Rapport de projet Machine Learning

Problème de la rétention de clientèle

Travail réalisé par:

Nadhir BOUHAOUALA

Haroun ELLEUCH

Mohamed Anas FATTOUM

Saïda MAJBOUR

Iskander REGAÏEG

Youssef Aziz ZGHAL

2020 - 2021

Table des matières

Introduction

Dans le cadre de la quatrième année d'études à ESPRIT, les étudiants de l'option Science des Données sont amenés à réaliser un projet en Machine Learning relatif à des problèmes socio-économiques récurrents des temps modernes.

Ce projet vise entre autres à appliquer les notions vues et traitées en cours et à les confronter à des cas pratiques réels.

Pour l'année universitaire 2020 - 2021, c'est la problématique de la rétention de client ("Customer churn") qui sera traitée.

1. Cadre du projet

1.1. Contexte

L'accroissement de la concurrence ainsi que la diversité sans précédent des offres sur le marché ont fait de la rétention de clients un problème proéminent. Dans l'industrie des télécommunications européenne par exemple, le départ d'un client coûte environ 500 euros [1].

Le libre échange européen a exposé les opérateurs à une concurrence plus hostile, notamment due à l'intervention d'opérateurs étranger. On recense un taux de churn moyen mensuel compris entre 8 et 12% et annuel allant de 20% à 40% [1], ce qui est loin d'être négligeable. Sachant qu'il est en général cinq fois plus coûteux d'acquérir un nouveau client que d'en fidéliser un [1], une nouvelle stratégie de marketing dite "défensive" a vu le jour. Et c'est dans ce contexte qu'intervient le Machine Learning afin de déceler les causes de départ des clients ainsi que de les prévenir dans le futur.

Il est aussi intéressant de noter l'existence du problème similaire qu'est l' "employee turnover". Dans ce cadre économique difficile et complexe, ces deux phénomènes constituent un défi majeur pour ces entreprises.

1.2. Objectifs et inspirations

L'objectif précis de ce travail est d'évaluer et de réaliser une "peer review" de trois articles scientifiques à impacts variés. Concrètement, les méthodes ainsi que les algorithmes utilisés seront reproduits et appliqués aux mêmes data sets afin de comparer les résultats obtenus et d'attester de leur précision. Une synthèse sera effectuée en tirant profit des conclusions de chaque article afin d'obtenir la démarche la plus adéquate et optimale.

1.3. Méthodologie CRISP

CRISP-DM pour Cross Industry Standard Process for Data Mining a été développé dans les années 1960 par IBM. Cette méthode se base sur 6 étapes distinctes (voir figure 1). A savoir: La compréhension du problème métier, la compréhension des données, la préparation des données, la modélisation, l'évaluation et enfin le déploiement.

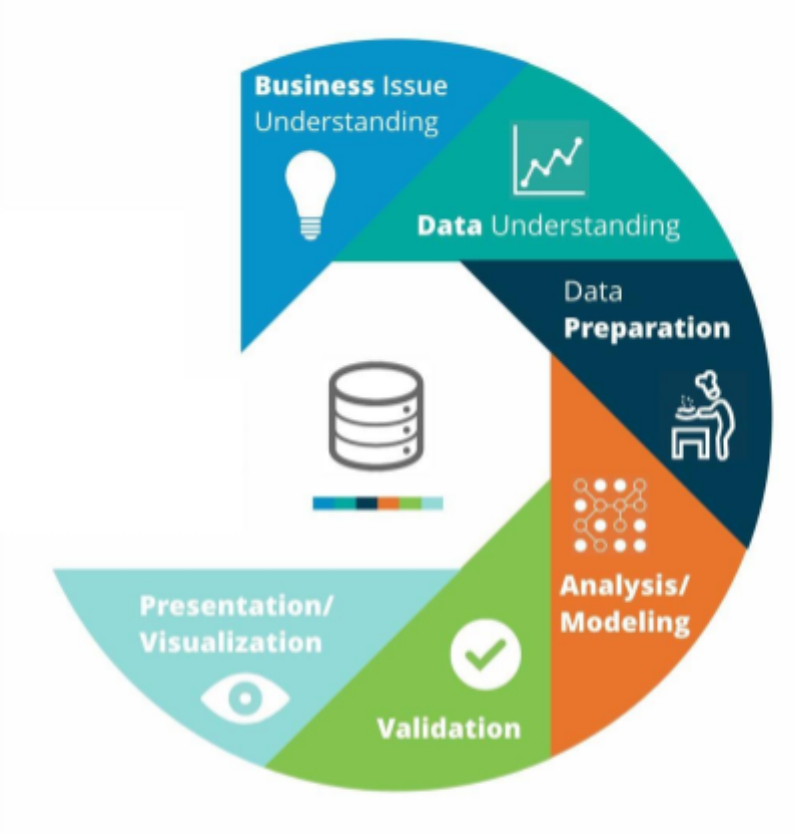


Figure 1 - Etapes du CRISP

2. Compréhension métier

La première étape consiste à bien comprendre les éléments métiers et problématiques que la Data Science vise à résoudre ou à améliorer.

Appliquée à cette situation, la compréhension métier se ramène principalement à l'étude du phénomène de churn dans l'industrie et la télécommunication . En d'autres termes à répondre aux questions suivantes : Qu'est ce que le customer churn? Quelles sont ses implications sur les opérateurs téléphoniques? Et quelles sont les stratégies envisageables pour y remédier?

Pour ce qui est des deux premières interrogations , les réponses ont déjà été apportées en amont dans ce rapport : Le churn , aussi appelé rétention client est un problème majeur d'enjeu économique important pour les entreprises modernes visant à conserver leurs clients plutôt quand en acquérir de nouveau. En effet, cela est plus rentable sur le long terme . Néanmoins, l'agressivité de la concurrence ajoute une dimension de complexité à cette tâche déjà ardue.

Quant à la dernière question, y répondre nous amène à entamer l'étape suivante de cette étude.

3. Compréhension des données

Le fichier utilisé est “Telco_customer_churn.csv.xlsx“ [2], créé par IBM [3]. Dans ce fichier, une ligne représente un client (“customer”) et une colonne un attribut. Ci-dessous, une liste reprenant la signification de chaque colonne [4][5]:

Données démographiques:

CustomerID: Identifiant unique servant à désigner chaque client.

Count: Variable de dashboarding servant à faire la somme du nombre de clients.

