



Projet 4 - Anticipez les besoins en consommation de bâtiments

Prédire les émissions de CO2 et la consommation totale d'énergie de bâtiments non destinés à l'habitation pour la ville de Seattle sur la base de bâtiments pour lesquelles on les a mesurées.



Seattle

Sommaire



La problématique & Le jeu de données (analyse exploratoire)



Le Feature Engineering



Approche de modélisation



Choix du modèle



Intérêt de l'Energy Star Score

puis remarques & axes d'amélioration.

Problématique & Données





La problématique et le jeu de données



Seattle



Contexte : Seattle objectif ville neutre en émissions carbone en 2050. Travail au sein de l'équipe bâtiments non destinés à l'habitation.

Objectif : Prédire les émissions de CO₂ et la consommation totale d'énergie de ces bâtiments.

Moyens : Jeu de données de consommation de bâtiments de la ville de Seattle

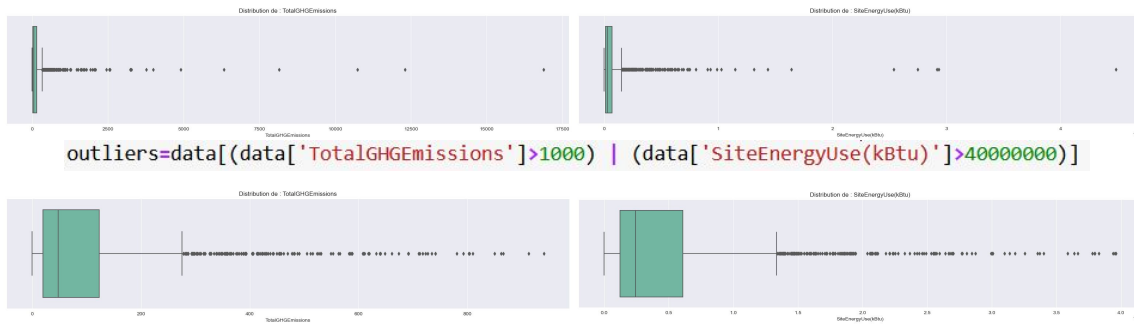
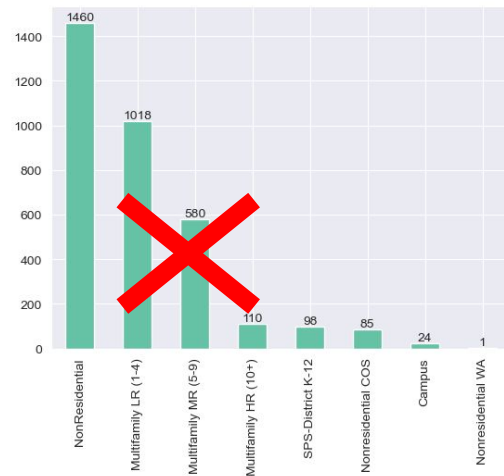



La problématique et le jeu de données

Nettoyage

3376 bâtiments & 46 variables

- Suppression des bâtiments destinés à l'habitation, des colonnes doublons avec unités différentes
- Suppression des valeurs négatives pour les émissions / consommations
- Numbers of Buildings & Number of floors NaN ou 0 \Rightarrow 1
- Création d'un dictionnaire pour les types d'utilisation des bâtiments, les faisant passer de 70 types différents à une vingtaine
- Remplacer les NaN par des unknown pour les types d'utilisation, 0 pour les surfaces
- Travail sur la distribution des valeurs pour les variables à prédire

