



# Parcours OpenClassRooms

## Data Scientist

---

### P4 Prédire les besoins en consommation électrique

Développé sur un Notebook Jupyter Colaboratory



# Sommaire

I. Problématique et pistes de recherches

II. Exploration, cleaning, feature engineering

III. Modélisations

IV. Optimisations

V. Conclusion

# I. Problématique et pistes de recherches

## Problématique

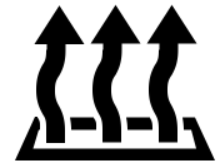
Prédire la consommation en énergie et le dégagement de CO<sub>2</sub> des bâtiments de Seattle.



Energy consumption



Seattle City



Heat emissions



CO<sub>2</sub> emissions

\$\$\$

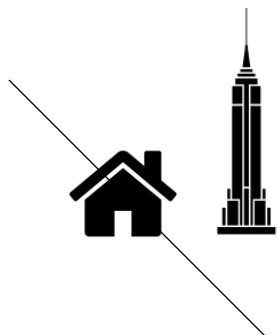


Thermometer

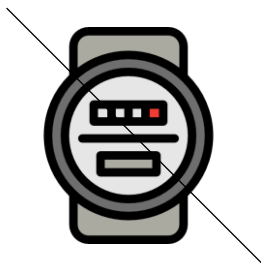
## Pistes de recherche



Distinction entre **consommations** et **émissions**.



On ne garde que les données des bâtiments non destinés à l'habitation



On essaie de se passer des **relevés annuels**.

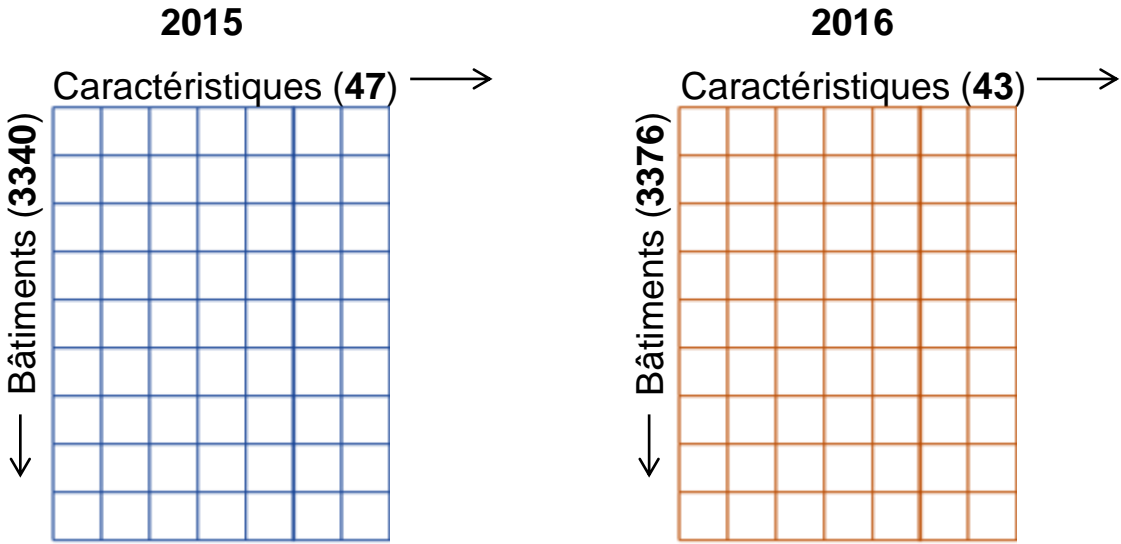


On estimera l'intérêt d'**ENERGYSTARScore**.

# Présentation des données

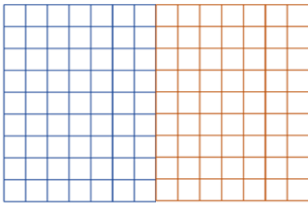
Le jeu de données se compose de deux tableaux, pour les années 2015 et 2016.

Chaque **ligne** correspond à un **bâtiment**, décrit par les **caractéristiques** en **colonnes**

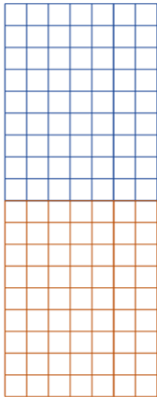


Plusieurs possibilités pour les rassembler :

1. La fusion

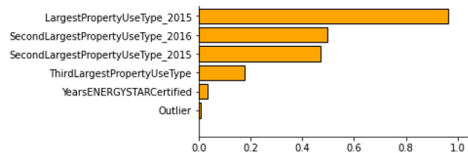


2. La concaténation



## II. Exploration, cleaning, feature engineering

### 1. Nos données sont-elles exploitables ?

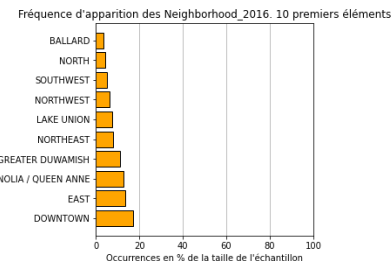
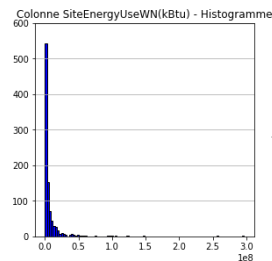


Proportion de NaN

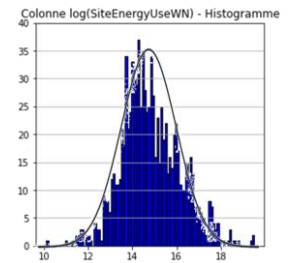
### 2. Que contiennent nos données ?



