Mercari Price Suggestion Challenge

Gautam Dayal

March 4th, 2018

1. Introduction

1.1. Background

Today, market faces new challenges and needs to learn very fast to process all available data, gain useful insights, and evaluate outcomes. The easiest way to achieve this is by having a dynamic pricing strategy that uses machine learning techniques.

If you follow market news, you're aware of different tips and tricks retailers around the globe use to keep customer engagement high. They create cashier-free stores, self-driving shops, and use new engagement techniques in old-fashioned offline stores. But all these are lame attempts as long as the price is still a prevailing reason to buy stuff for 60% of shoppers.

1.2. Goals and Outline

Modern retailers need an intelligent dynamic pricing strategy. Although new ways of collecting and processing data greatly help them with that, most specialists still use manual crawling techniques, Excel Spreadsheets, or simple price tracking solutions like Price2Spy/Prisync.

Price adjustments made for the whole inventory in just a second, in response to real-time demand, is much more effective than those set manually with all the human mistakes. It is where machine learning steps into the room, giving retailers an option to optimize not only prices, but also business strategy, costs, and managers' efficacy.

Keep reading to find out how exactly a retailer can employ strategic pricing and outperform his competitors.

2. Methods

2.1. Process

We run all of our images for both training and testing through a sequence of steps in order to improve the effectiveness of the system as a whole.
We begin by reading in each observation and pre-processing it as outlined below in order to reduce the variation between different examples. This allows the system to create a much clearer picture of the distinctions between different labels. The next step is to detect features in the observations and place them in a feature vector. Finally, we train and test several different machine learning models on our generated features.

We use a "pipeline analysis" approach in our project. Since our system consists of a series of separate steps, the overall performance is affected by several factors. At each step of the process, we compare different methods used for pre-processing, feature detection, and learning. In doing so, we can achieve a more complete understanding of how each step affects the system.

2.2. Software

Our main tool for analysis is R using existing libraries, for feature detection and learning model we will use Python. This software package includes utilities for every computer and Machine learning functionality that we require for this project.

2.3. Data Set

All training and test data contains observations and features in the form of plain text.

Features:

The files consist of a list of product listings. These files are tab-delimited.

- train_id or test_id - the id of the listing
- name - the title of the listing. Note that we have cleaned the data to remove text that look like prices (e.g. $20) to avoid leakage. These removed prices are represented as [rm]
- item_condition_id - the condition of the items provided by the seller
- category_name - category of the listing
- brand_name – name of the brand the product belongs to
- price - the price that the item was sold for. This is the target variable that you will predict. The unit is USD. This column doesn't exist in test.tsv since that is what you will predict.
- shipping - 1 if shipping fee is paid by seller and 0 by buyer
- item_description - the full description of the item. Note that we have cleaned the data to remove text that look like prices (e.g. $20) to avoid leakage. These removed prices are represented as [rm]

During our pre-processing & feature detection steps we will modify the structure and format of these features to fit our model for training and then prediction.

2.4. Feature Detection

During our pre-processing step we will first remove rows or observations that don’t have price value or the value is 0.

For transforming data and its structure we will use python's library *sklearn.preprocessing*. This library will transform text to encoded integer value provided the text should be hashable.

We will do encoding for two features category_name and brand_name as they are factors with predefined and limited set of values.

Then we will break the name and item_description features to tokens using Tokenizer available in library *keras.preprocessing.text,* these are the two features where user has provided free text.

Tokenization is breaking the sentence into words and punctuation, and it is the first step to processing text.

To keep same shape of the input features, we will use *pad_sequence* available under *keras.preprocessing.sequence* library for name and item_description features.

3. Models

We use three different learning algorithms in our project: Random Forest and Neural Networks. Here we outline each of these algorithms.

3.1. Random Forest

The random forest model bootstraps by selecting, with replacement, a random subset of the data for training each new decision tree (bagging). Furthermore, a random subset of the data is selected at each decision point.

