Tecnológico Nacional de México

INSTITUTO TECNOLÓGICO CAMPUS TIJUANA

ING. EN SISTEMAS COMPUTACIONALES

Subdirección Académica

Departamento de Sistemas y Computación

BDD-1704 TI9A - 6:00pm-7:00pm

ASIGNATURA:

Datos Masivos

SEMESTRE:

Septiembre- Enero 2020

Tarea:

Explicación de lo que yo entendí del código:

RamdomForestClassifier.scala

MAESTRO:

JOSE CHRISTIAN ROMERO HERNANDEZ

24/11/2020

//importamos las librerias que utilizaemos

import org.apache.spark.ml.Pipeline

import org.apache.spark.ml.classification.{RandomForestClassificationModel, RandomForestClassifier}

import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator

import org.apache.spark.ml.feature.{IndexToString, StringIndexer, VectorIndexer}

import org.apache.spark.ml.attribute.Attribute

//importamos la sesion de spark

import org.apache.spark.sql.SparkSession

//creamos un objeto llamado RamdomForestClassifier el cual contendra todos los metodos

object RamdomForestClassifier {

def main(args: Array[String]): Unit = {

//creamos la variable spark y le asignamos el nombre RamdomForestClassifier

val spark = SparkSession.builder.appName("RamdomForestClassifier").getOrCreate()

// Cargamos el archivo de datos y lo convertimos en un DataFrame.

val data = spark.read.format("libsvm").load("C:/Users/DELL/Desktop/Ejercicio\_equipo3\_ramdom\_forest\_classifier/sample\_libsvm\_data.txt")

// Etiquetas de índice, agregando metadatos a la columna de etiquetas.

// ajustamos el conjunto de datos para incluir todas las etiquetas en el índice.

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

val indexed = labelIndexer.transform(data)

// Identifica automáticamente características categóricas e indexalas.

// Establezca maxCategories para que las entidades con> 4 valores distintos se traten como continuas.

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

// Divida los datos en conjuntos de prueba y entrenamiento (el 30% se reserva para probar).

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

// Entrena un modelo RandomForest.

val rf = new RandomForestClassifier().setLabelCol("indexedLabel").setFeaturesCol("indexedFeatures").setNumTrees(10)

// Convertir etiquetas indexadas de nuevo a etiquetas originales.

val labelConverter = new IndexToString().setInputCol("prediction").setOutputCol("predictedLabel").setLabels(labelIndexer.labels)

// Aqui marca error <console>: 32: error: value labelsArray no es miembro de org.apache.spark.ml.feature.StringIndexer

// labelConverter.transform (indexado)

// Cadena de indexadores y bosque en un Pipeline.

val pipeline = new Pipeline().setStages(Array(labelIndexer, featureIndexer, rf, labelConverter))

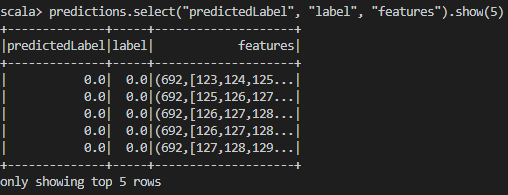
// Modelo de tren. Esto también ejecuta los indexadores.

val model = pipeline.fit(trainingData)

// Hacer predicciones.

val predictions = model.transform(testData)

//Resultados



// Seleccione filas de ejemplo para mostrar.

predictions.select("predictedLabel", "label", "features").show(5)

//predictions.select("prediction", "rawPrediction", "indexedFeatures", "label", "features", "indexedLabel", "probability").show(5)

// Seleccione (predicción, etiqueta verdadera) y calcule el error de prueba.

val evaluator = new MulticlassClassificationEvaluator().setLabelCol("indexedLabel").setPredictionCol("prediction").setMetricName("accuracy")

val accuracy = evaluator.evaluate(predictions)

//se imprime el test de error

println(s"Test Error = ${(1.0 - accuracy)}")

//Resultados



//creamos la variable rfModelel la cual contendra la libreria RandomForestClassificationModel

val rfModel = model.stages(2).asInstanceOf[RandomForestClassificationModel]

//se imprime el modelo clasificatorio forest

println(s"Learned classification forest model:\n ${rfModel.toDebugString}")

spark.stop()

}

}

//Resultados

