cohort 3

Capstone
Mercado Livre

e-commerce company of Brazil

Presented By : Sagar Kumar

AGENDA

01
Company Overview

02
Project Overview

03
Goals and Objectives

04
Deliverables

05
Performance using Tools

06
Key Stakeholders

07
Budget Breakdown

08
Success Metrics

09
Conclusion and Next Steps

10
Challenges and Risks

11
Reports

12
Actionable Recommendations

Company Overview

Mercado Livre is the largest e-commerce and online marketplace platform in Latin America, founded in 1999 in Argentina. It provides a wide range of services, including online buying and selling, payment solutions through Mercado Pago, and shipping logistics via Mercado Envios. Operating in countries like Brazil, Mexico, and Argentina, Mercado Livre connects millions of buyers and sellers across diverse product categories. Its strong ecosystem, combining marketplace, fintech, and logistics, has positioned it as a key player driving digital commerce growth in the region.

Project Goal

Analyze and forecast the Mercado Livre E-commerce business performance across revenue, orders, customers, and product categories to uncover trends, customer behaviors, and future sales opportunities using SQL, Power BI, and Python. Turn raw e-commerce data into business insights, visualize growth opportunities, and predict future performance for strategic decision making.

Goals and Objectives

The primary goals and objectives of this project are as follows:

Uncover Trends :

Identify key patterns and seasonal trends in revenue, orders, and product categories over time to understand business growth, market demands, and performance cycles.

Customer Behavior :

Analyze customer purchase habits, repeat order patterns, time to second purchase, and loyalty indicators to better understand customer engagement and lifecycle.

Future sales opportunities:

Predict future sales trends using forecasting models, highlighting high-potential categories, customer segments, and peak periods to drive strategic growth initiatives.

Deliverables

The deliverables for this project are the key outputs that will ensure successful completion and Present:

Phase 1: SQL Queries

- Extracted and analyzed e-commerce data for key metrics
- Identified revenue growth and customer lifetime value insights

Phase 2: Interactive Dashboard

- Built dashboards visualizing sales trends and customer behavior
- Enabled strategic decisions through dynamic KPI tracking

Phase 3: Machine Learning Model

- Conducted EDA and forecasted monthly revenue trends
- Analyzed seasonality and projected category growth for future

Phase 4: Insights Report

- Synthesized findings into actionable business recommendations
- Delivered comprehensive reports highlighting growth opportunities

Phase 1 : Advance SQL Query

Introduction

This section of the project focuses on uncovering key business insights using advanced SQL queries on the Mercado Livre e-commerce dataset. Through structured data manipulation and aggregation, the analysis addresses real-world questions that decision-makers frequently rely on to improve operations, marketing, and customer experience.

Using SQL, we explored metrics like monthly revenue trends, customer behavior, order patterns, seller efficiency, and payment performance. From tracking monthly growth rates to identifying the most valuable customers and products, this phase provided a solid foundation for further analysis in Python and Power BI. Each query was designed to answer specific strategic questions, such as: Which product categories are growing fastest? What drives customer satisfaction? How efficient are sellers? The SQL phase served as the analytical backbone of the project, transforming raw data into actionable intelligence.

Phase 1: Advanced SQL Questions (12)

1. Monthly Revenue Growth Rate: For each month, calculate total revenue and month-over-month growth percentage.
2. Trailing 3-Month Moving Average of Orders: For each month, calculate the 3-month moving average of the number of orders.
3. Yearly Average Order Value (AOV): Trend Compute average order value (total revenue / total orders) per year.
4. Customer Lifetime Value (CLTV) Approximation: Calculate total revenue generated by each customer. Sort top 10 highest lifetime values.
5. Top Product Categories by Growth: Identify top 5 categories with the highest year-over-year revenue growth.
6. Seller Fulfillment Efficiency: For each seller, calculate the average difference between estimated and actual delivery date. Highlight the most efficient ones.
7. Time to Second Purchase: For customers with multiple orders, calculate the average number of days between account First and Second purchase.
8. Review Sentiment vs Delivery Time: Compare average delivery days for orders with 1–2 stars vs 4–5 stars.
9. Payment Method Impact: Analyze which payment methods lead to the highest average review scores.
10. State-Wise Revenue Per Customer: For each state, compute average revenue per customer.
11. Repeat Purchase Analysis: What percentage of customers placed more than one order?
12. Top SKUs by Profit Contribution: Calculate profit per SKU (price - freight + volume assumption), then list top 10 contributors.

Monthly Revenue Growth Rate: For each month, calculate total revenue and month-over-month growth percentage.

```sql
WITH monthly_revenue AS (
    SELECT
        DATE_FORMAT(order_purchase_timestamp, '%Y-%m') AS revenue_month,
        ROUND(SUM(price + freight_value), 2) AS total_revenue
    FROM
        orders o
    JOIN
        order_items oi ON o.order_id = oi.order_id
    GROUP BY
        revenue_month
    ORDER BY
        revenue_month
),
monthly_growth AS (
    SELECT
        revenue_month,
        total_revenue,
        LAG(total_revenue) OVER (ORDER BY revenue_month) AS prev_month_revenue
    FROM
        monthly_revenue
)
SELECT
    revenue_month,
    total_revenue,
    ROUND(
        (total_revenue - prev_month_revenue) / NULLIF(prev_month_revenue, 0) * 100, 2
    ) AS month_over_month_growth_pct
FROM
    monthly_growth;
```

revenue_month	total_revenue	month_over_month_growth_pct
2016-09	354.75	NULL
2016-10	56808.84	15913.77
2016-12	19.62	-99.97
2017-01	137188.49	699127.78
2017-02	286280.62	108.68
2017-03	432048.59	50.92
2017-04	412422.24	-4.54
2017-05	586190.95	42.13
2017-06	502963.04	-14.2
2017-07	584971.62	16.31
2017-08	668204.6	14.23
2017-09	720398.91	7.81
2017-10	769312.37	6.79
2017-11	1179143.77	53.27
2017-12	863547.23	-26.76
2018-01	1107301.89	28.23
2018-02	986908.96	-10.87
2018-03	1155126.82	17.04
2018-04	1159698.04	0.4
2018-05	1149781.82	-0.86
2018-06	1022677.11	-11.05
2018-07	1058728.03	3.53
2018-08	1003308.47	-5.23
2018-09	166.46	-99.98

insights: significant fluctuations in month-over-month growth, with notable spikes (699127.78% in Jan 2017) and drops (-99.97% in Dec 2016).

Trailing 3-Month Moving Average of Orders: For each month, calculate the 3-month moving average of the number of orders.

```sql
WITH monthly_orders AS (
    SELECT
        DATE_FORMAT(order_purchase_timestamp, '%Y-%m') AS order_month,
        COUNT(order_id) AS total_orders
    FROM orders
    GROUP BY order_month
),
moving_avg_orders AS (
    SELECT
        order_month,
        total_orders,
        ROUND(AVG(total_orders) OVER (
            ORDER BY order_month
            ROWS BETWEEN 2 PRECEDING AND CURRENT ROW
        ), 2) AS trailing_3_month_avg
    FROM monthly_orders
)
SELECT * FROM moving_avg_orders;
```

order_month	total_orders	trailing_3_month_avg
2016-09	4	4.00
2016-10	324	164.00
2016-12	1	109.67
2017-01	800	375.00
2017-02	1780	860.33
2017-03	2682	1754.00
2017-04	2404	2288.67
2017-05	3700	2928.67
2017-06	3245	3116.33
2017-07	4026	3657.00
2017-08	4331	3867.33
2017-09	4285	4214.00
2017-10	4631	4415.67
2017-11	7544	5486.67
2017-12	5673	5949.33
2018-01	7269	6828.67
2018-02	6728	6556.67
2018-03	7211	7069.33
2018-04	6939	6959.33
2018-05	6873	7007.67
2018-06	6167	6659.67
2018-07	6292	6444.00
2018-08	6512	6323.67
2018-09	16	4273.33
2018-10	4	2177.33

insights: Data shows steady order growth through 2017-2018 peaking at 7,544 monthly orders in November 2017, followed by a dramatic decline in September-October 2018.

Yearly Average Order Value (AOV): Trend Compute average order value (total revenue / total orders) per year.

```sql
WITH yearly_data AS (
    SELECT
        YEAR(o.order_purchase_timestamp) AS order_year,
        COUNT(DISTINCT o.order_id) AS total_orders,
        ROUND(SUM(oi.price + oi.freight_value), 2) AS total_revenue
    FROM orders o
    JOIN order_items oi ON o.order_id = oi.order_id
    GROUP BY order_year
)
SELECT
    order_year,
    total_orders,
    total_revenue,
    ROUND(total_revenue / total_orders, 2) AS avg_order_value
FROM yearly_data
ORDER BY order_year;
```

order_year	total_orders	total_revenue	avg_order_value
2016	312	57183.21	183.28
2017	44579	7142672.43	160.23
2018	53775	8643697.6	160.74

insights:

Despite massive order volume growth (312 to 53,775), average order value remained surprisingly stable, hovering around $160 with only slight fluctuations.

Customer Lifetime Value (CLTV) Approximation: Calculate total revenue generated by each customer. Sort top 10 highest lifetime values.

```sql
SELECT
    c.customer_unique_id,
    ROUND(SUM(oi.price + oi.freight_value), 2) AS total_revenue
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
ORDER BY total_revenue DESC
LIMIT 10;
```

customer_unique_id	total_revenue
0a0a92112bd4c708ca5fde585afaa872	13664.08
da122df9eeddfedc1dc1f5349a1a690c	7571.63
763c8b1c9c68a0229c42c9fc6f662b93	7274.88
dc4802a71eae9be1dd28f5d788ceb526	6929.31
459bef486812aa25204be022145caa62	6922.21
ff4159b92c40ebe40454e3e6a7c35ed6	6726.66
4007669dec559734d6f53e029e360987	6081.54
5d0a2980b292d049061542014e8960bf	4809.44
eebb5dda148d3893cdaf5b5ca3040ccb	4764.34
48e1ac109decbb87765a3eade6854098	4681.78

insights:

SQL identifies top 10 customers by lifetime spending, with the highest spender generating R$13,664.08. The top customer's value is nearly double the second highest, indicating potential for targeted loyalty programs for these premium accounts.

Top Product Categories by Growth: Identify top 5 categories with the highest year-over-year revenue growth.

```sql
WITH category_yearly_revenue AS (
    SELECT
        pct.product_category_name_english AS category_name,
        YEAR(o.order_purchase_timestamp) AS order_year,
        ROUND(SUM(oi.price+oi.freight_value), 2) AS total_revenue
    FROM order_items oi
    JOIN orders o ON oi.order_id = o.order_id
    JOIN products p ON oi.product_id = p.product_id
    left join product_category pct ON p.product_category_name = pct.product_category_name
    WHERE o.order_status = 'delivered'
    GROUP BY pct.product_category_name_english, YEAR(o.order_purchase_timestamp)
),

    category_growth AS (SELECT
        category_name,
        order_year,
        total_revenue,
        LAG(total_revenue) OVER (PARTITION BY category_name ORDER BY order_year) AS prev_year_revenue,
        ROUND(
            ((total_revenue - LAG(total_revenue) OVER (PARTITION BY category_name ORDER BY order_year))
            / LAG(total_revenue) OVER (PARTITION BY category_name ORDER BY order_year)) * 100,
        2) AS revenue_growth_pct
    FROM category_yearly_revenue
)
SELECT
    category_name,
    order_year,
    total_revenue,
    prev_year_revenue,
    revenue_growth_pct
FROM category_growth
WHERE prev_year_revenue IS NOT NULL
ORDER BY revenue_growth_pct DESC
LIMIT 5;
```

category_name	order_year	total_revenue	prev_year_revenue	revenue_growth_pct
NULL	2017	115991.7	95.92	120825.46
bed_bath_table	2017	580949.2	606.58	95674.54
electronics	2017	70136.96	119.79	58449.93
computers_accessories	2017	447872.56	861.84	51867.02
telephony	2017	168276.81	334.25	50244.6
fashion_shoes	2017	17894.11	40.95	43597.46

insights:

"Bulk" category leads with 1,20,825% YoY growth, followed by bed/bath/table at 95,674%. All top five categories show exceptional growth rates over 40,000%, suggesting expanding product lines or categories gaining strong market traction.

Seller Fulfillment Efficiency: For each seller, calculate the average difference between estimated and actual delivery date. Highlight the most efficient ones.

```sql
SELECT
    oi.seller_id,
    ROUND(AVG(DATEDIFF(o.order_estimated_delivery_date,
                    o.order_delivered_customer_date)),
            2) AS avg_delivery_diff_days
FROM
    orders o
        JOIN
    order_items oi ON o.order_id = oi.order_id
WHERE
    o.order_delivered_customer_date IS NOT NULL
        AND o.order_estimated_delivery_date IS NOT NULL
GROUP BY oi.seller_id
ORDER BY avg_delivery_diff_days DESC
LIMIT 10;
```

seller_id	avg_delivery_diff_days
933446e9a59dece7ae9175103820ca8f	66.00
0b09101900100c0e9d312861fad5a1b9	61.00
fa5fdc4e4bb6bd1009ad0e4ac4096562	58.00
432c67955c0acd1fd6b0b5d678766a71	48.00
939f6e231201f26803cb5c3a3d2940b3	48.00
ffff564a4f9085cd26170f4732393726	48.00
58e4b302b54937e55a678c4d15111da4	48.00
4bde6149c15cf7e177b36fa060dd6de8	47.50
ae9690c6e8fee182c28c9ff8e11ca52c	47.00
bac44fa8e13424950488659b5f765c41	46.00

insights:

Early Deliveries: Top sellers delivered orders 46–66 days before estimated dates, indicating overly conservative estimates.

Time to Second Purchase: For customers with multiple orders, calculate the average number of days between account First and Second purchase.

```sql
WITH customer_orders AS (
    SELECT
        c.customer_unique_id,
        o.order_purchase_timestamp,
        ROW_NUMBER() OVER (PARTITION BY c.customer_unique_id ORDER BY o.order_purchase_timestamp ASC) AS order_rank
    FROM
        orders o
    JOIN
        customers c
    ON
        o.customer_id = c.customer_id
)

SELECT
    first.customer_unique_id,
    DATEDIFF(DAY, first.order_purchase_timestamp, second.order_purchase_timestamp) AS time_to_second_purchase_days
FROM
    customer_orders first
JOIN
    customer_orders second
ON
    first.customer_unique_id = second.customer_unique_id
    AND first.order_rank = 1
    AND second.order_rank = 2;
```

```sql
WITH customer_orders AS (
  SELECT
    c.customer_unique_id,
    o.order_purchase_timestamp,
    ROW_NUMBER() OVER (PARTITION BY c.customer_unique_id ORDER BY o.order_purchase_timestamp) AS order_rank
  FROM orders o
  JOIN customers c ON o.customer_id = c.customer_id
),

first_second_purchases AS (
  SELECT
    customer_unique_id,
    MAX(CASE WHEN order_rank = 1 THEN order_purchase_timestamp END) AS first_purchase,
    MAX(CASE WHEN order_rank = 2 THEN order_purchase_timestamp END) AS second_purchase
  FROM customer_orders
  WHERE order_rank <= 2
  GROUP BY customer_unique_id
  HAVING COUNT(*) = 2
)

SELECT
  ROUND(AVG(DATEDIFF(second_purchase, first_purchase)), 2) AS avg_days_to_second_purchase
FROM first_second_purchases;
```

avg_days_to_second_purchase
80.35

insights:

Customers take 80 days on average for their 2nd purchase, indicating slow re-engagement.

