

# Projet 8 - Déployez un modèle dans le cloud

Création d'une architecture Big Data dans un environnement AWS (EMR, S3, IAM) via des scripts PySpark pour préparer le passage à l'échelle en termes de volume de données

[Lien livrables en fin de présentation](#)



# Sommaire

---



Rappel de la problématique



Présentation du jeu de données



Processus de création de l'environnement Big Data, S3 et EMR



Réalisation de la chaîne de traitement des images dans un environnement Big Data dans le Cloud



Démonstration d'exécution du Script PySpark

puis remarques & axes d'amélioration.



# Rappel de la Problématique



# Rappel de la problématique



## Fruits!

**La start-up :** Fruits! est une start-up de l'AgriTech qui souhaite proposer des solutions innovantes pour la récolte des fruits.

**Concept :** Développer des robots cueilleurs intelligents pour préserver la biodiversité des fruits en permettant des traitements spécifiques.

**Objectif :** Se faire connaître en mettant à disposition une app mobile pour que les users (grand public) puisse obtenir des informations sur le fruit pris en photo.

**Solution :** Mise en place d'une première version du moteur de classification des images de fruits, d'une première version de l'architecture Big Data nécessaire.

### Contraintes :

- Le volume de données va augmenter très rapidement
- RGPD - serveurs européens
- Les coûts à surveiller



A top-down view of several ripe yellow bananas scattered on a brown background. The bananas are arranged in a way that they form a loose, abstract shape. The text "Présentation du jeu de données" is centered over the image in a white serif font.

# Présentation du jeu de données

# Présentation du jeu de données

- Source du jeu de données : [Kaggle](#)
- Type de données = Images de fruits et de légumes sur fond blanc, sous différents angles, extraites de vidéos de type timelapse
- Configuration du dossier : chaque variété est un dossier qui comporte les photos de ladite variété
- Type d'images = JPG 100 x 100
- Taille du dataset utilisé : 22688 images

J'ai fait 2 itérations du projet, la première fois j'ai utilisé seulement 3 variétés, et seulement une partie des images donc 33 images au total

 Blueberry



 Pear Kaiser



 Tomato Maroon

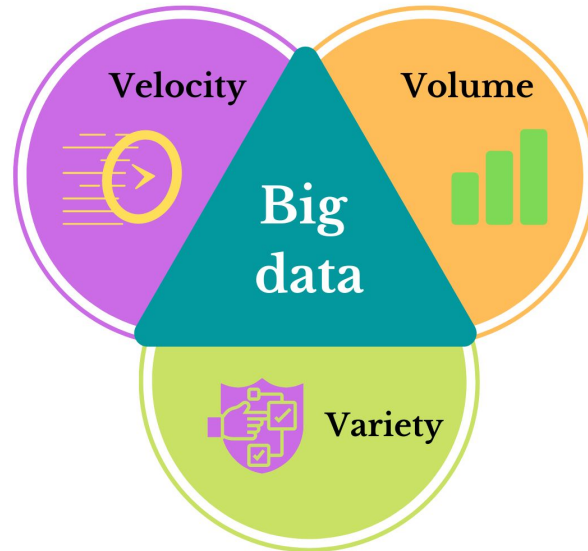


# Processus de création de l'environnement Big Data, S3 et EMR



# Une problématique Big Data

“Le volume de données va augmenter rapidement”





# Apache Spark pour les calculs distribués



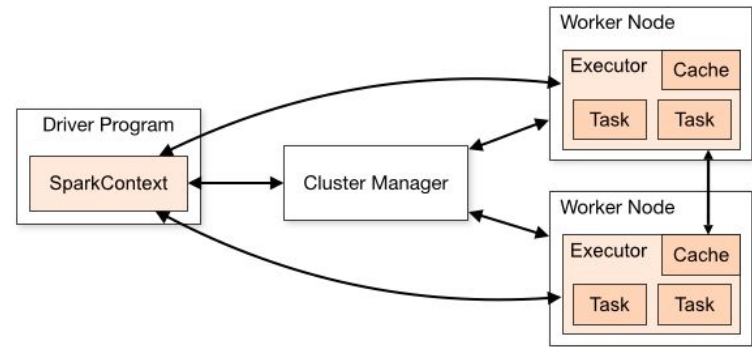
Les ressources d'une seule machine sont limitées et ne permettent pas d'effectuer un grand nombre de tâches.



