MERCARI PRICE SUGGESTION

Team : The Drifters

Hruthik Chevuri
IIIT
BANGALORE
IMT2018507
hruthik.chevuri@iiitb.org

Anirith Pampati
IIIT
BANGALORE
IMT2018516
anirith.pampati@iiitb.org

Rohan Kumar
IIIT
BANGALORE
IMT2018515
rohan.kumar@iiitb.org

***Abstract*—In the online shopping market, pricing of any online product is very crucial. It affects numerous aspects in the business such as finances, marketing and sales. An inappropriately priced product could lead to your company's death while an appropriately priced product could place you as a prime competitor. Mercari, Japan's biggest community-powered shopping app, knows this problem deeply. They'd like to offer pricing suggestions to sellers, but this is tough because their sellers are enabled to put just about anything, or any bundle of things, on Mercari's marketplace.**

In this project, we are trying to build an algorithm using traditional machine learning algorithms to suggest right prices for the products in the market.

***Index Terms*—Pre-processing, Label encoding, One hot encoding, feature Scaling, Regression, Random Forest Classifier, Ridge regularization, hyper parameters, Random forest regressor, LGBM regressor, XGB regressor.**

I. Introduction

In this decade, There is a huge growth in the online shopping market. there is a lot of advancement in the technology. The number of people using internet, laptops, mobiles and other devices are increasing exponentially. As internet is available to more number of people, more people are willing to buy products online. since a difference can be observed in terms of pricing compared to the offline market, this pricing attracts more people for online shopping. As the interest in online shopping increases year by year, there is a heavy competition formed in each and every category and in every segment. Therefore an appropriate price should be given to a product to sustain in this huge competition.

Mercari is an online shopping marketplace which is powered by one of the biggest community of Japan where users can sell pretty much anything. An appropriately priced product could place you as a prime competitor. So it is very crucial to sell the product at the right price. So in this project, by observing the price trends with respect to brand name, category name and description etc. from the collected data and suggest a price to the new product that the seller wants to sell.

The challenge here is to build an algorithm that automatically suggests the right product prices. You'll be provided user-inputted text descriptions of their products, including details like product category name, brand name and item condition.

The report tells the methods and analysis of the large information that has been provided by mercari for this price suggestion challenge. It also tells the methods and traditional machine learning algorithms used for suggesting the right price for the item. We also included our approach in deciding our final model.

The report will discuss these topics as follows: **Sec-II** discusses the beginning and some related work on the price suggestion. In **Sec-III** , we describe our dataset and it's features.**Sec-IV** shows our visualizations for understanding our dataset.**Sec-V** describes about the pre-processing steps taken to make our data ready for building our model and also feature scaling. **Sec-VI** describes our model.**Sec-VII** will describe the parameters of the regressors we have taken and **Sec-VIII** has references and conclusion.

II. Related Work

In the online shopping , the consumers have the luxury to compare the prices offered for the same product by all the sellers and select the product which is reasonable with respect to the price. This scenario is different from the offline market. So pricing an item in the online market has always been a critical task to stand in the competition.

Machine learning based price suggestions. Recently machine learning models like linear regression with ridge

regularization, XGBM regression etc.. has been used to predict the appropriate price.An article on Price Suggestion and Price Prediction Algorithms[1] uses fast and high-performance decision tree algorithm called Light GBM for price suggestion which can be a solution to such kind of issues. This has been our inspiration for our approach for this price suggestion problem. While our work focuses on the price suggestion of the product to the seller, this article describes various algorithms for price suggestion and stock price prediction.

III. DATASET AND EVALUATION METRIC

With more than million consumers and Japan's largest community-powered marketplace with over JPY 10 billion in transactions carried out on the platform each month and more than 100000item updates per day, mercari is trying to suggest automatic prices for their online sellers in the website. The dataset used in this project is the mercari price suggestion dataset which is hosted on kaggle .

Each datapoint in the training data consists of 8 coloumns. There are 1037774 datapoints in the training data and the primary test data consists of 444761 datapoints for which the label 'Price' is not released which has to be determined.

