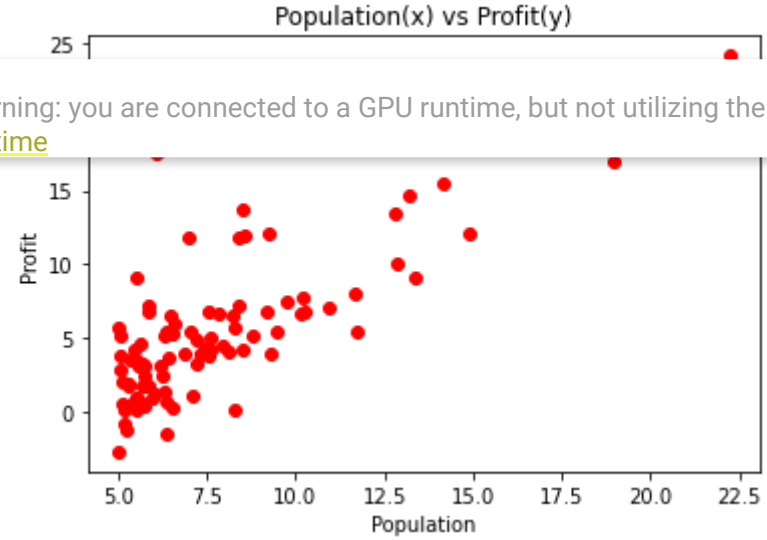
- name - Pankaj Singh Napalchyal
- roll no - 2018IMT-062
- Course: Machine Learning Lab
- Course Code: ITIT-4107-2021
- Deadline: 18 Oct 2021

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

+ Code + Text

```
data_n= pd.read_csv("https://raw.githubusercontent.com/AlfTang/Linear-Regression/master/ex
X=data_n[:,0]
y=data_n[:,1]


plt.title('Population(x) vs Profit(y)')
plt.xlabel('Population')
plt.ylabel('Profit')
plt.scatter(X,y,color='red')
plt.show()
```




```
def computeCost(X,y,h):
    return np.sum((y-h)*(y-h))/(2*len(X))


def gradientDescent(X,y,num_iters,alpha):
    theta0 = 0
    theta1 = 0
    for i in range(1, num_iters):
        h=(theta0*X) + theta1
        theta0-= alpha * ((-1.0/len(X)) * (np.sum(X*(y-h))))
        theta1-= alpha * ((-1.0/len(X))* (np.sum(y-h)))

    return theta0, theta1
```

```
theta0=0
theta1=0
h = (theta0*X) + theta1
print("Cost value(MSE) for 0 iterations = " + str(computeCost(X, y, h)))
```

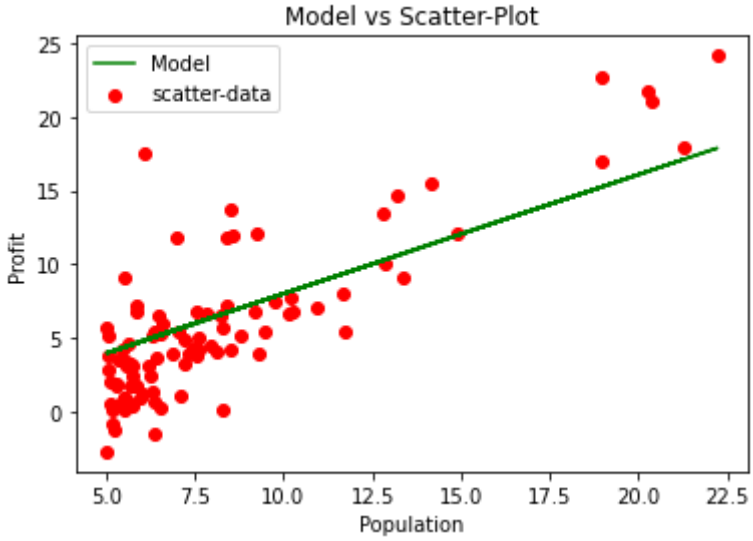
```
 Cost value(MSE) for 0 iterations = 32.072733877455676
```

