UNIVERSITY OF LEEDS

Using LSTM Model for Meta-labeling in Quantitative Trading

A Practical Analysis with Minute Stock Data

Abstract

This research investigates the efficacy of Long Short-Term Memory (LSTM) models in predicting stock price movements, using minute-level data for Apple Inc. Based on this prediction, a thresholding mechanism is used to construct the underlying trading strategy. Furthermore, it assesses the potential of meta-labeling techniques to refine trading signals, filtering false positives and improving the profitability of trading strategies. The LSTM model proved capable of predicting stock price movements at minute frequency. Meanwhile, the application of meta-labeling, particularly the Empirical Cumulative Distribution Function (ECDF) method, showed potential in enhancing investment metrics, especially risk-adjusted return. However, the study also revealed that the efficacy of meta-labeling varies significantly depending on the trading threshold and method used. The results highlight the importance of calibrating these elements based on the unique characteristics of each trading strategy and investment style.

Keywords— Quantitative Trading, Long Short-term Memory, Meta-labeling, Stock Price Forecasting

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Financial time series forecasting is pivotal in finance for decision-making, risk management, and algorithmic trading. It offers insights into future trends of financial instruments like stocks, commodities, and currencies. This is particularly important in algorithmic trading, where automated models execute trades with minimal human input.

However, forecasting this data is challenging due to its complex statistical nature, marked by non-stationarity and volatility clustering. Additionally, financial data often shows leptokurtosis, deviating significantly from a normal distribution. Such complexities pose difficulties for traditional models, which assume stationary and normally-distributed data.

In recent years, machine learning (ML) and deep learning (DL) have risen in importance in finance due to their ability to handle complex, non-linear relationships in high-dimensional data. Unlike traditional methods, they don't rely on preset assumptions about data structure. Models like Recurrent Neural Networks (RNNs) and their variant, Long Short-Term Memory (LSTM) networks, excel in financial time series forecasting.

RNNs are designed for sequential data, capturing temporal patterns through recurrent connections. LSTMs further specialize in long-term dependencies using memory cells and gating mechanisms. This allows them to effectively filter and update relevant information over time, making them particularly useful for high-frequency, noisy financial data like minute-level stock prices (Fischer and Krauss 2018).

One key use of financial time series forecasting is in quantitative trading, where algorithms execute trades automatically. This approach offers benefits like quicker execution, reduced human bias, and real-time data processing, thus increasing efficiency and potential profits.

However, a reliable forecasting model alone doesn't ensure profitability; effective risk management is also critical. Failing to manage risks like position sizing and stop-loss levels can lead to losses, even with accurate forecasts. For example, overly conservative or aggressive position sizing can result in underinvestment or poor risk-adjusted returns, respectively.

To address this, meta-labeling is emerging as a solution. It's a machine learning layer that refines base trading strategies, helping to determine optimal position sizes and filter false signals. This improves key metrics like the Sharpe ratio and maximum drawdown (Joubert 2022). Although still a newer concept, meta-labeling offers benefits such as improved interpretability, improved flexibility, avoidance of overfitting, improved risk-return metrics, etc. Meta-labeling has promising applications and there are still many worthwhile parts to be investigated, especially its lack of validation using real market data.

1.2 Research Objectives

The central aim of this dissertation is to delve into the potential of LSTM models for financial time series forecasting, specifically within the domain of minute-level stock volume and price data. Concurrently, this research endeavors to uncover how meta-labelling can be instrumental in refining trading strategy performances by filtering out false positives and enhancing risk-return metrics. In pursuit of this overarching objective, the subsequent pivotal questions arise:

1. How can LSTM models predict the future movements of stocks using minute-level stock volume and price data?
2. In what ways can meta-labeling assist in sieving out false positive trading signals and enhance risk-return metrics?
3. Upon the application of LSTM to meta-labelling, which trading strategies might emerge?

In the realm of quantitative trading, this dissertation offers the ensuing contributions:

1. Practical Application of LSTM with Minute-Level Data: Harnessing the power of LSTM models, this study applies them to minute-level stock data, paving the way for enhanced trading signal generation. Such an approach ensures adaptability to the volatile intraday market movements, granting traders the proficiency to seize short-term opportunities.
2. Real Data Validation for the Benefits of Meta-labeling: By utilizing genuine historical data to craft trading strategies, this research stands to authenticate the professed advantages of meta-labelling, assuring their applicability in tangible trading contexts.
3. Evolution of Trading Strategies via Advanced Deep Learning Models: The amalgamation of state-of-the-art deep learning techniques, primarily LSTM models, into quantitative trading strategies

holds promise for precision in signal generation and a deeper comprehension of intricate market dynamics.

1.3 Overview

The paper's structure is as follows: Chapter 2 provides a literature review on LSTM and meta-labelling. Chapter 3 details the technical aspects of meta-labelling. Chapter 4 discusses the utilised data and constructed features. Chapter 5 focuses on the technical details of the LSTM model. Chapter 6 presents the formation and backtesting of trading strategies. Chapter 7 covers the designed experiments and corresponding results. Lastly, Chapter 8 concludes the study and outlines future research directions.

Chapter 2

Literature Review

2.1 Forecast of Financial Time Series and Deep Learning

Financial time series have a variety of statistical properties that are difficult to model, including non-linearity and non-stationarity, fat tails, heteroskedasticity, and volatility clustering. Cont (2001) applies non-parametric methods to empirical data of asset return, describing the existence of the above unpleasant properties as constraints. This illustrates the difficulty of many conventional statistical models to model financial time series well, because the assumptions hidden behind them often contradict the reality.

Efforts have been made to address these problems. For example, Engle (1982) and Bollerslev (1986) proposed ARCH and GARCH models. By incorporating lagged squared errors and/or lagged conditional variances, ARCH and GARCH models capture the heteroskedasticity present in financial time series data allowing for improved modeling and forecasting of asset returns. Although part of the problem has been solved to some extent, the forecasting of financial time series remains problematic.

Machine learning, and particularly deep learning, provides a promising solution to tackle these challenges. The inherent flexibility and non-linear nature of deep learning models make them well-suited for handling the intricate statistical properties of financial time series. White (1988) proposes an early study applying deep neural networks to the analysis of asset price changes. White tries to predict IBM's daily stock returns and concluded that even simple neural networks are able to predict extremely dynamic behaviour, but that overfitting is a potential obstacle.

Gandhmal and Kumar (2019) systematically reviews techniques used in stock market forecasting, including Bayesian model, Fuzzy classifier, artificial neural networks (ANN), support vector machine and others. It is concluded that ANN is the most common model that gives effective forecasting results in terms of accuracy, mean absolute percentage error (MAPE), and root mean squared error (RMSE). Data pre-processing is also mentioned as a complex task for predicting accurate stock price.

Among various variants of ANN, Recurrent Neural Networks (RNNs) have gained popularity for their ability to model sequential data and capture temporal dependencies and context. One of the most popular RNN units is proposed by Elman (1990). It models sequential data by implicitly representing time through recurrent connections, which endows the networks with a dynamic memory. Unlike traditional feedforward neural networks, RNNs have the ability to retain information from previous time steps, allowing them to capture temporal dependencies and context. This is achieved by feeding back hidden unit patterns to themselves, which enables the internal representations to reflect the task demands within the context of prior internal states.

