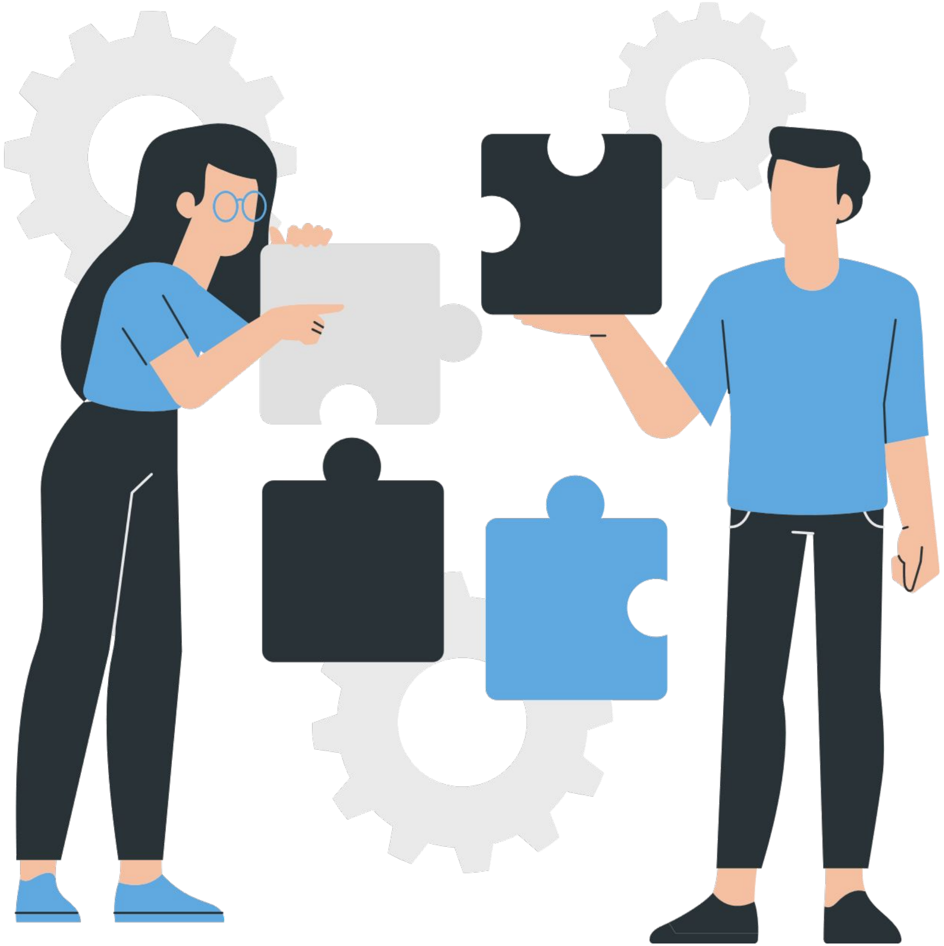
Predicting Airbnb Prices in New York City

Avila Cañibe Aiza
Chen Yaqi
Ela Essola Michele Natacha
He Xingshan
Hong Victor

What’s the problem?

A brief description of what we are doing



What's our goal?

Goal

Given the data of Airbnb accommodations in NY, we aim to find a model that best predicts the price of the accommodation given fixed and engineered features

