Loading the data.

The data is in json format,so the first step inolves in converting the .json to pandas dataframe.

In [111]:

```
import numpy as np
import pandas as pd
import json
import gzip

#Converting into panda's dataframe
yesterday=pd.read_json('C:/Users/Srikanth/Desktop/DatraWeave/yesterday.json.gz_out',lines='true')
yesterday.shape   # It can be observed that there are almost 1.42 lakhs rows(records)
```

Out[111]:

```
(141699, 9)
```

In [112]:

```
Id=range(141699)
yesterday['Id']=Id
yesterday.head()
# Adding an additional column "Id" for my convenience.
```

Out[112]:

	available_price	category	crawl_date	http_status	mrp	pack_size	subcategory	title	
0	5.99	Household	20180726	200	0.0	30 ct	Trash Bags & Liners	Seventh Generation 13 Gallon Extra Strong Flap...	1ef0f41ebcee030a354404(
1	5.49	Household	20180726	200	0.0	25 ct	Trash Bags & Liners	BioBag Food Scrap Bags Small 3 Gallon - 25 CT ...	c566570abdbe95b5e6a78e
2	5.69	Household	20180726	200	0.0	30.0 ct	Trash Bags & Liners	If You Care Certified Compostable Food Waste B...	3b22b28b351dec75664b17
3	5.99	Household	20180726	200	0.0	12 ct	Trash Bags & Liners	BioBag Food Scrap Bags Tall Kitchen 13 Gallon ...	7ff787f49e137237054f95e7
4	5.99	Household	20180726	200	0.0	20 ct	Trash Bags & Liners	Seventh Generation 13 Gal with Drawstring Clos...	a7ad16e71257855c033aac

In [113]:

```
today=pd.read_json('C:/Users/Srikanth/Desktop/DatraWeave/today.json.gz_out',lines='true')
today.shape # Alomst 1.423 lakhs records, a bit more than yesterday.
```

Out[113]:

```
(142291, 9)
```

In [114]:

```
Id=range(142291)
today['Id']=Id
today.head()
```

Out[114]:

	available_price	category	crawl_date	http_status	mrp	pack_size	subcategory	title	
0	5.99	Household	20180727	200	0.0	30 ct	Trash Bags & Liners	Seventh Generation 13 Gallon Extra Strong Flap...	1ef0f41ebcee030a354404(
1	5.49	Household	20180727	200	0.0	25 ct	Trash Bags & Liners	BioBag Food Scrap Bags Small 3 Gallon - 25 CT ...	c566570abdbe95b5e6a78e
2	5.69	Household	20180727	200	0.0	30.0 ct	Trash Bags & Liners	If You Care Certified Compostable Food Waste B...	3b22b28b351dec75664b17
3	5.99	Household	20180727	200	0.0	12 ct	Trash Bags & Liners	BioBag Food Scrap Bags Tall Kitchen 13 Gallon ...	7ff787f49e137237054f95e7
4	5.99	Household	20180727	200	0.0	20 ct	Trash Bags & Liners	Seventh Generation 13 Gal with Drawstring Clos...	a7ad16e71257855c033aad

In [115]:

```
a=today
b=yesterday
#Assigning the original data(yesterday and today) to "a" and "b" since it is not wise to work on o
riginal data.
```

1.Number of Urls which are overlapping.

By the term overlapping I assume that, the no. of rows with urls which are common in both dataframes.

In [6]:

```
intersection=list(set(a['urlh']).intersection(set(b['urlh']))) # All the urlhs in both today and
yesterday's data.
#Here i did set intersection and converted it into list and found the length.
intersection[0:4]
#These are the urlhs which are present in both dataframes.
```

Out[6]:

```
['323a8894106bad1f43402d0f6e0aa5feb49635d4',
 '503020b52b0ffa31e593a127e359617392841b95',
 '8a256673e44ebdac0d6c6b5d702118f0313ce91c',
 'df6a05434334b6017bca2ba14ed96e2af1b71e35']
```

In [7]:

```
len(intersection)
# 129673 records have overlapping
```

Out[7]:

```
129673
```

In [8]:

```
overlap_today=[dict(a.loc[i]) for i in range(len(a)) if a['urlh'][i] in intersection]
#Here we find the records which are having urlhs in intersection.
```

In [9]:

```
len(overlap_today)
#Out of 142291 records in today, 140141 records have urlhs which are present in yesterday.
```