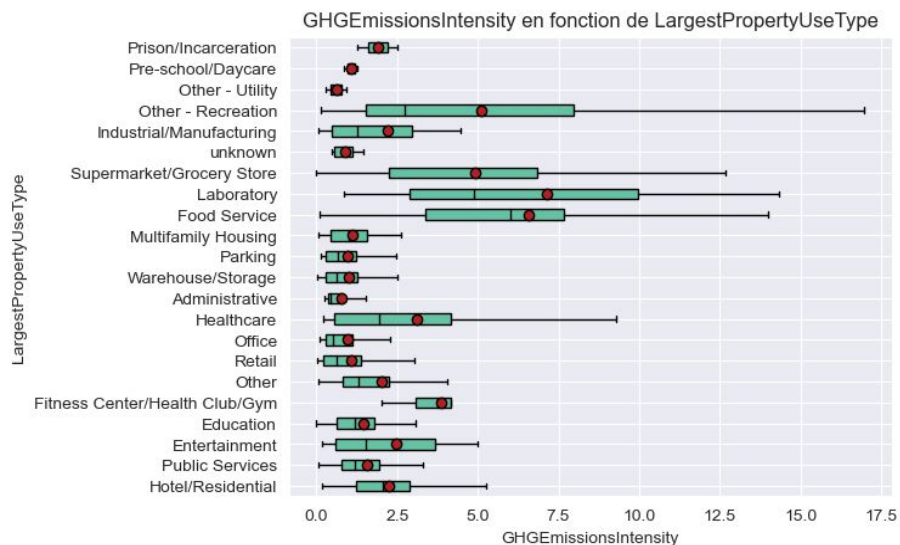


3376  1561
- 54%

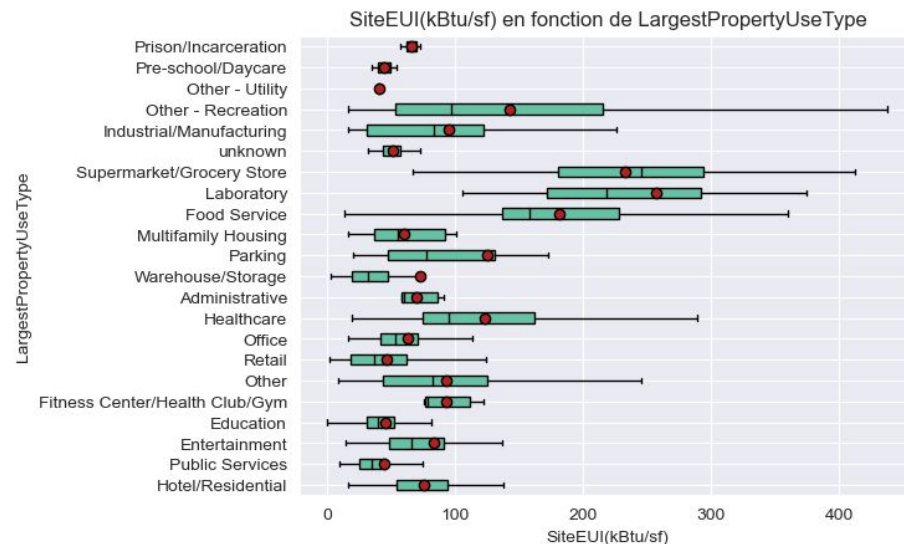


La problématique et le jeu de données

Anova - Boxplot : Mise en évidence de l'importance des Property Use Type

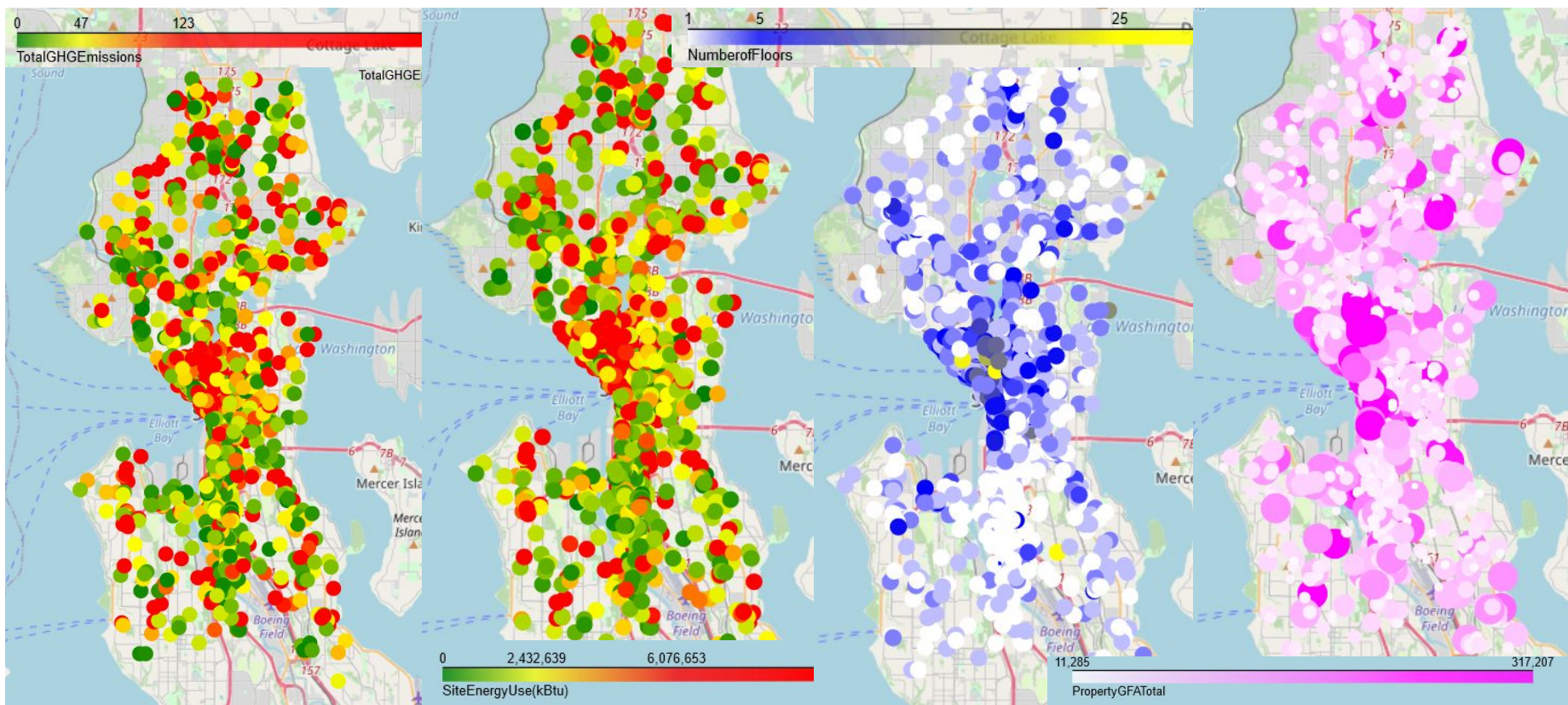


Eta-Squared 0,249

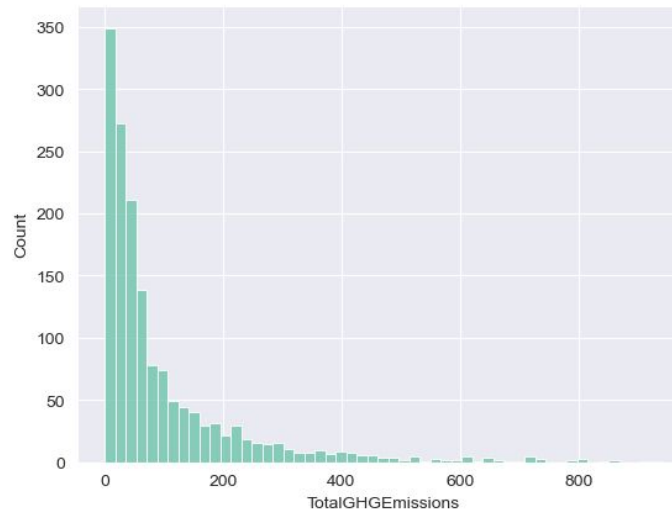
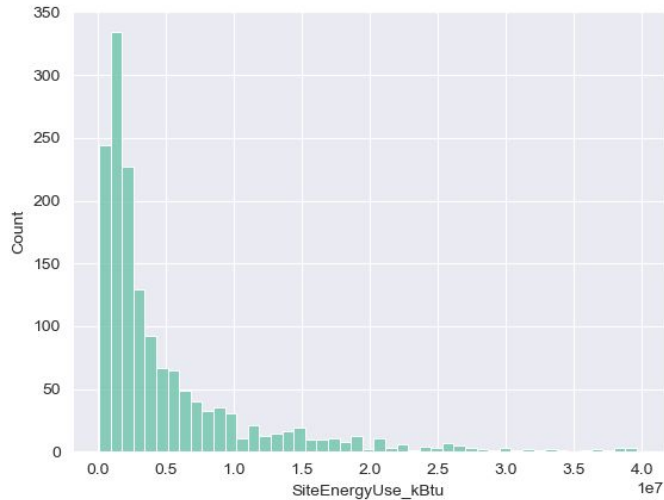


