In [186…
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
os.getcwd()
```

Out[1863]:
```
'C:\\Users\\pha-a\\OneDrive\\Desktop\\4105'
```

In [186…
```
dataset = pd.read_csv('C:\\Users\\pha-a\\OneDrive\\Desktop\\4105\\D3.csv')
print(dataset)
dataset.head()
m = len(dataset)
print(m)
```

```
          X1        X2        X3         Y
0   0.000000  3.440000  0.440000  4.387545
1   0.040404  0.134949  0.888485  2.679650
2   0.080808  0.829899  1.336970  2.968490
3   0.121212  1.524848  1.785455  3.254065
4   0.161616  2.219798  2.233939  3.536375
..       ...       ...       ...       ...
95  3.838384  1.460202  3.046061 -4.440595
96  3.878788  2.155152  3.494545 -4.458663
97  3.919192  2.850101  3.943030 -4.479995
98  3.959596  3.545051  0.391515 -3.304593
99  4.000000  0.240000  0.840000 -5.332455

[100 rows x 4 columns]
100
```

In [186…
```
x0 = np.ones((m, 1))
x1 = dataset.values[:,0]
x2 = dataset.values[:,1]
x3 = dataset.values[:,2]
y = dataset.values[:,3]

print("x0 = ", x0[:10])
print("x1 = ", x1[:10])
print("x2 = ", x2[:10])
print("x3 = ", x3[:10])
print("y = ", y[:10])
```

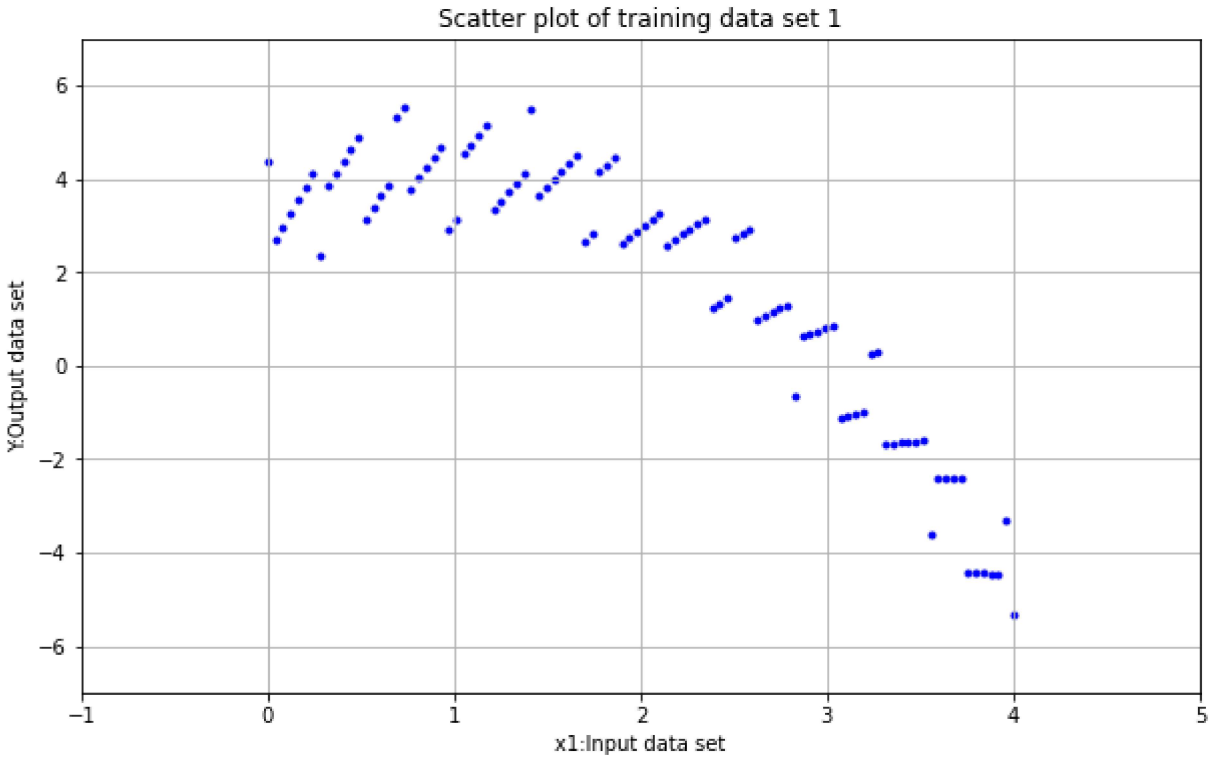
```
x0 =  [[1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]]
x1 =  [0.         0.04040404 0.08080808 0.12121212 0.16161616 0.2020202
 0.24242424 0.28282828 0.32323232 0.36363636]
x2 =  [3.44       0.1349495  0.82989899 1.52484848 2.21979798 2.91474747
 3.60969697 0.30464646 0.99959596 1.69454546]
x3 =  [0.44       0.88848485 1.3369697  1.78545454 2.23393939 2.68242424
 3.13090909 3.57939394 0.02787879 0.47636364]
y =  [4.38754501 2.6796499  2.96848981 3.25406475 3.53637472 3.81541972
 4.09119974 2.36371479 3.83296487 4.09894997]
```

In [186…

```
plt.scatter(x1,y, color='blue', marker='.')
plt.grid()
plt.xlim([-1,5])
plt.ylim([-7,7])
plt.xlabel('x1:Input data set')
plt.ylabel('Y:Output data set')
plt.title('Scatter plot of training data set 1')
```

Out[1866]:

```
Text(0.5, 1.0, 'Scatter plot of training data set 1')
```

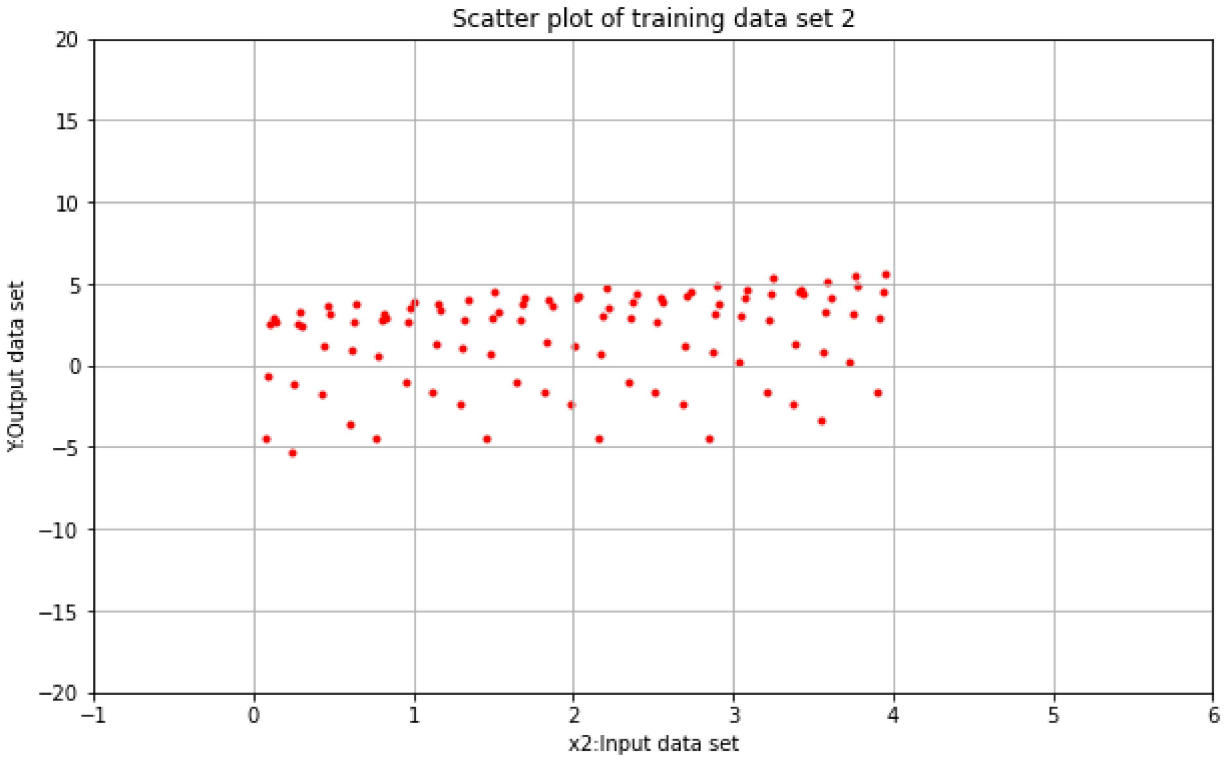


In [186…

```
plt.scatter(x2,y, color='red', marker='.')
plt.grid()
plt.xlim([-1,6])
plt.ylim([-20,20])
plt.xlabel('x2:Input data set')
```

```
plt.ylabel('Y:Output data set')
plt.title('Scatter plot of training data set 2')
```

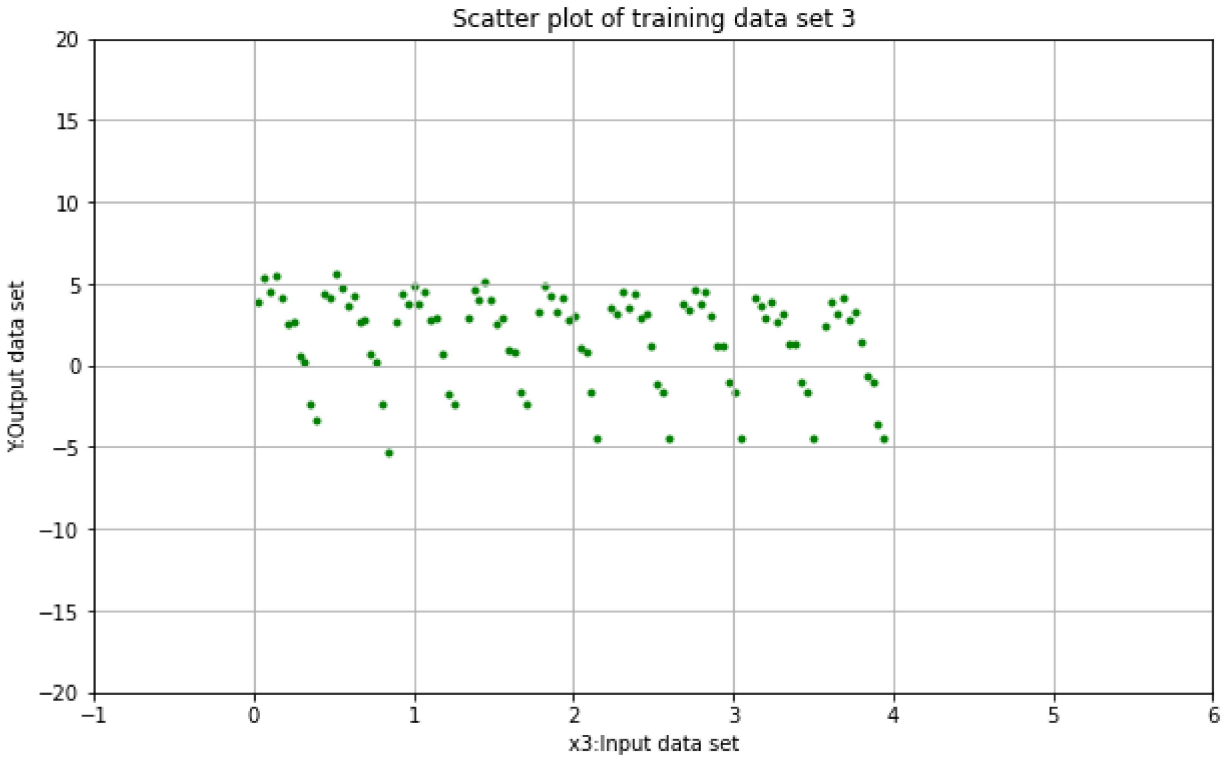
Out[1867]: Text(0.5, 1.0, 'Scatter plot of training data set 2')



In [186…

```
plt.scatter(x3,y, color='green', marker='.')
plt.grid()
plt.xlim([-1,6])
plt.ylim([-20,20])
plt.xlabel('x3:Input data set')
plt.ylabel('Y:Output data set')
plt.title('Scatter plot of training data set 3')
```

Out[1868]: Text(0.5, 1.0, 'Scatter plot of training data set 3')



In [186…

```
x1 = x1.reshape(m,1)
x2 = x2.reshape(m,1)
x3 = x3.reshape(m,1)
print("x1 = ", x1[:10])
print("x2 = ", x2[:10])
print("x3 = ", x3[:10])
```

```
x1 =  [[0.        ]
 [0.04040404]
 [0.08080808]
 [0.12121212]
 [0.16161616]
 [0.2020202 ]
 [0.24242424]
 [0.28282828]
 [0.32323232]
 [0.36363636]]
x2 =  [[3.44      ]
 [0.1349495 ]
 [0.82989899]
 [1.52484848]
 [2.21979798]
 [2.91474747]
 [3.60969697]
 [0.30464646]
 [0.99959596]
 [1.69454546]]
x3 =  [[0.44      ]
 [0.88848485]
 [1.3369697 ]
 [1.78545454]
 [2.23393939]
 [2.68242424]
 [3.13090909]
 [3.57939394]
 [0.02787879]
 [0.47636364]]
```

In [187…

