Projet de prédiction d'intérêt pour l'assurance véhicule

Luc Allart et Erwan-Henri Burlisson

2025-01-07

Contents

1 Introduction

Dans le cadre de ce projet de Master 2 à l'ISUP, nous avons pour objectif de développer un modèle prédictif pour une compagnie d'assurance. Ce modèle devra déterminer la probabilité qu'un client ayant souscrit une assurance santé s'intéresse également à une assurance véhicule. Cette prédiction permettra à la compagnie d'optimiser ses stratégies de communication et de maximiser ses revenus en ciblant les clients les plus susceptibles de souscrire à une nouvelle assurance.

Les données utilisées dans ce projet proviennent de Kaggle, où elles ont été initialement proposées dans le cadre d'une compétition intitulée **Health Insurance Cross Sell Prediction** (https://www.kaggle.com/datasets/anmolkumar/health-insurance-cross-sell-prediction). La compétition avait pour objectif de prédire si **un client actuel serait-il intéressé par une assurance véhicule ?** En utilisant les informations des clients.

1.1 Critère de notation

Le critère de performance utilisé pour cette compétition était le **score ROC_AUC** (Receiver Operating Characteristic - Area Under the Curve), une métrique largement utilisée pour évaluer les performances des modèles de classification. Ce test évalue spécifiquement la capacité du modèle à prédire correctement la variable cible `Response`, qui indique si un client est intéressé par l'assurance véhicule (`1`) ou non (`0`).

1.2 Contexte des données

Les données fournies ne précisent pas le pays d'origine de l'assureur ou du portefeuille étudié. Cela limite certaines interprétations concernant le contexte socio-économique ou réglementaire, mais cela ne remet pas en question l'objectif d'analyse et de modélisation. Nous supposerons donc que les informations données sont génériques et applicables à un contexte international.

2 Description des données

Les données utilisées pour ce projet sont structurées comme suit :

2.1 Variables et leur définition

Variable	Définition
`id`	Identifiant unique pour chaque client
`Gender`	Genre du client
`Age`	Âge du client
`Driving_License`	1 : Le client possède un permis, 0 : Le client n'a pas de permis
`Region_Code`	Code unique pour la région du client
`Previously_Insured`	1 : Le client a déjà une assurance véhicule, 0 : Pas d'assurance véhicule
`Vehicle_Age`	Âge du véhicule
`Vehicle_Damage`	1 : Le véhicule a été endommagé, 0 : Pas de dommage
`Annual_Premium`	Montant de la prime annuelle
`Policy_Sales_Channel`	Code anonymisé pour le canal de vente utilisé
`Vintage`	Nombre de jours depuis l'association du client avec l'entreprise
`Response`	1 : Le client est intéressé, 0 : Pas d'intérêt

2.2 Structure des données et variable cible

Le jeu de données contient deux tables principales :

- **Données d'entraînement** : utilisées pour entraîner et tester le modèle (380 000 individus).
- **Données de test** : Cette base de donnée utilisée pour la soumission du hackathon ne contient pas la variable reponse et ne sera donc pas utilisée

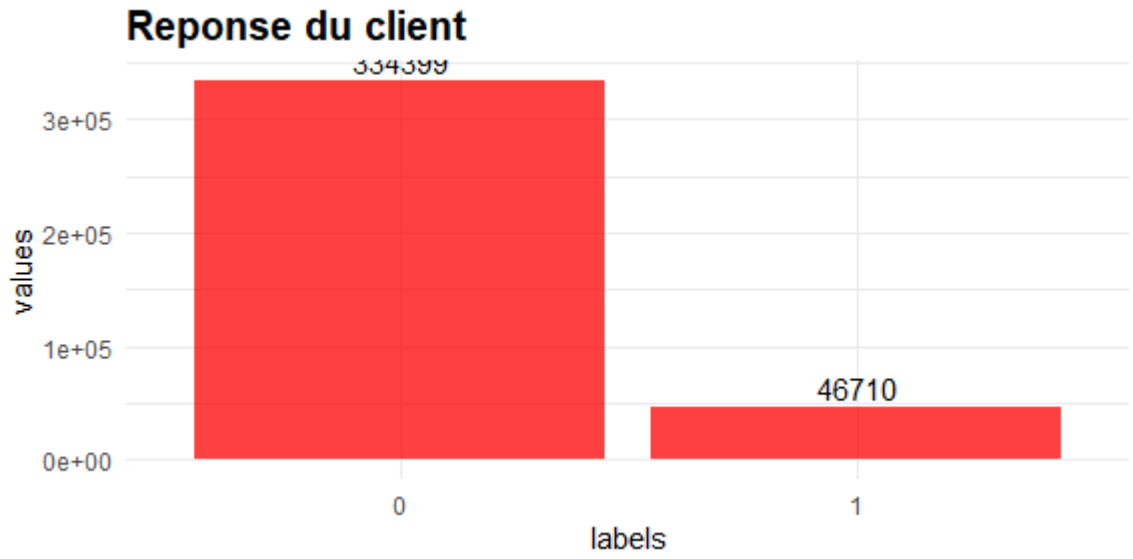
Chaque observation correspond à un client unique avec ses caractéristiques et son intérêt ou non pour l'assurance véhicule.

