Model Monitoring & Observability

CS 203: Software Tools and Techniques for AI

Prof. Nipun Batra, IIT Gandhinagar

The "Silent Failure" of ML

Traditional Software fails loudly:

- `NullPointerException`, `500 Internal Server Error`.
- You know immediately.

ML Models fail silently:

- The API returns `200 OK`.
- The prediction is `0.85`.
- But the prediction is **wrong** because the world has changed.

Monitoring is how we detect silent failures.

Types of Drift

1. **Data Drift** (Input changes):
    - Training data: High quality images.
    - Production data: Blurry, dark images.
    - $P(X)$ changes.
2. **Concept Drift** (Relationship changes):
    - Housing Price Model trained in 2019.
    - 2020: COVID hits, market behavior changes.
    - Inputs (X) are same (sq ft, rooms), but Output (Y) relation changes.
    - $P(Y|X)$ changes.
3. **Label Drift** (Target changes):

Detection Strategies

1. **Reference vs. Current Distribution**
   - Compare `Training Set` (Reference) vs `Last 24h Production Data` (Current).
2. **Statistical Tests**:
   - **Numerical**: Kolmogorov-Smirnov (KS) test, Wasserstein Distance, KL Divergence.
   - **Categorical**: Chi-Square test, Population Stability Index (PSI).
3. **Rule-based**:
   - "If percentage of nulls > 10% -> Alert"
   - "If mean input value > 5 std dev from train mean -> Alert"

Tooling: Evidently AI

Open-source Python library for evaluating, testing, and monitoring data/models.

```
pip install evidently
```

Generate a Dashboard:

```
import pandas as pd
from evidently.report import Report
from evidently.metric_preset import DataDriftPreset

ref_data = pd.read_csv("train.csv")
curr_data = pd.read_csv("production_log.csv")

report = Report(metrics=[
    DataDriftPreset(),
])

report.run(reference_data=ref_data, current_data=curr_data)
```

Tooling: Alibi Detect

Focused on outlier detection, adversarial detection, and drift detection.

```
from alibi_detect.cd import KSDrift

# Initialize drift detector with reference data
cd = KSDrift(X_ref, p_val=0.05)

# Predict drift on new batch
preds = cd.predict(X_new)

print(f"Drift detected: {preds['data']['is_drift']}")
```

Monitoring Architecture

1. **Prediction Service**: Logs inputs/outputs (async) to storage.
2. **Storage**: Logs stored in Data Warehouse (BigQuery/Postgres) or Object Store (S3).
3. **Analyzer**: Scheduled job (Airflow) runs daily.
    - Pulls logs.
    - Calculates drift metrics.
    - Stores metrics in Time Series DB (Prometheus).
4. **Dashboard**: Grafana visualizes metrics over time.
5. **Alerter**: Slack/Email alert if drift > threshold.

Prometheus & Grafana

Prometheus: Time-series database designed for monitoring.

Grafana: Visualization UI.

Exposing Metrics in Python:

```
from prometheus_client import start_http_server, Summary, Counter

REQUEST_TIME = Summary('request_processing_seconds', 'Time spent processing request')
PREDICTION_COUNTER = Counter('predictions_total', 'Total predictions')

@REQUEST_TIME.time()
def process_request(t):
    PREDICTION_COUNTER.inc()
    # ... logic
```

Feedback Loops

Drift detection is a proxy. The *gold standard* is actual performance.

Challenge: Ground truth comes late (delayed labels).

- Credit Risk: Know default status after months.
- E-commerce Recs: Know click/purchase immediately.

Logging Feedback:

- Log `prediction_id` with prediction.
- Later, log `prediction_id` + `actual_outcome`.
- Join them to calculate Accuracy/F1 over time.

Lab: Drift Detection Dashboard

Objective: Simulate model decay and detect it.

Steps:

1. **Train**: Train a model on "Old" data (e.g., Bike Sharing 2011).
2. **Simulate Serving**: Predict on "New" data (Bike Sharing 2012) batch by batch.
3. **Monitor**:
    - Calculate KS-test p-values for key features for each batch.
    - Use **Evidently AI** to generate HTML reports.
4. **Visualize**: Plot drift scores over time.

Resources

- **Evidently AI Tutorials**: evidentlyai.com
- **Made With ML (Monitoring)**: madewithml.com/courses/mlops/monitoring/
- **Google Cloud Architecture**: "Architecture for MLOps using TFX, Kubeflow Pipelines, and Cloud Build"

Questions?