PROJET 8 : DÉPLOYEZ UN MODÈLE DANS LE CLOUD

LAURENT CAGNIART - OPENCLASSROOMS

PROBLÉMATIQUE

Le Digital au service de l'agriculture

Préserver la biodiversité des fruits en permettant des traitements spécifiques pour chaque espèce de fruits en développant des robots cueilleurs intelligents.

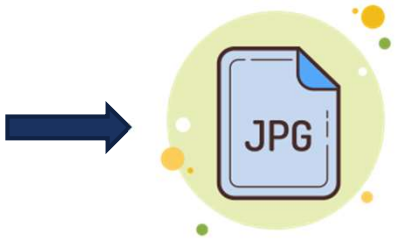
Mettre à disposition une application mobile grand public permettant de **reconnaitre un fruit à partir de sa photo** et d'en obtenir des informations.

- Développer dans un environnement Big Data une première chaîne de traitement des données qui comprendra le preprocessing et une étape de réduction de dimension.
- Prendre en compte le passage à l'échelle des calculs du fait que le volume de données va augmenter très rapidement après la livraison du projet en :
  - Ecrivant des scripts en PySpark.
  - Déployant le modèle sur le Cloud

DONNÉES DE BASE

kaggle

(un fruit ou légume par image)



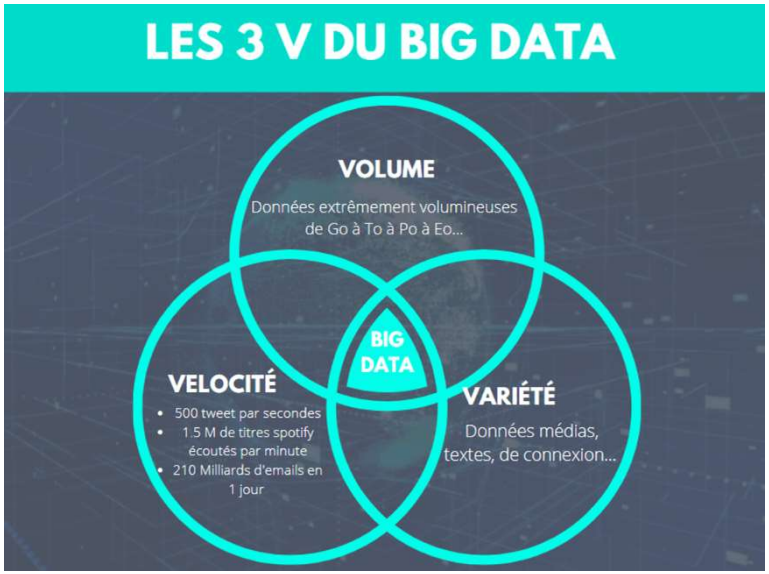
- **Training dataset** : 67,692 images
- **Test dataset** : 22,688 images

- Le jeu de données est un ensemble d'images de fruits et de labels associés (ex. Apple golden)

- Variétés : 131 fruits ou légumes
- Taille image (pixels) : 100 x 100
- Format : jpg
- Photos prises sous différents angles et états (ex. « rotten»)

BIG DATA

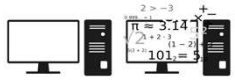
Nous considérerons que l'on fait du ***big data*** à partir du moment où la quantité de données excède la faculté d'une machine à les stocker et les analyser en un temps acceptable



- Le **Volume** des données générées nécessite de repenser la manière dont elles sont stockées.
- La **Vélocité** à laquelle nous parviennent ces données implique de mettre en place des solutions de traitement en temps réel qui ne paralysent pas le reste de l'application.
- Les données se présentent sous une grande **Variété** de formats : ces données peuvent être structurées (documents JSON), semi-structurées (fichiers de log) ou non structurées (textes, images). L'ingestion, l'analyse et la rétention de ces données prendront des formes différentes selon leur nature, ce qui implique de mettre en place des outils appropriés.

Distribuer de manière intelligente et efficace le stockage et les traitements de ces données

Calcul



Stockage

BIG DATA

La solution à notre problème consiste à paralléliser les calculs sur plusieurs machines différentes.

