Final Assignement WorkEarly Report

Christos Dimas

March 2023

1 Introduction

This is a brief report regarding the Final WorkEarly Data Science School Project (https://github.com/Workearly/Final-Assignment). A dataset that contains Liquor Sales in the state of Iowa in USA between 2012-2020 is given. The purpose is to find the most popular item per zipcode and the percentage of sales per store in the period between 2016-2019.

2 Brief Workflow

- Write a simple MySQL query to collect data from `finance_liquor_sales.sql` dataset that includes the sales that took place between 01-01-2016 and 31-12-2019.
- Save MySQL output to a csv file, called `Liquor_Sales_2016_2019.csv`.
- Write a Python code to read, inspect, process the data and produce the desired outputs.
- Visualize the outputs using *matplotlib* and *seaborn*
- Re-process the data and visualize the outputs using *Tableau*.

3 Data Processing

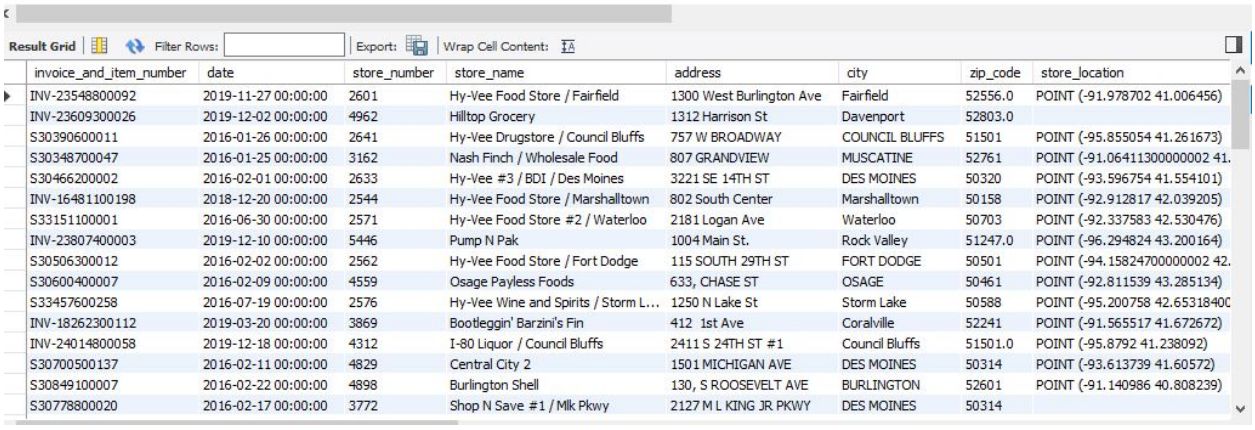
3.1 Get 2016-2019 data using MySQL

The `finance_liquor_sales.sql` database was loaded in MySQL Workbench and then a simple query was written in order to get the sales data between 01-01-2016 and 31-12-2019. The query as well as the output is demonstrated in Fig. 1. At first, the database was loaded and inspected in order to understand its structure. Then we use the *WHERE* and *BETWEEN* commands to keep the sales between 2016-2019.

After obtaining the results above, we export them to the `Liquor_Sales_2016_2019.csv` file, in order to further process them via Python.

```sql
show databases;
use liquorsales;
show tables;
SELECT * FROM finance_liquor_sales;

SELECT * FROM finance_liquor_sales
WHERE date BETWEEN '2016-01-01 00:00:00' AND '2019-12-31 23:59:59';
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

invoice_and_item_number	date	store_number	store_name	address	city	zip_code	store_location
INV-23548800092	2019-11-27 00:00:00	2601	Hy-Vee Food Store / Fairfield	1300 West Burlington Ave	Fairfield	52556.0	POINT (-91.978702 41.006456)
INV-23609300026	2019-12-02 00:00:00	4962	Hilltop Grocery	1312 Harrison St	Davenport	52803.0	
S30390600011	2016-01-26 00:00:00	2641	Hy-Vee Drugstore / Council Bluffs	757 W BROADWAY	COUNCIL BLUFFS	51501	POINT (-95.855054 41.261673)
S30348700047	2016-01-25 00:00:00	3162	Nash Finch / Wholesale Food	807 GRANDVIEW	MUSCATINE	52761	POINT (-91.06411300000002 41.
S30466200002	2016-02-01 00:00:00	2633	Hy-Vee #3 / BDI / Des Moines	3221 SE 14TH ST	DES MOINES	50320	POINT (-93.596754 41.554101)
INV-16481100198	2018-12-20 00:00:00	2544	Hy-Vee Food Store / Marshalltown	802 South Center	Marshalltown	50158	POINT (-92.912817 42.039205)
S33151100001	2016-06-30 00:00:00	2571	Hy-Vee Food Store #2 / Waterloo	2181 Logan Ave	Waterloo	50703	POINT (-92.337583 42.530476)
INV-23807400003	2019-12-10 00:00:00	5446	Pump N Pak	1004 Main St.	Rock Valley	51247.0	POINT (-96.294824 43.200164)
S30506300012	2016-02-02 00:00:00	2562	Hy-Vee Food Store / Fort Dodge	115 SOUTH 29TH ST	FORT DODGE	50501	POINT (-94.15824700000002 42.
S30600400007	2016-02-09 00:00:00	4559	Osage Payless Foods	633, CHASE ST	OSAGE	50461	POINT (-92.811539 43.285134)
S33457600258	2016-07-19 00:00:00	2576	Hy-Vee Wine and Spirits / Storm L...	1250 N Lake St	Storm Lake	50588	POINT (-95.200758 42.65318400
INV-18262300112	2019-03-20 00:00:00	3869	Bootleggin' Barzini's Fin	412 1st Ave	Coralville	52241	POINT (-91.565517 41.672672)
INV-24014800058	2019-12-18 00:00:00	4312	I-80 Liquor / Council Bluffs	2411 S 24TH ST #1	Council Bluffs	51501.0	POINT (-95.8792 41.238092)
S30700500137	2016-02-11 00:00:00	4829	Central City 2	1501 MICHIGAN AVE	DES MOINES	50314	POINT (-93.613739 41.60572)
S30849100007	2016-02-22 00:00:00	4898	Burlington Shell	130, S ROOSEVELT AVE	BURLINGTON	52601	POINT (-91.140986 40.808239)
S30778800020	2016-02-17 00:00:00	3772	Shop N Save #1 / Mlk Pkwy	2127 M L KING JR PKWY	DES MOINES	50314	

Figure 1: MySQL Query with corresponding results

3.2 Data Processing using Python/Pandas and Visualization

At first, we import the essential packages and read the csv file using Pandas:

```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt

###### import Liquor_Sales_2016_2019.csv which includes
###### Liquor Sales
###### in the state of Iowa in USA between 2016-2019.

file2read='Liquor_Sales_2016_2019.csv'
df=pd.read_csv(file2read)
```

At next, we perform a first inspection of the dataframe's content, in order to better understand it. For this reason, we print the columns' names and get the 10 first rows' values (4 columns selected to appear through the Nc variable):

```
Nc=4

#########----first inspection-----
###### get columns' names:

print(df.columns,'\n')

##### select to display Nc columns

pd.set_option('display.max_columns', Nc)

###### check headers (first 10 rows)'

print(df.head(10))
```

The output appeared is shown in Fig. 2:

Furthermore, we reveal the popularity of each item's in terms of sales (Fig. 3:

```
###### show most popular (top sales) item IDs in Iowa
print(df['item_number'].value_counts())
```

Then, in order to reveal the summary of bottles sold per item ID and zip code, we use the *groupby* operation, summing the bottles sold per each group. The produced dataframe's indices as well as the bottles' summaries are transformed to (parallel) numpy arrays and properly grouped in a single *data1* numpy array.

```
Index(['invoice_and_item_number', 'date', 'store_number', 'store_name',
       'address', 'city', 'zip_code', 'store_location', 'county_number',
       'county', 'category', 'category_name', 'vendor_number',
'vendor_name',
       'item_number', 'item_description', 'pack', 'bottle_volume_ml',
       'state_bottle_cost', 'state_bottle_retail', 'bottles_sold',
       'sale_dollars', 'volume_sold_liters', 'volume_sold_gallons'],
      dtype='object')
```

Figure 2: Dataset columns' names

```
In [3]: print(df['item_number'].value_counts())
86251     6
48099     4
43031     4
77487     3
43034     3
67557     2
86112     2
67586     2
56193     2
67524     2
75087     2
35917     1
43040     1
38089     1
67526     1
86843     1
82187     1
86739     1
```

Figure 3: Top sales item IDs

```
####### group by zip code and item number, sum of bottles
####### sold per item in each zip code

bottles_items_by_zip=df.groupby(['zip_code','item_number'\
,'item_description'])['bottles_sold'].sum()

#### to numpy

zipcodes=np.array(bottles_items_by_zip.index.\
get_level_values(0),dtype=np.intc)
itemids=np.array(bottles_items_by_zip.index.\
get_level_values(1),dtype=np.intc)
itemdesc=np.array(bottles_items_by_zip.index.\
get_level_values(2))
bottles1=bottles_items_by_zip.to_numpy(dtype=np.intc)
data1=np.array([zipcodes,itemids,itemdesc,bottles1])
```

To visualize this data we use a *seaborn* package scatterplot:

```
axes1=sns.scatterplot(x=zipcodes,y=bottles1,hue=itemdesc,\
                      data=itemids)
axes1.set(xlabel='Zip Code', ylabel='# of bottles sold')
plt.title('Bottles sold by zip code and item')
plt.legend(ncol=4,shadow=True,bbox_to_anchor=(0.5,-0.55),\
           loc='lower center', borderaxespad=0)
plt.setp(axes1.get_legend().get_texts(), fontsize='7')
plt.subplots_adjust(bottom=0.35)
plt.xlim(min(zipcodes),max(zipcodes))
plt.ylim(-10,max(bottles1)+100)
major_ticks = np.arange(min(zipcodes), max(zipcodes), 250)
minor_ticks = np.arange(min(zipcodes), max(zipcodes), 50)
axes1.set_xticks(major_ticks)
axes1.set_xticks(minor_ticks, minor=True)
axes1.grid(which='minor', alpha=0.2)
axes1.grid(which='major', alpha=0.5)
manager = plt.get_current_fig_manager()
manager.window.showMaximized()
```

The following figure appears (Fig. 4):

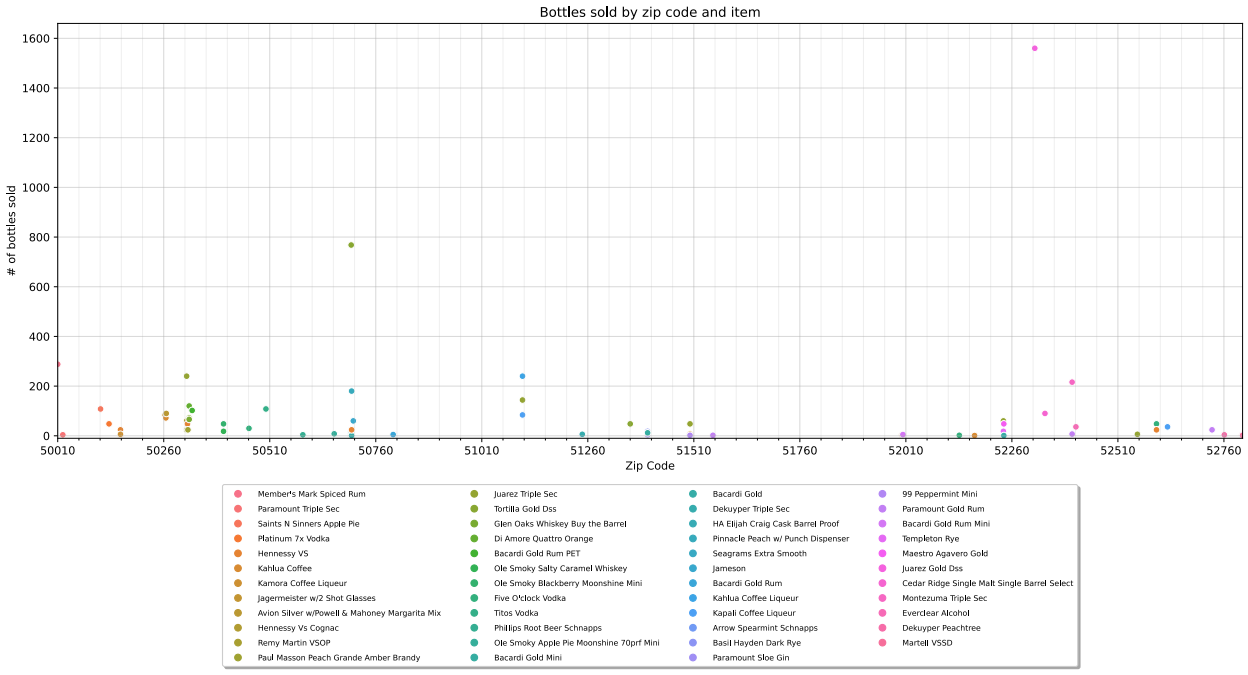


Figure 4: Bottles sold by zip code and item.

Then we create a horizontal bar chart of the top 20 stores by the number of bottles sold in descending order (the top number can be configured through the *TopN* variable. In this part of code, we call the customly created `create_horz_barplot` function.