Out[9]:

```
140414
```

In [10]:

```
overlap_yesterday=[dict(b.ix[i]) for i in range(len(b)) if b['urlh'][i] in intersection]
#Out of 141699, 140413 records have urlhs intersection
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: DeprecationWarning:
.ix is deprecated. Please use
.loc for label based indexing or
.iloc for positional indexing

See the documentation here:
http://pandas.pydata.org/pandas-docs/stable/indexing.html#ix-indexer-is-deprecated
  """Entry point for launching an IPython kernel.
```

In [11]:

```
len(overlap_yesterday)
```

Out[11]:

```
140413
```

In [12]:

```
overlap_yesterday[1]
#It is a list of dicts, so 1st index of this list is same as the first row of dataframe.
```

Out[12]:

```
{'available_price': 5.49,
 'category': 'Household',
 'crawl_date': 20180726,
 'http_status': 200,
 'mrp': 0.0,
 'pack_size': '25 ct',
 'subcategory': 'Trash Bags & Liners',
 'title': 'BioBag Food Scrap Bags Small 3 Gallon - 25 CT - 25 ct',
 'urlh': 'c566570abdbe95b5e6a78eb8b3b2424108bf36a2',
 'Id': 1}
```

In [13]:

```
overlap_today1=pd.DataFrame.from_records(overlap_today)
#Here i converted the list of dicts to dataframe for easy understanding.
```

2. Calculate price difference.

In [398]:

```
yesterday_dup=[]
for i in range(len(overlap_yesterday)):
    if overlap_yesterday[i]['urlh']==overlap_yesterday[i]['urlh'] and overlap_yesterday[i]['http_st
atus']==200:
        yesterday_dup.append(overlap_yesterday[i])
# In 3rd line, all the urls with http=200 are taken and appended in the list named yesterday_dup.
#This list will contain duplicate urls. So in next step we will drop duplicates.
```

In [399]:

```
len(yesterday_dup)
# It can be observed that therse are only 8 duplicate urls in yesterday's which has http-value oth
er than 200.
# But point to be kept in mind is that there could be lot of duplicate urls with http value=200.
```

Out[399]:

```
140405
```

In [400]:

```
overlap_yesterday1=pd.DataFrame.from_records(yesterday_dup)
#Here i converted the list of dicts to dataframe for easy understanding.
```

In [401]:

```
overlap_dup1=overlap_yesterday1.drop_duplicates(subset={"urlh"}, keep='first', inplace=False)
overlap_dup1.shape
#here it can be observed that after dropping duplicate urlhs with http=200, the yesterday's record
s
# have dropped to 129673
```

Out[401]:

```
(129673, 10)
```

In [402]:

```
today_dup=[]
for i in range(len(overlap_today)):
    if overlap_today[i]['urlh']==overlap_today[i]['urlh'] and overlap_today[i]['http_status']==200:
        today_dup.append(overlap_today[i])
```

In [403]:

```
len(today_dup)
```

Out[403]:

```
140407
```

In [404]:

```
overlap_today1=pd.DataFrame.from_records(today_dup)
#Here i converted the list of dicts to dataframe for easy understanding.
```

In [405]:

```
overlap_dup2=overlap_today1.drop_duplicates(subset={"urlh"}, keep='first', inplace=False)
overlap_dup2.shape
#here it can be observed that after dropping duplicate urlhs with http=200, the yesterday's record
s
# have dropped to 129673
```

Out[405]:

```
(129673, 10)
```

In [406]:

```
price_difference=overlap_dup2['available_price']-overlap_dup1['available_price']
```

In [407]:

```
price_difference[::-1]
#Here i printed the list in reverse order since price difference is observed in last records not i
n the staring records.
```

Out[407]:

```
140404      NaN
140403      NaN
140402     2.01
140401     0.00
140400    20.99
140399     0.00
140398   -30.00
140397    -1.00
140396    -1.00
140395     2.00
140394     1.00
140393    -2.00
140392     6.00
140391     2.00
140390     4.00
140389     5.20
140388     6.00
140387    -6.20
140386   -10.00
140385     7.50
140384     2.50
140383    -6.50
140382     0.50
140381     3.00
140380    -1.00
140379    -0.50
140378     5.00
140377     0.51
140376   -13.00
140375     1.99
          ...
29         0.00
28         0.00
27         0.00
26         0.00
25         0.00
24         0.00
23         0.00
22         0.00
21         0.00
20         0.00
19         0.00
18         0.00
17         0.00
16         0.00
15         0.00
14         0.00
13         0.00
12         0.00
11         0.00
10         0.00
9          0.00
8          0.00
7          0.00
6          0.00
5          0.00
4          0.00
3          0.00
2          0.00
1          0.00
0          0.00
Name: available price, Length: 132698, dtype: float64
```

