

Projet 4 : Anticipez les besoins en consommation électrique de bâtiments

Lancelot LECLERCQ

14 décembre 2021

Sommaire

1. Introduction
2. Nettoyage du jeu de données
3. Étapes des modélisations
4. Modélisation des émissions de carbone
5. Modélisation de la consommation énergétique
6. Conclusion

Introduction

Problématique

- Objectif de la ville de Seattle : atteindre la neutralité en émissions de carbone
- La ville s'intéresse aux émissions des bâtiments non destinés à l'habitation
- Pour cela des relevés de consommation ont été réalisés mais ils sont coûteux à obtenir
- Est-il possible de prédire les émissions et de la consommation d'énergie pour des bâtiments pour lesquels les relevés n'ont pas été réalisés à partir des relevés déjà obtenus



Seattle

Jeu de données et modélisation

- Base de données issue de l'initiative de la ville de Seattle de proposer ses données en accès libre (Open Data)
- Données concernant les batiments de la ville, caractérise :
 - le type,
 - la surface,
 - le nombre d'étages,
 - la consommation énergétique,
 - les émissions de carbone,
 - :
- Données des années 2015 et 2016
- Objectif : trouver le modèle le plus performant
- Test de modèles de différents type :
 - Linéaires :
 - Ridge
 - Lasso
 - ElasticNet
 - Plus proches voisins :
 - KNeighborsRegressor
 - Ensemblistes :
 - RandomForestRegressor
 - AdaBoostRegressor
 - GradientBoostingRegressor
- GridSearch afin de trouver les paramètres optimaux pour chacun de ces modèles

Nettoyage du jeu de données

[illegible]

- Nombre de données par colonnes après suppression
des colonnes ayant moins de 50% de données
-
- | Indicateurs | Nombre de données |
|-------------------------------|-------------------|
| ENERGYSTARScore | 2200 |
| LatestPropertyUseType | 3200 |
| LargestPropertyUseTypeGFA | 3200 |
| ListOfAllPropertyUseTypes | 3200 |
| ZipCode | 3200 |
| GHGmissionsIntensity | 3200 |
| TotalGHGmissions | 3200 |
| NaturalGas(Kbu) | 3200 |
| Electricity(Kbu) | 3200 |
| Steamuse(Kbu) | 3200 |
| SiteEnergyuse(Kbu) | 3200 |
| SourceUnit(Kbu/sf) | 3200 |
| SiteEUt(Kbu/sf) | 3200 |
| NumberofFloors | 3200 |
| NumberofBuildings | 3200 |
| TaxParcelIdentificationNumber | 3200 |
| DefaultData | 3200 |
| PropertyName | 3200 |
| PrimaryPropertyType | 3200 |
| BuildingType | 3200 |
| CouncilDistrictCode | 3200 |
| Neighborhood | 3200 |
| YearBuilt | 3200 |
| PropertyGFABuilding(s) | 3200 |
| PropertyGFATotal | 3200 |
| PropertyGFAParking | 3200 |
| Longitude | 3200 |
| Address | 3200 |
| Latitude | 3200 |
| ComplianceStatus | 3200 |
| OSBuildingID | 3200 |

- 7/34

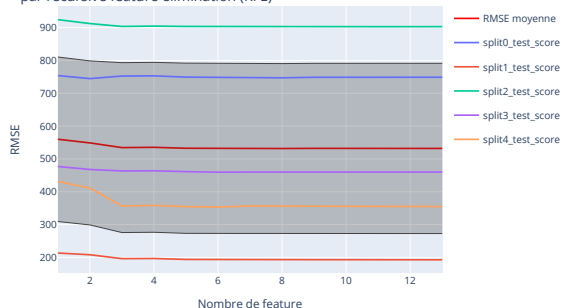
Nettoyage du jeu de données : Selections des variables

Élimination récursive des variables (RFE) et matrice de corrélation

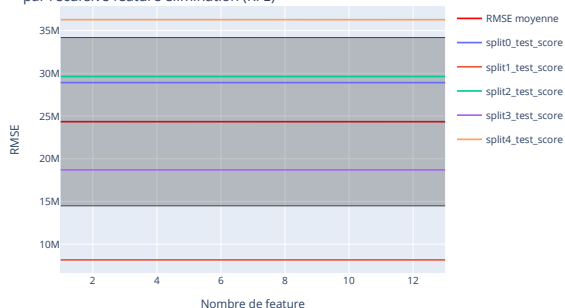
Variables pertinentes pour les émissions

Variables pertinentes pour la consommation

RMSE pour la variable TotalGHGEmissions en fonction du nombre de feature sélectionnées par recursive feature elimination (RFE)



RMSE pour la variable SiteEnergyUse en fonction du nombre de feature sélectionnées par recursive feature elimination (RFE)



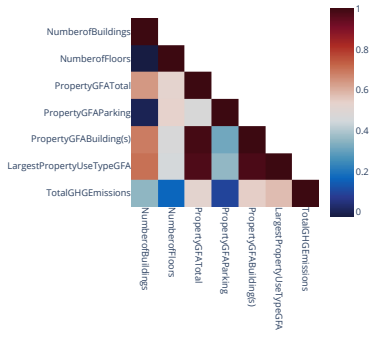
- Selection des variables les plus pertinentes par elimination recursive des variables (RFE)
- Réduction efficace pour les émissions
- Pas de réel changement de RMSE pour la consommation

