Machine Learning Engineer Nanodegree

Capstone Project

Jiri Manak
August 8th, 2017

I. Definition

Project Overview

A purchase of a house is usually one of the largest investments in life of most people. For a proper personal finance planning a knowledge of property price is a first step towards this goal.

The price of a house is determined by number of features. The number of square meters, number of rooms and house location are the common attributes. Anyhow, price of the house can be influenced by some specific features. For a buyer, it would be interesting to know not only an average price of the property with required attributes but also which features influence the price most. Comparing features, which have the strongest impact on the price with the personal preferences, allows buyer to make a tradeoff between the dream house and budget.

Problem Statement

We have a collection of data about houses which were already sold. Base on this data, can we predict how much will cost my dream house? Or a property owner may ask: „How much is my house worth? "

This problem can be solved by a predictive regression model which will be able predict the price of an unsold house . The task to do is to create such a model.

Second goal is to create list of the features sorted by its importance to influence the price of the house.

Third goal is to compare results from different regression models.

Datasets and Inputs

Collect data from usual sources as web portals, advertisements, real estate agencies offerings will be an enormous task. Therefore, for purpose of this project I used data publicly available on Kaggle: House Prices: Advanced Regression Techniques.

https://www.kaggle.com/c/house-prices-advanced-regression-techniques

This collection consists of 1462 data points for training and 1461 data points for testing. Data points for testing are delivered without labels and are supposed to be submitted to the Kaggle for competition with other data analysts.

Metrics

To stay compatible with Kaggle metrics I will evaluate the model on RMSE.

- **Root-Mean-Squared-Error (RMSE)** - the logarithm of the predicted value and the logarithm of the observed sales price.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\ln(\hat{y}_i) - ln(y_i))^2}{n}}$$

where $\hat{y}_i$ is predicted value of observation i of regression dependable variable y_i computed for n different predictions

For evaluation and comparison of different models I will also use

- **R^2 score – coefficient of determination –** which is a proportion of the variance in the dependent variable that is predictable from the independent variable. An R2 of 0 means that the dependent variable cannot be predicted from the independent variable. An R2 of 1 means the dependent variable can be predicted without error from the independent variable.

$$R^2 = 1 - \frac{SS_E}{SS_T} \qquad SS_E = \sum_i (y_i - \hat{y}_i)^2 \qquad SS_T = \sum_i (y_i - \bar{y})^2$$

where $\hat{y}_i$ is predicted value of observation i of regression dependable variable y_i computed for n different predictions

https://en.wikipedia.org/wiki/Coefficient_of_determination

http://stattrek.com/statistics/dictionary.aspx?definition=coefficient_of_determination

II. Analysis

Data Exploration

Features

There are almost 80 features for each record which describe each house from more aspects. Description of the features is delivered with the data in separate file. Partial information are visible on following web page:

https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

Sale Price

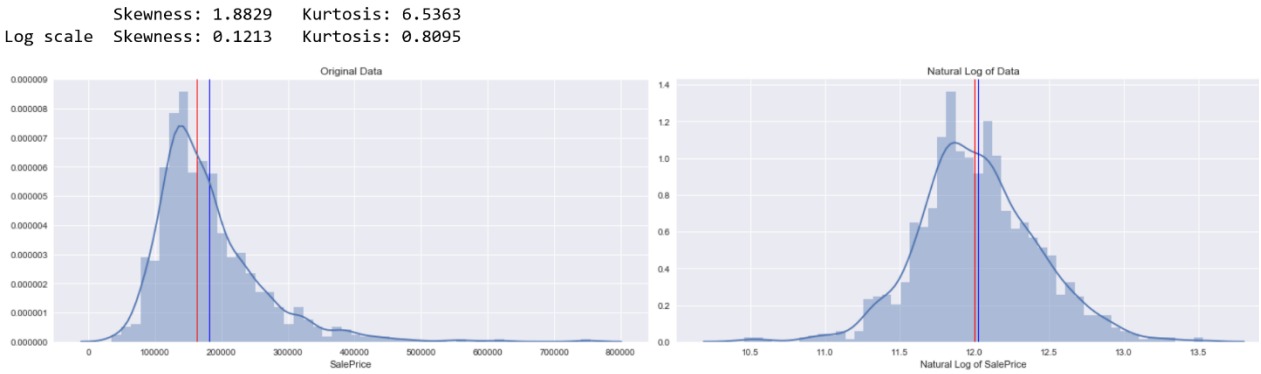
'SalePrice' as a sale price of the house is the target variable. I will finde and train a model which will be able to predict this value based on specified features.