Site Web de Seattle

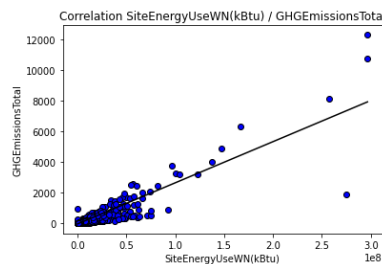


Histogrammes

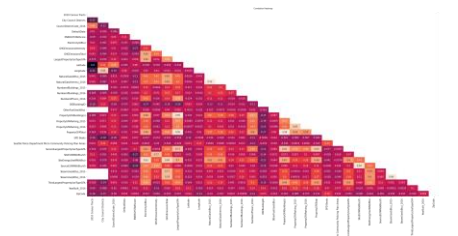


Transformation logarithmique

### 3. Quelles sont les données utiles ?



Corrélations (targets)



Corrélations (features)

## Que contiennent les données ?



Pour la majorité des caractéristiques, un descriptif est disponible sur le site internet de la municipalité de Seattle.

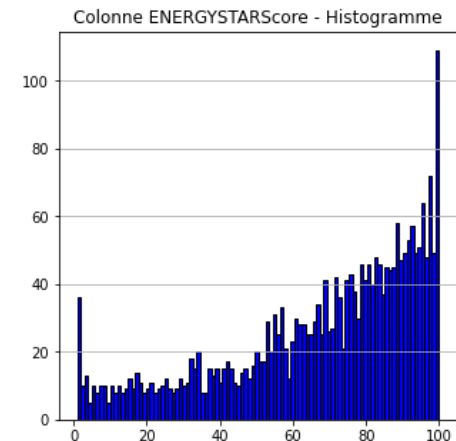
### ENERGYSTARScore

*An EPA calculated 1-100 rating that assesses a property's overall energy performance, based on national data to control for differences among climate, building uses, and operations. A score of 50 represents the national median.*

On peut distinguer plusieurs catégories de features :

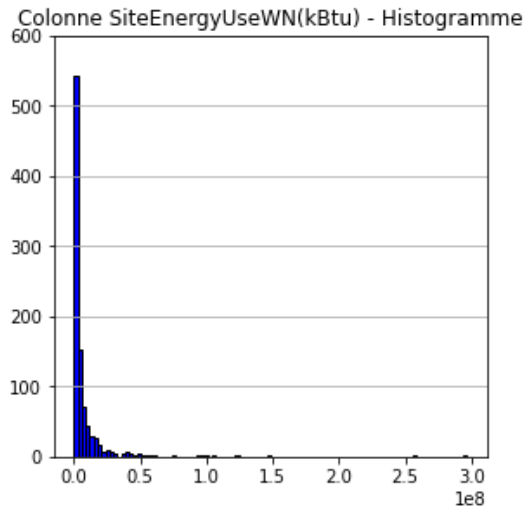
- Les données propres aux bâtiments
- Les données concernant l'utilisation des bâtiments
- Les relevés annuels (dont on cherche à se passer)

2016



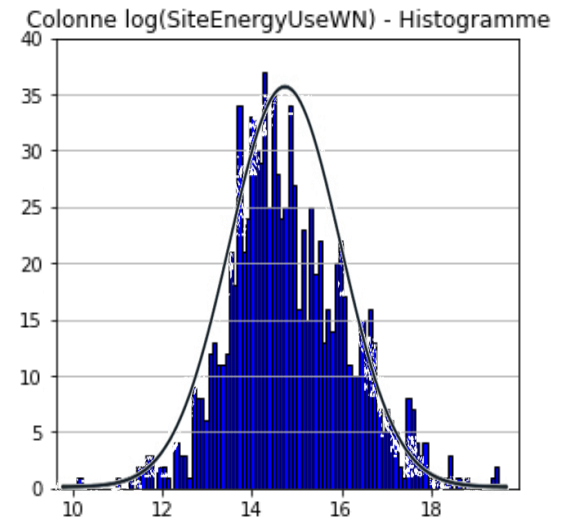
## Quelles sont les données utiles ?

On transforme certaines caractéristiques de manière à obtenir une distribution gaussienne (passage au log).



