CAPSTONE INTERIM REPORT

Batch details	PGPDSE-FT Chennai July 22
Team members	1. SWETHA R VE 2. NARASHIMMHAN 3. MUTHURAM PANDIAN 4. NANDAKUMAR A K 5. JEEVA ANAND B
Domain of Project	SALES ANALYSIS
Proposed project title	Unlocking Sales Potential in Lowa Liquor through Data Analytics.
Group Number	Team 04
Team Leader	SWETHA R VE
Mentor Name	Pratik Sonar

Date:09/06/2023

Signature of the Mentor

Signature of the Team Leader

TABLE OF CONTENT

1. BUSINESS UNDERSTANDING:

As we know that the liquor sales is one of the worlds biggest business market all around the world where it can generate a large impact on the revenue .
Revenue in the Alcoholic Drinks market amounts to US$1,609.00bn in recent years. The market is expected to grow annually by 5.42% . In global comparison, most revenue is generated in China
In relation to total population figures, per person revenues of US$209.40 are generated.

So that as per the studies and data people all around the world use different types of liquor and each country is getting taxes and other benefits through the sales. There are different regulations in this industry as we know that consumption of alcohol is injuries to the health and people may get addicted to this habit.Keeping all these factors we can analyse the Lowa Liquor sales and different market studies and how to increase the sales by proper marketing and personalised advertisements.

1.1 BUSINESS PROBLEM STATEMENT:

Lowa Liquor is a retail store that specializes in selling various types of alcoholic beverages. The store has been facing a decline in sales over the past year, and the management team is concerned about the reasons behind this decline. The store wants to identify the factors that are contributing to the decline in sales and find ways to improve the sales performance

Business Objective:

The objective of the business is to identify the factors that are causing the decline in sales and develop strategies to increase sales revenue. The business wants to analyze sales data and customer behavior to identify patterns and trends that can help them make informed decisions about how to improve their business operations

1.1 TOPIC SURVEY :

1. Problem understanding:

The problem is that Lowa Liquor, a retail store specializing in selling alcoholic beverages, has experienced a decline in sales over the past year. The management team is concerned about the reasons behind this decline and wants to identify the factors contributing to it.

2. Current solution to the problem:

There is currently no specific solution in place to address the decline in sales at Lowa Liquor. The store may be implementing general strategies such as 5 | P a g e marketing and promotion campaigns, but there is no evidence that these strategies are effective.

3. Proposed solution to the problem:

The proposed solution is to use data analysis and machine learning techniques to identify the factors contributing to the decline in sales and develop strategies to improve sales revenue. This may involve analyzing sales data and customer behavior, identifying patterns and trends, and using this information to make datadriven decisions about pricing, product mix, promotions, and inventory management.

2. DATA UNDERSTANDING:

2.1 DATA DICTIONARY:

S.No	Feature Name	Feature Description
1.	Invoice and item number	Invoice number for the purchased product
2.	Date	Date of the product purchase
3.	Store number	Product sold store number
4.	Store name	Product sold store name
5.	Address	Product sold store Address
6.	City	Product sold store city
7.	Zip code	Product sold store zip code
8.	Store location	Product sold store location
9.	County number	Product sold country number
10.	County	Product sold country number
11.	Category	Category number of Product sold
12.	Category name	Category name of Product sold
13.	Vendor number	Vendor number for the product distributed to the stores
14.	Vendor name	Vendor name for the product
15.	Item number	item number for the product
16.	Item description	Description of the item sold
17.	Pack	Number of bottles in a pack
18.	Bottle volume (ml)	Quantity per bottle
19.	State bottle cost	Cost of the bottle state wise (whole sale)
20.	State bottle retail	Cost of the bottle retail
21.	Bottle sold	Number bottle bought
	Sales in dollar	Price in dollar
	Volume sold in litres	Quantity sold in liters
	Volume sold in gallons	Quantity sold in gallons

2.2 VARIABLE CATEGORIZATION :

Independent variables:

Numerical column: 14

Categorical column: 10

Target variable:

Quantity sold in litres : Numerical

3.DATA PREPROCESSING:

```
df_filter.loc[df_filter['store_name'] == 'Kum & Go 202 / 4Th St Waukee', ['store_name', 'City']] = 'Kum & Go 202','Waukee'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store  2783/ Urband', ['store_name', 'City']] = 'Caseys General Store 27
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1125 / Humest', ['store_name', 'City']] = 'Caseys General Store 11
df_filter.loc[df_filter['store_name'] == 'Kum & Go  532/ West Dsm', ['store_name', 'City']] = 'Kum & Go  532','West Des Moines'
df_filter.loc[df_filter['store_name'] == 'Fareway Stores 703 / Humbolt', ['store_name', 'City']] = 'Fareway Stores 703','Humboldt
df_filter.loc[df_filter['store_name'] == 'D And S Grocery', 'City'] = 'Melcherdallas'
df_filter.loc[df_filter['store_name'] == 'Point Liquor & Tobacco', 'City'] = 'Cedar Rapids'
df_filter.loc[df_filter['store_name'] == 'North American Spirits', 'City'] = 'Urbandale'
df_filter.loc[(df_filter['store_name'] == 'The Secret Cellar') & (df_filter['date'] < '2017-09-12'), 'City'] = 'Swisher'
df_filter.loc[(df_filter['store_name'] == 'The Secret Cellar') & (df_filter['date'] >= '2017-09-12'), 'City'] = 'Shueyville'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store  2560', 'City'] = 'Ames'
df_filter.loc[df_filter['store_name'] == 'Gameday Liquor', 'City'] = 'Glenwood'
df_filter.loc[df_filter['store_name'] == 'Liquor And Grocery Depot', 'City'] = 'Marshalltown'
df_filter.loc[df_filter['store_name'] == 'Express Mart', 'City'] = 'Muscatine'
df_filter.loc[df_filter['store_name'] == 'Av Superstop', 'City'] = 'Des Moines'
df_filter.loc[df_filter['store_name'] == 'U S Gas', 'City'] = 'Des Moines'
df_filter.loc[df_filter['store_name'] == 'River Mart', 'City'] = 'West Burlington'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1548 / Ankeny', 'City'] = 'Ankeny'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 3075 / Ankeny', 'City'] = 'Ankeny'
df_filter.loc[df_filter['store_name'] == 'Hyvee Wine And Spirits / Estherville', 'City'] = 'Estherville'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 3508/ Marsha', 'City'] = 'Marsha'
df_filter.loc[df_filter['store_name'] == 'Fareway Stores 151 / Cedar Rapids', 'City'] = 'Cedar Rapids'
df_filter.loc[df_filter['store_name'] == 'Mmdg Spirits / Ames', 'City'] ='Ames'
df_filter.loc[df_filter['store_name'] == 'Kum & Go 438 / Muscatine', 'City'] = 'Muscatine'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1567 / Anita', 'City'] = 'Anita'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1503 / Tabor', 'City'] = 'Tabor'
df_filter.loc[df_filter['store_name'] == 'Hyvee Food And Drug 6 / Cedar Rapids', 'City'] ='Cedar Rapids'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1365 / Paullina','City'] = 'Paullina'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1680 / Adel', 'City'] = 'Adel'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store  2598/ Pella', 'City'] ='Pella'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1617 / Jefferson', 'City'] = 'Jefferson'
df_filter.loc[df_filter['store_name'] == 'Indy 66 West 929 / Indianola', 'City'] = 'Indianola'
df_filter.loc[df_filter['store_name'] == 'Kum & Go 4098 / Windsor Heights', 'City'] = 'Windsor Heights'
df_filter.loc[df_filter['store_name'] == 'Jeffs Market / Wilton', 'City'] = 'Wilton'
df_filter.loc[df_filter['store_name'] == 'Kum & Go 502 / Iowa City', 'City'] = 'Iowa City'
df_filter.loc[df_filter['store_name'] == 'Kum & Go 201 / Coralville','City'] = 'Coralville'
df_filter.loc[df_filter['store_name'] == 'Kum & Go 768 / Hospers','City'] = 'Hospers'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store  2417/ Newton', 'City'] = 'Newton'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1493 / Van Meter', 'City'] = 'Van Meter'
df_filter.loc[df_filter['store_name'] == 'Caseys General Store 1684 / Emmetsburg', 'City'] = 'Emmetsburg'
df_filter.loc[df_filter['store_name'] == 'Kum & Go 521  /  Coralville', 'City'] ='Coralville'
df_filter.loc[df_filter['store_name'] == 'Kum & Go 540 / Waukee', 'City'] = 'Waukee'
```

```
#Changing category_name based on the Category Number
df_filter.loc[df_filter['category_name'] == 'Single Barrel Bourbon Whiskies', 'Category'] = 1011300.0
df_filter.loc[df_filter['category_name'] == 'Temporary & Specialty Packages', 'Category'] = 1700000.0
df_filter.loc[df_filter['category_name'] == 'Corn Whiskies', 'Category'] = 1011700.0
df_filter.loc[df_filter['category_name'] == 'American Vodkas', ['Category', 'category_name']] = 1031000.0,'American Vodka'
df_filter.loc[df_filter['category_name'] == 'Straight Rye Whiskies', 'Category'] = 1011600.0
df_filter.loc[df_filter['category_name'] == 'Bottled In Bond Bourbon', 'Category'] = 1011500.0
df_filter.loc[df_filter['category_name'] == 'Tennessee Whiskies', 'Category'] = 1011400.0
df_filter.loc[df_filter['category_name'] == 'Single Malt Scotch', 'Category'] = 1012210.0
df_filter.loc[df_filter['category_name'] == 'Irish Whiskies', 'Category'] = 1012300.0
df_filter.loc[df_filter['category_name'] == 'Flavored Gins', 'Category'] = 1041000.0
df_filter.loc[df_filter['category_name'] == 'Cocktails /Rtd', 'Category'] = 1070000.0
df_filter.loc[df_filter['category_name'] == 'Spiced Rum', 'Category'] = 1062310.0
df_filter.loc[df_filter['category_name'] == 'Imported Vodkas', ['Category', 'category_name']] = 1032000.0,'American Vodka'
df_filter.loc[df_filter['category_name'] == 'Imported Vodka', 'Category'] = 1032000.0
df_filter.loc[df_filter['category_name'] == 'Flavored Rum', 'Category'] = 1062500.0
df_filter.loc[df_filter['category_name'] == 'Cocktails /Rtd', 'category_name'] = 'Cocktails / Rtd'
df_filter.loc[df_filter['category_name'] == 'Coffee Liqueurs', 'Category'] = 1081030.0
df_filter.loc[df_filter['category_name'] == 'American Dry Gins', 'Category'] = 1041100.0
df_filter.loc[df_filter['category_name'] == 'American Vodka', 'Category'] = 1031000.0
df_filter.loc[df_filter['category_name'] == 'American Sloe Gins', 'Category'] = 1041300.0
df_filter.loc[df_filter['category_name'] == 'American Cordials & Liqueurs', 'Category'] = 1081000.0
df_filter.loc[df_filter['category_name'] == 'Imported Distilled Spirits Specialty', 'Category'] = 1092000.0
df_filter.loc[df_filter['category_name'] == 'Imported Cordials & Liqueur', ['Category', 'category_name']] = 1082000.0,'Imported C
df_filter.loc[df_filter['category_name'] == 'American Cordials & Liqueur', ['Category', 'category_name']] = 1081000,'American Cor
df_filter.loc[df_filter['category_name'] == 'Imported Distilled Spirit Specialty', ['Category', 'category_name']] = 1092000.0,'Im
df_filter.loc[df_filter['category_name'] == 'Temporary &  Specialty Packages', 'category_name'] = 'Temporary & Specialty Packages
df_filter.loc[df_filter['category_name'] == 'Vodka Flavored', 'category_name'] = 'American Flavored Vodka'
df_filter.loc[df_filter['category_name'] == 'American Vodka Flavored', 'category_name'] = 'American Flavored Vodka'
df_filter.loc[(df_filter['category_name'] == 'Tequila')|(df_filter['category_name'] == 'Mixto'), 'category_name'] = 'Mixto Tequil
df_filter.loc[df_filter['category_name'] == 'Imported Vodka  Misc', 'category_name'] = 'Imported Flavored Vodka'
df_filter.loc[df_filter['category_name'] == 'American Gins', 'category_name'] = 'Flavored Gins'
df_filter.loc[(df_filter['category_name'] == 'Apricot Brandies')|(df_filter['category_name'] == 'American Brandies'), 'category_n
df_filter.loc[(df_filter['category_name'] == 'Jamaica Rum')|(df_filter['category_name'] == 'Gold Rum'), 'category_name'] = 'Jamai
df_filter.loc[(df_filter['category_name'] == 'Puerto Rico & Virgin Islands Rum')|(df_filter['category_name'] == 'White Rum'), 'ca
df_filter.loc[(df_filter['category_name'] == 'Triple Sec') & (df_filter['Category'] == 1081400.0), 'category_name'] = 'American S
df_filter.loc[(df_filter['category_name'] == 'American Schnapps') & (df_filter['Category'] == 1081400.0), 'category_name'] = 'Ame
```

2.3 NULL VALUE TREATMENT:

Null value treatment is essential to building most of the commonly used machine learning classification models such as logistic regression, decision tree, KNN, and others. To infer that we have used isnull() function the null values from the dataset.

```
In [37]: df1.isna().sum()
```

```
Out[37]: invoice_and_item_number         0
         date                            0
         store_number                    0
         store_name                      0
         address                        64
         city                           64
         zip_code                       64
         store_location             117706
         county_number                  64
         county                         64
         category                        0
         category_name                   0
         vendor_number                   3
         vendor_name                     3
         item_number                     0
         item_description                0
         pack                            0
         bottle_volume_ml                0
         state_bottle_cost               0
         state_bottle_retail             0
         bottles_sold                    0
         sale_dollars                    0
         volume_sold_liters              0
         volume_sold_gallons             0
```

From the above figure, it is evident that the maximum of missing value is **117706** which is observed only in store location column. Since we have store address ,city name and zip code we will be dropping the column store location.

Missing values in columns **address,city ,zip code,**county number and county were represented as null . We had replaced it with NaN for the ease of processing.

