Introduction

The premise of this project is based on the fact that it often becomes imperative for a business to understand the trends and patterns in sales to drive customer satisfaction and increase sales which would increase the revenue generated through sales. Hence sales analysis is a vital activity that would aid a business in generating insights from sales data. The idea of the project is to generate insights from sales data, trends in sales, which product sells best, which are the top performing products and which underperform, and which combination of products gives maximum revenue, and build a model that analyses the purchasing pattern of consumers so that "Store" can achieve maximum profits. The intention of the project is to implement the apriori algorithm to generate association rules for a frequent itemset and justify the findings by computing three important metrics like support, confidence, and lift.

The project follows the general steps towards a data analytics project involving data acquisition, data preparation, information modeling, and visualization.

Data Acquisition

The dataset titled “Sales Product Data” was acquired from Kaggle, and contained data on product sales like quantity ordered, price of the commodity sold, purchase address, and order date. This dataset was taken on every month throughout the year 2019.

The initial hypothesis is that consumers tend to purchase a combination of products that are similar in characteristics to their liking. Thus, the objective of the store is to market similar products together so that the consumer buys the combination of products, which increases customer satisfaction and in retrospect sales, which would increase profits.

The objective of the project would be to use Python to perform the initial data processing, visualize data using Tableau and implement the apriori algorithm to generate association rules. This would help the “store” to market products together so that the customers see, that Product A is frequently bought with Product B and thus allowing the buyer to make a purchase if they intend to.

The data had both numerical and categorical data. The numerical data was the selling price and the quantity ordered. The categorical data was the type of product sold. There was also data about the consumer like billing address and shipping address from which one can analyze which city or state had the most purchases made in the year or in a given month.

On initial inspection of the dataset, there was a need to combine each month's dataset to get the metrics for the entire year. There was a date format issue where the date format was inconsistent in each record. Additionally, there were missing values that either had to be imputed or discarded.

Key questions and hypothesis

The overall goal of the project was to perform exploratory data analysis and represent the visualizations in Tableau and build a model using the apriori algorithm to generate association rules which aided in finding and recommending products to the consumer so that they buy the related product based on their interest and liking. EDA is performed to find answers to the below initial set of questions

1. What was the best month for sales?

2. Which product or products were the most sold in the month?
3. Which City had the most sales in the year?
4. Which State had the most sales in the year?
5. Should products be sold together to increase sales? If so, how many products can be sold together?
6. What are the top 5 products that consumers tend to buy?
7. What is the peak hour when the orders are placed?

The initial hypothesis is that consumers tend to purchase a combination of products that are similar in characteristics to their liking. Thus, the objective of the store is to market similar products together so that the consumer buys the combination of products, which increases customer satisfaction and in retrospect sales, which would increase profits.

Data Preparation and Processing

The dataset obtained from Kaggle had data for the entire year in 2019 and monthly sales data. In this project, the dataset for yearly sales in 2019 is used. The downloaded dataset was in CSV (Comma Separated Value) format and on initial inspection, there was a date format issue where the date format was inconsistent in each record. Additionally, there were missing values that were discarded.

```
all_months_sales.head()
```

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	236670	Wired Headphones	2	11.99	08/31/19 22:21	359 Spruce St, Seattle, WA 98101
1	236671	Bose SoundSport Headphones	1	99.99	08/15/19 15:11	492 Ridge St, Dallas, TX 75001
2	236672	iPhone	1	700.0	08/06/19 14:40	149 7th St, Portland, OR 97035
3	236673	AA Batteries (4-pack)	2	3.84	08/29/19 20:59	631 2nd St, Los Angeles, CA 90001
4	236674	AA Batteries (4-pack)	2	3.84	08/15/19 19:53	736 14th St, New York City, NY 10001

Source: Jupyter Notebook running Python 3

Figure 1 The first five rows of the imported dataset

```
all_months_sales.tail()
```

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
186845	319666	Lightning Charging Cable	1	14.95	12/11/19 20:58	14 Madison St, San Francisco, CA 94016
186846	319667	AA Batteries (4-pack)	2	3.84	12/01/19 12:01	549 Willow St, Los Angeles, CA 90001
186847	319668	Vareebadd Phone	1	400	12/09/19 06:43	273 Wilson St, Seattle, WA 98101
186848	319669	Wired Headphones	1	11.99	12/03/19 10:39	778 River St, Dallas, TX 75001
186849	319670	Bose SoundSport Headphones	1	99.99	12/21/19 21:45	747 Chestnut St, Los Angeles, CA 90001

Source: Jupyter Notebook running Python 3

Figure 2 The last five rows of the imported dataset

1. **Clean Data**

The following libraries were used in Python - Pandas, Numpy, Matplotlib, and Seaborn to inspect and visualize the data. Once the dataset was imported into the Jupyter notebook, a

quick check was made to find out if the right data was imported. The dataset had 186850 rows and 6 columns. The data type of each column was inspected and they were all found to be categorical. This required manipulation as some of the analyses required numerical data types. A list was created to find out the numerical and the categorical columns using the "select. dtypes" function in python.

There were only categorical columns and this needed remediation.

The next step was to find out if there were any null values present in the dataset, to do so the "isnull()" function was used. The null values in the dataset were represented as a percentage to determine the number of null values. The dataset had 1.75% null values. Since the missing data was insignificant and hence dropped.

"Quantity ordered" and "Price Each" was present as objects that are, they are string values and this had to be converted to a numeric type. Since we are dealing with electronic goods, the quantity ordered needs to be a finite number and not float values. The price of each commodity could be a float value. Since the datatypes were coerced, the data had to be checked for "NA" values. The same approach was used earlier, first, the count of "NA" values was found and then the percentage of "NA" values were found. It was found that 0.38% of the dataset had "NA" values post-coercion. These were dropped since the number was not significant.

The final step in data cleaning was to change the data type of the "Order Date" column to a "DateTime" format. To accomplish this, the "to_datetime ()" function was used.

```
all_months_sales.shape
```

```
(186850, 6)
```

```
all_months_sales.dtypes
```

```
Order ID            object
Product             object
Quantity Ordered    object
Price Each          object
Order Date          object
Purchase Address    object
dtype: object
```

Source: Jupyter Notebook running Python 3

Figure 3 Shows the shape of the data frame along with the datatypes of the columns

2. Data Processing

Once the data was cleaned, the next step involved the manipulation of data to prepare for exploratory data analysis and model building. A few columns were broken down into new columns to get more metrics. Two columns, "Order Date" and "Purchase Address" were renamed to aid in data manipulation. The goal of this was to split the "Order Date" column into subsequent columns like "Year", "Month", "Date", "Hour" and, "Minute". Similarly, the "Purchase Address" column was broken down into subsequent columns like "City", "State",

"Postal Code", and "House and Street Number". These tasks were accomplished using the "DateTime" functions and the "str. split ()" functions. The last step in data manipulation was to compute the sales column by multiplying the quantity ordered and the price of each commodity. The final dataset for analysis contained 185950 rows and 16 columns, which were exported into an excel file to be imported into Tableau to make visualizations and create an interactive dashboard.

```
all_months_sales
```

