International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-10 Issue-6, August 2021

An Optimized way to Solve Regression Problems

Jyothi Vishnu Vardhan Kolla, Poorna Chandra Vemula, Vanapala Sai Mohit

Abstract: ***In many real world scenarios, regression is a commonly used technique to predict continuous variables. In case of noisy(inconsistent) and incomplete datasets, a large number of previous works adopted complex non traditional machine learning approaches in order to get accurate predictions. However, compromising on time and space overheads. In this paper, we work with complex data yet by using traditional machine learning regression algorithms by working on data cleaning and data transformation according to the working principle of those machine learning algorithms.***

Keywords: Regression, Data processing, Noisy data, Random sampling.

I. INTRODUCTION

In today's scenario, Due to availability of abundant data, Many Companies use regression techniques to predict their revenue outcomes and decide their investment strategy. In Case of stock market predictions, many investment firms use regression techniques to analyse the trends, and decide the buying and selling of stocks. Also, In the Real Estate domain, companies use regression techniques to predict the prices of the assets. If we have a scenario, where a user gets accurate prediction but if it takes a good amount of time, to get the results; Then the users may not be inclined to use this kind of application. In the other case, even if the user gets slightly less accurate prediction, but if the results are pretty fast then the user will be more inclined to use the application further. So, In most of the real world scenarios; It is very much preferable to use less complex algorithms even if they give a slightly less accuracy. Several studies solved the problem using complex regression techniques to achieve lower error rates yet resulting in high time and space overheads. S. Lu et al. did conduct an experiment using a hybrid regression, but it requires intensive parameter tuning[2], similarly using neural networks may give better results but adds in more complexity[4], in order to make the model generalized, but our paper focuses on building simple yet optimal models.

Manuscript received on June 20, 2021.
Revised Manuscript received on August 09, 2021.
Manuscript published on August 30, 2021.
* Correspondence Author
Jyothi Vishnu Vardhan Kola, Pursuing, BTech, Department of Computer Science and Engineering, Gitam university vishakapatnam, Andhra Pradesh
Poorna Chandra Vemula*, Pursuing, Bachelors of Technology, Department of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu
Vanapala Sai Mohit, Pursuing, Bachelors of Technology, Department of Computer Science and Engineering, Gitam Institute of Technology, Visakhapatnam, Andhra Pradesh

In this paper, to drive our point as stated in the title, a complex dataset have been taken yet solved using traditional regression algorithms like linear regression, Decision tree regression, Random Forest regression, Gradient Boosting regression with lower error rates by transforming the features in the dataset such that it suits the mathematical foundation of the algorithms. For Example, If the model is built on geometric concepts like distance then the model prefers scaled features, so this paper mainly focuses on tuning the features in a way it suits such models rather than building complex models. The dataset we have used is gathered by Cock, Dean.[1](The dataset may be updated by the time you view the paper, we worked on the dataset, available to us while writing this paper. At the time of writing this paper, there are 81 columns and 1460 rows).

II. METHODOLOGY

A. Data Preprocessing and Visualizations:

In the dataset "Advanced house price prediction" we have a total of 80 features which are representing the way in which the house prices are affected in the regions of Ames,Lowa.

Step1: Firstly we have started to observe whether there are any null values in our dataset, if there are any present we have to check whether these null values are having any impact on our target variable

Steps followed:

- Go through each and every feature in the dataset and replace with '0' in place of null value and replace with '1' incase of non-null values.
- Groupby each feature with sale price and calculate their median and plot a bar-chart for better visual understanding of how the null values affected our target variable
- From figure(1) we can understand that null values do affect our target variable, we can see a similar kind of pattern in other features as well containing null values.

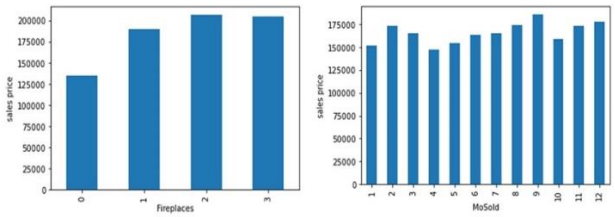

Fig. 1. Effect of null values on sales price

- After a proper scrutiny on visualization it is evident that null values do affect our target variable, it's better to find an alternative method to impute them rather than just dropping them from the dataset.

