In [166…
```
import copy, math
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [167…
```
dataset=pd.read_csv("HousingPrice.csv")
```

In [168…
```
dataset.head()
```

Out[168]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	furnished
1	12250000	8960	4	4	4	yes	no	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	unfurnished
3	12215000	7500	4	2	2	yes	no	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	furnished

In [196…
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   price             545 non-null    int64
 1   area              545 non-null    int64
 2   bedrooms          545 non-null    int64
 3   bathrooms         545 non-null    int64
 4   stories           545 non-null    int64
 5   mainroad          545 non-null    int64
 6   guestroom         545 non-null    int64
 7   basement          545 non-null    int64
 8   furnishingstatus  545 non-null    int64
dtypes: int64(9)
memory usage: 38.4 KB
```

In [197…
```
from sklearn.preprocessing import LabelEncoder
for x in dataset:
    dataset["mainroad"] = LabelEncoder().fit_transform(dataset["mainroad"])
    dataset["guestroom"] = LabelEncoder().fit_transform(dataset["guestroom"])
    dataset["basement"] = LabelEncoder().fit_transform(dataset["basement"])
    dataset["furnishingstatus"] = LabelEncoder().fit_transform(dataset["furnishingstatus"])
```

In [198…
```
dataset.head()
```

Out[198]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	furnishingstatus
0	13300000	7420	4	2	3	1	0	0	0
1	12250000	8960	4	4	4	1	0	0	0
2	12250000	9960	3	2	2	1	0	1	1
3	12215000	7500	4	2	2	1	0	1	0
4	11410000	7420	4	1	2	1	1	1	0

In [199…
```
dataset.shape
```

Out[199]:
```
(545, 9)
```

In [200…
```
x=dataset.iloc[:,1:].values
y=dataset.iloc[:,0].values
```

In [201…
```
(x)
```

Out[201]:
```
array([[7420,    4,    2, ...,    0,    0,    0],
       [8960,    4,    4, ...,    0,    0,    0],
       [9960,    3,    2, ...,    0,    1,    1],
       ...,
       [3620,    2,    1, ...,    0,    0,    1],
       [2910,    3,    1, ...,    0,    0,    0],
       [3850,    3,    1, ...,    0,    0,    1]], dtype=int64)
```

