……………………………………………...……………………………………………...……………………………………………...……………………………………………………………...………………

**Name:** Ankit Singh

**Email address:** singhankitsstc@gmail.com

**Contact number:** 8770508488

**Anydesk address:**

**Years of Work Experience:** 5

**Date:** 07-07-2022

……………………………………………...……………………………………………...……………………………………………...……………………………………………………………...………………

**Self Case Study -1:** Mercari Price Suggestion Challenge

“After you have completed the document, please submit it in the classroom in the pdf format.”

Please check this video before you get started: <https://www.youtube.com/watch?time_continue=1&v=LBGU1_JO3kg>

# **Overview**

\*\*\* Write an overview of the case study that you are working on. ***(MINIMUM 200 words)*** \*\*\*

1. Mercari is a Japanese e-commerce company where users can sell their used products by uploading the pictures of their products on the Mercari’s website. The website displays more than 350k items are listed every day on the website which reflects its popularity among users. By Mercari Price Suggestion Challenge the company aims to build a model which will predict the price of the product uploaded by users on Mercari website the time they upload their products.

Below are the features of the dataset some of which are the details users have to provide while uploading their product’s image.

* 1. train\_id or test\_id - the id of the listing
  2. name - the title of the listing.
  3. item\_condition\_id - the condition of the items provided by the seller. Seller needs to provide the condition their product is in from 1 to 5 with 1 being the best condition.
  4. category\_name - category of the listing
  5. brand\_name – Name of the brand of the product. The feature also contains null values.
  6. price - the price that the item was sold for. This is the target variable that we will predict. The unit is USD. This column doesn't exist in test.tsv since that is what we will predict.
  7. shipping - 1 if shipping fee is paid by seller and 0 by buyer.
  8. item\_description - the full description of the item.

1. Formulating this as a Machine Learning problem, we have a regression problem at hand where the model has to predict the price of the product user is uploading. Constraint is that the model has 60 minutes to train and predict. Because of this constraint, we cannot create huge ensembles and it has a low latency requirement.
2. The performance metric is Root Mean Squared Logarithmic Error.
   1. The choice of this metric is due to the following reasons,
   2. It is robust to outliers.
   3. It is scale invariant.
   4. It will penalize under-estimation more, than over-estimation. This makes sense from the business perspective, because ideally the company would like to sell the products at a reasonably higher price. This also increases the customer (seller) satisfaction as well as the commission that the company might get for each sale. Another reason is that, the price distribution is highly skewed to the right (this we will see in the EDA). Meaning there are a lot of low priced products as compared to very high priced products.

# **Research-Papers/Solutions/Architectures/Kernels**

\*\*\* Mention the urls of existing research-papers/solutions/kernels on your problem statement and in your own words write a detailed summary for each one of them. If needed you can include images or explain with your own diagrams. it is mandatory to write a brief description about that paper. Without understanding of the resource please don’t mention it\*\*\*

Problem details:

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

1. <https://towardsdatascience.com/mercari-price-prediction-challenge-3a8ea00a7d33>

EDA/Observations:

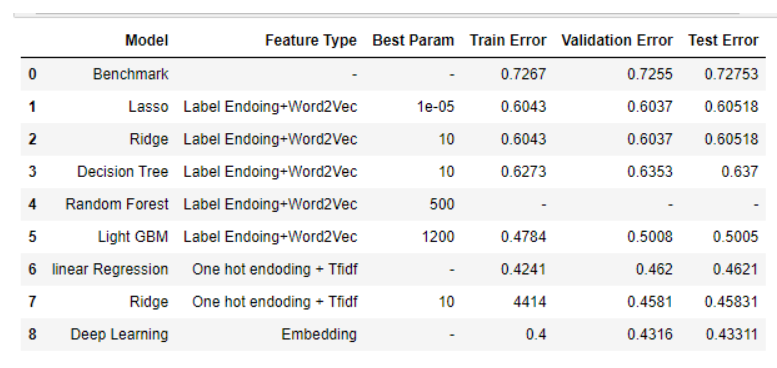
* The distribution for Price (dependent variable) is skewed so we will use Log of Price, which gives a normal distribution.
* Brand feature contains lot of null values, we are going to replace missing values with the value ‘missing’ during preprocessing. There are 4800 unique brands.
* In the category column, categories are hierarchical. one main category holds sub-categories and so on or we can say there are different Tiers separated by ‘/’.
* In the first level, there is a total of 10 categories, In the second tier, we have more than 100 categories, In the third tier, we can observe more than 800 categories.
* The item description is an important feature for modeling. It holds descriptive information about the product given by the seller.

