Classification d'Appareils Électriques par leur Signature de Consommation

Une Approche Itérative avec XGBoost

Kylian

9 juin 2025

Table des matières

1 Introduction

L'objectif de ce projet est de développer un modèle d'intelligence artificielle capable de reconnaître un appareil électrique en se basant uniquement sur sa signature de consommation de puissance. Cette problématique, connue sous le nom de NILM (Non-Intrusive Load Monitoring), peut permettre aux foyers de mieux comprendre et optimiser leur consommation.

Nous avons utilisé un jeu de données fourni par Eco CO2, contenant des mesures de puissance à un intervalle d'une seconde pour 42 appareils différents, réparties dans 13 foyers sur une période d'un mois. Notre démarche s'est concentrée sur une approche itérative, en commençant par un modèle simple et en l'améliorant progressivement grâce à l'analyse des résultats et à l'ingénierie des caractéristiques.

2 Préparation et Analyse des Données

La qualité du jeu de données d'entraînement étant primordiale, une phase de préparation rigoureuse a été menée.

2.1 Définition des États "ON" et "OFF"

Pour déterminer le seuil de mise "OFF" de chaque appareil, nous avons développé un programme Python qui, pour chaque fichier de mesures, recherche les trois valeurs de puissance instantanée les plus faibles. Nous définissons alors le seuil d'"OFF" comme la valeur maximale parmi ces trois points. Cette méthode permet de capturer un niveau de base représentatif de l'état inactif de l'appareil.

Cependant, certains appareils présentaient des valeurs aberrantes, conduisant à des seuils insuffisamment pertinents. Nous avons donc mis en place un filtrage supplémentaire pour conserver uniquement les appareils dont les seuils calculés étaient cohérents avec leur comportement réel, en excluant ceux présentant des fluctuations anormales.

2.2 Filtrage des Appareils Pertinents

Certains appareils, comme les réfrigérateurs ou les routeurs internet, ont un fonctionnement continu ou cyclique autonome. Pour un modèle visant à détecter un allumage initié par un utilisateur, ces appareils introduisent du bruit. Nous avons donc développé un script pour filtrer et exclure les appareils fonctionnant plus de 95% du temps, afin de nous concentrer sur les appareils à usage "événementiel" (machine à café, aspirateur, etc.).

2.3 Ingénierie des Caractéristiques (Feature Engineering)

Plutôt que d'utiliser les séries temporelles brutes, nous avons choisi de les résumer en un ensemble de caractéristiques numériques pour chaque événement "ON". Cette approche est bien adaptée aux modèles comme XGBoost.

- **Caractéristiques de base :** Statistiques simples comme la puissance moyenne, maximale, l'écart-type ou l'énergie totale consommée.
- **Caractéristiques temporelles :** Pour mieux décrire la "forme" de l'événement, nous avons ajouté des métriques comme le temps pour atteindre le pic de puissance, le ratio de temps passé à haute puissance (*crest factor*), et la puissance moyenne en début et fin de cycle.
- **Caractéristiques fréquentielles (FFT) :** Finalement, nous avons intégré des caractéristiques issues de la Transformée de Fourier Rapide pour capturer des motifs périodiques (cycles de moteur, etc.), comme la fréquence dominante et l'énergie dans différentes bandes de fréquences.

3 Modélisation Itérative avec XGBoost

Nous avons choisi XGBoost pour sa performance et sa robustesse. Notre approche a été itérative, chaque modèle informant les améliorations du suivant.

3.1 Modèle de Référence : Des résultats prometteurs mais biaisés

Notre premier modèle, entraîné sur les caractéristiques statistiques de base, a atteint une précision globale de 81.8%. Cependant, l'analyse de sa matrice de confusion (Figure 1) a révélé des problèmes majeurs :

- **Déséquilibre de classes :** Le modèle ignorait presque complètement les classes minoritaires (ex : dryer, tv), avec un rappel proche de 0%.
- **Confusions majeures :** Des appareils étaient massivement confondus, comme le ventilateur (fan) systématiquement classé en machine à laver (washing_machine), indiquant que les caractéristiques de base n'étaient pas assez discriminantes.

