Mercari price suggestion challenge

Ankit Bhadu
2018CSB1073
Adarsh Kumar
2018CSB1066
Indian Institute of Technology Ropar
.
Guided By :
Dr Subramanian

Abstract

Mercari is Japan's biggest community powered shopping app where people can sell and buy a variety of brand new and used products of different brands, from sweaters to smartphones. Mercari would like to suggest the correct prices to the sellers but this is tough because their sellers are enabled to put just about anything, or any bundle of things on Mercari's marketplace. Our objective is to build a model for Mercari that automatically suggests the right product prices to the sellers.

1 Introduction

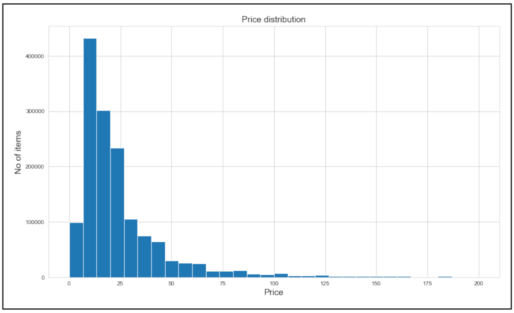
To start with we try out various regression techniques as this problem lies under the category of supervised regression machine learning.

1.1 Exploratory Data Analysis

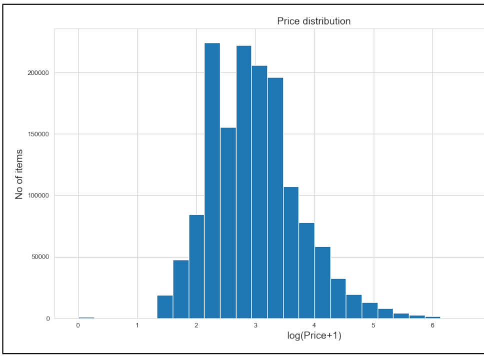
1. **Contents :-**
   Train id — the id of the product
   Name — the title of the product
   Item condition id — the condition of the product provided by the sellers
   Category name — category of the product
   Brand name — the product's brand name
   Shipping — 1 if shipping fee is paid by seller and 0 if shipping fee is paid by buyer
   Item description — the full description of the product
   Price(Target Variable) — the price that the product was sold for

2. **Price Distribution:-**
   We plot a histogram to observe how the price of items is distributed and come to know that most items are priced between [15,25]. Since the data is highly right skewed we take logarithm. Since price can have value 0 we take log(price+1) to avoid log of 0. After this transformation the plot seems to be well distributed as shown in FIG-1(b). We can observe that now most of the price are between 2-4.

3. **How bearer of shipping charges affect price**
   We plot another histogram(Figure-2) to see how it changes price of an item when



(a)



(b)

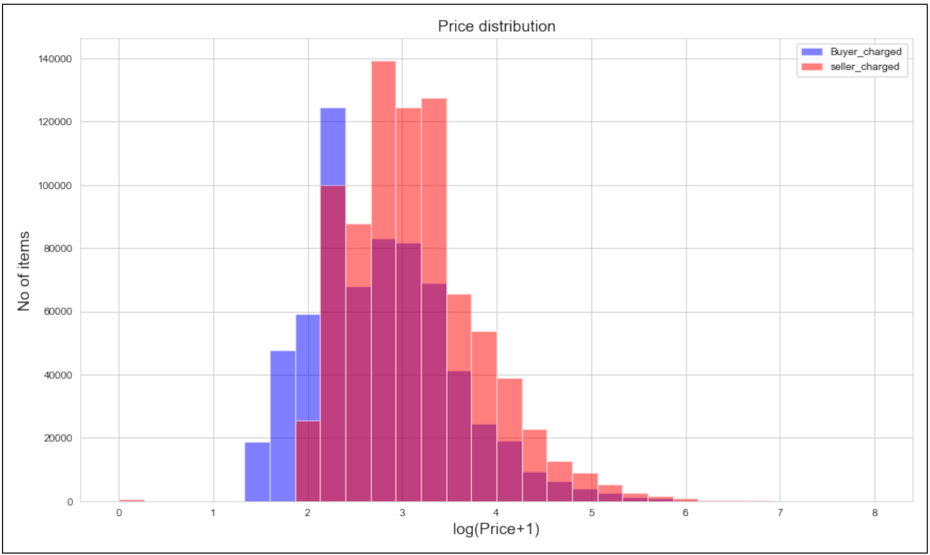
Figure 1: price distribution (a) original (b) logarithmic transform

shipping charges are paid of by the buyer vs when they are paid by the seller.We can observe shipping charges are more often paid often by the seller as opposed for the higher priced products where its paid by the buyers. It is probably due to profit reasons for the lower priced products however as we can see there is lot of overlap too between the two categories.

