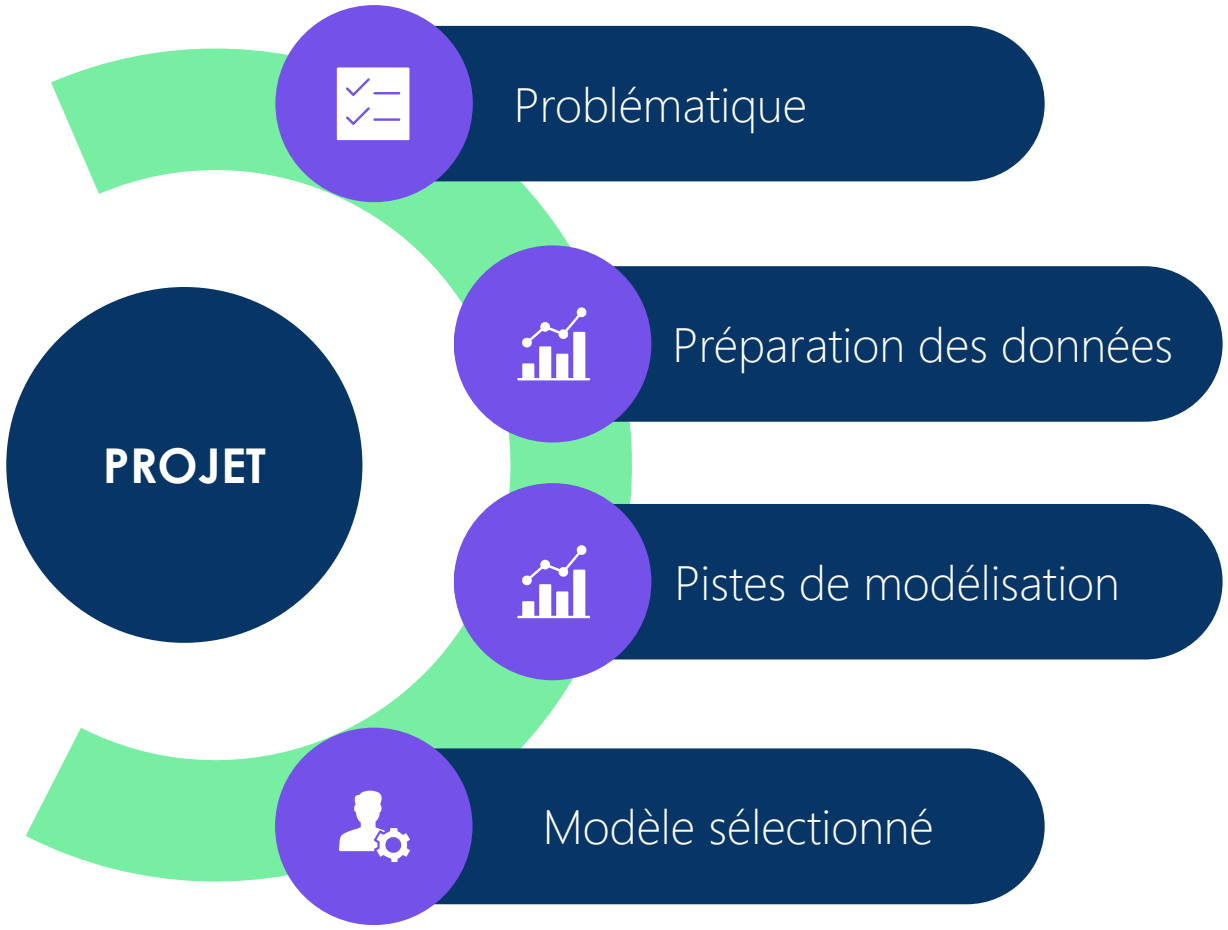
Projet N°4 :
Anticipez les besoins en consommation électrique de bâtiments

Agustin Bunader (autofinancé)
Soutenance de Projet
Février 2020

Programme


PROJET
Problématique
Préparation des données
Pistes de modélisation
Modèle sélectionné

Problématique – Présentation

Problématique :

- Prédire les émissions de CO2 et la consommation totale d'énergie de bâtiments pour lesquels les valeurs n'ont pas encore été mesurées
- Evaluer l'intérêt de l'Energy Star Score pour la prédiction d'émissions ou consommation total d'énergie (fastidieux à calculer)
- Source des données : des relevés minutieux pour les années 2015 et 2016

Seattle

Mission :

- Réaliser une courte analyse exploratoire
- Tester différents modèles de prédiction
- Nouvelles informations ?
- Optimiser les performances en appliquant des transformations aux variables
- Evaluation des performances de la régression
- Optimiser les hyperparamètres et la choix d'algorithme de ML à l'aide d'une validation croisée

ENERGY STAR

Problématique – Interprétation

Prédiction des émissions de CO2 :

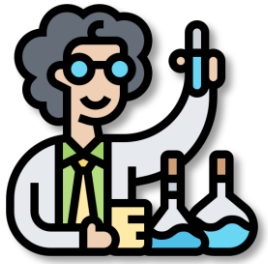
- On interprète que la consigne fait référence à la colonne TotalGHGEmissions
- La colonne GHGEmissionsIntensity est un calcul réalisé sur la colonne TotalGHGEmissions comme décrit dans la source :

