GT Username: sanne31@gatech.edu

Part 1. Implementation

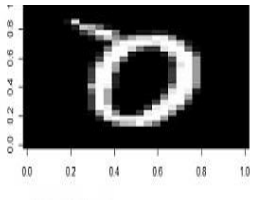
- The highest number of epochs I used were 100
- Set the initial epsilon value to 0.001
- Convergence criteria I used was
  - $|norm(\Theta - \alpha * \text{gradient}) - norm(\Theta)| < (\varepsilon)$
- To avoid errors like the below
  - *Error in x %*% theta: requires numeric/complex matrix/vector arguments*

  I converted all elements to a matrix format to enable easier processing
- Was able to avoid any for loops for easier and quicker training => using vectorization

Correct and Incorrect samples are highlighted in green and red respectively: (plotted using the predicted labels and not the true labels so the incorrect ones could be observed)

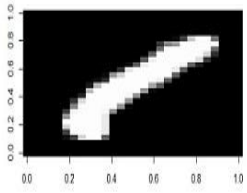
Class label 0

Predicted label - 0



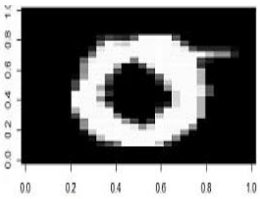
Class label 1

Predicted label - 1



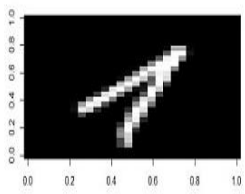
Class label 1

Predicted label - 0



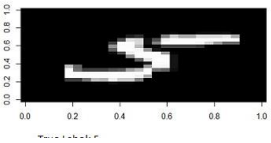
Class label 0

Predicted label - 1



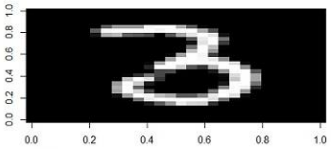
Class label 5

Predicted label - 5



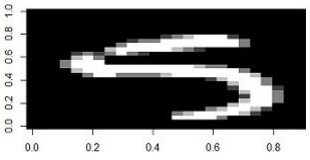
Class label 3

Predicted label - 3



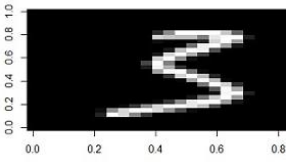
Class label 3

Predicted label - 5



Class label 5

Predicted label - 3



Part 2. Modeling

- Learning rates from 0.1 to 0.6 with an interval of 0.05

variable_learning_rate	train_accuracy_01	test_accuracy_01	train_accuracy_35	test_accuracy_35
0.1	0.9857865	0.9895932	0.799169	0.7865405
0.15	0.9887871	0.9924314	0.8310249	0.8138801
0.2	0.9908402	0.9924314	0.849723	0.8349106
0.25	0.9921036	0.9933775	0.8601108	0.8506835
0.3	0.9928932	0.9943236	0.8713643	0.8559411
0.35	0.993525	0.9952696	0.8812327	0.8664564
0.4	0.9941567	0.9952696	0.8860803	0.8748686
0.45	0.9947884	0.9962157	0.8902355	0.8822292
0.5	0.9951042	0.9962157	0.8935249	0.8895899
0.55	0.9951042	0.9962157	0.8973338	0.893796
0.6	0.9957359	0.9962157	0.900277	0.893796

