House Price Prediction

Plan

01

BUSINESS UNDERSTANDING

02

DATA UNDERSTANDING

03

DATA PREPROCESSING

04

MODELING

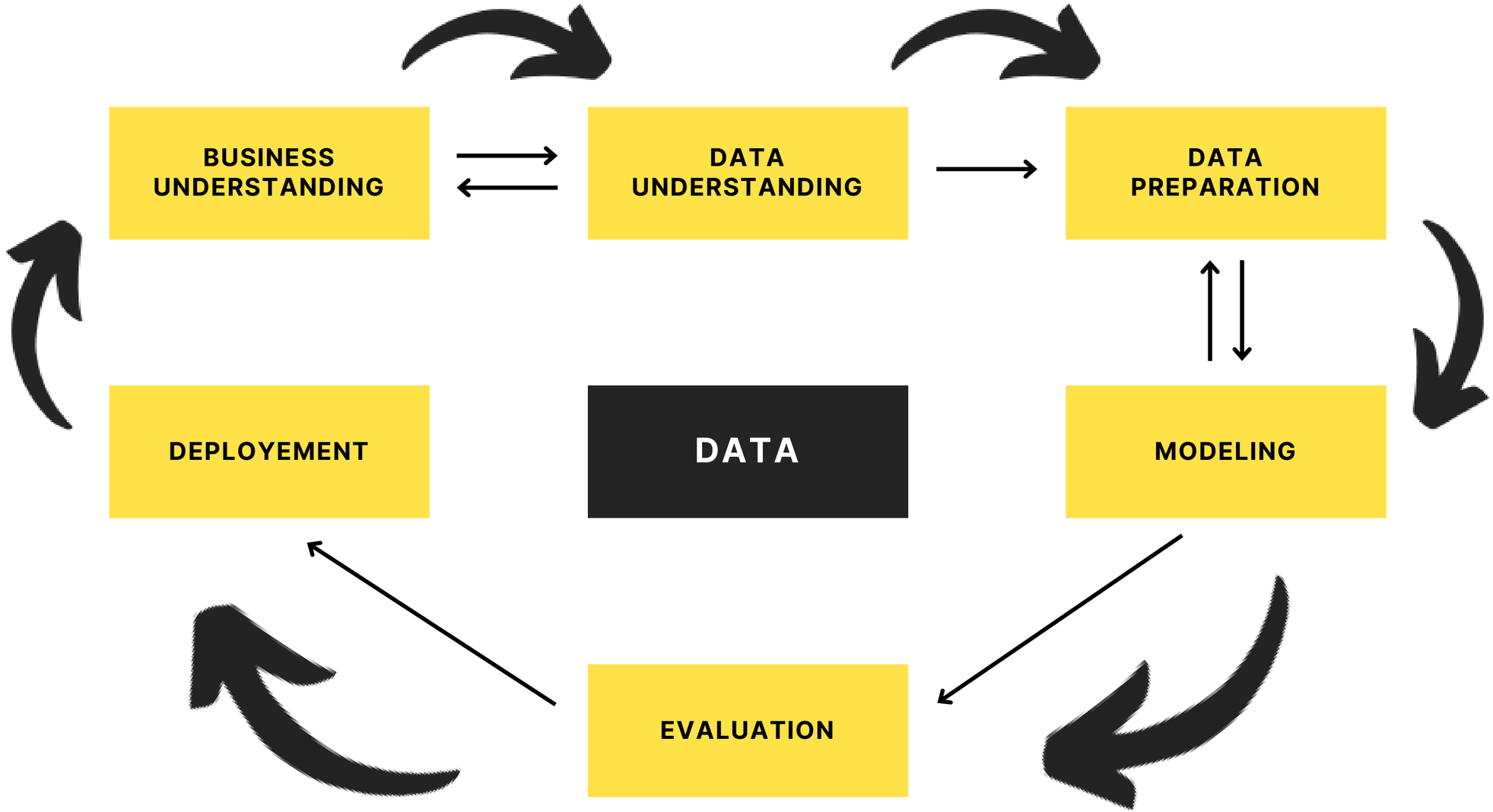
05

EVALUATION

06

DEPLOYMENT

CRISP-DM methodology



BUSINESS UNDERSTANDING

01

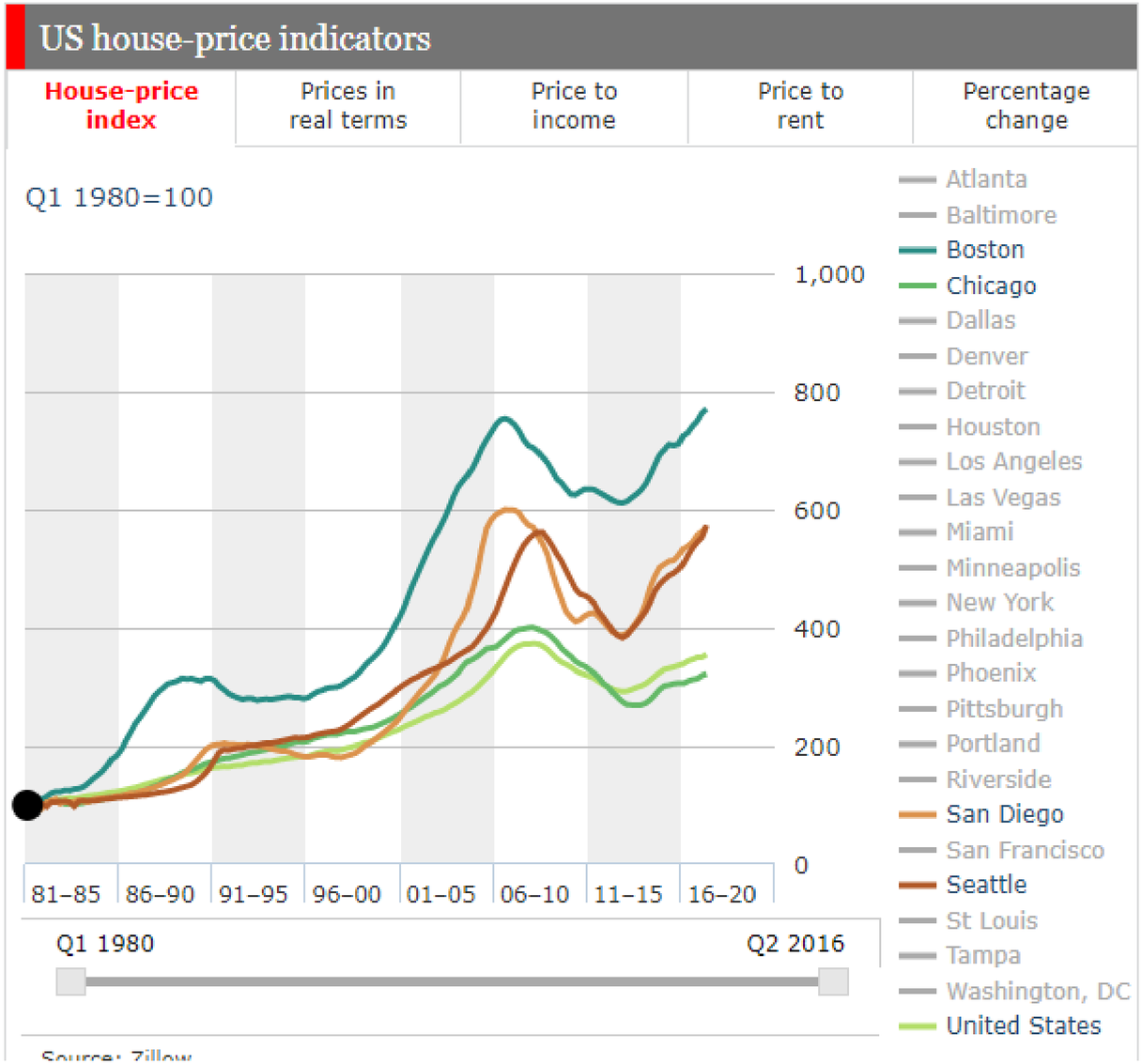
Housing Market Definition:

The housing market represents the market for properties bought and sold either directly to buyers or through realtors.

The price of housing in relation to time

Historical housing price data indicates that real estate prices tend to increase over time.

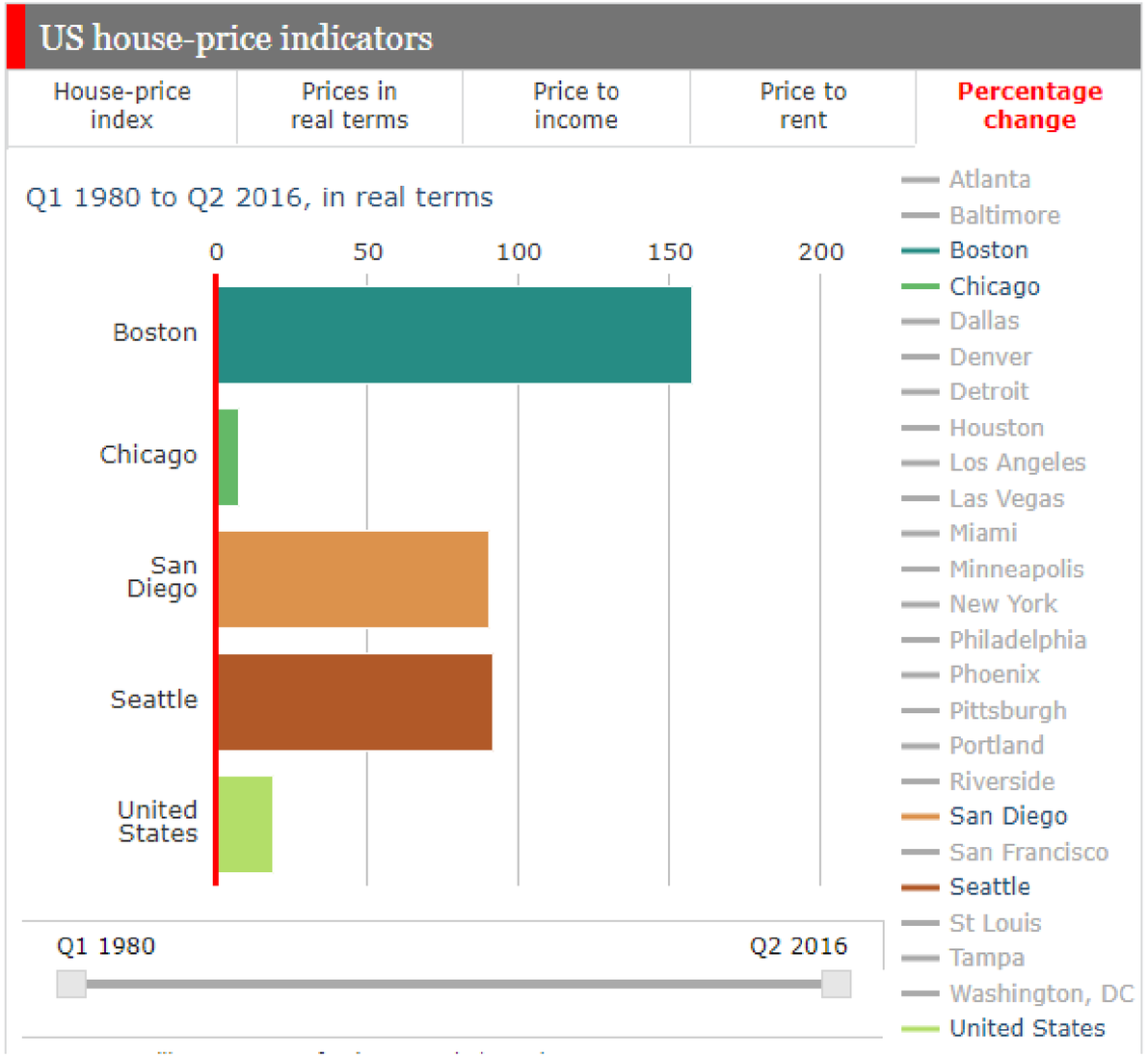
This extraordinary growth in home values can be attributed to increased demand and rising construction costs.



Percentage change

As we can see, in major U.S. cities, from 1980 to 2016, housing prices have increased significantly.

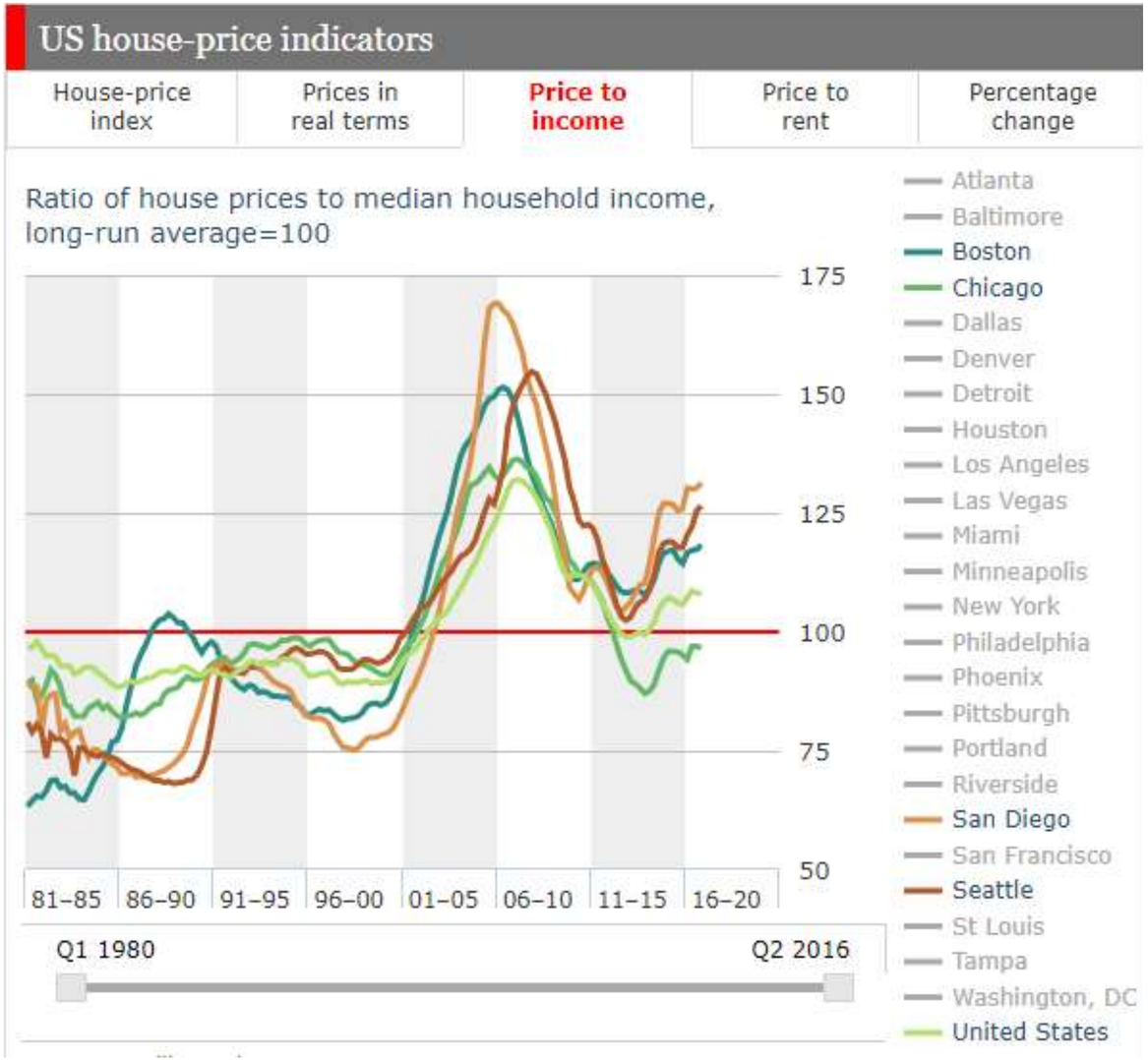
Example: Boston 160%, San Diego and Seattle 80%.