```
#########################################################
####### group by store (number), sum of sales
####### (in bottles) per store
TopN=20
saleskey='(bottles)'
bottles_by_store=df.groupby(['store_number','city'])\
['bottles_sold'].sum()
TotalSalesBottles=np.sum(bottles_by_store)
percofbottles_by_store=100*bottles_by_store/TotalSalesBottles
percofbottles_by_store_sort=percofbottles_by_store.sort_values\
(ascending=False).round(decimals=2)
palette='GnBu_d'
create_horz_barplot(percofbottles_by_store_sort,TopN,palette,saleskey)
```

The code written to implement the `create_horz_barplot` function is demonstrated below:

```
##### function visualize with Seaborn horizontal barplot
def create_horz_barplot(df,TopN,palette,saleskey):
    storeids=np.array(df.index.get_level_values(0),dtype=np.intc)
    store_city=np.array(df.index.get_level_values(1))
    storeinfo=np.column_stack((storeids,store_city))
    storeinfoall=np.apply_along_axis(lambda d: str(d[0]) + ', '\
    + d[1], 1, storeinfo)
    percvals=df.to_numpy()
    plt.figure()
    axes=sns.barplot(y=storeinfoall[0:TopN],x=percvals[0:TopN],\
    palette=palette)
    axes.set(xlabel='Sales Percentage (%)', ylabel='Store, City')
    plt.title('Top '+str(TopN)+' Stores by sales '+saleskey+' \
    percentage')
    axes.grid()
    manager = plt.get_current_fig_manager()
    manager.window.showMaximized()
```

The produced barchart is depicted in Fig. 5 below:

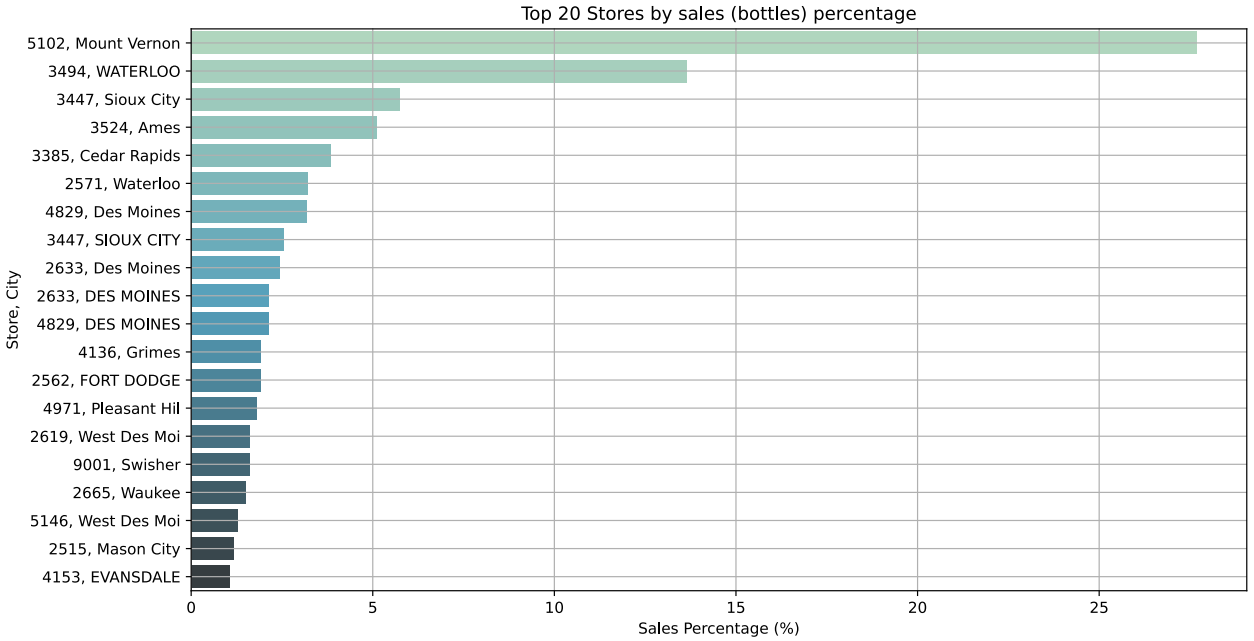


Figure 5: Top 20 stores by sales (number of bottles) percentage

The same process was followed to reveal the first 20 stores by sales in dollars (percentages):

```
####### group by store (number), sum of sales ($) per store
TopN=20
saleskey='($)'
sales_by_store=df.groupby(['store_number','city'])['sale_dollars'].sum()
TotalSales=np.sum(sales_by_store)
##### get percentages
percofsales_by_store=100*sales_by_store/TotalSales
percofsales_by_store_sort=percofsales_by_store.sort_values\
(ascending=False).round(decimals=2)
palette='YlOrBr'
create_horz_barplot(percofsales_by_store_sort,TopN,palette,saleskey)
```

The produced barchart is depicted in Fig. 6 below:

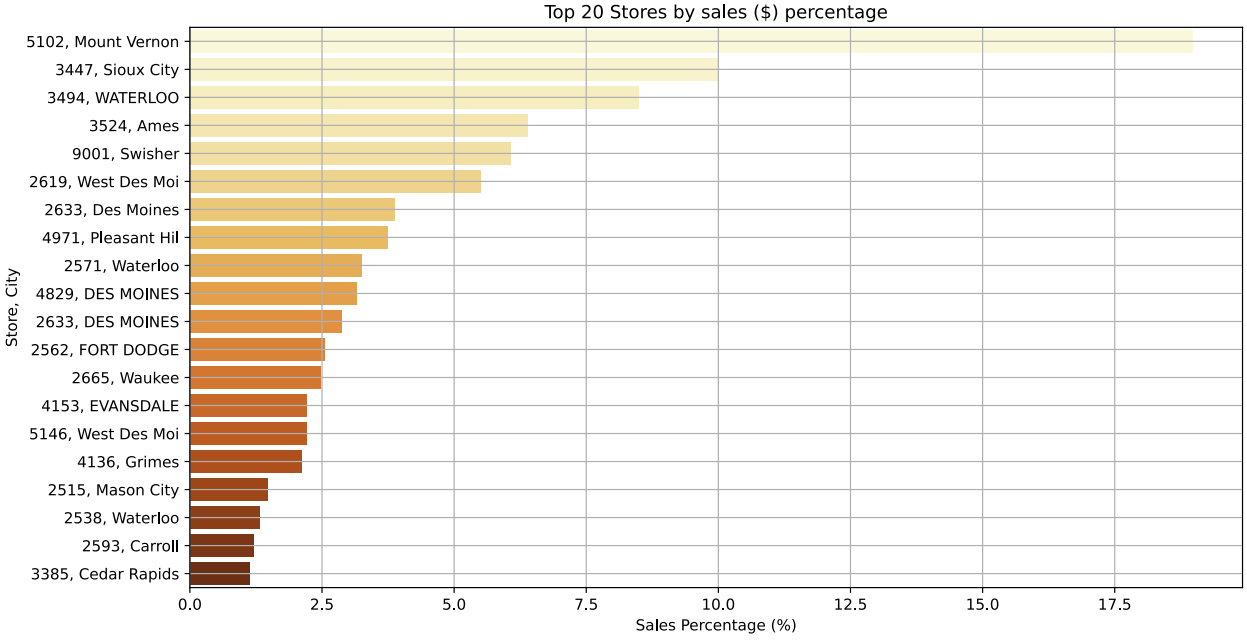


Figure 6: Top 20 stores by sales ($) percentage

The same process was followed to reveal the first 20 stores by sales in terms of gallons (percentages):

