CSC 242 Project 4

Elana Chen-Jones

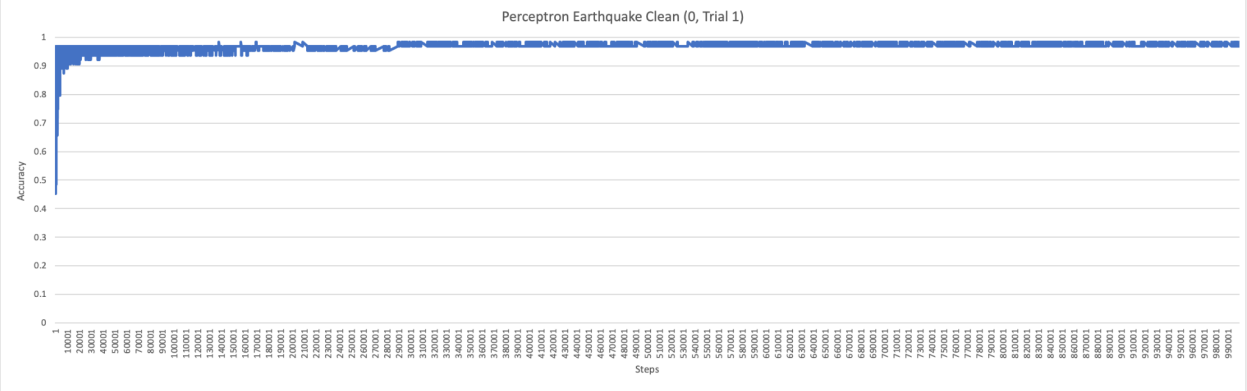
echenjon@u.rochester.edu

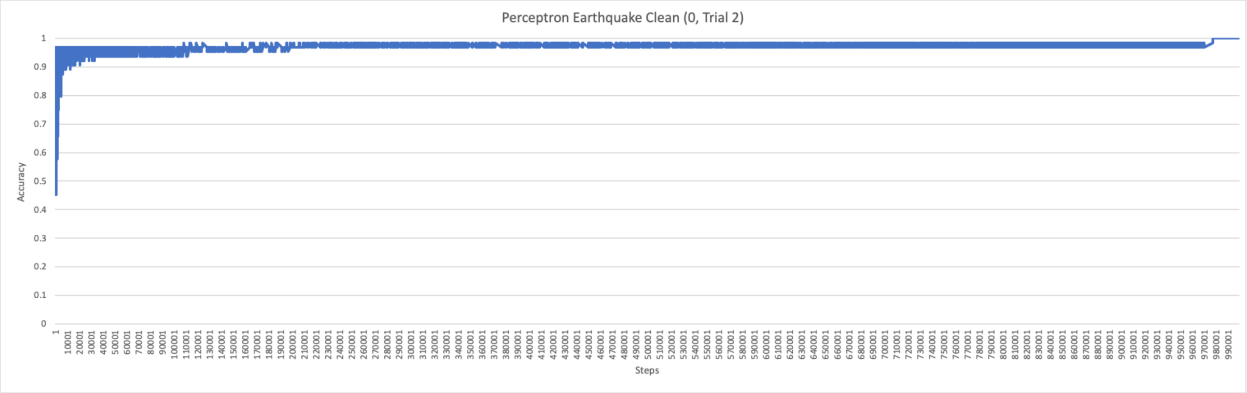
Project Report

As a part of my arguments, the program gives the option to either try a decaying learning rate or a constant learning rate. I ran four trials for each classifier on each example file: two with a decaying learning rate, and two with a constant learning rate. In my first set of trials (perceptron classifier, clean earthquake data), I found that to guarantee convergence at the optimal accuracy, you had to use a constant learning rate. In one of the two decaying learning rate trials, it converged to 1 at the very end, but upon further testing, it seemed to be a rare occurrence. The constant learning rate converged faster, but with a large range of accuracy for a longer time than with the decaying learning rate. In the second set of trials (perceptron classifier, noisy earthquake data), I found that the constant learning rate was much worse on average, as it consistently maintained a large range of accuracy, while the decaying learning rate had a much smaller range. Between the two sets of trials, the noisy earthquake data was much more varied in its accuracy, and never managed the same levels of accuracy as the clean data. With the third set of trials (logistic classifier, clean earthquake data), the results of the decaying learning rate were similar to the perceptron classifier, but the accuracy had pretty much evened out by the end. The constant learning rate managed to converge on 1 at the end, but took much longer to get stable than with the perceptron classifier. In the fourth set of trials (logistic classifier, noisy earthquake data) the decaying learning rate maintained a small range of accuracy, and the constant learning rate maintained a larger range of accuracy throughout. Neither learning rate reached the same level of accuracy as the clean data. The fifth set of trials (perceptron classifier, votes data) had an interesting difference between the two learning rates. The decaying learning rate rapidly increased in accuracy and appeared to be becoming stable, before resuming a greater range of accuracy for the remaining tests, whereas the constant learning rate maintained its range of accuracy throughout the tests. Both rates reached similar levels of accuracy, and neither ever evened out. In contrast, the final set of trials (logistic classifier, votes data) became stable very quickly, with very little range in accuracy after the first jump, and both learning rates neared an optimal level of accuracy. (As a note, the y-axis in the last graph goes from 0.8 to 1, not 0 to 1 as in the others.)