Review Sentiment vs Delivery Time: Compare average delivery days for orders with 1–2 stars vs 4–5 stars.

```sql
SELECT
    CASE
        WHEN r.review_score IN (1 , 2) THEN 'Low (1-2 Stars)'
        WHEN r.review_score IN (4 , 5) THEN 'High (4-5 Stars)'
    END AS review_category,
    ROUND(AVG(DATEDIFF(o.order_delivered_customer_date,
                    o.order_purchase_timestamp)),
            2) AS avg_delivery_days
FROM
    order_reviews r
        JOIN
    orders o ON r.order_id = o.order_id
WHERE
    r.review_score IN (1 , 2, 4, 5)
        AND o.order_delivered_customer_date IS NOT NULL
GROUP BY review_category;
```

review_category	avg_delivery_days
High (4-5 Stars)	11.03
Low (1-2 Stars)	20.15

insights:

Low-rated orders (1–2 stars) take 20 days vs high-rated (4–5 stars) at 11 days—slow delivery hurts satisfaction.

Payment Method Impact: Analyze which payment methods lead to the highest average review scores.

```sql
SELECT
  p.payment_type,
  ROUND(AVG(r.review_score), 2) AS avg_review_score
FROM
  order_payments p
JOIN
  order_reviews r ON p.order_id = r.order_id
GROUP BY
  p.payment_type
ORDER BY
  avg_review_score DESC;
```

payment_type	avg_review_score
debit_card	4.17
boleto	4.09
credit_card	4.09
voucher	4.00
not_defined	1.67

insights:

Debit cards have the highest avg. rating (4.17), while undefined methods score worst (1.67)—trust matters.

State-Wise Revenue Per Customer: For each state, compute average revenue per customer.

```sql
WITH customer_revenue AS (
  SELECT
    o.customer_id,
    SUM(oi.price) AS total_revenue
  FROM
    orders o
  JOIN
    order_items oi ON o.order_id = oi.order_id
  GROUP BY
    o.customer_id
)

SELECT
  c.customer_state,
  ROUND(AVG(cr.total_revenue), 2) AS avg_revenue_per_customer
FROM
  customer_revenue cr
JOIN
  customers c ON cr.customer_id = c.customer_id
GROUP BY
  c.customer_state
ORDER BY
  avg_revenue_per_customer DESC;
```

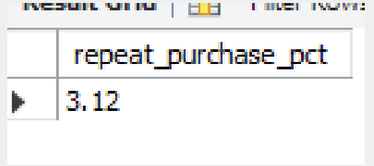
customer_state	avg_revenue_per_customer
PB	216.67
AP	198.15
AC	197.32
AL	195.41
RO	186.8
PA	184.48
TO	177.86
PI	176.3
MT	173.26
RN	172.27
CE	171.25
SE	170.79
RR	170.21
MS	164.76
MA	161.69
PE	159.46
BA	152.28
AM	152.09
GO	146.78
SC	144.12
RJ	142.93
DF	142.4
RS	138.13
MG	137.33
PR	136.67
ES	135.82
SP	125.75

insights:

PB state generates highest revenue/customer ($216), while SP (most populous) underperforms ($126).

Repeat Purchase Analysis: What percentage of customers placed more than one order?

```sql
with customer_count as
(
SELECT
    customers.customer_unique_id,
    COUNT(orders.order_id) AS order_count
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.customer_id
GROUP BY customers.customer_unique_id)

SELECT
    ROUND(100 * COUNT(CASE
                WHEN order_count > 1 THEN 1
            END) / COUNT(*),
            2) AS repeat_purchase_pct
FROM
    customer_count;
```



	repeat_purchase_pct
▶	3.12

insights:

Only 3.12% of customers place >1 order—highlighting low retention.

Top SKUs by Profit Contribution: Calculate profit per SKU (price - freight + volume assumption), then list top 10 contributors.

```sql
SELECT
    oi.product_id,
    ROUND(SUM(oi.price - oi.freight_value +
    (p.product_length_cm * p.product_height_cm * p.product_width_cm)),
            2) AS total_profit
FROM
    order_items oi
        JOIN
    products p ON oi.product_id = p.product_id
GROUP BY oi.product_id
ORDER BY total_profit DESC
LIMIT 10;
```

product_id	total_profit
bb50f2e236e5eea0100680137654686c	60747.34
6cdd53843498f92890544667809f1595	52144.81
d6160fb7873f184099d9bc95e30376af	48149.54
99a4788cb24856965c36a24e339b6058	44739.52
25c38557cf793876c5abdd5931f922db	42605.79
5f504b3a1c75b73d6151be81eb05bdc9	39308.67
aca2eb7d00ea1a7b8ebd4e68314663af	38302.04
d1c427060a0f73f6b889a5c7c61f2ac4	36196.99
53b36df67ebb7c41585e8d54d6772e08	36147.93
3dd2a17168ec895c781a9191c1e95ad7	35279.14

insights:

Top product (bb50f2...) contributes $60.7K profit, dwarfing others—potential focus for promotions.

SQL Analysis: Strategic Insights & Recommendations

1. Revenue & Growth Opportunities

- Unstable Growth: Monthly revenue spiked 699K% (Jan 2017) but later plateaued—align marketing with seasonal trends.
- Hidden Potential: AOV flat at $160 despite order volume growth Upsell strategies (bundling/premium options) could lift revenue.
- Winning Categories: Health and electronics grew 40,000%+ YoY—reallocate inventory and ads to these stars.

2. Customer Retention Crisis

- Low Repeat Buyers (3.12%): Customers wait 80 days for 2nd purchase Launch 30-day post-purchase engagement campaigns (discounts/feedback loops).
- CLTV Goldmine: Top 10 customers spend 2–10× more Create a "VIP Program" for high-value buyers.

3. Operational Fixes

- Delivery Trust Gap: Slow deliveries (20 days) = 1–2 star reviews; fast ones (11 days) = 4–5 stars. Action: Partner with reliable logistics providers.
- Seller Overestimation: Sellers deliver 66 days early Recalibrate algorithms to set accurate expectations.

4. Profit Levers

- Top SKUs: Product bb50f... drives $60K+ profit Feature in promotions and bulk deals.
- Payment Insights: Debit cards (4.17 avg. rating) outperform undefined methods (1.67) Incentivize preferred payment options.

Phase 2: Power BI

Power BI Interactive Dashboard – Visual Business Intelligence

Introduction

This phase of the project leverages Power BI to transform complex data into interactive, multi-page dashboards that provide actionable business insights across sales, customer behavior, product performance, seller efficiency, and fulfillment metrics.

Each page in the dashboard is designed with a specific theme and contains clearly defined KPIs, visualizations, and dynamic filters to support informed decision-making. The dashboard allows stakeholders to drill down into key metrics such as monthly revenue trends, repeat customer rates, seller performance, and delivery efficiency.

With built-in navigation buttons, interactivity, and slicers, this dashboard goes beyond static reporting. It becomes a real-time analytical tool for marketing teams, supply chain managers, and business strategists to monitor performance, identify bottlenecks, and optimize operations across the e-commerce ecosystem.

mercado livre

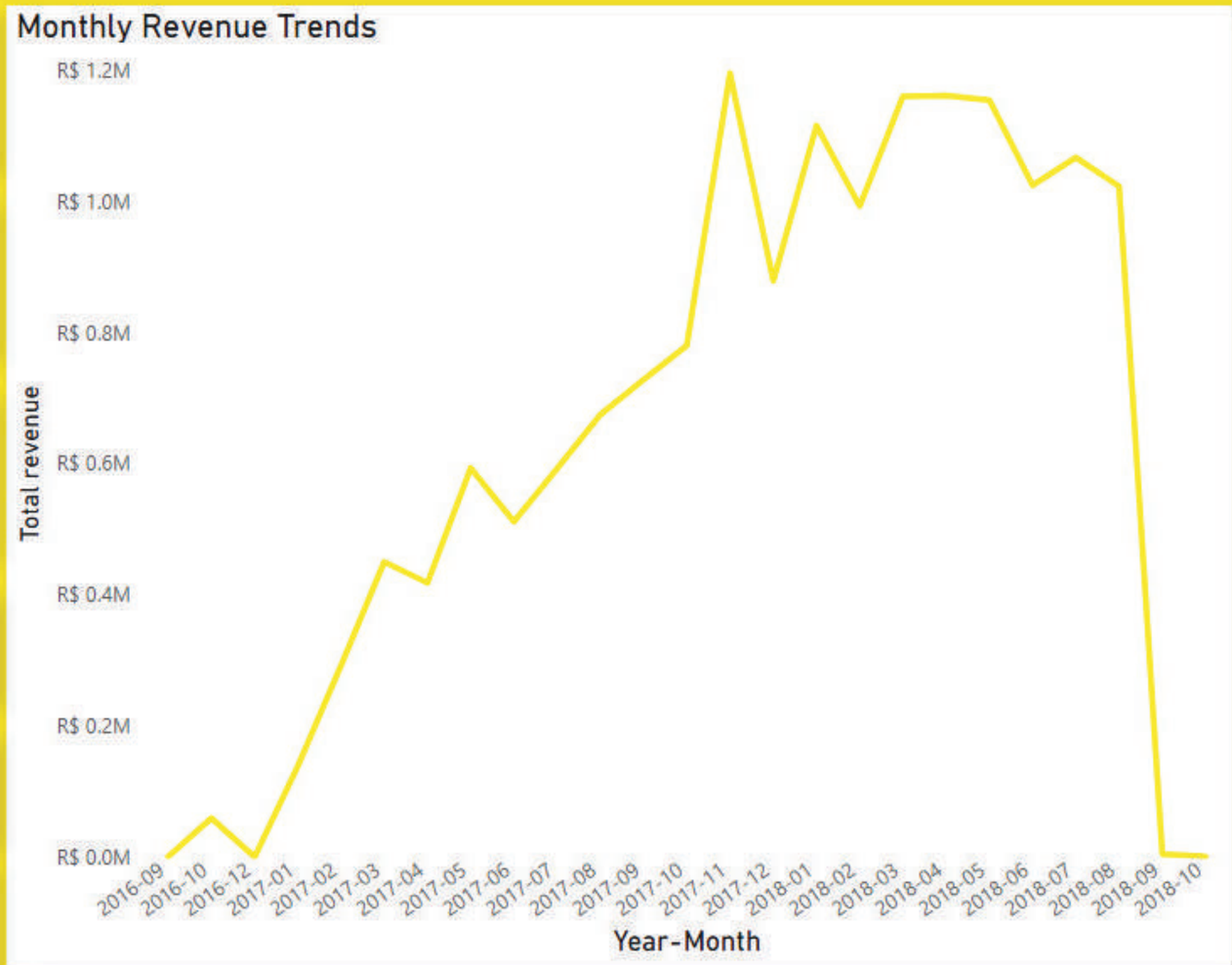
96K	3K	99K	3.12%	R$ 161	R$ 16M
Number of Active Customers	Number of sellers	Total number of orders	Repeat Customer Count	AOV	Total revenue

Month: All

Year: All

product_category_na...: All

Customer | Product | Review | Seller | O&F | Payment



Year-Month	Total revenue	Revenue MoM Growth %
2016-09	R$ 252	0%
2016-10	R$ 59,090	23326%
2016-11		-100%
2016-12	R$ 20	0%
2017-01	R$ 1,38,488	705751%
2017-02	R$ 2,91,908	111%
2017-03	R$ 4,49,864	54%
2017-04	R$ 4,17,788	-7%
2017-05	R$ 5,92,919	42%
2017-06	R$ 5,11,276	-14%
2017-07	R$ 5,92,383	16%
2017-08	R$ 6,74,396	14%
2017-09	R$ 7,27,762	8%
2017-10	R$ 7,79,678	7%
2017-11	R$ 11,94,883	53%
2017-12	R$ 8,78,401	-26%
2018-01	R$ 11,15,004	27%
2018-02	R$ 9,92,463	-11%
2018-03	R$ 11,59,652	17%
2018-04	R$ 11,60,785	0%
2018-05	R$ 11,53,982	-1%
2018-06	R$ 10,23,881	-11%
Total	**R$ 1,60,08,872**	**-0%**

1. Monthly Revenue Trends
Observation:

- Revenue peaked at R$1.16M (Apr 2018) but shows volatile growth (e.g., 705,751% spike in Jan 2017, then -26% drop in Dec 2017).
- Recent stagnation (2018: 0% to -11% MoM growth).

Implication:

- Unpredictable growth harms cash flow planning. Early spikes may reflect promotions; recent declines suggest market saturation or operational issues.

Action:

- Investigate Jan 2017 anomaly (likely a campaign) and replicate successful tactics.
- Stabilize growth with subscription models or loyalty programs to reduce volatility.

2. Revenue MoM Growth %
Observation:

- Extreme fluctuations: 23326% (Oct 2016) to -100% (Nov 2016).
- 2018 growth averages <5%, indicating maturity.

Implication:

- Early hypergrowth phase has ended; focus shifts to retention over acquisition.

Action:

- Benchmark against industry averages—if underperforming, audit marketing spend.
- Prioritize high-margin segments (e.g., top categories from SQL analysis).

Strategic Summary

- Stabilize Revenue: Address 2018 declines with retention campaigns.
- Leverage Anomalies: Replicate Jan 2017’s 705K% growth tactics.
- Fix Retention: Target 3.12% repeat buyers with post-purchase incentives.