Given all the columns of the data, the column named 'category name' can be divided into sub categories. Also there are 8 features in the training data, which consists of columns with different data types like objects, integers.

Description about features of the dataset:

These are the columns:-
train id : unique id for all taraining datapoints.
name : name of the product.
item condition id : Describes the present condition of the product on a scale from 1 to 5.
brand name :brand name of the product.
item description : Little descripton about the product and it's features.
shipping : it is a binary variable either 0 or 1.
category name : Provides which category the product belongs.

In the category name column, the string can be splited by using 'ínto 3 sub-levels. We have named them as main category, sub category-1 and sub category-2.for example, if the data in the category name is :-

Vintage & Collectibles/Electronics/Video Game

Here the main category is the vintage and collectibles, sub1 is electronics and sub2 is video Game. In this the category name column can be splited into 3 diferent features.

Evaluation:

The evaluation metric for this competition is root mean squared logarithmic error(RMSLE).

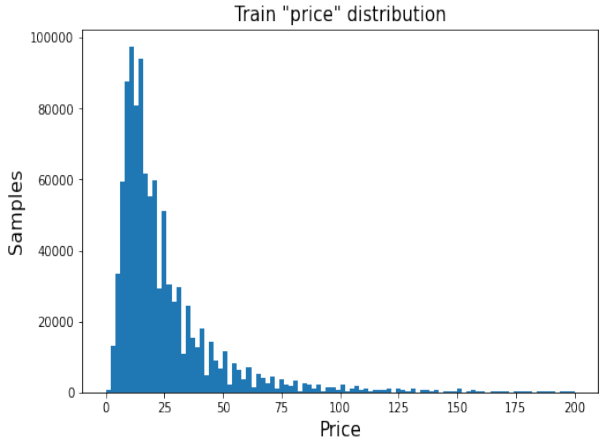
$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(p_i+1) - \log(a_i+1))^2}$$

Where:
n is the total number of observations in the (public/private) data set, pi is your prediction of price, and ai is the actual sale price for i. log(x) is the natural logarithm of x.

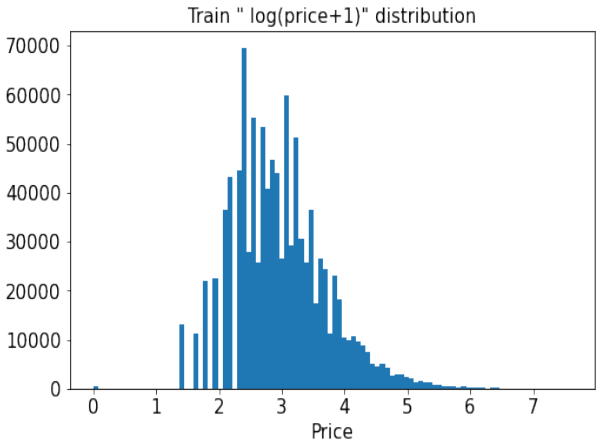
IV. OBSERVATIONS AND DATA ANALYSIS

In building any machine learning model,we need to observe some trends of features in the dataset which helps us in understanding the distribution of the data. So here are few of those observations:-

A. *Hist plots of price distribution*



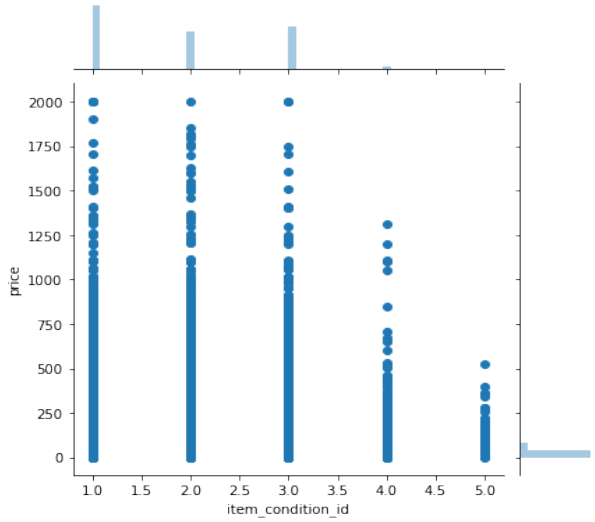
From this distribution plot, we can infer that the it is heavily skewed to the left. so we can remove the skewness by applying logarithm to it. Then the distribution of price is changed as follows. In this plot, the price is centerd around 3 and has longer tail on the right.



