REPLY WHITEHALL

LUISS

Forecasting the Unexpected

Anomaly Detection Without a Crystal Ball (Just Spikes)

Developed for Reply and brought to you by the team:

NAPOLETANI A ROCCARASO

Gabriele Rizzo

Annalaura Granata

Daniele Lupico

Introduction to Our Work

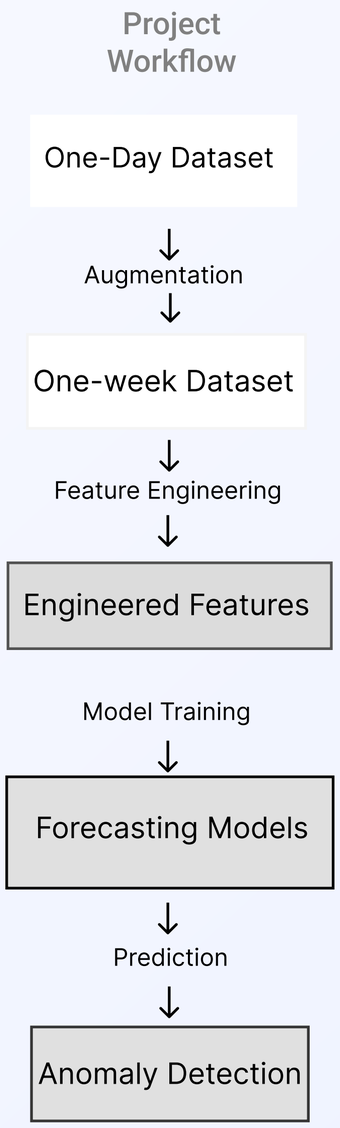
Time Series Analysis & Anomaly Detection

Project Objective

- Transform a one-day transaction dataset into a full week of **realistic data**
- Engineer useful features for improved analysis
- Apply **forecasting** and **anomaly detection** models

Implementation Tools

seaborn scikit-learn tensorflow



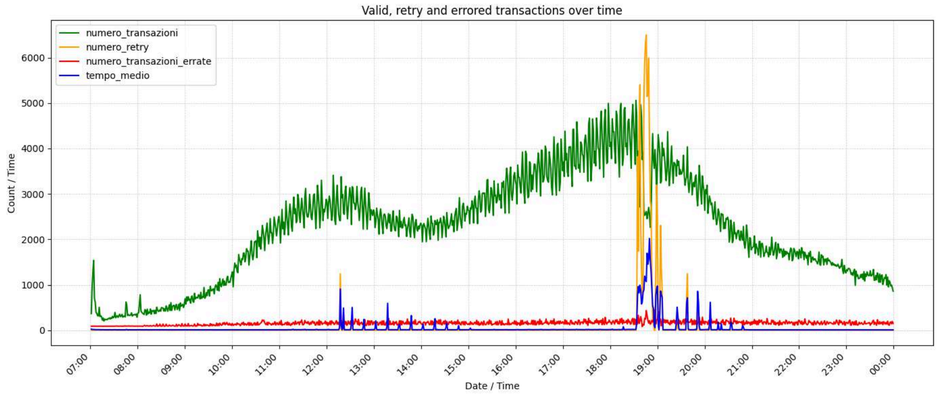
Dataset and Augmentation Strategy

Extending a single-day transaction dataset to a full week of realistic data

Original Dataset

From Client-provided fields, we selected the following:

- numero_transazioni
- tempo_medio
- numero_retry
- numero_transazioni_errate



Augmentation Strategy

One-Day Dataset → One-week Dataset

Predefined Rules

- Augmentation with clipped noise
- Augmentation smoothed outliers, avoiding the model to be too influenced by few extreme spikes
- Pattern preservation

Advanced Techniques

- Rule-based noise injection based on variable probability created labeled test scenarios
- “Slow Day” introduction helped testing model robustness