Training RNNs, however, presents practical challenges due to their complex nature and the difficulty of optimizing the loss function. The vanishing or exploding gradient problem can hinder the learning of long-term dependencies in RNNs. To overcome these challenges, the Long Short-Term Memory (LSTM) architecture was introduced by Hochreiter and Schmidhuber (1997). LSTMs incorporate specialized memory cells and gating mechanisms, facilitating the preservation of relevant information and regulating information flow within the network. This enables LSTMs to capture long-term dependencies effectively, making them a preferred choice for time series forecasting, particularly in financial markets.

Studies have shown the effectiveness of LSTM in stock market forecasting. Fischer and Krauss (2018) applied LSTM networks to predict out-of-sample stock price changes and found them to outperform other memory-free models, including logistic regression, artificial neural networks (ANN), and random forest. They use a simple rules-based trading strategy, buying the recent losers and selling the recent winners. The LSTM was found to be more profitable than other models.

Li and Bastos (2020) provides a systematic review of the use of deep learning for stock market prediction. It was found that 73.5% of the studies used LSTM techniques because of their ability to store memory and solve the problem of gradient disappearance. This percentage includes use alone and mixed with other models. And the limitations of the existing literature are noted, with just over half of the articles having a trading strategy and only 35.3% of the studies examining profitability, with even fewer using risk management.

Furthermore, most of the studies with trading systems use simple rule-based strategies and do not have a regular risk management process (Li and Bastos 2020). The results of this systematic review show us the gap from asset return forecasting to profitable trading strategies, and the difficulty of finding effective position sizing methods.

In conclusion, financial time series forecasting presents challenges due to their complex statistical properties. Deep learning techniques, particularly LSTM, show promise in capturing the characteristics of financial data and outperforming traditional models. Further research is needed to develop profitable trading strategies by integrating accurate predictions with robust risk management practices.

2.2 Meta-labeling

Meta-labeling, initially proposed by Lopez de Prado (2018), addresses the challenge faced by practitioners who possess a model for predicting the direction of investment (long or short) but require a separate model to determine the position size (amount of money investing in that bet). López de Prado emphasizes that binary classification in trading involves a trade-off between type-I errors (false positives) and type-II errors (false negatives). Meta-labeling proves particularly effective in achieving better risk-return metrics.

López de Prado points out four benefits of meta-labeling. Firstly, it allows machine learning models to be built on top of white-box models rooted in economic theory, addressing concerns about interpretability. Secondly, by separating side prediction from size prediction, overfitting issues are alleviated, as the primary and secondary models can use appropriate features independently. Thirdly, the decoupling of side prediction from size prediction offers greater flexibility. Separate secondary models can be built exclusively for long and short positions, enabling better adaptation to market conditions and the construction of more suitable models. Lastly, meta-labeling improves risk-return metrics by focusing on getting the most important predictions correct, preventing low returns from high accuracy on small bets and low accuracy on large bets.

Building upon López de Prado's work, Joubert (2022) presents a framework for applying meta-labeling to design investment strategies. The authors divide meta-labeling into three components: information advantage, filtering for false positives, and position sizing. Synthetic data generated by an autoregressive process of order 3, which eliminates confounding variables associated with real financial data, is employed in controlled experiments to verify the claimed effects of meta-labeling. For the secondary model, which assesses the accuracy of the primary model's output, logistic regression is employed. The output of the logistic regression, falling within the range of [0, 1], is then passed through an empirical cumulative distribution function (ECDF) fitted on the training data. This process results in a position size that invests a larger amount when the outcome has a higher probability of accuracy. The findings confirm the validation of all three claimed effects of meta-labeling, ultimately leading to improved model performance and profitability metric.

As Joubert (2022) uses ECDF to transform the output of the secondary model into a position size, there is room for improvement in the process of position sizing. Meyer, Barziy, et al. (n.d.) explores the calibration of the secondary model's output to align it closer to the true posterior probability. The study compares different position sizing algorithms, including five established methods and a novel approach, assessing their strategy metrics such as the maximum drawdown and Sharpe ratio. The findings highlight that calibration enhances returns. Furthermore, if the goal is to maximize returns, an all-or-nothing position sizing method may be suitable, whereas the ECDF method may be preferable for risk minimization.

The flexibility offered by the decoupling of side prediction and size prediction in meta-labeling architectures is explored in the work of Meyer, Joubert, et al. (n.d.). The authors classify five meta-labeling architectures and discuss their characteristics. Notably, the Discrete Long and Short Meta-labeling Architecture employs different secondary models for long and short positions, while the Conditional Meta-labeling Architecture incorporates different secondary models based on specific conditions of input features. Additionally, Thumm et al. (2022) examines the application of ensemble methods to meta-labeling and finds that ensembles are particularly beneficial when the data consist of multiple regimes and exhibit nonlinearity.

In conclusion, meta-labeling proves to be an effective approach in trading and investment strategies. Although meta-labeling is a relatively new concept, there is sufficient research to form the basis of subsequent studies and to provide a framework. A possible drawback of the existing literature is that there are not many articles using real data to analyse the benefits of meta-labeling.

2.3 Summary

In conclusion, deep learning, particularly LSTM, has exhibited strong capabilities in modeling sequential data, finding numerous applications in financial markets. Similarly, meta-labeling, though relatively novel, has shown promise in quantitative investing. Nevertheless, current literature on stock price forecasting lacks sufficient exploration of strategy profitability and primarily concentrates on daily frequency data. Moreover, research on meta-labeling lacks validation using real-world data. To address these gaps, this paper's innovation lies in utilizing LSTM for meta-labeling in quantitative trading, specifically with minute-level stock data. This approach aims to overcome the limitations of prior studies and contribute to a more robust and practical understanding of quantitative trading strategies.

Chapter 3

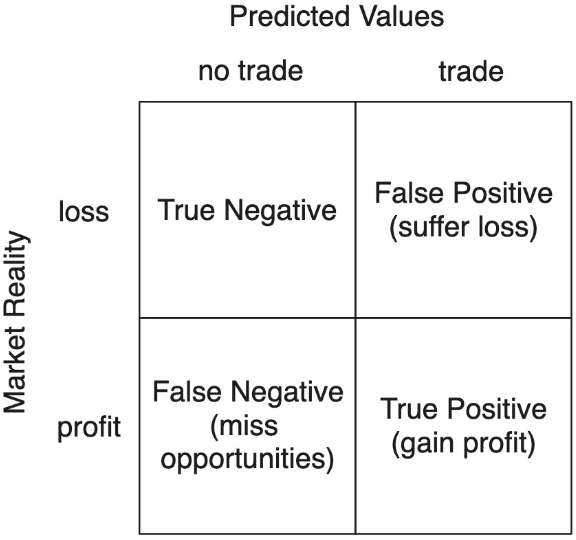
Meta-labeling

3.1 Introduction to Meta-labeling

Meta-labeling involves applying a secondary machine learning model on top of a primary classifier to assess the quality or reliability of the primary model. By combining the secondary and primary models, By combining the two parts, it is possible to filter out false positives, which in turn improves the strategy's risk-return metrics. This section presents the motivation for applying meta-labeling to quantitative trading, its potential to enhance quantitative trading models, and the underlying components.

