Final Project - Fintech

Detecting abnormal markets: Early Warning Systems

Team Meambers: **Sara Auletta, Riccardo Besana, Francesco Colombo, Giulia Di Vincenzo, Benedetta Gnugnoli**

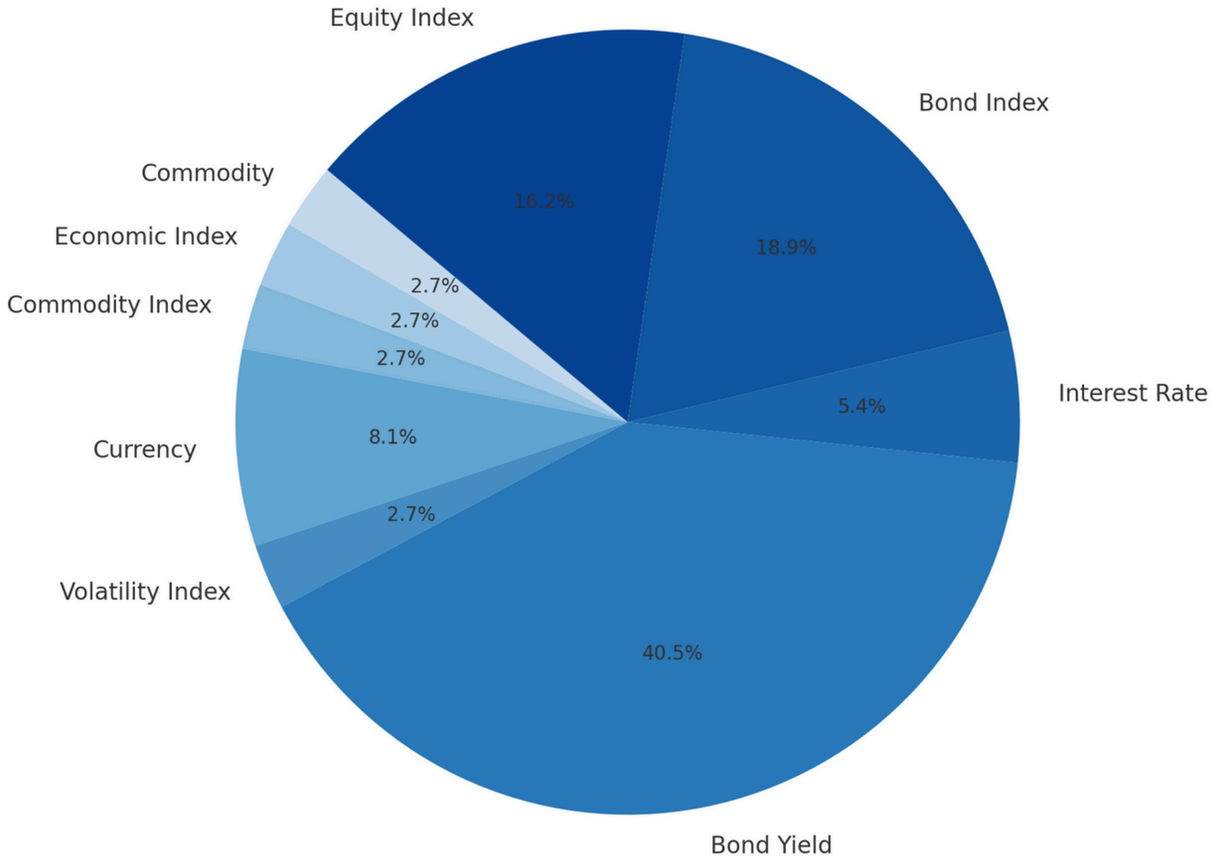
Dataset

The dataset consists of **43 financial indicators** and one **label Y**, indicating whether the observation corresponds to an anomaly (Y=1) or a normal market condition (Y=0)

Main Groups:

- **Bond Yield** (e.g. GT10, USGG2YR,GTDEM10Y)
- **Bond Index** (e.g. LUMSTRUU, LG30TRUU , EMUSTRUU)
- **Equity Index** (e.g. MXUS, MXWO, MXEU)

