Multinomial Regression to Predict House Price

Shubham Kumar
Machine Learning Intern
AI Technology and Systems
skagnihotri1@gmail.com
www.ai-techsystems.com

***Abstract*— The objective of this report is to predict the house price using two multinomial regression algorithms. Linear Regression and Support Vector Machine are used to predict the prices. The dataset is taken from Kaggle. Dataset consist of different aspects which affect house price. Hyper tuning and Regularization are used to make model more robust and powerful.**

Keywords—Linear Regression, Support Vector Machine (SVM), Regularization, Encoding, Grid Search

I. Introduction

Regression is the technique to predict the continuous data. It consists of the predicting function known as Hypothesis, which is a function of features (dependent variables). For accuracy to hypothesis, Cost(loss) function which calculates the root mean square distance between the true and predicted values. Gradient Descent is used to minimize the cost function and make hypothesis more powerful. The algorithms used are Linear Regression with l1 and l2 regularization and Support Vector Machines. Grid Search and Cross Val Score are used to tune the hyper parameters and make Bias and Variance small and converge the algorithm to global minima.

II. Machine Learning

Machine learning (ML) is a class of algorithm that allows software applications to become more accurate in forecast outcomes without being precisely programmed. The basic assertion of machine learning is to construct algorithms that accepts input data and use statistical analysis to forecast an output while updating outputs as new data becomes available. The actions involved in machine learning are similar to that of data mining and predictive modeling. Both require searching through data to seek patterns and adjusting program actions accordingly. Many people are accustomed with machine learning from shopping on the internet and being served ads related to their purchase. This happens because recommendation engines use machine learning to customize online ad deliveries in almost real time. Beyond customized marketing, other prevalent machine learning cases include fraud detection, spam filtering, network security threat detection, predictive maintenance and building news feeds. Machine learning algorithms are often classed as supervised or unsupervised. Supervised algorithms require a data scientist or data analyst with machine learning skills to administer both input and desired output, in addition to feedback about the accuracy of predictions during algorithm training. Data scientists determine which variables, or features, the model should analyze and to develop predictions. When training is complete, the algorithm will apply what was learned to new data. Unsupervised algorithms do not require to be trained with desired outcome data. Instead, they use a progressive approach called deep learning to review data and arrive at conclusions. Unsupervised learning algorithms – also known as neural networks– are used for more complex processing tasks than supervised learning systems, including image recognition, speech-to-text and natural language generation.

III. Preprocessing And Multinomial Regression Methods

A. Preprocessing

Preprocessing means cleaning of data i.e. removal of “nan” values, one hot encoding, filling of “nan” values, etc.

“nan” values are if present in bulk in a certain column or in some rows then the rows or columns are just dropped. But if, they are very few we replace them with mean, median or mode values according to the need.

Categorical columns are the columns which only have string value, so to use them they are one hot encoded and dropped and the new generated column are added. It is important to drop one column from every one hot encoded column to save the algorithm from dummy variable trap.

TABLE 1. Column Type

	Number
Categorical Variables	40
Non-Categorical Variable	12

As the preprocessing steps are completed, we train out model on the training set.

There are many regression algorithms but most popular regression algorithms are Linear Regression and SVM. First the model is trained on the training set and then accuracy is checked on the validation set.

After training is done model accuracy was analyzed on the validation data. The cross_val_score of the model were plotted.

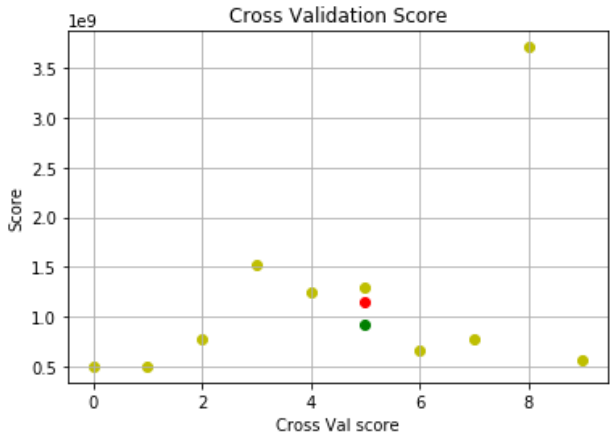

Fig 1. Cross Validation score (R2 score) of Linear Regression with L1 regularization.

