Mercari Price Suggestion

Data Science Career Track Capstone Project

By: Joseph Sudibyo

The Problem

- Product pricing gets harder at scale, considering just how many products are sold online. Clothing has strong seasonal pricing trends and is heavily influenced by brand names, while electronics have fluctuating prices based on product specs.
- Mercari, Japan’s biggest community-powered shopping app, knows this problem deeply and would like to offer pricing suggestions to sellers.
- The Challenge: Sellers can put just about anything, or any bundle of things, on Mercari's marketplace.
- **The Goal**: To build a model that automatically suggests the right product prices.

The Data

https://www.kaggle.com/c/mercari-price-suggestion-challenge/data

- 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
- 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]

Missing Values

- category_ name: NaNs are replaced with ‘Other’
- brand_name: NaNs are replaced with ‘Not Specified’
- item_description: NaNs are replaced with ‘No description yet’ because a lot of the items have ‘No description yet’ in the item_description column

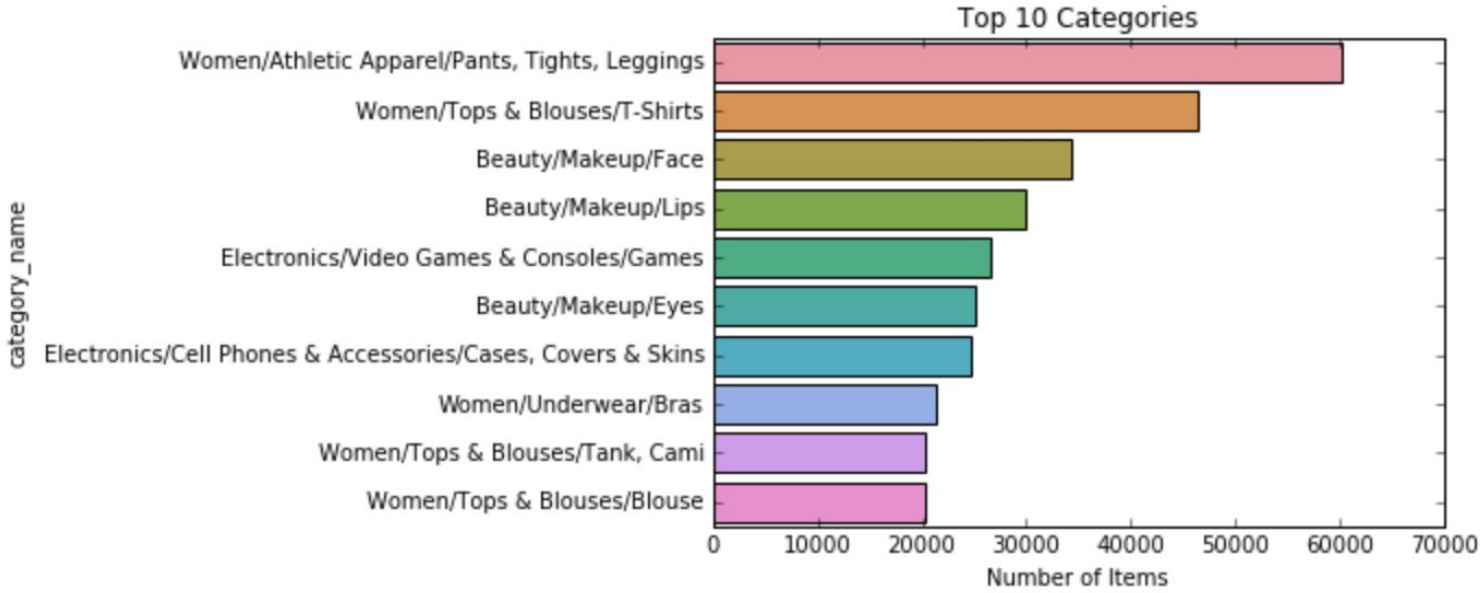
```
train_id              0
name                  0
item_condition_id      0
category_name      6327
brand_name       632682
price                 0
shipping              0
item_description      4
dtype: int64
```

Analysis of Categories

- Category name is 3 layers deep. For example: Men/Tops/T-shirt. Men will be the 1st category, Tops the 2nd category, Men/Tops the 1_2 category, and T-shirt the 3rd category. There are 10 unique values for category 1, 113 unique values for category 2, and 870 unique values for category 3.
- The same analysis was done to the overall category, category 1, category 1_2, and category 3. For the complete results of the analysis, refer to the final report and jupyter notebook.
- In general, Women category has the most number of items. The most expensive categories are: Electronics, Men, and Women (because of luxury items as can be seen from the brands analysis in the next section).

