INTERNATIONAL UNIVERSITY
VIETNAM NATIONAL UNIVERSITY HCMC

May 23, 2021

School of Computer Science and Engineering

GROUP PROJECT REPORT

INTRODUCTION TO DATA MINING

Lecturer: Dr. Nguyen Thi Thanh Sang
Lab lecturer: MSc. Nguyen Quang Phu

Group H members:
Nguyen Huynh Phuong Thanh ITITIU18115

Tran Cong Man ITITIU18285

Dang Ngoc Minh Huy ITITIU18052

Phan Thi Nhu Yen ITDSIU18048

Nguyen Thi Truong An ITDSIU18002

Contents

List of Figures

1 Introduction

1.1 Background

Data mining is a process of analyzing large pre-existing data from different perspective and summarizing it into useful information that can be used to increase revenue, cost cutting or both. It also can be considered as a computational process of discovering patterns in large datasets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems.

Manual finding of Terabytes or Petabytes of data is very difficult, Data Mining helps to find information automatically. Therfore, it becomes important for today's business analysis and decision making.

1.2 Motivations

Having the desire not solely to fulfill the requirements for the Introduction to Data Mining course we take this semester, but to have a better understanding of the concepts in this state-of-the-art course, we decided to have a go and build a data mining framework from scratch.

Given a certain training dataset following a specific data format, our framework should be able to generate a dataset of event sequences, build a clustering process and a sequence mining process as a prediction engine which will be used to predict next events in the application domain.

We will then test our data mining framework with the real-world e-commerce dataset to evaluate the quality of the processes.

In order to build this classification framework, we need to finish four steps as follows:
- Step 1: Extract sequential datasets of events based on timestamps and generate training and testing datasets.
- Step 2: Implement a clustering algorithm, e.g., DBSCAN, and apply it to the sequential datasets. (able to refer to the Weka library)
- Step 3: Implement a sequence mining algorithm (or association rules), e.g. FPGrowth, to build a prediction model based on the sequences in the (clustered) training datasets.
- Step 4: Test the prediction model with and without clustering. Evaluate its performance and write a report.

2 Data Preprocessing

Link of the e-commerce dataset:https://www.kaggle.com/carrie1/ecommerce-data

In this initial step, we have the input is a raw dataset and the expected output are transactions.

2.1 Data Analysis: step by step approach

The purpose of this data analysis is to find key insights and derive meaning from them. We will try to derive some business-impacting insights by trying to answer relevant questions like:

- Is the company's performance improving or degrading over time?
- What are some important trends visible in the sales data and insights?
- How can we measure our performance in terms of customer acquisition and building customer loyalty?
- Can we take some initiatives based on the data to increase sales?
- Based on data can we avoid out-of-stock situations?
- What kind of customer does typically buy from us?

2.1.1 Data Description

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

This is a transnational data set that contains all the transactions occurring between **01/12/2010** and **09/12/2011** for a UK-based and registered non-store online retail. The company mainly sells unique all-occasion gifts. Many customers of the company are wholesalers

Column	**Description**
InvoiceNo	Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transaction. **If this code starts with the letter 'c', it indicates a cancellation**
StockCode	Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each distinct product
Description	Product (item) name. Nominal
InvoiceDate	Invoice Date and time. Numeric, the day and time when each transaction was generated
UnitPrice	Numeric, Product price per unit in sterling
CustomerID	Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer
Country	Country name. Nominal, the name of the country where each customer resides

```
df.describe()
```

	Quantity	UnitPrice	CustomerID
count	541909.000000	541909.000000	406829.000000
mean	9.552250	4.611114	15287.690570
std	218.081158	96.759853	1713.600303
min	-80995.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13953.000000
50%	3.000000	2.080000	15152.000000
75%	10.000000	4.130000	16791.000000
max	80995.000000	38970.000000	18287.000000

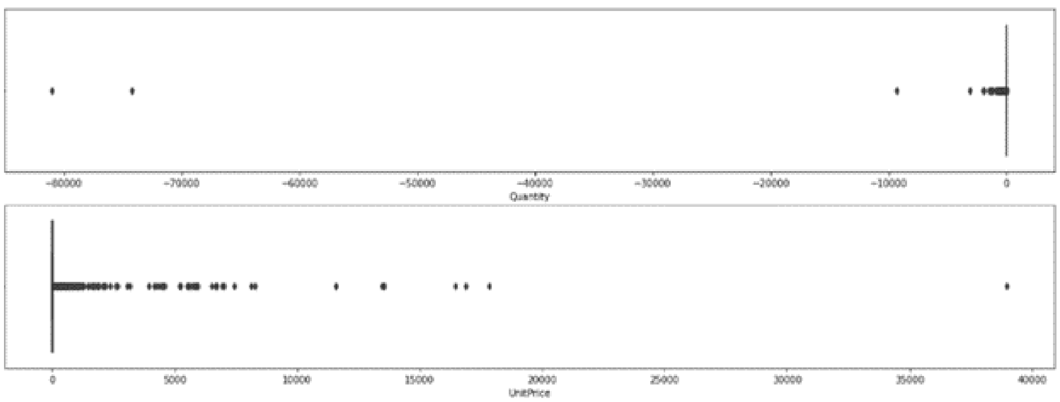
As described in the dataset, the number of transactions is too large and noisy to train the model. Some data is unlogical like the negative of Quantity, UnitPrice. There is also a lack of data of CustomerID.

2.1.2 Analyzing negative Quantity and UnitPrice

```
df1 = df[(df['InvoiceNo'].str[0] == "C")|(df['InvoiceNo'].str[0] == 'c')]['Quantity']
```

```
df2 = df[(df['InvoiceNo'].str[0] == "C")|(df['InvoiceNo'].str[0] == 'c')]['UnitPrice']
```

```
fig, ax = plt.subplots (nrows = 2, figsize = (20,7))
sns.boxplot(df[(df['InvoiceNo'].str[0] == "C")|(df['InvoiceNo'].str[0] == "c")]['Quantity'], ax = ax[0])
sns.boxplot(df[(df['InvoiceNo'].str[0] == "C") | (df['InvoiceNo'].str[0] == "c")]['UnitPrice'], ax = ax[1])
```



The reason that causes negative of Quantity is canceled orders and there is no negative of UnitPrice of the cancelled order. Next, we will check if the negative Quantity without ‘c’ or ’C’ in InvoiceNo

```
neg_quantity = df[df['Quantity'] < 0]
neg_quantity_without_C = neg_quantity[(neg_quantity['InvoiceNo'].str[0] != "C") & (neg_quantity['InvoiceNo'].str[0] != 'c')]
```

```
Negative Quantity without 'C' and 'c' in InvoiceNo
 Unit Prices: [0.]
 CustomerIDs: [nan]
```

[∗] Therefore, we don’t need to worry about the negative quantity without C or c in front of InvoiceNo as their unit price is 0. They will not affect the calculation later on. These entries have no CustomerID associated with them. However, to make sure that the negative Quantity comes from previous order cancellations, we need to prove the hypothesis being accepted.

