ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR

Membre de HONORIS UNITED UNIVERSITIES

# PROJET DE FIN D'ANNEE

**5ème Année en Ingénierie Informatique et Réseaux**

# Prédiction du churn client et recommandations d'actions de rétention en e-commerce


**Réalisé par :** Abdelkader Arras

Yassine El Maghraoui

**Encadrant Pédagogique :** M. Kamal Najem


**Année universitaire : 2025/2026**

# Table des matières

# Table des figures

# Liste des tableaux

# Introduction générale

L'introduction générale comporte, généralement, deux parties.

Dans la première partie, nous présenterons notre sujet à travers des informations précises et poserons par la suite la problématique à résoudre avec clarté et sans évocation de résultats.

Dans la deuxième partie, nous présenterons le plan de notre rapport en évoquant, brièvement, le contenu de chaque chapitre.

*[Le texte de votre introduction générale doit être rédigé ici en respectant la mise en forme : justifié, première ligne avec retrait de 0.8 cm, interligne 1.5 ligne, Times New Roman 12pts.]*

# Chapitre 1

# Contexte Général du Projet

## 1.1 Contexte général

Avec la digitalisation croissante du commerce, le secteur du e-commerce connaît une expansion rapide et une concurrence de plus en plus intense. Les consommateurs disposent aujourd'hui d'un large choix de plateformes pour effectuer leurs achats en ligne, ce qui rend leur fidélisation particulièrement complexe. Dans ce contexte, la capacité d'une entreprise à retenir ses clients est devenue un enjeu stratégique majeur.

Le **churn client**, qui désigne le phénomène par lequel un client cesse d'utiliser les services d'une entreprise, représente un risque important pour les plateformes e-commerce. Une augmentation du taux de churn peut entraîner une baisse significative du chiffre d'affaires, une diminution de la valeur client à long terme (**Customer Lifetime Value**) et une augmentation des coûts marketing. En effet, il est généralement admis que le coût d'acquisition d'un nouveau client est largement supérieur au coût de fidélisation d'un client existant.

Face à ces enjeux, les entreprises cherchent de plus en plus à exploiter leurs données clients afin de mieux comprendre les comportements d'achat, d'anticiper les risques de départ et de mettre en place des stratégies de rétention efficaces. L'essor des techniques de **data science** et de **machine learning** permet aujourd'hui d'analyser de grands volumes de données et de détecter des patterns complexes qui seraient difficiles à identifier par des méthodes traditionnelles.

Cependant, la performance prédictive seule ne suffit pas. Les décideurs métiers ont également besoin de modèles interprétables, capables d'expliquer pourquoi un client est considéré comme à risque. Cette **interprétabilité** est essentielle pour instaurer la confiance dans les modèles, faciliter la prise de décision et proposer des actions concrètes et ciblées.

C'est dans ce cadre que s'inscrit ce projet, qui vise à combiner prédiction du churn, interprétabilité des résultats et recommandations métiers, à travers le développement d'un modèle

de machine learning explicable et d'une application interactive destinée à l'aide à la décision.

## 1.2 Problématique

Dans un environnement e-commerce fortement concurrentiel, la perte de clients représente un défi majeur pour les entreprises. Malgré la disponibilité d'un volume important de données liées aux comportements des utilisateurs (historique des commandes, satisfaction, interactions avec la plateforme, réclamations, etc.), ces informations sont souvent sous-exploitées ou utilisées de manière descriptive, sans permettre une anticipation efficace du churn.

L'un des principaux problèmes rencontrés par les entreprises est l'incapacité à identifier à l'avance les clients susceptibles de quitter la plateforme. Les approches traditionnelles, basées sur des règles métier simples ou des analyses statistiques classiques, montrent rapidement leurs limites face à la complexité et à la diversité des comportements clients. Elles ne permettent ni une détection précoce des risques, ni une personnalisation des actions de rétention.

Par ailleurs, même lorsque des modèles de machine learning sont utilisés pour prédire le churn, ceux-ci sont souvent perçus comme des **boîtes noires**. Les équipes métiers peuvent alors difficilement comprendre les raisons derrière une prédiction donnée, ce qui limite l'exploitation opérationnelle des résultats. Or, sans explication claire des facteurs influençant le churn, il devient complexe de définir des actions concrètes et adaptées pour chaque client.

**Comment prédire efficacement le churn des clients e-commerce tout en fournissant des explications interprétables et exploitables, afin d'aider les décideurs à mettre en place des stratégies de rétention ciblées et personnalisées ?**

## 1.3 Analyse de besoins

### 1.3.1 Besoins fonctionnels

Les besoins fonctionnels définissent les fonctionnalités que le système doit offrir afin de répondre aux attentes des utilisateurs.

Tout d'abord, le système doit permettre la prédiction du churn client à partir des données disponibles, en fournissant une **probabilité de départ** pour chaque client. Cette prédiction doit être accompagnée d'une **segmentation du risque** (faible, moyen, élevé) afin de faciliter la prise de décision.

Ensuite, l'application doit offrir une interprétation des résultats du modèle. Pour cela, elle doit afficher les facteurs les plus influents dans la prédiction du churn à l'aide des **valeurs SHAP**, permettant ainsi de comprendre les raisons sous-jacentes au risque identifié.

Le système doit également permettre la **génération de recommandations personnalisées**, basées sur les variables les plus influentes, afin d'aider les équipes métier à définir des actions de rétention adaptées à chaque client.

Par ailleurs, l'utilisateur doit pouvoir consulter les informations d'un client existant en saisissant son identifiant, ou **simuler un nouveau client** en entrant manuellement ses caractéristiques via une interface intuitive.

Enfin, l'application doit proposer des **visualisations graphiques claires** (graphiques SHAP, indicateurs clés, tableaux récapitulatifs) pour faciliter l'analyse et l'exploitation des résultats.

### 1.3.2 Besoins non fonctionnels

Les besoins non fonctionnels définissent les contraintes de qualité auxquelles le système doit répondre.

En termes de **performance**, l'application doit fournir les prédictions et les visualisations dans un délai court afin de garantir une utilisation fluide et interactive. Le temps de réponse doit rester acceptable même lors de l'analyse de clients complexes.

Concernant l'**ergonomie**, l'interface doit être simple, intuitive et accessible à des utilisateurs non techniques. Les résultats doivent être présentés de manière lisible, avec des indicateurs clairs et des visualisations compréhensibles.

La **fiabilité** du système constitue un critère essentiel. Les prédictions doivent être cohérentes, reproductibles et basées sur un modèle entraîné et évalué de manière rigoureuse.

La **maintenabilité** est également un besoin important : la solution doit permettre une mise à jour facile du modèle ou des règles de recommandation, afin de s'adapter à l'évolution des données et des besoins métier.

Enfin, le système doit respecter les exigences de **sécurité et de confidentialité** des données, en assurant la protection des informations clients et en limitant l'accès aux données sensibles aux utilisateurs autorisés uniquement.

## 1.4 Objectifs

### 1.4.1 Objectifs du projet

L'objectif principal de ce projet est de concevoir et de mettre en œuvre une solution intelligente permettant de prédire le churn des clients e-commerce, tout en fournissant des explications claires et interprétables sur les décisions du modèle afin de faciliter la prise de décision métier.

Cet objectif général se décline en plusieurs objectifs spécifiques :

### 1.4.2 Objectifs techniques

- Analyser et préparer un jeu de données e-commerce contenant des informations comportementales, transactionnelles et démographiques des clients.
- Mettre en place un processus de prétraitement des données incluant le nettoyage, l'encodage des variables catégorielles et la normalisation des variables numériques.
- Développer un modèle de machine learning performant pour la prédiction du churn, basé sur des algorithmes adaptés aux données tabulaires (notamment **XGBoost**).
- Évaluer les performances du modèle à l'aide de métriques appropriées (**accuracy, precision, recall, F1-score, AUC, etc.**) tout en évitant les problèmes de surapprentissage et de **data leakage**.
- Intégrer des méthodes d'explicabilité du modèle, en particulier **SHAP (SHapley Additive exPlanations)**, afin d'identifier les variables les plus influentes à la fois au niveau global et individuel.

