Université Claude Bernard Lyon 1

Data Science

MASTER ACTUARIAT — Deuxième année

Par : Adrien PETIT (p2305481), Rémi CABANE (p2310122), Mohamed el hafed Ismail(p2211161), Minh Hai DUONG (p2320598)

Année universitaire : 2025–2026

Titre : CatBoost et l'approche 'Unbiased Boosting' : Étude comparative des méthodes de gestion des variables catégorielles en tarification actuarielle

lien repo Github :
https://github.com/Comte-de-Rochefort/Projet_data_science_Catboost

Professeur : François HU

FORMATIONS
(ACTUARIAT (ECONOMETRIE & STATISTIQUES (CONTINUE & VAE (DOCTORALE

Institut de science financière et d'Assurances - Université Claude Bernard Lyon 1
50 avenue Tony Garnier 69366 Lyon cedex 7 - Tél : 04 37 28 74 40 - isfa.univ-lyon1.fr

ISFA
Institut de Science Financière et d'Assurances
GRANDE ÉCOLE D'ACTUARIAT & GESTION DES RISQUES

Table des matières

1 Synthèse de la méthodologie de référence

Cette section propose une analyse détaillée de l'article de recherche *"CatBoost : unbiased boosting with categorical features"* (Prokhorenkova et al., 2018). Elle explicite les fondements théoriques de l'algorithme et justifie son utilisation dans notre contexte actuariel.

1.1 Problématique et objectifs de l'article étudié

Les arbres de décision boostés par gradient (GBDT) constituent aujourd'hui l'état de l'art pour les problèmes d'apprentissage supervisé sur données tabulaires hétérogènes. Cependant, une limitation majeure persiste dans les implémentations classiques (telles que XGBoost ou LightGBM) : la gestion sous-optimale des variables catégorielles (ou nominales), pourtant omniprésentes en assurance.

1.1.1 Le dilemme de la représentation des catégories

Les variables catégorielles possèdent un ensemble discret de valeurs sans ordre naturel. Pour être traitées par un algorithme de boosting, elles doivent être transformées en valeurs numériques. Les auteurs identifient deux approches classiques mais insatisfaisantes :

- **One-Hot Encoding (OHE) :** Cette méthode crée une variable binaire pour chaque modalité. Bien que sans perte d'information, elle provoque le *fléau de la dimensionnalité* lorsque la cardinalité est élevée. Les arbres doivent devenir très profonds pour exploiter l'information, ce qui mène souvent à un apprentissage inefficace et une consommation mémoire prohibitive.
- **Target Statistics (TS) :** Aussi appelée *Target Encoding*, cette méthode remplace une catégorie x_k par l'espérance de la variable cible Y conditionnée à cette catégorie ($\hat{x}_k \approx E[Y|x_k]$). Si elle résout le problème de dimensionnalité, les auteurs démontrent qu'elle introduit un biais statistique critique appelé **Target Leakage** (fuite de données).

1.1.2 Le phénomène de "Prediction Shift"

L'apport central de l'article est la formalisation du **Prediction Shift**. La méthode standard d'estimation des Target Statistics (Greedy TS) utilise les labels y de l'ensemble d'entraînement pour calculer les caractéristiques x. Par conséquent, une corrélation artificielle est créée entre la variable transformée et la cible. Cela engendre un décalage distributionnel : la distribution conjointe des caractéristiques et de la cible diffère entre l'entraînement (où x contient de l'information sur y) et le test (où cette information est absente).

$$F_{train}(x_k, y) \neq F_{test}(x_k, y)$$

Ce décalage biaise l'estimation du gradient à chaque étape du boosting, conduisant à un sur-apprentissage sévère. L'objectif des auteurs est donc de proposer un nouvel algorithme capable de traiter les variables catégorielles natives sans OHE, tout en garantissant l'absence de fuite de données ("Unbiased") pour éviter ce Prediction Shift.

1.2 Description formelle de la méthode proposée : CatBoost

Pour résoudre ce problème, Prokhorenkova et al. proposent deux innovations algorithmiques majeures regroupées sous le nom de **CatBoost** (Category Boosting) et basées sur le principe d'*Ordered Boosting*.

