Systematic Trading Strategies with Machine Learning Algorithms

Introduction to Systematic Strategies with ML

April 10, 2025

Outline

Imperial College Business School

Course Structure, Evaluation, and Logistics

Fundamentals of Machine Learning for Systematic Strategies

- Labeling Methods
- Evaluation Metrics
- Hyperparameter Tuning

The Meta Labeling approach

Programming Session 1: Trend Scanning Implementation

Course Syllabus Overview

Imperial College Business School

Total: 9 Lectures

- Mix of theory and practice.
- 3 full programming sessions: Lectures 3, 5, and 8.
- Weekly office hours: **Fridays, 4pm–5pm**.
- Optional lectures during office hours (Weeks 6, 8, 9).

Syllabus Breakdown:

Imperial College Business School

Lecture	Theory	Programming Session
Lecture 1: *Introduction to Systematic Strategies with Machine Learning*	- Introducing systematic strategies using machine learning, covering labeling methods, evaluation metrics, feature importance analysis, and hyperparameter optimization tailored for sequential data. - Introducing the Meta-labeling approach, which strategically balances recall and precision by identifying conditions under which a primary model yields optimal trading performance.	- Practical session on financial data labeling for trend detection.
Lecture 2: *Unsupervised Learning and Clustering Techniques*	- Introducing clustering algorithms and dimensionality reduction techniques, highlighting their role in addressing substitution effects in feature importance analysis.	- Improving machine learning model performance by reducing substitution effects among correlated features using cluster-level feature importance analysis.
Lecture 3: *Database Management for Financial Data*	—	- This programming-focused lecture covers database management fundamentals. - The data will be used for subsequent parts of the course.
Lecture 4: *Supervised Learning Algorithms*	- Overview of key supervised learning algorithms with emphasis on ensemble models (Bagging and Boosting) and feature importance analysis. - Introducing Gated Residual Networks (GRN) and Variable Selection Networks (VSN).	- Forecasting time series with gradient boosting.
Lecture 5: *Enhancing Strategy Performance in Crypto Markets*	—	- Applying concepts from previous sessions to filter out non-profitable trades from a primary model proposed on crypto data, using a tree-based secondary model with an emphasis on cluster-level feature importance analysis.

Syllabus Breakdown:

Imperial College Business School

Lecture	Theory	Programming Session
Lecture 6: *Latent Variable Models in Financial Asset Regime Detection*	- Introducing Hidden Markov Models (HMMs) for predicting financial market regimes.	- Implementation of the HMM in a simple case of discrete observations.
Lecture 7: *Neural Networks for Interpretable Time Series Forecasting*	- Exploration of neural network models for interpretable time series forecasting, including GRUs, LSTMs, N-Beats and Transformers.	- Coding a Transformer from scratch.
Lecture 8: *Volatility Forecasting with Temporal Fusion Transformers*	—	- In-depth exploration of sequential neural network architectures for predicting volatility, focusing on the replication of the Temporal Fusion Transformer study, and evaluating its performance against the models discussed in earlier sessions.
Lecture 9: *Review*	- Mock exam and revision elements to prepare for the final exam.	—

Optional Lectures

Scheduled during Friday office hours (4pm–5pm). These lectures explore advanced topics not required for the final evaluation, but highly relevant for students interested in cutting-edge research or industry applications.

- **Week 6:** Introducing Deep Generative Models
- **Week 8:** Graph Representation Learning
- **Week 9:** Generative AI: Finetuning an LLM to Create an AI Trading Assistant

Imperial College Business School

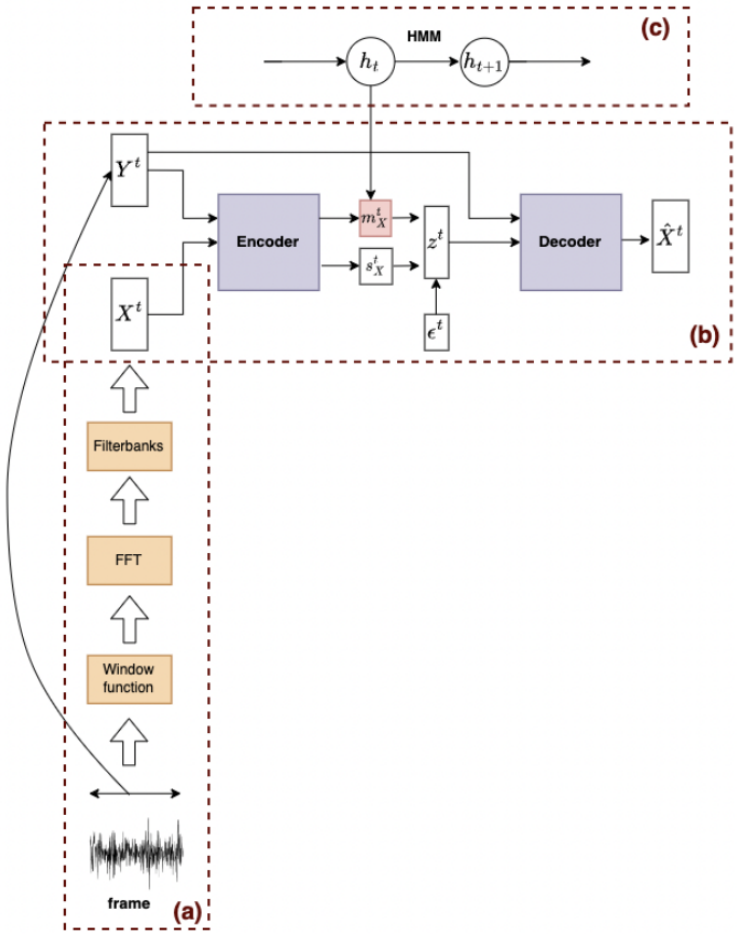
The Low Turbulence Model [4]

→ **Processing the time series:**

- Step (a): Time Frequency Analysis
  - Extracting the content of the data in the time-frequency domain.
  - Getting the spectral vectors.
- Step (b): Dimensionality Reduction
  - Getting the latent spectral vectors.

→ **Detection of Regimes**

- Step (c): HMM



Imperial College Business School

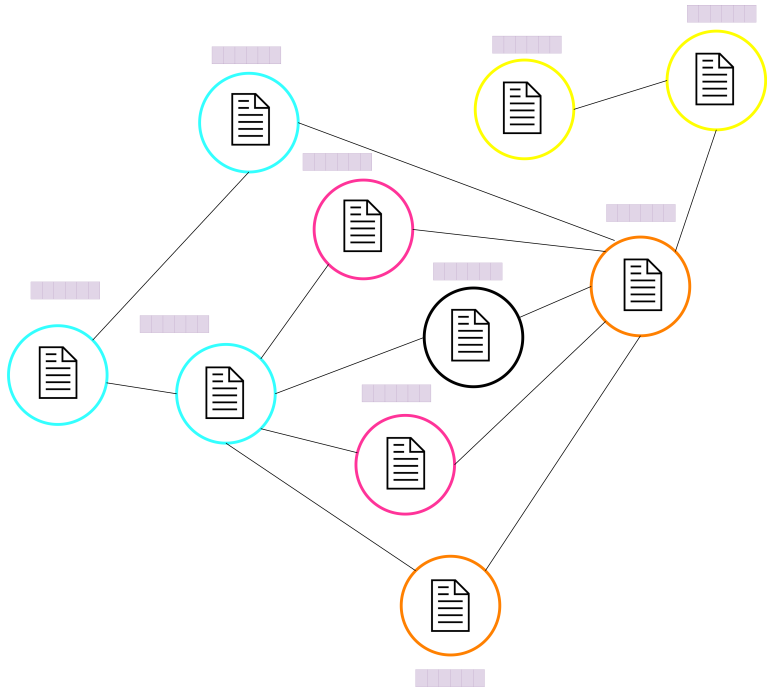
Graph Representation Learning
We cover two key approaches:

