House Prices: Advanced Regression Techniques

A Project report for the assignment submitted in fulfillment of the requirements for the COMP SCI 7209 Big Data Analysis and Project

MASTER OF DATA SCIENCE

by

Arpit Garg

Register No: A1784072

Semester: III, Year: 2

THE UNIVERSITY of ADELAIDE

SCHOOL OF COMPUTER SCIENCE

THE UNIVERSITY OF ADELAIDE

ADELAIDE - 5000

AUGUST 2020

Abstract

In this project, we are using machine learning regression algorithms to predict house prices. The dataset is available on Kaggle. In the dataset, we have approximately eighty parameters on which prices could depend. In particular, we will focus on the parameters and clean all the parameters and will drop some of the unwanted parameters before training our model. In modeling, we will use different machine learning regression algorithms such as Linear, Ridge, Lasso, ElasticNet and Boosting algorithms with Stack Regression to build our model and predict the sale prices. Moreover, we have compared our models based on RMSE. We are using visualisation at different places for comparison. This report focuses on solving the assignment and Kaggle question with detailed analysis and in-depth context related to Machine Learning Regression techniques. Using the approach, we secured a place in the top 20% at the time of submission.

Keywords : Machine Learning, Ridge, Lasso, ElasticNet

1 Introduction

Housing is one of the necessities of human life, and demand is increasing day by day. It is now essential to predict house prices as accurately as possible. The price is dependent on various factors while predicting. House prediction is useful for not only buyers and sellers but also helps in maintaining the economy of the region. The main aim of this project is to predict the sale prices of residential houses in Ames, Iowa. For achieving our purpose, we are using the dataset provided on Kaggle for training and testing purposes. In this dataset, we have 79 parameters that we could use in our model. This dataset contains almost all the parameters that affect the price of houses. Dean De Cock prepared this dataset for data science education purposes [2]. For training purposes, we have different 1460 houses with 79 different features and output sale prices. For testing purposes, we have 1459 observations with the same features excluding sale price because that is what our model will predict. The primary purpose is to develop a model that can predict house prices based on these parameters accurately.

2 Methodology

2.1 Data Cleaning and Pre-processing

In this dataset, 79 features need to handle. Our target is Sale Price. We have plotted the typical histogram of SalePrice and log transform of SalePrice, respectively, to observe the skewness.

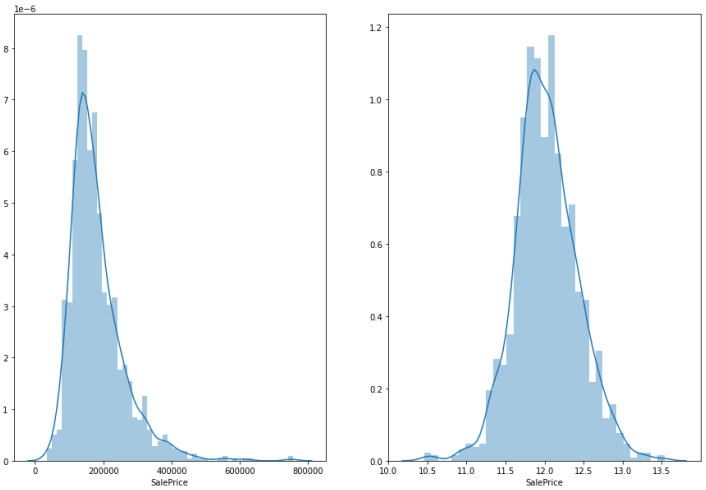

Figure 1: Histogram of SalePrice and log(SalePrice)

So we will use the log transform for prediction and nd the inverse of the log at last for the final price on test cases. We need to clean the data and handle the missing values. We were combining the train and test data and labelled them so that after cleaning, we can again separate them. We calculated the missing value percentage according to the features and handled the features accordingly. We dropped some parameters by observing the missing percentage, the parameters in which more than 75% values are missing. For some of the features, we have considered the correlation between other parameters to ll the missing values like for LotFrontage. Some columns are dropped based on the correlation value calculate. For a few parameters, we have filled the missing values with mean and mode.. By observing the missing percentage, the parameters in which more that 75% values are missing are dropped. For some of the features, we have considered the correlation between other parameters to fill the missing values like for LotFrontage. Correlation between all the parameters and target is calculated and based on that some parame-