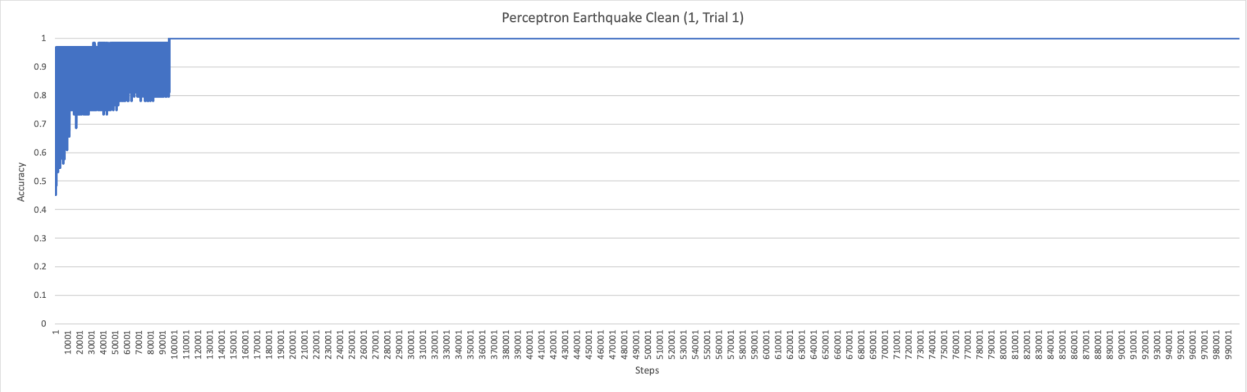
Trial Set 1 (Perceptron Classifier, Clean Earthquake Data):

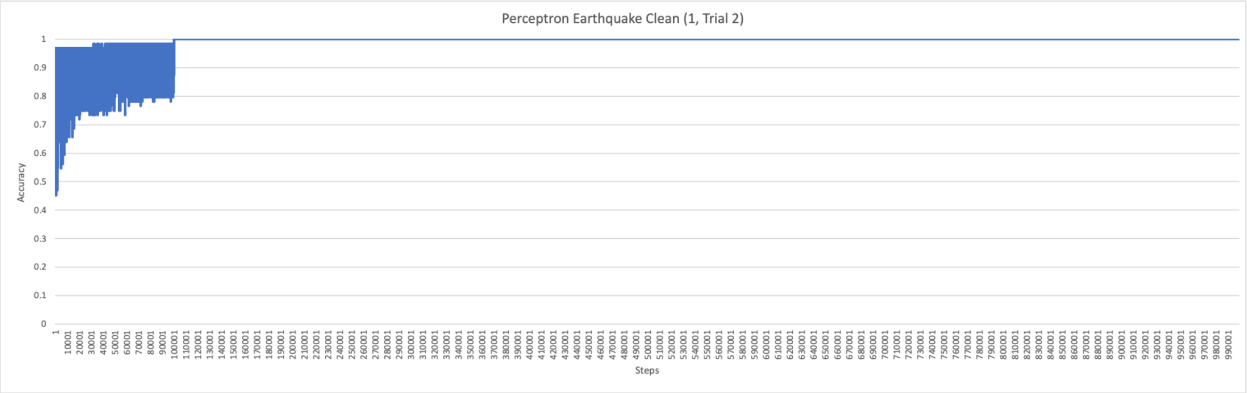
Command: `java LinearClassifier p ../earthquake-clean.data.txt 0`





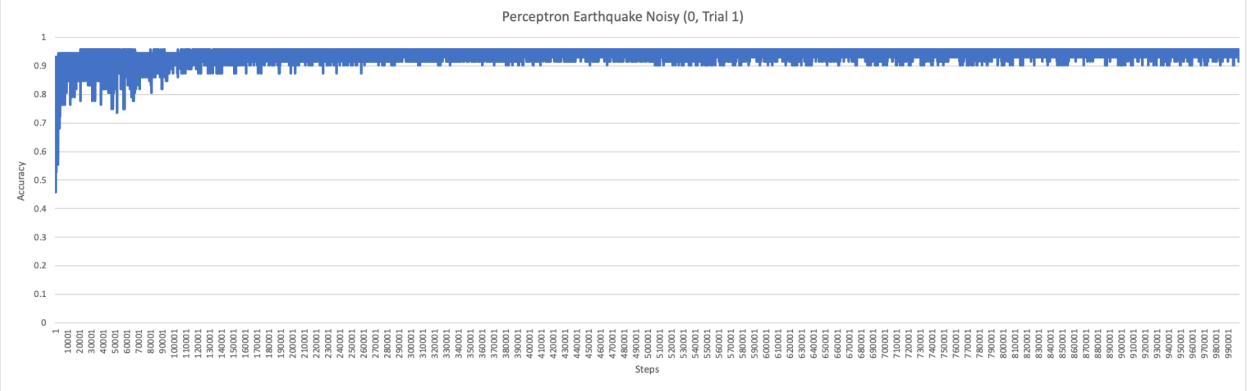
Command: `java LinearClassifier p ../earthquake-clean.data.txt 1`

