Data Analytics with Cognos Project

PROJECT TITLE:

PRODUCT SALES ANALYSIS

PROJECT DOCUMENTATION
AND SUBMISSION PHASE

TEAM MEMBERS:

MUTHU RATHNASHRI C	2021115068
NITHYA SREE K	2021115071
MUTHAMIZH VAANJINATHAN M	2021115067
NAREN KARTHICK THIRUVENGADAM	2021115069
PRASAAD S	2021115307

OBJECTIVE

The primary goal of the project is to optimize product sales through innovative analysis methods, leveraging data-driven insights.

DESIGN THINKING PROCESS

Empathize:

Understanding the needs of customers, market trends, and sales patterns.

Define:

Defining clear objectives for the analysis, focusing on improving sales performance and customer satisfaction.

Ideate:

Generating creative ideas for data collection, preprocessing, analysis techniques, and visualization methods.

Prototype:

Developing a preliminary model for analysis, incorporating various data sources and tools.

Test:

Validating the prototype, refining the analysis techniques, and ensuring the actionable insights are meaningful and practical.

DEVELOPMENT PHASES

Data Collection and Aggregation:

Gathering sales transactions, customer demographics, and market trend data from diverse sources.

Data Cleaning and Preprocessing:

Ensuring accuracy and consistency of the collected data, preparing it for analysis.

Advanced Analytics:

Utilizing techniques like data mining and machine learning to identify patterns, correlations, and outliers.

Data Visualization:

Using IBM Cognos and other visualization tools to present the findings in a visually comprehensible format.

Predictive and Prescriptive Analytics:

Implementing predictive models for sales forecasting and prescriptive analytics to provide actionable recommendations.

MONITORING AND FEEDBACK

Establishing continuous monitoring processes and feedback loops to adapt strategies based on market dynamics.

Analysis Objectives, Data Collection Process, Data Visualization, and Actionable Insights

Analysis Objectives:

Enhancing sales performance.

Understanding customer behavior and preferences.

Identifying market trends and patterns.

Improving inventory management and marketing strategies.

Data Collection Process:

Gathering sales transactions data from POS systems.

Collecting customer demographics through surveys and online profiles.

Tracking market trends using industry reports and online databases.

Data Visualization using IBM Cognos:

Utilizing IBM Cognos to create interactive dashboards and visual representations of sales data.

Visualizing customer segmentation, product performance, and market trends for comprehensive analysis.

DERIVED ACTIONABLE INSIGHTS

Customer Segmentation:

Identifying high-value customer segments for targeted marketing efforts.

Product Performance Analysis:

Evaluating the popularity of products to optimize inventory levels.

Market Trends:

Understanding emerging trends to introduce new products or modify existing ones.

Seasonal Analysis:

Anticipating demand fluctuations based on seasonal patterns for inventory planning.

GUIDING INVENTORY MANAGEMENT AND MARKETING STRATEGIES USING INSIGHTS

Inventory Management

Optimizing Stock Levels:

Using demand forecasts to maintain optimal inventory levels, preventing overstocking or stockouts.

Introducing Seasonal Products:

Introducing seasonal products based on historical trends to capitalize on peak demand periods.

Supplier Collaboration:

Collaborating with suppliers based on demand predictions, ensuring timely restocking.

MARKETING STRATEGIES

Targeted Marketing Campaigns:

Tailoring marketing campaigns to specific customer segments identified through analysis, increasing engagement and conversions.

Personalized Offers:

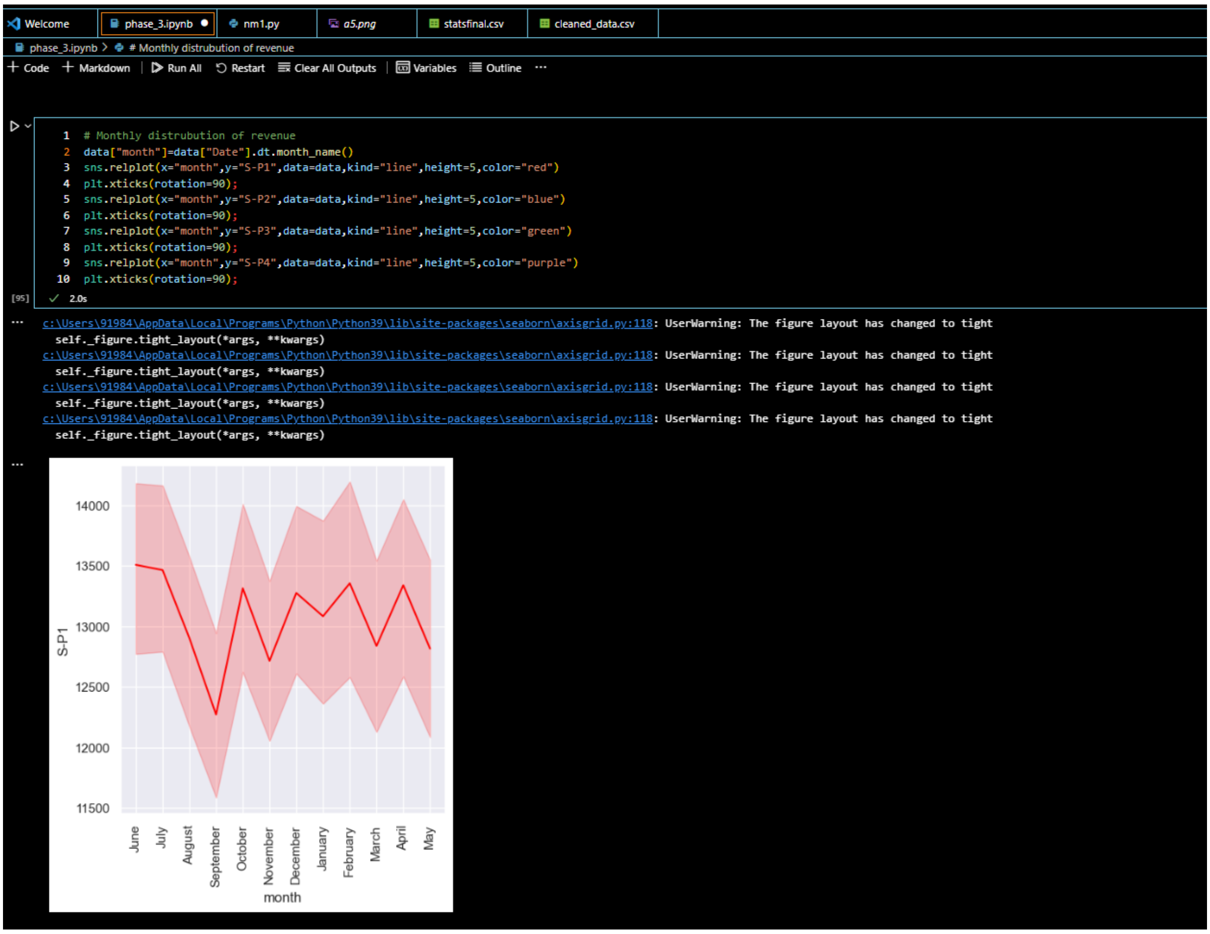
Creating personalized offers for customers based on their preferences, increasing customer loyalty and sales.

Market Expansion:

Identifying untapped markets or customer segments for expanding marketing efforts, driving growth opportunities.

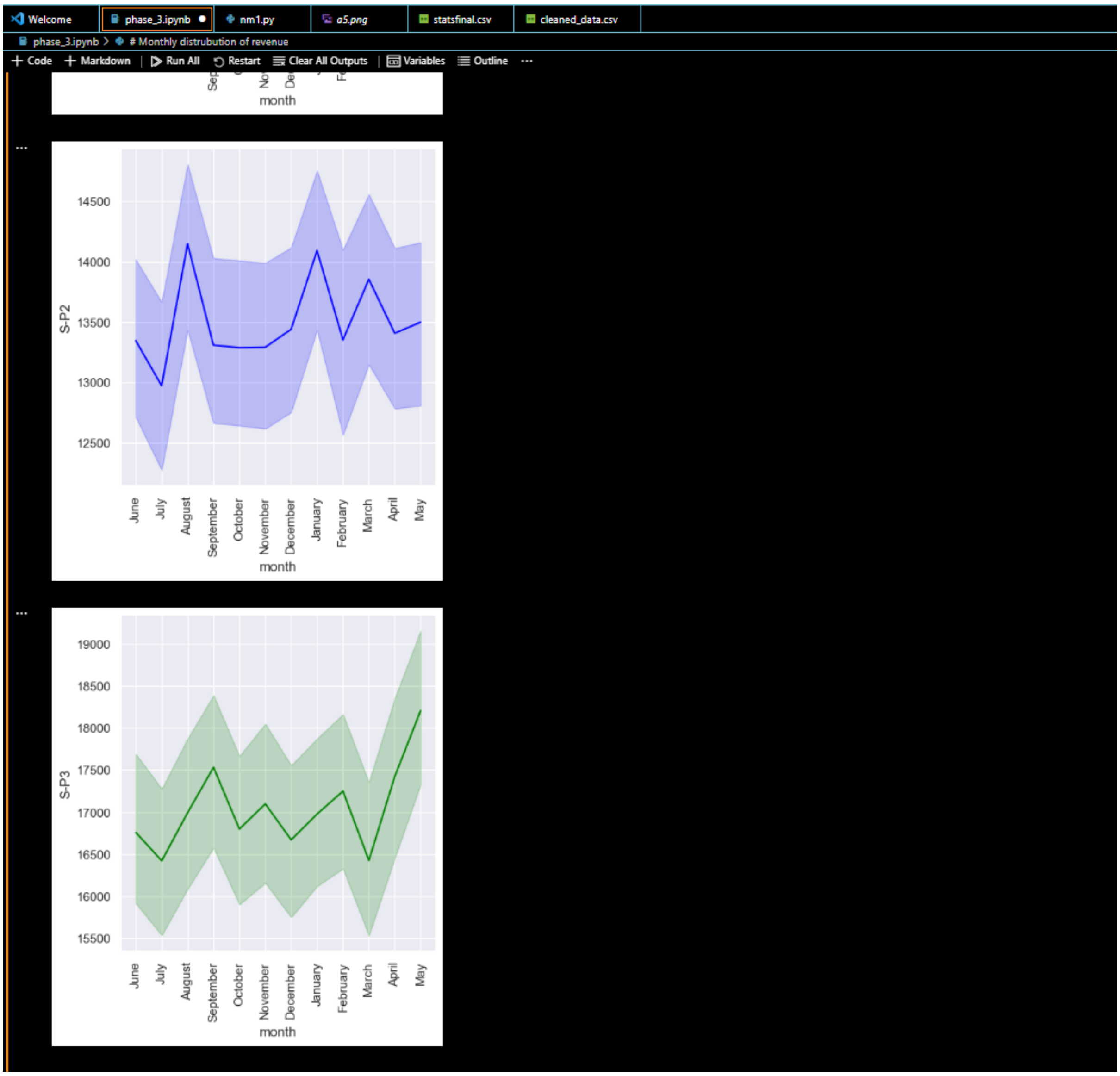
Visualization :