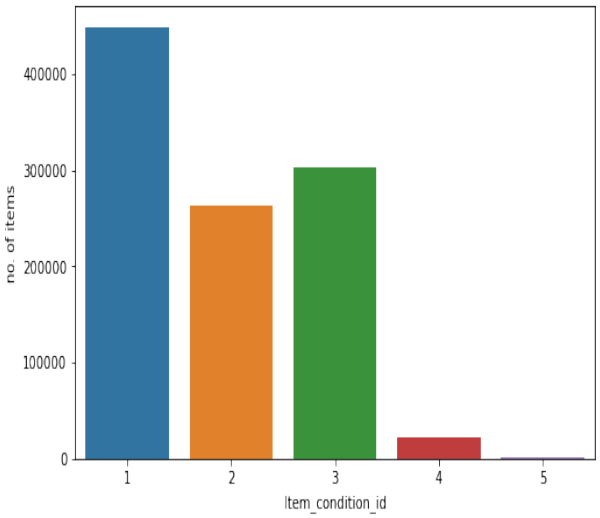
So here there are many ways for skew removal like square root transformation, reciprocal transformation, Bx-cox transformation, log transformation etc.. we have applied logarithmic function to do that job.It is important for us to remove skewness . If it is skewed, it will be trained on moderate values and it is less likely to successfully predict the price of the expensive products in this example. Transforming skewed data is one critical task in data cleaning step before proceeding with data preprocessing.

B. *Price distribution wrt item condition*

The feature column item condition id has taken 5 different values. on a scale from 1 to 5. 1 means that the item is new and unused and the number increases based on the condition of the product.

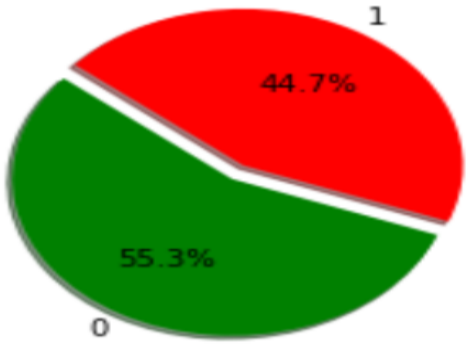


As expected , the items with condition id = 1(new item) will be at a higher price and the items with condition id large has less price. Therefore the product condition will impact it's selling price. This can be inferred from the plot.

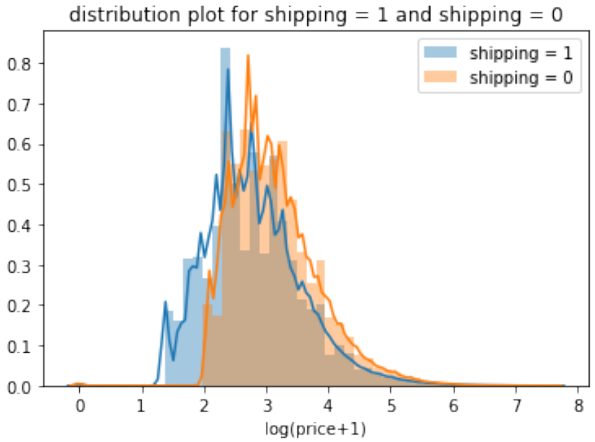


There are more products whose item condition is new than the used products.i.e the count of items with item condition =1 is more than the remaining. Here the count of new items is more than 400000. The products with worst item condition are relatively very low.