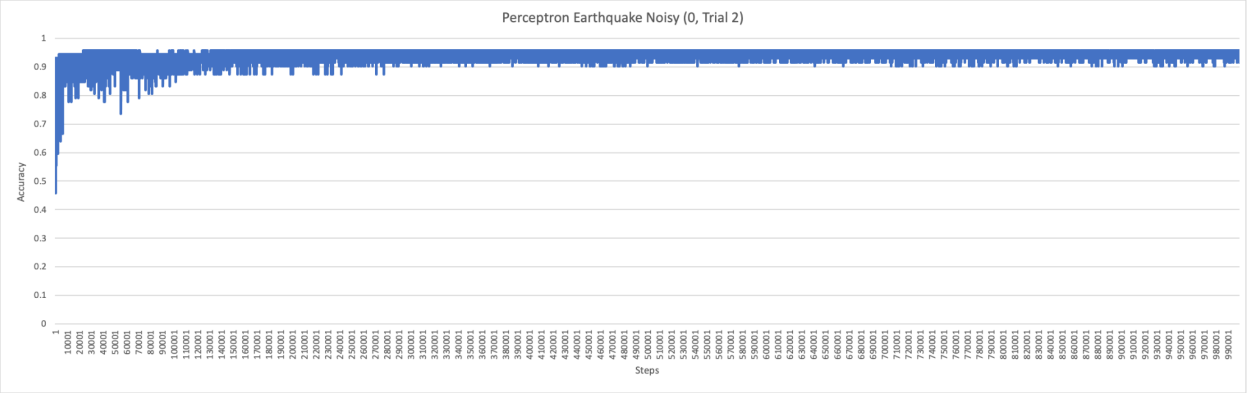



Perceptron Earthquake Clean (1, Trial 2)
Accuracy
Steps

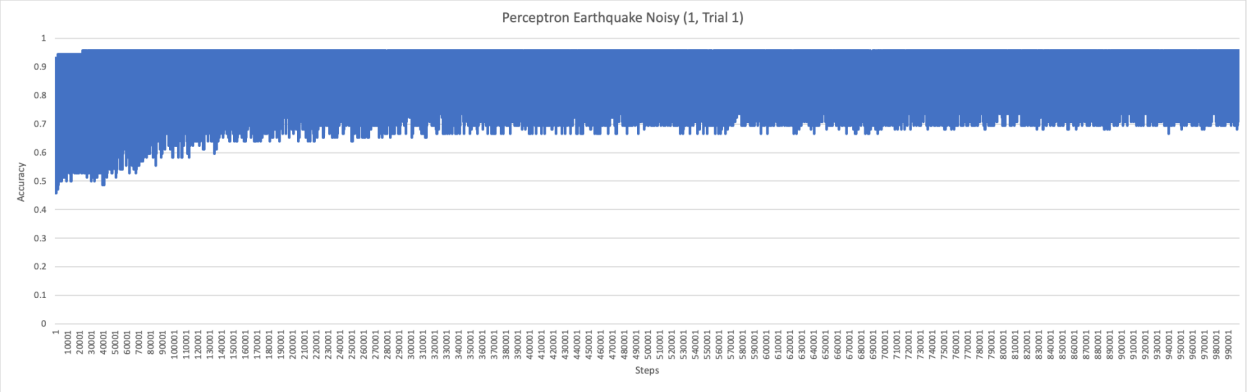
Trial Set 2 (Perceptron Classifier, Noisy Earthquake Data):

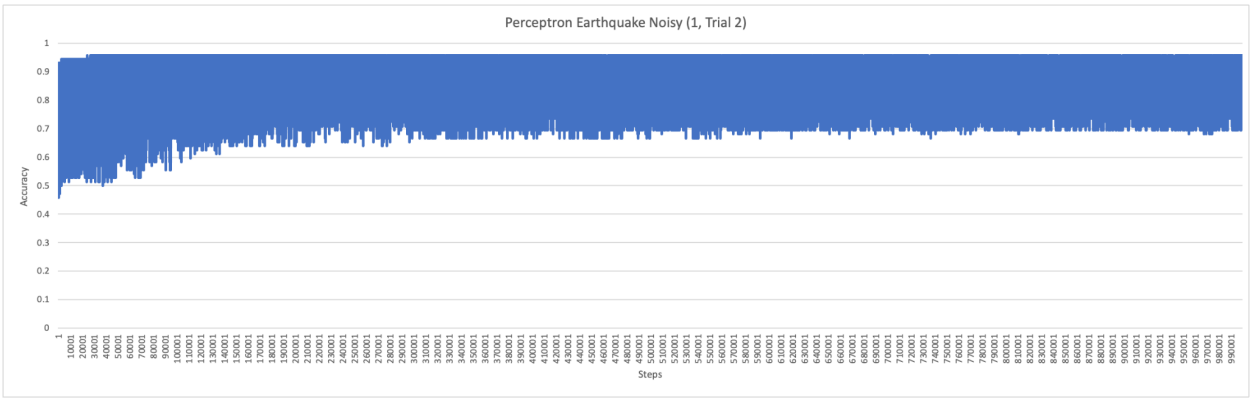
Command: `java LinearClassifier p ../earthquake-noisy.data.txt 0`





