

Motor Insurance Pricing Dashboard

Guide Technique Complet

De A a Z : architecture, modeles, metriques et monitoring

Auteur : Francois

Role : Actuarial Data Scientist

Projet : Stellantis Insurance

Stack : Python / Streamlit / GLM / XGBoost

Table des matieres

- 1.** Vue d'ensemble de l'application
- 2.** Architecture et structure des fichiers
- 3.** Les donnees : freMTPL2freq et freMTPL2sev
- 4.** Le pipeline de donnees (data_loader.py)
- 5.** Les modeles (models.py)
- 6.** Onglet 1 : Portfolio Overview
- 7.** Onglet 2 : GLM Pricing Model
- 8.** Onglet 3 : Pure Premium
- 9.** Onglet 4 : GLM vs XGBoost
- 10.** Onglet 5 : Model Monitoring
- 11.** Questions techniques frequentes

1. Vue d'ensemble de l'application

Ce que fait l'application

L'application est un **dashboard interactif** construit avec Streamlit qui simule le workflow complet d'un pricing actuary en assurance automobile :

- **Analyser** un portefeuille d'assurance auto (exploration, segmentation, poches de risque)
- **Modeliser** la frequence des sinistres avec un GLM Poisson (approche actuarielle classique)
- **Modeliser** la severite avec un GLM Gamma (cout moyen par sinistre)
- **Calculer** la prime pure = frequence x severite
- **Challenger** le GLM avec un modele Machine Learning (XGBoost) et expliquer avec SHAP
- **Monitorer** la performance des modeles dans le temps (A/E, PSI, derive)

Pourquoi ces choix technologiques ?

Choix	Justification
Streamlit	Framework Python le plus utilise en data science pour le prototypage rapide.
GLM Poisson + Gamma	Standard de l'industrie en tarification non-vie. Accepte par les regulateurs (Solvabilite II).
XGBoost	Benchmark ML le plus courant en actuariat. Objective count:poisson specifique.
SHAP	Reference en explicabilite ML. Rend un modele boite noire interpretable.
Plotly	Graphiques interactifs (hover, zoom) adaptes a un dashboard web.

2. Architecture et structure des fichiers

Fichier	Role
app.py	Application Streamlit principale (UI + 5 onglets)
data_loader.py	Telechargement, nettoyage, feature engineering
models.py	Entrainement GLM freq, GLM sev, XGBoost + metriques
requirements.txt	Dependances Python
.streamlit/config.toml	Theme et configuration Streamlit

Flux de donnees

OpenML (internet) → **data_loader.py** (telechargement + cache + nettoyage + features) → **models.py** (GLM Poisson freq, GLM Gamma sev, XGBoost, metriques) → **app.py** (visualisations Plotly, 5 onglets) → **Navigateur** (utilisateur)

Le systeme de cache Streamlit

L'application utilise 3 niveaux de cache pour eviter de recalculer a chaque interaction :

- **@st.cache_data** pour load_data() : les donnees ne sont telechargees qu'une fois
- **@st.cache_data** pour get_modeling_data() : le feature engineering est fait une fois
- **@st.cache_resource** pour run_models() : les modeles ne sont entraines qu'une fois

Quand l'utilisateur change un slider ou un filtre, seule la visualisation est recalculée, pas les modèles.

3. Les donnees : freMTPL2freq et freMTPL2sev

Les datasets **freMTPL2** sont des datasets de reference en actuariat, provenant du package R CASdatasets. Ils contiennent des donnees reelles anonymisees de Responsabilite Civile automobile en France.

