# 

# HOUSE:PRICE PREDICTION

Submitted by:

Jyoti Trivedi

**ACKNOWLEDGMENT**

I would like to express my gratitude towards

Ms. Sapna Verma

for guiding me throughout the project.

I also feel thankful and express my kind gratitude towards

Flip Robo Technologies

for allowing me to conduct House Price Prediction project.

1. **Introduction**

**1.1 Context**

This project has been done as part of my internship for the FlipRobo Technologies. Supervised by Ms. Sapna Verma, I had two weeks to fulfill the requirements in order to succeed the module.

**1.2 Motivations**

Being extremely interested in everything having a relation with the Machine Learning, the independent project was a great occasion to give me the time to learn and confirm my interest for this field. The fact that we can make estimations, predictions and give the ability for machines to learn by themselves is both powerful and limitless in term of application possibilities. We can use Machine Learning in Finance, Medicine, almost everywhere. That’swhy I decided to conduct my project around the Machine Learning.

**1.3 Idea**

As a first experience, I wanted to make my project as much didactic as possible by approaching every different steps of the machine learning process and trying to understand them deeply. I chose to take House Price Prediction as approach. I was required to model the price of houses with the available independent variables. This model will then be used by the management to understand how exactly the prices vary with the variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns. Further, the model will be a good way for the management to understand the pricing dynamics of a new market.

**1.4 Sources**

Because I truly think that sharing sources and knowledge allow to help others but

also ourselves, the sources of the project are available at the following link:

<https://github.com/jyotitrivedi/flip_robo_internship>

Feel free to give me your point of view or ideas for anything you want.

**2. The Project**

**2.1 Data**

The crucial element in machine learning task for which a particular attention should be clearly taken is the data. Indeed the results will be highly influenced by the data based on where did we find them, how are they formatted, are they consistent, is there any outlier and so on. At this step, many questions should be answered in order to guarantee that the learning algorithm will be efficient and accurate.

Many sub steps are taken to get, clean and transform the data. I am going to explain each one of them to show how they have been applied on my project why they are useful for the machine learning part.

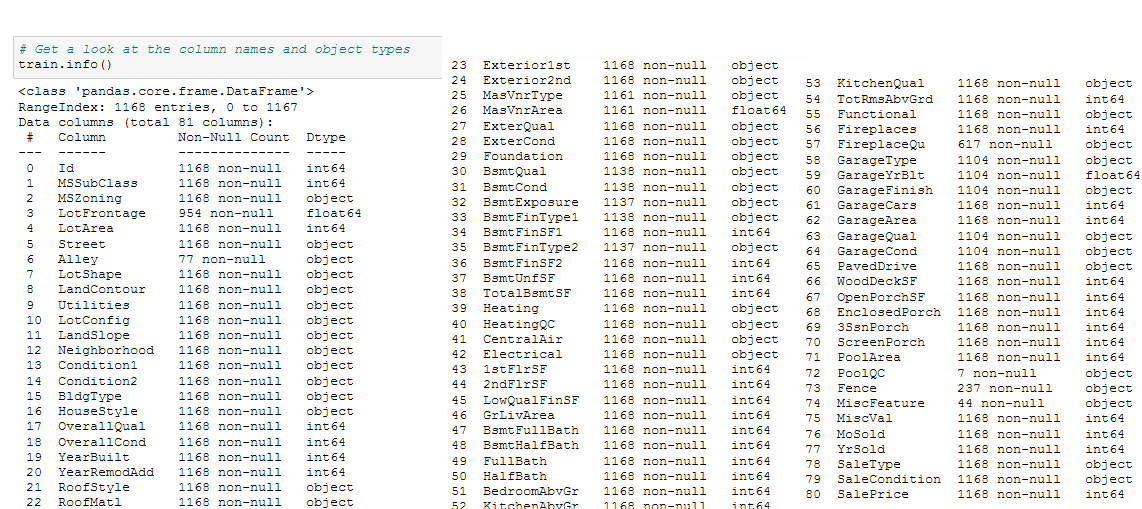
2.1.1 Getting the data

I had a csv file containing all the details about the dataset and a excel file containing description of all the features. These details are provided by the company.