3.2 Classifier's Performance Metrics vs. Strategy's Risk-return Metrics

To begin, let's focus on the performance metrics of the classifier. Assuming we already have a primary classifier capable of providing signals for "trade" versus "no trade", we can construct a confusion matrix based on actual market conditions (Figure: 3.1). Four outcomes can arise from this matrix. The first outcome occurs when a trade is predicted, and it aligns with the market reality, resulting in a profitable trade (true positive). The second outcome arises when a trade is predicted, but it goes against the market reality, resulting in a loss (false positive). The third and fourth outcomes correspond to cases where the prediction calls for no trade, encompassing both true negatives and false negatives.



		Predicted Values	
		no trade	trade
Market Reality	loss	True Negative	False Positive (suffer loss)
	profit	False Negative (miss opportunities)	True Positive (gain profit)

Figure 3.1: Trading Confusion Matrix

To evaluate the effectiveness of the classification model, precision, recall, and F1-score serve as common performance metrics. Precision measures the proportion of correctly predicted positive instances (profitable trades) out of all instances predicted as positive, indicating the model's ability to avoid false positives (losing trades). Recall measures the proportion of correctly predicted positive instances (profitable trades) out of all actual positive instances (potential opportunities), reflecting the model's ability to avoid false negatives or capture profitable trades. F1 score is a combination of both.

$$Precision = \frac{\#TruePositives}{\#TruePositives + \#FalsePositives} = \frac{\#ProfitableTrades}{\#ProfitableTrades + \#LosingTrades},$$

$$Recall = \frac{\#TruePositives}{\#TruePositives + \#FalseNegatives} = \frac{\#ProfitableTrades}{\#ProfitableTrades + \#MissedOpportunities},$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}.$$

Next, let's shift our focus to the risk-return metrics of the trading strategy. An effective trading strategy aims to maximize returns while minimizing risks. The risk-return metrics commonly used to assess the quality of a strategy include annualized return, annualized standard deviation, and the Sharpe ratio (the ratio of return to standard deviation).

The question then arises: how can the model's performance metrics, such as precision and recall, be related to the risk-return metrics of the strategy? The answer lies in recognizing that a high Sharpe ratio strategy is the result of a combination of high recall and high precision. This connection between recall, precision, and the Sharpe ratio can be illustrated through the following example.

This example is derived from Lopez de Prado (2018). Consider a strategy that generates n independent and identically distributed (IID) bets per year, with each bet having a probability p of generating a profit of π and a probability $1-p$ of incurring a loss of $-\pi$. If we define X_i as the outcome, with $P[X_i = \pi] = p$ and $P[X_i = -\pi] = 1 - p$, then p represents the precision of a binary classifier. The annualized Sharpe ratio can be calculated as follows:

$$SharpeRatio = \frac{Return}{StandardDeviation} = \frac{nE[X_i]}{\sqrt{nV[X_i]}} = \frac{2p-1}{2\sqrt{p(1-p)}}\sqrt{n}$$

In the case of symmetric profits and losses, the risk-adjusted performance of the strategy is positively correlated with precision and the number of investment opportunities (recall). These conclusions hold true for asymmetric payouts as well.

In conclusion, to achieve a well-performing, risk-adjusted strategy, both high-recall and high-precision classifiers are necessary for predicting whether to trade or not.

3.3 Motivation for Meta-labeling

The problem, however, is that it can be challenging to train a classifier that achieves both high recall and high precision due to the inherent trade-off between these two metrics. Increasing recall involves capturing a broader range of positive instances, but it can lead to more false positives and lower precision. Conversely, maximizing precision requires being more cautious in classifying instances, potentially resulting in missed positive instances and lower recall.

One motivation for employing meta-labeling is to effectively use this trade-off between recall and precision to achieve better risk-return performance. The appropriate balance between these metrics depends on the specific characteristics of the investment strategy. For example, in high-frequency trading, precision may be prioritized due to the limited profit per trade and the abundance of trading opportunities. Conversely, in medium to long-term momentum strategies, recall may be more crucial given the potential profitability of individual trades.

Furthermore, meta-labeling offers additional motivations stemming from its architecture, which involves separating side prediction from size prediction. This separation brings several benefits, including improved interpretability, prevention of overfitting, and enhanced flexibility. In the subsequent sections, we will discuss the framework of meta-labeling and elucidate how it contributes to these aforementioned advantages.

3.4 Primary Model: Predicting the Side

The meta-labeling framework consists of a primary model and a secondary model. The primary model is responsible for predicting the side of a profitable trade, indicating whether to buy, sell, or close the position. In Figure 3.2, the upper part illustrates the three possible labels: -1 for sell, 1 for buy, and 0 for no trade. This primary model represents a generic and commonly used approach for side prediction in trading.

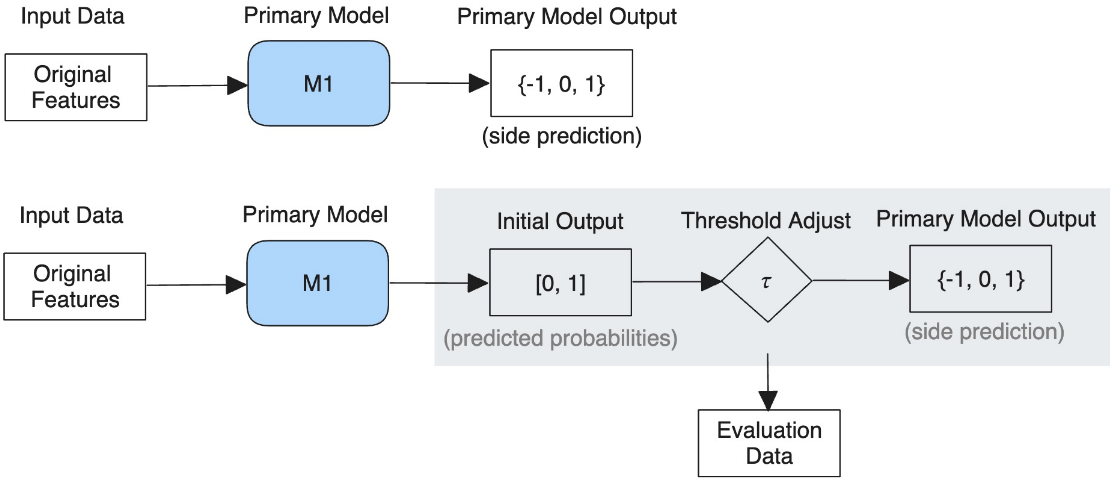


Figure 3.2: Two Types of Primary Model (Adapted from Joubert 2022)

The primary model can take various forms, encompassing discretionary or quantitative strategies, rule-based or model-based approaches, and traditional methods or machine learning algorithms. One notable benefit of meta-labeling is its interpretability. By incorporating a machine learning layer on top of any form of strategy, meta-labeling has the potentially to address the concern of machine learning strategies being criticized as ”black boxes.” However, in this study we use LSTM as the primary model and therefore does not reflect this benefit.