Methodology

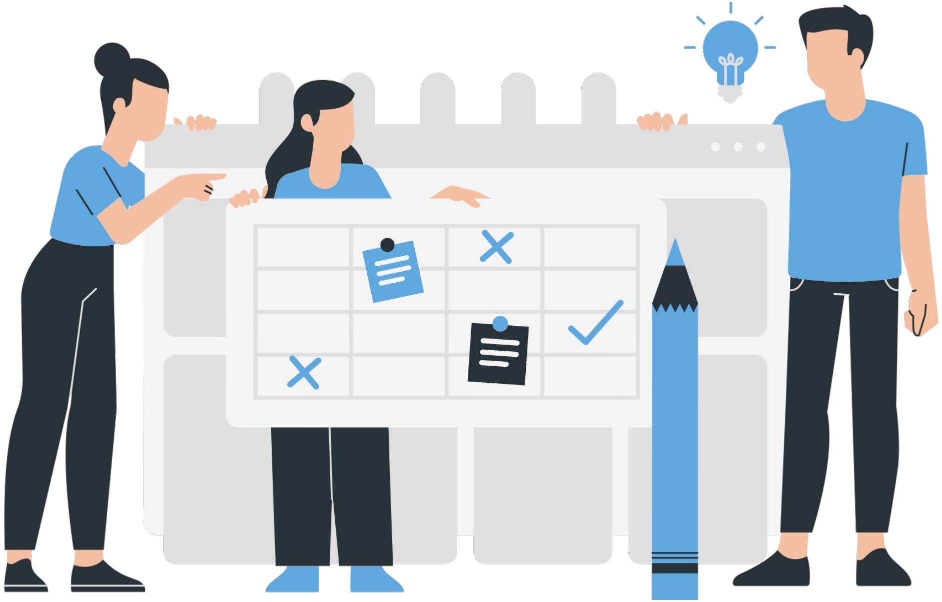
For this purpose we propose to compare the results of the following methods:

- Decision Tree Regressor
- RandomForestRegressor
- ExtraTreesRegressor
- XGB Regressor
- AdaBoost
- CatBoost (Best Accuracy)

As well as modifications and tuning of data, in order to determine the model that best predicts the price of the accommodation

What data do we have?

Summary of the key elements of the data



Exploratory Analysis of the data

Facts

The data contains information collected in 2019 about the various homes available on the Airbnb hosting platform.

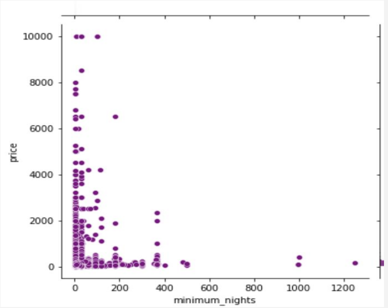
These are the features:

- name (string)
- host_id (key)
- host_name (string)
- neighborhood_group (string)
- neighborhood (string)
- latitude (varchar)
- longitude (varchar)
- room_type (string)
- price (numeric)
- minimum_nights (numeric)
- number_of_reviews (numeric)
- last_review (date)
- reviews_per_month (numeric)
- calculated_host_listings_count (numeric)
- availability_365 (numeric)

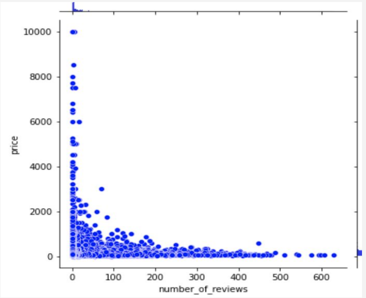
Detected issues

Accommodation prices were heavily right skewed
Variables show correlation with the price feature, such as:

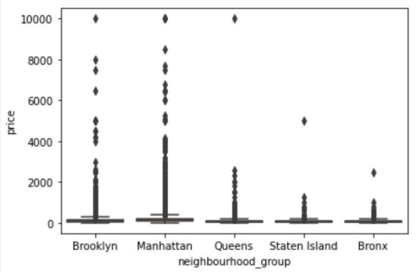
minimum_nights



number_of_review

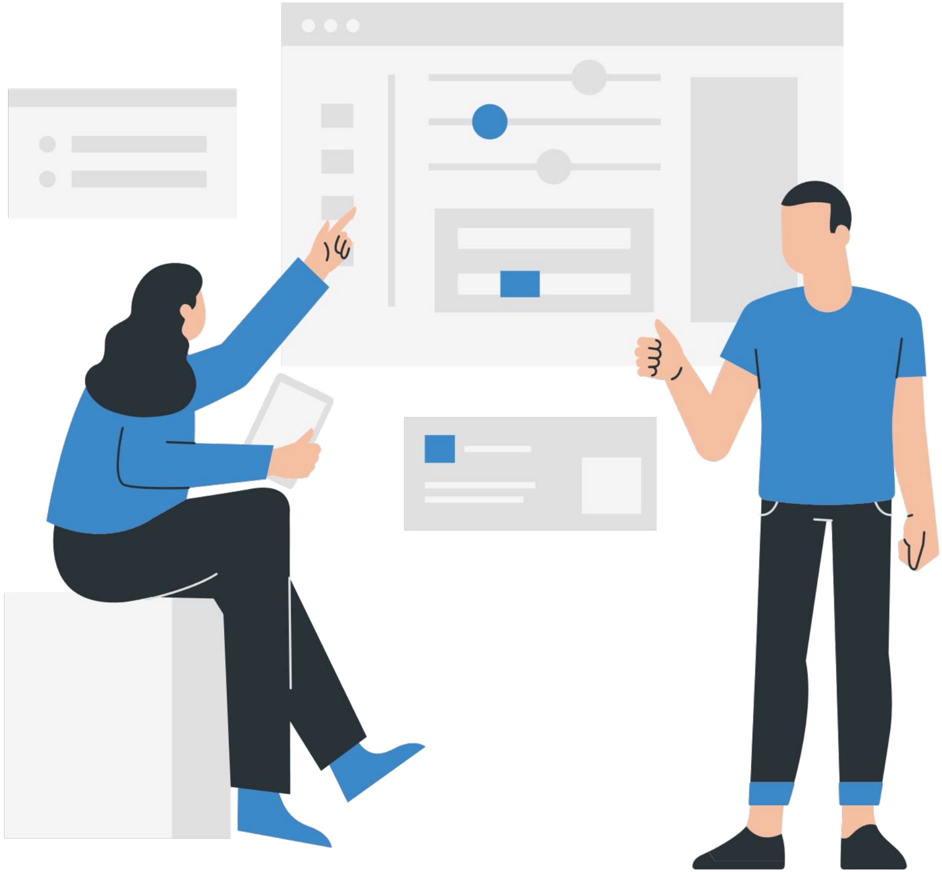


In some neighborhood groups, the prices are more concentrated in the lower end, while in others the prices are more loosely scattered.



Data engineering

Changes applied to the data to get the best fit



Enhancing of the data

Target encoding

Technique that replaces each category with the average target value for that category

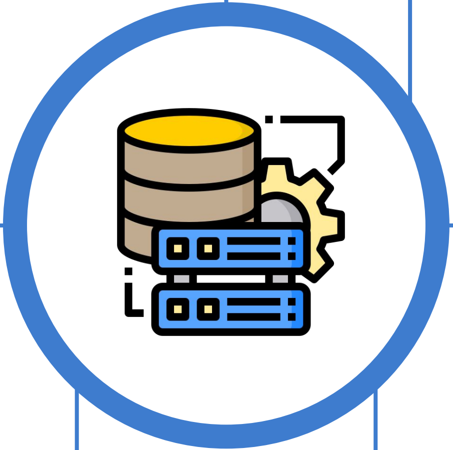
- Group the data by neighborhood_group and neighborhood
- Calculate the mean price for each combination
- Create mapping dictionary that maps each neighborhood_group and neighborhood combination to its mean price
- Replace the neighborhood column with the mean prices

Standard Label Encoding

- neighrborhood_group
- room_type

Eliminated Features

- Id
- host_id
- name



Target variable price

Transformed into log value to minimize accuracy loss, incurred from loose distribution of price range.

