

# Document Technique

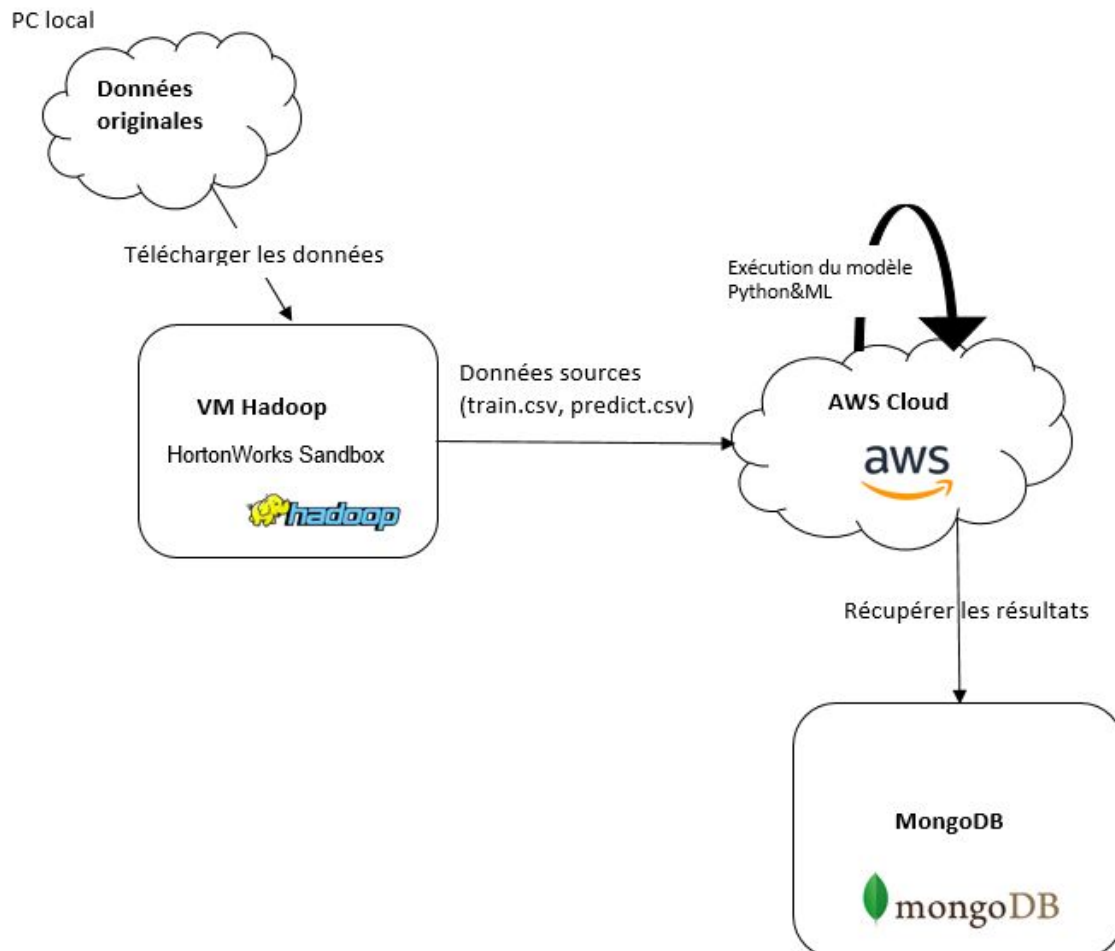
## Projet Big Data

- Réalisé par:
  - Manal MOUAYANI
  - Qi LI
  - Lucas MARIE

# INTRODUCTION

**Objectif du projet :** Dans ce projet, le but principal est d'aider une compagnie d'assurance à développer un modèle prédictif qui lui permettra cette dernière de connaître automatiquement les risques liés à l'ouverture d'un nouveau contrat d'un client. Pour ceci, on se base sur des anciens clients, il faut qu'on détermine le risque associé pour chacun des clients (ce dernier doit être une valeur entre 1 et 8). Pour ce faire, nous devons travailler sur un environnement Hadoop pour stocker des données massives et entraîner les modèles de Machine Learning avec Python dans l'environnement AWS Cloud. Ensuite, il nous faudra télécharger le résultat de la prédiction dans une base de données NoSql (MongoDB).

