Reliability Testing

Evaluation of Prediction Uncertainty

July 5, 2025

Outline

What is Model Reliability?

Definition

Reliability in predictive models refers to their ability to produce consistent and trustworthy outputs by accurately quantifying prediction uncertainty.

Why Is Reliability Important?

- Enables risk assessment in predictions
- Critical in high-stakes domains like finance, healthcare, or autonomous systems
- Helps users understand when to trust model outputs
- Identifies when a model is "not sure" about its predictions
- Guides data collection and model improvement efforts

Key Insight

A reliable model is not necessarily more accurate, but it knows when it doesn't know

Sources of Uncertainty in ML Models

Data-Related Uncertainty

- Noisy measurements or labels
- Sparse or limited training data
- Missing features or values
- Data quality issues
- Sampling bias

Model-Related Uncertainty

- Structural limitations
- Parameter uncertainty
- Inherent model limitations
- Optimization challenges

Environment-Related Uncertainty

- Distribution shift
- Novel or unfamiliar inputs
- Adversarial scenarios
- Edge cases not seen during training

Predictive Uncertainty Types

- Aleatoric uncertainty (inherent randomness)
- Epistemic uncertainty (model/knowledge limitations)
- Distributional uncertainty (out-of-distribution data)

Investigating Model Reliability

Steps for Investigating and Improving Model Reliability:

1. **Uncertainty Quantification**
   - Use Conformal Prediction to generate prediction intervals
   - Provide a range within which true values likely fall
   - Establish guaranteed coverage levels
2. **Identification of Less Reliable Regions**
   - Identify high uncertainty regions in feature space
   - Detect areas with outside-of-coverage predictions
   - Analyze patterns in unreliable predictions
3. **Determine Impactful Variables**
   - Identify key variables contributing to uncertainty
   - Quantify feature importance for reliability
   - Analyze feature interactions affecting reliability
4. **Enhance the Model**
   - Address identified weaknesses
   - Apply data-centric and model-centric approaches
   - Improve reliability in targeted regions

Introduction to Conformal Prediction

Concept

Conformal prediction provides a model-agnostic framework for constructing prediction intervals with guaranteed coverage under minimal assumptions.

Key Properties:

- Distribution-free coverage guarantees
- Requires only that data are exchangeable (i.i.d. is sufficient)
- Works with any base model or predictor
- For a desired miscoverage rate α, produces intervals containing the true value with probability $1 - \alpha$
- Applicable to both classification and regression

Core Idea

Quantify how different a new test point is from the training data using a nonconformity score

Full Conformal Prediction

Full Conformal Prediction is the most faithful implementation of the conformal prediction framework. It provides finite-sample, distribution-free validity guarantees under the assumption that the data are exchangeable.

Given a training dataset

$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$$

and a new input x_{n+1}, the goal of full conformal prediction is to construct a prediction set $\Gamma_n(x_{n+1})$ such that:

$$\mathbb{P}(y_{n+1} \in \Gamma_n(x_{n+1})) \geq 1 - \alpha,$$

where $\alpha \in (0, 1)$ is a user-defined miscoverage rate.

Full Conformal Prediction Algorithm (1)

Step 1: Define a Nonconformity Score

Let $A(f, (x, y))$ be a nonconformity score function. For each $i = 1, \ldots, n$, train a model f_i on $\mathcal{D}_{-i} = \mathcal{D} \setminus \{(x_i, y_i)\}$, and compute:

$$\alpha_i = A(f_i, (x_i, y_i)).$$

For example:

- Regression: $\alpha_i = |f_i(x_i) - y_i|$
- Classification: $\alpha_i = 1 - \hat{P}_{f_i}(y_i \mid x_i)$

Step 2: Evaluate Candidate Labels

For a new input x_{n+1}, and for each candidate label or value $y \in \mathcal{Y}$, augment the dataset:

$$\mathcal{D}' = \mathcal{D} \cup \{(x_{n+1}, y)\}.$$

Then, for each $i = 1, \ldots, n+1$, train a model on $\mathcal{D}' \setminus \{(x_i, y_i)\}$, and compute nonconformity scores:

$$\alpha_i^{(y)} = A(f_i^{(y)}, (x_i, y_i)).$$

Full Conformal Prediction Algorithm (2)

Let $\alpha_{n+1}^{(y)}$ denote the nonconformity score for the trial point (x_{n+1}, y).

