March 16, 2018

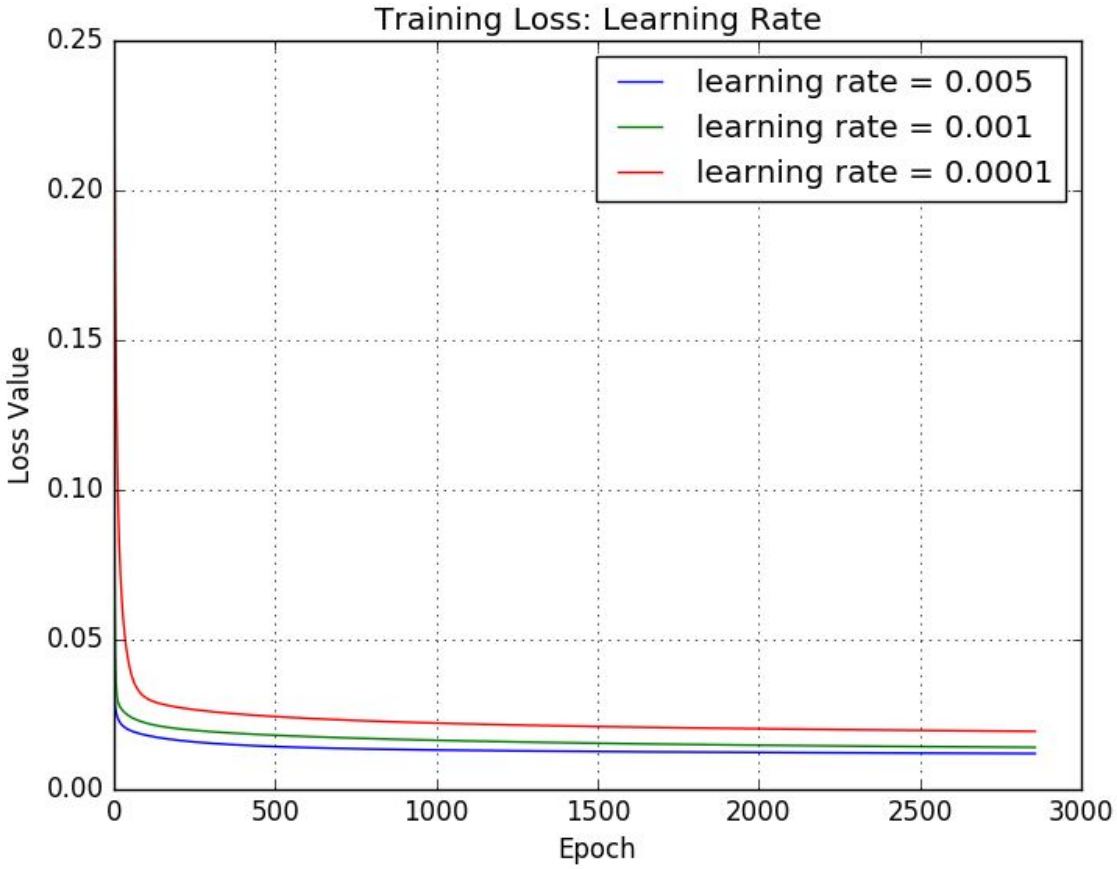
ECE521: Assignment 2

Linear and Logistic Regression

Younghoo Cho (1000594270), Deng Pan (1000548874), Keon Young Park (1004686253)
Equal contribution from each member.

1 Linear Regression

1.1 Tuning the learning rate



As shown above, a learning rate of 0.005 leads to the lowest loss value. As can be seen, a lower learning rate causes a much slower convergence than higher learning rates. In the early epochs, because the weight vector is probably very far off from its convergent value, there's a lot of change to cause the relatively large drop in loss value. Additionally, it can be seen that the lowest learning rate takes the longest time to get to the turning point where the slope changes suddenly changes to a much smaller value. The 0.0001 learning rate has that change somewhere around 100 epochs, while the others seem to have them near 10 epochs. This

makes sense since having a smaller learning rate means you can't make as much of a change to the weight vectors as higher learning rates.

