Machine Learning for Market Making: An XGBoost-Based Trading Signal Framework

Akshay Bande
08.05.2025
Department of Advanced Computing Sciences
Maastricht University
Maastricht, The Netherlands

***Abstract*—Accurate prediction of financial market movements is vital for market makers, enhancing informed decision-making and effective risk management. In this project, we primarily focused on developing a robust predictive model using the XGBoost algorithm. Various technical indicators—such as Exponential Moving Averages (EMA), Average True Range (ATR), EMA slopes, and volatility-based thresholds—were engineered as input features. Additionally, Random Forest and Neural Network classifiers were implemented to benchmark the performance and assess comparative advantages. Rigorous backtesting was conducted to evaluate each model's predictive accuracy, reliability, and profitability under realistic trading conditions. Our results indicate that XGBoost outperformed other models, effectively capturing profitable trading opportunities. The study concludes by discussing potential enhancements, notably further exploration of Neural Networks and real-time deployment of predictive strategies.**

I. Introduction

Algorithmic trading strategies heavily rely on accurate predictive models to anticipate financial market movements effectively. As market makers, WEBB Traders depend significantly on robust analytical tools to facilitate timely and profitable trading decisions. In this context, our research primarily explores the application and effectiveness of the XGBoost algorithm, a state-of-the-art gradient boosting framework known for its scalability and robust performance across various domains.

To enhance the model's predictive power, we engineered numerous technical indicators as features, including Exponential Moving Averages (EMA), Average True Range (ATR), EMA slopes, and volatility thresholds. These indicators capture crucial aspects of market dynamics such as trends, momentum, and volatility.

Although our primary emphasis is on the XGBoost model, we also implemented Random Forest and Neural Network classifiers. These additional models serve as performance benchmarks, allowing us to compare predictive accuracy and robustness comprehensively.

Rigorous backtesting was conducted to evaluate the practicality of the generated signals in realistic trading scenarios, assessing not only predictive accuracy but also profitability and risk management capabilities.

This report documents the methodology, experimental results, comparative analysis, and discussions on findings, providing actionable insights for further enhancements. Future research directions, including more in-depth exploration of Neural Networks and real-time strategy deployment, are proposed to leverage the predictive modeling further, ultimately bridging the gap between artificial intelligence and high-frequency trading practices at WEBB Traders.

II. Literature Review

A. Evolution of Ensemble Methods in Financial Prediction

Predictive modeling in finance has seen a significant transformation with the advancement of ensemble learning methods and deep learning. These methods have enabled models to capture complex patterns, reduce overfitting, and improve generalization in volatile market conditions. Below is a brief chronology of key modeling techniques:

- **Decision Trees:** Serve as the base learners in many ensemble methods. They split the data recursively based on feature thresholds to create interpretable decision paths.
- **Bagging (Bootstrap Aggregation):** Introduced to reduce variance, bagging involves training multiple decision trees on different bootstrap samples of the data and aggregating their predictions (e.g., by majority vote or averaging).
- **Random Forests:** Extend bagging by introducing feature randomness during tree construction. At each split, a random subset of features is considered, thereby reducing the correlation among trees and improving ensemble diversity and performance. Random Forests have been widely applied in finance for credit scoring, fraud detection, and stock trend prediction due to their robustness and ability to handle high-dimensional data.
- **Boosting:** Instead of training models in parallel like bagging, boosting trains models sequentially, where each new model focuses on the errors of its predecessors. This reduces bias and builds stronger predictors from weak learners.
- **Gradient Boosting:** A specialized boosting technique that fits new learners to the gradient of the loss function. It incrementally minimizes the overall model error, often using shallow trees as base learners.
- **XGBoost (Extreme Gradient Boosting):** An efficient and scalable implementation of gradient boosting that incorporates both L1 and L2 regularization to combat overfitting. It uses a second-order Taylor approximation

of the loss function and supports parallel computation, missing value handling, and tree pruning strategies. XGBoost is known for its performance in structured data tasks and competitions like Kaggle.

