Titre  
Machine Learning

Telco Customer Churn Prediction



Mohamed Iyadh Tajouri (4DS3)

Sommaire

1. introduction generale
2. chapitre introductif
3. chapitre : Business Understanding
4. chapitre 2 : Data understanding
5. chapitre 3 : Data preparation
6. chapitre 4 : Modeling
7. Chapitre 5 : Perspectives
8. Chapitre 6 : Apports académique et professionnel
9. chapitre 7 : Analyse des articles

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# Introduction générale

# Dans le monde d'aujourd'hui, l'information est un outil coûteux et important qui permet aux entreprises de se développer et d'augmenter de plus en plus leur chiffre d'affaires. Il ne fait aucun doute que l'information est le pouvoir des entreprises. En fait, la science s'est développée d'une manière qui pourrait nous donner un aperçu de l'avenir dans diverses industries et c'est ce sur quoi les entreprises se concentrent actuellement pour accroître leur croissance. Les technologies de l'information et de la prévision ont été les sciences les plus "attrayantes" et les plus innovantes et les plus utilisées au cours des dernières décennies. En effet, cela est directement lié à l'apparition d'une quantité massive de données inexploitées que nous devons analyser pour atteindre les objectifs commerciaux de l'entreprise en projetant une image personnalisée et réduite à partir d'un grand volume de données. C'est dans ce domaine que les entreprises sont à la pointe des nouvelles technologies émergentes et doivent assurer une sorte de "Prévision du futur" pour garantir une prise de décision bien adaptée aux besoins de l'entreprise et de son secteur. Compte tenu des objectifs de ce projet, notre travail sera organisé en sept chapitres comme suit : Le premier chapitre sera une introduction au projet, expliquant son contexte et présentant l'organisation hôte et les méthodologies suivies. Quant au deuxième chapitre, il représentera une vision axée sur le modèle de compréhension de l'entreprise et les objectifs appropriés en matière de data mining. Dans le troisième chapitre, nous parlerons de la compréhension des données avant d'aborder la préparation des données dans le quatrième chapitre afin de pouvoir passer à la modélisation dans le cinquième chapitre où nous nous concentrerons sur le choix et le test des différents algorithmes. Le sixième chapitre est un chapitre d'évaluation au cours duquel nous vérifions si le modèle choisi répond aux objectifs fixés. A la fin de ce projet, nous passons au déploiement du projet et à la création d'une interface web pour pouvoir utiliser correctement le système de prédiction créé.

# 

# 

# 

# 

# 

# 

# 

# CHAPITRE 1 : INTRODUCTION

## 1.1 Introduction

# Dans ce chapitre, nous procéderons à la présentation du cadre du projet qui nous permettra de mieux comprendre le problème. Ensuite, nous terminerons ce chapitre en soulignant l'importance du sujet de l'étude et en justifiant le choix de la méthodologie adoptée tout au long de ce projet.

## 1.2 Contexte et problématique

# Au cours des dernières décennies, les télécommunications mobiles sont devenues le moyen de communication dominant dans le monde entier. Dans plusieurs pays, la saturation du marché a atteint un niveau tel que chaque client potentiel doit être gagné à la concurrence. Dans le même temps, la normalisation des infrastructures mobiles et la réglementation publique ont permis au client de passer facilement d'un réseau à un autre, ce qui entraîne un marché fluide.

# Comme le coût de l'acquisition d'un nouveau client est plus élevé que celui de la fidélisation d'un client existant, les opérateurs de téléphonie mobile ont déplacé leur attention de l'acquisition à la fidélisation des clients. Il est donc nécessaire que ces entreprises aient la capacité de prévoir les clients qui leur seront fidèles sans intervention qui pourrait entraîner une perte de revenus, des dépenses supplémentaires pour la fidélisation et la reconquête des clients, des coûts publicitaires supplémentaires, un chaos organisationnel et budgétaire. En outre, les clients qui quittent l'entreprise de télécommunications pourraient inciter d'autres entreprises à faire de même. Par la suite, afin de maintenir la propension ou la tendance des clients à l'égard de l'entreprise, ils doivent tenir compte du comportement des clients et fournir les meilleurs services en fonction de leurs préférences. C'est ce que l'on appelle la "prédiction du taux de désaffection des clients". Il est essentiel de mettre en œuvre la prédiction du taux d'attrition dans leur approche de prévision des clients à haut risque.

## 1.3 Objectifs

# Notre objectif au cours de ce projet est de mettre l'apprentissage machine au service de la détection des clients qui vont quitter l'opérateur téléphonique (Churn customers).

## 1.4 Notions théoriques

### 1.4.1 Apprentissage machine (ML)

# De nombreuses approches sont utilisées pour découvrir les propriétés des ensembles de données. L'apprentissage Machine est l'un d'entre eux. Dans l'approche ML elle-même, différentes techniques sont proposées. Elles sont à choisir en fonction de la nature des données et du type d'étude :

# Méthodes utilisant des techniques de classification et de segmentation

# Méthodes utilisant les principes des arbres de décision assez proches des techniques de classification

# Méthodes basées sur des principes et des règles d'association ou d'analogie

# Méthodes exploitant les capacités d'apprentissage des réseaux de neurones

# Algorithmes de Bayes naïfs, séries temporelles, régression linéaire...

# L'apprentissage machine se concentre principalement sur la conception d'algorithmes qui peuvent tirer des enseignements des bases de données et de faire des prédictions en se basant sur ce qu'il a appris, ou il peut extraire des connaissances sans besoin d'apprendre à partir de données historiques. On distingue donc deux types de ML : l'apprentissage supervisé et les méthodes d'apprentissage non supervisé.

### 1.4.2 Apprentissage non supervisé

# L'apprentissage non supervisé consiste à diviser un groupe hétérogène de données en sous-groupes afin que les données considérées comme les plus similaires sont associées au sein d'un groupe homogène afin que les données considérées comme différentes se retrouvent dans d'autres groupes distincts, l'objectif principal est une extraction de connaissances organisées à partir de ces données. Il y a trois tâches principales réalisable à partir de méthodes d'apprentissage non supervisées :

# Partitionnement des données (clustering).

# La détection d'éléments atypiques (détection d'aberrations).

# Réduction des dimensions.

### 1.4.3 Apprentissage supervisé

L'apprentissage supervisé, par contre, est une technique d'apprentissage automatique où l'on cherche à produire automatiquement des règles à partir d'une base de données d'apprentissage contenant (généralement des cas déjà traités et validés). De nombreuses tâches peuvent être réalisées à partir des méthodes d'apprentissage supervisé grâce aux énormes quantités de données que nous collectons ces dernières années :

* Vision par ordinateur
* Reconnaissance des formulaires
* Reconnaissance de l'écriture manuscrite
* Reconnaissance de la parole

## 1.5 Modèle de processus du projet : CRISP-DM

La méthode CRISP-DM (The Cross-Industry Standard Process for Data Mining) a été

développé à l'origine par IBM dans les années 1960 pour réaliser des projets de Data Mining. Elle reste aujourd'hui la seule méthode qui puisse être utilisée efficacement pour tous les projets de Data Science. Le CRISP-DM n'a pas été construit dans un cadre théorique, académique, ni par un comité d'élite de derrière des portes fermées. Ces deux approches du développement de méthodes ont été essayées dans le passé, mais ont rarement conduit à des normes pratiques, avec un succès largement adopté. Le CRISP-DM a réussi parce qu'il est solidement fondé sur l'expérience pratique, le monde réel de la façon dont les spécialistes gèrent les projets de Data Mining.

### 1.5.1 Caractéristiques

CRISP DM est tout simplement le modèle de référence des projets de Data Mining. Le modèle de processus fournit une vue d'ensemble du cycle de vie d'un projet de Data Mining tel qu'il est et tel qu'il devrait être. Il contient les phases d'un projet, leurs tâches respectives et les relations entre ces tâches. Cette méthode est agile et itérative, c'est-à-dire que chaque itération apporte des connaissances métier supplémentaires qui permettent de mieux aborder l'itération suivante. C'est pour cette raison que, même si nous la vendons comme un projet, la Data Science est plus une approche globale qu'un projet simple. Le cycle de vie d'un projet de Data Mining comprend six phases. La figure suivante montre les phases d'un processus CRISP-DM. L'ordre des phases est flexible : un aller-retour entre les différentes phases est toujours nécessaire, il dépend du résultat de chaque phase.

# 

# 

# 

# 

# 

# 

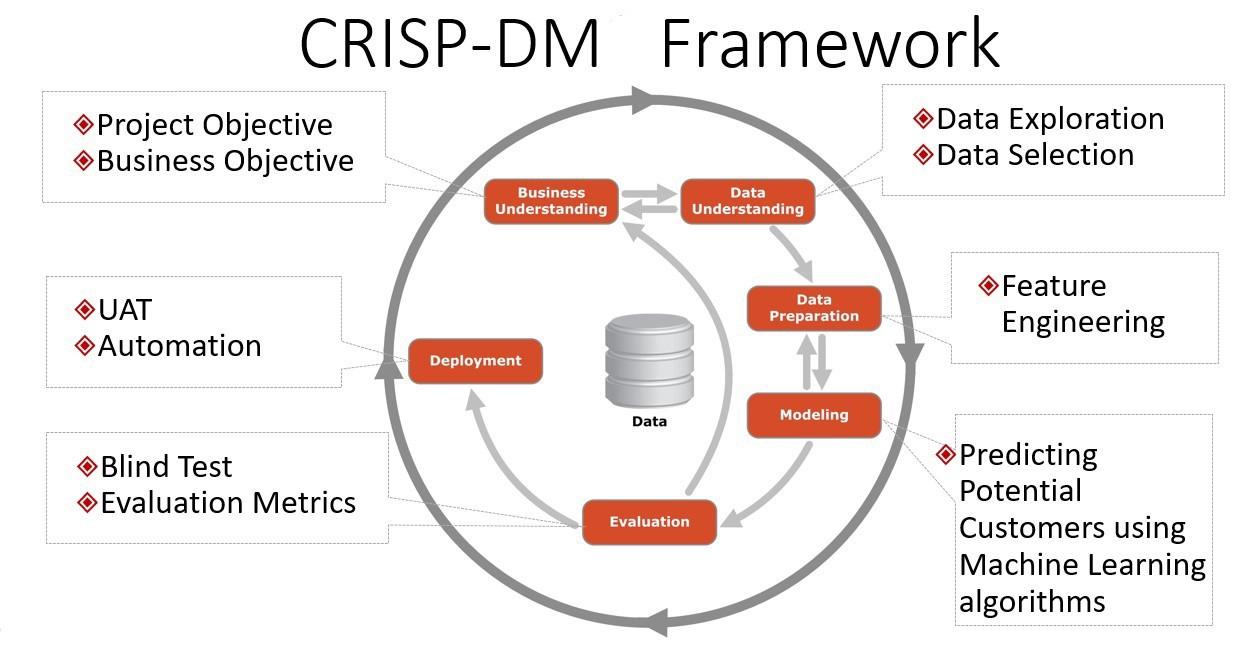
# 

# 

# 

# Introduction

CRISP DM est simplement le modèle de référence des projets de Data Mining. Le modèle de processus fournit une vue d'ensemble du cycle de vie d'un projet d'exploration de données tel qu'il est et tel qu'il devrait être. Il contient les phases d'un projet, leurs tâches respectives et les relations entre ces tâches.

****

## Business Understanding

La méthode CRISP-DM (The Cross-Industry Standard Process for Data Mining) a été développée à l'origine par IBM dans les années 1960 pour mener à bien des projets d'exploration de données. Elle reste aujourd'hui une méthode qui peut être utilisée efficacement pour tous les projets de Data Science.

## Data Understanding

Les objectifs principaux de cette phase sont d'associer les données à leur signification d'un point de vue métier, de déterminer précisément les données à analyser et de déterminer leur qualité .

## Data Preparation

Cette phase de préparation des données regroupe les activités liées à la construction de l'ensemble précis de données à analyser, à partir des données brutes. Il comprend la classification des données selon des critères choisis, le nettoyage des données, et surtout de les rendre compatibles avec les algorithmes qui seront utilisés. Cette phase est extrêmement importante et doit être effectuée avec soin afin d'éviter d'avoir de faux résultats des algorithmes utilisés dans la phase suivante.

## Modeling

Des méthodes statistiques ou mathématiques seront utilisées dans cette phase pour résoudre le problème de l’entreprise, en utilisant des outils de modélisation, nous devons préparer un ou plusieurs modèles sur l’ensemble de données préparées au cours de l’étape précédente.

## Evaluation

Dans cette phase, nous devons évaluer les résultats obtenus du modèle(s) et voir s’ils répondent au besoin du client et s'assurer que les résultats admettent une solution quant au problème évoqué.

## Deployment

C'est la dernière étape du processus. Il consiste à mettre en production pour les utilisateurs finaux les modèles obtenus. Son objectif est de mettre les connaissances obtenues par modélisation sous une forme adaptée et de les intégrer dans le processus décisionnel.

conclusion

# Chapitre 1: Business Understanding

Avant de commencer un projet, nous devons avoir des idées claires sur ce que nous essayons d'accomplir. Il est très important de comprendre toutes les exigences du projet d'un point de vue commercial, de convertir ces exigences en problématiques d'exploration de données, puis de préparer un premier plan de projet.

## Objectif du projet

L'objectif de ce projet est de prédire le taux de désabonnement des clients afin que les entreprises de télécommunications puissent immédiatement prendre des mesures pour fidéliser les clients existants en améliorant les services à la clientèle, la qualité des produits , car la perte de clients pourrait entraîner une perte de revenus importante. que conserver l'ancien . Il peut atteindre jusqu'à cinq à six fois moins cher que de trouver un nouveau client.

## Déterminer la data mining goals

En prévoyant le taux de désabonnement des clients, les entreprises peuvent immédiatement prendre des mesures pour fidéliser leurs clients. La prédiction peut être effectuée en analysant les données des clients à l'aide de techniques d'exploration de données.

Nombreux algorithmes de classification peuvent être appliqués tels que (KNN K-Nearest neighbor , Naive Bayes, Logistic Regression …….

# Chapitre 2 : Data Understanding

## Introduction

Le but de cette phase est de déterminer précisément les données à analyser, d'identifier la qualité des données disponibles et de lier les données à leur signification d'un point de vue commercial.

## Source des données

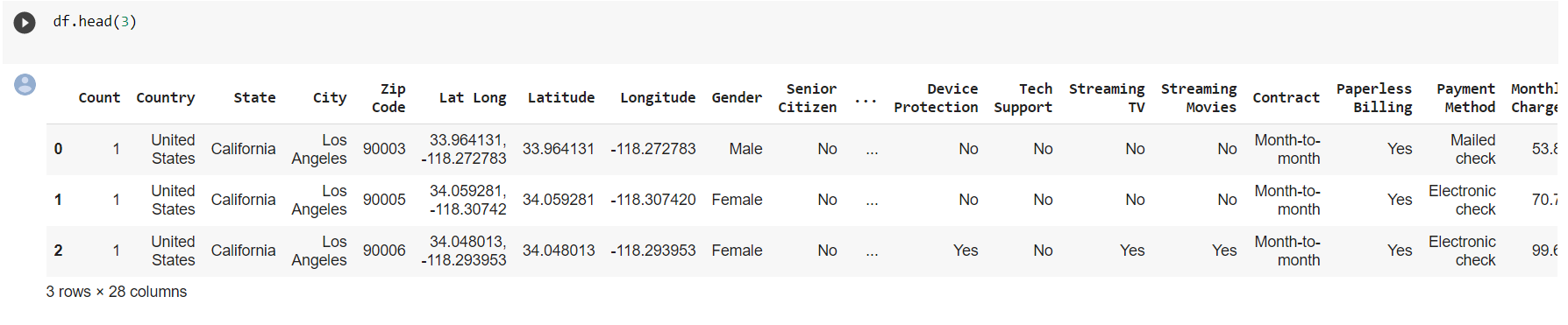
Une variété de sources de données est nécessaire pour créer des prévisions de désabonnement et des modèles de segmentation des clients comprenant leur démographie, les informations géographiques et les appels registres détaillés, etc. Dans le contexte des travaux en cours, les données démographiques et l'information géographique étaient disponibles, car la plupart des sources de données comprennent des Informations, c'est-à-dire CDR, opérations, etc.

L'ensemble de données utilisé pour l'expérimentation est l'ensemble de données de désabonnement des clients Telco obtenu à partir de “<https://www.kaggle.com/blastchar/telco-customer-churn>.”

## Chargement des données

Le nom de l'ensemble de données est colab-Notebooks/Telco\_Customer\_Churn.csv que nous avons mis à jour de Kaggle.

### 



## Exploration des données

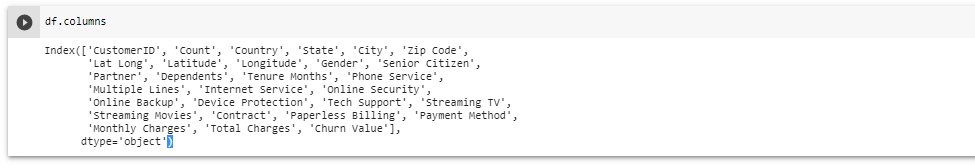
L'ensemble de données contient 29 entités, une étiquette et 7043 enregistrements. L'ensemble de données contient des entités avec des types catégoriques et numériques. Le modèle proposé ne peut traiter que des données numériques. Un prétraitement doit donc être effectué pour convertir toutes les caractéristiques catégorielles en types numériques.



### 

La première chose que nous allons explorer, c’est nos labels cibles comme on peut le constater dans les figures ci-dessous qui montrent le nombre de clients actifs et désabonnés dans notre ensemble de données.

Il y a 5174 clients actifs et seulement 1869 clients désabonnés et nous pouvons le voir plus clairement dans cette figure 6 ci-dessous



### Informations sur les services

* **Phone Service** - Si le client dispose d'un service téléphonique (Oui, Non)
* **Multiple Lines** - Si le client dispose de plusieurs lignes (oui, non, pas de service téléphonique)
* **Internet Service** - Fournisseur de services Internet du client (DSL, fibre optique, non)
* **Online Security** - Si le client dispose d'une sécurité en ligne (oui, non, pas de service Internet)
* **Online Backup** - Si le client dispose d'une sauvegarde en ligne (oui, non, pas de service Internet)
* **Device Protection** - Si le client dispose d'une protection de l'appareil (oui, non, pas de service Internet)
* **TechSupport** - Si le client dispose d'un support technique (oui, non, pas de service Internet)
* **Streaming TV** - Si le client dispose de la télévision en continu (Oui, Non, Pas de service Internet)
* **Streaming Movies** - Si le client a des films en streaming (Oui, Non, Pas de service Internet)

### Informations sur le compte client

* **Durée du mandat** - Nombre de mois pendant lesquels le client est resté dans l'entreprise
* **Contrat** - La durée du contrat du client (mois par mois, un an, deux ans)
* **Facturation sans papier** - Indique si le client dispose d'une facturation sans papier (Oui, Non)
* **Payment Method** - Le mode de paiement du client (chèque électronique, chèque postal, virement bancaire (automatique), carte de crédit (automatique))
* **Monthly Charges** - Le montant facturé au client chaque mois
* **Total Charges** - Le montant total facturé au client

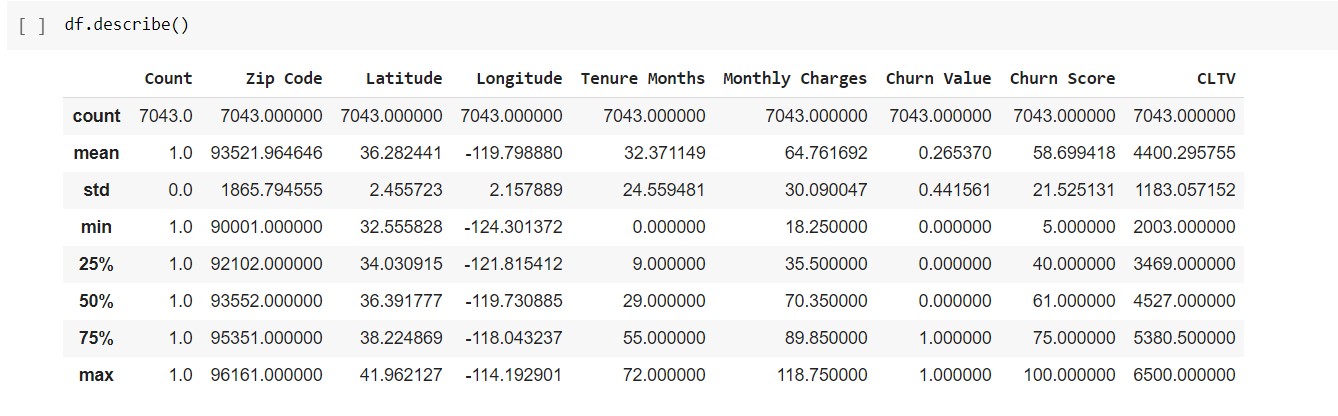
### Informations démographiques des clients

* **customerID** - ID client
* **Sexe** - Que le client soit un homme ou une femme
* **Senior Citizen** - Que le client soit une personne âgée ou non (1, 0)
* **Partenaire** - Que le client ait un partenaire ou non (Oui, Non)
* **Personnes à charge** - Que le client ait des personnes à charge ou non (Oui, Non)

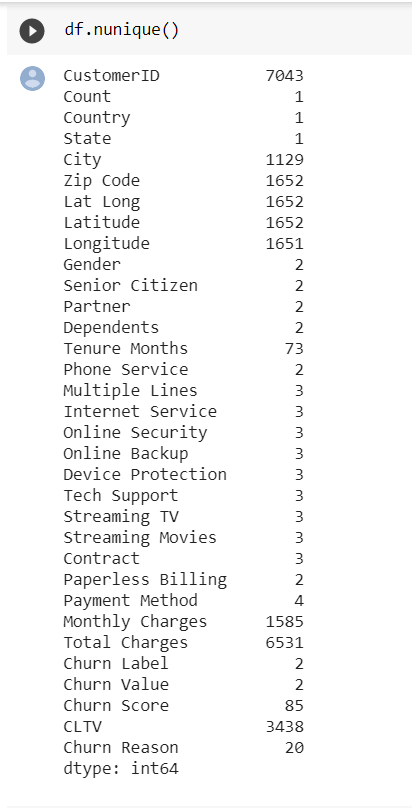
### 

### Audit des données

describe() est utilisé pour afficher certains détails statistiques de base tels que le centile, la moyenne, la norme, etc. d'une base de données ou d'une série de valeurs numériques. Lorsque cette méthode est appliquée à une série de chaînes, elle renvoie une sortie différente, illustrée dans les exemples ci-dessous.