2.2.1 variable cible : Response

La variable cible dans notre analyse est `Response`, où :

- `1` signifie que le client est **intéressé par l'assurance véhicule**.
- `0` signifie que le client **n'est pas intéressé par l'assurance véhicule**.

Ce projet est donc une **tâche de classification binaire**, où l'objectif est de prédire si un client existant serait intéressé par une assurance véhicule.



Déséquilibre des classes

En analysant la distribution de la variable cible `Response`, nous constatons un **déséquilibre important** entre les deux classes (`0` et `1`). Ce type de déséquilibre est courant dans les jeux de données réels, mais il peut nuire à la performance des modèles de classification, car ils ont tendance à privilégier la classe majoritaire.

Pour pallier ce problème, nous pouvons utiliser des techniques de **suréchantillonnage** (oversampling) ou **sous-échantillonnage** (undersampling).

Suréchantillonnage et sous-échantillonnage

Ces techniques sont utilisées pour ajuster la répartition des classes dans un jeu de données, et elles sont courantes en apprentissage automatique et en analyse statistique :

- **Suréchantillonnage** : Augmente le nombre d'exemples de la classe minoritaire en dupliquant des observations ou en générant de nouvelles données synthétiques. Par exemple, l'algorithme SMOTE (Synthetic Minority Oversampling Technique) est souvent utilisé.
- **Sous-échantillonnage** : Réduit le nombre d'exemples de la classe majoritaire pour équilibrer les classes, mais cela peut entraîner une perte d'informations.

Approche proposée

Pour notre analyse, nous examinerons si l'impact du **suréchantillonnage** permet d'augmenter la précision globale ou la sensibilité aux prédictions de la classe minoritaire.

2.2.2 Etude de la variable : Gender

En analysant la variable `Gender`, qui représente le genre des clients, nous constatons que la distribution des données est **équilibrée**. Cela signifie que les deux catégories (hommes et femmes) sont représentées de manière à peu près égale dans le jeu de données.

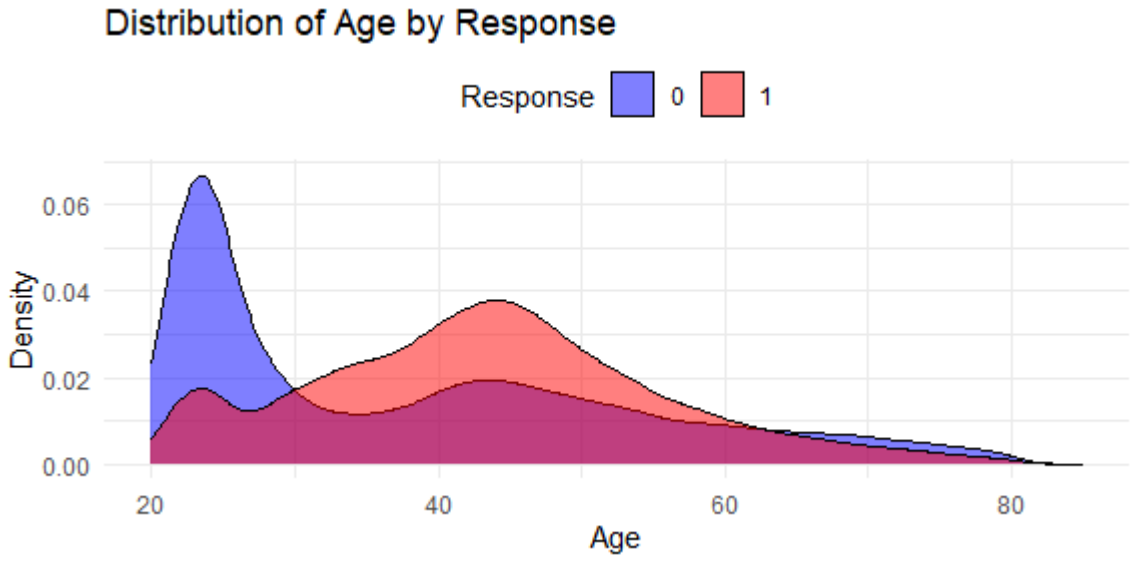
Observation

La présence d'une distribution équilibrée pour cette variable est importante, car cela réduit le risque de biais potentiel dans les modèles de classification qui pourraient autrement surpondérer une catégorie en raison d'un déséquilibre. Nous n'avons donc pas besoin d'appliquer des techniques de rééquilibrage pour cette variable.

Genre	Nombre
`Male`	206 089
`Female`	175 020

2.2.3 Etude de la variable : Age

L'âge des clients peut avoir une influence significative sur leur intérêt pour l'assurance véhicule. Pour mieux comprendre cette relation, nous avons regroupé les âges en classes afin d'analyser la répartition des réponses (`Response`) en fonction des tranches d'âge.



