ST126199 - Biki Nath Newa
ST126459 - Hein Min Htet

Progress Report - 1

1. Baseline to Be Implemented This Week

Baseline Model: Random Forest
We will implement a random forest model because according to our correlation matrix, although we have some level of correlation between our features and our target, it is not instantly obvious. We believe that this is due to the use of pearson correlation being used which is mostly limited to linear relationships.

Thus, taking this into account, we have decided to instead settle for a tree-based model such as a Random Forest which isn't limited to a linear scope. Furthermore, in its final form, our model also has 112 columns after One Hot Encoding which causes the curse of dimensionality forcing many distance and neighbour based models to suffer in terms of their performance.

Rationale:
Aside from being transparent, quick to train and providing a reliable and a stable reference for other adversarial methods such as XGBoost, tree-based models also happen to be the most commonly used methods by other researchers and thus, mirrors common practice in challenging baselines before introducing more complex models.

2. Preprocessing and Metrics Policy (Aligned with the Challenge)

1. Data Preparation and Cleaning

The dataset initially had rows containing 150,000 samples and was filtered heavily to create a cleaner and more relevant training-dataset specific to our problem.

1.1 Defining the Target Variable

The target variable, **“is_cancelled”** which was created by isolating only the cancellations which were due to the specific cancellation reason: “**Driver is not moving towards pickup location”**.

- **Filtering:** All other status types (Incomplete, No Driver Found, Cancelled by Driver) and non-target cancellation reasons (Wrong Address, Change of plans, Driver asked to cancel, AC is not working) were removed as they are too random or simply not possible to predict with the currently available features.
- **Imputation:** Aside from dropping the cases where the cancellation sample cases where the features available were irrelevant for explaining them. Several attempts of imputation were also made to salvage as many features as possible.
  Inherently, the core problem that we faced with our dataset was when a ride is cancelled, the only numeric feature that it retained was avg_VTAT which is the average time it takes for the nearest Uber vehicle to arrive towards the customer with everything else being NaN.
- **Final Sample Size:** The cleaned dataset size is 95,235 rows.

1.2 Addressing Class Imbalance

The cleaned data exhibits a severe class imbalance, which is critical for model training:

- Non-Cancelled Rides (0) - 97.65%
- Cancelled Rides (1) - 2.35%

This imbalance necessitates the use of metrics beyond accuracy (like Precision and Recall) and the training strategy of class_weight='balanced' to force the model to learn from the rare positive examples.

1.3 Missing Data Handling

- **Ride Distance / Booking Value:** We imputed **Ride Distance** and **Booking Value** by getting the average value of the same features given that **Pickup Location** and **Dropoff Location** match. Unfortunately, there were still around 100 samples which could not be imputed via this method and had to be dropped..
- **Customer Ratings**: Imputing Customer Rating by using historic data via grouping by **Timestamp** and **Customer ID** and then doing forward fill to add in the “**Customer Rating”**. This way the missing ratings were imputed using a time-based forward-fill

approach and using a customer's last known rating to avoid data leakage. Ratings were then categorized into Good, Average, Poor, and Not Rated.

Unfortunately, since **Driver ID** was not provided, it was not possible to use this technique for salvaging **Driver Ratings** and the feature had to be dropped entirely.

2. Feature Engineering

Domain knowledge was used to create several predictive features from raw data, notably enriching the feature set with behavioral and external data.

Feature Category	**New Features Created**	**Methodology and Rationale**
Location Features	pickup_zone, drop_zone	Raw street addresses were generalized into major metropolitan zones (e.g., 'Gurgaon', 'South Delhi', 'Noida'). This allows the model to capture generalized risk associated with high-demand or high-traffic areas, rather than over-fitting to specific street names. Furthermore, with the original 172 unique values for the original location features, it was difficult to justify carrying out One Hot Encoding for the model, however by doing this, we were able to compress them to 11 categories: South Delhi, Gurgaon, West Delhi, North Delhi, Central Delhi, East Delhi, Noida, Ghaziabad, Outer NCR, Airport Area and Other (Delhi).

Customer Behavior	customer_patience	This is a time-series aggregation: the historical average time a customer waited before cancelling their *prior* ride. This feature provides a powerful behavioral score, estimating how quickly a specific customer is likely to abandon the current ride.
Temporal & Meteorological	hour, day, month, day_of_the_month, weather_condition	Standard temporal features were extracted in an attempt to capture cyclical hourly and monthly trends. Additionally, hourly historical weather data (temperature, humidity, precipitation_mm) for the region was merged into the dataset using the booking_timestamp. This was done in hopes that it may allow the model to account for some external factors, such as customers unwilling to be patient in poor weather (e.g., heavy rain).