Système distribué
un ensemble de programmes informatiques qui utilisent des ressources informatiques sur plusieurs nœuds de calcul distincts pour atteindre un objectif commun et partagé

Les systèmes distribués :

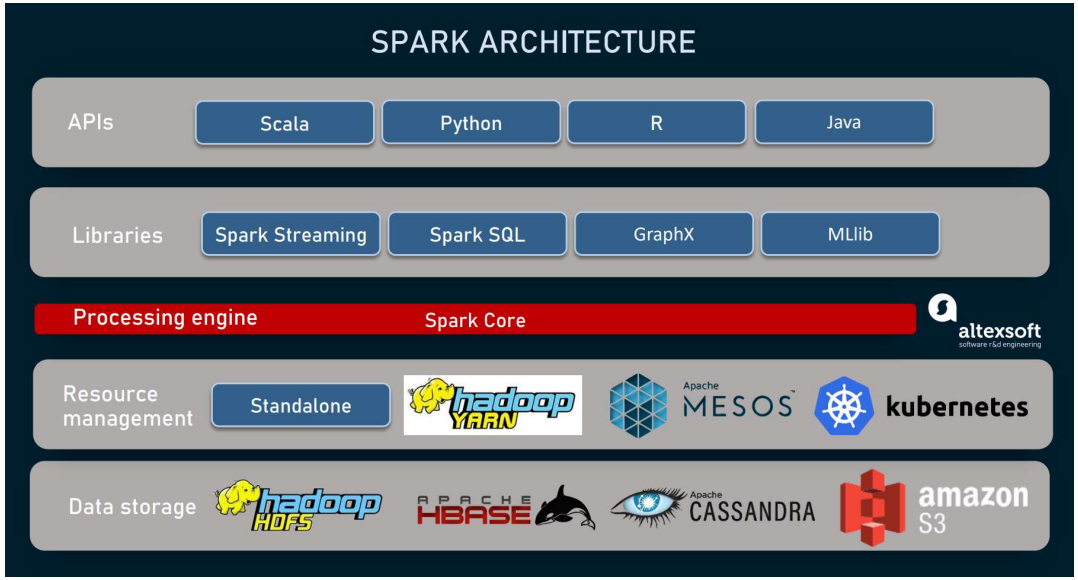
- Sont extensibles
- Offrent une évolutivité.
- Et des facilités de partage des ressources

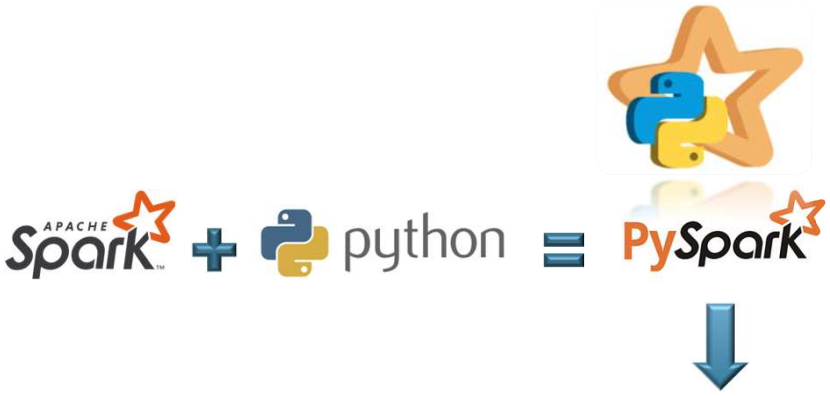
Stockage	Utiliser des systèmes de fichiers => Hadoop HDFS
Traitement des données	MapReduce ou Spark

- Stratégie de distribution ? Agrégation des résultats ? Prise en compte des pannes lors de l'exécution du calcul ? Optimisation des coûts ?

Apache Spark est un framework open source de calcul distribué in-memory pour le traitement et l'analyse de données massives par le biais de Clusters (cluster computing).

Spark exécute des programmes jusqu'à 100 fois plus vite que Hadoop MapReduce en mémoire, ou 10 fois plus vite sur disque.