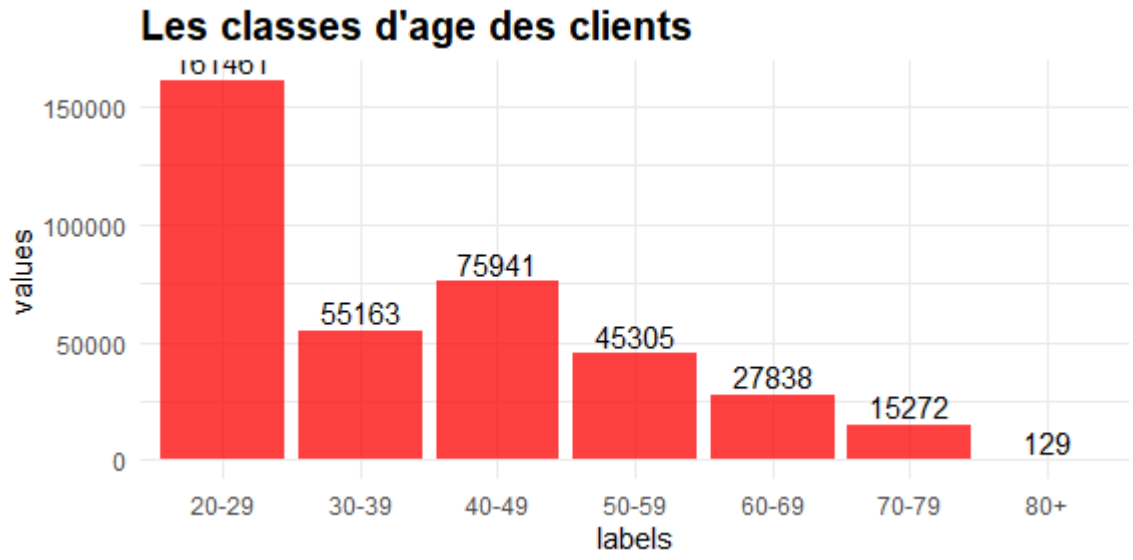
Observations initiales

Certaines tranches d'âge semblent montrer un plus grand intérêt pour l'assurance véhicule, tandis que d'autres en montrent moins. Cela pourrait indiquer une tendance ou un comportement spécifique à certaines catégories démographiques. Par exemple :

- Les clients âgés de **40-49 ans** et **70-79 ans** semblent avoir un intérêt notable.
- Les jeunes adultes (20-29 ans) pourraient être moins intéressés, mais cela nécessitera une analyse plus approfondie.

Groupement des âges

Cette classification permet d'identifier les tendances globales tout en simplifiant l'analyse.



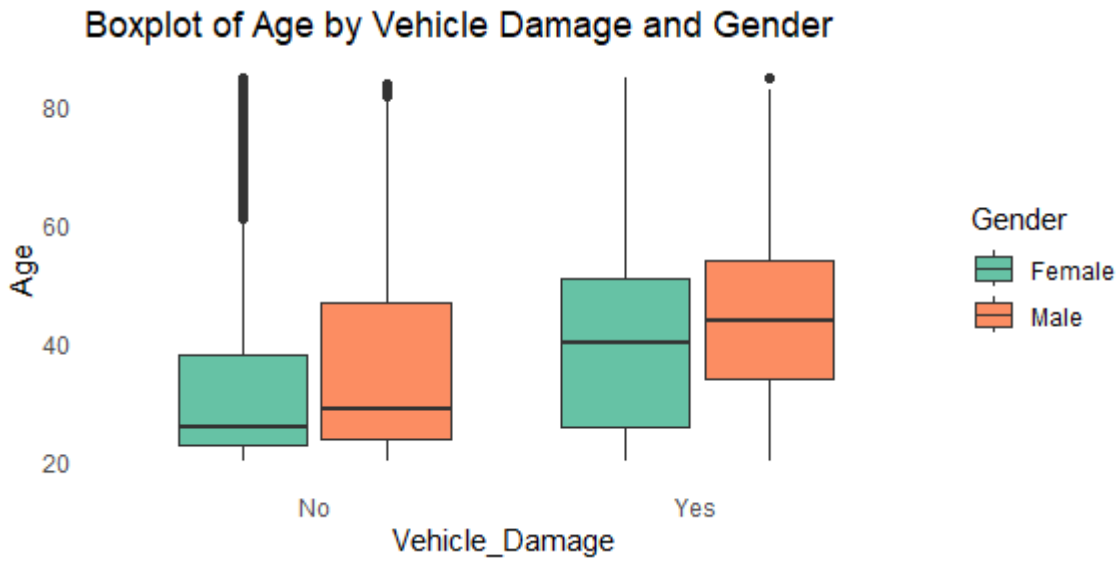
2.2.4 Etude de la variable : Vehicle Damage

Analyser la relation entre l'âge des clients et les dommages aux véhicules peut fournir des informations utiles pour mieux comprendre les comportements des clients et leur probabilité d'intérêt pour une assurance véhicule.

Observation

D'après les données, il semble que les personnes âgées soient plus susceptibles de posséder des véhicules ayant subi des dommages. Cela pourrait s'expliquer par plusieurs facteurs :

- Les conducteurs plus âgés possèdent peut-être des véhicules plus anciens ou moins bien entretenus.
- Ils peuvent également avoir un comportement de conduite différent, influençant le risque de dommages.



3 Prétraitement des Données

3.1 Détection de valeurs aberrantes et manquantes

Definition of Outliers

En statistique, un **outlier** (ou valeur aberrante) est un point de données qui diffère significativement des autres observations. Les outliers peuvent apparaître pour plusieurs raisons :

- **Variabilité dans les mesures** : Certaines valeurs extrêmes peuvent être des cas valides, mais inhabituels.
- **Erreurs expérimentales** : Ces erreurs peuvent provenir de problèmes de saisie ou de mesure et doivent souvent être corrigées ou exclues.

