

AKADEMI Bootcamp 2025
Data Science + AI

Projet Phase 3 **Classification des puits d'eau en Tanzanie**

Préparé par : Saint Germain Emode



Introduction

L'accès à l'eau potable est crucial pour la santé publique. En Tanzanie, certains puits fonctionnent bien, d'autres nécessitent des réparations, et certains sont hors service.

L'objectif de ce projet est de prévoir l'état des puits (functional, needs repair, non functional) à partir des caractéristiques disponibles dans le dataset.





Objectif

L'objectif de ce projet est de créer un classificateur capable de prédire l'état d'un puits d'eau (**status_group**) en utilisant des informations telles que :

Type de pompe - Date d'installation - Localisation GPS - Funder et Installer - Qualité de l'eau et population desservie

Ce classificateur pourra être utilisé par :

- Une ONG souhaitant localiser rapidement les puits nécessitant des réparations
- Le gouvernement tanzanien pour identifier les tendances des puits non fonctionnels et optimiser la construction de nouveaux puits

Approche ML

- Nous commencerons par une **classification ternaire** (fonctionnel / à réparer / hors service), avec la possibilité de transformer le problème en **binaire** pour simplifier la détection des puits à risque.
- Le notebook est structuré en plusieurs étapes : **EDA, nettoyage et préparation des données, modélisation ML, évaluation et insights opérationnels**

Impact attendu

Les résultats permettront d'orienter les décisions stratégiques pour la maintenance et la construction de nouveaux puits d'eau, optimisant ainsi l'accès à l'eau potable pour la population.

Machine Learning

Le Machine Learning permet à un ordinateur d'apprendre à partir des données pour faire des prédictions.

Dans ce projet, nous voulons prédire l'état des puits d'eau en Tanzanie (fonctionnel, besoin de réparation, hors service) à partir de leurs caractéristiques comme le type de pompe, la source d'eau ou la gestion du puits.

Les modèles que je prévois de tester sont:

XGBoost

XGBoost : Méthode de boosting qui entraîne séquentiellement les arbres, chaque nouvel arbre corigeant les erreurs des précédents, avec régularisation pour prévenir le surapprentissage,

Forêts aléatoires

Ensemble d'arbres de décision créé sur des sous-échantillons aléatoires et des sous-ensembles de variables, combinés par vote majoritaire pour améliorer la précision et réduire le surapprentissage.

Arbres de décision

Modèle qui divise les données en branches à partir de questions sur les variables, permettant de pré

KNN

(k plus proches voisins) : Algorithme basé sur les voisins les plus proches dans l'espace des caractéristiques.



Quelques Rapport EDA

Nous avons commencé avec 41 colonnes (y compris la variable cible). Nous avons cartographié les colonnes dans un mindmap selon des concepts similaires et obtenu les catégories suivantes de variables :

- Localisation (9) - Source d'eau (5) - Qualité de l'eau (2)
- Type de pompe (7) - Gestion (6) - Dates (2) - Coût (2)
- Utilisateurs (1)- Administratif (4) - Identité (2) - Cible
=status_group

Je fais l'hypothèse que, pour déterminer si un puits fonctionne ou non, les caractéristiques de la source d'eau et du type de pompe sont plus importantes que certains aspects de la gestion.