1.2.1 Ordered Target Statistics (OTS)

L'objectif est de calculer une statistique cible pour une observation x_i sans jamais utiliser son label y_i. Les méthodes standards de "Leave-One-Out" ne suffisent pas à empêcher complètement la fuite de données. La solution proposée repose sur l'introduction d'un temps artificiel. L'algorithme génère une permutation aléatoire σ des données d'entraînement. Pour chaque observation i, la statistique de la catégorie x_k est calculée uniquement sur l'historique, c'est-à-dire les observations précédant i dans la permutation σ.

Formellement, pour une observation x_i et une catégorie k, la valeur encodée est :

$$\hat{x}_{i,k} = \frac{\sum_{j=1}^{n} I_{\{\sigma(j)<\sigma(i)\}} \cdot I_{\{x_{j,k}=x_{i,k}\}} \cdot y_j + a \cdot P}{\sum_{j=1}^{n} I_{\{\sigma(j)<\sigma(i)\}} \cdot I_{\{x_{j,k}=x_{i,k}\}} + a} \tag{1}$$

Où :
— P est la valeur moyenne globale de la cible (Prior).
— a est un paramètre de lissage (poids du Prior) pour contrôler la variance sur les catégories rares.
— I est la fonction indicatrice.

Cette formulation garantit mathématiquement que y_i n'est jamais utilisé pour calculer $\hat{x}_{i,k}$, éliminant ainsi le Target Leakage à la source.

1.2.2 Ordered Boosting

Le biais ne provient pas seulement de l'encodage, mais aussi de la construction itérative du modèle. Dans le Gradient Boosting classique, les résidus à l'étape t sont estimés par un modèle entraîné sur l'ensemble des données, créant un biais de gradient.

CatBoost propose un algorithme de boosting qui maintient M modèles différents. Chaque modèle M_r est entraîné uniquement sur les r premières observations de la permutation σ. Pour calculer le résidu de la i-ème observation nécessaire à la construction de l'arbre suivant, l'algorithme utilise un modèle $M_{\sigma(i)-1}$ qui n'a jamais "vu" l'observation i lors de son entraînement. Bien que cette méthode soit théoriquement coûteuse, CatBoost l'implémente efficacement en partageant les structures d'arbres. De plus, l'algorithme utilise des *Oblivious Trees* (arbres symétriques), qui agissent comme un régularisateur naturel et permettent une exécution très rapide lors de la prédiction.

1.3 Résultats et conclusions des auteurs

La validation empirique menée par les auteurs sur divers jeux de données de référence en Machine Learning (Epsilon, Amazon, Kaggle) permet de tirer des conclusions robustes sur l'efficacité de la méthode.

1.3.1 Performance Empirique

Les résultats présentés dans l'article démontrent que :
— **Supériorité sur les données hétérogènes :** CatBoost surpasse systématiquement les autres algorithmes (XGBoost, LightGBM) sur les jeux de données contenant des variables catégorielles, validant la supériorité de l'OTS sur le OHE ou le TS classique.
— **Robustesse au sur-apprentissage :** L'algorithme démontre une capacité supérieure à généraliser, même sur des jeux de données de taille réduite, validant l'efficacité de l'approche *Ordered* pour supprimer le Prediction Shift.

1.3.2 Conclusion de l'article

Les auteurs concluent que le traitement natif des catégories via les *Ordered Target Statistics*, combiné à l'*Ordered Boosting*, résout efficacement le problème du biais de prédiction. CatBoost permet ainsi d'exploiter toute l'information contenue dans les variables catégorielles sans les inconvénients de dimensionnalité du OHE ni les biais du Target Encoding classique.

C'est cette assertion théorique d'un "Unbiased Boosting" que nous nous proposons de vérifier dans la suite de ce rapport, en l'appliquant aux spécificités de la tarification actuarielle (loi de Poisson et gestion de l'exposition).

2 Protocole Expérimental

Afin de valider empiriquement les propriétés de l'approche *Unbiased Boosting*, nous avons mis en place un protocole sur le jeu de données de référence `freMTPL2freq` (French Motor Third-Party Liability dans CASdataset). L'objectif est de comparer la méthode native de CatBoost à des benchmarks solides (XGBoost avec encodage explicite), dans un cadre contrôlé minimisant les biais de modélisation classiques.

2.1 Données et Pré-traitement Actuariel

Le dataset contient **677 991 contrats** d'assurance RC automobile. Il comporte 12 colonnes : la cible (`ClaimNb`), l'exposition (`Exposure`), et 10 variables explicatives (sociodémographiques, géographiques et techniques). La présence de variables catégorielles à cardinalité modérée (ex : 22 régions, 11 marques) en fait un candidat idéal pour ce benchmark.