2.4 DISTRIBUTION OF VARIABLES:

The Lowa Liquor dataset which we had selected have 1048575 rows and 24 columns. The data consists of Numerical and Categorical data. While further analyzing the data we find that there is 14 numerical data and 10 categorical data. We found that there is 8 columns which have the presence of null variable in which 7 of them can be negligible but the column store there is about 117706 null values which need to be treated or the column need to be ruled out. The numerical features have different scales, which may be a problem for some machine learning algorithms. The features should be rescaled to have similar scale

Distribution of Numeric Variables Original Data:

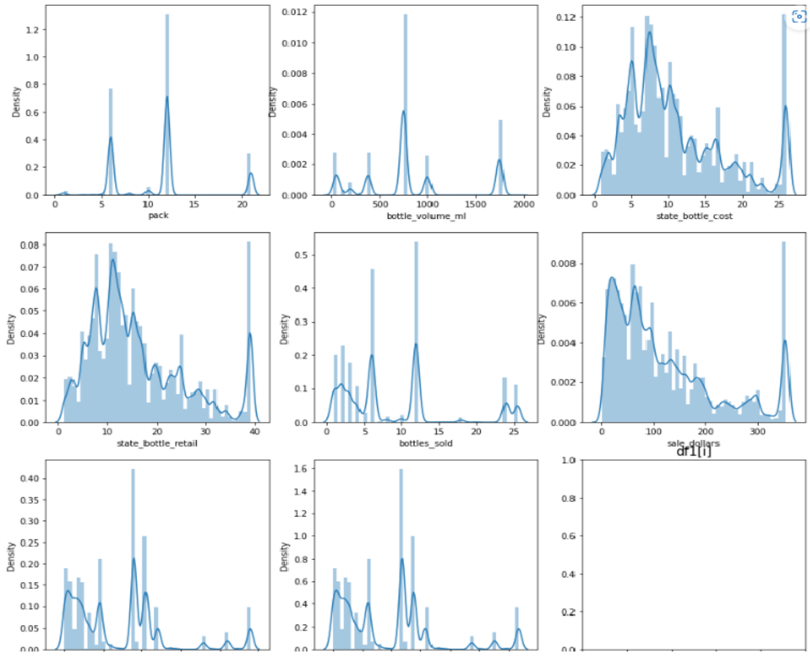
As we are analyzing the sales we will be mainly dealing with the numerical data more than the categorical one. So that as a primary step we will be sorting the numerical columns separately for analyzing the data.

Mainly we are taking 8 numerical columns for the analysis of the sales and the distribution of the numerical variables is here:

Wapello	2496
Webster	821

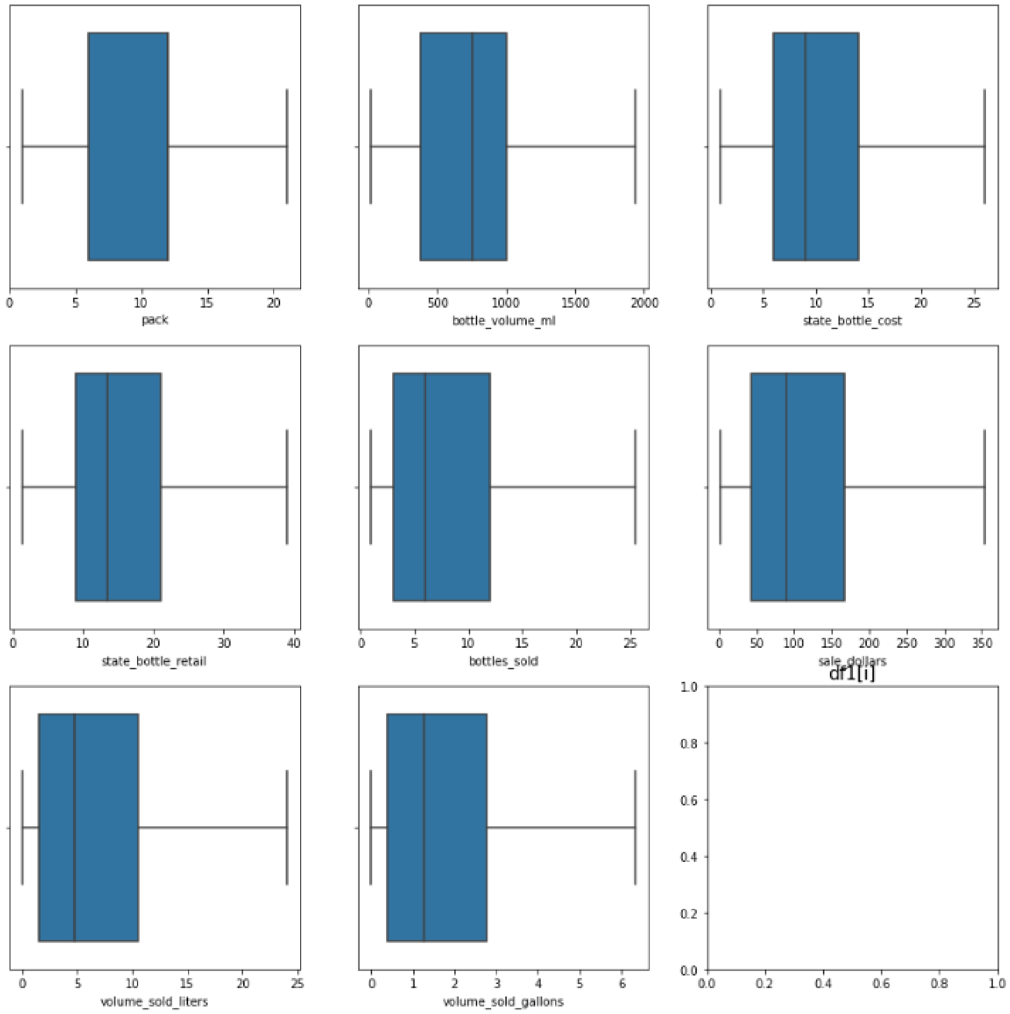
119 rows × 1 columns

In [105]:
```
fig,axs = plt.subplots(3,3,figsize =(15,15))
for i, subplots in zip(num_colums,axs.flatten()):
    sns.distplot(df1[i],ax=subplots)
    plt.title('df1[i]',fontsize = 15)
plt.show()
```



Outliers of Numeric Variables Original Data:

```
In [103]: fig,axs = plt.subplots(3,3,figsize =(15,15))
for i, subplots in zip(num_colums,axs.flatten()):
    sns.boxplot(df1[i],ax=subplots)
    plt.title('df1[i]',fontsize = 15)
plt.show()
```



