Run Apriori algorithm to find frequent itemsets and association rules

In [177]:
```
import pandas as pd
import numpy as np
df = pd.read_csv("/content/drive/MyDrive/Datasets/groceries.csv",sep=',')
df = df.dropna(how='all', axis=1)
df
```

Out[177]:

	Item(s)	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10	Item 11	Item 12	Item 13	Item 14
0	4	citrus fruit	semi-finished bread	margarine	ready soups	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	3	tropical fruit	yogurt	coffee	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	1	whole milk	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	4	pip fruit	yogurt	cream cheese	meat spreads	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	4	other vegetables	whole milk	condensed milk	long life bakery product	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9830	17	sausage	chicken	beef	hamburger meat	citrus fruit	grapes	root vegetables	whole milk	butter	whipped/sour cream	flour	coffee	red/blush wine	salty snack
9831	1	cooking chocolate	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9832	10	chicken	citrus fruit	other vegetables	butter	yogurt	frozen dessert	domestic eggs	rolls/buns	rum	cling film/bags	NaN	NaN	NaN	NaN
9833	4	semi-finished bread	bottled water	soda	bottled beer	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9834	5	chicken	tropical fruit	other vegetables	vinegar	shopping bags	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

9835 rows × 33 columns

In [178]:
```
df = df.iloc[:,1:]
df = df.replace(to_replace = np.nan, value = 'nan')
df
```

Out[178]:

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10	Item 11	Item 12	Item 13	Item 14	Item
0	citrus fruit	semi-finished bread	margarine	ready soups	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	r
1	tropical fruit	yogurt	coffee	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	r
2	whole milk	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	r
3	pip fruit	yogurt	cream cheese	meat spreads	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	r
4	other vegetables	whole milk	condensed milk	long life bakery product	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	r
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9830	sausage	chicken	beef	hamburger meat	citrus fruit	grapes	root vegetables	whole milk	butter	whipped/sour cream	flour	coffee	red/blush wine	salty snack	chocol
9831	cooking chocolate	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	r
9832	chicken	citrus fruit	other vegetables	butter	yogurt	frozen dessert	domestic eggs	rolls/buns	rum	cling film/bags	nan	nan	nan	nan	r
9833	semi-finished bread	bottled water	soda	bottled beer	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	r
9834	chicken	tropical fruit	other vegetables	vinegar	shopping bags	nan	nan	nan	nan	nan	nan	nan	nan	nan	r

9835 rows × 32 columns

In [179]:
```
df.shape
```

Out[179]:
```
(9835, 32)
```

In [180]:
```
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
import matplotlib.pyplot as plt
```

In [181]:
```
transactions = []

for i in range(0, 9835):
  temp=[]
  for u in df.columns:
    if df[u][i]!='nan':
      temp.append(df[u][i])
  transactions.append(temp)
```

In [182]:
```
TE = TransactionEncoder()
dataset = TE.fit(transactions).transform(transactions)
new_df = pd.DataFrame(dataset,columns = TE.columns_)
new_df
```

Out[182]:

	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics	baby food	bags	baking powder	bathroom cleaner	beef	berries	beverages	bottled beer	bottled water	brandy	brow brea
0	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9830	False	False	False	False	False	False	False	False	False	True	False	False	False	False	False	Fals
9831	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	Fals
9832	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	Fals
9833	False	False	False	False	False	False	False	False	False	False	False	False	True	True	False	Fals
9834	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	Fals

