

Projet 9

Réalisez un traitement dans un environnement Big Data sur le Cloud

Aout 2024



Introduction

CONTEXTE DU PROJET

Présentation de la start-up "Fruits!" et son ambition de préserver la biodiversité des fruits avec des robots cueilleurs intelligents.

- Préservation de la biodiversité.
- Utilisation de la technologie pour l'agriculture.
- Développement d'une application mobile pour identifier les fruits.



Problématique

Pourquoi la migration vers un environnement Big Data est essentielle pour ce projet.

- Augmentation rapide du volume de données.
- Besoin de traitement scalable.
- Importance de la conformité RGPD.

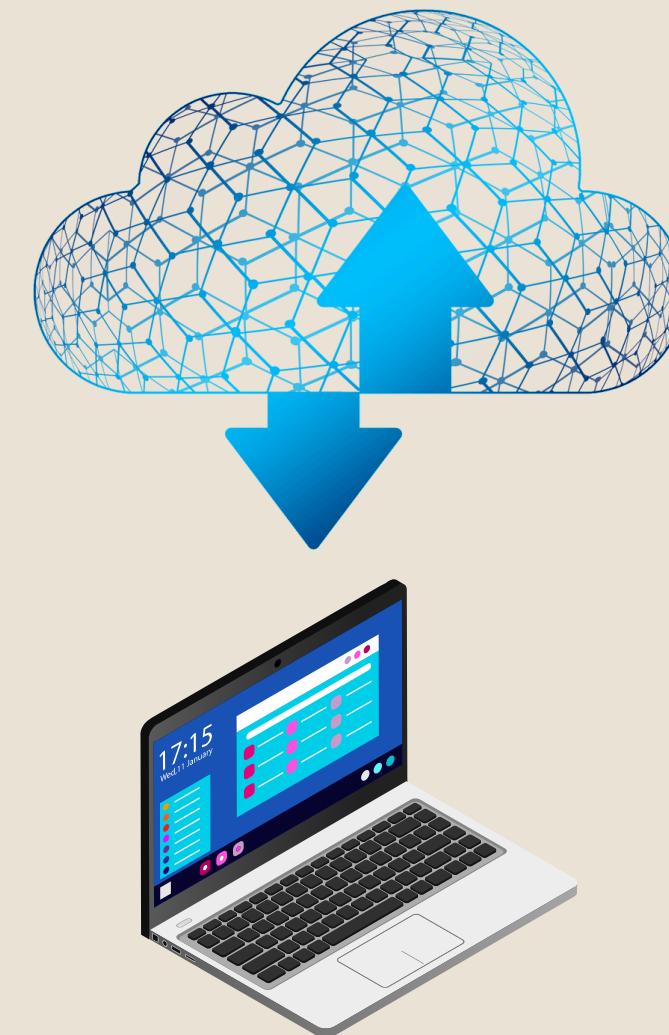


Objectifs du Projet



LES OBJECTIFS PRINCIPAUX DU PROJET ET LEUR IMPACT SUR L'ENTREPRISE

- Migrer un projet de Data Science vers un environnement Big Data.
- Mettre en place une architecture scalable.
- Assurer la conformité RGPD.



Présentation des Données

DESCRIPTION DU JEU DE DONNÉES UTILISÉ POUR
L'ENTRAÎNEMENT ET LE TRAITEMENT

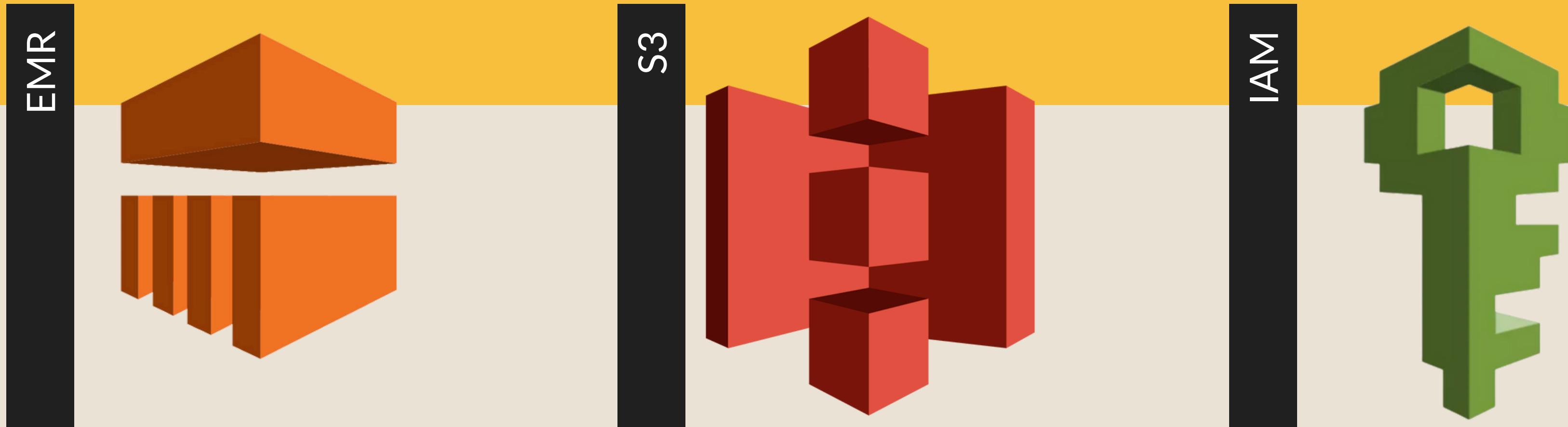
- **Source** : Images de fruits stockées sur AWS S3.
- **Contenu** : Images avec labels associés.
- **Volume attendu** : Croissance rapide post-livraison.