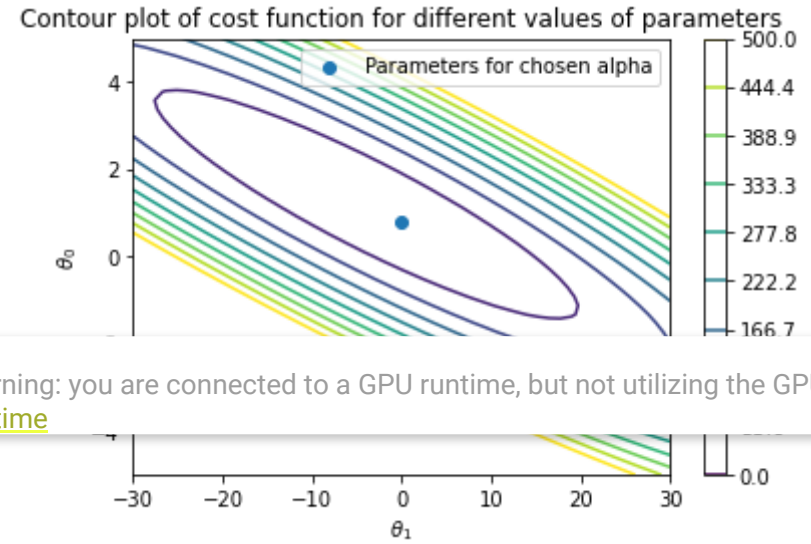
```
def plot_model_fit(X,y,theta0, theta1):
    h = (theta0*X) + theta1
    plt.title('Model vs Scatter-Plot')
    plt.xlabel('Population')
    plt.ylabel('Profit')
    plt.scatter(X,y,color='red',label='scatter-data')
    plt.plot(X,h,color='green',label='Model')
    plt.legend(loc='best')
    plt.show()

    plt.xlabel('$\\theta_{1}$');
    plt.ylabel("$\\theta_{0}$")
    plt.title("Contour plot of cost function for different values of parameters");
    plt.scatter(theta1, theta0,label='Parameters for chosen alpha')
    plt.colorbar(plt.contour(theta1_vals, theta0_vals, costFunction.T,levels=np.linspace(0
    plt.legend(loc='best')
    plt.show()
```



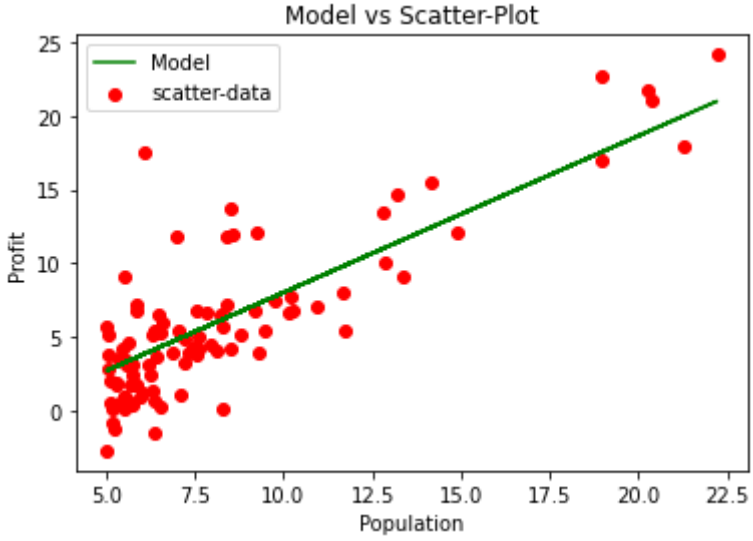
```
costFunction = np.zeros((theta1_vals.shape[0], theta0_vals.shape[0]))

for i, teta1 in enumerate(theta1_vals):
    for j, teta0 in enumerate(theta0_vals):
        h = (teta0 * X) + teta1
        costFunction[i, j] = computeCost(X, y, h)


alpha_values = np.arange(0.0001, 0.021, 0.003)

for i in range(len(alpha_values)):
    alpha = alpha_values[i]

    theta0, theta1 = gradientDescent(X,y,2000,alpha)
    h = (theta0*X) + theta1
    print(str(i + 1) + '. ALPHA value,i.e. learning_rate: {}'.format(alpha))
    print('Hypothesis Function: h(x) = ' + str(theta0) + "x + "+ str(theta1))
    print('Cost Value(Mean Square Error) = ' + str(computeCost(X, y, h)))
    plot_model_fit(X, y,theta0, theta1)
```

