Walmart

End-to-end analytics : SQL-Power BI -Python ML

KAILASH BISHT

Introduction

As a Data Analyst at Walmart, your primary role is to turn raw transactional and operational data into actionable business insights. In this capstone, you will simulate real-world analytics challenges that Walmart faces — from identifying profitable product lines, analyzing customer patterns, improving logistics, to predicting future sales and returns.

This project will guide you through three key stages that mirror actual data workflows at large retailers:

1. Advanced Data Extraction & Analysis with SQL
2. Business Visualization through Power BI / Tableau
3. Predictive & Prescriptive Modeling using Python

Dataset

The dataset provided captures Walmart’s operations across multiple regions and includes:

- Customers: Customer IDs, location details (city, state).
- Orders: Order lifecycle timestamps & statuses.
- Order Items: Products sold, prices, freight costs.
- Products: Product categories and basic metadata.
- Payments: Payment values and types.
- Sellers: Seller operational states & cities.
- Geolocation: Additional location context (optional).

Dataset Overview

Row Counts by Table

Customers-99,441 rows
Sellers-3,095 rows
Products-32,951 rows
Orders-99,441 rows
Orders Items-112,650 rows
Payments-103,886 rows
Geolocations-1,000,163 rows

Phase 1: Advanced SQL Analysis

Tasks & Questions

1. Calculate total sales revenue and quantity sold by product category and customer_state.
2. Identify the top 5 products by total sales revenue across all Walmart regions.
3. Find customers with the highest number of orders and total spend, ranking them as Walmart's most valuable customers.
4. Determine customer states with the highest average order value (AOV).
5. Compute average delivery time (in days) by seller state, calculated as the difference between order_purchase_timestamp and order_delivered_customer_date.
6. List the top 5 sellers based on total revenue earned.
7. Analyze the monthly revenue trend over the last 12 months to track Walmart's growth.
8. Calculate the number of new unique customers acquired each month, based on customer_unique_id.
9. Rank customers by lifetime spend within each customer state using SQL window functions.
10. Compute the rolling 3-month average revenue trend, to visualize sales momentum.

```sql
--Calculate total sales revenue and quantity sold by product category and customer_state.
SELECT
    p.[product category] AS product_category,
    c.customer_state,
    SUM(oi.price) AS total_sales_revenue,
    COUNT(oi.order_item_id) AS total_quantity_sold
FROM order_items oi
JOIN products p ON oi.product_id = p.product_id
JOIN orders o ON oi.order_id = o.order_id
JOIN customers c ON o.customer_id = c.customer_id
GROUP BY p.[product category], c.customer_state
ORDER BY total_sales_revenue DESC;
```

	product_category	customer_state	total_sales_revenue	total_quantity_sold
1	bed table bath	SP	478284.519999997	5235
2	HEALTH BEAUTY	SP	462305.219999998	4204
3	Watches present	SP	435009.919999999	2281
4	sport leisure	SP	386357.009999999	3667
5	computer accessories	SP	350747.88	3170
6	Furniture Decoration	SP	286708.020000001	3531
7	housewares	SP	275378.630000001	3265
8	automotive	SP	214277.270000001	1747
9	Cool Stuff	SP	213186.89	1364

```sql
--Identify the top 5 products by total sales revenue across all Walmart regions.
SELECT TOP 5
    oi.product_id,
    p.[product category] AS product_category,
    SUM(oi.price) AS total_sales_revenue,
    COUNT(oi.order_item_id) AS total_quantity_sold
FROM order_items oi
JOIN products p
    ON oi.product_id = p.product_id
GROUP BY oi.product_id, p.[product category]
ORDER BY total_sales_revenue DESC;
```

product_id	product_category	total_revenue
fca62108387e25005784da2f551466a1	Furniture Decoration	249.9
9d395a6bc9dbad2dfe60c8c92d8df397	Furniture Decoration	148
36ed077d2f4915bc4bd14ead16a028de	foods	129.99
dd2ca29c9d0f93d3b1d28d7d7ee3025c	Room Furniture	99
b0cd550945da1de6eb9cb69595d540df	sport leisure	84.9

```sql
--Find customers with the highest number of orders and total spend,
--ranking them as Walmart's most valuable customers.

SELECT
    c.customer_id,
    COUNT(DISTINCT o.order_id) AS total_orders,
    SUM(oi.price) AS total_spend,
    RANK() OVER (ORDER BY SUM(oi.price) DESC) AS spend_rank
FROM customers c
JOIN orders o
    ON c.customer_id = o.customer_id
JOIN order_items oi
    ON o.order_id = oi.order_id
GROUP BY c.customer_id
ORDER BY total_spend DESC, total_orders DESC;
```

	customer_id	total_orders	total_spend	spend_rank
1	1617b1357756262bfa56ab541c47bc16	1	13440	1
2	ec5b2ba62e5743423868716 31fafd3fc	1	7160	2
3	c6e2731c5b391845f6800c97401a43a9	1	6735	3
4	f48d464a0baaea338cb25f816991ab1f	1	6729	4
5	3fd6777bbce08a352fddd04e4a7cc8f6	1	6499	5
6	05455dfa7cd02f13d132aa7a6a9729c6	1	5934.6	6
7	df55c14d1476a9a3467f131269c2477f	1	4799	7
8	24bbf5fd2f2e1b359ee7de94defc4a15	1	4690	8

```sql
--Determine customer states with the highest average order value (AOV).
SELECT
    c.customer_state,
    COUNT(DISTINCT o.order_id) AS total_orders,
    SUM(oi.price) AS total_revenue,
    CAST(SUM(oi.price) AS FLOAT) / COUNT(DISTINCT o.order_id) AS avg_order_value
FROM customers c
JOIN orders o
    ON c.customer_id = o.customer_id
JOIN order_items oi
    ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY avg_order_value DESC;
```

	customer_state	total_orders	total_revenue	avg_order_value
1	PB	532	115268.08	216.669323308271
2	AP	68	13474.3	198.151470588235
3	AC	81	15982.95	197.32037037037
4	AL	411	80314.81	195.413163017032
5	RO	247	46140.6400000001	186.804210526316
6	PA	970	178947.809999999	184.482278350514
7	TO	279	49621.7400000001	177.855698924731
8	PI	493	86914.08	176.29630831643
9	MT	903	156453.529999999	173.259723145071
10	RN	482	83034.98	172.271742738589
11	CE	1327	227254.709999998	171.254491333834
12	SE	345	58920.8500000001	170.785072463768

```sql
--Compute average delivery time (in days) by seller state, calculated
--as the difference between order_purchase_timestamp and order_delivered_customer_date.
SELECT
    s.seller_state,
    AVG(DATEDIFF(DAY, o.order_purchase_timestamp, o.order_delivered_customer_date)) AS avg_delivery_days
FROM orders o
JOIN order_items oi
    ON o.order_id = oi.order_id
JOIN sellers s
    ON oi.seller_id = s.seller_id
WHERE o.order_delivered_customer_date IS NOT NULL
GROUP BY s.seller_state
ORDER BY avg_delivery_days;
```

	seller_state	avg_delivery_days
1	RJ	11
2	RS	11
3	MG	12
4	PB	12
5	GO	12
6	RN	12
7	PE	12
8	DF	12
9	SE	12
10	MS	12
11	ES	12

```sql
--List the top 5 sellers based on total revenue earned.
SELECT TOP 5
    s.seller_id,
    s.seller_state,
    SUM(oi.price) AS total_revenue
FROM order_items oi
JOIN sellers s
    ON oi.seller_id = s.seller_id
GROUP BY s.seller_id, s.seller_state
ORDER BY total_revenue DESC;
```