*Images capturées avec une caméra Logitech C920 sur pivot
Algorithme utilisé pour isoler les fruits du fond, fond blanc
Données étiquetées stockées sur AWS S3 pour une croissance rapide.*



Architecture Big Data

PRÉSENTATION DE L'ARCHITECTURE CHOISIE POUR LE PROJET



Utilisation d'AWS EMR pour le traitement distribué

EC2 1 primaire, 2 principal
m5.xlarge, facturation on-demand

Stockage des données sur AWS S3

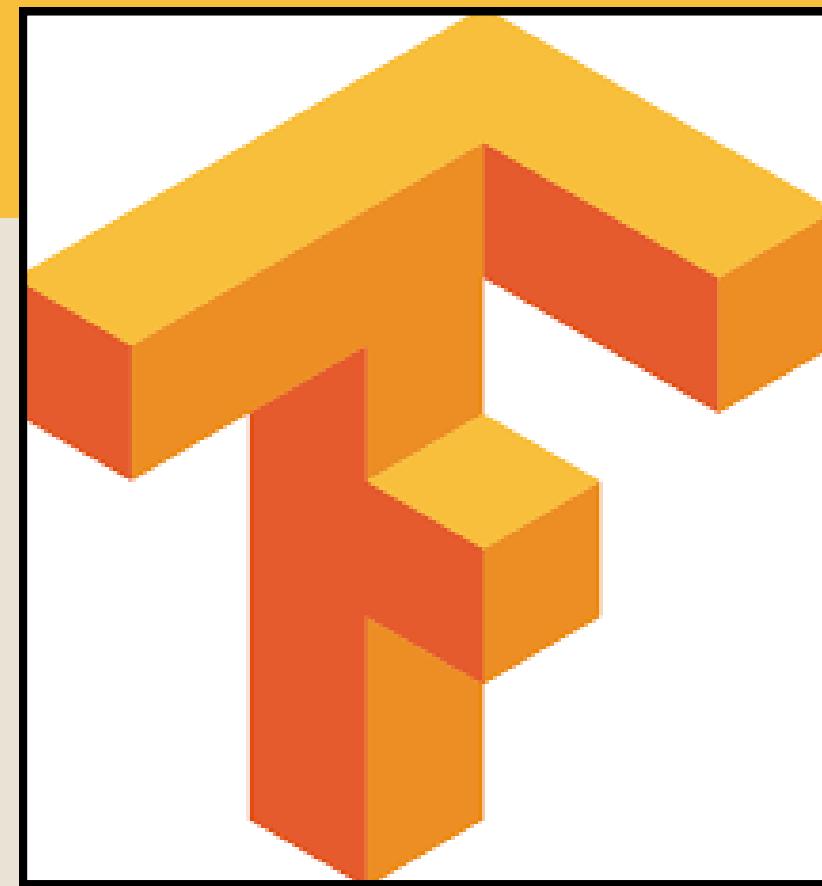
Gestion des identités et accès avec IAM

Outils utilisés

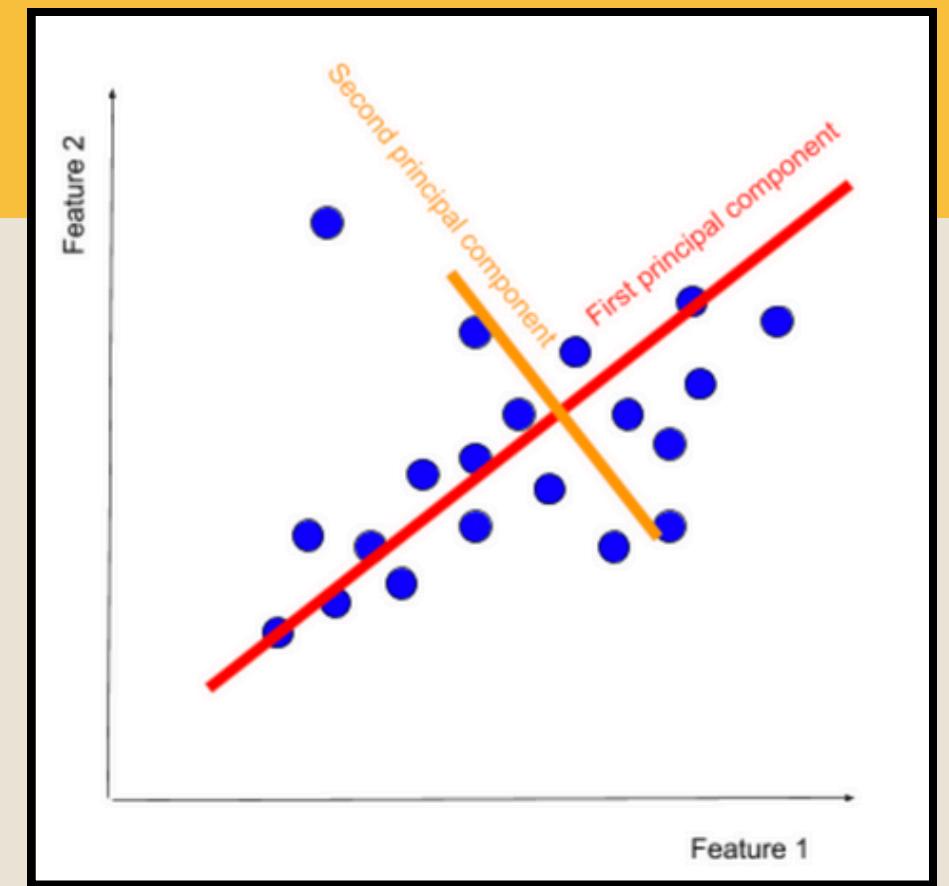
LISTE DES OUTILS PRINCIPAUX UTILISÉS DANS LE PROJET



PySpark pour le traitement distribué



TensorFlow pour l'extraction des features



PCA pour la réduction de dimension

Chargement des Données

EXPLICATION DU PROCESSUS DE CHARGEMENT DES DONNÉES DEPUIS S3

- Utilisation de spark.read.format("binaryFile").
- Extraction des labels depuis les chemins de fichiers.
- Vérification et affichage des premières lignes.

```
images = spark.read.format("binaryFile") \  
.option("pathGlobFilter", "*.jpg") \  