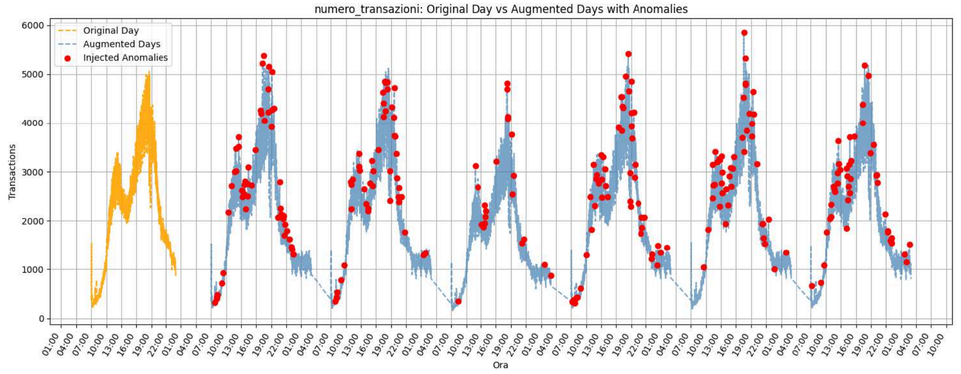
Data Integrity Preservation

Correlation Preservation

- Maintained relationships between variables
- Realistic dependency structures

Rhythm Preservation

- Business cycle simulation: handled interval 00:00-02:59 and represented night time business pause 03:00-06:59



Exploratory Data Analysis

Visualizing and comparing distributions between original and generated datasets

Distribution Comparisons

Original (1 Day) → Generated (1 Week)

Verification Points

Controlled Outliers

tempo_max and tempo_min used for clipping
prevents unrealistic values

Coherent Distributions

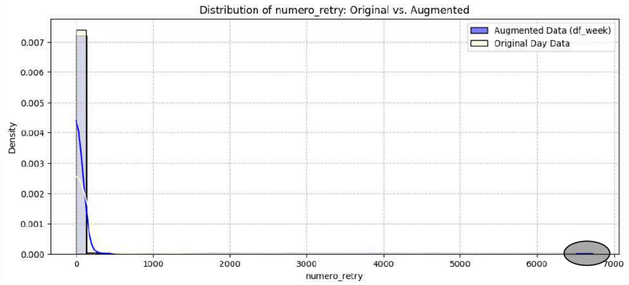
Generated data preserves key statistical properties:

- Similar means and medians
- Comparable variance
- Maintained correlation structure

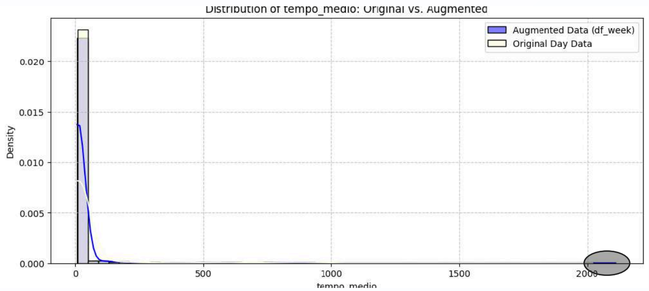
Realistic Noise

- Non-destructive variation
- Natural pattern fluctuations
- Balanced signal/noise ratio