Step 2: separate the features in the dataset into their respective forms for instance create four new variables and store continuous features, discrete features,categorical features.
Steps to follow:

- Impute all the missing values in numerical features using median of that particular feature, as median is more robust towards outliers.
- Check whether all the continuous variables are normally distributed, if not apply log transformation and standardize them. As we can see in the dataset, some features are not normally distributed, so we need to standardize them.
- We can see from figure(2) that not all the features are normally distributed, and similar patterns can be observed in the remaining features, which are not displayed in the above figure. So, In order to make the features normally distributed log transformation needs to be performed.
- Then, Find the number of distinct categories in each categorical feature and perform GroupBy with the target variable by applying median as an aggregate function to analyse the relation between the distinct categories and the target variable.

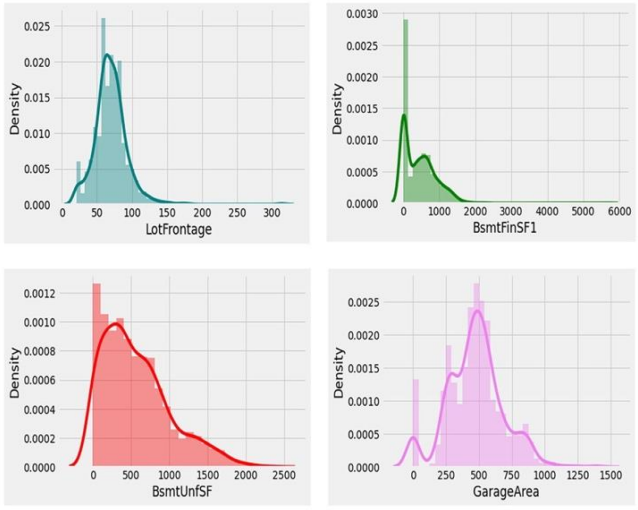

Fig. 2. Distribution of various features

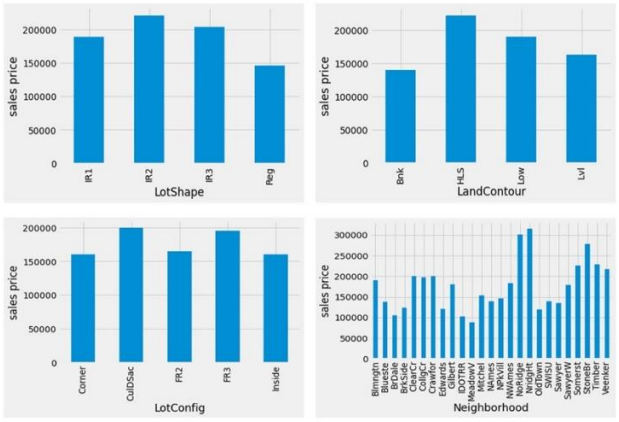

Fig. 3. Effect of distinct categories on sales price

- Each and every distinct category in our categorical features, have an impact on our target variable as in figure(3). So instead of dropping the null values in each categorical feature, impute them with a new category ‘missing’.

Step 3: Store all the year related features(‘YearBuilt', 'YearRemodAdd', 'GarageYrBlt','YrSold’) into ‘year features’ variable.

- Now, traverse through all the year related features except ‘YrSold’ and update them with the difference between corresponding values and YrSold.
- Drop the ‘YrSold’ feature from ‘D’.

After Data preprocessing, the dataset is checked for outliers(excluding year features). Through Inter-Quartile Range (IQR), an outlier *xi* can be detected if:

$$X < Q1 - 1.5 \cdot IQR \quad \text{OR} \quad Q3 + 1.5 \cdot IQR < x$$

where:
$Q1$ = 25 percentile; $Q3$ = 75 percentile; $IQR = Q3 - Q1$
Out of 77 features(excluded 3 year features); 34 are numerical features(17 : continuous, 17 : discrete), 43 are categorical features.

These features are checked for outliers using above expression. If any found they are removed from our dataset.

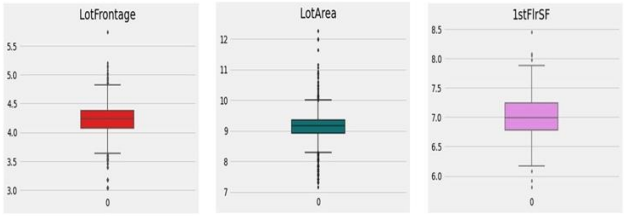

Fig. 4. representing outliers using whisker plots

As mentioned above, the presence of outliers can be detected using whisker plots as shown in Figure(4).

B. Model Selection:

These Notations will be used to explain the algorithms used to solve the regression problems,
D : Dataset; X : Set of all input features; Y : Target Variable.