Eta-Squared 0,295

La problématique et le jeu de données - Visualisations



La problématique et le jeu de données



on veut tester
son intérêt

Mesures = à ne
pas utiliser (fuite
données)

Ce que l'on
veut prédire

Types d'utilisation
bâtiments

Localisation

Outlier
ThirdLargestPropertyUseType
ThirdLargestPropertyUseTypeGFA
SecondLargestPropertyUseTypeGFA
SecondLargestPropertyUseType
ENERGYSTARScore
LargestPropertyUseTypeGFA
LargestPropertyUseType
SiteEUIWN(kBtu/sf)
SiteEnergyUseWN(kBtu)
SiteEUI(kBtu/sf)
SourceEUI(kBtu/sf)
SiteEnergyUse(kBtu)
SteamUse(kBtu)
Electricity(kBtu)
NaturalGas(kBtu)
SourceEUIWN(kBtu/sf)
GHGEmissionsIntensity
TotalGHGEmissions
ListOfAllPropertyUseTypes
Longitude
PropertyGFABuilding(s)
PropertyGFAParking
PropertyGFATotal
NumberOfFloors
NumberOfBuildings
YearBuilt
Neighborhood
CouncilDistrictCode
Address
PropertyName
PrimaryPropertyType
BuildingType
Latitude
OSEBuildingID

Feature Engineering





Le Feature Engineering

Base

- Suppression des variables de localisation autre que longitude et latitude
- YearBuilt \Rightarrow BuildingAge
- Binarisation de l'utilisation des énergies (pour le risque de fuite de données)

```
data['ElectricityUse'] = data['Electricity(kBtu)'].apply(lambda x: 0 if x == 0 else 1)
data['SteamUse'] = data['SteamUse(kBtu)'].apply(lambda x: 0 if x == 0 else 1)
data['NaturalGasUse'] = data['NaturalGas(kBtu)'].apply(lambda x: 0 if x == 0 else 1)
```

- Primary Use Type encodage de type One Hot

Test 1

- Number of Energies
- One Hot Encoding pour Largest, Second & Third Largest

96 variables au total

Test 2

- Une colonne par type d'utilisation avec la surface correspondante dedans

50 variables au total

+ Min Max Scaler

on veut tester son intérêt

Mesures = à ne pas utiliser (fuite données)

Ce que l'on veut prédire

Types d'utilisation bâtiments

Localisation

✗ Outlier

ThirdLargestPropertyUseType

ThirdLargestPropertyUseTypeGFA

SecondLargestPropertyUseTypeGFA

SecondLargestPropertyUseType

ENERGYSTARScore

✗ LargestPropertyUseTypeGFA

LargestPropertyUseType

SiteEUIWN(kBtu/sf)

SiteEnergyUseWN(kBtu)

SiteEUI(kBtu/sf)

SourceEUI(kBtu/sf)

SiteEnergyUse(kBtu)

SteamUse(kBtu)

Electricity(kBtu)

NaturalGas(kBtu)

SourceEUIWN(kBtu/sf)

GHGEmissionsIntensity

TotalGHGEmissions

✗ ListOfAllPropertyUseTypes

Longitude

✗ PropertyGFABuilding(s)