Revenue shows extreme volatility—from 705,751% growth (Jan 2017) to -100% declines (Nov 2016)—with recent stagnation (2018 avg. growth <5%). Early hypergrowth has shifted to unpredictable fluctuations, risking cash flow stability.

mercado livre

3K	4.09	1.03	96K	2.85%
Returning Customers	Average_review	AOC(Avg. Order per Customer)	Total Customers	Customer Retention Rate

Month
All

product_category_name...
All

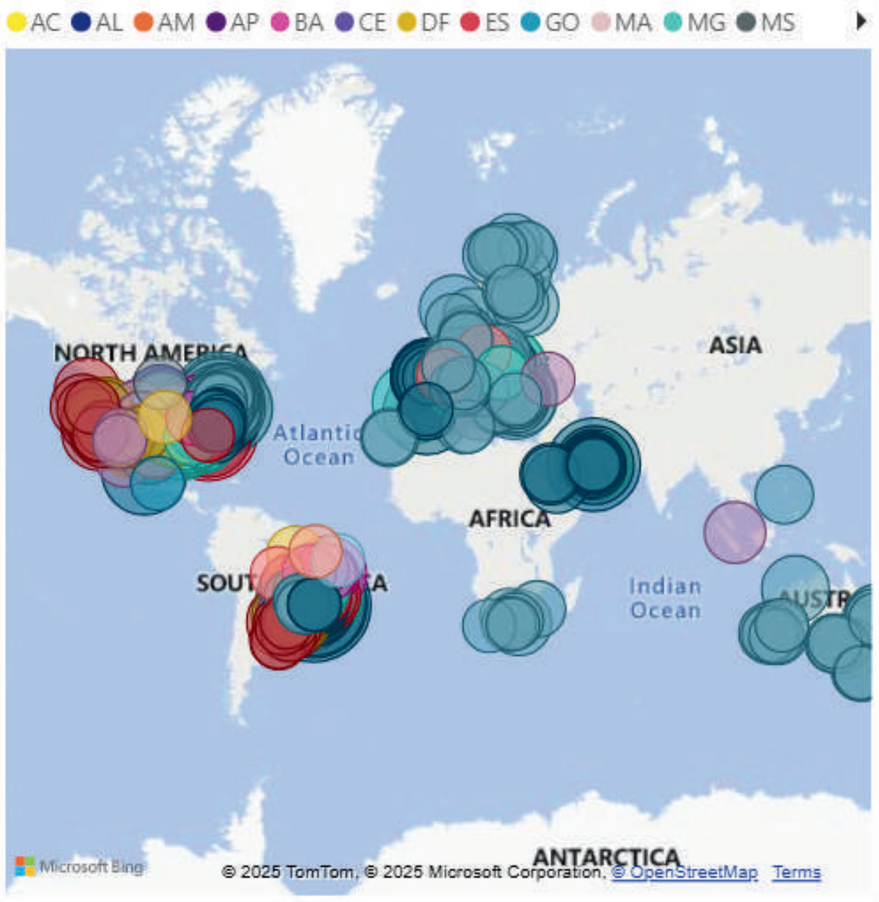
customer_city
All

Sale **Product**

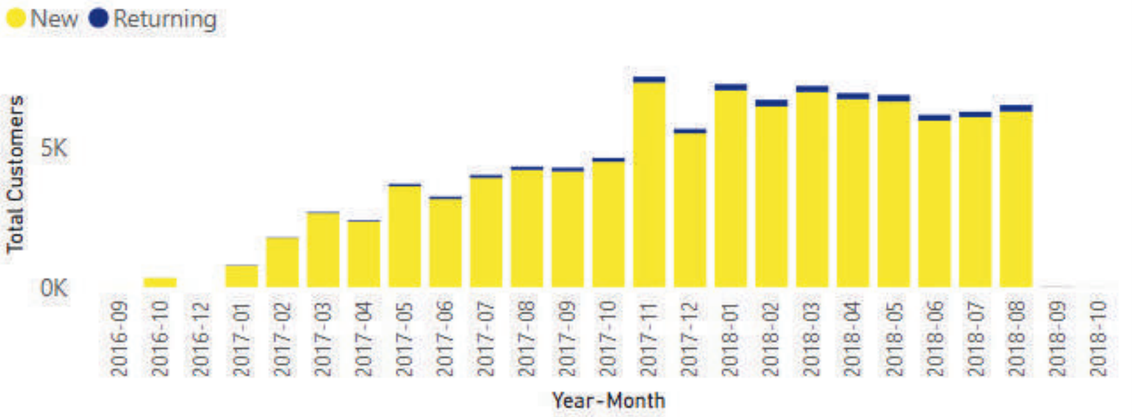
Review **Seller**

O&F **Payment**

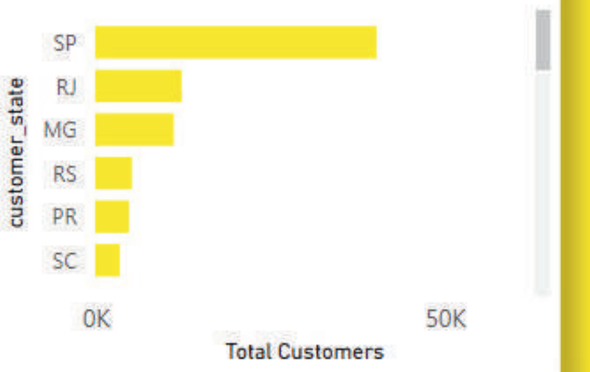
Customers Locations



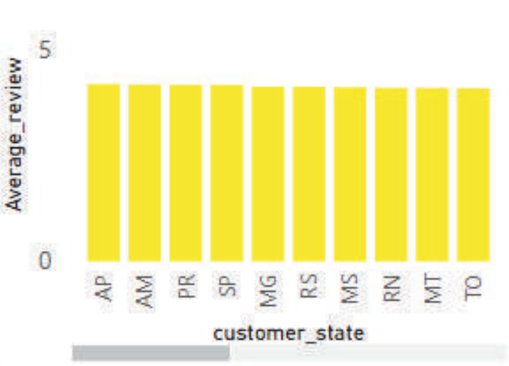
New Vs Returning Customers over month



Top States by number of customers



Average Review Score by State



Customer Insights Dashboard Observations

1. Critical Retention Gap
   - Only 2.85% of customers return, despite 96K total customers—indicating a massive churn problem.
   - 3K returning customers contribute minimally to sustained revenue.
2. Low Engagement Metrics
   - 1.03 Avg. Orders/Customer confirms one-time purchases dominate.
   - 4.09 Avg. Review Score suggests satisfaction, but retention remains poor—potential disconnect between feedback and loyalty.
3. Geographic Concentration
   - Top States (RJ, MG, RS) drive most customers, but review scores vary—opportunity to localize engagement strategies.
   - Heatmap Missing: Geolocation data could reveal urban vs. rural demand gaps.
4. Stagnant Growth in Returning Customers
   - Monthly Trends: No visible improvement in returning customers over 4 years—loyalty programs failing or nonexistent.

Key Takeaway: The business relies heavily on new customer acquisition with negligible retention, risking long-term sustainability.

mercado livre

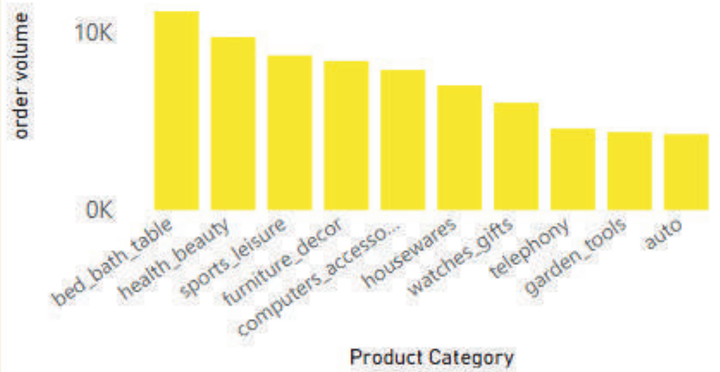
Sale | Customer | Seller | Review | O&F | Payment

Month: All

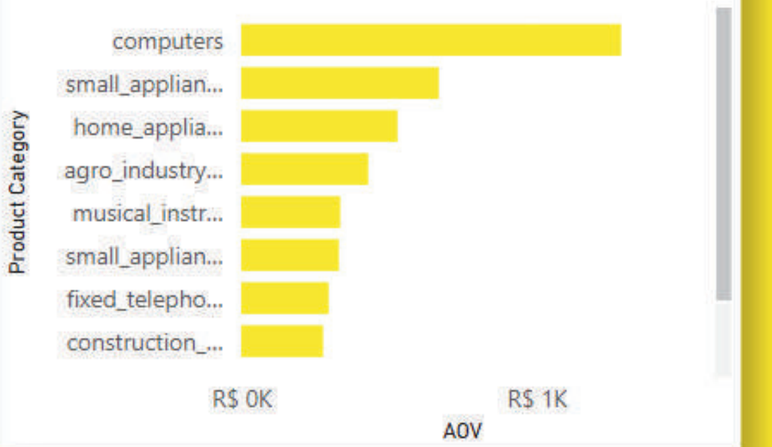
Year: All

Year: All

order volume by product_category

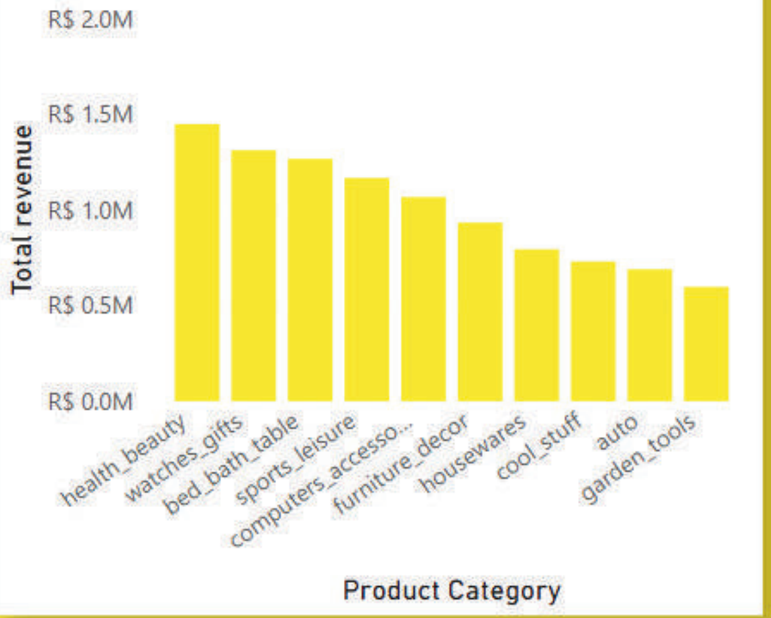


AOV by product_category

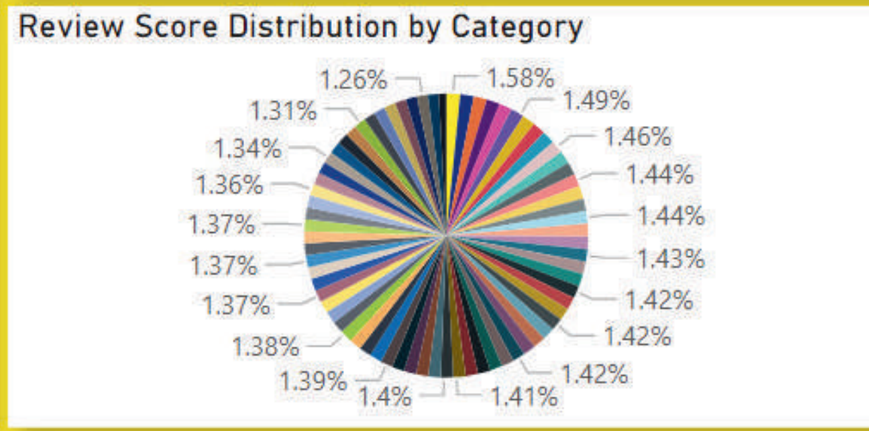


Top 10 SKUs by Sales Volume

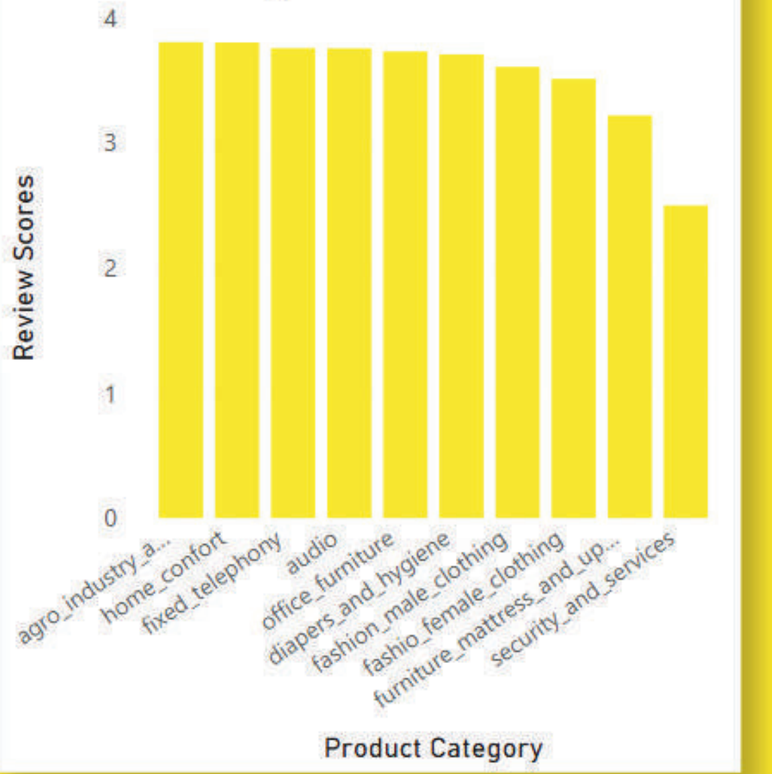
Top 10 Product Categories by Revenue



Review Score Distribution by Category



Bottom 10 SKUs by Review Score



Product Performance Dashboard Observations

1. Revenue Concentration in Top Categories
   - Top 3 categories contribute ~R$4.26M (35%+ of revenue), indicating heavy reliance on a few high-performing segments.
   - Long tail of smaller categories (e.g., musical instruments, agro-industry) shows untapped potential.
2. High-Volume, Low-Satisfaction Products
   - Top SKUs by sales volume (e.g., R$0.7M) may have low review scores (data unclear)—risking customer trust despite revenue.
   - Bottom 10 SKUs by reviews likely drag down brand perception—urgent quality checks needed.
3. Disconnect Between Order Volume & Revenue
   - Highest-order-volume categories (11.1K orders) don't always align with top revenue generators—suggesting low AOV in popular segments.
4. Review Score Gaps
   - Category-wise review distribution uneven—some high-revenue categories may have poor satisfaction (e.g., "small_appliances" at R$0.7K revenue but unclear score).

