

“Déployer un modèle dans le cloud”

Projet n°8

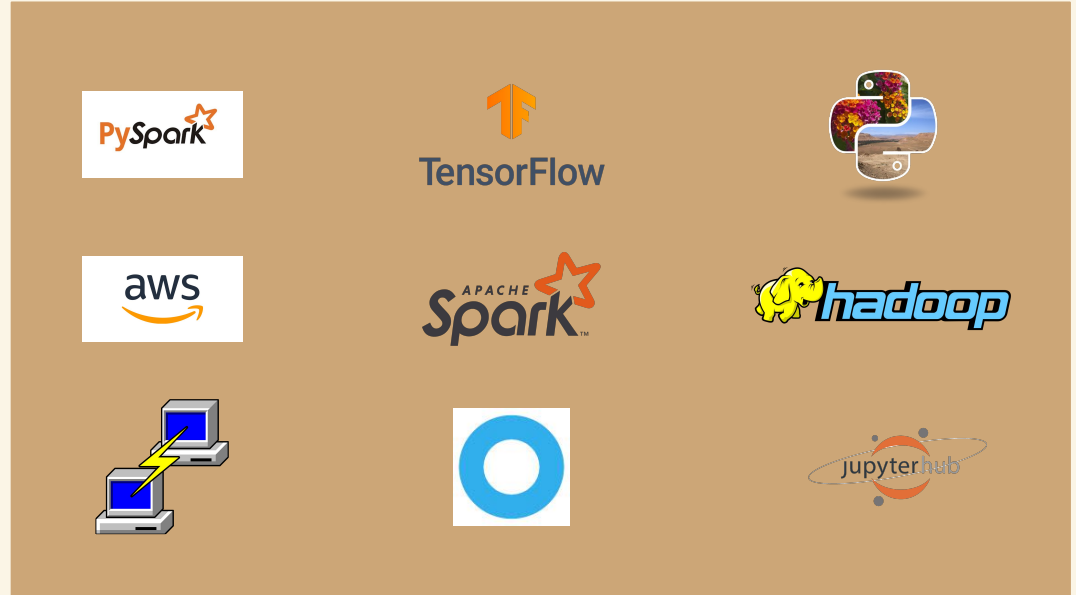
Léa ZADIKIAN
Parcours Data Scientist
23 Juin 2023

Implémentez un modèle de scoring

- I. Problématique et jeu de données**
- II. Processus de création de l'environnement Big Data**
- III. Chaîne de traitement d'images**
- IV. Démonstration d'exécution du script PySpark sur le cloud**

Environnement technique

- Notebook Jupyter 6.4.8
- Python 3.9.12
- **Librairies :**
 - Pandas, Numpy
 - PIL
 - PySpark
 - TensorFlow
- AWS (Amazon Web Services)
- PuTTY
- SwitchyOmega



I. Problématique et jeu de données

Déployer un modèle dans le cloud

- **"Fruits !" :** jeune start-up de l'AgriTech qui cherche à proposer des **solutions innovantes pour la récolte de fruits**. Son ambition : développer des **robots cueilleurs intelligents** qui permettraient des traitements spécifiques pour chaque espèce de fruits, et ainsi mieux **préserver la biodiversité** des fruits.
- **"Fruits !"** souhaite se faire connaître grâce à **une application grand public** qui permettrait de prendre en photo un fruit et d'obtenir des informations sur ce fruit.
- **Cette application** permettrait de :
 - Sensibiliser le grand public à la biodiversité des fruits ;
 - Mettre en place **une première version** d'un **moteur de classification** des images de fruits et de **l'architecture Big Data** nécessaire.



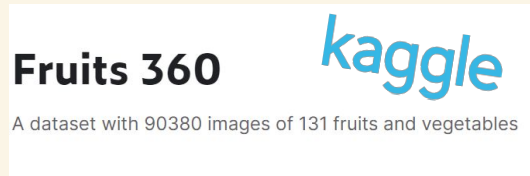
→ Notre mission :








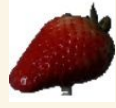



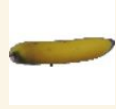
1. Mettre en place **l'architecture Big Data** nécessaire pour le passage à l'échelle en termes de volume de données.
2. À partir d'un jeu de données constitué des images de fruits et des labels associés, s'approprier **la chaîne de traitement d'images** et la compléter par une étape de réduction de dimensions.

Le jeu de données “Fruits !”

https://www.kaggle.com/moltean/fruits?select=fruits-360_dataset

- Base de données d'images **Fruits 360** sur Kaggle.
- Jeu de test comprenant **22.688 images de fruits**, un fruit par image.
- **131 classes** : *Apple Golden, Banana, Kiwi, Strawberry...*
- Un répertoire par classe, avec **plusieurs photos du même fruit sous différents angles**.
- Taille des images : **100x100 pixels**.
- Sur fond blanc uniformisé.