PropertyGFAParking

PropertyGFATotal

NumberOfFloors

NumberOfBuildings

YearBuilt

Neighborhood

CouncilDistrictCode

Address

✗ PropertyName

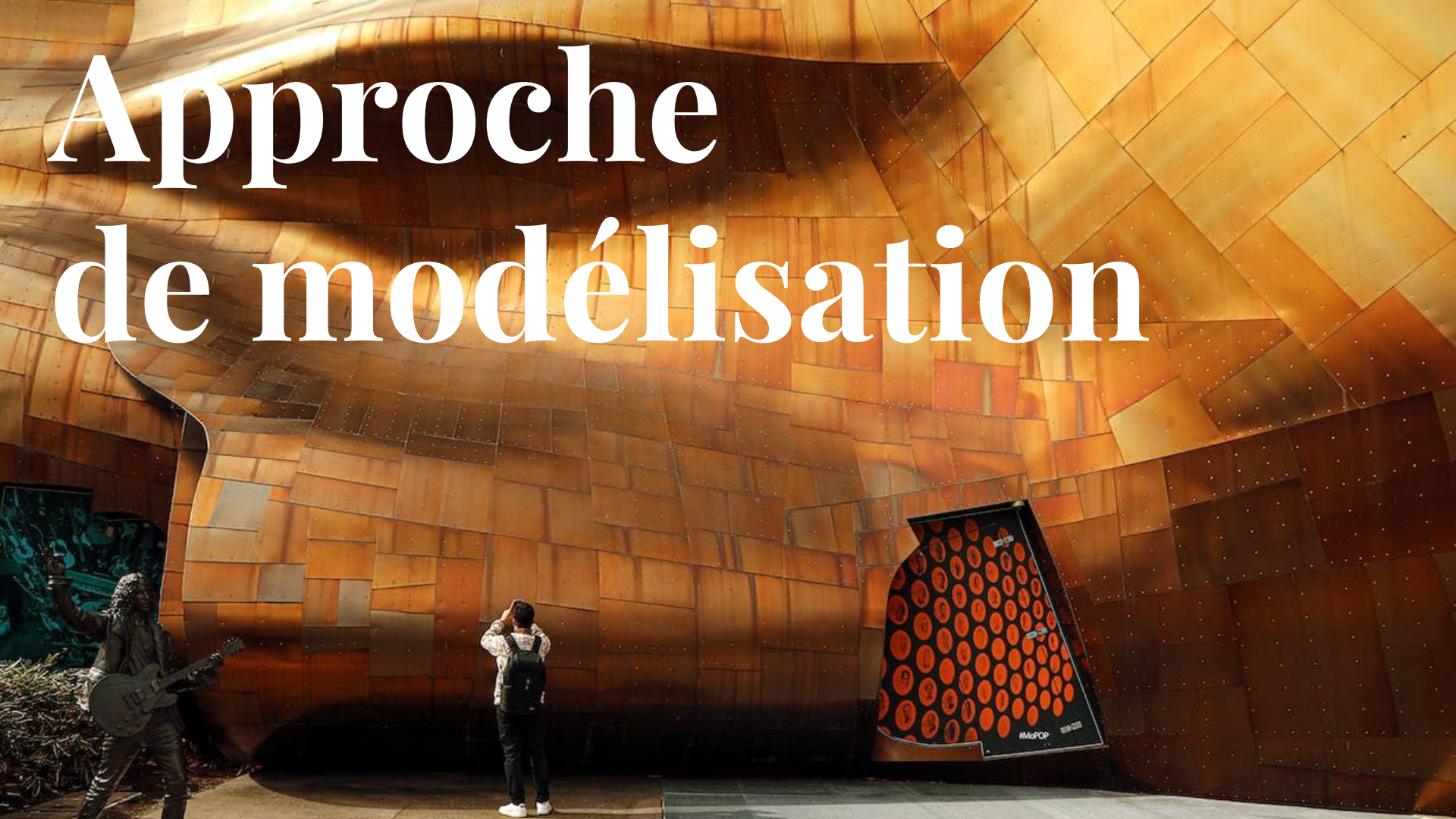
PrimaryPropertyType

✗ BuildingType

Latitude

✗ OSEBuildingID

Approche de modélisation

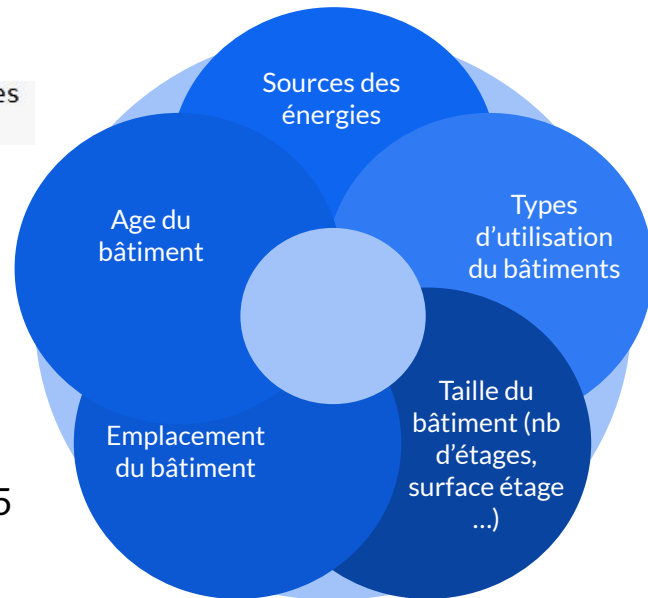




Approche de modélisation

- Utilisation de mon deuxième feature engineering (50 variables)
- Variables à prédire :

```
y_energy=data.copy()['SiteEnergyUse_kBtu'].values  
y_ghg=data.copy()['TotalGHGEmissions'].values
```
- Un notebook par prédiction
- Mise de côté de l'Energy Star Score
- Comparaison des modèles via les scores : Root Mean Squared Error (+normalized), R2 score
- Séparation en jeux d'entraînement et test via Cross Validation à 5 folds, 3 répétitions



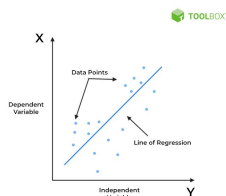


Approche de modélisation

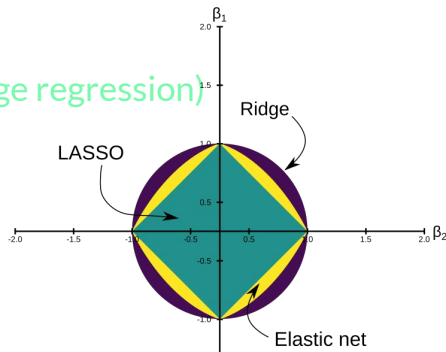
Méthodes testées pour modéliser :

Dummy Regressor (mean) / Naïf

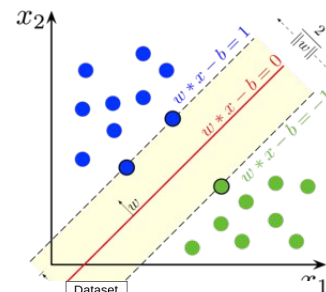
Linear Regression



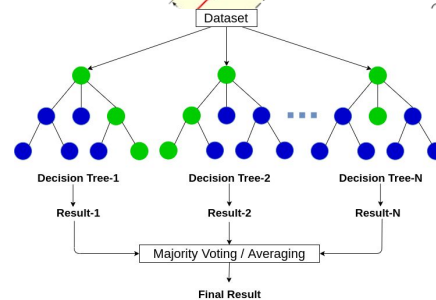
Elastic Net
(Lasso / Ridge regression)