C. *Shipping w.r.t price*



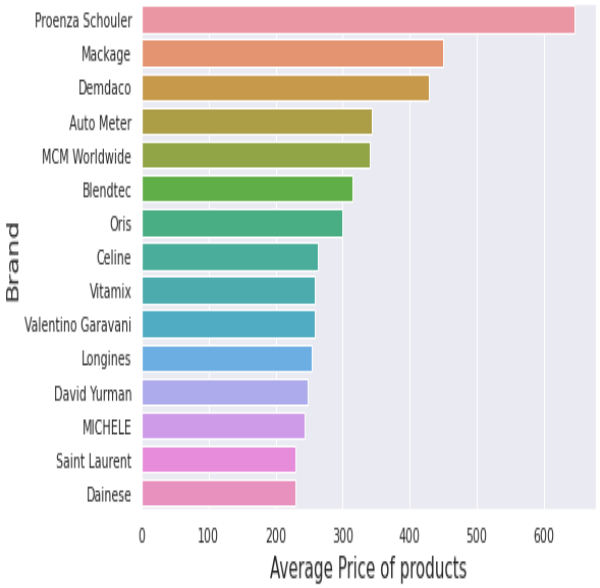
Approximately more than 55% products don't have shipping charges. Now we will see how price of the product varies as this binary variable shipping changes.



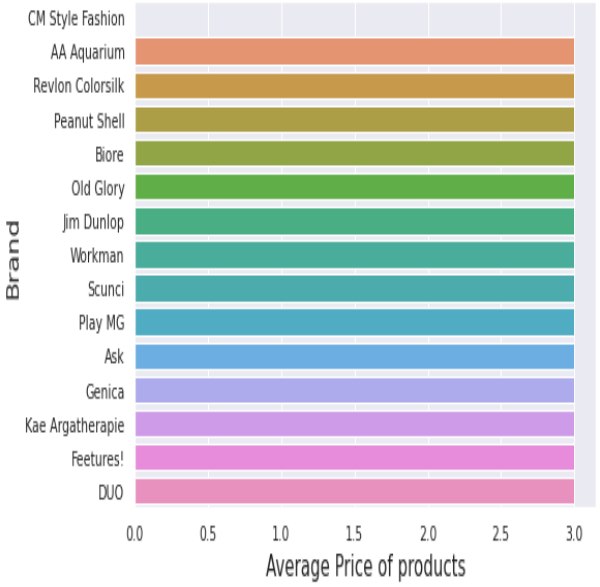
distribution plot of shipping with 0 has high peakedness than the pdf plot of shipping with 1. This plot is plotted using distplot in seaborn library.

D. *price distribution plot of different brands*

There are millions of data points (products) are given here. Out of them there are totally 4414 different brands. So based on average price of products of each brand , we can get premium brands and cheapest brands available in the mercari marketplace.



The above barplot shows top 15 brand names in the mercari dataset for which the average price of products belonging to the brand is highest. So these brands are expensive in general because the products belonging to this brand have higher price than the others in general. So we can observe some premium brands available in mercari marketplace from this barplot.



Similarly the above barplot shows the 15 brands with least average price from the total 4414 brands in the marketplace. So we can say that these brands provide products that are of less cost compared to remaining brands. Nothing important related to model can be inferred from here, but some information about these brands.

