# ***Modeling***

In order to predict a value for the target variable, assessland, five models were used. The process of tuning the models is described below, while model comparison and results are presented below. Used algorithms are as follows:

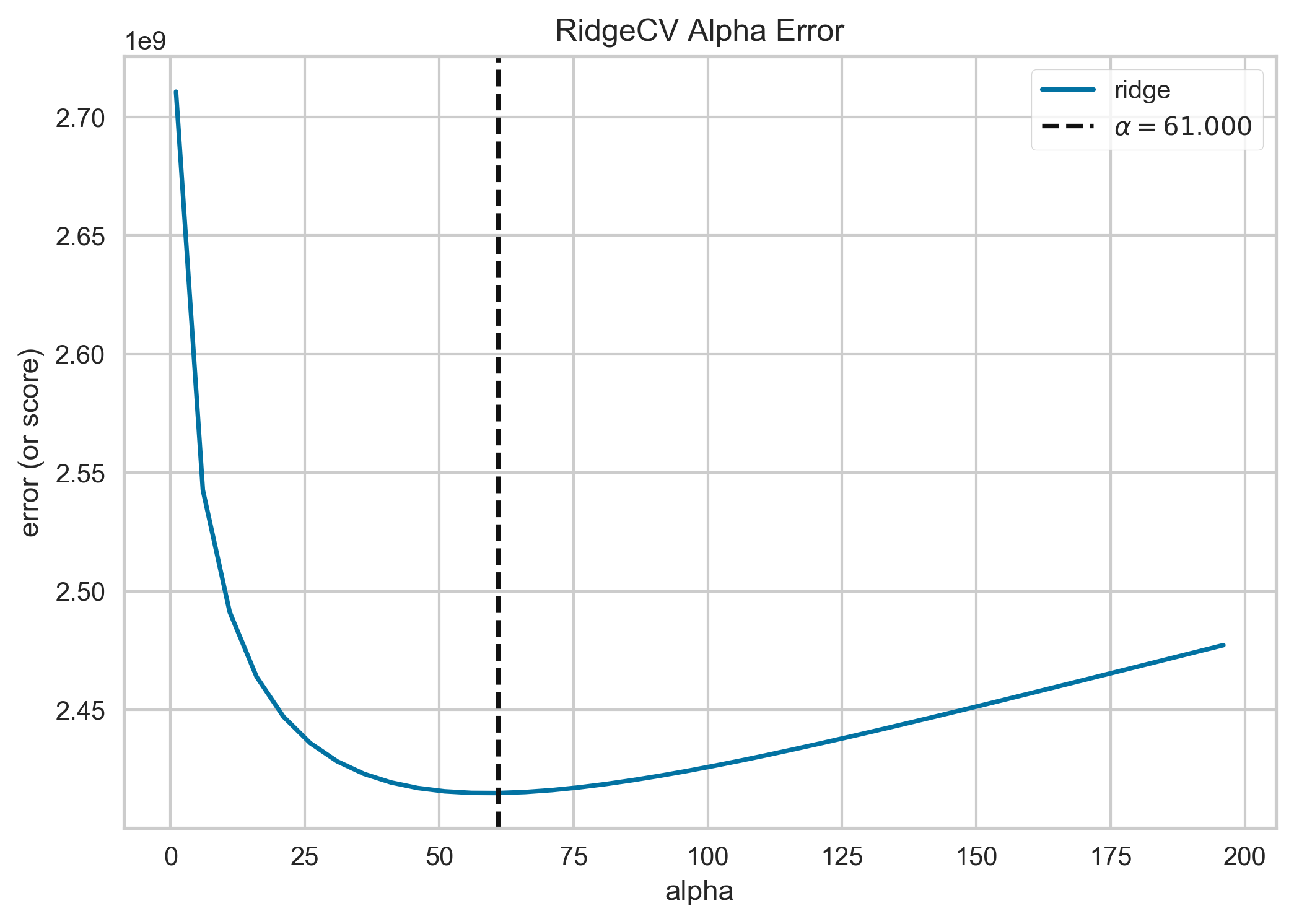
* Ridge Regression
* Lasso Regression
* KNN
* Random Forest
* Neural Network

At the very beginning the simplest regression techniques were tested. Thus Ridge Regression and Lasso Regression models will be presented first.