```
X_1 = np.hstack((x0, x1))
X_2 = np.hstack((x0, x2))
X_3 = np.hstack((x0, x3))
print("x1 = ", X_1[:10])
print("x2 = ", X_2[:10])
print("x3 = ", X_3[:10])
```

```
x1 =  [[1.         0.        ]
 [1.         0.04040404]
 [1.         0.08080808]
 [1.         0.12121212]
 [1.         0.16161616]
 [1.         0.2020202 ]
 [1.         0.24242424]
 [1.         0.28282828]
 [1.         0.32323232]
 [1.         0.36363636]]
x2 =  [[1.         3.44      ]
 [1.         0.1349495 ]
 [1.         0.82989899]
 [1.         1.52484848]
 [1.         2.21979798]
 [1.         2.91474747]
 [1.         3.60969697]
 [1.         0.30464646]
 [1.         0.99959596]
 [1.         1.69454546]]
x3 =  [[1.         0.44      ]
 [1.         0.88848485]
 [1.         1.3369697 ]
 [1.         1.78545454]
 [1.         2.23393939]
 [1.         2.68242424]
 [1.         3.13090909]
 [1.         3.57939394]
 [1.         0.02787879]
 [1.         0.47636364]]
```

In [187…
```
theta1 = np.zeros(2)
print(theta1)
```

```
[0. 0.]
```

In [187…
```
def calculateCost(X, Y, theta):
    predictions = X.dot(theta)
    errors = np.subtract(predictions, Y)
    sqrErrors = np.square(errors)
    J = 1 / (2 * m) * np.sum(sqrErrors)
    return J
```

In [187…
```
cost = calculateCost(X_1, y, theta1)
print(cost)
```

```
5.524438459196242
```

In [187…
```
def gradientDescent(X, Y, theta, alpha, iter):
    cost_history = np.zeros(iter)

    for i in range(iter):
        predictions = X.dot(theta)
        errors = np.subtract(predictions, Y)
        sum_delta = (alpha / m) * X.transpose().dot(errors);
        theta = theta - sum_delta;
        cost_history[i] = calculateCost(X, Y, theta)
    return theta, cost_history
```

In [187…
```
theta1 = [0., 0.]
```

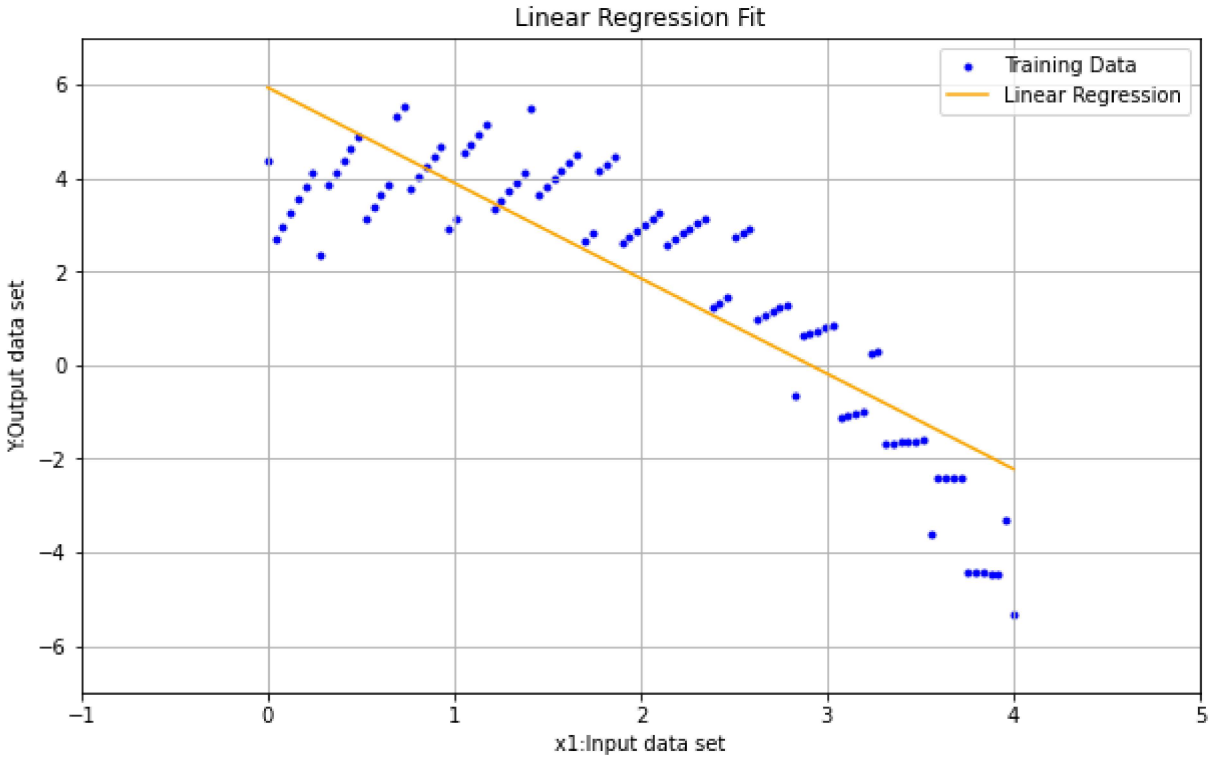
```
iter1 = 2000;
alpha1 = 0.05;
```

In [187…
```
theta1, cost_history = gradientDescent(X_1, y, theta1, alpha1, iter1)
print('Final value of theta1 =', theta1)
print('cost_history =', cost_history)
```

