# Predicting the sale price.

**Are you buying or selling a house? What price should you set ? You probably have potential benefits to set your price higher related to specific properties that you do not know.  Which property is more or less expensive?All of these correlations are good to observe which is what I am going to do in this project.Whatever it takes: far from a highway, far from a mall? What does really impact on a price or what is optimal price, or is there an outlier? A lot of questions and considerations. I will train a model to predict sale prices having data as a knowledge base  and known algorithms.**

**Clients here can be anybody. Whoever sells or buys a house. Banks and loan providers. Businesses and individuals.  All of them are my potential clients.They  would care about this since the market changes overtime. Anytime would be helpful to predict the best price. Why not request the use of computers and Artificial Intelligence? It requires data analysis first from the collected data, then stating a hypothesis , testing it. It requires a touch of a data scientist's job.**

## Data Wrangling

### Overview

This section describes the various data cleaning and data wrangling methods applied to the “house-prices-advanced-regression-techniques” data.

### Summary Files

This dataset is split into two separate files:train and test sets.
I took a train.csv file since I have a SalePrice target variable in there to investigate dependencies and this file is basically used to train a ML model.

Original version of the dataset
https://www.kaggle.com/c/house-prices-advanced-regression-techniques.

### Duplicates and Missing Values

For the analysis it was necessary to check for duplicates.
The dataset also had a lot of missing values due to the not being available in the Database, excluding categorical values.The loss is about 27%,  this is a significant loss of data. The difficulty  of handling missing data  in my dataset is that I do not have the same datatype for all the data.
There are 6 ways to handle missing values in a dataset:

- 1- Do Nothing:However, other algorithms will panic and throw an error complaining about the missing values (ie. Scikit learn — LinearRegression). In that case, you will need to handle the missing data and clean it before feeding it to the algorithm.

- 2- Imputation Using (Mean/Median) Values:

## Pros:

- Easy and fast.
- Works well with small numerical datasets.

**Cons**:

- Doesn’t factor the correlations between features. It only works on the column level.
- Will give poor results on encoded categorical features (do NOT use it on categorical features).
- Not very accurate.
- Doesn’t account for the uncertainty in the imputations.

## • 3- Imputation Using (Most Frequent) or (Zero/Constant) Values:

**Pros:**

- Works well with categorical features.

**Cons:**

- It also doesn’t factor the correlations between features.
- It can introduce bias in the data.

## • 4- Imputation Using k-NN:

**Pros:**

- Can be much more accurate than the mean, median or most frequent imputation methods (It depends on the dataset).

**Cons:**

- Computationally expensive. KNN works by storing the whole training dataset in memory.
- K-NN is quite sensitive to outliers in the data (**unlike SVM**)

## 5- Imputation Using Multivariate Imputation by Chained Equation (MICE)

## 6- Imputation Using Deep Learning (Datawig):

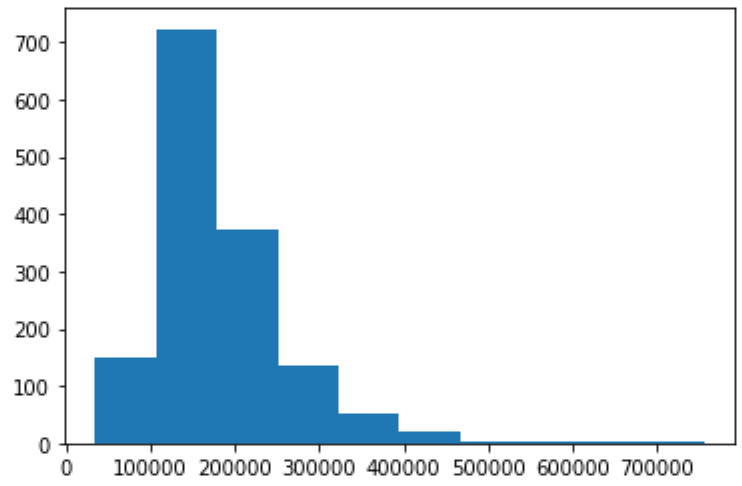
I applied only pandas methods in python . I have not touched scikit learn yet, this is more advanced, but probably can do it.
I  replaced missing data with the most common values for object data type values, but  columns that have 90% missing I dropped since It can introduce bias in the data. For int or float data type values I  replaced missing values with the median since the median will be the same for any distribution, still not very accurate for training a prediction model.
Also, tested melting, pivoting, unstacking, and more techniques to sharpen my skills and keeping the original.
For easy navigation through my dataset I performed sorting of column names in an alphabetical order.