* Ridge Regression

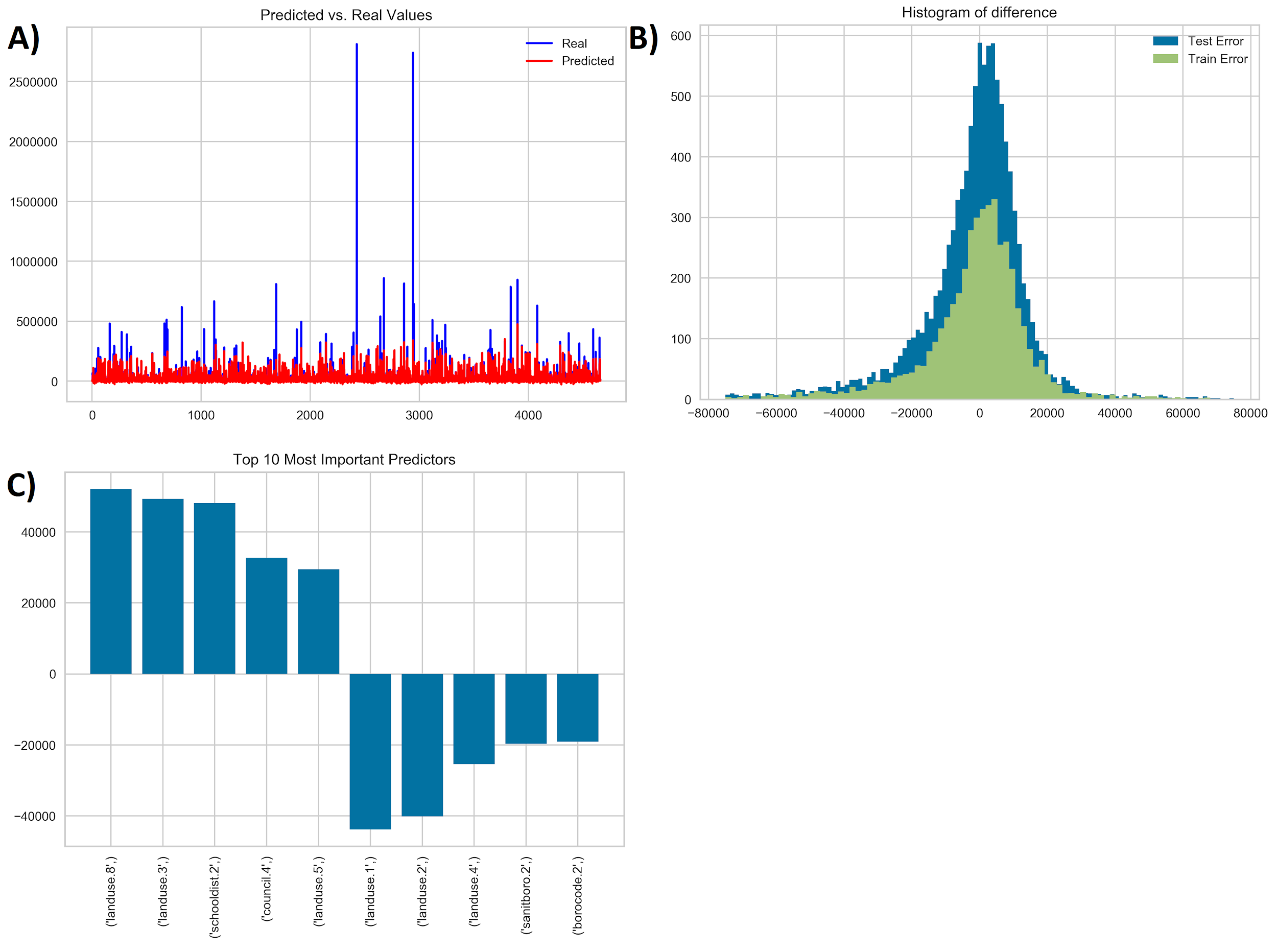
a) Tuning the model

In order to find the best parameter alpha, which describes the penalty level for variables which account for a low prediction power, the Cross Validation technique were used. On the Figure below can be seen that while alpha grows up the error is going down. That means that penalty level for predictors should be relatively high and the cross validation yields the best value for alpha equal to 61.



b) Model Performance

In order to better understand the model behavior several plots and measurements are presented below.



