By VamseeKiran Chokkam

  Student ID: 2436615

DatA Mining project

**Objective:**

The objective of this project is to examine the data seized by the Police from Tump’s computers and provide insights on the discrepancies in data, distribution of data, and produce a report that can be understood by everyone having no knowledge in the field of analytics.

**Section 1:**

I have received a total of 5 data sets to analyze, examine and produce the report. The 5 data sets provide enough information to understand the income made by the company, the items they sell, the number of employees involved, and customer behavior.

 The five data sets I have received are Customers, Orders, OrderInfo, Items, and Employee data. All the datasets doesn’t have any **null** value.

**How is the data distributed (how much data is there and where is it)?**

The orderInfo table holds the data where the information regarding all other details can be navigated through it. So, I call the orderInfo as the index table where the other important tables are navigated. The orders table which is connected to orderInfo table contains and connects to all the information except the Item information. So, on applying the joins operation on the data frame orderInfo with Items and Orders all data can be accessed.

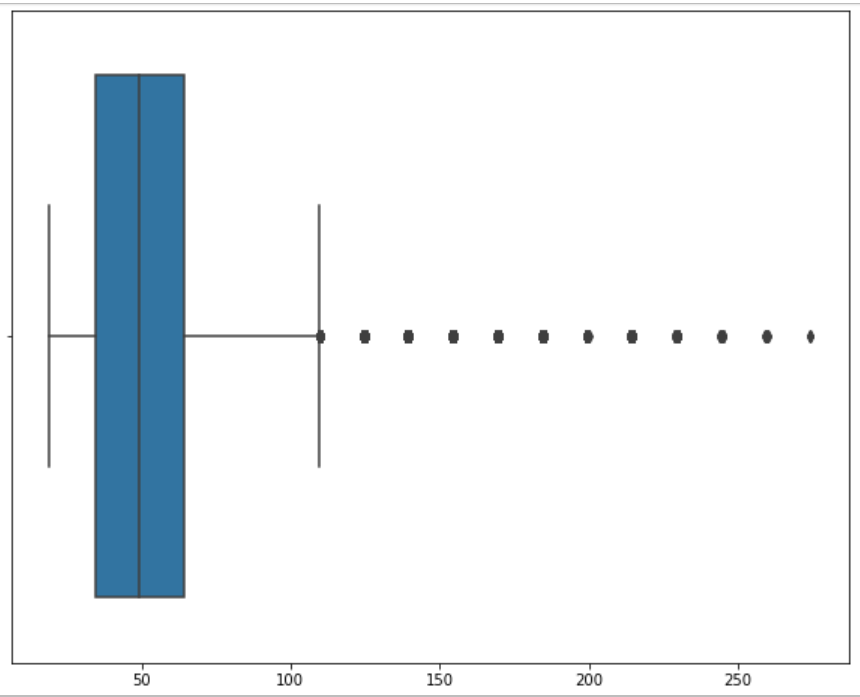
**OrderInfo:**

There is a total of 461840 rows and 3 columns in orderInfo but 149425 entries in the orders table. It is because of the multiple repetitions of order numbers with different Item No in the orderInfo table. The type of all three columns in **orderInfo** table is Int64. Though these columns are integers they just denote a connection with the other tables. So, there is no correlation between any of the fields in the orderInfo table. The **quantity** feature denotes the number of items purchased by having the same order number. The maximum quantity recorded is 3 and the minimum is 1.

**Orders:**

The orders table is so crucial in the analysis as it holds most of the data and provides important insights into the analysis. It has 149425 rows and 5 columns, among the 5 columns the order No, Customer ID and Employee Code is in data, the Order Date and Dispatch date is Date time and the total is the total amount paid for the Order No by the customer with the Customer ID. So, the Order Date, Dispatch Date, and total features are used here to understand the data by connecting the employee and Customer tables. From the analyses, it has been observed that the Order Date is not greater than the Dispatch date. So, there are no wrong entries in the date, and also the total sum of item price of that order No is equals to the **total** value in the orders table. This shows that billing and data entry is correctly made.

The minimum total amount spent by a customer on a single purchase is 19 and the maximum total amount spent is 274.4. The mean of the total amount spent is 52.8. The mean is much closer to the minimum value which mean there are outliers with very high total value. The box plot attached below shows the distribution of outliers