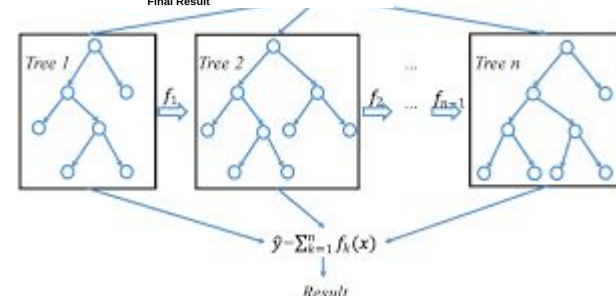
Séparateur à Vaste Marge



Random Forest
(parallèle)



XGBoost
(séquentielle)



Méthodes ENSEMBLISTES



Approche de modélisation

Recherche des hyperparamètres

```
1 ##### on cherche les meilleures valeurs alpha et l1_ratio pour l'elastic net
2
3
4 ##### define model evaluation method
5 cv = RepeatedKfold(n_splits=5, n_repeats=3, random_state=1)
6 ##### define model
7 ratios = arange(0, 1, 0.025)
8 alphas = [1e-4, 1e-3, 1e-2, 0.1, 1]
9 max_iters=[10000]
10
11 ##### Define the parameter grid to search
12 param_grid = {
13     'alpha': alphas,
14     'l1_ratio': ratios,
15     'max_iter' : max_iters
16 }
17
18 ##### Create the GridSearchCV object
19 grid_search = GridSearchCV(estimator=ElasticNet(), param_grid=param_grid, cv=cv, n_jobs=-1)
20
21 ##### Fit the GridSearchCV object to the data
22 grid_search.fit(X, y)
23
24 ##### Print the best hyperparameters
25 print('Best alpha:', grid_search.best_estimator_.alpha)
26 print('Best l1_ratio:', grid_search.best_estimator_.l1_ratio)
27
```




Approche de modélisation

Score du modèle et visualisation

```
1 #evaluer le modele elastic net avec les hyperpar trouvés
2 # define model
3 model = ElasticNet(alpha=0.01, l1_ratio=0.925)
4 # define model evaluation method
5 cv = RepeatedKFold(n_splits=5, n_repeats=3)
6 # evaluate model RMSE
7 scoresRMSE = cross_val_score(model, X, y, scoring='neg_root_mean_squared_error', cv=cv, n_jobs=-1)
8 scoresRMSE = np.absolute(scoresRMSE)
9 RMSE_model=np.mean(scoresRMSE)
10 print('Mean RMSE: %.3f (%.3f)' % (np.mean(scoresRMSE), np.std(scoresRMSE)))
11 # evaluate model R2
12 scoresR2 = cross_val_score(model, X, y, scoring='r2', cv=cv, n_jobs=-1)
13 scoresR2 = np.absolute(scoresR2)
14 R2_model=np.mean(scoresR2)
15 print('Mean R2: %.3f (%.3f)' % (np.mean(scoresR2), np.std(scoresR2)))
16
17 # prediction via cross val
18 y_pred = cross_val_predict(model, X, y, cv=5)
19
20 # Tracer un graphique de dispersion
21 plt.scatter(y, y_pred)
22 plt.plot([0, max(y)], [0, max(y)], 'k--', lw=2) # droite d'ajustement
23 plt.xlabel('Vraies valeurs')
24 plt.ylabel('Valeurs prédites')
25 plt.title(str(model) + f'Prédictions vs vraies données (R² = {R2_model:.2f})')
26 plt.show()
```



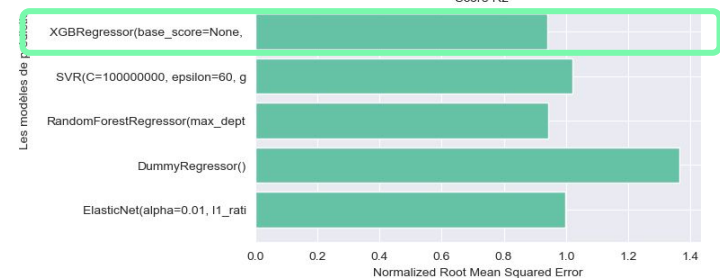
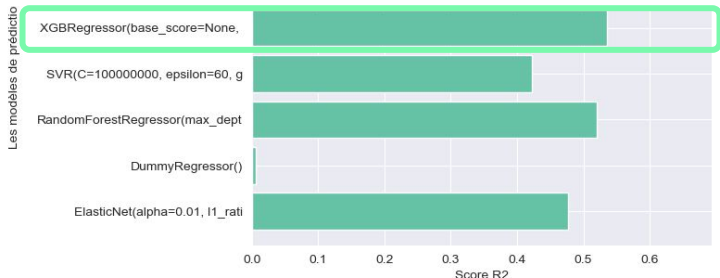
Choix du modèle



