DÉPLOIEMENT D'UN MODÈLE DANS LE CLOUD

Fruits!

HTTPS://GITHUB.COM/PIMOUSS75/PROJET_8

MARC SELLAM

PROJET 8

DATA SCIENTIST

AVRIL 2023

SOMMAIRE

Je suis Data Scientist dans une start-up nommée Fruits! .

Elle souhaite une application mobile qui permettrait aux utilisateurs de prendre en photo un fruit et d'en obtenir les informations de mettre en place une première version d'un moteur de classification des images de fruits.

MISSION 1

- Un notebook testé dans un environnement Big Data AWS EMR.

- Un dossier compressé d'images de fruits au format jpg dont le lien en téléchargement direct a été fourni, contenant :
  - 131 dossiers de Fruits différents
  - Un total de 22688 images au format 100 x 100

LES INFORMATIONS FOURNIES

- En raison de l'augmentation du volume de données après la livraison de ce projet, on souhaite mettre en place une solution Big Data.

- Démonstration d'une instance dans le Cloud et expliquer script PySpark, qui aura été complété :
  - d'un traitement de diffusion des poids.
  - d'une étape de réduction de dimension de type PCA en PySpark.
  - d'une sortie d'une matrice écrite sur un fichier au format csv.

- Respect des contraintes du RGPD .

- Un retour critique de cette solution.

LES CONTRAINTES 3

- Complétion du script PySpark et test en local avec un petit volume de données (3 Fruits).

Puis :

- Mise en place de l'environnement Big Data dans le cloud.
- Déploiement du traitement des données dans l'environnement Big Data.
- Test du script avec un volume de données plus importants (7 Fruits) dans le cloud en mode de calcul distribué.
- Surveiller les couts d'utilisation du serveur dans le cloud.

MISE EN ŒUVRE

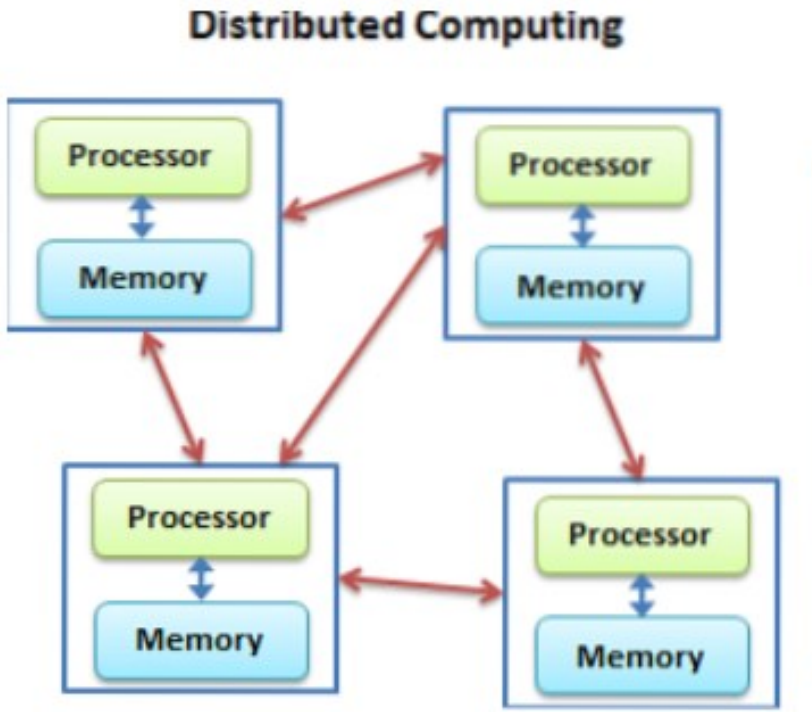
Qu’est-ce que le Big Data ?

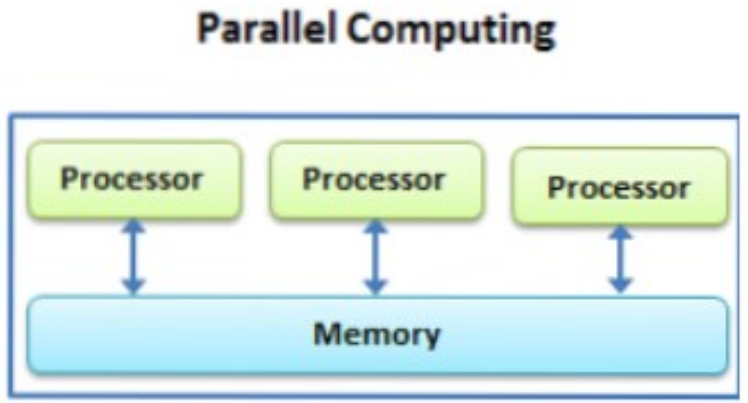
Les trois « V » du Big Data

Volume	La quantité de données a son importance. Avec le Big Data, vous devrez traiter de gros volumes de données non structurées et à faible densité. Il peut s’agir de données de valeur inconnue, comme des flux de données Twitter, des flux de clics sur une page Internet ou une application mobile ou d’un appareil équipé d’un capteur. Pour certaines entreprises, cela peut correspondre à des dizaines de téraoctets de données. Pour d’autres, il peut s’agir de centaines de pétaoctets.
Vitesse	La vitesse à laquelle les données sont reçues et éventuellement traitées. Normalement, les données haute vitesse sont transmises directement à la mémoire, plutôt que d’être écrites sur le disque. Certains produits intelligents accessibles via Internet opèrent en temps réel ou quasi réel et nécessitent une évaluation et une action en temps réel.
Variété	La variété fait allusion aux nombreux types de données disponibles. Les types de données traditionnels ont été structurés et trouvent naturellement leur place dans une base de données relationnelle. Avec l’augmentation du Big Data, les données ne sont pas nécessairement structurées. Les types de données non structurés et semi-structurés, tels que le texte, l’audio et la vidéo, nécessitent un prétraitement supplémentaire pour en déduire le sens et prendre en charge les métadonnées.

Le Calcul distribué

Le modèle distribué permet une plus grande tolérance aux pannes : lorsqu'un nœud du cluster subit une panne, il suffit d'affecter la tâche qu'il était en train de traiter à un autre nœud, alors que dans le modèle parallèle la machine sur laquelle le calcul est exécuté constitue un point unique de défaillance.

