Déployer un modèle dans le Cloud

Soutenance Projet 8
OpenClassrooms - Quérin Yannick - 20/09/2024

Déployer un modèle dans le Cloud

1. Problématique et jeu de données
2. Processus de création de l’environnement Big Data
3. Chaine de traitement d’images dans le cloud
4. Démonstration exécution script Spark dans le cloud

Environnement technique

- Notebook Jupyter 6.4.8
- Python 3.9.12

- Librairies :
  - Pandas, Numpy
  - PIL
  - PySpark
  - TensorFlow

- AWS (Amazon Web Services)
- PuTTY

1. Problématique et jeu de données

Problématique

Fruits!

AGRITECH Entreprise ***Fruits***
- Start-up de l'AgriTech
- L'IA au service de l'agriculture

Phase 1:
application mobile grand public de reconnaissance de fruits par photographie
- Classification d'images (Volume accru d'images)

Phase 2:
Robots cueilleurs intelligents (au sein d'une maison)

Mission

- ❑ **Mettre en place une architecture Big Data**
- ❑ **Préparer les données:**
  - **Pré-processing**
  - **Réduction de dimension**

Contraintes

- ❑ **Anticiper le passage à l'échelle (volume accru, calculs distribués)**
- ❑ **Scripts PySpark**
- ❑ **Déploiement cloud**

Objectifs

- ❑ **Promouvoir la start-up**
- ❑ **Classification d'images pour application mobile**

Jeu de données

90380 Images
131 Classes

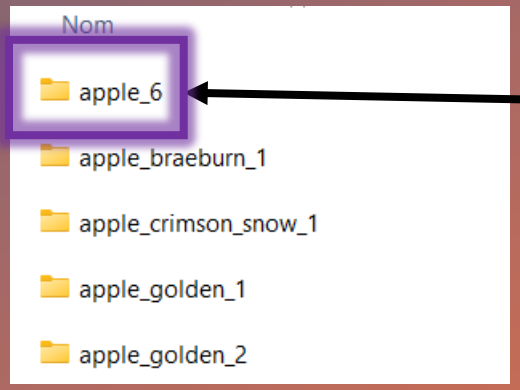
Jeu d'étiquettes

67692 Images
131 Classes

Jeu d'entrainement

22688 Images
131 Classes

Jeu test



- **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…,avec 120 variétés différentes**
- **Un répertoire par classe, avec plusieurs photos du même fruit sous différents angles**
- **Taille des images : 100x100 pixels.**
- **Sur fond blanc uniformisé.**

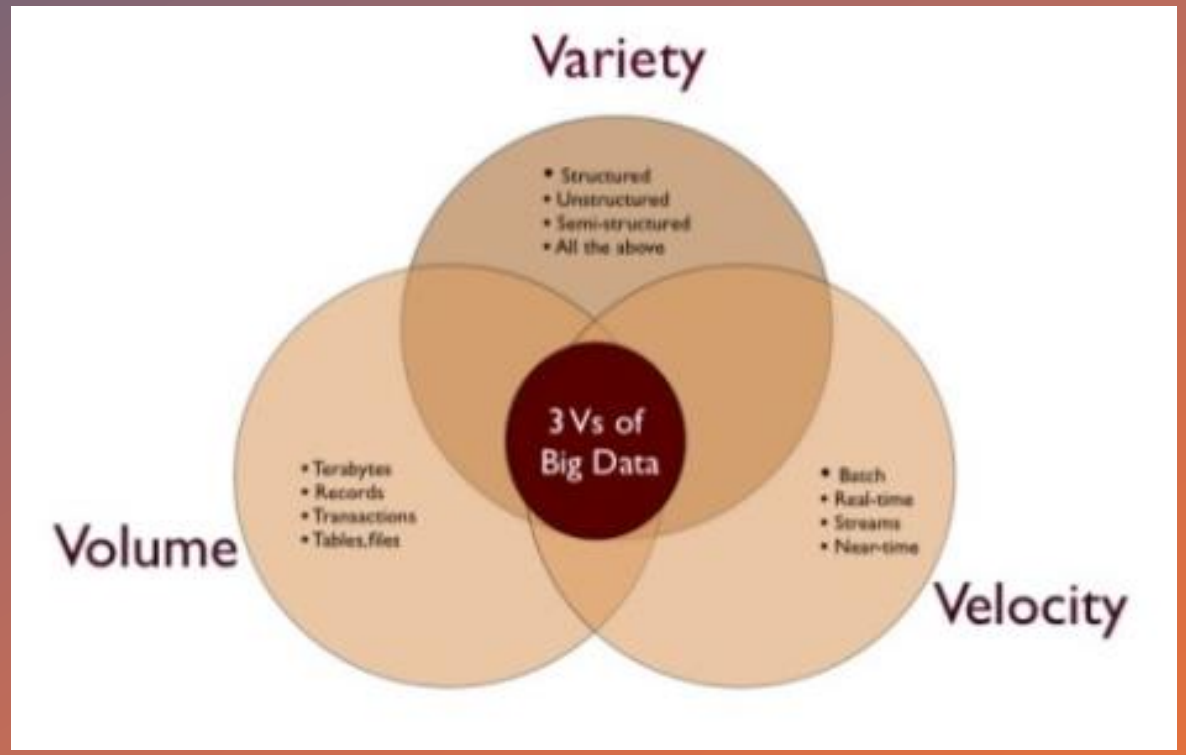
Photo de pommes en 360° sur différents axes

2. Processus de création de l’environnement Big Data

Big Data – Données massives - enjeux

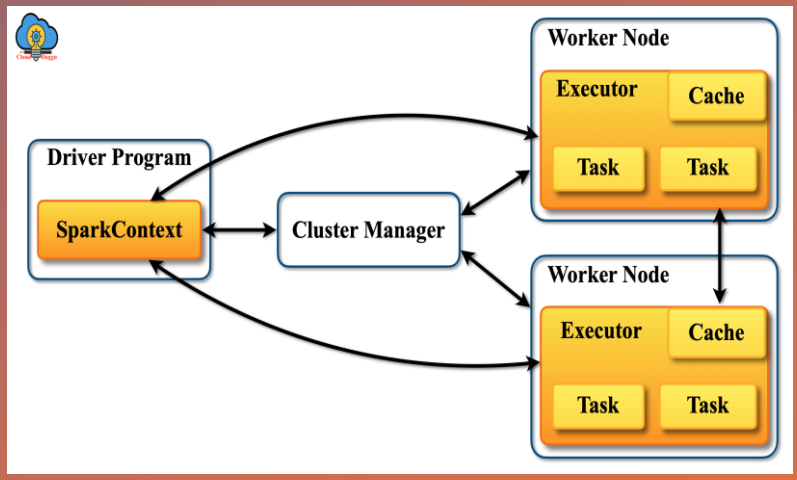
Volume exponentielle de données
Partage des données
Analyse/ Stockage des données
Traitement des flux de données

Big Data



Big Data – Outils et usages

Calculs distribués : distribution du stockage et des traitements 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.