	seller_id	seller_state	total_revenue
1	4869f7a5dfa277a7dca6462dcf3b52b2	SP	229472.629999998
2	53243585a1d6dc2643021fd1853d8905	BA	222776.05
3	4a3ca9315b744ce9f8e9374361493884	SP	200472.919999995
4	fa1c13f2614d7b5c4749cbc52fecda94	SP	194042.029999998
5	7c67e1448b00f6e969d365cea6b010ab	SP	187923.889999999

```sql
--Analyze the monthly revenue trend over the last 12 months to track Walmart’s growth.

SELECT
    FORMAT(CONVERT(DATETIME, o.order_purchase_timestamp, 120), 'yyyy-MM') AS year_month,
    SUM(oi.price) AS total_revenue
FROM orders o
JOIN order_items oi
    ON o.order_id = oi.order_id
GROUP BY FORMAT(CONVERT(DATETIME, o.order_purchase_timestamp, 120), 'yyyy-MM')
ORDER BY year_month;
```

	year_month	total_revenue
1	2016-09	267.36
2	2016-10	49507.6599999999
3	2016-12	10.9
4	2017-01	120312.87
5	2017-02	247303.020000001
6	2017-03	374344.3
7	2017-04	359927.23
8	2017-05	506071.139999998
9	2017-06	433038.599999999
10	2017-07	498031.479999998
11	2017-08	573971.679999996

```sql
--Calculate the number of new unique customers acquired each month, based on customer_unique_id.
WITH FirstOrders AS (
    SELECT
        c.customer_unique_id,
        MIN(o.order_purchase_timestamp) AS first_order_date
    FROM customers c
    JOIN orders o
        ON c.customer_id = o.customer_id
    GROUP BY c.customer_unique_id
)
SELECT
    FORMAT(CAST(first_order_date AS DATETIME), 'yyyy-MM') AS year_month,
    COUNT(DISTINCT customer_unique_id) AS new_customers
FROM FirstOrders
GROUP BY FORMAT(CAST(first_order_date AS DATETIME), 'yyyy-MM')
ORDER BY year_month;
```

	year_month	new_customers
1	2016-09	4
2	2016-10	321
3	2016-12	1
4	2017-01	764
5	2017-02	1752
6	2017-03	2636
7	2017-04	2352
8	2017-05	3596
9	2017-06	3139
10	2017-07	3894
11	2017-08	4184
12	2017-09	4130
13	2017-10	4470
14	2017-11	7304
15	2017-12	5487

```sql
--Rank customers by lifetime spend within each customer state using SQL window functions.
WITH CustomerSpend AS (
    SELECT
        c.customer_unique_id,
        c.customer_state,
        SUM(oi.price) AS lifetime_spend
    FROM customers c
    JOIN orders o
        ON c.customer_id = o.customer_id
    JOIN order_items oi
        ON o.order_id = oi.order_id
    GROUP BY c.customer_unique_id, c.customer_state
)
SELECT
    customer_state,
    customer_unique_id,
    lifetime_spend,
    RANK() OVER (PARTITION BY customer_state ORDER BY lifetime_spend DESC) AS state_rank
FROM CustomerSpend
ORDER BY customer_state, state_rank;
```

	customer_state	customer_unique_id	lifetime_spend	state_rank
1	AC	62a459e5629b03dd73134964df732077	1200	1
2	AC	086d6b5b5ba195a91aa0a6ec8e75d1a4	961.6	2
3	AC	3947ca729a860c522a64a49d762baada	839.99	3
4	AC	3e5c928acf49c4b95e57af1f350d3493	809.1	4
5	AC	28989ef45087c96e5a4346e88216c2ba	589.6	5
6	AC	22c739518f5240ffd2f80e0087aebe26	549	6
7	AC	c6bc2bf4b75f3f9da5ce95971a2f463b	527.9	7
8	AC	0845d810b482f4d469d3c88380f4a7d5	524.9	8
9	AC	2ec67750cd5b985538c03061ebee01ef	510	9
10	AC	12e92c0f870fc6941c2bfafae6357c4b	479.4	10
11	AC	8c792a4fb7f5d708257abf55fa899ce8	399	11
12	AC	85c3726baee0e11d07b88888586cb59f	388	12
13	AC	aab7b3f97f05bbb173c7d408f8ee96a0	369.8	13

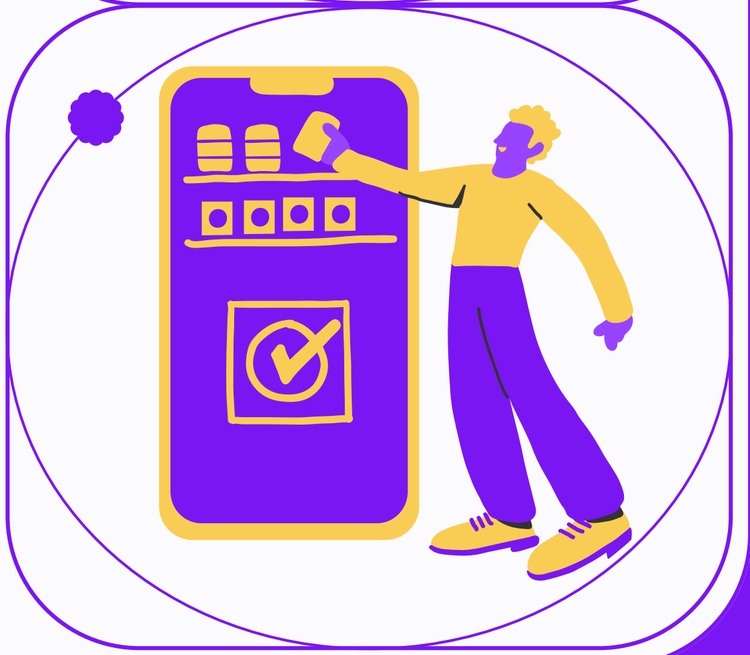
```sql
--Compute the rolling 3-month average revenue trend, to visualize sales momentum.
-- Step 1: Aggregate monthly revenue
WITH MonthlyRevenue AS (
    SELECT
        CAST(YEAR(CONVERT(DATETIME, o.order_purchase_timestamp, 120)) AS VARCHAR(4)) + '-' +
        RIGHT('0' + CAST(MONTH(CONVERT(DATETIME, o.order_purchase_timestamp, 120)) AS VARCHAR(2)), 2) AS year_month,
        SUM(oi.price) AS monthly_revenue
    FROM orders o
    JOIN order_items oi
        ON o.order_id = oi.order_id
    GROUP BY YEAR(CONVERT(DATETIME, o.order_purchase_timestamp, 120)),
             MONTH(CONVERT(DATETIME, o.order_purchase_timestamp, 120))
)

-- Step 2: Compute rolling 3-month average
SELECT
    year_month,
    monthly_revenue,
    AVG(monthly_revenue) OVER (
        ORDER BY year_month
        ROWS BETWEEN 2 PRECEDING AND CURRENT ROW
    ) AS rolling_3_month_avg_revenue
FROM MonthlyRevenue
ORDER BY year_month;
```

	year_month	monthly_revenue	rolling_3_month_avg_revenue
1	2016-09	267.36	267.36
2	2016-10	49507.6600000002	24887.5100000001
3	2016-12	10.9	16595.3066666667
4	2017-01	120312.87	56610.4766666666
5	2017-02	247303.019999995	122542.263333332
6	2017-03	374344.300000001	247320.063333332
7	2017-04	359927.230000002	327191.516666666
8	2017-05	506071.14000001	413447.556666671
9	2017-06	433038.600000005	433012.323333339
10	2017-07	498031.48000001	479047.073333342
11	2017-08	573971.680000013	501680.586666676
12	2017-09	624401.690000015	565468.283333346
13	2017-10	664219.430000018	620864.266666682
14	2017-11	1010271.37000004	766297.49666669
15	2017-12	743914.17000002	806134.990000025
16	2018-01	950030.360000038	901405.300000032

Phase 2: Business Visualization – Power BI / Tableau

Objective

Build a rich, interactive dashboard for Walmart's executive team that combines sales, customer, regional, operational, and time-based insights, enabling them to drill down into any area of concern or opportunity.



Key Metrics & Visual Sections

Sales Performance

- Total Sales Revenue: Overall revenue across all time.
- Total Quantity Sold: Items sold by Walmart.
- Average Order Value (AOV): Average spend per order.
- Sales by Product Category & Customer State: Understand which categories perform best across regions.
- Top-Selling Products: Top 5 products by revenue.