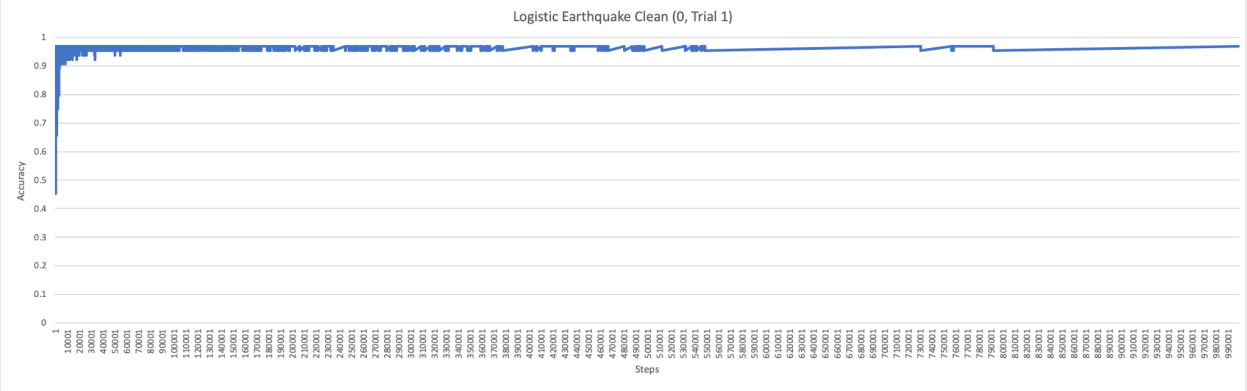
Command: `java LinearClassifier p ../earthquake-noisy.data.txt 1`

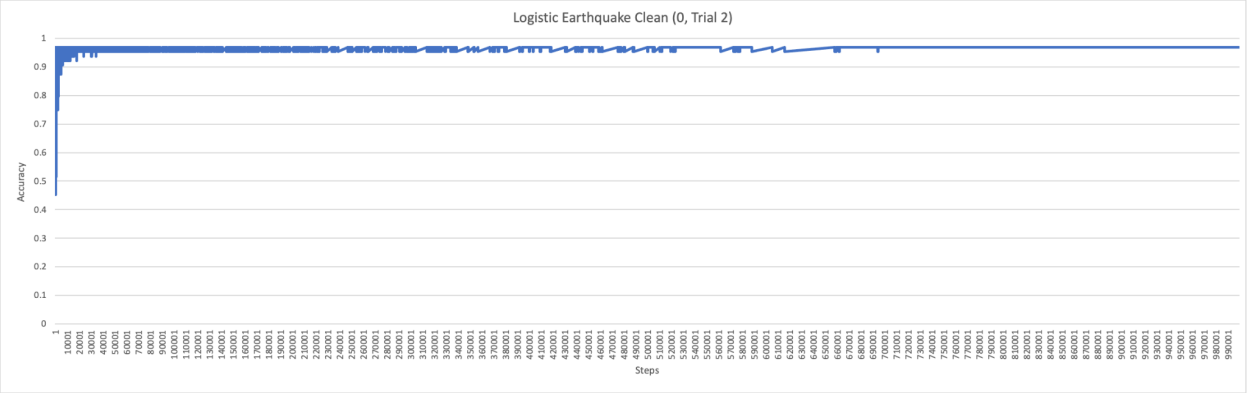



Perceptron Earthquake Noisy (1, Trial 2)
Accuracy
Steps

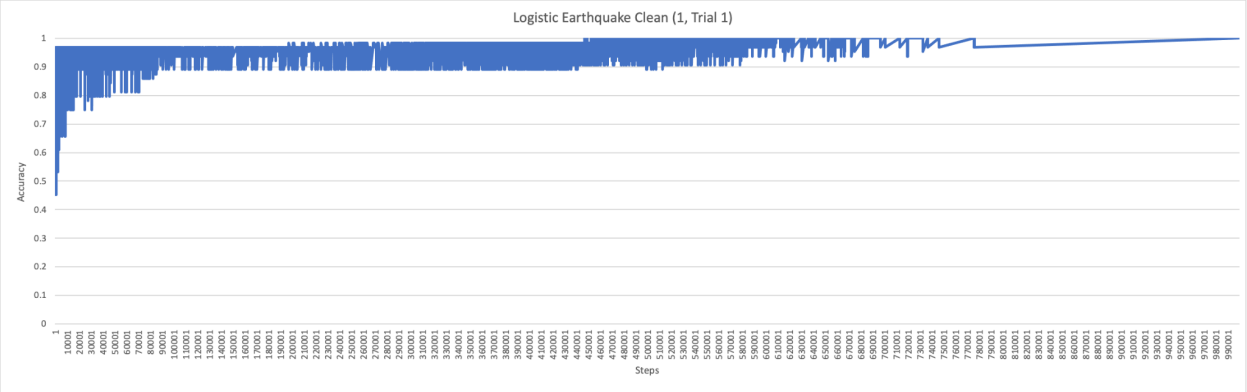
Trial Set 3 (Logistic Classifier, Clean Earthquake Data):