In [241]:

```
def diff(s):
"""Here we are gettin the index where the available_price is not same"""
    index=[]
    for i in s.index:
        if s[i]!=0:
            index.append(i)
    return(index)
aa=diff(price_difference)
#Here we are getting the indexes where the available_price is not zero(prices are different)
```

In [244]:

```
len(aa)
```

Out[244]:

82918

3. Unique Categories in both files.

In [27]:

```
count1=[]
unique1=today['category'].unique()
for i in unique1:
    count1.append(i)
len(count1)
#There are 50 unique categories in today dataframe
```

Out[27]:

50

In [28]:

```
count_tday_unique=today.category.unique()

#These are the unique values, if we count_tday_unique.size...we can get the size too but still I w
rote/
#the code to get the unique values in the above cell.

count_tday_unique=list(count_tday_unique)# Conver the array to list.
```

In [410]:

```
count2=[]
unique2=yesterday['category'].unique()
for i in unique2:
    count2.append(i)
len(count2)
#There are 50 unique categories in today dataframe
```

Out[410]:

50

4. List of Categories not overlapping

In [411]:

```
%%time
not_overlap=[]
i=0
while i< len(b): #Here we chose lenght of b cuz records in yesterday is lesser than today.
    if a['category'][i] != b['category'][i]:
```

```
        not_overlap.append(a['category'][i])
    i+=1
```

```
Wall time: 2.68 s
```

In [412]:

```
len(not_overlap)
```

Out[412]:

```
19577
```

5. Generate the stats with count for all taxonomies.

In [371]:

```
#group = a.groupby('category')

#df2 = group.apply(lambda x: x['subcategory'])
#Here we will get a dataframe where we will get All the 50 Categories and their corresponding Subc
ategories.
```

In [18]:

```
#today_tex=[]
#for i in count_tday_unique[]:
#    if a['category'][i]==i:
 #       today_tex.append(a['subcategory'][i])
```

In [16]:

```
df_tax=a[['category', 'subcategory']]
```

In [25]:

```
df_tax[:10]
```

Out[25]:

	category	subcategory
0	Household	Trash Bags & Liners
1	Household	Trash Bags & Liners
2	Household	Trash Bags & Liners
3	Household	Trash Bags & Liners
4	Household	Trash Bags & Liners
5	Household	Paper Goods
6	Household	Paper Goods
7	Household	Paper Goods
8	Household	Paper Goods
9	Household	Paper Goods

In [123]:

```
tax_dict={k: v["category"].tolist() for k,v in df_tax.groupby("subcategory")}
tax_dict['Silver Rum'][:6]
#Here in this cell we get dicts where the unique categories will be the keys
#and the corresponding subcategories will be inthe list as values to the key.
```