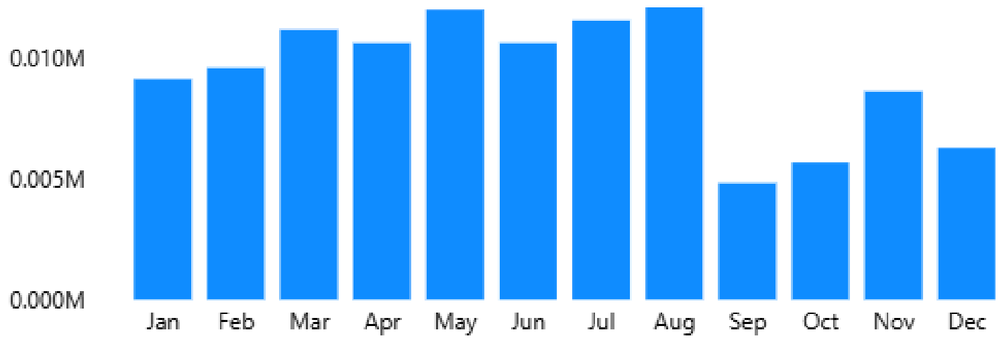
SALES PERFORMANCE

154.84
Average_Order_Value

15.40M
Total_Sales_Revenue

113K
otal_Quantity_Sold

otal_Quantity_Sold by MONTH

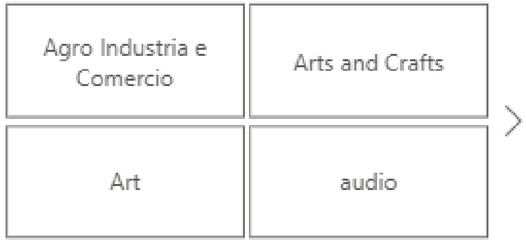


Total_Selling_5_product_id

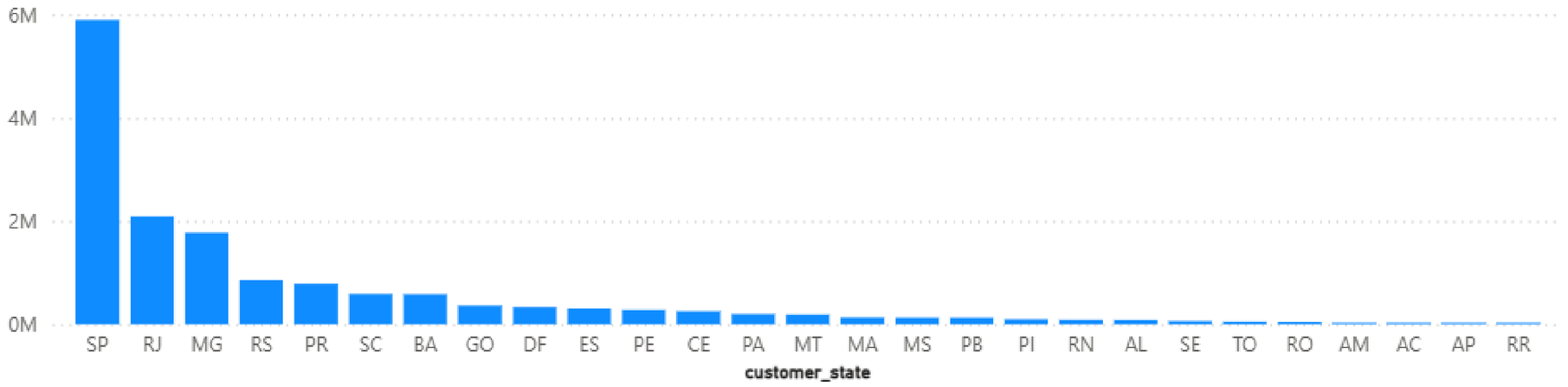
MONTH

geolocation_state

Product_Category



otal_Sales_Revenue by Customer_State



Product_Category	Total_Sales_Revenue
Watches present	1259634.58
toys	507961.96
telephony	360139.72
technical books	19726.84
stationary store	245569.71
sport leisure	1082435.42
song	6059.32
SIGNALIZATION AND SAFETY	34797.13
Room Furniture	87166.25
pet Shop	237722.39
Total	**15397738.61**

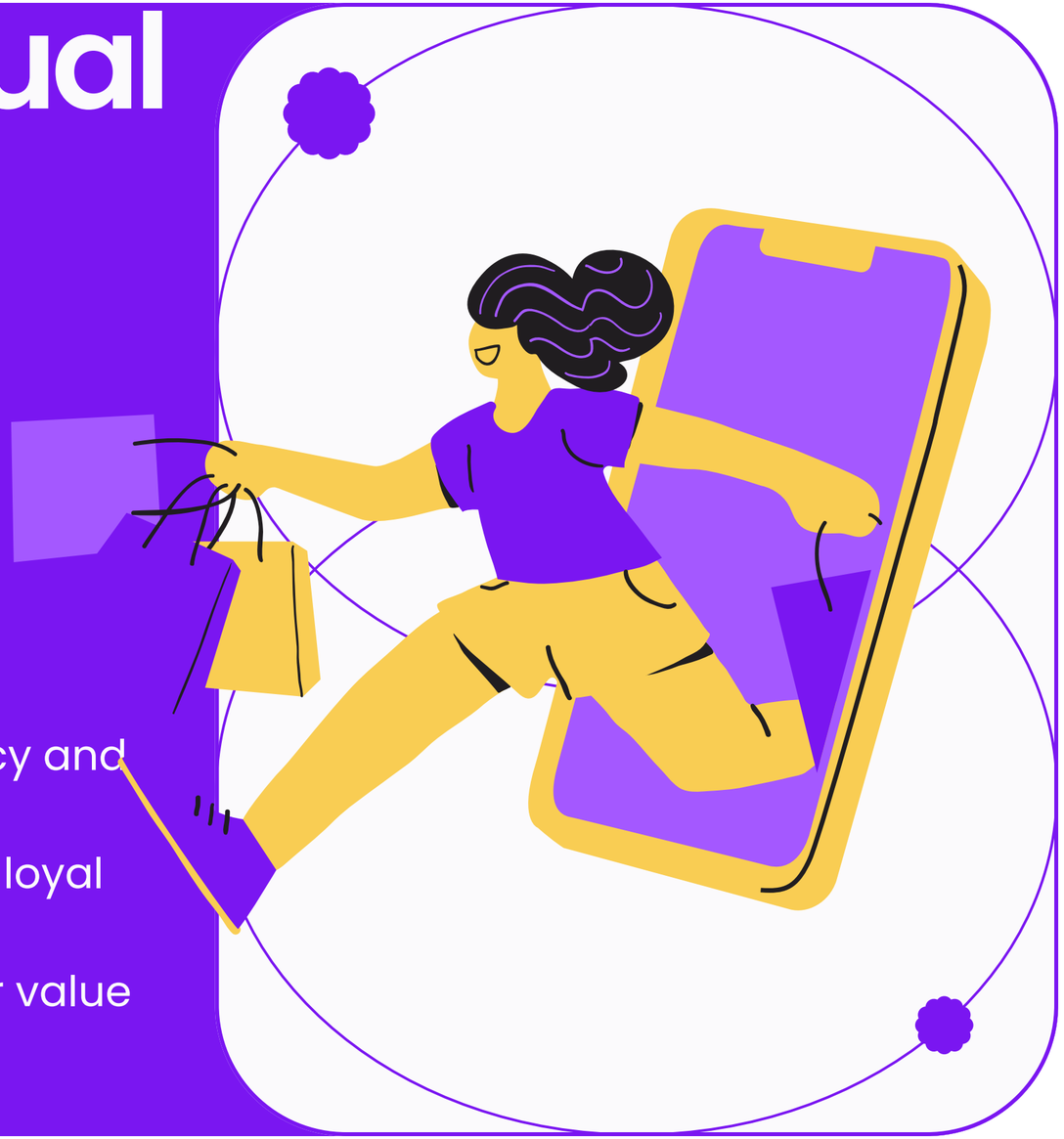
Filters

SALES PERFORMANCE | CUSTOMER INSIGHTS | OPERATIONAL MATRIX | Time Trends & Comparisons

Key Metrics & Visual Sections

Customer Insights

- Customer Lifetime Value (CLV): Total revenue by customer.
- Top 10 Loyal Customers: Highest purchase frequency and spend.
- Customer Segments: From clustering analysis (e.g. loyal high spenders, discount-driven, churn risks).
- Customer State AOV: Comparison of average order value across different customer states.



