Information

$Writer : Junhyuck Woo$
$Std.ID : 20145337$
$Assignment02 : Linear Regression$
$Deadline : Apr 2, 2020$

Library

In [0]:
```
import matplotlib.pyplot as plt; import numpy as np
```

Data

Function

$m = 50$
$\hat{y} = 4x + 50$
$y = \hat{y} + n, n is N(0, \sigma^2)$

In [0]:
```
m = 1000
x = np.arange(-150, 150, 0.1)
y_ = 4*x + 5
n = np.int64(np.random.normal(loc=0, scale=100, size=m))
m_list = np.int64(np.random.normal(loc=0, scale=30, size=m))
y = 4*m_list+5 + n
```

Linear Model

$h_\theta(x) = \theta_0 + \theta_1 x$
$\theta_0 = 0$
$\theta_1 = 1$

In [0]:
```
theta0 = 0; theta0_old = 0
theta1 = 1; theta1_old = 0
theta0_history = [theta0]
theta1_history = [theta1]
h = theta0 + theta1*m_list
```

Objective Function

$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$

In [0]:
```
j = np.sum((h - y)**2) / (2*m)
j_old = 0
```

```
j_history = [j]
```

Gradient Descent

$\theta_0^{(t+1)} := \theta_0^{(t)} - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$
$\theta_1^{(t+1)} := \theta_1^{(t)} - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$
$\alpha = 0.1$

In [0]:
```
alpha = 0.001
```

Training Process

In [0]:
```
# Check the number of iteration
iteration = 1

while  (abs(j - j_old) + abs(theta0 - theta0_old) + abs(theta1 - theta1_old)) != 0:
    # Calculate the theta
    theta0_old = theta0
    theta1_old = theta1
    theta0 = theta0 - alpha*np.sum(h-y)/m
    theta1 = theta1 - alpha*np.sum((h-y)*m_list)/m

    # Update the j, h
    j_old = j
    h = theta0 + theta1*m_list
    j = np.sum((h - y)**2) / (2*m)

    # Record the history of parameter
    theta0_history.append(theta0)
    theta1_history.append(theta1)
    j_history.append(j)

    iteration = iteration +1
```

Check the Convergence

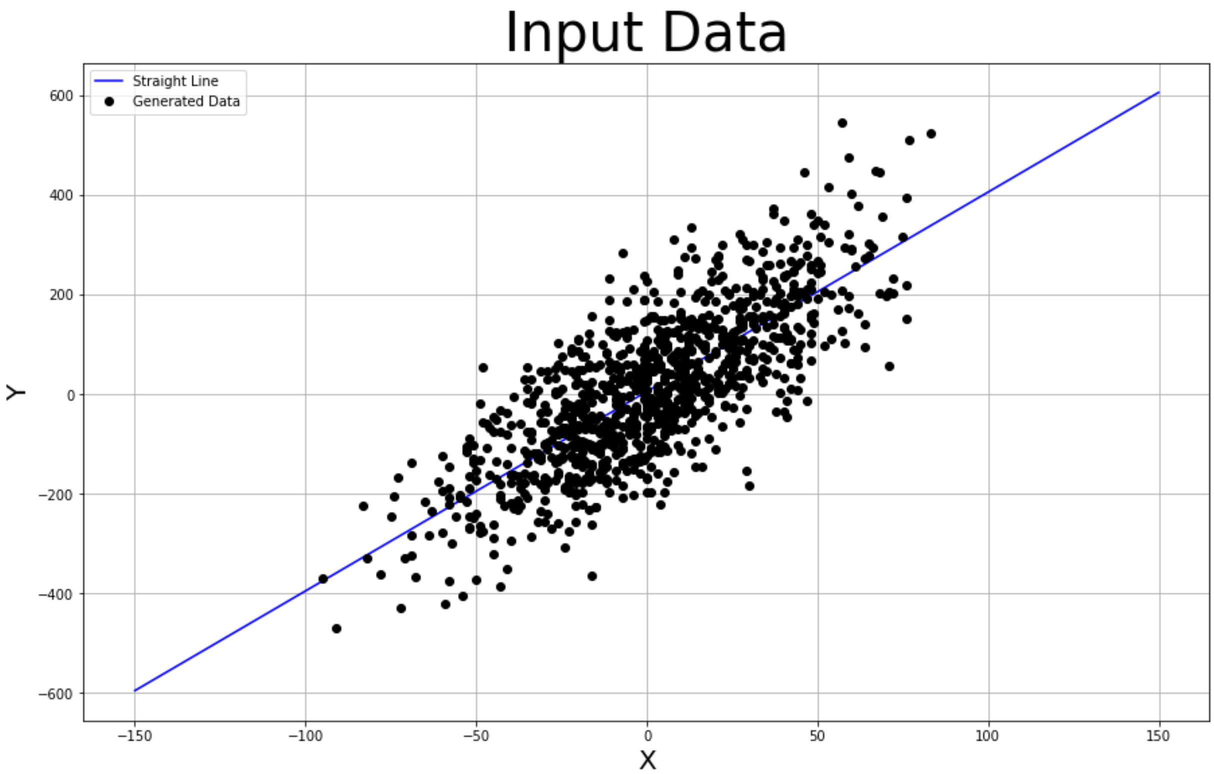
In [7]:
```
# Iteration
print("# Iteration: " + str(iteration) + '\n')

# Theta 0
print("Updated Theta0: "+ str(theta0))
print("Old Theta0: " + str(theta0_old))
print("Diff: " + str(theta0 - theta0_old) + '\n')

# Theta 1
print("Updated Theta1: "+ str(theta1))
print("Old Theta1: " + str(theta1_old))
print("Diff: " + str(theta1 - theta1_old) + '\n')

# J, Energy Value
print("Updated J: "+ str(j))
print("Old J: " + str(j_old))
print("Diff: " + str(j - j_old) + '\n')
```

```
# Iteration: 30372

Updated Theta0: 7.278621357367933
Old Theta0: 7.278621357367933
Diff: 0.0

Updated Theta1: 3.96233898211217
Old Theta1: 3.96233898211217
Diff: 0.0

Updated J: 4729.6843818666475
Old J: 4729.6843818666475
Diff: 0.0
```

