Tose’e Azad Education Institute

Intelligent house buying consulting system

Advisor

Dr.Yaser Zerehsaz

By

Mobina Khademi

Abstract

In this project we are facing with two datasets that are concluded variables covering almost aspect of a residential house. After exploratory data analysis (EDA) process, we choose the Xgboost model with 90% accuracy as the best model for prediction of house sale price. Then we propose a recommendation system based on cosin similarity for finding best options for client with specific requests. This system provides for a person with a certain amount of capital, House area, selected area, number of bedrooms, Oldness of house, general zoning (commercial, residential, agricultural, ...) and the number of specific floors, can make the best possible choices.

Introduction

Prediction house prices are expected to help people who plan to buy a house so they can know the price range, then they can plan their finance well. In addition, house price predictions are also beneficial for property investors to know the trend of housing prices in a certain location.

Most people know the benefit of owning a house, because buying a house is considered the most utilized and profitable investment and for this reason, it's necessary to choose best choices based on their preferences.

There are many factors that can be influence the price of a house. In this project we are facing with 79 of features that can describe the different aspects of house. Some of them are more important than others. In first section of this project the dataset must be cleaned that means we must choose the best strategy for treating of missing values and work with outliers. Then we must fit some models that can predict the sale price of house and choose the best model.

In the second part we must provide a recommendation system. Recommendation systems provide suggestions to users in various fields like movies, videos, restaurants, hotels and other items. Here we provide a recommendation system for finding most similar houses based on the preferences of customer. For do this section we use the cosin similarity concept. Also, we can predict the price of house based on the preferences of customer with the best model of the previous stage.

Define the problem

The purpose of this project is to provide a model to better predict the selling price of residential houses. Having such models can be useful as a criterion in the pricing process, estimating the profit of construction projects, determining the trading strategy of housing consultants and providing consulting services automatically.

Approach

First part: Provide a forecasting model

1. Import data

This project is about house price prediction, where the used data set is on the house sale prices of residential houses in Ames, Iowa. For the training set, it gives information of totally 1314 houses, with each house described into 80 variables. There are totally 43 categorical variables, and 37 numerical variables. The test set contains 1605 house records without the last column of sale price.

2. Data Analysis

We did thorough data analysis on each of the variable and their respective correlations with the house prices to be predicted. We can see in Fig.1. the distribution of sale price.

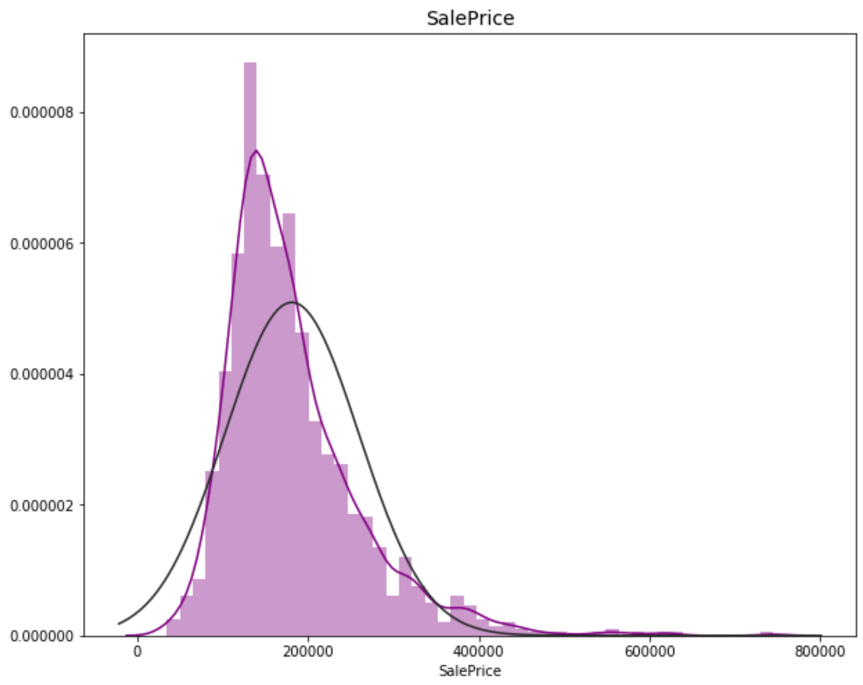


Fig.1. distribution of sale price

After normalization and log transformed, as can be seen in Fig.2. the distribution of target variable (sale price) has better fit with normal distribution.