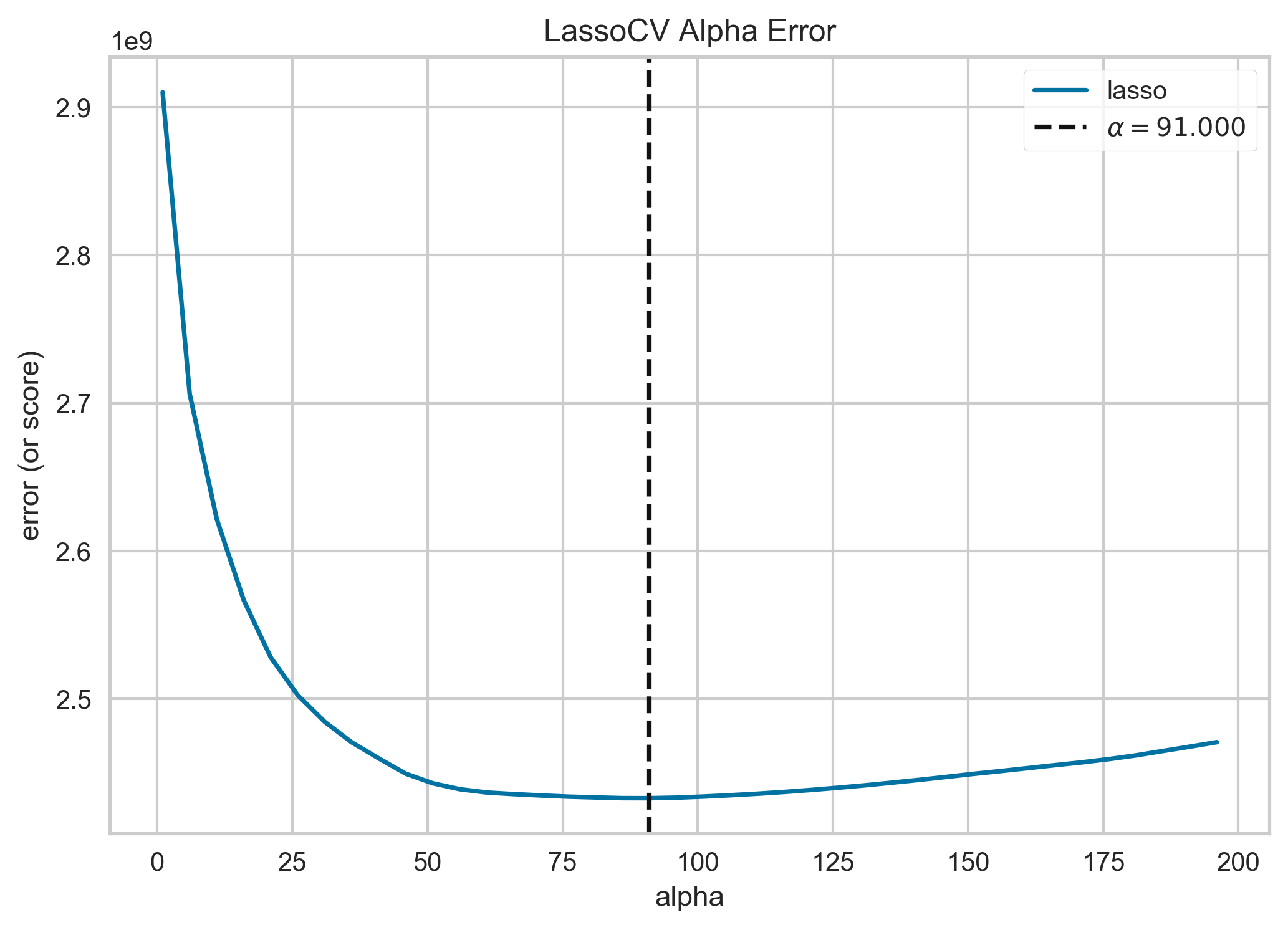
The graph (A) compares true value of assessland and predicted one. As we can see blue lines represent real value of the land while red values describe predicted value. The simple regression model performs quite good when it comes to predict small values of the prices, however when it comes to higher values in general Ridge Regression underestimate assessland.

Another interesting measure is histogram of difference (B). For both training and testing set the difference between true value and predicted value of assessland are on the graph.The plot tells us that on average differences are around zero as well as differences in training and testing sets have similar distribution, which means that there is no overfitting problem in this case. The issue in this case is that, while most of the errors are around zero, in some cases the model makes big mistakes, however they are rather rare.