Out[123]:

```
['Rum', 'Rum', 'Rum', 'Rum', 'Rum', 'Rum']
```

In [84]:

```
for myKey in tax_dict.keys():
    print(myKey,'<', Counter(list(tax_dict[myKey])))

    ##Here we get the key '<' Subcategory: count
```

```
Adult Care < Counter({'Personal Care': 67})
Aged Rum < Counter({'Rum': 70})
Air Fresheners & Candles < Counter({'Household': 809, 'Personal Care': 94})
Albarino < Counter({'White Wine': 10})
Allergy < Counter({'Allergy & Cold Essentials': 52})
Amber/Red Ale < Counter({'Ale Beer': 63})
Amber/Red Lager < Counter({'Lager Beer': 18})
American All-Malt Lager < Counter({'Lager Beer': 16})
American Wild Ale < Counter({'Specialty Style Beer': 8})
American-Style Lager < Counter({'Lager Beer': 110})
Anejo < Counter({'Tequila': 57})
Aperitif < Counter({'Dessert & Fortified Wine': 25})
Apple Cider < Counter({'Cider': 118})
Armagnac < Counter({'Brandy & Cognac': 7})
Aromatherapy < Counter({'Personal Care': 34})
Asian Foods < Counter({'International': 945})
Baby Accessories < Counter({'Babies': 253, 'Baby & Child': 19})
Baby Bath & Body Care < Counter({'Babies': 285, 'Baby & Child': 66})
Baby First Aid & Vitamins < Counter({'Babies': 197, 'Baby & Child': 47})
Baby Food & Formula < Counter({'Babies': 1294, 'Baby & Child': 25})
Bakery & Bread < Counter({'Grocery': 1})
Bakery Desserts < Counter({'Bakery': 1048})
Baking Ingredients < Counter({'Pantry': 1157})
Baking Supplies < Counter({'Dry Goods & Pasta': 1})
Baking Supplies & Decor < Counter({'Pantry': 749})
Bar Tools < Counter({'Glassware & Accessories': 25})
Barbera < Counter({'Red Wine': 18})
Bars < Counter({'Grocery': 2})
Beauty < Counter({'Personal Care': 5731})
Beer Glassware < Counter({'Glassware & Accessories': 6})
Beers & Coolers < Counter({'Alcohol': 197})
Belgian-Style Ale < Counter({'Ale Beer': 142})
Beverages < Counter({'Holiday Favorites': 3})
Bitters < Counter({'Mixers, Water & Soda': 8})
Blanco/Silver < Counter({'Tequila': 96})
Blended Scotch < Counter({'Scotch': 93})
Blond Ale < Counter({'Ale Beer': 59})
Bock/Doppelbock/Maibock < Counter({'Lager Beer': 12})
Body Lotions & Soap < Counter({'Personal Care': 2431})
Books & Magazines < Counter({'Household': 1})
Bordeaux Blend < Counter({'Red Wine': 192, 'Dessert & Fortified Wine': 32, 'White Wine': 16})
Brandy < Counter({'Brandy & Cognac': 57})
Bread < Counter({'Bakery': 1194})
Breakfast Bakery < Counter({'Bakery': 509})
Breakfast Bars & Pastries < Counter({'Breakfast': 393})
Brown Ale < Counter({'Ale Beer': 24})
Bulk Candies & Chocolates < Counter({'Bulk': 77})
Bulk Containers < Counter({'Bulk': 5})
Bulk Dried Fruits & Vegetables < Counter({'Bulk': 83})
Bulk Flours & Powders < Counter({'Bulk': 139})
Bulk Grains, Rice & Dried Beans < Counter({'Bulk': 196})
Bulk Grains, Rice & Dried Goods < Counter({'Bulk': 57})
Bulk Granola & Cereals < Counter({'Bulk': 30})
Bulk Home & Personal Care < Counter({'Bulk': 3})
Bulk Nuts & Seeds < Counter({'Bulk': 242})
Bulk Pasta < Counter({'Bulk': 11})
Bulk Soup Mix < Counter({'Bulk': 10})
Bulk Spices & Seasoning < Counter({'Bulk': 5})
Bulk Spices & Seasonings < Counter({'Bulk': 133})
Bulk Spreads Butter, Honey, Syrup < Counter({'Bulk': 19})
Bulk Sugar & Sweeteners < Counter({'Bulk': 14})
Bulk Tea & Coffee < Counter({'Bulk': 169})
Bulk Trail Mix & Snack Mix < Counter({'Bulk': 61})
Buns & Rolls < Counter({'Bakery': 417})
Butter < Counter({'Dairy & Eggs': 289})
```

```
Cabernet Franc < Counter({'Red Wine': 24})
Cabernet Sauvignon < Counter({'Red Wine': 657})
Cachaca < Counter({'Rum': 7})
Calvados < Counter({'Brandy & Cognac': 11})
Candles & Kitchenwares < Counter({'Holiday Favorites': 10})
Candy & Chocolate < Counter({'Snacks': 3085})
Canned & Jarred Vegetables < Counter({'Canned Goods': 950})
Canned Fruit & Applesauce < Counter({'Canned Goods': 543})
