Ferroni Sandro
Gassem Aymen
Moyo-Kamdem Auren
Promo : 2025
SCIA-G

Projet mutualisé : ML - BDA - DV

Analyse et prédiction des churns dans un service d'abonnement Internet.

Version 1 : 06/06/2024

Table des matières

Introduction

Un des plus grands problèmes dans l'économie des services internet est le désabonnement des clients, ce qui empêche les entreprises de pouvoir accroître leurs économies et ainsi leur développement.

De nos jours, le machine learning prend de plus en plus de place dans nos vies. Cet outil permet notamment de faire des analyses et des prédictions avec une précision bien supérieure à celle d'un humain, et ce, dans un temps bien plus rapide. Cependant, pour que ces prédictions soient justes et efficaces, elles doivent être construites de manière précise et sans erreurs. Pour cela, des outils tels que la gestion de grandes données (Big Data Analytics) et la visualisation de données (DataViz) nous permettent de comprendre les paramètres d'entrée dont notre modèle a besoin afin de l'entraîner dans les meilleures conditions.

Utiliser l'intelligence artificielle et plus précisément le machine learning, le big data analytics et la data visualisation pourrait ainsi permettre de prédire et détecter avec une grande précision les clients susceptibles de se désabonner de notre service internet. Cela permettrait de mettre en place des méthodes de marketing ciblées sur ce type de personnes dans l'objectif de conserver leur abonnement et ainsi accroître notre nombre d'abonnés.

C'est ce que nous avons essayé de faire dans le cadre de notre projet mutualisé. Pour cela, nous avons dans un premier temps effectué un traitement des données, puis nous avons analysé nos features dans l'objectif de pouvoir ensuite développer un modèle de machine learning permettant de faire des prédictions sur les clients. Nous avons par la suite évalué notre modèle, puis cherché à visualiser et interpréter nos résultats.

Traitement des données

Dans l'objectif de créer un modèle de Machine Learning bien entraîné, nous devons dans un premier temps effectuer un traitement des données afin de garder des enregistrements cohérents et complets.

Après avoir importé nos données, récupérées sur HuggingFace (https://huggingface.co/datasets/d0r1h/customer_churn), nous les avons stockées dans un dataframe Spark dans le but de mettre en pratique nos connaissances acquises en BDA. Nous avons commencé par analyser nos données en les affichant pour les découvrir dans un premier temps. Notre dataset est composé de 23 colonnes et 36 992 lignes. Nos features sont :

- age: âge de notre client [int]
- avg_frequency_login_days: moyenne de la fréquence de connexion par jour [string]
- avg_time_spent: moyenne du temps passé lors de la connexion [float]
- avg_transaction_value: moyenne du nombre de transactions [float]
- churn_risk_score: score de prédiction de risque de désabonnement [int]
- complaint_status: statut de la plainte [string]
- days_since_last_login: nombre de jours depuis la dernière connexion [int]
- feedback: retour d'expérience des clients [string]
- gender: genre du client [string]
- internet_option: options d'internet choisies [string]
- joined_through_referral: si le client a rejoint par une référence [string]
- joining_date: date de début d'abonnement du client [string]
- last_visit_time: temps passé durant la dernière connexion [string]
- medium_of_operation: moyen d'opération choisi par l'utilisateur [string]
- membership_category: catégorie d'adhésion du client [string]
- offer_application_preference: si l'utilisateur a une offre de préférence [string]
- past_complaint: si l'utilisateur a déjà fait une plainte [string]
- points_in_wallet: nombre de points cumulés par le client [float]
- preferred_offer_types: le type d'offre préféré de nos clients [string]
- referral_id: id de l'utilisateur [string]
- region_category: localisation de l'utilisateur [string]
- security_no: token de sécurité [string]
- used_special_discount: si l'utilisateur a eu droit à une remise [string]

Nous avons ensuite cherché le nombre de valeurs nulles présentes dans notre modèle :

- 3443 : points_in_wallet
- 288 : preferred_offer_types
- 5428 : region_category

Nous allons régler cela après avoir analysé toutes les valeurs de chaque colonne. On constate des valeurs aberrantes : valeurs négatives sur un nombre de jours, sur le temps passé et des '?', 'Error' et 'None' dans les valeurs de colonnes. Nous avons donc transformé toutes ces valeurs en None.

Après avoir supprimé les valeurs aberrantes et None pour les valeurs de type int ou float, nous avons fait le choix de définir les None des colonnes de type string en 'Unknown' afin de ne pas supprimer plus de valeurs et de ne pas influencer sur des valeurs en remplaçant par une valeur déjà existante.

Nous avons également choisi de supprimer 2 colonnes : security_no et referral_id. Puisque celles-ci n'auront aucun impact sur la prédiction de notre risque de désabonnement et permettent une anonymisation totale de nos utilisateurs pour conserver leur sécurité et leurs données confidentielles.

Puis avons fait le choix de créer 2 colonnes à la place de joining_date :

- month_joining_date: représentant le mois d'inscription de l'utilisateur dans le but de détecter des achats compulsifs dus aux soldes ou aux fêtes de Noël par exemple.
- joining_date: représentant le nombre de mois depuis l'abonnement.

Nous avons converti notre colonne last_visit_time en nombre de minutes passées.