	Order ID	Product	Quantity Ordered	Price Each	OrderDate	PurchaseAddress	Year	Month	Date	Hour	Minute	HouseStreet	City	State	Postal Code
0	236670	Wired Headphones	2	11.99	2019-08-31 22:21:00	359 Spruce St, Seattle, WA 98101	2019	8	31	22	21	359 Spruce St	Seattle	WA	98101
1	236671	Bose SoundSport Headphones	1	99.99	2019-08-15 15:11:00	492 Ridge St, Dallas, TX 75001	2019	8	15	15	11	492 Ridge St	Dallas	TX	75001
2	236672	iPhone	1	700.00	2019-08-06 14:40:00	149 7th St, Portland, OR 97035	2019	8	6	14	40	149 7th St	Portland	OR	97035
3	236673	AA Batteries (4-pack)	2	3.84	2019-08-29 20:59:00	631 2nd St, Los Angeles, CA 90001	2019	8	29	20	59	631 2nd St	Los Angeles	CA	90001
4	236674	AA Batteries (4-pack)	2	3.84	2019-08-15 19:53:00	736 14th St, New York City, NY 10001	2019	8	15	19	53	736 14th St	New York City	NY	10001
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
186845	319666	Lightning Charging Cable	1	14.95	2019-12-11 20:58:00	14 Madison St, San Francisco, CA 94016	2019	12	11	20	58	14 Madison St	San Francisco	CA	94016
186846	319667	AA Batteries (4-pack)	2	3.84	2019-12-01 12:01:00	549 Willow St, Los Angeles, CA 90001	2019	12	1	12	1	549 Willow St	Los Angeles	CA	90001
186847	319668	Vareebadd Phone	1	400.00	2019-12-09 06:43:00	273 Wilson St, Seattle, WA 98101	2019	12	9	6	43	273 Wilson St	Seattle	WA	98101
186848	319669	Wired Headphones	1	11.99	2019-12-03 10:39:00	778 River St, Dallas, TX 75001	2019	12	3	10	39	778 River St	Dallas	TX	75001
186849	319670	Bose SoundSport Headphones	1	99.99	2019-12-21 21:45:00	747 Chestnut St, Los Angeles, CA 90001	2019	12	21	21	45	747 Chestnut St	Los Angeles	CA	90001

185950 rows × 15 columns

Source: Jupyter Notebook running Python 3

Figure 4 Shows the cleaned and processed data in a data frame

```
#Add another column called Sales which displays the sales as per the quantity ordered and the price for each commodity

all_months_sales['Sales'] = all_months_sales['Quantity Ordered'] * all_months_sales['Price Each']
```

Source: Jupyter Notebook running Python 3

Figure 5 Shows the computation of the sales column

	Order ID	Product	Quantity Ordered	Price Each	OrderDate	PurchaseAddress	Year	Month	Date	Hour	Minute	HouseStreet	City	State	Postal Code	Sales
0	236670	Wired Headphones	2	11.99	2019-08-31 22:21:00	359 Spruce St, Seattle, WA 98101	2019	8	31	22	21	359 Spruce St	Seattle	WA	98101	23.98
1	236671	Bose SoundSport Headphones	1	99.99	2019-08-15 15:11:00	492 Ridge St, Dallas, TX 75001	2019	8	15	15	11	492 Ridge St	Dallas	TX	75001	99.99
2	236672	iPhone	1	700.00	2019-08-06 14:40:00	149 7th St, Portland, OR 97035	2019	8	6	14	40	149 7th St	Portland	OR	97035	700.00
3	236673	AA Batteries (4-pack)	2	3.84	2019-08-29 20:59:00	631 2nd St, Los Angeles, CA 90001	2019	8	29	20	59	631 2nd St	Los Angeles	CA	90001	7.68
4	236674	AA Batteries (4-pack)	2	3.84	2019-08-15 19:53:00	736 14th St, New York City, NY 10001	2019	8	15	19	53	736 14th St	New York City	NY	10001	7.68
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
186845	319666	Lightning Charging Cable	1	14.95	2019-12-11 20:58:00	14 Madison St, San Francisco, CA 94016	2019	12	11	20	58	14 Madison St	San Francisco	CA	94016	14.95
186846	319667	AA Batteries (4-pack)	2	3.84	2019-12-01 12:01:00	549 Willow St, Los Angeles, CA 90001	2019	12	1	12	1	549 Willow St	Los Angeles	CA	90001	7.68
186847	319668	Vareebadd Phone	1	400.00	2019-12-09 06:43:00	273 Wilson St, Seattle, WA 98101	2019	12	9	6	43	273 Wilson St	Seattle	WA	98101	400.00
186848	319669	Wired Headphones	1	11.99	2019-12-03 10:39:00	778 River St, Dallas, TX 75001	2019	12	3	10	39	778 River St	Dallas	TX	75001	11.99
186849	319670	Bose SoundSport Headphones	1	99.99	2019-12-21 21:45:00	747 Chestnut St, Los Angeles, CA 90001	2019	12	21	21	45	747 Chestnut St	Los Angeles	CA	90001	99.99

Source: Jupyter Notebook running Python 3

Figure 6 The final data frame post clean up and manipulation showing the number of rows and columns

3. Exploratory Data Analysis

To find out the correlation between each variable in the dataset, a correlation plot was constructed. The findings are evident.

- Sales and Price Each are showing a strong positive correlation as the coefficient is 1.0, which indicates a positive correlation. This is represented by the brighter colors in the heatmap
- The other variables in the dataset have low coefficients and hence are not correlated to each other.

Source: Jupyter Notebook running Python 3

Figure 7 A correlation heatmap showing the correlation between variables

```
#analyze the sales variable against each variable in the dataset, sales has a strong positive co-relation with the price variable
all_months_sales.corr()['Sales'].sort_values(ascending = False).to_frame().T
```

	Sales	Price Each	Year	Hour	Minute	Date	Month	Quantity Ordered
Sales	1.0	0.999203	0.002824	0.001668	-0.002162	-0.00258	-0.003466	-0.139417

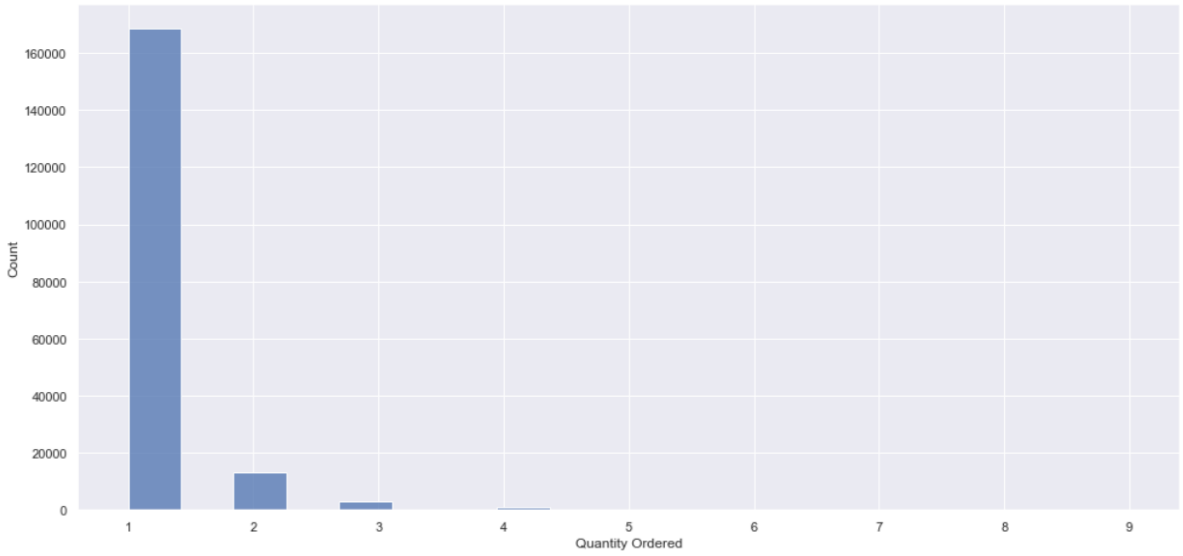
Source: Jupyter Notebook running Python 3

Figure 8 Showing the correlation of each variable with the sales variable

Univariate analysis was performed on the variables in the dataset to find out the distribution. The findings from the univariate analysis are listed below

