



Réalisez un traitement dans un environnement Big Data sur le Cloud

-
Synthèse

Sommaire

- 1 – Rappel de la problématique
 - 1.1 – Use case
 - 1.2 – Données
 - 1.3 – Big Data : les 4 Vs
- 2 – Choix technologiques
 - 2.1 – Cloud provider
 - 2.2 – Architecture & outils
 - 2.3 – Considérations RGPD
- 3 – Proof of Concept
 - 3.1 – Test en local
 - 3.1.2 – Environnement & contraintes
 - 3.2.2 – Résultats
 - 3.2 – Déploiement dans AWS Cloud
 - 3.2.1 – Paramétrage
 - 3.2.2 – Monitoring
 - 3.2.3 – Résultats



1 – Rappel de la problématique

1.1 – Use case

- ❖ Moteur de classification d'images de fruits
 - Machine learning – classification supervisée
- ❖ Usage grand public
 - Considérations RGPD
- ❖ Dans une application mobile
 - Choix du modèle – usage memoire, batterie & vitesse de calcul



1 – Rappel de la problématique

1.2 – Données

❖ Modèle inputs:

- Non structurées (Images .jpg)
- Large volumétrie de départ (94,110 images)
- Forte croissance anticipée (contributions utilisateurs via application mobile)

❖ Modèle outputs:

- Semi-structurées (résultats modèle .parquet)
- Structurées (résultats ACP .csv)