```
Final value of theta1 = [ 5.92794892 -2.03833663]
cost_history = [5.32852962 5.18676104 5.07204859 ... 0.98499308 0.98499308 0.9849930
8]
```

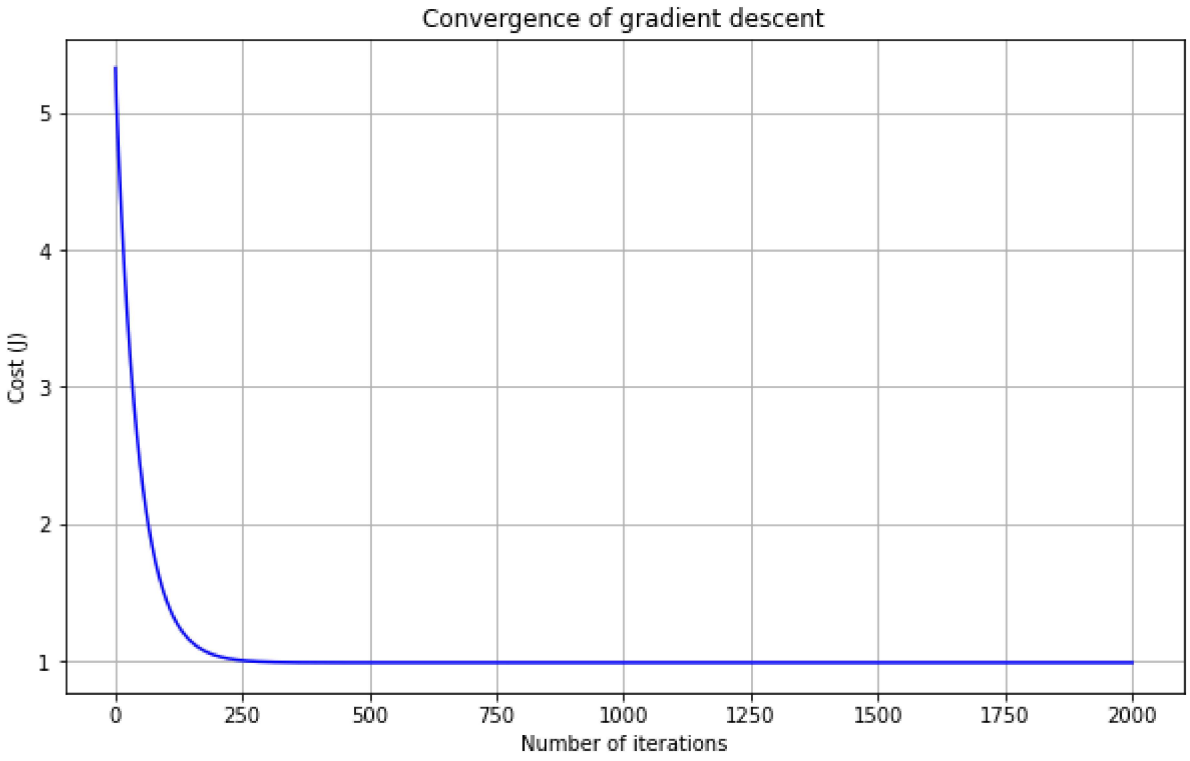
In [187…
```
plt.scatter(X_1[:,1], y, color='blue', marker= '.', label= 'Training Data')
plt.plot(X_1[:,1],X_1.dot(theta1), color='orange', label='Linear Regression')
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.xlim([-1,5])
plt.ylim([-7,7])
plt.xlabel('x1:Input data set')
plt.ylabel('Y:Output data set')
plt.title('Linear Regression Fit')
plt.legend()
```

Out[1877]:
```
<matplotlib.legend.Legend at 0x19e2c028520>
```



In [187…
```
plt.plot(range(1, iter1 + 1),cost_history, color='blue')
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Convergence of gradient descent')
```

Out[1878]:
```
Text(0.5, 1.0, 'Convergence of gradient descent')
```



In [187…
```
theta2 = np.zeros(2)
print(theta2)
```

```
[0. 0.]
```

In [188…
```
cost = calculateCost(X_2, y, theta2)
print(cost)
```

```
5.524438459196242
```

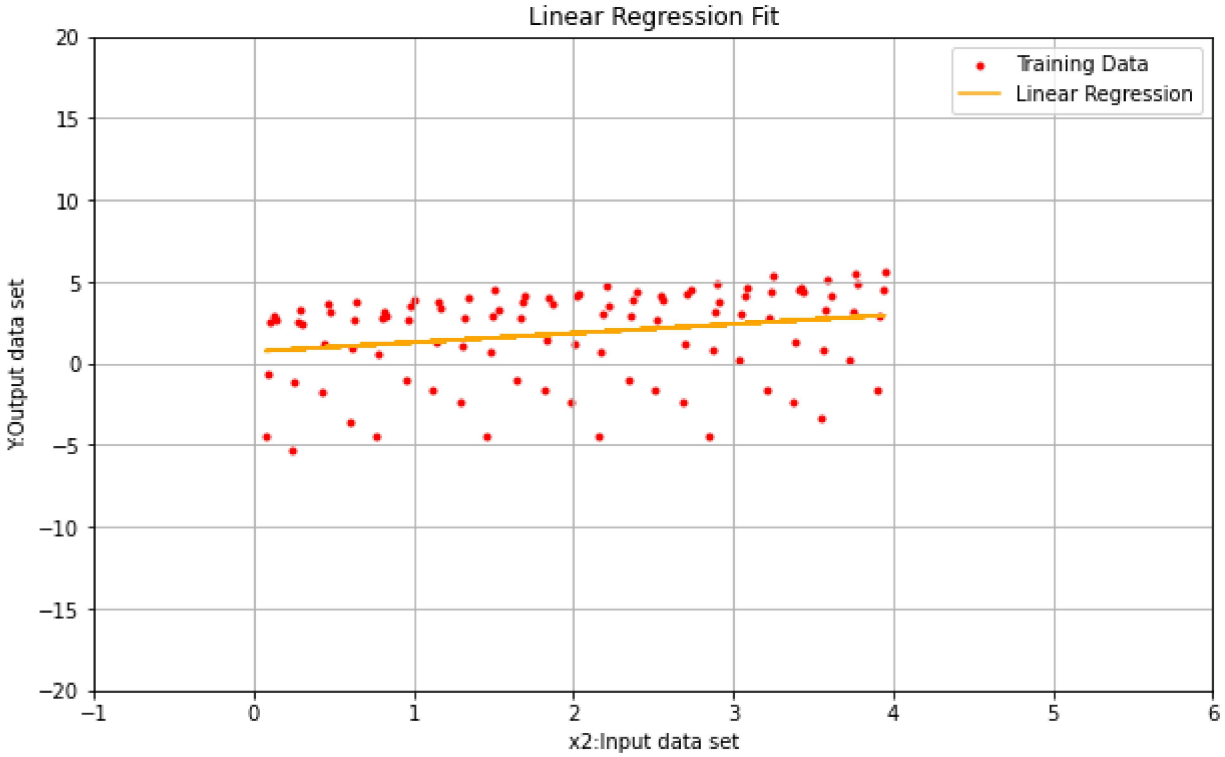
In [188…
```
theta2 = [0., 0.]
iter2 = 2000;
alpha2 = 0.01;
```

In [188…
```
theta2, cost_history = gradientDescent(X_2, y, theta2, alpha2, iter2)
print('Final value of theta2 =', theta2)
print('cost_history =', cost_history)
```

