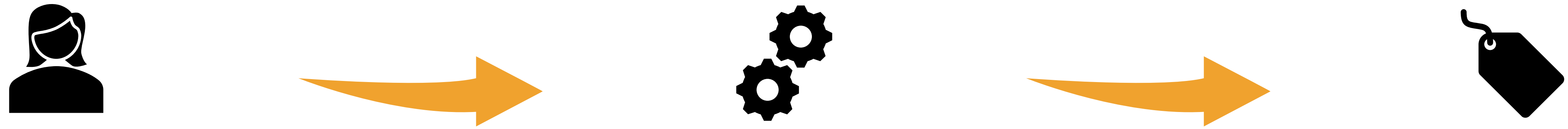
PRICE RECOMMENDATION USING NLP

OUTLINE

- Background
- Value
- Approach
- Exploratory Data Analysis (EDA)
- Data Pre-Processing
- Model Development
- Model Evaluation
- Results
- Future Scope

PROBLEM STATEMENT

To build an algorithm that automatically suggests the right product prices based on the user provided textual product descriptions, and other details like product category, brand name, and item condition.



Seller provides product listing on Mercari app

Algorithm recommends listing selling price

Better pricing suggestion delivers better profits for sellers

BACKGROUND

Dataset

The dataset is part of the Mercari Price Suggestion Challenge on Kaggle(https://www.kaggle.com/c/mercari-price-suggestion-challenge). Goal is to predict the sale price of a listing based on user provided information for the listing

Company

- Mercari, Inc. is a Japanese e-commerce company founded in February 2013 and currently operating in Japan and the United States.
- Their main product, the Mercari marketplace app has become Japan's largest community-powered marketplace with over JPY 10 billion in transactions carried out on the platform each month.

VALUE

Why is Mercari interested in this competition?

- Help sellers judge the prices of their products
- List the “best” possible selling price to increase profits

Why are we interested in this project?

- Real-world business problem with a large dataset
- Challenges us to clean unstructured data and perform EDA
- Opportunity to learn text pre-processing, text mining and encoding techniques.
- Apply various modeling techniques learnt in the data mining course and figure out the most optimal one for text mining

APPROACH

Exploratory Data Analysis

Analyze the dataset to develop initial hypotheses

Understand which variables can help predict product price

Text Pre-Processing

Transform text data into a form suitable for modeling

Examples: Perform lemmatization, stemming, vectorization, etc.

Model Development

Develop various models to predict the price

Different models that can be considered are CNN, RNN, xgboost, etc.

Model Evaluation

Compare the models developed based on the prediction score to select the best model