 Apple Golden			
 Strawberry			
 Banana			



II. Processus de création de l'environnement Big Data



Pourquoi un environnement Big data ?

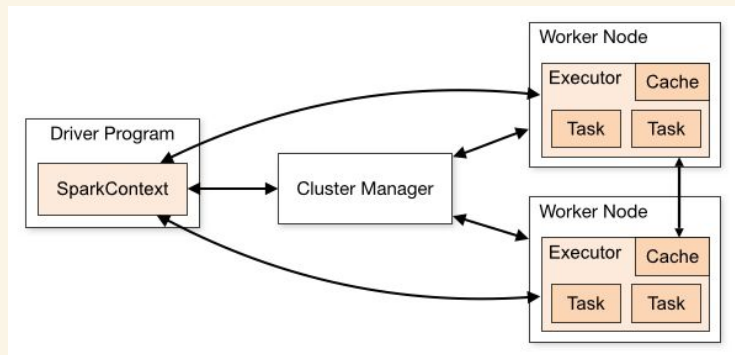
Big Data (ou données massives) : données telles que les solutions classiques de stockage, de gestion et de traitement ne suffisent plus.

Les 3V du Big Data

Volume	Le Volume des données générées nécessite de repenser la manière dont elles sont stockées.	<ul style="list-style-type: none">• Avec une croissance rapide de l'application, le volume de données collectées augmentera considérablement.• Le Big Data permet de stocker, gérer et analyser ces grandes quantités de données de manière évolutive.
Vélocité	La Vélocité à laquelle nous parvenons ces données implique de mettre en place des solutions de traitement en temps réel qui ne paralysent pas le reste de l'application.	<ul style="list-style-type: none">• Le nombre d'utilisateurs et de photos prises augmentera rapidement, générant une quantité importante de données en temps réel.• Le Big Data offre les outils nécessaires pour traiter et analyser ces données à grande vitesse.
Variété	Les données se présentent sous une grande Variété de formats : structurées (doc JSON), semi structurées (fichiers de log), non structurées (textes, images)...	<ul style="list-style-type: none">• L'application mobile générera différents types de données, tels que des images de fruits, des informations associées et des métadonnées.• Le Big Data permet de traiter et d'analyser efficacement ces multiples sources de données hétérogènes.

Les outils du Big Data

- **Calculs distribués** : distribution du stockage et des traitement des données sur plusieurs unités de calcul réparties en clusters, au profit d'un seul projet afin de diviser le temps d'exécution d'une requête.
 - **Apach Spark** : framework open-source permettant de traiter des bases de données massives en utilisant le calcul distribué (in-memory). Outil qui permet de gérer et de coordonner l'exécution de tâches sur des données à travers un groupe d'ordinateurs.
 - **Algorithme MapReduce** :
 - Largement utilisé pour le traitement parallèle et distribué de grandes quantités de données.
 - Permet de diviser les données en ensembles plus petits, de les traiter indépendamment (MAP) et de les agréger pour obtenir le résultat final (REDUCE).
 - Développement des scripts en **pySpark**, la librairie python (proche de pandas) permettant de communiquer avec Spark.
- ⇒ **Avantages** : évolutivité (ajout de ressources supplémentaires), performances (accélération du temps de calculs), tolérance aux pannes (plus résilients aux pannes ou erreurs).



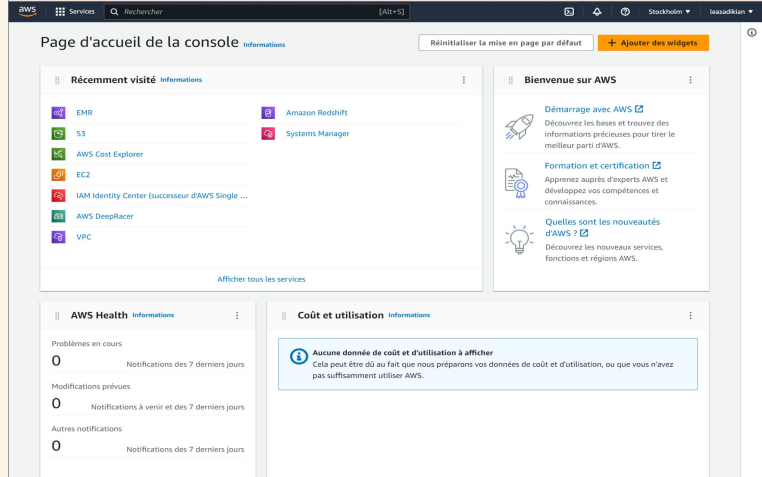
Application Spark

Le **driver** distribue et planifie les tâches entre les différents **exécuteurs** qui les exécutent et permettent un traitement réparti. Il est le responsable de l'exécution du code sur les différentes machines.

Cluster Manager : assure le suivi des ressources disponibles.