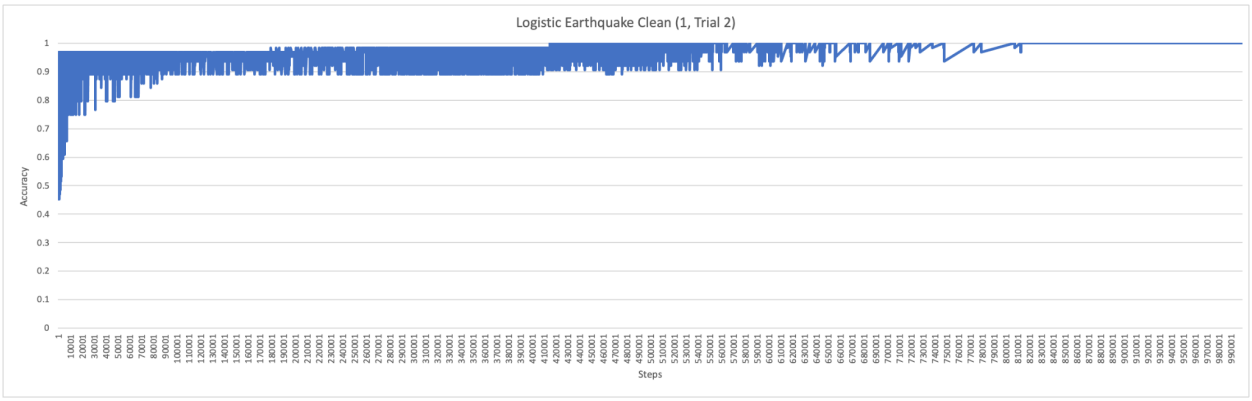
Command: `java LinearClassifier l ../earthquake-clean.data.txt 0`





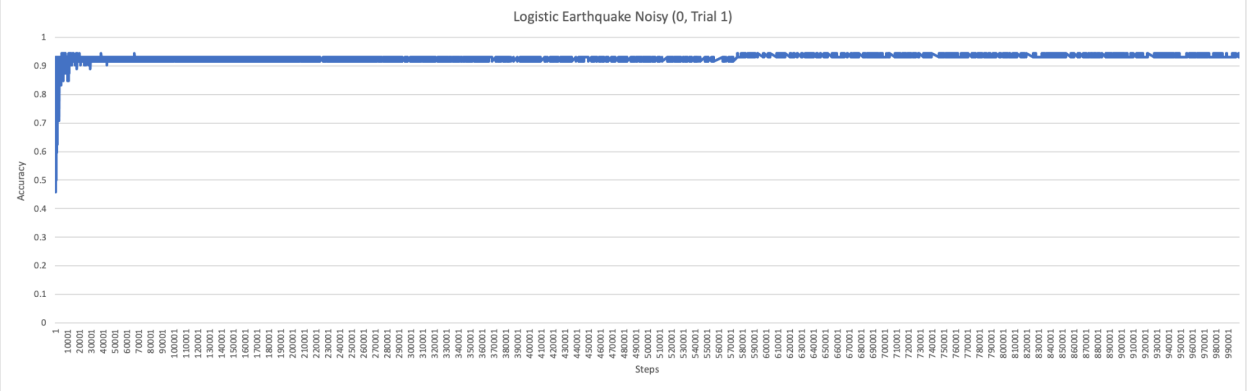
Command: `java LinearClassifier l ../earthquake-clean.data.txt 1`

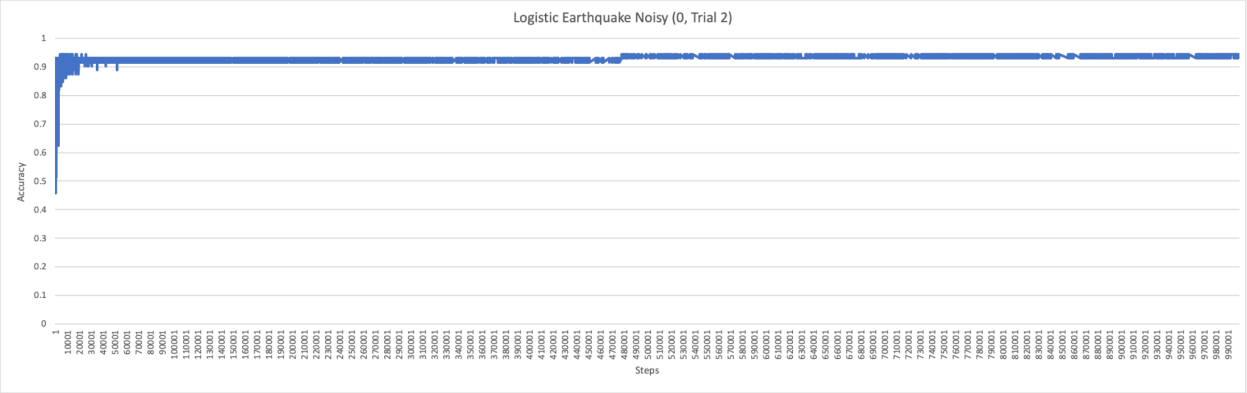



Logistic Earthquake Clean (1, Trial 2)
Accuracy
Steps

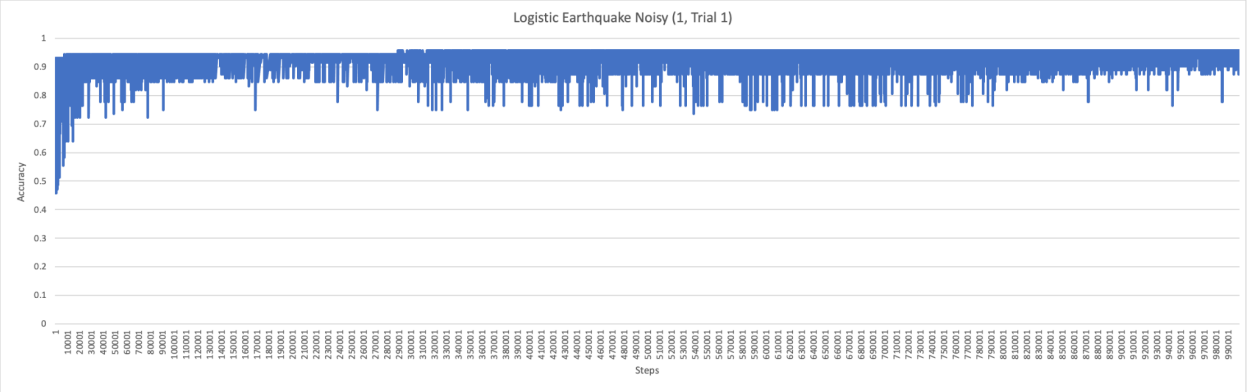
Trial Set 4 (Logistic Classifier, Noisy Earthquake Data):