Price vs. income

The graph shows the evolution of the relationship between household income and house prices.

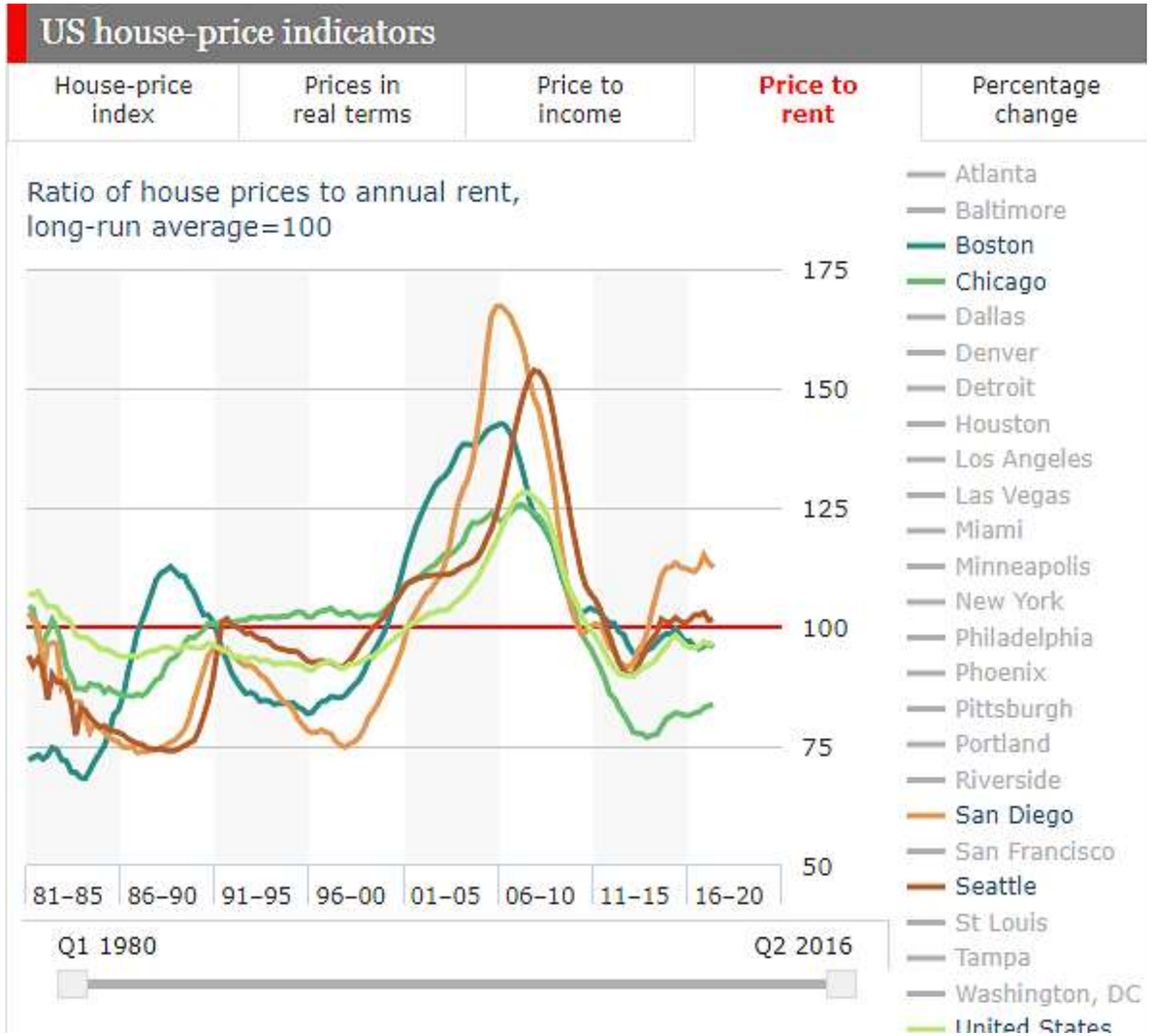
This shows a strong correlation between household wealth, neighborhood affordability and home prices.



Rental price

The graph shows the evolution of the ratio between the price of housing and the annual rent.

The higher the price of the unit, the higher the rent.



Top 4 factors that drive home prices:

01

ECONOMIC CHANGES

When the economy is booming, housing prices rise.
When the economy slows down, housing prices go down.

02

THE SUPPLY AND DEMAND

When supply is low and demand is high, it is a seller's market.
Conversely, it is a buyers' market when supply is high but demand for housing is low.

03

MORTGAGE INTEREST RATES

the Federal Reserve has the power to raise or lower interest rates at specific times of the year. That said, interest rate changes can have an impact on home prices.

04

THE LOCATION

Certain location factors can influence housing prices. If you are looking to buy a home in an area with restaurants, attractions and good schools, home prices will be higher. Other factors that can cause prices to fluctuate include crime rates, access to quality health care and proximity to highways.

DATA UNDERSTANDING

02

Importing data:

Our dataset contains the sales prices of homes in the state of washington ,united states. It includes homes sold between May 2014 and July 2014 (3 months) by Gold Real Estate, a real estate agency in the US.

```
data=pd.read_excel("AgencyDataset.xlsx",header=0)
```

```
data.shape
```

```
(4600, 18)
```

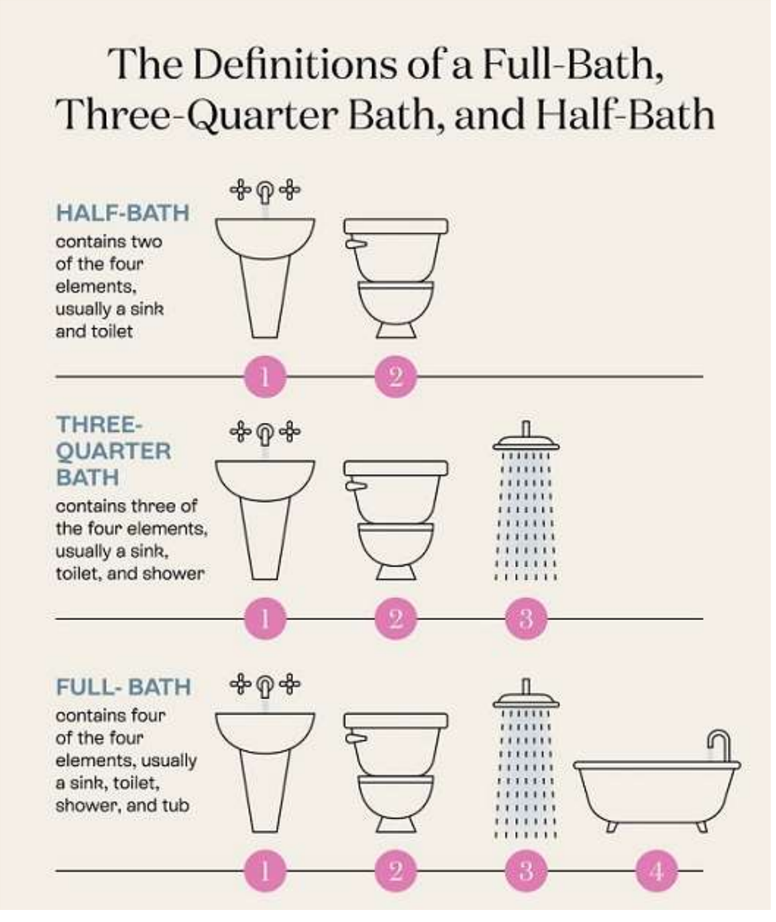
```
data.dtypes.value_counts()
```

```
int64             10
object             4
float64            3
datetime64[ns]     1
dtype: int64
```

Previews:

- **Date:** day/month/year of house data collection
- **price: The price of the house**
- **Bedrooms:** Most homes contain 3 bedrooms (3.4 on average), with the exception of a few homes with up to 9 bedrooms. These are luxury homes but will be treated as outliers.

- **Bathrooms:** At first sight, the reason why bathrooms need to be floating point seems a little strange, but it makes sense if we put it this way:



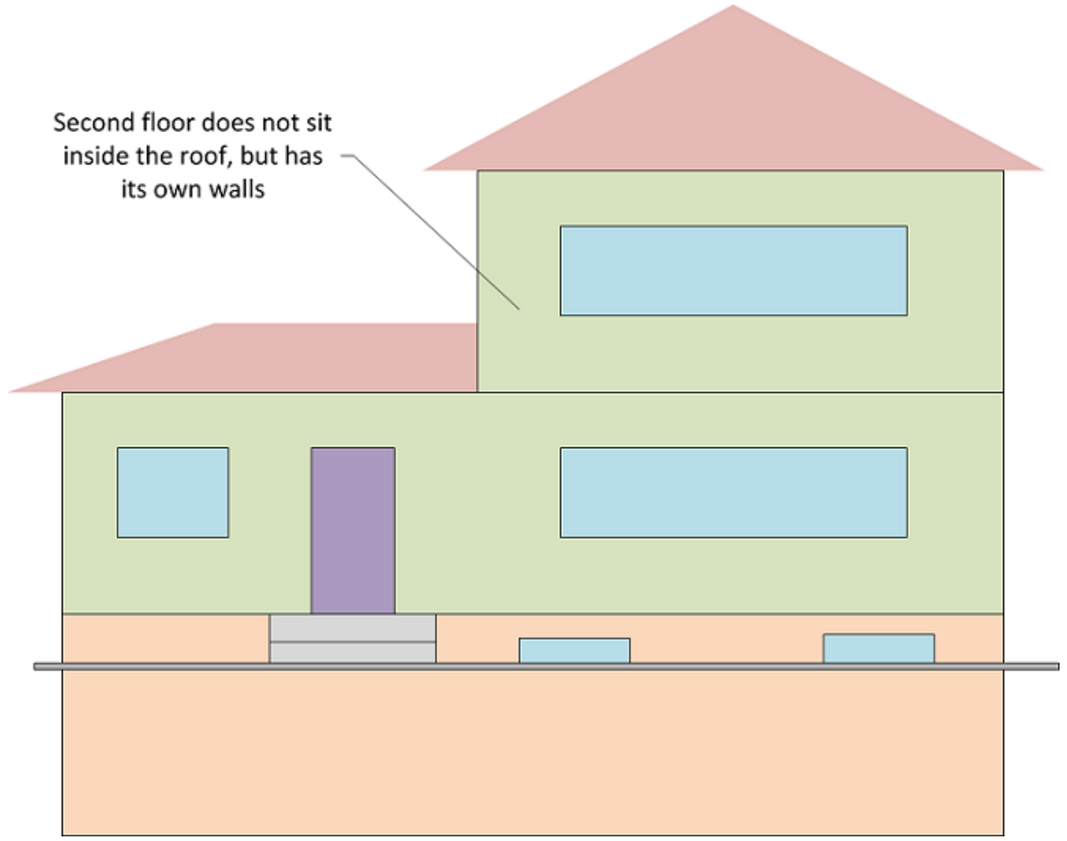
- **Sqft Living:** the surface of the common room
- **Sqft lot:** the total surface of the house
- **Floors:** Number of floors per house

The Floors column contains float values, which makes sense if you present it this way:

1.5 floors means 1 floor and a partial second floor.

2.5 means 2 floors and a partial 3rd floor.

3.5 means 3 floors and a partial 4th floor.



- **Waterfront:**

It is a categorical characteristic that concerns the houses that have a sea view.
« 0 » if it does not have a view
« 1 » if it is in front
We can see that more than 99% of the houses do not have a sea view. This column has low variance and will not add significant value to our model.

- **View:**

This is a categorical characteristic that indicates on a scale of 0 to 4 the quality of the view.
« 0 »: none
« 1 » : poor
« 2 »: fair
« 3 »: Good
« 4 » : Very good
The variance is low because more than 90% of the data have the same value.

- **Condition:** This is a categorical characteristic. It represents the condition of the house on a scale of 1 to 5.

  « 1 »: poor

  « 2 »: fairly poor

  « 3 »: fair

  « 4 »: good

  « 5 »: very good

- **Sqft-above :** roof area in square feet
- **Sqft-basement :** the surface of the underground part (Cave) in square feet
- **Year built :** The year the house was built
- **Year renovated :** The year of renovation of the house
- **Street :** the street on which the house is located
- **City :** The city where the house is located
- **State-zip :** Postal code of the state where the residence is located
- **country :**This is the country where the house is located.

