ASSIGNMENT 6

In [44]:

```
import pandas as pd
import numpy as np
import sklearn
```

In [45]:

```
train=pd.read_csv(r"C:\Users\VARNIKA\Desktop\Realestate.csv")
```

In [46]:

```
train
```

Out[46]:

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y house price of unit area
0	1	2012.917	32.0	84.87882	10	24.98298	121.54024	37.9
1	2	2012.917	19.5	306.59470	9	24.98034	121.53951	42.2
2	3	2013.583	13.3	561.98450	5	24.98746	121.54391	47.3
3	4	2013.500	13.3	561.98450	5	24.98746	121.54391	54.8
4	5	2012.833	5.0	390.56840	5	24.97937	121.54245	43.1
...	...	...	...	...	...	...	...	...
409	410	2013.000	13.7	4082.01500	0	24.94155	121.50381	15.4
410	411	2012.667	5.6	90.45606	9	24.97433	121.54310	50.0
411	412	2013.250	18.8	390.96960	7	24.97923	121.53986	40.6
412	413	2013.000	8.1	104.81010	5	24.96674	121.54067	52.5
413	414	2013.500	6.5	90.45606	9	24.97433	121.54310	63.9

414 rows × 8 columns

In [47]:

```
from sklearn.model_selection import train_test_split
x=train.drop('Y house price of unit area', axis=1)
y=train['Y house price of unit area']
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2)
```

In [48]:

```
from sklearn.linear_model import LinearRegression
linearreg=LinearRegression()
linearreg.fit(xtrain,ytrain)
```

Out[48]:

```
LinearRegression()
```

In [49]:

```
ypred=linearreg.predict(xtest)
print(ypred)
```

```
[47.02145994 35.67595974 45.24344794 12.17261921 37.72379818 49.38223505
 37.63778053 25.06681688 50.02672502 37.20242394 51.68595087 35.90202882
 46.05468489 35.61090722 40.90406044 44.07905914 41.29562615 15.16144082
 12.81092717 52.19687089 47.70974012 39.24740397 35.19736087 45.32412127
 47.82358845 16.52227373 13.50993987 35.16627715 31.53919528 42.0170308
 40.28366157 32.02905185 37.50414517 38.77175124 41.29153466 49.81202013
 14.99480708 46.86561121 45.5363228  48.74422461 34.85864872 33.08094502
 41.78363622 29.51941068 42.05376096 45.26568234 48.89134059 28.38343878
 39.36151193 40.11109773 11.65976738 46.76462528 32.60607056 36.5628347
 29.77868934 44.50254851 35.90119583 43.83783853  2.6689077  37.13318616
 44.3418886  32.4234277  43.67118869 35.7318615  29.05156319 47.36248451
 36.45065929 36.66676709 26.14519593 39.44025684 31.44431646 35.61249273
 44.4940641  29.70398763 31.79598365 39.23000774 47.56181749 48.27876703
 41.85102617 40.2835892  44.43673434 42.72457866 42.33605727]
```

In [50]:

```
from sklearn.metrics import r2_score
r2_score(ytest,ypred)
```

Out[50]:

```
0.4070079376688831
```

In [51]:

```
from sklearn import metrics
mse=metrics.mean_squared_error(ytest,ypred)
rmse=np.sqrt(mse)
rmse
```

Out[51]:

```
10.822305845414613
```

In [52]:

```
train['Y house price of unit area'].mean()
```

Out[52]:

```
37.98019323671498
```

In [53]:

```
#calculating error for each col
error=ytest-ypred
error
```

Out[53]:

```
26     9.178540
230   -2.275960
403   -5.543448
232    5.227381
91     5.476202
         ...   
217   -1.051026
6      0.016411
136    2.363266
142   -5.224579
224    3.663943
Name: Y house price of unit area, Length: 83, dtype: float64
```

In [54]:

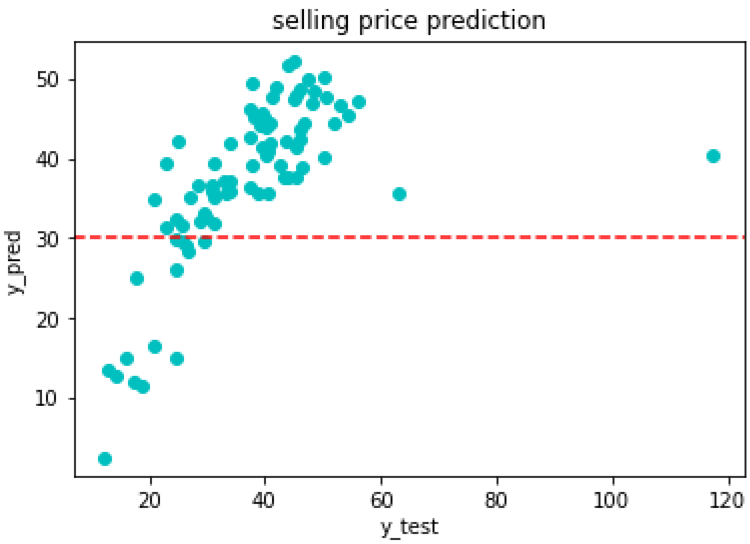
```
import matplotlib.pyplot as plt
```

In [55]:

```
plt.scatter(x=ytest,y=ypred,color='c')
plt.axhline(y=30,color='r',ls='--')
plt.xlabel("y_test")
plt.ylabel("y_pred")
plt.title("selling price prediction")
```