  Total Greenhouse Gas Emissions divided by property's gross floor area, measured in kilograms of carbon dioxide equivalent per square foot. This calculation uses a GHG emissions factor from Seattle City Light's portfolio of generating resources

Prédiction de la consommation totale d'énergie :

- On assume qu'il s'agit de la colonne SiteEnergyUse(kBtu) étant donné sa définition dans la source :

  The annual amount of energy consumed by the property from all sources of energy



Problématique – Pistes de recherche

Les jeux de données de 2015 et 2016 seront homogénéisés et concaténés

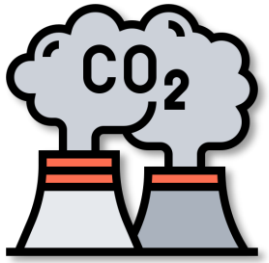
Prévision

Comme il faut prédire deux variables et évaluer l'intérêt de l'Energy Star Score, la modélisation sera réalisée sur quatre jeux de données :

- Deux jeux de données pour prédire la consommation énergétique
- Deux jeux de données pour prédire les émissions de CO2

Pourquoi deux jeux de données pour variable cible ?

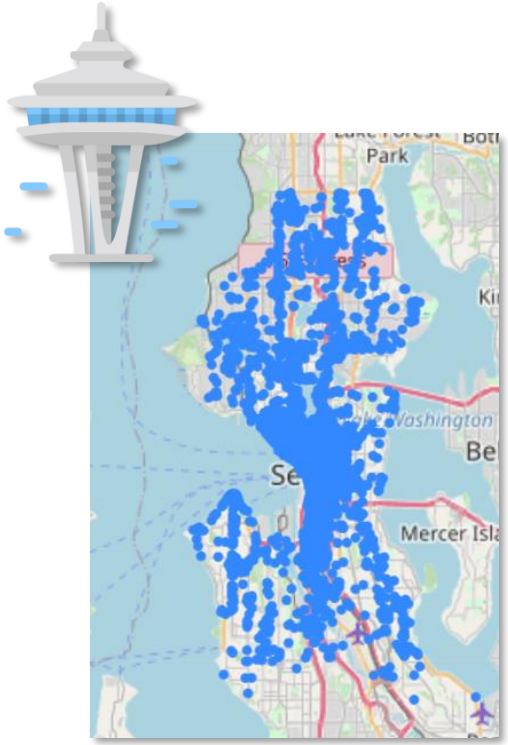
Comme il faut évaluer l'intérêt de l'Energy Star Score, on doit réaliser les mêmes modélisations avec et sans la colonne ENERGYSTARScore



Features ?

Les jeux de données utilisés pour prédire la consommation énergétique et les émissions de CO2 ne contiendront pas forcément les mêmes features

Préparation des données – Nettoyage

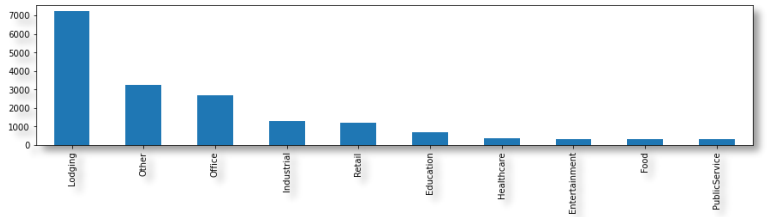


- Différences
  Le format des données n'est pas homogène. L'ordre des colonnes change d'un csv à l'autre, certains noms de colonnes ne sont pas les mêmes et les informations ne sont pas forcément enregistrées avec le même format (notamment les adresses, code postale, latitude et longitude)

- NaN
  La concaténation des jeux de données sources a 6716 lignes. A la fin du nettoyage on garde 4896 lignes pour la modélisation

- Outliers
  - Les coordonnées (*Latitude* et *Longitude*) sont bien placées dans la ville de Seattle
  - *NumberofFloors* : Chinese Baptist Church avec 99 étages, en réalité elle n'a qu'un seul étage. Les bâtiments avec 0 étage sont corrigés selon la moyenne de son *PrimaryPropertyType*
  - *NumberofBuildings* : La valeur 0 et NaN sont remplacés par 1
  - On efface les lignes NaN et de valeur inférieure a 1 dans les colonnes des variables cibles

- PropertyType (Primary, Secondary, Third and Largest property use type)
  On regroupe et réduit les types de propriété en 10 catégories pour aider la modélisation lors de l'utilisation des dummy variables
  Catégories : Lodging, Other, Office, Industrial, Retail, Education, Healthcare, Entertainment, Food, PublicService



