

Challenge par Egis

"Prédisez les infractions de stationnement !"

Soutenance de projet — Génération de Données en IA par
Transport et Débruitage

Colin COËRCHON & Chloé HABASQUE

ENS Paris-Saclay

19 mars 2025

Informations

- Professeur : **Stéphane Mallat**
- Président du Jury : **Enzo Dalby**
- Providers : **Daniel Raybaut** et **Raphael Vienne**

école
normale
supérieure
paris-saclay

 MATHÉMATIQUES
VISION
APPRENTISSAGE

 egis
datacraft*

Sommaire

- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale
- 4 Modélisation
- 5 Autres idées

- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale
- 4 Modélisation
- 5 Autres idées

Contexte et objectif

Contexte

- Contrôle du stationnement payant en ville
- Connaître les zones avec le plus fort taux de contravention.
- Optimisation des contrôles

Objectif du Challenge

Le but est de proposer un modèle de **régression** permettant de **prédire le taux d'infraction** futur en fonction de données

- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale
- 4 Modélisation
- 5 Autres idées

Données à disposition

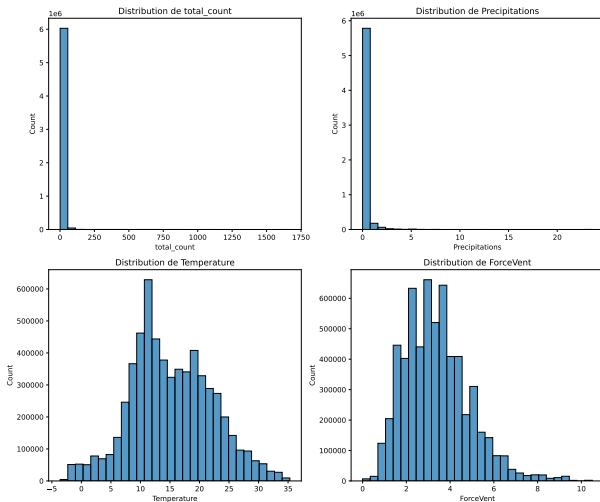
- **Taux d'infraction constaté (Variable cible)**
- **Nombre total de véhicules contrôlés dans la zone**
- **Latitude et longitude normalisées**
- **Horodatage** : Mois (de 1 à 12), jour de la semaine (de 1 à 6) et heure du contrôle (de 6 à 19).
- **Données météorologiques** : Les précipitations, la hauteur de neige, la température et la force du vent.

Données à disposition

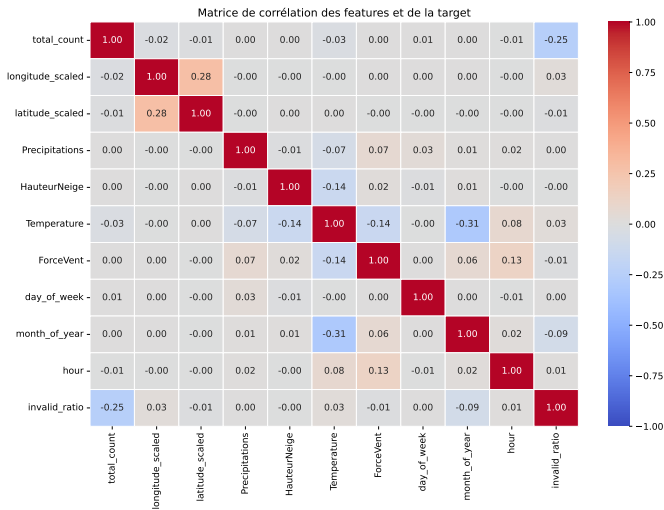
	Données	Valeurs manquantes	Variables
X_{train}	6 076 546	170 098	10
Y_{train}	6 076 546	0	1
X_{test}	2 028 750	7 906	10

Table – Résumé des caractéristiques des données utilisées

Quelques histogrammes



Corrélation



Prétraitement des données (1/2)

- **One-hot encoding** des variables temporelles.
 - ↪ Exemple : transformation des mois en saisons : Winter, Spring, Summer, Fall
- **Nettoyage des valeurs manquantes.**
 - ↪ HauteurNeige (2.7%) et ForceVent (0.1%)
 - ↪ Toutes les valeurs manquantes remplacées par 0.