1 – Rappel de la problématique

1.3 – Big data : les 4 V

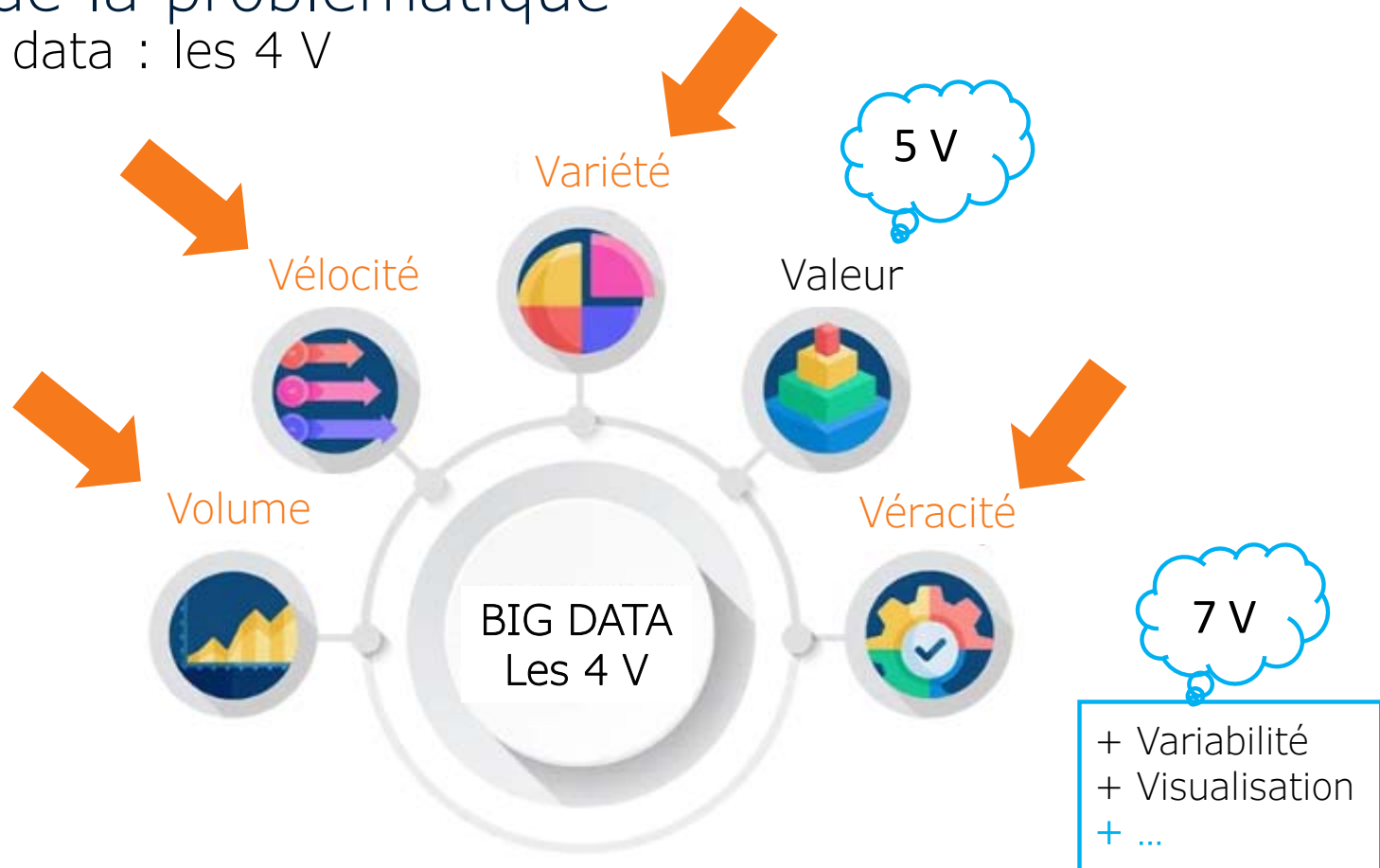


Image credits : <https://www.nexsoftsys.com/services/big-data-services.html>

2 – Choix technologiques

Contraintes:

- ❖ Grand volume (à terme) de données non-, semi- et structurées
 - object storage élastique (ni file, ni block storage adaptés)
- ❖ Machine learning – classification supervisée
 - capacités de calcul distribué et élastique (complexité / volume / rapidité)
- ❖ Solution connectée, interfaçable, économe en mémoire & évolutive
 - distribuable rapidement vers une application mobile
 - accompagnée d'un catalogue de services full-stack