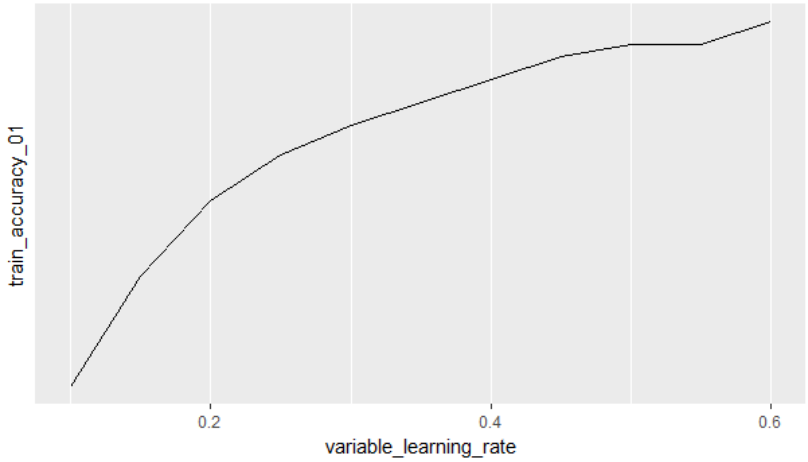
- As we can see from the above table, the accuracies for both training datasets and testing datasets have been compiled again for both classification i.e. 0/1 dataset and 3/5 dataset
- But for variable learning rates, it is not entirely simple to decide which dataset has the highest accuracy so I compiled another vector having the means of all the accuracies for all the learning rates tested.

Mean Accuracies	0.992620455	0.994323555	0.869097836	0.858426536

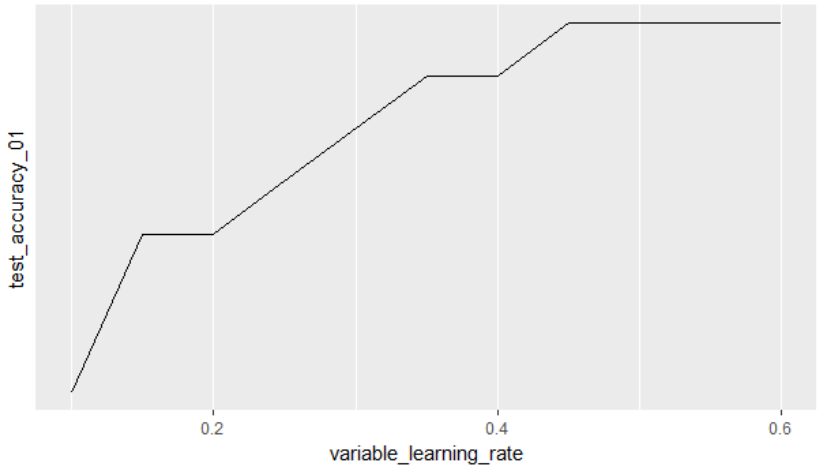
- Thereby, we can now deduce that the highest accuracy is for the *testing dataset for the classification of the 0/1 dataset*. The highest accuracies for each dataset are given below
    - Training – 0/1 -> 99.57%
    - Testing – 0/1 -> 99.62%
    - Training – 3/5 -> 90.03%
    - Testing – 3/5 -> 89.38 %

- The plots for each dataset against the learning rates are given below.

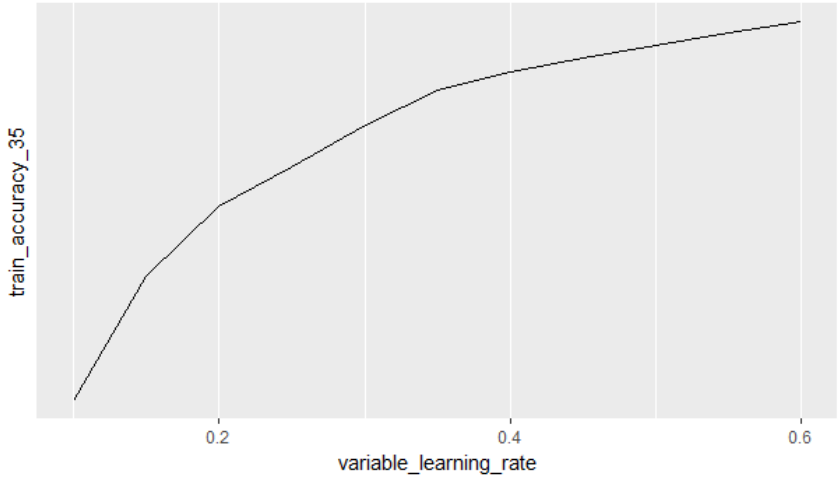
Training Accuracy 0/1 set vs Variable Learning rate