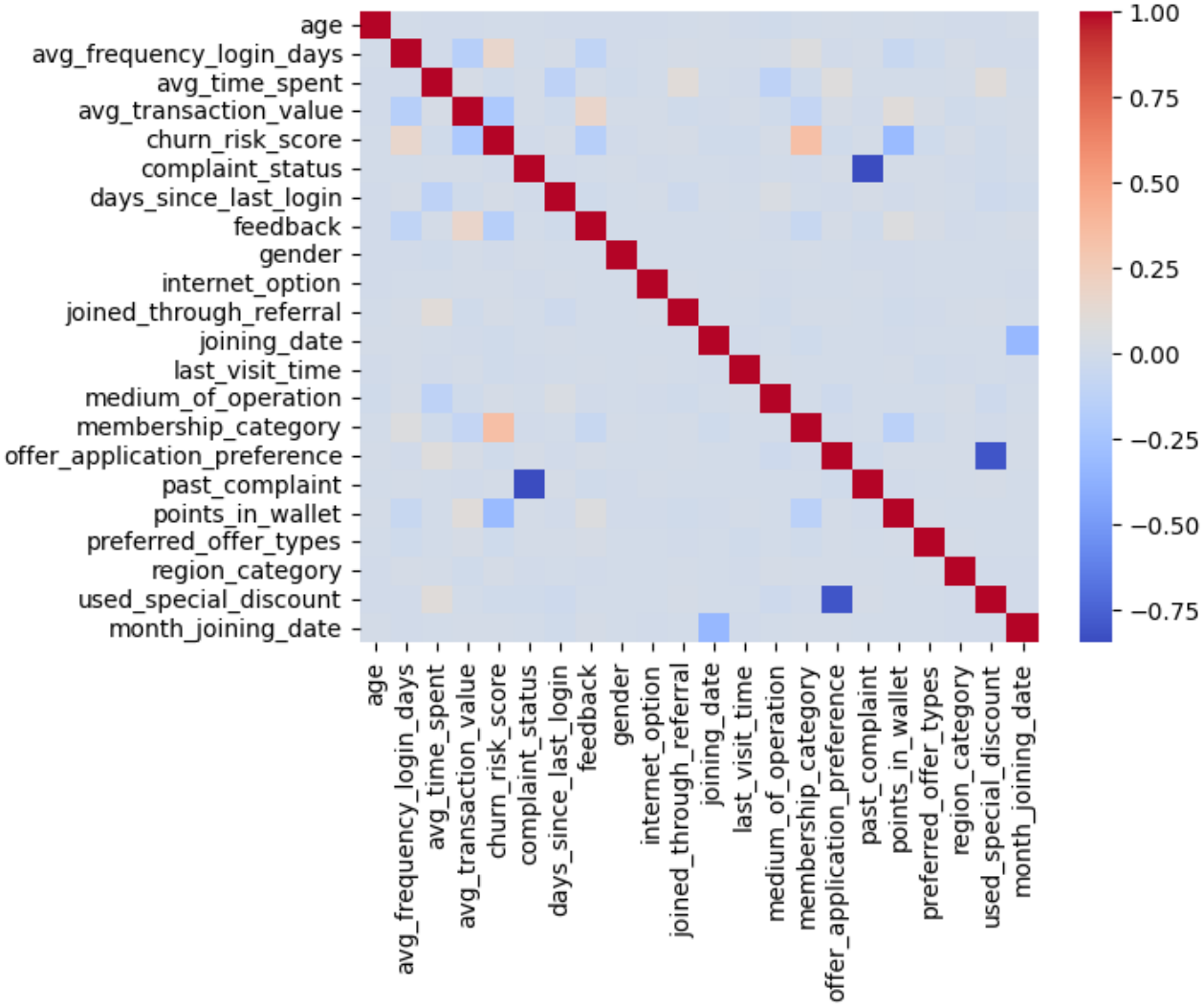
Et nous avons également créé un second dataframe composé uniquement de valeurs numériques où les string sont stockées avec un int dans une bibliothèque dans l'objectif de créer une matrice de corrélation dans la prochaine partie.

Maintenant que toutes nos données ont été triées correctement, nous pouvons passer à la partie analyse et visualisation de nos données.

Analyse, visualisation et exploration des données

Nos données sont à présent nettoyées. Nous pouvons alors commencer notre analyse sur l'influence de nos données sur la prédiction du risque de désabonnement de nos clients.

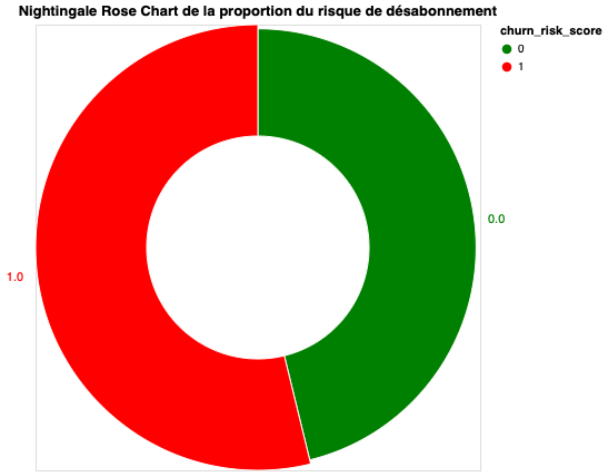
Nous avons commencé par afficher notre matrice de corrélation. On constate que 5 colonnes ont plus ou moins une influence sur notre valeur churn_risk_score : avg_frequency_login_days, avg_transaction_value, feedback, membership_category et points_in_wallet.



Matrice de corrélation

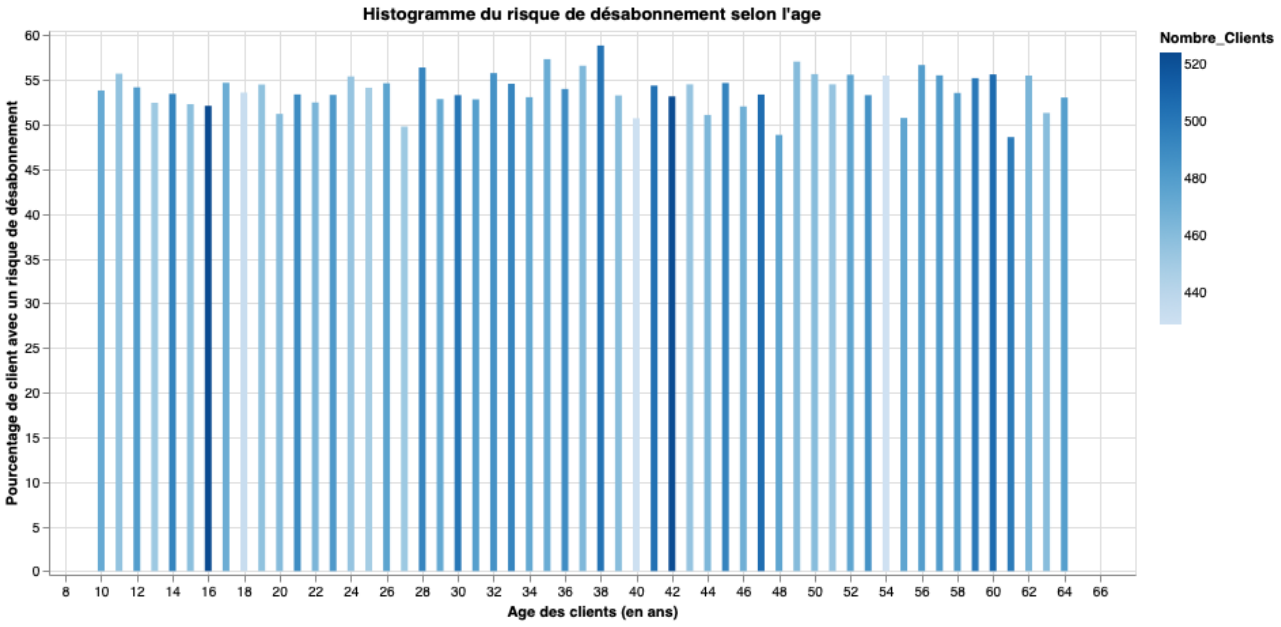
Pour nos prochaines visualisations, nous allons calculer la proportion de clients ayant un risque de désabonnement selon différents features.