2.1.2 Attribute Types

After having collected the data, an important step is the data preprocessing. What is done during this phase will influence directly the result of the machine learning algorithm. I was firstly interested by what are my attribute types, how can I describe my data with basic statistical measures to get first insights, how my attributes are similar to each other and what kind of techniques can I use to visualize my data.

The figure shows the attributes with their respective types:



This gave me also the information that the label, that is the variable that I want to

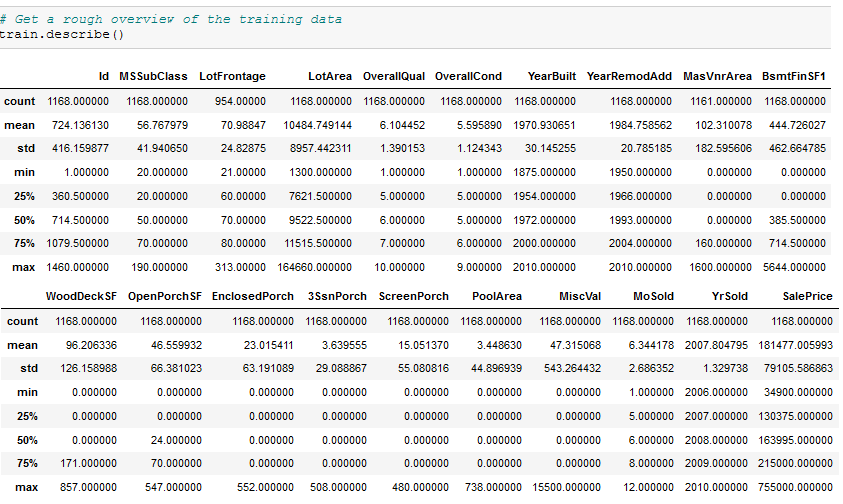
predict, is continuous meaning I have a **regression problem** and not a classification problem

that would be the case if label was discrete.

2.1.3 Basic Statistical Measures

With the type of my attributes in mind, I used basic statistical measures on my data such as mean , mode, median, standard deviation etc.

The results of the measures are shown in the figure:



With the following result I was able to notify some interesting points such as :

● There is no missing value..

● The Q1, Q2, Q3 (25%, 50%, 100%) are good indicators of the shape of the different

attribute. The Q2 being the median that is not influenced by outliers unlike the mean.

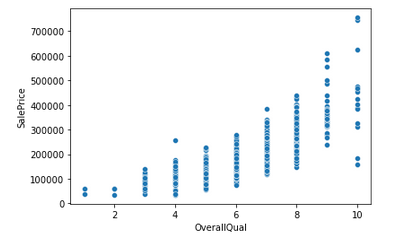
● According to the max values, there are obviously outliers

● The standard deviation gives also an indication about what should be considered as outliers, but it is not a robust technique since the standard deviation use the mean to

be computed.

2.1.4 Visualizing The Data

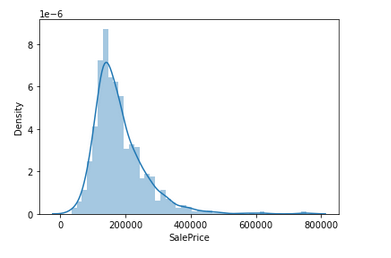
Next, I tried to figure out what kind of chart I could use and it turns out that each one has its own advantage depending on what we want to visualize. Let’s start with scatter plot. Scatter plots are used to observe relationship between variables and uses dots to represent the relationship between them.The Matplotlib module has a method for drawing scatter plots, it needs two arrays of the same length, one for the values of the x-axis, and one for the values of the y-axis. The scatter plot gave me a better intuition of how my variables were distributed.



We can see the significant Linear Correlation between the the target variable and OverallQaual

Another approach that I used to see more precisely the distribution of my continuous

variables was by using density plot. Unlike histogram that use bins, density plot use a curve to represent the distribution more accurately.