In the end, we ended up with these following features for our X:

- **Numerical Features:** 'vehicle_arrival_time', 'distance', 'ride_cost','temperature', 'humidity' and 'precipitation_mm'.
- **Categorical Features:** 'vehicle_type', 'customer_patience', 'pickup_zone', 'drop_zone', 'historical_customer_rating_binned', 'hour', 'day', 'month', 'day_of_month' and 'weather_condition'.

3. Modeling and Evaluation

A Random Forest Classifier was selected as a robust baseline, trained on the engineered features and evaluated using relevant metrics.

3.1 Key Model Performance Metrics

Metric	Result	Interpretation
Accuracy	(98%)	Misleading: High due to the massive number of correctly predicted non-cancellations (the majority class).
Precision	(100%)	Excellent: Every time the model predicted a cancellation, it was correct. The model is highly certain and conservative in its positive predictions.
Recall	(32%)	Weak Point: The model only identified 32% of all actual cancellations, missing the other 68%.

Model 1 & 2: Random Forest Without SMOTE & Stratified Random Forest Without SMOTE

	precision	recall	f1-score	support
0	0.98	1.00	0.99	18600
1	1.00	0.32	0.48	447
accuracy			0.98	19047
macro avg	0.99	0.66	0.74	19047
weighted avg	0.98	0.98	0.98	19047

The model without SMOTE is 0.98 accurate but fails the minority class (support 447), achieving only 0.32 recall despite 1.00 precision. It misses two-thirds of the positive cases.

Model 3 & 4: Random Forest With SMOTE &
Stratified KFold Random Forest With SMOTE

	precision	recall	f1-score	support
0	0.98	1.00	0.99	18600
1	1.00	0.32	0.48	447
accuracy			0.98	19047
macro avg	0.99	0.66	0.74	19047
weighted avg	0.98	0.98	0.98	19047

The model is highly accurate (0.98) by dominating Class 0 (0.99 f1-score). However, it critically fails Class 1, achieving only 0.32 recall, showing it misses two-thirds of the positive cases despite SMOTE.

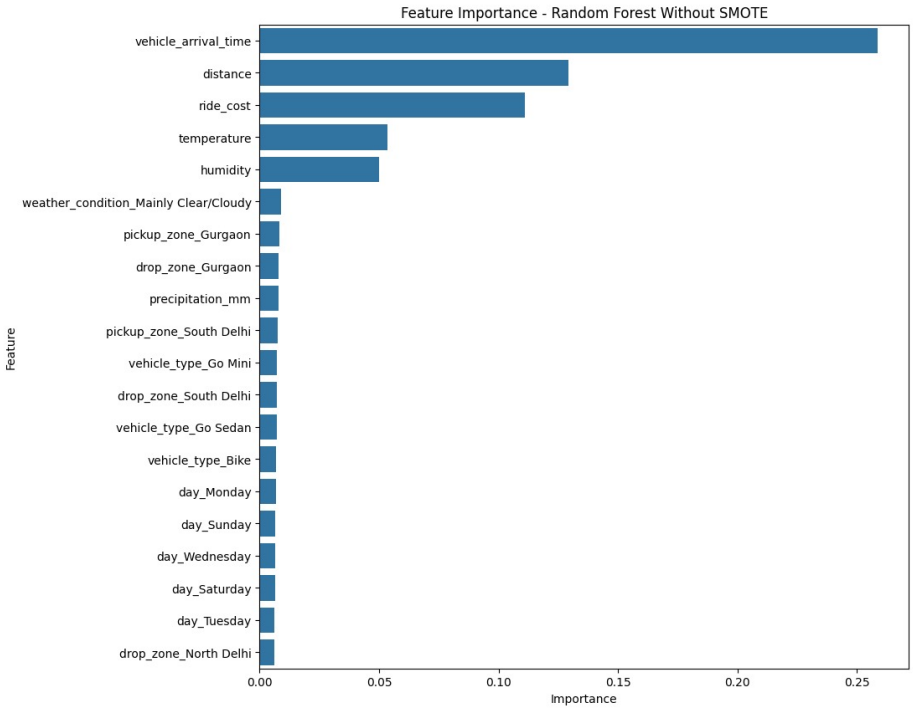
Model 5: Stratified KFold XGBoost (No SMOTE)

	precision	recall	f1-score	support
0	0.98	0.99	0.99	18600
1	0.47	0.34	0.39	447
accuracy			0.98	19047
macro avg	0.73	0.66	0.69	19047
weighted avg	0.97	0.98	0.97	19047