- The histogram plot of the quantity ordered shows that a majority of customers have ordered only one product.
- There are a few records that show that customers do order more than 1 product. But it is significantly less than the number of customers who order just one product
- There are a few customers who order more than 2 or 3 products, but the frequency is very less compared to single orders.
- This can be indicative of a sales manager who wants to increase sales and has to come up with a way so that customers buy multiple products instead of being skewed towards buying just one product.

```
#Lets find out the distributions of each variable in the data
snb.histplot(data = all_months_sales['Quantity Ordered'])
plt.show()
```

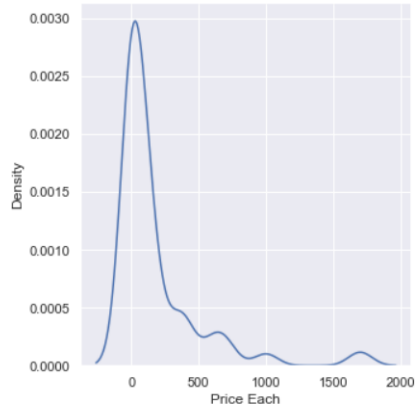


Source: Jupyter Notebook running Python 3

Figure 9 A histogram showing the frequency of the quantity of orders

- A distribution plot was constructed to find how the price of each commodity is distributed. From the distribution plot it appears that the distribution is skewed as not many products' prices exceed 500 dollars.

```
snb.displot(all_months_sales['Price Each'],kind = "kde", bw_adjust = 3)
plt.show()
```

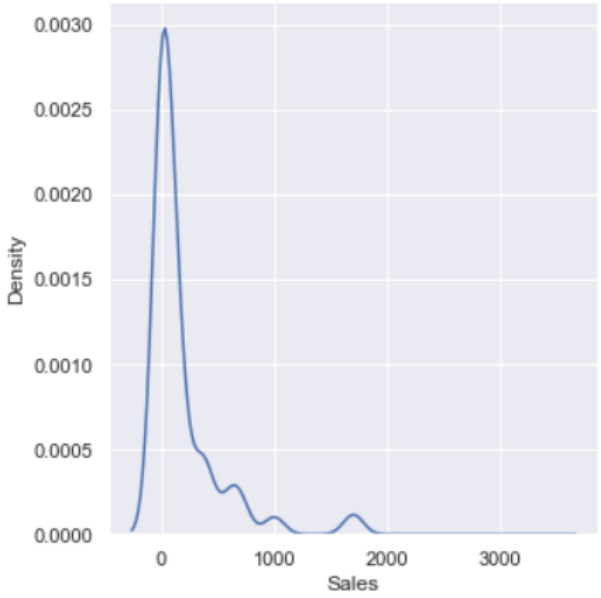


Source: Jupyter Notebook running Python 3

Figure 10 A distribution plot of the price of each commodity

- Similarly, a distribution plot was constructed for the sales variable and similarly, this too is skewed with not many sales exceeding 1000 dollars.

```
snb.displot(all_months_sales['Sales'],kind = "kde", bw_adjust = 3)
plt.show()
```

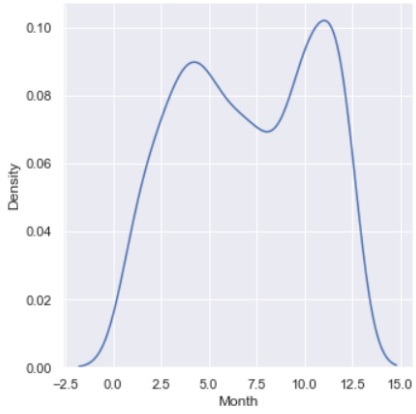


Source: Jupyter Notebook running Python 3

Figure 11 A distribution plot showing the distribution of sales

- The distribution plot of the "Month" variable shows bimodality. There are two instances where the orders were more compared to the rest. This was also confirmed by plotting a histogram to know the frequency of orders.

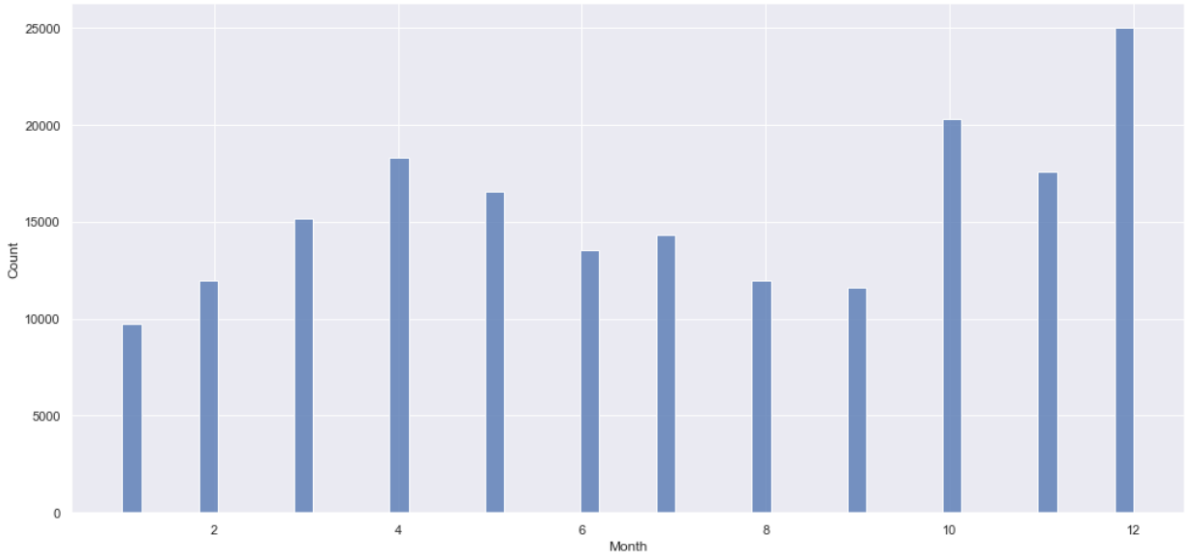
```
snb.displot(all_months_sales['Month'], kind = 'kde', bw_adjust = 3)
plt.show()
```



Source: Jupyter Notebook running Python 3

Figure 12 The distribution plot for the month variable

```
snb.histplot(all_months_sales['Month'])
plt.show()
```

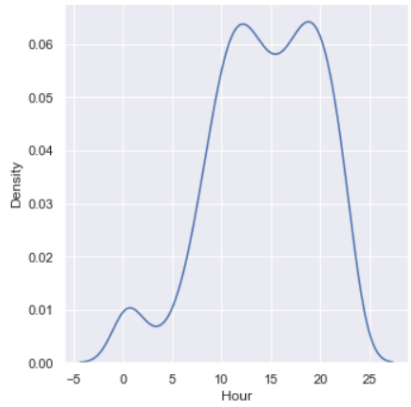


Source: Jupyter Notebook running Python 3

Figure 13 A histogram showing the frequency of orders per month

- The distribution plot for hours suggests that a majority of orders come between 10 AM - 12 pm and then dip and then pick up at around 6 PM - 7 PM and then dip as indicated by the density parameter in the distribution plot.

```
snb.displot(all_months_sales['Hour'], kind = 'kde', bw_adjust = 3)
plt.show()
```

