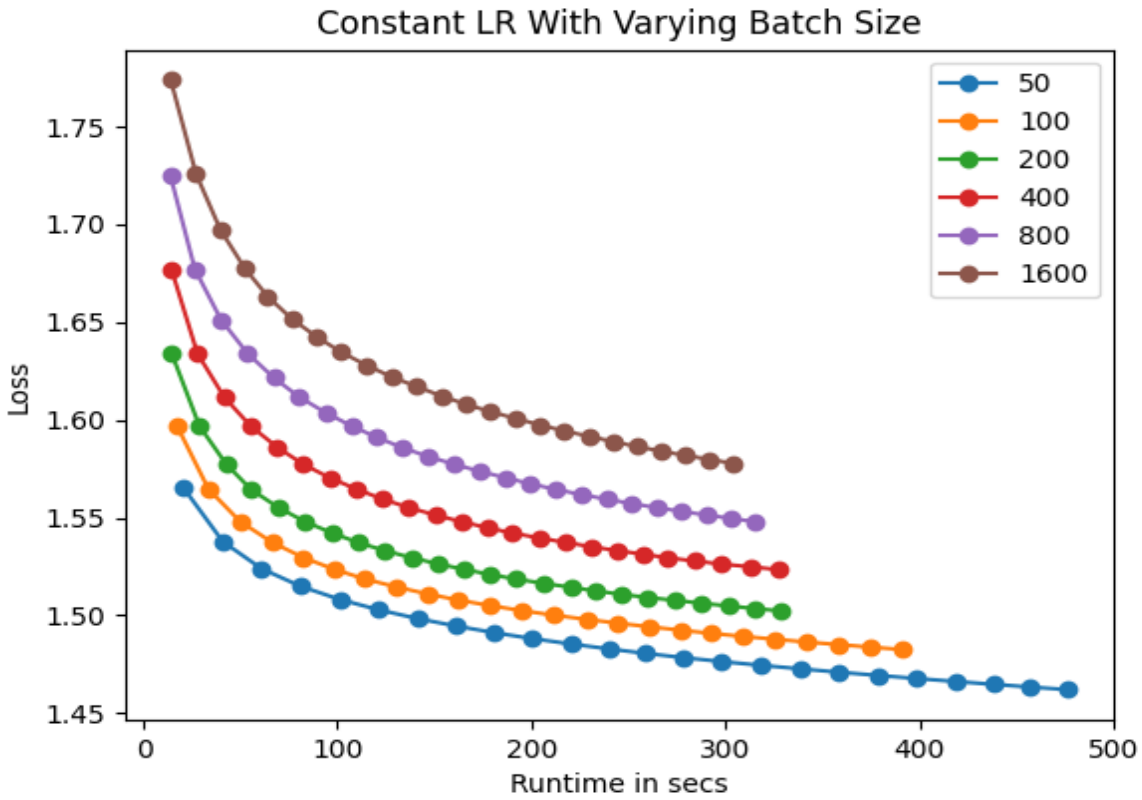
Assignment 1.2 - Report

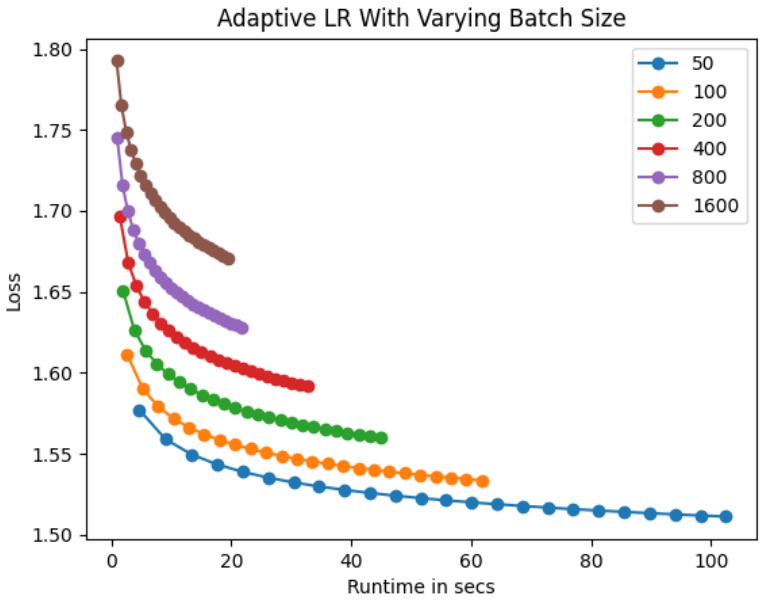
1. Choosing Best Algorithm:

A. Method 1 - Constant Learning Rate

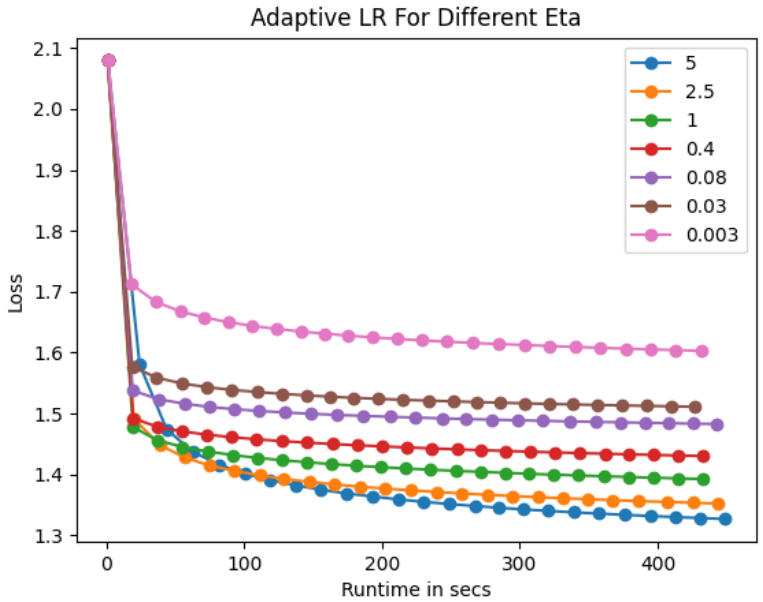
In the figure given below, we have plot of loss vs runtime of minibatch gradient descent having constant learning rate. We have size plots corresponding to 6 different vatch sizes. As clear from the plot, as the bach size decreases, loss decreeases wherase the runtime increases. Without **considering the runtime constraint, best batch size is 50**. But, considering the time constraint, then the **optimal batch size would be 200.**