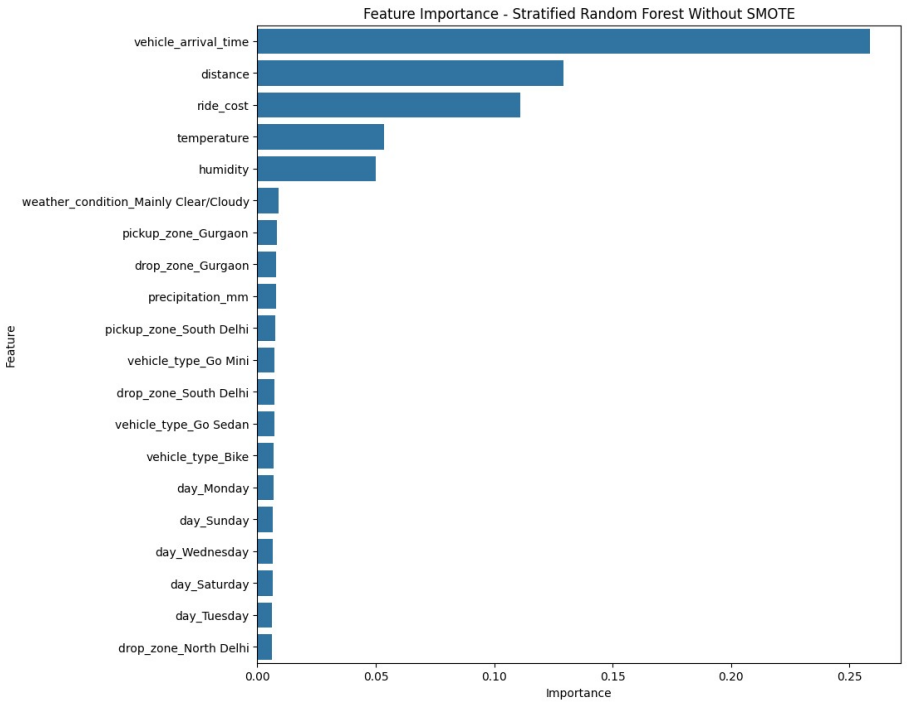
The XGBoost model is 0.98 accurate but has poor Class 1 performance. Its 0.34 recall means it misses over two-thirds of the minority cases, despite strong 0.99 f1-score for Class 0.

