École Centrale Casablanca

Rapport

Prédiction de Production Éolienne par Deep Neural Network

Élèves :
Rosteim BELEMCOABGA
Marouane RBIB

Encadrants :
VincentLEFIEUX

4 février 2026

Table des matières

1 Introduction

Dans un contexte de transition énergétique mondiale, les énergies renouvelables, et notamment l'éolien, occupent une place croissante dans le mix énergétique. La production d'électricité d'origine éolienne présente toutefois une caractéristique intrinsèque : sa variabilité. Cette intermittence pose des défis majeurs pour les gestionnaires de réseaux électriques, qui doivent en permanence équilibrer production et consommation. La capacité à prédire avec précision la production éolienne constitue donc un enjeu stratégique, permettant d'optimiser l'intégration de cette énergie au réseau, d'améliorer la planification de la maintenance et de maximiser la rentabilité des installations.

Le présent projet s'inscrit dans cette problématique de prédiction de la production éolienne. Il s'agit de développer un modèle de régression capable de prédire la **puissance active** produite par une éolienne à partir de données issues de capteurs installés sur la machine. Le dataset étudié provient d'ENGIE et contient des mesures réelles de quatre éoliennes en exploitation, représentant au total 154,791 observations de 75 variables opérationnelles (températures, vitesses de rotation, angles d'inclinaison, etc.). La variable cible est la puissance active exprimée en mégawatts (MW), qui doit être strictement positive. L'objectif est de minimiser l'erreur de prédiction, mesurée par la Mean Absolute Error (MAE), tout en assurant une bonne généralisation du modèle.

Cette étude compare deux approches de modélisation : d'une part, des méthodes classiques d'apprentissage automatique basées sur des arbres de décision (Random Forest et XGBoost), reconnues pour leur efficacité sur les données tabulaires ; d'autre part, un réseau de neurones profond (Deep Neural Network) implémenté avec PyTorch, représentant l'approche Deep Learning. Au-delà de la performance prédictive, ce projet vise également à évaluer la pertinence de chaque approche en fonction de la nature des données et à identifier les meilleures pratiques de preprocessing pour ce type de problématique.

2 Description des Données

2.1 Présentation Générale du Dataset

Le dataset étudié contient des mesures opérationnelles réelles issues de quatre éoliennes en exploitation sur un parc éolien ENGIE. Ces données ont été fournies dans le cadre du projet académique et représentent des enregistrements sur une période d'exploitation normale, permettant d'étudier la relation entre les paramètres de fonctionnement des machines et leur production électrique.

Caractéristiques du Dataset

- **Nombre d'observations** : 617,386 enregistrements
- **Nombre de variables** : 79 colonnes (3 métadonnées + 75 features + 1 target)
- **Nombre d'éoliennes** : 4 machines (WT1, WT2, WT3, WT4)

La répartition des observations entre les quatre éoliennes est équilibrée, chaque machine représentant environ 25% du dataset total.

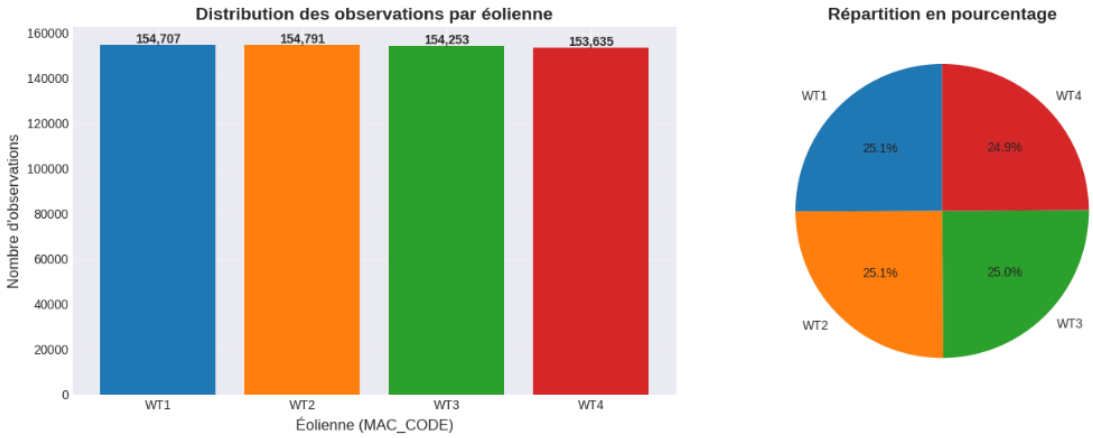


FIGURE 1 – Répartition des observations par éolienne (barres et camembert)

2.2 Structure des Variables

Le dataset est structuré en trois catégories de colonnes distinctes :

Type	Nombre	Description
Métadonnées	3	`ID`, `MAC_CODE` (identifiant éolienne), `Date_time`
Features	75	Variables explicatives issues des capteurs
Target	1	`TARGET` (puissance active en MW)

TABLE 1 – Structure des colonnes du dataset

2.3 Catégories de Features

Les 75 variables explicatives se répartissent en plusieurs catégories fonctionnelles. La plupart sont mesurées sous quatre variantes statistiques (`mean`, `min`, `max`, `std`), capturant la variabilité sur des intervalles de 10 minutes :