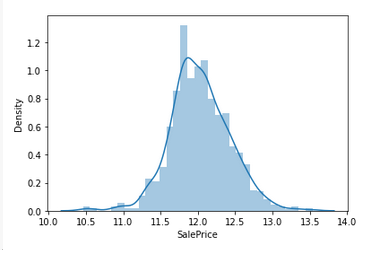


The following cells explore the distribution of the Target Variable and it can be seen that it is left-skewed and has Outliers (as seen from the Percentiles and the critical values). Some ways to deal with such distributions are to:

* Perform a logarithmic transformation on the values (Normalization and Standardization do not alter the skewness, just the scale)
* Explore the Percentiles to come up with some upper/lower thresholds beyond which the values are set to something more common (Boxplots also help here)

We use a Log Transform here: it is simple and the resulting distribution is visually very similar to a Gaussian.

So After applying log transformation, we can see now our target variable is normally distributed.



**2.1.5 Correlation Analysis And Redundancies**

After seeing the distribution of my variables, I computed the correlation between the

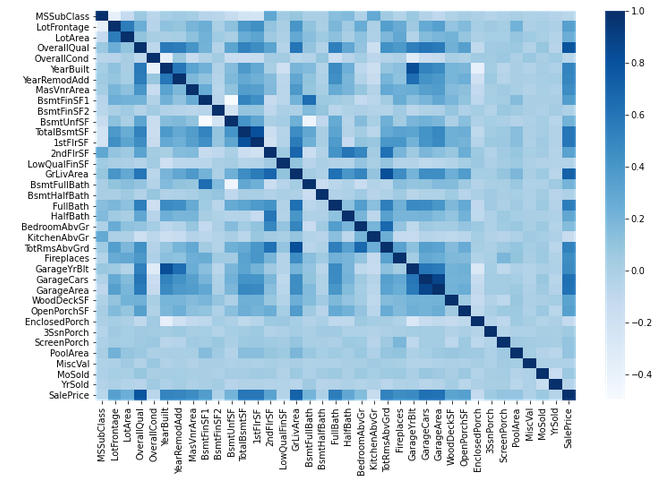
different attributes and how each one is correlated to the label

A better way to see that is graphically using a heat map as shown below: The diagonal is obviously equal to 1, since it represents the correlation between the

same attributes.

Exploring the Linear Correlation between Features is helpful since it can:

* Highlight which features look to be the most useful for predicting the target variable
* Shed some light on which features are highly correlated together in which case they could be mutually redundant (this helps in Feature Selection)



We find that the two features OverallQual and GrLivArea are the most linearly correlated with our target so we can choose to explore them a bit further.

2.1.6 Feature Engineering and Feature Selection

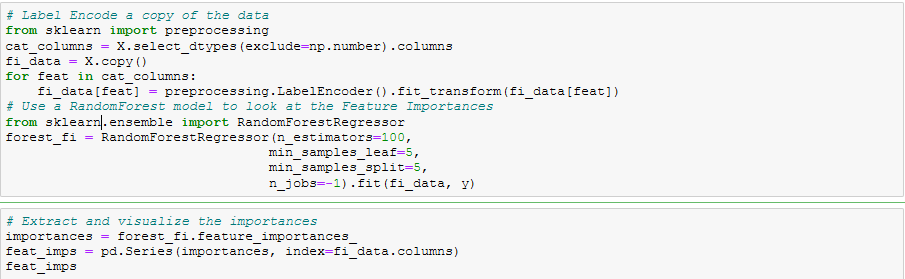
In Data Science, the performance of the model is depending on data preprocessing and data handling. Suppose if we build a model without Handling data, we got an accuracy of around 70%. By applying the Feature engineering on the same model there is a chance to increase the performance from 70% to more.

Simply, by using Feature Engineering we improve the performance of the model.