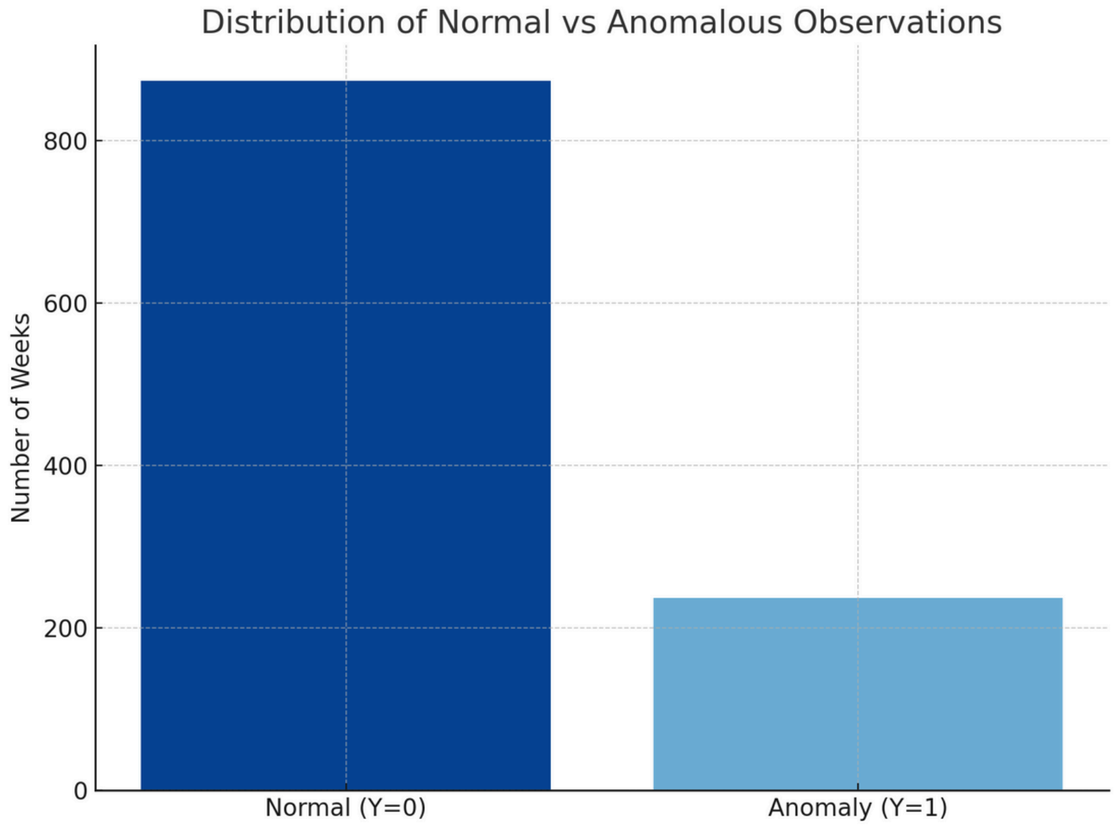
The MetaData also included Futures Contract but the datas were undefined



Dataset

Data imbalance

As expected, the number of anomalies in our dataset is significantly lower compared to normal observations. This imbalance is consistent with the **nature of financial markets**, where abnormal conditions occur less frequently compared to normal ones.



What we have done

DEVELOPED A DATA-DRIVEN EARLY WARNING SYSTEM

To detect abnormal patterns in global financial markets

DATA & PREPROCESSING

Starting from a dataset of market indicators, we conducted exploratory data analysis and ensured the stationarity of features.

CATEGORIZATION OF ANOMALIES

Our system identifies groups of financial assets showing anomalous behavior and categorizes them as either "Risk-on" or "Risk-off" based on statistical deviations.

MODELS

We implemented supervised, unsupervised, deep learning and ensemble models to test different approaches for anomaly detection.