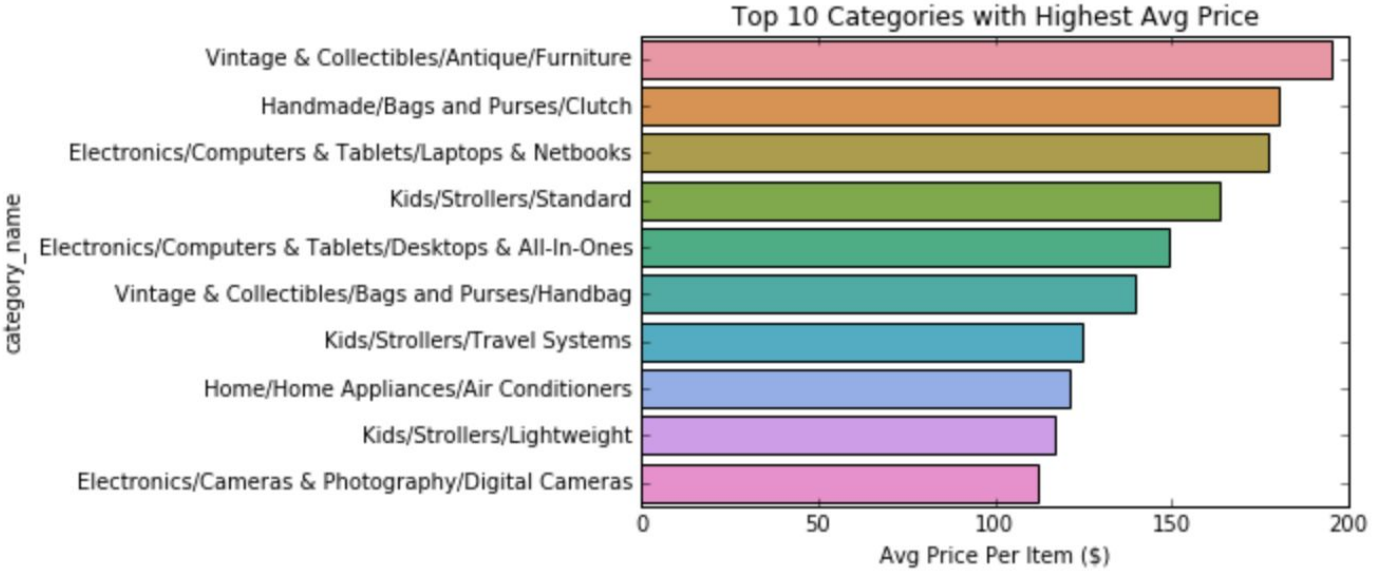
Analysis of Categories: Top 10 Categories

- Among the top 10 categories, 8 of them are in 'Women' category and the other 2 are 'Electronics'.
- The most popular category is Women/Athletic Apparel/Pants, Tights, Leggings



