CS434 Assignment #2 Report

Group: Tsewei Peng

Part I: Model Selection For KNN

Note: Model for normalization: (Xi - Xmin) / (Xmax - Xmin)

Problem 1.

K-th nearest neighbor algorithm implemented inside knn.py

Problem 2.

Figure 1. The number of errors of knn algorithm in relation to number k using

1. Training data
2. Testing data
3. Leave-one-out cross validation

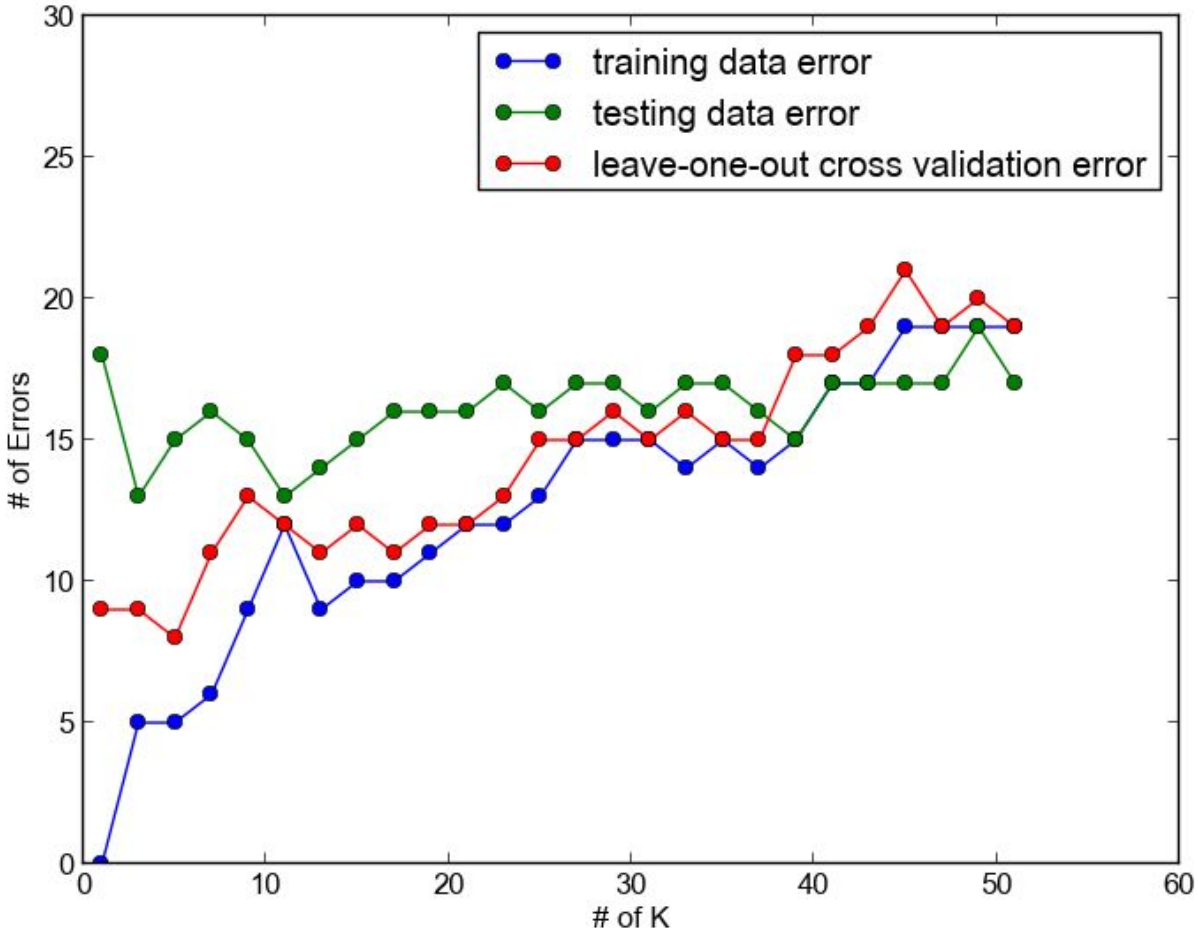
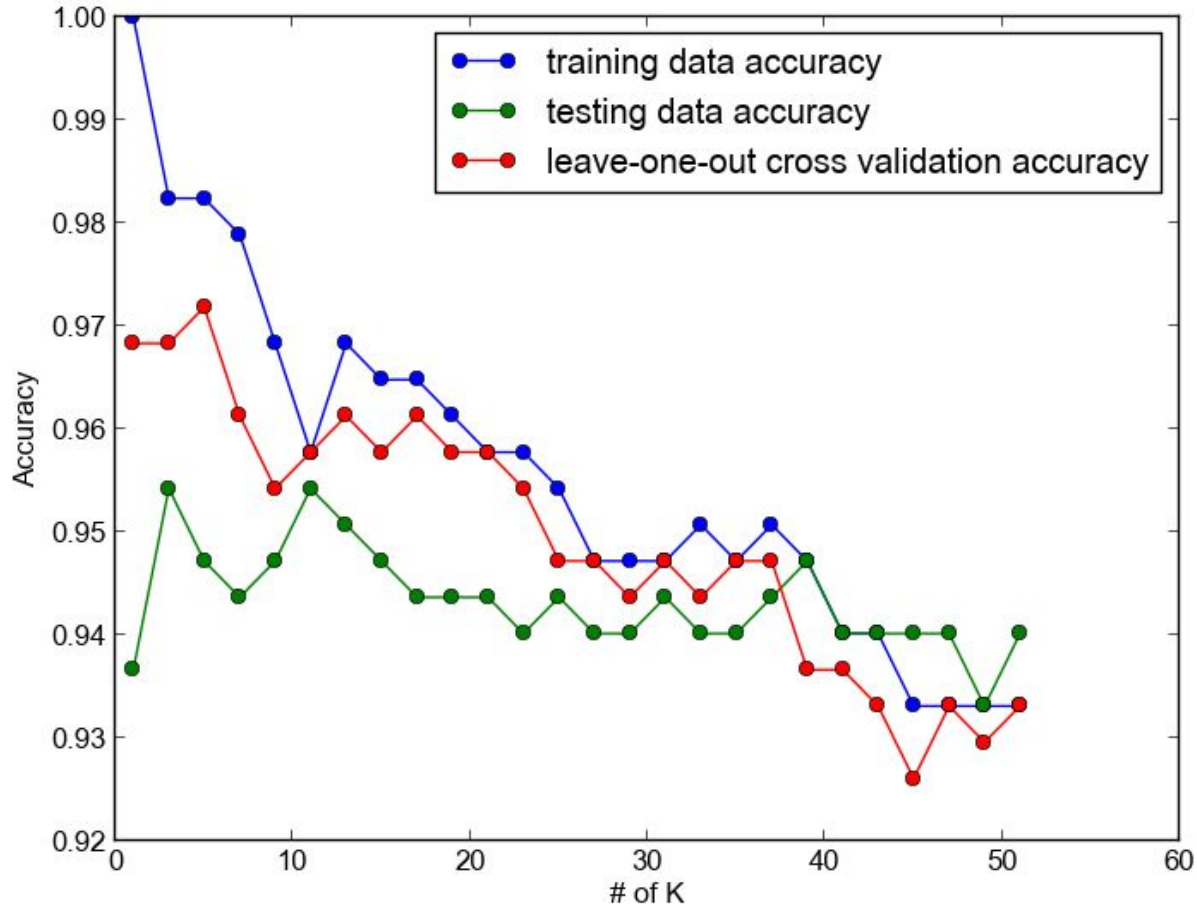