CUSTOMER INSIGHTS

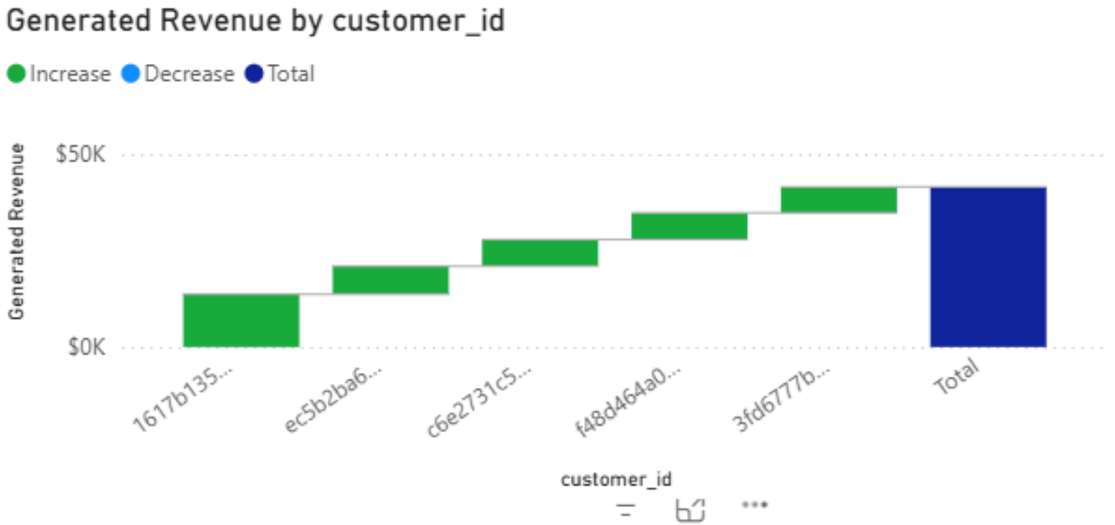
160.99

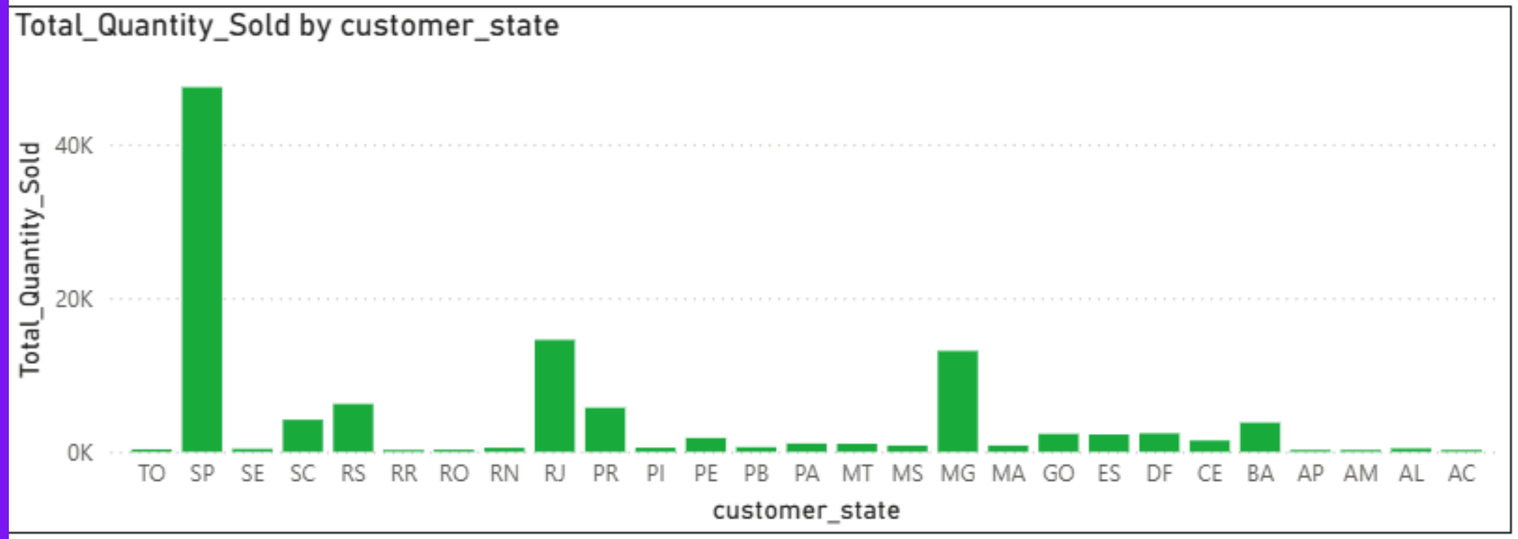
Sum of State_Wise_Average_Order_Value

13.59M

Customer_Lifetime_Value

customer_id	Generated Revenue
1617b1357756262bfa56ab541c47bc16	$13,664.08
ec5b2ba62e574342386871631fafd3fc	$7,274.88
c6e2731c5b391845f6800c97401a43a9	$6,929.31
f48d464a0baaea338cb25f816991ab1f	$6,922.21
3fd6777bbce08a352fddd04e4a7cc8f6	$6,726.66
Total	**$41,517.14**





customer_id	Total_Sales_Revenue
ffffa3172527f765de70084a7e53aae8	32.70
ffff42319e9b2d713724ae527742af25	199.90
fffeda5b6d849fbd39689bb92087f431	47.90
fffecc9f79fd8c764f843e9951b11341	54.90
fffcb937e9dd47a13f05ecb8290f4d3e	78.00
fffb97495f78be80e2759335275df2aa	45.90
fffa0238b217e18a8adeeda0669923a3	35.00
fff93c1da78dafaaa304ff032abc6205	388.32
fff906ecb75de5809be384e0f8d65e45	79.00
fff89c8ed4fcf69a823c1d149e429a0b	30.00
fff7466a253c0e59499ea943462c10f9	152.99
fff675a0d5924b9162b4a1bf410466cd	59.90

MONTH

Jan	Apr	Jul	Oct
Feb	May	Aug	Nov
Mar	Jun	Sep	Dec

geolocation_state

AC	AM	BA	DF	GO
AL	AP	CE	ES	MA

Product_Category

Agro Industria e Comercio	Arts and Crafts
Art	audio

Filters

SALES PERFORMANCE | CUSTOMER INSIGHTS | OPERATIONAL MATRIX | Time Trends & Seasonality

Key Metrics & Visual Sections

Regional & Seller Metrics

- Revenue by Customer State: Identify top contributing states.
- Average Delivery Time by Seller State: Understand regional operational efficiency.
- Top Sellers by Revenue: Showcase sellers driving the most sales.