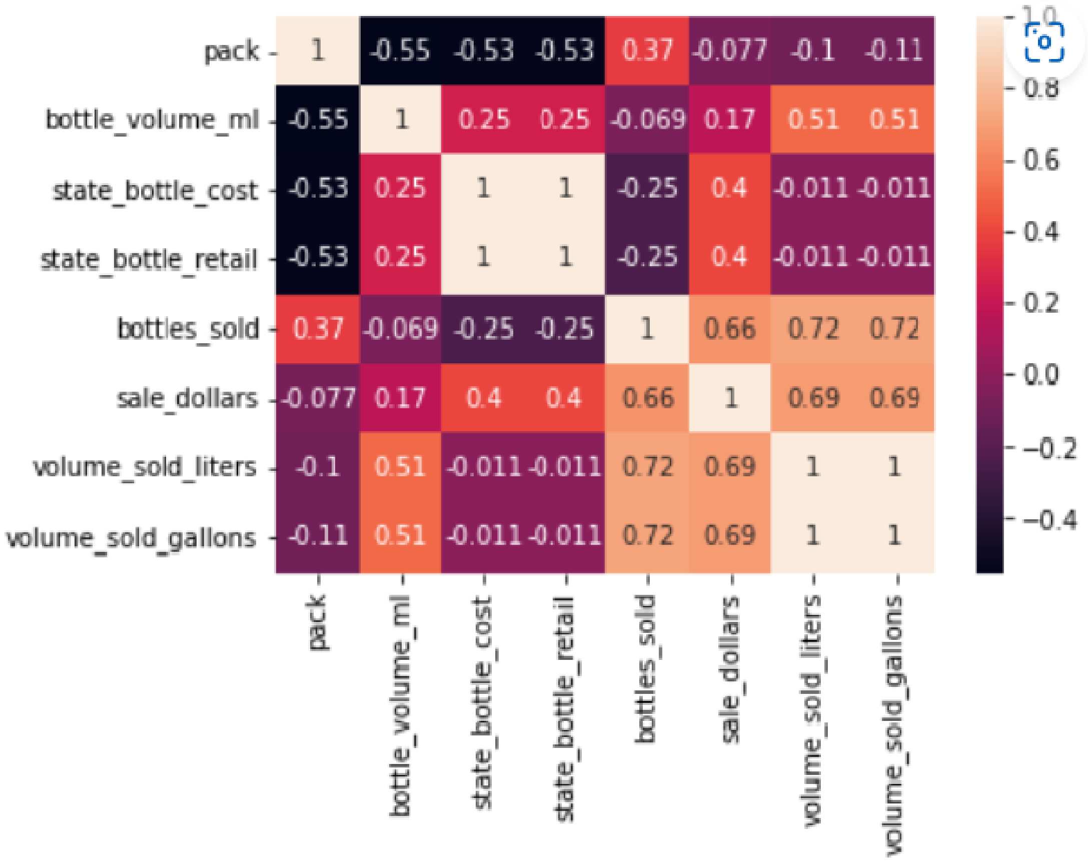
2.5 Correlation between the variables.

As we are considering the dependent numerical variable we need to look into the correlation between the variables for better analysis.

Here is the heat map :

```
sns.heatmap(df1.corr(),annot=True)
```

```
<AxesSubplot:>
```

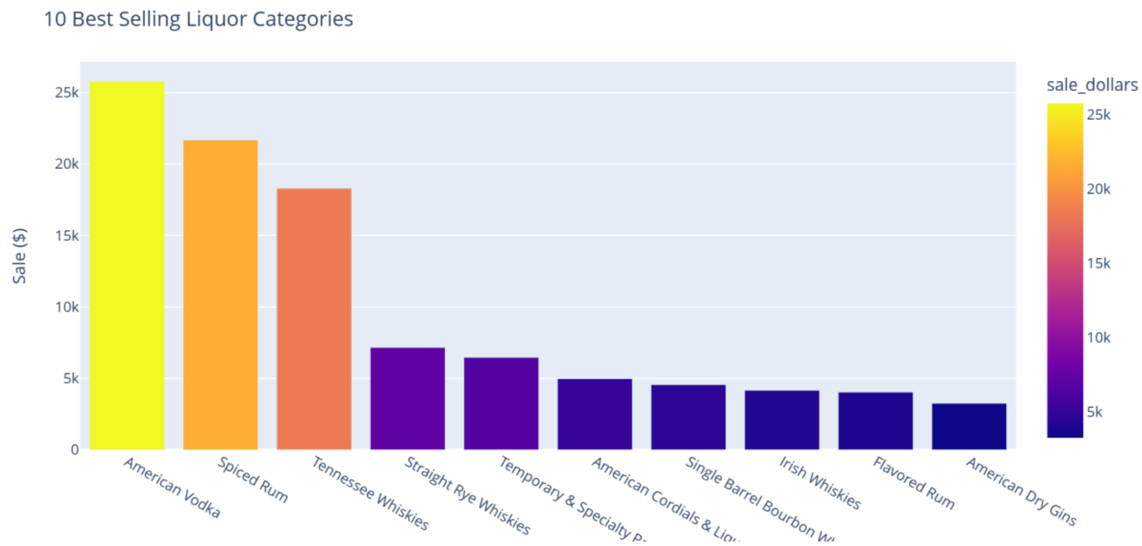


Vendor number, store number, county number , pre_icu_los_days – Since these feautures have no impact on the future prediction of the volume of liquor sold we will be dropping this feautures(store name,vendor name,county name is already mentioned in dataset)

3.Explore Data Analysis:

10 liquor categories with highest sale

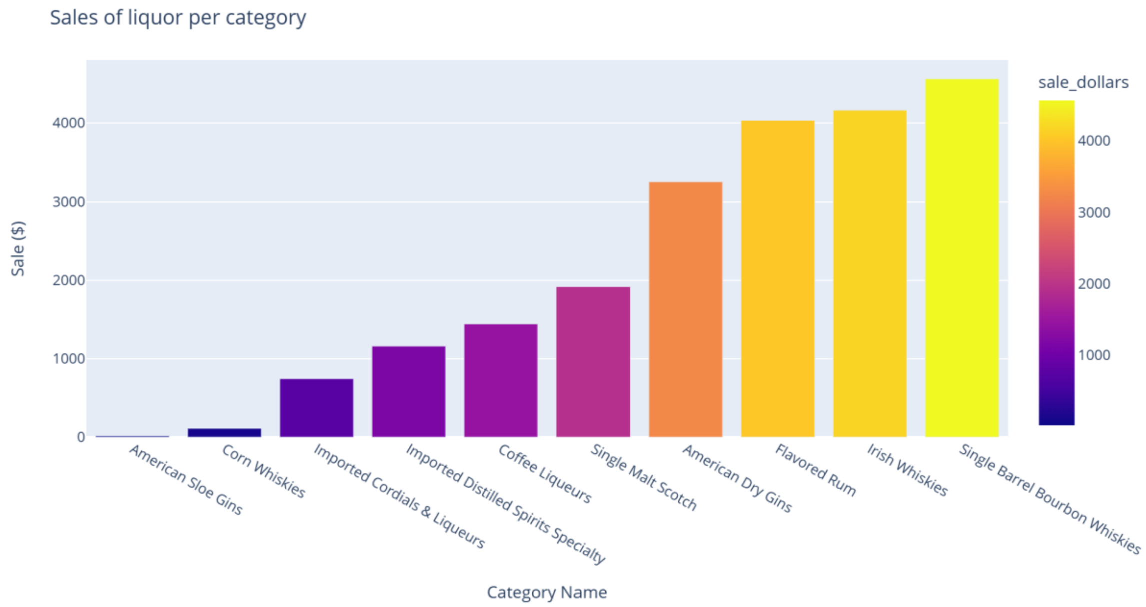
```
import plotly.graph_objects as go
import plotly.express as px
import plotly as py

best10 = df_filter.groupby(['category_name','pack','date'])['sale_dollars'].sum().groupby(['category_name','pack']).max().sort_va

best10_plot = px.bar(best10.head(10),x=best10['category_name'].head(10), y='sale_dollars',color='sale_dollars')
best10_plot.update_layout(
    title="10 Best Selling Liquor Categories",
    xaxis_title="Category Name",
    yaxis_title="Sale ($)")
best10_plot.show()
```



Above Plot shows top 10 best selling liquor, In that we can see American Vodka contributing more.