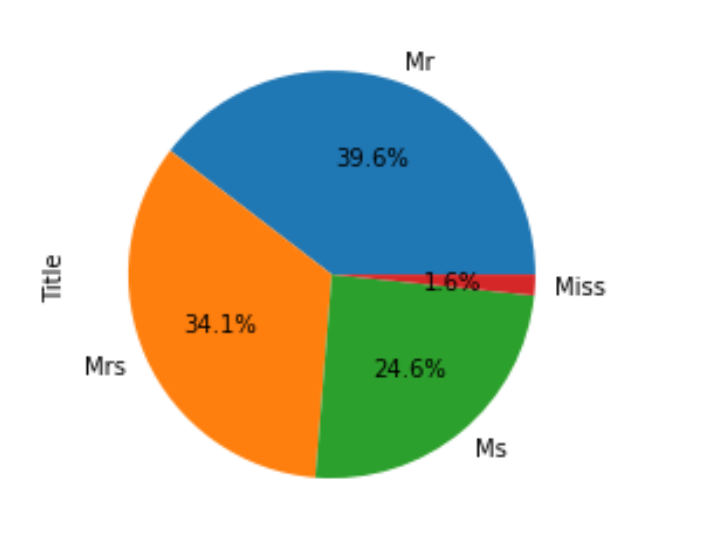
**Items:**

This is also an important data frame that contributes data to the orders table. There is a total of 150 available. There are 150 CD ROM available without the repetition of the same item and each item has given with an **Item No** which is unique and cost price and Nominal Sale Price. These two values are continuous values and the difference between the Nominal Sale Price to Cost Price gives the benefit gained over the product by the company. The maximum difference between this is 2.32 and the minimum difference is 0.003, The average difference between them is 0.62.

**Customers:**

There are 8000 customer data recorded in the customer table. The table contains 4 columns with a customer ID that is unique to each customer, a Title that represents the age and gender of the customer, first name, and last name of the customer. There is no correlation between the features inside this table. But the interesting insights taken from this table are with the title feature.

Out of 8000 customers, the total number of males is 3157 and the females are 4843 which is 39.4% and 60.5% respectively. The females are again categorized into married, unmarried (not sure of marital status and age over 30) and unmarried with age under 30 and the available percentage is 34.2, 24.6, and 1.6 respectively.

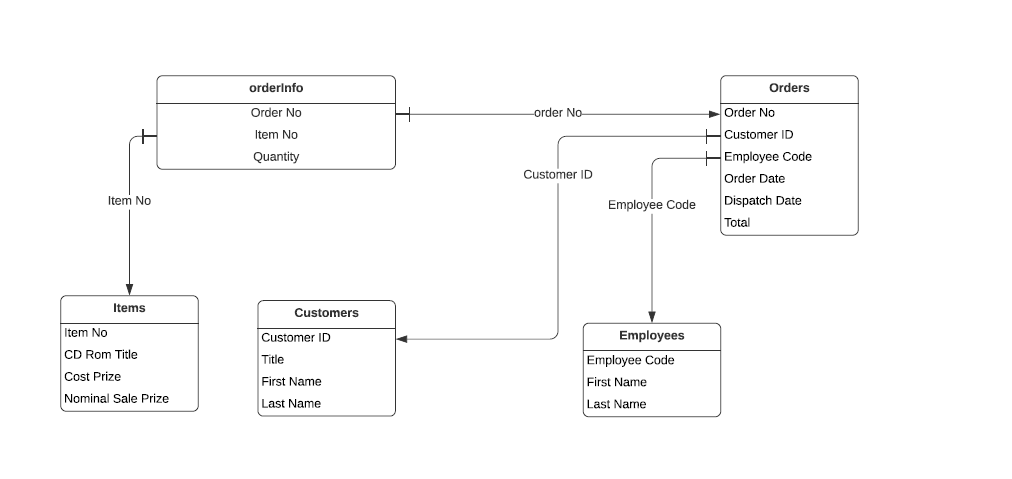


**Employee:**

There are in total of 200 employees with different unique IDs. The Employee data, when joined with the orders table, gives insight into the number of customers and the products handled by the employee which in turn results in the performance of each employee.

**Are the five sets of data independent or related in some way?**

The relation from one data set to other can be better visualized [2] in the UML diagram attached below.



**Source**: Created using <https://lucid.app/>

From the orderInfo, the orders data and items data are related. The orders dataframe on the other hand holds the brief details of the orders like the customer ID who made the order, the Employee Code to identify the employee who was involved in processing the order, and other details like Order date, dispatched date, and total money spent on the order. On the other hand, Item No acts as a foreign key that connects the orderInfo table with the Items table. Both Employees and Customers tables can be navigated from the orders table.

**The given data is transactional data because of the following reasons**

All the tables provided for analysis is transactional data. The reasons for mentioning them as Transaction data are –

1. All the tables satisfy all 3 normal forms.
2. The data is not huge as it can be processed in a system with average computing power.
3. There is no unstructured data like images, videos, voice notes inside the given data.
4. The data holds the information about the customer purchase activity and holds the details of all transactions.