In [202…
```
y
```

Out[202]:
```
array([13300000, 12250000, 12250000, 12215000, 11410000, 10850000,
       10150000, 10150000,  9870000,  9800000,  9800000,  9681000,
        9310000,  9240000,  9240000,  9100000,  9100000,  8960000,
        8890000,  8855000,  8750000,  8680000,  8645000,  8645000,
        8575000,  8540000,  8463000,  8400000,  8400000,  8400000,
        8400000,  8400000,  8295000,  8190000,  8120000,  8080940,
        8043000,  7980000,  7962500,  7910000,  7875000,  7840000,
        7700000,  7700000,  7560000,  7560000,  7525000,  7490000,
        7455000,  7420000,  7420000,  7420000,  7350000,  7350000,
        7350000,  7350000,  7343000,  7245000,  7210000,  7210000,
        7140000,  7070000,  7070000,  7035000,  7000000,  6930000,
        6930000,  6895000,  6860000,  6790000,  6790000,  6755000,
        6720000,  6685000,  6650000,  6650000,  6650000,  6650000,
        6650000,  6650000,  6629000,  6615000,  6615000,  6580000,
        6510000,  6510000,  6510000,  6475000,  6475000,  6440000,
        6440000,  6419000,  6405000,  6300000,  6300000,  6300000,
        6300000,  6300000,  6293000,  6265000,  6230000,  6230000,
        6195000,  6195000,  6195000,  6160000,  6160000,  6125000,
        6107500,  6090000,  6090000,  6090000,  6083000,  6083000,
        6020000,  6020000,  6020000,  5950000,  5950000,  5950000,
        5950000,  5950000,  5950000,  5950000,  5950000,  5943000,
        5880000,  5880000,  5873000,  5873000,  5866000,  5810000,
        5810000,  5810000,  5803000,  5775000,  5740000,  5740000,
        5740000,  5740000,  5740000,  5652500,  5600000,  5600000,
        5600000,  5600000,  5600000,  5600000,  5600000,  5600000,
        5600000,  5565000,  5565000,  5530000,  5530000,  5530000,
        5523000,  5495000,  5495000,  5460000,  5460000,  5460000,
        5460000,  5425000,  5390000,  5383000,  5320000,  5285000,
        5250000,  5250000,  5250000,  5250000,  5250000,  5250000,
        5250000,  5250000,  5250000,  5243000,  5229000,  5215000,
        5215000,  5215000,  5145000,  5145000,  5110000,  5110000,
        5110000,  5110000,  5075000,  5040000,  5040000,  5040000,
        5040000,  5033000,  5005000,  4970000,  4970000,  4956000,
        4935000,  4907000,  4900000,  4900000,  4900000,  4900000,
        4900000,  4900000,  4900000,  4900000,  4900000,  4900000,
        4900000,  4900000,  4893000,  4893000,  4865000,  4830000,
        4830000,  4830000,  4830000,  4795000,  4795000,  4767000,
        4760000,  4760000,  4760000,  4753000,  4690000,  4690000,
        4690000,  4690000,  4690000,  4690000,  4655000,  4620000,
        4620000,  4620000,  4620000,  4620000,  4613000,  4585000,
        4585000,  4550000,  4550000,  4550000,  4550000,  4550000,
        4550000,  4550000,  4543000,  4543000,  4515000,  4515000,
        4515000,  4515000,  4480000,  4480000,  4480000,  4480000,
