Harry Zachariou
hcz1
10 November 2017
3 Pages

Coursework 2
Machine Learning K-NN Algorithm

I will explain the methods in the order in which I completed them.

myKnn()

This constructor method of the myKnn class calls the super class knnParent.

determineMinMaxAttributeValues()

The method works by looping though the trainingData at each data set, putting the highest found attributes into the max array and the smallest found attributes into the min array, and sorting the array, this method has an if statement that checks for the class index and categorical and assigns them a vector in the array but sets the values to 0. The method will work for any class attribute position.

normaliseNumericInputAttributesTrainingData()

This method, loops through all instances in m_TrainingData, limited but the number of instances in this case 77, instantiate the instance for the training data, then call the normaliseNumericInputAttributes() method with the training data instance as an argument.

normaliseNumericInputAttributes()

This method loops through the training data at an index, limited by the number of values the instance has. During each loop cycle the attribute is tested if its a numeric attribute and also if its not the class index/output attribute the class attribute is not to be normalised, each attribute is then put through the normalisation equation, the loop counter increments to index the correct max and min values for the respective attribute. Each normalised attribute is put into the array. Each normalised attribute will have a value between 0.0 and 1.0.

euclideanDistance()

This method, loops through each attribute and tests if the attribute is the class attribute to ensure the class attribute isn't computed. If the attribute in question is not numerical then it will compare the double WEKA has stored for each category and if they are different it adds 1 to the sum. Then it adds a running total of each attributes differences squared $(x_1 - x_2)^2$. After the loop the sum is square rooted, which is returned as the calculated euclidean distance.

findNearestNeighbours()

This method returns an Instance array of the K nearest neighbours according to the K value specified. I set up a double array called distances, this array will store the euclidean distances, and I have also set up an instance array called nearest which will store the Instance with the nearest distance to the specified K value. The first loop, stores the euclidean distance of the Instance passed in and the training data, in the distances array, and stores the training data at the loop index in the nearest array. The second loop starts at the K value specified to find the nearest neighbours from K. The euclidean distance of the Instance passed in and the training data is stored in the distance variable. The furthestNearestNeighbourIndex and maxDistance are initialised to be tested against. The next loop will find the max index, if it appears twice as specified in the java docs, it will return the one that occurred first - the smallest index. The variable furthestNearestNeighbour is initialised to the furthestNearestNeighbourIndex, to be used as the index value for the distances array. The if statement tests if the distance variable is smaller than the value of the distances array with the furthestNearestNeighbour index. If true then the index value of the distances array will become the distance variable, and the nearest array at index furthestNearestNeighbour will become the training data at index of the loop. If K is greater than the number of Instances then it will return all the Instances.

determinePredictedOutput()

The algorithm solves for regression problems, and therefore has a numerical output. This method works by looping through the nearest neighbours array, sums them and then takes the average, more specifically the mean, and returns it as the predicted output.

```
=== Run information ===

Scheme:       weka.classifiers.lazy.MyKnn -K 1
Relation:     desharnais.csv-
weka.filters.unsupervised.attribute.Remove-R1,5
Instances:    77
Attributes:   10
              TeamExp
              ManagerExp
              YearEnd
              Transactions
              Entities
              PointsNonAdjust
              Adjustment
              PointsAjust
              Language
              Effort
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

MyKnn classifier using k=1.
Trained on 77 examples.


Time taken to build model: 0 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                  0.4708
Mean absolute error                   2700.3377
Root mean squared error               4077.8434
Relative absolute error                 88.7807 %
Root relative squared error             97.0147 %
Total Number of Instances               77
```