Nettoyage du jeu de données : Selections des variables

Élimination récursive des variables (RFE) et matrice de corrélation

Variables pertinentes pour les émissions

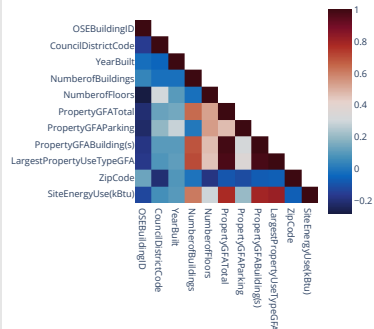
Matrice des corrélations sur les variables sélectionnées par RFE pour les émissions



- Observation des résultats de RFE par les matrices de corrélation
- Les variables les plus corrélées sont communes aux deux sélection
- Conservation de 6 variables jugées pertinentes

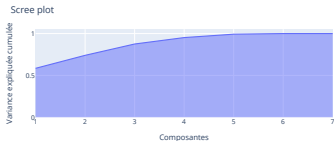
Variables pertinentes pour la consommation

Matrice des corrélations sur les variables sélectionnées par RFE pour la consommation



Nettoyage du jeu de données : Selections des variables

Analyse en composantes principales (PCA)

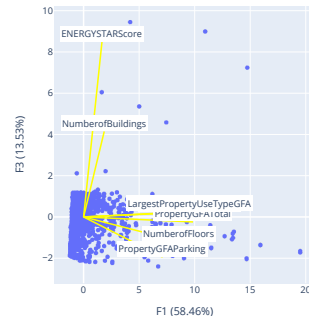


- Le graphique de la variance expliquée cumulée nous montre que 99% de la matrice est expliquée avec 5 variables
- Les quatres variables les plus corrélées se retrouvent sur l'axe F1
- L'EnergyStar score semble avoir une certaine importance car il explique une grande partie de l'axe F3

PCA F1 et F2



PCA F1 et F3

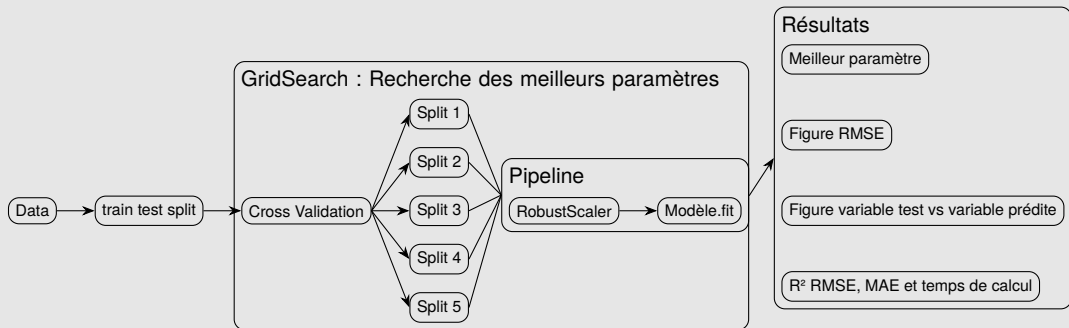


Étapes des modélisations

Étapes des modélisations

Afin de comparer les différents modèles

- Split commun à chaque modèle (varie selon la variable modélisée)
- Pour chaque modèle (boucle) :
 - GridSearch des meilleurs paramètres avec validation croisée
 - Création d'un pipeline : scaling et fit du modèle
 - Scaling par RobustScaler car plus résistant aux valeurs aberrantes selon la documentation
- La boucle retourne :
 - Le(s) meilleur(s) paramètre(s) (gridsearch)
 - La RMSE en fonction du paramètre le plus évolutif (validation croisée)
 - La figure de la variable étudiée vs ses prédictions
 - Le R^2 , la RMSE, la MAE (mean absolute error) et le temps de calcul du modèle

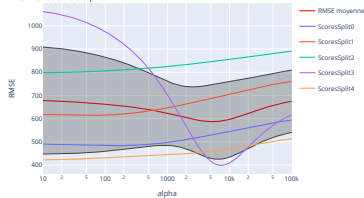


Modélisation émissions

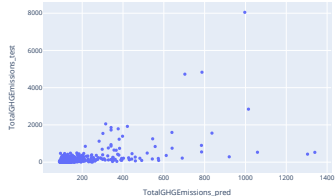
Modèle Ridge

Variable non modifiée

RMSE du modèle Ridge
pour la variable TotalGHGEmissions
en fonction de alpha



Visualisation des données de TotalGHGEmissions
prédites par le modèle Ridge()
vs les données test



←

paramètre	Ridge()
alpha	5094.14

←

paramètre	Ridge()
alpha	6428.07

- Modèle de régression linéaire
introduisant un coefficient cherchant
à minimiser l'erreur quadratique

