


Anticipez les besoins en consommation électrique de bâtiments

Morgan May

Projet n°3 du parcours Ingénieur Machine Learning



Sommaire

1. Sujet & interprétation
 2. Préparation et exploration des données
 3. Feature engineering et Baseline
 4. Modélisation de 'SiteEnergyUse': sélection de modèles
 5. Modélisation de 'SiteEnergyUse': Optimisation et modèles finaux
 6. Modélisation de 'TotalGHGEmissions': sélection de modèles
 7. Modélisation de 'TotalGHGEmissions': Optimisation et modèles finaux
 8. Conclusion et réflexion sur l'EnergySTAR Score
- 



1. Sujet & interprétation

1. Sujet, interprétation & pistes de recherche

Le sujet de ce projet est d'**étudier la faisabilité d'estimer les performances de consommation d'énergie et d'émissions de CO2 de bâtiments non-résidentiels** de Seattle à partir des données déclaratives de leur permis d'exploitation commerciale (taille, surface, usage...).

Pour cela, nous avons à disposition 2 jeux de données (de 2015 et 2016) relatifs à un certain nombre de bâtiments de Seattle incluant ces données déclaratives et des mesures d'émissions et de consommation d'énergie.

Le problème ainsi décrit est un cas classique d'**application de techniques d'apprentissage supervisé avec des algorithmes de régression**.



2. Préparation et exploration des données

2. Préparation et exploration des données

Une première rapide exploration des 2 jeux de données (de 2015 et 2016) nous permet de voir qu'il s'agit des mesures relatives à peu près aux mêmes bâtiments pour ces 2 années.

Le premier est composé de 3340 individus pour 47 colonnes et le second de 3376 individus et 46 colonnes.

Nous choisissons de **regrouper ces jeux de données dans un unique dataset**. Pour cela, la première tâche consiste à **homogénéiser les colonnes** entre ces deux datasets :

These columns are in the 2015 dataset but not in the one from 2016 : ['Location', 'OtherFuelUse(kBtu)', 'GHGEmissions(Metric TonsCO2e)', 'GHGEmissionsIntensity(kgCO2e/ft2)', 'Comment', '2010 Census Tracts', 'Seattle Police Department Micro Community Policing Plan Areas', 'City Council Districts', 'SPD Beats', 'Zip Codes']

These columns are in the 2016 dataset but not in the one from 2015 : ['Address', 'City', 'State', 'ZipCode', 'Latitude', 'Longitude', 'Comments', 'TotalGHGEmissions', 'GHGEmissionsIntensity']

Certaines colonnes sont simplement dénommées différemment ('Zip Codes' et 'ZipCode', 'Comment' et 'Comments...'); un simple **renommage** suffisant alors; tandis que d'autres regroupements nécessitent une opération un peu plus complexe.

2. Préparation et exploration des données

Ainsi, la variable 'Location' de data_2015 comprend sous la forme d'un dictionnaire des informations correspondantes à différentes variables de data_2016 :

```
{'latitude': '47.61219025', 'longitude': '-122.33799744', 'human_address': '{"address": "405 OLIVE WAY", "city": "SEATTLE", "state": "WA", "zip": "98101"}'}
```

<class 'dict'>

Une extraction de ces informations nous permet de créer les mêmes colonnes dans data_2015.

D'autres variables de data_2015 telles que 'OtherFuelUse(kBtu)', '2010 Census Tracts', 'Seattle Police Department Micro Community Policing Plan Areas', 'City Council Districts' et 'SPD Beats' non en revanche pas d'équivalent dans data_2016 et sont donc supprimées.

Enfin, **les 2 datasets sont concaténés en un seul** en faisant attention aux changements de types de données de certaines variables avec cette opération. **Le dataset ainsi formé contient 6716 individus pour 46 colonnes.**

2. Préparation et exploration des données

Note : à partir de cette phase (et à des fins pédagogiques), nous avons pris le parti de réaliser les opérations de préparation suivantes à l'aide de transformateurs regroupés dans une pipeline :

- Comme l'indique le sujet, notre étude ne concerne pas les bâtiments résidentiels, nous avons donc directement filtré les bâtiments ne correspondant pas à l'aide de la variable 'BuildingType'.
- Nous avons supprimé les variables 'Comments' et 'YearsENERGYSTARCertified' du fait d'une proportion très importante de valeurs manquantes.
- Nous avons exploité la variable 'Outlier' présente dans le dataset permettant d'identifier des valeurs atypiques et avons choisi de les supprimer simplement car ne représentant qu'une partie très faible des individus (environ 1%). Nous supprimons également cette variable désormais inutile.
- De la même manière, nous avons exploité la variable 'ComplianceStatus' afin de supprimer les 3 individus explicités comme non-conformes. Nous supprimons également cette colonne une fois cela fait.

2. Préparation et exploration des données

Notre exploration des potentielles variables cibles 'TotalGHGEmissions', 'SiteEnergyUse(kBtu)' et 'SiteEnergyUseWN(kBtu)' nous permet notamment de conclure à une forte corrélation entre ces 2 dernières variables relatives à la consommation d'énergie desquelles nous choisissons de nous séparer de la dernière; 'SiteEnergyUseWN(kBtu)'; car normalisée avec les conditions climatiques dont nous disposons d'aucune information.

Sur un principe similaire, l'étape suivante consiste à supprimer du jeu de données les autres variables relatives à la consommation d'énergie ou aux émissions afin d'**éviter tout risque de 'data leakage'** : 'SiteEUI(kBtu/sf)', 'SiteEUIWN(kBtu/sf)', 'SourceEUI(kBtu/sf)', 'SourceEUIWN(kBtu/sf)', 'SteamUse(kBtu)', 'Electricity(kWh)', 'Electricity(kBtu)', 'NaturalGas(therms)', 'NaturalGas(kBtu)' and 'GHGEmissionsIntensity'.

2. Préparation et exploration des données

Les étapes suivantes de préparation nous amènent à un jeu de données de 3167 individus par 22 colonnes :

- Passage des données textuelles en minuscules (notamment pour regrouper certaines catégories entre elles).
- Suppression de variables a priori inutiles à notre modélisation : 'OSEBuildingID', 'City', 'State', 'PropertyName', 'Address', 'TaxParcelIdentificationNumber', 'ListOfAllPropertyUseTypes', 'Latitude' et 'Longitude'.

Afin de sélectionner les variables qualitatives pertinentes pour modéliser nos deux variables cibles, nous réalisons une ANOVA (analyse de variance) entre chacune d'entre elles et nos deux variables cibles distinctement.