- **Apache 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)**

Big Data – Descriptif solution Cloud

Données

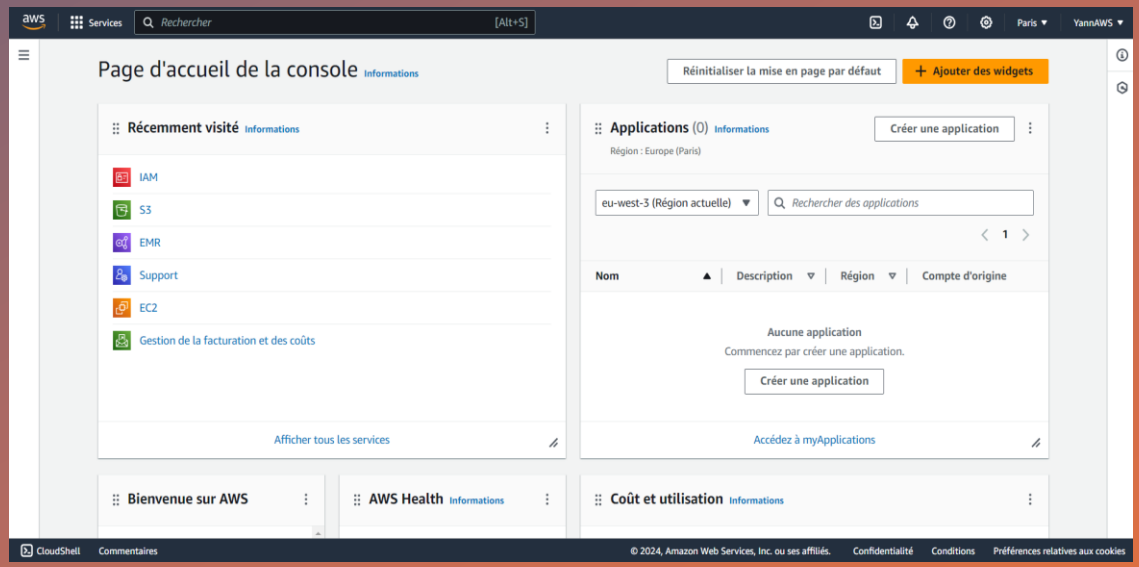
- ❑ Echelle illimitée
- ❑ Durabilité, disponibilité
- ❑ Géo-réplication

Sécurité

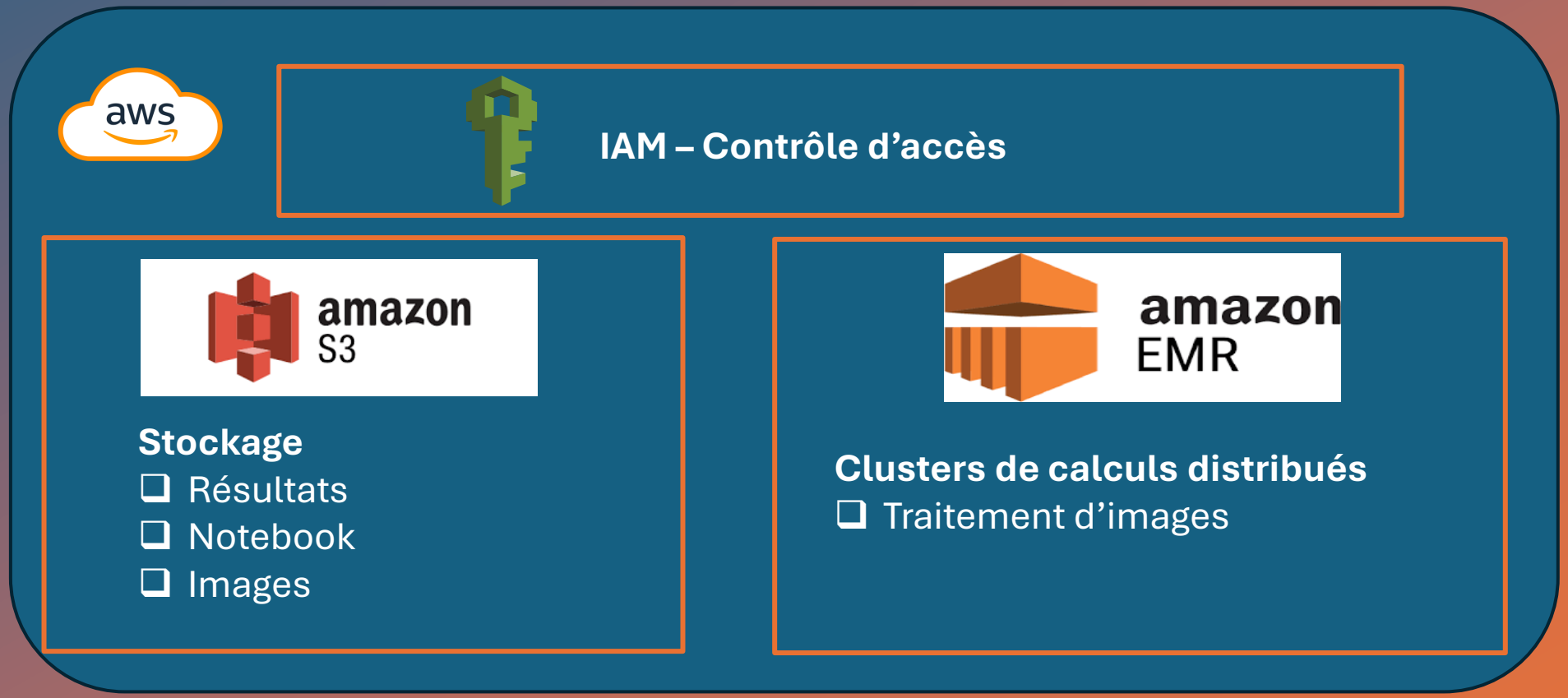
- ❑ Contrôle d’accès, authentification (rôles IAM)
- ❑ Chiffrement et contrôle réseau

