

The background is a gradient of dark blue and purple, transitioning from a lighter purple at the top to a darker blue at the bottom. It is decorated with various abstract circular elements: concentric circles, dashed lines, and solid lines with arrows indicating a clockwise direction. Some of these circles have numerical labels like 150, 160, 170, 180, 190, 200, 210, 220, 230, 240, 250, and 260. The overall aesthetic is futuristic and technical, resembling a space-themed interface or a complex data visualization.

# DÉPLOYEZ UN MODÈLE DANS LE CLOUD

HAMIDI MOHAMED

# PROBLEMATIQUE

- L'entreprise Fruits veut développer une application mobile :
  - Sensibiliser le grand public à la biodiversité de fruits
  - Mettre en place une première version du moteur de classification
- Featurisation de 22 688 images, beaucoup trop lourd en local

# LES DONNÉES

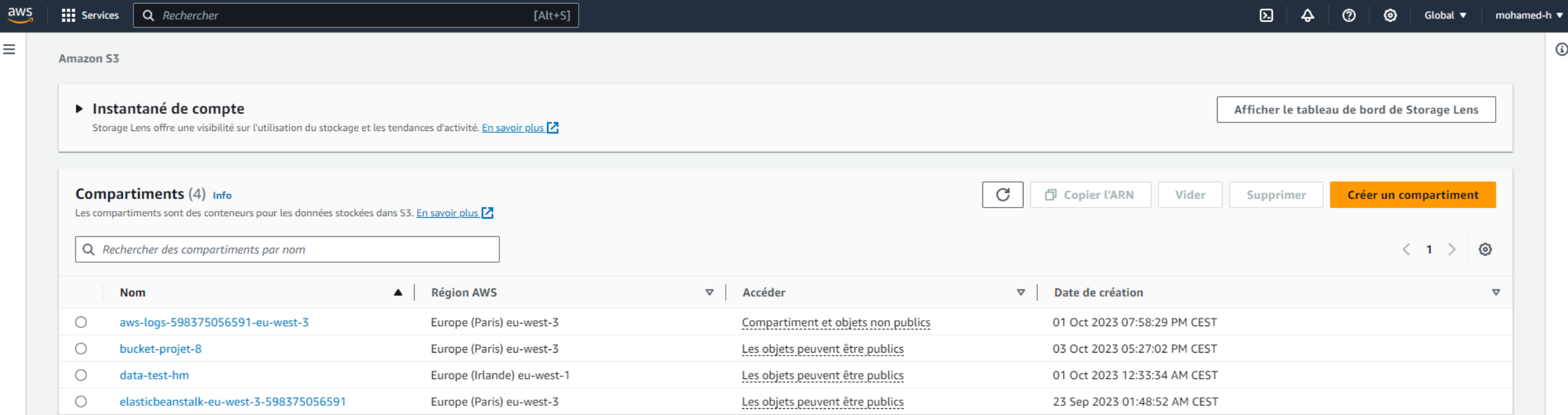
- Document laissé par l'alternant
- Jeu de données composé de 131 dossiers (les fruits), chaque dossier possédant autour des 160 images (pour l'entraînement)

# ENVIRONNEMENT DE TRAVAIL

- Installation d'une machine virtuel : Travailler dans un environnement Linux
- Installation et configuration de AWS Cli
- Création d'une paire de clés : Evite d'avoir à entre login + mot de passe



# CHARGEMENT DES DONNÉES SUR S3



The screenshot shows the Amazon S3 console interface. At the top, there's a navigation bar with the AWS logo, 'Services' menu, a search bar containing 'Rechercher', and a '[Alt+S]' shortcut. On the right, there are icons for help, notifications, and settings, along with 'Global' and a user profile 'mohamed-h'. Below the navigation bar, the main content area is titled 'Amazon S3'. It features a 'Compartiments (4) Info' section with a description: 'Les compartiments sont des conteneurs pour les données stockées dans S3. En savoir plus'. To the right of this section are buttons: 'Copier l'ARN', 'Vider', 'Supprimer', and a prominent orange 'Créer un compartiment' button. Below the description is a search bar 'Rechercher des compartiments par nom'. A table lists the four buckets with columns for 'Nom', 'Région AWS', 'Accéder', and 'Date de création'. The 'bucket-projet-8' bucket is highlighted in blue.