Ceci nous amène à **conserver les variables catégorielles** 'BuildingType', 'PrimaryPropertyType', 'Neighborhood', 'LargestPropertyUseType', 'SecondLargestPropertyUseType', 'ThirdLargestPropertyUseType', 'CouncilDistrictCode', 'ZipCode' et 'YearBuilt' tandis que 'DefaultData' et 'DataYear' sont **supprimées** car ne rejetant pas l'hypothèse nulle d'indépendance avec chacune de nos 2 variables cibles.

2. Préparation et exploration des données

De la même manière qu'avec les variables catégorielles, nous regardons les corrélations linéaires entre chacune de nos variables cibles et les variables numériques restantes afin de supprimer celles ne semblant n'avoir aucun lien significatif : 'NumberOfBuildings' et 'ENERGYStarScore'. Les variables numériques suivantes sont donc conservées : **'NumberOfFloors', 'LargestPropertyUseTypeGFA', 'SecondLargestPropertyUseTypeGFA', 'ThirdLargestPropertyUseTypeGFA', 'PropertyGFATotal', 'PropertyGFAParking', 'PropertyGFABuilding(s)'**.

Enfin, les dernières étapes de préparation des données sont dépendantes du type de données (catégorielle, catégorielle numérique et numérique) et sont donc réalisées à travers des pipelines mis en parallèle dans le pipeline final.

A noter que les variables cibles 'TotalGHGEmissions' et 'SiteEnergyUse' passent également par les premières étapes de préparation (ne subissant elles-mêmes aucune transformation) afin que les individus supprimés lors de certaines étapes soient bien retirés de ces colonnes également.

2. Préparation et exploration des données

Ces étapes de préparation relatives au type de données sont les suivantes :


Variables catégorielles	Variables catégorielles-numériques	Variables numériques
<ul style="list-style-type: none">• Correction d'erreurs de typographie• Imputation par le mode le plus fréquent• Encodage OneHot	<ul style="list-style-type: none">• Suppression des quelques individus avec ZipCode manquant• Imputation des autres valeurs manquantes éventuelles par le mode le plus fréquent	<ul style="list-style-type: none">• Suppression des quelques individus avec NumberOfFloors manquant• Imputation par 0 des valeurs manquantes de 'SecondLargestPropertyUseTypeGFA' et 'ThirdLargestPropertyUseTypeGFA'• Imputation par la médiane de toute autre valeur manquante• Normalisation

2. Préparation et exploration des données


Après l'ensemble de ces étapes de préparation, notre dataset comporte 3167 individus pour 204 colonnes (dont les colonnes d'encodage) correspondantes à 16 variables d'entrée et les 2 variables cibles.

Nos variables d'entrées conservées et traitées sont les suivantes :

- "BuildingType",
- "PrimaryPropertyType"
- "Neighborhood"
- "LargestPropertyUseType"
- "SecondLargestPropertyUseType"
- "ThirdLargestPropertyUseType"
- "ZipCode"
- "CouncilDistrictCode"
- "YearBuilt"
- "NumberOfFloors"
- "LargestPropertyUseTypeGFA"
- "SecondLargestPropertyUseTypeGFA"
- "ThirdLargestPropertyUseTypeGFA"
- "PropertyGFATotal"
- "PropertyGFAParking"
- "PropertyGFABuilding(s)"



3. Feature engineering et Baseline



3. Feature engineering

Avant de créer de nouvelles variables à partir de celles déjà présentes initialement dans le dataset, nous créons une baseline afin de mesurer l'effet de ces créations de nouvelles variables.

Baseline :

S'agissant d'un problème de régression, nous choisissons comme baseline un **modèle de régression relativement simple de type Lasso**, sans optimisation de ses hyperparamètres (ici, le coefficient de régularisation Lasso est donc de 1 par défaut).

Métriques :

Afin d'évaluer les performances de cette baseline et celles des modèles futurs, nous choisissons 3 métriques :

- La **RMSE** : qui nous permet d'avoir une mesure d'erreur correspondante à une distance entre prédiction et valeur réelle, indépendante du signe de la différence entre ces deux valeurs.
- La **MAE** : qui nous permet d'avoir une mesure d'erreur absolue entre prédiction et valeur réelle.
- Le coefficient de détermination **R²** : qui nous permet de juger de la capacité d'un modèle à expliquer la variance des données.

3. Feature engineering

Découpage Train set - Test set :

Afin de prévenir une situation dans laquelle le test set aurait une distribution de valeurs de la variable cible significativement différente de la distribution du jeu d'entraînement, nous choisissons de réaliser un **découpage stratifié en fonction de chaque variable cible**.

On a donc un premier jeu d'entraînement `X1_train` et `y1_train` ainsi que son jeu de test correspondant `X2_test` et `y1_test` dont le découpage est stratifié par rapport à la variable 'SiteEnergyUse'.

Nous avons également un second jeu d'entraînement `X2_train` et `y2_train` ainsi que son jeu de test correspondant `X2_test` et `y2_test` dont le découpage est stratifié par rapport à la seconde variable cible 'TotalGHGEmissions'.

Nous utilisons pour cela la fonction `StratifiedShuffleSplit` de Scikit-learn avec une **proportion de 20% des données réservés aux jeux de tests**.

Les jeux d'entraînement comporte ainsi 2533 individus et ceux de test 634 individus.

3. Feature engineering

Performances de la Baseline pour 'SiteEnergyUse' :

Après entraînement sur le premier jeu d'entraînement X1_train_BL et y1_trainBL (stratifié sur cette variable cible), nous comparons les résultats de prédictions obtenus par cette baseline sur le jeu de test X1_test_BL avec les valeurs réelles de ce jeu de test y1_test_BL. Nous obtenons les performances 'brutes' suivantes :

Performances of this baseline model :

RMSE between y1_test_BL and y1_pred_BL : 1.4928682367008055e+07

MAE between y1_test_BL and y1_pred_BL : 5.8450003170222165e+06

R² between y1_test_BL and y1_pred_BL : 0.6503769029942598

Reminder of the mean of this target : 8.71763356659227e+06

Performances de la Baseline pour 'TotalGHGEmissions':

Après apprentissage sur les seconds jeux d'entraînement, nous obtenons les résultats suivants pour le second jeu de test :

Performances of this baseline model :