➤ Solution intégrée => Cloud



2 – Choix technologiques

2.1 – Cloud provider



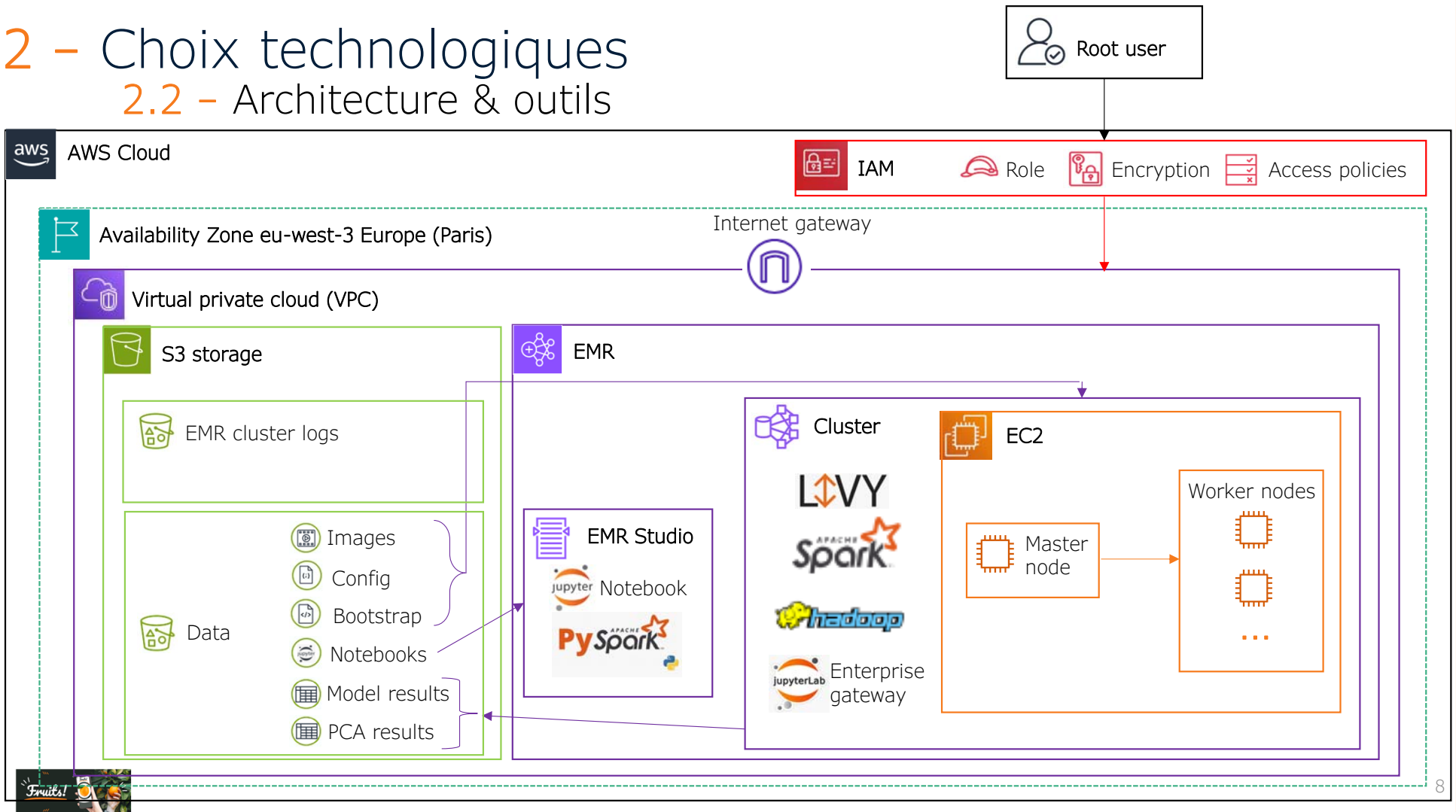
| Service category | Service type | Google Cloud product | Google Cloud product description | AWS offering | Azure offering |
|------------------|-----------------|----------------------|--|---|--------------------------------------|
| Storage | Object storage | Cloud Storage | Store any amount of data and retrieve it as often as you'd like, using Google Cloud's object storage offering. | AWS Simple Storage Service (S3) | Azure Blob Storage |
| Compute | Core compute | Compute Engine | Accelerate your digital transformation with high-performance VMs. | Amazon Elastic Compute Cloud (EC2) | Azure Virtual Machines |
| Data analytics | Data processing | Dataproc | Deploy open-source data and analytics processing services (Apache Hadoop, Apache Spark, etc.) with improved efficiency and security. | Amazon Elastic MapReduce (EMR), AWS Batch, AWS Glue | Azure Data Lake Analytics, HDInsight |



Source : <https://cloud.google.com/docs/get-started/aws-azure-gcp-service-comparison>

2 – Choix technologiques

2.2 – Architecture & outils



2 – Choix technologiques

2.3 – Considérations RGPD

- ❖ Définit les règles de collecte, traitement & conservation des données personnelles des citoyens Européens:
 - Pas de données personnelles dans le jeu de données test...
 - ...mais on ne peut exclure leur présence à l'avenir (métadonnées des photos)
- ❖ Mesures de précaution:
 - Sécurisation : traitement dans un Virtual Private Cloud (VPC) sur AWS EMR
 - Accès, rectification & droit à l'oubli: AWS S3
 - Confidentialité : gestion des accès via AWS IAM
 - Localisation : AZ => eu-west-3 Europe (Paris)



3 – Proof of Concept

3.1 – Test en local

3.1.1 – Environnement & contraintes

- ❖ Distribution Anaconda
- ❖ Jupyter Notebooks / Python 3 kernel
- ❖ Environnement Windows : ajustements du code
 - `os.environ['PYSPARK_PYTHON'] = sys.executable`
 - `os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable`
 - Utilisation de FindSpark
 - Installation de Hadoop 3.0.0 dll et Winutils
 - `Model.predict` avec `verbose=0` (utf-8 encoding errors)
 - Voir notes détaillées dans le code