Couts

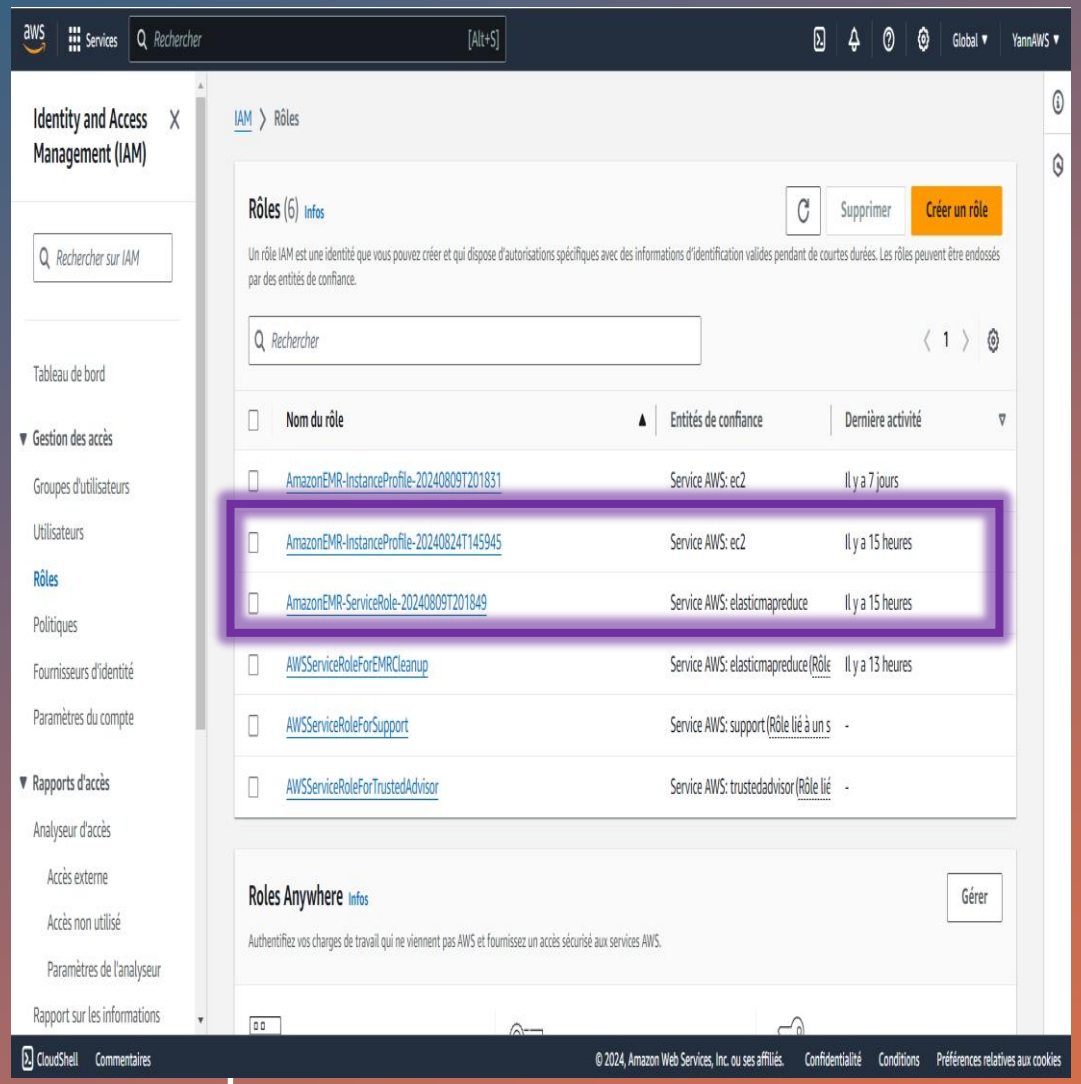
- ❑ Diminution des couts par rapport à un serveur complet



Big Data – Architecture AWS



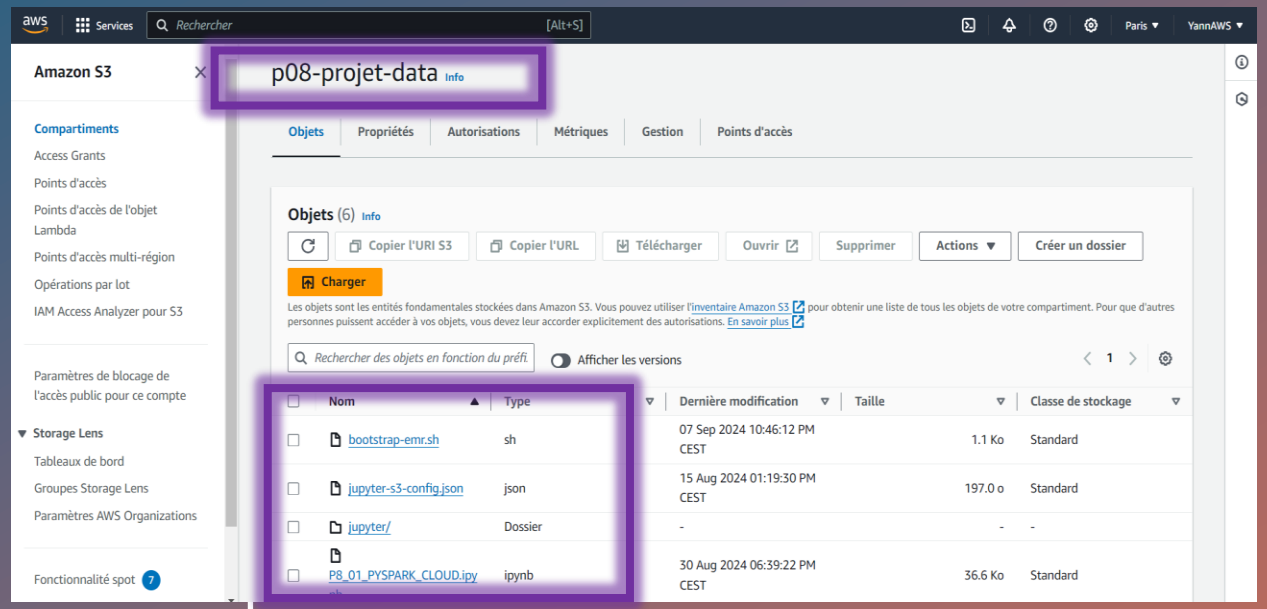
Big Data – Configuration environnement de travail



Service IAM (Identity and Access Management)

- 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
- Ajout des 3 politiques d'autorisations: *AmazonEC2FullAccess*, *AmazonElasticMapReduceFullAccess* et *AmazonS3FullAccess au sein des rôles utilisés (AmazonEMR-ServiceRole-20240809T201849* et *AmazonEMR-InstanceProfile-20240824T145945)* afin de résoudre les problèmes d'autorisations et garantir que votre cluster EMR dispose de toutes les permissions nécessaires pour fonctionner.