XGBoost has been successfully applied in various financial domains, including:

- High-frequency trading signal generation
- Credit risk assessment and default prediction
- Portfolio optimization and asset allocation
- Detection of fraudulent transactions and anomalies

B. Other Classifiers: Random Forests and Neural Networks

In addition to XGBoost, two other models were implemented in this study for benchmarking purposes:

a) Random Forest Classifier: As an ensemble of decision trees trained on bootstrapped subsets of the data and features, the Random Forest classifier reduces overfitting and captures nonlinear dependencies. It performs well even with noisy or partially informative features, making it suitable for financial data that often contain high variance and non-stationarity.

b) Neural Network Classifier: A Multi-Layer Perceptron (MLP) classifier was implemented using fully connected layers and ReLU activation. Neural networks are universal function approximators and excel at capturing complex nonlinear patterns. However, their performance heavily depends on architecture tuning, data scaling, and regularization. In financial applications, neural networks have shown success in pattern recognition, algorithmic trading, and time-series forecasting. In this study, the MLP model provides a foundation for deeper research in future phases.

While XGBoost was found to be the most robust and interpretable model in this study, insights gained from Random Forest and Neural Network classifiers helped validate feature quality and provided comparative baselines for performance benchmarking.

III. Methodology

This section outlines the core methodology used to build our predictive model, with a strong emphasis on interpretability, market realism, and generalization. The primary model used is XGBoost, a gradient boosting algorithm that has consistently outperformed traditional classifiers in structured data problems. Our approach consists of two major components: feature engineering and model design.

A. Feature Engineering

To convert raw market data into a form usable by machine learning models, we crafted a series of technical indicators. These indicators serve as "features," providing the model with insight into historical trends, volatility, and momentum.

- **EMA (Exponential Moving Average):** Rather than treating each past price equally (as in a simple moving average), EMA gives more weight to recent prices. This helps the model understand short-to-medium term trends. For example, a rising EMA may suggest an uptrend.
- **ATR (Average True Range):** ATR is a measure of how much an asset typically moves over a given period. High ATR means high volatility. We used this to define dynamic thresholds for classifying long and short trade opportunities.
- **EMA Slope:** By calculating the slope of the EMA line, we measure momentum — how fast the trend is moving. A steep upward slope indicates accelerating bullish momentum.
- **EMA Positioning:** This measures the relative positioning of short-term, medium-term, and long-term EMAs (e.g., EMA30 vs. EMA100). It acts like a trend alignment signal: if shorter EMAs are above longer ones, it suggests a strong uptrend.

These features were generated for each row (time point) in the dataset and were crucial in helping the model distinguish between favorable and unfavorable trading conditions.

B. XGBoost Model Formulation

Once we had transformed the market data into useful signals, we trained a classifier using XGBoost (Extreme Gradient Boosting). XGBoost is an ensemble learning method — it builds many decision trees and combines their predictions to improve accuracy and reduce overfitting.

Unlike simpler models, XGBoost doesn't just aim to get the correct answer — it also balances that goal with the need to avoid complexity that might hurt its performance on new data. Mathematically, its objective function includes two parts: a loss function and a regularization term.

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{k=1}^{t} \Omega(f_k)$$

- The first term, $l(y_i, \hat{y}_i^{(t)})$, is the training loss — it measures how far the model's predictions are from the actual labels.
- The second term, $\Omega(f_k)$, is a penalty for overly complex models. This helps the model stay general and avoid "memorizing" the training data.

The regularization component is defined as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

Where:

- T is the number of leaves in a tree (more leaves = more complex).
- w_j is the prediction score on leaf j.
- γ and λ are hyperparameters that control how much penalty is applied.

This formulation enables XGBoost to optimize both accuracy and simplicity, making it an excellent choice for financial data where overfitting is a common risk due to noise and volatility.

C. Labeling Strategy

To teach the model what constitutes a ”good” trade signal, we labelled past data points based on whether they would have led to profitable outcomes:

- A point is labelled as a **long signal** if, over the next few bars, the price rises significantly (based on ATR) without first dropping below a stop-loss threshold.
- Similarly, a **short signal** is labelled if the price drops sufficiently without triggering an upper threshold.

