Business Objective

The primary business objective, as given in the problem statement, is to predict the possible number of scanned receipts for a given future month (specifically for each month of 2022). This forecasting could serve multiple business purposes:

- **Marketing and Strategy**: If there are expected peaks or troughs, marketing and promotional activities can be adjusted accordingly.
- **Operational Readiness**: On-ground teams or support teams can be prepared in advance if high volumes are anticipated.
- **Financial Forecasting**: Predicting operational metrics can feed into revenue and cost forecasting models.

System Performance

- **Latency**: For a forecasting task, real-time predictions are not crucial. Therefore, a batch processing approach would be sufficient.
- **Accuracy**: Since the forecast will be used for business planning, a balance between precision and generalization is essential. Overly precise models might be overfit to past data and may not generalize well. A model that captures the general trend and seasonality would be more valuable.
- **Robustness**: Given the potential scale and importance of this system, it should be resilient to data anomalies or missing data.

ML Objective

- **Goal**: Minimize the forecasting error between the predicted number of scanned receipts and the actual number of scanned receipts for each month in 2022.
- **Evaluation Metric**: Root Mean Squared Error (RMSE) is a good metric for this problem.

System Inputs and Outputs

- **Inputs**:
  - Date: Daily date for the entire year of 2021.
  - Number of scanned receipts for each day in 2021.

- **Outputs**: Predicted number of scanned receipts for each month in 2022. Confidence intervals for the predictions can also be provided, which give a range in which the true value is likely to fall.

Approach

I have used January-October 2021 data as the training data and November-December 2021 data to test and compare the different algorithms.

Approach 1: Naive Approach

The naive approach predicts the next day's receipt count to be the same as the current day's receipt count.

This approach is straightforward and requires no computation. It's often used as a benchmark to compare with more sophisticated models.

Advantages:

- Easy to implement.
- No need for historical data.

Disadvantages:

- Doesn't consider any seasonality or trend in the data.
- May not be accurate if the data has variability or if there are known external factors affecting receipt counts.

Approach 2: Exponential Smoothing

We try out different exponential smoothing models to predict the receipt count. The models are:

1. Simple Exponential Smoothing
2. Holt's Linear Trend Model
3. Holt-Winters Seasonal Model

Advantages:

- Simple approach to forecasting and relatively easy to understand.
- Acts as a good starting point before transitioning to more complex models.
- Provides a good baseline to compare with other models.

Disadvantages:

- Doesn't consider external factors.

Note: Only the "Simple Exponential Smoothing" doesn't account for seasonality or trend. Holt's Linear models for trend and Holt-Winters captures both trend and seasonality.

Approach 3: ARIMA

We try out different ARIMA models to predict the receipt count. The models are:

- ARIMA
- SARIMA

Advantages:

- Captures autocorrelations in time series data.
- Flexible with parameters to handle non-seasonal (ARIMA) and seasonal data (SARIMA).
- Considers the trend in the data.

Disadvantages:

- Requires careful parameter tuning.
- Doesn't consider external factors.
- Assumes data is stationary or can be made stationary.

Approach 4: Facebook Prophet

We try out Facebook Prophet to predict the receipt count.

Advantages:

- Designed to handle daily time series data with strong seasonal patterns.
- Automatically detects seasonality's.
- Can include holidays and special events which might affect the predictions.

Disadvantages:

- It might not perform as well on non-daily data or data without strong seasonality.
- Like the other models, it doesn't inherently account for external factors, but unlike other models, Prophet provides an intuitive way to include known future external events.

Results

Based on my experiments I found that using the Holt-Winters model gave the least error and used it as my final model.

There are two models with the difference with the damped parameter. Why are there two models? Why do we use damping?

1. **Realistic Forecasts**: In many real-world scenarios, it's unlikely for a trend to continue indefinitely at the same rate. For example, if sales of a product are increasing, they might not keep increasing forever at the same rate. After a certain point, the growth might slow down. Damping takes this into account.

2. **Avoid Over-optimistic or Pessimistic Predictions**: Without damping, the model could make overly optimistic (for upward trends) or overly pessimistic (for downward trends) predictions for long-term forecasts.

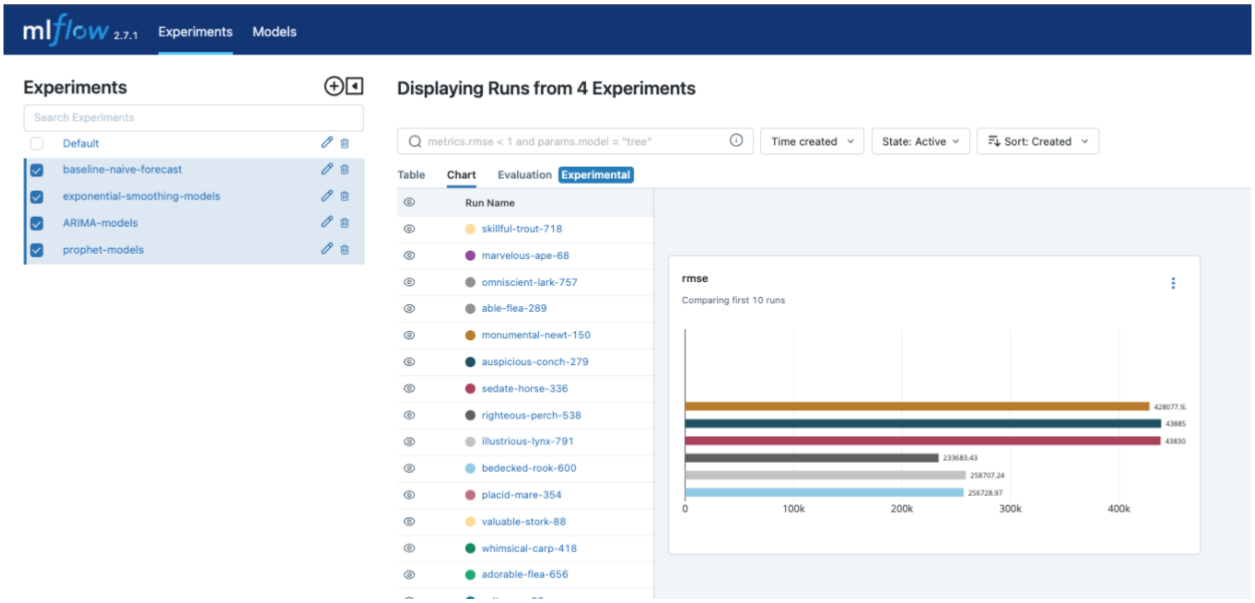
3. **Stability**: Damped models often provide more stable long-term forecasts, especially when the data has some inherent variability or noise.