RMSE between y2_test_BL and y2_pred_BL : 3.4480264961187584e+02

MAE between y2_test_BL and y2_pred_BL : 1.5086355186338338e+02

R² between y2_test_BL and y2_pred_BL : 0.5796323475156633

Reminder of the mean of this target : 1.6351458990536278e+02

3. Feature engineering

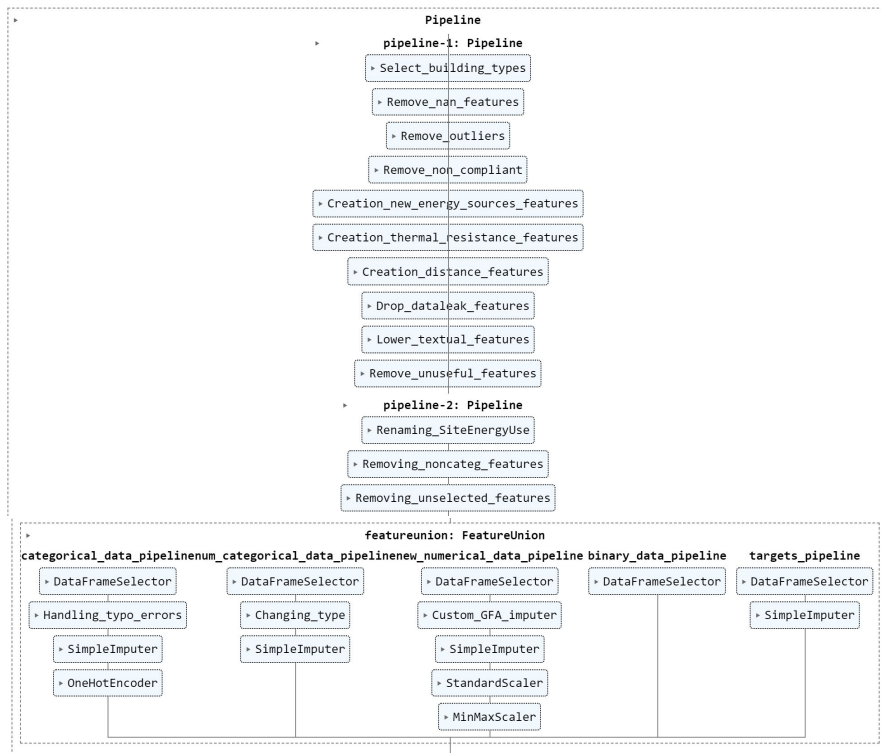
Création de nouvelles variables :

Afin d'améliorer les performances de notre modèle, nous allons créer **5 nouvelles variables à partir des variables initialement présentes dans le dataset**. Ces étapes de création sont insérées dans les pipelines de préparation des données avant la suppression éventuelle des variables utilisées pour les créer. Ces variables respectent la contrainte de pouvoir se trouver sur un bail d'exploitation commerciale :

- **1 variable binaire 'UseElectricity'** correspondante à l'utilisation d'une source d'énergie électrique ou non par le bâtiment (déterminée avec les variables 'Electricity(kWh)' et 'Electricity(kBtu)').
- **1 variable binaire 'UseNaturalGas'** correspondante à l'utilisation d'une source d'énergie de type gaz naturel ou non par le bâtiment (déterminée avec les variables 'NaturalGas(therms)' et 'NaturalGas(kBtu)').
- **1 variable numérique 'thermal_resistance_approach_1'** déterminée comme le ratio des valeurs de 'PropertyGFABuilding(s)' sur 'NumberofFloors' et permettant de rendre compte de la proportion de surface alaire (à même d'échange thermique avec l'extérieur) par rapport à la surface totale d'un bâtiment.
- **1 variable numérique 'thermal_resistance_approach_2'** déterminée comme le ratio des valeurs de 'PropertyGFABuilding(s)' sur 'NumberofBuildings' et permettant de la même façon de rendre compte de la proportion de surface alaire (à même d'échange thermique avec l'extérieur) par rapport à la surface totale d'un bâtiment.
- **1 variable numérique 'distance'** créée à partir des variables 'Longitude' et 'Latitude' et faisant référence à une distance euclidienne par rapport à un point de référence située au centre de la ville de Seattle.

3. Feature engineering

Schéma
représentant la
pipeline finale
complète de
transformation



3. Feature engineering

Performances de la Baseline pour 'SiteEnergyUse' sur les jeux de données incluant les nouvelles variables :

RMSE between y1_test and y1_pred : 1.4499350820492152e+07
MAE between y1_test and y1_pred : 5.461872684842862e+06
R² between y1_test and y1_pred : 0.6701972474574196

Reminder of the mean of this target : 8.71763356659227e+06

Difference between new RMSE and former RMSE with new engineered features : -429331.5465159025
Difference between new MAE and former MAE with new engineered features : -383127.6321793543
Difference between new R² and former R² with new engineered features : 0.019820344463159834

Performances de la Baseline pour 'TotalGHGEmissions' sur les jeux de données incluant les nouvelles variables :

RMSE between y2_test and y2_pred : 3.4393755516165805e+02
MAE between y2_test and y2_pred : 1.5087389774670368e+02
R² between y2_test and y2_pred : 0.5817390686973604


Reminder of the mean of this target : 1.6351458990536278e+02

Difference between new RMSE and former RMSE with new engineered features for TotalGHGEmissions: -0.8650944502177822
Difference between new MAE and former MAE with new engineered features for TotalGHGEmissions: 0.010345883320297844
Difference between new R² and former R² with new engineered features for TotalGHGEmissions: 0.00210672118169708

Nous pouvons voir ici que le feature engineering a permis d'améliorer assez substantiellement les performances de cette baseline sur chacune des deux variables cibles.



4. Modélisation de 'SiteEnergyUse' : sélection de modèles



4. Modélisation de 'SiteEnergyUse' : sélection de modèles

L'étape de sélection de modèles consiste ici à tester les performances de différents types d'algorithmes de régression sans optimisation de leurs hyperparamètres afin d'obtenir une idée de l'ordre de grandeur de leurs performances atteignables sur nos jeux de données.

Pour cela, nous testons deux principaux types d'algorithmes : des algorithmes à méthodes ensemblistes et non-ensemblistes.

Algorithmes testés pour chacun des 2 modélisations à effectuer :

Méthodes non-ensemblistes	Méthodes ensemblistes
<ul style="list-style-type: none">• Algorithme de régression Lasso• Algorithme de régression Ridge• Algorithme ElasticNet• SVM à noyau rbf ('Radial Basis Function')• SVM à noyau linéaire	<ul style="list-style-type: none">• Arbre de décision• Forêt aléatoire• AdaBoost• Bagging• Gradient Boosting

4. Modélisation de 'SiteEnergyUse' : sélection de modèles

Nous nous concentrons dans un premier temps sur la modélisation de la variable cible 'SiteEnergyUse'.