Nom	Région AWS	Accéder	Date de création
aws-logs-598375056591-eu-west-3	Europe (Paris) eu-west-3	Compartiment et objets non publics	01 Oct 2023 07:58:29 PM CEST
bucket-projet-8	Europe (Paris) eu-west-3	Les objets peuvent être publics	03 Oct 2023 05:27:02 PM CEST
data-test-hm	Europe (Irlande) eu-west-1	Les objets peuvent être publics	01 Oct 2023 12:33:34 AM CEST
elasticbeanstalk-eu-west-3-598375056591	Europe (Paris) eu-west-3	Les objets peuvent être publics	23 Sep 2023 01:48:52 AM CEST

- Création d'un bucket directement sur S3
- Accès à la console AWS Cli :
  - On se place à l'adresse bucket-projet-8/Test
  - On applique la commande : `aws sync . s3://bucket-projet-8/Test` (Pour synchroniser les répertoires)

# CONFIGURATION DU SERVEUR EMR

Amazon EMR > EMR sur EC2: Clusters > Créer un cluster

Cloner « Fruits »

Info

Nom et applications

Info

Nom

Fruits

Version Amazon EMR

Info

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

emr-6.13.0

Offre d'applications

Spark

Core Hadoop

Flink

HBase

Presto

Trino

Custom

☐ Flink 1.17.0

☐ HCatalog 3.1.3

☐ Hue 4.11.0

☐ Livy 0.7.1

☐ Phoenix 5.1.3

☒ Spark 3.4.1

☐ Tez 0.10.2

☐ ZooKeeper 3.5.10

☐ Ganglia 3.7.2

☒ Hadoop 3.3.3

☐ JupyterEnterpriseGateway 2.6.0

☐ MXNet 1.9.1

☐ Pig 0.17.0

☐ Sqoop 1.4.7

☐ Trino 414

☐ HBase 2.4.17

☐ Hive 3.1.3

☒ JupyterHub 1.5.0

☐ Oozie 5.2.1

☐ Presto 0.281

☒ TensorFlow 2.11.0

☒ Zeppelin 0.10.1

Configuration de cluster

Info

Choisissez une méthode de configuration pour les groupes de nœuds primaires, 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

m5.2xlarge

8 vCore 32 GiB mémoire EBS uniquement stockage

Prix à la demande : 0.448 USD par instance/heure

Prix Spot le plus bas : \$0.129 (eu-west-3a)

Actions

☐ Utiliser plusieurs nœuds primaires

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

► Configuration de nœud - facultatif

Unité principale

Choisir un type d'instance EC2

m5.2xlarge

8 vCore 32 GiB mémoire EBS uniquement stockage

Prix à la demande : 0.448 USD par instance/heure

Prix Spot le plus bas : \$0.129 (eu-west-3a)

Actions

Retirer le groupe d'instances

► Configuration de nœud - facultatif

# CONFIGURATION DU SERVEUR EMR (PARE-FEU)

Nom	Type d'instance	Taille de l'instance(s)	Utiliser l'option d'achat Spot
Unité principale	m5.2xlarge	2	<input type="checkbox"/>

Règles entrantes (12)

Filtrer les règles des groupes de sécurité

1

<input type="checkbox"/>	Name	ID de règle de grou...	Version IP	Type	Protocole	Plage de ports	Source	Description
<input type="checkbox"/>	-	sgr-09f2dbc667b876a6b	-	Tous les UDP	UDP	0 - 65535	sg-02a0ad3fb371f14c...	-
<input type="checkbox"/>	-	sgr-0b4f6bfe47c0e1591	IPv6	SSH	TCP	22	::/0	-
<input type="checkbox"/>	-	sgr-030fd691edc1bbe5f	-	Tous les TCP	TCP	0 - 65535	sg-02a0ad3fb371f14c...	-
<input type="checkbox"/>	-	sgr-0c7b02d3d033ebe...	IPv4	TCP personnalisé	TCP	8888	15.237.36.45/32	-
<input type="checkbox"/>	-	sgr-091d80676824c8...	IPv4	SSH	TCP	22	0.0.0.0/0	-

