BANA 212 - Final Report

MERCARI PRICE SUGGESTION CHALLENGE (Kaggle competition)

Group 16

Bowen Yang

Justin Yeung

Swathi Mudhelli

Viktoriia Pinsker

Tzu-Ying Yu (Winnie)

Table of Contents

Abstract

During the last decade, the e-commerce market has been growing exponentially as the platform for purchasing goods has transitioned from retail stores to an overflow of online selling platforms. Especially in the time of the pandemic, people prefer staying at home to buy items ranging from electronics, clothes, and even groceries instead of venturing out to stores or shopping malls. With the increase in online shopping traffic, users have also flocked to buying and selling pre-owned products online like never before.

One considerable problem with the exchange of pre-owned products in comparison to new products is their pricing. Users in online marketplaces, especially first time sellers, often find it challenging to price their products. The advance of online marketplaces has led to the appeal in writing algorithms to help all sellers with price recommendations for their items. More than that, the rise of machine learning and big data in predictive modeling has boosted the relevance of using them to revise the pricing in the pre-used products sector. By getting a solution for price prediction through product qualities for B2C and C2C online retailers, sellers will have an easier process and it will help increase the number of sellers in user-based marketplaces. This can also be a major competitive advantage for organizations or individual sellers having exceptionally accurate pricing decision-support.

In our project, we try to build a price suggestion model for Japan's biggest community-powered shopping platform, Mercari. The goal is to predict the prices of different products by using the item description and other textual features without looking at the image of the product. We will perform exploratory data analysis to understand our data and use analytical techniques to clean and prepare the data for modeling. We further built LightGBM and Ridge regression models to predict the price of an item based on the categorical and textual features.

The first half of our report is focused on the executive summary, our primary objective, data explanation, and the description of the models we utilize. The second half of our report highlights the steps of our algorithm and ends with the key takeaways.

1.1 Executive Summary and the Objective

The project is based on Mercari price suggestion challenge hosted on kaggle by Mercari.

Mercari is a Japanese online marketplace that allows sellers to sell easily, ship, and earn money for used products they do not need anymore. With close to a million downloads on the App Store, Mercari is a highly sought-after marketplace to buy and sell used fashion, accessories, and hundreds of other product types.

Mercari would like to grow and sustain this community, and product pricing is at the core of this marketplace. It is challenging to predict what a pre-owned product is worth, even harder in the absence of an image. It gets even harder at scale, considering just how many products are sold online. Factors influencing product price vary by category; clothing is heavily influenced by brand names while electronics are influenced by product specs.

Mercari would like to offer pricing suggestions to sellers, but this is tough because sellers are enabled to put just about anything, or any bundle of things, on Mercari's marketplace. The idea is to build a model that automatically suggests the right product prices based on attributes.

What variables can influence the price of a product? For example, one of these sweaters costs $335 and the other $9.99. Can we guess which one's which without the image of the product?

Sweater A:	Sweater B:
"Vince Long-Sleeve Turtleneck Pullover sweater, Black, Women's, Size L, great condition"	"St. John's Bay Long-Sleeve turtleneck, Pullover sweater, size L, great condition"

Two-sided marketplaces like Mercari face unique challenges - they have to attract both buyers and sellers. But often, sellers want to sell at high prices and buyers want to buy at low prices. So it becomes essential to strike a balance between the motivations of sell-side and buy-side with the best prices. Sellers don't have the sophistication to analyze pricing or the effectiveness of apt pricing. An individual merchant might not know what price to charge. Even if he could do the analysis, he wouldn't have enough data points. Mercari can help them with pricing suggestions because of access to the aggregate data from all sellers and products on the site, which can help optimize pricing (Croll and Yoskovitz). Price is the most influential influencer of engagement on the platform. Mercari can attract more buyers if the products are priced well. More buyers will bring more sellers, which leads to higher engagement on the platform. In a way, the success of the platform depends on pricing.

As Mercari strongly advocates for simplicity in selling, it can be insightful to understand which variables are most important so that sellers can utilize those in attracting more attention and price the items better. Alternatively, this also assists buyers in recognizing when a particular item may be undervalued.

Through this project we would like to answer the following questions:

1. Can we predict the price of a secondhand product with user input text information (without the image)?
2. Which variables are the most reliable predictors of the price?
3. What are the top 3 variables that influence the price most?
4. Does adding a detailed item description impact price?