### 1.4.3 Objectifs métiers

- Segmenter les clients selon leur niveau de risque de churn (**faible, moyen, élevé**) à partir des probabilités prédites.
- Identifier, pour chaque client, les **facteurs clés** contribuant à l'augmentation ou à la diminution du risque de churn.
- Associer ces facteurs à des **actions de rétention personnalisées**, adaptées au contexte et au profil du client.
- Faciliter la compréhension et l'appropriation des résultats par les équipes non techniques (marketing, CRM, service client).

### 1.4.4 Objectifs fonctionnels

Concevoir une interface interactive permettant :

— d'estimer la probabilité de churn d'un client existant ou d'un client hypothétique,

— de visualiser les principaux **drivers du churn**,

— d'afficher des recommandations concrètes pour réduire le risque de départ.

Présenter les résultats sous forme de visualisations claires (graphiques, tableaux, indicateurs clés) afin de favoriser une analyse rapide et efficace.

Proposer une solution **modulaire et évolutive** pouvant être intégrée ultérieurement dans un système décisionnel ou un outil CRM.

### 1.4.5 Objectif pédagogique

— Mettre en pratique les concepts étudiés en data science, machine learning et interprétabilité des modèles.

— Démontrer la capacité à mener un projet de bout en bout, depuis la compréhension du besoin métier jusqu'au déploiement d'une solution exploitable.

# Chapitre 2

# Exploration et prétraitement des données

## 2.1 Présentation du Jeu de Données

Le présent projet s'appuie sur un jeu de données public relatif à l'analyse et à la prédiction du churn client dans le secteur du e-commerce, disponible sur la plateforme Kaggle (référence : *Ecommerce Customer Churn Analysis and Prediction*).

Ce jeu de données, qui se présente sous une forme tabulaire, compile des informations à la fois transactionnelles, démographiques et comportementales pour un ensemble de clients.

### 2.1.1 Structure et Variable Cible

Le dataset comprend **15 variables d'entrée** (ou *features*) et **une variable cible** pour un total de 5630 observations, représentant chacune un client unique.

La variable cible de notre étude est :

— **Churn :** Il s'agit d'une variable binaire (drapeau de désabonnement, ou *Churn Flag*) indiquant si le client est considéré comme ayant quitté la plateforme (valeur 1) ou non (valeur 0). Sa définition opérationnelle est cruciale, et nous analyserons sa distribution dans la section suivante.

### 2.1.2 Description des Variables

| **Catégorie** | **Variable** | **Description** | **Type de Donnée Estimé** |
|---|---|---|---|
| Identifiant | CustomerID | Identifiant unique du client. | Qualitatif (ID) |

| | | | |
|---|---|---|---|
| Cible | Churn | Indicateur de départ du client (Flag). | Qualitatif (Binaire) |
| Démographie | Tenure | Ancienneté du client au sein de l'organisation. | Quantitatif Discret |
| Démographie | CityTier | Niveau de la ville du client (Tier 1, 2, 3...). | Qualitatif Ordinal |
| Démographie | Gender | Genre du client. | Qualitatif Nominal |
| Démographie | MaritalStatus | Statut marital du client. | Qualitatif Nominal |
| Démographie | NumberOfAddress | Nombre total d'adresses ajoutées par le client. | Quantitatif Discret |
| Comportement | PreferredLoginDevice | Appareil de connexion préféré. | Qualitatif Nominal |
| Comportement | HourSpendOnApp | Nombre d'heures passées sur l'application mobile ou le site. | Quantitatif Continu |
| Comportement | NumberOfDeviceRegistered | Nombre total d'appareils enregistrés. | Quantitatif Discret |
| Comportement | PreferredOrderCat | Catégorie de commande préférée le mois dernier. | Qualitatif Nominal |
| Comportement | SatisfactionScore | Score de satisfaction du client. | Quantitatif Discret |
| Comportement | Complain | Indicateur si une plainte a été soulevée le mois dernier. | Qualitatif Binaire |
| Transactionnel | WarehouseToHome | Distance entre l'entrepôt et le domicile du client (en km). | Quantitatif Continu |
| Transactionnel | PreferredPaymentMode | Mode de paiement préféré. | Qualitatif Nominal |

| | | | |
|---|---|---|---|
| Transactionnel | OrderAmountHikeFromLastYear | Pourcentage d'augmentation des commandes par rapport à l'année dernière. | Quantitatif Continu |
| Transactionnel | CouponUsed | Nombre total de coupons utilisés le mois dernier. | Quantitatif Discret |
| Transactionnel | OrderCount | Nombre total de commandes passées le mois dernier. | Quantitatif Discret |
| Transactionnel | DaySinceLastOrder | Nombre de jours écoulés depuis la dernière commande. | Quantitatif Discret |
| Transactionnel | CashbackAmount | Montant moyen de cashback le mois dernier. | Quantitatif Continu |

TABLE 2.1: Description des variables du jeu de données

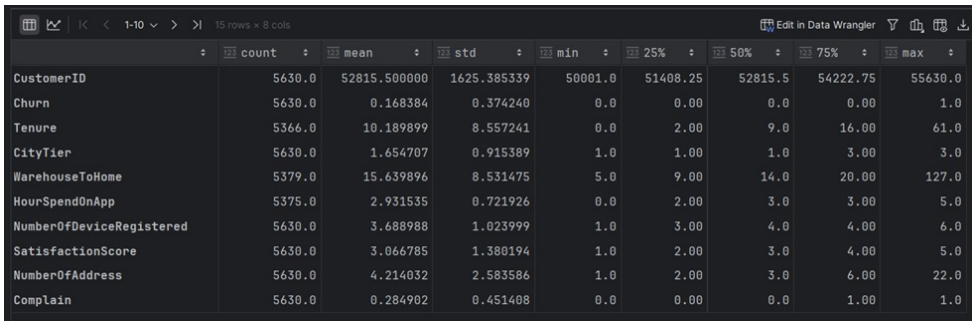
### 2.1.3 Structure Initiale du Jeu de Données et Qualité

```
#   Column                       Non-Null Count  Dtype
--  ------                       --------------  -----
0   CustomerID                   5630 non-null   int64
1   Churn                        5630 non-null   int64
2   Tenure                       5366 non-null   float64
3   PreferredLoginDevice         5630 non-null   object
4   CityTier                     5630 non-null   int64
5   WarehouseToHome              5379 non-null   float64
6   PreferredPaymentMode         5630 non-null   object
7   Gender                       5630 non-null   object
8   HourSpendOnApp               5375 non-null   float64
9   NumberOfDeviceRegistered     5630 non-null   int64
10  PreferedOrderCat             5630 non-null   object
11  SatisfactionScore            5630 non-null   int64
12  MaritalStatus                5630 non-null   object
13  NumberOfAddress              5630 non-null   int64
14  Complain                     5630 non-null   int64
15  OrderAmountHikeFromlastYear  5365 non-null   float64
16  CouponUsed                   5374 non-null   float64
17  OrderCount                   5372 non-null   float64
18  DaySinceLastOrder            5323 non-null   float64
19  CashbackAmount               5630 non-null   float64
dtypes: float64(8), int64(7), object(5)
memory usage: 879.8+ KB
```

FIGURE 2.1 – Structure initiale du jeu de données (df.info())

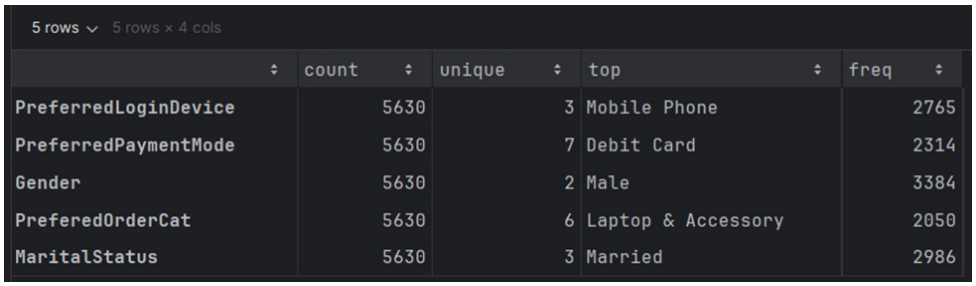
L'analyse de la structure initiale du jeu de données, basée sur le résultat de df.info(), révèle un ensemble de **5630 observations** et **20 variables**. Les types de données se répartissent en **8 variables de type float64**, **7 variables de type int64**, et **5 variables de type object** (catégorielles/textuelles). L'espace mémoire utilisé est d'environ **879.8+ KB**.

