House Price Prediction

Xiquan Jiang; Cindy Li; Bo Zhang; Xiaoran Zhu
Department of Statistics
University of California, Davis
`xiqjiang@ucdvis.edu; whali@ucdavis.edu; bzbzhang@ucdavis.edu; xrnzhu@ucdavis.edu`

Abstract

Obtaining the most accurate trends for house prices is critical for both sellers and consumers, which affects the real estate market significantly. Lots of efforts have been taken to improve the accuracy of forecasting house prices all over the world. Many machine learning algorithms, whether traditional or deep learning, have been experimented with a large amount of house price dataset and proved to be effective for performing such predictions. In this study, four machine learning models, including multiple linear regression, LASSO regression, random forest regression, and fully connected deep neural networks from deep learning, are trained on the given house prices dataset. The models are evaluated using their mean-squared errors (MSE) and the model with the best performance (smallest MSE) is identified. Furthermore, the data preprocessing pipeline and exploratory data analysis prior to constructing the models are also discussed in detail in the data analysis section.

1 Introduction

1.1 Motivation

It is shown in the research that the housing market contains a large amount of demand and supply. However, there are tons of parameters that could affect the price of housing, for example, the number of bedrooms, the number of bathrooms, the area of the yard, remodel year, etc. Thus it would always be a complicated task for the seller and buyer to negotiate the price of the house, which leads to the tendency of authorizing a real estate agency to quote and negotiate the price for both buyer and seller [1]. In this process, knowing the market and knowing the matching of the price with the house conditions would be a really important task for the buyers to lower costs, and for the sellers to maximize the profits.

1.2 Problem Statement

The main objective of this project is to train proper models for predicting the house price with relatively high accuracy. Based on the prediction of the house price, the buyer could have a better understanding of the average house price of the type of the house they are looking for, and the seller would obtain a general idea of the market price of the house they are trying to put on the market. This could bridge the information gap between the buyer or the seller and the market.

1.3 Data Description

The ‘House Prices - Advanced Regression Techniques’ dataset was retrieved from Kaggle[2] . There are 1460 observations and 81 columns which include 21 numerical variables (ex. Id, lot area, sale price, etc.) and 60 categorical variables(street, lot shape, utilities, etc.) in total. ‘Sale price’ would be the response variable, ‘Id’ will be removed from the dataset, and all other variables would be the predictor variables.

2 Project Outline

Since the task of this project is predicting house prices with multiple features, regression analysis is utilized. Regression is a widely used technique for studying the relationship between a target value (in this study, the house price) and the independent predictors. There are various regression models available for machine learning tasks and their performance differs from each other based on the dimension of feature space, type of input variables, the relationship between input and output, etc.[3] Hence, for generating the best possible prediction results, it is critical for selecting the most suitable regression models based on the dataset. After careful research and comparison, we decided to apply Multiple Linear Regression, LASSO Regression, Random Forest Regression, and Fully-Connected Deep Neural Networks to the dataset for predicting house prices.

Linear Regression is one of the most popular and light-weighted regression models. For this study, **Multiple Linear Regression (MLR)** and **LASSO Regression** are used since there are 79 independent variables in total. **Random Forest (RF) Regression** is an ensemble learning method, which generates the final prediction results by averaging the votes from a large number of decision trees. **Fully Connected Deep Neural Networks** is one of the commonly used deep learning models, which is constructed by a sequence of fully connected layers enabling every neuron in a certain layer to be connected to every neuron in the other layer.

Prior to fitting the machine learning and deep learning models mentioned above, a fair amount of effort is devoted to data cleaning and wrangling, as well as feature engineering. Tasks including but not limited to variable type identification, missing values imputation, data transformation, and feature standardization are performed, with the help of insights obtained from exploratory data analysis (EDA). Afterwards, the train-test-split is done on the train data provided by the website for the reason that there are no house prices available in the given test data.

After training and tuning the selected machine learning models and deep learning architectures on training data, predictions are made using the testing data. In this study, the mean-squared error (MSE), as well as the goodness-of-fit (R^2), are two major metrics for comparing the prediction accuracy for methods. As a result, the fully connected deep neural network tends to have the best model performance for having the lowest MSE (0.024) among all three algorithms, while multiple linear regression and LASSO become the benchmark for prediction. The random forest model achieves a moderate prediction performance.

3 Related Work

Bruno Klaus de Aquino Afonso, Et al.[1] proposed two models in their study of predicting the property values in Brazil, which are Random Forest Regression and Bidirectional LSTM (Long-Short Term Memory algorithm). Before fitting the models, they conducted data enrichment (filling the missing values for important features) and data pre-processing as preparation. In this study, RMSLE (root mean squared logarithmic error) is used as a metric for evaluating the model performance. The enriched Random Forest model turned out to be a benchmark for regression analysis, while the deep learning model gave a slightly better result. The ensemble model (RF and NN) was shown to be the best performing architecture for the data.

