Prédiction des besoins énergétique de bâtiments

DATASET : RELEVÉS EFFECTUÉS PAR SEATTLE EN 2016 : HTTPS://WWW.SEATTLE.GOV/ENERGYBENCHMARKING

Plan

Présentation du jeu de données

Analyse exploratoire

Feature Engineering

Prédictions GHGE

Prédictions EnergyUse

Conclusion

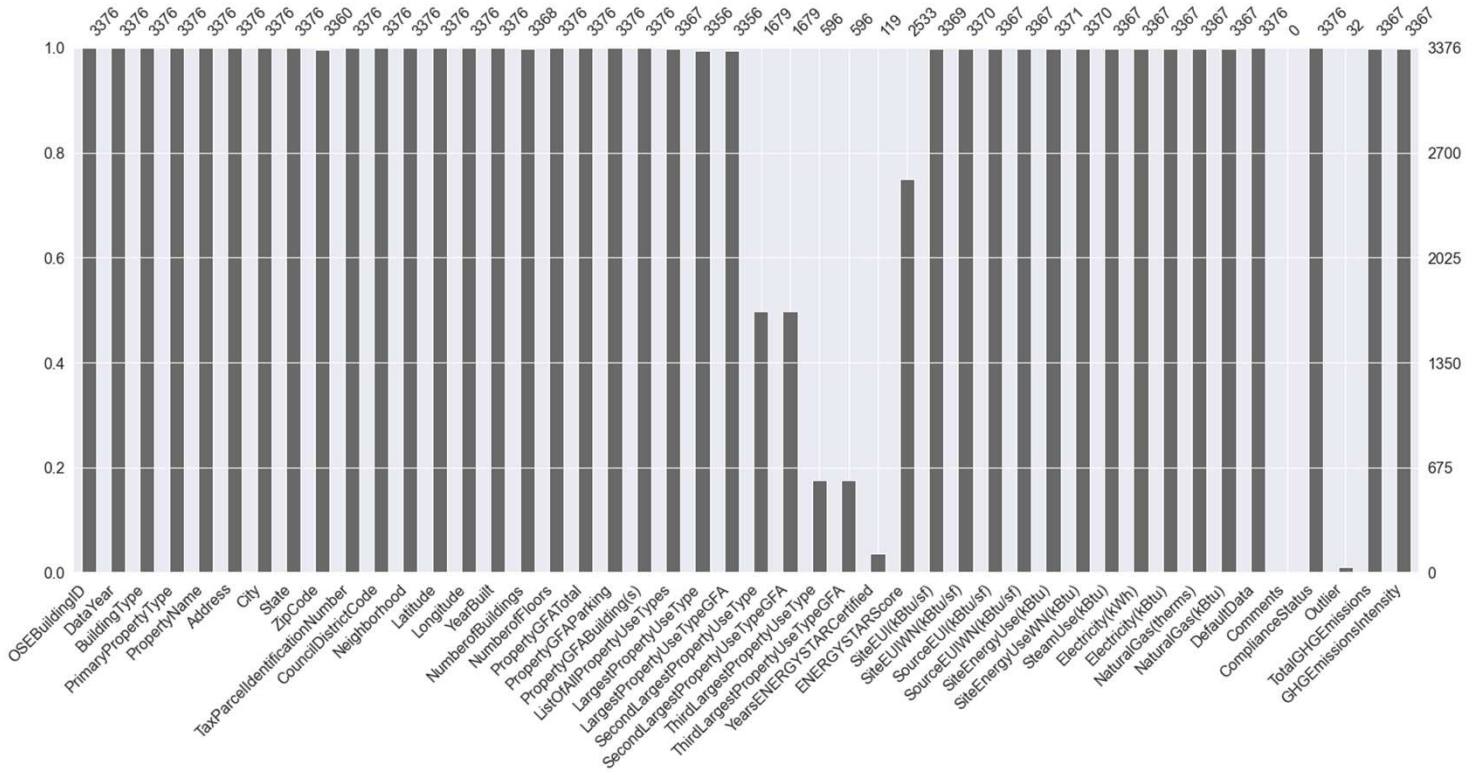
Analyse exploratoire

Présentation du jeu de données

Dense Dataset 3376 x 46

Sélection des bâtiments Non-Résidentiels

Dense Dataset 1668 x 46



Analyse exploratoire

Evaluation des indicateurs structurels

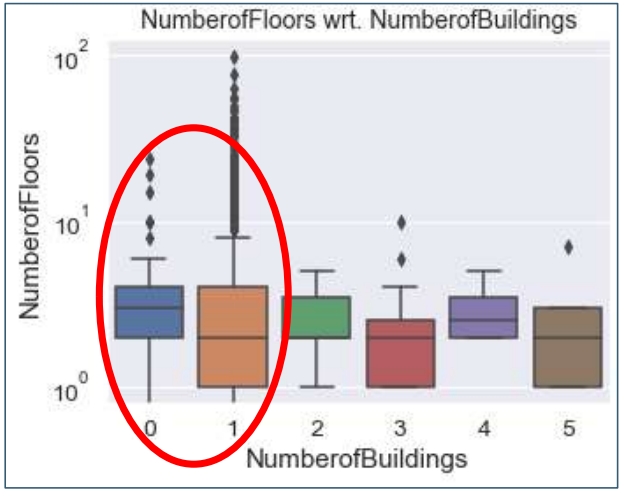
Propriété[Number of Buildings = 0]