freMTPL2freq (fréquence) : environ 670 000 polices

Variable	Type	Description
IDpol	ID	Identifiant unique de la police
ClaimNb	Entier	Nombre de sinistres sur la periode
Exposure	Reel [0,1]	Duree d'exposition en annees (ex: 0.5 = 6 mois)
VehPower	Entier	Categorie de puissance du vehicule (4 a 15)
VehAge	Entier	Age du vehicule en annees
DrivAge	Entier	Age du conducteur
BonusMalus	Entier	Coefficient bonus-malus (100=neutre, <100=bon, >100=mauvais)
VehBrand	Categoriel	Marque du vehicule (anonymisee : B1 a B14)
VehGas	Categoriel	Type de carburant (Regular / Diesel)
Area	Categoriel	Zone de densite (A=rural a F=tres urbain)
Density	Entier	Densite de population (hab/km2) de la commune
Region	Categoriel	Region administrative (R11 a R94)

freMTPL2sev (severite) : environ 26 000 sinistres

Variable	Type	Description
IDpol	ID	Identifiant de la police (lien avec freq)
ClaimAmount	Reel	Montant du sinistre (en euros)

Une police peut avoir plusieurs lignes dans sev (plusieurs sinistres). C'est pourquoi on fait un groupby(IDpol) pour agreger le cout total et le cout moyen.

4. Le pipeline de donnees (data_loader.py)

4.1 Telechargement

Le fichier utilise **requests** (et non fetch_openml de sklearn) pour eviter les problemes SSL sur macOS. Les donnees sont cachees localement dans `~/.cache/pricing_dashboard/` apres le premier telechargement. Le systeme essaie 3 sources : API OpenML, GitHub raw (miroir CSV), puis fetch_openml en dernier recours.

4.2 Nettoyage actuarial

```
df = df[df['Exposure'].between(0.01, 1.0)] # Expositions valides
df = df[df['BonusMalus'].between(50, 230)] # Plage reglementaire du CRM
```

- **Exposure** : une exposition de 0 ou > 1 est incoherente pour une modelisation annuelle.
- **BonusMalus** : le CRM francais va de 50 a 350, mais au-delà de 230 c'est souvent des erreurs.

4.3 Capping de la severite

```
sev_threshold = df.loc[mask, 'TotalClaimAmount'].quantile(0.995)
df['TotalClaimAmount'] = df['TotalClaimAmount'].clip(upper=sev_threshold)
```

On plafonne au 99.5e percentile pour eviter que les sinistres extremes ne deforment le modele. En production, ces sinistres graves sont traites separement (reassurance, provisions individuelles).

4.4 Feature engineering

Feature	Formule	Pourquoi
Frequency	ClaimNb / Exposure	Variable cible du modele de frequence
PurePremium	TotalClaimAmount / Exposure	Variable cible pour le monitoring
Severity	TotalClaimAmount / ClaimNb	Variable cible du modele de severite
LogDensity	log(1 + Density)	Linearise la densite (tres asymetrique) pour le GLM
DrivAge_bin, etc.	pd.cut()	Bins pour la visualisation et le monitoring

4.5 Encodage pour la modelisation

```
pd.get_dummies(df, columns=cat_cols, drop_first=True)
```

drop_first=True est essentiel : pour eviter la multicolinearite dans le GLM, on supprime le premier niveau de chaque variable categorielle qui devient la **reference**. Exemple : Area_A est la reference, toutes les relativites d'Area sont par rapport a A.

5. Les modeles (models.py)

5.1 GLM Poisson : modele de frequence

```
glm = sm.GLM(
    y_train, # Frequence
    X_train_c, # Features + constante
    family=Poisson(link=Log()), # Distribution Poisson, lien log
    freq_weights=w_train # Ponderation par l'exposition
)
```

Pourquoi Poisson ?

La distribution de Poisson modelise le **nombre d'evenements** dans un intervalle. C'est le choix naturel pour les sinistres : evenements discrets (0, 1, 2...), relativement independants, et la variance est proportionnelle a la moyenne.

Pourquoi le lien Log ?

Le lien log garantit des predictions toujours positives et rend le modele **multiplicatif** :

```
log(frequence) = b0 + b1*VehPower + b2*DrivAge + ...
frequence = exp(b0) * exp(b1*VehPower) * exp(b2*DrivAge) * ...
```

Chaque $\exp(b_i)$ est une **relativite** : le facteur multiplicatif sur la frequence de base.

Pourquoi freq_weights = Exposure ?

L'exposition est la duree d'observation. Un assure observe 6 mois ($exposure=0.5$) apporte moins d'information qu'un assure observe 12 mois ($exposure=1.0$). Les poids ajustent l'importance de chaque observation proportionnellement.