Yihao Chen, Runtian Xue, Yu Zhang[4] predicted house price in “House price prediction based on machine learning and deep learning methods” using Linear Regression, Bayesian, Back propagation neural network, SVM, and Deep Neural Network. The prediction was evaluated by MSE, RMSE, MAE, and R^2. It is very helpful for us to choose metrics to evaluate the performance of the model prediction. SVM can be considered the most suitable method for prediction with the R square of 0.9908.

CH.Raga Madhuri, Anuradha G, M.Vani Pujitha[5] compared Multiple Linear Regression, Ridge Regression, LASSO Regression, Elastic Net Regression, Ada Boosting Regression, and Gradient Boosting using the accuracy value of the algorithm. Different regression methods were analyzed by calculating MSE and RMSE. Based on the experiments, the gradient boosting algorithm with a high score of 0.9177.

4 Proposed Method

4.1 Multiple Linear Regression

The purpose of multiple linear regression is to fit a regression equation (1) by using many explanatory variables to estimate the dependent variable in order to explain and predict the value of the dependent variable.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \epsilon \tag{1}$$

where X_k is an explanatory variable, Y is the dependent variable, β_0 is the constant term, β_k is slope coefficient for each explanatory variable and ϵ is the residual[6].

In multiple linear regression, there are $k+1$ regression coefficients that need to be obtained using least-squares estimation. Mean-square error (2) is the expected value of the difference between the predicted and true values, which is often used to evaluate the accuracy of the predicted value.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2 \tag{2}$$

The coefficient of determination (3), known as R^2, reflects the proportion of the total variance in the dependent variable explained by the explanatory variables in the model[7]. The larger the coefficient of determination is, the better the model fits the data.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y_i} - y_i)^2}{\sum_{i=1}^{n}(\hat{y_i} - \overline{y}_i)^2} \tag{3}$$

4.2 LASSO Regression

Lasso regression performs variable selection by adding regularization term to linear regression. Variable selection refers to fitting a model with selective variables instead of putting all the variables into the model, in order to get rid of the multicollinearity and get better performance using selective parameters. Regularization is mainly used to avoid overfitting by adding a penalty term to the model. The LASSO Regression model can be built on both continuous and discrete dependent variables to do the prediction[8].

In linear regression, the cost is calculated by mean square error. It is likely for the regression to be overfitted since there is no penalty for the variables when choosing weight. However, the cost function in Lasso Regression adds the sum of the L1 norm of the weight for the penalty[9]. The cost function(4) for Lasso is:

$$\mathcal{L}(\omega) = \sum_{i=1}^{n}(y_i - \omega^T x_i)^2 + \lambda\|\omega\|_1 \tag{4}$$

where $\sum_{i=1}^{n}(y_i - \omega^T x_i)^2$ is the error term, λ is the regularization coefficient and ω is the sum of square coefficients.

In order to obtain a higher coefficient of the variable, the added regularization term needs to be minimized. There are 78 features in our study, which makes it necessary to select variables to reduce multicollinearity.

4.3 Random Forest Regression

Random Forest is an ensemble learning method, which generates the final prediction results by averaging the votes from a large number of decision trees. When building a decision tree, we randomly select a part of the samples from the training data with replacement, and randomly select features for training instead of using all the features of the data (known as bootstrap aggregation). In short, such random processes reduce the correlation among different decision trees and achieve higher stability for the entire model. Consequently, random forest regression is suitable for noisy data and not prone to overfitting [10]. RF is a relatively user-friendly model since there are only two major hyper-parameters that need to be tuned, which are the number of decision trees and the number of features per split[11].

4.4 Deep Learning and Fully Connected Neural Network

Although linear regression and random forest are relatively easy and effective to deploy, their performances are limited by the potential "curse of dimensionality" when it comes to high-dimensional data (there are 78 features in our study). In addition, complicated feature engineering and selection are needed for achieving better model performances. To overcome such deficiencies of traditional regression models, we decided to apply deep learning for prediction.

Basically, deep learning simulates the patterns of human beings' knowledge acquisitions by constructing artificial neural networks, in which neurons are the basic units. Figure 1 can be used to illustrate the structure of a neuron.

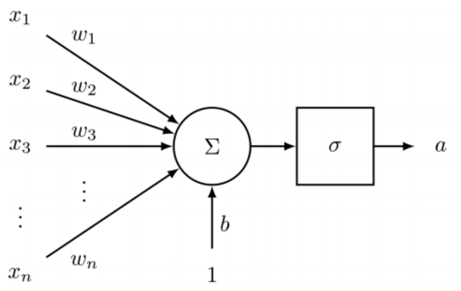

Figure 1: Neural network structure

x_0, x_1, $\ldots$, x_d are the vector of inputs, b is the bias, and ω is the corresponding weight for every input. After summing up all the weighted inputs and setting the activation function, such as sigmoid function and Rectifier neural networks (Relu), the output a can be obtained from the model. Also, more complicated neural networks can be constructed based on different add-on layers[12].