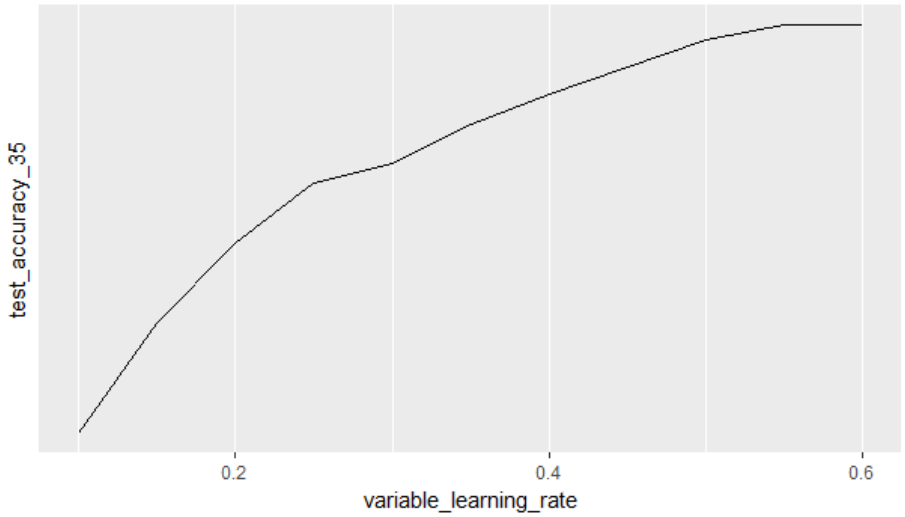
Testing Accuracy 0/1 set vs Variable Learning rate



Training Accuracy 3/5 set vs Variable Learning rate



Testing Accuracy 3/5 set vs Variable Learning rate



- Further observations lead to the result that the 0/1 datasets generally have a higher accuracy rate than the accuracy rate of the 3/5 datasets. The reason might be that the 3/5 set might be more cluttered leading to a denser matrix than that of the 0/1 dataset.

- Dealing with multi class classification
    - Present classification is binary => either 0 / 1 or 3 / 5
    - When multiple classification situations arrive, I think the best way is to convert that multiple class into a separate binary class
        - 1 class and the remaining sets as 1 class => 2 classes => binary classification

Part 3. Learning Curves

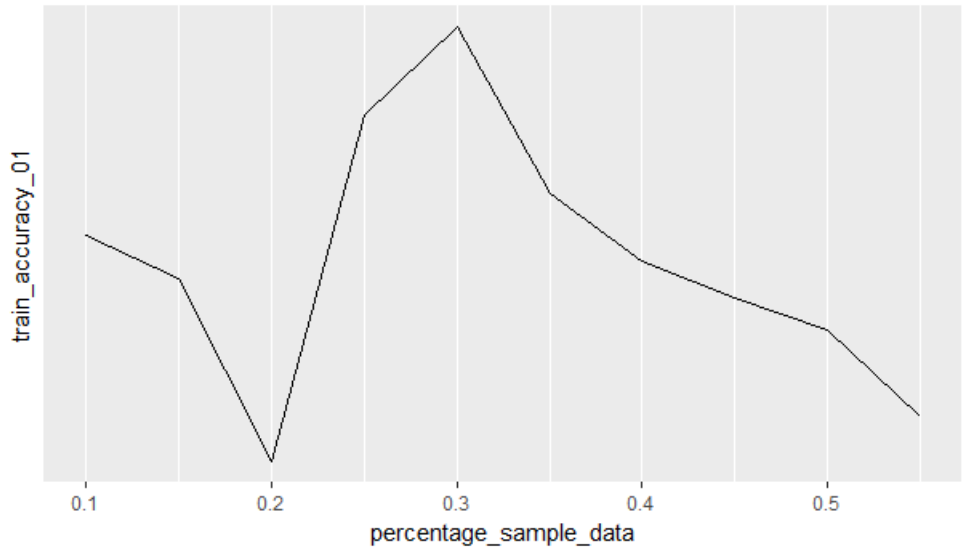
- This task tests the sample size with a fixed learning rate – kind of the inverse of what was done in the previous task.
- Might be difficult to generalize the accuracy trend for the entire dataset but can be able to distinguish upon further classification between the 0/1 and the 3/5 sets.
- Accuracies for datasets varying with the sample size is given below.

