Master bio-informatique et modélisation des systèmes complexes appliquée à la santé

**Partie 2 :**

**Projet Big data**

Réalisé par : -Yasine lhamiani

-BENSOUDANE Osama

Table de matières

[1 Téléchargement d'Anaconda et installation de PySpark 3](#_Toc91717595)

[2 Initialisation de SparkSession 4](#_Toc91717596)

[3 Création de blocs de données 4](#_Toc91717597)

[4 Requêtes 4](#_Toc91717598)

[4.1 Telecharger Dataset 4](#_Toc91717599)

[6](#_Toc91717600)

[4.2 Création des tables sql : 6](#_Toc91717601)

[4.3 Visualisation : 6](#_Toc91717602)

[4.3.1 Consultez la liste totale des étudiants (e-mails) du Cours de génie logiciel, classés par ordre alphabétique 7](#_Toc91717603)

[4.3.2 Vérifiez le nombre de présences enregistrées par élève; en outre, la date et l'heure de la première participation et la date et l'heure de la dernière participation 7](#_Toc91717604)

[4.3.3 Vérifiez la liste des étudiants qui ont assisté à 2 présentations ou moins (une). 8](#_Toc91717605)

[Consultez la liste des étudiants qui n'ont assisté à aucune présentation. 8](#_Toc91717606)

[4.3.4 Vérifiez les membres de chaque équipe à laquelle ils appartiennent. 8](#_Toc91717607)

[4.3.5 Vérifiez le nombre de participants par présentation, sans tenir compte des participants qui appartiennent à l'équipe qui a fait la présentation. 9](#_Toc91717608)

[Découvrez quels membres ont évalué leur propre équipe. Ces évaluations ne seront pas valides, en tant que membre ne peut pas évaluer sa propre équipe 9](#_Toc91717609)

[5 Projet Machine learning avec PySpark MLlib 10](#_Toc91717610)

[5.1 Petit rappel de Machine Learning 10](#_Toc91717611)

[5.2 Régression logistique avec Spark MLlib : 11](#_Toc91717612)

[5.2.1 Quand et pourquoi utiliser PySpark MLlib 11](#_Toc91717613)

[5.2.2 Les types de données spécifiques à MLlib : 12](#_Toc91717614)

[5.2.3 Chargement du jeu de données et sélection des entités 12](#_Toc91717615)

[5.2.4 Divisez l'ensemble de données en un train et un ensemble de test 14](#_Toc91717616)

[5.2.5 Définir les types de données 14](#_Toc91717617)

[5.2.6 One hot encoding à l'aide de PySpark MLlib 15](#_Toc91717618)

[One hot encoding 15](#_Toc91717619)

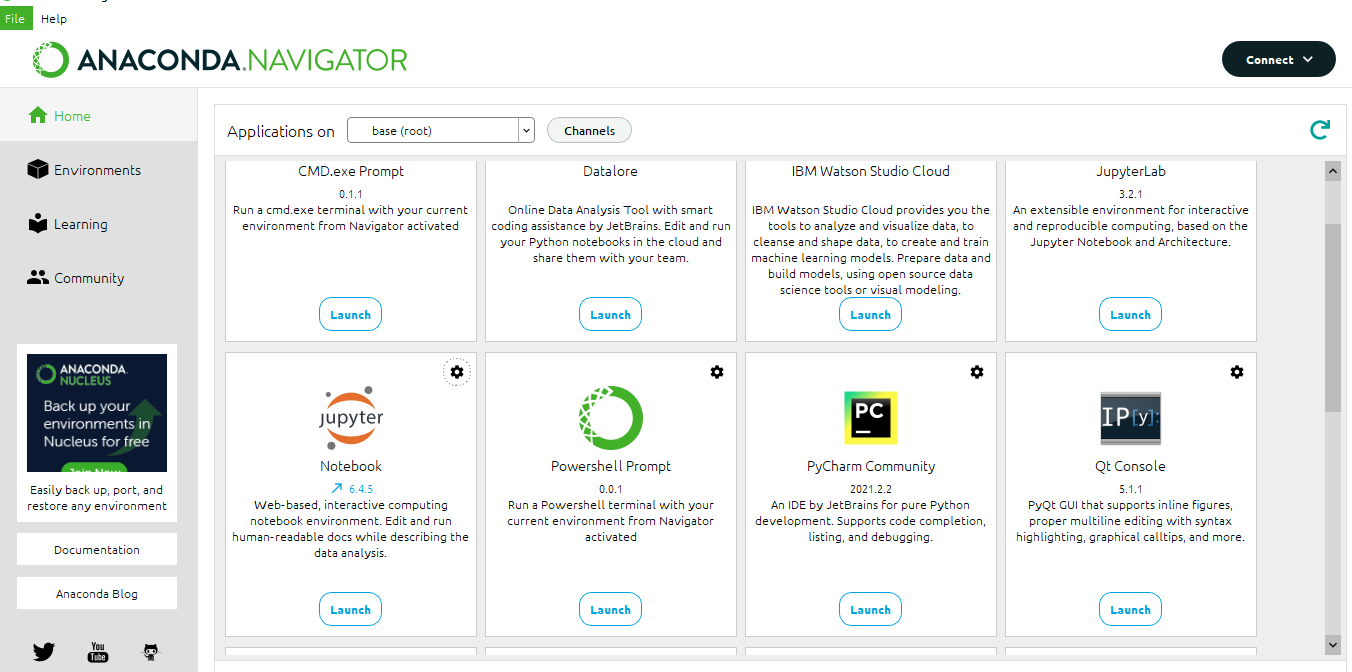
[5.2.7 Comprendre VectorAssembler : combiner les valeurs de caractéristiques d'entrée dans un seul vecteur 15](#_Toc91717620)