Voici un schéma qui résume les étapes nécessaires pour la réalisation de ce projet :



Dans la réalisation de ce projet on est passé par les étapes suivantes:

## Etape 0:

- On a commencé par installer la Vm de cloudera ou il y'a hadoop et pour ensuite exécuter les commandes suivantes:

### 1-Pour la Création d'un dossier en HDFS:

```
sudo -u hdfs hadoop dfs -chown admin:hdfs /input
```

### 2-Pour la modification des autorisations de fichier dans hdfs

```
sudo -u hdfs Hadoop fs -chmod 777 /input
```

### 3-Pour importer les données du PC Local vers la VM Hadoop

- **Pour train:** scp -P 2222 train.csv root@localhost:
- **Pour predict:** scp -P 2222 predict.csv root@localhost:

### 4-Pour importer le fichier du VM Hadoop HDFS

- **Pour train:** hadoop fs -put train.csv/input
- **Pour predict:** hadoop fs -put predict.csv/input

## Etape 1:

-Voici la commande qui permet de récupérer les données de HDFS vers la VM Local

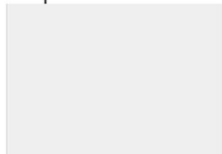
- **Pour train:** Hadoop fs -get /input /train.csv
- **Pour predict:** Hadoop fs -get /input /predict.csv


## Etape 2:

-Pour cette partie, on a commencé par créer une vm ubuntu dans le Cloud AWS

Step 1: Choose an Amazon Machine Image (AMI)

[Cancel and Exit](#)





**Ubuntu Server 18.04 LTS (HVM), SSD Volume Type** - ami-04b9e92b5572fa0d1 (64-bit x86) / ami-0bba96c31d87e65d9 (64-bit Arm)

Free tier eligible

Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Root device type: ebs

Virtualization type: hvm

ENA Enabled: Yes

Select

☒ 64-bit (x86)

☐ 64-bit (Arm)

Launch Instance

Connect

Actions

Filter by tags and attributes or search by keyword

1 to 5 of 5

<input type="checkbox"/>	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
<input type="checkbox"/>		i-0106c43af67cad103	t2.micro	us-east-1a	terminated		No Data	
<input type="checkbox"/>		i-0e0ec07166a1dc069	t2.micro	us-east-1a	terminated		None	
<input type="checkbox"/>		i-06a63dc45f36fb473	t2.micro	us-east-1c	terminated		None	
<input type="checkbox"/>		i-0dc0a53b794cb5928	t2.micro	us-east-1c	stopped		None	
<input checked="" type="checkbox"/>		i-0f21f372d942a6826	t2.micro	us-east-1c	running	2/2 checks ...	None	ec2-3-86-149

Après, on a utilisé cette commande pour importer le fichier du VM Local vers l'instance Ubuntu qu'on a créé avant :

- **Pour train:** `scp -i hadoop.pem train.csv ec2-user@ec18-234-56-202.compute-1.amazonaws.com:`
- **Pour predict:** `scp -i hadoop.pem predict.csv ec2-user@ec18-234-56-202.compute-1.amazonaws.com:`

## Etape 3:

### Traitement des données

Pour développer un modèle prédictif, la première étape consiste à améliorer la précision du modèle en traitant les données.

- 1、 Tout d'abord, nous avons décidé d'utiliser LinearSVC comme modèle de base.
- 2、 Ensuite, nous utilisons la méthode des variables de contrôle et augmentons le traitement des données, puis observons si la précision du modèle s'améliore.
- 3、 Afin d'évaluer plus précisément la précision du modèle, nous utilisons la validation croisée pour calculer la précision moyenne du modèle.
- 4、 Au même temps, nous utilisons le principe du rasoir d'Occam, c'est-à-dire avec la même précision, le modèle le plus simple est le meilleur.
- 5、 Nous avons comparé différents modèles et décidé de choisir GBDT comme modèle final. Ensuite, nous utilisons la méthode de recherche de grille pour déterminer les paramètres optimaux du modèle et obtenir une plus grande précision.
- 6、 Enfin, nous formons un modèle sur toutes les données, puis utilisons ce modèle pour prédire de nouvelles données et produire le résultat final.