### 2.1.4 Statistiques Descriptives et Distribution des Variables



| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| CustomerID | 5630.0 | 52815.500000 | 1625.385339 | 50001.0 | 51408.25 | 52815.5 | 54222.75 | 55630.0 |
| Churn | 5630.0 | 0.168384 | 0.374240 | 0.0 | 0.00 | 0.0 | 0.00 | 1.0 |
| Tenure | 5366.0 | 10.189899 | 8.557241 | 0.0 | 2.00 | 9.0 | 16.00 | 61.0 |
| CityTier | 5630.0 | 1.654707 | 0.915389 | 1.0 | 1.00 | 1.0 | 3.00 | 3.0 |
| WarehouseToHome | 5379.0 | 15.639896 | 8.531475 | 5.0 | 9.00 | 14.0 | 20.00 | 127.0 |
| HourSpendOnApp | 5375.0 | 2.931535 | 0.721926 | 0.0 | 2.00 | 3.0 | 3.00 | 5.0 |
| NumberOfDeviceRegistered | 5630.0 | 3.688988 | 1.023999 | 1.0 | 3.00 | 4.0 | 4.00 | 6.0 |
| SatisfactionScore | 5630.0 | 3.066785 | 1.380194 | 1.0 | 2.00 | 3.0 | 4.00 | 5.0 |
| NumberOfAddress | 5630.0 | 4.214032 | 2.583586 | 1.0 | 2.00 | 3.0 | 6.00 | 22.0 |
| Complain | 5630.0 | 0.284902 | 0.451408 | 0.0 | 0.00 | 0.0 | 1.00 | 1.0 |

FIGURE 2.2 – Statistiques descriptives des variables numériques



| | count | unique | top | freq |
|---|---|---|---|---|
| PreferredLoginDevice | 5630 | 3 | Mobile Phone | 2765 |
| PreferredPaymentMode | 5630 | 7 | Debit Card | 2314 |
| Gender | 5630 | 2 | Male | 3384 |
| PreferedOrderCat | 5630 | 6 | Laptop & Accessory | 2050 |
| MaritalStatus | 5630 | 3 | Married | 2986 |

FIGURE 2.3 – Statistiques descriptives des variables catégorielles

L'analyse exploratoire a permis d'établir les premières statistiques descriptives sur les **5630 clients** du jeu de données, révélant plusieurs caractéristiques clés. Premièrement, la variable cible présente un fort déséquilibre, avec un taux de *churn* moyen de seulement **16.38%** (moyenne de Churn à 0.1638), ce qui indique que la grande majorité (environ 83.62%) des clients est fidèle ; ce déséquilibre nécessitera une attention particulière lors de la modélisation. Concernant l'ancienneté (Tenure), la moyenne est d'environ **10.19 mois**, proche de la médiane à **9 mois**, mais le maximum à 61 mois suggère une large hétérogénéité des profils, incluant des clients de très longue date. Parmi les autres variables numériques, la distance moyenne de l'entrepôt au domicile (WarehouseToHome) est d'environ **15.64 km**, et le score de satisfaction moyen des clients est de **3.06 sur 5**. Du côté des variables catégorielles, les habitudes des clients sont marquées par une connexion majoritaire via **"Mobile Phone"** (2765 occurrences), l'utilisation privilégiée de la **"Debit Card"** pour le paiement (2314 occurrences), et une préférence pour la catégorie de commandes **"Laptop & Accessory"** (2050 occurrences). Ces distributions initiales fournissent un aperçu du profil type du client e-commerce et soulignent la nécessité de

techniques de rééquilibrage pour la variable cible, ouvrant la voie à l'examen approfondi de la relation entre ces variables et le risque de départ.

## 2.2 Identification et Quantification des Valeurs Manquantes

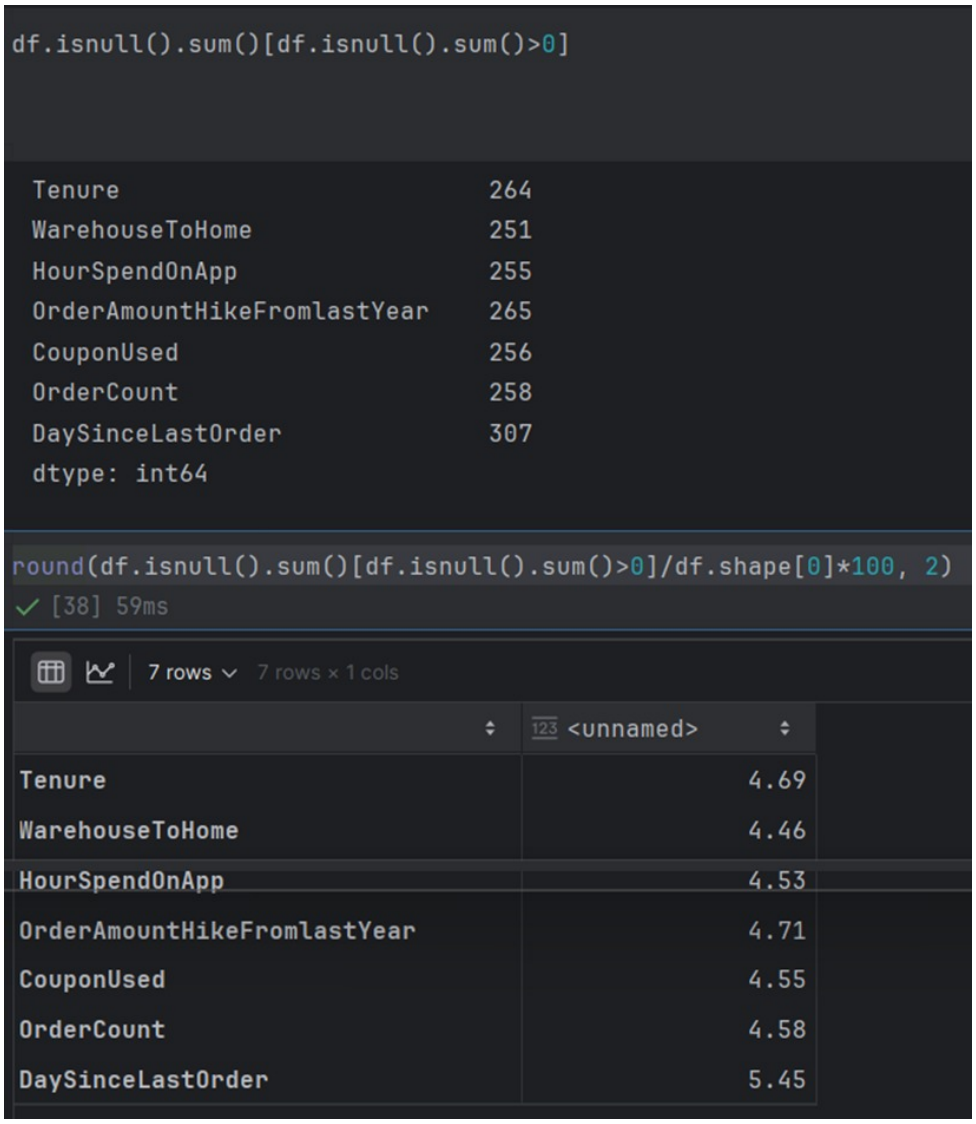



FIGURE 2.4 – Décompte des valeurs manquantes par variable


FIGURE 2.5 – Visualisation des valeurs manquantes

L'analyse de la qualité du jeu de données a révélé une problématique significative de valeurs manquantes (NA).

Le décompte global indique la présence de **1856 valeurs manquantes** dans le jeu de données. Ce total représente **32.97%** des données, un pourcentage élevé qui doit être traité avec soin. De plus, il est établi que le nombre de lignes contenant au moins une valeur manquante est de **1856**, ce qui signifie que toutes les valeurs manquantes sont réparties sur des lignes distinctes (une valeur manquante par ligne affectée).

En examinant la répartition par variable, il est clair que **sept variables numériques** sont concernées. Le nombre de valeurs manquantes varie de **251** pour WarehouseToHome à **307** pour DaySinceLastOrder. En termes de proportion, ces valeurs manquantes représentent un pourcentage relativement homogène, allant de **4.46%** (WarehouseToHome) à **5.45%** (DaySinceLastOrder) du total des 5630 observations.