B. Method 2 - Adaptive Learning Rate



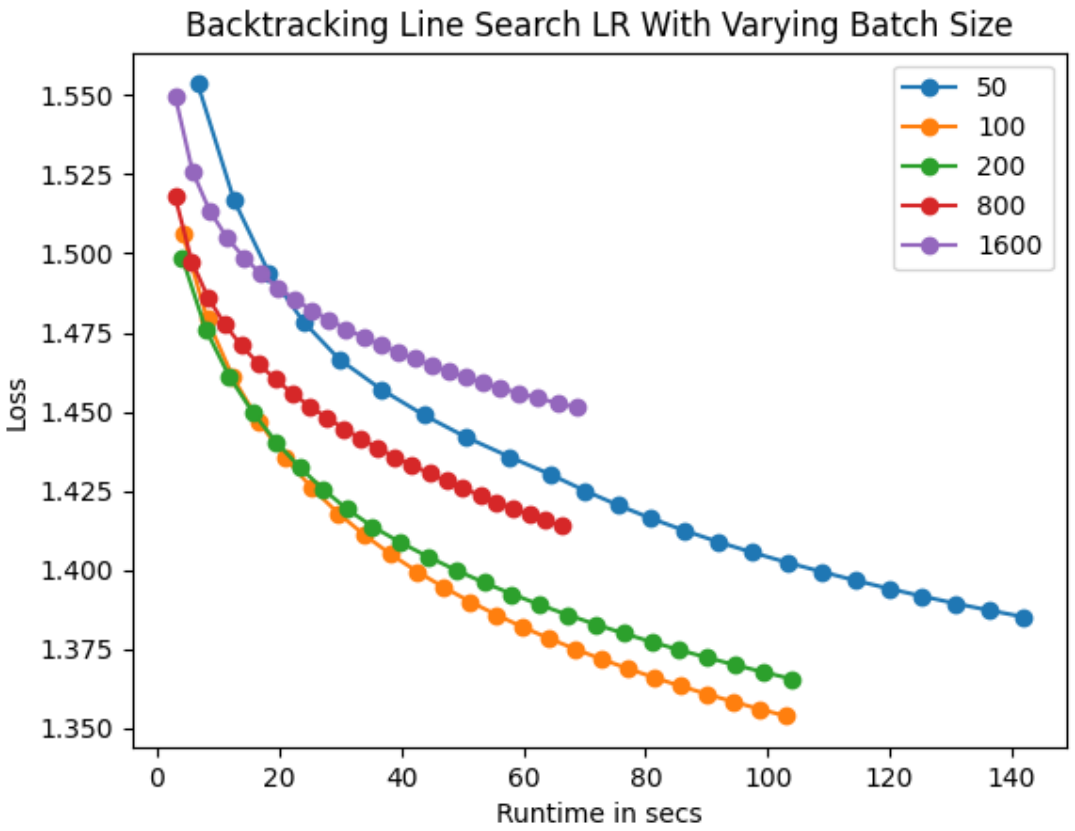
In the figure given above, we have plot of loss vs runtime of minibatch gradient descent having adaptive learning rate. We have size plots corresponding to 6 different vatch sizes. As clear from the plot, as the bach size decreases, loss decreeases wherase the runtime increases. Without **considering the runtime constraint, best batch size is 50**. But, considering the time constraint, then the **optimal batch size would be 100.**



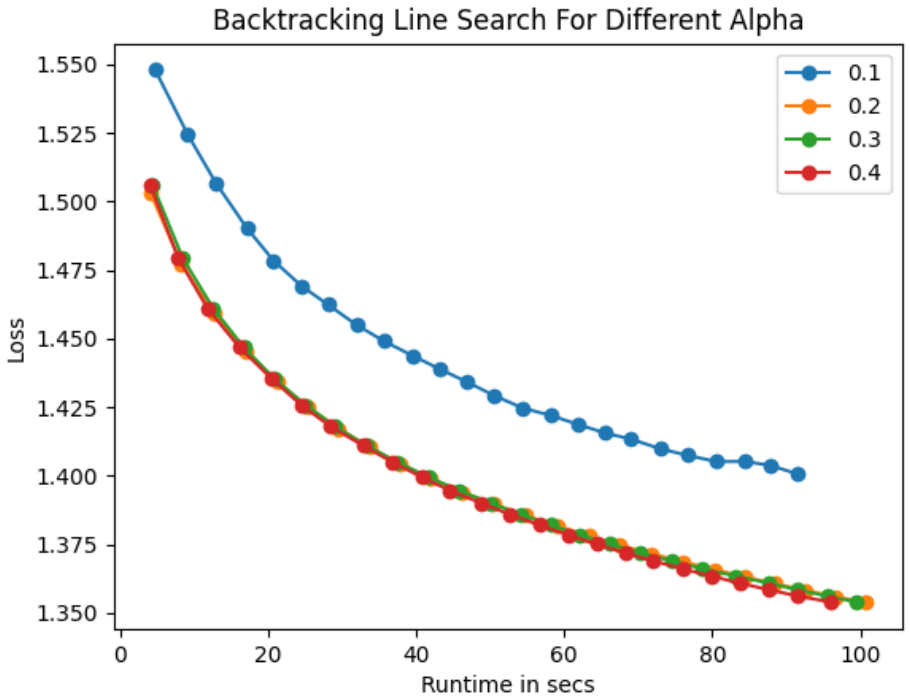
In order to find the best eta for adaptive learning rate, I have plotted variation of loss function as function of runtime in the figure given above. The above plot shows varations for six different eta. **The best value eta as clear from the plot is 5.** For more higher values of eta, the loss function oscillates and does not converge.