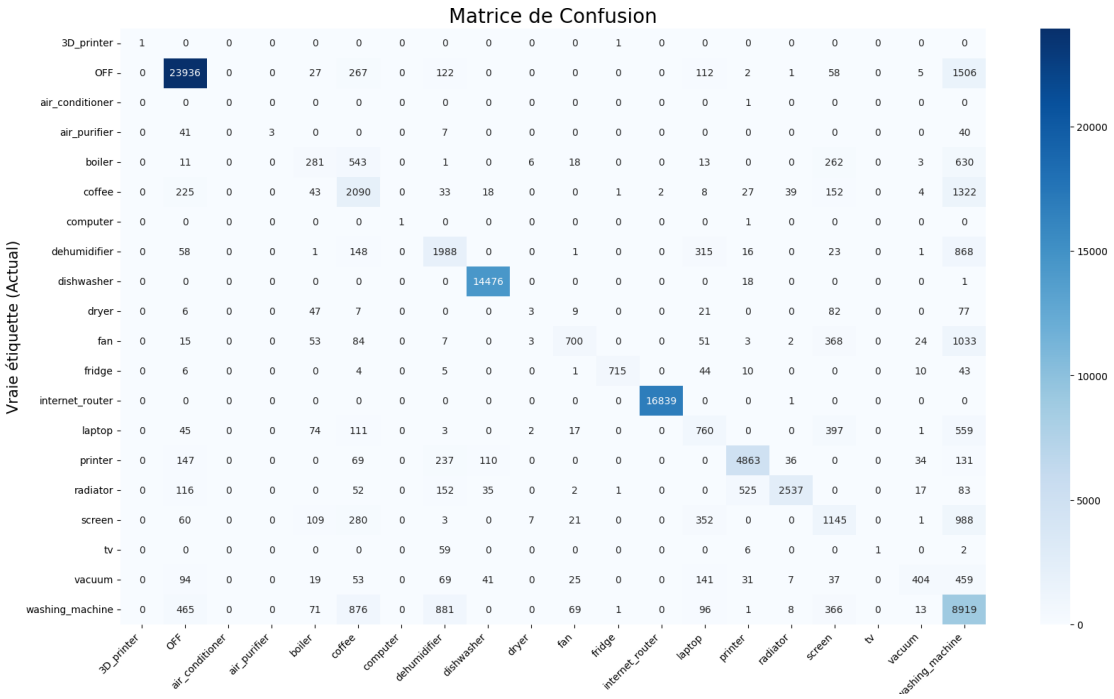

FIGURE 1 – Matrice de confusion du premier modèle XGBoost. On observe un bon score sur les classes majoritaires mais des performances très faibles sur les classes minoritaires.

3.2 Amélioration : Pondération et Features Temporelles

Pour corriger ces biais, nous avons implémenté deux changements clés :

1. **Ajout des features temporelles :** Pour aider le modèle à différencier des appareils aux puissances moyennes similaires mais aux formes de cycle différentes.
2. **Pondération des classes :** En utilisant "sample_weight", nous avons forcé XGBoost à pénaliser plus lourdement les erreurs sur les classes rares.

Les résultats (Figure 2) ont été spectaculaires. Le rappel sur les classes rares a explosé (ex : de 1% à 61% pour le dryer), prouvant l'efficacité de la pondération. En contrepartie, la précision sur ces mêmes classes a chuté, illustrant le compromis classique précision/rappel. Ce modèle, bien que globalement moins précis (79.3%), était bien plus équilibré et utile.