### 2.2.1 Visualisation de la Distribution des Valeurs Manquantes (Matrice MSNO)

La **Matrice de Rareté des Données (MSNO Matrix)** offre une visualisation rapide et synthétique de la présence et de la répartition des valeurs manquantes dans le jeu de données. Chaque ligne de la matrice représente une observation (client), et chaque colonne une variable. La couleur noire indique la présence d'une donnée, tandis que les lignes blanches représentent les valeurs manquantes.

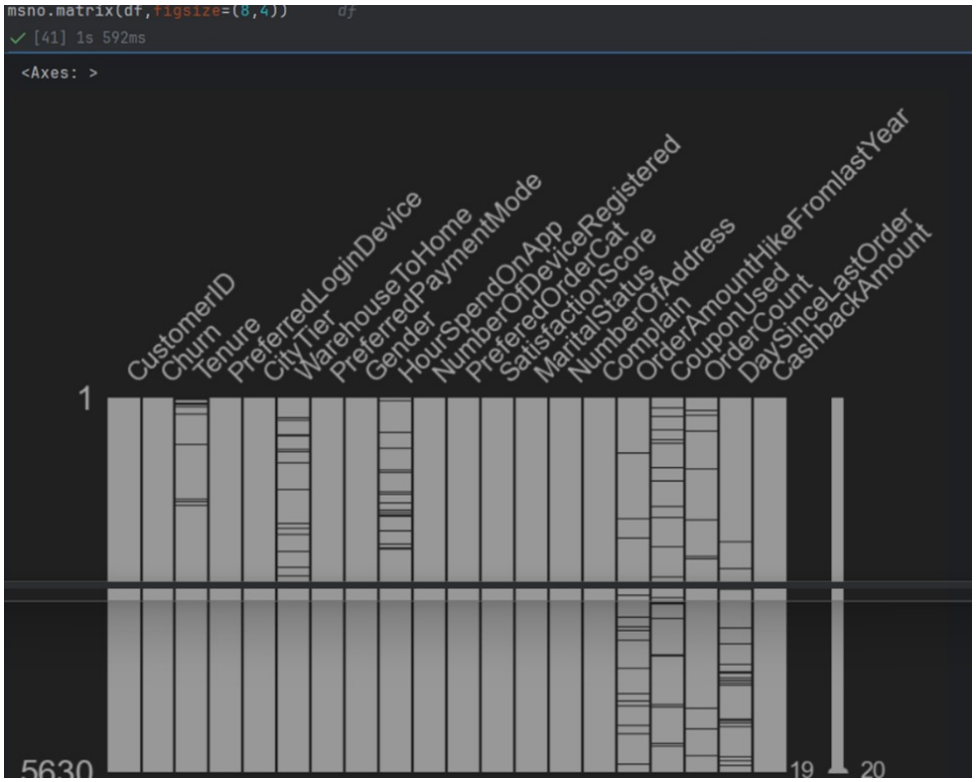



FIGURE 2.6 – Matrice de rareté des données (MSNO Matrix)

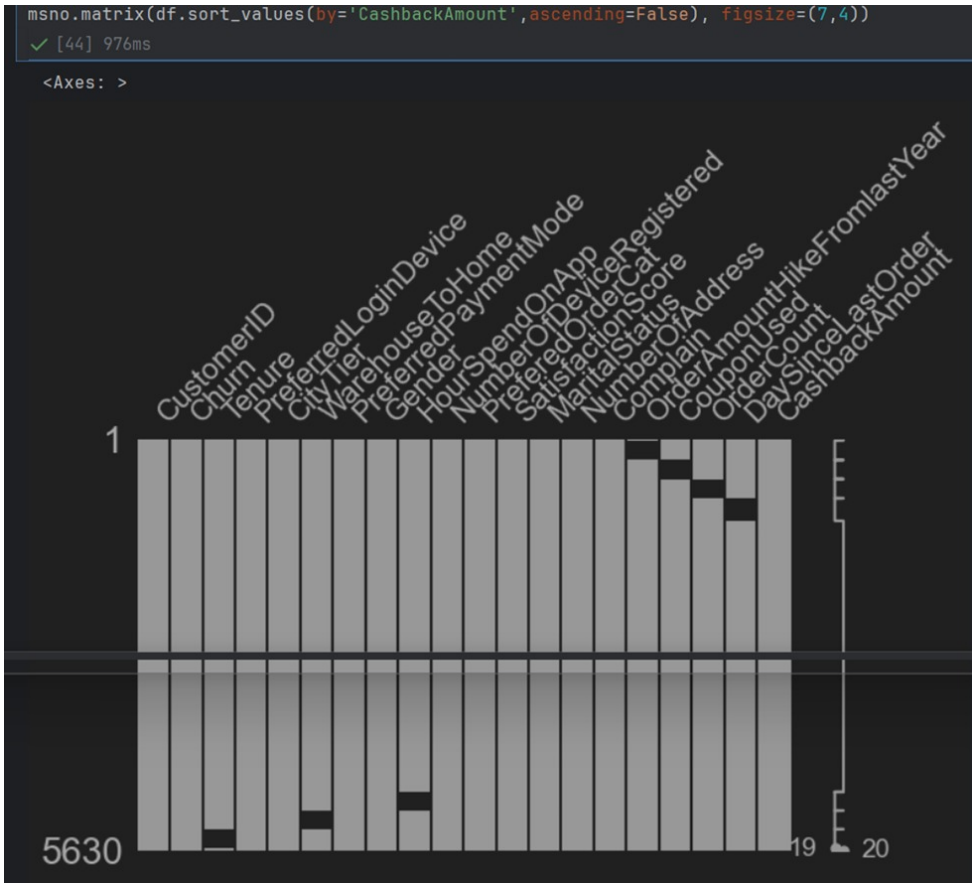



FIGURE 2.7 – Matrice de rareté des données triée par CashbackAmount

Visuellement, la matrice de base ne présente **aucune ligne entièrement vide** et ne montre **aucun grand bloc de blanc** aligné horizontalement, indiquant qu'aucune observation n'est complètement inutilisable et que l'absence de données dans une variable n'est pas fortement corrélée à l'absence de données dans une autre. Les lacunes apparaissent réparties de manière relativement aléatoire sur l'ensemble des 5630 observations, suggérant un mécanisme de type **Missing Completely At Random (MCAR)** ou **Missing At Random (MAR)**. Cette observation est renforcée par l'analyse de la matrice triée par CashbackAmount, où les lignes blanches restent dispersées, prouvant que le modèle de données manquantes n'est pas concentré ou systématiquement lié aux clients ayant des montants de *cashback* spécifiques. Cette absence de systématicité dans le modèle de données manquantes justifie et soutient l'approche d'imputation statistique pour remplacer les valeurs NA, car elle minimise le risque de biais d'échantillonnage majeur.

