Sorbonne Université

Rapport du projet

Interpolation

Nom, prénom et spécialité :

YUAN Fangzheng	DAC
ZHUANG Pascal	DAC
HUANG Bozhang	DAC

Tuteurs :
M. Vincent Guigue
M. Nicolas Baskiotis

Table des matières

1 Introduction

De nos jours, de plus en plus de données sont disponibles dans le transport. L'utilité d'analyser des données GPS permet de mieux comprendre la mobilité telles que les habitudes des personnes dans la journée, et permet également une meilleure compréhension des territoires comme connaître les routes qui sont généralement plus encombrées que d'autres. Nous allons alors vous présenter des stratégies d'analyses. Au cours de ce processus, certains problèmes surviennent, comme des limitations dans la récupération des données ou des dispositifs qui fonctionnent mal, ce qui entraîne des données partiellement faussées ou manquantes. L'idée est donc de développer des algorithmes prédictifs et d'interpolation sur les trajectoires GPS en se basant sur des observations du passé.

2 Data Source et Data Cleansing

2.1 Data Source

Les données sont des données GPS issues de *data.gov*, un site contenant des open data du gouvernement américain :
`https://catalog.data.gov/dataset/safety-pilot-model-deployment-data`.

Le fichier *DataGpsDas.csv* est celui dont nous utiliserons pour notre projet.

Ce fichier contient **41.021.227 enregistrements** et **17 attributs (colonnes)** :

Device	Trip	Time	GpsTime	GpsWeek
GpsHeading	GpsSpeed	Latitude	Longitude	Altitude
NumberOfSats	Differential	FixMode	Pdop	GpsBytes
UtcTime	UtcWeek			

Voici la description des colonnes dont nous utiliserons :
- **Trip :** Le numéro du trajet GPS
- **GPS Time :** Heure du GPS en millisecondes
- **GPS Heading :** La direction du GPS en degrés
- **GPS Speed :** La vitesse du GPS en mètre par seconde
- **Latitude :** La position latitude du véhicule
- **Longitude :** La position longitude du véhicule

2.2 Data Cleansing

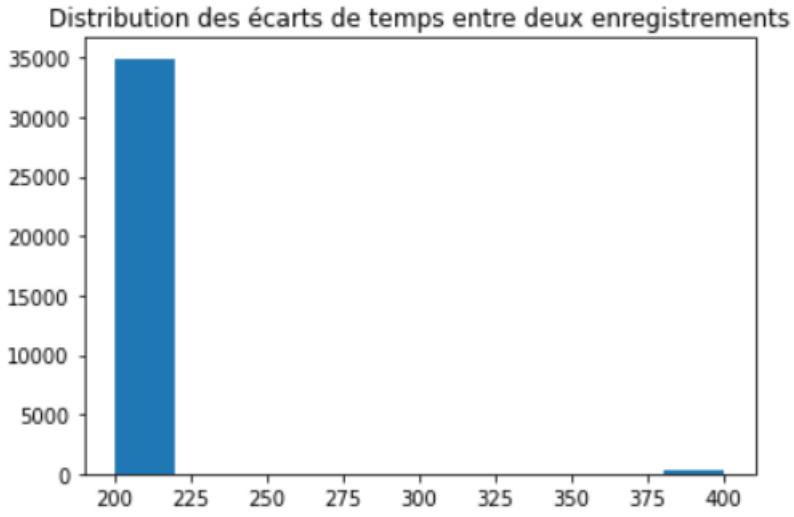
Nous avons d'abord commencé par **réduire nos données** comme dans le document *Motion ResNet : An efficient data imputation method for spatio-temporal series* fourni par nos enseignants :

1. "We took trajectories only passing in a defined perimeter centered in latitude 42.282970 and longitude -83.735390, all positions within **latitude 42.282970±0.003000** and **longitude 83.735390±0.003000** are kept."
2. "Also we keep only trajectories with **at least 100 data** points so that we have enough dynamics to learn something."

Après ce nettoyage, nous nous retrouvons avec **13 trajets GPS** avec plus ou moins d'enregistrements pour chacun d'entre-eux :

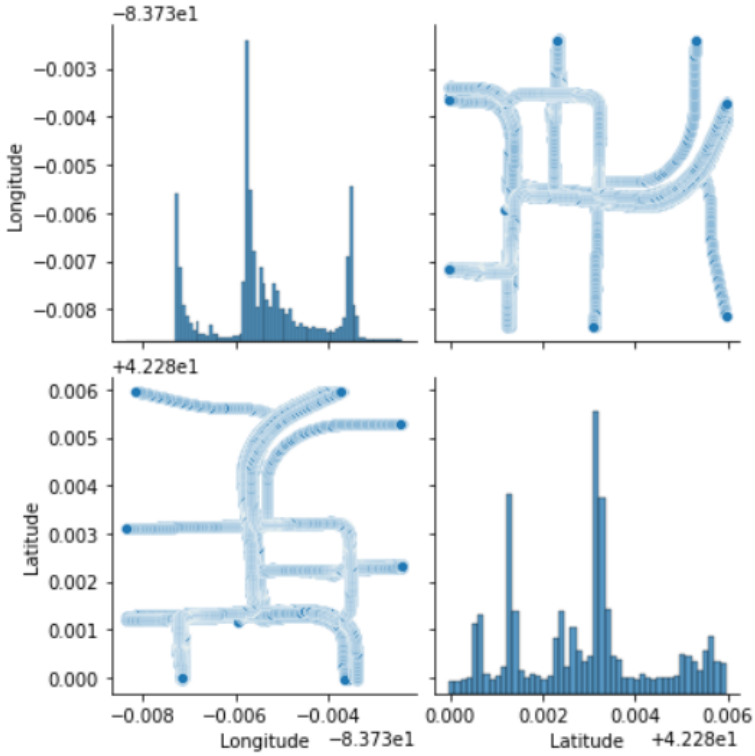
```
Trajet n°1 : 488 enregistrements
Trajet n°2 : 4094 enregistrements
Trajet n°3 : 528 enregistrements
Trajet n°4 : 3438 enregistrements
Trajet n°5 : 4932 enregistrements
Trajet n°6 : 5700 enregistrements
Trajet n°7 : 13317 enregistrements
Trajet n°8 : 649 enregistrements
Trajet n°9 : 135 enregistrements
Trajet n°10 : 1036 enregistrements
Trajet n°11 : 301 enregistrements
Trajet n°12 : 502 enregistrements
Trajet n°13 : 252 enregistrements
```

