UNIVERSITÉ PARIS 8 – VINCENNES SAINT-DENIS

UFR : Mathématiques, Informatique et Technologies

Techniques d'apprentissage automatique

Projet finale : Prédiction du nombre totale de cas de la maladie de Dengue (Driven Data)

Étudiant : ALI Ait Moussa
Numéro étudiant : 82504858
Formation : Master 1 Ingénierie de l'Intelligence Artificielle

Responsable du module : Mr. Karim HAROUN

Date : Janvier 2026

Année universitaire : 2025-2026

Abstract

Ce projet s'inscrit dans le cadre du challenge DengAI : Predicting Disease Spread proposé par DrivenData. L'objectif est de prédire le nombre hebdomadaire de cas de dengue dans deux villes tropicales (San Juan et Iquitos) à partir de variables climatiques, environnementales et temporelles. Il s'agit d'un problème de régression supervisée, avec une forte dimension temporelle et saisonnière. Dans ce travail, nous avons mis en place un pipeline complet de Machine Learning incluant l'analyse exploratoire des données, le prétraitement, la sélection de modèles, l'optimisation des hyperparamètres et l'évaluation des performances. Plusieurs modèles de régression ont été testés, notamment des modèles basés sur les arbres (Random Forest, Gradient Boosting), qui sont bien adaptés aux relations non linéaires présentes dans les données climatiques. Les résultats montrent que les modèles ensemblistes offrent de bonnes performances et une robustesse face au bruit et aux valeurs manquantes. Une analyse détaillée des résultats permet de mettre en évidence les forces et limites de l'approche proposée ainsi que des pistes d'amélioration, notamment l'intégration de modèles temporels plus avancés.

Table des matières

Chapitre 1

Introduction

1.1 Contexte

La dengue est une maladie virale transmise par les moustiques, représentant un problème majeur de santé publique dans de nombreuses régions tropicales. Sa propagation est fortement influencée par des facteurs climatiques et environnementaux tels que la température, l'humidité et les précipitations. La prédiction du nombre de cas de dengue permet d'anticiper les épidémies et d'aider les autorités sanitaires à mieux planifier les actions de prévention.

1.2 Motivations

Les relations entre les variables climatiques et la propagation de la dengue sont complexes, non linéaires et marquées par une forte saisonnalité. Les méthodes statistiques classiques montrent souvent leurs limites face à ce type de données réelles. Les approches de Machine Learning offrent une alternative pertinente, car elles permettent de modéliser efficacement des interactions complexes à partir de données hétérogènes et bruitées.

1.3 Objectifs scientifiques

L'objectif de ce projet est de concevoir un pipeline complet de Machine Learning pour prédire le nombre hebdomadaire de cas de dengue dans deux villes tropicales (San Juan et Iquitos), à partir de données climatiques et temporelles. Plus précisément, il s'agit d'analyser l'influence des variables environnementales, de comparer plusieurs modèles de régression et d'évaluer leurs performances de manière rigoureuse et reproductible.

1.4 Travaux existants

Plusieurs travaux ont étudié la prédiction des maladies vectorielles à l'aide de modèles statistiques et de Machine Learning. Des approches basées sur des modèles linéaires ou autoregressifs ont été proposées, mais elles capturent difficilement la non-linéarité des données climatiques. Des modèles ensemblistes tels que les Random Forests et le Gradient Boosting ont montré de bonnes performances pour ce type de problème, ce qui motive leur utilisation dans ce projet.

Chapitre 2

Méthodologie

2.1 Type de tâche

Le problème étudié est un problème de régression supervisée. La variable cible est le nombre total de cas de dengue observés par semaine dans chaque ville.

2.2 Choix des modèles

Les choix des modèles pour ce problème de régression on été fait tout en respectant les conditions qu'on a décrit dans la section precedente.

Les modèles qu'on a choisi sont les suivants :

— **Random Forest**
— **XGBoost**
— **Gradient Boosting**
— **MLPRegressor**
— **ANN (Réseau de neurones créer avec PyTorch)**

La sélection de ces modèles a été faite car ils ont la capacité a modéliser des relations non linéaires et a capturer les motifs subtils, leur robustesse face aux distributions asymétriques et aux valeurs extrêmes, ainsi que pour leur aptitude à gérer les interactions non linéaires sans nécessiter de prétraitement intensif.

2.3 Hyperparamètres, stratégie d'optimisation

Dans ce projet on a utiliser plusieurs techniques afin d'évaluer les modèles qu'on a choisi.

— La première technique, est l'utilisation de gridSearch pour chaque modèle afin d'identifier les valeurs des hyperparametres les plus performants car ces derniers ont un impacte sur les modeles.

— La deuxieme technique, est d'utiliser cross validation afin d'evaluer les vrai performances de chaque modele.
— La derniere technique, est l'application du PCA pour simplifier la tache aux reseaux de neurones qu'on a utiliser pour ce projet.

Chapitre 3

Experiences

3.1 Dataset et settings

3.1.1 Prétraitement des données

L'étape de préparation et de prétraitement des données est une étape cruciale avant de passer à l'entraînement des modèles.

Elle consiste à transformer les données non numériques en données numériques, à encoder les variables catégorielles, et à créer de nouvelles variables afin de préserver la cohérence des données et de mieux modéliser les relations entre elles, notamment dans le cas des données temporelles (c'est-à-dire lorsque les observations dépendent des données passées sur un intervalle de temps).

Cette étape inclut également la gestion des valeurs manquantes afin d'éviter tout biais susceptible d'affecter les performances des modèles.