[5.2.8 le modèle de régression logistique et évaluation des résultats 16](#_Toc91717621)

[6 Références 17](#_Toc91717622)

# Téléchargement d'Anaconda et installation de PySpark

Une fois l'installation terminée, la page d'accueil d'Anaconda Navigator s'ouvrira. Pour utiliser Python, il suffit de cliquer sur le bouton "Lancer" du module "Notebook".



Pour pouvoir utiliser Spark via Anaconda, les étapes d'installation du package suivantes doivent être suivies

*Anaconda Prompt terminal*

*conda install pyspark*

*conda install*[*pyarrow*](https://pypi.org/project/pyarrow/)

Une fois l'installation des packages PySpark et PyArrow terminée, on ferme simplement le terminal et on revenue à Jupyter Notebook et importez les packages requis en haut de notre code.

# Initialisation de SparkSession

Tout d'abord, une session Spark doit être initialisée. Avec l'aide de SparkSession, DataFrame peut être créé et enregistré en tant que tables. De plus, les tables SQL sont exécutées, les tables peuvent être mises en cache et les fichiers au format CSV peuvent être lus.

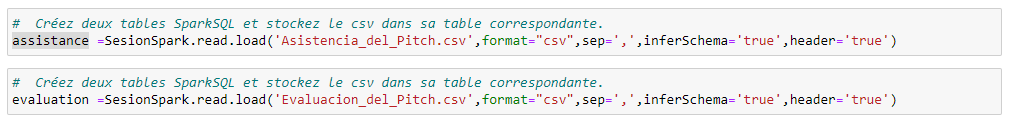
# Création de blocs de données

. Un DataFrame peut être accepté comme une collection distribuée et tabulée de colonnes intitulées qui est similaire à une table dans une base de données relationnelle. Dans ce rapport, nous utiliserons les opérations DataFrame sur l'API PySpark tout en travaillant avec des ensembles de données.