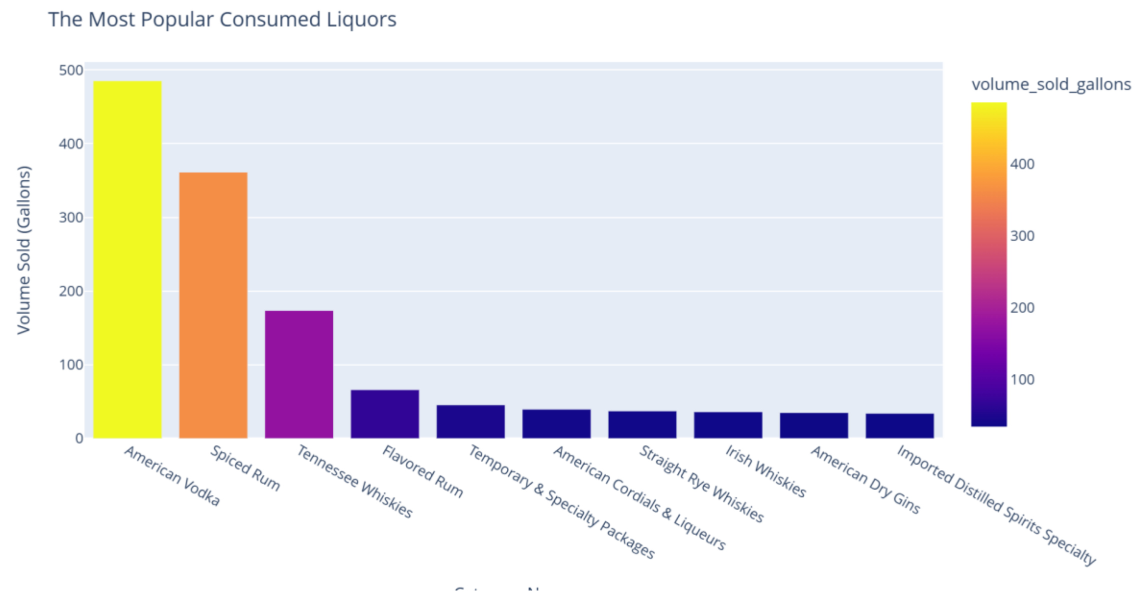
10 liquor categories with lowest sale

```
lowest10 = df_filter.groupby(['category_name','pack','date'])['sale_dollars'].sum().groupby(['category_name','pack']).max().sort_

lowest10_plot = px.bar(lowest10.head(10),x=lowest10['category_name'].head(10), y='sale_dollars',color='sale_dollars')
lowest10_plot.update_layout(
    title="Sales of liquor per category",
    xaxis_title="Category Name",
    yaxis_title="Sale ($)")
lowest10_plot.show()
```



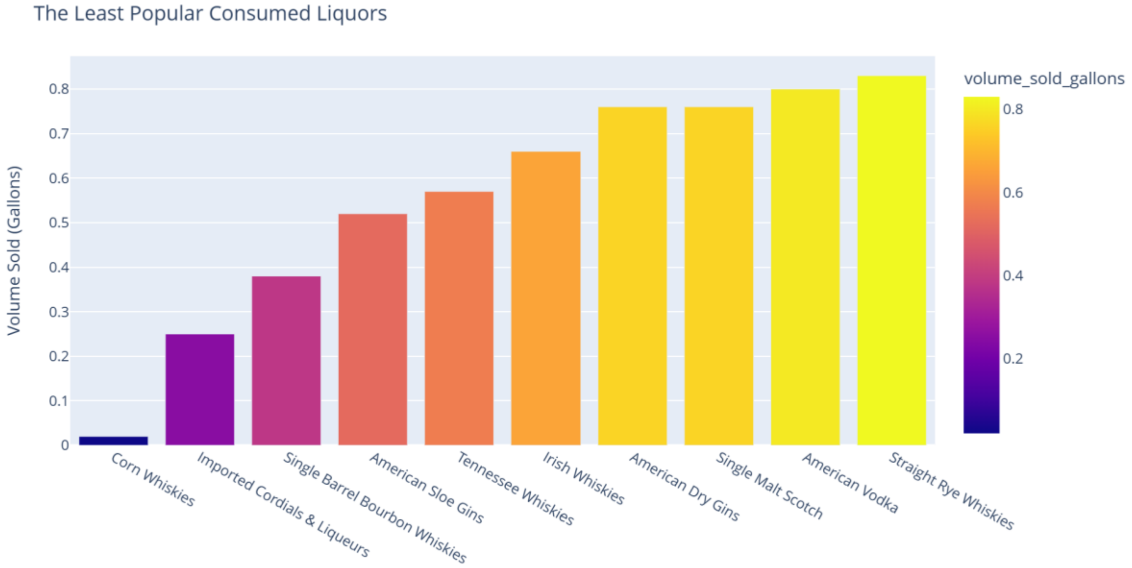
The Most Popular Consumed Liquors

```
mostpopular = df_filter.groupby(['category_name','pack','date'])['volume_sold_gallons'].sum().groupby(['category_name','pack']).n

mostpopular_plot = px.bar(mostpopular.head(10),x=mostpopular['category_name'].head(10), y='volume_sold_gallons',color='volume_sol
mostpopular_plot.update_layout(
    title="The Most Popular Consumed Liquors",
    xaxis_title='Category Name',
    yaxis_title="Volume Sold (Gallons)")
mostpopular_plot.show()
```



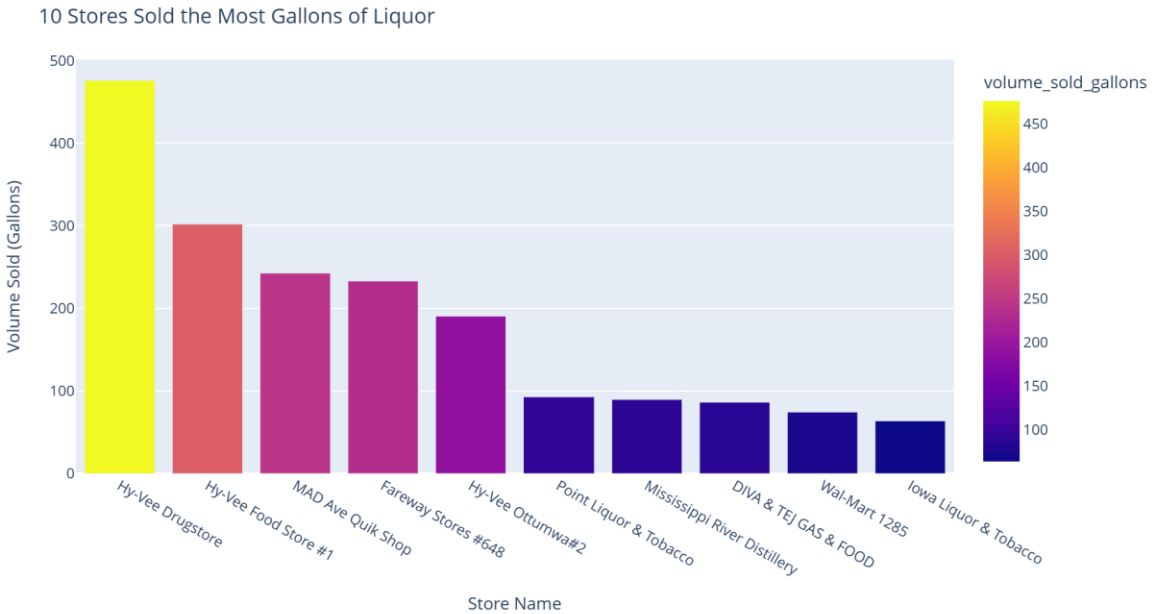
The Least Popular Consumed Liquors

```
leastpopular = df_filter.groupby(['category_name','pack','date'])['volume_sold_gallons'].sum().groupby(['category_name','pack']).

leastpopular_plot = px.bar(leastpopular.head(10),x=leastpopular['category_name'].head(10), y='volume_sold_gallons',color='volume_
leastpopular_plot.update_layout(
    title="The Least Popular Consumed Liquors",
    xaxis_title='Category Name',
    yaxis_title="Volume Sold (Gallons)")
leastpopular_plot.show()
```