Canned Meals & Beans < Counter({'Canned Goods': 797})
Canned Meat & Seafood < Counter({'Canned Goods': 636})
Carmenere < Counter({'Red Wine': 11})
Cat Food & Care < Counter({'Pets': 1281})
Cava < Counter({'Champagne & Sparkling Wine': 38})
Cereal < Counter({'Breakfast': 1144})
Champagne < Counter({'Champagne & Sparkling Wine': 107})
Chardonnay < Counter({'White Wine': 408})
Cheese < Counter({'Food': 2})
Cheese & Appetizers < Counter({'Holiday Favorites': 11})
Chenin Blanc < Counter({'White Wine': 17})
Chili Beer < Counter({'Specialty Style Beer': 2})
Chips & Pretzels < Counter({'Snacks': 1961})
Chips, Puffs & Pretzels < Counter({'Food': 1})
Chocolate & Cream Drinks < Counter({'Ready to Drink': 7})
Chocolate Wine < Counter({'Dessert & Fortified Wine': 3})
Chocolate, Candy & Mints < Counter({'Food': 8})
Chocolate; Sweets & Candy < Counter({'Liqueurs/Cordials/Schnapps': 49})
Christmas < Counter({'Christmas': 5392})
Citrus & Triple Sec < Counter({'Liqueurs/Cordials/Schnapps': 73})
Cleaning Products < Counter({'Household': 1542})
Cleanses & Detoxes < Counter({'Personal Care': 28})
Clothing < Counter({'Personal Care': 34})
Club Soda < Counter({'Mixers, Water & Soda': 5})
Cocktail Mixes < Counter({'Mixers, Water & Soda': 8, 'Alcohol': 7})
Cocktail Rimmers < Counter({'Mixers, Water & Soda': 1})
Cocoa & Drink Mixes < Counter({'Beverages': 495})
Coconut < Counter({'Liqueurs/Cordials/Schnapps': 4})
Coffee < Counter({'Beverages': 2052, 'Liqueurs/Cordials/Schnapps': 32})
Cognac < Counter({'Brandy & Cognac': 88})
Cold, Flu & Allergy < Counter({'Personal Care': 1218})
Condiments < Counter({'Pantry': 1467, 'Food': 1})
Cookies & Cakes < Counter({'Snacks': 1659})
Corvina < Counter({'Red Wine': 21})
Cosmopolitan < Counter({'Ready to Drink': 1})
Cough & Cold Medicine < Counter({'Allergy & Cold Essentials': 30})
Crackers < Counter({'Snacks': 1320})
Cream < Counter({'Dairy & Eggs': 348, 'Liqueurs/Cordials/Schnapps': 64})
Cream Ale < Counter({'Specialty Style Beer': 3})
Czech & German Pilsner < Counter({'Lager Beer': 79})
Daiquiris & Rum Drinks < Counter({'Ready to Drink': 12})
Dairy Alternatives < Counter({'Dairy & Eggs': 115})
Dark Rum < Counter({'Rum': 16})
Deli Meat < Counter({'Deli': 83})
Deodorants < Counter({'Personal Care': 882})
Desserts & Bakery < Counter({'Holiday Favorites': 6})
Diapers & Wipes < Counter({'Babies': 543, 'Baby & Child': 98})
Digestion < Counter({'Personal Care': 1146})
Dish Detergents < Counter({'Household': 391})
Dog Food & Care < Counter({'Pets': 1540})
Doughs, Gelatins & Bake Mixes < Counter({'Pantry': 831})
Dry Pasta < Counter({'Dry Goods & Pasta': 757})
Eau de Vie < Counter({'Brandy & Cognac': 3})
Eggs < Counter({'Dairy & Eggs': 179})
Energy & Granola Bars < Counter({'Snacks': 1708})
Energy & Sports Drinks < Counter({'Beverages': 984})
Entrees < Counter({'Holiday Favorites': 5})
Essentials < Counter({'Allergy & Cold Essentials': 34})
Euro Pale Lager < Counter({'Lager Beer': 42})
Extra Anejo < Counter({'Tequila': 6})
Eye & Ear Care < Counter({'Personal Care': 660})
Facial Care < Counter({'Personal Care': 1659})
Family Planning < Counter({'Personal Care': 259})
Feminine Care < Counter({'Personal Care': 1119})
First Aid < Counter({'Personal Care': 1185})
Flavored Brandy < Counter({'Brandy & Cognac': 21})
Flavored Malt Beverages < Counter({'Coolers & Malt Beverages': 60})
Flavored Rum < Counter({'Rum': 66})
Flavored Sparkling Wine < Counter({'Champagne & Sparkling Wine': 34})
```

```
Flavored Tequila < Counter({'Tequila': 15})
Flavored Vodka < Counter({'Vodka': 261})
Food Storage < Counter({'Household': 629})
Foot Care < Counter({'Personal Care': 139})
Fresh Dips & Tapenades < Counter({'Deli': 416})
Fresh Fruits < Counter({'Produce': 723})