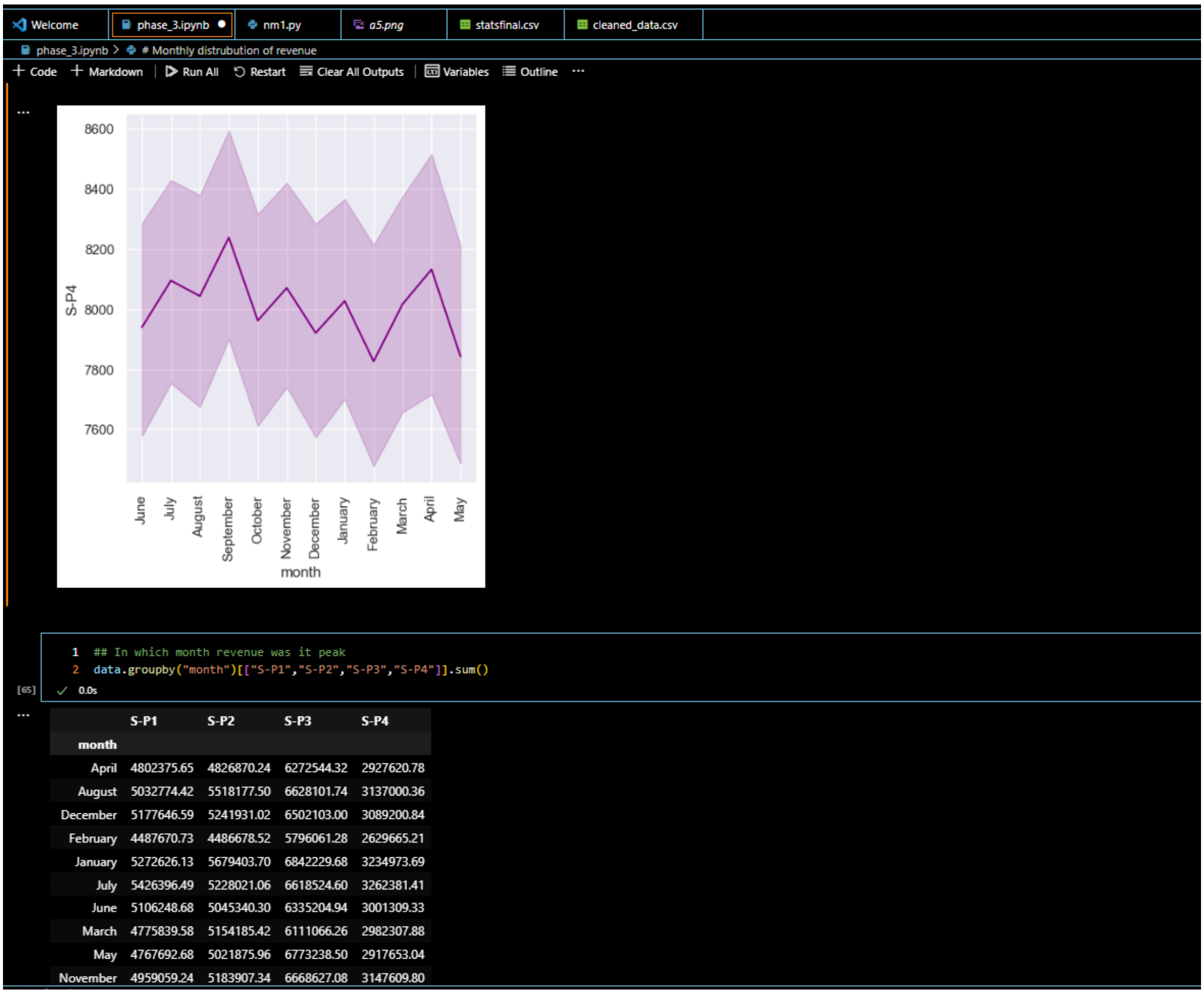
Monthly distribution of revenue



```python
# Monthly distrubution of revenue
data["month"]=data["Date"].dt.month_name()
sns.relplot(x="month",y="S-P1",data=data,kind="line",height=5,color="red")
plt.xticks(rotation=90);
sns.relplot(x="month",y="S-P2",data=data,kind="line",height=5,color="blue")
plt.xticks(rotation=90);
sns.relplot(x="month",y="S-P3",data=data,kind="line",height=5,color="green")
plt.xticks(rotation=90);
sns.relplot(x="month",y="S-P4",data=data,kind="line",height=5,color="purple")
plt.xticks(rotation=90);
```

```
c:\Users\91984\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
c:\Users\91984\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
c:\Users\91984\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
c:\Users\91984\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

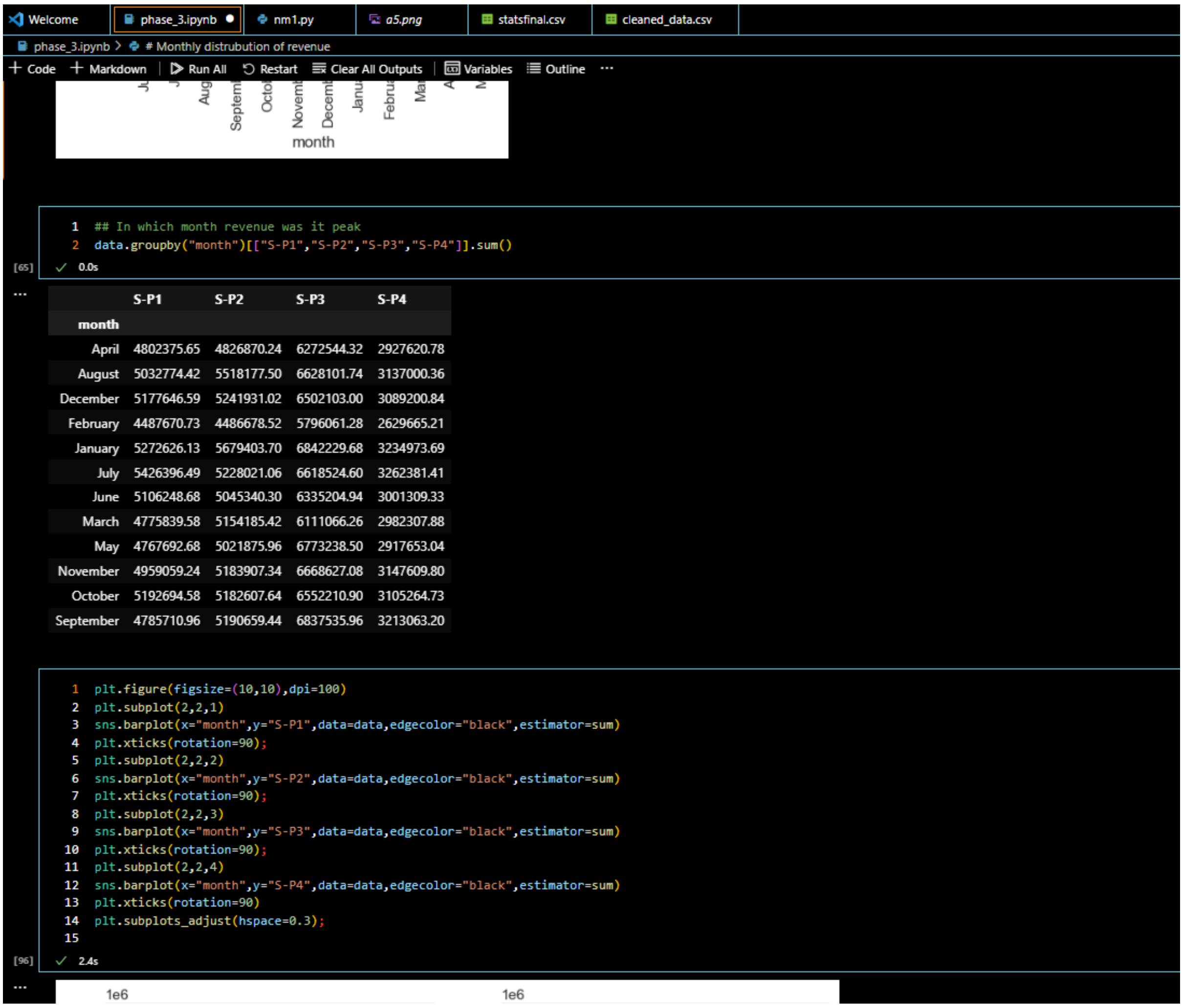

Welcome
phase_3.ipynb
nm1.py
a5.png
statsfinal.csv
cleaned_data.csv
phase_3.ipynb > # Monthly distrubution of revenue
+ Code
+ Markdown
Run All
Restart
Clear All Outputs
Variables
Outline
month
S-P2
14500
14000
13500
13000
12500
June
July
August
September
October
November
December
January
February
March
April
May
month
S-P3
19000
18500
18000
17500
17000
16500
16000
15500
June
July
August
September
October
November
December
January
February
March
April
May
month