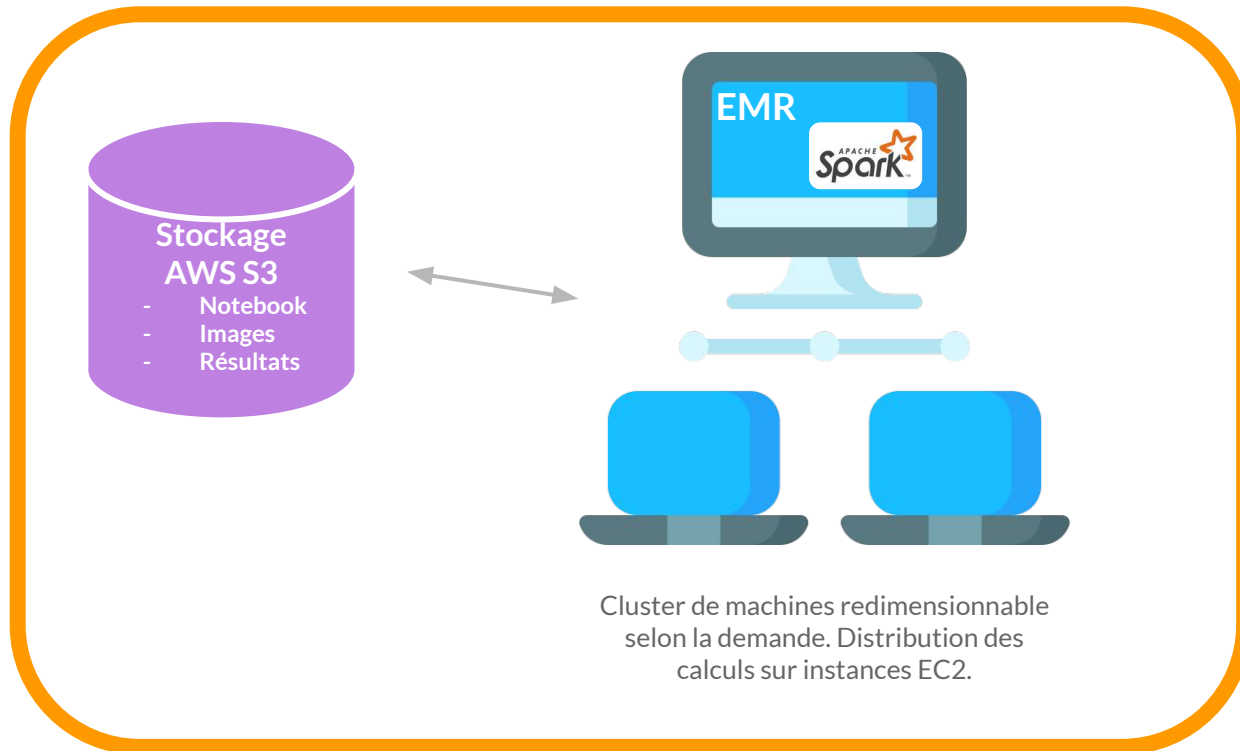
Il faut diviser les tâches en sous-tâches et les distribuer (map) sur plusieurs machines, une fois effectuées il faut les réassembler (reduce).



C'est ce que permet de faire Apache Spark (ainsi qu'Hadoop).



# L'architecture retenue : les services Amazon



**Accès Sécurisé via IAM**  
(Identity and Access Management)



**RGPD**

- Données en Europe / Stockholm
- informer les utilisateurs de l'utilisation de leurs données
- durée de conservation pour le traitement seulement (<15j)

# Paramétrage du cluster

## Informations sur le cluster

ID de cluster

j-3PFNW0WICHUJH

Configuration de cluster

Groupes d'instances

Capacité

1 primaire(s) 2 unité(s) principale(s) 0 tâche(s)

## Applications

Version d'Amazon EMR

emr-6.3.0

Applications installées

Hadoop 3.2.1, JupyterHub 1.2.0, Spark 3.1.1,

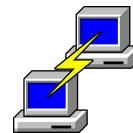
TensorFlow 2.4.1



Connexion via tunnel SSH

avec Putty

puis FoxyProxy



## Installation Bootstrap : action d'amorçage

```
#!/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
```

Travail du script sur :