3.2. Neural Networks

Deep learning refers to neural networks with multiple hidden layers that can learn increasingly abstract representations of the input data.

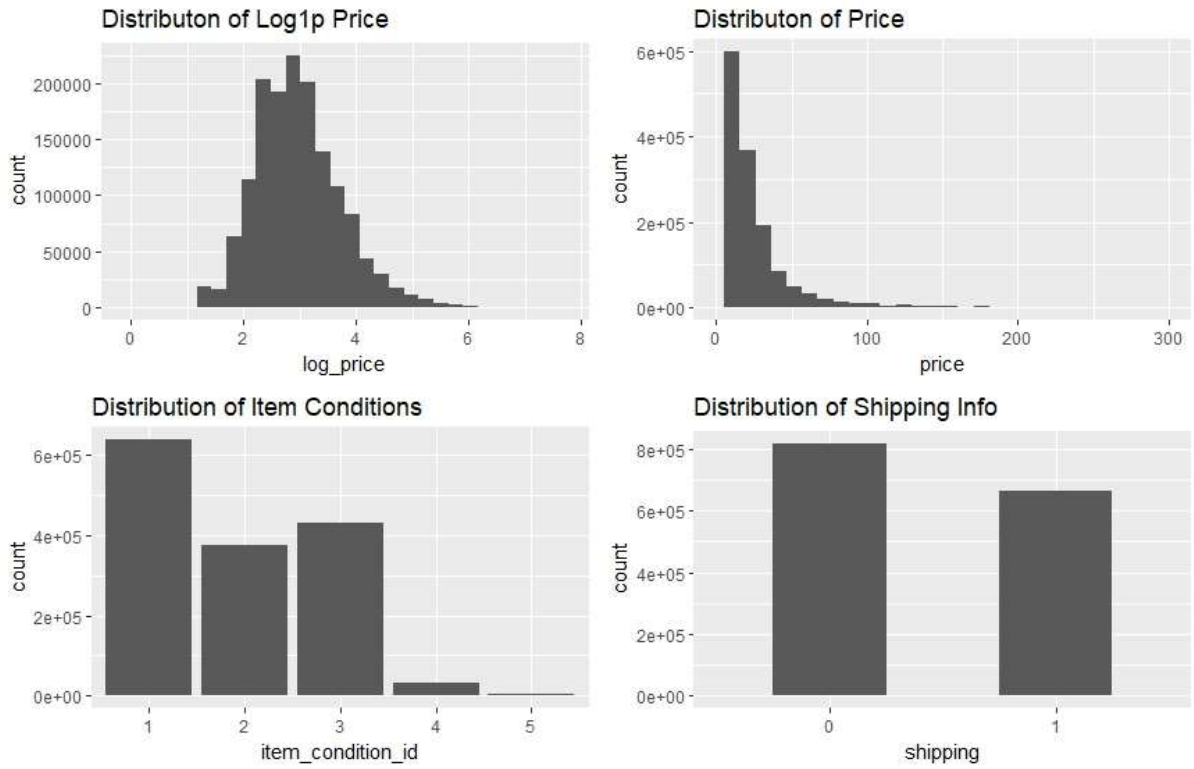
Convolutional Neural Networks (CNN's) are multi-layer neural networks (sometimes up to 17 or more layers) that assume the input data to be images.

