Users\309962\Desktop\Sampledata_OneHotEncoding.txt")
```

In [7]:
```
#Using pandas to create dummy variables
dummies = pd.get_dummies(df.town)
dummies
```

Out[7]:

	monroe township	robinsville	west windsor
0	1	0	0
1	1	0	0
2	1	0	0
3	1	0	0
4	1	0	0
5	0	0	1
6	0	0	1
7	0	0	1
8	0	0	1
9	0	1	0
10	0	1	0
11	0	1	0
12	0	1	0

In [8]:
```
# Merging the data set with dummies
merged = pd.concat([df,dummies],axis='columns')
merged
```

Out[8]:

	town	area	price	monroe township	robinsville	west windsor
0	monroe township	2600	550000	1	0	0
1	monroe township	3000	565000	1	0	0
2	monroe township	3200	610000	1	0	0
3	monroe township	3600	680000	1	0	0
4	monroe township	4000	725000	1	0	0
5	west windsor	2600	585000	0	0	1
6	west windsor	2800	615000	0	0	1
7	west windsor	3300	650000	0	0	1
8	west windsor	3600	710000	0	0	1
9	robinsville	2600	575000	0	1	0
10	robinsville	2900	600000	0	1	0
11	robinsville	3100	620000	0	1	0
12	robinsville	3600	695000	0	1	0

In [9]:
```
# drop the township column as it is no longer relevant
final = merged.drop(['town'], axis='columns')
final
```

Out[9]:

	area	price	monroe township	robinsville	west windsor
0	2600	550000	1	0	0
1	3000	565000	1	0	0
2	3200	610000	1	0	0
3	3600	680000	1	0	0
4	4000	725000	1	0	0
5	2600	585000	0	0	1
6	2800	615000	0	0	1
7	3300	650000	0	0	1
8	3600	710000	0	0	1
9	2600	575000	0	1	0
10	2900	600000	0	1	0
11	3100	620000	0	1	0
12	3600	695000	0	1	0

In [10]:
```
#Dummy Variable Trap
#When you can derive one variable from other variables, they are known to be mult
#In this situation linear regression won't work as expected. Hence you need to dr
final = final.drop(['west windsor'], axis='columns')
final
```

Out[10]:

	area	price	monroe township	robinsville
0	2600	550000	1	0
1	3000	565000	1	0
2	3200	610000	1	0
3	3600	680000	1	0
4	4000	725000	1	0
5	2600	585000	0	0
6	2800	615000	0	0
7	3300	650000	0	0
8	3600	710000	0	0
9	2600	575000	0	1
10	2900	600000	0	1
11	3100	620000	0	1
12	3600	695000	0	1

In [11]:
```
# Drop price column as well as this is not required as it is a dependent variable
X = final.drop('price', axis='columns')
X
```

Out[11]:

	area	monroe township	robinsville
0	2600	1	0
1	3000	1	0
2	3200	1	0
3	3600	1	0
4	4000	1	0
5	2600	0	0
6	2800	0	0
7	3300	0	0
8	3600	0	0
9	2600	0	1
10	2900	0	1
11	3100	0	1
12	3600	0	1

In [12]:
```
y = final.price
```

In [13]:
```
# Create a Linear regression model object
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

In [14]:
```
# train the model based on data
model.fit(X,y)
```

Out[14]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [15]:
```
# Predict the price of a house
model.predict(X) # 2600 sqr ft home in new jersey
```

Out[15]:
```
array([539709.7398409 , 590468.71640508, 615848.20468716, 666607.18125134,
       717366.15781551, 579723.71533005, 605103.20361213, 668551.92431735,
       706621.15674048, 565396.15136531, 603465.38378844, 628844.87207052,
       692293.59277574])
```

In [16]:
```
# Identify the efficiency of the model
model.score(X,y)
```

Out[16]:
```
0.9573929037221873
```

In [17]:
```
# Predict the value of house in West Windsor
model.predict([[3400,0,0]]) # 3400 sqr ft home in west windsor
```