Pour la sélection de ces modèles, nous réalisons une **validation croisée sur 10 échantillons** via la fonction `cross_val_score()` de Scikit-Learn. Nous obtenons les résultats suivants sur les jeux de validation pour les méthodes non-ensemblistes :

	Model	Average RMSE	[RMSE std] / [RMSE average]	Standard deviation of RMSE	Average MAE	[MAE std] / [MAE average]	Standard deviation of MAE	Average R ²	[R ² std] / [Average R ²]	Standard Deviation of R ²
0	Lasso	1.333e+07	46.6%	6.208e+06	4.975e+06	12.6%	6.27147474163017e+05	0.607	25.8%	0.157
1	Ridge	1.584e+07	72.7%	1.152e+07	5.000e+06	16.3%	8.141156454406264e+05	0.537	30.7%	0.165
2	ElasticNet	2.173e+07	58.8%	1.277e+07	7.65e+06	12.1%	9.272154416491776e+05	0.103	54.900000000000006%	0.057
3	SVM_rbf	2.338e+07	53.0%	1.239e+07	6.835e+06	16.5%	1.1262072866910065e+06	-0.082	-44.5%	0.036
4	SVM_linear	2.332e+07	53.2%	1.242e+07	6.763e+06	16.6%	1.1231525715527001e+06	-0.075	-44.4%	0.033

Nous y voyons notamment que **la MAE apparaît comme étant la métrique la plus stable sur les jeux de validation** (un indicateur de stabilité étant ici calculé via le ratio entre l'écart-type et la moyenne obtenus sur l'ensemble des jeux de validation). **Nous utiliserons donc principalement la MAE comme métrique d'optimisation** pour les algorithmes d'optimisation des hyperparamètres plus tard.

Ici, **l'algorithme de régression Lasso semble le plus performant**. Nous le conservons donc pour optimisation future.

4. Modélisation de 'SiteEnergyUse' : sélection de modèles

Sur les méthodes ensemblistes, nous observons les résultats suivants :

	Model	Average RMSE	[RMSE std] / [RMSE average]	Standard deviation of RMSE	Average MAE	[MAE std] / [MAE average]	Standard deviation of MAE	Average R ²	[R ² std] / [Average R ²]	Standard Deviation of R ²
0	DecisionTreeRegressor	1.563e+07	93.0%	1.454e+07	2.982e+06	38.5%	1.147683243575143e+06	0.547	65.9%	0.361
1	RandomForestRegressor	1.423e+07	93.30000000000001%	1.328e+07	2.918e+06	30.2%	8.804812116250739e+05	0.654	36.3%	0.238
2	AdaBoostRegressor	2.483e+07	43.8%	1.088e+07	1.740e+07	26.1%	4.5416437881012065e+06	-0.548	-160.9%	0.882
3	BaggingRegressor	1.412e+07	93.4%	1.319e+07	3.021e+06	32.2%	9.734843532660975e+05	0.593	44.3%	0.263
4	GradientBoostingRegressor	1.476e+07	90.7%	1.338e+07	3.607e+06	25.8%	9.321860442605843e+05	0.642	35.9%	0.231

Nous pouvons remarquer ici aussi la meilleure stabilité de la MAE.

Souhaitant à ce stade conserver une méthode ensembliste parallèle et une autre séquentielle, **nous sélectionnons l'algorithme de forêt aléatoire ainsi que celui de Gradient Boosting** en regardant l'ensemble des métriques.



5. Modélisation de 'SiteEnergyUse' :Optimisation et modèles finaux



5. Modélisation de 'SiteEnergyUse' :Optimisation et modèles finaux

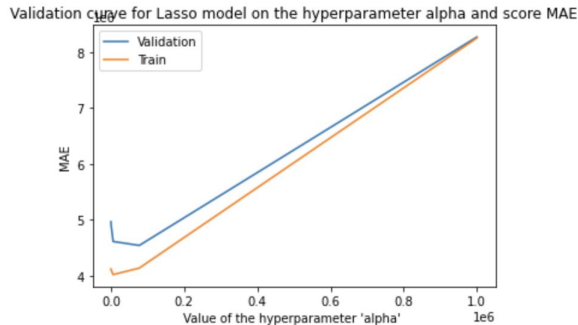
Afin d'optimiser les hyperparamètres de nos 3 modèles retenus jusqu'ici, nous procédons en 2 étapes distinctes :

- une première recherche de l'influence de la variation d'un hyperparamètre sur la métrique d'optimisation (la MAE ici) via l'affichage d'une courbe de validation avec la fonction `validation_curve()` de Scikit-Learn
- une seconde étape consistant en une recherche sur grille avec validation croisée des valeurs optimales des hyperparamètres via la fonction `GridSearchCV()` de Scikit-learn.

5. Modélisation de 'SiteEnergyUse' :Optimisation et modèles finaux

Optimisation du modèle Lasso :

Pour étudier l'influence de la variation d'un hyperparamètre sur la MAE, nous affichons une courbe de validation, ici sur l'unique hyperparamètre que nous optimisons, alpha, le coefficient de régularisation Lasso. Voici la courbe obtenue après calibration sur un intervalle nous permettant de distinguer clairement un optimum :



Nous pouvons voir que la valeur du coefficient de régularisation minimisant la MAE se situe globalement entre ~ 1 et $2e+05$, nous effectuons donc une recherche sur grille (sur 20 valeurs) de la valeur optimale de ce coefficient sur cet intervalle.

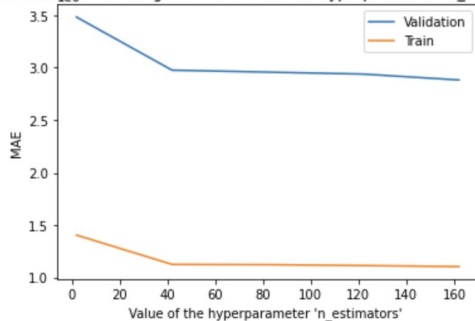
Cette recherche sur grille nous permet de déterminer $\alpha=16238$ comme valeur optimisant la moyenne des MAE obtenues sur les échantillons de validation croisée ($4.54e+06$).