4. Results and Analysis

4.1. Analysis

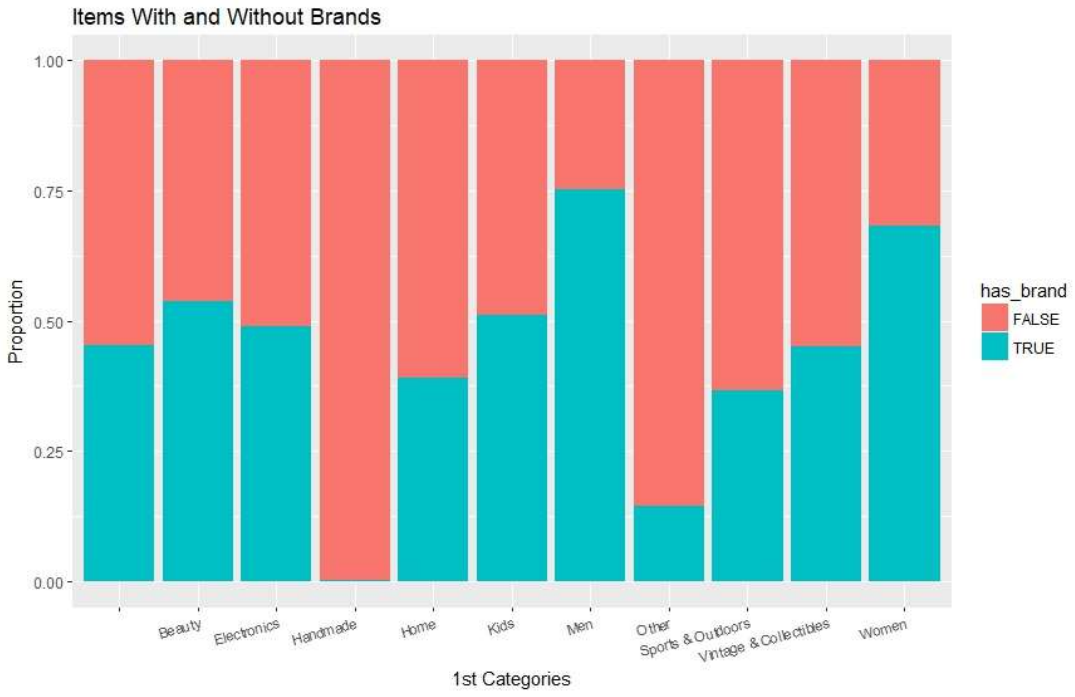
The training data has 1482535 observations with 7 features. The test data has 693359 rows that we need to predict. The evaluation metrics is RMSLE, which gives more penalty on under-estimating prices rather than over-estimating. Due to the nature of RMSLE we take log of the price in our analysis.

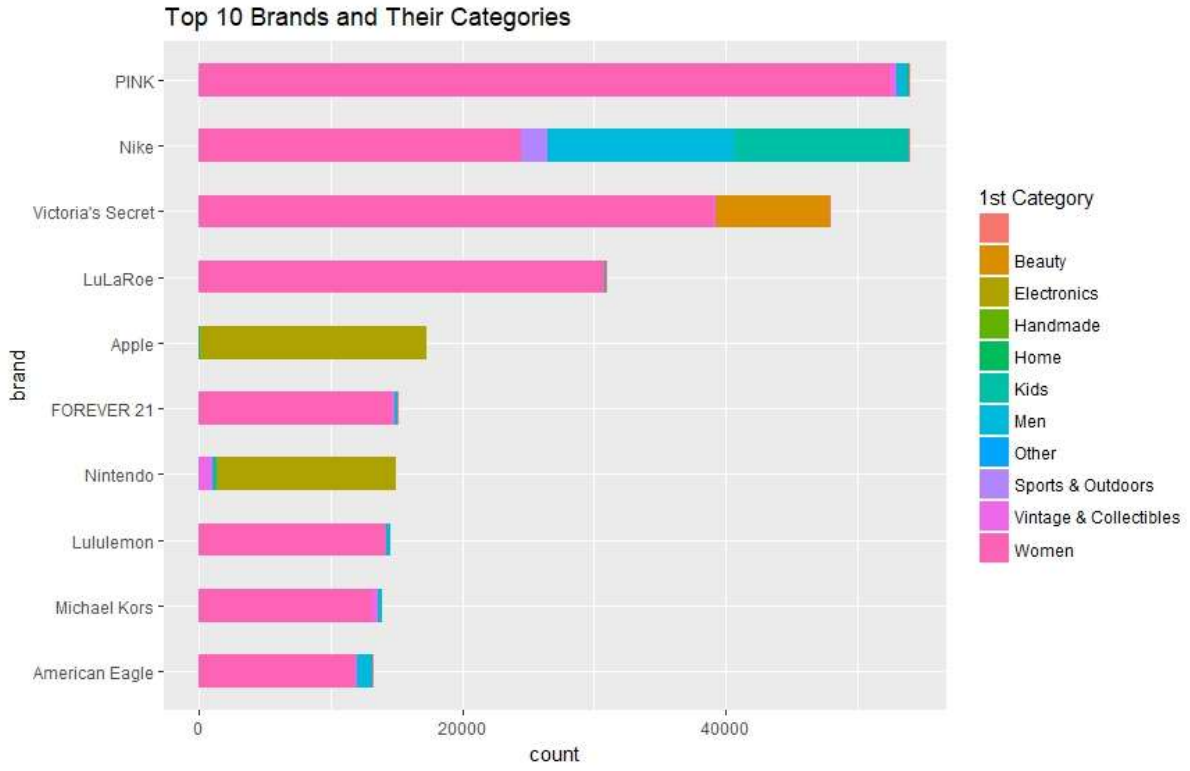
As the figures below show, the distribution of log price is more normally-distributed than the orginal scale of price.



