



PROJET ACADÉMIQUE

PRÉDICTION DU CHURN CLIENT PAR DES TECHNIQUES DE MACHINE LEARNING

Sous l'encadrement de :

Pr. EL AZHARI KHADIJA

Pr. BENJBARA Chaimae

Préparé par :

Ilyass BOUCHNAFA -- 22014351

Année Universitaire 2025-2026



Remerciements

Au terme de ce travail, je souhaite adresser mes sincères remerciements à **Pr. BENJBARA Chaimae** et **Pr. EL AZHARI Khadija** pour nous avoir donné l'opportunité de réaliser ce projet académique.

Leur encadrement, leur disponibilité et leurs orientations pédagogiques nous ont permis de mettre en pratique les connaissances théoriques acquises, de renforcer notre esprit d'analyse et de développer une méthodologie de travail rigoureuse.

Table des matières

I.	INTRODUCTION ET CADRAGE DU PROJET	1
1.1.	Contexte du projet :	1
1.2.	Description des données :	1
1.3.	Problématique :	2
1.4.	Objectifs du projet :	2
1.5.	Méthodologie et démarche scientifique :	2
II.	EXPLORATION ET ANALYSE DE LA QUALITÉ DES DONNÉES (EDA)	3
2.1.	Dictionnaire des données et consolidation	3
2.2.	Audit de la structure et de la qualité :	4
2.3.	Analyse de la distribution cible :	6
2.4.	Analyse des Facteurs d'Influence :	7
2.5.	Analyse des corrélations :	10
III.	PRÉTRAITEMENT DES DONNÉES (PREPROCESSING)	10
3.1.	Nettoyage et standardisation :	10
3.2.	Encodage des variables :	11
3.3.	Conversion et Matrice des Données :	11
3.4.	Séparation, Normalisation et Sauvegarde :	12
IV.	MODÉLISATION ET ÉVALUATION DES PERFORMANCES	12
4.1.	Approches linéaires et probabilistes (Logistic & Naive Bayes) :	13
4.2.	Approches non-linéaires (KNN & Arbre de Décision) :	14
4.3.	Méthodes Ensemblistes (Random Forest et XGBoost) :	15
4.5.	Choix du Modèle et Validation Finale :	15
4.6.	Interprétabilité et Feature Importance :	17
V.	ARCHITECTURE TECHNIQUE ET INDUSTRIALISATION	18
VI.	DÉPLOIEMENT DE L'INTERFACE UTILISATEUR (STREAMLIT)	19
	CONCLUSION GÉNÉRALE	22
	WEBOGRAPHIE	23

I. INTRODUCTION ET CADRAGE DU PROJET

1.1. Contexte du projet :

Dans un environnement économique numérisé et saturé, la fidélisation de la clientèle s'est imposée comme un levier de croissance plus critique que la simple conquête de parts de marché. L'attrition client, ou "Churn", ne désigne pas seulement la fin d'un contrat, mais l'échec d'une relation commerciale face à des consommateurs devenus extrêmement volatils et exigeants.

L'impact économique de ce phénomène est souvent sous-estimé. Si la littérature marketing ([Harvard Business Review](#)) établit que l'acquisition d'un nouveau client coûte entre **5** et **25** fois plus cher que la rétention d'un client existant, le coût réel inclut également la perte de la "Valeur Vie Client" (*Customer Lifetime Value*). Un client qui part, c'est un flux de revenus futurs qui disparaît instantanément. À l'inverse, une augmentation du taux de rétention de seulement **5 %** peut générer une hausse des profits allant de **25 % à 95 %**.

Dès lors, le taux de Churn devient un indicateur de bonne santé vitale pour l'entreprise. Cependant, les méthodes traditionnelles d'analyse *a posteriori* (constater le départ et en chercher les causes) sont désormais obsolètes car trop tardives. L'enjeu n'est plus de comprendre pourquoi un client est parti hier, mais d'identifier qui risque de partir demain. Les entreprises doivent opérer un changement de paradigme, passant d'une gestion réactive des résiliations à une stratégie proactive de rétention, pilotée par la donnée.

1.2. Description des données :

Pour mener à bien cette étude empirique, nous nous sommes appuyés sur un jeu de données public (issu de la plateforme [Kaggle](#)), constituant un échantillon représentatif de la clientèle. Ce dataset multidimensionnel regroupe un ensemble de variables explicatives (*features*) structurées en trois catégories majeures :

- **Données Sociodémographiques** : Profilage du client incluant l'âge, le genre et la zone géographique.

- **Données Comportementales** : Métriques d'usage telles que la fréquence de connexion, l'historique des interactions avec le support technique et les incidents de paiement.
- **Données Contractuelles et Transactionnelles** : Caractéristiques de l'abonnement souscrit, durée d'engagement et montant total des dépenses générées.

L'objectif est de prédire la variable cible (*target*) nommée "Churn". Il s'agit d'une variable binaire utilisée dans un contexte d'apprentissage supervisé, où la valeur **1** signale un client ayant résilié son contrat (classe positive) et la valeur **0** un client actif (classe négative).

1.3. Problématique :

La volumétrie et la complexité de ces données rendent toute analyse manuelle inopérante et coûteuse en temps. L'enjeu ne réside plus dans la simple collecte de l'information, mais dans sa valorisation intelligente. La problématique centrale qui structure ce projet se formule donc ainsi :

"Comment l'exploitation des données comportementales historiques à l'aide de techniques d'apprentissage automatique permet-elle d'anticiper les risques de rupture de contrat et d'améliorer les stratégies de fidélisation client ?"

1.4. Objectifs du projet :

Ce projet poursuit une double finalité, à la fois analytique et opérationnelle. Il s'agit, dans un premier temps, d'identifier les leviers d'action en déterminant les facteurs prépondérants influençant la décision de départ (Feature Importance). Sur cette base, l'objectif central est de concevoir et d'entraîner un modèle de Machine Learning robuste, capable de classifier avec précision les clients selon leur risque d'attrition. Enfin, pour rendre ces prédictions concrètement exploitables par les équipes métier, le projet aboutira au déploiement d'une interface web interactive d'aide à la décision, permettant la mise en place d'une stratégie de rétention proactive.

1.5. Méthodologie et démarche scientifique :

Pour mener à bien ce projet, nous avons suivi une approche progressive structurée en cinq étapes clés, allant de la donnée brute à l'application finale.