.option("recursiveFileLookup", "true") \  
.load(PATH_Data)
```

```
images.show(5)
```

```
+-----+-----+-----+  
| path | modificationTime | length | content |  
+-----+-----+-----+  
| s3://p8-data-rogu... | 2024-08-09 07:30:26 | 7353 | [FF D8 FF E0 00 1... |  
| s3://p8-data-rogu... | 2024-08-09 07:30:26 | 7350 | [FF D8 FF E0 00 1... |  
| s3://p8-data-rogu... | 2024-08-09 07:30:26 | 7349 | [FF D8 FF E0 00 1... |  
| s3://p8-data-rogu... | 2024-08-09 07:30:26 | 7348 | [FF D8 FF E0 00 1... |  
| s3://p8-data-rogu... | 2024-08-09 07:30:27 | 7328 | [FF D8 FF E0 00 1... |  
+-----+-----+-----+  
only showing top 5 rows
```

```
images = images.withColumn('label', element_at(split(images['path'], '/'), -2))  
print(images.printSchema())  
print(images.select('path', 'label').show(5, False))
```

```
root  
| -- path: string (nullable = true)  
| -- modificationTime: timestamp (nullable = true)  
| -- length: long (nullable = true)  
| -- content: binary (nullable = true)  
| -- label: string (nullable = true)
```