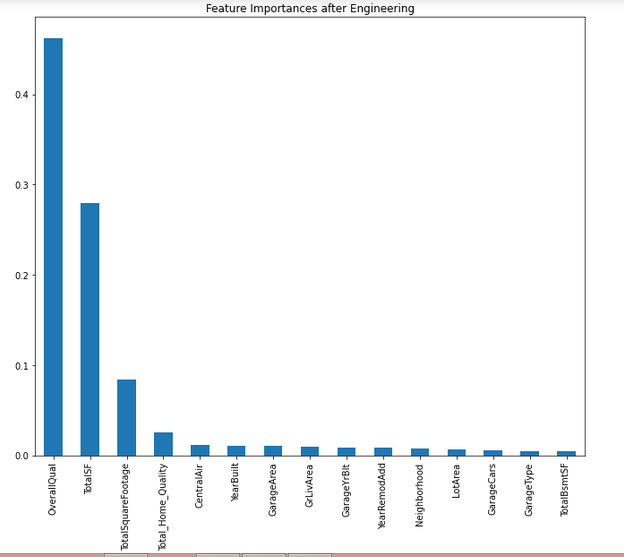
Feature selection is nothing but a selection of required independent features. Selecting the important independent features which have more relation with the dependent feature will help to build a good model.

Some aspects of Feature Engineering include:

Create Aggregated Features (can result in more robust models, as will be seen later) Dealing with heavily Skewed features (either dropping them, transforming them or doing nothing) Scaling/Normalizing numeric variables (really only required for non-Tree-based algorithms like SVMs, Linear Regression, MLP etc.) Encodings for Categorical Variables (the only thing that's necessary to carry out here)







Three of our aggregated features are among the best that can be used for predicting the Target Variable.

The OverallQual feature still reigns the highest and by a significant margin at that. Other than that we can also see the YearBuilt feature among the top predictors which implies that the later a house was built, the price tended to increase a lot more.

Another interesting note is that there are no Basement related features in this top-15 list, but there are a number of Garage related features. This could imply that in practice, Garages are more valuable than Basements which makes sense.

**With the data processing done, the next part is the Machine Learning.**

**2.2 Machine Learning**

The Machine Learning part is about trying to find the best learning algorithm for a

given problem even if it is highly conditioned by how well the data has been processed and

tune some parameters to improve it. Depending on the problem, if it is supervised (meaning

we build a model from labeled training set, the value of the dependent variable is known) or if it unsupervised (the model is built on unstructured and unlabeled data), if it is a regression or classification problem, many learning algorithm exist each with their benefits and drawbacks.

2.2.1 Sampling

Given my dataset, I applied a sampling technique in order to divide it into different

subset having each its own utility. It is commonly assumed that more we have data to build a model more it will have tend to give good results. Usually the dataset is divided as follow with their respective utility:

Used to validate the model

Used to build the model

The training set as its name indicates it is used to train the learning algorithm. The

test set is used to validate the model and make optimizations. Indeed, because

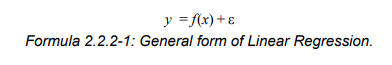
the model is built from the training set, we can’t check it on the same set because the result

would be overly optimistic. The test set is used **only** to see how well the learning algorithm is generalized, meaning how it performs with unknown data.

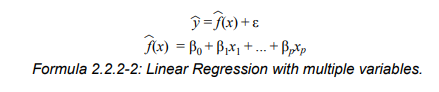
2.2.2 Learning Algorithm

1. Linear Regression

Among the large choice of learning algorithm I chose to use Linear Regression because my dependent variable is continuous. The goal was to model a relationship between y , the dependent variable and x , .. , x , the independent variable where is the number of 1 . p p them. The general form of this is given by the following equation :

. 

Since I had several independent variable (features) to predict the dependent variable (Price),I was in the case of multiple linear regression (the ^ just means estimation), given by:

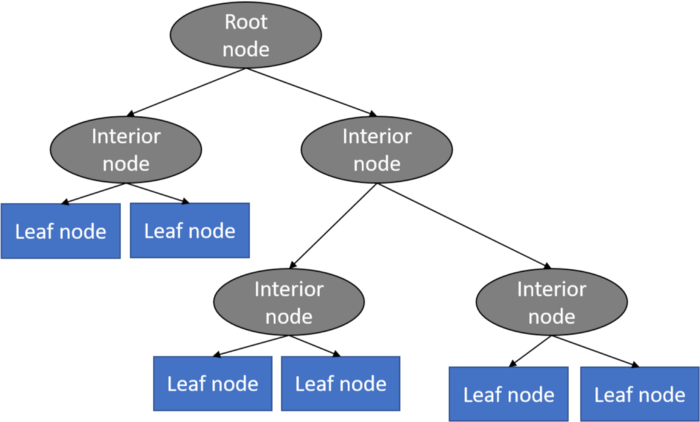


This learning method is parametric because I assumed that the shape of my model (function) was linear that have simplified greatly the problem because I only needed to estimate a set of parameters. Nonetheless, the disadvantage of using a parametric method is that the model chosen will usually not match the true unknown form of f (function connecting the input variable x to the output variable y ). This problem can be addressed by turning the model into something more flexible that can fit many different functional forms for f but can lean to other problems such as overfitting because we need to estimate more parameters. A precision about the error term. Actually it exists two kind of error, the reducible error also known as residual term (distance between the estimated regression line and the data points), and the irreducible error also known as the error term (distance between the true regression line and the data points). The reducible error is what I tried to minimize as much as possible and it is actually the goal of the optimization algorithm used to find the unknown parameters

2. Decision Tree Regression

Decision Tree is one of the most commonly used, practical approaches for supervised learning. It can be used to solve both Regression and Classification tasks with the latter being put more into practical application.

It is a tree-structured classifier with three types of nodes. The **Root Node** is the initial node which represents the entire sample and may get split further into further nodes. The **Interior Nodes** represent the features of a data set and the branches represent the decision rules. Finally, the **Leaf Nodes** represent the outcome. This algorithm is very useful for solving decision-related problems.



With a particular data point, it is run completely through the entirely tree by answering True/False questions till it reaches the leaf node. The final prediction is the average of the value of the dependent variable in that particular leaf node. Through multiple iterations, the Tree is able to predict a proper value for the data point.

.

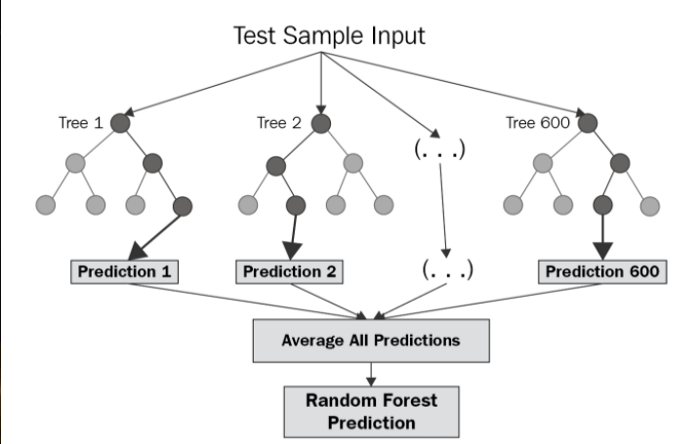
3. K-Nearest Neighbors Regression

KNN regression is a non-parametric method that, in an intuitive manner, approximates the association between independent variables and the continuous outcome by averaging the observations in the same neighborhood. The size of the neighborhood needs to be set by the analyst or can be chosen using cross-validation (we will see this later) to select the size that minimizes the mean-squared error.

While the method is quite appealing, it quickly becomes impractical when the dimension increases, i.e., when there are many independent variables.

4. Random Forest Regression

**Random Forest Regression** is a supervised learning algorithm that uses **ensemble learning** method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.



The diagram above shows the structure of a Random Forest. You can notice that the trees run in parallel with no interaction amongst them. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees. To get a better understanding of the Random Forest algorithm, let’s walk through the steps:

1. Pick at random *k* data points from the training set.
2. Build a decision tree associated to these *k* data points.
3. Choose the number *N* of trees you want to build and repeat steps 1 and 2.
4. For a new data point, make each one of your *N*-tree trees predict the value of *y* for the data point in question and assign the new data point to the average across all of the predicted *y* values.

A Random Forest Regression model is powerful and accurate. It usually performs great on many problems, including features with non-linear relationships. Disadvantages, however, include the following: there is no interpretability, overfitting may easily occur, we must choose the number of trees to include in the model.

2.3 Results

In this section, we present analytic results of the various machine learning models adopted in this paper. All model diagnostic metrics in this paper are based on the validation/test set.