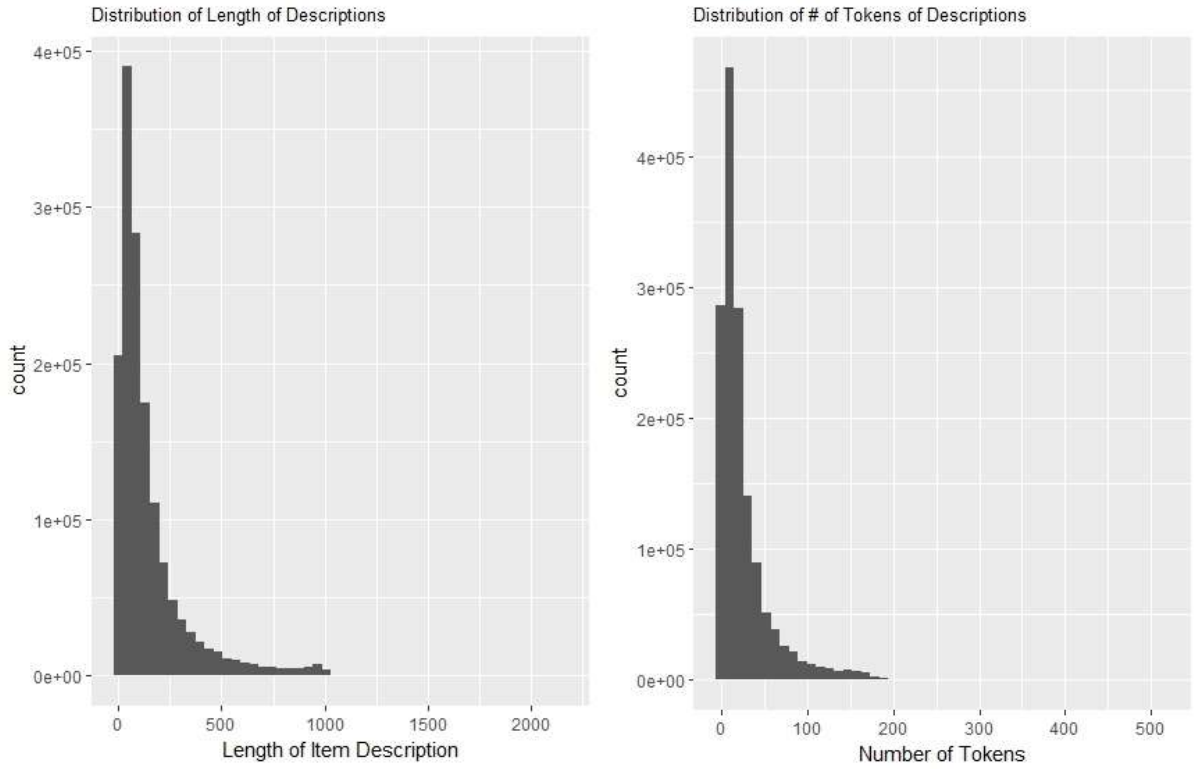
Nearly half of the items don't have brands. The proportions of items that have brands vary in different categories. For example, nearly all handmade items don't have brand names, of course.