Gender: Le sexe du client : Homme / Femme.

Senior Citizen: Indique si le client a plus de 65 ans (à la fin du trimestre fiscal): Oui / Non

Partner: Indique si le client est marié: Oui / Non.

Dependents: Indique si le client vit avec des personnes à sa charge: Oui / Non. Ces personnes peuvent être des seniors, des enfants, etc…

Données géographiques:

Country: Le pays de résidence principal du client.

State: L’état (lieu) de résidence principal du client.

City: Ville de résidence principale du client.

Zip Code: Code postal du lieu de résidence principal du client.

Lat Long: Latitude et longitude combinées du lieu de résidence principal du client.

Latitude: Latitude du lieu de résidence principal du client.

Longitude: Longitude du lieu de résidence principal du client.

Données relatives aux services:

Tenure Months: Variable numérique exprimée en mois indiquant la durée pendant laquelle le client est resté avec l’entreprise. Valeur calculée à la fin du trimestre fiscal en cours.

Phone Service: Indique si le client est abonné à un service de téléphonie fixe (domestique): Oui / Non.

Multiple Lines: Indique si le client a plusieurs lignes. Trois modalités: Oui / Non / Pas de service téléphonique.

Internet Service: Indique si le client est abonné à un service Internet. Trois modalités: Fibre optique / DSL /Non.

Online Security: Indique si le client a recours à un service de sécurité Internet. Trois modalités: Oui / Non / Pas de service Internet.

Online Backup: Indique si le client a recours à un service de sauvegarde des données en ligne. Trois modalités: Oui / Non / Pas de service Internet.

Device Protection: Indique si le client a recours à un service de protection de sa machine. Trois modalités: Oui / Non / Pas de service Internet.

Tech Support: Indique si le client a recours au service technique. Trois modalités: Oui / Non / Pas de service Internet.

Streaming TV: Indique si le client a recours à un service de Streaming TV. Trois modalités: Oui / Non / Pas de service Internet.

Streaming Movies: Indique si le client a recours à un service de TV à la demande. Trois modalités: Oui / Non / Pas de service Internet.

Contract: Renseigne le type de durée du contrat. Trois modalités: Mois-par-mois / Annuel / deux ans.

Paperless Billing: Indique si le client utilise la facturation électronique: Oui / Non.

Payment method: Indique la méthode de paiement choisie par le client. Quatre modalités: Chèque électronique / Chèque par voie postale / Virement bancaire automatique / Carte de crédit automatique.

Monthly Charges: Variable numérique représentant les charges mensuelles du client calculées à la fin du trimestre fiscal en cours.

Total Charges: Variable numérique représentant les charges totales du client calculées à la fin du trimestre fiscal en cours.

Données relatives aux statuts des clients:

Churn Label: Indique si le client a résilié son contrat au cours du trimestre fiscal courant (Y a-t-il eu “churn” ?) : Oui / Non.

Churn Value: Indique si le client a résilié son contrat de manière binaire. 1 : Le client a quitté l'entreprise durant le trimestre fiscal courant. 0 : sinon. **Churn Score:** Variable scalaire allant de 0 à 100 utilisant l'outil prédictif IBM SPSS Modeler. Ce modèle intègre plusieurs facteurs causant le "churn". Plus le score est élevé, plus le client est susceptible de quitter l'entreprise.

CLTV: Pour **C**ustomer **L**ife**t**ime **V**alue. Ce score est calculé en utilisant des formules spécifiques au domaine. Les clients les plus importants ont les scores les plus élevés. Le Churn Score de ces derniers doit être surveillé de près.

Churn Reason: Indique la raison du "churn" du client.

L'algorithme Python qui a servi à l'identification et compréhension des variables ainsi qu'au recensement de leurs différentes modalités est détaillé dans l'annexe A.

Il est aussi pertinent de s'intéresser aux statistiques des différentes variables (voir Tableau 1) ainsi qu'aux éventuelles valeurs manquantes et leur proportions (Voir figure 2).

	Count	Zip Code	Latitude	Longitude	Tenure Months	Monthly Charges	Churn Value	Churn Score	CLTV
count	7043.0	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000	7043.000000
mean	1.0	93521.964646	36.282441	-119.798880	32.371149	64.761692	0.265370	58.699418	4400.295755
std	0.0	1865.794555	2.455723	2.157889	24.559481	30.090047	0.441561	21.525131	1183.057152
min	1.0	90001.000000	32.555828	-124.301372	0.000000	18.250000	0.000000	5.000000	2003.000000
25%	1.0	92102.000000	34.030915	-121.815412	9.000000	35.500000	0.000000	40.000000	3469.000000
50%	1.0	93552.000000	36.391777	-119.730885	29.000000	70.350000	0.000000	61.000000	4527.000000
75%	1.0	95351.000000	38.224869	-118.043237	55.000000	89.850000	1.000000	75.000000	5380.500000
max	1.0	96161.000000	41.962127	-114.192901	72.000000	118.750000	1.000000	100.000000	6500.000000

Tableau 1 - Statistiques des variables

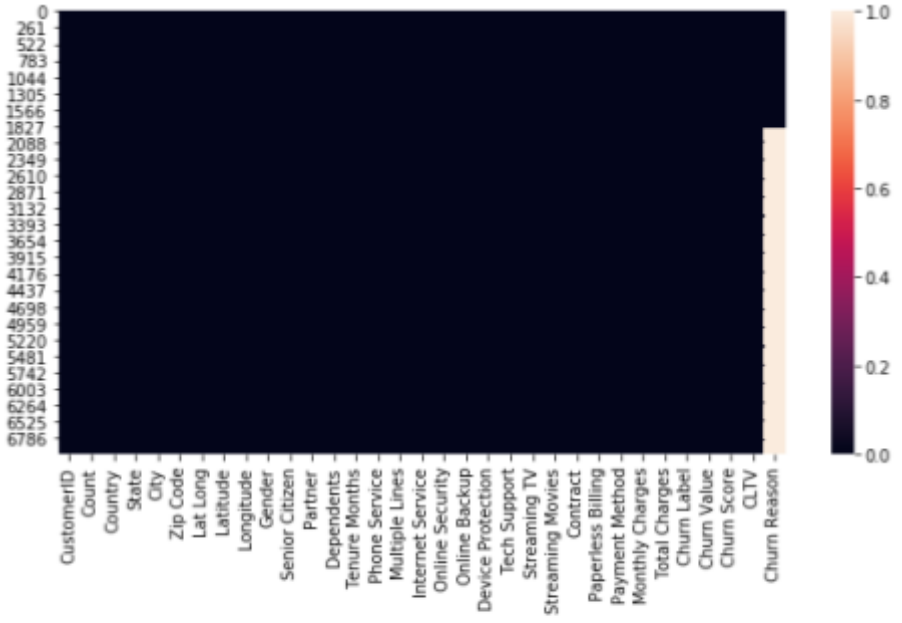


Figure 2 - Visualisation des valeurs manquantes

4. Préparation des données

Une fois qu'une compréhension plus consistante des données a été assurée, il est possible de passer à l'étape suivante, à savoir, la préparation des données. Néanmoins, il est intéressant de noter la nature itérative du processus de compréhension.

