Ensemble Learning Python Project

27/03/2023

Chenxi HE
ESSEC Business School and Centrale Supélec
b00803893@essec.edu

Shiyun WANG
ESSEC Business School and Centrale Supélec
b00802405@essec.edu

Yunjing JIANG
ESSEC Business School and Centrale Supélec
b00794792@essec.edu

Yunqiu ZHANG
ESSEC Business School and Centrale Supélec
b00794126@essec.edu

Table of Contents

Part 1

The Airbnb python project

- Motivations
- Dataset
- Data Preprocessing
- Models and tuning
- Conclude

Part 2

The Decision tree project

- Pseudo code
- GitHub
- Dataset
- Application
- Conclude

ESSEC BUSINESS SCHOOL CentraleSupélec

Part 1

New York City Airbnb house price prediction

Reference data source:
https://www.kaggle.com/datasets/dgomonov/new-york-city-airbnb-open-data

ESSEC BUSINESS SCHOOL

CentraleSupélec

Introduction and Motivations

●**Situation**: The Airbnb price market is dynamic and subject to many factors: changes in supply and demand conditions, different locations, and the type of the rooms.

●**Motivation:** In order to make the renters to get an accurate sense of fair pricing and the customers have more foreseeability to book the house ahead.

●**Objective**: Using different Ensemble Learning models to build robust prediction models for the price of the Airbnb in New York.

Data preprocessing

- Change the last_review format to DateTime
- Impute missing values

  last_review - maximum value; reviews_per_month - 0; name, host_name - empty string

- Extract another two features

  'no_review': show if the house has no review;

  ‘count_name': indicate the number of words in ‘name’

- Manage outliers

  'minimum_nights' - the 99th percentile

  'price' - the 1st and 99th percentiles

- Apply Log-transformation of the target variable
- One-hot encoding for categorical features and MinMax scaling for numerical features.