$x : \log(x)$

→



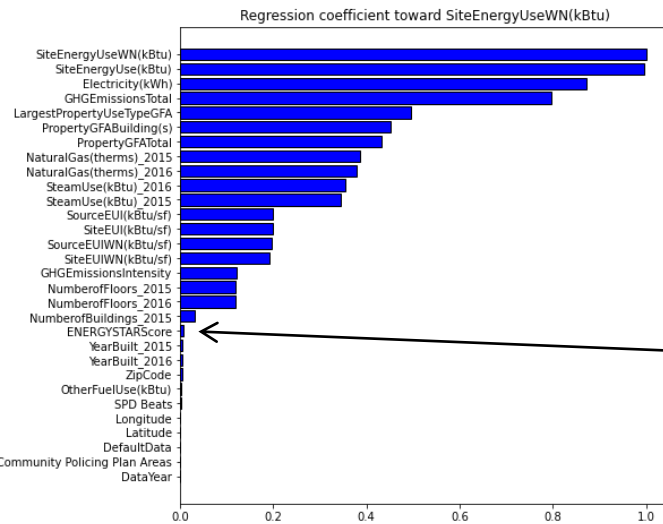
## Intérêt de la transformation

→ Sous cette forme, la caractéristique peut être utilisée pour des calculs statistiques.



## Quelles sont les données utiles ?

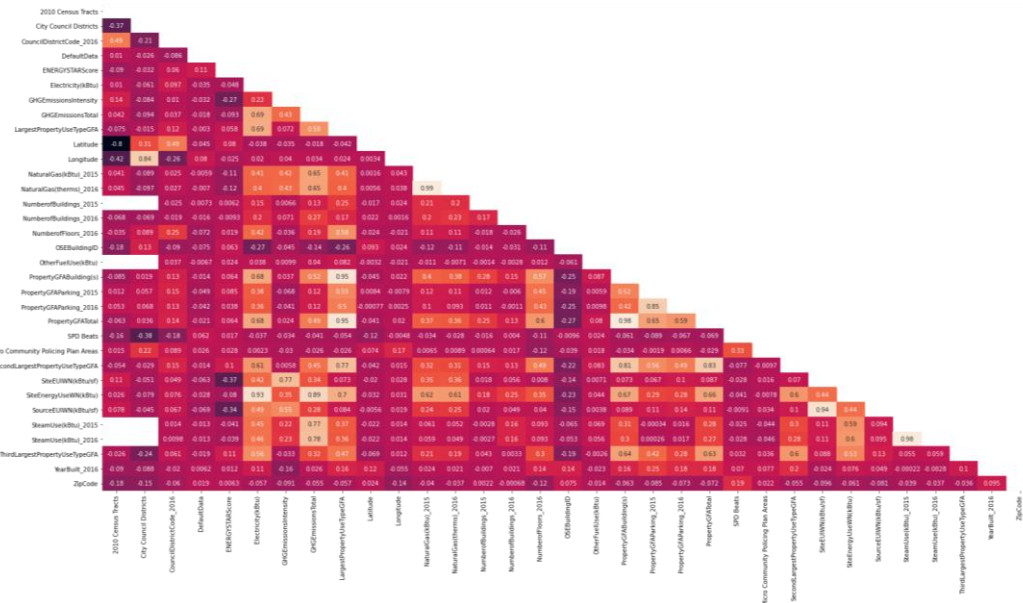
## Etude des corrélations



Avec une colonne target, ici SiteEnergyUseWN(kBtu)

L'ENERGYSTARScore ne semble pas très corrélé avec la target SiteEnergyUseWN(kBtu).

Toutes les colonnes entre elles, à l'aide d'une heatmap



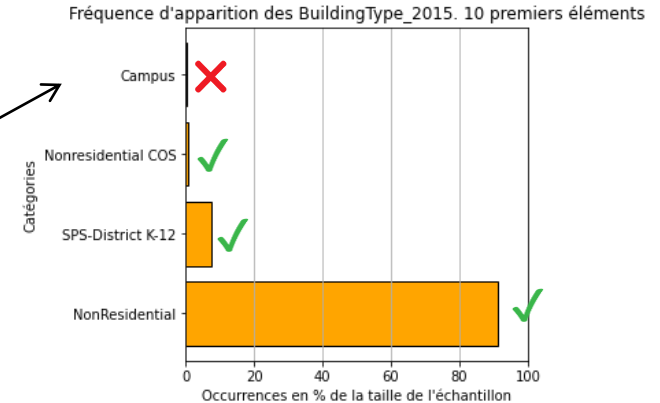
# Feature Engineering

## OneHotEncoder

Le OneHotEncoder transforme naturellement nos colonnes catégorielles en **102** colonnes.