Variable	Valeurs manquantes	Pourcentage
HauteurNeige	164016	2.7
ForceVent	6082	0.1

Prétraitement des données (2/2)

- **Filtrage du nombre de voitures contrôlées.**

- ↪ Seuil 45 : Excellente performance en validation, mais mauvaise généralisation.

- ↪ Seuil 10 ou 0 : Performance moyenne mais cohérente en validation et test.

- **Normalisation des variables.**

- ↪ Transformation logarithmique sur `total_count`.

- ↪ Normalisation standard de `Temperature` et `ForceVent`.

Analyse de données de position mise à l'échelle

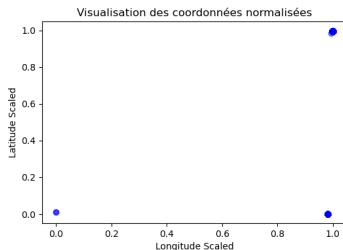


Figure – Position des points initiaux

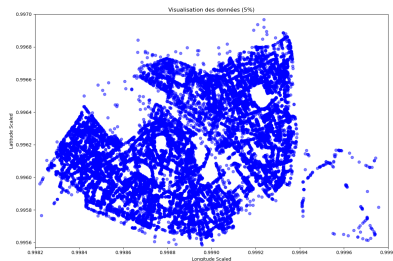
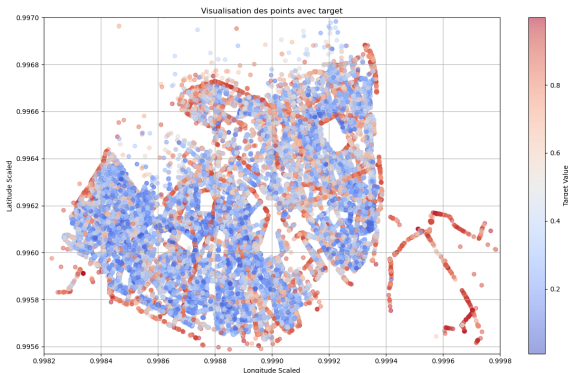


Figure – Zoom sur les données de position retenues

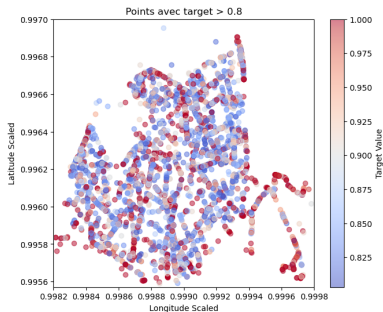
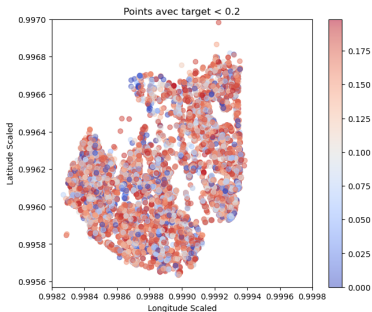
- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale
 - Analyse en amont
 - Heuristique 1
 - Heuristique 2
 - Heuristique 3
- 4 Modélisation
- 5 Autres idées

Préparation des données pour le clustering

- Prise en compte des quelques points isolés (loin de Paris)
- Conservation du reste sous le nom de `clean_data`.



- Lorsque la *target* est faible (< 0.2), les points semblent assez diffus.
- Quand la *target* est élevée (> 0.8), on observe de nombreux « segments » de points (rues).
- La *bordure* du bloc principal est très rouge (cible élevée).
- Au sud-est (ex. bois de Vincennes), la *target* est > 0.85 .



- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale**
 - Analyse en amont
 - Heuristique 1**
 - Heuristique 2
 - Heuristique 3
- 4 Modélisation
- 5 Autres idées

Clustering simple : K-means

- 1 Extraire un **échantillon** à 10% de `clean_data`.
- 2 Appliquer un **K-means** (nombre de clusters à fixer) sur cet échantillon.
- 3 Entraîner un `nn-model` (1 voisin) pour *propager* les étiquettes de cluster à la totalité de `clean_data`.
- 4 Idem pour `x_test`, où chaque point se voit attribuer le cluster de son plus proche voisin dans l'échantillon.