3.2 Feature Importance

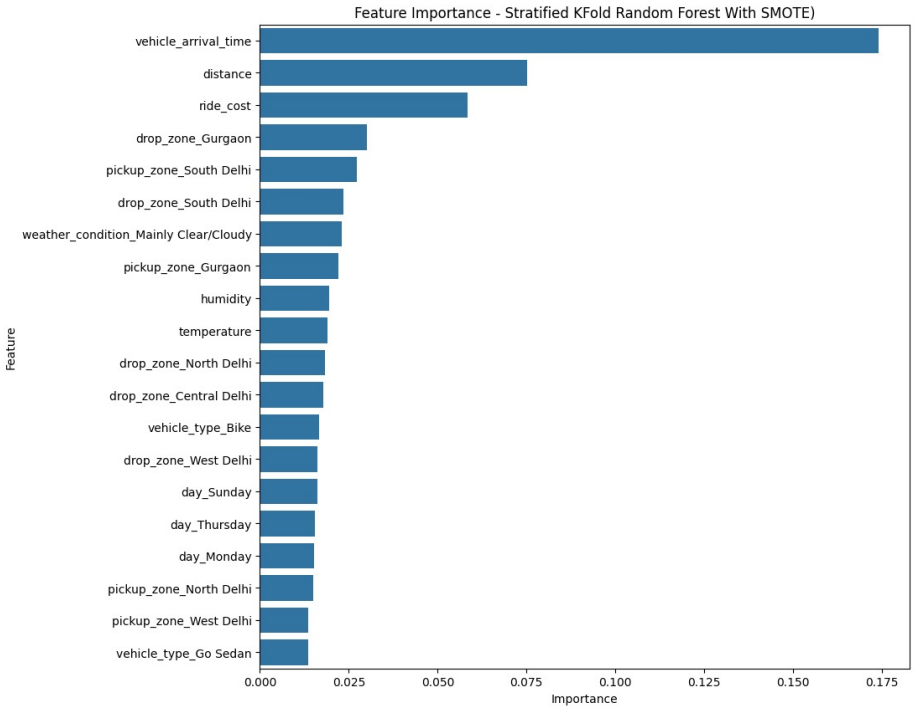
Model 1 - Random Forest Without SMOTE



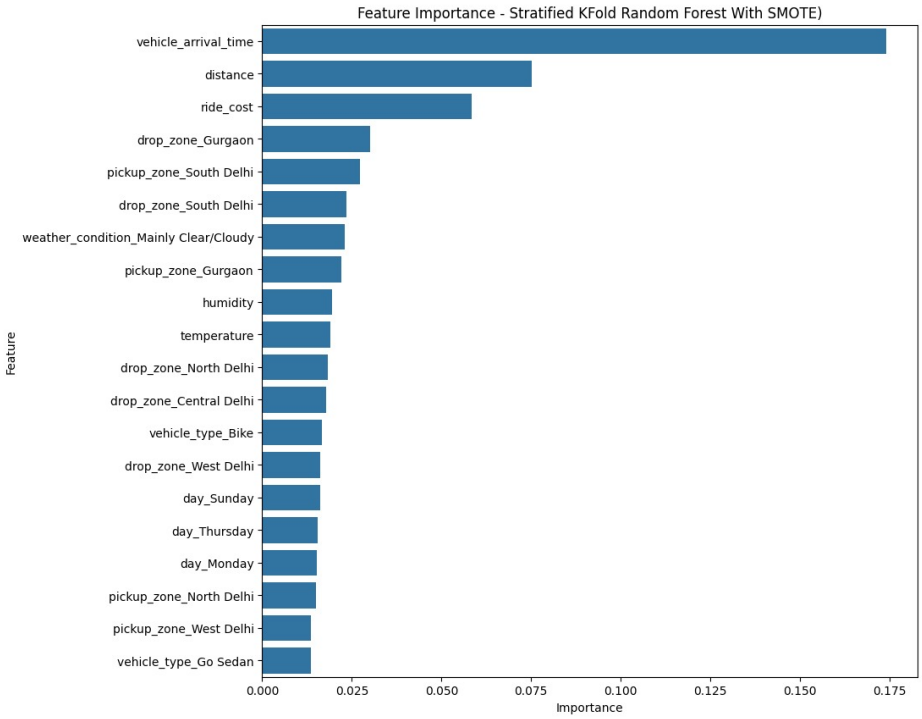
Model 2 - Stratified Random Forest Without SMOTE



