

PROJET N° 9

Réalisez un traitement dans un environnement Big Data sur le Cloud



Jean Vallée

Objectif : Data Scientist - 2024

Etapes

INTRODUCTION

1

ARCHITECTURE RETENUE

Cluster EMR
Stockage S3
Serveur JupyterHub

2

MISE EN OEUVRE

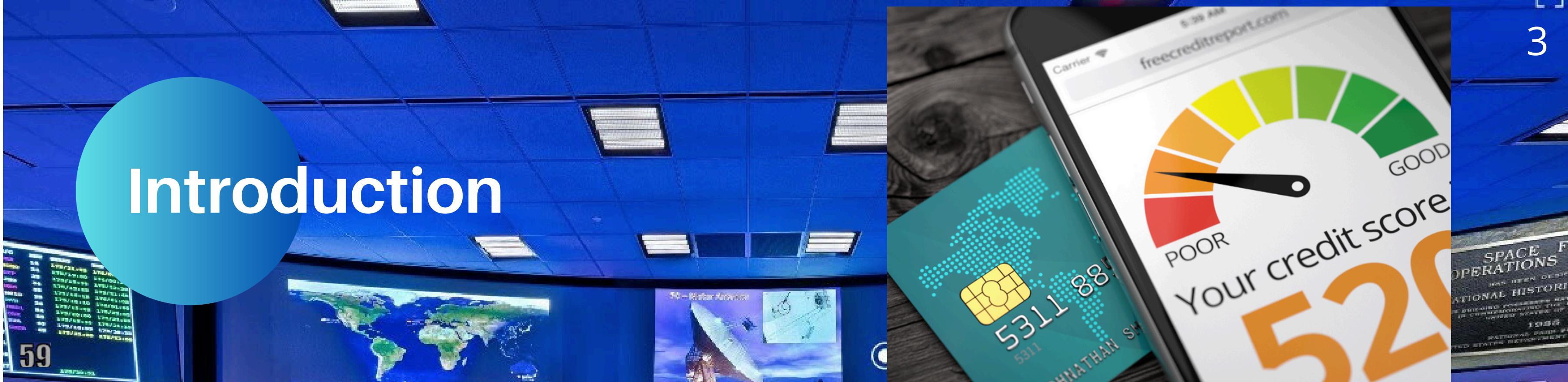
Création de cluster EMR
JDD sur S3
Analyse de coûts

3

TRAITEMENT PYSPARK

Obtention d'attributs
Réduction de dimensions
(Classification supervisée)

CONCLUSION



Contexte client

- start-up d'AgriTech "Fruits!"
- solutions de récolte

Objectifs

- app mobile (photo de fruit-->infos)
- moteur de classification d'images
- draft d'architecture Big Data

Contexte prestataire

- Notebook existant
 - modèle pré-entraîné
 - génère attributs
 - diffusion de poids
 - exécuté sur le Cloud
- RAF
 - réduction du nombre d'attributs
 - (classification d'images)



Mission & technique

- **Mes missions :**
 - créer cluster AWS EMR
 - stocker données & notebook sur S3
 - lancer le traitement des 1es couches
 - démo du cluster
 - expliquer notebook

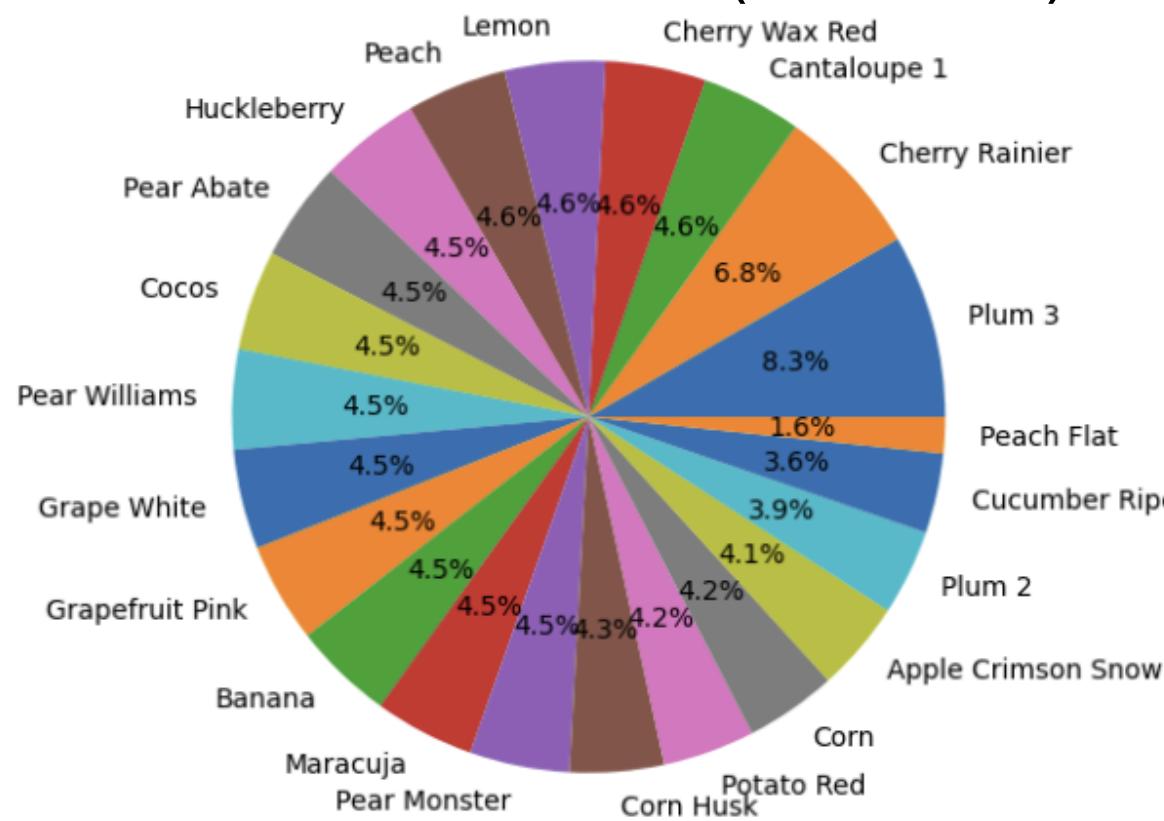
- **Outils :**
 - AWS EMR, EC2 & S3
 - PySpark
 - TensorFlow
 - JupyterHub
 - FoxyProxy
 - Bash