Figure 2. The accuracies of knn algorithm in relation to number k using:

1. Training data
2. Testing data
3. Leave-one-out cross validation



Problem 3.

For training data, initially when k is 1, the training does not produce any errors, which is because a training sample's nearest neighbor is always itself. As the number k increases, the number of errors increases and the accuracy decreases. Leave-one-out cross validation starts out making some mistakes, but as k increases, the number of errors converges with the number of training data errors. This is because in leave-one-out cross validation, one of the training sample is left out as testing data. In terms of testing data errors, initially testing

data have the most errors, but as the number of k increases, the testing error rate decreases, and then increases to around its initial number.

Part II: Decision Tree

Note: Minimum size before node classification (making the node a leaf node): 5

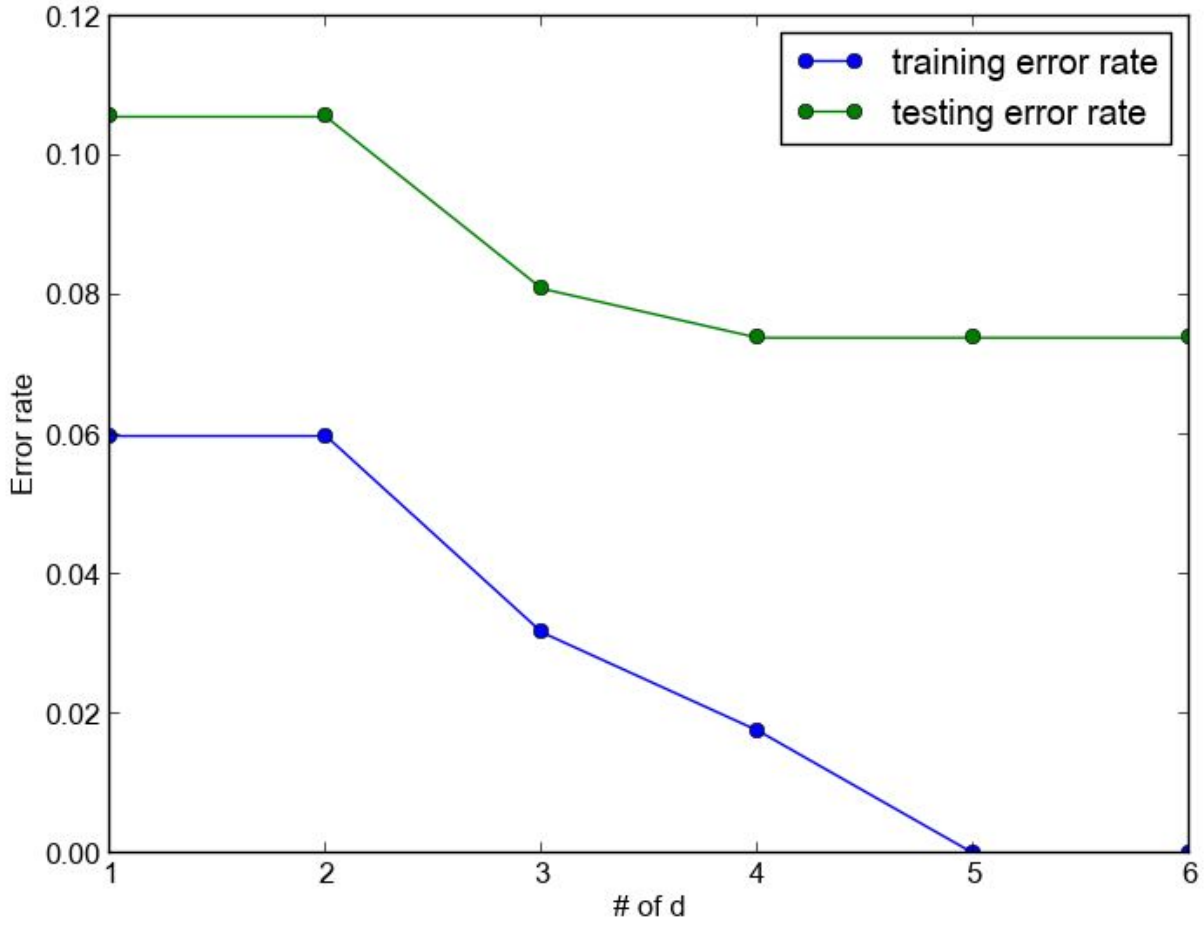
Problem 1.

Decision stump is a decision tree with depth 1.
Learned decision stump:
Feature 23 < 115.7: (Entropy: 0.94, Gain: 0.64)
--Predict Y = -1 (Entropy: 0.42)
Feature 23 >= 115.7: (Entropy: 0.94, Gain: 0.64)
--Predict Y = 1 (Entropy: 0.0)

Information gain for this decision stump: 0.643536804675
Training error rate: 0.059859154929577496
Testing error rate: 0.10563380281690138

Problem 2.

d value (depth)	Training Error Rate	Testing Error Rate
1	0.059859154929577496	0.10563380281690138
2	0.059859154929577496	0.10563380281690138
3	0.03169014084507038	0.08098591549295775
4	0.017605633802816878	0.073943661971831
5	0	0.073943661971831
6	0	0.073943661971831



The training error rate significantly decreases to 0 as the depth of the decision tree increases. This is because as depth increases, the tree's branches fits more to the data it is trained on. At depth level 5, the training error rate is 0.

The Testing error rate is high when the depth value is 1. It then decreases when depth goes from 2 to 4, but stays the same from depth level 4 to 6. The testing error rate decreases because the tree's branches are more specialized. I cannot explain why the testing error rate stops decreasing after a certain point, I can only make assumptions. I think it is because either the testing error has some noise, or that the decision overfitted the training data after depth level 4.