1. Graph Structure-Based Embeddings:

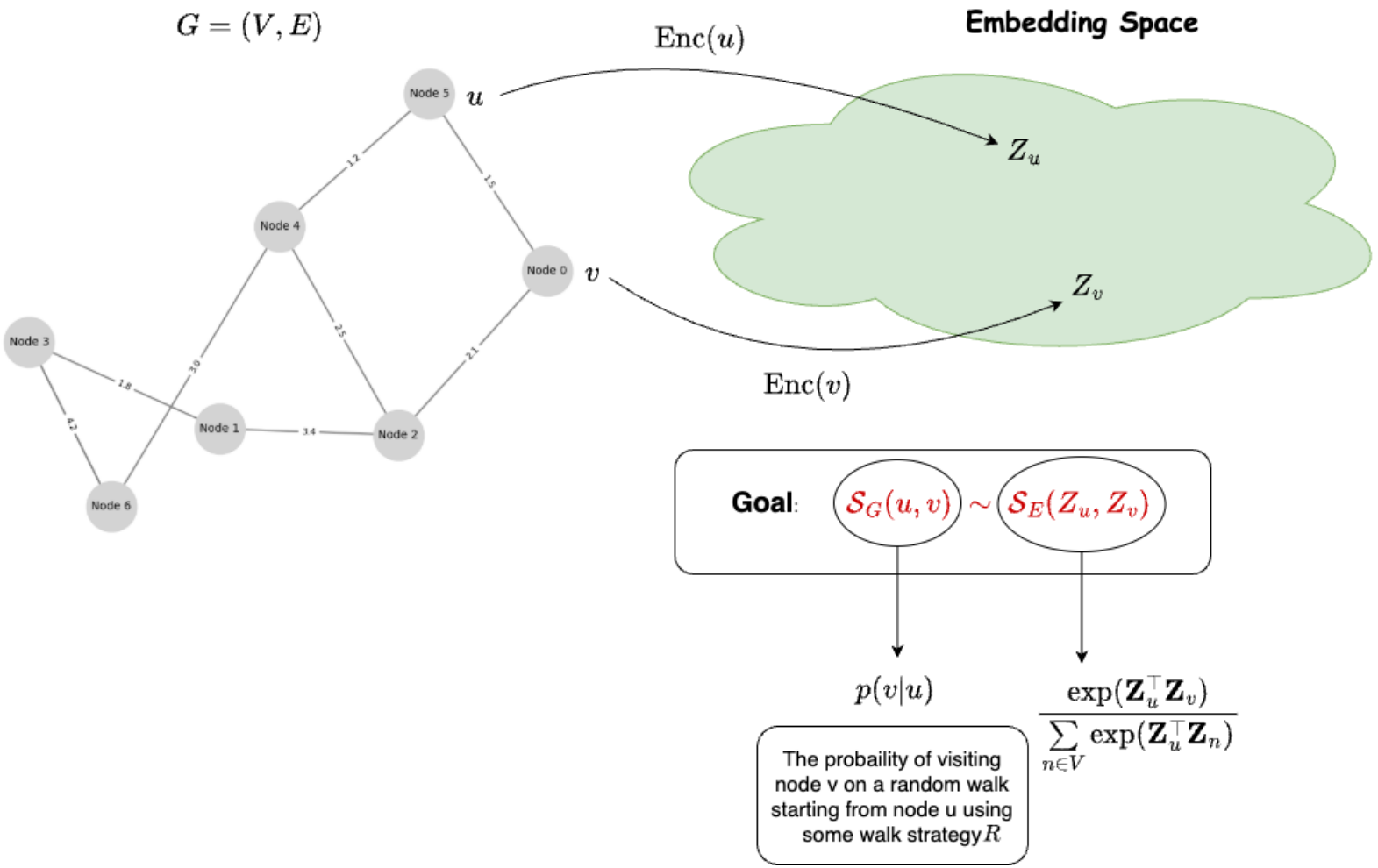
- Use biased random walks to learn embeddings that preserve local/global structure.

2. Message-Passing Graph Neural Networks:

- Iteratively aggregate and update node features using neighbors' embeddings.



Graph Structure-Based Embeddings:



Algorithm Message Passing Framework

Require: Graph $G = (V, E)$, node features $\{\mathbf{x}_v \mid v \in V\}$, number of iterations K, $f_{\text{aggregate}}$, f_{update}

Ensure: Final node embeddings $\{\mathbf{h}_v^{(K)} \mid v \in V\}$

1: Initialize embeddings: $\mathbf{h}_v^{(0)} \leftarrow \mathbf{x}_v$ for all $v \in V$

2: **for** $k = 1$ to K **do**

3: **for** each node $v \in V$ **do**

$$\mathbf{a}_v^{(k)} \leftarrow f_{\text{aggregate}}\left(\{\mathbf{h}_u^{(k-1)} \mid u \in \mathcal{N}(v)\}\right)$$

$$\mathbf{h}_v^{(k)} \leftarrow f_{\text{update}}(\mathbf{a}_v^{(k)}, \mathbf{h}_v^{(k-1)})$$