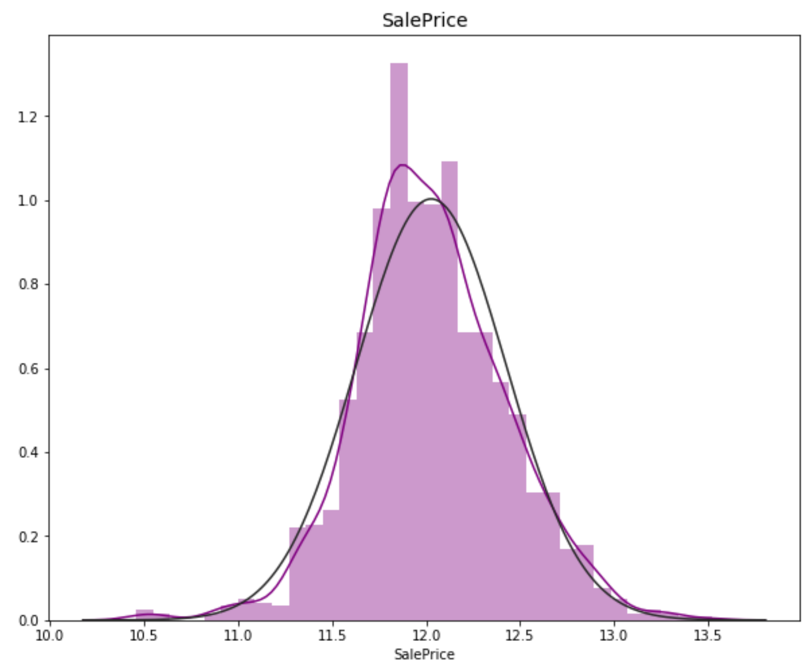
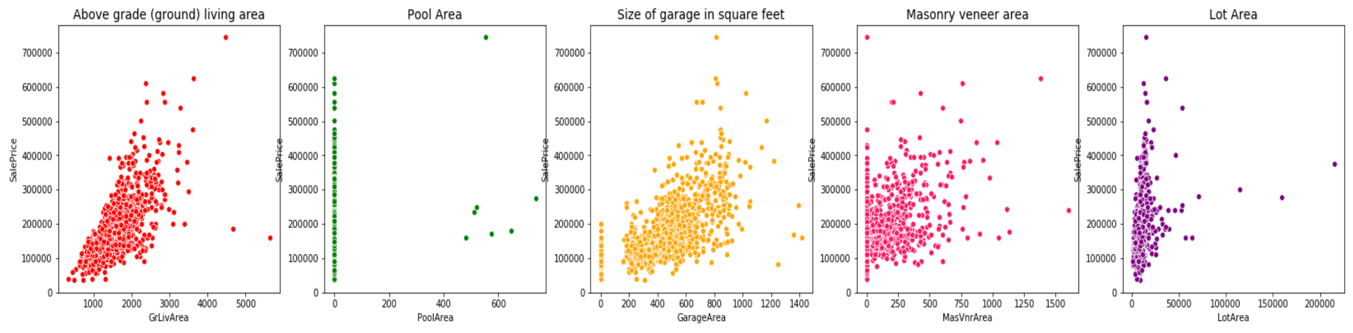


Fig.2. distribution of sale price after normalization

For numerical variables, Fig.3. Shows the scatter plots of top 15 numerical features with highest correlation with the house sale price. These plots are useful for seeing features correlation with sale price also having hint on out layers.