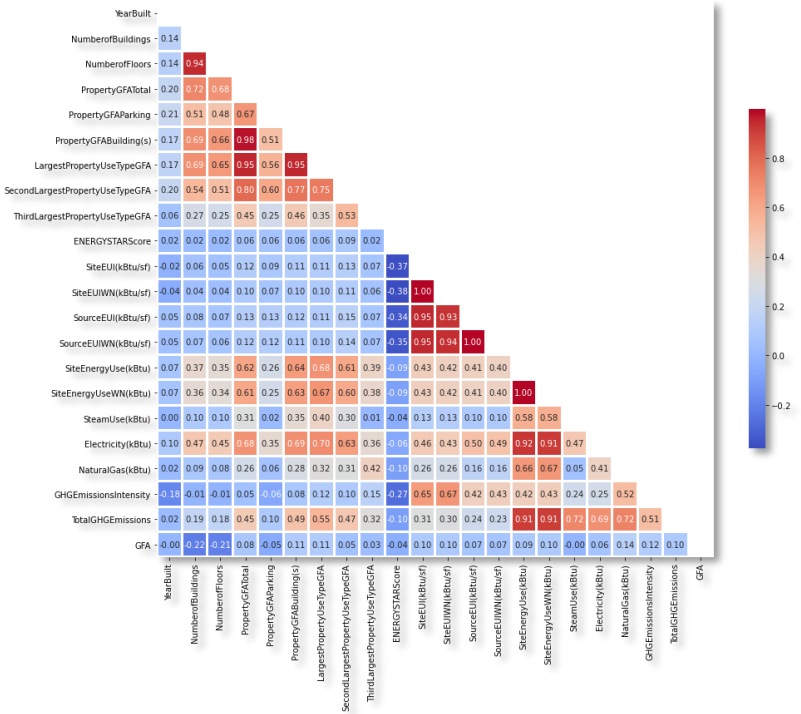
Préparation des données – Features

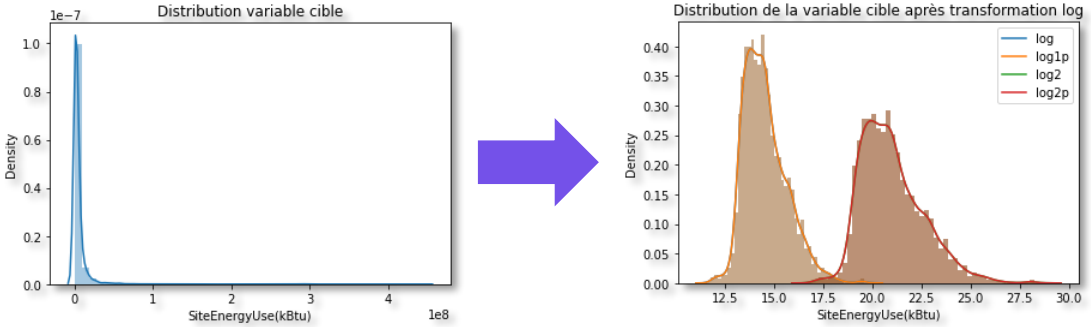
Nouveaux features testés :

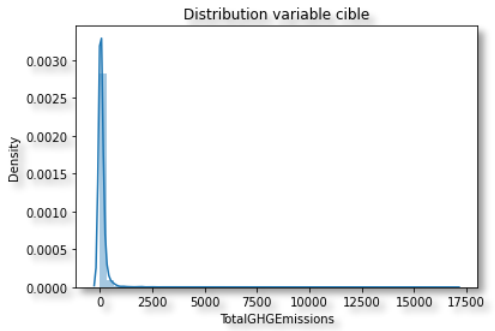
- GFA : Surface par étage calculée à partir de *PropertyGFATotal / NumberofFloors*
- Age : Age du bâtiment calculé à partir de *DataYear – YearBuilt*

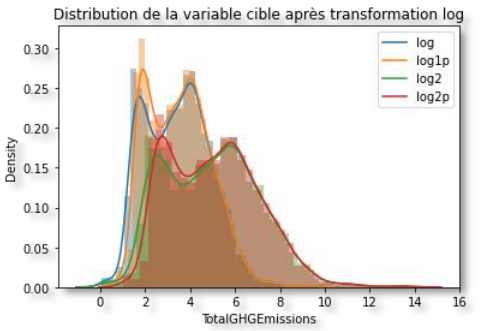
Feature engineering :

- Première étape, retirer les colonnes redondantes ou qui n'ont pas d'utilité pour résoudre la problématique et les colonnes contenant la consommation énergétique par source (elles sont compliquées à mesurer)
- Deuxième étape, créer des variables dummy pour les colonnes contenant des catégories
- Troisième étape, analyser la corrélation entre les variables (matrice)
- Quatrième étape, transformer les variables à prédire
- Dernière étape, sélection finale des features avec sklearn









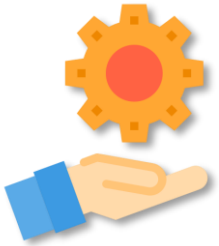
Préparation des données – Features

Utilisant les conversions :

- SiteEnergyUse(kBtu)_log2p = np.log2(1+data['SiteEnergyUse(kBtu)'])
- TotalGHGEmissions_log2 = np.log2(data['TotalGHGEmissions'])

En appliquant les méthodes suivantes sur le jeu de données 2015+2016 :

- Corrélation de Pearson
- Chi2
- Recursive Feature Elimination (RFE)
- Select From Model (SFM) : Lasso
- SFM : Ridge
- SFM : DecisionT ree Classifier
- SFM : Random Forest Classifier



On obtient les features suivants pour les deux variables cibles :

- YearBuilt
- PropertyGFATotal
- PropertyGFABuilding(s)
- LargestPropertyUseTypeGFA
- Electricity(kBtu)
- SecondLargestPropertyUseTypeGFA
- Longitude
- Latitude
- Age
- GFA
- BuildingType_NonResidential
- BuildingType_Multifamily LR (1-4)
- ENERGYSTARScore (uniquement appliqué aux jeux de données identifiés par _ES)