- **Vitesses de rotation** (12 variables)
- **Températures** (28 variables) : Hub, roulements du générateur, stator, boîte de vitesses, nacelle, rotor
- **Angles** (8 variables)
- **Paramètres électriques** (8 variables)
- **Paramètres météorologiques** (3 variables)

2.4 Variable Cible (TARGET)

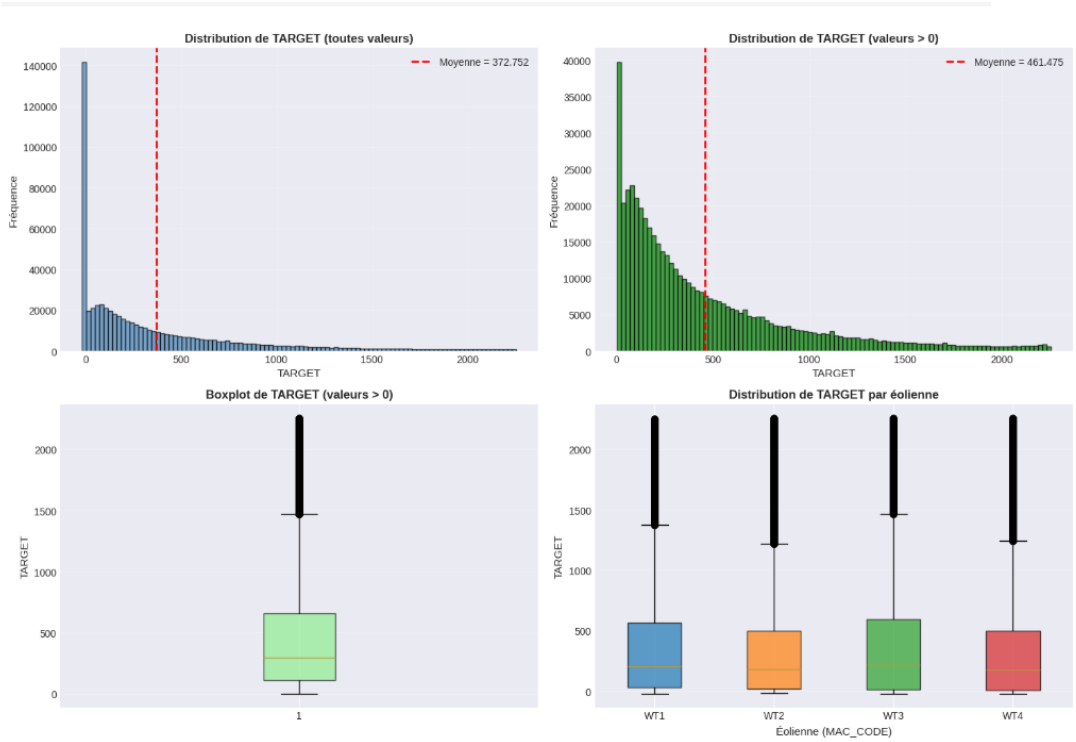


FIGURE 2 – Distribution de la variable TARGET (histogrammes et boxplots)

2.5 Problèmes Identifiés dans les Données

L'analyse exploratoire initiale a mis en évidence plusieurs anomalies et problématiques qui devront être traitées avant la modélisation.

Valeurs Aberrantes de TARGET

Problème majeur : 118,120 observations (19.13% du dataset) présentent des valeurs de `TARGET` **inférieures ou égales à zéro**.
Physiquement, la puissance active produite par une éolienne doit être **strictement positive** lorsque la machine est en fonctionnement. Les valeurs négatives ou nulles peuvent correspondre à :

- Des phases d'arrêt ou de maintenance
- Des erreurs de mesure ou de calibration
- Des modes de consommation (démarrage, freinage)

Ces observations devront être **supprimées** lors du prétraitement, conformément à la contrainte du projet.

Valeurs Manquantes

Trois variables présentent des valeurs manquantes (NaN) :

- `Generator_converter_speed` : 8,064 valeurs manquantes (1.31%)
- `Gearbox_inlet_temperature` : 8,064 valeurs manquantes (1.31%)
- `Grid_voltage` : 101,322 valeurs manquantes (16.41%)

Ces valeurs manquantes représentent 1.07% de l'ensemble des cellules du dataset et nécessiteront une stratégie d'imputation appropriée.