C. Method 3 - Adaptive Learning Rate using backtracking line Search

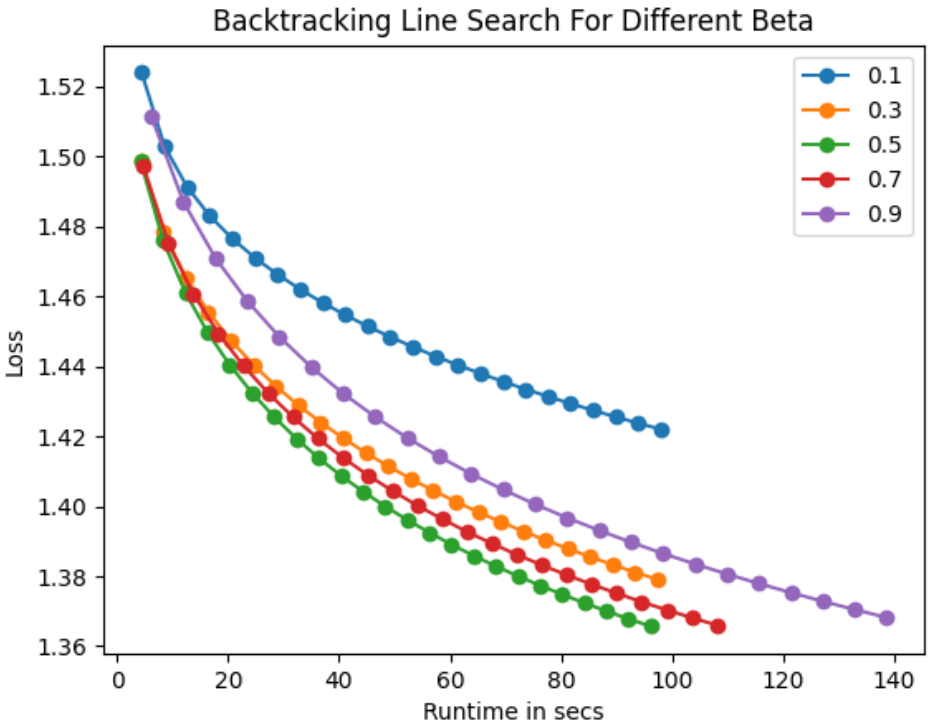
In the figure given below, we have plot of loss vs runtime of minibatch gradient descent having using bakctracking line search. We have size plots corresponding to 6 different vatch sizes. As clear from the plot, loss function does not show any fixed trend with varying batch size. From the plot, we observe that **best batch size is 100**.



In the figure given below, we have plot of loss vs runtime for different values of alpha in the backtracking line search. We have size plots corresponding to 4 different alpha. From the plot, we observe that **best value of alpha can be 0.2, 0.3, 04**.



In the figure given below, we have plot of loss vs runtime for different values of beta in the backtracking line search. We have size plots corresponding to 5 different beta. From the plot, we observe that **best value of beta is 0.5**.



D. Conclusion

Since we have time constraint as the data set is huge, so we will prefer to use mini-batch gradient descent algorithm.

As clear from the above plots, the loss function decrease in the range between 1.3 to 1.4 for both adaptive line search in method 2 and backtracking line search. Now, in order to select out of these two algorithms, we will compare the run time. Method 2 is faster than the backtracking line search. Thus, we will prefer method 2 over method 3.

The optimal values for method 2 is eta = 10 and batch size = 100. We have considered both runtime and loss value in order to select the optimal value

2. Feature Engineering

I did many experiments. Some of them failed and some showed improvement which have been incorporated in the code. Following are the methods which I tried to improve the model:

A. Feature Creation - Tried to create new features by combining two features. This method also did not show any improvement. On the other hand, it showed slight decrease in the accuracy.

B. Feature Extraction - Tried removing the less important features like payment topology, birth weight, etc. manually, but the accuracy decreased. So, this experimentation failed.

C. One Hot Encoding - Since, most of the categorical, so, one hot encoding has been used. This method shows improvement in the model but it also slows down the code.

D. Feature Selection - In order to select best 500 featuresa out of created features after one hot encoding, SelectKBest method of the sklearn has been used. This function selects the best 500 features which are more relevant i.e. has higher co-variance with the output.