



COFFEE CHAIN ANALYSIS

CIS5270 Business Intelligence

Pratiksha Yadav
Sushmitha Dandu

Contents

A. Introduction	2
B. Data set Description	4
C. Data Cleaning	6
D. Summary Statistics	16
E. Analysis & Visualizations	27
F. References	54

A. Introduction

The first coffee chains were established in the early 1900s. These early chains were small and independent, and they typically served only coffee and doughnuts. In the 1960s, the coffee chain industry began to grow rapidly. This growth was driven by several factors, including the increasing popularity of coffee, the growing availability of coffee beans, and the development of new coffee-making technologies. In the 1980s, the coffee chain industry experienced another period of rapid growth. This growth was driven by the introduction of new coffee drinks, such as the latte and the cappuccino. The coffee chain industry continued to grow in the 1990s and 2000s, and today there are thousands of coffee chains operating around the world.^[1]

The coffee chain industry is highly competitive. The top five coffee chains in the world are Starbucks, Dunkin' Donuts, McDonald's, Tim Hortons, and Costa Coffee^[2]. These chains account for a significant share of the global coffee market. The global coffee industry has a total addressable market of more than \$400 billion (annual sales). Of that, about \$8 in every \$10 is spent in Coffee Away from Home (coffee shops, vending machines, etc.) The coffee industry is a large and rapidly expanding market with intense competition. Consequently, there is an abundance of data that can be utilized to gain insights into the industry's business operations ^[3]. Coffee is one of the most consumed beverages around the world. The Coffee Chains Dataset is a comprehensive dataset that provides detailed information on various aspects of the coffee chains in the United States with the business operations, including market size, product lines, marketing strategies, and profitability. Our project's primary focus is on creating meaningful visualizations using Python libraries like Matplotlib and Seaborn to understand the patterns and trends within the dataset. The objective of analyzing the coffee chain data using Python is to extract valuable insights and make informed business decisions. By analyzing the data, businesses can identify trends, understand customer

preferences, and optimize their operations to improve profitability. These visualizations will help identify the best-selling products, sales of products in different parts of the US and identify the most profitable markets that would help the business understand the seasonal demand for coffee. Overall, this project motivates us to apply our theoretical knowledge to a real-world scenario and gain valuable insights that could enhance the coffee chains' business operations. to make them manage their inventory and give them the ability to make informed decisions for the profitability of the business.

B. Data set Description

The dataset for this project has been taken from the Kaggle website and below is the link:

Dataset URL:

<https://www.kaggle.com/datasets/arjunbhaybhang/coffee-chains-dataset>

Total Rows: 4248

Total Columns: 20

Data Format: CSV

Number of Files: 1

Below is the overview of the dataset with the columns and first few rows.

	Area				Product	Product				Budget	Budget	Budget	Budget								Total	
1	Code	Date	Market	Market Size	Product	Line	Type	State	Type	COGS	Margin	Profit	Sales	COGS	Inventory	Margin	Marketing	Profit	Sales	Expenses		
2	719	40909	Central	Major Marke	Amaretto	Beans	Coffee	Colorad	Regular	90	130	100	220	89	777	130	24	94	219	36		
3	970	40909	Central	Major Marke	Colombian	Beans	Coffee	Colorad	Regular	80	110	80	190	83	623	107	27	68	190	39		
4	970	40909	Central	Major Marke	Decaf Irish C	Beans	Coffee	Colorad	Decaf	100	140	110	240	95	821	139	26	101	234	38		
5	303	40909	Central	Major Marke	Green Tea	Leaves	Tea	Colorad	Regular	30	50	30	80	44	623	56	14	30	100	26		
6	303	40909	Central	Major Marke	Caffe Moch	Beans	Espresso	Colorad	Regular	60	90	70	150	54	456	80	15	54	134	26		
7	720	40909	Central	Major Marke	Decaf Espre	Beans	Espresso	Colorad	Decaf	80	130	80	210	72	558	108	23	53	180	55		
8	970	40909	Central	Major Marke	Chamomile	Leaves	Herbal Te	Colorad	Decaf	140	160	110	300	170	1091	171	47	99	341	72		
9	719	40909	Central	Major Marke	Lemon	Leaves	Herbal Te	Colorad	Decaf	50	80	20	130	63	435	87	57	0	150	87		
10	970	40909	Central	Major Marke	Mint	Leaves	Herbal Te	Colorad	Decaf	50	70	40	120	60	336	80	19	33	140	47		

Data Description:

Data is an important source of information in any kind of project as it allows the analyst to build relationships between what is happening in different measures and categories.

Field Name	Description	Example
Area Code	Code that indicates the geographical area of Coffee chain operations	970, 710
Date	The specific date or time period in which the data pertains	40909
Market	A specific market or target audience for a product or service.	Central, East, West, South

Market Size	The size of the market being analyzed, typically measured in terms of revenue or sales	Major Market and Small Market
Product	A specific item or group of items that a company sells to customers.	Decaf Espresso, Chamomile, Green Tea
Product Line	A group of related products sold by a company	Beans, Leaves
Product Type	Different types of coffee products are being used.	Coffee, Tea, Espresso, Herbal Tea
State	The state or province in which a company operates, or a customer is located.	Colorado, Illinois, New Hampshire
Type	The type of product being preferred.	Regular and Decaf
Budget COGS	The budgeted cost of goods sold (COGS) for a particular product or service.	10, 160, 30
Budget Margin	The budgeted profit margin for a particular product or service.	130, 70, 210
Budget Profit	The budgeted profit for a particular product or service.	100, 110, 20
Budget Sales	The budgeted sales revenue for a particular product or service.	220, 190, 240
COGS	The actual cost of goods sold for a particular product or service	89, 83, 95
Inventory	The value of the inventory held by the company.	777, 623, 821
Marketing	The amount spent on marketing activities for a particular product or service.	24, 27, 16
Margin	The profit earned by a company on each unit sold or each transaction made.	103, 107, 139
Profit	The actual profit earned from the sale of a particular product or service.	94, 68, 101
Sales	The actual sales revenue is generated from a particular product or service.	219, 190, 234
Total Expenses	Sum of all the costs incurred	36, 39, 38

C. Data Cleaning

Before starting with the data cleaning, we first read the data from our CSV files into a data frame and print it. The code below is used to run in Spyder IDE.

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Coffee.csv')
print(coffee)

# View the top 10 values
print(coffee.head(10))
```

Code text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Coffee.csv')
print(coffee)

# View the top 10 values
print(coffee.head(10))
```

```
In [51]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/DataCleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work')
Area Code Date Market ... Profit Sales Total Expenses
0 719 40909 Central ... 94.0 219 36
1 970 40909 Central ... 68.0 190 39
2 970 40909 Central ... 101.0 234 38
3 303 40909 Central ... 30.0 100 26
4 303 40909 Central ... 54.0 134 26
... ..
4243 206 41609 West ... 19.0 60 19
4244 509 41609 West ... 34.0 155 57
4245 360 41609 West ... 76.0 188 45
4246 360 41609 West ... 86.0 188 46
4247 206 41609 West ... 30.0 266 125

[4248 rows x 20 columns]
```

We printed the top 10 rows of our data set and the result looked as follows:

```
In [4]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work')

   Area Code  Date  Market  ... Profit Sales Total Expenses
0      719  40909  Central  ...   94.0   219           36
1      970  40909  Central  ...   68.0   190           39
2      970  40909  Central  ...  101.0   234           38
3      303  40909  Central  ...   30.0   100           26
4      303  40909  Central  ...   54.0   134           26
5      720  40909  Central  ...   53.0   180           55
6      970  40909  Central  ...   99.0   341           72
7      719  40909  Central  ...    NaN   150           87
8      970  40909  Central  ...   33.0   140           47
9      719  40909  Central  ...   17.0   130           55

[10 rows x 20 columns]
```

To view the information on data frame, we used the pandas data frame info() method as follows:

```
#View the information of the dataframe, col name, data type, total entries etc.
print(coffee.info())
```

```
#View the information of the dataframe, col name, data type, total entries etc.

print(coffee.info())
```

Below screenshot displays the data frame information:

```
In [3]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4248 entries, 0 to 4247
Data columns (total 20 columns):
#   Column              Non-Null Count  Dtype  
---  -
0   Area Code           4248 non-null  int64  
1   Date                4248 non-null  int64  
2   Market              4248 non-null  object  
3   Market Size         4248 non-null  object  
4   Product             4248 non-null  object  
5   Product Line        4248 non-null  object  
6   Product Type        4248 non-null  object  
7   State               4248 non-null  object  
8   Type                4248 non-null  object  
9   Budget COGS         4248 non-null  int64  
10  Budget Margin       4248 non-null  int64  
11  Budget Profit       4248 non-null  int64  
12  Budget Sales        4248 non-null  int64  
13  COGS                4248 non-null  int64  
14  Inventory           4248 non-null  int64  
15  Margin              4248 non-null  int64  
16  Marketing           4248 non-null  int64  
17  Profit              4246 non-null  float64 
18  Sales               4248 non-null  int64  
19  Total Expenses      4248 non-null  int64  
dtypes: float64(1), int64(12), object(7)
memory usage: 663.9+ KB
```

For data cleaning purposes, we used different techniques based on our scenario which are explained in-depth as follows:

1. Checking the Null values

To address the null values in our dataset we are using the `isnull.sum()` function to check the null values in our dataset.

Before Cleaning:

```
#Checking the Null values
print(coffee.isnull().sum())

print('\n Profit has null rows:' + str(coffee['Profit'].isnull().sum()))
```

Code text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Coffee.csv')

#Checking the Null values
print(coffee.isnull().sum())

print('\n Profit has null rows:' + str(coffee['Profit'].isnull().sum()))
```

Using the above code we checked that the Profit column has 2 Null Values in it. We will be removing those Null values.

```
In [6]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_work/New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_work')
Area Code      0
Date           0
Market         0
Market Size    0
Product        0
Product Line   0
Product Type   0
State          0
Type           0
Budget COGS    0
Budget Margin  0
Budget Profit  0
Budget Sales   0
COGS           0
Inventory      0
Margin         0
Marketing      0
Profit         2
Sales          0
Total Expenses 0
dtype: int64

Profit has null rows:2
```

After Cleaning:

Removing the NAN values from 'Profit' column using below code.

```
# drop rows with NaN values in the Profit column
coffee.dropna(subset=['Profit'], inplace=True)
```

Code text:

```
#Drop NAN values from the Profit column
coffee.dropna(subset=['Profit'], inplace=True)
print(coffee.isnull().sum())
print("\n Profit has null rows:" + str(coffee['Profit'].isnull().sum()))
```

After implementing the above code, we can see that the 'Null' values have been removed from Profit column.

```
In [16]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work')
Area Code      0
Date           0
Market         0
Market Size    0
Product        0
Product Line   0
Product Type   0
State          0
Type           0
Budget COGS    0
Budget Margin  0
Budget Profit  0
Budget Sales   0
COGS           0
Inventory      0
Margin         0
Marketing      0
Profit         0
Sales         0
Total Expenses 0
dtype: int64

Profit has null rows:0
```

2. Standardize the Data:

Inaccurate or inconsistent date formats can lead to errors in data analysis. Standardizing the date format ensures that the data is accurate and reliable for insightful analysis.

Before cleaning:

We can see from the below screenshot that the Date column is not in the date format. We will be converting it to date format so we can analyze the data appropriately.

```
In [2]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/
New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work')
Area Code Date Market ... Profit Sales Total Expenses
0 719 40909 Central ... 94.0 219 36
1 970 40909 Central ... 68.0 190 39
2 970 40909 Central ... 101.0 234 38
3 303 40909 Central ... 30.0 100 26
4 303 40909 Central ... 54.0 134 26
... ..
4243 206 41609 West ... 19.0 60 19
4244 509 41609 West ... 34.0 155 57
4245 360 41609 West ... 76.0 188 45
4246 360 41609 West ... 86.0 188 46
4247 206 41609 West ... 30.0 266 125
```

We will be using the below code to convert date integers to formatted strings

Code text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns
print(coffee)

#Read the CSV file into the dataframe
coffee = pd.read_csv('Coffee.csv')

# define lambda function to convert date integers to formatted strings

to_date_string = lambda x: datetime.fromordinal(datetime(1900, 1, 1).toordinal() + x
- 2).strftime('%m/%d/%Y')

#apply lambda function to 'Date' column with formatted date strings
coffee['Date'] = coffee['Date'].apply(to_date_string)
print(coffee)
```

Screenshot of Code:

```
# define lambda function to convert date integers to formatted strings
to_date_string = lambda x: datetime.fromordinal(datetime(1900, 1, 1).toordinal() + x - 2).strftime('%m/%d/%Y')

#apply lambda function to 'Date' column with formatted date strings
coffee['Date'] = coffee['Date'].apply(to_date_string)

print(coffee)
```

After Cleaning:

We can see the Date column is now changed to the appropriate format.

```
In [18]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/
New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work')

```

	Area	Code	Date	Market	...	Profit	Sales	Total	Expenses
0		719	01/01/2012	Central	...	94.0	219		36
1		970	01/01/2012	Central	...	68.0	190		39
2		970	01/01/2012	Central	...	101.0	234		38
3		303	01/01/2012	Central	...	30.0	100		26
4		303	01/01/2012	Central	...	54.0	134		26
...
4243		206	12/01/2013	West	...	19.0	60		19
4244		509	12/01/2013	West	...	34.0	155		57
4245		360	12/01/2013	West	...	76.0	188		45
4246		360	12/01/2013	West	...	86.0	188		46
4247		206	12/01/2013	West	...	30.0	266		125

[4248 rows x 20 columns]

3. Splitting the Date column into Month, Date, and Year components:

We split the date column which is in format mm/dd/year to Month, Date, and Year columns

to make it easier to group and analyze the data based on these different time components.