⇨ représentent ~100 propriétés

Hypothèse

Propriété[Number of Buildings = 0]
⇔buildings partielles

Conservé à 0 : statut spécifique



Similarité 0 ou 1 Building

Kruskal test	0, 1	1, 2
TotalGHGEmissions	0.035713	0.897314
SiteEUI(kBtu/sf)	0.010706	0.824940

Analyse exploratoire

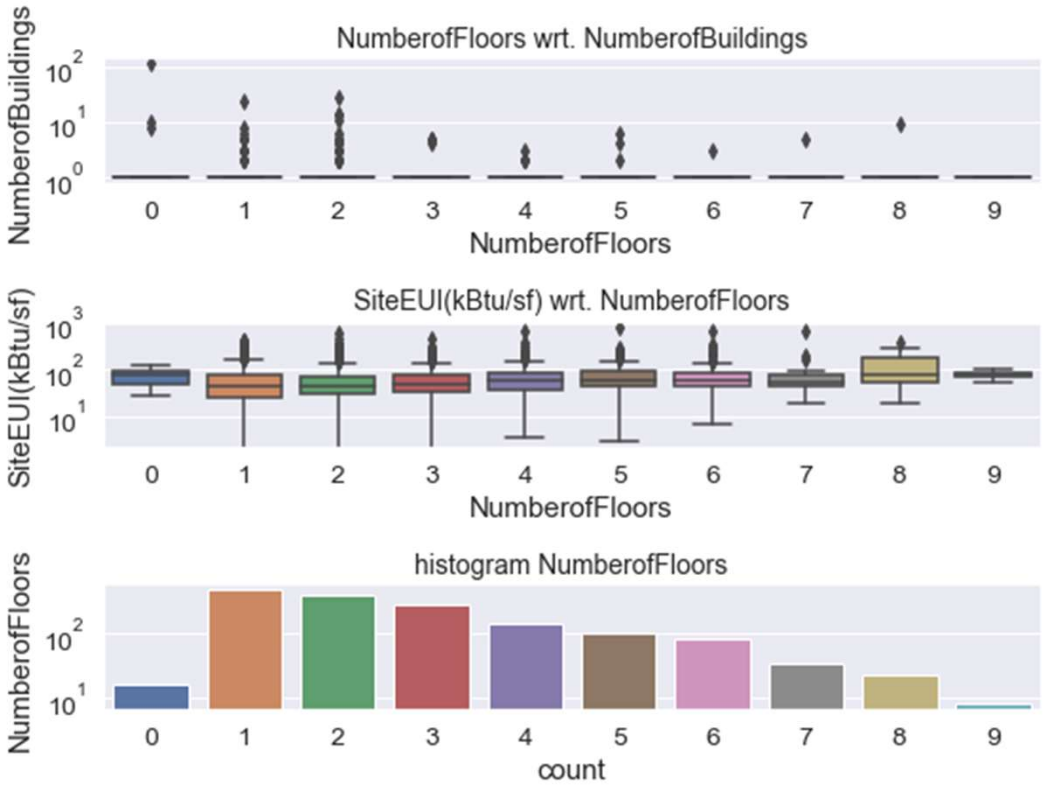
Evaluation des indicateurs structurels

Propriété[Number of Floors]

0
- Campus (111 Buildings) => median
- Recherche sur internet

99

Suppression
des valeurs non-corrigibles (5)

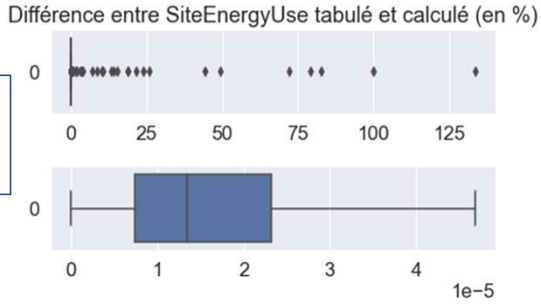


Analyse exploratoire

Evaluation des indicateurs liés aux énergies

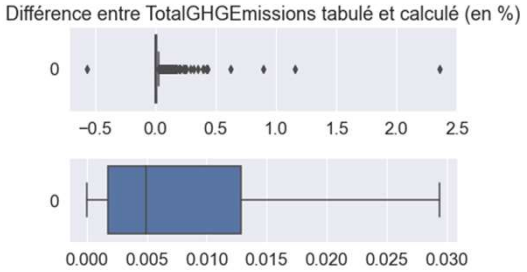
❑ **cohérence de SiteEnergyUse(kBtu)**

SiteEnergyUse = ElectricityUse + NaturalGasUse + SteamUse



❑ **cohérence de TotalGHGEmissions**

TotalGHGEmissions = ElectricityUse.α + NaturalGasUse.β + SteamUse.γ



❑ **Incohérence : Propriété[TotalGHGEmissions==0] et Propriété[SiteEnergyUse(kBtu)==0]**