Step 3: Compute p-value
Define the p-value for the candidate y as:

$$p(y) = \frac{1}{n+1} \sum_{i=1}^{n+1} \mathbf{1} \left\{ \alpha_i^{(y)} \geq \alpha_{n+1}^{(y)} \right\}.$$

Step 4: Construct the Prediction Set
The conformal prediction set is:

$$\Gamma_n(x_{n+1}) = \{ y \in \mathcal{Y} : p(y) > \alpha \}.$$

Under the assumption of exchangeability of the data:

$$\mathbb{P}(y_{n+1} \in \Gamma_n(x_{n+1})) \geq 1 - \alpha,$$

holds for any finite sample size n, without distributional assumptions.

Example of Full Conformal Prediction Calculation (1)

We use logistic regression:

$$\hat{P}(1 \mid x) = \sigma(-2 + 1.5x), \quad \text{where } \sigma(z) = \frac{1}{1 + e^{-z}}$$

Training Data and Nonconformity Scores

x_i	y_i	$\hat{P}(y_i \mid x_i)$	$\alpha_i = 1 - \hat{P}(y_i \mid x_i)$
0	0	0.881	0.119
1	0	0.623	0.377
2	1	0.731	0.269
3	1	0.924	0.076

Example: Confidence Level 0.8, Test Input $x = 0.75$

$$z = -2 + 1.5 \cdot 0.75 = -0.875, \quad \sigma(z) \approx 0.294$$

Example of Full Conformal Prediction Calculation (2)

Case 1: Assume $y = 0$

$$\hat{P}(0 \mid x) = 1 - 0.294 = 0.706, \quad \alpha_5^{(0)} = 1 - 0.706 = 0.294$$

Compare with training scores: $\{0.119, 0.377, 0.269, 0.076\}$

$$\Rightarrow \# \geq 0.294 = 1 + 1 \text{ (test)} = 2 \Rightarrow p(0) = \frac{2}{5} = 0.4 > \alpha = 0.2$$

Include $y = 0$ in prediction set.

Case 2: Assume $y = 1$

$$\hat{P}(1 \mid x) = 0.294, \quad \alpha_5^{(1)} = 1 - 0.294 = 0.706$$

$$\# \geq 0.706 = 0 + 1 \text{ (test)} = 1 \Rightarrow p(1) = \frac{1}{5} = 0.2 \not> 0.2$$

Do not include $y = 1$.

$$\Gamma(x = 0.75) = \{0\}$$

Note on Full Conformal Prediction Algorithm

In the example, for simplicity, we are using approximation where we did not refit the model for each test point and candidate value.

Full Conformal Prediction

- Requires refitting model for each test point and candidate value
- P-value calculation for each potential output
- Computationally expensive but theoretically optimal
- Strongest coverage guarantees

Split Conformal Prediction

- Splits data into training and calibration sets
- Train model once on training set
- Compute nonconformity scores on calibration set
- Use empirical quantiles to construct intervals
- Much more computationally efficient
- Slight reduction in statistical efficiency

Algorithm:

1. Split data: D_{train} and D_{cal}
2. Train model on D_{train}
3. Compute scores on D_{cal}
4. Use score quantiles for intervals

Conformalized Residual Quantile Regression (CRQR)

A Powerful Approach for Regression Problems

1. **Base Model Fitting**
   - Fit a base model $f(X)$ to estimate conditional mean
   - Calculate residuals: $r_i = y_i - f(X_i)$
2. **Residual Quantile Regression**
   - Train quantile regression models for lower and upper quantiles
   - Predict residual quantiles: $\hat{q}_{\tau_1}(X)$, $\hat{q}_{\tau_2}(X)$
   - Capture heteroscedasticity (changing variance across feature space)
3. **Conformalization**
   - Compute nonconformity scores on calibration set
   - Adjust quantile predictions to ensure coverage guarantee
4. **Prediction Intervals**
   - For new point X, construct interval:
   - $C(X) = [f(X) + \hat{q}_{\tau_1}(X) - Q_{1-\alpha}(s), f(X) + \hat{q}_{\tau_2}(X) + Q_{1-\alpha}(s)]$
   - Combines flexibility of quantile regression with conformal guarantees

Nonconformity Scores

For Regression:

- Absolute residual: $s_i = |y_i - \hat{f}(X_i)|$
- Normalized residual: $s_i = \frac{|y_i - \hat{f}(X_i)|}{\hat{\sigma}(X_i)}$
- CRQR score: $s_i = \max\{q_{\tau_1}(X_i) - r_i, r_i - q_{\tau_2}(X_i)\}$

For Binary Classification:

- Score based on predicted probabilities:
- $s_i = \begin{cases} 1 - \hat{p}_i & \text{if } y_i = 1 \\ \hat{p}_i & \text{if } y_i = 0 \end{cases}$
- Lower score indicates better conformity with training data

Advantages and Considerations:

- Distribution-free coverage guarantees
- Works with any base predictor
- Split conformal trades statistical efficiency for computational efficiency
- Can capture heteroscedasticity with appropriate scoring functions
- Can be extended to handle covariate shift with weighted approaches

Basic Reliability Assessment in ModEva

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

# reliability assessment using Split Conformal Prediction
results = ts.diagnose_reliability(
    train_dataset="test", test_dataset="test",
    test_size=0.5, alpha=0.1,
    max_depth=5)            # depth for quantile regression model
results.table

```

This generates a table showing prediction interval width and coverage

Key Configuration

alpha=0.1 means we expect 90% of true values to fall within prediction intervals

Understanding Reliability Issues

Two Key Aspects of Conformal Prediction Outputs:

1. High Uncertainty

- Measured by prediction interval width
- For regression: wide intervals indicate high uncertainty
- Flag points with width threshold: $W(X) > Q_{1-\beta}(W)$
- β typically 0.9 for top 10% widest intervals
- For classification: empty or multi-class prediction sets

2. Coverage Violations

- Occurs when true value falls outside prediction interval
- For regression: $y \notin C_{n,\alpha}(X)$
- For classification: true class not in prediction set
- Expected violation rate should match α
- Higher-than-expected violations indicate model issues

Common Patterns Indicating Model Weakness:

- High uncertainty + good coverage: Model is honest about uncertainty
- High uncertainty + poor coverage: Model may be misspecified
- Low uncertainty + poor coverage: Model is overconfident

Analyzing Reliability Issues in MoDeVa

```
# Analyze distribution differences between less reliable
    regions and overall data
data_results = ds.data_drift_test(
    **results.value["data_info"],
    distance_metric="PSI", psi_method="uniform", psi_bins
    =10)

# Display summary of distribution differences ranked by PSI
data_results.plot("summary")
# Visualize density comparison for a specific feature
data_results.plot(("density", "hr"))
```

This ompares the distribution of features in unreliable regions vs. the overall dataset

Key outputs:

- PSI summary: Features ranked by distribution difference
- Density plots: Visual comparison of distributions
- Histogram comparisons: Bin-level differences

Feature Slicing for Reliability Analysis

```python
# Univariate slicing: analyze reliability across a single feature
results = ts.diagnose_slicing_reliability(
    features="hr",                # feature to analyze
    train_dataset="train", test_dataset="test",
    test_size=0.5,  metric="width")
results.plot()

```

This shows how prediction interval width varies across different hours of the day

What to look for:

- Features or feature values with unusually high uncertainty
- Consistent patterns in uncertainty across feature ranges
- Times of day, categories, or value ranges with reliability issues

Multiple Feature Analysis

```python
# Analyze multiple features independently
results = ts.diagnose_slicing_reliability(
    features=(("hr",), ("atemp",), ("weekday",)),
    train_dataset="train", test_dataset="test",
    test_size=0.5, metric="coverage")
# View results for each feature
results.plot("hr")
results.plot("atemp")
results.plot("weekday")
```

This analyzes coverage across multiple features independently

Coverage Analysis

Comparing actual coverage to expected coverage (1-α) reveals regions where the model is systematically over- or under-confident

Feature Interaction Analysis

```python
# Two-dimensional slicing: analyze feature interactions
results = ts.diagnose_slicing_reliability(
    features=("hr", "atemp"),    # feature pair to analyze
    train_dataset="train",
    test_dataset="test",
    test_size=0.5,
    random_state=0)
results.plot()

```

This visualizes how combinations of feature values affect reliability

Benefits:

- Identifies complex interactions affecting model reliability
- Reveals conditional dependencies in uncertainty
- Shows where feature combinations create problematic regions
- Helps detect subtle reliability patterns missed in univariate analysis

Model Comparison

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

# Compare overall reliability metrics
results = tsc.compare_reliability(
    train_dataset="train", test_dataset="test",
    test_size=0.5, alpha=0.1, max_depth=5)
results.table