For brands, they are not in a hierarchical order and there are too many to be fitted in one graph. So I plotted the count of top 10 most frequent brands for a rough look. Each brand contains items from 1 or more major categories. Not surprisingly, the top brands are dominated by women items except Apple and Nintendo.



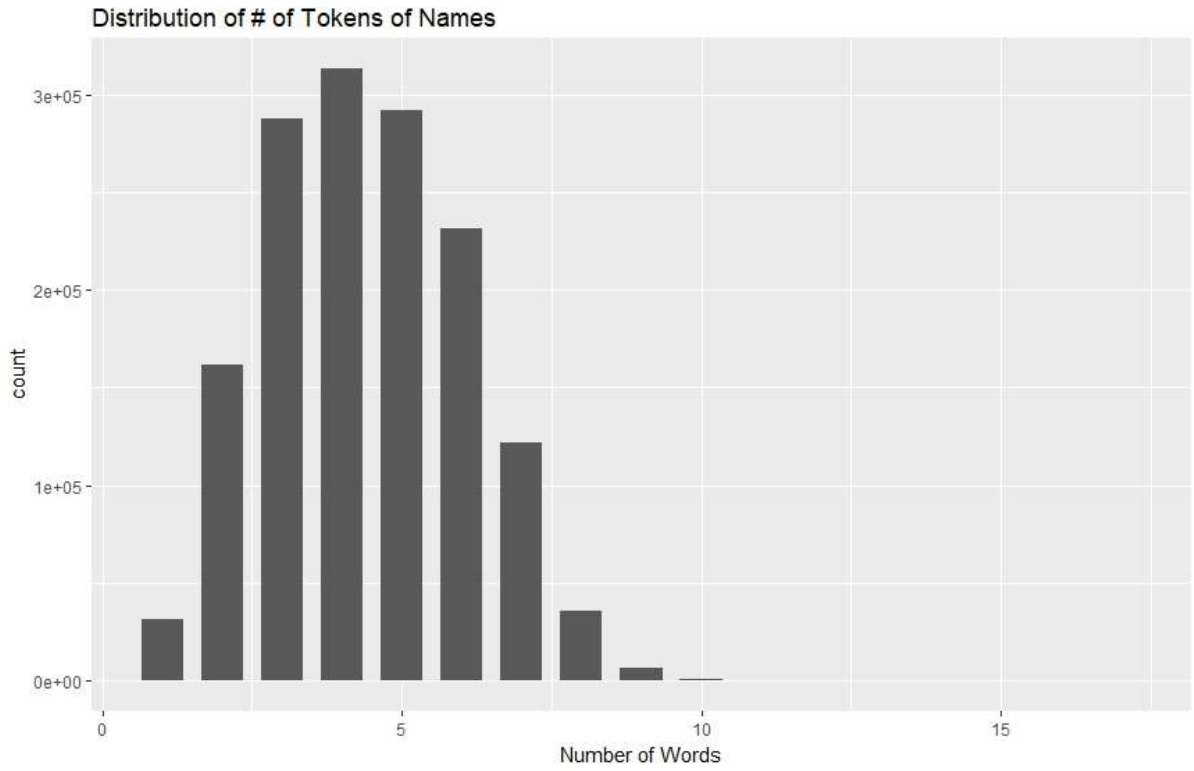


For the item description, I extract two most simple features of the text: the length of the text and the number of words of the text. It shows that both of them have a exponential distribution with a long tail.