```
## In which month revenue was it peak
data.groupby("month")[["S-P1","S-P2","S-P3","S-P4"]].sum()
```

	S-P1	S-P2	S-P3	S-P4
month				
April	4802375.65	4826870.24	6272544.32	2927620.78
August	5032774.42	5518177.50	6628101.74	3137000.36
December	5177646.59	5241931.02	6502103.00	3089200.84
February	4487670.73	4486678.52	5796061.28	2629665.21
January	5272626.13	5679403.70	6842229.68	3234973.69
July	5426396.49	5228021.06	6618524.60	3262381.41
June	5106248.68	5045340.30	6335204.94	3001309.33
March	4775839.58	5154185.42	6111066.26	2982307.88
May	4767692.68	5021875.96	6773238.50	2917653.04
November	4959059.24	5183907.34	6668627.08	3147609.80

Which month revenue was it peak

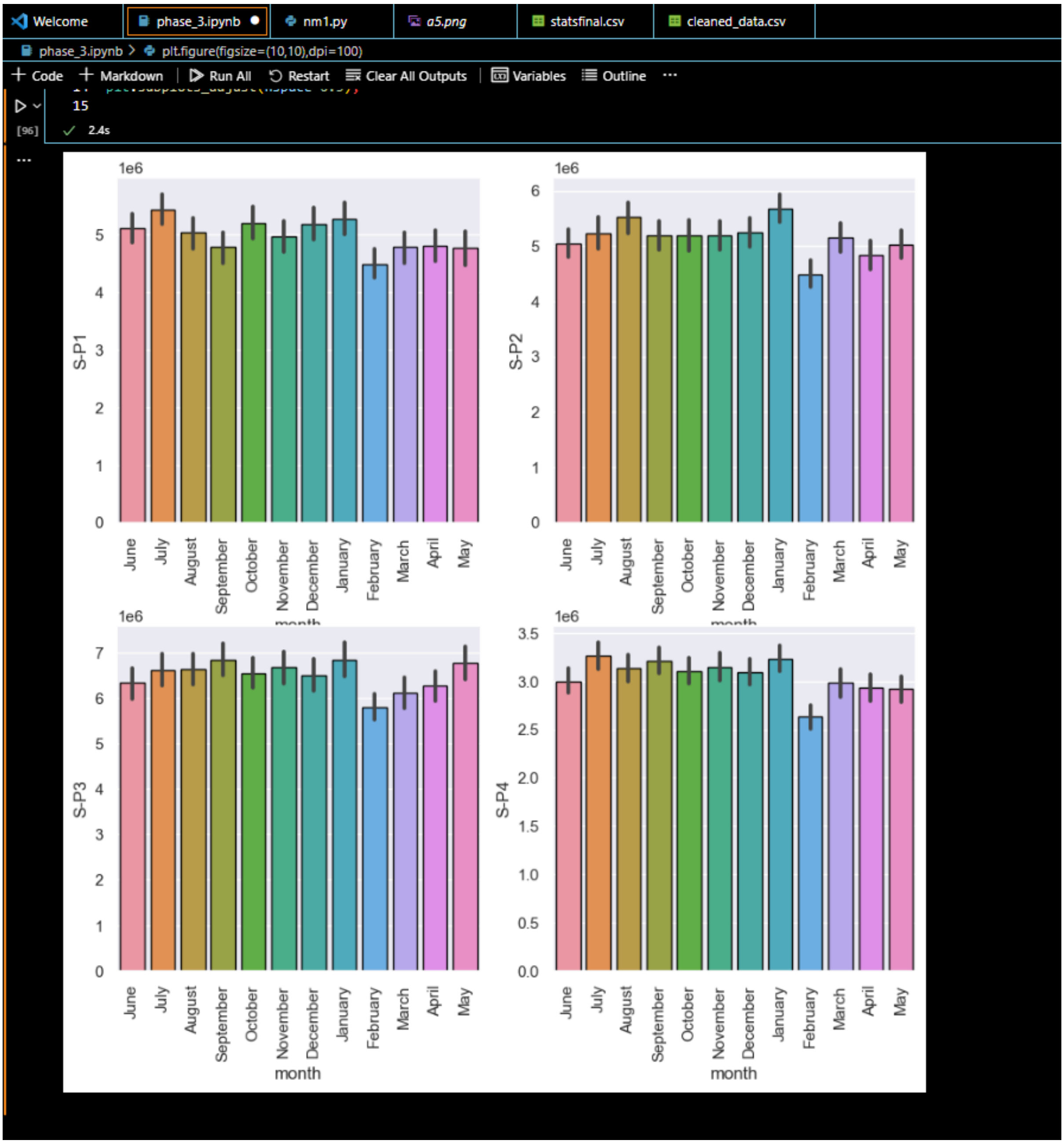


```
## In which month revenue was it peak
data.groupby("month")[["S-P1","S-P2","S-P3","S-P4"]].sum()
```

	S-P1	S-P2	S-P3	S-P4
month				
April	4802375.65	4826870.24	6272544.32	2927620.78
August	5032774.42	5518177.50	6628101.74	3137000.36
December	5177646.59	5241931.02	6502103.00	3089200.84
February	4487670.73	4486678.52	5796061.28	2629665.21
January	5272626.13	5679403.70	6842229.68	3234973.69
July	5426396.49	5228021.06	6618524.60	3262381.41
June	5106248.68	5045340.30	6335204.94	3001309.33
March	4775839.58	5154185.42	6111066.26	2982307.88
May	4767692.68	5021875.96	6773238.50	2917653.04
November	4959059.24	5183907.34	6668627.08	3147609.80
October	5192694.58	5182607.64	6552210.90	3105264.73
September	4785710.96	5190659.44	6837535.96	3213063.20

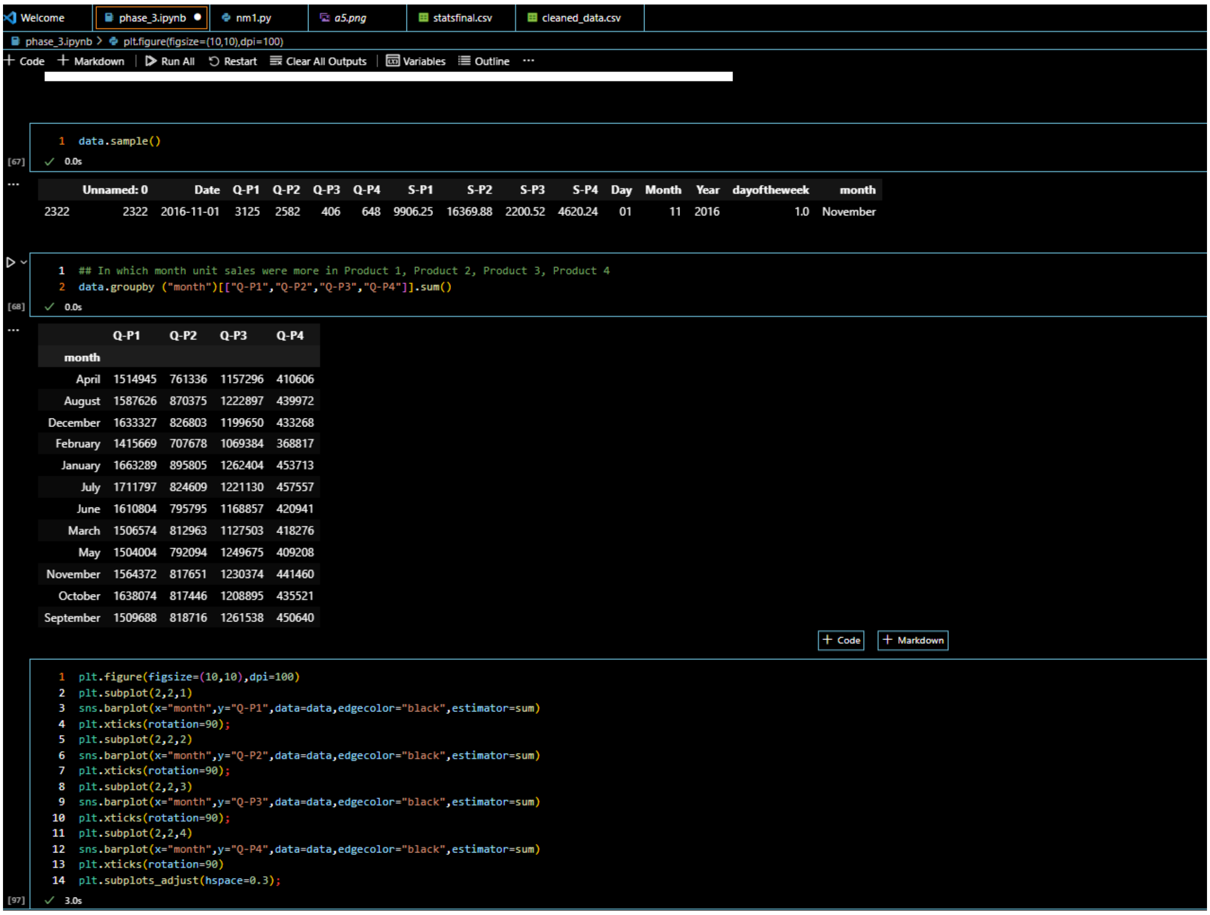
```
plt.figure(figsize=(10,10),dpi=100)
plt.subplot(2,2,1)
sns.barplot(x="month",y="S-P1",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,2)
sns.barplot(x="month",y="S-P2",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,3)
sns.barplot(x="month",y="S-P3",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,4)
sns.barplot(x="month",y="S-P4",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90)
plt.subplots_adjust(hspace=0.3);

```

Dashboards:



Welcome
phase_3.ipynb
nm1.py
a5.png
statsfinal.csv
cleaned_data.csv
phase_3.ipynb > plt.figure(figsize=(10,10),dpi=100)
+ Code
+ Markdown
Run All
Restart
Clear All Outputs
Variables
Outline
[96]
2.4s
1e6
S-P1
S-P2
S-P3
S-P4
0
1
2
3
4
5
6
7
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
June
July
August
September
October
November
December
January
February
March
April
May
month

Which month unit sales were more in product 1,2,3,4