**Outliers and handling them**

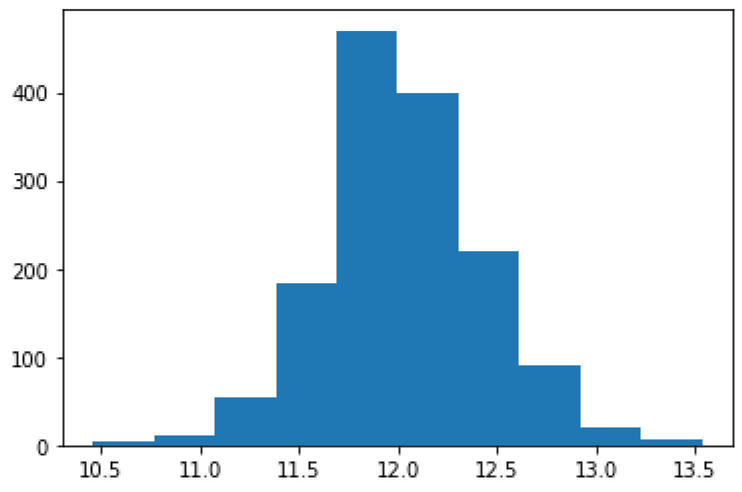
After making a plot of SalePrices  it is easy to see outliers.(unusual pattern of data that is really different or far from normal) But are those really outliers?To decide outliers or not, I performed an IQR test for Sale Prices. So this test gives me an outcome of where my outliers are and knowing that I can remove this data so that would not create a problem for training a prediction model.
This is my approach at this point.

## Exploratory data analysis

For EDA I started to explore how my SalePrice distribution looks visually  and testing it for normality.



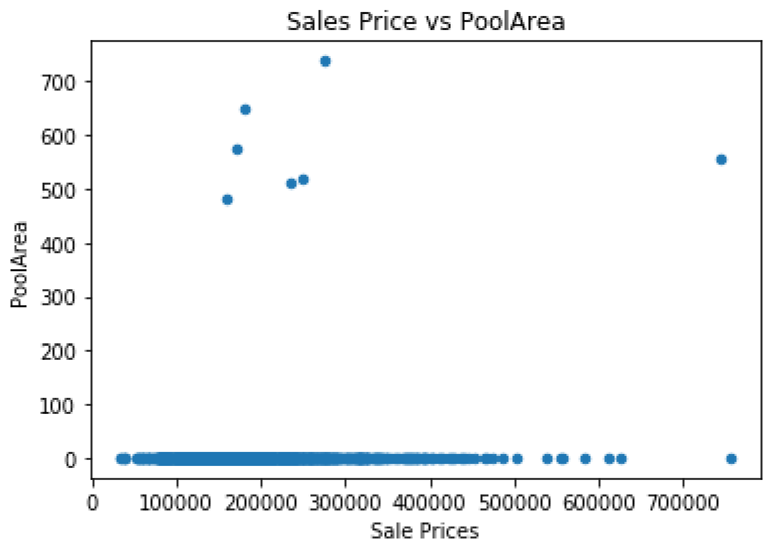
Also, I changed this distribution using a log-transformation to make it as close to normal as possible. This step is reserved for training a ML model for later.