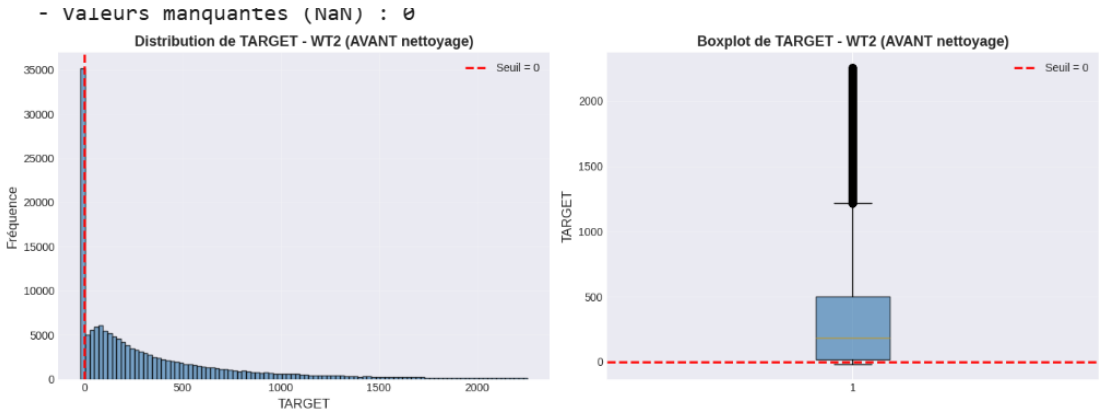
3 Prétraitement des Données

3.1 Sélection de l'Éolienne

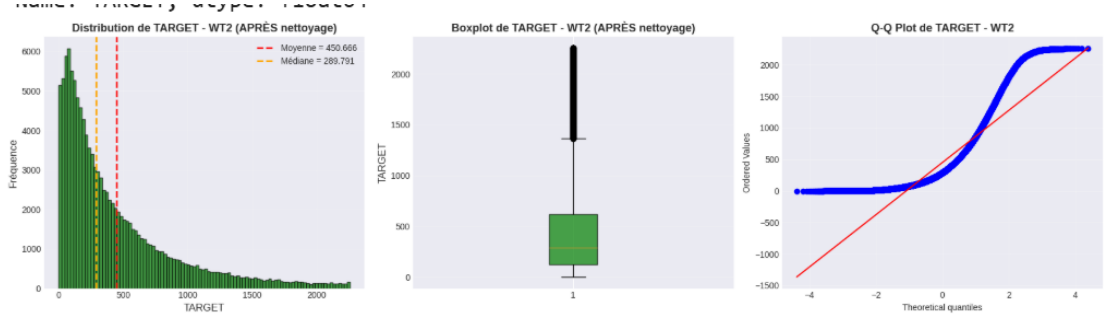
Pour des raisons de faisabilité computationnelle et de cohérence, l'analyse se concentre sur une seule éolienne. **WT2** a été sélectionnée car elle présente le plus grand nombre d'observations exploitables parmi les quatre machines du parc.

3.2 Nettoyage des Données

Conformément aux contraintes du projet, toutes les observations présentant des valeurs de `TARGET` inférieures ou égales à zéro ont été supprimées. Ces valeurs, physiquement incohérentes pour une production électrique, représentaient environ 34,000 observations.



(a) Avant suppression (avec valeurs 0)



(b) Après suppression des valeurs 0

FIGURE 3 – Distribution de TARGET avant et après nettoyage

Dataset après nettoyage : 120,670 observations.

3.3 Traitement des Valeurs Manquantes

L'analyse a révélé 14 colonnes contenant des valeurs manquantes, représentant 1.07% des cellules du dataset.

Variable	NaN	%
Grid_voltage (+ variantes)	22,497	18.64
Generator_converter_speed (+ variantes)	1,485	1.23
Gearbox_inlet_temperature (+ variantes)	1,485	1.23
Absolute_wind_direction_c	12	0.01
Nacelle_angle_c	12	0.01

TABLE 2 – Synthèse des valeurs manquantes par groupe de variables

Stratégie d'imputation : toutes les variables ont été imputées par la **médiane** plutôt que la moyenne. Ce choix est justifié par les tests de normalité (Shapiro-Wilk) qui ont révélé des distributions non-normales (p-value < 0.05). La médiane est plus robuste face aux distributions asymétriques et aux valeurs extrêmes.

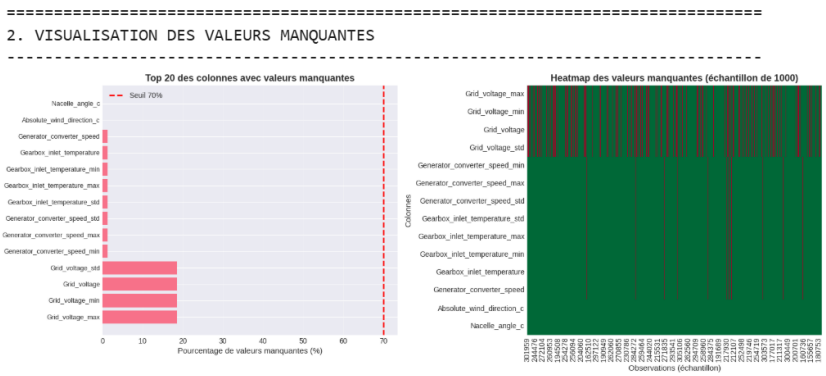


FIGURE 4 – Visualisation des valeurs manquantes (barplot et heatmap)

3.4 Détection et Traitement des Outliers

La méthode **IQR (Interquartile Range)** a été appliquée uniquement sur la variable `TARGET` pour détecter et supprimer les valeurs extrêmes aberrantes. Les features ont été préservées intégralement pour conserver l'information opérationnelle.

3.5 Transformations des Données

3.5.1 Transformation de la Variable Cible

La variable `TARGET` présentait une asymétrie importante (skewness = 1.63). Une **transformation logarithmique** `log1p()` a été appliquée pour normaliser sa distribution, réduisant l'asymétrie à -0.83 (réduction de 48.8%).