Le processus débute par l'extraction des données depuis la plateforme [Kaggle](#). Ces données alimentent ensuite notre pipeline de Machine Learning automatisé, qui se charge de nettoyer l'information, d'entraîner les différents algorithmes et d'évaluer leurs performances.

Une fois le meilleur modèle identifié, nous procédons à sa sauvegarde (sérialisation) pour figer ses connaissances. Enfin, la dernière étape consiste à intégrer ce modèle dans une interface interactive, rendant ainsi l'outil directement utilisable pour la prise de décision.

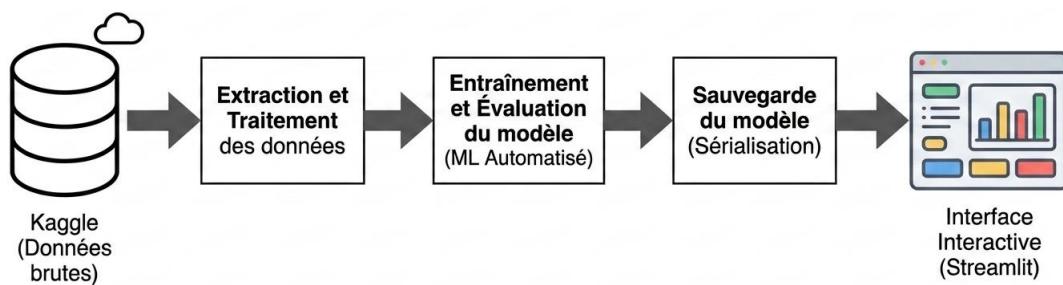


Figure 1 : Pipeline méthodologique, de la collecte Kaggle au déploiement de l'interface

II. EXPLORATION ET ANALYSE DE LA QUALITÉ DES DONNÉES (EDA)

Avant d'entamer l'exploration visuelle, nous avons procédé à une vérification rigoureuse de la structure et de la qualité des données pour garantir la fiabilité de nos modèles.

2.1. Dictionnaire des données et consolidation

La première étape a consisté à consolider nos sources d'information en concaténant les deux jeux de données disponibles afin d'obtenir une base unifiée et cohérente.

```

train_df = pd.read_csv("../data/raw/customer_churn_dataset_training.csv")
test_df = pd.read_csv("../data/raw/customer_churn_dataset-testing.csv")

# Concatenation des deux datasets
df = pd.concat([train_df, test_df], axis=0)

df.to_csv('../data/raw/churn.csv', index=False)

df.head()
✓ 5.2s

```

Python

CustomerID	Age	Gender	Tenure	Usage Frequency	Support Calls	Payment Delay	Subscription Type	Contract Length	Total Spend	Last Interaction	Churn	
0	2.0	30.0	Female	39.0	14.0	5.0	18.0	Standard	Annual	932.0	17.0	1.0
1	3.0	65.0	Female	49.0	1.0	10.0	8.0	Basic	Monthly	557.0	6.0	1.0
2	4.0	55.0	Female	14.0	4.0	6.0	18.0	Basic	Quarterly	185.0	3.0	1.0
3	5.0	58.0	Male	38.0	21.0	7.0	7.0	Standard	Monthly	396.0	29.0	1.0
4	6.0	23.0	Male	32.0	20.0	5.0	8.0	Basic	Monthly	617.0	20.0	1.0

Figure 1 : Consolidation des données et aperçu du dataset unifié.

Le dataset final comprend les variables suivantes :

Variable	Type	Description
CustomerID	Identifiant	Identifiant unique pour chaque client
Age	Numérique	Âge du client
Gender	Catégoriel	Genre (Homme / Femme)
Tenure	Numérique	Nombre de mois depuis l'inscription du client
Usage Frequency	Numérique	Fréquence d'utilisation du service par mois
Support Calls	Numérique	Nombre d'appels au service client
Payment Delay	Numérique	Retard de paiement (en jours)
Subscription Type	Catégoriel	Type d'abonnement (Basic / Standard / Premium)
Contract Length	Catégoriel	Durée du contrat (Mensuel / Trimestriel / Annuel)
Total Spend	Numérique	Montant total dépensé par le client
Last Interaction	Numérique	Jours écoulés depuis la dernière interaction
Churn	Cible (Target)	0 = Client Actif, 1 = Client Parti (Churn)

Tableau 1 : Dictionnaire des variables (Feature Description)

2.2. Audit de la structure et de la qualité :

Nous avons analysé la cardinalité de chaque colonne (nombre de valeurs uniques). Cette étape nous a permis de distinguer rapidement les variables catégorielles (ex: *Subscription Type* contient 3 valeurs uniques) des variables continues (ex: *Total*

Spend). Cela est indispensable pour choisir les bonnes méthodes d'encodage par la suite.

```
# Les valeurs uniques des variables catégorielles
for col in df.select_dtypes(include=['object']).columns:
    print(f'{col}: {df[col].unique()}')


# Nombre de valeurs uniques par colonne de type float64
for col in df.select_dtypes(include=['float64']).columns:
    print(f'{col}: {len(df[col].unique())}')


Age: 49
Tenure: 61
Usage Frequency: 31
Support Calls: 12
Payment Delay: 32
Total Spend: 68364
Last Interaction: 31
Churn: 3
```

Figure 2 : Analyse de la cardinalité et distinction des variables (Catégorielles vs Continues)

La fiabilité de nos prédictions repose sur la propreté des données initiales. Nous avons débuté par un audit de l'intégrité du fichier. L'analyse a révélé l'existence d'une unique ligne contenant des valeurs manquantes (*NaN*). Compte tenu de ce volume négligeable, nous avons procédé à la suppression de cet enregistrement. Parallèlement, la vérification des doublons a confirmé l'unicité de chaque client, garantissant une base saine.

```
# Afficher les lignes avec des valeurs manquantes
print("\nLignes avec des valeurs manquantes:")
display(df.isna().any(axis=1))

print("Nombre de doublons: {df.duplicated().sum()}")
# Aucun doublon détecté
```

Figure 3 : Identification des valeurs manquantes et vérification des doublons.

Dans un second temps, nous avons analysé la distribution des variables numériques via des diagrammes en boîte (*Boxplots*) afin de détecter d'éventuelles anomalies.