        4480000,  4473000,  4473000,  4473000,  4445000,  4410000,
        4410000,  4403000,  4403000,  4403000,  4382000,  4375000,
        4340000,  4340000,  4340000,  4340000,  4340000,  4319000,
        4305000,  4305000,  4277000,  4270000,  4270000,  4270000,
```

```
4270000,  4270000,  4270000,  4235000,  4235000,  4200000,
4200000,  4200000,  4200000,  4200000,  4200000,  4200000,
4200000,  4200000,  4200000,  4200000,  4200000,  4200000,
4200000,  4200000,  4200000,  4200000,  4193000,  4193000,
4165000,  4165000,  4165000,  4130000,  4130000,  4123000,
4098500,  4095000,  4095000,  4095000,  4060000,  4060000,
4060000,  4060000,  4060000,  4025000,  4025000,  4025000,
4007500,  4007500,  3990000,  3990000,  3990000,  3990000,
3990000,  3920000,  3920000,  3920000,  3920000,  3920000,
3920000,  3920000,  3885000,  3885000,  3850000,  3850000,
3850000,  3850000,  3850000,  3850000,  3850000,  3836000,
3815000,  3780000,  3780000,  3780000,  3780000,  3780000,
3780000,  3773000,  3773000,  3773000,  3745000,  3710000,
3710000,  3710000,  3710000,  3710000,  3703000,  3703000,
3675000,  3675000,  3675000,  3675000,  3640000,  3640000,
3640000,  3640000,  3640000,  3640000,  3640000,  3640000,
3640000,  3633000,  3605000,  3605000,  3570000,  3570000,
3570000,  3570000,  3535000,  3500000,  3500000,  3500000,
3500000,  3500000,  3500000,  3500000,  3500000,  3500000,
3500000,  3500000,  3500000,  3500000,  3500000,  3500000,
3500000,  3500000,  3493000,  3465000,  3465000,  3465000,
3430000,  3430000,  3430000,  3430000,  3430000,  3430000,
3423000,  3395000,  3395000,  3395000,  3360000,  3360000,
3360000,  3360000,  3360000,  3360000,  3360000,  3360000,
3353000,  3332000,  3325000,  3325000,  3290000,  3290000,
3290000,  3290000,  3290000,  3290000,  3290000,  3290000,
3255000,  3255000,  3234000,  3220000,  3220000,  3220000,
3220000,  3150000,  3150000,  3150000,  3150000,  3150000,
3150000,  3150000,  3150000,  3150000,  3143000,  3129000,
3118850,  3115000,  3115000,  3115000,  3087000,  3080000,
3080000,  3080000,  3080000,  3045000,  3010000,  3010000,
3010000,  3010000,  3010000,  3010000,  3010000,  3003000,
2975000,  2961000,  2940000,  2940000,  2940000,  2940000,
2940000,  2940000,  2940000,  2940000,  2870000,  2870000,
2870000,  2870000,  2852500,  2835000,  2835000,  2835000,
2800000,  2800000,  2730000,  2730000,  2695000,  2660000,
2660000,  2660000,  2660000,  2660000,  2660000,  2660000,
2653000,  2653000,  2604000,  2590000,  2590000,  2590000,
2520000,  2520000,  2520000,  2485000,  2485000,  2450000,
2450000,  2450000,  2450000,  2450000,  2450000,  2408000,
2380000,  2380000,  2380000,  2345000,  2310000,  2275000,
2275000,  2275000,  2240000,  2233000,  2135000,  2100000,
2100000,  2100000,  1960000,  1890000,  1890000,  1855000,
1820000,  1767150,  1750000,  1750000,  1750000], dtype=int64)
```