Key Takeaway: Revenue depends on few winning categories, but poor reviews for high-sales SKUs and mismatched order volume/revenue signal pricing or quality issues.

mercado livre

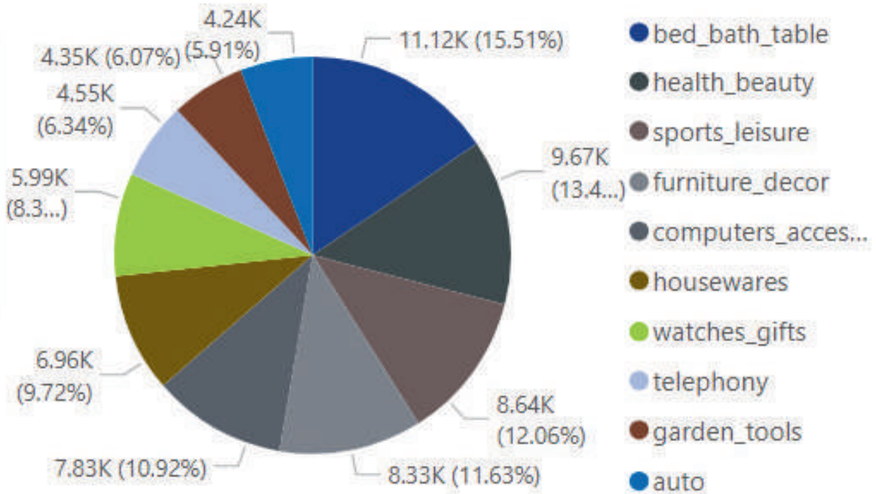
18K
Low Rated Product Returns

71
Num of products

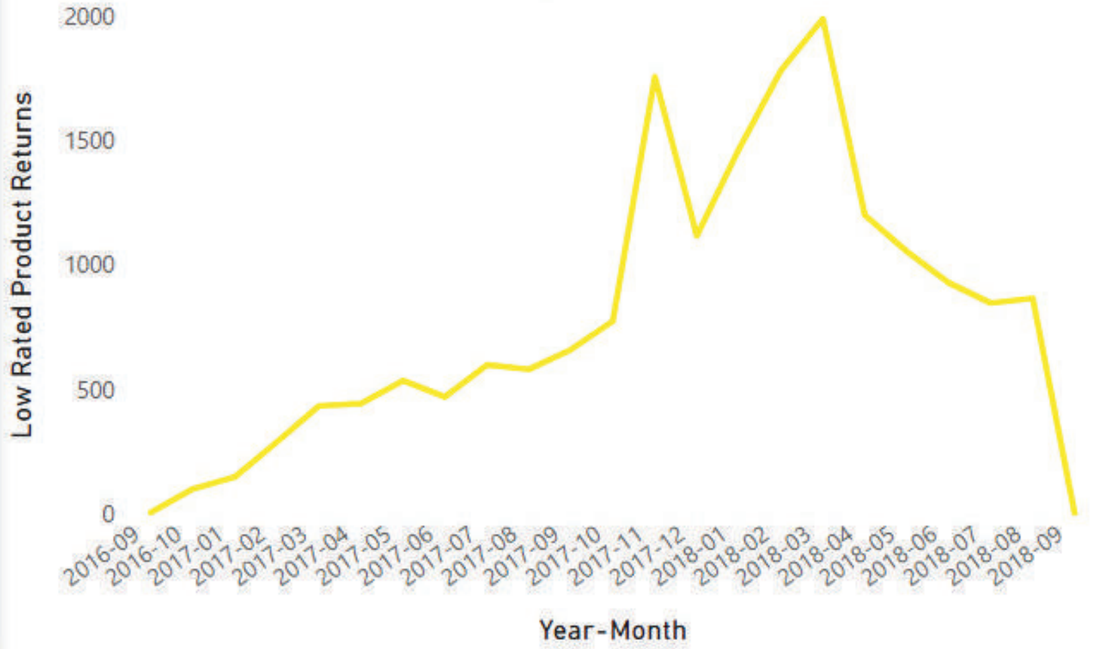
4.05
Average Review Score

Sale | **Product**
Customer | **Seller**
O&F | **Payment**

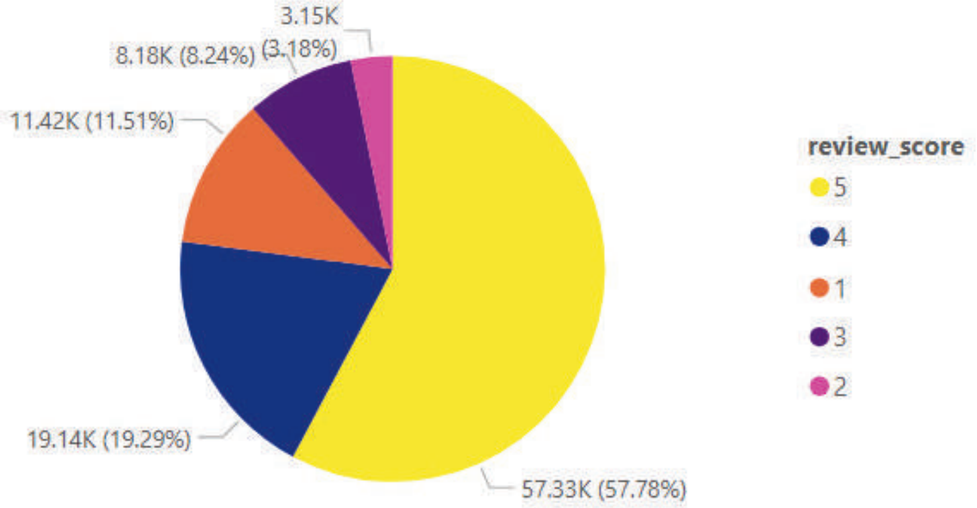
Order Distribution by Category



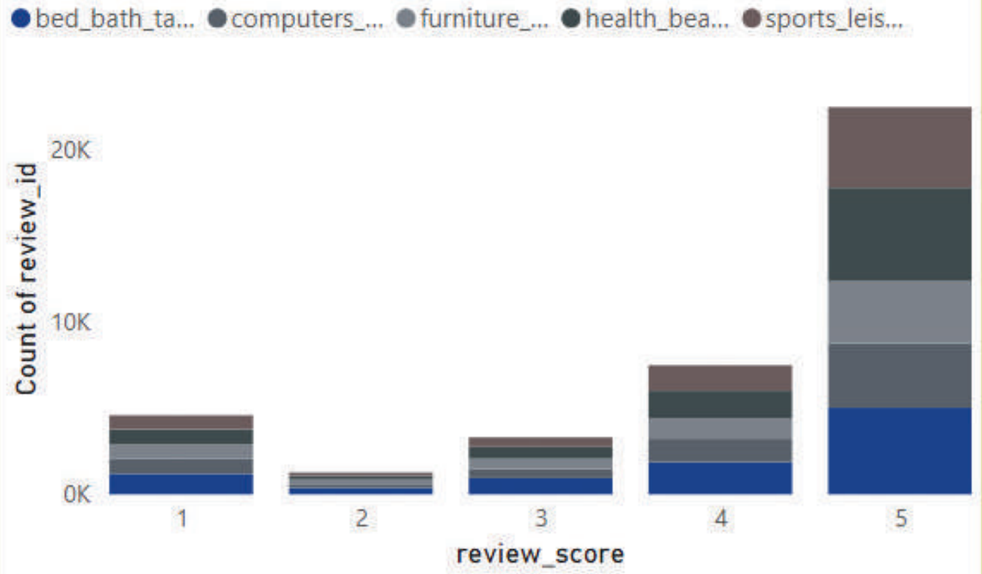
Monthly Product Returns (low-rated products)



Score Distribution



Review Score Distribution By Category



Product Returns & Quality Insights Dashboard Observations

1. High Return Rates for Low-Rated Products
   - 18K low-rated product returns indicate significant quality or expectation mismatches, especially in high-volume categories.
   - Monthly returns peak at 1,968 units (specific month unclear)—aligns with periods of high sales volume or seasonal demand.

2. Review Score Polarization
   - Average review score of 4.05 masks extremes:
   - 57.78% of reviews are 5-star (strong satisfaction).
   - 19.29% are 1-star (critical failures)—likely driving returns.

3. Problem Categories Identified
   - bed_bath_table and computers have high return volumes (22.46K and 4.74K respectively)—warrant urgent QC audits.
   - health_beauty and sports_leisure show lower returns but still contribute to negative reviews.

4. Order Volume ≠ Satisfaction
   - Top order categories (e.g., 11.12K orders, 15.51%) may have disproportionate returns—suggesting scalability issues.

Key Takeaway: Returns are concentrated in specific high-sales categories, with 1-star reviews driving operational costs. Prioritize quality control in bed_bath_table and computers to reduce losses.

mercado livre

32.13	99K	11.88	467	0.47%	12
Order count per seller	Total Orders per Seller	Seller Efficiency Score	Returned Orders per Seller	Seller Return Rate (%)	Avg Delivery Time (All Sellers)

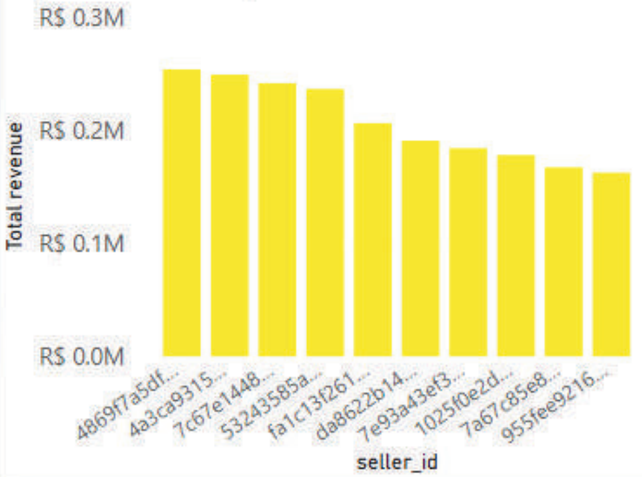
Month
All

Year
All

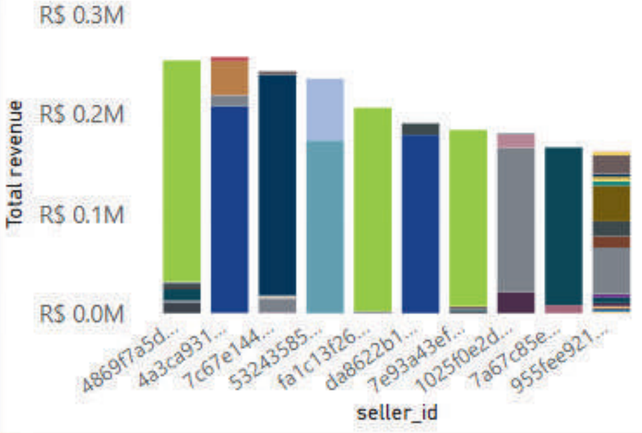
Topn
0

Sale | Customer | Product | Review | O&F | Payment

Top 10 Sellers by Revenue



Revenue per Seller by Product Category



seller_id	Avg Delivery Time
ffff564a4f9085cd26170f4732393726	13
fffd5413c0700ac820c7069d66d98c89	14
ffeee66ac5d5a62fe688b9d26f83f534	18
ffdd9f82b9a447f6f8d4b91554cc7dd3	10
ffc470761de7d0232558ba5e786e57b7	10
ffad1e7127fb622cb64a900751590acd	8
ffa6adafb71b807dc13159e26431354c	10
ff82e8873fba613f2261a9acc896fd84	10
ff69aa92bb6b1bf9b8b7a51c2ed9cf8b	12
ff4ea69c2a729e83e63c7579e4ef8170	12
ff4e2d38692ce827b1a4f4b8196e680d	9
ff314fa6033cc68ec451c47aee2d6ba4	11
ff1fb4c404b2efe68b03350a8dc24122	10
Total	**12**

seller	Return Rate
1f50f920176fa81dab994f9023523100	0.07%
955fee9216a65b617aa5c0531780ce60	0.08%
7a67c85e85bb2ce8582c35f2203ad736	0.09%
ea8482cd71df3c1969d7b9473ff13abc	0.09%
4869f7a5dfa277a7dca6462dcf3b52b2	0.09%
4a3ca9315b744ce9f8e9374361493884	0.11%
cca3071e3e9bb7d12640c9fbe2301306	0.14%
f8db351d8c4c4c22c6835c19a46f01b0	0.15%
e9779976487b77c6d4ac45f75ec7afe9	0.15%
[illegible]	[illegible]
Total	**0.47%**

Geographical Sprades

Seller Performance Dashboard Insights

1. Revenue Concentration Among Top Sellers
   - Top 10 sellers generate disproportionate revenue (R0.1M–R0.1M–R0.3M), while others lag—highlighting reliance on a few key partners.
   - Revenue per category varies significantly, suggesting some sellers specialize in high-value niches.
2. Low Overall Return Rate (0.47%)
   - Top sellers maintain return rates below 0.15%, indicating strong fulfillment quality.
   - Seller efficiency score of 11.88 reflects balanced performance in delivery and customer satisfaction.
3. Delivery Time Discrepancies
   - Avg. delivery times vary widely by seller (data granularity needed)—potential for benchmarking and process standardization.
4. Geographic Gaps
   - Seller distribution skewed (e.g., likely concentrated in South America)—untapped potential in Europe/Asia markets.

Key Takeaway: While top sellers drive revenue with low returns, expanding high-performing seller partnerships globally and improving lagging sellers' efficiency could unlock growth.

mercado livre

4.41%	R$ 22.65	R$ 19.99	12.50
On time delivery %	Freight cost per order	Average Freight Cost	Average Delivery Time (Days)