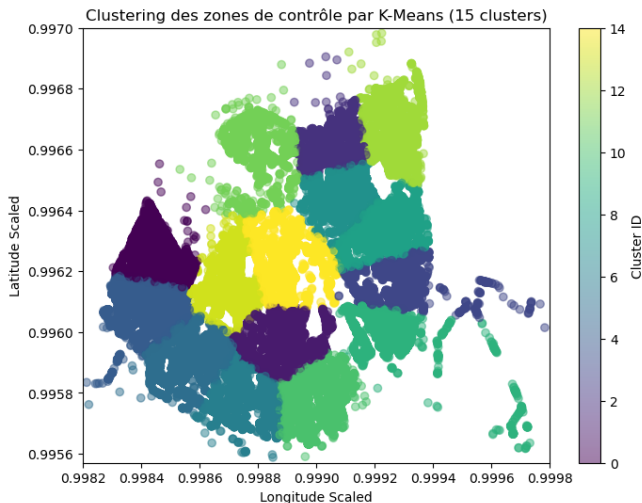
Clustering simple : K-means

- 1 Extraire un **échantillon** à 10% de `clean_data`.
- 2 Appliquer un **K-means** (nombre de clusters à fixer) sur cet échantillon.
- 3 Entraîner un `nn-model` (1 voisin) pour *propager* les étiquettes de cluster à la totalité de `clean_data`.
- 4 Idem pour `x_test`, où chaque point se voit attribuer le cluster de son plus proche voisin dans l'échantillon.

Remarque

On obtient ainsi un partitionnement purement « K-means ». Le choix de n_{clusters} (ex. 15) permet de retrouver une structure des « arrondissements » de Paris, mais sans exploiter la notion de « rue ».

Résultat



- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale**
 - Analyse en amont
 - Heuristique 1
 - Heuristique 2**
 - Heuristique 3
- 4 Modélisation
- 5 Autres idées

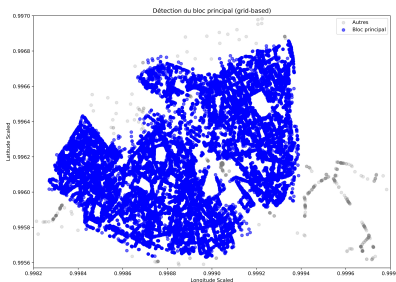
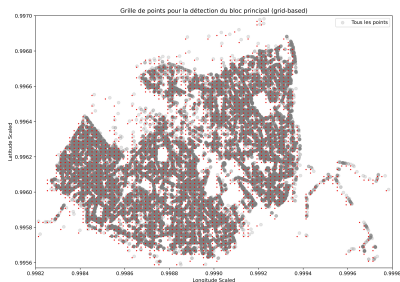
Heuristique 2 : Bordure + K-means + Nord/Sud

- ❶ **Bloc principal** via `find_main_block_grid` (approche *grid-based*)
→ isole la plus grosse composante.
- ❷ **Enveloppe convexe & bordure** : calculer `ConvexHull` → points proches de la frontière = bordure.
- ❸ **K-means** sur l'intérieur du bloc :
`inside = bloc_principal - bordure`.
- ❹ **Séparation Nord/Sud** arbitraire à
`latitude_scaled = 0.99625`.
- ❺ **Propagation 1-NN** pour étiqueter tout `clean_data` & `x_test`.

Exemple : bloc principal & bordure

`main_block` via `find_main_block_grid`

La fonction parcourt des cellules régulières (`cell_size`) dans la bounding-box, forme un graphe entre cellules voisines, et retient la plus grande composante connexe.