Les outliers peuvent causer de sérieux problèmes dans les analyses statistiques en influençant de manière disproportionnée les résultats, en particulier dans des modèles sensibles comme la régression.

Nous allons utilisé la méthode "Using Interquartile Range (IQR)" pour la détection de valeurs aberrantes

L'**intervalle interquartile (IQR)** est une méthode couramment utilisée pour détecter les outliers dans un jeu de données. Cette méthode repose sur les quartiles :

- **Q1** : Le 1er quartile représente la valeur sous laquelle se trouve 25 % des données.
- **Q3** : Le 3ème quartile représente la valeur sous laquelle se trouve 75 % des données.
- **IQR** : Calculé comme la différence entre Q3 et Q1 ($IQR = Q3 - Q1$).

Un point est considéré comme un **outlier** s'il se situe :

- En dessous de $Q1 - 1.5 \times IQR$,
- Au-dessus de $Q3 + 1.5 \times IQR$.

Conclusion

Valeurs manquantes : Aucune valeur manquante n'a été détectée dans le jeu de données.

Valeurs aberrantes (outliers) :

- La variable `Annual_Premium` contient **10 320 outliers** détectés à l'aide de la méthode de l'intervalle interquartile (IQR).
- Ces valeurs aberrantes pourraient influencer négativement les résultats de l'analyse et les performances des modèles prédictifs.

Pour garantir la robustesse des analyses et des modèles, nous avons décidé de **supprimer les lignes contenant des valeurs aberrantes** dans la variable `Annual_Premium`. Cette approche permet de travailler avec des données plus homogènes tout en réduisant l'impact des valeurs extrêmes sur les résultats.

3.2 Division en ensembles d'entraînement et de test et encodage des variables catégoriques

3.2.1 Division en ensembles d'entraînement et de test

Pour entraîner et évaluer le modèle, nous divisons les données nettoyées en deux ensembles :

- **Ensemble d'entraînement** : 67 % des données, utilisé pour construire le modèle.
- **Ensemble de test** : 33 % des données, utilisé pour évaluer les performances du modèle sur des données non vues.

3.2.2 Encodage des variables catégoriques

Les variables catégorielles doivent être encodées pour être utilisées dans les modèles d'apprentissage automatique. Dans ce projet, nous avons appliqué l'**Ordinal Encoding**. Méthode utilisée lorsque l'ordre des catégories est important (par exemple : **"cold", "warm", "hot"**). Cela permet de représenter chaque catégorie par une valeur numérique correspondant à son rang.

Gestion des valeurs inconnues :

Lors de l'encodage, nous avons remarqué que deux lignes dans l'ensemble de test contenaient des valeurs inconnues pour la variable `Policy_Sales_Channel` (`141` et `142`). Ces catégories n'étant pas présentes dans l'ensemble d'entraînement, nous les avons remplacées par une catégorie existante (`140`) afin de garantir une cohérence entre les ensembles de données.

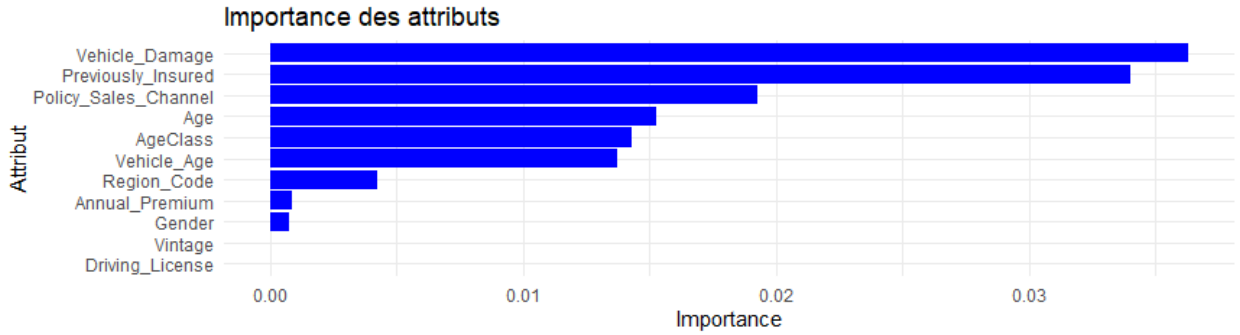
3.3 Sélection de variables

La sélection des variables est une étape essentielle dans la construction d'un modèle d'apprentissage automatique, car elle permet de réduire la complexité, d'améliorer les performances et de diminuer le risque de surapprentissage.

Observations sur l'importance des variables

Le graphique ci-dessous montre l'importance des variables calculée à l'aide de cette méthode :

- `Vehicle_Damage` et `Previously_Insured` sont les deux variables les plus importantes pour prédire l'intérêt pour l'assurance véhicule.
- `Age` et `AgeClass` ont une importance similaire, ce qui montre que l'ajout de `AgeClass` comme nouvelle variable ne perd pas d'information significative.
- Les variables comme `Vintage` et `Driving_License` ont des scores très faibles et pourraient être supprimées sans affecter les performances du modèle.