4: **end for**

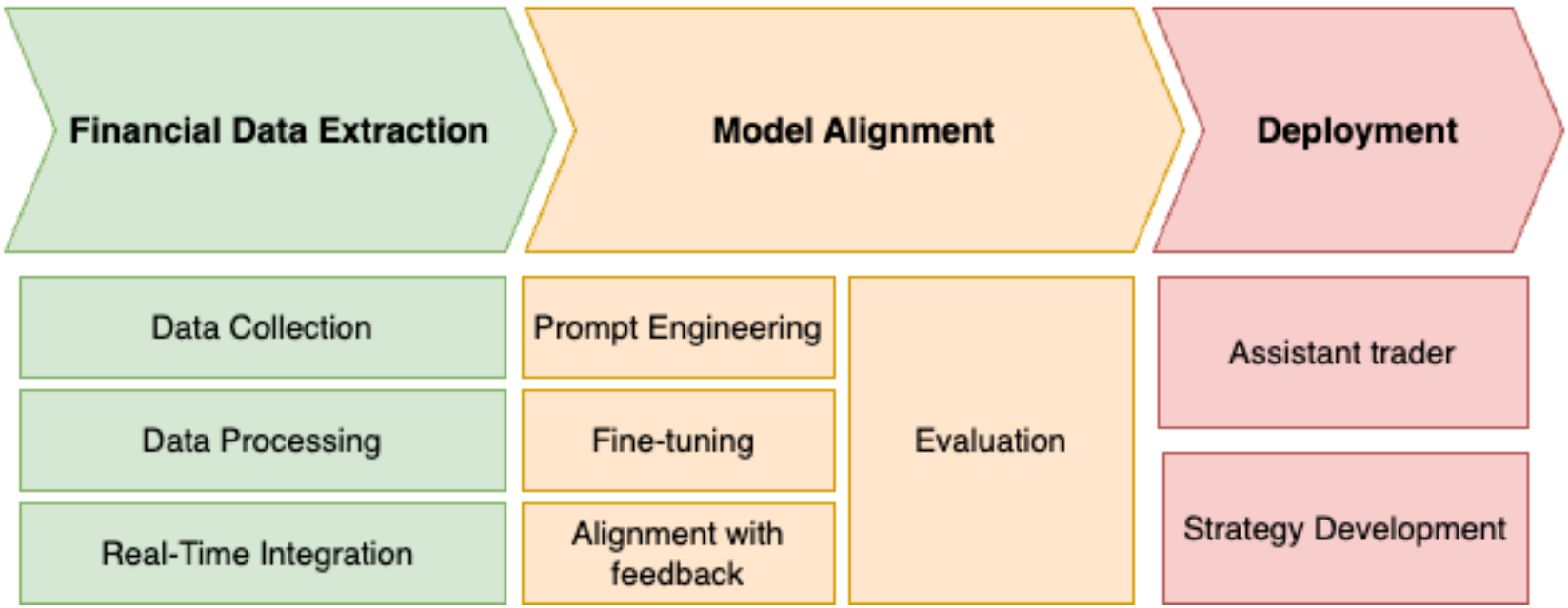
5: **end for**

6: **return** $\{\mathbf{h}_v^{(K)} \mid v \in V\}$

Imperial College Business School

Finetuning an LLM to Build an Assistant trader

We will walk through the process of using Generative AI to build an assistant trader.



Imperial College Business School

Guest Speaker – Prof. Salem Lahlou

- Co-author of all major GFlowNet papers with Prof. Yoshua Bengio. (See [1], [3], [5])
- Developer of the open-source library `torchgfn`.
- Will present how GFlowNets can be used to improve decision making in finance.

Prof. Salem Lahlou

On GFlowNets – Yoshua Bengio (Turing Award, 2018):

"I have rarely been as enthusiastic about a new research direction. We call them GFlowNets, for Generative Flow Networks. They live somewhere at the intersection of reinforcement learning, deep generative models and energy-based probabilistic modelling."

Imperial College Business School

Your final grade will be based on:

- **50% Final Exam**
  Standard written evaluation based on the course content.
- **50% Project with Brevan Howard (JJ's Team)**
  You will work in teams on a machine learning challenge applied to real-world financial data.

Jean-Jacques – Portfolio Manager at Brevan Howard

MCQs and Office Hours

- After each lecture, you will find a short set of MCQs on the course platform.
- These are **optional**, not graded, and meant to test your understanding of key concepts.
- They can also serve as a self-check before tackling the project or exam.

Need help?
Feel free to send me an email anytime. I'll try to respond as quickly as possible.

I highly encourage you to attend office hours every Friday from 4pm to 5pm.

Getting to Know You Poll!

Click here to participate in the poll

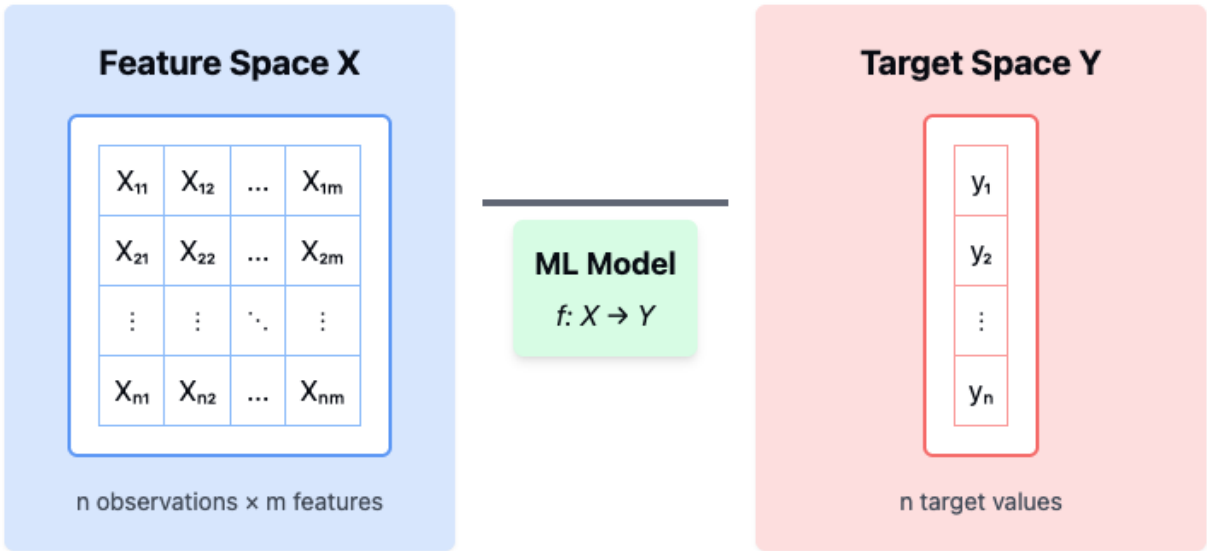
Outline

Outline

Imperial College Business School

Supervised Learning:

- Learn a function $f : X \to Y$ from labeled data.
- **Feature space** X: matrix of features, n observations $\times$ m features.
- **Target space** Y: vector of n labels (or target values).

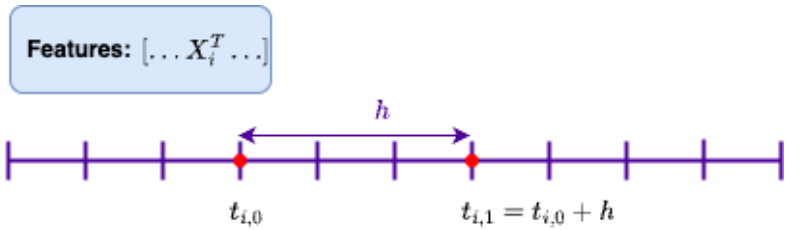


Regression vs Classification

Imperial College Business School

Supervised Learning Tasks:

- **Regression:**
  - Predict a continuous outcome.
  - $y_i \in \mathbb{R}$, e.g., temperature, price, return value.
- **Classification:**
  - Predict a discrete label from a finite set.
  - $y_i \in \{c_1, c_2, \ldots, c_K\}$, e.g., up/down, buy/sell/hold.



Regression Task: $r_{t_{i,0},t_{i,0}+h} = \frac{p_{t_{i,0}+h}}{p_{t_{i,0}}} - 1$

Classification Task: $r_{t_{i,0},t_{i,0}+h} \to y_i \in \{1, -1, 0\}$

Labeling: Why It Matters

Imperial College Business School

Labeling defines the task that the ML algorithm learns — it encodes the investment hypothesis.