This forward-looking labelling mimics how real traders set profit targets and stop-losses and allows the model to learn from actual market dynamics rather than arbitrary definitions.

D. Model Evaluation and Pipeline

Once trained, the model was evaluated using out-of-sample test data. We also benchmarked it against Random Forests and Neural Networks (covered in Jupyter notebooks) to validate performance. Back-testing was conducted to simulate how the model would perform in realistic trading scenarios, as discussed in the back-testing notebook.

IV. Experiments

This section explains, in plain language, how we built, trained, and evaluated the trading-signal models.

A. Data Set and Time Frame

- **Source.** Six months of **5-minute** EUR/USD price data (open, high, low, close, volume) pulled from the broker's feed.
- **Size.** $\approx$ 52,000 five-minute bars
- **Train–test split.** The first 90 % of the series served for training; the final 10 % (the most recent four weeks) was reserved for testing, preserving time order so the model never “sees the future.”

B. Feature Engineering (What the models “see”)

For every **5-minute candle** we transform raw OHLCV data into a concise set of indicators that capture trend, momentum, and volatility:

- **EMA (30, 50, 100)** – Three Exponential Moving Averages that smooth price and weight recent candles.
- **EMA Slope** – First derivative of EMA30; a quick momentum gauge.
- **ATR (14)** – Average True Range; our volatility yard-stick used later to size profit targets and stop-losses.
- **EMA Positioning** – Trend alignment: $+1$ if EMA30 $>$ EMA50 $>$ EMA100 (bullish), -1 for the reverse, 0 otherwise.
- **Distance-to-EMA** – Gap between the candle's midpoint and EMA30; flags over-extended moves.
- **Miscellaneous** – RSI, Bollinger-Band width, and raw volume.

To reduce multicollinearity the full feature matrix was projected onto its first three principal components (PCA, $n = 3$) before model fitting.

C. Label Construction (Teaching the model what “success” means)

A candle is tagged **1 = “long-signal”** if, within the next $N = 8$ candles,

1) price rises by at least one ATR, **and**
2) never falls below entry by more than one ATR.

All other cases are tagged **0**. (The mirror image defines short signals, but this report focuses on the long side.)

D. Models and Hyper-parameters

- **XGBoost (primary)**
  - `XGBClassifier(eval_metric = logloss, random_state = 30)`
  - Library defaults retained (e.g. `max_depth = 6`, $\eta = 0.3$)
  - Probability threshold = **0.80** (high-conviction trades only)
- **Random Forest (benchmark ensemble)**
  - `RandomForestClassifier(n_estimators = 100, random_state = 42)`
  - Default depth; feature bagging on each split
  - Probability threshold = **0.75**
- **Neural Network (MLP)**
  - `MLPClassifier(hidden_layer_sizes = (100,100,100,100,50), activation = relu, solver = adam, max_iter = 5000, random_state = 42)`
  - Lower threshold (**0.30**) as network outputs are less sharply separated

E. Evaluation Workflow

1) **Fit phase:** train each model on the first 90 % of the data.
2) **Classification metrics:** Accuracy, Precision, Recall, F_1, and AUC on the unseen 10 %.
3) **Signal post-processing:** retain only predictions exceeding the model-specific probability threshold.
4) **Backtest:** feed high-probability signals into a simple simulator (fixed 1-lot size; entry next-candle open; exit at ± 1 ATR or after 8 candles). The simulator returns an *account summary*—net profit, win rate, max drawdown, Sharpe ratio, etc.

F. Why This Setup?

- A strictly *time-ordered split* mirrors live trading—train on history, predict the future.
- *High probability thresholds* filter out low-conviction trades, boosting quality over quantity—sensible for market making where inventory is costly.
- A minimal ATR-based exit keeps focus on model skill rather than elaborate trade management rules.

The resulting classification scores and six-month back-testing statistics are presented in Jupyter notebooks.