Aim of this project

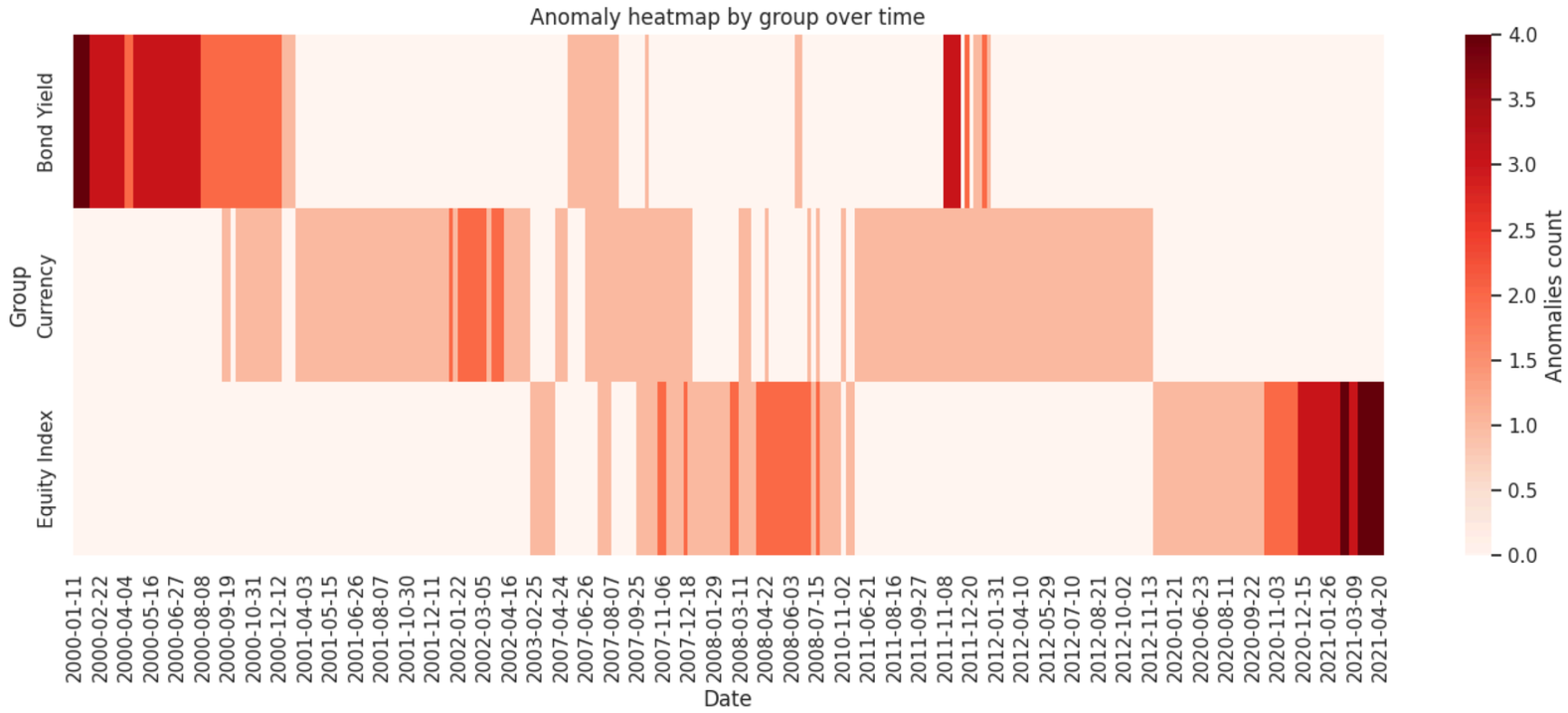
Sudden shifts in asset behavior can lead to significant losses for investors and institutions. Detecting the early signs of instability in the markets is crucial: it enables **risk mitigation**, **strengthening investment strategies** and overall **improved** financial **performance**.
Rather than attempting to forecast crises long in advance, the goal is to **recognize their early manifestations**: by identifying the initial signs of instability early, the system can **minimize financial damage**.

Data analysis

The anomaly detection was performed by measuring deviations from the mean and standard deviation, classifying observations as Risk-on (above average) or Risk-off (below average)

Groups with many anomalies were labeled as “Risky”.
To ensure fairness, we tested different anomaly thresholds, as group sizes vary. Still, the same three groups consistently appeared as the riskiest:

- Equity Index
- Bond Yield
- Currency



Data Preprocessing

To prepare the dataset for modeling, the features are first transformed to ensure stationarity, then the data is shuffled to remove temporal dependencies and split into three sets, finally the features are standardized

Coping with **stationarity**:

- so the statistical properties of our features remain stable over time
- since this prevents misleading results from shifting trends

Only the features are transformed—the target variable (Y) indicating anomalies remains unchanged.

Data splitting into three sets:

- **Training set**: 80% of normal data, used to train the model
- **Cross-validation set**: 10% of normal data + 50% of anomalies, used to tune thresholds
- **Test set**: 10% of normal data + 50% of anomalies, used for final evaluation

Standardization of the features:

- allows us to train on only normal data and evaluate on mixed sets to detect anomalies

MODELS

We compare a variety of anomaly detection approaches — ranging from classical statistical models to advanced deep learning techniques.