```
Final value of theta2 = [0.73072498 0.55968422]
cost_history = [5.29831663 5.09909109 4.92356115 ... 3.59936968 3.59936967 3.5993696
5]
```

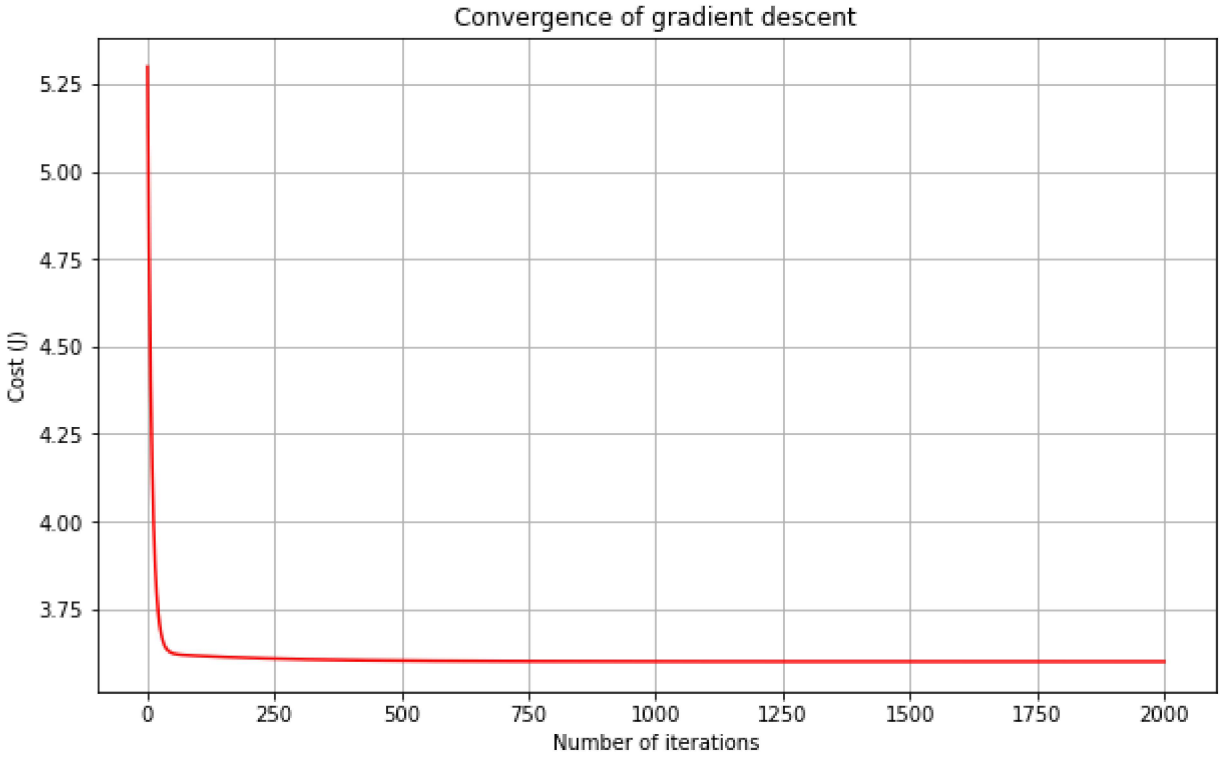
In [188…
```
plt.scatter(X_2[:,1], y, color='red', marker= '.', label= 'Training Data')
plt.plot(X_2[:,1],X_2.dot(theta2), color='orange', label='Linear Regression')
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.xlim([-1,6])
plt.ylim([-20,20])
plt.xlabel('x2:Input data set')
plt.ylabel('Y:Output data set')
plt.title('Linear Regression Fit')
plt.legend()
```

Out[1883]:
```
<matplotlib.legend.Legend at 0x19e2c0eb880>
```



In [188…
```
plt.plot(range(1, iter2 + 1),cost_history, color='red')
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Convergence of gradient descent')
```

Out[1884]:
```
Text(0.5, 1.0, 'Convergence of gradient descent')
```



In [188…
```
theta3 = np.zeros(2)
```

```
print(theta3)
```

```
[0. 0.]
```

In [188…
```
cost = calculateCost(X_3, y, theta3)
print(cost)
```

```
5.524438459196242
```

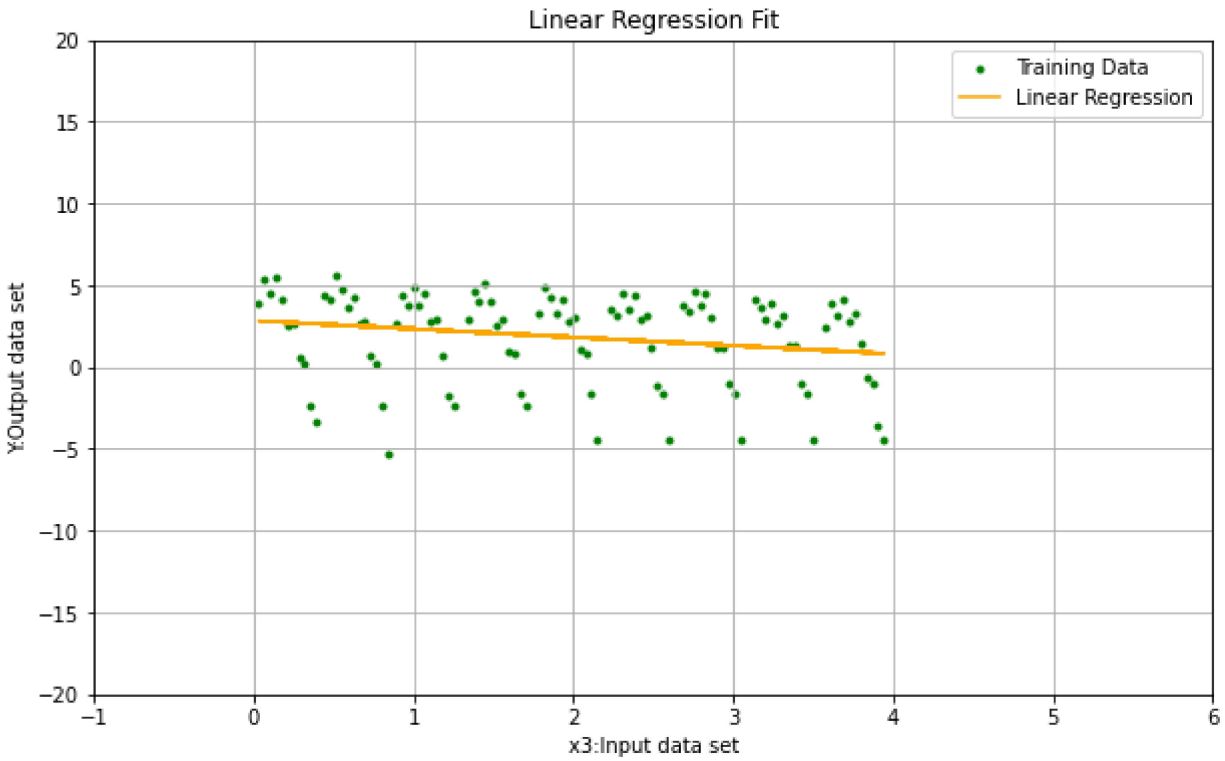
In [188…
```
theta3 = [0., 0.]
iter3 = 2000;
alpha3 = 0.01;
```

In [188…
```
theta3, cost_history = gradientDescent(X_3, y, theta3, alpha3, iter3)
print('Final value of theta3 =', theta3)
print('cost_history =', cost_history)
```