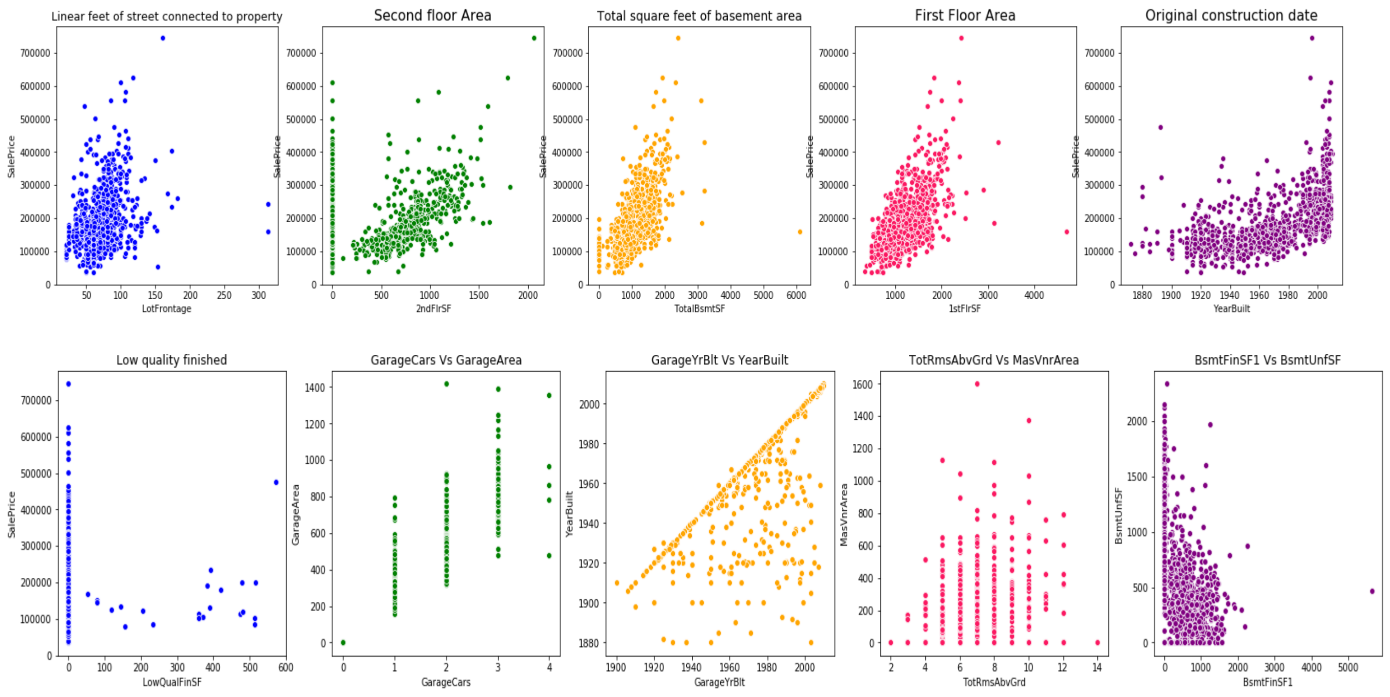


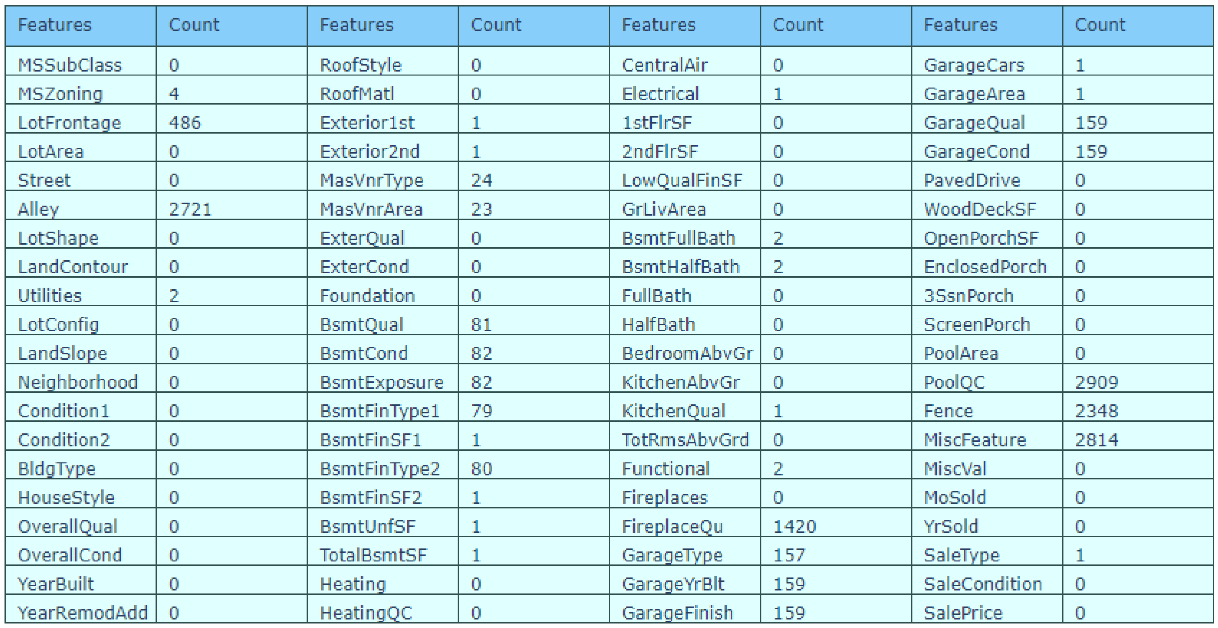
Fig.3.Scatter plot of first 15 numerical variables

3. **Data Munging**

- Treating missing values

At the first step we must deal with missing values of the both data sets. So, we add a column with name ‘sale price’ to test set and concatenate them to each other.

Table.1. shows the total number of missing values of our dataset. 34 variables have missing values.



Features	Count	Features	Count	Features	Count	Features	Count
MSSubClass	0	RoofStyle	0	CentralAir	0	GarageCars	1
MSZoning	4	RoofMatl	0	Electrical	1	GarageArea	1
LotFrontage	486	Exterior1st	1	1stFlrSF	0	GarageQual	159
LotArea	0	Exterior2nd	1	2ndFlrSF	0	GarageCond	159
Street	0	MasVnrType	24	LowQualFinSF	0	PavedDrive	0
Alley	2721	MasVnrArea	23	GrLivArea	0	WoodDeckSF	0
LotShape	0	ExterQual	0	BsmtFullBath	2	OpenPorchSF	0
LandContour	0	ExterCond	0	BsmtHalfBath	2	EnclosedPorch	0
Utilities	2	Foundation	0	FullBath	0	3SsnPorch	0
LotConfig	0	BsmtQual	81	HalfBath	0	ScreenPorch	0
LandSlope	0	BsmtCond	82	BedroomAbvGr	0	PoolArea	0
Neighborhood	0	BsmtExposure	82	KitchenAbvGr	0	PoolQC	2909
Condition1	0	BsmtFinType1	79	KitchenQual	1	Fence	2348
Condition2	0	BsmtFinSF1	1	TotRmsAbvGrd	0	MiscFeature	2814
BldgType	0	BsmtFinType2	80	Functional	2	MiscVal	0
HouseStyle	0	BsmtFinSF2	1	Fireplaces	0	MoSold	0
OverallQual	0	BsmtUnfSF	1	FireplaceQu	1420	YrSold	0
OverallCond	0	TotalBsmtSF	1	GarageType	157	SaleType	1
YearBuilt	0	Heating	0	GarageYrBlt	159	SaleCondition	0
YearRemodAdd	0	HeatingQC	0	GarageFinish	159	SalePrice	0