1.2 Effect of the mini-batch size

The learning rate chosen from Section 1.1 is 0.005. Here are the final training MSE for each mini-batch value, as well as their training time:

Mini-batch value	**Final Training MSE**	**Training TIme**
500	0.011970211983829403	86.972562535
1500	0.01213451006782122	257.009459338
3500	0.01196846592921197	775.468186075

The best mini-batch value in terms of training time is 500. This makes sense as the number of iterations were determined to be how many mini-batches are run through the learning algorithm. Therefore, if the number of iterations is 'n', then for a mini-batch value of 500, n * 500 inputs are parsed and learned on, while for a mini-batch value of 350, n * 3500 inputs are parsed and learned on. In the learning algorithm, gradient descent is used. Therefore, reducing the number of gradients that need to be run causes improved performance. Since a mini-batch of 500 gives less inputs to calculate gradients on, the performance is improved and therefore the training time is reduced.

1.3 Generalization

Below shows the validation set accuracy and mean-squared error values for different decay coefficients in the loss function.

Decay coefficient	**Validation Set Accuracy**	**MSE**
0	0.99	0.011970211983829403
0.001	0.98	0.012178365737502195
0.1	0.98	0.013448585357307669
1	0.98	0.016551587136539023

As shown above, a decay coefficient of 0 caused the highest validation set accuracy. Using this decay coefficient, the test set accuracy is 0.9655172413793104.

Seeing the chart above, having higher decay coefficient causes a relatively smaller validation set accuracy over lower decay coefficients. However, the validation set accuracies are so similar that it's hard to say if this is due to the decay coefficient or unlucky inputs. The decay coefficient should be tuned by the validation set over the training set because having both the weights and the decay coefficient be tuned on one set could lead to overfitting.

1.4 Comparing SGD with normal equation

Using the normal equation of the least squares formula, the following are the relevant statistics. For the SGD statistics, the model from Section 1.3 is used, with the decay coefficient set to 0.

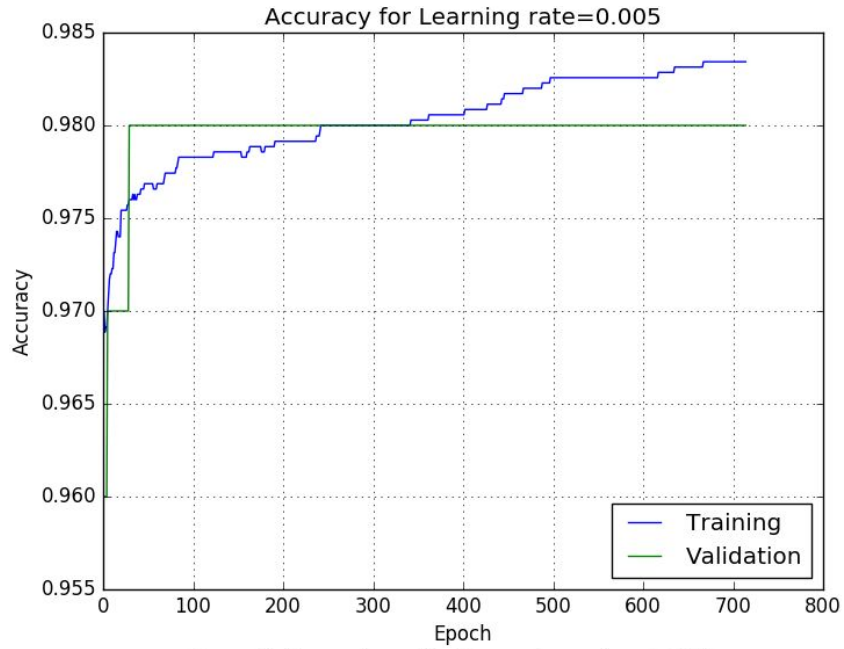
	Final Training MSE	Accuracy	Computation Time (s)
Normal equation	0.009391804	0.9934286	0.019635438919067
SGD	0.011970212	0.9655172413793104	74.748332335