Scale it for production or industry-level deployment

The objective is to ensure that the model is robust, performs efficiently, and integrates seamlessly with other production systems. Here are the steps you can take:

1. **Data Pipeline Integration**:
    - Ensure that the model can easily access and process the data it needs. This often involves integration with data warehouses, databases, or other data storage solutions.
    - Use ETL (Extract, Transform, Load) processes to automate data preparation and feature engineering, ensuring data quality and consistency.
2. **Infrastructure and Deployment**:
    - Consider deploying the model as a microservice using tools like Docker. This encapsulates the model and all its dependencies in a consistent environment.
    - For cloud solutions, AWS SageMaker, Google Cloud AI Platform, or Azure Machine Learning can be considered.
3. **Batch vs Real-Time Forecasting**:
    - Determine if forecasts are needed in real-time or can be generated in batches. Real-time forecasting may require a more robust infrastructure.
    - Use message queues (e.g., Kafka, RabbitMQ) if you need to handle streaming data for real-time predictions.
    - Ideally for our use we would be utilizing batch time forecasting since we need yearly forecast.
4. **Monitoring and Alerts**:
    - Monitor the performance and health of your deployed model. Tools like Grafana, Prometheus, or cloud-specific solutions can help.
    - Implement alerting mechanisms to notify you of any issues or significant drops in model performance.
5. **Version Control**:
    - Use model versioning to keep track of different model iterations. This allows for easy rollback to previous versions if a new version has issues.
    - Platforms like MLflow or DVC can help with model versioning and experiment tracking.
6. **Documentation and Maintenance**:
    - Document the deployment architecture, data flow, model versions, and other relevant details.
    - Regularly maintain and update the system to accommodate new data sources, changes in data quality, or other operational changes.

Screenshots



Experiments

Search Experiments

□ Default
☑ baseline-naive-forecast
☑ exponential-smoothing-models
☑ ARIMA-models
☑ prophet-models

Displaying Runs from 4 Experiments

metrics.rmse < 1 and params.model = "tree"

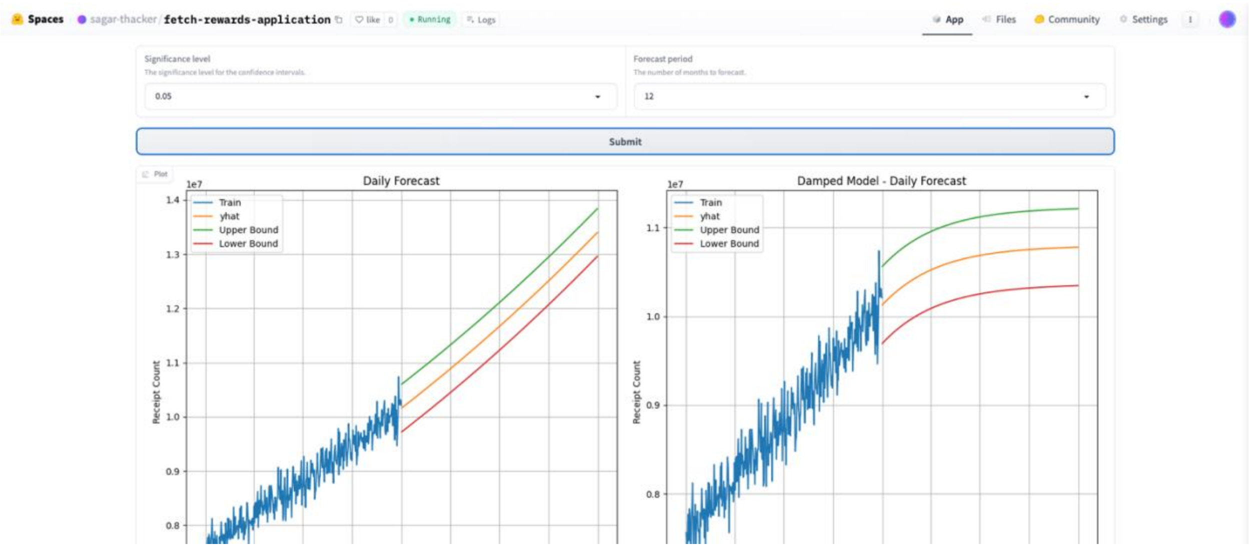
Time created

State: Active

Sort: Created

Run Name
skillful-trout-718
marvelous-ape-68
omniscient-lark-757
able-flea-289
monumental-newt-150
auspicious-conch-279
sedate-horse-336
righteous-perch-538
illustrious-lynx-791
bedecked-rook-600
placid-mare-354
valuable-stork-88
whimsical-carp-418
adorable-flea-656

Figure 1: MLflow UI



Significance level
The significance level for the confidence intervals.
0.05

Forecast period
The number of months to forecast.
12

Figure 2: HuggingFace Spaces