Before Cleaning:

```
In [18]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/
Project_Work/New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS
5270-BI/Python_Project/Project_Work')

```

	Area	Code	Date	Market	...	Profit	Sales	Total	Expenses
0		719	01/01/2012	Central	...	94.0	219		36
1		970	01/01/2012	Central	...	68.0	190		39
2		970	01/01/2012	Central	...	101.0	234		38
3		303	01/01/2012	Central	...	30.0	100		26
4		303	01/01/2012	Central	...	54.0	134		26
...
4243		206	12/01/2013	West	...	19.0	60		19
4244		509	12/01/2013	West	...	34.0	155		57
4245		360	12/01/2013	West	...	76.0	188		45
4246		360	12/01/2013	West	...	86.0	188		46
4247		206	12/01/2013	West	...	30.0	266		125

```
print(coffee)

# split date column into month, date, and year columns
coffee[['month', 'date', 'year']] = coffee['Date'].str.split('/', expand=True)

# print the modified dataframe
print(coffee)
```

Code text:

```
# split date column into month, date, and year columns
coffee[['month', 'date', 'year']] = coffee['Date'].str.split('/', expand=True)

# print the modified dataframe
print(coffee)
```

After Cleaning:

```
In [20]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/
Project_Work/New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS
5270-BI/Python_Project/Project_Work')
   Area Code      Date Market ... month date year
0      719 01/01/2012  Central ...   01   01  2012
1      970 01/01/2012  Central ...   01   01  2012
2      970 01/01/2012  Central ...   01   01  2012
3      303 01/01/2012  Central ...   01   01  2012
4      303 01/01/2012  Central ...   01   01  2012
...      ...      ...      ... ...   ...   ...   ...
4243    206 12/01/2013    West ...   12   01  2013
4244    509 12/01/2013    West ...   12   01  2013
4245    360 12/01/2013    West ...   12   01  2013
4246    360 12/01/2013    West ...   12   01  2013
4247    206 12/01/2013    West ...   12   01  2013

[4248 rows x 23 columns]
```

4. Converting to title Case:

Before Cleaning:

Here in the below screenshot, we can see that the Type column is not in the title case so we are converting it to the title case to keep the consistency for the visuals.

```
In [21]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/
Project_Work/New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS
5270-BI/Python_Project/Project_Work')
0      regular
1      regular
2      decaf
3      regular
4      regular
...
4243    regular
4244    regular
4245     decaf
4246    regular
4247     decaf
Name: Type, Length: 4248, dtype: object
```

Code Screenshot:

```
Created on Fri May 12 09:24:13 2023

@author: pyadav
"""

import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Coffee.csv')
#print(coffee)

#Before cleaning check
print(coffee['Type'])

#Convert the 'Type' column to title case using the apply() method
coffee['Type'] = coffee['Type'].apply(lambda x: x.title())

# Print the resulting DataFrame
print(coffee['Type'])
```

We will be using the below code

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Coffee.csv')

#Before cleaning check
print(coffee['Type'])

#Convert the 'Type' column to title case using the apply() method
coffee['Type'] = coffee['Type'].apply(lambda x: x.title())

# Print the resulting DataFrame
print(coffee['Type'])
```

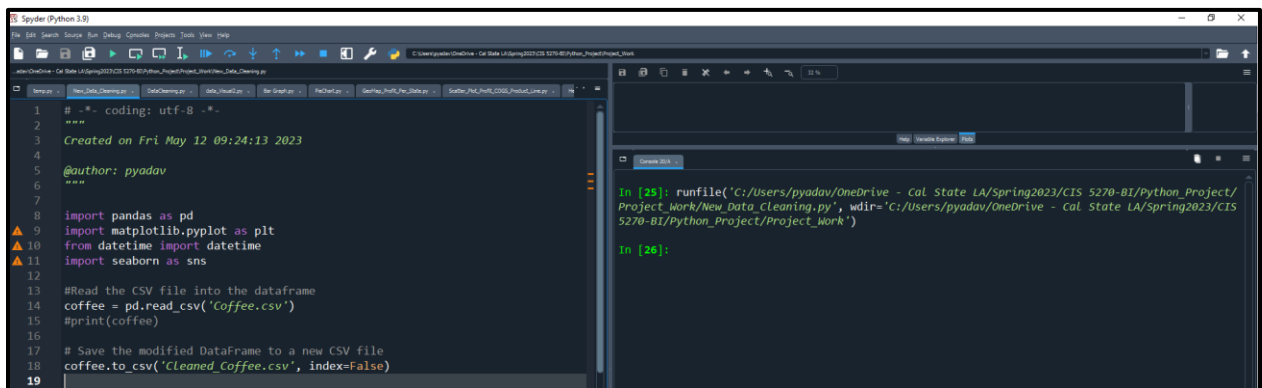
After Cleaning:

```
In [23]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/New_Data_Cleaning.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work')
0      Regular
1      Regular
2      Decaf
3      Regular
4      Regular
...
4243   Regular
4244   Regular
4245   Decaf
4246   Regular
4247   Decaf
Name: Type, Length: 4248, dtype: object
```

Saving the modified data frame to the new file:

After performing all the data cleaning, we have saved the modified file and downloaded it using the below code. We will be using the 'Cleaned_Coffee.csv' file for performing summary statistics and analysis.

Full Screenshot:



Code Screenshot:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Coffee.csv')
#print(coffee)

# Save the modified DataFrame to a new CSV file
coffee.to_csv('Cleaned_Coffee.csv', index=False)
```

Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Coffee.csv')
#print(coffee)

# Save the modified DataFrame to a new CSV file
coffee.to_csv('Cleaned_Coffee.csv', index=False)
```


D. Summary Statistics

1. Summary Statistics for the Profit Column

The below code is used to view the summary statistics for the ***“Profit”*** column of our dataset.

The summary statistics can be shown using the describe() or by using the individual functions as shown below in the code snippet:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)
print('\nSummary of the Profit column:\n' + str(coffee['Profit'].describe()))
print('\nPrinting individual statistics for the Profit:')
print('\nMean of the Profit:', "{0:.2f}".format(coffee.Profit.mean()))
print('\nMinimum of the Profit:', "{0:.2f}".format(coffee.Profit.min()))
print('\nMaximum of the Profit:', "{0:.2f}".format(coffee.Profit.max()))
print('\nStandard Deviation of the Profit:', "{0:.2f}".format(coffee.Profit.std()))
coffee['Profit'].plot(kind='box')
plt.show()
```

Code text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns
#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)
print('\nSummary of the Profit column:\n' + str(coffee['Profit'].describe()))
#print(coffee['Profit'].describe())
print('\nPrinting individual statistics for the Profit:')
print('\nMean of the Profit:', "{0:.2f}".format(coffee.Profit.mean()))
print('\nMinimum of the Profit:', "{0:.2f}".format(coffee.Profit.min()))
print('\nMaximum of the Profit:', "{0:.2f}".format(coffee.Profit.max()))
print('\nStandard Deviation of the Profit:', "{0:.2f}".format(coffee.Profit.std()))
coffee['Profit'].plot(kind='box')
plt.show()
```

The output of the above code is as follows:

```
In [87]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/
Profit_SummaryStatistics.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/
Project_Work')

Summary of the Profit column:
count    4246.000000
mean      61.384126
std      100.872136
min     -638.000000
25%       17.000000
50%       40.000000
75%       92.000000
max       778.000000
Name: Profit, dtype: float64

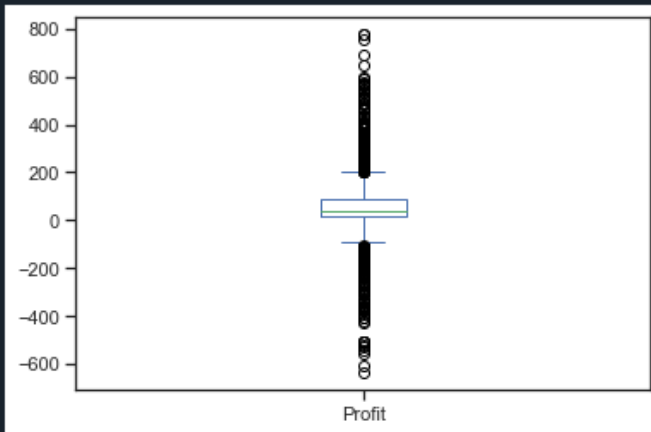
Printing individual statistics for the Profit:

Mean of the Profit: 61.38

Minimum of the Profit: -638.00

Maximum of the Profit: 778.00

Standard Deviation of the Profit: 100.87
```



The summary statistics for the Profit column provide valuable insights into the coffee chain sales dataset. The **count** of 4,246 observations indicates the number of observations available in the Profit column. The **mean** profit value of 61.38 serves as an estimate of the central tendency, representing a typical profit level. The **standard deviation** of 100.87 measures the dispersion or variability of profits around the mean, showing a wider range of profitability for higher values. The **minimum** value of -638.00 identifies outliers or extreme negative profits that could impact overall profitability. The **25th and 75th percentiles** of 17.00 and 92.00, respectively,

help us understand the lower and upper ends of the profit distribution. The **median** profit value of 40.00 represents the midpoint, indicating the typical profit level. The **maximum** profit of 778.00 identifies outliers or exceptional positive profits. Collectively, these statistics offer a comprehensive view of profit distribution, identification of typical profit levels, outliers, and the overall spread of profitability. Analysts can leverage this information to assess performance, compare product lines, and make informed business decisions based on the coffee chain's profitability.

2. Summary Statistics for the Sales Column

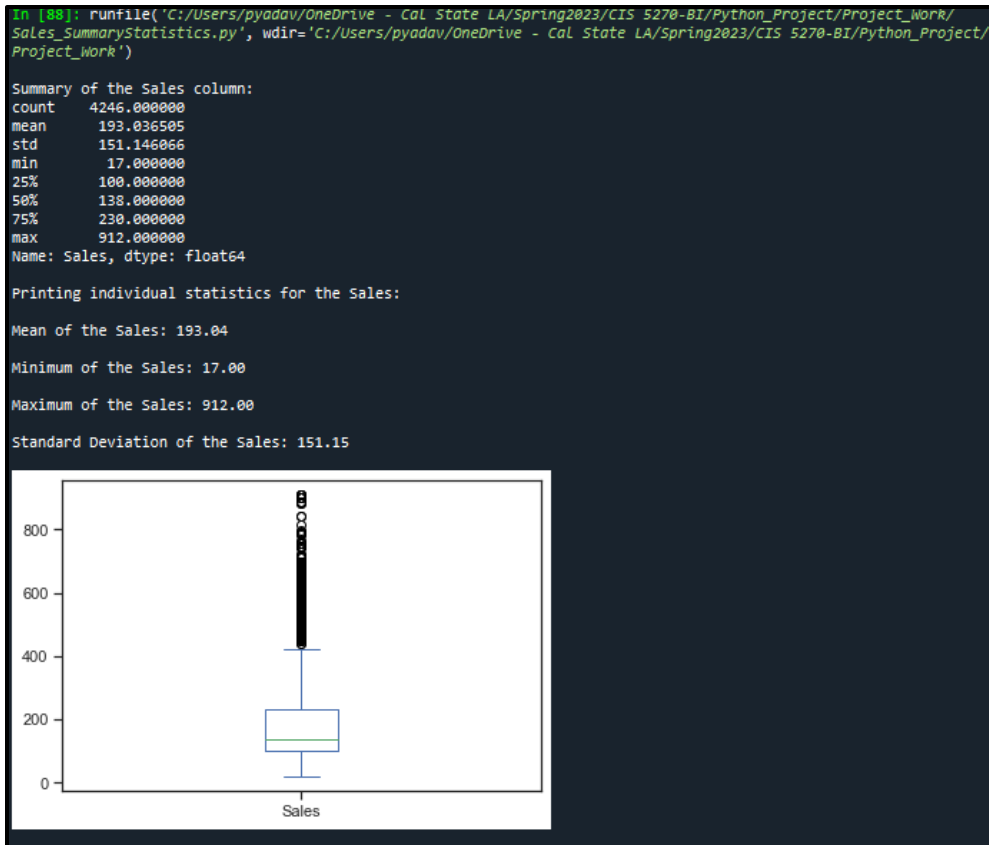
The below code is used to view the summary statistics for the “*Sales*” column of our dataset.

The summary statistics can be shown using the describe() or by using the individual functions as shown below in the code snippet:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
#print(coffee)
print('\nSummary of the Sales column:\n' + str(coffee['Sales'].describe()))
print('\nPrinting individual statistics for the Sales:')
print('\nMean of the Sales:', "{0:.2f}".format(coffee.Sales.mean()))
print('\nMinimum of the Sales:', "{0:.2f}".format(coffee.Sales.min()))
print('\nMaximum of the Sales:', "{0:.2f}".format(coffee.Sales.max()))
print('\nStandard Deviation of the Sales:', "{0:.2f}".format(coffee.Sales.std()))
coffee['Sales'].plot(kind='box')
plt.show()
```

