**TECNOLÓGICO NACIONAL DE MÉXICO**

**INSTITUTO TECNOLÓGICO DE TIJUANA**

**DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN**

**ENERO - JUNIO 2020**

**INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES**

**DATOS MASIVOS**

**BDD-1704IF9A**

**“Examen Unidad 2”**

**INTEGRANTES:**  
Enciso Maldonado Aileen Yurely No. Control: 15210329  
García Bautista Ana Laura No. Control: 15210793

**PROFESOR:**

Romero Hernández Jose Christian

**FECHA DE ENTREGA**

**03/Junio/2020**

**TECNOLÓGICO NACIONAL DE MÉXICO**

***INSTITUTO TECNOLÓGICO DE TIJUANA* SUBDIRECCIÓN ACADÉMICA**

Departamento de Sistemas y Computación

**EXAMEN**

|  |  |
| --- | --- |
| Carrera: Ingeniería En Sistemas Computacionales/ Tecnologías de la información/ Informática | Período:**Enero-Junio 2020** |
| Materia: Datos Masivos Grupo: BDD-1704/IF9A | Salón: |
| Unidad (es) a evaluar: Unidad 2 Tipo de examen: Práctico | Fecha: |
| Catedrático: Jose Christian Romero Hernandez Firma del maestro: | Calificación: |

Alumno: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ No. Control: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Instrucciones

**Desarrollar las siguientes instrucciones en Spark con el lenguaje de programación Scala, utilizando solo la documentacion de la librería de Machine Learning Mllib de Spark y Google.**

1. Del dataset **Iris.csv** que se encuentra en [**https://github.com/jcromerohdz/iris**,](https://github.com/jcromerohdz/iris) elaborar la limpieza de datos necesaria para ser procesado por el siguiente algoritmo (I**mportante, esta limpieza debe ser por medio de script de Scala en Spark**) . a. Utilice la librería Mllib de Spark el algoritmo de Machine Learning llamado **multilayer perceptrón**
2. ¿Cuáles son los nombres de las columnas?

"sepal\_length","sepal\_width", "petal\_length","petal\_width"

1. ¿Cómo es el esquema?
2. Imprime las primeras 5 columnas.
3. Usa el método describe () para aprender más sobre los datos del DataFrame.
4. Haga la transformación pertinente para los datos categóricos los cuales serán nuestras etiquetas a clasificar.
5. Construya el modelo de clasificación y explique su arquitectura.
6. Imprima los resultados del modelo

**Instrucciones de evaluación**

* + Tiempo de entrega 4 días
  + Al terminar poner el código y la explicación en la rama (branch) correspondiente de su github así mismo realizar su explicación de la solución en su Google drive.
  + Finalmente defender su desarrollo en un video de 8-10 min el cual servirá para darse calificación, este video debe subirse a YouTube para ser compartido por un link.

// Agregamos las lbrerias necesarias para trabajar con el algortimo Multilayer Perceptron.

import org.apache.spark.ml.classification.MultilayerPerceptronClassifier

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

// Del data set Iris.cvs, elaborar la limpieza de datos necesaria por medio de un scrip en scala spark, impportamos las librerias necesarias para la limpieza.

import org.apache.spark.ml.feature.VectorAssembler

import org.apache.spark.ml.feature.{VectorAssembler, StringIndexer}

import org.apache.spark.ml.linalg.Vectors

// Se cargan los datos del dataser iris.csv en la variable "data"

val data  = spark.read.option("header","true").option("inferSchema", "true").format("csv").load("iris.csv")

// Vemos el esquema para comprobar que todos los valores estan calsificados correctamente en el datset

data.printSchema()

// Se eliminan los campos null

val dataClean = data.na.drop()

// Se declara un vector que se transforma los datos a la variable "features"

val vectorFeatures = (new VectorAssembler().setInputCols(Array("sepal\_length","sepal\_width", "petal\_length","petal\_width")).setOutputCol("features"))

// Se transforman los features usando el dataframe

val features = vectorFeatures.transform(dataClean)

// Se declara un "StringIndexer" que transformada los datos en "species" en datos numericos

val speciesIndexer = new StringIndexer().setInputCol("species").setOutputCol("label")

// Ajustamos las especies indexadas con el vector features

val dataIndexed = speciesIndexer.fit(features).transform(features)

// Con la variable "splits" hacemos un corte de forma aleatoria

val splits = dataIndexed.randomSplit(Array(0.6, 0.4), seed = 1234L)

// Se declara la variable "train" la cual tendra el 60% de los datos

val train = splits(0)

// Se declara la variable "test" la cual tendra el 40% de los datos

val test = splits(1)

// Se establece la configuracion de las capas para el modelo de redes neuronales artificiales

val layers = Array[Int](4, 2, 2, 3)

// Se configura el entrenador del algoritmo Multilayer con sus respectivos parametros

val trainer = new MultilayerPerceptronClassifier().setLayers(layers).setBlockSize(128).setSeed(1234L).setMaxIter(100)

// Se entrena el modelo con los datos de entrenamiento

val model = trainer.fit(train)

// Se prueban ya entrenado el modelo

val result = model.transform(test)

// Se selecciona la prediccion y la etiqueta que seran guardado en la variable

val predictionAndLabels = result.select("prediction", "label")

// Se muestran algunos datos

predictionAndLabels.show()

// Se ejecuta la estimacion de la precision del modelo

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

val accuracy = evaluator.evaluate(predictionAndLabels)

// Se imprime el error del modelo

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