Out[17]:
```
array([681241.66845839])
```

In [19]:
```
# Predict the value of house in robbinsville
model.predict([[2800,0,1]]) # 2800 sqr ft home in robbinsville
```

Out[19]:
```
array([590775.63964739])
```

In [21]:
```
# Use Sklearn Label Encoder to encode the values

from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [40]:
```
dfle = df
dfle.town = le.fit_transform(dfle.town)
dfle
```

Out[40]:

	town	area	price
0	0	2600	550000
1	0	3000	565000
2	0	3200	610000
3	0	3600	680000
4	0	4000	725000
5	2	2600	585000
6	2	2800	615000
7	2	3300	650000
8	2	3600	710000
9	1	2600	575000
10	1	2900	600000
11	1	3100	620000
12	1	3600	695000

In [43]:
```
# Create an X variable which will feed to model
X = dfle[['town','area']].values
X
```

Out[43]:
```
array([[   0, 2600],
       [   0, 3000],
       [   0, 3200],
       [   0, 3600],
       [   0, 4000],
       [   2, 2600],
       [   2, 2800],
       [   2, 3300],
       [   2, 3600],
       [   1, 2600],
       [   1, 2900],
       [   1, 3100],
       [   1, 3600]], dtype=int64)
```

In [45]:
```
# It is a dependent variable
y = dfle.price
y
```

Out[45]:
```
0     550000
1     565000
2     610000
3     680000
4     725000
5     585000
6     615000
7     650000
8     710000
9     575000
10    600000
11    620000
12    695000
Name: price, dtype: int64
```

In [46]:
```
from sklearn.preprocessing import OneHotEncoder

ohe = OneHotEncoder(categorical_features = [0])
```

In [51]:
```
X = ohe.fit_transform(X).toarray()
X
```

Out[51]:
```
array([[1.0e+00, 0.0e+00, 2.6e+03],
       [1.0e+00, 0.0e+00, 3.0e+03],
       [1.0e+00, 0.0e+00, 3.2e+03],
       [1.0e+00, 0.0e+00, 3.6e+03],
       [1.0e+00, 0.0e+00, 4.0e+03],
       [0.0e+00, 1.0e+00, 2.6e+03],
       [0.0e+00, 1.0e+00, 2.8e+03],
       [0.0e+00, 1.0e+00, 3.3e+03],
       [0.0e+00, 1.0e+00, 3.6e+03],
       [1.0e+00, 0.0e+00, 2.6e+03],
       [1.0e+00, 0.0e+00, 2.9e+03],
       [1.0e+00, 0.0e+00, 3.1e+03],
       [1.0e+00, 0.0e+00, 3.6e+03]])
```

In [50]:
```
X = X[:,1:]
X
```

Out[50]:
```
array([[0.0e+00, 2.6e+03],
       [0.0e+00, 3.0e+03],
       [0.0e+00, 3.2e+03],
       [0.0e+00, 3.6e+03],
       [0.0e+00, 4.0e+03],
       [1.0e+00, 2.6e+03],
       [1.0e+00, 2.8e+03],
       [1.0e+00, 3.3e+03],
       [1.0e+00, 3.6e+03],
       [0.0e+00, 2.6e+03],
       [0.0e+00, 2.9e+03],
       [0.0e+00, 3.1e+03],
       [0.0e+00, 3.6e+03]])
```

In [52]:
```
model.fit(X,y)
```

Out[52]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [53]:
```
model.predict([[0,1,3400]]) # 3400 sqr ft home in west windsor
```

Out[53]:
```
array([679495.16770893])
```

In [54]:
```
model.predict([[1,0,2800]]) # 2800 sqr ft home in robbinsville
```

Out[54]:
```
array([578535.53155202])
```

In [ ]:
```

```