Table.1. number of missing values

According to the project description, many categorical features ('BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'FireplaceQu', 'GarageType', 'GarageFinish', 'GarageQual', 'GarageCond') have "NA" as one of their valid levels, which really means a certain categorical. So, to handle them we replace them with ‘Unknow’. After that, the real NA values are left.

Some features like 'Alley', 'Fence', 'FireplaceQu’, ‘MiscFeature' and 'PoolQC' whose amount of missing data is more than 70%, we remove them.

In the following, we had 3 strategies for dealing with missing values in order to finding best way to deal with missing values:

1. LotFrontage has most of missing values by 486, We fill them with mean of LotFrontage. Second high miss value by 159 is for GarageYrBlt. We understand miss values in this feature are for those that don’t have Garage. so, we decide to replace them with 1890 (older than any garage bult in that order). Then the number of missing values in our dataset is less than 3 percent of all data so let’s remove them.

2. In second strategy we do with missing values of LotFrontage and GarageYrBlt as the first strategy. Then we fill missing values of other features with most frequency(mod) in each category.

3. In third strategy missing values of some features are imputed based on correlation between the features. The features (Exterior1st, MSZoning, MasVnrType, BsmtFinSF2, KitchenAbvGr) are estimated according to the correlation feature using the 'mode' method. For example, considering the importance and coordination between exterior covering on house (Exterior1st) and the type of roof of the house, it was decided to use 'Plywood ' to estimate the missing values. Also, in some of the features, because the features related to them are None and 0, so we estimated 0 for them. For example, in the MasVnrArea feature, because the sample had a Missing Value (MasVnrType = None), it was estimated to be 0. The GarageYrBlt feature has 150 missing values, which can be removed. For some features (Utilities, BsmtUnfSF, TotalBsmtSF, Electrical, SaleType, Functional) we use ‘mode’ as the most frequency. LotFrontage is a numerical feature, so for impute missing values of this feature, we can’t use the average of train dataset to estimate test dataset. because we get data leakage. For this reason, we will create an array that contains random numbers, which are computed based on the mean LotFrontage.

We made 3 datasets based on above strategies and after the modeling, third dataset was the best in prediction accuracy. So, third data set was selected to continue the project.

- Working with outliers

So, none of the features have missing values. In this stage we changed categorical variables to dummy variables. After that we must find outliers in train dataset. An outlier is a value or an observation that is distant from other observations, that is to say, a data point that differs significantly from other data points.

We fit a full model on all of the variables. And after that we plot fitted values versus residuals.

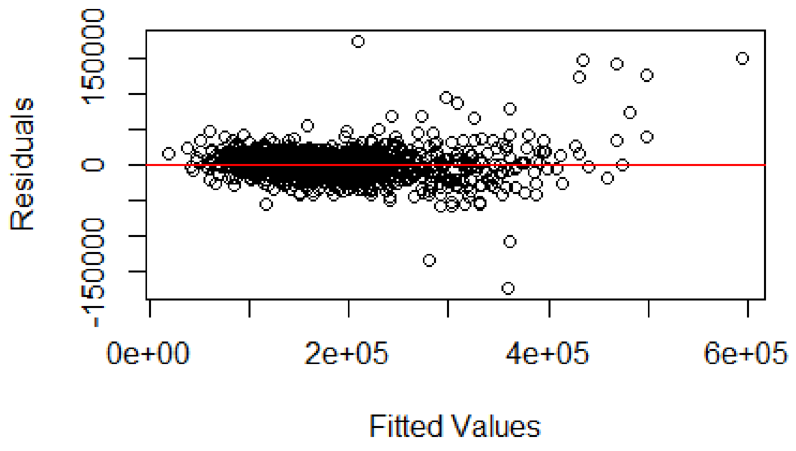


Fig.5. fitted values versus residuals

As can be seen in Fig 5 the residuals seem to be independent and we have constant variance in our dataset.

After that we use half-normal plot. This plot use to find large leverages and can show the divergence is large enough. We choose high leverage points in a list is called ‘HLV’. Records leverage are depicted in Fig.6. and those with high leverage are suspected to be outlier.