3.4 Suréchantillonnage avec ROSE

Dans notre projet, nous avons observé un déséquilibre dans la distribution de la variable cible `Response`. Ce déséquilibre peut affecter les performances des modèles prédictifs, car ils risquent de privilégier la classe majoritaire. Pour résoudre ce problème, nous appliquons un **suréchantillonnage** à l'aide du package **ROSE**.

Méthode utilisée : ROSE (Random Over-Sampling Examples)

Le package **ROSE** propose une approche de suréchantillonnage qui génère des données synthétiques pour équilibrer les classes. Contrairement au simple suréchantillonnage (duplication des observations de la classe minoritaire), ROSE crée de nouvelles observations basées sur une approche probabiliste, ce qui améliore la diversité et réduit le risque de surajustement.

3.5 Normalisation des jeux de données

La normalisation est une étape cruciale pour standardiser les données avant de les utiliser dans des algorithmes de modélisation. Cette étape garantit que toutes les variables ont une échelle comparable, ce qui est particulièrement important pour les modèles sensibles aux écarts de valeurs (par exemple, la régression logistique, les réseaux neuronaux ou les SVM).

Méthode utilisée : Normalisation avec `caret`

Nous avons utilisé la bibliothèque **caret** pour appliquer une normalisation sur les ensembles de données d'entraînement et de test. Les étapes sont les suivantes :

- Entraîner un normaliseur sur l'ensemble d'entraînement encodé.
- Transformer les ensembles d'entraînement et de test pour centrer les données (moyenne = 0) et les réduire (écart-type = 1).

Après normalisation :

- Les données sont centrées (moyenne = 0) et réduites (écart-type = 1).
- Cela améliore la performance des algorithmes de modélisation en garantissant que toutes les variables contribuent équitablement.

4 Entrainement et comparaison des modèles

Dans cette section, nous avons entraîné plusieurs modèles d'apprentissage supervisé sur les données transformées et suréchantillonnées. L'objectif est de comparer leurs performances en termes de **ROC AUC**, une métrique appropriée pour les problèmes de classification déséquilibrés.

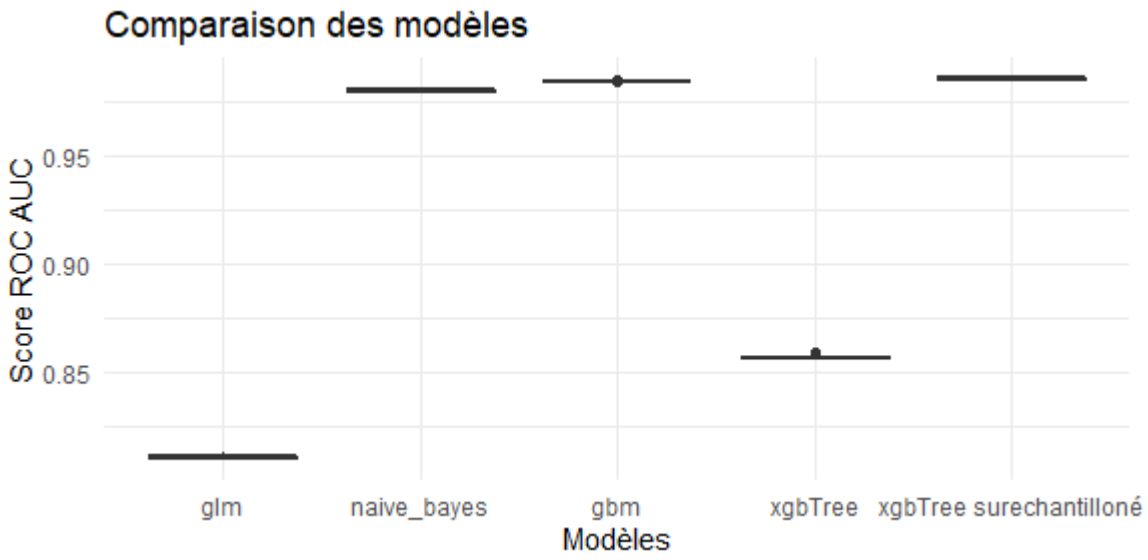
4.1 Modèles entraînés

Les modèles suivants ont été évalués :

1. **GLM (Generalized Linear Model)** : Régression logistique classique La régression logistique est un modèle linéaire généralisé adapté aux données binaires (ou multicatégorielles). Elle prédit la probabilité d'appartenance à une classe en appliquant une fonction logistique (sigmoïde) à une combinaison linéaire des variables explicatives.
2. **Naive Bayes** : Modèle probabiliste basé sur le théorème de Bayes Naive Bayes est un modèle de classification basé sur le théorème de Bayes, supposant l'indépendance conditionnelle des variables explicatives. Il calcule la probabilité d'appartenance d'un individu à chaque classe et choisit la classe avec la probabilité la plus élevée.
3. **GBM (Gradient Boosting Machine)** : Modèle de boosting pour améliorer la performance GBM construit un ensemble de modèles faibles (typiquement des arbres de décision) de manière séquentielle. Chaque nouveau modèle corrige les erreurs commises par les modèles précédents, en minimisant une fonction de perte grâce à une descente de gradient.
4. **XGBoost sans suréchantillonnage** : XGBoost est une version optimisée de GBM qui utilise des techniques avancées (régularisation, gestion efficace de la mémoire) pour améliorer la vitesse et les performances. Dans cette version, l'algorithme est entraîné sur les données originales, non équilibrées, ce qui peut conduire à un biais en cas de classes déséquilibrées.
5. **XGBoost avec suréchantillonnage** : Cette version de XGBoost est entraînée sur des données équilibrées à l'aide de la méthode ROSE (Random OverSampling Examples), qui crée des échantillons synthétiques pour la classe minoritaire. Cela vise à atténuer les effets du déséquilibre des classes et à améliorer la performance sur la classe minoritaire.

