



# Parcours Data Scientist

---



## Projet N°8 : Déployez un modèle dans le cloud



# Fruits!

Daniel CHASTANET



# Sommaire

**Rappel de la  
problématique**



**Base de  
données**

**Le big Data**



**Les  
solutions  
AWS**

**Traitement**



**Conclusion**



# Rappel de la problématique

« **Fruits!** », start-up de l'AgriTech

But à long terme :

Préserver la biodiversité des fruits par le traitement spécifique de chaque espèce via la cueillette par des robots cueilleurs intelligents.

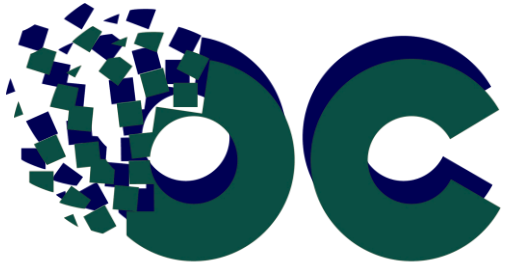
1ere étape : se faire connaître :

Application mobile : Une photo d'un fruit → des informations sur le fruit.

Objectifs de la mission :

- Mettre en place une première version du moteur de classification des images de fruits.
- Construire une première version de l'architecture Big Data nécessaire.





# Partie I – Le jeu de données



Image : [facilogi.com](https://www.facilogi.com)

## Fruits 360

A dataset with 90380 images of 131 fruits and vegetables



**Pommes :**



**Avocats :**



**Bananes :**

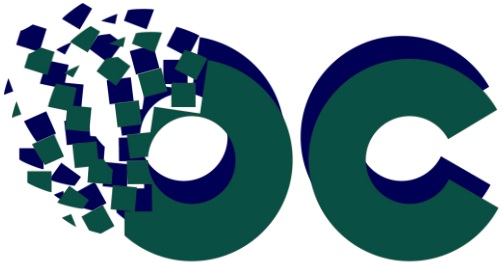


**Carambolas :**



**Choux fleur :**





## Partie II – Le big DATA

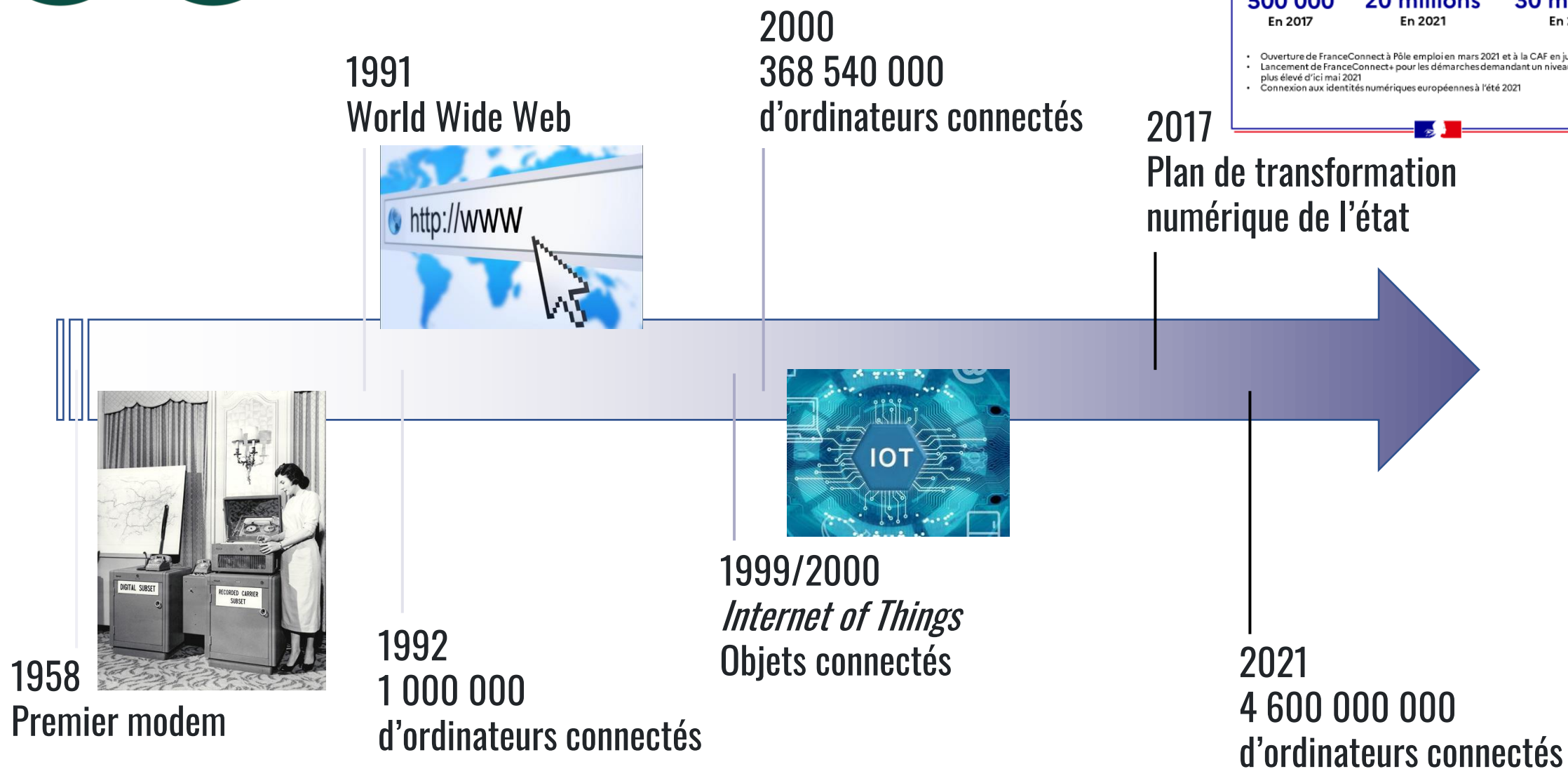


Image : [acualiteinformatique](#)