### 2.2.2 Analyse de la distribution des variables

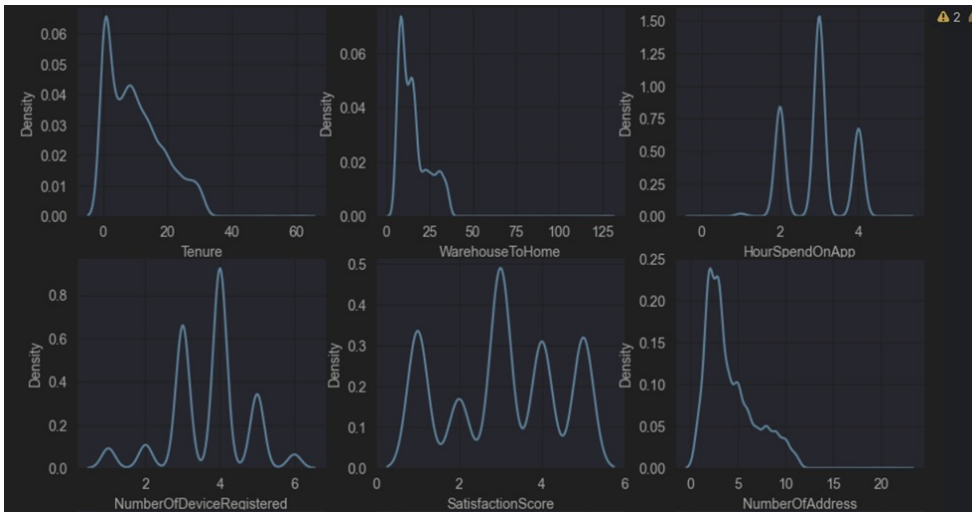



FIGURE 2.8 – Distribution des variables numériques

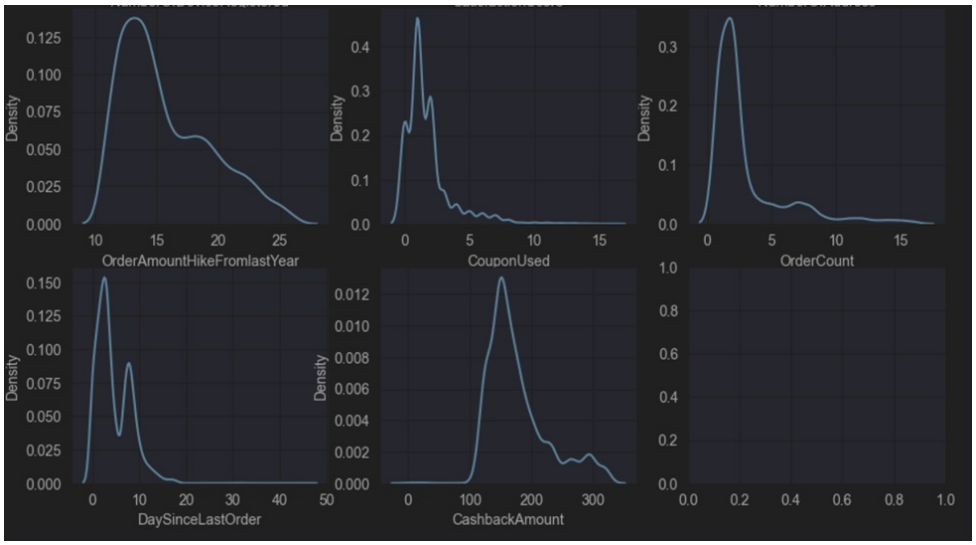



FIGURE 2.9 – Distribution des variables numériques (suite)

Afin de définir une stratégie d'imputation adaptée aux valeurs manquantes identifiées précédemment, une analyse approfondie de la distribution et de l'asymétrie (**skewness**) des variables numériques a été réalisée. Cette étape est essentielle, car le choix entre une imputation par la moyenne ou par la médiane dépend fortement de la forme de la distribution des données.

Les graphiques de densité présentés mettent en évidence que plusieurs variables numériques ne suivent pas une distribution symétrique. En particulier, des variables telles que *OrderCount*, *CouponUsed*, *DaySinceLastOrder*, *NumberOfAddress* ou encore *WarehouseToHome* présentent

une **asymétrie marquée à droite (right-skewed)**, caractérisée par une concentration des valeurs faibles et la présence de valeurs extrêmes élevées. De même, certaines variables montrent des distributions multimodales ou fortement étalées, reflétant une hétérogénéité importante des comportements clients.

Dans ce contexte, l'imputation par la moyenne apparaît inadaptée, car elle est sensible aux valeurs extrêmes et risquerait de biaiser la distribution originale des données en surestimant les valeurs imputées. À l'inverse, la médiane constitue une mesure de tendance centrale plus robuste face aux distributions asymétriques et aux outliers.

### 2.2.3 Conclusion sur la gestion des valeurs manquantes

Au total, le jeu de données contient 1856 valeurs manquantes, réparties sur 1856 lignes distinctes, ce qui signifie que chaque ligne affectée ne comporte qu'une seule valeur manquante. Si l'on supprimait l'ensemble de ces lignes, cela entraînerait la perte de 32,97 % du jeu de données, ce qui représente une perte d'information considérable et inacceptable dans le cadre de ce projet. Par ailleurs, le taux de valeurs manquantes par variable n'excède pas 6 %, et leur répartition apparaît aléatoire à l'échelle du dataset.

L'analyse des motifs de données manquantes a montré que l'absence de certaines valeurs dépend de la variable CashbackAmount, qui est entièrement observée. Cela indique que les données manquantes relèvent d'un mécanisme de type **Missing At Random (MAR)**. Ce type de mécanisme autorise l'utilisation de méthodes d'imputation avancées, fondées sur des modèles, sans introduire de biais majeur.

Étant donné qu'environ un tiers des observations contiennent des valeurs manquantes, la suppression des lignes concernées réduirait fortement la taille de l'échantillon et nuirait à la représentativité des données. Une **stratégie d'imputation** s'impose donc comme une nécessité.

Les méthodes d'imputation simples, telles que l'imputation par la moyenne ou la médiane, présentent plusieurs limites. L'imputation par la moyenne est inadaptée en raison de l'asymétrie marquée de nombreuses variables, ce qui entraînerait une distorsion de leurs distributions. Bien que la médiane soit plus robuste face aux distributions skewed, elle ne prend pas en compte les relations entre les variables et tend à appauvrir la structure multivariée des données.

L'imputation par **K plus proches voisins (KNN)** constitue une alternative plus élaborée, car elle repose sur la similarité entre observations. Toutefois, cette méthode est coûteuse en termes de calcul, sensible à l'échelle des variables et ne garantit pas une préservation optimale

des corrélations entre les features.

Dans ce contexte, la méthode la plus adaptée à ce jeu de données est l'**Iterative Imputer (MICE – Multiple Imputation by Chained Equations)**. Cette approche consiste à modéliser chaque variable contenant des valeurs manquantes en fonction des autres variables du dataset. Elle est particulièrement bien adaptée aux données de type MAR et permet de mieux préserver les corrélations, la variance et la distribution globale des variables.

Ainsi, l'Iterative Imputer a été retenu comme solution finale d'imputation, et sera intégré au pipeline de prétraitement des données avant l'entraînement des modèles de machine learning.

## 2.3 Analyse statistique et relationnelle des données

### 2.3.1 Analyse de la variable cible (Churn vs Non-Churn)

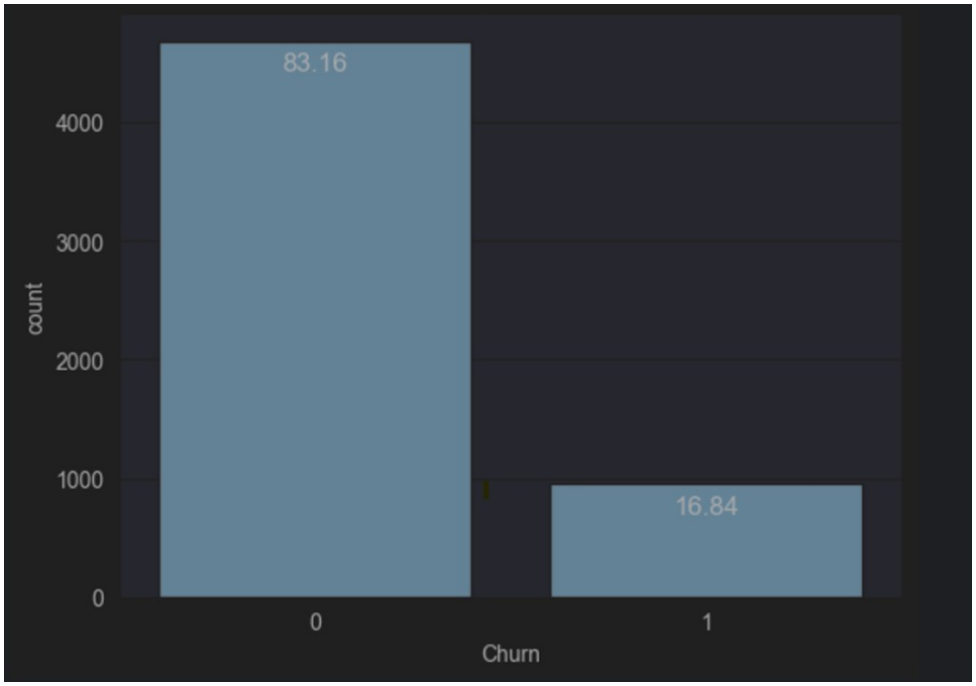



FIGURE 2.10 – Distribution de la variable cible Churn

L'analyse de la variable cible *Churn* constitue une étape essentielle de l'exploration des données, car elle permet de comprendre la répartition des clients ayant quitté la plateforme par rapport à ceux qui sont restés fidèles. La variable *Churn* est de nature binaire, prenant la valeur 1 lorsqu'un client est considéré comme churné et 0 dans le cas contraire.