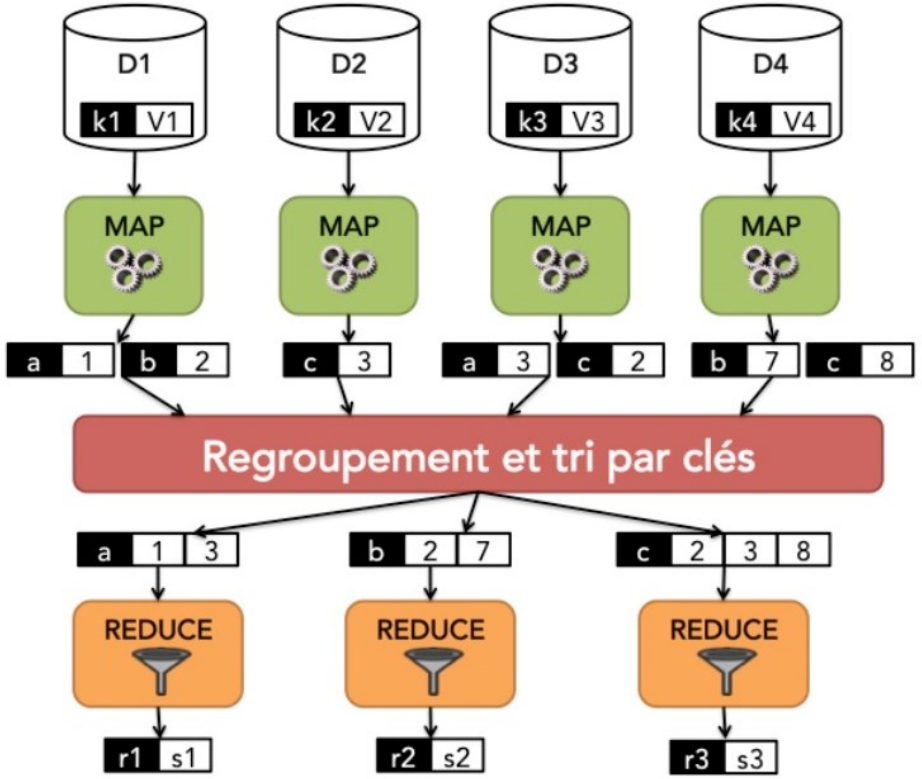




MapReduce

SPLIT → MAP → SHUFFLE and SORT → REDUCE

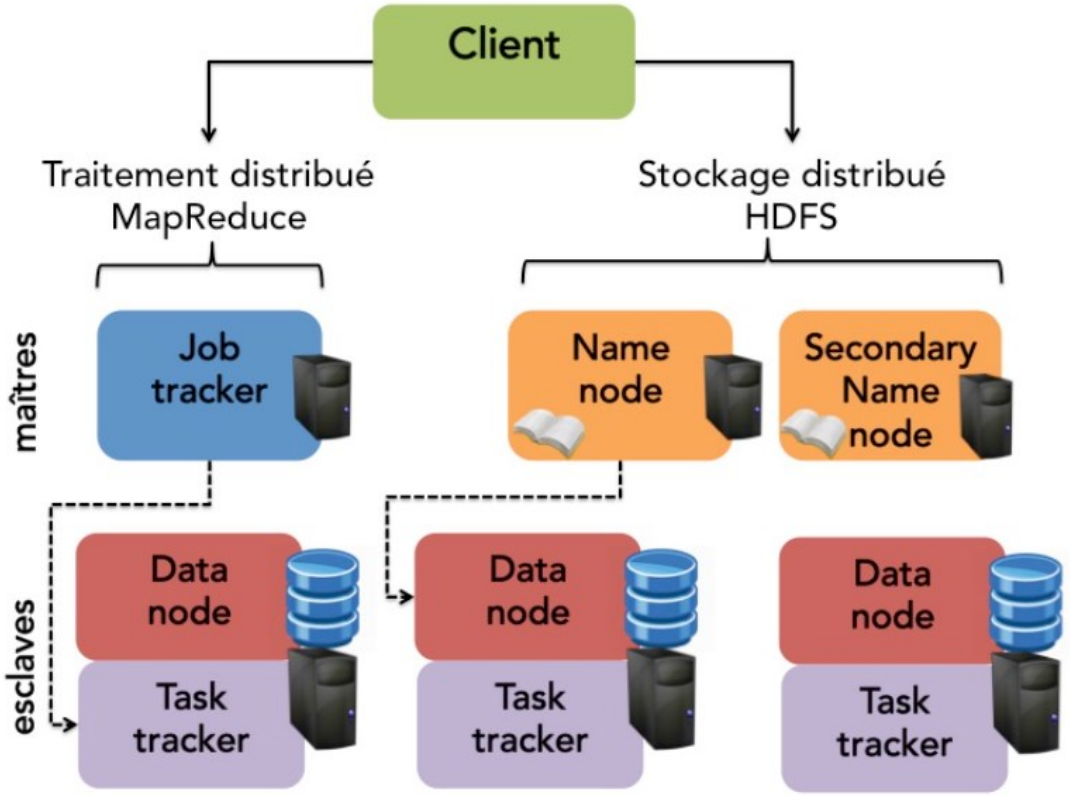
DECOUPER TRANSFORMER GROUPER REDUIRE

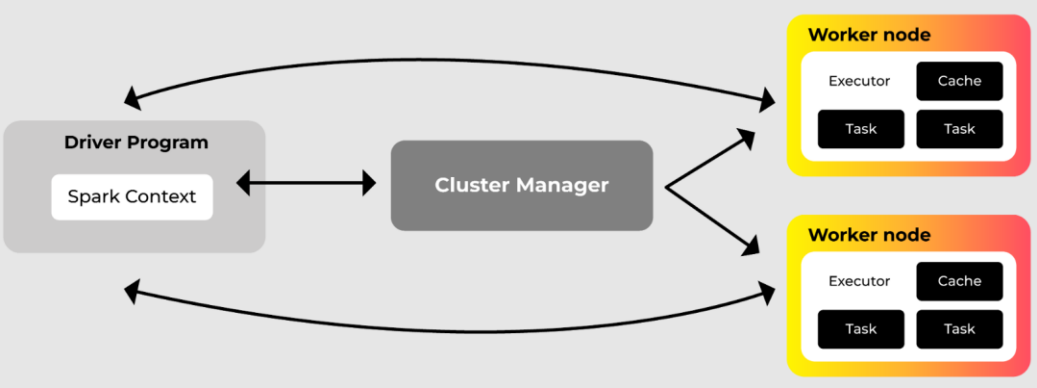


Socle technique hadoop

- Toute l'architecture support nécessaire pour l'orchestration de MapReduce

- Un système de fichiers HDFS





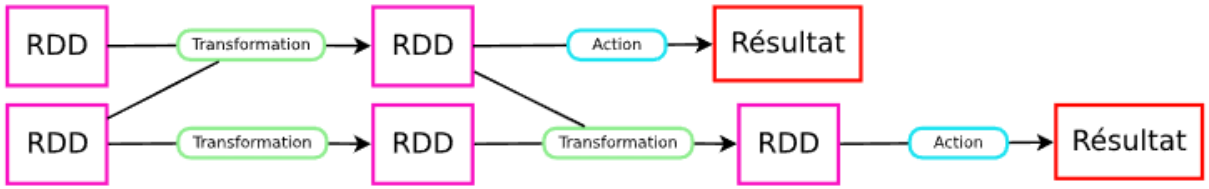
Apache Spark est une alternative à Hadoop MapReduce pour le **calcul distribué** qui vise à **résoudre** ces deux défauts:
-Des écritures et lectures sont coûteuses en temps.
-Le jeu d'expressions composé exclusivement d'opérations map et reduce est très limité et peu expressif.

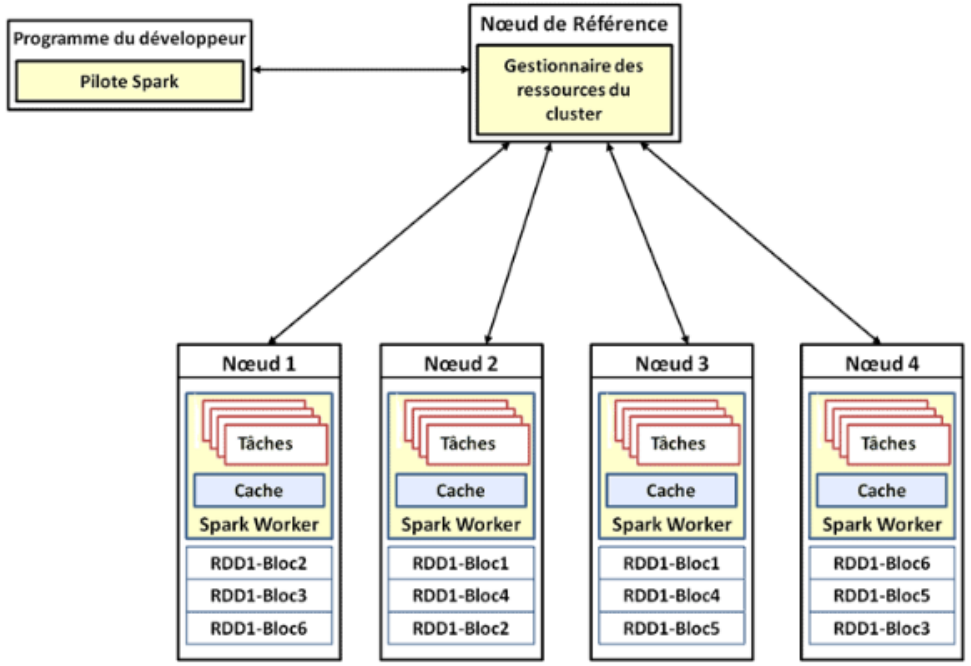
3 composants principaux :
•Un **driver**
•Un **cluster manager**
•Un ou plusieurs **workers**

La différence fondamentale entre Hadoop MapReduce et Spark est que Spark écrit les données en RAM, et non sur disque ainsi que sur l'architecture globale de Spark.

CONCEPTS CLÉS

Calculs distribués sous forme de graphe avec les DAG





- Un **job** Spark correspond à une action sur un RDD et est composé de plusieurs **étapes** séparées par des **shuffles**.
- Chaque **étape** est composée de **tâches**.
- Chaque tâche s'exécute sur une **partition** différente des données.
- Les partitions sont réparties sur les différents **executors**.
- Les partitions sont créées par les **Resilient Distributed Datasets** (RDD).