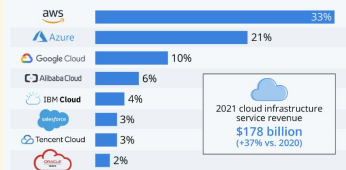
Déploiement de la solution dans le cloud

- **Louer de la puissance de calcul à la demande** : pouvoir, quel que soit la charge de travail, obtenir suffisamment de puissance de calcul pour pouvoir traiter les images, même si le volume de données venait à fortement augmenter.
- **Diminuer les coûts** si l'on compare les coûts d'une location de serveur complet sur une durée fixe (1 mois, 1 année...).
- Le prestataire le plus connu et qui offre à ce jour l'offre la plus large dans le cloud est **Amazon Web Services (AWS)**.



Amazon Leads \$180-Billion Cloud Market

Worldwide market share of leading cloud infrastructure service providers in Q4 2021*



* Includes platform as a service (PaaS) and infrastructure as a service (IaaS) as well as hosted private cloud services
Source: Synergy Research Group

statista

Briques de l'architecture Big Data avec AWS



IAM

Contrôles d'accès



S3

Stockage

- Images
- Résultats
- Notebook



EMR

**Cluster de calculs
distribués**

- Traitement des images

Configuration de l'environnement de travail

The screenshot shows the AWS IAM console interface. On the left, the navigation pane is visible with sections like 'Gestion des accès' and 'Rapports d'accès'. The main content area displays the 'Informations d'identification de sécurité' for a user named 'leazadikian'. A dropdown menu is open, showing options like 'Compte', 'Organisation', 'Service Quotas', 'Tableau de bord de facturation', 'Informations d'identification de sécurité' (highlighted), and 'Paramètres'. Below the dropdown, the 'Détails du compte' section shows the user's name, email, and ID. The 'Authentification multi-factor (MFA)' section is also visible, along with the 'Clés d'accès' section where a 'Créer une clé d'accès' button is highlighted.

■ Service IAM (Identity and Access Management)

- Création d'un utilisateur
- Gestion des droits (contrôle S3) (Politiques)
- Création d'une paire de clés qui nous permettra de nous connecter devoir saisir systématiquement login/mot de passe (**Informations d'identification de sécurité / Créer une clé d'accès**)



- Installation et configuration de **AWS Cli** (interface en ligne de commande d'AWS, permet d'interagir avec les différents services d'AWS)

Stockage des données sur S3 (Simple Storage Service)



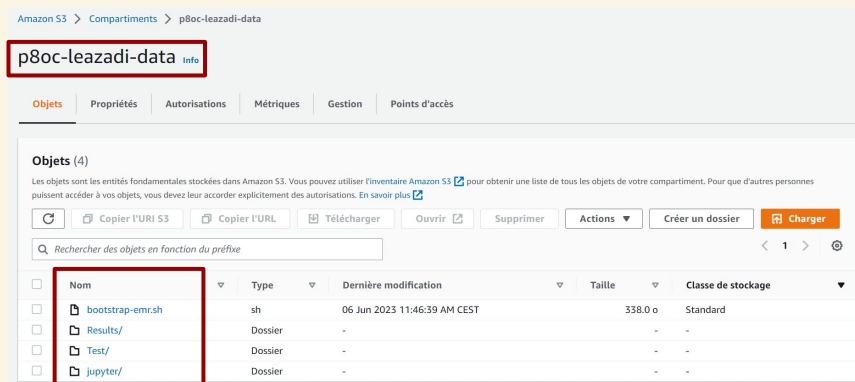
S3 : Solution pour la gestion du stockage des données

- Stockage d'une grande variété d'objets (fichiers, image, vidéos...)
- Évolutivité avec espace disponible illimité.
- Indépendant des serveurs EC2.
- Accès aux données très rapide.
- Possibilité de définir des politiques d'accès IAM pour contrôler les autorisations. d'accès aux buckets et aux objets.
- Chiffrement côté serveur pour sécuriser les données stockées dans S3.
- Classes de stockage (options) adaptées à l'utilisation.



Mise en oeuvre :

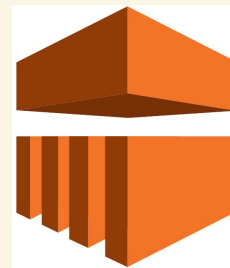
- Création d'un compartiment ("bucket") : **p8oc-leazadi-data**
- Choisir la même région pour les serveurs EC2 et S3.
- Chargement des données sur le bucket S3 :
 - Fichier de configuration avec amorçage
 - Répertoire des images **Test**
 - Notebook avec Script (JupyterHub)
- Écriture des résultats dans le répertoire **Results**.