HYPOTHESIS: Rows with negative quantities means that the order was previously ordered and canceled later on. If this true, the majority of negative quantities must satisfy the following conditions:

1. CustomerID (if exist) must match
2. Quantity ordered must smaller than or equal to Quantity canceled
3. Order date must before Cancel date

```python
def check_hypothesis_cancelled_order (df):
  failed = 0
  passed = 0
  neg_quantity = df[df['Quantity'] < 0]
  pos_quantity = df[~df['Quantity'] < 0]
  for index in neg_quantity.index:
    if (neg_quantity['CustomerID'][index]):
      p = pos_quantity[
          (pos_quantity['CustomerID'] == neg_quantity['CustomerID'][index])
          & (pos_quantity['Quantity'] <= abs(neg_quantity['Quantity'][index]))
          & ((pos_quantity['InvoiceDate'] - neg_quantity['InvoiceDate'][index]).dt.total_seconds() >= 0)
      ]
      if (len(p) == 0):
        failed +=1
      else:
        passed +=1

  if (failed > passed):
    print("Hypothesis rejected")
    print("Failed Counts: " + str(failed) + " Passed Counts: " + str(passed))
    print("Approximately " + str(int(failed/(failed + passed)*100)) + "% rows did not satisfy the condition")
  else:
    print("Hypothese accepted")
    print("Failed Counts: " + str(failed) + " Passed Counts: " + str(passed))
    print("Approximately " + str(int(passed/(failed + passed)*100)) + "% rows satisfy the condition")
```

Result:

```
Hypothese accepted
Failed Counts: 4665 Passed Counts: 5959
Approximately 56% rows satisfy the condition
```

[∗] **We can surely conclude that negative quantity entries are for some previous order cancellation**

Negative Price: There are 2 transactions without the id of the customer make the prices negative

```python
neg_price = df[df['UnitPrice'] < 0]
neg_price
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
299983	A563186	B	Adjust bad debt	1	2011-08-12 14:51:00	-11062.06	NaN	United Kingdom
299984	A563187	B	Adjust bad debt	1	2011-08-12 14:52:00	-11062.06	NaN	United Kingdom

2.2 Data Cleaning as a process

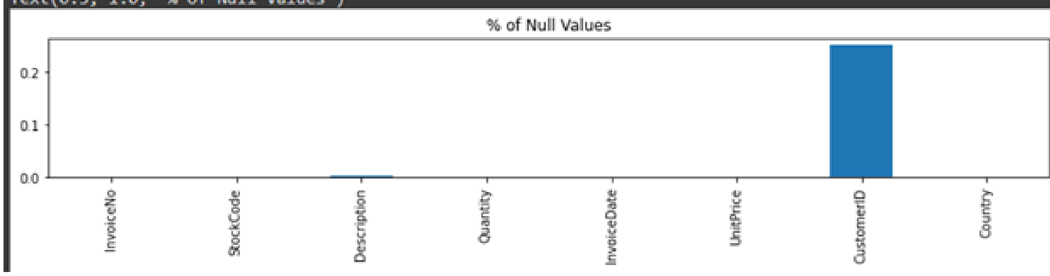
2.2.1 Duplicated Deletion

```python
df.drop_duplicates(inplace=True)
```

2.2.2 Handling Missing Values

```
fig,ax = plt.subplots(figsize = (14,2))
(df.isnull().sum()/len(df)).plot.bar(ax = ax)
ax.set_title("% of Null Values")
```

```
Text(0.5, 1.0, '% of Null Values')
```



There are missing values in Description and CustomerID. Nearly 25% of value missing in CustomerID and 0.3% value missing in Description. We can not do anything with CustomerID

Check if each StockCode has a unique Description

```
x = pd.DataFrame(df.groupby("StockCode")['Description'].value_counts())
y = x.droplevel(level = 1).index
y = y[y.duplicated()]
test = df[['StockCode','Description']]
test = test.drop_duplicates()
test1 = test[test['StockCode'].isin(y)]
test2 = pd.DataFrame(test1.groupby("StockCode")['Description'].value_counts())
test2.head(20)
```

Result:

		Description
StockCode	Description	
10080	GROOVY CACTUS INFLATABLE	1
	check	1
10133	COLOURING PENCILS BROWN TUBE	1
	damaged	1
15058A	BLUE POLKADOT GARDEN PARASOL	1
	wet/rusty	1
15058C	ICE CREAM DESIGN GARDEN PARASOL	1
	wet/rusty	1
16008	SMALL FOLDING SCISSOR(POINTED EDGE)	1
	check	1
16045	POPART WOODEN PENCILS ASST	1
	check	1
16156L	WRAP CAROUSEL	1
	WRAP, CAROUSEL	1
16162M	THE KING GIFT BAG 25x24x12cm	1
	alan hodge cant mamage this section	1
16168M	FUNKY MONKEY GIFT BAG MEDIUM	1
	found	1
16169E	WRAP 50'S CHRISTMAS	1
	check	1

So some products have varied Descriptions but belong to the same StockCode. To solve this, create the Product Name that corresponding with StockCode and use this column to predict the buying trends of Customer.

Find the Product Name