The basic statistic of the sale price:

Maximum price:	*$755,000.00*
Minimum price:	*$34,900.00*
Mean price:	*$180,921.20*
Median price	*$163,000.00*

Standard deviation of prices: *$79,415.29*

The distribution of 'SalePrice' deviates from normal distribution, it shows skewness to the left (towards lower prices) which is not a surprise, because maximal price of a house has no limits, but something like 'smallest' usable house does exist.



Categorical and Continuous values

Simple request on data type stated, that thera are 27 features which are numerical and 43 are categorical.

The features describing measurable attributes as for example square footage are described by continues values, or it can hold number specifying of a number of objects.

Typical continues variable features are for example:

- GrLivArea – Above grade (ground) living area square feet
- LotArea: Lot size in square feet
- BsmtFinSF1: Type 1 finished square feet
- TotalBsmtSF: Total square feet of basement area
- PoolArea: Pool area in square feet
- Fireplaces: Number of fireplaces

There is also a number of categorical values which can hold one value from specified set of values.

Typical categorical varibales are Overal Quality

- KitchenQual: Kitchen quality
- FireplaceQu: Fireplace quality
- Functional: Home functionality rating

Missing Data

Exploring the data by looking for missing values shows, that there is a quite a large number of missing data. Anyhow, in data description it is stated, that values are usually missing if the object doesn't exist. For example, a pool. If property has a pool, then feature 'PoolSF'(pool square footage) is filled out with a number of pools area in square feet. In case there is no pool, then the feature value isn't filled out. In this case I can simply replace missing value by zero. For categorical features I can use value *'None'*.

There are really only few features in dataset with missing values. These values were replaced by the most frequent values for particular feature in the dataset.

Outliers

There are more possible ways how to identify outliers. Removing them might by risky because it can influence the model in negative way. Therefore, I will take a conservative approach. Correlation graph of 'GrLiveArea' with respect to 'SalePrice' shows two data points in upper left corner indicating large living area with unusual low price. I have identified these data points as outliers.