# Requêtes

## Telecharger Dataset

**À partir de sources de données Spark**

**Créez deux tables SparkSQL et stockez le csv dans sa table correspondante**

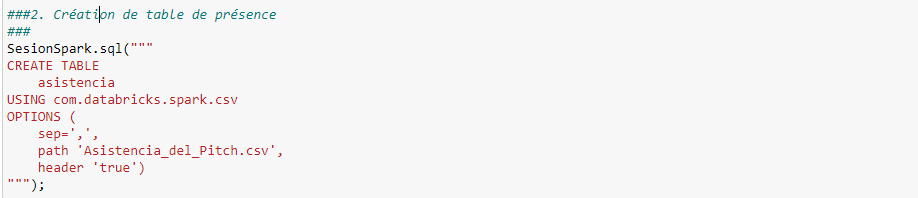
**Assistance :**

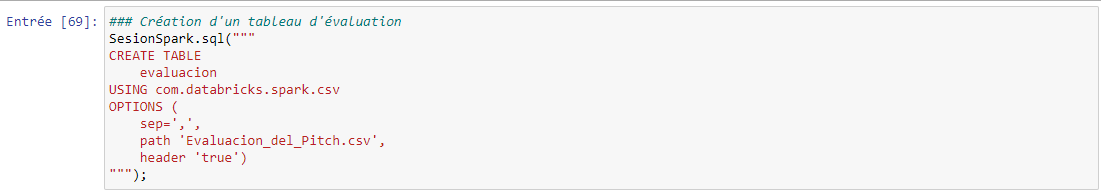
****

**Evaluation :**

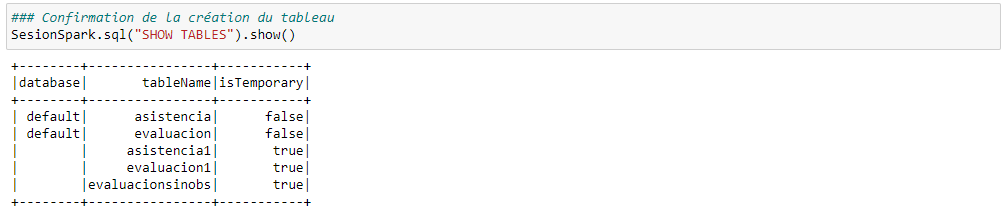
# C:\Users\Yousef\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Capture d’écran 2021-12-29 224212.png

## Création des tables sql :

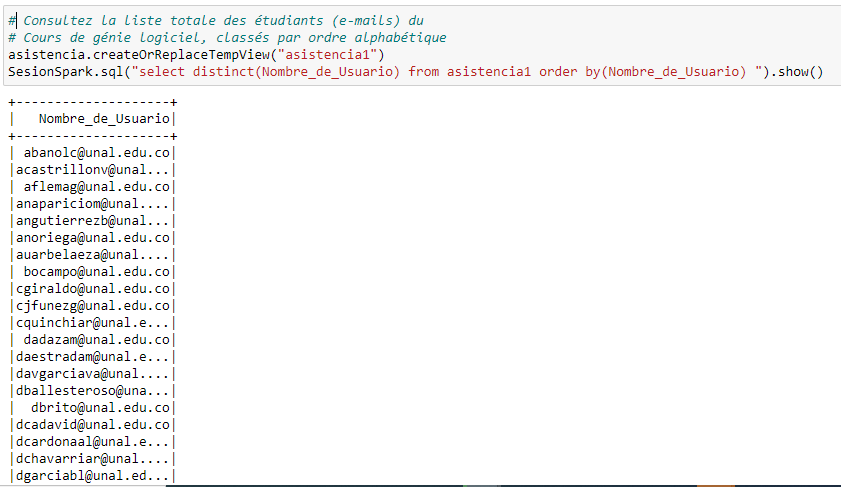
****



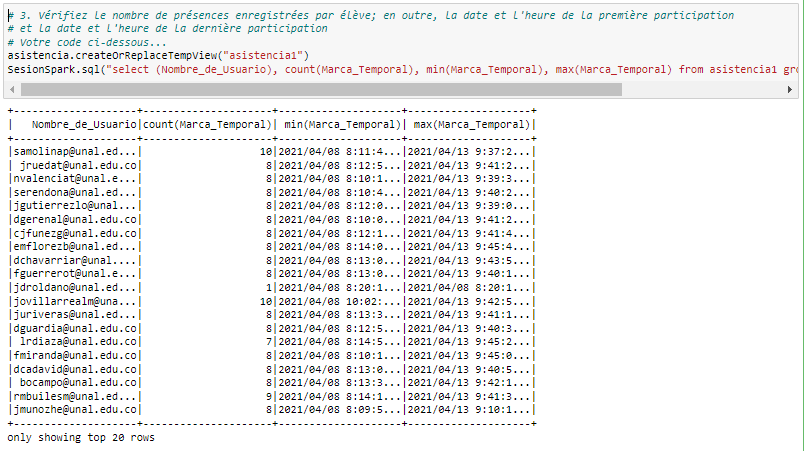
## Visualisation :



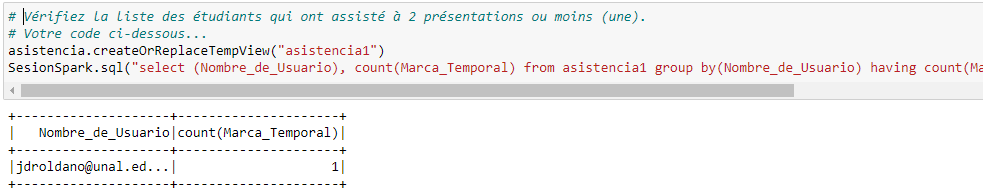
### Consultez la liste totale des étudiants (e-mails) du Cours de génie logiciel, classés par ordre alphabétique