```python
[ ] def get_product_name(x):
        max_upper_count = 0
        product_name = ''
        for i in x:
            if(i==i):  #To Check for NaN
                count = 0
                for letter in i:
                    if(letter.isupper()):
                        count = count+1
                if count>max_upper_count:
                    max_upper_count = count
                    product_name = i
        return product_name
```

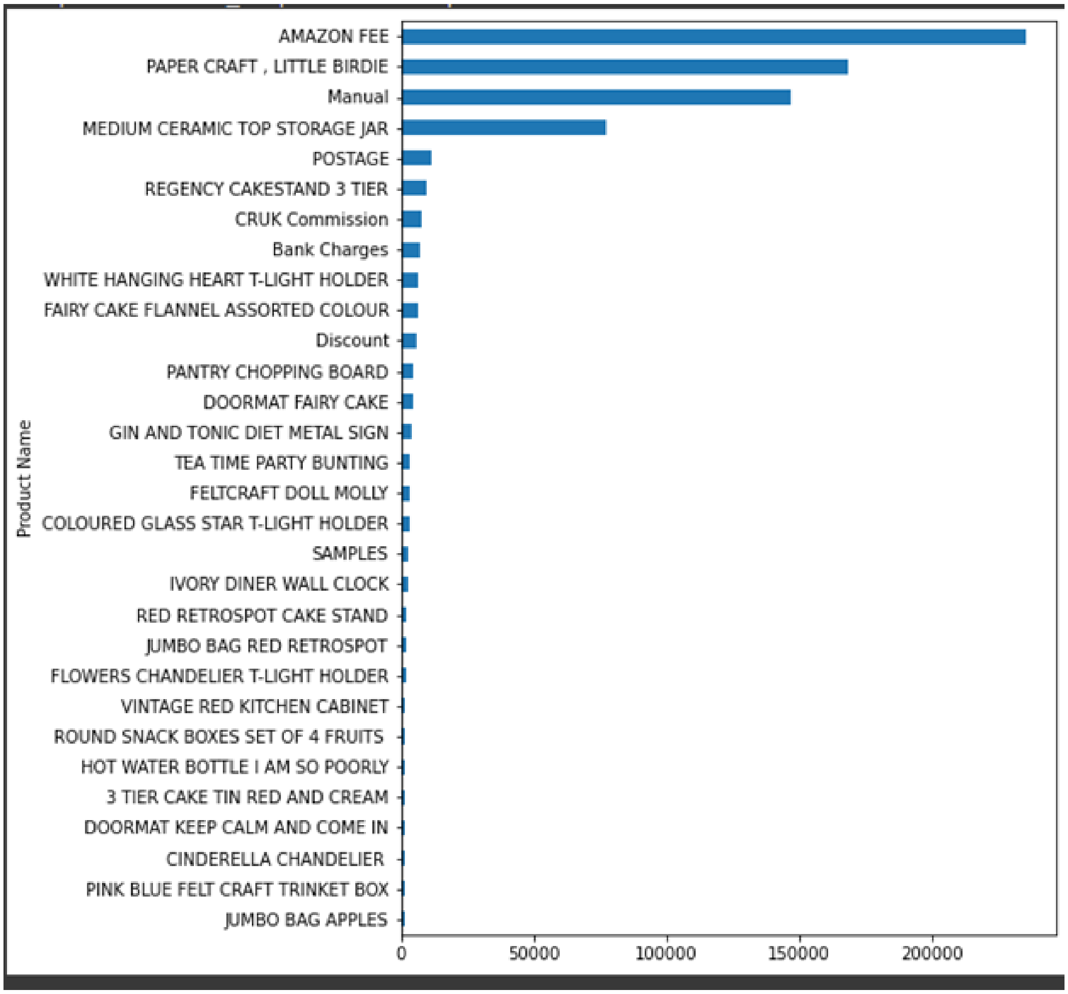
```python
[ ] grouped = df.groupby("StockCode")['Description'].unique()
    lookup = grouped.apply(get_product_name)
```

```python
[ ] df = df.join(other=lookup, on='StockCode', how='left', rsuffix='ProductName')
    df = df.rename(columns={'DescriptionProductName':'ProductName'})
```

The data after filling Product Name:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	TotalPrice	ProductName
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom	15.30	WHITE HANGING HEART T-LIGHT HOLDER
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34	WHITE MOROCCAN METAL LANTERN
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom	22.00	CREAM CUPID HEARTS COAT HANGER
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34	KNITTED UNION FLAG HOT WATER BOTTLE
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34	RED WOOLLY HOTTIE WHITE HEART.
5	536365	22752	SET 7 BABUSHKA NESTING BOXES	2	2010-12-01 08:26:00	7.65	17850.0	United Kingdom	15.30	SET 7 BABUSHKA NESTING BOXES
6	536365	21730	GLASS STAR FROSTED T-LIGHT HOLDER	6	2010-12-01 08:26:00	4.25	17850.0	United Kingdom	25.50	GLASS STAR FROSTED T-LIGHT HOLDER
7	536366	22633	HAND WARMER UNION JACK	6	2010-12-01 08:28:00	1.85	17850.0	United Kingdom	11.10	HAND WARMER UNION JACK
8	536366	22632	HAND WARMER RED POLKA DOT	6	2010-12-01 08:28:00	1.85	17850.0	United Kingdom	11.10	HAND WARMER RED RETROSPOT
9	536367	84879	ASSORTED COLOUR BIRD ORNAMENT	32	2010-12-01 08:34:00	1.69	13047.0	United Kingdom	54.08	ASSORTED COLOUR BIRD ORNAMENT

[∗] Now, each transaction is more clearly than the original



The chart shows the negative total price in this dataset. As the figure, AMAZON FEE, Paper, Craft, Manual,.. are some of the main contributors for negative priced entries.

Removing "C" Products

```
cancelled_df = df[(df['InvoiceNo'].str[0] == "C")]
df = df[~(df['InvoiceNo'].str[0] == 'C')]
cancelled_df = cancelled_df.reset_index(drop=True)
```

Removing negative of Unit Price

```
df.drop(df[df['UnitPrice'] <= 0].index, inplace=True)
```

Transform Country and ProductName data to numeric. We need the ProductName attribute to Cluster and train the model so nominal type cannot do it. So it is necessary to Label Encoder them.