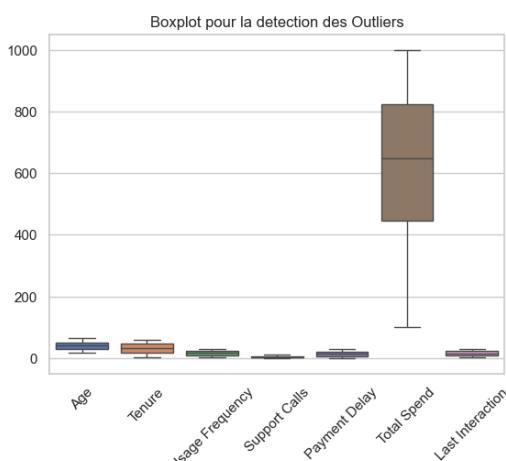


Figure 4 : Analyse des distributions par Boxplot

L'examen de la *Figure 3* met en évidence des valeurs très élevées pour la variable Total Spend. Une analyse approfondie montre qu'il ne s'agit pas d'erreurs de saisie, mais de clients à forte contribution ayant réalisé des dépenses importantes. Ces profils étant stratégiques pour l'analyse du chiffre d'affaires, leur suppression constituerait une perte d'information critique. Nous avons donc validé la conservation de l'intégralité de ces données pour l'entraînement, ce que confirme la cohérence globale validée par la méthode statistique IQR ci-dessous.

```
# Détection des outliers

# Calcule global des bornes (Q1, Q2, Q3) pour toutes les colonnes d'un coup
Q1 = df[numerical_cols].quantile(0.25)
Q3 = df[numerical_cols].quantile(0.75)
IQR = Q3 - Q1

mask_outliers = (df[numerical_cols] < (Q1 - 1.5 * IQR)) | (df[numerical_cols] > (Q3 + 1.5 * IQR))

print("Nombre d'outliers détectés par variable :")
print(mask_outliers.sum().sort_values(ascending=False))

# Pas d'outliers
✓ 0.1s
```

Python

```
Nombre d'outliers détectés par variable :
Age          0
Tenure        0
Usage Frequency 0
Support Calls 0
Payment Delay 0
Total Spend   0
Last Interaction 0
dtype: int64
```

Figure 5 : Validation statistique par la méthode IQR

2.3. Analyse de la distribution cible :

Enfin, nous avons examiné la proportion de clients ayant quitté l'entreprise par rapport à ceux qui sont restés.

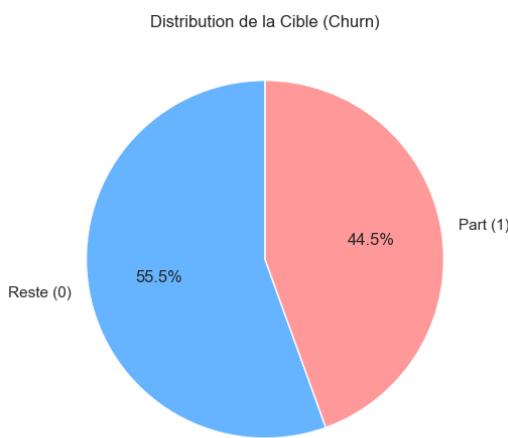


Figure 6 : Répartition de la variable cible (Churn vs Non-Churn)

Cette visualisation est capitale : elle nous informe si nous avons affaire à un problème de classes déséquilibrées (*Imbalanced Dataset*), ce qui influencerait le choix de nos métriques d'évaluation (privilégier le Rappel/Recall plutôt que l'Accuracy).

2.4. Analyse des Facteurs d'Influence :

Dans le but d'identifier les leviers d'action pour réduire l'attrition, nous avons mené une analyse bivariée exhaustive en croisant la variable cible (Churn) avec l'ensemble des caractéristiques clients. Cette démarche nous a permis d'examiner d'abord les comportements par segments (analyse catégorielle) avant de mesurer les liens statistiques bruts (analyse quantitative), offrant ainsi une vision complète des profils à risque.

Concernant les variables catégorielles, l'analyse visuelle des distributions révèle des disparités comportementales frappantes qui nous permettent de dresser un portrait précis du client volatile. Comme l'illustrent les figures ci-dessous, la durée d'engagement apparaît comme le facteur le plus déterminant : les contrats mensuels génèrent un taux de départ massif, tandis que les engagements annuels favorisent une quasi-fidélité. Ce constat s'accompagne d'une anomalie démographique sur le segment féminin, qui présente un taux de désabonnement bien supérieur à celui des hommes, ainsi qu'une performance décevante des offres d'entrée de gamme (Basic et Standard) qui retiennent moins bien les clients que l'offre Premium.

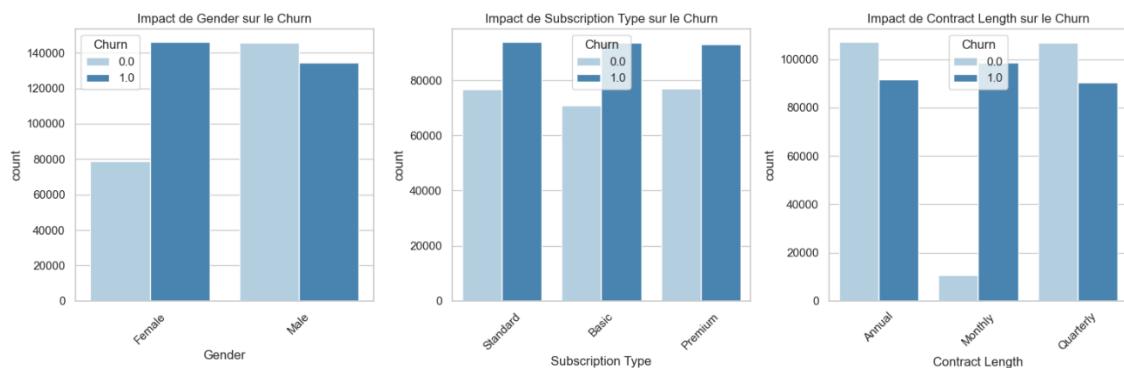


Figure 7 : Tendances de désabonnement par Contrat, Genre et Abonnement

En complément de l'analyse catégorielle, nous avons examiné la distribution des variables numériques. Un premier constat méthodologique s'impose à la lecture des graphiques : les distributions présentent des formes très segmentées avec des seuils de rupture nets (aspect en "escalier"). Cette structure particulière suggère que les données pourraient être issues de règles de gestion strictes, rendant les frontières de décision extrêmement lisibles.