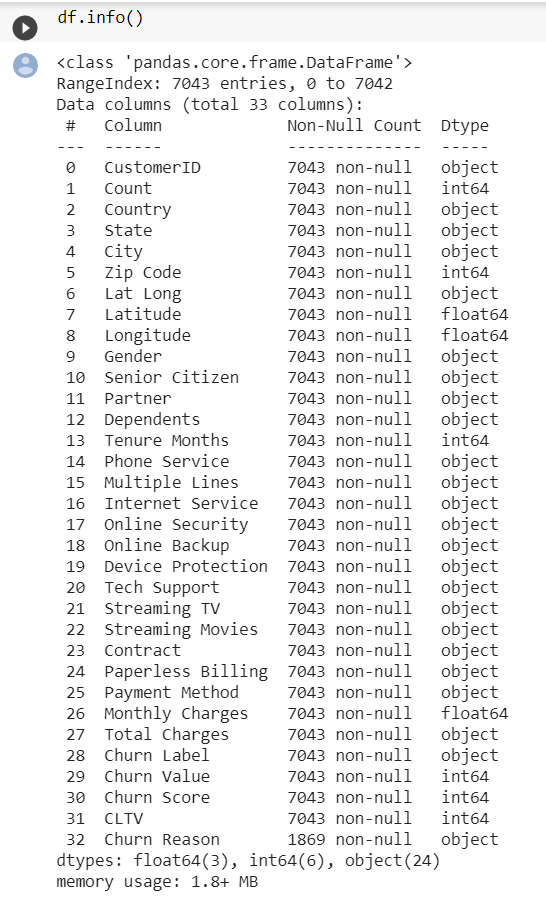


La fonction unique () renvoie une série avec le nombre d'observations distinctes sur l'axe demandé. Si nous définissons la valeur de l'axe sur 0, alors il trouve le nombre total d'observations uniques sur l'axe d'index comme le montre la figure ci-dessous:



La fonction info () est utilisée pour imprimer un résumé concis d'un DataFrame. Cette méthode imprime des informations sur un DataFrame, y compris les dtypes d'index et de colonne, les valeurs non nulles et l'utilisation de la mémoire.

la figure ci-dessous nous montre le résultat obtenu:

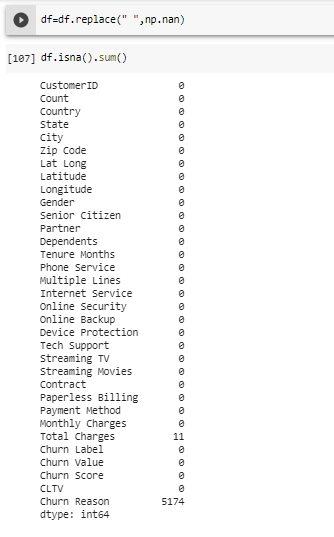


## Visualisation des données

Les valeurs manquantes

Maintenant nous voulons voir le nombre d' éléments manquants dans un ensemble de données ou dans notre dataset.

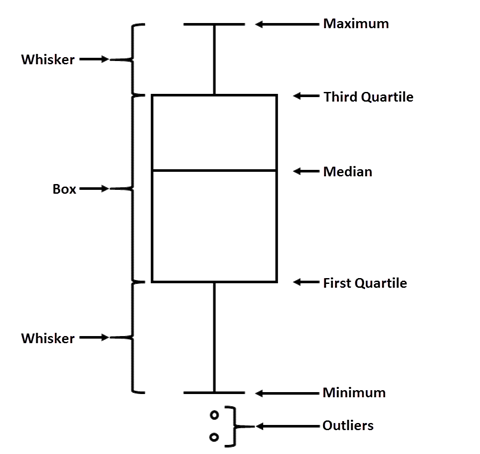
L’objectif de cette étape est de remplacer chaque espace trouvé dans l’ensemble des lignes par NAN comme le montre la figure ci-dessous afin que la fonction **isna** puisse calculer la somme des valeurs manquantes par la suite.



* On remarque l'existence des valeurs manquantes au niveau de la colonne ‘TOTAL CHARGES’ .

Nous avons converti le type de la variable total charges de type objet vers le type numérique afin qu’on puisse réaliser la présentation en boîte à moustache par la suite.

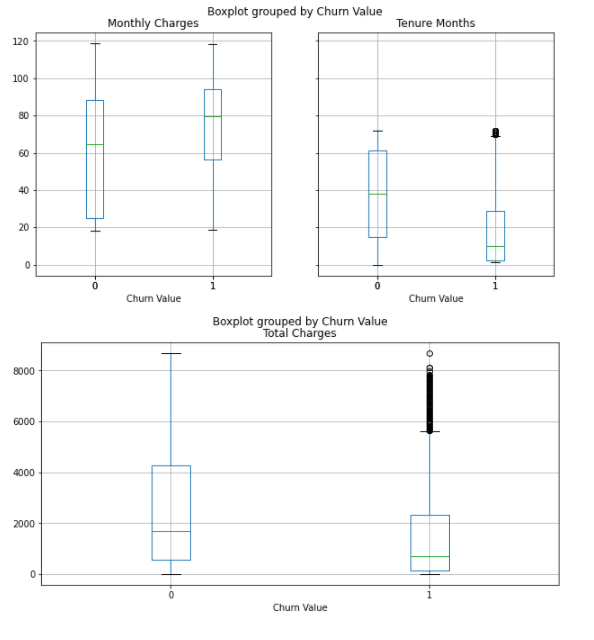
### Boîte à moustache



Les boîtes à moustaches obtenues dans la figure ci-dessous nous proposent un résumé visuel de la variabilité des valeurs ‘Monthly Charges’,’Tenure Months’ et ‘Total Charges’.

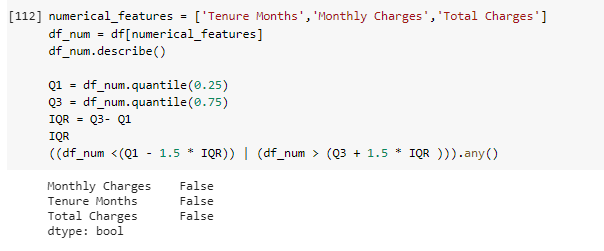
Ils présentent les valeurs de médiane, quartiles supérieur et inférieur, minimale et maximale, ainsi que tout point aberrant du jeu de données. Les points aberrants peuvent révéler des erreurs ou des occurrences inhabituelles dans les données. Une boîte à moustaches est créée grâce à un champ numérique ou de taux/ratio sur l’axe des y.





* Les clients en désabonnement ont des frais mensuels plus élevés avec une médiane d'environ env. 80 USD et un intervalle interquartile bien inférieur à celui des non-churners (médiane d'environ 65 USD).
* Les clients en déshérence ont une ancienneté beaucoup plus faible avec une médiane de ca. 10 mois par rapport à une médiane de non désabonnés de ca. 38 mois.
* Les charges totales sont le résultat de l'ancienneté et des charges mensuelles, qui sont plus pertinentes sur une base individuelle.

vérifier les valeurs aberrantes en appliquant la méthode IQR si la valeur est en dehors des limites IQR.



* Le code ci-dessus donne une sortie avec des valeurs vraies et fausses. Le point de données où nous avons False signifie que ces valeurs sont valides alors que True indique la présence d'une valeur aberrante.

### Visualisation

Comme la plupart des données étaient catégoriques, des graphiques à barres ont été utilisés pour tracer la relation entre la variable cible et une colonne particulière

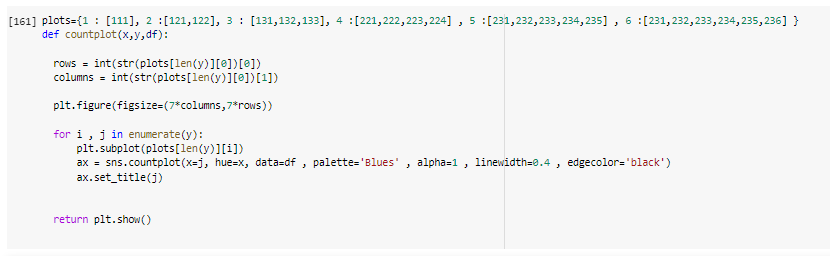
Dans une première étape, nous avons cherché à visualiser le nombre total des personnes actives et ceux qui se sont désabonnés.

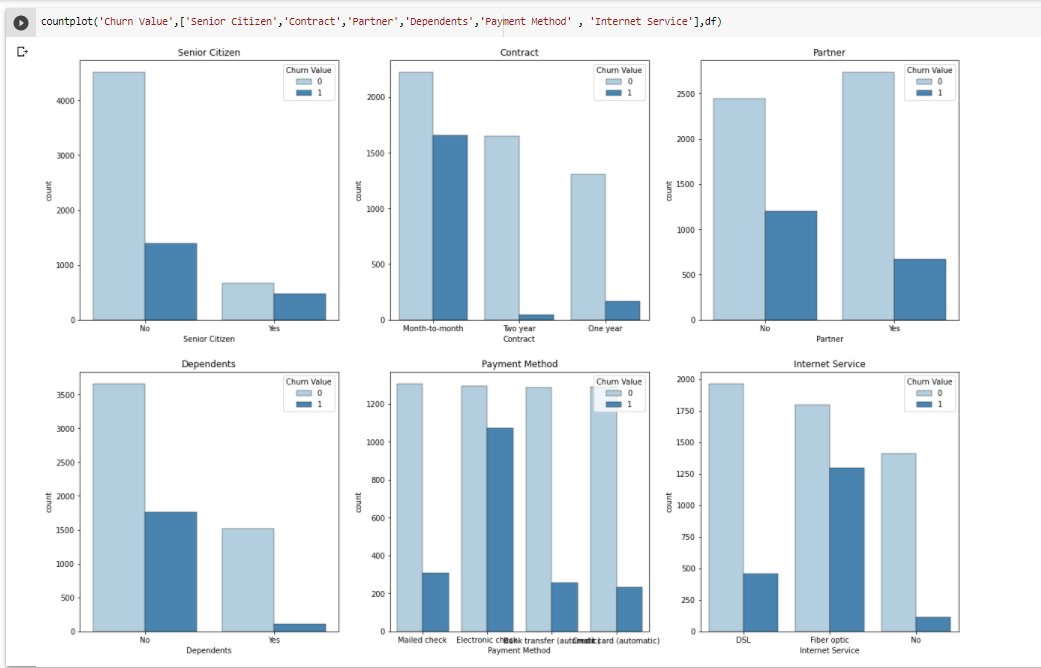


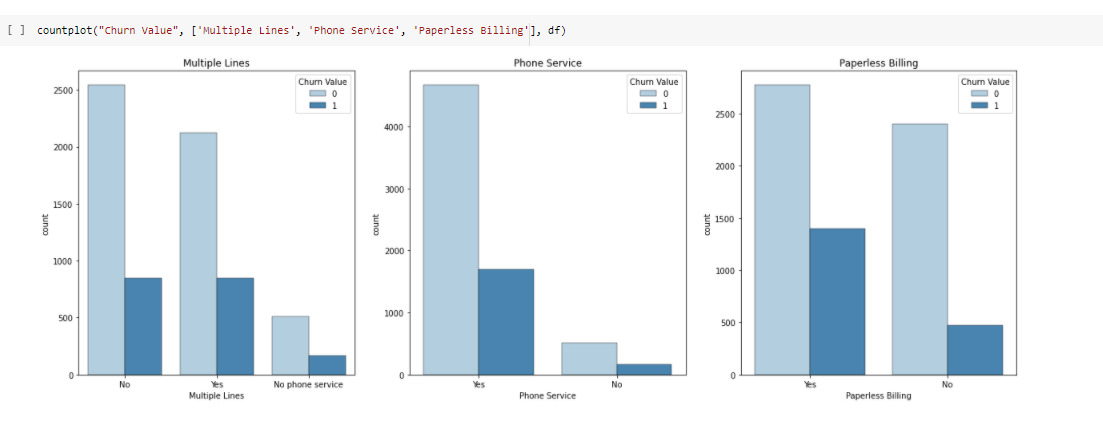
* Le graphique montre un déséquilibre de classe des données entre les personne désabonnée et non des abonnées

### Aperçus :

La méthode ci-dessous permet l’affichage de plusieurs graphiques en barres des variables (‘**senior Citizen**', '**Contract**', '**Partner**', '**Dependents**', '**Payment Method**', '**Internet Service**’, '**Multiple Lines**', '**Phone Service**', '**Paperless Billing**') en fonction de la variable target **churn value**.







**On remarque que :**

* 42% des personnes les plus âgées vont désabonner alors que 23% des personnes les moins âgées vont désabonner. De plus, on remarque que le nombre des personnes les moins âgées qui vont quitter est largement grand par rapport au nombre des personnes les plus âgées.
* Le nombre des personnes qui se désabonnent qui ont un partenaire sont plus élevés que les churn qui n'ont pas de partenaire.
* une grande différence entre les personnes qui se désabonnent, qui utilisent dsl et qui utilisent la fibre optique de plus on a un pourcentage faible pour qui n'ont pas d'internet ,donc on peut conclure que cette colonne donne beaucoup d'informations sur le churn status.
* La plupart des personnes qui se désabonnent sont celles qui ont le contrat month to month après ça on trouve le contrat annuel et le plus petit pourcentage est celui de contrat de deux ans.
* On remarque que la plupart des personnes qui se désabonnent sont les personnes qui payent avec l'electronic bank aprés ca on trouve la méthode mailed check et la plus petite pourcentage est celle de bank transfer et credit card.
* On remarque une grosse différence entre les personnes qui se désabonnent non dépendantes et les churn dépendants de plus les dépendants contient un nombre de churn très petit qu'on peut le négliger devant le nombre de dépendants.

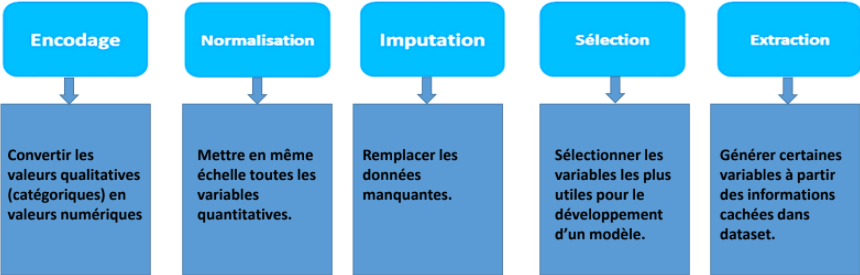
## conclusion

Dans ce chapitre, nous avons établi une explication du processus que nous avons suivi pour garantir l'étape de compréhension des données, traitant des données manquantes ainsi que des données illogiques, et on a maintenu un aperçu des données. Ces données seront transformées et corrigées au cours de la phase de préparation qui fera l’objet du chapitre suivant.

# Chapitre 3 : Data Preparation

Introduction :

L’une des phases les plus importantes et les plus longues dans un projet Data Mining est la préparation des données. La préparation des données pourrait consacrer 70% du temps et de l’effort d’un projet. Le fait de consacrer l’énergie adéquate aux phases antérieures de compréhension métier et de compréhension des données joue bien évidemment un rôle important dans la minimisation de ce surcoût. Nous verrons comment les données seront transformées et préparées pour le prochain chapitre « Modélisation ».



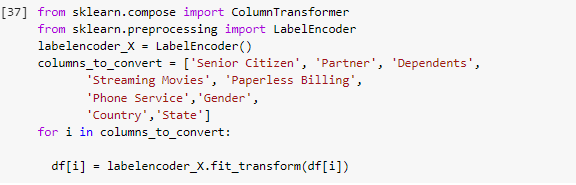
[.](https://fr.wikipedia.org/wiki/R%C3%A9duction_de_la_dimensionnalit%C3%A9)

## Encodage des données :

De nombreux algorithmes d'apprentissage peuvent prendre en charge des valeurs catégoriques sans autre manipulation, mais il existe de nombreux autres algorithmes qui ne le font pas. Les variables catégoriques ne peuvent prendre qu'un nombre limité et généralement fixe de valeurs possibles. Certains exemples incluent la couleur («rouge», «jaune», «bleu»), la taille («petit», «moyen», «grand») ou les désignations géographiques (état ou pays). Encore une fois, il n’existe pas de réponse unique sur la manière d’aborder ce problème, Voici nos principales stratégies:

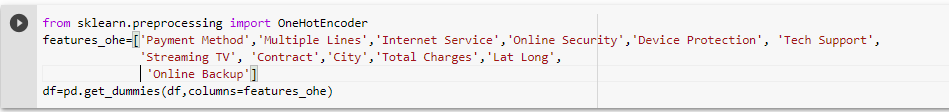
Label Encoding :

Les fonctionnalités suivantes sont catégoriques et prennent chacune 2 valeurs (principalement oui / non) - sont donc transformées en entiers binaires



One-Hot Encoding :

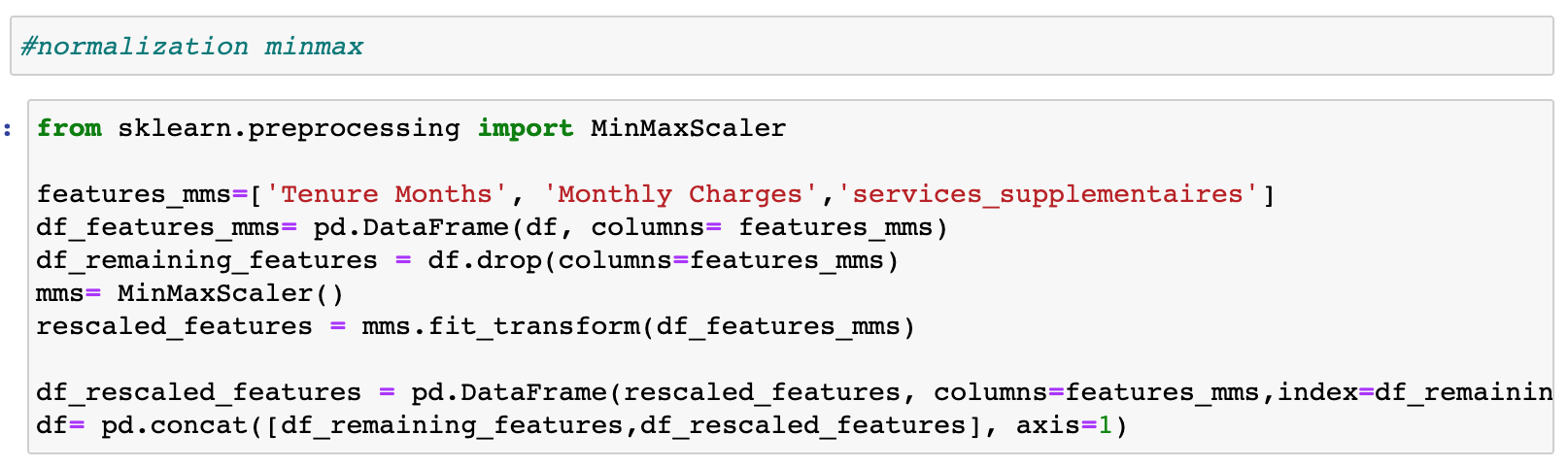
Les caractéristiques suivantes sont catégoriques, mais pas ordinaires (pas de classement), mais prennent plus de 2 valeurs. Pour chaque valeur, une nouvelle variable est créée avec un entier binaire indiquant si la valeur est apparue dans une entrée de données ou non (1 ou 0).



## 

## Normalisation :

MinMaxScaler :

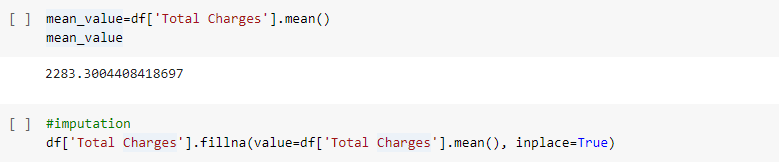
Les valeurs des entités numériques sont remises à l'échelle entre une plage de 0 et 1. La mise à l'échelle min-max est l'approche standard pour la mise à l'échelle. Pour les fonctionnalités normalement distribuées, un scaler standard peut être utilisé, qui met à l'échelle les valeurs autour d'une moyenne de 0 et d'un écart type de 1. Pour plus de simplicité, nous utilisons un scaler min-max pour toutes les caractéristiques numériques

## Nettoyage des données

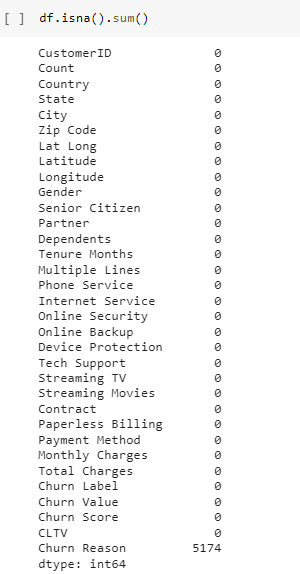
Dans chaque projet data science, il est pratiquement impossible d’avoir un cas parfait et une base de données aussi propre, homogène et complète. Ainsi, il faut absolument corriger plusieurs ambiguïtés.

Dans ce cas, nous avons proposé au client quelques modifications nécessaires concernant les données manquantes et la suppression de certaines colonnes inappropriées qui n'affectent pas la cohérence de la base de données

### Elimination des données manquantes :

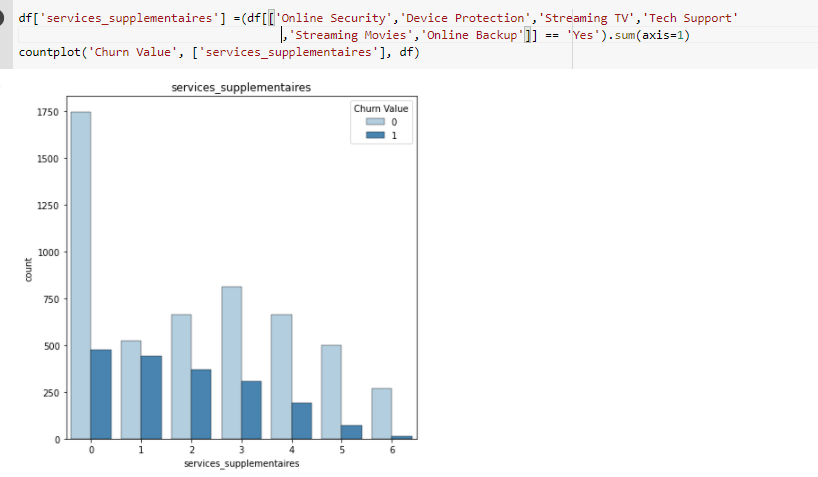


Nous avons remplacé les 11 valeurs manquantes de la colonne Total charges par le moyenne de ses valeurs.