Actions d'amorçage

Les actions d'amorçage sont des scripts exécutés lors de la configuration avant le démarrage de Hadoop sur chaque nœud de cluster. Vous pouvez les utiliser pour installer des logiciels supplémentaires et personnaliser vos applications. [En savoir plus](#)

Type d'action d'amorçage	Nom	Emplacement JAR	Arguments facultatifs
Action personnalisée	Action personnalisée	s3://p8-data/bootstrap-emr.sh	

Ajouter une action d'amorçage

Sélectionner une action d'amorçage

Sélectionner une action d'amorçage

Exécuter si

Action personnalisée

Configurer et ajouter

Annuler

Précédent

Suivant

# CONFIGURATION DU SERVEUR EMR

```
#!/bin/bash
sudo python3 -m pip install -U setuptools
sudo python3 -m pip install -U pip
sudo python3 -m pip install wheel
sudo python3 -m pip install pillow
sudo python3 -m pip install pandas==1.2.5
sudo python3 -m pip install pyarrow
sudo python3 -m pip install boto3
sudo python3 -m pip install s3fs
sudo python3 -m pip install fsspec
```



Fichier de Bootstrapping

## 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.



Parcourir [↗](#)

Créer une configuration de sécurité [↗](#)

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



Parcourir

Créer une paire de clés [↗](#)



# CONFIGURATION DU SERVEUR EMR (IAM)

Politiques des autorisations (2) [Infos](#)

Simuler [↗](#)

Supprimer

Ajouter des autorisations ▼

Rechercher

Filtrer par Type

Tous les types ▼

< 1 > ⚙

<input type="checkbox"/>	Nom de la politique <a href="#">↗</a>	Type	Entités attachées
<input type="checkbox"/>	<a href="#">+ AmazonEMR-InstanceProfile-Policy-20231003T183038</a>	Gérées par le client	1
<input type="checkbox"/>	<a href="#">+ AmazonS3FullAccess</a>	Gérées par AWS	4

Politiques des autorisations (3) [Infos](#)

Simuler [↗](#)

Supprimer

Ajouter des autorisations ▼

Rechercher

Filtrer par Type

Tous les types ▼

< 1 > ⚙

<input type="checkbox"/>	Nom de la politique <a href="#">↗</a>	Type	Entités attachées
<input type="checkbox"/>	<a href="#">+ AmazonEMR-ServiceRole-Policy-20231003T183424</a>	Gérées par le client	1
<input type="checkbox"/>	<a href="#">+ AmazonEMRServicePolicy_v2</a>	Gérées par AWS	2

# LANCEMENT DU SERVEUR EMR

Amazon EMR > EMR sur EC2: Clusters > Fruits

Fruits Mise à jour il y a moins d'une minute 🔄 Résilier Cloner dans AWS CLI Cloner

▼ Récapitulatif

<b>Informations sur le cluster</b>  ID de cluster j-3QU2VDIN72P8V  Configuration de cluster Groupes d'instances  Capacité 1 primaire(s) 2 unité(s) principale(s) 0 tâche(s)	<b>Applications</b>  Version d'Amazon EMR emr-6.13.0  Applications installées Hadoop 3.3.3, JupyterHub 1.5.0, Spark 3.4.1, TensorFlow 2.11.0, Zeppelin 0.10.1	<b>Gestion des clusters</b>  Destination des journaux dans Amazon S3 <a href="#">aws-logs-598375056591-eu-west-3/elasticmapreduce</a>  DNS public du nœud primaire <a href="#">ec2-15-237-100-218.eu-west-3.compute.amazonaws.com</a> <a href="#">Connexion au nœud primaire à l'aide de SSH</a>	<b>Statut et heure</b>  Statut <a href="#">🔄 Action d'amorçage</a>  Heure de création 6 octobre 2023 17:19 (UTC+02:00)  Temps écoulé 2 minutes, 57 secondes
--	---	---	--

[Propriétés](#) | [Actions d'amorçage](#) | [Instances \(Matériel\)](#) | [Étapes](#) | [Applications](#) | [Configurations](#) | [Surveillance](#) | [Événements](#) | [identifications \(1\)](#)