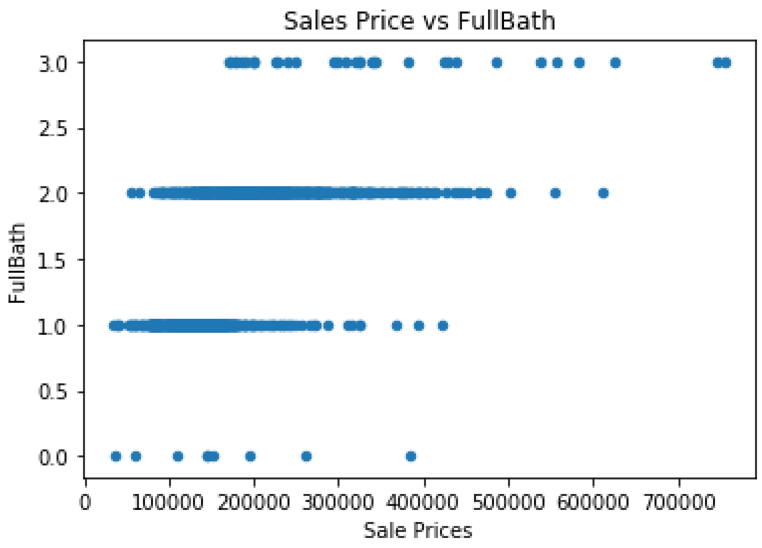
Original plot tells us trends of SalePrices over the number of houses. So data tend to radically positively grow in numbers and to decay as prices go higher. To fit our model , not to overfit it, would be good to normalize our data. For this purpose we do a log-transformation. Now our data looks almost a normal distribution and it gives us a good distribution of our data to fit a trained model.

Next, I compared two related quantities or more. I visualized it to have an idea of how correlated the data and what else interesting or facts to notice. I made  several scatterplots  and it is a really helpful tool  that tells a lot about the data.

Only a few houses have Pool Area but almost all of them do not have so I should drop and remove this column to avoid a noise creation in my dataset.



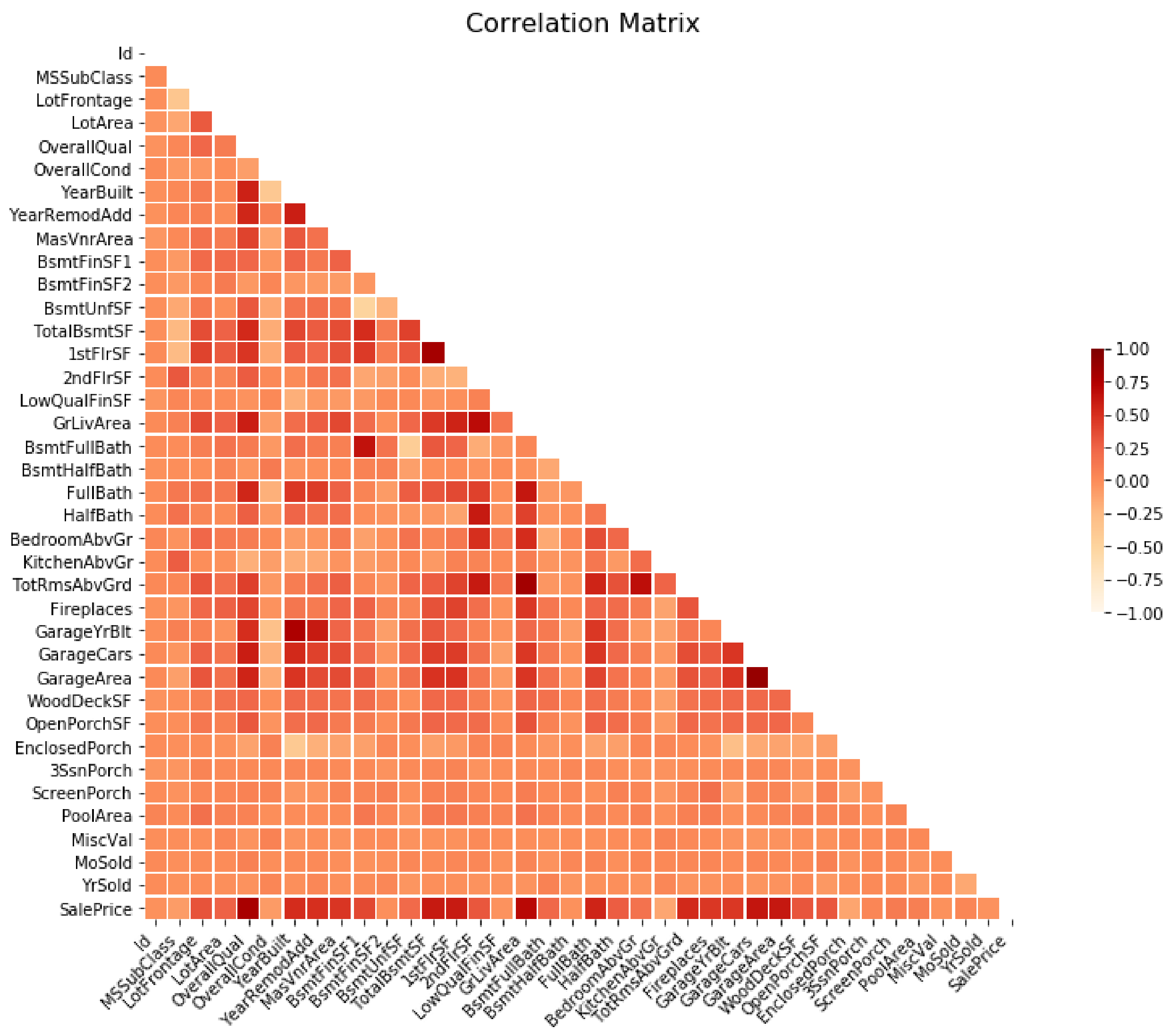
Obviously, Houses that have 3 full baths are more expensive.