En étudiant l'**échantillonnage** de nos données, nous remarquons que pour la majorité de nos données, l'intervalle de temps entre deux enregistrements est de **200 millisecondes** et plus rarement **400 millisecondes**. D'autres écarts de temps apparaissent mais seulement une seule fois. Nous choisissons alors de ne pas représenter les valeurs qui n'apparaissent qu'une fois.



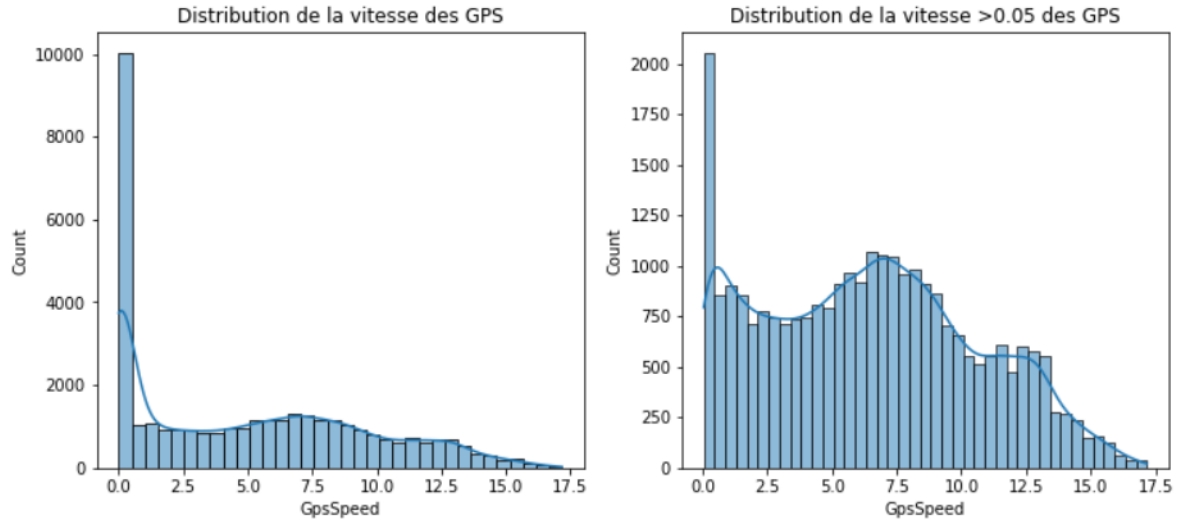
Nous pouvons ainsi considérer que les enregistrements des trajets dans nos données sont séparés de **200 millisecondes**.

Visualisation de la distribution des colonnes Longitude et Latitude :



Nous pouvons voir sur cette figure **l'allure des routes** que composent nos données. Nous pouvons observer sur les histogrammes des pics à certains niveaux qui correspondent vraisemblablement à des croisements de routes tels que les carrefours.

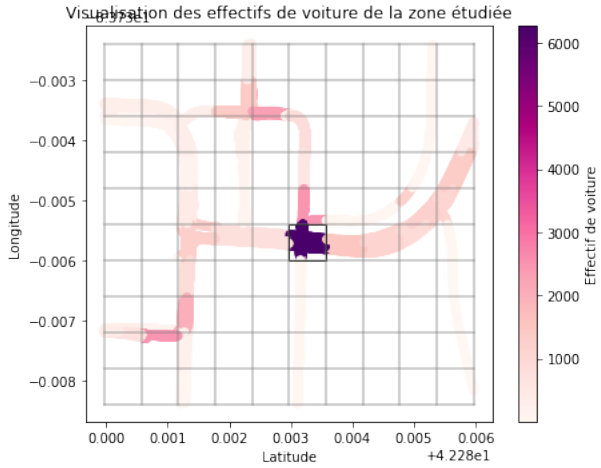
Visualisation de la distribution de la vitesse dans nos données :

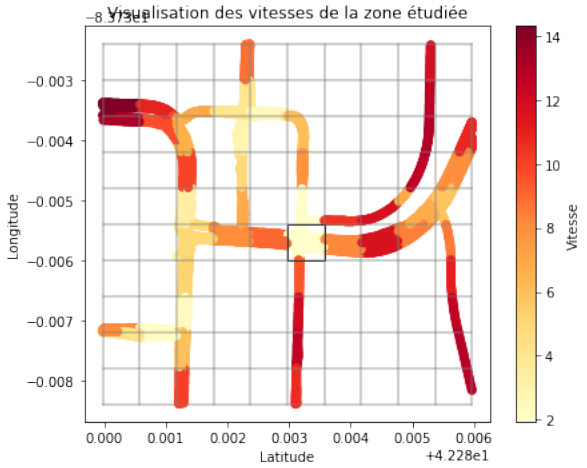


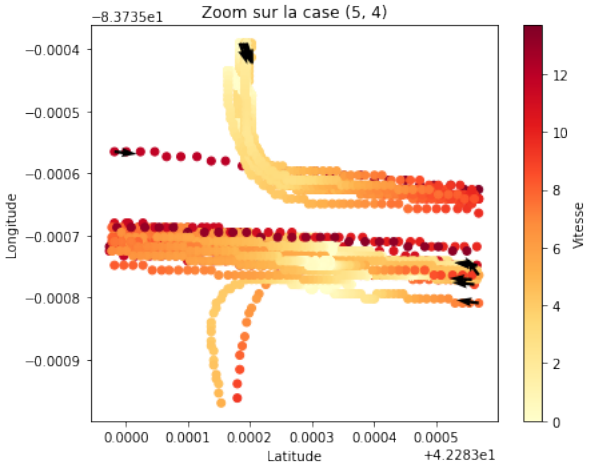
Nous avons à gauche la distribution des valeurs de la colonne correspondant aux valeurs de la **vitesse enregistrées par les GPS**. Nous constatons que beaucoup de valeurs sont à 0, c'est-à-dire que les véhicules sont souvent à l'arrêt, ce qui ne facilite pas la visualisation des autres vitesses et qui nous amène au deuxième histogramme, qui cette fois ne comprend pas les valeurs nulles. Nous observons que la plupart des **vitesses se situent entre 0 et 13** puis quelques-unes au-delà.