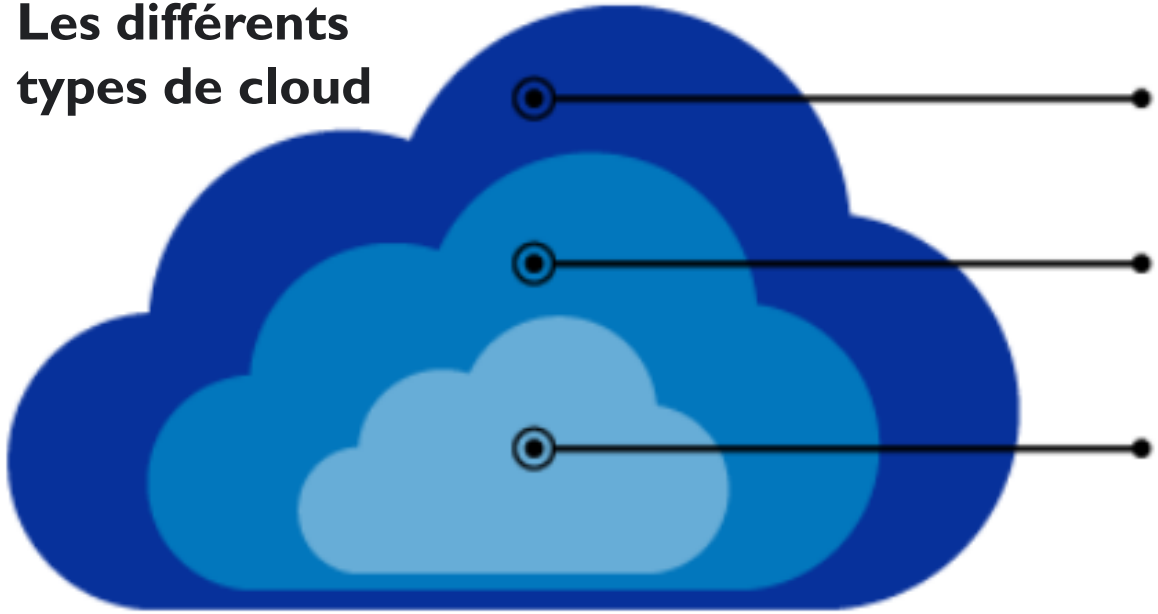
PySpark

PySpark fournit une API Python pour utiliser les fonctionnalités de Spark, ce qui facilite l'écriture de programmes Spark en Python.

Le Cloud

Le cloud est l'accès et le stockage grâce à Internet. Il s'oppose ainsi à l'utilisation d'un stockage physique ou local par l'intermédiaire d'un disque dur d'ordinateur.

Les différents types de cloud



SaaS
Utiliser

(Software as a Service) : on vous fournit l'accès à un logiciel sous forme de service sous la forme d'une application web (ex. : Microsoft Office)

PaaS
Déployer

(Platform as a Service) : non seulement on vous fournit un accès à l'infrastructure, mais on s'occupe aussi de gérer le nombre de machines nécessaires pour que votre application fonctionne bien.

IaaS
Héberger

(Infrastructure as a Service) : un prestataire vous fournit un accès à tout ou partie de son infrastructure technique, c'est-à-dire à ses serveurs.

Diffusion des poids du modèle de transfert learning

On charge les poids du model pour etre exécuté sur tous les nœuds

In [16]:
```
brodcast_weights = sc.broadcast(new_model.get_weights())
```

Mettons cela sous forme de fonction :

In [17]:
```
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
```

Ajout d'une étape de réduction de dimension

```
n_components = 100
pca = PCA(
    k = n_components,
    inputCol = 'scaledFeatures',
    outputCol = 'pcaFeatures'
).fit(df_scaled)

df_pca = pca.transform(df_scaled)
df_pca.show(6)
```

```
+--------------------+-----------+--------------------+--------------------+
|                path|      label|      scaledFeatures|         pcaFeatures|
+--------------------+-----------+--------------------+--------------------+
|file:/C:/Users/mu...|Apple Red 1|[-1.1977509841651...|[-6.2871393865860...|
|file:/C:/Users/mu...|Apple Red 1|[-1.1977509841651...|[-5.9051367560626...|
|file:/C:/Users/mu...|Apple Red 1|[-0.9021714007865...|[-6.7050055947995...|
|file:/C:/Users/mu...|Apple Red 1|[-0.7579063437929...|[-7.3259522917575...|
|file:/C:/Users/mu...|Apple Red 1|[-0.8929905795362...|[-6.6761411351968...|
|file:/C:/Users/mu...|Apple Red 1|[0.60797174798140...|[-10.620241979613...|
+--------------------+-----------+--------------------+--------------------+
only showing top 6 rows
```

```
print('Explained Variance Ratio', pca.explainedVariance.sum())
```

```
Explained Variance Ratio 0.9240176084006985
```

Sortie de la réduction de dimension (fichier au format csv)

```
+-----------+-----+
|      label|count|
+-----------+-----+
|      Dates|  166|
|       Kaki|  166|
|Apple Red 1|  164|
+-----------+-----+
```

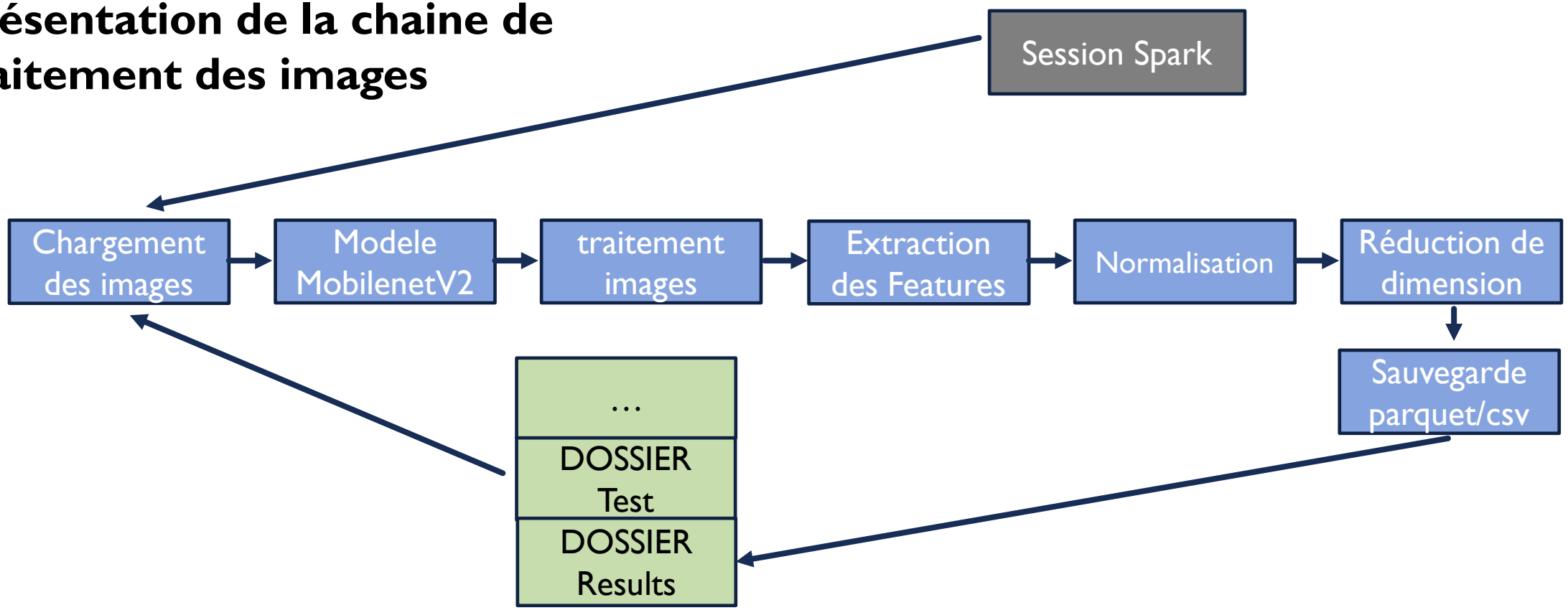
```
df = df_pca.select("*").toPandas()
```

```
df.head()
```

