K Nearest Neighbors Algorithm

Abstract

The K-nn algorithm requires a set of stored records. By calculating distance between records based on distance metric, classify the unknown records based on the nearest k number of neighbors. And it is a lazy learner which does not need to generate a model to classify unknown records. Each time we only need to compute distance to other training records and use class labels of nearest neighbors to determine its label (by taking majority vote). As a result, in each classification phase, all records need to be considered.

Algorithm

1. Enter the number k which decides how many neighbors need to be take into account and transfer the nominal feature in to binary fashion. (eg, if only two results, then change the first one to (0,1), the second to (1,0))

2. **for** each fold **do**

3.Compute the distance between the test records and every other example.

4.Choose K closest training examplesand then use majority votes to determine the label. for unknown records.

1. **end for**
2. Compute the mean of accuracy, precision, recall and F1 value from these ten folds

Implementation

In my implementations, I defined two functions to help me normalize the data and calculate the accuracy and other measurements.

I defined the first function called norm\_data(matrix), which takes the original records as the only arguments. It will normalize each feature by method. Thus, all feature values would be in range (0,1). The reason why we do this is because the distance computed from one feature may dominate all other features.

Then I defined the second function called accu\_cal(truth, result), which take the ground truth of test labels and predicted labels to generate all four measurements.

During the preprocessing phase, for convenience, all nominal features are transferred to binary format so that all values from that feature has equal distance.

A linalg.norm() function was used to calculate Euclidean distance. I created four lists for accuracy and other measurements. For each fold, I add these measurements to the list. The final value for these measurements are the mean value of the results from 10 folds.

Results

Pros

1. Simple mechanism.
2. Time saving compared to other complex classification methods
3. Outliers are not sensitive for Knn algorithm

Cons

1. Feature values needs to be normalized to avoid converging to one dominant feature
2. Large amount of data may cause high memory occupation and plenty of calculations
3. The result is sensitive to k value