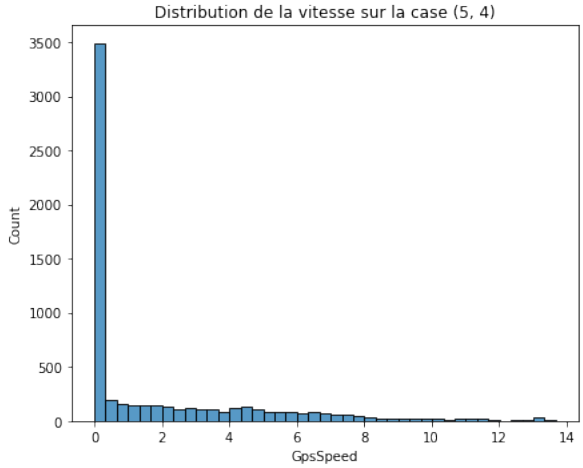
La prochaine étape était la **discrétisation de l'espace**. Pour cela, nous avons créé une fonction qui, en passant un dataset et un nombre d'intervalles, nous renvoie : le minimum et le maximum de la latitude et de la longitude, et les écarts en hauteur et en largeur de chaque case.
Nous avons ensuite pu calculer pour chaque case, l'effectif de points et la vitesse moyenne.

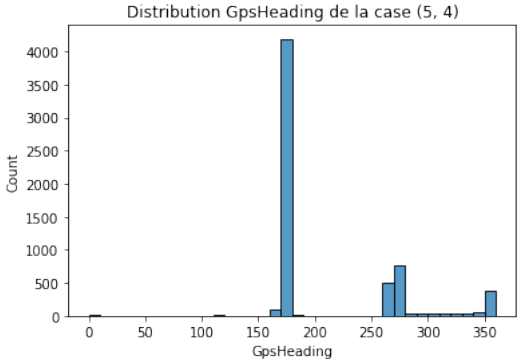
Visualisation des trajectoires GPS en fonction des effectifs et des vitesses moyennes :

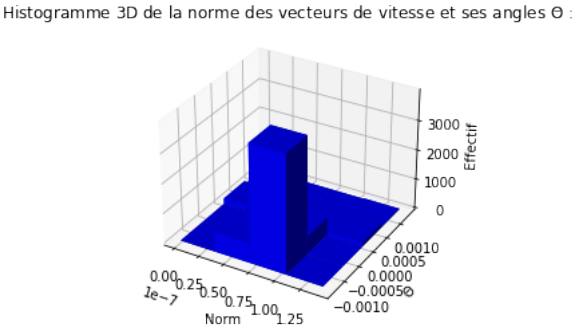












Les quatre dernières figures représentent des informations sur **une case** donnée, ici la case (5,4), encadrée en noir sur les deux premières figures. Nous pouvons remarquer que plus une case a un grand effectif de points et plus la vitesse moyenne dans cette même case est petite.
D'après nos figures, cette case correspond à plusieurs routes très proches les unes des autres et de

nombreux véhicules y passent avec une vitesse plus ou moins modérée.

Pour éviter de taper dans la **précision machine**, nous avons finalement normalisé nos données (les colonnes : *Latitude, Longitude, GpsSpeed, GpsHeading* et *GpsTime*) avec la **standardisation**, c'est-à-dire retirer la moyenne et diviser par l'écart-type.

Afin d'entraîner, de tester et d'évaluer différents modèles de Machine Learning, nous avons dans un premier temps **séparer les données en deux parties**, une partie d'apprentissage et une autre de test avec une fréquence d'échantillonnage. Nous allons ainsi considérer différentes fréquences d'échantillonnage et ces différentes fréquences d'échantillonnage peuvent correspondre à différents problématiques. Par exemple, une fréquence d'échantillonnage très faible pourrait nous amener à étudier le chemin et le temps que les gens prennent pour effecteur leur parcours, et une fréquence très élevée nous permettrait d'essayer de caractériser l'état d'esprit de l'utilisateur grâce à sa conduite. Nous nous rapprochons alors de la modélisation du comportement du conducteur. Et l'intuition serait que plus nous diminuons la fréquence d'échantillonnage et plus la prédiction des positions sera dure.
L'ordre chronologique étant important, la séparation des données se fait non pas directement sur les lignes du dataset mais sur les différents numéros de trajets. Nous prenons donc de manière aléatoire des trajets pour l'apprentissage et d'autres pour le test.