←

R ²	RMSE	MAE	MAE%	FitTime(s)
0.24	423.80	150.95	5.72	0.01

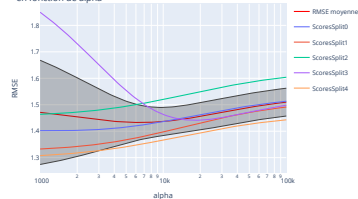
←

R ²	RMSE	MAE	MAE%	FitTime(s)
0.16	487.86	135.35	2.12	0.02

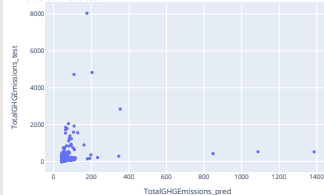
←

Variable au log

RMSE du modèle Ridge
pour la variable TotalGHGEmissions_log
en fonction de alpha



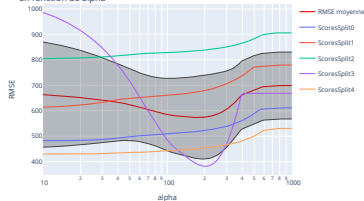
Visualisation des données de TotalGHGEmissions_log
prédites par le modèle Ridge()
vs les données test



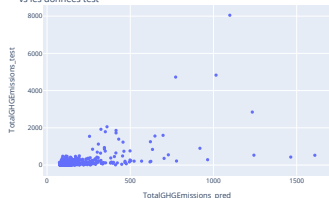
Modèle Lasso

Variable non modifiée

RMSE du modèle Lasso
pour la variable TotalGHGEmissions
en fonction de alpha



Visualisation des données de TotalGHGEmissions
prédites par le modèle Lasso()
vs les données test



←

paramètre	Lasso()
alpha	178.86

←

paramètre	Lasso()
alpha	0.34

⇒

⇒

- Similaire à la regression ridge
- Coefficient est réduit à zéro pour les variables peu corrélées
- Peut être utilisé pour la sélection de feature

←

R ²	RMSE	MAE	MAE%	FitTime(s)
0.26	417.95	150.97	5.52	0.02

←

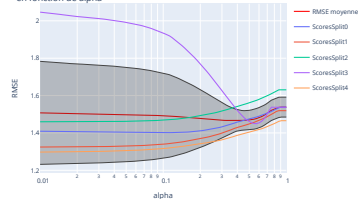
R ²	RMSE	MAE	MAE%	FitTime(s)
0.12	490.73	136.13	2.25	0.02

⇒

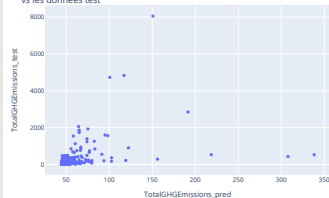
⇒

Variable au log

RMSE du modèle Lasso
pour la variable TotalGHGEmissions_log
en fonction de alpha



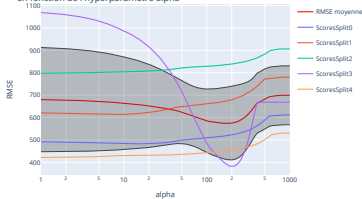
Visualisation des données de TotalGHGEmissions_log
prédites par le modèle Lasso()
vs les données test



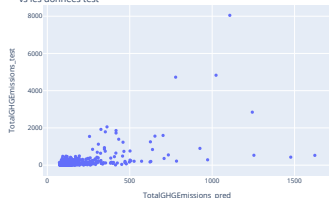
Modèle ElasticNet

Variable non modifiée

RMSE du modèle ElasticNet pour la variable
TotalGHGEmissions avec le paramètre l1_ratio=1.0
en fonction de l'hyperparamètre alpha



Visualisation des données de TotalGHGEmissions
prédites par le modèle ElasticNet()
vs les données test



paramètre	ElasticNet()
alpha	174.75
l1_ratio	1.00



paramètre	ElasticNet()
alpha	1.29
l1_ratio	0.10

- Combine les coefficients des
regressions ridge et lasso



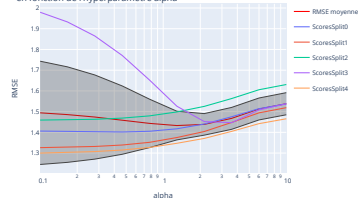
R ²	RMSE	MAE	MAE%	FitTime(s)
0.26	417.53	150.73	5.48	0.01



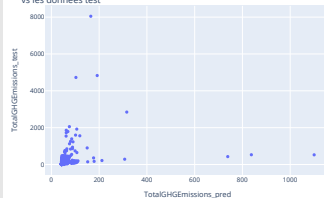
R ²	RMSE	MAE	MAE%	FitTime(s)
0.16	487.75	134.58	2.13	0.02

Variable au log

RMSE du modèle ElasticNet pour la variable
TotalGHGEmissions_log avec le paramètre l1_ratio=0.1
en fonction de l'hyperparamètre alpha



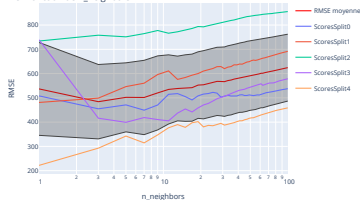
Visualisation des données de TotalGHGEmissions_log
prédites par le modèle ElasticNet()
vs les données test