But the given tables fall under many categories of transactional data[3][1], like listed below

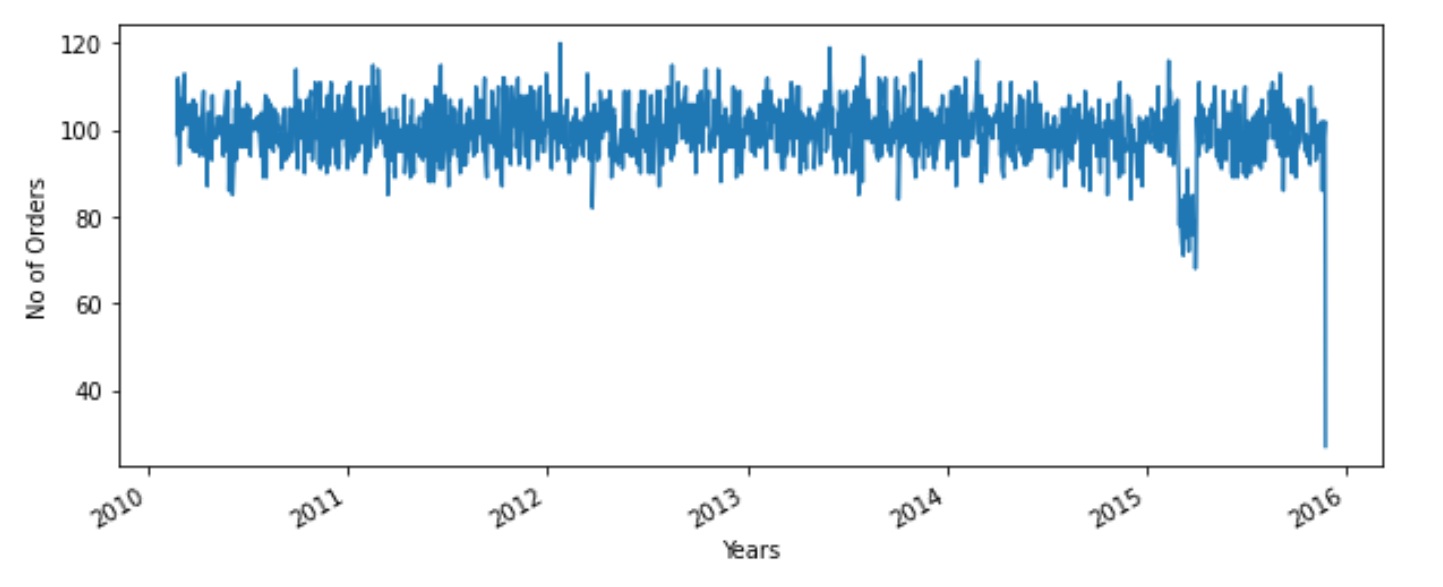
1. **Customer data** is master data. As per the definition of master data, the data of this type is used to describe the people, places, and things. The Customer data table provided gives only details about the customer.
2. **Employee Data** is master data where the table describes the Employee details.
3. **Items data** – This is reference Data, where the data inside the reference data are referred to by transactional and master records. In our case, the Items data has been referred by OrderInfo table to get the details of Items.
4. **orderInfo data** – This table holds the data about the data like orders, Items, and quantity. The data about data falls under the Metadata category. So, the orderInfo data is a metadata.
5. **orders** – This is transactional data where the table contains all the details about transactions. This data contains the transaction records which include master and reference data. The orders data contains the transaction records of master data which is Customer data and Employee data and the reference data called Items data.

**Section 2 – data**

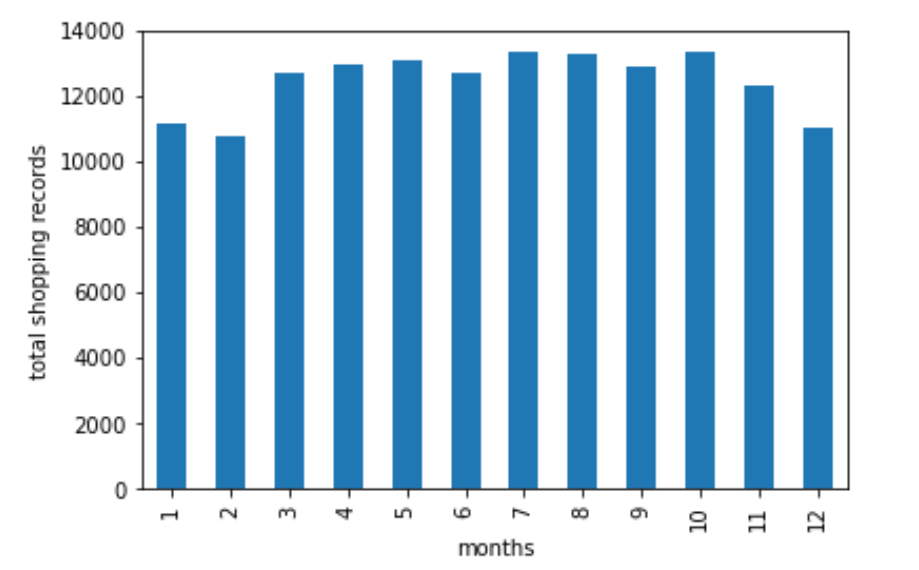
The different observations made on the given data are as follows

From the orders table, I have observed that the minimum number of orders made is on 2015- 11- 24 which is 27 and the maximum number of orders were made on 2012-01-24 which is 120.

When plotted the graph with the years with the number of transactions, there is a great drop in the orders in the year 2016, but in the rest of the years, the number of orders remained similar with the lower boundary as 80 and upper boundary as 120.