Firstly, a single layer fully connected neural network with 128 units is built. We then improve the architecture by adding a second hidden layer, with 128 units on the first layer and 32 units on the second layer. We choose Relu as the activation function, and Adam optimizer for improving the model learning performance.

We then set the batch size to 50 (50 data observations are input into the model every time) and the number of epochs to 80 for training. Predictions are made using the validation set and MSEs are calculated and stored per two epochs.

5 Data Preprocessing

Before training proper models or architectures and tuning model parameters, data preprocessing and feature engineering are extremely crucial to obtain high-performance predictive models.

5.1 Variable Type Setting

The 81 variables should be defined and separated into categorical and numerical variables. The variable 'Id' has been removed from the dataset since all observations provide unique values irrelevant to the house price. To separate variables into corresponding data types, a threshold has been set:

variables having larger than or equal to 25 levels will be set as numerical variables; categorical variables are those who have less than 25 levels. A manual check has been done for misclassified variables later on. ‘MSSubClass’(the building class), ‘MoSold’(month sold), ‘YrSold’(year sold) are reclassified into categorical variables based on the domain knowledge that the month and year in which the house has been sold would be more reasonable to be considered as categories instead of time series.

5.2 Data Visualization

The histograms (Fig. 2) indicate that the value ranges for some numerical variables, especially ‘LotArea’ and ‘EnclosedPorch’, are wide. Hence, standardization can be used to unify the magnitudes and units of the data to improve model performances, and avoid the effect of some features with extreme values in model training. Scaled data is critical for the linear regression and neural network model, but not essential for the tree based model. We choose standardization over normalization due to its capability for dealing with outliers, which is beneficial for our outlier sensitive models.

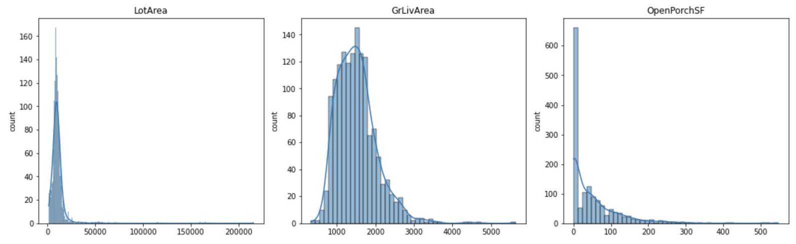

Figure 2: Histogram for area variables example

We notice that there are 13 different categorical variables related to quality including ‘ExterQual’, ‘ExterCond’ and ‘BsmtQual’. From the histograms of the quality categorical variables(Fig. 3), we see that most of the house quality levels are ‘TA’ (typical or average).

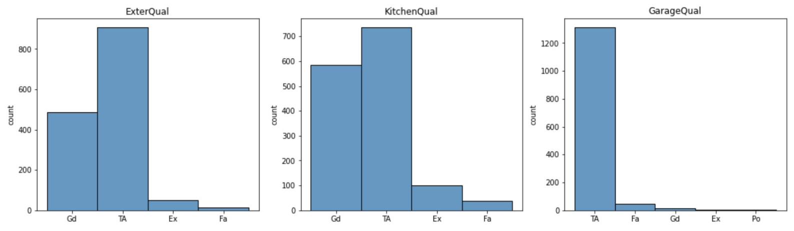

Figure 3: Histogram for quality variables example

5.3 Data Transformation (Response Variable)

To fit the regression models, the model assumption for the response variable should be satisfied. The histogram of the response variable ‘sale price’ indicates that it is not in normal distribution. Thus, a $log(1 + x)$ transformation has been applied to the response variable. After the log transformation, the response variable is approximately normally distributed as shown in the histogram and QQ-plot.

5.4 Missing Values

After summing up the number of missing values for each variable, we see that there are 19 variables containing missing values. The variable ‘poolqc’ (the quality of the pool belonging to the house) contains 1453 missing values, which takes the largest portion among all the missing values. The variable ‘electrical’ (the Electrical system provided in the house) contains the least number of missing values, which is 1.