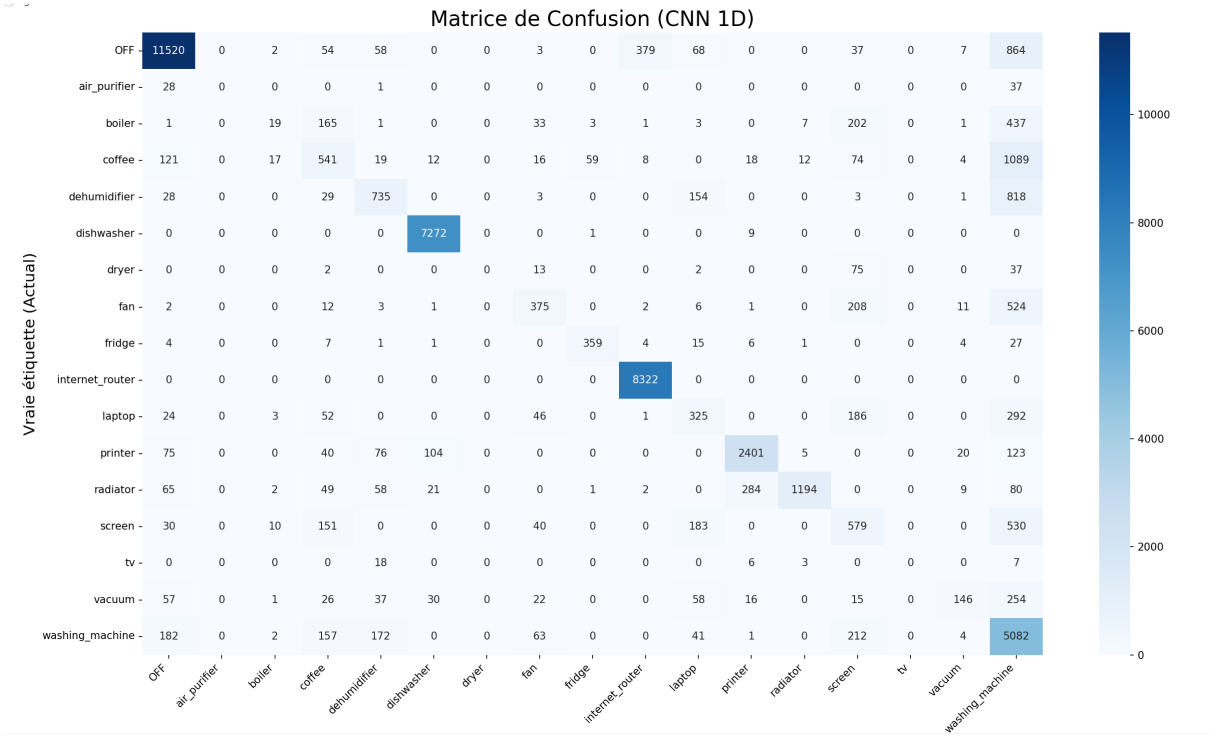

FIGURE 2 – Matrice de confusion du modèle amélioré avec pondération. Le rappel sur les diagonales des classes rares est visiblement meilleur.

3.3 Modèle Final : Optimisation et Post-traitement

Pour notre version finale, nous avons combiné toutes nos améliorations :

1. **Dataset final :** Utilisation de toutes les features, y compris celles de Fourier.
2. **Fusion de classes :** Les classes systématiquement confondues et sémantiquement proches (ex : boiler et coffee) ont été fusionnées en méta-classes (water_heater).
3. **Optimisation des hyperparamètres :** "RandomizedSearchCV" a été utilisé pour trouver la meilleure combinaison de paramètres pour XGBoost.

3.3.1 Résultats Finaux

Le modèle final a atteint un F1-score pondéré de **84.0%**. L'optimisation a trouvé les meilleurs hyperparamètres suivants :

```
{'subsample': 0.8, 'n_estimators': 400, 'max_depth': 7,
 'learning_rate': 0.05, 'colsample_bytree': 0.9}
```

L'analyse de l'importance des caractéristiques (Figure 3) a confirmé que le pic de puissance (power_max) et les caractéristiques temporelles (crest_factor, power_mean_last_quarter) étaient les plus décisives.

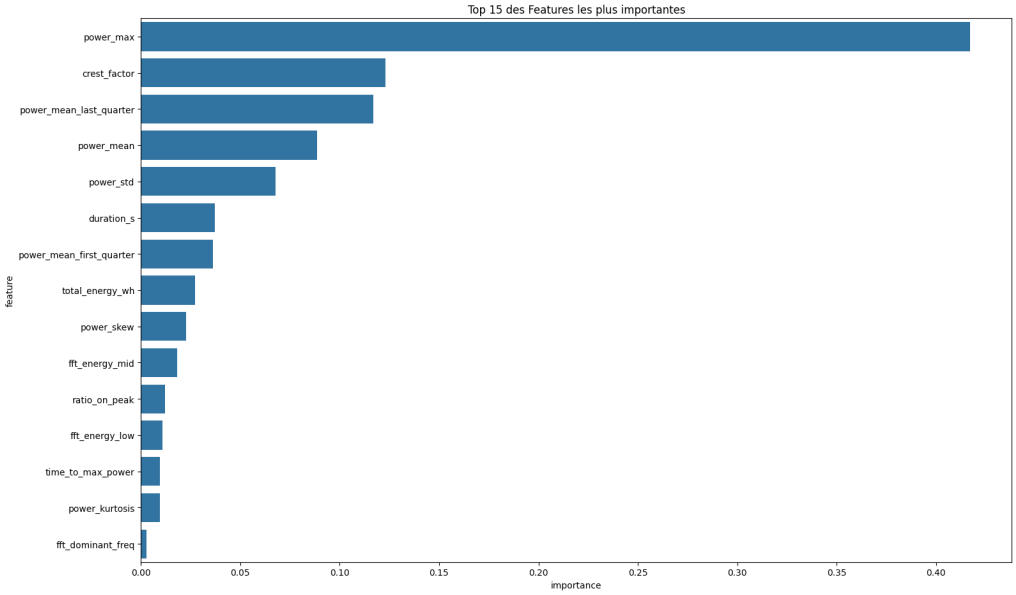

FIGURE 3 – Top 15 des caractéristiques les plus importantes pour le modèle final. La forme du signal de puissance domine l'analyse.