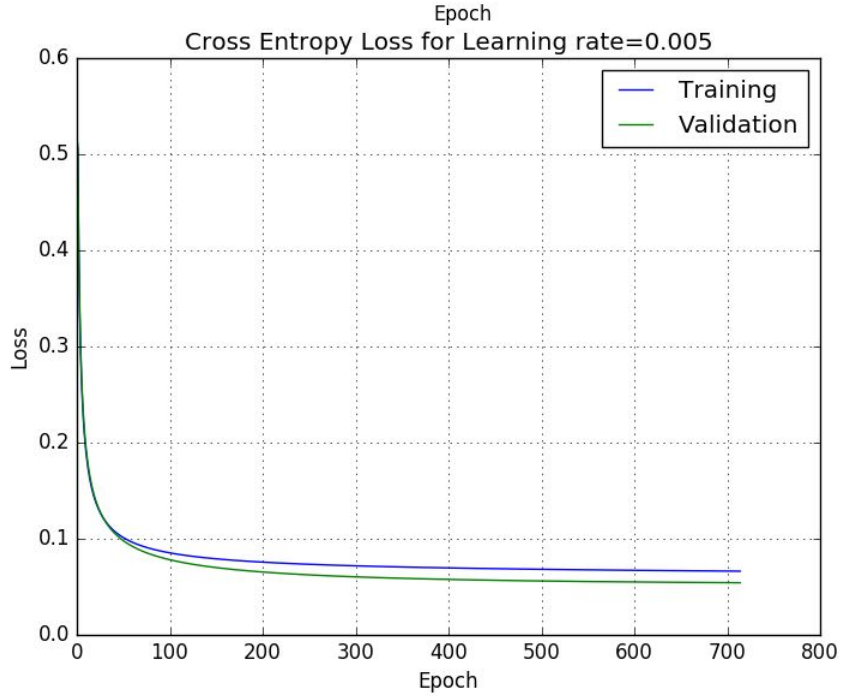
As shown above, the normal equation creates a weight vector that has lower training MSE, higher accuracy, and lower computation time compared to SGD. This makes sense because the normal equation is the theoretical optimal value for the weight vectors; SGD approximates that value. Additionally, the computation time is significantly higher for SGD than normal equation. This also makes sense because the SGD method needs to learn by iterating and improve its weight vector while the normal equation analytically gives the optimal value. However, there may be scenarios where the normal equation would take longer than SGD. For instance, the normal equation does a matrix inverse in order to calculate the optimal value. This can cause significant computation time if the matrix being inversed is large. In that case, SGD may be more practical than using the normal equation. Additionally, the normal equation is specifically for linear regression. SGD can be used with any loss function, and therefore is more flexible and general in that way.