However, when the primary model is a machine learning model, there are two additional benefits, as depicted in the lower part of Figure 3.2. Many machine learning classifiers output probabilities, [0, 1], instead of direct class predictions. These probabilities are then subjected to a thresholding operation to obtain class predictions. For example, if the initial output probability is higher than a threshold τ_1, it is transformed into a buy signal, if it is lower than τ_2, it becomes a sell signal, and otherwise, it is transformed into a “no trade” signal (0).

The first benefit of machine learning models is that the trade-off between recall and precision can be adjusted through the threshold values. If two thresholds τ_1 and τ_2 overlaps, where all signals are either buy or sell, recall is 1 because no trades are missed, but precision is typically low. By setting a higher gap between two thresholds, the frequency of trades is reduced, leading to lower recall but potentially higher precision.

The second benefit is the ability to utilize predicted probabilities to compute evaluation metrics for the primary model, such as rolling cross-entropy or rolling accuracy. These metrics can serve as inputs for the secondary model, allowing for performance evaluation of the primary model over time.

3.5 Secondary Model: Predicting the Size

The secondary model is a key component of the meta-labeling framework. It receives more input data than the primary model and produces a meta-label indicating whether the output of the primary model is correct. Specifically, the secondary model assesses whether the predictions made by the primary model will result in a profit or a loss. This assessment allows for adjustments to the size of the actual position, so it is said that the secondary model predicts the size of positions.

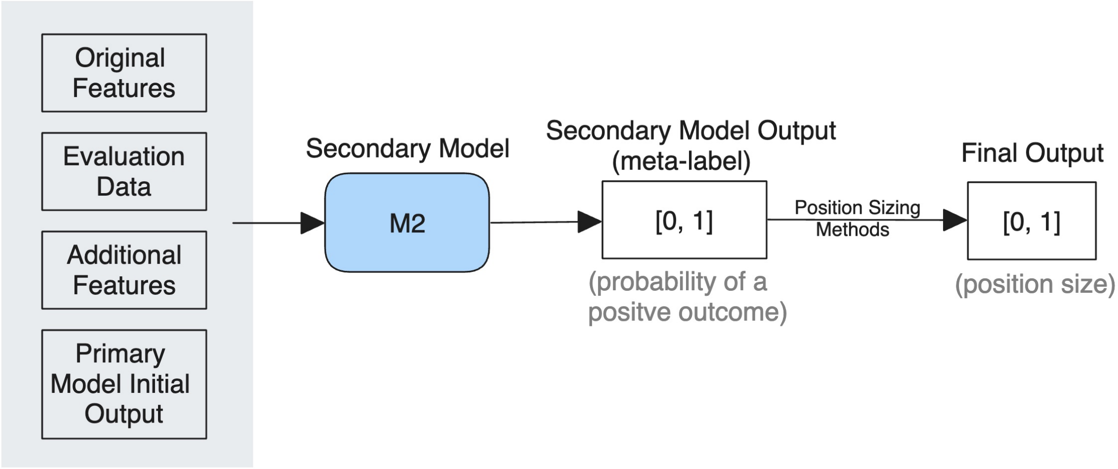


Figure 3.3: Secondary Model (Adapted from Joubert 2022)

The initial output of the secondary model is a value between 0 and 1, representing the confidence level of the primary model. Various position sizing methods can be applied based on this output. For instance, an all-or-nothing strategy invests a full unit of position when the secondary model's probability of a positive outcome exceeds a certain threshold, while not investing at all when it falls below the threshold. A more sophisticated position sizing method allocates more capital to high-confidence trading signals. The higher the probability of true positive (profitable trade), the greater the investment amount. Approaches such as using the empirical cumulative distribution function can be employed for this purpose.

The secondary model takes four types of data as inputs. Firstly, it receives the output of the primary model. Secondly, it incorporates the original features used by the primary model. As the secondary model may employ a different model type, it has the potential to uncover distinct information, providing an informational advantage. The third input is evaluation data, which assesses the recent performance of the primary model. This evaluation helps determine if the primary model should be deactivated. Lastly, additional features are utilized to assess the suitability of the primary model for the current market conditions. These additional features can include macro indicators (e.g., inflation rate, yield curve slope) that relate to the market environment. Other options encompass price moments (e.g., skewness, kurtosis), which reflect shifts in the statistical distribution of an asset. Such shifts may render the primary model inappropriate for the current market.

In summary, the secondary model uses more features and different model, which helps to screen out false negatives. By this, informational advantage is gained and overfitting is avoided by decoupling the prediction of side and size. Additionally, multiple position sizing methods can be implemented to suit specific investment needs. Finally, more complex frameworks can be applied, such as having separate secondary models for buy and sell signals.

Chapter 4

Data and Features

4.1 Data Collection: Sources and Timeframe

The selected dataset consists of price and volume data for Apple's stock at a minute-by-minute granularity. The data spans from October 2013 to December 2019, comprising a total of 610,980 entries. The decision to use minute-level data is driven by its suitability for the technical indicator investment strategy, which will be discussed further. Apple's stock is chosen as it is representative and highly liquid, facilitating a realistic implementation of the investment strategy. The chosen time period takes into consideration data availability and the impact of COVID-19. Data earlier than COVID-19 are used to avoid complicating the experiment. Additionally, price and volume data for the S&P 500 is used as a feature.

4.2 Features for the Primary Model: Technical Indicators

For the primary model, which is also the base strategy, we choose technical indicators as input features. Technical indicators are mathematical transformation of the volume and price data, which can reveal specific patterns or trends in prices. The remainder of this section discusses the motivations of using technical indicators and the selection of technical indicators.

First, technical indicators are valuable for forecasting price changes. Although the Market Effectiveness Hypothesis argues that all relevant information is already priced into an asset, raw price and volume data are often cluttered with noise. Technical indicators help filter out this noise by highlighting essential data, thus aiding models like LSTM in pattern recognition. For example, moving averages can smooth out price volatility, and the relative strength index (RSI) can indicate overbought or oversold states.

Second, technical indicators are convenient for data acquisition and model construction. They offer data at minute-level frequencies, which align with our goal of predicting minute-level stock movements. In contrast, fundamental data often come in daily frequencies, making them less suitable for minute-level predictions and complicating the modeling process.

In summary, technical indicators offer a dual advantage: they are effective in predicting price movements and provide data at the necessary minute-level frequency.

For indicator selection and calculations, we employ the TA-lib software library. It categorizes indicators into five groups—Overlap Studies, Momentum Indicators, Volume Indicators, Volatility Indicators, and Cycle Indicators—making technical analysis more streamlined.

To ensure low correlation between features, we carefully selected one to four technical indicators from each category based on their popularity and relevance (see Table 4.1). The formulas for the chosen technical indicators can be found in Appendix A.