4. **Category of product**
   Category column can be divided into 3 based on '/' say main category and then sub categories. We found that data contains 1288 categories and in main category section women had maximum frequency and in sub category Athletic apparel was the top one.We plotted several plots for distribution according to categories and observing relations between them but it was of very less use to the main problem,

5. **Description length**
   A scatter plot[figure 4] of Distribution length vs price showed that higher priced items required less description as compared to lower priced ones. It shows that branded items which are usually higher priced do not need much description,i.e. they sell themselves.

6. **Correlation**
   The correlation matrix[figure 5] makes the picture altogether clear as the relations drawn untill now are supported by it. We can see that description length has a fair correlation with the price of an item and hence we have selected it as a feature in addition to item_description.

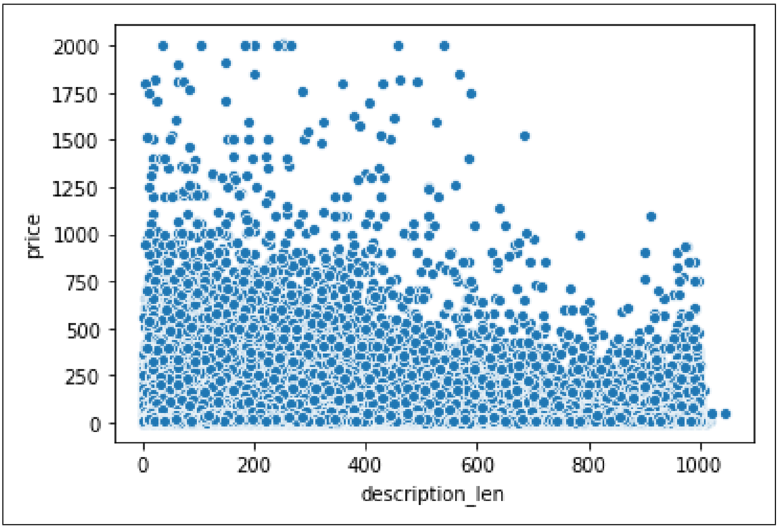


(a)

Figure 2: How bearer of shipping charges affect price

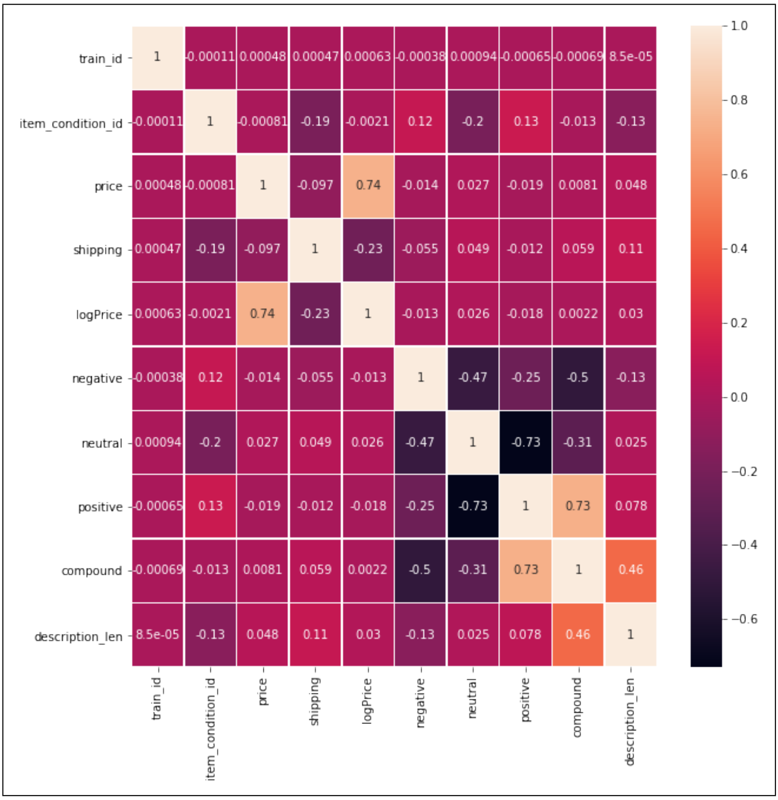
(a)

Figure 3: visualization of frequently used words in description through wordcloud



(a)

Figure 4: Price vs Description length



(a)

Figure 5: Correlation between different features

1.2 Pre-processing

Firstly we change all short forms of words like aren't, we've, we'll to their complete forms to avoid separate vectorization of say aren't and "are not". Then we remove all stop words which won't convey any meaning. All stopwords have been taken from "https://gist.github.com/sebleier/554280" which contains NLTK's all stop words.
Next we perform one hot encoding to convert our categorical values to numerical values. Now to represent the column 'item_description' numerically we use the technique of "bag of words".In this model, a text is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. We ignore punctuations and frequently occuring meaningless words to deal with the large set of vocabulary.