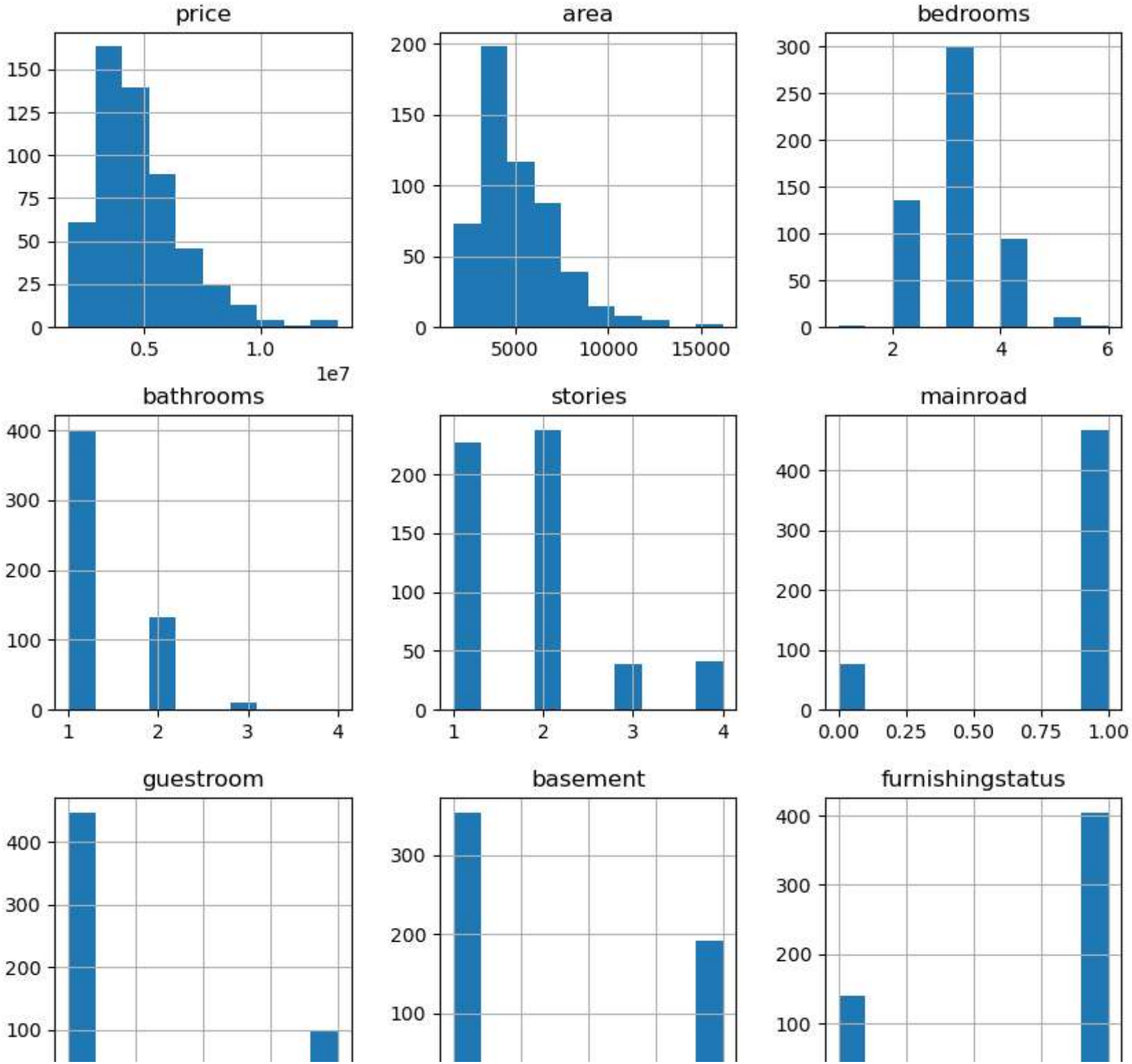
In [203…

```
dataset.isna().sum()
```

Out[203]:
```
price               0
area                0
bedrooms            0
bathrooms           0
stories             0
mainroad            0
guestroom           0
basement            0
furnishingstatus    0
dtype: int64
```

In [204…
```
dataset.hist(bins=10, figsize=(10, 10))
```

Out[204]:
```
array([[<Axes: title={'center': 'price'}>,
        <Axes: title={'center': 'area'}>,
        <Axes: title={'center': 'bedrooms'}>],
       [<Axes: title={'center': 'bathrooms'}>,
        <Axes: title={'center': 'stories'}>,
        <Axes: title={'center': 'mainroad'}>],
       [<Axes: title={'center': 'guestroom'}>,
        <Axes: title={'center': 'basement'}>,
        <Axes: title={'center': 'furnishingstatus'}>]], dtype=object)
```


price
1e7
area
bedrooms
bathrooms
stories
mainroad
guestroom
basement
furnishingstatus

In [205…
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 0)
```

In [211…
```
b_init = 909.909840909
w_init = np.array([ 0.39133572,0.39133535, 18.39133535, 0.10003020, 0.42131618, 0.39133535, 0.10003020, 0.42131618])
```

In [214…
```
def compute_cost(x, y, w, b):
    m = x.shape[0]
    cost = 0.0
    for i in range(m):
        f_wb_i = np.dot(x[i], w) + b
        cost = cost + (f_wb_i - y[i])**2
    cost = cost / (2 * m)
    return cost
```

In [215…
```
cost = compute_cost(x_train, y_train, w_init, b_init)
print(cost)
```

```
13353346324298.469
```

In [219…
```
def dwdb(x, y, w, b):
    m = x.shape[0]
    dj_dw = np.zeros((n,))
    c=0.0
    for i in range(m):
        f_wb_i = np.dot(x[i], w)+b
        for j in range(n):
            dj_dw[j] = dj_dw[j]+(f_wb_i-y[i])*x[i,j]
        c+=(f_wb_i - y[i])
    dj_dw =  dj_dw/ (m)
    c=c/(m)
    return dj_dw,c
```

In [220…
```
def gradient_descent(X, y, w_in, b_in, cost_function, gradient_function, alpha, num_iters):
    J_history = []
    w = w_in.copy()
    b = b_in
    for i in range(num_iters):
        dj_db,dj_dw = gradient_function(X, y, w, b)
        w=w-alpha*dj_dw
        b=b-alpha*dj_db
```

