Predicting Delivery Delays and Customer Satisfaction in Food Delivery Services

Group Details:
Member 1: Oscar J. Fernandez Urias – ojf12
Member 2: Amrit Chandrasekaran – ac2532
Member 3: Tanav Bollam – tb811

1. Project Definition:

Food delivery platforms have become a key part of many people's lives, especially those living in urban cities, where the speed and convenience of food deliveries increase customer satisfaction the most. One major factor that shapes a user's experience is whether the order arrives on time or not. Late deliveries very often lead to cold food, customer dissatisfaction, and customer distrust in deliveries. On the other hand, consistent and on-time deliveries help build trust and long-term customer loyalty.

The goal of our project was to build an end-to-end data pipeline that predicts whether delivery delays were late or not and how this aspect leads to customer satisfaction or dissatisfaction. Using a structured dataset of complete food orders, we analyzed the operational features, including delivery distance, weather, traffic level, food preparation time, experience of drivers, etc. From these features, we engineer new variables that allow us to answer 2 key questions:

1) Can we predict whether a delivery will be late based on operational conditions?
2) When there is a delay, what can we predict about the customer's satisfaction outcome?

Our approach follows the principles that were taught in CS210: data cleaning, preprocessing, SQL storage, querying, exploratory data analysis, and machine learning model development. We cleaned the dataset and engineered some additional features. After that, we stored the processed data in SQLite and performed some SQL queries to analyze the data. To further understand the data, we visualized it using histograms, countplots, and heatmaps, which were a part of our exploratory data analysis step. After that, we trained classification models, including Logistic Regression and Random Forest, to predict late deliveries. Finally, we evaluated model performance and connected delivery delays to a satisfaction proxy to capture real-world customer experience.

2. Introduction:

The food delivery market depends heavily on speed, reliability, and quality of food. Mobile apps make ordering food easier than ever, but they also increase customer expectations and reduce switching costs. One unsatisfactory experience can result in a major loss and cause a user to switch to another competing platform. Among the several challenges faced by delivery systems, late deliveries remain one of the most significant predictors of negative customer experience. Even when the food quality is high, longer delays may have a large negative impact on the value of the order.

Delivery time is influenced by several factors - some are external, such as weather and traffic, and others are internal, such as preparation time or vehicle type. Understanding these influences requires a data-driven approach and cannot rely on intuition alone. Machine learning and statistical analysis allow us to uncover these relationships and make accurate predictions by providing powerful tools.

In this project, we analyze a real-world Kaggle dataset titled Food_Delivery_Times.csv, which includes key operational features for completed deliveries. Although the dataset does not include specific customer feedback, we can engineer a satisfaction proxy based on delivery time. This allows us to not only predict whether a delivery will be late but also find out how lateness correlates with customer satisfaction.

The novelty of our project lies in combining lateness prediction with a simple but realistic satisfaction model. Rather than stopping at binary classification (late vs. on-time), we categorize satisfaction into three levels—*Likely Satisfied*, *Neutral at Risk*, and *Likely Dissatisfied*. This additional layer provides more actionable insights for delivery platforms, helping identify where operational improvements would have the greatest impact on customer experience.

3. Methodology:

Our methodology follows the full lifecycle of a data science project and intentionally reflects the structure provided in the CS210 final project instructions. We divide our work into three main components: the Data Science component, the Database / SQL component, and the Machine Learning component.

- Data Science Component:

We started by importing a raw CSV dataset, inspecting its structure, and computing descriptive statistics. This first step allowed us to find out that the data types were correct and no corrupt values were present. We then addressed missing data. We saw that many of the rows that lacked missing values had missing values for multiple columns. Therefore, we decided that instead of filling null values, we could just remove the null values entirely, since removing the null values didn't remove that much from our dataset. The number of entries went from 1000 to 883, which was completely fine.

Next, we performed feature engineering. The most important engineered feature is a binary column called 'Late', which was our target variable. If the delivery time exceeded 30 mins, we set it to 1; otherwise, we kept it at 0. While the 30-minute cutoff may seem odd, it does align with many delivery app expectations and provides a target variable for binary classification modelling.