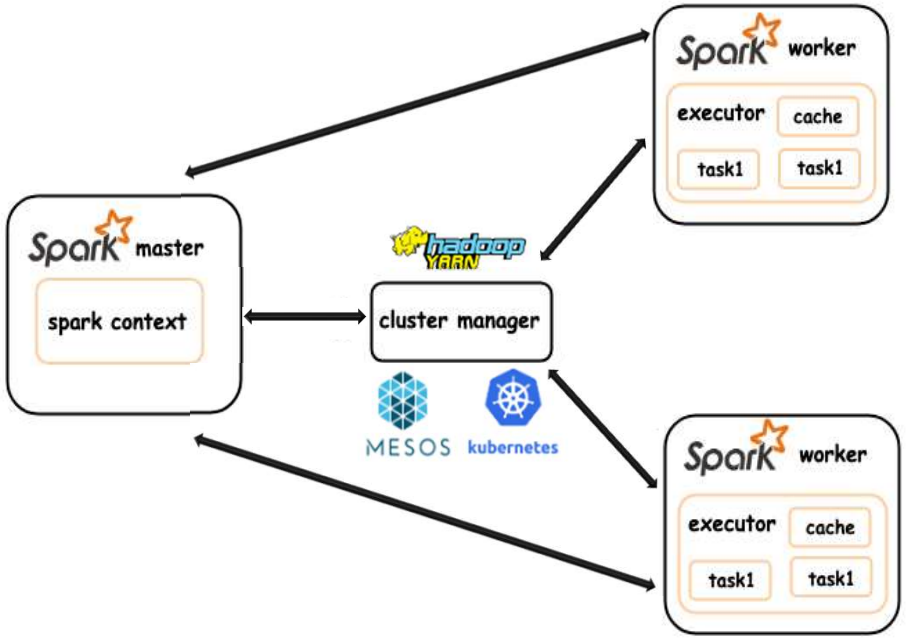




PySpark est une interface pour Apache Spark en Python.

En mode local, le parallélisme des calculs se fera en fonction du nombre de cœurs disponibles sur la machine

Architecture Spark : maître-esclave



- **Driver** : L'objet Sparkcontext ou Sparksession instancié et configuré va créer une machine virtuelle (Java) responsable de la mise en œuvre de l'application et de la distribution et l'agrégation des calculs (RDD)

- **Cluster manager** : il permet de :
  - Gérer l'allocation de ressources.
  - Gérer la division des programmes.
  - Gérer l'exécution du programme.

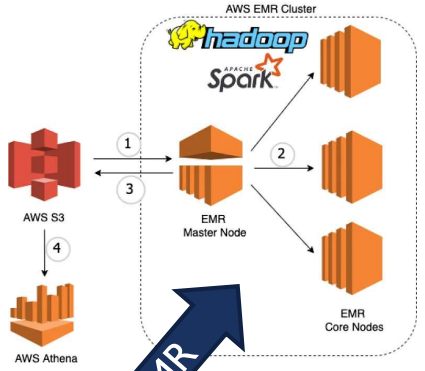
- **Worker** : reçoit une tâche du pilote et instancie un «executor» chargé d'exécuter les différentes tâches de calcul

AWS

Amazon Web Services (AWS) est la plateforme cloud la plus complète et la plus largement adoptée au monde. Elle propose plus de 200 services complets issus de centres de données du monde entier.

AWS IAM

Amazon Identity and Access Management est un service qui permet de contrôler l'accès aux ressources AWS. Il permet la Configuration des « rôles » sur notre compte AWS pour permettre à nos différentes instances de communiquer entre elles



EMR

Amazon S3

Amazon Simple Storage Service est un service de stockage d'objets qui offre une capacité de mise à l'échelle.

RGPD

Amazon EC2

Amazon Elastic Compute Cloud est un service permettant de louer des serveurs. EC2 permet un déploiement extensible des applications à travers une interface web où le client peut créer des machines virtuelles (instances du serveur).

Instances et bucket S3 créés sur le territoire européen (France et Irlande)

AWS

IAM

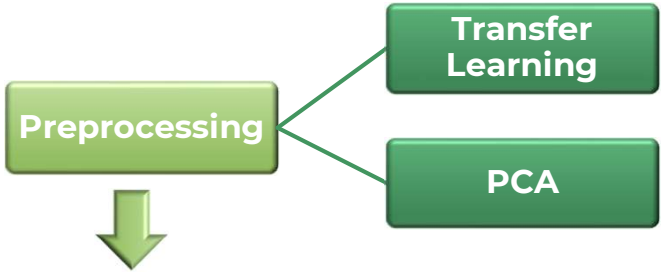
- Définition des rôles par défaut et utilisation du compte Root pour les besoins limités sur les 1ères étapes du projet