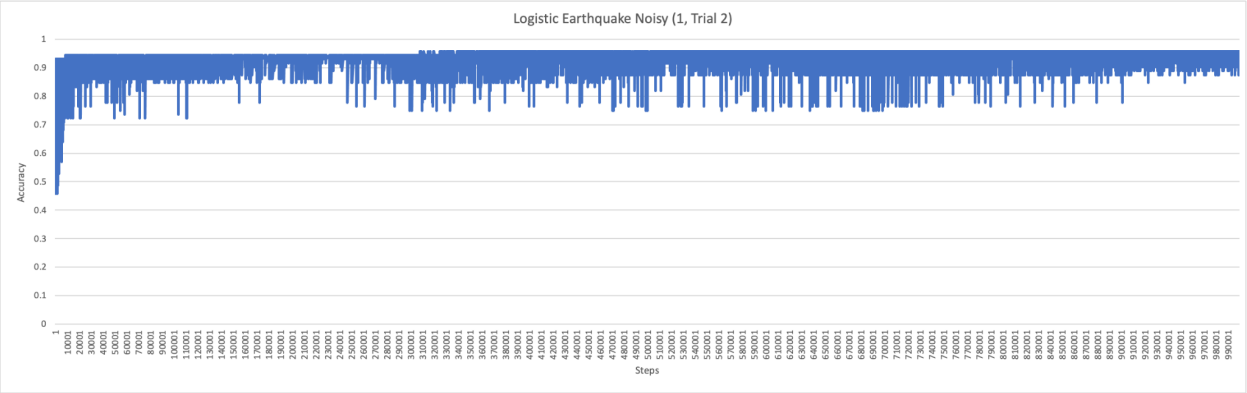
Command: `java LinearClassifier l ../earthquake-noisy.data.txt 0`





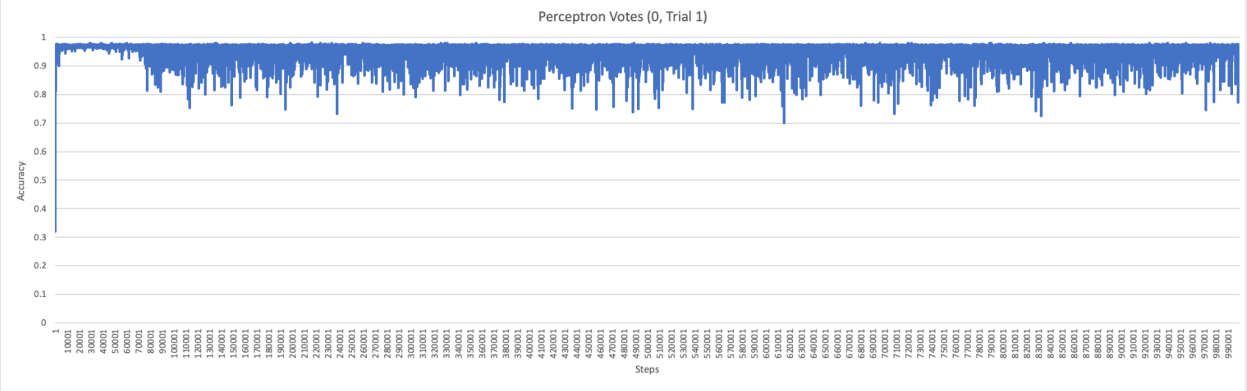
Command: `java LinearClassifier l ../earthquake-noisy.data.txt 1`

