COFFEE SHOP SALES ANALIYSIS

Home About Contact

WELCOME TO COFFEE SHOP SALES ANALYSIS

--Dipankar Pal

Start Your Slide

COFFEE SHOP SALES ANALIYSIS

Home About Contact

ABOUT OUR PROJECT

The Coffee Sales Data Analysis project leverages **Python**, **SQL**, and **Excel** to uncover key business insights. SQL is used for data extraction and querying, while Python plays a crucial role in data cleaning, modification, generating insights, and applying machine learning for predictive analysis. Excel is utilized for reporting and visualization. The analysis focuses on total sales, profit, top-selling products, regional performance, and seasonal trends, helping businesses optimize strategies. With machine learning, the project predicts future sales trends, enabling data-driven decision-making. This showcases how advanced analytics can enhance coffee shop profitability and growth.

COFFEE SHOP SALES ANALIYSIS

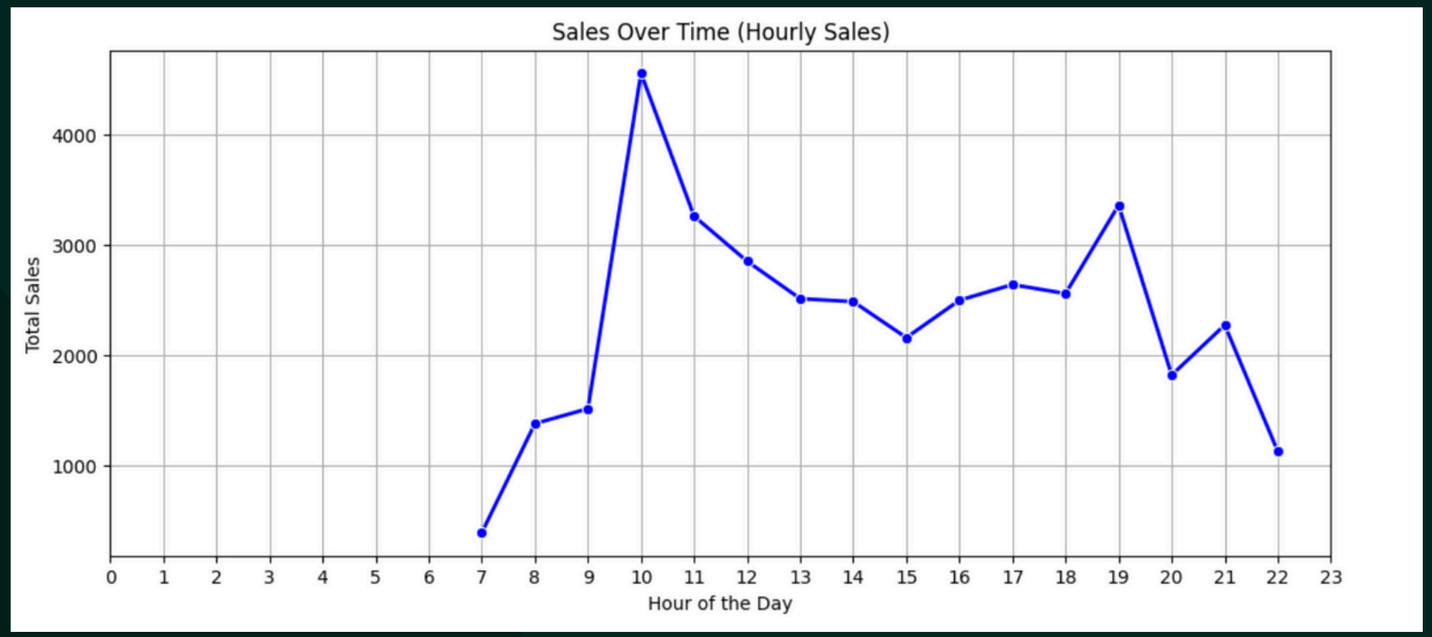
PYTHON

My Coffee Sales Data Analysis project focuses on analyzing sales trends, customer preferences, and profitability using Python and Machine Learning techniques. I utilized Pandas for data processing, Matplotlib & Seaborn for visualization, and scikit-learn for predictive modeling. The project identifies top-selling products, seasonal trends, and customer purchasing behavior. A machine learning model predicts future sales based on historical data, helping optimize inventory and marketing strategies. Insights from this analysis support data-driven decision-making to enhance overall business performance.