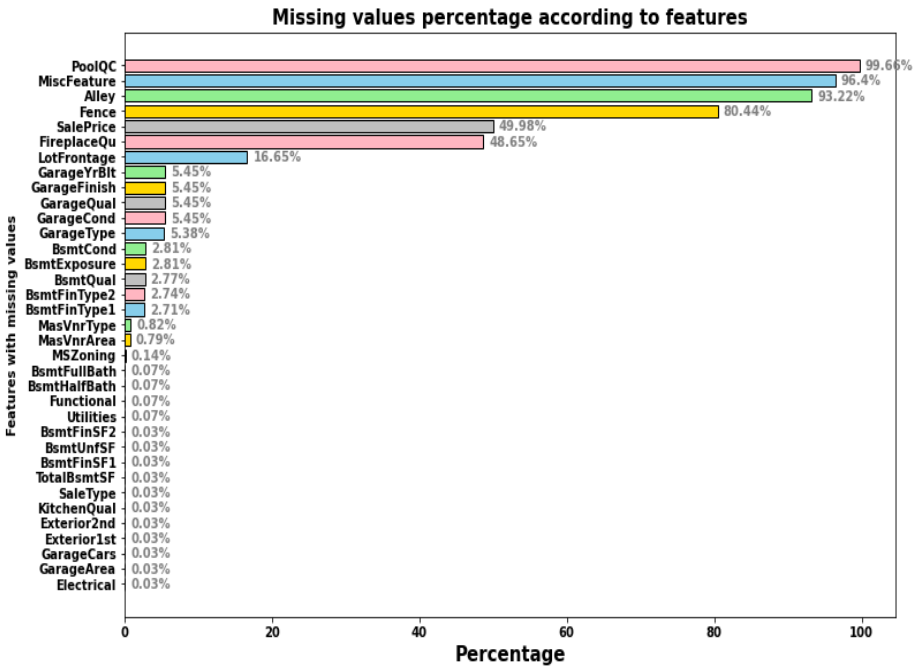

Figure 2: Missing Percentage

ters are dropped. For few parameters we have filled the missing values with mean and mode.

2.2 Data Modelling

To train our model, we have used different regression models and compared based on Root mean square error.

2.2.1 Linear Regression:

Linear Regression is the fundamental and old technique for predicting quantitative values, but it is still a practical approach. In linear regression, it assumes that there is a linear relationship between the output Y and input X. In our case, Y is the target price, and it assumes that it will be having the linear relationship between all the features acting as X. Mathematically, we can denote it as:

$$Y \approx \beta_0 + \beta_1 * X_1 + ... + \beta_1 * X_N \tag{1}$$

Where Y is output and X_1 to X_N are the N input features and it uses the given data to estimate the values of β's. They are selected to minimize the sum of squared residuals. We fit the straight line that best fits in our train-

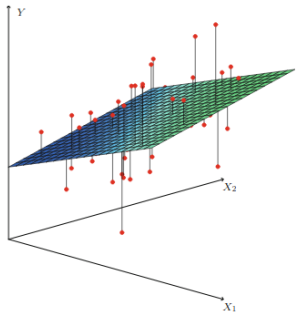

Figure 3: Linear Regression [4]

ing sample and predict the values based on the fitted line. We applied linear regression for our model as the first regression model.

2.2.2 Ridge Regression and Lasso Regression:

Next, we have evaluated our results on the ridge and lasso regression. Both of these regressions are similar to least squares, but for minimizing the β coefficients, a different mathematical concept used.
For Ridge:

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{P}j_j^2=\mathrm{RSS}+\lambda\sum_{j=1}^{\mathrm{P}}\beta_j^2 \tag{2}$$

where λ is tuning parameter and the second λ term is known as shrinkage penalty. The tuning parameter λ used to adjust the relative effect of two terms on the coefficient of regression.

For Lasso:

$$\sum_{i=1}^{n}\left(y_i-\sum_{j}x_{ij}\beta_j\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| \tag{3}$$

Lasso regression is similar to the ridge but overcomes the disadvantage of Ridge Regression. Ridge regression will reduce the magnitude of coefficients,

but it will never exclude any parameter, but it is possible with lasso regression. We have trained our model with both ridge and lasso. By comparing with all the mentioned regression model, we can observe in our evaluation part that we are getting good results in lasso, then slightly lower results with ridge then slightly lower with linear because of the mentioned reasons.

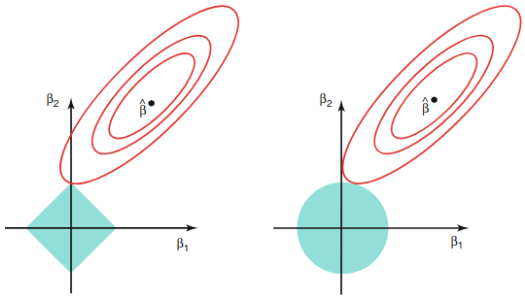

Figure 4: Lasso(left) Ridge(right) [4]

There are a few disadvantages of Lasso regression also. Like sometimes in case of more than one highly correlated variable, it will select one and ignore the others. To overcome these, we have ElasticNet, which denoted as:

$$\beta = argmin \left[\sum_{i=1}^{n} \left[v_r\left(y_i\right) - \sum_{j=1}^{k} v_j\left(x_{ij}\right)\beta_j \right]^2 + \lambda_1 \sum_{j=1}^{k} |\beta_j| \ + \lambda_2 \sum_{j=1}^{k} \beta_j^2 \right] \quad (4)$$

It is having a unique minimum because of quadratic term that's make the strongly convex loss function. We can observe a better results than lasso in this while training our model.