3.1.2 Type de données

Le dataset contient plusieurs caractéristiques (colonnes), et on vas les résumer sous la figure suivante :

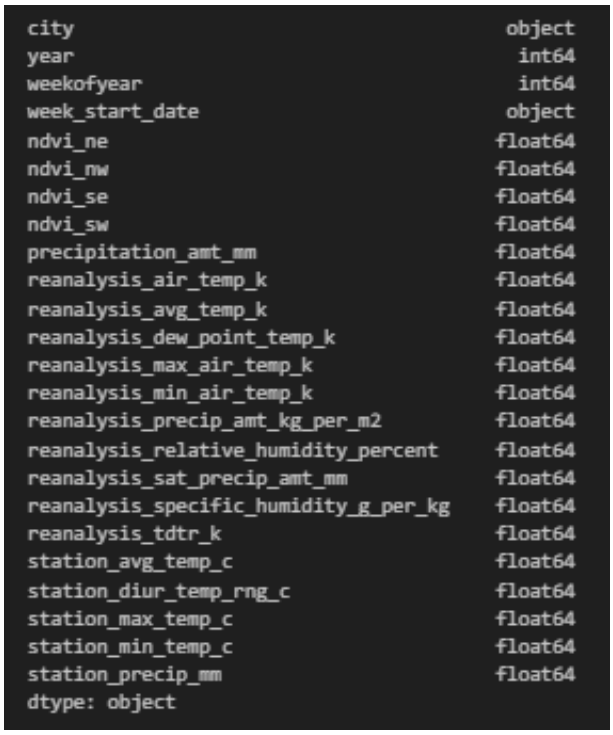


FIGURE 3.1 – Présentation des caractéristiques

Donc la plupart des caractéristiques sont numériques, mais y'en a qui sont de type chaîne de caractère (object) qui nécessite des transformations numérique pour qu'on puisse entraîner nos modèles.

3.1.3 Gestion des caractéristiques non numérique

Encodage de la colonne week_start_date

La variable week_start_date est convertie au format datetime afin de permettre une manipulation correcte des informations temporelles. À partir de cette date, deux nouvelles variables sont construites :

— l'année **(year)**, qui permet de capturer des tendances longues ou des effets inter-annuels.

— Un index temporel **(time_index)**, représentant le nombre de semaines écoulées depuis le début de la série. Cet index fournit une représentation numérique continue du temps, exploitable directement par les modèles de Machine Learning.

Les données sont ensuite triées par date afin de garantir la cohérence chronologique des observations, étape indispensable avant toute création de variables retardées.

Création des variables retardées (lag features)

Une étape centrale du prétraitement consiste à créer des variables retardées à partir de la variable cible **total_cases**.

Les variables **lag_1, lag_2, lag_4 et lag_52** correspondent respectivement au nombre de cas observés une semaine, deux semaines, quatre semaines et cinquante-deux semaines auparavant.

L'introduction de ces variables est motivée par la nature temporelle et épidémiologique du problème.

La propagation de la dengue dépend des conditions passées en raison du cycle de vie du moustique, du temps d'incubation du virus et des dynamiques saisonnières.

Le lag à 52 semaines permet notamment de capturer la saisonnalité annuelle, tandis que les lags courts modélisent l'inertie à court terme du nombre de cas.

Ces variables permettent à des modèles non explicitement temporels, tels que les Random Forests ou le Gradient Boosting, d'exploiter l'information historique et d'apprendre des dépendances temporelles sans recourir à des modèles de séries temporelles dédiés.

Gestion des valeurs city

Vu que le challenge se repose sur la prédiction du nombre totale des cas de la maladie de la dengue pour les deux différentes villes (sj et iq), nous allons d'abord analyser la distribution des valeurs de ces deux villes.

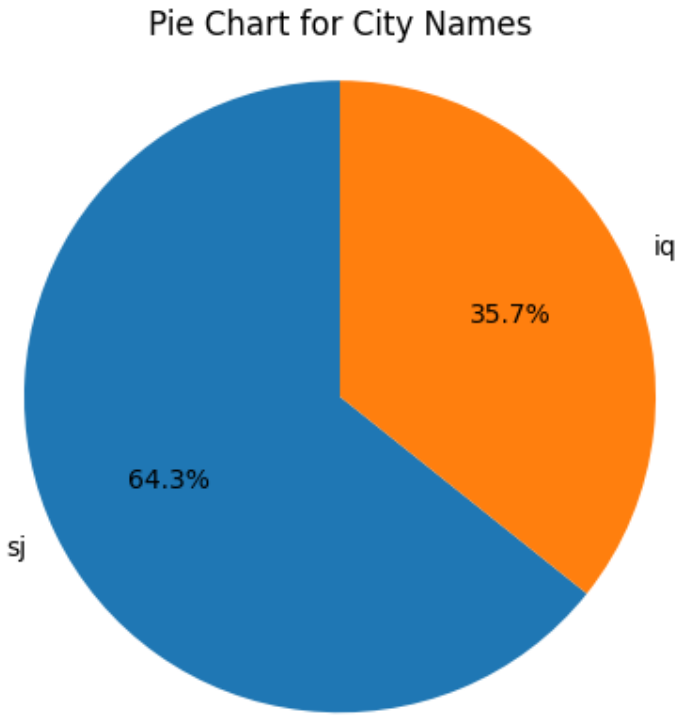


FIGURE 3.2 – Distribution de la variable city

D'après **la figure 3.2** on remarque que la ville sj domine la la ville iq, et donc pour ne pas biaiser nos modèles, et pour ne pas rendre le modèle apprendre majoritairement sur la ville sj, nous allons séparer les données de chaque ville, et on vas entraîner nos modèles sur chaque ville ensuite on analyse les résultats qu'on vas obtenir.

Vu que **city** contient seulement les valeurs (sj et iq), nous allons encoder cette dernière par un codage binaire au lieu d'utiliser le onehotencoding car cela vas nous créer une nouvelle variable qui vas nous servir a rien.