La figure ci-dessus illustre clairement la distribution de cette variable au sein du jeu de

données. On observe une forte majorité de clients non churnés (Churn = 0), représentant environ 83,16 % de l'ensemble des observations, contre seulement 16,84 % de clients churnés (Churn = 1). Le ratio entre les clients retenus et les clients churnés est donc approximativement de 5 pour 1, ce qui indique un taux de churn relativement faible.

Cette situation est tout à fait réaliste dans un contexte e-commerce, où la majorité des clients restent généralement actifs sur la plateforme, tandis qu'une proportion plus restreinte cesse d'utiliser les services. Toutefois, cette distribution met en évidence un **déséquilibre de classes (class imbalance)** au niveau de la variable cible. Ce déséquilibre peut avoir un impact significatif sur l'apprentissage des modèles de machine learning, ceux-ci ayant tendance à privilégier la classe majoritaire au détriment de la classe minoritaire, ce qui peut conduire à des performances trompeuses si l'on se base uniquement sur des métriques globales comme l'accuracy.

Bien que la gestion explicite du déséquilibre des données (data balancing) ne soit pas strictement indispensable à ce stade de l'analyse, cette observation souligne l'importance de choisir des métriques d'évaluation adaptées et d'adopter une stratégie de modélisation rigoureuse lors des phases ultérieures. Cette analyse préliminaire de la variable cible permet ainsi de poser les bases nécessaires pour une modélisation fiable et interprétable du churn client.

### 2.3.2 Analyse des corrélations entre les variables

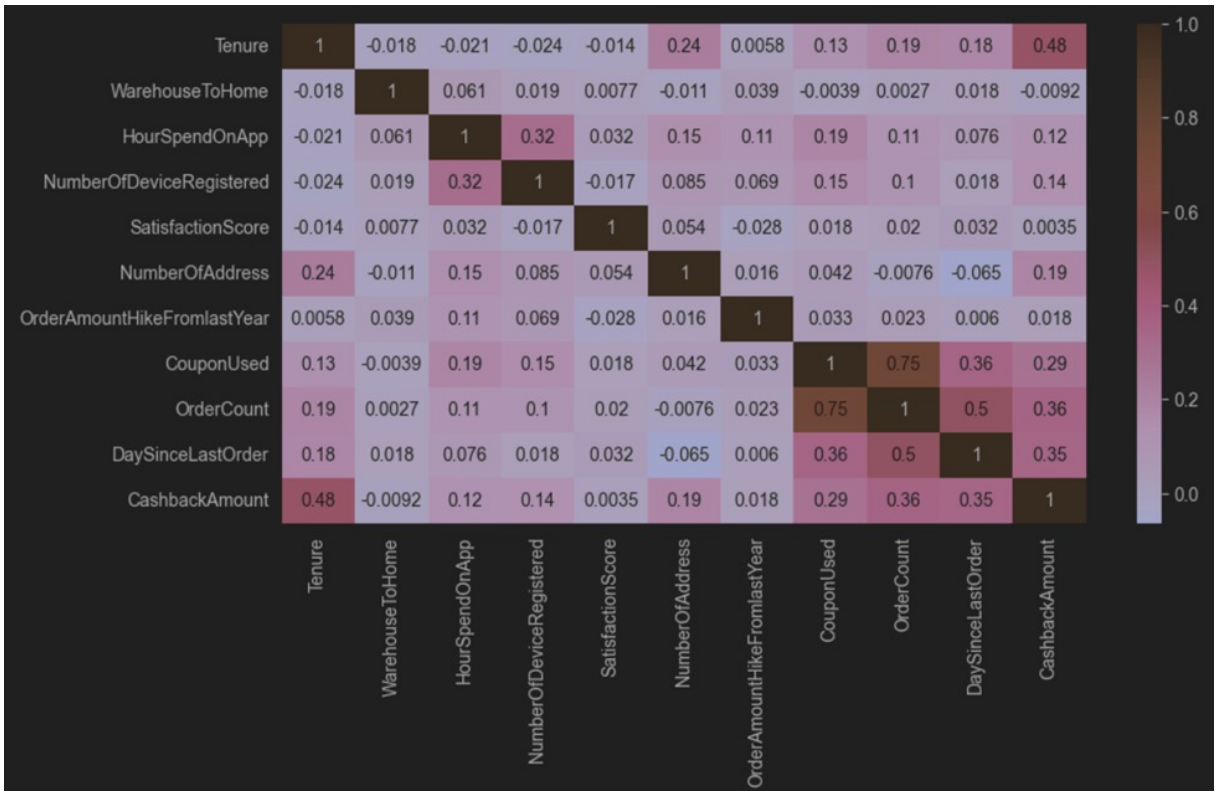



FIGURE 2.11 – Matrice de corrélation des variables numériques

Afin de mieux comprendre les relations existantes entre les variables explicatives du jeu de données, une analyse des corrélations a été réalisée à l'aide du coefficient de corrélation de Pearson. La figure ci-dessus présente une matrice de corrélation sous forme de heatmap, où chaque cellule représente le degré de corrélation linéaire entre deux variables numériques. Les valeurs proches de 1 ou -1 indiquent une forte corrélation positive ou négative, tandis que les valeurs proches de 0 traduisent une faible relation linéaire.

Globalement, la majorité des variables présentent des corrélations faibles à modérées, ce qui est un indicateur positif pour la phase de modélisation, car cela limite les risques de **multicolinéarité** excessive. Certaines relations notables peuvent toutefois être observées. La corrélation la plus élevée concerne les variables *CouponUsed* et *OrderCount*, avec un coefficient d'environ **0,75**, traduisant une relation logique : les clients passant davantage de commandes ont tendance à utiliser plus de coupons. De même, *OrderCount* présente une corrélation modérée avec *DaySinceLastOrder* (environ **0,50**), indiquant que les clients ayant commandé récemment ont généralement un volume de commandes plus élevé.

Par ailleurs, la variable *CashbackAmount* est modérément corrélée avec *Tenure* (**0,48**), ce qui suggère que les clients les plus anciens bénéficient en moyenne de montants de cashback plus importants, probablement en raison de leur fidélité et de leur historique d'achats. Les autres variables, telles que *SatisfactionScore*, *WarehouseToHome* ou *OrderAmountHikeFromLastYear*, montrent des corrélations faibles avec le reste des features, indiquant qu'elles apportent une information complémentaire et relativement indépendante.

Cette analyse confirme que les variables du dataset capturent des dimensions différentes du comportement client, sans redondance excessive. Elle justifie ainsi leur conservation pour les étapes suivantes de modélisation, tout en soulignant l'intérêt de modèles capables de gérer des relations non linéaires et complexes entre les variables, comme les algorithmes basés sur les **arbres de décision**.