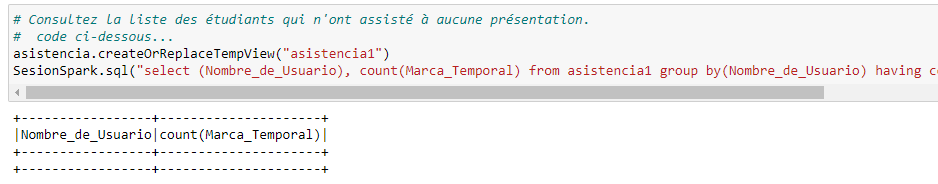


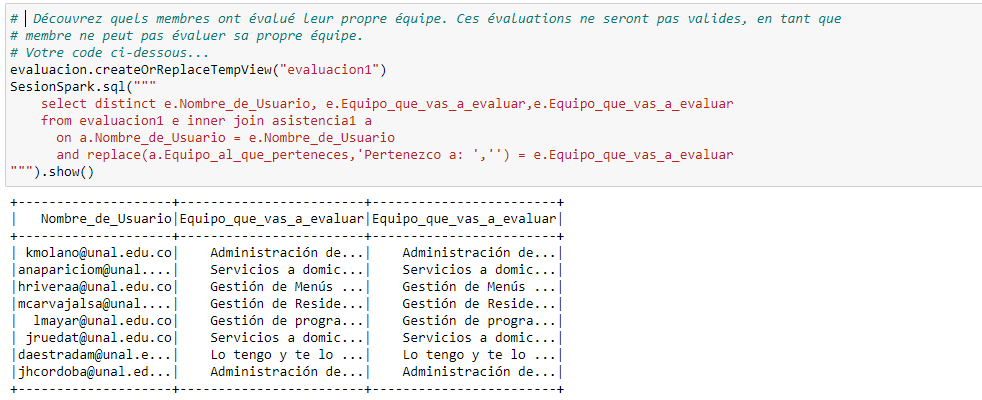
### Vérifiez le nombre de présences enregistrées par élève; en outre, la date et l'heure de la première participation et la date et l'heure de la dernière participation



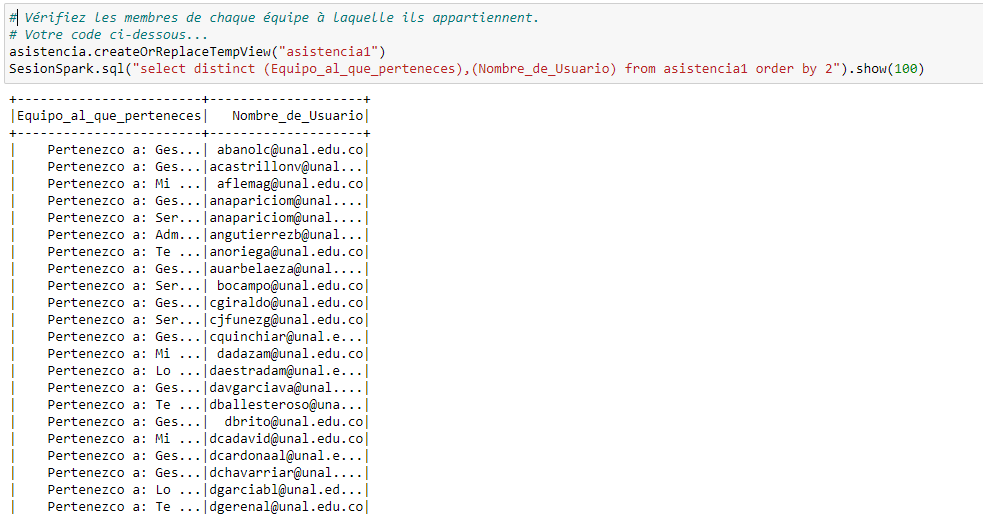
### Vérifiez la liste des étudiants qui ont assisté à 2 présentations ou moins (une).



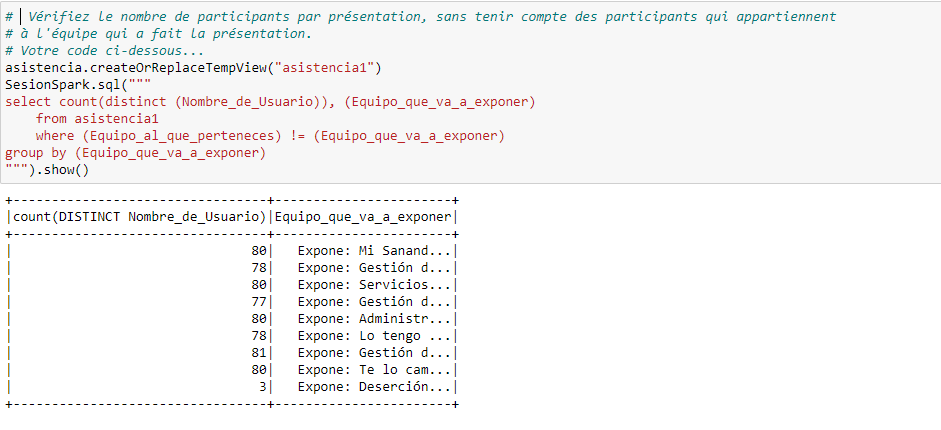
Consultez la liste des étudiants qui n'ont assisté à aucune présentation. 