Dans un premier temps, pour pouvoir comparer, nous allons afficher la proportion de clients ayant un risque de désabonnement dans notre dataset.

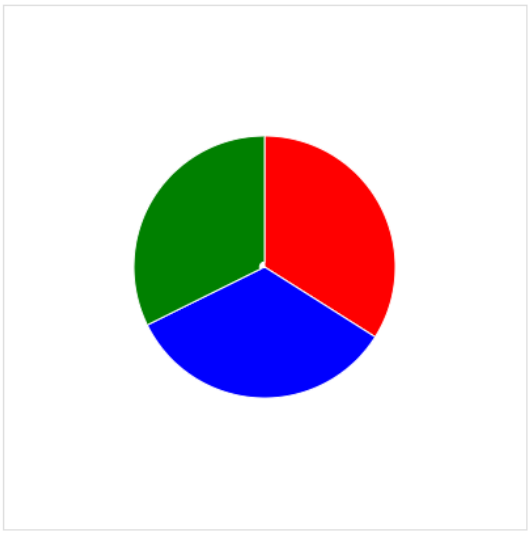


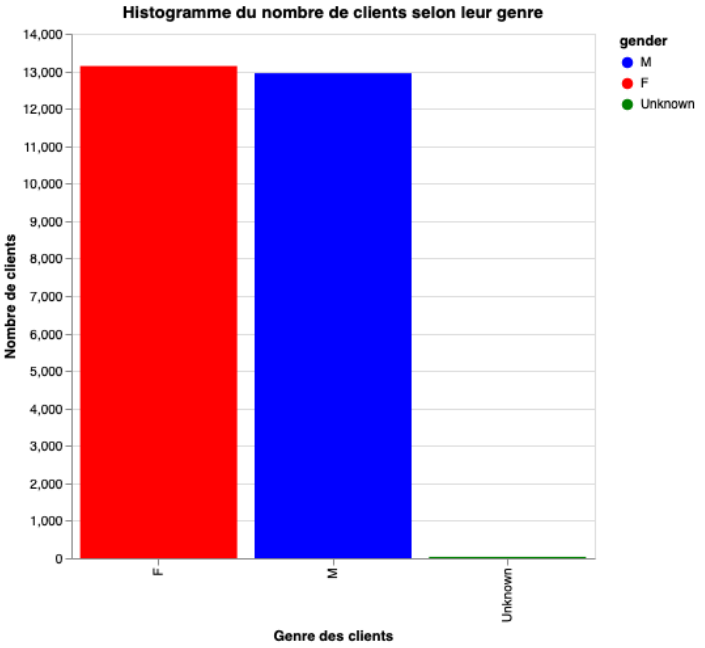
À noter que nos graphiques possèdent l'option tooltip permettant d'afficher les statistiques et chiffres lorsque l'on passe notre souris dessus.

Nos premières visualisations sont celles n'ayant pas de réelle corrélation avec notre prédiction. Nous avons commencé par comparer le risque de désabonnement selon l'âge. Mais nous constatons que cette feature ne nous donne pas de résultats concluants.