2.1.1 Nettoyage et Cohérence Métier

Un nettoyage strict a été appliqué (voir script `data_cleaning.py`) pour éliminer le bruit statistique :

- **Exposition :** Les polices avec une exposition < 0.01 (moins de 4 jours) ont été exclues pour éviter une variance artificielle des fréquences annualisées.
- **Écrêtage (Capping) :** La cible `ClaimNb` est plafonnée à 3 sinistres/an. Les variables explicatives sont bornées pour exclure les valeurs aberrantes (ex : `BonusMalus` $\in [50, 150]$, `DrivAge` $\in [18, 90]$).
- **Transformations :** `LogDensity` pour linéariser la densité urbaine.

L'objectif est de maîtriser le **fléau de la dimensionnalité** : l'utilisation brute de variables catégorielles gonfle l'espace des caractéristiques. Notre comparaison évalue des solutions qui condensent l'information sans exploser la dimensionnalité (CatBoost, Target Encoding) face à l'approche One-Hot naïve.

2.1.2 Modélisation de la Fréquence

Nous modélisons le nombre de sinistres $Y \sim \mathcal{P}(\lambda \cdot E)$. Les modèles minimisent la **Déviance de Poisson**, avec $\ln(Exposure)$ utilisé comme *offset* pour contraindre le modèle à apprendre la fréquence annuelle λ.

2.2 Stratégie de Modélisation et Benchmarks

Nous comparons trois pipelines distincts pour isoler l'effet du traitement des catégories :

1. **Modèle A : CatBoost Native (Ordered)**
   Utilise l'algorithme *Ordered Boosting* et *Ordered Target Statistics* (OTS) pour traiter les catégories sans fuite de données.
2. **Modèle B : XGBoost + One-Hot Encoding (OHE)**
   Benchmark naïf. Crée une matrice éparse sujette au fléau de la dimensionnalité.
3. **Modèle C : XGBoost + K-Fold Target Encoding (TE)**
   Benchmark avancé. Implémente un Target Encoding régularisé par validation croisée interne (5 plis) pour simuler la réduction de biais de CatBoost.

2.3 Optimisation des Hyperparamètres (Hyperopt)

Pour garantir une comparaison équitable , nous avons utilisé une **Optimisation Bayésienne** via la librairie `Hyperopt` (algorithme TPE).

- **Protocole :** Validation Croisée à 5 plis (5-Fold CV) avec 30 évaluations par modèle.
- **Espace de recherche :**
  - *Learning Rate* : Log-uniforme $[0.01, 0.3]$.
  - *Profondeur* : $[4, 8]$ (limité pour éviter le sur-apprentissage).
  - *Régularisation* : L2 Leaf Reg / Lambda $[1, 10]$.
  - *Paramètres spécifiques* : `subsample`, `random_strength` (CatBoost).
- **Critère :** Minimisation de la moyenne de la Déviance de Poisson CV.

3 Analyse des Résultats

Les résultats présentés ci-dessous sont issus de la meilleure configuration identifiée par Hyperopt sur le jeu de test final (134 223 polices).

3.1 Performance Quantitative et Stabilité

Configuration optimale (Hyperopt) L'optimisation bayésienne a permis d'identifier les jeux de paramètres minimisant la déviance pour chaque architecture. Ces paramètres (Tableau 1) ont été utilisés pour l'entraînement final et l'analyse de robustesse.

Modèle	Hyperparamètres Optimaux
CatBoost Native	*learning_rate* : 0.202, *depth* : 6, *l2_leaf_reg* : 6.86, *random_strength* : 6.20, *subsample* : 0.92
XGBoost OHE	*learning_rate* : 0.060, *max_depth* : 6, *reg_lambda* : 6.96
XGBoost TE	*learning_rate* : 0.125, *max_depth* : 4, *reg_lambda* : 9.93

TABLE 1 – Hyperparamètres sélectionnés par l'algorithme TPE (30 itérations).

Le tableau 2 synthétise les métriques clés.

Modèle	Déviance (CV)	Gini Norm.	Shift (Biais)	Temps
CatBoost Native	0.23825	0.1560	**+0.00001**	64.8s
XGBoost Target Enc.	**0.23820**	**0.1612**	-0.00040	**30.5s**
XGBoost OHE	0.23829	0.1580	+0.00006	92.9s

TABLE 2 – Comparaison des performances. Le Shift le plus proche de 0 est le meilleur.