Le but principal de cette étape est l'élimination des données jugées non pertinentes (telles que les noms des clients ou leurs identifiants), la gestion des données manquantes, l'encodage et la mise à l'échelle des données, l'extraction de la variable labellisée cible etc...

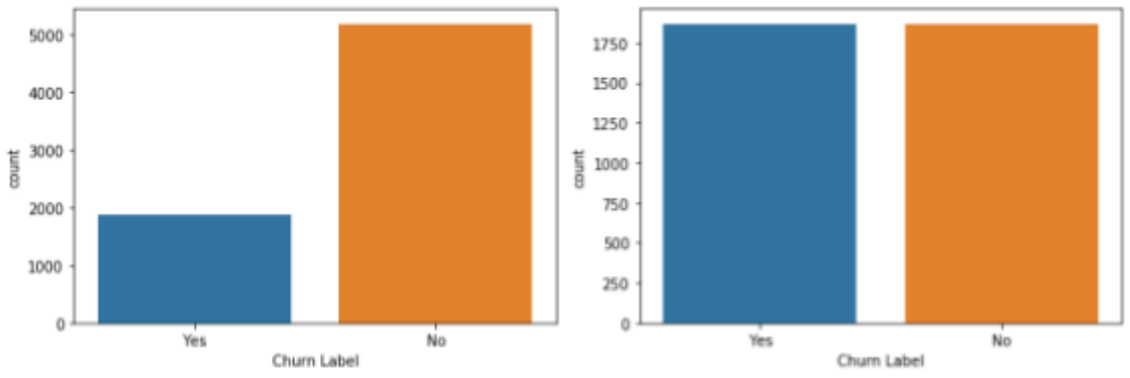
En sortie de cette étape, les données seront nettoyées et optimales pour la modélisation.

Concrètement, trois possibilités d'approche se présentent. Il est possible de traiter cette étape en commun pour les trois articles scientifiques (Voir annexe B), ou encore d'utiliser la même préparation pour les articles A et B avant de traiter l'article C séparément ou enfin, une approche traitant chaque article séparément. Mais puisque le dataset est commun aux trois articles, le travail réalisé sera identique pour chacun d'entre eux.

Les traitements réalisés sont usuels et bien détaillés dans les Notebooks joints à ce présent rapport ("Article_A.ipynb" etc..). Néanmoins ils peuvent se résumer principalement en l'utilisation des bibliothèques Pandas, NumPy, MatPlotLib et SeaBorn de Python.

4.1. Extraction de la variable cible

Après une exploration sommaire des données, la variable cible est extraite (voir figure 3-a), il s'agit de la colonne "Churn Label". On constate, en outre, un déséquilibre au



(a) - Extraction (b) - Equilibrage

Figure 3 - Variable cible

niveau des modalités “No” et “Yes”. Il faut donc équilibrer ces dernières avant de procéder à l’étape suivante et éviter ainsi de biaiser les modèles. Pour cela, deux approches sont possibles: l’*oversampling* et l’*undersampling.* Le choix s’est finalement porté sur l’undersampling puisqu’il est plus facilement réalisable et qu’en sortie, il reste encore plus de 1800 observations(voir figure 3-b).

4.2. Réduction de dimension

L’élimination de certaines colonnes a aussi été réalisée, pour diverses raisons (Feature extraction / Réduction de dimension). “Churn Reason” présente plus 73% de valeurs manquantes (voir figure 2), elle a donc été retirée. D’autres variables unimodales, redondantes ou non significatives ont, elles-aussi, été retirées telles que: CustomerID, Country, State, Churn Label, Latitude, Longitude, Lat Long, Count et City.

4.3. Encodage et standardisation des données

D’autres variables présentent trois modalités, une réduction à 2 seulement est possible et a été réalisée. L’encodage des données a ensuite été réalisé pour les variables catégorielles. On passe ensuite au centrage-réduction (standardisation) des données afin de les mettre à la même échelle.

4.4. Sélection de caractéristiques

A ce niveau, il est aussi possible de réaliser une sélection des caractéristiques (Feature selection) afin d’améliorer les performances des futurs modèles en temps d’apprentissage et en précision. Pour cela, selon l’article considéré, différentes méthodes ont été adoptées. La Matrice de Corrélation de Pearson pour l’article A, les différents types de sélection séquentielle (à savoir: SFS, SFFS, SBS et SBFS) pour l’article B et une synthèse de ces résultats pour l’article C, puisqu’il a été traité en dernier lieu.

a. Matrice de Corrélation de Pearson

La matrice de corrélation indique les valeurs de corrélation, qui mesurent le degré de relation linéaire entre chaque paire de variables. Les valeurs de corrélation peuvent être

comprises entre -1 et +1. Si les deux variables ont tendance à augmenter et à diminuer en même temps, la valeur de corrélation est positive. Lorsqu'une variable augmente alors que l'autre diminue, la valeur de corrélation est négative.

L'affichage de cette dernière a été fait en utilisant le *cluster map* de la bibliothèque *Seaborn* et le calcul grâce à *Pandas*.