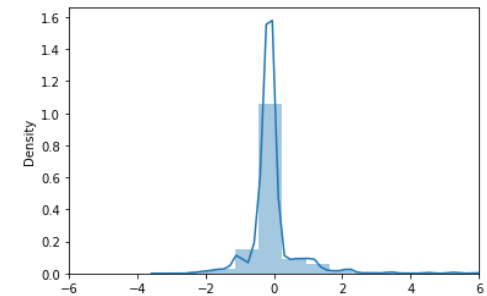
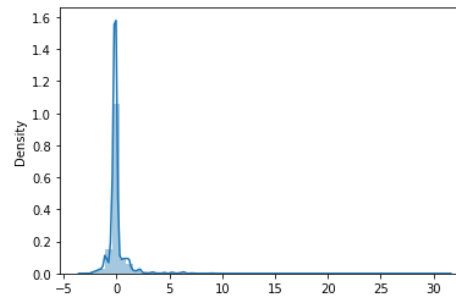
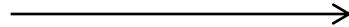
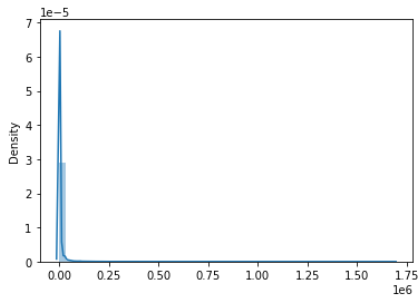
Cela semble beaucoup : nous allons rejeter les éléments comptant pour moins de 1% de la colonne (c'est-à-dire, pour une colonne de 1491 éléments, moins de 14 occurrences).



## StandardScaler

Grâce au passage au log, nous avons des données quantitatives avec des distributions normales (ou approchées).

Pour chaque colonne de caractéristique quantitative, on ramène la plus grande partie des valeurs dans l'intervalle  $[-1 ; 1]$  (les 6 sigma).



