Ingeniería en sistemas computaciones

Datos masivos

Practica 1 - Decision tree classifier

Alumnos:

Marquez Millan Seashell Vanessa

Galaviz Lona Oscar Eduardo

Developement

The first thing is import all librarys to need in these case was

```
import org.apache.spark.ml.Pipeline
import org.apache.sparkmlclassificationDecisionTreeClassificationModel
import org.apache.spark.ml.classification.DecisionTreeClassifier
import org.apache.spark.ml.evaluationMulticlassClassificationEvaluator
import org.apache.spark.ml.feature.{IndexToString, StringIndexer,
VectorIndexer}
```

Here only load the dataframe(these df you need to have in the principal directory)

```
 val data = spark.read.format("libsvm").load("sample.txt")
```

Fit on whole dataset to include all labels in index.

```
val labelIndexer = new
StringIndexer().setInputCol("label").setOutputCol("indexedLabel").fit(data)
```

Automatically identify categorical features, and index them

```
val featureIndexer = new
VectorIndexer().setInputCol("features").setOutputCol("indexedFeatures").set
MaxCategories(4).fit(data)
```

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

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

Train a DecisionTree model

```
val dt = new
DecisionTreeClassifier().setLabelCol("indexedLabel").setFeaturesCol("indexe
dFeatures")
```

Chain indexers and tree in a Pipeline.

```
val pipeline = new Pipeline().setStages(Array(labelIndexer, featureIndexer,
dt))
```

Chain indexers and tree in a Pipeline.

```
    val pipeline = new Pipeline().setStages(Array(labelIndexer,
featureIndexer, dt))
```

Train model. This also runs the indexers.

```

```

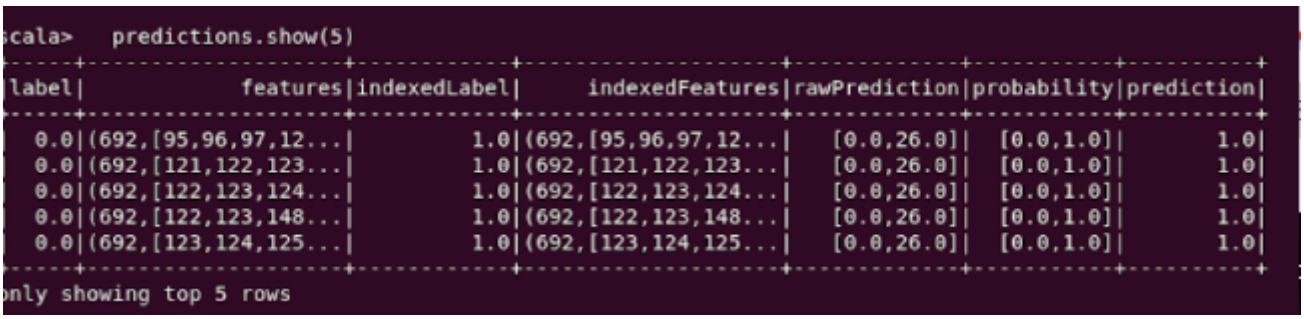
```
    val model = pipeline.fit(trainingData)
```

Make predictions.

```
    val predictions = model.transform(testData)
```

Select example rows to display.

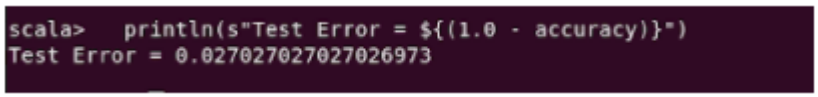
```
    predictions.show(5)
```



```
scala>   predictions.show(5)
+-----+--------------------+------------+--------------------+-------------+-----------+----------+
|label|            features|indexedLabel|     indexedFeatures|rawPrediction|probability|prediction|
+-----+--------------------+------------+--------------------+-------------+-----------+----------+
|  0.0|(692,[95,96,97,12...|         1.0|(692,[95,96,97,12...|   [0.0,26.0]|  [0.0,1.0]|       1.0|
|  0.0|(692,[121,122,123...|         1.0|(692,[121,122,123...|   [0.0,26.0]|  [0.0,1.0]|       1.0|
|  0.0|(692,[122,123,124...|         1.0|(692,[122,123,124...|   [0.0,26.0]|  [0.0,1.0]|       1.0|
|  0.0|(692,[122,123,148...|         1.0|(692,[122,123,148...|   [0.0,26.0]|  [0.0,1.0]|       1.0|
|  0.0|(692,[123,124,125...|         1.0|(692,[123,124,125...|   [0.0,26.0]|  [0.0,1.0]|       1.0|
+-----+--------------------+------------+--------------------+-------------+-----------+----------+
only showing top 5 rows
```

Select (prediction, true label) and compute test error.

```
    val evaluator = new
MulticlassClassificationEvaluator().setLabelCol("indexedLabel").setPredicti
onCol("prediction").setMetricName("accuracy")
    val accuracy = evaluator.evaluate(predictions)
    println(s"Test Error = ${(1.0 - accuracy)}")
```



```
scala>   println(s"Test Error = ${(1.0 - accuracy)}")
Test Error = 0.027027027027026973
```