Fresh Herbs < Counter({'Produce': 145})
Fresh Pasta < Counter({'Dry Goods & Pasta': 183})
Fresh Vegetables < Counter({'Produce': 1173})
Frozen < Counter({'Grocery': 27})
Frozen Appetizers & Sides < Counter({'Frozen': 685})
Frozen Breads & Doughs < Counter({'Frozen': 113})
Frozen Breakfast < Counter({'Frozen': 478})
Frozen Dessert < Counter({'Frozen': 299})
Frozen Juice < Counter({'Frozen': 162})
Frozen Meals < Counter({'Frozen': 1953})
Frozen Meat & Seafood < Counter({'Frozen': 480})
Frozen Pizza < Counter({'Frozen': 593})
Frozen Produce < Counter({'Frozen': 707})
Frozen Vegan & Vegetarian < Counter({'Frozen': 378})
Fruit < Counter({'Liqueurs/Cordials/Schnapps': 63})
Fruit & Vegetable Snacks < Counter({'Snacks': 464})
Fruit Beer < Counter({'Specialty Style Beer': 37})
Fruit Blends < Counter({'Fruit Wine': 35, 'Liqueurs/Cordials/Schnapps': 27})
Fruit-Based Dessert Wine < Counter({'Dessert & Fortified Wine': 8})
Gamay < Counter({'Red Wine': 22})
Garden < Counter({'Household': 57})
Garganega < Counter({'White Wine': 6})
Gewurztraminer < Counter({'White Wine': 16})
Gift Baskets < Counter({'Gift Baskets & Sets': 3})
Gift Sets < Counter({'Variety Packs': 9, 'Gift Baskets & Sets': 5})
Gold < Counter({'Tequila': 12})
Gold Rum < Counter({'Rum': 23})
Grains, Rice & Dried Goods < Counter({'Dry Goods & Pasta': 609})
Granola < Counter({'Breakfast': 346})
Grappa < Counter({'Brandy & Cognac': 7})
Grenache < Counter({'Red Wine': 32})
Gruner Veltliner < Counter({'White Wine': 9})
Hair Care < Counter({'Personal Care': 4369})
Hand Care < Counter({'Personal Care': 243})
Herbal & Spice < Counter({'Liqueurs/Cordials/Schnapps': 111})
Herbed/Spiced Beer < Counter({'Specialty Style Beer': 21})
Holiday Pantry < Counter({'Holiday Favorites': 22})
Honeys, Syrups & Nectars < Counter({'Pantry': 465})
Hot Cereal & Pancake Mixes < Counter({'Breakfast': 557})
Hot Dogs, Bacon & Sausage < Counter({'Meat & Seafood': 910})
IPA (India Pale Ale) < Counter({'Ale Beer': 389})
Ice Cream & Ice < Counter({'Frozen': 2507})
Ice Cream Toppings < Counter({'Snacks': 191})
Ice Wine < Counter({'Dessert & Fortified Wine': 8})
Indian Foods < Counter({'International': 148})
Instant Foods < Counter({'Dry Goods & Pasta': 1031, 'Grocery': 11})
Japanese Rice Lager < Counter({'Lager Beer': 5})
Juice < Counter({'Mixers, Water & Soda': 10})
Juice & Nectars < Counter({'Beverages': 2224})
Kitchen Supplies < Counter({'Household': 1163})
Kosher Foods < Counter({'International': 179})
Latino Foods < Counter({'International': 480})
Laundry < Counter({'Household': 1051})
Lemonades & Citrus Drinks < Counter({'Ready to Drink': 12})
Light Lager < Counter({'Lager Beer': 62})
Lunch Meat < Counter({'Deli': 827})
Madeira < Counter({'Dessert & Fortified Wine': 2})
Malbec < Counter({'Red Wine': 85})
Manhattans & Bourbon Drinks < Counter({'Ready to Drink': 1})
Margaritas < Counter({'Ready to Drink': 23})
Marinades & Meat Preparation < Counter({'Pantry': 891})
Marsala < Counter({'Dessert & Fortified Wine': 3})
Mead < Counter({'Dessert & Fortified Wine': 27})
Meat < Counter({'Meat & Seafood': 198})
Meat Counter < Counter({'Meat & Seafood': 397})
Medical Supplies/Aid < Counter({'Personal Care': 46})
Merlot < Counter({'Red Wine': 187})
Milk < Counter({'Dairy & Eggs': 708})
Mint & Gum < Counter({'Snacks': 501})
Mojito < Counter({'Ready to Drink': 2})
Montepulciano < Counter({'Red Wine': 16})
```

```
More Household < Counter({'Household': 3950})
More International Foods < Counter({'International': 21})
Mourvedre/Monastrell < Counter({'Red Wine': 7})
Muscat/Moscato < Counter({'White Wine': 41, 'Champagne & Sparkling Wine': 30})
Muscat/Muscadine < Counter({'Dessert & Fortified Wine': 10})
Muscles, Joints & Pain Relief < Counter({'Personal Care': 991})