1. Input Data

In [8]:
```
plt.figure(figsize=(15,9))
plt.plot(x, y_, color='blue', label='Straight Line')
plt.plot(m_list, y, 'ro', color='black', label='Generated Data')
plt.title('Input Data', fontsize=40)
plt.grid()
plt.legend()
plt.xlabel('X', fontsize=20)
plt.ylabel('Y', fontsize=20)
```

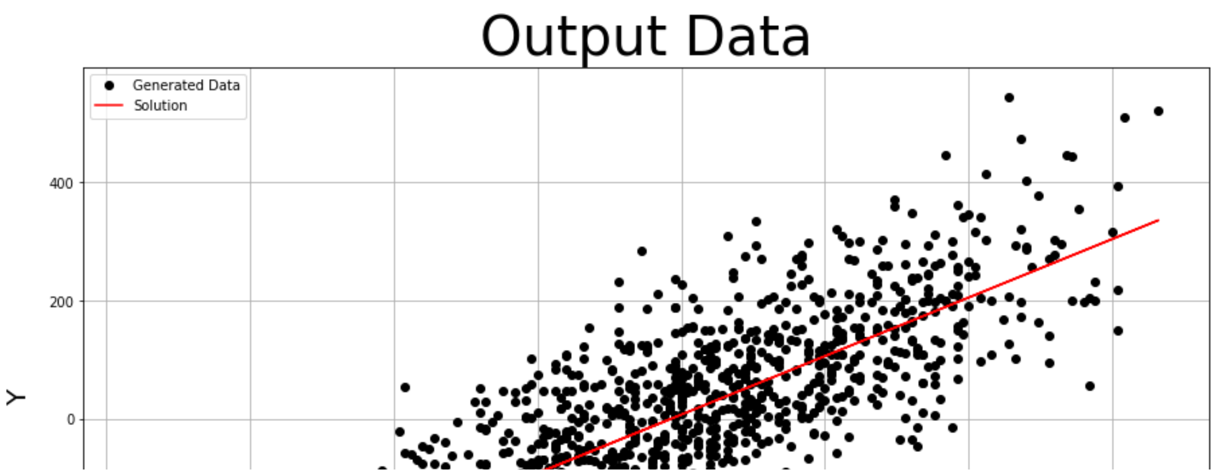
Out[8]: Text(0, 0.5, 'Y')