2 Literature Survey

2.1 Existing Models

Since this is a Natural Language Processing problem, existing models have used Recurrent Neural Networks and Gated Recurrent Units, such models have provided RMSLE of 0.43. Some ensemble models which combined Ridge Regression and RNNs also exist,having RMSLE of 0.44.
Some models have also used the Gradient Boost Algorithm (paper link:https: "//arxiv.org/pdf/1603.02754") which hasn't given good results because it tends to overfit on the training data when principal features aren't identified correctly.

2.2 Regression

Regression techniques like logistic regression, SGD have long been applied to NLP based classification problems. This article ("https://medium.com/natural-language-processing-machine-learning/nlp-for-beginners-how-simple-machine-learning-model-compete-with-the-complex-neural-network-on-b9f7f93c79e6") shows how a simple machine learning model like Naive-Bayes logistic regression can result in a performance as good as an RNN (LSTM) models. Regression models have been successfully applied to large-scale and sparse machine learning problems often encountered in text classification and natural language processing.

2.3 Recurrent Neural Networks

Such kind of neural nets are used when there is a temporal component in the input data.For eg:speech recognition or language processing tasks like sentiment analysis.They were based on David Rumelhart's work in 1986.Hopfield networks - a special kind of RNN - were discovered by John Hopfield in 1982.There are various kinds of RNNs like LSTM or Long Short Term Memory networks,also improved large-vocabulary speech recognition and text-to-speech synthesis and was used in Google Android.In 2015, Google's speech recognition reportedly experienced a dramatic performance jump of 49% through the use of LSTM networks, which was used by Google voice search.

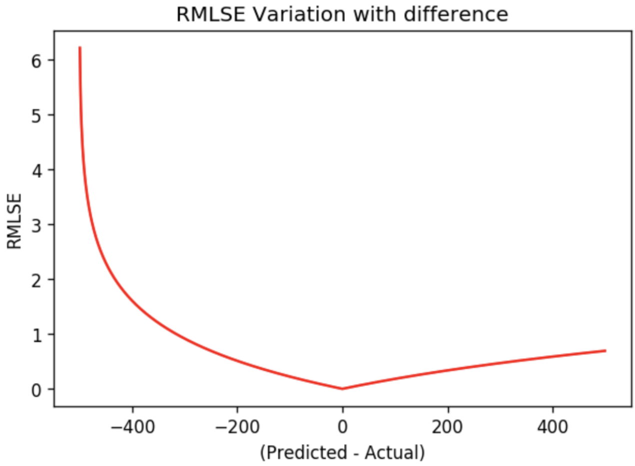
3 Performance Metric and Models Used

3.1 Performance Metric

Since we had to predict the price of a product and provide it to a seller, if the predicted price is higher than the actual price, it is of not much concern to the seller, but if the predicted price is lesser than the actual price, then the seller might suffer a loss. Hence, while training the model, we should incur a larger penalty if the predicted value is lesser than the actual value. Hence the performance metric that we chose is Root Mean Squared Log Error. We didn't choose Root Mean Squared Error because it incurs an equal penalty for over estimation and under estimation.
The formula for RMSLE is:

$$\text{RMSLE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\log(y_i+1)-\log(\hat{y}_i+1)\right)^2}$$

This plot shows the penalty incurred by RMSLE vs the difference in actual and predicted values.



Hence we have applied the algorithms after changing the actual and predicted prices p to $log(p+1)$.

3.2 Models

We have experimented with a lot of machine learning models of various types such as:

1. Linear Models
   - Linear Regression
   - Stochastic Gradient Descent Regression

- Support Vector Regression

2. Non-Linear Models

   - Decision Tree Regression
   - Light Gradient Boosting Machine Regression

3. Ensemble Models

   - XGBoost Regression

To choose the hyperparameters for these models,we assigned them random values and then used 5-fold cross validation to find out which value gives the best results.

4 Results

The table shows the value of RMSLE for each of the models used:

Model	RMSLE Value
Linear Regression	0.47
SGD Regression	0.48
SVM Regression	0.52
Tree Regression	0.5
LGBM Regression	0.48
XGB Regression	0.59

We can see that the least RMSLE value is for Linear Regression,hence indicating that the dataset is linearly separable to some extent.
Also the Gradient Boost regression method has performed poorly because it isn't good at identifying the principal features,hence it tends to overfit the data.