Assignment: Advanced Machine Learning and Computational techniques for Industry 4.0

Prediction of House Prices :

Advanced Regression Techniques

Alper Soysal

Contents

Introduction

Purchasing a dream house with affordable price or selling a house with desired price are one of the difficulties that people have to face nowadays. Objective of this assignment is to predict the sales price of the houses in Ames, Lowa using by Machine Learning Techniques. Result of the prediction could be used by individuals to evaluate expected price of their dream houses and help with their decisions. Findings also could be used by real estate agent to maximize their profits and set a proper price for their estates.[1]

Data Description

As stated above, The Ames Housing dataset complied by Dean De Cock was used for the assignment. The whole dataset comprises immense number of explanatories and 2930 observations. The explanatories consist categorical and numerical values.

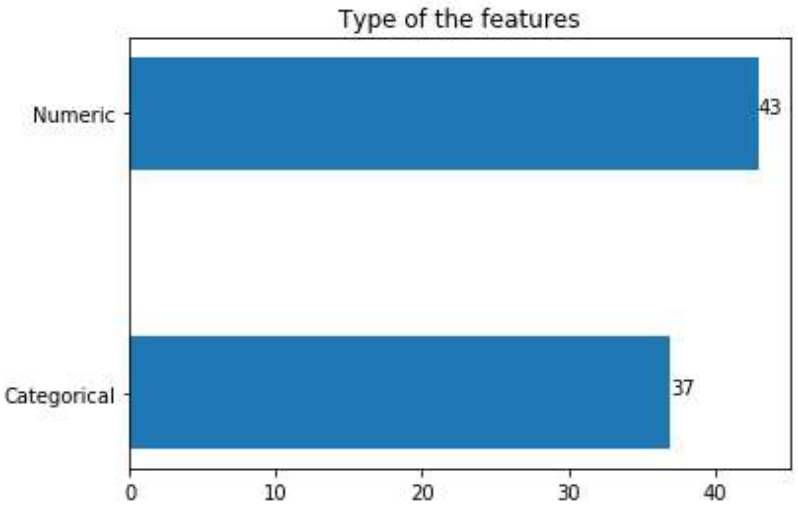

Figure 1

As the bar graph is shown above, the number of categorical and numeric features are 37 and 43, respectively.

The whole dataset is divided into 2 called “train” and “test” for the Kaggle competition. Train dataset is going to use for the train a model. Test dataset comprises the information of the houses that is needed to predict the sale price. It can be seen that the difference between train and test dataset is only the number of the observation. In addition, obviously, the test dataset has one less feature “Sale Price” that is the target value of the problem.

Data Pre-Processing

Before creating regression models, the dataset should be cleaned and data transformations need to be made. To avoid inconsistency, both ‘’train’’ and ‘’test’’ datasets are needed to concatenate.

Data cleaning

The data possibly include some NaN values which means those specific values are not presented in the house. Most of the machine learning models could not accept data which have a NaN values as an input. Moreover, those NaN values have an enormous impact on Model accuracy. For this reason, necessary transformation must be made for the NaN values.

Deal with Nan values

There are a variety number of methods for how to deal with NaN values in given Data set. The most known method is to delete the observation with missing values. Before the action has taken, the number of the NaN Value for each feature is needed to detect. Graph below shows the features which have a Nan value over the half of the observation.