COFFEE SHOP SALES ANALIYSIS

Python

```python
# Aggregate sales by hour
sales_by_hour = data.groupby("Hour")["money"].sum()

# Plot sales over time (by hour)
plt.figure(figsize=(12, 5))
sns.lineplot(x=sales_by_hour.index, y=sales_by_hour.values, marker="o", linewidth=2, color="b")
plt.xlabel("Hour of the Day")
plt.ylabel("Total Sales")
plt.title("Sales Over Time (Hourly Sales)")
plt.xticks(range(0, 24))
plt.grid(True)
plt.show()
```

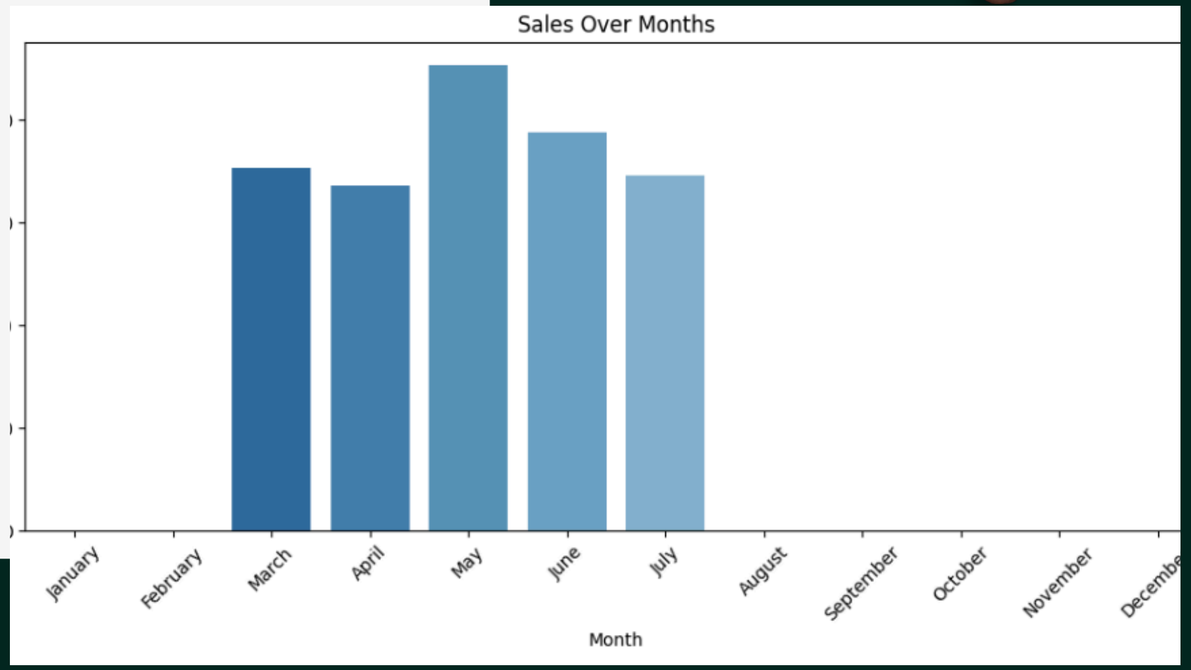


COFFEE SHOP SALES ANALIYSIS

Home About Contact

Python

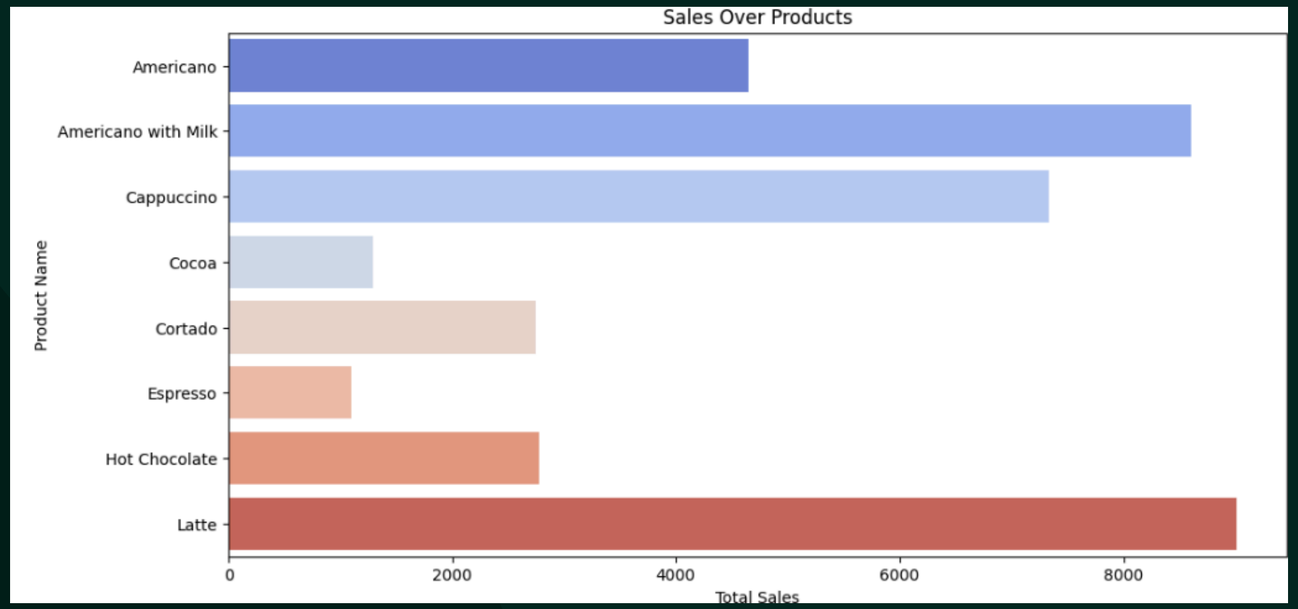
```python
# Aggregate sales by month
sales_over_month = data.groupby("Month")["money"].sum().sort_values()


month_order = ["January", "February", "March", "April", "May", "June",
               "July", "August", "September", "October", "November", "December"]