The advantage of regression is that for each predictor used in the model, it is estimated its coefficient. In the graph (C), the 10 predictors, both with positive and negative coefficients are presented. The variables which presence increases the price of the land are, for example, Landuse 8, 3 and 5. According to the data documentation those lands refers to: Public Facilities & Institutions, Multi-Family Elevator Buildings and Commercial & Office Buildings. However, variables which presence decrease the land value are for example, Landuse 1, 2 and 4. Those variables represents: One & Two Family Buildings, Multi-Family Walk-Up Buildings, Mixed Residential & Commercial Buildings. As a result we can conclude that lands which are related to public institutions, lands with tall buildings where elevator is needed as well as lads where offices are placed are higher priced.

* Lasso Regression

The idea of the Lasso Regression is very similar to Ridge Regression, however the first one penalize predictors more harshly. When the variables is not useful as a predictor its coefficient will be reduced to zero. Because the data presented in this project contain more than thousands of dummy variables it is reasonable to use Lasso Regression.



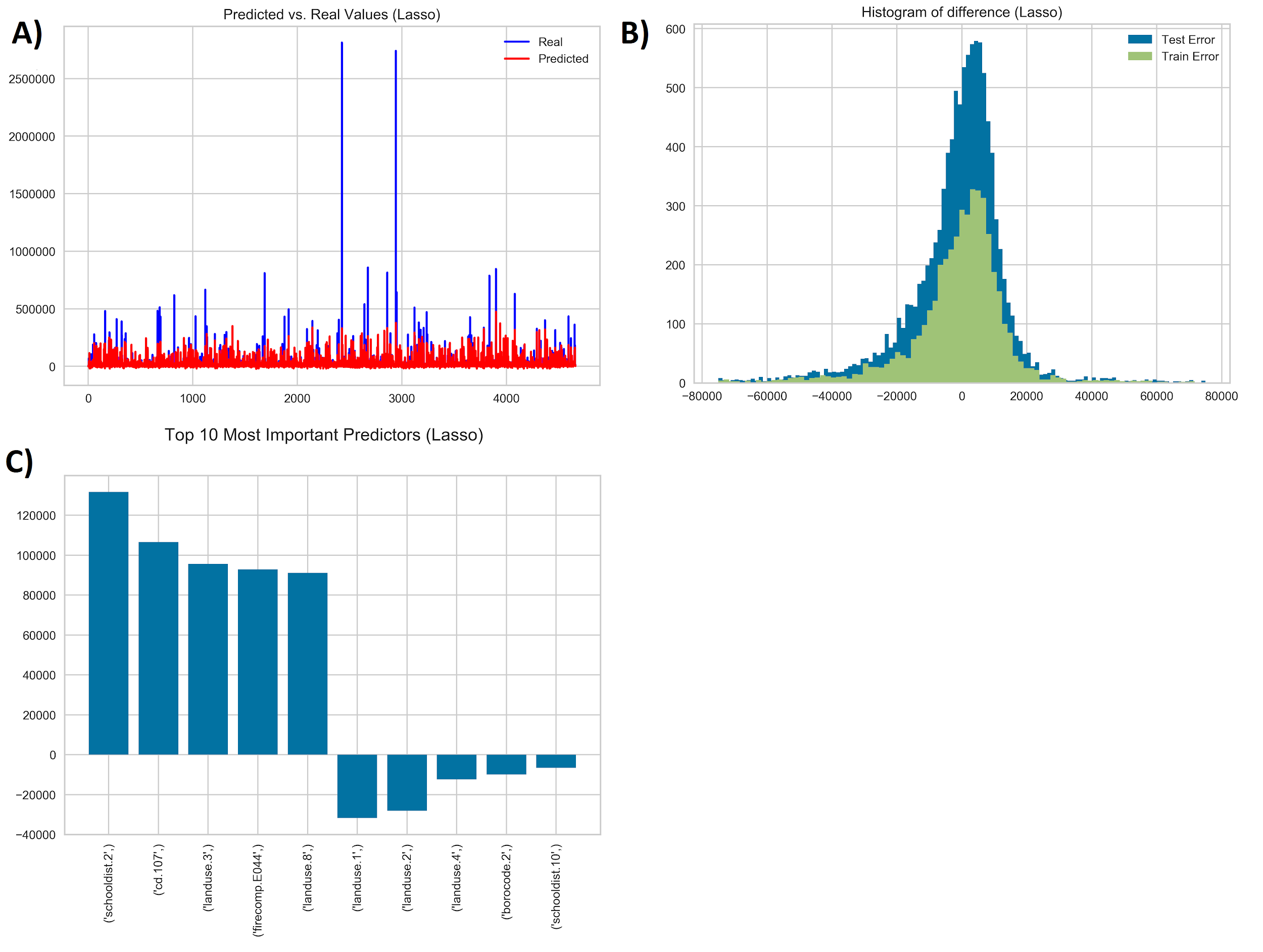
a) Tuning the model

Similarly to the previous Regression problem, the alpha value was determined by using Cross Validation technique. The graph is presented below.