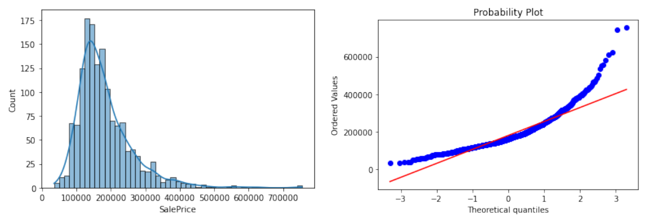

Figure 4: Histogram and QQ-plot for response variable before transformation

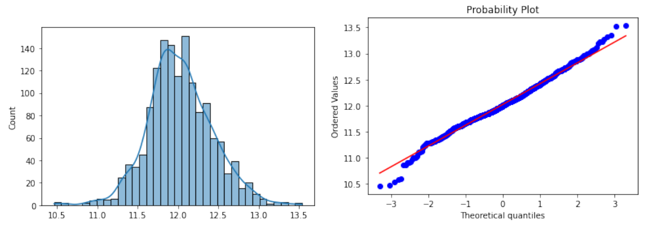

Figure 5: Histogram and QQ-plot for response variable after transformation

We use different imputation methods for filling those missing values. The mode has been filled for the ‘electrical’ variable. However, for the ‘utilities’ (utility type of the house) variable, there are only two types of utilities: ‘AllPub’, and ‘NoSeWa’ having only one observation. Therefore, ‘utillities’ variable has been removed from the dataset. The missing values in ’MasVnrArea’ and ’GarageYrBlt’, which are Masonry veneer area in square feet, and Year garage was built, are filled by 0. For those categorical variables, such as ’MasVnrType’, ’BsmtQual’, ’BsmtCond’, ’BsmtExposure’, ’BsmtFinType1’, ’BsmtFinType2’, ’GarageType’, ’GarageFinish’, ’GarageQual’, ’GarageCond’, ’FireplaceQu’, ’Fence’, ’Alley’, ’MiscFeature’, and ’PoolQC’, missing values are filled in ‘None’ for future analysis. As for the variable ‘Lotfrontage’, which is the linear feet of a street connected to property, the median of the lot frontage group by the neighborhood has been used for filling, based on the knowledge that the street connected to property should have generally similar length by the regulation of the society.

5.5 Feature Encoding

We used label encoding and one-hot encoding for the categorical variables. We applied label encodings for 26 variables associated with quality or conditions, such as ’FireplaceQu’ (fireplace quality), ’BsmtQual’ (basement quality). Since these variables have ordered rankings as their factor levels, label encoding converts those levels into corresponding numbered rankings. Otherwise, one-hot encoding is done for variables having no internal logic within their factor levels. However, one-hot encoding would lead to a dimensional explosion for high dimensional variables. Hence, high-dimensional variables, such as ’FireplaceQu’, ’BsmtQual’, ’BsmtCond’, ’GarageQual’, ’GarageCond’, ’ExterQual’, ’ExterCond’, ’HeatingQC’, ’PoolQC’, ’KitchenQual’, ’BsmtFinType1’, ’BsmtFinType2’, ’Functional’, ’Fence’, ’BsmtExposure’, ’GarageFinish’, ’LotShape’, ’LandSlope’, ’PavedDrive’, ’Street’, ’Alley’, ’MSSubClass’, ’MoSold’, ’YrSold’, ’CentralAir’, ’OverallCond’ are encoded with label encoder. All the rest of categorical variables are encoded with one-hot encoding[13].

5.6 Feature Standardization

As for numerical predictors, to fulfill the regression model assumptions, Box-Cox transformation has been made to transform numerical variables to present approximately normal distribution. By using Box-Cox transformation. It will automatically return the value of lambda used for the below functions that provides the best approximation for the normal distribution by transform the predictors[14].

$$y(\lambda) = \begin{cases} \dfrac{y^\lambda - 1}{\lambda}, & \lambda \neq 0, \\ \log y, & \lambda = 0. \end{cases} \quad (5)$$

5.7 Train-Test Split

Since there are no house prices in the original test data set, only the train data set will be used for analysis. In order to fit the dataset into the proposed methods, a train-test split (with 80% being train data, and 20% being the test data) is done on the original train data set. The models are fitted on our new train set from split and predictions are made using the new test set.

6 Model Results

After fitting all four models on the pre-processed training data and making predictions with the test data, the Mean Squared Error (MSEs) and the Coefficient of determination (R^2) are obtained and displayed in the following table.

Table 1: R^2 and MSE for different model

Model	MSE	R^2
Linear Regression	420.538	0.887
LASSO Regression	118.741	0.506
Random Forest	0.057	0.684
Fully Connected NN (1 layer)	0.031	
Fully Connected NN (2 layers)	0.025	

We can see that the MSEs output from multiple linear regression is pretty high, which means the prediction accuracy for the model is not ideal. The R^2 calculated from multiple linear regression model is relatively high, proving that there are strong relationships between independent variables and the dependent variable house price.

After applying regularization in LASSO to reduce the effect of overfitting, only eight features are kept (weights not being set to 0) and the prediction accuracy increases (MSE is smaller). However, the R^2 drops drastically since the majority of features are dropped while fitting the model. Thus, we conclude that the performance of LASSO is not good on our data set either.