The evaluation metric is Root_mean_squared_log_error, also called RMSLE, with the following formula. The objective is to get the lowest error rate.

The RMSLE is calculated as

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

Where:

ϵ is the RMSLE value (score)
n is the total number of observations in the (public/private) data set,
p_i is your prediction of price, and
a_i is the actual sale price for i.
$\log(x)$ is the natural logarithm of x

1.2 Data and Visualizations

The original dataset for Mercari had 700,000 product listings in a text file format. However, we decided to work with a subset of 85,000 listings for our project and had two separate datasets, one containing for training the model and the other as a testing set. The variables included in both datasets were:

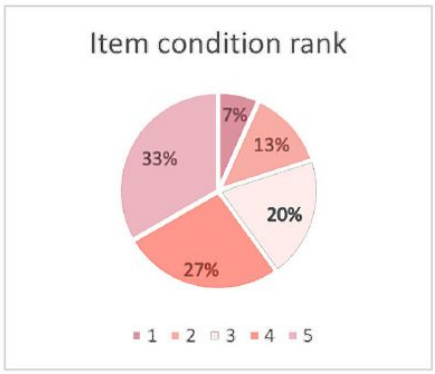
- Item Id
- Name - user input name of the product
- Item Condition (Ranked 1-5 with 5 being highest rated)
- Category Name
    - Main Category
    - Sub Category 1
    - Sub Category 2
- Brand Name
- Shipping (Binary variable equal to 0 if seller paid shipping fee or 1 if buyer paid shipping fee)
- Item Description - user input description of the product

The training set also provided an additional response variable, price, which was used to train the model and predict prices using the test set. Below is an example of the testing dataset.

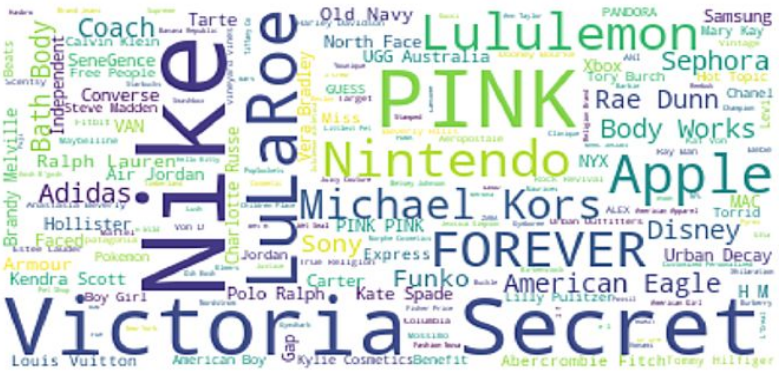
	test_id	name	item_condition_id	category_name	brand_name	shipping	item_description	main_category	subcat_1	subcat_2
0	0	Breast cancer "I fight like a girl" ring	1	Women/Jewelry/Rings	NaN	1	Size 7	Women	Jewelry	Rings
1	1	25 pcs NEW 7.5"x12" Kraft Bubble Mailers	1	Other/Office supplies/Shipping Supplies	NaN	1	25 pcs NEW 7.5"x12" Kraft Bubble Mailers Lined...	Other	Office supplies	Shipping Supplies
2	2	Coach bag	1	Vintage & Collectibles/Bags and Purses/Handbag	Coach	1	Brand new coach bag. Bought for [rm] at a Coac...	Vintage & Collectibles	Bags and Purses	Handbag
3	3	Floral Kimono	2	Women/Sweaters/Cardigan	NaN	0	-floral kimono -never worn -lightweight and pe...	Women	Sweaters	Cardigan
4	4	Life after Death	3	Other/Books/Religion & Spirituality	NaN	1	Rediscovering life after the loss of a loved o...	Other	Books	Religion & Spirituality

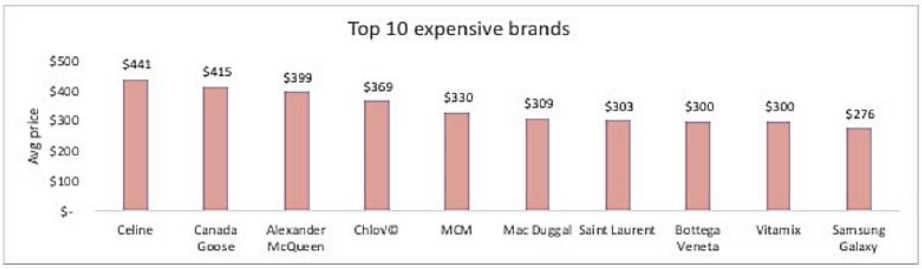
Analysis:

From our dataset of the 85,000 products, we found there were 1,652 different brands of items sold. After separating the “category_name” variable into “main_category” and two subcategories, we found that there were only 10 distinct main categories, and 762 different subcategories. Among these products, the majority of item conditions were ranked highly, receiving either a 4 or a 5 and the majority of shipping fees were often paid for by the buyer.