Nebbiolo < Counter({'Red Wine': 44})
Newly Added < Counter({'Deli': 5})
Nuts < Counter({'Grocery': 7, 'Food': 6})
Nuts & Amaretto < Counter({'Liqueurs/Cordials/Schnapps': 37})
Nuts, Seeds & Dried Fruit < Counter({'Snacks': 1411})
Oils & Tinctures < Counter({'Personal Care': 52})
Oils & Vinegars < Counter({'Pantry': 948, 'Food': 2})
Oral Hygiene < Counter({'Personal Care': 1978})
Other < Counter({'Bourbon': 101, 'American Whiskey': 87, 'Ready to Drink': 13,
'Liqueurs/Cordials/Schnapps': 12, 'Vodka': 9, 'Rum': 7, 'Brandy & Cognac': 2, 'Tequila': 2})
Other Bulk < Counter({'Bulk': 46})
Other Creams & Cheeses < Counter({'Dairy & Eggs': 417})
Other Dark Lager < Counter({'Lager Beer': 26})
Other Dessert & Fortified Wines < Counter({'Dessert & Fortified Wine': 11})
Other Fruit Cider < Counter({'Cider': 81})
Other Glassware < Counter({'Glassware & Accessories': 14})
Other Pale Lager < Counter({'Lager Beer': 25})
Other Red Wines < Counter({'Red Wine': 78})
Other White Wines < Counter({'White Wine': 56})
Packaged Cheese < Counter({'Dairy & Eggs': 1605})
Packaged Meat < Counter({'Meat & Seafood': 628})
Packaged Poultry < Counter({'Meat & Seafood': 304})
Packaged Seafood < Counter({'Meat & Seafood': 219})
Packaged Vegetables & Fruits < Counter({'Produce': 991})
Pain & Fever < Counter({'Allergy & Cold Essentials': 31})
Pale Ale < Counter({'Ale Beer': 115})
Paper Goods < Counter({'Household': 601})
Pasta & Pasta Sauce < Counter({'Grocery': 12})
Pasta Sauce < Counter({'Dry Goods & Pasta': 713})
Perry (Pear Cider) < Counter({'Cider': 22})
Petite Sirah < Counter({'Red Wine': 17})
Pickled Goods & Olives < Counter({'Pantry': 784})
Pinot Blanc < Counter({'White Wine': 4})
Pinot Grigio/Pinot Gris < Counter({'White Wine': 76})
Pinot Noir < Counter({'Red Wine': 312})
Pisco < Counter({'Brandy & Cognac': 2})
Plates, Bowls, Cups & Flatware < Counter({'Household': 500})
Plum Wine < Counter({'Sake & Plum Wine': 6})
Popcorn & Jerky < Counter({'Snacks': 772})
Port < Counter({'Dessert & Fortified Wine': 72})
Porter < Counter({'Ale Beer': 38})
Poultry < Counter({'Meat & Seafood': 46})
Poultry Counter < Counter({'Meat & Seafood': 162})
Prepared Meals < Counter({'Deli': 1273})
Prepared Soups & Salads < Counter({'Deli': 439})
Preserved Dips & Spreads < Counter({'Pantry': 632})
Produce < Counter({'Holiday Favorites': 32})
Prosecco < Counter({'Champagne & Sparkling Wine': 41})
Protein & Meal Replacements < Counter({'Personal Care': 728})
Red Blend < Counter({'Red Wine': 363})
Red Blend Dessert Wine < Counter({'Dessert & Fortified Wine': 7})
Red Wines < Counter({'Alcohol': 224})
Refrigerated < Counter({'Beverages': 1189})
Refrigerated Pudding & Desserts < Counter({'Dairy & Eggs': 186})
Reposado < Counter({'Tequila': 72})
Rhone Blend < Counter({'Red Wine': 96, 'White Wine': 11})
Riesling < Counter({'White Wine': 91, 'Dessert & Fortified Wine': 13})
Rosés < Counter({'Alcohol': 26})
Rye Whiskey < Counter({'American Whiskey': 76})
Sake < Counter({'Sake & Plum Wine': 42})
Salad Dressing & Toppings < Counter({'Pantry': 1182})
Sangiovese < Counter({'Red Wine': 162})
Sangria < Counter({'Fruit Wine': 28})
Sauvignon Blanc < Counter({'White Wine': 89})
Seafood < Counter({'Meat & Seafood': 66})
Seafood Counter < Counter({'Meat & Seafood': 162})
Seasonal < Counter({'Seasonal/Special Release Beer': 138})
Seltzer Water < Counter({'Mixers, Water & Soda': 4})
Shandy/Radler < Counter({'Specialty Style Beer': 14})
Shave Needs < Counter({'Personal Care': 774})
Sherry < Counter({'Dessert & Fortified Wine': 22})
```

```
Shots < Counter({'Ready to Drink': 9})
Sides & Salads < Counter({'Holiday Favorites': 3})
Silver Rum < Counter({'Rum': 43})
Single Malt < Counter({'Scotch': 103})
Skin Care < Counter({'Personal Care': 250})