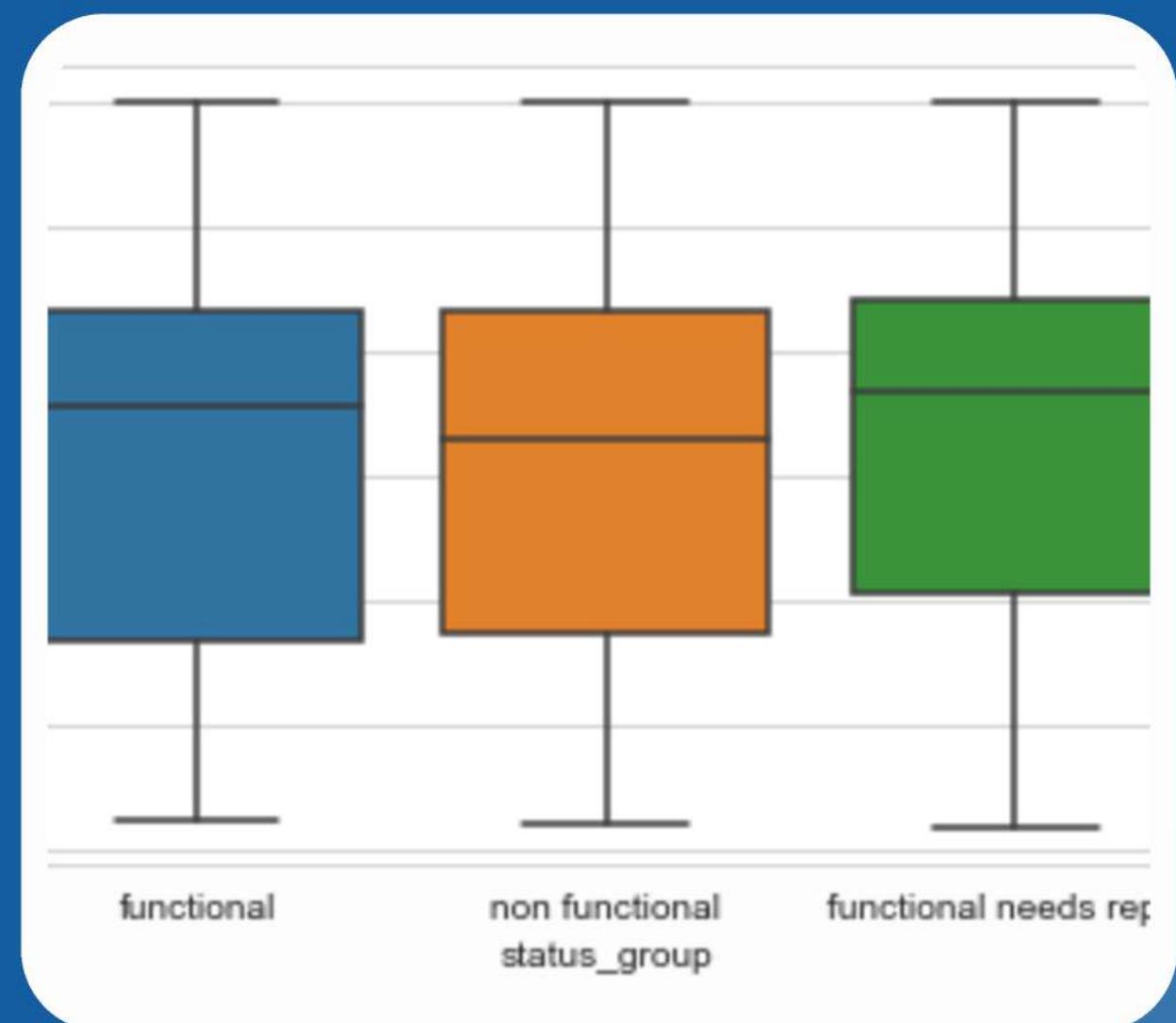
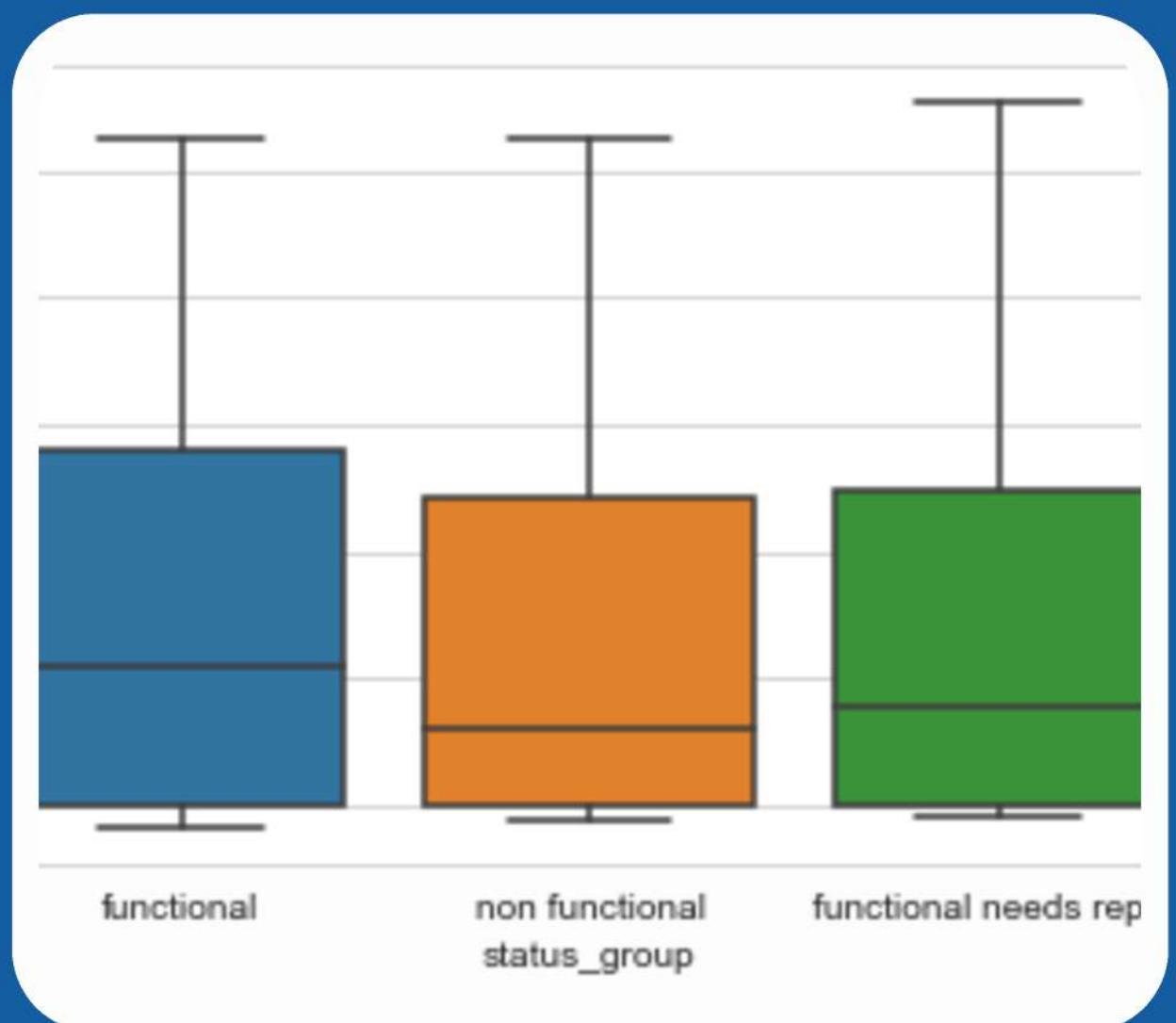
Je vais également conserver quelques variables de localisation, mais pas toutes, car elles pourraient être trop redondantes.