3 Axis de Modèles

3.1 Modèle Physique 1

L'idée est de faire la **prédiction** à partir du vecteur de vitesse.
Notons pour chaque trajectoire, A_i le i-ème point, t_i le GpsTime du i-ème point, et, x_i et y_i sa latitude et sa longitude. Nous voulons calculer la latitude et la longitude du point suivant A_{i+1} dans un temps futur t'.
Le calcul de A_{i+1} se fait par :

$$t_0 = t_i - t_{i-1}, \vec{v_i} = \begin{bmatrix} \frac{x_{i-1}-x_i}{t_0} \\ \frac{y_{i-1}-y_i}{t_0} \end{bmatrix}, \; A_{i+1} = \vec{v_i} \cdot t' + \begin{bmatrix} x_i \\ y_i \end{bmatrix},$$

En particulier, nous utilisons le GpsHeading pour ajuster la prédiction. Soit θ la direction moyenne dans cette région, nous avons en plus :

$$\vec{v_i} = \begin{bmatrix} \frac{x_{i-1}-x_i}{t_0} \\ \frac{y_{i-1}-y_i}{t_0} \end{bmatrix} \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}$$

3.2 Modèle Physique 2

Toujours dans le cadre du modèle physique, nous disposons cette fois des attributs GpsHeading et GpsSpeed. La formule a été donné par un expert en aviation.

$$
\begin{aligned}
distance &= speed * time \\
lat2 &= asin(sin(lat1) * cos(d) + cos(lat1) * sin(d) * cos(tc)) \\
dlon &= atan2(sin(tc) * sin(d) * cos(lat1), cos(d) - sin(lat1) * sin(lat2)) \\
lon2 &= mod(lon1 - dlon + pi, 2 * pi) - pi
\end{aligned}
$$

Référence : `http://www.edwilliams.org/avform147.htm#LL`

3.3 Linear Regression

En choisissant **Latitude, Longitude, GpsHeading, GpsTime** comme les attributs de X, nous appliquons le modèle linéaire :

$$X.W = Y$$

3.4 Knn Regression

Suite à nos observations sur les comportements des trajectoires de véhicules des différents trips, nous faisons l'hypothèse que les trajectoires proches suivent des **chemins similaires**. Nous choisissons alors d'utiliser le Knn Regression.

3.5 Long-Short Term Memory

Après avoir observé de mauvaises prédictions dans des régions compliquées comme un carrefour, nous avons essayé le modèle Long-Short Term Memory (LSTM) pour la prédiction en fonction d'**une série de temps**. L'architecture du réseau est définie avec :

1. Les données divisées en batchs. Un batch contient un certain nombre de trajectoires dont les points sont paddés avec zéros (padding) pour qu'ils atteignent la longueur maximum.
2. Des dimensions de données. Certains attributs sont potentiellement des facteurs importants pour la prédiction tels que la latitude et la longitude.
3. Un certain nombre de neurones cachés et de couches cachées.
4. La couche de sortie est un Linear qui résume les statuts cachés en 2 dimensions correspondant à la latitude et la longitude.
5. Le critère MSE et l'optimisateur Adam.

4 Train et Test

4.1 Train

Modèle Physique 1 et 2

Ces deux modèles n'ont ni paramètres ni entraînement.

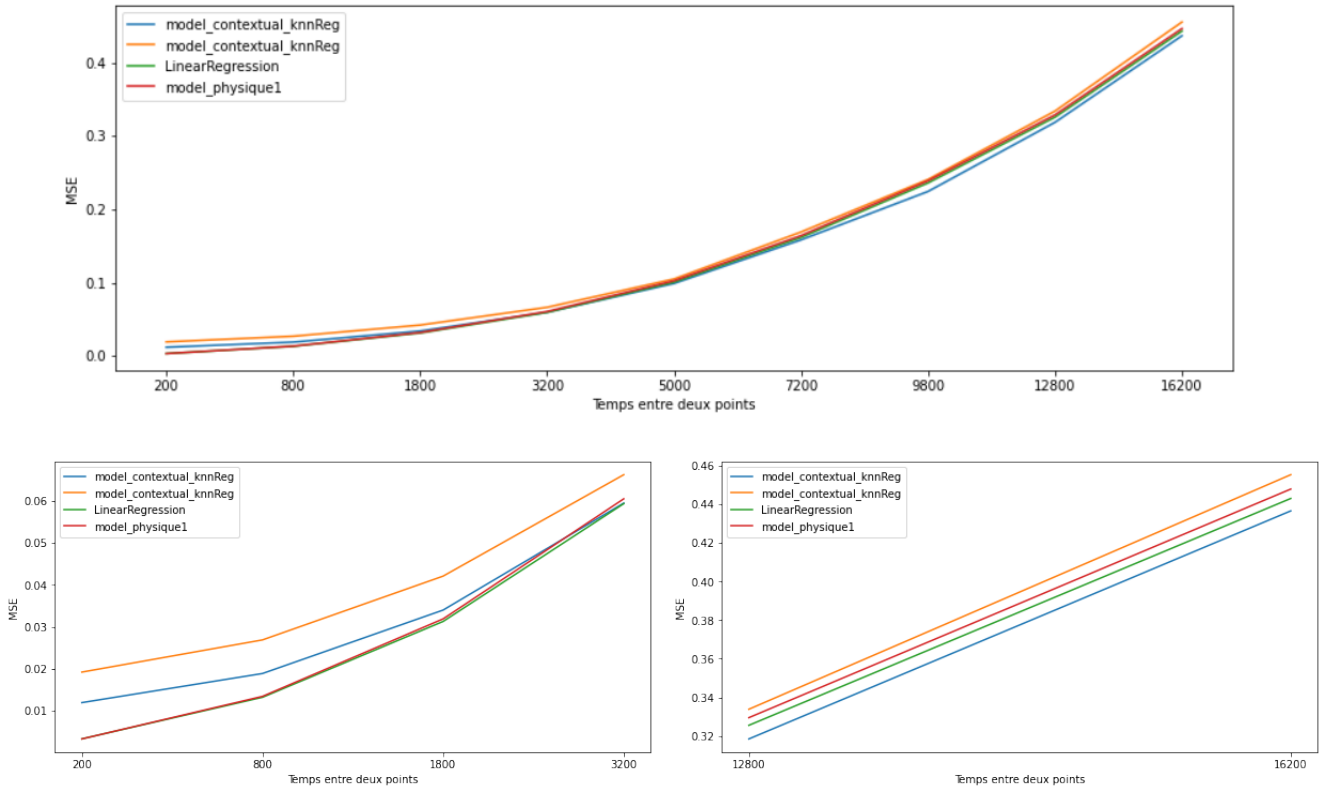
Linear Regression et Knn Regression

Les entraînements sont gérés par sklearn. En général, les modèles calculent la loss MSE et font une descente de gradient.