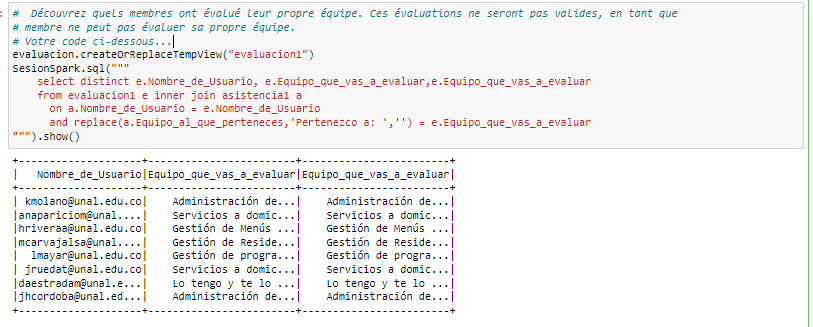
### Vérifiez les membres de chaque équipe à laquelle ils appartiennent.



### Vérifiez le nombre de participants par présentation, sans tenir compte des participants qui appartiennent à l'équipe qui a fait la présentation.



Découvrez quels membres ont évalué leur propre équipe. Ces évaluations ne seront pas valides, en tant que membre ne peut pas évaluer sa propre équipe.



# Projet Machine learning avec PySpark MLlib

## Petit rappel de Machine Learning

Pour employer de manière efficace MLlib, il est bien entendu nécessaire d’avoir quelques bases en Machine Learning. Le Machine Learning est une branche de l’Intelligence Artificielle qui permet l’analyse et la construction d’algorithmes capables d’apprendre à partir de données d’entrée. On peut distinguer deux catégories principales d’algorithmes: de type supervisé ou non supervisé.

En apprentissage supervisé, on dispose d’un dataset composé de caractéristiques (features) associées à des labels (target). L’objectif est de construire un estimateur capable de prédire le label d’un objet à partir de ses features. L’algorithme apprend alors à partir de données dont on connait le label et est ensuite capable de faire de la prédiction sur de nouvelles données dont on ne connaît pas le label. On distingue les algorithmes de classification, pour lesquels le label à prédire est une classe (prédire un mail comme étant spam / non spam), de ceux de régression, pour lequels il faut prédire une variable continue (prédire la taille d’une personne en fonction de son poids et de son âge par exemple). En apprentissage non-supervisé, on ne dispose pas de label pour nos données. L’objectif est alors de trouver des similarités entre les objets observés, pour les regrouper au sein de clusters.

On peut de plus citer les algorithmes dédiés aux systèmes de recommandation (collaborative filtering), ainsi que l’apprentissage par renforcement, qui regroupe un ensemble d’algorithmes qui vont faire leurs prédictions en apprenant de leurs erreurs au fur et à mesure, et s’adapteront aux éventuels changements.

## Régression logistique avec Spark MLlib :

Dans ce projet pratique d'apprentissage automatique, nous passerons en revue tout ce que vous devez savoir pour créer un modèle d'apprentissage automatique (régression logistique dans cet exemple) en utilisant Spark MLlib.

MLlib est la bibliothèque d'apprentissage automatique évolutive d'Apache Spark. Son objectif est de rendre l'apprentissage machine pratique évolutif et facile. MLlib contient de nombreux algorithmes et utilitaires, notamment : Classification : Régression logistique, Régression bayésienne naïve : Régression linéaire généralisée, Arbres de décision, Forêts aléatoires, Arbres améliorés de gradient, Clustering : K-mean, mélanges gaussiens (GMM) et bien d'autres.

**Apprentissage automatique avec PySpark MLlib**

### Quand et pourquoi utiliser PySpark MLlib

MLlib est la librairie de Machine Learning de Spark. Tous les algorithmes de cette librairie sont conçus de manière à être optimisés pour le calcul en parallèle sur un cluster. Une des conséquences directes à cela est que, pour de petits datasets qui tiennent en mémoire, un algorithme lancé depuis Spark en local sur votre machine mettra beaucoup plus de temps à s’exécuter que le même algorithme lancé depuis Python ou R, qui sont optimisés pour le mode local. En revanche, les performances deviennent extrêmement intéressantes lorsque les volumétries sont très importantes.