Small Animal Care < Counter({'Pets': 153})
Small Batch Bourbon < Counter({'Bourbon': 140})
Soap < Counter({'Personal Care': 299})
Soda < Counter({'Mixers, Water & Soda': 5})
Soft Drinks < Counter({'Beverages': 1567})
Soup, Broth & Bouillon < Counter({'Canned Goods': 1594})
Soy & Lactose-Free < Counter({'Dairy & Eggs': 560})
Sparkling Cider < Counter({'Mixers, Water & Soda': 9})
Sparkling Red Wine < Counter({'Champagne & Sparkling Wine': 45})
Sparkling Wine < Counter({'Champagne & Sparkling Wine': 165})
Special Release < Counter({'Seasonal/Special Release Beer': 80})
Specialty Beer < Counter({'Specialty Style Beer': 57})
Specialty Cheeses < Counter({'Deli': 957})
Specialty Wines & Champagnes < Counter({'Alcohol': 45})
Spiced Rum < Counter({'Rum': 52})
Spices & Seasoning < Counter({'Dry Goods & Pasta': 7})
Spices & Seasonings < Counter({'Pantry': 1977})
Spirits < Counter({'Alcohol': 5})
Spirits Glassware < Counter({'Glassware & Accessories': 14})
Sports Drinks < Counter({'Mixers, Water & Soda': 1})
Spreads < Counter({'Pantry': 974})
Stout < Counter({'Ale Beer': 85})
Stress & Sleep Aids < Counter({'Personal Care': 259})
Strong Ale & Barley Wine < Counter({'Ale Beer': 54})
Sugar & Sugar Substitutes < Counter({'Grocery': 1})
Sun Care < Counter({'Personal Care': 12})
Syrah/Shiraz < Counter({'Red Wine': 148})
Syrups < Counter({'Mixers, Water & Soda': 2})
Tea < Counter({'Beverages': 1877, 'Ready to Drink': 5})
Tempranillo < Counter({'Red Wine': 93})
Tennessee Whiskey < Counter({'American Whiskey': 31})
Tofu & Meat Alternatives < Counter({'Deli': 103, 'Dairy & Eggs': 72, 'Produce': 2})
Tonic Water < Counter({'Mixers, Water & Soda': 12})
Torrontes < Counter({'White Wine': 5})
Tortillas & Flat Bread < Counter({'Bakery': 398})
Trail Mix & Snack Mix < Counter({'Snacks': 254})
Trash Bags & Liners < Counter({'Household': 175})
Trebbiano/Ugni Blanc < Counter({'White Wine': 4})
Unique Cocktails < Counter({'Ready to Drink': 13})
Unique Flavors < Counter({'Liqueurs/Cordials/Schnapps': 13})
Variety Packs < Counter({'Variety Packs': 24, 'Vodka': 1})
Vermouth < Counter({'Dessert & Fortified Wine': 35})
Viognier < Counter({'White Wine': 15})
Vitamins & Supplements < Counter({'Personal Care': 4247})
Vodka < Counter({'Vodka': 325})
Water < Counter({'Mixers, Water & Soda': 40})
Water, Seltzer & Sparkling Water < Counter({'Beverages': 1212})
Wheat Ale < Counter({'Ale Beer': 101})
Whiskey < Counter({'American Whiskey': 226})
White Blend < Counter({'White Wine': 66})
White Blend Dessert Wine < Counter({'Dessert & Fortified Wine': 6})
White Wines < Counter({'Alcohol': 63})
Whole & Ground Seeds < Counter({'Dry Goods & Pasta': 3})
Wine Accessories < Counter({'Glassware & Accessories': 33})
Wine Coolers < Counter({'Coolers & Malt Beverages': 7})
Wine Glassware < Counter({'Glassware & Accessories': 36})
Wine Storage & Transport < Counter({'Glassware & Accessories': 5})
Yogurt < Counter({'Dairy & Eggs': 1816})
Zinfandel < Counter({'Red Wine': 93})
```

6. Generate a new file where mrp is normalized.

In [ ]:

```
%%time
a
for i in range(len(a)): # Here we are taking today's data
    if a['mrp'][i]==0 or a['mrp'][i]=='nan' :
        a['mrp'][i]='NA'
```

```
# isinstance(value,float)=True means values in float.
# If mrp value is 'na' or non-float or 0 then, it has been set to 'NA'
```

The above code was taking awefully long time to execute. But the code is fine and working.

In [133]:

```
with open('a11.json', 'w') as f:
    f.write(a.to_json(orient='records', lines=True))
```

Thank YOU

This assignment was really fun. You explicitly mentioned in 6th point to have fun. Yes I had fun, learnt a a lot doing this assignment.

My understanding in ML is good, can't wait to share my knowledge with you people. Thank You.