Elimination de 30 relevés incohérents

Analyse exploratoire

Présentation du jeu de données

Avant nettoyage SHAPE X = (1668, 43)

Nettoyage des relevés

- Opération décrite précédemment

Suppression des features :

- peu ou pas renseignées

- sans impact *a priori* sur la consommation du bâtiment

- Information Redondante (linéairement dépendant ou fortement corrélé à d'autres features)

Après nettoyage SHAPE X = (1526, 17)

Analyse exploratoire

Choix des features

'BuildingType', 'PrimaryPropertyType', 'Latitude', 'Longitude', 'YearBuilt', 'NumberofBuildings', 'NumberofFloors', 'PropertyGFATotal', 'PropertyGFABuilding(s)', 'LargestPropertyUseType', 'LargestPropertyUseTypeGFA', 'SecondLargestPropertyUseType', 'SecondLargestPropertyUseTypeGFA', 'ThirdLargestPropertyUseType', 'ThirdLargestPropertyUseTypeGFA', 'SteamUse', 'NaturalGasUse'
Option 'ENERGYSTARScore'

Remplacement valeur kBtu par [1 ou 0] pour "SteamUse", "NaturalGasUse" "ElectricityUse"

fillna(0.0/unknown)

SHAPE X = (1526, 17)

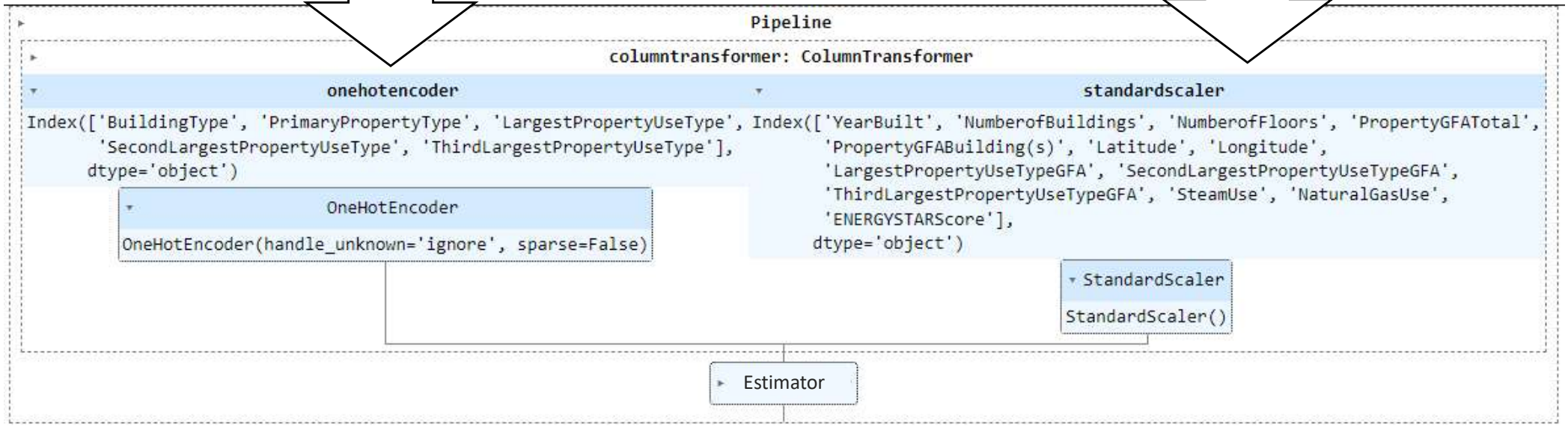
Target : y = **'SitetnergyUse(kBtu)'** **OU** **'TotalGHGEmissions'**, shape(y) = (1526,)