Results

Using the final model, predict the price for the test dataset

EDA: ABOUT THE DATASET

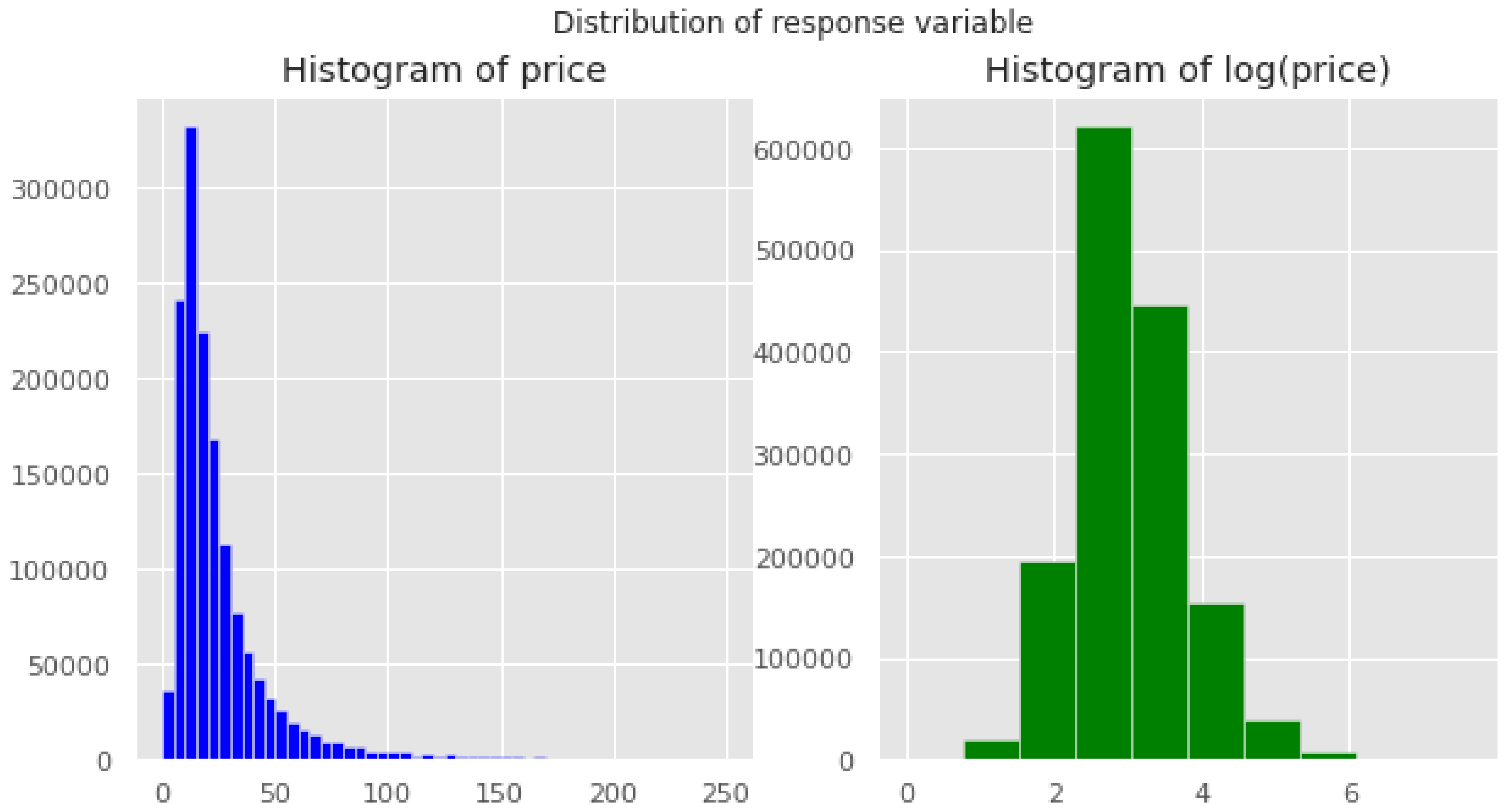
- The dataset consists of around 1.48 million rows and 8 columns

	train_id	name	item_condition_id	category_name	brand_name	price	shipping	item_description
0	0	MLB Cincinnati Reds T Shirt Size XL	3	Men/Tops/T-shirts	NaN	10.0	1	No description yet
1	1	Razer BlackWidow Chroma Keyboard	3	Electronics/Computers & Tablets/Components & P...	Razer	52.0	0	This keyboard is in great condition and works ...
2	2	AVA-VIV Blouse	1	Women/Tops & Blouses/Blouse	Target	10.0	1	Adorable top with a hint of lace and a key hol...
3	3	Leather Horse Statues	1	Home/Home Décor/Home Décor Accents	NaN	35.0	1	New with tags. Leather horses. Retail for [rm]...
4	4	24K GOLD plated rose	1	Women/Jewelry/Necklaces	NaN	44.0	0	Complete with certificate of authenticity