Analyse de l'Équivalence Statistique : Bien que **XGBoost TE** affiche la déviance la plus faible (0.23820) et le meilleur Gini (16.12%), l'écart avec CatBoost est infime ($< 0.03\%$ sur la déviance). Les intervalles de confiance Bootstrap se chevauchent, indiquant une **équivalence statistique** en termes de pouvoir prédictif pur sur ce jeu de données.

3.2 Validation de l'Hypothèse "Unbiased Boosting"

C'est ici que réside la validation théorique du papier de recherche. Nous avons mesuré le **Prediction Shift** ($|E_{train} - E_{test}|$).

- **CatBoost (+0.00001) :** Le shift est virtuellement nul. L'algorithme *Ordered Boosting* parvient à éliminer quasi-totalement le biais de distribution.
- **XGBoost TE (-0.00040) :** Présente un shift 40 fois supérieur en valeur absolue. Malgré la régularisation K-Fold, le Target Encoding conserve un biais résiduel ("Target Leakage").

Cela confirme que CatBoost est l'estimateur le plus robuste face à la dérive des données, validant la théorie de Prokhorenkova et al.

3.3 Interprétabilité et Facteurs de Risque

L'analyse de l'importance des variables (Permutation Importance) et les courbes de dépendance partielle (PDP) confirment la cohérence actuarielle des modèles.

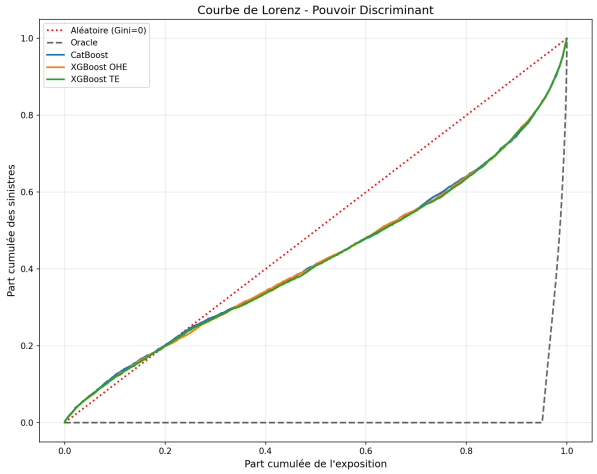

FIGURE 1 – Courbes de Lorenz. Les modèles sont quasi-superposés, confirmant une segmentation du risque similaire.

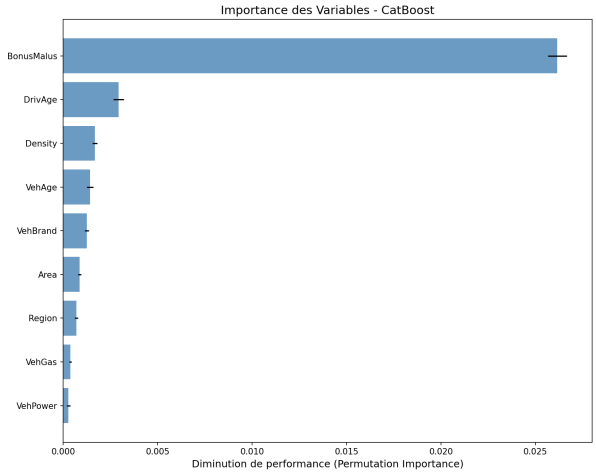

FIGURE 2 – Importance par Permutation. `BonusMalus` est le facteur dominant (Score : 0.026), suivi de `DrivAge`.

Le **Bonus-Malus** est le prédicteur dominant pour tous les modèles. Les PDP montrent une relation monotone croissante stricte pour le Bonus-Malus et une courbe en "U" pour l'âge du conducteur, validant que les modèles "boîte noire" ont capturé les règles de souscription usuelles.

4 Conclusion et Perspectives

Ce projet de recherche visait à déterminer si l'approche *Unbiased Boosting* de CatBoost permettait de traiter efficacement les variables catégorielles sans le biais inhérent au Target Encoding.