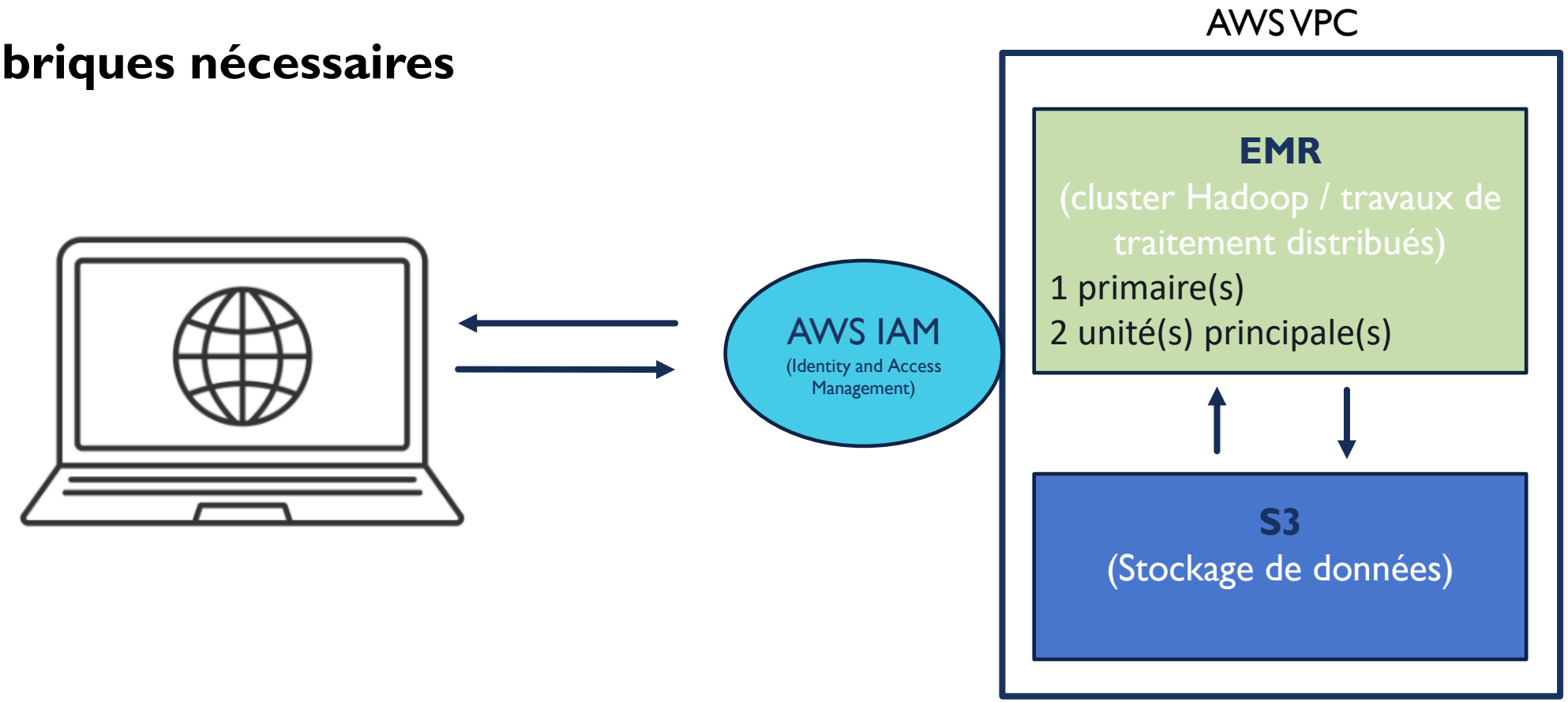
	path	label	pcaFeatures
0	file:/C:/Users/multi/Desktop/Projets_DS/P8_ds/...	Apple Red 1	[-6.287139386586003, -17.056464787267654, -2.4...
1	file:/C:/Users/multi/Desktop/Projets_DS/P8_ds/...	Apple Red 1	[-5.905136756062631, -17.296257849603094, -2.5...
2	file:/C:/Users/multi/Desktop/Projets_DS/P8_ds/...	Apple Red 1	[-6.705005594799506, -12.167035846753548, -3.6...
3	file:/C:/Users/multi/Desktop/Projets_DS/P8_ds/...	Apple Red 1	[-7.3259522917575355, -16.293423930727133, -5....
4	file:/C:/Users/multi/Desktop/Projets_DS/P8_ds/...	Apple Red 1	[-6.676141135196839, -16.836243784648985, -6.7...

```
df.to_csv("{}/results.csv".format(PATH_Result), index=False)
```

Présentation de la chaine de traitement des images



Les briques nécessaires



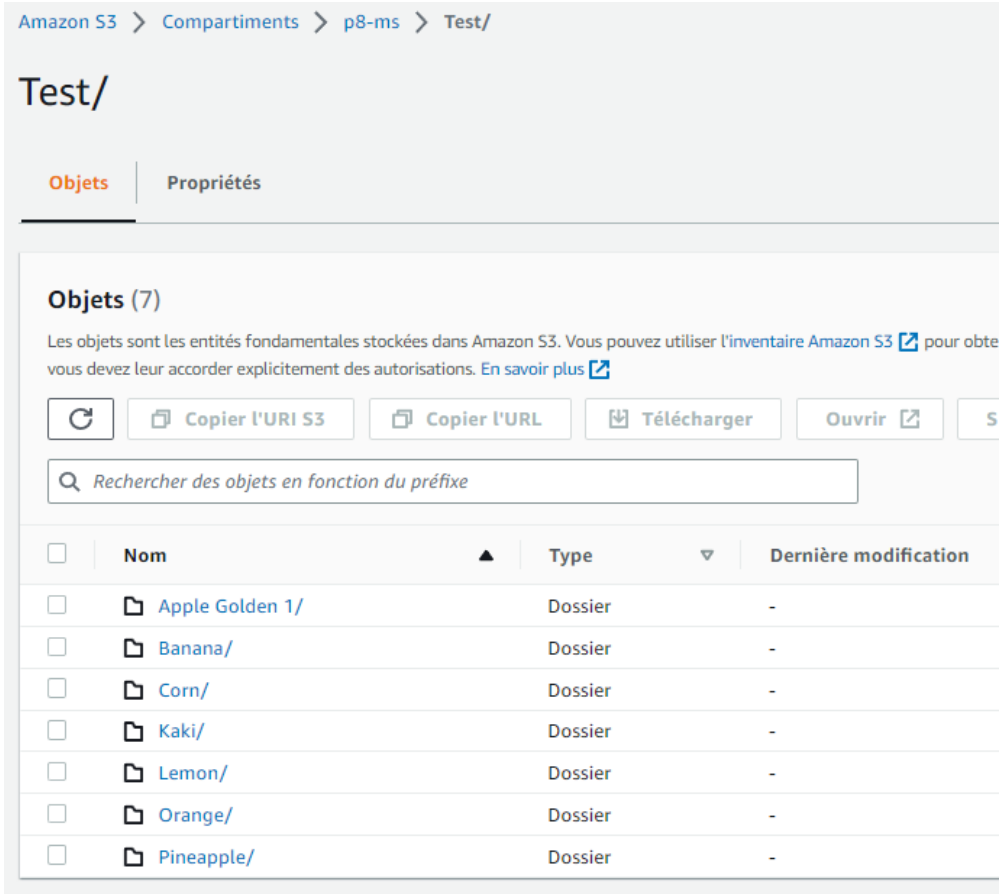
Configuration AWS

- Création d'une clé d'accès AWS (AWS IAM)
- Installation et configuration de AWS CLI

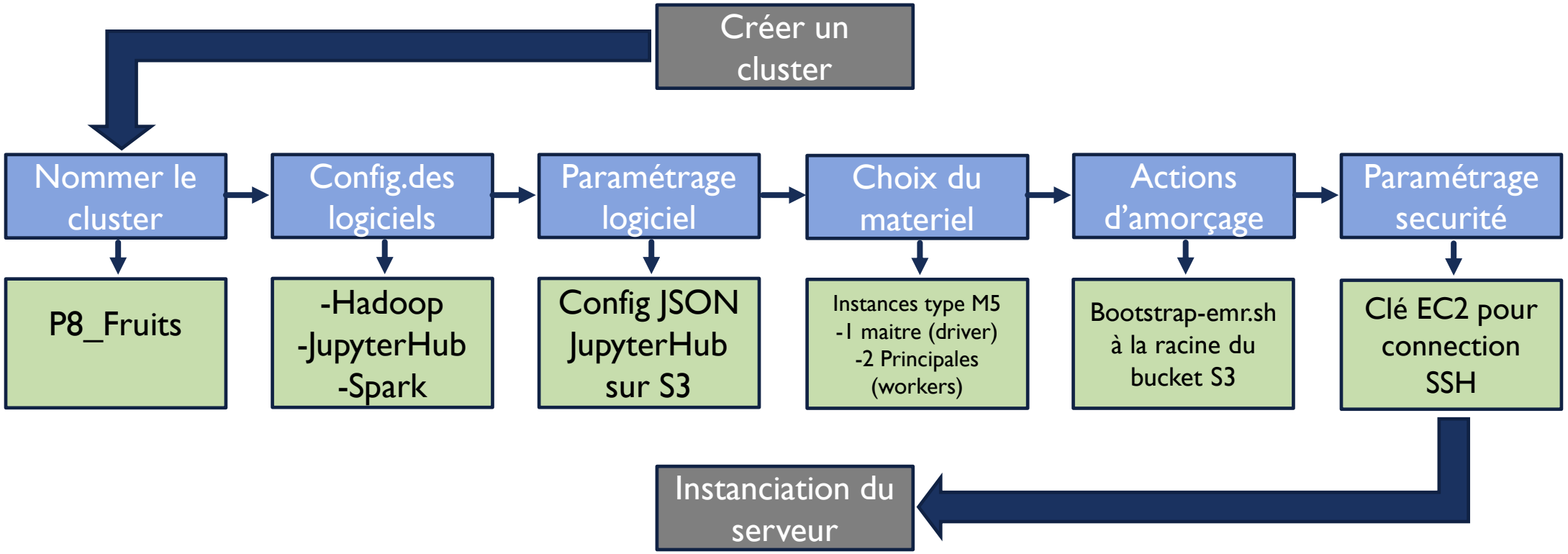
```
C:\Users\multi\Desktop>aws configure
AWS Access Key ID [None]: XXXXXXXXXXPJMM6A67NM
AWS Secret Access Key [None]: XXXXXXXXXXcZbApzy5mCDUGgm2OnLz9FKEvf/sSD
Default region name [None]: eu-west-3
Default output format [None]: json

aws swf list-domains --registration-status REGISTERED --output json
```