The variance is zero because all the data have the same value.

DATA PREPROCESSING

03

Data Preprocessing

01 **Data Visualization**

02 **Categorical Features Encoding**

03 **Feature Engineering**

04 **Outlier Handling**

05 **Feature Scaling**

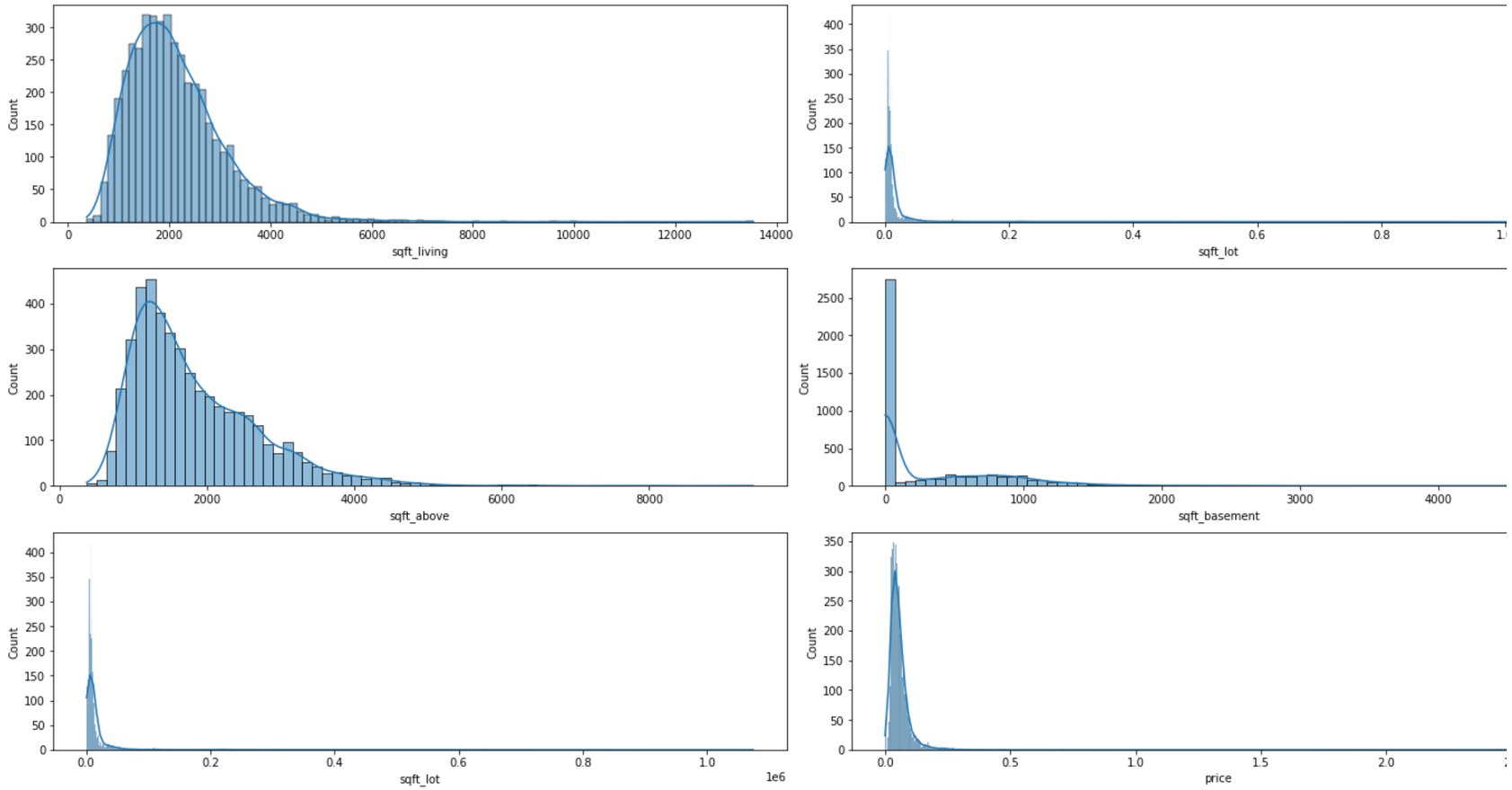
06 **Handling The Missing Values**

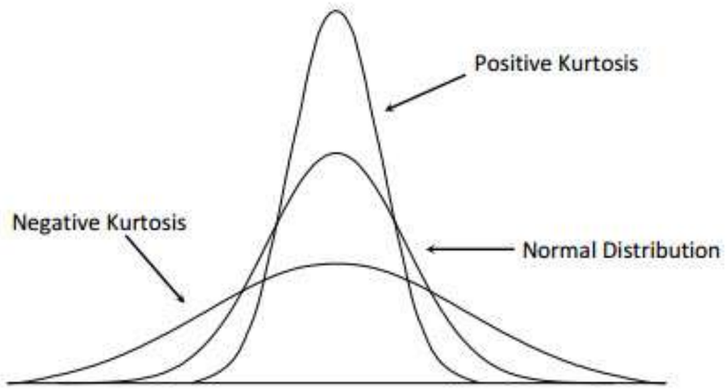
07 **Feature Selection**

08 **Feature Reduction**

Data visualization

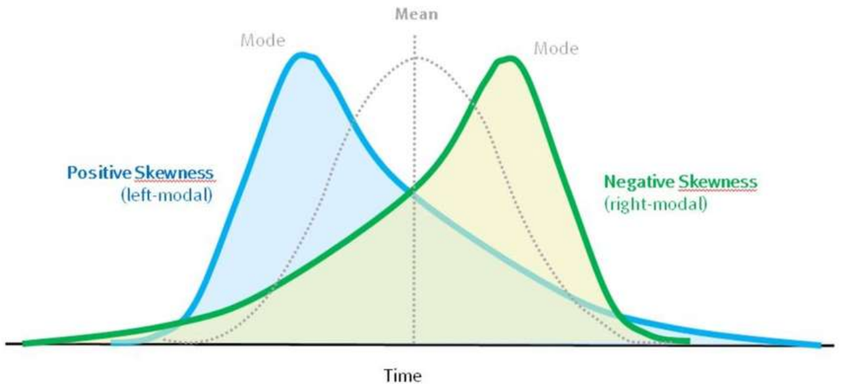
- From the visual examination of the graphical representation, these histograms are used for continuous quantitative variables. They allow to inspect the shape of the distribution.
- Indeed, they represent an asymmetric distribution with respect to a mean.






```
data.kurt()
```

```
price            1044.352151
bedrooms            1.235377
bathrooms           1.865905
sqft_living         8.291683
sqft_lot          219.872987
floors             -0.538852
waterfront        134.548673
view               10.464178
condition           0.197730
sqft_above          4.070138
sqft_basement       4.082380
yr_built           -0.670076
yr_renovated       -1.851111
dtype: float64
```



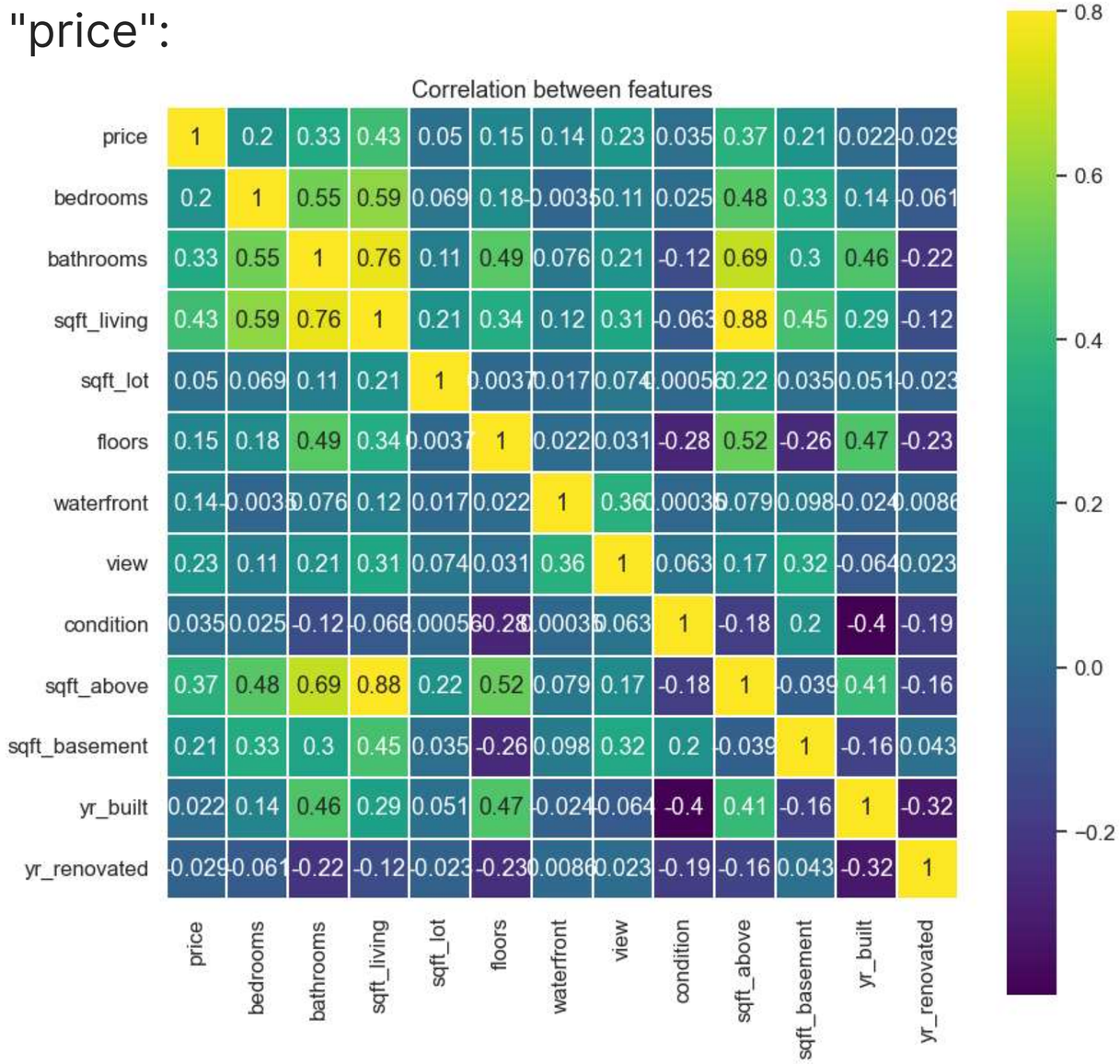

```
data.skew()
```

```
price            24.790933
bedrooms          0.456447
bathrooms         0.616033
sqft_living       1.723513
sqft_lot         11.307139
floors            0.551441
waterfront       11.682901
view              3.341586
condition         0.959068
sqft_above        1.494211
sqft_basement     1.642732
yr_built         -0.502155
yr_renovated      0.385919
dtype: float64
```

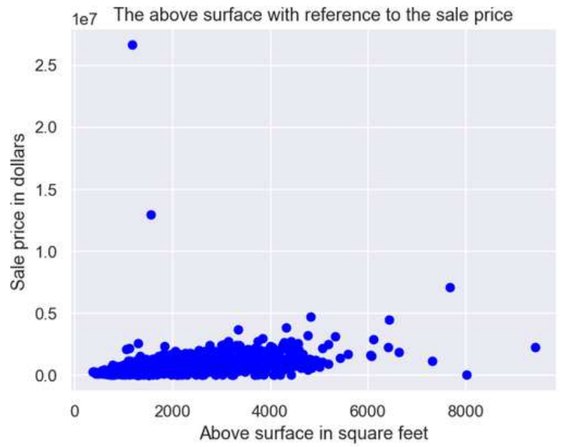
Data correlation

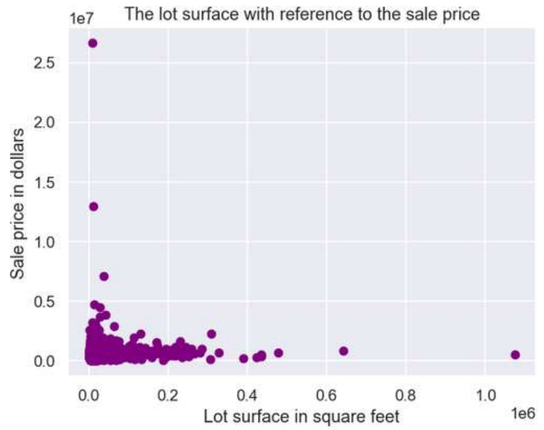
Correlation of the features with our target "price":

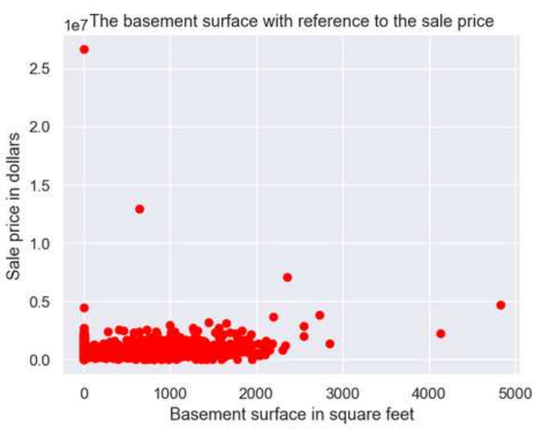
```
price            1.000000
sqft_living      0.430410
sqft_above       0.367570
bathrooms        0.327110
view             0.228504
sqft_basement    0.210427
bedrooms         0.200336
floors           0.151461
waterfront       0.135648
sqft_lot         0.050451
condition        0.034915
yr_built         0.021857
yr_renovated    -0.028774
Name: price, dtype: float64
```