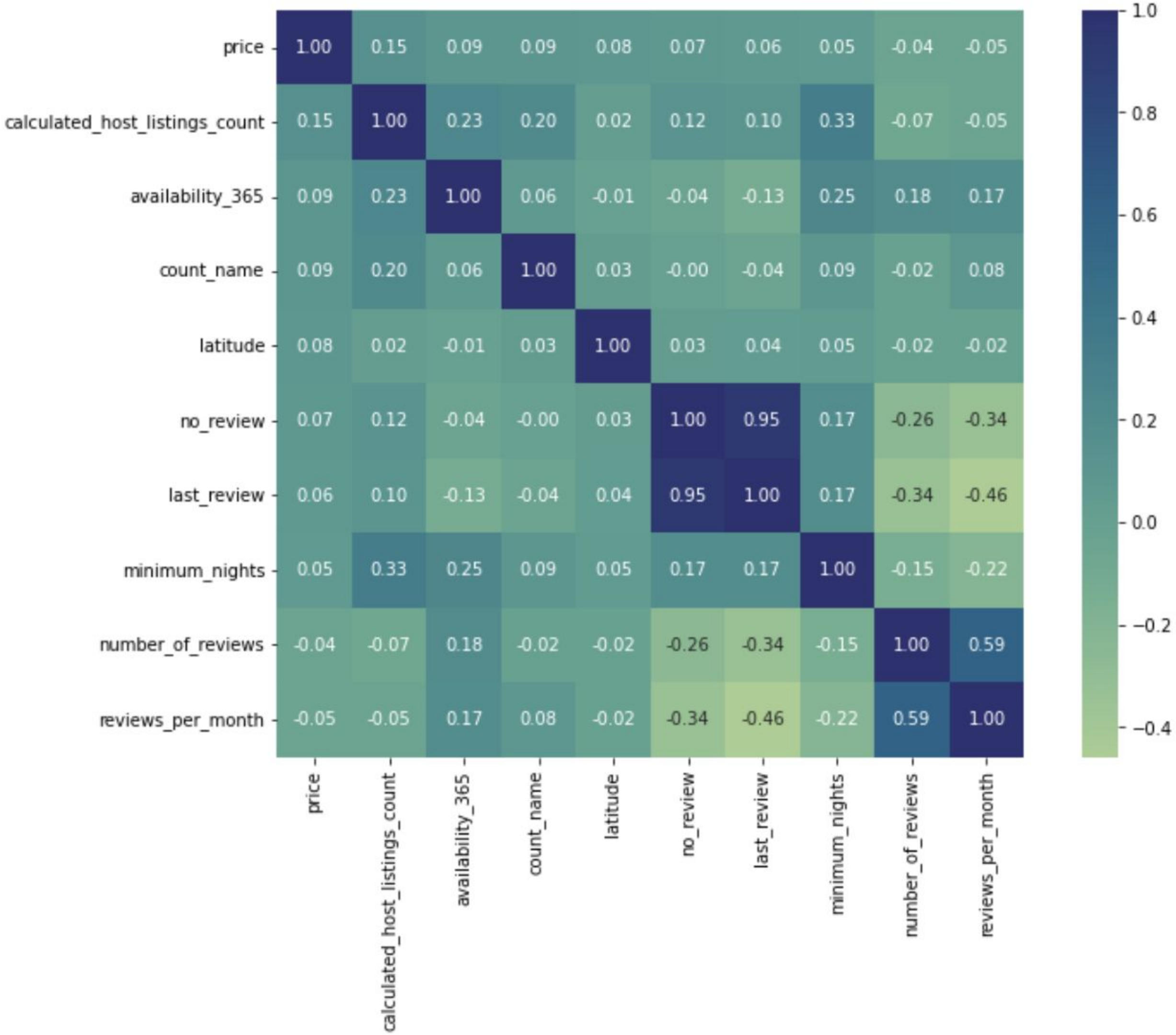
ESSEC BUSINESS SCHOOL CentraleSupélec

Exploration the dataset

Feature selection

For Numerical Features

- Create the Correlation Heatmap
- Choose factors that have the highest correlation with prices (since there are only 10 numerical features in total,so we choose all)

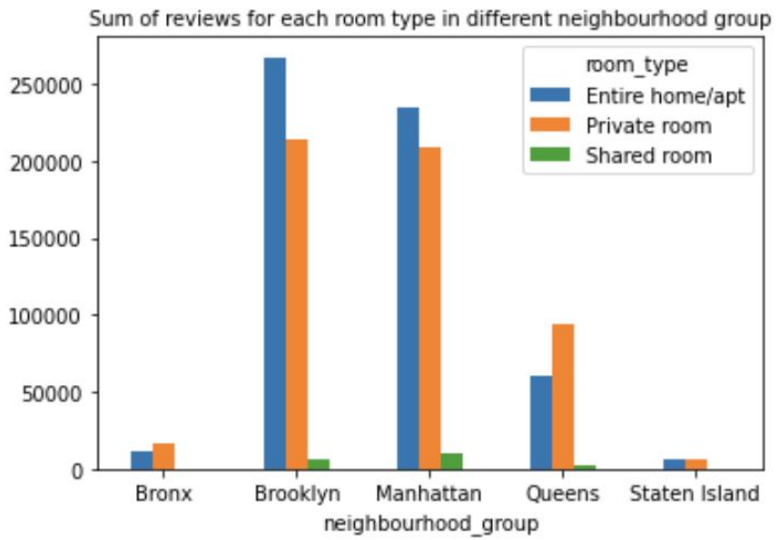


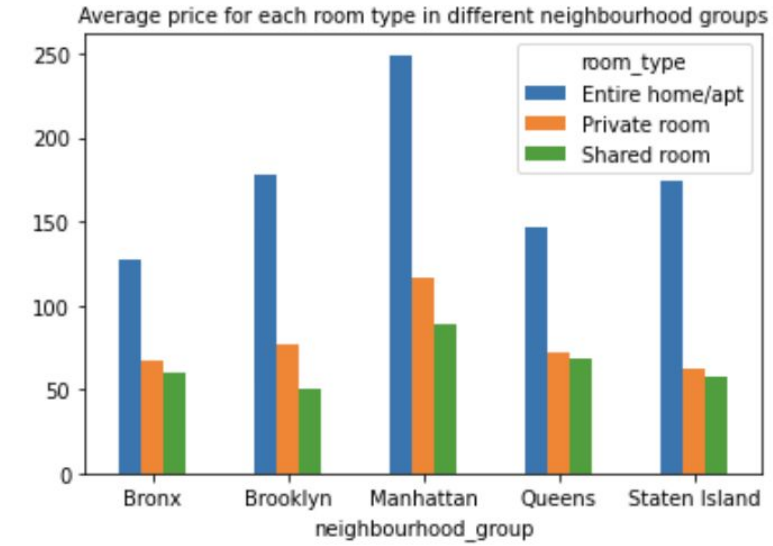
Exploration the dataset

Feature selection

For Categorical Features

- Check the features one by one; remove all the columns just for naming items
- Remove columns which have distributions with too few unique values.
- Final we choose three as the categorical features: 'neighbourhood_group', 'neighbourhood', 'room_type'





