Zomato Delivery Efficiency Analytics

Project Report

Nishanthini Vasudevan
Nov’25

Table of Contents

Project Report: Zomato Delivery Efficiency Analytics

Executive Summary

This project delivers an end-to-end data analytics solution to evaluate Zomato's delivery efficiency over a period of one quarter along with identifying bottlenecks and areas of improvement. Using a complete ETL pipeline built in Python, SQL and Power BI with a 45000+ dataset, enriched with 15+ engineered features and analyzed through a series of high-impact operational case studies.

The study presents actionable recommendations aimed at improving delivery quality and customer satisfaction. The project provides a comprehensive data-backed framework that supports strategic decision-making and operational process optimization.

Business Problem

The food delivery business needed a structured operations analysis framework to monitor delivery performance across multiple factors—such as city, traffic density, timing, festivals, vehicle type, weather conditions, multi-order behavior, etc.—and identify key bottlenecks and improvement areas that directly impact customer experience, delivery efficiency, and operational scalability.

Tech Stack

Python

- pandas
- numpy
- data cleaning
- custom functions
- datetime handling

SQL (MySQL)

- Window functions
- Aggregations
- Joins
- CTEs
- Performance queries

Power BI

- Interactive dashboard
- DAX for calculated measures
- Slicers, cards, tree maps, line charts, bar charts

Data Understanding

The delivery dataset exported from Kaggle *(find it here)* spans over a time period of three months (Feb – Apr’22). This dataset provides a comprehensive view of delivery operations, including delivery person details, order timestamps, weather conditions, and more.

Shape of Dataset: (45584, 21)

Columns:

ID	Unique identifier for each delivery
Delivery_person_ID	Unique identifier for each delivery person
Delivery_person_Age	Age of the delivery person
Delivery_person_Ratings	Ratings assigned to the delivery person
Restaurant_latitude	Latitude of the restaurant
Restaurant_longitude	Longitude of the restaurant
Delivery_location_latitude	Latitude of the delivery location
Delivery_location_longitude	Longitude of the delivery location
Order_Date	Date of the order
Time_Ordered	Time the order was placed
Time_Order_picked	Time the order was picked up for delivery
Weather_conditions	Weather conditions at the time of delivery
Road_traffic_density	Density of road traffic during delivery
Vehicle_condition	Condition of the delivery vehicle
Type_of_order	Type of order (e.g., dine-in, takeaway, delivery)
Type_of_vehicle	Type of vehicle used for delivery
Multiple_deliveries	Indicator of whether multiple deliveries were made in same trip
Festival	Indicator of whether the delivery coincided with a festival
City	City where the delivery took place
Time_taken (min)	Time taken for delivery in minutes

Data Preparation

Extract:

The raw csv downloaded from Kaggle and added to the project repository. In a new Jupiter notebook, the data uploaded into the dataframe using panda.

Transform:

Data cleaning was performed using pandas. Including:

- Assigning the right data type for datetime columns
- Custom functions to convert time columns into a timestamp datatype
- Renaming column names (Time_taken (min)) to eliminate spaces in name
- Adding an inferred column named Restaurant_distance
- Imputing missing values with mean/mode depending on the column type
- Number of null values per column:

ID	0	
Delivery_person_ID	0	
Delivery_person_Age	1854	Mean imputation
Delivery_person_Ratings	1908	Mean imputation
Restaurant_latitude	0	
Restaurant_longitude	0	

Delivery_location_latitude	0	
Delivery_location_longitude	0	
Order_Date	0	
Time_Orderd	5799	
Time_Order_picked	5007	
Weather_conditions	616	Mode imputation
Road_traffic_density	601	Mode imputation
Vehicle_condition	0	
Type_of_order	0	
Type_of_vehicle	0	
multiple_deliveries	993	Mean imputation
Festival	228	Mode imputation
City	1200	Mode imputation
Time_taken_mins	0	
Restaurant_distance	0	

- Handling outliers (rating more than 5.0 changed to 5.0)

Load:

The cleaned dataframe uploaded to the MySQL server using the sqlalchemy library.

```
%pip install sqlalchemy pymysql

%pip install cryptography

from sqlalchemy import create_engine

engine = create_engine(f"mysql+pymysql://{username}:{password}@{host}/{database}")

df.to_sql('zomato', engine, if_exists='replace', index=False)
```

SQL Analysis

The dataset is analyzed based on specific factors and queries are grouped similarly.

