House Prices: Advanced Regression Techniques

Sai Anirudh Reddy Avilala
Department of Computer Science
University of Rochester
Rochester, NY.
savilala@ur.rochester.edu

ABSTRACT

Housing prices are indicative of the economy of the locality or city. In this project we predict the final sale price of the houses based on the explanatory variables describing the house. This project is a regression task where we predict the continuous sale price of the house. Several regression techniques like decision trees, random forests, regularization models, gradient boosting regressor, artificial neural networks and linear regression have been used for predicting the sale prices. The goal of the project is to create a regression model that predicts the final sale price of the house given the features with the least error.

1 INTRODUCTION

For many a house is a peaceful, supportive environment Having a house. With the growth of the housing industry there has been an increasing interest from both the buyers and sellers. The notion of having an own home has been growing steadily. In this project we predict the final sale price of the houses using the Ames, Iowa dataset obtained from the Kaggle website. The goal of the project is to create a regression model that predicts the final sale price of the house given the features with the least error. The task at hand is to predict the final sale price of the houses based on the given self-explanatory features describing almost all possible characteristics of a typical house. As the task is to predict the continuous target variable of sale price, regression techniques have been used. The regression analysis is a set of statistical processes for estimating the relationships among the given variables. A few of the regression techniques like decision trees, random forests, regularization models, gradient boosting regressor, artificial neural networks and linear regression have been used for predicting the sale prices. The whole project has been written in python programming language. All the prediction models in this project have been implemented using the sklearn[1] package.

2 DATA

2.1 The Data Set

The original dataset is the Ames, Iowa Housing Dataset[2] compiled by Professor Dean De Cock, Department of Statistics, Truman State University, for use in data science education. The dataset contains 2390 observations with around 79 attributes. This dataset was amassed as an alternative to the outdated Boston Housing Dataset, the Ames, Iowa Housing dataset[2] is a modernized and expanded version. The dataset used in this project is obtained from the Kaggle website. The data set is split into two csv files, train and test. The train dataset consists of the data of 1460 houses while the test data set consists data of 1459 houses but without the target variable. There are 36 numerical and 43 categorical attributes. There are a few ordinal attributes among the categorical variables.

2.2 Missing Values in the Data Set

The data set in total, i.e., the train and test data sets altogether contain 34 attributes with missing values. The missing value count among the attributes can be seen in the figure 1.

Missing Values

	Total	Percent
PoolQC	2909	0.996574
MiscFeature	2814	0.964029
Alley	2721	0.932169
Fence	2348	0.804385
FireplaceQu	1420	0.486468
LotFrontage	486	0.166495
GarageCond	159	0.054471
GarageQual	159	0.054471
GarageYrBlt	159	0.054471
GarageFinish	159	0.054471
GarageType	157	0.053786
BsmtCond	82	0.028092
BsmtExposure	82	0.028092
BsmtQual	81	0.027749
BsmtFinType2	80	0.027407

Figure 1: **The missing value counts and proportions of the top 15 attributes with missing values.**

Among these attributes there are 16 attributes whose values have been misrepresented as missing values. For example, the attribute PoolQC which describes the quality of the pool in the house contains 2909 observations as missing values which make upto 99.65% of the total values in PoolQC. According to the data description, a value of 'NA' is assigned to the PoolQC attribute if there is no pool in the house. This 'NA' value has taken to be a missing value in the data set. There are 15 more attributes including, PoolQC, MiscFeature, Alley, Fence, FireplaceQu, etc. All the missing values in these attributes relating to the missing quality/property in the house have all been replace with a string value 'None', hereby reducing the count of the missing values.

After converting the misrepresented missing values of the attributes, we are left with 18 columns in total with actual missing values. There are in total 670 missing values. The distribution can be seen in figure 2.