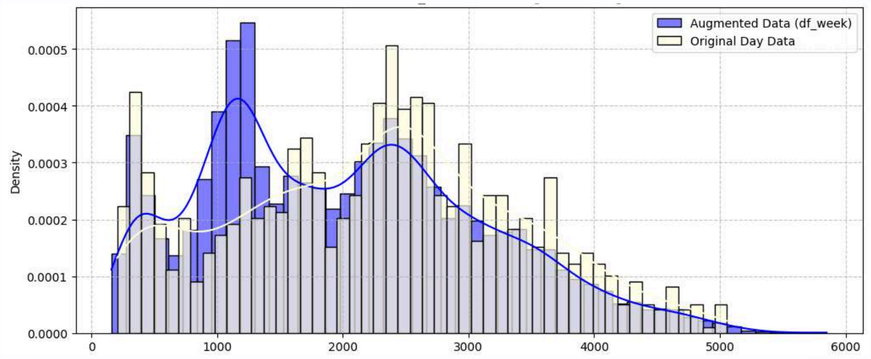
Retry Count



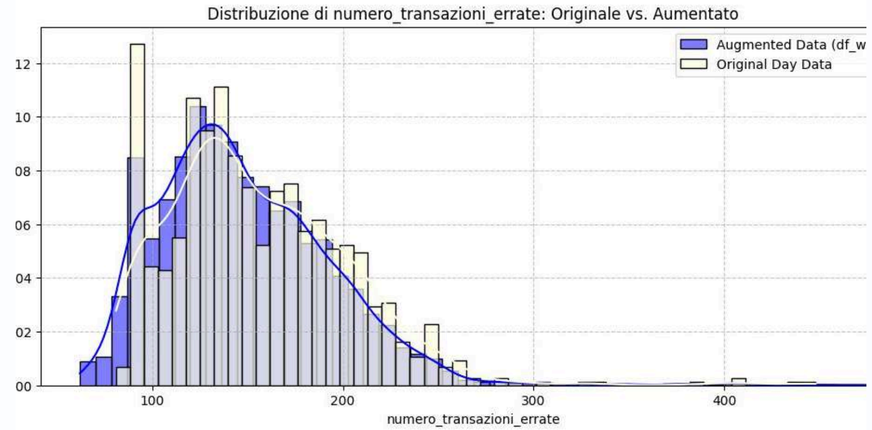
Average Response Time



Number of Transaction



Number Failed Transactions



Final Dataset Preparation

Transforming the engineered dataset into a format ready for modeling

Full Dataset → Feature Selection → Chronological Split → Scaled Features → Sequence Creation → Train/Val/Test Sets → Preserve Metadata → Final Analysis-Ready Data

Feature Selection

Removed Features

~~tempo_min~~ ~~tempo_max~~

These features were redundant with other engineered metrics and contained outliers that could negatively impact model performance.

Retained Features

numero_transazioni | tempo_medio | numero_retry | numero_transazioni_errate | hour_of_day | day_of_week | is_anomaly

- By extracting from data_ora features such as day_of_week and hour_of_day, we empowered our models to learn richer temporal patterns and relationships.
- Data_ora used as index
- In modeling phase only is_anomaly==0 considered

Feature Scaling

StandardScaler applied to numeric features:

- Zero mean, unit variance transformation
- Trained on training set only
- Applied to validation and test sets

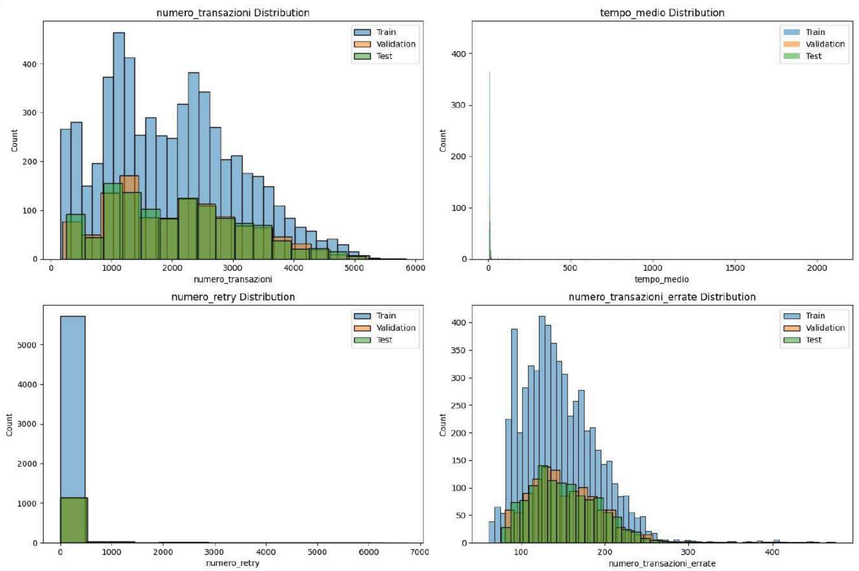
Data Splitting

Split Proportions

70% Train | 15% Val | 15% Test

Chronological split to maintain temporal integrity of data

Preserved Metadata



Final Dataset Results

- **~8200 usable rows (approximately 7 days of data)**
- Feature-rich dataset ready for time series modeling

Forecasting Models

Predictive approaches used to forecast transaction metrics

Model Selection Approach

Instead of Traditional:

- ARIMA
- Prophet

We Used:

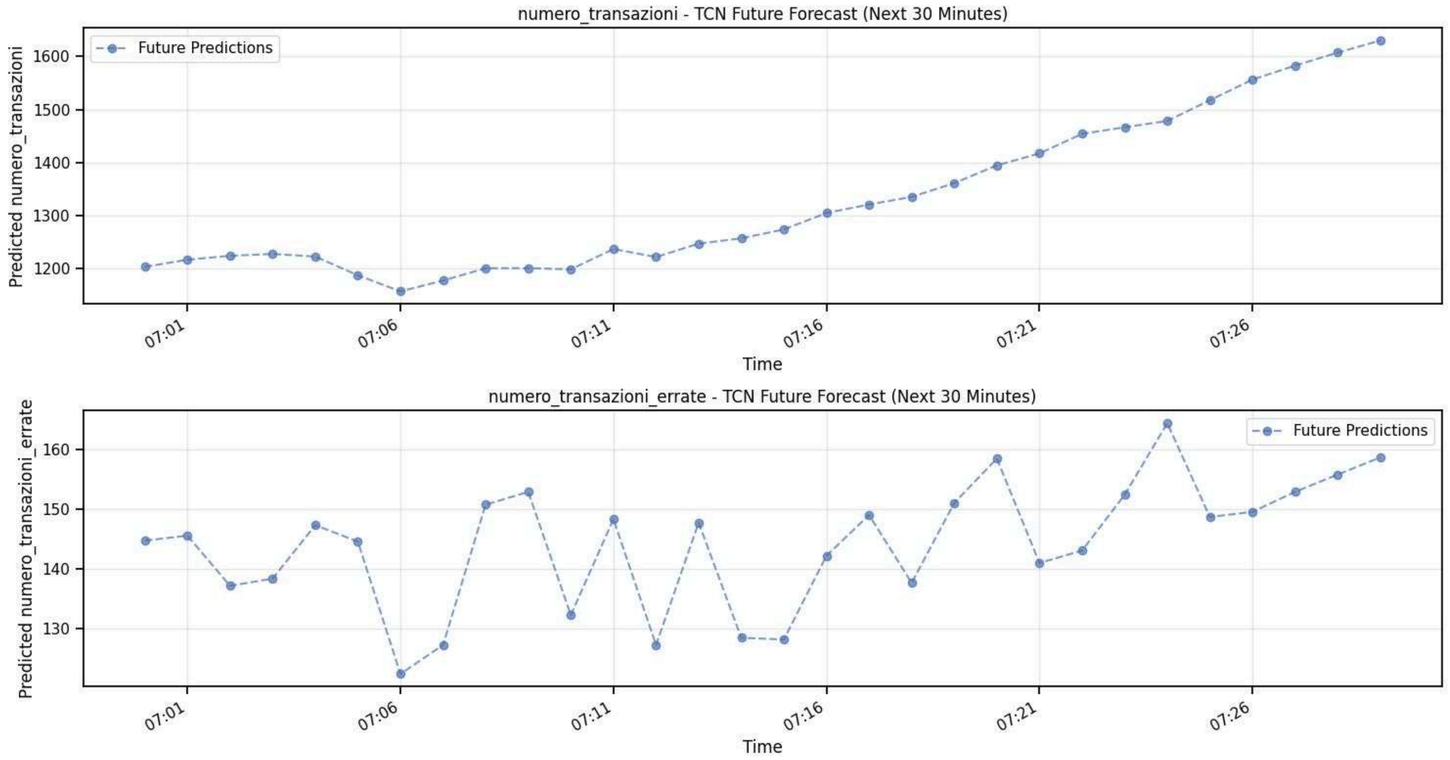
- TCN
- LSTM

Forecasting Goal Features:

- Numero_transazioni
- Numero_transazioni_errate

Key Results:

- Predicts **30 minutes ahead**
- Utilizes engineered temporal features



TCN Model Architecture

Input Sequence

Conv1D + BN

TCN Blocks Stack

Global Avg Pooling 1D

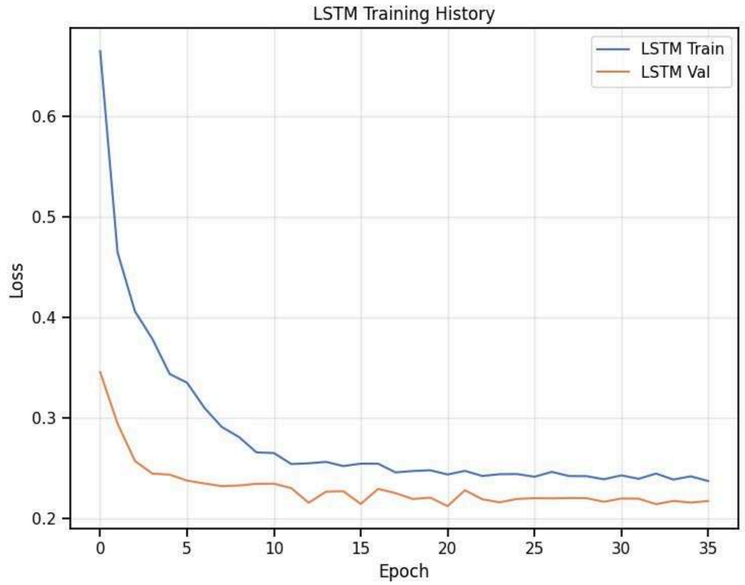
MLP Test (Classificator):
1. Dense(32,ReLU)+BN+Dropout
2. Dense(16,ReLU)+BN+Dropout

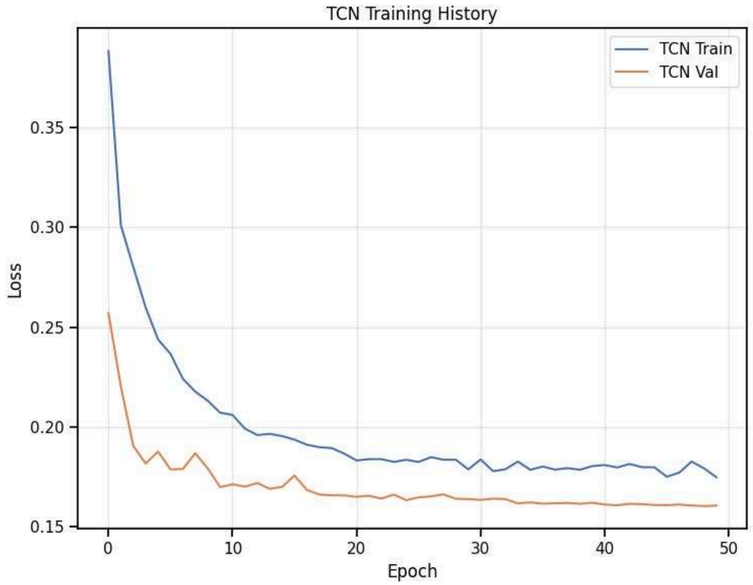
Output (Dense, 2)