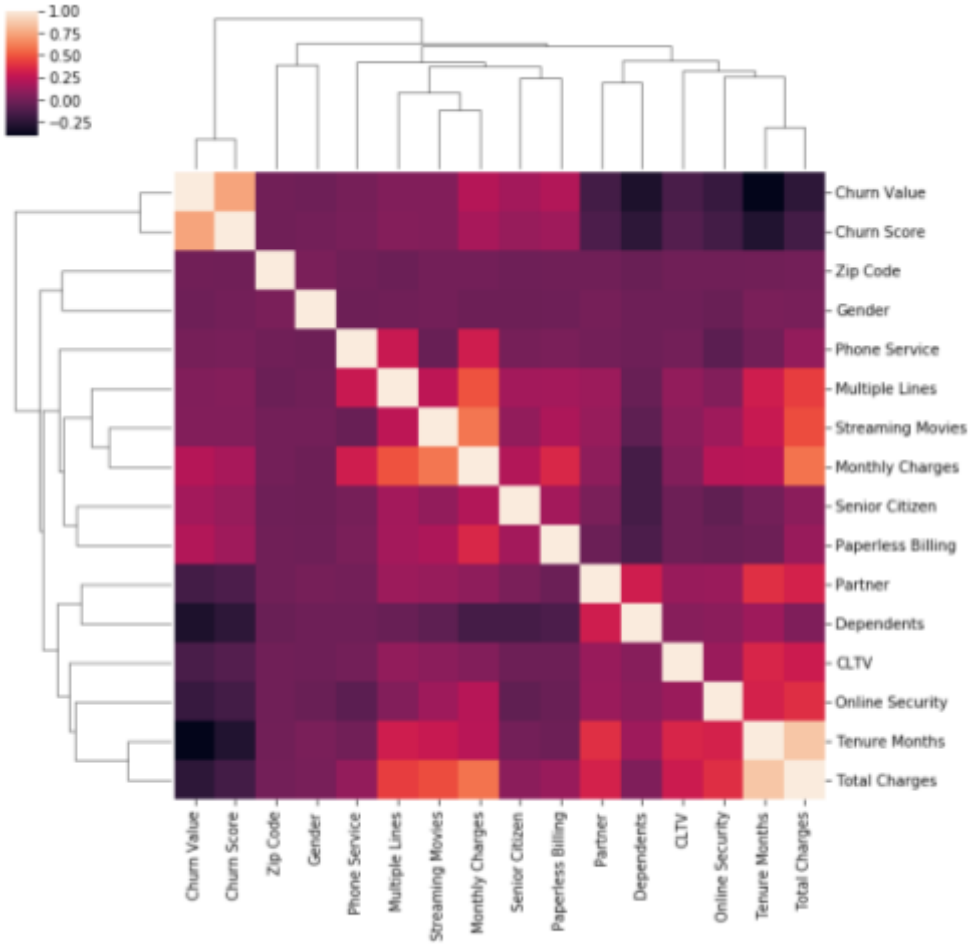


Figure 4 - Matrice de Corrélation de Pearson

L'inspection de cette dernière (voir figure 4) indique que la majorité des variables ne sont pas corrélées, exception faite de *Tenure Months* et *Total Charges* qui en présentent une.

b. Sélection Séquentielle

La *Sélection Séquentielle* est une famille d'algorithmes de recherches gloutons. Le fonctionnement général peut être résumé comme suit: Un paramètre *k* représentant le

nombre de caractéristiques (features) est introduit. L'algorithme va parcourir la liste de ces dernières et choisir, une à la fois, les *k* caractéristiques les plus pertinentes. L'implémentation des quatre variantes a été faite en important la bibliothèque *SequentialFeatureSelector* de *mlxtend* (Voir "Article_B.ipnyb").

La première variante implémentée est **SFS** pour Sequential Forward Selection. Ici, le parcours se fait en partant d'une liste vide de caractéristiques et le remplissage se fait progressivement. Par exemple, pour *k* = 5, les caractéristiques retenues sont : (1, 5, 10, 13, 15)

La deuxième variante implémentée est **SBS** pour Sequential Backward Selection. Ici, le parcours se fait en partant d'une liste remplie de caractéristiques et l'élimination se fait progressivement. En gardant pour *k* = 5, les caractéristiques retenues sont : (5, 10, 11, 13, 24), ce qui est différent du résultat obtenu précédemment. Dès lors, une question se pose: laquelle des deux variantes est-elle la meilleure ? Pour répondre à cela, il faudrait d'abord terminer la modélisation et discuter les résultats (Voir partie modélisation). Néanmoins, la nature de SBS fait qu'elle est plus précise puisqu'elle prend en considération l'interaction entre la caractéristique à sélectionner avec un sous-ensemble plus grand de caractéristiques.

Ensuite, viennent les variables à sélection flottante. Ici, l'addition ou le retrait d'une caractéristique ne sont pas définitifs et une variable peut ainsi être sélectionnée lors d'une itération ultérieure.

La première variante à sélection variable implémentée est **SFFS** pour Sequential Forward Floating Selection. L'algorithme est donc basé sur le SFS, comme l'indique son nom. Son exécution pour *k* = 5 donne le résultat suivant (0, 5, 10, 13, 15). Le résultat est encore une fois différent des autres variantes mais certaines caractéristiques restent en commun, notamment celles numérotées 5,10 et 13 (en partant de zéro).

La dernière variante utilisée est **SBFS** pour Sequential Backward Floating Selection, qui reprend le fonctionnement de SBS et y ajoute le flottaison de la sélection. Cet algorithme donne la liste de caractéristiques suivantes pour *k* = 5 : (5, 11, 13, 15, 18) qui présente elle aussi des éléments en commun (5 et 13) avec les précédents algorithmes.

4.5. Division des données

Avant de pouvoir entamer pleinement l'étape de la modélisation, il reste à diviser les données en 2 sous-ensembles: Un jeu dédié à l'entraînement (training dataset) et un autre pour le test (test dataset) . En règle de bonne pratique, la division se fait comme suit:

- 70% des données pour le training dataset
- 30% des données pour le test dataset.

Cette division est, bien évidemment, aléatoire et servira à entraîner les différents algorithmes lors de l'apprentissage. (Voir parties Modélisation et Évaluation).

.

5. Modélisation

L'étape de la modélisation est l'une des plus cruciales. En effet, c'est lors de cette dernière que les modèles prédictifs et ceux de segmentation sont construits, notamment au moyen d'algorithmes enchaînés.

Il existe pour cela plusieurs types d'algorithmes: ceux de l'apprentissage supervisé permettant la classification à partir de données étiquetées (labellisées) et ceux servant pour l'apprentissage *non* supervisé servant pour le regroupement (*Clustering,* segmentation) de profils homogènes à partir de données non étiquetées. Cependant, ces derniers nécessitent une étape additionnelle, le *Profiling* afin d'identifier chaque cluster à un profil, puisque les données ne sont pas labellisées.