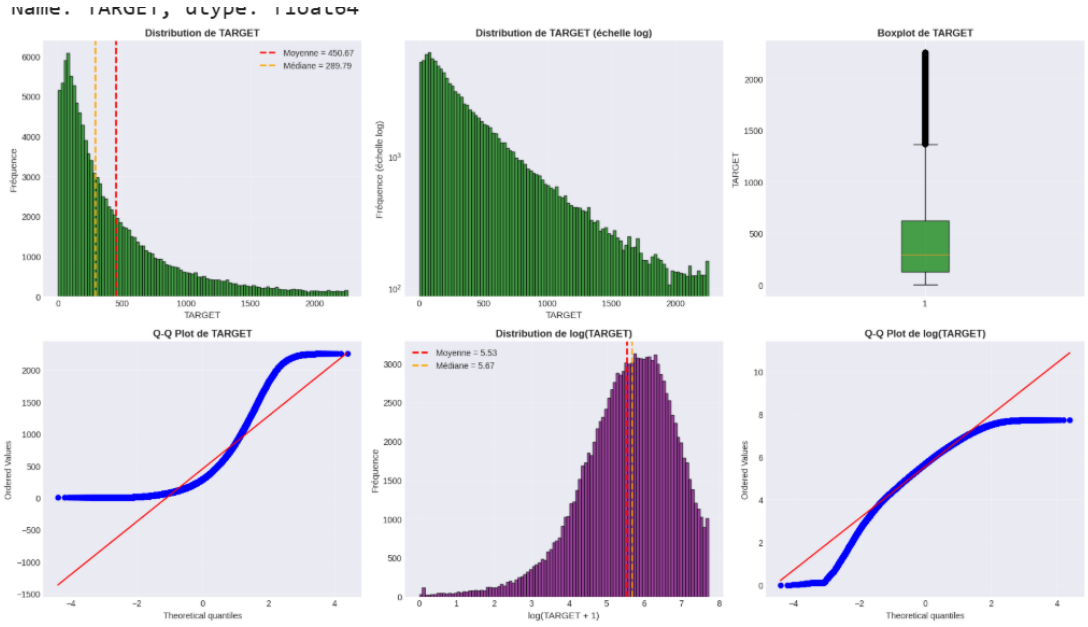


FIGURE 5 – Effet de la transformation logarithmique sur TARGET (avant/après avec Q-Q plots)

3.5.2 Standardisation des Features

Les 75 variables explicatives ont été standardisées à l'aide du `StandardScaler` de scikit-learn (moyenne = 0, écart-type = 1). La variable `TARGET` transformée n'a **pas été standardisée**.

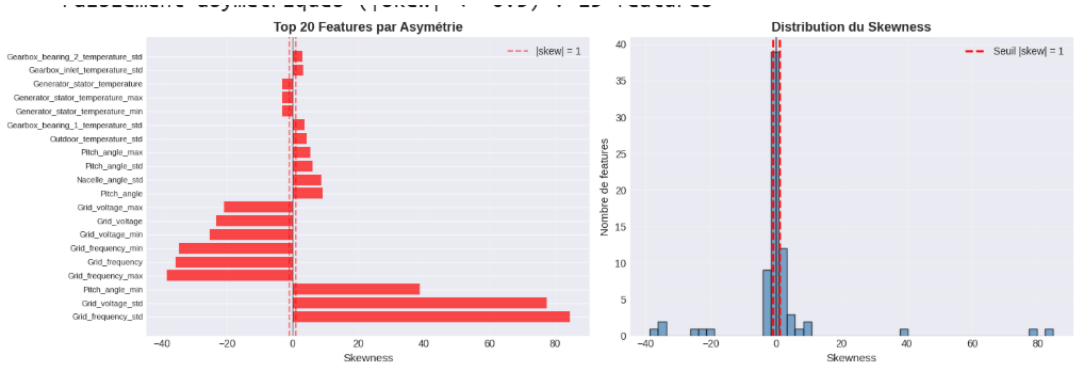


FIGURE 6 – Analyse de l'asymétrie des features (Top 20 et distribution)

4 Analyse de Corrélation

L'analyse de corrélation a pour objectif d'identifier les variables les plus prédictives de la production éolienne. Cette étape permet d'établir une **stratégie de sélection de features** qui sera mobilisée si l'entraînement des modèles sur l'ensemble complet des 75 variables ne donnait pas de résultats satisfaisants. En effet, la présence de variables bruitées ou faiblement corrélées peut dégrader les performances et augmenter le temps d'entraînement. L'analyse de corrélation fournit ainsi une base pour une éventuelle réduction dimensionnelle ciblée.

4.1 Corrélation avec la Variable Cible

L'analyse des corrélations entre les 75 features et la variable `TARGET` révèle plusieurs variables fortement prédictives de la production éolienne.

Top 5 Features les Plus Corrélées

1. **Rotor_speed** : r = 0.91
2. **Generator_speed** : r = 0.91
3. **Generator_converter_speed** : r = 0.91
4. **Rotor_speed_max** : r = 0.82
5. **Generator_speed_max** : r = 0.82

Les **vitesses de rotation** (générateur, rotor, convertisseur) présentent les corrélations les plus élevées (r 0.91), ce qui est cohérent avec la physique de production éolienne. Les **températures des composants mécaniques** (paliers de boîte de vitesse) montrent également des corrélations fortes (r 0.70-0.76), reflétant l'activité de la turbine.