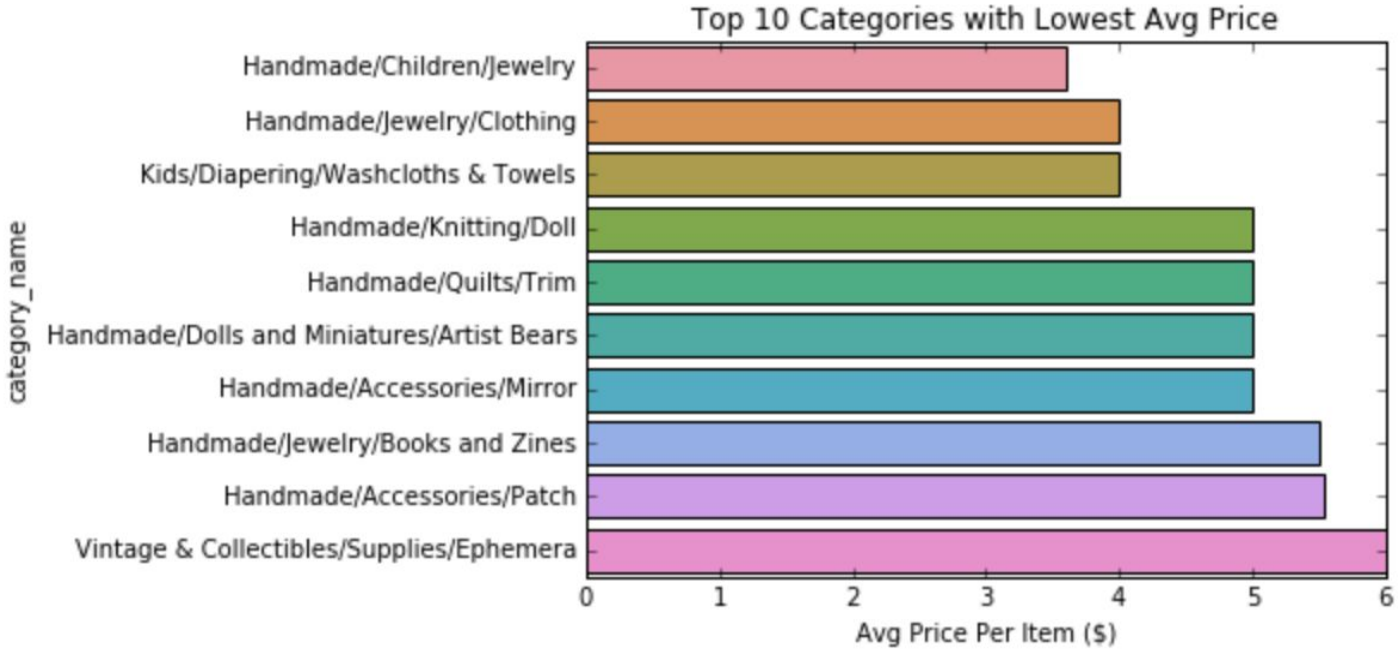
Analysis of Categories: Top 10 Most Expensive Categories

- Top 10 most expensive categories, as can be seen from the plot below, are as expected with furniture, electronics, handbag, and strollers dominating.
- The most expensive category is Vintage & Collectibles/Antique/Furniture which items in that category have an average price of almost $200.



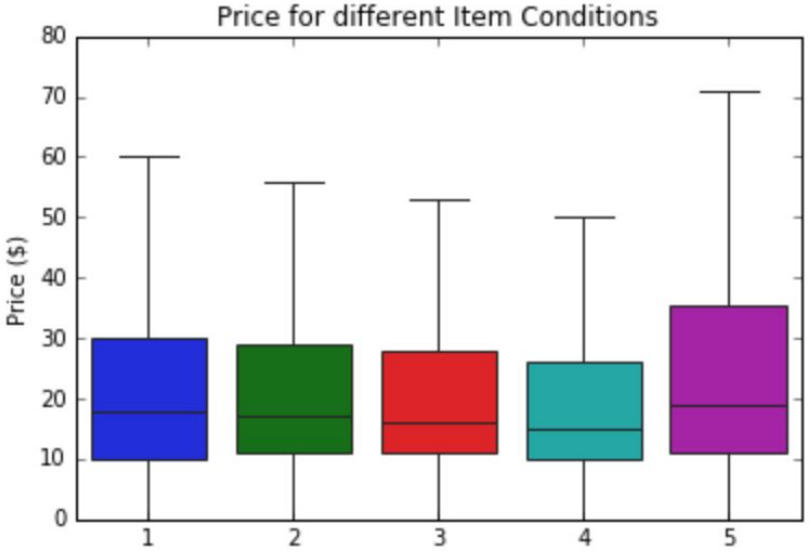
Analysis of Categories: Top 10 Least Expensive Categories