2 Logistic Regression

2.1 Binary cross-entropy loss

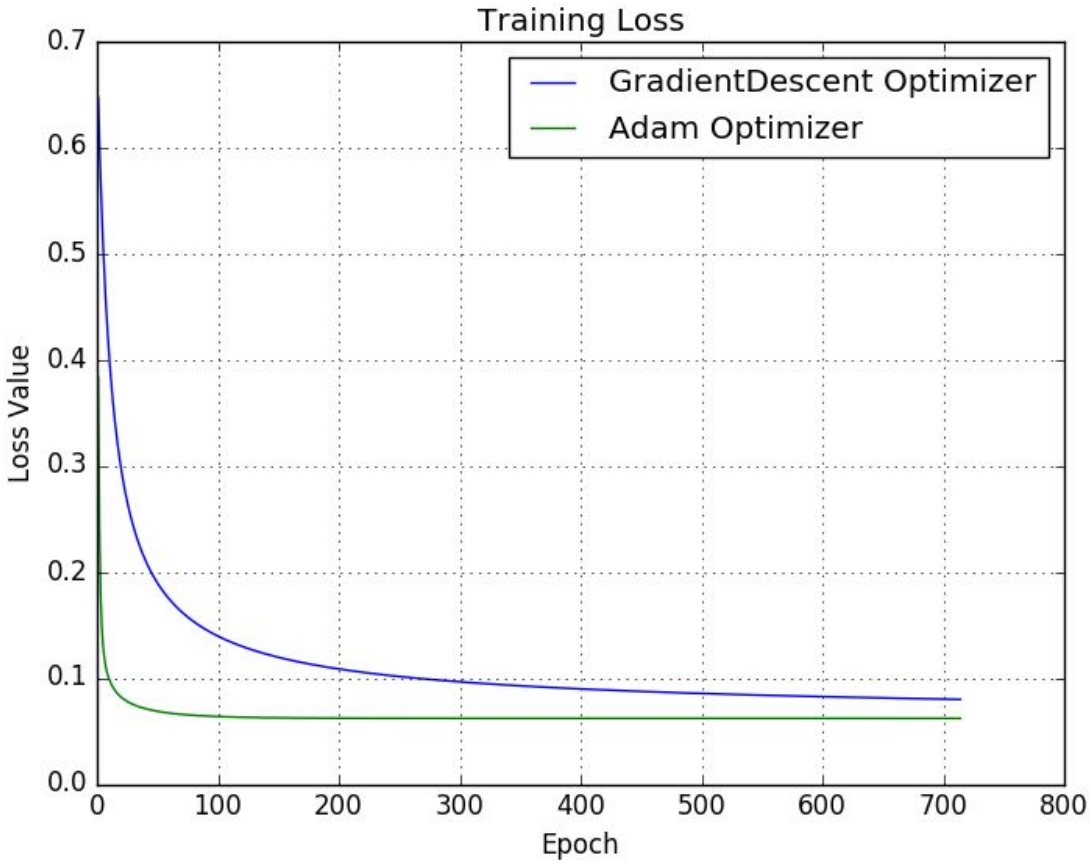
2.1.1 Learning





The learning rate was tuned to be 0.005 at it gave the best accuracy. The test accuracy obtained from a learning rate of 0.005 using the logistic regression model is 0.9834286.