ESSEC BUSINESS SCHOOL
CentraleSupélec

Models and tuning strategy

Model	Mean R-squared score
XG Boost	**0.639**
Gradient Boosting	**0.631**
Random Forest	**0.642**
AdaBoost	**0.531**
Bagging	**0.629**

Hyperparameters tuning: GridSearchCV.

for random forest, param_grid= { 'n_estimators': [100, 500, 1000],
'max_features': [16, 32, 64] }

Performance evaluation: cross-validation with five folds and the mean R2 score and standard deviation of the scores

ESSEC BUSINESS SCHOOL CentraleSupélec

Conclusion

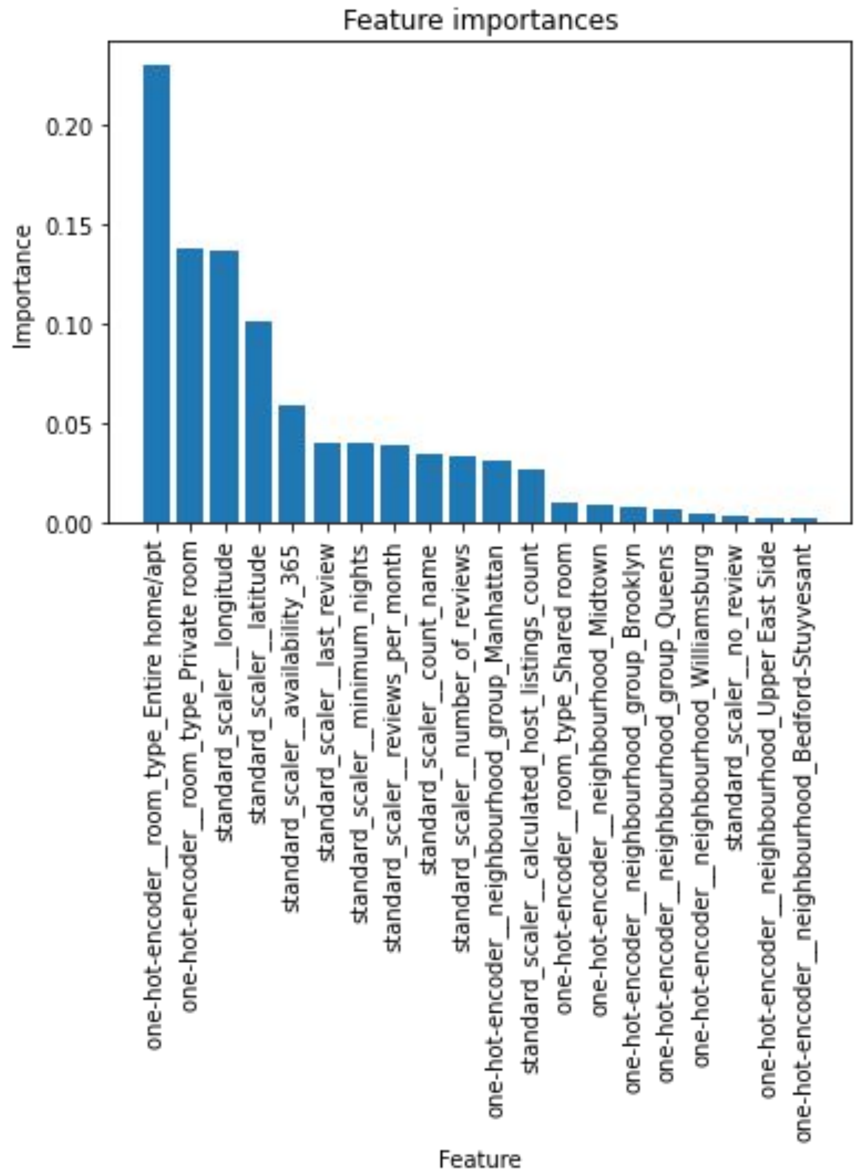
1. Random forest outperform other ensemble models with R2 value of 0.642

2. Feature importance:

   room type_entire home/apt is the most important one and followed by room type_private room and longitude.