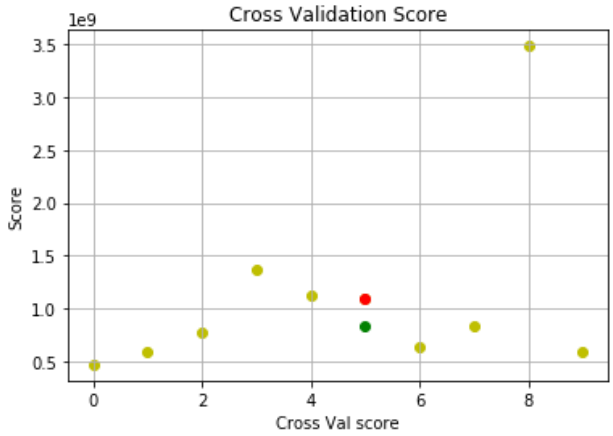

Fig 2. Cross Validation score (R2 score) of Linear Regression with L2 regularization.

And for SVM Grid Search was used to hyper tune the values of parameter "C".

B. Linear Regression

The benefit of linear regression is that it is the simplest regressor and is readily interpretable. Linear Regression Model Representation Linear regression is an attractive model because the representation is so simple. The representation is a linear equation that combines a specific set of input variables

(x) the solution to which is the predicted output for that set of input variables (y). As such, both the input variables(x) and the output variable are numeric. The linear equation allocates one scale factor to each input column, called a coefficient and represented by the Beta(B). Adding an additional coefficient also gives the line added degree of freedom and is often called the intercept or the bias coefficient. For example, in a simple regression problem (a single x and a single y, the form of the model would be:

y= M0 + M1*x ————(1)

In higher dimensions when we have more than one input (x), the line is called a plane or a hyper-plane. The representation is the form of the equation and the specific values used for the coefficients (e.g. M0 and M1 in the above example). To talk about the complexity of the linear model refers to talk about the no of coefficients used in the model Predictions using Linear Regression: Given the representation is a linear equation, making predictions is as simple as solving the equation for a specific set of inputs. Let's make this concrete with an example. Imagine we are predicting output (y) from input (x). Our linear regression model representation for this problem would be:

y = M0 + M1 * x1————(2) or

output =M0 +M1 * input. ————(3)

Where M0 is the bias coefficient and M1 is the coefficient for the input column. We use a learning technique to found a good set of coefficient values. Once found, we can plug in different input values to predict the output.

Stochastic Gradient Descent is a basic yet very suitable approach to learning of linear classifiers under convex loss functions such as (linear) Support Vector Machines and Logistic Regression21. Ever since SGD has been around in the machine learning community, it has now received a considerable amount of attention just recently in the context of large-scale learning.

SGD has been successfully applied to large-scale and sparse machine learning problems often encountered in text classification and natural language processing. Given that the sparse data, the classifiers in this section easily scale to problems with more than 105 training examples and more than 105 features. The class SGD Regressor executes a plain stochastic gradient descent learning routine which supports different loss functions and penalties to t linear regression models. SGD Regressor is well suited for regression problems with a large number of training samples, for other problems were commend Ridge, Lasso, or ElasticNet.

The concrete loss function can be accessed via the loss parameter. SGD Regressor supports the following loss functions: loss=squared-loss: Ordinary least squares,

loss=Huber: Huber loss for robust regression,

loss=epsilon-insensitive: linear Support Vector Regression. The Huber and epsilon insensitive loss functions may be used for robust regression. The width of the insensitive region has to be specified through the parameter epsilon. This factor depends on the scale of the target variables. SGD Regressor supports averaged SGD as SGD Classier. Averaging can be enabled by setting average=True.