Au total, 22 colonnes ont été identifiées pour suppression.



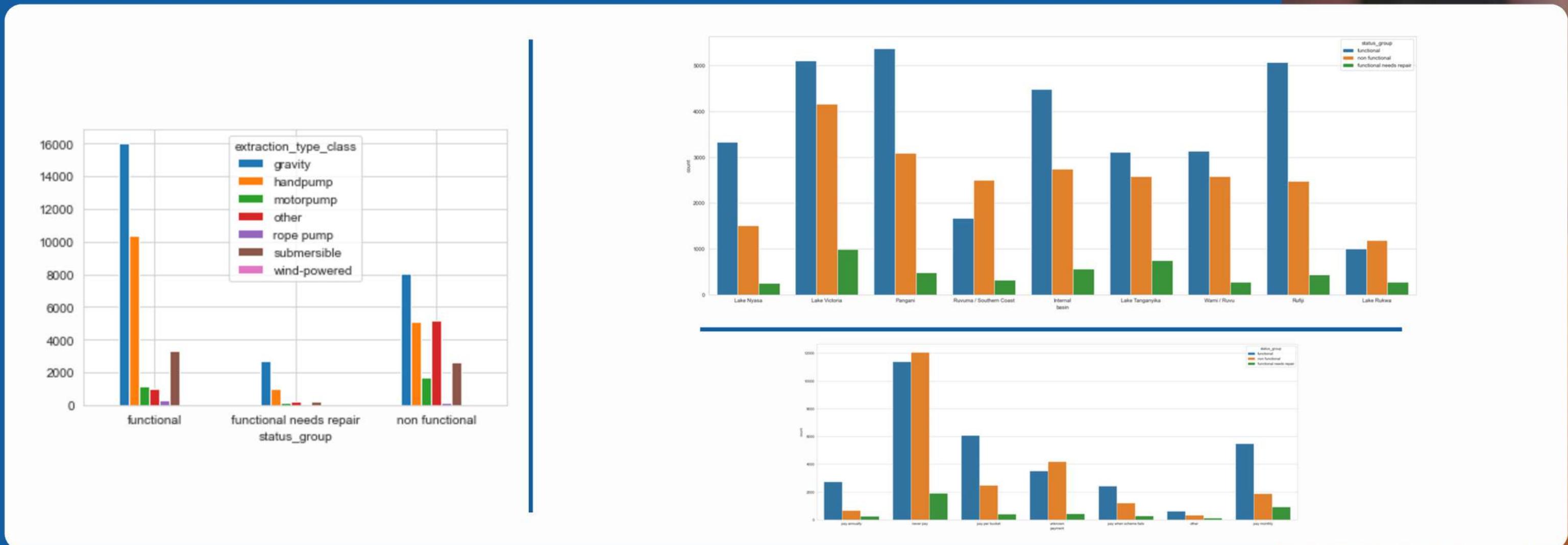
Visualisation initiale des données

Certaines variables numériques comme amount_tsh, gps_height et la localisation géographique montrent des corrélations visibles avec le statut des puits et méritent d'être conservées pour la modélisation. D'autres variables moins informatives pourraient être candidates à la suppression ou à la transformation.



Visualisation

visualiser la répartition de la variable cible status_group (état du puits) selon différentes variables catégorielles importantes (type de pompe, type d'eau, région, mode de paiement, etc.).



Introduction au Machine Learning

Le Machine Learning permet à un ordinateur d'apprendre à partir des données pour faire des prédictions.

Dans ce projet, nous voulons prédire l'état des puits d'eau en Tanzanie (fonctionnel, besoin de réparation, hors service) à partir de leurs caractéristiques comme le type de pompe, la source d'eau ou la gestion du puits.

01

Préparer les données (nettoyage et encodage)

02

Tester plusieurs modèles de classification : Logistic Regression, Random Forest, XGBoost.

03

Évaluer les performances avec Accuracy, Recall et F1-score.