Préparation des données – Features

Méthodes utilisés

- Corrélation de Pearson : Valeur absolue de la corrélation de Pearson entre la cible et les features dans les jeux de données pour énergie et émissions
- Chi2 : Métrique calculée entre la cible et les features, on sélectionne les variables avec les valeurs maximales. Les variables sont forcées à discrètes et doivent être passées par le MixMaxScaler avec un default range de (0,1)

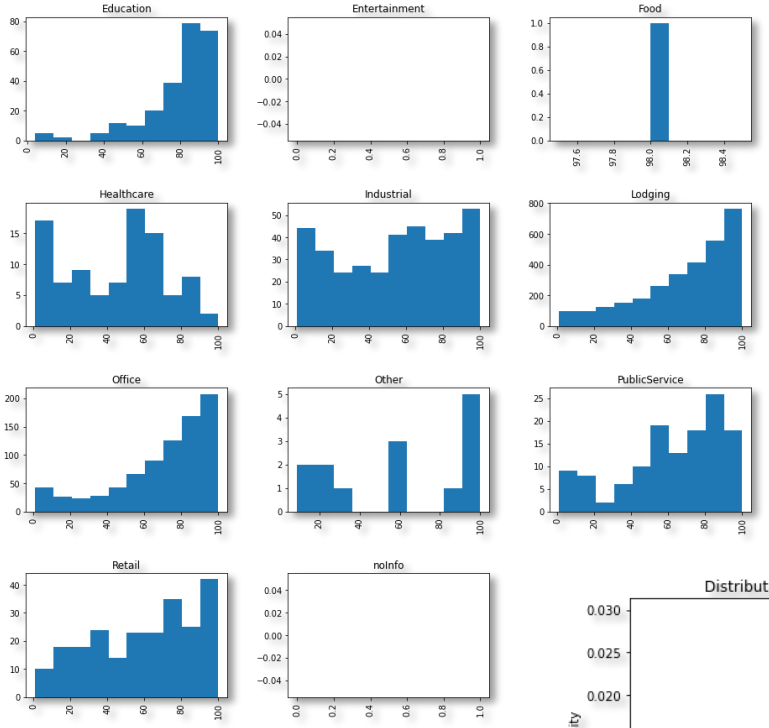
- Recursive Feature Elimination (RFE) : LogisticRegression comme estimateur qui attribue des poids pour sélectionner des features en considérant de manière récursive des ensembles de features de plus en plus petits jusqu'à ce que le nombre souhaité à sélectionner soit finalement atteint
- Select From Model (SFM) – Lasso : Méthode qui utilise L1 comme pénalité pour la sélection de features et réduction de dimension supervisée

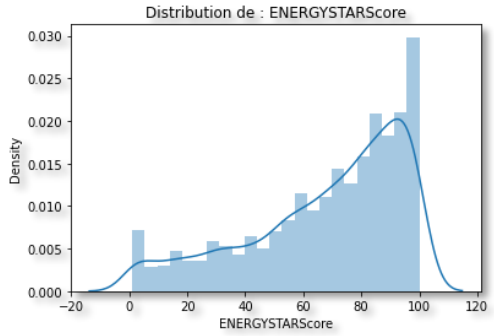
- SFM – Ridge : Utilise L2 comme pénalité pour diminuer la complexité du modèle mais ne réduit pas le nombre de variables car elle ne conduit jamais à un coefficient égal à zéro mais le minimise
- SFM – Decision Tree Classifier : Modèle de Machine Learning simple utilisé dans les problèmes de classification et régression. C'est l'un des modèles d'apprentissage automatique les plus simples
- SFM – Random Forest Classifier : Estimateur qui ajuste un certain nombre de classificateurs d'arbres de décision sur divers sous-échantillons de l'ensemble des données et utilise la moyenne pour améliorer la précision prédictive et contrôler le surapprentissage