Calculation of Recency

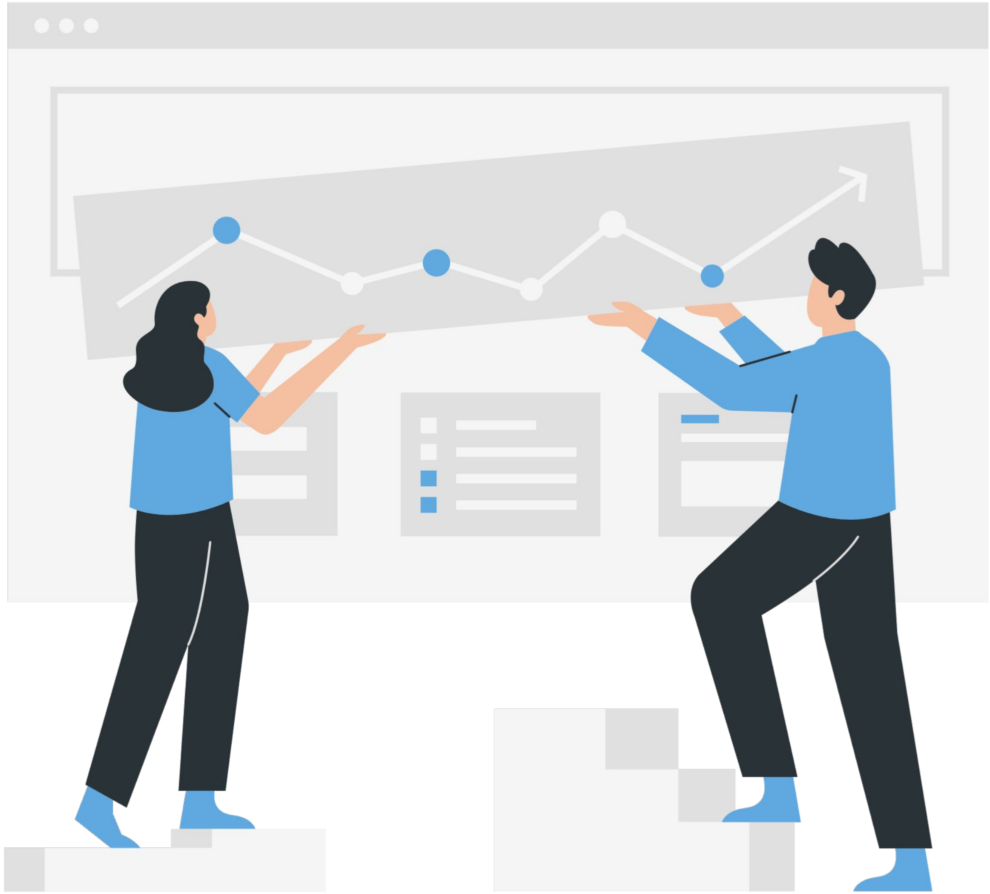
log value that represents the time length difference between the last date, and the last date in the dataset