L'analyse conjointe des indicateurs financiers et du support client révèle des facteurs discriminants majeurs. D'une part, le montant total des dépenses (Total Spend) agit

comme un filtre de fidélité : une séparation quasi parfaite s'opère autour de 500 unités monétaires, seuil au-delà duquel la fidélité est acquise. D'autre part, la gestion des incidents montre des points de non-retour critiques : le risque de départ explose dès qu'un client dépasse 20 jours de retard de paiement ou effectue plus de 5 appels au support technique. Ces valeurs agissent comme des "bascules" : avant ce seuil, le client est patient ; après, la rupture est quasi systématique.



Figure 8 : Impact des dépenses, des retards de paiement et des appels au support

Un autre signal d'alerte critique concerne l'activité du compte. La variable « Last Interaction » démontre que le silence est précurseur de rupture. Dès qu'un client cesse d'interagir avec les services de l'entreprise pendant plus de 15 jours, la probabilité de churn devient maximale.

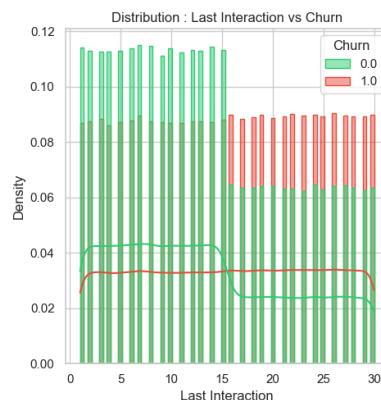


Figure 9 : Impact de l'inactivité (Jours depuis la dernière interaction)

En parallèle, la variable Age offre une perspective démographique nuancée. Contrairement aux variables précédentes qui fonctionnent par seuils stricts, l'âge montre que la fidélité se concentre essentiellement sur la tranche active (30-50 ans). À l'inverse, les populations plus jeunes (<30 ans) et les seniors (>50 ans) affichent une

propension au départ beaucoup plus élevée, signalant une inadéquation probable de l'offre actuelle pour ces deux extrémités démographiques.

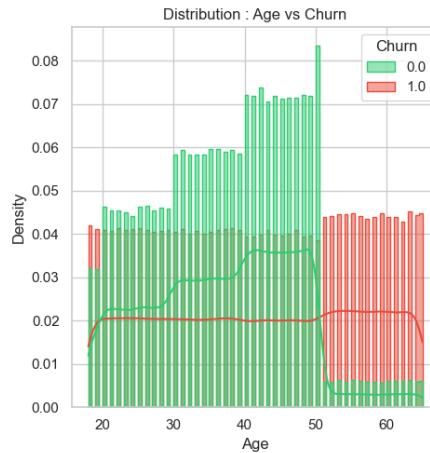


Figure 10 : Distribution des désabonnements par tranche d'âge

Enfin, il est crucial de noter que certaines variables, pourtant intuitives, ne semblent pas influencer le désabonnement dans ce contexte. L'analyse de l'ancienneté (Tenure) et de la fréquence d'utilisation (Usage Frequency) montre une superposition quasi parfaite des courbes de fidélité et de départ. Que le client soit nouveau ou ancien, ou qu'il utilise le service intensivement ou non, cela ne permet pas de prédire son départ. Ces variables, considérées comme du "bruit", auront probablement un poids négligeable dans notre future modélisation.

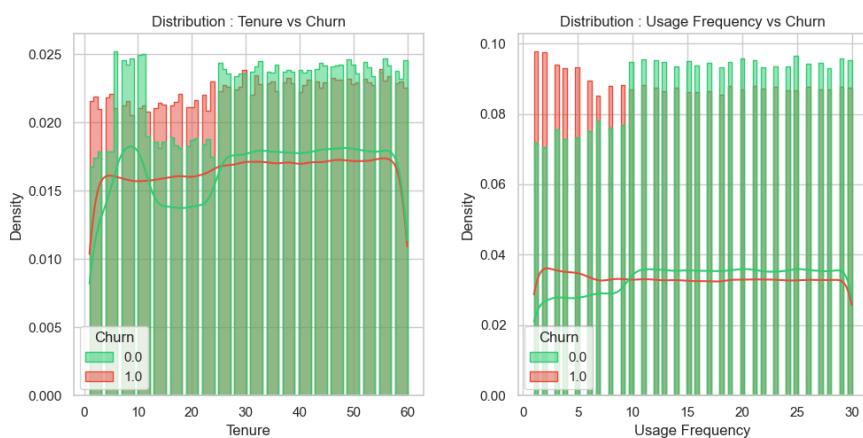


Figure 11 : Variables non-discriminantes (Ancienneté et Fréquence)

L'analyse croisée nous permet désormais de définir avec précision le "profil robot" du client à haut risque : il s'agit prioritairement d'un utilisateur jeune ou senior, ayant dépensé moins de 500\$, accumulant des retards de paiement, inactif depuis plus de 15 jours et ayant sollicité le support à de multiples reprises (plus de 5 fois).

2.5. Analyse des corrélations :

Nous avons analysé les interactions entre les variables à l'aide d'une matrice de corrélation pour identifier les liens les plus forts avec la variable cible (Churn).



Figure 12 : Matrice de corrélation des variables

La Figure 12 montre une corrélation nette entre les incidents (Retards de paiement, Appels support) et le départ des clients. Par ailleurs, les variables explicatives sont peu corrélées entre elles. Cette absence de redondance confirme la qualité du dataset : chaque variable apporte une information unique et utile pour l'entraînement du modèle.

III. PRÉTRAITEMENT DES DONNÉES (PREPROCESSING)

L'analyse exploratoire a permis d'identifier les variables pertinentes et de comprendre la structure de nos données. Cependant, les algorithmes d'apprentissage automatique (*Machine Learning*) ne peuvent pas traiter directement des données brutes (texte, échelles différentes, etc.). Cette phase de prétraitement est donc cruciale : elle vise à transformer notre dataset en une matrice numérique propre et normalisée.