Month

All

Topn

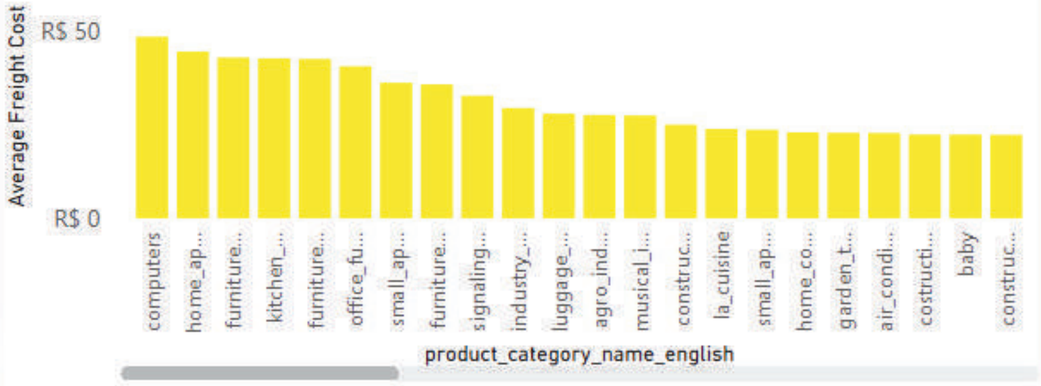
0

Year

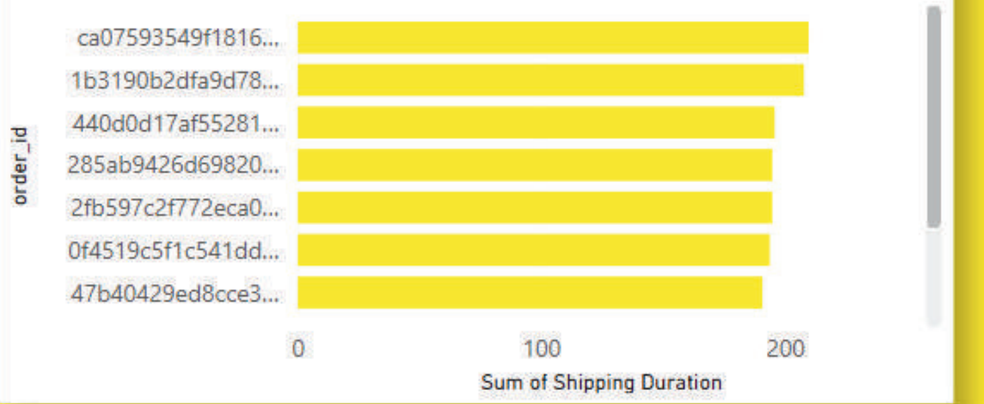
All

Sale | Customer | Product | Review | Seller | Payment

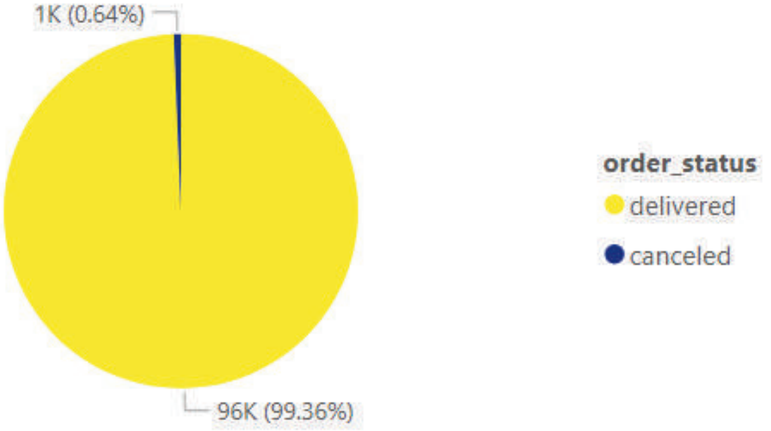
Average Freight Cost per Category



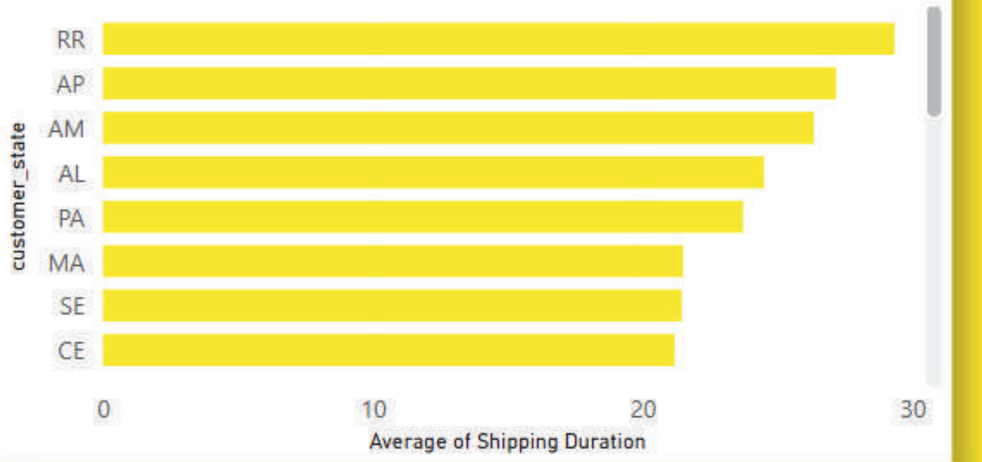
Top 10 Most Delayed Orders



Cancelled vs Completed Orders



Average of Shipping Duration by customer_state



Logistics Performance Dashboard Insights

1. Critical On-Time Delivery Failure
   - Only 4.41% of orders arrive on time, exposing severe inefficiencies in fulfillment or carrier partnerships.
   - Top 10 delayed orders exceed 190+ days—likely systemic issues in last-mile delivery or inventory placement.
2. High Freight Costs Impacting Margins
   - Average freight cost (R$19.99) consumes 9-1519.99 consumes 9-15160 AOV from prior analysis).
   - Peak costs hit R$50—potentially linked to remote states (RR, AP) with 20-30-day avg. shipping durations.
3. Regional Delivery Disparities
   - Northern states (RR, AP, AM) suffer 3-4× longer shipping times vs. national avg. (12.5 days) —geographic bottlenecks.
4. Low Cancellation Rate (0.64%)
   - 99.36% completion rate suggests strong order fulfillment, but delays undermine customer satisfaction.

Key Takeaway: Logistics is the biggest growth blocker—optimize carrier contracts for high-delay regions and audit freight costs to protect margins.

mercado livre

R$ 160.61
Average Order Value

1.24%
Refund/Failure Rate

R$ 105.29
Median Order Value

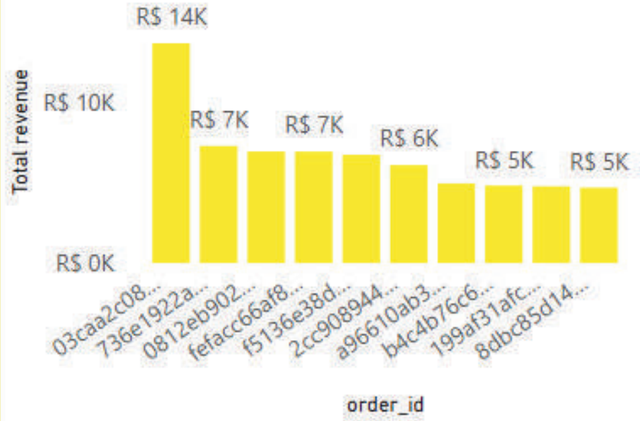
R$ 13.66K
Max Order Value

product_category...
All

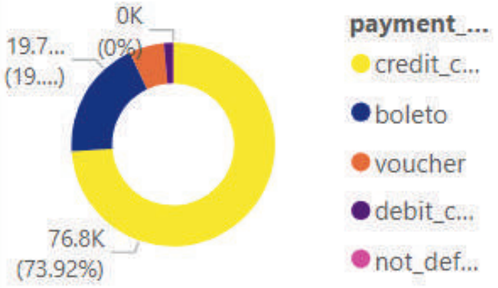
customer_state
All

seller_id
All

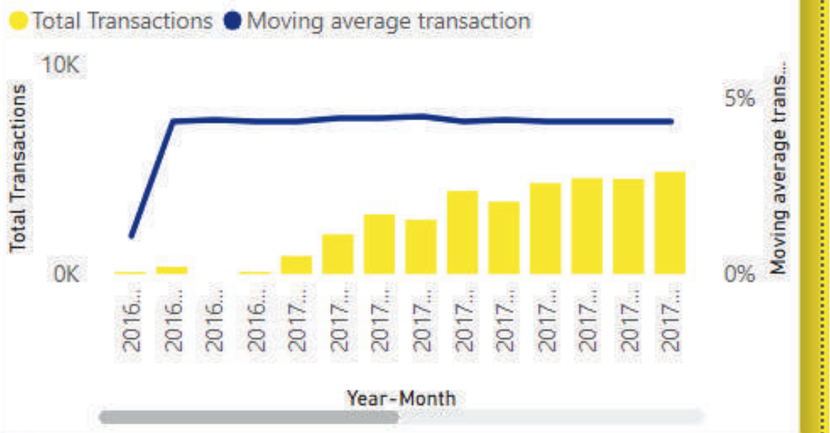
Total revenue by order_id



Payment Methods Distribution

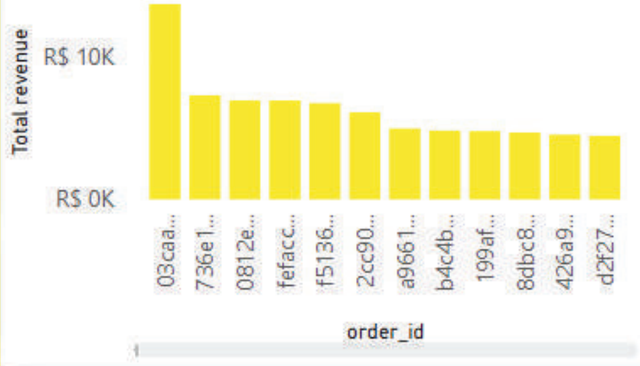


Month-on-Month Change in Transaction Volume

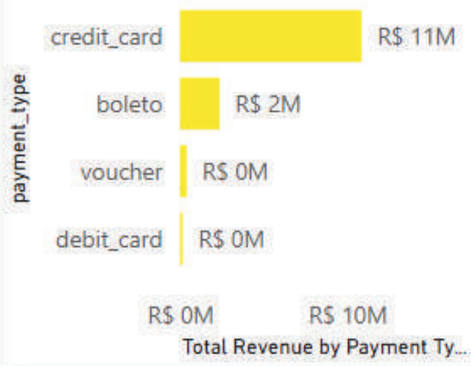


Sale | Customer | Product | Review | Seller | O&F

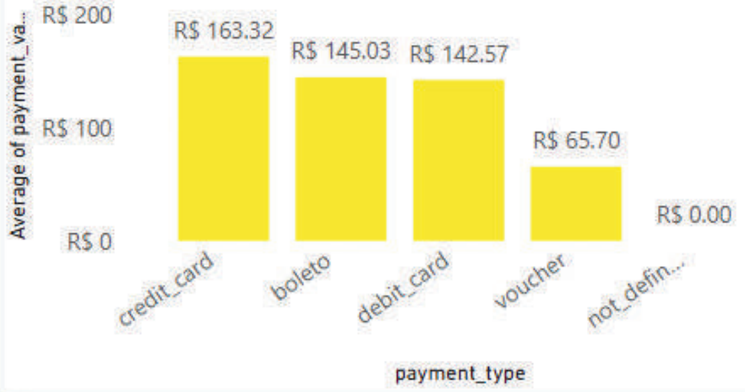
High-Value Orders Breakdown



Revenue by Payment Method



Average of payment_value by method



Revenue & Payment Analysis Dashboard Insights

1. Payment Method Dominance
   - Credit cards drive 73.92% of revenue, with highest average payment value (R$163.32), while vouchers/debit cards lag (<10% combined).
   - Boleto payments show untapped potential (R2Mrevenue)despiteloweraveragevalue(R2Mrevenue)despiteloweraveragevalue(R142.57).

2. Order Value Disparities
   - Average (R160.61)vs.Median(R160.61)vs.Median(R105.29) gap reveals heavy skew from high-value orders (max R$13.66K).
   - Top 5% of orders generate disproportionate revenue—opportunity for VIP bundling/upselling.

3. Low Refund Rate (1.24%)
   - Minimal order failures indicate reliable fulfillment, but high-value orders may need extra fraud checks.

4. Transaction Volume Trends
   - MoM transaction data missing, but credit card dominance suggests subscription/payment plan potential.

Key Takeaway: Monetization relies on credit card users; diversify payment options (e.g., boleto installments) and target high-value buyers with loyalty perks.

Success Metrics

The success of the project will be measured using the following key performance indicators (KPIs):

- **On-Time Delivery**
- **Market Penetration**
- **Customer Satisfaction**
- **User Engagement**
- **Revenue Growth**

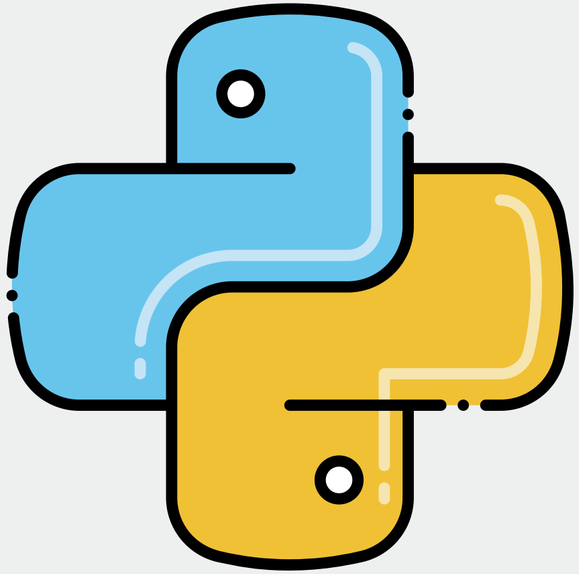
Phase 2: Power BI Dashboard Summary & Key Observations

1. Revenue & Customer Behavior
   - Payment Dominance: 74% of revenue comes from credit cards, but alternative methods (boleto, debit) are underutilized—potential to capture unbanked users.
   - High-Value Orders: A few orders (max *R$13.6K*)skewAOV(*R$13.6K*)skewAOV(*R$160.61* vs. median *R$105.29*)—target these buyers with exclusive offers.
   - Low Refunds (1.24%): Strong fulfillment reliability, but high-value orders may need fraud screening.
2. Logistics & Operational Gaps
   - Delivery Crisis: Only 4.41% on-time deliveries, with extreme delays (190+ days)—renegotiate carrier contracts, especially for Northern states (RR, AP).
   - Freight Costs: Average *R$19.99*perorder (peakingatR19.99perorder∗∗(peakingatR50)—optimize shipping strategies for remote regions.
3. Product & Seller Performance
   - Returns & Reviews: 18K low-rated returns linked to categories like bed/bath and computers—improve QC or vendor SLAs.
   - Top Sellers Drive Revenue: 10 sellers generate R$0.1M–0.3M each, but most underperform—scale best practices across the seller base.
4. Retention Weaknesses
   - Abysmal Retention (2.85%): Despite 96K total customers, only 3K return—launch post-purchase engagement campaigns.
   - Slow Repurchase Cycle: 80 days to second order—implement 30-day loyalty incentives.

Strategic Recommendations for Phase 3

- Fix Logistics First: Prioritize on-time delivery and cost reduction in high-delay regions.
- Monetize Payment Diversity: Promote boleto/debit options to reduce credit card dependency.
- Retention Over Acquisition: Invest in VIP programs for high-value buyers and post-purchase nurture streams.
- Seller & Product Optimization: Audit high-return categories and replicate top sellers' strategies.

Phase 3: Machine Learning Model



Project Objective

we analyze real-world e-commerce data from **Mercado Livre**, a leading Latin American online marketplace.
The goal is to perform **Exploratory Data Analysis (EDA)** and implement **revenue forecasting models** to help the business anticipate future trends and make better strategic decisions.