```
data.sample()
```

[67] ✓ 0.0s

	Unnamed: 0	Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4	Day	Month	Year	dayoftheweek	month
2322	2322	2016-11-01	3125	2582	406	648	9906.25	16369.88	2200.52	4620.24	01	11	2016	1.0	November

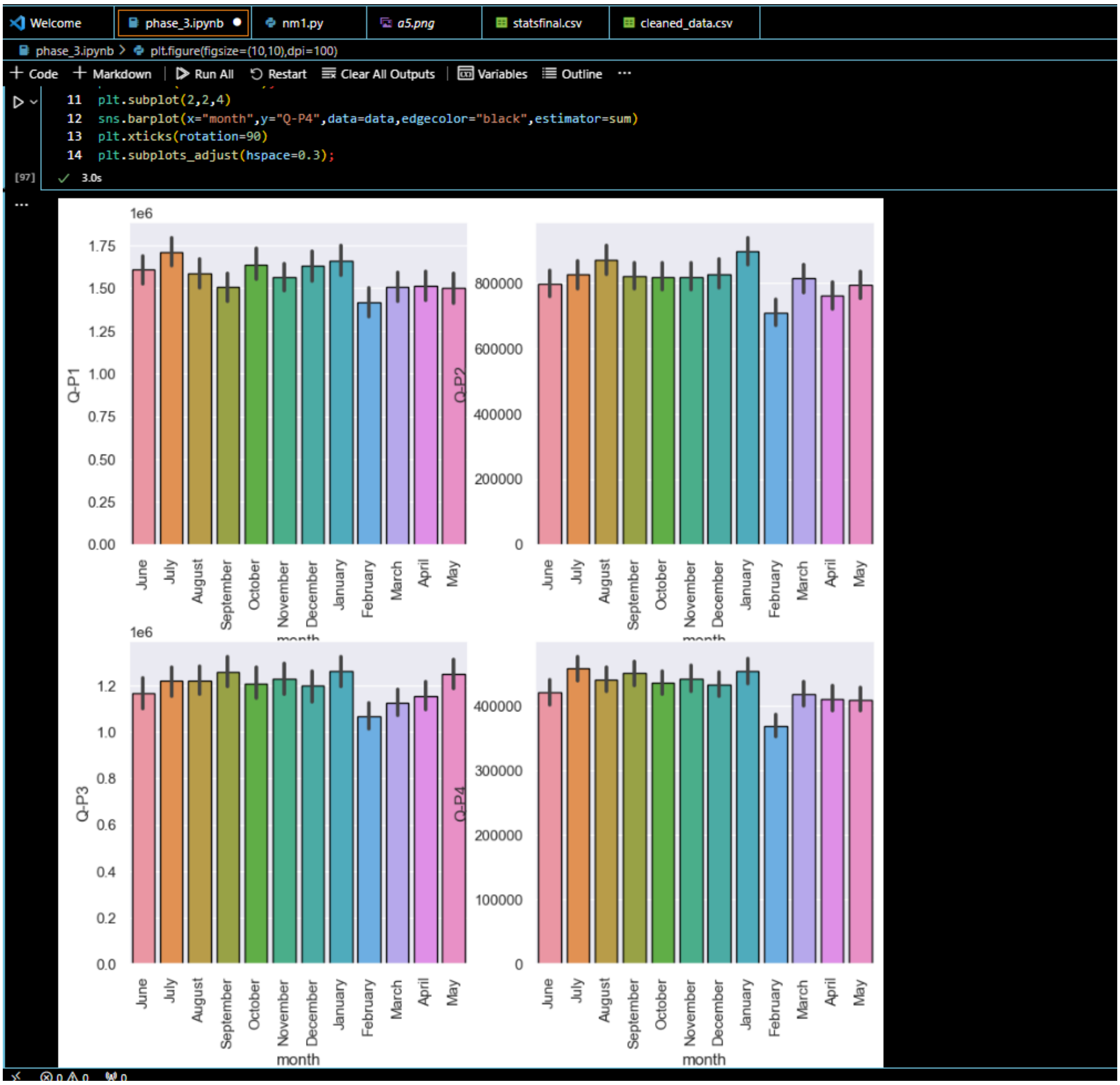
```
## In which month unit sales were more in Product 1, Product 2, Product 3, Product 4
data.groupby ("month")[["Q-P1","Q-P2","Q-P3","Q-P4"]].sum()
```

[68] ✓ 0.0s

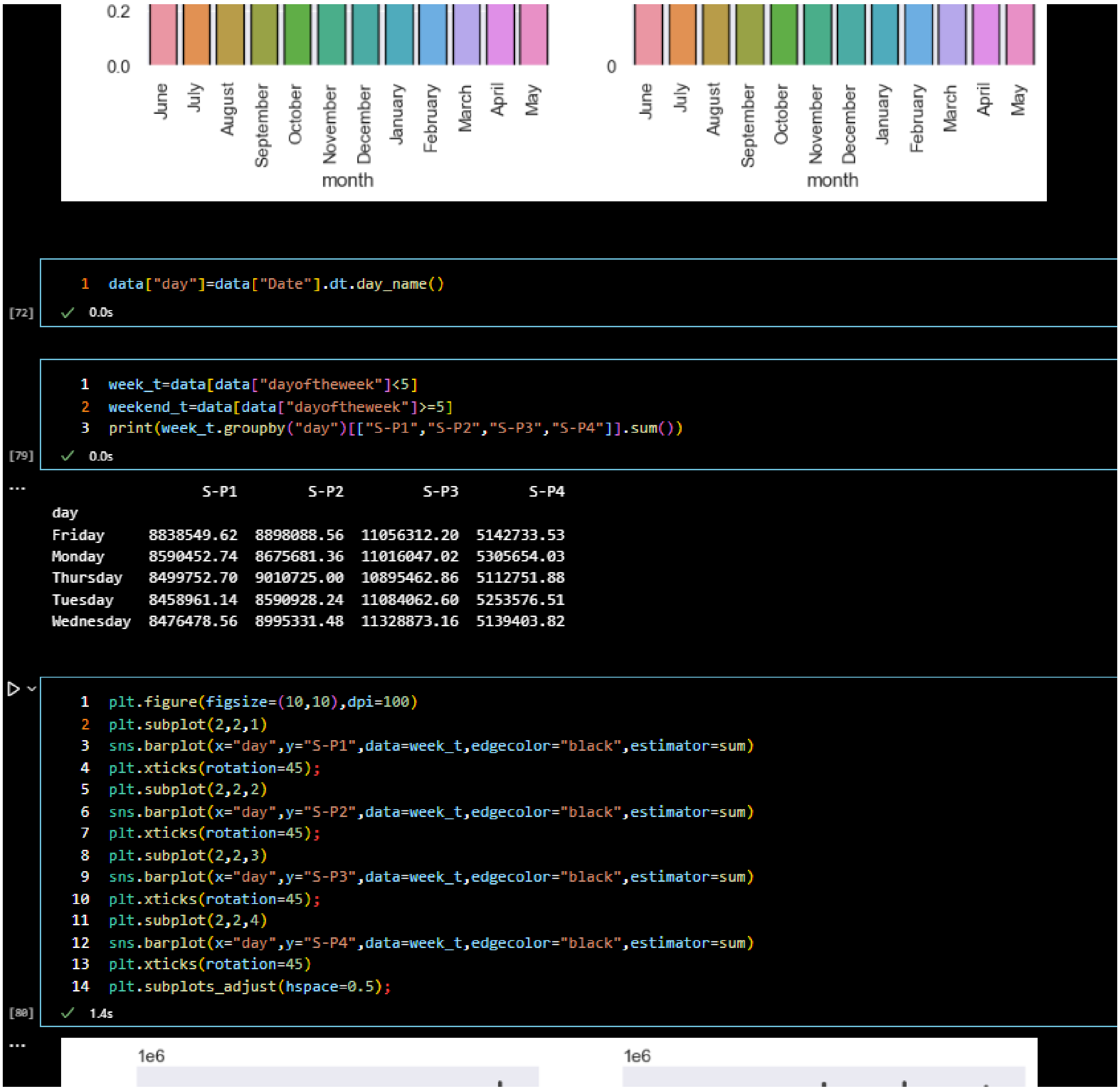
	Q-P1	Q-P2	Q-P3	Q-P4
month				
April	1514945	761336	1157296	410606
August	1587626	870375	1222897	439972
December	1633327	826803	1199650	433268
February	1415669	707678	1069384	368817
January	1663289	895805	1262404	453713
July	1711797	824609	1221130	457557
June	1610804	795795	1168857	420941
March	1506574	812963	1127503	418276
May	1504004	792094	1249675	409208
November	1564372	817651	1230374	441460
October	1638074	817446	1208895	435521
September	1509688	818716	1261538	450640

```
plt.figure(figsize=(10,10),dpi=100)
plt.subplot(2,2,1)
sns.barplot(x="month",y="Q-P1",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,2)
sns.barplot(x="month",y="Q-P2",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,3)
sns.barplot(x="month",y="Q-P3",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,4)
sns.barplot(x="month",y="Q-P4",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90)
plt.subplots_adjust(hspace=0.3);
```

[97] ✓ 3.0s