5. Modélisation de 'SiteEnergyUse' :Optimisation et modèles finaux

Optimisation du modèle RandomForestRegressor :

Nous choisissons pour ce modèle une optimisation des paramètres `n_estimators` (nombre d'estimateurs de base) et `max_depth` (profondeur des estimateurs de base) afin de contrôler également la complexité du modèle et donc sa capacité de généralisation.

Validation curve for RandomForestRegressor model on the hyperparameter `n_estimators` and score MAE

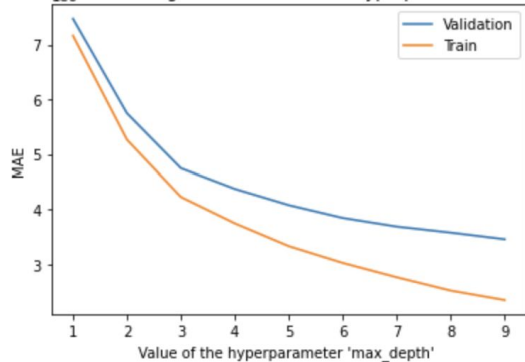


Nous pouvons voir une forte erreur de généralisation sur la première courbe peu importe la valeur de `n_estimators` mais avec très peu de progrès après ~80.

5. Modélisation de 'SiteEnergyUse' :Optimisation et modèles finaux

suite de l'Optimisation du modèle RandomForestRegressor :

Validation curve for RandomForestRegressor model on the hyperparameter max_depth and score MAE



Cette seconde courbe nous permet de choisir une limite pour max_depth à 6 correspondant à un point d'inflexion et limitant autant que se peut l'erreur de généralisation.

La recherche sur grille nous permet de déterminer n_estimators=50 et max_depth=5 comme hyperparamètres optimisant la moyenne des MAE obtenues sur les échantillons de validation croisée (3.81e+06).

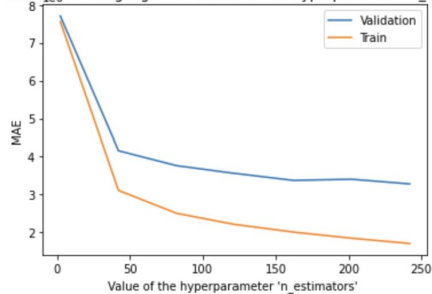
Lors de cette recherche sur grille nous testons également de modifier l'hyperparamètre 'bootstrap' en le qualifiant comme 'False' et 'criterion' en lui désignant l'erreur absolue comme fonction de perte. Cette dernière valeur est retenue et permet d'obtenir le score optimal dans les conditions décrites ci-dessus.

5. Modélisation de 'SiteEnergyUse' :Optimisation et modèles finaux

Optimisation du modèle GradientBoostingRegressor :

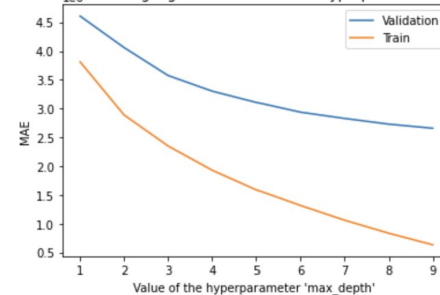
Nous choisissons pour ce modèle une optimisation des paramètres `n_estimators` (nombre d'estimateurs de base), `max_depth` (profondeur des estimateurs de base) et '`subsample`' afin de contrôler également la complexité du modèle (et donc sa capacité de généralisation). Nous optimisons aussi le taux d'apprentissage ('`learning_rate`').

Validation curve for GradientBoostingRegressor model on the hyperparameter `n_estimators` and score MAE



Aux dépens d'une capacité de généralisation affectée, nous choisissons ici de réaliser la recherche sur grille entre `n_estimators=50` et `n_estimators=200`.

Validation curve for GradientBoostingRegressor model on the hyperparameter `max_depth` and score MAE

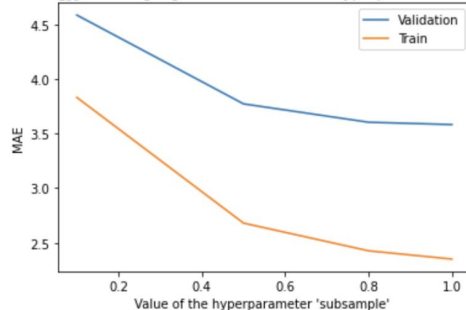


La courbe ci-dessus nous invite utiliser 6 comme limite à `max_depth` afin de limiter un overfitting inutile pour réduire l'erreur sur les jeux de validation.

5. Modélisation de 'SiteEnergyUse' :Optimisation et modèles finaux

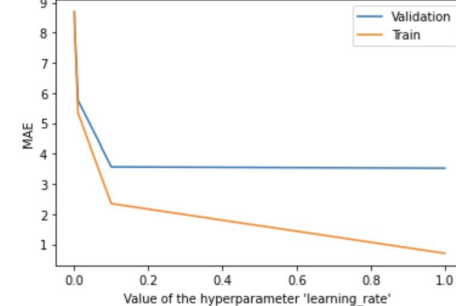
suite de l'Optimisation du modèle GradientBoostingRegressor :

Validation curve for GradientBoostingRegressor model on the hyperparameter subsample and score MAE



La courbe ci-dessus nous invite à chercher l'optimum de 'subsample' entre 0.5 et 0.8.

Validation curve for GradientBoostingRegressor model on the hyperparameter learning_rate and score MAE



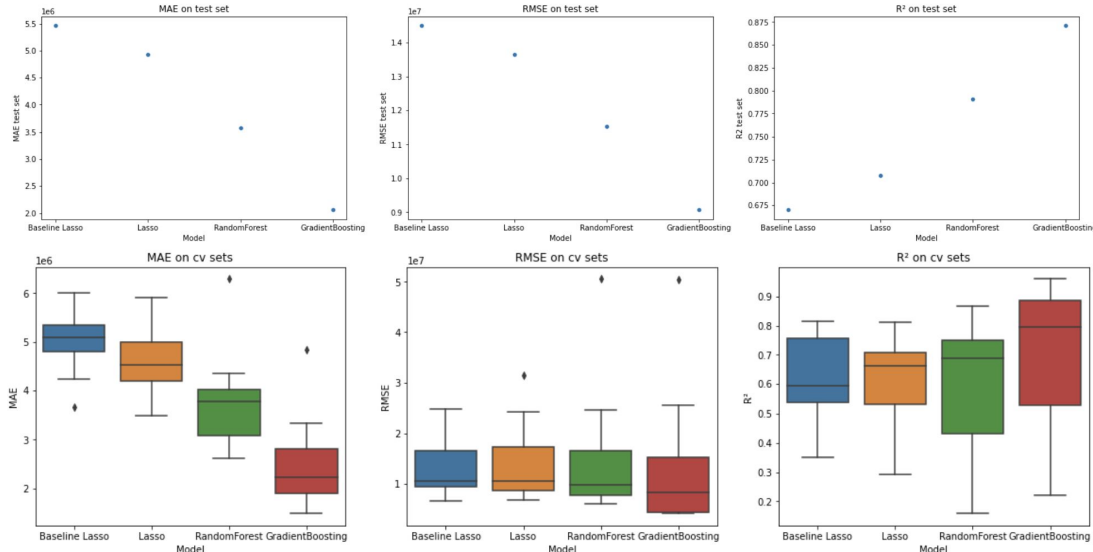
La courbe ci-dessus nous invite à chercher l'optimum de 'learning_rate' entre 0.01 et 0.2.