```
TopN=20
saleskey='(gallons)'
gallons_by_store=df.groupby(['store_number','city'])\
['volume_sold_gallons'].sum()
TotalGallons=np.sum(gallons_by_store)
##### get percentages
percofgallons_by_store=100*gallons_by_store/TotalGallons
percofgallons_by_store_sort=percofgallons_by_store.sort_values\
(ascending=False).round(decimals=2)
palette='icefire'
create_horz_barplot(percofgallons_by_store_sort,TopN,palette,saleskey)
```

with the following barchart produced (Fig. 7):

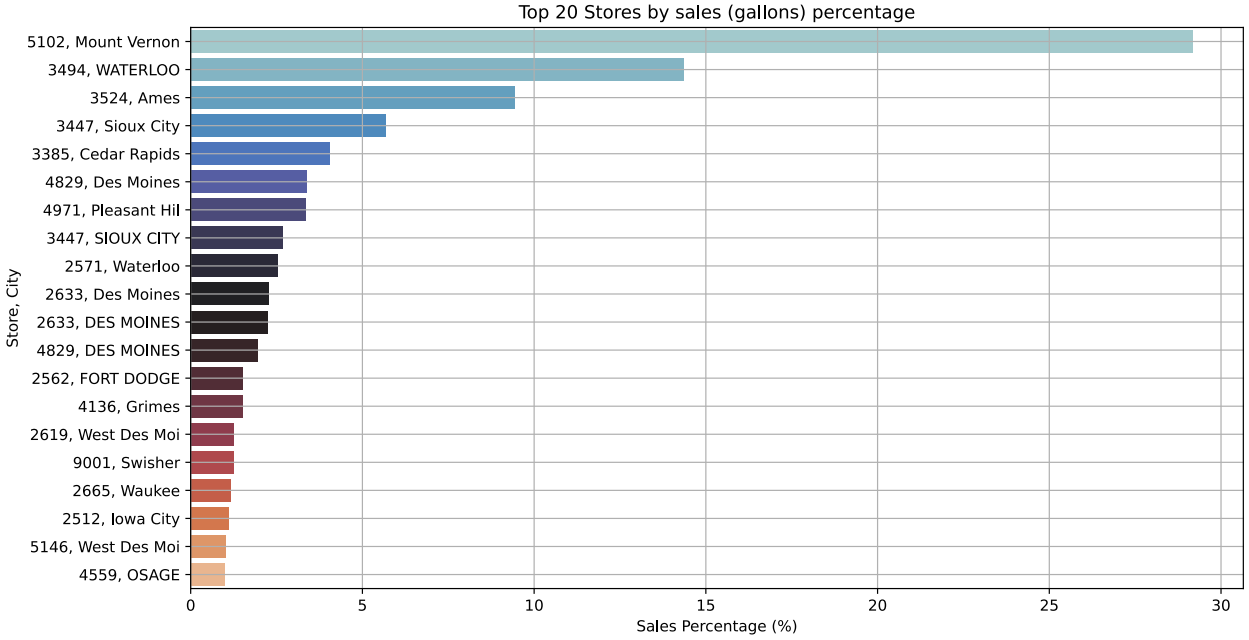


Figure 7: Top 20 stores by sales (gallons) percentage

The whole code written to implement the processing and visualization above is presented below:

```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt


###### import Liquor_Sales_2016_2019.csv which includes Liquor Sales
###### in the state of Iowa in USA between 2016-2019.
file2read='Liquor_Sales_2016_2019.csv'
df=pd.read_csv(file2read)

Nc=4
#########---first inspection-----
###### get columns' names:
print(df.columns,'\n')
##### select to display Nc columns
pd.set_option('display.max_columns', Nc)
###### check headers (first 10 rows)'
print(df.head(10))

###### show most popular (top sales) item IDs in Iowa
print(df['item_number'].value_counts())

####### group by zip code and item number, sum of bottles
####### sold per item in each zip code
bottles_items_by_zip=df.groupby(['zip_code','item_number','item_description'])\
    ['bottles_sold'].sum()
#### to numpy
zipcodes=np.array(bottles_items_by_zip.index.get_level_values(0),dtype=np.intc)
itemids=np.array(bottles_items_by_zip.index.get_level_values(1),dtype=np.intc)
itemdesc=np.array(bottles_items_by_zip.index.get_level_values(2))
bottles1=bottles_items_by_zip.to_numpy(dtype=np.intc)
data1=np.array([zipcodes,itemids,itemdesc,bottles1])

####### visualize with Seaborn scatterplot
axes1=sns.scatterplot(x=zipcodes,y=bottles1,hue=itemdesc,\
                      data=itemids)
axes1.set(xlabel='Zip Code', ylabel='# of bottles sold')
plt.title('Bottles sold by zip code and item')

plt.legend(ncol=4,shadow=True,bbox_to_anchor=(0.5, -0.55), \
           loc='lower center', borderaxespad=0)
plt.setp(axes1.get_legend().get_texts(), fontsize='7')
```