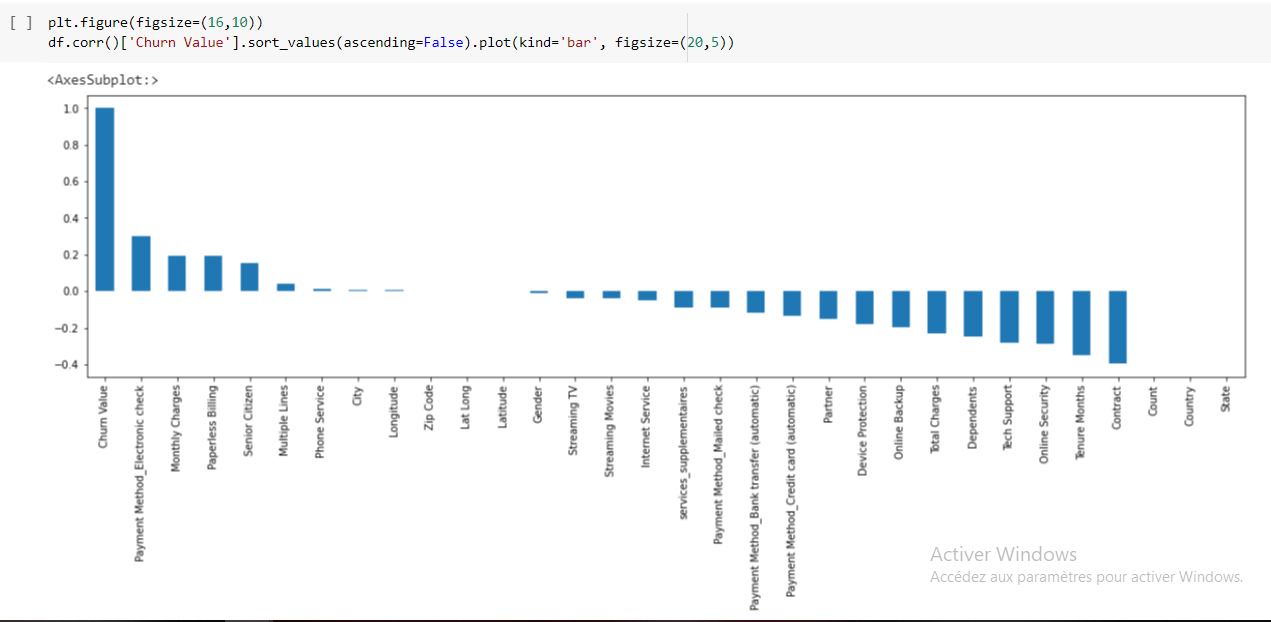


## Features selection :

La **sélection de caractéristique** (ou **sélection d'attribut** ou **de variable**) est un processus utilisé en [apprentissage automatique](https://fr.wikipedia.org/wiki/Apprentissage_automatique) et en [traitement de données](https://fr.wikipedia.org/wiki/Traitement_de_donn%C3%A9es). Il consiste, étant donné des données dans un espace de grande dimension, à trouver un sous-ensemble de variables pertinentes . C'est-à-dire que l'on cherche à minimiser la perte d'information venant de la suppression de toutes les autres variables. C'est une méthode de [réduction de la dimensionnalité](https://fr.wikipedia.org/wiki/R%C3%A9duction_de_la_dimensionnalit%C3%A9)



* Le countplot montre un taux de désabonnement très élevé pour les clients qui ont 1 service supplémentaire.
* Les clients disposant d'un très grand nombre de services supplémentaires ont un faible taux de désabonnement.

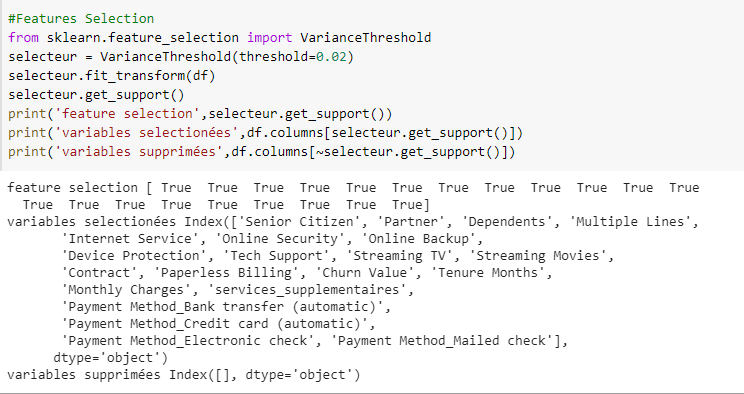


Dans cette partie, nous supprimerons les colonnes que nous avons conclues après avoir compris les données qu'elles ne nous donnent pas d'informations sur la variable cible



VarianceThreshold :

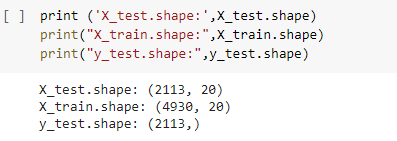
Supprime toutes les fonctionnalités à faible variance. Cet algorithme de sélection de features ne prend en compte que les features (X), et non pas les outputs (y), et peut donc être utilisé pour un apprentissage non supervisé.



# 

## Train Data and Test Data :

Pour la conduite des étapes d'entraînement et de test du modèle, l'ensemble de données est divisé en 70% de données d'entraînement et 30% de données de test. La colonne «Churn» est définie comme la classe (le «y»), les colonnes restantes comme les caractéristiques (le «X»).



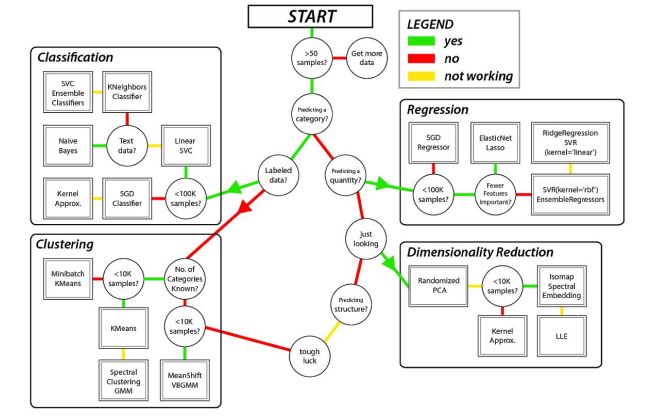
## Conclusion :

La phase de préparation des données prend plus de 70% du temps estimé du projet.Ces eﬀorts sont consacrés à la collecte, au nettoyage et à la préparation des données pour une analyse. Les données sont désormais utilisables et prêtes à attaquer la phase de modélisation dans laquelle nous allons créer nos diﬀérents modèles.

# Chapitre 4 : Modélisation.

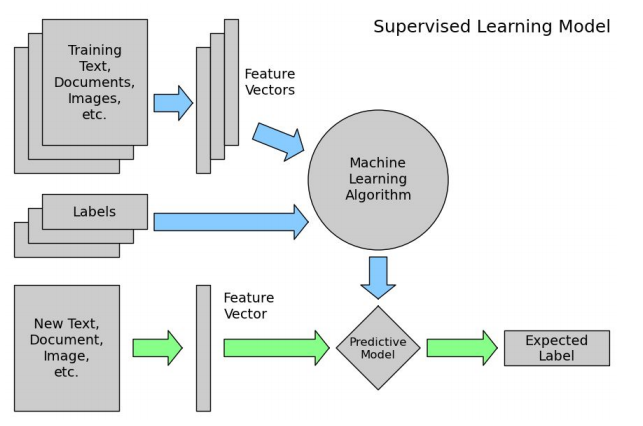
Après la phase de préparation des données, dans ce qui suit, nous basons notre travail sur la bibliothèque Scikit-Learn pour la modélisation et l'apprentissage.

Scikit-Learn est une bibliothèque open source de modèles d'apprentissage statistique ou d'apprentissage automatique développés à l'aide du langage de programmation Python.



### Technique de modélisation

L'objectif de la classification supervisée est principalement de définir des règles de classement des objets en classe à partir de variables qualitatives ou quantitatives caractérisant ces objets. Il y a d'abord un ensemble d'apprentissage, où les étiquettes sont connues. L'ensemble d'apprentissage est utilisé pour apprendre le modèle. règles de classification. Il est nécessaire d'étudier la précision des différents modèles, de les comparer et de les appliquer, d'évaluer les cas de sous ou de sur-ajustement. Nous utilisons généralement un deuxième ensemble indépendant, appelé ensemble de validation ou de test.

Le graphique ci-dessous montre la technique de modélisation pour un modèle d'apprentissage supervisé général:

### Indicateurs de test

In the following we focus on the indicators that allow us to measure the quality of the model and to choose the right classifier for our project. This will lead us to explain that calculating these indicators is not always sufficient because of the phenomenon of overfitting. In this context, we will describe the contours of a methodology in order to obtain an efficient model.

### Précision

La mesure la plus simple qui peut être utilisée pour estimer un classificateur est la précision. La précision mesure le pourcentage d'enregistrements de l'ensemble de test qui ont été correctement étiquetés par le classificateur de l'ensemble complet.

Par exemple, un classificateur qui prédit les étiquettes correctes 80 fois dans un ensemble de test contenant 100 enregistrements aurait une précision de 80/100 = 80%. Cependant, la précision de la classification n'est pas suffisante pour juger de la performance, car elle peut donner une bonne fausse idée des résultats prédits, en particulier lorsque nous traitons des classes déséquilibrées. Par exemple, pour un classificateur qui prédit si le texte d'un e-mail est du spam ou du ham, nous pouvons avoir une bonne précision de 95%. En fait, le classificateur peut bien fonctionner sur la classe majoritaire qui est évidemment ham, et traiter la classe minoritaire (spam) comme des valeurs aberrantes. Ainsi, une matrice de confusion est pertinente dans ce cas pour éviter cette confusion.

### Matrice de confusion

La matrice de confusion est un moyen clair d'évaluer un modèle de classification car elle efface toute ambiguïté sur le

performance du modèle sur chaque classe.

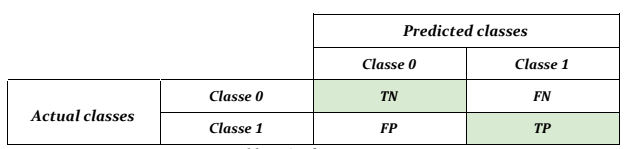
Restons dans l'exemple d'un classificateur binaire, qui prédit 2 classes: la classe 0 et la classe 1.

Pour mesurer les performances de ce classificateur, il faut distinguer 4 types d'éléments:

* True Positive (TP): éléments de la classe 1 correctement prédits
* True Negative (TN): éléments de classe 0 correctement prédits
* Faux positifs (FP): éléments de classe 1 mal prédits
* Faux négatif (FN): éléments de classe 0 mal prédits

Ces informations peuvent être rassemblées et affichées sous forme de tableau dans une matrice de confusion. Dans le cas

d'un classificateur binaire, on obtient:



### Precision and Recall

It is possible to calculate several indicators summarizing the confusion matrix. For example, if we want to account for the quality of the prediction on class 1, we define:

Precision​: Proportion of elements well classified for a given class :



Recall​ : Proportion d'éléments bien classés par rapport au nombre d'éléments de la classe à prédire:

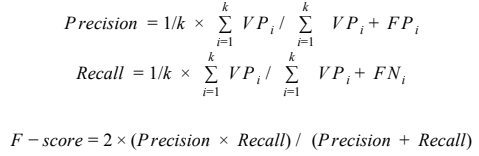


F1-score​: Compound measure of precision and recall:



It is possible to calculate all these indicators for each class. The average for each class of these

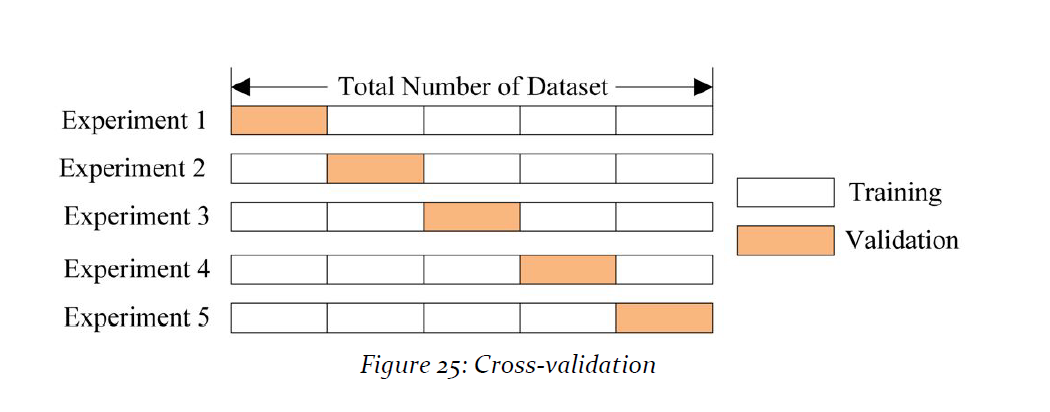
indicators gives global indicators on the quality of the classifier:



### Cross-Validation

La version la plus simple de la validation croisée consiste à diviser l'ensemble de données en deux sous-ensembles: ensemble de formation et ensemble de test. La formation du modèle se déroulerait sur le premier sous-ensemble de données.

Ensuite, les indicateurs de performance sont comparés en appliquant l'algorithme au premier et deuxième ensemble de données. Si les indicateurs trouvés pour l'ensemble d'apprentissage sont bien supérieurs à ceux trouvés sur l'ensemble de test, cela signifie que l'algorithme nécessite un surajustement des données qui serait nécessaire pour paramétrer le modèle et pour améliorer ses performances sur l'ensemble de test. Sinon, il est nécessaire de penser à augmenter la complexité du modèle afin d'obtenir une meilleure performance. La technique de validation croisée peut aller plus loin en divisant l'historique en k sous-ensembles également appelés repli et résultant d'une «validation croisée de k-fold». La formation se déroule sur un ensemble et on teste sur les k-1 ensembles restants, k fois. Nous comparons les moyennes des indicateurs sur les ensembles d'entraînement et de test pour voir si le modèle est surajouté.



Construction des modèles

Même les data scientists les plus expérimentés peuvent ne pas savoir quel algorithme fonctionnera le mieux avant d’essayer. Ainsi, la réponse courte à la question "Quel algorithme d'apprentissage automatique devons-nous utiliser? “ est toujours " Cela dépend de ".

En fait, le choix d'un algorithme d'apprentissage automatique dépend de plusieurs facteurs:

● La taille, la qualité et la nature des données.

● Objectifs commerciaux et d'exploration de données

● Les ressources (logiciels, matériels, humains)

● Le calendrier et la date limite

En outre, les aspects de l'algorithme eux-mêmes doivent être pris en considération. Dans ce qui suit, nous présenterons certains aspects importants que nous devons tenir en compte lors du choix du bon algorithme.

Aspects des algorithmes

**Précision**

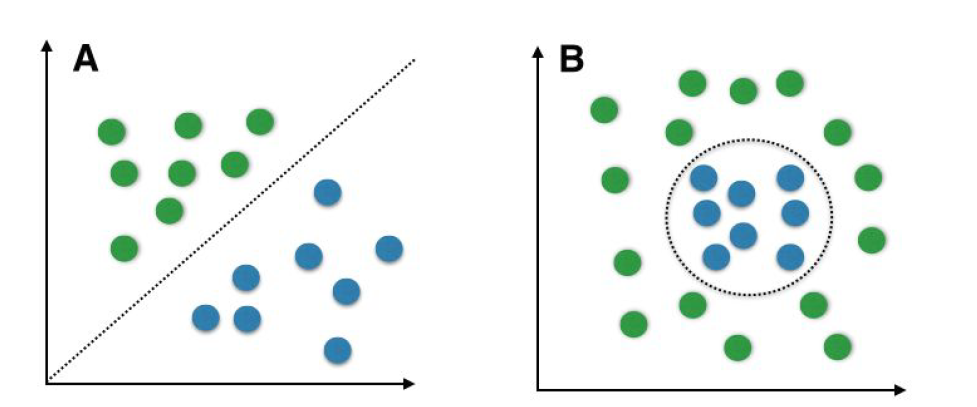
L'obtention du résultat le plus précis n'est pas toujours nécessaire. Parfois, selon notre utilisation, une approximation suffit. Si tel est le cas, vous pourrez peut-être réduire le temps de traitement considérablement en utilisant des méthodes plus approximatives. Un autre avantage des méthodes approximatives, c'est qu'elles ont naturellement tendance à éviter le surajustement.

**Temps d'apprentissage**

Le nombre de minutes ou d'heures nécessaires pour apprendre un modèle varie considérablement selon les algorithmes. La durée de l'apprentissage est souvent étroitement liée à la précision. De plus, certains algorithmes sont plus sensibles au nombre de points de données que d'autres. Un temps limité peut guider le choix de l'algorithme, en particulier lorsque l'ensemble de données est volumineux.

**Linéarité**

Un grand nombre d'algorithmes d'apprentissage automatique utilisent la linéarité. Les algorithmes de classification linéaire supposent que les classes peuvent être séparées par une ligne droite (ou son analogie dimensionnelle supérieure). Il s'agit notamment de la régression logistique et des machines vectorielles de support. Les algorithmes de régression linéaire supposent que les tendances des données suivent une ligne droite. Ces hypothèses ne sont pas fausses pour certains problèmes, mais réduisent la précision pour les autres. Dans la figure ci-dessous, nous montrons Linéaire (A) vs non linéaire (B):



Deux classes différentes sont représentées sous forme de cercles colorés et les lignes pointillées/cercle indiquent la classe limites que les classificateurs tentent de calculer. Le problème non linéaire (B) serait un cas où les classificateurs linéaires, tels que SVM linéaire ou Naïve Bayes, ne conviendraient pas car les classes sont non séparables linéairement. Dans un tel scénario, les classificateurs non linéaires tels que SVM ou Les classificateurs de l’arbre de Décision devraient être préférés.

**Nombre de paramètres**

Les paramètres sont les points d’entrée permettant de configurer un algorithme en affectant le comportement de l’algorithme, comme la tolérance d’erreur ou le nombre d’itérations ou de variantes du comportement de l’algorithme. Le temps d'apprentissage et la précision de l'algorithme peuvent parfois fortement dépendre du choix des paramètres appropriés.

En règle générale, les algorithmes avec un grand nombre de paramètres nécessitent plus de tests pour trouver la bonne combinaison. Bien que ce soit un excellent moyen de vous assurer que vous avez examiné l'espace des paramètres, le temps nécessaire pour former un modèle augmente de façon exponentielle avec le nombre de paramètres. Avoir de nombreux paramètres indique généralement qu'un algorithme a une plus grande flexibilité. Il peut souvent obtenir une excellente précision. Si vous trouvez la bonne combinaison de paramètres.

**Nombre de fonctionnalités**

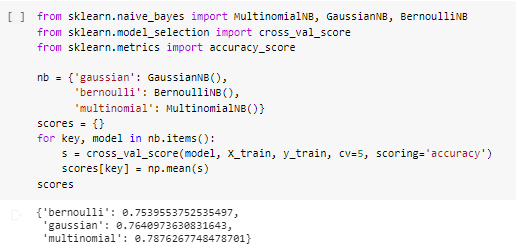
Pour certains types de données, le nombre d'entités peut être très important par rapport au nombre de points de données. C'est souvent le cas des données génétiques ou textuelles. Le grand nombre de fonctionnalités peut ralentir certains algorithmes d'apprentissage et entraîner un temps d'apprentissage inutilisable.

## Models training (Naive bayes)

In the following we evaluate In the following we evaluate 6 classification algorithms (Naive bayes , Random forest , Support vector machine , Logistic regression , Decision tree , k-nearest neighbor ) .

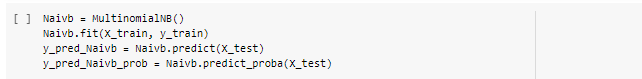
#### Naive Bayes Classifier

Le classifieur naïf bayésien est l'une des méthodes les plus simples d'apprentissage supervisé basées sur le théorème de Bayes. Naive Bayes n'est pas très utilisé par les data scientist par rapport aux méthodes traditionnelles telles que les arbres de décision ou les régressions logistiques. Un avantage de cette méthode est la simplicité de programmation, la facilité d'estimation des paramètres et sa rapidité (même sur de très grandes bases de données). Malgré ses avantages, Naive Bayes peu utilisée dans la pratique vient en partie du fait que sans un modèle explicitement simple (l'explication de la probabilité conditionnelle a priori), l'intérêt pratique d'une telle technique peut ne plus être pertinent.



Dans ce qui suit, nous appliquons la classe sklearn.naive\_bayes. MultinomialNB sur notre data frame que nous avons déjà préparée dans le chapitre précédent.

Ajustement du modèle par défaut aux données d'apprentissage



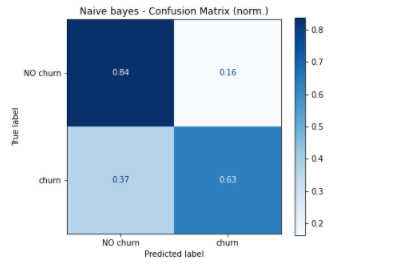
### evaluation

Nous évaluons le modèle formé avec la précision et la matrice de confusion expliquées précédemment.

Évaluation du modèle formé sur les données de test.



#### confusion matrix

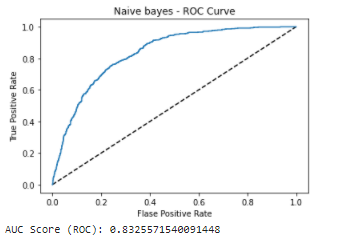


Bien que la précision soit supérieure à 78%, il est assez clair que le modèle peut parfaitement gérer les problèmes de données car nous avons clairement une matrice de confusion diagonale en ce qui concerne les enregistrements correctement classés.