### III. Modélisation

#### Sept modèle essayés :

##### 1. Linéaires

1. LinearRegression
2. Ridge
3. Lasso
4. Elasticnet

##### 2. Arbres

5. DecisionTreeRegressor

##### 3. Ensembles non linéaires

1. RandomForest
2. GradientBoosting
3. XGBoost

##### 4. Dummy Regressor

1. Mean
2. Median

#### Métriques utilisées

1. R2
2. Root Mean Squared Error

#### Autres métriques possibles

1. Mean Average Error
2. Median Average Error

# Stratégie

## Model selection

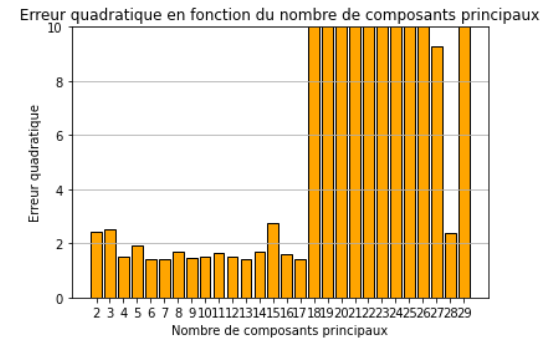
1. Split 80/20

2. PCA

3. Validation croisée (CV)

4. Validation croisée mélangée (SCV)

Comparaison sur  
le **test** set



Comparaison sur  
le **train** set



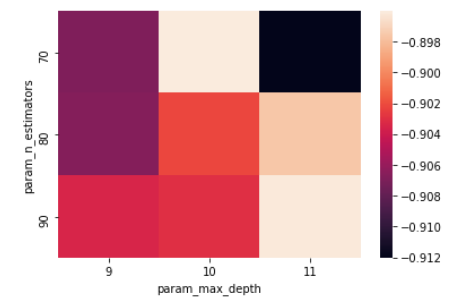
## Optimisations

1. Paramètres optimaux

2. GridSearchCV

3. GridSearchCV + CV

4. GridSearchCV + SCV



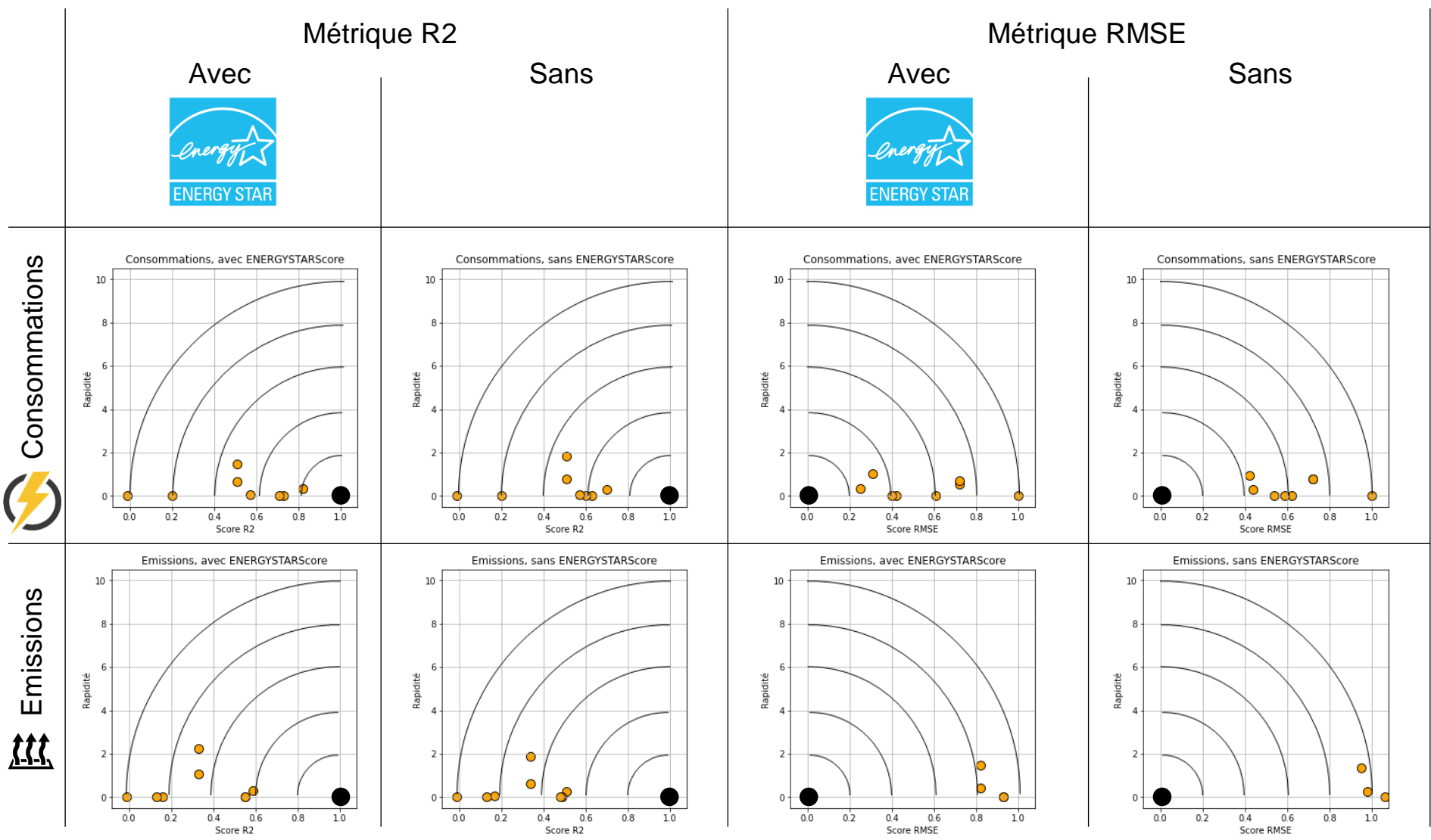
## Comparaison des modèles

	Model	Config	Métrique	Score	Rapidité
0	LinearRegression	Split	R2	0.710000	0.02
2	LinearRegression	PCA	R2	0.510000	2.31
4	Ridge	Split	R2	0.730000	0.02
6	Ridge	PCA	R2	0.510000	0.85
8	Lasso	Split	R2	-0.010000	0.01
10	Lasso	PCA	R2	NaN	NaN
12	ElasticNet	Split	R2	0.200000	0.01
14	ElasticNet	PCA	R2	NaN	NaN
16	DecisionTreeRegressor	Split	R2	0.570000	0.04
19	GradientBoosting	Split	R2	0.820000	0.32
22	DummyRegressor	Mean	R2	-0.000283	0.00
23	DummyRegressor	Median	R2	-0.014922	0.00





# Comparaison des modèles





Le point noir ● correspond au cas idéal : prédiction parfaite, temps de calcul nul.

Un point orange ● correspond à un modèle.



## Bilan des meilleures performances (avant optimisation)

Données de test utilisées	Métrique	R <sup>2</sup>		RMSE	
	ENERGYSTARScore	Avec 	Sans	Avec 	Sans
<b>Consommations</b> 	Meilleur modèle	GradientBoosting	GradientBoosting	Gradientboosting	RandomForest
	Prédiction	0,82	0,7	0,25	0,42
	Rapidité	0,32 s	0,29 s	0,32 s	0,87 s
<b>Emissions</b> 	Meilleur modèle	Gradientboosting	GradientBoosting	Gradientboosting	GradientBoosting
	Prédiction	0,59	0,51	0,82	0,93
	Rapidité	0,30 s	0,28 s	0,31 s	0,88 s

Données d'entraînement uniquement	Métrique	R <sup>2</sup>		RMSE	
	ENERGYSTARScore	Avec 	Sans	Avec 	Sans
<b>Consommations</b> 	Meilleur modèle	GradientBoosting	GradientBoosting	GradientBoosting	GradientBoosting
	Pré-optimisation	SCV	SCV	CV	SCV
	Prédiction	0,83	0,72	0,51	0,63
	Rapidité	1,23 s	1,11 s	1,23s	1,13 s
<b>Emissions</b> 	Meilleur modèle	GradientBoosting	GradientBoosting	GradientBoosting	RandomForest
	Pré-optimisation	SCV	CV	SCV	SCV
	Prédiction	0,6	0,54	0,89	0,95
	Rapidité	1,22 s	1,13 s	1,25 s	3,38 s

## **Bilan des meilleures performances (avant optimisation)**

Le bilan des performances montre que le GradientBoosting donne le meilleur compromis précision / rapidité.

Cela dit, sur internet, le XGBoost a la réputation de donner les meilleures performances dans la majorité des cas. Nous allons donc optimiser GradientBoosting, et, au cas où, XGBoost également.



## IV. Optimisations

### Méthodes d'optimisation

#### GridSearchCV

paramètres

valeurs discrètes

```
param_grid = {'n_estimators':[100, 1000],  
              'learning_rate':[0.01, 0.2],  
              'max_depth':[2, 5]}
```

Recherche exhaustive, effectuée avec toutes les combinaisons possibles de paramètres.  
Vorace en temps de calcul, surtout en ajoutant des paramètres (particulièrement *n-estimators*).