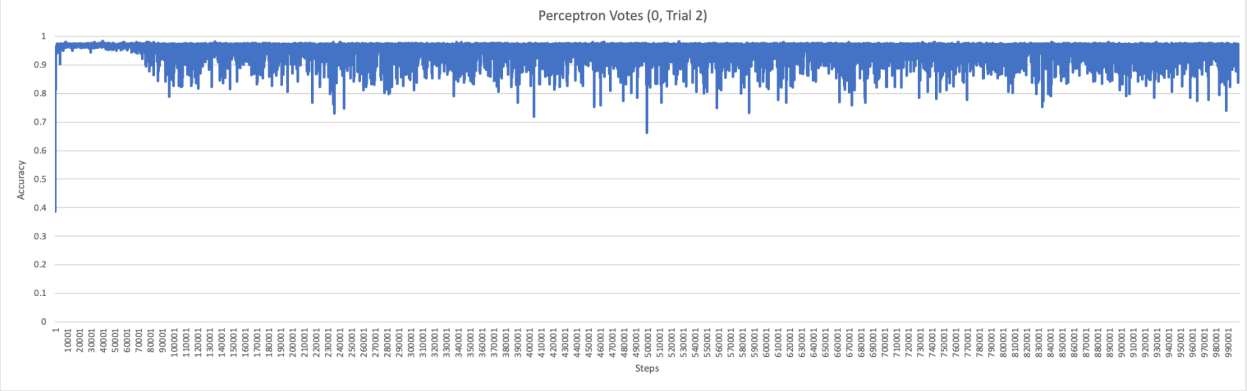



Logistic Earthquake Noisy (1, Trial 2)
Accuracy
Steps

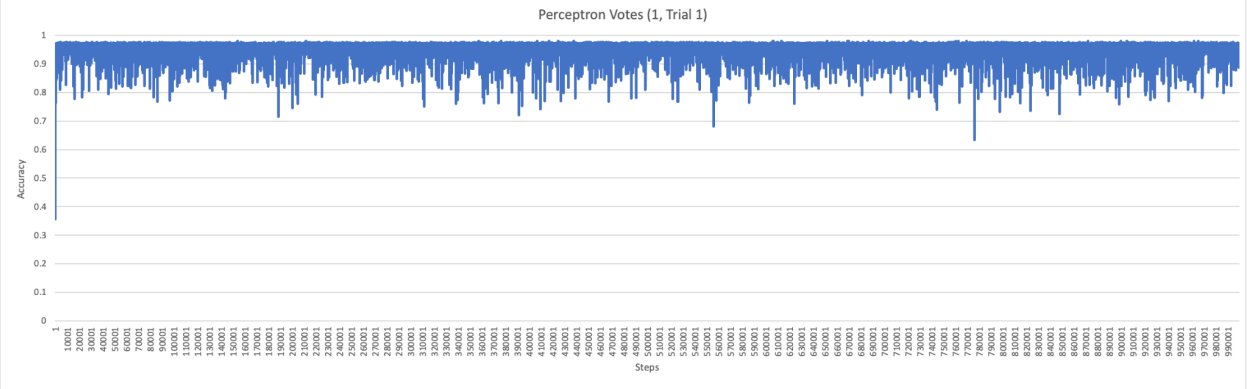
Trial Set 5 (Perceptron Classifier, Votes Data):

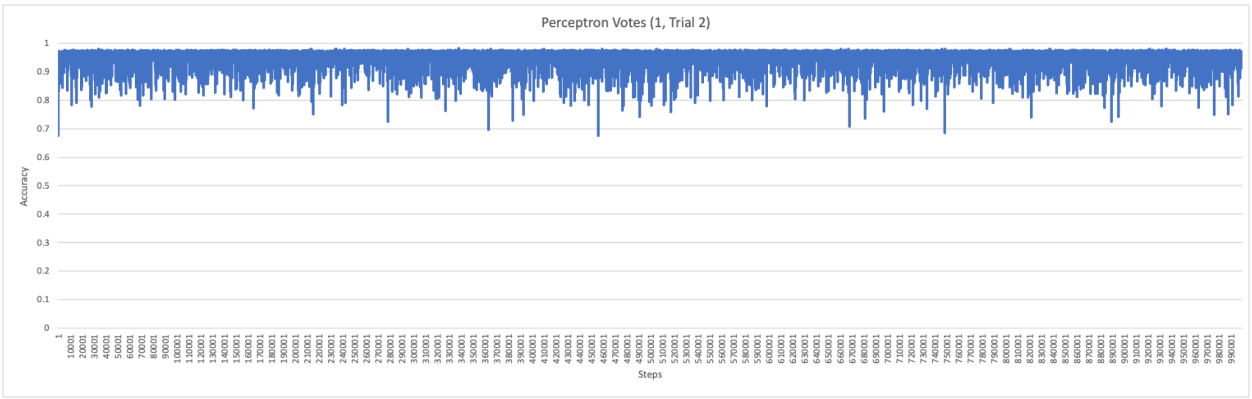
Command: `java LinearClassifier p ../house-votes-84.data.num.txt 0`





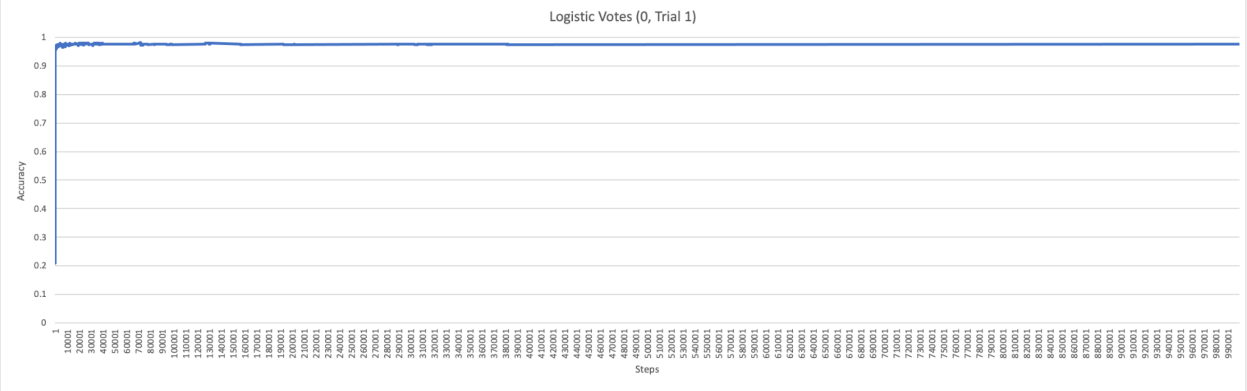
Command: `java LinearClassifier p ../house-votes-84.data.num.txt 1`

