Licenciatura en Ciencia de Datos
Datos Masivos I

Tarea 2

Fecha de entrega**: 22 de Marzo a las 09:00 hrs**

Datos sobre el reporte:
- Escribir el código en el reporte (copy-paste).
- Adjuntar los archivos .py dentro de la carpeta enviada al correo (zip).
- La discusión y los resultados deben estar presentes en el informe.
- Enviar reporte en archivo PDF.

1. K-means (Spark) y Clasificación (Spark MLib) – (100 %)

A) Clustering K-means iterative (50%)

Programar el algoritmo K-means en Spark. El pseudocódigo del algoritmo es el siguiente.

Algorithm 1 Iterative k-Means Algorithm

```
1: procedure ITERATIVE k-MEANS
2:     Select k points as initial centroids of the k clusters.
3:     for iterations := 1 to MAX_ITER do
4:         for each point p in the dataset do
5:             Assign point p to the cluster with the closest centroid
6:         end for
7:         for each cluster c do
8:             Recompute the centroid of c as the mean of all the data points assigned to c
9:         end for
10:    end for
11: end procedure
```

Descripción del problema.
Para saber que es *más cercano*, se tiene que definir una forma de medirla. En esta tarea, se pide usar la distancia L2 (Euclidiana) y L1 (Manhattan) como las medidas de similitud.

Distancia L2:

$$||a - b|| = \sqrt{\sum_{i=1}^{d} (a_i - b_i)^2}$$

Distancia L1:

$$|a - b| = \sum_{i=1}^{d} |a_i - b_i|$$

El algoritmo de clustering intenta minimizar la función de costo “dentro del clúster”, de la siguiente forma:

$$\phi = \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} ||x - c||^2$$

$$\psi = \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} |x - c|$$

Utilizar los datos proporcionados (en la carpeta “pregunta1a”) para realizar la agrupación (clustering) de los documentos.
Un documento se representa como un vector de características de 58 dimensiones.
Cada dimensión (o característica) en el vector representa la importancia de una palabra en el documento.
La idea es que los documentos con conjuntos similares de importancia de palabras puedan tratar sobre el mismo tema.

Los datos contienen 3 archivos:
(1) *data.txt* contiene la versión vectorizada de documentos, que tiene 4601 filas y 58 columnas.
(2) *c1.txt* contiene *k* centroides de clúster iniciales. Estos centroides fueron elegidos al seleccionar *k* puntos aleatorios de los datos de entrada.
(3) *c2.txt* contiene centroides de clúster iniciales que están lo más lejos posible. Puedes hacer esto al elegir el primer centroide *c1* al azar (aleatoriamente), y luego encontrar el punto *c2* que está más alejado de *c1*, posteriormente, seleccionar *c3*, el cual está más alejado de *c1* y *c2*, y así sucesivamente.

Entonces lo que se pide es:
Implementar K-means iterativas en Spark.
Se proporciona el código de inicio *kmeans.py*, que se encarga de la carga de datos.
Completar la lógica dentro del bucle *for* y ejecutar el código con diferentes estrategias de inicialización y funciones de pérdida. Siéntete libre de cambiar el código si es necesario, o pegarlo en un notebook Jupyter.

Para las preguntas de esta tarea, se debe establecer el número de iteraciones en **20** y el número de grupos *k* en **10**.

a) Ejecutar la agrupación (clustering) en *data.txt* con *c1.txt* y *c2.txt* como centroides iniciales y use la distancia **L1** como medida de similitud. Calcular y graficar el costo “dentro del clúster” para cada iteración. Se deberán generar dos gráficas (captura de pantalla de la ejecución de su programa, y la gráfica requerida).

b) Ejecutar la agrupación (clustering) en *data.txt* con *c1.txt* y *c2.txt* como centroides iniciales y use la distancia **L2** como medida de similitud. Calcule y trace el costo “dentro del clúster” para cada iteración. Se tienen que generar dos gráficas.

c) **t-SNE** es un método de reducción de dimensionalidad particularmente adecuado para la visualización de datos de alta dimensión. Visualice el resultado de la asignación de

agrupamiento de **b)** reduciendo la dimensión a un espacio 2D. Se deberán generar dos gráficas para este inciso.

d) Para la distancia L2 y L1, ¿la inicialización aleatoria de *K-means* usando *c1.txt* es mejor que la inicialización usando *c2.txt* en términos de costo? Explique su razonamiento y discuta los resultados.

e) ¿Cuál es la complejidad del algoritmo K-medias iterativas?