After that, we decided to feature engineer a satisfaction bucket named 'Satisfaction_Bucket'. This variable had 3 levels. Deliveries in the 31-45 mins range were labeled 'Neutral / At Risk', deliveries over 45 mins were labeled 'Likely Dissatisfied', and deliveries completed in 30 or under minutes were labeled 'Likely Satisfied'. Although this is a proxy as opposed to true customer ratings, it provides a realistic approximation of customer perception and helps us analyze satisfaction trends.

After generating these new columns, we conducted exploratory data analysis to better understand the dataset. Histograms of delivery time showed that most orders are not completed within 30 minutes. And a lot of them took longer than 45 minutes. Distance distributions did not indicate much. Later, we produced bar charts and countplots to analyze categorical variables, and this gave us a great understanding of the data that we would later model.

- <u>Database / SQL Component:</u>

After cleaning and exploring the dataset, we focused on the database component. We used the .to_sql() function in Pandas to store a cleaned database in SQLite called 'food_delivery.db'. The main table, deliveries, contains one row per order with columns representing distance, weather, traffic level, time of day, vehicle type, prep time, courier experience, delivery time, the Late label, and the Satisfaction_Bucket.

With the dataset stored in SQLite, we executed a series of SQL queries. Our first query computed the average delivery time for each weather condition, allowing us to evaluate whether rain, snow, or fog had an impact on delivery delays compared to clear weather. A second query showed the relationship between traffic level and lateness by calculating both the number of orders and the average Late rate for each traffic category. This analysis showed a strong pattern: higher traffic means later deliveries. Additional queries analyzed satisfaction buckets and vehicle types. A query grouped by Satisfaction_Bucket revealed how many orders fell into each satisfaction tier and the average delivery time within those tiers. Another query we wrote grouped by Vehicle_Type compared delivery performance for bikes, scooters, and cars. We also queried late rates by time of day and found that nighttime deliveries were less likely to be late than morning or afternoon deliveries, with not much of a difference. These SQL analyses both satisfied the project's database requirements and helped validate patterns explored in our visualizations and modelling.

- Exploratory Data Analysis Component

After cleaning and engineering new features, we did exploratory analysis to understand the structure and behavior of the dataset before building the machine learning models. Our EDA focused on visualizing distributions, finding trends, and examining relationships between key variables.

We first generated histograms for continuous variables such as Delivery_Time_min and Distance_km. These visualizations revealed that most deliveries were completed between 30 minutes and 70 minutes. Next, we created bar charts and countplots for categorical variables like Weather, Traffic_Level, and Time_of_Day to observe how frequently each category occurred. These plots helped us understand operational patterns, such as peak delivery times and the dominance of clear weather conditions.

We then used a correlation heatmap to examine how numerical features and one-hot-encoded categorical features interacted with each other. While no single correlation was extremely high, the heatmap helped confirm expected relationships, such as a moderate correlation between Distance_km and Delivery_Time_min. This step was especially important to check for multicollinearity before training machine learning models.

Together, these EDA techniques provided a thorough understanding of the dataset's distributional shape, category frequencies, and feature relationships, which informed both our SQL analysis and our modeling decisions.

- Machine Learning Component:

The final major component of our methodology involved training machine learning models to predict whether a delivery would be late. Before modeling, we converted categorical variables—including Weather, Traffic_Level, Time_of_Day, and Vehicle_Type—into numerical format using one-hot encoding. This produced a complete feature matrix in which each row represented a delivery and each column represented either a numerical attribute or an encoded category. We then split the dataset into an 80% training set and a 20% test set, stratified by the Late label to maintain balanced representation.

We trained two classification models: Logistic Regression and Random Forest. Logistic Regression served as a simple, interpretable baseline model that assumes a linear relationship between features and the probability of being late. Random Forest, in contrast, builds many decision trees from random subsets of the data, averages their predictions, and captures more complex nonlinear interactions. It also provides intuitive feature importance scores.

For evaluation, we computed accuracy, precision, recall, F1-score, and confusion matrices for each model. Accuracy reflects the overall correctness of predictions, while precision and recall specifically measure how well the models identify late deliveries. The F1-score combines precision and recall into a single balanced metric. Finally, we examined Random Forest's feature importance rankings to understand which variables most strongly influenced lateness predictions. These evaluation steps allowed us to compare both models and determine which would be more successful in a real-world delivery prediction system.