Feature Engineering

Transformation des données d'entrées

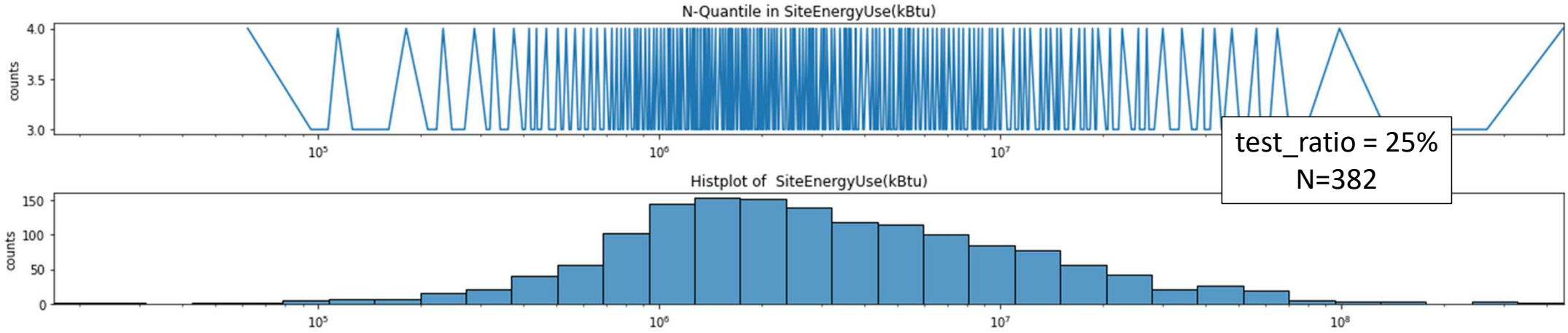
Transformation des variables numériques

Transformation des variables catégorielles



Feature Engineering

Stratification GHGE



test_ratio = 25%
N=382

DataSet de départ:
1526 lignes, 17 indicateurs

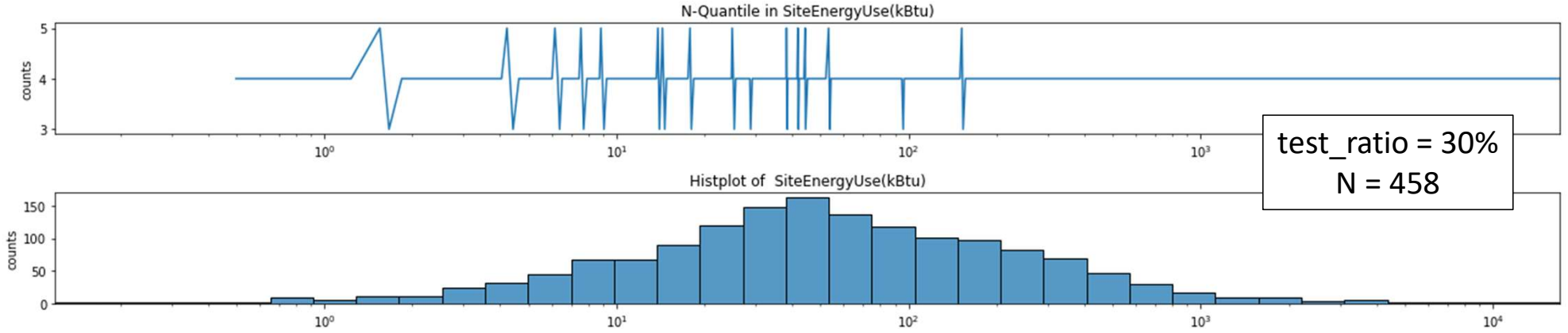
Feature Engineering

Training Set
1144 relevés, 182 indicateurs

Test set
382 relevés, 182 indicateurs

Feature Engineering

Stratification EnergyUse



test_ratio = 30%
N = 458

DataSet de départ:
1526 lignes, 17 indicateurs

Feature Engineering

Training Set
1068 relevés, 172 indicateurs

Test set
458 relevés, 172 indicateurs

Prédictions

Comparaison modèles pour prédiction TotalGHGEmissions

Modèles linéaires

algorithme	Hyperparamètres testés*	Nb steps
KNeighboorsRegressor	'n_neighbors':np.arange(3, 30, 2), 'algorithm': ["ball_tree", "kd_tree", "brute"]	200 x 3
LinearRegression		
Ridge	alphas = np.logspace(-5, 5, 200)	200
Lasso	alphas = np.logspace(-5, 5, 200)	200
Elastic_net	alphas = np.logspace(5, -5, 200) l1_ratio=[0.9999, 0.99985, 0.9998, 0.99975, 0.9997]	200 x 5
SVR	'epsilon':np.logspace(-5, 5, n_alphas), C=1	50

*GridSearch avec CV = 5

Prédictions

Comparaison modèles pour prédiction TotalGHGEmissions

Modèles non-linéaires

algorithme	Hyperparamètres testés	Nb steps
KRR	alphas = np.logspace(-2, 2, 5), gammas = np.logspace(-2, 1, 4)	20
Random Tree	"n_estimators": [500, 1000], "max_depth": [50, 75], "min_samples_split": [2, 3]}	8
Gradient Boosting Regression Tree	"n_estimators": [500, 1000], "max_depth": [7, 10], "min_samples_split": [5, 10], "learning_rate": [0.1, 0.05]	16
XGBOOST Regressor	'max_depth': [3, 4], 'n_estimators': [1000], "learning_rate": [0.05, 0.01], "min_child_weight":[3, 5]	8