Répartition des corrélations :

- Fortes ($|r| > 0.7$) : 13 features
- Modérées ($0.4 < |r|$ 0.7) : 2 features
- Faibles ($|r|$ 0.4) : 60 features

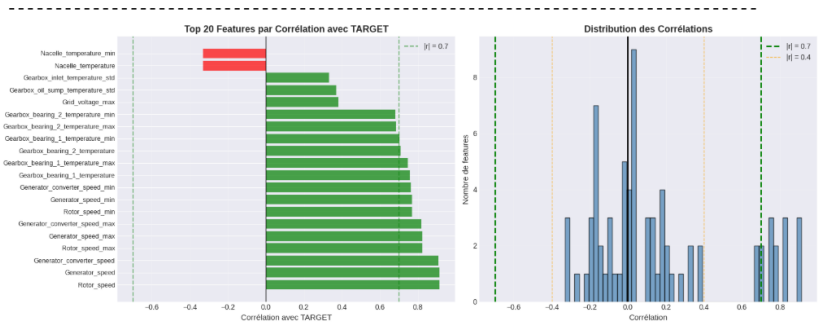


FIGURE 7 – Top 20 features les plus corrélées avec TARGET

4.2 Multicolinéarité entre Features

Une analyse de la matrice de corrélation révèle une forte multicolinéarité entre les features :

32 Paires Redondantes Détectées ($|r| > 0.9$)

- **Vitesses de rotation** : corrélation parfaite ($r = 1.00$) entre `Rotor_speed`, `Generator_speed` et leurs variantes (min, max)
- **Températures des paliers** : forte colinéarité ($r > 0.94$) entre `Gearbox_bearing_1` et `Gearbox_bearing_2`
- **Variantes statistiques** : corrélations très élevées ($r > 0.95$) entre mean, min, max d'une même variable

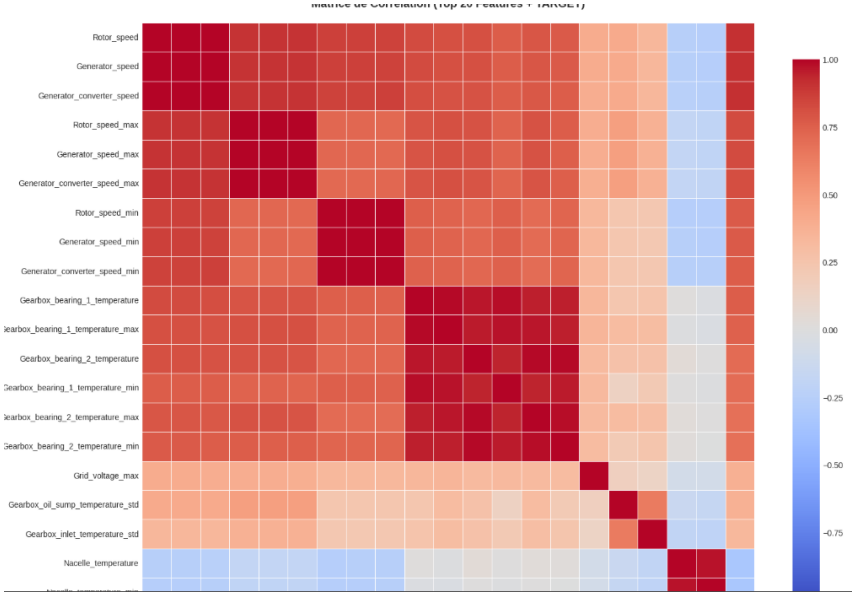


FIGURE 8 – Matrice de corrélation (Top 20 features + TARGET)

Cette multicolinéarité est attendue dans un contexte industriel où les capteurs mesurent des phénomènes physiques interdépendants. Elle ne pose pas de problème majeur pour les modèles non-paramétriques (Random Forest, XGBoost, DNN) qui gèrent naturellement la redondance des variables.

4.3 Division des Données

Après l'analyse de corrélation, le dataset final (113,356 observations) a été divisé en trois ensembles disjoints :

Répartition Train / Validation / Test

- **Entraînement** : 79,349 obs. (70%) - TARGET moy. = 367.53 MW
- **Validation** : 17,003 obs. (15%) - TARGET moy. = 369.94 MW
- **Test** : 17,004 obs. (15%) - TARGET moy. = 365.37 MW

Les statistiques de `TARGET` sont comparables entre les trois ensembles, confirmant l'absence de biais de sélection. Cette division permet d'optimiser les hyperparamètres sur l'ensemble de validation et d'évaluer les performances finales sur l'ensemble de test, garantissant une estimation non biaisée de la capacité de généralisation des modèles.