Actual Missing Values

	Total	Percent
LotFrontage	486	0.166495
GarageYrBlt	159	0.054471
MSZoning	4	0.001370
Functional	2	0.000685
BsmtHalfBath	2	0.000685
BsmtFullBath	2	0.000685
Utilities	2	0.000685
BsmtFinSF1	1	0.000343
GarageCond	1	0.000343
Exterior1st	1	0.000343
Exterior2nd	1	0.000343
GarageFinish	1	0.000343
MasVnrType	1	0.000343
GarageQual	1	0.000343
KitchenQual	1	0.000343

Figure 2: The actual missing value counts and proportions of the top 15 attributes with missing values.

3 DATA PREPROCESSINGRESULTS AND DISCUSSION

3.1 Handling Missing Values

The given dataset like most of the real world data contains missing values. Missing values are usually contained in real world datasets, which cause errors or inefficiency in a lot of machine learning algorithms. There are several ways to handle to deal with these. Being explicit and thoughtful about how we handle missing values will get us the very best results.

The different ways to handle the missing values include deleting the rows containing missing values, Imputing the missing values, using prediction algorithms to fill in the missing values, etc. In this project to avoid information loss by deleting the rows containing the missing values, the Missing value Imputation has been chosen. The numeric attributes with the missing values are filled in with the respective column median and the categorical attributes with the missing values are filled in with the most frequent value of the respective column. A total of 670 missing values have been replaced with either median or the most frequently occurring value in column.

3.2 Handling Outliers

An outlier is an observation point that is distant from other observations. An outlier may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the data set. An outlier can cause serious problems in statistical analyses. Outliers, being the most extreme observations, may include the sample maximum or sample minimum, or both, depending on whether they are extremely high or low. However, the sample maximum and minimum are not always outliers because they may not be unusually far from other observations.

The outliers in the data can influence the prediction model and reduce the efficiency of the model. For example, consider the below plot in figure 3 of the between the Above ground living area in square feet (GrLivArea) and the Final Sale Price of the house, this plot has been plotted using the matplotlib[4] package.

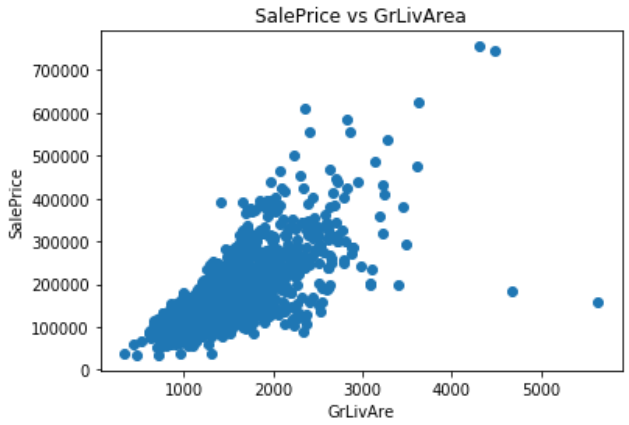


Figure 3: Scatter plot SalePrice vs GrLivArea

In the above figure it can be seen that there are two points with high the Above ground living area in square feet value but a low Sale Price value. These outliers will cause the prediction model to deviate from the true prediction line. These outlier values were removed from the data.

3.3 Multicollinearity

In regression, multicollinearity[5] refers to predictors that are correlated with other predictors. Multicollinearity occurs when the model includes multiple factors that are correlated not just to the target variable, but also to each other. In other words, it results when we have factors that are a bit redundant. Multicollinearity increases the standard errors of the coefficients. Increased standard errors in turn means that coefficients for some independent variables may be found not to be significantly different from 0. In other words, by overinflating the standard errors, multicollinearity makes some variables statistically insignificant when they should be significant. Without multicollinearity (and thus, with lower standard errors), those coefficients might be significant.