4. Results:

Our project's results can be grouped into three categories: insights from SQL queries, patterns found through exploratory data analysis, and the performance and interpretation of our machine learning models. Together, these results provide a coherent story about what drives delivery delays and how those delays affect likely customer satisfaction.

From the SQL analysis, we confirmed that environmental and situational factors have noticeable effects on delivery time. Average delivery time was higher in rainy or snowy weather compared to clear conditions, although the difference was not as dramatic as we initially expected. Traffic level, however, showed somewhat of a relationship for lateness. Our query for the average Late rate by Traffic_Level demonstrated that low-traffic deliveries had the lower proportion of late orders, while high-traffic deliveries had higher proportions.

Vehicle type and time of day were other crucial factors. Those made by bike or scooter tended to arrive faster than those made by cars, most likely because cars find it difficult to navigate through traffic. When we grouped by time of day, evening orders had greater late rates than morning or afternoon orders, which supports the idea that dinner is a peak demand period with significant traffic and restaurant congestion.

Our exploratory data analysis produced a more graphic representation of these trends. Histograms indicate that most orders were delivered between 30 minutes and 70 minutes. Plots of distance and their respective counts show that no particular distance was greater than other distances. Furthermore, bar plots comparing traffic levels and satisfaction buckets demonstrated that high traffic not only increases overall delivery time but also shifts a larger share of orders into the neutral or dissatisfied categories. Alas, the heatmap gave us quite a few insights. It showed that distance_km and Delivery_Time_min had the most impact on whether deliveries were late. Preparation_Time_min had a smaller impact, and Order_Id and Courier_Experience_yrs had basically no impact.

The machine learning results helped us to go from description to prediction. Logistic Regression created a precision of 0.98 and a recall of 0.962. This means it created an F1-score of 0.971. Random Forest created a precision of 0.944 and a recall of 0.968. This means it created an F1-score of 0.956. By these results, we can conclude that the Logistic Regression model is more accurate than the Random Forest model we created.

We then analyzed the feature coefficients for Logistic Regression and the feature importance for Random Forest to understand what features had the most impact in both models. We saw that in our Logistic Regression model, Lower Traffic played the largest role in decreasing the chance of it predicting a late delivery, and Snowy Weather played the largest role in increasing the chance of the model predicting a late delivery. We saw in our Random Forest model that the Distance_km and Preparation_Time_min played the biggest roles in predicting lateness.

These results reflect a strong operational message. To minimize late deliveries and customer dissatisfaction, delivery platforms should prioritize managing long-distance orders, improving restaurant preparation efficiency, and supporting courier training and routing. For example, the platform may change delivery timings for long-distance orders during peak traffic hours or encourage businesses with long preparation times to improve certain procedures.

Although our project uses a relatively simple dataset and basic models, the findings demonstrate how a CS210‑level data pipeline can produce practical insights. Using SQL, EDA, and machine learning, we were able to cross-check patterns and provide a consistent view of delivery performance. In a real‑world setting, the same approach could be extended with additional features such as restaurant category, order size, real customer ratings, or dynamic traffic information to build even more powerful predictive systems.

5. Contributions:

- Oscar J. Fernandez Urias (ojf12): Led the overall structure of the project. He was primarily responsible for data cleaning, feature engineering, and the construction of the SQL database. Additionally, he wrote many of the exploratory data analysis plots, implemented the Logistic Regression model, and helped interpret the model outputs in the context of customer satisfaction

- Amrit Chandrasekaran (ac2532): Focused on the machine learning depth of the project. He implemented and turned the Random Forest classifier, analyzed feature importance, and compared model performance metrics. Additionally, he contributed to the discussion of which features were most significant operationally and helped make sure that the modeling choices stayed aligned with the goals of the project.

- Tanav Bollam (tb811) - Contributed to dataset cleaning, supported satisfaction analysis, and data preparation.

6. References:

Food Delivery Time Prediction Dataset. Kaggle. https://www.kaggle.com/datasets/denkuznetz/food-delivery-time-prediction?resource=download