# Les données



# Le Big Data

Volume



Velocity



Variety



Image : DataBricks

## **Contraintes à respecter :**

- ☐ L'équilibrage de la charge
- ☐ L'optimisation des transferts
- ☐ La tolérance aux pannes





# Map Reduce – Hadoop

2002, Doug Cutting et Mike Cafarella – Projet Nutch

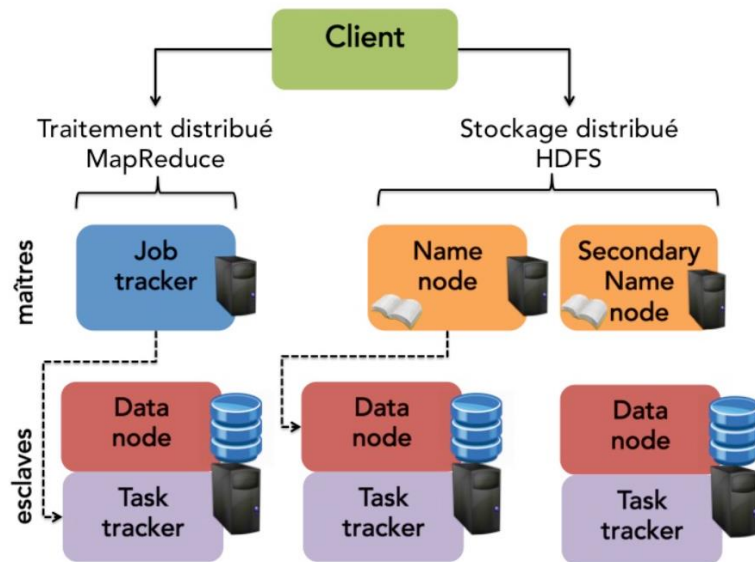


Projet Hadoop 2006  
Fondation Apache 2008



# Map Reduce – Hadoop

2002, Doug Cutting et Mike Cafarella – Projet Nutch



Projet Hadoop 2006  
Fondation Apache 2008

I. Système HDFS (Hadoop Distributed File System)  
Distribuer / Répliquer / Optimiser

II. Map Reduce  
Diviser / Résoudre / Réduire

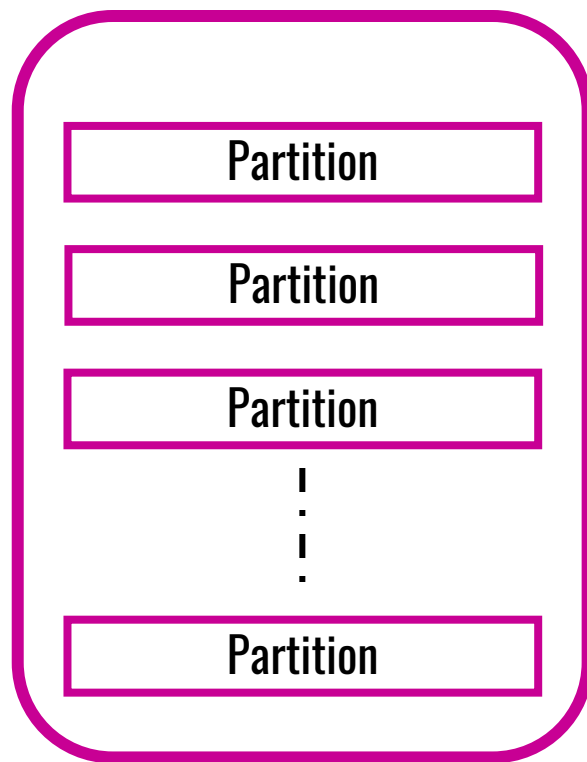
III. Yarn (Yet Another Resource Negotiator)  
Gestion des ressources / tâches



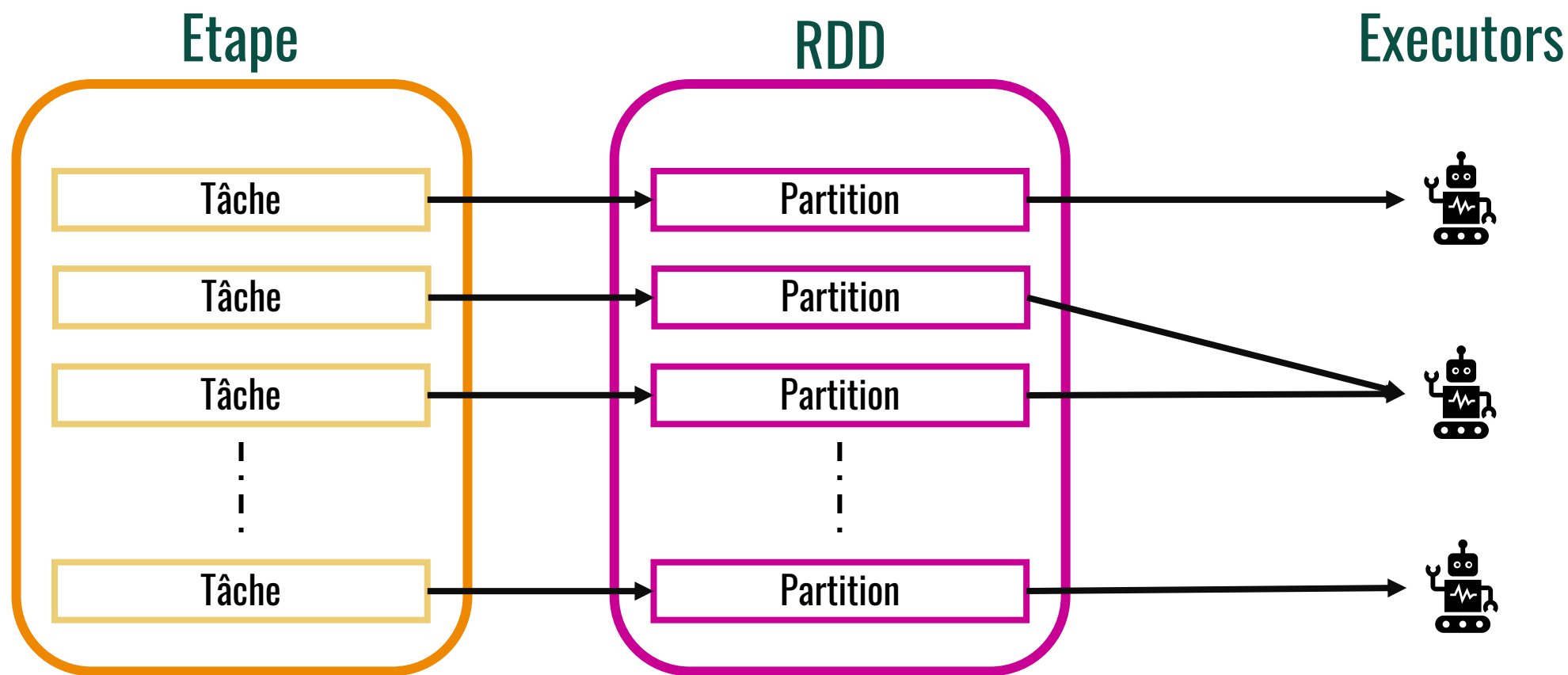
- Elargissement du cadre map reduce
- Ram vs Disque dur
- Resilient Distributed Dataset (RDD)