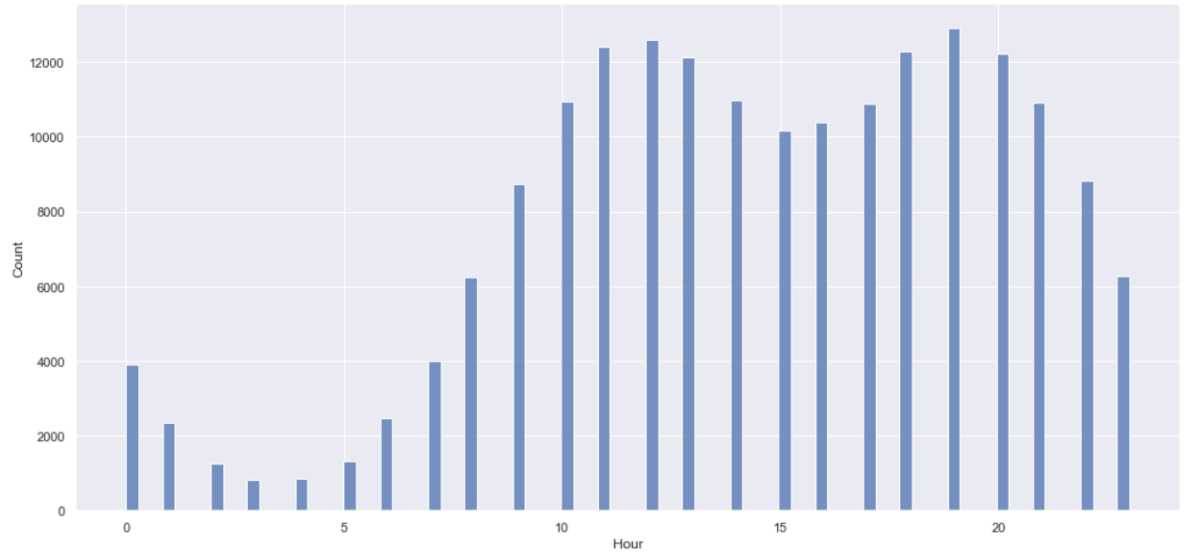


Source: Jupyter Notebook running Python 3

Figure 14 A distribution plot of the hour's variable

- A histogram was plotted to conform with the above findings from the distribution plot.

```
snb.histplot(all_months_sales['Hour'])
plt.show()
```

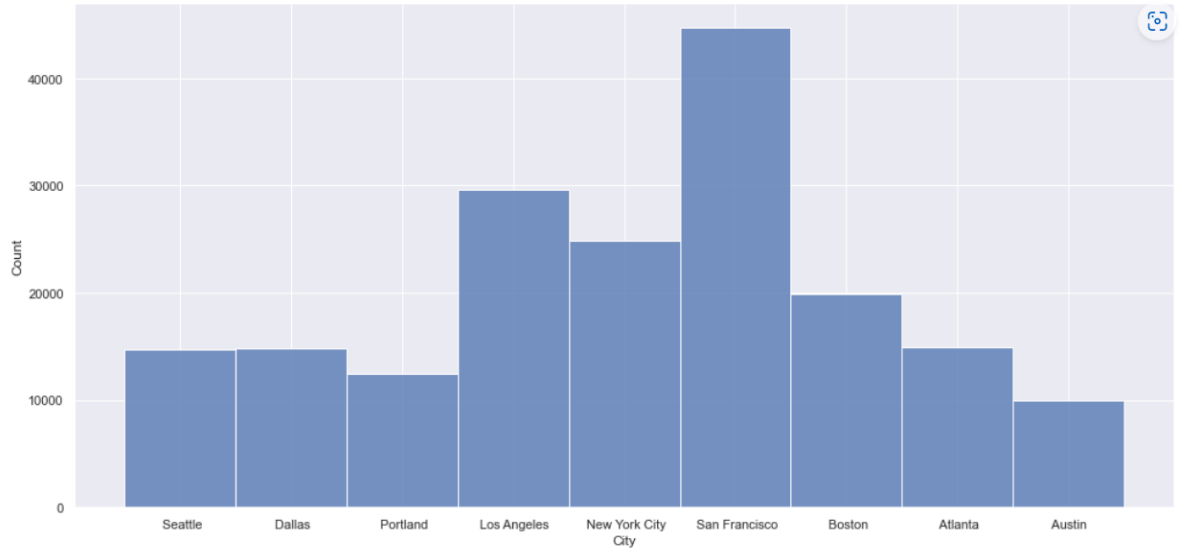


Source: Jupyter Notebook running Python 3

Figure 15 A histogram showing the distribution of the hour's variable

- Finally, a histogram was plotted to find out the frequency of orders from each of the eight cities present in the dataset. It appears that the city of San Francisco has the highest number of orders.

```
snb.histplot(all_months_sales['City'])
plt.show()
```



Source: Jupyter Notebook running Python 3

Figure 16 A histogram showing the frequency of orders per city

Tableau Dashboard

The objective was to build an interactive dashboard in Tableau using the exported cleaned dataset in an excel file. The image below is from the dashboard and shows the total revenue, "total quantity ordered", and revenue by each City and State. To find out the top 5 products that the customer purchased, I constructed a simple bar chart to show the top-performing products. The dashboard also shows the revenue in each month and the peak hours in which most purchases are made.