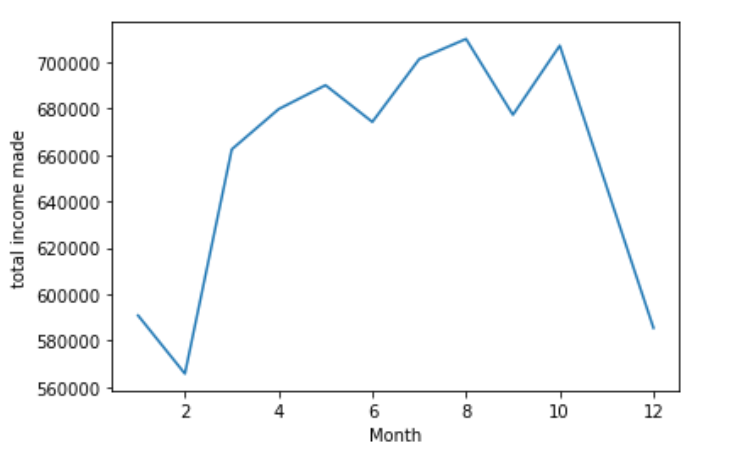


To understand the trends of the sales, I have plotted the graph that takes the monthly results of all years and shows the results



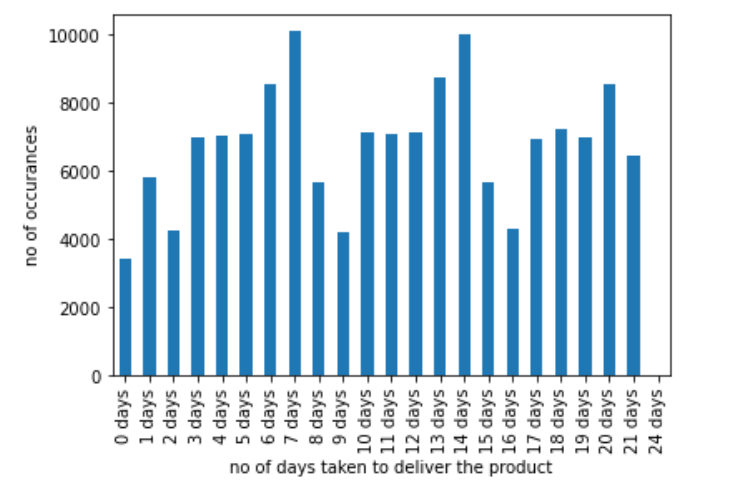
The bar graph describes the total shopping records created in different months overall years. It can be observed that the highest number of sales has occurred in October which is 13336 and the lowest number of sales has occurred in the month of February which is 10736. There is no sudden drop and leaps in the sales over the months but there is a continuous drop in sales from October to December.

The total income generated by the company on the monthly basis is calculated by grouping the months and adding the **total** of each month and the plot formed is as below.



The plot shows that the maximum income has made on 8 (August) that is 709967 and the minimum income was made on 2(February) that is 565683.96. from last two graphs it is observed that though the maximum number of sales happened in October, the highest income was made on August which means there are many products with the higher price was sold out in August.

In the same table, I have analyzed the days taken by the company to deliver the product to the customers,



The plot describes that most of the products like 10097 products are delivered in 7 days and 10028 products are delivered in 14 days. But there are 25 products which have taken 24 days to get delivered to the customers. So, the average number of days taken by the company to deliver a product is 11.

Also from the table, I have found that the dispatched data is always greater than the order date, so there are no mistakes in the data recorded for those fields.

**Customer table analysis:**

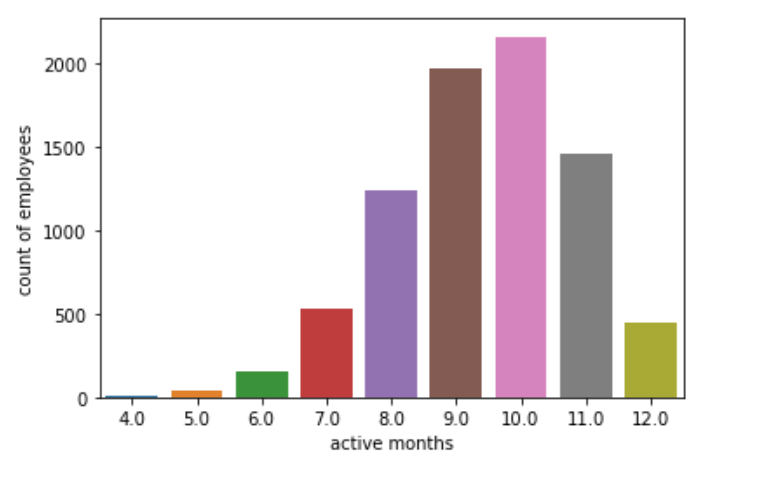
The customer table is used to understand the purchase history of people of different gender and different ages as follows:

The below plot shows the number of orders made by each category of the customer in different months over the years.

4000 
3000 
Month 

From the plot, it can be observed that the purchase behavior of the “Miss” category remained similar in all the months and the category “Ms” though there are small changes in the distribution but doesn’t show much difference. The Mrs category has the highest count of purchases in the month of October and the Mr category has also made the highest number of purchases in the month of October followed by August.