## Resilient Distributed Dataset (RDD)

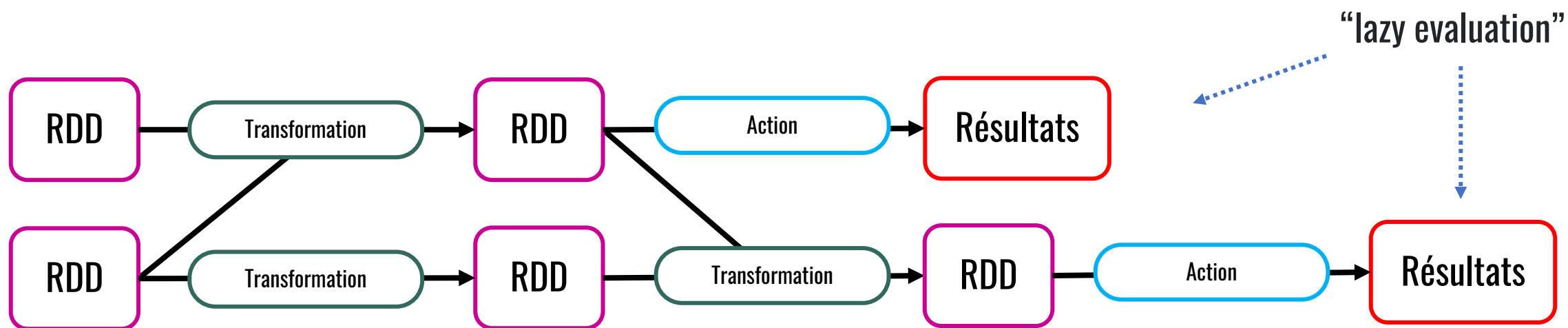


## Job Spark



## Job Spark

Directed acyclic graph (DAG)