Big Data – Stockage sur Amazon S3



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.

Cas pratique:

- ❑ Création d'un compartiment ("bucket") : *p8-projet-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.

Big Data – Création cluster avec EMR (Elastic Map Reduce)

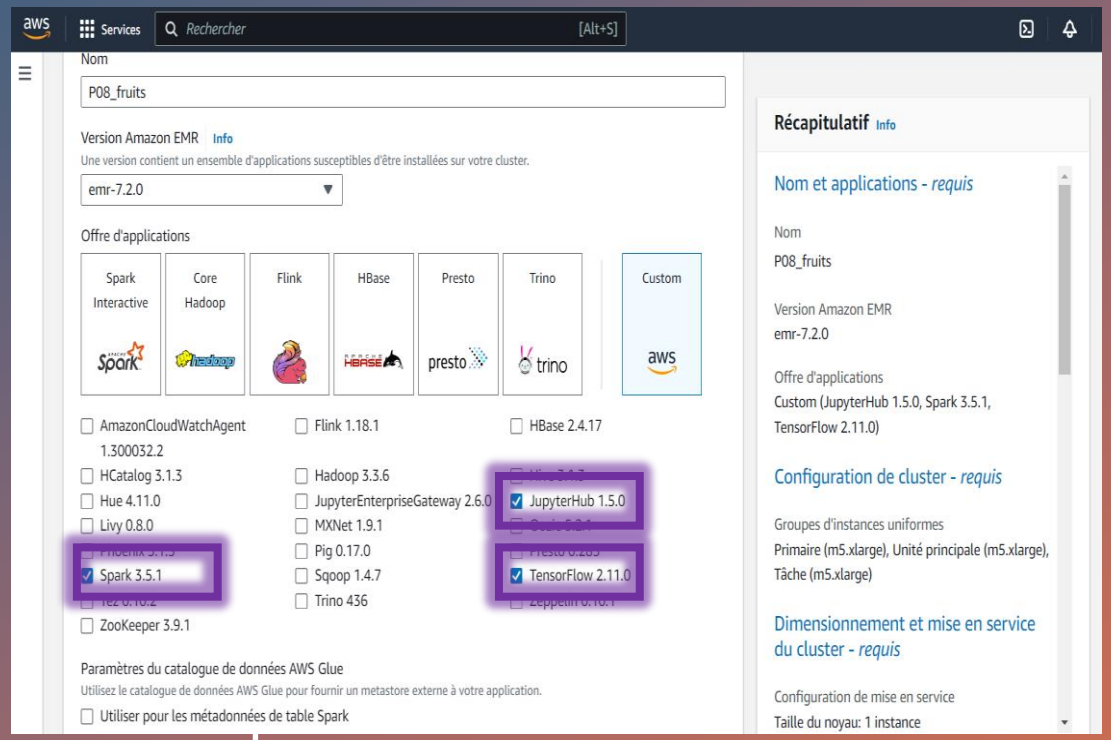
- *Elastic MapReduce* (EMR) : plateforme permettant d'exécuter des traitements de données distribuées à grande échelle, en utilisant des frameworks tels que Hadoop et Spark.
- Il utilise des instances EC2 (*Elastic compute cloud*, serveur) avec des applications préinstallées et configurées pour créer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

Création du serveur EMR en 4 phases:

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

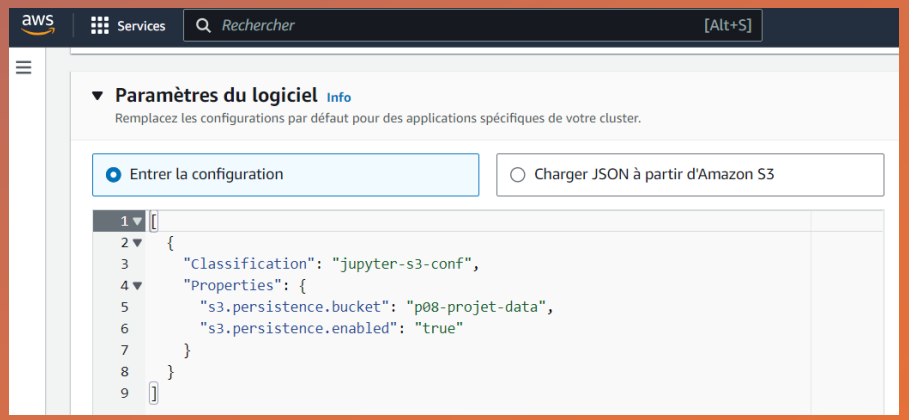
Big Data – Création cluster avec EMR (Elastic Map Reduce)

1. EMR - Configuration logiciel



- **Choix des logiciels :**
  - **Spark : calculs distribués.**
  - **TensorFlow : import du modèle et transfert learning.**
  - **JupyterHub : exécution des scripts Pyspark du Notebook.**