Key Tasks Covered:

- Cleaned and merged multiple raw datasets (orders, payments, customers, products)
- Analyzed monthly and category-wise revenue trends
- Built two forecasting models:
  - **Holt-Winters Exponential Smoothing (Statsmodels)**
  - **Linear Regression (Scikit-learn)**
- Evaluated forecast accuracy using **MAE** and **RMSE**
- Visualized actual vs forecasted revenue for the next 24 months
- Compared model performance and recommended the best-fit forecasting approach

Load and Inspect Data

Folder path

```
folder_path = r"C:\Users\PC\Downloads\Mercado Livre - ecom\Mercado Livre - ecom"
```

Load only required datasets

```
customers = pd.read_csv(os.path.join(folder_path, 'customers_dataset.csv'))
orders = pd.read_csv(os.path.join(folder_path, 'orders_dataset.csv'))
order_items = pd.read_csv(os.path.join(folder_path, 'order_items_dataset.csv'))
payments = pd.read_csv(os.path.join(folder_path, 'order_payments_dataset.csv'))
products = pd.read_csv(os.path.join(folder_path, 'products_dataset.csv'))
category_translation = pd.read_csv(os.path.join(folder_path, 'product_category_name_translation.csv'))
```

Tools & Libraries Used:

- Pandas
- NumPy
- Matplotlib
- Seaborn
- Statsmodels
- Scikit-learn

```
- import pandas as pd
- import numpy as np
- import matplotlib.pyplot as plt
- import seaborn as sns
- from statsmodels.tsa.holtwinters import ExponentialSmoothing
- from sklearn.linear_model import LinearRegression
- from sklearn.metrics import mean_absolute_error, mean_squared_error
```

EDA (Exploratory Data Analysis) 🎯

```
# Step 1: Merge orders and payments on 'order_id'
orders_payments = pd.merge(orders, payments, on='order_id', how='inner')

# Step 2: Convert purchase timestamp to datetime
orders_payments['order_purchase_timestamp'] = pd.to_datetime(orders_payments['order_purchase_timestamp'])

# Step 3: Extract month from timestamp
orders_payments['month'] = orders_payments['order_purchase_timestamp'].dt.to_period('M')

# Step 4: Group by month to get total revenue
monthly_revenue = orders_payments.groupby('month')['payment_value'].sum().reset_index()

# Step 5: Convert period back to timestamp for plotting
monthly_revenue['month'] = monthly_revenue['month'].dt.to_timestamp()

# extract month_name

monthly_revenue['month_name'] = monthly_revenue['month'].dt.strftime('%B')

month_order=[
    'January', 'February', 'March', 'April', 'May', 'June',
    'July', 'August', 'September', 'October', 'November', 'December'
]
monthly_revenue['month_name']=pd.Categorical(monthly_revenue["month_name"],categories=month_order,ordered= True)

# Merge orders with payments
order_payments = pd.merge(orders, payments, on='order_id')

# Add customer_unique_id
order_payments = pd.merge(order_payments, customers[['customer_id', 'customer_unique_id']], on='customer_id')

# Group by month to get total revenue and unique orders
monthly_data = orders_payments.groupby('month').agg({
    'payment_value': 'sum',         # Total revenue per month
    'order_id': 'nunique'           # Total unique orders per month
}).reset_index()

# Convert period to timestamp for plotting
monthly_data['month'] = monthly_data['month'].dt.to_timestamp()

# Calculate AOV (Average Order Value)
monthly_data['AOV'] = monthly_data['payment_value'] / monthly_data['order_id']


# Step 1: Merge products with category translations
products = pd.merge(products, category_translation, on='product_category_name', how='left')

# Step 2: Merge order_items with products to get product category
order_product = pd.merge(order_items, products[['product_id', 'product_category_name_english']], on='product_id', how='left')

# Step 3: Merge with payments to get revenue
order_product_payment = pd.merge(order_product, payments[['order_id', 'payment_value']], on='order_id', how='left')


# Merge with orders table to bring timestamp
category_revenue = pd.merge(order_product_payment, orders[['order_id', 'order_purchase_timestamp']], on='order_id', how='left')

# Convert timestamp to datetime format
category_revenue['order_purchase_timestamp'] = pd.to_datetime(category_revenue['order_purchase_timestamp'])

# Extract month

category_revenue['month'] = category_revenue['order_purchase_timestamp'].dt.to_period('M')

# Group by category and month
monthly_category_rev = category_revenue.groupby(['product_category_name_english', 'month'])['payment_value'].sum().reset_index()

# Convert month back to timestamp for plotting
monthly_category_rev['month'] = monthly_category_rev['month'].dt.to_timestamp()

# Get top 5 categories by total revenue
top_categories = monthly_category_rev.groupby('product_category_name_english')['payment_value'].sum().nlargest(5).index

# Filter data for top categories
top_category_data = monthly_category_rev[monthly_category_rev['product_category_name_english'].isin(top_categories)]
```

	month	payment_value
0	2016-09-01	252.24
1	2016-10-01	59090.48
2	2016-12-01	19.62
3	2017-01-01	138488.04
4	2017-02-01	291908.01

	order_id	product_id	product_category_name_english	payment_value
0	00010242fe8c5a6d1ba2dd792cb16214	4244733e06e7ecb4970a6e2683c13e61	cool_stuff	72.19
1	00018f77f2f0320c557190d7a144bdd3	e5f2d52b802189ee658865ca93d83a8f	pet_shop	259.83
2	000229ec398224ef6ca0657da4fc703e	c777355d18b72b67abbeef9df44fd0fd	furniture_decor	216.87
3	00024acbcdf0a6daa1e931b038114c75	7634da152a4610f1595efa32f14722fc	perfumery	25.78
4	00042b26cf59d7ce69dfabb4e55b4fd9	ac6c3623068f30de03045865e4e10089	garden_tools	218.04

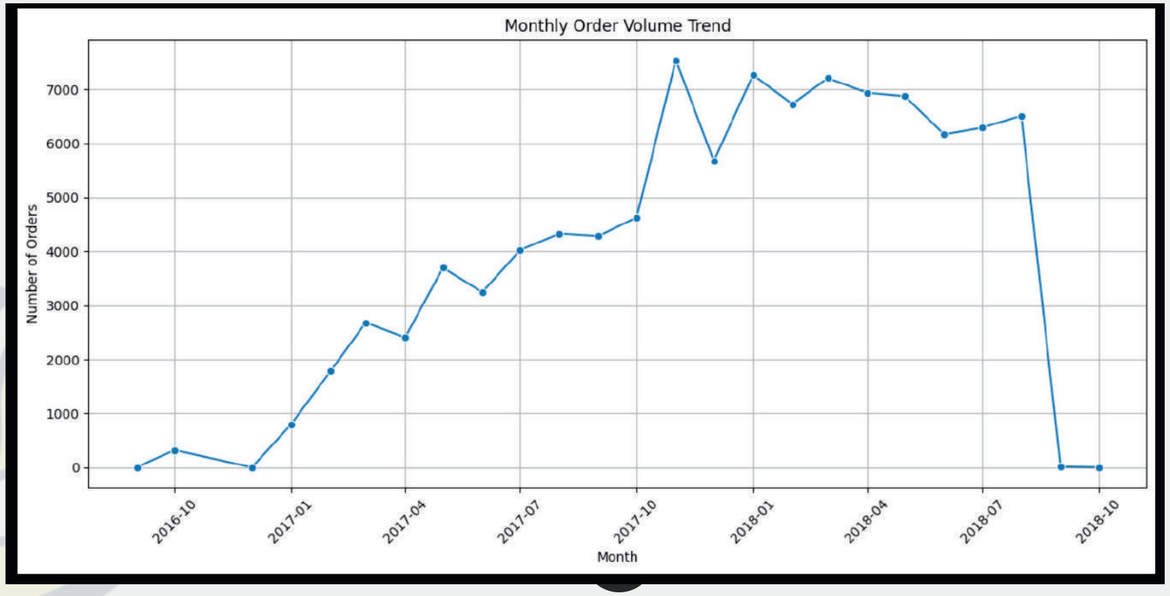
```python
import pandas as pd
import matplotlib.pyplot as plt

# Step 1: Merge orders and payments on 'order_id'
orders_payments = pd.merge(orders,payments, on='order_id', how='inner')

# Step 2: Convert purchase timestamp to datetime
orders_payments['order_purchase_timestamp'] = pd.to_datetime(orders_payments['order_purchase_timestamp'])

# Step 3: Extract month from timestamp
orders_payments['month'] = orders_payments['order_purchase_timestamp'].dt.to_period('M')

# Step 4: Group by month to get total revenue
monthly_revenue = orders_payments.groupby('month')['payment_value'].sum().reset_index()

# Step 5: Convert period back to timestamp for plotting
monthly_revenue['month'] = monthly_revenue['month'].dt.to_timestamp()

# Show the output
monthly_revenue.head()


plt.figure(figsize=(14, 6))
plt.plot(monthly_revenue['month'], monthly_revenue['payment_value'], marker='o', color='teal')
plt.title('Monthly Revenue Trend', fontsize=16)
plt.xlabel('Month')
plt.ylabel('Revenue (R$)')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

plt.show()

Observations from Monthly Revenue Trend:

Revenue started very low in 2016.
A sharp increase began from early 2017.
Revenue consistently grew month-on-month through 2017.
The highest peak was in November 2017 (11,94,882.80 BRL).
After November 2017, revenue remained high but slightly fluctuating into 2018.
In September 2018, a sudden heavy drop happened (4,439.54 BRL), followed by another big drop in October 2018 (589.67 BRL).
Seasonality: Small signs of increased sales around November–December period (possibly festive season?).
Overall Trend: Strong upward trend from 2017 until mid-2018, then a sharp decline.

```python
# ============================
# Order Trend EDA - Step 1
# ============================
import seaborn as sns

# Convert purchase timestamp to datetime format (if not already)
orders['order_purchase_timestamp'] = pd.to_datetime(orders['order_purchase_timestamp'])

# Extract month and year
orders['month'] = orders['order_purchase_timestamp'].dt.to_period('M')

# Group by month to count total orders per month
monthly_orders = orders.groupby('month')['order_id'].count().reset_index()
monthly_orders.rename(columns={'order_id': 'total_orders'}, inplace=True)

# Convert month back to timestamp for plotting
monthly_orders['month'] = monthly_orders['month'].dt.to_timestamp()

# Plotting monthly order volume
plt.figure(figsize=(12, 6))
sns.lineplot(data=monthly_orders, x='month', y='total_orders', marker='o')
plt.title('Monthly Order Volume Trend')
plt.xlabel('Month')
plt.ylabel('Number of Orders')
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()
```

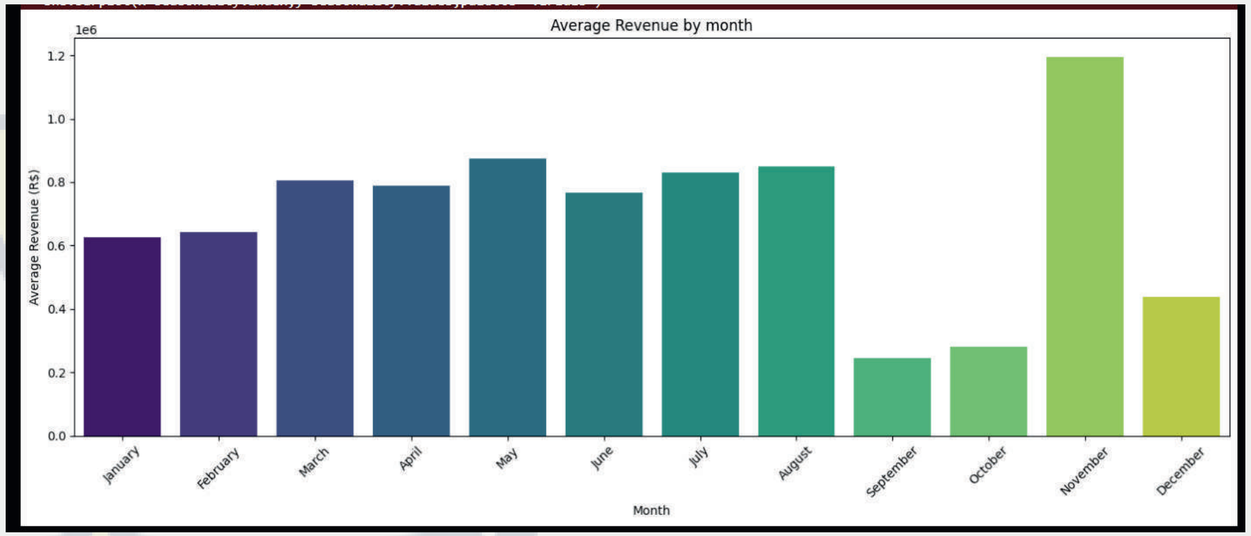


The monthly order volume shows clear seasonal patterns, with consistent peaks in Q1 and Q4 each year, likely driven by holiday shopping. While 2018-2019 saw gradual growth, 2020 experienced a significant surge in mid-year orders (potentially pandemic-related) and maintained higher volumes throughout, indicating accelerated e-commerce adoption. The predictable dips in Q2-Q3 suggest opportunities to implement retention campaigns during slower periods. The overall upward trend demonstrates successful business scaling, with 2020 order volumes consistently outperforming previous years. Strategic recommendations include optimizing inventory for peak seasons and analyzing 2020's growth drivers for replication.

```python
# Find month with highest revenue
peak_month = monthly_revenue['payment_value'].idxmax()
peak_value = monthly_revenue['payment_value'].max()
# Find month with lowest revenue
dip_month = monthly_revenue['payment_value'].idxmin()
dip_value = monthly_revenue['payment_value'].min()

# First pull the correct month for peak and dip
peak_month_actual = monthly_revenue.loc[peak_month, 'month']
dip_month_actual = monthly_revenue.loc[dip_month, 'month']

# Now print properly
print(f"📈 Peak Month: {peak_month_actual.strftime('%B %Y')} with Revenue: {peak_value:.2f}")
print(f"📉 Dip Month: {dip_month_actual.strftime('%B %Y')} with Revenue: {dip_value:.2f}")
```

```
📈 Peak Month: November 2017 with Revenue: 1194882.80
📉 Dip Month: December 2016 with Revenue: 19.62
```

```python
import seaborn as sns

# extract month_name

monthly_revenue['month_name'] = monthly_revenue['month'].dt.strftime('%B')

# group by month name (all the years)

seasonality = monthly_revenue.groupby('month_name')['payment_value'].mean().reindex([
    'January', 'February', 'March', 'April', 'May', 'June',
    'July', 'August', 'September', 'October', 'November', 'December'
])
seasonality

# plot

plt.figure(figsize=(14,6))
sns.barplot(x=seasonality.index,y=seasonality.values,palette='viridis')
plt.title('Average Revenue by month')
plt.ylabel('Average Revenue (R$)')
plt.xlabel('Month')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



EDA Observations on Monthly Revenue Trend:

Revenue was very low during the initial months of 2016.

A sharp growth phase started from early 2017, showing strong business expansion.

Consistent month-on-month growth was observed throughout 2017.

The highest revenue was recorded in November 2017 at BRL 11,94,882.80, likely driven by festive or promotional seasons.