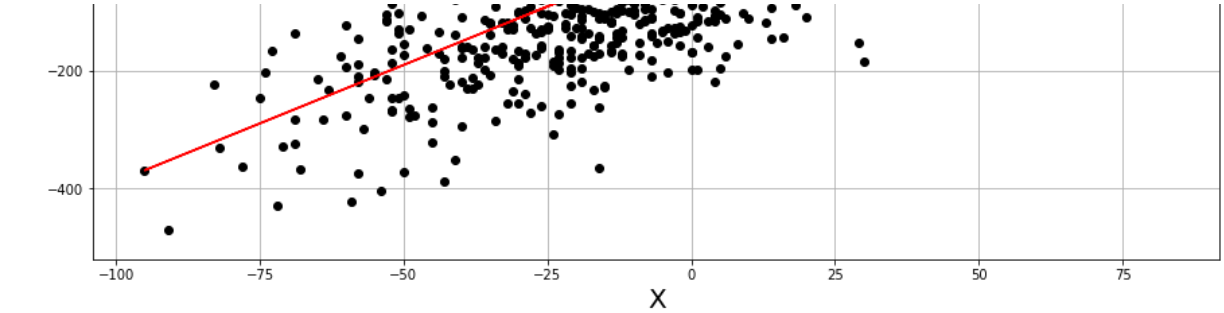


2. Output Result

In [9]:
```
plt.figure(figsize=(15,9))
plt.plot(m_list, y, 'ro', color='black', label='Generated Data')
plt.plot(m_list, h, color='red', label='Solution')
plt.title('Output Data', fontsize=40)
plt.grid()
plt.legend()
plt.xlabel('X', fontsize=20)
plt.ylabel('Y', fontsize=20)
```

Out[9]: Text(0, 0.5, 'Y')