```
Final value of theta3 = [ 2.84191038 -0.50881117]
cost_history = [5.40768785 5.30397076 5.21178297 ... 3.62956537 3.62956486 3.6295643
5]
```

In [188…
```
plt.scatter(X_3[:,1], y, color='green', marker= '.', label= 'Training Data')
plt.plot(X_3[:,1],X_3.dot(theta3), color='orange', label='Linear Regression')
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.xlim([-1,6])
plt.ylim([-20,20])
plt.xlabel('x3:Input data set')
plt.ylabel('Y:Output data set')
plt.title('Linear Regression Fit')
plt.legend()
```

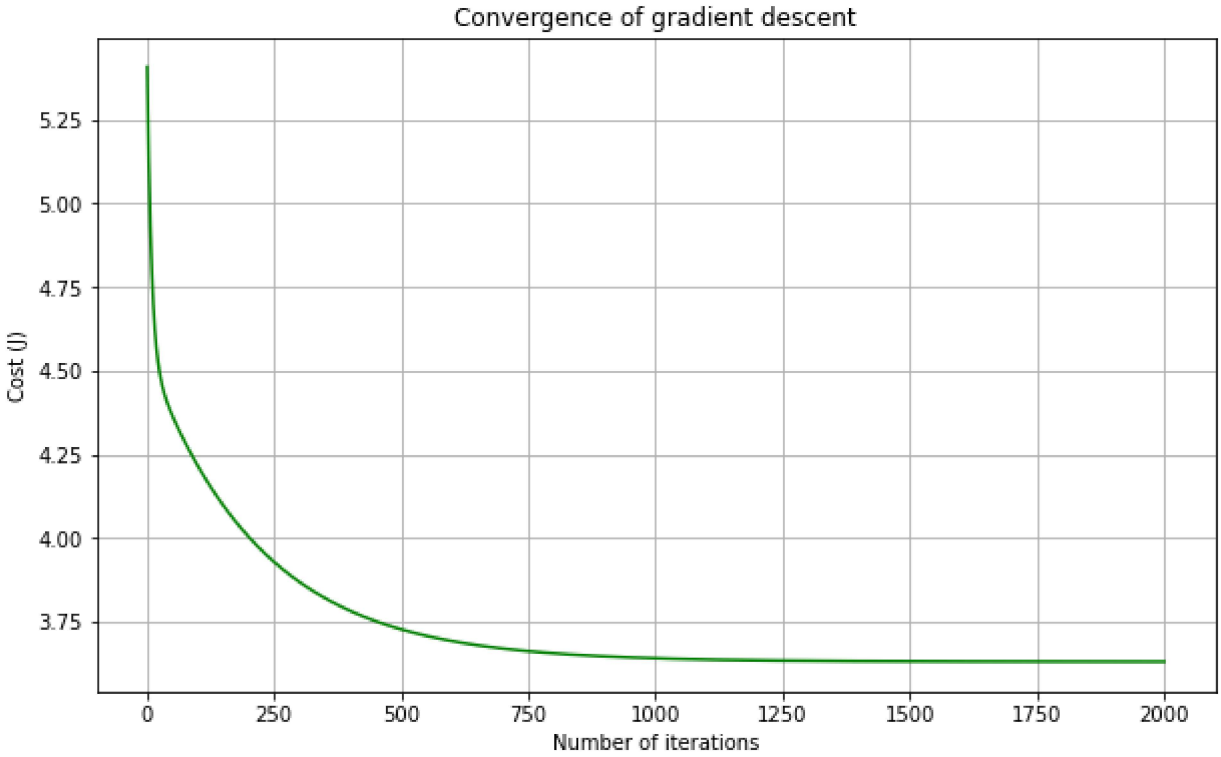
Out[1889]:
```
<matplotlib.legend.Legend at 0x19e2d1abca0>
```



In [189…
```
plt.plot(range(1, iter3 + 1),cost_history, color='green')
plt.rcParams["figure.figsize"] = (10,6)
```

```
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Convergence of gradient descent')
```

Out[1890]: Text(0.5, 1.0, 'Convergence of gradient descent')



In [ ]:

In [103…
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [103…
```
dataset = pd.read_csv('C:\\Users\\pha-a\\OneDrive\\Desktop\\4105\\D3.csv')
print(dataset)
dataset.head()
m = len(dataset)
print(m)
```

```
          X1        X2        X3         Y
0   0.000000  3.440000  0.440000  4.387545
1   0.040404  0.134949  0.888485  2.679650
2   0.080808  0.829899  1.336970  2.968490
3   0.121212  1.524848  1.785455  3.254065
4   0.161616  2.219798  2.233939  3.536375
..       ...       ...       ...       ...
95  3.838384  1.460202  3.046061 -4.440595
96  3.878788  2.155152  3.494545 -4.458663
97  3.919192  2.850101  3.943030 -4.479995
98  3.959596  3.545051  0.391515 -3.304593
99  4.000000  0.240000  0.840000 -5.332455