Création d'un bucket S3 et upload des images dans le dossier Test

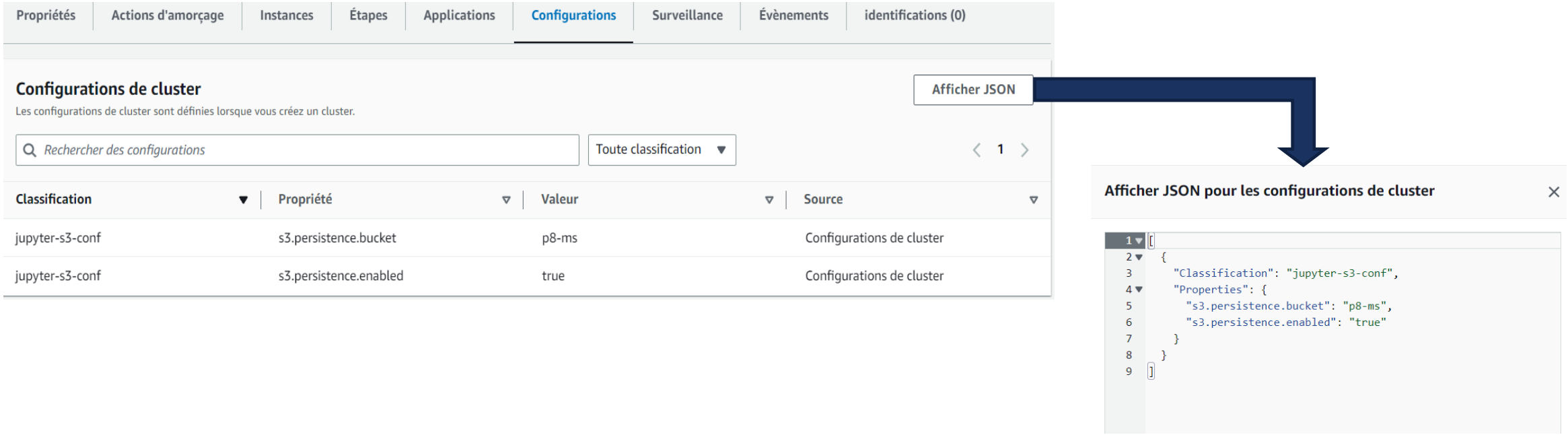
```
C:\Users\multi\Desktop>aws s3 mb s3://p8-data

C:\Users\multi\Desktop>aws s3 ls
2023-03-28 14:57:27 p8-ms

aws s3 sync Test/ s3://p8-ms/Test
```



Configuration de l'EMR

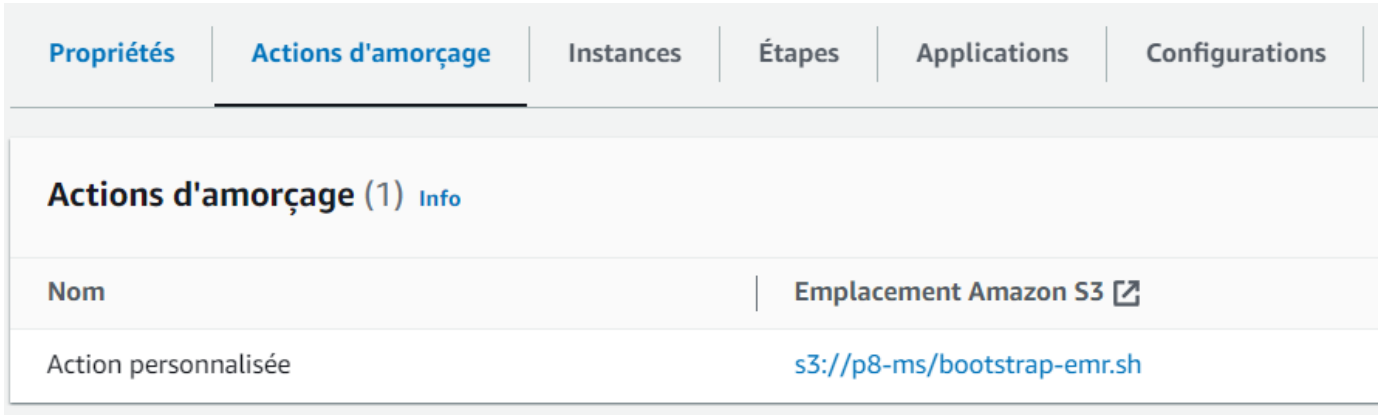


Paramétrage logiciel



Actions d'amorçage

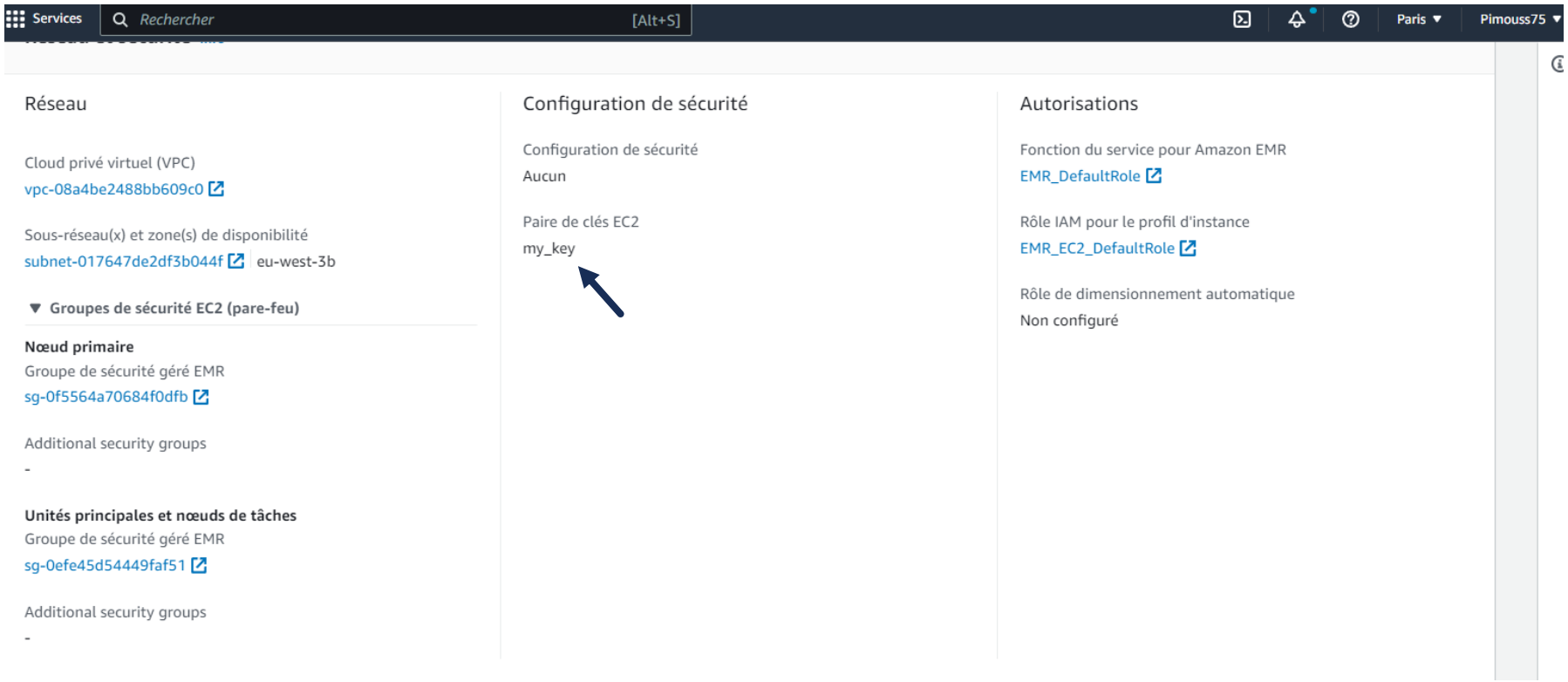
```
C:\Users\multi\Desktop>aws s3 cp bootstrap-emr.sh s3://p8-ms/
upload: .\bootstrap-emr.sh to s3://p8-ms/bootstrap-emr.sh
```