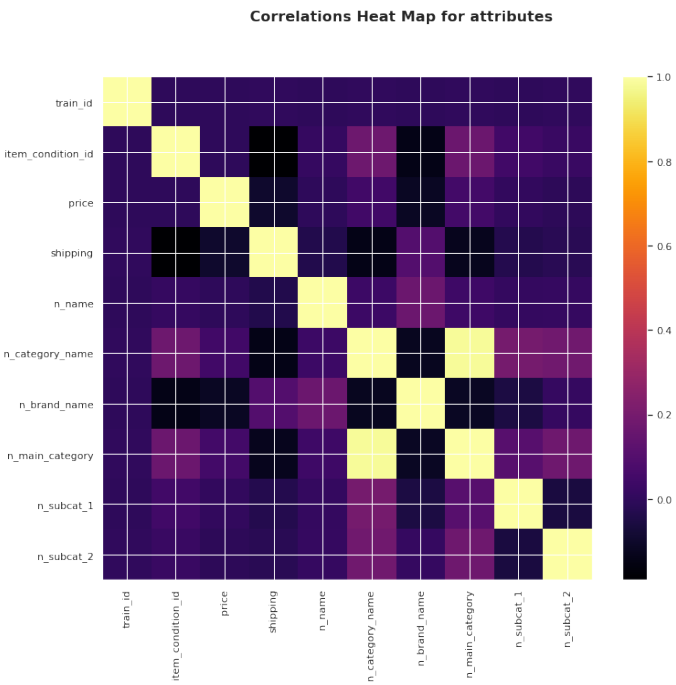
E. Wordclouds of name and description feature

Name and item-description are text fields. There is a necessity to understand the kind of data we are dealing with. Apart from the observed data distribution. Word Clouds (also known as wordle, word collage or tag cloud) are visual representations of words that give greater prominence to words that appear more frequently. these will be useful in understanding frequency of a word in the description section and in name feature.This type of visualization can help presenters to quickly collect data from their audience, highlight the most common answers and present the data in a way that everyone can understand.

wordcloud of name feature:

wordcloud of item-description:

The importance of each word is being shown in the word cloud with font size of color. Nothing much can be inferred from these wordclouds apart from the kinds of words we are dealing with here.



Correlation is not good for attributes to predict price. The above correlation matrix shows the correlation of different features. it is implemented with the help of sns heatmap.

V. DATA PREPROCESSING

Data preprocessing is a technique for transforming our collected raw data. This raw data may be skewed, different data types, consists of noise and some values may be even incomplete.The raw data may even consist of numbers in string format. Our aim through preprocessing is to transform this raw data into an understandable format with proper interpretations.

As we know that the efficiency of our machine learning model highly depends on the quality of the data that we give to it for training. Therefore, preprocessing step in training our model plays a vital role in improving our overall model performance.

We have done these preprocessing steps to transform our raw data in a better way. 1) Filling the missing values 2) spliting the categeory-name feature into sub -columns 3)Categorical encoding(one hot encoding and TF-IDF vectorizer) 4) Feature scaling

Filling the missing values: As we know that the collected raw data may consist of many missing values in many different columns. We can fill the missing values for different features by creating a new class ' Not given' or using mean or mode of the column which suits the column better. We have filled the missing values in various features using a new class label 'Not given'. If the missing value percentage in the column is large, it might not be a right decision.but after trying to fill these null values with mean, mode etc.. we found that filling the missing values with this new class label yield better results. In the category name column, the string can be splited by using 'ínto 3 sub-levels. We have named them as main category, sub category-1 and sub category-2.

Columns with missing values	
Feature name	Filling values
Brand	No brand given
Category name	No main categ given/ No sub 1 categ given/ No sub2 categ given

Next after filling the nulls, we tried few pre-processing steps as below. 1)Contractions:- There are many words like "will not" "I am" in the document, so we tried to convert them to "won't", "I'm". But the accuracy was better before contractions so we commented that steps. 2) pre-processing:- we did few things like removing new lines, extra spaces, everything other things than alphabets and numbers like emojis and etc. But there presence was important because they also create a difference between 2 descriptions. So at last we decided to keep everything without removing them to increase the difference between descriptions and improve score. 3)Stemmer:-Stemming is basically reducing a given word into its root-base form. We used porterstemmer function. 4)Deep pre-processing:- we also did a two more important steps like (a)Removing Punctuations, we tried removing punctuations. (b) Removing stop words. We wrote a function to remove stop words but then we saw that TF-IDF has in-built function to remove stop words so we used that inbuilt facility (added the stop-words line into tf-idf) and removed the function we wrote.