The categories and examples of queries run:

1. Delivery Efficiency

```
-- Q1.5) What is the typical delivery time distribution (e.g., median, p90,
p95)?
WITH per_cte AS (
    SELECT
        Time_taken_mins,
        NTILE(100) OVER (ORDER BY Time_taken_mins) AS percentile
    FROM zomato
)
SELECT
    MAX(CASE WHEN percentile = 50 THEN Time_taken_mins END) AS p50_median,
    MAX(CASE WHEN percentile = 90 THEN Time_taken_mins END) AS p90,
    MAX(CASE WHEN percentile = 95 THEN Time_taken_mins END) AS p95
FROM per_cte;
```

Result:

p50_median	P90	P95
26	40	45

```
-- The median delivery time is around 26 minutes, with p90 and p95 being
around 40 and 44 minutes respectively.
```

2. Delivery Partner Performance

```
-- Q2.2) Do higher-rated riders deliver faster?
SELECT ROUND(AVG(Time_taken_mins),2) AS highrated_timings
FROM(
SELECT Delivery_person_ID, ROUND(AVG(Delivery_person_Ratings),2) AS
avg_ratings, Time_taken_mins
FROM zomato
GROUP BY Delivery_person_ID, Time_taken_mins
ORDER BY avg_ratings DESC
LIMIT 20 ) a;
```

Result:

highrated_timings
29.20

-- Yes, higher-rated riders tend to have faster delivery times averaging around 25.45 minutes while average delivery time across all riders is around 26.29 minutes.

3. Geographical and Distance Analysis

```
-- Q3.1) How does the distance between restaurant and customer affect
delivery time?
SELECT CASE
WHEN Restaurant_distance < 50 THEN 'Very Near'
WHEN Restaurant_distance >= 50 AND Restaurant_distance < 100 THEN 'Near'
WHEN Restaurant_distance >= 100 AND Restaurant_distance < 150 THEN 'Far'
ELSE 'Very Far' END AS distance_description,
ROUND(AVG(Time_taken_mins),2) AS avg_time
FROM zomato
GROUP BY distance_description
ORDER BY avg_time DESC;
```

Result:

distance_description	avg_time
Very Near	26.30
Very Far	25.81
Near	24.64

-- Contrary to expectations, 'Very Near' deliveries take the longest time on average, 'Very Far' taking second longest and 'Near' taking the least time. This may indicate other factors influencing delivery time beyond just distance.

4. Weather Impact Analysis

```
-- Q4.3) Is rain traffic worse than peak-hour traffic?
SELECT CASE
    WHEN Road_traffic_density = 'High' THEN 'Peak Hour Traffic'
    WHEN Weather_conditions = 'Stormy' THEN 'Rain Traffic'
    ELSE 'Other'
END AS traffic_type,
ROUND(AVG(Time_taken_mins),2) AS avg_time
FROM zomato
GROUP BY traffic_type
ORDER BY avg_time DESC;
```

Result:

traffic_type	avg_time
Peak Hour Traffic	27.24
Other	26.30
Rain Traffic	25.66

-- No peak hour traffic has higher average delivery time around 27 minutes compared to rain traffic around 25 minutes.

5. Time of day and Day of week patterns

```
-- Q5.3) Are weekends slower than weekdays?
SELECT CASE
WHEN DATE_FORMAT(Order_Date, '%W') IN ('Saturday','Sunday') THEN 'Weekend'
ELSE 'Weekday'
END AS day_of_week,
ROUND(AVG(Time_taken_mins),2) AS avg_time,
COUNT(*) AS no_of_orders
FROM zomato
GROUP BY day_of_week
ORDER BY avg_time DESC;
```

Result:

day_of_week	avg_time	no_of_orders
Weekday	26.31	33049
Weekend	26.25	12535

-- The average delivery time doesn't vary significantly between weekends and weekdays. However, weekdays see a higher number of orders around 33000 compared to weekends around 12000. Possible reasons: 1. More days in weekdays 2. Weekends have more people dining out reducing delivery orders.