Long-Short Term Memory

Nous définissons une centaine d'itérations, et à chaque itération nous passons les batchs au réseau de neurones au lieu de passer toutes les données en une seule fois. Cela peut mitiger le problème de l'explosion du gradient.

4.2 Comparaisons des modèles



Nous pouvons observer que globalement sur 9 fréquences différentes, les modèles évoluent plus ou moins de la même manière.
En regardant plus précisément certaines valeurs des fréquences, nous pouvons voir qu'au départ les modèles contextuels k-NN semblent moins bons par rapport à la régression linéaire et au modèle physique. Cependant, au fur et à mesure que la fréquence augmente, l'un des modèles contextuels fait de moins en moins d'erreurs par rapport aux trois autres modèles. Ce dernier possède, contrairement à l'autre modèle k-NN, le vecteur de vitesse instantanée, ce qui peut expliquer la différence de performance.
Nous pouvons en déduire que les modèles contextuelles deviennent très pertinents lorsque la fréquence d'échantillonnage est très basse, car notre problème devient de plus en plus dur à chaque fois que la fréquence baisse et il vaudra mieux à la fin prédire à partir des voisins proches.

4.3 Performances des modèles

Nous avons mis dans la partie Annexes, pour chaque modèle :

— Deux cartes, l'une avec la visualisation de l'effectif dans chaque case, et l'autre avec la vitesse moyenne ;

— Une matrice d'erreurs du modèle dans l'espace discrétisé avec les données de test ;
— Un histogramme contenant les valeurs MSE en général ;
— Un histogramme représentant les valeurs moyennes MSE en fonction de l'effectif de véhicules dans les cases ;
— Un histogramme avec les valeurs moyennes MSE en fonction de la vitesse moyenne des véhicules dans les cases ;
— Un histogramme qui contient les valeurs moyennes MSE en fonction de la variance des vitesses dans les cases.

Contrairement à ce que nous pouvons penser, les erreurs des modèles ne proviennent pas principalement des cases où l'effectif de véhicules est haute mais proviennent plutôt de cases où il y a peu d'enregistrements et où la vitesse moyenne est assez grande.
Le deuxième modèle k-NN qui n'a pas le vecteur de vitesse instantanée se trompe beaucoup plus par rapport aux autres modèles, sur des cases où la vitesse moyenne est faible.
Nous observons des performances assez similaires entre le modèle de régression linéaire et le modèle physique.

5 Difficultés et Améliorations

5.1 Difficultés

1. Les données concernent la géographie dont nous n'avons pas de connaissance à priori.
2. Les données sont réelles et contiennent alors des bruits dont nous n'avons pas forcément assez d'expérience pour les gérer de la meilleure façon. Des fois, les résultats ne correspondent pas à la théorie.
3. Nous n'avons pas assez d'expérience pour bien entraîner un réseau de neurones sur nos problèmes.

5.2 Améliorations

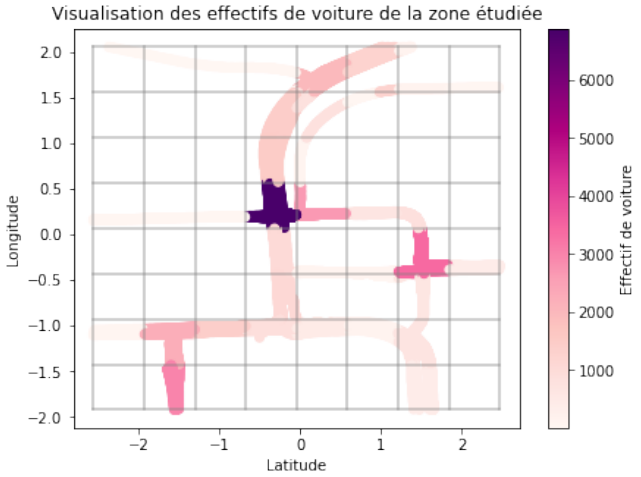
1. Faire plus de recherche et consulter les experts dans le domaine géographique.
2. Une manière de faire est la régularisation, par exemple réduire la couche cachée ou le nombre de neurones cachés.
3. Faire plus de recherches et éventuellement lire des articles scientifiques.
4. Attaquer la partie d'interpolation.

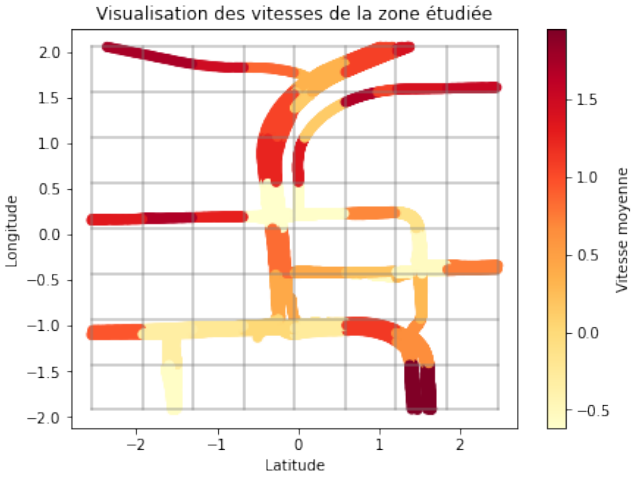
6 Conclusion

Nous voyons que les comportements des différents modèles sur ce jeu de données sont assez similaires dans l'ensemble. Les choses les plus importantes ou ce qui affectent le plus le comportement d'un modèle peut être la source et le pre-processing de nos données.
Ce projet nous a permis d'avoir une approche originale sur la prédiction de positions GPS. Nous avons tenté de prédire la position des véhicules à partir d'enregistrements GPS au lieu d'une carte.
L'anonymisation des données GPS est aussi un point difficile à aborder. La manipulation de données GPS pose de vrais problèmes légaux. Les données GPS sont extrêmement dures à anonymiser, c'est assez facile de retrouver à qui appartient une trace.