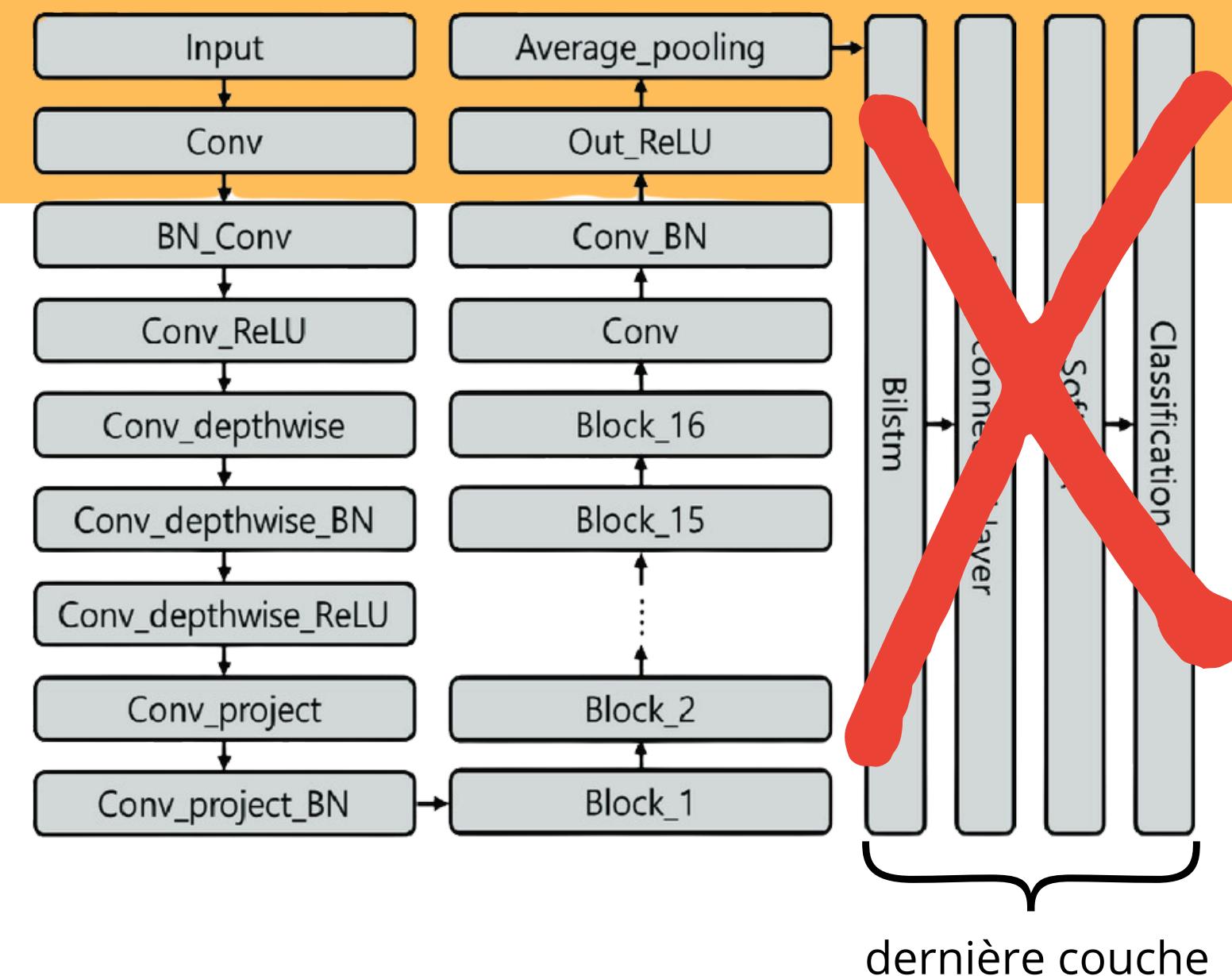
```
None
```

```
+-----+-----+  
| path | label |  
+-----+-----+  
| s3://p8-data-rogues-alex/Test/Watermelon/r_106_100.jpg | Watermelon |  
| s3://p8-data-rogues-alex/Test/Watermelon/r_109_100.jpg | Watermelon |  
| s3://p8-data-rogues-alex/Test/Watermelon/r_108_100.jpg | Watermelon |  
| s3://p8-data-rogues-alex/Test/Watermelon/r_107_100.jpg | Watermelon |  
| s3://p8-data-rogues-alex/Test/Watermelon/r_95_100.jpg | Watermelon |  
+-----+-----+  
only showing top 5 rows
```

Modélisation avec MobileNetV2

PRÉSENTATION DU MODÈLE MOBILENETV2 UTILISÉ POUR L'EXTRACTION DES FEATURES

- Pré-entraînement du modèle sur ImageNet.
- Extraction des features (dim 1280) en enlevant la dernière couche.
- Diffusion des poids sur le cluster Spark.



```
def preprocess(content):
    """
    Preprocesses raw image bytes for prediction.
    """
    img = Image.open(io.BytesIO(content)).resize([224, 224])
    arr = img_to_array(img)
    return preprocess_input(arr)
```