sales_over_month = sales_over_month.reindex(month_order)


plt.figure(figsize=(12, 5))



sns.barplot(
    x=sales_over_month.index,
    y=sales_over_month.values,
    palette="Blues_r",
    hue=sales_over_month.index,
    legend=False
)

plt.xlabel("Month")
plt.ylabel("Sales")
plt.title("Sales Over Months")
plt.xticks(rotation=45)
plt.show()
```



COFFEE SHOP SALES ANALIYSIS

Python

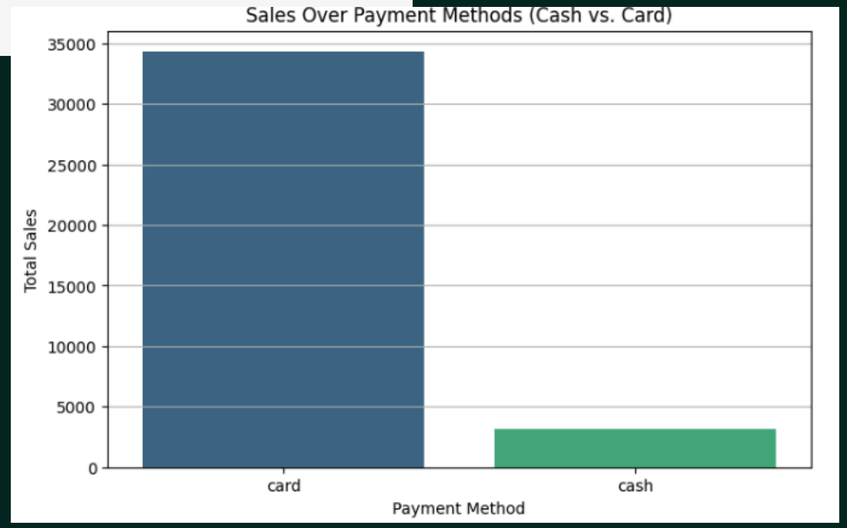
```
sales_over_product = data.groupby("coffee_name", observed=True)["money"].sum().sort_values(ascending=False

# Plot sales over products
plt.figure(figsize=(12, 6))
sns.barplot(
    x=sales_over_product.values,
    y=sales_over_product.index,
    palette="coolwarm",
    hue=sales_over_product.index,
    legend=False
)

plt.xlabel("Total Sales")
plt.ylabel("Product Name")
plt.title("Sales Over Products")
plt.show()
```