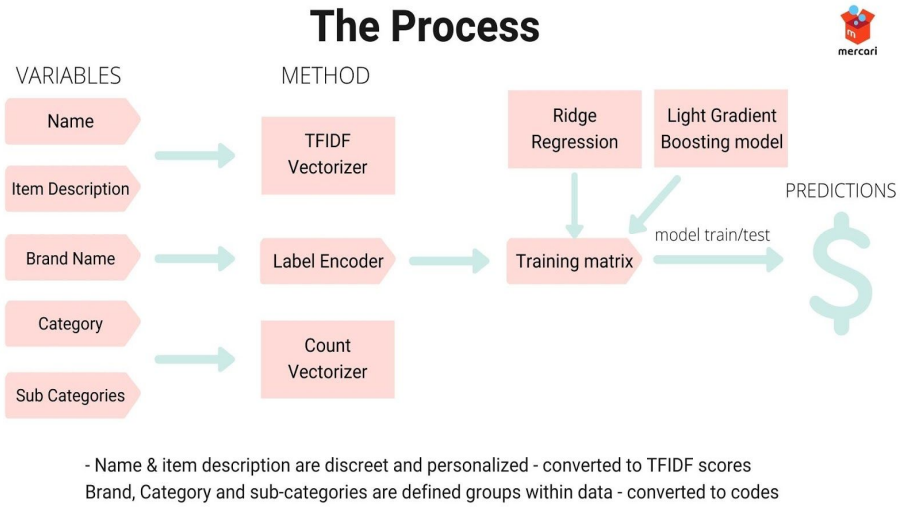


To gain further insight on the dataset, we examined the top brands that were being sold on Mercari. Using a word cloud, we can visualize which brands are most popular and using a bar graph, we can see which brands ranked highest in price. Looking at the bar graph, we can determine that the most expensive brands are either women products or electronics. This is also reflected in the word cloud where the majority of popular items appear to be clothing or electronics related.





Process:



The first step was to clean the dataset. Approximately 20% of the dataset contained missing values in brand name, price and item description column. These variables are very important in predicting price. We first took our dataset's variables and used various different methods to create our models. After getting these models, we can apply our testing set and make price predictions.

1.3 Models Explanation

Ridge Regression:

Ridge Regression is a variant of linear regression. The reason why we use Ridge Regression is that we prefer a model that catches general patterns to predict the price of a second hand product. The other reason is that we aim at predicting it from new data, not specific data. Therefore, our model evaluation is based on a testing set, not a training set.

Since our dataset contains text classification problems which tend to be high dimensional and likely to be linearly separable, ridge regression seems to be a choice in our project (Marsupial). It controls the complexity of the classifier and helps to avoid overfitting issues. Unlike the OLS method does where an unbiased model finds the coefficients that best fit the training data and the unbiased coefficients to cause overfitting issues, ridge regression can accept its variables unequally to treat each predictor differently by tuning the lambda parameter so that model coefficients change (Qshick). When the lambda equals to zero, ridge regression equals least squares regression.

Light Gradient Boosting method (LGBM):

One of the most effective techniques for creating predictive models is gradient boosting. In the sense that models are optimized by gradient descent on generic differentiable loss functions, but also enjoy the natural benefits of tree-based models such as automatic null-handling, automatic selection and interaction of features, invariance of scale, and ability to capture highly non-linear feature-target related functions (Eddy). These features make gradient-boosted models very robust and very easy to train for all kinds of distribution among the variables in the project. Boosting algorithms often typically perform very well for most tabular datasets in the first place.

Among gradient boosting libraries such as LightGB, XGBoost, CatBoost, LightGB, using tree-based learning algorithms, seems to be the quickest and most versatile one (Eddy). In particular, LGB uses highly optimized binning of histograms to speed up the process

of tree-splitting (Eddy). Also, it's fast to build, uses low memory and is highly accurate. Therefore as our second algorithm, we chose LGBM.