2.1.2 Beyond plain SGD



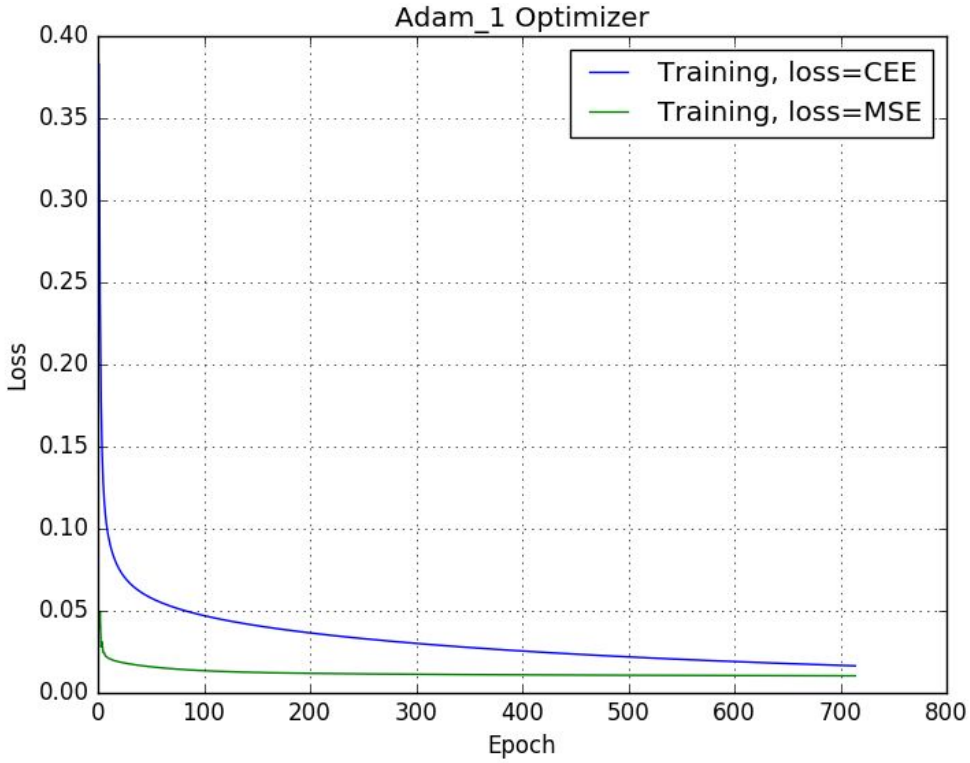
With a weight decay coefficient = 0.01, learning rate = 0.001, and a minibatch size of 500, the Adam optimizer yields a much lower cross entropy loss value as shown in the above graph. The primary motivation of using the Adam optimizer is to counter the difficulty in selecting hyperparameters, especially the learning rate. Thus, researchers have discovered that using adaptive learning rate algorithms such as the Adam Optimizer facilitated by the techniques of momentums and rescaling can lead to faster convergence rates compared to fixed learning rate procedures such as SGD. This trend is evidently displayed in the graph above as the Adam optimizer converges much faster than the SDG optimizer around the first 100 epochs, demonstrating its superiority in expending less computational resources.

2.1.3 Comparison with linear regression

The following is the accuracies for both the linear regression solution from the normal equation, and the optimal logistic regression learnt without weight decay. The logistic regression uses a learning rate of 0.005 which was found to be the most optimal.

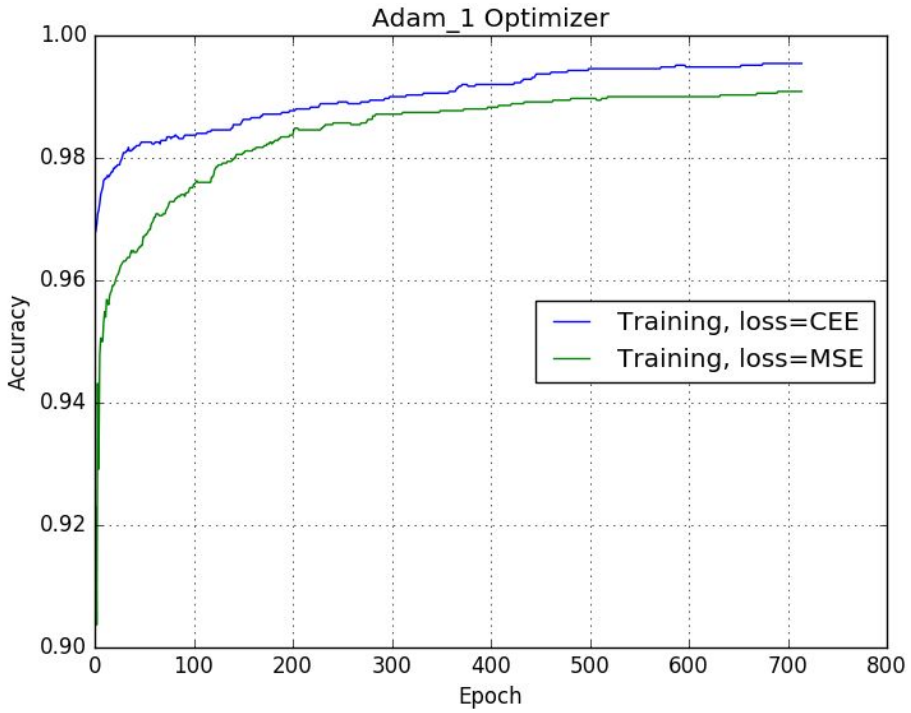
	Training Accuracy	**Validation Accuracy**	**Test Accuracy**
Normal Equation Linear Regression	0.9934286	0.97	0.95862067
Logistic Regression	0.9997143	0.99	0.97931033