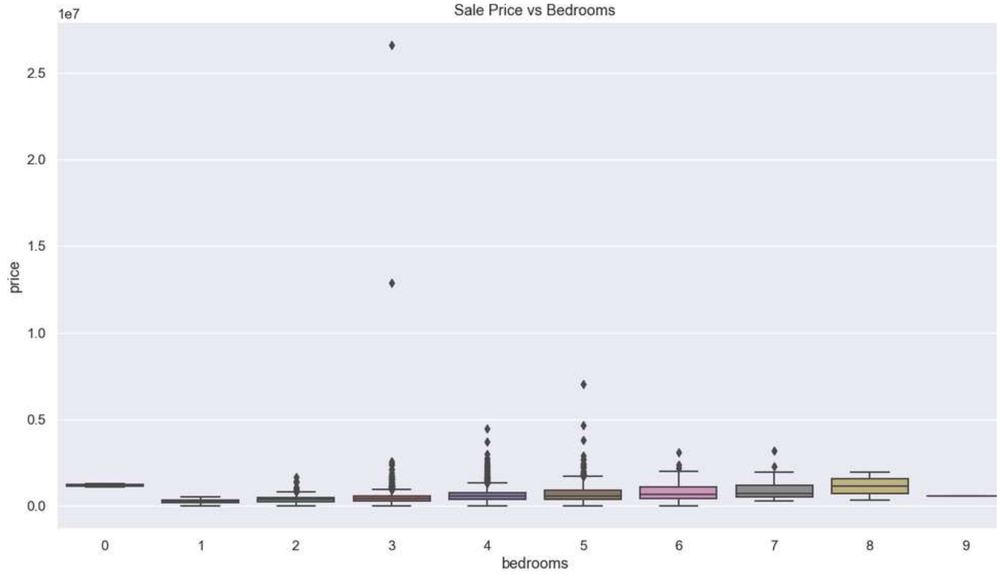


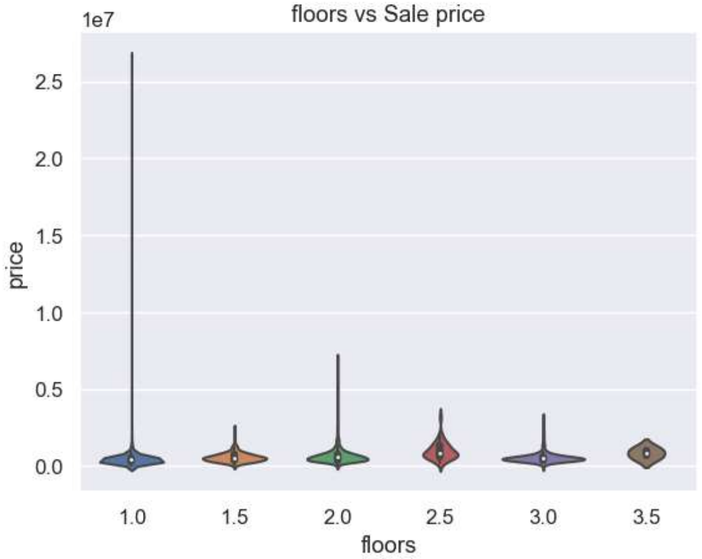
Variation of variables with target

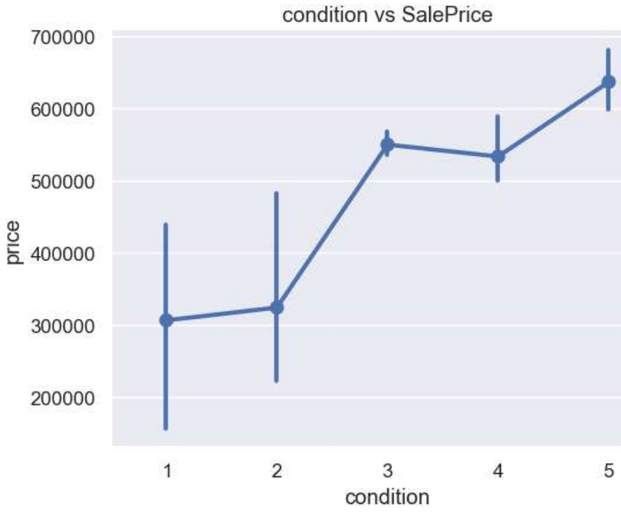


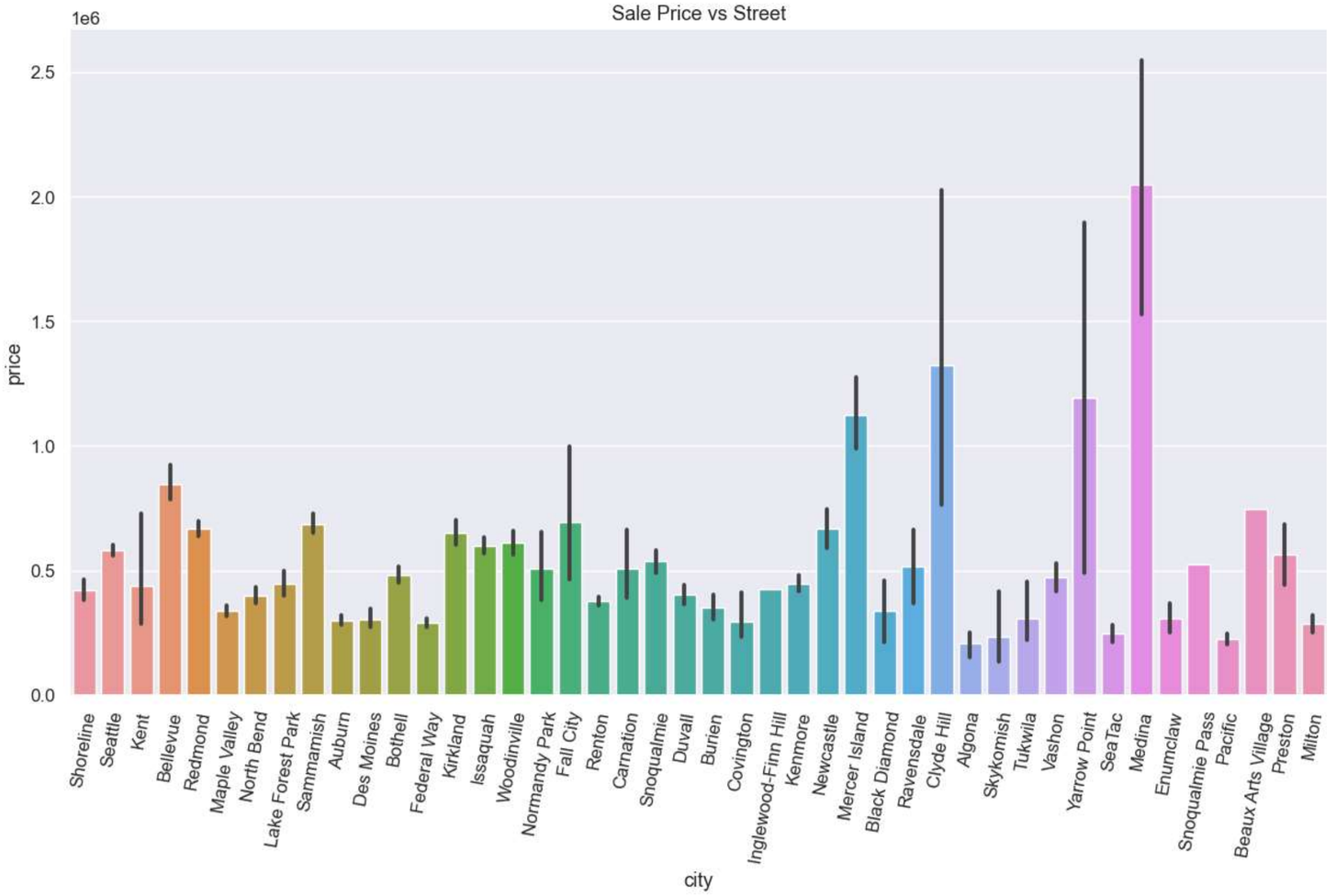


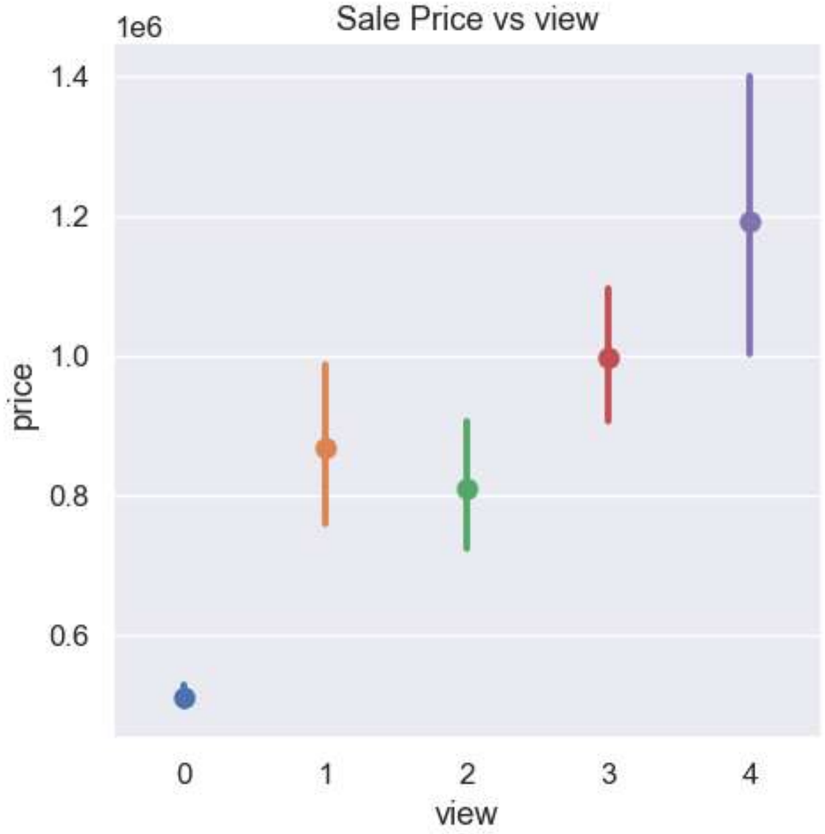









Sale Price vs Street
1e6
2.5
2.0
1.5
1.0
0.5
0.0
price
Shoreline
Seattle
Kent
Bellevue
Redmond
Maple Valley
North Bend
Lake Forest Park
Sammamish
Auburn
Des Moines
Bothell
Federal Way
Kirkland
Issaquah
Woodinville
Normandy Park
Fall City
Renton
Carnation
Snoqualmie
Duvall
Burien
Covington
Inglewood-Finn Hill
Kenmore
Newcastle
Mercer Island
Black Diamond
Ravensdale
Clyde Hill
Algona
Skykomish
Tukwila
Vashon
Yarrow Point
SeaTac
Medina
Enumclaw
Snoqualmie Pass
Pacific
Beaux Arts Village
Preston
Milton
city


Sale Price vs view
1e6
1.4
1.2
1.0
0.8
0.6
price
0
1
2
3
4
view

Data encoding

Statistical learning algorithms only process numerical values as input. Our dataset is made of five qualitative variables, so it was necessary for us to proceed to a transformation of these variables in order to exploit them.

date
2014-05-02
2014-05-02
2014-05-02
2014-05-02
2014-05-02

date
20140502
20140502
20140502
20140502
20140502

Feature Engineering

DATA

FEATURE ENGINEERING
CREATE

NEW FEATURES

GOAL

AVOID THE LOSS OF INFORMATION

IMPROVE MODEL ACCURACY

Feature Engineering

IN OUR DATASET :

PROBLEM: **ADDRESS** → STREET, CITY, STATEZIP, COUNTRY

- **GREAT VARIETY OF OBSERVATIONS**
- **OBJECT**

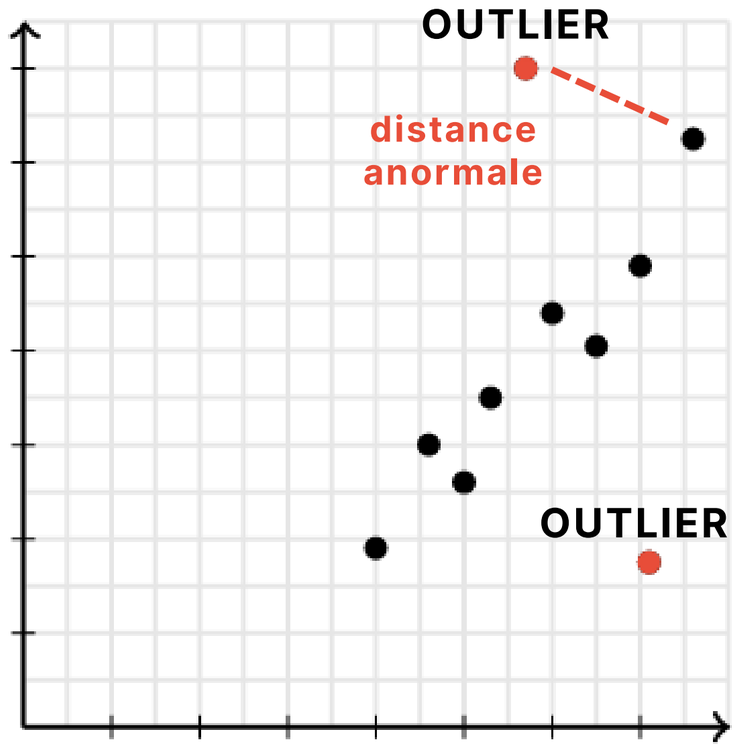
SOLUTION: GeoPy

street	city	statezip	country
18810 Densmore Ave N	Shoreline	WA 98133	USA
709 W Blaine St	Seattle	WA 98119	USA
26206-26214 143rd Ave SE	Kent	WA 98042	USA
857 170th Pl NE	Bellevue	WA 98008	USA

lat	lon
0.677687	0.011479
0.671825	0.011255
0.831570	1.000000
0.671075	0.013282

Outliers

An outlier is an observation that is an abnormal distance from other values in a random sample of a population.



IN OUR DATASET :

We have several observations with:

- over 1,074,218 sq. ft.
- more than 8 bathrooms

=> the validity of these observations should be questioned.