Tous les modèles sont entraînés en utilisant la fonction `caret::train()`. Et on utilise la méthode de validation croisée (5-fold) pour évaluer les performances des modèles avec comme métrique principale : **ROC AUC**.

4.1.1 Résultats des modèles



Les résultats moyens de la métrique **ROC AUC** sur l'ensemble d'entraînement sont les suivants :

Model Name	Train ROC AUC Mean
GLM	0.8114
Naive Bayes	0.9622
GBM	0.9811
XGBoost sans suréchantillonnage	0.8539
XGBoost surechantilloné	0.9845

Observations

1. **Meilleure performance** :
   - Le modèle **XGBoost avec suréchantillonnage** a obtenu le **ROC AUC le plus élevé (0.9845)**, confirmant l'importance du rééquilibrage des données pour améliorer les performances.
2. **Performance du Naive Bayes** :
   - Malgré sa simplicité, le modèle Naive Bayes a atteint un ROC AUC élevé (0.9622), démontrant sa robustesse dans les données équilibrées.
3. **Impact du suréchantillonnage** :
   - Comparativement à **XGBoost sans suréchantillonnage (0.8539)**, l'utilisation de données équilibrées a significativement amélioré les performances.

Conclusion

Le modèle **XGBoost avec suréchantillonnage** est identifié comme le meilleur candidat pour résoudre ce problème de classification. Les prochaines étapes consisteront à tester ce modèle sur l'ensemble de test et à analyser ses performances sur des données non vues.

4.2 Modèle Sélectionné : XGBoost

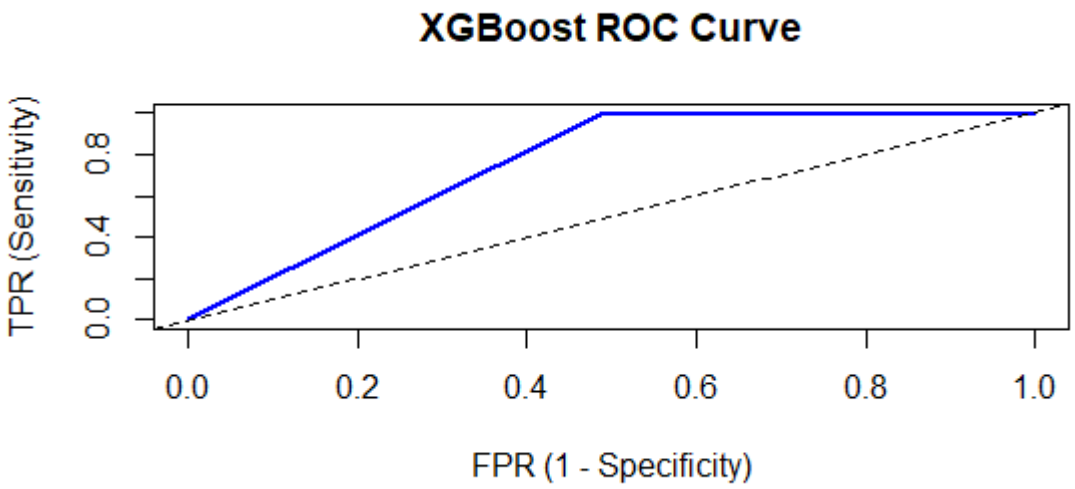
Dans cette section, nous évaluons les performances du modèle **XGBoost**.

4.2.1 Résultats de l'évaluation (sur l'ensemble de test)

ROC AUC : La valeur de **ROC AUC** obtenue est de **0.7553**.

Erreur de classification : Le taux d'erreur de classification est de **42.74 %**.

Le graphique ci-dessous représente la **courbe ROC** obtenue pour le modèle **XGBoost** sur l'ensemble de test.