L'ensemble de ces optimisations nous a permis de déterminer, via une recherche sur grille, un score optimal sur les jeux de validation de $2.75e+06$ pour 'learning_rate'=0.2, 'max_depth'=6, 'n_estimators'=200 et 'subsample'=0.8.

5. Modélisation de 'SiteEnergyUse' :Optimisation et modèles finaux

Comparaison des modèles et sélection du modèle final :

	Model	MAE train set	MAE cv set	MAE test set	RMSE train set	RMSE cv set	RMSE test set	R2 train set	R2 cv set	R2 test set
0	Baseline Lasso	4.192e+06	4.975e+06	5462000.0	9.183e+06	1.333e+07	14500000.0	0.874	0.607	0.670
1	Lasso	4.013e+06	4.536e+06	4927000.0	9.687e+06	1.39e+07	13650000.0	0.860	0.605	0.708
2	RandomForest	2.999e+06	3.801e+06	3580000.0	9.632e+06	1.539e+07	11530000.0	0.861	0.592	0.791
3	GradientBoosting	4.74e+05	2.499e+06	2056000.0	6.462e+05	1.39e+07	9064000.0	0.999	0.692	0.871



L'ensemble de ces résultats démontrent la supériorité du modèle basé sur l'algorithme de Gradient Boosting (malgré un surajustement de la variance du jeu d'entraînement).

Modèle final retenu :

```
GradientBoostingRegressor(learning_rate=0.2, max_depth=6, n_estimators=200, subsample=0.8)
```




6. Modélisation de 'TotalGHGEmissions' : sélection de modèles



6. Modélisation de 'TotalGHGEmissions' : sélection de modèles

Comme pour la variable cible 'SiteEnergyUse', nous réalisons une sélection de modèles en testant les performances des mêmes types d'algorithmes selon les mêmes métriques (MAE, RMSE et R^2).

Nous obtenons les résultats suivants sur les méthodes non-ensemblistes :

	Model	Average RMSE	[RMSE std] / [RMSE average]	Standard deviation of RMSE	Average MAE	[MAE std] / [MAE average]	Standard deviation of MAE	Average R^2	[R^2 std] / [Average R^2]	Standard Deviation of R^2
0	Lasso	4.490e+02	47.099999999999994%	2.115e+02	1.596e+02	13.200000000000001%	2.103558027468965e+01	0.297	108.2%	0.321
1	Ridge	4.762e+02	47.3%	2.250e+02	1.764e+02	12.4%	2.194012211959467e+01	0.221	155.29999999999998%	0.344
2	ElasticNet	5.961e+02	60.199999999999996%	3.587e+02	1.995e+02	25.900000000000002%	5.1614440520523274e+01	0.073	53.6%	0.039
3	SVM_rbf	6.287e+02	57.3%	3.603e+02	1.628e+02	35.0%	5.699258611079536e+01	-0.068	-49.7%	0.034

Ici, l'algorithme de régression Lasso semble le plus performant. Nous le conservons donc pour optimisation future.

6. Modélisation de 'TotalGHGEmissions' : sélection de modèles

Sur les méthodes ensemblistes, nous observons les résultats suivants :

	Model	Average RMSE	[RMSE std] / [RMSE average]	Standard deviation of RMSE	Average MAE	[MAE std] / [MAE average]	Standard deviation of MAE	Average R ²	[R ² std] / [Average R ²]	Standard Deviation of R ²
0	DecisionTreeRegressor	5.019e+02	72.8%	3.652e+02	8.92e+01	37.3%	3.324135907972711e+01	0.017	6524.9%	1.081
1	RandomForestRegressor	3.573e+02	78.0%	2.787e+02	8.366e+01	40.699999999999996%	3.4011602593180925e+01	0.563	69.6%	0.392
2	AdaBoostRegressor	5.257e+02	44.0%	2.311e+02	3.952e+02	19.3%	7.615692223229193e+01	-0.690	-199.9%	1.379
3	BaggingRegressor	3.640e+02	72.8%	2.651e+02	8.787e+01	40.8%	3.589097267610121e+01	0.512	88.2%	0.451
4	GradientBoostingRegressor	3.603e+02	79.2%	2.854e+02	9.982e+01	33.2%	3.3188518346350605e+01	0.559	59.3%	0.332

Nous pouvons remarquer ici aussi la meilleure stabilité de la MAE.

Souhaitant à ce stade conserver une méthode ensembliste parallèle et une autre séquentielle, nous sélectionnons l'algorithme de forêt aléatoire ainsi que celui de Gradient Boosting en regardant l'ensemble des métriques.



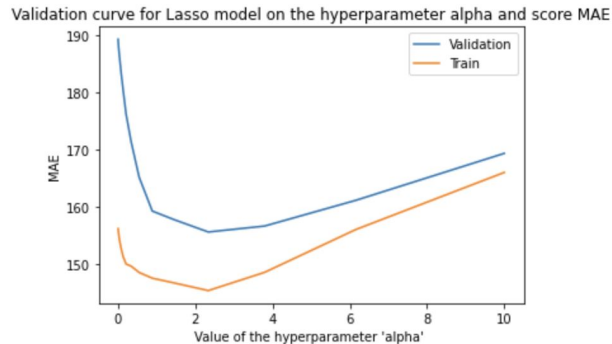
7. Modélisation de 'TotalGHGEmissions' : Optimisation et modèles finaux



7. Modélisation de 'TotalGHGEmissions' : Optimisation et modèles finaux

Optimisation du modèle Lasso :

Voici la courbe de validation de l'unique hyperparamètre que nous optimisons pour ce modèle, alpha, le coefficient de régularisation Lasso; après calibration sur un intervalle nous permettant de distinguer clairement un optimum :



Nous pouvons voir que la valeur du coefficient de régularisation minimisant la MAE se situe globalement entre ~ 0.001 et 10, nous effectuons donc une recherche sur grille (sur 20 valeurs) de la valeur optimale de ce coefficient sur cet intervalle.