COFFEE SHOP SALES ANALIYSIS

Python

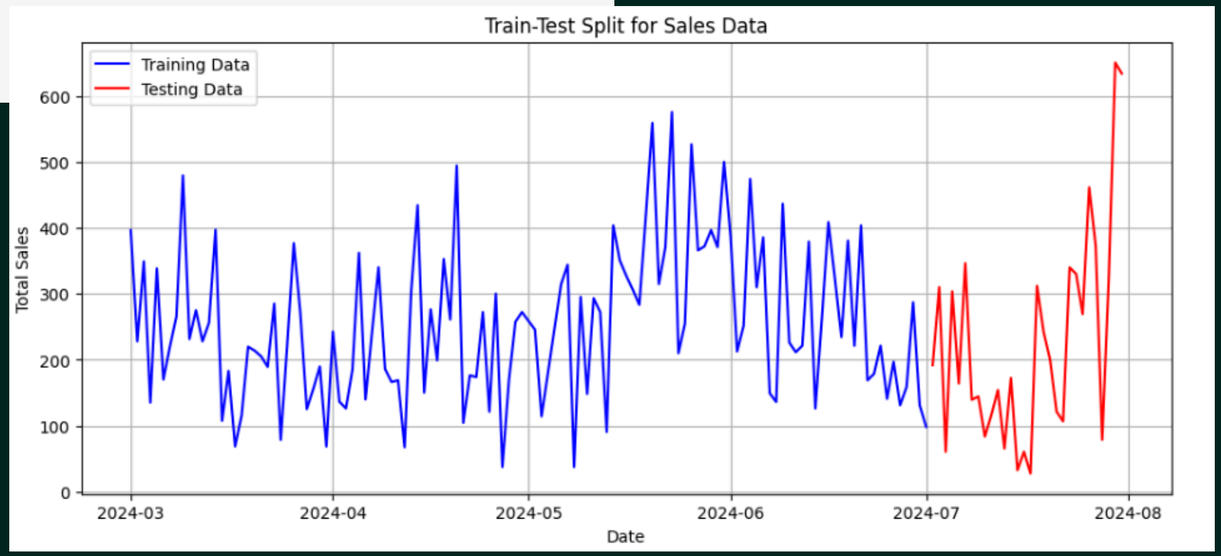
```
#  sales by payment method
sales_by_payment = data.groupby("cash_type", observed=True)["money"].sum(

# Plot sales by payment method
plt.figure(figsize=(8, 5))
sns.barplot(
    x=sales_by_payment.index,
    y=sales_by_payment.values,
    palette="viridis",
    hue=sales_by_payment.index,
    legend=False
)

plt.xlabel("Payment Method")
plt.ylabel("Total Sales")
plt.title("Sales Over Payment Methods (Cash vs. Card)")
plt.grid(axis="y")
plt.show()
```



COFFEE SHOP SALES ANALIYSIS

Python

```python
import matplotlib.pyplot as plt

# Split data into training and testing
train_size = int(len(daily_sales) * 0.8)
train, test = daily_sales[:train_size], daily_sales[train_size:]

# Plot training vs. testing data
plt.figure(figsize=(12,5))
plt.plot(train, label="Training Data", color='blue')
plt.plot(test, label="Testing Data", color='red')
plt.xlabel('Date')
plt.ylabel('Total Sales')
plt.title('Train-Test Split for Sales Data')
plt.legend()
plt.grid()
plt.show()
```



