Mercari Price Suggestion Challenge

Ali Ezzat, Mohammed Ali

December 30, 2017

Contents

About

Mercari's challenge is to build an algorithm that automatically suggests the right product prices. You'll be provided user-inputted text descriptions of their products, including details like product category name, brand name, and item condition.

Exploratory Data Analysis

Data overview

- Load training data

```
train <- fread("data/train.tsv", sep = "\t", stringsAsFactors = FALSE, showProgress = FALSE)
```

- Inspect structure

```
glimpse(train)
```

```
## Observations: 1,482,535
## Variables: 8
## $ train_id          <int> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13...
## $ name              <chr> "MLB Cincinnati Reds T Shirt Size XL", "Raze...
## $ item_condition_id <int> 3, 3, 1, 1, 1, 3, 3, 3, 3, 3, 2, 1, 2, 1, 3,...
## $ category_name     <chr> "Men/Tops/T-shirts", "Electronics/Computers ...
```

```
## $ brand_name        <chr> "", "Razer", "Target", "", "", "", "Acacia S...
## $ price             <dbl> 10, 52, 10, 35, 44, 59, 64, 6, 19, 8, 8, 34,...
## $ shipping          <int> 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0,...
## $ item_description  <chr> "No description yet", "This keyboard is in g...
```

At the first glance, excluding the *train_id* column, the numeric data are only 3 columns one of them is the response variable *price*, while the other two are factor data which mean that we have a very limited number of features and that mean we will need the help of the other four text predictors to build our model. So let us investigate these features one by one to see what we can do.

Price (The response/target variable)

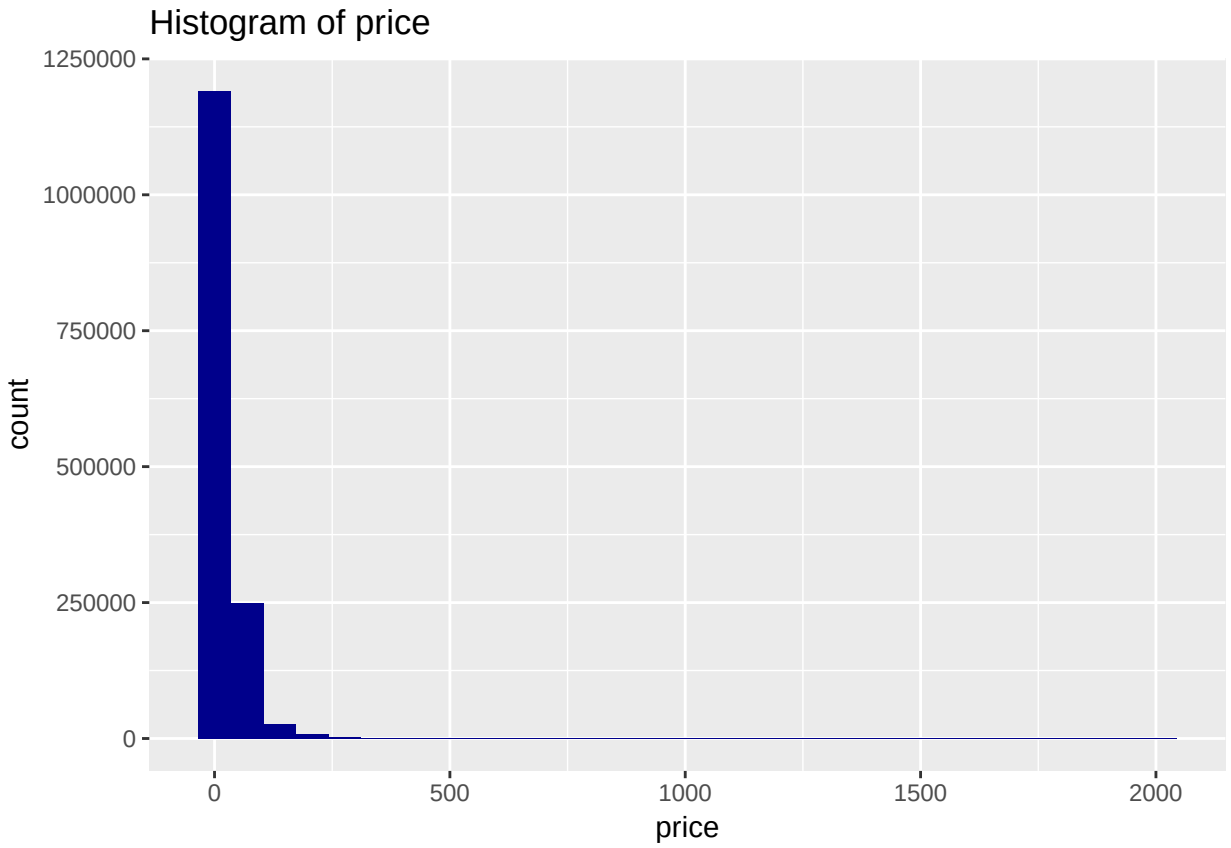
Let's start with an analysis of the response0 variable: price. First, the range of item prices:

```
summary(train$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   10.00   17.00   26.74   29.00 2009.00
```

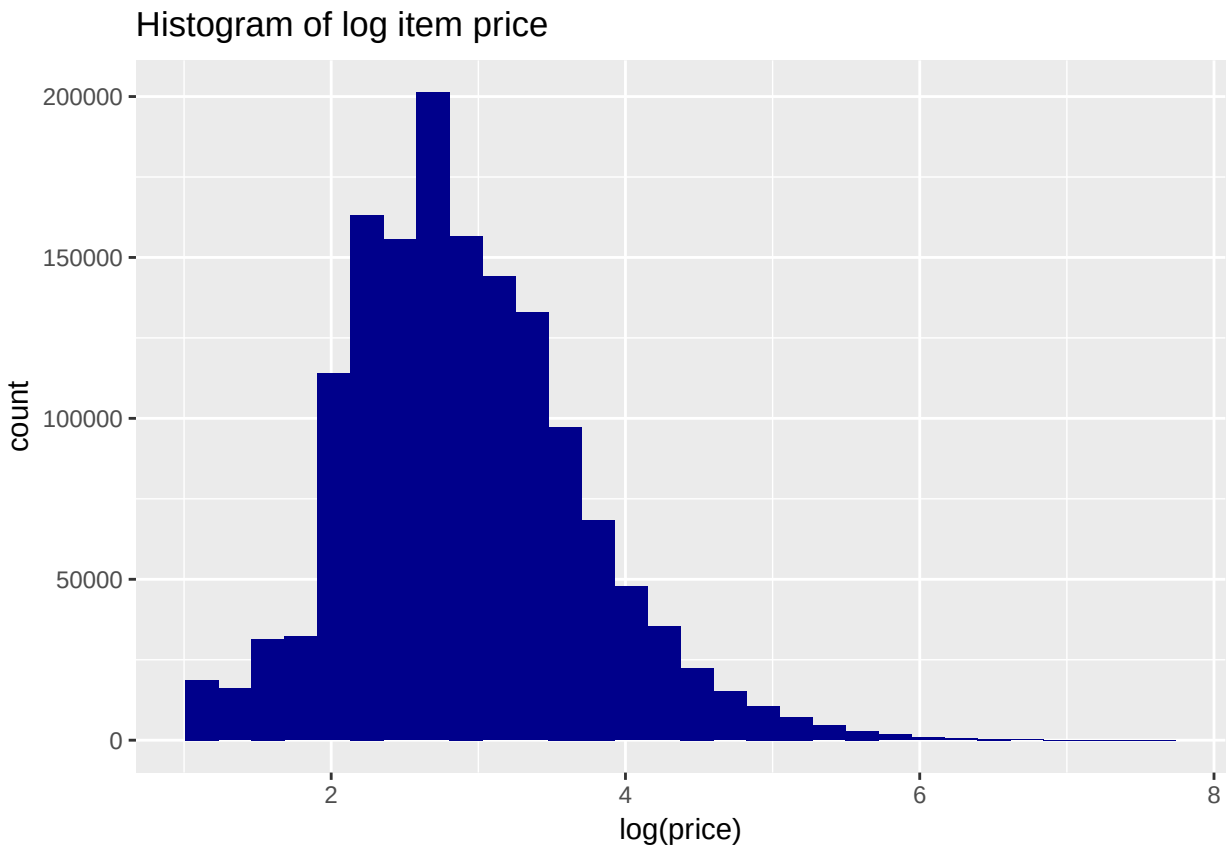
It seems there are items are givem as gifts (min. is 0), and we have a few very expensive items as well. Let us visualize it for more clear picture

```
ggplot(data = train, aes(x = price)) +
  geom_histogram(fill = 'darkblue') +
  labs(title = 'Histogram of price')
```



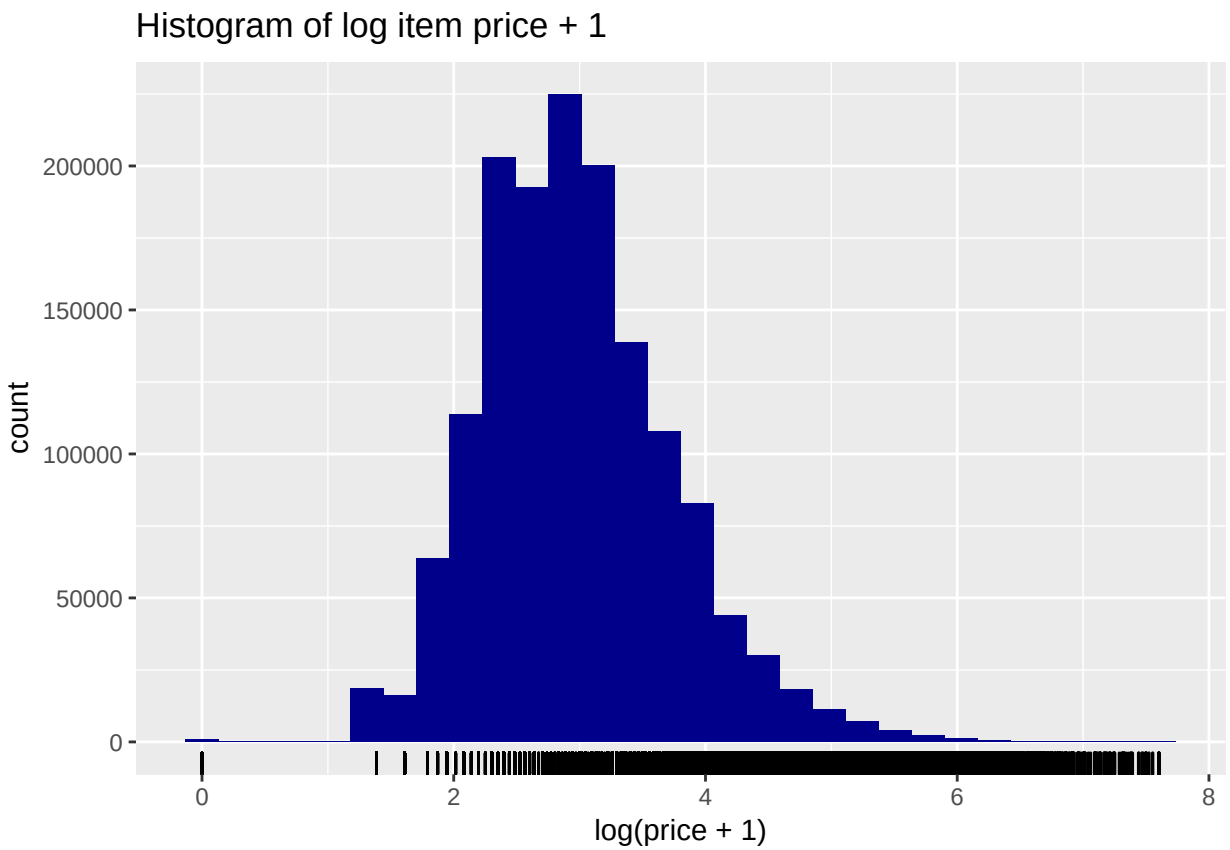
It seems because we have a very expensive items (and very view) we will need to take the log for better analysis

```
ggplot(data = train, aes(x = log(price))) +
  geom_histogram(fill = 'darkblue') +
  labs(title = 'Histogram of log item price')
```



Now, it is much better and clear, but it seems taking log of prices made the free/gift items to be omitted, as taking log for 0 is undefined, so we will have to add a dummy 1 to include it with us in the plot, consider it a tax :P .

```
ggplot(data = train, aes(x = log(price + 1))) +
    geom_histogram(fill = 'darkblue') +
    labs(title = 'Histogram of log item price + 1') +
  geom_rug()
```



So, finally we have the gift/free items in the plot and it seems like an outlier along with the pricy data, as we have a nearly normal distibution for the prices. Let us investigate the distribution more. It seems the data is centered between 3 and 7, as also the following statistics confirm.

```
summary(log(train$price + 1))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.398   2.890   2.979   3.401   7.606
```

Item Condition

Item condition is a factor data, so let us convert it to a factor first.

```
#item condition factor
train$item_condition_id <- as.factor(train$item_condition_id)
levels(train$item_condition_id)
```

```
## [1] "1" "2" "3" "4" "5"
```