But that is not all , there is a lot more to research.

Moreover, in addition to scatterplots, I made a correlation matrix, a powerful exploratory tool, that tells me so much about correlation of features in the dataset. I conveyed what SalePrice is correlated to visually using a heatmap.

Let's look at the correlation matrix and what it will convey.

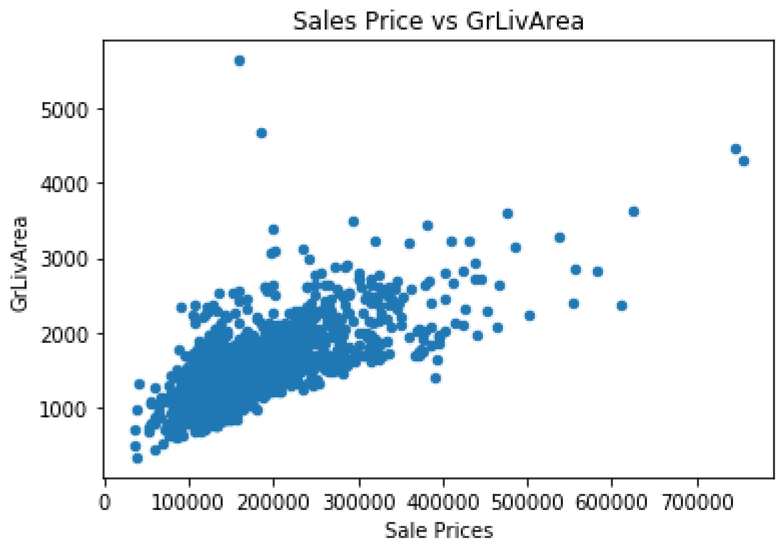


SalePrice (target) is strongly correlated to OverallQual,

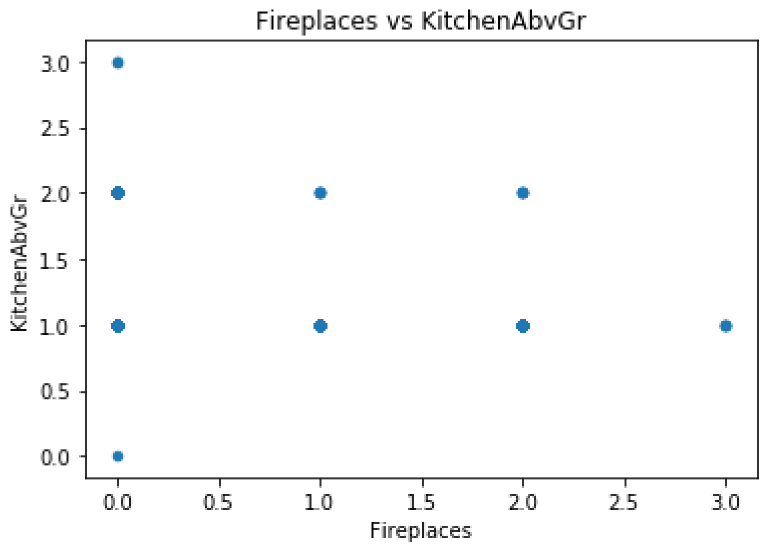