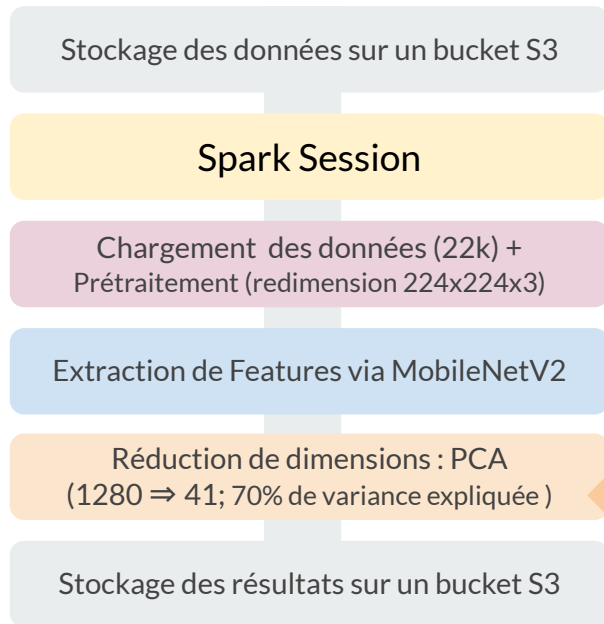




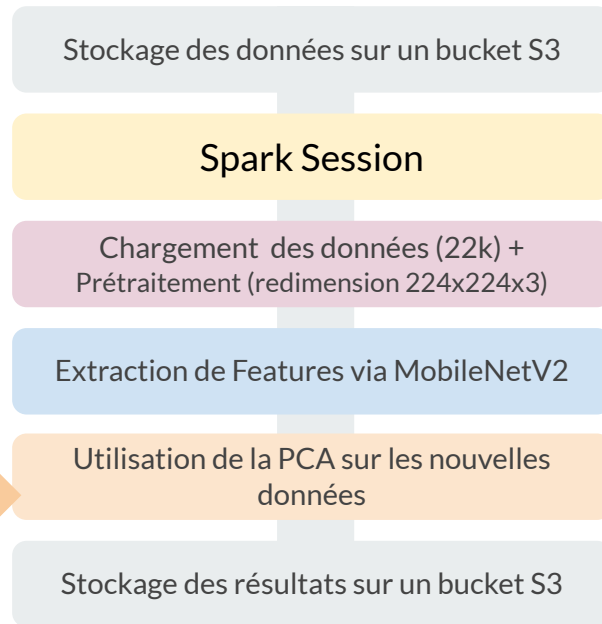
# Chaîne de traitement des images

# Chaîne de traitement des images

## 1ère utilisation sur 22k images et entraînement de la PCA



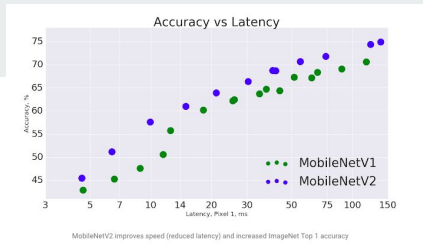
## Réutilisation test sur une sélection d'images avec import de la PCA



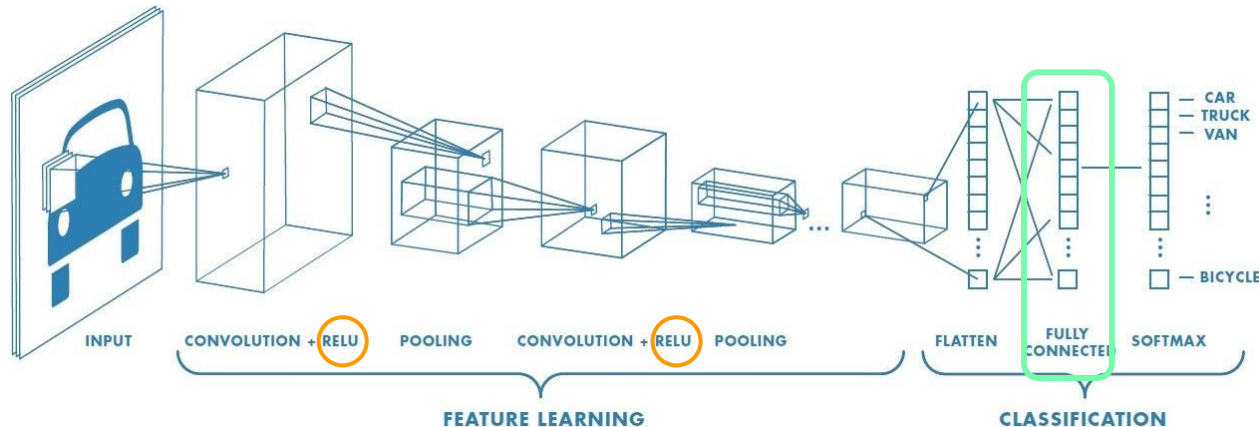
Sauvegarde  
de la PCA et  
réutilisation



# Zoom sur MobileNetV2



- Google - 2018, version améliorée de MobileNetV1. Conçue pour fonctionner sur mobile. Edge Computing.
- Les poids des modèles sont diffusés sur l'ensemble des nodes du cluster.
- Modèle pré-entraîné sur plus d'un million d'images classées en 1000 catégories différentes : on peut extraire des features sans entraînement.
- Convolutional Neural Network (CNN), Linear Bottlenecks (X fonction d'activation), Inverted Residuals
- Conservation de l'avant dernière couche (et non celle de classification - softmax)
- 1280 features en sortie