3.1. Nettoyage et standardisation :

Nous avons débuté par une hygiène de base du jeu de données. Afin d'éviter les erreurs de manipulation dans le code, nous avons harmonisé les noms des colonnes

(minuscules, remplacement des espaces par des tirets bas). Parallèlement, nous avons supprimé l'identifiant unique CustomerID, inutile pour la prédiction, ainsi que les lignes contenant des valeurs manquantes pour garantir l'intégrité du calcul.

```
# Suppression de la colonne 'CustomerID'
df_clean.drop(columns=['CustomerID'], inplace=True)

# Standardisation des noms de colonnes
df_clean.columns = df_clean.columns.str.lower().str.replace(' ', '_')

print("Nouveaux noms de colonnes : ", list(df_clean.columns))
✓ 0.0s
```

Python

Nouveaux noms de colonnes : ['age', 'gender', 'tenure', 'usage_frequency', 'support_ca

Figure 13 : Standardisation des colonnes et nettoyage initial

3.2. Encodage des variables :

Pour convertir les données textuelles en valeurs numériques, nous avons adopté une double stratégie. D'une part, l'Encodage Ordinal a été appliqué aux variables hiérarchiques (Contract Length, Subscription Type) pour préserver la notion de progression (ex: Premium > Basic). D'autre part, le One-Hot Encoding (à l'aide de get_dummies) a été utilisé pour le Gender (Homme/Femme) afin d'éviter d'introduire une hiérarchie artificielle entre les sexes.

```
# On définit manuellement l'ordre pour que le modèle comprenne la progression
ordinal_cols = ['subscription_type', 'contract_length']
ordinal_order = [
    ['Basic', 'Standard', 'Premium'],           # 0=Basic, 1=Standard, 2=Premium
    ['Monthly', 'Quarterly', 'Annual']          # 0=Mensuel, 1=Trimestriel, 2=Annuel
]

# On applique l'encodage
ord_enc = OrdinalEncoder(categories=ordinal_order)
# On force la conversion en entiers (int) pour être propre
df_clean[ordinal_cols] = ord_enc.fit_transform(df_clean[ordinal_cols]).astype(int)
✓ 0.2s
```

Python

```
df_encoded = pd.get_dummies(df_clean, columns=['gender'], drop_first=True, dtype=int)

# On re-standardise tout en minuscule pour être sûr d'avoir 'gender_male'
df_encoded.columns = df_encoded.columns.str.lower()
```

Python

Figure 14 : Stratégie de transformation des variables textuelles

3.3. Conversion et Matrice des Données :

À l'issue de ces transformations, nous obtenons un jeu de données intégralement numérique. Cette étape marque le passage d'une vue "métier" à une structure mathématique exploitable. Comme l'illustre la figure ci-dessous, le tableau se présente

désormais sous la forme d'une matrice où chaque client est défini exclusivement par des valeurs quantitatives, prêtes à être ingérées par les algorithmes.

	age	tenure	usage_frequency	support_calls	payment_delay	subscription_type	contract_length	total_spend	last_interaction	churn	gender_male
0	30.0	39.0	14.0	5.0	18.0	1	2	932.0	17.0	1.0	0
1	65.0	49.0	1.0	10.0	8.0	0	0	557.0	6.0	1.0	0
2	55.0	14.0	4.0	6.0	18.0	0	1	185.0	3.0	1.0	0
3	58.0	38.0	21.0	7.0	7.0	1	0	396.0	29.0	1.0	1
4	23.0	32.0	20.0	5.0	8.0	0	0	617.0	20.0	1.0	1

Figure 15 : Aperçu de la matrice de données avant normalisation

3.4. Séparation, Normalisation et Sauvegarde :

Pour finaliser la préparation, nous avons divisé le dataset en deux ensembles : 80% pour l'entraînement (*Train*) et 20% pour l'évaluation (*Test*), en utilisant une stratification pour conserver l'équilibre des classes. Nous avons ensuite appliqué une normalisation (*StandardScaler*) pour harmoniser les échelles de valeurs (âge vs dépenses). Enfin, pour garantir la reproductibilité, le scaler a été sauvegardé (scaler.pkl) et les jeux de données finaux exportés en fichiers CSV distincts.

```
X = df_encoded.drop('churn', axis=1)
y = df_encoded['churn']

# split stratifié
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)

✓ 0.3s Python

X_train shape: (404164, 10)
X_test shape: (101042, 10)
y_train shape: (404164,)
y_test shape: (101042,)

# Standardisation
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

✓ 0.2s Python

# Conversion en DataFrame final
X_train_final = pd.DataFrame(X_train_scaled, columns=X.columns)
X_test_final = pd.DataFrame(X_test_scaled, columns=X.columns)

✓ 0.0s Python
```

```
# Enregistrement des 4 fichiers
X_train_final.to_csv('../data/processed/X_train.csv', index=False)
X_test_final.to_csv('../data/processed/X_test.csv', index=False)
y_train.to_csv('../data/processed/y_train.csv', index=False)
y_test.to_csv('../data/processed/y_test.csv', index=False)

✓ 5.9s Python
```

```
# Sauvegarde du scaler
joblib.dump(scaler, '../models/scaler.pkl')

['../models/scaler.pkl'] Python
```

Figure 16 : Processus de séparation, normalisation et sauvegarde des données

IV. MODÉLISATION ET ÉVALUATION DES PERFORMANCES

Notre démarche de modélisation vise à identifier l'algorithme capable de prédire le départ des clients avec la plus grande fiabilité. Nous avons adopté une approche progressive, allant des modèles les plus naïfs aux méthodes ensemblistes complexes, afin de valider la pertinence de chaque gain de performance.

- **Établissement de la Baseline (Dummy Classifier)**

Avant de tester des algorithmes avancés, il est essentiel de définir un seuil de performance minimal. Nous avons pour cela utilisé un "Dummy Classifier" avec une stratégie de fréquence majoritaire. Ce modèle prédit systématiquement qu'un client va rester, sans analyser ses caractéristiques. Bien que ses résultats soient logiquement médiocres, ce score de référence nous permet de juger l'utilité réelle des modèles suivants. Tout algorithme retenu devra impérativement surperformer cette base.

```
from sklearn.dummy import DummyClassifier

# Stratégie "most_frequent" : on prédit toujours la classe majoritaire (0: Reste)
dummy = DummyClassifier(strategy="most_frequent")
dummy.fit(x_train, y_train)
baseline_score = dummy.score(x_test, y_test)

print(f"Score de base (Baseline) : {baseline_score:.2%}")
print("Si nos modèles font moins que ça, ils sont inutiles.")

✓ 0.0s
```

Score de base (Baseline) : 55.52%
Si nos modèles font moins que ça, ils sont inutiles.

Python

Figure 17 : Performance du modèle de référence (Baseline)

4.1. Approches linéaires et probabilistes (Logistic & Naïve Bayes) :

Dans un second temps, nous avons évalué des modèles simples comme la Régression Logistique et le Naïve Bayes. Si ces algorithmes offrent l'avantage de la rapidité, l'analyse détaillée de leurs matrices de confusion révèle une faiblesse critique : un taux élevé de Faux Négatifs. Concrètement, ces modèles échouent à détecter une grande partie des clients démissionnaires, les classant à tort comme fidèles. Cette incapacité à saisir les ruptures brutales (comme les seuils d'appels ou de retards identifiés précédemment) confirme que la frontière de décision n'est pas linéaire. Ces modèles sont donc insuffisants pour sécuriser le parc client.