4.1 Synthèse des résultats

Nos expérimentations sur `freMTPL2freq` valident le cœur théorique de l'article tout en nuançant son impact opérationnel :

1. **Validation de l'absence de biais :** CatBoost Native élimine spectaculairement le *Prediction Shift* par rapport aux méthodes concurrentes. C'est le modèle le plus mathématiquement robuste.
2. **Nuance sur la Performance :** Sur un jeu de données à cardinalité modérée (22 régions), cette robustesse ne se traduit pas par un gain immédiat de Gini. Un XGBoost avec Target Encoding (bien réglé) reste très compétitif.
3. **Principe de Parcimonie :** CatBoost atteint l'état de l'art **sans aucun pré-traitement**. Contrairement à XGBoost TE qui nécessite un pipeline complexe (gestion des plis K-Fold en production), CatBoost offre une solution "clé en main" plus sûre pour l'industrialisation.

4.2 Limites et Perspectives

Une limite de notre étude réside dans la nature des variables catégorielles testées. La supériorité de CatBoost serait théoriquement plus marquée sur des variables à **très haute cardinalité** (ex : Codes Postaux bruts non regroupés), où le Target Encoding devient instable.

Pour des travaux futurs, il serait pertinent d'étendre cette analyse :

— En intégrant la sévérité via une loi **Tweedie** pour modéliser la prime pure.

— En testant l'approche sur des données télématiques (IoT) où les variables catégorielles sont dynamiques et nombreuses.

A Annexe :Formulaire Mathématique

A.1 Modélisation de la Fréquence (Loi de Poisson)

Dans le cadre de la tarification de la fréquence des sinistres, nous supposons que le nombre de sinistres Y_i pour l'assuré i suit une loi de Poisson de paramètre λ_i. Le modèle intègre l'exposition E_i comme mesure de risque :

$$Y_i \sim \mathcal{P}(\lambda_i) \quad avec \quad \lambda_i = E_i \cdot \exp(F(x_i)) \tag{2}$$

Où :
— E_i est l'exposition (durée du contrat en années).
— $F(x_i)$ est la fonction de prédiction (ici, la somme des arbres de boosting).
— En passant au log, on fait apparaître l'offset : $\ln(\lambda_i) = \ln(E_i) + F(x_i)$.

A.2 Fonction de Perte et Métriques

Déviance de Poisson La métrique principale utilisée pour évaluer la qualité de l'ajustement et entraîner le modèle (Loss Function) est la Déviance de Poisson. Pour un ensemble de n observations, elle est définie par :

$$D = 2\sum_{i=1}^{n}\left[y_i \ln\left(\frac{y_i}{\hat{y}_i}\right) - (y_i - \hat{y}_i)\right] \tag{3}$$

Note : Si $y_i = 0$, le terme $y_i \ln(y_i/\hat{y}_i)$ est nul par convention.

Coefficient de Gini (Normalisé) Le coefficient de Gini mesure le pouvoir discriminant du modèle. Il est dérivé de la courbe de Lorenz $L(p)$, qui représente la proportion cumulée des sinistres observés par rapport à la proportion cumulée des sinistres prédits (triés du plus bas au plus haut risque).

$$Gini = \frac{A}{A+B} = 2\int_0^1 L(p)dp - 1 \tag{4}$$

Le Gini normalisé est le rapport entre le Gini du modèle et le Gini du "modèle oracle" (prédiction parfaite).

A.3 Comparaison des Encodages de Variables Catégorielles

Soit x_k une variable catégorielle (ex : Région) et Y la cible.

Standard Target Encoding (Smoothed) L'encodage classique (benchmark) pour une catégorie k est calculé sur l'ensemble des données (introduisant un biais) :

$$\hat{x}_k = \frac{\sum_{j=1}^{N} I_{\{x_j=k\}} \cdot y_j + \alpha \cdot \bar{y}_{global}}{\sum_{j=1}^{N} I_{\{x_j=k\}} + \alpha} \tag{5}$$

Où α est un paramètre de lissage pour gérer les catégories rares.

CatBoost Ordered Target Statistic (OTS) Pour éviter la fuite de données, CatBoost utilise une permutation aléatoire σ. La statistique pour l'observation i est calculée uniquement sur le passé artificiel :

$$\hat{x}_{i,k} = \frac{\sum_{j:\sigma(j)<\sigma(i)} I_{\{x_j=k\}} \cdot y_j + a \cdot P}{\sum_{j:\sigma(j)<\sigma(i)} I_{\{x_j=k\}} + a} \tag{6}$$

Cela garantit que y_i n'est jamais utilisé pour calculer sa propre variable explicative $\hat{x}_{i,k}$.