6. Demand and Customer behavior

```
-- Q6.1) Which order types are most popular in each city?
SELECT City, Type_of_order, COUNT(*) AS order_count
FROM zomato
GROUP BY City, Type_of_order
ORDER BY City, order_count DESC;
```

Result:

City	Type_of_order	order_count
Metropolitan	Meal	8937
Metropolitan	Snack	8901
Metropolitan	Buffet	8728
Metropolitan	Drinks	8721
Semi-Urban	Snack	54
Semi-Urban	Buffet	42
Semi-Urban	Meal	34
Semi-Urban	Drinks	34
Urban	Snack	2575
Urban	Drinks	2566
Urban	Buffet	2507
Urban	Meal	2485

-- Metropolitan: Meal, Semi-Urban: Snack, Urban: Snack are the most popular order types in respective cities.

7. Operational bottleneck identification

```
-- Q7.2) Are delays clustered around specific delivery partners?
SELECT Delivery_person_ID, ROUND(AVG(Time_taken_mins),2) AS avg_time,
COUNT(*) AS no_of_orders,
ROUND(AVG(Delivery_person_Ratings),1) AS avg_rating
FROM zomato
GROUP BY Delivery_person_ID
HAVING avg_time>30
ORDER BY avg_time DESC;
```

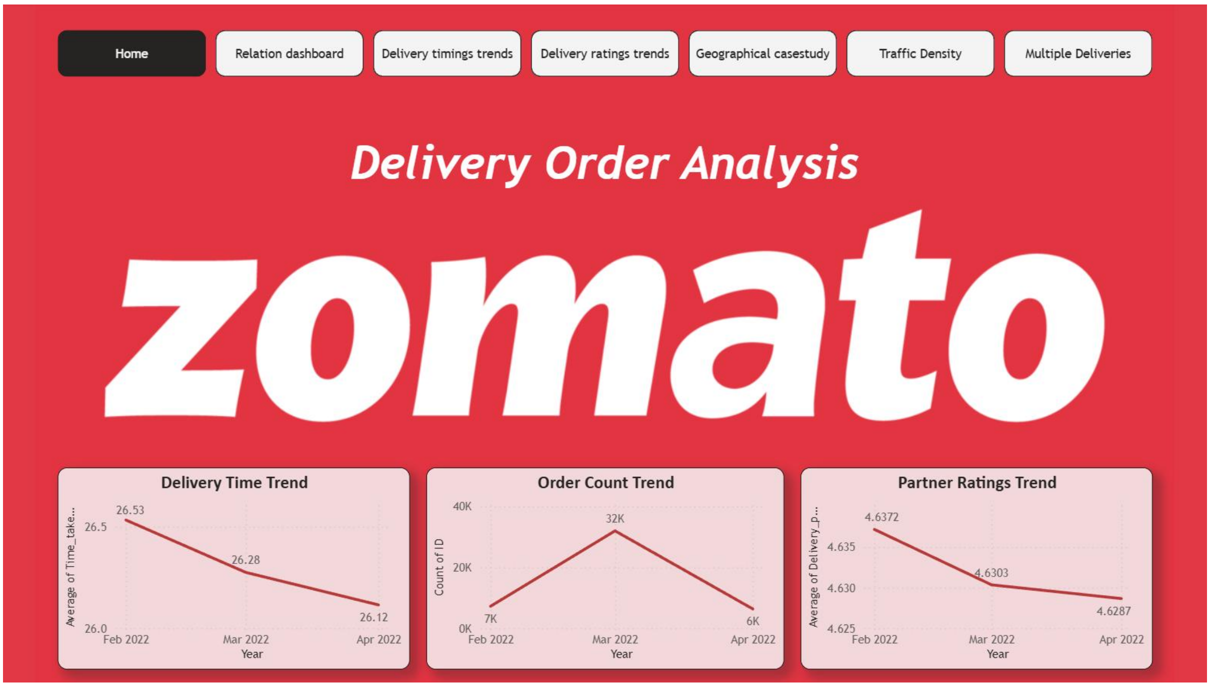
Result:

-- 46 riders have average delivery times above 30 minutes. Interestingly, many of these riders have average ratings above 4.5, Which clashes with earlier observation of low rated riders causing delays. Indicates other factors at play.