Data Preprocessing & Feature Engineering:

* For text features, de-contraction, removing special characters, stop-word removal and lowering the text.
* Gradient Boosting based technique proves to be most useful ML technique for predicting the scores.
* When we use model with high dimensions, we have Ridge Model give the best results.
* We can try RNN model which works well for text data.
* Various featurization tried –

1. Keeping dimensions low: Ordinal encoding on tier1, tier2 and tier3 features and branch\_name. Average Word-to-Vec on text data after joining name & item description columns. – For these features **LGBMRegressor** gave better results compared to Linear regression, Decision Tree & Random Forest models.
2. High num of features: One Hot Encoding for Categorical Data & TF-IDF for text data. They used bigram and maximum features up to 50000 for both Name and Item Description. (they calculated tfidf separately for Name & Description).

For these features, Ridge Model gives best results which are also better than the results in 1st approach.



2) <https://medium.com/swlh/mercari-price-suggestion-challenge-an-end-to-end-machine-learning-case-study-4a6d833fa1c7>

Another blog for Mercari Price Suggestion Challenge

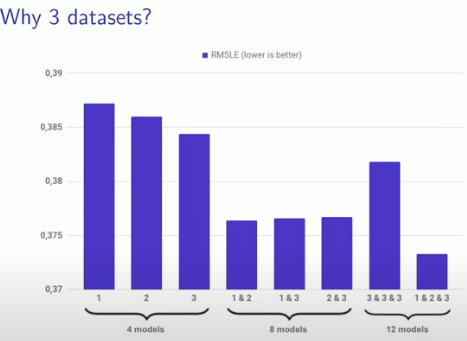
* Key Takeaway –
  + For brand\_name, not only the basic text preprocessing is done but as we discussed earlier in EDA that there are certain products that include brand\_names in their names, so we have to extract them.
  + To extract the brand names, what they did is:
    - Got list of existing brand\_names
    - Create dictionary of brand\_names. This will be used to check whether the “guessed” brand name is of same category as the brand name that we are filling. This is for extra caution.
    - Using this they filled around 27% of missing brand names.
  + They added a new feature which tells us if a brand is expensive or cheap. This was done by determining whether a brand is cheap or expensive for their respective category. E.g. if a T-shirt’s cost is in the range of 20-50$ then a brand whose t-shirts are of 100$ would be considered expensive.
  + Ridge / Lasso gave good results but they got better results with Deep Learning models like CNN, GRU and their combinations and ensembles.
  + They have also performed stemming.
  + They used tf-idf vectorizer for text featurization.

3) <https://www.youtube.com/watch?v=QFR0IHbzA30&t=1125s>

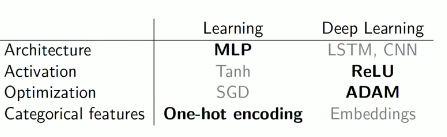
Finally, we have the 1st prize winning team’s solution

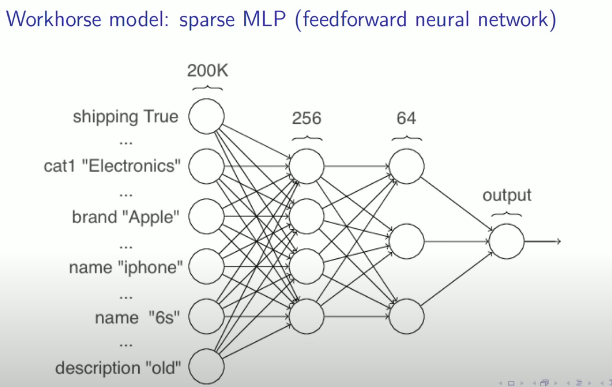
Key Takeaways –

* Feature engineering is not too important.
* Normalize the text to some extent?????
* Preprocessing used – Stemming for text processing, BOW – 1-2grams (with/without tfidf), One-hot encoding for categorical columns
* They used Bag of characters for Names – used trigrams and it gave good results
* They joined name, brand name and description into a single field so that we don’t have to pre-process them seperately.
* They used ensembles with 4 models and they used 3 datasets to train a total of 12 models. 3 datasets were used in combinations to diversify the data