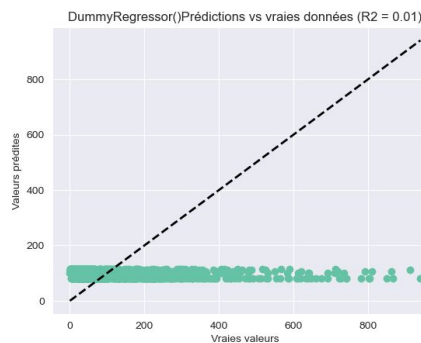
Choix du modèle - Émissions de CO₂

+ dur à prédire, moins
bons scores

Comparatif des différents modèles



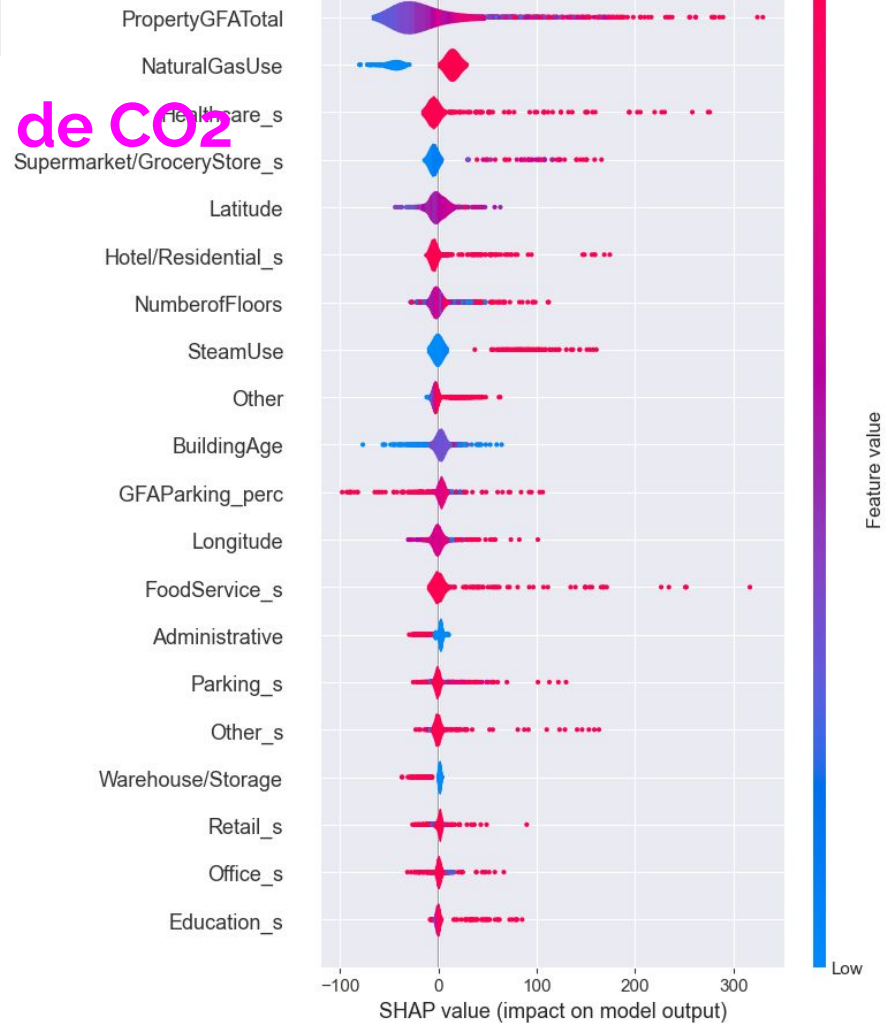
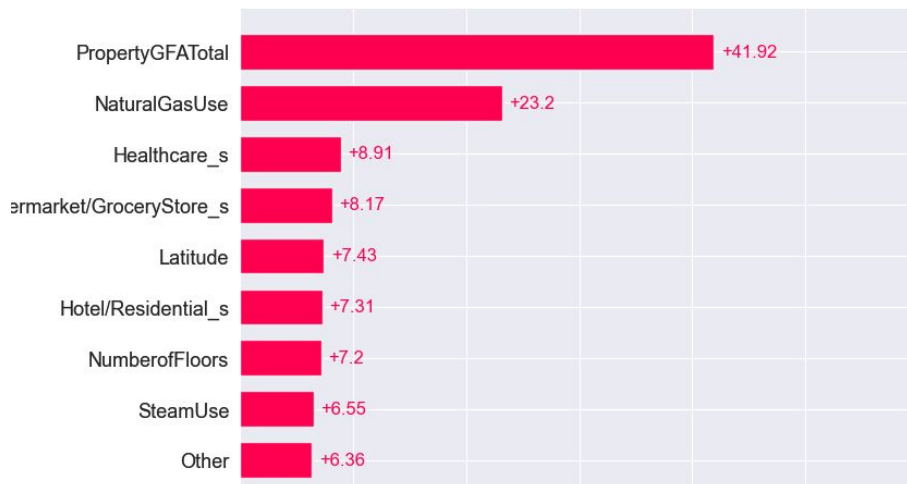
	Modèle	RMSE	nRMSE	R2
0	ElasticNet(alpha=0.01, l1_ratio=0.925)	102.637	0.998	0.477
1	DummyRegressor()	140.346	1.365	0.006
2	LinearRegression()	126.515	1.230	0.293
3	RandomForestRegressor(max_depth=80, max_features='sqrt', min_samples_split=4, n_estimators=400)	96.882	0.942	0.520
4	SVR(C=100000000, epsilon=60, gamma=1e-05)	105.115	1.022	0.423
5	XGBRegressor(base_score=None, booster=None, callbacks=None, colsample_bylevel=None, colsample_bynode=None, colsample_bytree=0.5, early_stopping_rounds=None, enable_categorical=False, eval_metric=None, feature_types=None, gamma=None, gpu_id=None, grow_policy=None, importance_type=None, interaction_constraints=None, learning_rate=0.03, max_bin=None, max_cat_threshold=None, max_cat_to_onehot=None)	96.578	0.939	0.535





Choix du modèle - Émissions de CO₂

Impact des features sur la prédictions - Shap Values

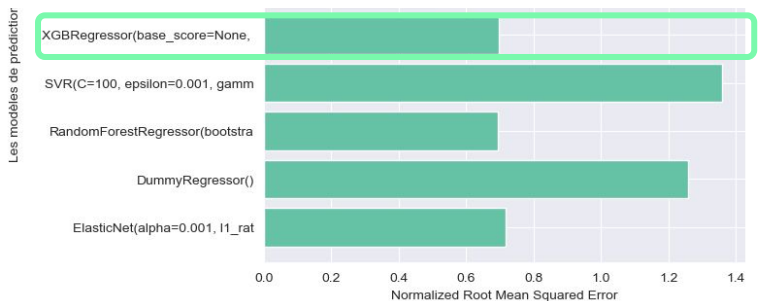
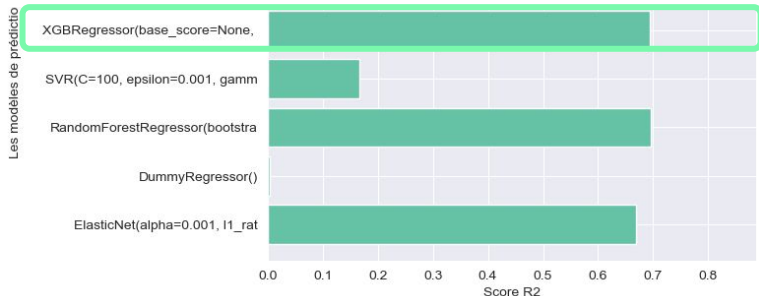