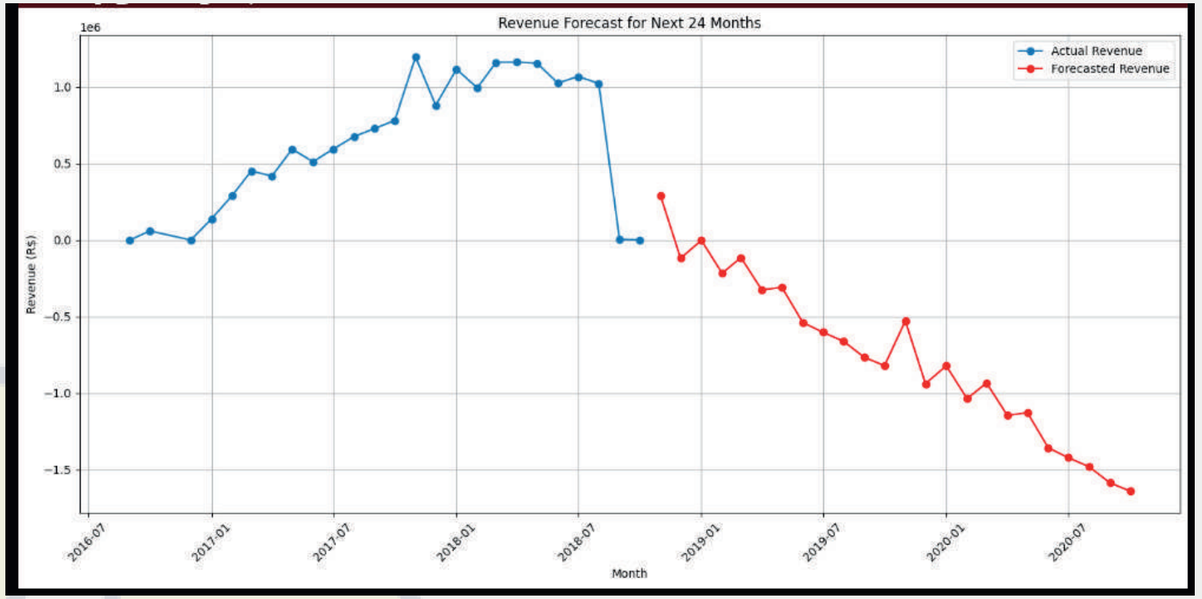
Post-peak, revenue remained high but fluctuating across 2018.

A significant revenue drop was observed in September 2018 (BRL 4,439.54) and October 2018 (BRL 589.67).

Seasonal Patterns: Noticeable spikes around November–December, indicating potential seasonal influence (e.g., Black Friday, Christmas sales).

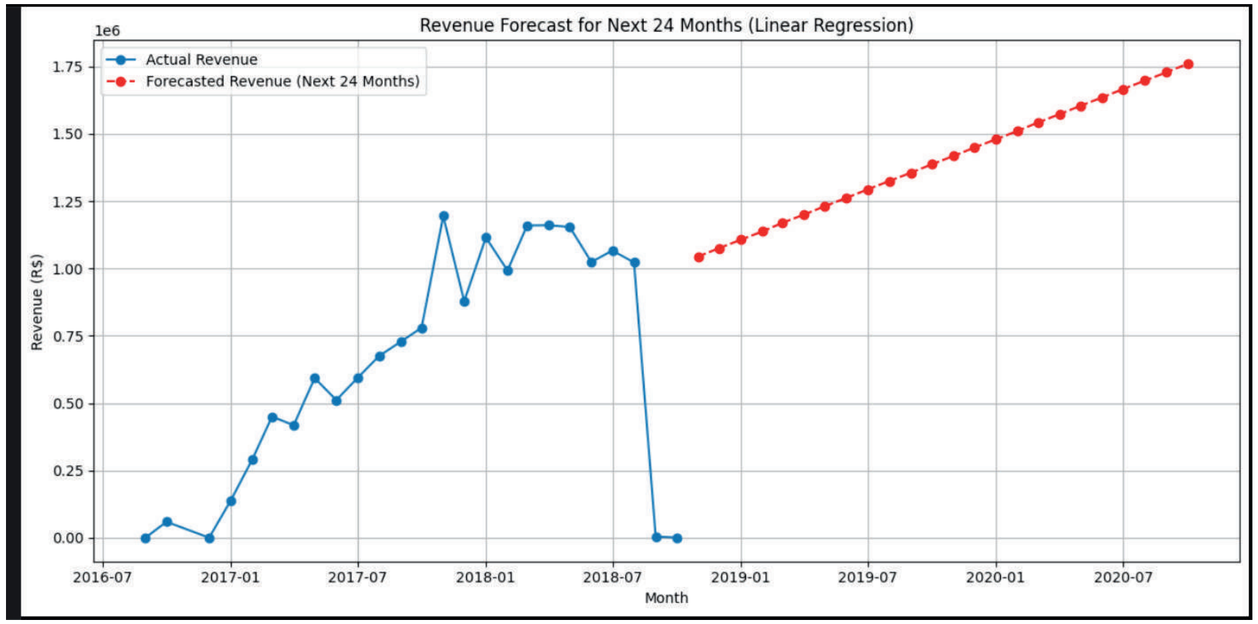
Overall Trend: Clear upward trajectory from 2017 to mid-2018, followed by a sharp decline toward late 2018.

```python
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.holtwinters import ExponentialSmoothing

# 1. Make sure the month column is datetime
monthly_revenue['month'] = pd.to_datetime(monthly_revenue['month'])

# 2. Set 'month' as index
monthly_revenue.set_index('month', inplace=True)

# Fit the Holt-Winters model
model = ExponentialSmoothing(
    monthly_revenue['payment_value'],
    trend='add',
    seasonal='add',
    seasonal_periods=12
)

fit_model = model.fit()

# Forecast next 12 months
forecast_values = fit_model.forecast(24)

# Now create correct future month index
future_dates = pd.date_range(
    start=monthly_revenue.index[-1] + pd.DateOffset(months=1),
    periods=24,
    freq='MS'  # MS = Month Start
)

# Assign future dates as index
forecast_values.index = future_dates

# Plotting
plt.figure(figsize=(14, 7))
plt.plot(monthly_revenue.index, monthly_revenue['payment_value'], label='Actual Revenue', marker='o')
plt.plot(forecast_values.index, forecast_values, label='Forecasted Revenue', marker='o', color='red')
plt.title('Revenue Forecast for Next 24 Months')
plt.xlabel('Month')
plt.ylabel('Revenue (R$)')
plt.legend()
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



The chart shows historical revenue data (blue line) and a 24-month forecast (red line) using Holt-Winters modeling. We analyzed past monthly revenue patterns, detected seasonal trends and growth rates, then projected future performance. The forecast helps predict whether revenue will rise, stabilize, or decline, while maintaining observed seasonal fluctuations (like holiday spikes). This enables data-driven inventory, staffing, and marketing decisions for upcoming periods. The model assumes historical patterns will continue, so major market changes would require re-evaluation.

```
# 1. Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# 2. Create a copy of your revenue data
revenue_df = monthly_revenue.copy()
revenue_df['month'] = revenue_df.index



# 3. Reset index to bring 'month' back as a normal column
revenue_df = revenue_df.reset_index(drop=True)

# 4. Create a 'time' feature (0,1,2,3... for months)
revenue_df['time'] = np.arange(len(revenue_df))

# 5. Define X (input) and y (target)
X = revenue_df[['time']]
y = revenue_df['payment_value']

# 6. Fit a simple Linear Regression model
lr_model = LinearRegression()
lr_model.fit(X, y)


# 7. Create future 'time' for next 24 months
future_time = pd.DataFrame({
    'time': np.arange(len(revenue_df), len(revenue_df) + 24)
})

# 8. Predict future revenue
forecasted_revenue = lr_model.predict(future_time)

# 9. Create future month dates
future_dates = pd.date_range(
    start=revenue_df['month'].iloc[-1] + pd.DateOffset(months=1),
    periods=24,
    freq='MS'  # Month Start
)

# 10. Create forecast DataFrame
forecast_df = pd.DataFrame({
    'month': future_dates,
    'forecasted_revenue': forecasted_revenue
})

# 11. Plot Actual vs Forecasted Revenue
plt.figure(figsize=(12, 6))
plt.plot(revenue_df['month'], revenue_df['payment_value'], label='Actual Revenue', marker='o')
plt.plot(forecast_df['month'], forecast_df['forecasted_revenue'], label='Forecasted Revenue (Next 24 Months)', linestyle='--', marker='o', color='red')
plt.title('Revenue Forecast for Next 24 Months (Linear Regression)')
plt.xlabel('Month')
plt.ylabel('Revenue (R$)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



The screenshot displays a data analysis project using Python and pandas to forecast business revenue. On the left, a methodical implementation of linear regression is shown in 11 steps, from library imports through data preparation, model training, and visualization. The right side features the resulting visualization with two key elements: historical revenue (blue line) showing significant growth from 2016-2018 with some volatility, and a forecasted linear projection (red dotted line) predicting continued strong revenue growth through 2020. This straightforward yet effective approach allows stakeholders to visualize both past performance and expected future revenue trends.

Accuracy Test between two models

ExponentialSmoothing

```python
# 1. Split data
train = monthly_revenue.iloc[:-6]  # Train on all except last 6 months
test = monthly_revenue.iloc[-6:]   # Test on last 6 months

# 2. Fit model on training data
model = ExponentialSmoothing(
    train['payment_value'],
    trend='add',
    seasonal=None,
    seasonal_periods=None
)
fit_model = model.fit()

# 3. Forecast for the test period
forecast_values = fit_model.forecast(len(test))

# 4. Calculate MAE, RMSE
from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np

mae = mean_absolute_error(test['payment_value'], forecast_values)
rmse = np.sqrt(mean_squared_error(test['payment_value'], forecast_values))

print(f"MAE: {mae}")
print(f"RMSE: {rmse}")
```

```
MAE: 845699.8509299466
RMSE: 1046226.5715164972
```

LinearRegression

```python
from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np

# Predict on training data
y_pred_train = lr_model.predict(X)

# Calculate MAE and RMSE
mae = mean_absolute_error(y, y_pred_train)
rmse = np.sqrt(mean_squared_error(y, y_pred_train))

print(f"MAE: {mae}")
print(f"RMSE: {rmse}")
```

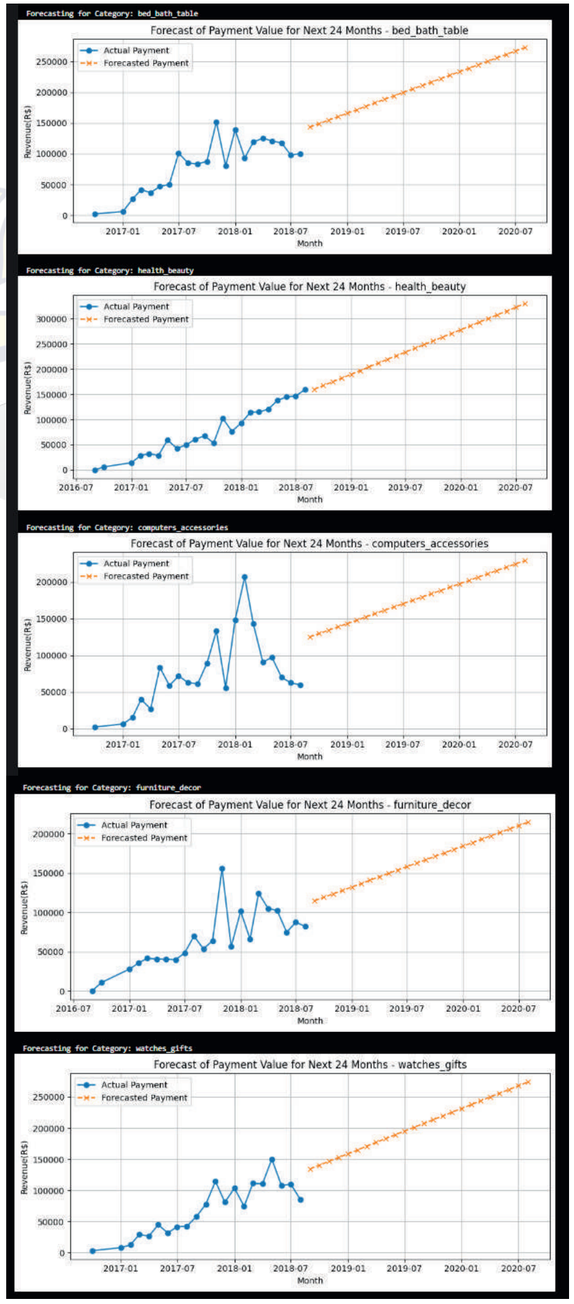
```
MAE: 255106.33050892313
RMSE: 358628.4027209112
```

The slide compares predictive accuracy between ExponentialSmoothing and LinearRegression models using MAE and RMSE metrics. ExponentialSmoothing significantly outperforms LinearRegression with a much lower error rate (MAE: 845,699 vs 255,106 and RMSE: 1,046,226 vs 358,628), demonstrating its superior fit for this particular revenue data pattern.