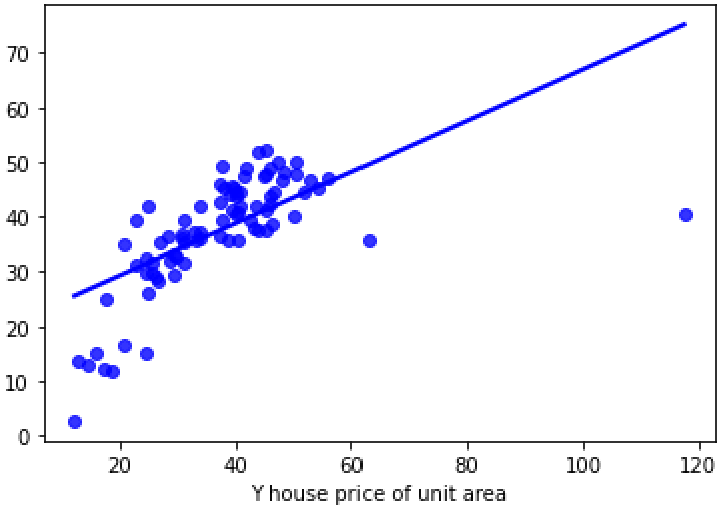
Out[55]:

```
Text(0.5, 1.0, 'selling price prediction')
```



```
In [56]: import seaborn as sns
         sns.regplot(x=ytest,y=ypred,ci=None, data=train,color='b')
```

```
Out[56]: <AxesSubplot:xlabel='Y house price of unit area'>
```



```
In [57]: y=train['Y house price of unit area']
         y=np.array(y)
         y.reshape(train.shape[0], 1)
```

```
Out[57]: array([[ 37.9],
                [ 42.2],
                [ 47.3],
                [ 54.8],
                [ 43.1],
                [ 32.1],
                [ 40.3],
                [ 46.7],
                [ 18.8],
                [ 22.1],
                [ 41.4],
                [ 58.1],
                [ 39.3],
                [ 23.8],
                [ 34.3],
                [ 50.5],
                [ 70.1],
                [ 37.4],
                [ 42.3],
                [ 47.7],
                [ 29.3],
                [ 51.6],
                [ 24.6],
                [ 47.9],
                [ 38.8],
                [ 27. ],
                [ 56.2],
                [ 33.6],
                [ 47. ],
                [ 57.1],
                [ 22.1],
                [ 25. ],
                [ 34.2],
                [ 49.3],
                [ 55.1],
                [ 27.3],
                [ 22.9],
                [ 25.3],
                [ 47.7],
                [ 46.2],
                [ 15.9],
                [ 18.2],
                [ 34.7],
                [ 34.1],
                [ 53.9],
                [ 38.3],
                [ 42. ],
                [ 61.5],
                [ 13.4],
                [ 13.2],
                [ 44.2],
                [ 20.7],
                [ 27. ],
                [ 38.9],
                [ 51.7],
                [ 13.7],
                [ 41.9],
                [ 53.5],
                [ 22.6],
                [ 42.4],
       [ 21.3],
       [ 63.2],
       [ 27.7],
       [ 55. ],
       [ 25.3],
       [ 44.3],
       [ 50.7],
       [ 56.8],
       [ 36.2],
       [ 42. ],
       [ 59. ],
       [ 40.8],
       [ 36.3],
       [ 20. ],
       [ 54.4],
       [ 29.5],
       [ 36.8],
       [ 25.6],
       [ 29.8],
       [ 26.5],
       [ 40.3],
       [ 36.8],
       [ 48.1],
       [ 17.7],
       [ 43.7],
       [ 50.8],
       [ 27. ],
       [ 18.3],
       [ 48. ],
       [ 25.3],
       [ 45.4],
       [ 43.2],
       [ 21.8],
       [ 16.1],
       [ 41. ],
       [ 51.8],
       [ 59.5],
       [ 34.6],
       [ 51. ],
       [ 62.2],
       [ 38.2],
       [ 32.9],
       [ 54.4],
       [ 45.7],
       [ 30.5],
       [ 71. ],
       [ 47.1],
       [ 26.6],
       [ 34.1],
       [ 28.4],
       [ 51.6],
       [ 39.4],
       [ 23.1],
       [  7.6],
       [ 53.3],
       [ 46.4],
       [ 12.2],
       [ 13. ],
       [ 30.6],
       [ 59.6],
[ 31.3],
[ 48. ],
[ 32.5],
[ 45.5],
[ 57.4],
[ 48.6],
[ 62.9],
[ 55. ],
[ 60.7],
[ 41. ],
[ 37.5],
[ 30.7],
[ 37.5],
[ 39.5],
[ 42.2],
[ 20.8],
[ 46.8],
[ 47.4],
[ 43.5],
[ 42.5],
[ 51.4],
[ 28.9],
[ 37.5],
[ 40.1],
[ 28.4],
[ 45.5],
[ 52.2],
[ 43.2],
[ 45.1],
[ 39.7],
[ 48.5],
[ 44.7],
[ 28.9],
[ 40.9],
[ 20.7],
[ 15.6],
[ 18.3],
[ 35.6],
[ 39.4],
[ 37.4],
[ 57.8],
[ 39.6],
[ 11.6],
[ 55.5],
[ 55.2],
[ 30.6],
[ 73.6],
[ 43.4],
[ 37.4],
[ 23.5],
[ 14.4],
[ 58.8],
[ 58.1],
[ 35.1],
[ 45.2],
[ 36.5],