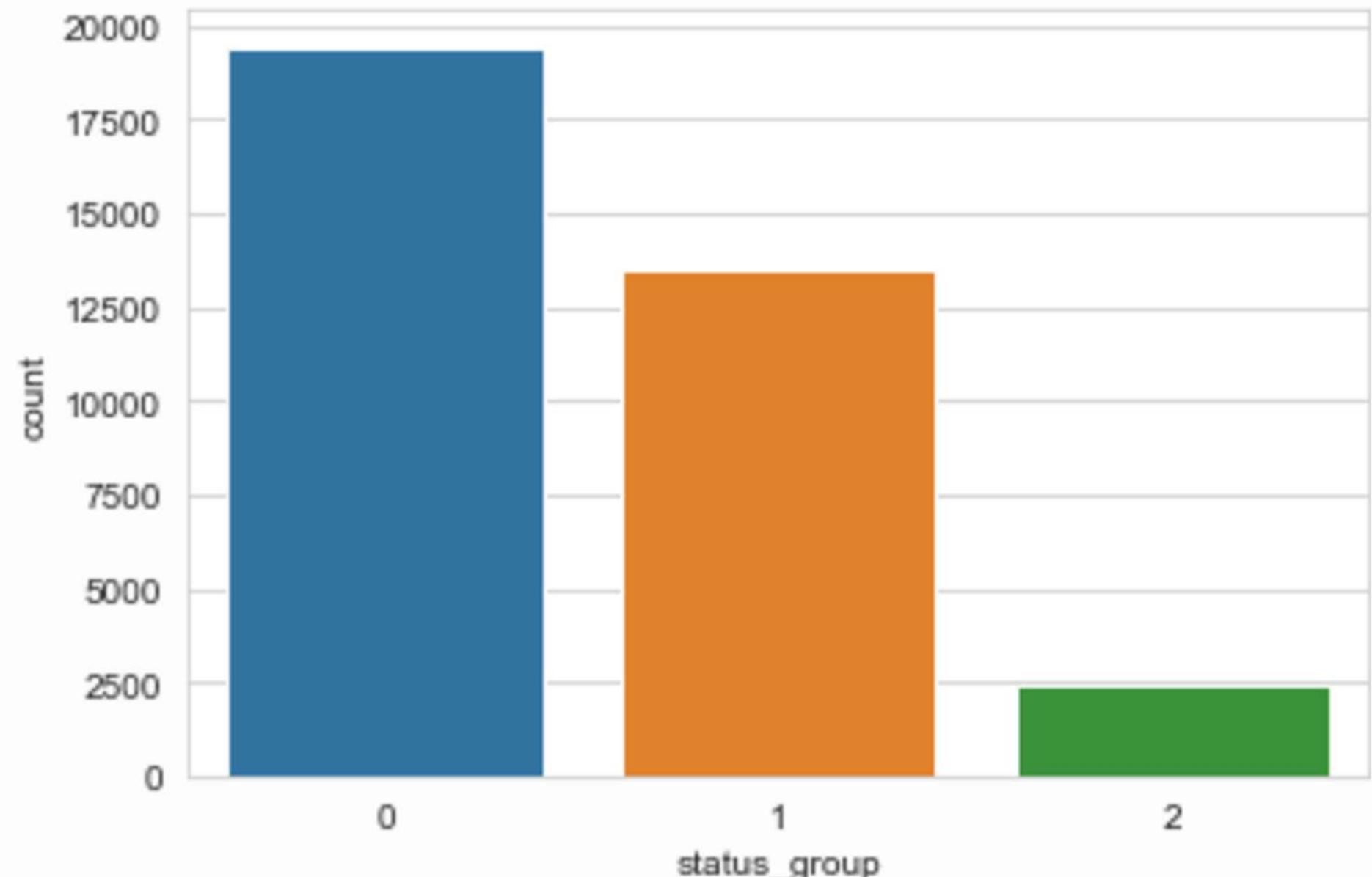
04

Optimiser les modèles avec GridSearchCV.

Nettoyage et préparation des données

Le nettoyage a permis d'éliminer les données aberrantes, de gérer les valeurs manquantes et d'encoder correctement les variables catégorielles. Le jeu de données est maintenant prêt pour l'entraînement de modèles de classification avec des variables fiables et normalisées.

- Base de données: 59 400 lignes × 41 colonnes.
- Colonnes inutiles supprimées → 19 colonnes restantes.
- Valeurs manquantes supprimées → 56 344 lignes.
- Correction des valeurs aberrantes : coordonnées, population, amount_tsh, num_private, année de construction.
- Encodage catégoriel (one-hot) → dataset final 35 303 lignes × 75 colonnes.
- Conversion permit en numérique (0/1).
- Séparation train/test : 26 477 lignes pour l'entraînement, 8 826 pour le test.



Modèle 1 : XGBoost

- Performance globale : Le modèle obtient 82.7 % de précision sur l'entraînement et 74.3 % sur le test, ce qui montre une bonne capacité de généralisation mais un petit surapprentissage.
- Classe 0 (puits fonctionnels) : Très bien prédite (précision $\approx 0.75\text{--}0.82$, rappel élevé $\approx 0.85\text{--}0.91$).
- Classe 1 (puits nécessitant réparation) : Performances correctes (précision $\approx 0.74\text{--}0.84$, rappel $\approx 0.69\text{--}0.78$).
- Classe 2 (puits non fonctionnels) : Faible rappel (0.19 en test) malgré une précision de 0.53, ce qui montre une difficulté à détecter cette classe minoritaire.
- Matrice de confusion : La majorité des erreurs viennent de la classe 2 sous-prédite (ex. beaucoup de puits non fonctionnels classés comme fonctionnels).



modèle (SMOTE + XGBoost)

- Rapport Insight sur le modèle (SMOTE + XGBoost)
- Avant SMOTE : forte déséquilibre (classe 2 sous-représentée avec 1804 vs ~10k-14k pour les autres). Après SMOTE : parfait équilibrage (14587 échantillons par classe). Résultats XGBoost :
- Train accuracy = 88.36% → bon apprentissage
- Test accuracy = 79.05% → bonne généralisation
- Les classes 0 et 1 sont bien prédites (82% de précision/recall), mais la classe 2 reste plus faible (~44% f1-score). Insight : SMOTE améliore l'équilibre global, mais la classe 2 reste difficile à distinguer.



Optimisation du modèle XGBoost

L'optimisation par GridSearchCV a permis d'identifier des paramètres optimaux (taux d'apprentissage, profondeur maximale, poids minimal, sous-échantillonnage).

Résultats du premier tuning :

- Précision entraînement : 86,5 %
- Précision validation : 78,06 %
- Bon équilibre entre précision et rappel, mais encore des erreurs de classification pour certaines classes.

Deuxième tuning :

- Ajustement des paramètres (`learning_rate` ↑, profondeur ↑, `min_child_weight` ↑).
- Résultats plus stables entre entraînement et test, réduisant légèrement le surapprentissage.

En résumé, le modèle atteint une bonne performance globale (~78–80 % sur données de validation), avec un rappel élevé pour la classe majoritaire et des marges d'amélioration possibles pour les classes minoritaires (où des techniques comme SMOTE ou un ajustement de poids de classes pourraient aider).