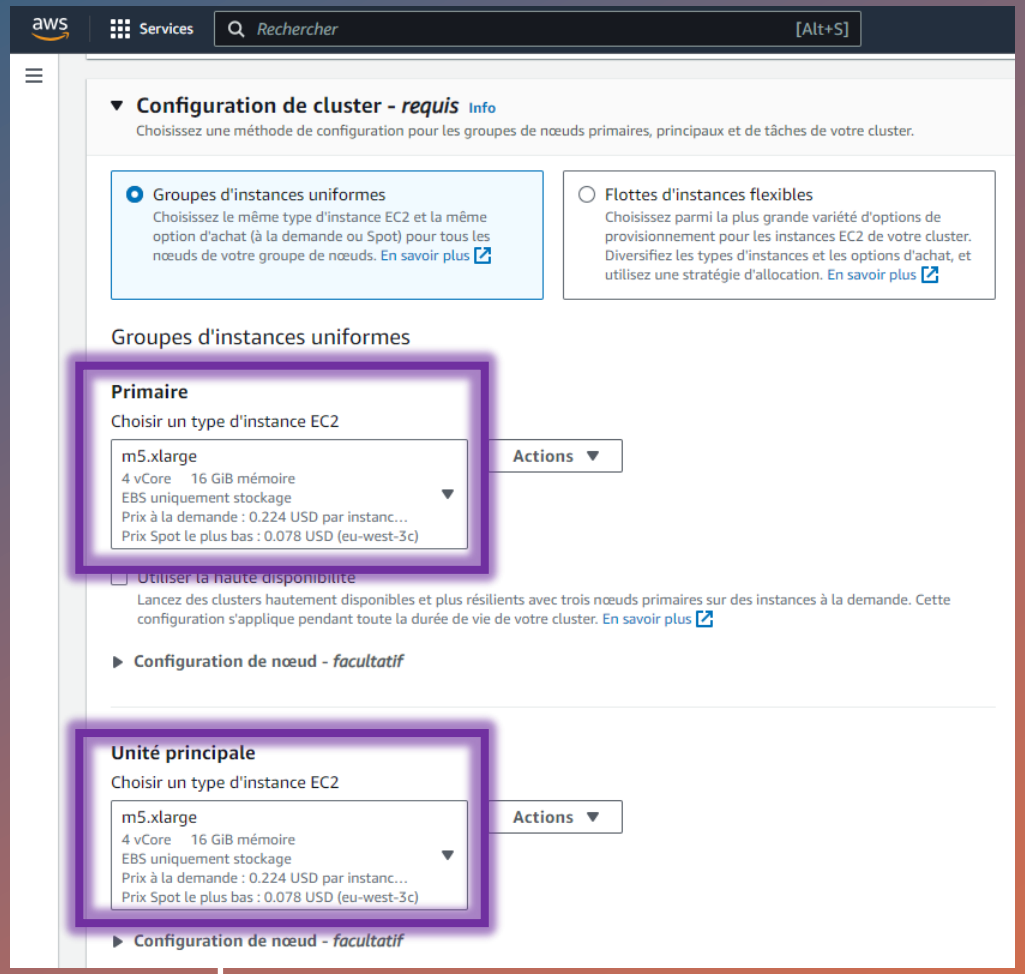
- **Paramétrage de la persistance des notebooks créés et ouverts via JupyterHub (configuration au format JSON)**



Configuration JSON de la persistence des notebooks

Big Data – Création cluster avec EMR (Elastic Map Reduce)

2. EMR - Configuration machine



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).**

Big Data – Création cluster avec EMR (Elastic Map Reduce)

3. EMR – Action d'amorçage (fichier bootstrap)

- ❑ Choix des packages manquants à installer, utiles pour l'exécution du notebook
- ❑ 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

bootstrap-emr.sh | P8_01_PYSPARK_CLOUD.ipynb

C: > Users > yanni > Downloads > $ bootstrap-emr.sh

```
#! /bin/bash

# Mettre à jour pip et setuptools (nécessite sudo pour l'installation globale)
sudo python3 -m pip install --upgrade pip setuptools wheel

# Installer les versions spécifiques des packages nécessaires
sudo python3 -m pip install numpy==1.21.6  # Compatible avec TensorFlow 2.11.0
sudo python3 -m pip install pillow==8.3.1
sudo python3 -m pip install pandas==1.2.5
sudo python3 -m pip install pyarrow==4.0.1
sudo python3 -m pip install boto3==1.18.0
sudo python3 -m pip install s3fs==2021.07.0
sudo python3 -m pip install fsspec==2021.07.0
sudo python3 -m pip install tensorflow==2.11.0
sudo python3 -m pip install pyspark==3.1.2

# Vérifier les versions installées (facultatif)
python3 -c "import numpy as np; import pandas as pd; import PIL; import pyarrow; import boto3; import s3fs; import tensorflow as tf; import pyspark; prin
```

Actions d'amorçage (1) Info

Supprimer | Modifier | Ajouter

Utilisez les actions d'amorçage pour installer des logiciels ou personnaliser la configuration de votre instance.

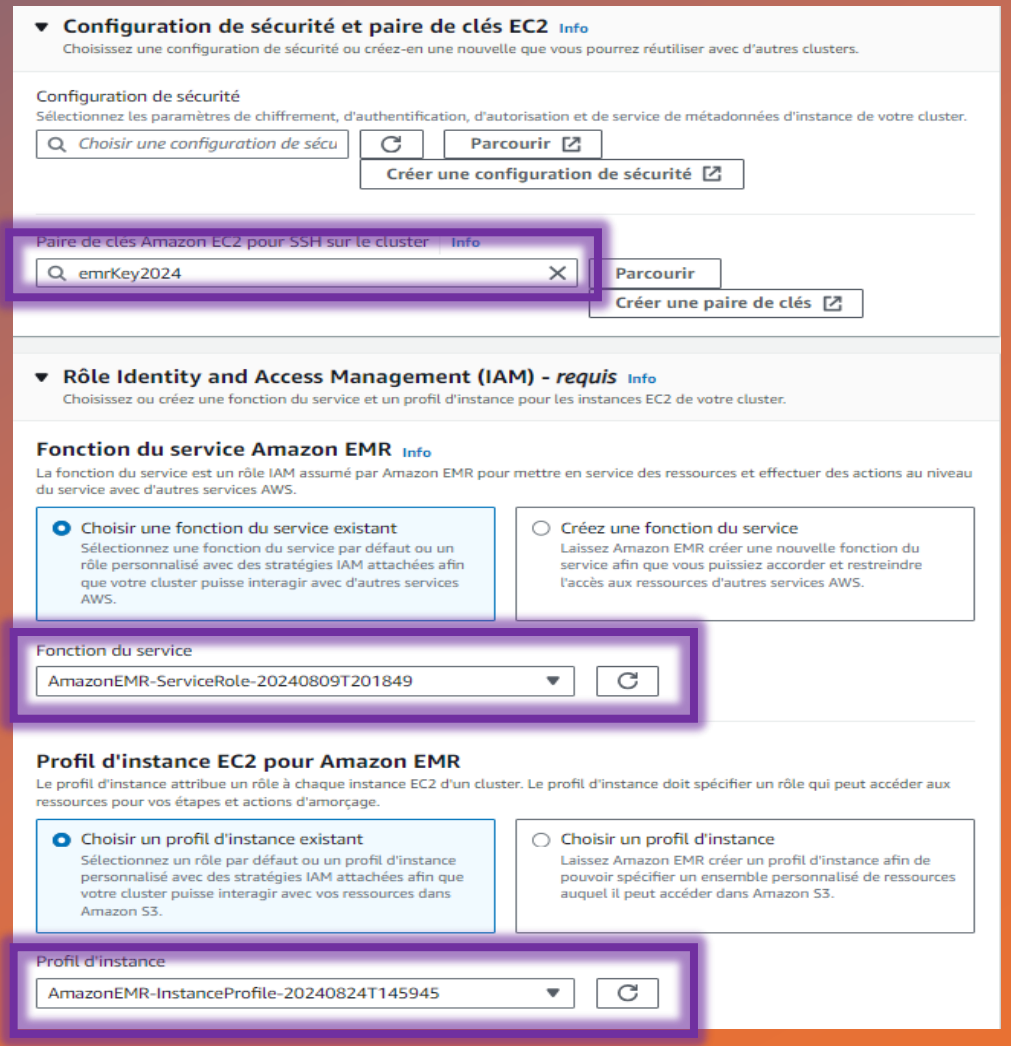
Nom	Emplacement Amazon S3	Arguments
Ajout librairies	s3://p08-projet-data/bootstrap-emr.sh	-