All models are evaluated on the same dataset using a consistent set of performance metrics: **Precision**, **Recall**, **F1 Score**, **Confusion Matrix**, and **ROC Curve.**

Baseline model: Multivariate Gaussian (MVG)

Unsupervised model with supervised threshold tuning

Steps:

- Estimate mean vector and covariance matrix from normal data
- Compute anomaly scores using the multivariate Gaussian PDF
- Tune threshold (ε) on validation set to maximize F1 score

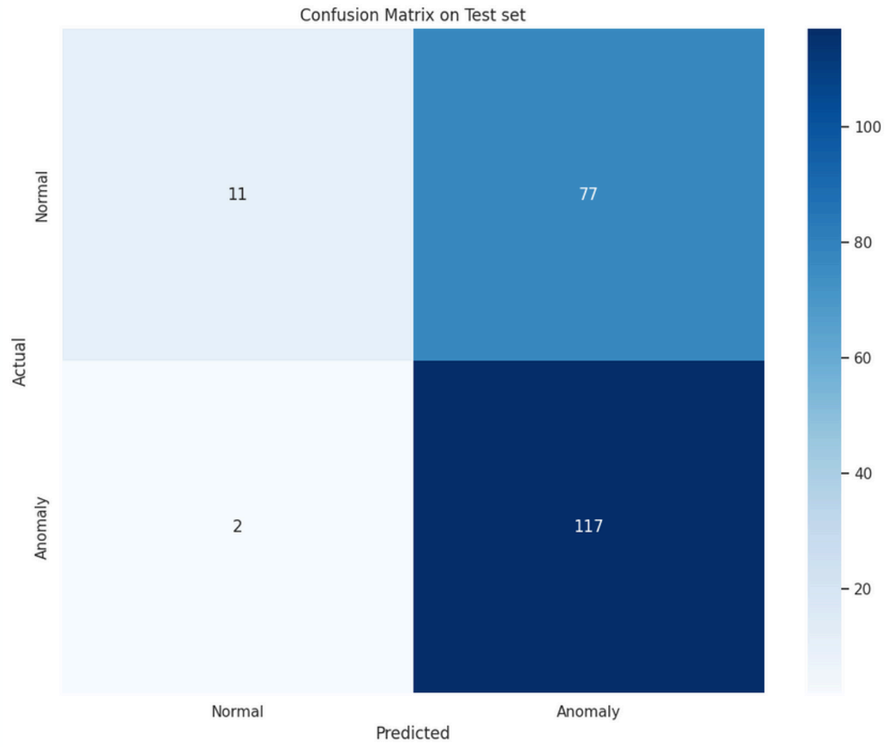
Test set performance:

- Precision: 0.6031
- Recall: 0.9832
- F1 Score: 0.7476

Good anomalies prediction but the precision score was low

Purpose:

- Serve as a **benchmark** to compare more complex models



Supervised

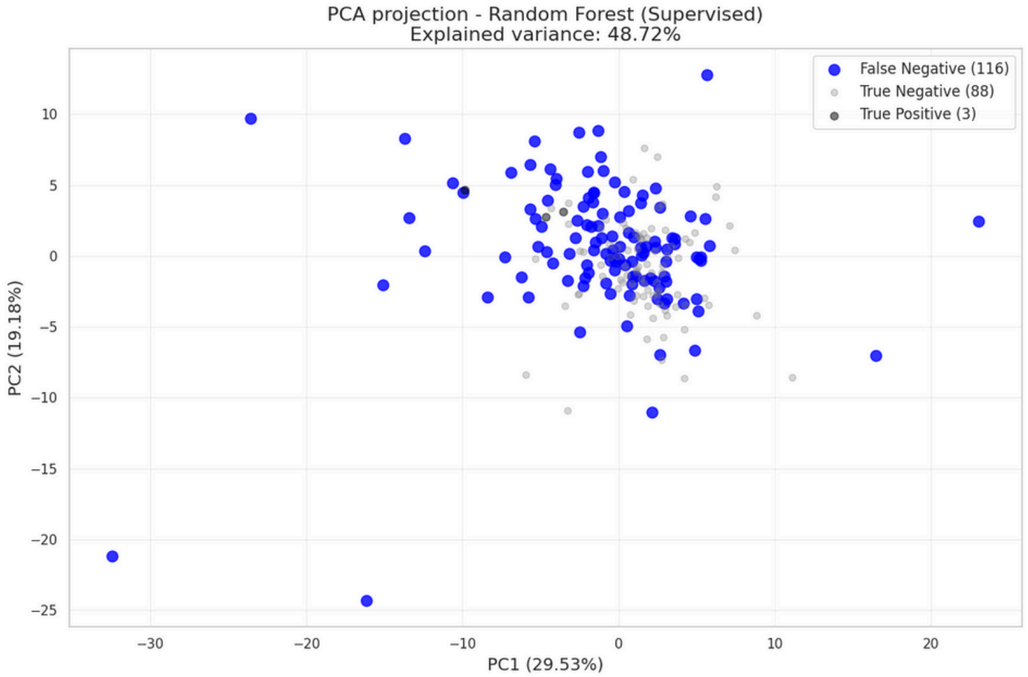
Random Forest

- relatively simple to implement
- easily interpretable

Results: the model learned very well how to predict normal data but failed in predicting anomalies. In fact only 3 out of 119 anomalies of our test set were predicted as such.

Test set performance:

- Precision: 1.0000
- Recall: 0.0252
- F1 Score: 0.0492



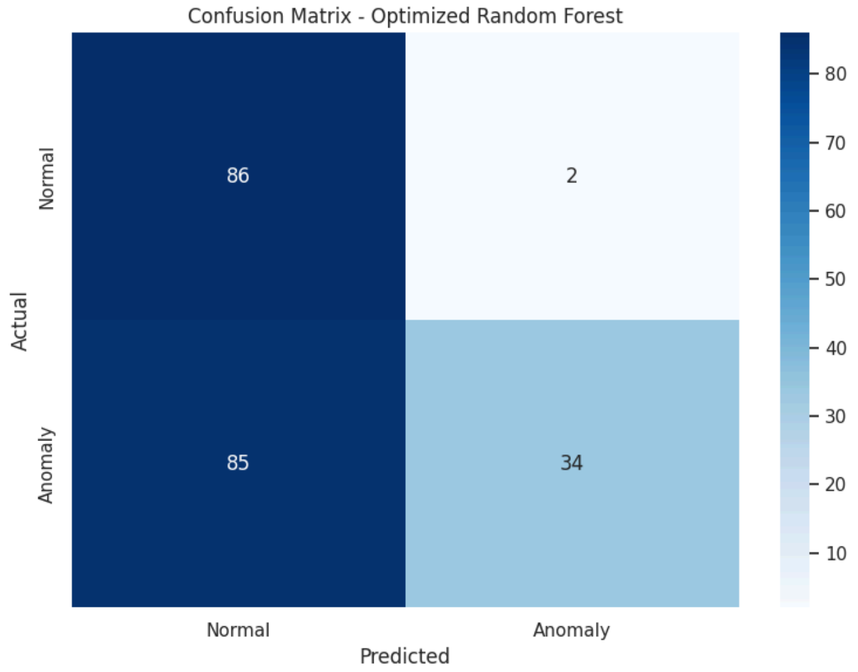
Optimize Random Forest

- since the initial performance of the Random Forest model was unsatisfactory, we applied hyperparameter optimization to try to improve its results

Results: we can see a high improvement as now 34 out of 119 anomalies were predicted as such, anyway still not a satisfying result

Test set performance:

- Precision: 0.9444
- Recall: 0.2857
- F1 Score: 0.4387



XGBoost

- it handles overfitting through regularization

Results: Significant boost in performance across all metrics.