```
plt.subplots_adjust(bottom=0.35)
plt.xlim(min(zipcodes),max(zipcodes))
plt.ylim(-10,max(bottles1)+100)
major_ticks = np.arange(min(zipcodes), max(zipcodes), 250)
minor_ticks = np.arange(min(zipcodes), max(zipcodes), 50)
axes1.set_xticks(major_ticks)
axes1.set_xticks(minor_ticks, minor=True)
axes1.grid(which='minor', alpha=0.2)
axes1.grid(which='major', alpha=0.5)
manager = plt.get_current_fig_manager()
manager.window.showMaximized()

##### function visualize with Seaborn horizontal barplot
def create_horz_barplot(df,TopN,palette,saleskey):
    storeids=np.array(df.index.get_level_values(0),dtype=np.intc)
    store_city=np.array(df.index.get_level_values(1))
    storeinfo=np.column_stack((storeids,store_city))
    storeinfoall=np.apply_along_axis(lambda d: str(d[0]) + ', ' +\
                                     d[1], 1, storeinfo)
    percvals=df.to_numpy()
    plt.figure()
    axes=sns.barplot(y=storeinfoall[0:TopN],x=percvals[0:TopN],palette=palette)
    axes.set(xlabel='Sales Percentage (%)', ylabel='Store, City')
    plt.title('Top '+str(TopN)+' Stores by sales '+saleskey+' percentage')
    axes.grid()
    manager = plt.get_current_fig_manager()
    manager.window.showMaximized()


#########################################################
####### group by store (number), sum of sales (in bottles) per store
TopN=20
saleskey='(bottles)'
bottles_by_store=df.groupby(['store_number','city'])['bottles_sold'].sum()
TotalSalesBottles=np.sum(bottles_by_store)
percofbottles_by_store=100*bottles_by_store/TotalSalesBottles
percofbottles_by_store_sort=percofbottles_by_store.sort_values\
    (ascending=False).round(decimals=2)
palette='GnBu_d'
create_horz_barplot(percofbottles_by_store_sort,TopN,palette,saleskey)

####### group by store (number), sum of sales ($) per store
TopN=20
saleskey='($)'
sales_by_store=df.groupby(['store_number','city'])['sale_dollars'].sum()
TotalSales=np.sum(sales_by_store)
```

```
##### get percentages
percofsales_by_store=100*sales_by_store/TotalSales
percofsales_by_store_sort=percofsales_by_store.sort_values\
    (ascending=False).round(decimals=2)
palette='YlOrBr'
create_horz_barplot(percofsales_by_store_sort,TopN,palette,saleskey)