Création d'un cluster de calculs distribués avec EMR (Elastic MapReduce)

- **Elastic MapReduce (EMR)** : plateforme permettant **d'exécuter des traitements de données distribuées à grande échelle**, en utilisant des frameworks tels que Hadoop et Spark.
- Il utilise des **instances EC2** (Elastic compute cloud, serveur) avec des **applications préinstallées** et configurées pour créer et gérer le cluster de calculs distribués.
- Le service est **entièrement géré par AWS**.

⇒ **Avantages** : évolutivité, flexibilité, gestion simplifiée.



Création du serveur EMR en 4 étapes :

1. Configuration logiciel
2. Configuration matériel
3. Actions d'amorçage
4. Options de sécurité

EMR / 1-Configuration Logiciel



- **Choix des logiciels :**
 - Hadoop et Spark : calculs distribués.
 - TensorFlow : import du modèle et transfert learning.
 - JupyterHub : exécution des scripts Pyspark du Notebook.
- **Paramétrage de la persistance** des notebooks créés et ouverts via JupyterHub (configuration au format JSON).

▼ Paramètres logiciels – facultatif [Info](#)

☒ Entrer la configuration ☐ Charger JSON à partir d'Amazon S3

```
1 {  
2   "Classification": "jupyter-s3-conf",  
3   "Properties": {  
4     "s3.persistence.bucket": "p8oc-leazadi-data",  
5     "s3.persistence.enabled": "true"  
6   }  
7 }  
8 ]  
9 }
```

Configuration JSON de la persistance des Notebooks

Créer un cluster [Info](#)

Nom et applications [Info](#)

Nom

P8_Fruits_cluster

Version Amazon EMR [Info](#)

Une version contient un ensemble d'applications susceptibles d'être installées sur votre cluster.

emr-6.7.0

Offre d'applications



▼ Personnaliser votre offre d'applications

Applications incluses dans l'offre

- | | | |
|---|---|--|
| <input type="checkbox"/> Flink 1.14.2 | <input type="checkbox"/> Ganglia 3.7.2 | <input type="checkbox"/> HBase 2.4.4 |
| <input type="checkbox"/> HCatalog 3.1.3 | <input checked="" type="checkbox"/> Hadoop 3.2.1 | <input type="checkbox"/> Hive 3.13 |
| <input type="checkbox"/> Hue 4.10.0 | <input type="checkbox"/> JupyterEnterpriseGateway 2.1.0 | <input checked="" type="checkbox"/> JupyterHub 1.4.1 |
| <input type="checkbox"/> Livy 0.7.1 | <input type="checkbox"/> MXNet 1.8.0 | <input type="checkbox"/> Oozie 5.2.1 |
| <input type="checkbox"/> Phoenix 5.1.2 | <input type="checkbox"/> Pig 0.17.0 | <input type="checkbox"/> Presto 0.272 |
| <input checked="" type="checkbox"/> Spark 3.2.1 | <input type="checkbox"/> Sqoop 1.4.7 | <input checked="" type="checkbox"/> TensorFlow 2.4.1 |
| <input type="checkbox"/> Tez 0.9.2 | <input type="checkbox"/> Trino 378 | <input type="checkbox"/> Zeppelin 0.10.0 |
| <input type="checkbox"/> ZooKeeper 3.5.7 | | |

EMR / 2- Configuration Matériel



Configuration Matériel (choix des instances) :

- 1 instance **Maître** (driver), 2 instances **principales** (workers)
- Instances de **type M5** (instances de type équilibrées), et **xlarge** (la moins onéreuse disponible).



vCPU : 4 / Mémoire (Gio) : 16
Bande passante réseau (Gbit/s) : jusqu'à 10
Bande passante EBS (Mbit/s) : Jusqu'à 4 750

Configuration de cluster [Info](#)
Choisissez une méthode de configuration pour les groupes de nœuds principaux, principaux et de tâches de votre cluster.

☒ **Groupes d'instances**
Choisir un type d'instance par groupe de nœuds

☐ **Flottes d'instances**
Choisir une combinaison de types d'instance au sein de chaque groupe de nœuds

Groupes d'instances

Primaire
Choisir un type d'instance EC2

m5.xlarge
4 vCore 16 GiB mémoire EBS uniquement stockage
Prix à la demande : 0.204 USD par instance/heure
Prix Spot le plus bas : \$0.057 (eu-north-1b)

Actions ▼

☐ **Utiliser plusieurs nœuds principaux**
Pour améliorer la disponibilité du cluster, utilisez trois nœuds principaux avec les mêmes actions de configuration et d'amorçage. Vous ne pouvez pas utiliser plusieurs nœuds principaux avec des flottes d'instances.