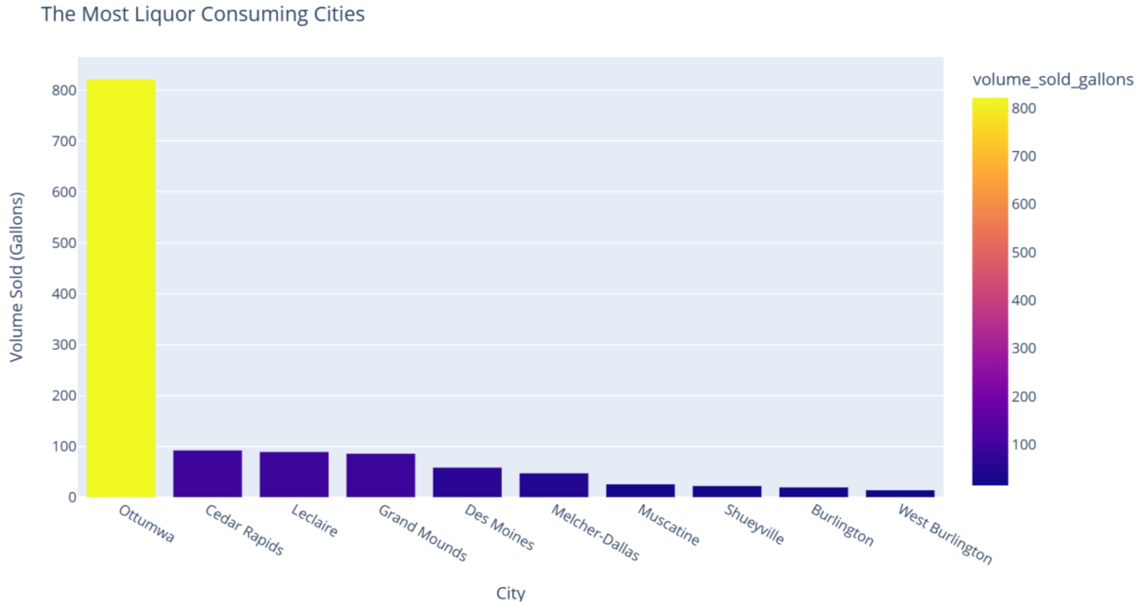
10 Stores Sold the Most Gallons of Liquor

```
stores = df_filter.groupby(['store_name','pack','date'])['volume_sold_gallons'].sum().groupby(['store_name','pack']).max().sort_

stores_plot = px.bar(stores.head(10),x=stores['store_name'].head(10), y='volume_sold_gallons',color='volume_sold_gallons')
stores_plot.update_layout(
    title="10 Stores Sold the Most Gallons of Liquor",
    xaxis_title='Store Name',
    yaxis_title="Volume Sold (Gallons)")
stores_plot.show()
```



The Most Liquor Consuming Cities

```
cities = df_filter.groupby(['city','pack','date'])['volume_sold_gallons'].sum().groupby(['city','pack']).max().sort_values().grou
cities_plot = px.bar(cities.head(12),x=cities.city.head(12), y='volume_sold_gallons',color='volume_sold_gallons')
cities_plot.update_layout(
    title="The Most Liquor Consuming Cities",
    xaxis_title='City',
    yaxis_title="Volume Sold (Gallons)")
cities_plot.show()
```