### 2.3.3 Détection et Analyse des Valeurs Aberrantes (*Outliers*)

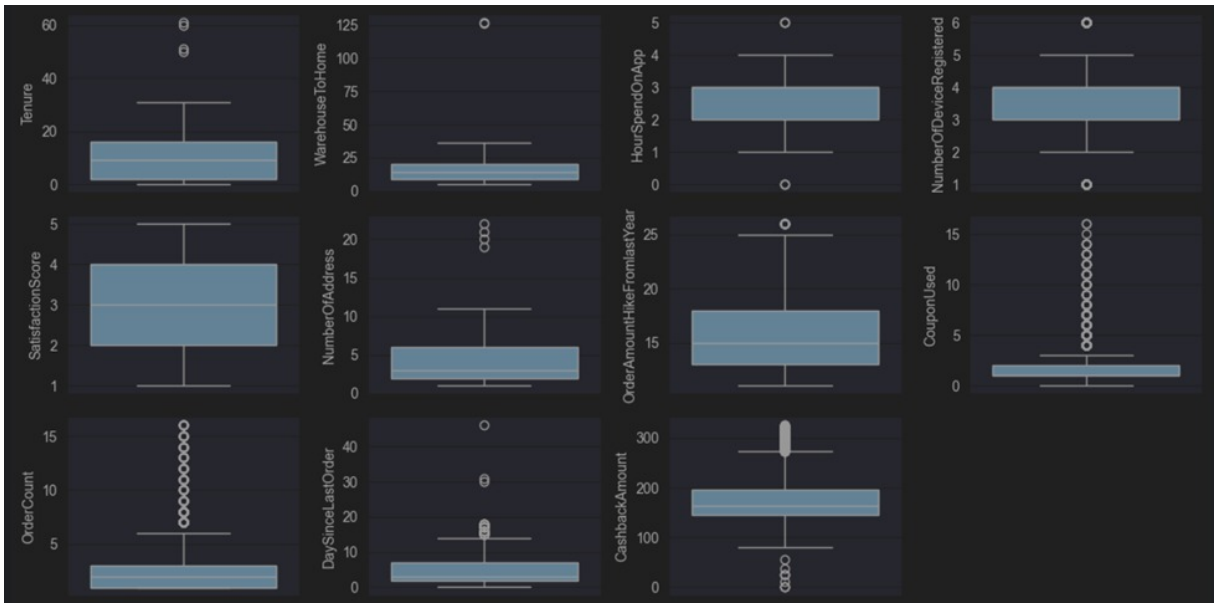



FIGURE 2.12 – Boîtes à moustaches des variables numériques

L'analyse des boîtes à moustaches (*Boxplots*) a été menée pour identifier la présence et l'étendue des valeurs aberrantes (*outliers*) dans les variables numériques, révélant que la majorité des colonnes en contiennent, notamment WarehouseToHome, DaySinceLastOrder, et CashbackAmount, ainsi que CouponUsed et OrderCount. Il est essentiel de considérer que ces valeurs extrêmes pourraient représenter des données réelles et importantes pour la prédiction du *churn* (par exemple, une distance élevée ou une longue inactivité), plutôt que de simples

erreurs. Pour cette raison, une suppression pure et simple de ces observations est écartée afin de préserver l'information pertinente. Étant donné que le projet s'oriente vers l'utilisation de modèles basés sur des arbres comme **XGBoost**, qui sont intrinsèquement robustes face aux *outliers* (car ils sont insensibles à la magnitude des valeurs), l'impact négatif de ces données aberrantes sur la performance du modèle est naturellement minimisé. Il est donc jugé suffisant d'appliquer une simple standardisation des données, sans nécessiter de techniques de gestion des valeurs extrêmes plus complexes comme la **Winsorisation**, avant l'étape de modélisation.

## 2.4 Préparation des données pour la modélisation

### 2.4.1 Séparation des jeux de données (Train / Test)

```
train_df,test_df=train_test_split(df, test_size=0.2,stratify=df['Churn'], random_state=42)

id_train=train_df['CustomerID'].values
id_test=test_df['CustomerID'].values

X_train=train_df.drop(columns=['CustomerID','Churn'])
y_train=train_df['Churn']

X_test=test_df.drop(columns=['CustomerID','Churn'])     X_test      test_df
y_test=test_df['Churn']
```

FIGURE 2.13 – Séparation des données en ensembles d'entraînement et de test

Avant toute étape de prétraitement et de modélisation, le jeu de données a été scindé en deux sous-ensembles distincts : un jeu d'entraînement (*training set*) et un jeu de test (*test set*). Cette séparation a pour objectif d'évaluer de manière fiable la capacité de généralisation des modèles de machine learning sur des données jamais vues lors de l'apprentissage.

Dans ce projet, 80 % des données ont été allouées à l'entraînement et 20 % au test.

La séparation a été réalisée à l'aide d'un **échantillonnage stratifié** sur la variable cible *Churn*, afin de conserver la même proportion de clients churnés et non churnés dans les deux jeux. Ce choix est particulièrement important dans un contexte de déséquilibre des classes, car il garantit une évaluation plus représentative et plus stable des performances du modèle.

Il est essentiel de souligner que l'ensemble des opérations de prétraitement (imputation des valeurs manquantes, encodage des variables catégorielles et standardisation des variables

numériques) est appliqué exclusivement sur les données d'entraînement. Les paramètres appris lors de ces transformations (statistiques d'imputation, moyennes et écarts-types pour la normalisation, catégories observées pour l'encodage) sont ensuite réutilisés tels quels sur le jeu de test. Cette approche permet d'éviter le **data leakage**, c'est-à-dire l'introduction d'informations issues du jeu de test dans le processus d'apprentissage, ce qui conduirait à une surestimation artificielle des performances du modèle.

Enfin, l'identifiant client (*CustomerID*) a été conservé séparément et exclu des variables explicatives, car il ne porte aucune information prédictive pertinente et pourrait biaiser l'apprentissage. La variable cible *Churn* a également été isolée afin de distinguer clairement les features ($X$) de la cible ($y$), conformément aux bonnes pratiques de modélisation supervisée.

### 2.4.2 Traitement des valeurs manquantes : Iterative Imputer (MICE)

Le traitement des valeurs manquantes constitue une étape clé du prétraitement des données, en particulier dans ce projet où près d'un tiers des observations comporte au moins une valeur manquante. À l'issue de l'analyse exploratoire, la méthode **Iterative Imputer**, basée sur le principe des **Multiple Imputation by Chained Equations (MICE)**, a été retenue. Ce choix s'explique par la capacité de cette approche à modéliser chaque variable contenant des valeurs manquantes en fonction des autres variables du jeu de données, permettant ainsi de préserver les relations statistiques, la variance et la structure multivariée des données. Cette méthode est particulièrement adaptée aux mécanismes de données manquantes de type *Missing At Random (MAR)* identifiés lors de la phase d'exploration.

Conformément aux bonnes pratiques en machine learning, l'imputation a été **appliquée exclusivement sur le jeu de données d'entraînement**. Les paramètres et modèles internes appris par l'Iterative Imputer à partir des données d'entraînement sont ensuite utilisés pour transformer le jeu de test, sans recalcul ni ajustement. Cette démarche est essentielle pour prévenir le **data leakage**, qui surviendrait si des informations statistiques issues du jeu de test influençaient le processus d'apprentissage du modèle.

Afin de garantir la reproductibilité et la cohérence du prétraitement, l'Iterative Imputer a été **intégré au sein d'un pipeline de prétraitement**. Cette intégration permet d'automatiser et d'enchaîner correctement les différentes transformations, tout en assurant que les mêmes opérations, dans le même ordre, sont appliquées aux données d'entraînement et de test. Elle facilite également la maintenance de la solution et son extension future, notamment lors de

l'ajout de nouveaux modèles ou de nouvelles données.

### 2.4.3 Mise à l'échelle des variables numériques : StandardScaler

Les variables numériques du jeu de données présentent des échelles et des ordres de grandeur très différents. Par exemple, certaines variables comme *Tenure* ou *OrderCount* prennent des valeurs relativement faibles, tandis que d'autres comme *CashbackAmount* ou *WarehouseToHome* peuvent atteindre des valeurs beaucoup plus élevées. Ces disparités peuvent biaiser l'apprentissage de certains algorithmes de machine learning, en donnant une importance excessive aux variables ayant les amplitudes les plus élevées.

Afin de pallier ce problème, une étape de mise à l'échelle des variables numériques a été appliquée à l'aide du **StandardScaler**. Cette méthode consiste à transformer chaque variable de manière à ce qu'elle ait une moyenne nulle et un écart-type égal à un, selon la formule de standardisation classique. Cette transformation permet d'harmoniser les distributions des variables et d'améliorer la stabilité numérique et la convergence des algorithmes d'apprentissage.

Conformément aux bonnes pratiques, le StandardScaler a été ajusté uniquement sur le jeu de données d'entraînement, en calculant les moyennes et écarts-types à partir de celui-ci. Ces paramètres sont ensuite réutilisés sans modification pour transformer le jeu de test. Cette démarche garantit la cohérence entre les jeux de données et permet d'éviter tout data leakage, qui pourrait survenir si des informations statistiques issues du jeu de test étaient utilisées lors de l'apprentissage.

L'intégration de la standardisation dans le pipeline de prétraitement assure une application systématique et reproductible de cette transformation, tant pour les données d'entraînement que pour les nouvelles données à prédire. Bien que certains modèles basés sur les arbres, comme XGBoost, soient moins sensibles à l'échelle des variables, cette étape reste pertinente dans une approche comparative et généralisable, notamment en vue de l'évaluation de plusieurs algorithmes et de l'évolution future de la solution.