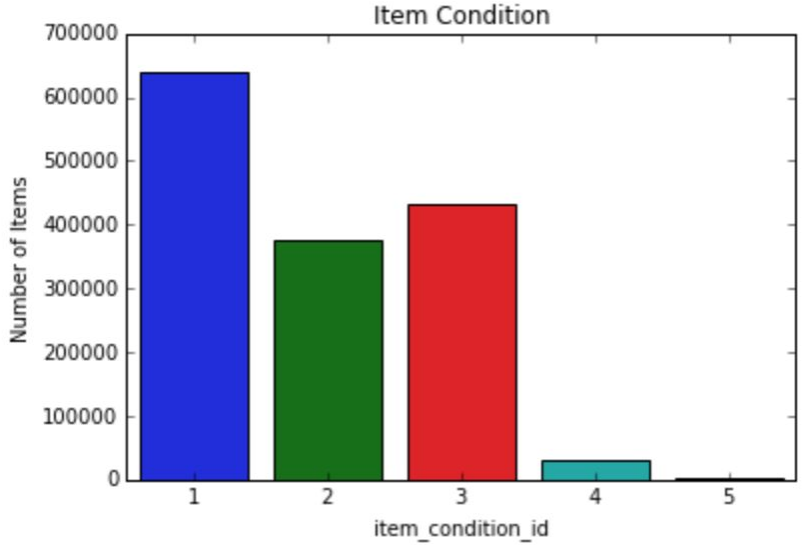
- The Top 10 least expensive categories are mostly handmade items (including jewelery which is somewhat counter intuitive).



Analysis of Item Conditions

- The item conditions are not really explained by Mercari so we don't really know whether 1 represents 'new' quality or 'poor' quality.
- Intuitively, the item condition (used, new, excellent, etc.) should play a role in determing the item's price as shown in the plot below.