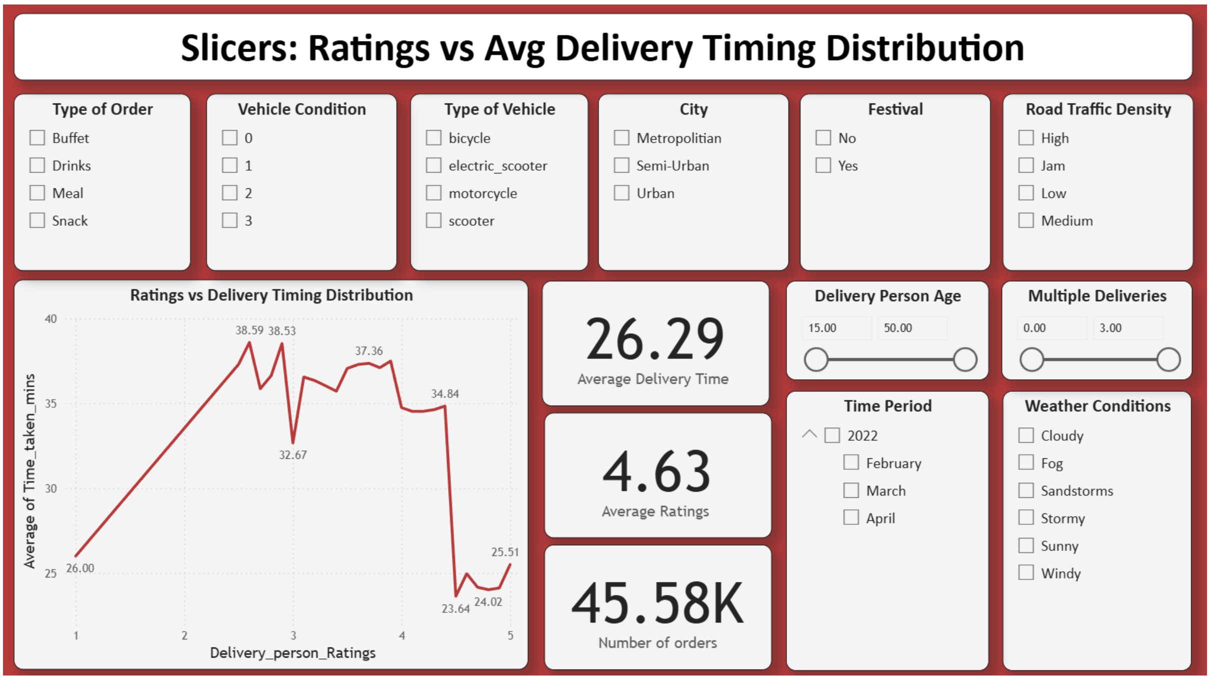
Dashboard & Visualization

Power BI dashboard prepared using the table loaded to the SQL server.

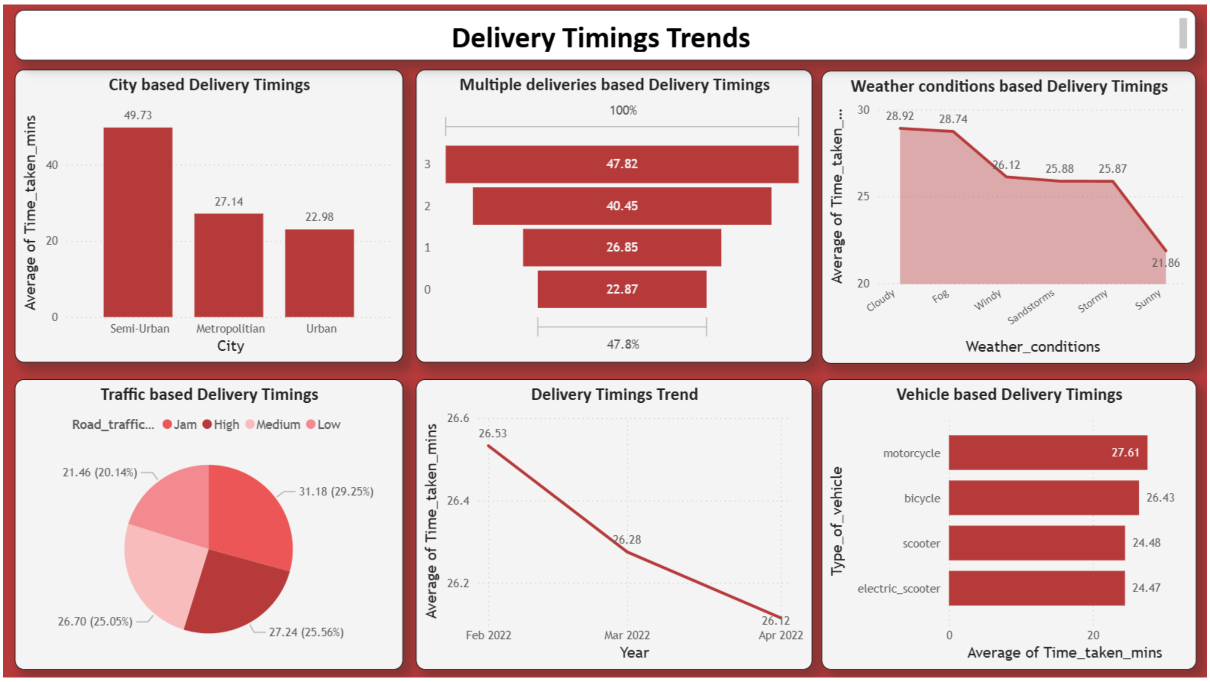
Screenshots:



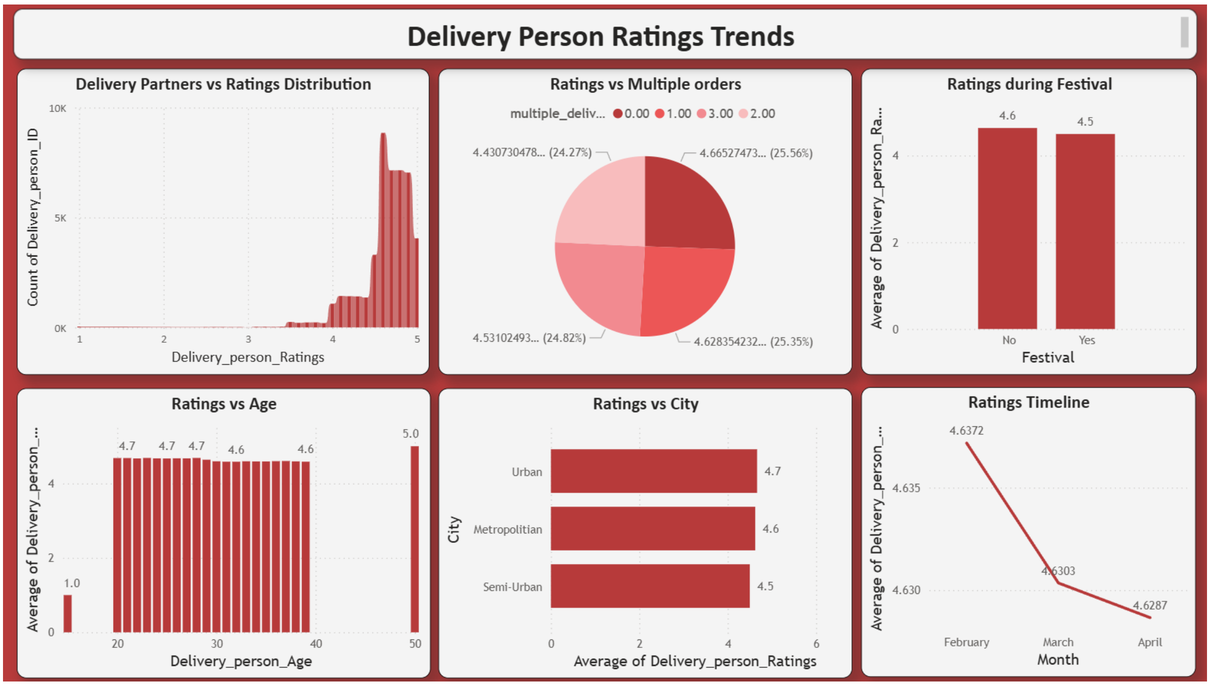
Homepage



Delivery Timing vs Ratings Relationship based on multiple factors filtering



Delivery Timings Trends



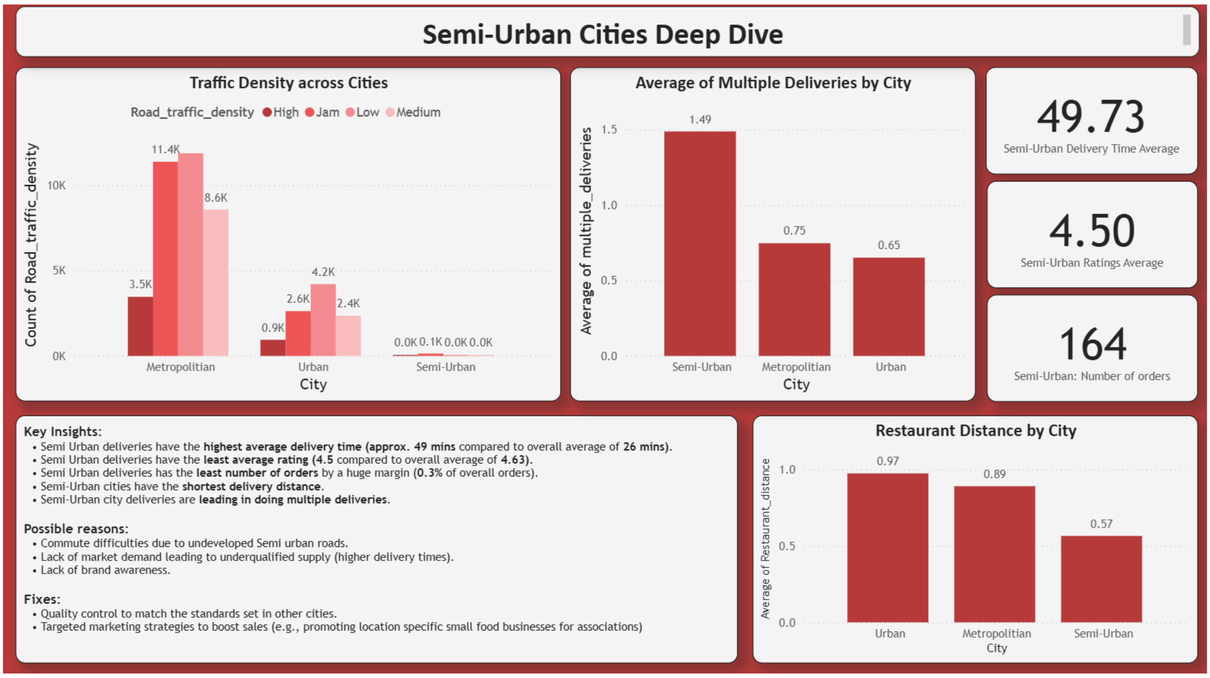
Delivery Partner Ratings Trends

Case Studies

This project investigates three major drivers: Geography, Traffic Density, and Multiple Deliveries.

1. Geographical Constraints: Semi-Urban Performance Gap

Semi-Urban regions consistently underperform across all delivery metrics.



Key Findings:

- Slowest delivery times: ~49 mins vs 26-min overall average
- Lowest customer ratings: 4.50 vs 4.63 overall
- Extremely low order volume: 0.3% of total orders
- Highest tendency for multiple deliveries, increasing delays
- Shortest distance (0.57 km), proving distance is not the cause of slow performance

Root Causes:

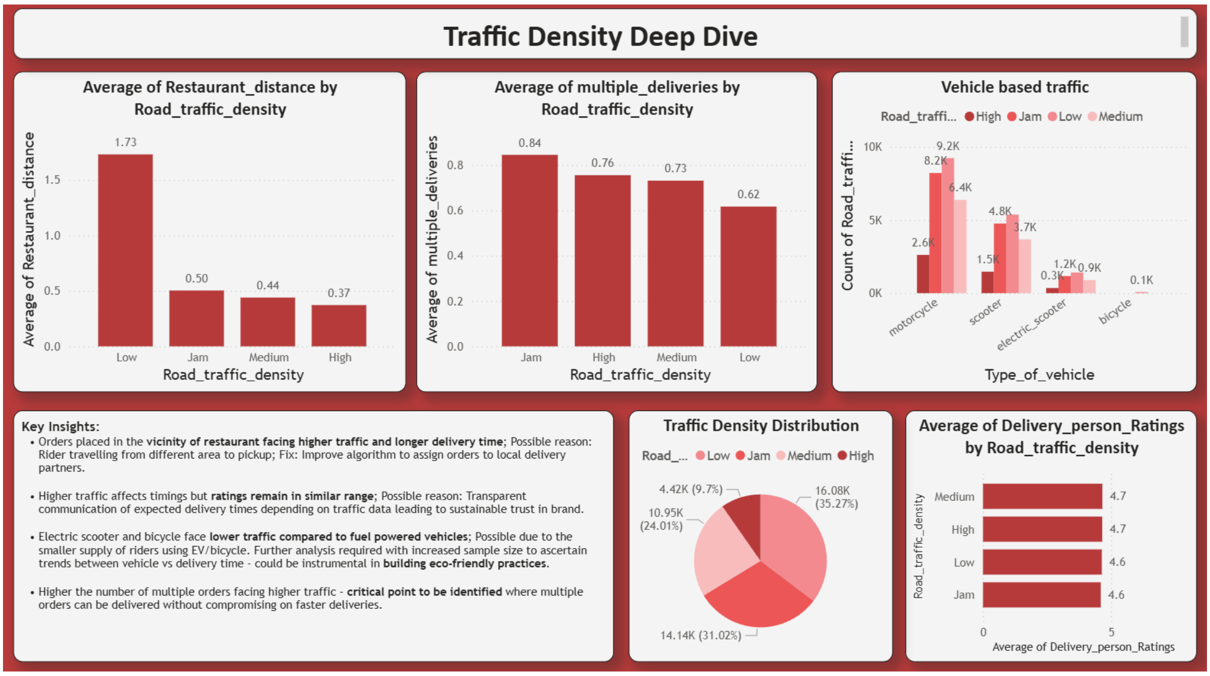
- Poor infrastructure slowing rider movement
- Low demand causing inconsistent or undertrained rider supply
- Limited brand presence and customer adoption

Recommendations:

- Strengthen quality control and rider training
- Increase brand awareness via local partnerships
- Limit multi-order assignments in Semi-Urban areas

2. Traffic Density: Rider Assignment Inefficiency

Orders placed near restaurants still face high delivery times in traffic-heavy zones.