2.2.3 Boosting:

For the comparisons, we have tried to train our model with also boosting to see the results. Boosting algorithms is an ensemble technique to convert weak learners to keen learners. Gradient Boosting do not change the weight Gradient Boosting do not change the weight of every incorrect regressor, but it will try to t the new predictor to the remaining error by the last predictors. XGBoost means eXtreme Gradient Boosting. Traditional Gradient boosting works in the sequential pattern, so it is slow. LightGBM and XGBoost both are an advancement of Gradient Boosting with high speed. We have used all these three models to train our model and calculated the r2 and RMSE.

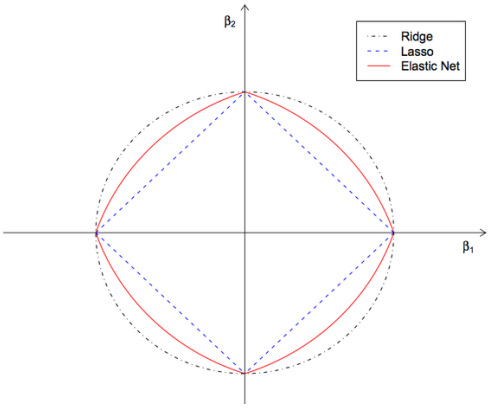

Figure 5: Penalty terms in the space of the model parameters [5]

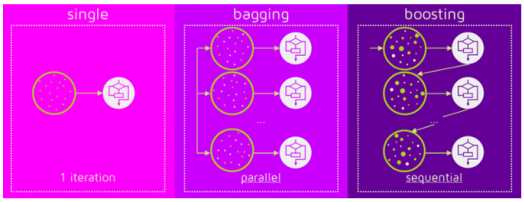

Figure 6: Bagging - Boosting [1]

2.2.4 Stacked Regression:

It is a way of generating a linear combination of different regression algorithm to predict good results. It uses least-square and cross-validation to calculate the contribution of each regression in the final result. We have used stack regression for our final output.

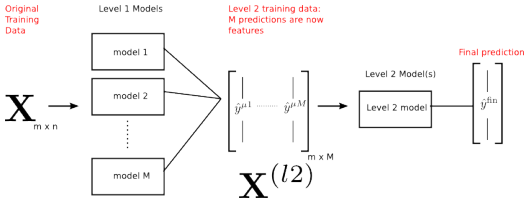

Figure 7: Stack Regression [3]

3 Implementation

Implementation of this project can be divided into three parts:

3.1 Exploratory Data Analysis (EDA)

Every project should start with the Exploratory Data Analysis (EDA) to gain insights from the data. It helps to examine the structure of dataset like patterns. For this purpose, we have used different python libraries like NumPy, pandas, matplotlib. We have printed the data shape with head and used describe the structure if dataset. We can identify that the data is skewed, but if we use a log transformation of the target variable, then it has normalized behaviour. We have also used correlation to nd the relation between all the variables with the output variable, to observe the contribution of each column with it. We have separated the data into two parts, one for categorical data and plotted pairwise and second for numerical data and plotted graph. It is also desirable to plot the correlation heatmap with missing percentage graph.

3.2 Data Pre-Processing

Pre-processing parts is very crucial to handle noisy or no data in the columns. In this dataset, we have approx 80 columns, and we need to pre-process data such as cleaning and filling missing values before modelling. Hence, all the columns with null value count are printed and based on that all the columns with more than 75% of values missing are removed. Furthermore, we calculated the correlation between all the variables with the target variable and dropped the columns or parameters with deficient correlation value. For parameters like FirePlaces and LotFrontage, we have found the correlation of these with different variables and ll the missing values accordingly. For example, the correlation between LotFrontage and sqrt(LotArea) is more significant than 0.60, so we have filled the missing values in LotFrontage, according to sqrt(LotArea). For some parameters like Garage and Basement, we have combined them and then replaced their missing values.
Some of the columns are transformed like we have added a standard total bathroom column instead of separate bathrooms and removed the previous variables. In some columns, missing values are replaced by either the most frequent value or mean value of the column. Some parameters, like YearBuilt,

is transformed into House age. At last, before training our data, we converted all the categorical data using one-hot encoding.