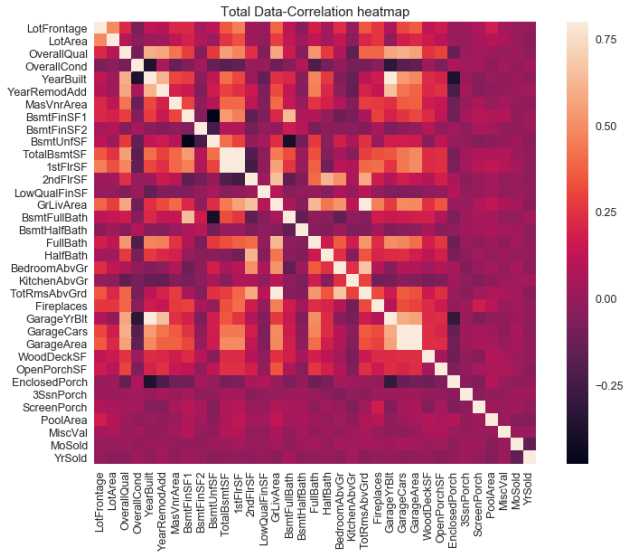


Figure 4: Correlation matrix plot of the combined data set

The multicollinearity in the data set can be seen through the correlation matrix plots, it is shown in the figure 4. The correlation plots provide a greater insight towards multicollinearity.

It can be seen in the above figure a few attributes have high correlation with other feature attributes. High correlation can be seen between the attributes, YearBuilt & GarageYrBlt, TotalBsmtSF & 1stFlrSF, GrLivArea & TotRmsAbvGrd, GarageCars & GarageArea. This high correlation among the feature attributes results in multicollinearity.

The correlation between the feature attributes and the target variable can be seen in figure 5. It can be seen in the figure the top 9 correlated feature attributes. The highly intercorrelated feature attributes are taken and the on among the two with lower correlation with the target variable has been removed. This practice is aimed at reducing the multicollinearity.

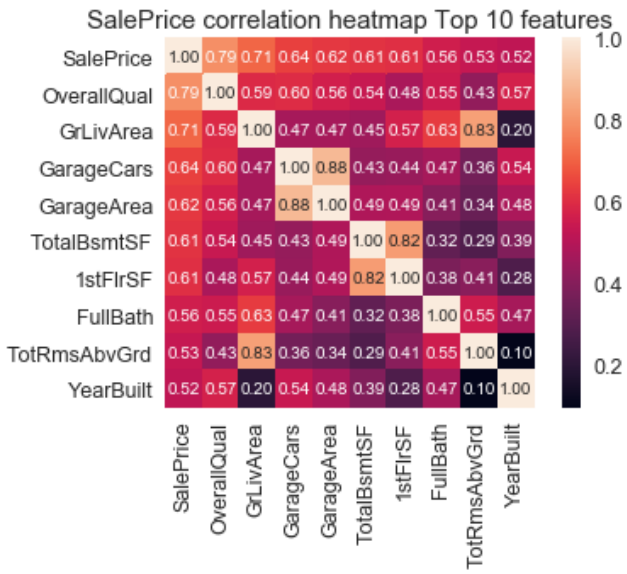
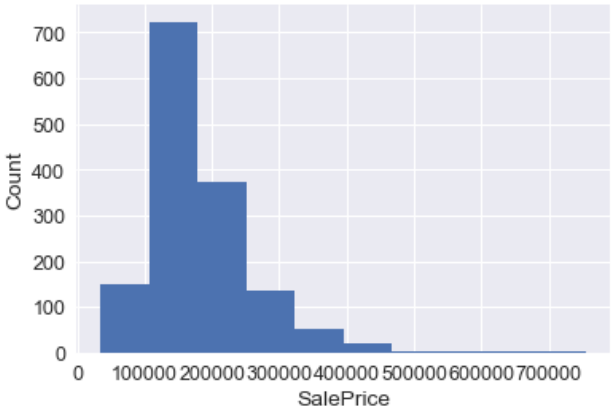