The output of the above code is as follows:



Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
#print(coffee)
print("\nSummary of the Sales column:\n" + str(coffee['Sales'].describe()))
print("\nPrinting individual statistics for the Sales:")
print("\nMean of the Sales:", "{0:.2f}".format(coffee.Sales.mean()))
print("\nMinimum of the Sales:", "{0:.2f}".format(coffee.Sales.min()))
print("\nMaximum of the Sales:", "{0:.2f}".format(coffee.Sales.max()))
print("\nStandard Deviation of the Sales:", "{0:.2f}".format(coffee.Sales.std()))
coffee['Sales'].plot(kind='box')
plt.show()
```

The summary statistics for the Sales column of the Coffee Chain dataset provide valuable insights into the distribution and characteristics of sales. The **minimum** sales value is 17, indicating the lowest recorded sales, while the **maximum** value is 912, representing the highest recorded sales. The **mean** sales value of 193.04 represents the average sales, giving an indication of the typical sales level. The **25th percentile** value of 100.00 signifies that 25% of the sales values fall below or equal to this value, providing insights into the lower end of the sales distribution. Similarly, the **75th percentile** value of 230.00 indicates that 75% of the sales values are below or equal to this value, offering insights into the upper end of the sales distribution. The **median value** of 138.00, which corresponds to the **50th percentile**, represents the middle value of the sales data and provides an estimate of the central tendency. Overall, these statistics reveal that the Coffee Chain sales range from 17 to 912, with an average sales value of 193.04. Additionally, the 25th and 75th percentiles highlight the lower and upper quartiles of the sales distribution, while the median provides the middle value. By leveraging these statistics, businesses can optimize their sales strategies, drive growth, and stay competitive in the market.

3. Summary Statistics for the Margin Column

The below code is used to view the summary statistics for the “Margin” column of our dataset. The summary statistics can be shown using the `describe()` or by using the individual functions as shown below in the code snippet:

```

import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
#print(coffee)
print('\nSummary of the Margin column:\n' + str(coffee['Margin'].describe()))
print('\nPrinting individual statistics for the Margin:')
print('\nMean of the Margin:', "{0:.2f}".format(coffee.Margin.mean()))
print('\nMinimum of the Margin:', "{0:.2f}".format(coffee.Margin.min()))
print('\nMaximum of the Margin:', "{0:.2f}".format(coffee.Margin.max()))
print('\nStandard Deviation of the Margin:', "{0:.2f}".format(coffee.Margin.std()))
coffee['Margin'].plot(kind='box')
plt.show()

```

The output of the above code is as follows:

```

In [93]: runfile('C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/Project_Work/
Margin_SummaryStatistics.py', wdir='C:/Users/pyadav/OneDrive - Cal State LA/Spring2023/CIS 5270-BI/Python_Project/
Project_Work')

```

Summary of the Margin column:

```

count    4246.000000
mean      104.383184
std        94.198534
min     -302.000000
25%        53.000000
50%        76.000000
75%       132.000000
max       613.000000
Name: Margin, dtype: float64

```

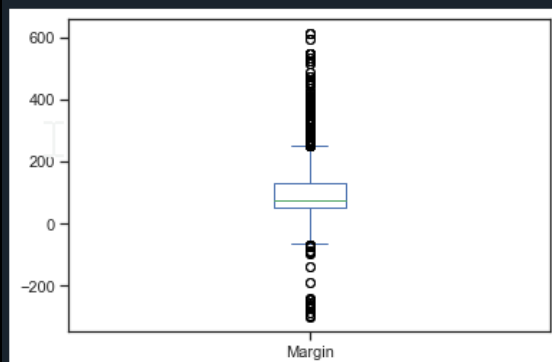
Printing individual statistics for the Margin:

Mean of the Margin: 104.38

Minimum of the Margin: -302.00

Maximum of the Margin: 613.00

Standard Deviation of the Margin: 94.20



Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
#print(coffee)
print("\nSummary of the Margin column:\n" + str(coffee['Margin'].describe()))
print("\nPrinting individual statistics for the Margin:")
print("\nMean of the Margin:", "{0:.2f}".format(coffee.Margin.mean()))
print("\nMinimum of the Margin:", "{0:.2f}".format(coffee.Margin.min()))
print("\nMaximum of the Margin:", "{0:.2f}".format(coffee.Margin.max()))
print("\nStandard Deviation of the Margin:", "{0:.2f}".format(coffee.Margin.std()))
coffee['Margin'].plot(kind='box')
plt.show()
```

The statistical results provide an overview of the "Margin" column in the dataset, which represents the margin values for a coffee chain. The count indicates the number of non-missing values in the "Margin" column. In this case, there are 4,246 valid margin values.

The mean represents the average margin value across all the data points, the mean margin for the coffee chain is approximately 104.38. The standard deviation measures the dispersion or variability of the margin values. A higher standard deviation indicates greater variability in the margins. In this case, the margin values have a standard deviation of approximately 94.20, suggesting a considerable spread in the margin data. The minimum value represents the smallest margin recorded in the dataset. In this case, the minimum margin is -302.00, which indicates a loss on some transactions.

The 25th percentile is the value below which 25% of the margin values fall. It is also known as the first quartile (Q1). In this case, the first quartile is 53.00, meaning that 25% of the margins are below this value. The median represents the middle value of the dataset when it is sorted in ascending order. It is also the 50th percentile or the second quartile (Q2). In this case, the median

margin is 76.00, indicating that 50% of the margins fall below this value. The 75th percentile is the value below which 75% of the margin values fall. It is also known as the third quartile (Q3). In this case, the third quartile is 132.00, meaning that 75% of the margins are below this value. The maximum value represents the largest margin recorded in the dataset. In this case, the maximum margin is 613.00, indicating the highest recorded margin value.

The summary statistics provide insights into the distribution and central tendencies of the margin column, allowing us to understand the average margin, the spread of values, and the range of margins achieved by the coffee chain. Additionally, examining quartiles helps to identify the spread of margins and the proportion of data falling within different percentile ranges.

4. Summary Statistics for the Total Expenses Column

The below code is used to view the summary statistics for the “Total Expenses” column of our dataset. The summary statistics can be shown using the `describe()` or by using the individual functions as shown below in the code snippet:


```

import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

# Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')

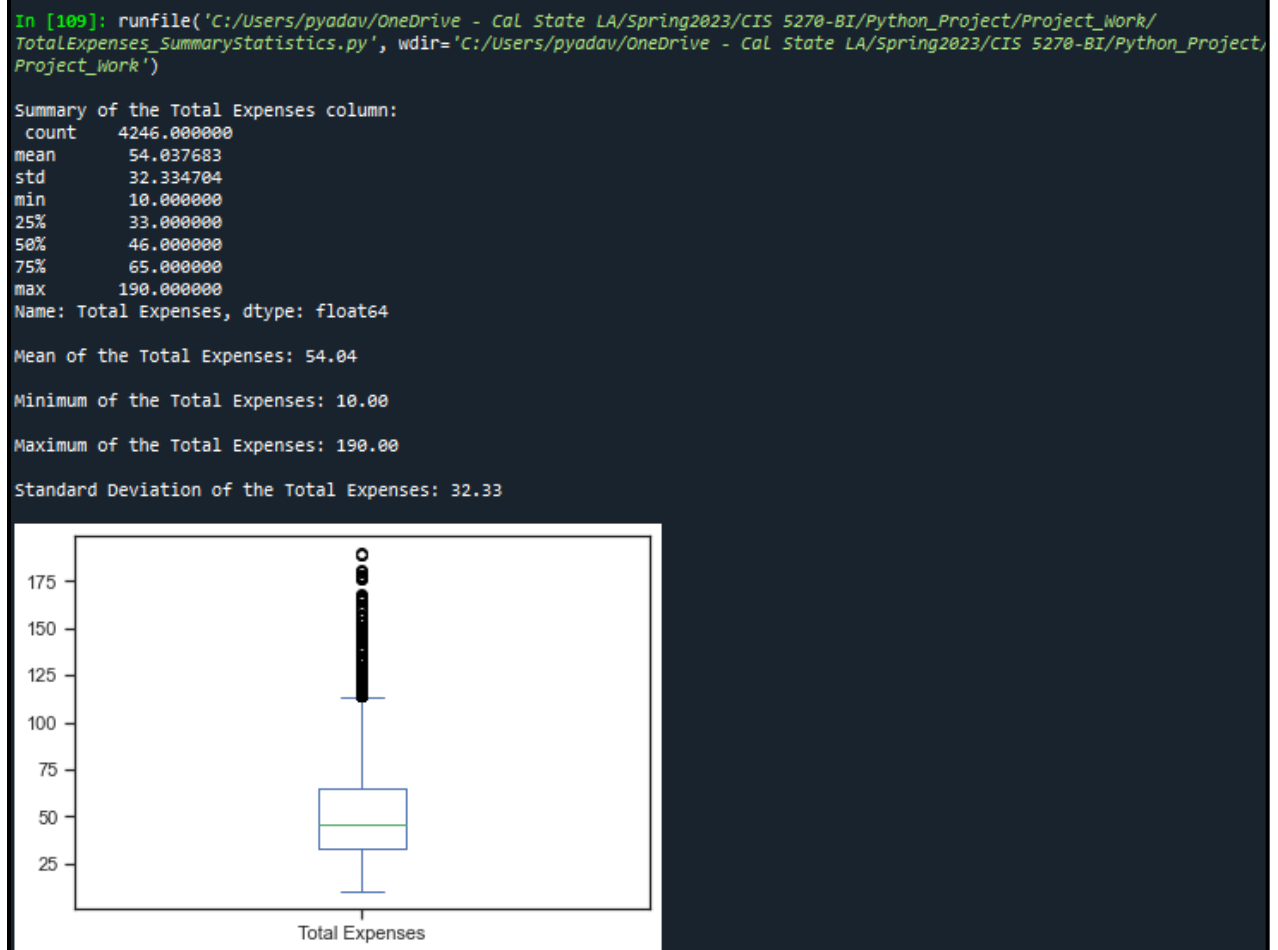
# Print summary of the 'Total Expenses' column
print('\nSummary of the Total Expenses column:\n', coffee['Total Expenses'].describe())

# Print individual statistics for the 'Total Expenses'
print('\nMean of the Total Expenses:', "{0:.2f}".format(coffee['Total Expenses'].mean()))
print('\nMinimum of the Total Expenses:', "{0:.2f}".format(coffee['Total Expenses'].min()))
print('\nMaximum of the Total Expenses:', "{0:.2f}".format(coffee['Total Expenses'].max()))
print('\nStandard Deviation of the Total Expenses:', "{0:.2f}".format(coffee['Total Expenses'].std()))

# Plot a boxplot of the 'Total Expenses' column
coffee['Total Expenses'].plot(kind='box')
plt.show()

```

The output of the above code is as follows:



Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

# Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')

# Print summary of the 'Total Expenses' column
print("\nSummary of the Total Expenses column:\n", coffee['Total Expenses'].describe())

# Print individual statistics for the 'Total Expenses'
print("\nMean of the Total Expenses:", "{0:.2f}".format(coffee['Total Expenses'].mean()))
print("\nMinimum of the Total Expenses:", "{0:.2f}".format(coffee['Total Expenses'].min()))
print("\nMaximum of the Total Expenses:", "{0:.2f}".format(coffee['Total Expenses'].max()))
print("\nStandard Deviation of the Total Expenses:", "{0:.2f}".format(coffee['Total Expenses'].std()))