- The choice of label determines what features are informative.
- Features that are irrelevant for one labeling scheme may be predictive under another.
- Understanding feature relevance requires proper **Feature Importance Analysis** (see Lecture 2).

Labeling Methods covered in the course:

- **Fixed-Time Horizon Labeling** — return over a fixed h period.
- **Triple Barrier Labeling** — incorporates profit-taking, stop-loss, and timeouts.
- **Trend Scanning** — detects regime changes (Programming Session 1).

Fixed-Time Horizon Labeling

Most common method in financial ML papers.

$$y_i = \begin{cases} -1 & \text{if } r_{t_{i,0},t_{i,0}+h} < -\tau \\ 0 & \text{if } |r_{t_{i,0},t_{i,0}+h}| \leq \tau \\ 1 & \text{if } r_{t_{i,0},t_{i,0}+h} > \tau \end{cases} \quad \text{with} \quad r_{t_{i,0},t_{i,0}+h} = \frac{p_{t_{i,0}+h}}{p_{t_{i,0}}} - 1$$

- h: number time bars.
- τ: return threshold.

Problems:

- Same threshold τ used regardless of volatility.
- Ignores price path — only cares about return at $t + h$.

Fixed-Time Horizon: Pitfalls

Imperial College Business School

Why this method is often flawed:

- Labeling does not account for volatility: a 1% move in low-volatility vs. high-volatility regimes is treated the same.
- It ignores stop-losses and the path of returns — key to real trading decisions.

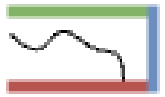
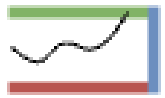
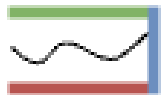
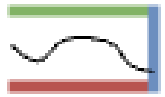
Alternatives:

- Use volatility-adjusted thresholds.
- Adopt path-aware methods (next slide).

The Triple Barrier Method

A more realistic approach based on trading logic.

- Label based on which of three events occurs first:
  1. Hits **stop-loss** barrier
     $\Rightarrow y_i = -1$
  2. Hits **profit-taking** barrier
     $\Rightarrow y_i = +1$
  3. **Time-out** at horizon h
     - $\Rightarrow y_i = 0$, or sign of return
- *Encodes risk management logic directly into the labeling process.*

Path	Label
	-1
	$+1$
	0 or $+1$
	0 or -1

Triple Barrier: Implementation Details

- Width of barriers can be based on predicted volatility over the interval.
- Records the time $t_{i,1}$ of the first barrier hit.

Advantage: Path-aware — labels depend on what happens during the position.

Caveat: Barrier crossing is a discrete event and may miss label transitions by a small margin.

Alternative: The caveat is addressed by the following method.

Labeling by Detecting Trends (Trend Scanning)

Imperial College Business School

Trend Scanning: label based on observed trends without explicit barriers or fixed horizon h.

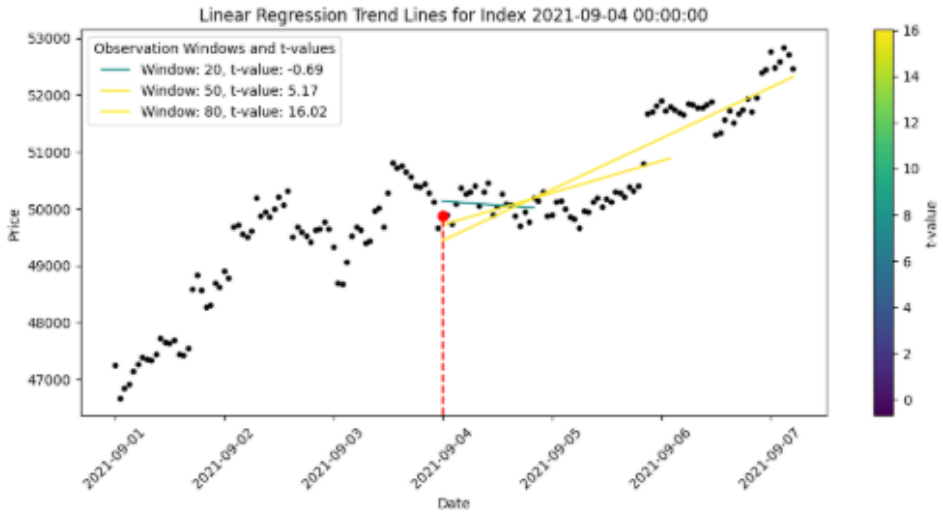
- Fit a linear model to a forward-looking window of length H:

$$x_{t+h} = \beta_0 + \beta_1 h + \varepsilon_{t+h}, \quad h = 0, \ldots, H-1 \quad \Rightarrow \quad \hat{t}_{\widehat{\beta}_1} = \frac{\widehat{\beta}_1}{\widehat{\sigma}_{\widehat{\beta}_1}}$$

- H is the **look-forward period**.
- Different values of H produce different t-statistics.
- We select the H that maximizes $|\hat{t}_{\widehat{\beta}_1}|$.
- Label is assigned based on the sign and magnitude of the most significant slope.

Trend Scanning in Practice

- Trend Scanning can also be used backward in time to generate **features** indicating historical trend strength.
- This method is data-driven and adaptive — it avoids arbitrary thresholds.
- **Programming Session 1:** Trend Scanning Implementation.



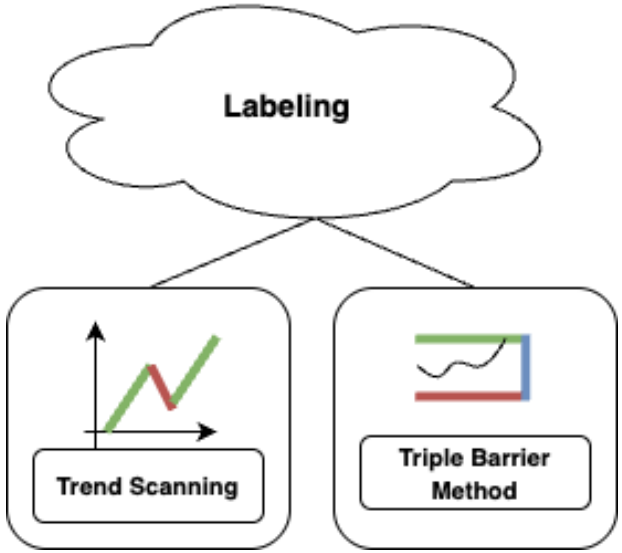
Labeling: Summary of Methods

Fixed-Time Horizon:

- Simplest method: return over a fixed period h.
- Ignores path-dependence, volatility, and trading constraints.

Path-Aware Methods:

- **Triple Barrier:** encodes risk management — profit-taking, stop-loss, and max-hold time.
- **Trend Scanning:** selects statistically significant trend based on forward regression.



Quiz Time!