####### group by store (number), sum of sold gallons per store
TopN=20
saleskey='(gallons)'
gallons_by_store=df.groupby(['store_number','city'])\
    ['volume_sold_gallons'].sum()
TotalGallons=np.sum(gallons_by_store)
##### get percentages
percofgallons_by_store=100*gallons_by_store/TotalGallons
percofgallons_by_store_sort=percofgallons_by_store.sort_values\
    (ascending=False).round(decimals=2)
palette='icefire'
create_horz_barplot(percofgallons_by_store_sort,TopN,palette,saleskey)
```

3.3 Visualization using Tableau

In this part, we exploit Tableau Public in order to visualize some aspects of the `Liquor_Sales_2016_2019.csv` dataset. Fig. 8 shows the data loaded in Tableau DataSource.

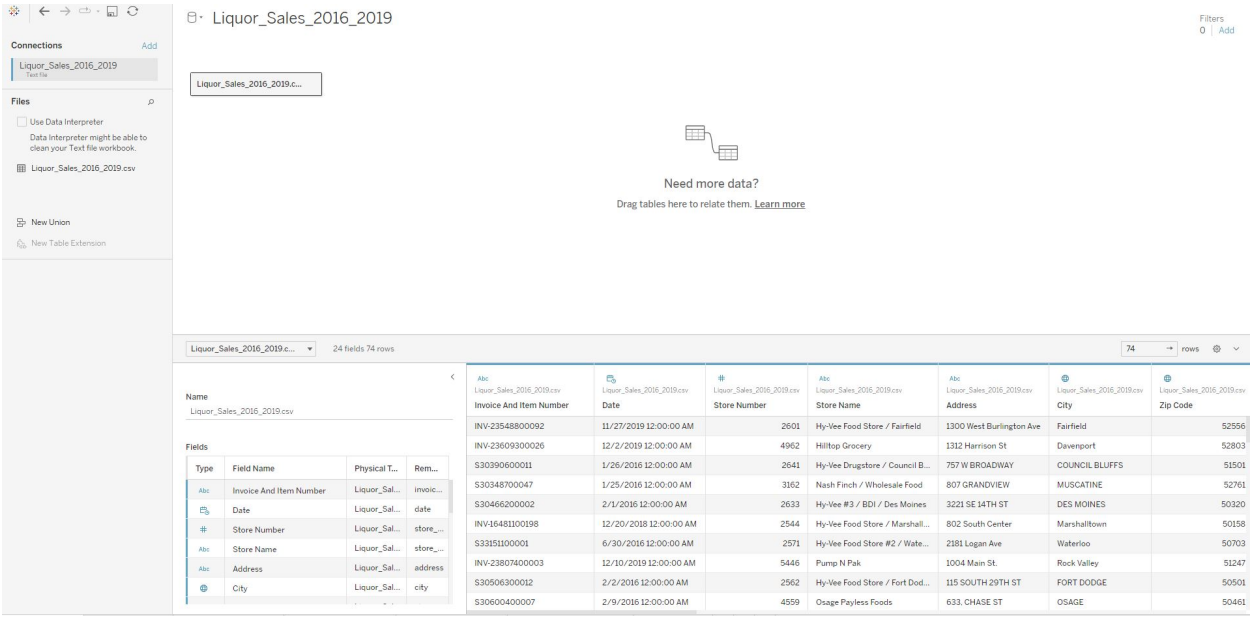


Figure 8: Tableau DataSource.