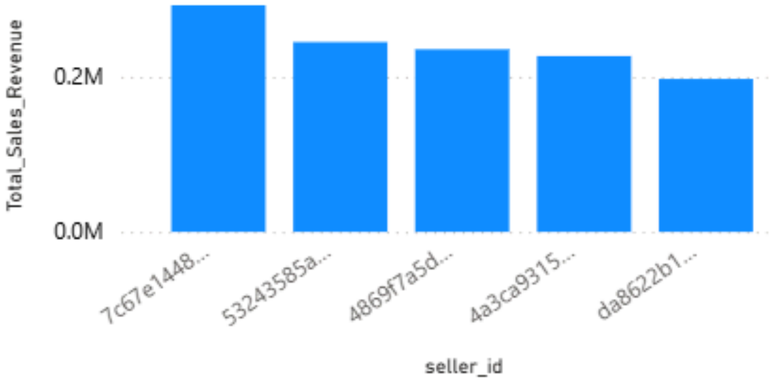
OPERATIONAL MATRIX

12.50
Average of Delivery_Time

16.57%
Freight Cost percent per product cost

Total_Quantity_Sold by MONTH

Total_Sales_Revenue by seller_id

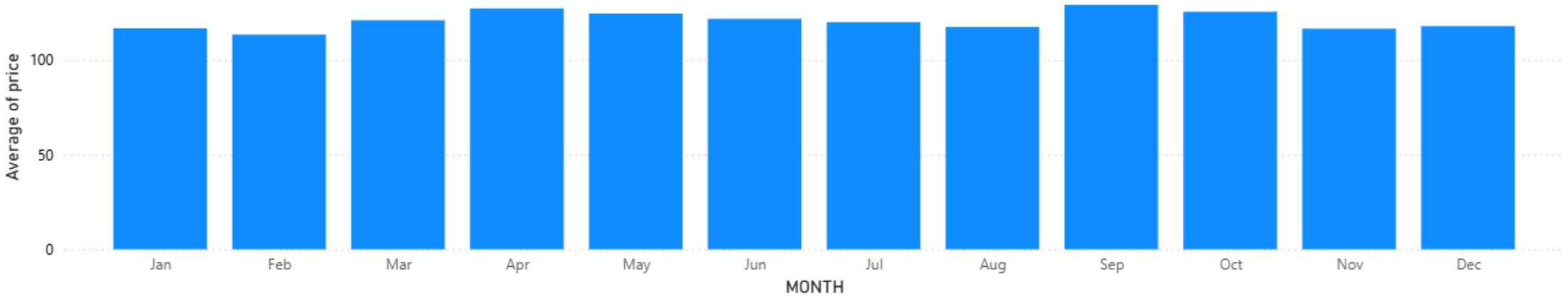


Product_Category

geolocation_state

MONTH

Average of price by MONTH



Filters

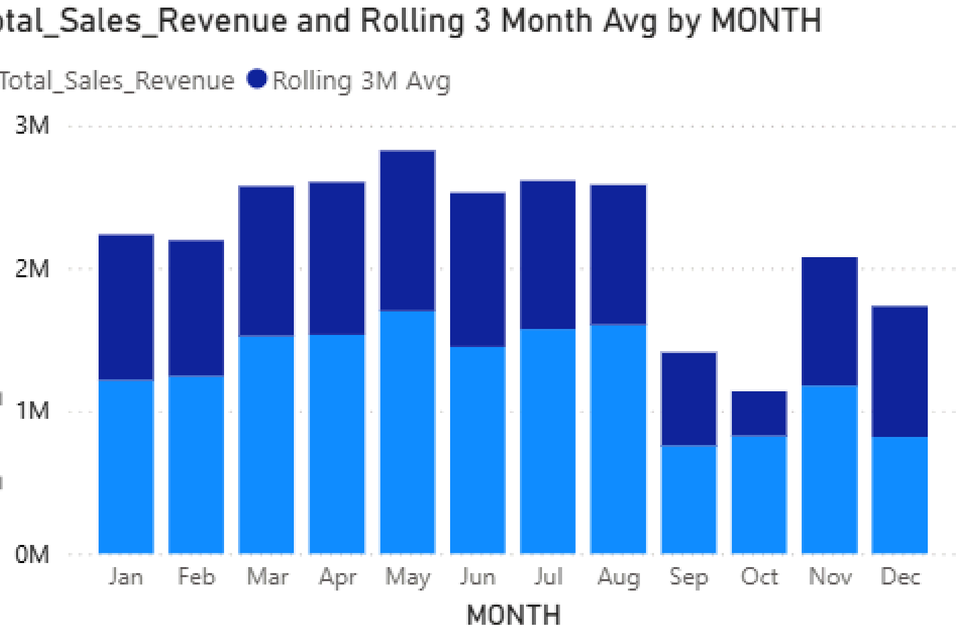
SALES PERFORMANCE | CUSTOMER INSIGHTS | OPERATIONAL MATRIX | Time Trends & Seasonality

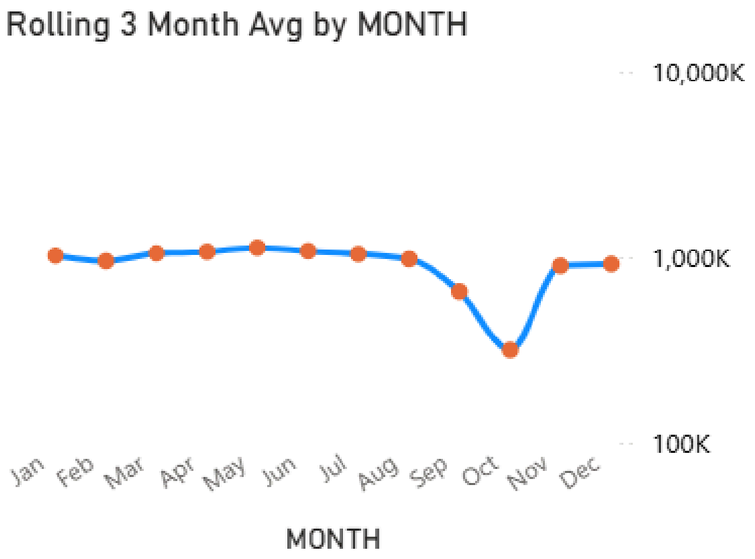
Key Metrics & Visual Sections

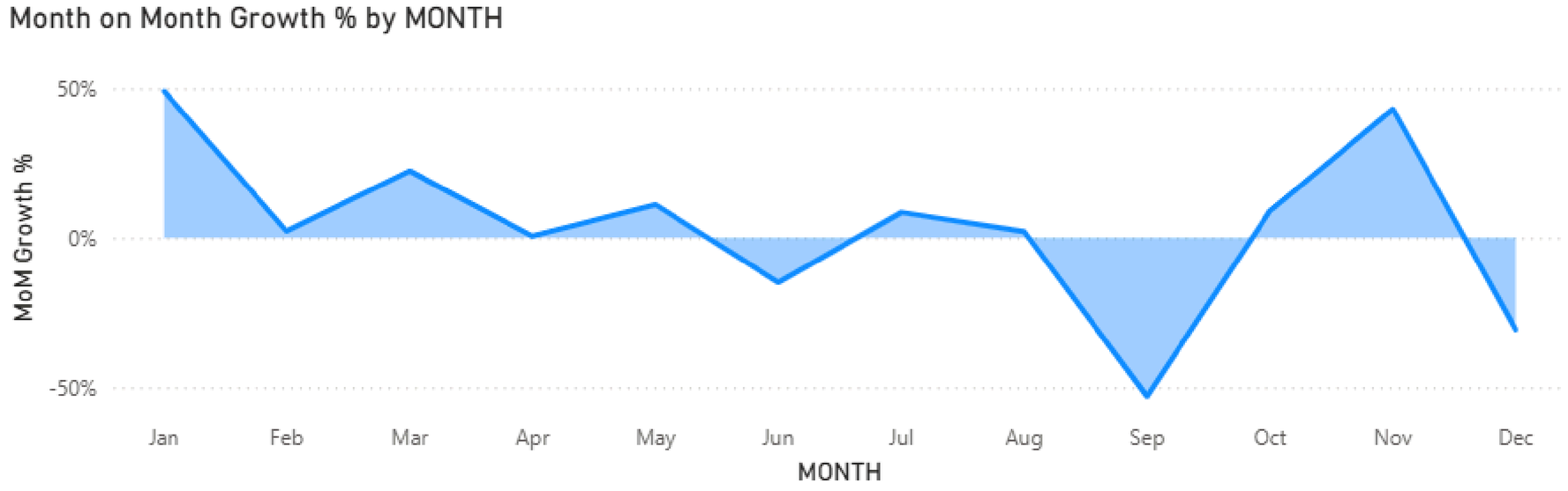
Operational Performance

- Average Delivery Time: Overall average, minimum, and maximum delivery times.
- Delivery SLA Compliance: % of orders delivered within target period (for example, within 5 days).
- Freight Cost Trends: Freight values relative to product price.