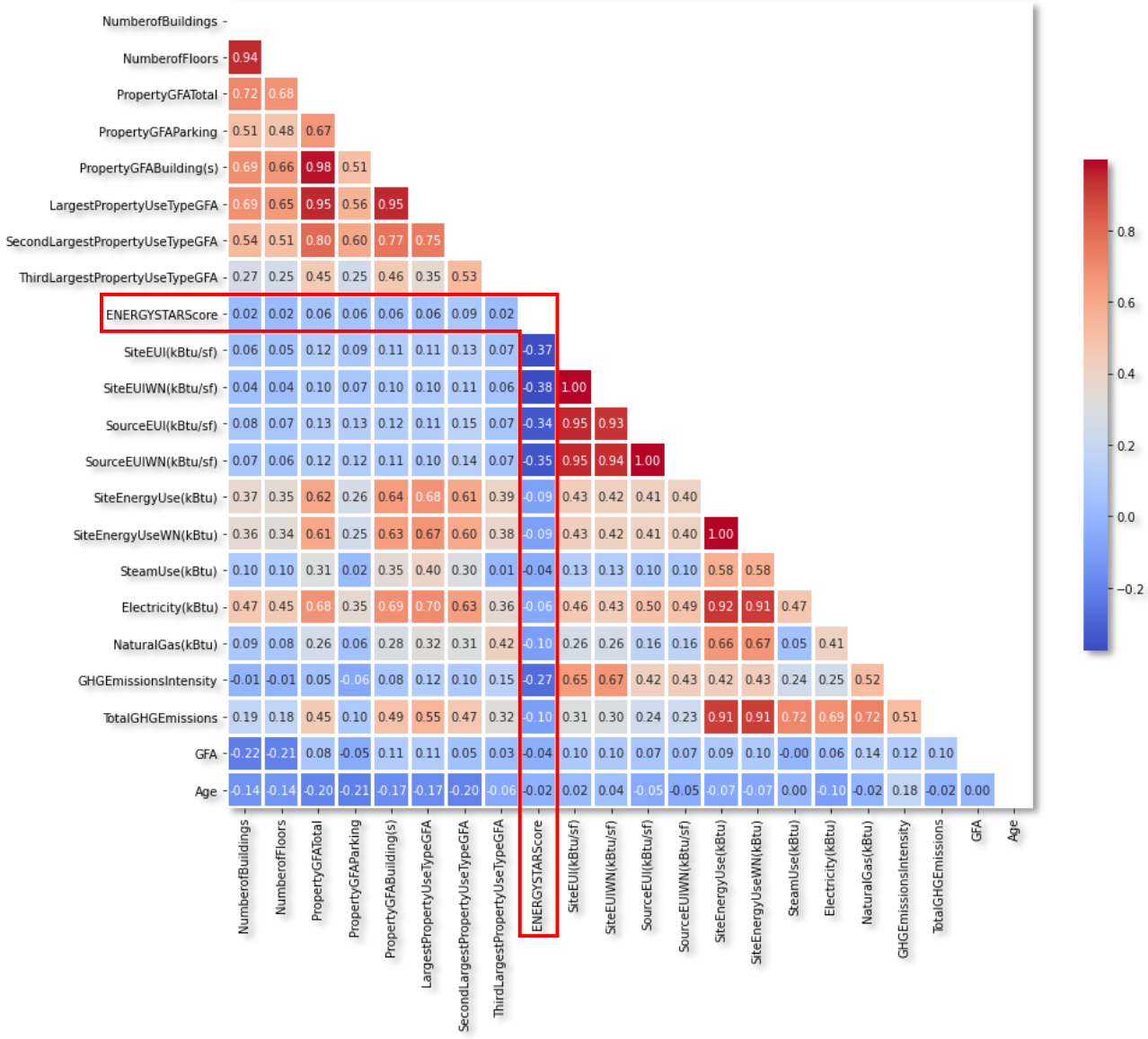
Préparation des données – Exploration

Energy Star Score

- Il n'y a pas de corrélations notables
- Distribution des scores selon *LargestPropertyUseType*

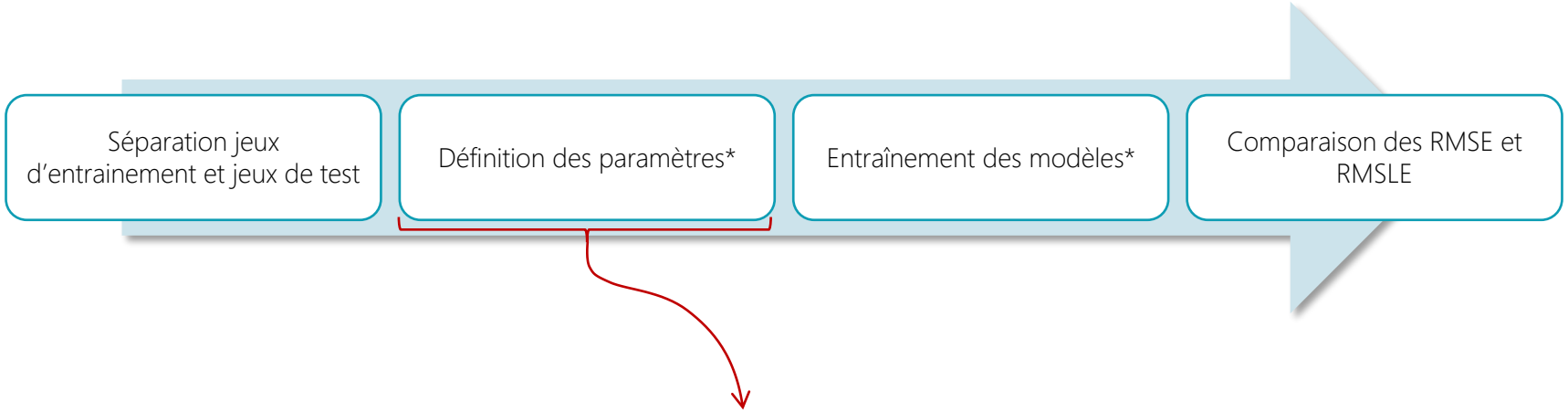






Pistes de modélisation – Démarche

Modèles utilisés : Linear Regression / Elastic Net / Random Forest / SVM



Elastic Net	Random Forest	SVM
tol	n_estimators	C
alpha	min_samples_leaf	epsilon
l1_ratio	max_features	gamma

*Pour chaque estimateur utilisé lors du GridSearchCV

Pistes de modélisation – Jeux de données

On a quatre jeux de données composés de la manière suivante :

- Jeu de données Energy :
  - X_train, X_test, y_train_energy, y_test_energy
- Jeu de données Energy avec Energy Star Score :
  - X_train_ES, X_test_ES, y_train_energy, y_test_energy
- Jeu de données Emissions :
  - X_train, X_test, y_train_emissions, y_test_emissions
- Jeu de données Emissions avec Energy Star Score :
  - X_train_ES, X_test_ES, y_train_emissions, y_test_emissions

