Nantes Université

Projet Data challenge M2 IS

Novembre 2024

Anticipez l'affluence au sein des gares SNCF-Transilien

Andréa KOUADIO

Moubarak LASSISSI

Prof : Claire BRECHETEAU

Contents

1 Introduction

Imaginez un lundi matin dans une gare très fréquentée de la région parisienne. Les voyageurs se présentent pour monter dans les trains, créant une densité de passagers élevée sur les quais et dans les wagons. S'il y a trop de monde, cela entraîne des retards, des trains surchargés et une expérience de trajet désagréable pour les usagers. Inversement, si le nombre de voyageurs est moins élevé que prévu, des ressources sont mobilisées inutilement, ce qui a un coût pour l'exploitant du réseau.

Dans ce contexte, il est essentiel pour SNCF-Transilien d'anticiper, à moyen et long terme, le nombre de validations quotidiennes dans chaque gare de Transilien. En anticipant les volumes de voyageurs, SNCF-Transilien peut améliorer la qualité de service tout en optimisant ses coûts opérationnels. C'est précisément l'objectif de ce projet, qui vise à comprendre et à prédire les flux de validation sur l'ensemble du réseau Île-de-France, afin d'assurer une meilleure gestion des flux passagers et d'anticiper les évolutions futures de la demande.

2 Présentation des Données

Pour notre étude, la base de données a été scindée en deux parties, un jeu de données d'entraînement **train** et un jeu de données **test**. Elle inclut des informations telles que la date, l'identifiant de la gare, des indicateurs de jours ouvrables, de jours fériés et de vacances scolaires, ainsi que le nombre de validations par jour et par station. Dans cette section, nous allons présenter la base de données dans le tableau 1.

Variables	**Type de variables**	**Description**
`date`	Temporelle	Date des validations (format YYYY-MM-DD).
`station`	Qualitative	Identifiant de la gare anonymisé (ex. 7RP, J3V).
`job`	Qualitative	vaut 1 si le jour est un jour ouvrable , 0 sinon.
`ferie`	Qualitative	vaut 1 si le jour est férié, 0 sinon.
`vacances`	Qualitative	vaut 1 si le jour est un jour de vacances scolaires, 0 sinon.
`y`	Quantitative	Nombre de validations par jour et par station.

Table 1: Description des variables du jeu de données

Le jeu de données train contient les données quotidiennes de validation des 448 stations du réseau SNCF-Transilien couvrant la période du 1er janvier 2015 au 31 décembre 2022, soit 2922 jours. Tandis que le jeu de données test couvre le premier semestre de 2023, soit les 181 premiers jours du 1er janvier 2023 au 30 juin 2023.

3 Analyse des données

3.1 Impact des variables contextuelles

Nous allons effectuer des méthodes d'analyse descriptive, en particulier une **analyse bivariée**, pour obtenir une vue d'ensemble de notre jeu de données et mieux en comprendre la structure avant de procéder aux étapes de modélisation. Notre première analyse consistera à explorer les relations entre notre variable cible y (le nombre de validations) et d'autres variables contextuelles afin d'identifier des tendances dans la fréquentation des stations.

Pour la figure 1 :

- Nous examinons la **relation entre y et les jours fériés** (`ferie`), en identifiant les stations les plus fréquentées les jours où cet indicateur est à 1. Cela nous permet de comprendre l'impact des jours fériés sur la fréquentation des stations.
- Nous analysons la **relation entre y et les vacances scolaires** (`vacances`), en isolant les périodes de vacances et en identifiant les 10 stations ayant le plus de validations durant ces périodes, ce qui nous donne des indications sur la fréquentation saisonnière.
- Nous étudions également la **relation entre y et les jours ouvrés** (`job`), pour voir quelles sont les 10 stations les plus fréquentées les jours de travail, reflétant les flux de déplacement régulier de la population active.