Time Trends & Seasonality







customer_state	Sum of Delivery Duration (Minutes)
AC	2423311
AL	14031208
AM	5517741
AP	2622807
BA	90656868
CE	39167831
DF	38840309
ES	45359385
GO	43979775
MA	22273641
MG	196381451
MS	15765699
MT	23036211
PA	32384361
PB	15207278
PE	42318869
PI	13336640
PR	85009448
RJ	272339325
RN	13158485
RO	6778657
RR	1735039
RS	117740818
SC	76407064
SE	10381139
SP	510896705
Total	**1744717214**

Key Metrics & Visual Sections

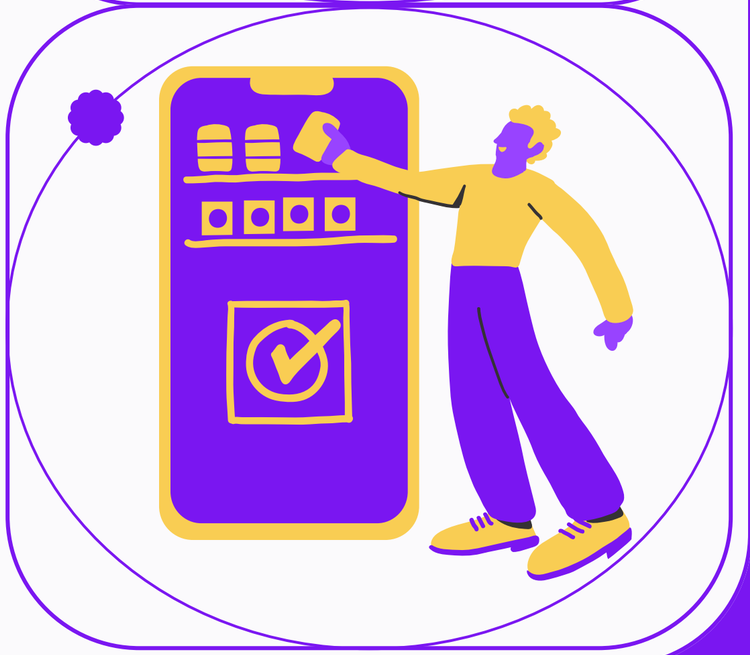
Time Trends & Seasonality

- Monthly Revenue Trend: Observe growth over time.
- Rolling 3-Month Average Sales: Highlight short-term shifts.
- Monthly New Customer Acquisition: Unique new customers joining each month.

Phase 3: Predictive Analytics & Machine Learning with Python

Objective

Deploy predictive models to strengthen Walmart’s future planning and customer strategies.