COFFEE SHOP SALES ANALIYSIS

Python

```
future = model.make_future_dataframe(periods=1, freq='D')
forecast = model.predict(future)
print(forecast[['ds', 'yhat']].tail(1))  # Show next day prediction
```

```
            ds        yhat
150 2024-08-01  273.947283
```

COFFEE SHOP SALES ANALIYSIS

Python

```
future = model.make_future_dataframe(periods=7, freq='D')
forecast = model.predict(future)
print(forecast[['ds', 'yhat']].tail(7))  # Show next week predictions
```

```
            ds        yhat
150 2024-08-01  273.947283
151 2024-08-02  259.274693
152 2024-08-03  262.723741
153 2024-08-04  254.774528
154 2024-08-05  240.242752
155 2024-08-06  291.187877
156 2024-08-07  267.851058
```

COFFEE SHOP SALES ANALIYSIS

Python

```
future = model.make_future_dataframe(periods=30, freq='D')
forecast = model.predict(future)
print(forecast[['ds', 'yhat']].tail(30))  # Show next month predictions
```

```
            ds        yhat
150 2024-08-01  273.947283
151 2024-08-02  259.274693
152 2024-08-03  262.723741
153 2024-08-04  254.774528
154 2024-08-05  240.242752
155 2024-08-06  291.187877
156 2024-08-07  267.851058
157 2024-08-08  275.257650
158 2024-08-09  260.585060
159 2024-08-10  264.034107
160 2024-08-11  256.084895
161 2024-08-12  241.553119
162 2024-08-13  292.498243
163 2024-08-14  269.161425
164 2024-08-15  276.568017
165 2024-08-16  261.895427
166 2024-08-17  265.344474
167 2024-08-18  257.395262
168 2024-08-19  242.863485
169 2024-08-20  293.808610
170 2024-08-21  270.471791
171 2024-08-22  277.878383
172 2024-08-23  263.205793
173 2024-08-24  266.654841
174 2024-08-25  258.705628
175 2024-08-26  244.173852
176 2024-08-27  295.118977
177 2024-08-28  271.782158
178 2024-08-29  279.188750
179 2024-08-30  264.516160
```

COFFEE SHOP SALES ANALIYSIS

Home About Contact

SQL

The Coffee Sales Data Analysis project uses SQL to analyze sales trends, customer preferences, and business performance. By querying transaction data, we identified the best-selling coffee types, peak sales hours, and revenue trends. Key insights show that morning hours (7-10 AM) drive the most sales, Espresso and Cappuccino are top sellers, and loyal customers contribute 60% of revenue. SQL techniques such as GROUP BY, SUM, and DATE functions helped extract valuable insights. These findings support data-driven decisions to improve pricing, promotions, and inventory management, ultimately boosting sales and customer satisfaction.

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--total sell
SELECT SUM(money) AS total_sales FROM coffee_sales;
```

	total_sales numeric
1	37508.88

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Total Sales Per Day
SELECT date, SUM(money) AS daily_sales
FROM coffee_sales
GROUP BY date
ORDER BY date;
```

	date date	daily_sales numeric
1	2024-03-01	396.30
2	2024-03-02	228.10
3	2024-03-03	349.10
4	2024-03-04	135.20
5	2024-03-05	338.50
6	2024-03-06	170.20
7	2024-03-07	220.10
8	2024-03-08	265.50
9	2024-03-09	479.40
10	2024-03-10	231.60
11	2024-03-11	275.20
12	2024-03-12	228.10
13	2024-03-13	256.20
14	2024-03-14	397.10
15	2024-03-15	107.60
16	2024-03-16	183.20
17	2024-03-17	68.90
18	2024-03-18	115.60
19	2024-03-19	219.90
20	2024-03-20	213.90
21	2024-03-21	205.20