Big Data – Création cluster avec EMR (Elastic Map Reduce)

4. EMR – Sécurité

- ❑ **Sélection de la paire de clés EC2 créée dans la partie ‘Réseau et Sécurité’ de l’instance EC2.**
- ❑ **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”)

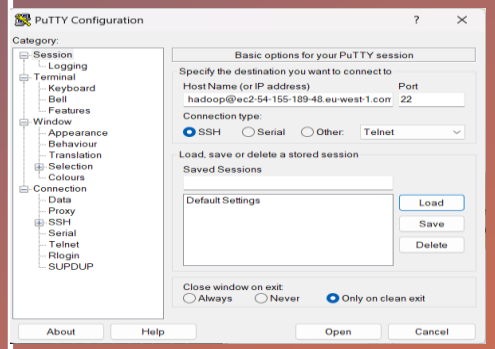


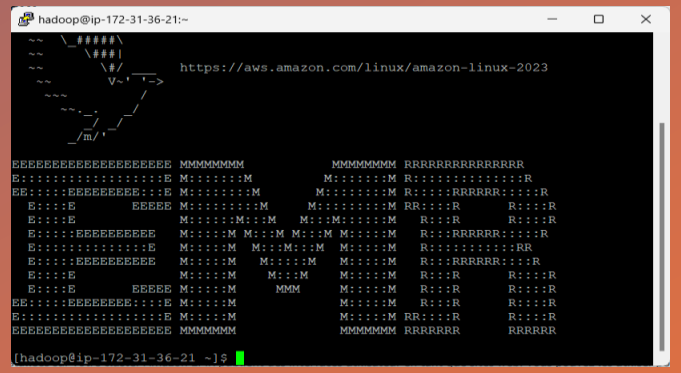
Big Data – Création tunnel SSH (puTTY) sur l'EC2

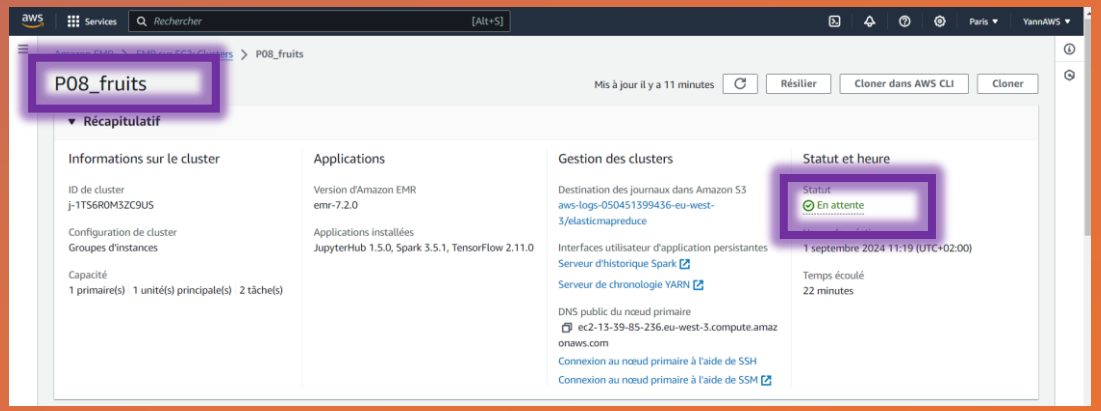
But: accès aux applications (JupyterHub, …) en créant un tunnel SSH vers le driver.

- ❑ **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) pour le *HostName*, et choix arbitraire d'un autre port (ex 8157) dans la section '*SSH/Tunnels*', et chargement de la clé SSH dans '*SSH/Auth/Credentials*'**

- ❑ **Création du tunnel SSH vers le driver avec PuTTY.**
- ❑ **Configuration de FoxyProxy: redirection des requêtes vers le port 8157 (similaire au port tunnel PuTTY).**
- ❑ **Accès aux applications du serveur EMR via le tunnel SSH**







3. Chaine de traitement d’images dans le cloud

Traitement d'images

- 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 datasets de Amazon S3

- Datasets stockés dans Amazon S3
- Chargement images dans des Spark Dataframe (après démarrage session)

Préprocessing

- Utilisation librairie PIL
- Redimensionnement des images (100,100,3) ⇒ (224, 224, 3)
- Fonction de preprocessing spécifique à MobileNet

Extraction features, Réduction dimensions

- Modèle MobileNetV2, pré-entraîné sur la base imageNet.
- Couche de sortie : avant dernière couche (extraction de features)
- Extraction de features par batch à l'aide de pandas UDF

Stockage résultats

- Écriture des résultats dans des fichiers Parquet.
- Stockage dans le compartiment S3.

