EPITA – Ing 2
Big Data Analytics – BDA

TP 5 : PySpark et SparkSQL

idir.benouaret@epita.fr

Consignes

- Travail à réaliser en binôme
- Il faut produire une archive zip contenant :
  1. votre code commenté, de préférence un fichier par exercice.
  2. un rapport pdf (*please use latex*), qui doit contenir vos explications, description de votre solution, analyse vos résultats, et tout ce qui vous semble utile.
- Rendu le 28 Mars 23h42.

Introduction

Vous allez explorer un jeu de données public sur les trajets en taxi à New York. Les données sont issues de cette page.

Il faut récupérer que les données "Yellow Taxi Trip Records" de l'année 2022. (Il existe un fichier de données pour chaque moi de l'année 2022).

Les données sont au format Apache Parquet (https ://parquet.apache.org/). C'est un format de fichiers orienté colonne qui est développé pour l'écosystème de calcul distribué. Il est compatible avec la plupart des frameworks de traitement de données large échelle. Il fournit des schémas efficaces de compression et de codage des données avec des performances améliorées pour gérer des données complexes en masse.

Vous pouvez transformer les données en format csv, si vous le souhaitez. Il suffit d'utiliser la librairie Pandas qui dispose aussi d'une méthode pour lire des données en format parquet :

```
import pandas as pd
df=pd.read_parquet("yellow_tripdata_2022-01.parquet")
df.to_csv("JAN22.csv", sep =";", index=False)
```

Vous pouvez alors constater le taux de compression des fichiers **parquet** par rapport à la taille du csv.

Exercice I : Lecture des données

La première étape est de lire les données. Pour cela, il faut d'abord instancier une SparkSession :

```
from pyspark.sql import SparkSession

spark = SparkSession.builder \
.master("local[*]")
.appName("taxi-ex1") \
.config("spark.driver.memory", "6g") \
.getOrCreate()

spark.sparkContext.setLogLevel("ERROR")
```

- `.master` permet de spécifier la machine maître
- `.appName` permet de nommer l'application
- `config("spark.driver.memory", "6g")` permet d'allouer la quantité de RAM souhaitée. Par exemple 6GO ici.
- `spark.sparkContext.setLogLevel("ERROR")`, permet de d'avoir seulement les erreurs et les résultats dans les logs.

Spark dispose d'une méthode pour lire les fichiers de type *Parquet*
`https://spark.apache.org/docs/latest/sql-data-sources-parquet.html`

Exemple pour lire les données du mois de Janvier :

```
data=spark.read.option("header","true").option("inferschema","true").parquet("file:///
    home/idir/Desktop/BIGDATA/TPs/TP5/Taxi_data/yellow_tripdata_2023-01.parquet")
```

Bien-sur, à adapter sur votre machine.

Q1 – Créer un dataframe qui va contenir les données des trajets en taxi sur les 12 mois de l'année 2022.

Q2 – Testez ensuite :

```
data.show()
data.printSchema()
```

Voici quelques descriptions des colonnes (In English) :

- **VendorID** : A code indicating the TPEP provider that provided the record. 1= Creative Mobile Technologies, LLC ; 2= VeriFone Inc

- **tpep_pickup_datetime** : The date and time when the meter was engaged
- **tpep_dropoff_datetime** : The date and time when the meter was disengaged
- **passenger_count** : The number of passengers in the vehicle. This is a driver-entered value
- **trip_distance** : The elapsed trip distance in miles reported by the taximeter
- **payment_type** : A numeric code signifying how the passenger paid for the trip. 1= Credit card 2= Cash 3= No charge 4= Dispute 5= Unknown 6= Voided trip
- **tip_amount** : Tip amount. This field is automatically populated for credit card tips. Cash tips are not included
- **total_amount** : The total amount charged to passengers. Does not include cash tips
- **PULocationID** : pick up location id, LC Taxi Zone in which the taximeter was engaged
- **DOLocationID** : dropout location id, TLC Taxi Zone in which the taximeter was disengaged

Exercice II : Data Cleaning

Le nettoyage de données, est une étape cruciale dans le processus d'analyse de données. Il consiste à détecter et à corriger les erreurs, les incohérences et les valeurs manquantes dans un ensemble de données. En éliminant ces imperfections, le data cleaning garantit la qualité et la fiabilité des données, ce qui permet une analyse précise et des résultats significatifs.