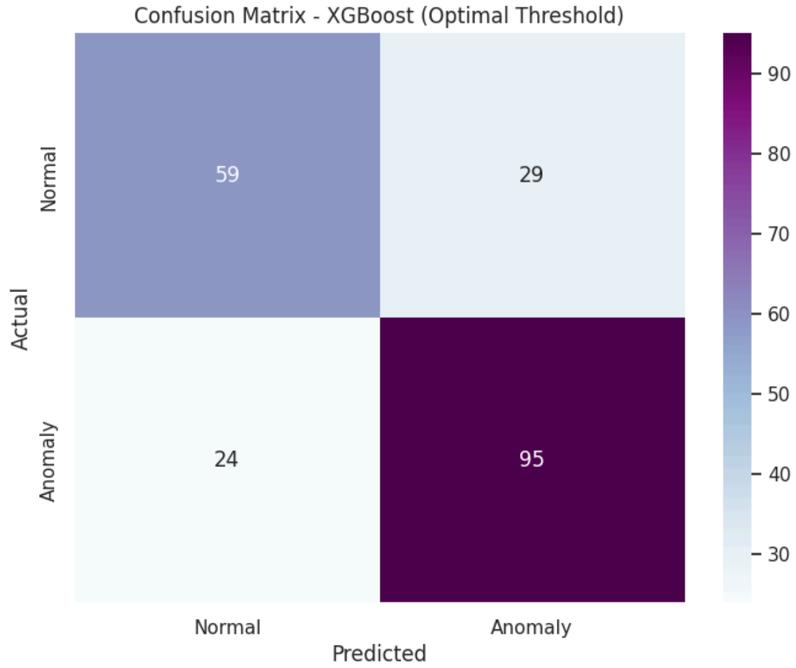
Thresholds tested:

- Fixed: 0.5
- Optimal (from ROC curve): 0.58

Similar performance with both thresholds, confirming model robustness.

Test set performance:

- Precision: 0.7661
- Recall: 0.7983
- F1 Score: 0.7819



Unsupervised

COPOD

It needs an estimated anomaly rate (contamination), which we calculate from the validation set only. To ensure stability, we cap it at 50%.

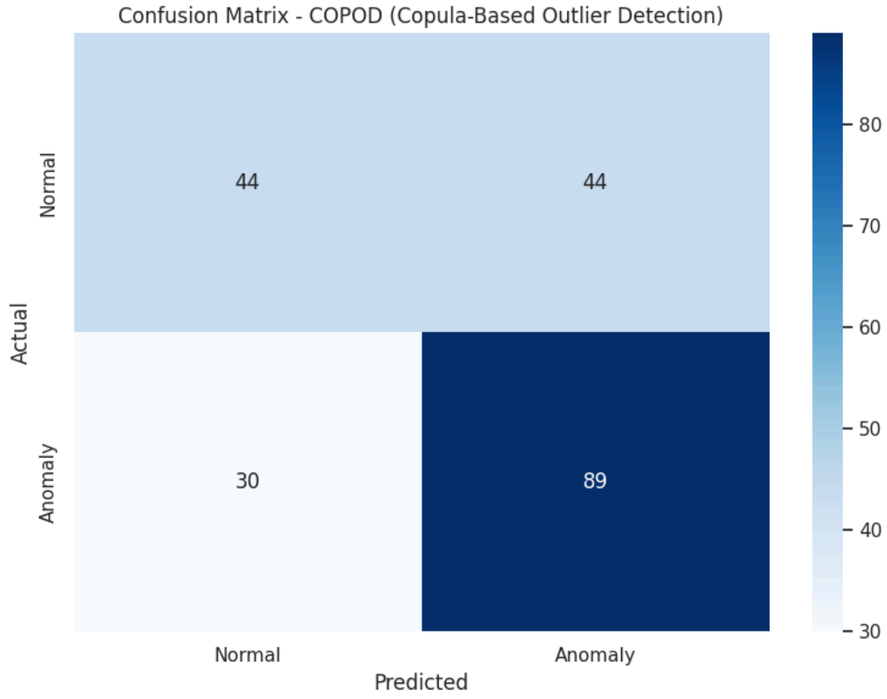
Performance: Underperforms vs. XGBoost and MVG baseline

Limitations:

- No access to labeled anomalies
- Relies on feature-wise distributions only
- Struggles with nonlinear structure of financial time series
- Poor distinction between true anomalies and harmless outliers

Test set performance:

- Precision: 0.6692
- Recall: 0.7479
- F1 Score: 0.7063



Deep Learning

LSTM

The model is trained to classify each day as either normal or anomalous, based on the recent behavior of the market. Specifically, it looks at a short time window of past days and learns to predict whether the current day is anomalous, given the pattern it has just seen **(supervised learning setting).**

Dataset Preprocessing

When splitting the dataset into training, validation, and test sets, we paid close attention to maintaining the temporal consistency.

Since the dataset is highly imbalanced, we compute and apply class weights during training to give more importance to the minority class (anomalies).