Problems

- Many Machine Learning algorithms are sensitive to outliers

=>This can mislead the training process of machine learning algorithms

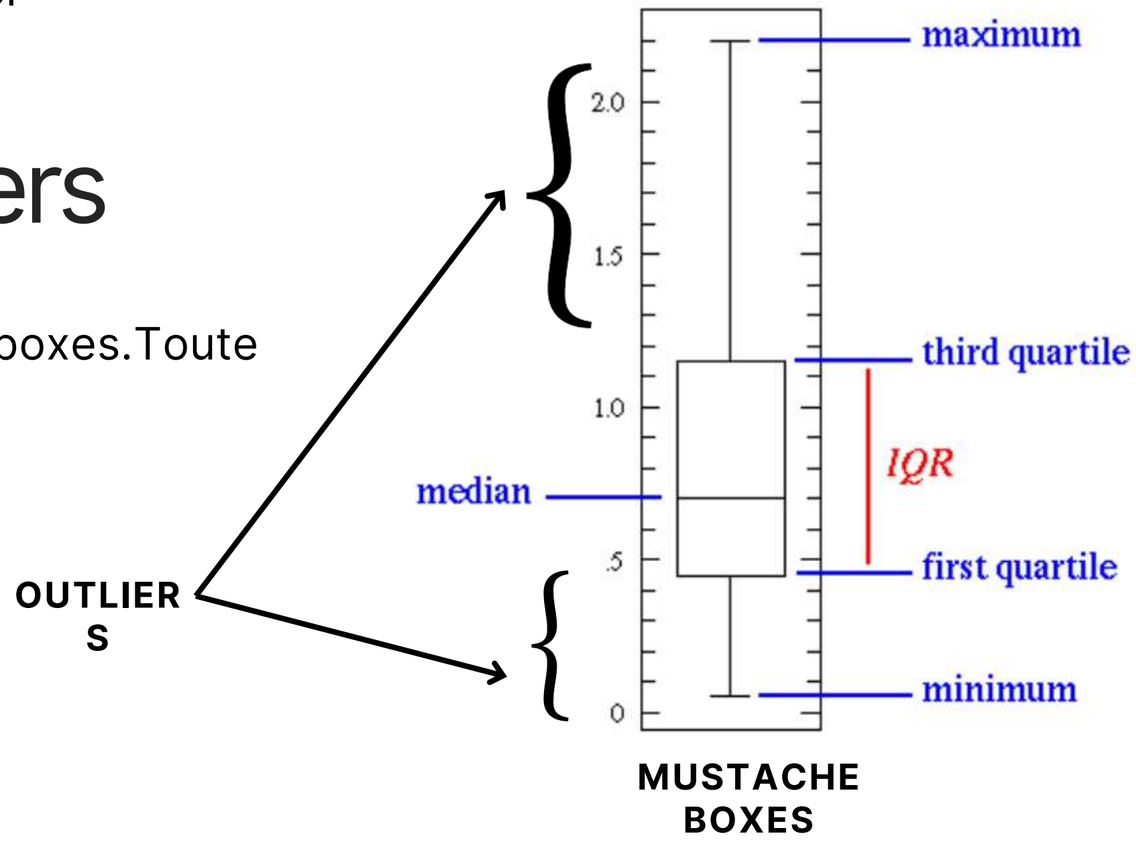
=>which leads to:

- A longer training period
- Less accurate model

Observations of outliers

To observe the outliers, we used whisker boxes.Toute observation ayant une valeur

- Above 3rd quartile
- Less than 1st quartile



Treatment of Outliers

01 **PERCENTILE-BASED METHOD**

02 **INTERQUARTILE RANGE METHOD**

03 **STANDARD DEVIATION METHOD (RULE OF THUMB)**

04 **DBSCAN AND RANDOM FOREST REGRESSOR**

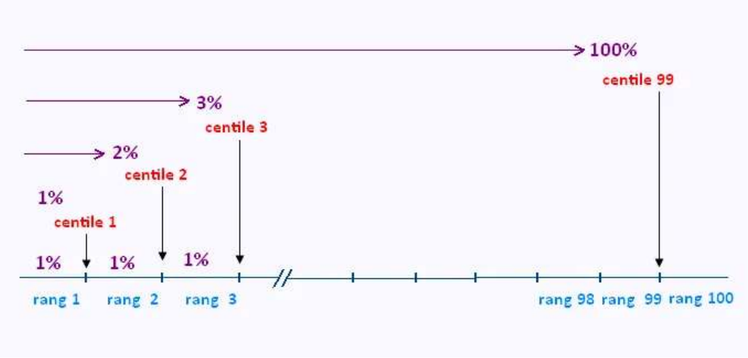
Percentile-based method

A percentile is a statistical concept used to determine the position of a data item in relation to a given group.
The group is divided into 99 shares, each corresponding to a percentile (1/100).
=>By giving as parameters the data and the percentage it returns us the value below which we will have data that corresponds to this percentage.

STEP1 :

```
upper = np.percentile(data[feature], 100 - p)

lower = np.percentile(data[feature], p)
```



STEP2 :

The outliers are compared to the lower and upper

- Obs value < lower → Obs value = lower
- Obs value > upper → Obs value= upper

Interquartile range method

Interquartile range is a measure of dispersion

STEP1 :

IQR = Q3 - Q1

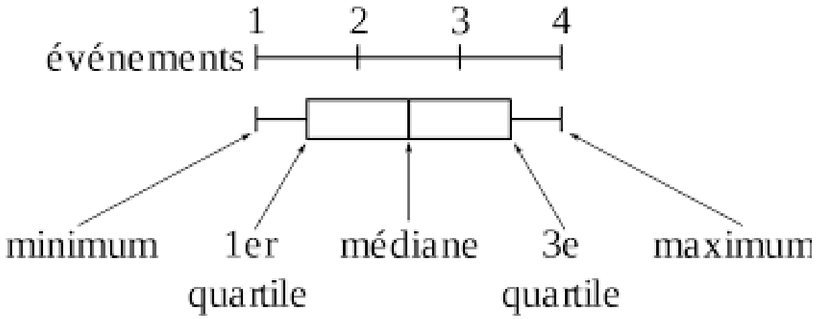
Q3 = np.percentile(data, 75)

Q1 = np.percentile(data, 25)

STEP2 :

IQRU = Q3 + 1.5 * IQR

IQRL = Q1 - 1.5 * IQR



STEP3 :

Outliers are compared to IQRU and IQRL

Obs value < IQRL → Obs value = IQRL

Obs value > IQRU → Obs value = IQRU

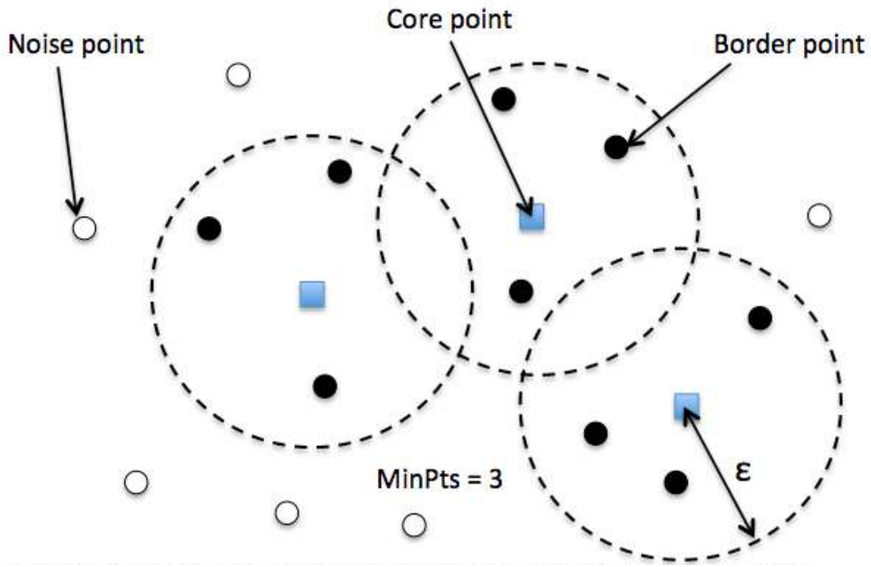
Standard deviation method (rule of thumb)

std = σ(data[feature])

upper_3std = E(data[feature]) + 3 * std

lower_3std = E(data[feature]) - 3 * std

DBSCAN and random Forest Regressor



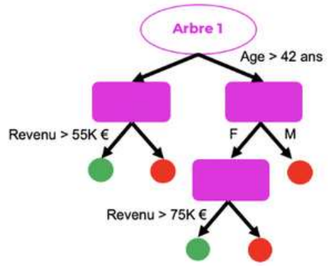
1 Construction de l'arbre N°1 du random forest

Feature sampling

Tree bagging

Id client	Age	Revenu	Sexe	Mariage	Propriétaire	CSP	Apport personnel
id0001	31	32K	M	1	0	Ouvriers qualifiés	0
id0002	71	42K	M	1	1	Non déterminé	45 000 €
id0003	53	72K	M	0	1	Employés qualifiés	0 €
id0004	45	105K	F	0	1	Cadres	115 000 €
id0005	32	55K	M	1	0	Employés qualifiés	45 000 €
id0006	35	58K	F	0	1	Cadres	25 000 €
id0007	66	35K	M	0	1	Non déterminé	15 000 €
id0008	47	112K	F	1	0	Cadres	245 000 €
id0009	51	85K	F	1	0	Cadres	65 000 €
id0010	52	80K	M	1	1	Cadres	85 000 €

Id client	Age	Revenu	Sexe
id0002	71	42K	M
id0003	53	72K	M
id0004	45	105K	F
id0005	32	55K	M
id0006	35	58K	F
id0007	66	35K	M



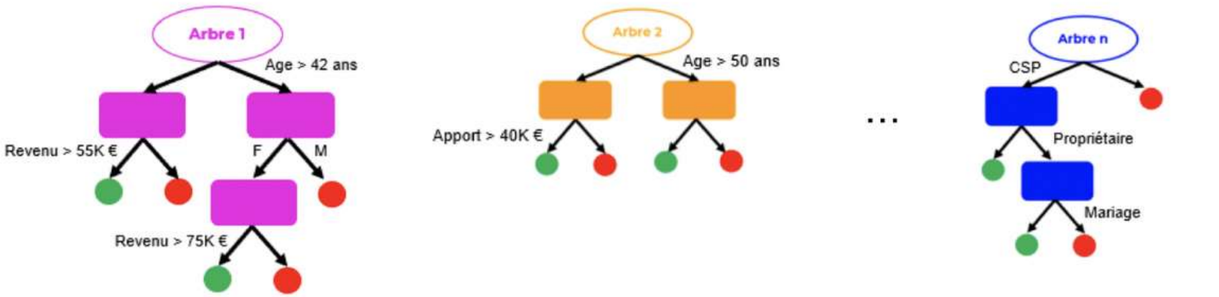
2 Mise en œuvre d'une forêt d'arbres

Id client	Age	Revenu	Sexe
id0002	71	42K	M
id0003	53	72K	M
id0004	45	105K	F
id0005	32	55K	M
id0006	35	58K	F
id0007	66	35K	M

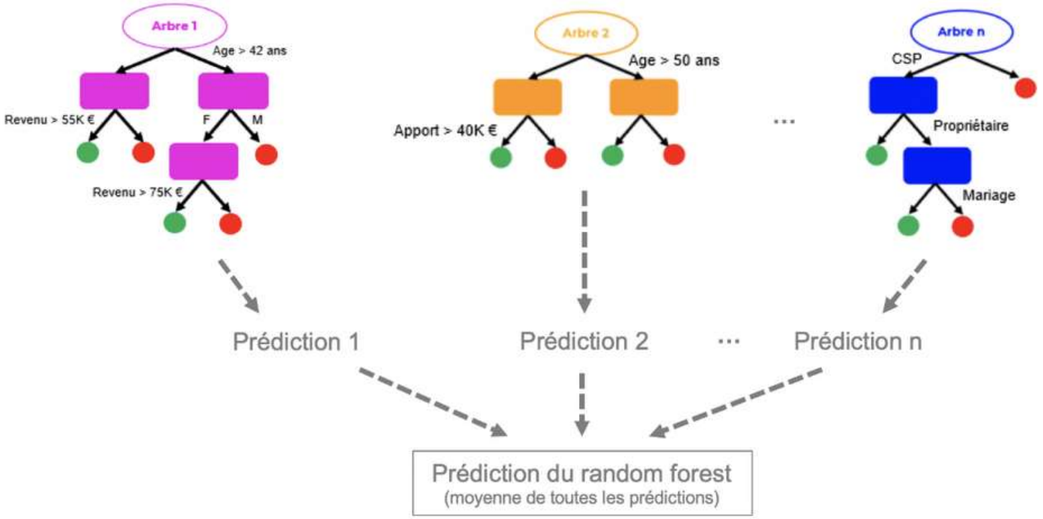
Id client	Age	CSP	Apport personnel
id0001	31	Ouvriers qualifiés	0
id0007	66	Non déterminé	15 000 €
id0008	47	Cadres	245 000 €
id0009	51	Cadres	65 000 €
id0010	52	Cadres	85 000 €

...