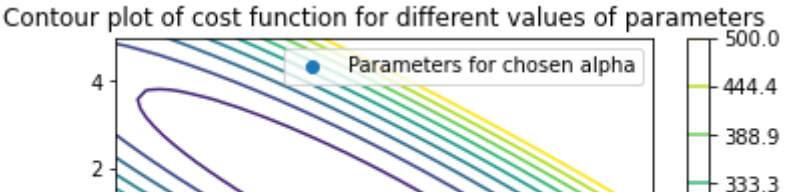
```
1. ALPHA value,i.e. learning_rate: 0.0001
Hypothesis Function: h(x) = 0.8077855783900526x + -0.060973353535627335
Cost Value(Mean Square Error) = 5.815774148036794
```

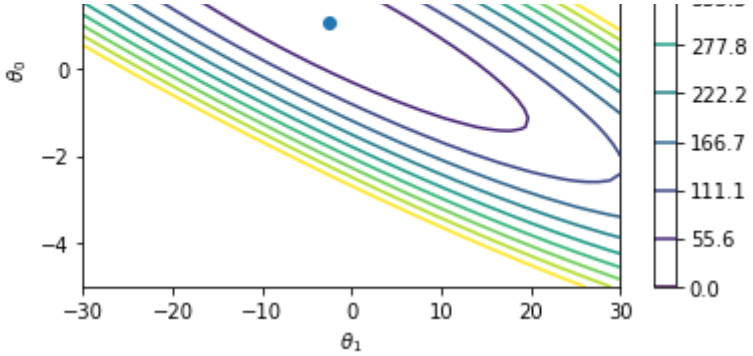




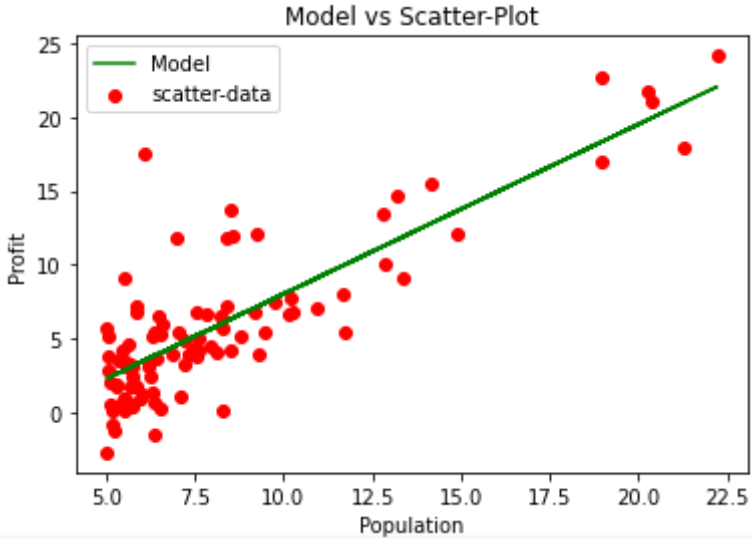

```
2. ALPHA value,i.e. learning_rate: 0.0031
Hypothesis Function: h(x) = 1.0623788480120766x + -2.5952264801035114
Cost Value(Mean Square Error) = 4.6309592432836695
```

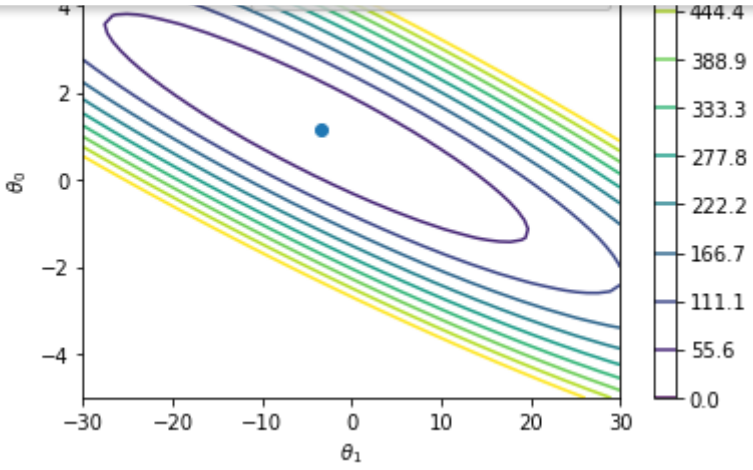






3. ALPHA value,i.e. learning_rate: 0.0061
Hypothesis Function: h(x) = 1.1487486962170874x + -3.454962894241694
Cost Value(Mean Square Error) = 4.494662200232537





4. ALPHA value,i.e. learning_rate: 0.0091
Hypothesis Function: h(x) = 1.1780322205072602x + -3.7464548059422054
Cost Value(Mean Square Error) = 4.479001397974859