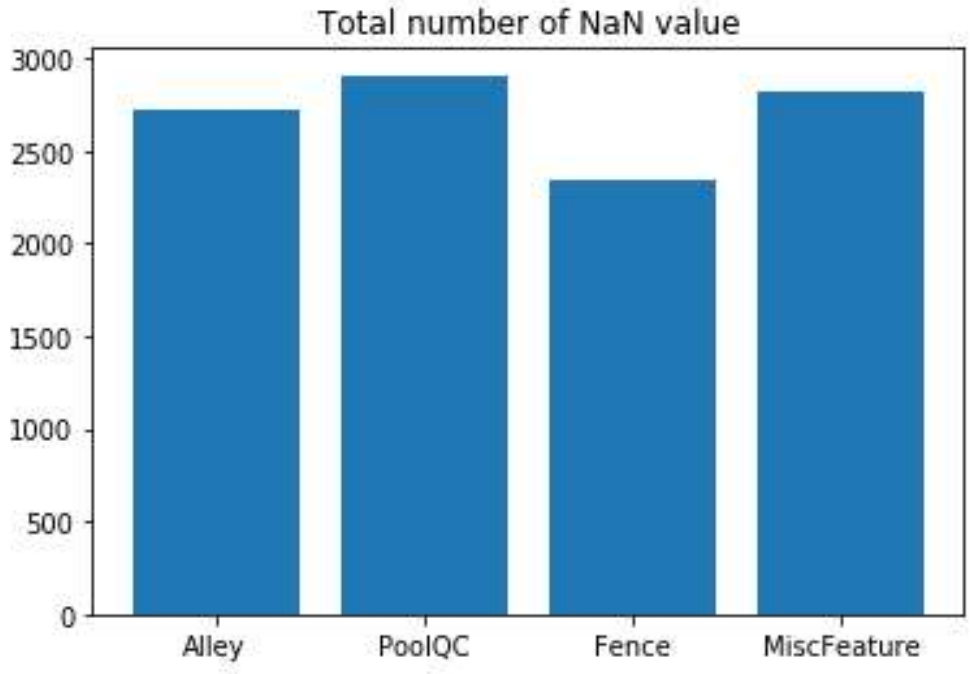

Figure 2

Transformation of the those missing values is not worth for the sale price prediction model because of the most of their values are missing. In this case, the best approach is to delete those features.

As stated before, the dataset comprises both numeric and categorical variables. The table shows the number of the NaN value for each categorical features.

	The number of NaN values
MSZoning	4
Utilities	2
Exterior1st	1
Exterior2nd	1
MasVnrType	24
BsmtQual	81
BsmtCond	82
BsmtExposure	82
BsmtFinType1	79
BsmtFinType2	80
Electrical	1
KitchenQual	1
Functional	2
FireplaceQu	1420
GarageType	157
GarageFinish	159
GarageQual	159
GarageCond	159
SaleType	1

Figure 3

Deleting features with missing values are not useful, considering the data set is unique like in this assignment. Some features with NaN value do not mean that those values are missing. For instance, Explanatory called “GarageType” includes some NaN values which represent that the house does not have garage. Considering unique features consists only categorical variable instead of deleting the observation. The proper approach is to replace the NaN variable by categorical variable “None”. According to data description, the names of the explanatory whose missing values need to be replaced with " None” are 'MasVnrType','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2','FireplaceQu','GarageType','GarageFinish','GarageQual','GarageCond'.

The rest of feature’s missing variables is replaced by its own most frequently categorical value.

Related features:
"MSZoning","Utilities","Exterior1st","Exterior2nd","Electrical","KitchenQual","Functional","SaleType"

the table shows the number of the NaN value for each Numeric Features.

	The number of NaN values
LotFrontage	486
MasVnrArea	23
BsmtFinSF1	1
BsmtFinSF2	1
BsmtUnfSF	1
TotalBsmtSF	1
BsmtFullBath	2
BsmtHalfBath	2
GarageYrBlt	159
GarageCars	1
GarageArea	1

Figure 4

As the table shows, the most variable of "LotFrontage" and "GarageYrBlt" features are missing. The proper approach to deal with those missing values is to replace them by mean value of related explanatory. The rest of the features filled with “0”.

After the making visual inspection on data and making some plots, some of the features with low variance were found.