Figure 18 : Résultats des modèles linéaires et Probabilistes

4.2. Approches non-linéaires (KNN & Arbre de Décision) :

Afin de dépasser les limites des modèles linéaires, nous avons évalué des approches capables de capturer des structures de données plus complexes : le KNN, basé sur la proximité, et l'Arbre de Décision, fondé sur la segmentation. L'analyse comparative démontre la nette supériorité de l'Arbre de Décision qui, contrairement au KNN souvent imprécis sur les frontières entre classes, parvient à isoler efficacement les profils à risque grâce à sa logique de division hiérarchique. Comme l'illustre la Figure 17, cette méthode réduit significativement les erreurs de classification (Faux Négatifs), s'avérant ainsi bien plus pertinente pour notre objectif de détection des départs.

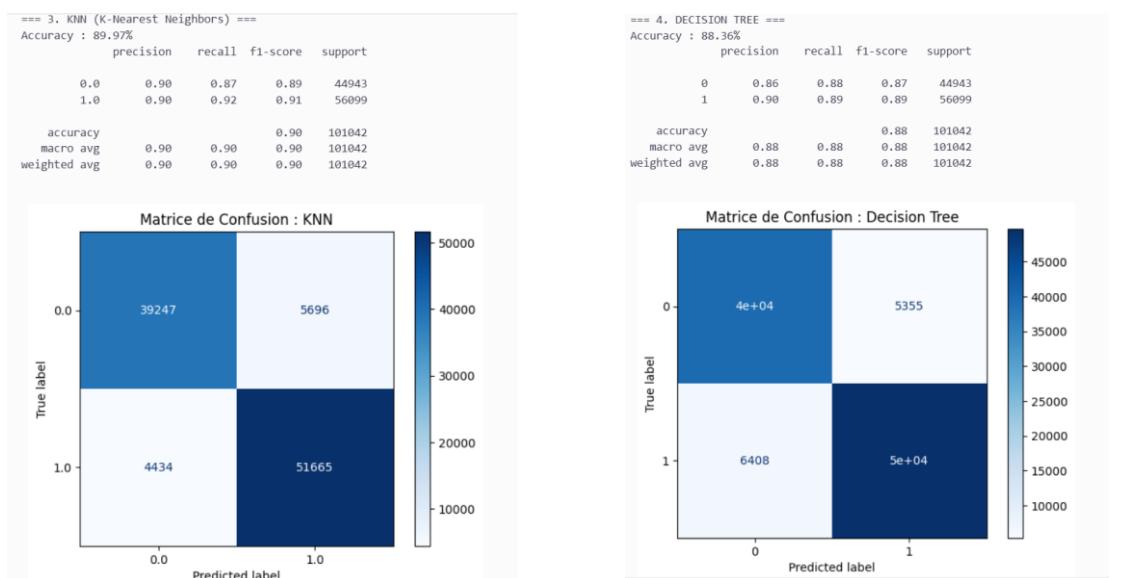


Figure 19 : Résultats des modèles non-linéaires

4.3. Méthodes Ensemblistes (Random Forest et XGBoost) :

Pour franchir un dernier palier de performance et fiabiliser les prédictions, nous avons mobilisé les méthodes ensemblistes. Contrairement aux modèles précédents qui opèrent isolément, le Random Forest et le XGBoost agrègent la puissance de multiples arbres de décision. Cette architecture leur permet de compenser les faiblesses individuelles de chaque arbre par un mécanisme de correction collective.

L'analyse des résultats confirme la supériorité de cette stratégie : on observe une réduction drastique des erreurs résiduelles par rapport à l'arbre de décision simple. Cette robustesse prouve que ces modèles parviennent à filtrer le "bruit" des données pour ne conserver que le signal prédictif réel, garantissant ainsi la stabilité indispensable pour une mise en production.

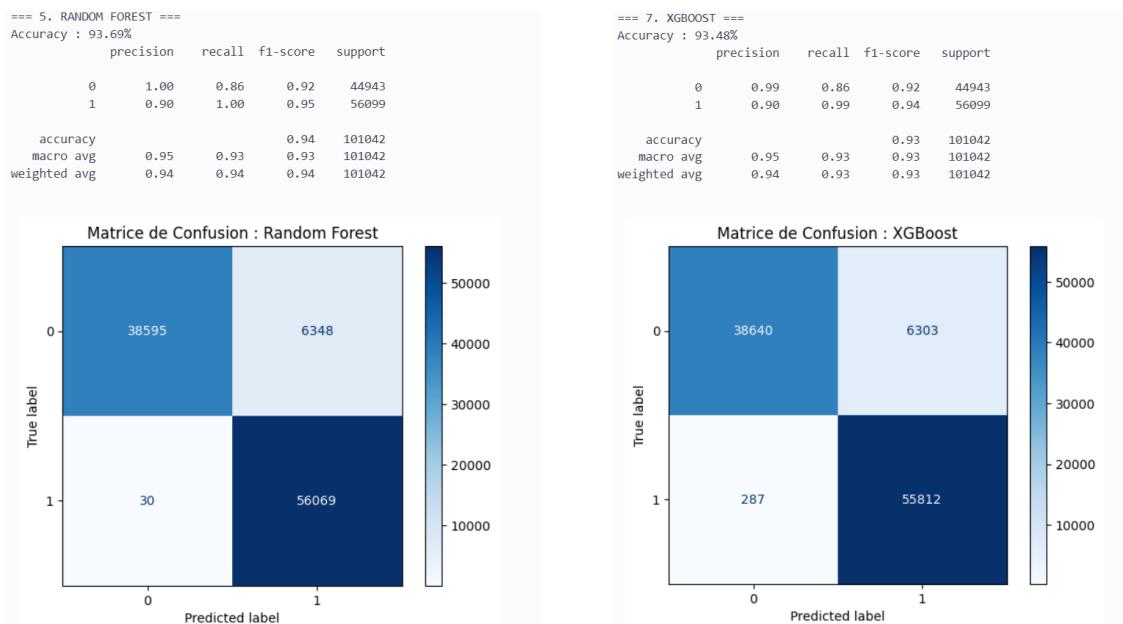


Figure 20 : Résultats des méthodes ensemblistes

4.5. Choix du Modèle et Validation Finale :

Au terme de notre processus de modélisation, une hiérarchie claire s'établit. Le Random Forest et le XGBoost se détachent nettement du lot, reléguant les modèles classiques (Régression Logistique, Arbre de Décision simple) au second plan.