Click here to take the quiz

Outline

General Setup:

- N: number of samples, L: number of classes
- $y_i \in \{1, \ldots, L\}$: true label for sample i
- $\hat{y}_i \in \{1, \ldots, L\}$: predicted label for sample i
- $\hat{\mathbf{s}}_i = f(\mathbf{x_i}) \in \mathbb{R}^L$: predicted score vector.
- $\hat{s}_{i,\ell}$: predicted score for class ℓ on sample i
- **Predicted Scores** ($\hat{s}_{i,\ell}$) are used for probabilistic evaluation (e.g., log loss, AUC).
- **Predicted Labels** are computed via:
  - **Multiclass:** $\hat{y}_i = \arg\max_\ell \hat{s}_{i,\ell}$
  - **Binary:** $\hat{y}_i = \mathbb{1}[\hat{s}_{i,1} > \tau]$, with threshold $\tau \in [0, 1]$

Evaluation Metrics – Accuracy

Imperial College Business School

Accuracy Score measures the proportion of correct predictions:

$$\text{Accuracy} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}[\hat{y}_i = y_i]$$

- Simple and intuitive.
- Works best for balanced classification problems.
- Can be misleading for imbalanced classes.

Confusion Matrix (Binary Classification):

	Predicted Positive	**Predicted Negative**
Actual Positive	TP	FN
Actual Negative	FP	TN

- TP: True Positives — correctly predicted positives
- FP: False Positives — incorrectly predicted positives
- FN: False Negatives — missed actual positives
- TN: True Negatives — correctly predicted negatives

Evaluation Metrics – Precision and Recall

Imperial College Business School

Precision: How many of the predicted positives are actually positive?

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{\text{TP}}{\text{Predicted Positive}}$$

Recall: How many of the actual positives did we correctly identify?

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{\text{TP}}{\text{Actual Positive}}$$

F1 Score: Harmonic mean of precision and recall

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Useful when dealing with class imbalance or cost-sensitive errors.

Binary Classification:

$$\text{LogLoss} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{s}_i) + (1 - y_i)\log(1 - \hat{s}_i)$$

Multiclass:

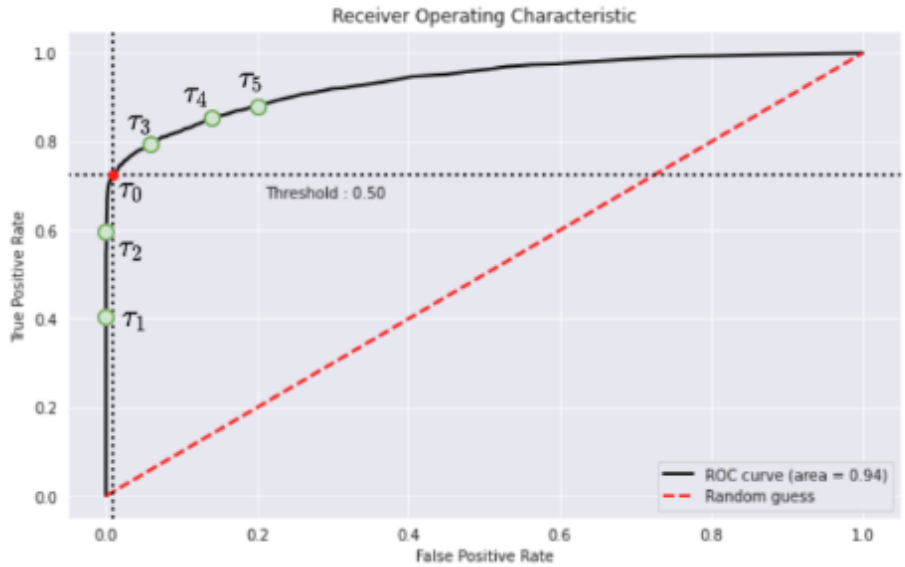
$$\text{LogLoss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{l=1}^{L} y_{il} \log(\hat{s}_{il})$$

Uses predicted scores — not just final decision. Strongly penalizes high confidence on wrong class.

Imperial College Business School

ROC (Receiver Operating Characteristic) Curve:

- Plots **True Positive Rate (TPR)** vs. **False Positive Rate (FPR)**.
- TPR (Sensitivity): $\frac{TP}{TP+FN}$
- FPR: $\frac{FP}{FP+TN}$
- Each point corresponds to a different threshold on predicted scores.

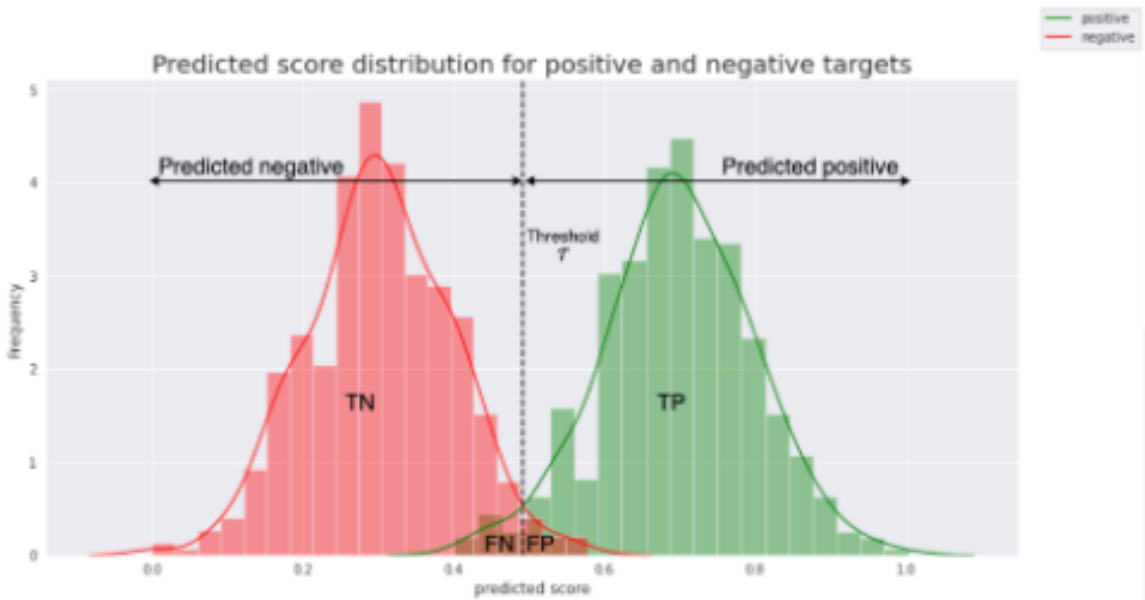


Threshold dependence: Changing the threshold traces the full ROC curve.

Imperial College Business School

AUC (Area Under ROC Curve):

- **Interpretation:** The probability that a randomly selected positive example has a higher score than a randomly selected negative one. (Optional: Click *here* to understand why both definitions are equivalent.)
- Threshold-independent: considers all possible classification thresholds.



Evaluation Metrics – Summary

Key Metrics to Evaluate Classification Models:

- **Accuracy** – Proportion of correct predictions. Works best with balanced classes.
- **Precision & Recall** – Precision: TP / Predicted Positive. Recall: TP / Actual Positive.
- **F1 Score** – Harmonic mean of precision and recall. Useful in imbalanced settings.
- **Log Loss** – Uses predicted scores. Penalizes overconfidence in wrong predictions.
- **ROC Curve** – Plots TPR vs. FPR for all thresholds.
- **AUC** – Probability that a randomly chosen positive is ranked above a negative.