Standard scaler

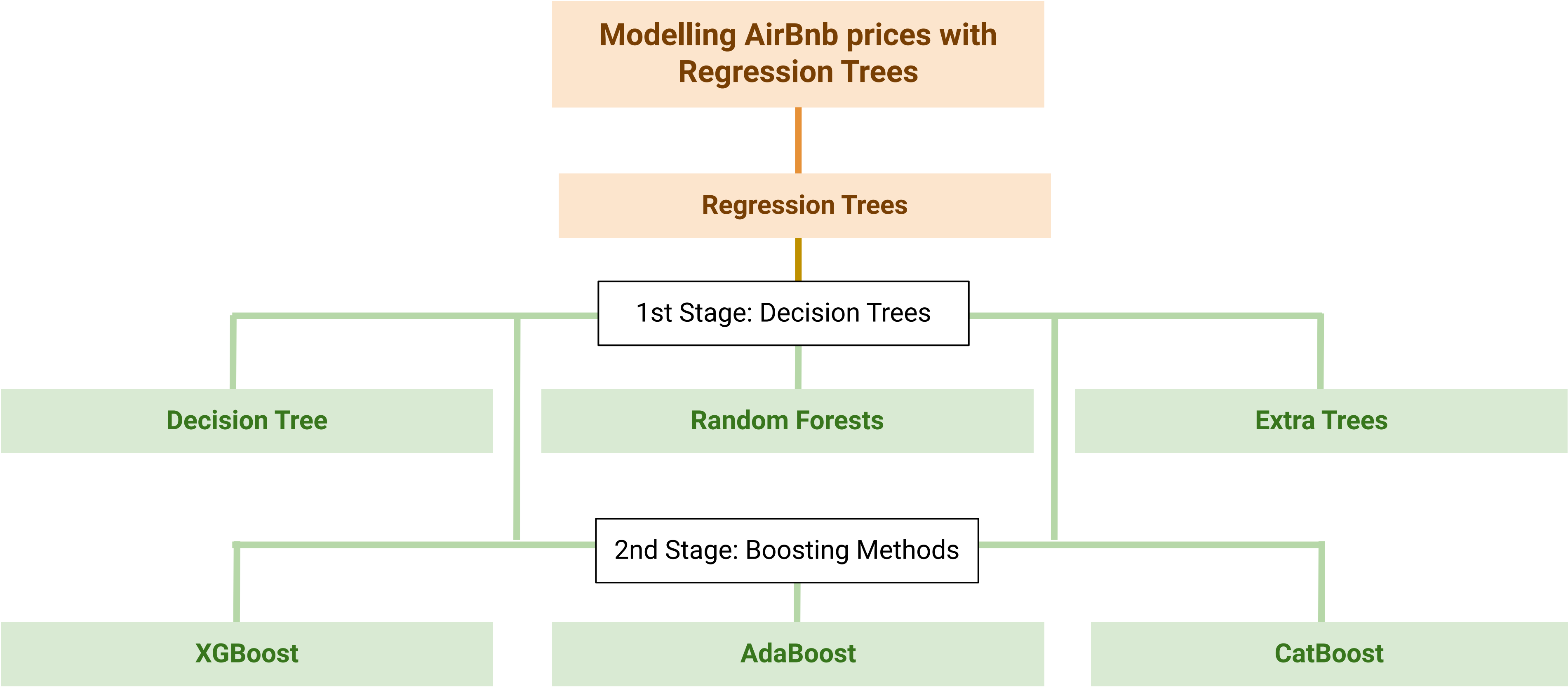
Used to normalize the numerical data

Methodology & results

Summary of the methods done and their results



Methodology & Results (Part 1)



Methodology & Results (Part 2)

Principles Guiding Choice of Models and Hyperparameter Tuning:

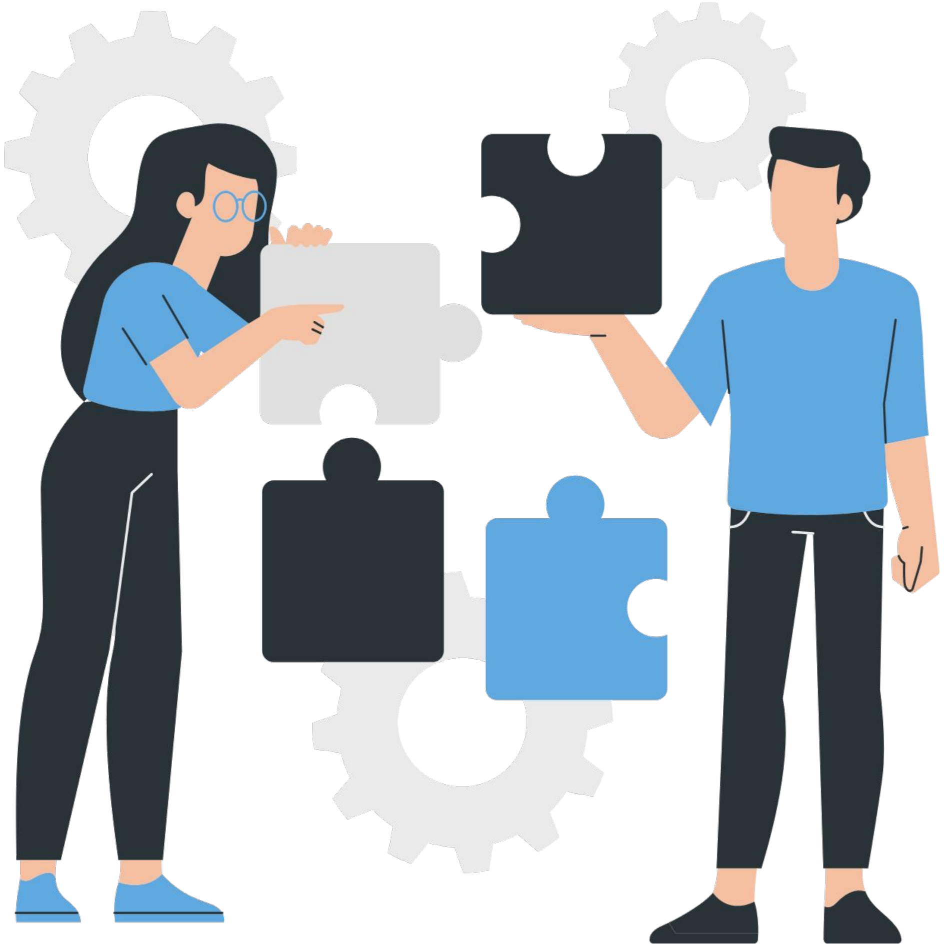
1. **Simplicity of Model:** Start with easy to implement models and observe initial results
2. **Progressive Testing:** Progressively implement more sophisticated models and compare results
3. **Gradual Improvements:** Employ hyperparameter tuning to improve results further (with GridSearch and bagging)
4. **Balancing Trade-offs:** Hyperparameter tuning was decided by improvements in score vs. computational time