Prétraitement des Images

DESCRIPTION DU PROCESSUS DE PRÉTRAITEMENT
DES IMAGES POUR LE MODÈLE

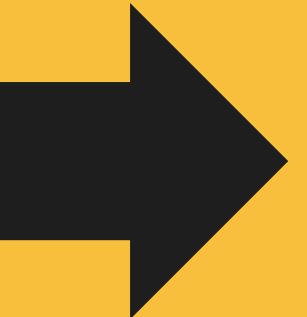
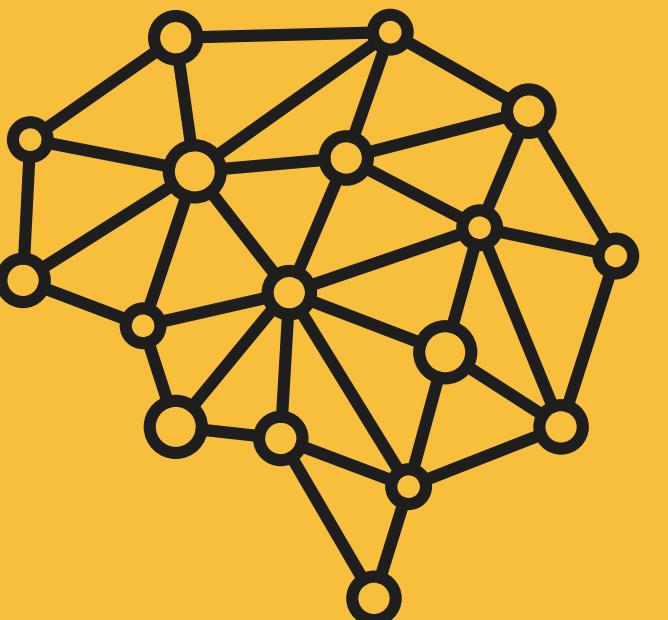
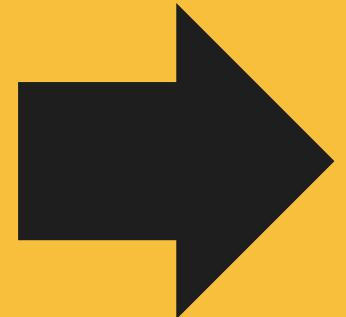


- Redimensionnement à 224x224 pixels.
- Conversion en tableau numpy.
- Application de preprocess_input (conversion, normalisation)

Extraction des Features

EXPLICATION DE L'EXTRACTION DES FEATURES VIA LE MODÈLE MOBILENETV2

- Application du modèle sur des batchs d'images.
- Conversion des sorties du modèle en vecteurs.
- Stockage des features pour l'étape suivante.



[12, -5, 0.5, ... 6.2]

Standardisation des Données

IMPORTANCE DE LA STANDARDISATION AVANT L'APPLICATION DE LA PCA

- Utilisation de StandardScaler pour centrer et réduire les features.
- Préparation des données pour la réduction de dimension.