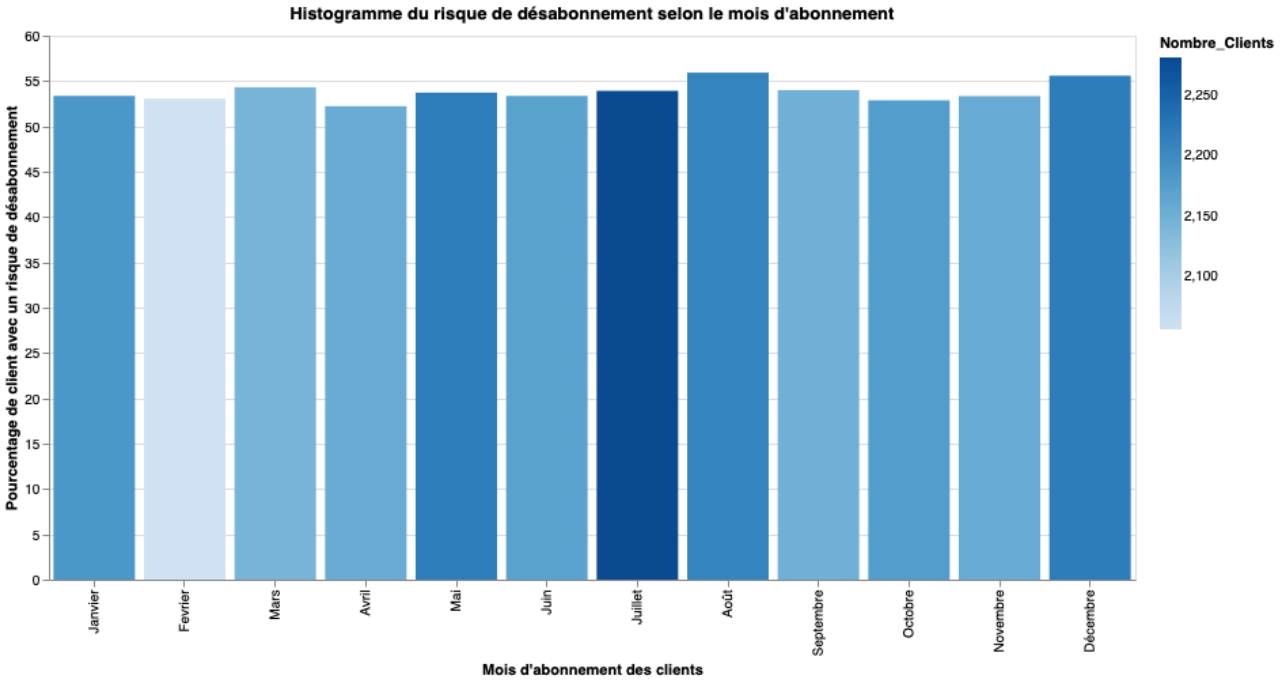


Ensuite, nous avons essayé de faire de même avec le genre de nos clients. Nous constatons des arcs de cercle de taille égale, ce qui signifie que les données le sont aussi. Ainsi, le genre, quelle que soit sa proportion que l'on voit à droite, n'influe pas significativement sur notre prédiction.

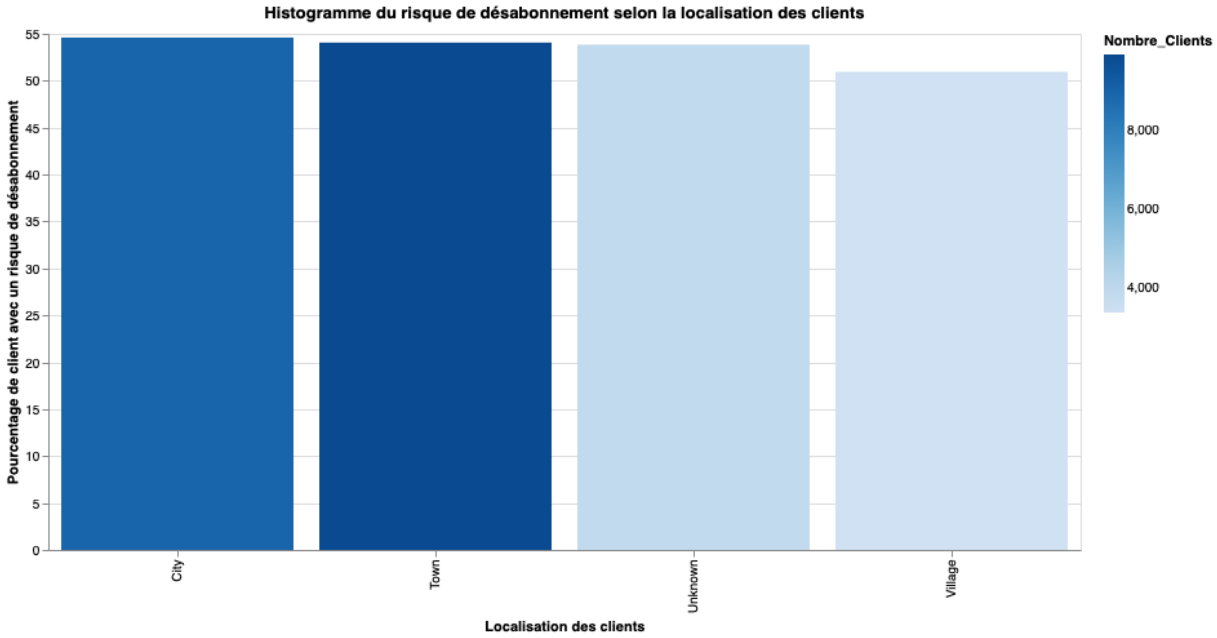




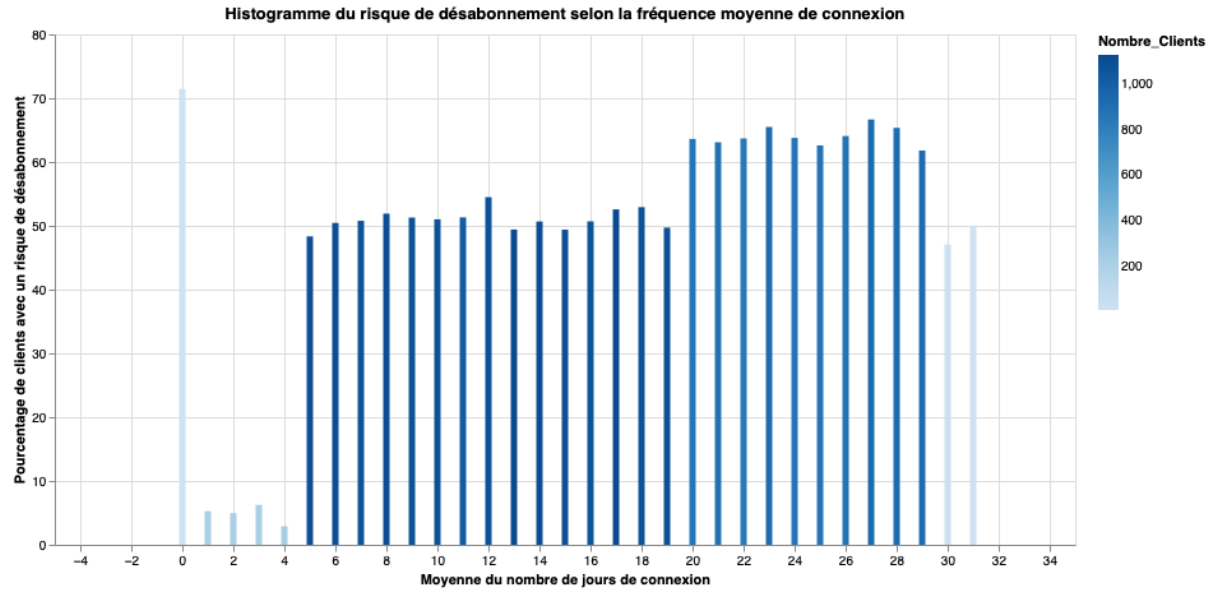
Par la suite, nous avons cherché à savoir si des achats compulsifs pouvaient rendre le risque de désabonnement plus élevé en regardant si les mois avec le plus d'inscriptions étaient aussi ceux avec le plus de risque. Nous ne constatons pas de grandes différences et plutôt des valeurs restant dans la moyenne.