# Démonstration

# Inventaire du bucket sur test partiel (33 images)



```
/root :
- Result/
- Test/
- final_features.csv/
- jupyter/

/Result :
- _SUCCESS
- part-00000-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00001-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00002-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00003-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
...
- part-00014-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00015-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00016-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00017-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00018-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00019-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet
- part-00020-3b1a07d5-c784-4322-b5db-f25e165c3d41-c000.snappy.parquet

/Test :
- Blueberry/
- Pear Kaiser/
- Tomato Maroon/

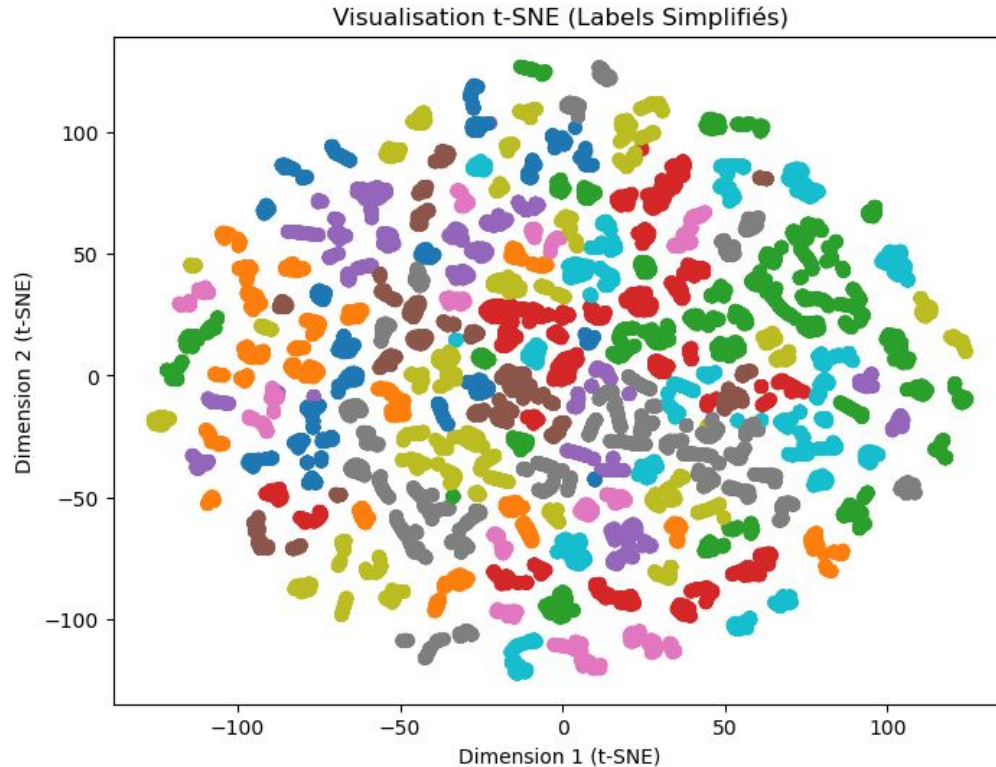
/jupyter/jovyan :
- .s3keep
- Guarneri_Naomi_1_notebook_fruits_pyspark_072023.ipynb
- notebook.ipynb
```



A photograph of a bustling fruit market stall. Numerous bunches of ripe yellow bananas hang from the ceiling. Below them, there are large green watermelons, several pineapples, and clusters of green and purple grapes. In the foreground, there are baskets of red and green apples. The stall is cluttered with various items, including cardboard boxes with the 'Capi' logo, a digital scale, and several price tags. One tag reads 'MELANCIA 2,00/kg', another 'Manga 5,50/kg', and a third '15'. In the background, there are signs for 'VISA' and 'RECANTO 007'. The overall atmosphere is one of a busy, traditional market.

# Conclusions & Remarques

# Visualisation de la classification non supervisée





# Conclusions & Remarques



Ce que nous soumettons :

- Une architecture qui permet une mise à l'échelle quand les données augmenteront, avec la possibilité d'augmenter automatiquement le nombre d'instances en fonction du nombre de données
- Une chaîne de traitement des images qui permet d'en extraire les features via MobileNetV2 et une réduction de dimension pour réduire le stockage

Difficultés rencontrées :

- Paramétrage de la console AWS complexe (énormément de possibilités)
- Difficile de savoir d'où viennent les problèmes, il faut faire preuve de persévérance

Pistes d'améliorations & idées :

- Utiliser les features pour entraîner un modèle de classification
- Automatisation de la chaîne de traitement : utilisateur scan son image et API pour prédire le fruit via modèle de classification
- Exploiter les photos prises par les utilisateurs, donner un statut de contributeur, pour identifier les images avec faible taux de précision (rgpd)
- Identifier les fruits mûrs
- Identifier les maladies
- Possibilité de passer par Google Cloud Platform (Google Cloud Dataproc / Google Cloud Storage)

# Lien livrables



- [Lien du projet sur GitHub](#) dont CSV de résultats
- [Lien du drive](#) avec images et CSV