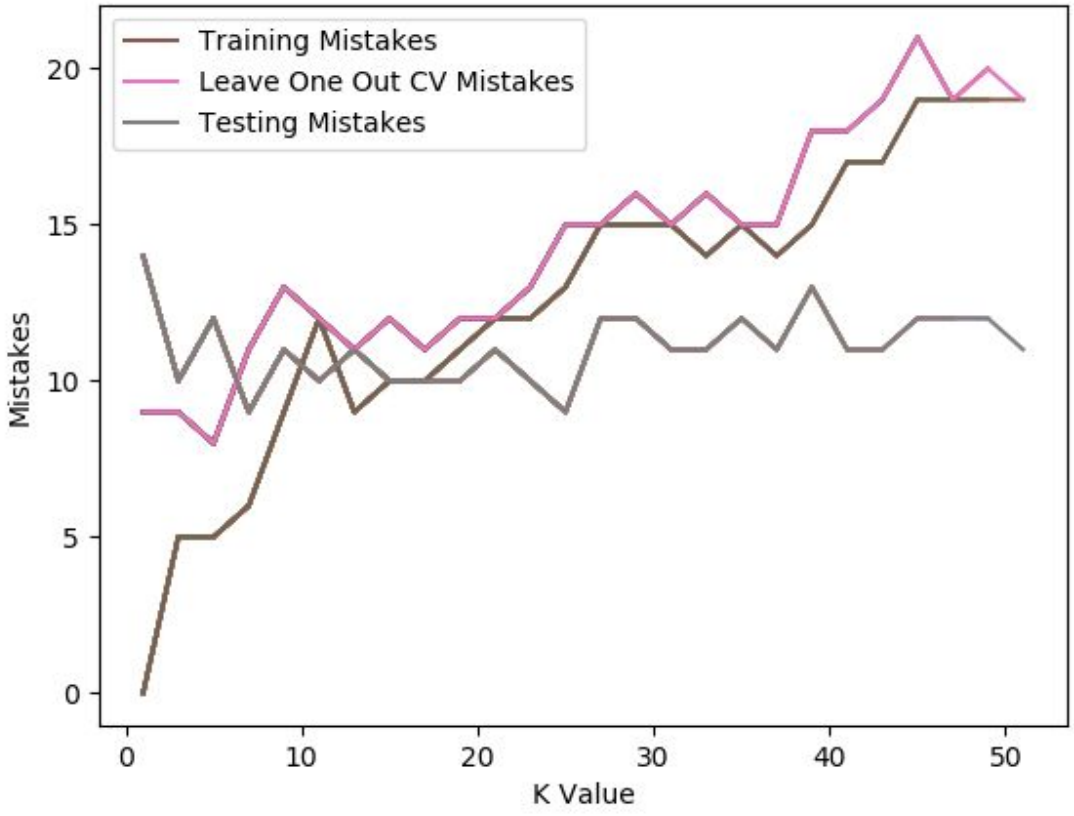
Team Members: Trevor Swope, William Selbie
Language: Python

Part One

Implement the K-nearest neighbor algorithm, where K is a parameter.

Test your KNN with the following range of K values: 1, 3, 5,..., 51. (This is a suggested range, feel free to explore more possible K values). For each possible value of K, please compute the following: 1) the training error (measured as the number of mistakes) 2) the leave-one-out cross-validation error on the training set; and 3) the number of errors on the provided test data. Plot these three errors as a function of K.



Discuss what you observe in terms of the relationship between these three different measure of errors. Perform model selection. What is your choice of K?

All three measures of error continue to increase as k does, which makes sense because as k increases, each point that votes is less and less similar to k. As can be imagined the number of mistakes is lowest for low ks on the training set, starting with 0 mistakes as when k=1, the nearest neighbour is the point itself in the training set. The best k appears to be somewhere between 5 and 9, most likely 7, as it is after this value that the mistakes continue to increase with k.