Table 4.1: Selected Technical Indicators as Input Features for the Primary Model

Category	**Feature**	**Description**
Price/Volume	Open	Opening price in the one-minute time frame
Price/Volume	High	Highest price in the one-minute time frame
Price/Volume	Low	Lowest price in the one-minute time frame
Price/Volume	Close	Closing price in the one-minute time frame
Price/Volume	Volume	Total trading volume in the one-minute time frame
Price/Volume	Return	Percentage change of price in the one-minute time frame
Overlap Studies	EMA(5 Min)	Exponential moving average of close price (5 minutes)
Overlap Studies	EMA(10 Min)	Exponential moving average of close price (10 minutes)
Overlap Studies	EMA(30 Min)	Exponential moving average of close price (30 minutes)
Overlap Studies	SMA(5 Min)	Simple moving average of close price (5 minutes)
Overlap Studies	SMA(10 Min)	Simple moving average of close price (10 minutes)
Overlap Studies	SMA(30 Min)	Simple moving average of close price (30 minutes)
Momentum Indicators	CCI	Commodity Channel Index
Momentum Indicators	MACD	Moving Average Convergence/Divergence
Momentum Indicators	STOCHRSI	Stochastic Relative Strength Index
Momentum Indicators	WILLR	Williams' %R
Volume Indicators	AD	Chaikin Accumulation/Distribution Line
Volume Indicators	ADOSC	Chaikin Accumulation/Distribution Oscillator
Volatility Indicators	ATR	Average True Range
Cycle Indicators	HT_INPHASE	Inphase phasor components of Hilbert transform
Cycle Indicators	HT_QUADRATURE	Quadrature phasor components of Hilbert transform
Statistical Function	VAR(30 Min)	Variance of the closing price (30 minutes)

By using TA-lib and carefully selecting technical indicators with low correlation, we can create a robust and efficient primary model to predict stock price movements.

4.3 Features for Secondary Model

As mentioned in earlier chapter, the secondary model encompasses a more extensive range of inputs compared to the primary model. It incorporates four distinct types of features, namely the primary model's original inputs and outputs, evaluation data, and additional features. This section will discuss each of these feature types in detail.

4.3.1 Incorporating Primary Model's Prediction

The first type of feature utilized by the secondary model is the primary model's prediction of a stock's upward or downward movement. This is a natural inclusion, as the secondary model's purpose is to assess the credibility of raw trading signals generated by the primary model.

4.3.2 Utilizing Primary Model's Original Inputs

The second type of feature integrated into the secondary model comprises the primary model's original inputs, specifically the technical indicators. The rationale behind this is that the secondary model may be able to extract further relevant information from these original features. By incorporating them, the secondary model can potentially utilise patterns and relationships in technical indicators to judge the credibility of the primary model

4.3.3 Leveraging Evaluation Data

The third type of features utilized by the secondary model consists of evaluation data, specifically focusing on three key performance metrics calculated from recent data: accuracy, precision, and cross-entropy, derived from the primary model's performance over the last thirty minutes. The primary model generates stock direction forecasts and raw trading signals at time t, and the secondary model assesses the credibility of these signals based on the primary model's recent performance during the time period $t-30$ to $t-1$.

Accuracy, precision and cross-entropy, commonly used in binary classification problems, provide valuable insights into the quality of the primary model's predictions. A recent decline in accuracy and precision accompanied by higher cross-entropy indicates potential unreliability in the primary model's recent forecasts. This integration of evaluation data enables the secondary model to make informed judgments and enhances the effectiveness of the meta-labeling approach in quantitative trading.

4.3.4 Additional Features for Secondary Model

The fourth type of additional features used by the secondary model aims to reflect the overall market conditions and the nature of stock characteristics. This allows us to observe the adaptability of the underlying strategy to changing market dynamics. For the market conditions, S&P 500's minute closing prices and returns are included. Significant changes in these values may indicate shifts in the overall market sentiment, which could impact the performance of the primary model. For the distributional characteristics of the stock prices, the skewness and kurtosis of the stock traded are included. These statistical measures reflect the statistical distribution of the stock prices. Drastic changes in these moments may indicate a failure of the primary strategy to adapt to the changing dynamics of the traded stock.

Overall, the incorporation of these diverse feature types equips the secondary model with a rich set of inputs, enabling it to make informed judgments about the reliability of raw trading signals.

4.4 Data Preprocessing

With features for both primary and secondary models in place, the next step is data preprocessing. We utilize min-max scaling to normalize all features, converting them to a common range, usually between 0 and 1. The formula for min-max scaling is as follows:

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{4.1}$$

This scaling technique benefits deep learning in several ways. First, it ensures no single feature disproportionately influences model training, aiding in faster and more stable convergence. Second, it pairs well with activation functions like sigmoid or tanh, commonly found in LSTMs, mitigating issues like vanishing or exploding gradients. This makes the learning process more efficient and numerically stable.

Lastly, to avoid information leakage, we apply min-max scaling only after partitioning the data into training and test sets. This maintains the integrity of the data by ensuring that future information doesn't influence past data points.

Chapter 5

LSTM

5.1 Technical Preliminaries: LSTM Networks

5.1.1 Recurrent Neural Network

Recurrent Neural Networks (RNNs) specialize in handling sequential data like time series, differing from feedforward networks by having looped connections. This structure enables them to capture temporal relationships and utilize past information in current computations.

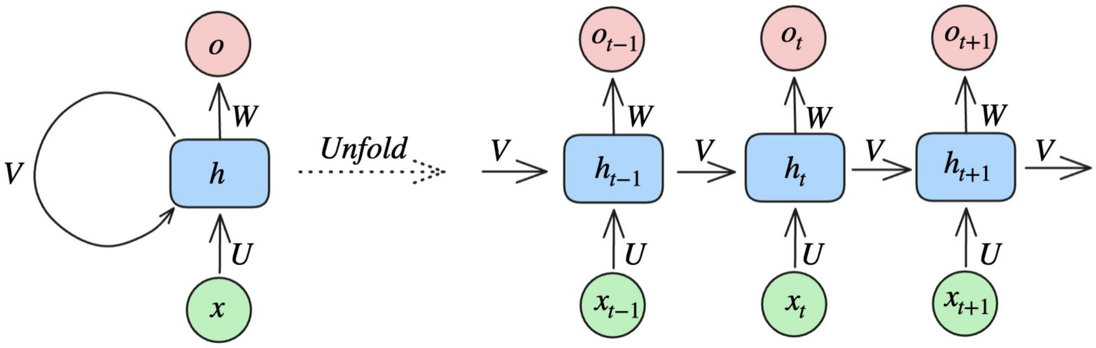


Figure 5.1: RNN

In Figure 5.1, the recurrent neuron receives an input vector at each time step and generates both an output and a hidden state. This hidden state acts as the network's memory, storing information from past inputs. Consequently, the output at each time step is influenced by both the current input and this stored memory.

During training, RNNs use a technique called backpropagation through time (BPTT) to learn from sequences. BPTT adapts the standard backpropagation method for recurrent connections by unfolding them over time. For each time step, a loss function quantifies the difference between predicted outputs and actual values. Gradients of this loss function concerning the weights W,V,U are then computed. These gradients guide the weight adjustments, optimizing the model using algorithms like stochastic gradient descent (SGD). This process iterates multiple times to minimize the loss function.

5.1.2 Long Short-Term Memory

LSTM, a specialized form of RNN, excels at managing long-term sequence dependencies thanks to its memory cells and gating mechanisms.

