Resilience Testing

Performance Evaluation Under Distribution Shifts

July 5, 2025

Outline

What is Model Resilience?

Definition

Resilience is the ability of a model to maintain accurate performance despite changes in input data distribution or external factors.

Why Is Resilience Important?

- Real-world data distributions constantly change
- Economic conditions evolve over time
- Customer behaviors shift
- Regulatory environments change
- Model performance can degrade if not resilient to these changes

Key Insight

Even well-performing models can fail when deployed if they lack resilience to distribution shifts

Types of Distribution Shifts

Covariate Shift

- Input distribution $P(X)$ changes
- Relationship $P(Y|X)$ remains the same
- Example: Income distribution shifts but impact on default risk remains consistent

Concept Drift

- Relationship $P(Y|X)$ changes
- Example: Same income level now indicates different default risk

Label Shift

- Target distribution $P(Y)$ changes
- Impacts conditional probability $P(X|Y)$
- Example: Overall default rate changes

Subpopulation Shift

- Relative proportions of data segments change
- Example: Higher proportion of new customers vs. existing ones

Investigating Model Resilience

Steps for Investigating and Improving Model Resilience:

1. **Apply Distribution Shift Scenarios**
    - Simulate conditions that may occur in deployment
    - Analyze when and how performance declines
    - Identify potential vulnerabilities
2. **Assess Variability and Segment Performance**
    - Evaluate across different data segments
    - Identify inconsistencies in performance
    - Detect high-variability areas
3. **Determine Impactful Variables**
    - Identify key variables driving performance degradation
    - Quantify feature drift magnitude
    - Prioritize variables for remediation
4. **Enhance the Model**
    - Address identified weaknesses
    - Apply data-centric and model-centric approaches
    - Validate improvements under simulated shifts

Simulating Distribution Shifts

Approach

MoDeVa provides various scenarios to simulate shifts between training/testing distributions and expected deployment distributions.

Simulation Strategy:

- Gradually increase proportion of "drifted" samples
- Observe the performance degradation curve
- Compare metrics under original vs. shifted distributions
- Identify thresholds where performance becomes unacceptable

Goal

Understand which aspects of distribution shifts most affect the model and how it might perform in challenging real-world conditions

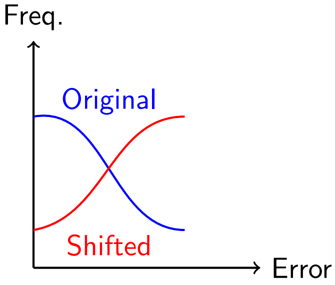
Scenario 1: Drift to Worst Performing Samples

Idea:

- Identify samples with poorest model performance
- Measured by error metrics or mispredictions
- Simulate drift by increasing their proportion

Rationale:

- Mimics a "worst-case" scenario
- Deployment data may disproportionately resemble "hard" cases
- Reveals model vulnerabilities when facing difficult examples



Distribution shifts toward high-error samples

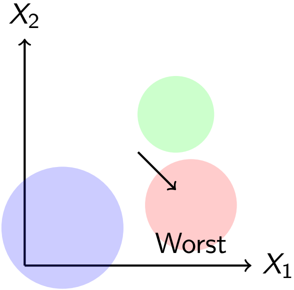
Scenario 2: Drift to Worst Performing Cluster

Idea:

- Use clustering (K-means) on feature space
- Identify cluster with worst performance
- Simulate drift toward this subgroup

Rationale:

- Clustering captures latent subpopulations
- Underperforming clusters may represent niche groups
- Real-world shifts often manifest as changing segment proportions
- Tests model adaptability to different data segments



Shifting toward worst-performing cluster

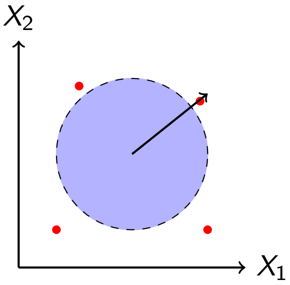
Scenario 3: Drift to Edge Samples

Idea:

- Identify samples at the periphery of data distribution
- Use distance metrics (e.g., Mahalanobis distance)
- Quantify how "far" samples are from distribution center
- Simulate drift toward these boundary cases

Rationale:

- Edge cases are less represented during training
- Behave like out-of-distribution (OOD) samples
- Tests generalization to extreme cases
- Reveals model behavior in unfamiliar territories



Shifting toward boundary/edge cases

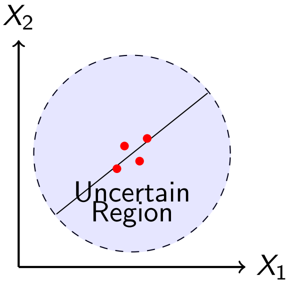
Scenario 4: Drift to Hard-to-Predict Samples

Idea:

- Identify samples the model finds uncertain
- Measured by predictive entropy, low confidence scores, or high error rates
- Simulate scenarios where these samples become more common

Rationale:

- These samples represent model's areas of highest uncertainty
- Tests how well model handles ambiguous cases
- Evaluates potential for uncertainty-aware approaches
- Reveals fundamental model limitations