Insights:

- Higher traffic leads to higher delivery timings but the ratings remain in same range
- Riders are often dispatched from distant areas, delaying pickup
- High traffic amplifies these delays, even for short-distance orders
- Multiple orders delivery facing higher traffic and subsequently higher delivery timings

Root Causes:

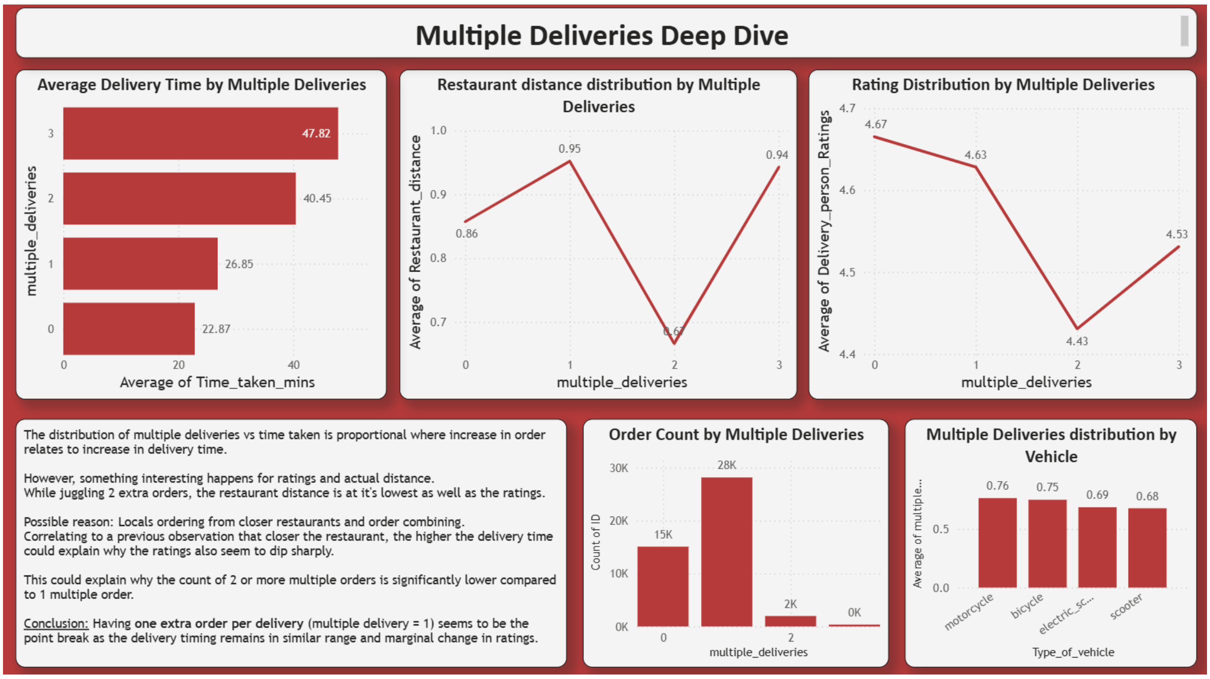
- Delivery partner assigned from farther distances increasing pickup time