Percentage of sample data	train 01	test 01	train 35	test 35
0.1	0.9973862	0.9971564	0.8986547	0.8850126
0.15	0.9971304	0.9961285	0.8999826	0.8978132
0.2	0.9960521	0.9976359	0.9125541	0.9021652
0.25	0.9981043	0.9981025	0.9143987	0.9158349
0.3	0.9986231	0.9973984	0.9151615	0.9192642

0.35	0.9976356	0.9969042	0.9032546	0.9287531
0.4	0.9972365	0.998818	0.9002897	0.9302546
0.45	0.9970207	0.9990451	0.9113725	0.9254873
0.5	0.9968254	0.9988623	0.9160385	0.9165807
0.55	0.9963153	0.998424	0.9119632	0.9100652

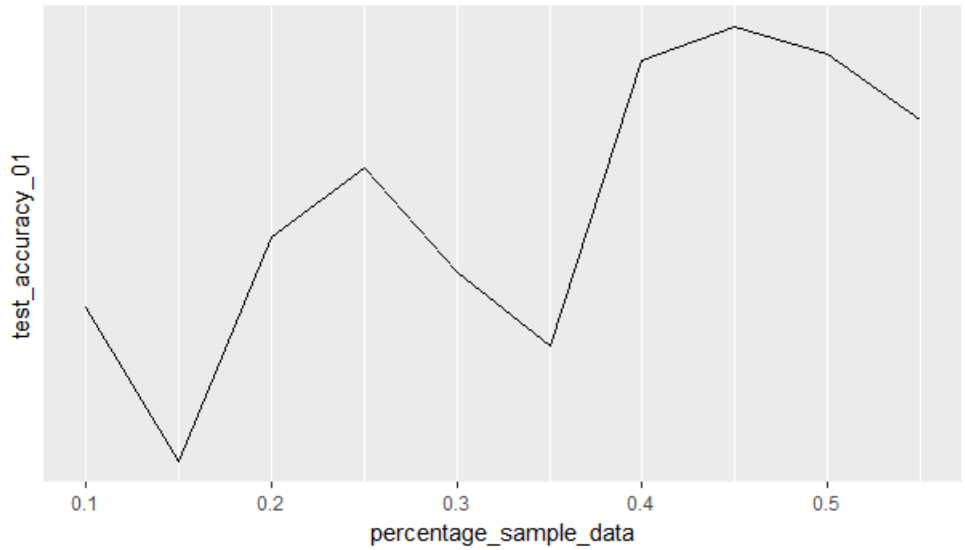
- From direct observation, we can see that the accuracies tend to fluctuate and there is no solid observation about whether they are increasing or decreasing with the sample size. Visualizations may help further.

Training Accuracy for 0/1 dataset vs sample size



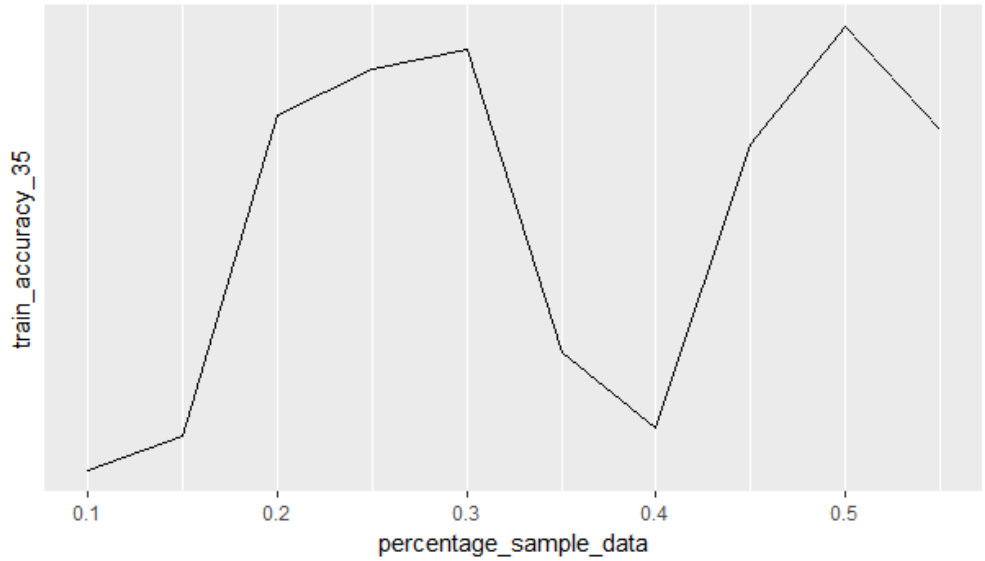
- Starts off decreasing and once it reaches a minimum value increases to a maximum for 30% sample size and then decreases gradually as the sample size increases. Final value is lower than the initial value so accuracy tends to decrease with increase in sample size.