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1482535 entries, 0 to 1482534
Data columns (total 8 columns):
 #   Column             Non-Null Count    Dtype
---  ------             --------------    -----
 0   train_id           1482535 non-null  int64
 1   name               1482535 non-null  object
 2   item_condition_id  1482535 non-null  int64
 3   category_name      1476208 non-null  object
 4   brand_name         849853 non-null   object
 5   price              1482535 non-null  float64
 6   shipping           1482535 non-null  int64
 7   item_description   1482531 non-null  object
dtypes: float64(1), int64(3), object(4)
memory usage: 90.5+ MB
```

	Count of missing values	% missing values
brand_name	632682	42.675687
category_name	6327	0.426769
item_description	4	0.000270

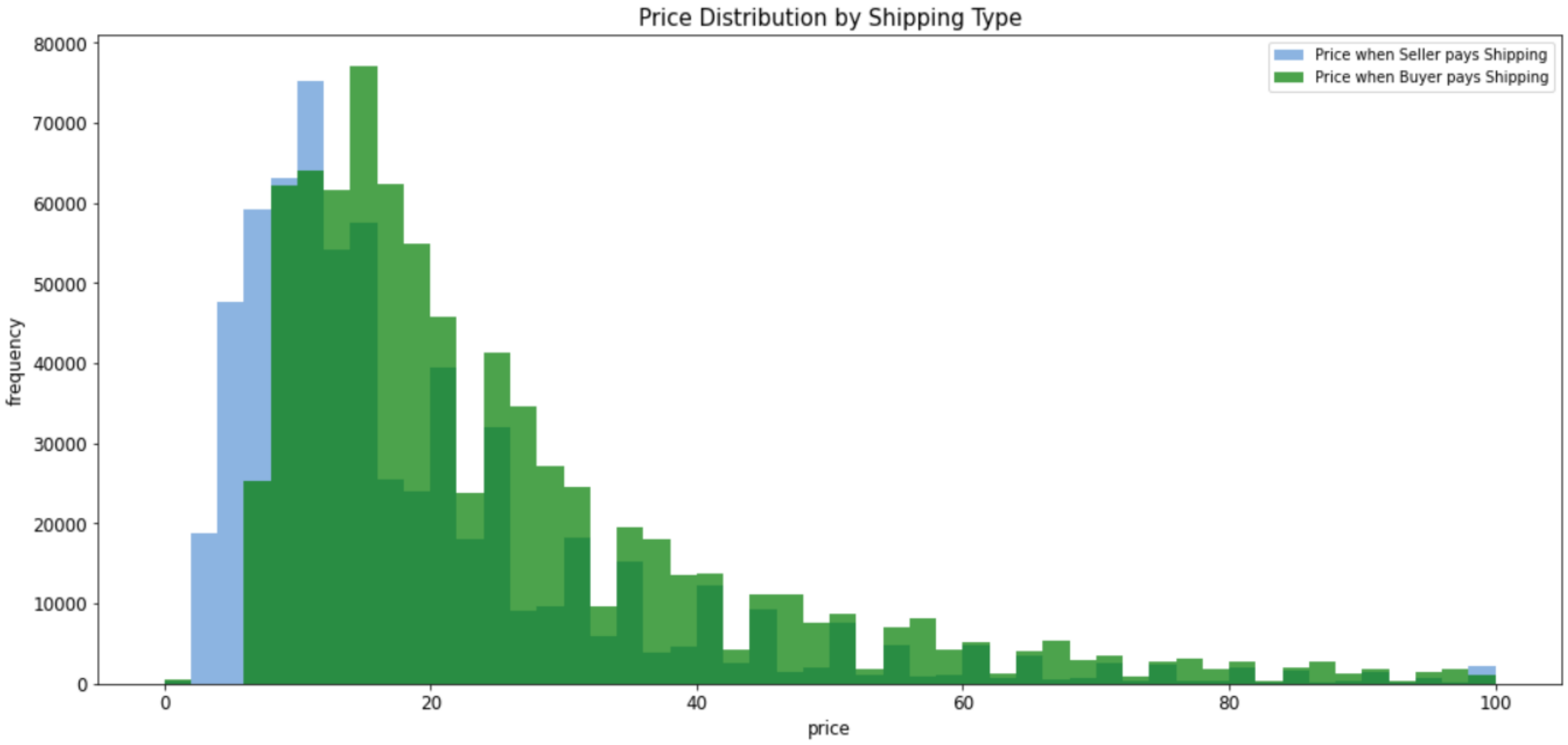
EDA – PRICE COLUMN