Introduction



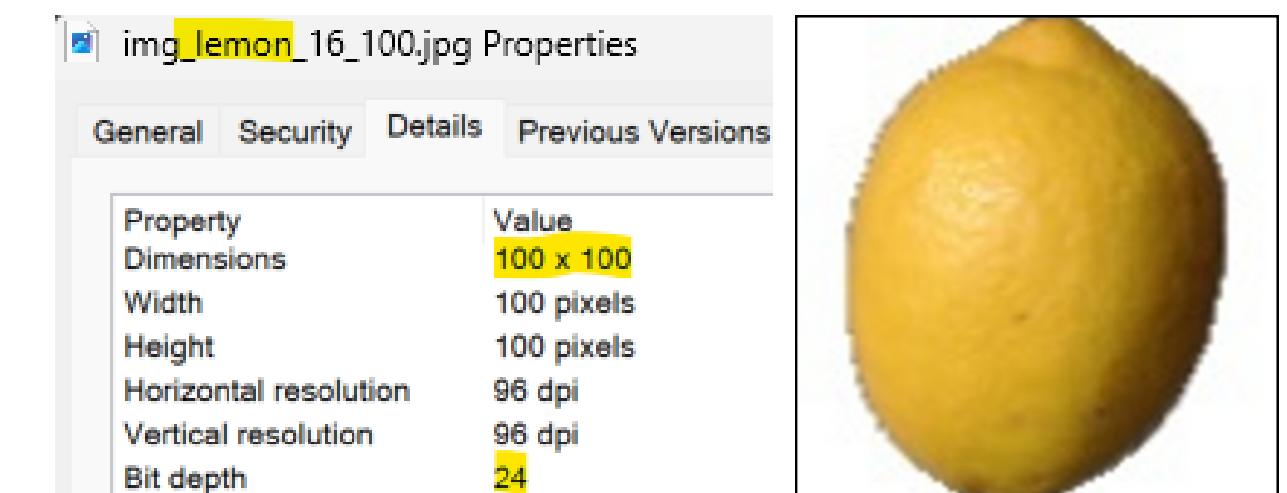
JDD réduit sur S3

- 10 812 images
- 22 catégories
- 30K attributs ($100^2 \times 3$)



JDD original sur [Kaggle fruits](#)

- 22K images .jpeg
- 131 catégories
- 150K attributs ($224^2 \times 3$)



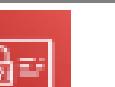
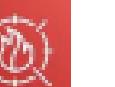


RGPD sur AWS

- cf. [Gérer conformité RGPD sur AWS](#)
- photos de fruits : RGPD N/A

KMS > Customer-managed keys

| Customer-managed keys (1) | | | | Key actions ▾ |
|--|---|---------|--|---------------|
| <input type="text"/> Filter keys by properties or tags | | | | |
| Aliases | Key ID | Status | | |
| <input type="checkbox"/> jv-emr-key-pair | d2ab8c7d-ac80-452c-a26a-7cdd80... | Enabled | | |

| Domaine | Description | Outils et services AWS |
|--|--|--|
| Protection de vos données sur AWS | Les organisations doivent « mettre en œuvre les mesures techniques et organisationnelles appropriées afin de garantir un niveau de sécurité adapté au risque, y compris la pseudonymisation et le chiffrement des données à caractère personnel ». |  AWS Certificate Manager |
| Contrôle d'accès aux données | Le responsable du traitement « doit mettre en œuvre les mesures techniques et organisationnelles appropriées pour garantir que, par défaut, seules les données à caractère personnel qui sont nécessaires au regard de chaque finalité spécifique du traitement sont |  AWS CloudHSM |
| | |  AWS Key Management Service |
| | |  AWS Identity and Access Management (IAM) |
| | |  Amazon Cognito |
| | |  AWS Shield et WAF |
| | |  AWS Resource Access Manager |

