**TECNOLÓGICO​ ​NACIONAL​ ​DE​ ​MÉXICO**

**INSTITUTO TECNOLÓGICO DE TIJUANA**

**SUBDIRECCIÓN ACADÉMICA**

**DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN**

**SEMESTRE:**Enero - Junio 2020

**CARRERA:**Ing. Tecnologías de la Información y Comunicaciones

**MATERIA:**Datos Masivos

**TÍTULO:​ ​**Proyecto de regresión logística Práctica # 2

**UNIDAD​ ​A​ ​EVALUAR:​**​Unidad​ ​2

**NOMBRE Y NÚMERO DE CONTROL DEL ALUMNO:**Diaz Martinez Ruben Emilio # 15210791

**NOMBRE DEL MAESTRO (A):**Dr. Christian Romero Hernandez

import org.apache.spark.ml.classification.LogisticRegression

import org.apache.spark.sql.SparkSession

import org.apache.log4j.\_

Logger.getLogger("org").setLevel(Level.ERROR)

val spark = SparkSession.builder().getOrCreate()

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

data.printSchema()

data.head(1)

val colnames = data.columns

val firstrow = data.head(1)(0)

println("\n")

println("Example data row")

for(ind <- Range(0, colnames.length)){

println(colnames(ind))

println(firstrow(ind))

println("\n")

}

val timedata = data.withColumn("Hour",hour(data("Timestamp")))

val logregdata = timedata.select(data("Clicked on Ad").as("label"), $"Daily Time Spent on Site", $"Age", $"Area Income", $"Daily Internet Usage", $"Hour", $"Male")

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

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

val assembler = (new VectorAssembler()

.setInputCols(Array("Daily Time Spent on Site", "Age","Area Income","Daily Internet Usage","Hour","Male"))

.setOutputCol("features"))

val Array(training, test) = logregdata.randomSplit(Array(0.7, 0.3), seed = 12345)

import org.apache.spark.ml.Pipeline

val lr = new LogisticRegression()

val pipeline = new Pipeline().setStages(Array(assembler, lr))

val model = pipeline.fit(training)

val results = model.transform(test)

import org.apache.spark.mllib.evaluation.MulticlassMetrics

val predictionAndLabels = results.select($"prediction",$"label").as[(Double, Double)].rdd

val metrics = new MulticlassMetrics(predictionAndLabels)

println("Confusion matrix:")

println(metrics.confusionMatrix)

metrics.accuracy