```
count    1.482535e+06
mean     2.673752e+01
std      3.858607e+01
min      0.000000e+00
25%      1.000000e+01
50%      1.700000e+01
75%      2.900000e+01
max      2.009000e+03
Name: price, dtype: float64
```

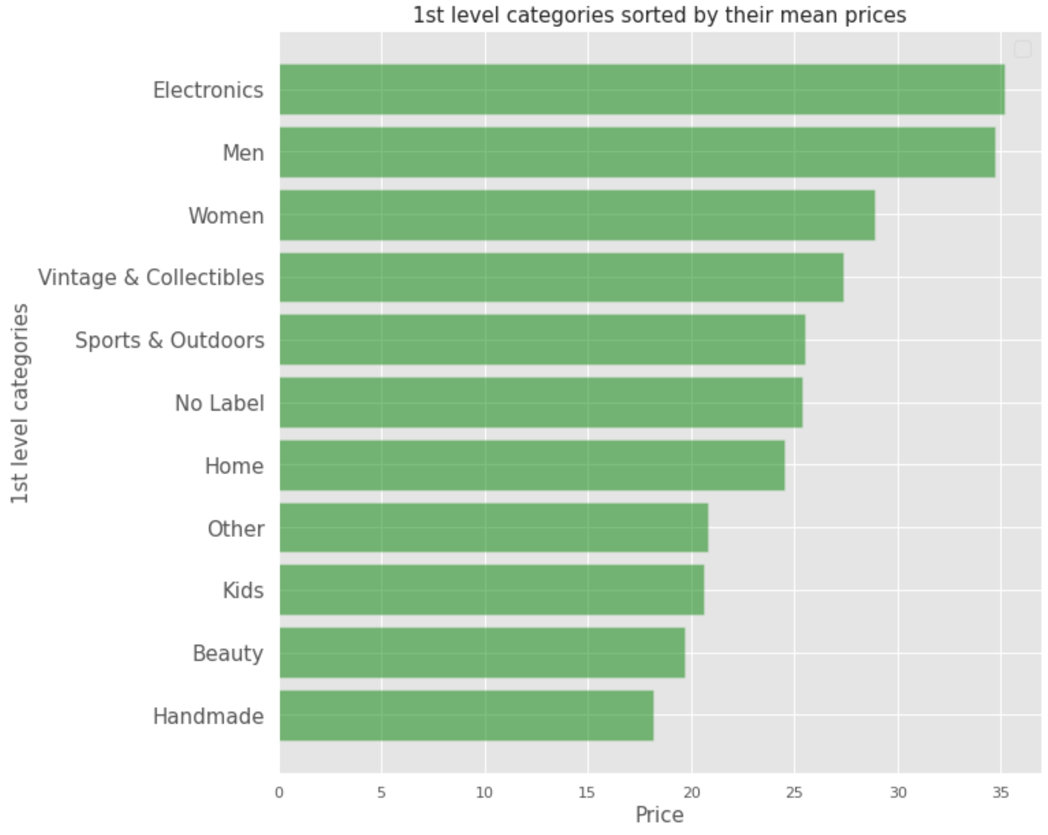
- The distribution of price is heavily skewed to the right and hence we decided to do a log transformation to correct that.

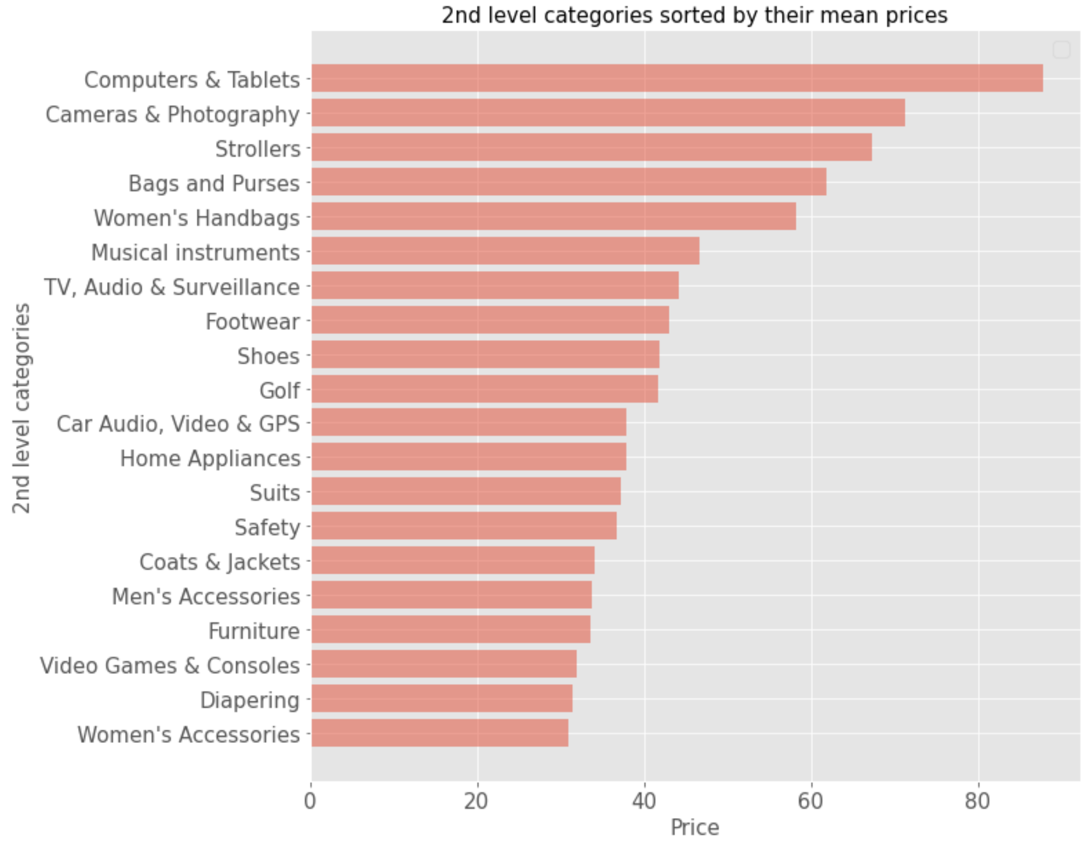
EDA: SHIPPING TYPE



- It was observed that for over 55% of items, shipping fee was paid by the buyers and the price is slightly higher if buyer pays for shipping