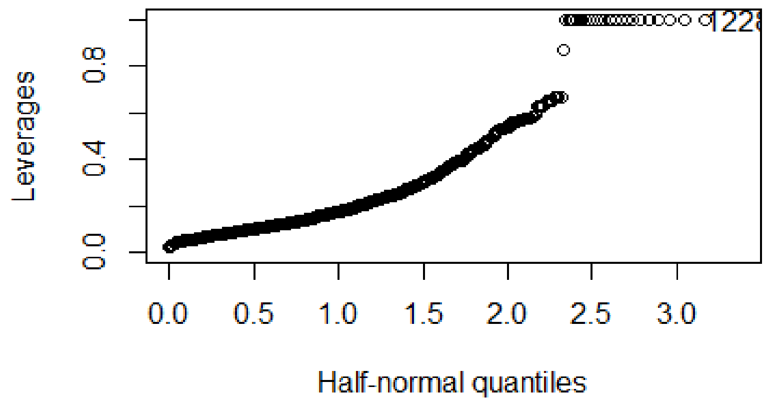


Fig.6. half-norm plot

Then we plot studentized residuals versus fitted values in Fig 7 and we choose high studentized residual points (bigger than 3 and lower than -3) in a list is called 'HSR'.

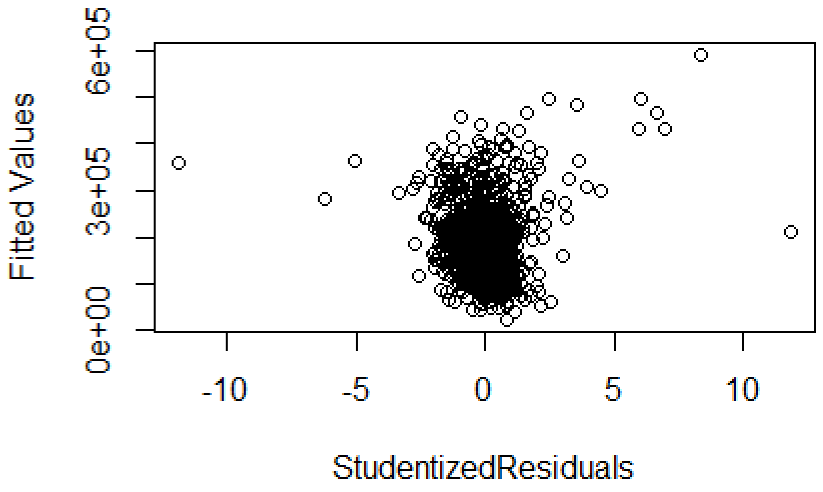


Fig.7. studentized residuals plot

The outliers are intersection of HLV and HSR. In this stage we detect the outliers and we choose to remove 21 detected outliers from the train dataset.

4. Feature engineering

In this part, we add 6 new features:

1) Age of building called 'AgeBlt'. This feature is bult from the difference between the year of sale (YrSold) and the year of building (YearBuilt).
2) The total number of the floors called 'Numfloors'.
3) The total feet square of porch called 'TotalPorchSF'. This feature is obtained from the sum of the values of 'OpenPorchSF' and 'EnclosedPorch' and '3SsnPorch' and 'ScreenPorch'
4) The total feet square of the house called 'TotalSF'.

We can see in Fig.7. relation between added features with sale price

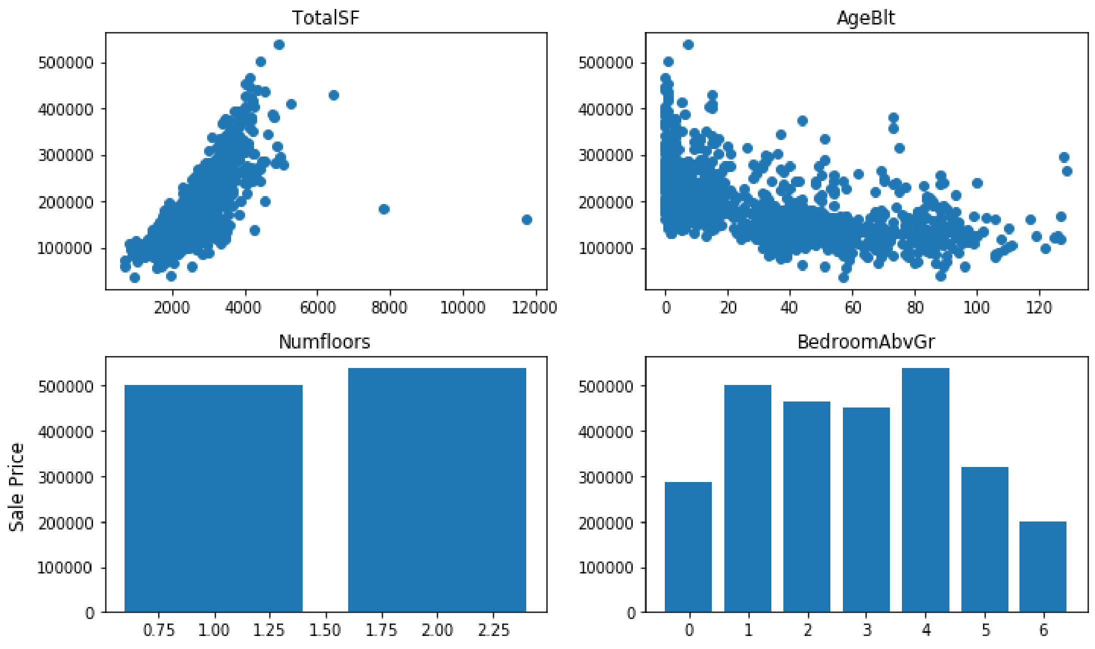


Fig.7. Relation between added features and sale price

Fig.8. and Fig.9 show relation between Neighborhood feature and MsZoning feature with sale price. From this graphs how expensive or cheap are different zoning and neighborhood are illustrated.