Another interesting pattern I have found during the analysis is the activity of the users that is how active the users are over the months. This data helps in targeting the users whose activity is high to make good profits in the business.

The plot drawn between the number of months user remained active vs the count of employees is as follows:

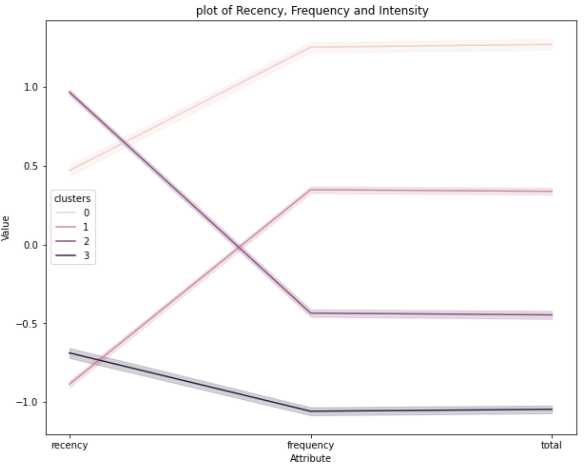
The number of customers who remained active the 12 months is 451 and the maximum number of customers have remained active for 10 months is 2157 and there is only 9 customer who remained active for only 4 months. So, based on this data the business domains can form insights on how to increase production in the business.

**The RFI and cluster distribution of the available user data:**

I have divided the customers into 4 clusters based on the RFI information calculated, the RFI or RFM indicates recency, frequency, and Intensity.

Recency is how recently did the customer made the purchase, Frequency indicates how often the customers made purchases and Monetary value represents how much the customer has spent [4].

I have calculated the RFI factors of the customers by applying K means to cluster [5] the whole set of customers into 4 clusters. The snake plot drawn using these factors gives the following observation



In cluster 3 it is observed that recency is greater than frequency and the total amount spent. The customers under this cluster had purchased products recently and don’t have frequent actions and spent more money on purchasing.

Cluster 2 has very high recency but the frequency and total are comparatively less. The customers under these clusters are new customers who might or might not continue purchasing but these customers also have to be targeted by ads to turn them into loyal customers.

Cluster 1 has very low recency when compared to the other two factors, so the customers in this cluster though don’t have any recent activity but they are frequent buyers and contributed to the good production for the company.

Cluster 0 has a very high frequency and total amount values, the recency is also high which is required to name the customers under this group as loyal customers. The customers of this type have to be given much importance as they contribute most of the income and production value to the company.

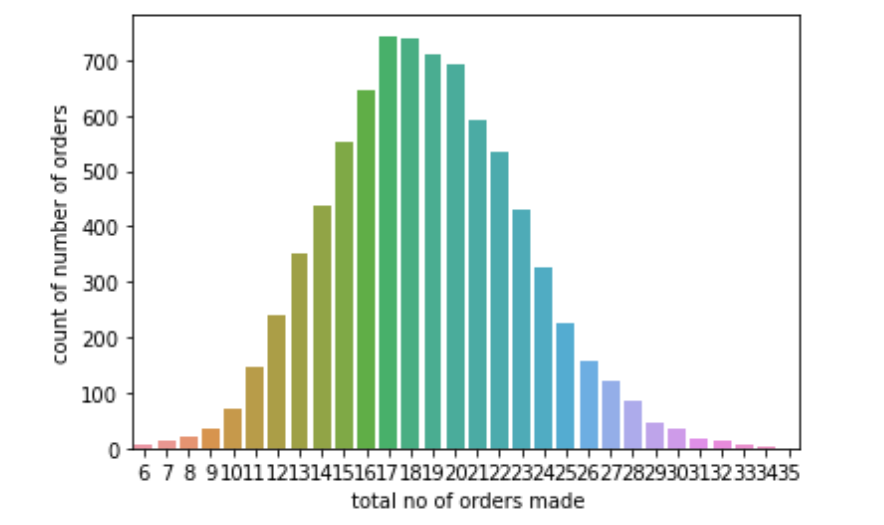
**Section 3: Real or Manufactured?**

After having a thorough understanding of all the given tables, I came to the conclusion that the given data is Manufactured. The reasons behind this conclusion are:

The customer ID is given to a customer based on the purchases and the orders table contains all the orders taken by the customers. There are only four columns in the customer table and the unique ID is provided to the customer based on their name and all orders made by the customer connect to the unique ID given to the customer. Suppose, if we have a new customer with the same first name and last name, the orders made by him fall under the existing customer with the same name. so, this is not the proper way data has to be stored.

The orders table contains the customer ID where the maximum number of transactions made by the customer is 35 and the minimum is 6. I have analyzed the customer (**customer ID** – 5912) who has the purchase record of 35 products and when applied joins among orders table and Items table, I have found that the items of the same type have been purchased 4 times in which 2 times in the same month. Also, the product “CD ROM” is a reusable product and is not required to buy the same product multiple times.