```
        if i<100000:
            J_history.append( cost_function(X, y, w, b))
        if i% math.ceil(num_iters / 10) == 0:
            print(f"Iteration {i:4d}: Cost {J_history[-1]:8.2f}   ")
    return w, b, J_history
```

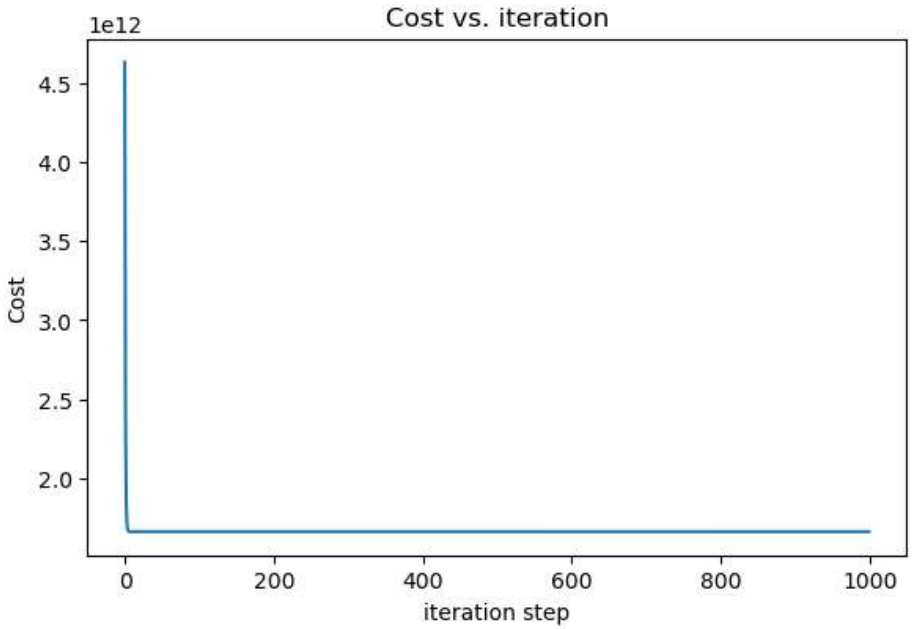
In [234…

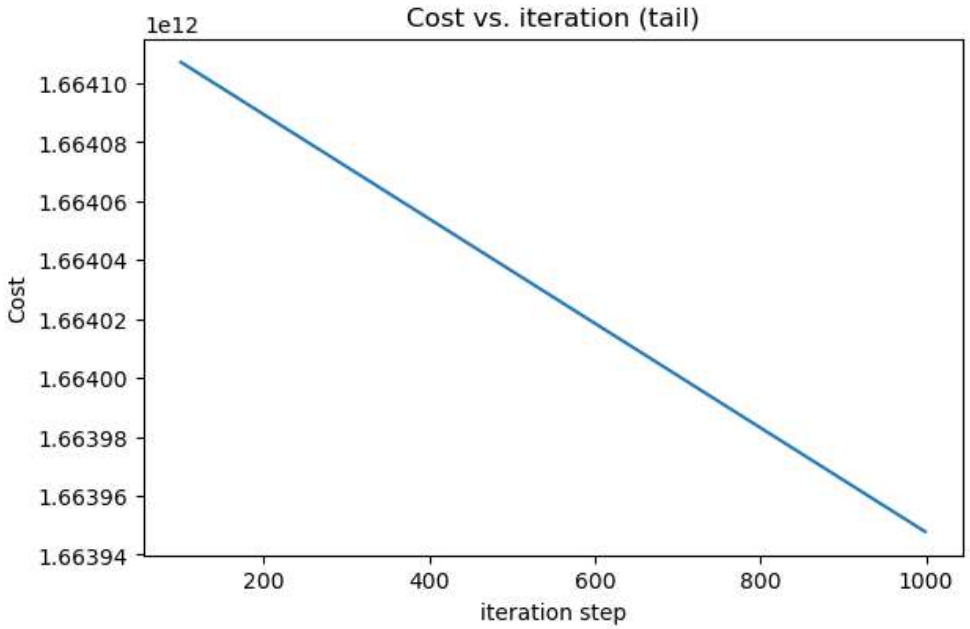
```
initial_w = np.zeros_like(w_init)
initial_b = 0.
iterations = 1000
alpha = 5.0e-8
w_final, b_final, J_hist = gradient_descent(x_train, y_train, initial_w, initial_b,compute_cost, compute_gradient, alpha, iterations
print(f"b,w found by gradient descent: {b_final:0.2f},{w_final} ")
m,n= x_test.shape
ypred2=[]
for i in range(m):
    ypred2.append(np.dot(x_test[i], w_final)+b_final)
```