Prédictions

Comparaison modèles pour prédiction TotalGHGEmissions

	quantité d'individus	% test	number of CV split	algorithm	hyperparameters	score_type	training score	test score	Score R^2	training_time
0	1526	0.25	5	KNN-Regressor	{'algorithm': 'ball_tree', 'n_neighbors': 5}	RMSE	499	582	0.208	0.0282
1	1526	0.25	0	LinearRegression	default	RMSE	370	3.34e+15	-2.6e+25	0.0142
2	1526	0.25	0	Ridge w/o CV	{'alpha': 1.1233}	None	364	582	0.563	0.00751
3	1526	0.25	5	RidgeCV	{'alpha': 1382.6222}	None	477	450	0.526	0.00486
4	1526	0.25	0	Lasso w/o CV	{'alpha': 1.1514}	None	365	450	0.572	0.0237
5	1526	0.25	0	ElasticNet	{'alpha': 1.1514, 'l1_ratio': 0.0}	None	775	450	-0.0324	0.18
6	1526	0.25	5	ElasticNetCV	{'alpha': 243.7444, 'l1_ratio': 0.9999}	None	614	450	0.431	0.00912
7	1526	0.25	0	ElasticNet w/o CV	{'alpha': 1.588565129428053e-05, 'l1_ratio': 0.9997}	None	364	432	0.564	17
8	1526	0.25	5	LinearSVR	{'epsilon': {'epsilon': 355.64803062231283}}	None	621	488	0.444	0.0069
9	1526	0.25	5	KRR	{'alpha': 0.1, 'gamma': 0.01}	None	102	290	0.803	0.0944
10	1526	0.25	5	Random Forest Regression	{'max_depth': 100, 'min_samples_split': 10, 'n_estimators': 500}	None	286	488	0.36	6.13
11	1526	0.25	0	Gradient Boost Regression Tree	{'learning_rate': 0.1, 'max_depth': 10, 'min_samples_split': 10, 'n_estimators': 1000}	None	0.00452	542	0.315	11.9
12	1526	0.25	5	XGBOOST	{'learning_rate': 0.01, 'max_depth': 3, 'min_child_weight': 5, 'n_estimators': 1000}	None	228	304	0.784	1.76

Prédictions

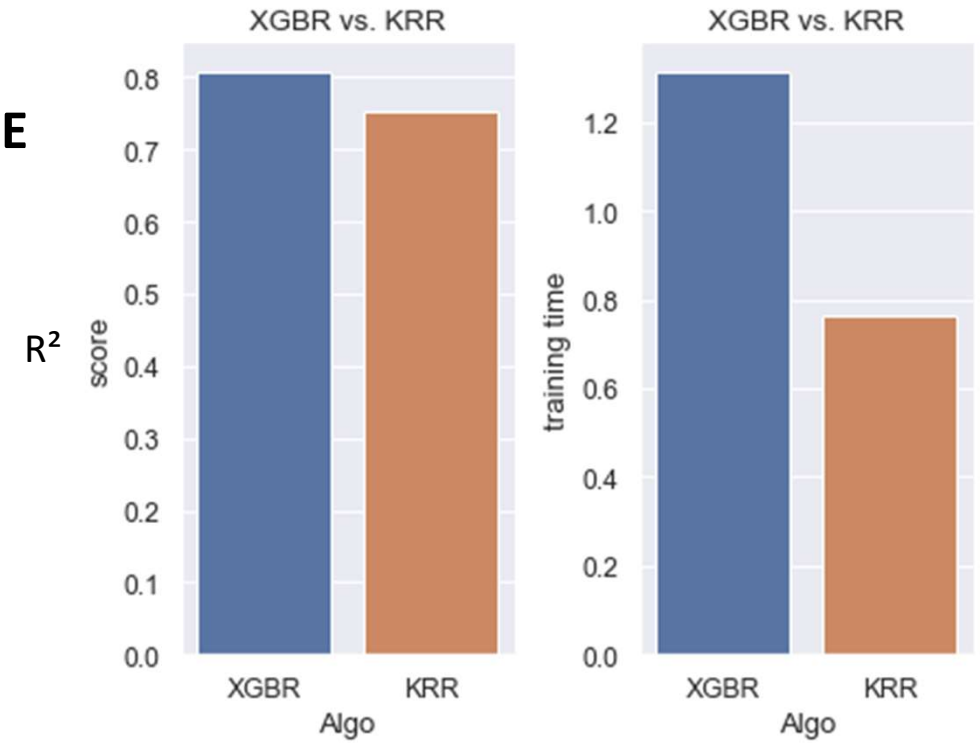
Sélection modèles pour prédictions TotalGHGEmissions