S3

- Création d'un bucket «oc-lolo-p8-data».
- Stockage des fichiers : images, résultats (parquet, json), notebook

EMR

- Création des instances Linux m5.xlarge avec choix des applications de base (Hadoop, spark, Jupyter, Tensorflow)
- Bootstrapping (pré installation des librairies et packages nécessaires : pandas, fsspec, etc…)
- Connection à EMR via tunnel SSH (Putty)
- Exécution du code sur Jupyter Notebook



Extraire les features afin de préparer les images pour la modélisation

Le ***Transfer Learning*** permet d'utiliser les connaissances acquises par un réseau de neurones lors de la résolution d'un problème afin d'en résoudre un autre plus ou moins similaire

Analyse des **C**omposantes **P**rincipales : Il s'agit de résumer l'information contenue dans une large base de données en un certain nombre de variables synthétiques appelées : Composantes Principales.

Pipeline Transfer Learning- MobileNet - PCA

Création Spark session — *SparkSession.builder.getOrCreate()*

Chargement des images — Création d'un Spark Dataframe avec les chemins des fichiers images avec *spark.read()*

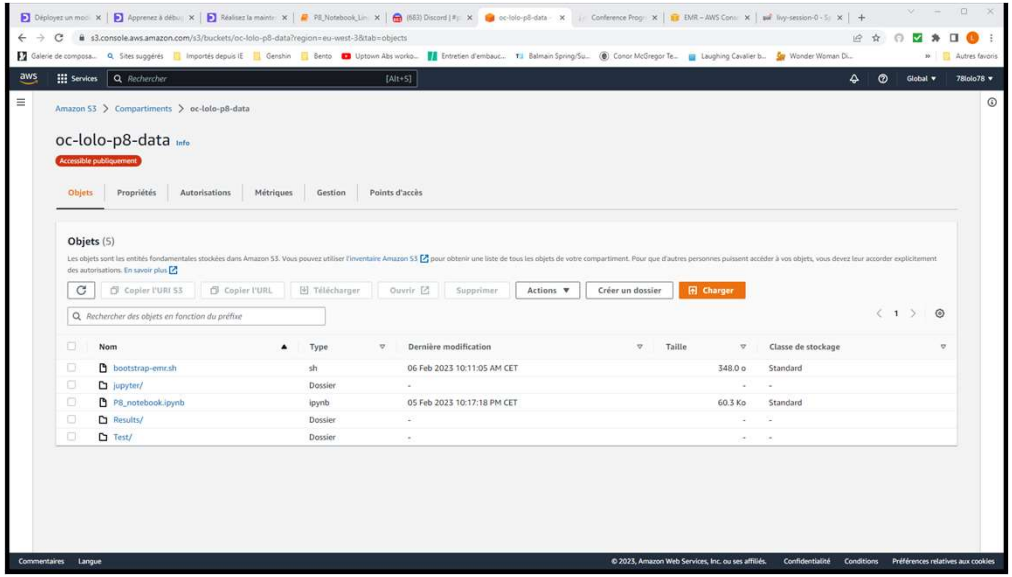
Prétraitement — Création des labels avec *spark.withColumn()*, chargement du modèle (CNN-MobileNet) et redimensionnement

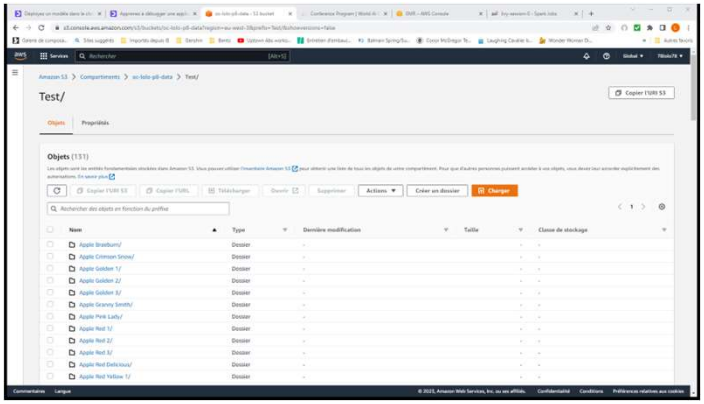
Extraction des features et réduction dimensionnelle

- Extraction des features
- UDF permet de distribuer le traitement à chaque ligne .