The overall distribution of the total number of orders made and the count of total orders made gives the distribution as follows

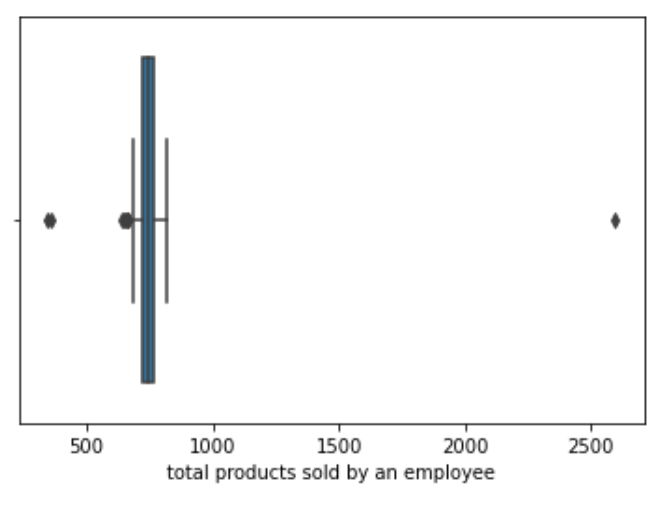


I have observed the normal symmetric distribution of this plot, where the average number of purchases made by a customer over all the years is 19.

The outlier observed in Employees handling the orders gives another reason to justify that the given data is manufactured.

Though there are only Employee names are given in the employee table, the relation of the employee table with the orders table helps us to understand the performance and products sold by each employee.

The boxplot for the 200 employees involved in handling the orders are as follows



From the above plot, it is observed that they are three outliers in which an employee with employee ID -3 and the name “Owen Edwards” has worked on **2596** orders which is far greater than the other employee's work. The employees with employee ID 4 and 2 have worked on 343 and 358 orders respectively which considers them as an outlier.

It’s not realistic to have an employee who worked on 2596 records which is far greater than the previous count that is 814.

The amount of income made by the company shows many discrepancies when calculated manually as follows:

The items table contains the total field which contains the total amount spent on the order with **Order No.**The total amount spent on all the orders in the orders table is **7889990.**

The orderInfo table contains Item No and Quantity and the items table contains Cost Price and Nominal Sale Price. On joining these two tables using Item No, I can manually calculate the total income made by the company.

The Cost Price is the amount spent by the company to manufacture or purchase the product and the Nominal Sale Price is the amount with which the products are sold by the company. The sum of the difference between Nominal Sale Price to the Cost Price gives the total profit made by the company. The total profit made by the company is **301442.50.**

*sum(orderinfo\_items["Nominal Sale Price"]\*orderinfo\_items["Quantity"] -orderinfo\_items["Cost Price"]\*orderinfo\_items["Quantity"] )*

In that case, the sum of Nominal sale price multiplied with quantity should result in the total income made by the company. The total obtained by performing the above calculation is 6813867.60, but the total obtained by adding the total amount in the orders table is **7889990**.

This shows the discrepancy obtained in calculating the over all data which is not possible with the real data.

**Conclusion:**

To conclude, the data provided is manufactured and there are many discrepancies in storing customer data and the total income made by the company. The total profit made by the company over 6 years starting from 2011 to 2016 is 301442.

And the total number of employees is 200, on dividing the profit with total employees it gives an amount of 1507 to each employee for the overall 6 years which is a very less profit to pay employees and maintain the business steadily.

**References:**

1. McGilvray, D. & Thomas, G. (2008) – ‘Definitions of Data Category’, *Executing Data Quality Projects: Ten Steps to Quality Data and Trusted Information.* Available at: <https://booksite.elsevier.com/9780123743695/10steps_DataCategories.pdf> (Accessed: 31 March 2021)
2. Borek, A. & Woodall, P. (2014) ‘Transactional Data’, [*Total Information Risk Management*](https://www.sciencedirect.com/book/9780124055476/total-information-risk-management)*,* pp 85-157.
3. Khajvand, M., et. al. (2011), ‘Estimating customer lifetime value based on RFM analysis of customer purchase behavior: Case study,’ *Procedia Computer Science* (3), pp 57-63.
4. Ferguson, M. (2014) ‘Big Data- Why Transaction Data is Mission Critical To Success’, *The Role of Transaction Data in a Big Data Environment*, V4.
5. Gustriansyah, R., Sensuse, D.I. & Ramadhan, A. (2017) ‘A Sales Prediction Model Adopted the Recency Frequency-Monetary Concept’, *Indonesian journal of Electrical Engineering and Computer Science*, Vol 6, No. 3, pp. 711-720. Doi: 10.11591/ijeecs.v6.i3.