Shifting toward decision boundary/uncertain regions

Resilience Assessment in MoDeVa

```python
# Create a testsuite that bundles dataset and model
from modeva import TestSuite
ts = TestSuite(ds, model_lgbm)

# resilience assessment using Worst-Sample scenario
results = ts.diagnose_resilience(
    method="worst-sample", metric="MSE")
results.plot()

# resilience assessment using Worst-Cluster scenario
results = ts.diagnose_resilience(
    method="worst-cluster",
    n_clusters=5, metric="MSE")
results.plot()

```

This generates visualizations showing performance degradation under distribution shifts

Additional Resilience Scenarios

```python
# resilience assessment using edge (outer) sample scenario
results = ts.diagnose_resilience(
    method="outer-sample", metric="MSE")
results.plot()

# resilience assessment using hard sample to predict scenario
results = ts.diagnose_resilience(
    method="hard-sample", metric="MSE")
results.plot()

```

These visualizations show how performance degrades as the proportion of challenging samples increases

Key Output

The plots show performance metrics (e.g., MSE) on the y-axis against the proportion of "shifted" samples on the x-axis, creating a degradation curve

Distribution Drift Metrics

Jensen-Shannon Divergence (Population Stability Index)

- Measures similarity between probability distributions
- Identifies variable distribution shifts
- Symmetric version of KL divergence
- Values closer to 0 indicate similar distributions

Wasserstein Distance

- Quantifies cost of transforming one distribution into another
- "Earth mover's distance" interpretation
- Robust to distributions with limited overlap
- Better for continuous distributions

Kolmogorov-Smirnov (KS) Statistic

- Identifies maximum difference between cumulative distributions
- Non-parametric test for distribution equality
- Simple to interpret: maximum vertical distance between CDFs
- Values closer to 0 indicate similar distributions

Analyzing Feature Drift Impact

```python
# resilience assessment using Worst-Sample scenario
results = ts.diagnose_resilience(method="worst-sample", metric="MSE")

# Analyze distribution drift for 10% worst samples
data_results = ds.data_drift_test(
    **results.value[0.1]["data_info"],
    distance_metric="PSI",
    psi_method="uniform", psi_bins=10)
data_results.plot()

```

This analyzes which features experience the most significant drift in the worst-performing samples

Key visualizations include:

- Summary of distribution shift for all features, ranked by PSI
- Marginal density comparison between original and shifted distributions
- Marginal histogram comparison showing bin-level differences

Model Comparison

```python
# Compare resilience between models
tsc = TestSuite(ds, models=[model_lgbm, model_xgb])

# resilience assessment using Worst-Cluster scenario
results = tsc.compare_resilience(
    n_clusters=5, method="worst-cluster", metric="MSE")
results.plot()

```

This compares how different models degrade under the same distribution shift scenario

What to look for:

- Which model degrades more gracefully?
- At which point does each model's performance become unacceptable?
- Are there crossover points where one model becomes better than another?
- Which model shows more consistent performance across different scenarios?

Remediation: Data-Centric Approaches

1. Targeted Data Augmentation

- Focus on regions with poor resilience
- Collect additional samples in weak regions
- Apply active learning to select informative samples

2. Feature Engineering

- Create interaction terms for regions with nonlinear patterns
- Develop domain-specific features for weak areas
- Transform features that experience significant drift
- Design features that are more stable across distributions

Key Principle

Targeted improvements in data quality and representation can significantly enhance model resilience to distribution shifts

Model-Centric Approaches

1. **Local Model Enhancement**
   - Train specialized models for weak regions
   - Implement segment-specific models
   - Use Mixture of Experts (MoE) approach
   - Weight models based on local performance
2. **Architecture Modifications**
   - Incorporate domain knowledge via constraints
   - Use robust loss functions
   - Add calibration layers
3. **Loss Function Adjustments**
   - Weight samples from vulnerable regions higher
   - Implement distribution-aware penalties
4. **Ensemble Strategies**
   - Combine models with different strengths
   - Weight models dynamically based on input
   - Implement model switching based on detected shifts

Implementation Framework

Diagnose

- Run multiple distribution shift scenarios
- Compare degradation patterns
- Analyze feature drift
- Identify worst-case scenarios

Prioritize

- Focus on high-impact features
- Rank scenarios by severity
- Consider business implications
- Balance effort vs. impact

Implement & Validate

- Apply targeted remediation
- Retest under same scenarios
- Measure improvement
- Iterate as needed

Systematic Approach

Improving resilience requires careful diagnosis, targeted interventions, and validation of improvements under simulated distribution shifts

Summary: Model Resilience Testing

- **Understanding Resilience**: A model's ability to maintain performance under distribution shifts is critical for real-world success
- **Distribution Shift Scenarios**: MoDeVa provides multiple methods to simulate realistic shifts and stress-test models
- **Feature Drift Analysis**: Identifying which variables experience significant shifts helps prioritize remediation efforts
- **Model Comparison**: Different models may show varying levels of resilience to distribution shifts
- **Targeted Remediation**: Combining data-centric and model-centric approaches can improve resilience to specific types of shifts
- **Systematic Implementation**: Diagnose, prioritize, implement, and validate in an iterative process

Key Takeaway

Resilience testing should be a standard part of model validation to ensure models perform reliably under the dynamic conditions encountered in real-world deployments.