```
from sklearn.preprocessing import LabelEncoder
labelEncoder = LabelEncoder()
df1['Country'] = labelEncoder.fit_transform(df1.Country)
df1['ProductName'] = labelEncoder.fit_transform(df1.ProductName)
```

The data after transforming

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	TotalPrice	ProductName
536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	36	15.30	3615
536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	36	20.34	3623
536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	36	22.00	845
536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	36	20.34	1749
536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	36	20.34	2691
...	...	...	...	...	...	...	...	...	...
581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	2011-12-09 12:50:00	0.85	12680.0	13	10.20	2187
581587	22899	CHILDREN'S APRON DOLLY GIRL	6	2011-12-09 12:50:00	2.10	12680.0	13	12.60	686
581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	2011-12-09 12:50:00	4.15	12680.0	13	16.60	692
581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	2011-12-09 12:50:00	4.15	12680.0	13	16.60	691
581587	22138	BAKING SET 9 PIECE RETROSPOT	3	2011-12-09 12:50:00	4.95	12680.0	13	14.85	265

Making and Removing Outliers

```
import numpy as np

i = 0
quartile_1, quartile_3 = np.percentile(df1['Quantity'],[25,75])
iqr = quartile_3 - quartile_1
lower_bound = quartile_1 - (iqr*1.5)
upper_bound = quartile_3 + (iqr*1.5)

while i <= len(df1) - 1:

  if df1.loc[i,'Quantity'] > upper_bound:
    df1.loc[i,'Outlier'] = 1
  elif df1.loc[i,'Quantity'] < lower_bound:
    ddf1b.loc[i,'Outlier'] = 1
  else:
    df1.loc[i,'Outlier'] = 0
  i = i + 1
```

```
df1 = df1[df1['Outlier'] == 0]
df1 = df1.drop(columns = ['Outlier'])
df1
```

We mark the point that greater than upper bound and lesser than lower bound to the Outlier 1 and the other to the Outlier 0. Then we get all data of Outlier is 0 to remove the outliers.

3 Implement a clustering algorithm

In this step, the DBSCAN algorithm is used to cluster the whole dataset into smaller groups that have similar attributes. The purpose of clustering is to separate the large dataset that has not relevant so it leads the prediction model to more accuracy.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a popular unsupervised learning method utilized in model building and machine learning algorithms. It is known as a method to separate clusters of high density from clusters of low density. It identified the areas in the dataset that have a high density of observations independently with ones that are not dense with observations.

Because of the lack of ram to execute a huge dataset, in this project, we used 60% sample of the original dataset.

```
data.describe()
```

	InvoiceNo	Quantity	UnitPrice	CustomerID	Country	TotalPrice	ProductName
count	367029.000000	367029.00000	367029.000000	367029.000000	367029.000000	367029.000000	367029.000000
mean	560632.234889	7.09217	3.240146	15302.801326	33.866278	14.505232	1961.875879
std	13109.064363	6.66746	22.973773	1712.053582	6.832654	28.221008	1063.090417
min	536365.000000	1.00000	0.001000	12347.000000	0.000000	0.001000	0.000000
25%	549238.000000	2.00000	1.250000	13975.000000	36.000000	4.250000	1102.000000
50%	561894.000000	4.00000	1.950000	15182.000000	36.000000	10.500000	1904.000000
75%	572103.000000	12.00000	3.750000	16807.000000	36.000000	17.700000	2898.000000
max	581587.000000	26.00000	8142.750000	18287.000000	37.000000	8142.750000	3780.000000

The describe of original data

```
sampled_data.describe()
```

	InvoiceNo	Quantity	UnitPrice	CustomerID	Country	TotalPrice	ProductName
count	220217.000000	220217.000000	220217.000000	220217.000000	220217.000000	220217.000000	220217.000000
mean	560653.626759	7.098730	3.252035	15302.211310	33.858599	14.508188	1962.935391
std	13108.330895	6.673643	25.466426	1711.610632	6.842773	30.146585	1064.396266
min	536365.000000	1.000000	0.001000	12347.000000	0.000000	0.001000	0.000000
25%	549251.000000	2.000000	1.250000	13972.000000	36.000000	4.250000	1102.000000
50%	561899.000000	4.000000	1.950000	15184.000000	36.000000	10.500000	1906.000000
75%	572129.000000	12.000000	3.750000	16806.000000	36.000000	17.700000	2900.000000
max	581587.000000	26.000000	8142.750000	18287.000000	37.000000	8142.750000	3780.000000

Sample data (60%). As result, the distribution of the two datasets is similar so it does not affect much when dealing with sample data.

3.1 Splitting data into Testing set and Training set

Split test - train set

```
[ ] from sklearn.model_selection import train_test_split
    train, test = train_test_split(sampled_data, test_size = 0.1, random_state=42)
```

The initial dataset is split into a training set and testing set with the ratio of 9:1.

3.2 Selecting important features to cluster

Our prediction model analyzes the consumer buying trends to predict the next product with a high probability that the user will buy. It means that the model will predict what is the next event of the customer to suggest one logically

```
train_dbscan = sampled_data[['Quantity','UnitPrice','ProductName']]
```

```
train_dbscan
```

	Quantity	UnitPrice	ProductName
239249	1	16.95	533
273205	2	4.95	3456
36744	1	0.85	2186
54235	18	1.45	1430
82387	2	1.95	2329
...	...	...	...
176029	3	4.95	2963
12650	2	2.10	727
121928	25	0.42	3726
200359	2	12.75	1864
321914	2	0.83	3515

[*] **Quantity, UnitPrice, ProductName are attributes is chosen to be clustered.**

3.3 Data Cleaning: Scaling dataset

The dataset contains features highly varying in magnitude (Quantity, Unit Price, and Product Name) that lead to the computations is hard and has a significant error. So this dataset is scaled before doing cluster algorithm.

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler().fit(train_dbscan)
clusters_scaled = train_dbscan.copy()
train_dbscan_scaler = scaler.transform(train_dbscan)
```

```
train_dbscan_scaler
```

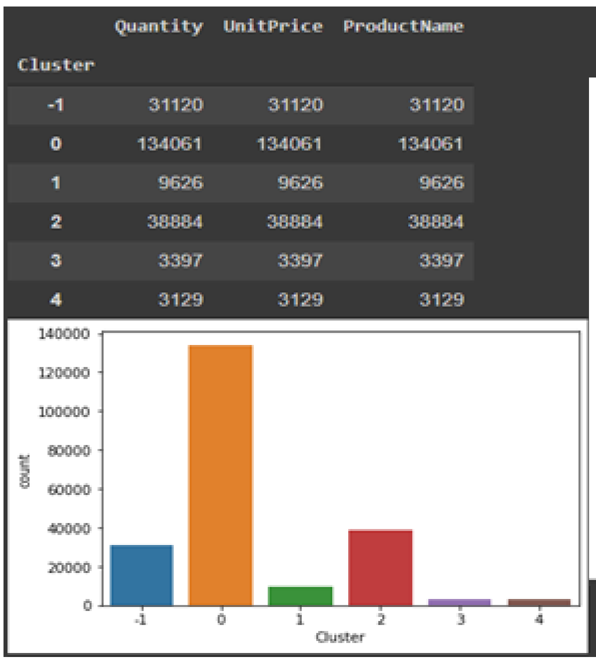
```
array([[-0.91385521,  0.53788452, -1.34342696],
       [-0.76401168,  0.06667481,  1.40273698],
       [-0.91385521, -0.09432184,  0.20956961],
       ...,
       [ 2.6823895 , -0.11120685,  1.65640248],
       [-0.76401168,  0.37296113, -0.09294998],
       [-0.76401168, -0.09510719,  1.45816759]])
```

We use Standardization (also called z-score normalization) to transform the above data such that the resulting distribution has a mean 0 and a standard deviation of 1.

$$Formula : x\prime = \frac{x - \bar{x}}{\sigma}$$

Where the denominator is the variance and x bar is the mean.

