

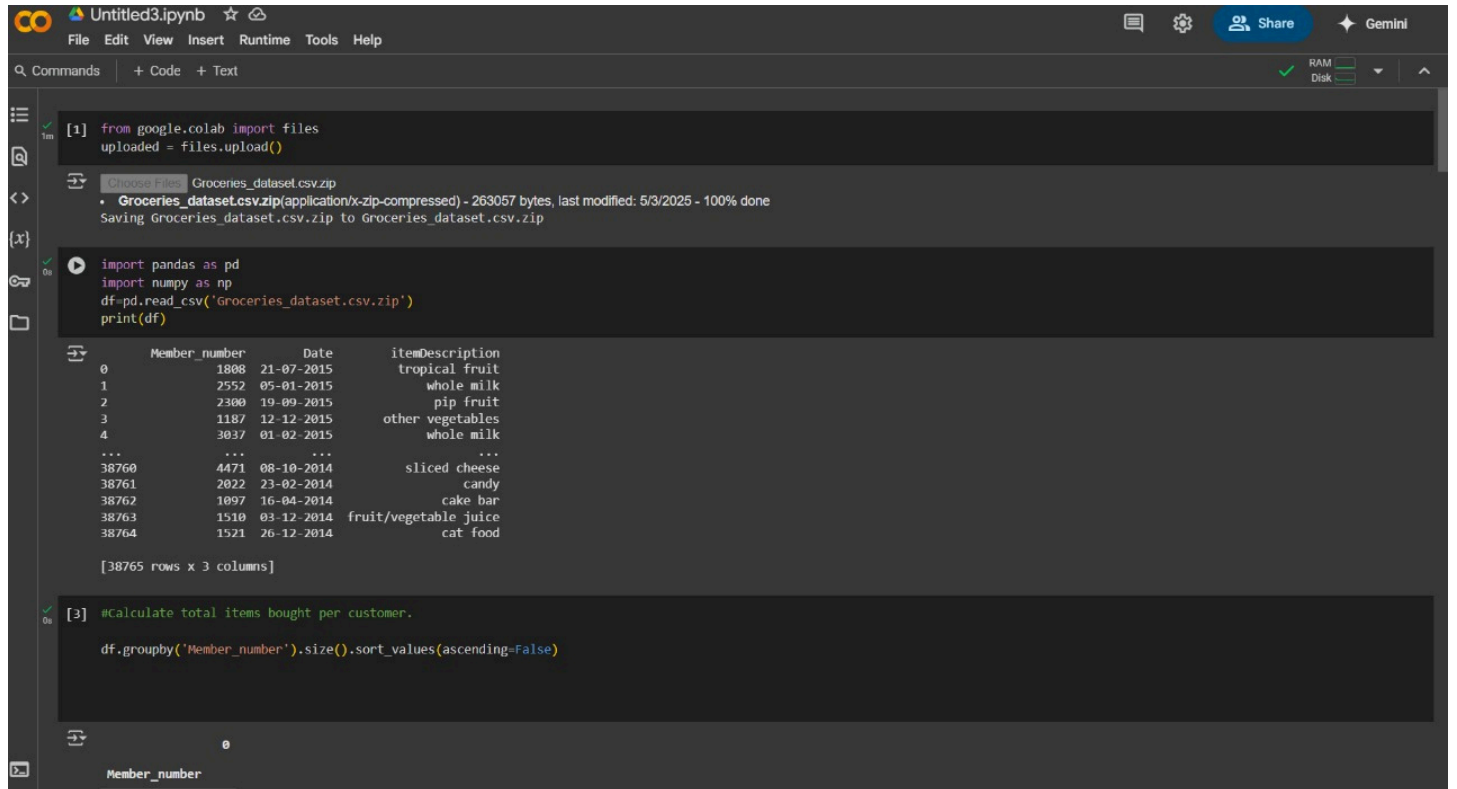
Name : Vaishnavi Mugale

Roll No: 65

class: CS5

PRN : 202401100030

Google collab :  Google Colab



The screenshot shows a Google Colab notebook titled 'Untitled3.ipynb'. The first cell contains code to upload a file from Google Drive. The second cell shows the file 'Groceries_dataset.csv.zip' being uploaded. The third cell contains code to load the dataset using pandas and print its structure. The output shows a DataFrame with 38765 rows and 3 columns: Member_number, Date, and itemDescription. The fourth cell contains code to calculate the total items bought per customer, and the output shows a single value '0' for the 'Member_number' column.

```
[1] from google.colab import files
uploaded = files.upload()

Choose Files Groceries_dataset.csv.zip
• Groceries_dataset.csv.zip(application/x-zip-compressed) - 263057 bytes, last modified: 5/3/2025 - 100% done