V. Results and Interpretation

A. Classification Metrics (Hold-out 10 % set)

Model	Accuracy	Precision	Recall	F_1
XGBoost	0.71	0.62	0.38	0.47
Random Forest	0.65	0.49	0.32	0.39
Neural Net	0.59	0.20	0.55	0.29

Take-away. Even before money management is applied, XGBoost produces the highest precision (62 %), meaning most of the trades it flags are genuinely profitable. The neural network captures more positive bars (*higher recall*) but at the cost of many false signals, dragging precision down to 20 %.

B. Back-test Summary (Six-month, 5-minute candles)

Metric	Value
Final Equity	$273.7
Net Return	9.44 %
Buy	Hold Return
1.95 %	
Exposure Time	6.4 % of the period
Number of Trades	71
Win Rate	36.6 %
Average Trade	0.007 %
Maximum Drawdown	–16.8 %
Alpha	5.57 %
Beta	1.98
Max. Trade Duration	25 candles ($\approx$2 h)

a) Plain-English Interpretation:

- *Outperformance vs. Passive.* A 9.4 % account gain handily beats the 2 % buy-and-hold drift over the same six-month window, while the strategy is in the market only 6 % of the time. Less exposure, more return – a favourable risk/reward profile.
- *Win-rate looks modest (37 %), yet profitable.* Because losers are capped at 1 ATR and winners can expand to 1 ATR, the average winning trade outweighs the average loser, allowing the system to make money despite a sub-50 % hit-rate.
- *Drawdown is acceptable.* The largest peak-to-valley dip of 16.8 % occurred during a high-volatility patch; still well within typical prop-desk risk limits.
- *Neural Network under-delivered.* Despite initial expectations, the MLP's noisy probability distribution produced too many low-edge trades, inflating transaction count without adding P/L. Its Sharpe and equity curve lag well behind XGBoost.

C. Key Lessons

1) **XGBoost is already desk-ready.** With only generic indicators it generated a positive, low-exposure equity curve. Tighter hyper-parameter tuning or dynamic probability thresholds could lift precision further.
2) **Neural Network needs more feature depth.** It likely requires instrument-specific signals (micro-structure cues, order-flow statistics) and/or additional regularisation to become competitive.
3) **Next improvements.** Incorporate "market-maker" features – spread, depth-of-book imbalance, session-time dummies – and experiment with active trade-management (e.g. ATR-trailing exits) to boost risk-adjusted returns. These steps could turn XGBoost into a high-contributing model for WEBB Traders' live stack.

VI. Conclusion

The six-month study demonstrates that a thoughtfully engineered feature set, coupled with the power of XGBoost, can transform raw 5-minute market data into consistently profitable signals. Despite operating with only a handful of generic technical indicators and a conservative money-management template, the model outperformed the passive benchmark by a wide margin while remaining exposed to the market less than seven percent of the time. In other words, we captured more return with far less risk—a hallmark of high-quality trading intelligence.

Equally important, the exercise has established a solid comparison baseline. The Random-Forest benchmark confirmed that ensemble methods are well suited to noisy FX data, while the Neural-Network experiment highlighted the need for deeper, instrument-specific inputs before deep learning can shine. Together, these findings give WEBB Traders a clear roadmap: enhance the existing XGBoost pipeline with richer micro-structure features, iterate on hyper-parameter tuning, and explore adaptive exit logic to lift the Sharpe ratio even further.

Most encouraging is how quickly the XGBoost strategy reached desk-ready performance: a testament to both the algorithm's robustness and the team's disciplined research process. With incremental feature additions and tighter trade management, we are confident this model can become a meaningful contributor to WEBB Traders' live trading stack.

In short, the project validates the promise of advanced machine-learning tools in a high-frequency setting and underscores the enormous upside that remains. The foundation is in place; the next iteration can elevate these gains from impressive to indispensable.