3.4 Result of Clustering



Cluster	Quantity	UnitPrice	ProductName
-1	31120	31120	31120
0	134061	134061	134061
1	9626	9626	9626
2	38884	38884	38884
3	3397	3397	3397
4	3129	3129	3129

Figure 1: Result of Clustering

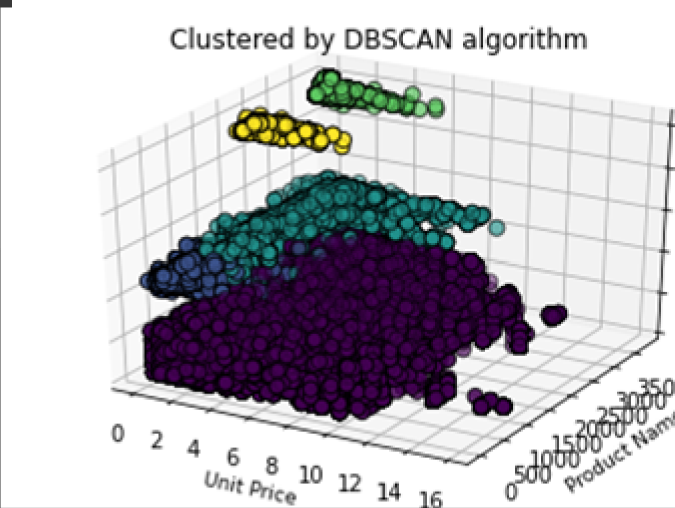


Figure 2: Clustered by DBSCAN algorithm

After clustering, 5 clusters are divided without balance instances of each of them. Compare with others, the first cluster account for most of the data with very high density. There still closed between the 3 most clusters.

Moreover, the outliers still large over 30.000 instances. To complete this step, we drop the outliers out of this dataset.

3.5 Advantages of using DBSCAN

- In this project, DBSCAN was used because of dealing with the thickness of the dataset and the ability to mark and remove outliers effectively.
- Can discover arbitrarily shaped clusters.
- Require just two points which are very insensitive to the ordering of the point

in the database
- It is great at separating clusters of high density versus clusters of low density within a given dataset.

3.6 Disadvantages of using DBSCAN

- Although it works well to solve density problems, DBSCAN might cause clusters with similar density and the difference between these clusters is slight.
- Fails to identify cluster if density varies and if the dataset is too sparse.

4 Implement a sequence mining algorithm

In this step, the FPGrowth is applied to implement an association rule and generate a set of association rules that predict the next event of transactions. It is an algorithm for frequent pattern mining that focuses on generating itemsets and discovering the most frequent them. It also helps to reduce the size of an itemset in the database.

Initially, our dataset has to be converted into transactions that contain a list of items. After executing this algorithm, a list of relevant items is out as rules, and the prediction is worked based on them.

4.1 Convert data into transactions and fit to the model

```
train = train_data.groupby('InvoiceNo')['ProductName'].apply(lambda x: list(set(x)))

train = pd.DataFrame(train, columns=['ProductName'])

model = FPGModel()
model.fit(data = train, itemsCol = 'ProductName', mSup=0.001, mConf=0.001)
model.show()
```

4.2 Result of FPModel

Model without Clustering

antecedent	consequent	confidence	lift	support
[132]	[134]	0.32894736842105265	39.620149827122546	0.001515059693351918
[132]	[131]	0.23684210526315788	75.15637651821862	0.001090842979213381
[667, 1881]	[2651]	0.3953488372093023	17.67927144387723	0.001030240591479...
[667, 1881]	[1887]	0.4418604651162791	14.075558947651972	0.001151445366947...
[962]	[8]	0.4	76.74883720930232	0.001575662081085...
[1703, 1880]	[1890]	0.29310344827586204	12.12155388471178	0.001030240591479...
[1703, 1880]	[1884]	0.3620689655172414	12.11866125760649	0.001272650142415...
[1703, 1880]	[1705]	0.29310344827586204	9.265325670498083	0.001030240591479...
[1703, 1880]	[1712]	0.3275862068965517	7.486842105263158	0.001151445366947...
[1703, 1880]	[1888]	0.3448275862068966	10.71563088512241	0.001212047754681...
[1703, 1880]	[1879]	0.3103448275862069	12.490243902439026	0.001090842979213381
[1703, 1878]	[1889]	0.3269230769230769	18.928272604588397	0.001030240591479...
[1703, 1878]	[1702]	0.38461538461538464	18.342596709648735	0.001212047754681...
[3358]	[2202]	0.1450381679389313	4.692695704235892	0.001151445366947...
[3358]	[1885]	0.13740458015267176	3.741440556269368	0.001090842979213381
[3358]	[3093]	0.13740458015267176	5.74003285341579	0.001090842979213381
[474]	[2676]	0.2761904761904762	28.48386904761905	0.001757469244288...
[474]	[2449]	0.3047619047619048	46.13647881170817	0.001939276407490...
[3630]	[2691]	0.10606060606060606	11.147172360548158	0.001272650142415...
[3630]	[685]	0.08585858585858586	9.259820426487092	0.001030240591479...

only showing top 20 rows

ProductName	prediction
[1395, 2691, 3623...	[1186, 2732, 2905...
[1510]	[1512, 2244, 2847...
[1865, 1177, 1652...	[527, 2244, 727, ...
[409, 2613]	[3743]
[1891, 3268, 2535...	[1706, 667, 669, ...
[1512]	[1871, 1749, 3509...
[3490, 3491, 1061...	[3678, 3674, 1186...
[3490, 3491, 1061...	[1865, 988, 46, 1...
[1512, 1510]	[2244, 2847, 2732...
[1700, 3110, 3309...	[1714, 1706, 1719...
[1679]	[1720, 1350, 2594...
[1153, 1058, 450,...	[644, 643, 3335, ...
[138, 721, 183, 1...	[2739, 3548, 312,...
[1102, 848, 1105,...	[1109, 3663, 2695...
[1680, 1706, 3403...	[2905, 1350, 2631...
[1865, 211, 979, ...	[1534, 3678, 2066...
[138, 2964, 3517,...	[385, 2202, 3064,...
[2779, 2732]	[1565, 2847, 2593...
[2272, 3031, 3287]	[3404, 265, 3408,...
[953, 2732, 3425]	[1565, 2847, 2593...