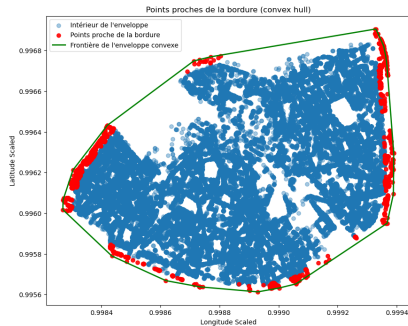


Exemple : enveloppe convexe

ConvexHull

On calcule l'enveloppe convexe, puis on repère les points dont la distance à la *boundary* du polygone est $< \text{seuil}$ (ex. $1e-4$).

↪ La bordure.



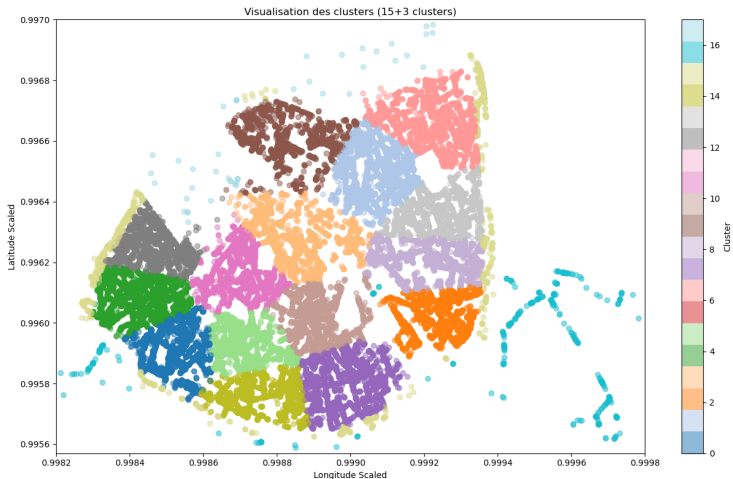
Séparation Nord/Sud & `isolated_data`

- On regroupe le bas-sud (ex. bois de Vincennes) en un cluster particulier, car $target \approx 0.85$ y est très forte.
- Les `isolated_data` (hors zone $[0.9, 1] \times [0.9, 1]$) sont aussi incorporées dans ce cluster sud (en raison de leur *target* ≈ 0.998).

Résultat final

On obtient un clustering plus "géographique", isolant la couronne (très rouge), les points internes (K-means) et les régions au sud / hors bloc.

Résultat final

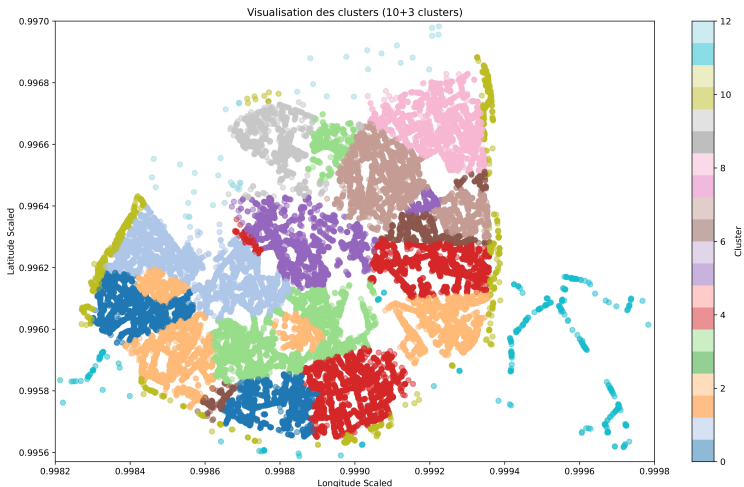


- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale**
 - Analyse en amont
 - Heuristique 1
 - Heuristique 2
 - **Heuristique 3**
- 4 Modélisation
- 5 Autres idées

Heuristique 3 : Variante (fusion par *target*)

- On reprend l'heuristique 2 : bloc principal, bordure, K-means sur l'intérieur, nord/sud.
- Sauf qu'on utilise un **K-means** plus fin (ex. 50 clusters sur l'intérieur).
- Puis on **fusionne** ces nombreux clusters selon leur *target* moyenne : par exemple en 10 bins (déciles) de *target*.
- Finalement, on propage (1-NN) sur `clean_data` et `x_test`.