IAM > Roles

| Roles (8) <small>Info</small> | | | |
|---|--|----------------|--|
| An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust. | | | |
| Role name | Trusted entities | Last activity | |
| <input type="checkbox"/> role_ec2_access_S3 | AWS Service: ec2 | 15 minutes ago | |
| <input type="checkbox"/> AWSServiceRoleForTrustedAdvisor | AWS Service: trustedadvisor (Service-Linked) | - | |
| <input type="checkbox"/> AWSServiceRoleForSupport | AWS Service: support (Service-Linked) | 24 days ago | |

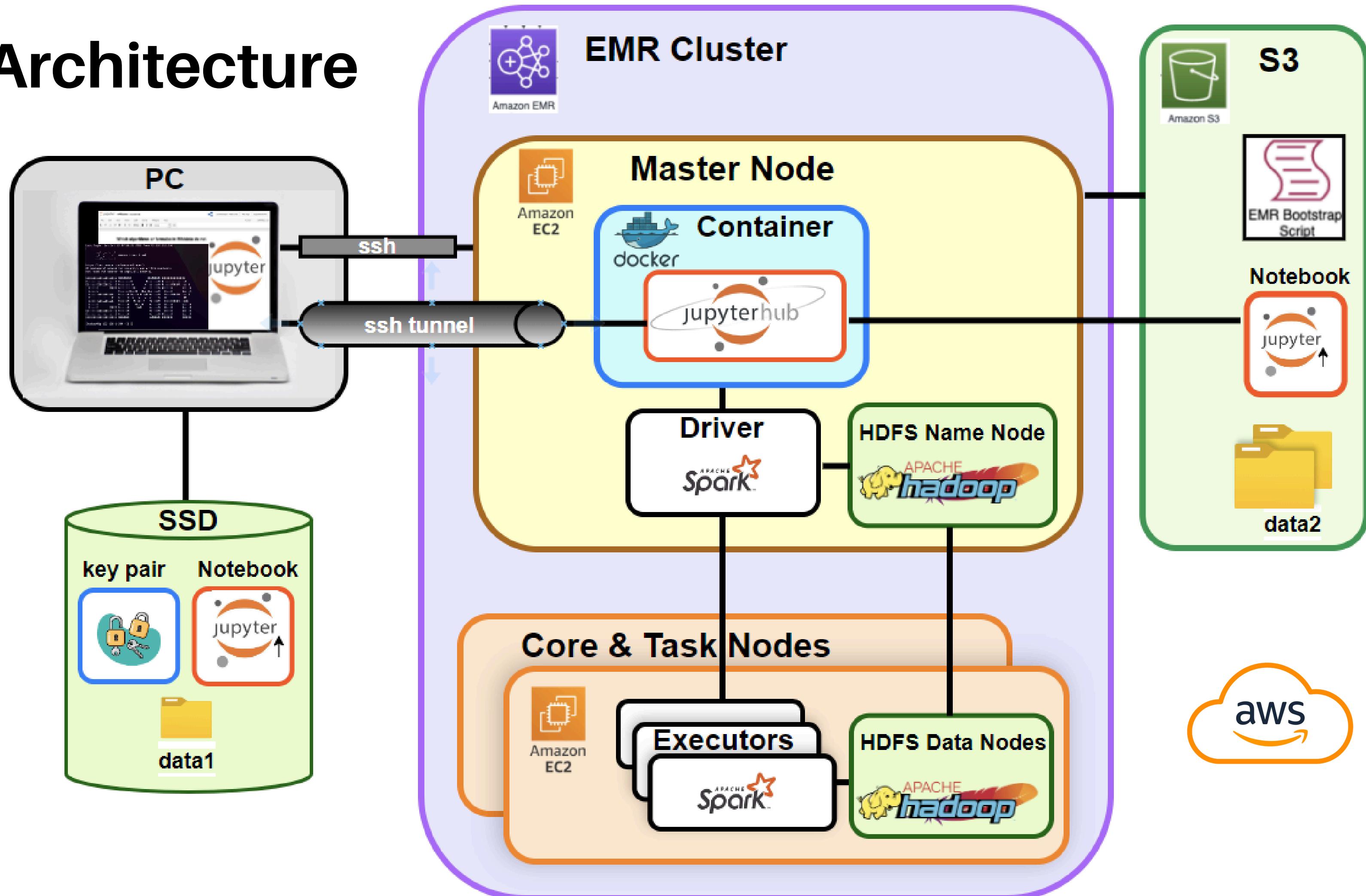
Architecture



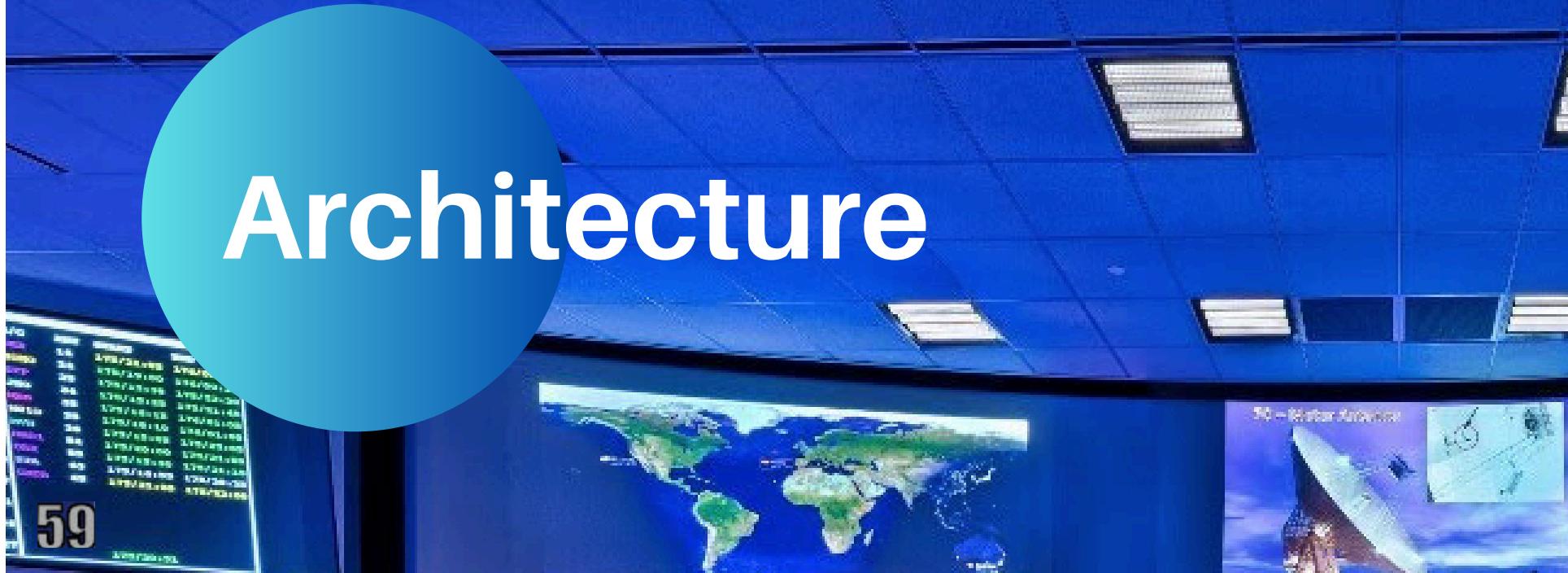
Services Cloud AWS

| Service Cloud | Description | Acronyme | Logo |
|---------------------|---|-------------------------------------|------|
| Stockage | stockage pérenne de fichiers | S3 Simple Storage Service | |
| Plateforme Big Data | cluster de noeuds pour traitement distribué | EMR Elastic Map-Reduce | |
| Machines virtuelles | noeuds du cluster | EC2 Elastic Compute Cloud | |

Architecture



Architecture



Création du cluster EMR

- Instances
 - Master généraliste
 - Core plus puissant et spécifique au calcul
 - Task idem

- Choix
 - prix spot (- cher, + performant)
 - création par clonage

Instance groups

Primary
Choose EC2 instance type

m5.xlarge
4 vCore 16 GiB memory
EBS only storage
On-demand price: USD 0.192 per instanc...
Lowest spot price: \$0.064 (us-east-1f)

Actions ▾

Use high availability
Launch highly available, more resilient cluster with three primary nodes on On-Demand Instances. Find out more ↗

▶ Node configuration - optional

Core
Choose EC2 instance type

c5ad.2xlarge
8 vCore 15.25 GiB memory
300 GiB storage
On-demand price: USD 0.344 per instanc...
Lowest spot price: \$0.146 (us-east-1f)

Actions ▾

Edit EBS volumes
64 GiB

Edit maximum spot price
Maximum USD 0.18/hr

Task 1 of 1

Name

Task - 2

Choose EC2 instance type

c5ad.2xlarge
8 vCore 15.25 GiB memory
300 GiB storage
On-demand price: USD 0.344 per instanc...
Lowest spot price: \$0.146 (us-east-1f)