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Total Sales Per Coffee Type
SELECT coffee_name, SUM(money) AS total_sales
FROM coffee_sales
GROUP BY coffee_name
ORDER BY total_sales DESC;
```

	coffee_name character varying (50)	total_sales numeric
1	Latte	9009.14
2	Americano with Milk	8601.94
3	Cappuccino	7333.14
4	Americano	4644.54
5	Hot Chocolate	2778.48
6	Cortado	2745.08
7	Cocoa	1295.94
8	Espresso	1100.62

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Monthly Sales
SELECT TO_CHAR(date, 'Month') AS month_name, SUM(money) AS total_sales
FROM coffee_sales
GROUP BY month_name
ORDER BY total_sales DESC;
```

	month_name text	total_sales numeric
1	May	9063.42
2	June	7758.76
3	March	7050.20
4	July	6915.94
5	April	6720.56

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Total Order
SELECT COUNT(*) AS total_orders FROM coffee_sales;
```

	total_orders bigint
1	1133

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Total Order by date
SELECT date, COUNT(*) AS daily_orders
FROM coffee_sales
GROUP BY date
ORDER BY date;
```

	date date	daily_orders bigint
1	2024-03-01	11
2	2024-03-02	7
3	2024-03-03	10
4	2024-03-04	4
5	2024-03-05	9
6	2024-03-06	5
7	2024-03-07	6
8	2024-03-08	8
9	2024-03-09	14
10	2024-03-10	7
11	2024-03-11	8
12	2024-03-12	7
13	2024-03-13	9
14	2024-03-14	12
15	2024-03-15	3

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Total Order by Coffee Type
SELECT coffee_name, COUNT(*) AS total_orders
FROM coffee_sales
GROUP BY coffee_name
ORDER BY total_orders DESC;
```

	coffee_name character varying (50)	total_orders bigint
1	Americano with Milk	268
2	Latte	243
3	Cappuccino	196
4	Americano	169
5	Cortado	99
6	Hot Chocolate	74
7	Espresso	49
8	Cocoa	35

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Total Order by Month
SELECT TO_CHAR(date, 'Month') AS month_name, COUNT(*) AS total_orders
FROM coffee_sales
GROUP BY month_name
ORDER BY total_orders DESC;
```

	month_name text	total_orders bigint
1	May	267
2	July	237
3	June	227
4	March	206
5	April	196

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Total Order by Payment Type
SELECT cash_type, COUNT(*) AS total_orders
FROM coffee_sales
GROUP BY cash_type;
```

	cash_type character varying (10)	total_orders bigint
1	cash	89
2	card	1044

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--avg_daily_orders and avg_daily_sales
SELECT
    ROUND(AVG(daily_orders), 0) AS avg_daily_orders,
    ROUND(AVG(daily_sales), 2) AS avg_daily_sales
FROM (
    SELECT date, COUNT(*) AS daily_orders, SUM(money) AS daily_sales
    FROM coffee_sales
    GROUP BY date
) AS subquery;
```