Config :

- ❖ `findspark==2.0.1`
- ❖ `numpy==1.26.4`
- ❖ `pandas==2.2.2`
- ❖ `pillow==9.2.0`
- ❖ `pyarrow==10.0.1`
- ❖ `pyspark==3.5.1`
- ❖ `scipy==1.13.1`
- ❖ `tensorflow==2.17.0`
- ❖ + Hadoop 3.0.0 dll & winutils

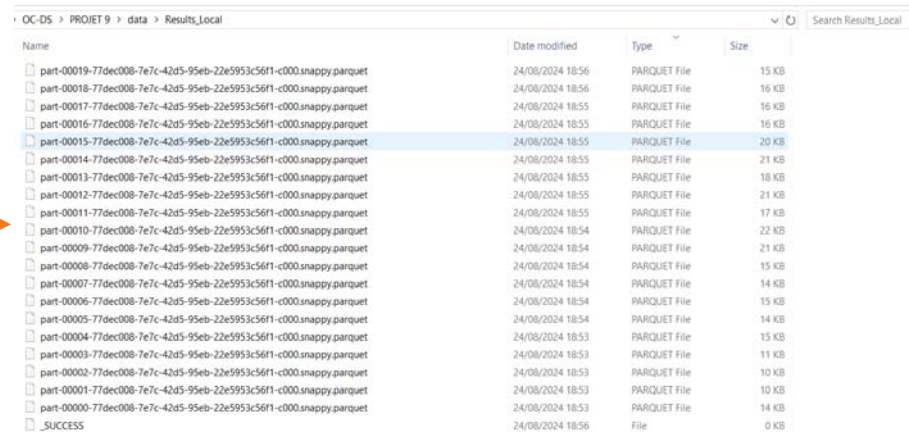


3 – Proof of Concept

3.1 – Test en local


3.1.2 – Résultats

- ❖ Dossier Results => 20 fichiers parquet de résultats du modèle + 1 fichier “_SUCCESS”



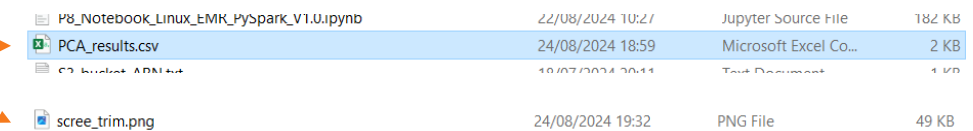
| Name | Date modified | Type | Size |
|---|------------------|--------------|-------|
| part-00019-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:56 | PARQUET File | 15 KB |
| part-00018-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:56 | PARQUET File | 16 KB |
| part-00017-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:55 | PARQUET File | 16 KB |
| part-00016-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:55 | PARQUET File | 16 KB |
| part-00015-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:55 | PARQUET File | 20 KB |
| part-00014-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:55 | PARQUET File | 21 KB |
| part-00013-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:55 | PARQUET File | 18 KB |
| part-00012-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:55 | PARQUET File | 21 KB |
| part-00011-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:55 | PARQUET File | 17 KB |
| part-00010-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:54 | PARQUET File | 22 KB |
| part-00009-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:54 | PARQUET File | 21 KB |
| part-00008-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:54 | PARQUET File | 15 KB |
| part-00007-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:54 | PARQUET File | 14 KB |
| part-00006-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:54 | PARQUET File | 15 KB |
| part-00005-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:54 | PARQUET File | 14 KB |
| part-00004-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:53 | PARQUET File | 15 KB |
| part-00003-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:53 | PARQUET File | 11 KB |
| part-00002-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:53 | PARQUET File | 10 KB |
| part-00001-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:53 | PARQUET File | 10 KB |
| part-00000-77dec008-7e7c-42d5-95eb-22e5953c56f1-c000.snappy.parquet | 24/08/2024 18:53 | PARQUET File | 14 KB |
| _SUCCESS | 24/08/2024 18:56 | File | 0 KB |