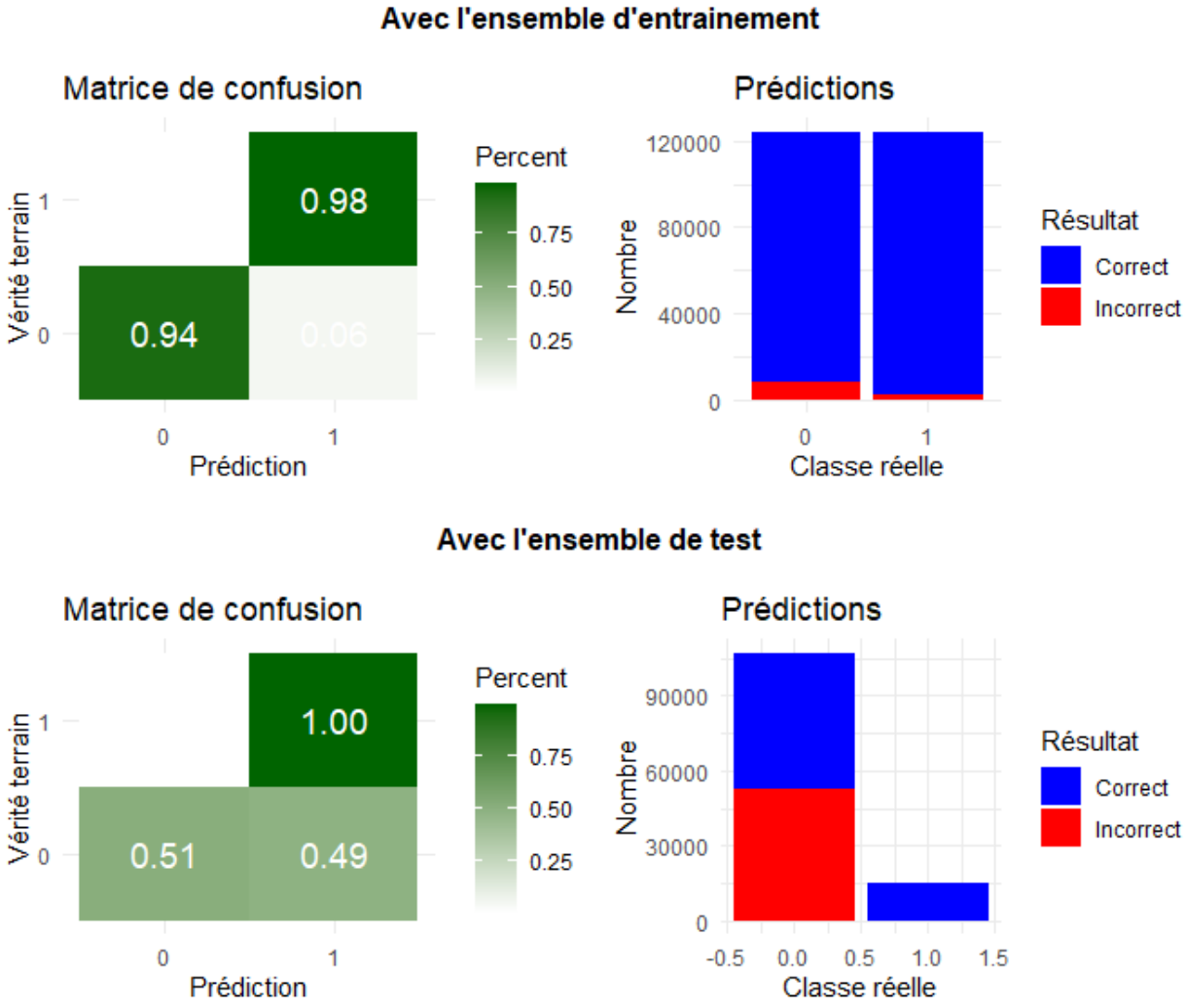
Interprétation :

- L'axe des abscisses représente le **FPR (1 - Specificité)**, qui mesure les faux positifs.
- L'axe des ordonnées représente le **TPR (Sensibilité)**, qui mesure les vrais positifs.
- La ligne en pointillés représente la ligne de base où **AUC = 0.5**, ce qui correspond à une classification aléatoire.

Le modèle affiche une courbe située au-dessus de la ligne de base, confirmant une performance au-dessus du hasard.

4.2.2 Prédictions et Matrice de confusion

La matrice de confusion ci-dessous montre les proportions des prédictions correctes et incorrectes. Les données ont été normalisées pour permettre une interprétation claire des pourcentages.



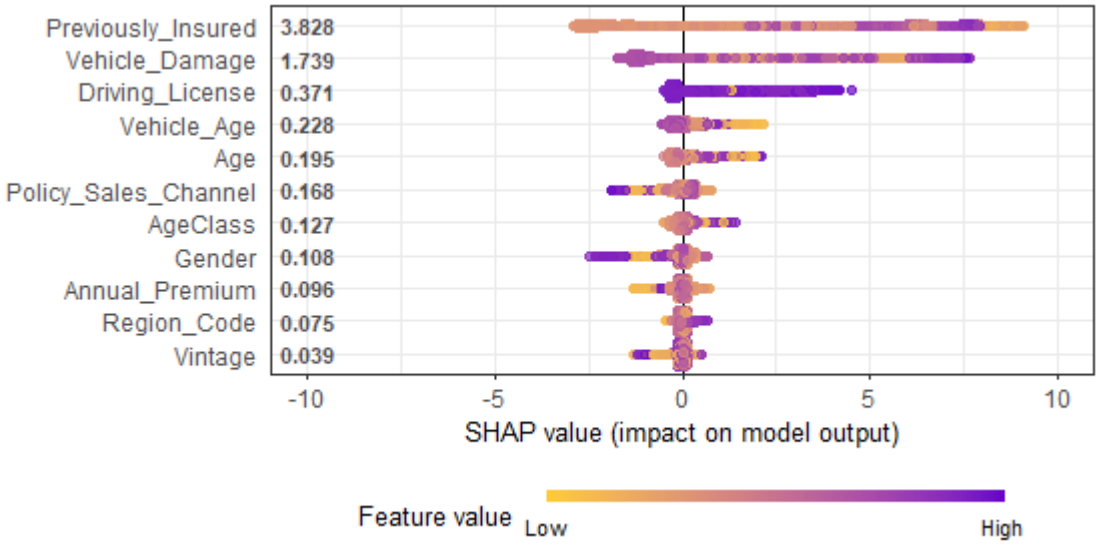
Observations :