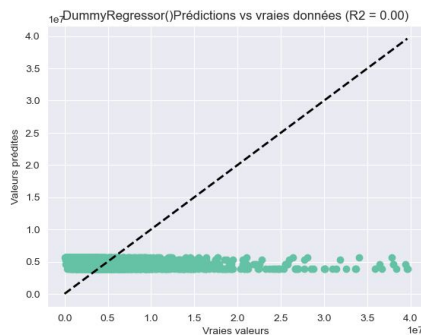
Choix du modèle - Consommation d'énergie

+ facile à prédire,
meilleurs scores

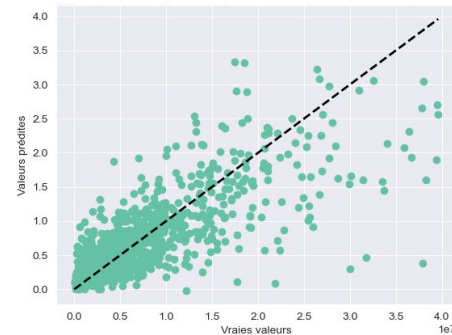
Comparatif des différents modèles



	Modèle	RMSE	nRMSE	R2
0	ElasticNet(alpha=0.001, l1_ratio=0.675)	3611133.488	0.716	0.669
1	DummyRegressor()	6340829.028	1.258	0.004
2	LinearRegression()	3773933.598	0.748	0.609
3	RandomForestRegressor(bootstrap=False, max_depth=80, max_features=10, min_samples_split=8, n_estimators=300)	3506088.943	0.695	0.696
4	SVR(C=100, epsilon=0.001, gamma=0.35)	6854078.079	1.359	0.167
5	XGBRegressor(base_score=None, booster=None, callbacks=None, colsample_bylevel=None, colsample_bynode=None, colsample_bytree=0.7, early_stopping_rounds=None, enable_categorical=False, eval_metric=None, feature_types=None, gamma=None, gpu_id=None, grow_policy=None, importance_type=None, interaction_constraints=None, learning_rate=0.03, max_delta_step=0.0001, max_leaf_nodes=None, min_child_weight=1, missing=None, monotone_constraints=None, multi_output_type='raw', n_jobs=-1, num_parallel_tree=1, random_state=None, reg_alpha=0.001, reg_lambda=0.675, scale_pos_weight=1.0, subsample=0.8, tree_method='gpu_hist', validate_parameters=None, verbosity=1)	3519258.249	0.698	0.693



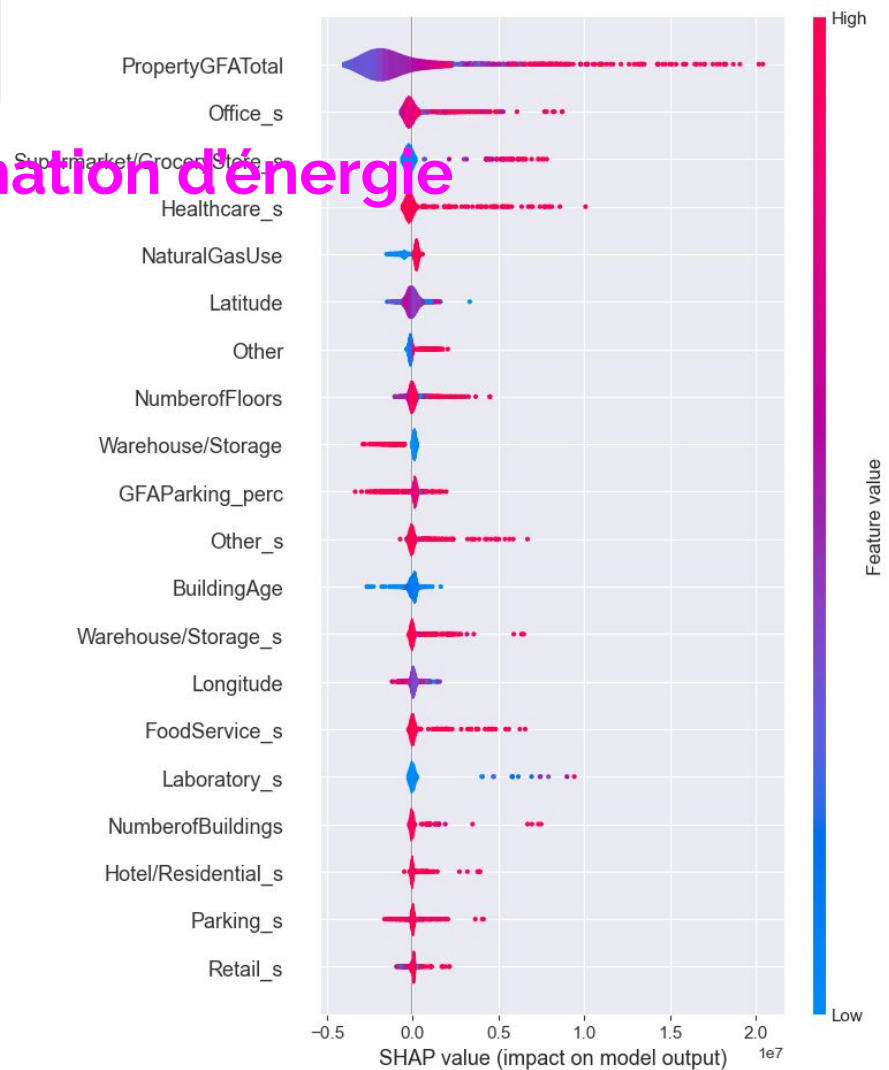
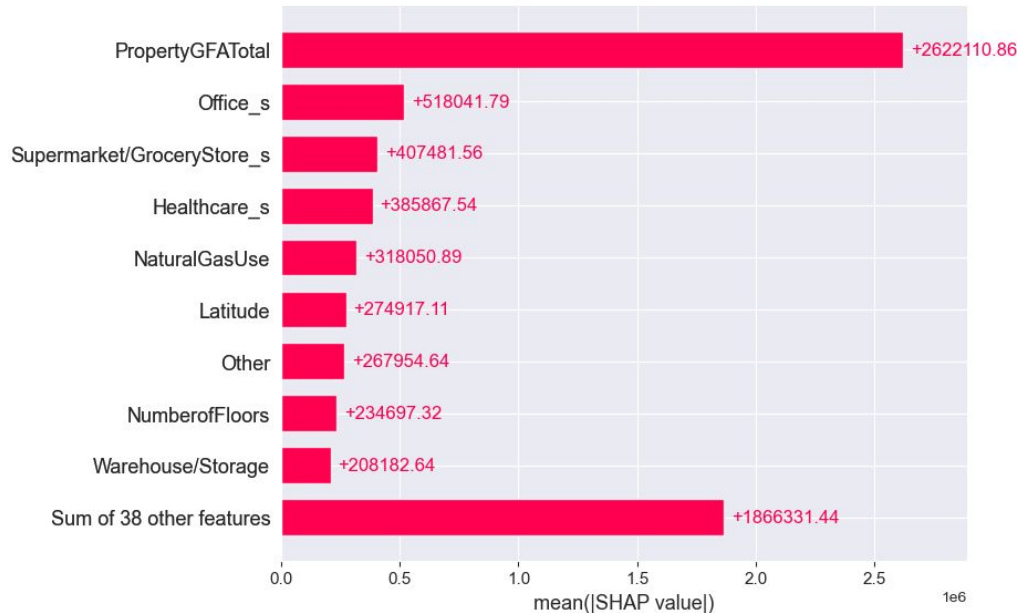
XGBoost