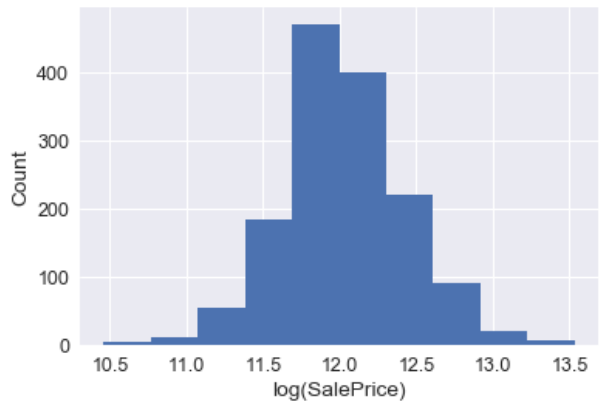
Figure 5: Correlation matrix plot of the combined data set

3.4 Normalizing Skewed Attributes

Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. The skewness value can be positive or negative, or undefined. Skewness is asymmetry in a statistical distribution, in which the curve appears distorted or skewed either to the left or to the right. Skewness can be quantified to define the extent to which a distribution differs from a normal distribution. One of the method to deal with skeweness, is by applying log transformations. The log transformation can be used to make highly skewed distributions less skewed. This can be valuable both for making patterns in the data more interpretable and for helping to meet the assumptions of inferential statistics. Log transformations increase the linearity of the distribution and decrease the variability in the data



(a)



(b)

Figure 6: (a) The original target variable distribution. (b) The log transformed target variable distribution

It can be seen in figure 6 (a) the target variable SalePrice is right skewed. The figure 6 (b) shows the log transformed version of the target variable Sale Price, it can be seen to have a more normal distribution.
The remaining feature attributes have also been transformed based on the skewness. The skewness is calculated from the scipy[3] library using the skew function. The skewness of the features can be seen in figure 7. Highly skewed data has been normalized by applying the log transformation.

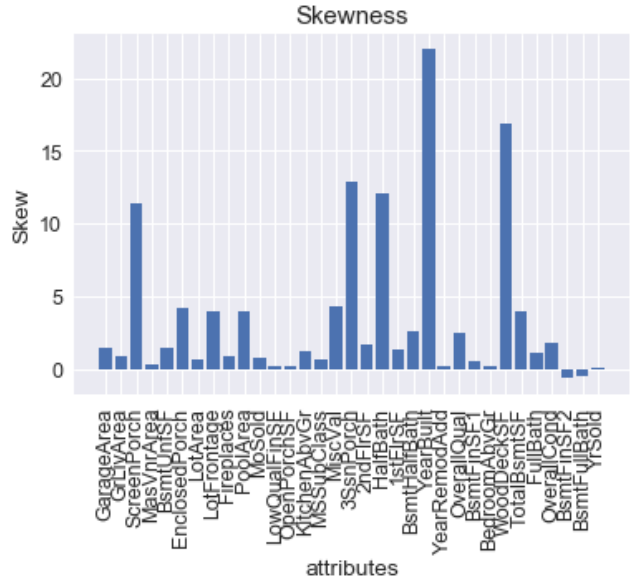


Figure 7: Skeweness of the feature attributes

3.5 Converting Categorical Data to Numerical Data

Most of the prediction models for regression do not work on categorical data. They require all input variables and output variables to be numeric. In general, this is mostly a constraint of the efficient implementation of machine learning algorithms rather than hard limitations on the algorithms themselves. There are several methods for converting categorical attributes to numerical values. In this project two methods, Label-Encoding and One-Hot Encoding have been used.
Label Encoding is an efficient encoding for ordinal attributes, and due to the presence of the ordinal attributes in the dataset Label encoding has been used. The one-hot encoding method has been used for the remaining categorical attributes.

4 PREDICTION MODELS

The data preprocessing steps help improve the prediction efficiency of the prediction models. After the all the data preprocessing steps we are left with a dataset of 2917 rows and 236 columns.

4.1 Decision Tree Regressor

A decision tree is a decision support tool that uses a tree like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements. Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. In this project the decision tree regressor[1] has been implemented.

4.2 Random Forests Regressor

Random forests[6] or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mean prediction (regression) of the individual trees. Random decision forests correct for decision tree's habit of overfitting to their training set. A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. In this project the random forests regressor[1] implemented is taken with a maximum depth of 15 and with 1000 as the number of estimators.