As shown above, the logistic regression model performs slightly better than the normal equation solution to the linear regression model since each accuracy for all three data sets are higher for logistic regression. Even though the normal equation gives the optimal solution to linear regression, it could perhaps perform worse than the logistic regression model because logistic regression is not linear regression. Perhaps the logistic regression models has some property that allows it to beat even the most optimal linear regression models.



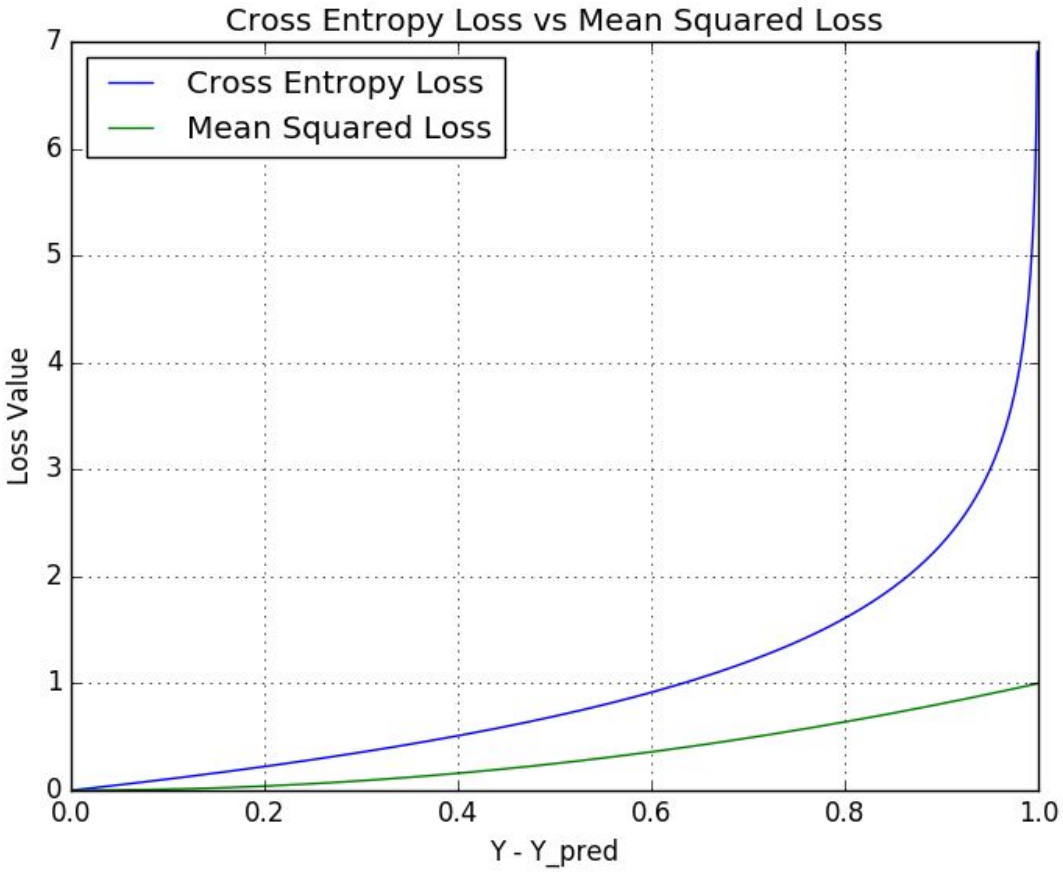
Above is the comparison between the loss values of cross-entropy (CEE) and mean-squared error (MSE), which refer to logistic regression and linear regression, respectively. As can be seen, the CEE loss is higher than MSE for all of the measured epochs.