**Appendix:**

**# importing libraries**

import pandas as pd

import numpy as np

import seaborn as sns

import sklearn

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans

customers = pd.read\_excel("CUSTOMERS.xlsx")

employees = pd.read\_excel("EMPLOYEES.XLSX")

items = pd.read\_excel("ITEMS.XLSX")

orderInfo = pd.read\_excel("ORDERINFO.XLSX")

orders = pd.read\_excel("ORDERS.XLSX")

print(customers.columns)

print(employees.columns)

print(items.columns)

print(orderInfo.columns)

print(orders.columns)

**#quick overview of tables:**

orderInfo.head()

orderInfo["Item No"].unique().size

orderInfo["Item No"].size

**#orders with quantity greater than 3**

orderInfo.loc[orderInfo.Quantity>1]

**#Total number of Employees**

orders["Employee Code"].unique()

**#total number of employees involved in orders page:**

orders["Employee Code"].value\_counts()

**Joins applied on data sets:**

orderdetails = pd.merge(orderInfo, orders, how="outer", on="Order No")

orderdetails.head(5)

orderinfo\_items = pd.merge(orderInfo,items, how="outer", on="Item No")

orderinfo\_items.sort\_values(by="Order No", ascending=True,inplace=True)

**### findind the highest no of products sold**

orderInfo["Item No"].value\_counts().sort\_values(ascending=True)

**### Orders table analysis**

**## total order distribution over years**

orders["Order Date"].value\_counts().sort\_values(ascending=False).plot(figsize = (10,4))

plt.xlabel("Years")

plt.ylabel('No of Orders')

orders["Order Date"].sort\_values(ascending=False)

**## understanding the customer purchases through the orders table**

orders["Customer ID"].value\_counts().mean()

total\_customer\_purchases = orders["Customer ID"].value\_counts()

**## count of sum of number of orders made**

sns.countplot(total\_customer\_purchases)

plt.xlabel(" total no of orders made")

plt.ylabel('count of number of orders')

**### stratification of purchases using gender**

customers["Title"].value\_counts()

order\_item= pd.merge(orderInfo, orders, how="inner", on="Order No" )

order\_item= order\_item[["Item No","Customer ID"]]

order\_item\_customer = pd.merge(order\_item, customers, how="inner", on ="Customer ID" )

order\_item\_customer\_final = pd.merge(order\_item\_customer, items, on="Item No", how="outer")

order\_item\_customer\_final= order\_item\_customer\_final[["Title","CD ROM Title"]]

order\_item\_customer\_final['Title'].sort\_values().value\_counts()

order\_item\_customer\_final['Title'].sort\_values().value\_counts().plot.pie(autopct="%.1f%%")

**## books bought by different customer category**

order\_item\_customer\_final.loc[order\_item\_customer\_final["Title"]=="Mr"]["CD ROM Title"].value\_counts().plot(kind="bar", figsize=(15,4))

order\_item\_customer\_final.loc[order\_item\_customer\_final["Title"]=="Mrs"]["CD ROM Title"].value\_counts().plot(kind="bar", figsize=(15,4))

order\_item\_customer\_final.loc[order\_item\_customer\_final["Title"]=="Ms"]["CD ROM Title"].value\_counts().plot(kind="bar", figsize=(15,4))

order\_item\_customer\_final.loc[order\_item\_customer\_final["Title"]=="Miss"]["CD ROM Title"].value\_counts().plot(kind="bar", figsize=(20,4))

**### montly trends of orders**

pd.to\_datetime(orders.iloc[1,3:4])["Order Date"].month

random\_date= []

for x in range(len(orders)):

#random\_date.append(pd.to\_datetime(x["Order Date"]).month)

##print(x)

random\_date.append(pd.to\_datetime(orders.iloc[x,3:4])["Order Date"].month)

random\_date = np.array(random\_date)

random\_date = pd.Series(random\_date)

random\_date.value\_counts().sort\_index().plot(kind = "bar")

plt.ylabel("total shopping records")

plt.xlabel('months')

random\_date.value\_counts().sort\_values()

**### male vs female monthly shopping patterns**

new\_orders["Month"]= random\_date

new\_orders\_customers= pd.merge(new\_orders, customers, on = "Customer ID", how="left")

new\_orders\_customers.groupby(["Month","Title"]).size()

fig, ax = plt.subplots(figsize=(10,4))

sns.countplot(data=new\_orders\_customers,x="Month", hue="Title",ax=ax)

**### checking if order date is greater than dispatch date**

orders[orders["Order Date"]> orders["Dispatch Date"]]

**### finding the maximum income made**

new\_orders\_customers.groupby(["Month"])["total"].sum().sort\_values()

new\_orders\_customers.groupby(["Month"])["total"].sum().sort\_index().plot(kind="line")

plt.ylabel("total income made")

**### employee customer relationship**

orders["Employee Code"].value\_counts().sort\_values()

employees.loc[employees["Employee Code"]==3]

emp\_data=orders["Employee Code"].value\_counts()

sns.boxplot(emp\_data)

plt.xlabel("total products sold by an employee")