# Plot a boxplot of the 'Total Expenses' column
coffee['Total Expenses'].plot(kind='box')
plt.show()
```

The statistical results provide an overview of the "Total Expenses" column in the dataset, which represents the expenses for the coffee chain. The **count** indicates the number of non-missing values in the "Total Expenses" column. In this case, there are **4,246** valid expense values.

The **mean** represents the average expense value across all the data points. The mean expense for the coffee chain is approximately 54.04. The **standard deviation** measures the dispersion or variability of the expense values. A higher standard deviation indicates greater variability in the expenses. In this case, the expense values have a standard deviation of approximately 32.33, suggesting a moderate spread in the expense data. The minimum value represents the smallest expense recorded in the dataset. In this case, the **minimum expense** is 10.00, indicating the lowest recorded expense value. The **25th percentile** is the value below which is 25% of the expense values fall. It is also known as the first quartile (Q1). In this case, the first quartile is 33.00, meaning

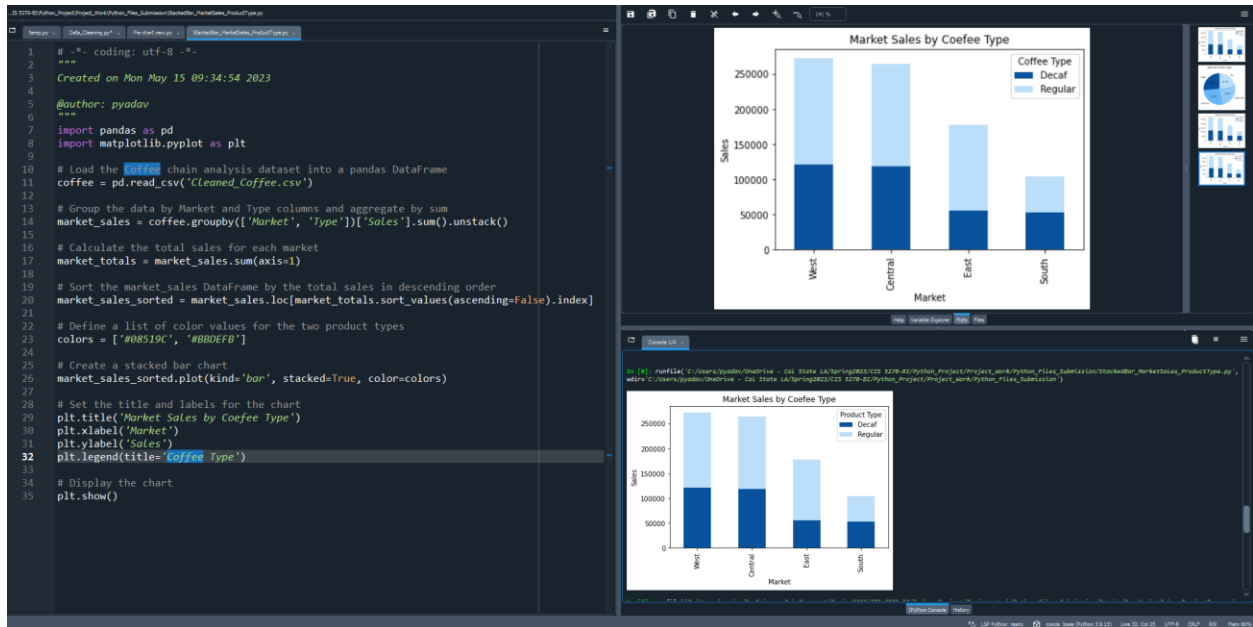
that 25% of the expenses are below this value. The **median** represents the middle value of the dataset when it is sorted in ascending order. It is also the **50th percentile** or the second quartile (Q2). In this case, the median expense is 46.00, indicating that 50% of the expenses fall below this value. The **75th percentile** is the value below which 75% of the expense values fall. It is also known as the third quartile (Q3). In this case, the third quartile is 65.00, meaning that 75% of the expenses are below this value. The **maximum value** represents the largest expense recorded in the dataset. In this case, the maximum expense is 190.00, indicating the highest recorded expense value.

The summary statistics provide insights into the distribution and central tendencies of the "Total Expenses" column. We can use this information to understand the average expenses, the spread of values, and the range of expenses incurred by the coffee chain. Additionally, examining quartiles helps identify the spread of expenses and the proportion of data falling within different percentile ranges.

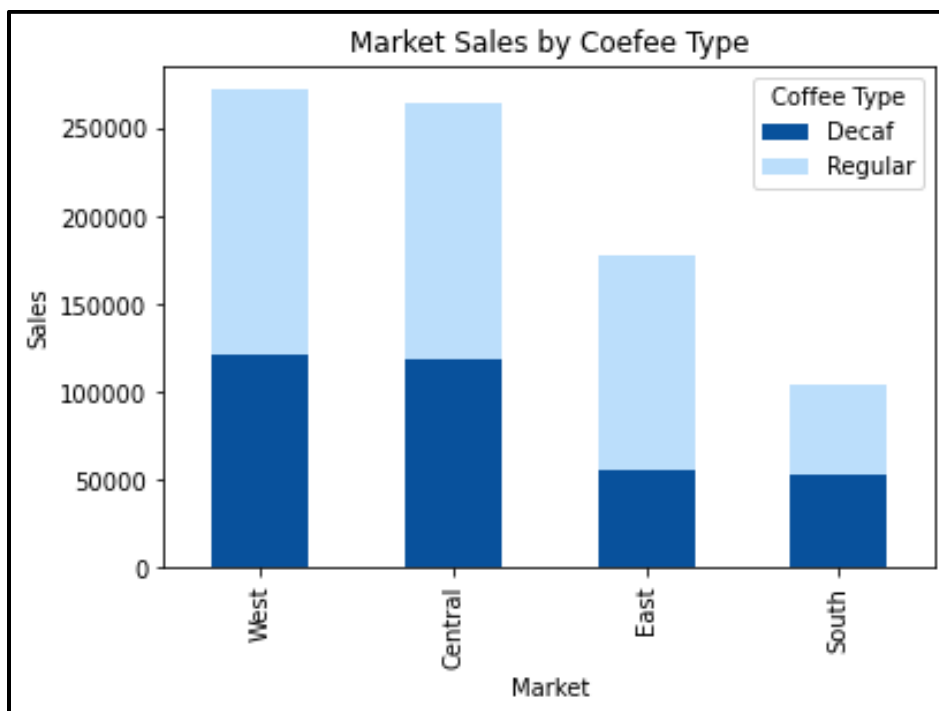
E. Analysis & Visualizations

1. What are the sales trends of different product types across various markets?

Full Screenshot:



Visual screenshot:



Plot Type: Stacked Bar Chart

Libraries: matplotlib.pyplot, pandas

Methods: read_csv(), sum(), groupby(), sort(), title(), xlabel(), ylabel(), legend(), show()

Insights:

The above-stacked bar chart analyzes the sales trends for different product types, namely Decaf and Regular Coffee, across various markets consisting of the West, Central, East, and South regions. By examining the relative heights of the bars in the graph, we can identify markets with higher and lower sales volumes and assess the contribution of each product type to the total sales in those markets. The visualization illustrates that the West and Central markets have the highest sales, exceeding \$250K, followed by the East, while the South market shows the lowest sales of around \$100K.

Furthermore, we can determine the product type with the highest sales and compare sales across different markets. The graph reveals that Regular Coffee, represented by the light blue color, is preferred over Decaf in all markets. This helps identify which product types are more dominant or popular within specific markets.

Overall, the stacked bar chart provides insights into sales distribution, market comparisons, and product type contributions, enabling us to identify market trends, preferences, and potential areas for growth or improvement in the coffee chain analysis.

Code Screenshot:

```

@author: pyadav
"""
import pandas as pd
import matplotlib.pyplot as plt

# Load the Coffee chain analysis dataset into a pandas DataFrame
coffee = pd.read_csv('Cleaned_Coffee.csv')

# Group the data by Market and Type columns and aggregate by sum
market_sales = coffee.groupby(['Market', 'Type'])['Sales'].sum().unstack()

# Calculate the total sales for each market
market_totals = market_sales.sum(axis=1)

# Sort the market_sales DataFrame by the total sales in descending order
market_sales_sorted = market_sales.loc[market_totals.sort_values(ascending=False).index]

# Define a list of color values for the two product types
colors = ['#08519C', '#BBDEFB']

# Create a stacked bar chart
market_sales_sorted.plot(kind='bar', stacked=True, color=colors)

# Set the title and labels for the chart
plt.title('Market Sales by Coffee Type')
plt.xlabel('Market')
plt.ylabel('Sales')
plt.legend(title='Coffee Type')

# Display the chart
plt.show()

```

Code Text:

```

import pandas as pd
import matplotlib.pyplot as plt

# Load the Coffee chain analysis dataset into a pandas DataFrame
coffee = pd.read_csv('Cleaned_Coffee.csv')

# Group the data by Market and Type columns and aggregate by sum
market_sales = coffee.groupby(['Market', 'Type'])['Sales'].sum().unstack()

# Calculate the total sales for each market
market_totals = market_sales.sum(axis=1)

# Sort the market_sales DataFrame by the total sales in descending order
market_sales_sorted = market_sales.loc[market_totals.sort_values(ascending=False).index]

# Define a list of color values for the two product types
colors = ['#08519C', '#BBDEFB']

# Create a stacked bar chart
market_sales_sorted.plot(kind='bar', stacked=True, color=colors)

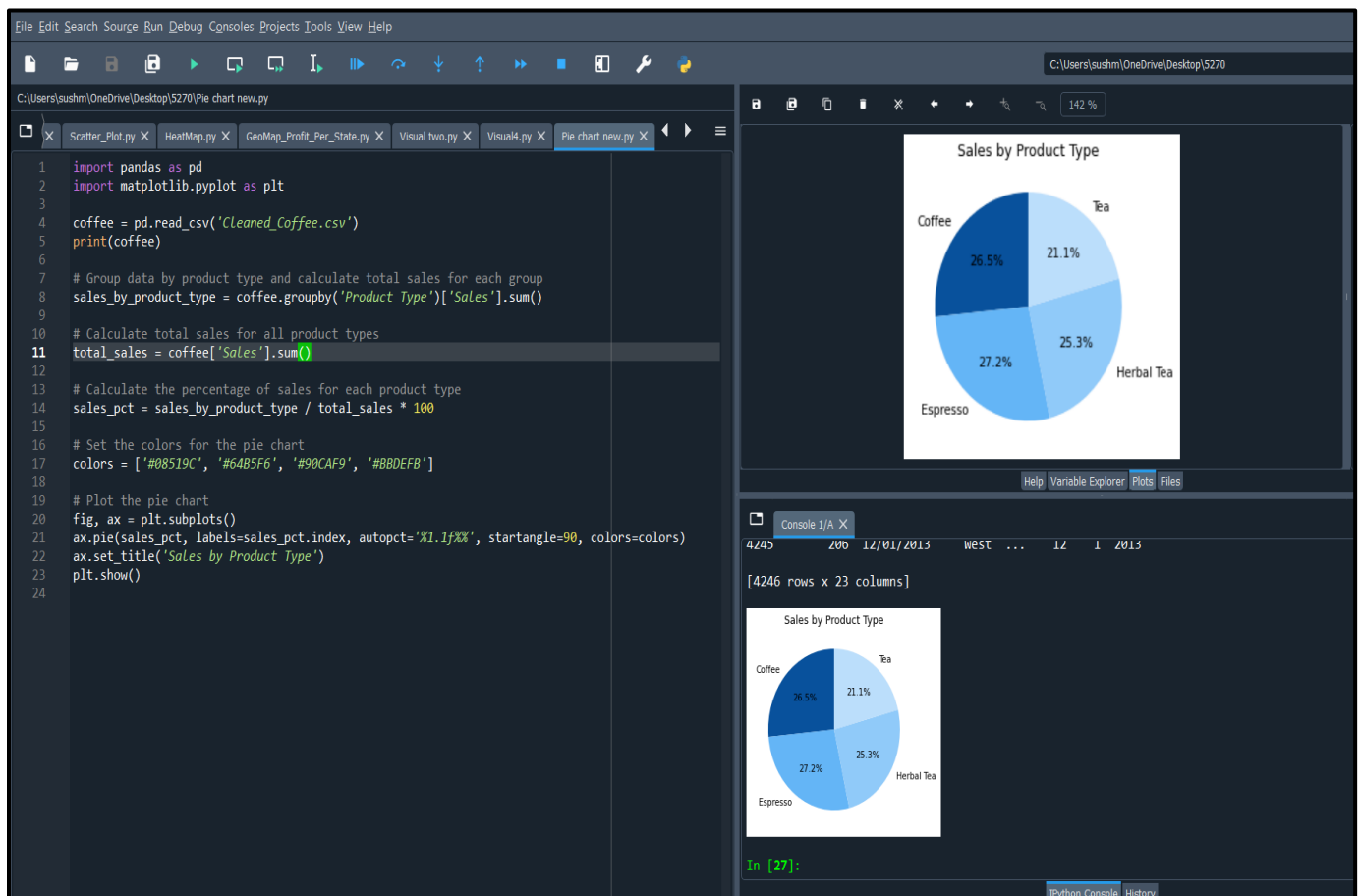
```

```
# Set the title and labels for the chart
plt.title('Market Sales by Coefee Type')
plt.xlabel('Market')
plt.ylabel('Sales')
plt.legend(title='Coffee Type')
```

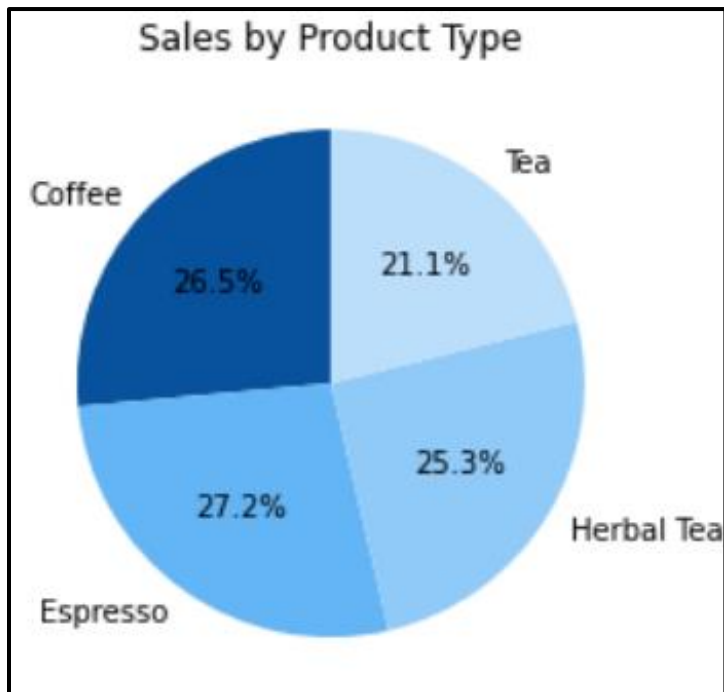
```
# Display the chart
plt.show()
```

2.What is the percentage contribution of each product type to total sales?

Full Screenshot



Visual screenshot



Plot Type: Pie Chart

Libraries: matplotlib.pyplot, pandas

Methods: read_csv(), sum(), groupby(), colors(), subplots(), set_title(), show()

Insights:

This analysis provides valuable insights into the percentage contribution of each product type to total sales in the coffee chain. Espresso stands out with the highest contribution of 27.2%, indicating strong demand and popularity. Coffee closely follows with 26.5% contribution, highlighting its prominence in the market. Herbal tea holds a notable share of 25.3%, while tea contributes 21.1% to total sales. These findings help businesses understand market dynamics, allocate resources effectively, develop targeted marketing strategies, and identify areas for product improvement. By capitalizing on these insights, businesses can optimize operations, meet customer demands, and drive growth in the coffee chain analysis.

Code Screenshot:

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 coffee = pd.read_csv('Cleaned_Coffee.csv')
5 print(coffee)
6
7 # Group data by product type and calculate total sales for each group
8 sales_by_product_type = coffee.groupby('Product Type')['Sales'].sum()
9
10 # Calculate total sales for all product types
11 total_sales = coffee['Sales'].sum()
12
13 # Calculate the percentage of sales for each product type
14 sales_pct = sales_by_product_type / total_sales * 100
15
16 # Set the colors for the pie chart
17 colors = ['#08519C', '#64B5F6', '#90CAF9', '#BBDEFB']
18
19 # Plot the pie chart
20 fig, ax = plt.subplots()
21 ax.pie(sales_pct, labels=sales_pct.index, autopct='%1.1f%%', startangle=90, colors=colors)
22 ax.set_title('Sales by Product Type')
23 plt.show()
24
```

Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt

coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)
# Group data by product type and calculate total sales for each group
sales_by_product_type = coffee.groupby('Product Type')['Sales'].sum()

# Calculate total sales for all product types
total_sales = coffee['Sales'].sum()

# Calculate the percentage of sales for each product type
sales_pct = sales_by_product_type / total_sales * 100

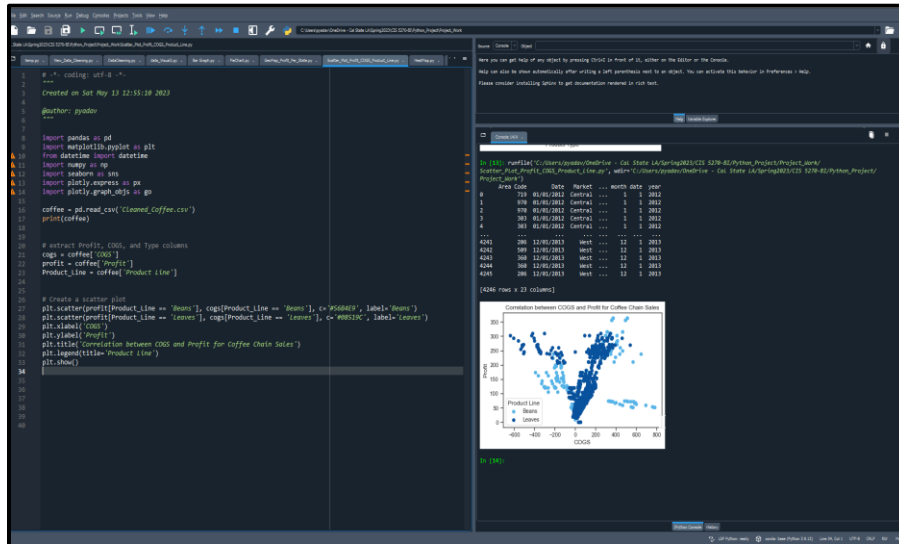
# Set the colors for the pie chart
colors = ['#08519C', '#64B5F6', '#90CAF9', '#BBDEFB']

# Plot the pie chart
fig, ax = plt.subplots()
```

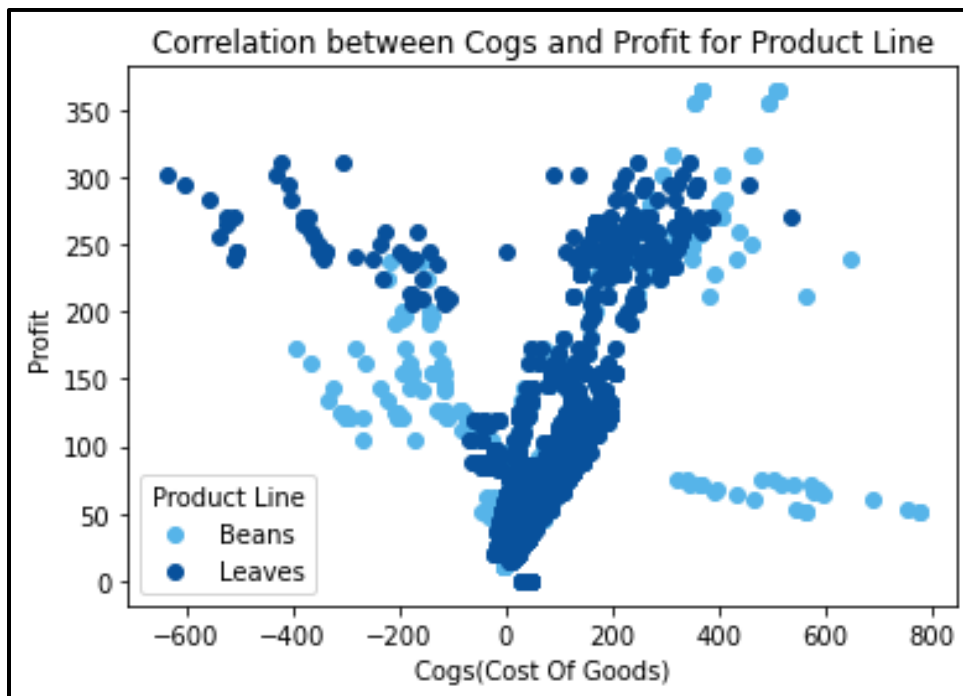
```
ax.pie(sales_pct, labels=sales_pct.index, autopct='%1.1f%%', startangle=90, colors=colors)
ax.set_title('Sales by Product Type')
plt.show()
```

3.How does the Profit and Cost of Goods correlate with the Product Line?

Full Screenshot



Visual screenshot



Plot Type: Scatter Plot

Libraries: matplotlib, pyplot, seaborn, pandas

Methods: read_csv(), sum(), groupby(), sort(), title(), xlabel(), ylabel(), legend(), show()

Insights:

From the scatter plot with COGS on the x-axis ranging from -600 to 800 and profit on the y-axis ranging from 0 to 350. The points that are higher on the y-axis indicate higher profits, while points that are lower indicate lower profits. The two distinct product lines are represented as Beans in light blue color and Leaves in dark blue color. This color differentiation helps us analyze and compare the performance of the two product lines. Each data point on the plot represents a specific product in the product line, with its corresponding profit and COGS values.

By considering the horizontal position of the data points (x-axis) for the cost of goods, we can evaluate the impact of the cost of goods sold on profitability. As we move from left to right on the x-axis (increasing COGS values), we can observe how profitability changes which indicates the positive correlation for Leaves Product. This helps us to provide insights into how changes in COGS impact profitability.

Overall, the scatter plot with distinct product lines allows us to visually analyze and compare the profitability of the beans and leaves product lines across different levels of cost of goods sold. It provides insights into the relationship between COGS and profit and helps identify trends or patterns that can inform strategic decision-making, such as optimizing pricing strategies, cost management, and resource allocation for each product line.

Code Screenshot:

```
@author: pyadav
"""
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)

# extract Profit, COGS, and Type columns
cogs = coffee['COGS']
profit = coffee['Profit']
Product_Line = coffee['Product Line']

# Create a scatter plot
plt.scatter(profit[Product_Line == 'Beans'], cogs[Product_Line == 'Beans'], c='#56B4E9', label='Beans')
plt.scatter(profit[Product_Line == 'Leaves'], cogs[Product_Line == 'Leaves'], c='#08519C', label='Leaves')
plt.xlabel('Cogs(Cost Of Goods)')
plt.ylabel('Profit')
plt.title('Correlation between Cogs and Profit for Product Line')
plt.legend(title='Product Line')
plt.show()
```

Code Text:

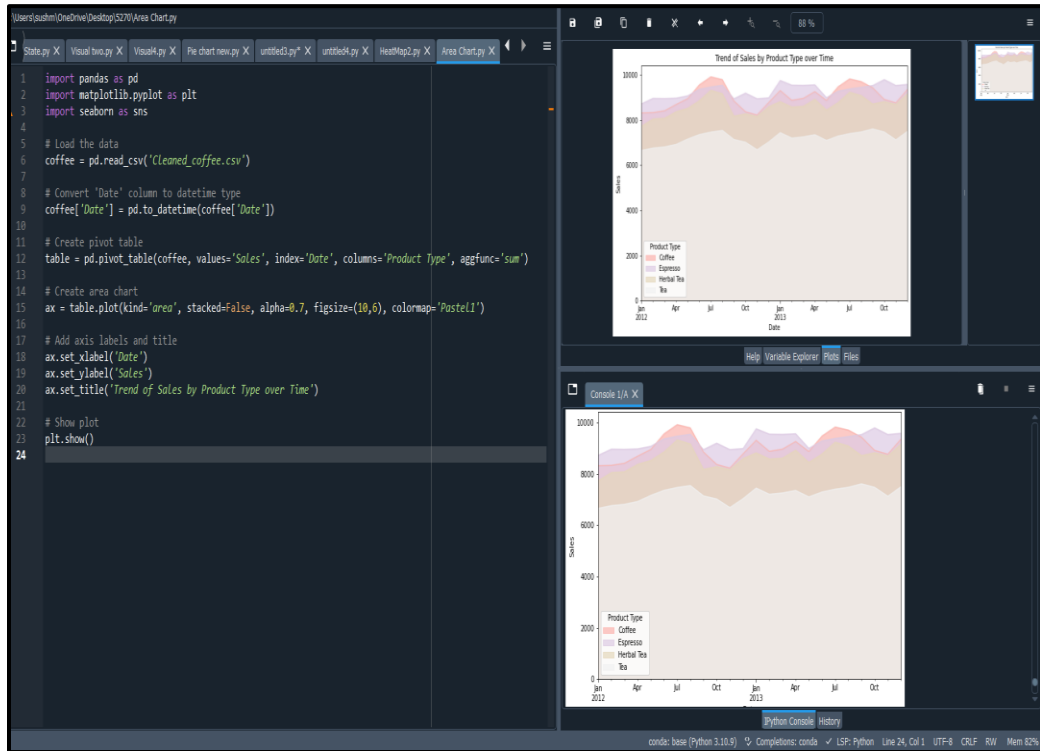
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)

# extract Profit, COGS, and Type columns
cogs = coffee['COGS']
profit = coffee['Profit']
Product_Line = coffee['Product Line']

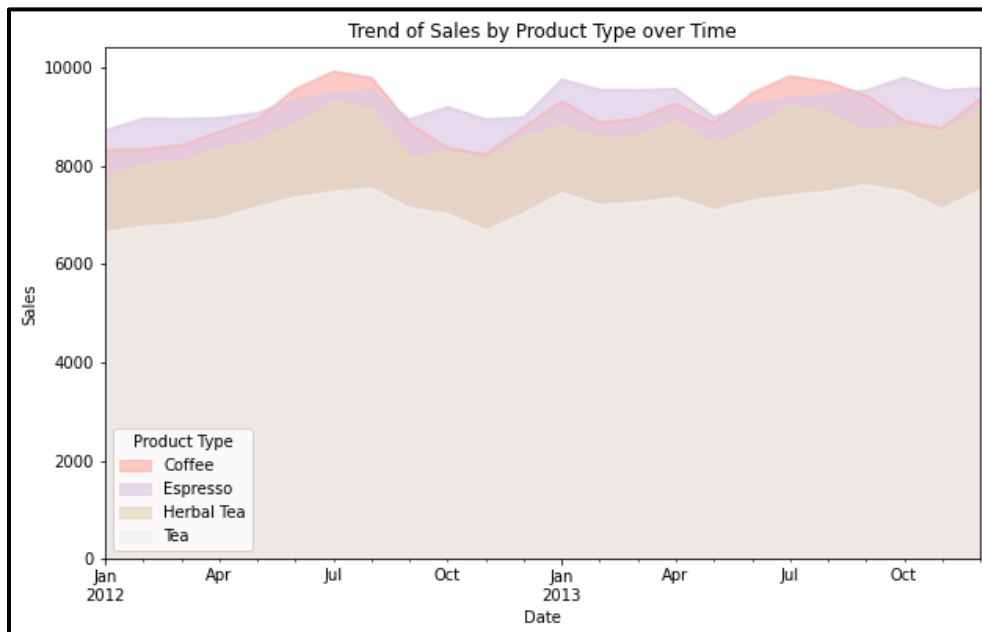
# Create a scatter plot
plt.scatter(profit[Product_Line == 'Beans'], cogs[Product_Line == 'Beans'], c='#56B4E9',
label='Beans')
plt.scatter(profit[Product_Line == 'Leaves'], cogs[Product_Line == 'Leaves'], c='#08519C',
label='Leaves')
plt.xlabel('Cogs(Cost Of Goods)')
plt.ylabel('Profit')
plt.title('Correlation between Cogs and Profit for Product Line')
plt.legend(title='Product Line')
plt.show()
```

4. What is the trend of sales for each product type over time?

Full Screenshot:



Visual screenshot



Plot Type: Area Line Chart

Libraries: matplotlib.pyplot, pandas

Methods: read_csv(), datetime(), pivot_table(), plot(), xlabel(), ylabel(), title(), show()

Insights:

The area line chart shows the trend of sales for each product type over time. Each product type is represented by a colored area on the chart, and the y-axis represents the total sales. By analyzing the chart, we can gain insights into the sales patterns and trends for each product type.