Pour ce qui est de l'apprentissage supervisé, il a servi principalement à la classification dans ce projet. Parmi les algorithmes employés, il est possible de citer: Les arbres de décision, Random Forest, K-Nearest Neighbors (k-NN), Support Vector Machine (SVM) Bayes Naïf, les réseaux de neurones artificiels (ANN) et l'Adaptive boosting (adaBoost).

En ce qui concerne l'apprentissage *non* supervisé, ce dernier a servi pour le clustering. La méthode des centres mobiles (K-Means) et la classification ascendante hiérarchique (CAH) sont les principaux algorithmes utilisés.

5.1. Apprentissage supervisé

a. K-Nearest Neighbors (k-NN)

Sans doute l'un des algorithmes les plus connus dans sa catégorie, notamment grâce à sa facilité d'appréhension, le K-NN est basé sur un calcul de distance entre une observation et ses voisins. En précisant le paramètre *k*, on retiendra ainsi les *k* plus proches voisins selon la distance choisie (Manhattan, Euclidienne, Minkowski ...).Il est en revanche important de souligner l'importance du paramètre *k*. En effet une valeur trop petite entraînera un sous apprentissage (*underfitting*) et au contraire, une valeur trop grande causera un sur apprentissage (*overfitting*), tous deux étant des effets indésirables.

Pour ce problème, la distance de Manhattan a été adoptée puisque les variables ne sont pas du même type. La valeur optimale du paramètre *k* est obtenue en utilisant l'algorithme GridSearchCV de la bibliothèque sklearn.model_selection qui sert à déterminer les hyper-paramètres optimaux. La valeur de *k* obtenue est enregistrée dans la variable *best_model*.

b. Arbres de décision

Un autre outil de création de modèles prédictifs est l'arbre de décision ; dans ce cas précisément, les arbres de classification (bibliothèque sklearn.tree). Le principe de fonctionnement est assez simple et se base sur une règle pour établir une prédiction binaire à chaque branche.

En pratique, un arbre de décision complet (sans nombre maximal de niveaux a été dessiné) à des fins de visualisation et compréhension. Un train score de 1.0 a été obtenu, ce qui est bien sûr indésirable. La deuxième étape fut donc d'utiliser GridSearchCV afin d'optimiser les paramètres tels que le nombre maximal de niveaux (*max_depth*). Une comparaison manuelle a ensuite été réalisée en prenant les valeurs voisines de celle en sortie du GridSearchCV sur et en traçant les matrices de confusion dans chaque cas (Voir "Article_C.ipynb") et en mettant l'accent sur les taux de faux positifs observés.

Les arbres de décision ont été tracés en utilisant la bibliothèque *sklearn.tree*.

c. Random Forest

Cet algorithme opère en générant des arbres de décision aléatoires, comme son nom l'indique. Le fait que les arbres générés sont faiblement corrélés en fait un outil très puissant. En effet, chaque arbre va prédire la classe d'après les données du problème et la classe ayant reçu le plus de "votes" par tous les arbres sera celle en sortie du modèle.

Cet algorithme est implémenté en important la bibliothèques sklearn.ensemble et le nombre de niveaux optimal est donné par GridSearchCV dans l'article A (voir "Article_A.ipynb") et par visualisation de l'erreur en fonction du nombre d'estimateurs *n_estimator* dans l'article C (voir "Article_B.ipynb").

d. Support Vector Machine (SVM)

Le but de cet algorithme est de trouver un hyperplan dans un espace de dimension N (N étant le nombre de caractéristiques, *features*) qui classifie distinctement les points de données [6].

L'algorithme va donc chercher l'hyperplan optimal en maximisant la marge représentant la distance entre les points de chaque classe.

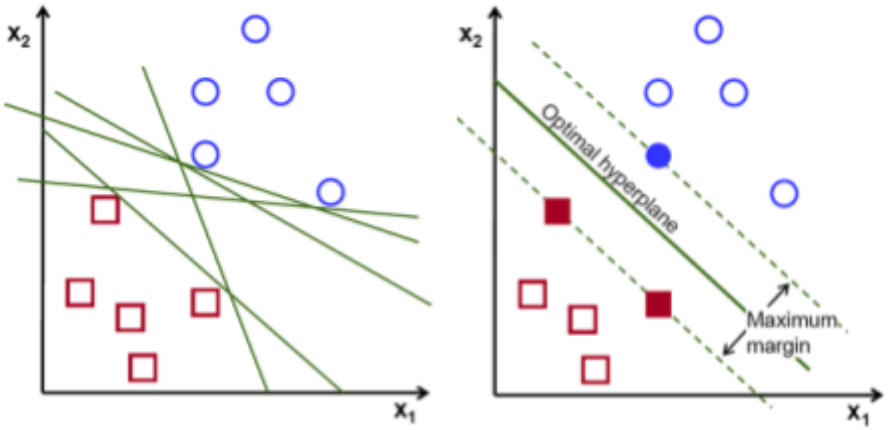


Figure 5 - Fonctionnement de SVM [6]

SVM a été implémenté en utilisant la bibliothèque *sklearn.svm* en utilisant GridSearchCV pour trouver les hyper-paramètres optimaux.

e. Bayes Naïf

Bayes Naïf (ou *Naïve Bayes*) est un algorithme à l'approche probabiliste qui se base, comme l'indique son nom, sur le théorème de Bayes impliquant les probabilités conditionnelles afin d'inférer la classe de l'individu. Cependant, le calcul des probabilités conjointes pour toutes les *features* est assez ardu. C'est pour cela que l'hypothèse naïve de l'indépendance de ces dernières est présumée.

Pour implémenter cette méthode, il faut importer la bibliothèque *sklearn.naive_bayes*.

Dans la résolution de ce problème, cet algorithme a été associé aux quatres algorithmes de sélection séquentielle détaillés dans le chapitre 4 (voir “Article_B.ipynb”). Les performances de ces associations seront comparées dans le prochain chapitre.

f. Réseaux de Neurones Artificiels (ANN)

Les réseaux neuronaux artificiels peuvent aussi être utilisés pour la classification. Le choix s’est porté sur la variante la plus simple: l’ANN qui est un réseau unidirectionnel à 3 couches (voir figure 6) : une couche d’entrée, une couche cachée responsable des différentes opérations de régressions logistiques et une couche de sortie. Parmi les variantes, on peut citer les RNN ou réseaux neuronaux récursifs ainsi que les CNN pour réseaux de neurones compliqués (*Convoluted NN*) [7].

