Model monitoring is essential to track changes in data and shifts in relationships between data and labels. It will ensure that the model performance remains consistent over time.

1) Establish baseline

First, we need to collate the statistical distributions of the features as well as the relationship between the feature variables and the labels for the training data. This will provide us a baseline for comparison. The baseline represents the context the model was trained on. Key performance metrics such as precision, recall, F1-score etc of the current model should be recorded.

2) Monitoring

All new predictions made by the model must be logged and stored in a database for evaluation. We should obtain the true labels for the predictions and store them as well if possible. This might be challenging as they might not be readily available. Key metrics like AUC-ROC, F1-Score, mean-squared error can then be calculated for the newly predicted data. Depending on the context and label availability, metrics can be calculated in real time or in batches. There are tests which can be used to track data and concept drift. For categorical features, tests like Chi-Square test or Population Stability Index can be used. For numerical features, tests like Kolmogorov-Smirnov test can be used. To track the relationship between features and labels we can use methods like SHAP or LIME. The metrics we use should be dependent on our data. Furthermore, we can use drift detection frameworks like Apache Kafka.

3) Re-Training Script/Pipeline

Depending on the context, we can establish thresholds for acceptable model performance. There might be instances where model accuracy drops due to unforeseen events and other times where there is a gradual change in patterns. To accommodate for all cases, we should implement model versioning. When the model metrics drops below a certain threshold, we might need to retrain the model. If possible, we can use the newly collected data to re-train the model to represent new relationships. If we are unable to retrain the model due to limited data or labels, we can explore other methods such as transfer learning and semi-supervised learning.

4) Dashboards

To facilitate the above, we can use existing model monitoring tools to monitor drifts and decide when we need to perform retraining. We can use tools like AWS SageMaker, MLflow to track performance. Dashboards can display key metrics like drift scores, and retraining status, providing a clear overview of the model’s health.

Relevant parties can get notified when model performance drops below a certain threshold.