III. LINEAR REGRESSION

Linear Regression is the most simplest of all the contemporary regression techniques. It is a distance based method, which works on principle of taking data points as inputs and predicting a target variable which is a real valued feature. It is of two types, univariate and Multivariate, Univariate Involves only a single input feature and a target variable whereas Multivariate can contain ‘n’ number of features.

The above can be mathematically represented as,

$$D = \{x_i, y_i\}_{i=1}^{n} \; where \; y_i \in \mathbb{R}^d$$

As Linear Regression is a distance based method, for good results all the features need to be of the same scale. So, feature scaling is significant in this case. In order to achieve feature scaling following statistical approach need to used,

$$\sum_{i=1}^{n} \frac{x_i - Mean(\mu)}{Standard\ Deviation(\sigma)}$$

The Optimized formulation of Linear Regression is represented as follows,

$$(\omega^*, \omega_0) = argmin_{\omega,\omega_0} \sum_{i=1}^{n} (y_i - (\omega^T x_i + \omega_0))^2 + \lambda \|\omega\|_2$$

The goal is to,

- Minimize the error between predicted target variable and actual target variable
- To find the correct values of (omega t) and (omega 0) in such a way that the model best fits training data.
- In order to find the right (omega t) and (omega 0) gradient descent is used.
- $$x_{i+1} = x_i - \gamma \left[\frac{\partial f}{\partial x}\right]_{x_i} \quad if\ a\ vector\ then\ (\bar{V}_x f)_{x_i}$$
- Make sure to take care of underfitting and overfitting by using L1 and L2 regularization as shown in the above equation.

In Order to check the performance of the algorithm, metrics such as RMSE, MSE.

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

$$RMSE = [\sum_{i=1}^{n} (x_{f_i} - x_{0_i})^2 / n]^{1/2}$$

1. Decision tree Regression

Decision tree is a tree based method, unlike SVM regression and Linear regression which are purely distance based methods. In a nutshell, Decision tree is a nested if-else statement.

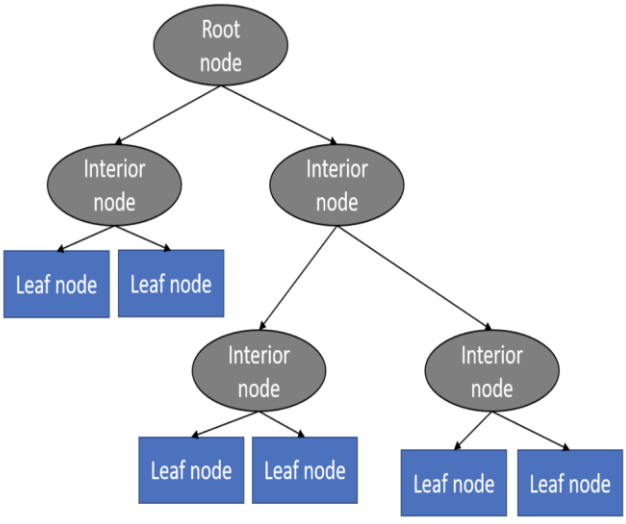

Fig. 5. Illustration of decision tree

The root node represents the entire sample, a threshold value is fixed in the root node and further splitting of nodes is based on this threshold value. The Leaf node represents the outcome of the tree. As the depth of the tree increases, the model becomes more robust.

The important factors used to build a decision tree,

Entropy:

$$H(Y) = -\sum_{i=1}^{k} P(y_i) \log_b (P(y_i))$$

Information gain:

$$\text{I.G.} = -\sum_{i=1}^{k} P(y_i) \log_2 (P(y_i))$$

Gini Impurity:

$$I_G(y) = 1 - \sum_{i=1}^{k} P(y_i)^2$$

After performing Data preprocessing and feature transformation, the decision tree is trained with the data with some default parameters provided by python sklearn library,

criterion='mse', max_depth=None, min_samples_split=2, min_samples_leaf=1, max_features=None, max_leaf_nodes=None [5]

Tuned parameters,

criterion='mse', max_depth=10, min_samples_split=4, min_samples_leaf=3, max_features=None, max_leaf_nodes=None[5]

2. Random forest Regression

Random forest is an ensemble technique which involves creating multiple decision trees in a way that each decision tree is trained with data taken randomly from the dataset, then these trees are aggregated together and majority voting is applied on them to predict the target variable.[6] The steps to build a random forest is summarized by S. Raschka and V. Mirjalili in their book "Python Machine Learning" are as follows[3]:

1. Draw a random bootstrap sample of size *n* (randomly choose *n* samples from the training set with replacement).
2. Grow a decision tree from the bootstrap sample, at each node:
   (a) Randomly select *d* features without replacement.
   (b) Split the node using the feature that provides the best split according to the objective function, for instance, maximizing the information gain.
3. Repeat the steps 1-2 *k* times.
4. Aggregate the prediction by each tree to assign the class label by majority vote.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication

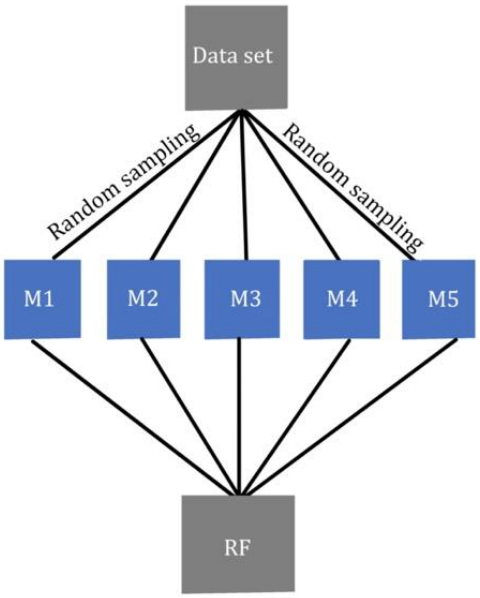

Fig. 6.Representation of Random Forest

To build Random Forest Regressor, python sklearn library is used with default parameters.

class sklearn.ensemble.**RandomForestRegressor**(*n_estimators=100, *, criterion='mse', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, ccp_alpha=0.0, max_samples=None*) [5]

Tuned Parameters are,

class sklearn.ensemble.RandomForestRegressor(n_estimators=400, *, criterion='mse', max_depth=70, min_samples_split=10, min_samples_leaf=4, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, ccp_alpha=0.0, max_samples=None)[5]

So as mentioned earlier, the features have been normalized. First the model is implemented by using default parameters provided by sklearn and then hyperparameter tuning is performed by using sklearn. model_selection. **Randomized Search CV**[5].

3. Gradient boosting Regression:

Boosting is an ensemble method which is of the idea that combining many weak learners into a single strong learner. Gradient boosting is one of the most used boosting algorithm. It works on the principle of sequentially adding predictors to the model in such a way that each one corrects its predecessor

Algorithm:

Input: Training set {xi,yi} containing n samples and a differentiable loss function L(y,f(x))

1. Initialize the model $F_0(x) = argmin_r \sum_{i=1}^{n} L(y_i, r)$
2. For m=1 to M,
   Compute so called pseudo residuals,

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$$

3. Fit a base learner to the pseudo-residuals and train it using the training set

4. Compute multiplier rm

$$r_m = argmin_r \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + rh_m(x_i))$$

5.update the model

$$F_m(x) = F_{m-1}(x) + r_m h_m x$$

6. Regularization and shrinkage:

$$F_m(x) = h_0(x) + \sum_{m=1}^{M} r_m h_m x$$

IV. RESULTS

The results of various regression models can be seen in the graphs shown in the figure(7) and metrics when various algorithms are used, is shown in table(1). As shown, the Random Forest Regressor performs better than other algorithms and Decision tree performed well with training data but not so good with testing data due to its nature of having low bias and high variance.