3.3 Data Modeling

In data modeling, we applied different regression models to compare between all. We started with linear regression. Before fitting the model, we separated the data into 70% for training and the other 30% for cross-validation. We first fitted the model using Linear Regression and keeping it with default parameters and calculated the RMSE and R2 score. Then we moved towards the regression models that requires regularization (penalty) like Ridge, Lasso and Elastic-Net. We have used LassoCV and RidgeCV as it will automatically select the best value of alphas. Then, we trained our model using boosting algorithms. Boosting algorithms used here includes Gradient Boosting Algorithm, XG-Boost Regressor and LightGBM Regressor, and all the evaluations are stored. At last, we used the advance approach of stack regression to include the contribution of each regression and predicts the output. Final submission on Kaggle competition made using the stack regression.

4 Evaluation

For the evaluation purposes, we are mainly focusing on Root Mean Square Error(RMSE), and we need low RMSE for each model. We have stored and displayed RMSE for all the seven regression models. Furthermore, to display the efficiency of our models, we have also displayed the R2 score. All these evaluations are performed on the cross-validation set. According to the constraint of this assignment, we have divided the rest 70% for the training purposes and remaining for cross-validation. For the testing purposes, we have predicted the outputs and uploaded them on Kaggle and got 0.12453 RMSE, and secured the position in top 20% solutions provided when submitted. We have stored the results in evaluation metrics table and plotted the scatter graph of the predicted output on the cross-validation set.

Link: https://www.kaggle.com/arpit2412/competitions

4.1 Evaluation Table:

Table 1: Evaluation Metrics

S.No.	Regression	RMSE	R2 Score
1	Linear	0.17029800799928	0.804938289128826
2	Lasso	0.239424613260201	0.614441322218327
3	Ridge	0.170689507975853	0.804040399174513
4	Elastic-Net	0.171258099468452	0.802732685033986
5	Gradient Boost	0.125469878450216	0.894115632275697
6	XG-Boost	0.123236082536514	0.897852283643857
7	LightBGM	0.129042763853327	0.887999437046273
8	Stack	0.129274221197501	0.887597297464596

4.2 Output Plots:

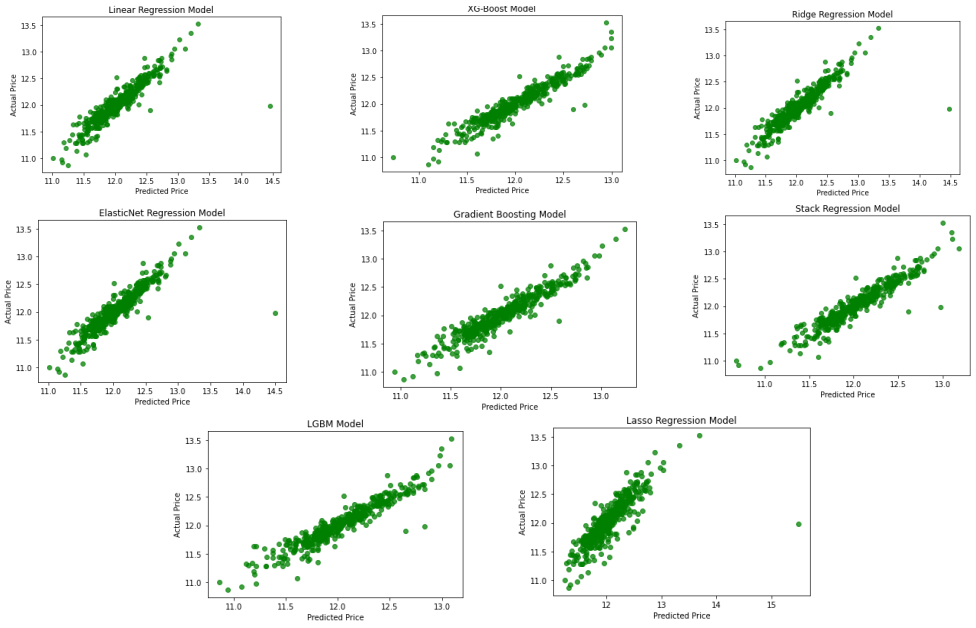

Figure 8: Output Plots

From the above Evaluation Table and Output Plots, we can observe that we are getting the best results in boosting algorithms. We are getting worst results in Lasso Regression. From the figures, we can see that all the boosting

regressors are testing the model well and predicting the proper results as compared to regularisations.

References

[1] aporras. what-is-the-difference-between-bagging-and-boosting.

[2] Dean De Cock. Ames, iowa: Alternative to the boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19(3), 2011.

[3] Burak Himmetoglu. Stacking models for improved predictions.

[4] Jianhua Z Huang. An introduction to statistical learning: With applications in r by gareth james, trevor hastie, robert tibshirani, daniela witten, 2014.

[5] A. Sosnovshchenko, O. Sosnovshchenko, and O. Baiev. *Machine Learning with Swift: Artificial Intelligence for IOS*. Packt Publishing, 2018.