2.1 Building the Models

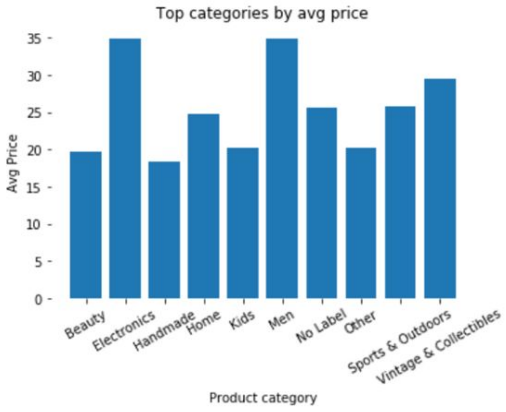
After reading the tsv files into python, the first step was to create categories and subcategories from the category_name column. We created a user-defined function to split the “category_name” column into “main category”, “subcat_1”, and “subcat_2”. This will allow a clearer understanding of product categories and their subcategories.

```
[7]: train_copy.head()
```

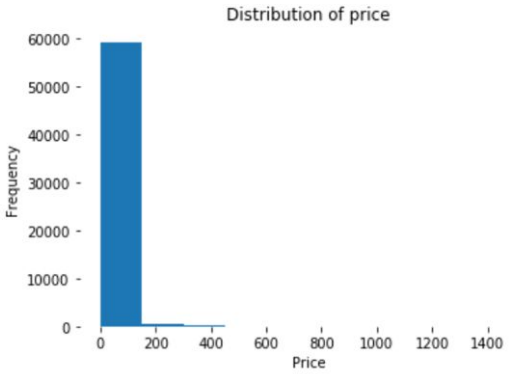
[7]:

	train_id	name	item_condition_id	category_name	brand_name	price	shipping	item_description	main_category	subcat_1	subcat_2
0	1	Razer BlackWidow Chroma Keyboard	3	Electronics/Computers & Tablets/Components & P...	Razer	52.0	0	This keyboard is in great condition and works ...	Electronics	Computers & Tablets	Components & Parts
1	2	AVA-VIV Blouse	1	Women/Tops & Blouses/Blouse	Target	10.0	1	Adorable top with a hint of lace and a key hol...	Women	Tops & Blouses	Blouse
2	3	Leather Horse Statues	1	Home/Home Décor/Home Décor Accents	NaN	35.0	1	New with tags. Leather horses. Retail for [rm]...	Home	Home Décor	Home Décor Accents
3	4	24K GOLD plated rose	1	Women/Jewelry/Necklaces	NaN	44.0	0	Complete with certificate of authenticity	Women	Jewelry	Necklaces
4	5	Bundled items requested for Ruie	3	Women/Other/Other	NaN	59.0	0	Banana republic bottoms, Candies skirt with ma...	Women	Other	Other

After splitting, we analyzed that most of the listings are related to women and electronics. Within those categories, the fashion and computers/tablets subcategories are the most sold.

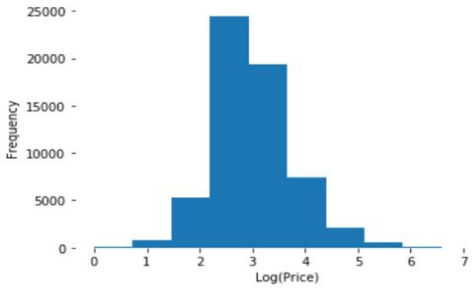


The next step was to analyze the response variable “Price.” We plotted a histogram to understand Price distribution and outliers.