In [20]:
```
from sklearn.model_selection import train_test_split

X_train_ES, X_test_ES, y_train, y_test = train_test_split(X_fit, y, test_size=0.25, random_state=42)
```

In [21]:
```
X_train = X_train_ES.drop('ENERGYSTARScore', axis=1)
X_test = X_test_ES.drop('ENERGYSTARScore', axis=1)

y_train_energy = y_train['SiteEnergyUse(kBtu)_log2p']
y_test_energy = y_test['SiteEnergyUse(kBtu)_log2p']
y_train_emissions = y_train['TotalGHGEmissions_log2']
y_test_emissions = y_test['TotalGHGEmissions_log2']
```

Pistes de modélisation – Paramètres

Hyperparamètres à utiliser :

Elastic Net	Random Forest	SVM
tol : [0.1, 0.01, 0.001, 0.0001]	n_estimators : [1, 10, 50, 100, 200, 250, 500, 1000]	C : [0.001, 0.01, 0.1, 1, 10]
alpha : [0.0001, 0.001, 0.01, 0.1, 0.0, 1.0, 10.0, 100.0] Si alpha=0 c'est le même modèle que LinearRegression	min_samples_leaf : [1, 2, 3, 4, 5, 10, 100]	epsilon : [0.0001, 0.001, 0.01, 0.1, 1, 2, 5]
l1_ratio : np.arrange(0, 0.1, 0.01) Si =1 donc L1 penalty (Lasso) Si =0 donc L2 penalty (Ridge)	max_features : ['auto', 'sqrt', 'log2']	gamma : [1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10]

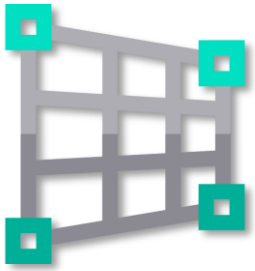
Pistes de modélisation – GridSearchCV

GridSearchCV

Processus de réglage des hyperparamètres afin de déterminer les valeurs optimales pour un modèle donné. La fonction essaie toutes les combinaisons des valeurs passées. GridSearchCV implémente une méthode *fit* et une méthode *score* pour choisir la combinaison avec la meilleure performance.

Principaux paramètres :

- estimator : l'instance de modèle pour laquelle nous souhaitons vérifier les hyperparamètres
- param_grid : l'objet dictionnaire contenant les hyperparamètres à essayer
- scoring : métrique d'évaluation à utiliser
- n_jobs : nombre de processus à exécuter en parallèle pour cette tâche
- cv : nombre des validations croisées à essayer pour chaque ensemble
- verbose : l'impression des détails lors des ajustements de GridSearchCV



Estimateurs utilisés :

- Elastic Net : Régression régularisée qui combine linéairement les pénalités L1 de Lasso et L2 de Ridge pour obtenir une solution moins parcimonieuse que Lasso mais plus stable
- Random Forest : Combiner plusieurs arbres de décision pour déterminer le résultat
- SVM : Plan de décision qui sépare et classe un ensemble de données

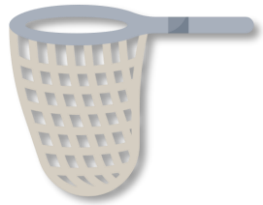
Pistes de modélisation – Estimateurs

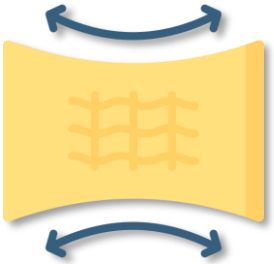
```
clf_energy = linear_model.LinearRegression().fit(X_train, y_train_energy)
```

```
elastic_grid_energy = GridSearchCV(estimator=linear_model.ElasticNet(),
                                   param_grid=parameters,
                                   scoring='neg_mean_squared_error',
                                   cv=5,
                                   verbose=5,
                                   n_jobs=-1,
                                   return_train_score=True)

elastic_grid_energy.fit(X_train, y_train_energy)
```

```
rfr_grid_energy =  GridSearchCV(estimator=RandomForestRegressor(),
                                param_grid=parameters,
                                scoring='neg_mean_squared_error',
                                cv=5,
                                verbose=5,
                                n_jobs=-1,
                                return_train_score=True)

rfr_grid_energy.fit(X_train, y_train_energy)
```

```
#default kernel='rbf' (radial basis function)
svm_energy = GridSearchCV(estimator=SVR(),
                          param_grid=parameters,
                          scoring='neg_mean_squared_error',
                          cv=5,
                          verbose=5,
                          n_jobs=-1,
                          return_train_score=True)

svm_energy.fit(X_train, y_train_energy)
```





Pistes de modélisation – RMSE vs RMSLE

RMSE et RMSLE comme mesures des différences entre les valeurs prédites par un modèle ou un estimateur et les valeurs observées