From the chart, we can observe the following insights:

Espresso: Espresso sales show a consistent and steady increase over time, indicating a positive sales trend for this product type.

Coffee: Coffee sales have a fluctuating pattern with periods of peaks and valleys. There are certain time periods where the sales increase significantly, followed by periods of slower sales.

Tea: Tea sales demonstrate a gradual upward trend, but the growth rate is relatively slower compared to Coffee and Espresso.

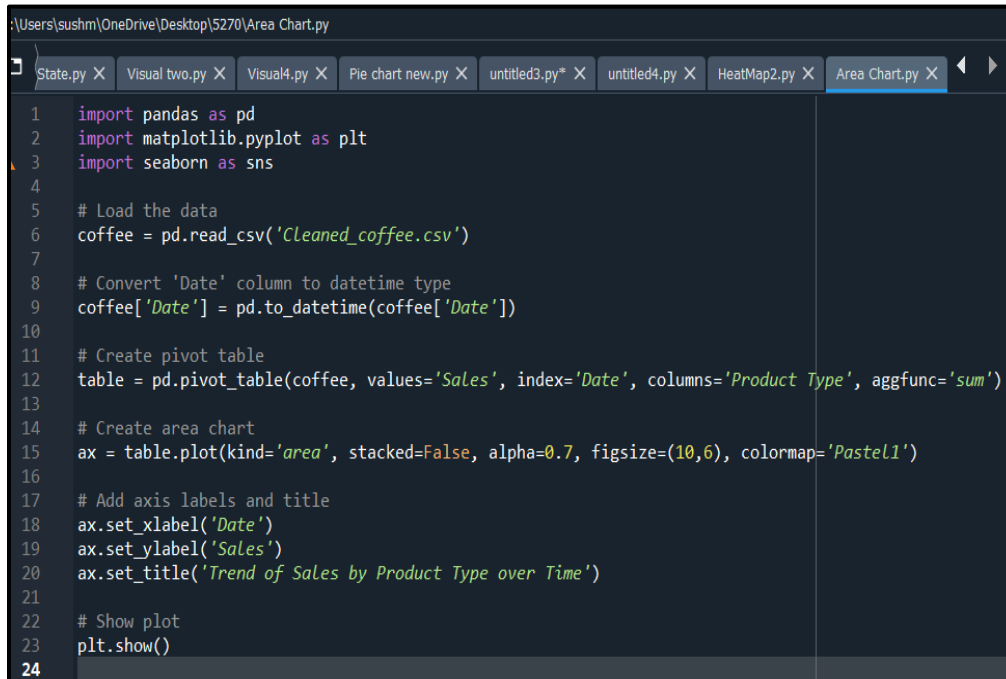
Herbal Tea: Herbal Tea sales show a relatively stable trend with minimal fluctuations. The sales remained consistent over time.

The area chart allows us to compare the sales trends for each product type and identify any significant changes or patterns. It provides a visual representation of the relative performance of each product type in terms of sales over time.

By analyzing the trend of sales for each product type over time, the coffee chain can gain insights into the popularity and demand for different products. This information can be used to optimize inventory management, adjust marketing strategies, and make informed decisions on product

offerings. For example, if Coffee sales continue to rise steadily, the coffee chain may consider allocating more resources to promote and expand its coffee product line.

Code Screenshot:

A screenshot of a code editor window with a dark theme. The title bar shows the file path: `\\Users\\sushm\\OneDrive\\Desktop\\5270\\Area Chart.py`. The editor has several tabs open: `State.py`, `Visual two.py`, `Visual4.py`, `Pie chart new.py`, `untitled3.py`, `untitled4.py`, `HeatMap2.py`, and `Area Chart.py` (which is the active tab). The code in the active tab is as follows:

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import seaborn as sns
4
5 # Load the data
6 coffee = pd.read_csv('Cleaned_coffee.csv')
7
8 # Convert 'Date' column to datetime type
9 coffee['Date'] = pd.to_datetime(coffee['Date'])
10
11 # Create pivot table
12 table = pd.pivot_table(coffee, values='Sales', index='Date', columns='Product Type', aggfunc='sum')
13
14 # Create area chart
15 ax = table.plot(kind='area', stacked=False, alpha=0.7, figsize=(10,6), colormap='Pastel1')
16
17 # Add axis labels and title
18 ax.set_xlabel('Date')
19 ax.set_ylabel('Sales')
20 ax.set_title('Trend of Sales by Product Type over Time')
21
22 # Show plot
23 plt.show()
24
```

Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the data
coffee = pd.read_csv('Cleaned_coffee.csv')

# Convert 'Date' column to datetime type
coffee['Date'] = pd.to_datetime(coffee['Date'])

# Create pivot table
table = pd.pivot_table(coffee, values='Sales', index='Date', columns='Product Type',
aggfunc='sum')

# Create area chart
ax = table.plot(kind='area', stacked=False, alpha=0.7, figsize=(10,6), colormap='Pastel1')

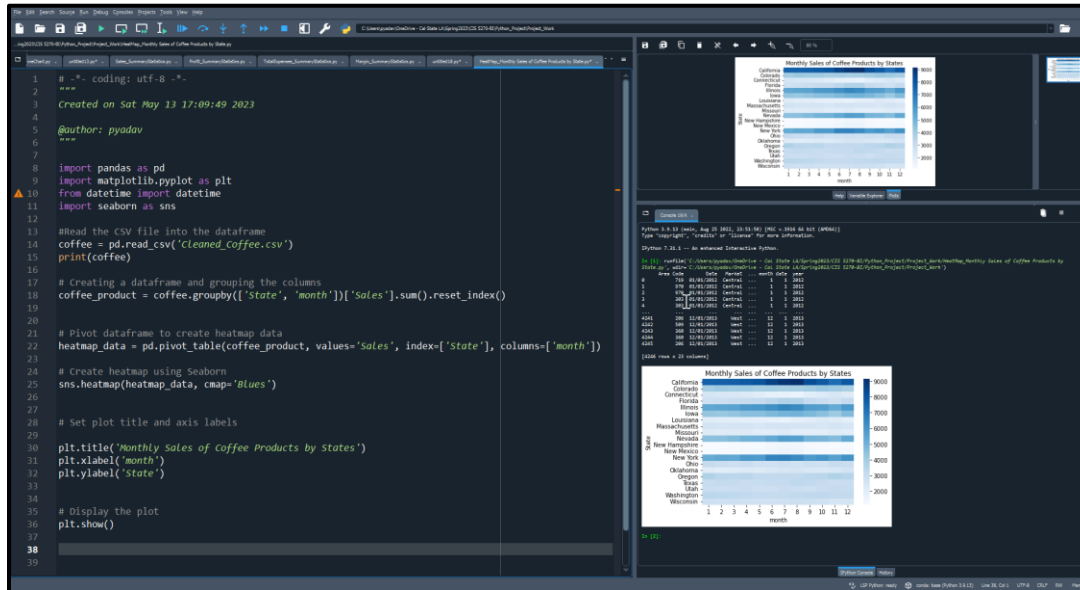
# Add axis labels and title
ax.set_xlabel('Date')
```

```
ax.set_ylabel('Sales')
ax.set_title('Trend of Sales by Product Type over Time')
```

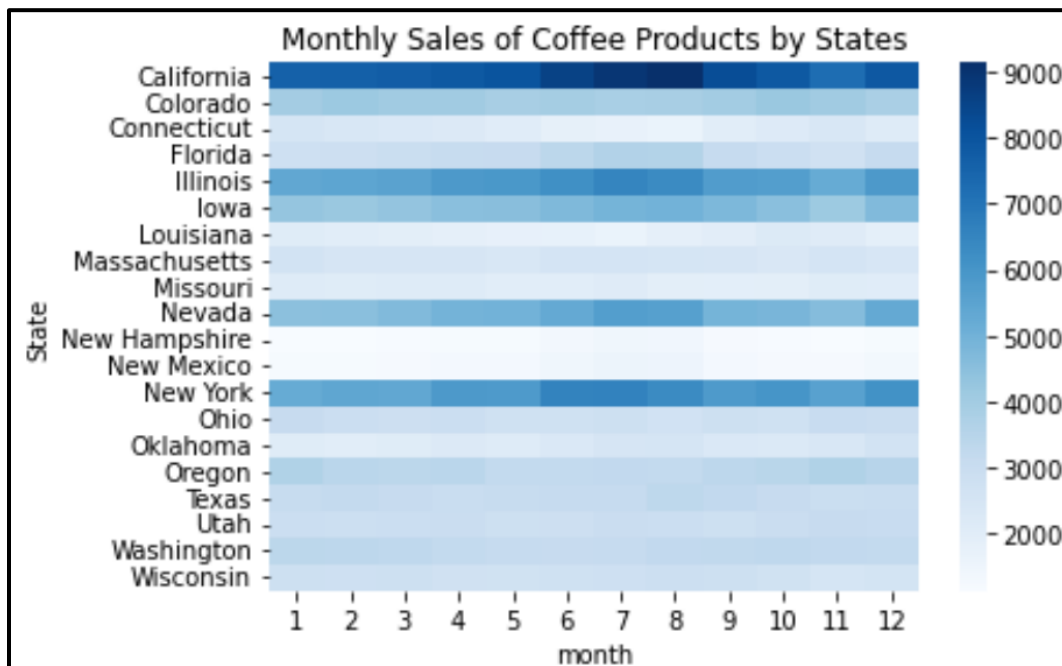
```
# Show plot
plt.show()
```

5. Which coffee product sells the most in a particular state?

Full Screenshot



Visual screenshot



Plot Type: Heat Map

Libraries: matplotlib.pyplot, pandas, Seaborn

Methods: read_csv(), groupby(), pivot_table(), heatmap(), xlabel(), ylabel(), title(), show()

Insights:

The heat map displays the monthly sales of coffee products across different states. By examining the map, we can observe the varying levels of sales in each state. The darker shades of blue represent higher sales, indicating that these states have a stronger demand for coffee products. Conversely, the lighter shades of blue indicate lower sales, suggesting that these states may have a relatively lower demand.

Based on the heatmap, we can identify that California has the highest sales among all the states. This implies that California has a significant market share and demonstrates a strong demand for coffee products. Following California, we can see that Colorado and Connecticut also have relatively higher sales compared to other states.

The insights provided by the heat map are beneficial for the coffee chain's decision-making process. By recognizing the states with the highest sales, the coffee chain can strategically allocate resources and develop tailored marketing strategies to further capitalize on these markets. For example, they might consider expanding their operations, enhancing distribution networks, or intensifying promotional efforts in states like California, Colorado, and Connecticut to maximize their sales potential.

Additionally, the heat map enables the coffee chain to identify states with lower sales. This information can help in pinpointing potential growth opportunities or areas that require targeted marketing campaigns to increase brand awareness and stimulate demand.

Overall, the heat map serves as a visual tool that offers a comprehensive overview of the sales performance across different states. It assists in identifying the most promising markets and guiding strategic decision-making to drive the coffee chain's growth and profitability.