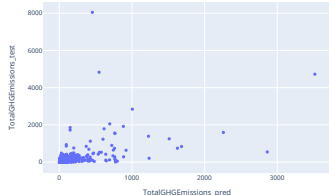
Modèle kNeighborsRegressor

Variable non modifiée

RMSE du modèle KNeighborsRegressor pour la variable TotalGHGEmissions en fonction de n_neighbors



Visualisation des données de TotalGHGEmissions prédites par le modèle KNeighborsRegressor() vs les données test



paramètre	KNeighborsRegressor()
n_neighbors	3



paramètre	KNeighborsRegressor()
n_neighbors	1



- Prédiction par interpolation avec les plus proches voisins dans le jeu de données



R ²	RMSE	MAE	MAE%	FitTime(s)
0.26	418.44	119.52	1.99	0.02

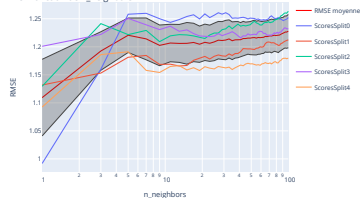


R ²	RMSE	MAE	MAE%	FitTime(s)
0.52	401.17	73.27	0.75	0.02

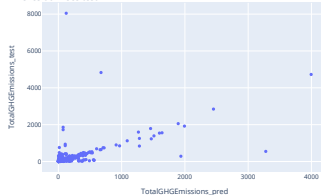


Variable au log

RMSE du modèle KNeighborsRegressor pour la variable TotalGHGEmissions_log en fonction de n_neighbors



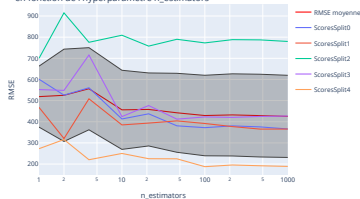
Visualisation des données de TotalGHGEmissions_log prédites par le modèle KNeighborsRegressor() vs les données test



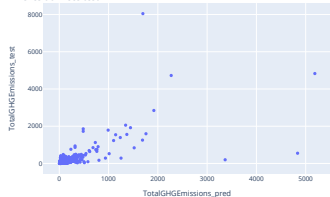
Modèle RandomForestRegressor

Variable non modifiée

RMSE du modèle RandomForestRegressor pour la variable TotalGHGEmissions avec le paramètre max_features=auto en fonction de l'hyperparamètre n_estimators



Visualisation des données de TotalGHGEmissions prédites par le modèle RandomForestRegressor() vs les données test



paramètre	RandomForestRegressor()
n_estimators	1000
max_features	auto



paramètre	RandomForestRegressor()
n_estimators	464
max_features	sqrt

- Classification des valeurs à partir d'arbre de décision aléatoire
- Prédiction à partir de ces classifieurs



R ²	RMSE	MAE	MAE%	FitTime(s)
0.42	371.52	89.73	1.44	11.48

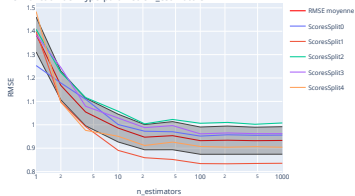


R ²	RMSE	MAE	MAE%	FitTime(s)
0.68	381.25	85.76	0.72	3.01

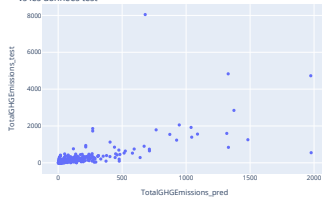


Variable au log

RMSE du modèle RandomForestRegressor pour la variable TotalGHGEmissions_log avec le paramètre max_features=sqrt en fonction de l'hyperparamètre n_estimators



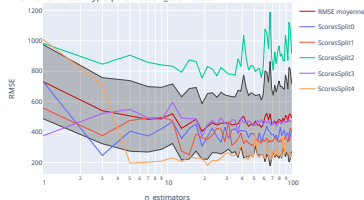
Visualisation des données de TotalGHGEmissions_log prédites par le modèle RandomForestRegressor() vs les données test



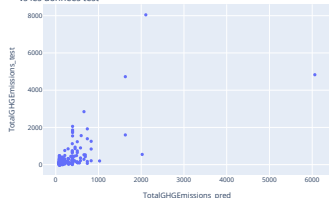
Modèle AdaBoostRegressor

Variable non modifiée

RMSE du modèle AdaBoostRegressor pour la variable TotalGHGEmissions avec le paramètre loss=square en fonction de l'hyperparamètre n_estimators



Visualisation des données de TotalGHGEmissions prédites par le modèle AdaBoostRegressor() vs les données test



paramètre	AdaBoostRegressor()
n_estimators	19
loss	square



paramètre	AdaBoostRegressor()
n_estimators	15
loss	linear



- Même principe que les forêts aléatoires
- Utilisation d'apprenants faibles (légèrement plus performants que la prédiction aléatoire similaire à de petits arbre de décision)
- Les prédictions des apprenants sont combinées avec un coefficient de poids
- À chaque itération le poids des mauvaises prédictions est augmenté ce qui pousse le modèle à se concentrer dessus