Saving Groceries_dataset.csv.zip to Groceries_dataset.csv.zip

import pandas as pd
import numpy as np
df=pd.read_csv('Groceries_dataset.csv.zip')
print(df)

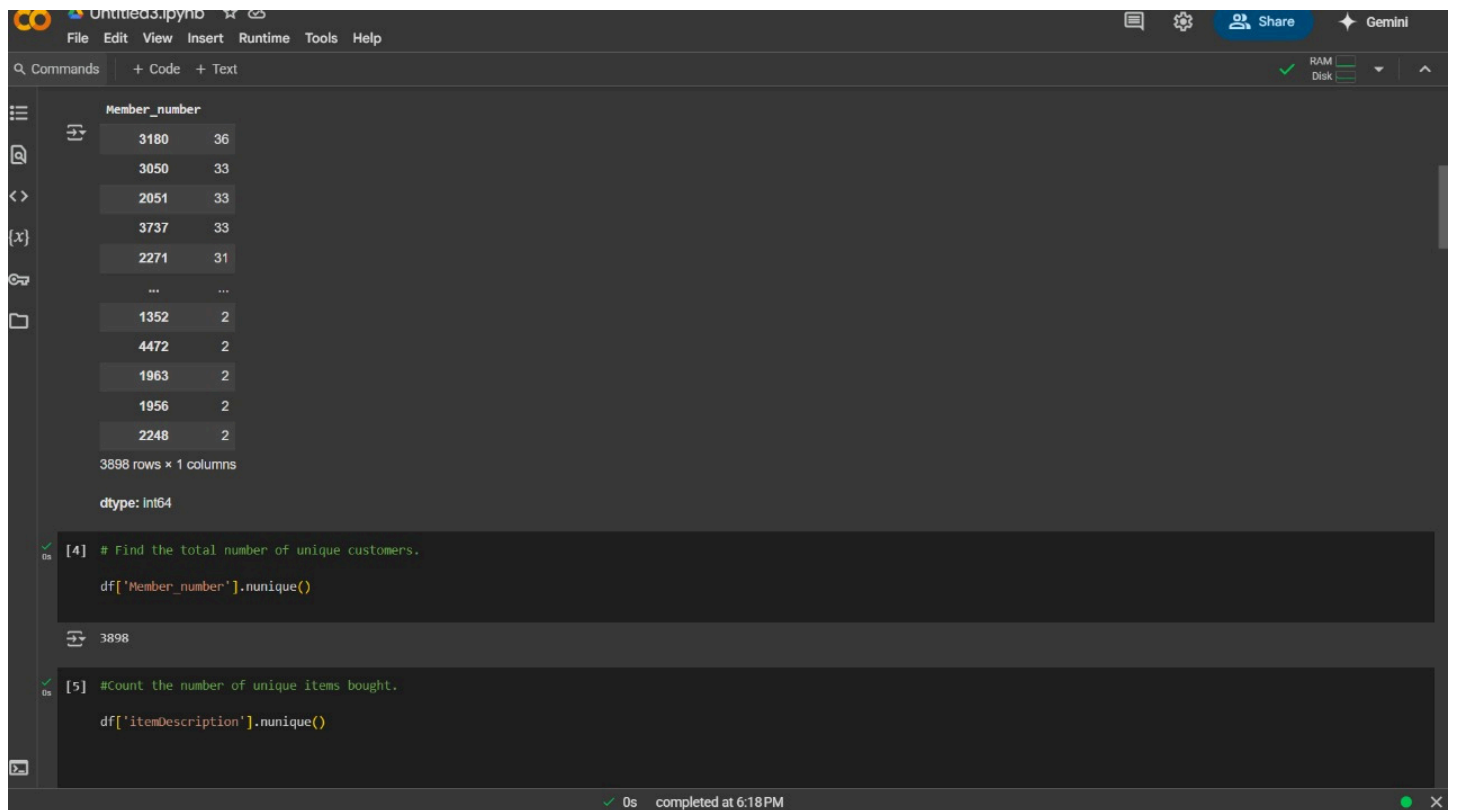
Member_number    Date    itemDescription
0         1888  21-07-2015    tropical fruit
1         2552   05-01-2015    whole milk
2         2300  19-09-2015      pip fruit
3         1187  12-12-2015    other vegetables
4         3037   01-02-2015    whole milk
...         ...         ...
38760      4471  08-10-2014    sliced cheese
38761      2022  23-02-2014      candy
38762      1097  16-04-2014    cake bar
38763      1510  03-12-2014  fruit/vegetable juice
38764      1521  26-12-2014    cat food

[38765 rows x 3 columns]

[3] #Calculate total items bought per customer.

df.groupby('Member_number').size().sort_values(ascending=False)

0
Member_number
```