Actions ▾

b_services_weekly ↵

Cost and usage graph Info

Total cost

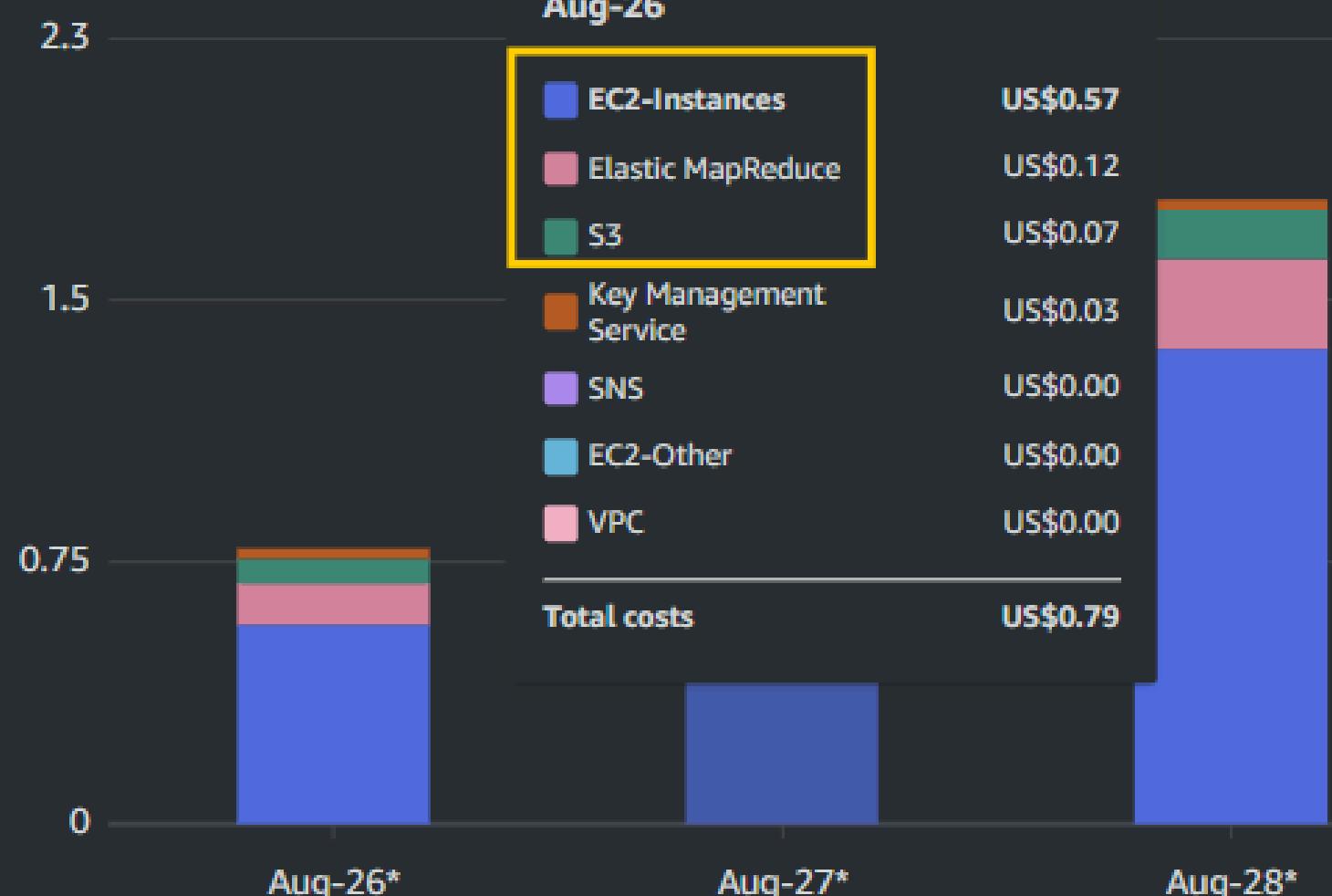
\$9.82

Average daily cost

\$1.40

Costs (\$)

3



Architecture



50 - Master Analytics

Facteur coût

- 3 mois d'essai
- *spot vs. on demand*
- rapport de suivi
- notifications SNS
 - état change
 - fichier supprimé