R ²	RMSE	MAE	MAE%	FitTime(s)
0.48	351.77	136.67	4.99	0.09

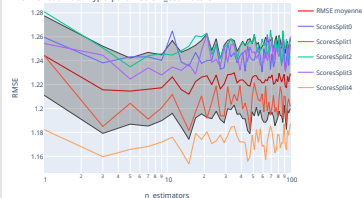


R ²	RMSE	MAE	MAE%	FitTime(s)
0.36	404.36	118.82	1.27	0.09

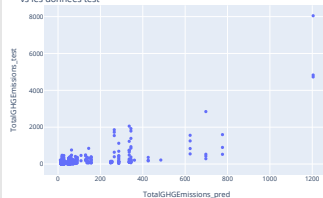


Variable au log

RMSE du modèle AdaBoostRegressor pour la variable TotalGHGEmissions_log avec le paramètre loss=linear en fonction de l'hyperparamètre n_estimators



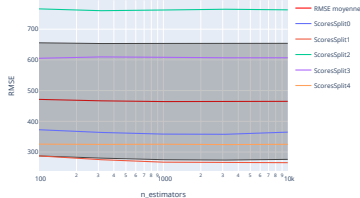
Visualisation des données de TotalGHGEmissions_log prédites par le modèle AdaBoostRegressor() vs les données test



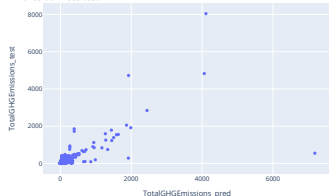
Modèle GradientBoostingRegressor

Variable non modifiée

RMSE du modèle GradientBoostingRegressor pour la variable TotalGHGEmissions avec le paramètre loss=squared_error en fonction de l'hyperparamètre n_estimators



Visualisation des données de TotalGHGEmissions prédites par le modèle GradientBoostingRegressor() vs les données test



paramètre	GradientBoostingRegressor()
n_estimators	3162
loss	squared_error

paramètre	GradientBoostingRegressor()
n_estimators	5623
loss	huber

- Similaire à AdaBoostRegressor
- Prend en compte une fonction objectif (loss fonction) plus complexe afin d'améliorer l'optimisation



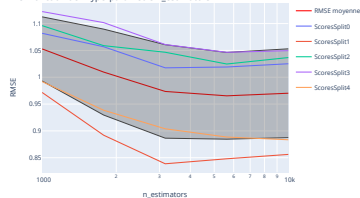
R ²	RMSE	MAE	MAE%	FitTime(s)
0.47	355.84	74.99	1.34	10.37

R ²	RMSE	MAE	MAE%	FitTime(s)
0.63	340.24	71.60	0.80	55.91

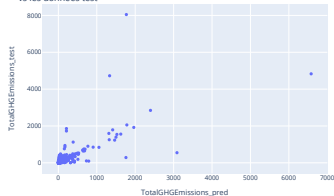


Variable au log

RMSE du modèle GradientBoostingRegressor pour la variable TotalGHGEmissions_log avec le paramètre loss=huber en fonction de l'hyperparamètre n_estimators

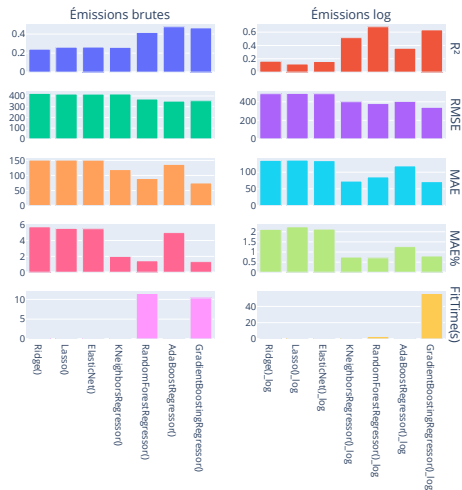


Visualisation des données de TotalGHGEmissions_log prédites par le modèle GradientBoostingRegressor() vs les données test



Comparaison des résultats selon que la variable est au log ou non

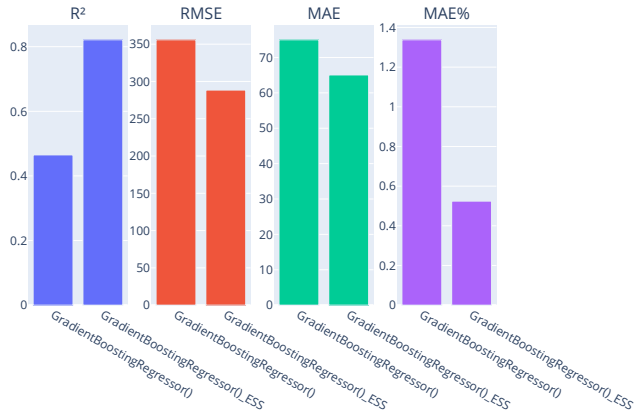
Comparaison des scores des modèles d'émissions



- RandomForestRegressor, AdaBoostRegressor et GradientBoostingRegressor ont des erreurs moins importantes et un R^2 plus grand quelque soit la variable modélisée
- KNeighborsRegressor est plus performant avec la variable au log
- Modèles linéaire : Ridge, Lasso et ElasticNet moins efficaces avec la variable au log
- Temps de modélisation de RandomForestRegressor et GradientBoostingRegressor plus importants que les autres
- Temps de modélisation de RandomForestRegressor avec la variable au log moindre qu'avec la variable non modifiée