only showing top 20 rows

Figure 3: Model without Clustering

Model with Clustering

antecedent	consequent	confidence	lift	support
[132]	[133]	0.2345679012345679	66.24588477366255	0.001401593390380...
[132]	[135]	0.24691358024691357	56.73153379368068	0.001475361463558...
[132]	[134]	0.3950617283950617	37.19067215363511	0.002360578341693715
[132]	[131]	0.2716049382716049	66.94320987654321	0.001622897609914429
[1714, 1885]	[1888]	0.4827586206896552	24.238058748403578	0.001032753024491...
[2696, 1469]	[2698]	0.4827586206896552	51.127155172413794	0.001032753024491...
[1061, 1059]	[1058]	0.5	53.79365079365079	0.001180289170846...
[722, 2847]	[1622]	0.21875	12.618617021276595	0.001032753024491...
[722, 2847]	[2732]	0.28125	12.418973941368078	0.001327825317202...
[722, 2847]	[1619]	0.25	15.617511520737327	0.001180289170846...
[1588, 1593]	[1591]	0.6666666666666666	139.03589743589743	0.001327825317202...
[1588, 1593]	[1592]	0.7037037037037037	146.76011396011396	0.001401593390380...
[1588, 1593]	[1590]	0.6666666666666666	150.62222222222223	0.001327825317202...
[1588, 1593]	[1589]	0.5555555555555556	150.62222222222223	0.001106521097668...
[3630]	[722]	0.11518324607329843	4.6749224064959085	0.001622897609914429
[3630]	[2249]	0.09424083769633508	4.562602842183994	0.001327825317202...
[3630]	[1622]	0.08900523560209424	5.1342764843488915	0.001254057244024786
[3630]	[1888]	0.07329842931937172	3.680124103160752	0.001032753024491...
[3630]	[1487]	0.13612565445026178	6.9113085083436285	0.001917969902626...
[3630]	[1619]	0.11518324607329843	7.195502690182643	0.001622897609914429

only showing top 20 rows

ProductName	prediction
[1395, 2691, 3623...	[1507, 2802, 2233...
[1510]	[1506, 1509, 1507...
[1865, 526, 527, ...	[723, 2314, 280, ...
[409, 2613]	[3743]
[1512]	[1507, 2500, 2501...
[3490, 3491, 1061...	[1487, 3224, 988,...
[3490, 3491, 1061...	[990, 3663, 2962,...
[1512, 1510]	[1506, 1509, 1507...
[3309]	[2802, 3006, 695,...
[1153, 1058, 450,...	[644, 3615, 643, ...
[138]	[3006, 1887, 3042...
[848, 1105, 1530,...	[1106, 1110, 1651...
[1680]	[988, 3006, 52, 1...
[1865, 979, 1599]	[988, 981, 990, 9...
[385, 138, 142, 2...	[1059, 1058, 3093...
[3031]	[3404, 52, 2962, ...
[3490, 1058, 2691...	[990, 3663, 2962,...
[978, 250, 3630, ...	[722, 2249, 1622,...
[1512]	[1507, 2500, 2501...
[2433, 2317, 270,...	[1888, 1877, 1880...

only showing top 20 rows

Figure 4: Model with Cluster 0

antecedent	consequent	confidence	lift	support
[698, 688]	[965]	0.35294117647058826	58.524064171123	0.001096491228070...
[714, 707]	[708]	0.6363636363636364	112.32844574780059	0.001279239766081...
[714, 707]	[713]	0.6363636363636364	133.93006993006995	0.001279239766081...
[733]	[732]	0.37209302325581395	46.27484143763214	0.002923976608187...
[748, 746]	[747]	0.6666666666666666	93.53846153846153	0.001461988304093...
[834, 1079]	[604]	0.24	17.747027027027027	0.001096491228070...
[754]	[753]	0.2608695652173913	75.1304347826087	0.001096491228070...
[754]	[681]	0.30434782608695654	79.30434782608697	0.001279239766081...
[748, 747]	[746]	0.5714285714285714	69.48571428571428	0.001461988304093...
[323]	[619]	0.1111111111111111	11.054545454545455	0.001279239766081...
[323]	[617]	0.09523809523809523	9.832884097035041	0.001096491228070...
[323]	[515]	0.1111111111111111	30.4	0.001279239766081...
[323]	[272]	0.1111111111111111	12.666666666666666	0.001279239766081...
[323]	[618]	0.09523809523809523	7.444897959183673	0.001096491228070...
[323]	[607]	0.09523809523809523	4.825396825396825	0.001096491228070...
[323]	[834]	0.09523809523809523	5.6036866359447	0.001096491228070...
[323]	[609]	0.12698412698412698	6.812324929971989	0.001461988304093...
[323]	[610]	0.14285714285714285	12.214285714285714	0.001644736842105263
[323]	[603]	0.19047619047619047	6.204081632653061	0.002192982456140...
[714]	[708]	0.39285714285714285	69.34562211981567	0.002010233918128655

only showing top 20 rows

ProductName	prediction
[138]	[141, 142]
[696]	[]
[721]	[]
[211]	[]
[953]	[]
[720, 793, 418, 47]	[486, 489, 100]
[721, 83, 436, 24...	[100, 253, 25, 23]
[363]	[]
[648, 643, 644]	[]
[614]	[272, 834, 1079, ...
[609, 603]	[619, 607, 670, 9...
[397]	[]
[367]	[]
[130]	[196]
[687, 45, 86, 87]	[688, 966]
[787]	[]
[819, 59, 39, 607]	[619, 617, 603, 2...
[746]	[748, 747]
[586, 52]	[]
[743]	[]