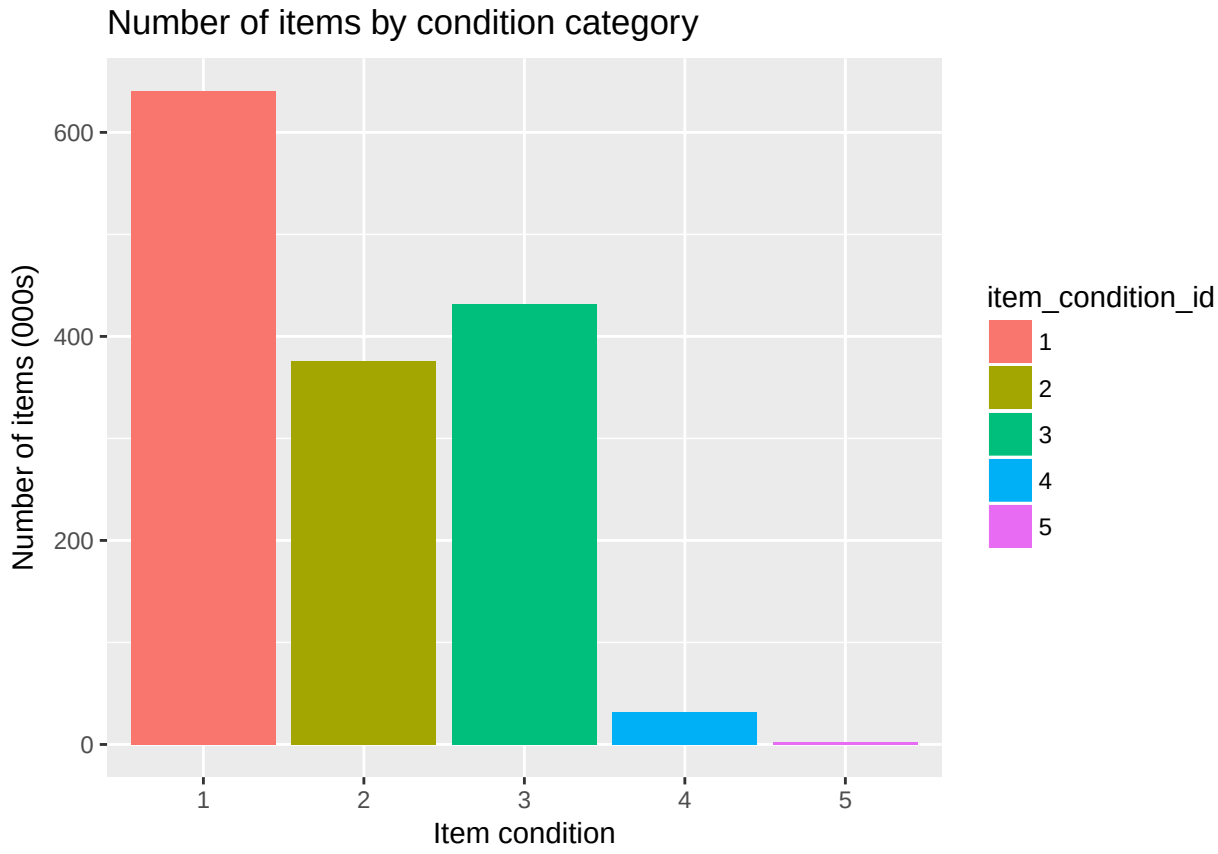
Now, let us see the statistics summary of items conditions

```
table(train$item_condition_id)
```

```
##
##      1      2      3      4      5
## 640549 375479 432161  31962   2384
```

We can confirm it more by the following plot

```
train[, .N, by = item_condition_id] %>% ggplot(aes(x = item_condition_id, y = N/1000,
    fill = item_condition_id)) + geom_bar(stat = "identity") + labs(x = "Item condition",
    y = "Number of items (000s)", title = "Number of items by condition category")
```



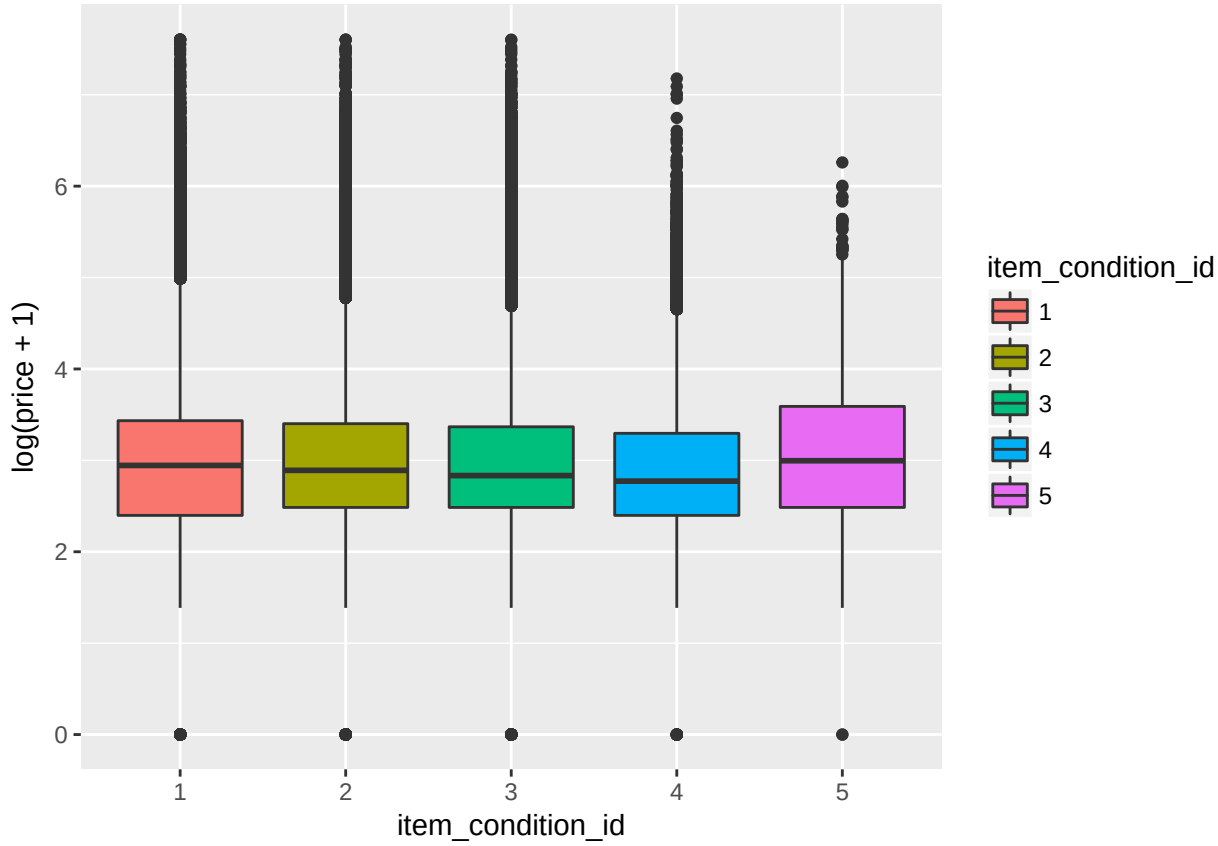
It seems most of the items are in condition *1* which is, I do not know there is no ordinal description in the compition. So we do not know if it *1* is the best or the worst. However, hanks to kaggler @Juraj for pointing out that in fact condition 1 is the best and 5 is the worst.

Now, let us compare the *item conditions* predictor aginst the response variable *price*

```
train[, .(.N, median_price = median(price)), by = item_condition_id][order(item_condition_id)]
```

```
##    item_condition_id      N median_price
## 1:                 1 640549           18
## 2:                 2 375479           17
## 3:                 3 432161           16
## 4:                 4  31962           15
## 5:                 5   2384           19
```

```
ggplot(data = train, aes(x = item_condition_id, y = log(price + 1), fill = item_condition_id)) +
    geom_boxplot()
```



It seems that item condition is not the main contributor to the price as the best price at condition *5* with few items and the second best price at condition *1* with a lot of items.

Shipping

It is the second numeric facotr predictor. It is simply *1* when shipping item is paied by the seller and *0* otherwise. Let us convert it to a facotr and see its statistics

```
#shipping
train$shipping <- as.factor(train$shipping)
levels(train$shipping)
```

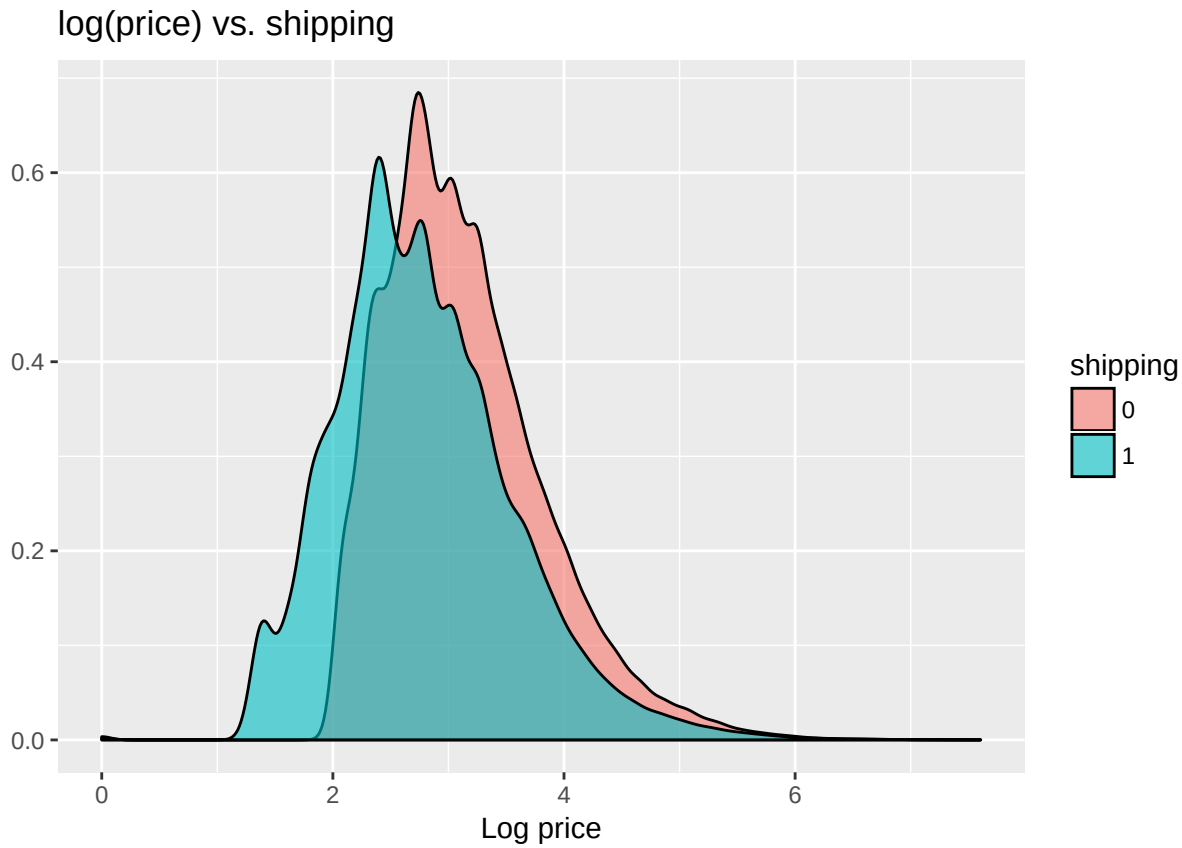
```
## [1] "0" "1"
```

```
table(train$shipping)
```

```
##
##      0      1
## 819435 663100
```

It does not seem contributing that much to the reponse variable, so let us investigate its relation with the *price*

```
train %>%
    ggplot(aes(x = log(price+1), fill = shipping)) +
    geom_density(adjust = 2, alpha = 0.6) +
    labs(x = 'Log price', y = '', title = 'log(price) vs. shipping')
```



It seems that if you are going to pay for the shipping you will have a little lower price, but not that much right !!