### 2.4.4 Pipeline de prétraitement global

```
num_pipeline=Pipeline([
    ("imputer",IterativeImputer(random_state=0)),
    ("scaler",StandardScaler())
])
cat_pipeline=Pipeline([
    ("one_encoder",OneHotEncoder(handle_unknown='ignore'))
])

preprocess=ColumnTransformer([
    ("num_preprocess",num_pipeline,num_cols),
    ("cat_preprocess",cat_pipeline,cat_cols)
])
model_log=Pipeline([
    ("preprocessor",preprocess),
    ("model",LogisticRegression(random_state=0))
])
```

FIGURE 2.14 – Pipeline de prétraitement global

Afin de garantir la cohérence, la reproductibilité et la robustesse du processus de préparation des données, l'ensemble des étapes de prétraitement a été intégré au sein d'un **pipeline de prétraitement global**. Ce pipeline regroupe successivement le traitement des valeurs manquantes à l'aide de l'Iterative Imputer (MICE), l'encodage des variables catégorielles par **One-Hot Encoding**, ainsi que la mise à l'échelle des variables numériques via le StandardScaler.

L'utilisation d'un pipeline permet d'assurer que les mêmes transformations, dans le même ordre, et avec les mêmes paramètres, sont appliquées aux données d'entraînement, aux données de test, ainsi qu'à toute nouvelle observation future. Cette approche est essentielle pour prévenir le **data leakage**, en garantissant que les paramètres des transformations sont appris exclusivement à partir du jeu d'entraînement et simplement appliqués aux autres jeux de données.

Par ailleurs, le pipeline facilite l'intégration du prétraitement avec les modèles de machine learning, en permettant une chaîne de traitement unique allant des données brutes jusqu'à la prédiction finale. Il améliore également la maintenabilité et l'évolutivité de la solution, en rendant possible la modification ou le remplacement d'une étape spécifique sans remettre en cause l'ensemble du processus.

À l'issue de cette phase de préparation, les données sont désormais propres, cohérentes et adaptées à l'apprentissage automatique. Le chapitre suivant sera consacré à la modélisation du churn client, incluant la présentation des algorithmes retenus, la stratégie d'évaluation des performances et l'analyse des résultats obtenus.

# Chapitre 3

# Modélisation et choix des modèles

## 3.1 Choix des modèles de machine learning

La prédiction du churn client constitue un problème de classification binaire, dont l'objectif est d'estimer la probabilité qu'un client quitte la plateforme e-commerce. Le choix des modèles de machine learning doit donc prendre en compte plusieurs contraintes spécifiques : la nature tabulaire des données, la présence de relations potentiellement non linéaires entre les variables explicatives, le déséquilibre de la variable cible ainsi que le besoin d'interprétabilité pour faciliter l'exploitation des résultats par les équipes métier.

Dans ce cadre, plusieurs algorithmes ont été sélectionnés afin de comparer leurs performances et d'identifier le modèle le plus adapté au problème étudié. Les modèles retenus couvrent différents niveaux de complexité, allant d'un modèle linéaire de référence à des méthodes d'ensemble basées sur les arbres de décision, reconnues pour leur efficacité sur des données tabulaires. Cette approche comparative permet d'évaluer les compromis entre performance prédictive, robustesse et interprétabilité.

### 3.1.1 Régression logistique

La **régression logistique** a été retenue comme modèle de référence pour ce projet. Il s'agit d'un algorithme de classification linéaire largement utilisé pour les problèmes de churn, en raison de sa simplicité et de sa forte interprétabilité. Ce modèle estime directement la probabilité d'appartenance à la classe positive (client churné) à partir d'une combinaison linéaire des variables explicatives, transformée par une fonction logistique.

L'un des principaux avantages de la régression logistique réside dans sa capacité à fournir des **coefficients interprétables**, permettant d'identifier facilement l'influence positive ou négative de chaque variable sur le risque de churn. Elle constitue ainsi un excellent point de

comparaison pour évaluer les gains apportés par des modèles plus complexes.

Cependant, ce modèle présente des limites importantes, notamment son incapacité à capturer des relations non linéaires complexes et des interactions entre les variables. Ces limitations justifient l'exploration de modèles plus avancés dans les sections suivantes.

### 3.1.2 Random Forest

Le **Random Forest** est un algorithme d'ensemble basé sur la combinaison de plusieurs arbres de décision entraînés sur des sous-échantillons aléatoires des données et des variables. Il a été sélectionné pour sa capacité à modéliser des relations non linéaires complexes et à gérer efficacement les interactions entre les variables explicatives.

Contrairement aux modèles linéaires, le Random Forest est peu sensible aux distributions des variables et relativement robuste au bruit et aux outliers. Il s'adapte particulièrement bien aux jeux de données tabulaires hétérogènes, comme celui utilisé dans ce projet, qui combine des variables démographiques, comportementales et transactionnelles.

De plus, le Random Forest permet d'obtenir des **mesures d'importance des variables**, offrant un premier niveau d'interprétation globale du modèle. Néanmoins, bien que plus performant que la régression logistique, il peut devenir coûteux en calcul et moins interprétable à l'échelle individuelle, ce qui motive l'utilisation de méthodes d'explicabilité complémentaires.

### 3.1.3 XGBoost

**XGBoost (Extreme Gradient Boosting)** est un algorithme de boosting particulièrement performant et largement utilisé dans les compétitions de data science et les applications industrielles. Il repose sur un apprentissage séquentiel d'arbres de décision, où chaque nouvel arbre cherche à corriger les erreurs commises par les précédents.

Ce modèle a été choisi pour sa grande capacité prédictive, sa gestion efficace des relations non linéaires et son aptitude à traiter des jeux de données déséquilibrés grâce à des paramètres spécifiques tels que *scale_pos_weight*. XGBoost intègre également des mécanismes avancés de régularisation, réduisant le risque de surapprentissage.

Un autre avantage majeur de XGBoost réside dans sa compatibilité avec des méthodes d'explicabilité modernes, notamment **SHAP**, qui permettent d'interpréter les prédictions à la fois au niveau global et individuel. Cette caractéristique est essentielle dans le cadre de ce projet,

où l'objectif ne se limite pas à la prédiction du churn, mais inclut également la compréhension des facteurs explicatifs.

### 3.1.4 LightGBM

**LightGBM** est un algorithme de gradient boosting optimisé pour la performance et la rapidité d'entraînement. Il se distingue par une approche de croissance des arbres par feuilles (*leaf-wise*), permettant d'obtenir de meilleures performances avec un temps de calcul réduit, notamment sur des jeux de données volumineux ou complexes.

Ce modèle a été retenu afin de comparer ses performances à celles de XGBoost, tout en évaluant les gains potentiels en termes de vitesse et d'efficacité. LightGBM est particulièrement bien adapté aux données tabulaires et supporte nativement les variables catégorielles, ce qui en fait un candidat pertinent pour les problématiques de churn client.

À l'instar de XGBoost, LightGBM est compatible avec les méthodes d'explicabilité basées sur **SHAP**, permettant une analyse fine des contributions des variables. Son inclusion dans ce projet permet ainsi d'explorer un compromis intéressant entre performance, efficacité computationnelle et interprétabilité.

## 3.2 Stratégie de Validation Croisée (*Cross-Validation*)

```
skfcv=StratifiedKFold(n_splits=5,shuffle=True,random_state=42)
```

FIGURE 3.1 – Stratégie de validation croisée stratifiée

Afin de garantir une évaluation fiable et non biaisée des performances de généralisation des modèles, une approche de **validation croisée** est indispensable, permettant d'utiliser l'intégralité du jeu d'entraînement pour l'apprentissage et l'évaluation, réduisant ainsi la dépendance à une seule partition aléatoire. Compte tenu du fort déséquilibre de la variable cible (Churn), la méthode **Stratified K-Fold** a été implémentée, car elle est conçue pour s'assurer que chaque pli (*fold*) possède la même proportion de la variable cible que le jeu de données d'entraînement original, prévenant ainsi un biais d'évaluation. L'implémentation a opté pour **K=5 plis** (n_splits=5), ce qui implique que le modèle sera entraîné et évalué cinq fois, utilisant à chaque itération un sous-ensemble distinct comme pli de validation, avec un mélange aléatoire (shuffle=True) et un

état aléatoire fixe (random_state=42) pour assurer la reproductibilité des résultats. Cette stratégie garantit que les métriques de performance obtenues sur les modèles (comme la Régression Logistique ou XGBoost) seront des estimateurs robustes et stables de la performance réelle sur des données non vues.