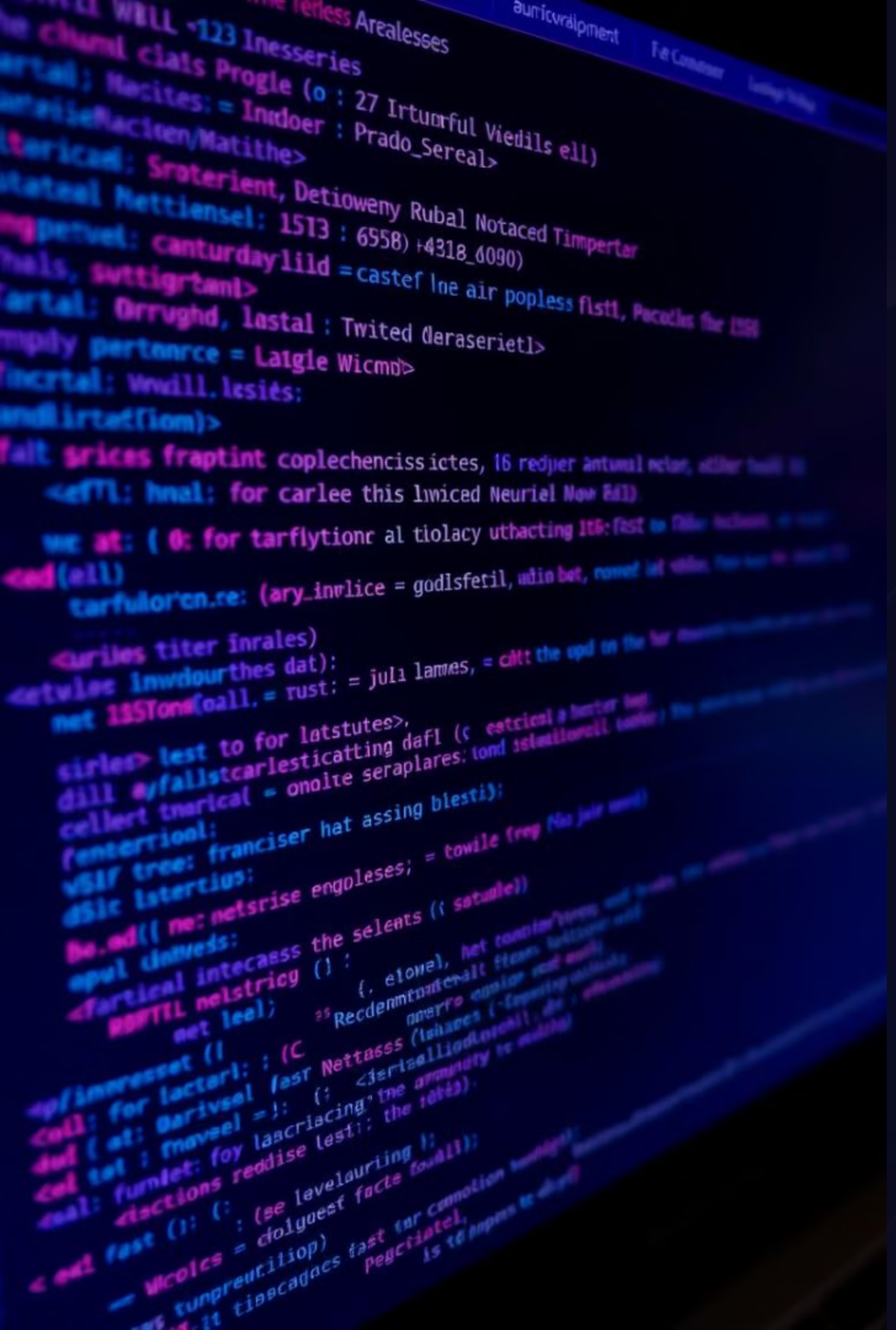
Conformal LSTM Classifier for Time-Series Anomaly Detection

Presented By: Anish Rao

Professor: Devesh Jawla



Project Objective & Scope

Main Objective

Use Julia to Implement a conformal LSTM classifier to detect anomalies

Scope of the Project

Why LSTM?

- A type of model that helps understand data over time
- It remembers patterns of the past to make better future decisions

Why Conformal Prediction?

- Helps us know how sure the model is about its decision
- Instead of guessing a number to detect anomalies, it **calculates a limit** from the data itself.

About NAB

- A large collection of time-based data
- Contains real and fake data
- It's commonly used to test how good anomaly detectors are.

NAB Leaderboard: What Top Models Do

The top-performing models on the NAB leaderboard use a **diverse mix of strategies**, mostly **unsupervised**, to detect anomalies in time series:

HTM (Hierarchical Temporal Memory):
Inspired by the human brain, it learns patterns over time and detects anything that breaks the flow

Bayesian Models:
Use probability to guess how likely a new value is — if it seems very unlikely, it's marked as an anomaly.

KNN-CAD (Nearest Neighbour):
Compares current patterns to past ones — if they look very different, something's probably wrong.

Forecasting Models (like ARIMA):
Try to predict what's coming next based on past trends — big surprises are flagged as anomalies.

Ensemble Models:
Combine the opinions of multiple small detectors to make better overall decisions.

Common Goal: Minimise false alarms while catching real, subtle anomalies — especially in **noisy, unlabelled, real-world data**.