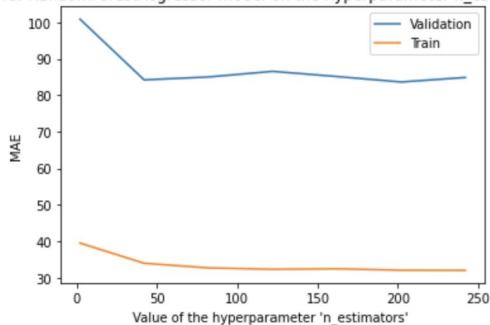
Cette recherche sur grille nous permet de déterminer $\alpha=2.34$ comme valeur optimisant la moyenne des MAE obtenues sur les échantillons de validation croisée (155.7).

7. Modélisation de 'TotalGHGEmissions' : Optimisation et modèles finaux

Optimisation du modèle RandomForestRegressor :

Nous choisissons pour ce modèle une optimisation des paramètres `n_estimators` (nombre d'estimateurs de base) et `max_depth` (profondeur des estimateurs de base) afin de contrôler également la complexité du modèle et donc sa capacité de généralisation.

Validation curve for RandomForestRegressor model on the hyperparameter `n_estimators` and score MAE

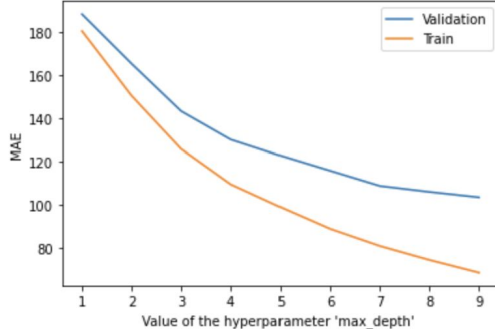


Nous pouvons voir une forte erreur de généralisation sur cette courbe peu importe la valeur de `n_estimators` mais avec très peu de progrès après ~80. On effectuera une recherche sur grille entre 40 et 80 pour cet hyperparamètre.

7. Modélisation de 'TotalGHGEmissions' : Optimisation et modèles finaux

suite de l'Optimisation du modèle RandomForestRegressor :

Validation curve for RandomForestRegressor model on the hyperparameter max_depth and score MAE



Cette seconde courbe nous permet de choisir une limite pour max_depth à 6 correspondant à un point d'inflexion et limitant autant que se peut l'erreur de généralisation.

La recherche sur grille nous permet de déterminer $n_estimators=70$ et $max_depth=5$ comme hyperparamètres optimisant la moyenne des MAE obtenues sur les échantillons de validation croisée (108.9).

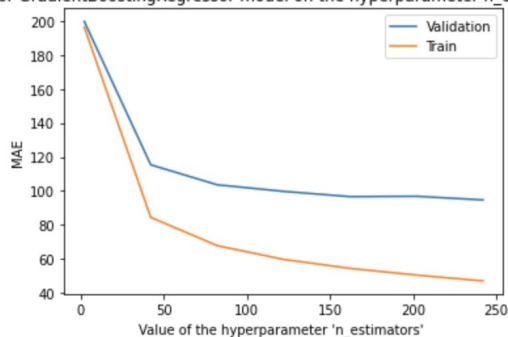
Lors de cette recherche sur grille nous testons également de modifier l'hyperparamètre 'bootstrap' en le qualifiant comme 'False' et 'criterion' en lui désignant l'erreur absolue comme fonction de perte. Comme pour la modélisation de la première variable cible, cette dernière valeur de 'criterion' est retenue et permet d'obtenir le score optimal dans les conditions décrites ci-dessus.

7. Modélisation de 'TotalGHGEmissions' : Optimisation et modèles finaux

Optimisation du modèle GradientBoostingRegressor :

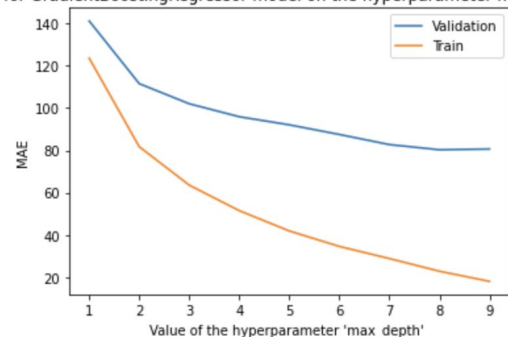
Nous choisissons pour ce modèle une optimisation des paramètres `n_estimators` (nombre d'estimateurs de base), `max_depth` (profondeur des estimateurs de base) et '`subsample`' afin de contrôler également la complexité du modèle (et donc sa capacité de généralisation). Nous optimisons aussi le taux d'apprentissage ('`learning_rate`').

Validation curve for GradientBoostingRegressor model on the hyperparameter `n_estimators` and score MAE



Aux dépens d'une capacité de généralisation affectée, nous choisissons ici de réaliser la recherche sur grille entre `n_estimators=50` et `n_estimators=150`.

Validation curve for GradientBoostingRegressor model on the hyperparameter `max_depth` and score MAE

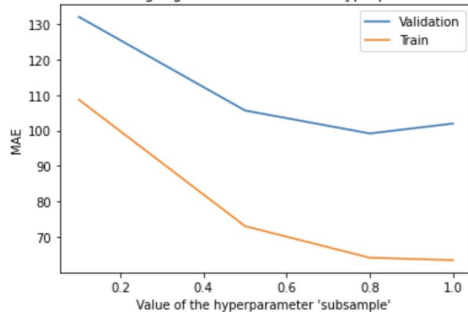


La courbe ci-dessus nous invite utiliser 7 comme limite à `max_depth` afin de limiter un overfitting inutile pour réduire l'erreur sur les jeux de validation.

