Machine Learning Lab – 20ISL68A

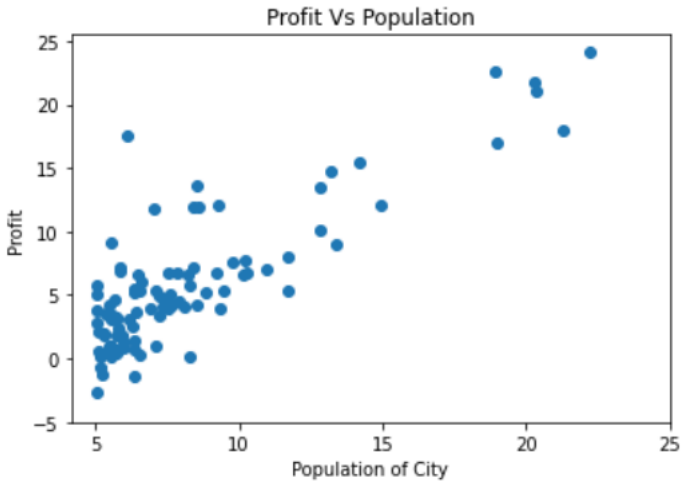
Program 5 - Develop a program to demonstrate the working of the Gradient Descent algorithm. Use an appropriate data set for building the model and apply this knowledge to predict a value for a test case.

Step 1: Importing Libraries and loading dataset

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
data=pd.read_csv(r"C:\Users\kvsuv\OneDrive\Desktop\Uni_linear.txt", header=None)
```

Step 2: Plotting Scatter Graph

```
plt.scatter(data[0],data[1])
plt.xlabel("Population of City")
plt.ylabel("Profit")
plt.title("Profit Vs Population")
```



Step 3: Function for Cost

```
def computeCost(X,y,theta):
    m=len(y)
    predictions=X.dot(theta)
    square_err=(predictions - y)**2
    return 1/(2*m) * np.sum(square_err)
```

Step 4: Calculating theta values

```
data=data.values
m=len(data[:,-1])
X=np.append(np.ones((m,1)),data[:,0].reshape(m,1),axis=1)
y=data[:,1].reshape(m,1)
theta=np.zeros((2,1))
computeCost(X,y,theta)
```

Step 5: Function for Gradient Descent

```
def gradientDescent(X,y,theta,alpha,num_iters):
    m=len(y)
    J_history=[]
    for i in range(num_iters):
        predictions = X.dot(theta)
        error = np.dot(X.transpose(),(predictions -y))
        descent=alpha * 1/m * error
        theta-=descent
        J_history.append(computeCost(X,y,theta))

    return theta, J_history
theta,J_history = gradientDescent(X,y,theta,0.01,1500)
```

Step 6: Plotting Cost function using Gradient Descent

```
plt.plot(J_history)
plt.title("Cost function using Gradient Descent")
```