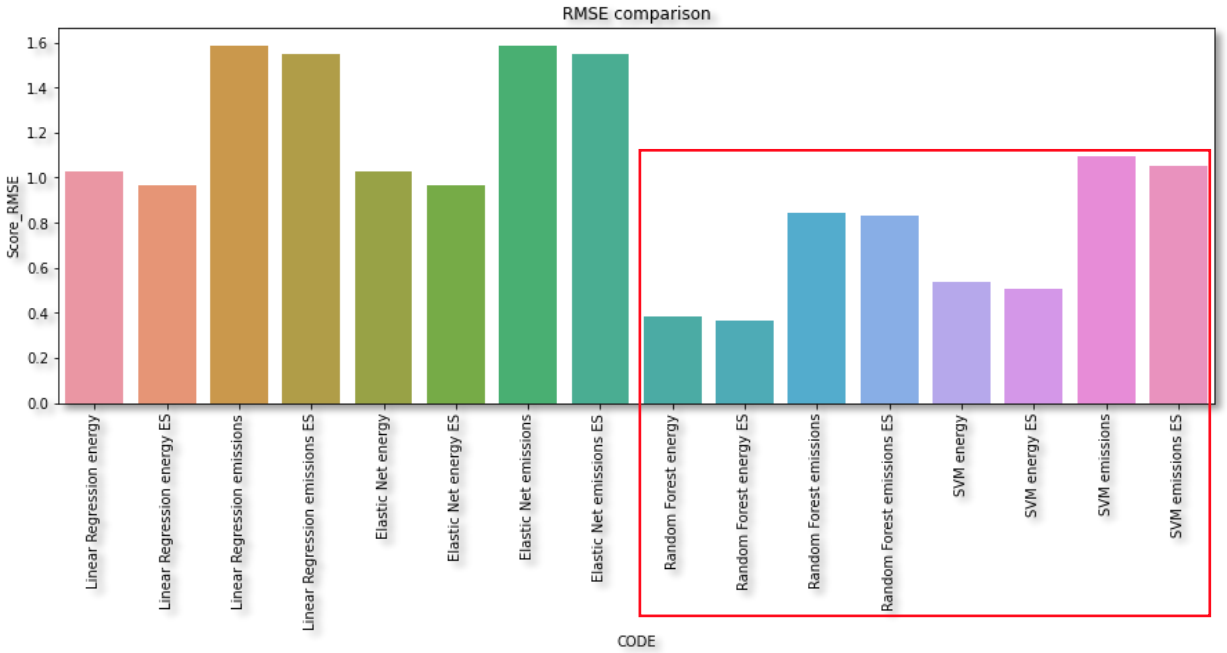
Root Mean Squared Error (**RMSE**) : est une mesure de la précision pour comparer les erreurs de prévision de différents modèles pour un ensemble de données particulier et non entre des ensembles de données car elle dépend de l'échelle (RMSE est sensible aux valeurs aberrantes)

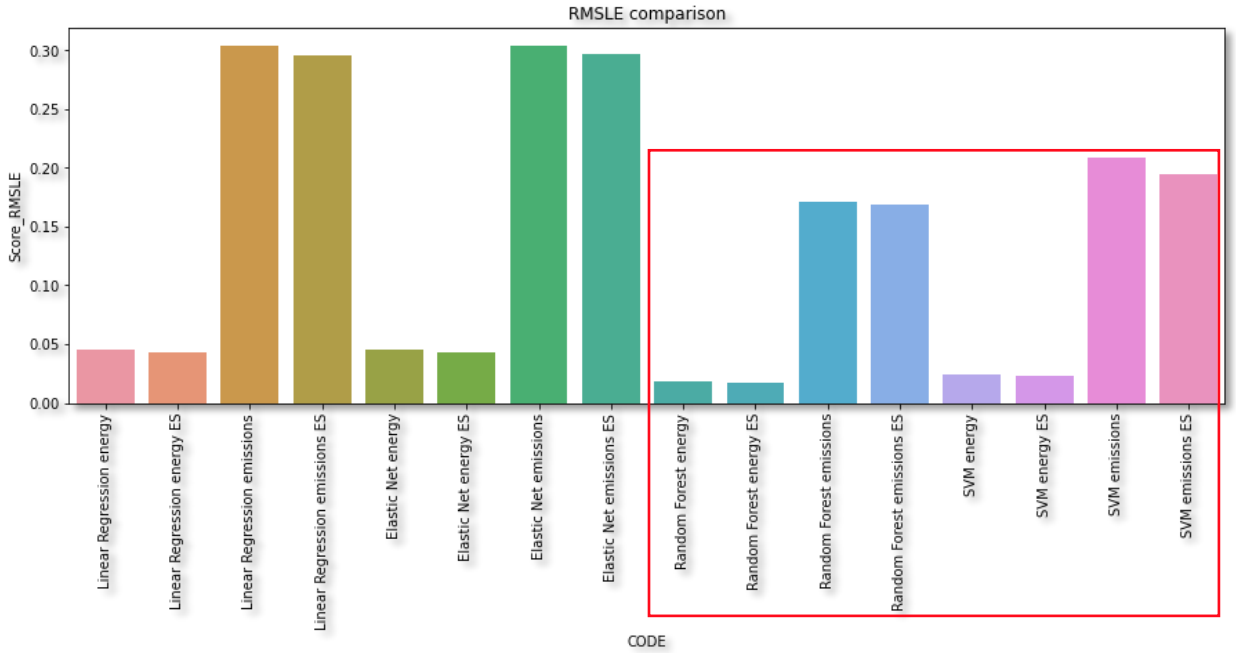
Root Mean Squared Log Error (**RMSLE**) : est une métrique d'évaluation qu'utilise le logarithme de les valeurs prédites et les valeurs observées. C'est cette petite différence avec le RMSE qui donne à le RMSLE ses propriétés uniques :