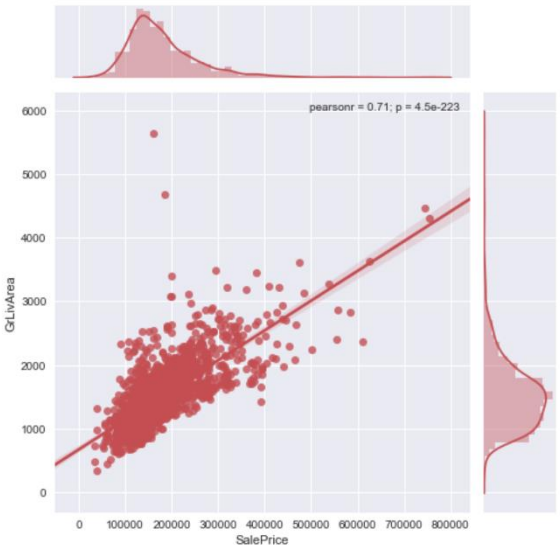


Figure 2. With outliers

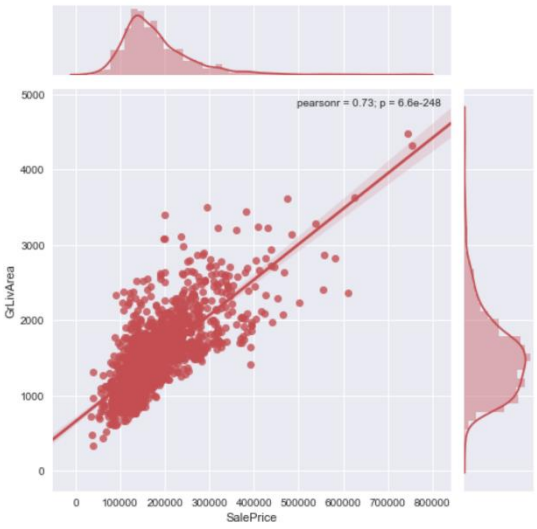
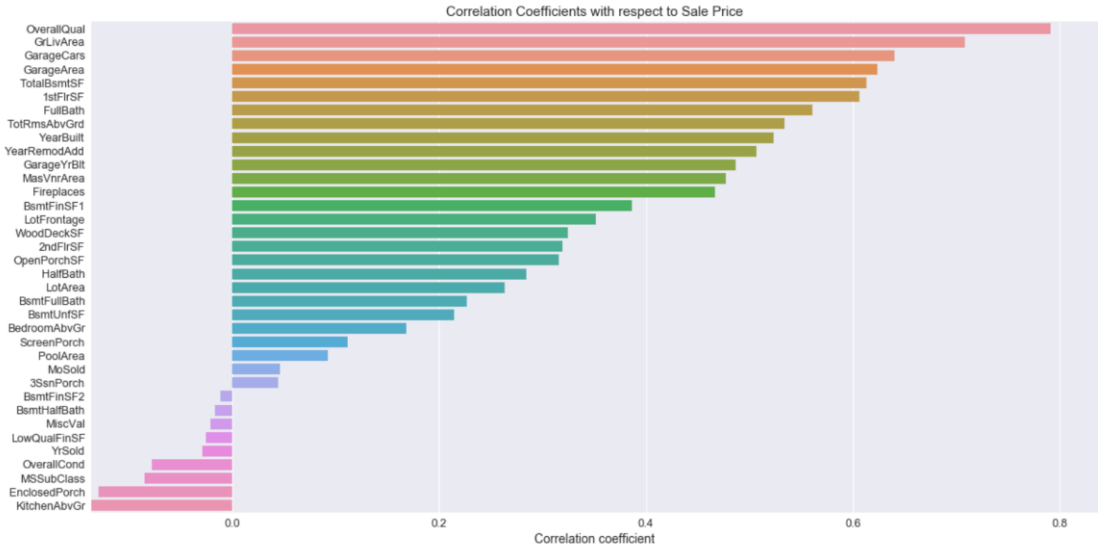


Figure 1. Without outliers

Exploratory Visualization

Basic data investigation shows features correlation with respect to SalePrice. Graph shows that there is a number of features which correlates with SalePrice.