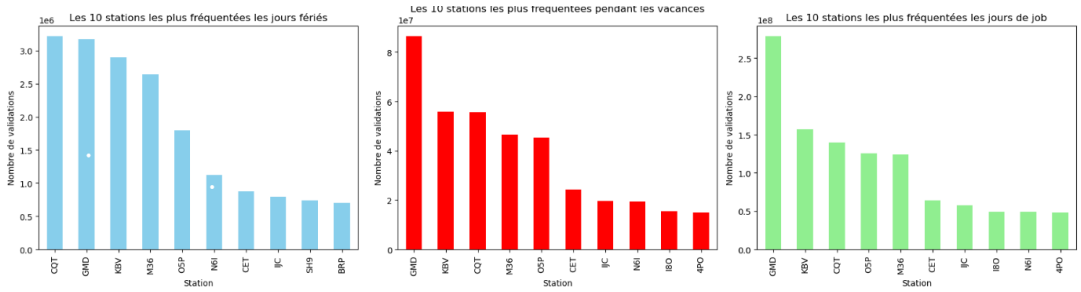

Figure 1: Les 10 stations ayant le plus d'affluence pendant les jours fériés, de vacances et de job.

Les barres représentent le nombre de validations pour chaque gare. Les gares sont listées sur l'axe des x, et le nombre de validations est sur l'axe des y.

Les gares "GMD", "KBV" et "CQT" apparaissent fréquemment parmi les plus fréquentées les jours fériés, les vacances et les jours de travail.

3.2 Analyse temporelle

Afin d'approfondir notre compréhension des dynamiques de fréquentation, nous allons concentrer notre analyse sur la variable contextuelle Date, qui inclut des informations sur le jour, le mois et l'année. En explorant les composantes jour et mois, nous pourrons identifier d'éventuelles tendances selon les jours et les périodes de l'année, et ainsi mieux comprendre les moments de forte affluence.

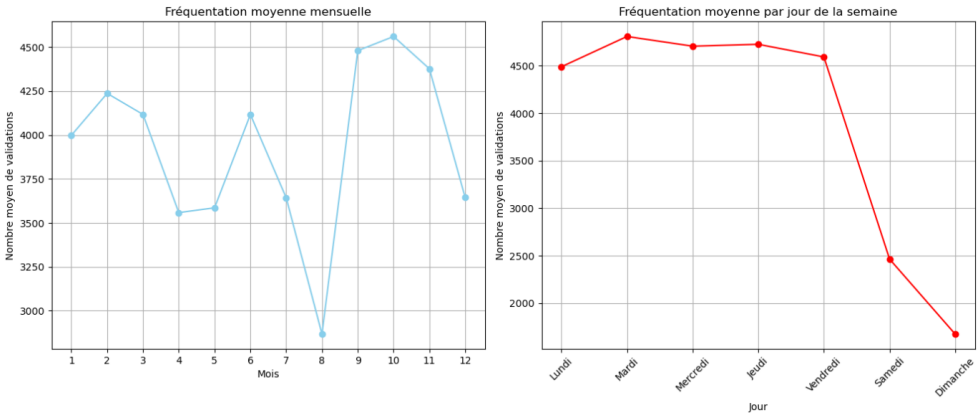

Figure 2: Le jour et le mois qui enregistrent les plus fortes affluences.

- D'après le graphique de droite, **le mardi** est le jour avec la fréquentation la plus élevée. Cela peut s'expliquer par le fait que les mardis sont souvent des jours de pleine activité, sans l'effet de "rentrée" du lundi ni la proximité du week-end qui peut réduire les déplacements. En revanche, la fréquentation diminue fortement le week-end, avec **le dimanche** ayant la fréquentation la plus basse, ce qui est typique des comportements de mobilité urbaine.
- Le graphique de gauche montre que **octobre** est le mois avec la fréquentation la plus élevée. Octobre se situe en plein milieu de la rentrée, sans vacances scolaires majeures, ce qui entraîne un retour aux déplacements réguliers pour le travail et les études. En revanche, **août** reste le mois avec la fréquentation la plus basse, correspondant aux vacances d'été, où de nombreuses personnes prennent des congés.

En résumé, la fréquentation maximale se produit **en octobre et le mardi**, périodes associées à des déplacements réguliers de la population active et des étudiants, en dehors des vacances et jours de repos. Toutefois, pour compléter cette analyse des tendances de validation, il est également pertinent de s'intéresser à l'évolution du réseau lui-même. En effet, au fil des années, des modifications ont été observées dans la liste des stations, avec des ajouts et des suppressions régulières. Le tableau 1 résume ces évolutions, montrant les années de disparition et de réapparition de certaines stations ainsi que les nouvelles gares ajoutées au cours du temps.