only showing top 20 rows

Figure 5: Model with Cluster 1

antecedent	consequent	confidence	lift	support
[1713, 1710]	[1715]	0.3684210526315789	22.854523950325248	0.00126788625249049
[1713, 1710]	[1711]	0.34210526315789475	23.758027143330025	0.0011773229487411
[1713, 1710]	[1712]	0.42105263157894735	10.218160786581839	0.001449012859989...
[2496]	[2501]	0.2777777777777778	19.917027417027416	0.0036225321499728
[2496]	[2498]	0.18055555555555555	14.139676910953506	0.0023546458974823
[2496]	[2499]	0.14583333333333334	14.507132132132133	0.001901829378735...
[1878, 1884]	[1879]	0.4666666666666667	20.36732542819499	0.00126788625249049
[1716]	[1719]	0.22077922077922077	15.236525974025973	0.0015395761637384
[1716]	[1703]	0.19480519480519481	9.777449822904368	0.001358449556239...
[1716]	[1722]	0.19480519480519481	10.49287298067786	0.001358449556239...
[1716]	[1711]	0.24675324675324675	17.13615943804623	0.001720702771237...
[1716]	[1710]	0.16883116883116883	8.24882197448569	0.0011773229487411
[1716]	[1712]	0.23376623376623376	5.673069787355502	0.0016301394674877
[1700, 1720]	[1712]	0.45161290322580644	10.959801488833747	0.00126788625249049
[2746]	[2752]	0.3137254901960784	44.412267471091	0.001449012859989...
[2759]	[3264]	0.26666666666666666	16.178754578754578	0.001086759644991...
[1703, 1880]	[1877]	0.35294117647058826	14.380724983720427	0.001086759644991...
[1703, 1880]	[1888]	0.38235294117647056	15.753511852502193	0.0011773229487411
[2687]	[2608]	0.07647058823529412	4.444148606811146	0.0011773229487411
[2687]	[1712]	0.08235294117647059	1.9985520361990952	0.00126788625249049

only showing top 20 rows

ProductName	prediction
[1700, 3110, 1711]	[1714, 1713, 1709...
[1720]	[1889, 1714, 1700...
[2066]	[]
[1706, 3403, 1679]	[2843, 2501, 2233...
[1532, 1302, 1612]	[1679, 3006, 3304...
[3517]	[3548, 3672]
[2272]	[]
[2732]	[]
[1622]	[]
[3674]	[3678]
[2960]	[]
[2378, 1508]	[1507, 1512, 1510...
[1612]	[1679, 3006, 3304]
[1687]	[]
[1888, 2278, 2279...	[1889, 1885, 1705...
[3170, 2156, 3181...	[3776, 1528, 3408...
[2174]	[]
[2670]	[]
[2169, 3038]	[]
[2792]	[]

only showing top 20 rows

Figure 6: Model with Cluster 2

antecedent	consequent	confidence	lift	support
[3736]	[3733]	0.057971014492753624	1.4785940363151757	0.001802613789995.
[3736]	[3712]	0.057971014492753624	0.9819670317513	0.001802613789995.
[3736]	[3734]	0.15942028985507245	3.609730848861283	0.0049571879224876
[3736]	[3723]	0.08695652173913043	1.3588487446417636	0.002703920684993.
[3736]	[3732]	0.11594202898550725	3.026768968456948	0.0036052275799909
[3736]	[3738]	0.10144927536231885	1.8154511453950446	0.003154574132492.
[3736]	[3709]	0.13043478260869565	1.9167866397926863	0.004055881027489
[3736]	[3719]	0.07246376811594203	2.772363818090955	0.0022532672374943
[3736]	[3722]	0.043478260869565216	1.786634460547504	0.001351960342496.
[3736]	[3726]	0.043478260869565216	1.929565217391304	0.001351960342496.
[3736]	[3735]	0.043478260869565216	1.929565217391304	0.001351960342496.
[3736]	[3720]	0.043478260869565216	2.4738015607580826	0.001351960342496.
[3736]	[3730]	0.08695652173913043	6.431884057971014	0.002703920684993.
[3724]	[3729]	0.07894736842105263	7.299342105263157	0.001351960342496.
[3724]	[3710]	0.07894736842105263	2.189802631578947	0.001351960342496.
[3724]	[3712]	0.13157894736842105	2.228806749698674	0.0022532672374943
[3724]	[3731]	0.10526315789473684	3.3852021357742177	0.001802613789995.
[3724]	[3732]	0.07894736842105263	2.0609907120743034	0.001351960342496.
[3724]	[3738]	0.13157894736842105	2.3546264855687604	0.0022532672374943
[3724]	[3735]	0.15789473684210525	7.007368421052631	0.002703920684993.

only showing top 20 rows

ProductName	prediction
[3517]	[3548, 3405]
[3720]	[3712, 3734, 3723...
[3490]	[]
[3681]	[]
[3672]	[3548]
[3684]	[3683, 3676, 3625]
[3475]	[]
[3390]	[]
[3684]	[3683, 3676, 3625]
[3621]	[]
[3740]	[3739, 3733, 3712...
[3726]	[3734, 3722, 3719...
[3523]	[]
[3632]	[3633]
[3578, 3757]	[]
[3732]	[3739, 3733, 3712...
[3708]	[]
[3425]	[]
[3726, 3405, 3734]	[3712, 3723, 3738...
[3408]	[3409]

only showing top 20 rows

Figure 7: Model with Cluster 3

antecedent	consequent	confidence	lift	support
[2163, 2175]	[2171]	0.6666666666666666	32.45528455284553	0.001002004008016032
[2163, 2175]	[2178]	0.6666666666666666	19.00952380952381	0.00100200400801
[1975, 1904, 1908]	[1979]	0.6666666666666666	42.924731182795696	0.00100200400801
[1975, 1904, 1908]	[2202]	0.6666666666666666	3.7695939565627947	0.00100200400801
[1975, 1904, 1908]	[1973]	0.6666666666666666	60.48484848484848	0.00100200400801
[2175, 2174, 2168]	[2178]	0.4	11.405714285714286	0.00100200400801
[2175, 2174, 2168]	[2176]	0.4	16.633333333333333	0.00100200400801
[2175, 2174, 2168]	[2202]	0.4	2.261756373937677	0.00100200400801
[2175, 2174, 2168]	[2172]	0.6	39.92	0.00150300601202
[2171, 2176, 2178]	[2202]	1.0	5.6543909348441925	0.00100200400801
[2171, 2176, 2178]	[2183]	1.0	37.66037735849057	0.00100200400801
[2046, 2178, 2202]	[2169]	1.0	34.41379310344828	0.00100200400801
[2046, 2178, 2202]	[1940]	1.0	83.16666666666667	0.00100200400801
[1974, 1904, 2203...	[1973]	1.0	90.72727272727272	0.00100200400801
[1973, 1975]	[1979]	0.8333333333333334	53.655913978494624	0.00250501002004
[1973, 1975]	[2179]	0.3333333333333333	7.003508771929824	0.00100200400801
[1973, 1975]	[1904]	0.5	23.209302325581394	0.00150300601202
[1973, 1975]	[1908]	0.3333333333333333	11.088888888888889	0.00100200400801
[1973, 1975]	[2203]	0.3333333333333333	5.591036414565826	0.00100200400801
[1973, 1975]	[2202]	0.3333333333333333	1.8847969782813974	0.00100200400801