3.1.4 Gestion des valeurs manquantes

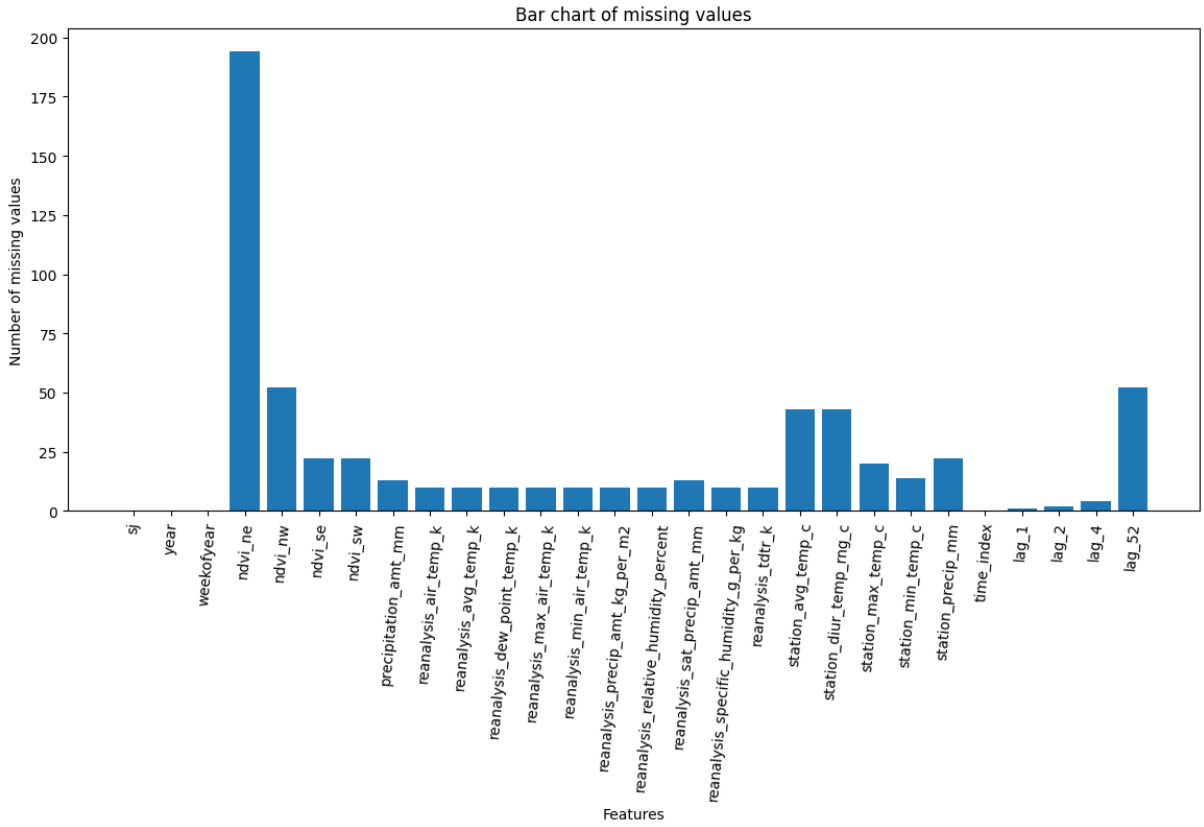


FIGURE 3.3 – Distribution des valeurs manquantes

D'après la distribution au niveau de l'histogramme, on remarque que le dataset contient plusieurs valeurs manquantes.

Donc après l'encodage des variables non-numérique et la gestion de la variable temporelles, maintenant on vas appliquer la stratégie **"mean"** afin de remplacer les valeurs manquantes.

Dans ce challenge on a opter pour la stratégie **"mean"** parce que :

— C'est plus facile a implémenter.

— Rapide, pas besoin de modèle supplémentaire.

— Ne change pas la distribution de façon radicale.

— Compatible avec la plupart des modèles ML.