```
plt.subplot(2,2,4)
sns.barplot(x="month",y="Q-P4",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90)
plt.subplots_adjust(hspace=0.3);
```

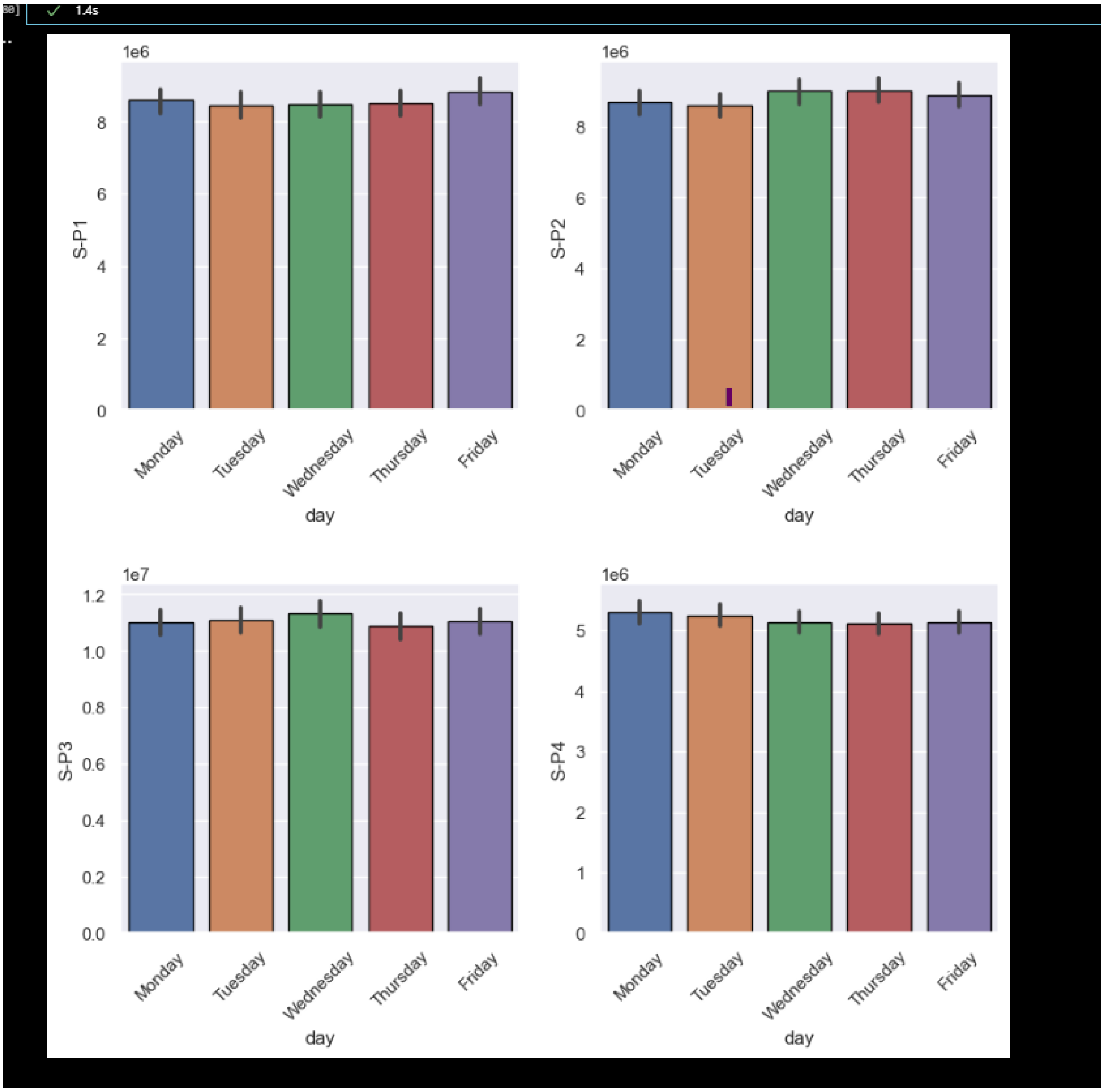


```
data["day"]=data["Date"].dt.day_name()
```

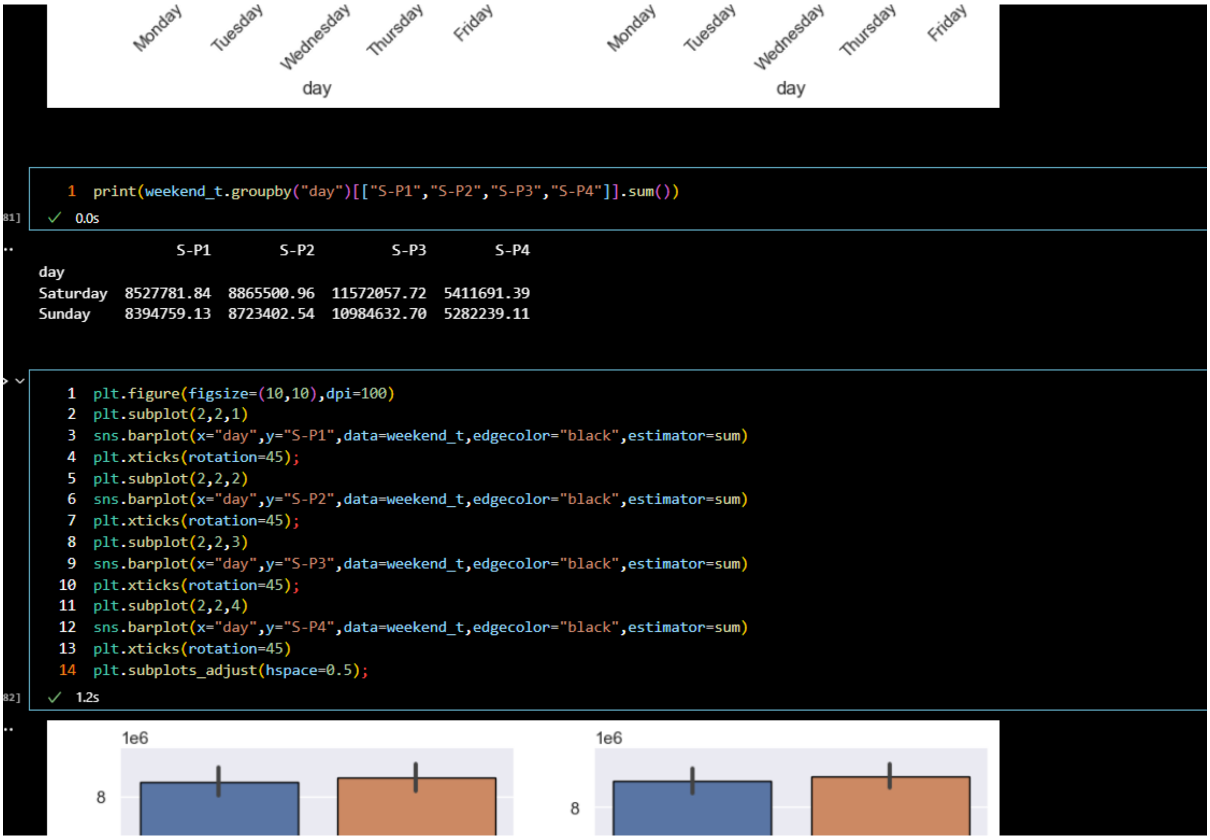
```
week_t=data[data["dayoftheweek"]<5]
weekend_t=data[data["dayoftheweek"]>=5]
print(week_t.groupby("day")[["S-P1","S-P2","S-P3","S-P4"]].sum())
```

```
                S-P1        S-P2         S-P3        S-P4
day
Friday     8838549.62  8898088.56  11056312.20  5142733.53
Monday     8590452.74  8675681.36  11016047.02  5305654.03
Thursday   8499752.70  9010725.00  10895462.86  5112751.88
Tuesday    8458961.14  8590928.24  11084062.60  5253576.51
Wednesday  8476478.56  8995331.48  11328873.16  5139403.82
```