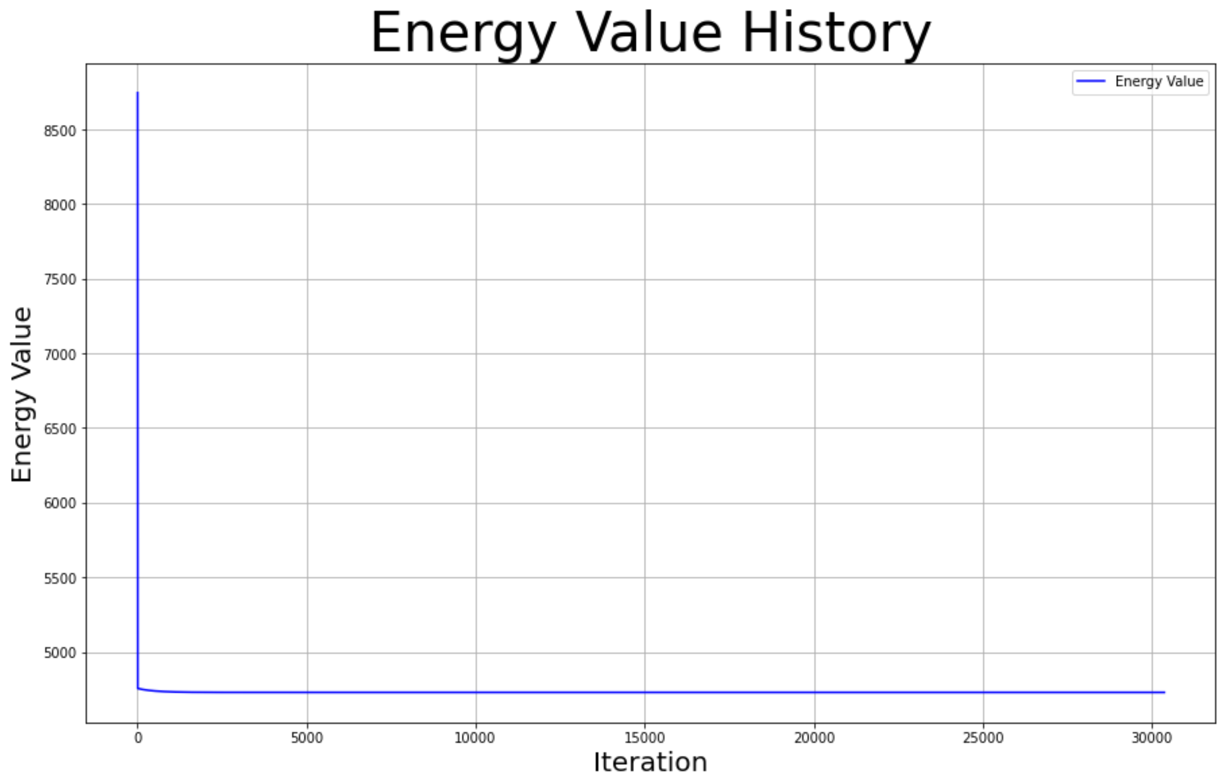




3. Energy Value

In [10]:
```
plt.figure(figsize=(15,9))
plt.plot(j_history, color='blue', label='Energy Value')
plt.title('Energy Value History', fontsize=40)
plt.grid()
plt.legend()
plt.xlabel('Iteration', fontsize=20)
plt.ylabel('Energy Value', fontsize=20)
```

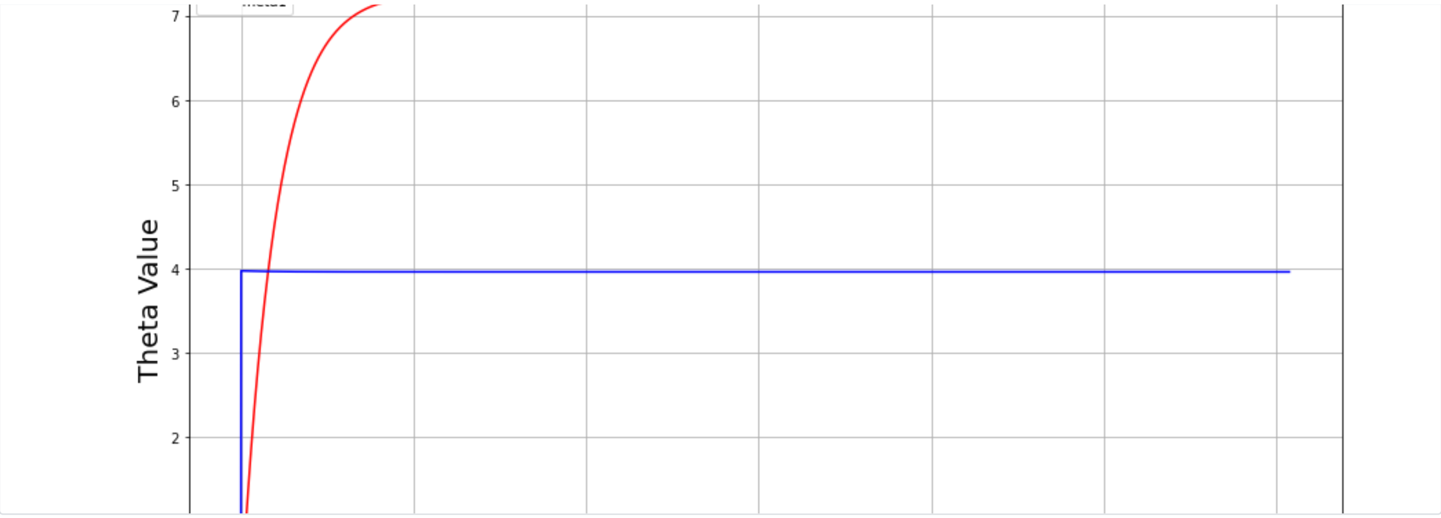
Out[10]: `Text(0, 0.5, 'Energy Value')`



4. Model Parameters

In [11]:
```
plt.figure(figsize=(15,9))
plt.plot(theta0_history, color='red', label='Theta0')
plt.plot(theta1_history, color='blue', label='Theta1')
plt.title('Model Parameter History', fontsize=40)
plt.grid()
plt.legend()
plt.xlabel('Iteration', fontsize=20)
plt.ylabel('Theta Value', fontsize=20)
```

Out[11]: `Text(0, 0.5, 'Theta Value')`


Theta Value
7
6
5
4
3
2