```
Iteration    0: Cost 4629415499395.88
Iteration  100: Cost 1664107146998.18
Iteration  200: Cost 1664089419542.41
Iteration  300: Cost 1664071692523.23
Iteration  400: Cost 1664053965940.63
Iteration  500: Cost 1664036239794.60
Iteration  600: Cost 1664018514085.13
Iteration  700: Cost 1664000788812.21
Iteration  800: Cost 1663983063975.82
Iteration  900: Cost 1663965339575.96
b,w found by gradient descent: 15.34,[882.28439695  62.51811989  36.52582778  55.06189717  14.20958023
   8.22053637  12.32476198   6.6619177 ]
```

In [235…

```
fig, (ax1, ax2) = plt.subplots(1, 2, constrained_layout=True, figsize=(12, 4))
ax1.plot(J_hist)
ax2.plot(100 + np.arange(len(J_hist[100:])), J_hist[100:])
ax1.set_title("Cost vs. iteration");  ax2.set_title("Cost vs. iteration (tail)")
ax1.set_ylabel('Cost')             ;  ax2.set_ylabel('Cost')
ax1.set_xlabel('iteration step')   ;  ax2.set_xlabel('iteration step')
plt.show()
```





In [236…

```
ypred2
```

Out[236]:
```
[3529501.3406306556,
 8487896.914373336,
 3053136.946313428,
 11646412.537346473,
 3229579.616123866,
 5602969.481732372,
 3397110.2811081833,
 3070705.906552376,
 3098835.636818063,
 7941132.493139452,
 5294285.964197079,
 3494209.9647525214,
 3044263.9089859873,
 5346972.681755754,
 5280799.79336558,
 2144252.227854181,
 4323513.766448471,
 5311667.422496969,
 2735450.1604720782,
 3970525.95909028,
 7190990.259133756,
 5691106.784313047,
 4411792.399501713,
 1500253.8895556328,
 5664703.62983463,
 2779561.6124492665,
 6176318.456273238,
 5011754.00996637,
 2911994.3010985507,
 5664586.8441225905,
 3882304.181312645,
 2007622.4797009828,
 7411579.894441482,
 3564681.798409436,
 2753095.8484111456,
 5682164.351097493,
 5294151.639336934,
 2678998.168647297,
 3044064.8241693685,
 1919345.3138265947,
 7499927.776542888,
 5205945.515889148,
 5117766.566686522,
 4848480.438858435,
 3105956.4301135754,
 5294168.384180024,
 3670638.989462075,
```

3564681.798409436,
3570105.4095701887,
5029350.298852545,
4270626.895989175,
8529533.269329056,
1456137.784889709,
3397047.7629882977,
5188147.606923499,
5046974.647188245,
7279301.615407378,
4094232.5347183906,
3176476.6637499584,
6176373.518170407,
11381777.411618374,
5823567.023873106,
1884069.9184778933,
5294216.784159493,
2329462.7711768,
2485864.278700619,
6313117.284165164,
3891112.8157019448,
3970600.801972143,
5558707.053643909,
1892876.109044154,
4853089.468136887,
3176544.844714121,
3759000.9897038983,
3335363.462397083,
2532484.661169527,
3017893.8789749458,
7570561.966572372,
2567742.076474092,
3529390.4225313016,
1892876.109044154,
6781573.773119614,
3979584.4137042155,
6310461.759827639,
4058941.1588402563,
2859089.440880859,
3017720.5341955232,
5117522.882345696,
2779571.927982279,
3595644.8157210397,
6550380.50623925,
7940880.588262253,
3211761.3777103922,
2779483.326130475,