Traitements PySpark



JDD réduit sur S3

- 10 812 images
- 22 catégories
- Partitions
 - train 16K images
 - test 4K

But = réduction de dimensions

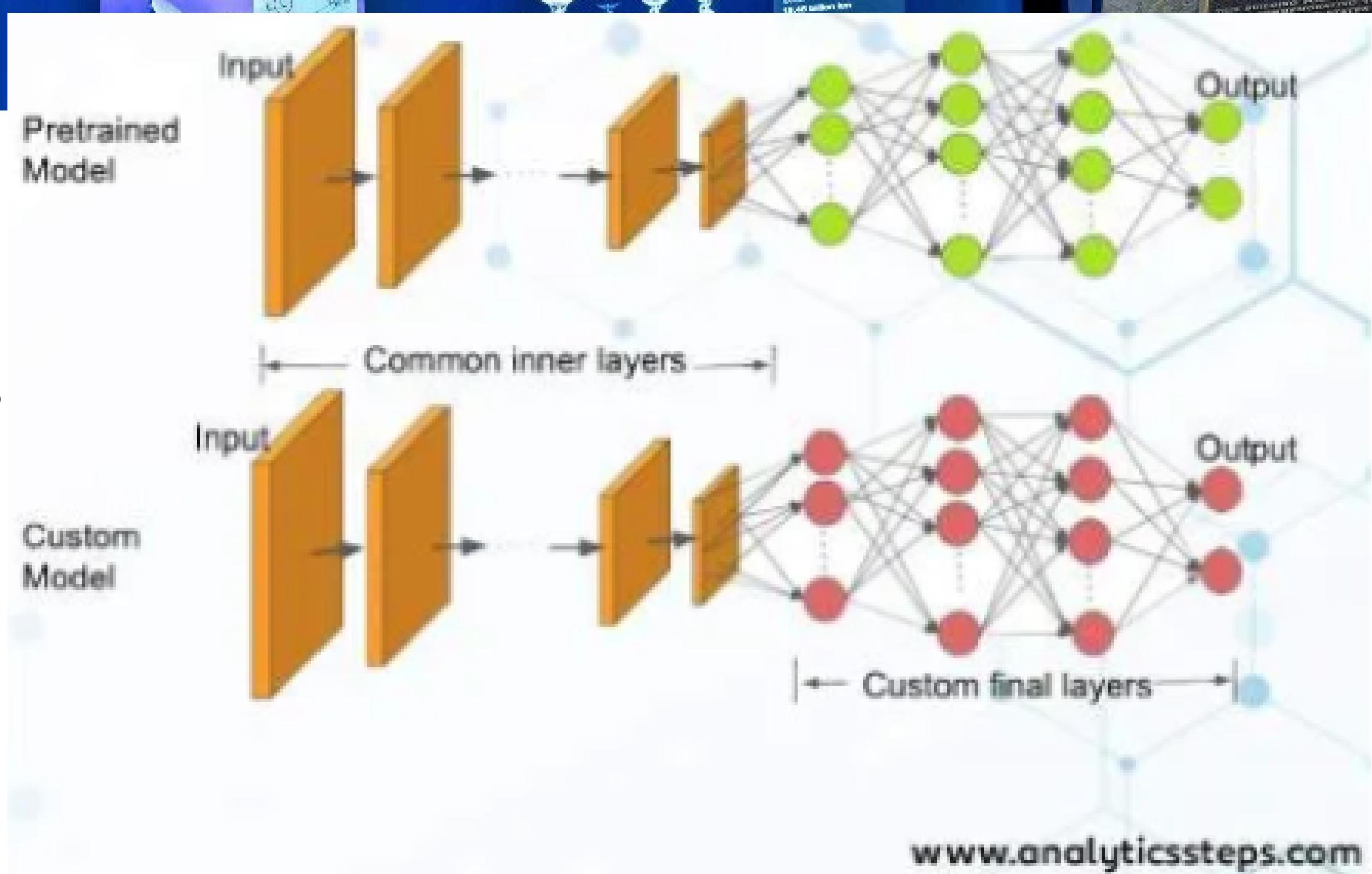
- Nombre d'attributs des images

RGB
150K

CNN
1280

PCA
10

Traitements PySpark



Transfer Learning

- modèle CNN *MobileNetV2*
 - *preprocess*
 - *troncature*

Traitements PySpark



1 CHARGEMENT D'IMAGES

Chargement dans Spark Dataframe

2 TRANSFER LEARNING

Obtention de 150K attributs significatifs

3 RÉDUCTION DE DIMENSIONS

10 attributs

4 [ENTRAÎNEMENT & VALIDATION]

Par régression linéaire

SAUVEGARDE DE RÉSULTATS

- attributs PCA
- [rapports]
- [durées]

Traitement pySpark



Etapes

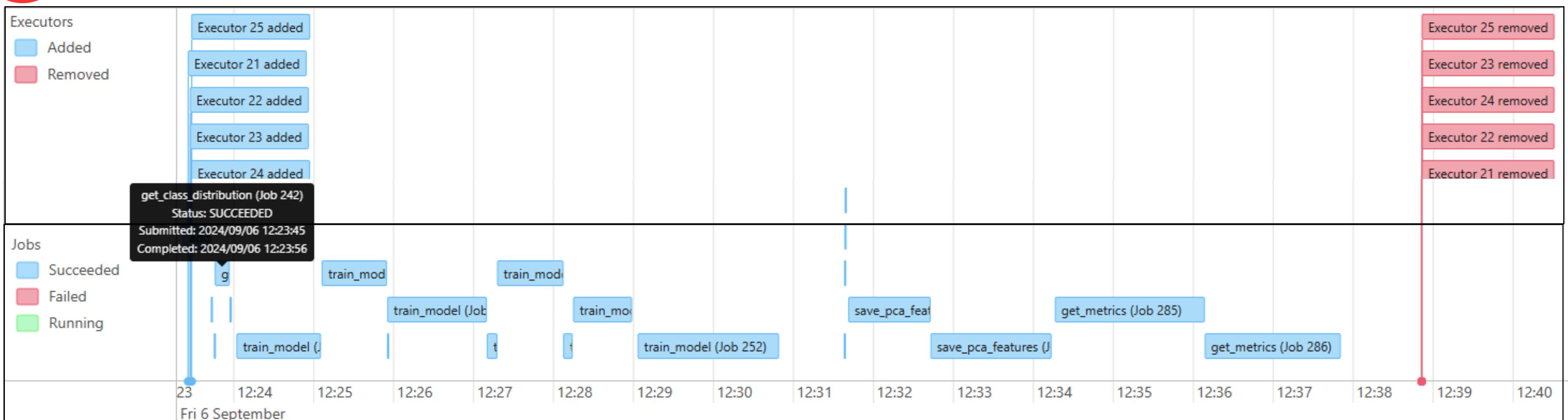
- ▶ 1. Session Spark
- ▶ 2. Modules & fonctions
- ▼ 3. Pré-traitement des DataFrame Spark
 - ▶ 3.1. Fichiers sur AWS S3
 - ▶ 3.2. Chargement des fichiers image
 - ▼ 3.3. _Feature Engineering_
 - 3.3.1. Ajout de _image_id_
 - 3.3.2. Ajout de _class_
 - 3.3.3. Ajout de _class_id_
 - 3.4. Suppression de colonnes inutiles
 - ▼ 3.5. Aggrégats
 - 3.5.1. Distribution de classes
 - 3.5.2. Nombre de classes
 - 3.5.3. Nombre d'images