Categorical encoding : Any structured dataset may contain a mixture of numbers and text. But a machine can understand numbers but not text. So there is a necessity to convert these text into some numerical format. Categorical encoding is a process of converting categories to numbers.

About Label Encoding : Label Encoding is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering. But when the label encoding is performed, it inherently gives some ordering of the data. Due to this, there is a very high probability that the model captures the relationship between data such that class1 ¡ class2 ¡ class3 etc. It is better to encode in this format if there exists some order for the class labels.

About One hot encoding : One-Hot Encoding is another popular technique for treating categorical variables. It simply

creates additional features based on the number of unique values in the categorical feature. Every unique value in the category will be added as a feature. If there are n unique values , it will create n additional features. It increases the computation cost but it will not give ordering like label encoding. As there are some features like brand name , categories and sub- categories where there is no implicit ordering between the classes of the features. One hot encoding can be implemented for these features

About TF-IDF vectorizer : TF-IDF is an abbreviation for Term Frequency Inverse Document Frequency. This is very common algorithm to transform text into a meaningful representation of numbers which is used to fit machine algorithm for prediction. So it is a statistical method to show the importance of word in the corpus. TF*IDF is an information retrieval technique that weighs a term's frequency (TF) and its inverse document frequency (IDF). Each word or term that occurs in the text has its respective TF and IDF score. The product of the TF and IDF scores of a term is called the TF*IDF weight of that term. Put simply, the higher the TF*IDF score . These TF-IDF vectors are calculated in the following way:

- Calculate term frequency of each document.
- Calculate the inverse document frequency (IDF): Take the total number of documents divided by the number of documents containing the word.
- Calculate TF-IDF: multiply TF and IDF together.

this vectorizer is used to transform name and item-description features. we used TF-IDF vectorizer over train data in order to capture all the words occurring in the data with maximum features as 10,00,000 for an n gram range of (1,3). And later stacked these matrices using scipy's hstack.

VI. Model Selection

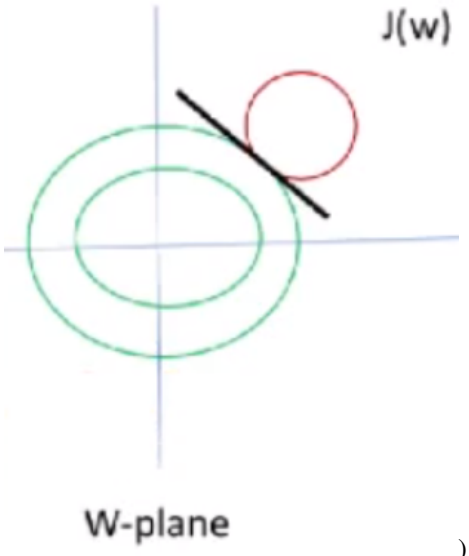
So far we have done preprocessing to our dataset to train our model better. The important task here is to implement suitable machine learning model to the problem which gives better results. Since it is a regression problem, We tried out basic linear regression initially. Regression is a statistical analysis for estimating the relationship between a dependent variable(target variable) and one or more independent variables(features). we have seen the concept of simple linear regression where a single predictor variable X was used to model the response variable Y . In many applications, there is more than one factor that influences the response. Multiple regression models thus describe how a single response variable Y depends linearly on a number of predictor variables.The aim in the process of linear regression to express the target variable as the linear combination of given features. Our aim is to find a weight vector which minimizes the cost function. Here the choice of cost function in linear regression is least squares cost function. here J(W) is the cost function to be minimized which is the sum of the squares of the residue.

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

By using this linear regression model , we got the metric score RMSLE for public leaderboard to be 0.55230. Then we approached towards regularization of the regression model using Ridge regularization. Here the cost function will be modified as follows

$$V(w) = J(w) + \lambda ||w||_2$$

Here lambda is a regularization coefficient (hyper parameter) to control the influence of regularizer.



)

Here the red contour is the initial cost function of the linear regression model and the green contour is the regularization parameter(second norm).We want to reach the minimal Value of V(w). What happens at the minimal value of V(w).