#### RandomisedSearchCV

paramètres

valeurs continues

```
distributions = dict(n_estimators=randint(100, 1000),  
                    learning_rate=uniform(loc=0.02, scale=1.98),  
                    max_depth=randint(2, 20),  
                    alpha=uniform(loc=0.1, scale=0.9),  
                    min_samples_leaf=randint(2, 20),  
                    min_samples_split=randint(2, 20))
```

Le nombre de recherches est spécifié, et l'algorithme choisit les combinaisons de paramètres.  
Plus rapide.

# Recherches d'optimisation

## Exemple du XGBoost

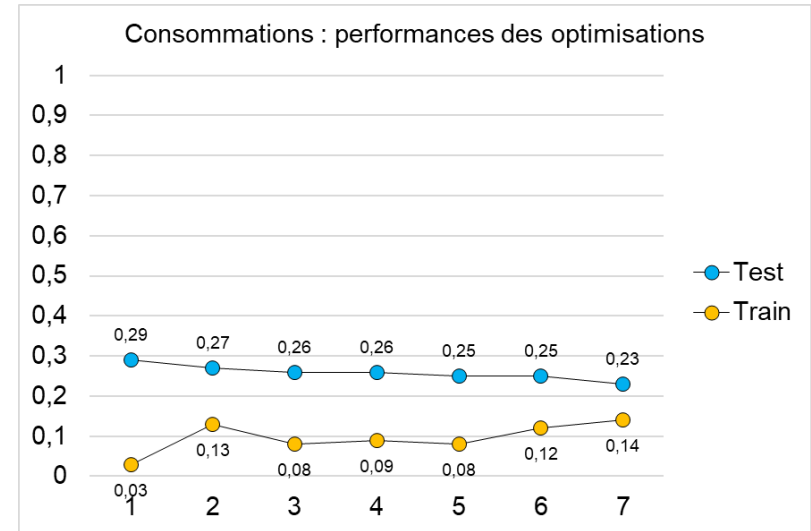
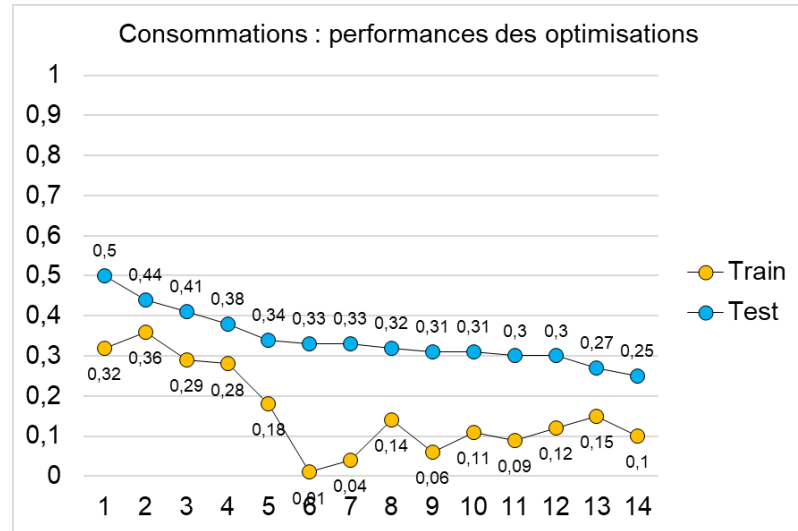
</

# Performances des optimisations (RMSE)

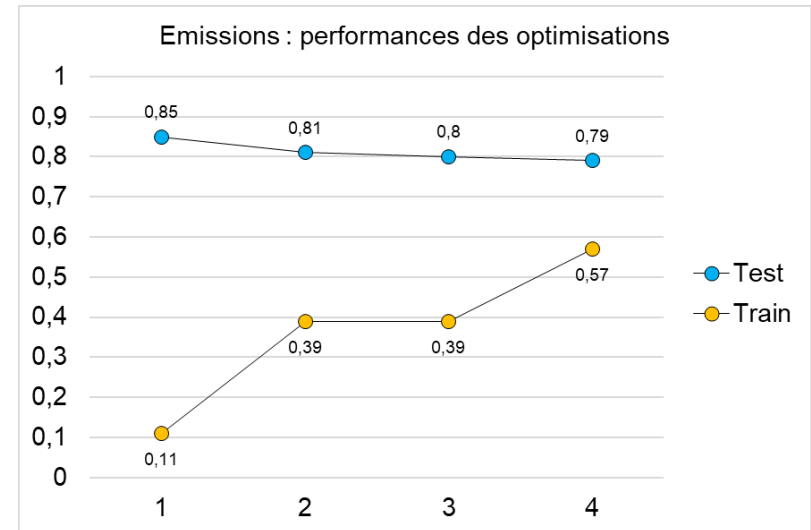
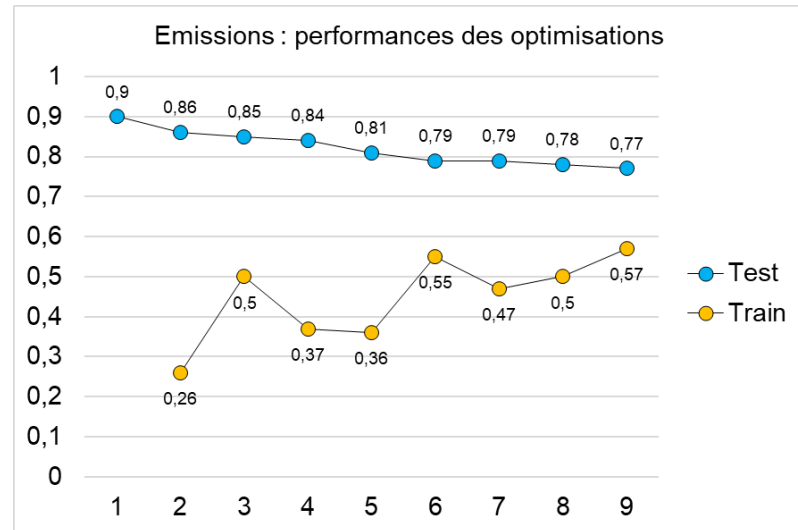
## Gradient Boosting