4.3 Linear Regression

Linear regression[7] is a linear approach to modelling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). In this project, for more than one explanatory variable, the process is called multiple linear regression. In linear regression, the relationships are modeled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called linear models. Most commonly, the conditional mean of the response given the values of the explanatory variables (or predictors) is assumed to be an affine function of those values; less commonly, the conditional

median or some other quantile is used. Like all forms of regression analysis, linear regression focuses on the conditional probability distribution of the response given the values of the predictors, rather than on the joint probability distribution of all of these variables, which is the domain of multivariate analysis.

4.4 Artificial Neural Networks

The artificial neural networks in this project have been implemented using the multi-layer perceptron regressor[1]. It is critical to use the correct hyperparameters for the ANNs.

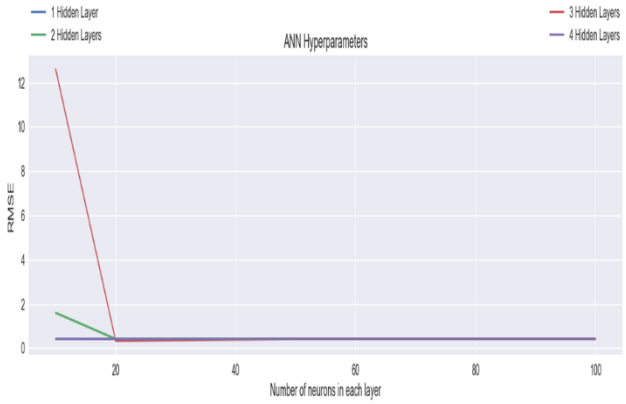


Figure 8: Skeweness of the feature attributes

The hyper parameters for the artificial neural networks include the number of hidden layers, number of neurons. The figure 8 shows the plot of the root mean square error and the hyperparameters. The near optimal value is found with 3 hidden layers and 50 neurons for each layer. The adam optimizer with a rectilinear activation function have been used for the neural network.

4.5 Regularization Models

Regularization is a form of regression, that constrains/ regularizes or shrinks the coefficient estimates towards zero. In other words, this technique discourages learning a more complex or flexible model, so as to avoid the risk of overfitting. There many regularization models, in this project the Ridge Regression and LASSO Regression.

4.5.1 Ridge Regression

Ridge Regression[8] is a technique for analyzing multiple regression data that suffer from multicollinearity. When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors. It is hoped that the net effect will be to give estimates that are more reliable.

In this project Ridge regression with built in cross validation is used with RidgeCV[1] function, so as to obtain & use the near optimal alpha value.

4.5.2 LASSO Regression

In statistics and machine learning, lasso (least absolute shrinkage and selection operator; also Lasso or LASSO) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. The difference between Ridge and LASSO regression is that the ability of the LASSO regression to actually pull the coefficients to zero. In this project LASSO regression with built in cross validation is used with LASSOCV[1] function, so as to obtain & use the near optimal alpha value.

4.6 Gradient Boosting Regressor

Gradient Boosting regressor[1] builds an additive model in a forward stage-wise fashion, it allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function. This ensemble model has been implemented with 3000 estimators, maximum depth of 3 and loss function huber.

5 RESULTS & CONCLUSION

The above implemented model have been compared based on the Coefficient of determination(R^2) obtain from building on the split test data set and the Root mean squared error obtained from the Kaggle website. The root mean squared error is based on the predicted values and 50% of the test data target variable.
R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression. It can be seen in the figure 8 the R^2 value is highest for Ridge regression equal to 0.92967 followed by the gradient boosting regressor equal to 0.92123 and so on.