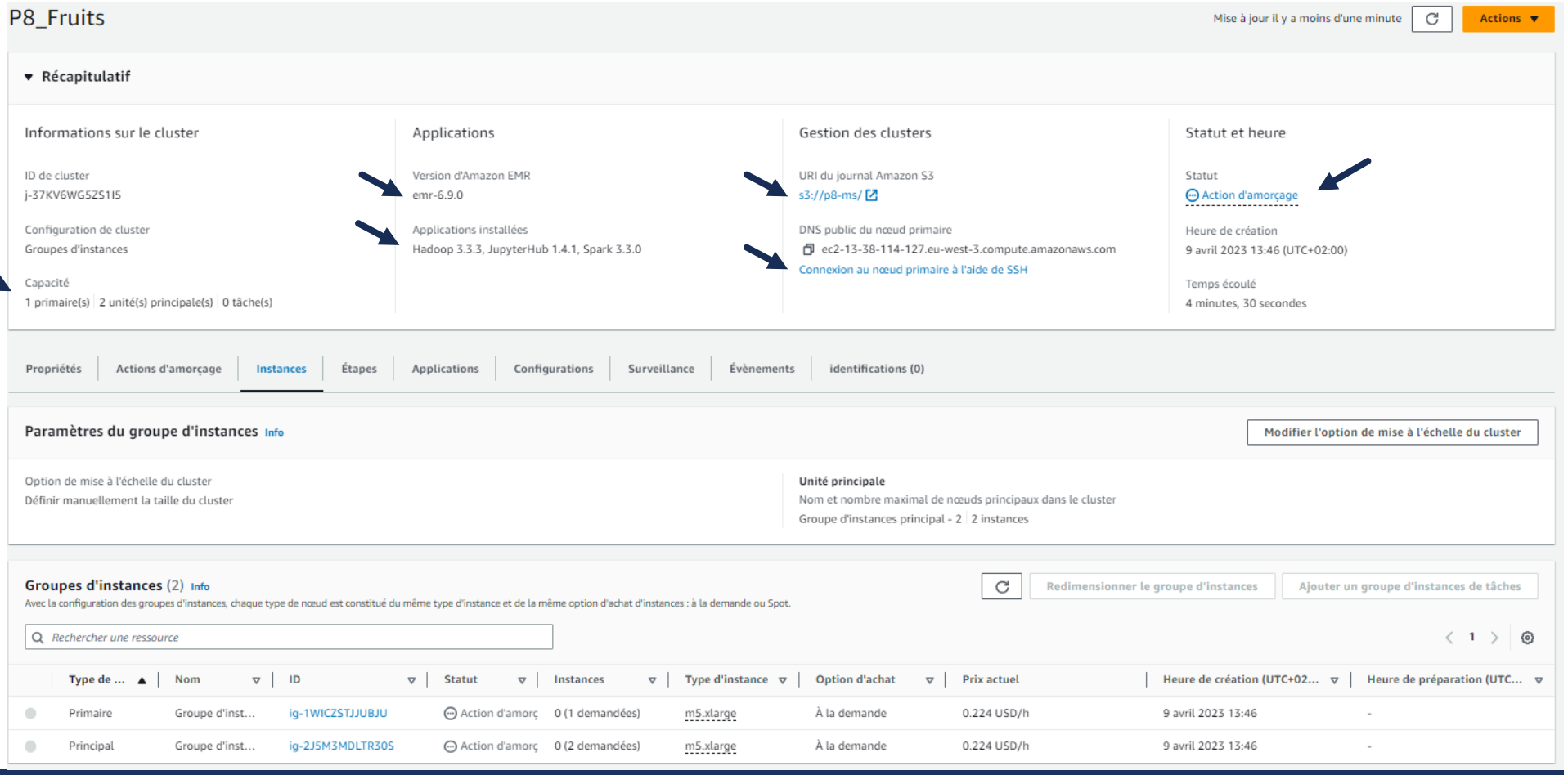
Contenu du fichier bootstrap-emr.sh

```
#! /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
sudo python3 -m pip install tensorflow
```

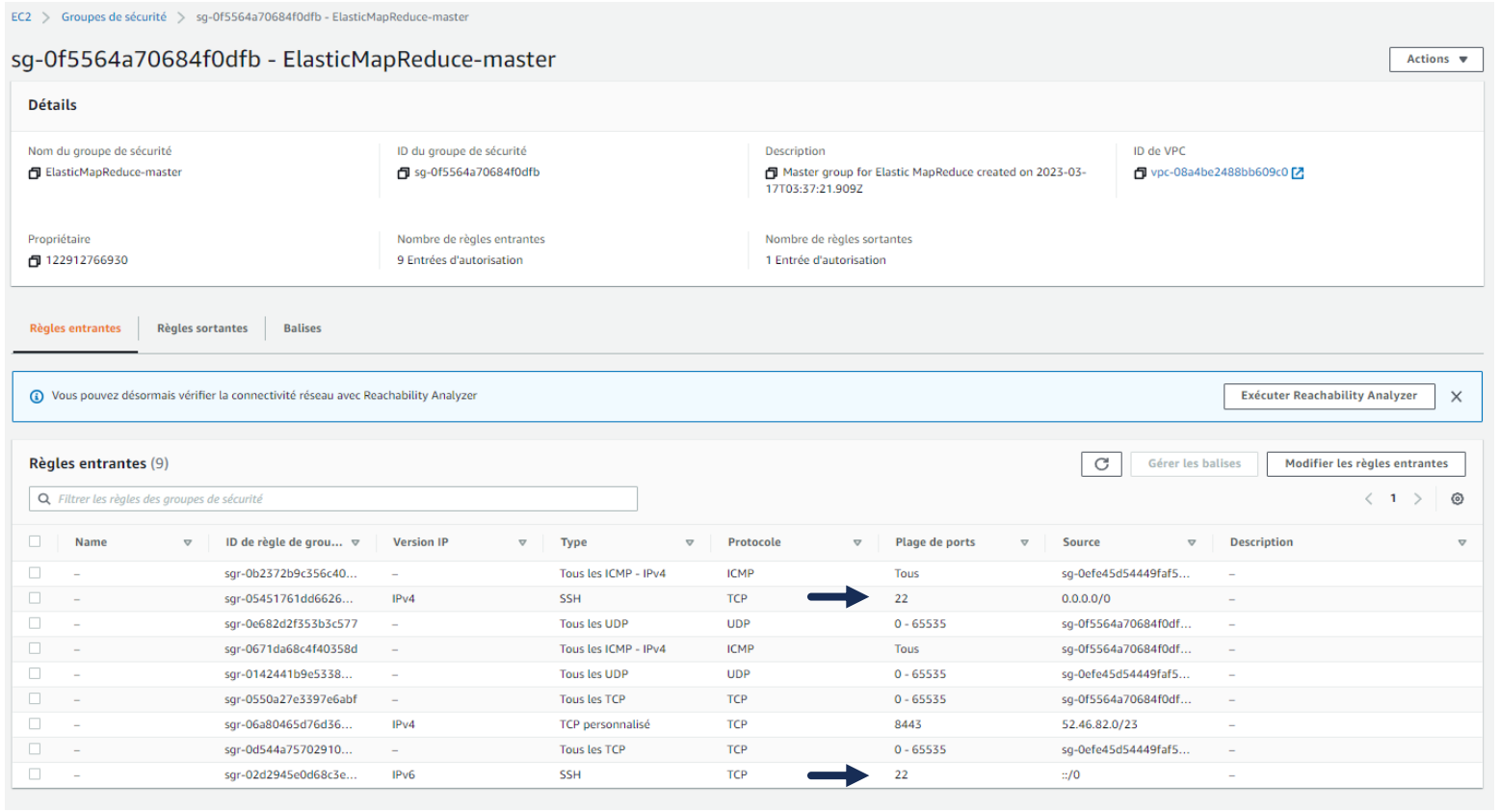
Paramétrage de sécurité



Instanciation du serveur



Création des autorisations sur les connexions entrantes



Création du tunnel SSH vers le driver et configuration de foxyproxy

```
C:\Users\multi\Desktop>ssh -i my_key.pem -D 8157 hadoop@ec2-52-47-35-61.eu-west-3.compute.amazonaws.com
The authenticity of host 'ec2-52-47-35-61.eu-west-3.compute.amazonaws.com (52.47.35.61)' can't be established.
ED25519 key fingerprint is SHA256:Rq7cLhUrQsdjBowXecRHqWMUt/GIMk5ActJ/54IRXqY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-47-35-61.eu-west-3.compute.amazonaws.com' (ED25519) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
22 package(s) needed for security, out of 25 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE::::::EEEEEEEEE:::E M::::::::M       M::::::::M R::::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR:::::R     R:::::R
  E::::E             M::::::M:::M   M:::M::::::M   R::::R     R:::::R
  E::::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R::::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R::::::::::::RR
  E::::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R::::RRRRRR:::::R
  E::::E             M:::::M    M:::M    M:::::M   R::::R     R:::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R::::R     R:::::R
EE::::::EEEEEEEE:::E M:::::M             M:::::M   R::::R     R:::::R
E::::::::::::::::::E M:::::M             M:::::M RR:::::R     R:::::R
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR     RRRRRRR

[hadoop@ip-172-31-16-206 ~]$
```