March 16, 2018



The above shows the comparison of the accuracy between logistic regression (CEE) and linear regression (MSE). As can be seen, the logistic regression gets better accuracy than linear regression over all epochs.

To explain the effect of the convergence behaviour of cross-entropy loss, the following is a graph that compares the functional behaviour between cross-entropy and squared error.



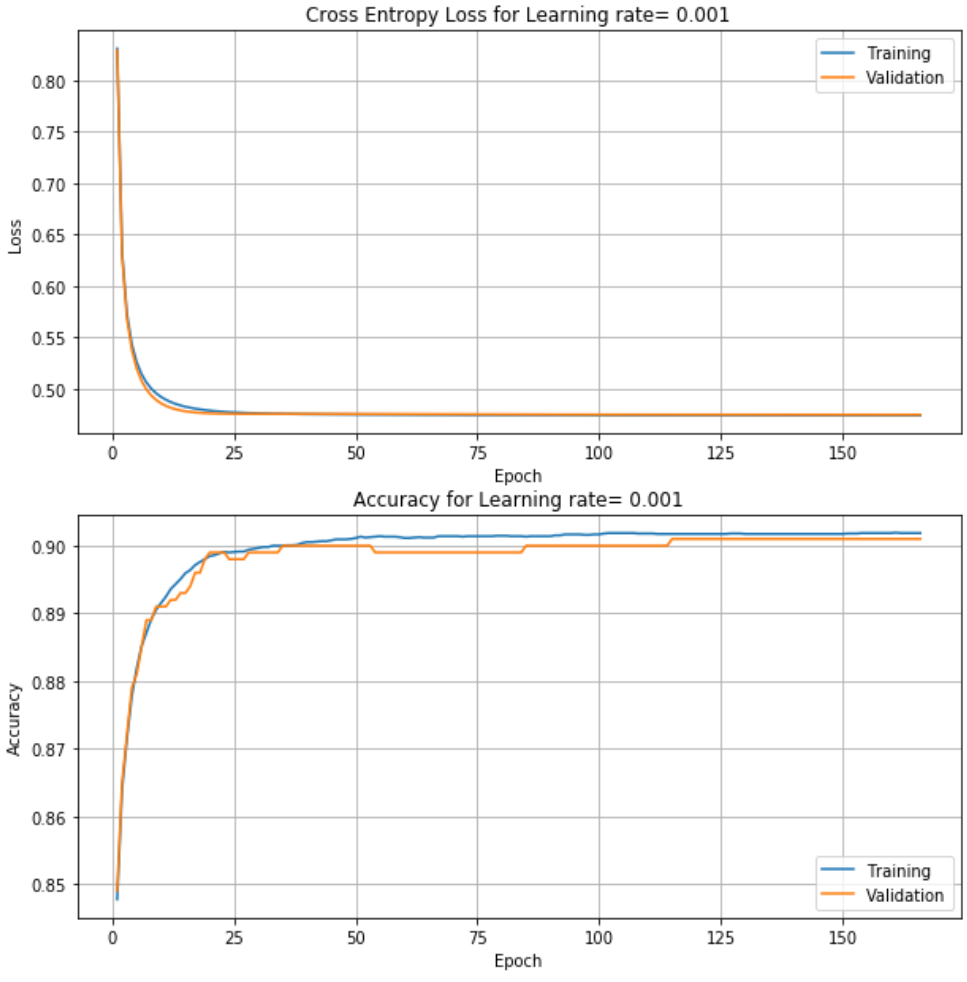
As can be seen above, the cross-entropy error formula is more sensitive to error, shown by the cross entropy loss value being higher than mean squared error's loss value. This essentially shows why the cross-entropy formula converges faster than mean squared error. This is because in the initial epochs, the weight vector will be very far from its convergent value. When this occurs, the cross entropy loss functions gives higher loss, causing the learning algorithm to more aggressively change its weight vector to try to reduce the error.