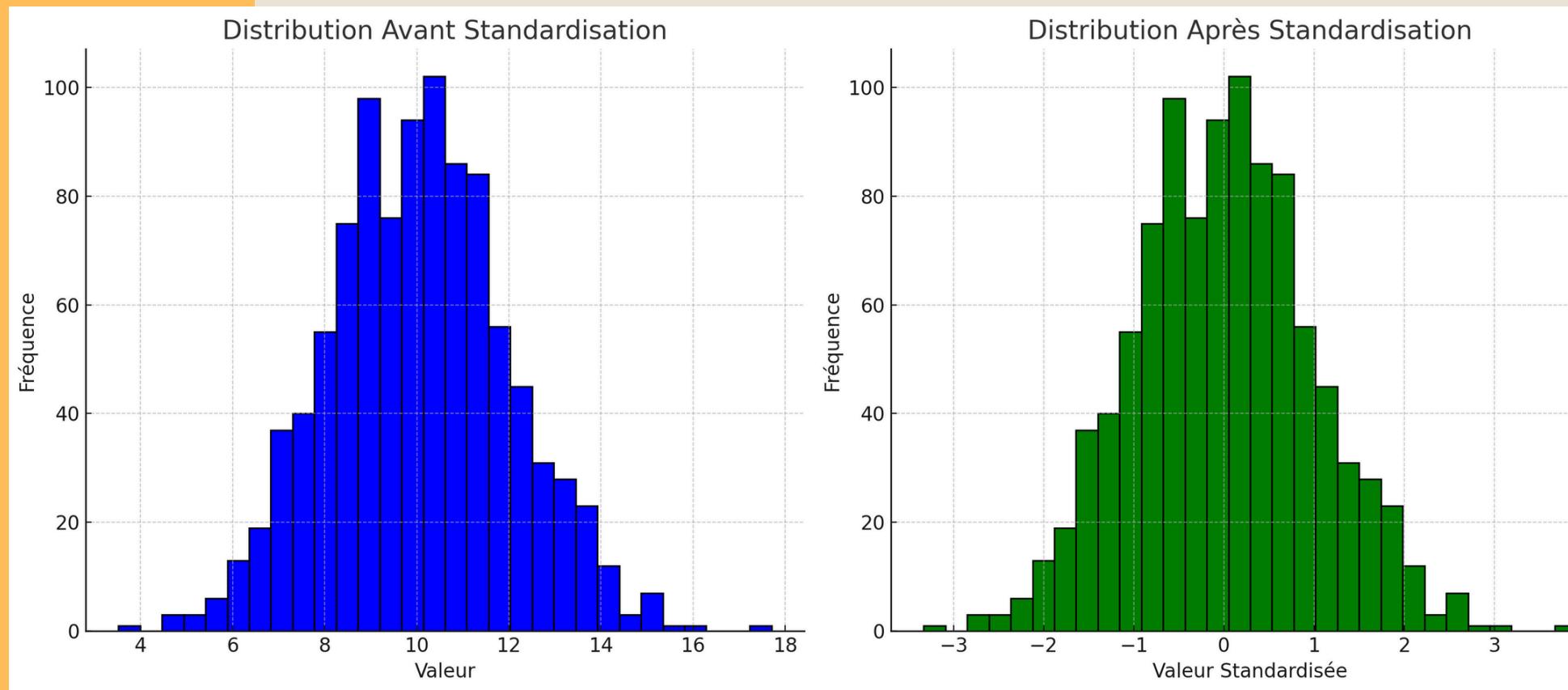
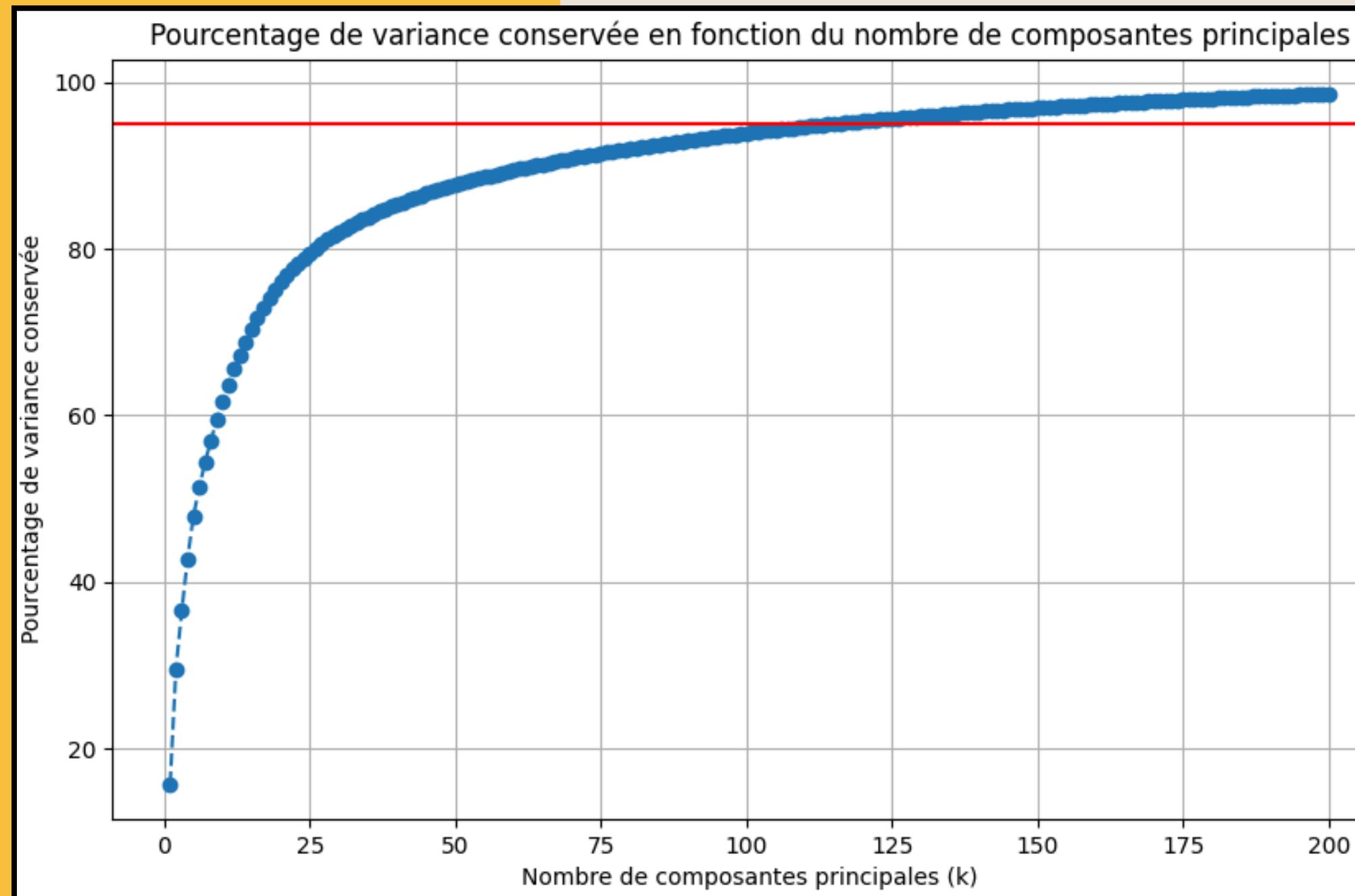