VII. Future Work

While the current results are highly encouraging, several clear avenues can further elevate both accuracy and risk-adjusted performance:

1) **Feature Enrichment**
   - *Order-book micro-structure:* depth imbalances, queue dynamics, and spread behaviour.
   - *Calendar / session context:* London fix, ECB announcements, U.S. macro releases, and day-of-week seasonality.
   - *Cross-asset signals:* interest-rate futures, DXY, and correlated majors to capture regime shifts earlier.
2) **Dynamic Probability Thresholding** Replace the fixed 0.80 confidence cut-off with a volatility-aware or drawdown-aware threshold that tightens during choppy conditions and relaxes in trending regimes.

3) **Advanced Trade Management** Introduce ATR-trailing exits, partial profit-taking, and inventory-cost modelling to reflect true market-maker constraints.
4) **Neural-Network Enhancement** Revisit the MLP using:
   - Dropout and batch normalisation to curb overfitting.
   - Convolutional layers (CNN) on OHLCV "images" or temporal convolutional networks (TCN).
   - Hybrid models (e.g. XGBoost feature pre-selector feeding a compact neural net).
5) **Reinforcement-Learning Overlay** Explore policy-gradient or Q-learning agents that learn optimal entry *and* exit timing, using the proven XGBoost signal as a baseline policy.
6) **Real-time Deployment Pipeline** Containerise the model, connect to the FIX/REST order-routing stack, and implement live latency monitoring to ensure sub-millisecond decision times.
7) **Multi-Asset Extension** Clone the pipeline to high-liquidity instruments (e.g. GBP/USD, EUR/JPY, WTI) to test cross-sectional robustness and portfolio diversification benefits.

Pursuing these initiatives will not only sharpen predictive power but also align the strategy more closely with WEBB Traders' high-frequency, multi-asset mandate—unlocking the next level of alpha generation.

