IoT Big Data Processing

Apache Spark MLLib Session Lab

Albert Bifet and Jacob Montiel

October 25, 2016

MLlib

- MLLib and spark.ml are the Machine Learning libraries for Spark
- Decision trees are easy to interpret, and are able to capture non-linearities
- Random forests and boosting are among the top performers for classification and regression tasks.

Apache Spark Session Lab

- Login in the Community Edition of Databricks:
  - `http://community.cloud.databricks.com/`
- Read the Apache MLLib Guide
  - `http://spark.apache.org/docs/latest/mllib-guide.html`
- Read the Decision Tree Guide
  - `http://spark.apache.org/docs/latest/mllib-decision-tree.html`

- Start creating a new notebook
- Create a cluster
- Attach cluster to notebook

Apache Spark Session Lab

- Import classes

```
import org.apache.spark.mllib.tree.DecisionTree
import org.apache.spark.mllib.tree.model.DecisionTreeModel
import org.apache.spark.mllib.util.MLUtils
```

- Load and parse the data file.

```
val data = MLUtils.loadLibSVMFile(sc,
 "/databricks-datasets/samples/data/mllib/sample_libsvm_data.txt")
```

- Split the data into training and test sets (30 % held out for testing)

```
val splits = data.randomSplit(Array(0.7, 0.3))
val (trainingData, testData) = (splits(0), splits(1))
```

Apache Spark Session Lab

- Train a DecisionTree model.

```
val numClasses = 2
val categoricalFeaturesInfo = Map[Int, Int]()
val impurity = ”gini”
val maxDepth = 5
val maxBins = 32

val model = DecisionTree.trainClassifier(trainingData,
  numClasses, categoricalFeaturesInfo,
  impurity, maxDepth, maxBins)
```

- Evaluate model on test instances and compute test error

```
val labelAndPreds = testData.map { point =>
  val prediction = model.predict(point.features)
  (point.label, prediction)
}

val testErr = labelAndPreds.filter(r => r._1 != r._2).
            count.toDouble / testData.count()
println(”Test Error = ” + testErr)
println(”Learned classification tree model:\n” +
        model.toDebugString)
```

Apache Spark Session Lab Assignment

Write a notebook on the following tasks, writing the code in Scala:

1. What is the error of the classifier with this dataset?
2. Improve error of the classifier (tuning parameters or using Random Forests)
3. OPTIONAL: Use cross-validation for the evaluation