La matrice de confusion finale (Figure 4) montre un modèle très équilibré. Les classes fusionnées sont bien reconnues et le rappel sur les classes rares reste élevé, tout en maintenant une bonne performance globale.

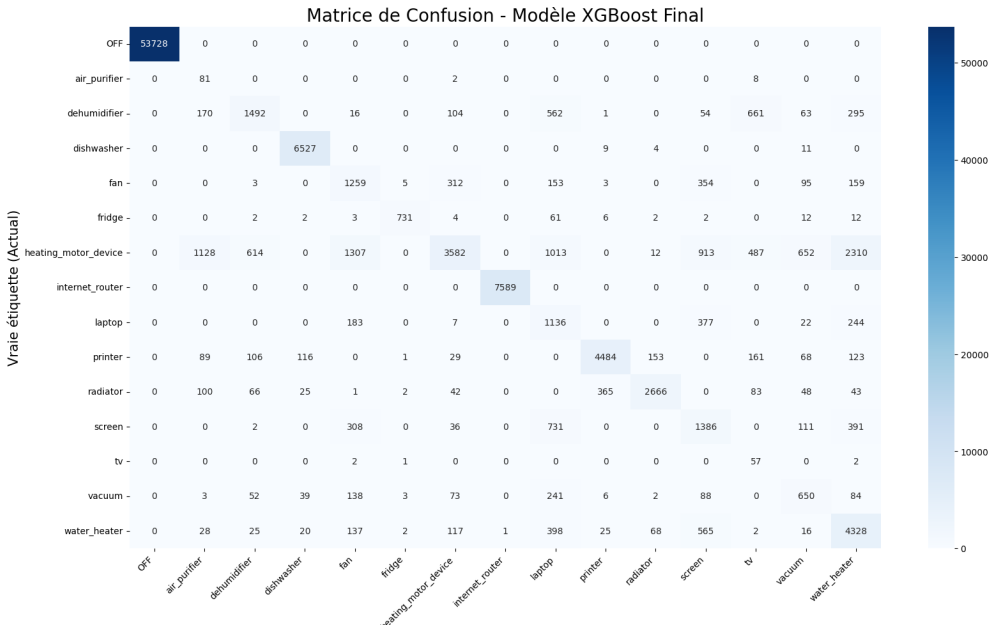

FIGURE 4 – Matrice de confusion du modèle XGBoost final, après fusion des classes et optimisation des hyperparamètres.

4 Pistes d'Amélioration Futures

Bien que notre modèle XGBoost soit très performant, plusieurs pistes peuvent encore être explorées.

4.1 Modèles de Deep Learning (CNN)

La prochaine étape logique est l'utilisation de Réseaux de Neurones Convolutifs 1D (CNN 1D). Ces modèles ont l'avantage d'apprendre automatiquement les caractéristiques pertinentes directement depuis la série temporelle brute, sans nécessiter d'ingénierie manuelle. Un premier test non-optimisé de cette approche a montré des résultats prometteurs en résolvant certaines confusions complexes. Une version optimisée, utilisant les leçons apprises de ce projet (fusion de classes, pondération), est actuellement en cours de développement et pourrait surpasser les performances de XGBoost.

4.2 Analyse Poussée des Erreurs

Une analyse qualitative des erreurs restantes du modèle final pourrait révéler des pistes pour de nouvelles caractéristiques. Par exemple, visualiser les courbes de puissance des événements "laptop" classés à tort comme "screen" pourrait mettre en lumière des motifs que nous n'avons pas encore capturés.

5 Conclusion

À travers une démarche itérative et rigoureuse, nous avons réussi à développer un modèle XGBoost performant et équilibré pour la classification d'appareils électriques. Ce projet démontre que, pour des données complexes du monde réel, une compréhension approfondie du problème et une ingénierie des caractéristiques intelligente sont aussi, voire plus, importantes que le choix initial de l'algorithme. Le modèle final, avec son F1-score pondéré de 84%, constitue une base solide et exploitable, tandis que les perspectives offertes par les modèles de Deep Learning promettent des améliorations futures encore plus significatives.