Figure 9 depicts the number of bottles sold, grouped by per Zip code and item (descending). The corresponding link to this sheet can be found in here.

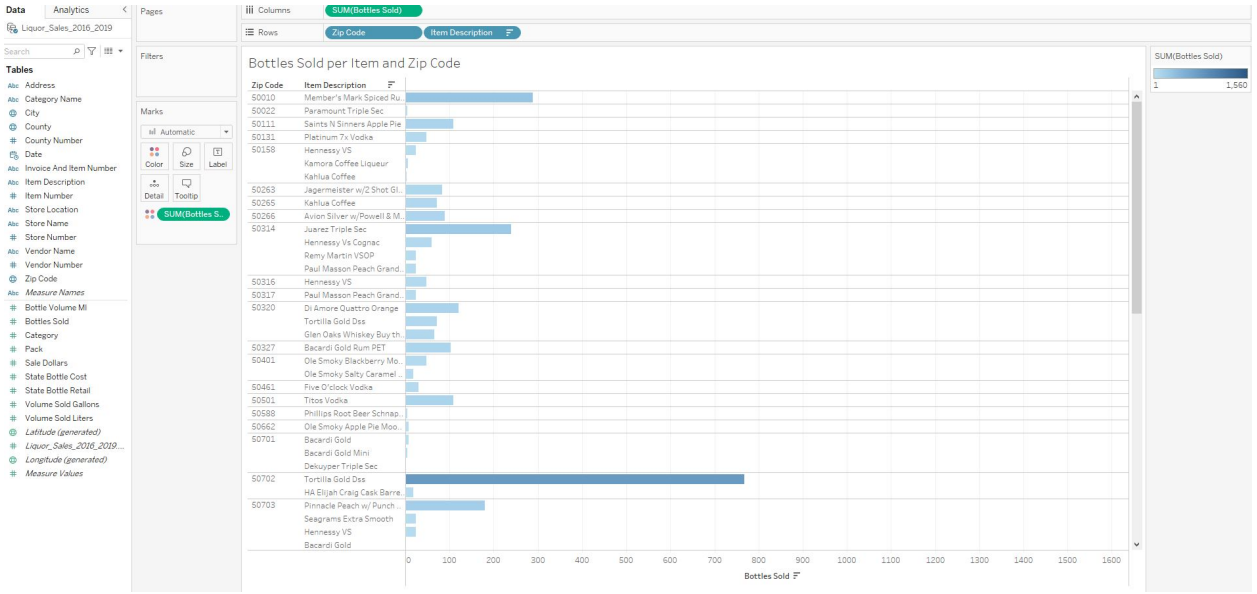


Figure 9: Bottles sold per zip code and item.

Figure 10 depicts the sales value in dollars per item store (raw $ values, not percentage). The corresponding link to this sheet can be found in here.

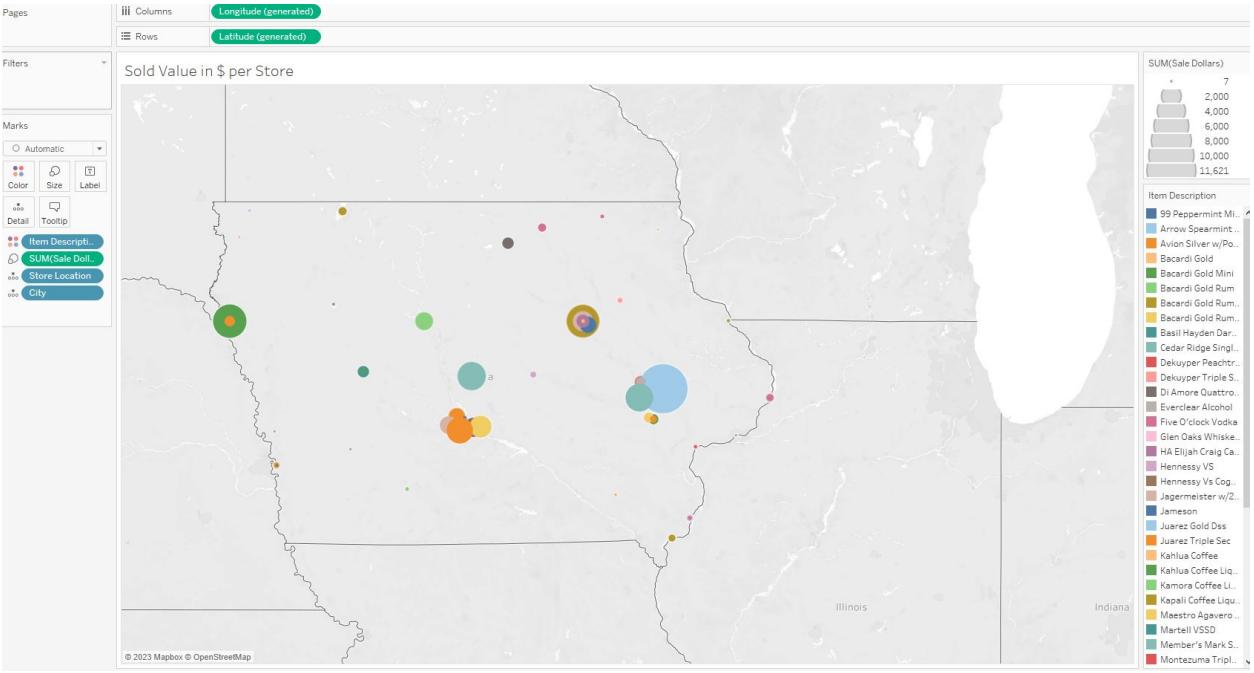


Figure 10: Sold value in ($) per store.

Figure 11 depicts the sales value in gallons per item store (raw gallons' values, not percentage). The corresponding link to this sheet can be found in here.

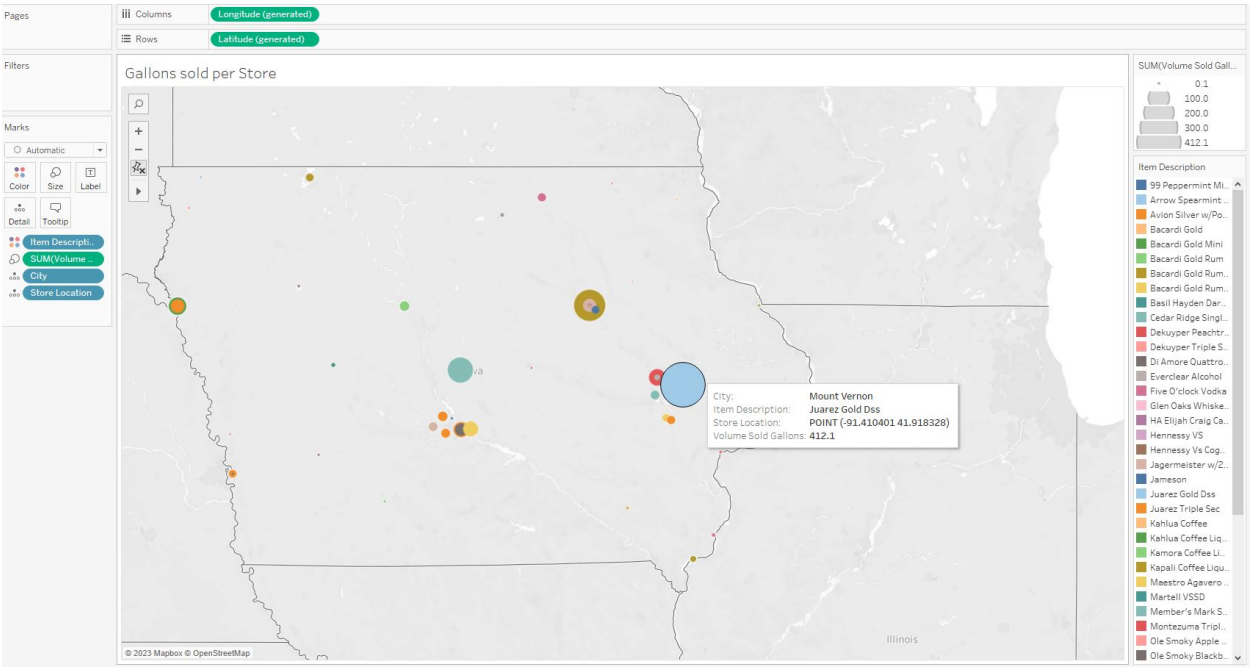


Figure 11: Sold value in gallons per store.

Figure 12 depicts the number of bottles sold per item store (number of bottles, not percentage). The corresponding link to this sheet can be found in here.