[100 rows x 4 columns]
100
```

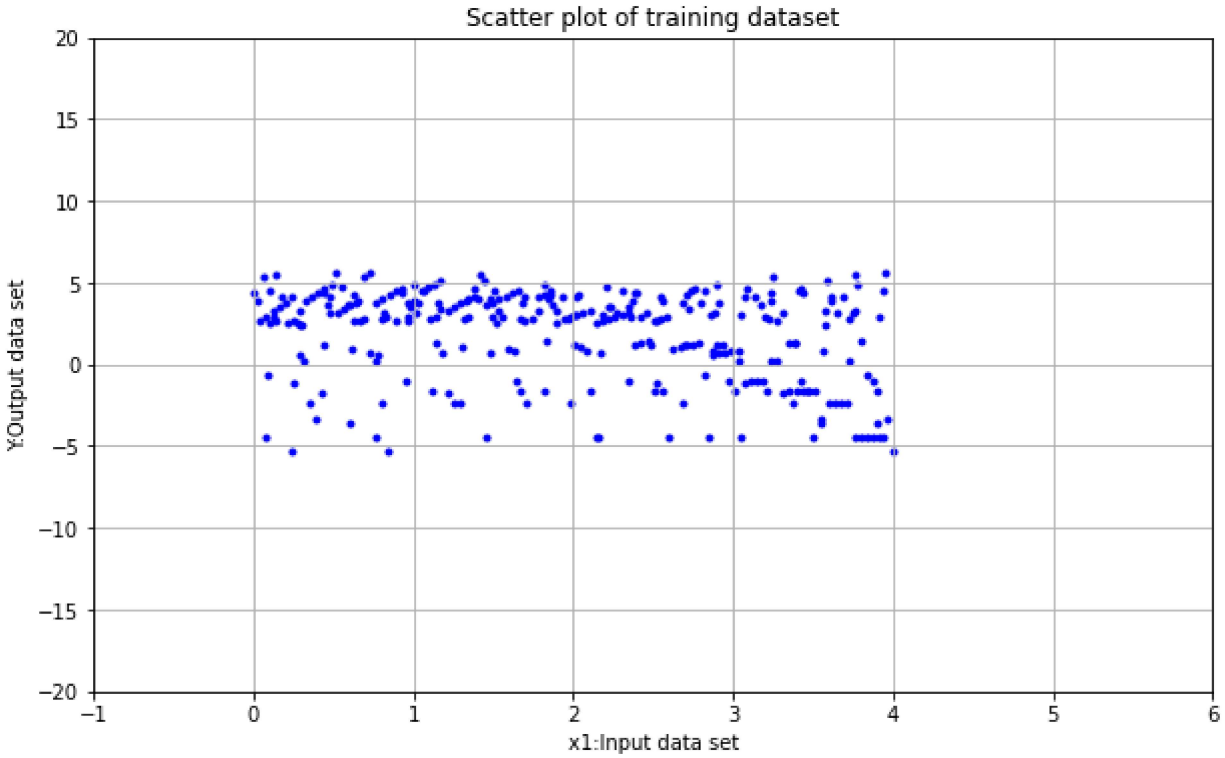
In [103…
```
x0 = np.ones((m, 1))
X = dataset.values[:,(0,1,2)]
Y = dataset.values[:,3]

print("x0 ", x0[:10])
print("X ", X[:10])
```

```
x0  [[1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]
 [1.]]
X  [[0.         3.44       0.44      ]
 [0.04040404 0.1349495  0.88848485]
 [0.08080808 0.82989899 1.3369697 ]
 [0.12121212 1.52484848 1.78545454]
 [0.16161616 2.21979798 2.23393939]
 [0.2020202  2.91474747 2.68242424]
 [0.24242424 3.60969697 3.13090909]
 [0.28282828 0.30464646 3.57939394]
 [0.32323232 0.99959596 0.02787879]
 [0.36363636 1.69454546 0.47636364]]
```

In [103…
```
for i in range (0,3):
    plt.scatter(X[:,i],Y, color='blue', marker='.')
plt.grid()
plt.xlim([-1,6])
```

```
plt.ylim([-20,20])
plt.xlabel('x1:Input data set')
plt.ylabel('Y:Output data set')
plt.title('Scatter plot of training dataset')
```

Out[1036]: Text(0.5, 1.0, 'Scatter plot of training dataset')



In [103…
```
X = np.hstack((x0, X))
print(X[:10])
```

```
[[1.         0.         3.44       0.44      ]
 [1.         0.04040404 0.1349495  0.88848485]
 [1.         0.08080808 0.82989899 1.3369697 ]
 [1.         0.12121212 1.52484848 1.78545454]
 [1.         0.16161616 2.21979798 2.23393939]
 [1.         0.2020202  2.91474747 2.68242424]
 [1.         0.24242424 3.60969697 3.13090909]
 [1.         0.28282828 0.30464646 3.57939394]
 [1.         0.32323232 0.99959596 0.02787879]
 [1.         0.36363636 1.69454546 0.47636364]]
```

In [103…
```
theta = np.zeros(4)
print(theta)
```

```
[0. 0. 0. 0.]
```

In [103…
```
def calculateCost(X, Y, theta):
    predictions = X.dot(theta)
    errors = np.subtract(predictions, Y)
    sqrErrors = np.square(errors)
    J = 1 / (2 * m) * np.sum(sqrErrors)
    return J
```

In [104…
```
def gradientDescent(X, Y, theta, alpha, iter):
    cost_history = np.zeros(iter)
```

```
    for i in range(iter):
        predictions = X.dot(theta)
        errors = np.subtract(predictions, Y)
        sum_delta = (alpha / m) * X.transpose().dot(errors);
        theta = theta - sum_delta;
        cost_history[i] = calculateCost(X, Y, theta)

    print(theta)
    return theta, cost_history
```

In [104…

```
cost = calculateCost(X, Y, theta)
print(cost)
```

```
5.524438459196242
```

In [104…

```
#theta = [0., 0., 0., 0.]
iter = 2000;
alpha = 0.1;

theta, cost_history = gradientDescent(X, Y, theta, alpha, iter)
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

