Compte rendu évaluation - NASA

Groupe 3 : Matthieu FRANCOIS, Aude LYSKAWA, Laurine HUSSONG, Tai Mui CHENH

Installation du cluster Spark/MySQL - Docker compose file

```
version: '3'
services:
  spark:
    build:
      dockerfile: docker/spark/Dockerfile
    container_name: spark-container
    ports:
      - "8081:8081" # Spark Web UI
    environment:
      - SPARK_MODE=master
      - SPARK_RPC_AUTH_SECRET=mysecret
    depends_on:
      - mysql
    links:
      - "mysql"
    volumes:
      - ./app:/app  # Mount your Spark application directory here

  mysql:
    image: mariadb:latest
    container_name: mariadb-container
    ports:
      - "3306:3306"  # MySQL port
    environment:
      MYSQL_ROOT_PASSWORD: verystrongrootpassword123*
      MYSQL_DATABASE: star_classification
      MYSQL_USER: nasa
      MYSQL_PASSWORD: verystrongpassword123*
    volumes:
       - ./docker/mariadb/data:/var/lib/mariadb  # Mount a data directory for
mariadb
```

Accès interfacé à la DB grâce au logiciel DBeaver

Utilisation de commandes “Make” pour gagner du temps (démarrage, build, arrêt du conteneur, lancement de commandes dans le conteneur…)

Utilisation de JDBC pour importer les données du fichier JSON dans la base de données.

Utilisation de deux conteneurs (un conteneur avec spark et un autre conteneur utilisé pour la base de données), coordonnés via un fichier docker compose

Ingestion des données via Spark (chargement des données)

Nous avons réalisé un fichier import_data.py pour importer les données grâce à Spark :

```
from pyspark.sql import SparkSession

# Create a Spark session
spark = SparkSession.builder \
     .appName("CSV to MariaDB") \
     .config("spark.jars", "./mysql-connector-j-8.1.0.jar") \
     .getOrCreate()

# Read CSV data
csv_data = spark.read.option("header", "true").csv("star_classification.csv")

# Configure the MariaDB JDBC connection
jdbc_hostname = "mysql"
jdbc_port = 3306
jdbc_database = "star_classification"
jdbc_username = "nasa"
jdbc_password = "verystrongpassword123*"

jdbc_url = f"jdbc:mysql://{jdbc_hostname}:{jdbc_port}/{jdbc_database}"

connection_properties = {
        "user": jdbc_username,
        "password": jdbc_password,
}

# Write the data to MariaDB
csv_data.write.mode("overwrite").option("driver",
"com.mysql.cj.jdbc.Driver").jdbc(jdbc_url,               "target_table_name",
properties=connection_properties)

# Stop the Spark session
spark.stop()
```

Exploration du jeu de données

Nous avons deux fichiers :

- “columns.txt” : qui décrit brièvement le contenu de chaque colonne
- “star_classification.csv” : un jeu de données concernant des caractéristiques d’étoiles

Nous avons 17 colonnes “features” contenant des données numériques et 1 colonne “class” qui classifie en “GALAXY”, “QSO” ou “STAR”.

Le jeu de données est composé de 100000 observations, et aucune donnée vide.

Affichage du jeu de données :

```
# Obtenir le décompte de chaque classe
class_counts = db_data.groupBy("class").count().orderBy("class")

class_names = class_counts.select("class").rdd.flatMap(lambda x: x).collect()
class_values = class_counts.select("count").rdd.flatMap(lambda x: x).collect()

colors = ["blue", "green", "red"]

plt.figure(figsize=(10, 6))
plt.bar(class_names, class_values, color=colors)
plt.xlabel("Classes")
plt.ylabel("Nombre d'occurrences")
plt.title("Répartition des classes")
plt.xticks(rotation=45)
plt.savefig("/app/class_distribution.png")
```