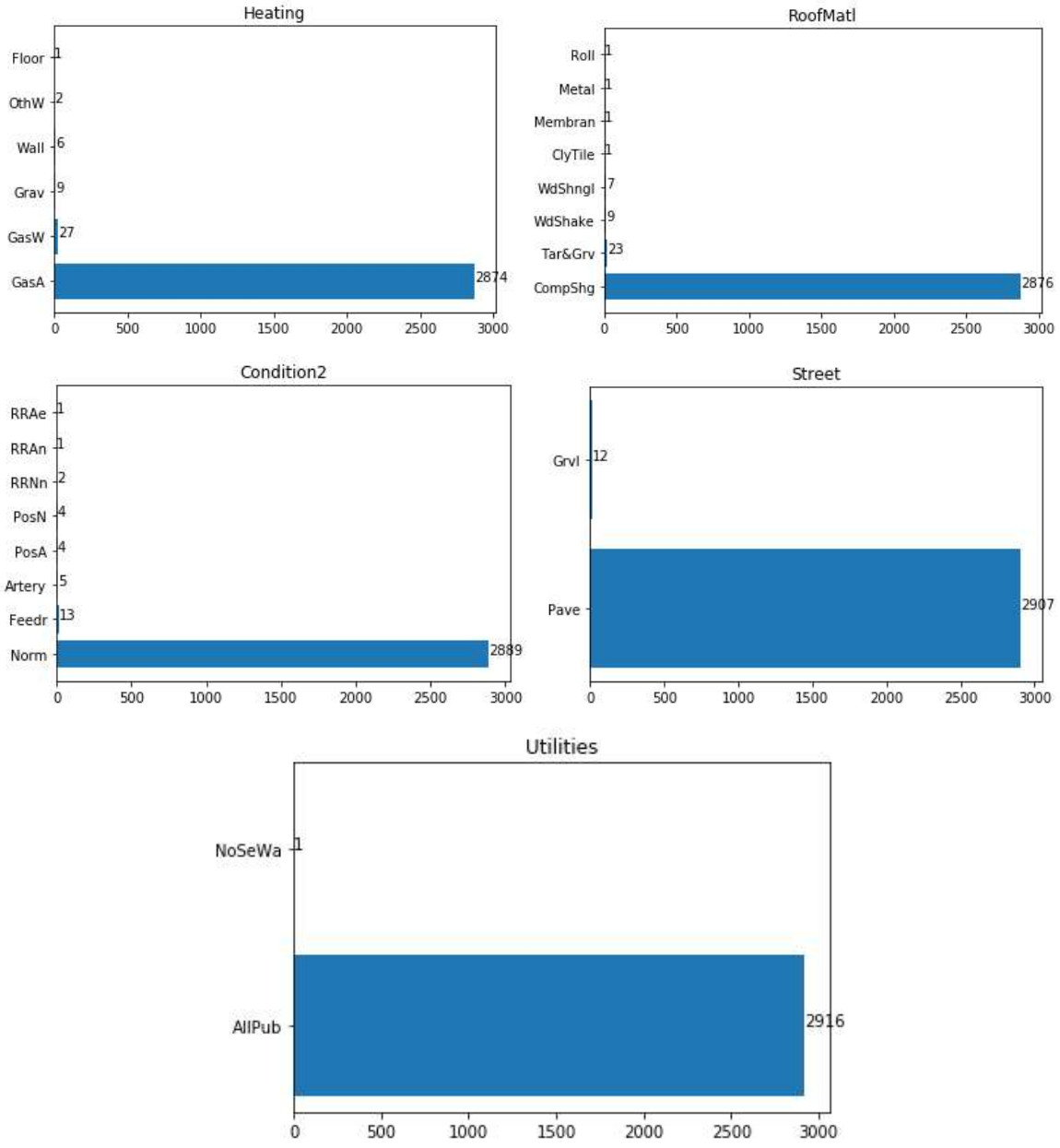

Figure 5

As the graph shows features with low variance has not any meaningful information for the model. For this reason, the best solution is to delete them to simplify the model.

In conclusion, clean dataset is obtained after the methods explained above are carried out.