Influence de l'EnergyStar score sur la prédiction des Émissions

Comparaison avec et sans ajout de l'energy score stars



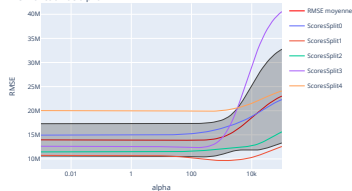
- GradientBoostingRegressor avec la variable au log (RMSE la plus petite)
- L'EnergyStar score améliore la RMSE
- Amélioration des autres mesures d'erreur et de corrélation

Modélisation consommation

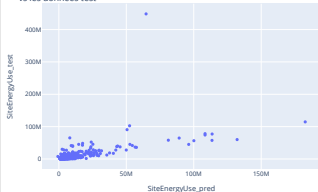
Modèle Ridge

Variable non modifiée

RMSE du modèle Ridge
pour la variable SiteEnergyUse
en fonction de alpha



Visualisation des données de SiteEnergyUse
prédites par le modèle Ridge()
vs les données test



paramètre	Ridge()
alpha	102.35

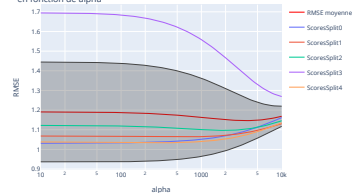
paramètre	Ridge()
alpha	3511.19

R ²	RMSE	MAE	MAE%	FitTime(s)
0.33	17660078.37	5153567.28	1.85	0.01

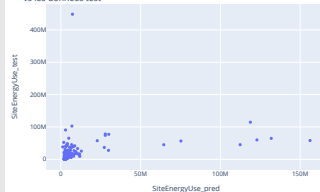
R ²	RMSE	MAE	MAE%	FitTime(s)
0.31	21043685.67	5666820.77	1.40	0.02

Variable au log

RMSE du modèle Ridge
pour la variable SiteEnergyUse_log
en fonction de alpha



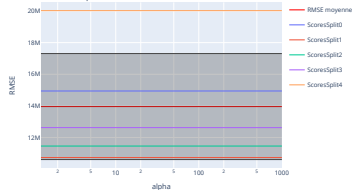
Visualisation des données de SiteEnergyUse_log
prédites par le modèle Ridge()
vs les données test



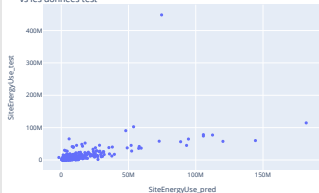
Modèle Lasso

Variable non modifiée

RMSE du modèle Lasso
pour la variable SiteEnergyUse
en fonction de alpha



Visualisation des données de SiteEnergyUse
prédites par le modèle Lasso()
vs les données test



paramètre	Lasso()
alpha	1000.00

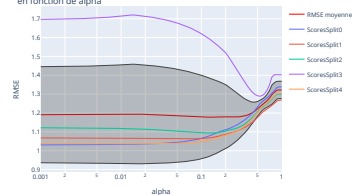
paramètre	Lasso()
alpha	0.12

R ²	RMSE	MAE	MAE%	FitTime(s)
0.34	17499302.40	5269886.33	1.88	0.04

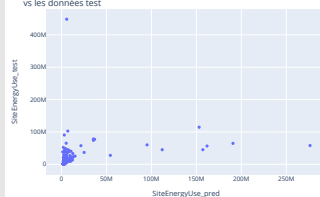
R ²	RMSE	MAE	MAE%	FitTime(s)
0.32	23496263.51	6175023.22	1.38	0.02

Variable au log

RMSE du modèle Lasso
pour la variable SiteEnergyUse_log
en fonction de alpha



Visualisation des données de SiteEnergyUse_log
prédites par le modèle Lasso()
vs les données test



Variable non modifiée

The graph displays the Root Mean Square Error (RMSE) on the y-axis (ranging from 10M to 40M) against the parameter α on the x-axis (logarithmic scale from 0.001 to 1000). The legend identifies six data series: RMSE moyenne (red), ScoresSplit0 (blue), ScoresSplit1 (orange), ScoresSplit2 (green), ScoresSplit3 (purple), and ScoresSplit4 (brown). ScoresSplit3 exhibits a significant increase in RMSE as α increases, starting around 15M and rising to nearly 40M. The other splits show much more stable performance, with RMSE values generally between 10M and 25M, and a slight upward trend as α increases.

paramètre	ElasticNet()
alpha	0.09
l1_ratio	0.46

paramètre	ElasticNet()
alpha	0.89
l1 ratio	0.10

R ²	RMSE	MAE	MAE%	FitTime(s)
0.33	17669838.00	5135486.35	1.85	0.03

R ²	RMSE	MAE	MAE%	FitTime(s)
0.30	20734563.65	5593976.90	1.41	0.02

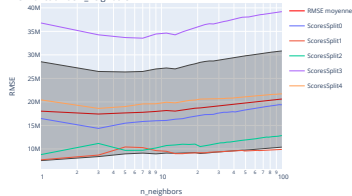
Variable au log

Figure 1 is a line plot showing the Root Mean Square Error (RMSE) on the y-axis (ranging from 0.9 to 1.7) against the regularization parameter α on the x-axis (logarithmic scale from 0.001 to 10). The plot includes a thick grey shaded area representing the range of RMSE values and several colored lines representing different scores. The legend indicates the following series: RMSE moyenne (red), ScoresSplit0 (blue), ScoresSplit1 (orange), ScoresSplit2 (green), ScoresSplit3 (purple), and ScoresSplit4 (brown). The purple line (ScoresSplit3) starts at a high RMSE of approximately 1.7 for small α and decreases sharply to about 1.25 at $\alpha=1$, then rises slightly. The other lines are clustered between 1.0 and 1.2, with the RMSE moyenne line (red) being the highest among them, starting around 1.1 and decreasing to about 1.05 at $\alpha=1$.