Q1 – Calculer le nombre total de trajets dans le dataset.

- **Erreurs potentielles sur les montants :**

Il existe probablement dans le dataset des enregistrements ou le `total_amount` est soit négatif (cela voudrait dire qu'on aurait été payé pour prendre un taxi) ou bien des montants largement excessifs.

Q2 – Calculer le plus petit montant et le plus grand montant enregistré.

Vous trouverez sûrement que le plus petit montant est négatif et le plus grand montant dépasse les 400.000$ (à moins d'être un trajet New York-Los Angeles, c'est sûrement une erreur de saisie)

Q3 – On se propose de garder uniquement les lignes avec des montants positifs ainsi que des montants inférieurs à 1000$. Faites les transformations nécessaires sur votre dataframe

- **Distance de trajet non raisonnable :**

Il existe des enregistrements ou la distance parcourue lors d'un trajet est erronée. Plusieurs '0', qui correspondent sûrement à des trajets annulés, ou bien des distances beaucoup trop grandes.

Q4 – Calculer des statistiques sur la distance des trajets : distance minimale, maximale, moyenne, distribution, etc

Q5 – Filtrer les données de sorte à ne garder que les trajets avec une distance strictement positive et ne dépassant pas les 100 miles (approximativement 160km)

- **Nombre de passagers :**

Il peut aussi exister des erreurs sur le nombre de passagers

Q6 – Filtrer les données de sorte à ne garder que les trajets avec un nombre de passagers cohérent (on se propose d'utiliser l'intervalle [1-10])

Q7 – Quels sont les nombres de passagers les plus fréquents ? Calculer la distribution du nombres de passagers

- **Année des trajets :**

Vous avez téléchargé des données sur des trajets de 2022.

Q8 – Vérifier que toutes les lignes ont bien une date de début et de fin de trajet en 2022. Sinon, supprimez les lignes qui n'ont pas une date correcte.

Q9 – Une fois, vos données nettoyées, vous pouvez les stocker et ne travailler que sur celles-ci pour la suite du TP.

Exercice III : Data Analysis (featuring DataViz)

L'objectif ici est d'analyser les données, répondre à des requêtes et produire des visualisations qui montrent les résultats.

On s'intéresse tout d'abord à voir comment le nombre de trajets évolue dans le temps

Q1 – Calculer le nombre de trajets effectués pour chaque jour de l'année

Q2 – Créer un graphique qui montre cette évolution

On s'intéresse maintenant à savoir quels jours de semaine compte le plus grand nombre de trajets

Q3 – Calculer le nombre de trajets pour chaque jour [Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday]

Q4 – Créer un histogramme qui montre la distribution

On s'intéresse maintenant à voir l'évolution du nombre de trajets, ainsi que la distance de trajet par heure. C'est à dire de minuit à 1h, de 1h à 2h, ainsi de suite jusqu'à 23h-minuit.

Q5 – Calculer le nombre de trajets et la distance moyenne pour chaque chaque intervalle

Q6 – Créer les visualisations qui montrent ces distributions ? à quelle heures sont les trajets les plus courts ? Pour quels raisons à votre avis ?

Q7 – Quelles méthodes de paiement sont les plus populaires ? Afficher le pourcentage de chacune des catégories de paiement

Q8 – Quelle est la durée moyenne d'un trajet pour chaque mois ?

Q9 – Est ce que les clients donnent plus de pourboires pendant les weekends comparé aux autres jours de la semaine ?

Exercice IV : Quartiers les plus prisées de New York

Pour répondre aux prochaines questions. Il faudra utiliser un deuxième dataset. Voici un lien de téléchargement. Ce dataset contient pour chacun location_id son arrondissement ainsi que sa zone.

Q1 – Faire la jointure entre les deux datasets pour récupérer l'arrondissement et la zone pour chaque trajet.