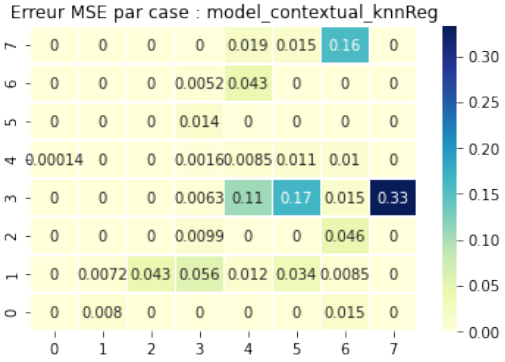
7 Annexes

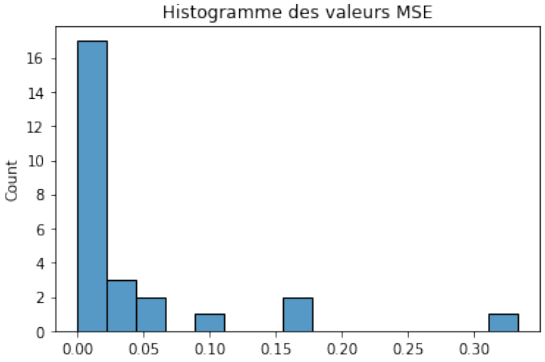
7.1 k-NN Regression 1

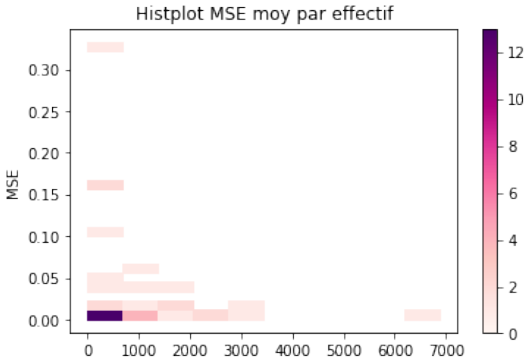


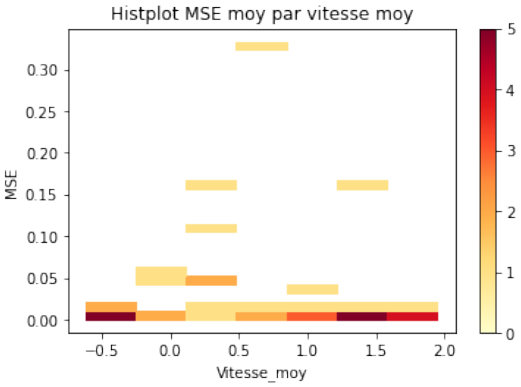