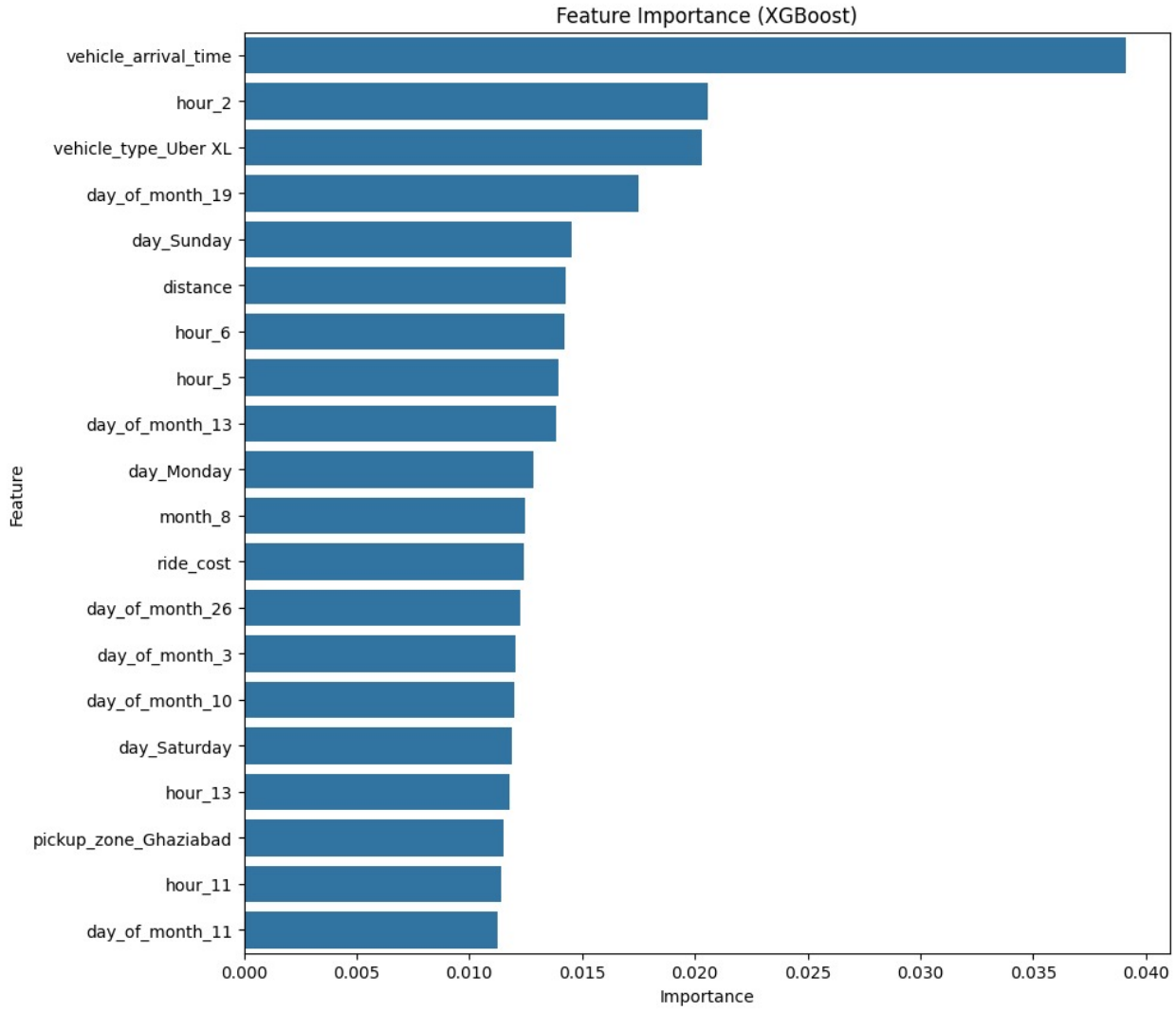
Model 3 - Random Forest With SMOTE



Model 4- Stratified KFold Random Forest With SMOTE



Model 5: Stratified KFold XGBoost (No SMOTE)



3.3 Conclusion on Performance

The baseline model currently acts as a high-confidence filter. Its **high precision** means any warning that it gives is highly reliable, however due to its **low recall** means it fails as an effective *early warning system* by missing the majority of problem cases.

Furthermore, despite the implementation of Class Weights and SMOTE technique, Recall refuses to go up more than 0.34, which probably implies that the model has concluded that 68% of our data is currently unpredictable with the features that we currently have without rising reduction in precision.

SMOTE could not fix the problem because most of the cancellations in the model are all defined by High Wait Time, so SMOTE just created a bunch of new synthetic rides that also have **High Wait Time** which didn't change its behavior at all.

4. Next Steps and Recommendations

The path forward must prioritize increasing the model's ability to identify the rare cancellation event.

1. **Improve Recall via Hyperparameter Tuning**: Systematically tune the Random Forest's parameters (e.g., max_depth, n_estimators) to make the model less conservative and increase its sensitivity to the positive class.
2. **Explore Alternative Models:** Test more powerful, modern algorithms such as LightGBM, which are specifically designed to handle imbalanced datasets and complex non-linear relationships, and may yield better Recall.
3. **Implement Sampling Techniques for XGBoost:** Experiment with data re-sampling techniques like SMOTE (Synthetic Minority Over-sampling Technique) or Under-sampling the majority class to artificially balance the training data, further improving the model's ability to learn the characteristics of a cancellation.

4. **More Advanced Feature Engineering:** The only other way is to add new data. The model is practically missing additional valuable information to make more accurate decisions.

3. (Refinement of the Provided Code)

Numerical Features (distance, ride_cost, etc.) are appropriately scaled (e.g., using StandardScaler).
Categorical Features (vehicle_type, pickup_zone, etc.) are converted to numerical format (e.g., using OneHotEncoder).
The pipeline only **learns** the transformations (like the mean and standard deviation for scaling, or the unique categories for encoding) from the **training data** (X_train) and applies them cleanly to the test data (X_test), preventing any data leakage.