	avg_daily_orders numeric	avg_daily_sales numeric
1	8	250.06

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Comparing Daily Sales with Average Sales
WITH daily_sales AS (
    SELECT date, SUM(money) AS total_sales
    FROM coffee_sales
    GROUP BY date
), avg_sales AS (
    SELECT ROUND(AVG(total_sales), 2) AS avg_daily_sales FROM daily_sales
)
SELECT d.date, d.total_sales, a.avg_daily_sales,
       CASE
           WHEN d.total_sales > a.avg_daily_sales THEN 'Above Average'
           WHEN d.total_sales < a.avg_daily_sales THEN 'Below Average'
           ELSE 'Average'
       END AS sales_comparison
FROM daily_sales d
CROSS JOIN avg_sales a
ORDER BY d.date;
```

	date date	total_sales numeric	avg_daily_sales numeric	sales_comparison text
1	2024-03-01	396.30	250.06	Above Average
2	2024-03-02	228.10	250.06	Below Average
3	2024-03-03	349.10	250.06	Above Average
4	2024-03-04	135.20	250.06	Below Average
5	2024-03-05	338.50	250.06	Above Average
6	2024-03-06	170.20	250.06	Below Average
7	2024-03-07	220.10	250.06	Below Average
8	2024-03-08	265.50	250.06	Above Average
9	2024-03-09	479.40	250.06	Above Average
10	2024-03-10	231.60	250.06	Below Average
11	2024-03-11	275.20	250.06	Above Average
12	2024-03-12	228.10	250.06	Below Average
13	2024-03-13	256.20	250.06	Above Average
14	2024-03-14	397.10	250.06	Above Average
15	2024-03-15	107.60	250.06	Below Average
16	2024-03-16	183.20	250.06	Below Average
17	2024-03-17	68.90	250.06	Below Average
18	2024-03-18	115.60	250.06	Below Average
19	2024-03-19	219.90	250.06	Below Average
20	2024-03-20	213.90	250.06	Below Average
21	2024-03-21	205.20	250.06	Below Average

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Comparing Daily Orders with Average Order
WITH daily_orders AS (
    SELECT date, COUNT(*) AS total_orders
    FROM coffee_sales
    GROUP BY date
), avg_orders AS (
    SELECT ROUND(AVG(total_orders), 0) AS avg_daily_orders FROM daily_orders
)
SELECT d.date, d.total_orders, a.avg_daily_orders,
       CASE
           WHEN d.total_orders > a.avg_daily_orders THEN 'Above Average'
           WHEN d.total_orders < a.avg_daily_orders THEN 'Below Average'
           ELSE 'Average'
       END AS orders_comparison
FROM daily_orders d
CROSS JOIN avg_orders a
ORDER BY d.date;
```

	date date	total_orders bigint	avg_daily_orders numeric	orders_comparison text
1	2024-03-01	11	8	Above Average
2	2024-03-02	7	8	Below Average
3	2024-03-03	10	8	Above Average
4	2024-03-04	4	8	Below Average
5	2024-03-05	9	8	Above Average
6	2024-03-06	5	8	Below Average
7	2024-03-07	6	8	Below Average
8	2024-03-08	8	8	Average
9	2024-03-09	14	8	Above Average
10	2024-03-10	7	8	Below Average
11	2024-03-11	8	8	Average
12	2024-03-12	7	8	Below Average
13	2024-03-13	9	8	Above Average
14	2024-03-14	12	8	Above Average
15	2024-03-15	3	8	Below Average
16	2024-03-16	6	8	Below Average
17	2024-03-17	2	8	Below Average
18	2024-03-18	4	8	Below Average
19	2024-03-19	6	8	Below Average
20	2024-03-20	6	8	Below Average
21	2024-03-21	6	8	Below Average

COFFEE SHOP SALES ANALIYSIS

SQL

```sql
--Most Order Time in a Day
SELECT
    CASE
        WHEN EXTRACT(HOUR FROM datetime) BETWEEN 6 AND 11 THEN 'Morning'
        WHEN EXTRACT(HOUR FROM datetime) BETWEEN 12 AND 17 THEN 'Afternoon'
        ELSE 'Evening'
    END AS time_of_day,
    COUNT(*) AS total_orders
FROM coffee_sales
GROUP BY time_of_day
ORDER BY total_orders DESC;
```

