Problem 3(b):

1. For learning rate = 5.0e-3, Train Set Accuracy is 0.9755011135857461 and Test set Accuracy is 0.9644444444444444

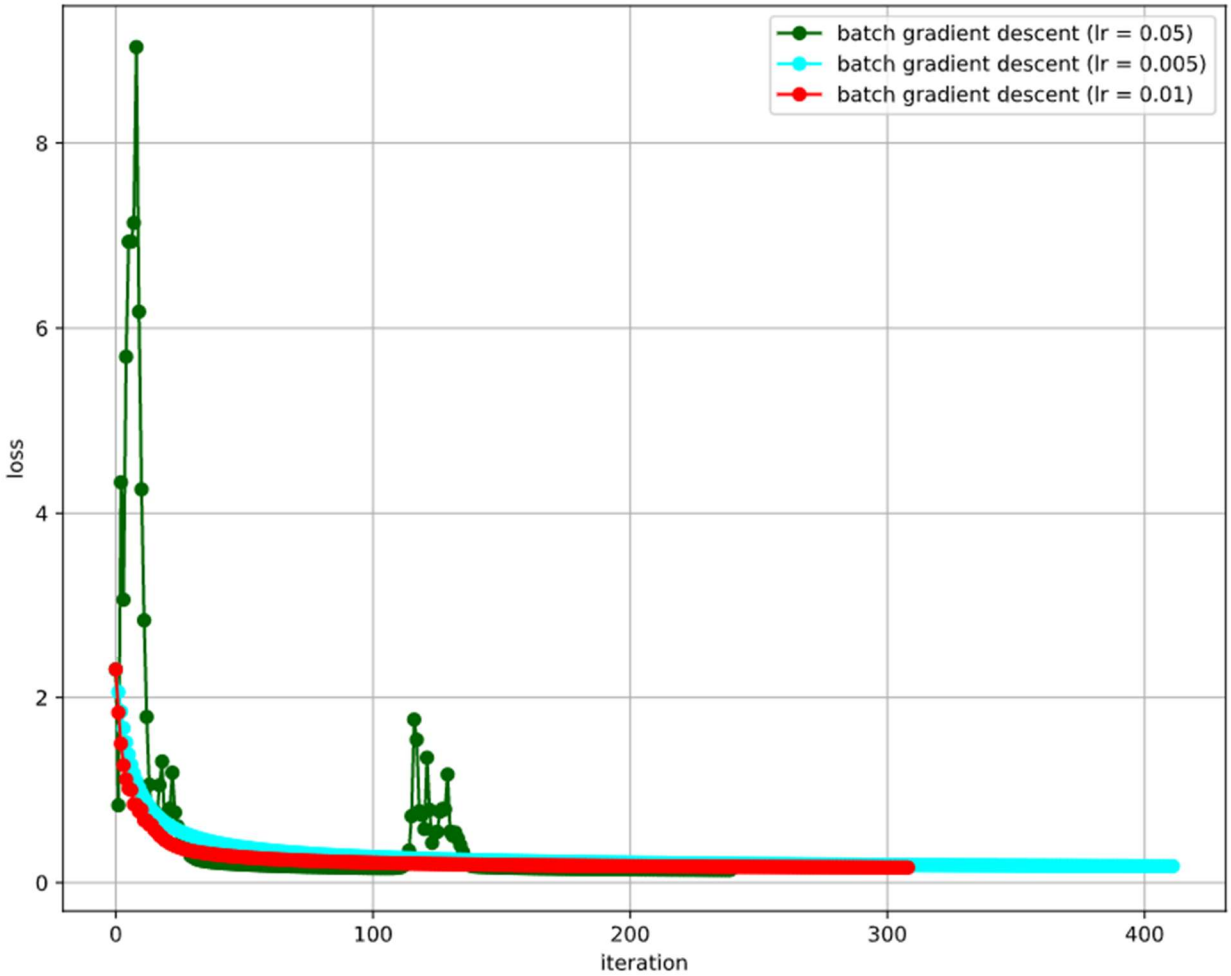
Final Value of F(w): 0.18985915843282414

2. For learning rate = 1.0e-2, Train Set Accuracy is 0.9836674090571641 and Test set Accuracy is 0.9733333333333334

Final Value of F(w): 0.1803129634628979

3. For learning rate = 5.0e-2, Train Set Accuracy is 0.9910913140311804 and Test set Accuracy is 0.9711111111111111

Final value of F(W): 0.1847604454138947



3C.