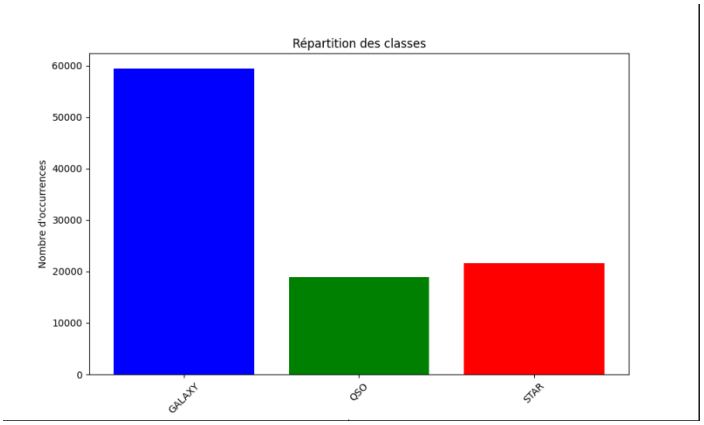
obj_ID	alpha	delta	u	g	r	i	z	run_ID	rerun_ID	cam_col	field_ID	spec_obj_ID	class	redshift	plate	MJD	fiber_ID
1237660961327743232	135.6891066036	32.4946318397087	23.87882	22.2753	20.39501	19.16573	18.79371	3606	301	2	79	6543777369295181824	GALAXY	0.6347936	5812	56354	171
1237659119865692672	232.4104400754488	44.1395225381926	22.79534	20.41623	19.36005	18.90266	18.46067	3177	301	4	158	9582671128893347840	GALAXY	0.1500074	8511	57901	498
1237659119865692672	232.4407304273024	44.1588994677042	22.79568	20.38425	19.30214	18.87287	18.53271	3177	301	4	158	9587058460048250880	GALAXY	0.1470362	8515	58192	75
1237661971184353280	182.224827835943	7.17572318774657	19.45763	17.50104	16.5029	16.07877	15.70743	3841	301	3	116	1827344121845540864	GALAXY	0.1018859	1623	53089	31
1237652947990872064	4.827196025892088	-9.666176187857862	21.31197	19.96298	19.39339	19.17418	19.00725	1740	301	4	54	2153944708919879680	STAR	-0.0004051681	1913	53321	357
1237664879951151360	144.8261005502560	31.27418489449390	24.77759	22.83188	22.58444	21.16812	21.61427	4518	301	5	119	1176014203670733414	GALAXY	0.779136	10445	58158	427
1237661360758850048	194.7675355400120	46.2749811437482	22.14477	21.40244	21.37678	21.11273	20.58925	3699	301	2	66	7451408550772103168	QSO	2.126301	6618	56401	738
1237663789023560192	109.9834984208920	41.3172622052681	19.40231	18.42656	18.10738	17.99906	17.97566	4264	301	5	29	4116425937033385984	STAR	-0.0003522841	3656	55241	494
1237661360758719232	194.2217897543360	46.4352862057077	22.65673	26.29403	24.3849	23.48402	19.58696	3699	301	2	64	8346404699763922944	QSO	2.208522	7413	56769	395
1237660961330430208	142.1887895625086	35.5824441819976	25.26307	22.66389	20.60976	19.34857	18.94827	3606	301	2	120	5152200256025548800	GALAXY	0.6441945	4576	55592	299
1237662225142644992	189.0769481184960	40.0862363301809	22.06921	21.34152	21.26156	21.38925	21.0959	3900	301	4	396	9947578689234688000	QSO	2.50185	8835	57427	920
1237657589784772864	178.1796980801857	54.2189396686821	17.6248	16.08449	15.40949	14.99461	14.70082	2821	301	2	176	1143942938942793728	GALAXY	0.05210746	1016	52759	104
1237663478724297984	338.7410377531460	-0.4028275745874820	22.13682	23.77656	21.61162	20.50454	19.2501	4192	301	3	214	1030107141295442124	GALAXY	0.9323456	9149	58039	775
1237661360759046656	195.3897856910620	46.3157372306352	23.41965	21.31521	21.33383	20.99591	21.07772	3699	301	2	69	7451440711487215616	STAR	0.0001209931	6618	56401	855
1237667549271818752	166.0966184563390	22.9527906116904	26.28256	22.10568	20.51727	19.74482	19.36206	5140	301	1	96	7237491139837974528	GALAXY	0.4497706	6428	56279	751
1237662225150640128	212.1752065912730	36.8875676955945	21.79773	19.04122	17.81297	17.30854	17.05966	3900	301	4	518	3300142171457349632	STAR	-0.0001960196	2931	54590	471
1237680272041378048	345.2825932100350	21.1838656010284	19.43718	17.58028	16.49747	15.97711	15.54461	8102	301	3	137	6891864880783316992	GALAXY	0.1161227	6121	56187	842
1237661970651873280	192.3135251041650	6.7568271512449	20.63207	19.40234	19.29327	19.21562	19.17073	3841	301	2	183	6094598550582482944	STAR	-2.562455e-05	5413	55976	372
1237655504569106944	253.3173682418580	40.4988074782155	21.82887	21.31457	21.10452	21.0298	20.8419	2335	301	6	38	5840258972733691904	QSO	0.9071854	5187	56074	786
1237680272039609088	340.9951205091910	20.5894762801019	23.48827	23.33776	21.32195	20.25615	19.54544	8102	301	3	110	5658976714552006656	QSO	1.424659	5026	55855	741