# Spark / Pyspark

PySpark 

Python

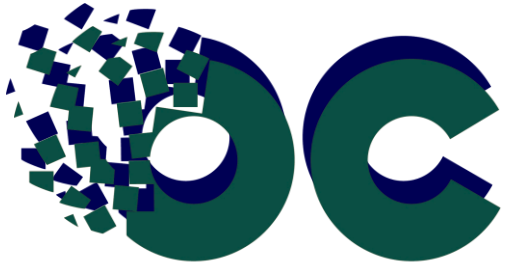


APACHE   
Spark™

Scala

**MLlib**  
*The Machine Learning Library*

 Spark SQL



# Partie III – Les solutions AWS

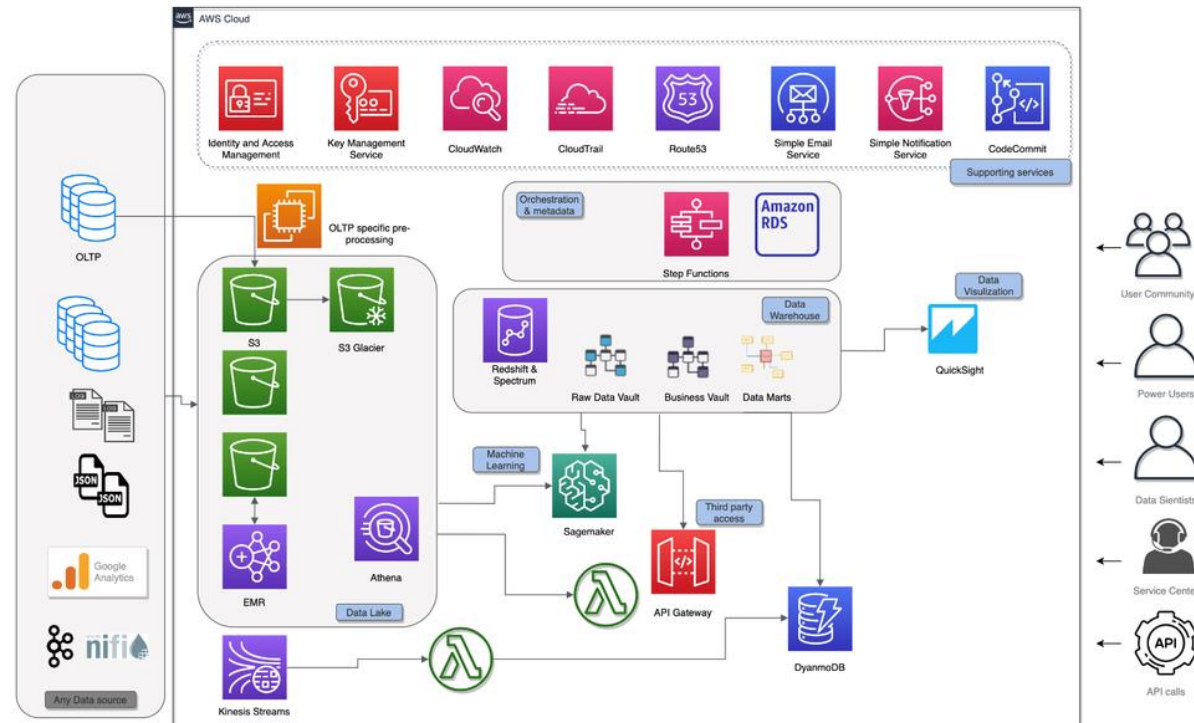


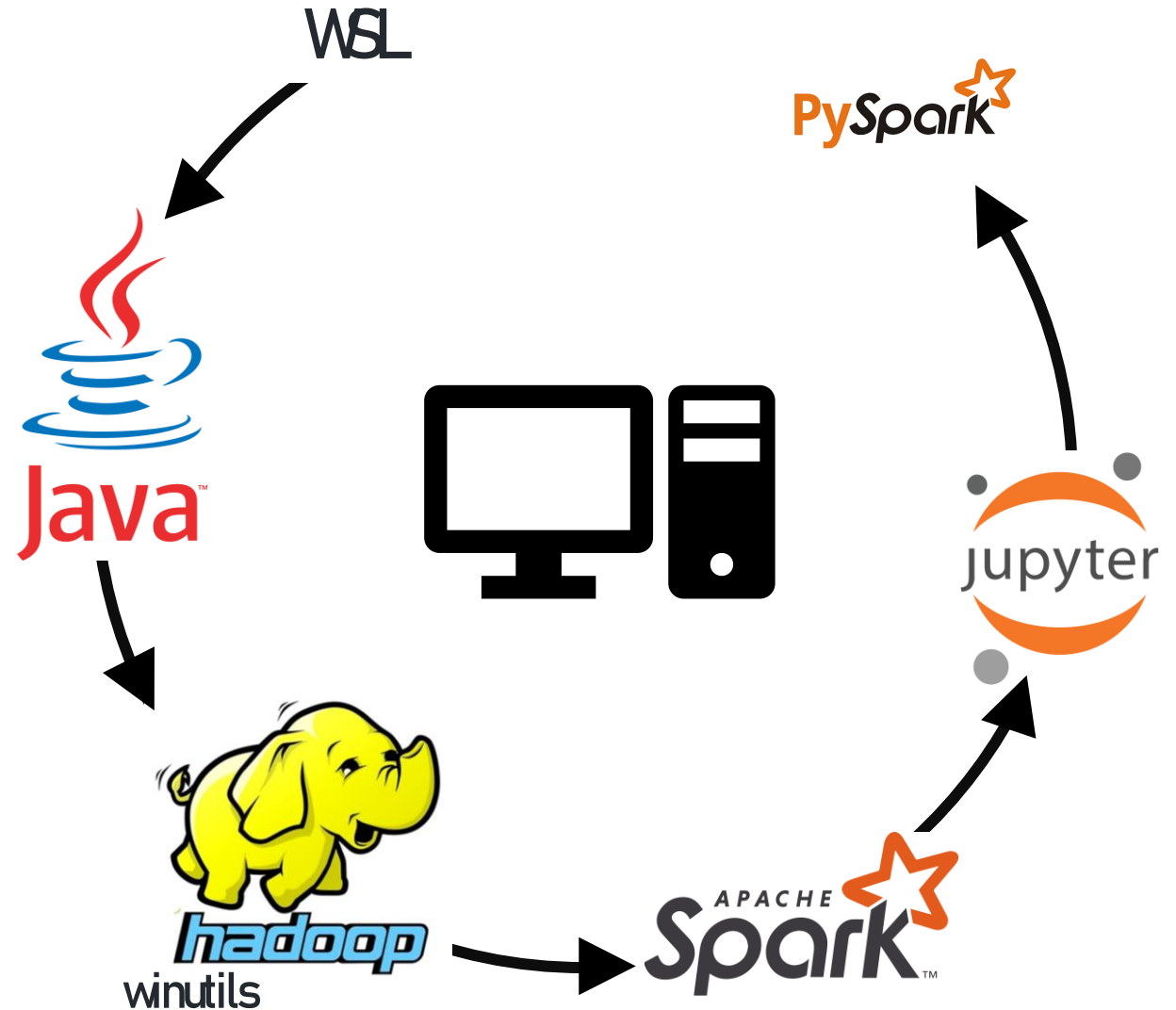
Image : [www.tomorrowsservices.lu](http://www.tomorrowsservices.lu)

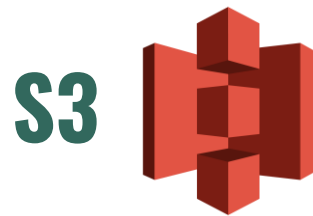


## Version locale

### Machine virtuelle en local

- Winutils
- Java
- Hadoop
- Spark
- Jupyter notebook
- Pyspark

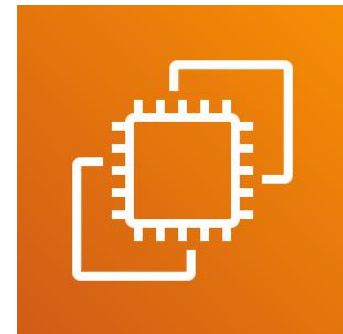




SageMaker  
€€€



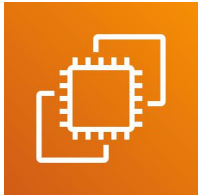
EMR  
€€



EC2  
€



# Elastic Compute Cloud (EC2)



- plateforme de calcul / + de 500 instances
- Tout OS (images)
- Bande passante (400 Gbit/s)

Instances (1) <a href="#">Info</a>				
<input type="text" value="Search"/>				
<input type="checkbox"/>	Name ▾	Instance ID	Instance state ▾	Instance type ▾
<input type="checkbox"/>	OCP8	i-0bdecbe21b3d4a460	⊖ Stopped 🔍	t2.micro



- ❖ Le moins cher
- ❖ Tout est personnalisable
- ❖ Diversité



- ❖ Pas de gestion de mise à l'échelle
- ❖ (Pas fait pour la Datascience)  
Il faut tout installer