Then we can tell there is a decent amount of improvement in model performance for random forest based on the fact that the MSE returned is very small. However, the goodness-of-fit for our random forest model is still low.

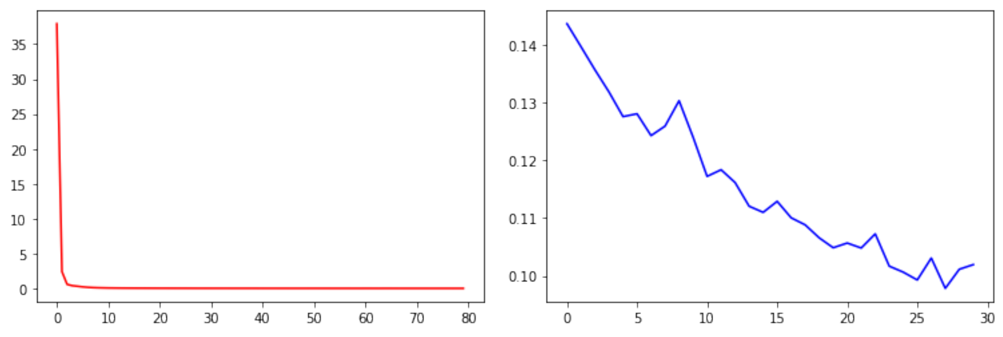

Figure 6: Neural network train(left) and test(right, 10-40 epoch) MSE (1 layers)

For the neural network structure, based on the results, while the MSE for the training set is still decreasing, the MSE for the testing set starts to increase after running certain epochs (Fig. 6). We conclude that more noise might be added to the model if we keep increasing the number of epochs. To avoid potential overfitting, we stop the training after the 52^{th} epoch for the single-layer model and 8^{th} epoch for the 2-layer model.

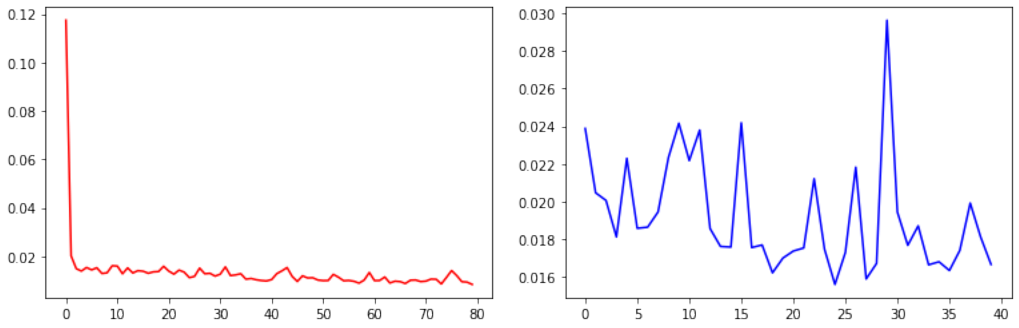

Figure 7: Neural network train(left) and test(right, all epoch) MSE (2 layers)

The single-layer NN gives us a much smaller MSE compared to the traditional regression models. After implementing a second hidden layer to the neural network, the model performance is further improved based on the 0.0248 MSE.

7 Conclusions and Discussions

From the evaluation metrics table above, we could conclude that the Fully Connected Neural Network with two hidden layers has the lowest MSE, hence generating the best performance on our dataset. Currently, the architecture of or Neural Network is not fully optimized. For future improvement, a dropout layer could be added to the neural network structure in order to eliminate the influence of overfitting brought by the fully connected layer. Besides, we could also try convolutional neural networks or recurrent neural networks to eliminate the influence of over-parameterized or gradient diffusion [15].

With our neural network model, house prices could be predicted with the sellers' or buyers' desired characteristics as input features. Provided the performance of our model, the predicted price could be close to the fair market values. In this way, the buyer will have little chance to bargain, while the seller will be less likely to overcharge. It will also prevent real estate agencies from jacking up the house prices, which further helps reduce the loss from asymmetric information exchange. Ultimately, a clean and fair house trading market will be established and price negotiations will be bounded by the prediction results.

References

[1] Bruno Afonso, Luckeciano Melo, Willian Oliveira, Samuel Sousa, and Lilian Berton. Housing prices prediction with a deep learning and random forest ensemble. In *Anais do XVI Encontro Nacional de Inteligência Artificial e Computacional*, pages 389–400. SBC, 2019.

[2] House prices - advanced regression techniques.

[3] Rohith Gandhi. Introduction to machine learning algorithms: Linear regression.

[4] Yihao Chen, Runtian Xue, and Yu Zhang. House price prediction based on machine learning and deep learning methods. In *2021 International Conference on Electronic Information Engineering and Computer Science (EIECS)*, pages 699–702, 2021.

[5] CH. Raga Madhuri, G. Anuradha, and M. Vani Pujitha. House price prediction using regression techniques: A comparative study. In *2019 International Conference on Smart Structures and Systems (ICSSS)*, pages 1–5, 2019.