- ❖ Dossier PCA_Results => 1 fichier parquet de résultats de l'ACP + 1 objet “_SUCCESS”



| Name | Date modified | Type | Size |
|---|------------------|--------------|------|
| part-00000-43eb28bb-7559-44aa-adcb-280f6cfa13f1-c000.snappy.parquet | 24/08/2024 18:57 | PARQUET File | 2 KB |
| _SUCCESS | 24/08/2024 18:57 | File | 0 KB |

- ❖ Dossier du notebook => 1 fichier csv de résultats de l'ACP+ 1 graph png

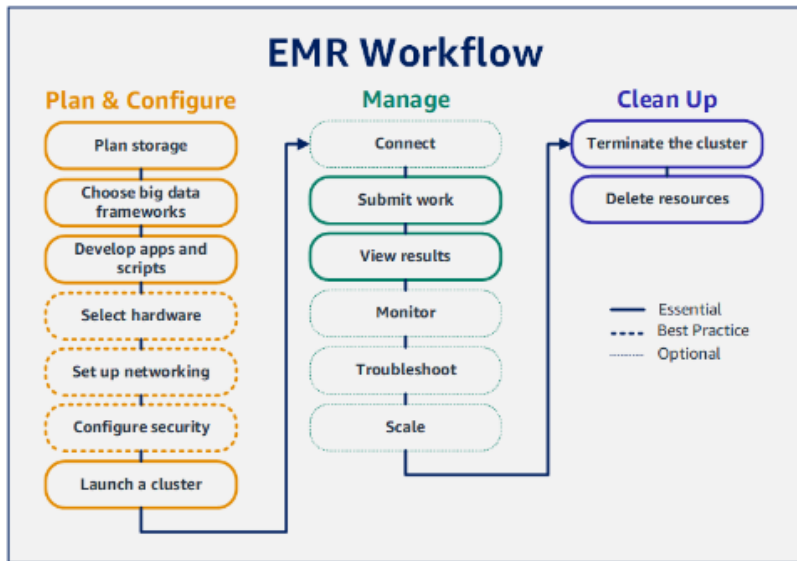


| Name | Date modified | Type | Size |
|--|------------------|-----------------------|--------|
| P8_Notebook_LINUX_EMK_PySpark_V1.0.ipynb | 22/08/2024 10:21 | Jupyter Source File | 182 KB |
| PCA_results.csv | 24/08/2024 18:59 | Microsoft Excel Co... | 2 KB |
| C2_Notebook_ABN...t | 18/07/2024 20:44 | Text Document | 1 KB |
| scree_trim.png | 24/08/2024 19:32 | PNG File | 49 KB |



3 – Proof of concept

3.2 – Déploiement dans AWS Cloud



Source : <https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-gs.html>



Création du bucket S3 & load photos



Instanciation du cluster → S3 logs bucket



Configure EMR Studio



Launch EMR Studio workspace



Open EMR Studio Jupyter Notebook Kernel



3 – Proof of concept

3.2 – Déploiement dans AWS Cloud

3.2.1 – Paramétrage

❖ Cluster:

emr-7.0.0

Application bundle

Spark
Interactive

Core
Hadoop

Flink

HBase

Presto

Trino

Custom
aws

☐ AmazonCloudWatchAgent 1.300031.1

☐ HCatalog 3.1.3

☐ Hue 4.11.0

☐ Livy 0.7.1

☐ Phoenix 5.1.3

☒ Spark 3.5.0

☐ Tez 0.10.2

☐ ZooKeeper 3.5.10

☐ Flink 1.18.0

☒ Hadoop 3.3.6

☒ JupyterEnterpriseGateway 2.6.0

☐ MXNet 1.9.1

☐ Pig 0.17.0

☐ Sqoop 1.4.7

☐ Trino 426

☐ HBase 2.4.17

☐ Hive 3.1.3

☐ JupyterHub 1.5.0

☐ Oozie 5.2.1

☐ Presto 0.283

☐ TensorFlow 2.11.0

☐ Zeppelin 0.10.1

❖ Nodes:

Provisioning configuration

Set the size of your core and task instance groups. Amazon EMR attempts to provision this capacity when you launch your cluster.

| Name | Instance type | Instance(s) size | Use spot purchasing option |
|----------|---------------|------------------|----------------------------|
| Task - 2 | m5.xlarge | 1 | <input type="checkbox"/> |
| Task - 1 | m5.xlarge | 1 | <input type="checkbox"/> |
| Core | m5.xlarge | 1 | <input type="checkbox"/> |



❖ Config:

▼ Software settings Info
Override the default configurations for specific applications on your clus

Enter configuration

```
1 {  
2   {  
3     "Classification": "jupyter-s3-conf",  
4     "Properties": {  
5       "s3.persistence.bucket": "ocds-p9-data",  
6       "s3.persistence.enabled": "true"  
7     }  
8   }  
9 }
```

❖ Bootstrap:

```
#!/bin/bash  
sudo python3 -m pip install pandas  
sudo python3 -m pip install tensorflow  
sudo python3 -m pip install pillow  
sudo python3 -m pip install pyarrow  
sudo python3 -m pip install aws-hadoop  
sudo python3 -m pip install fsspec  
sudo python3 -m pip install s3fs  
sudo python3 -m pip install matplotlib==3.5.2
```

3 – Proof of concept

3.2 – Déploiement dans AWS Cloud

3.2.2 – Monitoring

❖ Spark History Server UI:

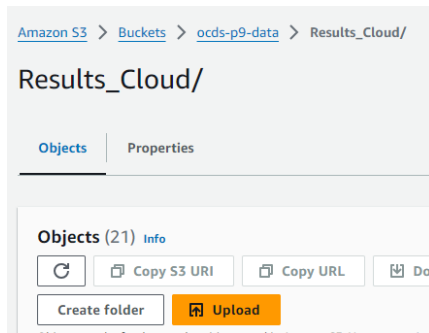


3 – Proof of concept

3.2 – Déploiement dans AWS Cloud

3.2.3 – Résultats

- ❖ S3 bucket – dossier Results_Cloud => 20 objets parquet + 1 objet “_SUCCESS”



| | | | | | |
|--------------------------|---|---------|---------------------------------------|---------|----------|
| <input type="checkbox"/> | part-00004-33b41c53-fcf8-4315-9bc9-0c9a81ae8ec4-c000.snappy.parquet | parquet | August 23, 2024, 16:08:58 (UTC+02:00) | 16.1 KB | Standard |
| <input type="checkbox"/> | part-00005-33b41c53-fcf8-4315-9bc9-0c9a81ae8ec4-c000.snappy.parquet | parquet | August 23, 2024, 16:08:58 (UTC+02:00) | 6.2 KB | Standard |

- ❖ S3 bucket => 1 fichier csv de résultats de l'ACP

| | | | | | |
|--------------------------|---------------------------------------|---|---------------------------------------|--------|----------|
| <input type="checkbox"/> | PCA_Results_Cloud_csv | - | August 23, 2024, 16:09:35 (UTC+02:00) | 1.6 KB | Standard |
|--------------------------|---------------------------------------|---|---------------------------------------|--------|----------|



Optimisations coût & performance possibles

- ❖ Intégration avec outils du catalogue AWS (Amplify / Glue / Sagemaker / CloudFront / Step Functions / Macie)
 - solution full-stack – pipeline données & traitements entièrement automatisés, détection automatique des données personnelles etc...
- ❖ Changement d’AZ en Europe
 - Dublin - cher que Paris sur EC2
- ❖ Optimisation du storage S3
 - supprimer les photos une fois traitées ou archive S3 Glacier (- cher que S3 standard)
- ❖ Spot purchase instances EC2
 - jusqu’à 90% - cher que On-Demand
- ❖ Choix des machines du cluster Compute Optimised (classe c) + rapides
 - au lieu de Memory Optimised (classe m)
- ❖ Auto-scaling, voire même migration vers EMR Serverless
 - gestion automatisée des clusters