# Amazon Elastic MapReduce (EMR)

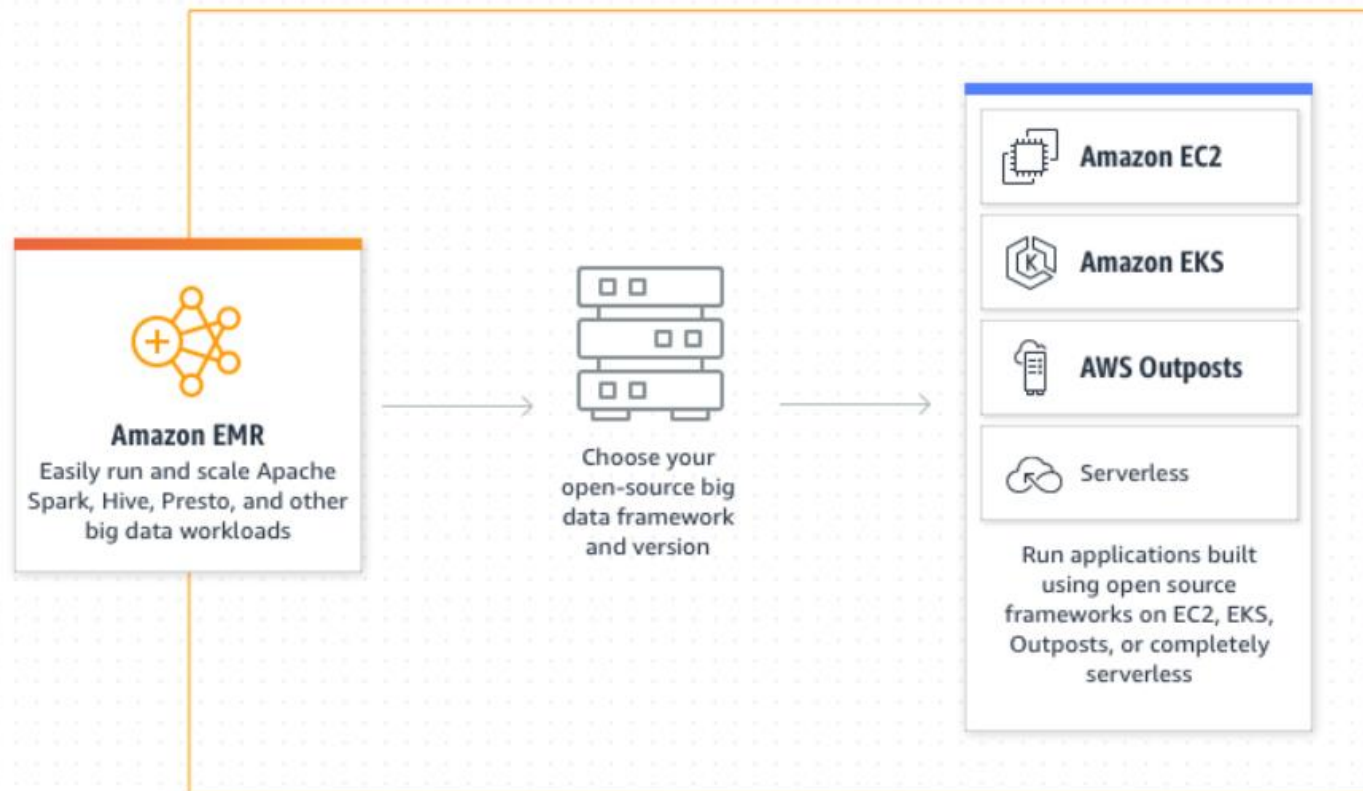
## Plateforme Big Data préconfigurée avec Hadoop