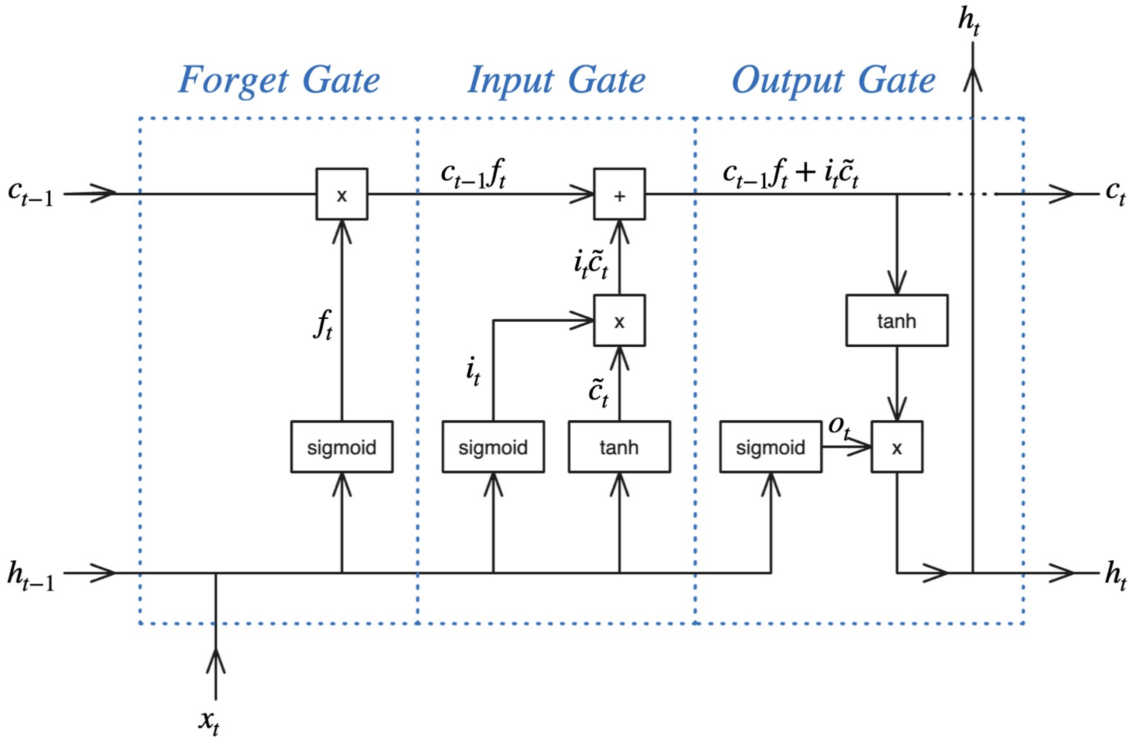


Figure 5.2: LSTM Cell

In Figure 5.2, an LSTM cell is shown to have input, forget, and output gates. These gates collectively handle information processing and retention over extended periods.

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \tag{5.1}$$

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \tag{5.2}$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \tag{5.3}$$

$$\tilde{c}_t = tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \tag{5.4}$$

$$c_t = c_{t-1} * f_t + i_t * \tilde{c}_t \tag{5.5}$$

$$h_t = o_t * tanh(c_t) \tag{5.6}$$

The forget gate determines the degree to which the previous memory cell state (c_{t-1}) is either forgotten or retained. It takes the current input (x_t) and the preceding hidden state (h_{t-1}) as inputs, which are then processed through a sigmoid activation function. The output, a forget gate vector (f_t in Equation 5.1), ranges from 0 to 1. This vector modulates the amount of the previous memory cell state (c_{t-1}) to retain ($c_{t-1}f_t$). A value close to 0 instructs the network to forget that component, whereas a value near 1 suggests retention.

The input gate manages the incorporation of new information into the memory cell. It utilizes the current input (x_t), and the prior cell and hidden states (c_{t-1} and h_{t-1}) as inputs. Through a sigmoid activation function, it produces a gate vector (i_t in Equation 5.2) ranging from 0 to 1 to decide the amount of new information to store. Additionally, a tanh activation function generates a cell state candidate (g_t in Equation 5.4). The gate vector (i_t) acts as a filter, determining which parts of g_t are incorporated into the memory cell ($i_t g_t$). A value of 0 for i_t indicates ignoring the corresponding component, whereas a value of 1 suggests full inclusion. This filtered information is combined with the prior memory cell state to update the current cell state (c_t in Equation 5.5).

The output gate regulates the final output from the memory cell. It takes the current input (x_t) and previous hidden state (h_{t-1}) as inputs and employs a sigmoid activation function to produce an output gate vector (o_t in Equation 5.3) ranging from 0 to 1. This vector governs the influence of the current cell state (c_t) on the LSTM cell's final output (h_t in Equation 5.6). A value of 0 for o_t suppresses the output, whereas a value of 1 fully expresses it.

Utilizing these three gates, an LSTM cell is proficient at managing and updating its memory, thus excelling at retaining crucial information over extended sequences. These gates enable the LSTM to selectively manipulate information flow, making it particularly well-suited for long-term dependencies. Although variants of LSTMs exist, their overall performance is generally comparable, as substantiated by extensive research (Greff et al. 2017).

5.1.3 Stacked LSTM

Building upon the fundamental principles of the LSTM unit, stacked LSTM incorporate multiple LSTM layers, thereby endowing the model with increased depth and expressive capacity.

Each layer in stacked LSTM takes the output sequence (h_t) from the layer below as input. This sequential flow carries knowledge through the layers, with each benefiting from the previous one.

For intricate time series with complex patterns and dependencies, a single LSTM layer may struggle to capture higher-level abstractions. Stacked LSTM, however, excels due to its multi-layer design, enabling better modeling of long-range dependencies and hierarchies of features.

5.2 Model Details

This section outlines the LSTM model employed in our research, covering its architecture, dropout layers, and applications.

5.2.1 Usages of LSTM: in Primary and Secondary Model

LSTM's distinct structure captures short and long-term temporal data effectively. We'll apply LSTM to both primary and secondary models for a comprehensive trading strategy benchmark. Our goals include verifying LSTM's time series prediction, strategy construction feasibility, and meta-labelling advantages. We'll discuss predictor variables (labels) for primary and secondary models.

In meta-labelling, primary and secondary models have separate target variables. Primary model predicts stock movement (up/down) in the next minute (Equation 5.7). These predictions form the raw trading signals, which are then transformed into buy, sell, or hold signals using a threshold mechanism, discussed in a later chapter.

The secondary model predicts if primary's signals align with market behavior. It binary predicts primary's profitability (Equation 5.8). If buy signal aligns with rising close or sell signal with falling close, profit happens. The secondary model's predictive class is the meta-label.

$$y_{1,t} = \begin{cases} 1 & \text{if the closing price of this minute is higher than the previous minute} \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

$$y_{2,t} = \begin{cases} 1 & \text{if the raw trading signal results in a profit} \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

Both primary and secondary models are binary classification tasks. The LSTM's output layer uses a sigmoid activation function, yielding probabilities for the positive classes (”price increase” in primary and ”credible” in secondary).

To sum up, the primary model predicts short-term price shifts for raw trading signals, and the secondary model assesses these signals' ability to yield profits.

5.2.2 Model Architecture

In this section, we highlight the benefits of stacked LSTM's multi-layer structure for intricate time series data. Selecting the right layer setup is crucial, as overly simple designs may underfit, while overly complex ones may overfit and incur high computation.