Nous réalisons un affichage sous forme d'histogramme pour visualiser la répartition :


```
# Obtenir le décompte de chaque classe
class_counts = db_data.groupBy("class").count().orderBy("class")

class_names = class_counts.select("class").rdd.flatMap(lambda x: x).collect()
class_values = class_counts.select("count").rdd.flatMap(lambda x: x).collect()

colors = ["blue", "green", "red"]

plt.figure(figsize=(10, 6))
plt.bar(class_names, class_values, color=colors)
plt.xlabel("Classes")
plt.ylabel("Nombre d'occurrences")
plt.title("Répartition des classes")
plt.xticks(rotation=45)
plt.savefig("/app/class_distribution.png")
```

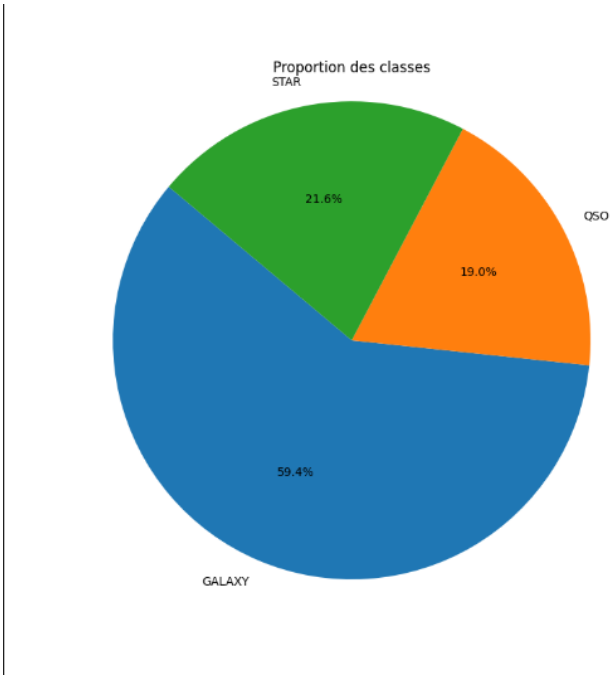
Résultat de l'affichage :

```
+------+-----+
| class|count|
+------+-----+
|GALAXY|59445|
|   QSO|18961|
|  STAR|21594|
+------+-----+
```



L'affichage sous forme de camembert permet de mieux se rendre compte de la répartition sur le total :

```
total_samples = db_data.count()
class_proportions = [(count / total_samples) for count in class_values]

plt.figure(figsize=(8, 8))
plt.pie(class_proportions, labels=class_names, autopct='%1.1f%%', startangle=140)
plt.axis('equal')
plt.title("Proportion des classes")
plt.savefig("/app/class_proportion.png")
plt.show()
```



Les données sont bien équilibrées dans l'ensemble.