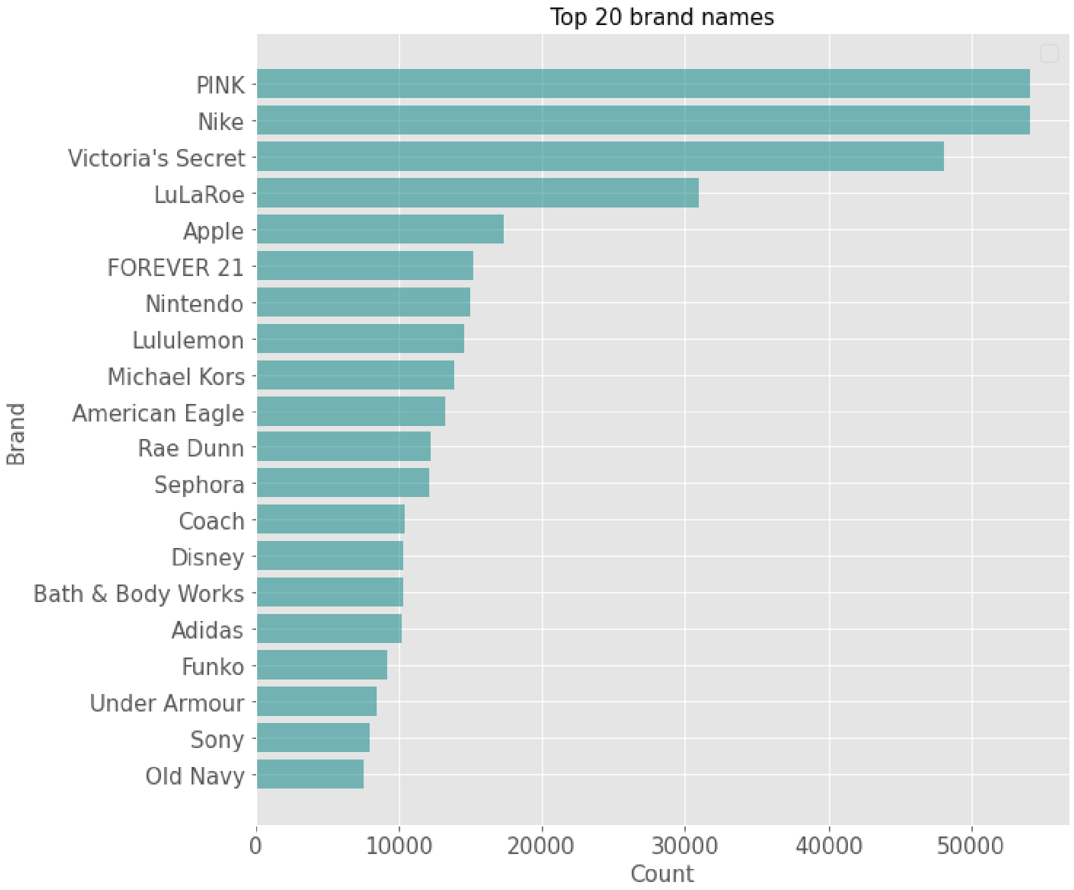
EDA: PRODUCT CATEGORIES





- We can observe that there is a huge difference of price looking into items categories. So, splitting categories into "levels" in our data can make a big difference when training our models.

EDA: BRAND NAMES



EDA: ITEM DESCRIPTION

- We can observe that words such as free shipping, good condition, excellent condition, occur frequently in the item description
- Item descriptions with certain words may have higher price associated with them

DATA PRE-PROCESSING