Prédictions

Comparaison modèles XGBOOST - KRR sur prédictions EnergyUse

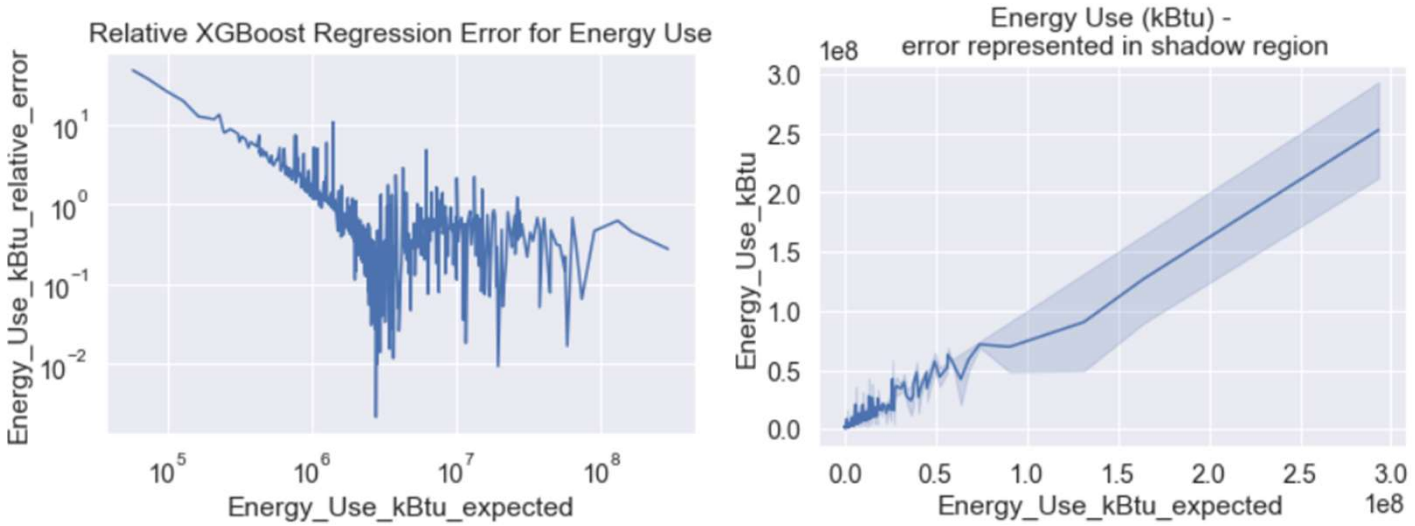
Même type de modèle que TotalGHGE

DataSet utilisé:
1526 lignes, 172 indicateurs



Prédictions

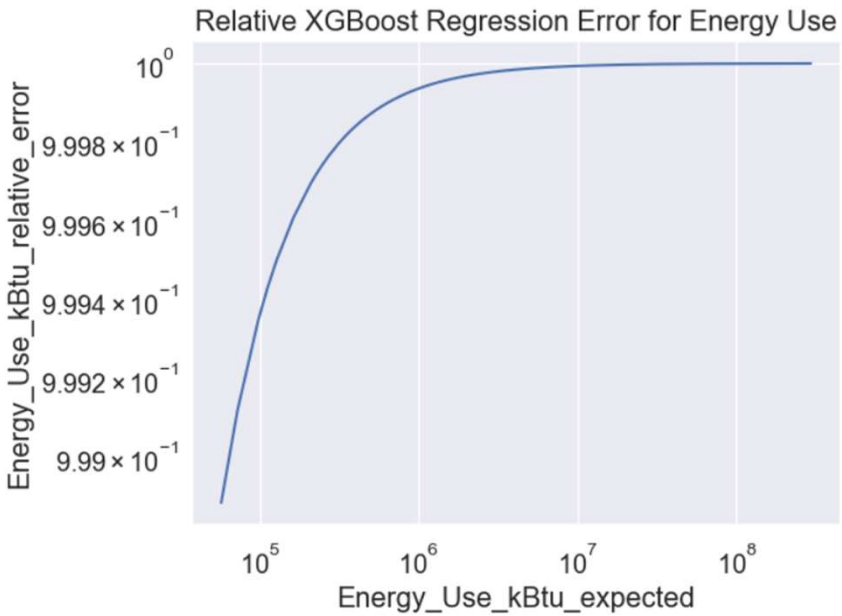
convergence XBOOST: RMSE

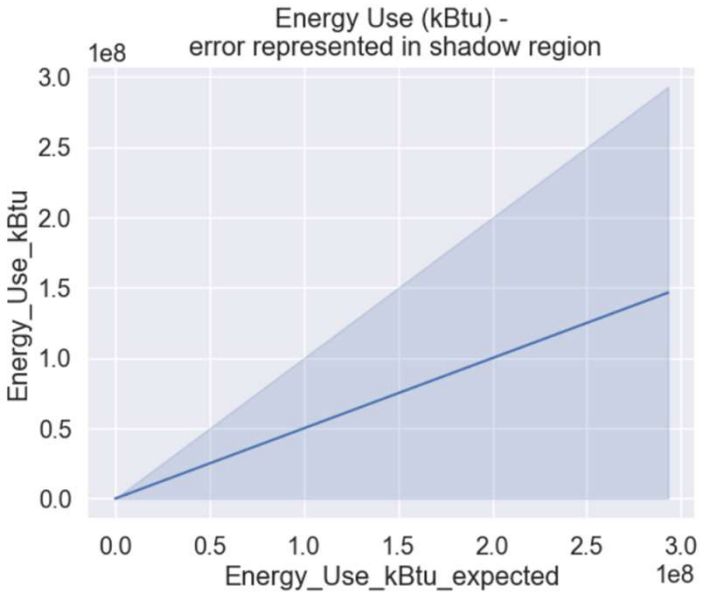


choix du score Eval_metric = "RMSE", permet d'avoir une erreur relative "contenu" (<1) pour les plus "gros" consommateurs, mais pénalise l'estimation sur des bâtiment faible consommateur.