The screenshot shows the continuation of the Google Colab notebook. The fifth cell contains code to find the total number of unique customers, and the output shows 3898 unique customers. The sixth cell contains code to count the number of unique items bought, and the output shows 3898 unique items.

```
Member_number
3180      36
3050      33
2051      33
3737      33
2271      31
...       ...
1352       2
4472       2
1963       2
1956       2
2248       2

3898 rows x 1 columns

dtype: int64

[4] # Find the total number of unique customers.

df['Member_number'].nunique()

3898

[5] #Count the number of unique items bought.

df['itemDescription'].nunique()
```

0s completed at 6:18 PM

```
Untitled3.ipynb ☆ ☁
File Edit View Insert Runtime Tools Help
Commands + Code + Text
RAM 100% Disk 100%

[6] #Identify the most purchased item.
df['itemDescription'].value_counts().idxmax()

'whole milk'

[7] # Get the top 10 most frequently bought items.
df['itemDescription'].value_counts().head(10)

itemDescription      count
whole milk           2502
other vegetables     1898
rolls/buns           1716
soda                 1514
yogurt               1334
root vegetables      1071
tropical fruit        1032
bottled water         933
sausage              924
citrus fruit          812

dtype: int64

[8] #Find which day had the highest number of transactions.
df['Date'].value_counts().idxmax()
```

```
Untitled3.ipynb ☆ ☁
File Edit View Insert Runtime Tools Help
Commands + Code + Text
RAM 100% Disk 100%

dtype: int64

[8] #Find which day had the highest number of transactions.
df['Date'].value_counts().idxmax()

'21-01-2015'

[11] # Determine the average number of items per transaction per customer.
df.groupby('Member_number').size().mean()

np.float64(9.944843509492047)

[12] #Identify customers who bought more than 100 items.
df.groupby('Member_number').size()[lambda x: x > 100]

0
Member_number
dtype: int64

[13] #find all items bought by a specific customer (e.g., Member_number 1808).
df[df['Member_number'] == 1808]['itemDescription'].unique()
```

```
array(['tropical fruit', 'long life bakery product', 'meat', 'sugar',  
      'rolls/buns', 'semi-finished bread', 'whole milk', 'citrus fruit',  
      'candy', 'napkins'], dtype=object)
```

```
[16] #Find which item is bought by the highest number of unique customers.  
  
df.groupby('itemDescription')['Member_number'].nunique().idxmax()  
  
'whole milk'
```

```
[17] #Find items that were bought only once.  
  
df['itemDescription'].value_counts()[lambda x: x == 1]  
  
      count  
itemDescription  
kitchen utensil      1  
preservation products 1  
  
dtype: int64
```

```
[18] #Calculate number of transactions per day.  
  
df.groupby('Date').size()
```

```
np.float64(8.918932786044126)
```

```
[20] #List customers who only bought one type of item.  
  
df.groupby('Member_number')['itemDescription'].nunique()[lambda x: x == 1]  
  
      itemDescription  
Member_number  
2640      1  
2717      1  
4029      1  
4151      1  
4565      1  
4816      1  
  
dtype: int64
```

```
[22] #What is the least frequently bought item?  
  
df['itemDescription'].value_counts().idxmin()  
  
'kitchen utensil'
```

```
[24] #How many items were bought in the first quarter of 2015?  
  
df[(df['Date'] >= '2015-01-01') & (df['Date'] < '2015-04-01')].shape[0]
```

Untitled3.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text

RAM Disk

[28]

(df['itemDescription'].value_counts(normalize=True)['whole milk'] * 100).round(2)

np.float64(6.45)

[29]

#Which customer bought the highest variety of items?
df.groupby('Member_number')['itemDescription'].nunique().idxmax()

np.int64(1379)

[30]

#How many customers bought 'yogurt' at least once?
df[df['itemDescription'] == 'yogurt']['Member_number'].nunique()

1103

[31]

#Calculate the median number of purchases per customer.
df.groupby('Member_number').size().median()

9.0