Missing Values treatment

- Replaced null values in `category_name` and `item_description` with the string ‘missing’
- Replaced missing values in `brand_name` by matching with computed ngrams values on product names

Text pre-processing

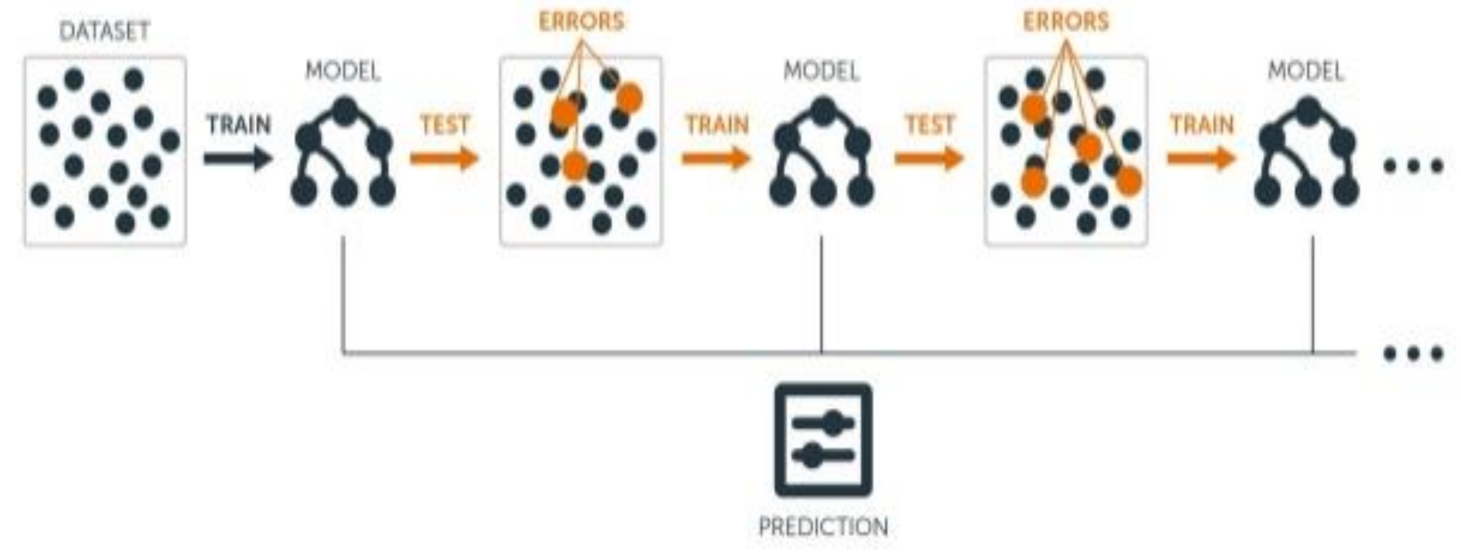
- Converting categorical features to numbers: One-hot encoding
- Converting text features to numbers: TF-IDF and Count vectorizer

Feature consolidation

- Stacked dense feature matrices with categorical and text vectors
- Converted the dense matrix into a sparse matrix

MODEL DEVELOPMENT