Station	**Année de Disparition**	**Année d'Ajout**
52H	2017	2019
TPX	2019	2022
EFN	2019	2022
92P	2019	2022
L58	2019	2022
B9M	2019	2022
48S	-	2017
N9K	-	2017
V2P	-	2017
OWM	-	2017
QD6	-	2022
BDC	-	2022
P6E	-	2022
W80	-	2022
W14	-	2022

Table 2: Années de disparition et d'ajout des stations

Certaines stations qui ont été supprimées puis réintroduites :

- Par exemple, la gare **52H** a été supprimée en 2017, puis réintroduite en 2019.
- D'autres gares, comme **TPX**, **EFN**, **92P**, **L58**, et **B9M**, ont été supprimées en 2019 puis réintroduites en 2022.

4 Prétraitement des données

Pour le traitement des données, nous avons adopté plusieurs méthodes. D'abord, nous avons converti la variable date qui contenait initialement des valeurs de type date en chaîne de caractères (str), afin de faciliter leur utilisation dans nos modèles de machine learning ou d'analyse. Cette transformation a permis de manipuler les dates sous un format plus simple et homogène, tout en offrant la possibilité d'extraire facilement des informations spécifiques telles que l'année, le mois, ou le jour, si nécessaire pour nos modèles.Ensuite, pour la colonne représentant les stations, nous

avons attribué des numéros aux stations en utilisant la méthode LabelEncoder de la bibliothèque sklearn. LabelEncoder transforme les valeurs catégorielles des stations en entiers numériques uniques. Cette étape est cruciale car les modèles de machine learning ne peuvent pas traiter directement les données sous forme de texte. En attribuant un numéro à chaque station, nous avons converti ces variables qualitatives en variables quantitatives, permettant ainsi à nos modèles de les utiliser pour l'entraînement.

Après ces étapes de traitement des données, nous avons testé plusieurs méthodes classiques de machine learning pour évaluer leurs performances et vérifier si nous étions proches du benchmark. Nous avons utilisé des modèles tels que la régression linéaire, les arbres de décision, ainsi que des méthodes d'ensemble comme les forêts aléatoires. Cependant, lors de l'évaluation des résultats, nous avons observé que certaines prédictions étaient négatives, ce qui n'est pas normal, car les valeurs prédites dans ce cas devraient être positives. Pour corriger ce problème. Nous avons appliqué une transformation logarithmique à la cible y en utilisant la fonction **log1p**. Cette fonction permet d'appliquer le logarithme de $1 + y$, ce qui est particulièrement utile pour éviter des valeurs négatives dans y tout en gérant les valeurs proches de zéro. Cette transformation aide à stabiliser la variance, rendre la distribution plus symétrique et améliorer la performance des modèles en garantissant que les valeurs transformées restent positives et mieux adaptées aux algorithmes de machine learning. Pour revenir à l'échelle d'origine après la prédiction, on utilise `expm1()`, qui est l'inverse de la fonction `log1p()`.

5 Modélisation

L'objectif fondamental de cette section est de proposer une solution au problème initial, qui consiste à prédire le **nombre de validations quotidiennes** dans les gares du réseau SNCF-Transilien. Pour cela, notre stratégie est d'identifier l'algorithme le plus adapté à cette problématique.

Après une analyse approfondie, nous avons identifié que ce problème se présente comme une **problématique de régression**. En effet, nous cherchons à expliquer une variable cible continue qui est le nombre de validations quotidiennes à partir d'un ensemble de variables indépendantes. Cette identification claire du problème nous permet de mobiliser des **méthodes de machine learning** appropriées, axées sur la régression, afin d'améliorer la précision de nos prédictions.

Pour approfondir notre analyse, nous avons abordé une autre approche qui applique des techniques d'analyse fonctionnelle, une méthode moderne particulièrement adaptée à l'exploration de données volumineuses. Cette approche permet de modéliser l'évolution des voyageurs pour chaque station sous forme de séries temporelles. Cela facilite non seulement la visualisation et l'analyse de l'évolution du nombre de validations au fil du temps, mais également la prédiction de ces tendances futures. Après une étude, nous avons déterminé que les bases de B-splines cubiques ou de Fourier, avec 21 composants, offrent une modélisation précise de ces évolutions. Cette méthode fournit des indications en prédisant d'abord les coefficients des bases fonctionnelles choisies, ce qui permet ensuite de représenter les courbes et d'identifier les pics du nombre de validations pour chaque station.

