Time Series Modeling Summary (Technical Report)

As the first step before starting the modeling, we examined the time series patterns using a time series plot. The plots showed that both sale count and redemption count were stationary, with no clear trend. However, they clearly displayed a seasonal pattern that repeats annually (a 365-day cycle), with a bell-shaped curve. This makes sense, as people tend to travel more during the summer, leading to higher sales.

During the preprocessing step, I noticed that missing values in the sale and redemption counts were filled with zeros. Although interpolation is commonly used, using zeros in this case is reasonable, as these may represent days when the business was closed.

Next, I considered the base model. The base model is purely deterministic—it uses the seasonal average by the day of the year and returns a single forecast value for each time step. I improved this logic by grouping the seasonal component based on month, quarter, and day of the week. To incorporate uncertainty, I calculated both the mean and standard deviation of the seasonal component within each group and included a 95% confidence interval.

For evaluation, I used RMSE and MSE in addition to MAPE. Since the target variable contains zero values, MAPE is not ideal. The MAPE formula divides by the actual value, which can lead to extremely large values or infinity when the actual value is zero.

The updated model showed slightly better results. However, I found that using more advanced models improved the forecast significantly. I tried XGBoost, which performed much better. In this model, I included lag features, rolling means, and rolling standard deviations. This was confirmed by lower RMSE and MSE values, as well as improved time series plots.

Several machine learning models were developed to forecast sale count. Given the clear seasonal pattern, I began with a classical time series model—SARIMA—and then tried more advanced models like XGBoost, Prophet, and DeepAR. While SARIMA (with a 365-day period) and DeepAR captured seasonality well, they were more computationally expensive. Among all models, XGBoost produced the best results. Given the trade-off between computational cost and accuracy, XGBoost presented a pragmatic balance. XGBoost outperformed others such as Prophet and DeepAR, likely due to its ability to capture complex nonlinear relationships and temporal dependencies.

If we had access to additional information, such as weather data, holidays, school vacations, or promotional days, we could further improve the models' predictive accuracy.