3. Further improvement:

   create new features from the existing ones



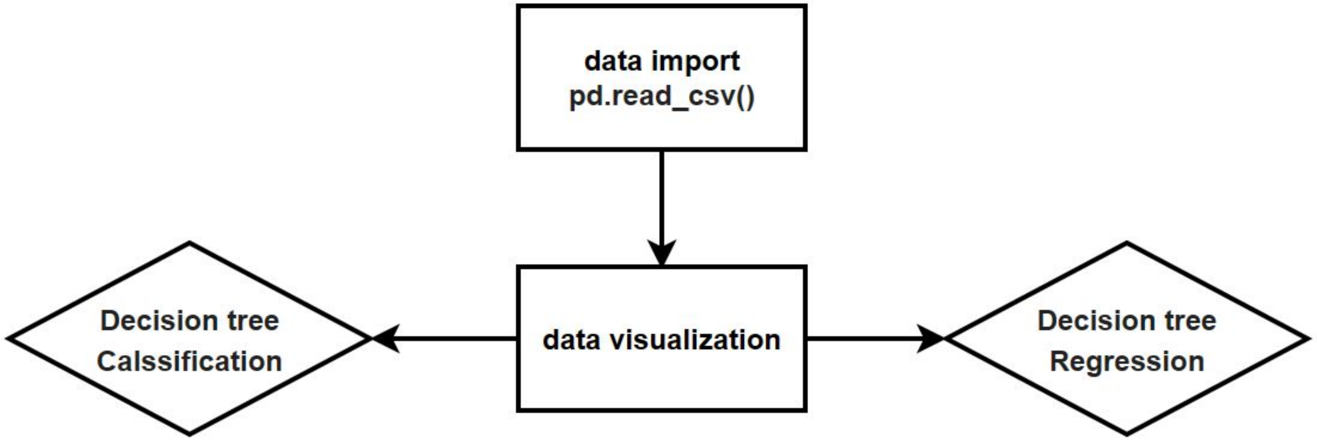
ESSEC BUSINESS SCHOOL CentraleSupélec

Part 2

Paris housing classification and regression

Reference data source:
https://www.kaggle.com/datasets/mssmartypants/paris-housing-classification

Pseudo Code

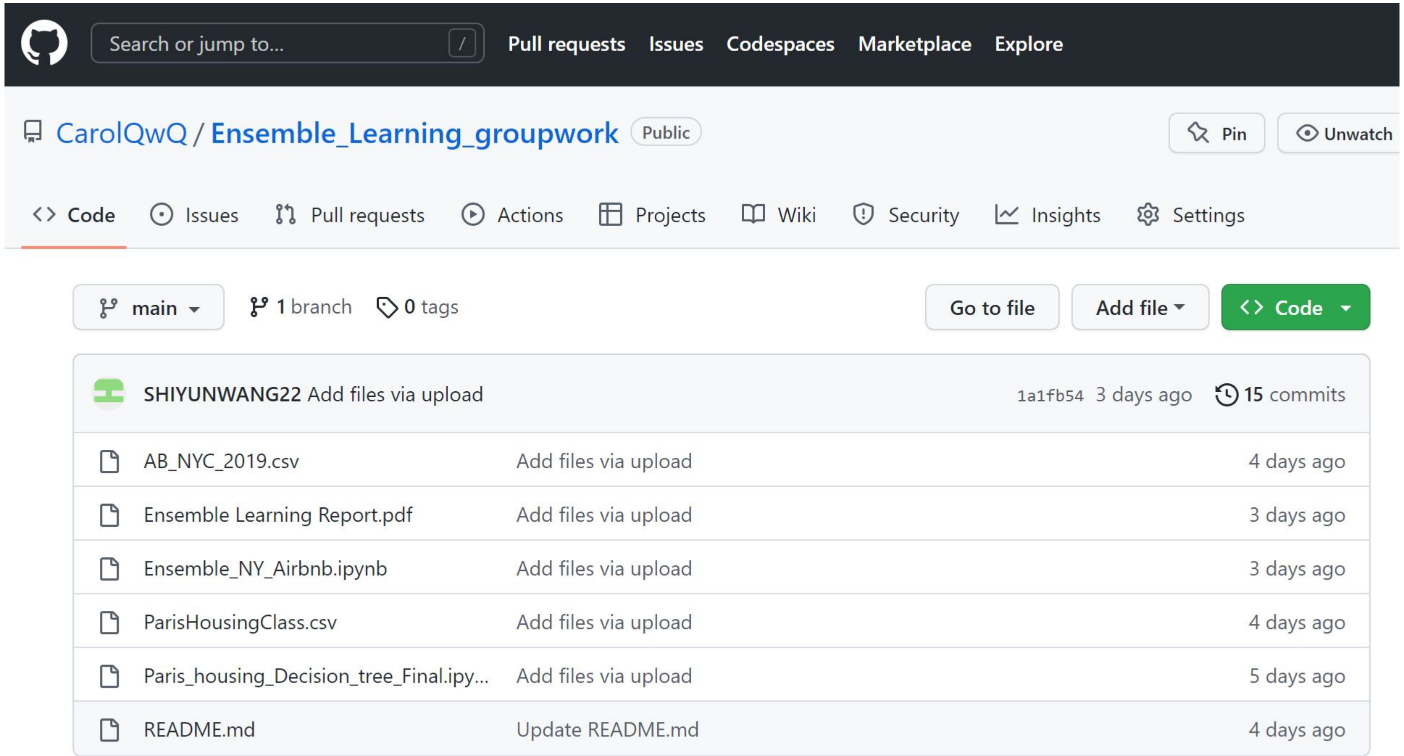



```
# Split data
X = df1.drop('category', axis = 1)
y = df1['category']
X_train, X_test, y_train, y_test=
train_test_split(X, y, test_size=0.3,
random_state=101)

# data standardize
scaler_train= StandardScaler()

# train classification decision tree model
tree_model = DecisionTreeClassifier()
tree_model =
tree_model.fit(X_train,y_train)
y_pred = tree_model.predict(X_test)

# Hyperparameter tuning
GridSearchCV(DecisionTreeClassifier(),
param_grid=pgrid, cv=10)
```