[6] Adam Hayes. Multiple linear regression (mlr) definition.

[7] Andrew Bloomenthal. How the coefficient of determination works.

[8] Great Learning Team. A complete understanding of lasso regression.

[9] Tavish. Aggarwal. Lasso and ridge regression detailed explanation.

[10] Yi Li, Changfu Zou, Maitane Berecibar, Elise Nanini-Maury, Jonathan C.-W. Chan, Peter van den Bossche, Joeri Van Mierlo, and Noshin Omar. Random forest regression for online capacity estimation of lithium-ion batteries. *Applied Energy*, 232:197–210, 2018.

[11] Chaya Bakshi. Random forest regression.

[12] IBM Cloud Education. What are neural networks?

[13] Alakh Sethi. Categorical encoding: One hot encoding vs label encoding.

[14] Andrew Plummer. Box-cox transformation: Explained.

[15] Yu Xue, Yankang Wang, and Jiayu Liang. A self-adaptive gradient descent search algorithm for fully-connected neural networks. *Neurocomputing*, 478:70–80, 2022.

8 Appendix

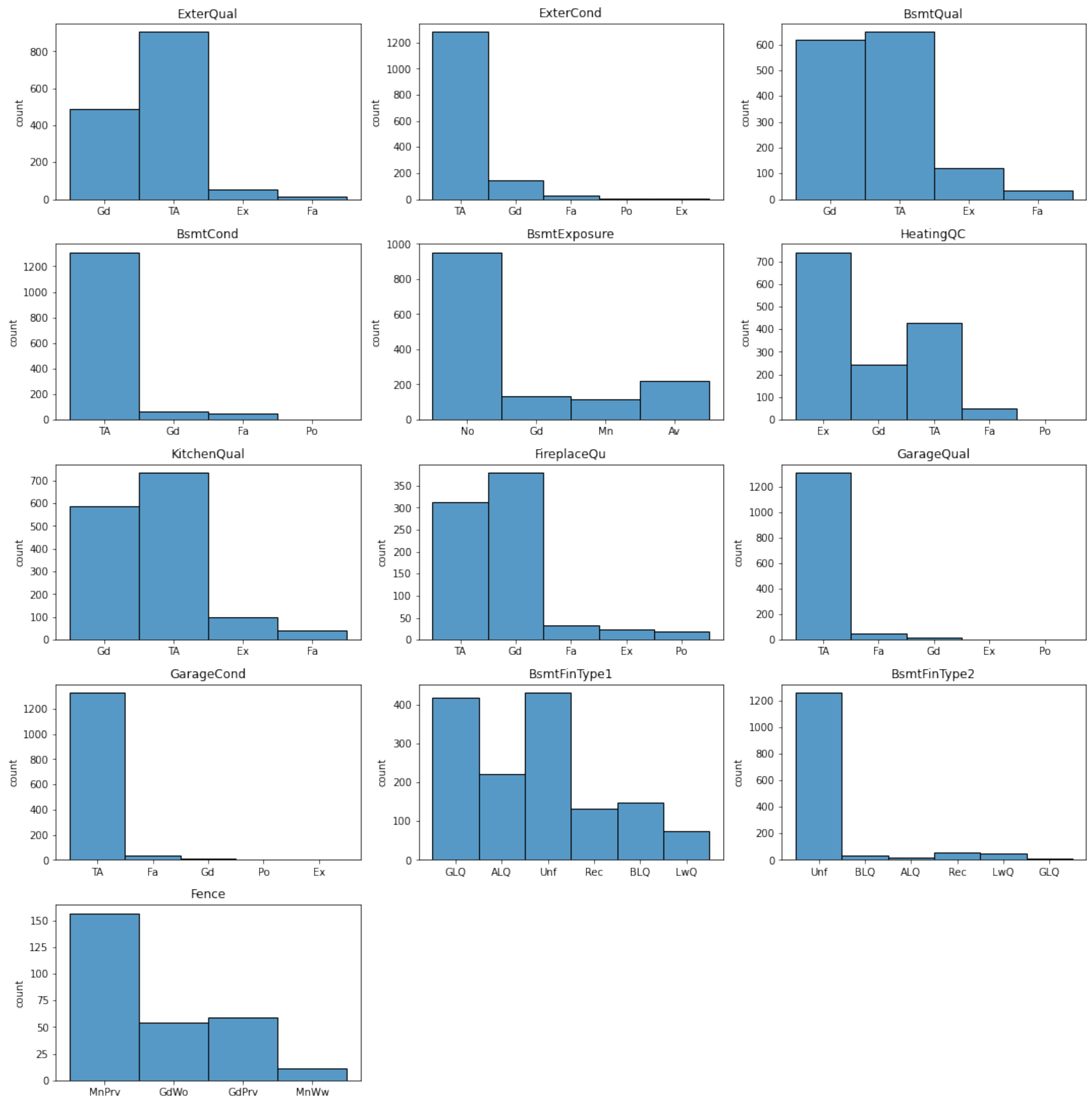

Figure 8: Histogram for quality variables

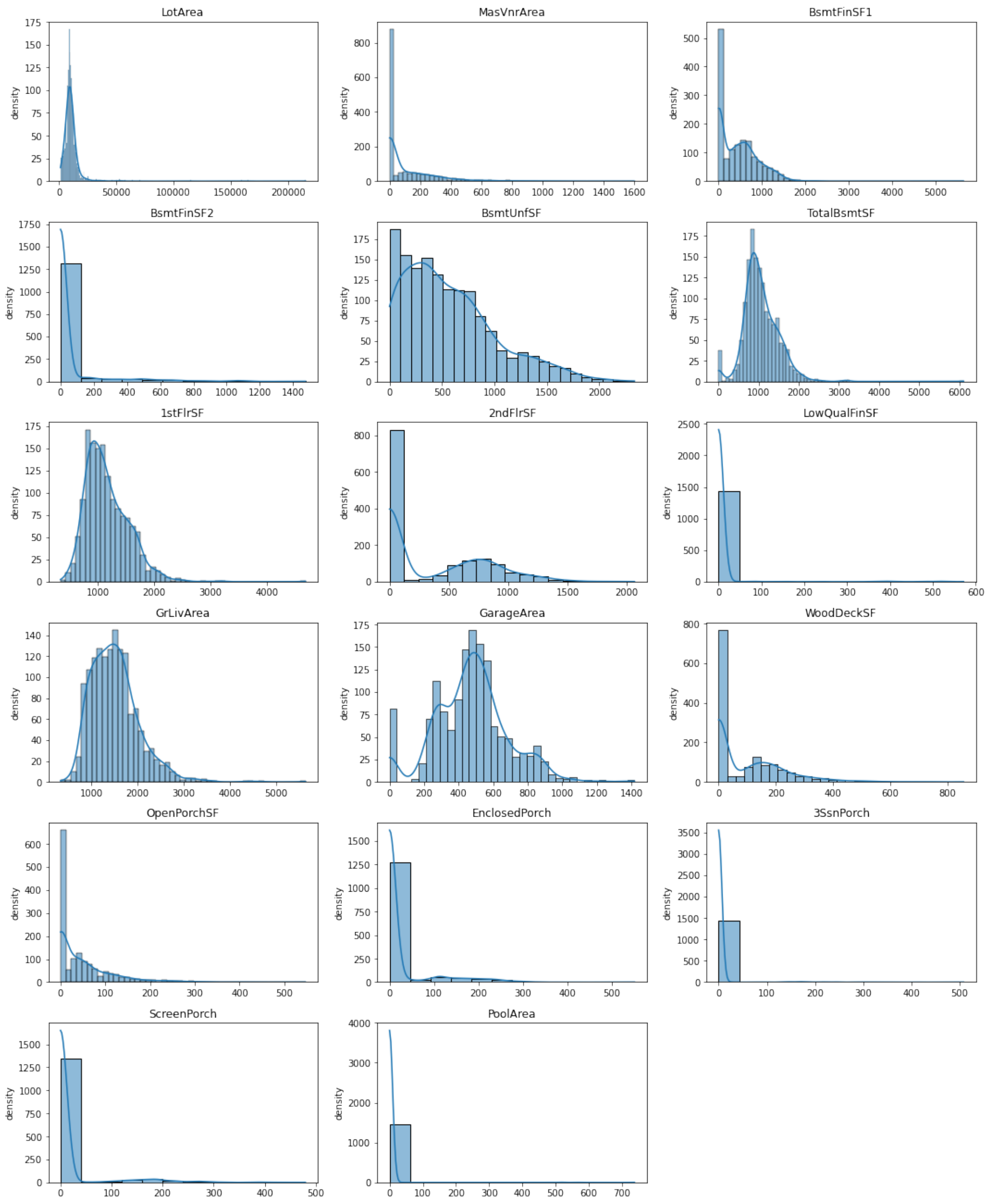

Figure 9: Histogram for area variables

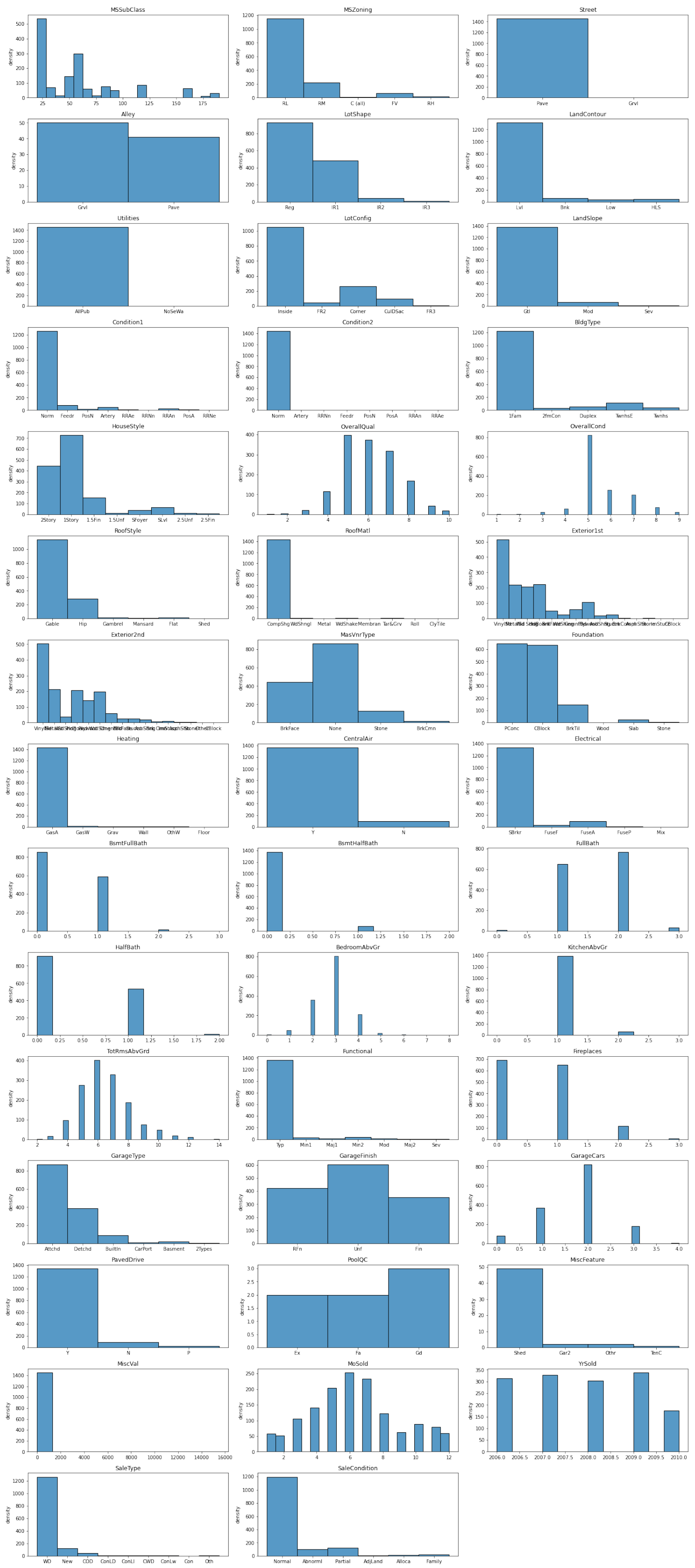

Figure 10: Histogram for categorical variables except qualities

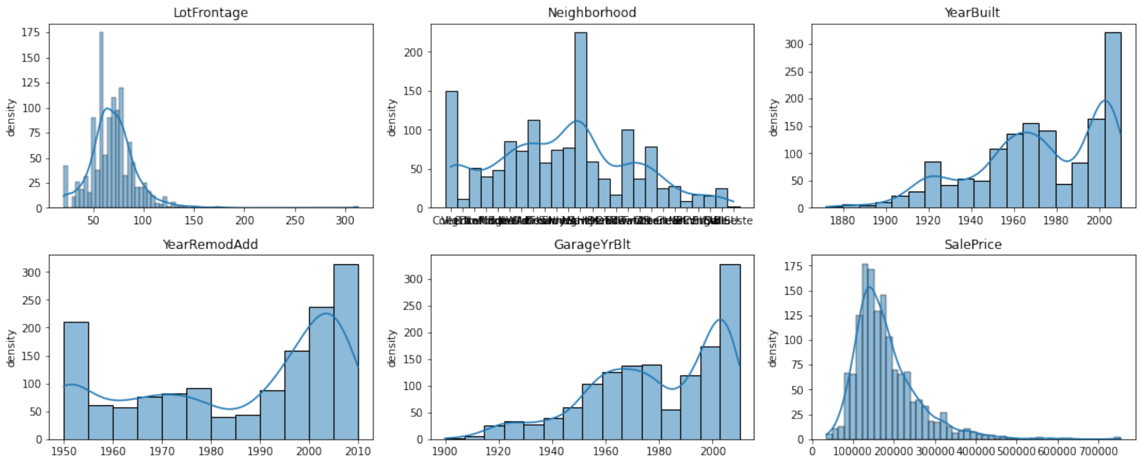

Figure 11: Histogram for numerical variables except areas