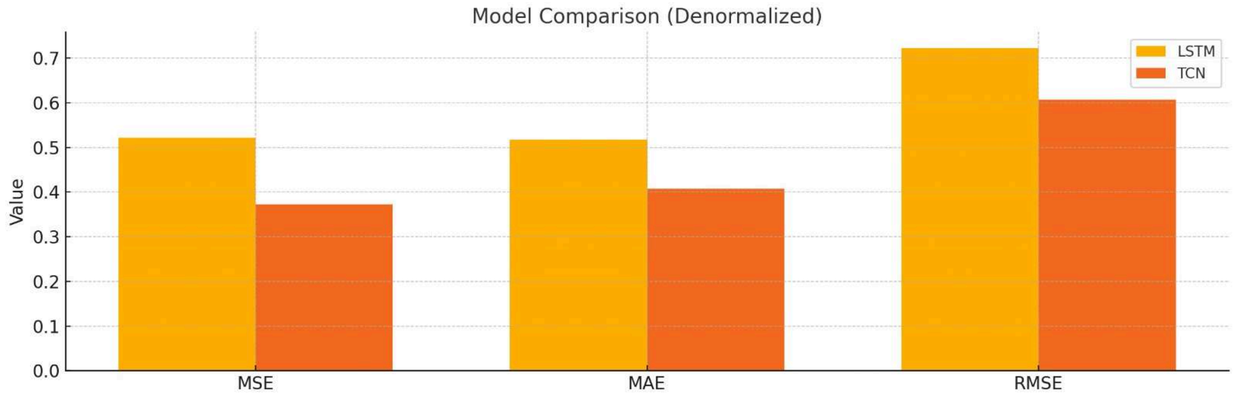
Forecast Evaluation

Assessing forecast accuracy and utility for anomaly detection

Evaluation Metrics



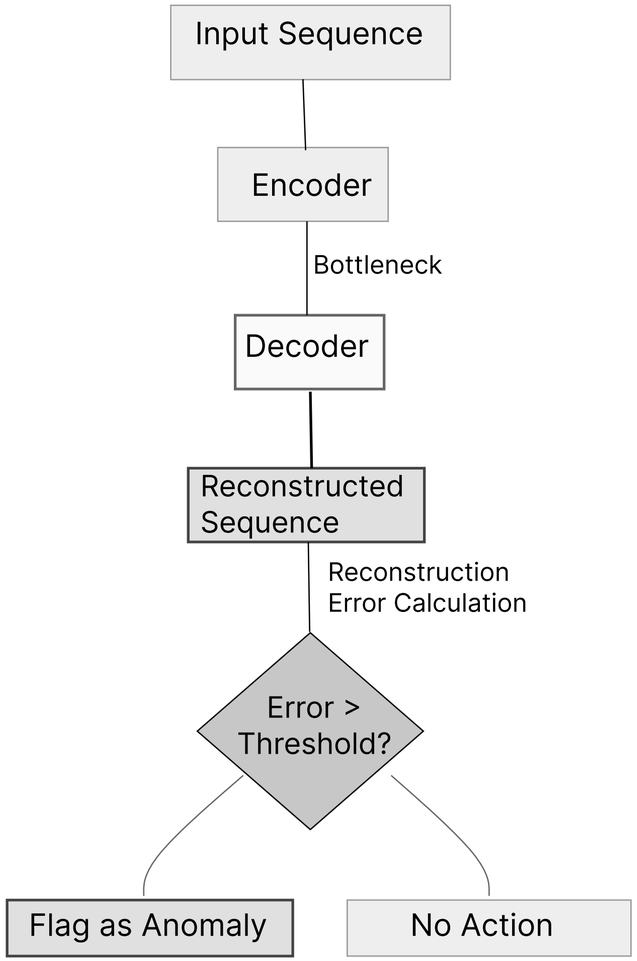




Anomaly Detection

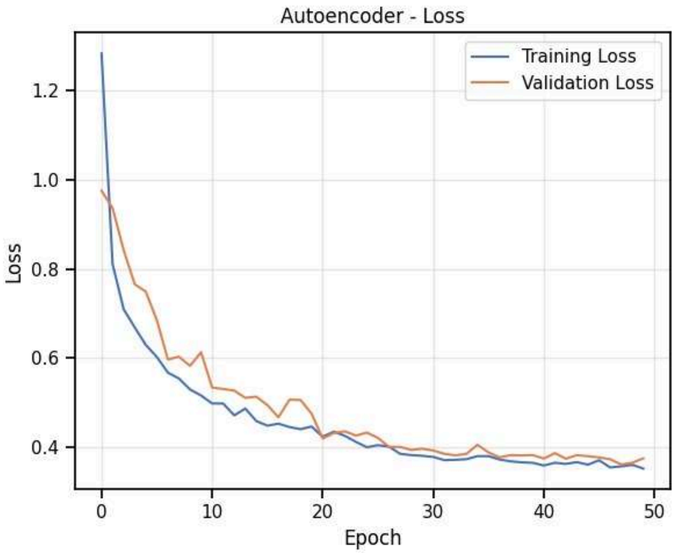
Using model predictions to identify and flag abnormal system behavior

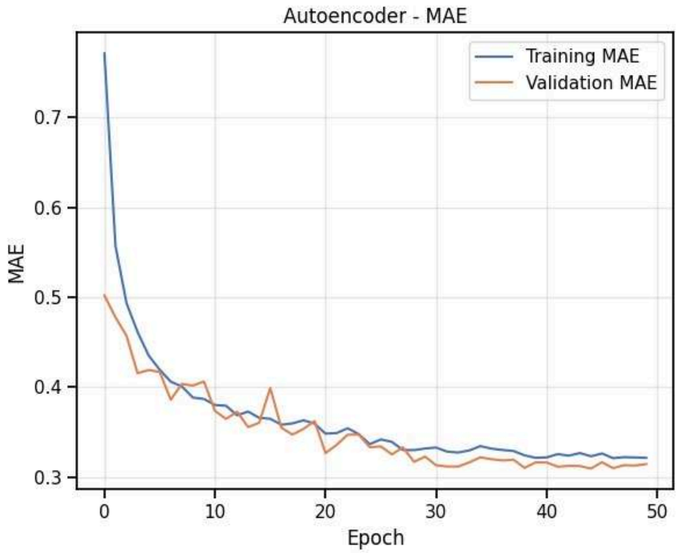
Anomaly Detection Approach



Autoencoder LSTM that detects anomalies based on input reconstruction error

- Learns normal patterns
- High reconstruction error signals anomaly
- Trained only on normal data