Prétraitement des données

Parmi les features, certaines ne semblent pas pertinentes, notamment les colonnes servant d'identifiants (clés étrangères) sur des tables qui n'ont pas été fournies :

- obj_ID
- run_ID
- rerun_ID
- cam_col
- field_ID
- spec_obj_ID
- plate
- fiber_ID

A priori la colonne "MJD" ne devrait pas être utilisée non plus pour notre cas d'utilisation, la classification d'étoiles.

Le retrait de ces features devraient grandement améliorer la précision du modèle (en éliminant le bruit), réduire la complexité du modèle (modèle plus interprétable, plus facile à entraîner et moins sujet à l'overfitting), et de gagner en temps et en efficacité.

La feature "class" doit également être prétraitée, car il s'agit d'un champ texte qui ne peut pas être utilisé directement pour entraîner un modèle :

```
indexer = StringIndexer(inputCol='class', outputCol='class_integer')
indexer_model = indexer.fit(db_data)
df_indexed = indexer_model.transform(db_data)

df_indexed.show()
```

```
only showing top 20 rows

+-------------------+-----------------+-----------------+-----------------+-----------------+-----------------+-----------------+------+-------+--------+--------+--------------------+-----+-------------+-----+-----+--------+-------------+
|             obj_ID|            alpha|            delta|                u|                g|                r|                i|     z|run_ID|rerun_ID|cam_col|field_ID|           spec_obj_ID|class|     redshift|plate|  MJD|fiber_ID|class_integer|
+-------------------+-----------------+-----------------+-----------------+-----------------+-----------------+-----------------+------+-------+--------+--------+--------+--------------------+-----+-------------+-----+-----+--------+-------------+
|1237652947990872064|4.82719602589208|-9.666176187857861|21.31197|19.96298|19.39339|19.17418|19.00725|  1740|     301|       4|      54| 2153944708919879680| STAR| -0.000405168| 1913|53321|     357|          1.0|
|1237661360758719232|194.22178975433616| 46.43528620570776|22.65673|26.29403| 24.3849|23.48402|19.58696|  3699|     301|       2|      64| 8346404699763922944|  QSO|     2.288522| 7413|56769|     395|          2.0|
|1237662225142644992|189.07694811849616| 40.08623633818092|22.06921|21.34152|21.26156|21.38925| 21.0959|  3900|     301|       4|     396| 9947578689234688000|  QSO|      2.58185| 8835|57427|     920|          2.0|
|1237662225150640128|212.17520659127376| 36.88756769555945|21.79773|19.84122|17.81297|17.30854|17.05966|  3900|     301|       4|     518| 3300142171457349632| STAR|-0.0001960196| 2931|54590|     471|          1.0|
|1237655504569106944|253.31736824185808| 40.49808747821556|21.82887|21.31457|21.10452| 21.0298| 20.8419|  2335|     301|       6|      38| 5840258972733691904|  QSO|    0.9071854| 5187|56074|     786|          2.0|
|1237655504569958912|254.60596629480616| 38.78058769544156|22.16304|20.94692| 20.8066|20.74476|20.83111|  2335|     301|       6|      51| 5840045117722089472| STAR|-0.0008139431| 5187|56074|       8|          1.0|
|1237655504569958912|254.74706259794516| 38.75950556439936|21.63496|20.67726| 20.4901|20.34973|20.61157|  2335|     301|       6|      51| 5610514364849477632|  QSO|     2.969364| 4983|55836|     564|          2.0|
| 1.237657629522e+18|159.06853503507352| 49.75408183589144|24.89028| 23.9512|22.13194|20.67298|19.87961|  2830|     301|       4|     310| 8317197827152566272|GALAXY|    0.6758067| 7387|57038|     637|          0.0|