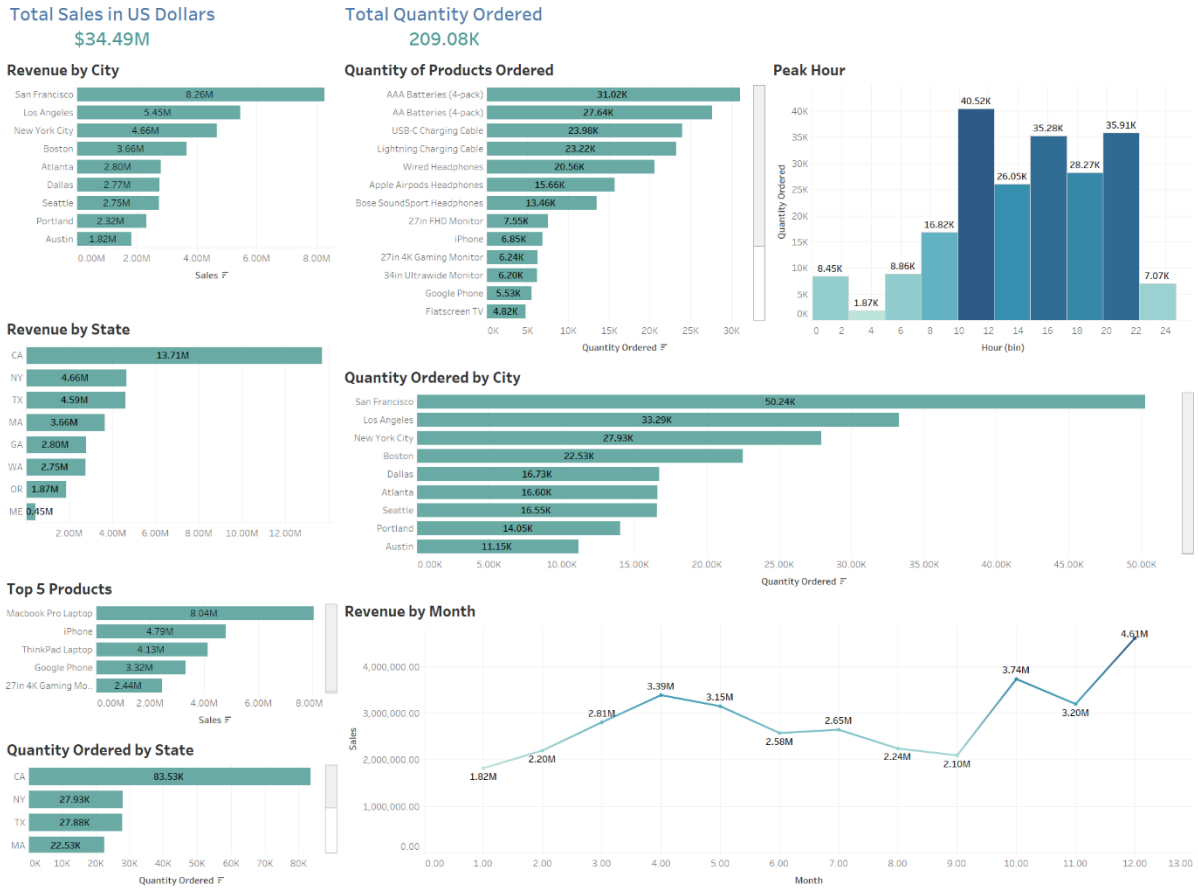


Figure 17 A dashboard built in Tableau show the sales analytics

Source: Tableau

Interpretations and Insights

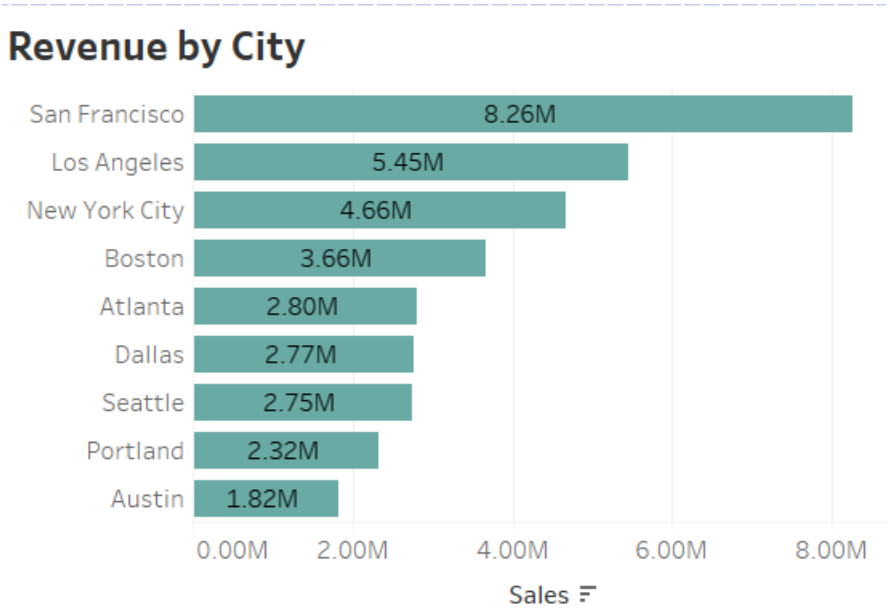
The purpose of building a dashboard to aid the viewer to visually interpret information as a whole. Using tableau one can add individual sheets and build each graph and then combine each of them to build the dashboard.

In the above dashboard, the total sales in the year 2019 is obtained by dragging the sales column and computing the sum. By looking at the visual one can gather that the year 2019 yielded 34.49 million in sales.

Similarly, the total quantity of products sold is determined by dragging the quantity ordered column and computing the sum. The audience can gather that in the year 2019, 209.08k products were sold.

Which City had the most sales in the year?

The revenue by City and State are simple bar charts in descending order of revenue. From the graph it can be found out that the city of San Francisco has the highest revenue by city with 8.26 million generated in sales in the year 2019. This also is true if the data is filtered by State. The state of California has the highest sales by revenue with 13.71 million in yearly sales.



Source: Tableau

Figure 18 A tableau bar chart showing the revenue per city

Which State had the most sales in the year?

The graph on "Quantity ordered by City" supplements the insight generated from the revenue by state graph. Since the top two cities in terms of quantity of products ordered are San Francisco and Los Angeles, making California the most profitable state in terms of Sales. On the contrary, one can look at the least performing state in terms of sales and take measures to improve the sales.

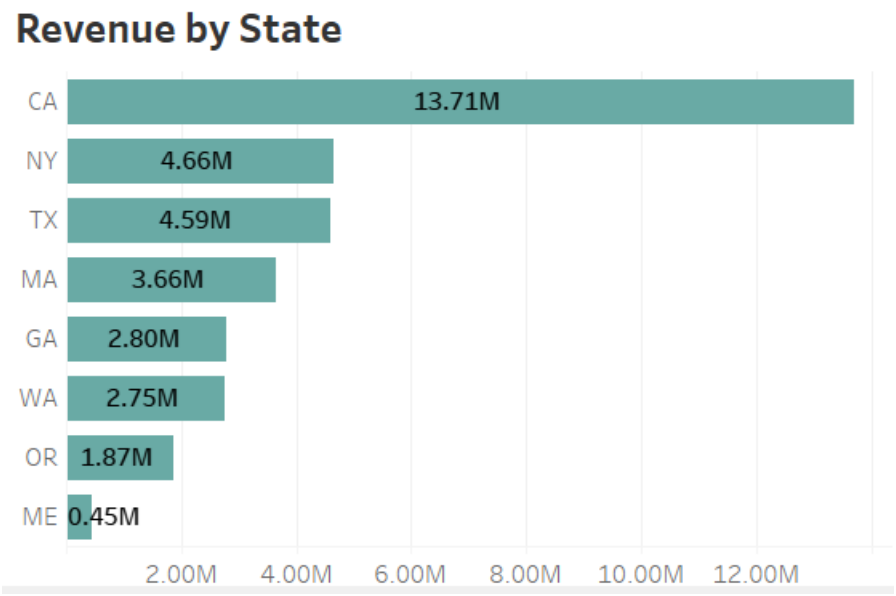
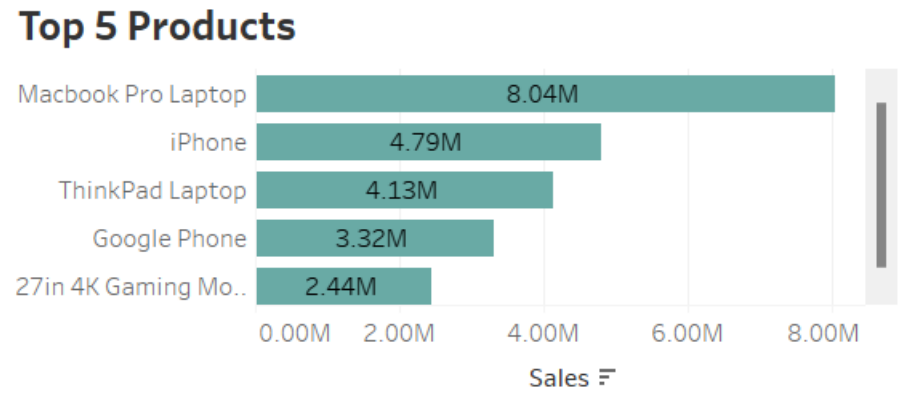


Figure 19 A tableau chart showing revenue per state

Source: Tableau

What are the top 5 products that consumers tend to buy?

Another key insight from the dashboard, the top 5 products in terms of sales. From the bar graph it can be interpreted that the MacBook Pro Laptop, iPhone, ThinkPad Laptop, Google Phone, 27-inch gaming monitor are the top five buys. Hence it would make sense that the store could market these items on a larger scale compared to products that do not sell well.



Source: Tableau

Figure 20 A tableau chart showing the top 5 products

What was the best month for sales?

By looking at the line graph of the revenue per month, it can be seen that, the month of December had the most revenue in terms of sales. This number amounting to 4.61 million dollars.