From the above experiment it was evident that while calculating the batch gradient descent for logistic regression, small learning rates made the gradient very slow. We can see from the image that learning rate 0.005 is taking about 400 iterations to converge. However, it converged ultimately. But when the learning rate is bigger like 0.05, it takes 250 iterations, but it is jumping/oscillating very rapidly overshooting the gradient descent to minima. So, when the learning rate is high although its very fast but sometimes it fails to converge, even diverge.

3d.

1. For batch = 10 and lr = 0.01, Train Set Accuracy is 0.991833704528582 and Test set Accuracy is 0.9733333333333334

Final Value of F(w) = 22.20

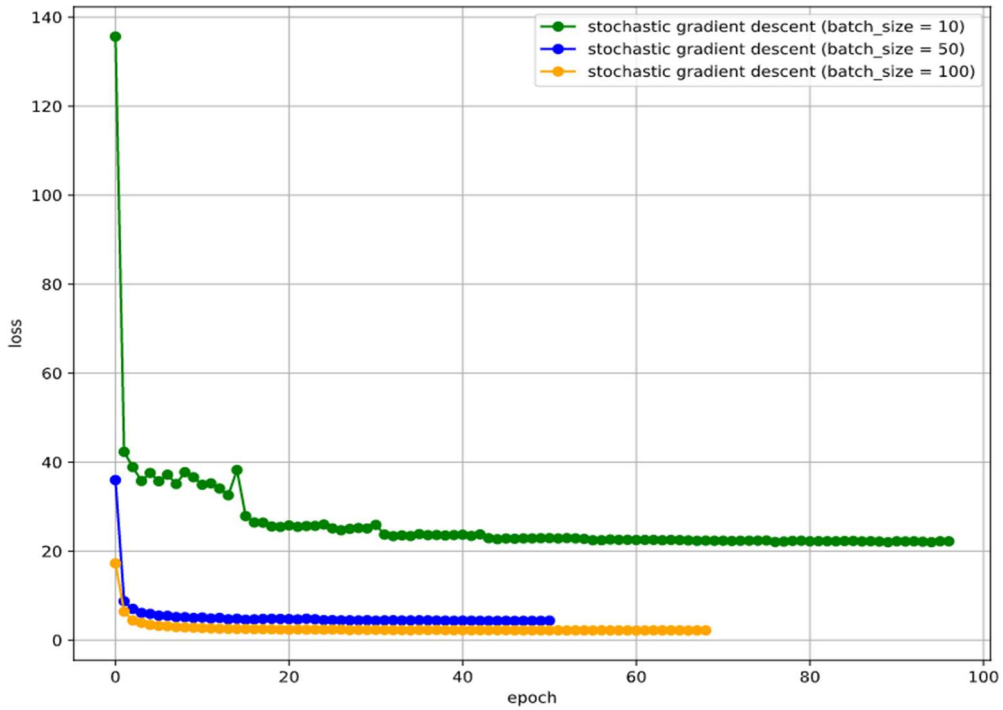
2. For batch = 50 and lr = 0.01, Train Set Accuracy is 0.9866369710467706 and Test set Accuracy is 0.9733333333333334