Prédictions

convergence XBOOST: RMSLE





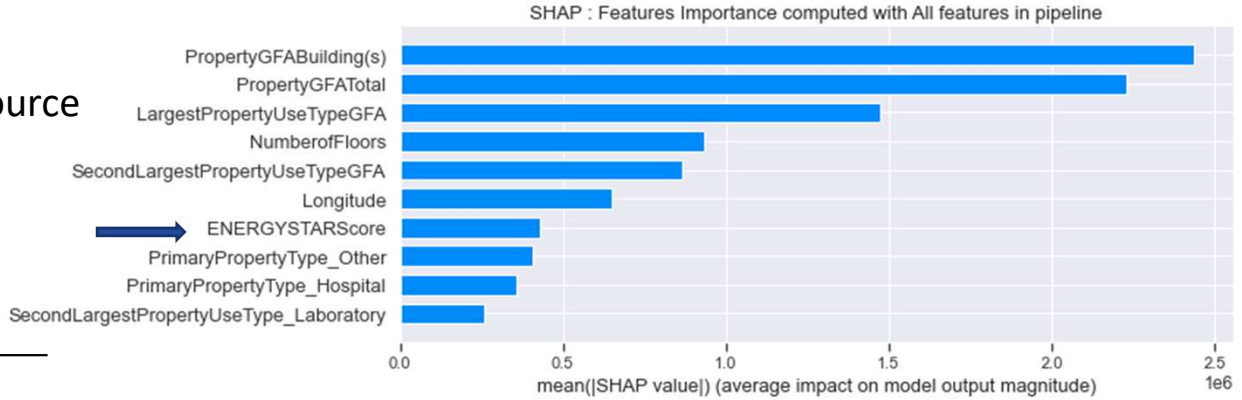
L'utilisation de "RMSLE" permet d'optimiser le modèle plutôt pour les petits consommateurs.

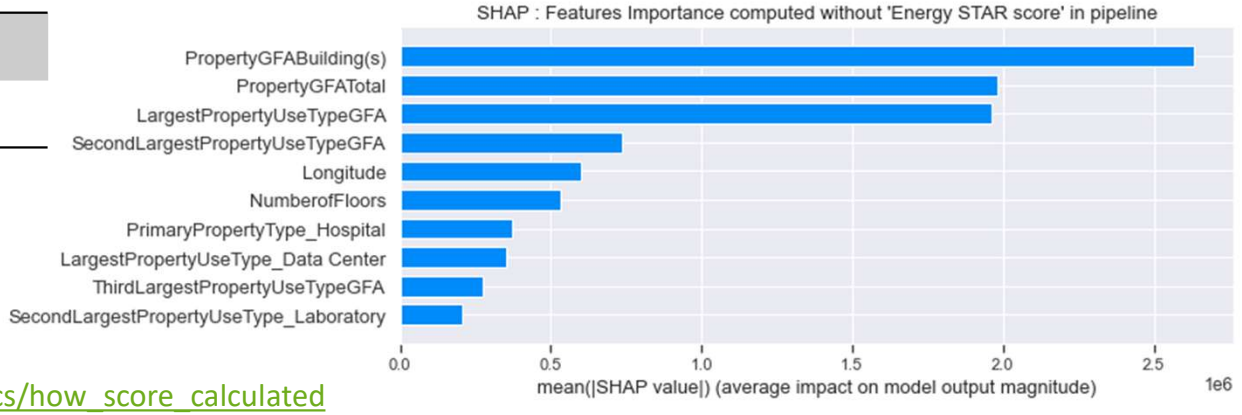
La fusion des 2 modèles nous permettra d'obtenir un compromis pour l'ensemble des bâtiments, si nécessaire.

