Karan Choudhary
CSE_DS
Rollno:07

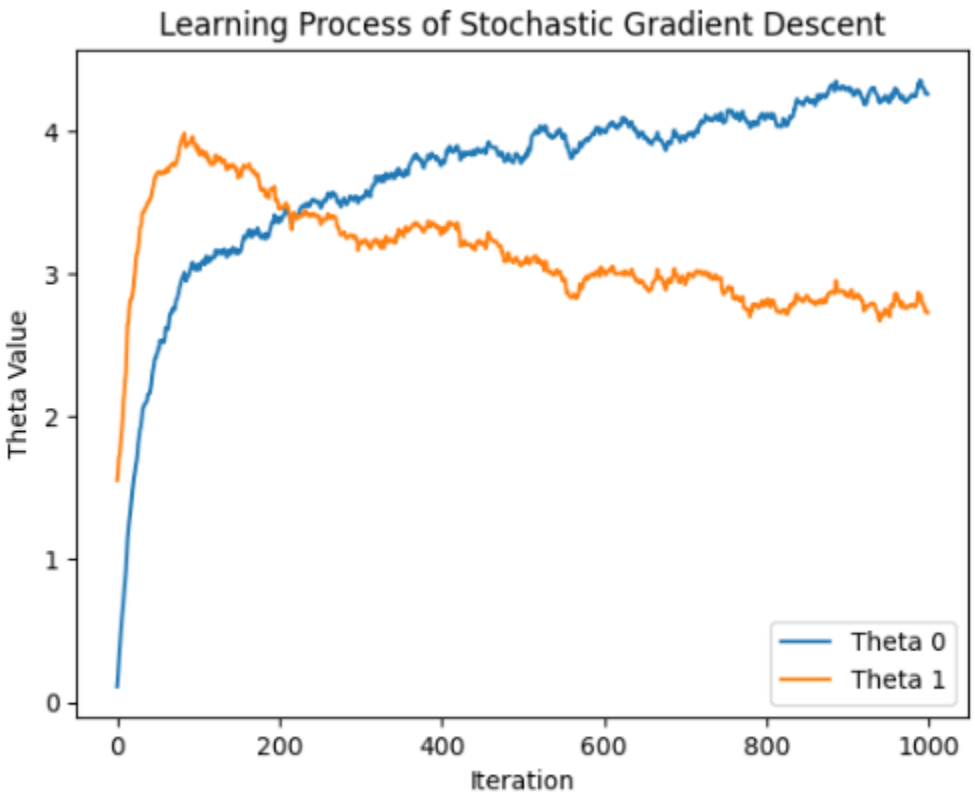
Deep Learning
Experiment No: 04

Stochastic gradient descent

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(42)
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1)

theta = np.random.randn(2, 1)

learning_rate = 0.01
n_iterations = 1000

theta_history = []

for iteration in range(n_iterations):
    random_index = np.random.randint(len(X))
    xi = np.hstack((np.ones((1, 1)), X[random_index:random_index+1]))
    yi = y[random_index:random_index+1]

    gradient = 2 * xi.T.dot(xi.dot(theta) - yi)
    theta = theta - learning_rate * gradient

    theta_history.append(theta.copy())


theta_history = np.array(theta_history).reshape(-1, 2)

plt.plot(theta_history[:, 0], label='Theta 0')
plt.plot(theta_history[:, 1], label='Theta 1')
plt.xlabel('Iteration')
```

```
plt.ylabel('Theta Value')
plt.title('Learning Process of Stochastic Gradient Descent')
plt.legend()
plt.show()

print("Final theta:", theta)
```

Output:



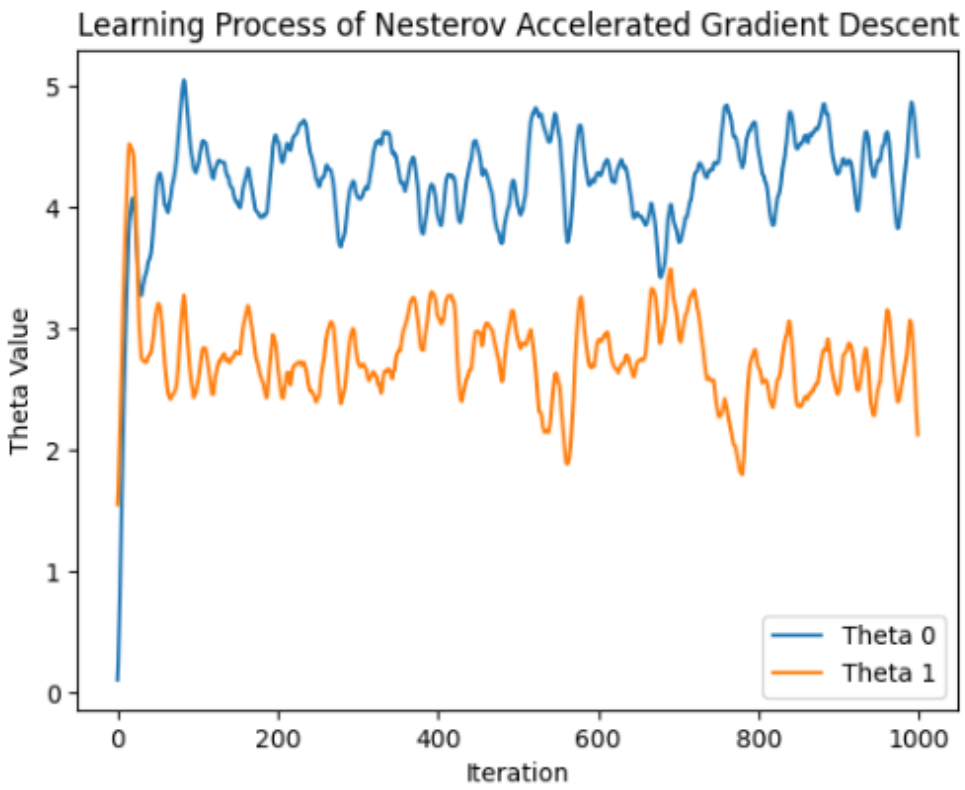

```
Final theta: [[4.25573545]
 [2.72608246]]
```

Nestorev Gradient Descent

```
import numpy as np
import matplotlib.pyplot as plt



np.random.seed(42)
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1)

theta = np.random.randn(2, 1)

learning_rate = 0.01
gamma = 0.9
n_iterations = 1000

momentum = np.zeros_like(theta)

theta_history = []

for iteration in range(n_iterations):
    random_index = np.random.randint(len(X))

    xi = np.hstack((np.ones((1, 1)), X[random_index:random_index+1]))  #
    yi = y[random_index:random_index+1]

    gradient = 2 * xi.T.dot(xi.dot(theta) - yi)

    momentum = gamma * momentum + learning_rate * gradient
    theta = theta - momentum

    theta_history.append(theta.copy())

theta_history = np.array(theta_history).reshape(-1, 2)

plt.plot(theta_history[:, 0], label='Theta 0')
plt.plot(theta_history[:, 1], label='Theta 1')
plt.xlabel('Iteration')
```

```
plt.ylabel('Theta Value')
plt.title('Learning Process of Nesterov Accelerated Gradient Descent')
plt.legend()
plt.show()

print("Final theta:", theta)
```

Output:




```
Final theta: [[4.41954999]
 [2.12700409]]
```