<b>Système d'exploitation</b> <a href="#">Info</a>  Version Amazon Linux : 2.0.20230906.0	<b>Journaux de cluster</b> <a href="#">Info</a>  Archiver les fichiers journaux dans Amazon S3 Activé  Emplacement Amazon S3 <a href="#">s3://aws-logs-598375056591-eu-west-3/elasticmapreduce/</a> <a href="#">🔗</a>  Chiffrement pour les journaux Désactivé	<b>Résiliation du cluster</b> <a href="#">Info</a> <span>Modifier la résiliation du cluster</span>  Option de résiliation Résilier automatiquement le cluster après le temps d'inactivité  Temps d'inactivité 3 heures  Protection contre la résiliation Désactivé
--	---	---

Cloner

Résilier

Exporter AWS CLI

Cluster : p8-fruits

En attente Cluster ready after last step completed.

Récapitulatif

Historique de l'application

Surveillance

Matériel

Configurations

Événements

Étapes

Actions d'amorçage

```
(base) walduch@walduch-VirtualBox:~/Documents/P8/bootstrap-emr$ ssh -i ~/.ssh/p8-ec2.pem -D 5555 hadoop@ec2-35-180-23-35.eu-west-3.compute.amazonaws.com
The authenticity of host 'ec2-35-180-23-35.eu-west-3.compute.amazonaws.com (35.180.23.35)' can't be established.
ECDSA key fingerprint is SHA256:nIOYfNaCcJirJvaZ2OH0dadZsk95RfDPwjjysjyWssQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-35-180-23-35.eu-west-3.compute.amazonaws.com,35.180.23.35' (ECDSA) to the list of known hosts.
Last login: Sat Jul 10 07:25:37 2021

  _ |  _ | _ )
 _ | (  _ | /
 _ | \ _ | _ |
                Amazon Linux 2 AMI
```

```

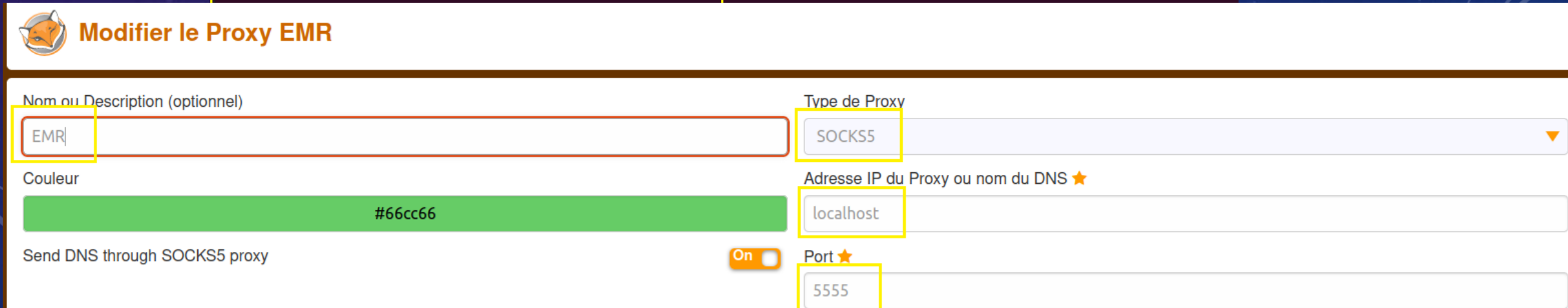
https://aws.amazon.com/amazon-linux-2/
49 package(s) needed for security, out of 89 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRRRRRRRRRR
E:::::: M::: M::: R:::
EE::: M::: M::: R:::
E::: E::: E::: M::: M::: M::: RR::: R:::
E::: E::: M::: M::: M::: M::: R::: R:::
E::: EEEEEEEEEEE M::: M M::: M M::: R::: RRRRRR:::
E::: E::: E M::: M M::: M M::: R::: R:::
E::: EEEEEEEEEEE M::: M M::: M M::: R::: RRRRRR:::
E::: E M::: M M::: M M::: R::: R:::
E::: E EEEEE M::: M M::: M M::: R::: R:::
EE::: EEEEEEEEEEE M::: M M::: M R::: R:::
E::: E::: E M::: M M::: M RR::: R:::
EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRR RRRRRR

[hadoop@ip-172-31-5-31 ~]$