2.3.1 R2 Score

R-squared (R2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a [regression](https://www.investopedia.com/terms/r/regression.asp) model. Whereas correlation explains the strength of the relationship between an independent and dependent variable, R-squared explains to what extent the variance of one variable explains the variance of the second variable. So, if the R2 of a model is 0.50, then approximately half of the observed variation can be explained by the model's inputs.

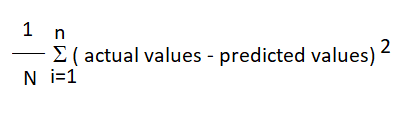
.

|  |  |
| --- | --- |
| **Model** | **R2 Score(%)** |
| Linear Regression | 90% |
| Random Forest Regression | 89% |
| KNN Regression | 80% |
| Decision Tree Regression | 81% |

2.3.2. Mean Squared Error or MSE

MSE is calculated by taking the average of the square of the difference between the original and predicted values of the data.

Hence, MSE =



Here N is the total number of observations/rows in the dataset. The sigma symbol denotes that the difference between actual and predicted values taken on every i value ranging from **1 to n**.

|  |  |
| --- | --- |
| **Model** | **MSE** |
| Linear Regression | 0.037568817732021476 |
| Random Forest Regression | 0.0222482972552489 |
| KNN Regression | 0.039799973297178284 |
| Decision Tree Regression | 0.037568817732021476 |

2.3.3. Root Mean Squared Error or RMSE

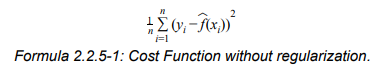
RMSE is the standard deviation of the errors which occur when a prediction is made on a dataset. This is the same as MSE (Mean Squared Error) but the root of the value is considered while determining the accuracy of the model.

In RMSE, the errors are squared before they are averaged. This basically implies that RMSE assigns a higher weight to larger errors. This indicates that RMSE is much more useful when large errors are present and they drastically affect the model's performance. It avoids taking the absolute value of the error and this trait is useful in many mathematical calculations. In this metric also, the lower the value, better is the performance of the model.

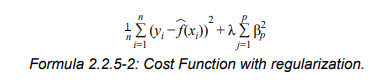
|  |  |
| --- | --- |
| **Model** | **RMSE** |
| Linear Regression | 0.13791732192020476 |
| Random Forest Regression | 0.14915863117918754 |
| KNN Regression | 0.19949930650801342 |
| Decision Tree Regression | 0.19382677248517935 |

2.3.4. Regularization

The regularization is a technique to address overfitting / high variance problem. Overfitting arise when the model fits the data point too precisely and catch also noisy or outlier points. The model does not generalize well with new unknown data. This problem can be due to a complex model with many features. The regularization will penalize the parameter β by reducing them. Given the initial cost function :

****

Regularized it becomes



Where λ is the regularization term determining how much the cost of the parameters will be inflated. Larger λ is, more the function will be smoothed, but a value too large will cause underfitting. Underfitting is when the model fails to catch the trend of the data (to fit the data), and so produce a large MSE. Note that we don’t penalize β .



2.3.5. Tuning of Hyperparameters:

In this subsection, we present the optimal hyperparameters obtained for Random Forest and Decision Tree Regressor Using GridSearchCV.

**Model HyperParameters Range Optimal Parameters**

Random Forestcriterion['mse','mae'] mse

n\_estimators [100,500] 500

max\_features [auto, sqrt, log2] sqrt

max\_depth [4,5,6,7,8] 8

**Model HyperParameters Range Optimal Parameters**

Decision TreeSplitter["best","random"] best

min\_samples\_leaf [1,2,3,4,5,6,7,8,9,10] 10

max\_features ["auto","log2","sqrt",None] None

max\_depth [1,3,5,7,9,11,12] 7

min\_weight\_fraction\_leaf [0.1,0.2,0.3,0.4,0.5] 0.1

max\_leaf\_nodes [None,10,20,30,40,50] 10

The r2 score we got after hyperparameter tuning for random forest regressor is 85% which is less than the default parameter model. So, we choose the Random Forest Classifier with the default value as the best model for prediction.