The process of cross validating the data decided optimal alpha to be 91, thus this value was utilized in the model.

b) Model Performance

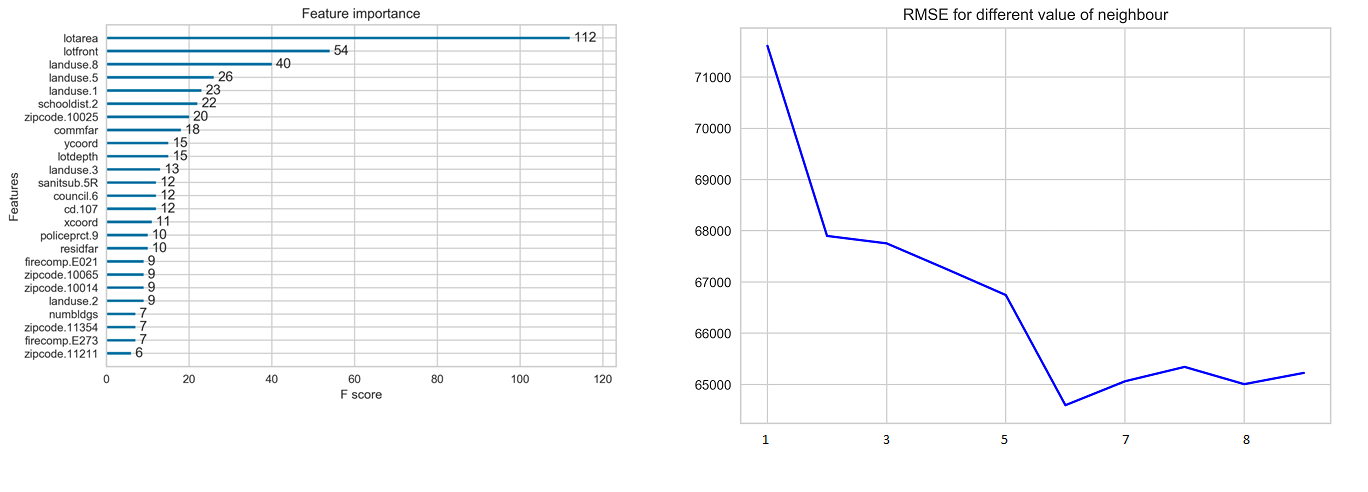
While comparing the prediction with real values of assessland the results are very similar to Ridge Regression. The model has difficulties to predicts values of assessland which are high than most of the data (A). From the histogram of difference (B) It can be intereferd that model does not overfit the data and most of the errors are placed around zero. The coefficients values displayed on plot (C) are very similar to those for Ridge Regression. Additionally, Lasso regression penalized 1027 variables and reduce their coefficient to zero.