## Étapes de formation du modèle

1. Traitement des données en double
2. Resampling
3. Traitement de la valeur manquante
4. Création de nouvelles variables
5. Standardisation et Numérisation des données
6. PCA dimensionality reduction
7. Sélection des caractéristiques
8. Partitionnement de l'ensemble des données
9. Formation et comparaison des modèles
10. Grid Search
11. Formation du modèle sur toutes les données
12. Génération des résultats de prédiction

Voici le lien de github qui contient plus d'informations sur la partie d'analyse et traitement des données.

[https://github.com/LickyQi/Big\\_Data\\_Project](https://github.com/LickyQi/Big_Data_Project)

## Etape 4:

-Après qu'on a développé notre modèle et qu'on a amélioré le score, on a exécuté ce dernier dans la VM de AWS . Ensuite, on a concaténé chaque ligne de predict.csv avec la prédiction de notre modèle dans un fichier resultats.csv et on l'a enregistré dans la VM avec la commande suivante:

```
scp -i hadoop.pem ubuntu@ec18-234-56-202.compute-1.amazonaws.com:resultats.csv ./
```

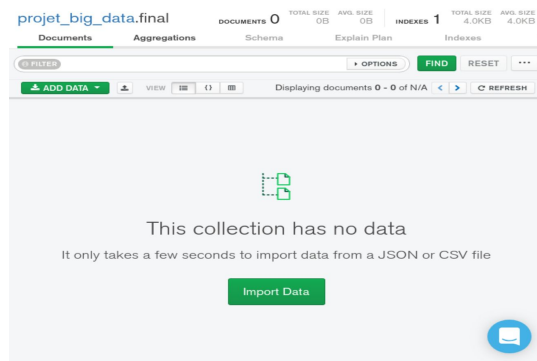
## Etape 5:

Dans cette étape, nous récupérons un fichier issu de la concaténation du fichier predict.csv et des résultats obtenus dans notre modèle en local. A partir de ce fichier de sauvegarde, nous pouvons créer une nouvelle collection dans une base de données MongoDB à l'aide de l'interface MongoDB Compass qui permet de créer la collection sans passer par un fichier Json et de visualiser les statistiques concernant les données présentes dans notre base de données.

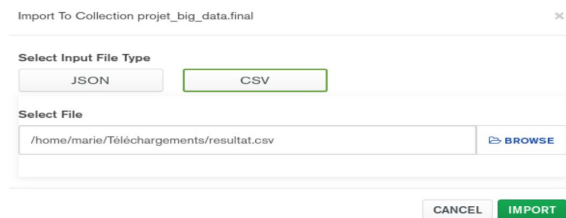
Ensuite, dans MongoDB Compass nous pouvons faire des agrégations en utilisant l'onglet dédié qui nous permet de visualiser des données plus précises en fonction de ce qu'on cherche comme élément. Nous pouvons par exemple regarder l'impact de l'âge ou du sexe d'une personne sur le risque qui sera donnée pour cette personne.

MongoDB Compass : <https://www.mongodb.com/products/compass>

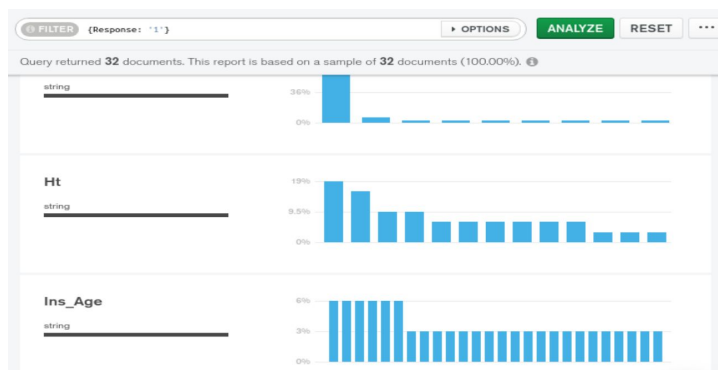
Après avoir créer une nouvelle collection vide nous arrivons à cette fenêtre :



Pour importer les données, nous avons le choix entre un fichier Json et un fichier CSV. Dans notre cas nous utilisons un CSV :



MongoDB permet de mettre en place des filtres permettant par exemple de visualiser une données en fonction de condition :



Dans ce cas ci nous affichons les caractéristiques des personnes pour lesquelles le fait de donner une assurance sera peu risqué.