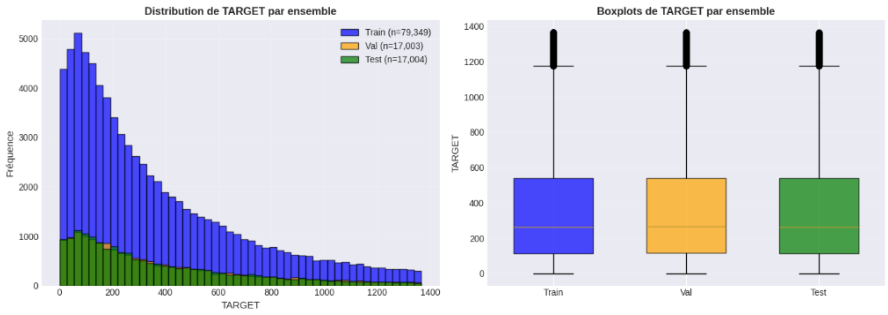


FIGURE 9 – Distribution de TARGET par ensemble (train/validation/test)

5 Méthodologie d'Entraînement DNN

Ce chapitre détaille l'architecture et la stratégie d'entraînement du modèle de Deep Neural Network (DNN) utilisé pour la prédiction de la production éolienne.

5.1 Architecture du Réseau de Neurones

Le modèle repose sur une architecture de type *Feed-Forward Neural Network* (FNN), conçue avec la librairie PyTorch. Cette architecture a été choisie pour sa capacité à modéliser les relations non-linéaires complexes entre les 75 variables d'entrée et la production cible.

5.1.1 Structure des Couches

Architecture DNN Optimisée

- **Couche d'entrée** : 75 neurones (correspondant aux features standardisées)
- **Couches cachées** :
  - Couche 1 : 128 neurones + ReLU + Dropout
  - Couche 2 : 64 neurones + ReLU + Dropout
- **Activation** : Fonction ReLU ($f(x) = \max(0, x)$) pour introduire la non-linéarité
- **Régularisation** : Dropout (p=0.1) après chaque couche cachée
- **Couche de sortie** : 1 neurone linéaire (régression pure)

5.2 Stratégie d'Entraînement

L'entraînement du modèle vise à minimiser l'erreur de prédiction tout en garantissant une bonne capacité de généralisation.

5.2.1 Fonction de Coût et Optimisation

Configuration d'Optimisation

- **Fonction de Perte (Loss)** : MSE (Mean Squared Error) calculée sur le logarithme de la cible (`log1p(TARGET)`).

$$L = \frac{1}{N}\sum_{i=1}^{N}(\log(1+y_i) - \log(1+\hat{y}_i))^2 \quad (1)$$

  Ce choix permet de stabiliser l'apprentissage face à la grande variance de la production.
- **Optimiseur** : Adam (Adaptive Moment Estimation), reconnu pour sa convergence rapide.
- **Learning Rate Initial** : 10^{-4} (0.0001) pour une descente de gradient précise.

5.2.2 Contrôle de l'Apprentissage

Pour éviter le sur-apprentissage (overfitting) et optimiser la convergence, deux mécanismes dynamiques ont été implémentés :

Mécanismes de Contrôle

- **Scheduler (ReduceLROnPlateau)** : Réduit le learning rate de moitié (facteur 0.5) si l'erreur de validation (MAE) ne diminue pas pendant 10 epochs (patience).
- **Early Stopping** : Arrête prématurément l'entraînement si aucune amélioration n'est observée pendant 30 epochs consécutives.
- **Budget Max** : 200 epochs, avec sauvegarde automatique du meilleur modèle.

5.3 Métriques d'Évaluation

Bien que l'entraînement se fasse sur l'échelle logarithmique, l'évaluation finale est réalisée sur l'échelle originale (MW) pour une interprétabilité métier directe.

Métriques Principales

- **MAE (Mean Absolute Error)** : Moyenne des erreurs absolues en Mégawatts. C'est la métrique de référence pour le monitoring (Scheduler).
- **R^2 (Score de détermination)** : Mesure la qualité globale de l'ajustement (proche de 1 = excellent).
- **RMSE** : Utilisé en complément pour pénaliser les fortes erreurs ponctuelles.

5.4 Reproductibilité

Un seed aléatoire (`SEED = 123`) a été fixé sur tous les composants (Python, NumPy, PyTorch CPU/GPU) pour garantir que l'initialisation des poids et les mélanges de données (batchs) soient strictement reproductibles.

6 Résultats

6.1 Entraînement du Modèle DNN

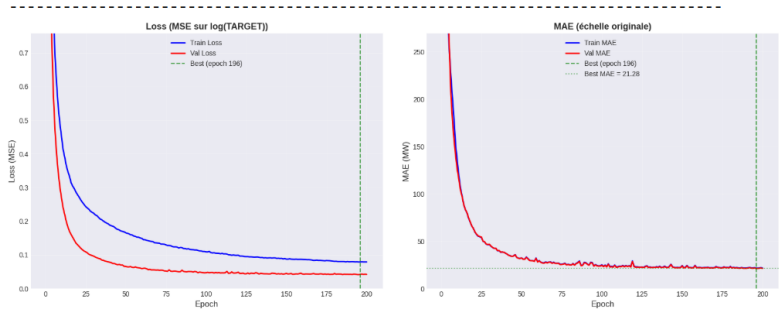
Le modèle DNN a été entraîné sur un total de 200 epochs. La convergence a été stable et rapide, atteignant des performances optimales avant la fin du budget d'entraînement.