For minute-level stock prediction, we compare single-layer LSTM with two-layer LSTM architectures. Generally, the two-layer LSTM performs better across various hyperparameter combinations.

Hence, we employ the two-layer LSTM for both primary and secondary models. It strikes a balance between efficiency and performance for our objectives. Note that a more rigorous approach using cross-validation could provide a stronger architecture assessment, but our resource limitations prevent its use here.

5.3 Model Training and Hyperparameter Tuning

This section will discuss model training and decision of hyperparameter.

5.3.1 Training

Training is the process of teaching a model to learn from a labeled dataset so that it can generalize and make accurate predictions on new, unseen data.

A key to training is the loss function, and since this study attempts to solve a binary classification problem, cross entropy is a natural loss function choice. It is well-suited for tasks with two classes, measuring the dissimilarity between predicted probabilities and true binary labels.

The LSTM model employs the "Adaptive Moment Estimation" (Adam) algorithm for optimization. Adam adapts learning rates, manages momentum, and suits complex LSTM tasks while maintaining low memory usage.

Regarding the division of the training and test sets, we use the first 80% of the data as the training set and the second 20% as the test set to ensure that future information from the test set is not captured in the training.

In addition, to prevent overfitting, we used the regularisation technique of dropout. During training, dropout randomly deactivates (sets to zero) a fraction of the neurons or units in the layer. By doing this, dropout introduces noise and variability into the model, which encourages robustness and prevents the model from relying too heavily on specific neurons or memorizing noise in the training data.

5.3.2 Hyperparameter Tuning

Model parameters that cannot be learnt during training are hyperparameters. Hyperparameters are related to the settings of the model and are an important part of the model performance. Key hyperparameters for this study encompass dropout rate in the dropout layer, time steps per input sequence, and hidden layer neuron count in the LSTM layer.

The tuning of hyperparameters is performed through cross-validation. It resamples data to assess model generalization and performance across various configurations. By iteratively training and evaluating on data folds, cross-validation offers an unbiased performance assessment.

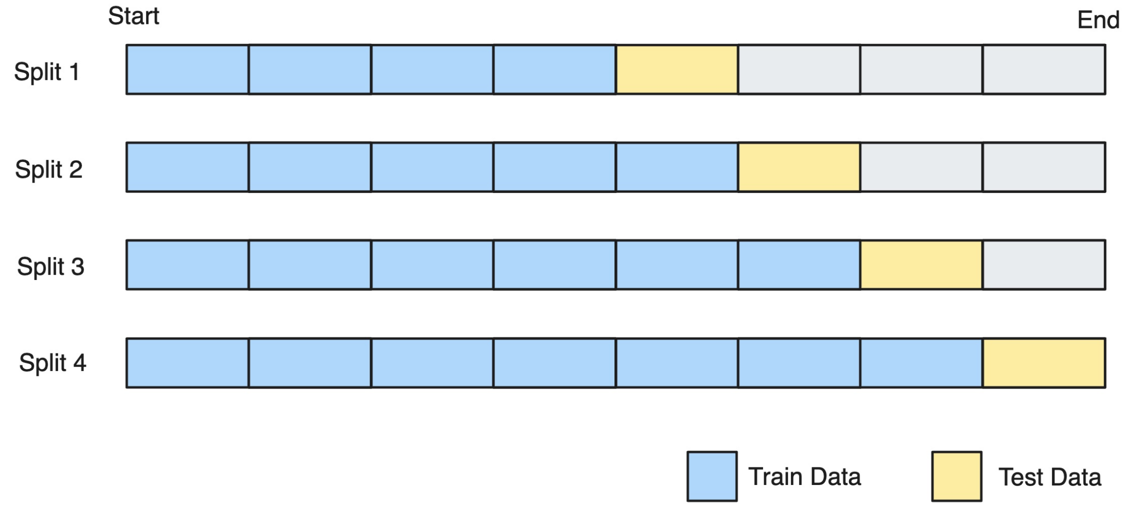


Figure 5.3: Expanding Window Cross-Validation

Time series data demands maintaining chronological order, unlike traditional k-fold cross-validation. Expanding window cross-validation is adopted, progressively widening the training window over time. As illustrated in Figure 5.3, each step includes all data up to a point, validating the subsequent data points. This approach captures temporal dependencies effectively.

For targeted hyperparameter searches, a grid search is employed. Dropout rates 0, 0.1, and 0.2, time steps 5, 10, and 20, and hidden layer neuron counts 5, 10, 20, and 30 are tested. All 36 combinations are evaluated based on average cross-entropy performance on the validation set. Optimal hyperparameters for the primary model are dropout rate 0.2, time steps 10, and hidden layer neurons 20.

Chapter 6

Trading and Backtesting

In the previous chapter, we discussed the LSTM model's ability to predict stock price movements. This section will focus on how the primary model translates these predictions into trading signals, the role of the secondary model in the final trading decision, and the essential process of backtesting.

6.1 Threshold Mechanism for Raw Trading Signals

In the context of trading, the basic classifications of "long", "short", and "close" indicate different trading actions. "Long" positions involve buying an asset with the expectation of its value increasing over time, while "short" positions involve selling an asset with the anticipation of its value decreasing. Closing a position is the act of reversing a trade, either to secure profits or limit losses. Additionally, when no positions are held, the "close" signal signifies no action, representing a default state.

With the LSTM model's probability prediction for a stock's upward movement, we need a method to generate trading signals. In this study, a threshold-based approach is used to derive trading signals. If the predicted probability of a stock going up exceeds a certain threshold, a signal to open a long position is triggered. If the probability falls below another threshold, a signal to open a short position is triggered. Otherwise, a signal to close the position is activated.

The threshold mechanism is defined as follows:

$$\begin{cases} \text{long, if } y_{1,t} > 0.5 + \tau \\ \text{short, if } y_{1,t} < 0.5 - \tau \\ \text{close, otherwise} \end{cases} \tag{6.1}$$

Here, $y_{1,t}$, which is defined in Equation 5.7, represents the prediction of the primary model. And τ is a parameter that determines the two thresholds. By adjusting τ, traders can customize their trading strategy based on their risk tolerance and market outlook. A larger τ leads to more conservative trading with fewer but potentially higher-probability trades. Conversely, a smaller τ results in more aggressive trading.

It is important to note that there is no universally optimal value for τ. The choice of τ is influenced by an investor's specific risk preference. In subsequent chapters, we will explore the effect of meta-labeling at different values of τ through empirical tests.

6.2 Generation of the Final Trading Strategy

In this section, we explore two methods of generating the final trading positions based on the predictions of the primary and secondary models. As discussed earlier in the introduction to meta-labeling, the secondary model assesses the credibility of the primary model's signals. The final trading signal is then determined by considering both models' outputs to enhance the trading strategy's performance.

As discussed earlier in the introduction to meta-labeling, the secondary model predicts the credibility of the primary model's signals. The final trading signal is determined by the combined output of both the primary and secondary models. To better demonstrate the effectiveness of meta-labelling, we will employ two methods of generating final trading positions by referring to Joubert (2022).