2.2 Multi-class classification

2.2.1 notMNIST Dataset

By using decay coefficient of 0.01 and batch size of 500, the learning rate that gives the best performance turned out to be **0.001.**

Learning Rate	Decay Coefficient	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
0.001	0.01	0.901867	0.901	0.474472	0.475037



Based on the best learning rate we got from validation set, the best test accuracy turned out to be **89.207%**

Learning Rate	Decay Coefficient	Training Accuracy	Test Accuracy	Training Loss	Test Loss
0.001	0.01	0.901867	0.89207	0.474472	0.512911

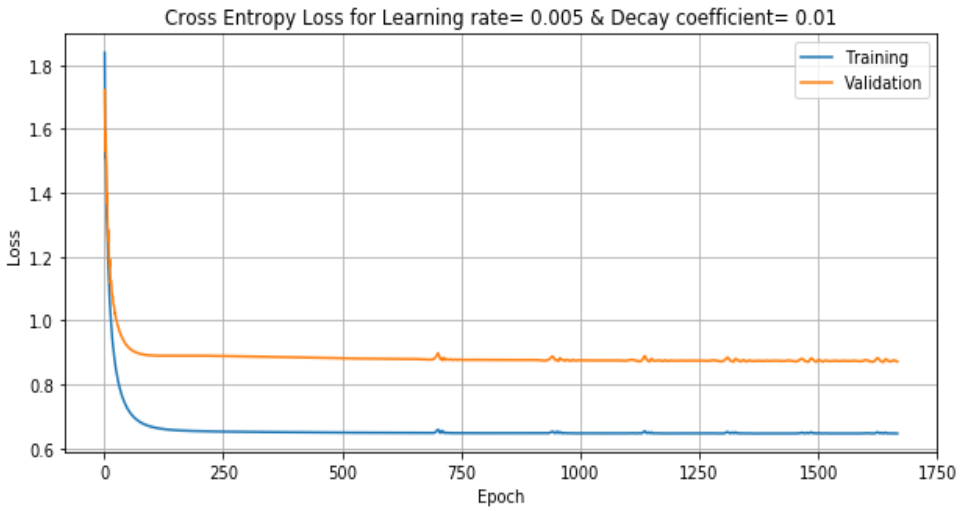
As can be seen by the validation accuracy above and the validation accuracy shown in the previous sections, the multiclass classification problem leads to less accurate models. This intuitively makes sense because there are now more options to choose from in terms of which classification a test point could refer to. Therefore, out of the 100 percentages given in the probability in choosing a label, the percentages are spread out more across more options, which gives the majority label with less percentages. Therefore, the model is more likely to be less sure in its predictions in the multi-class problem compared to the binary-class problem.

2.2.2 FaceScrub Dataset

By using a batch size of 300, the values of learning rate and weight decay that gives the best performance are **0.005 and 0.01** respectively.

Learning Rate	Decay Coefficient	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
0.005	0.01	0.919679	0.869565	0.645741	0.8712278

March 16, 2018



By using the best combination of learning rate and decay coefficient we got from validation set, the test accuracy value turned out to be **87.097%**

Learning Rate	Decay Coefficient	Training Accuracy	Test Accuracy	Training Loss	Test Loss
0.005	0.01	0.919679	0.870968	0.645741	0.802551

In the assignment 1, the best accuracy of k-NN algorithm over the test dataset was **70.968%** with k = 1, whereas the best accuracy over the test dataset using logistic regression is **87.097%**, which is approximately ***16% higher*** in the accuracy value than that of k-NN.