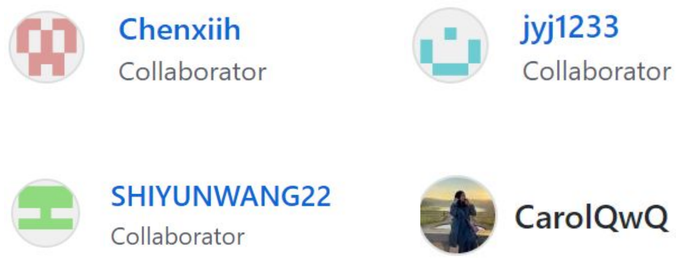
```
# Split data
X = df2.drop('price', axis = 1)
y = df2['price']
X_train, X_test, y_train, y_test=
train_test_split(X, y, test_size=0.3,
random_state=101)

# data standardize
scaler_train= StandardScaler()

# train classification decision tree model
tree_model = DecisionTreeRegressor()
tree_model =
tree_model.fit(X_train,y_train)
y_pred = tree_model.predict(X_test)

# Hyperparameter tuning
GridSearchCV(DecisionTreeRegressor(
), param_grid=pgrid, cv=10)
```

ESSEC BUSINESS SCHOOL CentraleSupélec

Github



Collaborators:

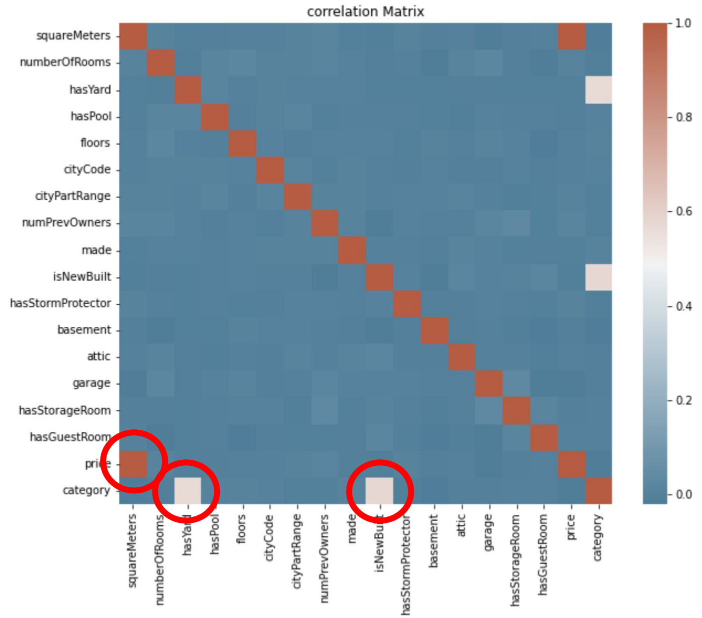


Github link: https://github.com/CarolQwQ/Ensemble_Learning_groupwork

Dataset

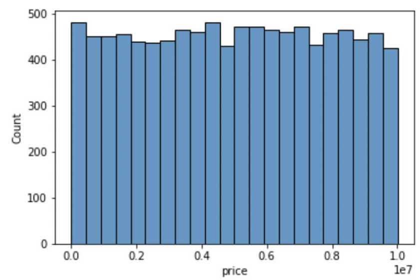
- Imaginary data about house prices in Paris,
- With 18 columns containing SquareMeters, NumberOfRooms, hasYard, etc
- All the features are numeric variables except the “category”.