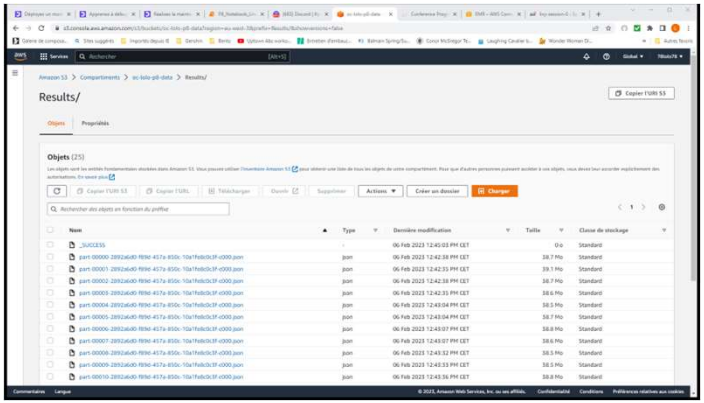
SCRIPTS PYPARK

- S3 console





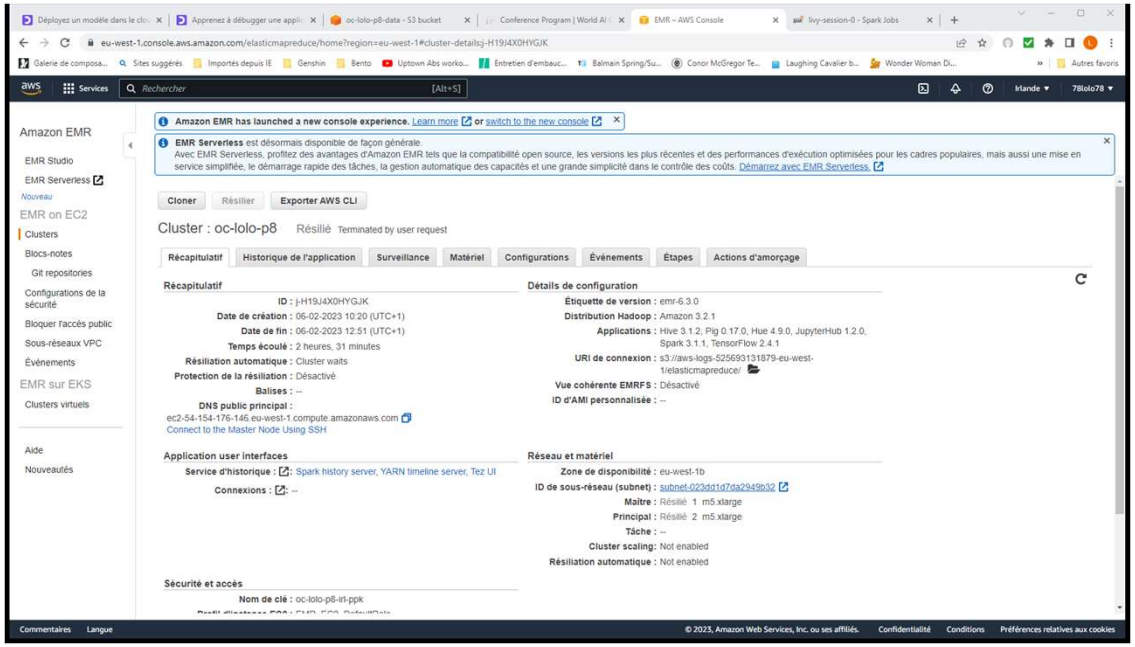
- Images dataset



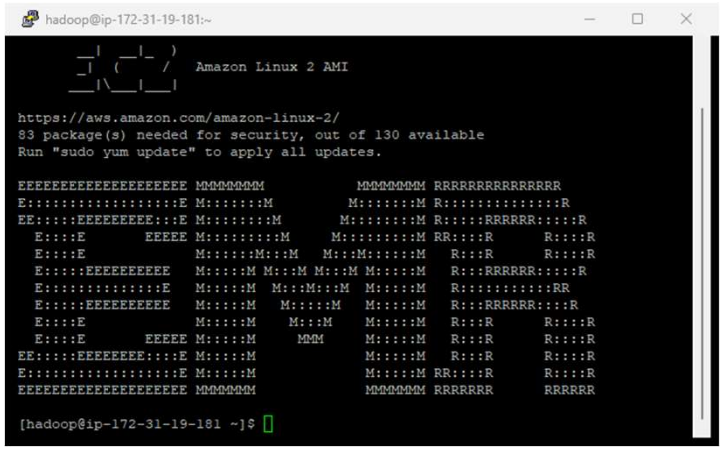
- Results files

AWS SCREENSHOTS – S3 BUCKET

- EMR console - cluster

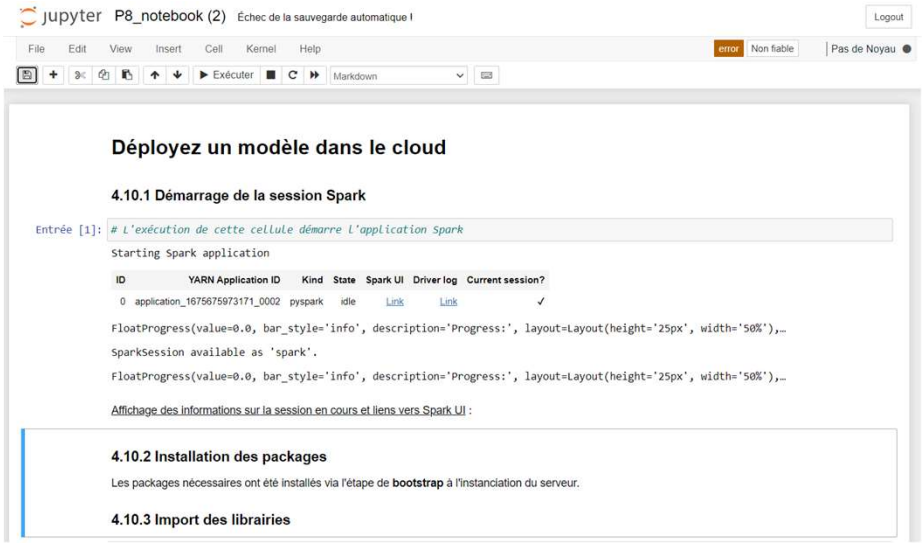


- Connexion instance EMR via tunnel SSH

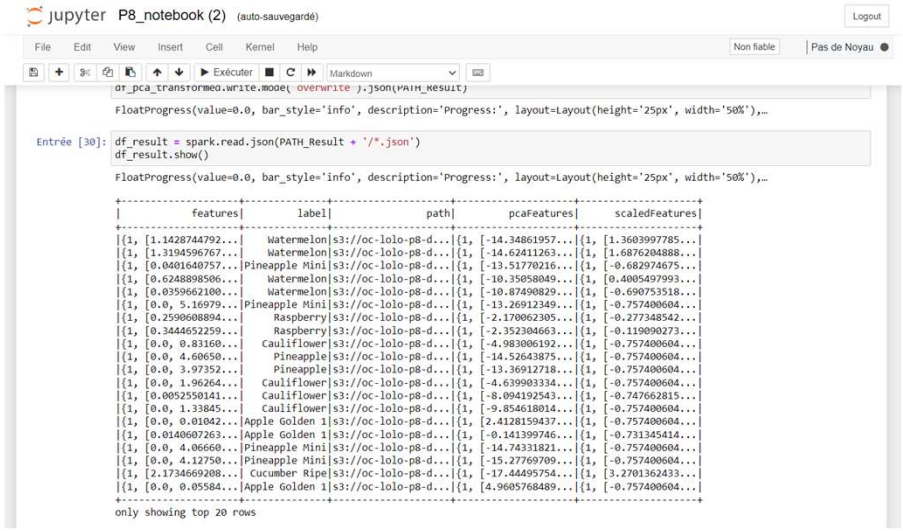