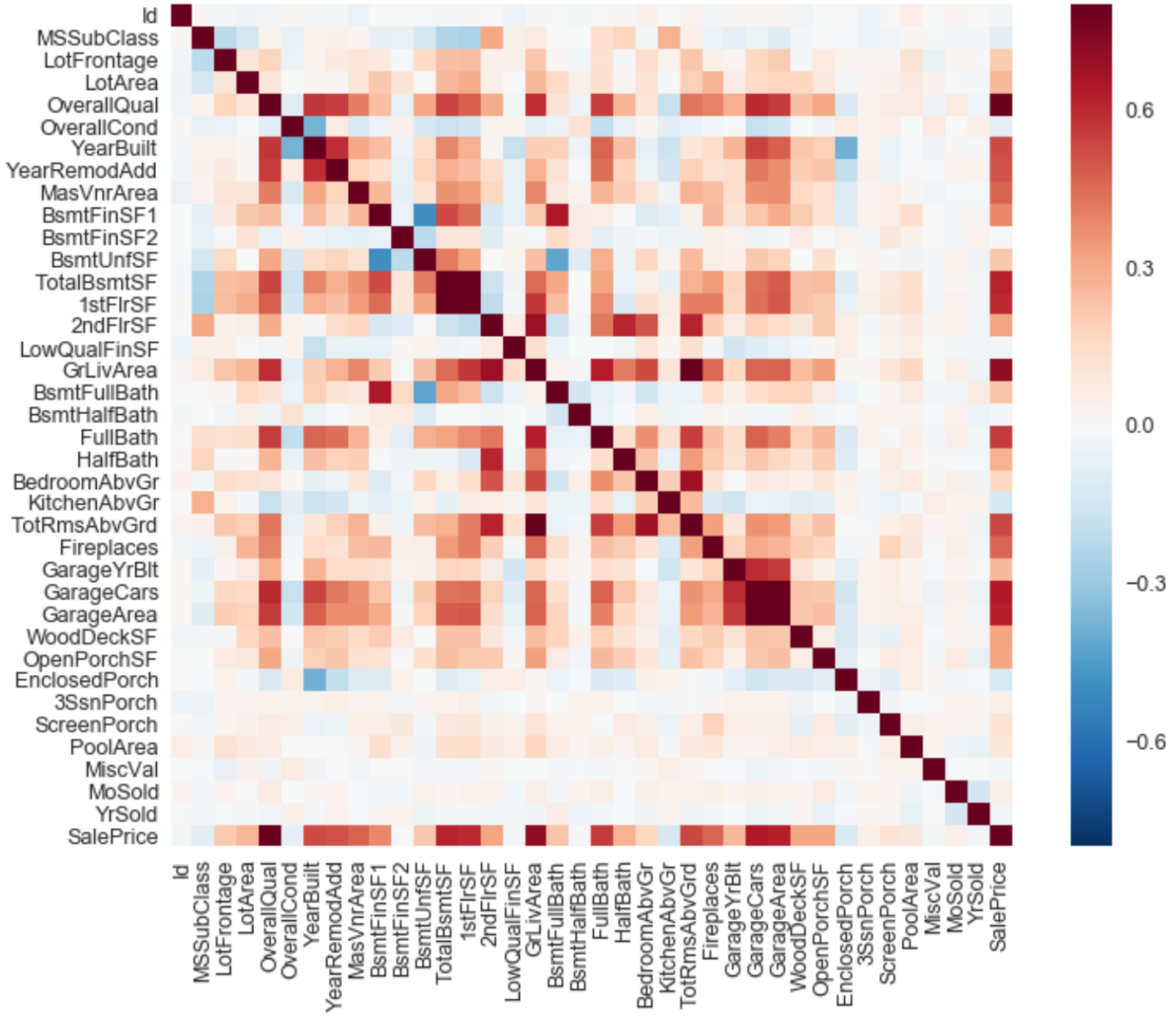


Top 5 features

- OverallQual Overall material and finish quality
- GrLivArea: Above grade (ground) living area square feet
- GarageCars: Size of garage in car capacity
- GarageArea: Size of garage in square feet
- TotalBsmtSF: Total square feet of basement area

Overall quality is a categorical value. Next four features describing a size of a property part. This is an evidence that size of house usually has the highest influence on the price.

Next chart shows correlation heatmap. Investigation shows that high correlated features usually describe the same thing. As for example 'GarageArea' and 'GarageCars'.



Algorithms and Techniques

For this project I chose three ensemble models based on decision tree regression techniques.

Decision tree technique builds models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets. Ensemble models combine results from different models.

Random Forest Regressor is an ensemble classifier made using many decision tree models. Creates multiple trees with randomly selected data.

Gradient Boosting Regressor takes many week learners and combine results of them.