Chargement datasets

- **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.**

```
+--------------------+------------------+------+--------------------+----------+
|                path|  modificationTime|length|             content|     label|
+--------------------+------------------+------+--------------------+----------+
|s3://p08-projet-d...|2024-08-09 16:55:30|125135|[FF D8 FF E0 00 1...|apple_hit_1|
|s3://p08-projet-d...|2024-08-09 16:55:31|124785|[FF D8 FF E0 00 1...|apple_hit_1|
|s3://p08-projet-d...|2024-08-09 16:55:27|123514|[FF D8 FF E0 00 1...|apple_hit_1|
|s3://p08-projet-d...|2024-08-09 16:55:36|122958|[FF D8 FF E0 00 1...|apple_hit_1|
|s3://p08-projet-d...|2024-08-09 16:55:29|122807|[FF D8 FF E0 00 1...|apple_hit_1|
+--------------------+------------------+------+--------------------+----------+
only showing top 5 rows
```

```
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 du champ label issu du chemin d'accès des fichiers : *label* représente la catégorie de l'image (nom du fruit) contenu dans le chemin *path***

```
+--------------------------------------------------+----------+
|path                                              |label     |
+--------------------------------------------------+----------+
|s3://p08-projet-data/Test/apple_hit_1/r0_115.jpg  |apple_hit_1|
|s3://p08-projet-data/Test/apple_hit_1/r0_119.jpg  |apple_hit_1|
|s3://p08-projet-data/Test/apple_hit_1/r0_107.jpg  |apple_hit_1|
|s3://p08-projet-data/Test/apple_hit_1/r0_143.jpg  |apple_hit_1|
|s3://p08-projet-data/Test/apple_hit_1/r0_111.jpg  |apple_hit_1|
+--------------------------------------------------+----------+
only showing top 5 rows
```

Modèle MobileNetV2 avec méthode 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, développée pour les applications mobiles
  - Pouvant être utilisé pour des applications en temps réel sur des images de fruits, comme la reconnaissance ou la classification.

- **Transfer Learning :**
  - Consiste à utiliser la connaissance déjà acquise par un modèle entraîné (ici MobileNetV2) en l'adaptant à notre problématique.

- **Mise en pratique:**
  - Via ce script, ci-dessus: récupération de l'avant dernière sortie du modèle MobileNetV2 -> vecteur de dimension 1280
  - Diffusion des poids du nouveau modèle sur workers.

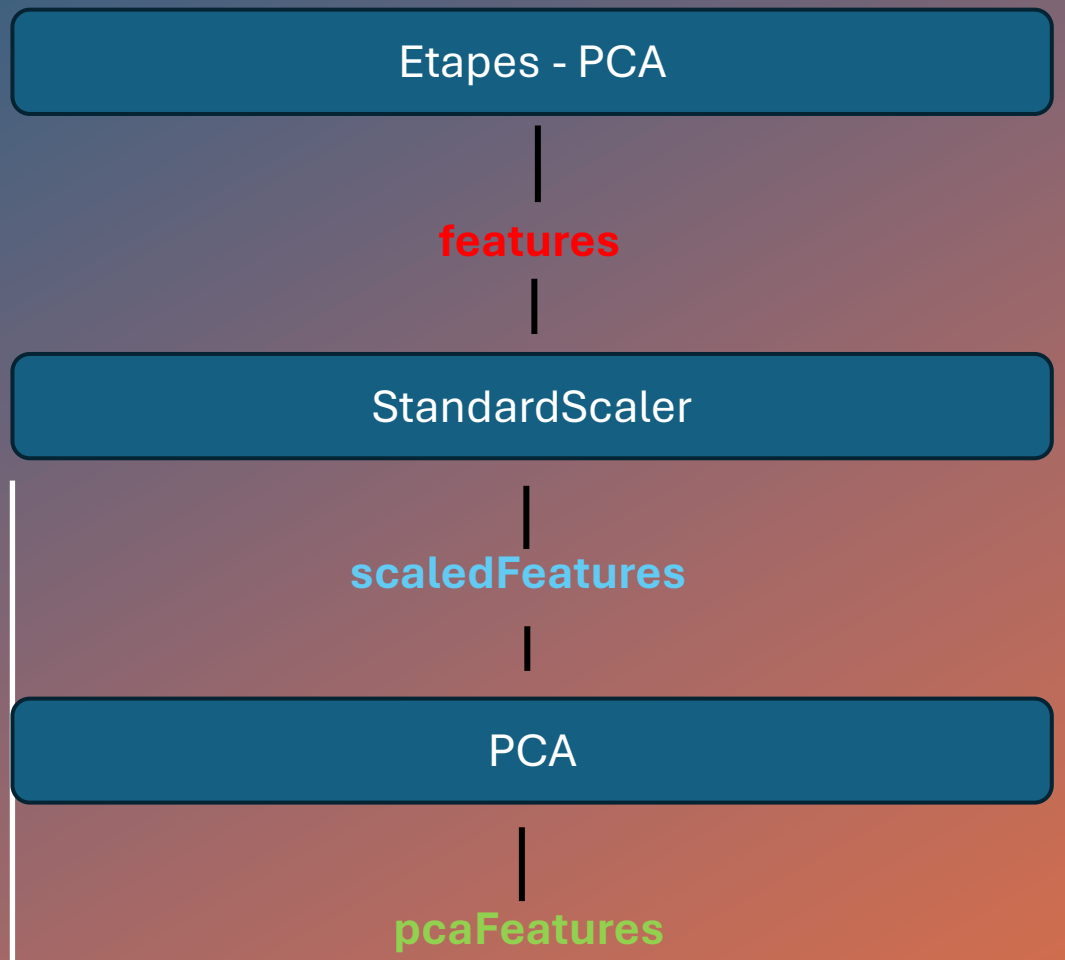
```python
def model_fn():
    """
    Returns a MobileNetV2 model with top layer removed