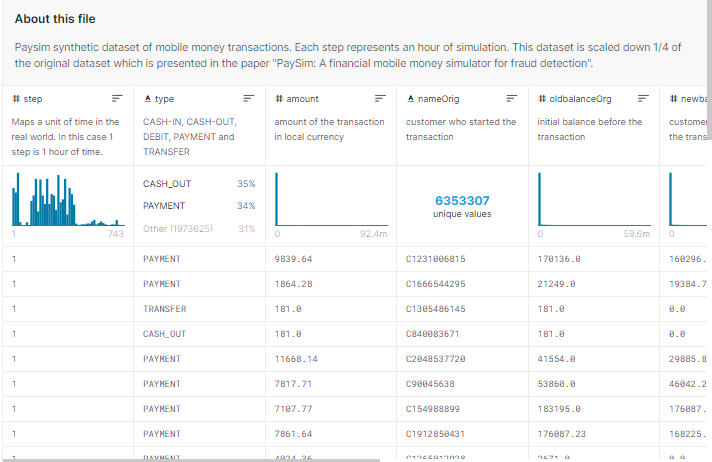
MLlib a été conçu pour une utilisation très simple des algorithmes en les appelant sur des RDD dans un format spécifique, quel que soit l’algorithme choisi. L’architecture se rapproche ainsi de ce que l’on trouve dans la librairie scikit-learn de Python, bien qu’il y ait encore des différences notables qui vont être effacées dans les prochaines versions de l’API.

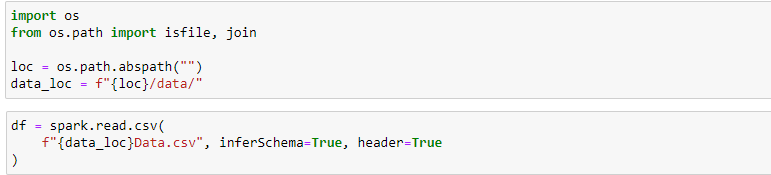
Les algorithmes présents dans MLlib sont, tout comme le reste du framework, développés en Scala, en se basant principalement sur le package d’algèbre linéaire [Breeze](http://www.scalanlp.org/) pour l’implémentation des algorithmes. De plus, pour faire fonctionner MLlib, il est nécessaire d’installer gfortran, ainsi que Numpy si vous utilisez l’API Python.

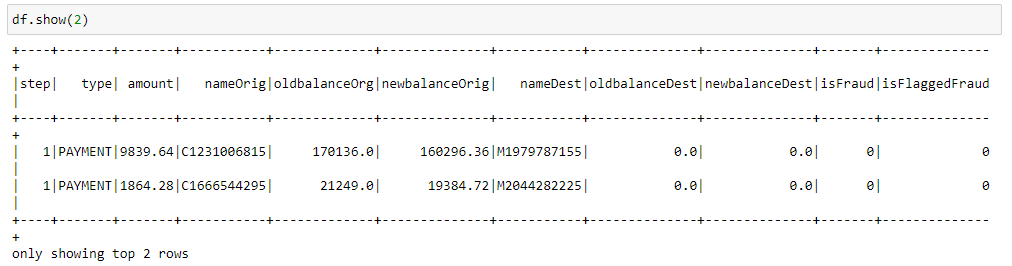
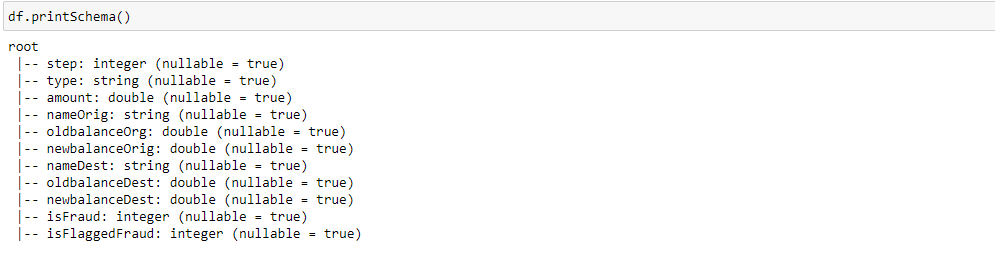
### Les types de données spécifiques à MLlib :