|1237662501074829568|227.91508504611816| 44.76497085787552|25.90415|23.13835|21.69955|20.60211| 19.7113|  3964|     301|       6|      99| 9591741829927426048|GALAXY|    0.7914561| 8519|58198|     729|          0.0|
|1237678619566408704|2.563749128732264| 2.994992457902249|23.99661| 22.2837|20.23466|18.83656| 18.3817|  7717|     301|       5|     346| 4839300685878286336|GALAXY|    0.6982945| 4298|55511|     665|          0.0|
|1237678619568046336|6.173583643151916| 3.096601657174384|23.05595| 20.3602|18.52163|17.84892|17.43123|  7717|     301|       5|     371| 4841619281308571648|GALAXY|    0.3626595| 4300|55528|     908|          0.0|
|1237678619571454720|14.104301352518304| 3.007742721881564| 22.5139|21.60116|19.91395|19.02777|18.58129|  7717|     301|       5|     423| 4850574529259657216|GALAXY|    0.4913472| 4308|55565|     719|          0.0|
|1237678619571701888|14.746158928509016| 3.078691026040834|22.54612|20.34517|18.74261|18.16549| 17.8226|  7717|     301|       5|     428| 4849277929660127024|GALAXY|    0.3320246| 4307|55531|      98|          0.0|
|1237678619572372224| 16.16435042350216| 3.057693263586664|23.50993|22.54079|20.94234| 19.7655|19.34329|  7717|     301|       5|     437| 4851584430069012480|GALAXY|    0.6097172| 4309|55528|     297|          0.0|
|1237648720681632512|152.09614334940616|-0.555865458635089|22.74037|22.10531|20.95744|20.02829|19.55473|   756|     301|       2|     242| 4311087599773308928|GALAXY|    0.6435596| 3829|55300|      61|          0.0|
|1237651755078779136|167.87235580312176| 3.336535719746464|19.22198|17.92459|17.37042|17.14866| 17.0773|  1462|     301|       6|     284| 3651466103592427520|  STAR| 0.0002802989| 3243|54910|     628|          1.0|
|1237665025440612608|181.21890772770776| 35.22439313446312|21.79423|21.38708|21.17512| 20.8861|21.52993|  4552|     301|       4|      80| 1153580905955112284|  QSO|     1.46838|10245|58161|     598|          2.0|
|1237665025443627264|189.71759681184376| 35.25771558557152|19.53029|18.49169|18.74819|18.97257|19.08105|  4552|     301|       4|     126| 3822583099080264704|  STAR|-7.010998e-05| 3395|55004|     556|          1.0|
|1237665025444806912|192.97941158775816| 34.95670103432016|21.40027|21.05039|20.82257|20.43502|20.14235|  4552|     301|       4|     144| 9985717175095416832|  QSO|     1.839288| 8869|57481|     403|          2.0|
|1237660961862320384|130.30821445608136| 29.91113355296916| 21.8483|21.70157|21.58038|21.00601|21.04556|  3606|     301|       3|      44|1206662231221877760|GALAXY|    0.9630224|10664|58464|      93|          0.0|
+-------------------+-----------------+-----------------+-----------------+-----------------+-----------------+-----------------+------+-------+--------+--------+--------+--------------------+-----+-------------+-----+-----+--------+-------------+
only showing top 20 rows

(base) laurine@laurine-IdeaPad: ~/Documents/M2I/archi_distri/stars/github/eval-nasa$
```

Les valeurs dans la colonne "class" sont maintenant de type 0.0, 1.0 et 2.0

```
# Liste des colonnes à supprimer
columns_to_drop = ["run_ID", "rerun_ID", "cam_col", "field_ID", "spec_obj_ID", "plate", "fiber_ID"]

# Supprimer les colonnes spécifiées
df_indexed = df_indexed.drop(*columns_to_drop)

# Affichez le DataFrame résultant
```

```
df_indexed.show()
```