- Robustesse à l'effet des valeurs aberrantes
- Erreur relative
- Pénalité biaisée

Pistes de modélisation – Résultats

Résultats de chaque modèle :





- Random Forest et SVM ont les RMSE et RMSLE les plus faibles dans les modèles analysés
- Les résultats obtenus avec *ENERGYSTARScore* sont similaires aux résultats obtenus sans lui pourtant il n'y a pas d'avantage tangible à l'utilisation du score lors des prédictions

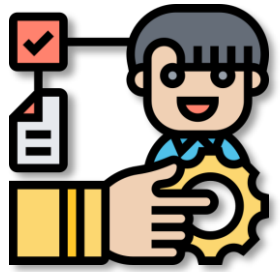
Pistes de modélisation – Résultats

Model	Predict	ENERGYSTAR	Score_R2	Score_RMSE	Score_RMSLE	CODE
Linear Regression	energy	no	0.609291	1.026396	0.045543	Linear Regression energy
Linear Regression	energy	yes	0.652965	0.967331	0.042744	Linear Regression energy ES
Linear Regression	emissions	no	0.441369	1.587136	0.303684	Linear Regression emissions
Linear Regression	emissions	yes	0.469671	1.546409	0.295606	Linear Regression emissions ES
Elastic Net	energy	no	0.608555	1.027362	0.045640	Elastic Net energy
Elastic Net	energy	yes	0.652364	0.968168	0.042870	Elastic Net energy ES
Elastic Net	emissions	no	0.441933	1.586335	0.304266	Elastic Net emissions
Elastic Net	emissions	yes	0.470273	1.545531	0.296327	Elastic Net emissions ES
Random Forest	energy	no	0.945354	0.383857	0.017702	Random Forest energy
Random Forest	energy	yes	0.950563	0.365101	0.016755	Random Forest energy ES
Random Forest	emissions	no	0.842939	0.841561	0.171091	Random Forest emissions
Random Forest	emissions	yes	0.846660	0.831532	0.168733	Random Forest emissions ES
SVM	energy	no	0.892745	0.537769	0.024694	SVM energy
SVM	energy	yes	0.905445	0.504930	0.023209	SVM energy ES
SVM	emissions	no	0.735513	1.092079	0.208179	SVM emissions
SVM	emissions	yes	0.755140	1.050777	0.194805	SVM emissions ES

Modèle sélectionné – Conclussions

Comme déjà signalé, les RMSE et RMSLE des modèles Random Forest et SVM sont les plus faibles et la différence de résultat entre les modèles avec et sans le feature *ENERGYSTARScore* sont négligeables. De plus, les Score R2 suivent un comportement similaire aux mesures d'erreur.

Donc, il est donc possible d'affirmer qu'il n'y a pas d'intérêt à utiliser l'Energy Star Score pour la prédiction d'émissions ou la consommation totale d'énergie



Modèle sélectionné – Conclussions

Finalement, pour prédire les émissions et la consommation totale d'énergie on peut garder les deux modèles suivants de Random Forest

```
GridSearchCV(cv=5, estimator=RandomForestRegressor(), n_jobs=-1,
             param_grid={'max_features': ['auto', 'sqrt', 'log2'],
                         'min_samples_leaf': [1, 2, 3, 4, 5, 10, 100],
                         'n_estimators': [1, 10, 50, 100, 200, 250, 500, 1000]},
             return_train_score=True, scoring='neg_mean_squared_error',
             verbose=5)
```

```
print(rfr_grid_energy.best_params_)
print('Mean absolute error (MAE):', rfr_grid_energy.best_score_)
```

```
{'max_features': 'log2', 'min_samples_leaf': 1, 'n_estimators': 1000}
Mean absolute error (MAE): -0.1576942928461
```

```
GridSearchCV(cv=5, estimator=RandomForestRegressor(), n_jobs=-1,
             param_grid={'max_features': ['auto', 'sqrt', 'log2'],
                         'min_samples_leaf': [1, 2, 3, 4, 5, 10, 100],
                         'n_estimators': [1, 10, 50, 100, 200, 250, 500, 1000]},
             return_train_score=True, scoring='neg_mean_squared_error',
             verbose=5)
```

```
print(rfr_grid_emissions.best_params_)
print('Mean absolute error (MAE):', rfr_grid_emissions.best_score_)
```

```
{'max_features': 'sqrt', 'min_samples_leaf': 1, 'n_estimators': 1000}
Mean absolute error (MAE): -0.8711422494980543
```

A prédire

```
y['SiteEnergyUse(kBtu)_log2p'] = np.log2(1+y['SiteEnergyUse(kBtu)'])
y['TotalGHGEmissions_log2'] = np.log2(y['TotalGHGEmissions'])
```

Features

```
['YearBuilt',
 'PropertyGFATotal',
 'PropertyGFABuilding(s)',
 'LargestPropertyUseTypeGFA',
 'Electricity(kBtu)',
 'SecondLargestPropertyUseTypeGFA',
 'Longitude',
 'Latitude',
 'Age',
 'GFA',
 'BuildingType_Multifamily LR (1-4)',
 'BuildingType_NonResidential']
```