* Approach used – they have taken combination of MLP and Deep Learning



* Architecture used by them, we can see the sparse MLP having 200k features.

4) <https://towardsdatascience.com/solving-regression-problems-by-combining-statistical-learning-with-machine-learning-82949f7ac18a>

This is a good blog which shows how we can use statistics to help determine if a feature is useful.

Key Takeaway:

* They have used statistical tests to determine if a feature is useful in predicting the price or not. For a categorical value if p value < 0.05, it means categorical feature is independent of the continuous variable and it does not have any predicting power because there is no observed variance in price between categorical levels.
* We can use the above tests to check during EDA if the categories are helpful in predicting the output. But the point to keep in mind is that Anova test is not very helpful if number of categories is high.

# **First Cut Approach**

\*\*\* Explain in steps about how you want to approach this problem and the initial experiments that you want to do. ***(MINIMUM 200 words)*** \*\*\*

\*\*\* When you are doing the basic EDA and building the First Cut Approach you should not refer any blogs or papers \*\*\*

1. We have very few features hence we don’t need to do feature selection. We need to do feature transformation for category feature by splitting it into 3 categories. One hot encoding works well for encoding category data after breaking it into 3 groups, I will try to experiment with PCA if that helps in improving results in the end if it can decrease dimensions by explaining preserving variance.
2. For missing values in Brand Name, I will follow the approach used by others where they extract the brand names from the Name of the product, remaining missing values we would replace with ‘missing’.
3. I have observed different teams performing different text vectorization, most of the solutions have combined ‘Name’ and ‘Description’ features but the winning team combined ‘Name’, ‘Description’ & Brand\_name column in a single combined feature before using vectorization. So I would like to experiment with both and see how the results change.
4. We have got good results with Ridge regression when we do one hot encoding but Deep learning models have given good results too so I will use them as well.
5. For encoding of text, tf-idf seems to be working well with Machine learning models and bow with deep learning models.
6. In the end, I will implement the ensemble model used by the winning team where they have used 12 models where they got good difference in results
7. Another trick winning team implemented was the use of character grams till tri-grams while working with deep learning model so I would like to experiement with that too.

**Notes when you build your final notebook**:

1. You should not train any model either it can be a ML model or DL model or Countvectorizer or even simple StandardScalar
2. You should not read train data files
3. The function1 takes only one argument “X” (a single data points i.e 1\*d feature) and the inside the function you will preprocess data point similar to the process you did while you featurize your train data
   1. Ex: consider you are doing taxi demand prediction case study (problem definition: given a time and location predict the number of pickups that can happen)
   2. so in your final notebook, you need to pass only those two values
   3. def final(X):

preprocess data i.e data cleaning, filling missing values etc

compute features based on this X

use pre trained model

return predicted outputs

final([time, location])

* 1. in the instructions, we have mentioned two functions one with original values and one without it
  2. final([time, location]) # in this function you need to return the predictions, no need to compute the metric
  3. final(set of [time, location] values, corresponding Y values) # when you pass the Y values, we can compute the error metric(Y, y\_predict)

1. After you have preprocessed the data point you will featurize it, with the help of trained vectorizers or methods you have followed for your train data
2. Assume this function is like you are productionizing the best model you have built, you need to measure the time for predicting and report the time. Make sure you keep the time as low as possible
3. Check this live session: <https://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4148/hands-on-live-session-deploy-an-ml-model-using-apis-on-aws/5/module-5-feature-engineering-productionization-and-deployment-of-ml-models>