3.1.5 Distribution des variables

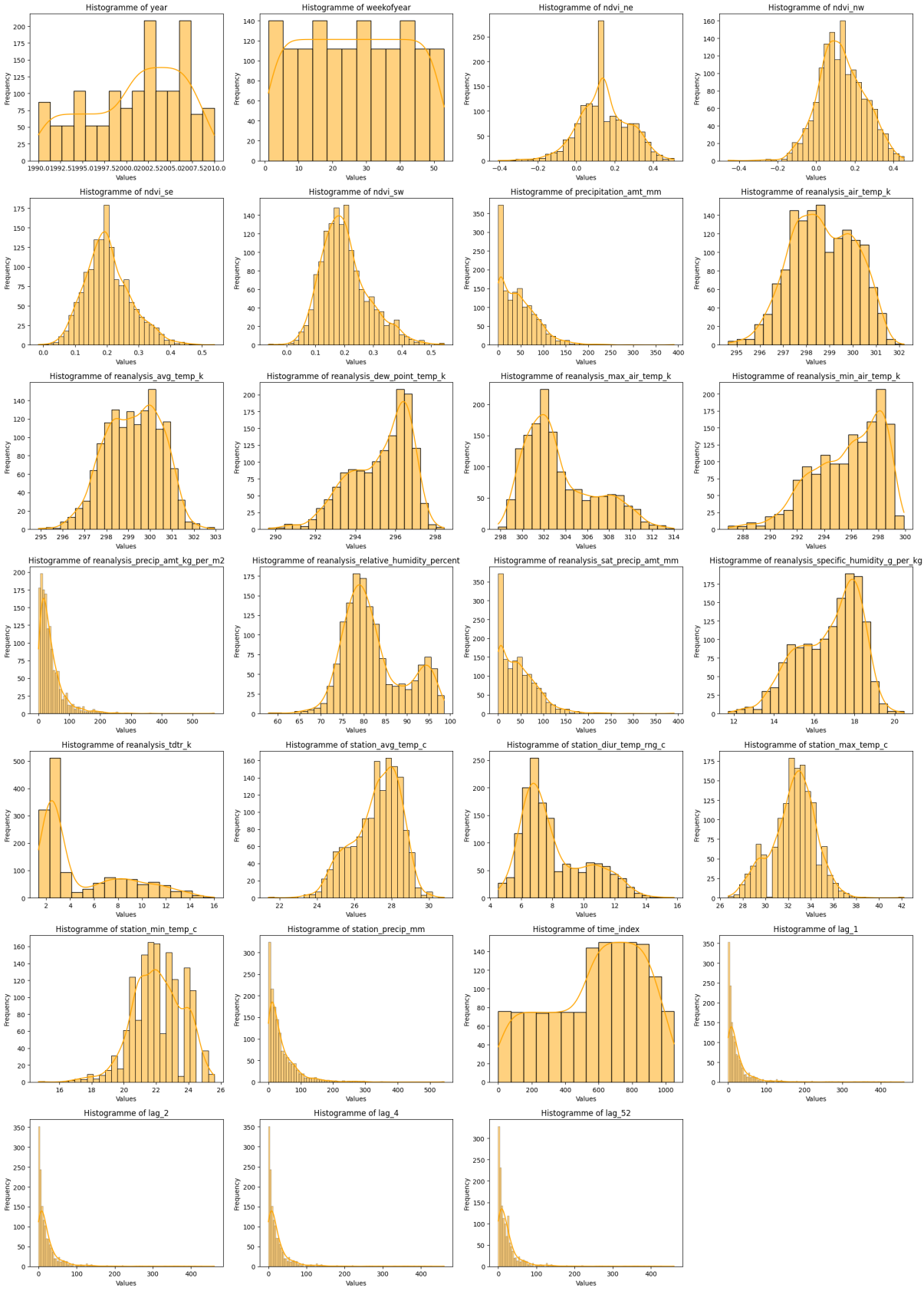


FIGURE 3.4 – Distribution des variables du dataset

D'après la figure **3.4** on remarque que :

— **Year, Weekofyear, Time_index :** Les distributions sont uniformes, ce qui suggère une répartition égale des données sur la période étudiée.

— **NDVI (NE, NW, SE, SW) :** Les distributions légèrement asymétriques à droite, ce qui signifie une légère concentration des valeurs vers la gauche.

— **Precipitation_amt_mm, Station_precip_mm, Reanalysis_precip_amt_kg_per_m Reanalysis_sat_precip_amt_mm :** Les distributions sont fortement asymétriques à droite, avec une longue traîne de valeurs élevées.

— **Températures :** La plupart des distributions sont symétriques.

— **Humidité :** Les distributions sont légèrement asymétriques a gauche.

— **Variables de lag :** Les distributions sont asymétriques à droite.

Donc pour conclure les modèles a choisir doivent être :

— capable a gérer des variables sans biais de distribution.

— doivent être robustes aux légères asymétries et capables de capturer les variations.

— doivent être capables de gérer les valeurs extrêmes et les distributions asymétriques.