Final Value of F(w) = 4.3854857636758

3. For batch = 100 and lr = 0.01, Train Set Accuracy is 0.9866369710467706 and Test set Accuracy is 0.9733333333333334

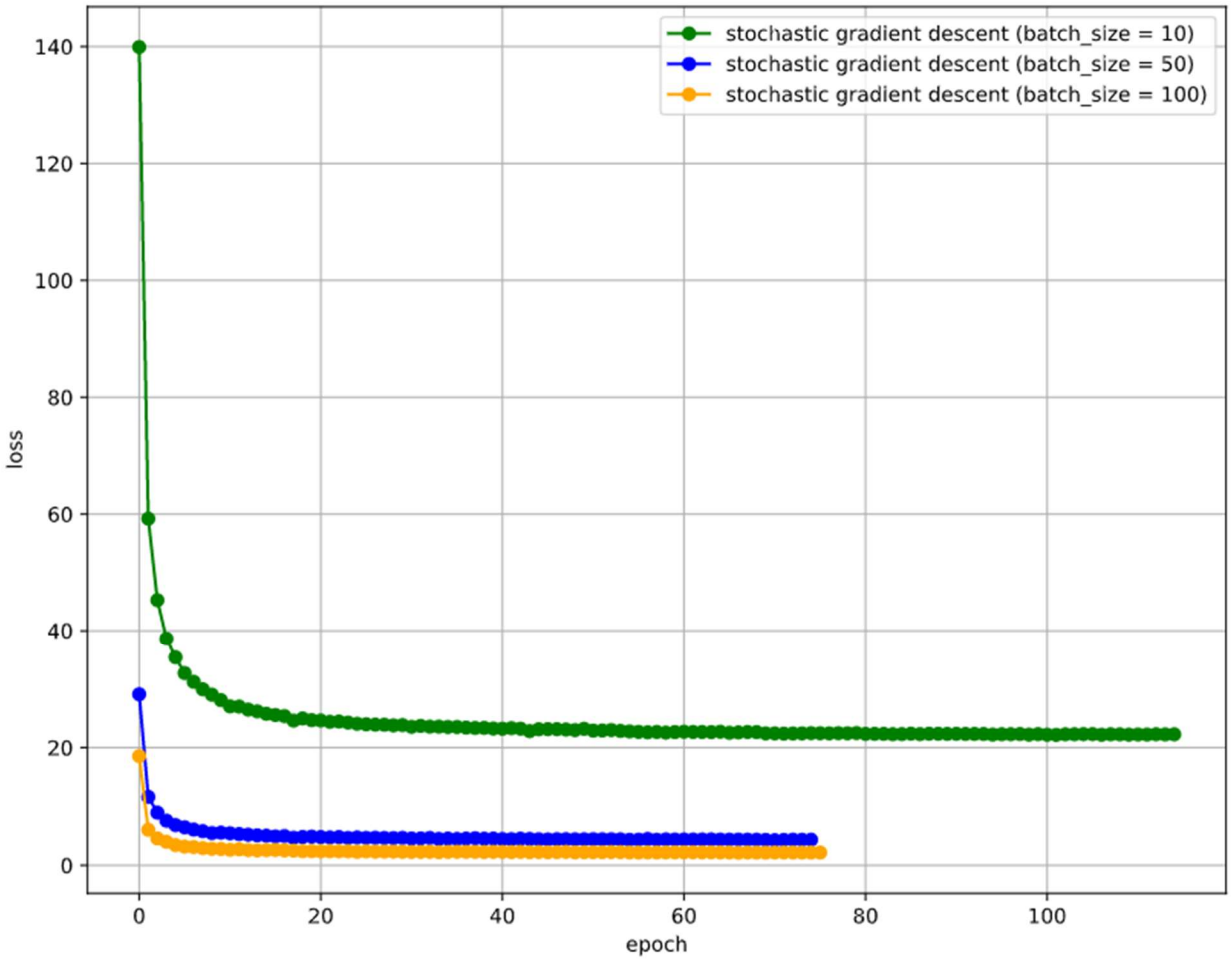
Final Value of F(w) = 0.1847604454138947

Stochastic Gradient Descent for 0.01 learning rate and different mini batches:



3e.