❖ Mise à l'échelle  
❖ préconfiguré



❖ Pas fait pour la  
Datasciences







## Plateforme Big Data de Datascience

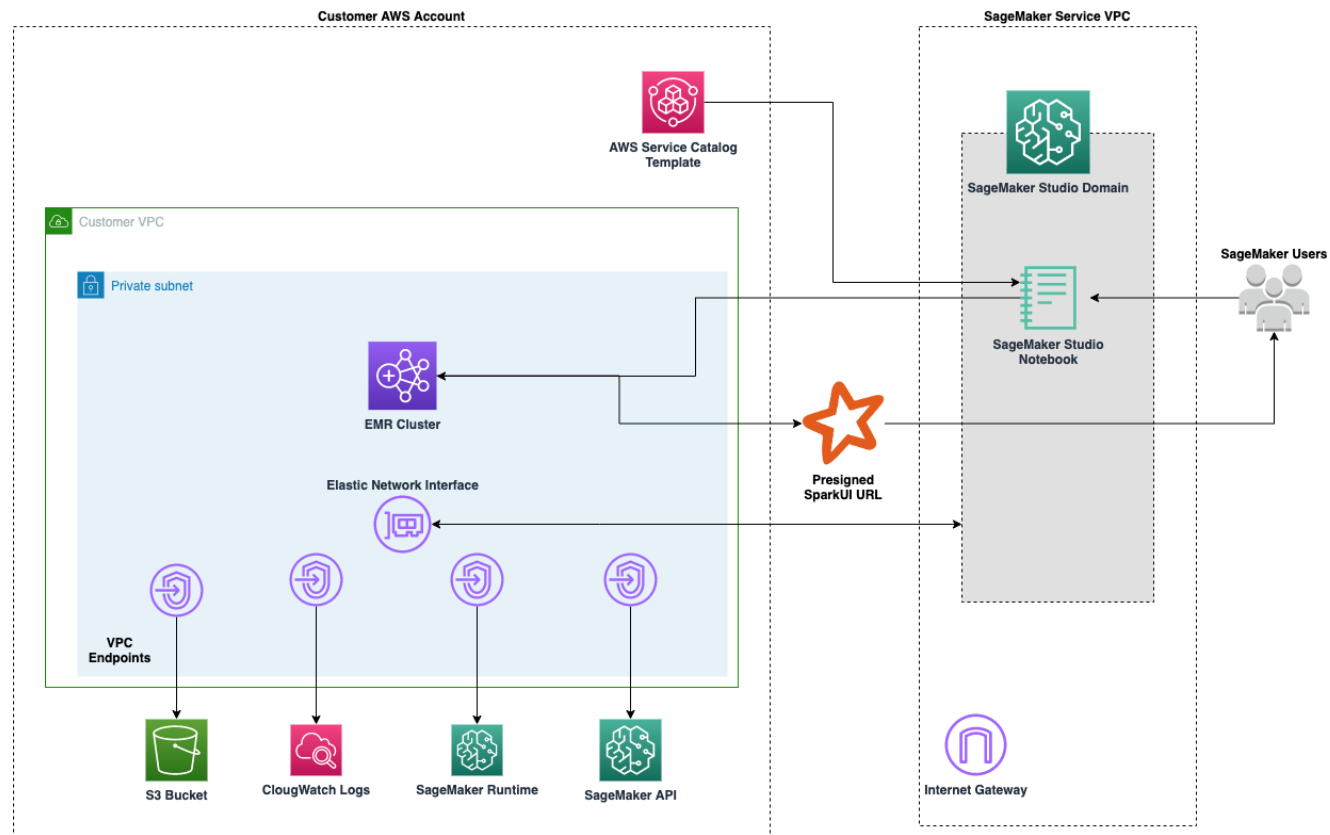


❖ Le tout compris



❖ Le plus cher

❖ Plus difficile à « gérer »



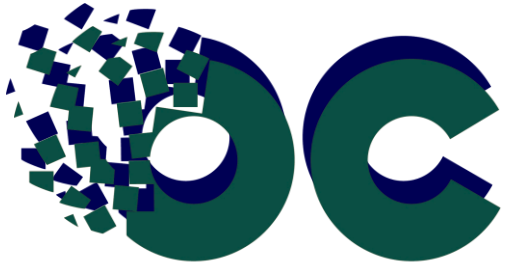
```
P8_cloud.ipynb  x  Untitled.ipynb  x
+  -  Copy  Paste  Run  Code  Settings  Git  2 vCPU + 4 GiB  Cluster  Python 3 (Data Science)  Share

[17]: %load_ext sagemaker_studio_analytics_extension.magics
      %sm_analytics emr connect --cluster-id j-3G66VFRG5F16R --auth-type None --language python

The sagemaker_studio_analytics_extension.magics extension is already loaded. To reload it, use:
      %reload_ext sagemaker_studio_analytics_extension.magics
Successfully read emr cluster(j-3G66VFRG5F16R) details
Initiating EMR connection..
The sparkmagic.magics extension is already loaded. To reload it, use:
      %reload_ext sparkmagic.magics
Starting Spark application

ID      YARN Application ID  Kind  State  Spark UI  Driver log  User  Current session?
1  application_1652797365845_0003  pyspark  idle  Link  Link  None  ✓

SparkSession available as 'spark'.
{"namespace": "sagemaker-analytics", "cluster_id": "j-3G66VFRG5F16R", "error_message": null, "success": true, "service": "emr", "operation": "connect"}
```



## Partie IV – Le traitement

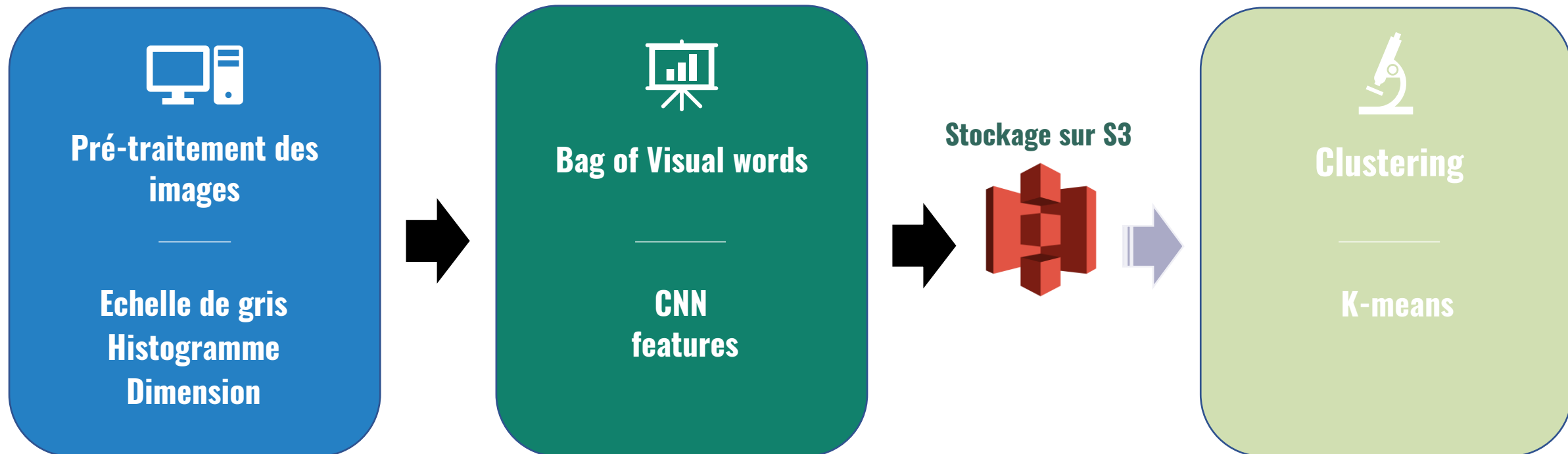


Image : [www.genaris.fr](http://www.genaris.fr)



# Le traitement des données

Transfert Learning (projet 6)