Le duel final s'est joué entre ces deux géants. Si le XGBoost est réputé pour sa précision (minimisation des fausses alertes), notre analyse montre que le Random Forest l'emporte sur le critère décisif du Rappel (Recall). Dans sa configuration actuelle, le XGBoost laisse échapper un léger pourcentage de clients démissionnaires

(Faux Négatifs), là où le Random Forest agit comme une "barrière étanche", capturant la quasi-totalité des profils à risque (Recall > 99%). De plus, le Random Forest démontre une robustesse immédiate ("out-of-the-box"), offrant des performances exceptionnelles sans nécessiter le réglage complexe d'hyperparamètres indispensable au XGBoost.

	Modèle	Accuracy	Precision	Recall	F1-Score
4	Random Forest	0.9369	0.8983	0.9995	0.9462
5	XGBoost	0.9348	0.8985	0.9949	0.9443
2	KNN	0.8997	0.9007	0.9210	0.9107
3	Decision Tree	0.8836	0.9027	0.8858	0.8942
1	Naive Bayes	0.8728	0.8978	0.8700	0.8837
0	Logistic Regression	0.8338	0.8568	0.8413	0.8490

Tableau 2 : Classement final et sélection du modèle

Privilégiant la sécurité opérationnelle (ne rater aucun départ potentiel), nous validons le Random Forest comme modèle final. C'est cet algorithme qui sera sauvegardé et utilisé pour la mise en production.

• Analyse de la Performance Globale via la Courbe ROC

Pour vérifier la fiabilité de nos modèles au-delà des simples pourcentages de réussite, nous avons utilisé l'analyse de la courbe ROC. Cet outil graphique permet d'évaluer la capacité de chaque algorithme à bien distinguer les clients fidèles des clients sur le départ. La règle de lecture est simple : plus la courbe se colle au coin supérieur gauche du graphique, plus le modèle est performant. Nous utilisons l'indicateur "AUC" (Aire Sous la Courbe) pour donner une note globale allant de **0,5** (le hasard) à **1** (la perfection).

L'observation de la *Figure 21* confirme sans ambiguïté la supériorité des méthodes avancées. Le Random Forest et le XGBoost se détachent nettement du lot avec une note identique et excellente de **0,95**. Visuellement, leurs courbes grimpent presque verticalement dès le début, ce qui signifie qu'ils parviennent à détecter la quasi-totalité des départs sans déclencher de fausses alertes inutiles. À l'inverse, les modèles plus simples comme la Régression Logistique ou l'Arbre de Décision (courbe verte) peinent davantage. Leurs courbes montent plus lentement ou par "paliers", ce qui se traduit par des scores plus faibles (entre **0,88** et **0,93**) et une précision moins fine.

En conclusion, cette analyse graphique valide que le Random Forest et le XGBoost sont les deux seuls modèles offrant une stabilité suffisante pour une utilisation réelle. Puisqu'ils font jeu égal sur ce critère de stabilité, le choix final se confirme en faveur du Random Forest. Comme vu précédemment, c'est lui qui offre la meilleure garantie de ne rater aucun client à risque (meilleur score de Rappel), ce qui en fait l'outil le plus sûr pour la stratégie de l'entreprise.

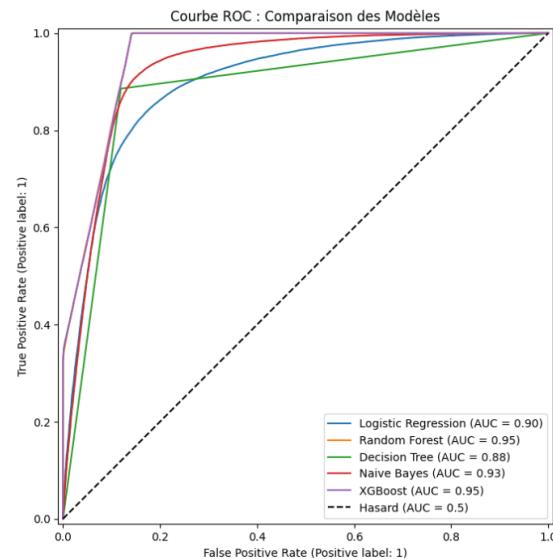


Figure 21 : Courbes ROC comparatives

4.6. Interprétabilité et Feature Importance :

L'analyse des variables prédictives permet de comprendre la logique décisionnelle du modèle Random Forest afin de ne pas l'utiliser comme une "boîte noire". L'examen du graphique d'importance *Figure 20* révèle sans équivoque que la fréquence des appels au service client (support_calls) constitue le facteur déterminant majeur, pesant pour plus de 26% dans la décision finale. Cette prédominance indique qu'une insatisfaction répétée, manifestée par de multiples contacts avec le support technique, est le signal le plus fiable d'un départ imminent.

En second plan, les indicateurs financiers tels que le montant total des dépenses et les retards de paiement jouent un rôle significatif. Ils agissent comme des signaux d'alerte comportementaux, traduisant souvent un désengagement progressif ou des difficultés de paiement de la part du client. À l'opposé, les données démographiques comme le genre ou le type d'abonnement affichent une influence négligeable sur la

prédition, prouvant que le départ est davantage lié à l'expérience vécue par le client (problèmes, interaction) qu'à son profil sociologique.