	time_of_day text	total_orders bigint
1	Afternoon	460
2	Morning	343
3	Evening	330

COFFEE SHOP SALES ANALIYSIS

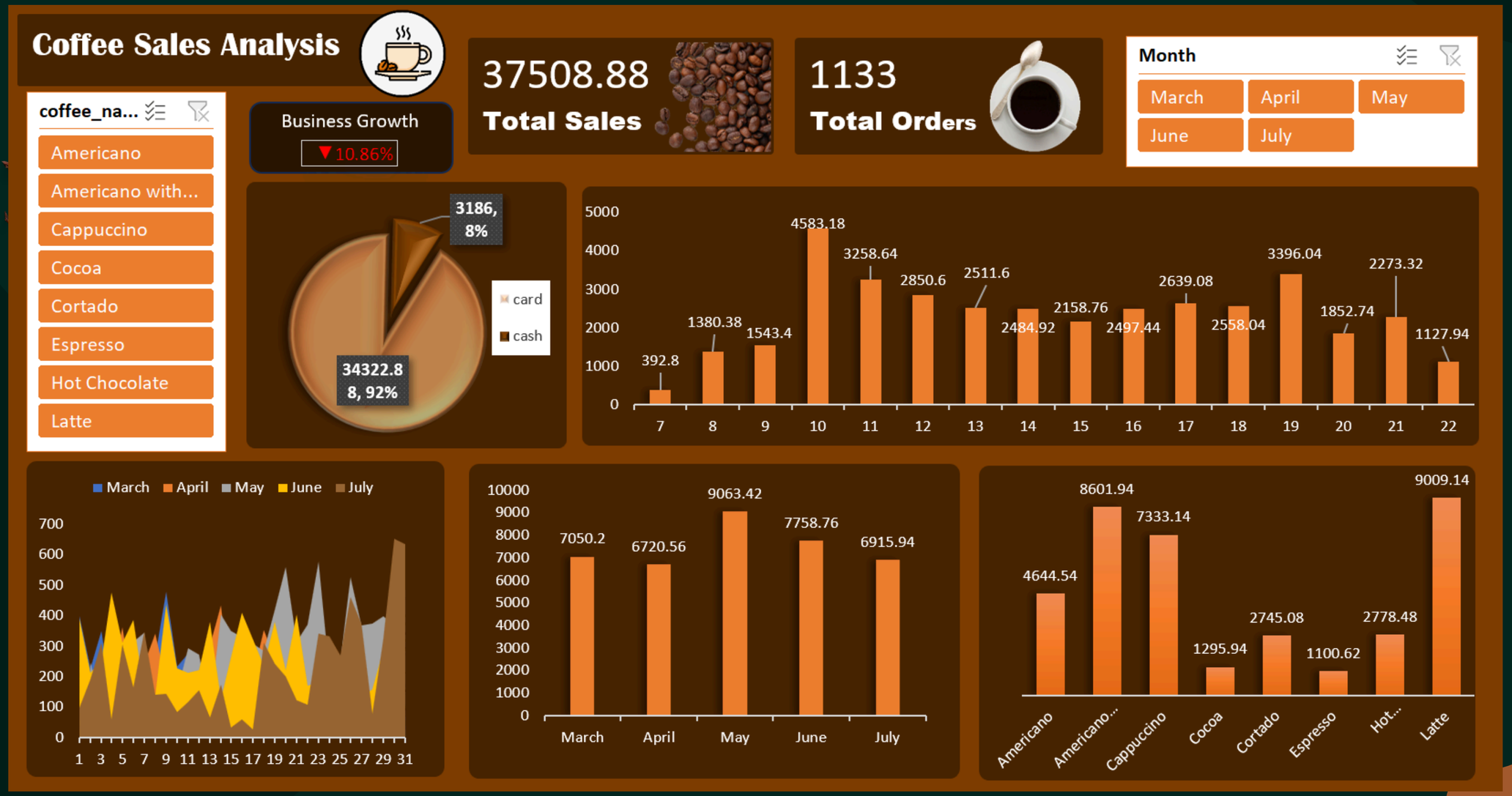
Excel

My Coffee Shop Sales Analysis project focuses on analyzing sales performance using Excel dashboards. The dashboard provides key insights, including total sales, profit, quantity sold, and order trends. It also highlights sales growth, profit margins, and top-selling products. Interactive slicers allow filtering by region, product category, and time period, making it easy to explore data. Visualizations like bar charts, line graphs, and pivot tables help in identifying trends and optimizing business strategies. This project demonstrates how Excel can be used for effective sales analysis and data-driven decision-making.

COFFEE SHOP SALES ANALIYSIS

Home About Contact

Excel Dashboard



COFFEE SHOP SALES ANALIYSIS

Home About Contact

Mr. Dipankar Pal

DATA ANALYST

Python || SQL || Power BI || Excel

click here