L’une des spécificités de MLlib (et peut-être une de ses faiblesses pour le moment) est qu’il nous contraint à utiliser des RDD aux types spécifiques. Les algorithmes implémentés nécessitent ainsi en entrée des RDD[Vector] (pour des données n’ayant pas de label), des RDD[LabeledPoint] (spécifiques à l’apprentissage supervisé) ou bien des RDD[Rating] (pour les systèmes de recommandation)

### Chargement du jeu de données et sélection des entités

Dataset : https://www.kaggle.com/arjunjoshua/predicting-fraud-in-financial-payment-services/data

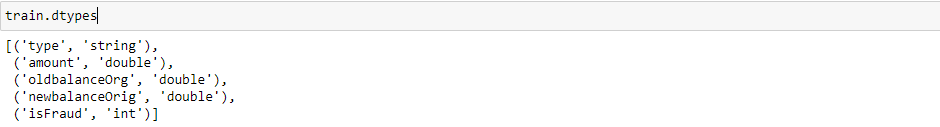


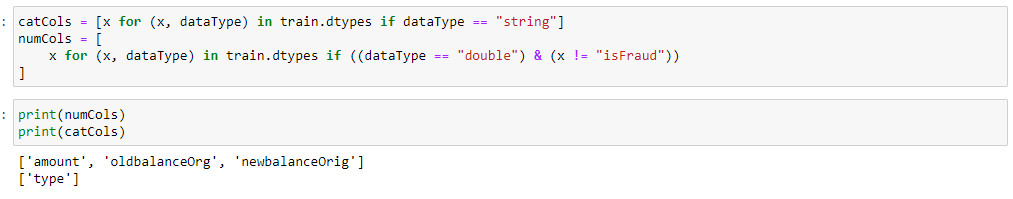


### Divisez l'ensemble de données en un train et un ensemble de test



### Définir les types de données







### One hot encoding à l'aide de PySpark MLlib

### One hot encoding

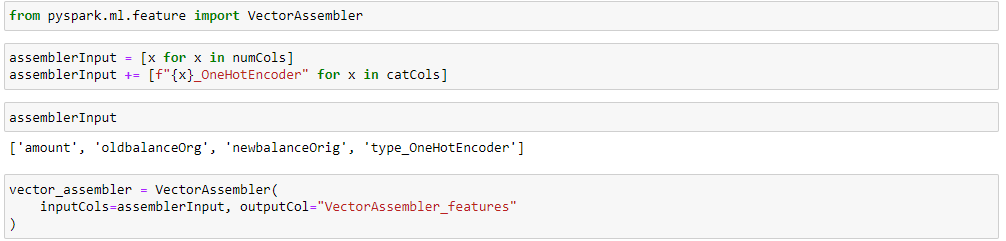
StringIndexer: Converts a single feature to an index feature. <http://spark.apache.org/docs/latest/ml-features#stringindexer>

OneHotEncoder: <http://spark.apache.org/docs/latest/ml-features#onehotencoder>

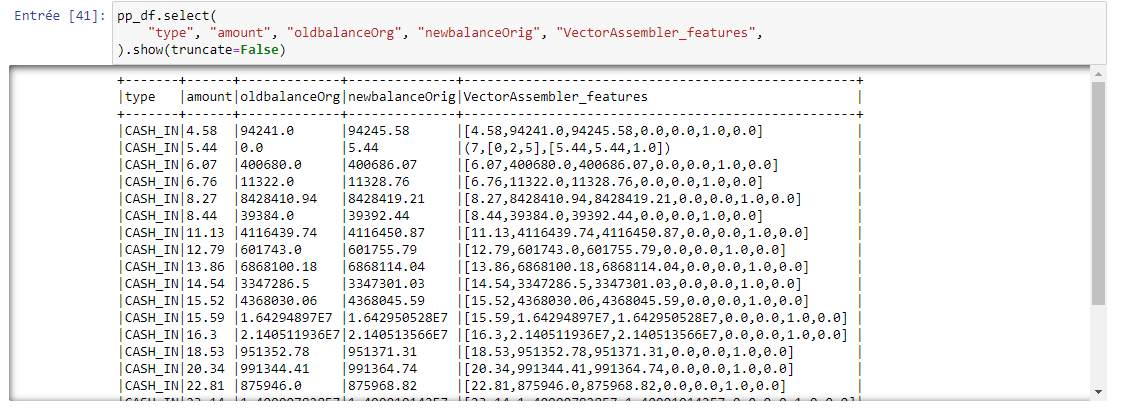
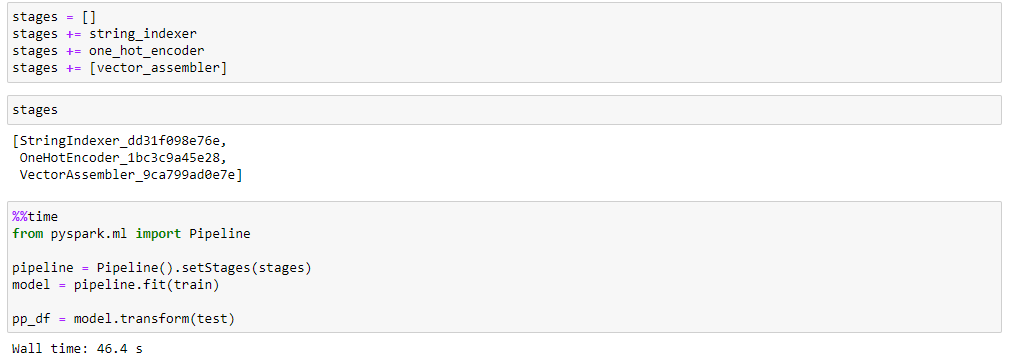
For more info: <http://spark.apache.org/docs/latest/ml-features>