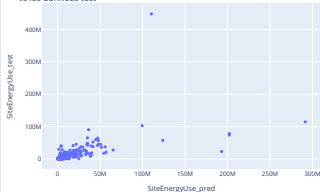
Modèle kNeighborsRegressor

Variable non modifiée

RMSE du modèle KNeighborsRegressor pour la variable SiteEnergyUse en fonction de $n_neighbors$



Visualisation des données de SiteEnergyUse prédites par le modèle KNeighborsRegressor() vs les données test



←

paramètre	KNeighborsRegressor()
$n_neighbors$	3

←

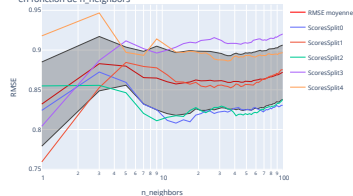
paramètre	KNeighborsRegressor()
$n_neighbors$	1

⇒

⇒

Variable au log

RMSE du modèle KNeighborsRegressor pour la variable SiteEnergyUse_log en fonction de $n_neighbors$



←

R^2	RMSE	MAE	MAE%	FitTime(s)
0.15	19891776.59	4958197.14	1.14	0.02

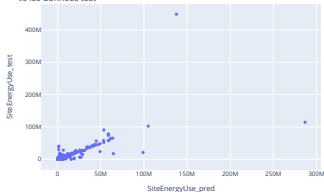
←

R^2	RMSE	MAE	MAE%	FitTime(s)
0.75	15125790.61	2521110.46	0.55	0.01

⇒

⇒

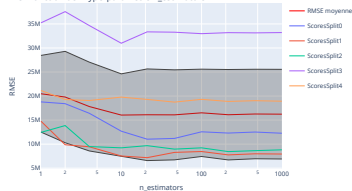
Visualisation des données de SiteEnergyUse_log prédites par le modèle KNeighborsRegressor() vs les données test



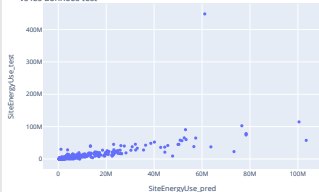
Modèle RandomForestRegressor

Variable non modifiée

RMSE du modèle RandomForestRegressor pour la variable SiteEnergyUse avec le paramètre max_features=log2 en fonction de l'hyperparamètre n_estimators



Visualisation des données de SiteEnergyUse prédites par le modèle RandomForestRegressor() vs les données test



paramètre	RandomForestRegressor()
n_estimators	10
max_features	log2

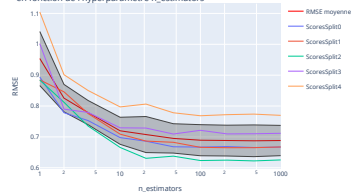


paramètre	RandomForestRegressor()
n_estimators	464
max_features	sqrt

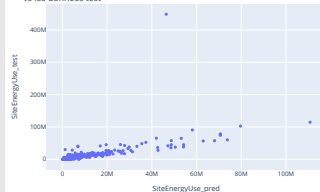


Variable au log

RMSE du modèle RandomForestRegressor pour la variable SiteEnergyUse_log avec le paramètre max_features=sqrt en fonction de l'hyperparamètre n_estimators



Visualisation des données de SiteEnergyUse_log prédites par le modèle RandomForestRegressor() vs les données test



R ²	RMSE	MAE	MAE%	FitTime(s)
0.43	16255496.44	3079266.36	0.85	0.09



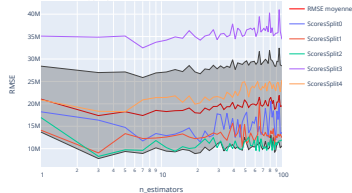
R ²	RMSE	MAE	MAE%	FitTime(s)
0.80	16533804.87	2771107.51	0.51	2.72



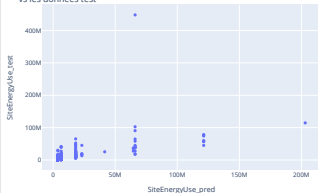
Modèle AdaBoostRegressor

Variable non modifiée

RMSE du modèle AdaBoostRegressor pour la variable SiteEnergyUse avec le paramètre loss=linear en fonction de l'hyperparamètre n_estimators