* KNN

1. Model Tuning

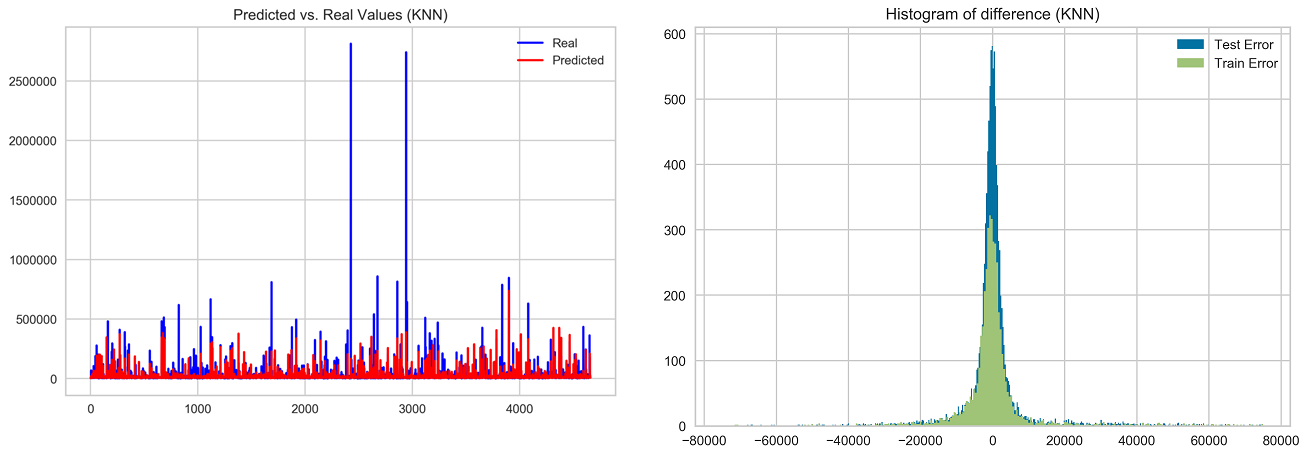
The algorithm use euclidean distance formula to find the most similar row for prediction. In the contrast to previous algorithms KNN does not have implemented any feature selection options. Thus, in order to improve model performance, gradient boosting - XGBoost technique were used to determine the best predictors. Additionally, in order to get the best result of prediction model was tested on different number of neighbours.



Graph on the left shows top 20 the most important variables for prediction used in KNN. As we can see among selected variables are also those which were chosen by Ridge and Lasso regression, for example, Landuse 8,5 and 1. The XGBoost algorithm choose lotarea as the most important variable for predicting price of land. Graph on the right side shows that the lowest error is presented when number of neighbours are equal to 6.

b) Model Performance

Histogram of difference shows that distribution of difference of error for both train and test are very similar. However, the graph on the left shows that predicted values do not align well with real values.



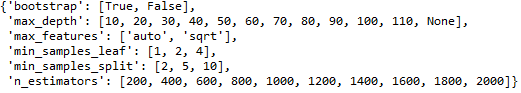
* Random Forest

<https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74>

<https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html>

1. Model Tuning

In order to determine the best parameters for the Random Forest cross-validated search over parameter settings were used. Below the list of different setting for each parameter is presented.



It can be seen above that there are 32 different parameters to check, so in combination there are 1024 different possibilities to check.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **RMSE train** | **RMSE test** | **R^2 value train** | **R^2 value test** | **Parameters** |
| **Ridge Regression** | 39028 | 62829 | 0.57 | 0.37 |  |
| **Lasso Regression** | 39508 | 62622 | 0.56 | 0.37 |  |
| **KNN** | 36621 | 64593 | 0.62 | 0.33 |  |
| **Random Forest Tree** |  |  |  |  |  |
| **Neural Network** |  |  |  |  |  |

* *Choice of models/algorithms to compare: regression, logistic regression, decision trees, neural nets, SVM, etc.*
* *Software platform(s) used.*
  + R was used primarily to merge and clean the data
    - Dropping unnecessary columns
  + Python was used primarily for modeling.
    - Create dummy variables, Logistic Regression, Neural Nets, etc.
* *Specific algorithms, parameter settings*
* *Test Design*
  + *Cross-validation of resulting models and to guard against overfitting.*
* *Comparative performance (see Blackboard folder on model evaluation metrics)*
  + *R-squared, RMSE, Std Error, MAE, Confusion matrix, true-positive and false-positive rates, precision, recall, accuracy, F-measure, etc.*
    - There are many measures to evaluate the models, such as: MAE, Mean Error, MPE, MAPE, and RMSE. All these measures are utilized in this project to compare models. However, the most value is associated with RMSE, on where the final decisions have been made. This measure checks how accurate the model predicts the response variable and the value has the same units as the outcome variable, in this case dollars. Hence, using RMSE we can evaluate the error in terms of dollars, which is the most important measurement for business needs.
  + *ROC/AUC analysis*
  + *Model tuning and optimizing*
* *Compare to best practices from literature review*
* *Tradeoffs between complexity, simplicity, performance, comprehensibility*
  + *(see Blackboard folder on Communication and Comprehensibility) Tables comparing alternative algorithms regarding performance metrics. Tables of relative importance of features.*
* *Graphical depictions of models and results*
* *Ensembles, boosting, bagging, random forests*
* *Flow diagrams of analytical modeling sequences*
  + See Appendix (Need to agree and complete before inputting into appendix)