Q2 – Quelles sont les tops destinations (top 4 des zones) ? et d'ou les trajets correspondants commencent t'ils ?

ici, il faudra calculer le nombre de trajets de chacune des destinations, prendre les 4 destinations les plus populaires et afficher le nombre total de trajets depuis les 5 zones de départ les plus fréquentes.

Exercice V : Impact de la météo sur les trajets

Pour répondre aux prochaines questions. Il faudra utiliser un jeu de données qui contient les données météorologiques sur la station de New York. Télécharger ce jeu de données à l'adresse suivante : `https://www1.ncdc.noaa.gov/pub/data/noaa/2022/725053-94728-2022.gz`. Un descriptif détaillé est présenté en figure 1

Ici, les DataFrames montrent leurs limites comparés aux RDDs. Les DataFrames sont inutilisables sur des données non structurées. C'est le cas ici ou un RDD est requis pour pouvoir traiter ces données.

Q1 – Comment le nombre de trajets varie en fonction de la température qu'il fait ?

On s'intéresse ici à analyser le nombre de trajets par jour en fonction de la température. Créer la visualisation en nuages de points correspondante.

Exercice VI : Machine Learning (Mllib) :

Spark.MLlib `https://spark.apache.org/docs/latest/ml-guide.html` est une librairie de machine learning pour Spark qui contient tous les algorithmes et utilitaires d'apprentissage classiques, comme la classification, la régression, le clustering, le filtrage collaboratif, la réduction de dimensions, en plus des primitives d'optimisation sous-jacentes.

Le but de cette exercice est de réimplémenter une des approches de régression que vous avez vu en cours de machine learning.

offset	taille	exemple	signification
4	6	332130	USAF weather station identifier
10	5	99999	WBAN weather station identifier
15	8	19500101	observation date YYYYMMDD
23	4	0300	observation time HHMM
28	6	+51317	latitude (degrees x 1000)
34	7	+028783	longitude (degrees x 1000)
46	5	+0171	elevation (meters)
60	3	320	wind direction (degrees)
63	1	1	wind direction quality code
65	4	0072	wind speed (meters per second x 10)
69	1	1	wind speed quality code
70	5	00450	sky ceiling height (meters)
75	1	1	sky ceiling height quality code
78	6	010000	visibility distance (meters)
84	1	1	visibility distance quality code
87	1	-	air temperature sign
88	4	0128	air temperature (degrees Celsius x 10)
92	1	1	air temperature quality code
93	1	-	dew point temperature sign
94	4	0139	dew point temperature (degrees Celsius x 10)
98	1	1	dew point temperature quality code
99	5	10268	atmospheric pressure (hectopascals x 10)
104	1	1	atmospheric pressure quality code

FIGURE 1: Description

Il s'agira de prédire le prix d'un trajet. Intuitivement, le montant total d'un trajet dépend fortement de la distance parcourue et de la durée du trajet.

Q1 – Implémenter un modèle de régression de votre choix avec tout le pipeline, préparation des données, splits, train, loss, évaluation des prédictions.

Documentation : `https://spark.apache.org/docs/latest/ml-classification-regression.html#classification-and-regression`

EXERCICE VII : Movies

Télécharger le jeu de données MovieLens10M

Vous allez travailler sur les deux fichiers suivants

- `movies.dat` : chaque ligne correspond à movieId : :title : :genres
- `ratings.dat` :chaque ligne correspond à userId,movieId,rating,timestamp

Q1 – Construire les deux dataframes correspondants aux jeux de données.

Q2 – Combien de films y a t'il dans le jeu de données ?

Q3 – Construire une nouvelle colonnes qui va contenir l'année de sortie de chaque film. Cette donnée se trouve en parenthèse à la fin de chaque titre de film

Q4 – Donner la liste de tous les genres de films disponible.

Q5 – Construire une liste de films par genres, i.e., répertorier tous les films pour les genres "Drama", "Romance", etc.

Q6 – Trouver le nombre de films pour chaque appréciation (rating 1, 2, 3, 4 et 5)

Q7 – Quels sont les 10 films qui ont été regardé le plus de fois (par le plus d'utilisateurs) ?