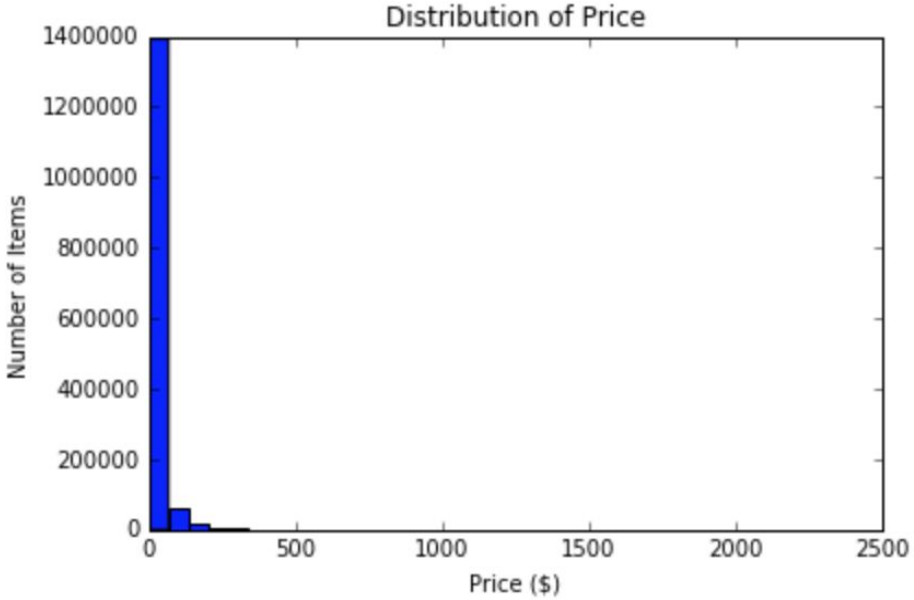


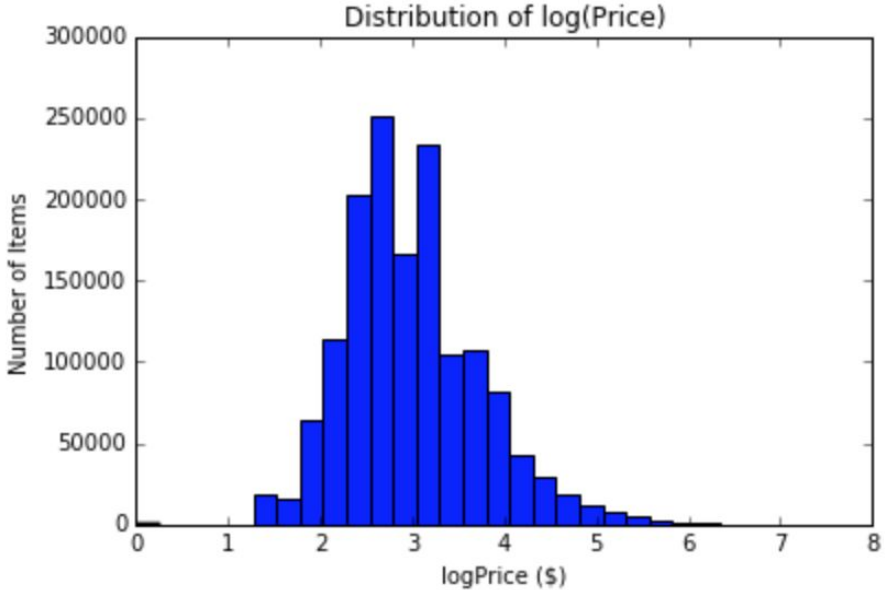
Analysis of Shipping

- The number of items that the shipping fee are paid by sellers is more than (but not by a lot) the number of items that the shipping fee are paid by buyers. It's as expected that the price of items are higher when sellers pay the shipping fee.

Analysis of Item Price

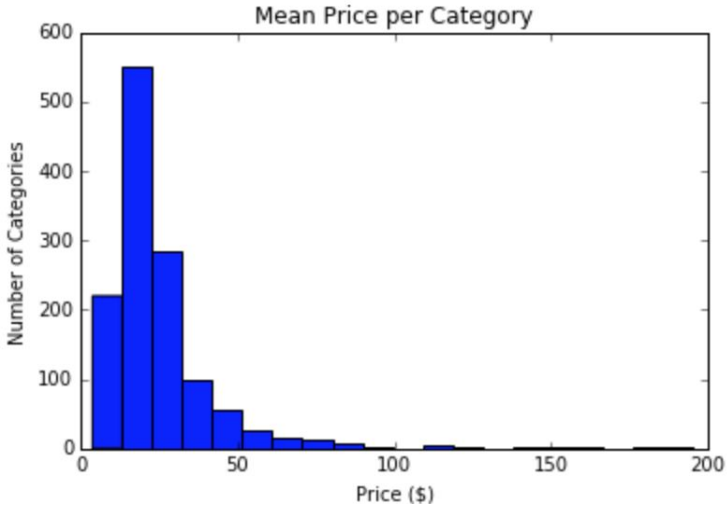
- Log transform the price in order to make the variable better fit the assumptions of underlying regression.





Analysis of Item Price Cont’d

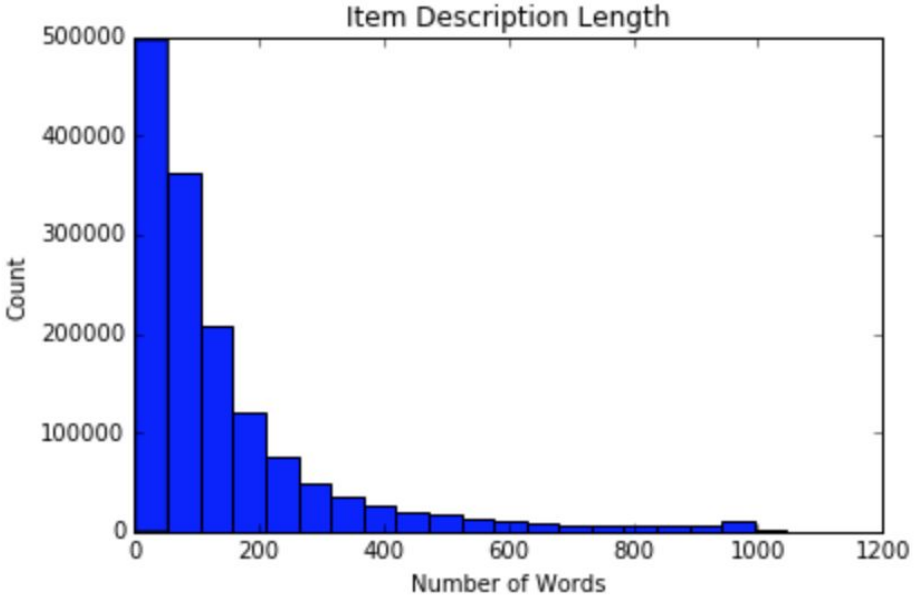
- Almost all categories have an average item price of <$100. With most of the categories having an average item price of <$50.