image d'illustration

Réduction de Dimension par PCA



APPLICATION DE LA PCA POUR RÉDUIRE LA DIMENSIONNALITÉ TOUT EN CONSERVANT 95% DE LA VARIANCE



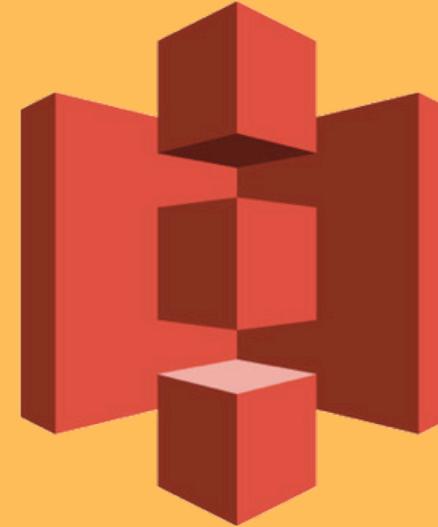
graph issu du notebook executé en local

- Calcul du nombre optimal de composantes.
- Application de la PCA avec ce nombre de composantes.
- Conversion des résultats en un DataFrame Spark.

Stockage des Résultats



SAUVEGARDE DES RÉSULTATS TRANSFORMÉS SUR AWS S3 POUR UNE UTILISATION FUTURE



Amazon S3



- Format Parquet choisi pour l'efficacité.
- Chemin de stockage sur S3.
- Chargement des résultats pour validation