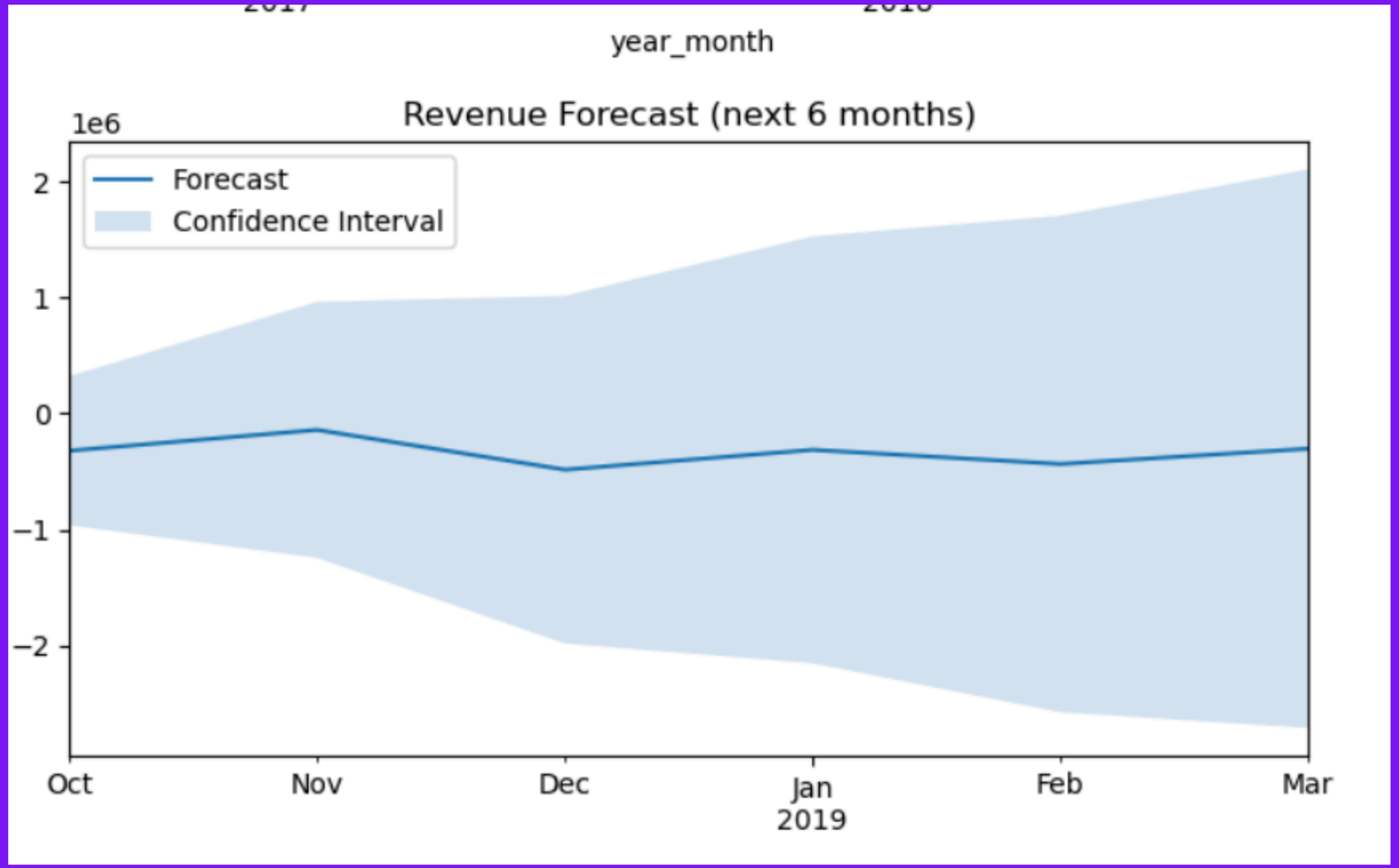
Tasks & Models

Sales & Customer Forecasting

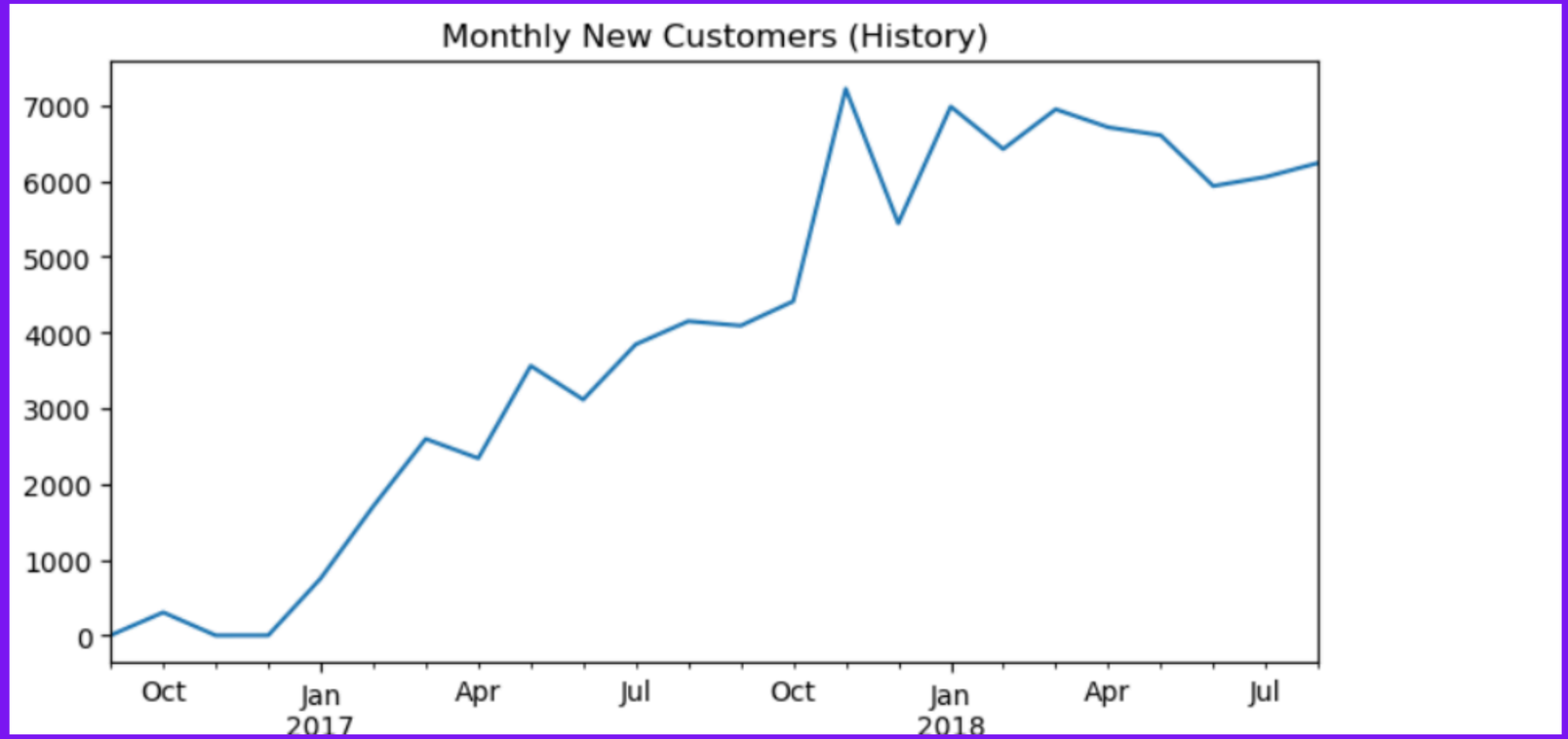
- Forecast overall sales and new customer acquisition for the next 6 months.
- Identify seasonality and trend patterns to anticipate peak periods.

Revenue Trends

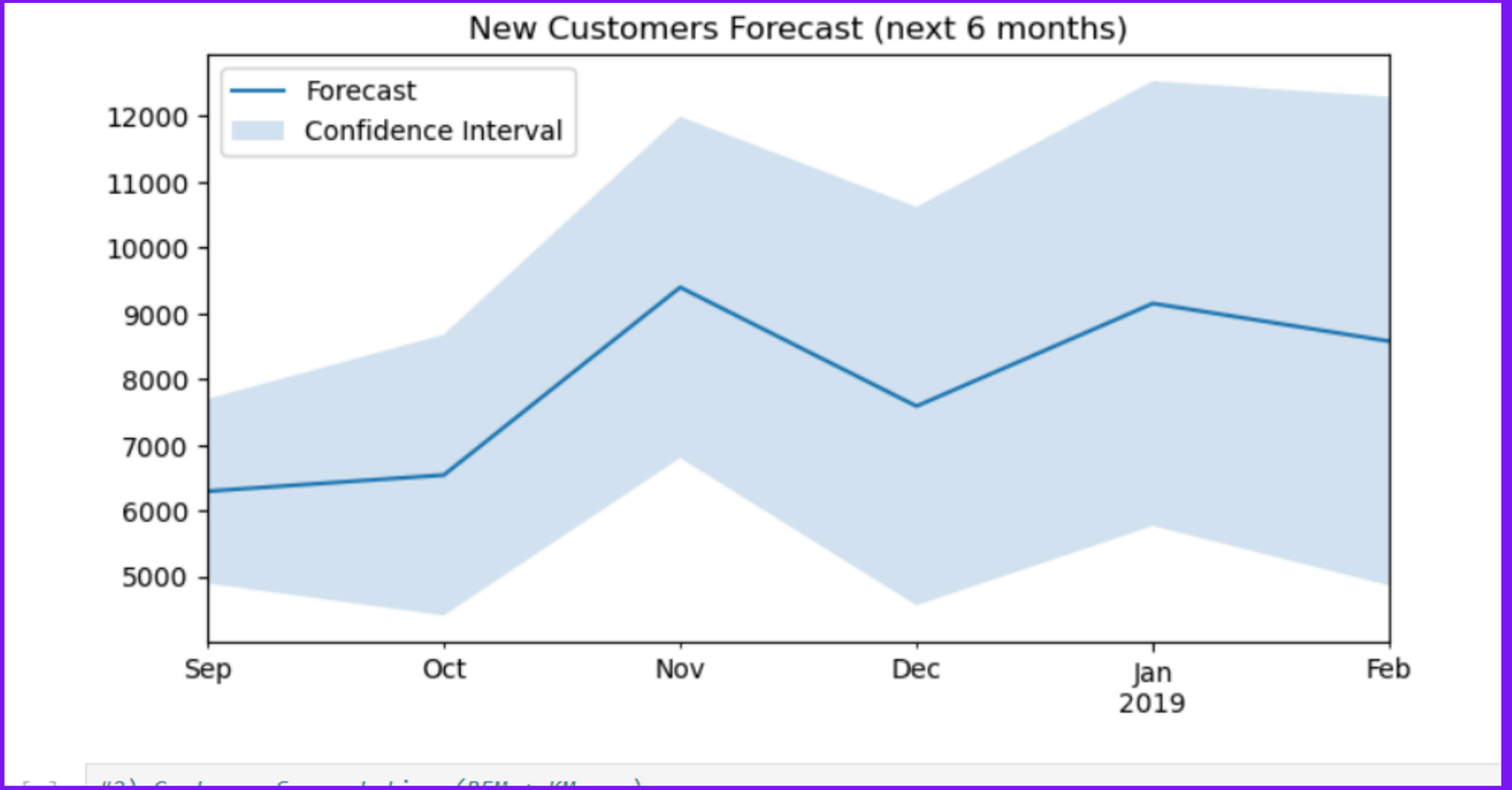
Revenue Forcast



Monthly New Customer



New Customer Forcast (Next 6 Months)

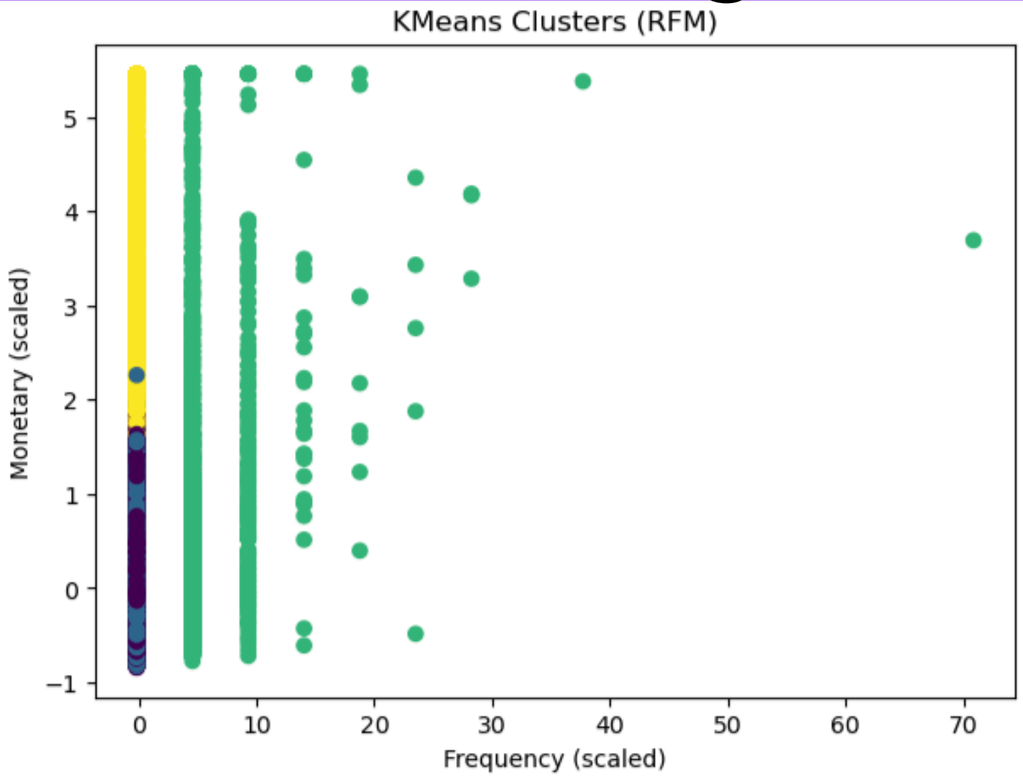


Tasks & Models

Customer Segmentation

- Perform RFM (Recency, Frequency, Monetary) analysis combined with KMeans clustering.
- Label segments such as:
- Loyal High-Value Customers
- Discount-Seeker
- Churn Risks