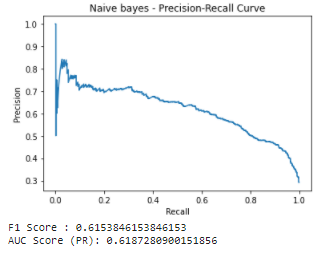
#### **ROC curve**

An **ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters :

* True Positive Rate
* False Positive Rate



L'AUC fournit une mesure globale des performances pour tous les seuils de classification possibles. Une façon d'interpréter l'AUC est la probabilité que le modèle classe un exemple positif aléatoire plus haut qu'un exemple négatif aléatoire.



* La courbe precision-recall curve montre le compromis entre précision et rappel pour différents seuils. Une zone élevée sous la courbe représente à la fois un recall élevé et une haute précision,
* une haute précision est liée à un faible taux de faux positifs et un rappel élevé correspond à un faible taux de faux négatifs
* . Des scores élevés pour les deux montrent que le classificateur renvoie des résultats précis (haute précision), ainsi que la majorité de tous les résultats positifs.

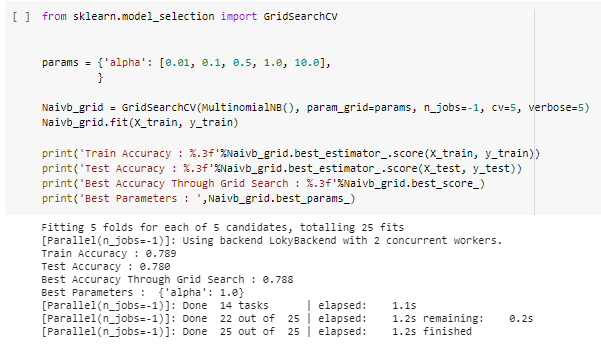
### optimisation du naive bayes

Ajuster le modèle en effectuant une recherche de grille sur divers hyperparamètres.

Vous trouverez ci-dessous une liste d'hyperparamètres courants qui doivent être ajustés pour obtenir le meilleur ajustement pour nos données. Nous allons essayer différents paramètres d'hyperparamètres pour différentes divisions de données de train / test pour trouver le meilleur ajustement qui aura presque la même précision pour l'ensemble de données de train et de test ou qui aura une différence bien moindre entre la précision.

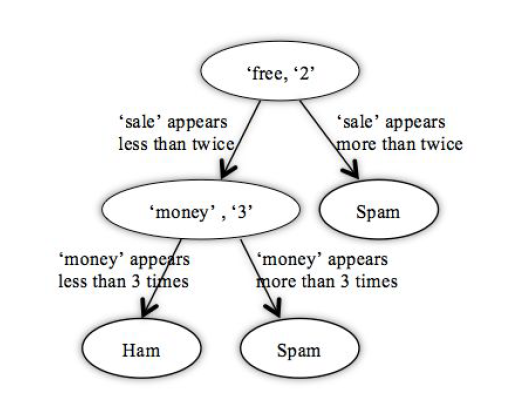
* **alpha** - It accepts float value representing the additive smoothing parameter. The value of 0.0 represents no smoothing. The default value of this parameter is 1.0.

Nous allons ci-dessous essayer différentes valeurs pour les hyperparamètres mentionnés ci-dessus afin de trouver le meilleur estimateur pour notre ensemble de données en fractionnant les données en 5 fold cross validation.

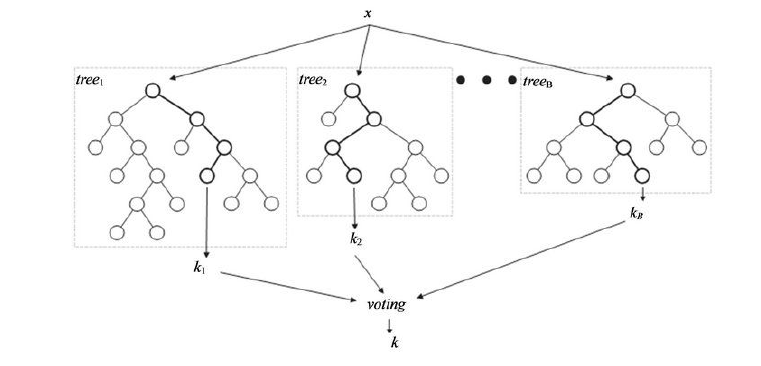


## Models training (**Classificateur de Random Forest**)

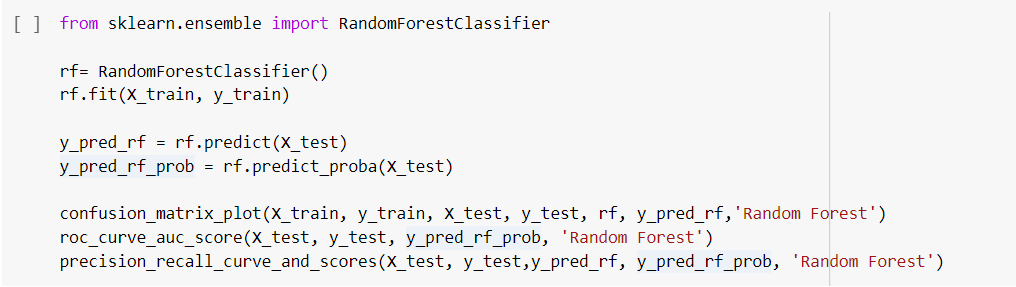
Les arbres de décision sont l'une des principales structures de données de l'apprentissage automatique. Leur fonctionnement repose sur des heuristiques qui, tout en satisfaisant l'intuition, donnent des résultats remarquables en pratique (surtout lorsqu'ils sont utilisés dans des «forêts aléatoires»). Leur structure arborescente les rend également lisibles par l'homme, contrairement à d'autres approches où le prédicteur construit est une «boîte noire». Un arbre de décision modélise une hiérarchie de test sur les valeurs d'un ensemble de variables appelées attributs. À la fin de ces tests, le prédicteur produit une valeur numérique ou choisit un élément dans un ensemble discret de conclusions. On parle de régression dans le premier cas et de classification dans le second. Par exemple, l'arborescence de la figure ci-dessous décide si un texte est «spam» ou «ham» en fonction de la présence et des occurrences de certains mots-clés (Un problème de classification de texte classique similaire au nôtre)



Une forêt aléatoire est un ensemble d'arbres de décision, utilisant la méthode de bagging, elle prend un sous-ensemble aléatoire d'entités à partir des données et crée n arbres aléatoires à partir de chaque sous-ensemble. Les arbres sont regroupés à la fin.



Dans ce qui suit, on commence par importer la classe sklearn.ensemble.RandomForestClassifier et appliquer notre modèle sur notre data frame que nous avons déjà préparée dans le chapitre précédent.



Évaluation

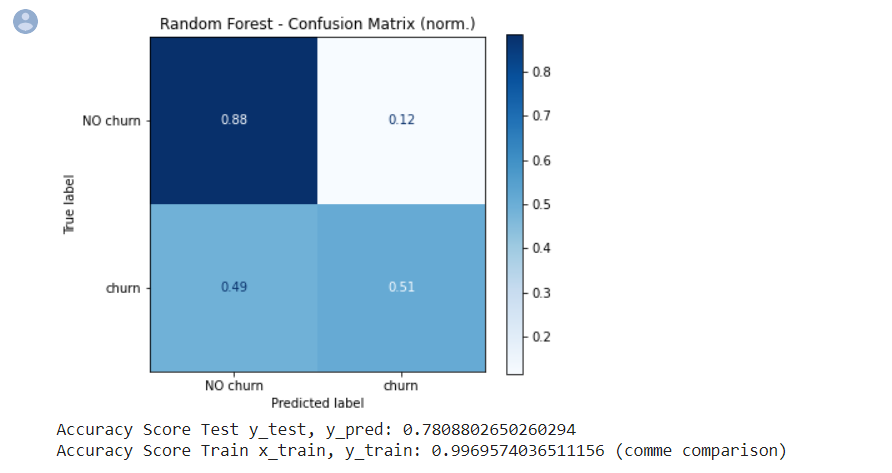
#### confusion matrix

On va maintenant calculer le score d’apprentissage / test en utilisant la matrice de confusion :

On va appeler la fonction de la matrice de confusion et interpréter le résultat :



Comme on peut le voir dans la figure ci-dessous , le score de précision de l’entraînement est presque égal à 99% ce qui est une indice sur le phénomène de sur-apprentissage (overfitting), On doit alors essayer de fixer ce problème ultérieurement .



Bien que la précision soit supérieure à 99%, il est assez clair que la précision du classificateur de forêt aléatoire est incroyablement élevée, et enfin, nous avons une matrice de confusion diagonale avec des enregistrements correctement classés.



Nous avons 4 valeurs tirées de la matrice ci-dessus :

1. **TP** : notre modèle a prédit que le client ne va pas se désabonner et la prédiction était correcte : 0.88 ( valeur normalisée )
2. **TN** : notre modèle a prédit que le client va se désabonner et la prédiction était correcte : 0.51
3. **FP** : notre modèle a prédit que le client ne va pas se désabonner mais la prédiction était fausse : 0.49
4. **FN** : notre modèle a prédit que le client va se désabonner mais la prédiction était fausse : 0.12

* On remarque que les dangereuses valeurs sont celles de **FP=0.49** et **FN=0.12** donc notre but est de nous concentrer sur ces derniers. d'où on doit améliorer surtout le Recall (doit être très proche de 1).

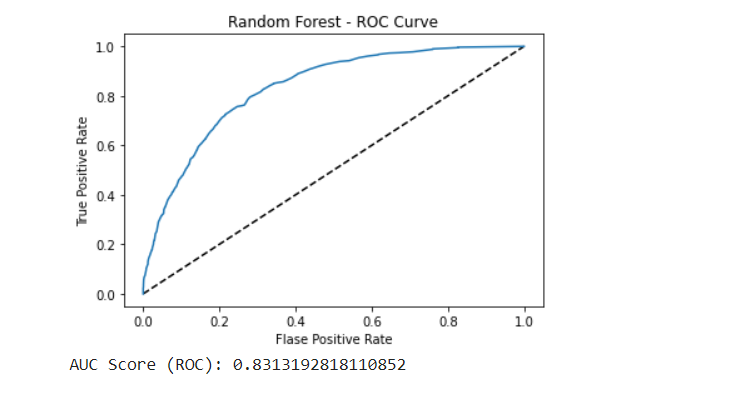
On va encore évaluer notre modèle mais cette fois en traçant **la courbe de ROC** :



#### **ROC curve**

Une courbe ROC (courbe caractéristique de fonctionnement du récepteur) est un graphique montrant les performances d'un modèle de classification à tous les seuils de classification. Cette courbe trace deux paramètres:

* **Taux de vrais positifs** = ( TP / (TP + FN ) )
* **Taux de faux négatifs** = ( FP/ ( FP + TN ) )



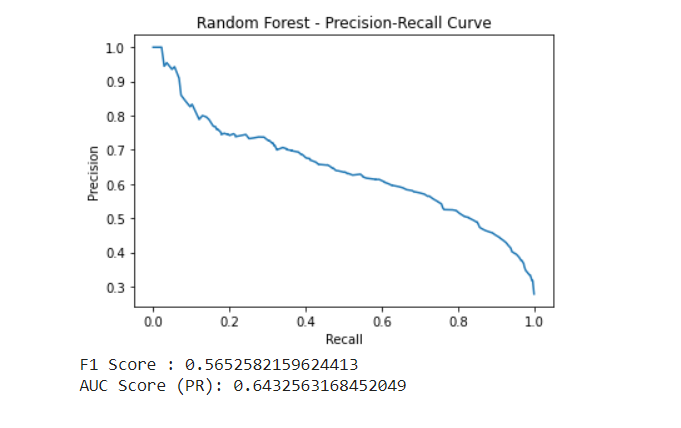
L'AUC fournit une mesure globale des performances pour tous les seuils de classification possibles. Une façon d'interpréter l'AUC est la probabilité que le modèle classe un exemple positif aléatoire plus haut qu'un exemple négatif aléatoire.

On remarque que le score de l’AUC est égal à **0.83** ,Cette valeur sera utilisée ultérieurement pour être comparé avec la valeur qu’on trouvera dans le modèle amélioré

On va encore évaluer notre modèle mais cette fois en traçant **la courbe precision-Recall Curve** :



##### ***Precision-Recall Curve***



La courbe precision-recall curve montre le compromis entre précision et rappel pour différents seuils. Une zone élevée sous la courbe représente à la fois un rappel élevé et une haute précision, où une haute précision est liée à un faible taux de faux positifs et un rappel élevé correspond à un faible taux de faux négatifs. Des scores élevés pour les deux montrent que le classificateur renvoie des résultats précis (haute précision), ainsi que la majorité de tous les résultats positifs.

**Recall** = ( TP / ( TP + FN ) )

**Précision** = ( TP / (TP + FP ) )

#### Optimisation de Random forest

**RandomizedSearchCV**

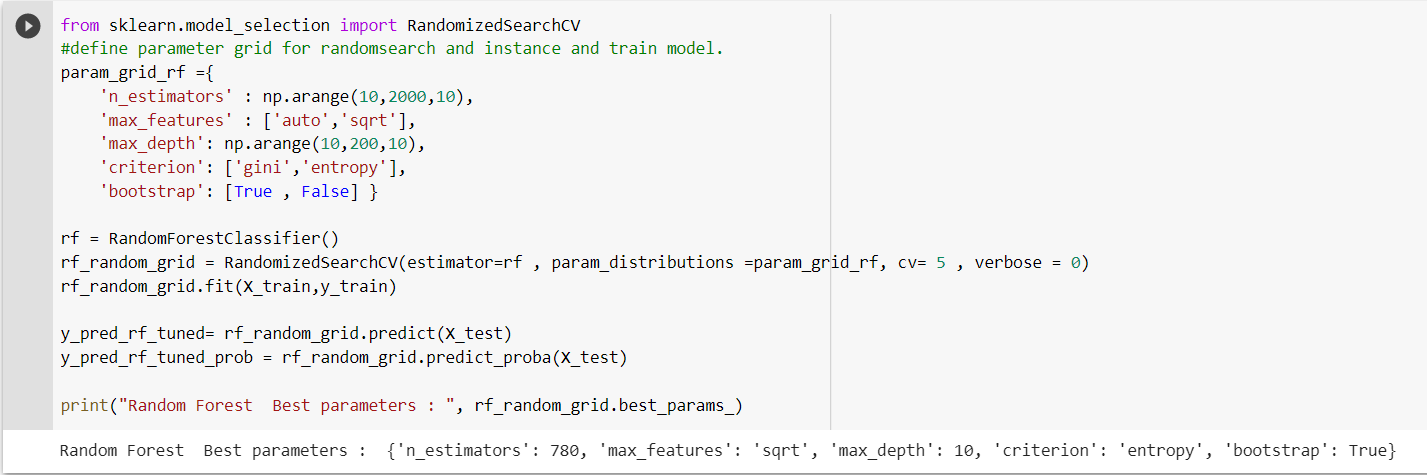
La recherche aléatoire remplace l'énumération exhaustive de toutes les combinaisons en les sélectionnant au hasard. Cela peut être simplement appliqué au réglage discret décrit ci-dessus, mais se généralise également aux espaces continus et mixtes

Un **hyperparamètre** de modèle est une caractéristique d'un modèle qui est externe au modèle et dont la valeur ne peut être estimée à partir de données. La valeur de l'hyperparamètre doit être définie avant le début du processus d'apprentissage .

Pour le modèle Random Forest, Randomized Search CV est utilisé pour optimiser plusieurs hyperparamètres, y compris n\_estimators, max\_features, max\_depth, critère et bootstrap

Dans ce qui suit, nous appliquons la classe sklearn.model\_selection.RandomizedSearchCV sur notre data frame que nous avons déjà préparée dans le chapitre précédent afin d’ajuster encore plus notre modèle.

-Nous allons entraîner notre modèle sur les sous-ensembles d’apprentissage .

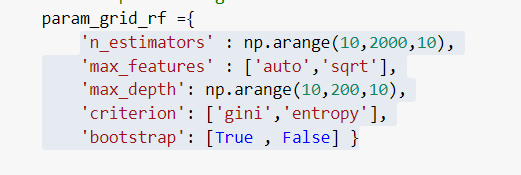


La pratique typique de la validation croisée consiste à segmenter à nouveau l'ensemble d'entraînement en sous-ensembles plus petits.

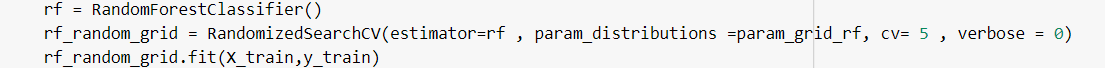
Nous allons ci-dessous essayer différentes valeurs pour les hyperparamètres mentionnés ci-dessus afin de trouver le meilleur estimateur pour notre ensemble de données en fractionnant les données en 5 fold cross validation.

**1/**on a affecté a nos hyperparamètres: (**n\_estimators**, **max\_features** ,**max\_depth**, **criterion**, **bootsrap** ) les valeurs respectives:

np.arange(10,2000,10),['auto','sqrt'],np.arange(10,200,10),['gini','entropy'],[True , False]



**2/**Pour notre cas, nous avons divisé en 5 sous-ensembles (Cv=5) (4 pour l'entraînement et 1 pour le score) estimés de manière similaire. Ensuite, nous recyclons le modèle sur quatre sections et calculons la précision sur le dernier jeu de validation.

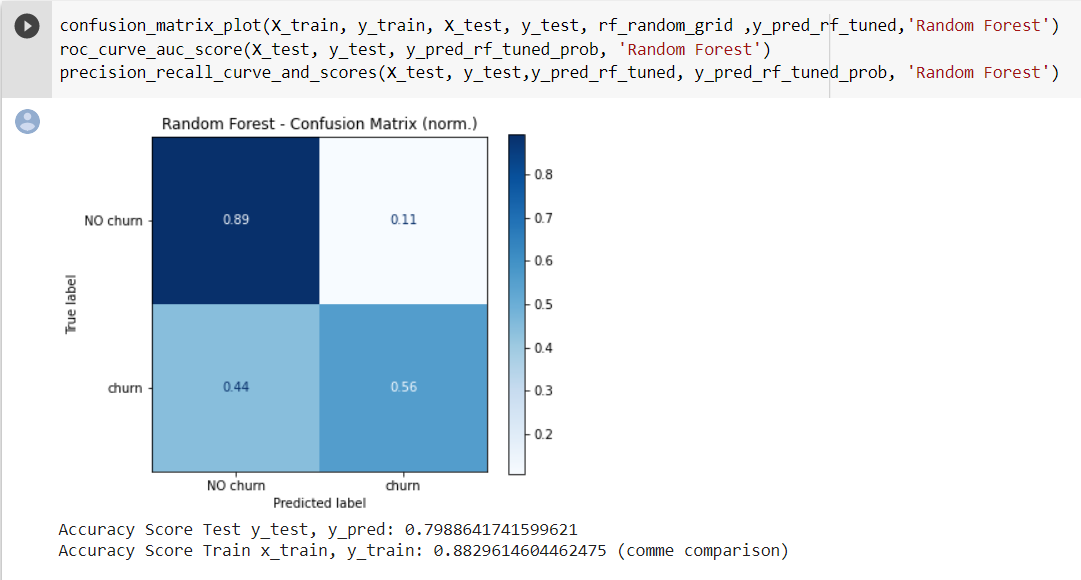


Évaluation

Afin d’essayer de régler le problème de (overfitting) nous évaluons le modèle formé avec la matrice de confusion expliquées précédemment, ROC et AUC .

#### confusion matrix

On va maintenant calculer le score d’apprentissage / test en utilisant la matrice de confusion :



On constate que les valeurs prédites cette fois ont plus de chance d’être correctes , ceci est justifié par les résultats obtenus ci-dessous:

* La valeur de **TP** est égale à 0.89 alors que précédemment on a trouvé 0.88
* La valeur de **FP** est égale à 0.44 alors que précédemment on a trouvé 0.49 ( le modèle a moins de chance de prédire une valeur fausse )
* La valeur de **TN** est égale à 0.56 alors que précédemment on a trouvé 0.51
* La valeur de **FN** est égale à 0.11 alors que précédemment on a trouvé 0.12

Si on compare le résultat trouvé avec celui avant l’optimisation des hyperparamètres , on remarque qu’on n’a plus le problème de sur-apprentissage ce qui est un bon indice sur la qualité de notre modèle .



Comme on peut le voir dans la figure ci-dessus , le score de précision de l’entraînement est égal à **88%** ce qui est très raisonnable si on le compare avec le score obtenu durant la première évaluation (sans optimisation)

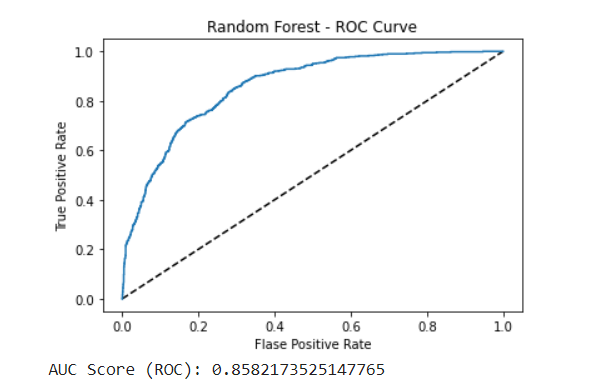
**conclusion**

que Les scores ne sont pas aussi parfaits que les **0.99** que nous avions sur l'ensemble de l'apprentissage, mais nous avons quand même un bon résultat de **0.88** (scores moyens)

On va encore évaluer notre modèle mais cette fois en traçant la **courbe de ROC** .