Id client	Mariage	Propriétaire	CSP
id0001	1	0	Ouvriers q
id0003	0	1	Employés c
id0005	1	0	Employés c
id0007	0	1	Non déte
id0009	1	0	Cadr

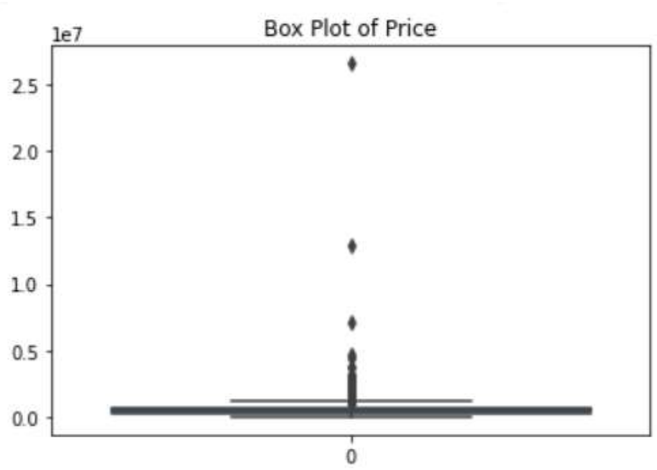


3 Prédiction du random forest



Choice of method for outlier imputation

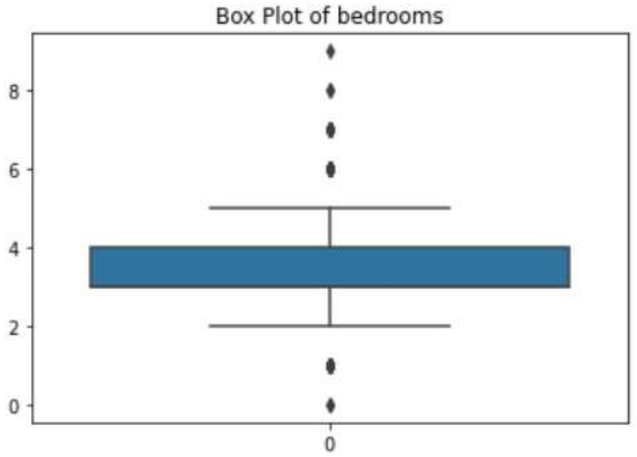
1. price




```
Kurtosis pct method:  5.140566941330823
Kurtosis iqr method:  88.98631677328197
Kurtosis std method:  6.723717169192335
Kurtosis RandomForest Regression:  7.684001585763996
```

```
data["price"]=data1["price"]
```

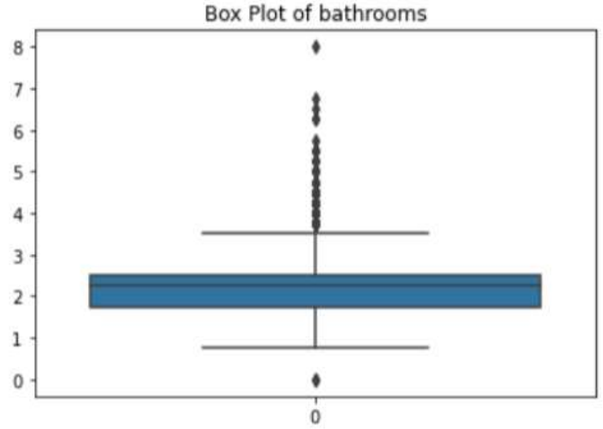
2. bedrooms




```
Kurtosis pct method:  0.17539658418041837
Kurtosis iqr method:  1.2353774286379018
Kurtosis std method:  0.3802873589431246
Kurtosis RandomForest Regression:  0.986884170118034
```

```
data["bedrooms"]=data1["bedrooms"]
```

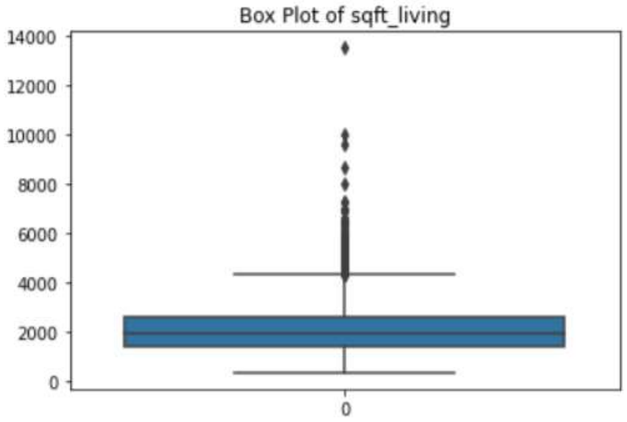
3. bathrooms




```
Kurtosis pct method:  0.2128525624666862
Kurtosis iqr method:  1.8659047097303292
Kurtosis std method:  0.2290019702023267
Kurtosis RandomForest Regression:  1.3050179890459
```

```
data["bathrooms"]=data1["bathrooms"]
```

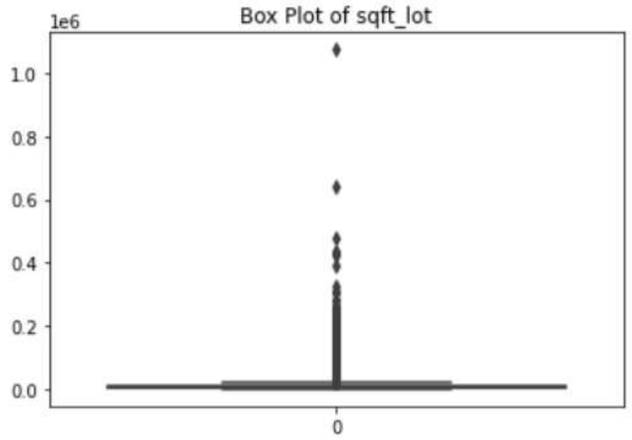
4. sqft_living




```
Kurtosis pct method:  0.879878939369182 6
Kurtosis iqr method:  0.704448622266372 4
Kurtosis std method:  0.720511221686066 1
Kurtosis RandomForest Regression:  0.223232320389650
```

```
data["sqft_living"]=data4["sqft_living"]
```

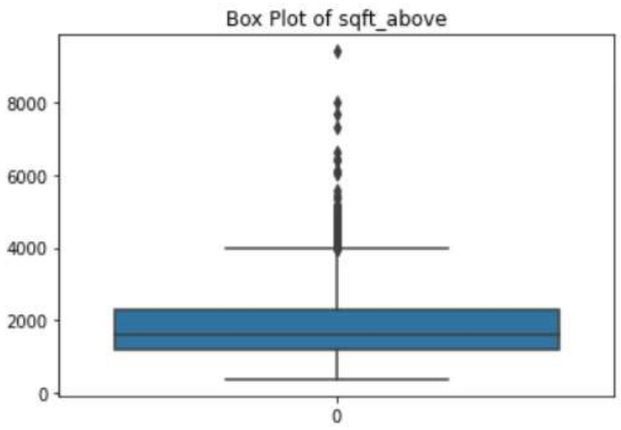
5. sqft_lot




```
Kurtosis pct method:  33.05859349606376
Kurtosis iqr method:  5.312783819748651
Kurtosis std method:  18.597967731444868
Kurtosis RandomForest Regression:  315.26852723370325
```

```
data["sqft_lot"]=data2["sqft_lot"]
```

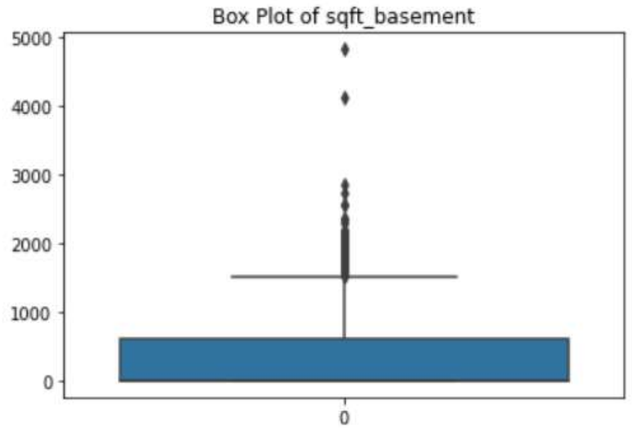
6. sqft_above




```
Kurtosis pct method:  0.7377662775301248
Kurtosis iqr method:  1.192825226789636
Kurtosis std method:  0.6856338939781694
Kurtosis RandomForest Regression:  0.4975085417058
```

```
data["sqft above"]=data4["sqft above"]
```

7. sqft_basement




```
Kurtosis pct method:  0.4598943593711282
Kurtosis iqr method:  4.082380024138892
Kurtosis std method:  0.4668054199276956
Kurtosis RandomForest Regression:  0.761148548580
```

```
data["sqft_basement"]=data1["sqft_basement"]
```

Feature Scaling

- Normalization and standardization are the two main techniques of the FS that allow to redefine the scale of the features.

When do we need feature scaling?

- **K-Nearest Neighbors (KNN)** with a Euclidean distance measure. (Supervised)
- **K-Means** uses the Euclidean distance measure (Unsupervised)
- Feature Scaling is essential when performing PCA. **PCA** tries to get the features with maximum variance, and the variance is high for the high magnitude features and directs PCA to the high magnitude features.
- We can speed up the **descent of the gradient** by scaling as it inefficiently oscillates to the optimum when the variables are very unequal.

Some ways to perform feature scaling:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

$$z = \frac{x - \mu}{\sigma}$$

$$x_{scaled} = \frac{x}{max(|x|)}$$

Min-Max scaler

This Scaler shrinks the data in the range of -1 to 1 if there are negative values. It responds well if the distribution is not Gaussian but the scaler is sensitive to outliers.

Standard Scaler

Scales the distribution to be centered around 0, with a standard deviation of 1. It assumes that the data are normally distributed in each feature.

Max Abs Scaler

Scales each feature by its maximum absolute value. It suffers from the presence of significant outliers.

date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_basement	yr_built	yr_renovated	lat	lon
20140502	3.130000e+05	3	1.50	1340.0	5012.5	1.5	0	0	3	1340.0	0	1955	2005	47.765638	-122.339029
20140502	2.005220e+06	5	2.50	3650.0	5012.5	2.0	0	4	5	3370.0	280	1921	0	47.634542	-122.366908
20140502	3.420000e+05	3	2.00	1930.0	5012.5	1.0	0	0	4	1930.0	0	1966	0	51.207075	0.721036
20140502	4.200000e+05	3	2.25	2000.0	5012.5	1.0	0	0	4	1000.0	1000	1963	0	47.617770	-122.114613
20140502	5.500000e+05	4	2.50	1940.0	5012.5	1.0	0	0	4	1140.0	800	1976	1992	47.683043	-122.113957

MIN-MAX SCALER

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_basement	yr_built	yr_renovated	lat	lon
0	0.000000	0.152797	0.25	0.142857	0.154959	1.0	0.2	0.0	0.0	0.50	0.180365	0.000000	0.482456	0.995531	0.677687	0.011479
1	0.000000	1.000000	0.75	0.428571	0.632231	1.0	0.4	0.0	1.0	1.00	0.643836	0.058091	0.184211	0.000000	0.671825	0.011255
2	0.000000	0.167316	0.25	0.285714	0.276860	1.0	0.0	0.0	0.0	0.75	0.315068	0.000000	0.578947	0.000000	0.831570	1.000000
3	0.000000	0.206366	0.25	0.357143	0.291322	1.0	0.0	0.0	0.0	0.75	0.102740	0.207469	0.552632	0.000000	0.671075	0.013282
4	0.000000	0.271450	0.50	0.428571	0.278926	1.0	0.0	0.0	0.0	0.75	0.134703	0.165975	0.666667	0.989076	0.673994	0.013287

Handling The Missing Values

Causes of missing data

- The data is not filled in intentionally, especially if it is an optional field.
- Corrupted data.
- Human error.
- Fraudulent behavior to intentionally delete data.

The problem of missing values:

Most machine learning algorithms are unable to process incomplete data.

Solutions :

- Delete columns with missing values

-> If you have a column with more than 80% missing values, it is better to delete it.

- Delete rows with missing values

-> If you have a small percentage of rows with missing values, you can simply delete these rows.

- Imputation of missing values

Handling The Missing Values

IN OUR DATASET :

We don't have any columns with more than 80% missing values, so deleting the price, bathrooms, sqft_living, and year_renovated columns is not an appropriate solution. We have a small dataset, so deleting rows with missing values is a waste of information.

```
print(data.isnull().sum()*100/len(data))
```

```
date             0.000000
price            1.065217
bedrooms         0.000000
bathrooms        0.043478
sqft_living      0.000000
sqft_lot         0.000000
floors           0.000000
waterfront       0.000000
view             0.000000
condition        0.000000
sqft_above       0.000000
sqft_basement    0.000000
yr_built         0.000000
yr_renovated     4.239130
street           0.000000
city             0.000000
statezip         0.000000
country          0.000000
dtype: float64
```

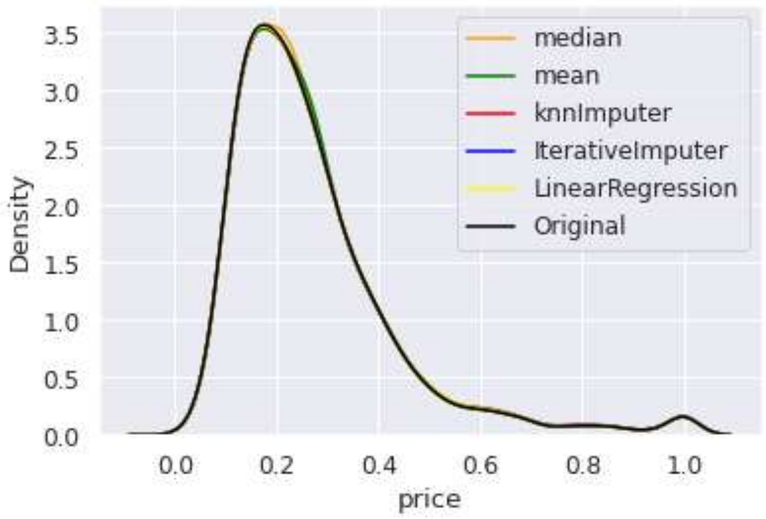
SOLUTION

The preferred solution is the imputation of missing values with which we have worked:

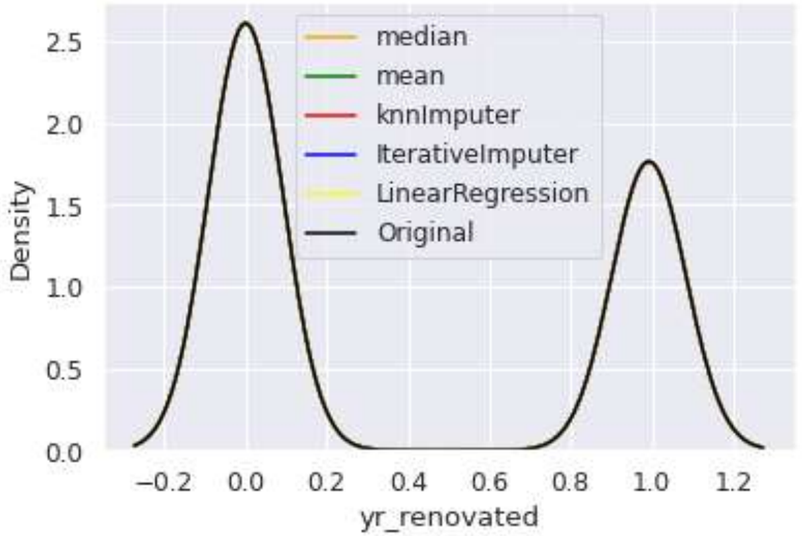
- mean
- median
- KNNImputer
- Iterative Imputer with Extra Trees Regressor as estimator
- Linear regression preceded by Simple Imputer

For the lines where we have price or bathromms null, and where we have year_built superior to yr_renovated and sqft_living superior to sqft_lot. We declare price , bathromms, sqft_living , yr_renovated missing values and we impute them by these algorithms.
Among all the imputation techniques, the one with the most compatible distribution curve on the original distribution curve is chosen.