4. Related Works Reviewed

(a) Predicting Uber Ride Cancellations
This machine learning project addresses the critical challenge of ride-hailing cancellations, aiming to develop a high-performance predictive model that flags high-risk bookings before the ride begins. By predicting cancellations, ride-hailing platforms can proactively dispatch drivers, minimize wasted effort, and improve customer satisfaction.

(b) Cancellations-and-how-data-science-can-help
This article, published by Spare Labs, provides a practical, business-focused perspective on the negative impacts of ride-hailing cancellations and demonstrates how predictive modeling can be used to mitigate them. It serves as excellent validation for the goals of your current project.

(c) Predictive Modeling for Uber Ride Cancellation and Price Estimation: An Integrated Approach
The core objective of this project is to build a Machine Learning-based Early Warning System to predict a specific, high-value problem: customer cancellations caused by driver delay or inaction ("Driver is not moving towards pickup location"). The goal is to maximize the model's Recall to catch potential issues, allowing the platform to proactively intervene.

Takeaway for our project:

From the above information, we have been able to formulate several key takeaways. Some of them are as follows:

1. **Data Leakage was the Primary Threat:** The single most critical finding was the initial data leak. The Driver_ratings_Not Rated and customer_ratings_Not Rated feature showed an impossibly high correlation with is_cancelled. We correctly identified that a "Not Rated" status was a *result* of a cancellation, not a *cause*. This taught us that **causality must be interrogated before correlation.** Fixing this by engineering a *historical*, point-in-time rating feature was the most important step in building a valid model.
2. **Model Value is Built on Feature Quality:** We proved that raw data is not enough. The two most valuable engineering steps were:
    a. **Location Zoning:** We transformed 176+ unique string locations (e.g., "Palham Vihar") into 10-12 categorical "zones" (e.g., "Gurgaon"). This solved the "curse of dimensionality" (where One-Hot Encoding would have created 350+ sparse columns) and made the data interpretable for the model.

    b. **External & Proxy Features:** We enriched the dataset with external weather data (via Meteo API). The models *immediately* identified these (especially temperature and humidity) as top-10 predictors, proving their value in explaining *why* wait times might be high.

3. **Gradient Boosting is Superior for This Problem:** The switch from Random Forest to XGBoost (a Gradient Boosting model) was the key. The Random Forest model was "stubborn" and hit a hard wall at **0.33 Recall**.
    a. The XGBoost K-Fold Cross-Validation *proved* it could achieve **0.51 Recall**.
    b. The Feature Importance plots showed why Random Forest only saw "obvious" features (wait_time, cost), while XGBoost successfully identified subtle, nuanced patterns (like hour_2, pickup_zone_Ghaziabad, weather_condition_Drizzle).

    The sequential, error-correcting nature of Gradient Boosting is fundamentally better suited for finding the complex, multi-variable interactions in this dataset.

4. **A Cautious Model is Still Valuable:** Our final Random Forest and initial XGBoost models consistently produced high-precision, low-recall results (e.g., Precision: 1.00, Recall: 0.34). While our goal was to improve recall, this in itself is a valuable business tool as sometimes a tool with 1.0 recall is more advantageous than 0.80 precision and 0.80 recall.

    We successfully built a model that is **100% correct** when it flags a ride as a cancellation risk. For a business, this "cautious but trustworthy" model is perfect for confidently acting (like sending a discount coupon) without wasting resources on false positives.

5. Comprehensive Summary

1. Justify your datasets:

We have gone through several datasets in an attempt to find a suitable alternative, however this dataset is the only one that we could find containing samples of cancelled rides. Some of the other datasets that we've explored include: Uber Pickups in New York City, Uber Fares, Uber Rides, etc.

2. What're your contributions compared to others?

Due to the extend of EDA, pre-processing and Feature Engineering that we have undergone for our problem, we have managed to transform the dataset from someone providing a vague resemblance of relevancy due to their low Pearson coefficient score to a dataset and a model with better Precision and F1-score compared to others who've attempted ride cancellation classification on the same dataset such as these attempts by Denver Magtibay and Rohit Singh Bani.

Furthermore, despite needing some improvements to our recall, our model has significant contributions due to being able to achieve a precision of 1.0 is very valuable for making very high risk business decisions for business managers and decision board as it practically guarantees a very high rate of guarantee in its predictions even though it won't capture every cases.