**### delivery time analysis**

orders["Delivery Time"]= orders["Dispatch Date"]- orders["Order Date"]

orders["Delivery Time"].value\_counts().sort\_values()

min(orders["Delivery Time"].value\_counts().sort\_index())

max(orders["Delivery Time"].value\_counts().sort\_index())

np.mean(orders["Delivery Time"].value\_counts().sort\_values())

deliveryTime = orders["Delivery Time"].value\_counts().sort\_values()

x= orders["Delivery Time"].value\_counts().sort\_index().plot(kind= "bar")

x.set\_xticklabels(["0 days","1 days","2 days","3 days","4 days","5 days","6 days","7 days","8 days","9 days","10 days","11 days","12 days","13 days","14 days","15 days","16 days","17 days","18 days","19 days","20 days","21 days","24 days"])

plt.xlabel("no of days taken to deliver the product")

plt.ylabel("no of occurances")

**### finding out customer activity status**

orders["Customer ID"].value\_counts().sort\_index()

dfMonths= pd.DataFrame(columns=["customer ID","months"])

res=[]

for a in range(1,len(customers)+1):

dfMonths.loc[a,"customer ID"]=a

z=orders[orders["Customer ID"]==a]["Order Date"]

for b in z:

res.append(b.month)

dfMonths.loc[a,"months"]=res

res=[]

**### counting the number of months the customers remained active**

ls=[]

months=[1,2,3,4,5,6,7,8,9,10,11,12]

for val in range(1,len(customers)+1):

ls=(dfMonths[dfMonths["customer ID"]==val]["months"].array)[0]

count=0

for a in months:

if a in set(ls):

count=count+1

dfMonths.loc[val,"active months"] =count

dfMonths["active months"].value\_counts().sort\_index()

sns.countplot(dfMonths["active months"])

plt.ylabel("count of employees")

**# RFI Calculation**

monetary =orders.groupby(["Customer ID"])["total"].sum()

monetary=monetary.reset\_index()

frequency = orders["Customer ID"].value\_counts()

frequency=frequency.reset\_index()

frequency.rename(columns={'Customer ID':'frequency','index':'Customer ID'},inplace=True)

max\_date =max(orders["Order Date"])

recency=pd.DataFrame()

recency["date diff"]=max\_date - orders["Order Date"]

**## merging all three tables**

RFM\_data= pd.merge(monetary,frequency, how="outer", on="Customer ID")

RFM\_data["recency"]=recency["date diff"].dt.days

sns.boxplot(data= RFM\_data[['total','frequency','recency']])

**## there are outliers observed in total amount feature, so lets remove the outliers**

Q1 = RFM\_data['total'].quantile(0.05)

Q3 = RFM\_data["total"].quantile(0.95)

IQR = Q3 - Q1

rfm = RFM\_data[(RFM\_data['total'] >= Q1 - 1.5\*IQR) & (RFM\_data["total"] <= Q3 + 1.5\*IQR)]

**## rescaling the data so that all the values lies in the proper boundry**

rfm\_df = RFM\_data[['total', 'frequency', 'recency']]

scaler = StandardScaler()

rfm\_df\_scaled = scaler.fit\_transform(rfm\_df)

rfm\_df\_scaled.shape

rfm\_df\_scaled = pd.DataFrame(rfm\_df\_scaled)

rfm\_df\_scaled.columns = ['total', 'frequency', 'recency']

rfm\_df\_scaled.head()

**### k means for understanding RFM**

kmeans = KMeans(n\_clusters=4, max\_iter=50)

kmeans.fit(rfm\_df\_scaled)

rfm\_df\_scaled["clusters"]= kmeans.labels\_

rfm\_df\_scaled.head()

**### box plot for cluster vs total**

sns.boxplot(x="clusters", y="total", data=rfm\_df\_scaled)

**### cluster vs frequency**

sns.boxplot(x="clusters", y="frequency", data=rfm\_df\_scaled)

**### clusters vs recency**

sns.boxplot(x="clusters", y="recency", data=rfm\_df\_scaled)

from sklearn.cluster import KMeans

kmeans = KMeans(n\_clusters= 4)

label = kmeans.fit\_predict(rfm\_df\_scaled)

print(label)

**### line plot to compare Recency, Intensity and total amount**

data\_melt = pd.melt(rfm\_df\_scaled.reset\_index(),

id\_vars='clusters',

value\_vars=['recency', 'frequency','total'],

var\_name='Attribute',

value\_name='Value')

# Building the snakeplot

plt.title(' plot of Recency, Frequency and Intensity')

sns.lineplot(x="Attribute", y="Value", hue='clusters', data=data\_melt)

**### outlier in total amount spend**

orders["total"].agg(['max',"min","mean"])

sns.boxplot(orders["total"])

**### work on orderInfo and items to estimate the profits made by the company**

orderinfo\_items= pd.merge(orderInfo, items, how="inner", on="Item No")

sum(orderinfo\_items["Nominal Sale Price"] - orderinfo\_items["Cost Price"])

sum(orderinfo\_items["Nominal Sale Price"]\*orderinfo\_items["Quantity"] -orderinfo\_items["Cost Price"]\*orderinfo\_items["Quantity"] )

sum(orders["total"])

sum(orderinfo\_items["Nominal Sale Price"])

sum(orderinfo\_items["Cost Price"]\*orderinfo\_items["Quantity"])

sum(orderinfo\_items["Nominal Sale Price"]\*orderinfo\_items["Quantity"])