TECNOLÓGICO NACIONAL DE MÉXICO INSTITUTO

TECNOLÓGICO DE TIJUANA
SUBDIRECCIÓN ACADÉMICA

DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN

SEMESTRE FEBRERO-JUNIO 2022

CARRERA

Ingeniería en informática e Ingeniería en Sistemas

Computacionales

MATERIA
Datos masivos
TÍTULO
Práctica evaluatoria, unidad #3

Integrantes:

Munguía Silva Edgar Geovanny #17212344
Perez López Alicia Guadalupe ##18210514

NOMBRE DEL MAESTRO
Jose Christian Romero Hernadez

Tijuana Baja California 01 de Junio del 2022

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/ Informatíca Período: **Febrero-Junio 2022** Materia: Datos Masivos Grupo: Salón: Unidad (es) a evaluar: Unidad 3 Tipo de examen:Práctico Fecha: Catedrático: Jose Christian Romero Hernandez Firma del maestro: Calificación:

Alumno: _____

No. Control: _____

Instrucciones

Desarrolle las siguientes instrucciones en Spark con el lenguaje de programación Scala.

Objetivo:

El objetivo de este examen práctico es tratar de agrupar los clientes de regiones específicas de un distribuidor al mayoreo. Esto en base a las ventas de algunas categorías de productos.

Las fuente de datos se encuentra en el repositorio:
https://github.com/jcromerohdz/BigData/blob/master/Spark_clustering/Wholesale %20customers%20data.csv

1. Importar una simple sesión Spark.
2. Utilice las lineas de código para minimizar errores
3. Cree una instancia de la sesión Spark
4. Importar la librería de Kmeans para el algoritmo de agrupamiento.
5. Carga el dataset de Wholesale Customers Data
6. Seleccione las siguientes columnas: Fresh, Milk, Grocery, Frozen, Detergents_Paper, Delicassen y llamar a este conjunto feature_data
7. Importar Vector Assembler y Vector
8. Crea un nuevo objeto Vector Assembler para las columnas de caracteristicas como un conjunto de entrada, recordando que no hay etiquetas
9. Utilice el objeto assembler para transformar feature_data
10. Crear un modelo Kmeans con K=3
11. Evalúe los grupos utilizando Within Set Sum of Squared Errors WSSSE e imprima los

centroides.

Instrucciones de evaluación

- Tiempo de entrega 3 dias
- Al terminar poner el codigo y la explicación en la rama (branch) correspondiete de su github asi mismo realizar su explicación de la solución en su google drive. - Finalmente defender su desarrollo en un video de 6-8 min el cual servira para dar su calificación, este video debe subirse a youtube para ser compartido por un link público .

Introducción.

El objetivo de esta práctica es tratar de agrupar clientes de regiones específicas de un distribuidor mayorista. Esto se basa en las ventas de algunas categorías de productos. Usaremos el algoritmo kmeans (usando bibliotecas) en Spark para usar el algoritmo de agrupamiento. En resumen, usamos este algoritmo para hacer pequeños grupos de datos y colocar ciertos datos en el grupo que es más similar en características, este es el objetivo principal de kmeans.

Code.

```
//1. Import a simple spark session.

import org.apache.spark.sql.SparkSession

//2. Use code lines to reduce code errors

import org.apache.log4j._
Logger.getLogger("org").setLevel(Level.ERROR)

// 3. Create a spark session instance

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

//4. Import Kmeans library to use the cluster algorithm

import org.apache.spark.ml.clustering.KMeans

// 5. Load  the dataset( Wholesale Customers Data.csv)

val data = spark.read.option("header",
"true").option("inferSchema","true")csv("Wholesale_customers_data.csv")

// 6. Make another data set selecting the follow columns and call the
dataframe "feature_data"(Fresh, Milk, Grocery, Frozen, Detergents_Paper,
Delicassen)
```

```
val feature_data = data.select("Fresh", "Milk", "Grocery", "Frozen",
"Detergents_Paper", "Delicassen")

// 7 Show the data of the new dataset
feature_data.show()


// 8 Importarting Vector Assembler y Vector libraries
import org.apache.spark.ml.feature.VectorAssembler
import org.apache.spark.ml.linalg.Vectors

// 9 Create a new Vector Assembler object for the feature columns as an
input set, remembering that there are no labels
val assembler=(new
VectorAssembler().setInputCols(Array("Fresh","Milk","Grocery","Frozen","D
etergents_Paper","Delicassen")).setOutputCol("features"))

// 10 Use the object assembler to transform feature_data
val transform =assembler.transform(feature_data)

// 11 Show the transformed results
transform.show()

// 12 Create kmeans model with k =3

val kmeans = new KMeans().setK(3).setSeed(1L)
val model = kmeans.fit(transform)

//13 Evaluating the groups using within set sum and print the centroids
val WSSSE = model.computeCost(transform)
println(s"Within Set Sum of Squared Errors = $WSSSE")

//14 Printing the centroids
println("Cluster Centers: ")
model.clusterCenters.foreach(println)
```

Results.

```
scala> println("Cluster Centers: ")
Cluster Centers:

scala> model.clusterCenters.foreach(println)
[7993.574780058651,4196.803519061584,5837.4926686217,2546.624633431085,2016.2873900293255,1151.4193548387098]
[9928.18918918919,21513.081081081084,30993.486486486487,2960.4324324324325,13996.594594594595,3772.3243243243246]
[35273.854838709674,5213.919354838709,5826.096774193548,6027.6612903225805,1006.9193548387096,2237.6290322580644]
```

Conclusions.

Edgar Munguia:

This practice was a little bit short, but the knowlege i got was big, because i learned how to use kmeans algorithm to cluster data. In this case, we worked with customer data to cluster the customers acording his features. I made this practice in data mining too(in a diferent context) so im sure i learned how to use it in all of possible cases.

Alicia López: It is interesting to see how the data can be controlled with a method, in addition to this the k-means method seemed particular to me since it can classify data depending on its characteristics.

In R we could observe the graphs of the data and it is still more visual how they are accommodated.

Video link (Youtube): https://www.youtube.com/watch?v=xg1Qbh2A-e0

Github repository link: *https://github.com/Aliciap26/DATOS-MASIVOS*