YearBuilt,YearRemodAdd,MasVnrArea,BsmntFinSF1,TotalBSmntSF, 1stFlrSF, GrLivArea,

FullBath,GarageCars,GarageArea.

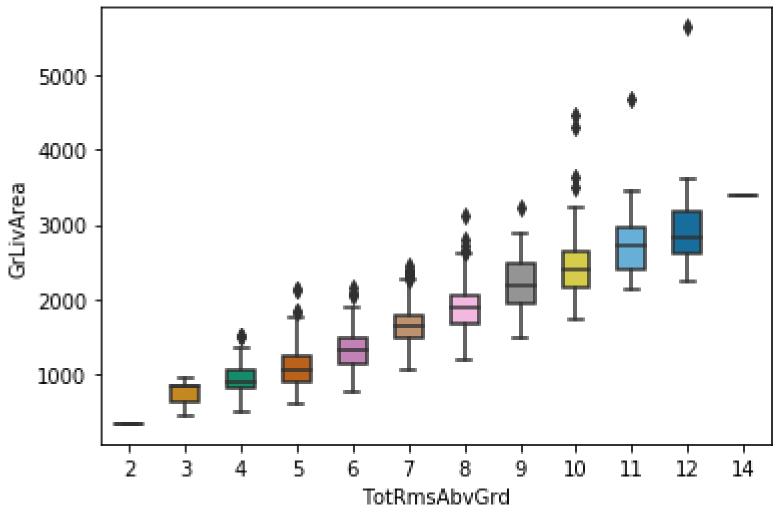
This is how plots look if there is a strong correlation.



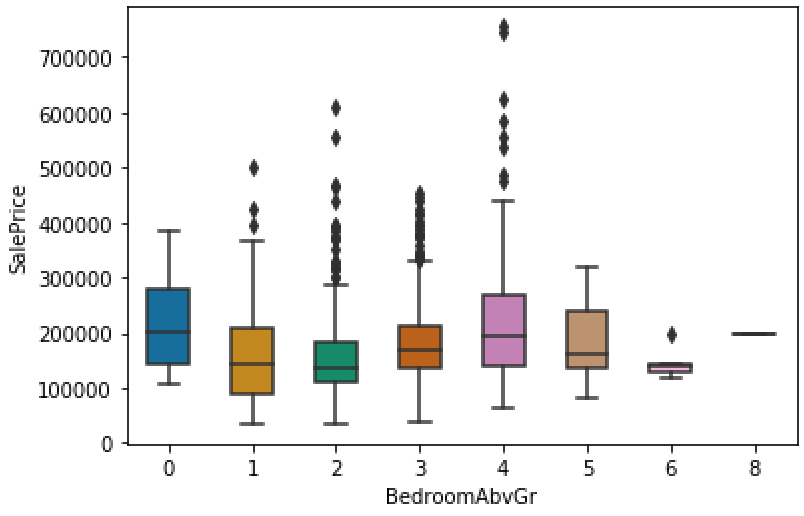
According to the Correlation Matrix I have data that are not correlated and if to visualize them plots will tell that.



According to the Correlation Matrix I have data that are strongly correlated between ech other and if to visualize them plots will tell that.