Code Screenshot:

```
# -*- coding: utf-8 -*-
"""
Created on Sat May 13 17:09:49 2023

@author: pyadav
"""

import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)

# Creating a dataframe and grouping the columns
coffee_product = coffee.groupby(['State', 'month'])['Sales'].sum().reset_index()

# Pivot dataframe to create heatmap data
heatmap_data = pd.pivot_table(coffee_product, values='Sales', index=['State'], columns=['month'])

# Create heatmap using Seaborn
sns.heatmap(heatmap_data, cmap='Blues')

# Set plot title and axis labels
plt.title('Monthly Sales of Coffee Products by States')
plt.xlabel('month')
plt.ylabel('State')

# Display the plot
plt.show()
```

Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns

#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)

# Creating a dataframe and grouping the columns
coffee_product = coffee.groupby(['State', 'month'])['Sales'].sum().reset_index()

# Pivot dataframe to create heatmap data
```

```
heatmap_data = pd.pivot_table(coffee_product, values='Sales', index=['State'],
columns=['month'])
```

```
# Create heatmap using Seaborn
sns.heatmap(heatmap_data, cmap='Blues')
```

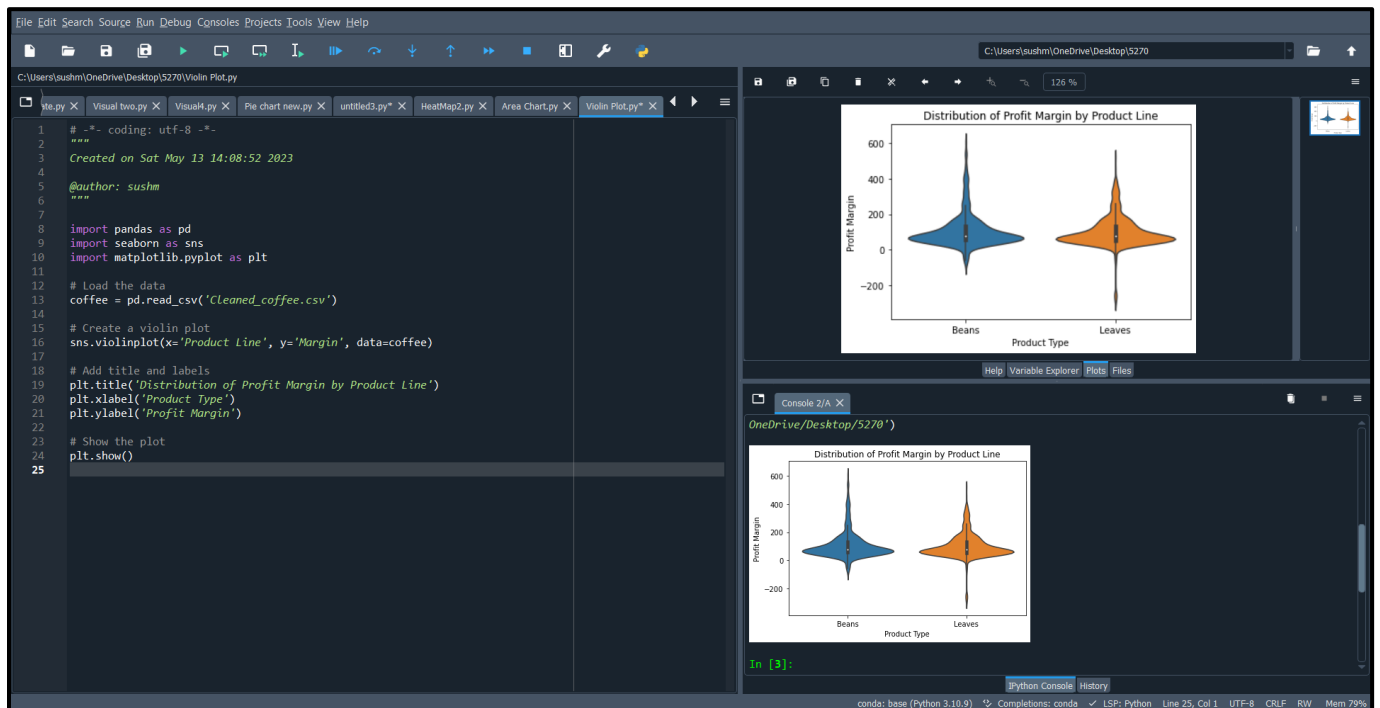
```
# Set plot title and axis labels
```

```
plt.title('Monthly Sales of Coffee Products by States')
plt.xlabel('month')
plt.ylabel('State')
```

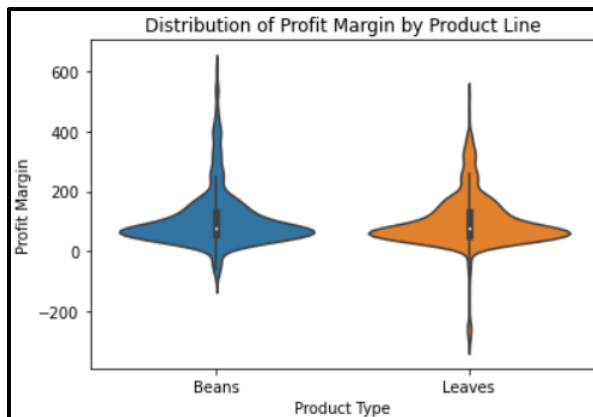
```
# Display the plot
plt.show()
```

6.What is the Profit Margin for each product type?

Full Screenshot



Visual screenshot



Plot Type: Violin Plot

Libraries: matplotlib.pyplot, pandas, Seaborn

Methods: read_csv(), violinplot(), xlabel(), ylabel(), show()

Insights:

The violin plot shows the distribution of profit margin for each product line in the coffee chain dataset. Each violin plot represents a product line, with the width of the plot indicating the frequency of the profit margin values and the height indicating the range of values.

From the plot, we can see that the profit margin distribution for each product line is roughly symmetric, with most of the profit margins falling in the middle of the range. This suggests that the coffee chain is generally successful at maintaining consistent profit margins across its product lines.

In terms of the differences between product lines, the violin plot suggests that the profit margins for Leaves are slightly lower than those for the other product line. However, the differences between the product lines are not very large.

The violin plot provides a useful summary of the distribution of profit margins for each product line in the coffee chain dataset, allowing us to compare the profitability of different product lines and identify any outliers or anomalies in the data.

Code Screenshot:

```
1  # -*- coding: utf-8 -*-
2  """
3  Created on Sat May 13 14:08:52 2023
4
5  @author: sushm
6  """
7
8  import pandas as pd
9  import seaborn as sns
10 import matplotlib.pyplot as plt
11
12 # Load the data
13 coffee = pd.read_csv('Cleaned_coffee.csv')
14
15 # Create a violin plot
16 sns.violinplot(x='Product Line', y='Margin', data=coffee)
17
18 # Add title and labels
19 plt.title('Distribution of Profit Margin by Product Line')
20 plt.xlabel('Product Type')
21 plt.ylabel('Profit Margin')
22
23 # Show the plot
24 plt.show()
25
```

Code Text:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the data
coffee = pd.read_csv('Cleaned_coffee.csv')

# Create a violin plot
sns.violinplot(x='Product Line', y='Margin', data=coffee)

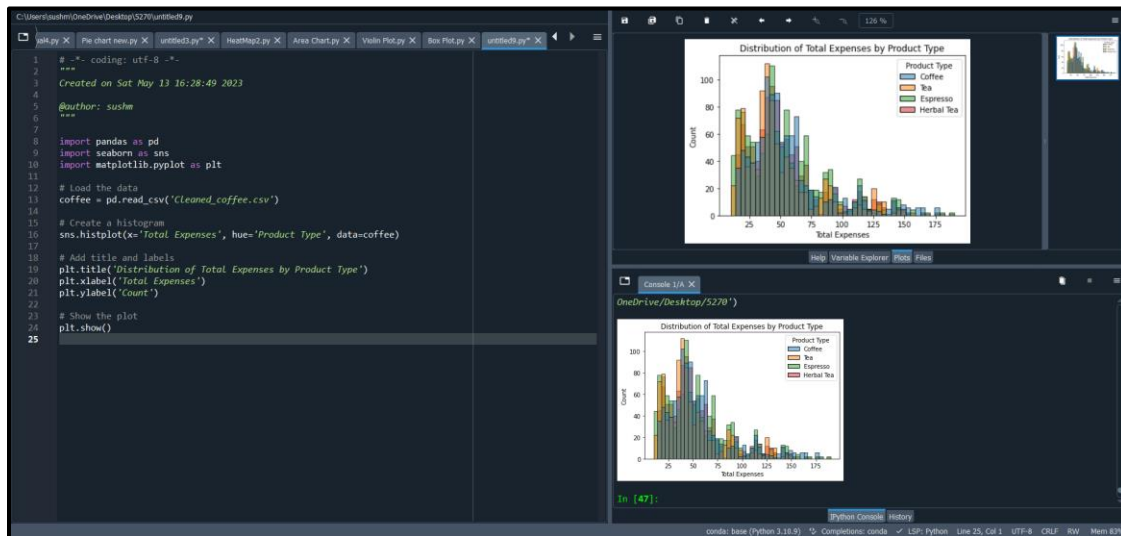
# Add title and labels
```

```
plt.title('Distribution of Profit Margin by Product Line')
plt.xlabel('Product Type')
plt.ylabel('Profit Margin')
```

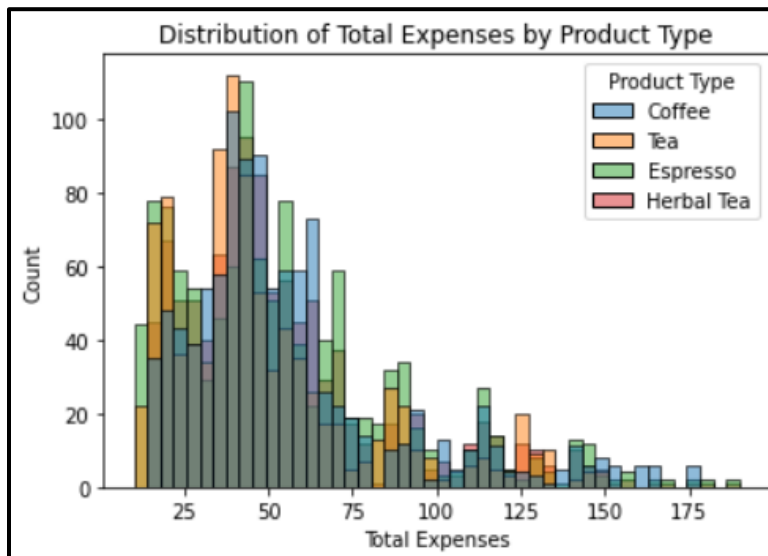
```
# Show the plot
plt.show()
```

7.What is the distribution of total expenses across different product types?

Full Screenshot



Visual screenshot



Plot Type: Histogram

Libraries: matplotlib.pyplot, pandas, Seaborn

Methods: read_csv(), histplot(), title(), xlabel(), ylabel(), show()

Insights:

Based on the histogram of total expenses for different product types, it appears that the coffee chain has been able to manage its expenses efficiently for most products as most expenses are clustered around the lower values. However, the skewed distribution towards the right suggests that there are some products that have significantly higher expenses compared to others, namely Coffee, Espresso, and Herbal Tea.

These insights could be useful for the coffee chain in making informed decisions about its resource allocation and investment strategies. For instance, the higher expenses for Coffee, Espresso, and Herbal Tea may indicate that these product types require more investment in terms of production or marketing to maintain their market share. Alternatively, it could also suggest that these products are more popular among customers and hence require higher expenses to cater to the demand.

Overall, the analysis of the histogram provides valuable insights into the distribution of total expenses across different product types, which can help the coffee chain in optimizing its operations and maximizing its profits.

Code Screenshot:

```
1  # -*- coding: utf-8 -*-
2  """
3  Created on Sat May 13 16:28:49 2023
4
5  @author: sushm
6  """
7
8  import pandas as pd
9  import seaborn as sns
10 import matplotlib.pyplot as plt
11
12 # Load the data
13 coffee = pd.read_csv('Cleaned_coffee.csv')
14
15 # Create a histogram
16 sns.histplot(x='Total Expenses', hue='Product Type', data=coffee)
17
18 # Add title and labels
19 plt.title('Distribution of Total Expenses by Product Type')
20 plt.xlabel('Total Expenses')
21 plt.ylabel('Count')
22
23 # Show the plot
24 plt.show()
```

Code Text:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the data
coffee = pd.read_csv('Cleaned_coffee.csv')

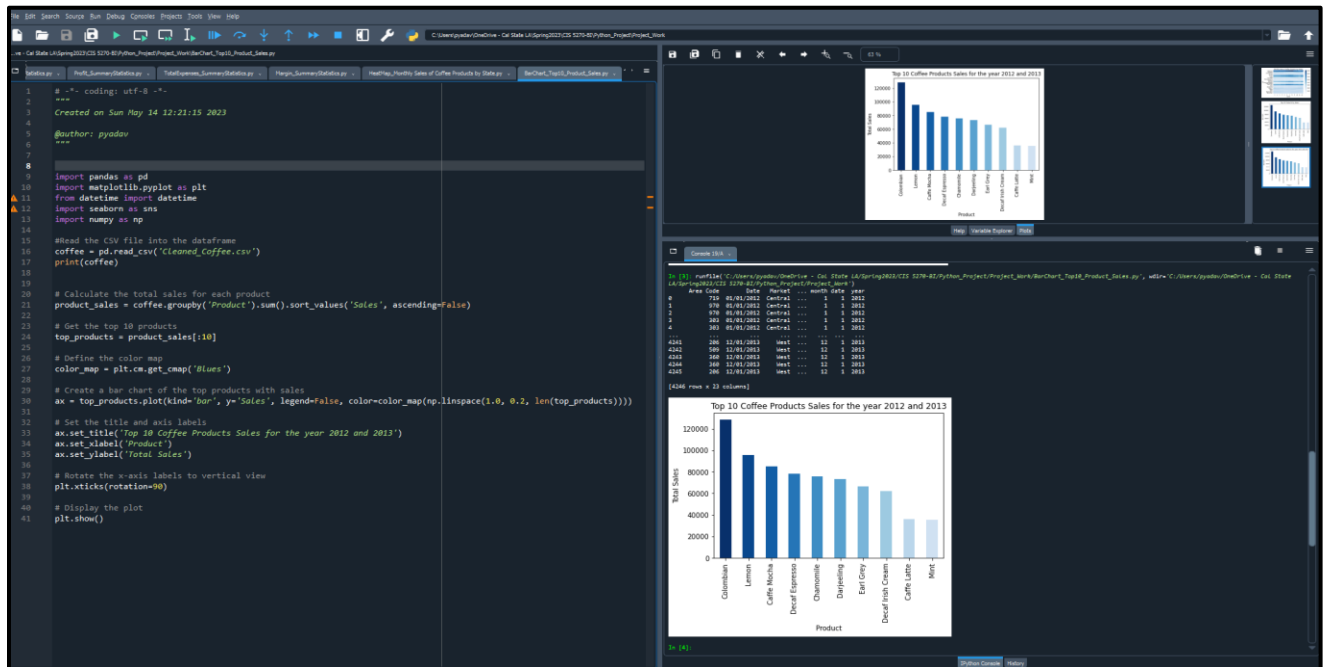
# Create a histogram
sns.histplot(x="Total Expenses", hue='Product Type', data=coffee)