Visualisation des données de SiteEnergyUse prédites par le modèle AdaBoostRegressor() vs les données test



paramètre	AdaBoostRegressor()
n_estimators	3
loss	linear

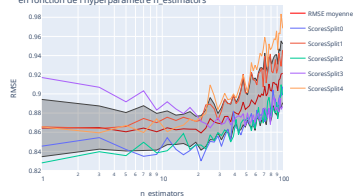
paramètre	AdaBoostRegressor()
n_estimators	21
loss	exponential

R ²	RMSE	MAE	MAE%	FitTime(s)
0.28	18239692.73	5482794.58	2.41	0.05

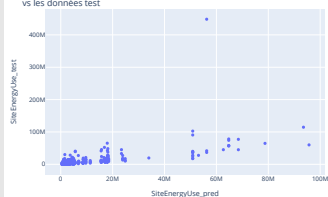
R ²	RMSE	MAE	MAE%	FitTime(s)
0.57	17101356.19	4203072.55	0.83	0.13

Variable au log

RMSE du modèle AdaBoostRegressor pour la variable SiteEnergyUse_log avec le paramètre loss=exponential en fonction de l'hyperparamètre n_estimators



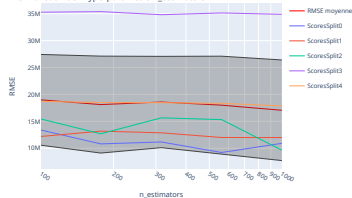
Visualisation des données de SiteEnergyUse_log prédites par le modèle AdaBoostRegressor() vs les données test



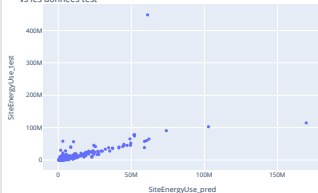
Modèle GradientBoostingRegressor

Variable non modifiée

RMSE du modèle GradientBoostingRegressor pour la variable SiteEnergyUse avec le paramètre loss=huber en fonction de l'hyperparamètre n_estimators



Visualisation des données de SiteEnergyUse prédites par le modèle GradientBoostingRegressor() vs les données test



paramètre	GradientBoostingRegressor()
n_estimators	1000
loss	huber

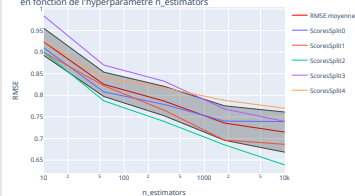


paramètre	GradientBoostingRegressor()
n_estimators	10000
loss	huber

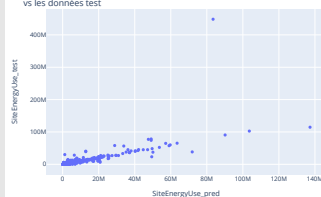


Variable au log

RMSE du modèle GradientBoostingRegressor pour la variable SiteEnergyUse_log avec le paramètre loss=huber en fonction de l'hyperparamètre n_estimators



Visualisation des données de SiteEnergyUse_log prédites par le modèle GradientBoostingRegressor() vs les données test



R ²	RMSE	MAE	MAE%	FitTime(s)
0.43	16292946.43	2980171.79	0.90	7.99

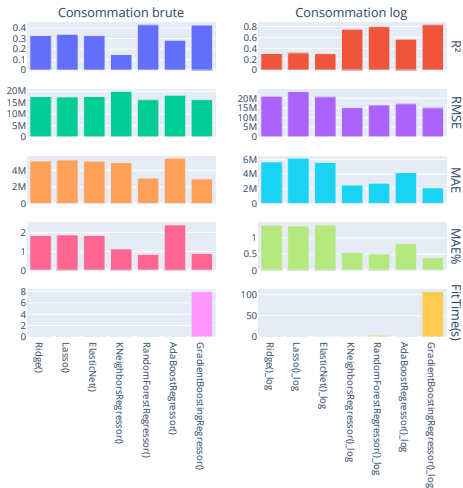


R ²	RMSE	MAE	MAE%	FitTime(s)
0.83	15038028.44	2135408.64	0.39	107.33



Comparaison des résultats selon que la variable est au log ou non

Comparaison des scores des modèles de consommation



- RMSE KNeighborsRegressor, RandomForestRegressor, AdaBoostRegressor et GradientBoostingRegressor inférieures avec la variable au log
- RMSE de RandomForestRegressor et GradientBoostingRegressor légèrement inférieures quelque soit la variable
- MAE de RandomForestRegressor et GradientBoostingRegressor plus significativement inférieures quelque soit la variable
- Temps de modélisation plus important pour GradientBoostingRegressor

Conclusion

Conclusion : Meilleurs modèles

- GradientBoostingRegressor est le modèle le plus performant dans les deux cas
- Cependant plus gourmand en ressources/temps de calcul
- Peut-être plus difficile à utiliser sur des jeu de données plus importants
- RandomForestRegressor semble être un bon compromis entre performance et temps de calcul
- KNeighborsRegressor semble aussi bien se défendre

Conclusion

- Découverte des différents modèles et de leur fonctionnement
- Obtention avec certains modèles d'une estimation avec moins de 1% d'écart à la moyenne absolue
- Si de nouveaux bâtiments ont été construits il peut être intéressant de rentrer leurs caractéristiques dans notre base de donnée et voir si on peut prédire leurs émissions et consommation quitte à faire des mesures pour estimer si ces prédictions sont bonnes