```
1892926.7530123545,
7905914.168916886,
1906102.8273359216,
5779432.258727088,
6855595.93712776,
7411435.107208375,
9264455.626692852,
3070768.4246722613,
2459389.411073901,
2647235.930356976,
5664721.168982734,
4588304.340789554,
5346908.605017196,
3970587.6829051496,
5382292.3629252985,
3194122.3516890253,
3529390.4225313016,
2682383.191875844,
3176476.6637499584,
8699639.506797865,
3501260.6922656153,
3388342.499035819,
8646864.769361874,
3211768.0396280927,
3105942.7625834513,
11646530.11736353,
2382399.834994002,
3794410.739904103,
3970518.4114277465,
4407437.284329424,
2707983.648967725,
3840120.41881559,
7835098.048352803,
11420604.58700202,
6317483.959785775,
3176781.798126669,
5867569.8750111265,
3573497.980461269,
2647353.5103740306,
3211768.0396280927,
4729414.782404922,
14293575.733274663,
5258805.9658752885]
```

In [237…

```
y_test
```

Out[237]:
```
array([ 4585000,  6083000,  4007500,  6930000,  2940000,  6195000,
        3535000,  2940000,  3500000,  7980000,  6755000,  3990000,
        3150000,  3290000,  4130000,  2660000,  4410000,  3710000,
        3360000,  4270000,  5005000,  5383000,  6440000,  1890000,
        6125000,  5460000,  5803000,  4620000,  5530000,  5950000,
        4305000,  3640000,  5250000,  3325000,  3703000,  4753000,
        9100000,  3500000,  3150000,  4270000,  8960000,  4060000,
        5740000,  3129000,  3633000,  7560000,  4620000,  3290000,
        4165000,  6650000,  4165000,  4690000,  3150000,  3850000,
        3290000,  5075000,  6510000,  5740000,  3780000,  4795000,
        4900000,  5460000,  3500000,  7525000,  2835000,  5495000,
        8680000,  4200000,  4200000,  4900000,  3332000,  6195000,
        4098500,  6650000,  3885000,  4620000,  1960000,  6440000,
        1750000,  3605000,  3290000,  4970000,  4613000,  3850000,
        3500000,  6107500,  3780000,  4900000,  3570000,  4340000,
        3500000,  6300000,  3395000,  3815000,  3920000, 12250000,
        3080000,  9310000,  4270000,  3780000,  5600000,  3290000,
        2380000,  5110000,  6650000,  5810000,  4123000,  3080000,
        5530000,  1750000,  2695000,  2870000,  2590000,  4515000,
        4410000,  4585000,  5250000,  3570000,  3640000,  9800000,
        2940000,  6083000,  3255000,  4893000,  3150000,  2975000,
        6930000,  3500000,  5880000,  3500000,  4235000,  3710000,
        4060000,  2345000,  4550000, 10150000,  3640000], dtype=int64)
```

In [206…
```
from sklearn.linear_model import LinearRegression
Regression = LinearRegression()
Regression.fit(x_train, y_train)
```

Out[206]:
▾ LinearRegression

LinearRegression()

In [207…
```
y_pred = Regression.predict(x_test)
```

In [209…
```
y_pred
```

Out[209]:
```
array([ 4260237.12243919,  6298881.98784387,  3968898.41792428,
        7325536.08692192,  3388571.68653583,  7272922.13712766,
        3348123.14061415,  3070351.52414203,  2973389.64489254,
        8485469.64748818,  7060401.91989973,  4245929.94944481,
        4246403.13053695,  5521116.50586652,  4646687.88896961,
        2188355.90295325,  4511200.26766419,  4124287.27555935,
        3469352.77047575,  3678806.57086361,  7086769.89847281,
        6478715.04621159,  4265886.69428888,  3085544.92225241,
        5815143.05545604,  5650403.02718496,  5009732.40370204,
        5004681.54971597,  6533017.94512741,  5154308.14829335,
        3291008.88536781,  3797642.86413291,  6512488.70311426,
        3162244.32841838,  3476506.35697294,  4020674.70808127,
        5412323.29284785,  4098323.8900451 ,  3049405.39584201,
        3634072.30950328,  8052792.10113933,  5894826.12066123,