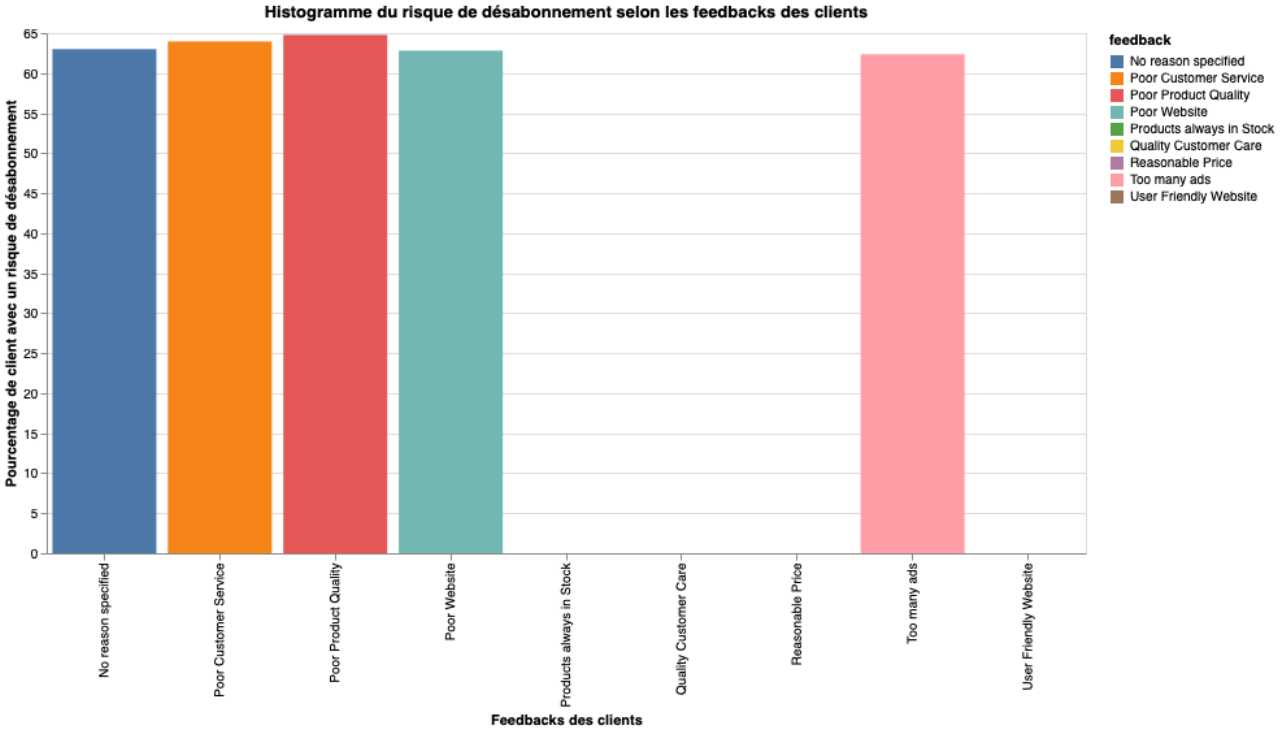
Il était alors possible que la localisation des utilisateurs ait une influence sur notre valeur churn. Cependant, mis à part le fait que les villageois soient à peine plus fidèles, on ne constate pas de grands écarts.



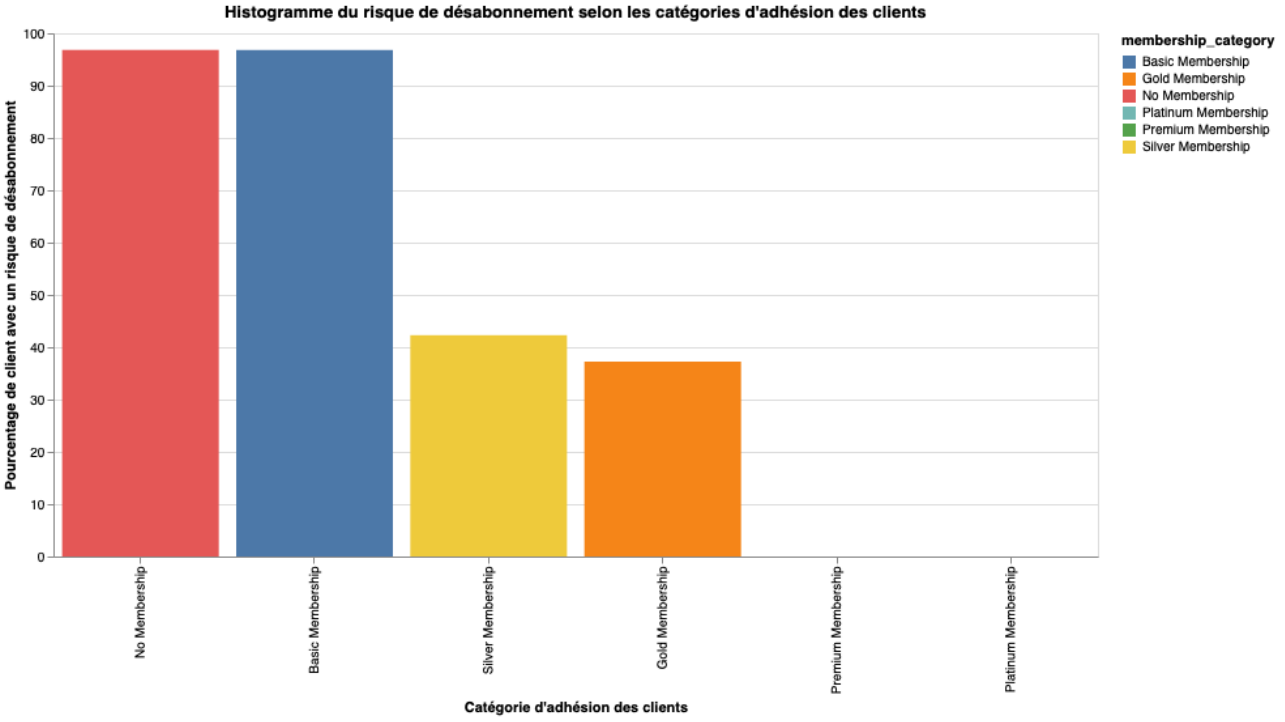
Nous nous sommes donc intéressés plus sérieusement à notre matrice de corrélation afin d'observer le réel impact des features ayant une grande influence sur notre valeur à prédire.
Nous avons commencé par observer les fréquences de connexions de nos utilisateurs et nous constatons que la plupart de nos valeurs se trouvent entre 5 et 29 jours, et que ces valeurs se divisent en deux parties distinctes : ceux ayant plus de 20 jours présentent un risque élevé, tandis que ceux à moins de 20 jours sont plus fidèles. Il est probable que les utilisateurs qui se connectent le plus cherchent des abonnements plus avantageux pour leurs besoins :