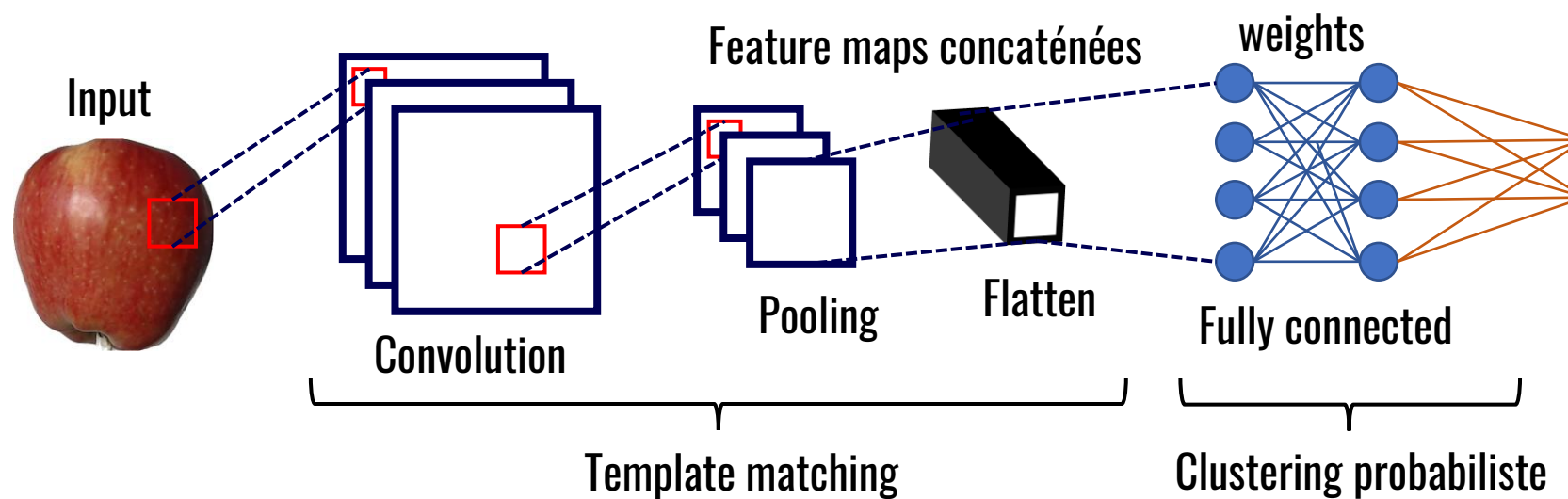


# Le traitement des données



Bag of Visual words

CNN  
K Keras



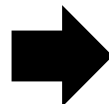


# Le traitement des données

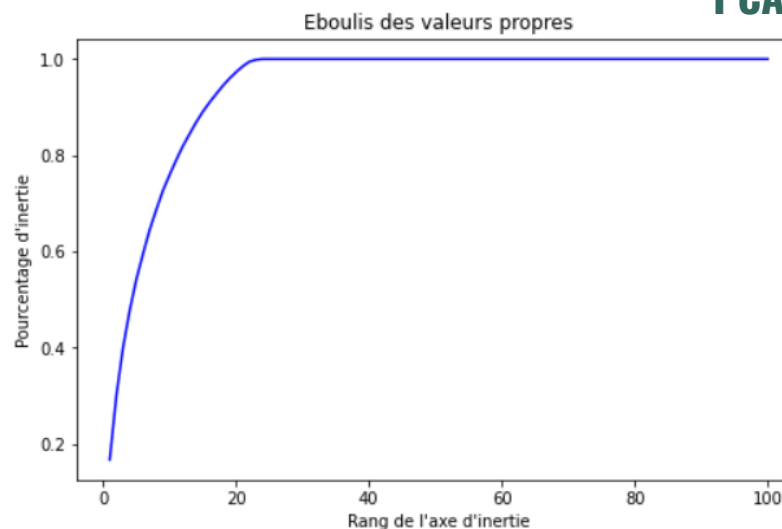
Prétraitement des images



Extraction des features VGG16



PCA



Resultats

	path	categ	pca_features_classic	vgg_features_scale	pca_features_vgg
0	5_classes_5_images/Ap...	Apple	[-24.096101642763358,...	[0.0,0.0,-0.538996857...	[3.4186094085814522,1...
1	5_classes_5_images/Ap...	Apple	[-5.750587401973896,9...	[0.0,0.0,-0.538996857...	[12.39954595401875,24...
2	5_classes_5_images/Ap...	Apple	[6.9212516552317,11.5...	[0.0,0.0,-0.538996857...	[9.836680579559298,18...
3	5_classes_5_images/Ap...	Apple	[-23.48175022493881,5...	[0.0,0.0,-0.538996857...	[5.949192329036307,17...
4	5_classes_5_images/Ap...	Apple	[3.561314541218871,11...	[0.0,0.0,-0.538996857...	[9.185863884796246,5...
5	5_classes_5_images/Av...	Avocado	[-22.149714135642597,...	[0.0,0.0,-0.538996857...	[10.683289615209658,2...



## Conclusion

- **Déploiement d'un modèle dans le cloud**
  - Via EC2 et l'Installations de tous les pré requis**
  - Via l'utilisation de Sagemaker studio**
  - data science ≠ data architecture**

## Perspectives

- **Optimiser l'utilisation de Spark / Debug UI**
  - Utiliser plus d'images**
  - Finaliser le modèle avec l'étape de clustering**