Conformité RGPD

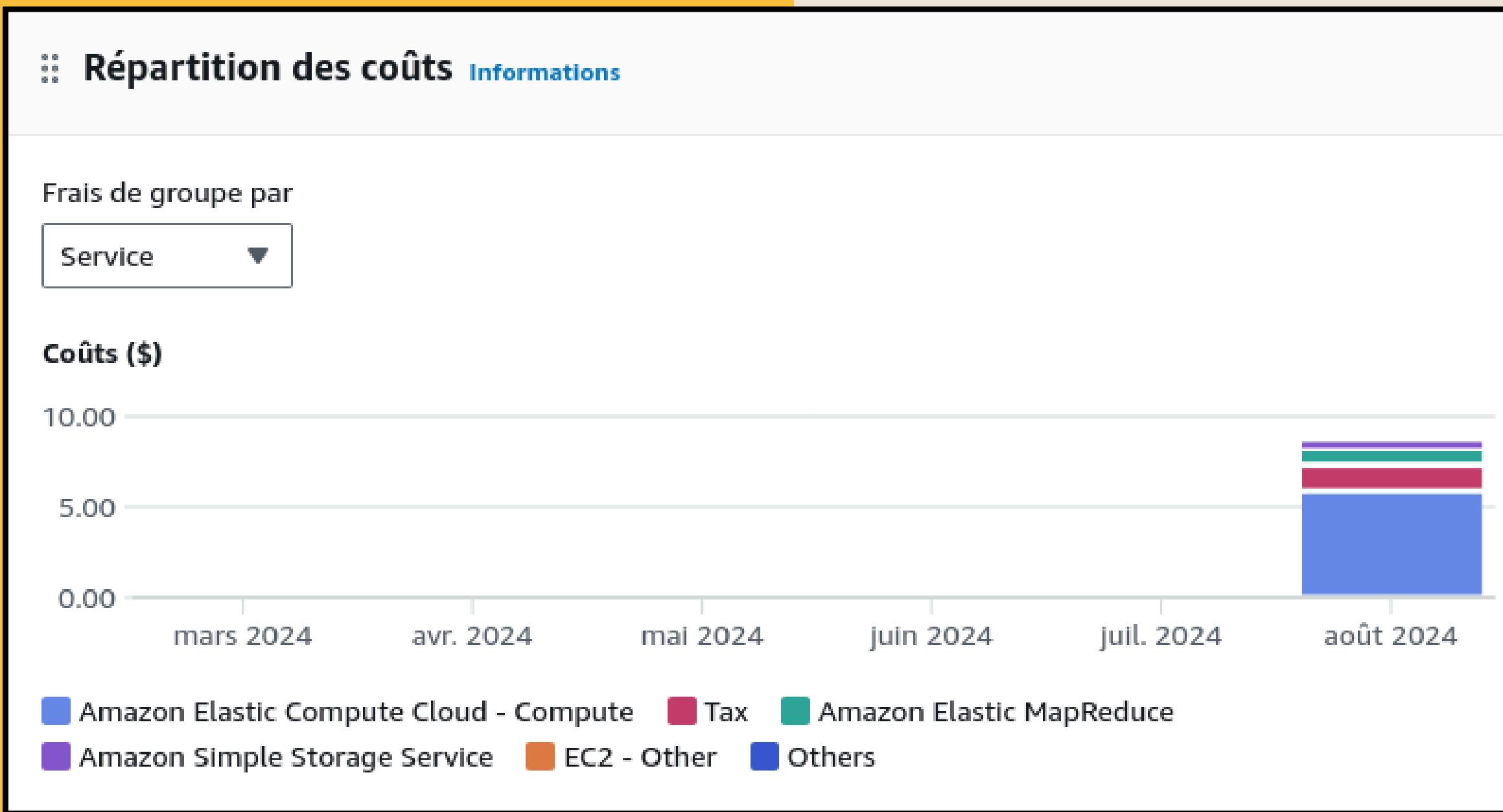
MESURES PRISES POUR GARANTIR LA CONFORMITÉ RGPD DANS LE PROJET



- Utilisation de serveurs AWS situés en Europe.
- Paramétrage des services pour respecter les normes.

Gestion des Coûts

STRATÉGIES POUR MINIMISER LES COÛTS LIÉS À L'UTILISATION DU CLOUD



- Arrêt des instances EMR après utilisation.
- Optimisation des partitions Spark pour les calculs.
- Usage de S3 pour le stockage, évitant des coûts de stockage local

Validation et Résultats

VALIDATION DES RÉSULTATS OBTENUS APRÈS LE TRAITEMENT DE LA CHAÎNE

- Vérification de la réduction de dimension.
- Comparaison avec les résultats attendus.
- Conclusion sur l'efficacité du traitement.

Vérification de la dimension du vecteur de caractéristiques (k attendus)

```
# Vérifier le nombre de dimensions des vecteurs dans 'pca_features'
num_dimensions = len(df['pca_features'][0])

result_message = (f"Les vecteurs contiennent {num_dimensions} dimensions.\n"
                  f"Le nombre de dimensions attendu est {k}.\n"
                  f"{('Tout est Ok !' if num_dimensions == k else 'Il y a un bug.'})")

print(result_message)
```

Les vecteurs contiennent 557 dimensions.
Le nombre de dimensions attendu est 557.
Tout est Ok !

Synthèse et Conclusion

RÉSUMÉ DES ACCOMPLISSEMENTS DU PROJET ET CONCLUSION SUR LA MISE EN PLACE DE L'ARCHITECTURE BIG DATA



- Migration réussie vers le cloud.
- Traitement scalable prêt pour de futurs volumes de données.
- Conformité avec les réglementations en vigueur.

Perspectives Futures

DISCUSSIONS SUR LES PROCHAINES ÉTAPES ET AMÉLIORATIONS POSSIBLES

- Extension du traitement à d'autres types de données.
- Optimisation continue des coûts.
- Déploiement de l'application mobile à grande échelle.





Merci