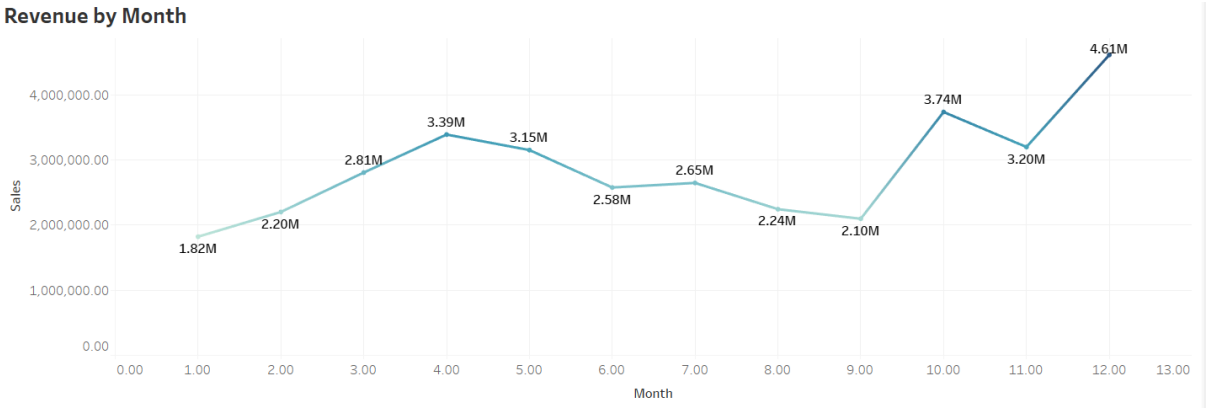


Figure 21 A tableau line chart showing revenue per month

Source: Tableau

What is the peak hour when the orders are placed?

It can be observed from the histogram that most orders were placed between 10-12 AM, which had 40.52K orders. The varying intensity of the color in the histogram also indicate the difference in terms of orders placed in a 24-hour time frame. The darker colors indicate more quantity of orders compared to lighter shades of blue.

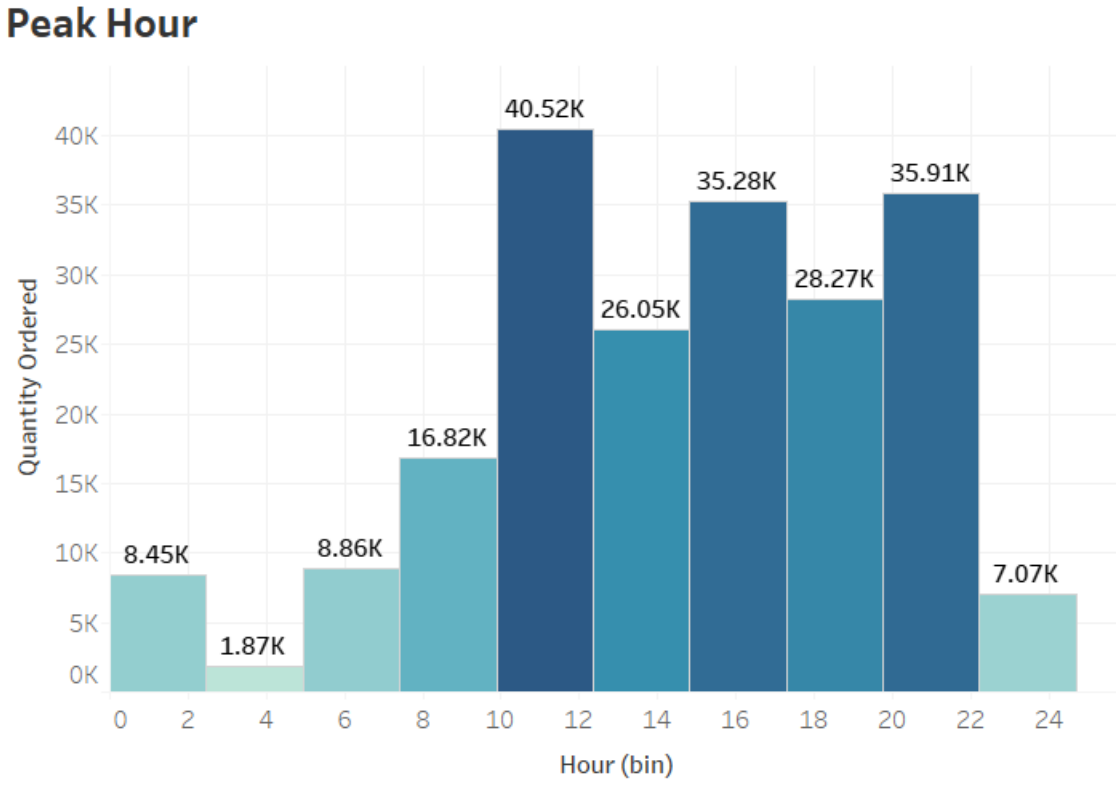


Figure 22 A tableau histogram showing the peak hours

Source: Tableau

4. Modeling and Algorithms

The idea of the project is to implement the apriori algorithm to find out item sets that are more frequently ordered so that there is some kind of a recommender system that suggests to the customer to buy another product associated with their current purchase. Thus the aim of the project is to generate association rules using the apriori algorithm and association rules to increase sales.

To do this in python, the data had to be generated by grouping the orders by purchase address. The idea was that orders which had the same purchase address were bought by the same customer, so the recommendation would be to buy a similar product together. The products which were ordered together were put into a list. Once the list of multiple products ordered was generated, the apriori algorithm was implemented to get a list of products along with three metrics - support, confidence and lift. Then this list was put into a for loop to generate the association rules.

```
print(multiple_products)
```

```
[['Wired Headphones', 'Macbook Pro Laptop'], ['Bose SoundSport Headphones', 'Lightning Charging Cable'], ['USB-C Charging Cab
le', 'Macbook Pro Laptop'], ['Lightning Charging Cable', 'USB-C Charging Cable'], ['USB-C Charging Cable', 'Flatscreen TV'],
['AA Batteries (4-pack)', '27in FHD Monitor'], ['iPhone', 'Lightning Charging Cable'], ['AA Batteries (4-pack)', 'Apple Airpo
ds Headphones', '27in 4K Gaming Monitor'], ['Google Phone', 'USB-C Charging Cable'], ['AAA Batteries (4-pack)', 'Bose SoundSp
ort Headphones', 'Apple Airpods Headphones'], ['AAA Batteries (4-pack)', 'ThinkPad Laptop'], ['34in Ultrawide Monitor', 'Bose
SoundSport Headphones'], ['AAA Batteries (4-pack)', '27in 4K Gaming Monitor'], ['USB-C Charging Cable', 'Bose SoundSport Head
phones'], ['27in FHD Monitor', 'Google Phone'], ['AAA Batteries (4-pack)', 'USB-C Charging Cable'], ['iPhone', 'Lightning Cha
rging Cable', 'Wired Headphones'], ['iPhone', 'Wired Headphones'], ['Lightning Charging Cable', '27in 4K Gaming Monitor'],
['Bose SoundSport Headphones', 'Apple Airpods Headphones', 'AA Batteries (4-pack)'], ['AAA Batteries (4-pack)', 'Bose SoundSp
ort Headphones', 'AA Batteries (4-pack)'], ['Apple Airpods Headphones', 'Google Phone'], ['Lightning Charging Cable', 'Apple
Airpods Headphones'], ['USB-C Charging Cable', '27in 4K Gaming Monitor', 'Lightning Charging Cable'], ['Lightning Charging Ca
ble', '34in Ultrawide Monitor'], ['Apple Airpods Headphones', 'Wired Headphones'], ['Apple Airpods Headphones', 'Bose SoundSp
ort Headphones', 'AA Batteries (4-pack)'], ['USB-C Charging Cable', 'Wired Headphones'], ['USB-C Charging Cable', 'AAA Batter
ies (4-pack)'], ['Google Phone', 'Wired Headphones'], ['Flatscreen TV', 'Flatscreen TV'], ['AA Batteries (4-pack)', 'AAA Batt
eries (4-pack)'], ['Lightning Charging Cable', 'LG Dryer'], ['Google Phone', 'Wired Headphones'], ['Flatscreen TV', 'iPhon
e'], ['Bose SoundSport Headphones', 'Lightning Charging Cable'], ['iPhone', 'Apple Airpods Headphones'], ['USB-C Charging Cab
le', 'USB-C Charging Cable'], ['USB-C Charging Cable', 'AAA Batteries (4-pack)'], ['Wired Headphones', 'Macbook Pro Laptop',
'USB-C Charging Cable', '34in Ultrawide Monitor'], ['Google Phone', 'Macbook Pro Laptop', 'Macbook Pro Laptop'], ['27in 4K Ga
ming Monitor', 'iPhone', 'iPhone'], ['AA Batteries (4-pack)', '27in FHD Monitor'], ['AA Batteries (4-pack)', 'Wired Headphone
```