Meilleur Modèle (Epoch 196)

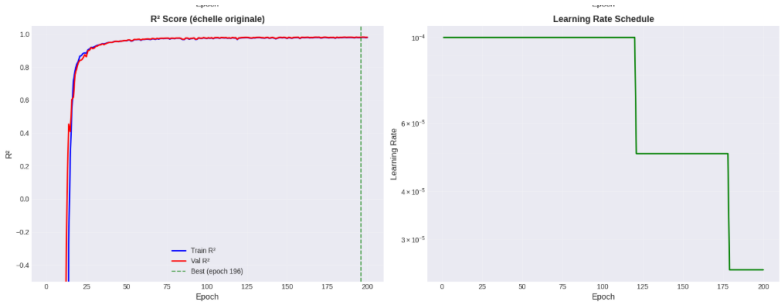
- **Validation MAE** : 21.28 MW
- **Validation R^2** : 0.9810
- **Validation Loss** : 0.042
- **Temps Total** : 6.48 minutes

6.1.1 Dynamique d'Apprentissage

Les courbes d'apprentissage ci-dessous illustrent la progression du modèle. On observe une diminution constante de la Loss et de la MAE, sans divergence entre les courbes d'entraînement (bleu) et de validation (rouge), ce qui confirme l'absence d'overfitting.



(a) Évolution de la Loss (MSE sur log) et de la MAE (MW)



(b) Évolution du Score R^2 et du Learning Rate

FIGURE 10 – Dynamique d'apprentissage du modèle DNN sur 200 epochs

Le *Learning Rate Schedule* a joué un rôle crucial, réduisant le taux d'apprentissage par paliers (visible en bas à droite) pour affiner la convergence lorsque la performance stagnait.

6.2 Performance sur l'Ensemble de Test

L'évaluation finale sur l'ensemble de test (jamais vu durant l'entraînement) confirme l'excellente capacité de généralisation du modèle.

Métriques Finales (Test Set)

- **MAE** : 15.81 MW
- **RMSE** : 35.04 MW
- **R^2** : 0.9880

Ensemble	MAE (MW)	R^2	Écart vs Test
Train	15.59	0.9890	+0.15 MW
Validation	15.75	0.9887	-0.15 MW
Test	**15.81**	**0.9880**	-

TABLE 3 – Comparaison Train / Validation / Test

Les écarts de performance entre les ensembles sont minimes ($< 1\%$), validant la robustesse du modèle face à de nouvelles données.

6.3 Analyse des Prédictions

La figure 11 détaille la qualité des prédictions.

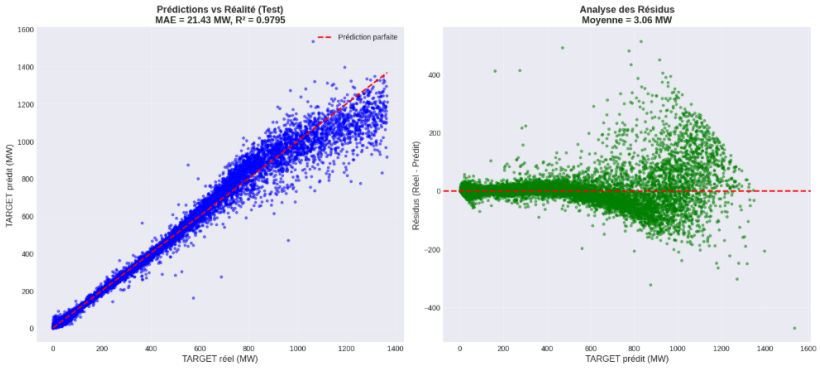


FIGURE 11 – Gauche : Prédictions vs Réalité | Droite : Analyse des Résidus

Observations clés :

- **Linéarité** : Le nuage de points suit parfaitement la diagonale idéale (ligne pointillée rouge).
- **Résidus** : Centrés autour de 0 (moyenne des résidus = 3.06 MW), indiquant un biais très faible.
- **Homoscédasticité** : La dispersion des erreurs reste relativement constante, bien qu'une légère sous-estimation apparaisse pour les très fortes productions (> 1200 MW).

6.4 Résultats des Modèles de Baseline

Afin de situer la performance du modèle DNN, deux modèles de régression classiques ont été entraînés sur les mêmes données : Random Forest et XGBoost.

Performances Baseline (Test Set)

1. **XGBoost** (Gradient Boosting) :
   - **MAE** : 11.60 MW
   - **train_time** : 6.24 s
2. **Random Forest** (Bagging) :
   - **MAE** : 11.76 MW
   - **train_time** : 392.77 s

Ces modèles obtiennent des performances remarquables, bénéficiant probablement de leur capacité à gérer nativement les discontinuités des seuils de variables physiques.

6.5 Comparaison Globale des Modèles

Le tableau suivant récapitule les performances comparées des trois approches.