Prédictions

Comparaison modèles sur prédictions EnergyUse

Energy Star Score :
Estimation de l'utilisation d'Energie à la Source



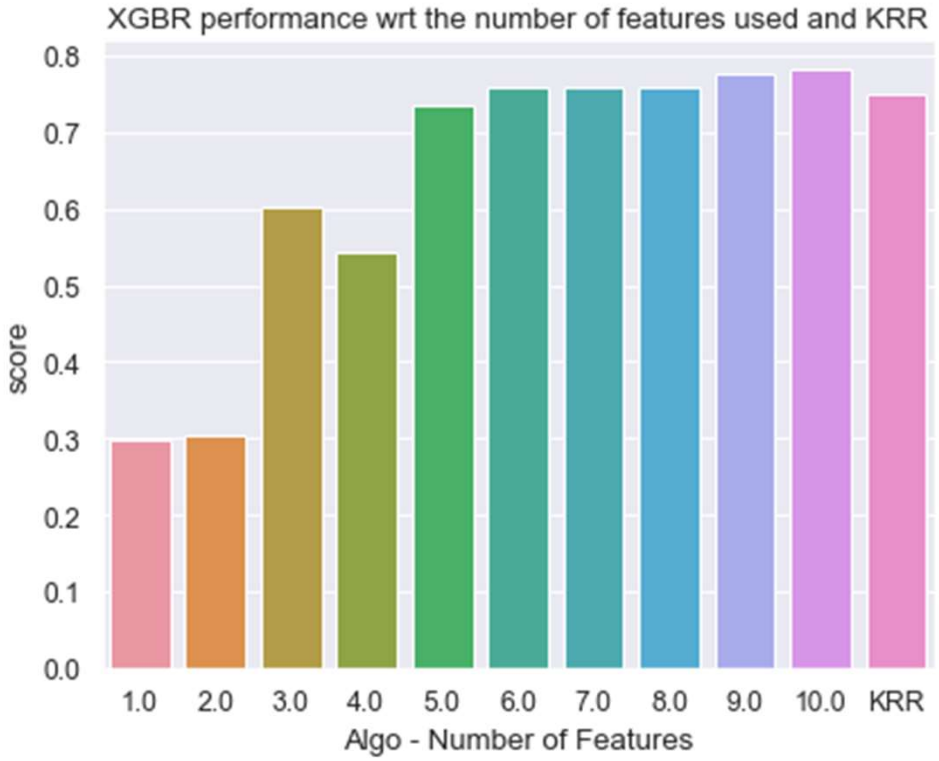


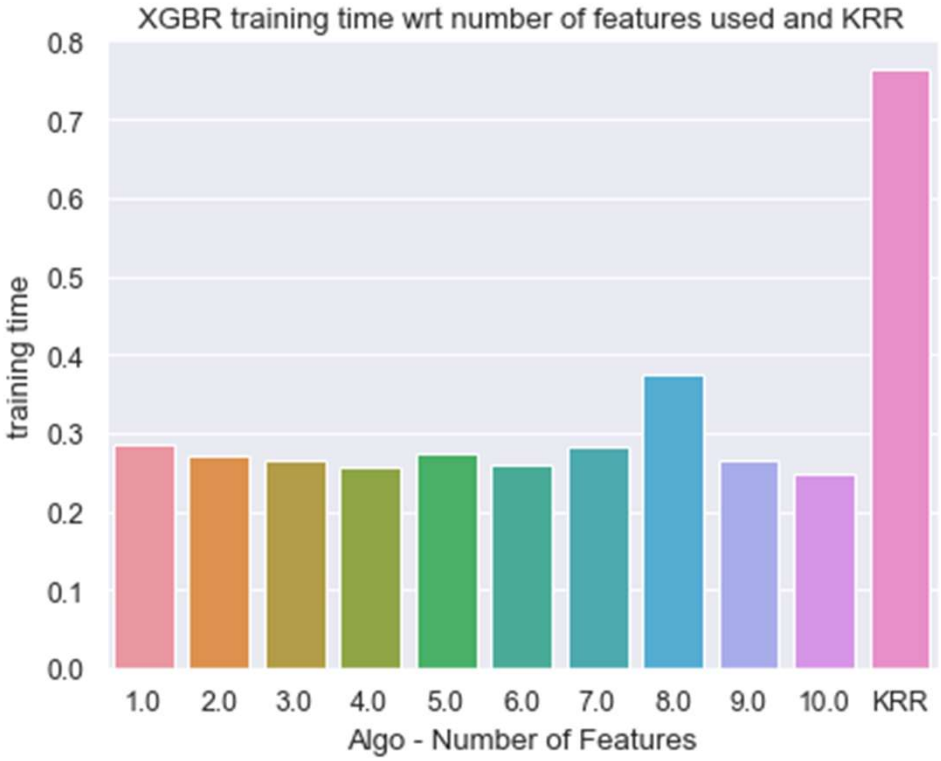
	Performance XGBR (R^2)
Avec Energy Star Score	0.79
Sans Energy Star Score	0.81

https://www.energystar.gov/buildings/benchmark/understand_metrics/how_score_calculated

Prédictions

XGBOOST optimisé vs. KRR





Conclusion

- Des algos « simples » comme KRR permettent d'obtenir des résultats intéressants.
- Peu de Features sont nécessaires pout optimiser les algos (comme XGBOOST testé ici)
- Energy STAR Score est inutile
- Préférence à XGBOOSTpour EnergyUse
  - KRR pour GHGE
  - XGBOOSTpour EnergyUse
- S'il ne fallait en garder qu'un : XGBOOST - Précision légèrement supérieur, temps entrainement équivalent, mais attention optimisation plus lente que KRR.
- L'utilisation de 2 modèles en fonction de la consommation du pour les faibles consommateur