Proxy mode: Use proxy localhost:8157 for all URLs

Proxies

Enabled	Color	Proxy Name	Proxy Notes	Host or IP Address	Port	SOCKS proxy?	SOCKS Version
✔		localhost:8157		localhost	8157	✔	5
✔		Default	These are the settings that are used when no patterns match an URL				5

Import your proxies from FoxyProxy on Mozilla Firefox or from another computer.

Please Donate | Buy Proxy Service

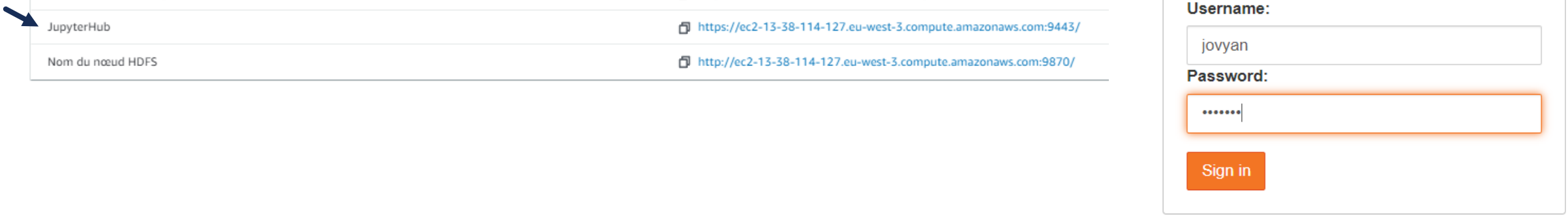
Connexion au notebook JupyterHub

jupyterhub

Interfaces utilisateur d'application sur le nœud primaire

Cela nécessite 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://ec2-13-38-114-127.eu-west-3.compute.amazonaws.com:8088/
Interface utilisateur Spark	http://ec2-13-38-114-127.eu-west-3.compute.amazonaws.com:18080/
JupyterHub	https://ec2-13-38-114-127.eu-west-3.compute.amazonaws.com:9443/
Nom du nœud HDFS	http://ec2-13-38-114-127.eu-west-3.compute.amazonaws.com:9870/



Session Spark

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

```
Starting Spark application
```

ID	YARN Application ID	Kind	State	Spark UI	Driver log	User	Current session?
0	application_1680809092392_0001	pyspark	idle	Link	Link	None	✓

```
SparkSession available as 'spark'.
```

Définition des PATH :

```
PATH:         s3://p8-ms
PATH_Data:    s3://p8-ms/Test
PATH_Result:  s3://p8-ms/Results
```

Chargement des images

Les images sont chargées au format binaire

```
+--------------------+-------------------+------+--------------------+
|                path|   modificationTime|length|             content|
+--------------------+-------------------+------+--------------------+
|s3://p8-ms/Test/P...|2023-04-04 09:03:12|  6574|[FF D8 FF E0 00 1...|
|s3://p8-ms/Test/P...|2023-04-04 09:03:09|  6573|[FF D8 FF E0 00 1...|
|s3://p8-ms/Test/P...|2023-04-04 09:03:11|  6573|[FF D8 FF E0 00 1...|
|s3://p8-ms/Test/P...|2023-04-04 09:02:57|  6572|[FF D8 FF E0 00 1...|
|s3://p8-ms/Test/P...|2023-04-04 09:03:06|  6572|[FF D8 FF E0 00 1...|
+--------------------+-------------------+------+--------------------+
```

Chargement du modèle MobilenetV2 en retirant l'avant dernière couche

```
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
14536120/14536120 [==============================] - 1s 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%'),…
```

en entrée : l'entrée du modèle MobileNetV2
en sortie : l'avant dernière couche du modèle MobileNetV2

Traitement des images et extraction des Features

```
+--------------+-----+
|         label|count|
+--------------+-----+
|        Orange|  160|
|     Pineapple|  166|
|Apple Golden 1|  160|
|          Kaki|  166|
|        Banana|  166|
|         Lemon|  164|
|          Corn|  150|
+--------------+-----+
```

Preprocessing → Redimension de l'image de 100 x 100 à 224 x 224

Broadcast_weights → On charge les poids du model pour etre exécuté sur tous les nœuds. (brodcast_weights = sc.broadcast(new_model.get_weights()))

Featurize_udf → Application de la featurisation des images à travers l'utilisation de pandas UDF.

```
+--------------------+--------------+--------------------+
|                path|         label|            features|
+--------------------+--------------+--------------------+
|s3://p8-ms/Test/P...|     Pineapple|[0.0, 3.632566, 0...|
|s3://p8-ms/Test/P...|     Pineapple|[0.0, 4.1071906, ...|
|s3://p8-ms/Test/P...|     Pineapple|[0.0, 3.6676247, ...|
|s3://p8-ms/Test/A...|Apple Golden 1|[0.0, 0.053870715...|
|s3://p8-ms/Test/P...|     Pineapple|[0.0039413804, 2....|
+--------------------+--------------+--------------------+
```

Vectorisation des features

```
# Apply the 'array_to_vector' function to the 'features_vectorized' column
features_df = features_df.withColumn("features_vectorized", array_to_vector("features"))
```

Normalisation des features

```
scaler = StandardScaler(
    inputCol = 'features_vectorized',
    outputCol = 'scaledFeatures',
    withMean = True,
    withStd = True
).fit(features_df)

# when we transform the dataframe, the old
# feature will still remain in it
df_scaled = scaler.transform(features_df)
df_scaled.show(6)
```

Application de l'ACP

```
n_components = 100
pca = PCA(
    k = n_components,
    inputCol = 'scaledFeatures',
    outputCol = 'pcaFeatures'
).fit(df_scaled)

df_pca = pca.transform(df_scaled)
df_pca.show(6)
```

```
+--------------------+-------------+--------------------+
|                path|        label|         pcaFeatures|
+--------------------+-------------+--------------------+
|s3://p8-ms/Test/P...|    Pineapple|[4.24952868426703...|
|s3://p8-ms/Test/P...|    Pineapple|[4.78663711046702...|
|s3://p8-ms/Test/P...|    Pineapple|[3.92177294440557...|
|s3://p8-ms/Test/A...|Apple Golden 1|[7.16397370645045...|
|s3://p8-ms/Test/P...|    Pineapple|[6.37464657896214...|
+--------------------+-------------+--------------------+
```

```
print('Variance expliquee:', pca.explainedVariance.sum())
```

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

Variance expliquee: 0.906408857244218
```

Enregistrement des Resultats

```
# au format parquet
df_pca.write.mode("overwrite").parquet(PATH_Result)
```

```
df = df_pca.select("*").toPandas()
```

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