```

```
[hadoop@ip-172-31-5-31 ~]$
```

[illegible]

Couleur

### Send DNS through SOCKS5 proxy

On

SOCKS5

localhost

Port ★

5555

# DÉMARRAGE DE SPARK

## 4.10.1 Démarrage de la session Spark

Entrée [1]: `# L'exécution de cette cellule démarre l'application Spark`

Starting Spark application

ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
0	application_1626050279029_0001	pyspark	idle	<a href="#">Link</a>	<a href="#">Link</a>	✓

FloatProgress(value=0.0, bar\_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...  
SparkSession available as 'spark'.

FloatProgress(value=0.0, bar\_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...

Affichage des informations sur la session en cours et liens vers Spark UI :

Entrée [2]: `%%info`

Current session configs: {'driverMemory': '1000M', 'executorCores': 2, 'proxyUser': 'jovyan', 'kind': 'pyspark'}

ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
0	application_1626050279029_0001	pyspark	idle	<a href="#">Link</a>	<a href="#">Link</a>	✓



# IMPORT DES LIBRAIRIES

## 4.10.3 Import des librairies

```
import pandas as pd
import numpy as np
import io
import os
import tensorflow as tf
from PIL import Image
from tensorflow.keras.applications.mobilenet_v2 import MobileNetV2, preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras import Model
from pyspark.sql.functions import col, pandas_udf, PandasUDFType, element_at, split
```

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

## 4.10.4 Définition des PATH pour charger les images et enregistrer les résultats

Nous accédons directement à nos **données sur S3** comme si elles étaient **stockées localement**.

```
PATH = 's3://p8-data'
PATH_Data = PATH+'/Test'
PATH_Result = PATH+'/Results'
print('PATH:      '+\
      PATH+'\nPATH_Data:  '+\
      PATH_Data+'\nPATH_Result: '+PATH_Result)
```

# CHARGEMENT DES DONNÉES

## 4.10.5.1 Chargement des données

```
images = spark.read.format("binaryFile") \  
  .option("pathGlobFilter", "*.jpg") \  
  .option("recursiveFileLookup", "true") \  
  .load(PATH_Data)
```

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

```
images.show(5)
```

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

path	modificationTime	length	content
s3://p8-data/Test...	2021-07-03 09:00:08	7353	[FF D8 FF E0 00 1...
s3://p8-data/Test...	2021-07-03 09:00:08	7350	[FF D8 FF E0 00 1...
s3://p8-data/Test...	2021-07-03 09:00:08	7349	[FF D8 FF E0 00 1...
s3://p8-data/Test...	2021-07-03 09:00:08	7348	[FF D8 FF E0 00 1...
s3://p8-data/Test...	2021-07-03 09:00:09	7328	[FF D8 FF E0 00 1...

only showing top 5 rows

# PRÉPARATION DU MODÈLE

## 4.10.5.2 Préparation du modèle

```
model = MobileNetV2(weights='imagenet',  
                    include_top=True,  
                    input_shape=(224, 224, 3))
```

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet\_v2/mobilenet\_v2\_weights\_tf\_dim\_ordering\_tf\_kernels\_1.0\_224.h5
```

```
14540800/14536120 [=====] - 0s 0us/step
```

```
new_model = Model(inputs=model.input,  
                  outputs=model.layers[-2].output)
```

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

```
broadcast_weights = sc.broadcast(new_model.get_weights())
```

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

```
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,
```

# FONCTION DE PRÉ-TRAITEMENT ET FEATURISATION

```
def preprocess(content):  
    """  
    Preprocesses raw image bytes for prediction.  
    """  
    img = Image.open(io.BytesIO(content)).resize([224, 224])  
    arr = img_to_array(img)  
    return preprocess_input(arr)  
  
def featurize_series(model, content_series):  
    """  
    Featurize a pd.Series of raw images using the input model.  
    :return: a pd.Series of image features  
    """  
    input = np.stack(content_series.map(preprocess))  
    preds = model.predict(input)  
    # For some layers, output features will be multi-dimensional tensors.  
    # We flatten the feature tensors to vectors for easier storage in Spark DataFrames.  
    output = [p.flatten() for p in preds]  
    return pd.Series(output)
```

```
@pandas_udf('array<float>', PandasUDFType.SCALAR_ITER)  
def featurize_udf(content_series_iter):  
    """  
    This method is a Scalar Iterator pandas UDF wrapping our featurization function.  
    The decorator specifies that this returns a Spark DataFrame column of type ArrayType(FloatType).  
  