- We divided our data into 80-20, train-test ratio & built a regression model using **`LightGBM`**. We further defined a grid of 5 tree parameters and conducted hyperparameter tuning using 3-fold Cross-validation to check model efficiency.

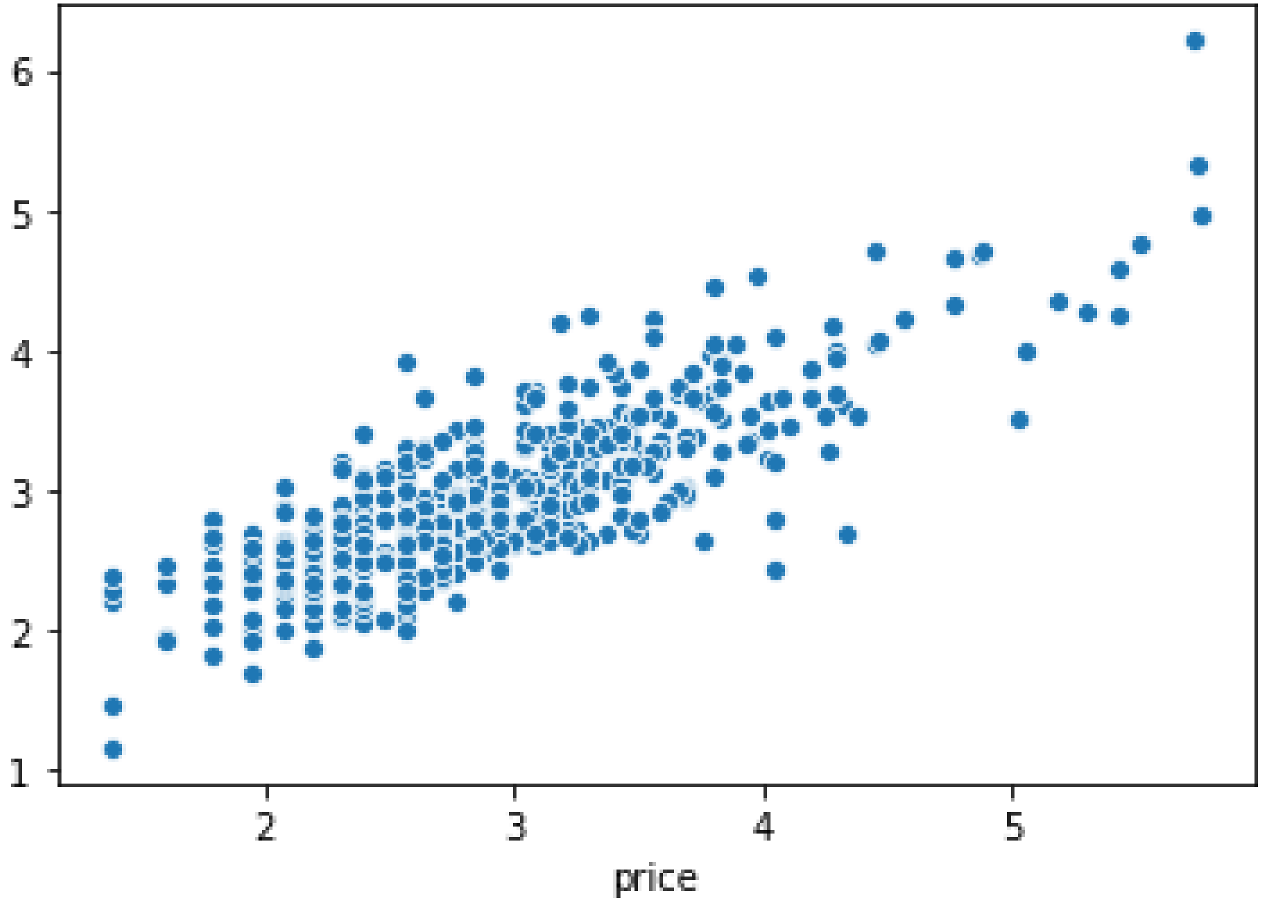


MODEL EVALUATION

- Below are the parameter values we obtained for our initial model and the model obtained using hyperparameter tuning

PARAMETERS	DEFAULT VALUE	TUNED VALUE
Learning Rate	0.75(Default value-0.1) *	0.24102
Number of Iterations	100	975
Number of Leaves	31	194