# Add title and labels
plt.title('Distribution of Total Expenses by Product Type')
plt.xlabel('Total Expenses')
plt.ylabel('Count')

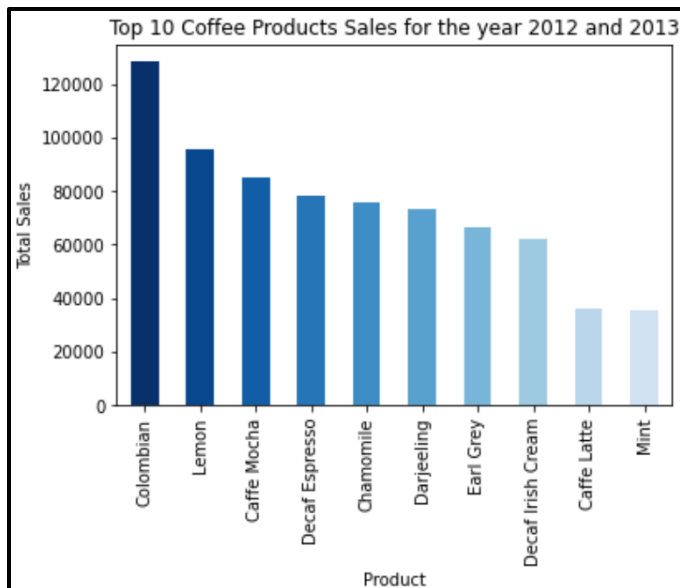
# Show the plot
plt.show()
```


8. Which Product had the highest sales in the year 2012 and 2013?

Full Screenshot



Visual screenshot



Plot Type: Bar Chart

Libraries: matplotlib.pyplot, pandas, Seaborn

Methods: read_csv(), groupby(), sum(), sort_values(), cmap(), Plot(), title(), xlabel(), ylabel(), show(), xticks(), rotation()

Insights:

Based on the bar chart, we can see that the product with the highest sales in both 2012 and 2013 was the Colombian. This indicates that Colombian was the most popular and profitable product during those years.

In addition, we can see that the top 10 products by sales were dominated by different variations of coffee, such as Decaf Coffee, Espresso, and Coffee Beans. This suggests that coffee products were the main driver of sales for the coffee chain during that period.

Overall, the bar chart provides insights into the sales performance of different coffee products for the years 2012 and 2013, which can help the coffee chain in identifying its top-performing products and focusing its marketing and operational strategies on those products to maximize its profits.

Code Screenshot:

```
# -*- coding: utf-8 -*-
"""
Created on Sun May 14 12:21:15 2023

@author: pyadav
"""

import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns
import numpy as np

# Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)

# Calculate the total sales for each product
product_sales = coffee.groupby('Product').sum().sort_values('Sales', ascending=False)

# Get the top 10 products
top_products = product_sales[:10]

# Define the color map
color_map = plt.cm.get_cmap('Blues')

# Create a bar chart of the top products with sales
ax = top_products.plot(kind='bar', y='Sales', legend=False, color=color_map(np.linspace(1.0, 0.2, len(top_products))))

# Set the title and axis labels
ax.set_title('Top 10 Coffee Products Sales for the year 2012 and 2013')
ax.set_xlabel('Product')
ax.set_ylabel('Total Sales')

# Rotate the x-axis labels to vertical view
plt.xticks(rotation=90)

# Display the plot
plt.show()
```

Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import seaborn as sns
import numpy as np

#Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)

# Calculate the total sales for each product
product_sales = coffee.groupby('Product').sum().sort_values('Sales', ascending=False)

# Get the top 10 products
top_products = product_sales[:10]

# Define the color map
color_map = plt.cm.get_cmap('Blues')

# Create a bar chart of the top products with sales
ax = top_products.plot(kind='bar', y='Sales', legend=False, color=color_map(np.linspace(1.0,
0.2, len(top_products))))

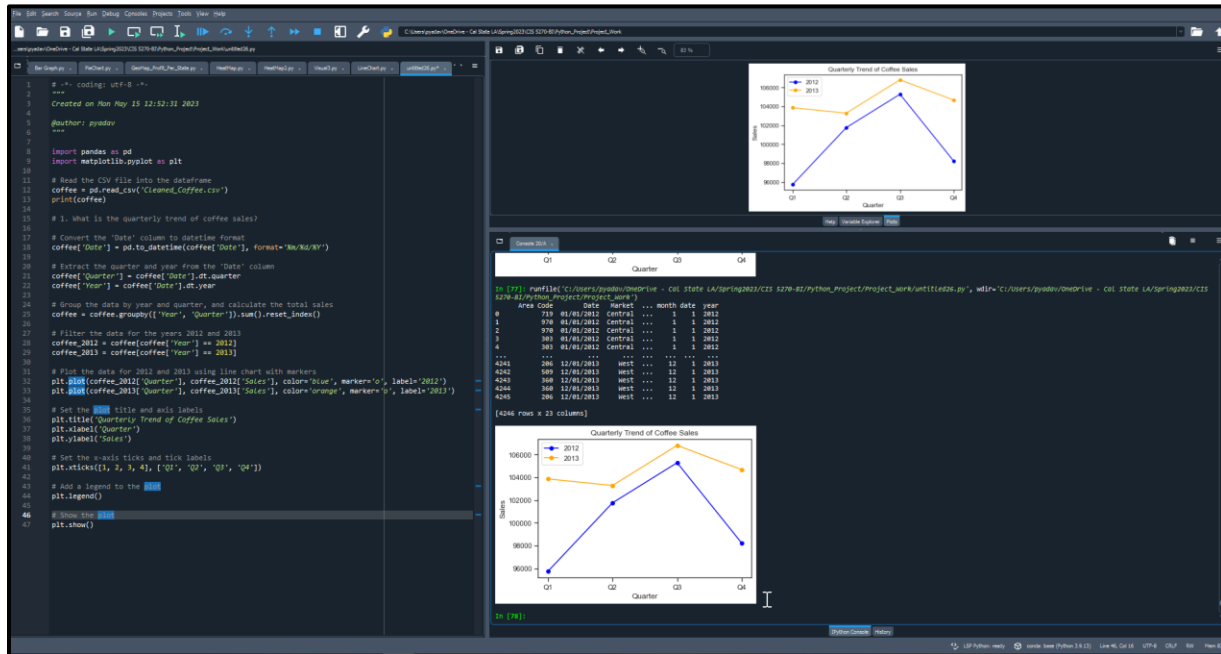
# Set the title and axis labels
ax.set_title('Top 10 Coffee Products Sales for the year 2012 and 2013')
ax.set_xlabel('Product')
ax.set_ylabel('Total Sales')

# Rotate the x-axis labels to vertical view
plt.xticks(rotation=90)

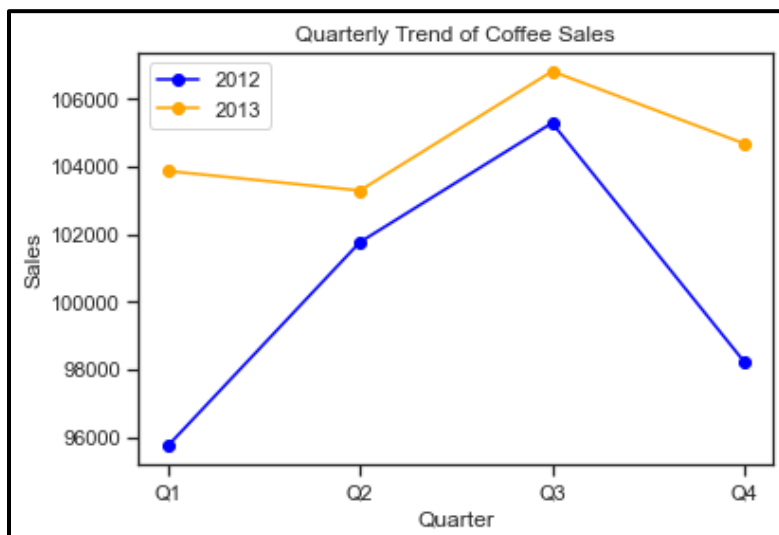
# Display the plot
plt.show()
```

9. What is the quarterly trend of coffee sales?

Full Screenshot



Visual screenshot



Plot Type: Line Chart with markers

Libraries: matplotlib.pyplot, pandas

Methods: read_csv(), sum(), groupby(), reset_index(), title(), xlabel(), ylabel(), legend(), show()

Insights:

The line chart showcases the quarterly trend of coffee sales for the years 2012 and 2013. The x-axis represents the quarters (Q1, Q2, Q3, Q4), while the y-axis represents the total sales. By observing the direction of the line graph, we can clearly determine that in the year 2013 coffee sales have been better as compared to the year 2012 with the highest sales of more than \$106K for the year 2013 Q3. Moreover, analyzing the peaks and valleys in the line graph reveals seasonal patterns in coffee sales. The sales are consistently increasing in the Q2 and Q3 quarters for both years which indicates the seasonal demand for coffee. This insight can help businesses align their strategies, such as inventory management, production planning, and marketing campaigns, to leverage periods of increased demand and optimize their coffee sales.

Code Screenshot:

```
import pandas as pd
import matplotlib.pyplot as plt

# Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)

# 1. What is the quarterly trend of coffee sales?

# Convert the 'Date' column to datetime format
coffee['Date'] = pd.to_datetime(coffee['Date'], format='%m/%d/%Y')

# Extract the quarter and year from the 'Date' column
coffee['Quarter'] = coffee['Date'].dt.quarter
coffee['Year'] = coffee['Date'].dt.year

# Group the data by year and quarter, and calculate the total sales
coffee = coffee.groupby(['Year', 'Quarter']).sum().reset_index()

# Filter the data for the years 2012 and 2013
coffee_2012 = coffee[coffee['Year'] == 2012]
coffee_2013 = coffee[coffee['Year'] == 2013]

# Plot the data for 2012 and 2013 using line chart with markers
plt.plot(coffee_2012['Quarter'], coffee_2012['Sales'], color='blue', marker='o', label='2012')
plt.plot(coffee_2013['Quarter'], coffee_2013['Sales'], color='orange', marker='o', label='2013')

# Set the plot title and axis labels
plt.title('Quarterly Trend of Coffee Sales')
plt.xlabel('Quarter')
plt.ylabel('Sales')

# Set the x-axis ticks and tick labels
plt.xticks([1, 2, 3, 4], ['Q1', 'Q2', 'Q3', 'Q4'])

# Add a legend to the plot
plt.legend()

# Show the plot
plt.show()
```

Code Text:

```
import pandas as pd
import matplotlib.pyplot as plt

# Read the CSV file into the dataframe
coffee = pd.read_csv('Cleaned_Coffee.csv')
print(coffee)

# 1. What is the quarterly trend of coffee sales?

# Convert the 'Date' column to datetime format
coffee['Date'] = pd.to_datetime(coffee['Date'], format='%m/%d/%Y')

# Extract the quarter and year from the 'Date' column
coffee['Quarter'] = coffee['Date'].dt.quarter
coffee['Year'] = coffee['Date'].dt.year

# Group the data by year and quarter, and calculate the total sales
coffee = coffee.groupby(['Year', 'Quarter']).sum().reset_index()

# Filter the data for the years 2012 and 2013
coffee_2012 = coffee[coffee['Year'] == 2012]
coffee_2013 = coffee[coffee['Year'] == 2013]

# Plot the data for 2012 and 2013 using line chart with markers
plt.plot(coffee_2012['Quarter'], coffee_2012['Sales'], color='blue', marker='o', label='2012')
plt.plot(coffee_2013['Quarter'], coffee_2013['Sales'], color='orange', marker='o', label='2013')

# Set the plot title and axis labels
plt.title('Quarterly Trend of Coffee Sales')
plt.xlabel('Quarter')
plt.ylabel('Sales')

# Set the x-axis ticks and tick labels
plt.xticks([1, 2, 3, 4], ['Q1', 'Q2', 'Q3', 'Q4'])

# Add a legend to the plot
plt.legend()

# Show the plot
plt.show()
```

F. References

- [1] Ashby, P. (2022, January 4). *The history of the US Coffee Shop*. Perfect Daily Grind. <https://perfectdailygrind.com/2022/01/the-history-of-the-us-coffee-shop/>
- [2] Brennan, S. (2023, February 6). *10 most famous coffee chains in the World (2023 update)*. Coffee Affection. <https://coffeeaffection.com/most-famous-coffee-chains/>
- [3] Allen, A. (2022, October 31). *The Ultimate Guide to the Coffee Shop Industry*^[OBJ]. Aaron Allen & Associates, Global Restaurant Consultants. <https://aaronallen.com/blog/coffee-shop-industry>