Window Size

We use a small window size (3) when constructing input sequences for the LSTM since in financial markets very recent history has the strongest influence on the present.

Looking too far back may introduce noise or correlations that are no longer meaningful due to market dynamics changing rapidly.

LSTM

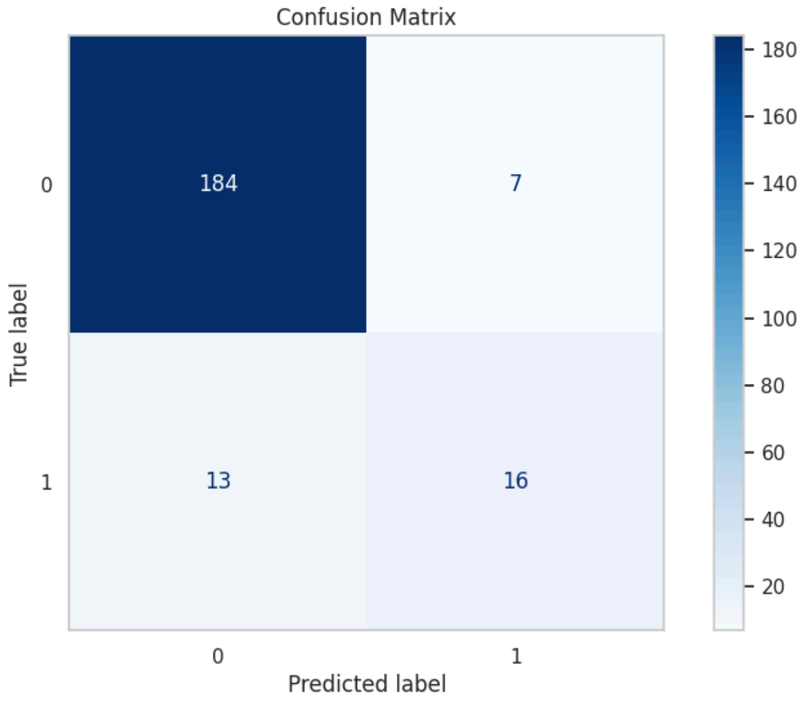
The loss curves show overfitting: it highlights the importance of using EarlyStopping and ReduceLROnPlateau to control it.

Results with threshold 0.54:

- Precision: 0.696
- Recall: 0.552
- F1-score: 0.615

The model correctly identifies 16 out of 29 anomalies while keeping false positives relatively low (7).

Despite limited data, the result is promising—showing the LSTM successfully captured meaningful temporal patterns.



LSTM Autoencoder

The model is trained in an **unsupervised way** on normal data, then iteratively improves by selecting high-error sequences as potential anomalies. It incorporates a semi-supervised component through active learning, where high-error samples are assumed to be anomalies and included in subsequent training rounds to refine the model.

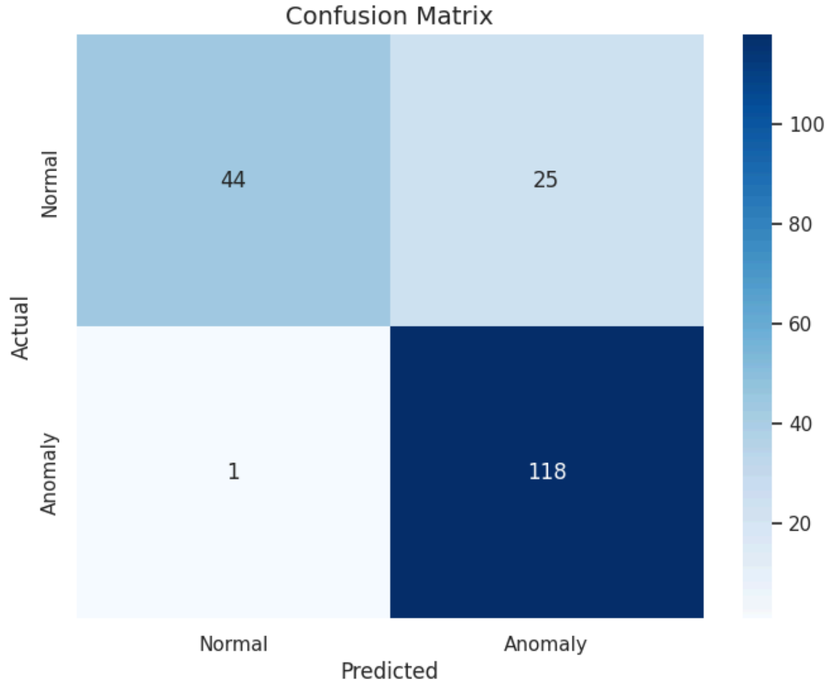
LSTM Autoencoder with dropout, batch norm, and latent dim = 8, with stacked LSTM layers (hidden size = 64).