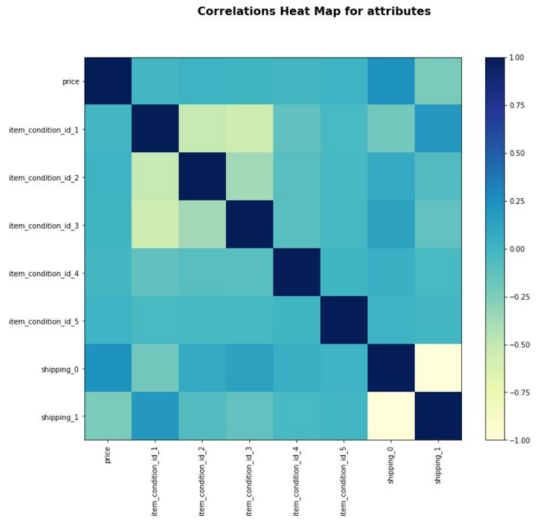
Based on the histogram, the price is highly right-skewed and contains outliers. Some products are listed at $1,400. Although these prices look like outliers, we decided to retain these values because they relate to “expensive products” and we wanted to develop a model that predicts prices for all categories of products. We then log transformed the price variable and plotted another histogram to observe the new distribution, which was symmetric like a parabola.

```
[18]: # price has a right skewed distribution
      # log transform price variable
      import numpy as np
      train_copy['price'] = np.log1p(train_copy['price'])
      plt.hist(train_copy.price)
      plt.xlabel("Log(Price)", fontsize = 10)
      plt.ylabel("Frequency ", fontsize = 10)
      plt.box(on=None)
```



The next question we addressed was to determine the most important input variables. As mentioned in the data section, the available input variables are product id, name, brand, shipping, item condition, main category, subcategory1, subcategory2. Product ID does not have any inherent meaning or bearing on the price, so it will not be included in the prediction. From our analysis, shipping and item condition had a strong correlation with price. We created dummy variables for item condition and shipping to observe their correlation with price. Based on the correlation matrix below, item conditions ranked 4 and 5 have a high correlation with price. Also, products for which shipping is paid by

buyers have a stronger correlation with the price (if a buyer pays the shipping cost, the price is usually higher).

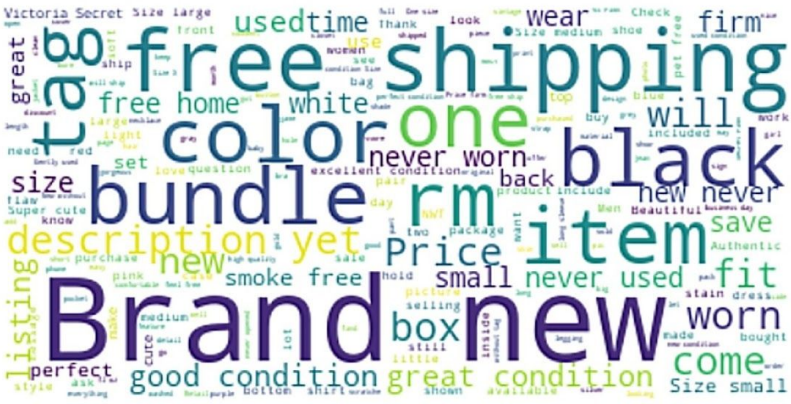


Other variables like product name, brand name, category, and description also have a strong bearing on the price. Although these variables may not help predict price individually, when used together, they are extremely important in determining the price of a used product.

The next step was to merge the test and train sets and perform data cleaning to fill missing values. We filled missing values in textual columns as “missing” and “0” if the price was missing. Then we split this dataset into test and train sets based on the length of the original train set.

Stage 1: Performing feature extraction on variables without processing text
We transformed the item description and product name columns using TFIDF vectorizer. The name and item description columns are custom text inputted by the sellers. Since they can input anything in these sections, the data for these columns were extremely wordy and structured. The best way to extract features was by using TFIDF – Term frequency inverse document frequency. TFIDF will generate scores for each word in these columns based on inverse frequency; commonly used words will get lower scores, and less frequently used words will receive higher scores. We plotted a word cloud to get an understanding of words in these columns.

Most Used Words in the Description



From the word cloud above, words like “brand, new, shipping, item, free” are the most commonly used. The TFIDF vectorizer will return low scores/weight for such words as it works on inverse document frequency. After applying TFIDF, the result is a sparse matrix of the format (A, B) C where:

- A: Document index
- B: Specific word-vector index
- C: TFIDF score for word B in document A

The same step was repeated for the “item_description” column. We applied “label encoder” on the “brand name” and count vectorizer for the “category name.” These two variables are grouping data into different subgroups and are not user-specific or extremely discreet. Hence, we applied a count vectorizer and label encoder to extract codes. Once again, the result is a sparse matrix of the form (A, B) C. Then, we created dummy variables for item_condition_id and shipping and converted them into a csr matrix. We then combined these 5 matrices to create a training matrix.