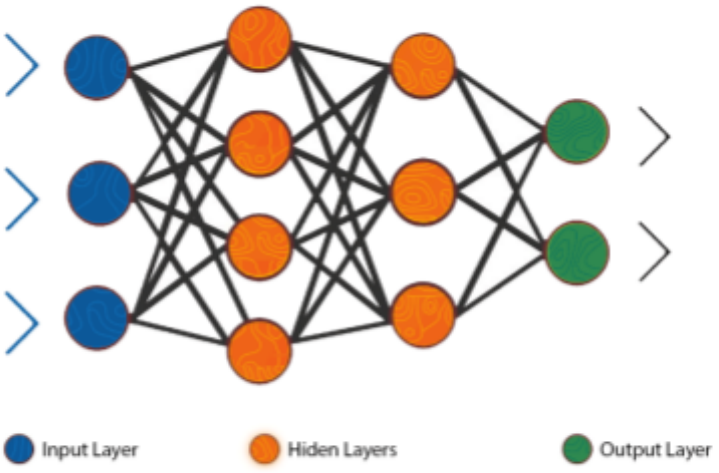


Figure 6 - Structure de l'ANN [7]

Dans le cas présent, deux couches cachées de 12 nœuds chacune ont été utilisées afin d'établir un bon compromis biais/variance (équilibre entre qualité d’apprentissage et qualité de prévision). La couche de sortie comprend un seul nœud qui contiendra le résultat de l'exécution.

g. Adaptive Boosting (adaBoost)

L'adaBoost est un algorithme qui a pour but d'améliorer les performances des classifieurs faibles en associant des poids aux sorties de ces derniers et en les sommant (voir figure 7).

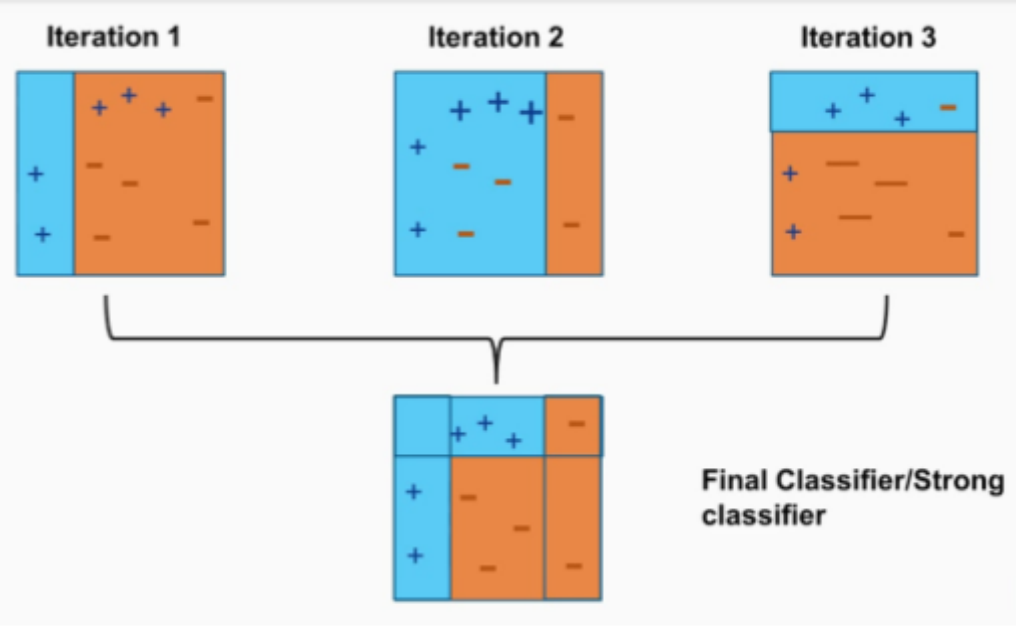


Figure 7 - Fonctionnement de l'Adaptative Boosting

Le choix s'est porté sur l'arbre de décision afin d'étudier les effets du boosting sur ce dernier et les comparer (Voir chapitre Evaluation).

5.2. Apprentissage non supervisé

a. Méthode des Centres Mobiles (K-Means)

Cette approche se base sur le théorème d'Huygens et a pour but de caractériser *k* groupes homogènes au sein d'une population hétérogène en maximisant l'inertie inter-classe tout en réduisant celle intra-classe. Cela se fait en plaçant *k* centres et à mettre à jour leur positions respectives en calculant leurs distances par rapport aux points de données.

K-Means a été implémenté en important la bibliothèque *sklearn* et donnant 2 comme nombre de clusters en paramètre.

b. Classification Ascendante Hiérarchique (CAH)

Cet algorithme va créer une structure hiérarchisée à partir des données entrées en groupant de proche en proche les individus selon le type de distance choisie (Euclidienne, Manhattan...). Le nombre de *clusters* n'est pas prédéfini, pour obtenir 2 groupes, il suffit de

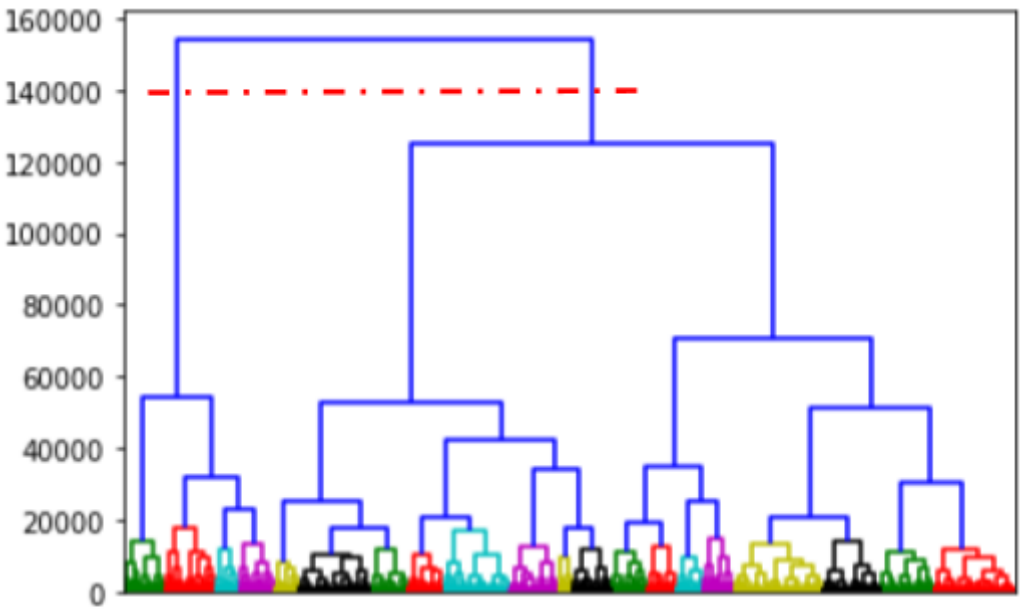


Figure 8 - Dendrogramme de la CAH

faire une coupe au niveau du dendrogramme obtenu en sortie de l'algorithme.