[ 19.2],
[ 42. ],
[ 36.7],
[ 42.6],
       [ 15.5],
       [ 55.9],
       [ 23.6],
       [ 18.8],
       [ 21.8],
       [ 21.5],
       [ 25.7],
       [ 22. ],
       [ 44.3],
       [ 20.5],
       [ 42.3],
       [ 37.8],
       [ 42.7],
       [ 49.3],
       [ 29.3],
       [ 34.6],
       [ 36.6],
       [ 48.2],
       [ 39.1],
       [ 31.6],
       [ 25.5],
       [ 45.9],
       [ 31.5],
       [ 46.1],
       [ 26.6],
       [ 21.4],
       [ 44. ],
       [ 34.2],
       [ 26.2],
       [ 40.9],
       [ 52.2],
       [ 43.5],
       [ 31.1],
       [ 58. ],
       [ 20.9],
       [ 48.1],
       [ 39.7],
       [ 40.8],
       [ 43.8],
       [ 40.2],
       [ 78.3],
       [ 38.5],
       [ 48.5],
       [ 42.3],
       [ 46. ],
       [ 49. ],
       [ 12.8],
       [ 40.2],
       [ 46.6],
       [ 19. ],
       [ 33.4],
       [ 14.7],
       [ 17.4],
       [ 32.4],
       [ 23.9],
       [ 39.3],
       [ 61.9],
       [ 39. ],
       [ 40.6],
       [ 29.7],
[ 28.8],
[ 41.4],
[ 33.4],
[ 48.2],
[ 21.7],
[ 40.8],
[ 40.6],
[ 23.1],
[ 22.3],
[ 15. ],
[ 30. ],
[ 13.8],
[ 52.7],
[ 25.9],
[ 51.8],
[ 17.4],
[ 26.5],
[ 43.9],
[ 63.3],
[ 28.8],
[ 30.7],
[ 24.4],
[ 53. ],
[ 31.7],
[ 40.6],
[ 38.1],
[ 23.7],
[ 41.1],
[ 40.1],
[ 23. ],
[117.5],
[ 26.5],
[ 40.5],
[ 29.3],
[ 41. ],
[ 49.7],
[ 34. ],
[ 27.7],
[ 44. ],
[ 31.1],
[ 45.4],
[ 44.8],
[ 25.6],
[ 23.5],
[ 34.4],
[ 55.3],
[ 56.3],
[ 32.9],
[ 51. ],
[ 44.5],
[ 37. ],
[ 54.4],
[ 24.5],
[ 42.5],
[ 38.1],
[ 21.8],
[ 34.1],
[ 28.5],
[ 16.7],
[ 46.1],
       [ 36.9],
       [ 35.7],
       [ 23.2],
       [ 38.4],
       [ 29.4],
       [ 55. ],
       [ 50.2],
       [ 24.7],
       [ 53. ],
       [ 19.1],
       [ 24.7],
       [ 42.2],
       [ 78. ],
       [ 42.8],
       [ 41.6],
       [ 27.3],
       [ 42. ],
       [ 37.5],
       [ 49.8],
       [ 26.9],
       [ 18.6],
       [ 37.7],
       [ 33.1],
       [ 42.5],
       [ 31.3],
       [ 38.1],
       [ 62.1],
       [ 36.7],
       [ 23.6],
       [ 19.2],
       [ 12.8],
       [ 15.6],
       [ 39.6],
       [ 38.4],
       [ 22.8],
       [ 36.5],
       [ 35.6],
       [ 30.9],
       [ 36.3],
       [ 50.4],
       [ 42.9],
       [ 37. ],
       [ 53.5],
       [ 46.6],
       [ 41.2],
       [ 37.9],
       [ 30.8],
       [ 11.2],
       [ 53.7],
       [ 47. ],
       [ 42.3],
       [ 28.6],
       [ 25.7],
       [ 31.3],
       [ 30.1],
       [ 60.7],
       [ 45.3],
       [ 44.9],
       [ 45.1],
       [ 24.7],
       [ 47.1],
       [ 63.3],
       [ 40. ],
       [ 48. ],
       [ 33.1],
       [ 29.5],
       [ 24.8],
       [ 20.9],
       [ 43.1],
       [ 22.8],
       [ 42.1],
       [ 51.7],
       [ 41.5],
       [ 52.2],
       [ 49.5],
       [ 23.8],
       [ 30.5],
       [ 56.8],
       [ 37.4],
       [ 69.7],
       [ 53.3],
       [ 47.3],
       [ 29.3],
       [ 40.3],
       [ 12.9],
       [ 46.6],
       [ 55.3],
       [ 25.6],
       [ 27.3],
       [ 67.7],
       [ 38.6],
       [ 31.3],
       [ 35.3],
       [ 40.3],
       [ 24.7],
       [ 42.5],
       [ 31.9],
       [ 32.2],
       [ 23. ],
       [ 37.3],
       [ 35.5],
       [ 27.7],
       [ 28.5],
       [ 39.7],
       [ 41.2],
       [ 37.2],
       [ 40.5],
       [ 22.3],
       [ 28.1],
       [ 15.4],
       [ 50. ],
       [ 40.6],
       [ 52.5],
       [ 63.9]])
```