Recommendation:

- Improve the rider–order matching algorithm to prioritize nearest available delivery partners
- Finding the threshold point for multiple deliveries ensuring shorter delivery timings and shortest distance travelled

3. Multiple Orders: The Multi-Delivery Threshold

Handling more orders increases delivery time as expected, but deeper patterns emerge.



Key Findings:

- Direct increase in delivery time with each additional order
- Surprisingly, restaurant distance decreases when handling 2+ orders
- Ratings drop sharply despite shorter distances
- Volume of 2+ multi-deliveries is significantly low, indicating natural or system-avoided overload

Interpretation:

- Customers ordering locally for combined deliveries create high workload with low travel distance
- Even short distances can see delays when rider allocation is inefficient

Conclusion:

The optimal operational threshold is 1 extra order:

- Delivery time remains stable
- Ratings show minimal decline
- Rider workload stays manageable

Beyond this point, performance and customer satisfaction drop significantly.

Insights & Recommendations

Overall delivery time is improving, but customer ratings are declining, indicating deeper operational factors at play.

Three separate bottlenecks are identified in the case studies with actionable plans to improve efficiency:

- Improving quality of services in Semi-Urban cities and marketing strategies for brand recognition
- Improving the rider–order matching algorithm to prioritize nearest available delivery partners
- Testing with the threshold point of clubbing 1 extra order to verify performance improvement

Conclusion

This project provides a comprehensive, end-to-end analysis of Zomato's delivery operations, uncovering key performance gaps across geography, traffic density, and multi-delivery behavior.

The insights generated enable data-backed decision-making aimed at improving delivery efficiency, customer satisfaction, and operational scalability. Implementing the recommended actions can significantly enhance service quality while reducing delivery variability.

Overall, this project demonstrates how structured data processing, analytical modeling, and visualization can transform raw operational data into actionable intelligence, providing a scalable framework for ongoing performance monitoring and future optimization initiatives.

Future Work

- Implementation of recommended plans of action and monitoring data to verify effectiveness.
- Looking into eco-friendly initiatives:
    - Electric scooter and bicycles shown to face lower traffic compared to other vehicles.
    - Can be studied further to adopt sustainable environmental measures
    - Offering incentives to partners opting for environmentally friendly modes of transport.

Appendix

Repository:

```
data-analytics-projects/
└── zomato_data/
    ├── data/
    │   ├── Zomato_Dataset.csv
    │   └── zomato_data.db
    ├── powerbi/
    │   ├── Zomato_delivery_dashboard.pbix
    │   ├── dashboard_homepage.png
    │   ├── delivery_timing_trends.png
    │   ├── geographical_casestudy.png
    │   ├── multiple_orders_casestudy.png
    │   ├── ratings_trends.png
    │   ├── slicers.png
    │   └── traffic_density_casestudy.png
    ├── python/
    │   ├── zomato_cleaning.ipynb
    │   └── zomato_data.db
    ├── sql/
    │   ├── 01_delivery_efficiency.sql
    │   ├── 02_delivery_partner_performance.sql
    │   ├── 03_geographic&distance_analysis.sql
    │   ├── 04_weather_impact_analysis.sql
    │   ├── 05_time-of-day&day-of-week_patterns.sql
    │   ├── 06_demand&customer_behavior.sql
    │   ├── 07_operational_bottleneck_identification.sql
    │   └── zomato_queries.sql
    └── README.md
```