- Dans la matrice de confusion sur l'ensemble d'entrainement :
  - **Classe `0` (Non intéressé)** :
    - **94 %** des instances ont été correctement classées.
    - **6 %** des instances ont été incorrectement classées comme `1`.
  - **Classe `1` (Intéressé)** :
    - **98 %** des instances ont été correctement classées.
    - **2 %** des instances ont été incorrectement classées comme `0`.
- Nous retrouvons les mêmes informations avec le nombre d'individus plutôt que des pourcentages dans le graphique de droite

Pour l'ensemble de test, les résultats sont bien moins convaincants, avec 49 % des 0 prédits comme des 1, mais 100 % des 1 correctement prédits. Cela pourrait s'expliquer par le fait que la base de données de test n'est pas équilibrée entre les 1 et les 0, contrairement à la base d'entraînement.

4.2.3 SHAP Analysis: Feature Importance and Impact

Les valeurs **SHAP** (SHapley Additive exPlanations) sont utilisées pour expliquer l'impact de chaque caractéristique sur les prédictions du modèle. Cette approche fournit une interprétation globale et locale des modèles complexes, tels que **XGBoost**.



Observations :

1. **Caractéristiques les plus influentes** :
   - `Age` a la plus grande influence sur les prédictions, avec des valeurs SHAP élevées.
   - `Policy_Sales_Channel` et `Vehicle_Damage` suivent, contribuant également de manière significative aux décisions du modèle.
2. **Impact des caractéristiques** :
   - Les caractéristiques avec des valeurs SHAP positives augmentent la probabilité d'appartenir à la classe cible (`1`), tandis que celles avec des valeurs négatives la réduisent.
   - Par exemple, les valeurs élevées de `Age` ont un impact positif significatif sur les prédictions.
3. **Importance globale vs locale** :
   - Les caractéristiques comme `Region_Code` et `Driving_License` ont un impact global plus faible, mais peuvent influencer certaines observations de manière importante.

Conclusion

L'analyse SHAP fournit une compréhension approfondie de l'impact des caractéristiques sur les prédictions du modèle :

1. `Age`, `Policy_Sales_Channel`, **et** `Vehicle_Damage` sont les caractéristiques les plus influentes.
2. L'interprétation des valeurs SHAP peut guider des ajustements futurs, comme le raffinement des caractéristiques ou l'exclusion des variables peu informatives.

5 Conclusion

5.1 Objectif et Résultats

Dans le cadre de ce projet, nous avons développé un modèle prédictif visant à déterminer si un client ayant souscrit à une assurance santé serait également intéressé par une assurance véhicule. Ce projet s'est articulé autour des étapes suivantes :

1. **Analyse exploratoire des données** :
   - Identification des tendances importantes dans les variables.
   - Mise en évidence de déséquilibres dans la variable cible `Response`.
2. **Préparation des données** :
   - Nettoyage des données, encodage des variables catégorielles, normalisation et gestion du déséquilibre par suréchantillonnage (ROSE).
3. **Modélisation et évaluation** :
   - Entraînement de plusieurs modèles, incluant GLM, Naive Bayes, Gradient Boosting (GBM) et XGBoost.
   - Évaluation des modèles à l'aide de métriques telles que le **ROC AUC** et la courbe ROC.

5.2 Résultats principaux

1. **Performance du meilleur modèle** :
   - Le modèle **XGBoost avec suréchantillonnage** a obtenu les meilleurs résultats, avec un **ROC AUC de 0.9845** sur l'ensemble d'entraînement.
   - Ce modèle s'est également révélé robuste lors de l'évaluation sur des données de test.
2. **Interprétation des caractéristiques** :
   - L'analyse des valeurs **SHAP** a montré que des variables telles que `Age`, `Policy_Sales_Channel`, et `Vehicle_Damage` étaient les plus influentes dans les prédictions.
   - Des caractéristiques moins influentes, comme `Vintage` ou `Region_Code`, pourraient être exclues dans de futures itérations.
3. **Gestion du déséquilibre des classes** :
   - L'utilisation de ROSE a permis d'équilibrer les classes, améliorant ainsi la performance des modèles, notamment pour la classe minoritaire (`1` - clients intéressés).

5.3 Limites et Perspectives

1. **Limites** :
   - Le déséquilibre initial de la variable cible a nécessité un suréchantillonnage, ce qui peut introduire des biais si les données synthétiques ne reflètent pas la réalité.
   - Certaines variables, comme `Region_Code` ou `Vintage`, ont montré un faible impact et pourraient être optimisées ou supprimées.
2. **Perspectives** :
   - Explorer des techniques d'optimisation des hyperparamètres, telles que l'optimisation bayésienne, pour améliorer davantage les performances des modèles.
   - Intégrer des données contextuelles (par exemple, plus d'informations sur les clients ou sur les politiques d'assurance) pour enrichir le modèle.