Data transformation

The most of machine learning algorithm expect numerical values, still some algorithms are able to work with non-numerical values. In this assignment, the machine learning model aimed to use requires

numerical values. As explained before in the data description section, the dataset consists some features with categorical variable. Hence, data transformation from categorical variable to the numerical to meet requirements of the machine learning model.[2]

There are variety of methods to convert categorical values into numerical values. 2 main methods are used for our dataset: Label-Encoder and One-Hot-Encoding.

Label-Encoder:

Label encoder method allows to convert the categorical values into numerical values in order from 0 to number of the categories. If the categorical data were ordinal , label encoder methods would be proper way to encode[3]. For instance, looking deeply into one of the features called “ExterCond” , the present condition of the material on the exterior evaluated by using ordinal categorical value such as Ex: Excellent Gd : Good etc. The features that have ordinal categorical data are shown below after taking a look in the date description:

"LotShape" "LandContour" "FireplaceQu" "LandSlope" "BldgType" "HouseStyle" "ExterQual" "ExterCond" "BsmtQual" "BsmtCond" "BsmtExposure" "BsmtFinType1" "BsmtFinType2" "HeatingQC" "Electrical" "KitchenQual" "Functional" "GarageFinish" "GarageQual" "GarageCond" "PavedDrive"

One-Hot-Encoding:

The idea of this method is that it converts each feature with categorical values into a new column and assigns a 1 or 0. Logic 1 represents the existence of the class and the meaning of logic 0 is the absence of the class. The advantages of this method are to avoid order issue. On the other hand, the methods produce more columns and it can affect the simplicity of the model.[4]

One hot encoding method is used for the nominal categorical and non-ordinal data. Related features are shown below:

"MSZoning" "LotConfig" " Neighborhood" " Condition1" "RoofStyle" "Exterior1st" " Exterior2nd" " MasVnrType" " Foundation" "CentralAir" " GarageType" " SaleType" " SaleCondition"

Sampling strategy

Sampling is essential to avoid overfitting the model. Simple Random Sampling is used in the assignment. Elementally, input data are divided randomly into sub-dataset with specific test size. In this type of sampling, each observation has an equal probability of being chosen.

End of the sampling process, “Train” dataset is divided into input training, output training, input validation and output validation dataset.

Modelling

After the data preprocessing, clean dataset is obtained, and it is ready to build a model.

Variety of regression algorithm is performed to find out the best model that fit our dataset.

Linear regressor

SVR is used to establish the linear relationship between an input variable and an output variable. In addition, validation set approach was implemented to test the quality of the models. The data were divided into two groups, one for training or building the model and the other for testing it.

XGBRegressor

XGBoost is an algorithm that designed as a result of implementation of gradient boosted decision trees. it is a popular supervised machine learning model with characteristics like computation speed, parallelization and performance. Because of its advantages, XGBRegressor model is defined for our structured dataset as the model requirements.[5] The parameters of the regressor are shown below:

```
#xgb regressor
xgb =XGBRegressor( booster='gbtree', colsample_bylevel=1,
            colsample_bynode=1, colsample_bytree=0.6, gamma=0,
            importance_type='gain', learning_rate=0.01, max_delta_step=0,
            max_depth=4, min_child_weight=1.5, n_estimators=2400,
            n_jobs=1, nthread=None, objective='reg:linear',
            reg_alpha=0.6, reg_lambda=0.6, scale_pos_weight=1,
            silent=None, subsample=0.8, verbosity=1)
```

LGBMRegressor (Light Gradient Boosting Machine)

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm. It is compatible with large dataset. It has a better accuracy and low memory usage. The parameters of the regressor are shown below:

```
#lgbm regressor
lgbm = LGBMRegressor(objective='regression',
                                        num_leaves=4,
                                        learning_rate=0.01,
                                        n_estimators=12000,
                                        max_bin=200,
                                        bagging_fraction=0.75,
                                        bagging_freq=5,
                                        bagging_seed=7,
                                        feature_fraction=0.4,
                                        )
```

Result

After the data is trained by using a-fore mentioned models, Mean Absolute Error (MAE), Root Mean Square Error (RMSE), R-Squared are calculated to evaluate the performance of a regression models that is implemented.