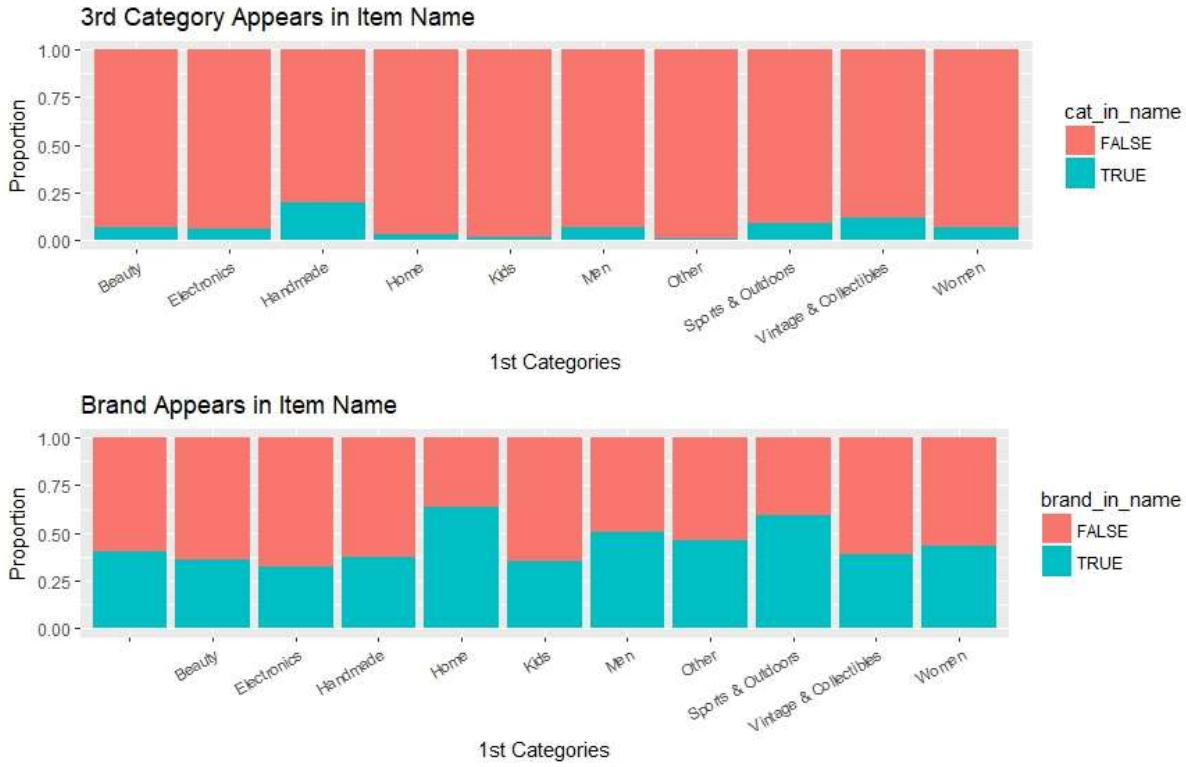


Name is a very important feature —— it directly tell us what the items are. But it's hard to be categorized or one-hot encoded because there are 1225273 unique

names for all the items. We should find some ways to extract the information from the item name.