5.1 Selection du meilleur modèle

La sélection des modèles en apprentissage automatique est en effet une étape clé du processus de développement, car elle permet d'identifier le modèle qui offrira les meilleures performances pour une tâche spécifique, tout en évitant des erreurs comme le surapprentissage ou le sous-apprentissage.

Dans cette étude, le benchmark, qui correspond à une performance de référence, est fixé à 177,0825. Nous utilisons le MAPE comme métrique principale pour évaluer les performances des modèles.

Mean Absolute Percentage Error (MAPE)

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{y_i}$$

- n : correspond au nombre d'index disponibles dans le jeu de données de test.
- y_i : valeur réelle (observée) pour l'observation i dans le jeu de données de test.
- $\hat{y}_i$: valeur prédite par le modèle pour la même observation i.

Nous avons décidé de commencer par tester des modèles de régression vus en cours sur nos données prétraitées afin d'établir une base de référence solide pour nos comparaisons futures. Ils offrent également une première estimation de la performance du modèle avant d'explorer d'autres modèles. Voici une description de quelques-uns de ces modèles.

- **La régression Ridge** est une technique de régularisation en apprentissage automatique qui ajoute une pénalité à la somme des carrés des coefficients du modèle linéaire.
- **Le BaggingRegressor** est une méthode d'ensemble qui combine plusieurs arbres de décision pour améliorer la précision des prédictions en régression. Chaque arbre est entraîné sur un sous-échantillon aléatoire des données d'entraînement (bootstrap), et les prédictions finales sont obtenues en moyennant celles des arbres.
- **Le réseau de neurones dense** est un type de réseau dans lequel chaque neurone d'une couche est connecté à tous les neurones de la couche suivante. Il est constitué d'une couche d'entrée représentant les caractéristiques, de couches cachées qui effectuent des transformations, et d'une couche de sortie générant les prédictions. Dans notre approche, les données sont d'abord normalisées à l'aide de StandardScaler. Le modèle séquentiel est conçu avec trois couches : une couche d'entrée avec 64 neurones, une couche cachée avec 32 neurones, et une couche de sortie avec 1 neurone pour la prédiction continue. Enfin, les prédictions sur le jeu de test sont inversées pour revenir aux valeurs réelles après la normalisation.

Nous avons également cherché à aller au-delà des limites des modèles vus en cours en explorant des méthodes plus avancées. L'objectif de cette démarche est de mieux capturer la complexité des données et d'améliorer la qualité des prédictions, tout en comparant différentes approches pour identifier celle qui est la plus adéquate à notre problématique. Pour cela , nous avons exploré de nouvelles méthodes de machine learning qui sont:

- **Le GradientBoostingRegressor** est une méthode d'ensemble qui construit une séquence d'arbres de décision de manière itérative, où chaque arbre corrige les erreurs (résidus) du précédent. Chaque nouvel arbre est ajusté pour minimiser une fonction de perte en utilisant la descente de gradient. L'algorithme combine les prédictions de tous les arbres pour produire une prédiction finale plus précise et robuste.
- **Le ExtraTreeRegressor** construit un seul arbre en choisissant aléatoirement les points de séparation, ce qui le rend plus rapide par rapport à un arbre de décision classique qui optimise chaque coupure. Cette randomisation améliore la diversité entre les arbres lorsqu'elle est utilisée dans des ensembles. Il prédit en utilisant la moyenne des valeurs cibles des observations dans chaque feuille.
- **Le ExtraTreesRegressor** est un modèle d'ensemble basé sur des arbres de décision fortement randomisés. Il construit plusieurs arbres en sélectionnant aléatoirement les points de séparation et les variables, ce qui améliore la diversité et réduit le risque de surapprentissage.

Nous comparons les performances de tous les modèles dans le tableau 3 afin de sélectionner celui qui est le mieux adapté à nos données.