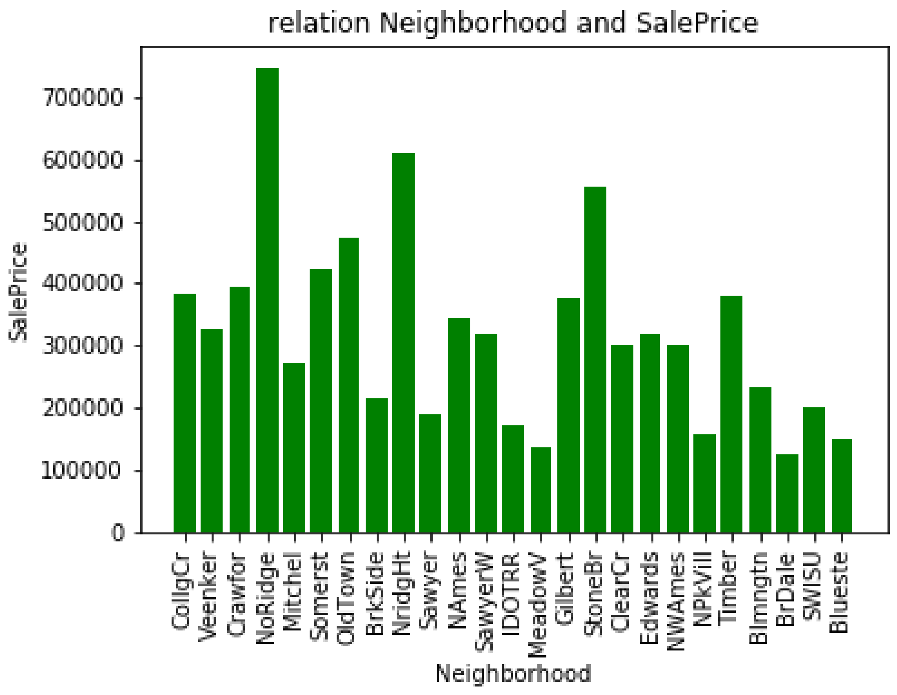


Fig.8. relation between Neighborhood and sale price

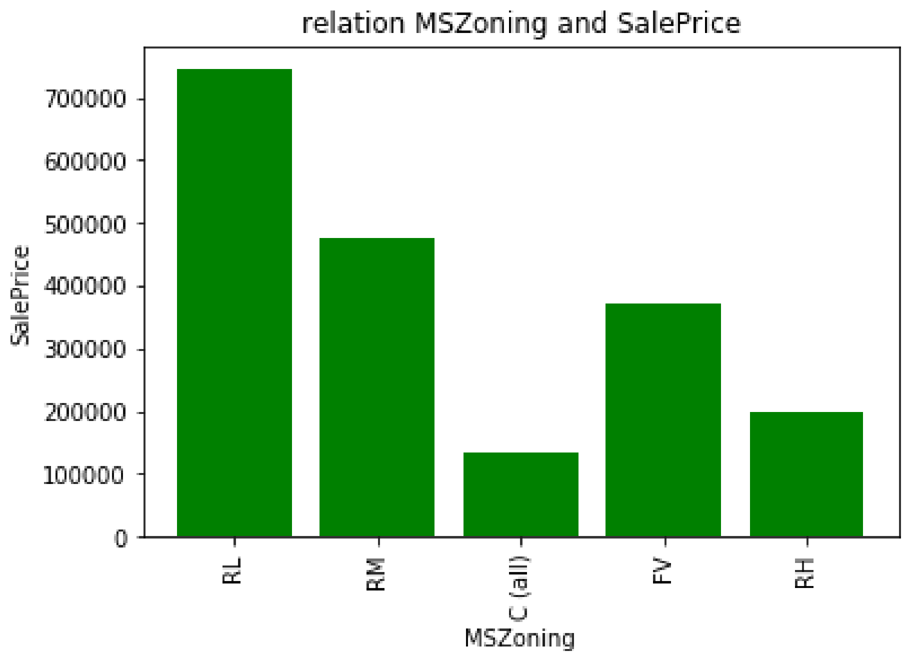


Fig.9. relation between MsZoning and sale price

5. Modeling

We leave 20% of the data for testing and use 80% for training the models. For linear models, we tried and tested with the regular OLS model, and the regularized linear models of Ridge, Lasso. For non-linear models, we tried decision tree, RF and xgboost. In the following we describe these models.

linear regression is a linear approach to modelling the relationship between a scalar response and one or more explanatory variables. In this method interpretability is difficult and variable selection is not straightforward and it might lead to overfitting.