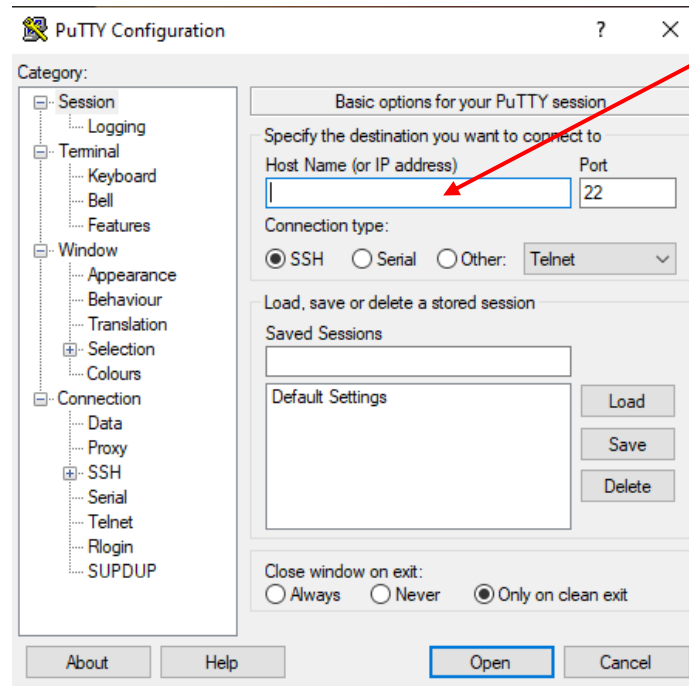
# Le code

## Login AWS EC2

### Connection via PuTTY

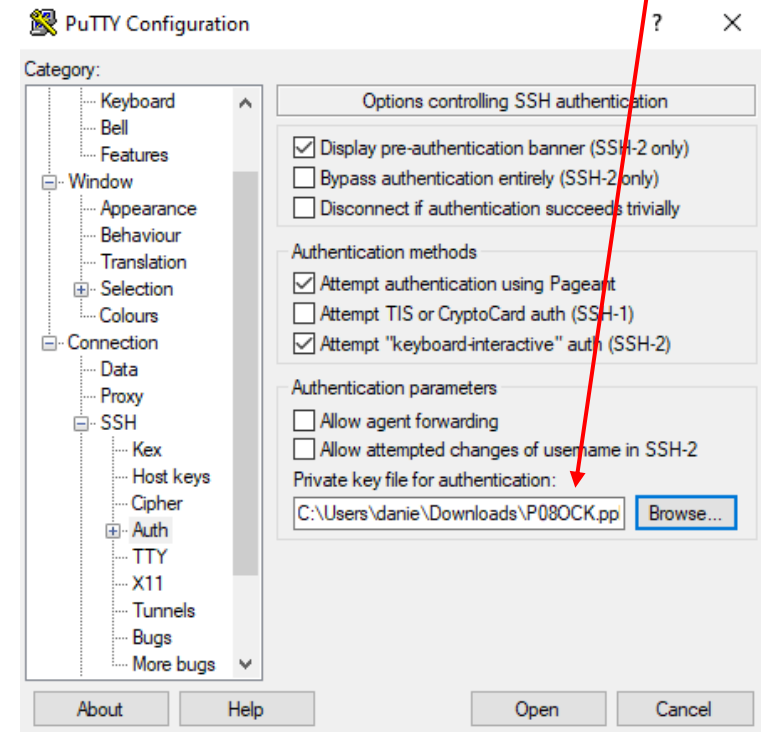


PuTTY



adresse du serveur

authentification via  
clé SSH



Installation de tous les packages nécessaires → `sudo / pip install`

# Le code











## Login AWS S3

```
1 session = boto3.Session(aws_access_key_id=access_id, aws_secret_access_key=access_key,)
```

```
1 s3 = session.resource("s3")
2 bucket = s3.Bucket("p08oc")
3 obj = bucket.Object(key="5_classes_5_images/Apple/2_100.jpg")
4 response = obj.get()
5 file_stream = response[u"Body"]
6 im = Image.open(file_stream)
7 im = im.resize((224, 224))
8 im_arr = np.asarray(im)
```

☐ p08oc EU (Paris) eu-west-3 Bucket and objects not public May 9, 2022, 18:00:10 (UTC+02:00)

☐  5\_classes\_5\_images/ Folder

<input type="checkbox"/>	Name	<input type="checkbox"/>	Name	▲	Type	▼	Last modified	▼	Size	▼
<input type="checkbox"/>	 Apple/	<input type="checkbox"/>	 2_100.jpg		jpg		May 9, 2022, 18:00:54 (UTC+02:00)		4.0 KB	
<input type="checkbox"/>	 Avocado/	<input type="checkbox"/>	 39_100.jpg		jpg		May 9, 2022, 18:00:54 (UTC+02:00)		4.0 KB	
<input type="checkbox"/>	 Banana/	<input type="checkbox"/>	 4_100.jpg		jpg		May 9, 2022, 18:00:54 (UTC+02:00)		4.1 KB	
<input type="checkbox"/>	 Carambola/	<input type="checkbox"/>	 70_100.jpg		jpg		May 9, 2022, 18:00:54 (UTC+02:00)		4.5 KB	
<input type="checkbox"/>	 Cauliflower/	<input type="checkbox"/>	 73_100.jpg		jpg		May 9, 2022, 18:00:55 (UTC+02:00)		5.7 KB	

# Le code

## Spark

```
1 from pyspark.sql import SparkSession
2
3 spark = (
4     SparkSession.builder.master("local")
12     .getOrCreate()
13 )
```

```
1 sc = spark.sparkContext
```

1 spark

**SparkSession - in-memory**  
**SparkContext**

[Spark UI](#)

**Version**

v3.2.1

**Master**

local

**AppName**

APP\_P08

## Création d'une RDD

```
1 paths = []
2 for file in bucket.objects.all():
3     if file.key.split("/")[0] == "5_classes_5_images":
4         paths.append(file.key)
```

```
1 rdd_paths = sc.parallelize(
2     paths
3 ) # Distribute a local Python collection to form an Resilient Distributed Datasets (RDD).
```

# Le code

## RDD to DataFrame

```
1 from pyspark.sql import Row
```

```
1 row_rdd_paths = rdd_paths.map(lambda x: Row(x))
```

```
1 rddCollect = row_rdd_paths.collect()
```

```
1 images_df = spark.createDataFrame(row_rdd_paths, ["path"])
```

```
1 images_df.persist(pyspark.StorageLevel.DISK_ONLY)
```

## Ajout de la catégorie et l'image sous forme d'array

# Le code

## Création des vecteurs

```
1 # conversion format vecteur dense
2 udf_vecto = udf(lambda r: Vectors.dense(r), VectorUDT())
3 images_df = images_df.withColumn("data", udf_vecto("data"))
```

## Standardization

```
1 # standardisation des données
2 standardizer = StandardScaler(
3     inputCol="data", outputCol="data_scale", withStd=True, withMean=True
4 )
5 model_std = standardizer.fit(images_df)
6 images_df = model_std.transform(images_df)
```

## Extraction de features via VGG16



**Merci de votre  
attention !**