Brand

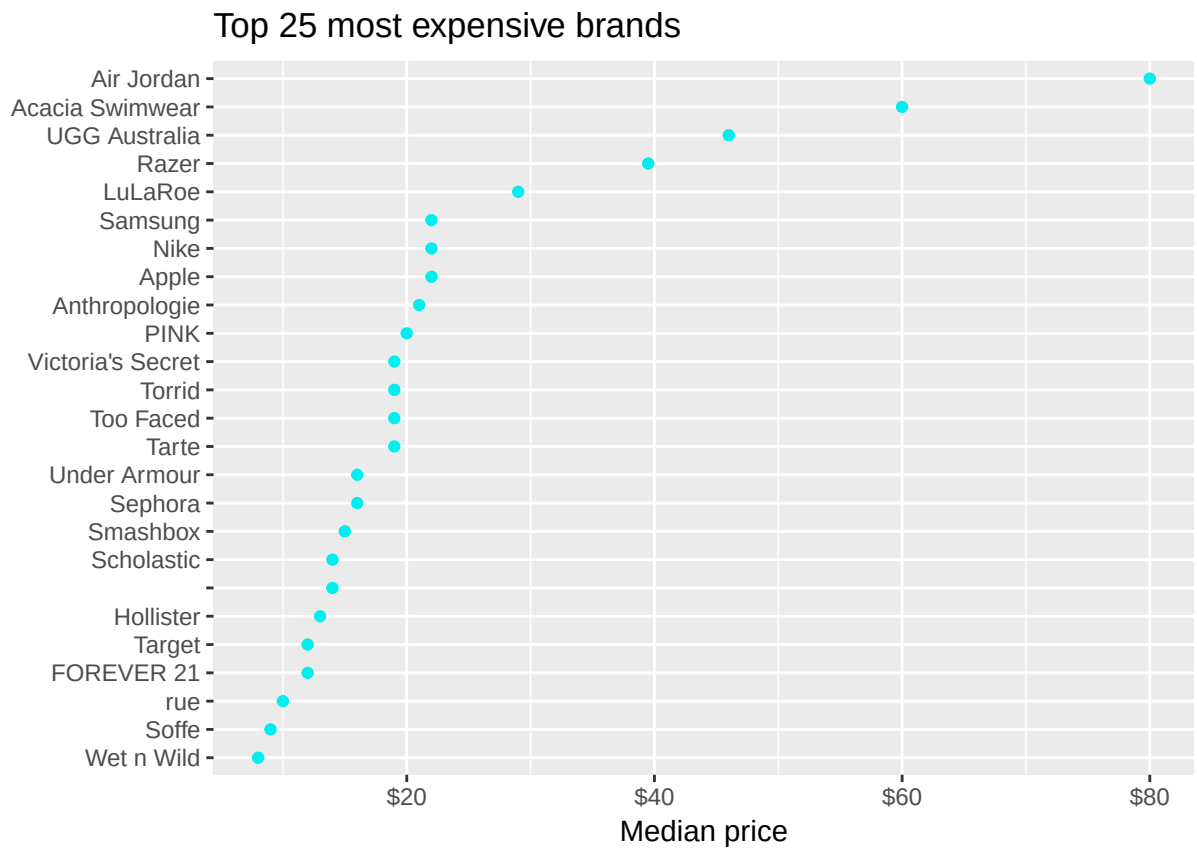
Clearly the two numeric predictor factors are not contributing so much into the response variable. How many brands we have?

```
length(unique(train$brand_name))
```

```
## [1] 4810
```

Wow, that is a lot. So, let us try to investigate the text predictors and begin with the *brand* predictor.

```
train[, .(median_price = median(price)), by = brand_name] %>%
  head(25) %>%
  ggplot(aes(x = reorder(brand_name, median_price), y = median_price)) +
  geom_point(color = 'cyan2') +
  scale_y_continuous(labels = scales::dollar) +
  coord_flip() +
  labs(x = '', y = 'Median price', title = 'Top 25 most expensive brands')
```



OK, finally we have a strongly contributer predictor. It seems that brands affect prices as it should be.

Item Categories

Now it is *item categories* turn, we expect a lot from this variable. Let us begin by see how many categoris are there?

```
length(unique(train$category_name))
```

```
## [1] 1288
```

OK, there are a lot of categories here. So, how it looks like?

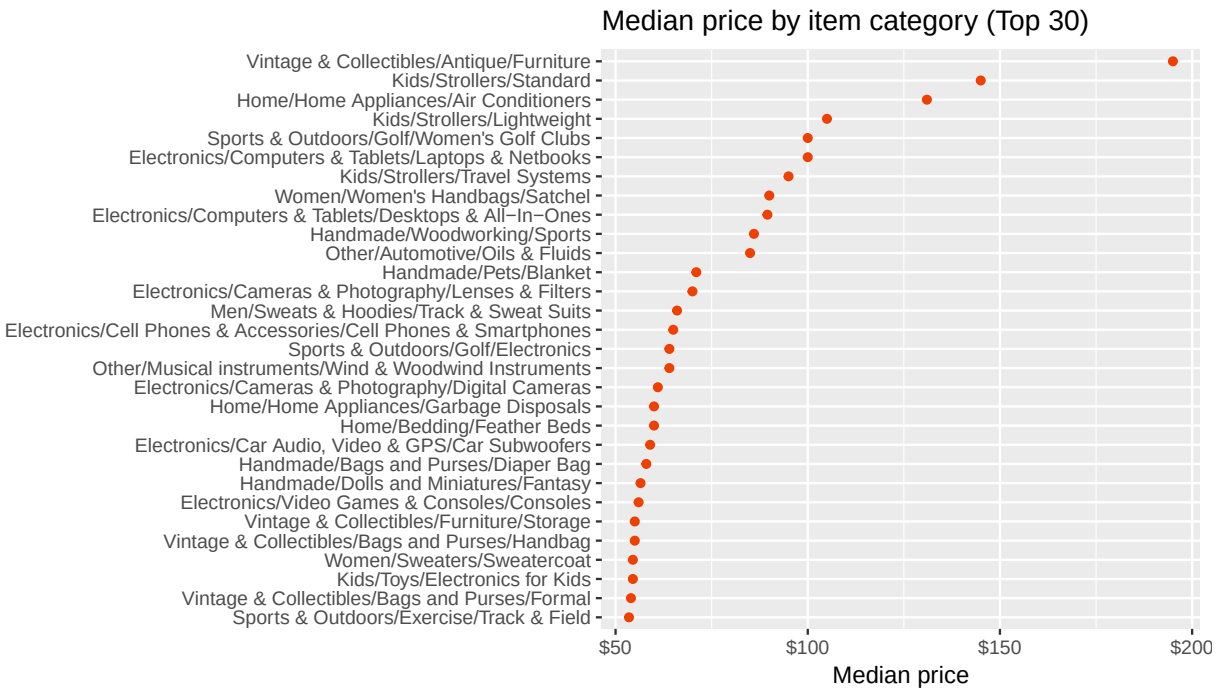
```
sort(table(train$category_name), decreasing = TRUE)[1:10]
```

```
##
##              Women/Athletic Apparel/Pants, Tights, Leggings
##                                                       60177
##                                  Women/Tops & Blouses/T-Shirts
##                                                       46380
##                                             Beauty/Makeup/Face
##                                                       34335
##                                             Beauty/Makeup/Lips
##                                                       29910
##                    Electronics/Video Games & Consoles/Games
##                                                       26557
##                                             Beauty/Makeup/Eyes
##                                                       25215
```

```
## Electronics/Cell Phones & Accessories/Cases, Covers & Skins
##                                                       24676
##                                          Women/Underwear/Bras
##                                                       21274
##                                 Women/Tops & Blouses/Blouse
##                                                       20284
##                               Women/Tops & Blouses/Tank, Cami
##                                                       20284
```

OK, there are a lot for women here :). It seems that we have a 3 level of category here, so let us investigate more. Now let us see how cateogries influnce *price*

```
train[, .(median = median(price)), by = category_name][order(median, decreasing = TRUE)][1:30] %>%
    ggplot(aes(x = reorder(category_name, median), y = median)) + geom_point(color = "orangered2") +
    coord_flip() + labs(x = "", y = "Median price", title = "Median price by item category (Top 30)") +
    scale_y_continuous(labels = scales::dollar)
```



As expected, categories are contributing so much into the response variable *price* with *Vintage & Collectibles* in the top followed by *kids toys* !! So let us split each category to its level to see how each level contributs in the *price*. Thanks to Abhinav Reddy Kaitha we knew that there are some items with four levels instead of three). However, after running data itseems we have even a more level so we have 5 levels in total

```
splitted_categ <- str_split_fixed(train$category_name, "/", 5)
train[, `:=`(c("level_1_cat", "level_2_cat", "level_3_cat", "level_4_cat", "level_5_cat"),
    .(splitted_categ[, 1], splitted_categ[, 2], splitted_categ[, 3], splitted_categ[,
        4], splitted_categ[, 5]))]

train %>% summarise(Num_Cat1 = length(unique(level_1_cat)), Num_Cat2 = length(unique(level_2_cat)),
    Num_Cat3 = length(unique(level_3_cat)), Num_Cat4 = length(unique(level_4_cat)),
    Num_Cat5 = length(unique(level_5_cat)))
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.2
```

```
##   Num_Cat1 Num_Cat2 Num_Cat3 Num_Cat4 Num_Cat5
```

```
## 1       11      114      871        7        3
```

We can now convert them to factors for easier and better analysis

```
#train$level_1_cat <- as.factor(train$level_1_cat)
#train$level_2_cat <- as.factor(train$level_2_cat)
#train$level_3_cat <- as.factor(train$level_3_cat)
#train$level_4_cat <- as.factor(train$level_4_cat)
#train$level_5_cat <- as.factor(train$level_5_cat)
# Full summary
#train %>% summarise(no_Level_1_Items = length(unique(level_1_cat)),
 #                   no_Level_2_Items = length(unique(level_2_cat)),
  #                  no_Level_3_Items = length(unique(level_3_cat)),
   #                 no_Level_4_Items = length(unique(level_4_cat)),
    #                no_Level_5_Items = length(unique(level_5_cat)))
```

Now let us investigate each level

level 1

The statistics of level1

```
length(levels(train$level_1_cat))
```

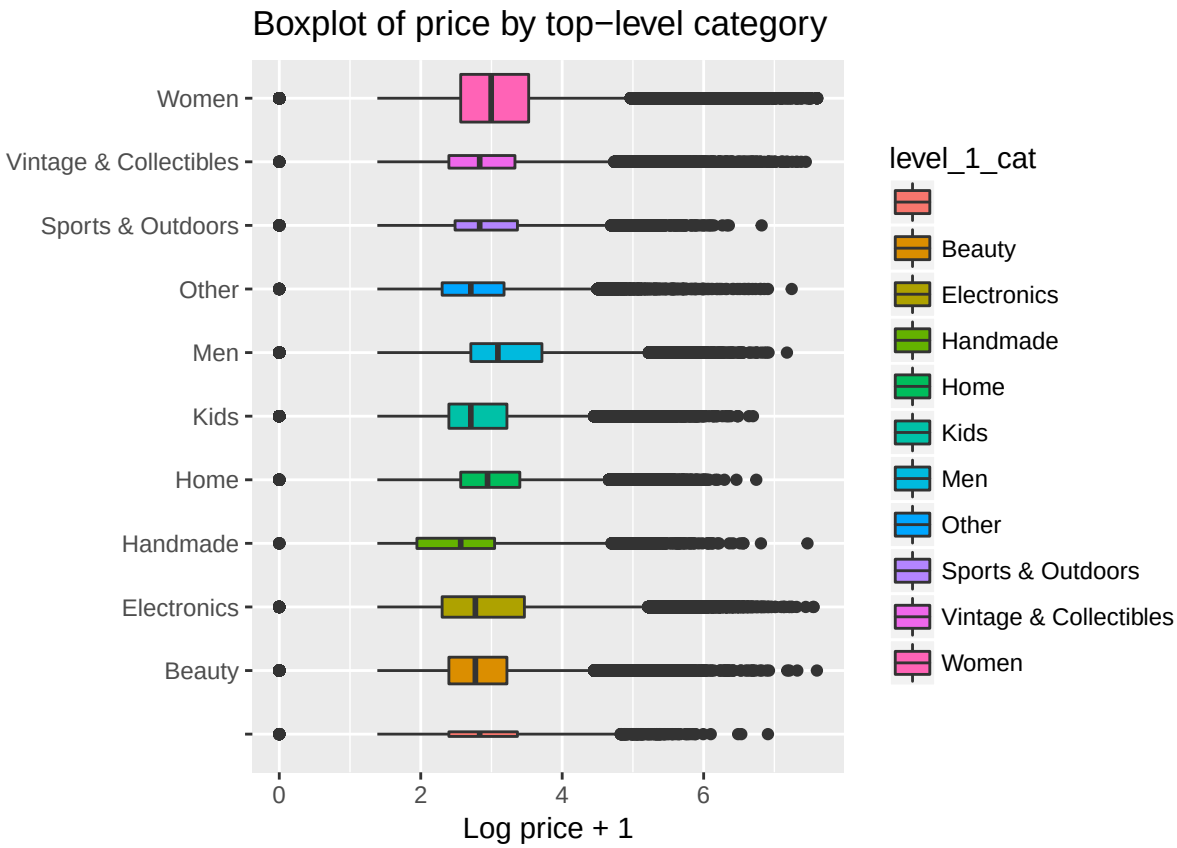
```
## [1] 0
```

```
table(train$level_1_cat)
```

```
##
##                                  Beauty            Electronics
##                    6327          207828                 122690
##                Handmade            Home                   Kids
##                   30842           67871                 171689
##                     Men           Other      Sports & Outdoors
##                   93680           45351                  25342
## Vintage & Collectibles           Women
##                   46530          664385
```

So, we have 10 items, let us its relation with *price*

```
train %>%
    ggplot(aes(x = level_1_cat, y = log(price+1), fill = level_1_cat)) +
    geom_boxplot(varwidth = TRUE) +
    coord_flip() +
    labs(x = '', y = 'Log price + 1', title = 'Boxplot of price by top-level category')
```