Analysis of Item Description: Most Common Words

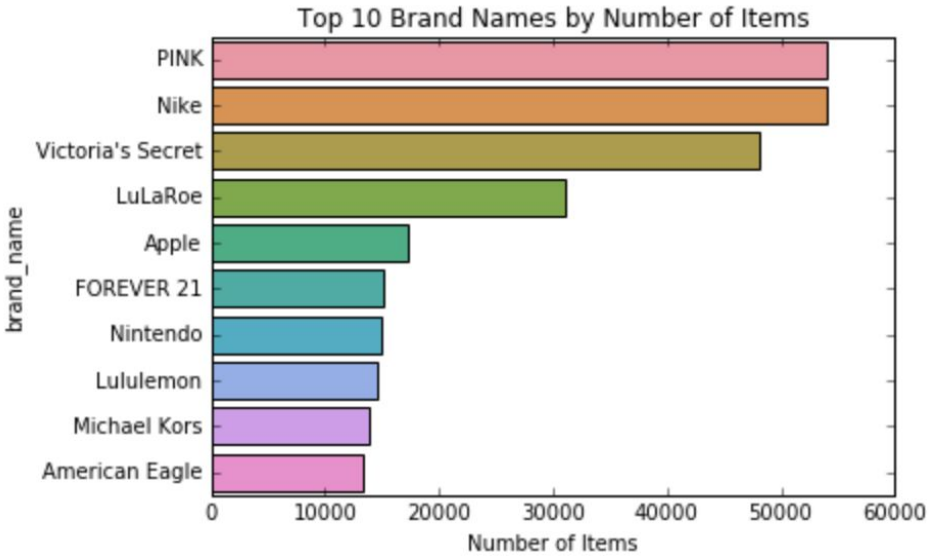
Analysis of Item Description: Length

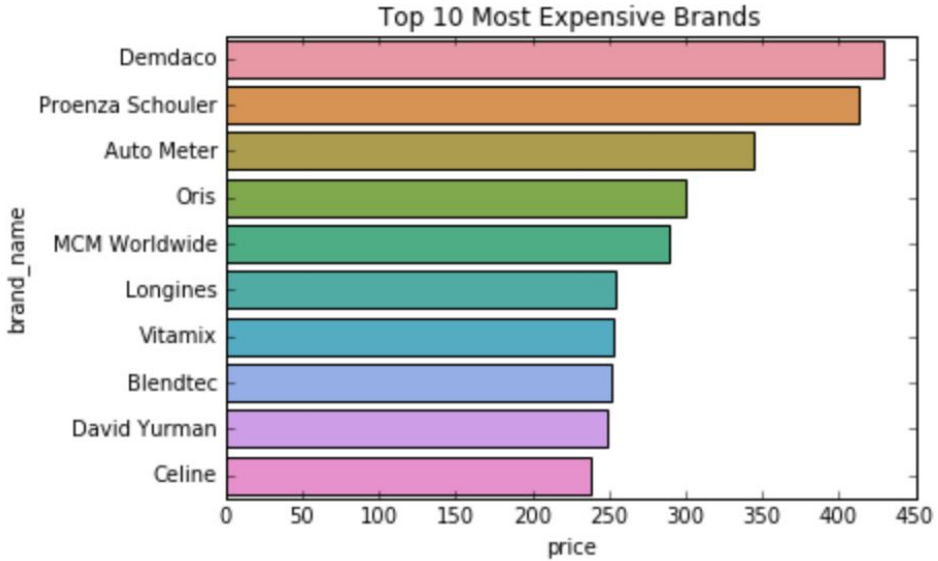
- From the scatter plots below, we can see that there is no clear relationship between item description length and price.