Customer Segmention:



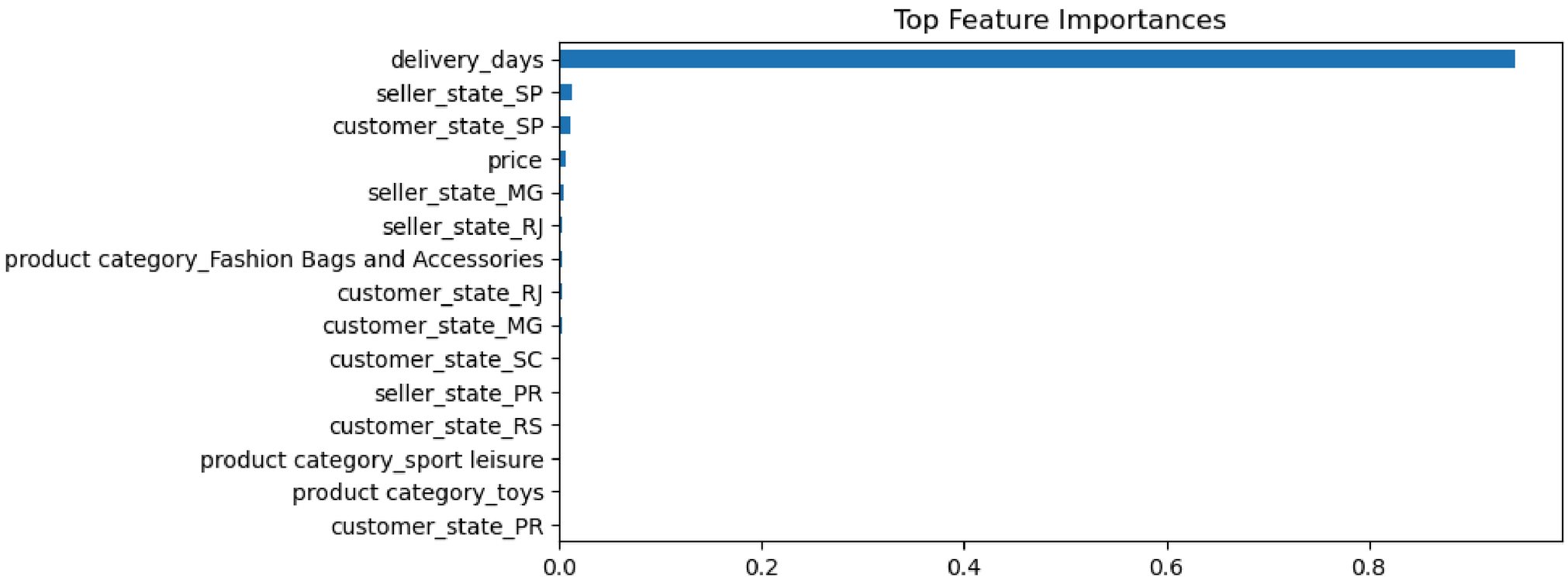
	customer_unique_id	recency_days	frequency	monetary	cluster	segment
0	0000366f3b9a7992bf8c76cfdf3221e2	116	1	129.90	0	Discount-Seeker / Low-Value
1	0000b849f77a49e4a4ce2b2a4ca5be3f	119	1	18.90	0	Discount-Seeker / Low-Value
2	0000f46a3911fa3c0805444483337064	542	1	69.00	1	Discount-Seeker / Low-Value
3	0000f6ccb0745a6a4b88665a16c9f078	326	1	25.99	1	Discount-Seeker / Low-Value
4	0004aac84e0df4da2b147fca70cf8255	293	1	180.00	1	Churn Risk

Tasks & Models

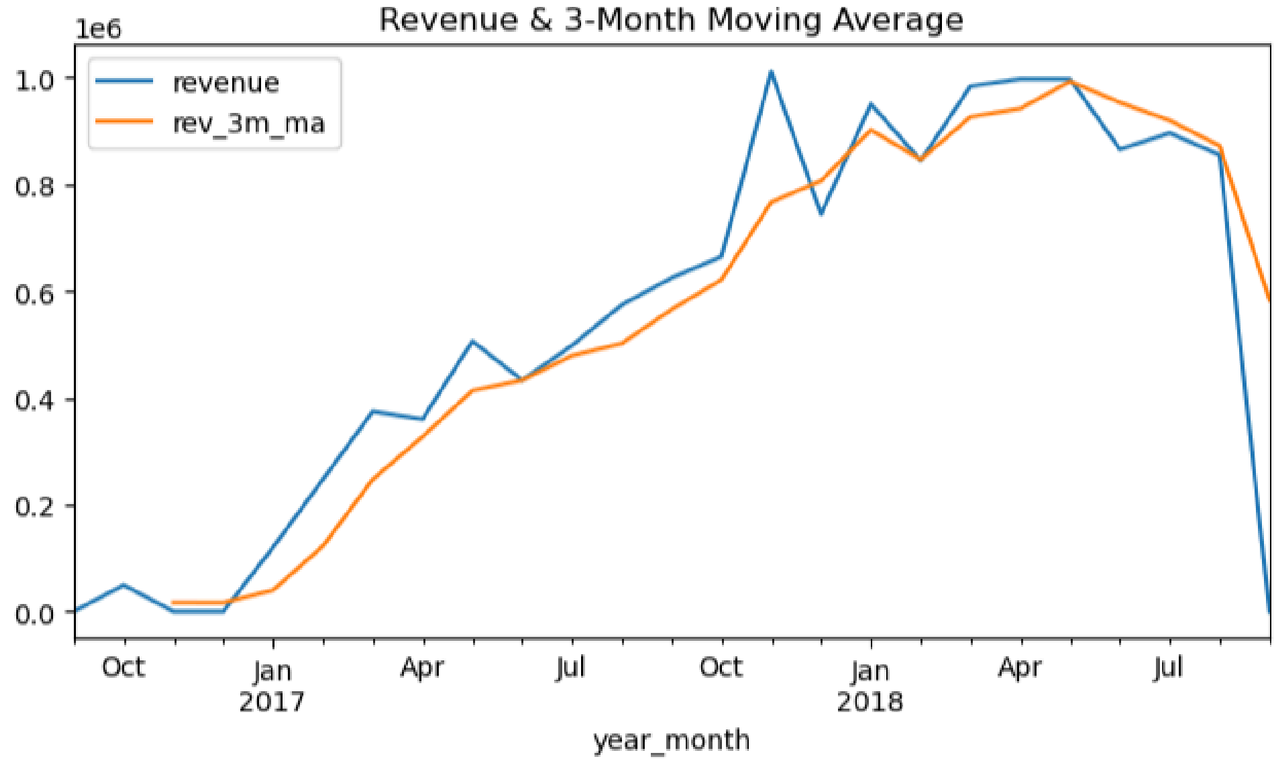
Return Prediction

- Build a classification model to predict if a product will likely be returned, based on:product category
- price
- delivery time
- (other relevant available features from your dataset)

Return Prediction



Revenue Trends with 3- (MMA)Month Moving Average



THE END

Walmart