3.1.6 Matrice de correlation

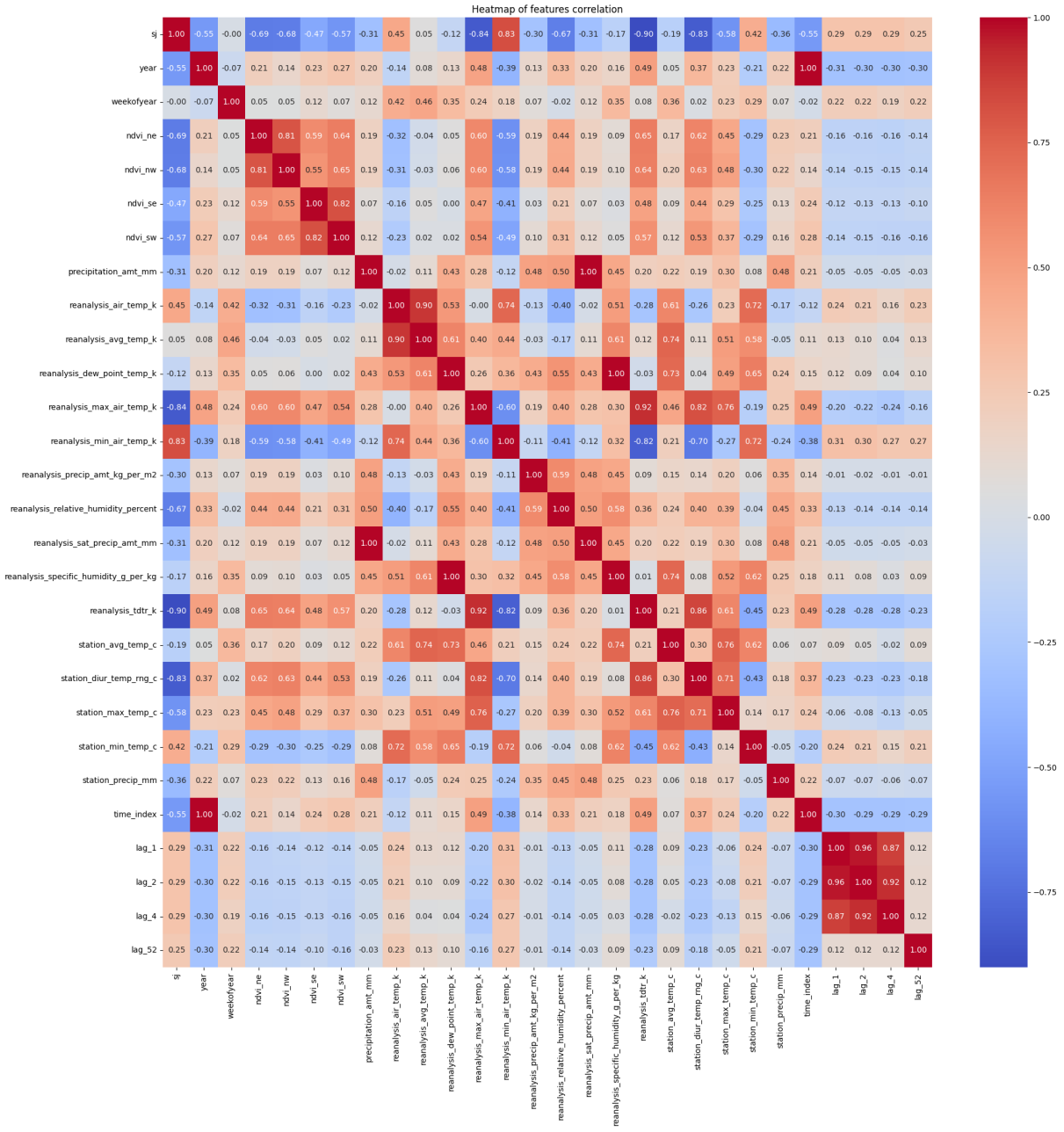


FIGURE 3.5 – HeatMap

Une analyse exploratoire est réalisée afin d'étudier la distribution des cas de dengue et leur évolution temporelle. La Figure de la matrice de corrélation (heatmap) permet d'analyser les relations linéaires entre les différentes variables climatiques, temporelles et les variables retardées.

La heatmap met en évidence plusieurs groupes de variables fortement corrélées entre elles. Les variables liées à la température (températures minimale, maximale, moyenne, issues des stations et des réanalyses) présentent des corrélations élevées, ce qui est attendu puisqu'elles mesurent des phénomènes physiques proches. Cette redondance justifie l'uti-

lisation de modèles capables de gérer la multicolinéarité, comme les modèles ensemblistes.

Les indices de végétation (NDVI) montrent également de fortes corrélations entre eux, reflétant des conditions environnementales similaires. Leur corrélation modérée avec certaines variables climatiques suggère un lien indirect entre la végétation, l'humidité et la prolifération des moustiques.

Les variables d'humidité et de précipitation présentent des corrélations positives avec certaines variables de température et avec la variable cible, confirmant leur rôle important dans la propagation de la dengue. Toutefois, ces corrélations restent modérées, ce qui indique que la relation entre climat et cas de dengue est complexe et non strictement linéaire.

Enfin, les variables retardées (lag_1, lag_2, lag_4, lag_52) montrent une forte corrélation entre elles et avec la variable cible, en particulier pour les lags courts. Le lag à 52 semaines met en évidence la saisonnalité annuelle marquée de la dengue. Ces observations confirment la pertinence de l'ingénierie de variables temporelles mise en place et expliquent l'amélioration des performances des modèles intégrant ces caractéristiques.

3.1.7 Detection des outliers

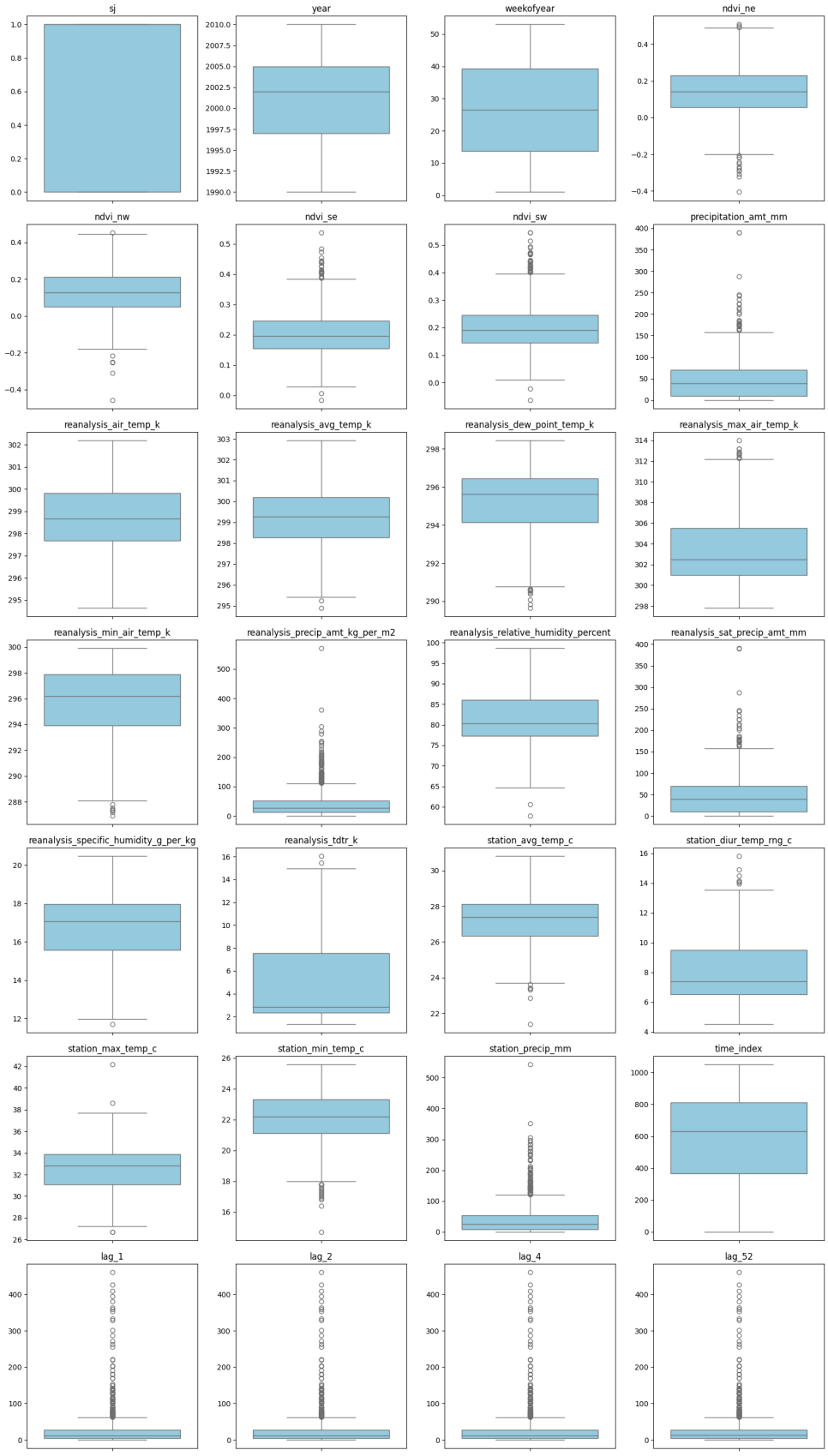


FIGURE 3.6 – Boxplots

D'après la figure 3.6, on remarque que y'a une présence des valeurs aberrantes dans plusieurs variables et donc une normalisation des donnees est necessaire pour les reseaux de neurones contrairement aux methodes ensemblistes.