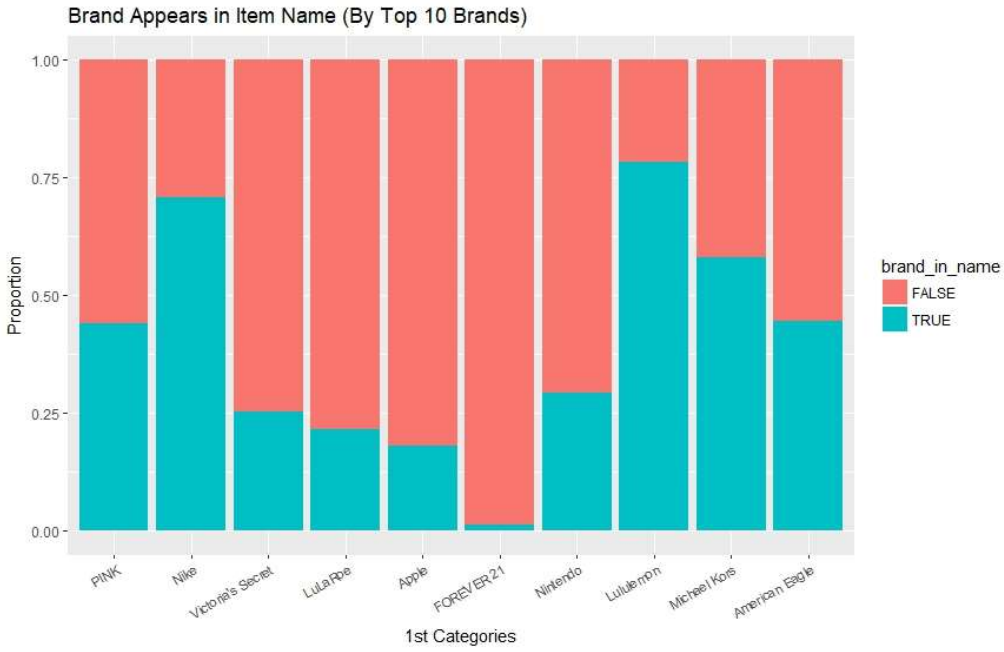


A very intuitive question would be: does the item name include its category and brand information? We know that the 3-level categories has been a very detailed classification of all the items. We can see how many items include their 3rd categories and brands in their names.



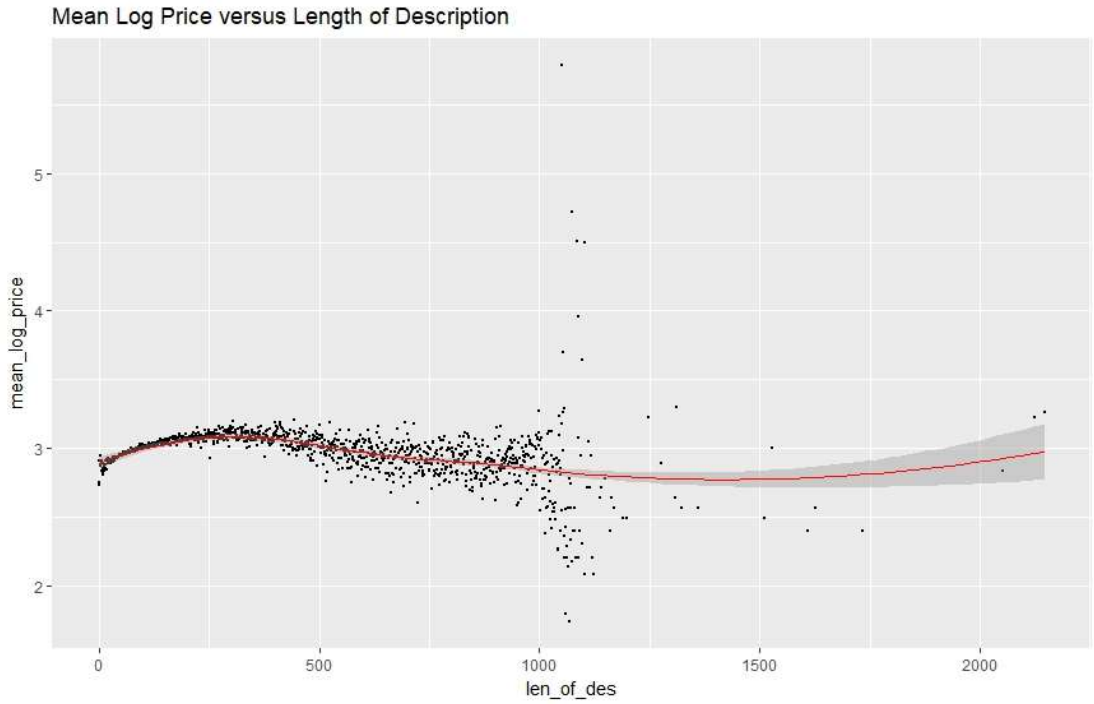
It appears that few item names include their categories, but many items include their brands.

