Assignment 3 Report - KNN and Decision Trees

Helena Bales and Natalie Suderman

CS343 - Spring 2017

May 1, 2017

Contents

1 Introduction

2 Model Selection for KNN

2.1 Implementing the K-Nearest Neighbors Algorithm

The K-Nearest Neighbors Algorithm is implemented in Python in the file knn.py

In implementing the KNN algorithm, we first read in the datat from the provided csv. The data is in the form of an Nx31 dimentional matrix. The dataset constitutes 30 features with the first value being the true class label. For example:

1, 19.59, 25, 127.7, 1191, 0.1032, 0.09871, 0.1655, 0.09063, ..., 0.2293, 0.06091

In this case, the true class value is 1, meaning that it is malignant. If it were benign, the true class value would be -1.

The implementation reads in each line of the csv, removes the first value as the true class value then uses the following 30 features to determine the distance between data points. Before the distance between two points can be calculated, the scale for each feature must be determined and the feature scaled to be all in the same range (for example [0, 1]).

We can use the Euclidian or straight line distance to determine the distance between two points x and x_i with m features.

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}, x_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{im} \end{bmatrix}$$

$$D(x, x_i) = ||x - x_i||$$

$$D(x, x_i) = \sqrt{(x - x_i)^T (x - x_i)}$$

$$D(x, x_i) = \sqrt{\sum_{j=1,\ldots,m} (x_j - x_{ij})^2}$$

For each line in the csv, its k nearest neighbors will be used to determine its value.

2.2 Evaluation of Error with Respect to K

2.2.1 Computed Training Error

The following list are the computed training error for various k values.

- **k=1**: 0.355633802817
- **k=3**: 0.644366197183
- **k=5**: 0.355633802817
- **k=7**: 0.355633802817
- **k=15**: 0.352112676056
- **k=21**: 0.355633802817
- **k=37**: 0.355633802817
- **k=43**: 0.355633802817
- **k=51**: 0.362676056338

2.2.2 Computed Leave-One-Out Cross-Validation Error

- **k=1**: 29
- **k=3**: 22
- **k=5**: 23
- **k=7**: 27
- **k=15**: 15
- **k=21**: 24
- **k=37**: 17
- **k=43**: 21
- **k=51**: 19

2.2.3 Number of Errors on Provided Test Data

- **k=1**: 183
- **k=3**: 183
- **k=5**: 101
- **k=7**: 139
- **k=15**: 101
- **k=21**: 100
- **k=37**: 151
- **k=43**: 101
- **k=51**: 100

2.2.4 Plot of Errors as a Function of K

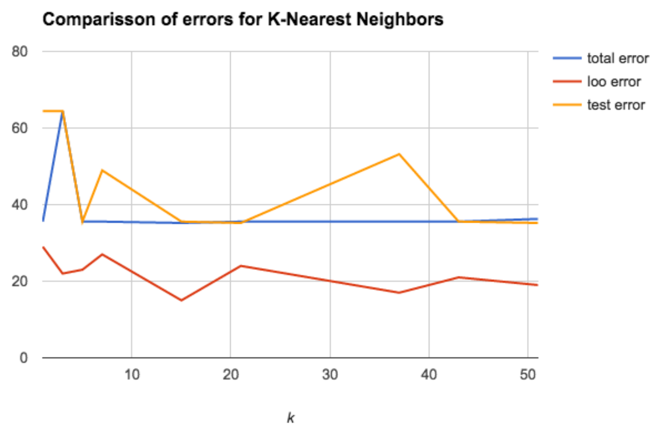


Figure 1: Training error, Leave-one-out Cross-validation error, and Testing error for various K values

2.3 Evaluation of Different Measures of Error and Model Selection

Based on the lecture, the leave-one-out cross-validation method of measuring error is the most informative. Given this, we should make the decision of which k value to use based on all three of the error values, however we should consider the leave-one-out error to be the most important.

From looking at the graph we can see that the lowest point in the leave-one-out error occurs when $k = 15$. At this point, the leave-one-out error is at its global minimum of 15 and the training and testing errors are at their local minimum values of approximately 35. Given this, we should select k to be 15.