No.	Model Choice	Parameters Tuned with GridSearch	Best Parameters	Results (R-squared)
1.	Decision Tree	pgrid = {max_depth: [4,6,8,10], min_samples_leaf: [0.12,0.14,0.16]}	Max_depth = 6 Min_samples = 0.12	0.5202
2.	Random Forests (with bootstrap)	pgrid = {n_estimators: [600,700,800], max_depth: [4,6,8,10,12,14]}	Max_depth = 10, N_estimators = 600	0.6244
3.	Extra Trees (with bootstrap)	pgrid = {max_depth: [4,6,8,10,12,14,16,18,20]}	Max_depth = 18	0.6268
4.	XGBoost	pgrid = {max_depth: [4,6,8,10,12,14,16,18,20]}	Max_depth = 6	0.6218
5.	AdaBoost	pgrid = {n_estimators: [300,400,500,600,700]}	N_estimators = 300	0.2383
6.	CatBoost	pgrid = {max_depth: [4,6,8,10]}		0.6327

Conclusions

1. Performance between decision tree, bagging and boosting methods did not differ significantly in terms of predictive accuracy
2. However, boosting methods required larger computational resources
3. Given the relative similarity in results, we conclude that we need to revisit the data (skewness was observed) to explore further feature engineering to improve the performance of predictions

Scratch Trees



Scratch trees

For this section we have 2 proposals available on Github

Proposal 1

Classification Tree (Entropy)

Algorithm:

```
Initialize best entropy, best dimension, best value
For columns {
        For rows{
                Calculate mean between pairs of
                observations
                Based on mean split data into left and
                right branches
                Calculate entropy
                Define node
                If entropy<0.000000001: return node
                Else recursive tree for the right and left
                splits
        }

}
```

Proposal 2

Classification Tree (Entropy)

Algorithm:

```
Define a class for classification{
        if the maximum depth of the tree hasn't been
        reached nor the minimum amount of elements in
        the sample continue
        for features in data{
                select the unique values of the feature
                select a split
                calculate information gain using entropy
                per split
        }
        select best possible split and divide data
        recursively by calling create tree

}
```

Scratch trees

For this section we have 2 proposals available on Github

Proposal 1

Regression Tree

Algorithm:

```
Define regression tree as class

for maximum levels of the tree update the tree by best
partitions until the node is a leaf {
      For parameters in data{
            split data in half
            For each half{
                  Determine the indexes for best
                  partition by squared error
                  if reached min squared error settle
                  the partition
            }


      }
stop growth if min samples in the partition are met
}
```

Proposal 2

Classification Tree (Entropy)

Algorithm:

```
Define a class for regression{
      if the maximum depth of the tree hasn't been
      reached nor the minimum amount of elements in
      the sample continue
      for features in data{
            select the unique values of the feature
            select a split
            calculate information gain using variance
            reduction
      }
      select best possible split and divide data
      recursively by calling create tree

}
```