Modèle 2 : Random Forest

- **Précision globale (accuracy) :**
 - Entraînement : **69,46 %**
 - Test : **69,49 %**
 - → Le modèle généralise correctement, sans surapprentissage.
- **Analyse par classe :**
 - **Classe 0 (puits fonctionnels)** : très bien prédite (rappel = 95 %, f1 = 0,78).
 - **Classe 1 (puits nécessitant réparation)** : modérément captée (précision = 82 %, rappel = 45 %).
 - **Classe 2 (puits non fonctionnels)** : totalement ignorée (f1 = 0,00).
- **Conclusion :**
 - Le modèle Random Forest capture bien les classes majoritaires (surtout la classe 0), mais échoue complètement à identifier la classe minoritaire (2). Cela est dû au **fort déséquilibre du dataset** : la classe 2 est sous-représentée.
 - Une technique comme **SMOTE** ou un **poids de classe équilibré** devrait être envisagée pour améliorer la détection des puits non fonctionnels.



Importance des variables (Random Forest)

- Le graphique d'importance montre quelles variables influencent le plus les prédictions du modèle.
- Les variables situées en haut du classement ont le plus grand impact sur la décision du Random Forest.
- Cela permet d'identifier les facteurs clés qui expliquent l'état des puits (par ex. localisation, type de construction, source d'eau, âge du puits).
- Les variables avec une importance proche de zéro contribuent très peu et pourraient être supprimées ou réduites lors du prétraitement.

En résumé : le graphe met en évidence les caractéristiques déterminantes pour la prédiction et aide à mieux comprendre le comportement du modèle.



Modèle 3 - Arbre de Décision

- Le modèle Decision Tree avec une profondeur maximale de 3 atteint une précision d'environ 67% aussi bien sur l'entraînement que sur le test, ce qui montre une performance stable mais modeste.
- La classe 0 (puits fonctionnels) est la mieux prédite avec un recall de 92%, ce qui signifie que la majorité des puits fonctionnels sont correctement identifiés.
- La classe 1 (puits nécessitant réparation) est prédite avec une précision correcte (0.76) mais un rappel faible (0.44), donc le modèle rate beaucoup de cas de cette classe.
- La classe 2 (puits non fonctionnels) est très mal prédite (precision, recall et f1 = 0.00), le modèle n'arrive pas du tout à la reconnaître. Les métriques macro moyennes (≈ 0.45) confirment un déséquilibre important entre les classes et une faiblesse du modèle pour les catégories minoritaires.
- En résumé : l'arbre de décision capte bien les puits fonctionnels, mais il échoue à distinguer correctement les puits en panne ou hors service. Une amélioration est nécessaire via des modèles plus complexes ou des techniques de rééquilibrage des classes.



Arbres de Décision avec SMOTE pour gérer le déséquilibre des classes

- Performance globale : le modèle atteint 61.5% d'accuracy en entraînement et 62% en test, ce qui montre une généralisation correcte sans surapprentissage marqué.
- Classe 0 (puits fonctionnels) : bien prédite (Précision et Recall ~74%), le modèle identifie correctement la majorité des puits fonctionnels.
- Classe 1 (puits réparables) : performance moyenne (Précision 0.69, Recall 0.49), avec beaucoup de cas encore mal classés.
- Classe 2 (puits non fonctionnels) : très faible performance (Précision 0.14, Recall 0.38), le modèle confond souvent cette classe avec les deux autres.
- Forces : bonne capacité de détection des puits fonctionnels.
- Faiblesses : difficulté majeure à prédire correctement les puits non fonctionnels (classe minoritaire).



Modèle 4 – KNN (K-Nearest Neighbors)

Le modèle KNN normalisé obtient 82,99% de précision sur l'entraînement et 76,97% sur le test, montrant un léger surapprentissage mais des performances globales correctes. Il détecte très bien les puits fonctionnels (classe 0), identifie assez bien les puits réparables (classe 1), mais a des difficultés à prédire les puits non fonctionnels (classe 2), qui sont souvent mal classés. Ces résultats suggèrent que KNN est efficace pour les classes majoritaires mais moins pour les classes minoritaires, et que l'utilisation d'un rééquilibrage des classes (SMOTE) ou de modèles plus robustes (Random Forest, XGBoost) pourrait améliorer la détection des puits cassés.



AKADEMI Bootcamp 2025
Data Science + AI

MERCI



Saint Germain emode

Data Scientist



+509 44 09 4159



germodee12@gmail.com

