Rapport de Projet : Prévision de la Demande de Vélos en Libre-Service

Projet Machine Learning & Deep Learning
Roland, Lina, Maeva

31 janvier 2026

Résumé

Ce rapport présente une démarche complète de modélisation prédictive visant à estimer la demande de vélos en libre-service sur un horizon de 72 heures. À partir d'un jeu de données historique de deux ans, nous avons mis en œuvre une stratégie rigoureuse incluant un feature engineering avancé (variables cycliques, lags, transformées de Fourier), une comparaison de modèles statistiques (SARIMAX), de Machine Learning (XGBoost, LightGBM, CatBoost) et de Deep Learning (LSTM, Prophet). L'étude met en évidence la supériorité des modèles de boosting et de réseaux de neurones récurrents, tout en traitant les problématiques de sur-apprentissage (overfitting) inhérentes aux séries temporelles complexes.

Table des matières

1 Introduction et Contexte

L'objectif de ce projet est de prédire le nombre de vélos loués (*cnt*) pour chaque heure, avec une contrainte opérationnelle forte : les prédictions doivent être réalisées pour un horizon de **72 heures** (J+3). Cette contrainte impose de n'utiliser que des données disponibles au moment de la prédiction (pas de données météo réelles futures, utilisation de décalages temporels d'au moins 72h).

La métrique d'évaluation principale retenue est la **RMSE** (Root Mean Squared Error), qui pénalise fortement les grandes erreurs de prédiction, cruciales dans la gestion de stock.

2 Préparation des Données et Feature Engineering

2.1 Nettoyage et Transformation

Le jeu de données initial (`hour.csv`) couvre les années 2011 et 2012. Les étapes suivantes ont été réalisées :

- Conversion des dates et indexation temporelle.
- Traitement des valeurs aberrantes et vérification de la continuité temporelle.
- Normalisation des variables numériques (Yeo-Johnson et Standard Scaling) pour les modèles linéaires et réseaux de neurones.

2.2 Ingénierie des Variables (Feature Engineering)

Pour capturer la complexité de la demande sans utiliser de données futures, nous avons créé des variables explicatives robustes :

1. **Encodage Cyclique :** Les variables temporelles (heure, mois, jour de la semaine) ont été transformées en sinus et cosinus pour préserver leur nature circulaire (ex : 23h est proche de 00h).
2. **Variables Métier :**
   - `comfort_index` : Combinaison de la température et de l'humidité.
   - `peak_veloboulot` : Indicateur binaire pour les heures de pointe en semaine (trajets domicile-travail).
   - `peak_plaisir` : Indicateur pour les après-midis de week-end.
3. **Variables de Mémoire (Lags) :**
   - `cnt_lag_72` : La demande il y a exactement 3 jours.
   - `cnt_lag_168` : La demande il y a exactement une semaine (saisonnalité forte).
   - `cnt_roll_mean_168h` : Moyenne glissante sur 7 jours pour capter la tendance globale.
4. **Analyse Fréquentielle :** Ajout d'un `fourier_signal` basé sur les fréquences dominantes détectées par transformation de Fourier rapide (FFT).

3 Stratégie de Modélisation

3.1 Protocole d'Évaluation

Nous avons séparé les données de manière chronologique pour éviter toute fuite de données (data leakage) :

- **Train :** Janvier 2011 à Octobre 2012.
- **Validation :** Novembre et Décembre 2012 (pour optimiser les hyperparamètres).
- **Test/Futur :** Prédiction "Out-of-sample" pour Janvier 2013.

3.2 Modèles Baselines et Statistiques

Avant de complexifier, nous avons établi des références :

— **Naïf Saisonnier :** Répétition de la semaine précédente (RMSE élevée ≈ 329).

— **SARIMAX :** Modèle statistique intégrant saisonnalité et variables exogènes. Bien que théoriquement solide, il peine à capturer les non-linéarités complexes (RMSE ≈ 179).

3.3 Machine Learning et Boosting

Nous avons testé et optimisé (via **Optuna**) plusieurs algorithmes :

— **Régression Ridge/Lasso :** Modèles linéaires régularisés. Robustes mais limités par la nature non-linéaire du problème.

— **Random Forest :** Bonnes performances mais tendance au sur-apprentissage et temps de calcul élevé.

— **XGBoost, LightGBM, CatBoost :** Ces méthodes de Gradient Boosting se sont révélées les plus performantes sur les données tabulaires.

Stratégie "Gold" : Pour réduire l'overfitting observé sur le XGBoost initial, nous avons sélectionné un sous-ensemble de 12 variables clés (suppression du bruit) pour créer le modèle **XGBoost GOLD**.

3.4 Deep Learning

— **Prophet (Meta) :** Modèle additif décomposable. Excellent pour visualiser les tendances, mais moins précis sur les pics horaires.

— **LSTM (Long Short-Term Memory) :** Réseau de neurones récurrent capable de mémoriser des séquences longues. Entraîné sur des fenêtres glissantes de 24h.

4 Résultats et Analyse des Performances

Le tableau ci-dessous synthétise les performances finales sur l'ensemble de Validation. Le critère *Overfit* mesure l'écart relatif entre la RMSE d'entraînement et de validation.