Le dataframe final :

alpha	delta	u	g	r	i	z	class	redshift	MJD	class_integer
135.6891066036	32.4946318397087	23.87882	22.2753	20.39501	19.16573	18.79371	GALAXY	0.6347936	56354	0.0
182.224827835943	7.17572318774657	19.45763	17.50104	16.5029	16.07877	15.70743	GALAXY	0.1018859	53089	0.0
144.826100550256	31.2741848944939	24.77759	22.83188	22.58444	21.16812	21.61427	GALAXY	0.779136	58158	0.0
109.983498420892	41.3172622052681	19.40231	18.42656	18.10738	17.99906	17.97566	STAR	-0.0003522841	55241	1.0
142.188789562506	35.5824441819976	25.26307	22.66389	20.60976	19.34857	18.94827	GALAXY	0.6441945	55592	0.0
4.82719602589208	-9.66617618785786	21.31197	19.96298	19.39339	19.17418	19.00725	STAR	-0.000405168	53321	1.0
178.179690801857	54.2189396686821	17.6248	16.08449	15.40949	14.99461	14.70002	GALAXY	0.05210746	52759	0.0
232.410440075488	44.1395225381926	22.79534	20.41623	19.36005	18.90266	18.46067	GALAXY	0.1500074	57901	0.0
338.741037753146	-0.402827574587482	22.13682	23.77656	21.61162	20.50454	19.2501	GALAXY	0.9323456	58039	0.0
192.313525104165	6.7568271512449	20.63207	19.40234	19.29327	19.21562	19.17073	STAR	-2.562455e-05	55976	1.0
194.221789754336	46.4352862057077	22.65673	26.29403	24.3849	23.48402	19.58696	QSO	2.208522	56769	2.0
345.282593210935	21.1838656010284	19.43718	17.58028	16.49747	15.97711	15.54461	GALAXY	0.1161227	56187	0.0
359.303623366272	15.0271254826645	19.5777	18.10893	17.29736	16.87648	16.56135	GALAXY	0.1146274	52226	0.0
232.440730473024	44.1588994677042	22.79568	20.38425	19.30214	18.87287	18.53271	GALAXY	0.1470362	58192	0.0
340.625336033754	13.8782007077051	17.46103	16.09259	15.40038	14.98329	14.72299	GALAXY	0.07564311	52263	0.0
189.076948118496	40.0862363301809	22.06921	21.34152	21.26156	21.38925	21.0959	QSO	2.50185	57427	2.0
194.767535540012	46.2749811437482	22.14477	21.40244	21.37678	21.11273	20.58925	QSO	2.126301	56401	2.0
347.735997853974	14.582385947365	19.61089	18.3114	17.62349	17.20762	16.97802	GALAXY	0.068248	52251	0.0
212.175206591273	36.8875676955945	21.79773	19.04122	17.81297	17.30854	17.05966	STAR	-0.0001960196	54590	1.0
340.995120509191	20.5894762801019	23.48827	23.33776	21.32195	20.25615	19.54544	QSO	1.424659	55855	2.0

Feature engineering

Pour entraîner le modèle, nous avons décidé de partir sur la répartition suivante :

- 60% pour l'entraînement
- 20% pour la cross-validation
- 20% pour les tests

```
X = df_indexed
y = df_indexed['class_integer'].values

train_df, test_df = df_indexed.randomSplit([0.8, 0.2], seed=42)

train_df, cv_df = df_indexed.randomSplit([0.8, 0.2], seed=42)
```

Nous n'avons pas pu terminer, mais l'idée était d'identifier les features les plus importantes avec un modèle de type random forest.

Modeling

Vu que le but est d'obtenir un résultat de type classification, nous avons envisagé d'utiliser un DecisionTreeClassifier ou un Random Forest.
Et enfin, afficher les matrices de confusions résultantes et évaluer les modèles.

Conclusion

Nous avons rencontré quelques difficultés avec la mise en place de la connexion avec la base de données et les différents environnements, ce qui ne nous a pas permis d’arriver au bout de l’exercice.