only showing top 20 rows

ProductName	prediction
[1891, 1973]	[1974, 1976, 1971...
[2198]	[2200, 2202, 2286...
[2200]	[2202, 2198, 2203...
[2174]	[2182, 2213, 2286...
[2201, 2203, 2198]	[2200, 2202, 2174...
[2202, 1900]	[1906, 2123, 2213...
[2200]	[2202, 2198, 2203...
[2203, 1904, 1908...	[2179, 2198, 2201...
[1904]	[1974, 2198, 2179...
[2172, 1973]	[1974, 1976, 1971...
[2166]	[2179, 1907]
[2202]	[1906, 2123, 2213...
[2157]	[]
[2208]	[]
[2203]	[2198, 2200, 2202...
[2202]	[1906, 2123, 2213...
[1925]	[]
[1900, 1925]	[2198, 1908, 2202]
[2178, 2175]	[2182, 2286, 2198...
[2179, 2166]	[2200, 2203, 2202...

only showing top 20 rows

Figure 8: Model with Cluster 4

As the result, there are many rules are generated, but the support stills small, the reason is that the frequency of each product is quite low compare with the number of datasets. Since the variant of products that causes the probability of re-buy, a specific product is decreased so we have to reduce the minSupport to fit with this situation.

4.3 Advantages of FPGrowth algorithm

-Compare to Apriori, FPGrowth is much faster.
-No need to generate candidate.
-It compresses the dataset to improve the time complexity.

4.4 Disadvantages of FPGrowth algorithm

-It wastes most of the memory and sometimes is not fit and causes ram crashed.
-It is very expensive to build if the dataset is huge.

5 Model Evaluation

Evaluation is an important step that must implement after proposing the model. It will guide in measuring the efficiency and effectiveness of the model. Based on the evaluation result, we can modify some steps or algorithms to improve the accuracy.

In this part, after using 90% of data to train, the rest is used to test and measures the accuracy of this prediction model. The evaluation algorithm is the ratio of the number of correct predicted items over the total of correct and wrong predicted items.

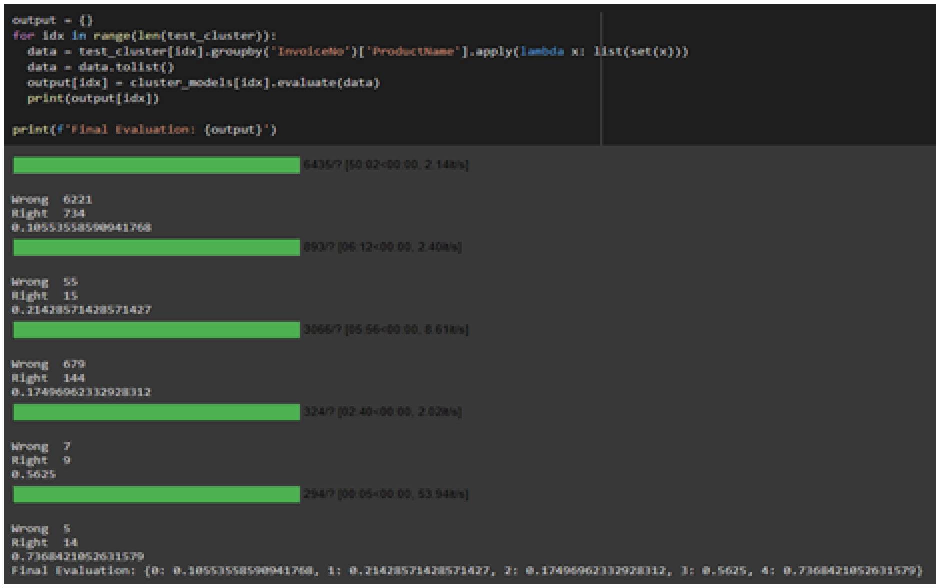
5.1 Evaluation without clustering (12,3%)

```
data = test_data.groupby('InvoiceNo')['ProductName'].apply(lambda x: list(set(x)))
data = data.tolist()
output = model.evaluate(data)
print("Evaluation: ", output)

9820/? [1:25:08<00:00, 1.92it/s]

Wrong  10711
Right  1465
Evaluation:  0.12031865965834429
```

5.2 Evaluation with clustering (35,88%)



[∗] As the results of model evaluation, the model with clustering has better performance than without clustering. The reason is that each cluster has similar data so the association rules have higher precision and when passing the test data, it identifies its cluster that it belongs to then proposes the prediction based on these rules of similarity neighbors.

However, generally, this evaluation of this model is quite low, which means that this model has not reached the threshold to be accepted. Through that, there are stills many problems from preprocessing data steps to the training model steps. Possible problems may come from too density data which have close clusters, parameters are not good enough to cluster or generate association rules, or the limitation of ram that leads to limited training data compare with the amount of kind of product.

6 Features to be updated

-Has more step in preprocessing data, make it cleaner and reduces outliers as much as possible.
-Improve the algorithm of DBSCAN and FPGrowth to reach the best of parameters, decrease time and space complexity. That leads to higher – evaluation percentage achievement.
-Suppose some methods that define which attributes are good to be clustered.

7 Conclusion

- We have the ability to get familiar with version of control management: github and gain better understanding the process of data mining by building a model from scratch.
- Develop skills in processing and transforming datasets.
- Develop skills in implementing machine learning algorithms.
- Develop skills in evaluating the data mining methods.
- Develop communication skills and growing creative thinking and logical thinking in imagination approach.
- We also have the chance to train our teamwork skills, making each of us an effective person with a work-ready attitude for the industry.

8 References

https://towardsdatascience.com/data-preprocessing-concepts-fa946d11c825
https://towardsdatascience.com/how-dbscan-works-and-why-should-i-use-it-443b4a191c80
https://spark.apache.org/docs/latest/ml-frequent-pattern-mining.html
https://towardsdatascience.com/fp-growth-frequent-pattern-generation-in-data-mining-with-python-impleme