Analysis of Brands

- There are 4810 unique brands.
- The most expensive brands are luxury handbags, watches, and jewelry brands. PINK, Nike, and Victoria's Secret are the top 3 brands with most number of items.





Features Engineering

To produce the final data frame that we can feed to the machine learning model, we need to create features/extract features from different level of aggregations. The total number of features are 65.

- item_condition_id: The condition of the item
- shipping: Whether or not buyer pays the shipping fee
- brand_yesno: Whether or not the the seller specify the brand name
- category_yesno: Whether or not the seller specify the category
- item_desc_yesno: Whether or not the seller specify item description
- item_desc_len: Number of words in item description
- tfidf: The average tfidf for words in the item description

Features Engineering Cont’d

- *_price_category_name: Max/Min/Mean/Median price for item in a specific category name
- *_price_category_1: Max/Min/Mean/Median price for item in a specific category 1
- *_price_category_2: Max/Min/Mean/Median price for item in a specific category 2
- *_price_category_1_2: Max/Min/Mean/Median price for item in a specific category 1_2
- *_price_category_3: Max/Min/Mean/Median price for item in a specific category 3
- *_price_cond_category_name: Max/Min/Mean/Median price for item in a specific category name with a specific item condition
- *_price_cond_category_1: Max/Min/Mean/Median price for item in a specific category 1 with a specific item condition
- *_price_cond_category_2: Max/Min/Mean/Median price for item in a specific category 2 with a specific item condition
- *_price_cond_category_1_2: Max/Min/Mean/Median price for item in a specific category 1_2 with a specific item condition
- *_price_cond_category_3: Max/Min/Mean/Median price for item in a specific category 3 with a specific item condition
- *_price_brand_category: Max/Min/Mean/Median price for item in a specific category with a specific brand name
- *_price_brand: Max/Min/Mean/Median price for item with a specific brand name
- *_price_brand_cond: Max/Min/Mean/Median price for item in a specific category with a specific item condition
- *_price_cond: Max/Min/Mean/Median price for item with a specific item condition
- brand_name_price: score(ranking) of brands by price
- brand_name_count: score(ranking) of brands by number of items for a specific brand.

