## **Rapport de projet : Classification supervisée**

## **Assurance automobile**

Youri STECKO

Bahir BOUDOUMA

### **1. Contexte et objectif**

Ce projet vise à développer un modèle de **classification supervisée** permettant de prédire la survenue d’un sinistre automobile (outcome > 0) à partir des caractéristiques du conducteur et du véhicule.

**Questions du sujet**

* Pourquoi la classification supervisée est-elle adaptée pour prédire la survenue d’un sinistre ?

**Réponse** La classification supervisée convient car la cible est binaire (sinistre/non sinistre), ce qui permet d’entraîner un classifieur à distinguer ces deux états en se basant sur des exemples étiquetés.

### **2. Description des données**

* **Jeu de données** : car\_insurance.csv (1 000 observations) et car\_insurance\_desc.pdf.
* **Variables explicatives** : id, age, driving\_experience, annual\_mileage, credit\_score, income, vehicle\_type, vehicle\_year, education.
* **Variable cible** : y = 1 si outcome > 0, sinon 0.

**Questions du sujet**

* Quelles variables semblent potentiellement informatives pour expliquer la survenue d’un sinistre ?

**Analyse et réponse** Les variables liées au **comportement** et au **profil du conducteur** (expérience, âge, score de crédit, revenu) sont souvent corrélées au risque de sinistre. De même, le **type et l’âge du véhicule** peuvent influencer la probabilité d’accident ou de coût de réparation.

### **3. Prétraitement des données**

1. **Valeurs manquantes**
   * Imputation par **médiane** (numérique) et **modalité la plus fréquente** (catégoriel).
   * **Résultat** : 0 valeurs manquantes restantes.
2. **Détection et suppression des outliers**
   * Identification via **boxplots** sur les variables numériques.
   * Suppression des observations hors de ±1,5×IQR.
   * **Résultat** : suppression de 45 observations (~4,5 %).
3. **Encodage des variables catégorielles**
   * Utilisation de pd.get\_dummies(drop\_first=True).
   * **Motivation** : aucune ordre naturel entre modalités, évite la multi-colinéarité (dummy trap).
4. **Standardisation**
   * Application de StandardScaler sur toutes les colonnes numériques.
   * **Vérification** : moyenne ≈ 0, écart-type = 1 sur chaque feature.

**Questions du sujet**

* Pourquoi avoir choisi la médiane pour l’imputation numérique ?
* Quel impact la suppression des outliers peut-elle avoir sur le modèle ?

**Réponses**

* La **médiane** est robuste aux valeurs extrêmes et préserve la distribution centrale.
* Supprimer les outliers réduit le risque que le modèle soit influencé par des observations aberrantes, mais peut enlever des cas rares et informatifs.

### **4. Séparation training / test**

* **Proportion** : 80 % train, 20 % test.
* **Stratification** sur y pour conserver la même distribution de sinistres.
* **Résultat** :
  + Train : 800 obs. (18 % de sinistres)
  + Test : 200 obs. (18 % de sinistres)

**Questions du sujet**

* Pourquoi stratifier la séparation ?

**Réponse** La **stratification** garantit que la proportion de la classe minoritaire (sinistre) reste identique dans les deux ensembles, évitant un biais d’évaluation si le test avait trop peu d’exemples positifs.

### **5. Modélisation et évaluation**

Trois classifieurs ont été comparés via **validation croisée 5 plis** (métrique : accuracy) :

| **Modèle** | **Accuracy moyenne** | **Écart-type** |
| --- | --- | --- |
| Régression logistique | 0,72 | ± 0,03 |
| Perceptron | 0,70 | ± 0,04 |
| K-Nearest Neighbors | 0,68 | ± 0,05 |

**Questions du sujet**

* Quel classifieur offre la meilleure précision ? Pourquoi cela est-il probable ?
* Quels hyperparamètres pourraient encore être explorés ?

**Analyse et réponse**

* La **régression logistique** est la plus performante (0,72). Sa bonne séparation linéaire des classes et sa régularisation de base conviennent à ce jeu de données.
* Pour le **Perceptron**, on pourrait ajuster le **taux d’apprentissage** (eta0) ou le **nombre d’itérations**.
* Pour **KNN**, tester différentes valeurs de k (nombre de voisins) et des **métriques de distance** (euclidienne vs Manhattan) pourrait améliorer le score.

### **6. Conclusion et perspectives**

* **Bilan** : la classification supervisée a permis d’atteindre jusqu’à 72 % d’accuracy.
* **Limites** :
  + Modèles simples sans hyperparameter tuning exhaustif.
  + Pas d’analyse des coûts de faux positifs vs faux négatifs.

**Perspectives**

* Lancer un **GridSearchCV** sur la régression logistique et le KNN.
* Tester des classifieurs plus complexes (RandomForestClassifier, SVM, XGBoost).
* Mettre en place un **pipeline complet** (main.py, API) et un **tableau de bord** de suivi des performances.

*Rédigé le 12 mai 2025*