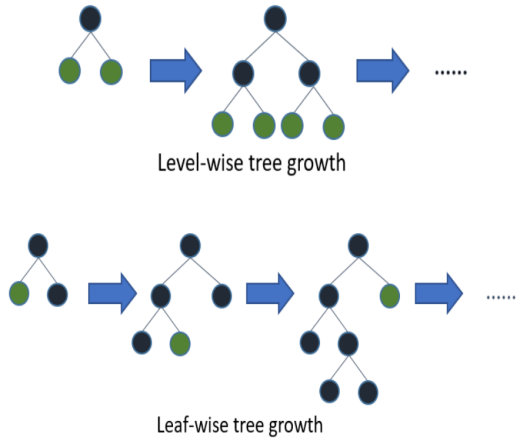
such points of Intersection: If you move away from the Tangential Intersection point, Upwards: R Increases, Hence V will increase. If you move away from the Tangential Intersection point, Downwards: J Increases, Hence V will increase. Hence these intersection points of two contours can be the solution points. By modified the cost function through ridge regularization RMSLE score reduced to 0.48907 for the public leaderboard in the kaggle competition.

We started to approach towards ensemble techniques like bagging, boosting techniques to improve the accuracy. We tried out with a bagging technique , random Forest regressor: Its is a classification/regression algorithm which consists of large number of individual decision trees that operates as an ensemble.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. A large number of relatively uncorrelated decision trees operating as a committee will give better performance than the individual decision trees. The important point to remember is that the decision trees are most sensitive to the training data. A slight change in the training data may result in significant change in the tree's structure.

Bagging can be thought as a committee of experts and boosting can be thought as a group of specialists. Both make the final decision by averaging the N learners but in case of bagging, it is an equally weighted average. In case of boosting, more weight will be to those learners which perform better. We can't say which ensemble technique outperform other technique but it will be based on the data. There are several ways to determine these weights in the boosting ensemble.

We tried those boosting ensemble technique XG-boost and Light-GBM. XGboost is an advanced implementation of gradient boosting algorithm(ensemble that works for both regression and classification problems). It also has regularization which reduces the chance to overfit and improves overall performance. Coming to the LGBM, it is also a gradient boosting ensemble technique but it is useful when the dataset is huge. The difference of LGBM from other gradient boosting algorithms is that it follows leaf-wise approach while other algorithms work in a level-wise approach. Leaf wise growth may overfit the model. so we need to take care for the max_depth parameter It can be understood better as below:



This picture has been taken from [8] These boosting techniques have significantly improved the model accuracy and also LGBM regressor takes less time to run on our dataset. This improved our model accuracy. By using LGBM regressor, we have got RMSLE score to be 0.46550.

Then we have came up with a model which is a weighted average for ridge regression and LGBM regression .Weighted average is averaging both the results but with specific weights assigned to each. Initially we tried with different weights for these two regression models. we gave 0.8 for the ridge regressor and 0.2 for the LGB regressor and computed the target variable and the RMSLE score got down further to 0.44 for the public leaderboard. Then we tuned for better weight combinations and we got better result when they are of equal weights. 1.e simply the average of both the regressors.

PARAMETER TUNING

We have selected the model which gives better accuracy in predicting the result.But we also need to take care of the parameters for our ridge and LGBM regression models.Changing these parameters might even overfit /underfit the model to the dataset. Do utmost care must be taken while tuning the parameters.

For ridge regressor : In this process we need to tune a single parameter to control the influence of the regularization term. if the term is approaching zero, then it will become normal linear regression , then it will not have any improvement for generalization. if the value of the hyper parametr is large, then it may cause underfitting which is not expected. So care must be taken while setting the parameter value. There are many ways that can be followed to estimate the parameter value.

For LGBM regressor: A similar approach as mentioned above is taken while tuning the parameters for this classifier too. 1) Fixing the learning rate.2) Fixing the tree specific parameters like setting the maximum depth limit for the tree. So the final parameters set are : " LGBMRegressor(max-depth=10, n-estimators=2000)".