        6735028.17100874,  4471425.01978844,  3230088.9634105 ,
        6300131.70589438,  4489496.03562356,  3162244.32841838,
        4459580.00815323,  5392351.94891708,  4208658.00231136,
        8389639.64829007,  2950717.74942942,  3094285.25669507,
        4074212.17007901,  4269577.92426866,  7456939.93983818,
        4390960.02125853,  3004865.42548018,  5516164.7337883 ,
        6937150.7836028 ,  7292637.15894587,  3759406.18886564,
        7158593.78899049,  2361660.43026451,  5639703.42965172,
        6730878.97023757,  2642723.08225605,  4467563.84456304,
        4224437.48652002,  4131661.36460509,  7123297.28807348,
        4145652.45218946,  5425175.23447161,  4606172.33834411,
        4280684.15272193,  3556566.53248862,  8746350.91145343,
        2712071.73189566,  3147937.155424  ,  4131661.36460509,
        5750673.33507828,  6049514.45546684,  5387642.35439645,
        4376652.84826415,  4396641.29654145,  3446915.47709173,
        4822252.24447411,  4491129.08608893,  4459093.2930105 ,
        3974606.45130177,  6077120.80643095,  3371202.3514844 ,
        3897099.87937307,  4697170.52208777, 10811429.18781319,
        3133134.20510778,  6244260.88839995,  4848417.96315429,
        5073755.9378158 ,  7657094.22159512,  3324189.40806111,
        5250343.34196335,  4085447.43435016,  6802386.77534523,
        4843854.88934705,  4419675.95441499,  3833209.14794003,
        6209984.12099799,  3012019.01197737,  3147937.155424  ,
        2152702.40719848,  3004865.42548018,  5497775.88301997,
        3244899.03467348,  3850978.67745182,  7902291.60871604,
        3019172.59847456,  4681303.82593146,  8085806.30092727,
        2383121.18975608,  6299187.928314  ,  2674914.2214934 ,
        5716556.59793745,  2814937.70397977,  3636085.77933863,
        7710764.97419377,  6600858.92088678,  5066961.09567957,
        4526649.29124278,  5351630.49482972,  3517850.87467681,
        4845717.64835551,  3019172.59847456,  4394651.25123832,
       11119837.61387326,  5639751.15045699])
```

In [210…]:
```
y_test
```

Out[210]:
```
array([ 4585000,  6083000,  4007500,  6930000,  2940000,  6195000,
        3535000,  2940000,  3500000,  7980000,  6755000,  3990000,
        3150000,  3290000,  4130000,  2660000,  4410000,  3710000,
        3360000,  4270000,  5005000,  5383000,  6440000,  1890000,
        6125000,  5460000,  5803000,  4620000,  5530000,  5950000,
        4305000,  3640000,  5250000,  3325000,  3703000,  4753000,
        9100000,  3500000,  3150000,  4270000,  8960000,  4060000,
        5740000,  3129000,  3633000,  7560000,  4620000,  3290000,
        4165000,  6650000,  4165000,  4690000,  3150000,  3850000,
        3290000,  5075000,  6510000,  5740000,  3780000,  4795000,
        4900000,  5460000,  3500000,  7525000,  2835000,  5495000,
        8680000,  4200000,  4200000,  4900000,  3332000,  6195000,
        4098500,  6650000,  3885000,  4620000,  1960000,  6440000,
        1750000,  3605000,  3290000,  4970000,  4613000,  3850000,
        3500000,  6107500,  3780000,  4900000,  3570000,  4340000,
        3500000,  6300000,  3395000,  3815000,  3920000, 12250000,
        3080000,  9310000,  4270000,  3780000,  5600000,  3290000,
        2380000,  5110000,  6650000,  5810000,  4123000,  3080000,
        5530000,  1750000,  2695000,  2870000,  2590000,  4515000,
        4410000,  4585000,  5250000,  3570000,  3640000,  9800000,
        2940000,  6083000,  3255000,  4893000,  3150000,  2975000,
        6930000,  3500000,  5880000,  3500000,  4235000,  3710000,
        4060000,  2345000,  4550000, 10150000,  3640000], dtype=int64)
```

In [ ]:
```

```

In [ ]:
```

```