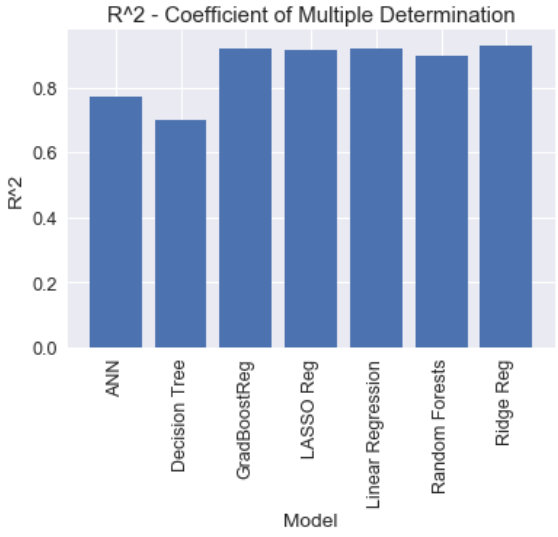


Figure 9: The R^2 value for the models based on the train dataset using test-train-split

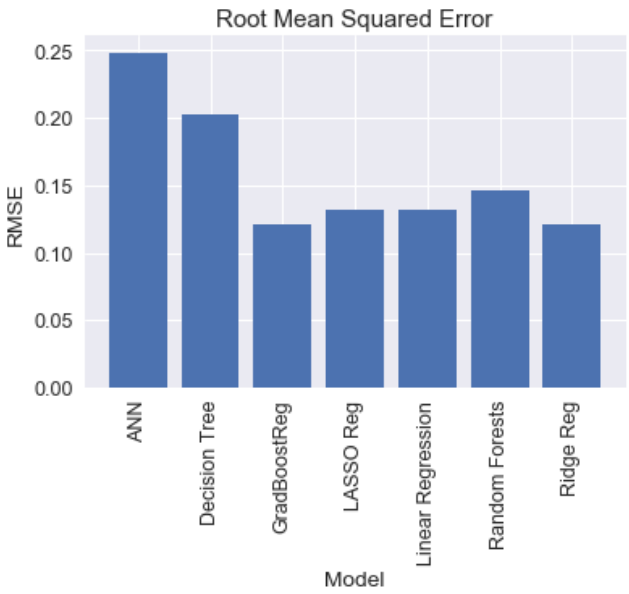


Figure 10: The root mean squared of the algorithms implemented obtained from Kaggle on the evaluation of 50% of test data

The gradient boosting regressor has the least root mean squared error of 0.12061 followed by the ridge regressor and so on.
The gradient boosting regressor has the least root mean squared error of 0.12061 followed by the ridge regressor and so on. The Artificial Neural Networks have the highest root mean squared error, this might be because of the small size of the dataset.
The best model implemented is the gradient boosting regressor. The model ranked in the top 20% in the Kaggle competition.

6 CONCLUSIONS

In summary, we have implemented several data preprocessing techniques as well as several prediction models. The Gradient boosting regressor was the best among the different models implemented. The project has brought me hands-on approach towards data mining.

ACKNOWLEDGMENTS

I would like to thank Professor Thaddeus F. Pawlicki for giving me the opportunity to do this project and helping me. I would also like to thank my classmates for helping me with critical concepts.

REFERENCES

[1] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011
[2] De Cock, D. Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project. *Journal of Statistics Education 19*, 3 (2011).
[3] Jones E, Oliphant E, Peterson P, *et al.* SciPy: Open Source Scientific Tools for Python, 2001-, http://www.scipy.org/[Online; accessed 2018-11-21]
[4] 10.Hunter, J. Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering 9*, 3 (2007), 90-95.
[5] *Blog.minitab.com*, 2018. http://blog.minitab.com/blog/understanding-statistics/handling-multicollinearity-in-regression-analysis.
[6] Random forest. *En.wikipedia.org*, 2018. https://en.wikipedia.org/wiki/Random_forest.
[7] Regression analysis. *En.wikipedia.org*, 2018. https://en.wikipedia.org/wiki/Regression_analysis.
[8] NCSS 12 Statistical Software (2018). NCSS, LLC. Kaysville, Utah, USA, ncss.com/software/ncss.