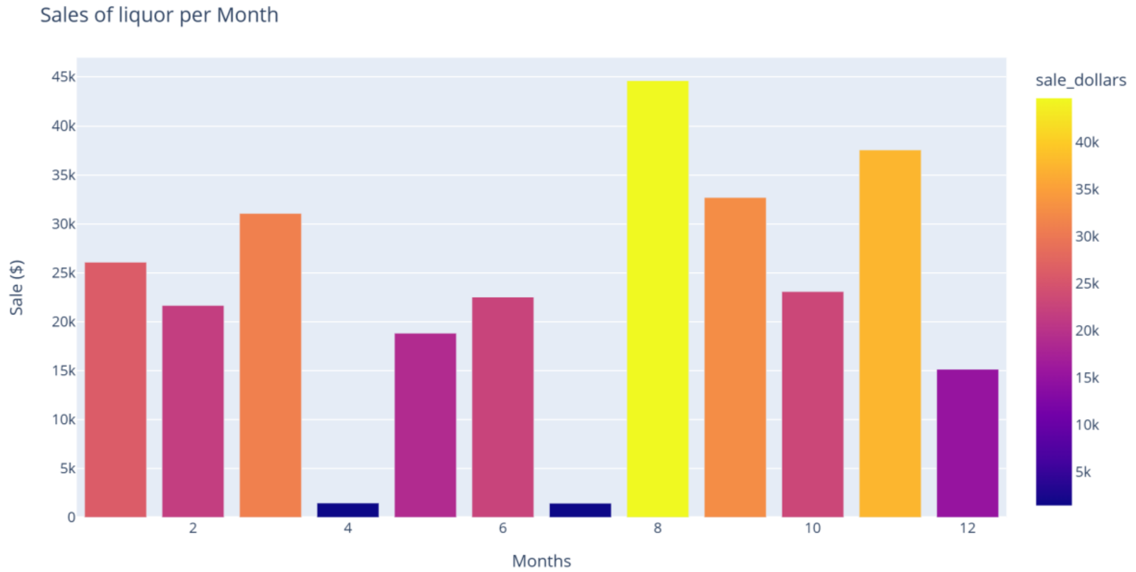


The Highest Profit Contributor Liquor Categories

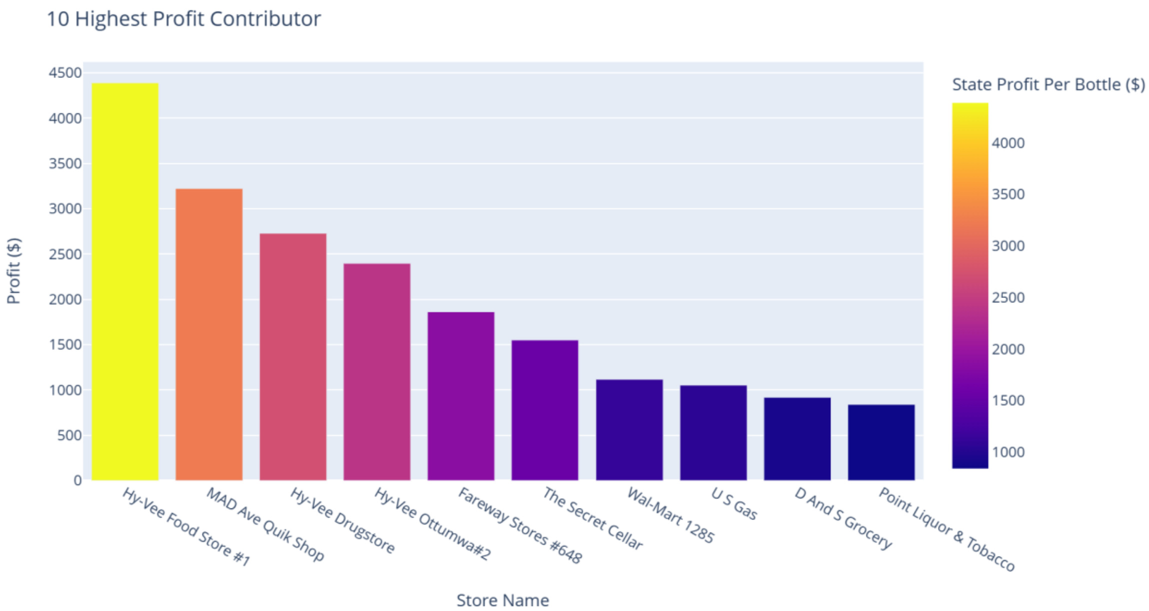
```
profit = df_filter.groupby(['category_name','pack','date'])['State Profit Total Bottle ($)'].sum().groupby(['category_name','pack

profit_plot = px.bar(profit.head(10),x=profit['category_name'].head(10), y='State Profit Total Bottle ($)',color='State Profit To
profit_plot.update_layout(
    title="The Highest Profit Contributor Liquor Categories",
    xaxis_title='Category Name',
    yaxis_title="State Profit Total Bottle ($)"
)
profit_plot.show()
```

Sales of Liquor per Month

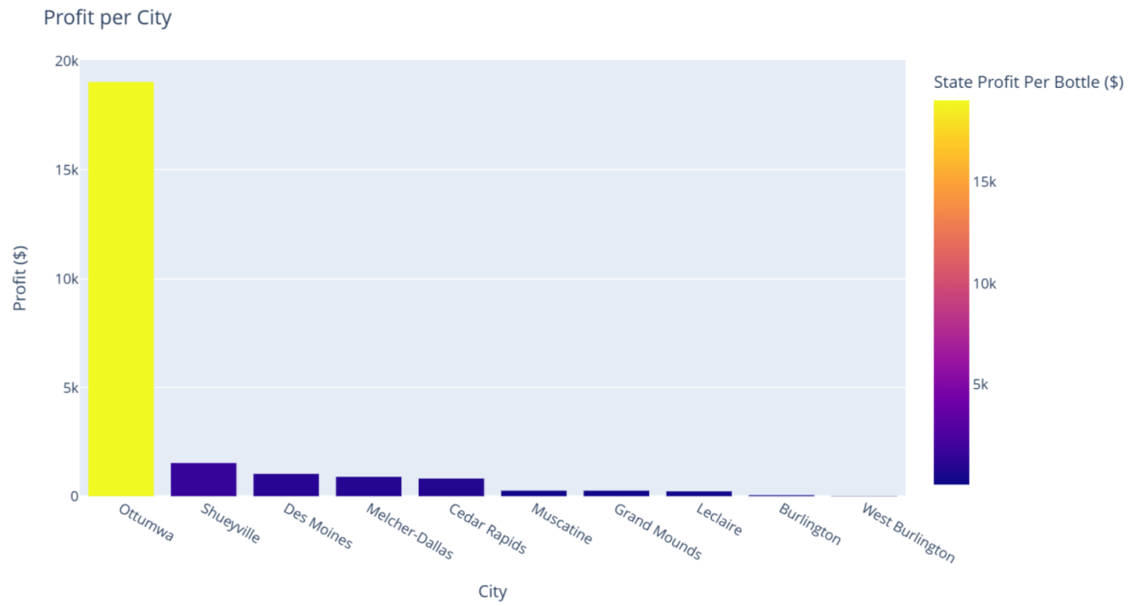
```
df_filter['Month'] = df_filter['date'].dt.month
df_filter['Year'] = df_filter['date'].dt.year
selling = df_filter.groupby(['Month','pack','date'])['sale_dollars'].sum().groupby(['Month','pack']).max().sort_values().groupby(
selling = pd.DataFrame(selling)

selling_plot = px.bar(selling,x=selling.Month, y='sale_dollars',color='sale_dollars')
selling_plot.update_layout(
    title="Sales of liquor per Month",
    xaxis_title='Months',
    yaxis_title="Sale ($)")
selling_plot.show()
```




```
profitpercity= (df_filter.groupby('store_name')['State Profit Per Bottle ($)'].sum().to_frame().sort_values('State Profit Per Bot
profitpercity = pd.DataFrame(profitpercity)

profitpercity_plot = px.bar(profitpercity.head(10),x=profitpercity['store_name'].head(10), y='State Profit Per Bottle ($)',color=
profitpercity_plot.update_layout(
    title="10 Highest Profit Contributor",
    xaxis_title='Store Name',
    yaxis_title="Profit ($)")
profitpercity_plot.show()
```



```
profitpercity= (df_filter.groupby('city')['State Profit Per Bottle ($)'].sum().to_frame().sort_values('State Profit Per Bottle ($
profitpercity = pd.DataFrame(profitpercity)

profitpercity_plot = px.bar(profitpercity.head(10),x=profitpercity.city.head(10), y='State Profit Per Bottle ($)',color='State Pr
profitpercity_plot.update_layout(
    title="Profit per City",
    xaxis_title='City',
    yaxis_title="Profit ($)")
profitpercity_plot.show()
```



4. Data Pre-processing for Model Building

Dropping the columns which is not significant for model building.

```
In [5]: df = df.drop(['invoice_and_item_number', 'date','store_name','address','zip_code', 'store_location', 'county_number',
                 'vendor_number','item_number','category','volume_sold_gallons'], axis = 1)
```

```
In [6]: df.info()
```

```
In [8]: df1 = df.drop(['store_number','city','county','category_name','vendor_name','item_description'],axis = 1)
```

```
In [9]: df1.describe()
```