► **Configuration de nœud - facultatif**

Unité principale
Choisir un type d'instance EC2

m5.xlarge
4 vCore 16 GiB mémoire EBS uniquement stockage
Prix à la demande : 0.204 USD par instance/heure
Prix Spot le plus bas : \$0.057 (eu-north-1b)

Actions ▼

Retirer le groupe d'instances

EMR / 3-Actions d'amorçage ou bootstrapping



- Choix des **packages manquants à installer**, utiles pour l'exécution du notebook.
- **A l'initialisation du serveur**, afin que les packages soient installés sur l'ensemble des machines du cluster et non pas uniquement sur le driver.
- Création du fichier "**bootstrap-emr.sh**" contenant commandes "**pip install**" pour installer les bibliothèques manquantes, et chargement sur le compartiment S3 (racine).
- Ajout du script dans les **actions d'amorçage**.

Modifier une action d'amorçage

Nom
Action personnalisée

Emplacement du script
Pour obtenir des performances optimales, stockez les actions d'amorçage personnalisées dans la même région AWS que votre cluster.
s3://p8oc-leazadi-data/bootstrap-emr.sh

Arguments - facultatif
Fournissez des arguments pour vos scripts d'action d'amorçage. Ces arguments envoient des références aux scripts activés à Amazon EMR.
Spécifier les arguments associés à vos scripts

Annuler Enregistrer

```
bootstrap-emr.sh
1  #!/bin/bash
2  sudo python3 -m pip install -U setuptools
3  sudo python3 -m pip install -U pip
4  sudo python3 -m pip install wheel
5  sudo python3 -m pip install pillow
6  sudo python3 -m pip install pandas==1.2.5
7  sudo python3 -m pip install pyarrow
8  sudo python3 -m pip install boto3
9  sudo python3 -m pip install s3fs
10 sudo python3 -m pip install fsspec
```

EMR / 4 - Sécurité



Configuration de sécurité et paire de clés EC2 – facultatif [Info](#)

Configuration de sécurité
Sélectionnez les paramètres de chiffrement, d'authentification, d'autorisation et de service de métadonnées d'instance de votre cluster.

Paire de clés Amazon EC2 pour SSH sur le cluster [Info](#)

Rôle Identity and Access Management (IAM) [Info](#)

Choisissez ou créez une fonction du service et un profil d'instance pour les instances EC2 de votre cluster.

Fonction du service Amazon EMR [Info](#)

La fonction du service est un rôle IAM assumé par Amazon EMR pour mettre en service des ressources et effectuer des actions au niveau du service avec d'autres services AWS.

☒ Choisir une fonction du service existant
Sélectionnez une fonction du service par défaut ou un rôle personnalisé avec des stratégies IAM attachées afin que votre cluster puisse interagir avec d'autres services AWS.

☐ Créer une fonction du service
Laissez Amazon EMR créer une nouvelle fonction du service afin que vous puissiez accorder et restreindre l'accès aux ressources d'autres services AWS.

Fonction du service

Profil d'instance EC2 pour Amazon EMR

Le profil d'instance attribue un rôle à chaque instance EC2 d'un cluster. Le profil d'instance doit spécifier un rôle qui peut accéder aux ressources pour vos étapes et actions d'amorçage.

☒ Choisir un profil d'instance existant
Sélectionnez un rôle par défaut ou un profil d'instance personnalisé avec des stratégies IAM attachées afin que votre cluster puisse interagir avec vos ressources dans Amazon S3.

☐ Choisir un profil d'instance
Laissez Amazon EMR créer un profil d'instance afin de pouvoir spécifier un ensemble personnalisé de ressources auquel il peut accéder dans Amazon S3.

Profil d'instance

- Sélection de la **paire de clés EC2** créée précédemment.
- Permet de se connecter en ssh aux instances EC2 sans avoir à entrer login / mot de passe.

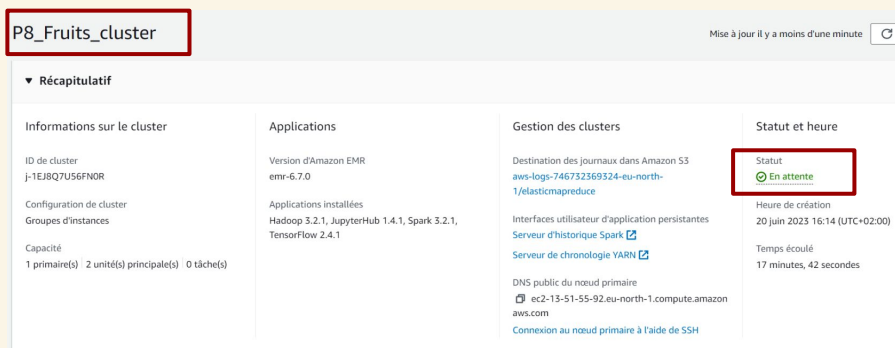
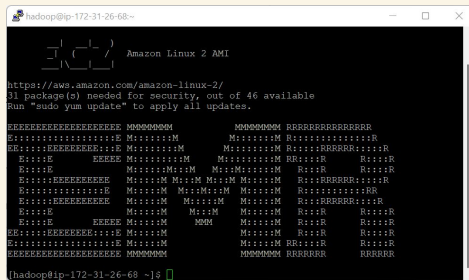
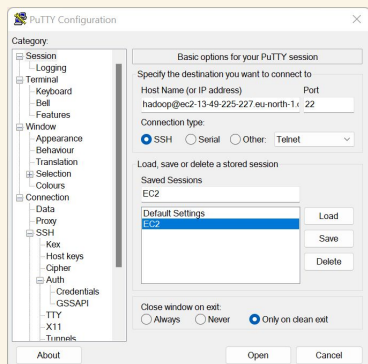
⇒ **Création du cluster,**
instanciation du serveur
(statut "En attente")