Nota. En los incisos a), b) y c) discuta muy brevemente y concluya en sus propias palabras sus observaciones.

B) Clasificación binaria (50%).

Se utilizará el Dataset de Adultos el cual es un conjunto de datos público en el repositorio de UCI Machine Learning. Se puede descargar el archivo *adult.data* del siguiente enlace. Convertir el archivo a ".csv" (agregando extensión o convertirlo usando Excel), para poder manejarlo en Spark (ya se encuentra en formato csv en la carpeta "pregunta1b").

Estos datos se derivan de los datos del censo y consisten en información sobre 48,000 personas y sus ingresos anuales. Se utilizarán estos datos para predecir si un individuo gana menos de 50,000 o más de 50,000 por año.
Es simple formular el problema en uno de clasificación binaria (dos clases: la primera para individuos que ganan menos de 50,000, la segunda para los que ganan más de 50,000).

El modelo que se usará para la clasificación binaria es la regresión logística, que es un modelo de clasificación.

Se necesita un paso más a comparación con la regresión lineal: aplicar la función sigmoidea para comprimir los valores de salida a 0 – 1 y generar el umbral.
Los resultados de la regresión lineal son valores numéricos continuos, mientras que los resultados de la regresión logística son valores de probabilidad que luego pueden asignarse a dos o más clases discretas. Se puede leer más detalles sobre cómo Spark realiza la optimización en el siguiente enlace.

a) Carga de datos. Leer el archivo *csv* en un DataFrame. Puede establecer "*inferschema*" en verdadero y cambiar el nombre de las columnas con la siguiente información: "edad", "clase_de_trabajo", "peso_final", "educación", "número_de_educación", "estado_marital", "ocupación", "relación" , "raza", "sexo", "ganancia_de_capital", "pérdida_de_capital", "horas_por_semana", "país_nativo", "ingresos". Puede obtener más información sobre Dataframe en el siguiente enlace.

b) Preprocesamiento de datos. Convierta las variables categóricas en variables numéricas con ML Pipelines y Feature Transformers. Probablemente, se necesitarán ejecutar las siguientes funciones: **OneHotEncoderEstimator**, **StringIndexer** y **VectorAssembler**. Se deben dividir los datos: un conjunto de entrenamiento y un conjunto de prueba con una proporción de 70 % y 30 %, respectivamente. Establecer la semilla del generador de números pseudoaleatorios en 100.

c) Modelado. Entrenar un modelo de regresión logística de Spark MLlib con un conjunto de entrenamiento. Después del entrenamiento, graficar la curva ROC y la curva Precision-Recall de su proceso de entrenamiento.

d) Evaluación. Aplicar el modelo entrenado en el conjunto de datos de prueba. Se debe reportar el valor del área bajo la curva ROC, la precisión, y la matriz de confusión. Nota: Se debería obtener una precisión de alrededor del 85%.

Nota. En los incisos b), c) y d) discuta muy brevemente y concluya en sus propias palabras sus observaciones.

Para todos los Ejercicios:
Nota General.
En los problemas de programación, aplicar buenas prácticas de programación.
Nombrado de variables. Ejemplo: variableUno, firstVariable
Nombrado de funciones. Ejemplo: removerPuntuación(), removePunctuation()
Nombrado para clases. ClaseExtraDos(), SecondExtraClass().
Agregar explicación de lo que hace cada función o variable de importancia para la resolución de problemas. Ejemplo: #Calcula las potencias del parámetro que recibe. # Ejecuta el análisis de sentimientos.

Opcional.
Siéntete libre de utilizar otras librerías (además de las solicitadas) para realizar los ejercicios, lo cual puede darte puntos extras sobre la calificación de la tarea.