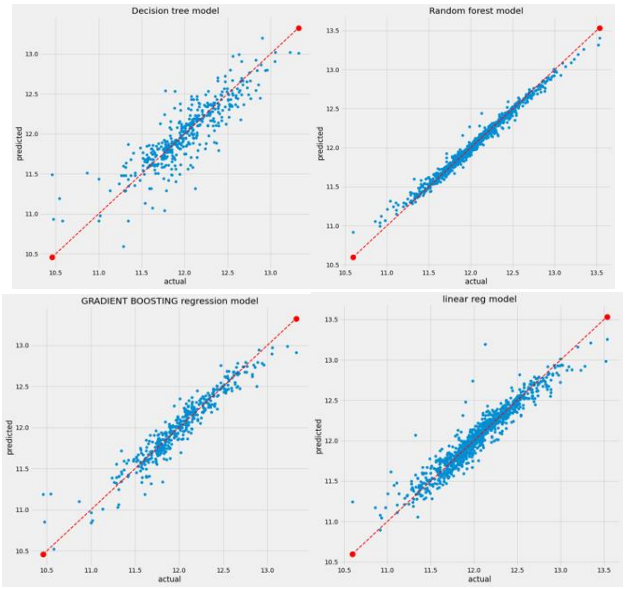

Fig. 7. Graph between actual and predicted values for different algorithms

V. CONCLUSION

This paper investigates different machine learning techniques to solve regression problems. A regression problem with a complex dataset is taken and solved using traditional regression algorithms such as Linear Regression, Decision Tree regression, Random Forest regression, Gradient Boosting Regression with lower error measures, this is achieved because of proper data cleaning, feature engineering. Even though all the models have achieved good results, they have their own advantages and disadvantages. The Decision tree Algorithm has the lowest error rate on the train data but it is prone to overfitting, Random Forest solves the problem of overfitting to some extent. Time complexity of Random Forest, Gradient boosting algorithm is high as the data has to fitted multiple times. The Linear Regression is a decent method when the data is linearly separable and is good in terms of time complexity. Though Linear Regression has slightly more error rate when compared to Random Forest Algorithm it is much better when we compare the time complexities. So, There is a tradeoff here between these algorithms in terms of accuracy and time complexities. Linear Regression Algorithm will be a better choice when the data is linearly separable and in case of low latency scenarios. More research is needed on the following topics to further study these models, especially the combination of different models:

- Coupling effects of multiple regression models.
- The ability to "re-learn" of machine learning models.
- Combination of machine learning and deep learning methods.
- Factors that drive good performance of tree-based models
- The fastest way to fit complex models
- Effects of addition of L2 norm or L1 norm to linear regression.

Table 1: Metrics for different algorithms

Algorithms	MSE	RMSE	R2 Score
Linear Regression	0.0156138	0.1249553	90%
Decision Tree Regression	0.0454337	0.2131518	84%
Random Forest Regression	0.0028343	0.0532385	98%
Gradient Boosting Regression	0.0156138	0.1249553	91%

REFERENCES

1. Cock, Dean. (2011). Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project. Journal of Statistics Education. 19. 10.1080/10691898.2011.11889627.
2. Lu S, Li Z, Qin Z, Yang X, Goh RSM. A hybrid regression technique for house prices prediction. 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM) 2017. doi:10.1109/ieem.2017.8289904.
3. S. Raschka and V. Mirjalili, Python Machine Learning, 2nd ed. Birmingham, UK: Packt Publishing, 2017.
4. Prediction of House Price Based on The Back Propagation Neural Network in The Keras Deep Learning Framework. 2019. 2019 6th International Conference on Systems and Informatics (ICSAI).doi : 10.1109/ICSAI48974.2019.9010071.
5. PedregosaF,VaroquauxG,GramfortA,MichelV,ThirionB,GriselO,etal .Scikitlearn:MachineLearninginPython.TheJournalofMachine Learning Research 2011;12:2825–30.
6. L. Breiman. Random forests. Machine learning, 45(1):5–32, 2001

AUTHORS PROFILE

Jyothi vishnu Vardhan kola, born in 2001 and is currently pursuing his BTech in computer science and engineering at Gitam university vishakapatnam. He is interested and enthusiastic about working on improving lifestyle of people through technology. He has a great deal of knowledge and experience with End to End machine learning applications. He has worked extensively on building better machine learning algorithms for various use cases. He also has an experience of teaching assistant, helping learners to understand the concepts of Machine learning and Data Science. Intelligence. His future goals include extensive research in the field of Data Science, especially in the areas of Computer Vision and Natural language processing , also working on applications of Data Science in solving crucial problems.

Poorna Chandra Vemula, born in 2001 and is currently pursuing his Bachelors' of Technology in computer science and engineering at vellore institute of technology, vellore. He worked on various research projects in the areas of Data Analytics, Machine Learning, Natural language processing and Computer Vision. He has experience working on real world applications dealing with large scale systems, and built numerous apps taking into account current demands of the stakeholders. He also has mentorship experience, explaining concepts in the field of Data Science. He is optimistic about the future developments in AI and looking forward to collaborating on solving problems primarily in the areas of Health, Agriculture, Education and sustainable development using Artificial Intelligence.

Vanapala Sai Mohit, born in 2000 and is currently pursuing my Bachelors' of Technology in computer science and engineering at gitam institute of technology, visakhapatnam .I am Passionate about simplifying things through technology. I had good understanding on python programming language , Machine learning, AWS cloud computing and data Analytics . I also worked on some real world complex data set and done various internships in the field of data science . I always love to slove problems that involve creativity and innovation. I found myself gravitating towards problems that involved analyzing and processing huge amounts of data.