So, we have items with no category but its prices are higher than ___Homemade_ items which is the *lowes* and less than **Men** items which are surprisingly higher than **Women** items in average. However, *Women* items still alot more than any other item.

level 2

Statistics of level 2

```
length(levels(train$level_2_cat))
```

```
## [1] 0
```

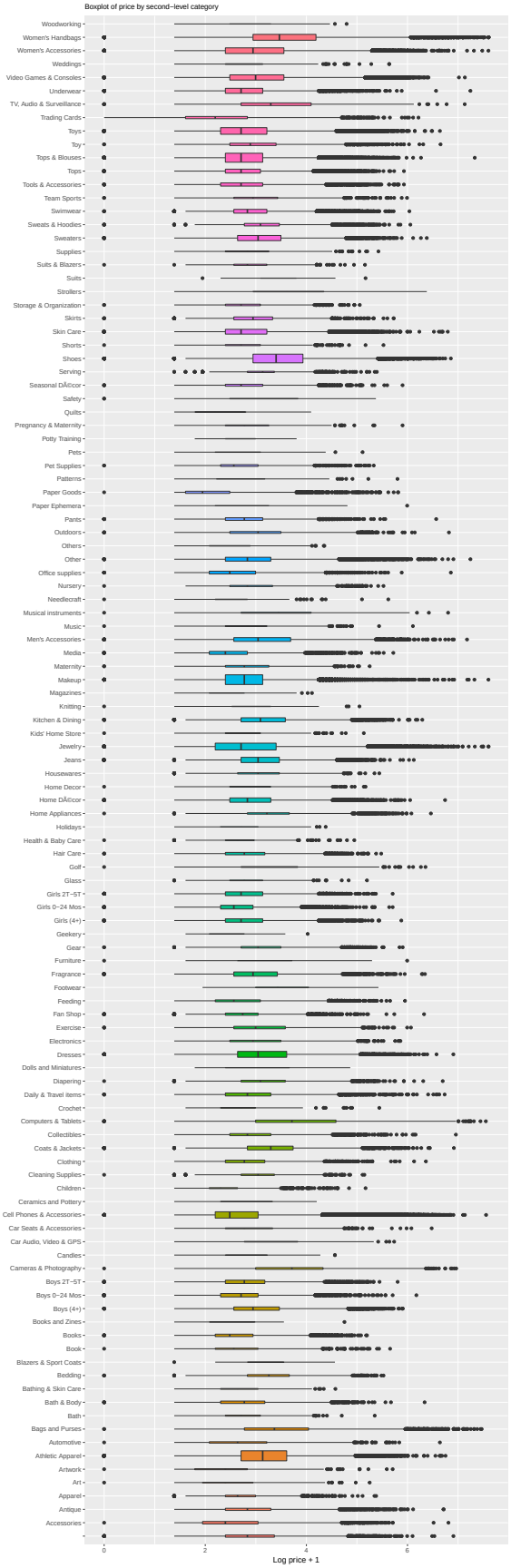
```
table(train$level_2_cat)[1:10]
```

```
##
##                       Accessories           Antique          Apparel
##              6327             8213              6093             2918
##               Art          Artwork Athletic Apparel       Automotive
##               656             1264            134383             2480
##  Bags and Purses             Bath
##              6338             1192
```

We have a lot more subcategories in level 2, let us see their prices.

```
train %>%
  ggplot(aes(x = level_2_cat, y = log(price+1), fill = level_2_cat)) +
  geom_boxplot(varwidth = TRUE) +
  coord_flip() +
```

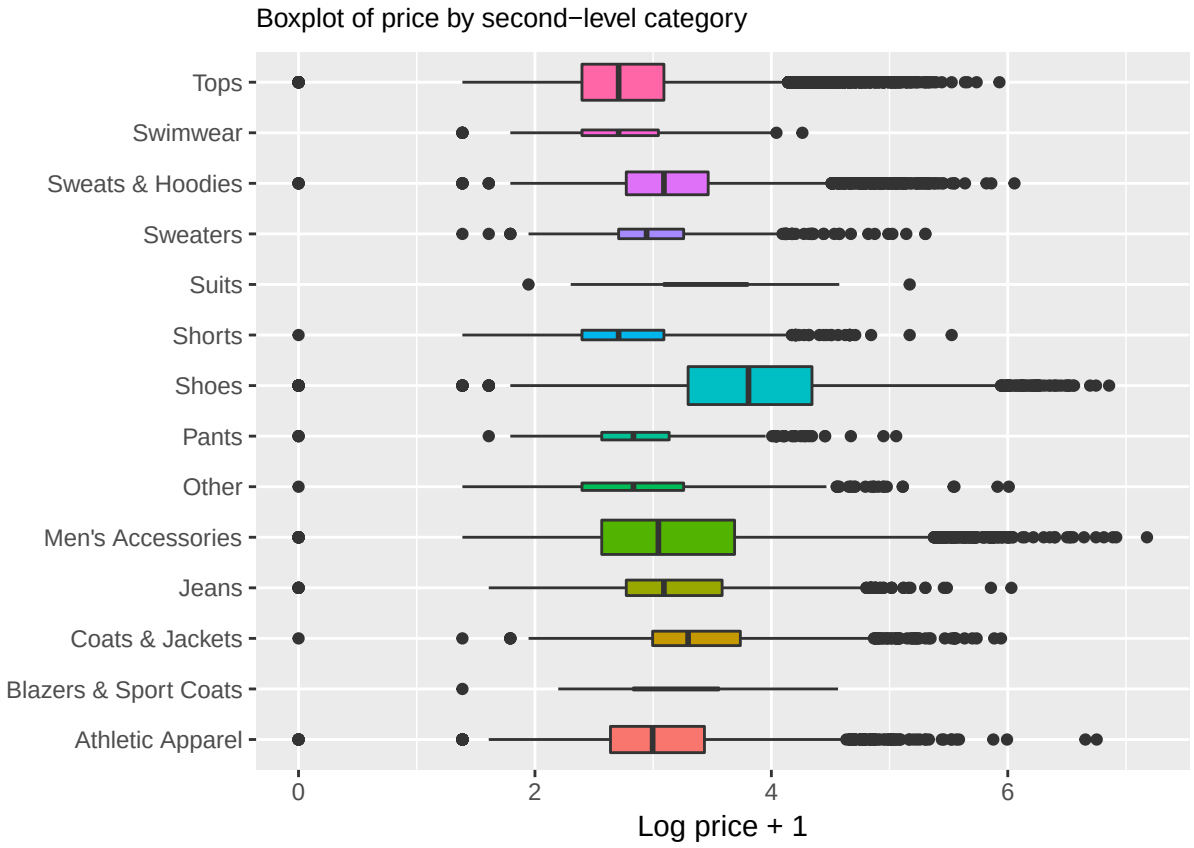
```
  labs(x = '', y = 'Log price + 1', title = 'Boxplot of price by second-level category') +
  theme(legend.position="none", plot.title = element_text(size=10))
```



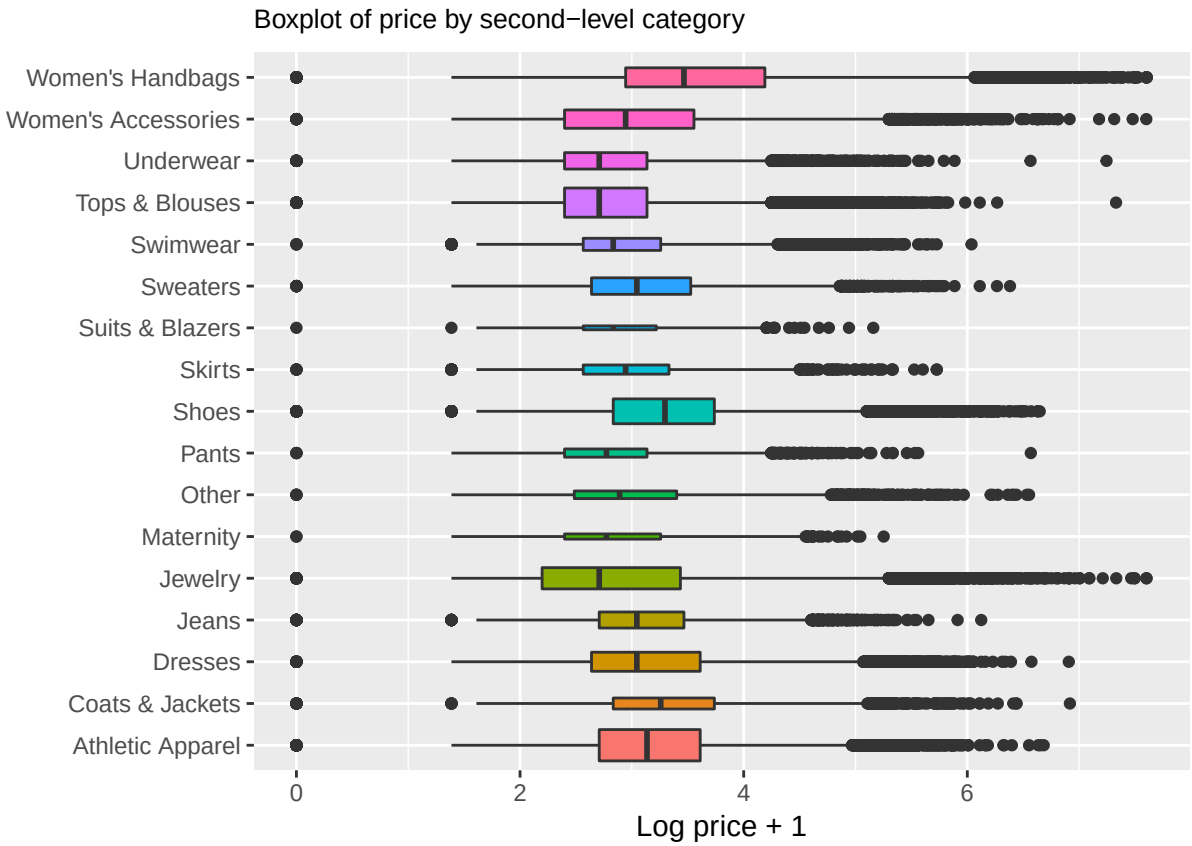
Still *Women* and *Kids* items are at the top. We can also

see second level category items distribution against *price* for specific first level, let us compare between *Men* and *Women* items

```
train %>%
  filter(level_1_cat =="Men") %>%
  ggplot(aes(x = level_2_cat, y = log(price+1), fill = level_2_cat)) +
  geom_boxplot(varwidth = TRUE) +
  coord_flip() +
  labs(x = '', y = 'Log price + 1', title = 'Boxplot of price by second-level category') +
  theme(legend.position="none", plot.title = element_text(size=10))
```




```
train %>%
  filter(level_1_cat =="Women") %>%
  ggplot(aes(x = level_2_cat, y = log(price+1), fill = level_2_cat)) +
  geom_boxplot(varwidth = TRUE) +
  coord_flip() +
  labs(x = '', y = 'Log price + 1', title = 'Boxplot of price by second-level category') +
  theme(legend.position="none", plot.title = element_text(size=10))
```



It seems Men's shoes and accessories are what *Men* items are higher than *Women* items. Otherwise, *Women* items are more pricy in general.