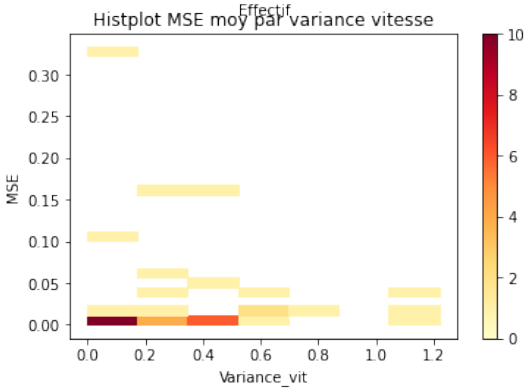
model_contextual_knnReg



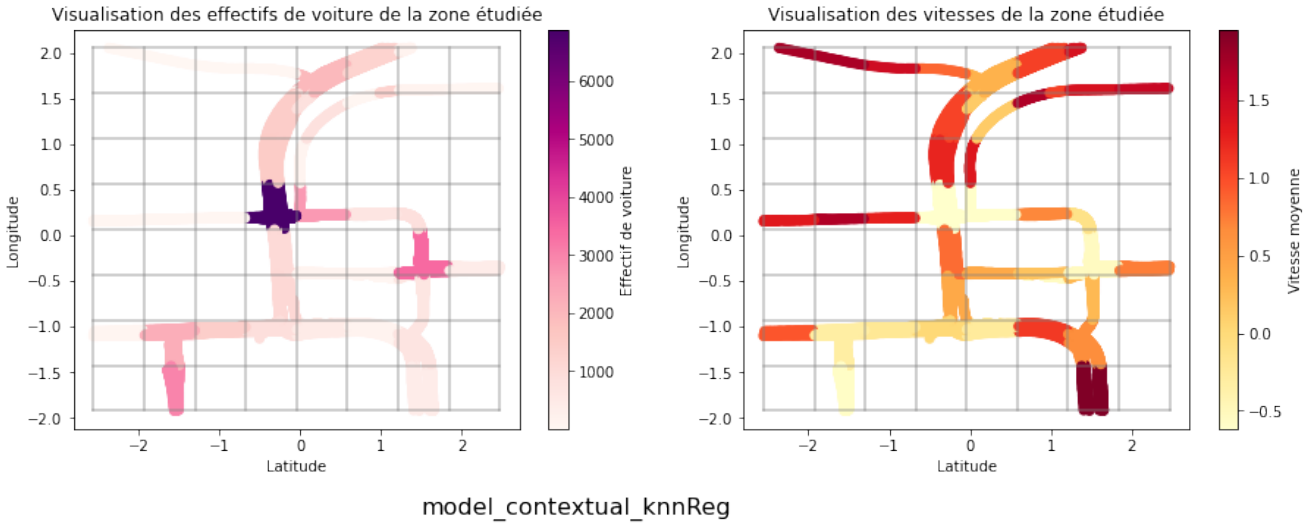




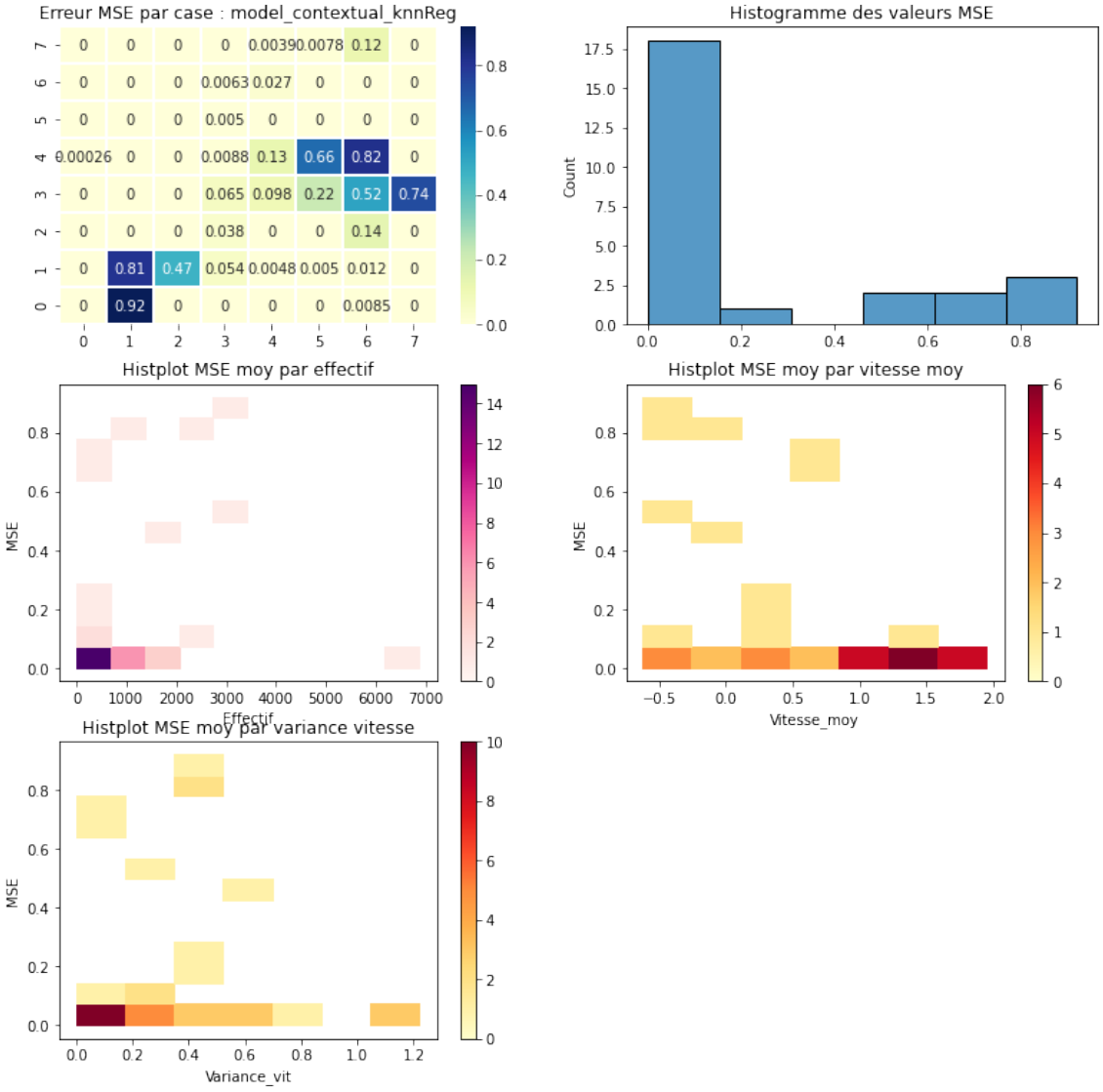




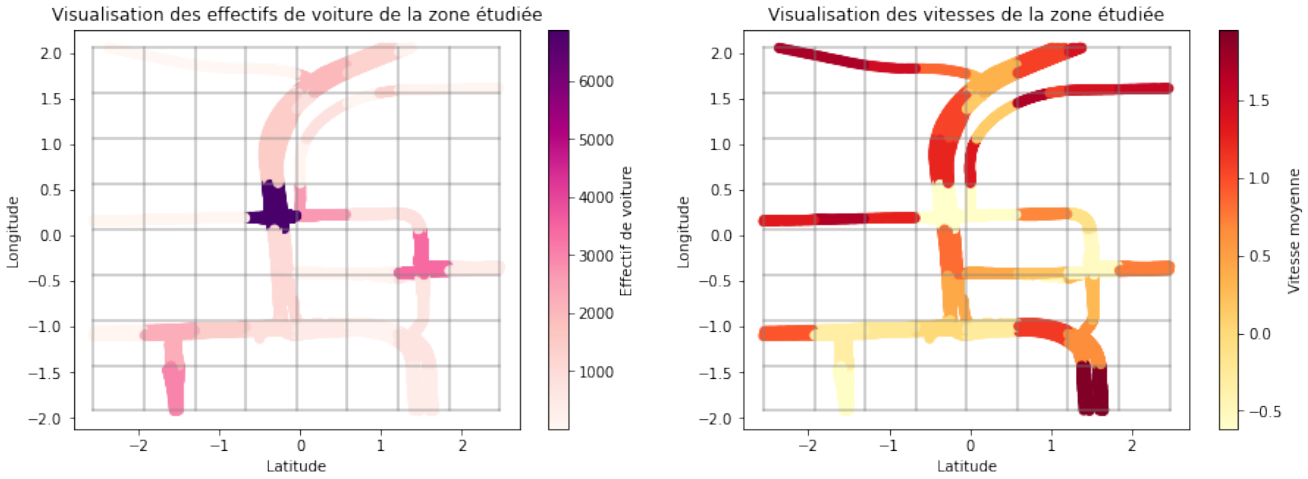