Maximum depth	3(Default value-infinite) *	13
Minimum Child weight	0.001	1.219
RMSE	0.4598	0.4111

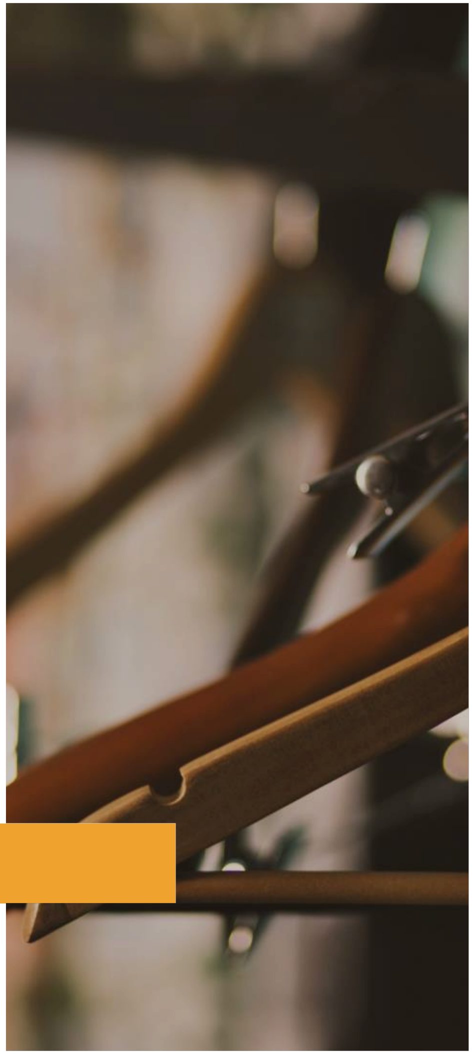
RESULTS



- Our Final model is a Gradient Boosted Regression Tree Model having a learning rate of 0.24 with a **RMSE (on the test dataset) of 0.4111**.

FUTURE SCOPE

- Use more complex models such as MLP, LSTMs, Convolutional Neural Nets and compare different models
- Other vectorization schemes such as Wordbatch, word2vec can be experimented with ML models.
- Regression models like Ridge, FTRL and FM_FTRL can also be tried.



THANK YOU