Son implémentation a été faite en utilisant la bibliothèque *scipy.cluseter.hierarchy* en utilisant la distance euclidienne comme métrique et le critère de Ward pour la différentiation entre les groupes.

Il est possible d'observer sur le dendrogramme obtenu (voir figure 8) que deux groupes sont bel et bien séparables au niveau de la droite hachurée.

6. Evaluation

Après avoir établi les différents modèles de prédiction, vient l'étape de l'évaluation de ces derniers.

Le premier groupe de modèle contient ceux implémentés pour l'étude de l'article A, à savoir le KNN, Random Forest et SVM. Une comparaison a été faite en traçant un tableau récapitulatif (Voir tableau 2).

	training	testing
Propose KNN	0.852	0.820
Random Forest	0.9307	0.9161
SVM	0.9173	0.9000

Tableau 2 - Scores pour KNN, RF et SVM

Première constatation, aucun des trois modèles n'a succombé au sur-apprentissage, puisque les valeurs des scores d'entraînement sont toutes inférieures à 1.0.

On remarque aussi que les valeurs des scores de test et d'entraînement sont assez proches les unes des autres pour chaque modèle. Ce qui indique une relative qualité de ces derniers.

En dernier lieu, reste à déterminer le meilleur des trois modèles. Random Forest présente le score de test le plus élevé avec 91%, ce qui en fait le modèle le plus performant parmi les 3.

Il est aussi possible d'affirmer cette conclusion en utilisant la courbe Receiver Operating Characteristic (ou *courbe ROC*) afin de visualiser le taux de vrais positifs en fonction du taux de faux positifs par modèle (Voir figure 9). La courbe de plus grande pente correspond au modèle le plus précis. On observe une quasi-superposition des courbes de SVM et de

Random Forest, cette dernière ayant une pente légèrement supérieure. La courbe de KNN est en dessous des deux autres. Ces résultats corroborent bel et bien ceux du tableau.

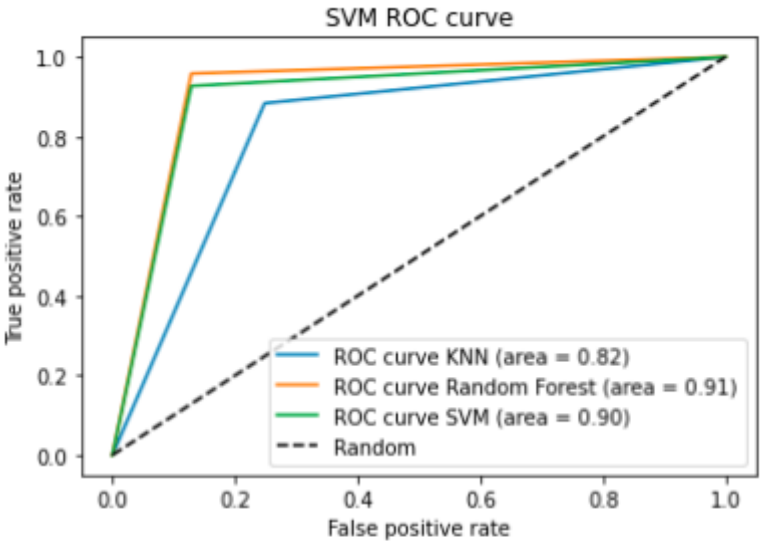


Figure 9 - Courbes ROC de KNN, RF et SVM

Pour ce qui est de l'article B, quatre modèles utilisant Bayes Naïf ont été créés basés sur des variantes de sélection séquentielle de *features*. Afin de les comparer, on a fait varier le nombre de features à sélectionner de 1 à 30 (nombre total de colonnes, donc valeur maximale possible) pour chacun des modèles et chaque fois, les métriques de performances ont été relevées: l'aire sous la courbe ROC (AUC - *area under curve*) et la précision (ACC - *accuracy*) (Voir Annexe C).

Un algorithme de recherche de maximum dans le tableau ainsi obtenu permet de trouver la valeur optimale recherchée par modèle (Voir figure 10). On constate que l'algorithme de Bayes Naïf sans sélection est moins précis que ceux ayant bénéficié d'une *feature selection* d'environ 10%. Le modèle utilisant Naïve Bayes avec SFS est le plus précis avec un nombre de caractéristiques égal à 16. Vient ensuite le modèle utilisant SFFS avec 8 caractéristiques suivi des modèles SBS et SBFS presqu'ex-aequo avec des nombres de caractéristiques sélectionnées respectifs de 9 et 4.

```
colonne|ACC_naive| max= 0.8171 ligne= 1
colonne|AUC_naive| max= 0.8173 ligne= 1
colonne|SFS_ACC| max= 0.9019 ligne= 16
colonne|SFS_AUC| max= 0.9019 ligne= 16
colonne|SBS_ACC| max= 0.8992 ligne= 9
colonne|SBS_AUC| max= 0.8993 ligne= 9
colonne|SFFS_ACC| max= 0.9001 ligne= 8
colonne|SFFS_AUC| max= 0.9002 ligne= 8
colonne|SBFS_ACC| max= 0.8956 ligne= 4
colonne|SBFS_AUC| max= 0.8957 ligne= 4
```

Figure 10 - Recherche des meilleures valeurs des modèles Bayésiens

Cependant, les différences relevées entre les quatres modèles ne sont pas assez significatives pour causer un quelconque impact. La différenciation pourrait se faire suivant le nombre de caractéristiques choisies. Ainsi un plus petit nombre entraînerait des calculs moins complexes.

Enfin, concernant le travail réalisé dans le cadre de l'article C, un tableau par famille de modèle a été tracé compilant les métriques de performances de chaque modèle (Voir figures 11 et 12).