5.2 GLM Gamma : modele de severite

```
glm_sev = sm.GLM(
    y_train, # Severite (cout moyen / sinistre)
    X_train_c,
    family=Gamma(link=Log()), # Distribution Gamma, lien log
    freq_weights=nb_claims_train # Poids = nombre de sinistres
)
```

Pourquoi Gamma ?

La distribution Gamma modelise des donnees **continues et strictement positives** avec une asymetrie a droite. C'est le profil typique des montants de sinistres : beaucoup de petits, peu de gros.

Pourquoi uniquement sur les sinistres observes ?

La frequence dit 'combien de sinistres ?'. La severite dit 'si sinistre, combien ca coute ?'. Les polices sans sinistre n'apportent aucune information sur le cout.

5.3 XGBoost : modele de frequence alternatif

Parametre	Valeur	Role
objective	count:poisson	Meme hypothese que le GLM pour comparaison equitable
max_depth	5	Limite la profondeur pour eviter l'overfitting
min_child_weight	50	Au moins 50 observations par feuille (credibilite)
reg_alpha / reg_lambda	1.0 / 5.0	Regularisation agressive pour la stabilite
n_estimators	300	Nombre d'arbres dans l'ensemble
learning_rate	0.05	Pas d'apprentissage conservateur

En production, ces hyperparametres seraient optimises par cross-validation (GridSearch ou Optuna).

5.4 Metriques d'évaluation

Coefficient de Gini

La metrique de reference en tarification non-vie. Mesure la capacite du modele a **discriminer** les bons risques des mauvais. Gini=0 : pas de discrimination. Gini=1 : parfait. En frequenc auto, un Gini de 0.05 a 0.15 est considere comme correct. Les sinistres sont des evenements rares et aleatoires : meme le meilleur modele ne peut pas predire individuellement.

Lift curves

On trie les assures en deciles de prediction (D1 = moins risques, D10 = plus risques) et on compare la frequenc reelle vs predite par decile. Un bon modele montre une frequenc croissante de D1 a D10.

Double lift

On trie par le ratio XGBoost/GLM. Ca montre les segments ou les deux modeles divergent le plus. Si le reel est plus proche de XGBoost, cela signifie que le XGBoost capture des patterns non-lineaires.

6. Onglet 1 : Portfolio Overview

Objectif : comprendre la composition et la structure de risque du portefeuille avant modelisation.

- **KPIs** : nombre de polices, exposition totale, sinistres, frequence moyenne, prime pure moyenne
- **Exposure Distribution** : ou se concentre le volume du portefeuille
- **Claim Frequency by Segment** : identification des segments sur/sous-sinistrant vs moyenne
- **Risk Heatmap** (DrivAge x VehPower) : detecte les combinaisons les plus risquees (jeunes + puissants)
- **Bonus-Malus Distribution** : la majorite des assurés sont au bonus maximum (50-60)

7. Onglet 2 : GLM Pricing Model

Relativites

Le coeur de la tarification. Chaque $\exp(b_i)$ est une relativité : le facteur multiplicatif sur la fréquence de base. Relativité = 1.05 pour DrivAge signifie : une année d'âge de plus multiplie la fréquence par 1.05 (soit +5%), toutes choses égales par ailleurs.

Simulateur de prime

L'utilisateur choisit un profil (âge, véhicule, BM, zone...) et le modèle prédit : (1) Fréquence via GLM Poisson, (2) Sévérité via GLM Gamma, (3) Prime pure = Fréquence x Sévérité. La comparaison avec la moyenne portefeuille donne le positionnement risque.

8. Onglet 3 : Pure Premium

La formule fondamentale

$$\text{Pure Premium} = E[\text{Fréquence}] \times E[\text{Sévérité} \mid \text{sinistre}]$$

C'est la base de TOUTE tarification non-vie. La prime pure est le montant minimum pour couvrir les sinistres attendus, **avant** les frais de gestion, la marge de profit, et la réassurance.