XGBoost Regressor is an optimized implementation of Gradient Boosting technique. More datails are to found at https://github.com/dmlc/xgboost

Benchmark

The property price strongly correlates with square footage, the simple model will be linear regression of square footage with respect to sale price.

Prediction of linear regression of 'SalePrice' with respect to 'GrLivArea' (above ground living area square feet) deliver following results:

```
RMSE               : 56034.3039
RMSE of logarithms : 0.2756
R2 score           : 0.502149
```

Summing square footage of overall basement area with first and second floor creates new feature ,TotalSF'. 'TotalSF' is a sum of basement, 1st floor, and 2nd floor square footage.

Prediction of linear regression of 'SalePrice' with respect to 'TotalSF' deliver even better results:

```
RMSE               : 49471.9159
RMSE of logarithms : 0.2406
R2 score           : 0.611931
```

III. Methodology

Data Preprocessing

Basic preprocessing

Data from Keggle's come in two sets. One part are data to be used for training and the second part are the test data to be used for prediction and is to be submitted to Kaggle competition.

Both sets consists of 'Id' column, which is only sequence number and doesn't hold any information about the house.

Basic preprocessing will include

- separation of 'SalePrice'
- droping of the 'Id' column
- joining of the training and testing sets together

Missing Data

Missing data will be proceed following way:

- numeric data – if missing, imputed by zero
- categorical data – if missing, imputed by 'None'
- features 'Electrical', 'Functional', 'KitchenQual', 'MSZoning', 'Exterior1st', 'Exterior2nd' and 'SaleType' were imputed by most frequent values in the data set

Removing Outliers

I will use simple rule for an elimination of outliers

'GrLivArea' > 4000 SF

and

'SalePrice' <300000

Processing categorical data

It will be necessary to transform categorical values to numbers. I will use label encoders and one hot encoding.

Dealing with skewness

Investigating of data shows number of features with skewed distribution.

Implementation

Data are processed and model is created in Python 2.7 programming language using libraries

- pandas - data analysis toolkit v0.20.3
- scikit learn – machine learning library in Python
- matplotlib
- seaborn
- XGBoost Library

Data are analyzed using jupyter notebook

- data_analysis.ipynb

and for model tuning I created another notebook

- model_tuning.ipynb

Some procedures are implemented in separate files

- usefull_methods.py
- do_actions.py
- stopwatch.py

Refinement

I used two techniques how to tune the model. I will implement each preprocessing procedure in separate function, that I can simply include or exclude it in data preprocessing. I will observe which combination of preprocessing steps deliver best results. The possible actions which can be turned on and of:

- elimination of outliers
- eliminate skew
- remove non-significant features

Second technique is tuning hyperparameters for chosen regressors. I will use grid search technique to tune *max_depth* and *n_estimators*.

max_depth is maximal depth of decision tree

n_estimators is number of estimators used by the ensemble regressor

Best performing hyperparameters will be used for final model

IV. Results

Tuning model by changing actions and searching for best hyperparameters delivered slightly different results. Best results delivered a combination of:

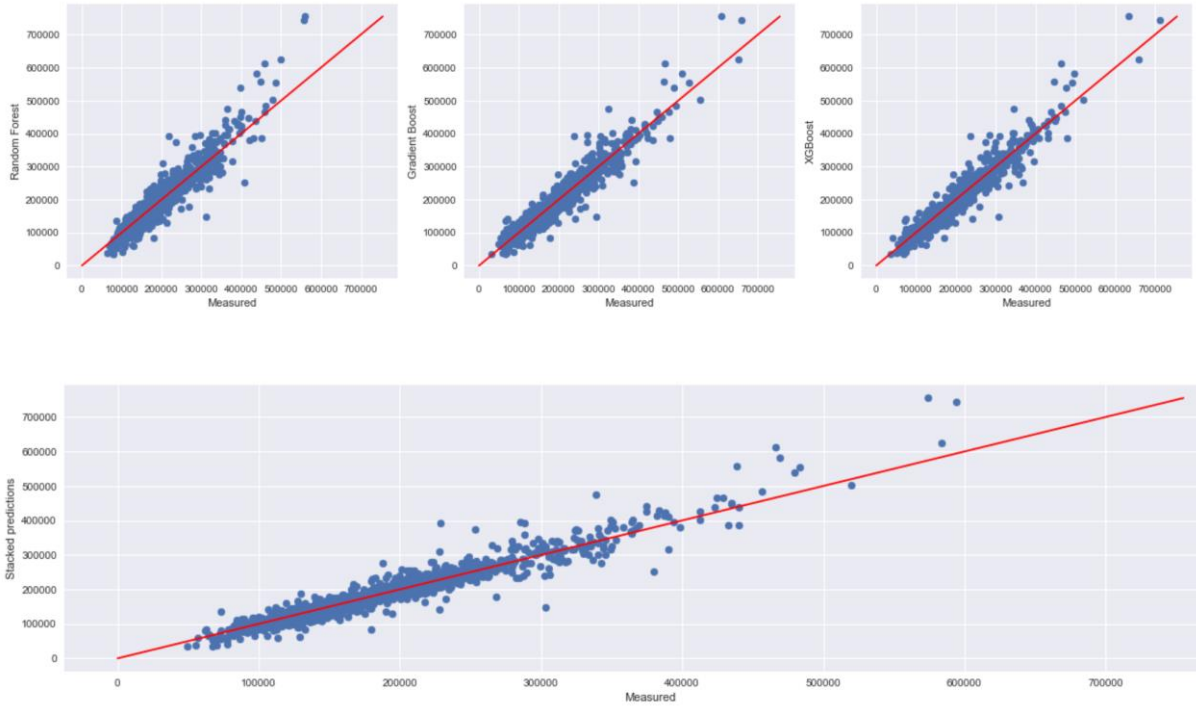
- elimination of outliers ON
- eliminate skew OFF
- remove non-significant features OFF (all features active)

Best results on training data

```
XGBoost
RMSE               : 21546.7702
RMSE of logarithms : 0.1181
R2 score           : 0.926484

Gradient Boost
RMSE               : 22004.9835
RMSE of logarithms : 0.1224
R2 score           : 0.923324

Random Forest
RMSE               : 25372.8165
RMSE of logarithms : 0.1352
R2 score           : 0.898058
```



Weights calculation for stacked regressions is calculated as relation of particular r2 score to sum of all r2 scores.

```
sum_r2 = xgb_r2 + gbr_r2 + rfr_r2
xgb_w = xgb_r2/sum_r2
gbr_w = gbr_r2/sum_r2
rfr_w = rfr_r2/sum_r2
```

Best results on test data

Result of stacked regressions gives better result in Kaggle competition: RMSE

- 0.12900 for stacked and
- 0.13289 for XGBoost.

V. Conclusion

Reflection

This was very interesting project. On the beginning, it looked like a simple task. But it is possible to spend weeks trying all ideas one can have.

I am satisfied with the results, and I believe that model is usable for estimation of house prices for given area and time period. I have tried to achieve better results on

the Kaggle competition and I spent lot of time with tuning but wasn't able really to make a significant step forward. I have learned, that one skill of the good machine learning engineer is also to know when to stop twisting the current model.

Improvement

There is a number of possible ways how to improve the model. One way is to use other methods of machine learning as for example:

- use other regressors
- tune more or other hyperparameters
- support vector regression (I tried to use this method, but it delivers strange results, therefore I skipped it. Probably because I didn't transform, normalize the data? Needs more time to investigate.
- Neural Networks Regressor – this method I had in the scope, but I read that this method might not work well because not sufficient data for NN training available. Anyhow would be nice to try, but it will costs me another month ☹.

There are still open possibilities how to tune current model (based on ensamble decision trees). Some ideas are:

- use another functions to unskew data
- precise selection of data to be droped
- tuning of weights of models stacking
- data transformation – data normalization