	Score accuracy training	Score accuracy testing
Arbre de decision	0.928	0.908
Random Forest	0.941	0.913
Arbre de decision+AdaBoost	1.000	0.921
Réseaux de neurones	0.962	0.888

Figure 11 - Performances de AD, RF, AD+adaBoost et ANN

On constate pour tous les modèles d'apprentissage supervisé de la liste que les scores d'entraînements sont supérieurs à ceux des tests (Voir figure 12). On remarque aussi que l'algorithme d'adaBoost a permis d'augmenter la précision du modèle de l'arbre de décision. Néanmoins, ce dernier est maintenant en sur-apprentissage (train_score = 1.0). En outre,

l'arbre de décision boosté est le plus précis suivi de random forest puis de l'arbre de décision. L'ANN est en dernière place.

Afin de comparer des modèles d'apprentissage non supervisés, il faut avoir recours à une métrique appelée "Silhouette" (Voir figure 12) qui est calculée utilisant la distance intra-cluster et la distance moyenne du plus proche cluster pour chaque individu. Un score de silhouette plus proche de 1 est plus désirable afin d'éviter l'empattement (overlapping) des clusters. Ainsi, K-Means est le modèle le plus adapté à ce problème.

	Silhouette
K-Means	0.406751
cah	0.348168

Figure 12 - Performances de K-Means et CAH

7. Déploiement

Cette étape intervient généralement en fin de cycle dans la méthodologie CRISP et consiste principalement à déployer le (ou les) modèles retenus dans un environnement de production. Cette étape ne fait pas partie de l'étendue de l'énoncé du présent projet.

Annexe

A. Compréhension des données

```
categorical_cols1 = data.columns[data.dtypes==object].tolist()
print('categorical cols(modalities) =
\n',data[categorical_cols1].nunique())

print("") #Saut de ligne.

for col in data.columns:
    if data[col].nunique() == 3:
        print(col,' ',data[col].unique())
```

B. Désignation des articles

Article A: A Customer Churn Prediction using Pearson Correlation Function and K Nearest Neighbor Algorithm for Telecommunication Industry

Auteurs: Nilam Nur Amir Sjarif, Muhammad Rusydi Mohd Yusof, Doris HooiTen Wong, Suraya Ya'akob, Roslina Ibrahim, Mohd Zamri Osman.

Article B: Sequential Feature Selection in Customer Churn Prediction Based on Naive Bayes

Auteurs: Y. Yulianti, A. Saifudin

Article C: CustomerChurnPredictionSegmentationandFraudDetectioninTelecommunicationIndustry

Auteurs: Ahsan Rehman , Abbas Raza Ali.

C. Extrait du tableau récapitulatif des résultats Naïve Bayes

	ACC_naive	AUC_naive	SFS_ACC	SFS_AUC	SBS_ACC	SBS_AUC	SFFS_ACC	SFFS_AUC	SBFS_ACC	SBFS_AUC
1	0.82694	0.825593	0.861731	0.860511	0.861731	0.860511	0.861731	0.860511	0.861731	0.860511
2	0.82694	0.825593	0.894737	0.893730	0.894737	0.893730	0.894737	0.893730	0.894737	0.893730
3	0.82694	0.825593	0.902765	0.901828	0.901873	0.900999	0.902765	0.901828	0.902765	0.901828
4	0.82694	0.825593	0.897413	0.896601	0.908118	0.907527	0.905442	0.904484	0.900981	0.900042
5	0.82694	0.825593	0.898305	0.897515	0.900089	0.899687	0.900981	0.899999	0.907226	0.906656
6	0.82694	0.825593	0.897413	0.896687	0.900981	0.900644	0.900089	0.899128	0.908118	0.907613
7	0.82694	0.825593	0.900981	0.900343	0.906334	0.906000	0.905442	0.904527	0.906334	0.905871
8	0.82694	0.825593	0.895629	0.894945	0.906334	0.905957	0.906334	0.905484	0.906334	0.905871
9	0.82694	0.825593	0.894737	0.894117	0.906334	0.905957	0.907226	0.906742	0.897413	0.896687
10	0.82694	0.825593	0.895629	0.895031	0.906334	0.905914	0.908118	0.907613	0.896521	0.895945
11	0.82694	0.825593	0.900089	0.899429	0.894737	0.894547	0.905442	0.904957	0.896521	0.895945
14	0.82694	0.825593	0.900089	0.899730	0.892953	0.893106	0.900089	0.899730	0.904550	0.904387
15	0.82694	0.825593	0.892061	0.891633	0.882248	0.882567	0.902765	0.902601	0.892061	0.892320
16	0.82694	0.825593	0.892953	0.893020	0.879572	0.879910	0.904550	0.904172	0.889384	0.889578
17	0.82694	0.825593	0.887600	0.887793	0.859054	0.859489	0.885816	0.885492	0.882248	0.881492
18	0.82694	0.825593	0.888492	0.887546	0.863515	0.863242	0.887600	0.887535	0.875112	0.874609
19	0.82694	0.825593	0.893845	0.893031	0.855486	0.855273	0.887600	0.887836	0.874219	0.873695
20	0.82694	0.825593	0.888492	0.887761	0.851918	0.851918	0.886708	0.885804	0.876004	0.875523
21	0.82694	0.825593	0.886708	0.886105	0.842997	0.842218	0.886708	0.886105	0.877788	0.877179
22	0.82694	0.825593	0.876004	0.875566	0.842997	0.842476	0.874219	0.873652	0.842997	0.842476
23	0.82694	0.825593	0.872435	0.871996	0.839429	0.838476	0.872435	0.871996	0.839429	0.838476
24	0.82694	0.825593	0.872435	0.872297	0.839429	0.838476	0.872435	0.872297	0.839429	0.838476
25	0.82694	0.825593	0.852810	0.852789	0.838537	0.837605	0.852810	0.852789	0.838537	0.837605

Bibliographie

[1] Sarah WARDY, Ilham BERRADA, Double problématique du churn et du turnover pour les organisations: Définitions et état de l'art.

[2]https://www.kaggle.com/blastchar/telco-customer-churn

[3]https://developer.ibm.com/technologies/data-science/patterns/predict-customer-churn-using-watson-studio-and-jupyter-notebooks/#

[4]https://www.kaggle.com/annichingwang/telco-customer-churn-analysis

[5]https://community.ibm.com/community/user/businessanalytics/blogs/steven-macko/2019/07/11/telco-customer-churn-1113

[6]https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[7]https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/