Index	mean absolute error	Root Mean Square Error	R-squared Error
XGBRegressor	15028.4	7.11508e+08	0.890085
LGBMRegressor	16043.6	27588.5	0.88242
LinearRegression	20501.4	39004.6	0.764978

Figure 6

Mean Absolute Error (MAE) refers the average error between real output value and predicted value. A value of 0 indicates a perfect fit.

Root Mean Square Error (RMSE) indicates the average error between real output value and predicted value. However, it penalizes larger errors more severely than MAE. A value of 0 indicates a perfect fit.

R-squared (R2) tells us the degree to which the model explains the variance in the data. In other words, it is how much better predict the mean. A value of 1 indicates a perfect fit. A value of 0 indicates a model no better than the mean. A value less than 0 indicates a model worse than just predicting the mean.

Conclusion

The results stated above shows that MAE and RMSE of XGBregressor close to zero than others and Its R-squared value closes to 1. It means that the model fits to our data quite well. The graphs below demonstrate model accuracy for the first 50 house in our data set. In general, model works well for the prediction problem. The accuracy and errors are quite sufficient that the data fit the model.

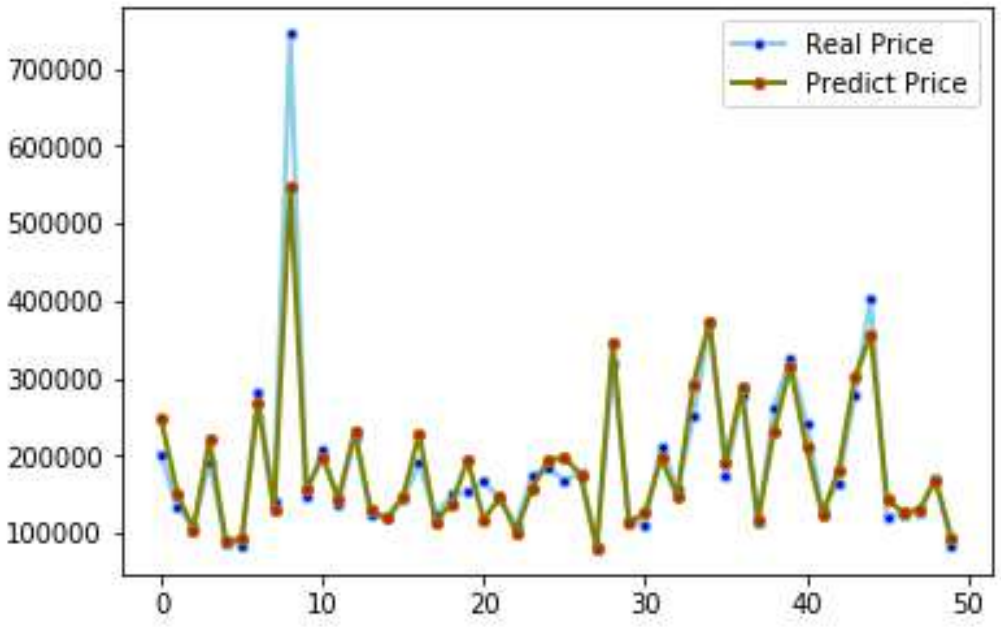

Figure 7

For future implementation, the other encoding method can be used instead of the method stated above. Features are quite sufficient but new features can be created by using existing information. Surely, more meaningful features increase the model accuracy. Furthermore, the other kind of sampling strategy may improve it.

Reference :

[1] https://medium.com/@kelly_tan/house-prices-advanced-regression-techniques-bc3054f189dc

[2] https://towardsdatascience.com/categorical-encoding-using-label-encoding-and-one-hot-encoder-911ef77fb5bd

[3] https://www.datacamp.com/community/tutorials/categorical-data

[4] https://towardsdatascience.com/basic-feature-engineering-to-reach-more-efficient-machine-learning-6294022e17a5

[5] https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/