Model 1: We applied ridge regression on the training matrix above using the least squares method. We kept the ridge parameter to a minimum value as increasing alpha increased the RMSLE. With using default parameters of a lsqr ridge regression, the model was able to predict prices for products. Ridge regression was able to predict with an RMSLE of 0.5268.

Model 2: We then applied a Light GBM regressor. The default parameters are as follows.

```
LGBMRegressor(boosting_type='gbdt', class_weight=None, colsample_bytree=1.0,
              importance_type='split', learning_rate=0.1, max_depth=-1,
              min_child_samples=20, min_child_weight=0.001, min_split_gain=0.0,
              n_estimators=100, n_jobs=-1, num_leaves=31, objective=None,
              random_state=None, reg_alpha=0.0, reg_lambda=0.0, silent=True,
              subsample=1.0, subsample_for_bin=200000, subsample_freq=0)
```

This model was able to predict prices with RMSLE of 0.5451.

Model 3: This time, we tuned the parameters of the LGBM model. After experimentation, an LGBM with the following parameters gave us the lowest RMSLE of 0.5228. We trained the model using different values of boosting rounds and got best results with 1,800 rounds of boosting.

- parameters = 'learning_rate': 0.15, 'application': 'regression', 'max_depth': 13, 'num_leaves': 400, 'verbosity': -1, 'data_random_seed': 1, 'bagging_fraction': 0.8, 'feature_fraction': 0.6, 'nthread': 4, 'lambda_l1': 10, 'lambda_l2': 10

Model	RMSLE (Root mean squared log error)
Linear Regression	0.5237
Light gradient boosting method	0.5456
LGBM (Custom parameters)	0.5217

Stage 2: We applied the following text processing techniques to the name and item description variables.

1. Remove punctuations
2. Remove numbers
3. Lower case
4. Remove stop words
5. Apply stemming

These steps were performed on both columns using functions from python's nlkt library like punctuation, porter stemmer, etc. We then repeated the same steps as in stage 1

(TFIDF, Count vectorizer and label encoder) and ran the 3 models mentioned above. Performing text processing decreased the RMSLE for all models significantly. LGBM (parameter tuned) performed better than the others with the lowest RMSLE of 0.2765.

Model	RMSLE (Root mean squared log error)
Linear Regression	0.2789
Light gradient boosting method	0.3008
LGBM (Custom parameters)	0.2765

2.2 Conclusion and Key Takeaways

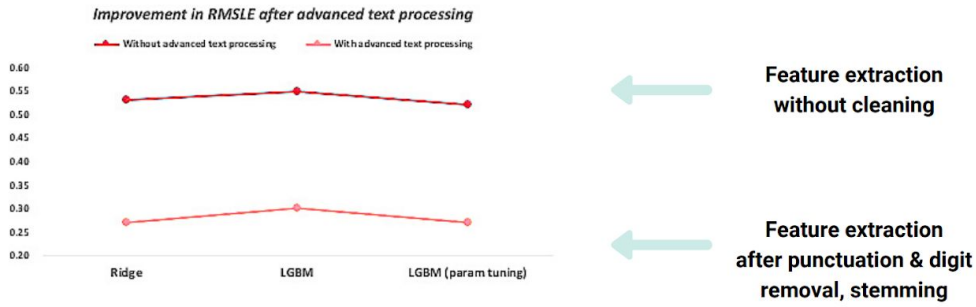
Price is the most important influencer of engagement on the Mercari marketplace platform. We tried to predict optimal prices to help balance both the buyer-side and seller-side to attract more buyers and more sellers. This can further help with the conversion rate and improve customer engagement.

In this project, we used the evaluation metric with Root_mean_squared_log_error (RMSLE). The lower the RMSLE, the smaller the error rate. Thus, the model which gives us the lowest RMSLE is the best model.

First, we used the Ridge Regression model to identify general patterns from new data to predict the price of a used product. We further used the LGBM with highly optimized histogram binning as a means of speeding up the tree-splitting process. Before text processing, we found that the RMSLE using Ridge Regression, LGBM, and LGBM after parameters were 0.5237, 0.5456, and 0.5217, respectively. However, high RMSLE was not good enough to make pricing predictions. Thus, we had to improve the models through advanced text processing techniques like punctuation, digit removal, stop word removal, and stemming. After performing text processing, we got a better RMSLE of 0.2789, 0.3008, and 0.276 for models using Ridge Regression, LGBM, and LGBM with tuned parameters respectively. Overall RMSLE has improved by more than 45% and much more accurate than the initial models we have built.

