**Dataset:**

1. The training file (.tsv) consist of a list of product listings which is tab-delimited.
2. train\_id - the id of the listing
3. name - the title of the listing
4. item\_condition\_id - the condition of the items provided by the seller. From 1 to 5. 1 being new and 5 being poor
5. category\_name - category of the listing
6. brand\_name
7. Price - the price that the item was sold for. This is the target variable that we will predict. The unit is USD.
8. shipping - 1 if shipping fee is paid by seller and 0 by buyer
9. item\_description - the full description of the item.
10. The size of the train.tsv = 322 MB.  
    Total number of product listings = 14, 82,535

**Steps taken in the project:**

1. Observe the target variable spread
2. Apply the log transformation on the target variable “Price”
3. Fill in the missing values
4. Textual data processing
   1. De contract text (Change don’t to do not)
   2. Stem sentences
   3. Pre-process text data
      1. Firstly de contract text
      2. Remove stop words
      3. Remove any special characters
      4. Stem sentences
5. Fill in missing brand names
   1. Save unique brand names
   2. Save unique brand names categories
   3. Check for the brand names in Names feature (as some might mention the name in this feature).

Check if brand name matches or is in Name feature and if the category matches the current product category, if yes then fill in the brand name

1. Split the categories
   1. Split the categories in three features
      1. General
      2. Sub cat 1
      3. Sub cat 2
2. Item Description and Item names length
3. Is\_expensive feature
   1. Calculate the 95 percentile for each general category
   2. Check if the (brand name, price) of the current product exceeds the 95 percentile of the general category which was calcualted
   3. If any brand name value has products those are costing above this mark then it is expensive brand
   4. Add another feature which shows if the current product is expensive or not
4. Make the shipping feature sparse
   1. Make the matrix have only non-zero values along with their index
   2. To save the storage space
5. Vectorise the data
   1. To transform the textual column data into numeric data, use the Tf-idf vectorizer which will produce max 100,000 features

**Feature Pipeline**

This function will do all the feature engineering on the input X\_data, and create a final data, ready for training.

1. Filling Missing Values
2. Pre-processing text data
3. Filling Missing Brand Values
4. Splitting Categories
5. Getting Lengths of text data
6. Getting Is\_expensive feature
7. Getting sparse shipping data
8. Vectorising the textual features (6 Features : General Category, Sub Cat 1, Sub Cat 2, Brand Name, Item Name, Item Description)
9. Creating the final dataset
   1. Horizontally stack the data
10. Done
11. general\_cat\_vectorizer, subcat\_1\_vectorizer, subcat\_2\_vectorizer, brand\_name\_vectorizer, item\_name\_vectorizer, item\_desc\_vectorizer, scaler\_name\_len, scaler\_desc\_len are returned when training data processed
12. These vectorizer are then passed to feature pipeline again so that we can process the test data using the knowledge of training dataset as we cannot use fit on test data

**Final Data Set Shape:**

**(1186028, 98635) (1186028,)**