7. Modélisation de 'TotalGHGEmissions' : Optimisation et modèles finaux

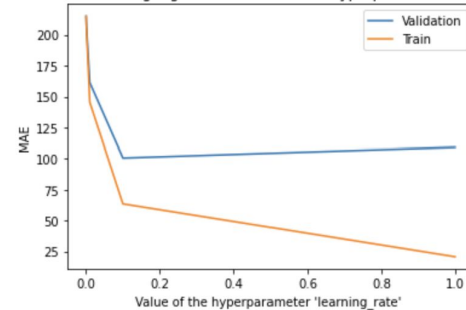
suite de l'Optimisation du modèle GradientBoostingRegressor :

Validation curve for GradientBoostingRegressor model on the hyperparameter subsample and score MAE



La courbe ci-dessus nous invite à chercher l'optimum de 'subsample' entre 0.5 et 0.8.

Validation curve for GradientBoostingRegressor model on the hyperparameter learning_rate and score MAE



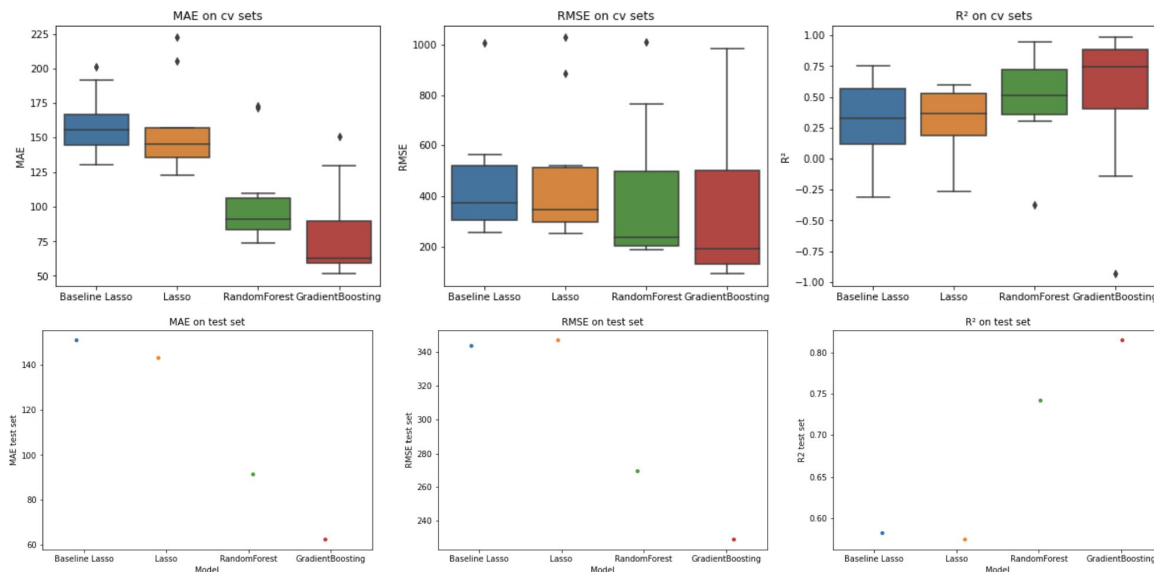
La courbe ci-dessus nous invite à chercher l'optimum de 'learning_rate' entre 0.01 et 0.2.

L'ensemble de ces optimisations nous a permis de déterminer, via une recherche sur grille, un score optimal sur les jeux de validation de 80.98 pour 'learning_rate'=0.1, 'max_depth'=7, 'n_estimators'=150 et 'subsample'=0.8.

7. Modélisation de 'TotalGHGEmissions' : Optimisation et modèles finaux

Comparaison des modèles et sélection du modèle final :


	Model	MAE train set	MAE cv set	MAE test set	RMSE train set	RMSE cv set	RMSE test set	R2 train set	R2 cv set	R2 test set
0	Baseline Lasso	1.477e+02	1.596e+02	150.90	4.146e+02	4.490e+02	343.9	0.661	0.297	0.582
1	Lasso	1.444e+02	1.557e+02	143.00	4.38e+02	4.732e+02	347.1	0.622	0.308	0.574
2	RandomForest	8.552e+01	1.053e+02	91.53	2.7e+02	3.903e+02	269.9	0.856	0.497	0.742
3	GradientBoosting	2.587e+01	8.036e+01	62.39	3.639e+01	3.484e+02	228.8	0.997	0.507	0.815




L'ensemble de ces résultats démontrent la supériorité du modèle basé sur l'algorithme de Gradient Boosting (malgré un surajustement de la variance du jeu d'entraînement), comme pour la première modélisation.

Modèle final retenu :

```
GradientBoostingRegressor(max_depth=7, n_estimators=150, subsample=0.8)
```

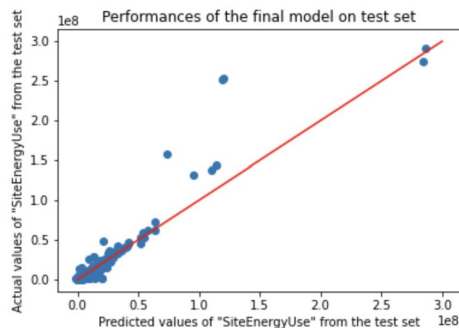


8. Conclusion et réflexion sur l'EnergySTAR score



8. Conclusion et réflexion sur l'EnergySTAR Score

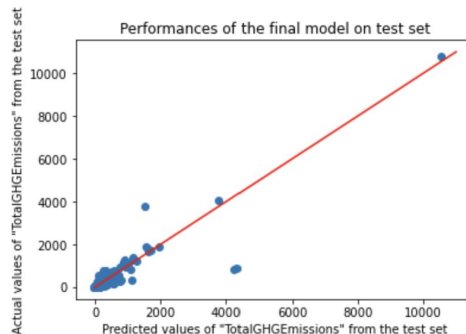
Comme souhaité, à partir de données que l'on peut retrouver sur un permis d'exploitation commerciale d'un bâtiment, nos deux modèles permettent respectivement d'estimer la consommation d'énergie et les émissions de CO2 de ce bâtiment.



[MAE sur test set]

[Valeur moyenne de 'SiteEnergyUse' sur test set]

≈ 25%



[MAE sur test set]

[Valeur moyenne de 'TotalGHGEmissions' sur test set]

≈ 38%

Comme on peut le voir sur ces graphiques et en assimilant les ratios ci-contre à une précision, on peut estimer que celle-ci soit suffisante pour notre problème.

Si ce n'est pas le cas, l'utilisation de l'EnergySTAR Score comme variable d'entrée de ce modèle permettra probablement d'améliorer ces performances mais il faudra pouvoir justifier l'intérêt du surcoût associé pour le gain de précision par rapport à ce simple modèle statistique.



Merci pour votre attention