```
plt.figure(figsize=(10,10),dpi=100)
plt.subplot(2,2,1)
sns.barplot(x="day",y="S-P1",data=week_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,2)
sns.barplot(x="day",y="S-P2",data=week_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,3)
sns.barplot(x="day",y="S-P3",data=week_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,4)
sns.barplot(x="day",y="S-P4",data=week_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45)
plt.subplots_adjust(hspace=0.5);
```



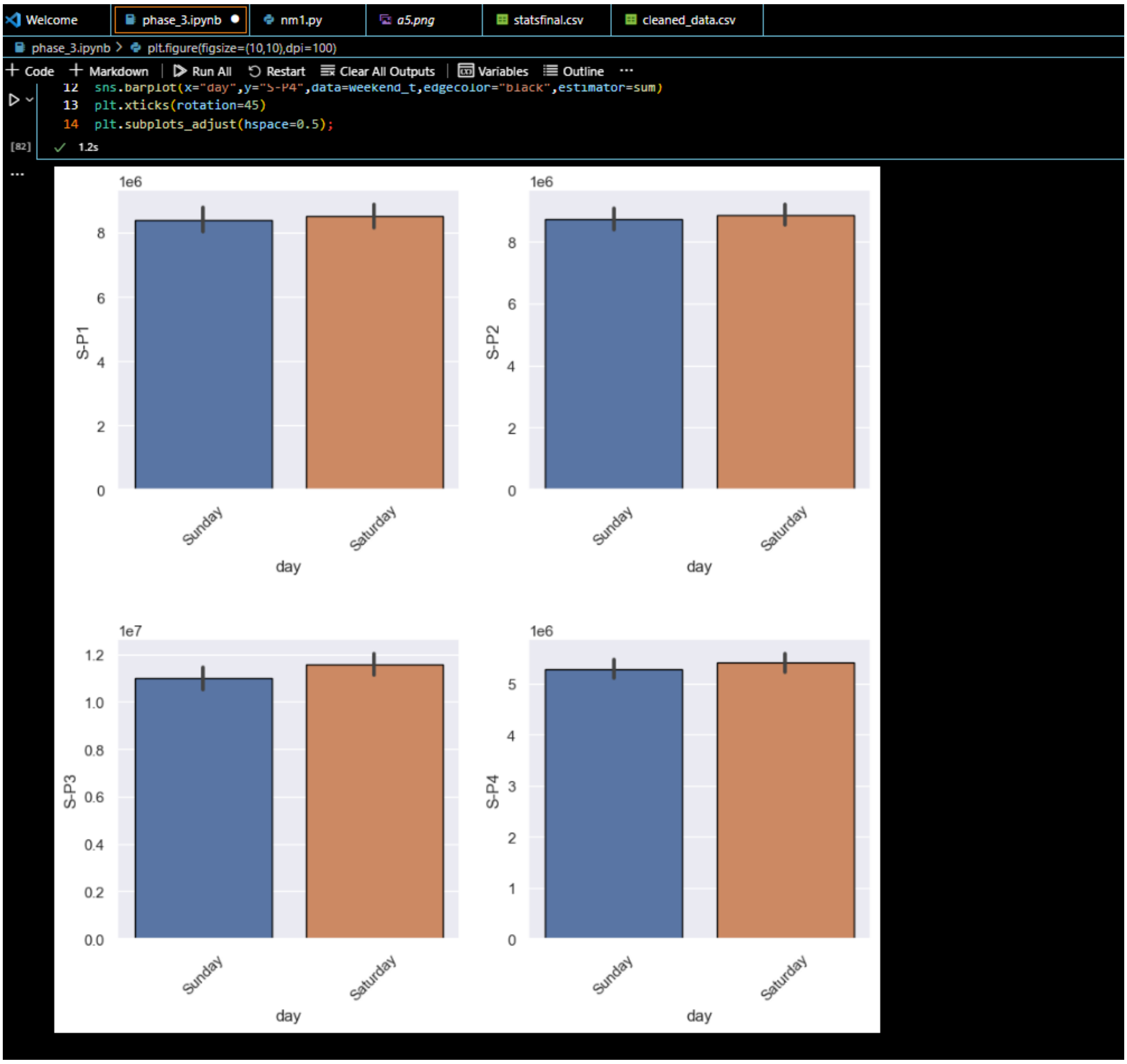
88]
✓ 1.4s
1e6
8
6
4
2
0
S-P1
Monday
Tuesday
Wednesday
Thursday
Friday
day
1e6
8
6
4
2
0
S-P2
Monday
Tuesday
Wednesday
Thursday
Friday
day
1e7
1.2
1.0
0.8
0.6
0.4
0.2
0.0
S-P3
Monday
Tuesday
Wednesday
Thursday
Friday
day
1e6
5
4
3
2
1
0
S-P4
Monday
Tuesday
Wednesday
Thursday
Friday
day



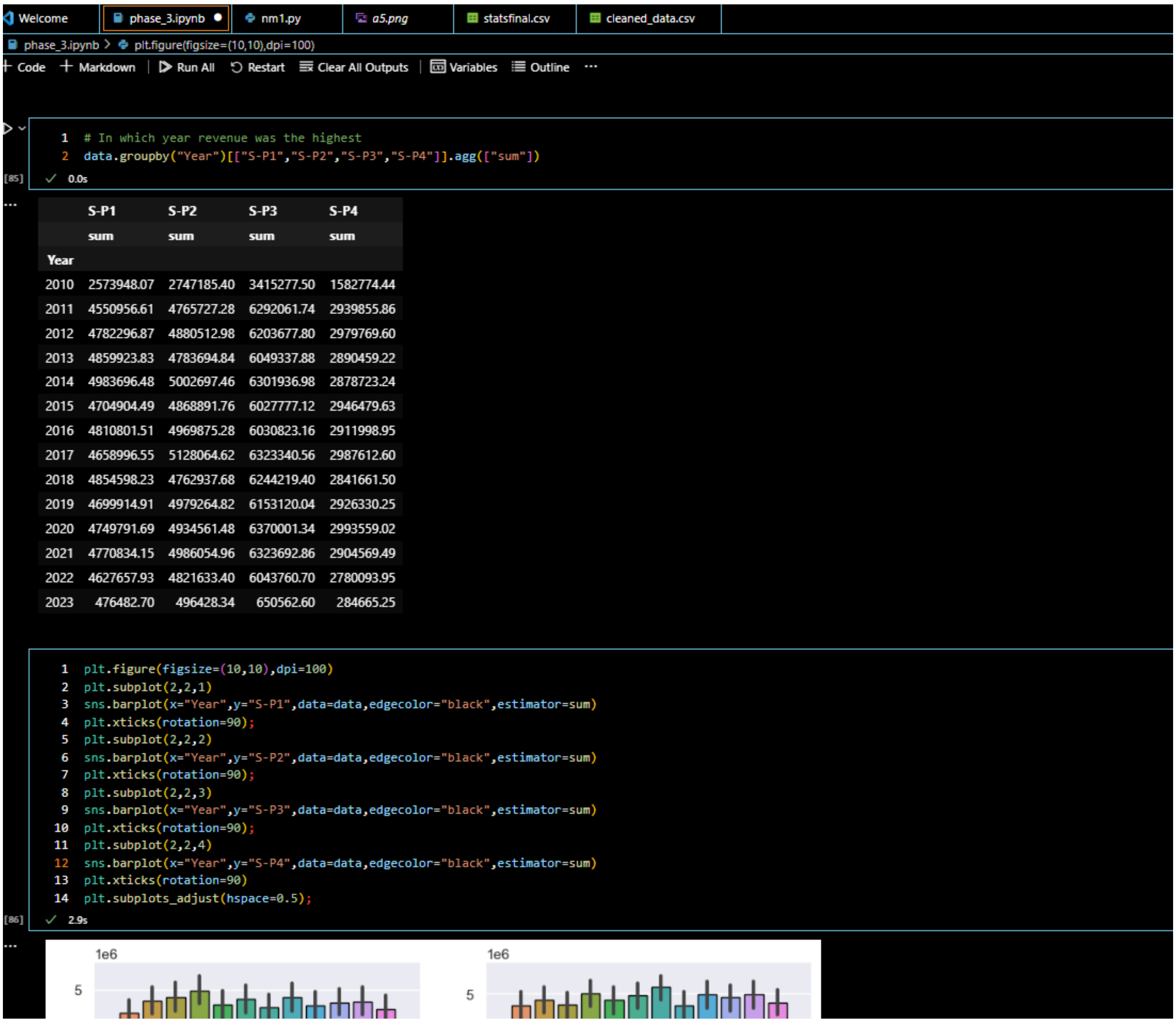
```python
print(weekend_t.groupby("day")[["S-P1","S-P2","S-P3","S-P4"]].sum())
```

```
                S-P1        S-P2         S-P3        S-P4
day
Saturday  8527781.84  8865500.96  11572057.72  5411691.39
Sunday    8394759.13  8723402.54  10984632.70  5282239.11
```

```python
plt.figure(figsize=(10,10),dpi=100)
plt.subplot(2,2,1)
sns.barplot(x="day",y="S-P1",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,2)