```

This compares reliability metrics between different models

What to compare:

- Average prediction interval width (narrower is better, if coverage is maintained)
- Actual coverage (closer to target 1-α is better)
- Consistency of coverage across feature space
- Trade-off between interval width and coverage

Supervised Machine Learning for Uncertainty Analysis

```python
# Use Random Forest clustering with prediction interval
    width as target
results = ts.diagnose_residual_cluster(
    dataset="test", response_type="pi_width", metric="MAE",
    n_clusters=10, cluster_method="pam", sample_size=2000,
    rf_n_estimators=100, rf_max_depth=5) # RF parameters
results.table
results.plot()

```

This uses Random Forest proximity to cluster similar samples based on prediction uncertainty

Key outputs:

- Cluster table: Performance metrics for each cluster
- Feature importance: Variables driving uncertainty clusters
- Cluster visualization: Similarity patterns in high-uncertainty regions

Detailed Cluster Analysis

```
# Analyze a specific high-uncertainty cluster
cluster_id = 2  # cluster with high uncertainty

# Compare cluster distribution to overall distribution
data_results = ds.data_drift_test(
    **results.value["clusters"][cluster_id]["data_info"],
    distance_metric="PSI", psi_method="uniform", psi_bins
   =10)
data_results.plot("summary")
data_results.plot(name=('density', 'atemp'))
```

This analyzes the feature distribution patterns of a specific high-uncertainty cluster

Insights from cluster analysis:

- Distinct feature patterns in high-uncertainty regions
- Feature interaction effects not visible in univariate analysis
- Natural groupings of similar uncertainty patterns
- Key drivers of prediction uncertainty clusters

Remediation: Data-Centric Approaches

1. Targeted Data Augmentation

- Focus on high-uncertainty regions
- Collect additional samples in weak regions
- Use active learning to select informative samples
- Prioritize areas with low coverage

2. Feature Engineering

- Create interaction terms for regions with nonlinear patterns
- Develop domain-specific features for high-uncertainty areas
- Transform features to better capture heteroscedasticity
- Add features that help discriminate in uncertain regions

Key Principle

Targeted data improvements in high-uncertainty regions can significantly enhance model reliability

Remediation: Model-Centric Approaches

1. **Local Model Enhancement**
   - Train specialized models for unreliable regions
   - Implement segment-specific models
   - Use Mixture of Experts (MoE) approach
   - Weight models based on local performance
2. **Architecture Modifications**
   - Add capacity in high-uncertainty regions
   - Try alternative modeling frameworks
3. **Loss Function Adjustments**
   - Weight samples from uncertain regions higher
   - Implement reliability-focused penalties
   - Balance overall performance with local improvements
4. **Ensemble Strategies**
   - Combine models with complementary reliability profiles
   - Weight ensemble components based on local uncertainty
   - Implement model switching based on detected uncertainty

Implementation Framework

Diagnose

- Apply conformal prediction
- Identify high-uncertainty regions
- Detect coverage violations
- Analyze feature patterns

Prioritize

- Focus on most unreliable regions
- Rank features by importance
- Consider business impact
- Balance effort vs. improvement

Implement & Validate

- Apply targeted remediation
- Rerun reliability testing
- Compare before/after metrics
- Iterate as needed

Systematic Approach

Improving reliability requires understanding uncertainty patterns, applying targeted interventions, and validating improvements

Summary: Model Reliability Testing

- **Understanding Reliability**: A model's ability to produce consistent outputs with appropriate uncertainty estimates
- **Conformal Prediction**: Framework for creating prediction intervals with guaranteed coverage
- **Reliability Analysis**: Identifying regions with high uncertainty or coverage violations
- **Feature Analysis**: Using slicing and clustering to understand patterns in unreliable predictions
- **Model Comparison**: Different models may show varying reliability profiles
- **Targeted Remediation**: Combining data-centric and model-centric approaches to improve reliability

Key Takeaway

Reliable models provide appropriate uncertainty estimates, allowing users to make informed decisions about when to trust the model's predictions