Résultat



Comparaison des heuristiques

Heuristique 1

K-means simple. Rapide, standard, mais ne tient pas compte de la « bordure rouge » ni des motifs de rue.

Heuristique 2

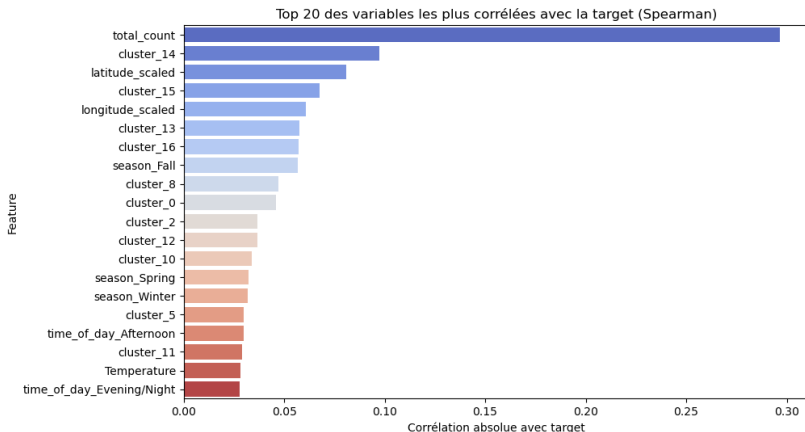
Exploite la *géographie* (bloc principal, enveloppe convexe, nord/sud). Mieux adaptée à la répartition spatiale de la *target*.

Heuristique 3

Identique à la 2, *plus* un **raffinement** : on unifie des clusters par rang de *target*.

↔ Performance qui s'avère finalement décevante...

Corrélations finales



- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale
- 4 Modélisation**
- 5 Autres idées

Métrique d'évaluation

Corrélation de Spearman

La corrélation de Spearman, notée ρ , mesure la relation entre deux variables à l'aide du rang.

Le rang est la position relative d'une donnée dans le dataset lorsqu'on trie les valeurs par ordre croissant.

Sa formule est donnée par :

$$r_s = \frac{\text{cov}(\text{rg}_X, \text{rg}_Y)}{\sigma_{\text{rg}_X} \sigma_{\text{rg}_Y}}$$

Modèles

Les différents modèles utilisés :

- Ridge et Lasso
- SVM
- RandomForest
- XGBoost

Modèle finale

Dataset publique

Avec **XGBoost optimisé**, nous avons atteint une **corrélation de 0.492** sur le dataset public, avec les hyperparamètres suivant :

- **subsample** : 1.0
- **n_estimators** : 300
- **max_depth** : 12
- **learning_rate** : 0.05
- **lambda** : 1, **gamma** : 0
- **colsample_bytree** : 1.0, **alpha** : 0

Modèle finale

Dataset privé

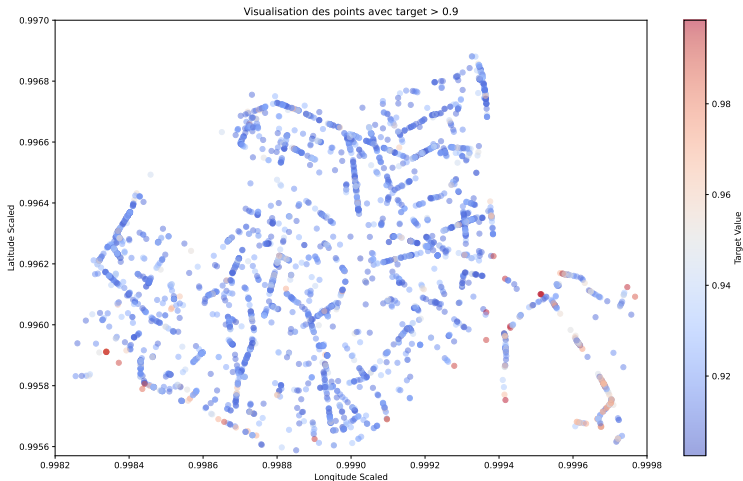
Le résultat du dataset privé a été entraîné avec une variante légèrement différente des hyperparamètres de XGBoost :

- **subsample** : 0.8, **n_estimators** : 800, **max_depth** : 15
- **learning_rate** : 0.03, **lambda** : 1, **gamma** : 0
- **colsample_bytree** : 0.8, **alpha** : 0

Avec ce modèle, on a obtenu une **corrélacion de 0.432** sur **l'ensemble privé**, contre 0,4329 sur l'ensemble publique, confirmant l'absence de sur-apprentissage.