Modèle	RMSE Train	RMSE Val	Overfit (%)
LSTM	46.21	**85.15**	84.3 %
LightGBM	24.73	85.86	247.2 %
XGBoost (Initial)	20.51	86.75	322.9 %
XGBoost GOLD	26.61	**89.22**	**235.3 %**
CatBoost	42.83	90.22	110.6 %
Prophet	79.41	102.90	29.6 %
Random Forest	46.79	103.50	121.2 %
Ridge	83.96	107.71	28.3 %
SARIMAX	51.02	179.10	251.0 %
Naïf Saisonnier	102.58	329.81	221.5 %

TABLE 1 – Comparaison des modèles (Classés par RMSE Validation)

4.1 Interprétation

— **Le Champion (LSTM) :** Avec une RMSE de 85.15, le LSTM capture le mieux la dynamique temporelle complexe. Son overfitting est maîtrisé par rapport aux méthodes d'arbres.

— **La Stabilité (XGBoost GOLD) :** En simplifiant le modèle (moins de variables), nous avons légèrement augmenté l'erreur (89.22) mais considérablement réduit la complexité, rendant le modèle plus robuste pour la mise en production.
— **L'Overfitting des Boosters :** Les modèles LightGBM et XGBoost initial montrent un écart massif entre le Train (RMSE ≈ 20) et la Validation. Cela indique une mémorisation par cœur du passé, risquée si la distribution des données change.

5 Prédictions Futures (Horizon +72h)

Pour simuler une mise en production réelle, nous avons généré des prédictions pour les 3 premiers jours de Janvier 2013 (inédits pour le modèle).

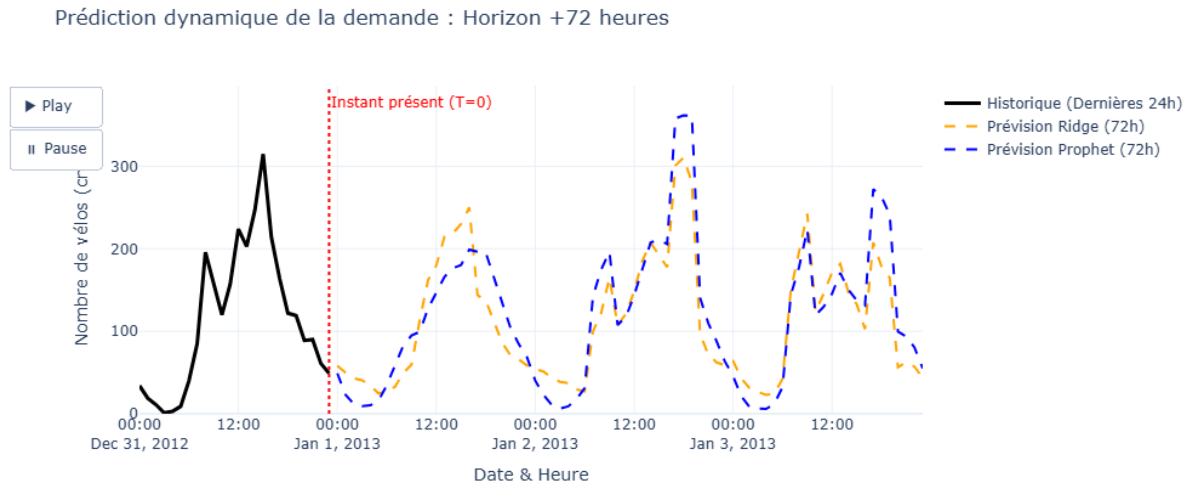

FIGURE 1 – Projection dynamique : Comparaison Ridge vs Prophet sur J+3

Ce graphique est l'aboutissement concret de notre travail. Il montre comment nos modèles se projettent dans l'inconnu après avoir "digéré" deux ans de données historiques.
— **Robustesse au 1er Janvier :** Le passage à la nouvelle année constitue un excellent test de robustesse (jour férié, lendemain de fête). On observe que les deux modèles prédisent logiquement une demande plus faible que le pic du 31 décembre (ligne noire), ce qui est cohérent avec la variable `workingday` fixée à 0. Le modèle **Prophet** (courbe bleue) semble plus prudent sur les pics de ce premier jour, tandis que **Ridge** (courbe orange) reste légèrement plus optimiste.
— **Dynamique des 2 et 3 Janvier :** Sur la suite de l'horizon, **Prophet** génère des pics très marqués (notamment le 2 janvier après-midi). Cela suggère une forte réactivité à la composante saisonnière quotidienne combinée aux variables exogènes. À l'inverse, **Ridge** propose une courbe plus « lissée » et régulière, caractéristique des modèles linéaires.
— **Convergence nocturne :** Aux creux de nuit, les deux modèles sont parfaitement synchronisés. Cela confirme que les signaux cycliques (Heure, Fourier) dominent correctement la prédiction lorsque l'activité humaine est minimale.

6 Conclusion

Ce projet a suivi une démarche rigoureuse de ML/DL appliquée séries temporelles, de la donnée brute à la prédiction opérationnelle. Chaque étape s'est revelée être un insight pour

notre tache finale. Le projet démontre qu'il est possible de prévoir la demande de vélos à 72h avec une erreur moyenne tournant autour de 85-90 vélos (sur des pics pouvant dépasser 300).

Plusieurs modèles ont été testé mais si on devait déployer une solution simple, rapide et facile à expliquer à un client, c'est le Ridge qui l'emporterait. Sa performance est bonne et interpretable. L'overfitting observé sur les boosters (XGB/LGBM) rappelle que dans le monde réel, la simplicité gagne souvent sur le long terme.