Création du tunnel SSH à l'instance EC2

Objectif : pouvoir accéder à nos applications en créant un tunnel SSH vers le driver.

- Modification du **groupe de sécurité EC2** du driver :
 - Autorisation sur les connexions entrantes du driver : **ouverture du port 22** (port d'écoute du serveur SSH).
- Création du **tunnel SSH vers le driver** avec **Putty**.
- Configuration de **SwitchyOmega** : redirection des requêtes vers le port 8157.
- Accès aux applications du serveur EMR via le tunnel SSH

⇒ Connexion au Notebook JupyterHub et exécution du code.



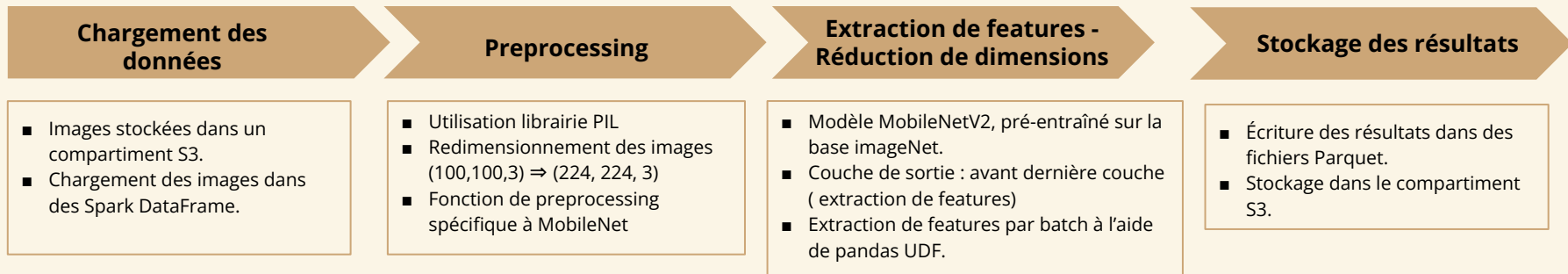
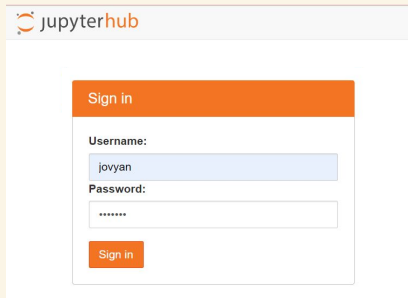


III. Chaîne de traitement d'images dans un environnement Big Data dans le cloud



Chaîne de traitement d'image

- Exécution du Notebook depuis **JupyterHub**, hébergé sur notre serveur EMR.
- Utilisation d'un **kernel pySpark**.
- Démarrage d'une **session Spark** à l'exécution de la première cellule.



Chargement des données

- Chargement des données avec ***spark.read()*** :
 - Traitement des fichiers en tant que **données binaires**.
 - À l'emplacement spécifié (compartiment S3), recherche récursive dans les sous-répertoires des fichiers avec l'extension **".jpg"**.
 - Chargement des images dans un **DataFrame Spark**.

```
root
|-- path: string (nullable = true)
|-- modificationTime: timestamp (nullable = true)
|-- length: long (nullable = true)
|-- content: binary (nullable = true)
|-- label: string (nullable = true)
```

Schéma du Spark DataFrame

- Ajout de la colonne **label** issu du chemin d'accès des fichier :
 - **label** représente la catégorie de l'image (nom du fruit), avant dernier élément (-2) du "path".

path	modificationTime	length	content
s3://p8oc-leazadi...	2023-06-02 13:00:59	7353	[FF D8 FF E0 00 1...
s3://p8oc-leazadi...	2023-06-02 13:00:59	7350	[FF D8 FF E0 00 1...
s3://p8oc-leazadi...	2023-06-02 13:00:59	7349	[FF D8 FF E0 00 1...
s3://p8oc-leazadi...	2023-06-02 13:00:59	7348	[FF D8 FF E0 00 1...
s3://p8oc-leazadi...	2023-06-02 13:00:59	7328	[FF D8 FF E0 00 1...
s3://p8oc-leazadi...	2023-06-02 13:00:59	7301	[FF D8 FF E0 00 1...
s3://p8oc-leazadi...	2023-06-02 13:00:59	7278	[FF D8 FF E0 00 1...
s3://p8oc-leazadi...	2023-06-02 13:00:59	7275	[FF D8 FF E0 00 1...
s3://p8oc-leazadi...	2023-06-02 13:00:59	7266	[FF D8 FF E0 00 1...