As a bonus visualization, let us see level 1 and level 2 interaction in different way

level 3

statistics of level3

```
length(levels(train$level_3_cat))
```

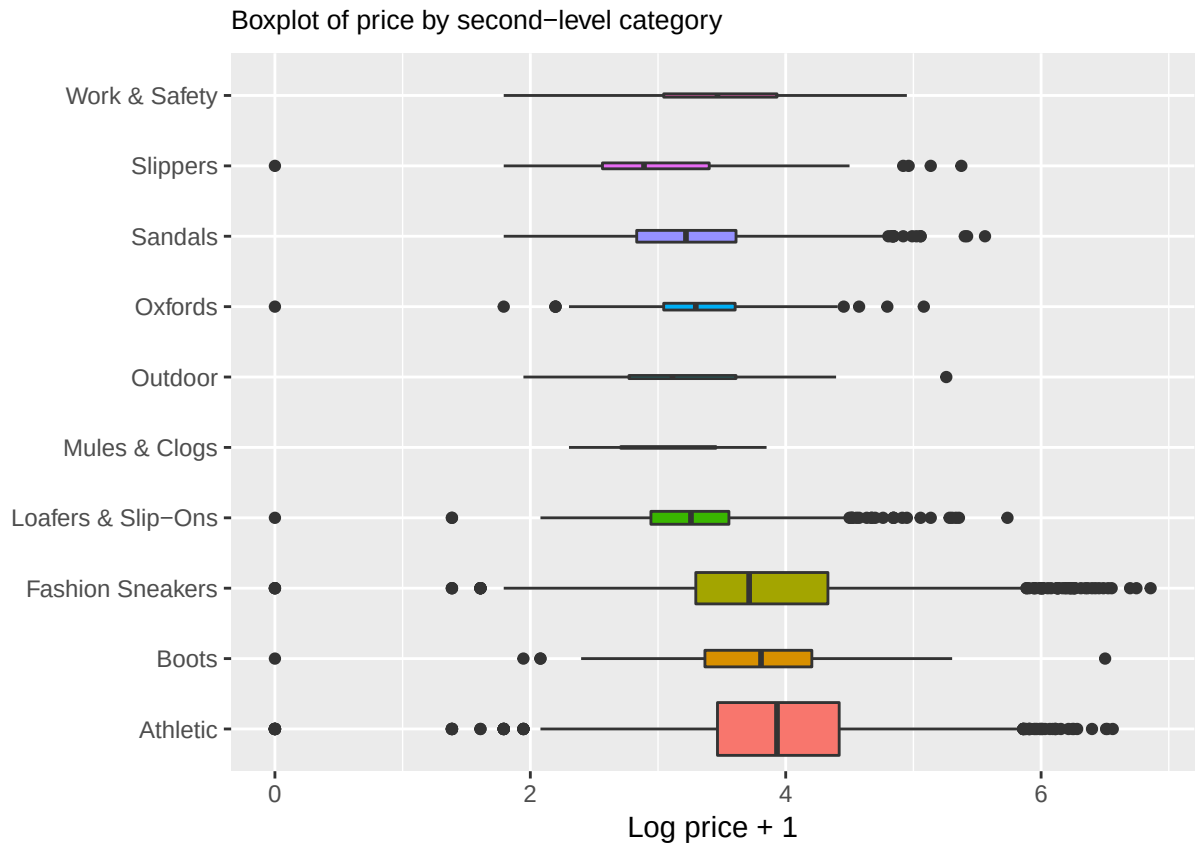
```
## [1] 0
```

```
table(train$level_3_cat)[1:10]
```

```
##
##                   100 Years or Older     50 To 75 Years
##                 6327                140                92
##    75 To 100 Years             A-Line   Above Knee, Mini
##                   42                756             20082
##        Accessories          Accessory               Aceo
##                10297                 31                 1
##      Action Figure
##                 5354
```

It will hard to visualize all these data at once, so let us seen what *brands* are pricy for *Men* and *Women* pricy items

```
train[level_1_cat =="Men" & level_2_cat == "Shoes"] %>%
  ggplot(aes(x = level_3_cat, y = log(price+1), fill = level_3_cat)) +
  geom_boxplot(varwidth = TRUE) +
  coord_flip() +
  labs(x = '', y = 'Log price + 1', title = 'Boxplot of price by second-level category') +
  theme(legend.position="none", plot.title = element_text(size=10))
```

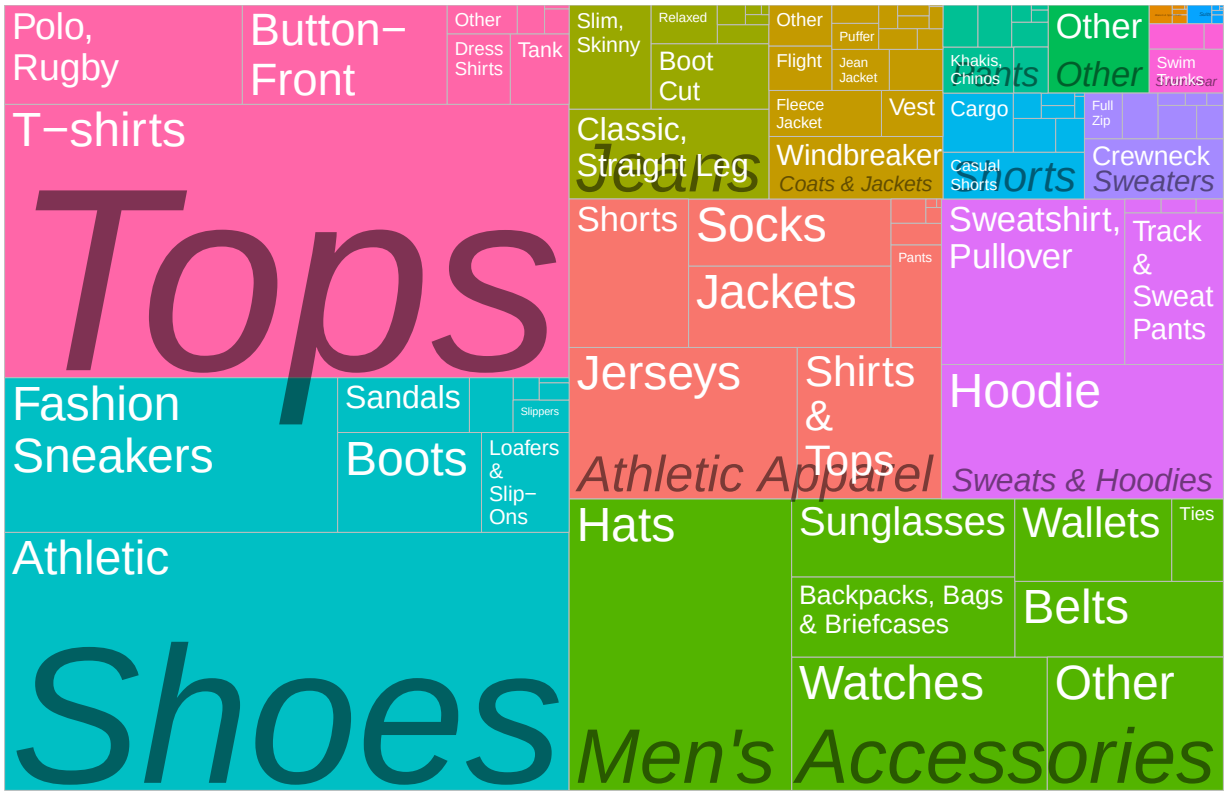


Ok, we have a lot of men are going to Gym here :).

Let us see the distribution in a different way

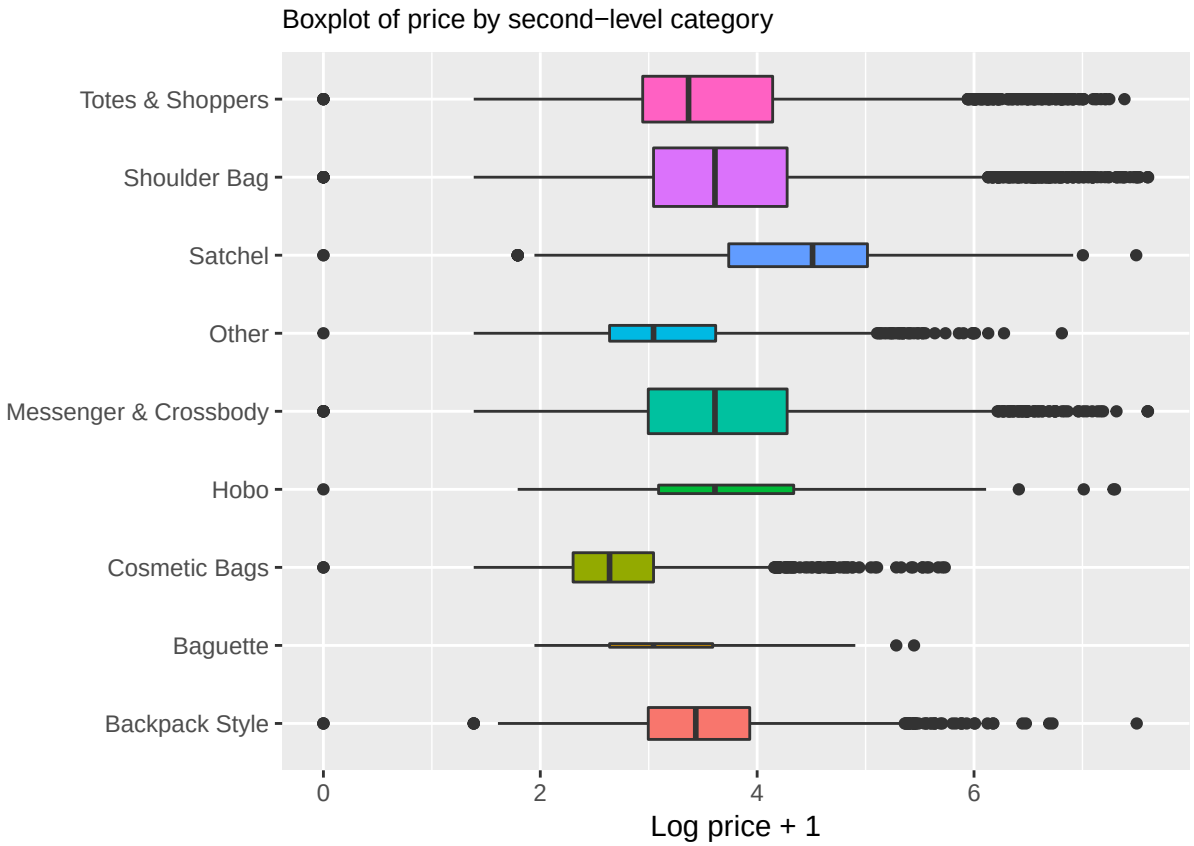
```
train[level_1_cat =="Men", .N, by = .(level_2_cat, level_3_cat)] %>%
  ggplot(aes(area=N, fill=level_2_cat, label=level_3_cat, subgroup=level_2_cat)) +
  geom_treemap() +
  geom_treemap_subgroup_text(grow = T, alpha = 0.5, colour =
                             "black", fontface = "italic", min.size = 0) +
  geom_treemap_text(colour = "white", place = "topleft", reflow = T) +
  theme(legend.position = "null") +
  ggtitle("2nd and 3rd Hierarchical Category Levels Under Men")
```

2nd and 3rd Hierarchical Category Levels Under Men



and for women

```
train[level_1_cat =="Women" & level_2_cat == "Women's Handbags"] %>%
  ggplot(aes(x = level_3_cat, y = log(price+1), fill = level_3_cat)) +
  geom_boxplot(varwidth = TRUE) +
  coord_flip() +
  labs(x = '', y = 'Log price + 1', title = 'Boxplot of price by second-level category') +
  theme(legend.position="none", plot.title = element_text(size=10))
```



Now, let us see the items distribution under women in different way

```
train[level_1_cat =="Women", .N, by = .(level_2_cat, level_3_cat)] %>%
  ggplot(aes(area=N, fill=level_2_cat, label=level_3_cat, subgroup=level_2_cat)) +
  geom_treemap() +
  geom_treemap_subgroup_text(grow = T, alpha = 0.5, colour =
                               "black", fontface = "italic", min.size = 0) +
  geom_treemap_text(colour = "white", place = "topleft", reflow = T) +
  theme(legend.position = "null") +
  ggtitle("2nd and 3rd Hierarchical Category Levels Under Women")
```

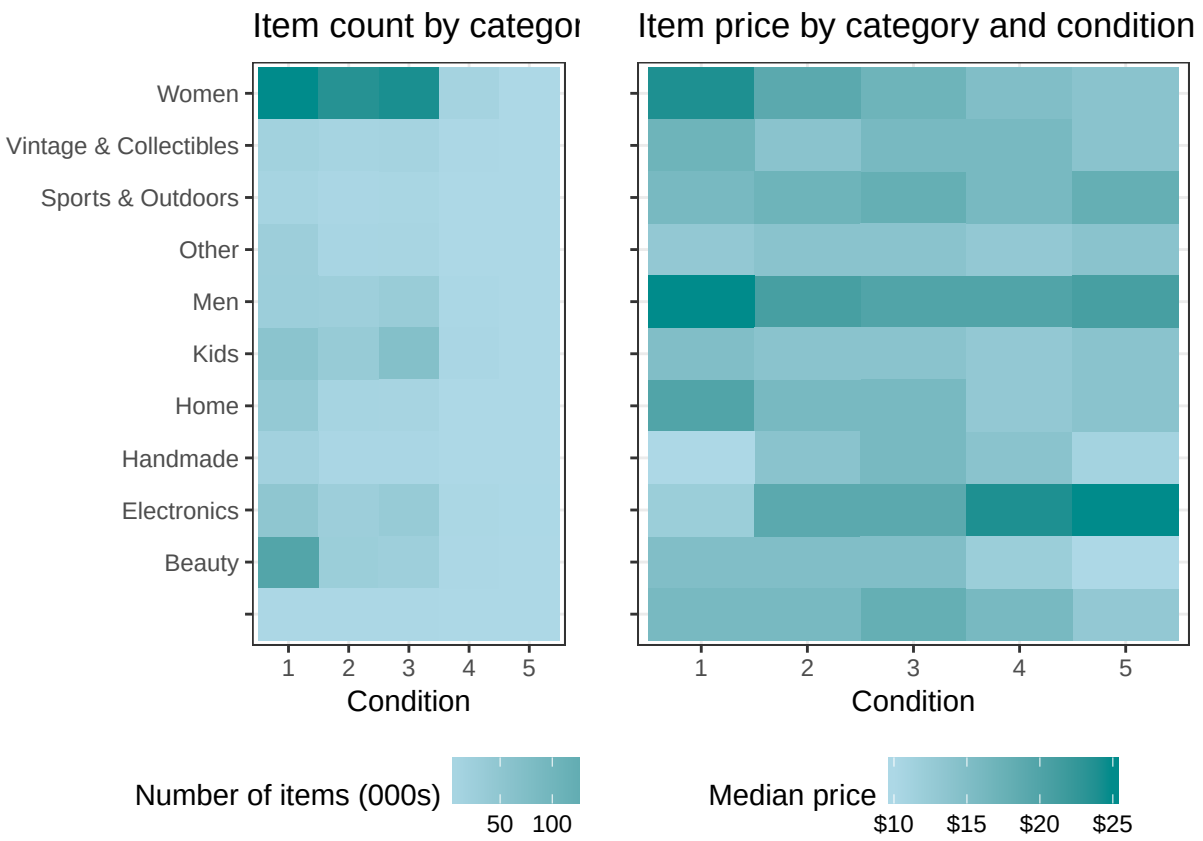
2nd and 3rd Hierarchical Category Levels Under Women

Relations with other items

with Item Condition

We can examine how item counts are distributed across top-level category and condition.