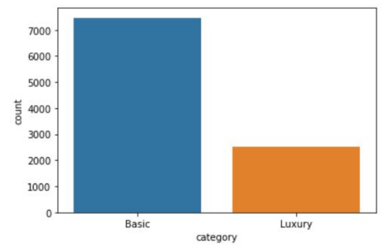
```
Data columns (total 18 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   squareMeters       10000 non-null  int64
 1   numberOfRooms      10000 non-null  int64
 2   hasYard            10000 non-null  int64
 3   hasPool            10000 non-null  int64
 4   floors             10000 non-null  int64
 5   cityCode           10000 non-null  int64
 6   cityPartRange      10000 non-null  int64
 7   numPrevOwners      10000 non-null  int64
 8   made               10000 non-null  int64
 9   isNewBuilt         10000 non-null  int64
 10  hasStormProtector  10000 non-null  int64
 11  basement           10000 non-null  int64
 12  attic              10000 non-null  int64
 13  garage             10000 non-null  int64
 14  hasStorageRoom     10000 non-null  int64
 15  hasGuestRoom       10000 non-null  int64
 16  price              10000 non-null  float64
 17  category           10000 non-null  object
```



The correlation between the price and squareMeters is super high and almost 1, and the correlation of category with hasYard and isNewBuilt is obvious higher then with other features

The distribution of two predicted variables

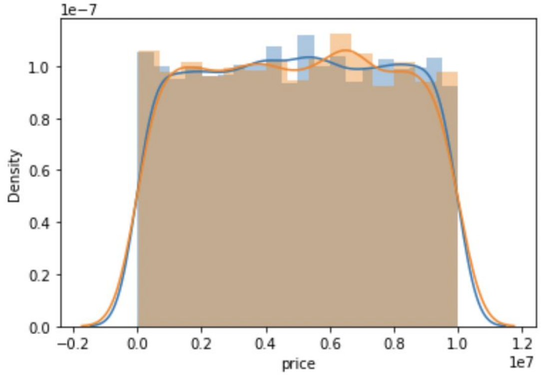




Application

1. Split 30% of the dataset into testset and the remaining is the train set.

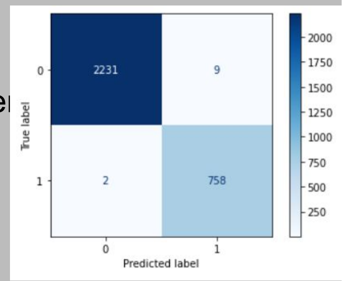
The density of the Y variable of both datasets.



2. Scaling on the training dataset, to ensure each input is on a similar scale

Decision Tree Classification

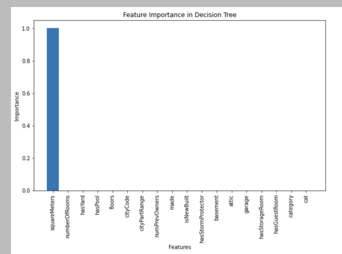
- with the default configuration of the mode
  <u>0.996</u> accuracy score and **<u>0.003</u>** mean square er



- "max_depth" ranging from 1 to 10 and
  "min_samples_split" ranging from 2 to 20
  <u>Result remain the same</u>

- feature_importance: the most important feature is "**<u>hasYard"</u>**

Decision Tree Regression



- with the default configuration of the mode
  R2 score: **<u>0.9999960258049935</u>**

- feature_importance: the only important feature is **<u>"squareMeters"</u>**

ESSEC BUSINESS SCHOOL

CentraleSupélec

Conclusion

- Decision can be **a powerful tool** for predicting house prices and categories.
- It is crucial to **gather as much relevant data as possible** and **fine-tune the model** to achieve the best results. For example, the max_depth and min_samples_split in the decision tree classification model.
- Feature Importance score can be used as **feature selection. explaining the model, and improving the model**

ESSEC BUSINESS SCHOOL CentraleSupélec