- ▼ 4. Remplacement d'attributs originaux
 - ▼ 4.1. Génération d'attributs plus représentatifs
 - ▶ 4.1.1. Récupération du modèle tronqué
 - 4.1.2. Fonction de changement de résolution des images
 - ▶ 4.1.3. Fonctions de génération d'attributs par lots
 - 4.1.4. Récupération d'attributs
 - 4.2. Vectorisation d'attributs
 - 4.3. Partition du jeu de données
 - ▼ 4.4. _Pipeline_
 - 4.4.1. Etapes
 - 4.4.2. Entraînement
 - 4.4.3. Prédictions
- ▼ 5. Sauvegarde des attributs issus de PCA
 - 5.1. Répertoires de travail et de sauvegarde pérenne
 - 5.2. Contenu du DataFrame
 - 5.3. Conversion des vecteurs en colonnes
 - 5.4. Génération du CSV
- 6. Evaluation
- 7. Sauvegarde de rapports Spark UI
- ▼ 8. Fin du traitement
 - ▼ 8.1. Evaluation de durée de traitement
 - 8.1.1. Durée globale
 - 8.1.2. Durée des principales opérations
 - 8.1.3. Extrapolation pour montée en échelle
 - 8.2. Envoi de notification de fin
- ▼ 9. Annexe : Détail des étapes du _pipeline_
 - 9.1. Normalisation des attributs
 - 9.2. Réduction du nombre d'attributs
 - 9.3. Classification par régression logistique
 - ▼ 9.4. Entraînement et test
 - 9.4.1. Entraînement sur jeu d'entraînement
 - 9.4.2. Prédiction sur jeu de test