Source: Jupyter Notebook running Python 3

Figure 23 The list of multiple products ordered by the same customer

```
output
```

```
[RelationRecord(items=frozenset({'iPhone', 'Wired Headphones', '20in Monitor', 'Apple Airpods Headphones'}), support=0.00011353
959693443088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'Wired Headphones', '20in Monitor', 'Apple Airpods Hea
dphones'}), items_add=frozenset({'iPhone'}), confidence=0.4444444444444444, lift=4.179865931067212)]),
 RelationRecord(items=frozenset({'Lightning Charging Cable', 'Macbook Pro Laptop', 'iPhone', 'Apple Airpods Headphones'}), supp
ort=0.00019869429463525404, ordered_statistics=[OrderedStatistic(items_base=frozenset({'iPhone', 'Macbook Pro Laptop', 'Apple A
irpods Headphones'}), items_add=frozenset({'Lightning Charging Cable'}), confidence=0.875, lift=3.5318801558203483)]),
 RelationRecord(items=frozenset({'Google Phone', 'USB-C Charging Cable', 'Bose SoundSport Headphones', 'Vareebadd Phone'}), sup
port=0.00011353959693443088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'Google Phone', 'Bose SoundSport Headph
ones', 'Vareebadd Phone'}), items_add=frozenset({'USB-C Charging Cable'}), confidence=1.0, lift=3.978093947606143)]),
 RelationRecord(items=frozenset({'USB-C Charging Cable', 'Google Phone', 'Wired Headphones', 'Flatscreen TV'}), support=0.00011
353959693443088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'USB-C Charging Cable', 'Wired Headphones', 'Flatsc
reen TV'}), items_add=frozenset({'Google Phone'}), confidence=0.26666666666666666, lift=3.0954420647995606)]),
 RelationRecord(items=frozenset({'27in FHD Monitor', 'iPhone', 'Wired Headphones', 'Apple Airpods Headphones', 'AA Batteries (4
-pack)'}), support=0.00011353959693443088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'27in FHD Monitor', 'iPho
ne', 'AA Batteries (4-pack)'}), items_add=frozenset({'Wired Headphones', 'Apple Airpods Headphones'}), confidence=0.26666666666
666666, lift=10.712276700874192), OrderedStatistic(items_base=frozenset({'27in FHD Monitor', 'iPhone', 'Apple Airpods Headphone
s'}), items_add=frozenset({'Wired Headphones', 'AA Batteries (4-pack)'}), confidence=0.25, lift=7.849821746880571), OrderedStat
istic(items_base=frozenset({'27in FHD Monitor', 'Wired Headphones', 'Apple Airpods Headphones', 'AA Batteries (4-pack)'}), item
s_add=frozenset({'iPhone'}), confidence=0.8, lift=7.523758675920983), OrderedStatistic(items_base=frozenset({'27in FHD Monito
r', 'iPhone', 'Apple Airpods Headphones', 'AA Batteries (4-pack)'}), items_add=frozenset({'Wired Headphones'}), confidence=0.8,
lift=3.6296200901481006), OrderedStatistic(items_base=frozenset({'27in FHD Monitor', 'iPhone', 'Wired Headphones', 'Apple Airpo
ds Headphones'}), items_add=frozenset({'AA Batteries (4-pack)'}), confidence=0.6666666666666666, lift=3.0438914809054776), Orde
redStatistic(items_base=frozenset({'iPhone', 'Wired Headphones', 'Apple Airpods Headphones', 'AA Batteries (4-pack)'}), items_a
dd=frozenset({'27in FHD Monitor'}), confidence=0.4, lift=4.749578699022582)]),
 RelationRecord(items=frozenset({'iPhone', 'Wired Headphones', 'Apple Airpods Headphones', 'Lightning Charging Cable', 'USB-C C
harging Cable'}), support=0.0001419244961680386, ordered_statistics=[OrderedStatistic(items_base=frozenset({'Lightning Charging
Cable', 'USB-C Charging Cable', 'Wired Headphones', 'Apple Airpods Headphones'}), items_add=frozenset({'iPhone'}), confidence=
0.5555555555555556, lift=5.224832413834016)])]
```

Source: Jupyter Notebook running Python 3

Figure 24 The output after implementing the apriori algorithm

```
for i in range (0, len(output)):
    print(output[i][0])
```

```
frozenset({'iPhone', 'Wired Headphones', '20in Monitor', 'Apple Airpods Headphones'})
frozenset({'Lightning Charging Cable', 'Macbook Pro Laptop', 'iPhone', 'Apple Airpods Headphones'})
frozenset({'Google Phone', 'USB-C Charging Cable', 'Bose SoundSport Headphones', 'Vareebadd Phone'})
frozenset({'USB-C Charging Cable', 'Google Phone', 'Wired Headphones', 'Flatscreen TV'})
frozenset({'27in FHD Monitor', 'iPhone', 'Wired Headphones', 'Apple Airpods Headphones', 'AA Batteries (4-pack)'})
frozenset({'iPhone', 'Wired Headphones', 'Apple Airpods Headphones', 'Lightning Charging Cable', 'USB-C Charging Cable'})
```

Source: Jupyter Notebook running Python 3

Figure 25 Generating the association rules

By running the below code, the rules are stored in a list. The list contains the item sets frequently ordered and the support, confidence and lift metrics.

```
association_rules = [list(output[i]) for i in range(0, len(output))]
```

Source: Jupyter Notebook running Python 3

Figure 26 The rules generated from the apriori algorithm are stored in list

```
print(association_rules)
```

```
[[frozenset({'iPhone', 'Wired Headphones', '20in Monitor', 'Apple Airpods Headphones'}), 0.00011353959693443088, [OrderedStat
istic(items_base=frozenset({'Wired Headphones', '20in Monitor', 'Apple Airpods Headphones'}), items_add=frozenset({'iPhon
e'}), confidence=0.4444444444444444, lift=4.179865931067212)]], [frozenset({'Lightning Charging Cable', 'Macbook Pro Laptop',
'iPhone', 'Apple Airpods Headphones'}), 0.00019869429463525404, [OrderedStatistic(items_base=frozenset({'iPhone', 'Macbook Pr
o Laptop', 'Apple Airpods Headphones'}), items_add=frozenset({'Lightning Charging Cable'}), confidence=0.875, lift=3.53188015
58203483)]], [frozenset({'Google Phone', 'USB-C Charging Cable', 'Bose SoundSport Headphones', 'Vareebadd Phone'}), 0.0001135
3959693443088, [OrderedStatistic(items_base=frozenset({'Google Phone', 'Bose SoundSport Headphones', 'Vareebadd Phone'}), ite
ms_add=frozenset({'USB-C Charging Cable'}), confidence=1.0, lift=3.978093947606143)]], [frozenset({'USB-C Charging Cable', 'G
oogle Phone', 'Wired Headphones', 'Flatscreen TV'}), 0.00011353959693443088, [OrderedStatistic(items_base=frozenset({'USB-C C
harging Cable', 'Wired Headphones', 'Flatscreen TV'}), items_add=frozenset({'Google Phone'}), confidence=0.26666666666666666,
lift=3.0954420647995606)]], [frozenset({'27in FHD Monitor', 'iPhone', 'Wired Headphones', 'Apple Airpods Headphones', 'AA Bat
teries (4-pack)'}), 0.00011353959693443088, [OrderedStatistic(items_base=frozenset({'27in FHD Monitor', 'iPhone', 'AA Batteri
es (4-pack)'}), items_add=frozenset({'Wired Headphones', 'Apple Airpods Headphones'}), confidence=0.26666666666666666, lift=1
0.712276700874192), OrderedStatistic(items_base=frozenset({'27in FHD Monitor', 'iPhone', 'Apple Airpods Headphones'}), items_
add=frozenset({'Wired Headphones', 'AA Batteries (4-pack)'}), confidence=0.25, lift=7.849821746880571), OrderedStatistic(item
s_base=frozenset({'27in FHD Monitor', 'Wired Headphones', 'Apple Airpods Headphones', 'AA Batteries (4-pack)'}), items_add=fr
ozenset({'iPhone'}), confidence=0.8, lift=7.523758675920983), OrderedStatistic(items_base=frozenset({'27in FHD Monitor', 'iPh
one', 'Apple Airpods Headphones', 'AA Batteries (4-pack)'}), items_add=frozenset({'Wired Headphones'}), confidence=0.8, lift=
3.6296200901481006), OrderedStatistic(items_base=frozenset({'27in FHD Monitor', 'iPhone', 'Wired Headphones', 'Apple Airpods
```