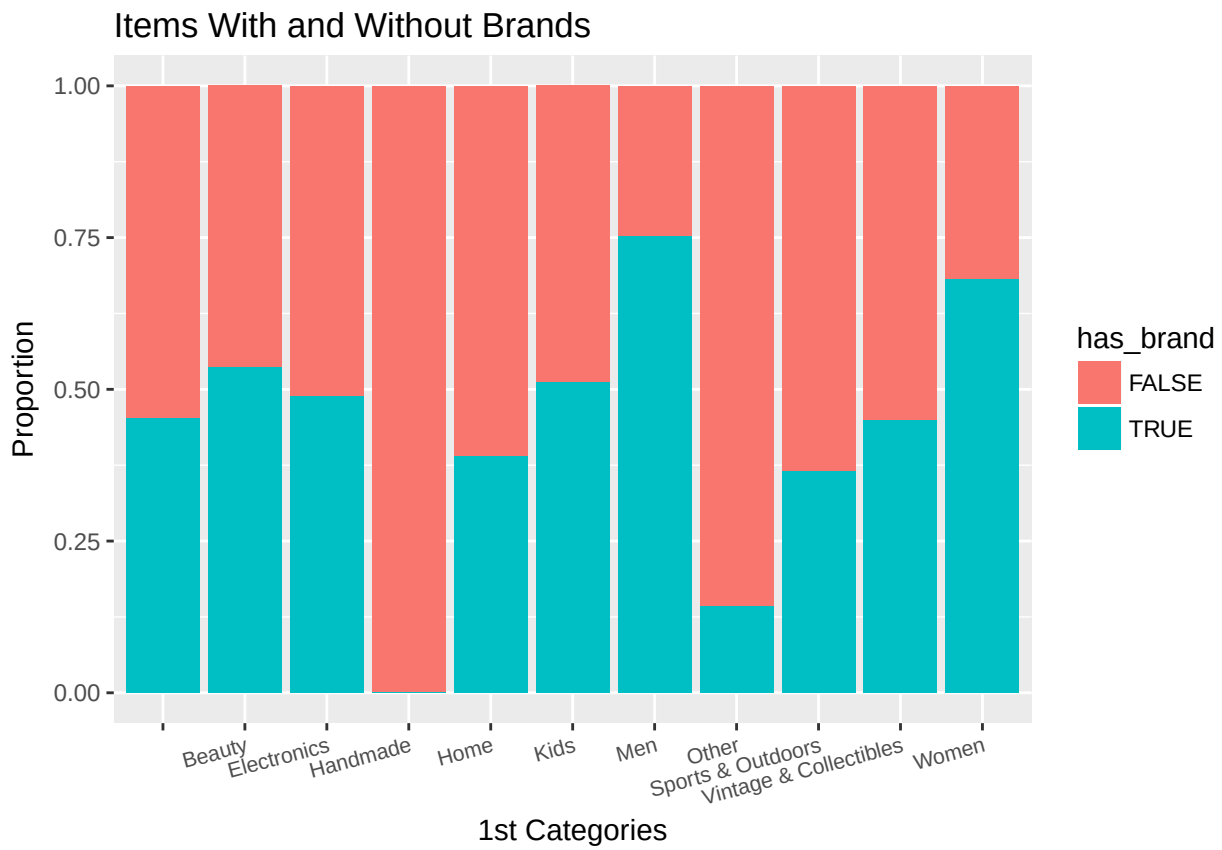
```
p1 <-
    train[, .N, by = c('level_1_cat', 'item_condition_id')] %>%
    ggplot(aes(x = item_condition_id, y = level_1_cat, fill = N/1000)) +
    geom_tile() +
    scale_fill_gradient(low = 'lightblue', high = 'cyan4') +
    labs(x = 'Condition', y = '', fill = 'Number of items (000s)', title = 'Item count by category and c
    theme_bw() +
    theme(legend.position = 'bottom')

p2 <-
    train[, .(median_price = median(price)), by = c('level_1_cat', 'item_condition_id')] %>%
    ggplot(aes(x = item_condition_id, y = level_1_cat, fill = median_price)) +
    geom_tile() +
    scale_fill_gradient(low = 'lightblue', high = 'cyan4', labels = dollar) +
    labs(x = 'Condition', y = '', fill = 'Median price', title = 'Item price by category and condition')
    theme_bw() +
    theme(legend.position = 'bottom', axis.text.y = element_blank())

grid.arrange(p1, p2, ncol = 2)
```



Women's items of condition 1,2, and 3 are the most numerous. This is followed by Beauty products. '

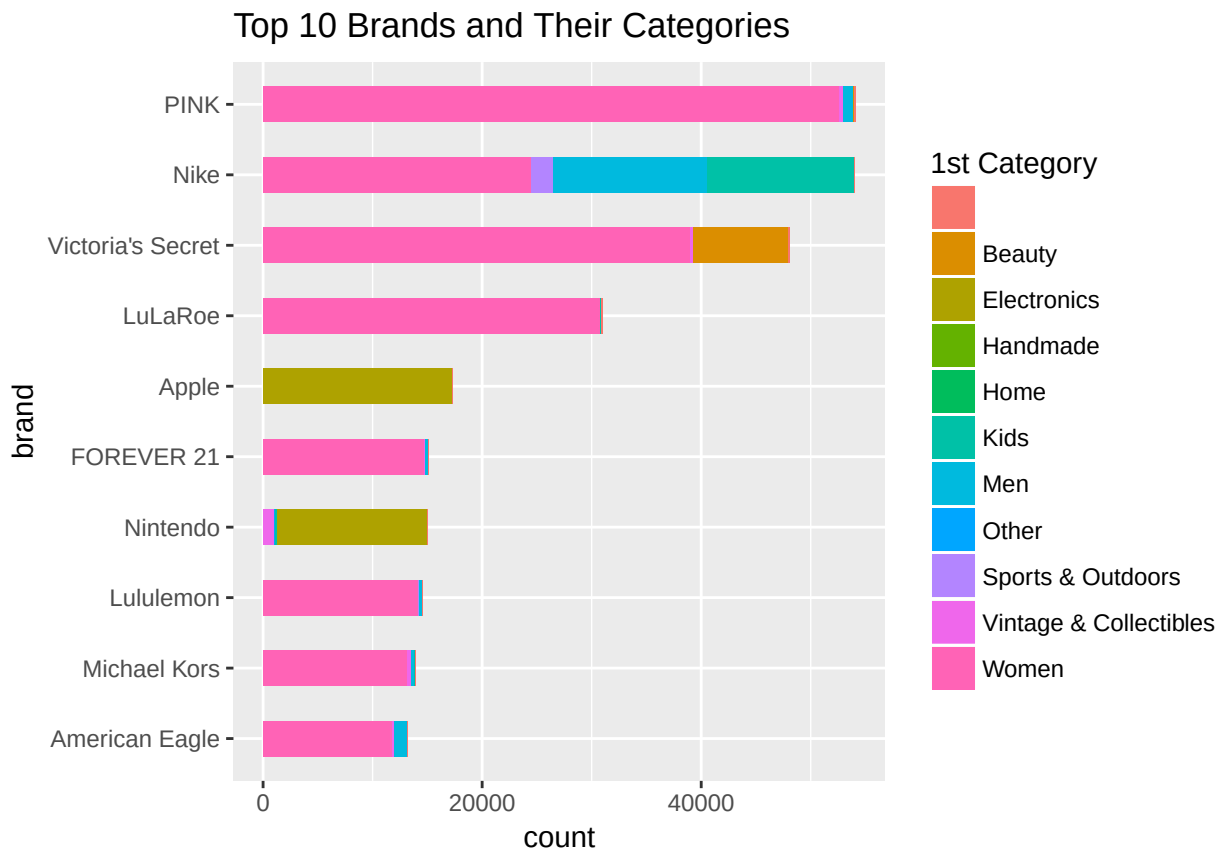
with Brands

```
train[, has_brand := (brand_name!='')] %>%
  ggplot(aes(x=level_1_cat, fill=has_brand)) +
  geom_bar(position='fill') +
  theme(axis.text.x=element_text(angle=15, hjust=1, size=8)) +
  xlab('1st Categories') +
  ylab('Proportion') +
  ggtitle('Items With and Without Brands')
```



top 10 brands

```
top10 <- train[brand_name !="", .N, by = .(bName = brand_name)][order(N, decreasing = T)][1:10]

train[brand_name %in% top10$bName] %>%
  ggplot(aes(x=factor(brand_name, levels=rev(top10$bName)), fill=level_1_cat)) +
  geom_bar(width=0.5) +
  coord_flip() +
  xlab('brand') +
  labs(fill='1st Category') +
  ggtitle('Top 10 Brands and Their Categories')
```



Ok. It seems textaul predictors have a much greater influnce over response variable than the numric ones. Therefore, we expect to get a lot from the last textual predictor *item description*.

Item Description

Before start working on our `corpus` objects, let us neturalize ***No description yet*** value from *item description* predictor to not affect our text analysis

```
train[item_description == 'No description yet', item_description := NA]

# create the corpus object from the item_description column
corpus <- corpus(train$item_description)

# check first few lines of summary frame
summary(corpus)[1:5, ]
```

```
## Corpus consisting of 1482535 documents, showing 100 documents:
##
##   Text Types Tokens Sentences
##  text1     1      1         0
##  text2    32     39         3
##  text3    26     32         2
##  text4    34     41         8
##  text5     5      5         1
##
## Source:  C:/Users/Nada Mamdouh/Documents/Compositions/Mercari-Price-Suggestion-Challenge/* on x86-64
```

```
## Created: Fri Jan 12 16:42:07 2018
## Notes:
```

Let us see some of these documents in action using `kwic` function and one of the phrases like *great condition*

```
kwic(corpus, phrase("great condition"), valuetype = "fixed") %>%
    head()
```

```
##
##     [text2, 5:6]   This keyboard is in | great condition |
##    [text13, 3:4]                   Xl, | great condition |
##    [text15, 7:8] . Suede fringe boots. | Great condition |
##    [text58, 1:2]                       | Great condition |
##   [text68, 20:21]       )- rarely worn, | great condition |
##    [text80, 1:2]                       | Great condition |
##
##  and works like it came
##
##  ! Size 7. If
##  ! No stains or tears
##  - no flaws- selling
##  sea wees size 0 brown
```

OK, that was some informatice samples to word in context.