But I want to investigate further if SalePrice means 2 and 3 BedroomAbvGr have a statistical significance. In other words, I want to investigate if there is no difference in their SalePrices.



## Testing a hypothesis

Finally, I have a hypothesis that houses and their SalePrice means for those that have 2 and 3 bedrooms above grade only do not have a difference.

To test my hypothesis I need to perform a ttest of 2 groups. To perform the ttest I have to compute standard deviations of tested groups.

I computed a standard deviation and mean for houses that have 2 Bedrooms only.

I computed a standard deviation and mean for houses that have 3 Bedrooms only.

So knowing that information I performed a t-test, trying to find evidence of a significant difference between population means (2-sample t). Put another way, T is simply the calculated difference represented in units of standard error. The greater the magnitude of T, the greater the evidence against the null hypothesis. This means there is greater evidence that there is a significant difference. The closer T is to 0, the more likely there isn't a significant difference.

My ttest computed that 1.5299459699837254e-06 is smaller than .05, so we're going to reject the null hypothesis which asserts there is no difference between our sample mean and the population mean. For this two-tail test, we reject the Null and we conclude that there is a statistically significant difference .

# Training a model

## Feature Engineering

In this section I had to prepare all the data to make it readable and standardized for computations. I had a lot of categorical data in this dataset which required me to get it converted in numbers. Then , I simply merge all as a one numerical dataframe and it is ready for computations.

## Training a model

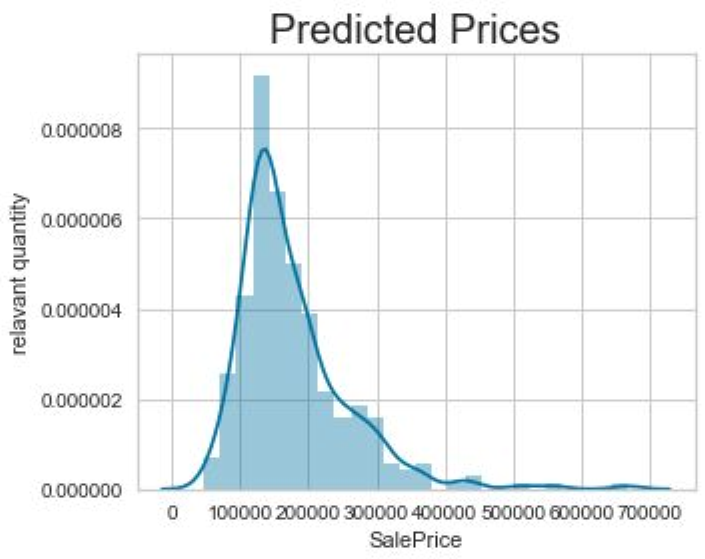
There are many algorithms invented to train a predictive model . This will take a lot of time to test them all. Only one model can take sometimes hours for training only. That's why I tried only three, the most common.:Ridge, Random Forest, XGBoost. Ridge is the extension of a linear regression but more advanced and less sensitive to new data. Random Forest and XGBoost are based on decision trees and feature importance classification.

## Findings and conclusions

| Model Algorithm | RMSE scores mean |
|---|---|
| Ridge: | 0.143 |
| Ridge after removing not correlated features | 0.143 |
| Pipeline Ridge | 0.141 |
| Random Forest | 0.144 |
| Random Forest after removing NOT correlated features | 0.144 |
| XGBoost | 0.130 |
| XGBoost after removing NOT correlated features | 0.130 |

Obviously, XGBoost performed the best score. Though it is the slowest model in my list. XGBoost algorithm here  makes predictions with the lowest  error rate. Of course , I want to use this algorithm to obtain my predicted values. Interestingly,  even after removing NOT correlated features all of the algorithms  did not have  changes in  error rate. I explain it because  algorithms  parameters  are smart calculations enough  to not  pay attention to outliers and not useful features. But  this also due to good preproccesing .

## Product of a model predictions


Predicted Prices
relavant quantity
SalePrice