### Comprendre VectorAssembler : combiner les valeurs de caractéristiques d'entrée dans un seul vecteur

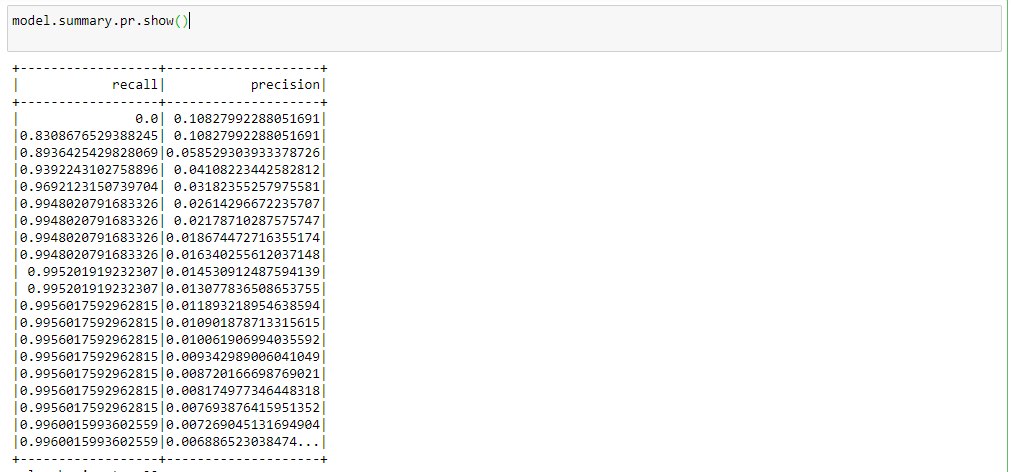
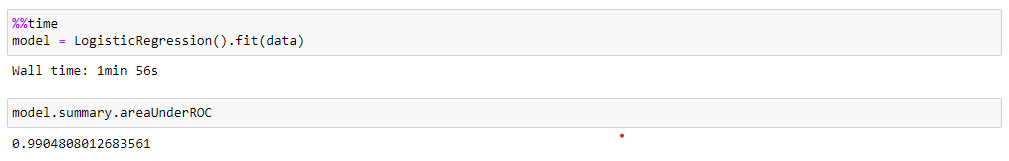


1. Créer un pipeline de prétraitement avec PySpark MLlib



### le modèle de régression logistique et évaluation des résultats





# Références

<https://towardsdatascience.com/pyspark-and-sparksql-basics-6cb4bf967e53>

<https://github.com/jpablo146unal/Trabajo_SparkQL>

<https://docs.microsoft.com/fr-fr/azure/hdinsight/spark/apache-spark-machine-learning-mllib-ipython>

<http://www.xavierdupre.fr/app/sparkouille/helpsphinx/notebooks/spark_mllib.html>

[https://www.decisionforest.com/downlo...](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqa1JSb21EUEFmOGJqLTVpaVZSVlhjcnQ3ZVdtQXxBQ3Jtc0trT0hyRjZSeEEtWmJDbXp0d3lCZWZ0TEpmU1dmdndHLWY4OHdmVnVBaDBNQVdVeVcxUEdyOVdycUJUam1SMG1zNkRBV0U4cTduOGFtQkhMaEwxb0Zhd2prT2lPdkdwOXB0ajlmNVVqN0tlRDhRVEV3MA&q=https%3A%2F%2Fwww.decisionforest.com%2Fdownloads%2F26)