9835 rows × 169 columns

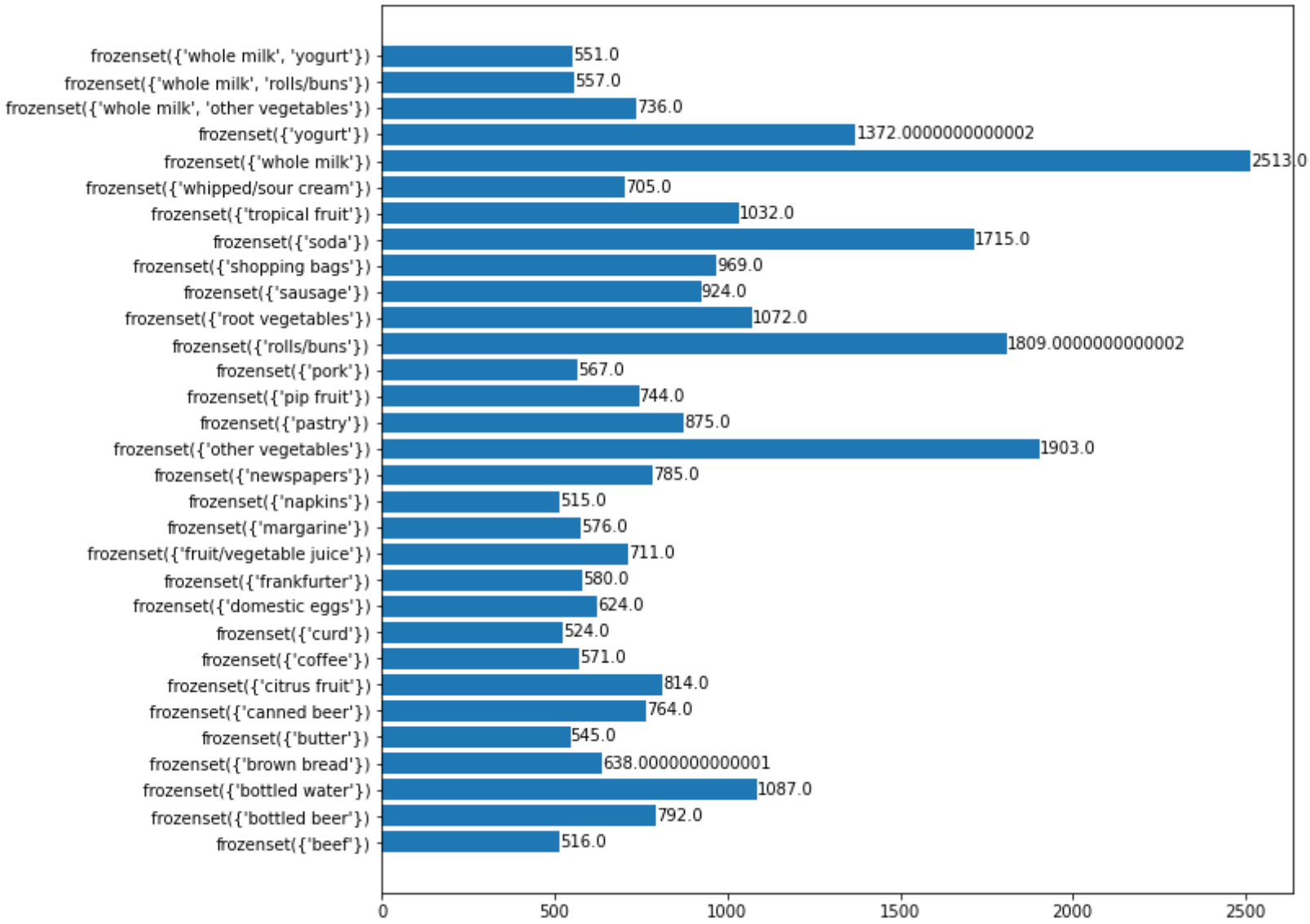
In [183]:
```
itemset = apriori(new_df,min_support=0.05,use_colnames=True)
itemset
```

Out[183]:

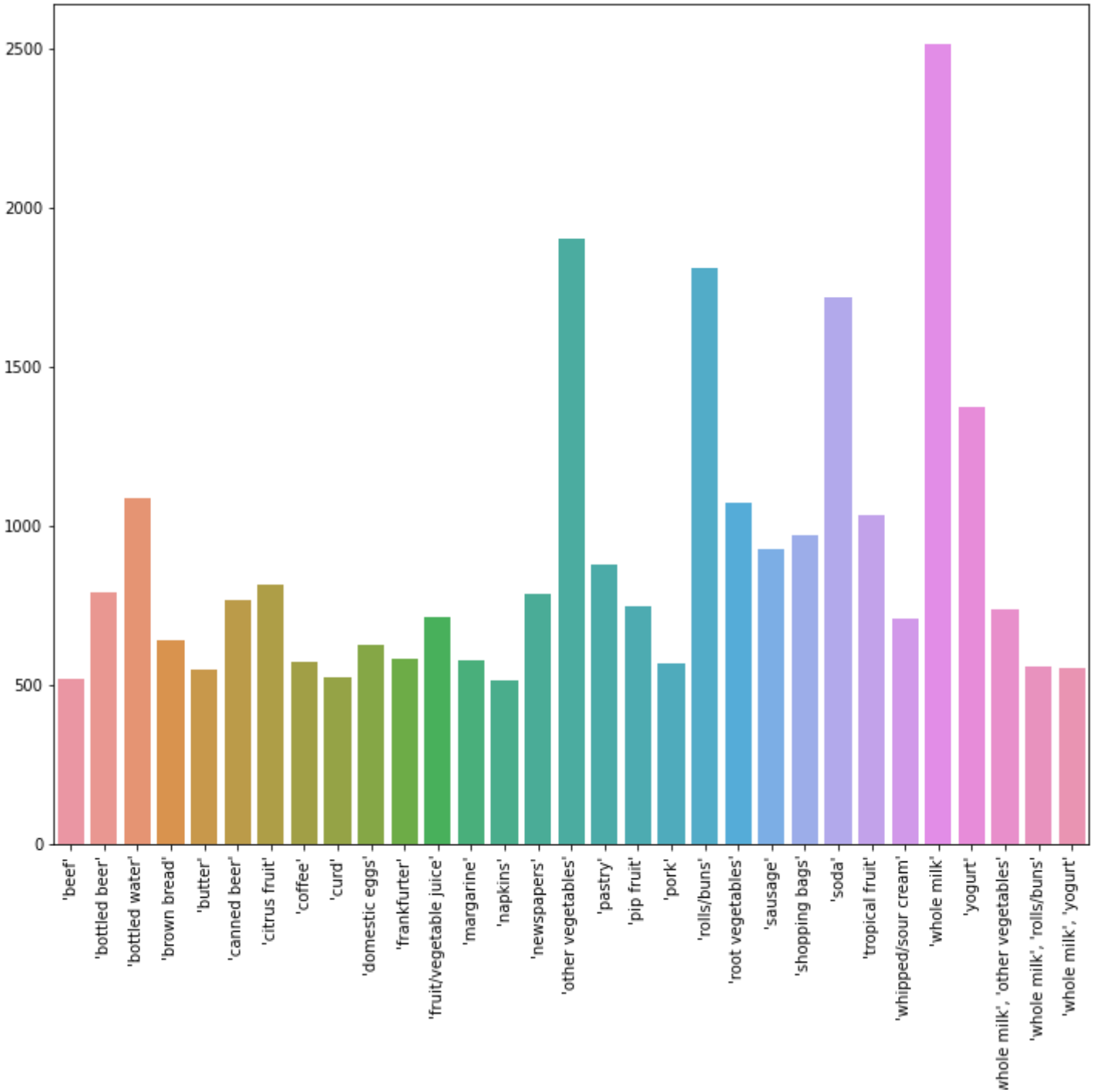
	support	itemsets
0	0.052466	(beef)
1	0.080529	(bottled beer)
2	0.110524	(bottled water)
3	0.064870	(brown bread)
4	0.055414	(butter)
...	...	...
26	0.255516	(whole milk)
27	0.139502	(yogurt)
28	0.074835	(whole milk, other vegetables)
29	0.056634	(whole milk, rolls/buns)
30	0.056024	(whole milk, yogurt)

31 rows × 2 columns

In [176]:
```
sp = [float(itemset['support'][i])*9835 for i in np.arange(0,itemset.shape[0])]
items = [str(itemset['itemsets'][i]) for i in np.arange(0,itemset.shape[0])]
plt.figure(figsize=(10,10))
plt.barh(items,sp)
for i in range(len(items)):
        plt.text(s=sp[i],x=sp[i]+0.1,y=i,va='center',ha="left")
```



In [184]:
```
import seaborn as sns
plt.figure(figsize=(12,10))
bar = sns.barplot(x=items,y=sp)
l=pd.DataFrame(bar.get_xticklabels(),columns=['label'])
l['label']=l['label'].apply(lambda x:x.get_text()[11:-2]).astype("unicode")
ax=bar.set_xticklabels(l['label'],rotation=90)
```


2500
2000
1500
1000
500
0
'beef'
'bottled beer'
'bottled water'
'brown bread'
'butter'
'canned beer'
'citrus fruit'
'coffee'
'curd'
'domestic eggs'
'frankfurter'
'fruit/vegetable juice'
'margarine'
'napkins'
'newspapers'
'other vegetables'
'pastry'
'pip fruit'
'pork'
'rolls/buns'
'root vegetables'
'sausage'
'shopping bags'
'soda'
'tropical fruit'
'whipped/sour cream'
'whole milk'
'yogurt'
whole milk', 'other vegetables'
'whole milk', 'rolls/buns'
'whole milk', 'yogurt'

In [185]:
```
association_rules(itemset,metric='confidence',min_threshold=0.3)
```

Out[185]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(other vegetables)	(whole milk)	0.193493	0.255516	0.074835	0.386758	1.513634	0.025394	1.214013
1	(rolls/buns)	(whole milk)	0.183935	0.255516	0.056634	0.307905	1.205032	0.009636	1.075696
2	(yogurt)	(whole milk)	0.139502	0.255516	0.056024	0.401603	1.571735	0.020379	1.244132

In [186]:
```
!cp  '/content/drive/MyDrive/Colab Notebooks/Practical 4.ipynb' ./
!jupyter nbconvert --to html "Practical 4"
```

```
[NbConvertApp] Converting notebook Practical 4.ipynb to html
[NbConvertApp] Writing 466277 bytes to Practical 4.html
```