The following is the table of evaluation metric of all approached algorithms implemented:-

Evaluation metric(RMSLE)		
Model	public leaderboard	public leaderboard
Linear regression	0.56659	0.55230
Ridge regression	0.48907	0.48957
LGBM regression	0.46550	0.46556
weighted average of LGBM and ridge	0.43877	0.43948
equal weighted average of both	0.43636	0.43666

Here the last model that we implemented is the late submission in the kaggle so the score is not reflected in the leaderboard score

VII. CONCLUSION

We would like to conclude that we have tried out the possible ways that we know so far to build the model that suggests right price to the sellers for the given mercari marketplace dataset. Projects like these have a scope of

improvement through advance machine learning models.

FUTURE SCOPE FOR IMPROVEMENT

Implementing some advanced machine learning techniques like neural networks implemetation might have yield better results.
Some other optimization techniques like multi core processing might give better result.

VIII. ACKNOWLEDGEMENT

We would like to our Professors G. Srinivas Raghavan, Neelam Sinha and our Machine Learning Teaching Assistants Amitesh Anand for helping us in the initial stage of the project and pushing us forward to complete the project and improve our model with new methods. The TA sessions were also very useful in understanding the concepts better so that they are useful in our project.

Leaderboard positions motivated us in working towards better optimizations everyday. We would like to thank our fellow teams for maintaining healthy competitions in the leaderboard. It was a good experience working with this project and learnt more while crossing some obstacles in the project.

IX. PROJECT LINK

The link for our project files are available at https://drive.google.com/drive/folders/1SVPU-mAy14sxQZ2heZX1dMrbR16v6rBX?usp=sharing

REFERENCES

[1] P. Raja Mani , Mallu Tejasri, B. V. S. Aneesh Kumar, Kunal Naskar, Raj Suvam Bisht proposed some Price Suggestion and Price Prediction Algorithms

[2] Santos et al. [22] proposed a hybrid malware detection tool based on machine learning algorithms called OPEM that utilizes a set of features obtained from static and dynamic analysis of malicious code.

[3] Akhil Kadiyala and Ashok Kumar. “Applications of python to evaluate the performance of decision tree-based boosting algorithms”.

[4] ALAKH SETHI, “One-Hot Encoding vs. Label Encoding using Scikit-Learn” 2017, [Online; posted, MARCH 6, 2020,].[Online]. Available: https://www.analyticsvidhya.com/blog/2020/03/one-hot-encoding-vs-label-encoding-using-scikit-learn

[5] Jason Brownlee, “How to Tune the Number and Size of Decision Trees with XGBoost in Python” 2017, [Online; posted,September 7, 2016].[Online]. Available:https://machinelearningmastery.com/tune-number-size-decision-trees-xgboost-python

[6] Scikit-learn documentation , ” GradientBoostingClassifier”

[7] Jason Brownlee, “HHow to Configure XGBoost for Imbalanced Classification” 2017, [Online; posted, February 5, 2020].[Online]. Available:https://machinelearningmastery.com/xgboost-for-imbalanced-classification

[8] AARSHAY JAIN, “Complete Guide to Parameter Tuning in XGBoost with codes in Python” 2016, [Online; posted,, MARCH 1, 2016,].[Online]. Available:https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/

[9] Guolin Ke et al. “Lightgbm: A highly efficient gradient boosting decision tree”. In: Advances in Neural Information Processing Systems. 2017, pp. 3146–3154

[10] Cory Maklin ” TFIDF - python example towards datascience”.[Online; posted,May 5, 2019].[Online]. Available:https://towardsdatascience.com/natural-language-processing-feature-engineering-using-tf-idf-e8b9d00e7e76

[11] Marius Balcanu, ”How to price your online products”. [Online; posted,September 23, 2020].[Online]. Available:https://blog.2checkout.com/how-to-price-a-product