Out[9]:

	pack	bottle_volume_ml	state_bottle_cost	state_bottle_retail	bottles_sold	sale_dollars	volume_sold_liters
count	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06
mean	1.198848e+01	8.248567e+02	1.126751e+01	1.690189e+01	1.186573e+01	1.610171e+02	9.385574e+00
std	7.881474e+00	5.229357e+02	1.129648e+01	1.694280e+01	3.148000e+01	4.850953e+02	3.787383e+01
min	1.000000e+00	2.000000e+01	8.900000e-01	1.340000e+00	1.000000e+00	1.340000e+00	2.000000e-02
25%	6.000000e+00	3.750000e+02	6.000000e+00	9.000000e+00	3.000000e+00	4.200000e+01	1.500000e+00
50%	1.200000e+01	7.500000e+02	8.980000e+00	1.347000e+01	6.000000e+00	8.952000e+01	4.800000e+00
75%	1.200000e+01	1.000000e+03	1.400000e+01	2.100000e+01	1.200000e+01	1.665000e+02	1.050000e+01
max	6.000000e+01	5.250000e+03	1.949020e+03	2.923530e+03	3.780000e+03	5.643000e+04	6.615000e+03

```
In [10]: df1.isnull().sum()
```

```
Out[10]: pack                   0
bottle_volume_ml       0
state_bottle_cost      0
state_bottle_retail    0
bottles_sold           0
sale_dollars           0
volume_sold_liters     0
dtype: int64
```

Data is Slightly skewed

```
scaler = StandardScaler()
ss =  scaler.fit_transform(df1)
df_ss = pd.DataFrame(ss, columns= df1.columns)
```

```
df_ss.skew()
```

```
pack                    0.689615
bottle_volume_ml        0.559649
state_bottle_cost       0.938417
state_bottle_retail     0.938333
bottles_sold            1.123899
sale_dollars            1.061998
volume_sold_liters      1.161543
dtype: float64
```

Statistical test on dependent variable

```
stat, p = stats.shapiro(df1['volume_sold_liters'])
alpha = 0.05  # significance level

print("Shapiro-Wilk test statistic:", stat)
print("p-value:", p)

if p < alpha:
    print("The data is normally distributed.")
else:
    print("The data is not normally distributed.")
```

```
Shapiro-Wilk test statistic: 0.8556922674179077
p-value: 0.0
The data is normally distributed.
```

Splitting the Test and Train data (80:20)

```
X = df_ss.drop(['volume_sold_liters'],axis =1)
y= df_ss.volume_sold_liters
```

```
X = sm.add_constant(X)
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=1,test_size=0.2)
```

1.2 Base Model

Fitting the base model using OLS method

```
model_scaled_ss = sm.OLS(y_train,X_train).fit()
model_scaled_ss.summary()
```

OLS Regression Results

Dep. Variable:	volume_sold_liters	**R-squared:**	0.864
Model:	OLS	**Adj. R-squared:**	0.864
Method:	Least Squares	**F-statistic:**	8.853e+05
Date:	Fri, 09 Jun 2023	**Prob (F-statistic):**	0.00
Time:	14:05:37	**Log-Likelihood:**	-3.5445e+05
No. Observations:	838860	**AIC:**	7.089e+05
Df Residuals:	838853	**BIC:**	7.090e+05
Df Model:	6		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	3.467e-05	0.000	0.086	0.931	-0.001	0.001
pack	-0.1115	0.001	-190.315	0.000	-0.113	-0.110
bottle_volume_ml	0.4776	0.000	969.835	0.000	0.477	0.479
state_bottle_cost	-0.0810	0.027	-2.992	0.003	-0.134	-0.028
state_bottle_retail	-0.1291	0.027	-4.767	0.000	-0.182	-0.076
bottles_sold	0.5046	0.001	608.904	0.000	0.503	0.506
sale_dollars	0.3537	0.001	412.285	0.000	0.352	0.355

Omnibus:	84645.774	**Durbin-Watson:**	1.999
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	663140.944
Skew:	-0.135	**Prob(JB):**	0.00
Kurtosis:	7.347	**Cond. No.**	160.

1.3 Fitting Multiple Regression Model:

```
MLR_Score_Card = MLR_Score_Card.sort_values('Test_RMSE').reset_index(drop = True)
MLR_Score_Card.style.highlight_min(color = 'lightblue', subset = 'Test_RMSE')
```

	Model_Name	Alpha	l1-ratio	R-Squared	Adj. R-Squared	Test_RMSE	Train_RMSE	Test_MAPE
0	Random Forest Regression	-	-	0.999167	0.999167	0.033200	0.028900	0.163628
1	decision tree Regression	-	-	0.999179	0.999179	0.033400	0.028600	0.159738
2	XGB Regression	-	-	0.999072	0.999072	0.034000	0.030500	0.679226
3	Stacking Regression	-	-	0.998826	0.998826	0.037000	0.034200	0.169593
4	Linear Regression (Standard Scaller)	-	-	0.863621	0.863620	0.369000	0.369200	38.580727
5	Ridge Regression (with alpha = 1)	1	-	0.863621	0.863620	0.369000	0.369200	38.580702
6	Ridge Regression (with alpha = 2)	2	-	0.863621	0.863620	0.369000	0.369200	38.580676
7	Ridge Regression (using GridSearchCV)	100	-	0.863621	0.863620	0.369000	0.369200	38.577992
8	Lasso Regression (using GridSearchCV)	0.000000	-	0.863618	0.863617	0.369000	0.369200	38.582514
9	Elastic Net Regression (using GridSearchCV)	0.001000	0.000100	0.863620	0.863619	0.369000	0.369200	38.556776
10	Linear Regression SGD	-	-	0.863352	0.863351	0.369300	0.369600	38.398406
11	Lasso Regression	0.01	-	0.862812	0.862810	0.370200	0.370300	38.195035
12	Elastic Net Regression	0.1	0.01	0.859033	0.859032	0.375400	0.375400	36.652401
13	Ada Boost Regression	-	-	0.832122	0.832121	0.410700	0.409600	57.784979
14	Decision Tree (Tunned Parameter)	-	-	0.370714	0.370709	0.794300	0.793100	87.704578

1.4 CONCLUSION:

Based on the Above information, the random forest regression model seems to have achieved excellent performance based on several evaluation metrics:

1. R-squared: The R-squared value of 0.99916 indicates that the model explains approximately 99.916% of the variance in the target variable. A high R-squared value suggests that the model fits the data very well, and the majority of the variability in the target variable is captured by the model.

2. Adjusted R-squared: The adjusted R-squared value of 0.99916 is identical to the R-squared value in this case. This suggests that the model contains no unnecessary variables or overfitting issues, as the adjusted R-squared is usually lower than the R-squared when there are excessive variables in the model.

3. Test RMSE: The test root mean squared error (RMSE) of 0.0332 indicates that, on average,

the model's predictions have an error of approximately 0.0332 units when applied to unseen test data. A lower RMSE value suggests better predictive performance, so the provided RMSE value is relatively low.

4. Train RMSE: The train RMSE of 0.0289 represents the average error of the model's predictions on the training data. A lower train RMSE suggests that the model is fitting the training data well, with small discrepancies between the predicted values and the actual values.

5. MAPE: The Mean Absolute Percentage Error (MAPE) of 0.163 indicates the average percentage difference between the predicted and actual values. A lower MAPE indicates better accuracy, and the provided MAPE value is relatively low.

Based on these metrics, the random forest regression model appears to be performing exceptionally well, demonstrating a high level of accuracy and predictive power.

Hence RANDOM FOREST REGRESSOR is predicting ‘Volume sold litre’ WELL compare to other Regression model.