In [58]:

```
print(xtrain.shape)
print(ytrain.shape)
print(xtest.shape)
print(ytest.shape)
```

```
(331, 7)
(331,)
(83, 7)
(83,)
```

In [59]:

```
#to do a proper matrix multiplication of X and theta, we will needto add a column of 1
#The reason for doingso is that we are multiplying ϑ 2 with x 2 , ϑ1 with x 1 and ther
X= np.vstack((np.ones((x.shape[0], )), x.T)).T
```

In [60]:

```
def model(X, y, learning_rate, iteration): # took 4 parameters,Iterations specifies hc
    m = y.size
    theta = np.zeros((X.shape[1], 1)) #theta will be the vector of zeros. so it will b
    cost_list = []                    #We will also keep track of our cost at every it
    for i in range(iteration):
        y_pred = np.dot(X, theta)
        cost = (1/(2*m))*np.sum(np.square(y_pred - y))
        d_theta = (1/m)*np.dot(X.T, y_pred - y)
        theta = theta - learning_rate*d_theta
        cost_list.append(cost)
        # to print the cost for 10 times
        if(i%(iteration/10) == 0):
            print("Cost is :", cost)
    return theta, cost_list
```

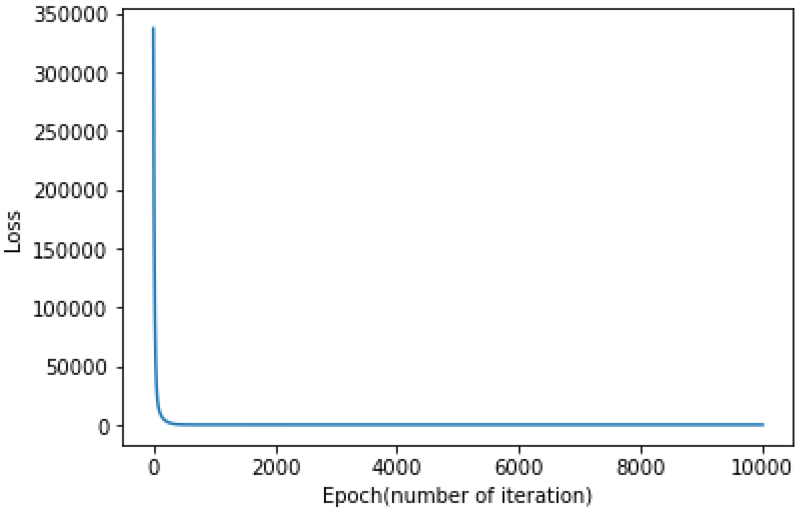
In [61]:

```
iteration = 10000
learning_rate = 0.000000005
theta, cost_list = model(X, y, learning_rate = learning_rate, iteration = iteration)
```

```
Cost is : 336827.17
Cost is : 11.246094111171214
Cost is : 9.452664173736165
Cost is : 8.208148638056988
Cost is : 7.127615180098399
Cost is : 6.189453914703845
Cost is : 5.3749057155778575
Cost is : 4.667683180863853
Cost is : 4.053644955806431
Cost is : 3.520512967077495
```

In [62]:

```
#We can see our cost decreasing with every iteration. We can also plot agraph of cost
rng = np.arange(0, iteration)
plt.plot(rng, cost_list)
plt.xlabel("Epoch(number of iteration)")
plt.ylabel("Loss")
plt.show()
```


350000
300000
250000
200000
150000
100000
50000
0
Loss
0
2000
4000
6000
8000
10000
Epoch(number of iteration)