How It Works

Unsupervised (Forecasting)

- **Data:** Only values
- **Training:** predict the next value and minimise the MSE between predicted and actual values.
- **Conformal Prediction:** use residuals (| predicted - actual |) to calculate threshold
- **Evaluation:** Predict next values, flag if residual > threshold

01. Data Preparation

Normalise Data → Split into train/ calibration/ test→ Create windows/sequences

02. Training

Create model → Train model

03. Conformal Prediction

Create residuals → Set threshold (Confidence level)

04. Evaluation

Predict anomalies → Compare to true labels for metrics

Supervised (Classification)

- **Data:** Both values and labels
- **Training:** Classify sequences as normal or anomaly using Sigmoid + Binary Cross-Entropy Loss.
- **Conformal Prediction:** use residuals (| probability - actual |) to calculate threshold
- **Evaluation:** Predict probabilities, flag if residual > threshold

Key Results

Unsupervised model outperformed the supervised one in 4 out of 6 real-world categories

Highest precision

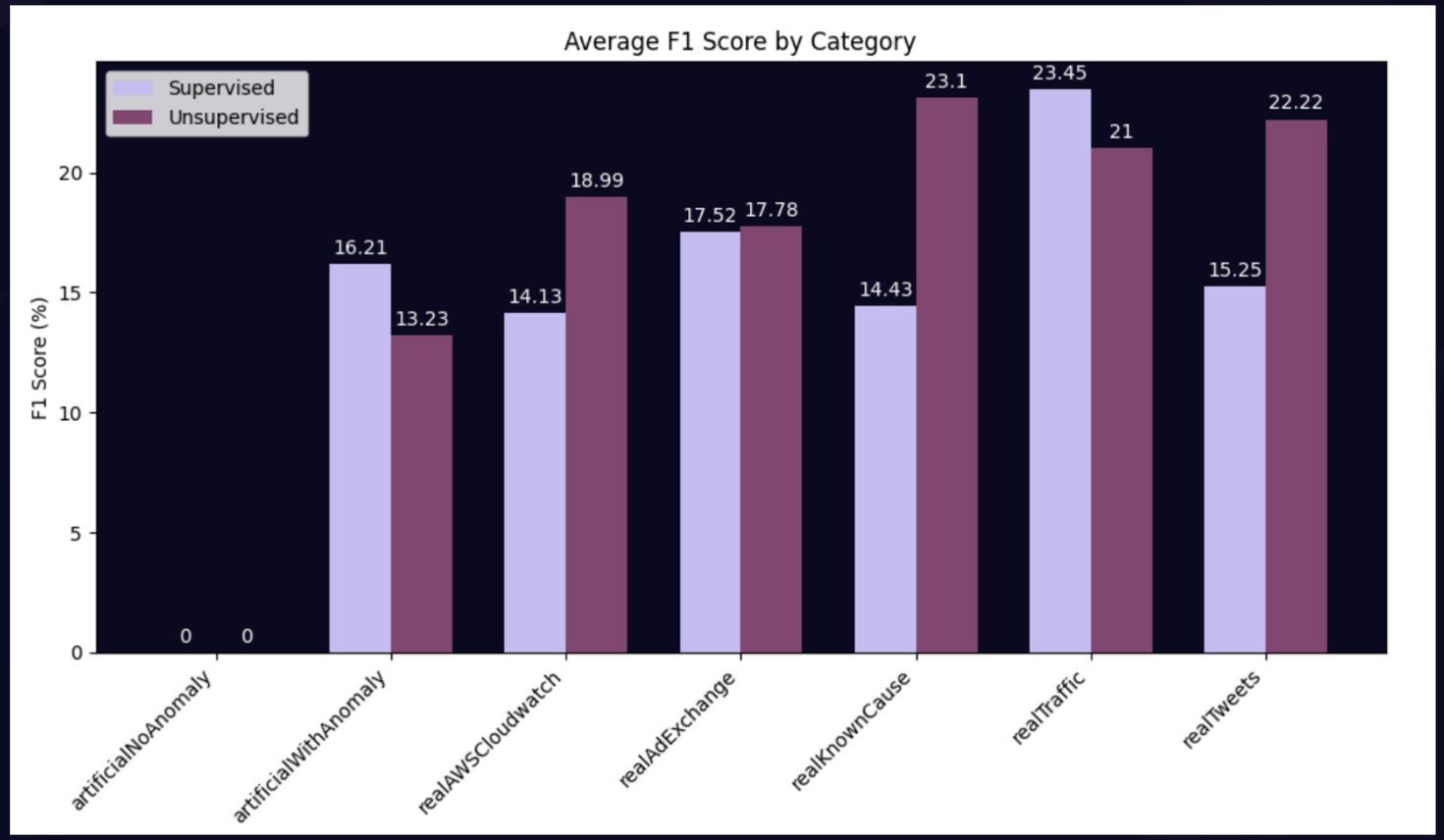
88.89	61.9
Supervised	Unsupervised

Highest recall

81.05	95.79
Supervised	Unsupervised

Unsupervised caught more anomalies, but also had more false alarms.

Supervised was more precise, but missed several real anomalies.



Model	TP (Correct Detections)	FP (False Alarms)	FN (Missed Anomalies)
Supervised	8780	37991	37362
Unsupervised	16585	76082	34569

Conclusion

1. The unsupervised model shows higher overall recall and slightly better F1-score, highlighting its strength in detecting anomalies broadly, while the supervised model achieves slightly higher precision, reducing false alarms.

2. **Unsupervised model** performed better on noisy, real-world data like server metrics and social media.

3. **Supervised model** excelled on clean, synthetic data with clearer anomaly patterns.