3 Decision Tree

3.1 Implementing Learning Decision Tree Stump

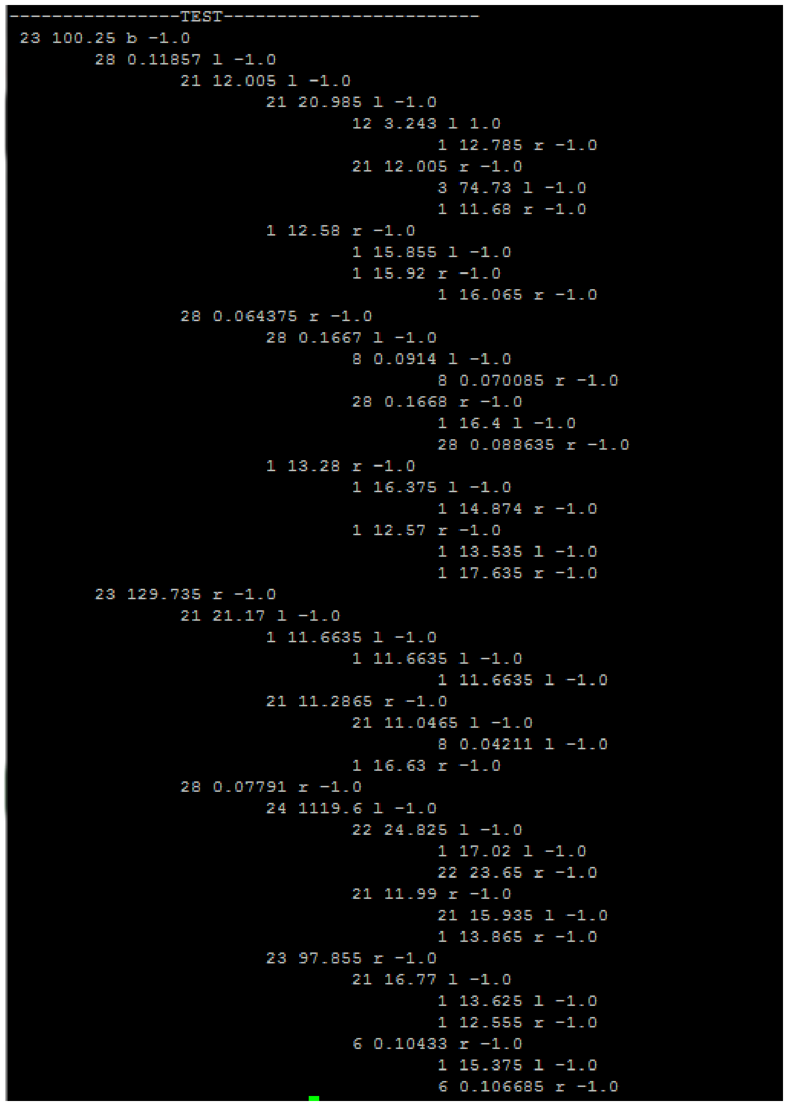
Best feature: 23 Best Split value: 100.25 Information Gain: .4272 Error: .3556 Test Error: .3908 Base Branch Label: -1.0 Left Branch Label: -1.0 Right Branch Label: 1.0

3.2 Implementing Top-Down Greedy Induction for Learning Decision Tree

Total training error:
0.557802806034
Total testing error:
0.645376637757

Since the decision tree is built using the training data, it makes sense that the test data would not be as fit to the decision tree, as the training data is.

Learned Decision Tree:
values are ordered: best feature, split value, and direction of split, and class the node/branch belongs to.



```
----------------TEST------------------------
 23 100.25 b -1.0
        28 0.11857 l -1.0
                21 12.005 l -1.0
                        21 20.985 l -1.0
                                12 3.243 l 1.0
                                        1 12.785 r -1.0
                                21 12.005 r -1.0
                                        3 74.73 l -1.0
                                        1 11.68 r -1.0
                        1 12.58 r -1.0
                                1 15.855 l -1.0
                                1 15.92 r -1.0
                                        1 16.065 r -1.0
                28 0.064375 r -1.0
                        28 0.1667 l -1.0
                                8 0.0914 l -1.0
                                        8 0.070085 r -1.0
                                28 0.1668 r -1.0
                                        1 16.4 l -1.0
                                        28 0.088635 r -1.0
                        1 13.28 r -1.0
                                1 16.375 l -1.0
                                        1 14.874 r -1.0
                                1 12.57 r -1.0
                                        1 13.535 l -1.0
                                        1 17.635 r -1.0
        23 129.735 r -1.0
                21 21.17 l -1.0
                        1 11.6635 l -1.0
                                1 11.6635 l -1.0
                                        1 11.6635 l -1.0
                        21 11.2865 r -1.0
                                21 11.0465 l -1.0
                                        8 0.04211 l -1.0
                                1 16.63 r -1.0
                28 0.07791 r -1.0
                        24 1119.6 l -1.0
                                22 24.825 l -1.0
                                        1 17.02 l -1.0
                                        22 23.65 r -1.0
                                21 11.99 r -1.0
                                        21 15.935 l -1.0
                                        1 13.865 r -1.0
                        23 97.855 r -1.0
                                21 16.77 l -1.0
                                        1 13.625 l -1.0
                                        1 12.555 r -1.0
                                6 0.10433 r -1.0
                                        1 15.375 l -1.0
                                        6 0.106685 r -1.0
```

Figure 2: Learned decision tree