N Gram 1

So, let us proceed to the next step and calculate DTM(*Doment-to-Term-Matrix*), we remove remove english stopwords and punctuation, and stem words. To find first the most top single words

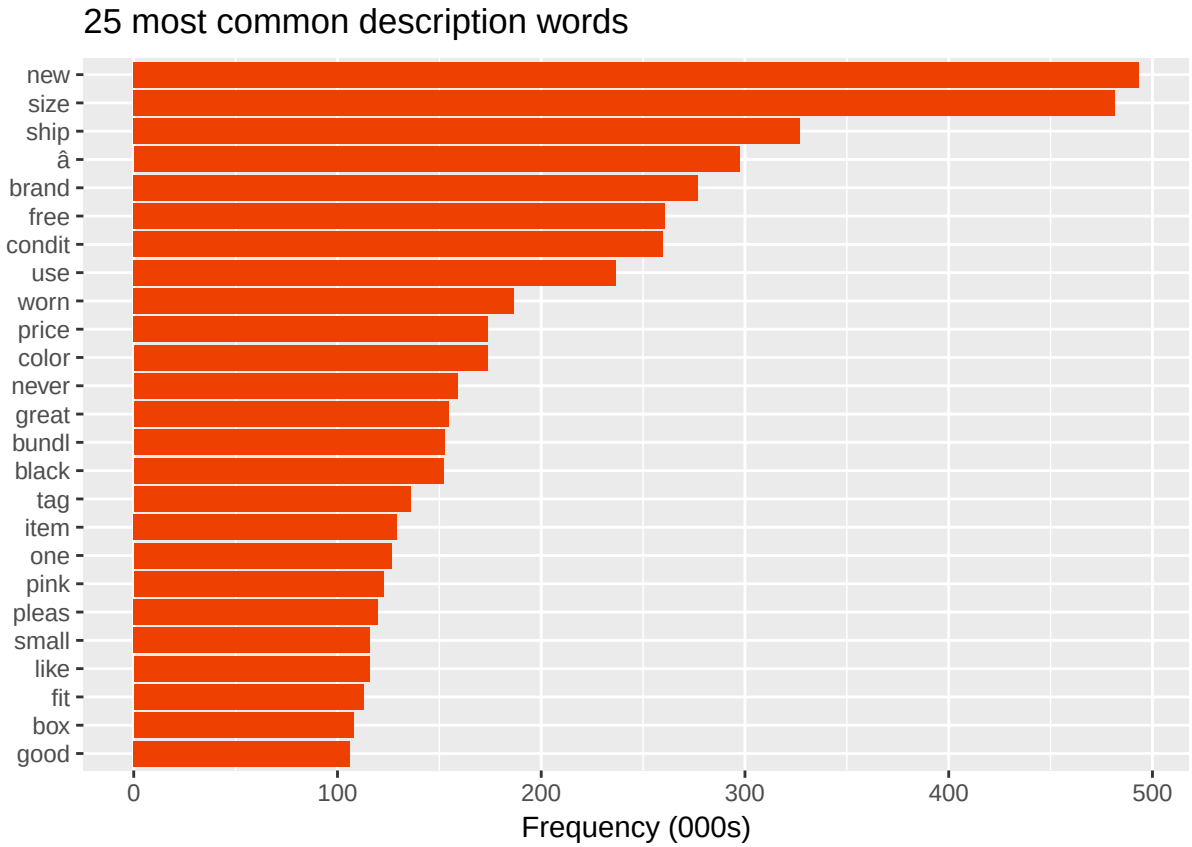
```
dfm <- dfm(
    corpus,
    ngrams = 1,
    remove = c("rm", stopwords("english")),
    remove_punct = TRUE,
    remove_numbers = TRUE,
    stem = TRUE)
```

Let us investigate the resultant document. First, let us get the 25 most common words

```
tf <- topfeatures(dfm, n = 25)
```

Now, let us visualize it

```
data.frame(term = names(tf), freq = unname(tf)) %>%
    ggplot(aes(x = reorder(term, freq), y = freq/1000)) +
    geom_bar(stat = 'identity', fill = 'orangered2') +
    labs(x = '', y = 'Frequency (000s)', title = '25 most common description words') +
    coord_flip()
```



It seems that there are care about the item condition, size, shipping and brand. Let us create a word cloud to see different point of view :)

```
textplot_wordcloud(dfm, min.freq = 3e4, random.order = FALSE,
                   rot.per = .25,
                   colors = RColorBrewer::brewer.pal(8,"Dark2"))
```

Ok, it is a confirmation.

N Gram 2

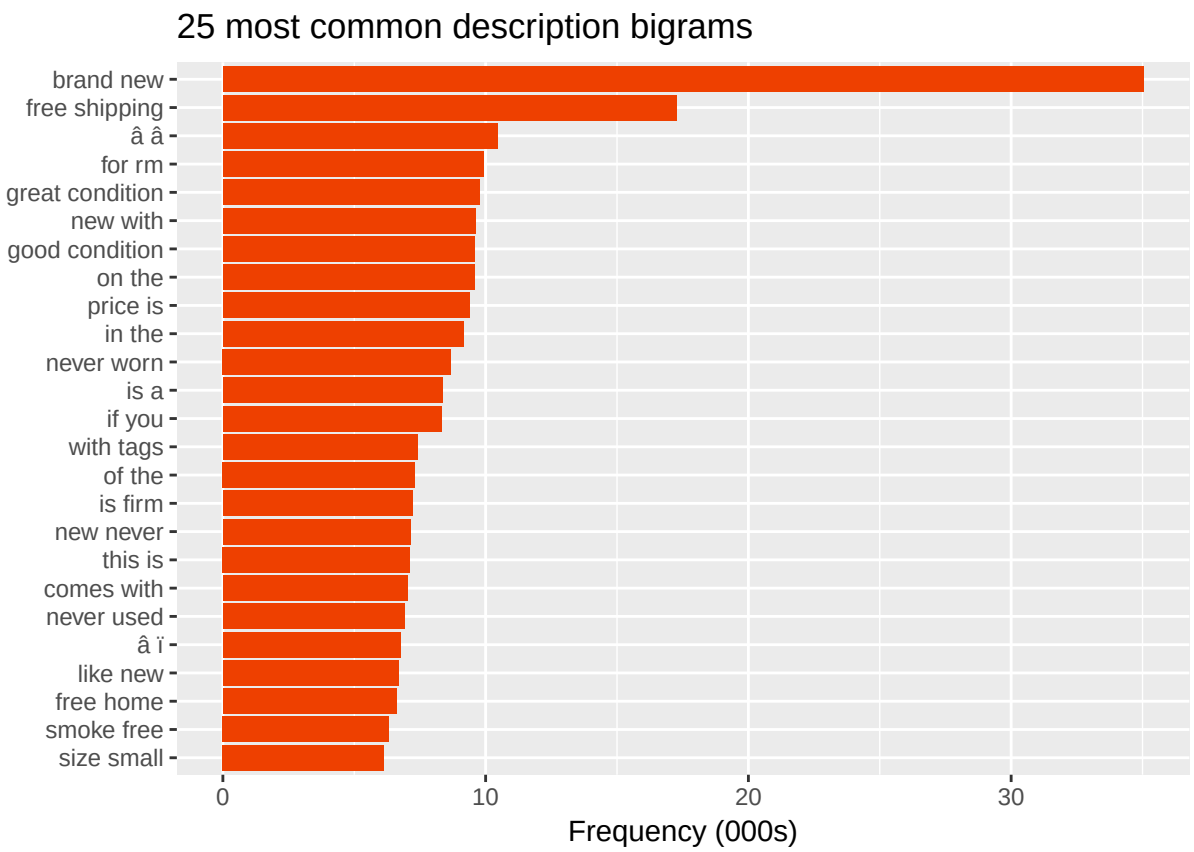
What about the most popular 2 words

```
dfm2 <- corpus %>%
   #to reduce the amound of data for performance issue
    corpus_sample(size = floor(ndoc(corpus) * 0.15)) %>%
    dfm(
        ngrams = 2,
        ignoredFeatures = c("rm", stopwords("english")),
        remove_punct = TRUE,
        remove_numbers = TRUE,
        concatenator = " "
    )
```

```
## Warning: Argument ignoredFeatures not used.
```

```
# get 25 most common bigrams
tf <- topfeatures(dfm2, n = 25)

# convert to df and plot
data.frame(term = names(tf), freq = unname(tf)) %>%
    ggplot(aes(x = reorder(term, freq), y = freq/1000)) +
    geom_bar(stat = 'identity', fill = 'orangered2') +
    labs(x = '', y = 'Frequency (000s)',
```

```
        title = '25 most common description bigrams') +
    coord_flip()
```



The results seem simirlar to the single phrase results. Let us confirm this by word cloud

```
textplot_wordcloud(dfm2, min.freq = 2000, random.order = FALSE,
                   rot.per = .25,
                   colors = RColorBrewer::brewer.pal(8,"Dark2"))
```

It is confirmed.

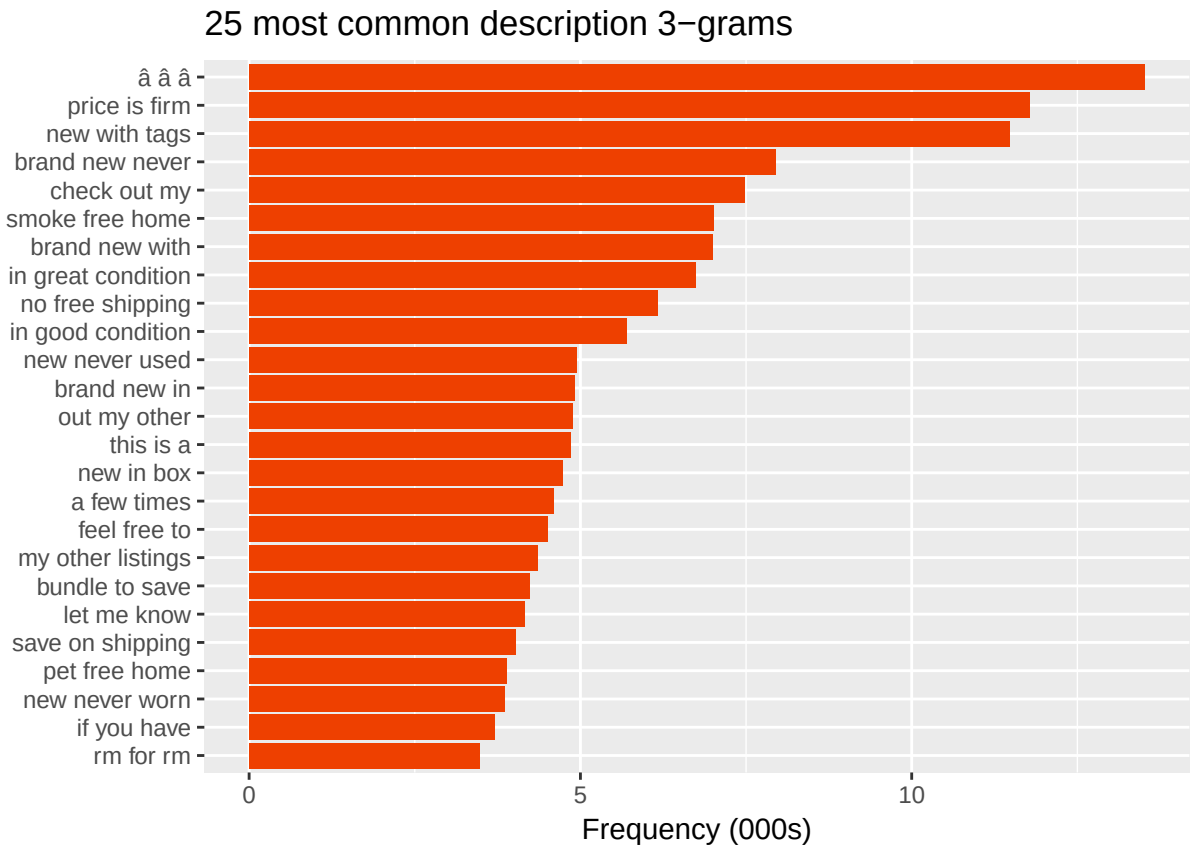
N Grams 3

What about 3 phrases

```
dfm3 <- corpus %>%
    corpus_sample(size = floor(ndoc(corpus) * 0.25)) %>%
    dfm(
        ngrams = 3,
        ignoredFeatures = c("rm", stopwords("english")),
        remove_punct = TRUE,
        remove_numbers = TRUE,
        concatenator = " "
    )
```

```
## Warning: Argument ignoredFeatures not used.
```

```
# get 25 most common trigrams
tf <- topfeatures(dfm3, n = 25)

# convert to df and plot
data.frame(term = names(tf), freq = unname(tf)) %>%
    ggplot(aes(x = reorder(term, freq), y = freq/1000)) +
    geom_bar(stat = 'identity', fill = 'orangered2') +
    labs(x = '', y = 'Frequency (000s)', title = '25 most common description 3-grams') +
    coord_flip()
```



Ok, now we knew what *free* word is used for :D. Now, let us confirm this by a word cloud.

```
#textplot_wordcloud(dfm3, min.freq = 1000, random.order = FALSE,
 #                  rot.per = .50,
  #                 colors = RColorBrewer::brewer.pal(8,"Dark2"))
```

Let us see the *level 1 category* in the documents

```
docvars(corpus, "level_1_cat") <- train$level_1_cat
p1 <- summary(corpus) %>%
  ggplot(aes(x = level_1_cat, y = Tokens)) +
  geom_boxplot(aes(fill = level_1_cat), color = 'grey') +
  coord_flip() +
  theme(legend.position = 'bottom') +
  labs(x = '', y = 'Number of tokens in description')

p2 <- summary(corpus) %>%
  ggplot(aes(x = Tokens)) +
  geom_density(aes(fill = level_1_cat), color = 'grey') +
  facet_wrap(~level_1_cat) +
  scale_y_continuous(limits = c(0,0.05)) +
  theme(legend.position = "none") +
  labs(x = 'Number of tokens in description')

grid.arrange(p1, p2, ncol = 2)
```

OK, altough men items are the highest but there are no much description for these items, it seems predicting these items will not be so much easy.