As a result, LGBM with parameter tuning showed the best results, with an RMSLE of 0.2765. This represents the lowest error rate with the text processing model.

How We Improved the Model



Step 1:

Feature extraction on unprocessed text -> Ridge Regression and Light Gradient Boosting Method -> LGBM model was run for a second time, with parameter tuning

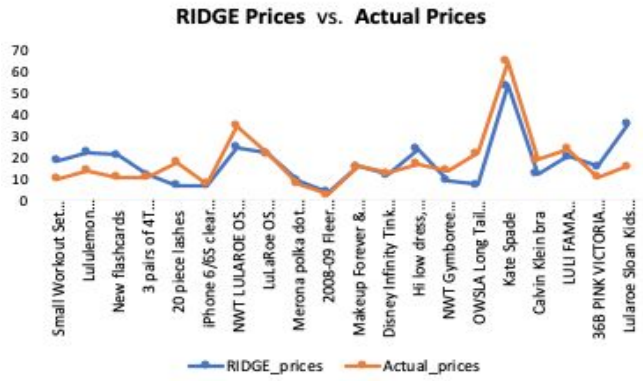
Step 2:

Feature extraction after preprocessing data -> Running the models -> RMSLE improved by more than 45% -> LGBM with parameter tuning showed best results with an RMSLE of 0.27 after text processing

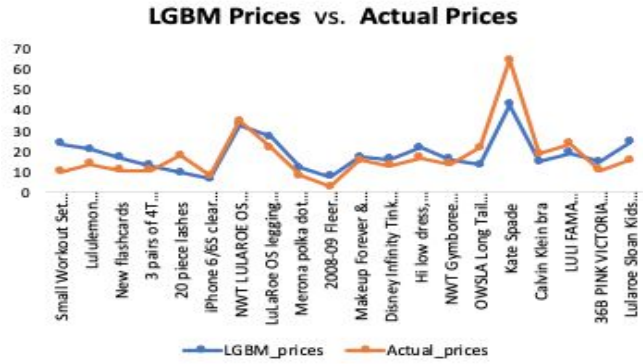
Among all variables, the top 3 variables that influenced the price the most are Item Description, Product Name, and Brand name. These are also the most reliable predictors of pricing suggestions. Adding detailed item descriptions using more specific words and discreet words will help predict the price better.

The following graphs provide an overview of actual and predicted prices for the first 20 product listings. From the first model, RIDGE prices are predicted fairly well in most products, but there are still spaces for improvements. From the second model, LGBM prices were not as good as the RIDGE price. From the third model, LGBM with custom parameters has slightly improved the previous model we used, showing that most product prices have fit the actual models with the least RMSLE.

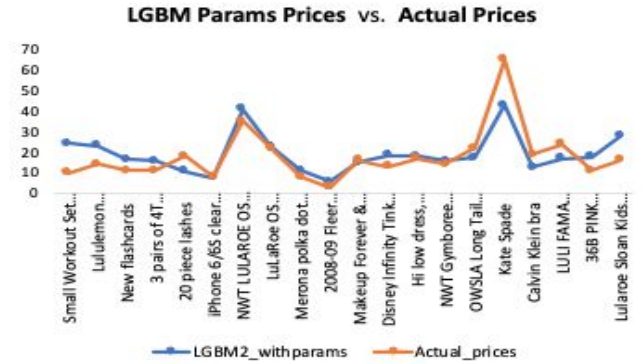
First Model: RIDGE Prices compare to Actual Prices.



Second Model: LGBM Prices compare to Actual Prices.



Third Model: LGBM Params Prices compare to Actual Prices.



Bibliography

Croll, Alistair, and Benjamin Yoskovitz. *Lean Analytics*. Universidad Internacional De La Rioja, S.A. (UNIR), 2014.

Eddy, Joe. “Google Analytics Customer Revenue Prediction.” *Kaggle*, www.kaggle.com/c/ga-customer-revenue-prediction/discussion/66048.

Marsupial, Dikran (https://stats.stackexchange.com/users/887/dikran-marsupial). “Why does ridge regression classifier work quite well for text classification?” *Stack Exchange*, stats.stackexchange.com/q/19482.

Qshick. “Ridge Regression for Better Usage.” *Medium*, Towards Data Science, 3 Jan. 2019, towardsdatascience.com/ridge-regression-for-better-usage-2f19b3a202db.