sns.barplot(x="day",y="S-P2",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,3)
sns.barplot(x="day",y="S-P3",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,4)
sns.barplot(x="day",y="S-P4",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45)
plt.subplots_adjust(hspace=0.5);
```



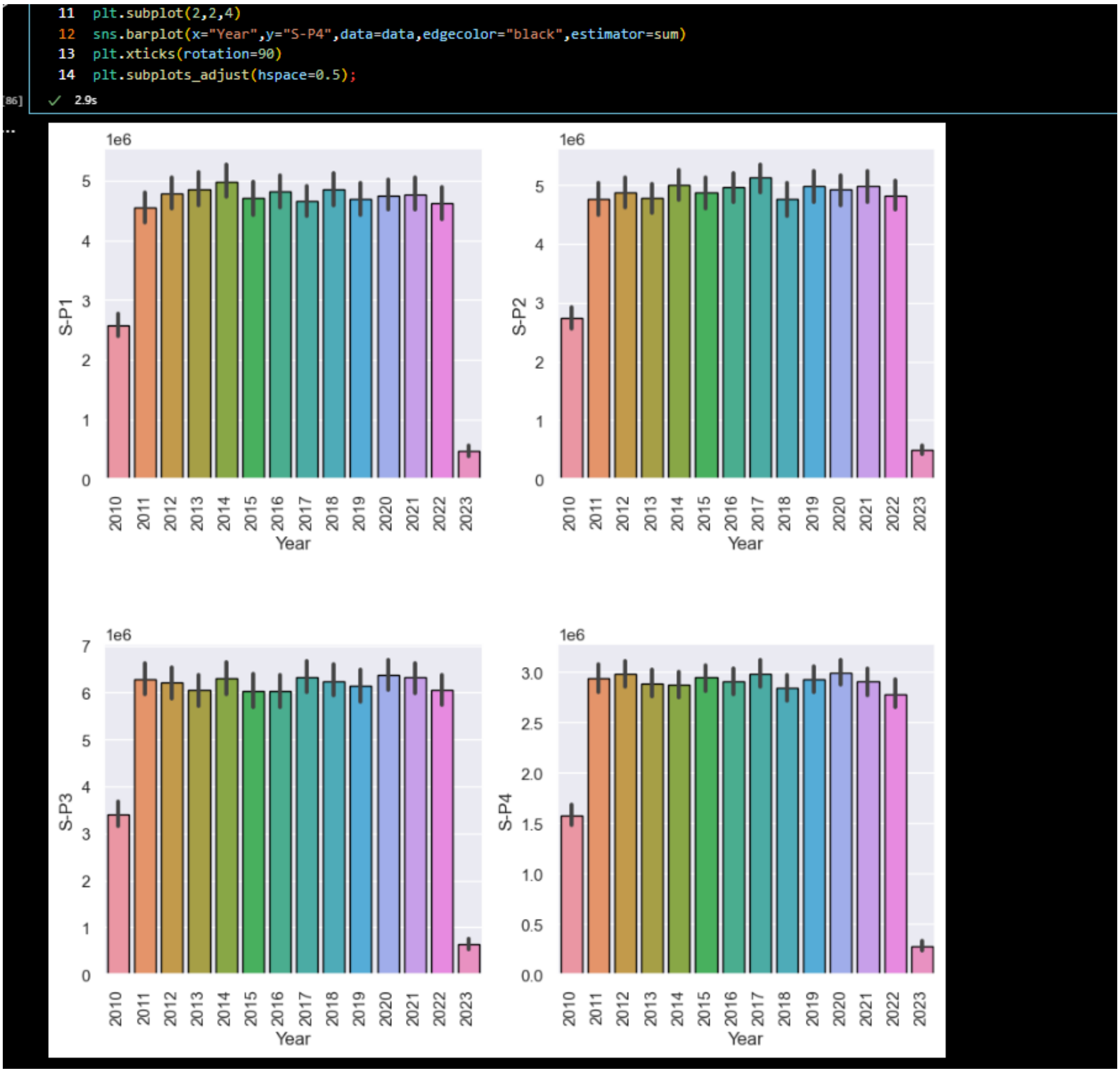
```
sns.barplot(x="day",y="S-P4",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45)
plt.subplots_adjust(hspace=0.5);
```



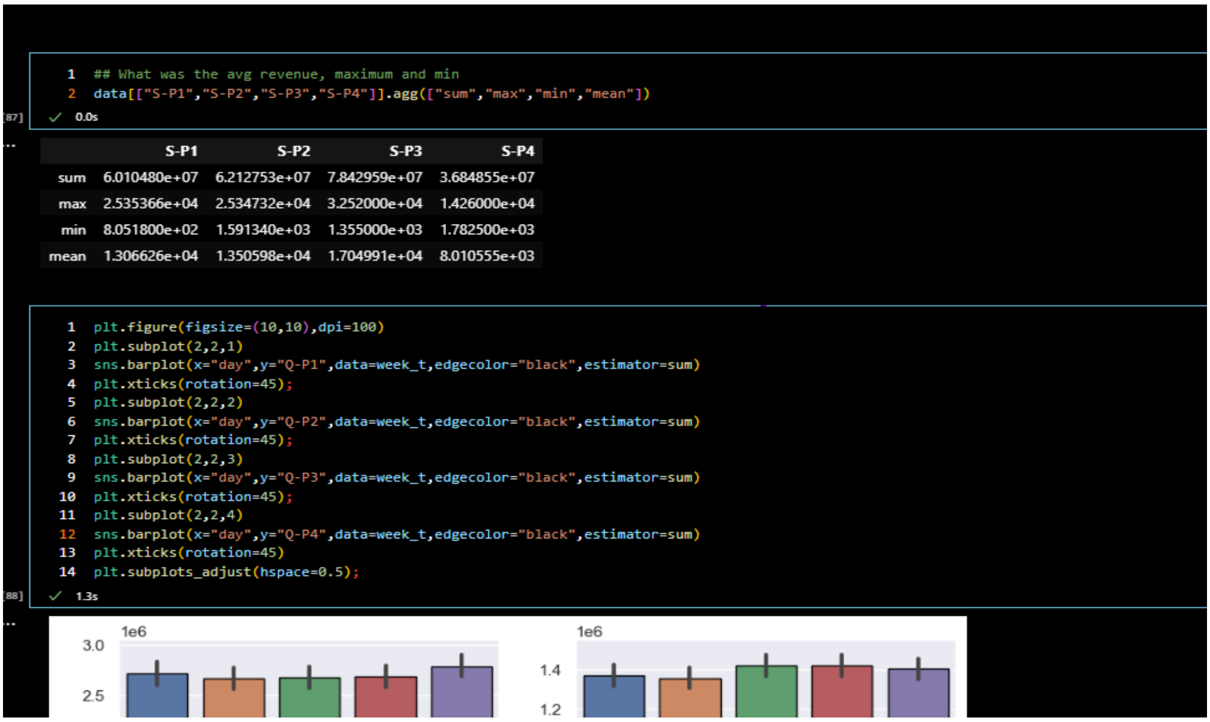
```
# In which year revenue was the highest
data.groupby("Year")[["S-P1","S-P2","S-P3","S-P4"]].agg(["sum"])
```

	S-P1	S-P2	S-P3	S-P4
	sum	sum	sum	sum
Year				
2010	2573948.07	2747185.40	3415277.50	1582774.44
2011	4550956.61	4765727.28	6292061.74	2939855.86
2012	4782296.87	4880512.98	6203677.80	2979769.60
2013	4859923.83	4783694.84	6049337.88	2890459.22
2014	4983696.48	5002697.46	6301936.98	2878723.24
2015	4704904.49	4868891.76	6027777.12	2946479.63
2016	4810801.51	4969875.28	6030823.16	2911998.95
2017	4658996.55	5128064.62	6323340.56	2987612.60
2018	4854598.23	4762937.68	6244219.40	2841661.50
2019	4699914.91	4979264.82	6153120.04	2926330.25
2020	4749791.69	4934561.48	6370001.34	2993559.02
2021	4770834.15	4986054.96	6323692.86	2904569.49
2022	4627657.93	4821633.40	6043760.70	2780093.95
2023	476482.70	496428.34	650562.60	284665.25