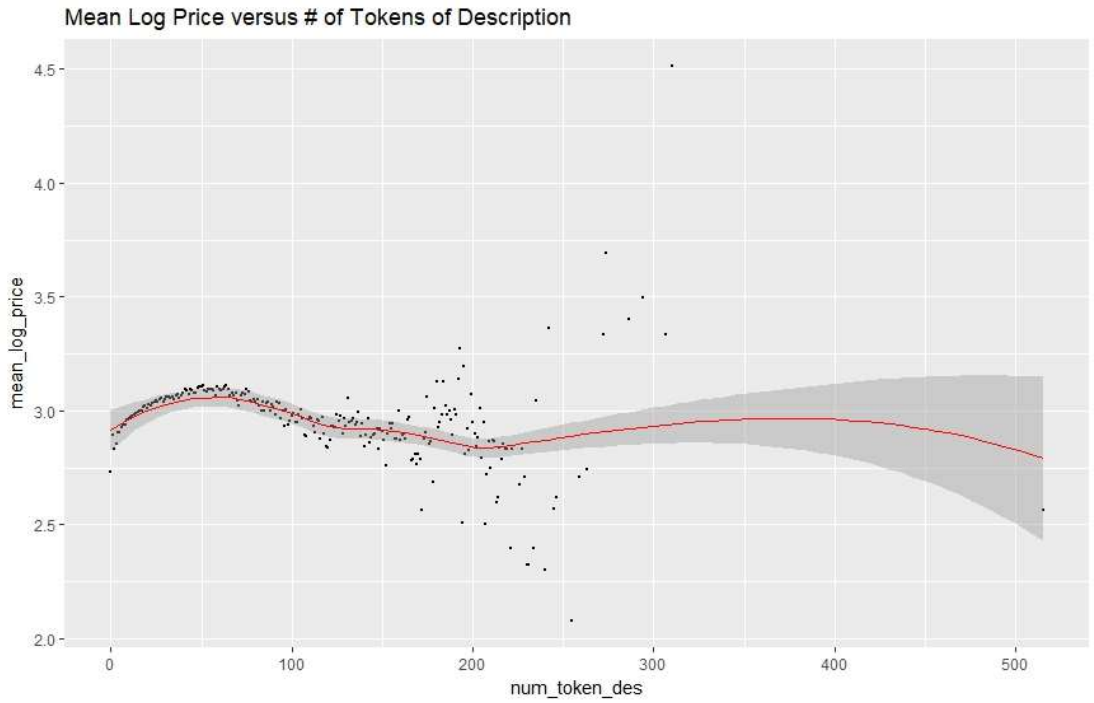
If we look at the brand_in_name information by brands, we can surprisingly see that it varies a lot by brands. For example, many of the Nike products contain 'Nike' in their names while Apple products don't show this trend —— perhaps they are named 'iphone' or 'macbook' and everyone knows they are from Apple.



The price don't differ too much between different categories. However, there is a clear trend that items with brands have higher price than items without brands, especially for the electronics.

There is a very interesting quadratic trend between # of words of description and the price. Note that when the # of words increases, the sample size for each point decreases exponentially, the variance of the mean log price then becomes large due to the central limit theorem.





4.2. Results

We found that the best performance came from Neural Network (Keras using Tensorflow as backend), Random forest was not able to provide target feature price values accurately. We were later able to achieve slightly better results using Threading from ThreadPool library. We called fit on the model 3 times for every prediction thread. The RMSLE (Root Mean Square Log Error) Metrix shows reduction in the value for the same data set, when the number of iterations increase, the reduction graph is exponential. Right balance is required on RMSLE value and the execution time.