3.1.8 Separation des données et normalisation

Après le grand nettoyage de données, et les transformer en format numerique, nous avons suivi les etapes suivantes afin de proceder a l'entrainement des modeles qu'on a choisi dans la section precedente :

- D'abord on a séparer nos données, ainsi que nos labels en deux section, la première section contient seulement les données de la ville sj, ainsi que ces labels, la deuxième section contient seulement les données relatives a la ville iq ainsi que ces labels.
- Ensuite, on a deviser chaque section de données en données d'entraînement et de test afin qu'on puisse entraîner et évaluer nos modèles.
- A la fin, on a aussi appliquer la normalisation sur chaque section car on vas entraîner des réseaux de neurones, et pour ne pas les biaiser, la normalisation est une étape crucial.

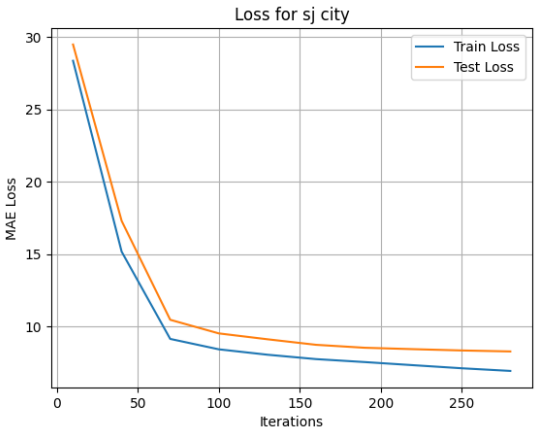
Donc a la fin, on aura des modèles qui vont utiliser les données non-normalisée car leur robustesse leur permet de bien séparer et suivre la distribution des données, et on aura aussi des modèles qui doivent utiliser les données normalisée afin qu'il puisse prédire d'une manière plus efficace.

3.2 Résultats d'entraînement et analyse

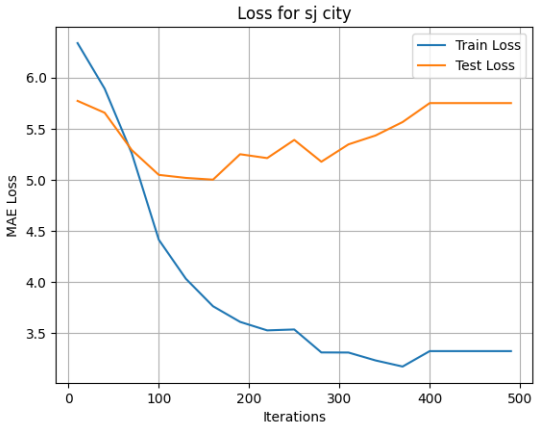
D'après la description du challenge sur driven data, le but est de réduire au maximum l'erreur absolue (MAE).

Pour cela on vas évaluer nos modèles sur deux métriques qui sont MAE et le R^2, d'où MAE est une métrique qui permet d'évaluer l'erreur, et R^2 afin d'évaluer la performance.

3.2.1 Modèle MLPRegressor :



(a) Loss pour la ville sj



(b) Loss pour la ville iq

FIGURE 3.7 – Fonction de perte pour les deux villes sur les données normalisée

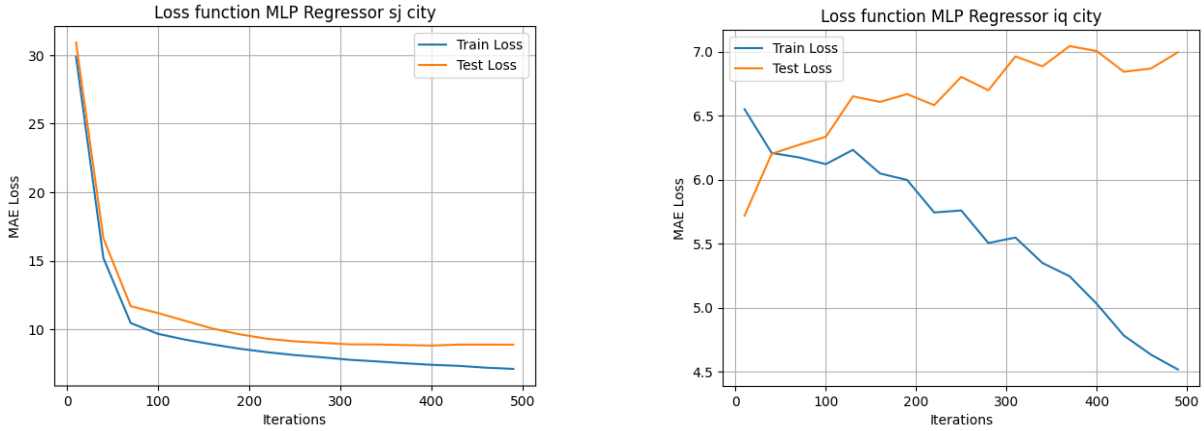
D'après la **figure 3.7** on constate que :

— Le modèle est parfait sur le dataset de la ville sj.

— Le modèle overfit sur le dataset de la ville iq, et ca c'est a cause de manque de données.

.