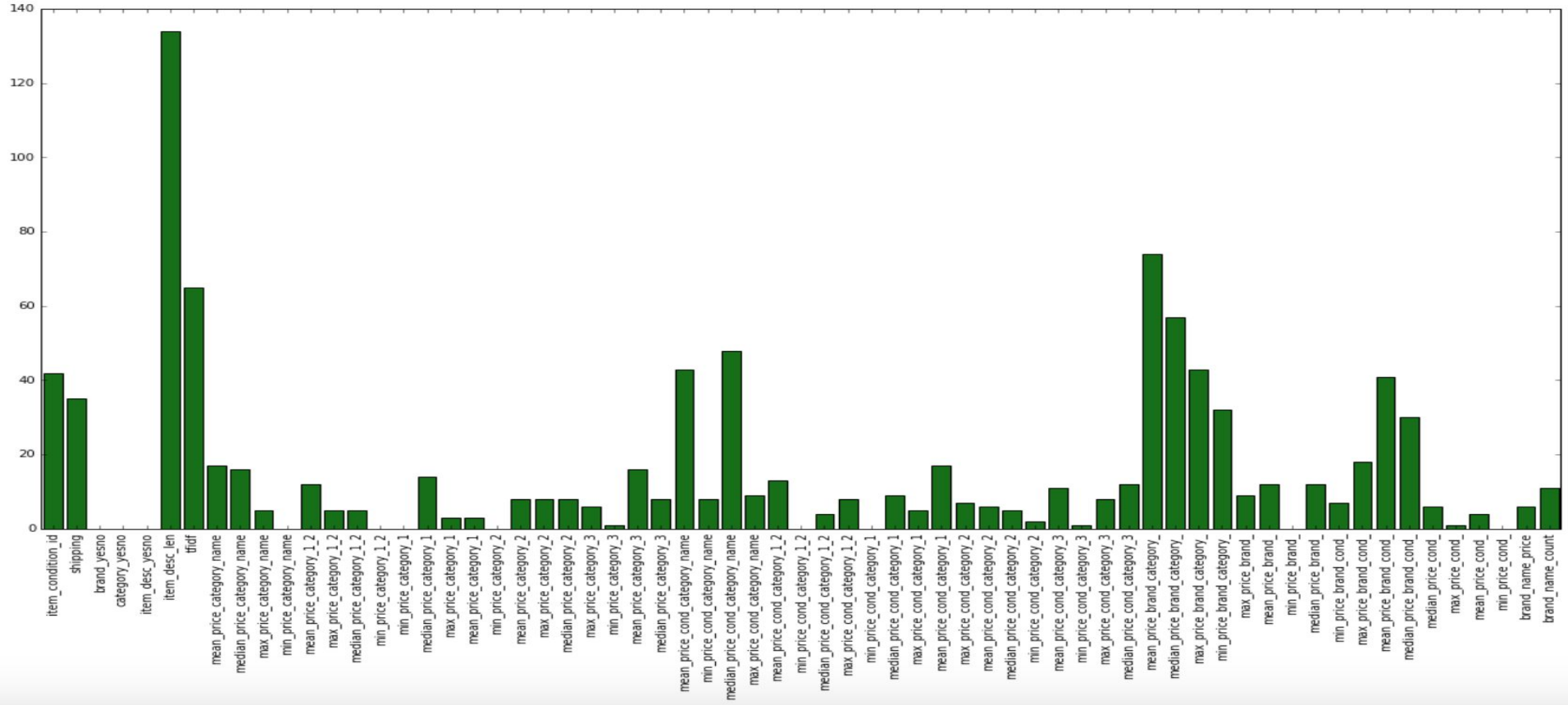
Train-Test Split

The original training data is split into new training data (80%) and new testing data (20%) so that we can fit the model using the new training data measure the performance of the machine learning model on the new testing data.

Light GBM Model Implementation

- The target variable that we want to predict is log(Price).
- The parameters tuned (Grid Search, 5-fold CV on training data) are: learning rate, number of leaves, and max depth.
- The performance of the model (on testing data) is:
  - mse on log(Price): 0.283866610945
  - rmse on log(Price): 0.532791339029
  - avg price error (average price difference): 11.604541683899463
- The best parameters combination that produces the least error (RMSE) is:
  - learning rate = 0.5
  - number of leaves = 100
  - max depth = 8

Light GBM Model Feature Importance



Light GBM Feature Reduction

After plotting the features importances, we can see that a few features are not important and the important features are only 25 (out of the original 65 features):

- Max_price_brand_cond_, Median_price_brand_, Mean_price_brand_category_, Median_price_cond_category_name, Mean_price_category_name, Mean_price_brand_cond_, Item_condition_id, Shipping, Mean_price_brand_, Brand_name_count, Mean_price_category_1_2, Median_price_brand_category_, Tfidf, Mean_price_cond_category_3, Max_price_brand_category_, Item_desc_len, Median_price_category_1, Mean_price_category_3, Median_price_cond_category_3, Median_price_brand_cond_, Mean_price_cond_category_1_2, Median_price_category_name, Min_price_brand_category_, Mean_price_cond_category_name, Mean_price_cond_category_1

Light GBM Final Result

Then, retrain the model using only these 25 important features. The performance of the model on the test data is very similar to the performance of the model when using 65 features.

- mse on log(Price): 0.283881667401
- rmse on log(Price): 0.532805468629
- avg price error (average price difference): 11.603250468427452

Ridge Regression Model Implementation

- Ridge Linear Least Squares Regression with CV is also performed as a comparison for the Light Gradient Boosting.
- The regularization coefficient alpha is also tuned to minimize error. The best alpha turns out to be 0.001.

The performance of the model (worse than the performance of Light Gradient Boosting):

- Best alpha: 0.001
- Mse on log(Price): 0.316908530359
- Rmse on log(Price): 0.562946294383
- avg price error (average price difference): 13.354043044080294

Conclusion

Light Gradient Boosting performs better than Ridge Regression to predict price in this problem.