Source: Jupyter Notebook running Python 3

Figure 27 Printing the generated rule from the apriori algorithm

```
[[frozenset({'Apple Airpods Headphones',
             'Lightning Charging Cable',
             'Macbook Pro Laptop',
             'iPhone'}),
  0.00019869429463525404,
  [OrderedStatistic(items_base=frozenset({'iPhone', 'Macbook Pro Laptop', 'Apple Airpods Headphones'}), items_add=frozenset({'L
ightning Charging Cable'}), confidence=0.875, lift=3.5318801558203483)]],
```

Source: Jupyter Notebook running Python 3

Figure 28 One of the rules displaying the corresponding item sets and the support, confidence and lift metrics

Set A = {Apple Air pods Headphones, Lightning Charging Cable, MacBook Pro Laptop, iPhone} -> This an item set that contains products frequently bought together. The apriori algorithm suggests that subsets of an item set are also assumed to be frequently bought. These subsets being:

Set B = {'Lightning Charging Cable'}

Set C = {'iPhone', 'MacBook Pro Laptop', 'Apple Air pods Headphones'}

The three metrics that provides a measure for the rules are – Support, Confidence and Lift.

In the above rule the support is 0.001, confidence is 0.875 and lift is 3.531, meaning that a customer who bought an iPhone is more likely to buy Apple Air Pods, lightning charging cable, a MacBook Pro Laptop with the confidence being 0.875. The lift is 3.531 which indicates the if a customer buys an Apple iPhone, then he is more likely to buy an Apple Air pods. Many rules can further be set up based on the minimum support threshold and the above metrics determined. Hence these rules provide an indication to the sales manager of the "Store" to sell related products for maximum profits through sales. The code below parses through the association rules generated in the previous step and prints in a more readable format.

```
for item in association_rules:
    pair = item[0]
    items = [x for x in pair]
    print("Rule: " + items[0] + "->" + items[1])
    print("Support: " + str(item[1]))
    print("Confidence: " + str(item[2][0][2]))
    print("Lift: " + str(item[2][0][3]))
    print("====================================")
```

Source: Jupyter Notebook running Python 3

Figure 17 Code to parse and print the generated association rules

```
Rule: 20in Monitor->Wired Headphones
Support: 0.00011353959693443088
Confidence: 0.4444444444444444
Lift: 4.179865931067212
====================================
Rule: Apple Airpods Headphones->Lightning Charging Cable
Support: 0.00019869429463525404
Confidence: 0.875
Lift: 3.5318801558203483
====================================
Rule: Google Phone->USB-C Charging Cable
Support: 0.00011353959693443088
Confidence: 1.0
Lift: 3.978093947606143
====================================
Rule: Google Phone->Wired Headphones
Support: 0.00011353959693443088
Confidence: 0.26666666666666666
Lift: 3.0954420647995606
====================================
Rule: Apple Airpods Headphones->AA Batteries (4-pack)
Support: 0.00011353959693443088
Confidence: 0.26666666666666666
Lift: 10.712276700874192
====================================
Rule: iPhone->Wired Headphones
Support: 0.0001419244961680386
Confidence: 0.5555555555555556
Lift: 5.224832413834016
====================================
```

Source: Jupyter Notebook running Python 3

Figure 18 The generated association rules along with support, confidence and lift metrics in a readable format

Challenges

Data acquisition, clean up and processing did not present themselves as a challenge. The implementation of the algorithm and parsing through the rules was quite challenging, as I found it tricky to parse through a nested list of association rules. I had to look for solutions to effectively parse through the nested list so that the rules along with the metrics could be more readable than having to look at the nested list.

Lessons Learned

Through the project, I learned a great deal about the apriori algorithm and frequent itemset mining, which was coincidentally also part of the course under “Module 7”. It was a great experience to implement what was taught in class into the project. Learning the concepts while performing the modeling and the analysis, reinforced the concepts learned.

Additionally, the other aspect that I learned was getting hands-on with Tableau, which is one of the key visualization tools in a data analyst's skill set.

Conclusion

In this project, both Python and Tableau were used to visualize data and information to gain insights into the distribution of the variables in the data and the correlations between each variable in the dataset. The objective of the project was to build an interactive dashboard in Tableau so that the audience gets an insight into the yearly sales in the year 2019 and answers some of the questions set up during the inception phase through EDA and dashboarding. Additionally, the apriori algorithm was implemented to find out the frequent item sets and to generate association rules between products with the intention of marketing them together to increase revenue from sales. The overall goal of the project was to simulate a full-fledged data analytics project using real-world data.

References

[1]. Knightbearr. (2021, November 4). Sales product data. Kaggle. Retrieved September 1, 2022, from Kaggle

[2]. Li, S. (2017, September 27). *A gentle introduction on Market Basket Analysis - Association rules*. Medium. Retrieved September 19, 2022, from https://towardsdatascience.com/a-gentle-introduction-on-market-basket-analysis-association-rules-fa4b986a40ce

[3]. Kadlaskar, A. (2022, July 27). *Market basket analysis: Guide on market basket analysis*. Analytics Vidhya. Retrieved September 19, 2022, from https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-market-basket-analysis/

[4]. *Apyori*. PyPI. (n.d.). Retrieved September 19, 2022, from https://pypi.org/project/apyori/

[5]. *Visualizing distributions of data#*. Visualizing distributions of data - seaborn 0.12.0 documentation. (n.d.). Retrieved September 19, 2022, from https://seaborn.pydata.org/tutorial/distributions.html

[6]. Dobilas, S. (2022, May 9). *Apriori algorithm for Association Rule learning - how to find clear links between transactions*. Medium. Retrieved September 19, 2022, from https://towardsdatascience.com/apriori-algorithm-for-association-rule-learning-how-to-find-clear-links-between-transactions-bf7ebc22cf0a

[7]. Visualizing distributions of data#. Visualizing distributions of data - seaborn 0.12.0 documentation. (n.d.). Retrieved October 2, 2022, from https://seaborn.pydata.org/tutorial/distributions.html

[8]. Seaborn.histplot#. seaborn.histplot - seaborn 0.12.0 documentation. (n.d.). Retrieved October 2, 2022, from https://seaborn.pydata.org/generated/seaborn.histplot.html

[9]. Tableau Tutorial. Tutorials Point. (n.d.). Retrieved October 2, 2022, from https://www.tutorialspoint.com/tableau/index.htm

[10]. Anaconda nucleus. Anaconda Nucleus. (n.d.). Retrieved October 2, 2022, from https://anaconda.cloud/getting-started-with-anaconda-distribution-notebook

Link to the dashboard on Tableau Public

https://public.tableau.com/views/SalesAnalyticsDashboard_16634874150670/Dashboard1?:language=en-US&:display_count=n&:origin=viz_share_link