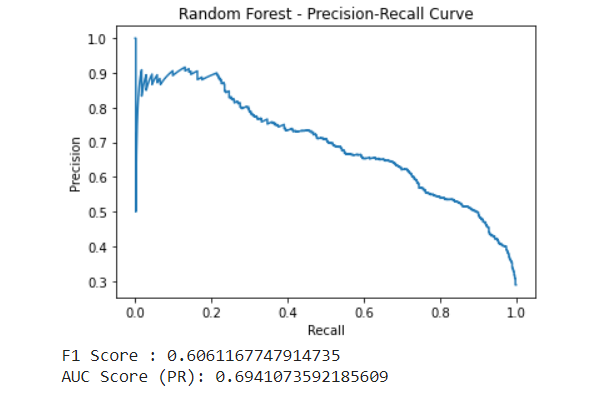
#### 

#### **ROC curve**



Le AUC Score trouvé cette fois est (0.858)**>** AUC Score qu’on a trouvé précédemment(0.831) !

##### ***Precision-Recall Curve***



Le AUC score trouvé ( 0.694 ) **>** AUC score trouvé précédemment ( 0.643) .

##### 

##### ***Conclusion***

l’amélioration du score de **précision** , **recall** , **f1-score** et **accuracy** est très satisfaisante, tous les résultats obtenus se sont remarquablement améliorés et ceci par conséquent grâce aux hyperparamètres .

|  |  |  |
| --- | --- | --- |
|  | Random Forest non optimisé | Random Forest avec hyperparamètres (**RandomizedSearchCV**) |
| AUC ROC | 0.831 | 0.858 |
| AUC PR | 0.643 | 0.694 |

## 

## 

## 

## Models training (K-Nearest Neighbor Classifier)

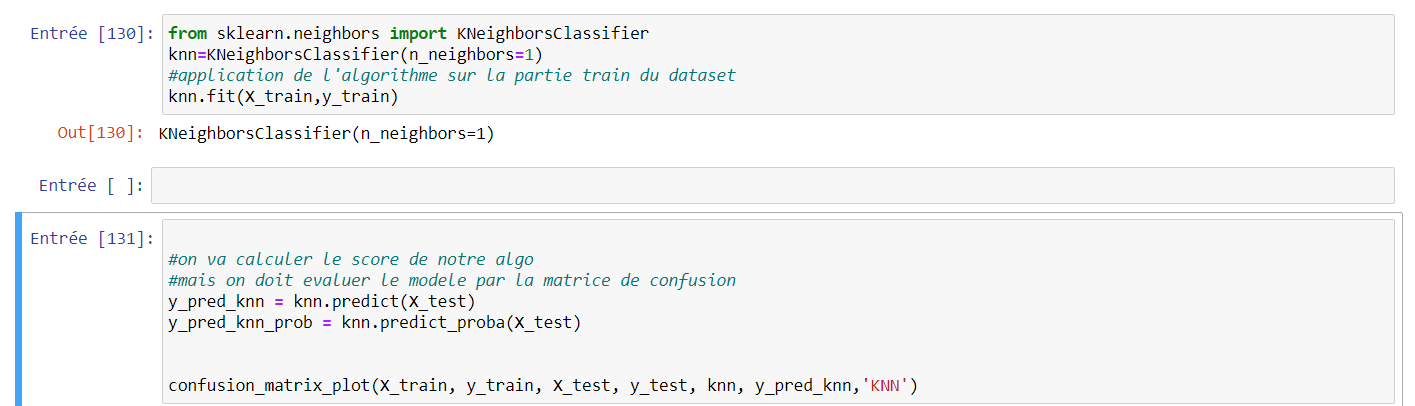
#### 

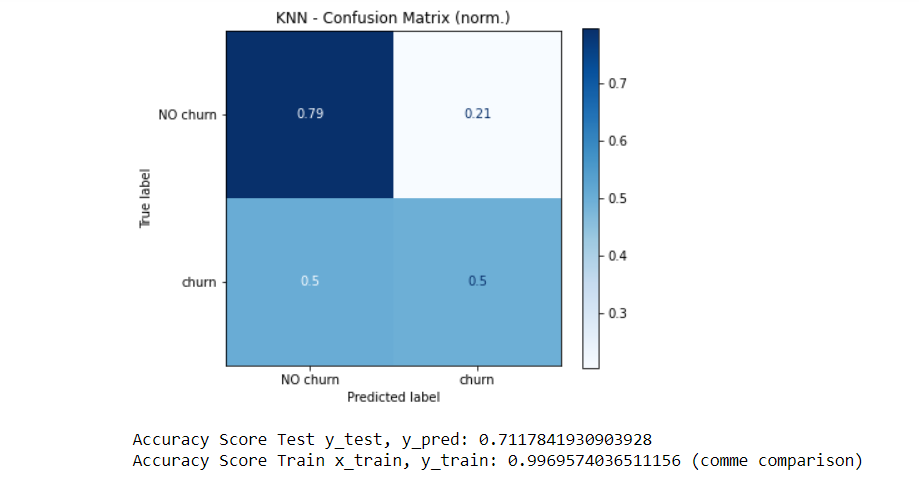
La méthode *k*-NN est basée sur l'apprentissage préalable, ou l'apprentissage faible, où la fonction est évaluée localement, le calcul définitif étant effectué à l'issue de la classification. L'algorithme *k*-NN est parmi les plus simples des algorithmes de machines learning.

Que ce soit pour la classification ou la régression, une technique efficace peut être utilisée pour pondérer l'influence contributive des voisinages, ainsi les plus proches voisins contribuent-ils plus à la moyenne que les voisins plus éloignés. Pour exemple, un schéma courant de pondération consiste à donner à chaque voisin une pondération de 1/*d*, ou *d* est la distance de l'élément, à classer ou à pondérer, de ce voisin.

Les voisins sont pris depuis un ensemble d'objets pour lesquels la classe (en classification k-NN) ou la valeur (pour une régression *k*-NN) est connue. Ceci peut être considéré comme l'ensemble d'entraînement pour l'algorithme, bien qu'un entraînement explicite ne soit pas particulièrement requis.

Dans ce qui suit, nous appliquons la classe sklearn.neighbors.KneighborsClassifier sur notre data frame que nous avons déjà préparée dans le chapitre précédent.





Comme on peut le voir dans la figure ci-dessous , le score de précision de l’entraînement est presque égal à 1 ( 100% ) ce qui est une indice sur le phénomène de sur-apprentissage ( overfitting ) . On doit alors essayer de fixer ce problème ultérieurement .

Nous avons 4 valeurs dans notre matrice :

1- TP : notre modèle a prédit que le client ne va pas se désabonner et la prédiction était d correcte : 0.79 ( valeur normalisée )

2- TN : notre modèle a prédit que le client va se désabonner et la prédiction était correcte : 0.5

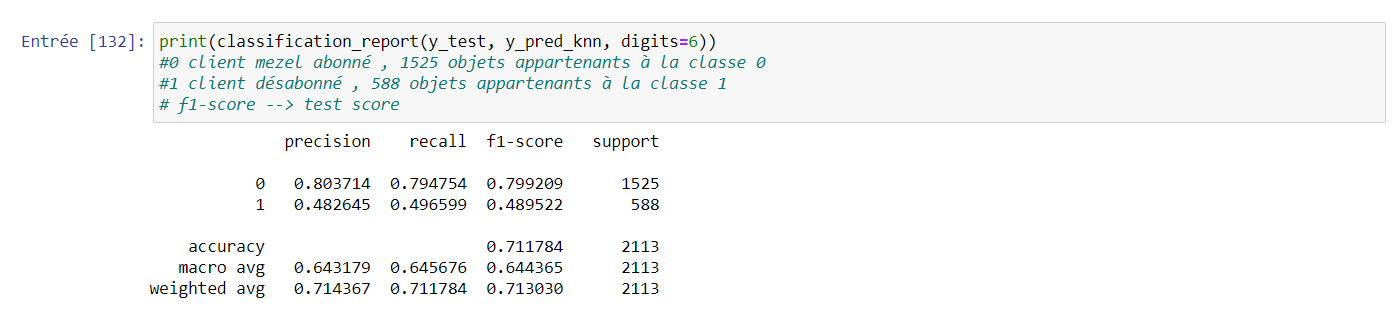
3- FP : notre modèle a prédit que le client ne va pas se désabonner mais la prédiction était fausse : 0.5

4- FN : notre modèle a prédit que le client va se désabonner mais la prédiction était

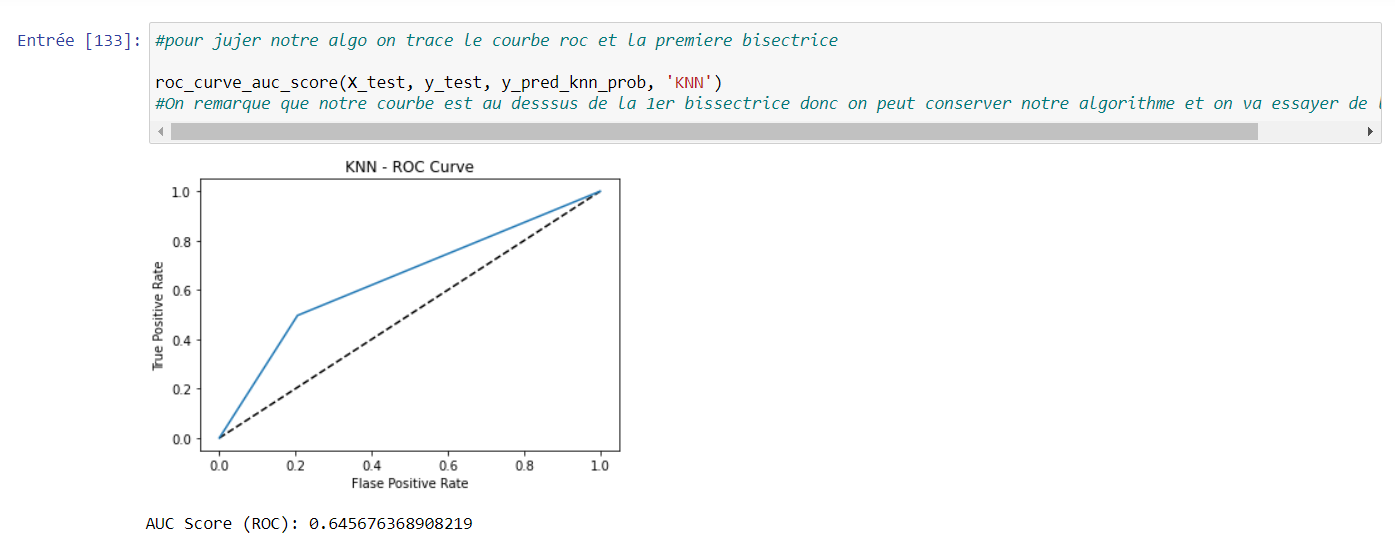
fausse : 0.21

* On remarque que les dangereuses valeurs sont celles de FP=0.5 et FN=0.21 donc notre but est de nous concentrer sur ces derniers. d'où on doit améliorer surtout le Recall (doit être très proche de 1).

Nous évaluons le modèle formé avec la précision et la matrice de confusion expliquées précédemment.



On va encore évaluer notre modèle mais cette fois en traçant la courbe de ROC :

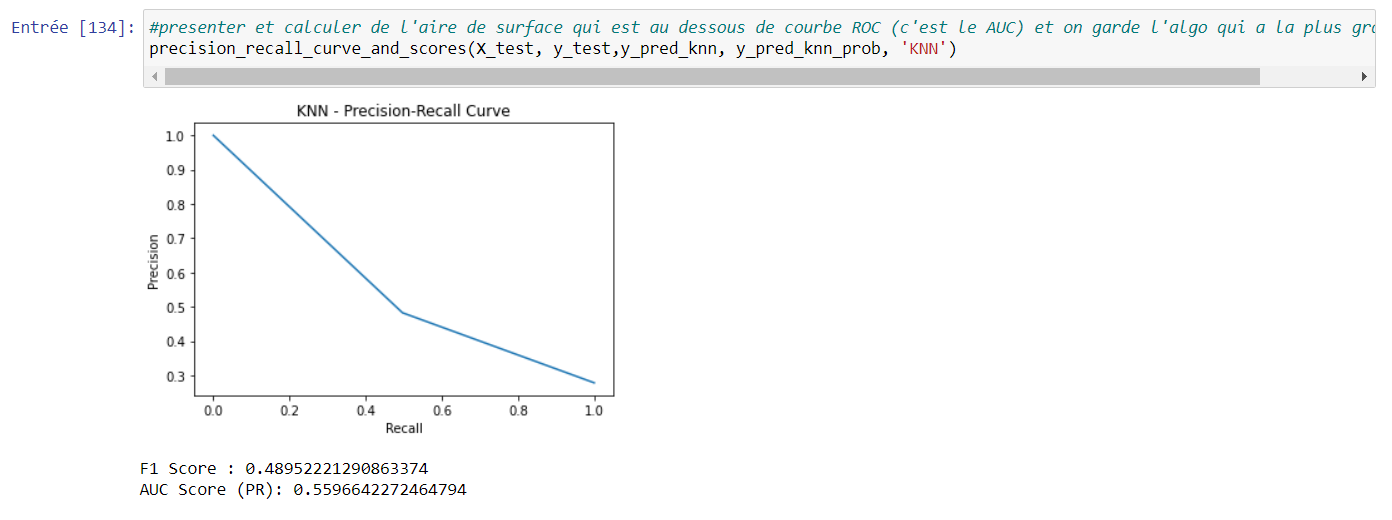


Une courbe ROC (courbe caractéristique de fonctionnement du récepteur) est un graphique montrant la performance d'un modèle de classification à tous les seuils de classification.

* Taux de vrais positifs
* Taux de faux négatifs

L'AUC fournit une mesure globale des performances pour tous les seuils de classification possibles. Une façon d'interpréter l'AUC est la probabilité que le modèle classe un exemple positif aléatoire plus haut qu'un exemple négatif aléatoire.

##### ***Precision-Recall Curve:***



La courbe precision-recall curve montre le compromis entre précision et rappel pour différents seuils. Une zone élevée sous la courbe représente à la fois un rappel élevé et une haute précision, où une haute précision est liée à un faible taux de faux positifs et un rappel élevé correspond à un faible taux de faux négatifs. Des scores élevés pour les deux montrent que le classificateur renvoie des résultats précis (haute précision), ainsi que la majorité de tous les résultats positifs.

#### Optimisation de KNN :

**Optimisation avec GridSearchCV**

GridsearchCV nous permet de trouver le modèle avec les meilleurs hyper-paramètres en comparant les différentes performances de chaque combinaison grâce à la technique de cross-validation(encore une fois). Donc on importe GridserchCV depuis le module model\_selection, ensuite on crée un dictionnaire qui contient les différents hyper-paramètres à régler ainsi que chaque valeur à tester pour ses hyper-paramètres. Ensuite on fait passer notre type de modèle dans la fonction GriSerchCV ainsi que le dictionnaire que l’on vient de créer et un nombre pour notre cross\_validation par exemple cv=5. Au final on a une grille qui contient plusieurs estimateurs. On va donc entraîner cette grille avec la méthode fit comme s’il s’agissait d’un estimateur en passant bien évidement les données du train set.

La figure ci-dessous nous montre que les meilleurs paramètres trouvés sont :

* Utilisation de la distance Euclidienne
* Le nombre de voisins k est de 12 .



On remarque dans la figure ci dessous la matrice de confusion avec le nouveau paramètre k=12 que :

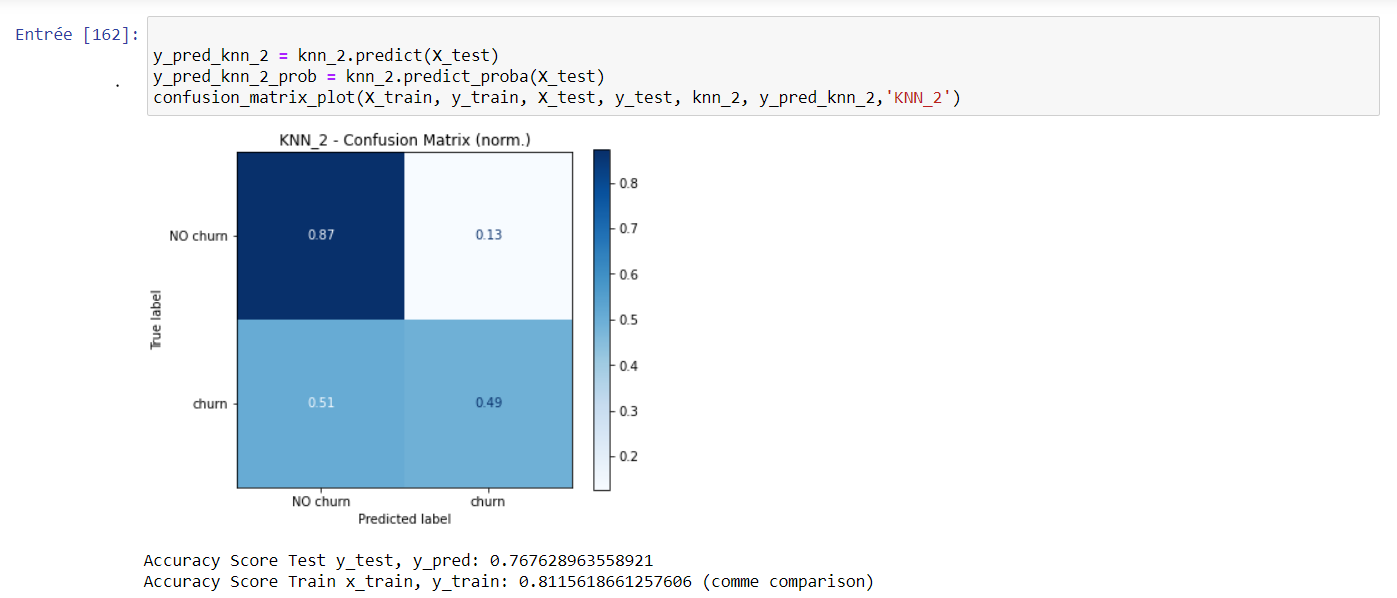
1- TP : notre modèle a prédit que le client ne va pas se désabonner et la prédiction était correcte : 0.87 ( valeur normalisée )

2- TN : notre modèle a prédit que le client va se désabonner et la prédiction était correcte : 0.49

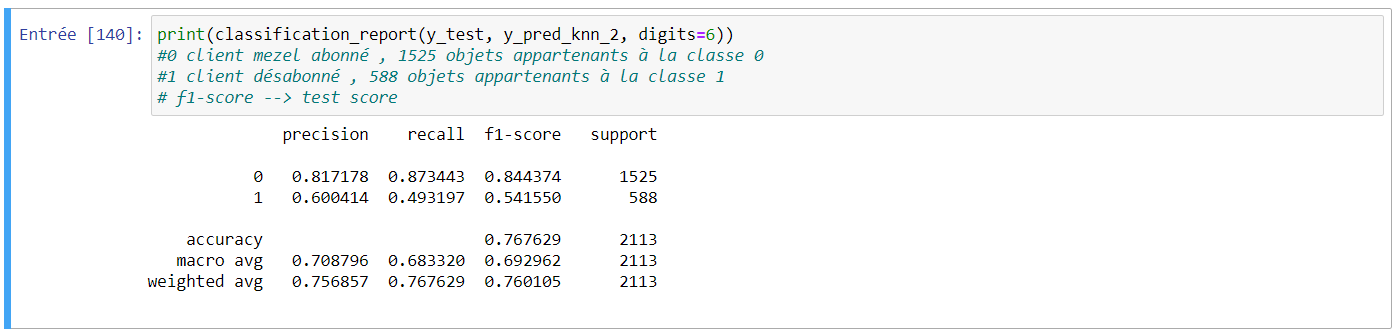
3- FP : notre modèle a prédit que le client ne va pas se désabonner mais la prédiction était fausse : 0.51

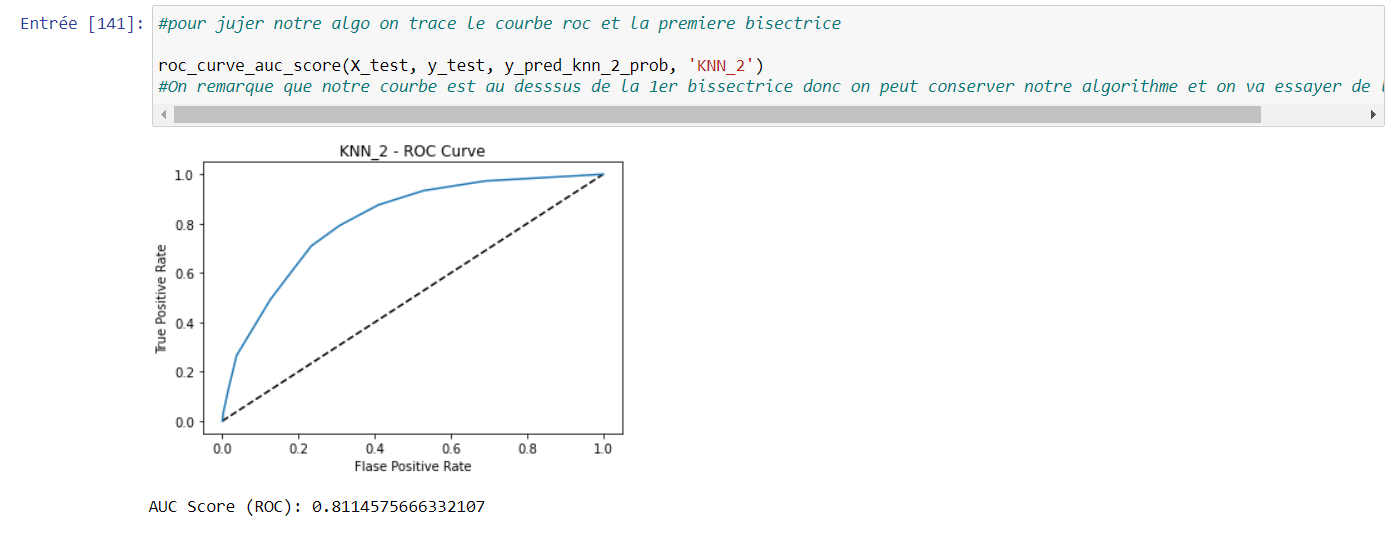
4- FN : notre modèle a prédit que le client va se désabonner mais la prédiction était

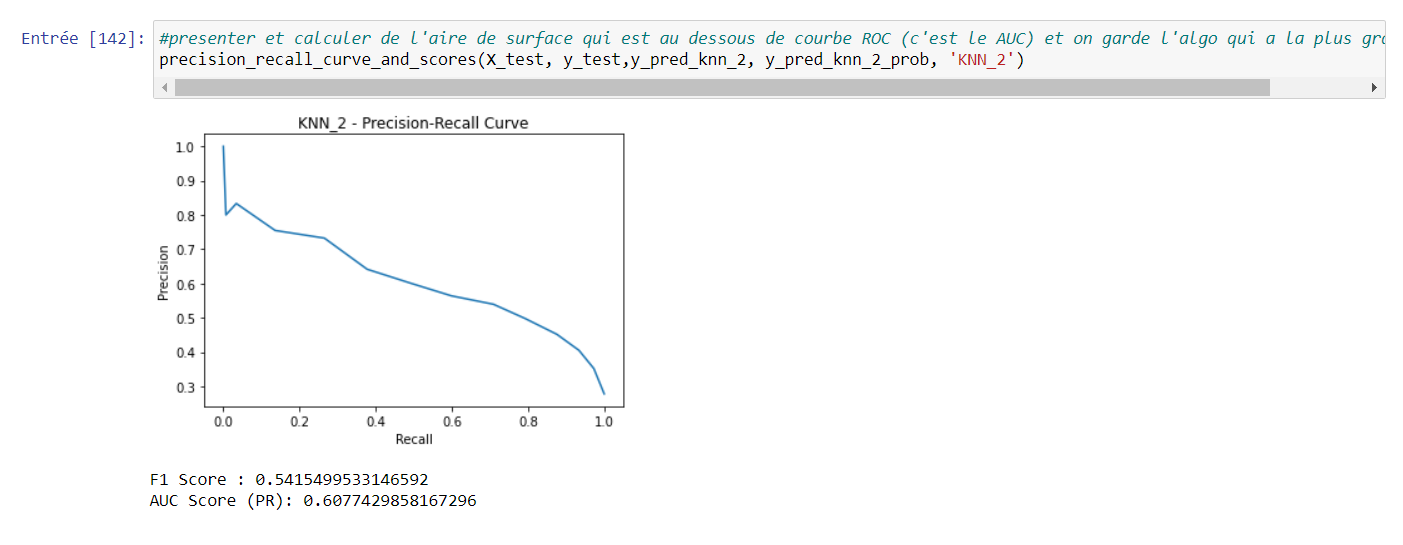
fausse : 0.13



Comme le montre la figure ci-dessous, la précision de l’algorithme à augmenté pour atteindre 76%.