```
df.head()
```

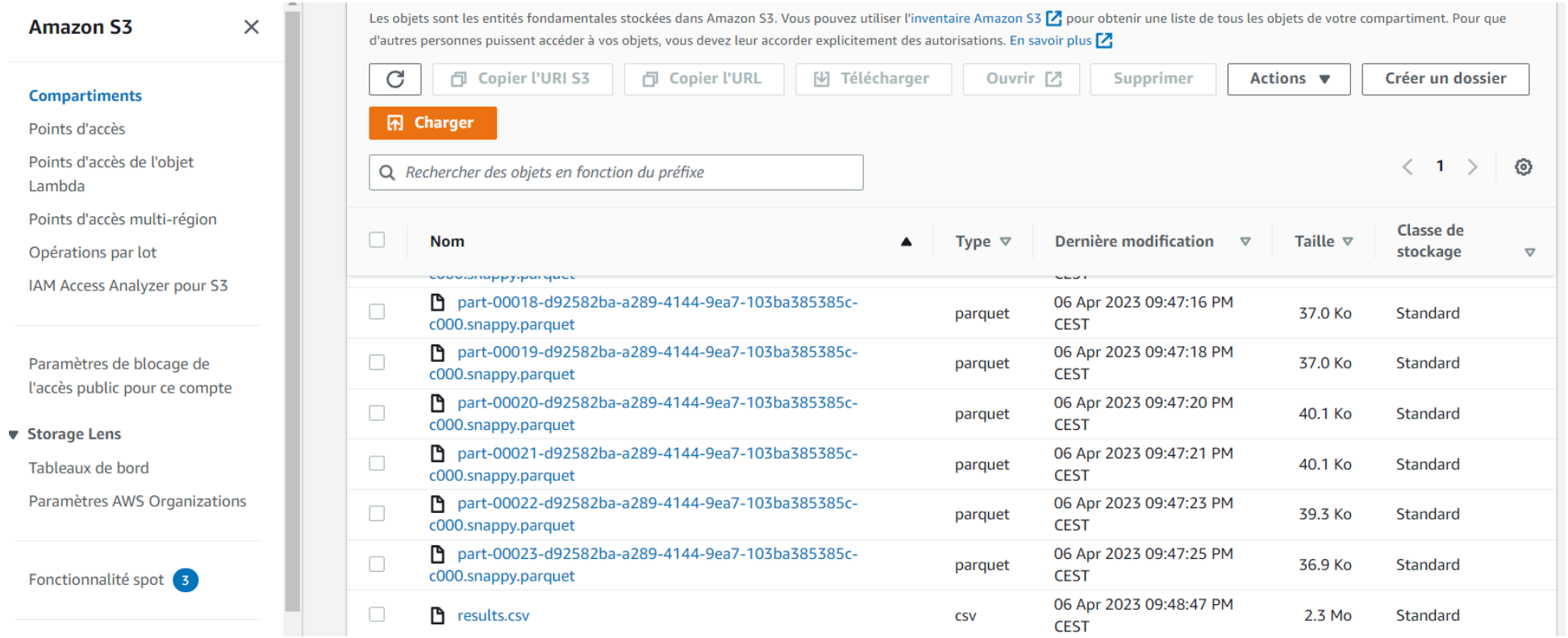
```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),…
```

	path	...	pcaFeatures
0	s3://p8-ms/Test/Pineapple/209_100.jpg	...	[4.249528684267031, 24.29155204063384, -16.101...
1	s3://p8-ms/Test/Pineapple/24_100.jpg	...	[4.786637110467027, 23.88517594245971, -15.595...
2	s3://p8-ms/Test/Pineapple/197_100.jpg	...	[3.9217729444055713, 23.539003012275177, -16.1...
3	s3://p8-ms/Test/Apple Golden 1/130_100.jpg	...	[7.163973706450456, 4.234909962497389, 10.7138...
4	s3://p8-ms/Test/Pineapple/r_81_100.jpg	...	[6.374646578962141, 18.568245346159973, -11.51...

[5 rows x 3 columns]

```
# au format csv
df.to_csv("{}/results.csv".format(PATH_Result), index=False)
```

Résultats sur le bucket S3



Historique Spark sur le serveur

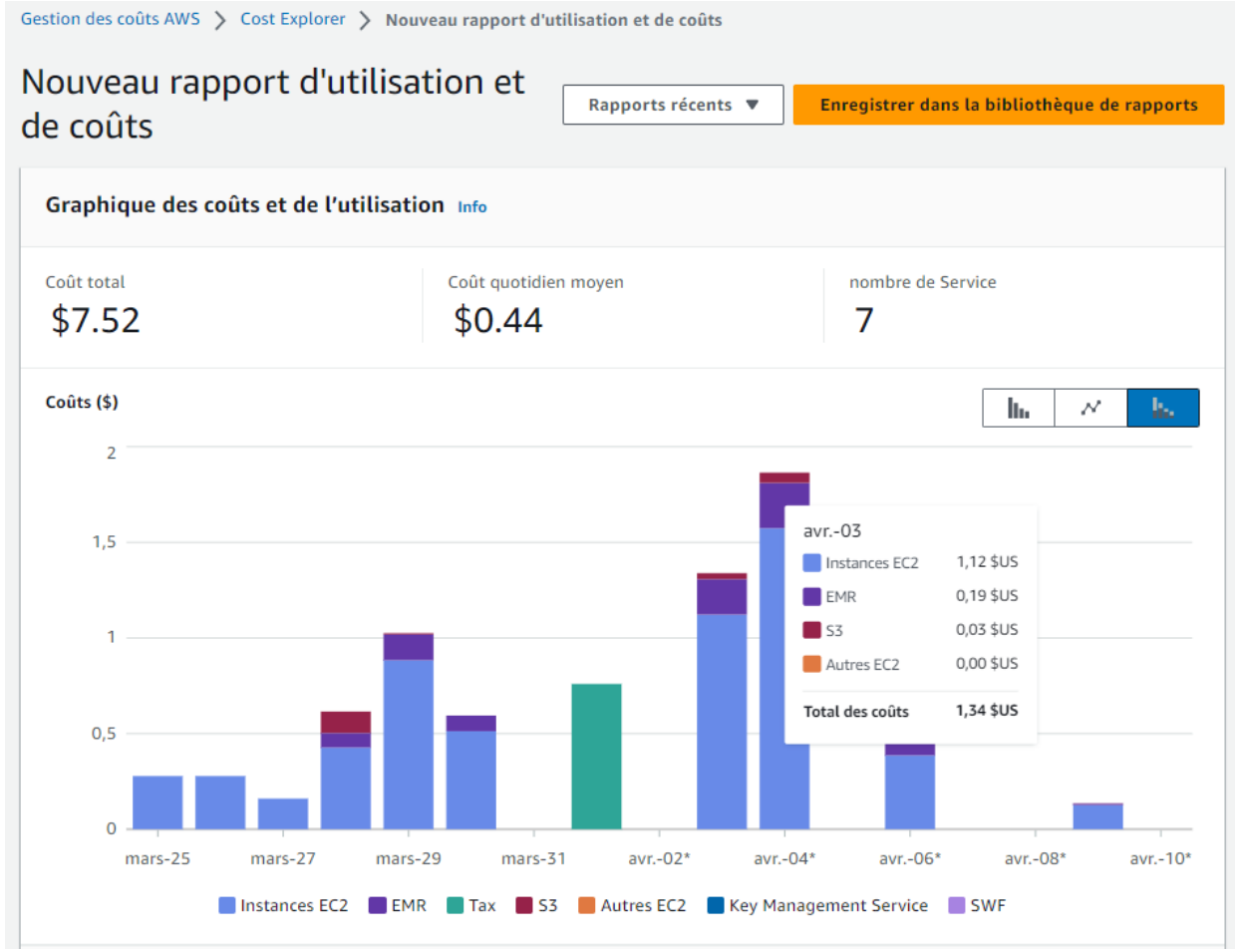
▾ Completed Jobs (38)

Page: 1

1 Pages. Jump to 1 . Show 100 items in a page. Go

Job Id (Job Group) ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
37 (40)	Job group for statement 40 toPandas at <stdin>:1	2023/04/06 19:47:56	45 s	1/1 (1 skipped)	24/24 (36 skipped)
36 (40)	Job group for statement 40 toPandas at <stdin>:1	2023/04/06 19:47:28	28 s	1/1	36/36
35 (38)	Job group for statement 38 parquet at NativeMethodAccessorImpl.java:0	2023/04/06 19:46:35	49 s	1/1 (1 skipped)	24/24 (36 skipped)
34 (38)	Job group for statement 38 parquet at NativeMethodAccessorImpl.java:0	2023/04/06 19:46:06	29 s	1/1	36/36
33 (35)	Job group for statement 35 showString at NativeMethodAccessorImpl.java:0	2023/04/06 19:45:47	3 s	1/1 (1 skipped)	1/1 (36 skipped)
32 (35)	Job group for statement 35 showString at NativeMethodAccessorImpl.java:0	2023/04/06 19:45:18	30 s	1/1	36/36
31 (31)	Job group for statement 31 collect at <stdin>:1	2023/04/06 19:44:12	43 s	1/1 (1 skipped)	24/24 (36 skipped)
30 (30)	Job group for statement 30 collect at <stdin>:1	2023/04/06 19:43:22	46 s	1/1 (1 skipped)	24/24 (36 skipped)
29 (30)	Job group for statement 30 javaToPython at NativeMethodAccessorImpl.java:0	2023/04/06 19:42:58	24 s	1/1	36/36

Evolution du cout



CONCLUSION

En résumé, nous avons déployé sur le cloud une solution qui permet avec notre script PySpark d'extraire les caractéristiques de chaque image avec un algorithme de transfert learning, d'effectuer une réduction de dimension (ACP) sur ces caractéristiques et leur sauvegarde pour une utilisation ultérieure.