Another model is Ridge regression, that is often used to solve the issues of high correlated variables (multicollinearity). This method shrinks the regression coefficients by imposing a penalty on their size and it addas a constraint on the size of the parameters.

Another model is lasso regression, that is a Shrinkage and Variable Selection method for linear regression models. this is often used when there is sparsity.

Sparsity means that some of coefficients are zero, and we can restrict the coefficients.

A decision tree is a supervised machine learning model used to predict a target by learning decision rules from features. As the name suggests, we can think of this model as breaking down our data by making a decision based on asking a series of questions. This method can be used for variable selection.

Another model is Random forest. This is a type of supervised machine learning algorithm based on ensemble learning.

The following are the basic steps involved in performing the random forest algorithm:

1. Pick N random records from the dataset.
2. Build a decision tree based on these N records.
3. Choose the number of trees you want in your algorithm and repeat steps 1 and 2.
4. In case of a regression problem, for a new record, each tree in the forest predicts a value for Y (output). The final value can be calculated by taking the average of all the values predicted by all the trees in forest.

The last model is Xgboost. Extreme Gradient Boosting (xgboost) is similar to gradient boosting framework but more efficient. It has both a linear model solver and tree learning algorithms. So, what makes it fast is its capacity to do parallel computation on a single machine.

For model tuning, Sklearn's grid search with CV function is used to find the optimal hyper-parameter values.

The first argument in GridSearchCV is the estimator. To define the estimator, we just define models without selecting the hyperparameters.

Second, we need to define the hyperparameters. Typically, a dictionary is used to define all parameters.

In linear regression the variable Hyperparameter is about normalization. The best score is obtained when normalization is *false*.

In lasso regression the variable Hyperparameters are about *alpha* and *selection*. In this model whose best tuned hyper-parameters are *alpha = 1* and *selection* is *random.*

In ridge regression the variable Hyperparameters are about *alpha* and *normalization*. The best score is obtained when normalization is *false* and *alpha* is 100.

In decision tree the best hyper-parameters are *max_depth=8*, *min_sample_split=11*, *splitter=random*, *ccp_alpha=0*, *max_feature=auto* and *criterion=mse*.

In random forest the variable hyper_parameters are a dictionary with some keys. The first key is the tuning parameter max_*depth* and the second key are *n_estimator* and other keys are about *max _feature* and *criterion.* In this model whose best tuned hyper-parameters are *max_depth = 15, n_estimator =600* and *max_feature = auto* and *criterion = mse.*

In Xgboost whose best hyper-parameters is *n_estimator =200* and lerarning_ *rate=0.1.*

The final achieved model performance of tested models for the local test set are listed as follows in Table.2.

	score
Linear regression	0.84
lasso	0.69
ridge	0.85
Decision tree	0.78
Random forest	0.88
Xgboost	0.90

Table.2. Result of models

As can be seen RF and Xgboost are better than others. we choose Xgboost to continue the second part.
We also used PCA, but it made the prediction accuracy worse (0.81).

Result of top 15 important features collected from best tuned RF and xgboost models is shown as follows in Fig.10. and Fig.11.

Due to their different learning behavior, the resultant feature importance results are also different.

