Atelier 4 : Spark

Partie 1 :

Installation avec Docker

```
docker pull jupyter/pyspark-notebook

docker run -it --rm -p 8888:8888 -p 4040:4040 -v <chemin de votre dossier>:/home/jovyan/work jupyter/pyspark-notebook
```

Se connecter sur Jupyter URL : localhost :8888/ par défaut

Exercice 1

- Créer une collection python de 10,000 entiers
- Créer un RDD à partir de cette collection
- Soustraire 1 de chaque valeur utilisant **map**
- Appliquer l'action **collect** pour voir le résultat
- Appliquer **filter** et visualiser les résultats avec **collect**

Exercice 2

Créer un fichier texte file.txt et tester l'exemple du cours sur word count.

Step 1 : Lire le fichier à partir de son emplacement et le stocker sous forme de RDD en utilisant l'opération « textFile ».

Vérifier que le fichier a été chargé avec succès en appelant la méthode count(), qui affiche le nombre d'éléments dans le RDD.

Step 2 : Diviser chaque ligne en des mots :

Ensuite, on va diviser chaque ligne en un ensemble de mots. Pour diviser chaque ligne en mots et les stocker dans un RDD appelé « words » , on utilise la méthode : « flatMap » et la fonction anonyme « Lambda »

Step 3 : Extraire sous forme clé, valeur :

Pour chaque mot on va attribuer la valeur 1 par défaut en utilisant la fonction prédéfinie « map »

Step 4 : Reduce : faire la somme des valeurs pour chaque mot :

La méthode reduceByKey() appelle l'expression lambda pour tous les tuples avec le même mot. L'expression lambda a deux arguments, a et b, qui sont les valeurs de comptage dans deux tuples

Step 5 : Écrire le résultat dans un fichier texte dans votre disque avec coalesce(1).saveAsTextFile()

La méthode coalesce() combine toutes les partitions RDD en une seule partition puisque on veut un seul fichier de sortie, et saveAsTextFile() écrit le RDD à l'emplacement spécifié.

Exercice 3

Ecrire un script PySpark qui permet d'analyser un fichier log « logFile.txt » et d'extraire les modules qui ont générés un grand nombre d'erreurs.

Le fichier log d'entrée contient des données sous la forme :

« timestamp – module – message »

Instructions :

1. Créer un RDD à partir du fichier log « logFile.txt »
2. Filtrer le contenu de RDD pour ne conserver que les lignes qui contiennent des messages d'erreur.
   Remarque : Une ligne contient un message d'erreur si elle contient le mot « erreur » (peu importe la casse).
3. Utiliser une transformation pour extraire les noms des modules qui génèrent des erreurs.
4. Effectuer un comptage du nombre d'erreurs par module en utilisant la transformation 'reduceByKey'
5. Filtrer les modules qui ont généré un grand nombre d'erreurs par exemple, plus de 20 erreurs.
6. Afficher les modules suspects ainsi le nombre d'erreurs associé.

Partie 2 : Manipulation des DataFrames

Créer une application pyspark et récupérer une session.

```
from pyspark.sql import SparkSession

spark = SparkSession.builder.master("local").appName("tp-spark").config("spark.ui.port",

"4050").getOrCreate()
```

Question 1 : Création d'un dataframe

Créer un dataFrame à partir de données saisies « à la main ». L'affichage du schema et du

dataframe montre le résultat suivant :

```
root
 |-- firstname: string (nullable = true)
 |-- middlename: string (nullable = true)
 |-- lastname: string (nullable = true)
 |-- id: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- salary: integer (nullable = true)

+---------+----------+--------+-----+------+------+
|firstname|middlename|lastname|id   |gender|salary|
+---------+----------+--------+-----+------+------+
|James    |          |Smith   |36636|M     |3000  |
|Michael  |Rose      |        |40288|M     |4000  |
|Robert   |          |Williams|42114|M     |4000  |
|Maria    |Anne      |Jones   |39192|F     |4000  |
|Jen      |Mary      |Brown   |     |F     |-1    |
+---------+----------+--------+-----+------+------+
```

Question 2 : Renommage

Renommer la colonne ‘id’ par ‘identifiant ‘ et affiche le nouveau schema

Question 3 :

Filtrage de données

a. Sélectionner les données sur les femmes seulement parmi les données du dataframe.
b. Sélectionner les personnes avec des salaires différents de 4000.

Question 4 : Tri de plusieurs colonnes à la fois

Triez selon le nom et le prénom.

Utiliser sort ou orderBy

Partie 3 : Machine learning

Exercice 1 : Système de recommandation

Les deux types de systèmes de recommandation les plus courants sont :

Le filtrage basé sur le contenu se concentre sur les attributs des éléments et vous donner des recommandations basées sur la similitude entre eux.

Le filtrage collaboratif (CF) : produit des recommandations basées sur la connaissance de l'attitude des utilisateurs envers les articles

spark.ml prend actuellement en charge le CF, dans lequel les utilisateurs et les produits sont décrits par un petit ensemble de facteurs latents qui peuvent être utilisés pour prédire les entrées manquantes.

spark.ml utilise l'algorithme Alternating Least Squares (ALS) pour apprendre ces facteurs latents. Les données doivent être dans un format spécifique pour fonctionner avec l'algorithme de recommandation ALS de Spark.

Exemple :

User	Movie	Rating
1	A	5
1	B	5
2	A	5
2	B	5
1	C	1
2	C	1
3	A	5

En se basant sur le CF, on va recommander le film B pour le user 3 car les users 1 et 2 préfèrent le film A aussi.

Dans cette partie on va essayer de modéliser cet aspect avec pyspark, pour ce faire, sous jupyter notebook :

1. Créer un sparkSession et le démarrer
2. Importer l'algo ALS :
```
from pyspark.ml.recommendation import ALS
from pyspark.ml.evaluation import RegressionEvaluator
```
3. Charger les données de « ratings.csv » dans un dataframe ratings
4. Afficher le contenu de ratings
5. Supprimer les colonnes inutilisables pour cet exercice
6. Diviser la dataset en training et testing sets (0.8, 0.2)
7. Créer le modèle ALS :
```
Als = ALS(maxIter=5, regParam=0.01, userCol='userId',
itemCol='movieId', ratingCol='rating')
```
8. Créer maintenant le modèle en utilisant als.fit(training)
9. Faire des prédictions sur l'ensemble de test
10. Evaluer le modèle en calculant la métrique RMSE (Root Mean Squared Error) entre la colonne rating et la colonne de prédictions.
11. Utiliser le modèle pour générer des recommandations personnalisées pour chaque utilisateur.
12. Afficher les recommandations pour un utilisateur spécifique.