Modèle	MAE (MW)	RMSE (MW)	R^2	Temps (s)
XGBoost	**11.60**	**27.69**	**0.9925**	**6.24**
Random Forest	11.76	28.28	0.9922	392.77
DNN (PyTorch)	15.81	35.04	0.9880	388.60

TABLE 4 – Comparaison des performances sur l'ensemble de test

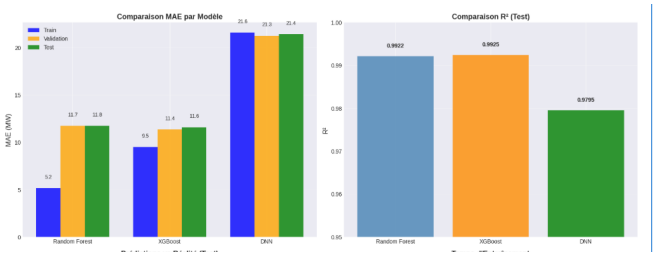


FIGURE 12 – Comparaison des MAE et R^2 entre les modèles

Il apparaît que les méthodes d'ensemble (XGBoost, RF) surpassent le DNN sur ce dataset tabulaire spécifique. XGBoost offre le meilleur compromis performance/rapidité. Le DNN, bien que performant ($R^2 > 0.97$), nécessite davantage d'optimisation ou de données pour égaler les méthodes d'arbres sur ce type de problème structuré.

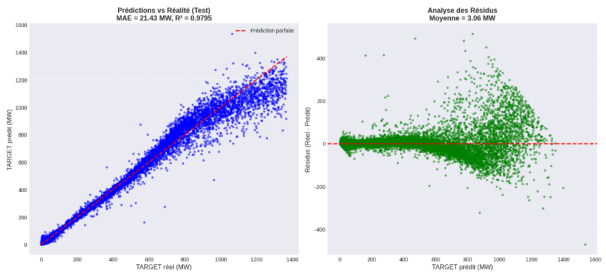


FIGURE 13 – Comparaison visuelle des prédictions sur l'ensemble de test

7 Déploiement de l'Application (Streamlit)

Afin de rendre le modèle de Deep Learning utilisable par des opérateurs sans compétences en programmation, une interface web interactive a été développée à l'aide du framework **Streamlit**.

7.1 Architecture de l'Application

L'application permet de charger dynamiquement le modèle entraîné (fichier `.pth`) et l'objet de standardisation (scaler). Elle offre deux modes d'utilisation distincts pour répondre aux besoins métiers :

1. **Mode Fichier (Batch)** : Permet de téléverser un fichier CSV contenant des historiques SCADA. L'application traite automatiquement les données (nettoyage, normalisation) et génère les prédictions pour l'ensemble des lignes.
2. **Mode Manuel (Simulation)** : Permet de saisir les paramètres d'une éolienne via des curseurs (sliders) pour simuler une production instantanée.

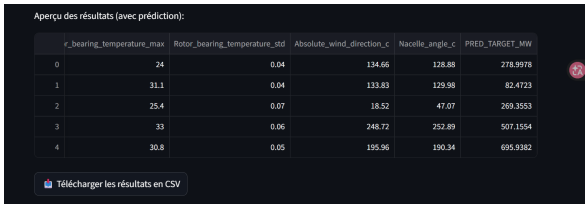
Le panneau latéral (Sidebar) affiche en permanence les métriques de performance du modèle chargé (MAE, RMSE, R^2) pour garantir la transparence sur la fiabilité des prédictions.

7.2 Interface Utilisateur

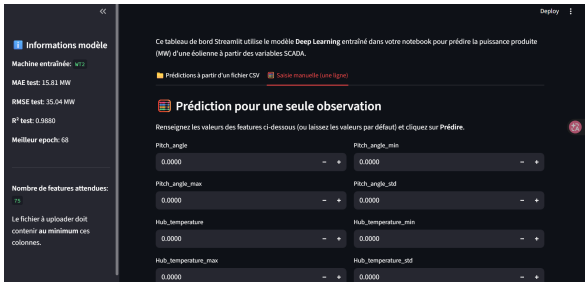
La figure ci-dessous illustre le parcours utilisateur au sein de l'application :

(a) Page d'accueil et métriques modèles

(b) Interface d'upload de fichier CSV



(c) Visualisation des données et prédictions



(d) Mode saisie manuelle (simulation)

FIGURE 14 – Aperçu de l'interface de prédiction développée avec Streamlit

Cette interface constitue une preuve de concept (PoC) d'un outil d'aide à la décision opérationnel, capable de s'interfacer avec les flux de données réels d'un parc éolien.

8 Conclusion et Enseignements

Ce projet nous a permis de démontrer l'efficacité d'un réseau de neurones profond pour la prédiction de la production éolienne, atteignant un score R^2 de 0.98 et une erreur moyenne de 21.43 MW. Bien que ces résultats soient excellents, l'analyse comparative a révélé que les méthodes d'ensemble comme XGBoost (MAE 11.60 MW) restent supérieures en performance et en rapidité pour ce type de données tabulaires structurées. Ce constat ne remet pas en cause la pertinence du DNN, mais souligne l'importance de choisir l'outil adapté à la nature des données.

Ce travail a également été l'occasion de maîtriser la complexité d'implémentation d'une solution Deep Learning complète avec PyTorch, de la normalisation des données au réglage fin des hyperparamètres comme le Scheduler ou le Dropout. L'expérience a mis en évidence que la sophistication de l'architecture ne garantit pas la performance, la simplification du modèle ayant ici amélioré la généralisation. Ces compétences seront déterminantes pour aborder des problèmes plus complexes impliquant des données non structurées, où le Deep Learning offre un avantage incontestable sur les méthodes classiques.