7.2 k-NN Regression 2



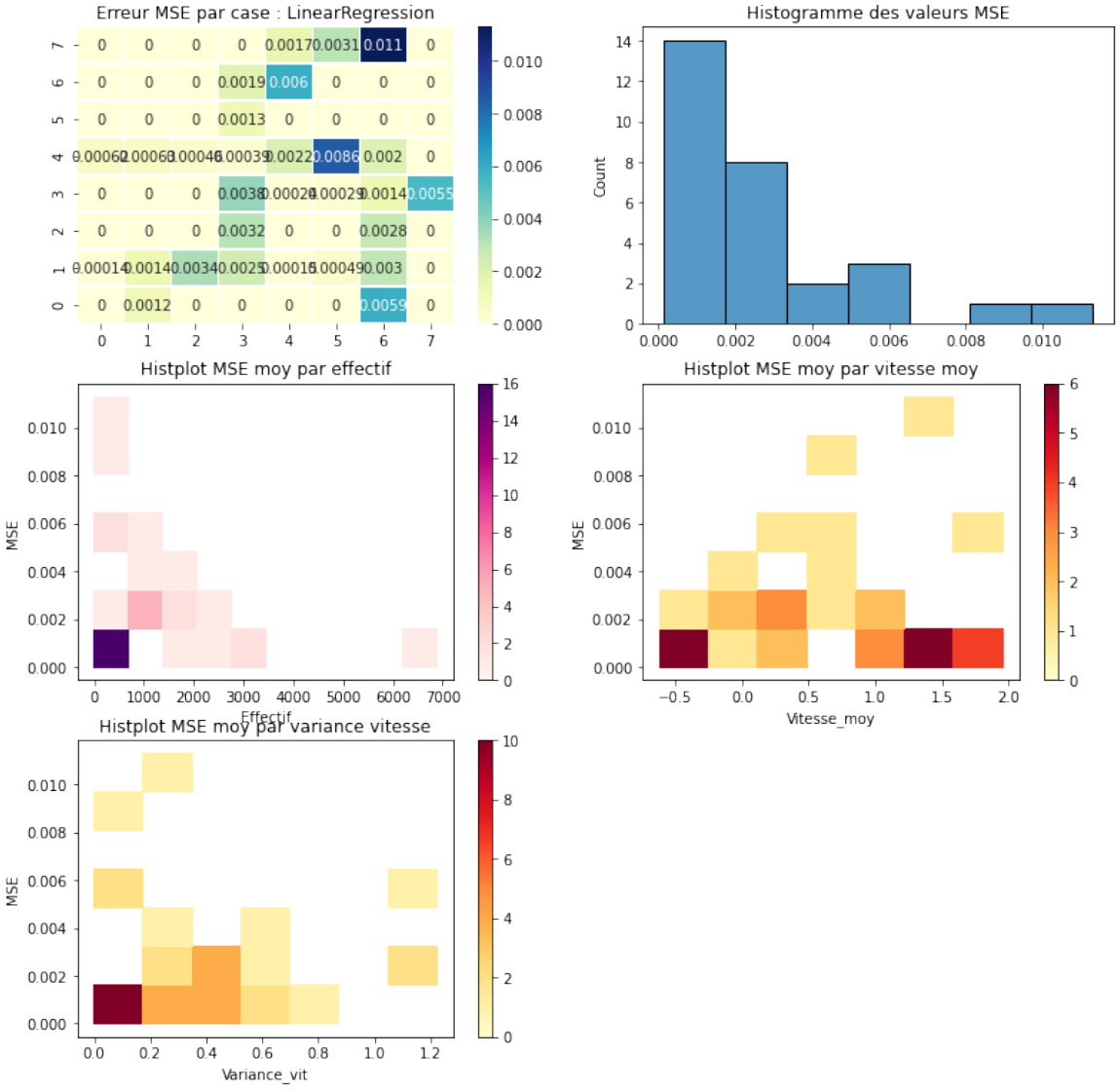
model_contextual_knnReg



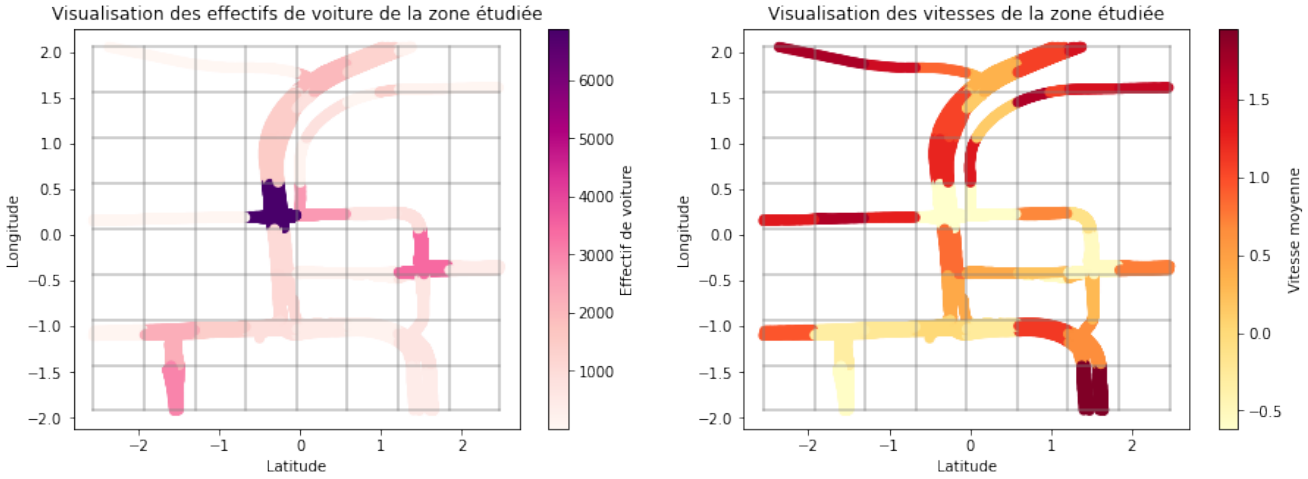
7.3 Linear Regression



LinearRegression



7.4 Modèle Physique 1



model_physique1