AWS SCREENSHOTS – EMR

- Lancement de la session Spark



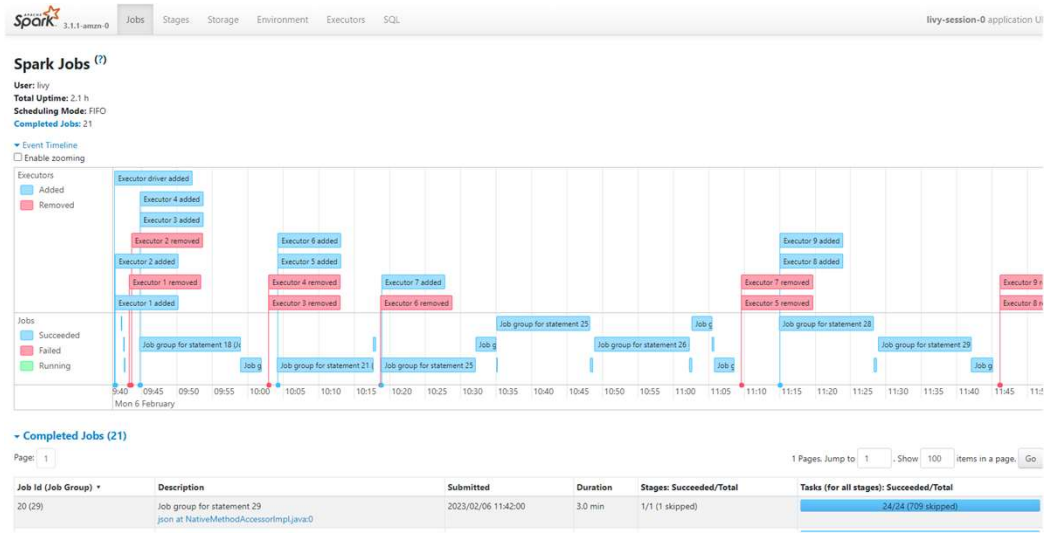
- Résultat des scripts PySpark

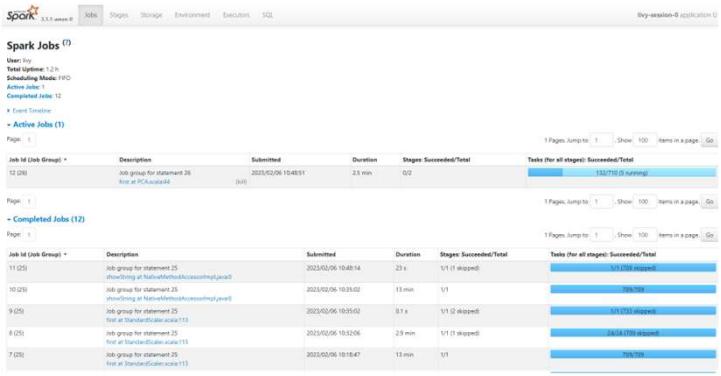


AWS SCREENSHOTS – JUPYTER NOTEBOOK

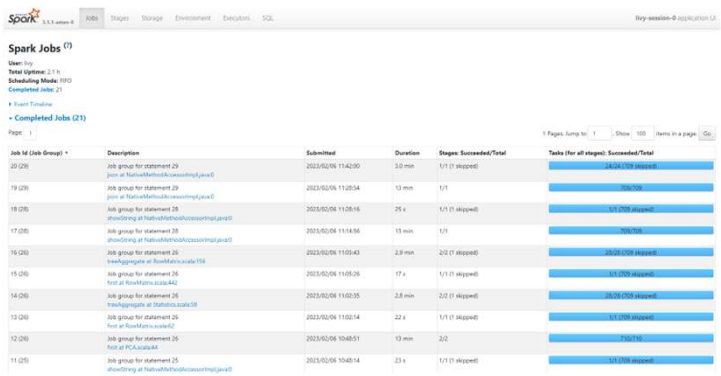
AWS SCREENSHOTS – SPARK JOBS

- Timeline





- Active job



- List of all jobs

AXES D'AMÉLIORATION

- Infrastructure AWS et scripts Pyspark créés permettant une future mise à l'échelle (revoir le scaling des clusters selon les besoins pour optimisation des coûts)
- Améliorer la sécurisation de l'ensemble (IAM, bucket S3), minimiser la surface d'attaque
- Affiner la PCA et finaliser la modélisation (classification des images)

MERCI DE VOTRE ATTENTION

LAURENT CAGNIART

MARQUEUR ICON BY ICONS8