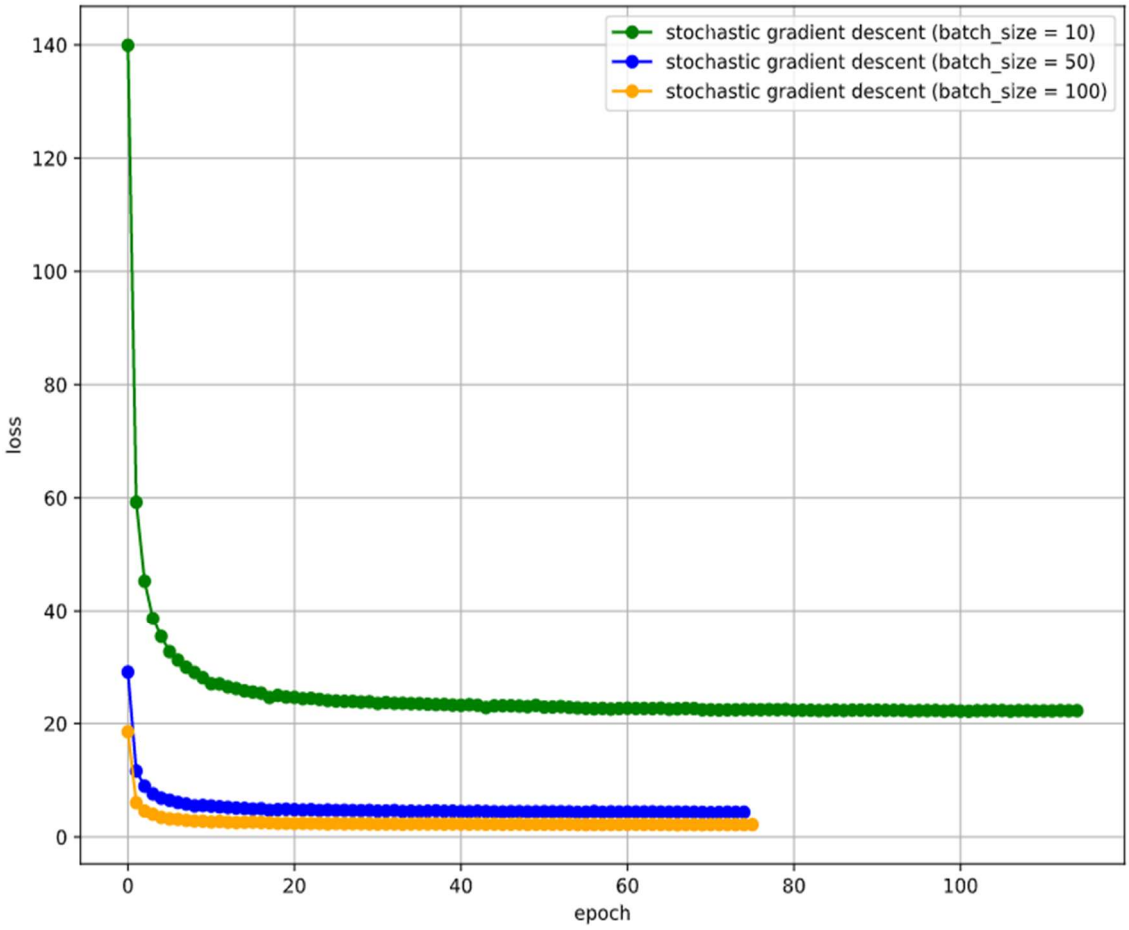
When the same learning rate (0.01) is used, the batches behave differently from each other. It is evident from the graph that, large values (50, 100) are more stable and smoother than small values. Small value (10) asserts instability by almost always oscillating and fluctuating until they converge. Also,



small value (50) take much time to converge than larger values (100)

New Convergence Curve after tuning:

As the rule of thumb is to scale the learning rate linearly with the batch size, for batch size 10, 0.001 learning, for batch size 50, 0.005 learning rate, for batch size 100, 0.1 learning rate has been used.



By comparing two graphs we can say that, when we tune the learning rate finely with respect to batch size the model becomes more stable and less fluctuates. Also, they converge very quickly in terms of wall clock time.

Mathematical Explanations:

Setting learning rate too high in SGD can cause the algorithm to diverge. Again, setting low learning rate can make the model to converge slowly. So, fast convergence requires large learning rates. The problem can be solved by considering implicit update using following equation, this is called iterative method of stochastic gradient descent:

$$w^{new} := w^{old} - \eta \nabla Q_i(w^{new}).$$