PART A:

For learning rate =100

| epochs | Θ0 | Θ1 | Cost |
|---|---|---|---|
| 0.05 | 85.72 | -5.86 | 15.15 |
| 0.005 | 12.23 | 15.32 | 560.4982 |
| 0.0005 | 2.85 | 7.084 | 1275.3828 |
| 0.00005 | -0.4567 | -1.567 | 2286.8252 |

For learning rate=500

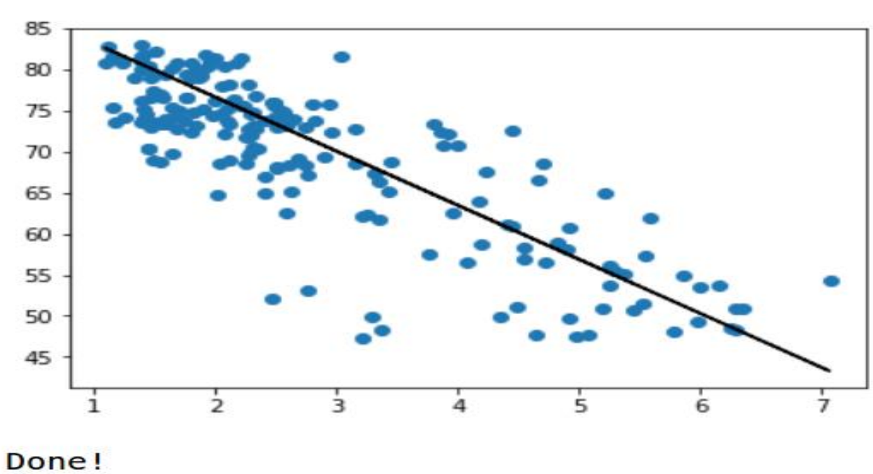
| epochs | Θ0 | Θ1 | Cost |
|---|---|---|---|
| 0.05 | 85.72 | -5.86 | 15.15 |
| 0.005 | 35.14 | 8.81 | 272.512 |
| 0.0005 | 8.40 | 15.31 | 622.44 |

For learning rate=2000

Epoch=0.01

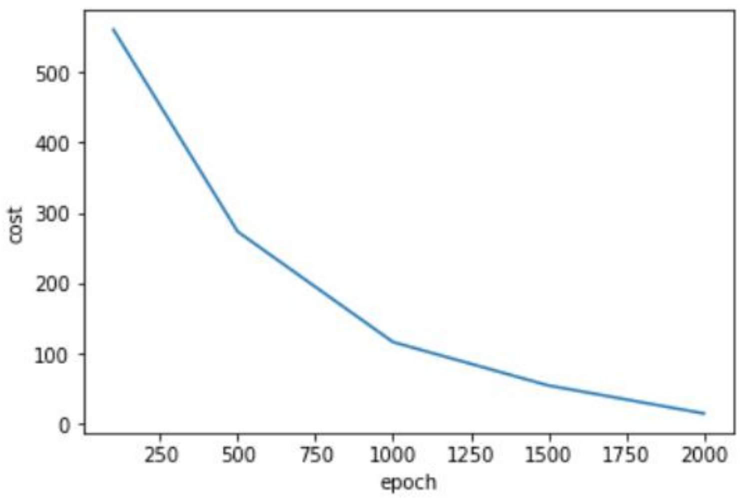
We get minimum cost =15.26

And Θ0=85.4567 Θ1=-5.678



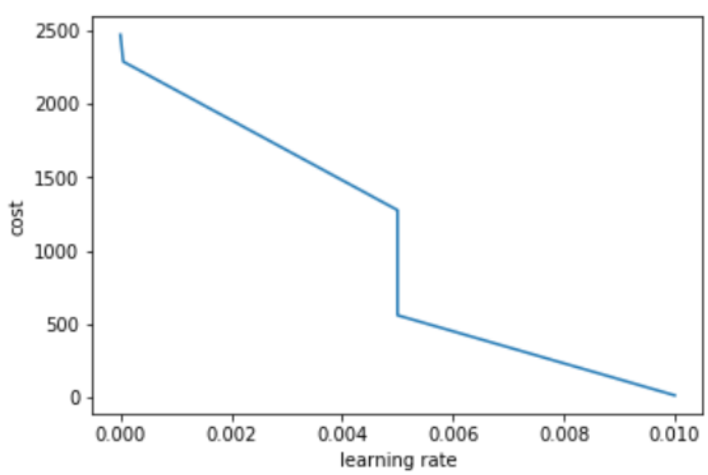

This graph is for train_x versus train_y where line is best fitted and minimum cost =15.26.

PART B:




The graph plotted for epoch versus cost.

As the epoch increases training cost decreases.




The graph plotted for learning rate versus cost .As learning rate increases cost decreases

For learning rate 0.01 and epoch =1000

| | Batch | Stochastic | Gradient |
|---|---|---|---|
| Θ0 | 89.75 | 80.8632 | 84.96896 |
| Θ1 | -6.578 | -5.523 | -5.5201364 |
| cost | 15.26 | 15.31 | 15.181876 |
| Time taken | 0.09828 | 0.4581 | 582.04 |

We observe that time taken the batch is lesser compared to stochastic and gradient.

i.e., batch<stochastic <gradient

therefore, batch gradient decent is efficient method to be followed.

PART 3.

a.

We observe from Part A that as learning rate increases training cost decreases.

And as epoch increases the training cost decreases.

But increase in learning rate leads to increasing error.

And also the approach of calculating cost takes a major role. Batch gradient descent is the best one to implement among all 3.

b.

Implementation can be done using different cost function.

$C^{NMI}$ =H(X,Y)/(H(X)+H(Y)