C. Support Vector Machine

SVM or Support Vector Machine is a linear algorithm for classification and regression tasks. It can solve linear (using 'linear' kernel) and nonlinear (using kernels such as 'rbf', 'polynomial' and 'sigmoid') task and work great for many daily life problems. The concept of SVM is easy: The algorithm creates a separation line and support vector lines which separates the data into different classes.

The optimal value of C and gamma using grid search approach. And choosing the linear (it works faster than other kernels) kernel. After the hyper parameters are tuned. The optimized SVM algorithm is trained and applied.

IV. Result

Linear Regression with l1 and l2 regularization is used and the following graphs between predicted values and true values were plotted.

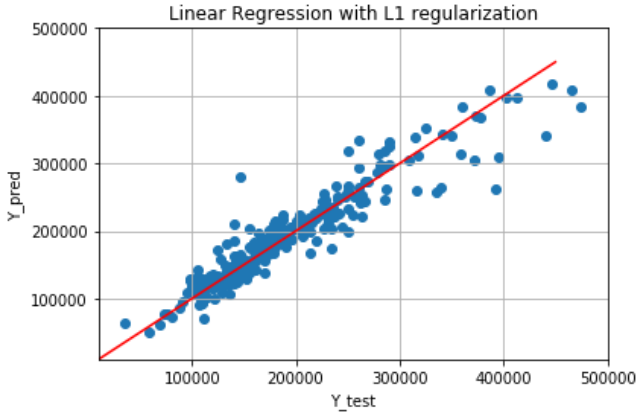

Fig. 1. Linear Regression with absolute mean Regularization.

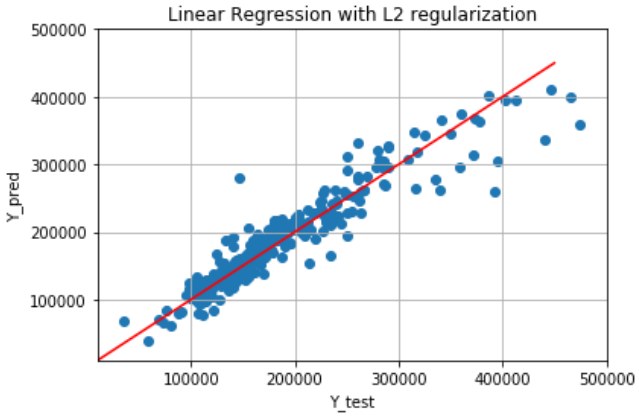

Fig. 2. Linear Regression with mean square Regularizations.

Support Vector Machine with Grid search hyper parameter tuning is used and a graph between predicated and true values is plotted.

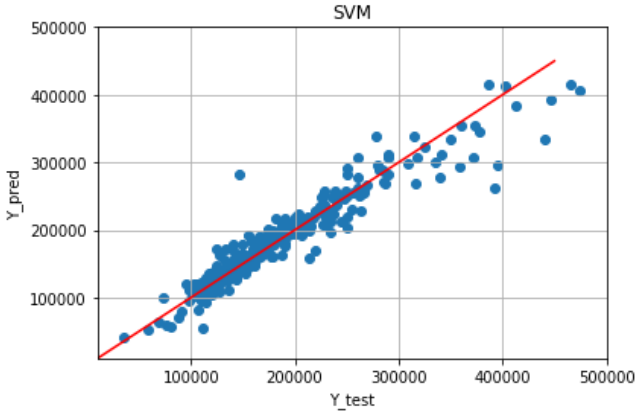

Fig. 3. Support Vector Machine with parameter tuned with k-fold cross validation and grid search.

V. Conclusion

Both are really powerful model and are generally used. The models perfumed really good on the data. Linear Regression giving 89%(approx.) accuracy i.e. R2 score where as SVM giving 85%(approx.) accuracy i.e. R2 score. Linear Regression performing better then support vector machine. Linear Regression time cost is also less than that of SVM.