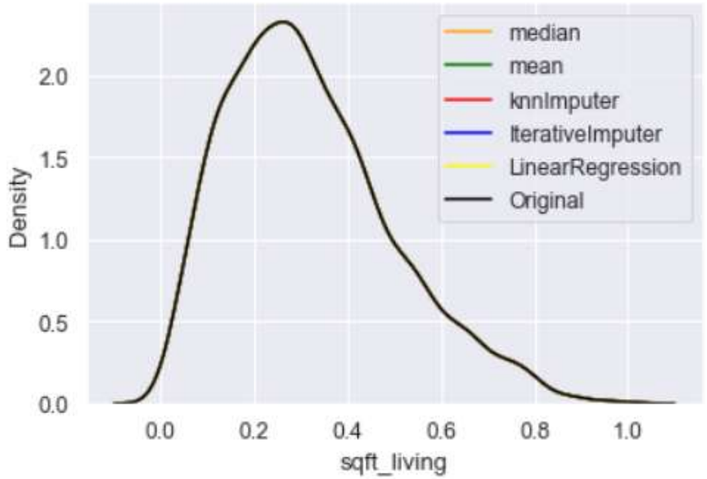
1.PRICE COLUMN



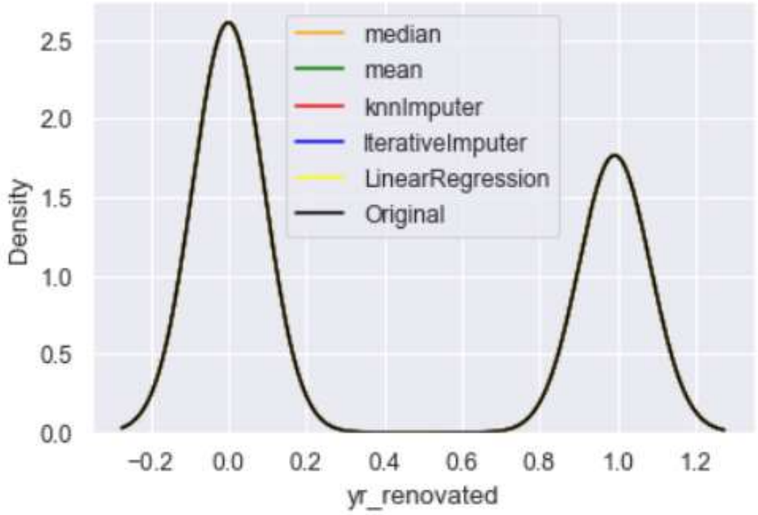
2.BATHROOM COLUMN



3.SQFT_LIVING COLUMN



4.YR_RENOVATED COLUMN



Handling The Missing Values

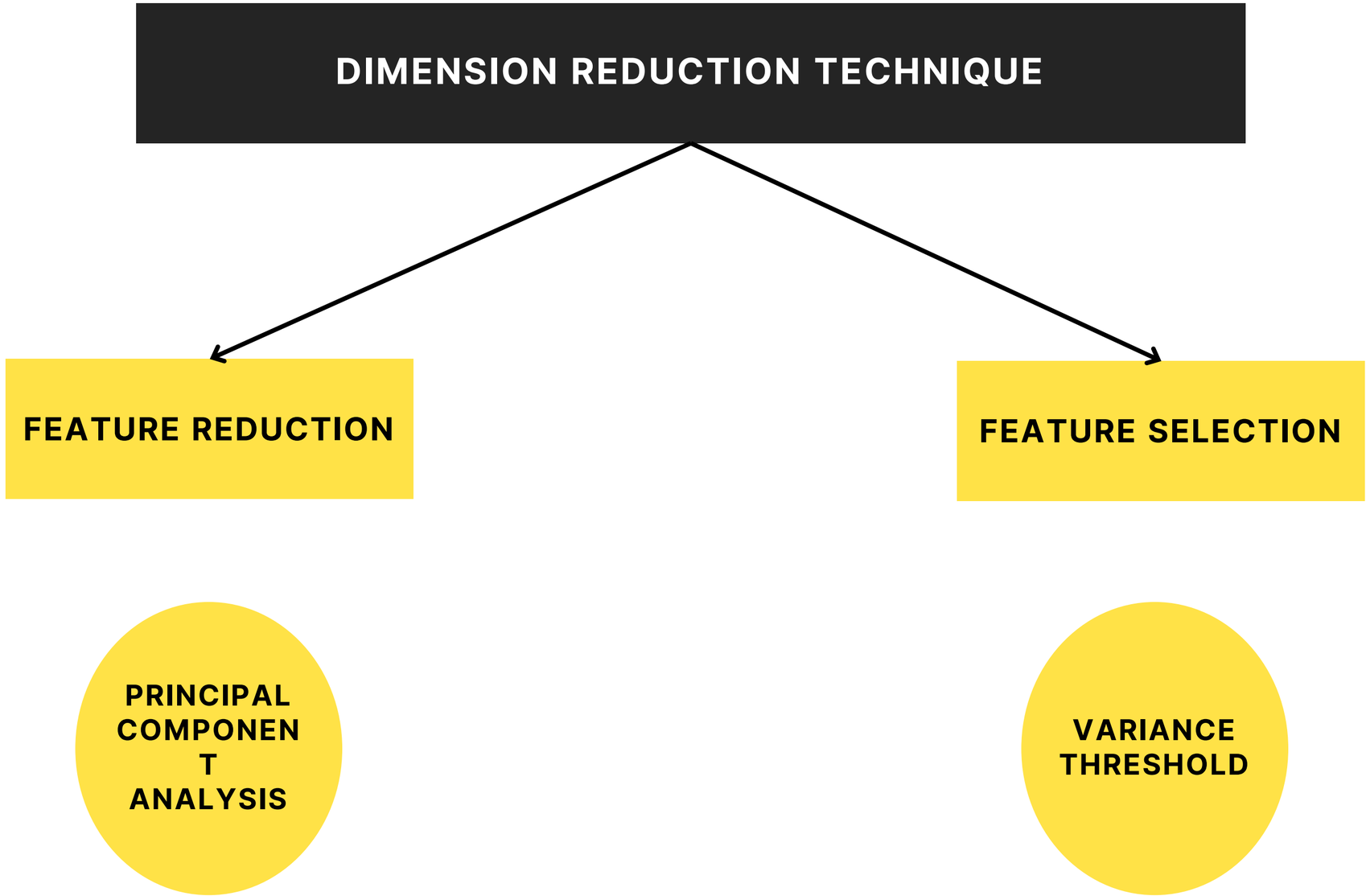
Simple imputation

QUANTITATIVE

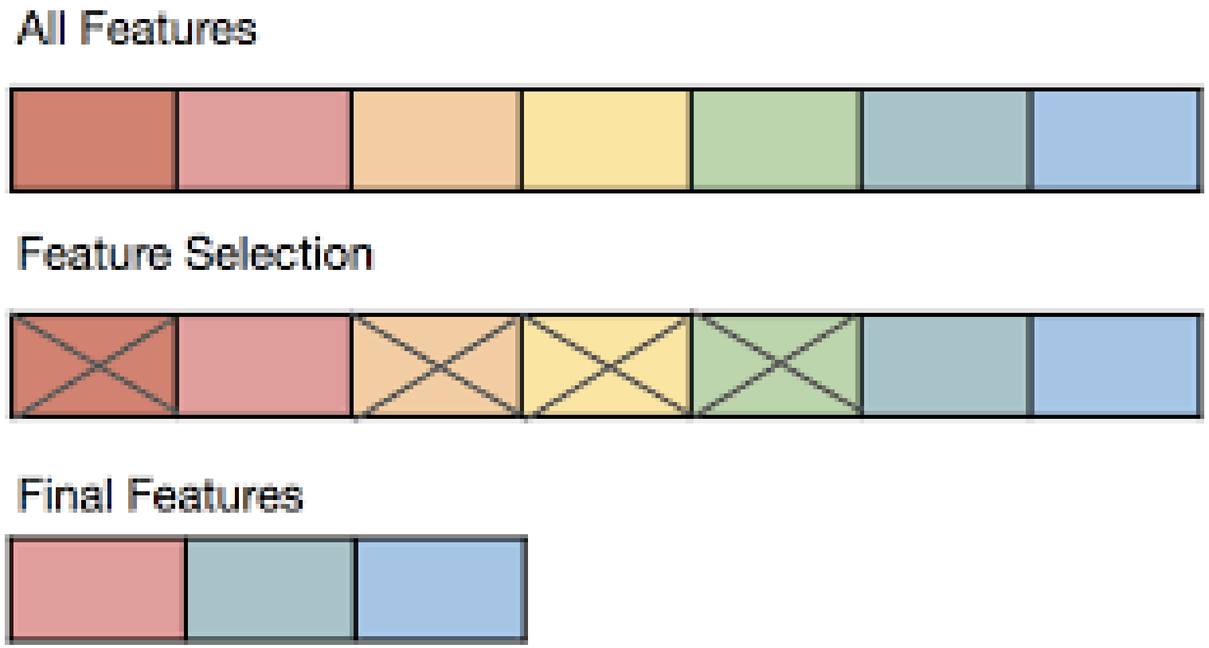
- **MEAN**
- **MEDIAN**

QUALITATIVE

- **MODE**


DIMENSION REDUCTION TECHNIQUE
FEATURE REDUCTION
FEATURE SELECTION
PRINCIPAL COMPONENT ANALYSIS
VARIANCE THRESHOLD

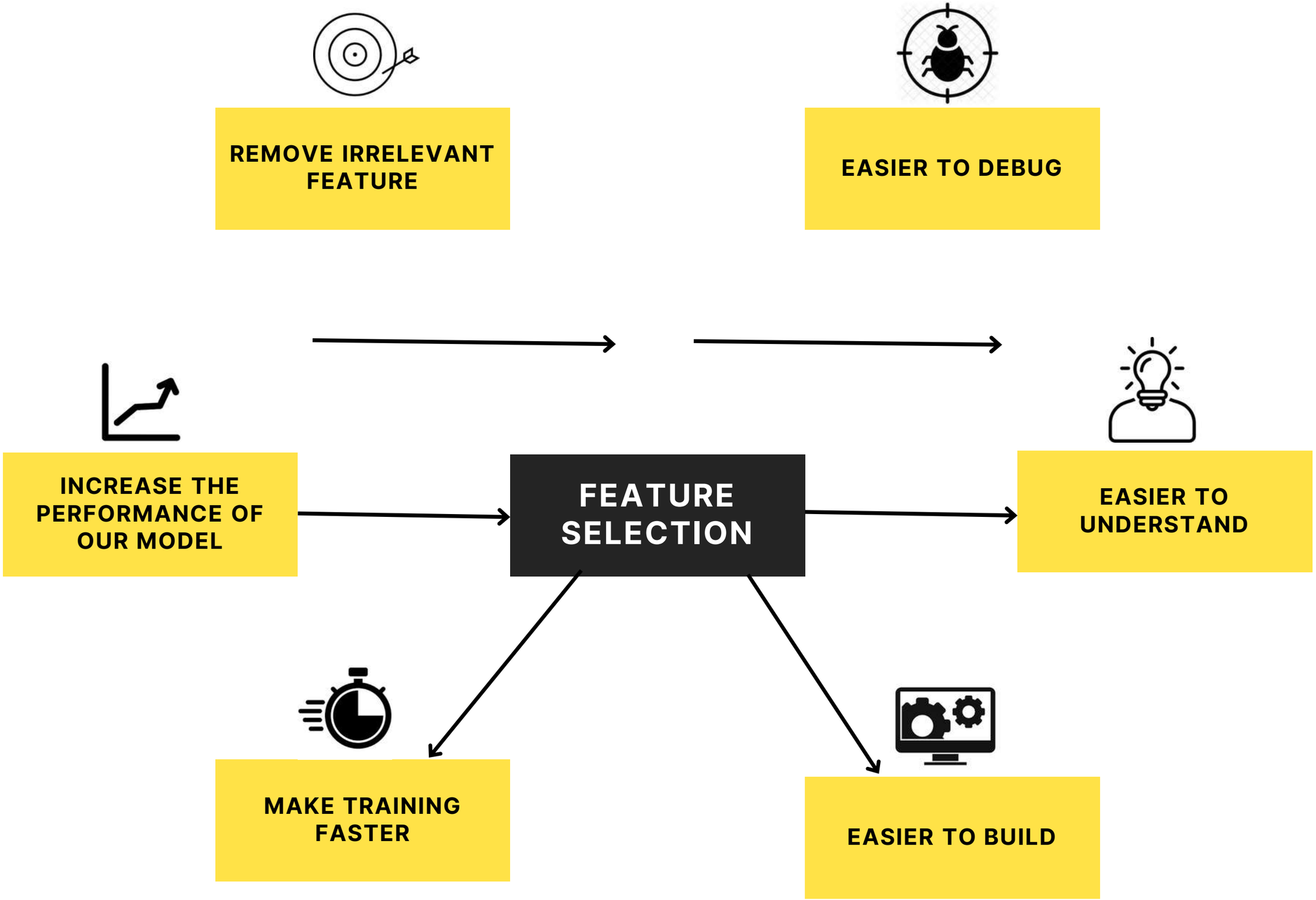
Feature Selection



Why is feature selection important?