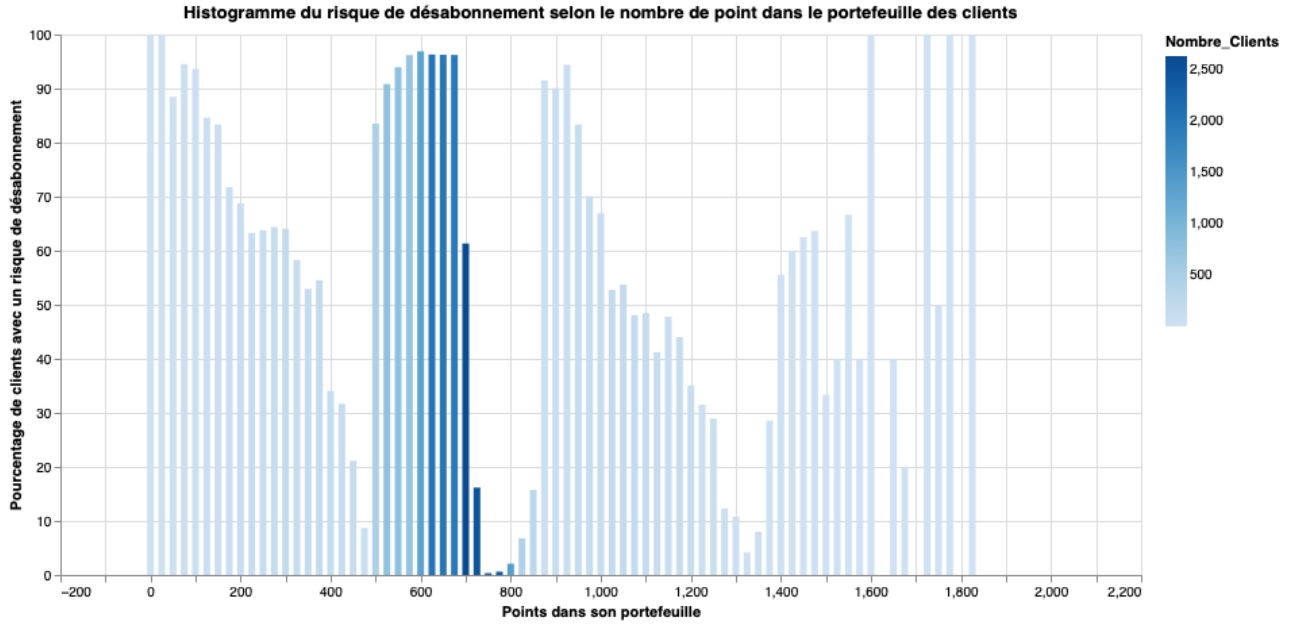
Les feedbacks ont aussi une grande corrélation. Comme on peut l'observer, les utilisateurs ayant des feedbacks positifs ont 0% de chances de quitter notre service, contrairement aux retours négatifs :



Les catégories d'adhésion sont aussi de bons indicateurs de fidélité. En effet, les utilisateurs avec les classes les plus élevées sont de moins en moins susceptibles d'interrompre leur abonnement, allant jusqu'à 0 client pour les 2 catégories les plus prestigieuses.



La dernière feature que nous allons vous présenter est celle du nombre de points de fidélité des utilisateurs. On constate une très grande concentration entre 450 points et 850. Entre eux, on constate un grand écart à 700 points, où les utilisateurs précédant ce chiffre ont tendance à mettre fin à leur abonnement, alors qu'au-dessus, la plupart des utilisateurs restent clients dans notre service.



Ainsi, avec toutes ces visualisations, nous avons pu comprendre et interpréter le comportement de nos features sur notre modèle, que nous allons présenter juste après. Cela nous permettra d'avoir une meilleure analyse de nos résultats et de comprendre comment ils sont influencés.

Description du problème sous forme ML

Maintenant que nos données sont nettoyées et analysées, nous allons pouvoir commencer à concevoir un modèle de prédiction dans le but de connaître le risque de désabonnement de nos clients.

Dans un premier temps, nous avons effectué une indexation et un encodage de nos données de type String à l'aide de StringIndexer, OneHotEncoder et Pipeline dans le but de transformer nos données en vecteurs (avec VectorAssembler), ce qui est nécessaire pour des modèles de machine learning sur un dataframe Spark. Ensuite, nous avons découpé notre jeu de données en 3 parties :

- TrainingSet : 75% = jeu de données pour entraîner nos modèles
- ValidationSet : 15% = jeu de données permettant de calculer les performances de nos modèles dans le but de valider leur réussite.
- TestSet : 10% = jeu de données de test permettant de vérifier le bon fonctionnement de notre modèle avant sa mise en production et permettant par la suite d'analyser les résultats.