Modèles	Score
LinearRegression	104
Ridge	103.6
RandomForestRegressor	147
BaggingRegressor	103.2
GradientBoostingRegressor	106
KNeighborsRegressor	165.92
ExtraTreeRegressor	**99.08**
ExtraTreesRegressor	99.23
DecisionTreeRegressor	99.38
Reseau neurone Dense	101.02

Table 3: Tableau comparatif des modèles

Après avoir testé et comparé plusieurs modèles, nous avons sélectionné celui qui offrait les meilleures performances. D'après le tableau 3, on peut conclure que notre meilleur modèle est **ExtraTreeRegressor** avec 99,08.

5.2 Modèle final avec les hyperparamètres optimaux

Une fois le modèle le plus performant sélectionné, l'étape suivante consiste à optimiser ses hyperparamètres afin d'améliorer ses performances. Pour ce faire, nous avons testé différentes méthodologies, notamment Grid Search et Random Search. La recherche en grille explore systématiquement toutes les combinaisons prédéfinies des hyperparamètres, garantissant une évaluation exhaustive mais au prix d'un coût computationnel élevé. En complément, la recherche aléatoire permet une exploration plus rapide en sélectionnant de manière stochastique un échantillon des combinaisons possibles.

L'optimisation des hyperparamètres a permis d'identifier une configuration optimale pour le modèle **ExtraTreeRegressor**. Cette configuration privilégie une profondeur illimitée des arbres (`max_depth=None`), ce qui permet au modèle d'exploiter pleinement les relations présentes dans les données. Par ailleurs, l'absence de restriction sur le nombre de caractéristiques sélectionnées à chaque division (`max_features=None`), la faible contrainte sur la taille des feuilles et des divisions (`min_samples_leaf=1` et `min_samples_split=3`) assurent une flexibilité maximale dans la construction des arbres. Cette combinaison d'hyperparamètres équilibre la précision du modèle tout en limitant les risques de surajustement, ce qui a considérablement amélioré ses performances sur les données du challenge.

Après optimisation des paramètres avec le modèle **ExtraTreeRegressor**, nous avons obtenu un score final de **96.02** qui est largement supérieur au benchmark du challenge qui était initialement 177.08, ce qui confirme la capacité de notre modèle à prédire avec précision tout en restant efficace sur de grands ensembles de données. En comparaison avec d'autres modèles testés précédemment, l'ExtraTree s'est montré supérieur en termes de précision et de capacité à traiter des données.

Bien que la validation croisée soit une pratique courante pour évaluer la robustesse des modèles, nous avons choisi de ne pas l'appliquer dans ce projet en raison des spécificités du challenge. De plus, nous n'avons pas créé d'échantillon de validation distinct. Cette décision s'explique par le fait que les organisateurs du challenge ne fournissaient pas les valeurs cibles $Y_{\mathbf{test}}$ pour l'ensemble de test, ce qui imposait une évaluation directe sur cet ensemble. En nous concentrant sur l'optimisation des hyperparamètres sur l'ensemble d'entraînement et en évaluant directement

les performances sur l'ensemble de test, nous avons maximisé l'utilisation des données disponibles tout en respectant les contraintes fixées. Cette approche, bien que différente des pratiques classiques, s'est révélée efficace pour ce contexte spécifique.

6 Conclusion

En conclusion, nous avons réalisé une analyse approfondie du jeu de données mis à disposition. L'objectif principal était de prédire avec précision le nombre de validations, un indicateur clé pour anticiper l'affluence dans les gares de Transilien à moyen et long terme. Nous avons commencé par une exploration approfondie des données, ce qui nous a permis de mieux comprendre leur structure et d'identifier les variables ayant un impact significatif sur les prévisions. L'analyse exploratoire a mis en évidence des relations importantes, notamment entre les jours de la semaine, les mois, les stations et le nombre de validations. Ces insights ont guidé nos choix de caractéristiques pour la modélisation, afin d'optimiser la précision des prédictions. Grâce à ces analyses et à l'application de techniques de machine learning, nous avons construit un modèle robuste capable de prédire les validations quotidiennes avec une certaine précision. Ce projet nous a offert l'opportunité d'appliquer des techniques de machine learning à un problème réel, en exploitant des données complexes pour générer des prédictions utiles.