Outline

Hyperparameter Tuning with Cross-Validation

Imperial College Business School

Goal: Find the best set of hyperparameters $h \in \mathcal{H}$ that generalizes well.

- For each hyperparameter configuration $h \in \mathcal{H}$, we evaluate the model using K-fold cross-validation.
- Performance is measured using an evaluation metric $\mathcal{E}$, e.g., accuracy, log-loss, AUC.
- Select $h^* = \arg\max_{h \in \mathcal{H}} \mathcal{E}^h$

Example (Tree-Based Models):

- $\mathcal{H} = \{\text{max depth}, \text{min samples split}, \text{criterion}, \dots\}$
- Use CV to compare scores across $h \in \mathcal{H}$

Algorithm Cross-Validation for Hyperparameter Tuning

Require: Dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$, hyperparameter grid $\mathcal{H}$, evaluation metric $\mathcal{E}$, number of folds K

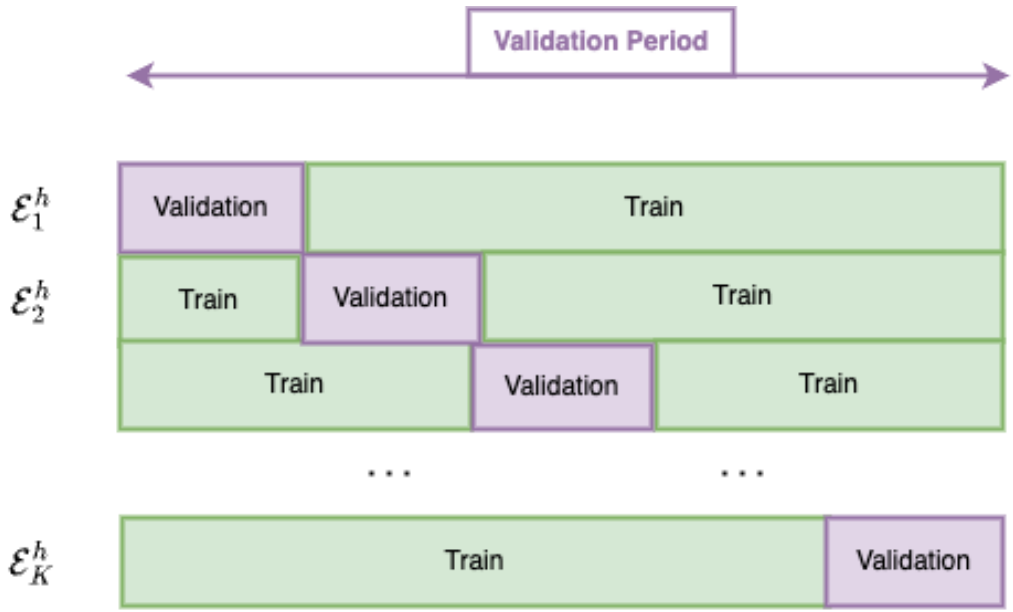
Ensure: Best hyperparameter h^*

for each $h \in \mathcal{H}$ **do**
  Initialize $\mathcal{E}^h \leftarrow 0$
  for each fold $k = 1$ to K **do**
    Split data into training set $D_{\text{train}}^{(k)}$, test set $D_{\text{test}}^{(k)}$
    Train model with h on $D_{\text{train}}^{(k)}$
    Compute score $\mathcal{E}_k^h \leftarrow \mathcal{E}(D_{\text{test}}^{(k)})$
  end for
  Average score: $\mathcal{E}^h \leftarrow \frac{1}{K} \sum_{k=1}^{K} \mathcal{E}_k^h$
end for
return $h^* = \arg\max_{h \in \mathcal{H}} \mathcal{E}^h$

Imperial College Business School

K-Fold Cross-Validation:

- Data is split into K folds.
- Each fold is used once as a test set, remaining as training.
- Final score is the average across all folds.



$$h \in \mathcal{H} \quad \rightarrow \quad \mathcal{E}^h = \frac{1}{K} \sum^{K} \mathcal{E}_k^h$$

Standard CV fails in finance due to time-series structure:

- Data is not IID — financial time series are autocorrelated.
- Overlapping labels cause **label leakage** between training and test sets.

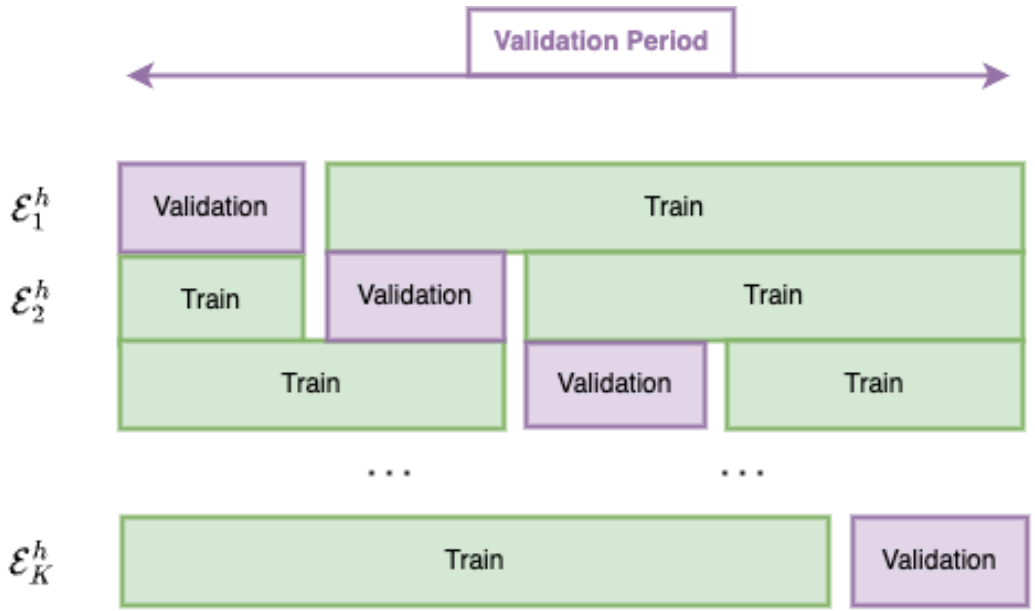
Two key techniques from to prevent leakage (from [2]):

- **Purging:** remove training observations whose label periods overlap with test labels.
- **Embargo:** exclude training data immediately following the test window to avoid serial correlation.

Imperial College Business School

Purged K-Fold Cross-Validation:

- Removes overlapping label periods (purging).
- Applies a time-based buffer around test windows (embargo).
- Ensures the training set is informationally clean.



$$h \in \mathcal{H} \quad \rightarrow \quad \mathcal{E}^h = \frac{1}{K} \sum_{k=1}^{K} \mathcal{E}_k^h$$

Feedback Poll

Click here to participate in the poll

Outline

Why Do We Need Meta-Models?

Imperial College Business School

A primary model — whether targeting trend detection or mean reversion — will perform well only in specific market contexts.