Nos Datasets sont composés de toutes les données que nous avons mentionnées dans la partie traitement de données, filtrées et nettoyées. Notre objectif est donc de prédire de manière binaire si le client a un risque de se désabonner dans un futur proche de notre service internet. Notre Target est donc : churn_risk_score. Nous allons donc effectuer des entraînements sur différents modèles avec différents hyperparamètres dans le but de trouver le modèle de classification le plus efficace.

Évaluation de plusieurs modèles avec les bonnes métriques

Pour la partie training, nous avons fait le choix de comparer directement 6 modèles, le tout avec des hyperparamètres différents que nous avons fait varier par méthode de Grid Search. Nous avons choisi cette méthode principalement pour des questions de temps. En effet, notre notebook prend actuellement un peu plus d'une heure à s'exécuter. Nos 6 modèles sont :

- Logistic Regression :
  - regParam (paramètre de régularisation) : valeurs testées 0.1, 0.01
  - elasticNetParam (paramètre ElasticNet) : valeurs testées 0.0, 0.5, 1.0
- Random Forest :
  - numTrees (nombre d'arbres) : valeurs testées 10, 20, 30
  - maxDepth (profondeur maximale) : valeurs testées 5, 10, 15
- Gradient Boosted Trees :
  - maxIter (nombre d'itérations) : valeurs testées 10, 20, 30
  - maxDepth (profondeur maximale) : valeurs testées 5, 10, 15
- Decision Tree :
  - maxDepth (profondeur maximale) : valeurs testées 5, 10, 15
  - maxBins (nombre de bacs) : valeurs testées 32, 64
- Naive Bayes :
  - smoothing (paramètre de lissage) : valeurs testées 0.0, 1.0, 10.0
- LinearSVC :
  - regParam (paramètre de régularisation) : valeurs testées 0.1, 0.01

Pour chaque modèle, après avoir entraîné, nous avons évalué les performances de nos modèles sur nos données de validation avec MulticlassClassificationEvaluator et BinaryClassificationEvaluator, permettant ainsi de calculer l'accuracy, la précision, le recall, le F1-score, le score ROC, une moyenne de la cross-validation et son écart type. Vous retrouverez nos métriques dans le tableau :

Tableau des métriques des performances de nos modèles

Modèle	Accuracy	Precsion	Recall	F1SCORE	ROC AUC	CV	Ecart type
LogisticRegression	0.845081	0.862164	0.845081	0.844851	0.956112	0.947996	0.011817
RandomForest	0.935451	0.935887	0.935451	0.935512	0.975889	0.967596	**0.005114**
GradientBoostedTrees	**0.939324**	**0.940108**	**0.939324**	**0.939400**	**0.977154**	**0.968434**	0.007340
DecisionTree	0.926414	0.926471	0.926414	0.926337	0.959165	0.890574	0.088222
NaiveBayes	0.782856	0.782593	0.782856	0.782288	0.699251	0.691863	0.006341
LinearSVC	0.848180	0.875806	0.848180	0.847330	0.948130	0.939414	0.009945

Pour choisir notre meilleur modèle, nous avons décidé de nous concentrer sur le meilleur F1-score en nous assurant que les autres performances soient également très bonnes.

Dans notre cas, le meilleur modèle est donc GradientBoostedTrees avec les hyperparamètres suivants :

- cacheNodeIds: False
- checkpointInterval: 10
- featureSubsetStrategy: all
- featuresCol: scaled_features
- impurity: variance
- labelCol: label
- lossType: logistic
- maxBins: 32
- maxDepth: 5
- maxIter: 30
- maxMemoryInMB: 256
- minInfoGain: 0.0
- minInstancesPerNode: 1
- minWeightFractionPerNode: 0.0
- predictionCol: prediction
- probabilityCol: probability
- rawPredictionCol: rawPrediction
- seed: -5727026145477880835
- stepSize: 0.1
- subsamplingRate: 1.0
- validationTol: 0.01

Dans le but d'améliorer nos performances, nous avons mis en place un modèle d'ensemble learning avec nos 3 meilleurs modèles : DecisionTree, RandomForest et GradientBoostedTrees. Nous avons donc sur chaque modèle testé toutes nos données de validation et avons comparé leurs résultats. Notre modèle d'ensemble learning renvoie ainsi la prédiction la plus représentée parmi les 3 prédictions. Nous avons ensuite calculé les performances de ce modèle :

Tableau des métriques des performances de l'ensemble learning

Accuracy	Precsion	Recall	F1SCORE	ROC AUC
0.93751613736122	0.9381054205114	0.9375161373612	0.93758521729403	0.938395223397464

On constate ainsi que notre modèle d'ensemble learning est à peine moins performant que notre meilleur modèle. Nous avons donc décidé de garder comme modèle final le GradientBoostedTrees défini précédemment.