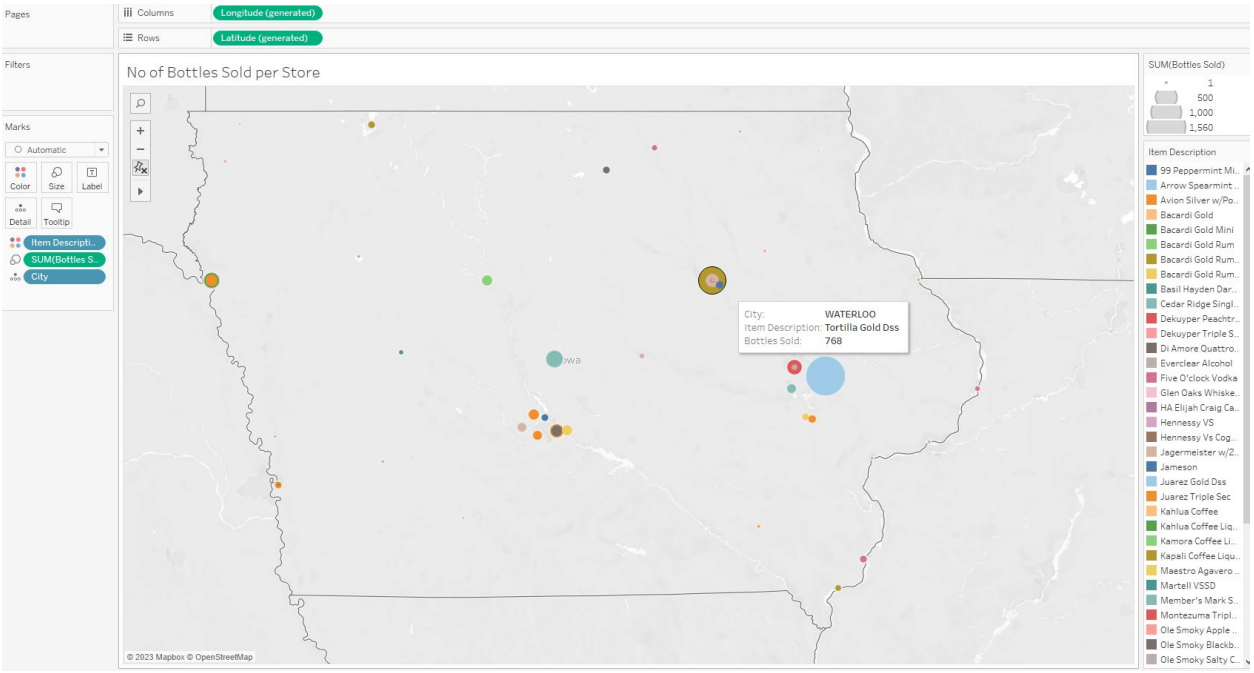


Figure 12: Sold number of bottles per store.

Figure 13 depicts the percentage of total bottles sold per item store (horizontal bar plot, descending order). The corresponding link to this sheet can be found in here.

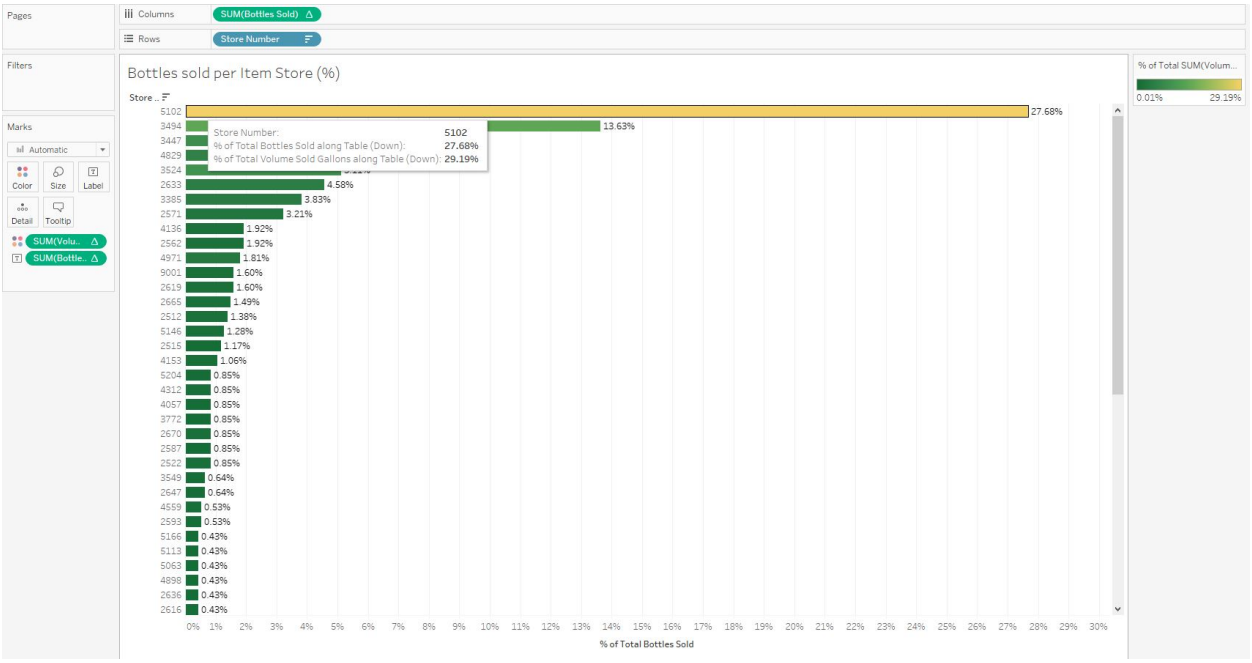


Figure 13: Sold percentage of bottles per store.

Figure 14 demonstrates the percentage of total sales in dollars per item store (circular area marks). The corresponding link to this sheet can be found in here.

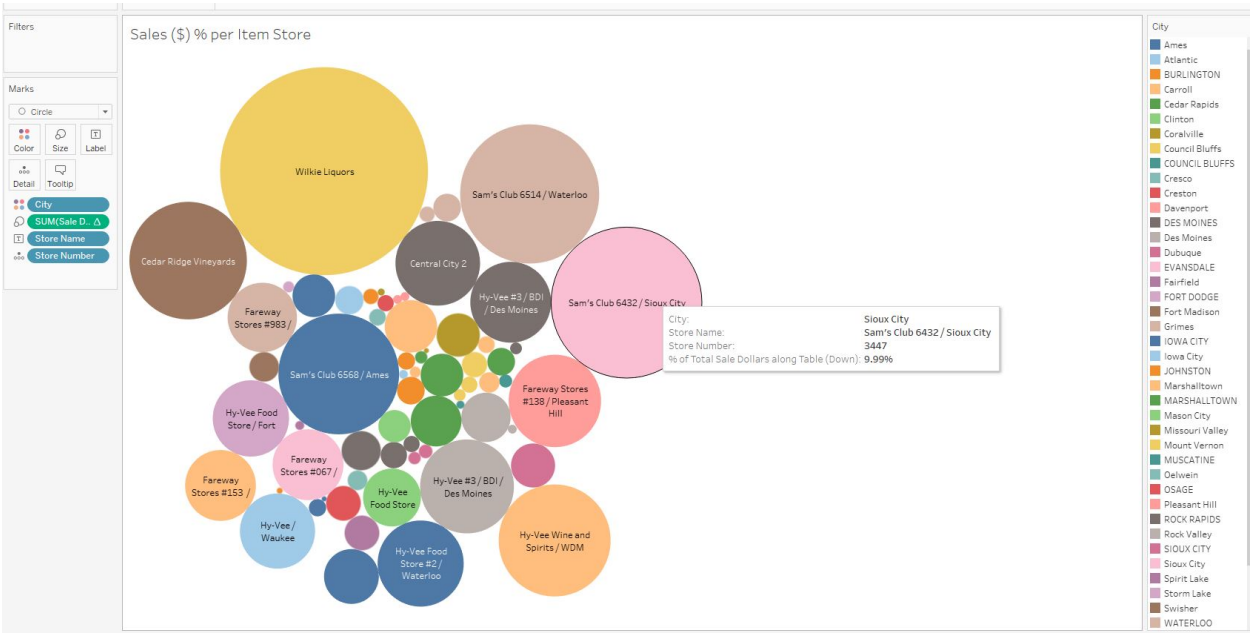


Figure 14: Percentage of dollars sales per store.

Finally, we create a dashboard which displays the whole diagrams/charts displayed in Dashboard, also shown in Fig. 15. Action highlights between different sheets have been added.

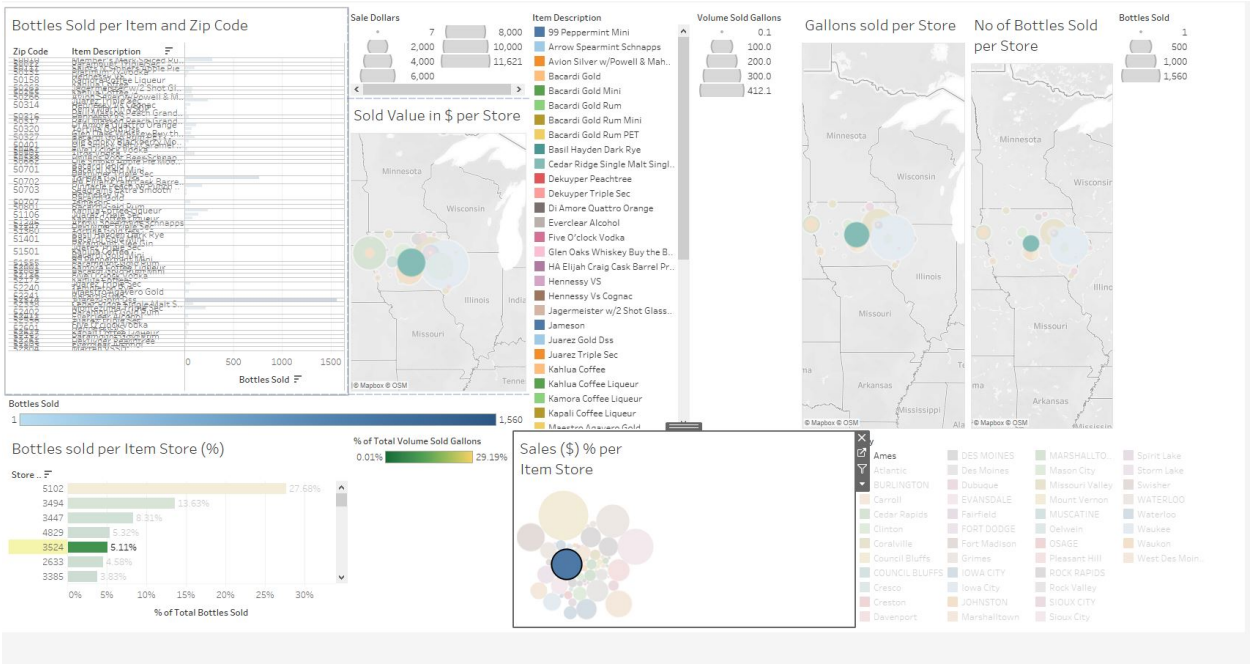


Figure 15: Dashboard