    :param content_series_iter: This argument is an iterator over batches of data, where each batch  
                                is a pandas Series of image data.  
    """  
    # With Scalar Iterator pandas UDFs, we can load the model once and then re-use it  
    # for multiple data batches. This amortizes the overhead of loading big models.  
    model = model_fn()  
    for content_series in content_series_iter:  
        yield featurize_series(model, content_series)
```



# REDUCTION DE DIMENSION PAR ACP

```
Entrée [16]: from pyspark.ml.feature import PCA
              from pyspark.ml.linalg import VectorUDT, Vectors
              from pyspark.sql.functions import udf
```

```
Entrée [17]: # Fonction UDF pour la conversion
              def array_to_vector(arr):
                  return Vectors.dense(arr)

              # Enregistrement de la fonction UDF
              array_to_vector_udf = udf(array_to_vector, VectorUDT())

              # Avant la réduction de dimension PCA, convertissez la colonne "features"
              features = features_df.withColumn("features", array_to_vector_udf("features"))
```

```
Entrée [18]: # Création d'une instance PCA en spécifiant le nombre de composantes principales souhaité
              pca = PCA(k=20, inputCol="features", outputCol="pcaFeatures")

              # Application de la transformation PCA sur features_df
              pcaModel = pca.fit(features)
              pcaResult = pcaModel.transform(features)

              # Affichage du résultat
              pcaResult.select("label", "pcaFeatures").show()
```

# ÉXECUTION DES ACTIONS

## 4.10.5.4 Exécutions des actions d'extractions de features

```
# spark.conf.set("spark.sql.execution.arrow.maxRecordsPerBatch", "1024")

FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...

features_df = images.repartition(24).select(col("path"),
                                             col("label"),
                                             featurize_udf("content").alias("features")
                                             )

FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...

print(PATH_Result)
```

Entrée [17]: `features_df.write.mode("overwrite").parquet(PATH_Result)`

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

# CHARGEMENT DES DONNEES ENREGISTREES

## 4.10.6 Chargement des données enregistrées et validation du résultat

```
df = pd.read_parquet(PATH_Result, engine='pyarrow')
```

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

```
df.head()
```

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...
```

	path	...	features
0	s3://p8-data/Test/Watermelon/r_174_100.jpg	...	[0.0059991637, 0.44703647, 0.0, 0.0, 3.3713572...
1	s3://p8-data/Test/Pineapple Mini/128_100.jpg	...	[0.0146466885, 4.080593, 0.055877004, 0.0, 0.0...
2	s3://p8-data/Test/Pineapple Mini/137_100.jpg	...	[0.0, 4.9659867, 0.0, 0.0, 0.0, 0.0, 0.5144821...
3	s3://p8-data/Test/Watermelon/275_100.jpg	...	[0.22511952, 0.07235509, 0.0, 0.0, 1.690149, 0...
4	s3://p8-data/Test/Watermelon/271_100.jpg	...	[0.3286234, 0.18830013, 0.0, 0.0, 1.9123534, 0...

```
[5 rows x 3 columns]
```

# CONCLUSION

- On a chargé un grand volume de données sur S3
- Initialisation d'un serveur EMR (1 unité maitre et 2 unités principales)
- Création d'un tunnel SSH + FoxyProxy afin de lier notre serveur EMR à JupyterHub
- Lancement du Notebook :
  - Session Spark
  - Chargement des données et préparation du modèle
  - Pré-traitement et Featurisation
  - ACP
  - Exécution des actions
- On récupère dans notre Bucket S3 les 22 688 images :
  - une featurisation par MobileNet (Récupération de 1200 features)
  - Un Acp (Synthétisant les 1200 features à 20)