path	label
s3://p8oc-leazadi-data/Test/Watermelon/r_106_100.jpg	Watermelon
s3://p8oc-leazadi-data/Test/Watermelon/r_109_100.jpg	Watermelon
s3://p8oc-leazadi-data/Test/Watermelon/r_108_100.jpg	Watermelon
s3://p8oc-leazadi-data/Test/Watermelon/r_107_100.jpg	Watermelon
s3://p8oc-leazadi-data/Test/Watermelon/r_95_100.jpg	Watermelon
s3://p8oc-leazadi-data/Test/Watermelon/r_79_100.jpg	Watermelon
s3://p8oc-leazadi-data/Test/Watermelon/r_84_100.jpg	Watermelon
s3://p8oc-leazadi-data/Test/Watermelon/r_94_100.jpg	Watermelon
s3://p8oc-leazadi-data/Test/Watermelon/r_88_100.jpg	Watermelon

Modèle MobileNetV2 avec Transfer Learning

■ Choix du modèle **MobileNetV2** :

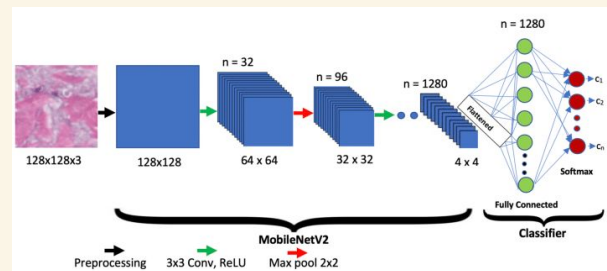
- Modèle de réseau de neurones convolutifs (CNN) pré-entraîné sur la base ImageNet pour la détection de features et la classification d'images.
- Spécialement conçu sur pour **appareils mobiles avec ressources limitées** :
 - Rapidité d'exécution (adapté pour le traitement d'un gros volume de données).
 - Faible dimensionnalité du vecteur de sortie (1,1,1280).

■ **Transfer Learning** :

- Consiste à utiliser la connaissance déjà acquise par un modèle entraîné (ici MobileNetV2) en l'adaptant à notre problématique.
- Création une instance du modèle MobileNetV2 pré-entraîné avec les poids du jeu de données ImageNet, incluant la couche de classification finale.

■ **Préparation du modèle** :

- Création d'un nouveau modèle avec pour couche de sortie l'avant-dernière couche (extraction des features images) du modèle MobileNetV2 .
- Dimension vecteur de sortie (1, 1, 1280).
- **Diffusion des poids** avec `sparkContext.broadcast()` de PySpark :
 - Chargement du modèle sur le driver puis diffusion des poids aux workers.
 - Permet de distribuer une variable à travers le cluster afin qu'elle soit disponible pour tous les nœuds de calcul.



Pre-processing

- Dimensions des **images d'origine** : **(100,100,3)** / (100*100 pixels et 3 canaux de couleur RVB).
- Dimensions des **images attendues en entrée de MobileNetV2** : **(224, 224, 3)**

⇒ **Nous devons les redimensionner avant de les confier en entrée du modèle.**

- Avec **librairie PIL** (Python Imaging Library) :
 - Ouverture des données binaires de l'image en tant qu'image.
 - **Redimensionnement de l'image** à une taille (224, 224,3).
- Application de la fonction *preprocess_input* de TensorFlow, **fonction de prétraitement spécifique** pour prétraiter les images avant de les passer en entrée du modèle MobileNet.



Traitement des données et stockage

■ Extraction de features

- A partir des images pré-traitées, répartition des données et application itérative du modèle aux batches de données d'images pour en extraire les features, en utilisant un pandas UDF.
- Résultat : DataFrame avec colonnes d'origine + features images.
- Les données sont traitées en parallèle sur différents nœuds, ce qui permet d'utiliser la puissance de calcul distribué du cluster. De plus, le modèle est chargé une seule fois et réutilisé pour tous les batches de données, ce qui évite les coûts de chargement répétitifs et réduit la consommation de mémoire.

■ Réduction de dimensions PCA : Analyse en composantes principales pour réduire la dimensionnalité tout en préservant un maximum d'informations.

■ Stockage des résultats

- Données du DataFrame écrites dans un fichier Parquet (format de stockage optimisé pour le Big Data).
- Mode "overwrite" : si le fichier de destination existe déjà, il sera écrasé.
- Dans le répertoire "Results" du compartiment S3.