- 1 Introduction
- 2 Analyse des données
- 3 Clustering spatiale
- 4 Modélisation
- 5 Autres idées**

Des « rues » ?



RANSAC

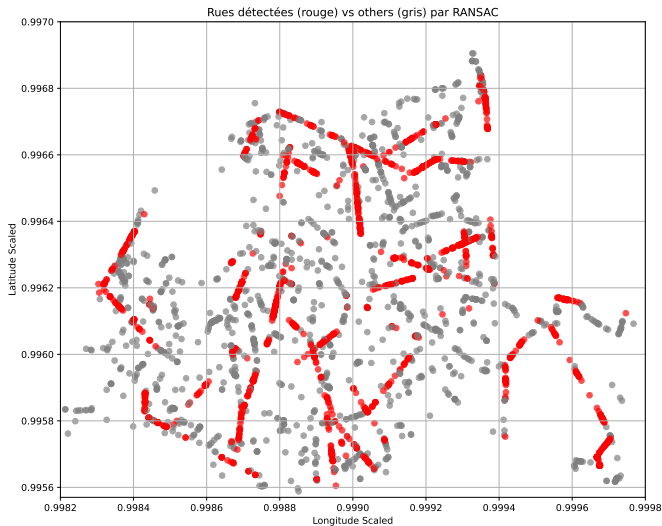
Comment exploiter cette structure particulière pour les données de valeur cible importante ?

RANSAC

Comment exploiter cette structure particulière pour les données de valeur cible importante ?

↪ Utiliser un algorithme de détection de lignes comme **RANSAC** !

Résultats



Problème

La détection de lignes fonctionne bien lorsqu'on **filtre** les données avec une target élevée, sinon, c'est vraiment difficile...

Solutions ?

- Laisser tomber l'idée.

Problème

La détection de lignes fonctionne bien lorsqu'on **filtre** les données avec une target élevée, sinon, c'est vraiment difficile...

Solutions ?

- Laisser tomber l'idée.

C'est une solution, mais nous sommes convaincus qu'on peut persévérer !

Problème

La détection de lignes fonctionne bien lorsqu'on **filtre** les données avec une target élevée, sinon, c'est vraiment difficile...

Solutions ?

- Laisser tomber l'idée.

C'est une solution, mais nous sommes convaincus qu'on peut persévérer !

- Essayer de détecter dans le nuage de points complet mais échantillonné ?

Problème

La détection de lignes fonctionne bien lorsqu'on **filtre** les données avec une target élevée, sinon, c'est vraiment difficile...

Solutions ?

- Laisser tomber l'idée.

C'est une solution, mais nous sommes convaincus qu'on peut persévérer !

- Essayer de détecter dans le nuage de points complet mais échantillonné ?

Peu de chance d'arriver à une bonne détection...

Problème

La détection de lignes fonctionne bien lorsqu'on **filtre** les données avec une target élevée, sinon, c'est vraiment difficile...

Solutions ?

- Laisser tomber l'idée.

C'est une solution, mais nous sommes convaincus qu'on peut persévérer !

- Essayer de détecter dans le nuage de points complet mais échantillonné ?

Peu de chance d'arriver à une bonne détection...

- Arrêter de détecter des points, mais plutôt des **zones géographiques** !

Problème

La détection de lignes fonctionne bien lorsqu'on **filtre** les données avec une target élevée, sinon, c'est vraiment difficile...

Solutions ?

- Laisser tomber l'idée.

C'est une solution, mais nous sommes convaincus qu'on peut persévérer !

- Essayer de détecter dans le nuage de points complet mais échantillonné ?

Peu de chance d'arriver à une bonne détection...

- Arrêter de détecter des points, mais plutôt des **zones géographiques** !

Très intéressant, mais non implémenté par manque de temps...