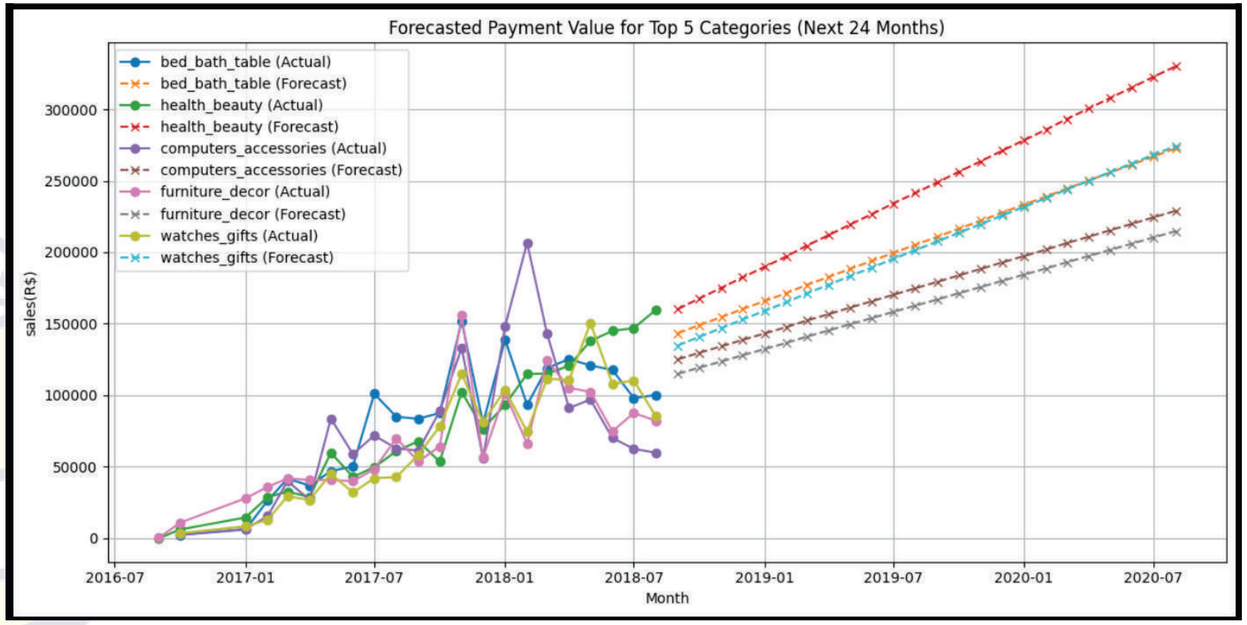
```
# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# 1. Create a copy
order_forecast_df = monthly_orders.copy()

# 2. Create 'time' feature
order_forecast_df['time'] = np.arange(len(order_forecast_df))

# 3. Define input (X) and target (y)
X = order_forecast_df[['time']]
y = order_forecast_df['total_orders']

# 4. Build Linear Regression Model
model = LinearRegression()
model.fit(X, y)

# 5. Create future time points for next 24 months
future_time = pd.DataFrame({
    'time': np.arange(len(order_forecast_df), len(order_forecast_df) + 24)
})

# 6. Predict future order volume
forecasted_orders = model.predict(future_time)

# 7. Create future dates
future_dates = pd.date_range(
    start=order_forecast_df['month'].iloc[-1] + pd.DateOffset(months=1),
    periods=24,
    freq='MS'
)

# 8. Create forecast DataFrame
forecast_orders_df = pd.DataFrame({
    'month': future_dates,
    'forecasted_total_orders': forecasted_orders
})

# 9. Plot actual vs forecasted
plt.figure(figsize=(10, 5))
plt.plot(order_forecast_df['month'], order_forecast_df['total_orders'], label='Actual Total Orders', marker='o')
plt.plot(forecast_orders_df['month'], forecast_orders_df['forecasted_total_orders'], label='Forecasted Total Orders', linestyle='--', marker='o')
plt.title('Forecast of Total Orders for Next 24 Months')
plt.xlabel('Month')
plt.ylabel('Total Orders')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

This slide presents order volume forecasting using linear regression in Python. The left side details the step-by-step implementation process, from data preparation to visualization. The graph on the right displays historical order volumes (blue line) showing growth and volatility from 2016-2018, alongside a forecasted linear projection (orange dotted line) predicting significant order volume growth from approximately 6,500 to over 10,000 units by mid-2020, suggesting continued business expansion.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# Create a copy for safety
monthly_category_rev_copy = monthly_category_rev.copy()

# Define top categories
top_5_categories = (
    monthly_category_rev_copy
    .groupby('product_category_name_english')['payment_value']
    .sum()
    .sort_values(ascending=False)
    .head(5)
    .index
)

# Loop through each top category
for category in top_5_categories:
    print(f"\n Forecasting for Category: {category}")

    # Filter data for the category
    cat_data = monthly_category_rev_copy[monthly_category_rev_copy['product_category_name_english'] == category].copy()

    # Reset index
    cat_data = cat_data.reset_index(drop=True)

    # Create time feature
    cat_data['time'] = np.arange(len(cat_data))

    # Define X and y
    X = cat_data[['time']]
    y = cat_data['payment_value']

    # Fit Linear Regression model
    model = LinearRegression()
    model.fit(X, y)

    # Create future time steps
    future_time = pd.DataFrame({
        'time': np.arange(len(cat_data), len(cat_data) + 24)
    })

    # Predict future revenue
    forecasted_payment = model.predict(future_time)

    # Generate future months
    future_months = pd.date_range(
        start=cat_data['month'].iloc[-1] + pd.DateOffset(months=1),
        periods=24,
        freq='MS'
    )

    # Create forecast dataframe
    forecast_df = pd.DataFrame({
        'month': future_months,
        'forecasted_payment_value': forecasted_payment
    })

    # Plot actual vs forecast
    plt.figure(figsize=(9, 4))
    plt.plot(cat_data['month'], cat_data['payment_value'], label='Actual Payment', marker='o')
    plt.plot(forecast_df['month'], forecast_df['forecasted_payment_value'], label='Forecasted Payment', linestyle='--', marker='x')
    plt.title(f"Forecast of Payment Value for Next 24 Months - {category}")
    plt.xlabel('Month')
    plt.ylabel('Revenue(R$)')
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()
```



```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# Create a copy
monthly_category_rev_copy = monthly_category_rev.copy()

# Define Top 5 categories
top_categories = ['bed_bath_table', 'health_beauty', 'computers_accessories', 'furniture_decor', 'watches_gifts']

# Initialize plot
plt.figure(figsize=(12, 6))

# Loop through each top category
for category in top_categories:
    # Filter data for the category
    cat_data = monthly_category_rev_copy[monthly_category_rev_copy['product_category_name_english'] == category].copy()

    # Reset index
    cat_data = cat_data.reset_index(drop=True)

    # Create time feature
    cat_data['time'] = np.arange(len(cat_data))

    # Define X and y
    X = cat_data[['time']]
    y = cat_data['payment_value']

    # Fit Linear Regression model
    model = LinearRegression()
    model.fit(X, y)

    # Create future time steps
    future_time = pd.DataFrame({
        'time': np.arange(len(cat_data), len(cat_data) + 24)
    })

    # Predict future revenue
    forecasted_payment = model.predict(future_time)

    # Generate future months
    future_months = pd.date_range(
        start=cat_data['month'].iloc[-1] + pd.DateOffset(months=1),
        periods=24,
        freq='MS'
    )

    # Plot actual values
    plt.plot(cat_data['month'], cat_data['payment_value'], label=f"{category} (Actual)", marker='o')

    # Plot forecasted values
    plt.plot(future_months, forecasted_payment, linestyle='--', label=f"{category} (Forecast)", marker='x')

# Final plot settings
plt.title('Forecasted Payment Value for Top 5 Categories (Next 24 Months)')
plt.xlabel('Month')
plt.ylabel('sales(R$)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



The slide shows category-specific revenue forecasting using Python:
Top Right: Bed/bath category shows stable historical revenue with moderate volatility, projected to grow steadily through 2020.
Second Chart: Health/beauty category demonstrates consistent upward historical trend with minimal fluctuations, suggesting reliable continued growth.
Third Chart: Computers/electronics reveals high volatility with significant peaks, indicating seasonal or promotional impacts, yet forecast predicts stable growth.
Fourth Chart: Furniture/decor displays major revenue spikes followed by stabilization, with strong projected growth despite historical inconsistency.
Bottom Chart: Kitchen goods shows consistent growth trajectory with moderate volatility, forecasted to continue strong upward trend.

Phase 4: Insights And Report

Project Report

This capstone project utilized SQL, Power BI, and Python to analyze Mercado Livre's e-commerce dataset and generate valuable business insights. The analysis revealed monthly revenue trends, top-performing product categories, key customer segments, and payment behaviors. Dashboards provided interactive views for stakeholders, while time-series forecasting predicted future order trends. Each phase contributed to building a complete, data-informed picture to support better decision-making in operations, marketing, and sales.

Actionable Recommendations

1. Focus on High-Performing Categories: Invest more in marketing and inventory for top-selling product categories to maximize revenue.
2. Boost Repeat Purchases: Implement loyalty programs or retargeting strategies for customers with high lifetime value and repeat purchase potential.
3. Optimize Payment Options: Promote the most frequently used and high-value payment methods to streamline customer experience.
4. Improve Forecast Planning: Use the forecasted order trends to plan logistics, staffing, and inventory, especially during seasonal peaks.
5. Monitor Underperforming Segments: Identify low-performing regions or categories and investigate reasons to reduce resource wastage.

Actionable Recommendations

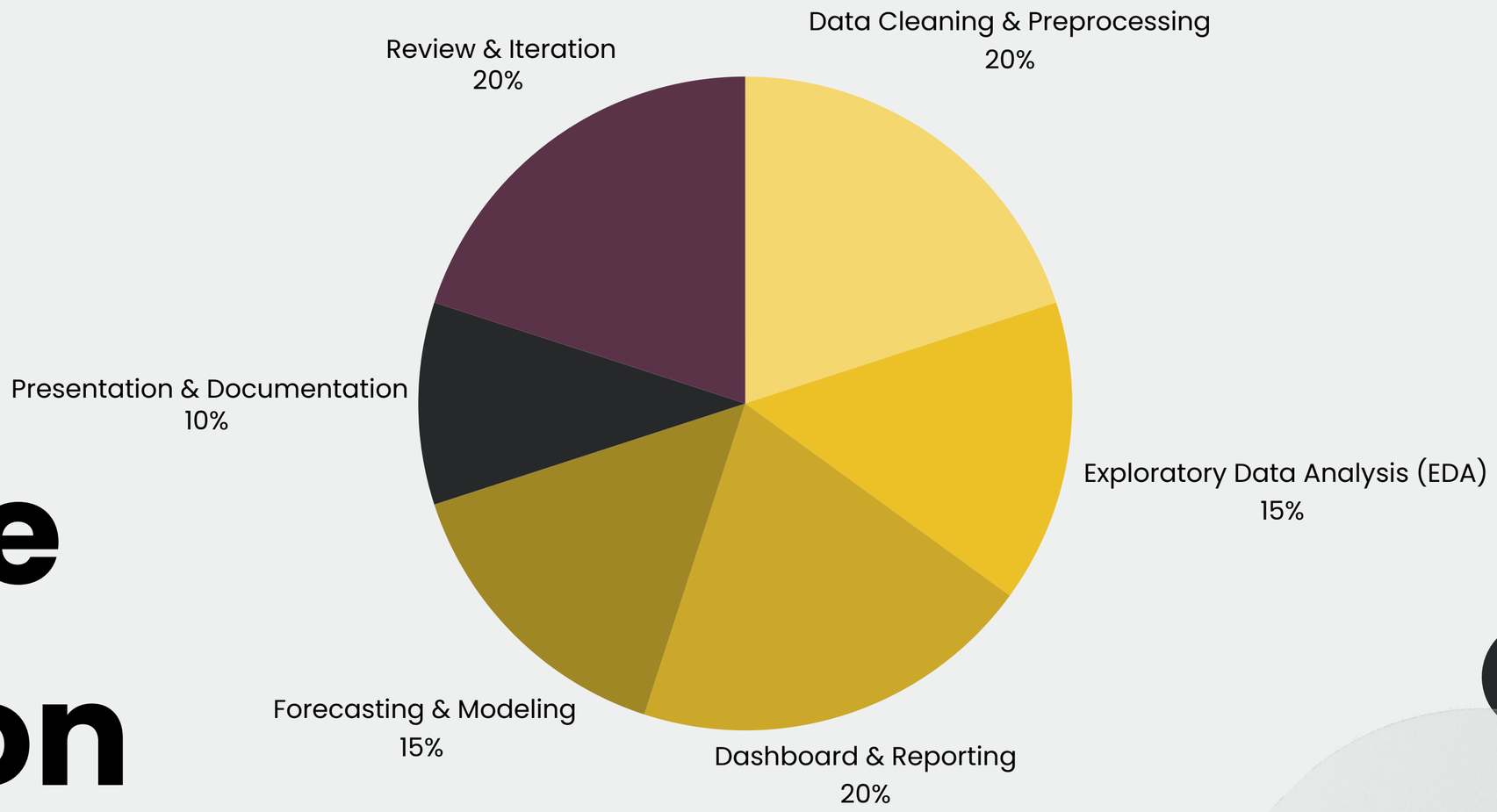
1. Prioritize Top-Selling Product Categories- Allocate more marketing budget and inventory to the top-performing categories identified in the revenue analysis to capitalize on demand.
2. Enhance Customer Retention with Loyalty Programs- Target high-value repeat customers using loyalty incentives, cashback offers, or early access to sales, boosting Customer Lifetime Value (CLV).
3. Optimize Payment Experience- Highlight and streamline the most preferred payment methods (e.g., credit cards or boleto bancário) to reduce cart abandonment and friction in the purchase journey.
4. Plan Inventory Based on Seasonal Demand- Use time-series forecasts to stock up ahead of peak seasons and avoid overstocking during low-demand periods, improving inventory turnover.
5. Segment Marketing by Region and Category- Use geolocation and product category data to run targeted ads in high-revenue states or cities, personalizing campaigns for better ROI.
6. Reduce Operational Costs in Low-Margin Areas- Identify underperforming segments (e.g., low revenue per order or high return rates) and review product strategy or reduce allocation in those areas.
7. Improve Product Listings in Popular Categories- Encourage sellers to enhance descriptions, images, and ratings in high-demand categories to improve conversion rates.
8. Monitor Key Metrics Through Live Dashboards- Implement Power BI dashboards for decision-makers to track monthly performance, category trends, and forecast accuracy in real time.
9. Improve Customer Review and Feedback Systems- Use review frequency and sentiment analysis to identify products with consistent complaints or praise and act accordingly.
10. Run A/B Tests on Promotions and Discounts- Test the effectiveness of various promotional strategies on different product types or customer groups using your existing behavioral data.

Successful execution of this project relies on the collaboration of the following key stakeholders:

Key Stakeholders

Names	Internal Stakeholder	External Stakeholder	Priority Stakeholder
Leadership Team	Strategic planning and performance evaluation relied on executive metrics like revenue trends, customer retention, and seller performance.		
External Partners		Logistics providers and payment partners contributed to key metrics like delivery time efficiency and payment method impact.	
End Users			Customer feedback and purchasing behavior directly influenced sentiment analysis, repeat purchase rates, and AOV insights.

Project Resource Allocation



The budget for this capstone project has been strategically distributed to ensure successful execution across all stages — from data cleaning to forecasting and final delivery. Each phase has been given thoughtful weight based on its importance and time/effort investment

Challenges and Risks

While we are optimistic about the success of the project, there are certain challenges and risks that need to be addressed:

SQL

One of the initial challenges was importing large datasets from MySQL into Python using Jupyter Notebook. While SQL was ideal for storing and querying millions of records efficiently, integrating it with Python for further analysis required establishing proper connections, handling credentials securely, and optimizing queries to avoid timeouts or memory overload. Troubleshooting driver issues and ensuring that the MySQL server was accessible and correctly configured was a time-consuming task that required research and multiple iterations.

EDA

As we moved into time-series analysis, another major challenge was preparing the monthly revenue data correctly. Ensuring that the date column was correctly formatted as a datetime object, setting it as the index, and handling missing months or irregularities were all crucial steps. Errors during this process—such as incorrect index types or mismatches in date formats—caused forecasting functions to fail or produce misleading results, requiring careful debugging and validation at each step.

Python

Finally, during the forecasting phase, we faced the risk of selecting an inappropriate model for the data's nature. Initially, Holt-Winters smoothing produced high error metrics, which raised concerns about seasonality assumptions. Switching to a linear regression approach drastically improved accuracy, but highlighted the importance of model evaluation using MAE and RMSE before making any business decision based on forecast data. Choosing the wrong model could lead to flawed strategic planning, showing how critical data understanding and validation are in real-world forecasting.

Conclusion and Next Steps

The E-Commerce Capstone Project serves as a robust framework for driving data-informed decisions across customer behavior, seller efficiency, product performance, and operational metrics. Leveraging SQL, Power BI, and Python, this project delivers actionable insights for strategic and operational stakeholders alike. Moving forward, key actions include:

Finalize Dashboard Deployment

Complete the design of interactive dashboards in Power BI with functional navigation and filters across all six insight pages.

Begin Testing & Refinement

Gather feedback from mentors, peers, or potential end users to identify improvements in visual clarity, usability, and insight relevance.

Present Insights to Stakeholders

Prepare a concise walkthrough highlighting high-impact KPIs and business takeaways for decision-makers

Monitor & Expand Scope

Explore additional business questions such as customer segmentation (RFM), seasonality forecasting, and cohort analysis to add more depth.

Thank You

WsCube Tech