IV. Démonstration d'exécution du script PySpark sur le cloud



Démonstration d'exécution dans le cloud

Amazon EMR > EMR sur EC2: Clusters > P8_Fruits_cluster

P8_Fruits_cluster Mise à jour il y a moins d'une minute

▼ Récapitulatif

Informations sur le cluster	Applications	Gestion des clusters	Statut et heure
ID de cluster j-1EJBQ7US6FNOR	Version d'Amazon EMR emr-6.7.0	Destination des journaux dans Amazon S3 aws-logs-74673269324-eu-north-1/elasticmapreduce	Statut En attente
Configuration de cluster Groupes d'instances	Applications installées Hadoop 3.2.1, JupyterHub 1.4.1, Spark 3.2.1, TensorFlow 2.4.1	Interfaces utilisateur d'application persistantes Serveur d'historique Spark Serveur de chronologie YARN	Heure de création 20 juin 2023 16:14 (UTC+02:00)
Capacité 1 primaire(s) 2 unité(s) principale(s) 0 tâche(s)		DNS public du nœud primaire ec2-13-51-55-92.eu-north-1.compute.amazonaws.com	Temps écoulé 17 minutes, 42 secondes
		Connexion au nœud primaire à l'aide de SSH	

Propriétés | Actions d'amorçage | Instances | Étapes | **Applications** | Configurations | Surveillance | Événements | Identifications (0)

Interfaces utilisateur d'application

Les applications installées sur votre cluster Amazon EMR publient des interfaces utilisateur en tant que sites web. Vous pouvez les utiliser pour surveiller l'activité du cluster.

☒ Interfaces utilisateur d'application sur le cluster
Les interfaces utilisateur sur le cluster sont disponibles uniquement pendant l'exécution de votre cluster. Utilisez les liens suivants pour démarrer. Pour accéder à toutes les interfaces utilisateur d'application, configurez le tunneling SSH.

☐ Interfaces utilisateur d'application persistantes
Les interfaces utilisateur persistantes ne nécessitent pas de tunneling SSH. Elles sont hébergées hors du cluster disponibles pendant 30 jours après la fin d'une application.

Interfaces utilisateur d'application en direct

Ces interfaces utilisateur d'application sur cluster sont disponibles sans tunneling SSH.

[Interfaces utilisateur d'application](#)

[Interface utilisateur du serveur d'historique Spark](#)

Interfaces utilisateur d'application sur le nœud primaire

Cela réenregistre l'activation du tunneling SSH. Suivez les instructions de la section [Afficher les interfaces web hébergées sur des clusters Amazon EMR](#).

Application	URL de l'interface utilisateur
Gestionnaire de ressources	http://ec2-13-51-55-92.eu-north-1.compute.amazonaws.com:8080/
JupyterHub	https://ec2-13-51-55-92.eu-north-1.compute.amazonaws.com:9443/
Nom du nœud HDFS	http://ec2-13-51-55-92.eu-north-1.compute.amazonaws.com:9870/
Serveur d'historique Spark	http://ec2-13-51-55-92.eu-north-1.compute.amazonaws.com:18080/

jupyterhub LZ_P8_Notebook_Linux_EMR_PySpark... Dernière Sauvegarde : il y a quelques secondes (auto-sauvegardé) Logout Control Panel

File Edit View Insert Cell Kernel Widgets Help Non fiable | PySpark

Exécuter

Déployez un modèle dans le cloud

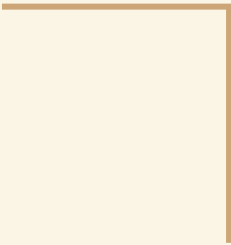
Sommaire :

Conclusion

- **Mise en place d'une architecture Big Data :**
 - **EMR** (Elastic MapReduce) avec Apache Spark pour le traitement distribué des données volumineuses, qui nous permet d'instancier un cluster avec les programmes et bibliothèques nécessaires : Spark, Hadoop, JupyterHub, TensorFlow...
 - **S3** (Simple Storage Service) pour le stockage des données : images d'origine et résultats.
 - **IAM** (Identity & Access Management) pour la gestion des contrôles d'accès.
- Appropriation de la **chaîne de traitement d'images** : chargement des données, preprocessing, préparation du modèle MobileNetV2 avec transfert learning et diffusion des poids, extraction de features, réduction de dimensions.
- **L'utilisation d'un environnement Big Data offre des avantages pour "Fruits!"** en termes de traitement des données, **de performance, d'évolutivité et de préparation pour l'avenir** :
 - Il sera facile de faire face à une montée de la charge de travail et **passer à l'échelle** en redimensionnant le cluster de machines.
 - Les coûts augmenteront en conséquence mais resteront inférieurs aux coûts engendrés par l'achat de matériels ou par la location de serveurs dédiés.
 - L'architecture Big Data jette les bases pour des fonctionnalités avancées, comme **l'entraînement de modèles de classification** des fruits.



Fruits!



Merci pour votre attention !
Des questions ?