Choix du modèle - Consommation d'énergie

Impact des features sur la prédictions - Shap Values



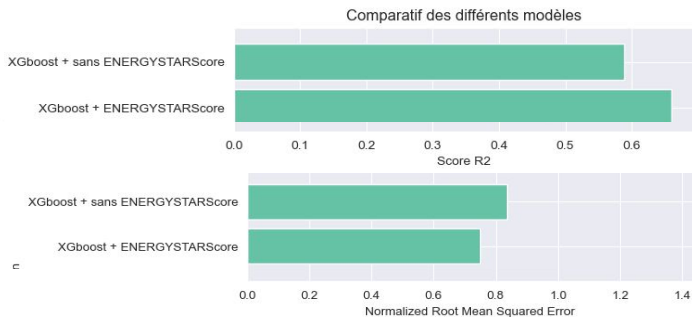
Intérêt de l'Energy Star Score





Intérêt de l'Energy Star Score

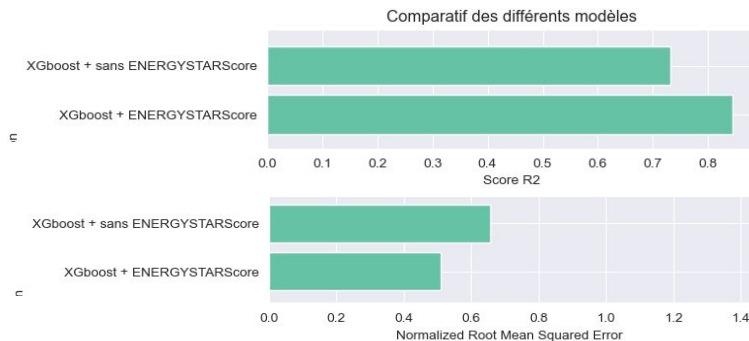
Émissions de CO2



Modèle	RMSE	nRMSE	R2
XGboost + ENERGYSTARScore	76.994	0.749	0.660
XGboost + sans ENERGYSTARScore	86.107	0.837	0.588

+ 17%

Consommations d'énergie



Modèle	RMSE	nRMSE	R2
XGboost + ENERGYSTARScore	2569723.949	0.510	0.844
XGboost + sans ENERGYSTARScore	3305370.579	0.656	0.731

+ 15%

L'intérêt de l'Energy Star Score est **discutable** : si son coût est élevé on peut s'en passer, même s'il précise la prédiction, l'objectif final est de baisser les émissions de carbone. Un **ordre de grandeur** peut être suffisant.

Conclusions & Remarques



Si c'était à refaire :

- Je rajouterais une mesure / un graphique de la dispersion des écarts à la prédiction pour voir si le modèle est suffisant pour donner un ordre de grandeur sans trop se tromper.
- Je ferais des essais avec beaucoup moins de variables aussi pour voir ce que ça donne (car là j'en utilise beaucoup quand même et certaines plus dures à obtenir que d'autres).

Avec plus de connaissances :

- Je mettrais en production mon modèle afin que l'on puisse directement saisir des variables et obtenir une prédiction pour un bâtiment donné, et j'ajouterais le détail de la contribution de chaque variable au calcul grâce aux shap values locales.
- J'essayerais d'appliquer une fonction réversible facilement sur mes étiquettes de prédiction pour pouvoir obtenir une meilleure répartition et une meilleure prédiction peut-être ? Faire du feature engineering sur mes variables à prédire aussi.

Pour les prochaines fois :

- Tout de suite mettre en place un protocole pour faire toujours le même code pour obtenir mes scores et entraîner mes modèles pour aller plus vite

Le plus important : j'ai gagné en compétences