**Optimisation avec RandomizedSearchCV**



On remarque dans la figure ci dessous la matrice de confusion avec le nouveau paramètre k=12 que :

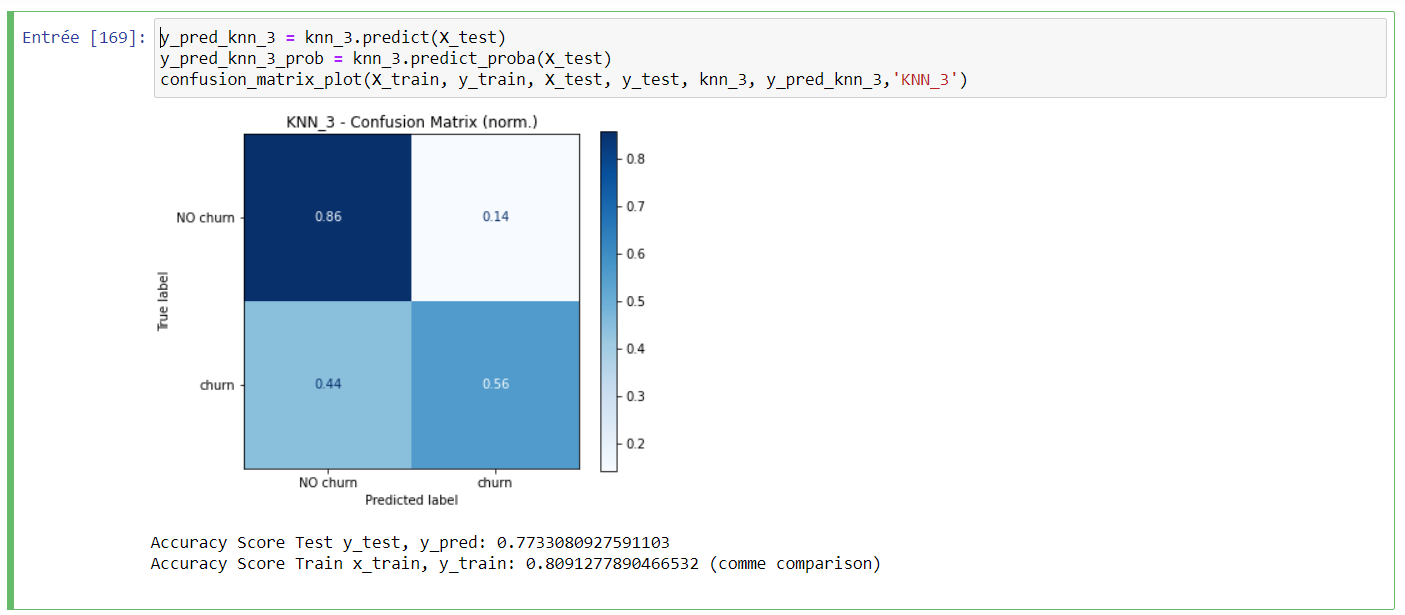
1- TP : notre modèle a prédit que le client ne va pas se désabonner et la prédiction était correcte : 0.86 ( valeur normalisée )

2- TN : notre modèle a prédit que le client va se désabonner et la prédiction était correcte : 0.56

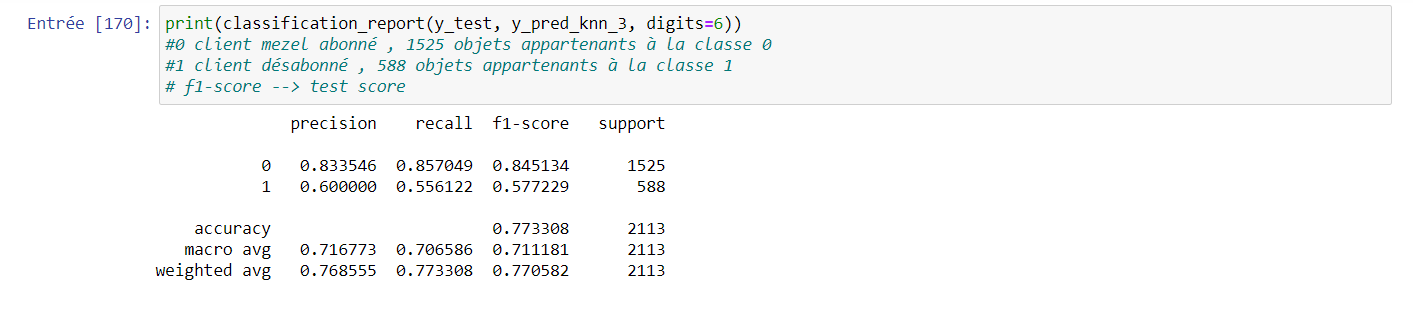
3- FP : notre modèle a prédit que le client ne va pas se désabonner mais la prédiction était fausse : 0.44

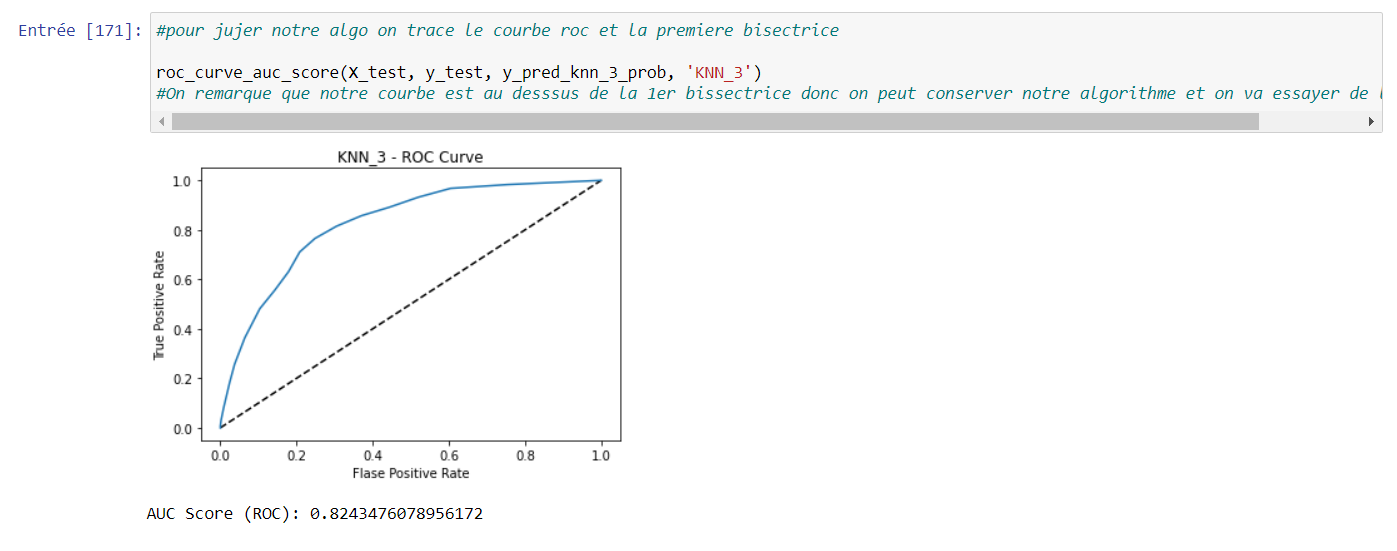
4- FN : notre modèle a prédit que le client va se désabonner mais la prédiction était

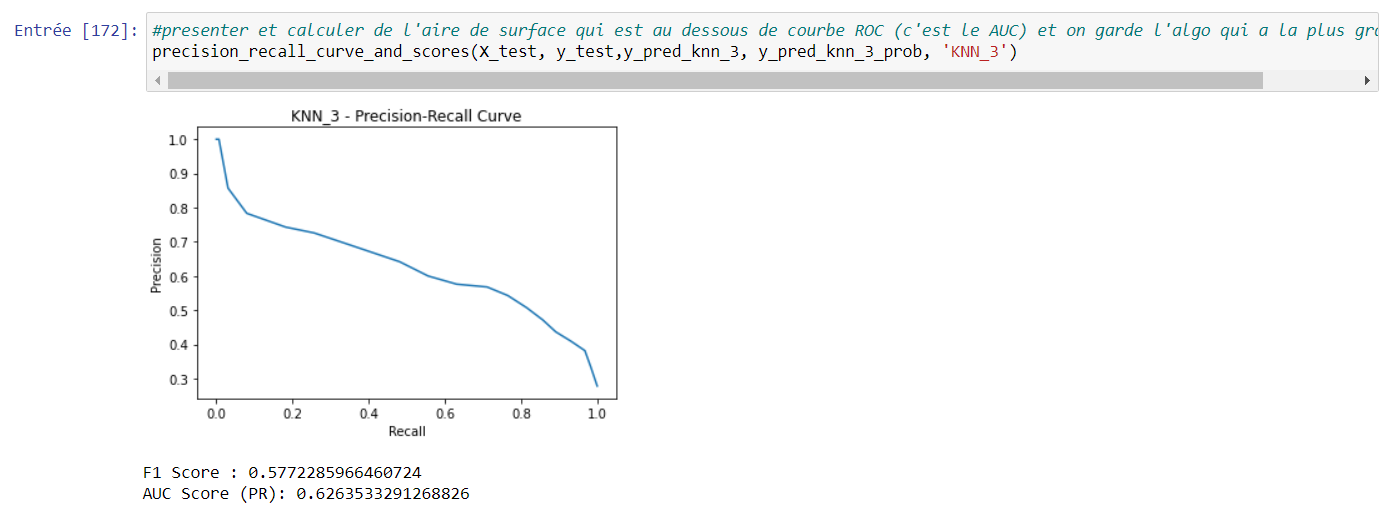
fausse : 0.14



Comme le montre la figure ci-dessous, la précision de l’algorithme à augmenté par rapport à k=12 pour atteindre 77%.







|  |  |  |  |
| --- | --- | --- | --- |
| AUC\* ROC\* | 0.64 (k=1) | 0.81 ( k=12) | 0.82 (k=18) |
| AUC PRC\* | 0.55(k=1) | 0.60( k=12) | 0.62(k=18) |

AUC = area under curve ( aire de la surface sous la courbe)

ROC =receiver operating characteristic ( fonction d’efficacité du récepteur)

PRC = precision recall curve (courbe précision-rappel)

L’AUC pour les ROC et PRC sont maximales pour le nombre de voisins k=18. On peut en conclure que l’algorithme est plus précis pour k=18 que pour k=12 ou k=1.

## 

## Models training (LOGISTIC REGRESSION Classifier)

La régression logistique est un modèle statistique utilisé pour déterminer si une variable indépendante a un effet sur une variable dépendante binaire. Cela signifie qu'il n'y a que deux résultats potentiels pour une contribution. Par exemple, il peut être utilisé pour déterminer si un e-mail est du spam ou non, en utilisant le taux de mots mal orthographiés, un signe courant de spam. D'autres formes d'analyse de régression, comme une régression linéaire, nécessitent la définition d'un seuil pour distinguer les classes binaires (par exemple <50% mal orthographié = pas de spam,> 50% mal orthographié = spam). La régression linéaire permet d'établir une probabilité, mais elle doit ensuite être appliquée à une régression logistique pour faire la classification distincte.

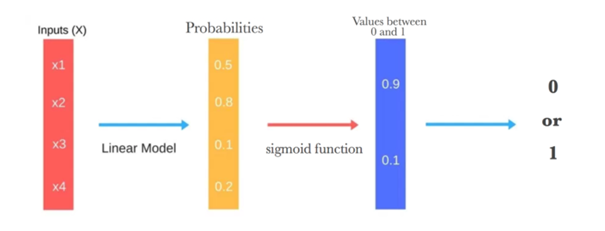
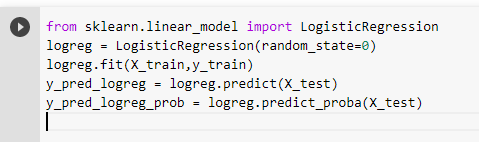


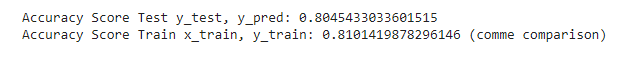
Figure : Logistic regression

Donc, fondamentalement, nous avons un vecteur de caractéristiques à valeur réelle et en appliquant une fonction sigmoïde, nous avons découvert des valeurs entre 0 et 1 qui sont la probabilité que y = 1 connaissant la valeur des autres variables (P (y = 1 / x)) . Notez que y est la variable cible et prend des valeurs binaires.

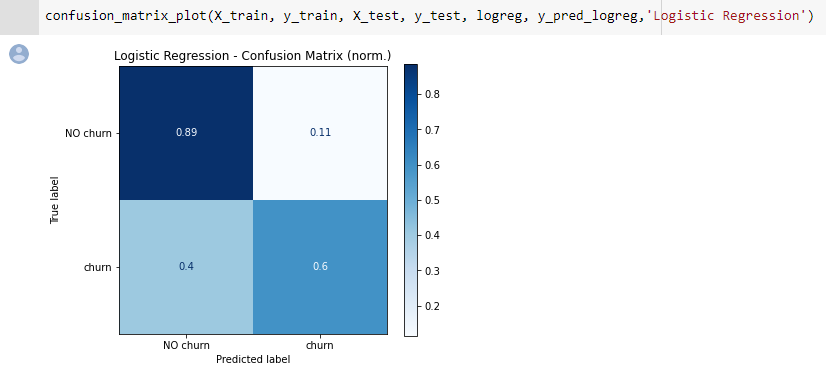
Dans ce qui suit, nous appliquons la classe sklearn.LogisticRegression sur notre data frame que nous avons déjà préparée dans le chapitre précédent. 

Évaluation

Nous évaluons le modèle formé avec la précision , la matrice de confusion expliquées précédemment, ROC et AUC .



Lors de la première prédiction, il avait un score d'entraînement de plus de 0.81 et un score de test de plus de 0.8%. Il est évident que le score du test ne présente pas le valeur optimal , il faut donc regarder la matrice de confusion pour mieux comprendre la fiabilité et les limites de ce modèle selon notre cas.



Selon la matrice de confusion, nous avons 4 valeurs qui sont:

TP = nous avons prédit que le client ne se désabonnerait pas et c'est vrai: 0.89

TN = nous avons prédit que le client se désabonnerait et c'est vrai: 0.6

FP = nous avons prédit qu'il se désinscrire mais il restera: 0.11

FN = nous avons prédit qu'il restera mais il se désinscrire: 0.4

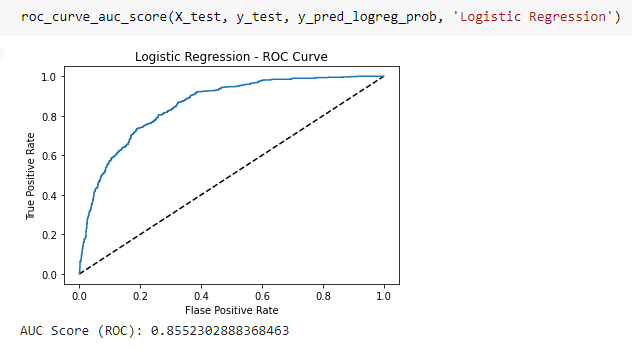
* On remarque que les dangereuses valeurs sont celles de FN et FP et FN et plus dangereuse donc notre but est de nous concentrer sur ces derniers. d'où on doit améliorer surtout la métrique de rappel (doit être très proche de 1).

#### 

##### ***ROC Curve***

Une courbe ROC (courbe caractéristique de fonctionnement du récepteur) est un graphique montrant la performance d'un modèle de classification à tous les seuils de classification.

* Taux de vrais positifs
* Taux de faux négatifs



L'AUC fournit une mesure globale des performances pour tous les seuils de classification possibles. Une façon d'interpréter l'AUC est la probabilité que le modèle classe un exemple positif aléatoire plus haut qu'un exemple négatif aléatoire .

On remarque que notre courbe est au dessus de la 1er bissectrice donc on peut conserver notre algorithme et on va essayer de l'améliorer

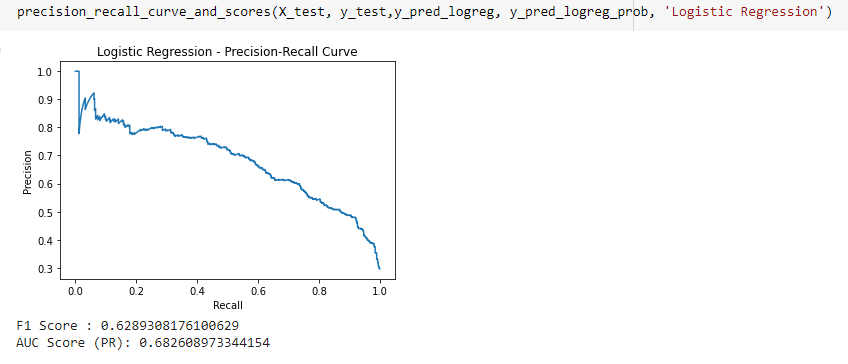
##### ***Precision-Recall Curve***

La courbe precision-recall curve montre le compromis entre précision et rappel pour différents seuils. Une zone élevée sous la courbe représente à la fois un rappel élevé et une haute précision, où une haute précision est liée à un faible taux de faux positifs et un rappel élevé correspond à un faible taux de faux négatifs. Des scores élevés pour les deux montrent que le classificateur renvoie des résultats précis (haute précision), ainsi que la majorité de tous les résultats positifs.

résultats positifs.

Recall = ( Vrais Positifs / ( vrais positifs + faux négatifs ) )

Précision = ( Vrais positifs / (Vrais Positifs + faux positifs ) )

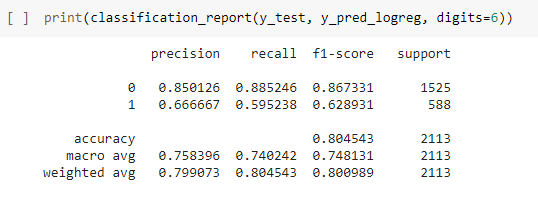


##### 

La courbe precision-recall curve montre le compromis entre précision et rappel pour différents seuils. Une zone élevée sous la courbe représente à la fois un rappel élevé et une haute précision, où une haute précision est liée à un faible taux de faux positifs et un rappel élevé correspond à un faible taux de faux négatifs. Des scores élevés pour les deux montrent que le classificateur renvoie des résultats précis (haute précision), ainsi que la majorité de tous les résultats positifs.

##### ***Rapport de Classification***

On va appeler la classe sklearn.matrics.classification report pour voir le rapport sur la précision , recall , f1-score et accuracy par classe ( 0 et 1 ) , On cherche à améliorer ces valeurs dans les étapes qui suivent .



* pour un taille de données 1525 de classe 0 (no churn) le modèle a une précision de 0.85 , un score test 0.867331
* pour un taille de données 588 de classe 1 (churn) la précision de modèle égale à 0.666.
* On va conserver notre algorithme et on va essayer de l'améliorer

### Optimisation de LOGISTIC REGRESSION Classifier :

Pour le modèle LOGISTIC REGRESSION ,ON va utilisé pour optimiser plusieurs hyperparamètres, y compris penalty, C, solver, max\_iter,

* penalty : {"l1", "l2", "elasticnet", "none"}, default = "l2" . Utilisé pour spécifier la norme utilisée dans la pénalisation. Les solveurs «newton-cg», «sag» et «lbfgs» ne prennent en charge que les pénalités de 12. ‘Elasticnet’ n’est pris en charge que par le solveur 'saga'. Si "aucun" (non pris en charge par le solveur liblinar), aucune régularisation n'est appliquée.
* C : par défaut = 1.0 Inverse de la force de régularisation, doit être un flottant positif.
* solver {‘newton-cg’, ‘lbfgs’, ‘liblinear’, ‘sag’, ‘saga’}, default = ’lbfgs’ Algorithme à utiliser dans le problème d'optimisation.

• Pour les petits ensembles de données, «liblinear» est un bon choix, tandis que «sag» et «saga» sont plus rapides pour les grands.

• Pour les problèmes multi classes, seuls «newton-cg», «sag», «saga» et «lbfgs» gèrent la perte multinomiale; «Liblinear» est limité aux régimes «un contre repos».

