**Random Forest Starter Code**

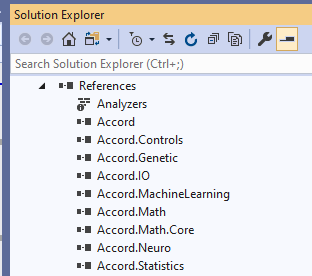
using Accord.MachineLearning.DecisionTrees;

using Accord.Math;

using Accord.Math.Optimization.Losses;

using Accord.Statistics.Analysis;

**References**



// MAIN Section

// Read the CSV file named “**HeartAttackTrainData**.csv “ to populate the variables

// **inputs**: 2d dimensional array data file

// **actualTargetValues**: 1d array of the target value for each observation

// **variables**: an array of DecisionVariable objects, one for each feature

//

double sampleRatio = 0.632;

double coverageRatio = 1;

// Use the data from the training date file to build the random forest model

// Normalize the training data to be between 0 and 1

RandomForestClassifier forest = new RandomForestClassifier(inputs, actualTargetValues, variables, sampleRatio, coverageRatio);

// Now that you have the forest model:

// Read the CSV file “**HeartAttackValidationData.csv** “ to populate the variables. It has the same format.

// **inputs**: 2d dimensional array data file. You can use the same array since you have built the model

// **novelTargets**: 1d array of the target value for each observation

//

// Normalize the training data to be between 0 and 1

// Use the previously created forest model

// Use the validation input

PredictFromRandomForestModel(forest, input, novelTargets)

// For the final section read the data file that does not have target data. Call the function

// “**PredicterOnly**” which will use Random Forest model to predict the outcome based on the /

// observation data.

// Read the CSV file named “**HeartAttackValidationDataNoTarget.csv**“ to populate the variables

// **inputs**: 2d dimensional array data file

// Use the previously created forest model

// Use the validation input

PredicterOnly(forest, input)

// End Main

///////////////////////////////////////////////////////////////////////////////////////

// Methods

public RandomForestClassifier

(double[][] inputData, int[] \_targetValues, DecisionVariable[] variables,

double sampleRatio, double coverageRatio)

{

// 1. I make a local copy of the \_variables array for later use

\_variables = new DecisionVariable[variables.Length];

for (int v = 0; v < variables.Length; v++)

\_variables[v] = variables[v];

// 2. Ensure we have reproducible results

Accord.Math.Random.Generator.Seed = 0;

// 3. Create a RandomForestLearning object named teacher and pass in

// my Decision variables

RandomForestLearning teacher = new RandomForestLearning(variables);

// 4. Populate two variables. SampleRatio and CoverageRatio

teacher.SampleRatio = sampleRatio; //the proportion of observations used

//to train each of the trees in the forest.Default is 0.632”.

teacher.CoverageRatio = coverageRatio;//Ratio is the proportion of the

// training features that will be used to train each of the trees.

// Default value is 1.

// 5. Use the teacher object's Learn method and pass in your dataset and

// the target values

forest = teacher.Learn(inputData, \_targetValues);

// 6. The classification is now complete, and we have a forest

predictedRF = forest.Decide(inputData);

// 7. We now use the forest's predictor against our own training data and

// create a confusion matrix as a baseline

forestConfusionMatrix = new ConfusionMatrix(predictedRF, \_targetValues);

// 8. See how many decision trees are in the forest

Console.WriteLine("Tree Count={0}, Accuracy={1,4:F2}",

forest.Trees.Length, forestConfusionMatrix.Accuracy);

Console.WriteLine("ConfusionMatrix: TP={0} TN={1} FP={2} FN={3}",

forestConfusionMatrix.TruePositives,

forestConfusionMatrix.TrueNegatives,

forestConfusionMatrix.FalsePositives,

forestConfusionMatrix.FalseNegatives);

trees = forest.Trees;// setting a property

}

//////////////////////////////////////////////////////////////////

public static void PredictFromRandomForestModel(RandomForest forest, double[][] input, int[] novelTargets)

{

// use the RandomForest object you created earlier or deserialized

// earlier. Then bring in the data from the new file "double[][] input" and

// the target data "int[] novelTarget"

ConfusionMatrix forestConfusionMatrix = new

ConfusionMatrix(forest.Decide(input), novelTargets, 1, 0);

Console.WriteLine("Tree Count={0}, Accuracy={1,4:F2}",

forest.Trees.Length, forestConfusionMatrix.Accuracy);

Console.WriteLine("ConfusionMatrix: TP={0} TN={1} FP={2} FN={3}",

forestConfusionMatrix.TruePositives,

forestConfusionMatrix.TrueNegatives,

forestConfusionMatrix.FalsePositives,

forestConfusionMatrix.FalseNegatives);

}

//////////////////////////////////////////////////////////////////

public static void PredicterOnly(RandomForest forest, double[][] input)

{

// If your Novell Data contains no known targets you cannot you have nothing

// to compare to so you can’t

// build a confusion matrix. You only list your predictions.

int predictedVal = 0;

for (int i = 0; i < input.Count(); i++)

{

predictedVal = forest.Decide(input[i]); // Use the previously trained

// tree to predict the success or fail of the test data

Console.WriteLine("Predict[{0}]={1}", i, predictedVal);

}

}

//////////////////////////////////////////////////////////////////

private void ModelSave()

{

IFormatter formatter = new BinaryFormatter();

string fileToSave = “myRFmodel.rfmdl";

string RFModelsDir = @“C\Temp”;

Stream stream = new FileStream(RFModelsDir + "\\" +

fileToSave, FileMode.Create, FileAccess.Write);

formatter.Serialize(stream, RandomForestModelNew);

stream.Close();

}