Application du PCA



(a) Loss pour la ville sj

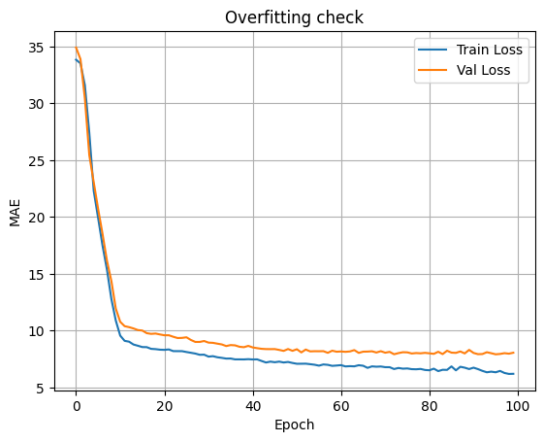
(b) Loss pour la ville iq

FIGURE 3.8 – Fonction de perte pour les deux villes sur les données normalisée + application du PCA

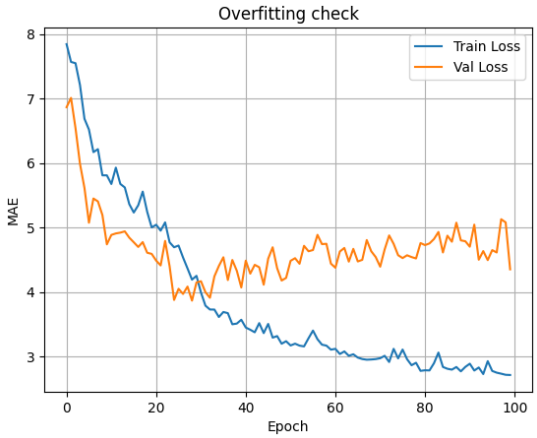
D'après **la figure 3.8** on remarque que y'a aucune amélioration au niveau du modèle MLPRegressor après l'application du PCA sur la ville iq, et donc le modèle n'arrive pas a généraliser, donc en vrai il est biaiser a cause du manque de données.

3.2.2 Modèle ANN

Ce modèle est un réseau de neurones qu'on a construit avec PyTorch, avec une régularisation L1 (Lasso).



(a) Loss pour la ville sj

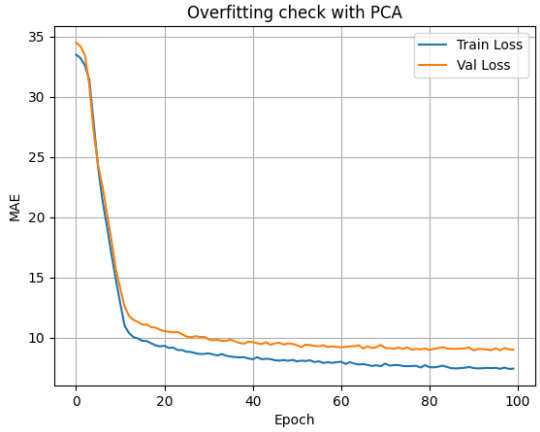


(b) Loss pour la ville iq

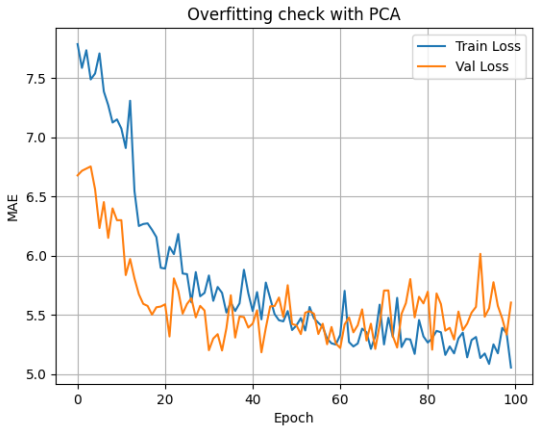
FIGURE 3.9 – Fonction de perte pour les deux villes sur les données normalisée -ANN-

Comparant a la **figure 3.7 et 3.8** on remarque que y'a une amélioration sur les données de la ville iq, cependant le modèle overfit et reste instable a cause du manque de données comparant aux données de la ville sj.

Application du PCA



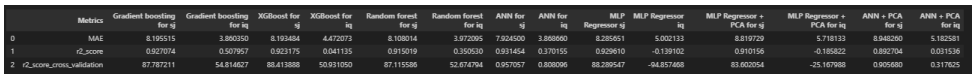
(a) Loss pour la ville sj



(b) Loss pour la ville iq

FIGURE 3.10 – Fonction de perte pour les deux villes sur les données normalisée + PCA -ANN-

D'après la **figure 3.10**, on remarque que y'a une amélioration significative sur les données de la ville iq, mais le modèle reste toujours instable car on observe des piques.



	Metrics	Gradient boosting for sj	Gradient boosting for iq	XGBoost for sj	XGBoost for iq	Random forest for sj	Random forest for iq	ANN for sj	ANN for iq	MLP Regressor sj	MLP Regressor iq	MLP Regressor + PCA for sj	MLP Regressor + PCA for iq	ANN + PCA for sj	ANN + PCA for iq
0	MAE	8.195515	3.860350	8.193484	4.472073	8.108014	3.972095	7.924500	3.868660	8.285651	5.002133	8.819729	5.718133	8.948260	5.182581
1	r2_score	0.927074	0.507957	0.923175	0.041135	0.915019	0.350530	0.931454	0.370155	0.929610	-0.139102	0.910156	-0.185822	0.892704	0.031536
2	r2_score_cross_validation	87.787211	54.814627	88.413888	50.931050	87.115586	52.674794	0.957057	0.808096	88.289547	-94.857468	83.602054	-25.167988	0.905680	0.317625

FIGURE 3.12 – Resultats des modeles

3.2.3 Gradient Boosting, XGBoost et Random Forest

Metrics	Gradient boosting for sj	Gradient boosting for iq	XGBoost for sj	XGBoost for iq	Random forest for sj	Random forest for iq
MAE	8.195515	3.860350	8.193484	4.472073	8.108014	3.972095
r2_score	0.927074	0.507957	0.923175	0.041135	0.915019	0.350530
r2_score_cross_validation	87.787211	54.814627	88.413888	50.931050	87.115586	52.674794

FIGURE 3.11 – resultats Gradient Boosting, XGBoost et Random Forest

3.3 Justification des resultats

— Les résultats expérimentaux montrent que les modèles basés sur les réseaux de neurones (ANN, MLP, ANN avec PCA) obtiennent des performances significativement inférieures à celles des modèles ensemblistes, aussi bien pour San Juan que pour Iquitos. Plusieurs facteurs expliquent ces résultats.

