Adversarial Robustness and Anomaly Detection for Financial ML Models

Introduction

Financial machine learning (ML) systems require robust defenses against adversarial perturbations and effective mechanisms to detect out-of-distribution (OOD) anomalies. These vulnerabilities pose serious risks, including fraud, abnormal trading signals, and market regime shifts. Ensuring both adversarial robustness and anomaly detection is critical for system integrity and regulatory compliance.

OOD Detection Algorithm

Our approach combines linear and nonlinear anomaly detection:

Mahalanobis Distance

We define Mahalanobis distance as:

$$D_M(x) = \sqrt{(x-\mu)^\top \Sigma^{-1}(x-\mu)},$$

where μ and Σ represent the mean vector and covariance matrix estimated from normal (in-distribution) data. Large $D_M(x)$ values indicate potential anomalies.

Autoencoder Reconstruction Error

An autoencoder compresses and reconstructs input data. The reconstruction error,

$$E(x) = \|x - \hat{x}\|^2,$$

remains small for in-distribution samples and increases for anomalies.

Ensemble Score

To capture both linear and nonlinear deviations, we combine normalized Mahalanobis distance and autoencoder reconstruction error:

$$S(x) = \frac{D_M(x)}{\max D_M} + \frac{E(x)}{\max E}.$$

A high ensemble score signals an outlier.

Extreme Value Theory Thresholding

Rather than setting arbitrary thresholds, we employ Extreme Value Theory (EVT) to model the score distribution tails. We fit a Generalized Pareto Distribution (GPD) to the score excesses, using the quantile function to determine robust thresholds, thus guaranteeing controlled false positive rates under theoretical assumptions.

Adversarial Robustness

To evaluate robustness, we use the Fast Gradient Sign Method (FGSM) to create adversarial examples:

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x L(x)),$$

where L is the autoencoder loss function. Our system demonstrates high resilience to such perturbations by effectively flagging adversarially perturbed data.

Statistical Confidence and Evaluation

We estimate the distribution of ROC AUC and PR AUC metrics using bootstrap resampling, providing 95% confidence intervals to ensure statistical rigor.

Python Implementation

Below is a concise code snippet highlighting core components:

```
cov = EmpiricalCovariance().fit(normal_data)
mahal_dist = cov.mahalanobis(all_data)

autoencoder.fit(normal_data, normal_data)
recon = autoencoder.predict(all_data)
recon_errors = np.mean((all_data - recon)**2, axis=1)

ensemble_score = (mahal_dist / np.max(mahal_dist)) + (recon_errors / np.max(recon_errors))

params = genpareto.fit(ensemble_score - np.min(ensemble_score))
threshold = np.min(ensemble_score) + genpareto.ppf(0.99, *params)

anomaly_flags = ensemble_score > threshold
```

For full reproducibility, refer to the repository code.

Conclusion

Our framework combines Mahalanobis distance, autoencoder reconstruction error, EVT-based thresholding, bootstrap confidence intervals, and adversarial tests to deliver a theoretically sound and empirically validated anomaly detection module. This approach significantly enhances the security and reliability of financial ML systems.