    and broadcasted pretrained weights.
    """
    model = MobileNetV2(weights='imagenet',
                        include_top=True,
                        input_shape=(224, 224, 3))
    for layer in model.layers:
        layer.trainable = False
    new_model = Model(inputs=model.input,
                      outputs=model.layers[-2].output)
    new_model.set_weights(brodcast_weights.value)
    return new_model
```

Pré-processing

- ❑ **Redimensionnement de l'image d'origine (100,100,3) / (100*100 pixels et 3 canaux de couleur RVB) à une taille (224, 224,3) conforme aux images d'entrée du modèle MobileNetV2.**

- ❑ **Mise en pratique:**
- **Usage de la librairie PIL: création de labels sur les données binaires d'images, et son redimensionnement.**
- **Usage de la fonction '*preprocess_input*' de tensorflow: phase de prétraitement des images avant de les passer en paramètre du modèle.**

Traitement dans le cloud - PCA

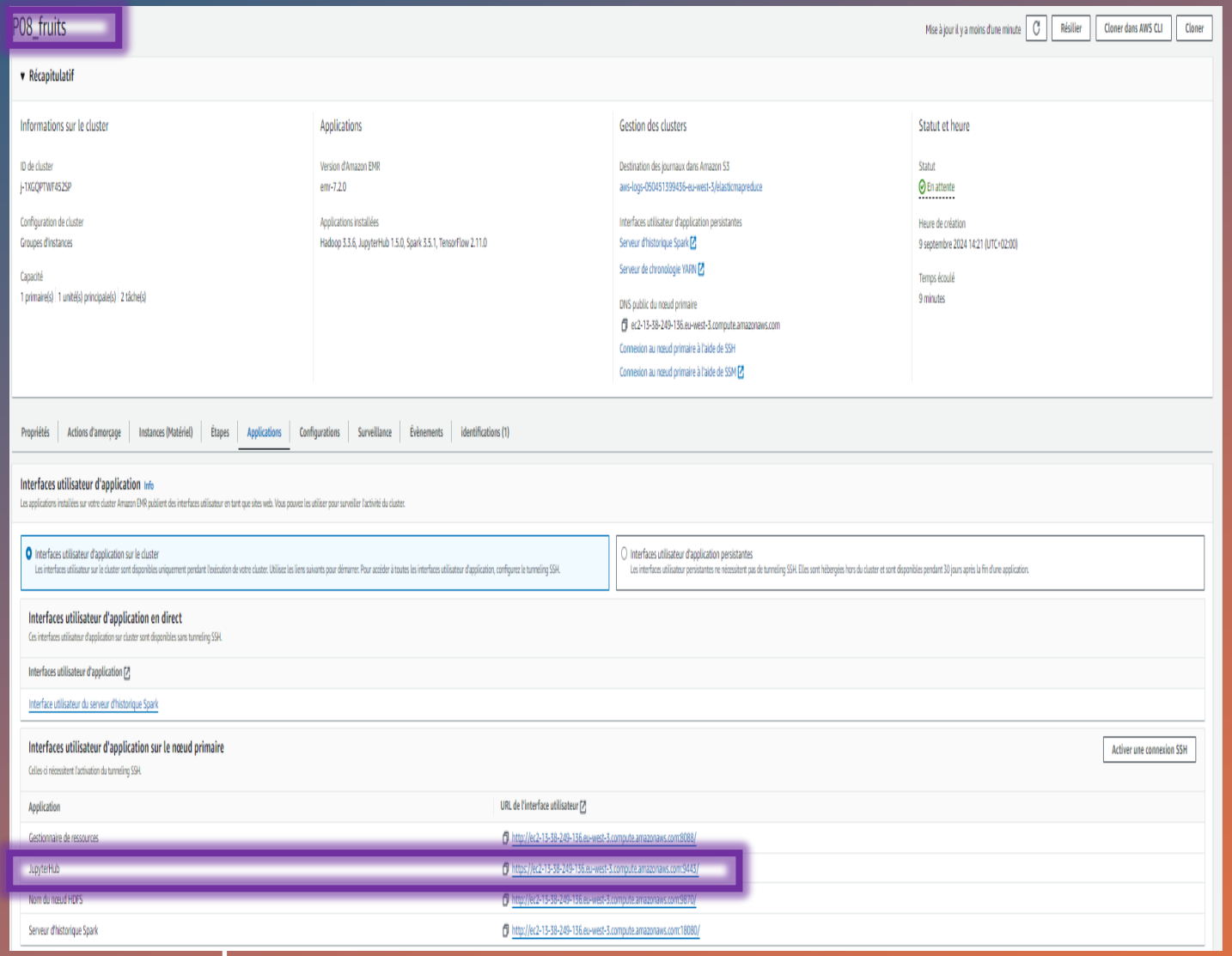


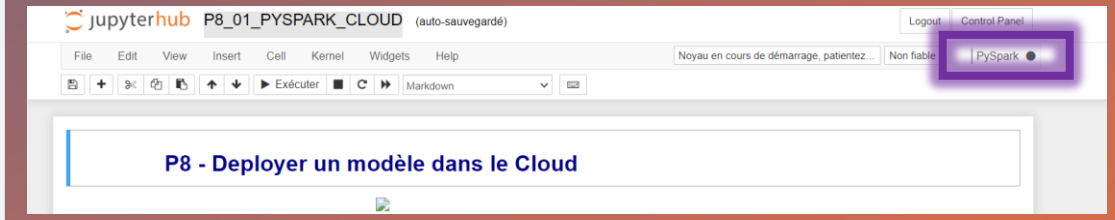
path	label	features	scaledFeatures	pcaFeatures
s3://p08-projet-d...	apple_hit_	[0.24259111285209...	0.3028012314020...	5.71751852949535...
s3://p08-projet-d...	apple_hit_	[0.98785418272018...	.42220802088516...	11.5832112283793...
s3://p08-projet-d...	apple_hit_	[0.14053782820701...	0.5390169616007...	3.74341504811110...
s3://p08-projet-d...	cabbage_white_	[0.0,0.5088325142...	0.8643102221354...	11.9971760288034...
s3://p08-projet-d...	apple_hit_	[0.94111198186874...	.31401705825599...	-1.3326905009276...
s3://p08-projet-d...	apple_hit_	[0.00654815183952...	0.8491536647193...	5.01081513848440...
s3://p08-projet-d...	apple_hit_	[1.00341403484344...	.45822334213487...	7.48031783818804...
s3://p08-projet-d...	pear_	[0.40664485096931...	).07692269101720...	-6.9077826873077...
s3://p08-projet-d...	pear_	[0.56835627555847...	).45122502115441...	-4.7300163366168...
s3://p08-projet-d...	pear_	[0.70505595207214...	).76763437552171...	-6.3404330908477...

only showing top 10 rows

4. Démonstration execution script spark dans le Cloud

Démonstration execution dans le Cloud





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 librairies 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 :
  - Accompagnement facilité de la montée en charge avec redimensionnement horizontal (nombre d'instances) et/ou vertical (puissance des clusters).