Decomposition par segment

Le graphique en 3 colonnes montre pour chaque segment la fréquence, la sévérité, et la prime pure. Deux segments peuvent avoir la même prime pure pour des raisons différentes : jeunes conducteurs = fréquence élevée / sévérité modérée ; véhicules puissants = inverse.

Heatmap Pure Premium

Le croisement DrivAge x BonusMalus montre les combinaisons les plus couteuses. Outil classique de pricing pour identifier les cellules tarifaires à ajuster.

9. Onglet 4 : GLM vs XGBoost

Pourquoi comparer ?

En actuariat moderne, le GLM reste le standard réglementaire, mais le ML sert de **challenger model** pour : (1) détecter des patterns non-linéaires, (2) quantifier le gap de performance, (3) guider l'ajout d'interactions dans le GLM.

SHAP (SHapley Additive exPlanations)

Vient de la théorie des jeux (valeurs de Shapley). Pour chaque prediction, SHAP calcule la contribution de chaque feature. **Mean |SHAP|** = importance moyenne absolue. **Dependence plot** = comment la contribution d'une feature change selon sa valeur.

10. Onglet 5 : Model Monitoring

A/E Ratio (Actual / Expected)

A/E = Fréquence observée / Fréquence prédictive

- A/E = 1.0 : le modèle prédit parfaitement
- A/E > 1.05 : sous-estimation du risque = sous-tarification
- A/E < 0.95 : surestimation = sur-tarification (perte de compétitivité)

PSI (Population Stability Index)

Mesure la dérive de la distribution des predictions entre deux périodes.

- PSI < 0.10 : pas de dérive significative
- 0.10 ≤ PSI < 0.25 : dérive modérée, à investiguer
- PSI ≥ 0.25 : dérive significative, recalibrer le modèle

Analyse des résidus et A/E par segment

Les résidus (Actual - Predicted) doivent être centrés autour de 0 sans pattern. L'A/E par segment vérifie que le modèle est bon partout, pas seulement en moyenne. Un A/E global à 1.0 peut masquer un segment à 0.8 et un autre à 1.2 qui se compensent.

11. Questions techniques fréquentes

Pourquoi ne pas utiliser uniquement XGBoost ?

- **Reglementaire** : Solvabilité II exige des modèles interprétables et validables.
- **Stabilité** : le GLM est plus stable dans le temps (moins de sur-ajustement).
- **Actuarial sign-off** : l'actuaire certifie doit pouvoir valider et signer les tarifs.

Le XGBoost sert de challenger : il indique où le GLM peut être amélioré.

Pourquoi modéliser fréquence et sévérité séparément ?

- Les **drivers** sont souvent différents (le BM impacte la fréquence, peu la sévérité).
- Les **distributions** sont différentes (Poisson pour comptages, Gamma pour montants).
- Plus de **contrôle** : on peut recalibrer l'un sans toucher l'autre.

Comment passer de la prime pure à la prime commerciale ?

```
Prime commerciale = Prime pure x (1+frais) x (1+marge) x (1+reassurance)
```

Typiquement : Prime pure x 1.30 à 1.50 selon les assureurs.

Que ferait-on de différent en production ?

- Données internes de l'assureur, pas un dataset public
- Variables UBI (télématique), historique sinistres individuel
- Optimisation des hyperparamètres par cross-validation (Optuna)
- Validation out-of-time (pas juste train/test random)
- Pipeline MLOps : re entraînement automatique, monitoring temps réel
- Gouvernance : documentation complète, validation actuarielle, comité de tarification

Pourquoi LogDensity plutôt que Density brute ?

La densité est très asymétrique (la plupart des communes ont une faible densité, quelques-unes comme Paris ont >20 000 hab/km²). Le log transforme en distribution plus symétrique, ce qui aide le GLM qui suppose une relation linéaire en log.

Le Gini est faible (0.05-0.15), est-ce normal ?

Oui ! En fréquence auto, c'est normal car les sinistres sont des événements rares et aléatoires. La valeur du modèle est dans sa capacité à segmenter correctement les populations à risque élevé vs faible, pas à prédire individuellement.