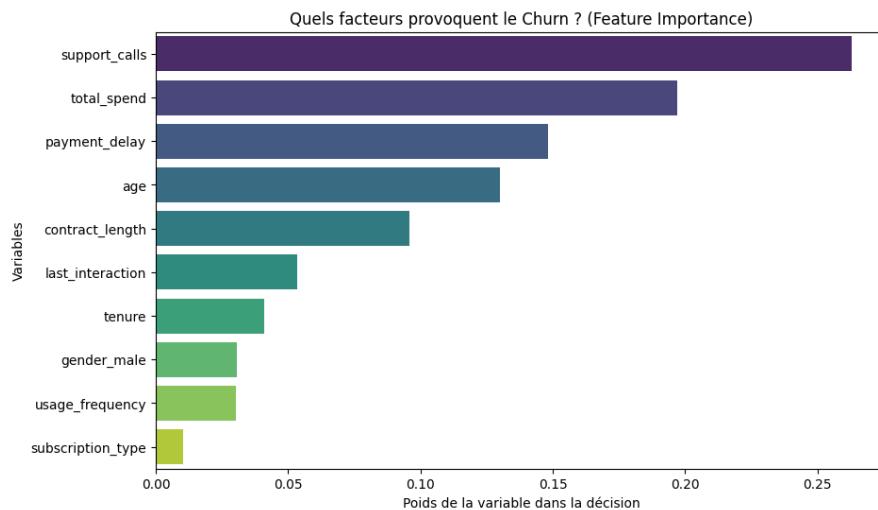


Figure 22 : Hiérarchie des variables prédictives

V. ARCHITECTURE TECHNIQUE ET INDUSTRIALISATION

La validation mathématique du modèle ne constitue qu'une étape du projet. Pour garantir que notre solution soit déployable et reproductible dans un environnement professionnel, nous avons structuré le code selon les standards du MLOps (Machine Learning Operations). Nous sommes passés d'une phase d'expérimentation (Notebooks) à une architecture logicielle modulaire.

Comme l'illustre la figure ci-dessous, l'arborescence du projet sépare clairement les responsabilités :

- **Data** : Une distinction stricte entre les données brutes (raw) et traitées (processed) évite toute contamination.
- **Notebooks** : Cet espace est réservé à l'exploration et aux tests préliminaires.
- **Models** : C'est le cœur du système. Une fois le Random Forest validé, il a été sérialisé (exporté) sous format binaire (.pkl) dans ce dossier, prêt à être appelé par l'application.
- **App** : Ce dossier contient le code source de l'interface utilisateur qui charge le modèle stocké.

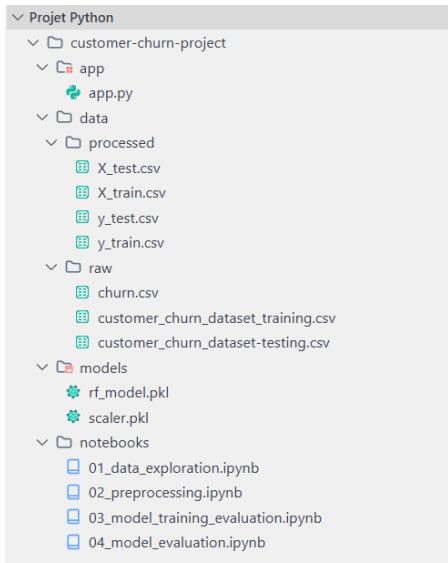


Figure 23 : Architecture structurée du projet pour la mise en production

VI. DÉPLOIEMENT DE L'INTERFACE UTILISATEUR (STREAMLIT)

Afin de rendre ces résultats accessibles aux équipes métier (marketing, service client) sans compétences techniques, nous avons développé une interface web interactive avec le framework Streamlit. Cette application transforme notre modèle statistique en un outil décisionnel concret et utilisable en temps réel.

Le fonctionnement est conçu pour être intuitif : l'utilisateur saisit les caractéristiques du client (âge, dépenses, nombre d'appels au support) via le panneau de contrôle latéral. En arrière-plan, l'application récupère ces données, les normalise et interroge le fichier rf_model.pkl sauvegardé précédemment.



Figure 24 : Interface de saisie des paramètres client

Le résultat s'affiche instantanément sous forme visuelle. L'application ne donne pas seulement une classe (0 ou 1), mais calcule la **probabilité exacte de départ**. Si ce score dépasse le seuil critique, une alerte visuelle (rouge) informe l'utilisateur qu'une action de rétention est nécessaire. Ce prototype fonctionnel valide la capacité du projet à s'intégrer dans les processus quotidiens de l'entreprise.



Figure 25 : Visualisation du risque de Churn en temps réel

Pour aller au-delà de la simple alerte et permettre une action ciblée, l'application fournit également une analyse explicative des causes. Comme illustré ci-dessous, le module identifie et hiérarchise les variables qui ont le plus pesé dans la décision du modèle pour ce client spécifique.



Figure 26 : Analyse des facteurs influençant le risque

Cette visualisation met en évidence les points critiques (en rouge) et les avertissements (en orange). Dans l'exemple ci-dessus, le modèle indique qu'une "Faible Ancienneté" est le facteur dominant du risque. Cette information est cruciale : elle permet au conseiller de ne pas agir à l'aveugle, mais de proposer une offre personnalisée (ex: une prolongation de période d'essai) répondant précisément au problème détecté.

CONCLUSION GÉNÉRALE

Ce projet s'est inscrit dans une démarche visant à anticiper le phénomène d'attrition client (Churn) à travers l'exploitation de données massives et l'apprentissage automatique. L'objectif principal était de fournir à l'entreprise un outil prédictif fiable capable d'identifier les profils à risque afin d'optimiser les stratégies de rétention et de préserver le chiffre d'affaires.

La méthodologie adoptée a suivi les étapes rigoureuses du processus de Data Science, débutant par une phase critique de nettoyage et d'exploration. La phase de modélisation a ensuite confronté six algorithmes distincts : Régression Logistique, Naive Bayes, KNN, Arbre de Décision, Random Forest et XGBoost. Cette étude comparative a démontré les limites des modèles linéaires classiques (Régression Logistique) face à la non-linéarité des comportements clients, consacrant la supériorité des méthodes ensemblistes.

Le modèle Random Forest a été retenu pour son équilibre optimal, affichant une précision de **93,69%** et une AUC de **0.95**. Au-delà de la performance brute, l'industrialisation du modèle via l'application Streamlit a apporté une valeur métier concrète. L'analyse de l'importance des variables, visualisée dans l'interface, a permis de cibler les appels au support technique et les dépenses comme indicateurs d'alerte prioritaires.

Néanmoins, ce travail ouvre la voie à plusieurs perspectives d'amélioration. Sur le plan technique, l'intégration de données non structurées, telles que les emails ou transcriptions d'appels, permettrait d'affiner la compréhension des motifs d'insatisfaction grâce au traitement du langage naturel (NLP). Sur le plan opérationnel, l'évolution du prototype actuel vers une API temps réel hébergée sur le Cloud constituerait la suite logique, permettant d'automatiser les actions marketing dès l'apparition de signaux faibles.

WEBOGRAPHIE

[Customer Churn Dataset](#)

[The Value of Keeping the Right Customers](#)

[scikit-learn: machine learning in Python](#)

[Stack Overflow](#)

[ChatGPT](#)