## XGBoost

Consommations



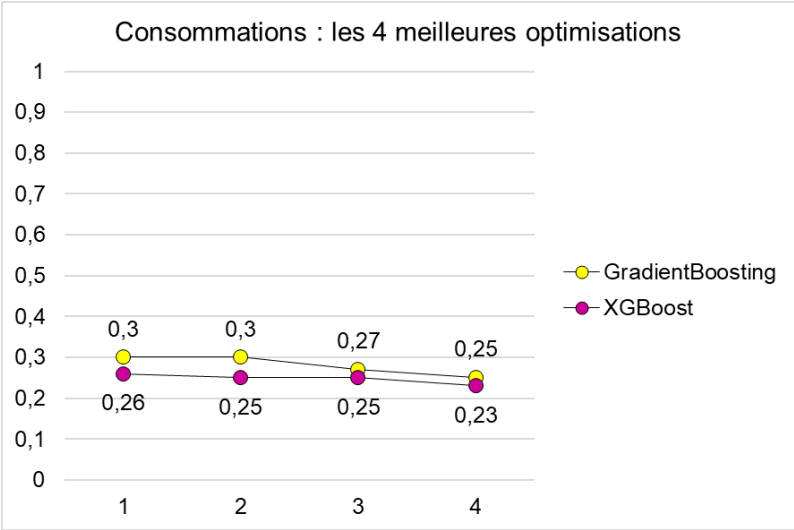
Emissions



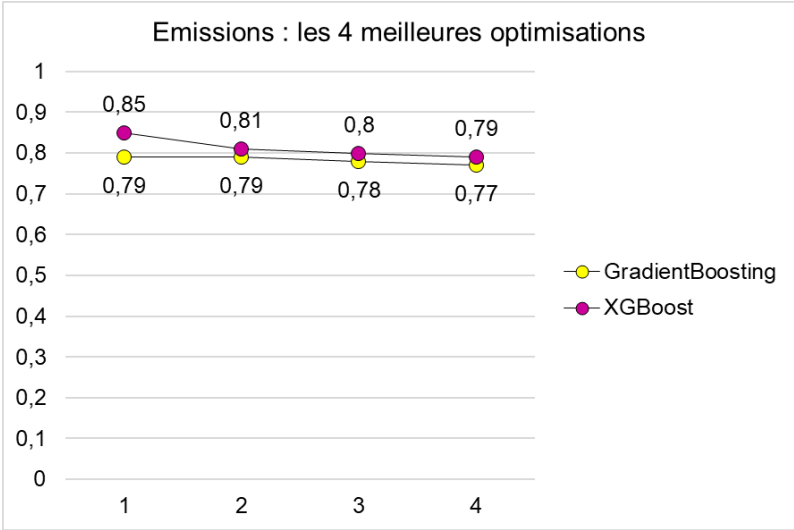
# Performances des optimisations (RMSE)