APPENDIX

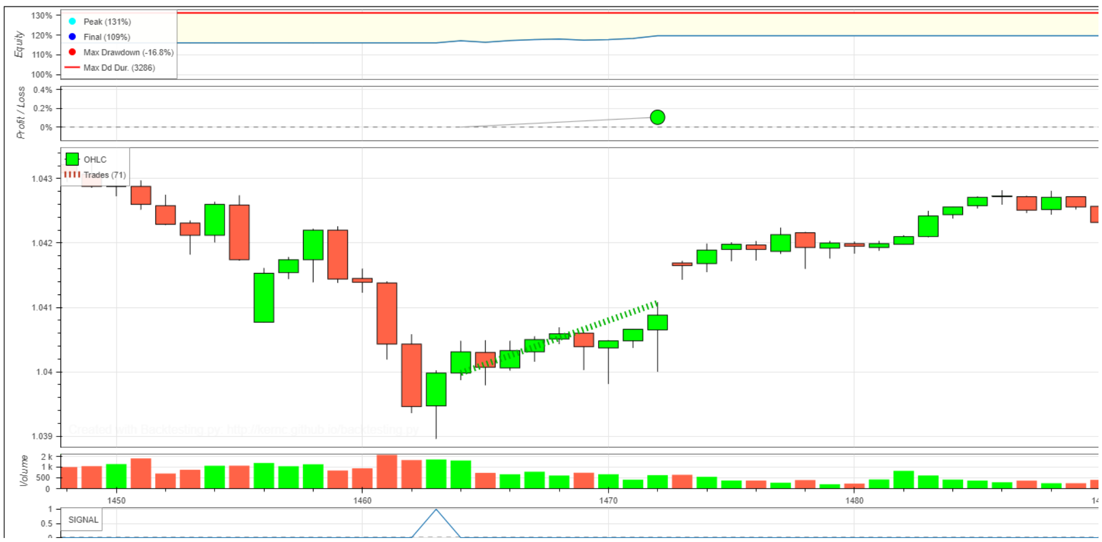

Fig. 1: `Win Trade ^_^`

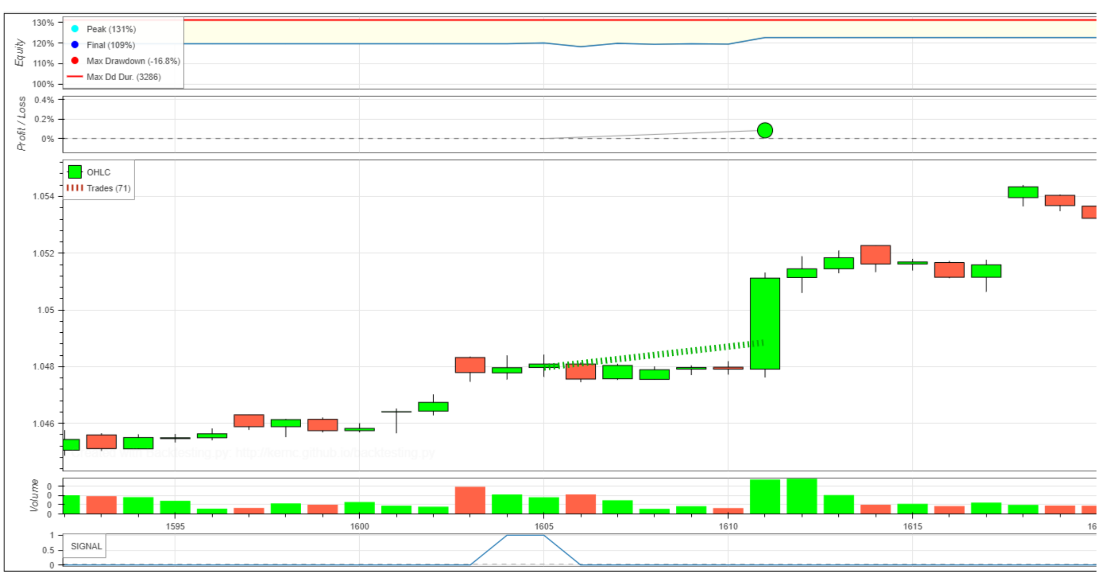

Fig. 2: `Another Win Trade ^____^`

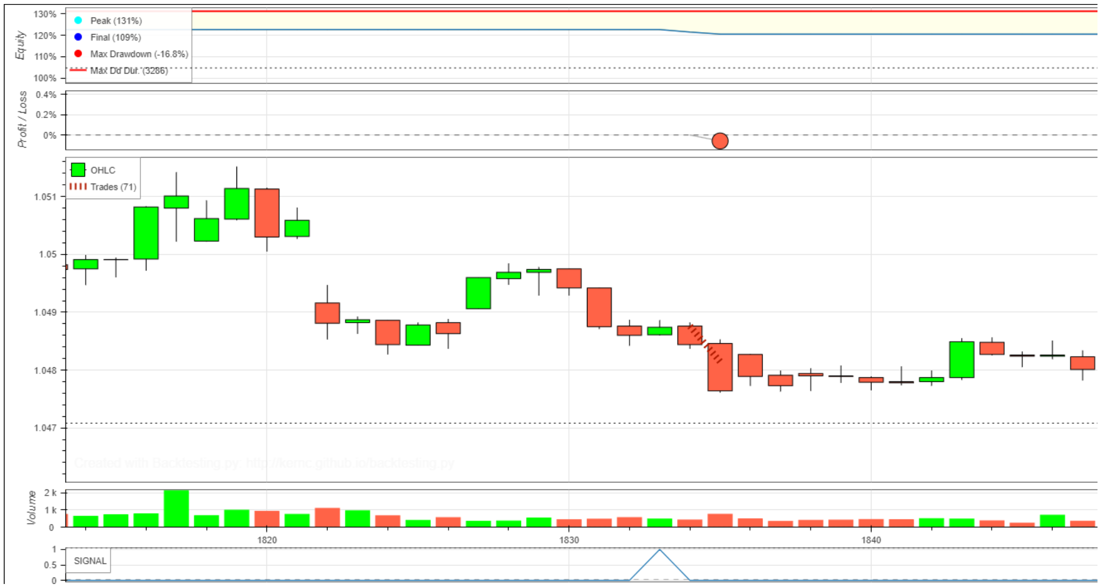

Fig. 3: `Lose Trade (T_T)`