```
plt.figure(figsize=(10,10),dpi=100)
plt.subplot(2,2,1)
sns.barplot(x="Year",y="S-P1",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,2)
sns.barplot(x="Year",y="S-P2",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,3)
sns.barplot(x="Year",y="S-P3",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90);
plt.subplot(2,2,4)
sns.barplot(x="Year",y="S-P4",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90)
plt.subplots_adjust(hspace=0.5);
```



```
plt.subplot(2,2,4)
sns.barplot(x="Year",y="S-P4",data=data,edgecolor="black",estimator=sum)
plt.xticks(rotation=90)
plt.subplots_adjust(hspace=0.5);
```



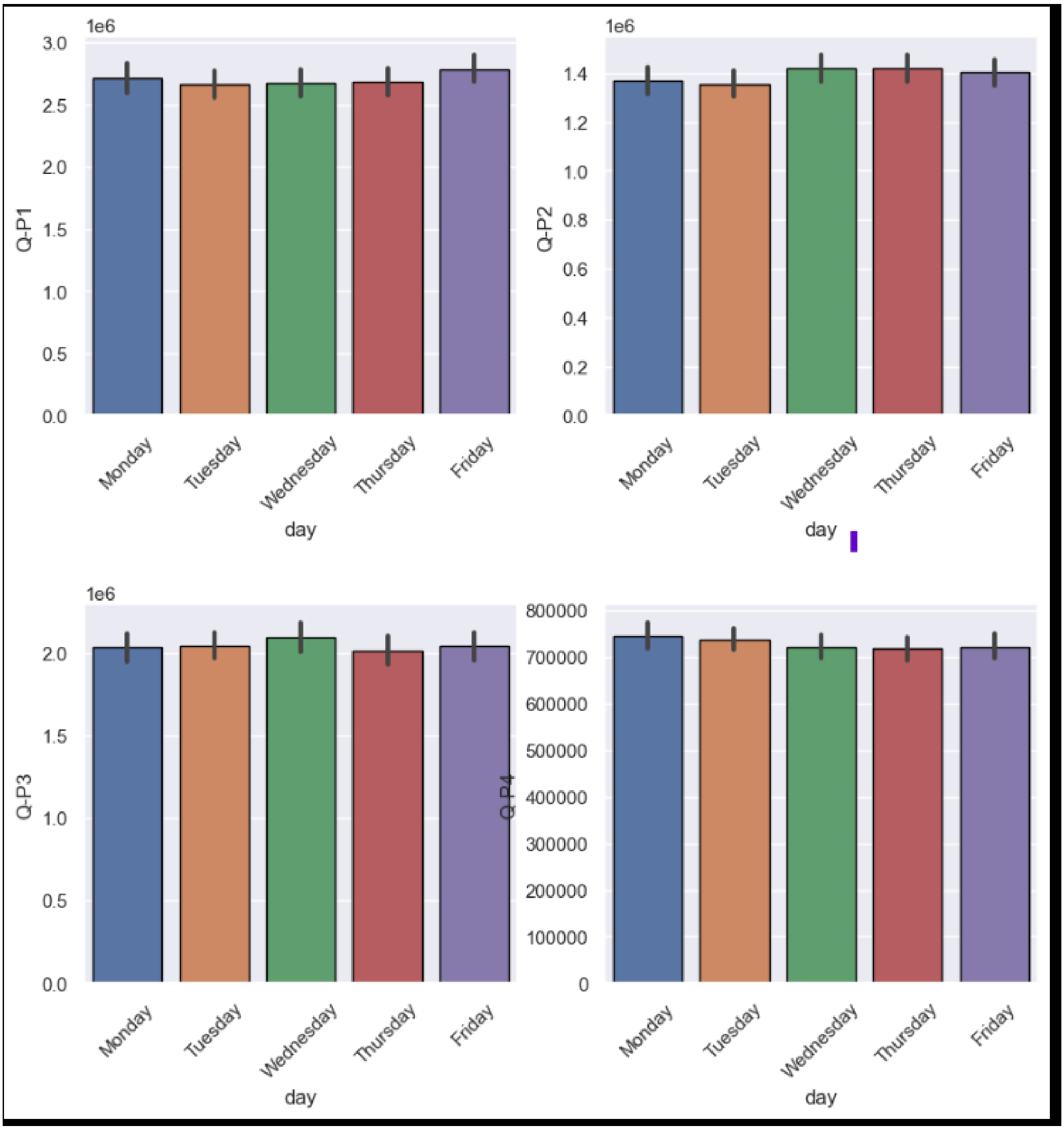
```
## What was the avg revenue, maximum and min
data[["S-P1","S-P2","S-P3","S-P4"]].agg(["sum","max","min","mean"])
```

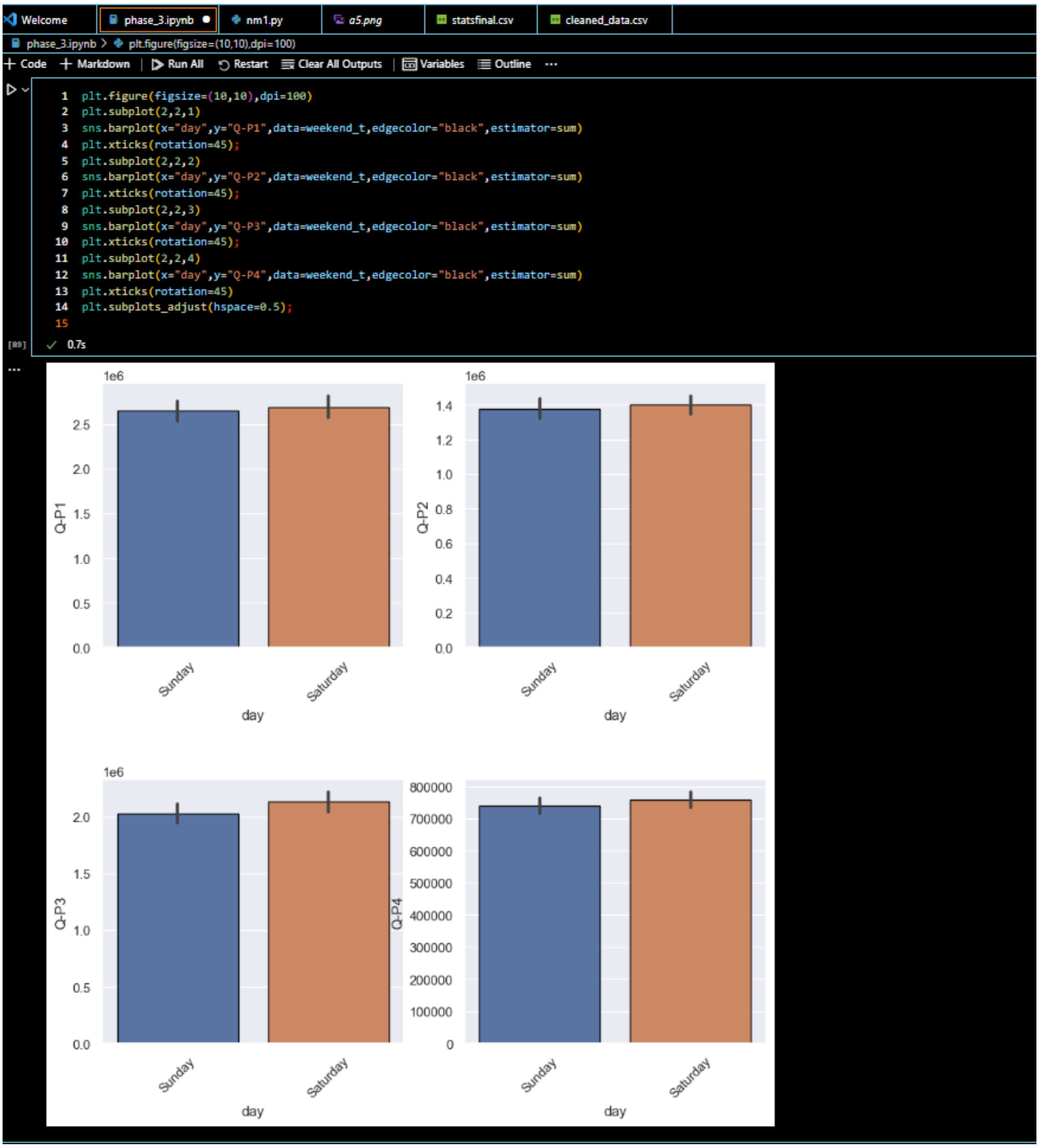
[87] ✓ 0.0s

	S-P1	S-P2	S-P3	S-P4
sum	6.010480e+07	6.212753e+07	7.842959e+07	3.684855e+07
max	2.535366e+04	2.534732e+04	3.252000e+04	1.426000e+04
min	8.051800e+02	1.591340e+03	1.355000e+03	1.782500e+03
mean	1.306626e+04	1.350598e+04	1.704991e+04	8.010555e+03

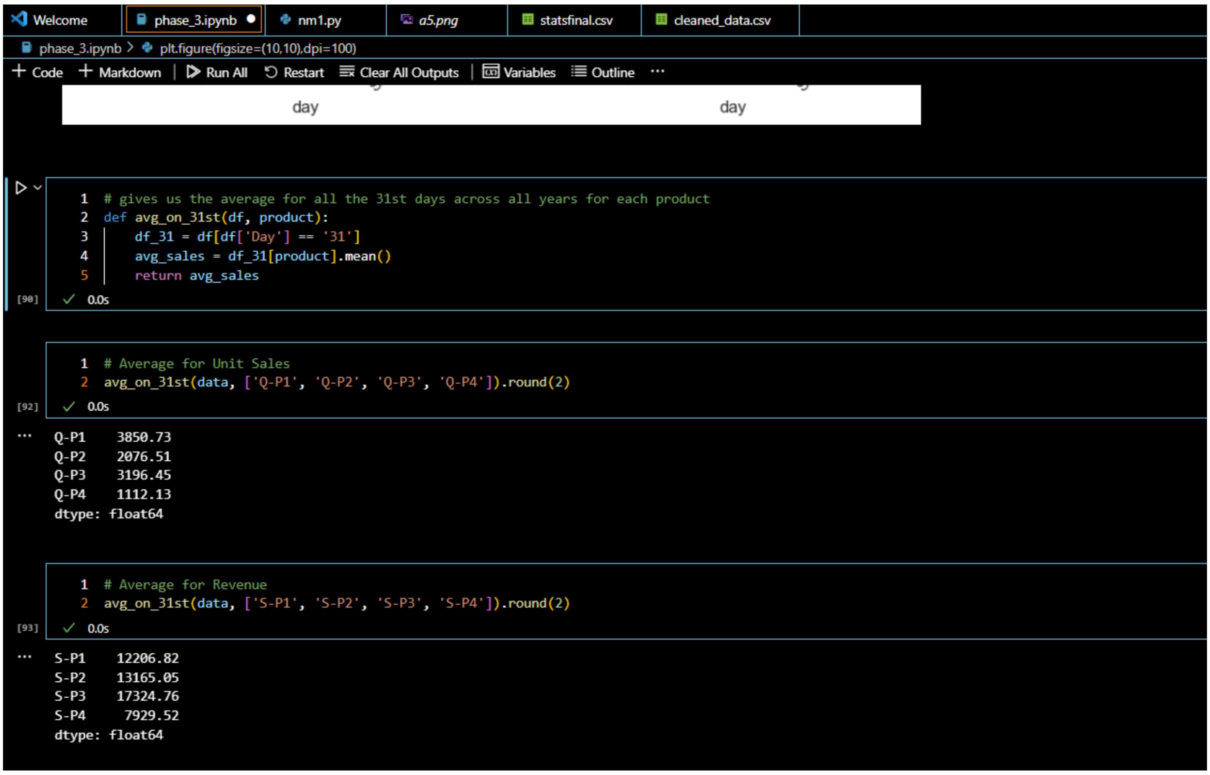
```
plt.figure(figsize=(10,10),dpi=100)
plt.subplot(2,2,1)
sns.barplot(x="day",y="Q-P1",data=week_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,2)
sns.barplot(x="day",y="Q-P2",data=week_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,3)
sns.barplot(x="day",y="Q-P3",data=week_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,4)
sns.barplot(x="day",y="Q-P4",data=week_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45)
plt.subplots_adjust(hspace=0.5);
```

[88] ✓ 1.3s


1e6
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Q-P1
Monday
Tuesday
Wednesday
Thursday
Friday
day
1e6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Q-P2
Monday
Tuesday
Wednesday
Thursday
Friday
day
1e6
2.0
1.5
1.0
0.5
0.0
Q-P3
Monday
Tuesday
Wednesday
Thursday
Friday
day
800000
700000
600000
500000
400000
300000
200000
100000
0
Q-P4
Monday
Tuesday
Wednesday
Thursday
Friday
day



```python
plt.figure(figsize=(10,10),dpi=100)
plt.subplot(2,2,1)
sns.barplot(x="day",y="Q-P1",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,2)
sns.barplot(x="day",y="Q-P2",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,3)
sns.barplot(x="day",y="Q-P3",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45);
plt.subplot(2,2,4)
sns.barplot(x="day",y="Q-P4",data=weekend_t,edgecolor="black",estimator=sum)
plt.xticks(rotation=45)
plt.subplots_adjust(hspace=0.5);
```



```python
# gives us the average for all the 31st days across all years for each product
def avg_on_31st(df, product):
    df_31 = df[df['Day'] == '31']
    avg_sales = df_31[product].mean()
    return avg_sales
```

```python
# Average for Unit Sales
avg_on_31st(data, ['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']).round(2)
```

```
Q-P1    3850.73
Q-P2    2076.51
Q-P3    3196.45
Q-P4    1112.13
dtype: float64
```

```python
# Average for Revenue
avg_on_31st(data, ['S-P1', 'S-P2', 'S-P3', 'S-P4']).round(2)
```

```
S-P1    12206.82
S-P2    13165.05
S-P3    17324.76
S-P4     7929.52
dtype: float64
```

Observation

- We can see that our previous observation correlate as Q-P1 has the higest estimate, follwed by Q-P3
- We can approxiamte that the company will make:
  - Q-P1: 3813.74
  - Q-P2: 2058.80
  - Q-P3: 3183.88
  - Q-P4: 1098.61

Linkcode

Conclusion

Unit Sales 2011 - 2022

- P1 has the highest unit sales for each year. And it's highest is in year 2014.
- We can observe that P4 has the lowest unit sales of all the products.

Revenues 2011 - 2022

- We can observe that P3 brought in the most revenue. This could be as a result of multiple things:
  - P3 was sold for higher than the rest, as it had the second highest unit sales for each year.
- We can observe than P1 and P2 brought in similar revenues for each year. With P2 bringing in slightly more.

- P1 despite having the most unit sold, brought in the second lowest revenue each year.

Average Month Sales 2011 - 2022

- We can observe that all Products unit sales drop in Feb.
- We can observe that Feb and Dec have the lowest sales for each product
- For P1 We can observe Mar - Jul having the highest unit sales
- For P2 We can observe Jan, Mar - Aug having the highest unit sales
- For P3 We can observe May & Sep having the highest unit sales
- For P4 We can observe uniform sales from Jan - Dec

Estimated Unit Sales for 31st of Dec

This value can not be properly estimated with out Machine Learning. Currently we used the average for all the 31st days across all years for each product.

- Overall we can see that P1 has the highest unit sales on the 31st for each year, except for 2021 and 2022. (These could be as a result to Covid and other economy issues.)
- P3 has the second highest unit sales for all the 31st in each year.
- We can see that our previous observation correlate as Q-P1 has the higest estimate, followed by Q-P3
- We can approxiamte that the company will make:
    - Q-P1: 3850.73
    - Q-P2: 2076.51
    - Q-P3: 3196.45
    - Q-P4: 1112.13

Insights

- Added columns month, day and day of the week and changing the dtype of date from object to datetime64 through feature engineering.
- Drop columns unnamed as it was not providing any usefull information.
- S-P3 has gained the most revenue but the unit sale of Q-P1 is more.
- In 2016 most revenue most revenue generated and on fridays and staurdays most revenue generated.
- On Weekdays and weekend the S-P3 has the highest revenue whereas on weekend and weekday the Q-P1 has more unit sales.
- In month of October unit sale and revenue was at peak.