Trained unsupervised on 100 normal sequences using MSELoss and Adam. Learning rate adjusted via ReduceLROnPlateau.

Active learning adds high-error samples as pseudo-anomalies.

Results:

- Precision: 0.8252
- Recall: 0.9916
- **F1 Score: 0.9008**



Ensemble

AE + XGBoost + LOF

Based on previous results, we combined supervised and unsupervised models to test if they could complement each other effectively.

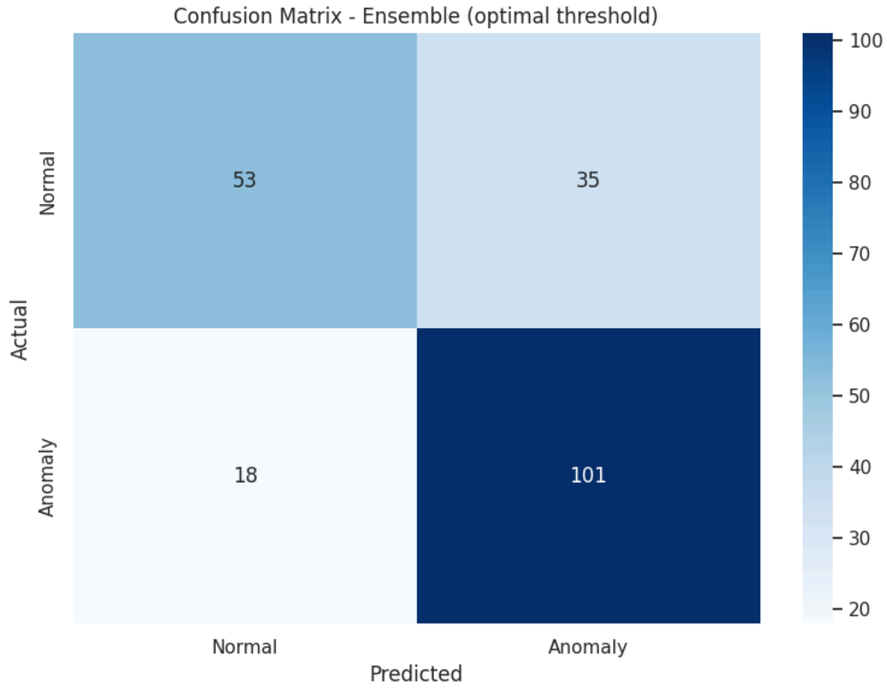
Thresholds tested: 0.5 (fixed) and 0.46 (optimal from ROC)

Results with threshold 0.46:

- Precision: 0.7426
- Recall: 0.8487
- F1 Score: 0.7922

Each model captures anomalies from a different perspective — reconstruction error, classification probability, and local density.

To ensure consistency, all scores are normalized to a [0, 1] range and then averaged to produce a final anomaly score.



Final Comments

We can conclude that we achieved the best results with the **LSTM Autoencoder combined with Active Learning**.
We are satisfied with our results, as this model was able to detect 118 out of 119 anomalies in the test set.

It is reasonable that the best-performing model is an LSTM, given that the **temporal aspect plays a crucial role in this analysis**, and it is precisely what LSTMs are designed to capture.

However, a major limitation lies in the fact that LSTMs are **black-box models**, which reduces interpretability, which is an essential aspect in finance, thus preventing a clear understanding of the causes behind the anomaly predictions.

Final Comments

LIMITATIONS:

- The small sample size made it difficult to build a robust model capable of generalizing well, especially for neural network models, which typically require large amounts of data.
- The dataset was imbalanced, with very few anomalies, making it challenging for the model to learn a wide range of anomalous patterns.

IMPROVEMENTS:

- An extension of the ensemble approach through bootstrapping on a larger dataset is expected to improve performance and enhance the model's robustness.

Taking these considerations into account, along with our resource limitations, and thanks to our continuous collaboration and teamwork, we are satisfied with the result achieved.