Traitements pySpark



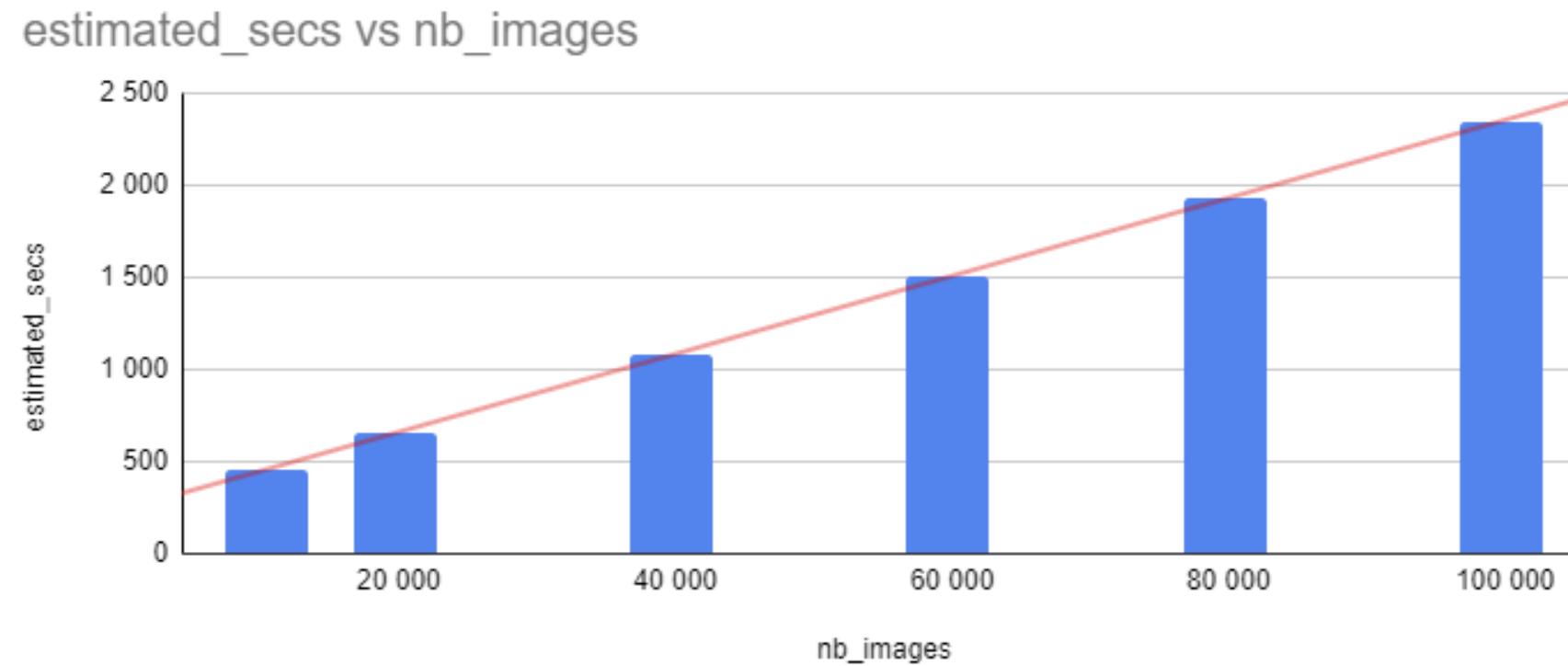
Rapport Spark UI



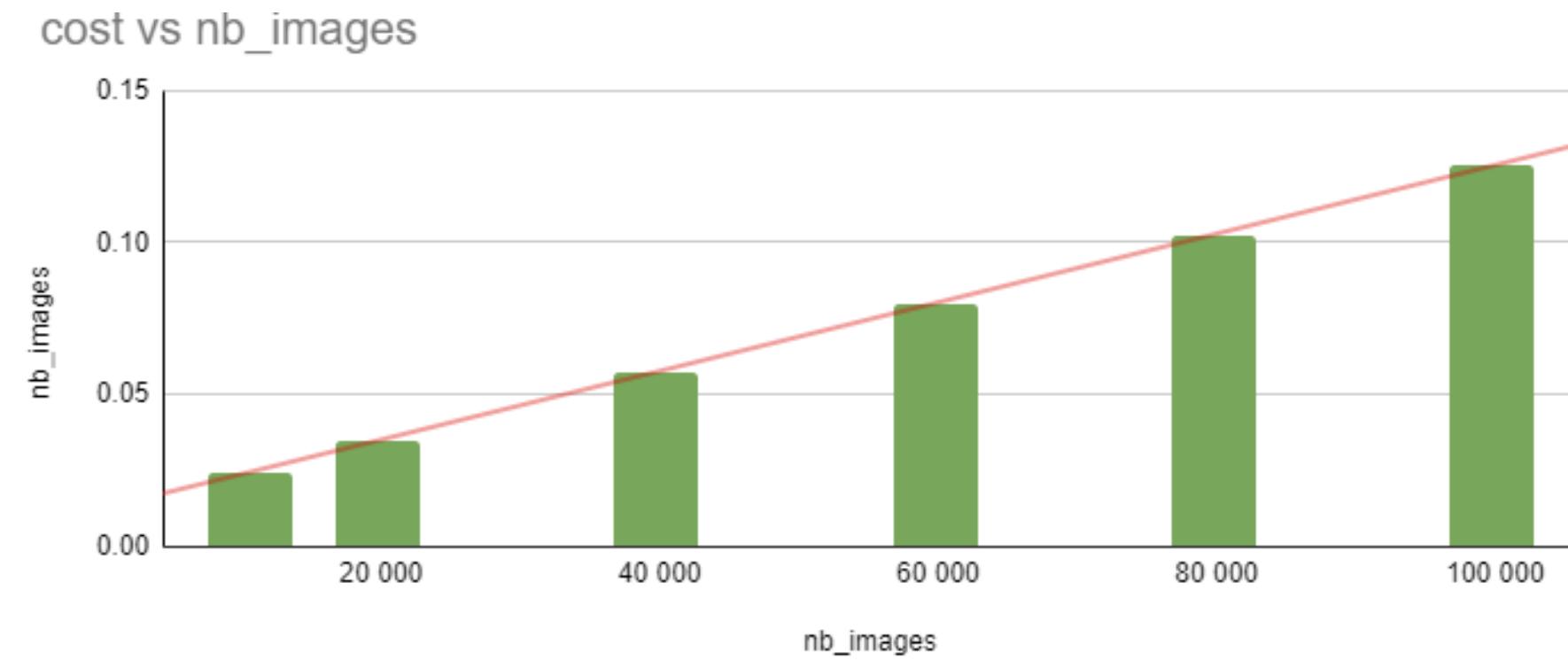
Prévision pour montée en charge



Durée de traitement



Coût



Conclusion



- Projet assez **riche**
 - compréhension d'architecture Spark
 - découverte des clusters EMR AWS
 - pré-configurés
 - clonage facile
 - analyse de coûts
 - notifications mail
 - interaction à distance
 - quelques limites
- pratique de pySpark
 - adaptation du code au traitement distribué
 - estimation de durée de traitement
 - quelques difficultés
- **Solution** appropriée à la problématique
 - temps d'exécution et coût raisonnables
 - configuration simplifiée

Questions