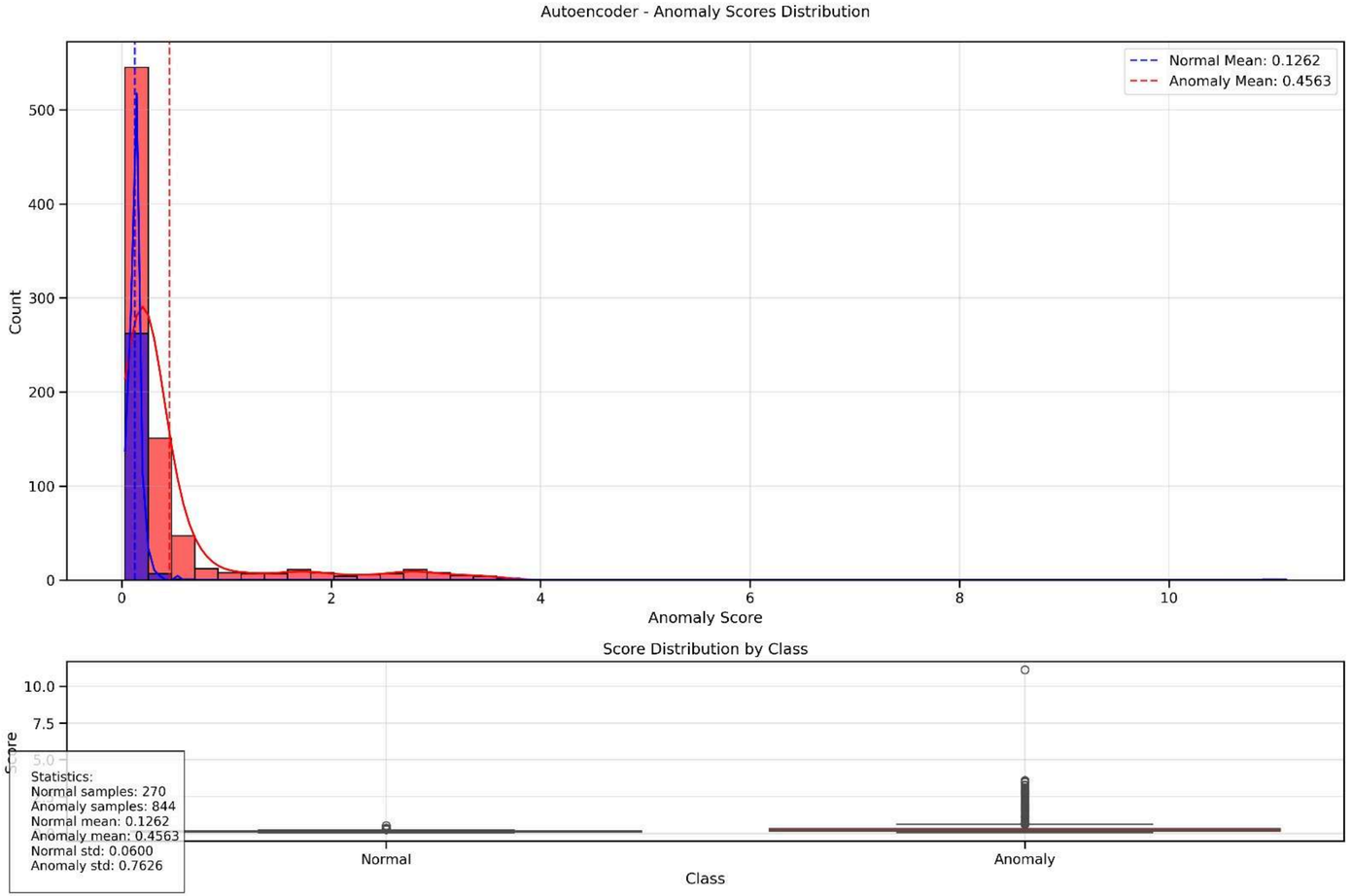




Anomaly Detection

Using model predictions to identify and flag abnormal system behavior

Anomaly Thresholds



Anomaly Detection

Using model predictions to identify and flag abnormal system behavior

Accuracy

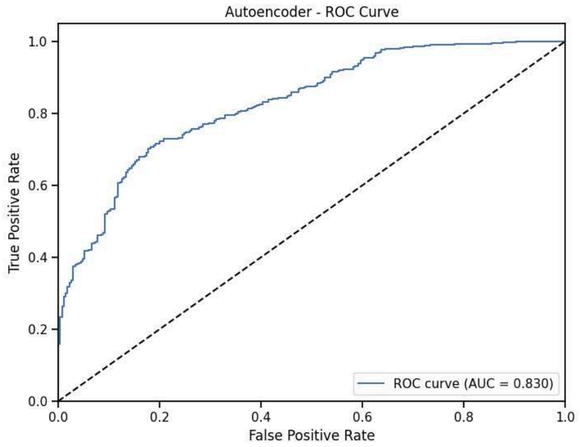
75% of actual anomalies correctly identified

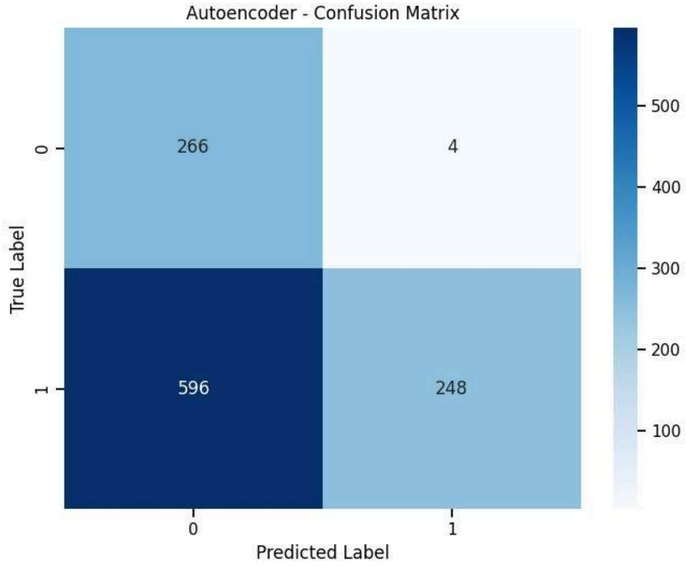
F1 Score

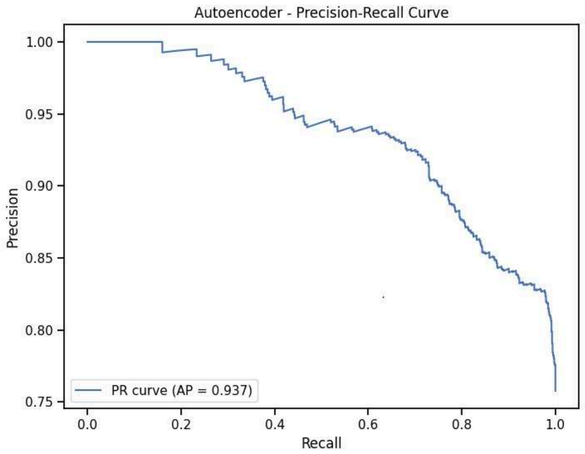
12% false alarms, primarily during transitional periods

Anomaly Detection Results

Transaction Spikes | Retry Spikes | Error Rate Anomalies







Limitations

Data Augmentation

- Relied solely on oversampling of known “anomalous” windows—no generative noise or SMOTE—so synthetic patterns may lack real-world variability.
- Augmented data still originates from a single-day snapshot, limiting coverage of seasonal or multi-day dynamics.

Feature Engineering

- Lack of external enrichments (weather, calendar events) means models miss potentially predictive context.

Forecasting Models

- Hyperparameters and architectures were fixed (no grid search or cross-validation), so models may not be optimally tuned.

Anomaly Detection

- 241 anomalies go unnoticed (FN = 241)
- Trained autoencoder threshold set by a single percentile cut—trade-off between false positives and negatives may not generalize.

Key Takeaways

Threshold tuning or cost-sensitive decision rule

Augment training data with more anomaly examples