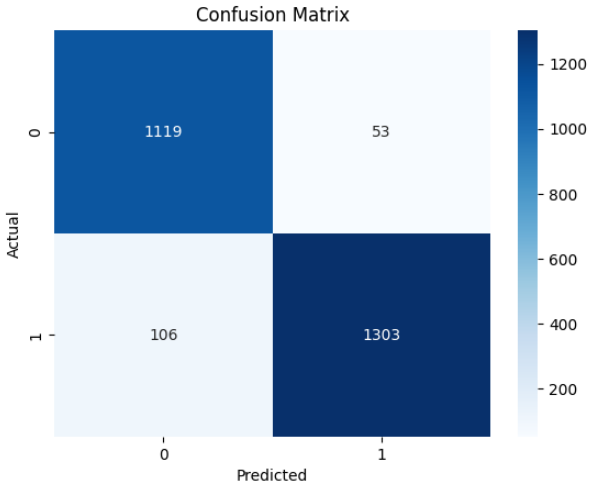
Visualiser et interpréter les résultats

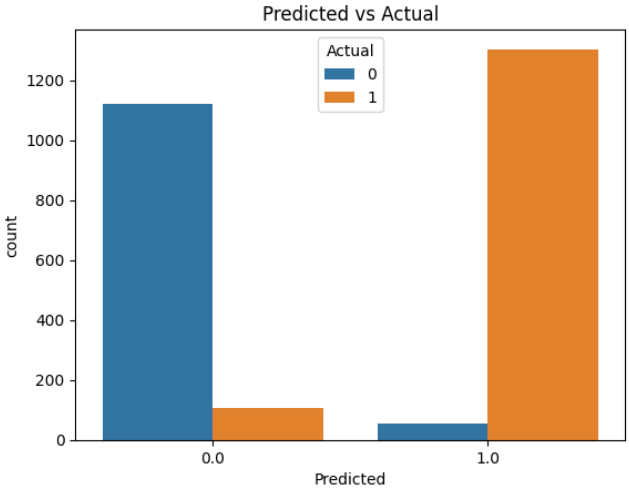
Dans l'objectif de simuler une mise en production, nous avons fait le choix d'effectuer une batterie de tests sur notre meilleur modèle déjà entraîné. Ces données sont issues de notre dataset de base, comme indiqué précédemment.
L'avantage de tester sur notre Dataset est que l'on peut à nouveau calculer nos performances. Sur nos tests, nous obtenons :

Tableau des métriques des performances sur les tests

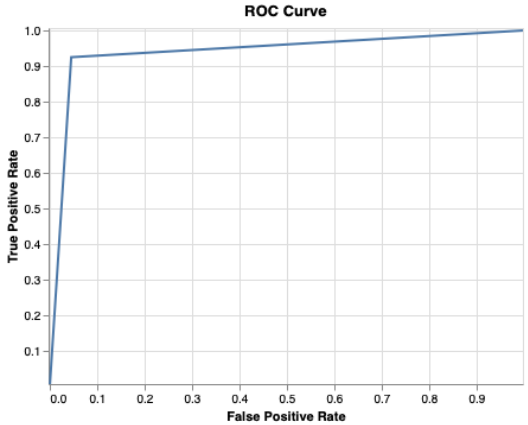
Accuracy	Precsion	Recall	F1SCORE	ROC AUC
0.93839597055405	0.9393702396457	0.9383959705540	0.93848661467523	0.939773748477002

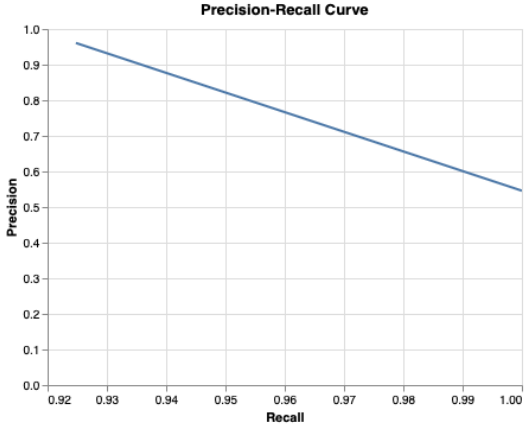
On constate que l'on obtient des résultats assez similaires à nos données de validation. Nous avons donc décidé d'afficher une matrice de confusion afin de mieux visualiser nos données prédites en comparaison de leur vraie prédiction. De plus, un histogramme montre que l'on a plus tendance à se tromper sur des clients fidèles, qui auront donc la chance d'être contactés par erreur par notre équipe marketing qui leur proposera des offres pour les fidéliser encore plus.





Enfin, pour finir, nous avons décidé d'afficher notre courbe ROC et notre courbe de Précision-Recall, mettant en avant les bonnes performances de notre modèle de prédiction qui possède une précision de 94 %.





Conclusions / Perspectives

Notre projet consistait donc à prédire le risque que les clients d'une entreprise de service internet se désabonnent de celle-ci. Pour ce faire, nous avons effectué un nettoyage et une analyse approfondis de nos données en utilisant notamment nos connaissances acquises lors de nos cours de Machine Learning et de Big Data, en utilisant des outils tels que Spark. Par la suite, nous avons effectué des visualisations dans le but de mieux comprendre nos données, notamment leur influence sur le risque de désabonnement, en utilisant notamment une matrice de corrélation. Nous avons ainsi pu définir un problème de Machine Learning afin de résoudre celui-ci et prédire à l'avance le risque de perte de nos clients.
Nous avons ensuite entraîné et comparé plusieurs modèles avec différents hyperparamètres après avoir divisé notre jeu de données en 3 : Training, Validation et Test. Nous avons remarqué que notre meilleur modèle était GradientBoostedTrees, après avoir comparé ses métriques de performances avec les autres méthodes. Nous avons privilégié le score F1 pour faire notre choix, mais avons remarqué que ce modèle était plus performant que les autres sur tous ses métriques. Nous l'avons ainsi conservé dans le but de préparer notre mise en production où nous avons essayé un ensemble de tests sur celui-ci et avons obtenu des résultats très favorables avec une précision de 94%. Nous avons ainsi pu montrer nos performances à l'aide de dernières visualisations graphiques.

Nous sommes conscients que ce projet possède des perspectives d'amélioration. Il aurait été judicieux de faire plus d'essais sur le dataset, en supprimant plus de données, en ajoutant d'autres données, en modifiant nos features d'entrée. Nous aurions également pu faire des visualisations sur tous les features, mais avons choisi de ne pas le faire pour éviter un encombrement dans notre notebook et rapport sur des visualisations moins pertinentes. Côté Machine Learning, nous aurions pu essayer davantage de modèles et utiliser la méthode de Random Search pour essayer plus de combinaisons d'hyperparamètres, le tout dans l'optique d'obtenir de meilleurs résultats.

Néanmoins, nos 94% de précision et nos tests effectués montrent que l'objectif de prédire le risque de désabonnement de nos clients est atteint. Ce modèle permet ainsi aux équipes de marketing de connaître les utilisateurs susceptibles de résilier leur abonnement dans un futur proche et permettant ainsi de leur proposer des offres alléchantes afin de les convertir en clients fidèles, permettant à notre entreprise d'accroître son développement.

Ferroni Sandro
Gassem Aymen
Moyo-Kamdem Auren
SCIA-G
PROMO 2025