Part Two

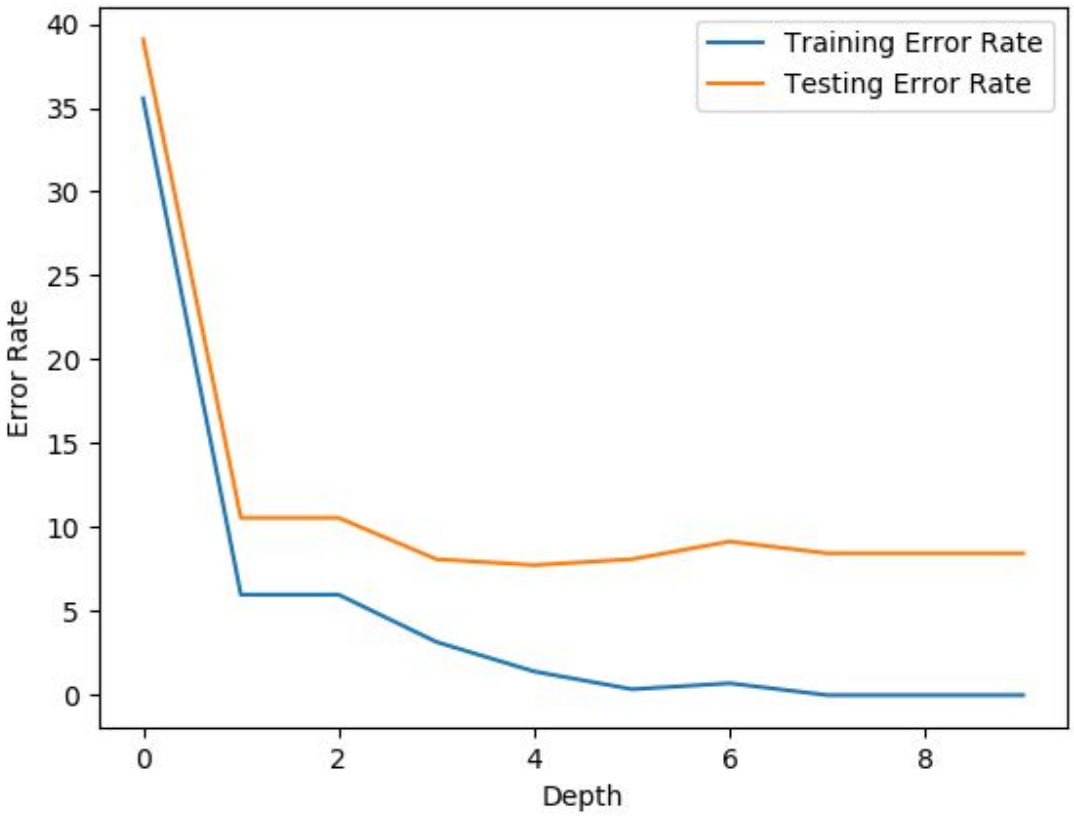
Implement the algorithm for learning a decision stump, i.e. a decision tree with only a single test. To build a decision stump, simply apply the top down decision tree induction algorithm to select the root test and then stop and label each of the branches with its majority class label. For this assignment, please use the information gain as the selection criterion (i.e., using entropy to measure uncertainty) for building the decision stump and consider only binary splits. In your report, please provide 1) the learned decision stump (be sure to provide the label for each branch); 2) the computed information gain of the selected test, 3) the training and testing error rates (in percentage) of the learned decision Stump.

The learned decision stump is the Decision Tree when depth = 1, ie there is only a single threshold node, and two leaf nodes. The threshold is checking whether the 23th feature is <= 0.368380757943, which corresponds to -1, if it is above, then this results in 1. The information gain is 0.1806 for the decision stump, with a training error of 5.99% and a testing error of 10.56%.

Implement the top-down greedy induction algorithm using the information gain criterion for learning a decision tree from the training data with a xed depth limit given by d (note when d = 1 this is the decision stump). Consider the following choices for d (1, 2, 3, 4, 5, 6) Similarly we will only consider binary splits. Please apply the learned decision trees to the training and testing data and report in a table the training and testing error rates of the learned decision trees for different d values. Plot the error rates as a function of d. What behavior do you observe? Provide an explanation for the observed behavior.

Depth	Training Error %	Testing Error %
1	5.99	10.56
2	5.99	10.56

3	3.16	8.10
4	1.41	7.74
5	0.35	8.10
6	0.7	9.15
7	0.0	8.45



As can be seen in the plot, the training and error rates go down as depth increases, although the error rate for testing error plateaus as the tree overfits to the point of correctly classifying each data point for the training data. At a depth of 7, the training error rate hits 0.0% because there are enough split to correctly classify each point. The testing rate is best at a depth of 4, most likely because this is before the data starts overfitting.