- Market regimes evolve: what works in one regime may fail in another.
- We want to either:
  - Adjust position sizing based on some level of confidence.
  - Filter out trades in regimes where the primary model underperforms.

Meta-labeling trains a secondary model to detect the contexts where the primary model has edge — and when to trust it.

What is a Meta-Model?

Imperial College Business School

Meta-Model: A secondary ML model learns when to trust the primary model.

- Introduced by Marcos López de Prado ([2]).
- Primary model predicts the trade direction.
- Metamodel decides whether to act (Trade T) or abstain ($\overline{T}$).
- Helps convert a weak signal into a strong one.

Applicable to any directional model or expert.

Metamodel Features vs Primary Model Features

Imperial College Business School

Key distinction:

- The **Primary Model** is trained to detect trends or predict return direction.
- The **Metamodel** is trained to detect the conditions under which the primary model works well.

Why separate features?

- Using the same features risks learning redundant information.
- The Metamodel is not predicting the market — it is predicting the *performance of a model*.

Think of the Metamodel as a regime detector: it learns the environments where the primary model has alpha.

Meta-Labels: The Triple Barrier Method

Imperial College Business School

We define a meta-label $(T, \overline{T})$ using the outcome of each trade:

- T: If the trade would have hit profit-taking before stop-loss.
- $\overline{T}$: Otherwise — trade not worth taking.

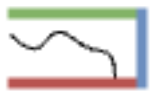
		
Primary Long signal	$\overline{T}$	T
Primary Short signal	T	$\overline{T}$

Figure: Meta-labels are derived from the result of the primary model's trade using the triple-barrier method.

Overview of the Meta-model Framework

The Metamodel takes as input the predictions of the primary model, as well as features describing market conditions, and learns to filter or size trades based on expected performance.

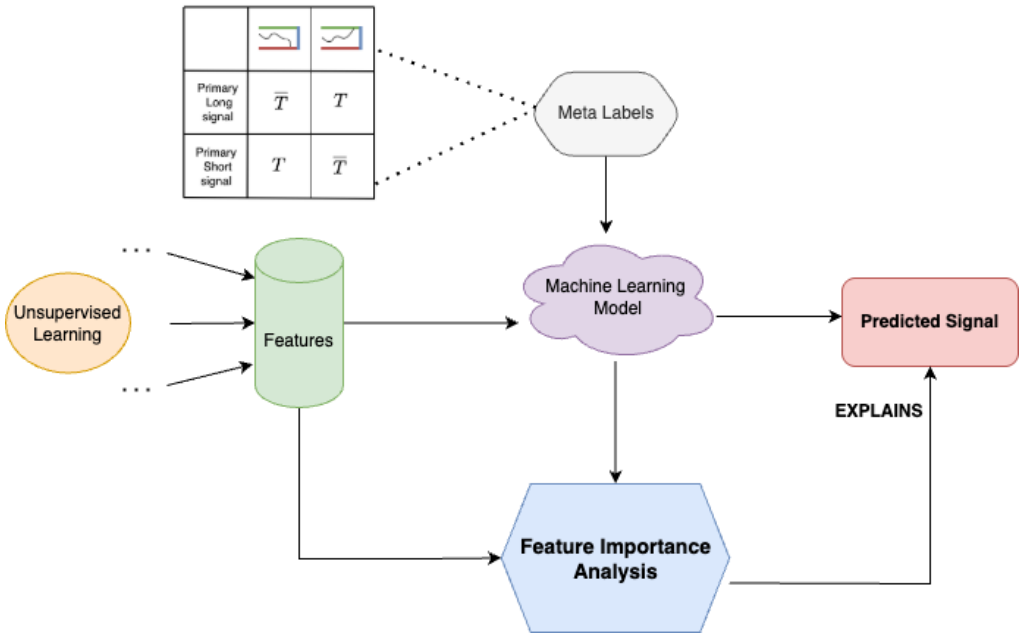


Figure: The Metamodel learns a mapping between market context and the reliability of the primary model's signal.

Quiz Time!

Click here to take the quiz

Trading Recall for Precision: Improving F1 Score

Imperial College Business School

Primary model: high recall (takes every trade), low precision.

Metamodel: reduces false positives to improve precision.

- The trade-off is controlled by a decision threshold τ.
- Goal: maximize F1-score = harmonic mean of precision and recall.

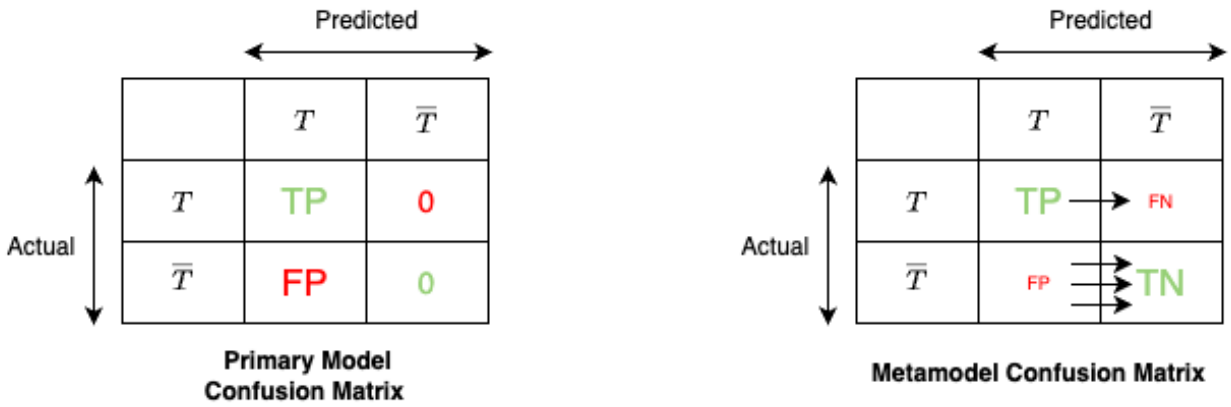


Figure: The Metamodel improves the confusion matrix by rejecting low-confidence trades.