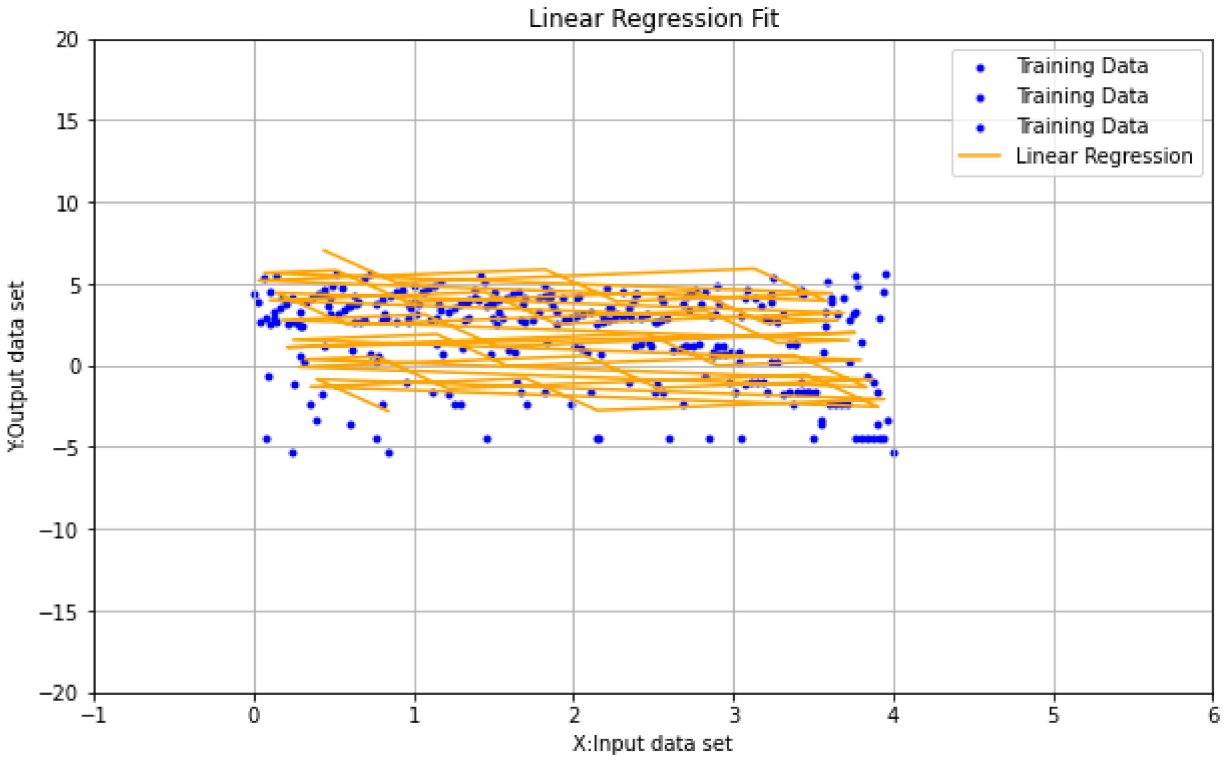
```
[ 5.31416716 -2.00371927  0.53256334 -0.26560186]
Final value of theta = [ 5.31416716 -2.00371927  0.53256334 -0.26560186]
cost_history = [4.13064348 3.51770697 3.12758306 ... 0.73846424 0.73846424 0.7384642
4]
```

In [104…

```
for i in range(1,4):
    plt.scatter(X[:,i], Y, color='blue', marker= '.', label= 'Training Data')
plt.plot(X[:,3],X.dot(theta), color='orange', label='Linear Regression')
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.xlim([-1,6])
plt.ylim([-20,20])
plt.xlabel('X:Input data set')
plt.ylabel('Y:Output data set')
plt.title('Linear Regression Fit')
plt.legend()
```

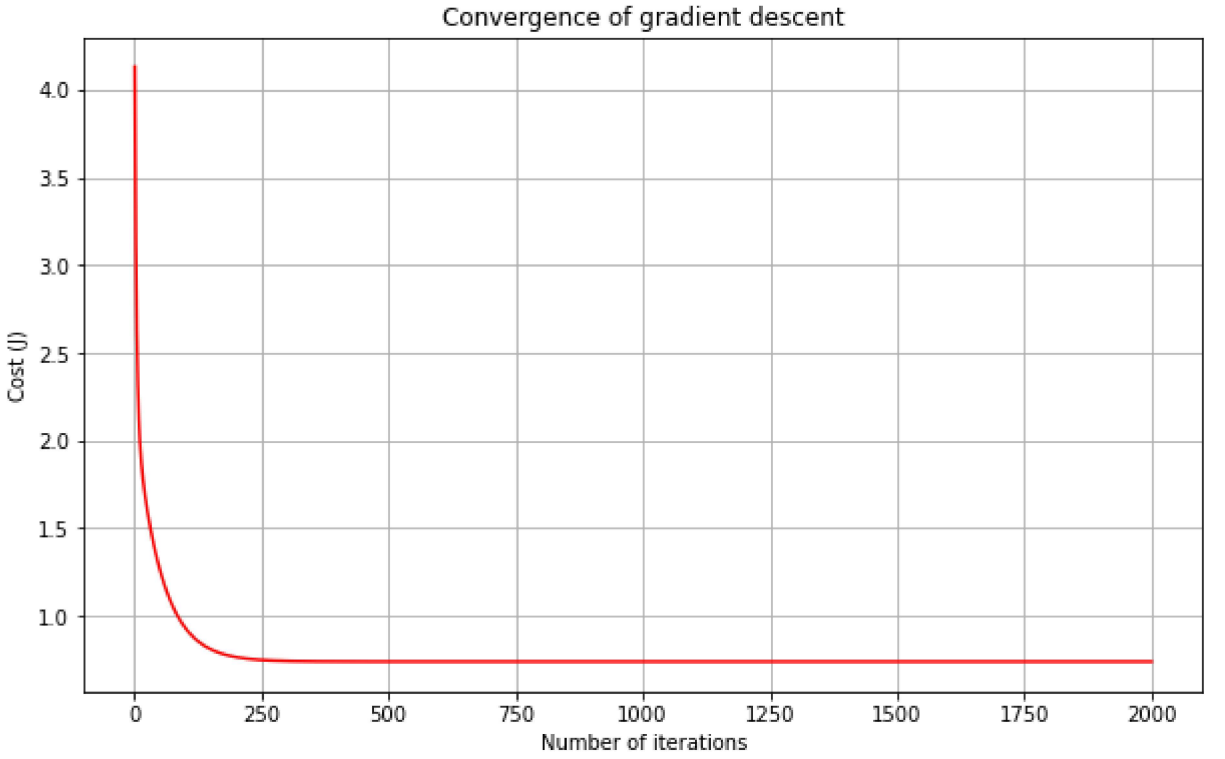
Out[1043]:

```
<matplotlib.legend.Legend at 0x292ffd89ca0>
```



In [104…
```
plt.plot(range(1, iter + 1),cost_history, color='red')
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Convergence of gradient descent')
```

Out[1044]:
```
Text(0.5, 1.0, 'Convergence of gradient descent')
```



In [104…
```
xInput =[1,1,1]
```

```
yPredict = theta[1]*xInput[0] + theta[2]*xInput[1] + theta[3]*xInput[2]
print(yPredict)
```

```
-1.7367577934853458
```

In [104…
```
xInput =[2,0,4]
yPredict = theta[1]*xInput[0] + theta[2]*xInput[1] + theta[3]*xInput[2]
print(yPredict)
```

```
-5.069845990658848
```

In [104…
```
xInput =[3,2,1]
yPredict = theta[1]*xInput[0] + theta[2]*xInput[1] + theta[3]*xInput[2]
print(yPredict)
```

```
-5.211632990272375
```

In [ ]:
```

```