Premièrement, la taille du jeu de données reste relativement limitée pour l'entraînement de réseaux de neurones profonds. Ces modèles nécessitent généralement un volume de données important afin de généraliser correctement et éviter le surapprentissage. Dans ce contexte, les réseaux de neurones ont tendance soit à sous-apprendre, soit à sur-apprendre les données d'entraînement.

Deuxièmement, les réseaux de neurones sont sensibles au prétraitement des données, notamment au scaling des variables et au choix de l'architecture. Malgré l'utilisation de PCA dans certaines configurations, la réduction de dimension peut entraîner une perte d'information pertinente, en particulier pour des variables climatiques dont les interactions sont complexes et non linéaires.

Enfin, les réseaux de neurones utilisés ne modélisent pas explicitement la dimension temporelle. Les dépendances temporelles sont uniquement capturées via les variables retardées, ce qui reste moins efficace que des architectures dédiées aux séries temporelles.

— Les modèles ensemblistes, en particulier le Gradient Boosting et le Random Forest, présentent les meilleures performances globales. Pour la ville de San Juan, le Gradient Boosting obtient une erreur MAE plus faible, ce qui s'explique par sa capacité à capturer des relations non linéaires fines et à modéliser efficacement les variations saisonnières marquées observées dans cette ville.

— Pour la ville d'Iquitos, le Random Forest montre une meilleure robustesse. Cette ville présente une variabilité plus irrégulière du nombre de cas et un volume de données plus réduit. Le Random Forest, grâce à l'agrégation de plusieurs arbres

indépendants, est moins sensible au bruit et offre une meilleure stabilité des prédictions dans ce contexte.

Ces résultats confirment que les modèles basés sur les arbres sont particulièrement adaptés aux données climatiques et épidémiologiques, car ils nécessitent peu de prétraitement, gèrent naturellement les interactions entre variables et sont robustes aux distributions hétérogènes.

3.4 Analyse critique

Ce projet met en évidence l'efficacité des modèles ensemblistes comme le Gradient Boosting et le Random Forest pour prédire les cas de dengue, confirmant leur robustesse face à des données climatiques complexes et bruitées.

Cependant, l'utilisation de réseaux de neurones, bien que théoriquement prometteuse, se heurte à la taille limitée des données, particulièrement pour la ville d'Iquitos, ce qui entraîne un surapprentissage et une instabilité des résultats.

La gestion des valeurs manquantes par imputation moyenne, bien que pratique, pourrait introduire un biais, et l'absence de modèles temporels dédiés (comme les LSTM ou ARIMA) limite la capture des dynamiques saisonnières et des dépendances temporelles.

Enfin, une analyse plus approfondie des erreurs et l'intégration de données externes (socio-économiques ou de mobilité) auraient pu enrichir l'interprétation et améliorer la précision des prédictions.

Chapitre 4

Conclusion

4.1 Synthèse du projet

Ce projet a permis de développer un pipeline complet de *Machine Learning* pour prédire le nombre hebdomadaire de cas de dengue dans deux villes tropicales, en exploitant des données climatiques et temporelles. Les modèles ensemblistes (*Gradient Boosting*, *Random Forest*) se sont révélés les plus performants, confirmant leur adéquation pour ce type de problème. Les réseaux de neurones, bien que prometteurs, ont montré des limites en raison de la taille réduite des données et de leur sensibilité au prétraitement.

4.2 Limites

— **Taille des données** : Le volume limité des données, surtout pour la ville d'Iquitos, a restreint la capacité des modèles à généraliser, en particulier pour les réseaux de neurones.

— **Modélisation temporelle** : Les modèles utilisés ne capturent pas explicitement la dynamique temporelle, ce qui pourrait être amélioré par l'utilisation de modèles dédiés aux séries temporelles.

— **Gestion des valeurs manquantes** : La stratégie de remplacement par la moyenne, bien que simple, peut introduire un biais dans les prédictions.

— **Interprétabilité** : Les modèles ensemblistes, bien que performants, sont peu interprétables. Une analyse plus poussée des variables importantes (via *SHAP* ou *LIME*) aurait pu apporter des *insights* supplémentaires sur les facteurs climatiques influençant la propagation de la dengue.

4.3 Pistes d'amélioration

— **Utilisation de modèles temporels** : Intégrer des modèles comme les *LSTM*, *ARIMA*, ou *Prophet* pour mieux capturer les dépendances temporelles et la sai-

sonnalité.

— **Augmentation des données** : Explorer des techniques d'augmentation de données ou de transfert d'apprentissage pour améliorer la généralisation des réseaux de neurones.
— **Optimisation des hyperparamètres** : Approfondir l'optimisation des hyperparamètres, par exemple en utilisant des méthodes bayésiennes ou des algorithmes génétiques.
— **Analyse des erreurs** : Étudier les erreurs de prédiction par période pour identifier des motifs récurrents et ajuster les modèles en conséquence.
— **Intégration de données externes** : Incorporer des données supplémentaires, comme des indicateurs socio-économiques ou des données de mobilité, pour enrichir l'analyse.

Ce projet a démontré l'efficacité des modèles ensemblistes pour prédire les cas de dengue à partir de données climatiques, tout en mettant en lumière les défis liés à la modélisation de séries temporelles complexes avec des données limitées. Les pistes d'amélioration proposées pourraient permettre de renforcer la robustesse et la précision des prédictions.