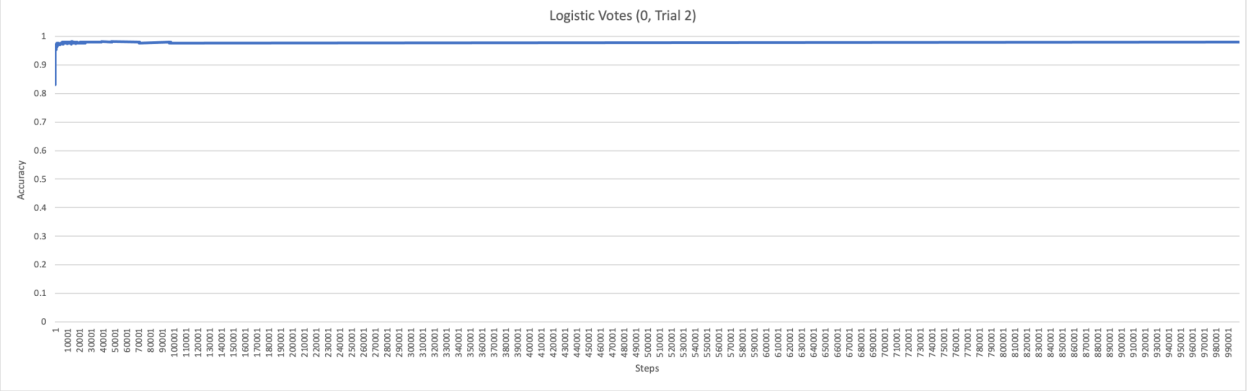



Perceptron Votes (1, Trial 2)
Accuracy
Steps

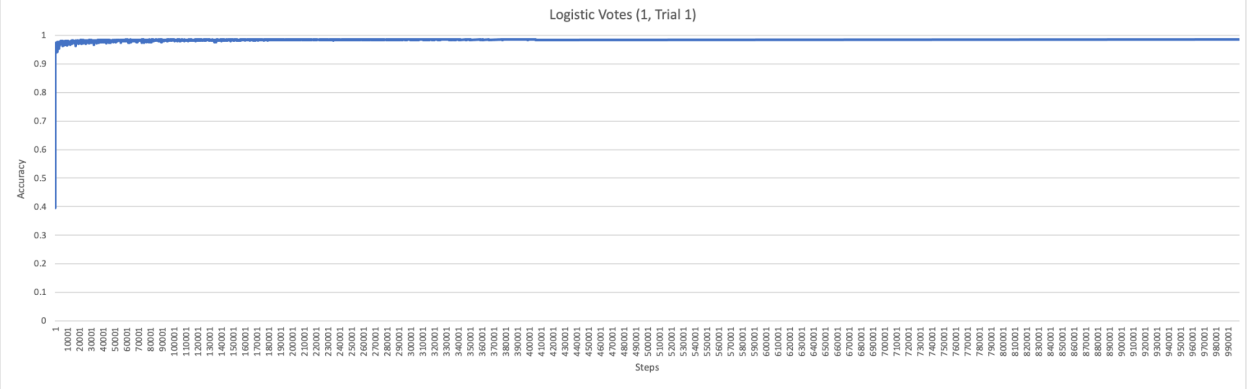
Trial Set 6 (Logistic Classifier, Clean Earthquake Data):

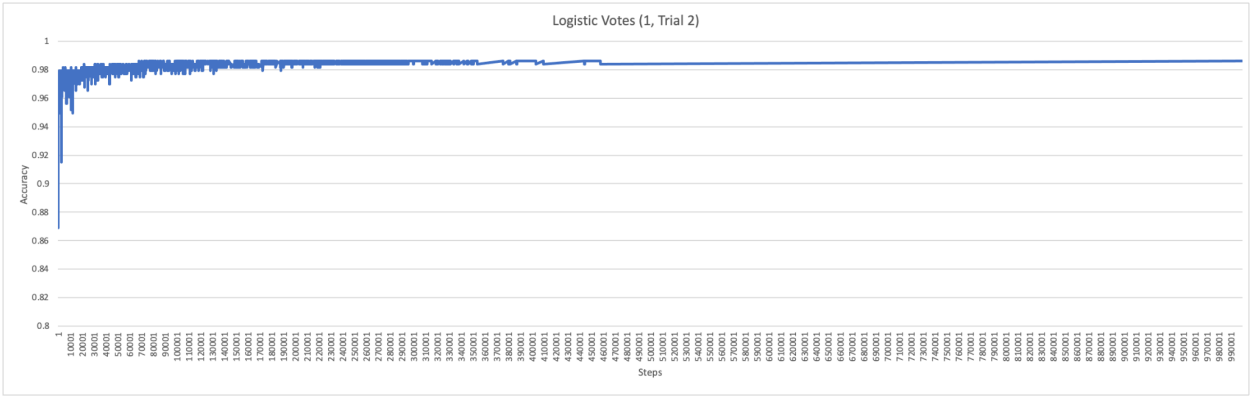
Command: `java LinearClassifier l ../house-votes-84.data.num.txt 0`





Command: `java LinearClassifier l ../house-votes-84.data.num.txt 1`




Logistic Votes (1, Trial 2)
Accuracy
Steps
1
0.98
0.96
0.94
0.92
0.9
0.88
0.86
0.84
0.82
0.8