Names

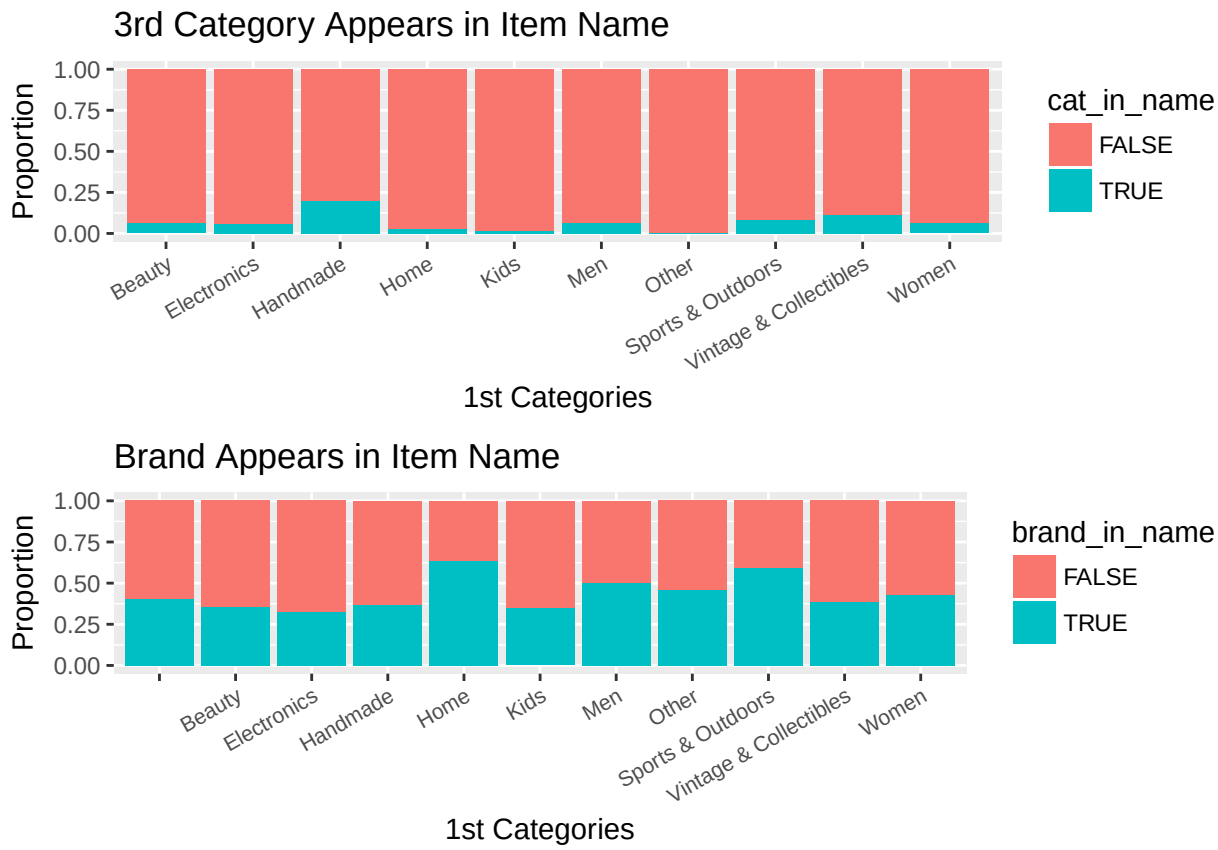
```
length(unique(train$name))
```

```
## [1] 1225273
```

Names and brands and conditions

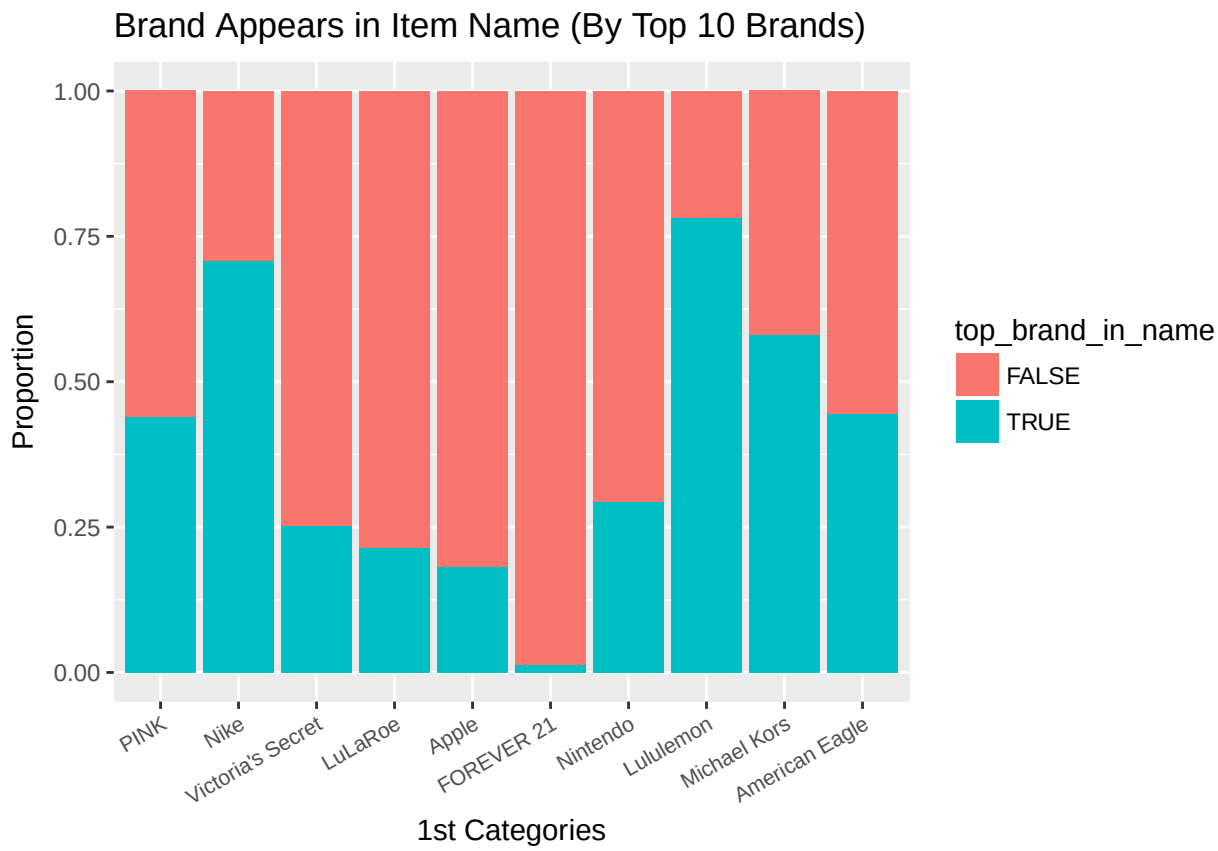
```
p1 <- train[level_1_cat != ""][, cat_in_name := (str_detect(name, level_3_cat))] %>%
  ggplot(aes(x=level_1_cat, fill=cat_in_name)) +
  geom_bar(position='fill') +
  theme(axis.text.x=element_text(angle=30, hjust=1, size=8)) +
  xlab('1st Categories') +
  ylab('Proportion') +
  ggtitle('3rd Category Appears in Item Name')

p2 <- train[train$has_brand][,brand_in_name := (str_detect(name, brand_name))] %>%
  ggplot(aes(x=level_1_cat, fill=brand_in_name)) +
  geom_bar(position='fill') +
  theme(axis.text.x=element_text(angle=30, hjust=1, size=8)) +
  xlab('1st Categories') +
  ylab('Proportion') +
  ggtitle('Brand Appears in Item Name')
```

```
grid.arrange(p1, p2, ncol=1)
```



top10

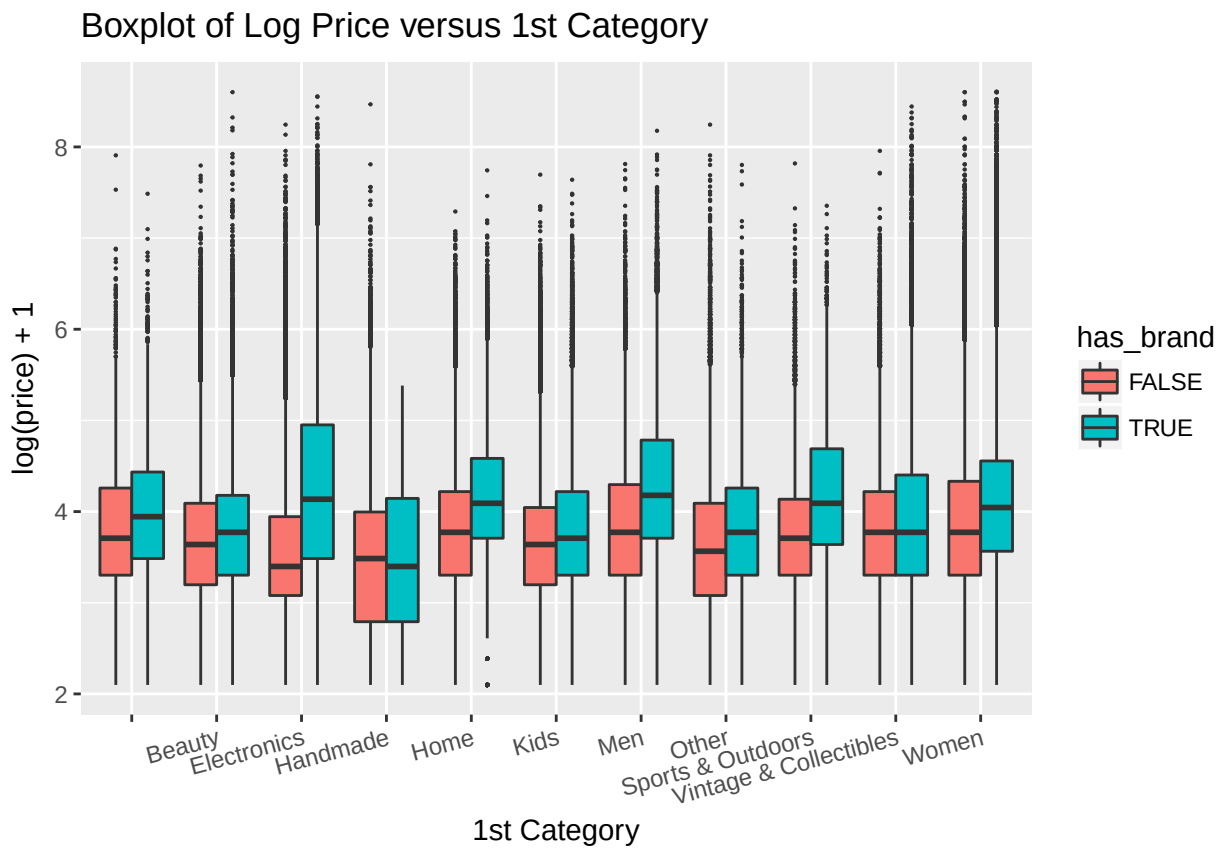
```
train[brand_name %in% top10$bName][, top_brand_in_name := (str_detect(name, brand_name))] %>%
  ggplot(aes(x=factor(brand_name, levels=top10$bName), fill=top_brand_in_name)) +
  geom_bar(position='fill') +
  theme(axis.text.x=element_text(angle=30, hjust=1, size=8)) +
  xlab('1st Categories') +
  ylab('Proportion') +
  ggtitle('Brand Appears in Item Name (By Top 10 Brands)')
```



brand vs price vs categ

```
train %>%
ggplot(aes(x=level_1_cat, y=log(price) + 1, fill=has_brand)) +
geom_boxplot(outlier.size=0.1) +
ggtitle('Boxplot of Log Price versus 1st Category') +
xlab('1st Category') +
theme(axis.text.x=element_text(angle=15, hjust=1))
```

```
## Warning: Removed 874 rows containing non-finite values (stat_boxplot).
```



Model Building

Conclusion

Credit

- Troy Walters
- Lingzhi

eBay acronyms

A-E B&W: black and white

BC: back cover (usually used as a description for books)

BIN: Buy It Now

CIP: customer initiated payment

DOA: dead on arrival (an item that doesn't work or is broken when it's received)

DSR: detailed seller rating (additional Feedback ratings buyers can give sellers)

EST: Eastern Standard Time

EUC: excellent used condition

F-I FAQ: frequently asked questions (a list of questions with answers.)

FB: Feedback

FC: fine condition

FOB: freight on board (usually means something has shipped)

FS: full screen (usually applied to a DVD or video format)

FVF: final value fee

G: good condition

GBP: Great Britain pounds

GU: gently used (item that has been used but shows little wear, accompanied by explanation of wear)

HP: home page

HTF: hard to find

HTML: hypertext markup language (the language used to create web pages)

IE: Internet Explorer

IM: instant messaging

INIT: initials

ISP: Internet service provider (a company that gives you access to the Internet)

J-M JPG: JPEG (preferred file format for pictures on eBay, pronounced "jay-peg")

LTBX: letterbox (video format that recreates a widescreen image)

LTD: limited edition

MNT: mint or in perfect condition (a subjective term that doesn't necessarily mean new)

MIB: mint in box

MIJ: made in Japan

MIMB: mint in mint box

MIMP: mint in mint package

MIP: mint in package

MNB: mint no box

MOC: mint on card

MOMC: mint on mint card

MONMC: mint on near mint card

MWBT: mint with both tags

MWMT: mint with mint tags

N-P NARU: not a registered user (also a suspended user)

NBW: never been worn

NC: no cover

NIB: new in box

NM: near mint

NOS: new old stock

NR: no reserve price (for an auction-style listing)

NRFB: never removed from box

NWT: new with tags

NWOB: new without box

NWOT: new without original tags

OEM: original equipment manufacturer

OOP: out of print

PST: Pacific Standard Time

Q-Z RET: retired

SCR: scratch

S/O: sold out

Sig: signature