Applying Meta-Labeling to a Crypto Strategy

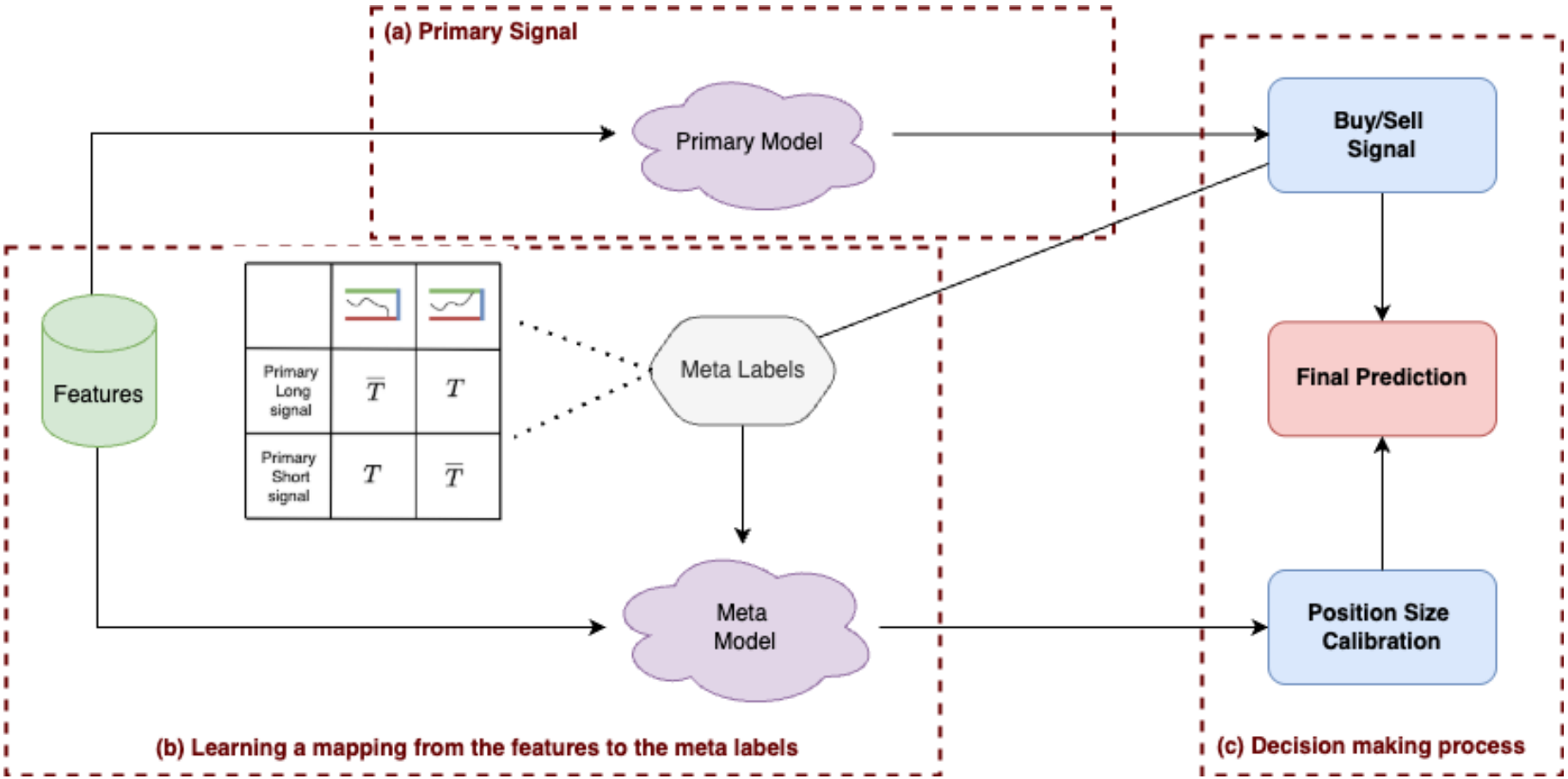
Imperial College Business School

We'll implement meta-labeling in Lecture 5 to filter out weak signals from a crypto directional model.

5 steps:

1. We have trained a primary model to predict direction (Long/Short).
2. Use the triple barrier method to label trades as T or $\overline{T}$.
3. Train a Metamodel to predict whether to take the trade.
4. Use Feature Importance Analysis to understand Market Regimes under which the primary model performs well.
5. Compare the performances of the primary model (Long/Short) with the Final Strategy (Long / Short / Neutral).

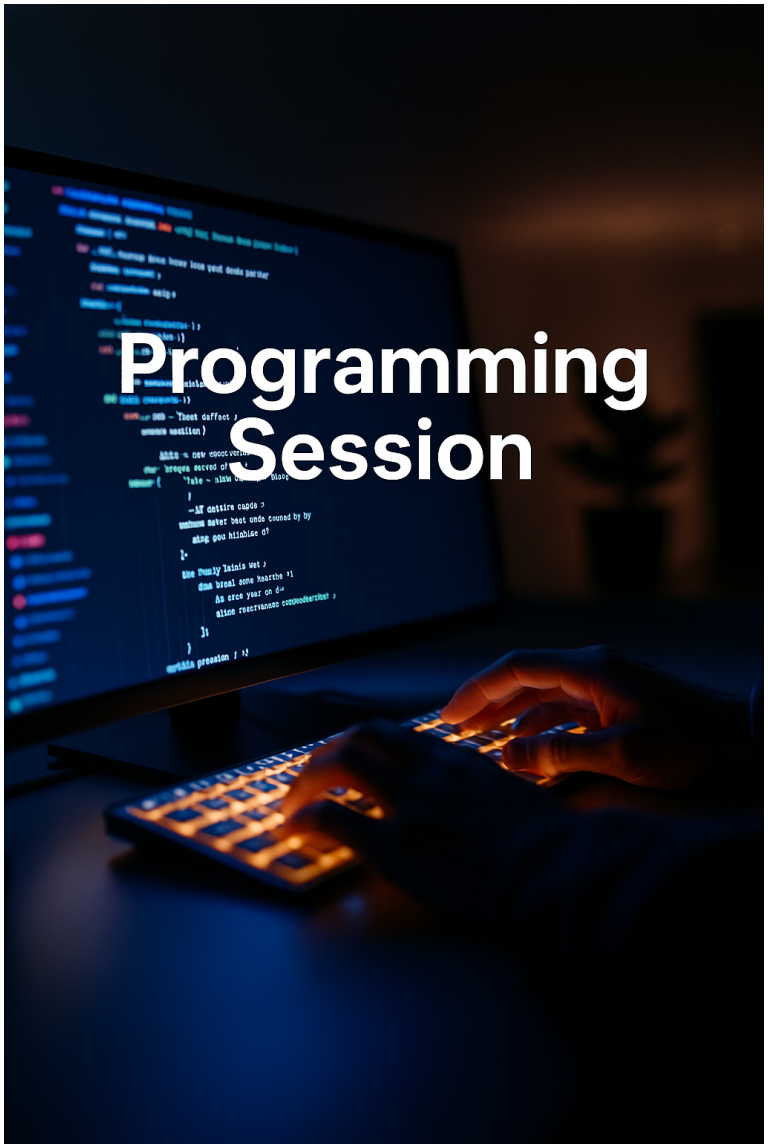
Applying Meta-Labeling to a Crypto Strategy



Feedback Poll

Click here to participate in the poll

Outline



Programming Session 1: Implementation of the Trend Scanning Method.

- *Click here to access the programming session*

Solution will be posted tonight on the GitHub page.

- *Click here to access ccess the GitHub Page*

Thank you for your attention

References I

[1] Yoshua Bengio et al. "Gflownet foundations". In: *Journal of Machine Learning Research* 24.210 (2023), pp. 1–55.

[2] Marcos Lopez De Prado. *Advances in financial machine learning*. John Wiley & Sons, 2018.

[3] Salem Lahlou et al. "A theory of continuous generative flow networks". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 18269–18300.

[4] Hachem Madmoun. "Creating Investment Strategies Based on Machine Learning Algorithms". PhD thesis. École des Ponts ParisTech, 2022.

[5] Nikolay Malkin et al. "GFlowNets and variational inference". In: *arXiv preprint arXiv:2210.00580* (2022).