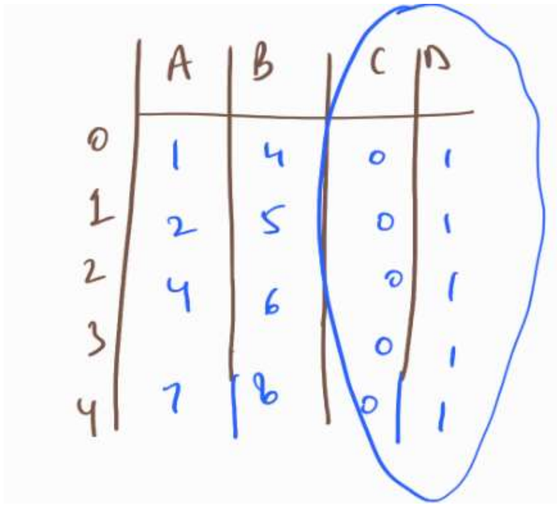
Feature selection is one of the key concepts in machine learning, which has a significant impact on model performance.

Irrelevant and misleading data features can have a negative impact on the performance of our model.


REMOVE IRRELEVANT FEATURE
EASIER TO DEBUG
INCREASE THE PERFORMANCE OF OUR MODEL
FEATURE SELECTION
EASIER TO UNDERSTAND
MAKE TRAINING FASTER
EASIER TO BUILD

Feature removal using Variance Threshold

- Constant features have similar values across all dataset observations. These features do not provide any information for the models to predict the target.



C,D COLUMNS HERE ARE CONSTANT FEATURES

- **Variance Threshold** is a feature selector that removes from the dataset all features whose variance is below the threshold.

```
from sklearn.feature_selection import VarianceThreshold
var_thr = VarianceThreshold(threshold = 0.008) #Removing both constant and quasi-constant
var_thr.fit(data)
```

OUTPUT:

True: High Variance
False: Low Variance

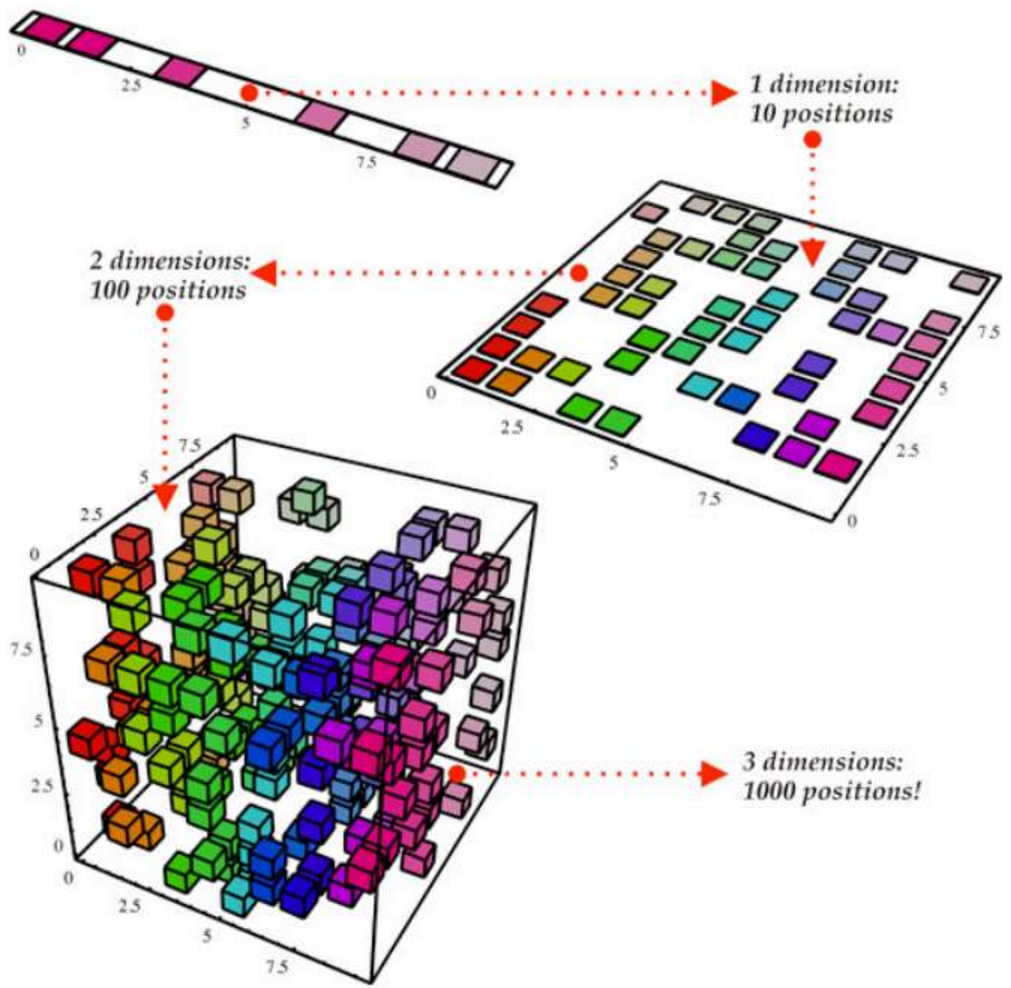
Feature Reduction

LARGE SIZE

Dimensional reduction

Reduce the number of variables

MORE LIMITED



What is the benefit of dimensionality reduction?

The reduction of the number of variables in the training data leads de facto to a better Robustness

- Stability of the algorithm

=> This allows for faster learning by:

- Limiting time
- Computing resources

What methods are used for dimensionality reduction?

PCA (principal component analysis) :

Principal component analysis is one of the factor analysis methods.
PCA will help us to get closer to an overall view of our data. And if it is visual and easy, we like it.

MODELING AND EVALUATION

04

Modelling and Evaluation

01 **Random Forest regressor**

02 **Support vector regression**

03 **Linear regression**

We used 3 methods: Random forest regression, Linear regression and SVM.

Split Train and Test set :

We separated the data between training and test data.

```
selection import train_test_split
train, y_test = train_test_split(X, y, test_size=0.3
```

Hyperparameter tuning :

For Random forest regression and svm, we estimated the hyperparameters using Gridsearch :

```
rf_random.best_params_
```

```
{'n_estimators': 140,
 'min_samples_split': 15,
 'min_samples_leaf': 2,
 'max_features': 'auto',
 'max_depth': 50}
```

```
# print best parameter after tuning
print(grid.best_params_)
```

```
{'C': 100, 'gamma': 0.01, 'kernel': 'rbf'}
```

Random forest regression

Training :

```
rf_random.fit(X_train, y_train)
```

Evaluation:

```
MAE: 0.049204290766787 3
MSE: 0.0079114740671526 22
RMSE: 0.088946467423684 8
Score (R^2): 0.713172836083841 4
```

Support vector regressor

Training :

```
grid.fit(X_train, y_train)
```

Evaluation :

```
MAE: 0.0747469100942391
MSE: 0.012677993786645943
RMSE: 0.11259659758023749
Score (R^2): 0.5403646689473265
```

Linear Regression

Training :

```
lr1.fit(X_train,y_train)
```

Evaluation :

```
MAE: 0.07634368527139931
MSE: 0.012902233786239808
RMSE: 0.11358800018593429
Score (R^2): 0.532234942100706 4
```

Comparison of models

According to the 3 models, we found that the model which has the least RSME and the greatest R squared is the best model which is the Random forest regression which will be deployed.

Model Comparaison

```
labels=['rf','svr','lr']
plot = pd.DataFrame({'labels':labels,"RSME":RSME,"R-sqr":R_sqr})
plot.head()
```

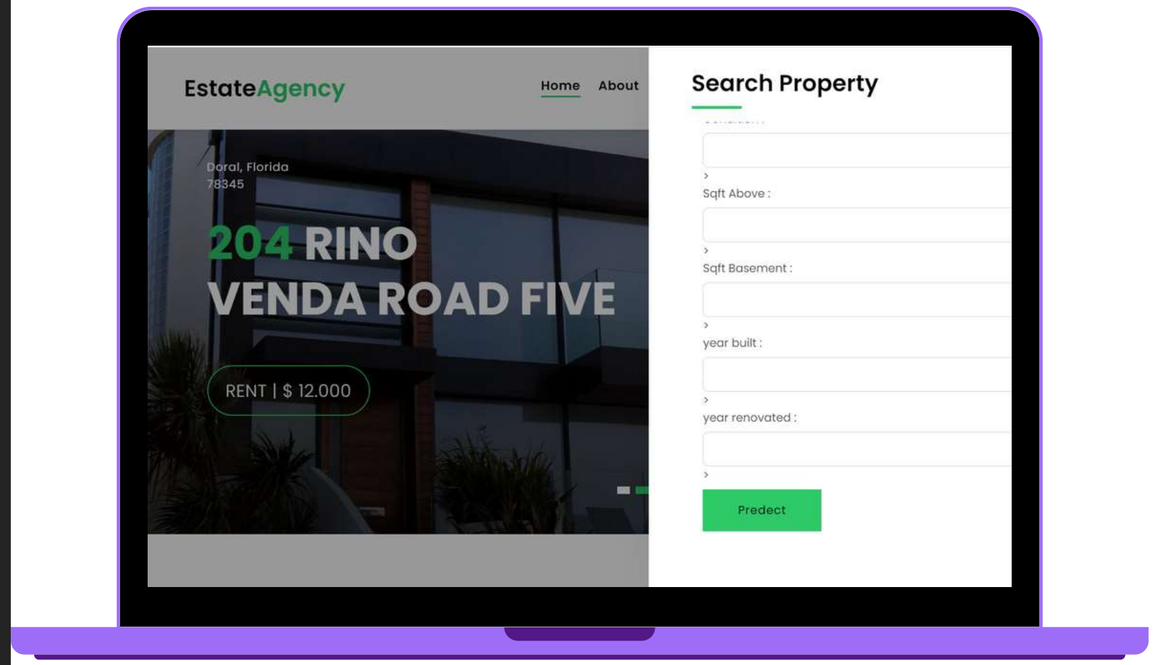
	labels	RSME	R-sqr
0	rf	0.088946	0.713173
1	svr	0.112597	0.540365
2	lr	0.113588	0.532235

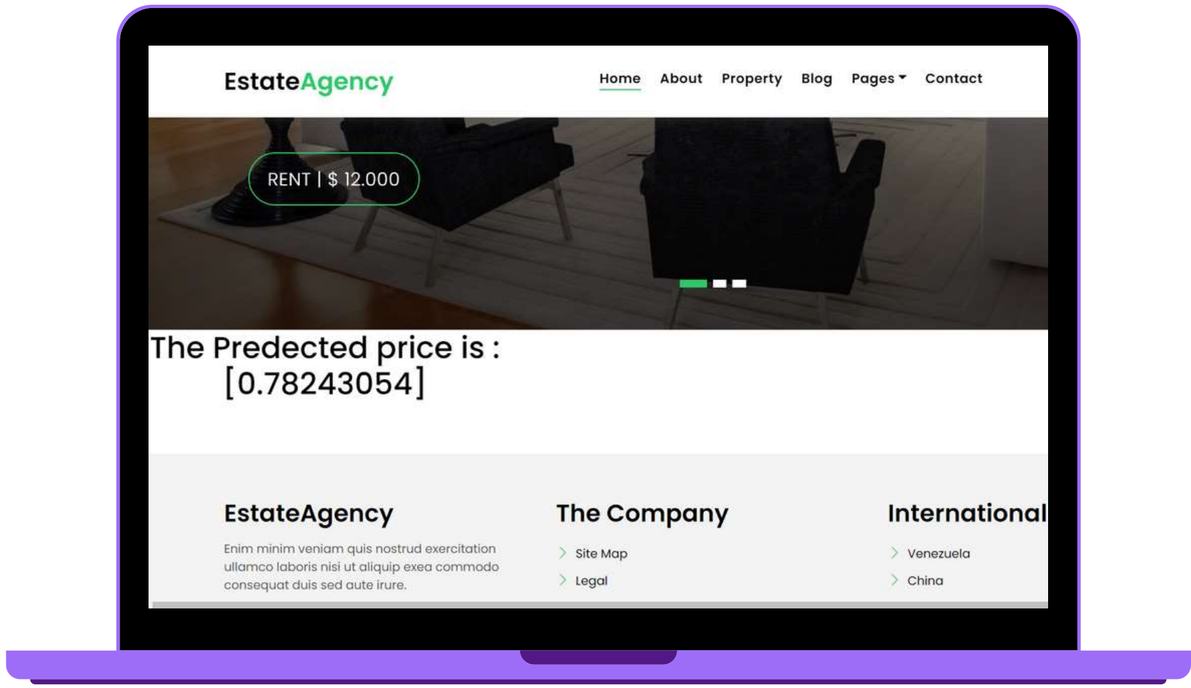
DEPLOYMENT

06

Deployment :

- It is good to implement a powerful machine learning model. But it will not be of much interest if it cannot be used in an "application".
- Indeed, after training a model, it is not quite ready to be used. Additional code must be added so that it can actually be used.
- Django is a high-level Python framework, allowing rapid development of secure and maintainable websites.
- Django has an MVT "Model View Template" architecture, it has several similarities with the Controller View Model architecture.


EstateAgency
Home
About
Doral, Florida
78345
204 RINO
VENDA ROAD FIVE
RENT | $ 12.000
Search Property
Sqft Above :
Sqft Basement :
year built :
year renovated :
Predect


EstateAgency
Home
About
Property
Blog
Pages
Contact
RENT | $ 12.000
The Predected price is :
[0.78243054]
EstateAgency
Enim minim veniam quis nostrud exercitation ullamco laboris nisi ut aliquip exea commodo consequat duis sed aute irure.
The Company
Site Map
Legal
International
Venezuela
China

THANK YOU FOR

YOUR ATTENTION