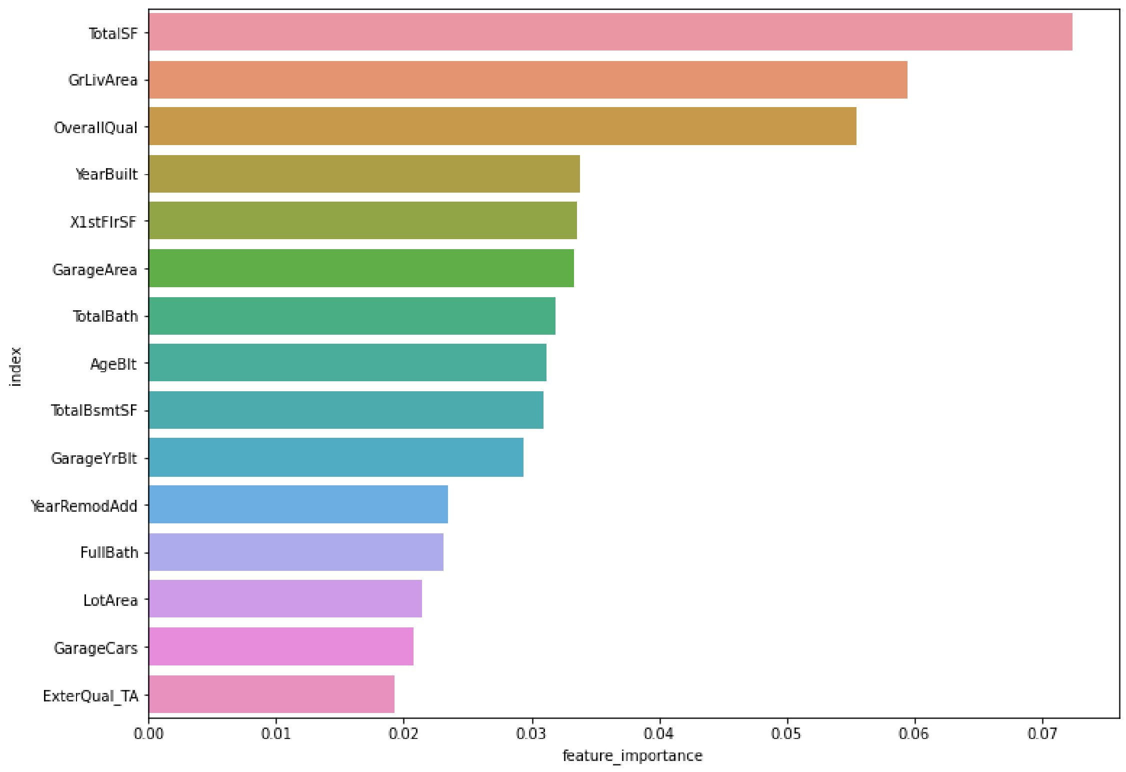


Fig.10.important features tuned of RF

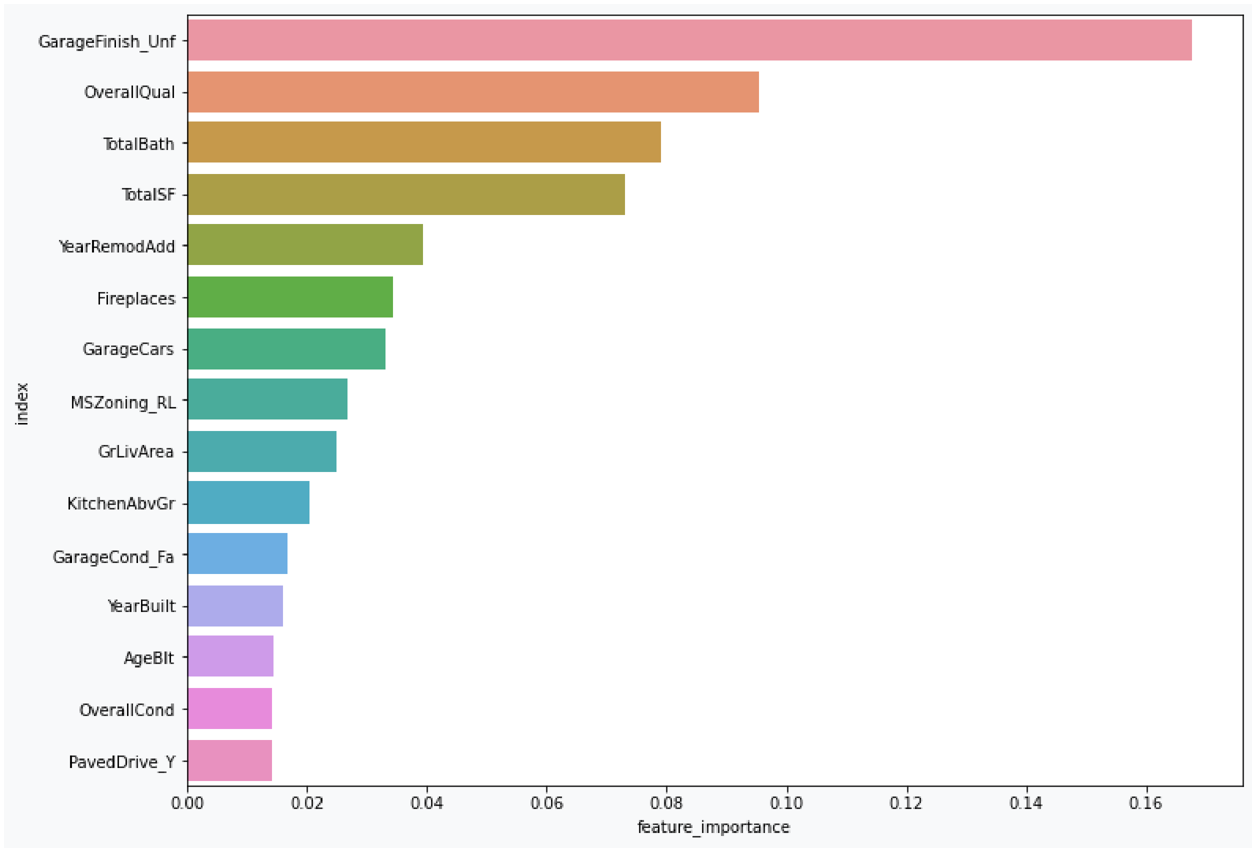


Fig.11.important features tuned of Xgboost

Results

This project is really a great learning experience, which lets us going through the whole entire process of building a machine learning model to solve a practical regression problem in real world. We start from the data analysis, cleaning, preparation, etc. we fit different models, and choose RF for predict sale price.