## Consommations



## Emissions

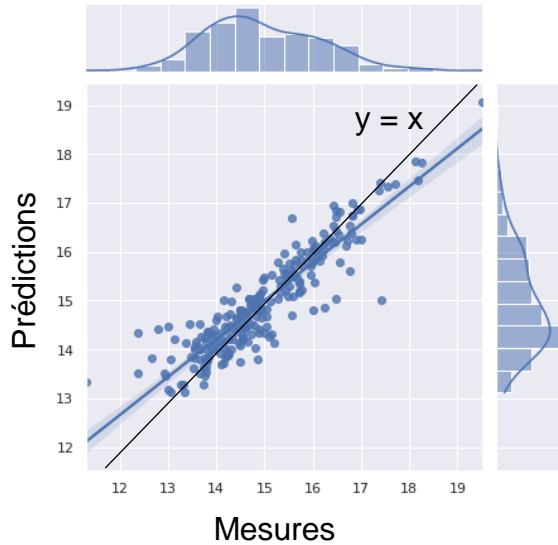


# Bilan sur l'ensemble de test

Consommations

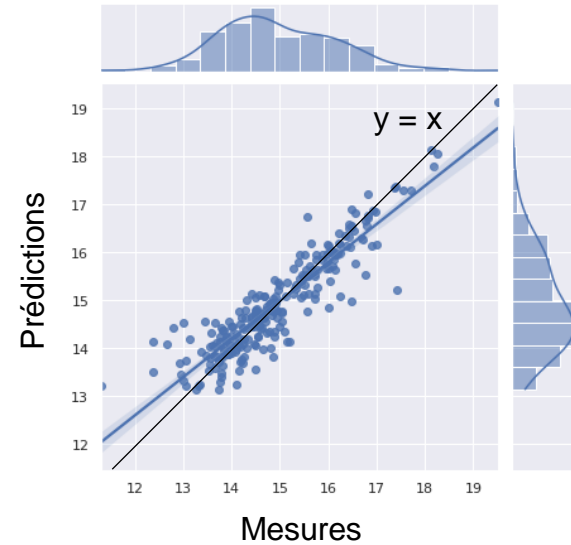


## Gradient Boosting



RMSE : 0,25

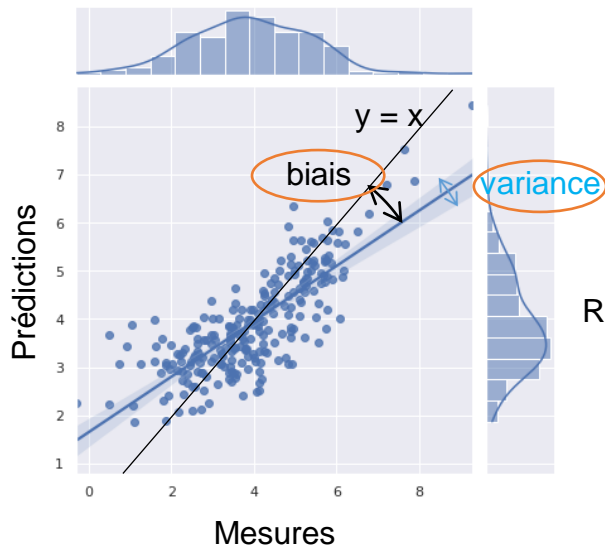
## XGBoost



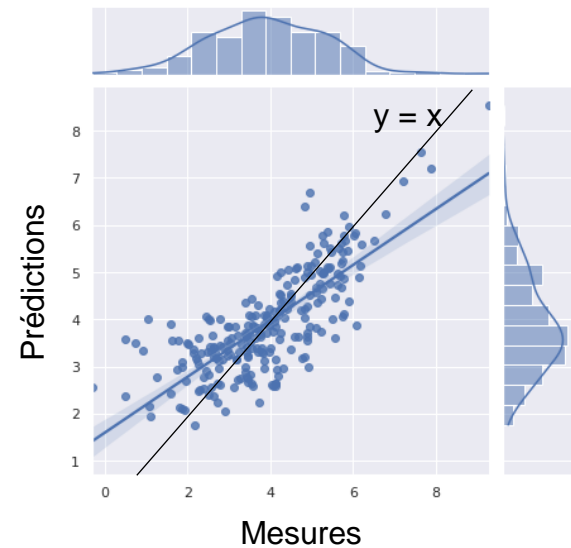
RMSE : 0,23



Emissions

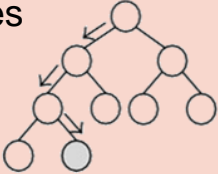



RMSE : 0,77



RMSE : 0,79

# Conclusions

Sujet	Commentaire
Modèle	<p>Le GradientBoosting se démarque largement par rapport aux autres algorithmes.</p> <ul style="list-style-type: none"><li>- sa prédiction est le plus souvent la meilleure,</li><li>- elle est aussi une des plus rapides.</li></ul> 
Métrique	<p>La RMSE semble la métrique la plus adaptée par rapport à R2, car R2 n'est pas utilisée par XGBoost.</p>
Optimisation	<p>Deux facteurs perturbent l'optimisation :</p> <ul style="list-style-type: none"><li>- la répartition aléatoire du <b>split train / set</b>,</li><li>- la nature aléatoire des <b>arbres</b> choisis par GradientBoosting et XGBoost.</li></ul> <p>RandomizedSearchCV semble être la meilleure méthode de recherche.</p>
ENERGYSTARScore	<p>L'indicateur permet à nos algorithmes d'obtenir globalement de meilleures performances.</p> 
Prédictions	<p>Les performances de prédiction sont meilleures pour la <b>consommation</b> que pour les émissions.</p> <p>Pour les deux cas, on a un <b>biais</b> et une certaine <b>variance</b> dans les prédictions.</p>

# Perspectives d'amélioration

Figurer le split train / test

Figurer le caractère aléatoire des SearchCV

Comparer

1. Dummy Regressor basé uniquement sur les features ENERGYSTARScore
2. Les modèles sans ENERGYSTARScore

Différencier temps d'entraînement et temps de prédiction

**Merci pour votre attention**