• «newton-cg», «lbfgs», «sag» et «saga» gèrent L2 ou pas de pénalité

• «liblinear» et «saga» gèrent également les pénalités de L1

• «saga» prend également en charge la pénalité «élastiquenet»

• "liblinear" ne prend pas en charge la définition de pénalité = "aucun"

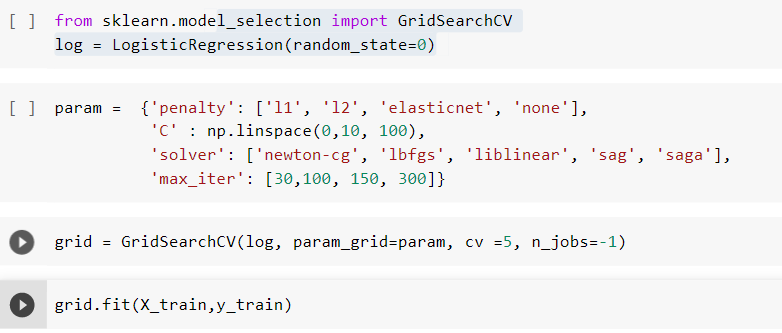
* max\_iter, par défaut = 100 , c’est le Nombre maximum d'itérations nécessaires pour que les solveurs convergent.

##### Logistic\_Regression with hyperparameter (GridSearchCV) :

La recherche aléatoire remplace l'énumération exhaustive de toutes les combinaisons en les sélectionnant au hasard. Cela peut être simplement appliqué au réglage discret décrit ci-dessus, mais se généralise également aux espaces continus et mixtes

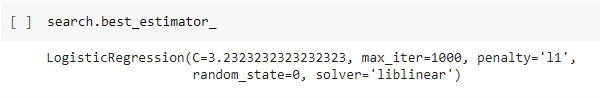
Pour le modèle Random Forest, GridSearchCV est utilisé pour optimiser plusieurs hyperparamètres, y compris n\_estimators, max\_features, max\_depth, critère et bootstrap

Dans ce qui suit, nous appliquons la classe sklearn.model\_selection.GridSearchCV sur notre data frame que nous avons déjà préparée dans le chapitre précédent.

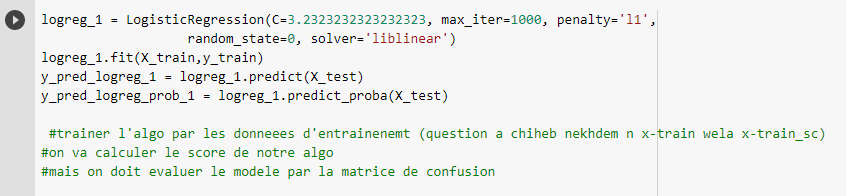


La figure ci-dessous nous montre que les meilleurs paramètres trouvés sont :

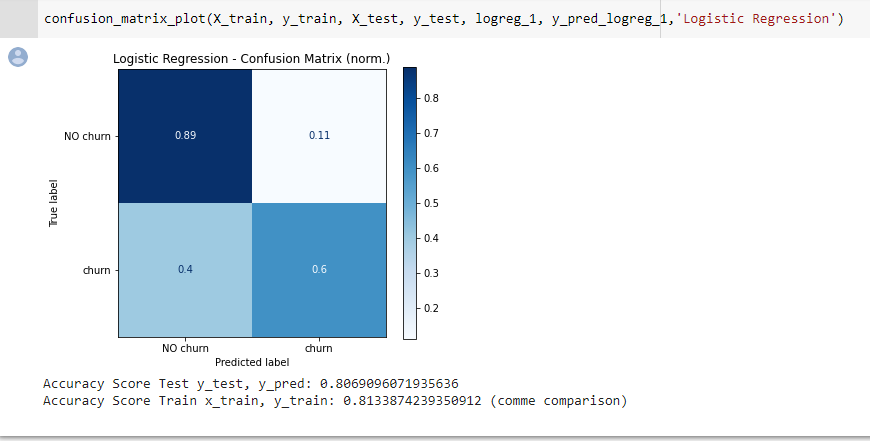
* C=3.2323232323232323, max\_iter=1000, penalty='l1', random\_state=0, solver='liblinear'



On va maintenant appeler la classe **Logistic Regression** mais cette fois ci en utilisant les meilleurs paramètres qu’on a trouvé à l’aide de GridSearch

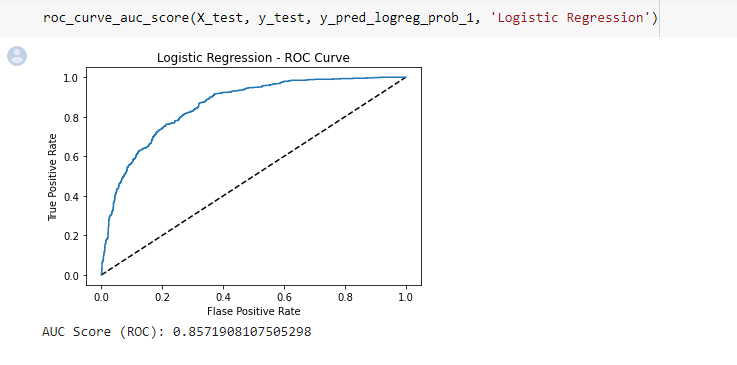


On va tester le score de précision et on va appeler la fonction de la **matrice de confusion** et interpréter le résultat :



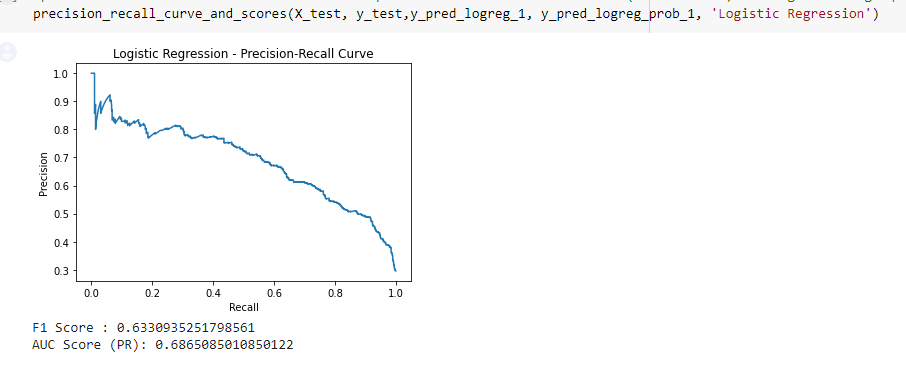
Si on compare le résultat trouvé avec celui avant l’optimisation des hyperparamètres , on remarque que le valeur de score test est restée presque la même .

On va maintenant tracer la courbe **ROC Curve**



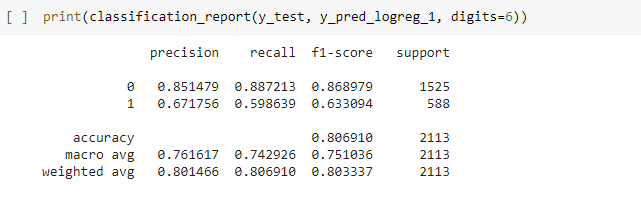
Le AUC Score trouvé cette fois est presque égal AUC Score qu’on a trouvé précédemment !

Essayons maintenant d’évaluer notre modèle en utilisant la courbe de **precision recall**



Le AUC score trouvé ( 0.63309)est presque égale AUC score trouvé précédemment ( 0.6) .

Finalement , on va appeler la fonction **classification report** et interpréter le résultat .

 Le score de la précision , recall , f1-score et accuracy sont tous restent les mêmes .

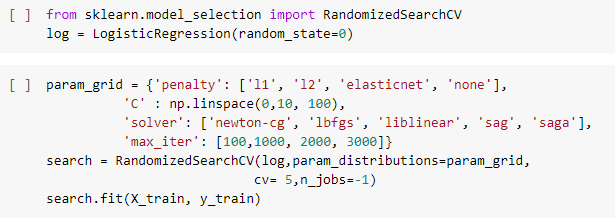
###### **Conclusion :**

Malgré qu’on a essayé d'améliorer notre modèle à travers l’affectation de hypermater par GridSearchCV , on remarque que le modèle n’a eu aucune amélioration. On va donc essayer avec un autre outil d'amélioration.

##### 

##### Logistic\_Regression with hyperparameter (RandomizedSearchCV) :

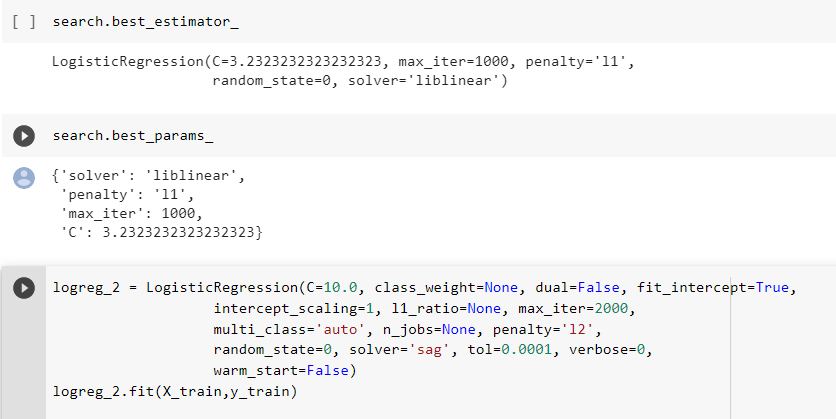
Dans ce qui suit, nous appliquons la classe sklearn.model\_selection.RandomizedSearchCV sur notre data frame que nous avons déjà préparée dans le chapitre précédent.



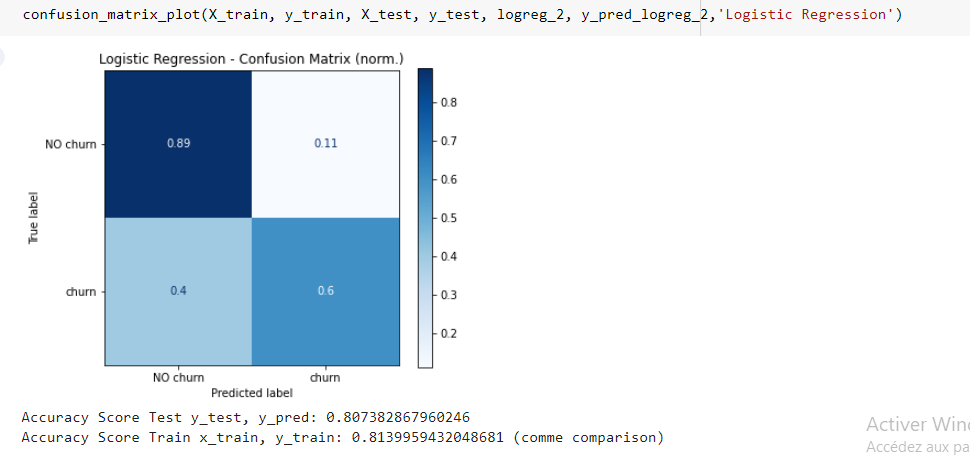
La figure ci-dessous nous montre que les meilleurs paramètres trouvés sont :

* C=3.2323232323232323, max\_iter=1000, penalty='l1', random\_state=0, solver='liblinear'

On va appeler la classe **Logistic Regression** mais cette fois ci en utilisant les meilleurs paramètres qu’on a trouvé à l’aide de RandomizedSearch

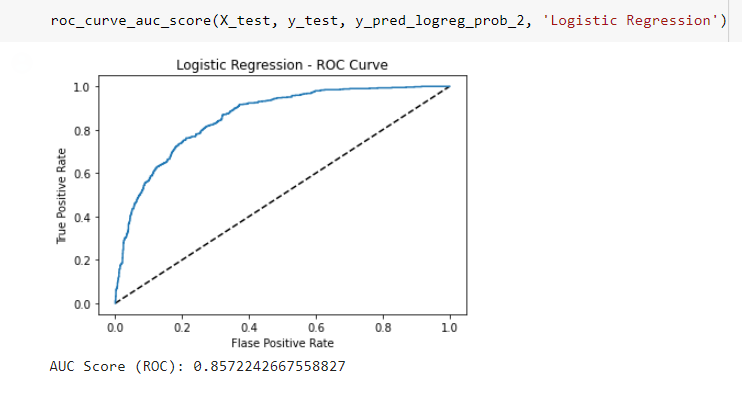


On va tester le score de précision et on va appeler la fonction de la **matrice de confusion** et interpréter le résultat :



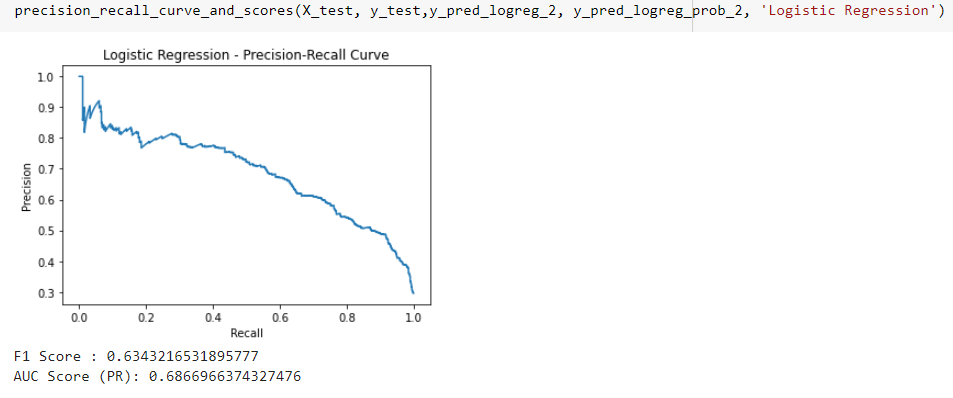
Si on compare le résultat trouvé avec celui avant l’optimisation des hyperparamètres , on remarque que le valeur de score test est restée presque la même .

On va maintenant tracer la courbe **ROC Curve**



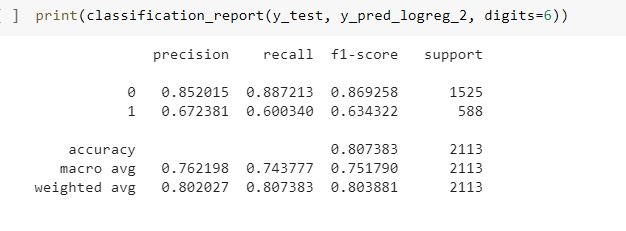
Le AUC Score trouvé cette fois est presque égal AUC Score qu’on a trouvé précédemment !

Essayons maintenant d’évaluer notre modèle en utilisant la courbe de **precision recall**



Le AUC score trouvé ( 0.63309)est presque égale AUC score trouvé précédemment ( 0.6) .

Finalement , on va appeler la fonction **classification report** et interpréter le résultat .



Le score de la précision , recall , f1-score et accuracy sont tous restent les mêmes .

##### Conclusion:

|  |  |  |  |
| --- | --- | --- | --- |
|  | LogisticRegression() | Logistic\_Regression with hyperparameter (GridSearchCV) | Logistic\_Regression with hyperparameter (RandomizedSearchCV |
| AUC ROC | 0.85 | 0.8571 | 0.8572 |
| AUC PR | 0.682 | 0.6865 | 0.6866 |

AUC = area under curve ( aire de la surface sous la courbe)

ROC =receiver operating characteristic ( fonction d’efficacité du récepteur)

PRC = precision recall curve (courbe précision-rappel)

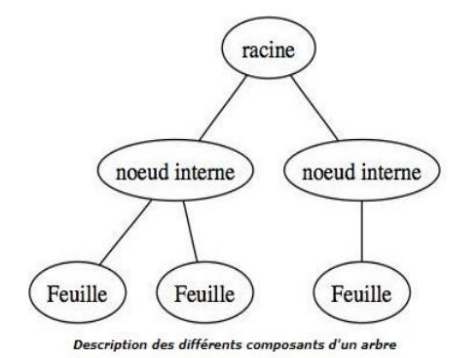
On remarque que malgré qu'on a essayé d'améliorer le modèle on n'a pas pu réussir à le faire .

## Models training (**Classificateur de L’arbre de décision**)

**DecisionTreeClassifier** est une classe capable d'effectuer une classification multi-classes sur un ensemble de données , Les arbres de décision (DT) sont une méthode d'apprentissage supervisé non paramétrique utilisée pour la classification et la régression. L'objectif est de créer un modèle qui prédit la valeur d'une variable cible en apprenant des règles de décision simples déduites des caractéristiques des données. Un arbre peut être vu comme une approximation constante par morceaux.

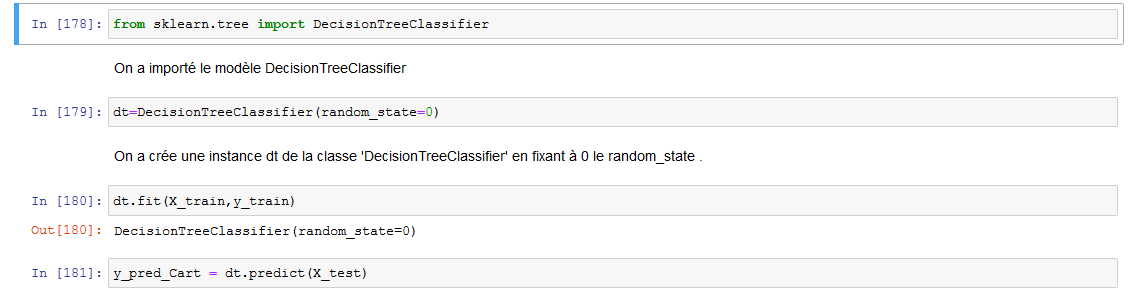
Les arbres de décision sont composés de *noeuds* et de *feuilles* reliés par des *branches* .

la racine est placée en haut et les feuilles en bas , Les noeuds internes sont appelés des noeuds de décision, les noeuds internes peuvent contenir une ou plusieurs règles (aussi appelées tests ou conditions)



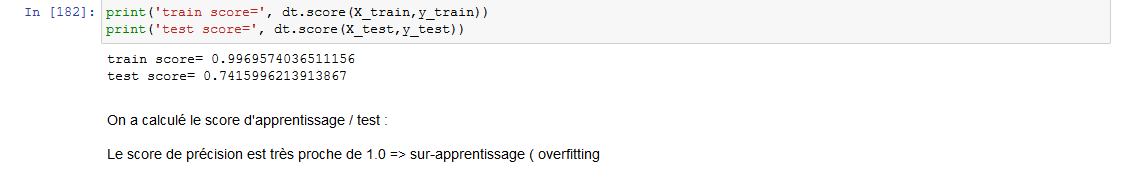
\*

On va importer tout d’abord le modèle sklearn.tree.DecisionTreeClassifier , créer une instance , fixer le paramètre random\_state à 0 et nous allons appliquer cette classe sur notre dataFrame que nous avons déja préparé dans le chapitre précédent ; nous allons entraîner notre modèle sur les sous-ensembles d’apprentissage .

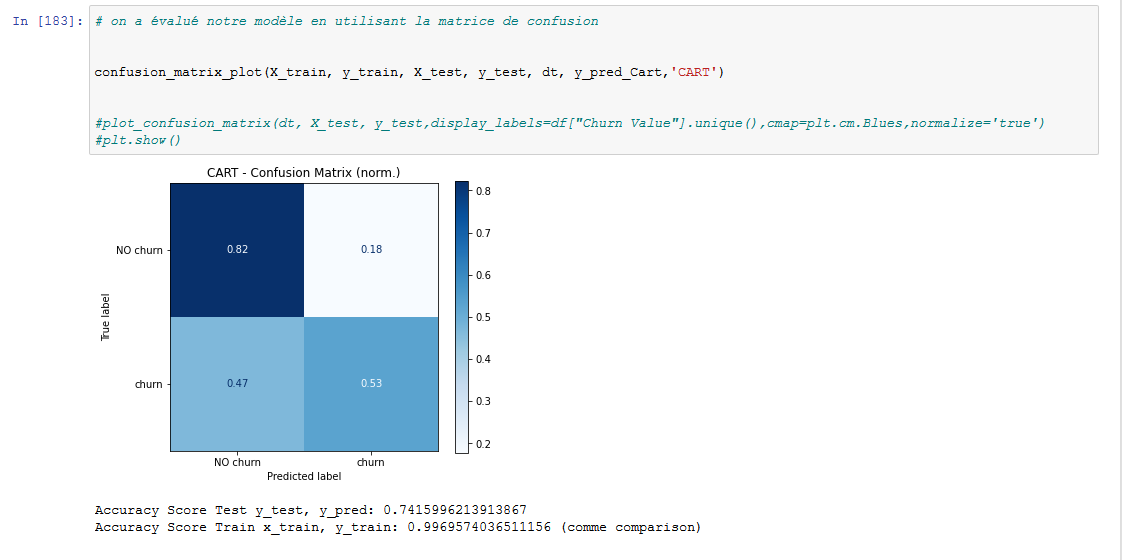


On va maintenant calculer le score d’apprentissage / test :

Comme on peut le voir dans la figure ci-dessous , le score de précision de l’entraînement est presque égal à 1 ( 100% ) ce qui est une indice sur le phénomène de sur-apprentissage ( overfitting ) . On doit alors essayer de fixer ce problème ultérieurement .



Par la suite , on évaluer notre modèle en utilisant la matrice de confusion :



Nous avons 4 valeurs dans notre matrice :

1. TP : notre modèle a prédit que le client ne va pas se désabonner et la prédiction était correcte : 0.82 ( valeur normalisée )
2. TN : notre modèle a prédit que le client va se désabonner et la prédiction était correcte : 0.53
3. FP : notre modèle a prédit que le client ne va pas se désabonner mais la prédiction était fausse : 0.47
4. FN : notre modèle a prédit que le client va se désabonner mais la prédiction était fausse : 0.18

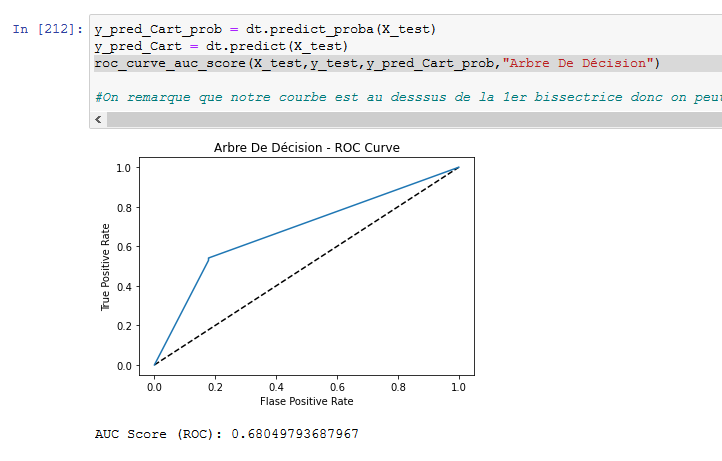
Évaluation

##### ***ROC Curve***

-On va évaluer notre modèle mais cette fois-ci en traçant la courbe de ROC .