Testing Accuracy for 0/1 dataset vs sample size



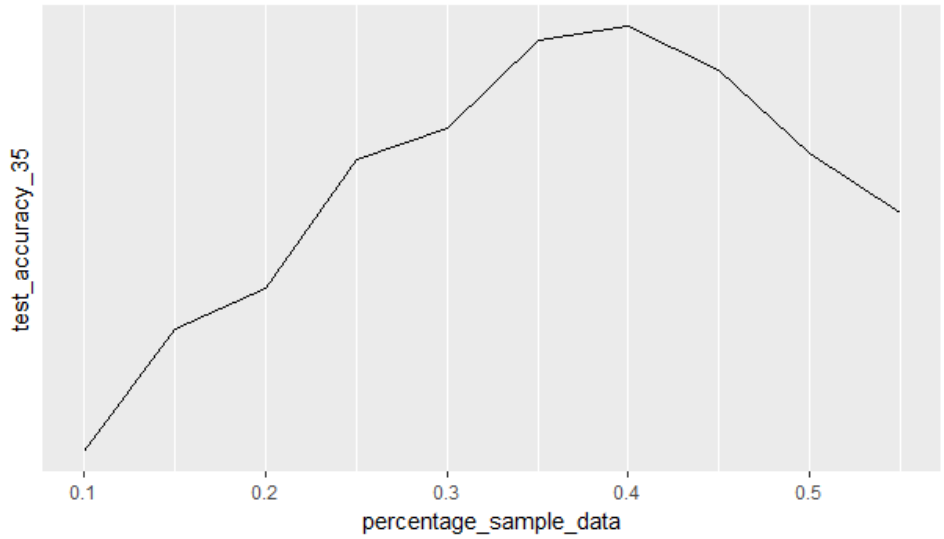
- Graph is shown with a lot of spikes going high and low for every increase in sample size but upon further investigation we can see that since the final value is higher than the initial value, we can safely say that the accuracy increases with increase in the sample size.

Training Accuracy for 3/5 dataset vs sample size



- Starts off at a very low accuracy rate for the smallest sample size but gradually increases and drops at some points but we can see that whatever minima the graph reaches, it is higher than the initial value so accuracy tends to increase for this dataset as the sample size increases.

Testing Accuracy for 3/5 dataset vs sample size



- This can be clearly seen as the accuracy slowly increases and reaches a peak at 40 % sample size but then starts decreasing at a similar rate. For the sample sizes that I have taken, I think the accuracy is increasing.

Final observation: Accuracy tends to increase with an increase in the sample size however there might not be a significant increase as observed in various other sizes in between the lowest and the maximum size.

References:

- https://datascienceplus.com/perform-logistic-regression-in-r/
- https://www.r-bloggers.com/logistic-regression-with-r-step-by-step-implementation-part-2/
- http://diffsharp.github.io/DiffSharp/examples-gradientdescent.html
- http://ethen8181.github.io/machine-learning/linear_regression/linear_regession.html
- https://stats.stackexchange.com/questions/65244/how-to-determine-the-accuracy-of-logistic-regression-in-r
- https://www.r-bloggers.com/evaluating-logistic-regression-models/
- https://rpubs.com/jpmurillo/153750
- https://stackoverflow.com/questions/22906804/matrix-expression-causes-error-requires-numeric-complex-matrix-vector-arguments