Une courbe ROC (courbe caractéristique de fonctionnement du récepteur) est un graphe qui montre la performance d'un modèle de classification à tous les seuils de classification.

* Taux de vrais positifs = ( vrais positifs / (vrais positifs + faux négatifs ) )
* Taux de faux négatifs = ( faux positifs/ ( faux positifs + vrais négatifs ) )



L'AUC fournit une mesure globale des performances pour tous les seuils de classification possibles. Une façon d'interpréter l'AUC est la probabilité que le modèle classe un exemple positif aléatoire plus haut qu'un exemple négatif aléatoire.

On remarque que le score de l’AUC est égal à 0.68 .

Cette valeur sera utilisée ultérieurement pour être comparée avec la valeur qu’on trouvera dans lemodèle amélioré

##### ***Precision-Recall Curve***

-La courbe precision-recall curve montre le compromis entre précision et rappel pour différents seuils. Une zone élevée sous la courbe représente à la fois un rappel élevé et une haute précision, où une haute précision est liée à un faible taux de faux positifs et un rappel élevé correspond à un faible taux de faux négatifs. Des scores élevés pour les deux montrent que le classificateur renvoie des résultats précis (haute précision), ainsi que la majorité de tous les résultats positifs.

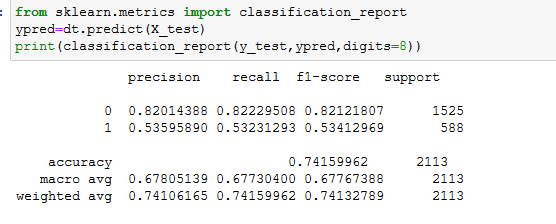
Recall = ( Vrais Positifs / ( vrais positifs + faux négatifs ) )

Précision = ( Vrais positifs / (Vrais Positifs + faux positifs ) )



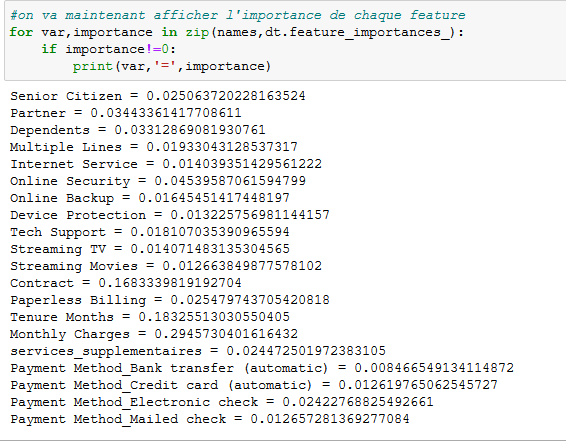
##### ***Rapport de Classification***

On va appeler la classe sklearn.matrics.classification report pour voir le rapport sur la précision , recall , f1-score et accuracy par classe ( 0 et 1 ) , On cherche à améliorer ces valeurs dans les étapes qui suivent .



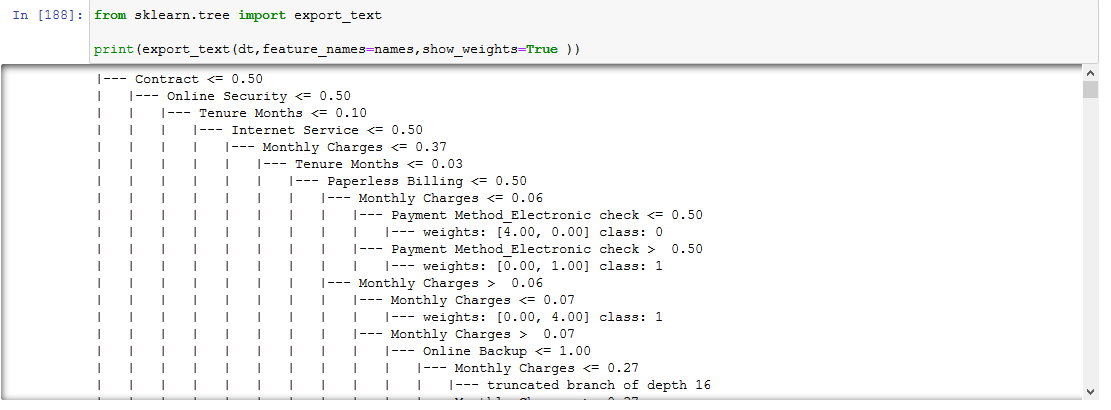
##### ***Affichage de l’arbre de décision***

dans la figure ci-dessous on a affiché l’importance de chaque feature de notre modèle .



Finalement on peut voir les règles de notre modèle pour avoir une idée sur les décisions prises par ce dernier .

La figure ci-dessous montre qu’il existe plusieures règles dans notre modèle ce qui le rend presque impossible à lire et à comprendre d’ou la nécessité de l’amélioration de notre modèle .



### Optimisation de Decision Tree Classifier :

Un **hyperparamètre** de modèle est une caractéristique d'un modèle qui est externe au modèle et dont la valeur ne peut être estimée à partir de données. La valeur de l'hyperparamètre doit être définie avant le début du processus d'apprentissage .

* on a fixé la profondeur maximale de l’arbre de notre modèle entre 1 et 10

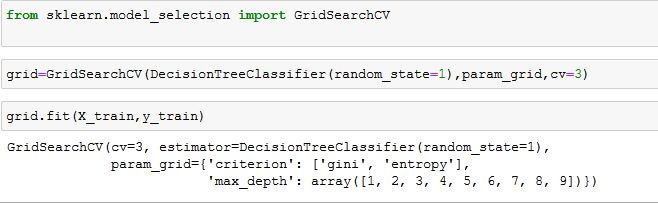
### 

##### ***Arbre de décision avec hyperparamètres ( GridSearchCV)***

La recherche par grille ( **GridSearch** ) est utilisée pour trouver les hyperparamètres optimaux d'un modèle qui aboutissent aux prédictions les plus «précises».

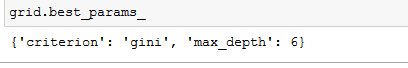
Dans notre cas on a choisi les paramètres criterion et max\_depth .

Dans ce qui suit , nous allons la classe sklearn.model\_selection.GridSearchCV pour pouvoir dégager les meilleurs paramètres .

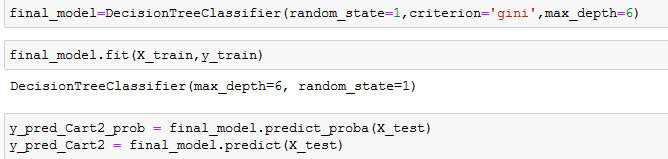


La figure ci-dessous nous montre que les meilleurs paramètres trouvés sont :

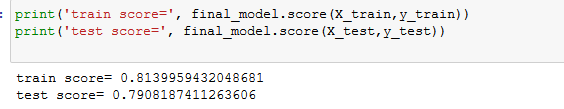
* L’indice de gini pour le paramètre ‘criterion’
* max\_depth doit être fixée à 6 .



On va maintenant appeler la classe **DecisionTreeClassifier** mais cette fois ci en utilisant les meilleurs paramètres qu’on a trouvé à l’aide de GridSearch

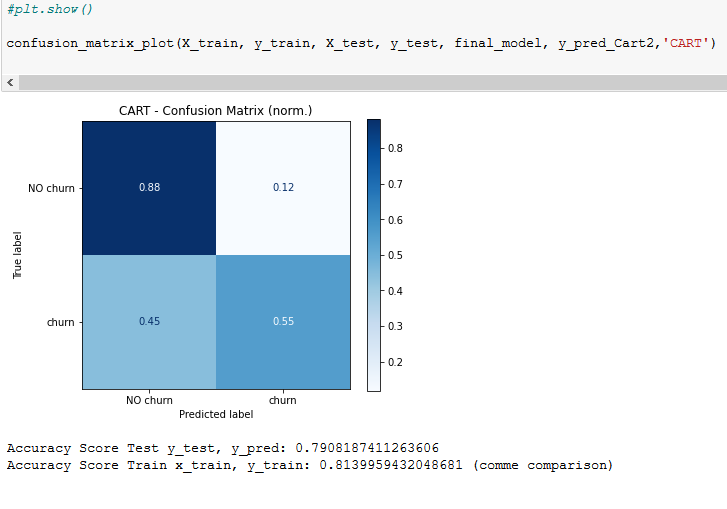


On va tester le score de précision :



Si on compare le résultat trouvé avec celui avant l’optimisation des hyperparamètres , on remarque qu’on n’a plus le problème de sur-apprentissage ce qui est une bonne indice sur la qualité de notre modèle .

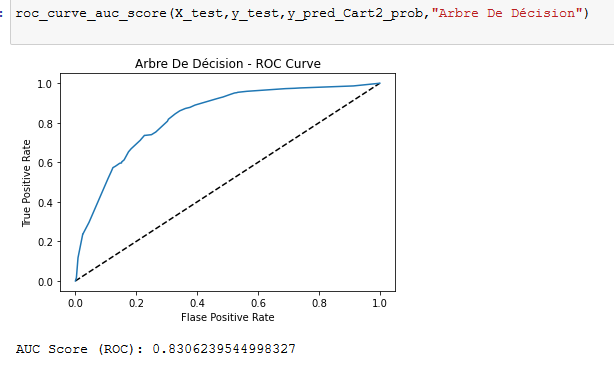
On va appeler la fonction de la **matrice de confusion** et interpréter le résultat :



On constate que les valeurs prédites cette fois ont plus de chance d’être correctes .

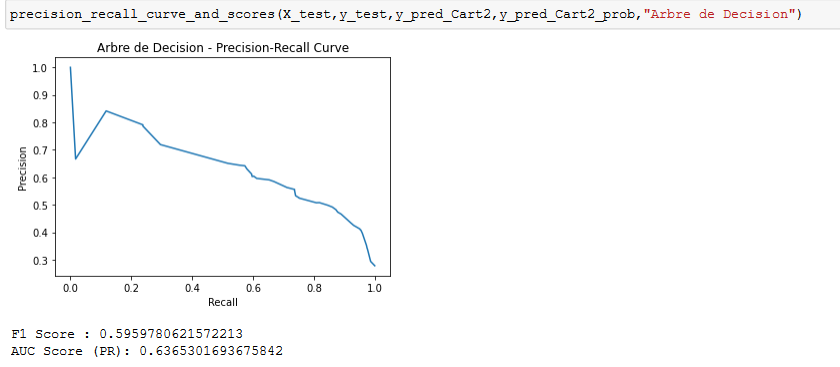
* La valeur de TP est égale à 0.88 alors que précédemment on a trouvé 0.82
* La valeur de FP est égale à 0.45 alors que précédemment on a trouvé 0.47 ( le modèle a moins de chance de prédire une valeur fausse )
* La valeur de TN est égale à 0.55 alors que précédemment on a trouvé 0.53
* La valeur de FN est égale à 0.12 alors que précédemment on a trouvé 0.18

On va maintenant tracer la courbe **ROC Curve**



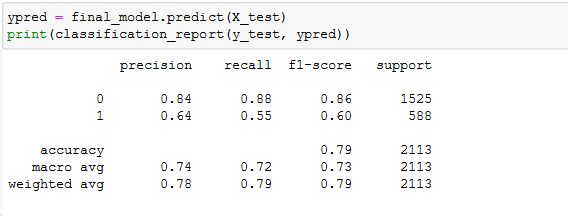
Le AUC Score trouvé cette fois est **>** AUC Score qu’on a trouvé précédemment !

Essayons maintenant d’évaluer notre modèle en utilisant la courbe de **precision recall**



Le AUC score trouvé ( 0.6365 ) **>** AUC score trouvé précédemment ( 0.6) .

Finalement , on va appeler la fonction **classification\_report** et interpréter le résultat .

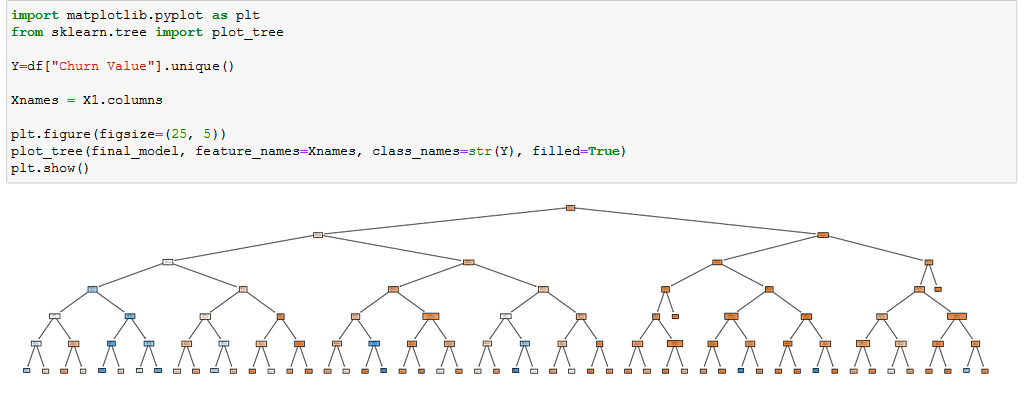


Le score de la précision , recall , f1-score et accuracy sont tous améliorés grâce aux hyperparamètres .

##### ***Affichage de l’arbre de décision***

On va afficher maintenant notre arbre de décision :

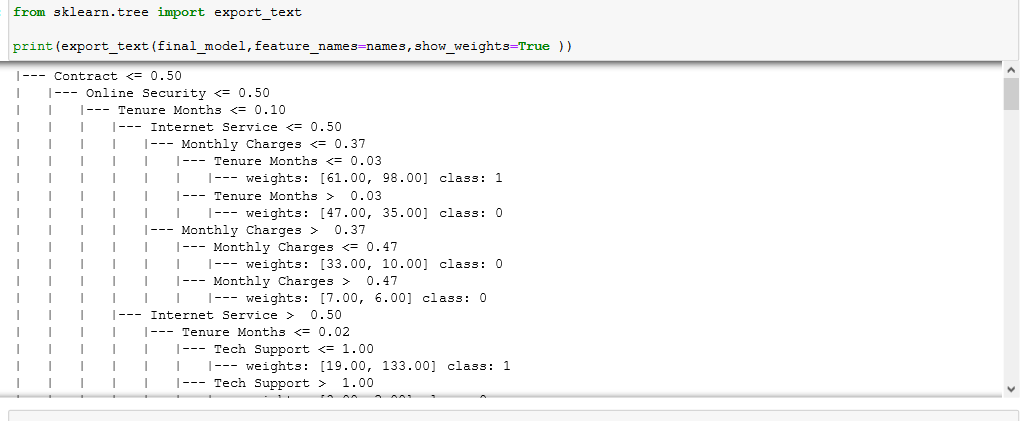
on va appeler la classe **sklearn.tree.plot\_tree** et on va tracer l’arbre



Interprétons ce résultat :

Notre arbre reste toujours compliquée et la lecture est presque impossible . Pour mieux interpréter le résultat on doit extraire les **règles** de cet arbre , sinon ce modèle a moins de feuilles que le modèle précédent , le temps d’exécution était rapide . Et si on interprète les règles montrées dans la figure ci-dessous on peut clairement voir qu’elles sont beaucoup plus faciles à lire .

On a moins de règles , la **complexité** de l’arbre a été améliorée :



##### ***Conclusion***

##### 

l’implémentation de la recherche par grille appliquée sur l’algorithme de classification par arbre de décision a **amélioré** notre modèle mais il reste toujours compliqué à interpréter .

|  |  |  |
| --- | --- | --- |
|  | Arbre de Décision | Arbre de Décision avec hyperparameteres (GridSearchCV) |
| AUC ROC | 0.68 | 0.83 |
| AUC PR | 0.6 | 0.6365 |

## Comparaison des algorithmes

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithme | Accuracy | AUC ROC | F1 score |
| KNN | 0.767 | 0.824 | 0.577 |
| Naive bayes | 0.789 | 0.832 | 0.615 |
| SVM | 0.80 | 0.833 | 0.6 |
| Decision Tree | 0.79 | 0.83 | 0.595 |
| Logistic regression | 0.807 | 0.857 | 0.634 |
| Random forest | 0.780 | 0.833 | 0.5671 |

Pour commencer si on compare par rapport à l'occurrence on remarque que l'algorithme SVM et Logistic Regression possèdent les valeurs les plus élevées

Ensuite une comparaison par rapport à la courbe AUC ROC nous donne que l'algorithme SVM, Logistic Regression et Random Forest possèdent les valeurs les plus élevées

Finalement une comparaison par rapport à la courbe F1 score nous donne que l'algorithme Naive Bayes , Logistic Regression possèdent les valeurs les plus élevées

**Pour conclure** et après tous les tests réalisés on peut déduire que l'algorithme de la Régression Linéaire est le meilleur d’entre tous.

# Chapitre 5 : Perspectives.

## 

On peut bien évidemment améliorer encore plus notre projet et les résultats finaux obtenu , pour cela nous avons pensé à divers solutions qui peuvent nous aider à atteindre cet objectif dans l’avenir tel que :

* Recherche de nouveaux algorithmes plus performants et efficaces afin qu’ils puissent nous garantir des meilleurs résultats
* Déploiement de notre projet afin de faciliter les recherches sur une base de données étendue et prédire les résultats d’une façon beaucoup plus rapide est souple cette automatisation garantit également une influence positive très remarquable sur les résultats finaux.

# Chapitre 6: Apports académiques et professionnels.

Ce projet est une synthèse de tout ce qu’on a vu durant ce semestre en Machine Learning.

Les compétance qu’on a acquis en travaillant sur ce module nous ont permis d’appliquer les notions du cours et de tester les différents classificateurs, dans le but de prédire le taux de désabonnement des clients et éviter des pertes financières aux entreprises de télécommunications.

grâce au compétences solides qu’on a acquis, L'obtention d’un résultat satisfaisant est à présent à notre porté en suivant une succession d’étapes bien particuliers qu’on peut les citer comme suit :

* Compréhension des données
* Préparation des données
* Implémentation du modèle
* Evaluation du modèle
* Optimisation
* Comparaison entre les différents modèles utilisés

En plus des connaissances acquises au niveau du cours consacré pour ce module, la recherche sur internet aussi nous a été de très grande aide puisqu’elle nous a fournis le nécessaire qui nous a permis de nous approfondir encore plus dans ce domaine intéressant et découvrir la plupart de ces avantages et points fort

# 

# Chapitre 7 : Analyse des articles.

# Sequential Feature Selection In Customer Churn Prediction Based on Naive Bayes

## Idée Générale de l’Article

Les progrès de la technologie aident les entreprises à résoudre de plus en plus de défis c'est pour cela les entreprises de télécommunications investissent d’avantage dans la recherche et le développement dans le domaine de data mining pour réduire l'activité de désabonnement qui a une grande influence sur les bénéfices totaux et l'image de l'entreprise, il est donc préférable qu'elle puisse être prévue et évitée. Par conséquent, une stratégie de développement, de gestion et de renforcement de la clientèle à long terme est nécessaire.L'un des défis pour une société de télécommunications est de conserver la fidélité du client , parce que la perte de clients pourrait entraîner une perte de revenus importante. Conserver l'ancien client est cinq à six fois moins coûteux que de trouver un nouveau client. Par conséquent, il est nécessaire que la société de télécommunications ait la capacité de prédire le client qui lui sera fidèle sans intervention qui pourrait causer une perte de revenus, des frais supplémentaires de fidélisation et de réacquisition de la clientèle.

Dans ce contexte, une étude expérimentale menée par Y.Yulianti et A.Saifuddin en proposant un modèle de prédiction du churn client, puis en l'appliquant à l'ensemble de données du churn client des télécommunications.

## Problématique

D'après les travaux présentés dans plusieurs articles traitant le sujet de la prédiction de désabonnement du client on peut conclure que l'algorithme le mieux adapté est le machine learning.

Pour augmenter la performance de cette prédiction, on traitera l'algorithme de K-nearest neighbor , Naive Bayes et Arbre de décision et on le comparera avec plusieurs algorithmes déjà existant pour comparer le taux de précision et de performance.

## Motivation

La quantité massive d'informations recueillies par les entreprises ces jours-ci incite les chercheurs à mener davantage d'expérimentations pour améliorer la qualité des services. qui est l'objectif principal pour toutes les entreprises après avoir remarqué la facilité de changement d'opérateur par les clients ce qui présente l'un des défis majeurs que doit relever l'industrie des télécommunications. c'est pourquoi ils essaient de développer des moyens de prédire les clients qui ont le potentiel de désabonnement, car attirer un nouveaux nécessite un coût beaucoup plus élevé que le maintien l'existants, cette prédiction sert à améliorer le service client et la qualité de produits. Pour faire des prédictions à l'aide de techniques Data mining, il fallait des données antérieures collectées, les données sur les consommateurs sont élargies dans les bases de données des entreprises. De nombreux algorithmes de classification ont été appliqués, tels que rule based classification, approche d'arbre de décision, réseaux de neurones, k-Nearest Neighbor, méthodes d'ensemble et méthodes statistiques classiques régression logistique, Naïve Bayes, Réseaux Bayésiens. Les méthodes statistiques classiques telles que la régression logistique et Naïve Bayes fournissent de bonnes notes, des résultats solides et sont faciles à appliquer.

L'ensemble de données utilisé pour les prédictions de désabonnement client a de nombreuses fonctionnalités, mais ces fonctionnalités ont la possibilité de redondance ou ne sont pas pertinentes afin de provoquer une diminution des performances du classificateur. Cette étude propose de mettre en œuvre la sélection des fonctionnalités pour sélectionner les fonctionnalités pertinentes et peut apporter des améliorations de performances aux modèles de prédiction de la rotation des clients.