6.2.1 Method 1: All-or-Nothing

$$\text{trading unit} = \begin{cases} 1, \text{ if } y_{2,t} > 0.5 \\ \\ 0, \text{ if } y_{2,t} \leq 0.5 \end{cases} \tag{6.2}$$

As shown in Equation 6.2, the first method involves an all-or-nothing approach, where the final trading unit is determined based on the credibility of the secondary model's output ($y_{2,t}$, defined in Equation 5.8) with respect to a threshold value of 0.5. The trading unit is set to 1 if $y_{2,t}$ is greater than 0.5, indicating high credibility in the primary model's signals. This means that we fully follow the primary model's signals and trade a complete unit. Conversely, if $y_{2,t}$ is less than or equal to 0.5, the trading unit is set to 0, indicating that we abstain from following the primary model's signals and do not trade at all regardless of the raw trading signals.

6.2.2 Method 2: ECDF

$$\text{trading unit} = ECDF(y_{2,t}) \tag{6.3}$$

As shown in Equation 6.2, the second method utilizes an empirical cumulative distribution function

(ECDF) to determine the final trading unit. The output of the secondary model ($y_{2,t}$) is passed through the pre-trained ECDF, resulting in a continuous value between 0 and 1. This approach offers a more granular response to the credibility of the primary model's signals. A higher value from the ECDF corresponds to higher credibility, leading to a higher number of final trades. Conversely, a lower value from the ECDF indicates lower credibility, resulting in fewer trades or no trades at all, even if the raw trading signals are present.

Obviously, compared to the first method, the second method is more refined and makes better use of the information advantages that meta-labelling brings.

6.3 Backtesting and Risk-return metrics

In addition to the performance metrics used to evaluate machine learning models, strategy-based metrics such as return and risk are essential for assessing the investability of a trading strategy. Backtesting is a crucial process that generates these strategy-based metrics by evaluating the performance of a trading or investment strategy using historical market data to gauge its effectiveness and potential profitability. Specifically, backtesting produces a curve of net asset value, illustrating how the asset value would have evolved over time, assuming one unit of money was invested at the beginning of the period. In other words, backtesting somewhat reflects what we would have gained if we had actually invested money into this strategy.

To improve efficiency and cut costs, we employ vectorized backtesting. However, transaction costs and commissions aren't considered, affecting real-world results. Despite this, vectorized backtesting effectively highlights profitable strategies and meta-labeling advantages. It offers insights into strategy history, aiding real trading viability assessment.

Using the net asset value (NAV) curve, we calculate key risk-return metrics, including annualized returns, standard deviation, maximum drawdown, Sharpe ratio, and Calmar ratio. Definitions are in Table 6.1.

Table 6.1: Risk-Return Metrics

Metrics	Definition
Annualized Return	The average rate of return on an investment over a specified period, expressed as a percentage per year.
Annualized Standard Deviation	The measure of the volatility of an investment's returns over a specific period, expressed as a percentage per year.
Maximum Drawdown	The largest daily peak-to-trough decline in an investment's value over a specific period, highlighting the worst loss experienced.
Sharpe Ratio	The risk-adjusted performance metric that assesses the excess return of an investment relative to its risk, expressed as the return per unit of risk.
Calmar Ratio	The risk-adjusted performance metric that compares the annualized return to the maximum drawdown, indicating the return generated for each unit of maximum drawdown experienced.

Chapter 7

Experiment and Result

In this section, we will train the model, build the strategy and backtest it to obtain the NAV curve. We will use real data to achieve the goals mentioned in Chapter 1, including verifying the ability of LSTM models to predict stock movements on minute data, and meta-labelling to filter false positives in order to improve the strategy's risk-return metrics.

Experiment 1 is the primary model. It will use Apple's price and volume and 17 technical indicators, use an LSTM model to output a prediction of the next minute's rise or fall, and generate raw trading signals based on this prediction to form the base strategy. Experiment 2 is the secondary model with all-or-nothing trading method. It will predict the credibility of the raw trading signals from experiment 1, using LSTM models and more features. The final strategy will combine the primary and secondary models by all-or-nothing method to generate trading signals. Experiment 3 is the secondary model with ECDF trading method. It utilises the same credibility predictions as Experiment 2 but uses a different method to generate the final trading signal.

7.1 Metrics

To assess the performance of the LSTM model, we employ a range of evaluation metrics suitable for binary classification tasks, which was also mentioned in Section 3.2. The primary metrics used include:

1. **Accuracy**: The ratio of correctly predicted instances to the total number of instances in the dataset, indicating the overall model performance.
2. **Precision**: The proportion of true positive predictions over the total number of predicted positive instances, providing insight into the model's ability to avoid false positives.
3. **Recall**: The ratio of true positive predictions over the total number of actual positive instances, reflecting the model's ability to capture positive instances.

4. **F1-score**: The harmonic mean of precision and recall, offering a balanced performance metric for binary classification tasks.

5. **Cross-Entropy Loss**: The cross-entropy loss function measures the dissimilarity between the predicted probability distribution and the actual binary labels, providing insight into the model's calibration and prediction accuracy. See Equation 7.1.

$$L = -\frac{1}{N}\sum_{i=1}^{N}(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)) \tag{7.1}$$

In addition to machine learning related metrics, we will also use risk-return metrics, including annualised return, annualised standard deviation, maximum drawdown, Sharpe ratio and Calmar ratio, to measure the investability of the strategy.

7.2 Experiment 1: Primary Model

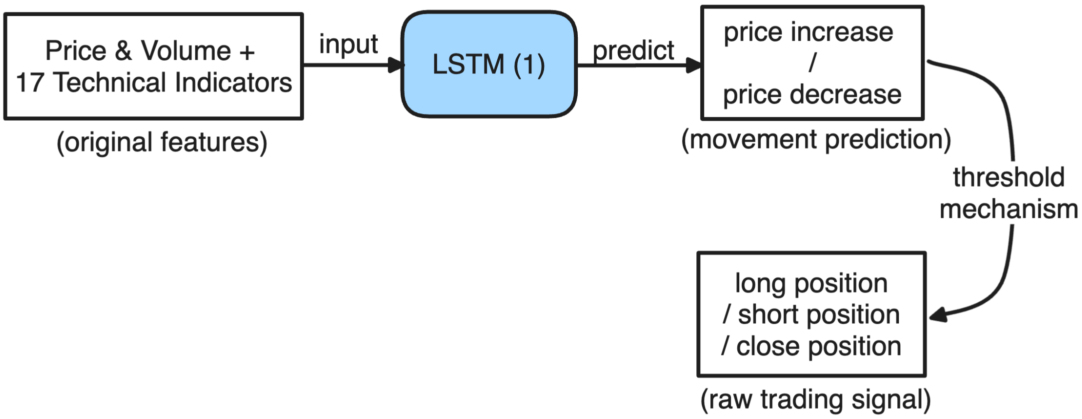


Figure 7.1: Experiment 1

The primary LSTM model, utilizing technical indicators as input, achieved an accuracy of 51.89% and a cross-entropy loss of 0.6923 on the test set. As the dataset does not have imbalanced labels and we equally care about both the increase and decrease of stock prices, precision, recall, and F1-score are not as meaningful in this context.

Despite the primary model showing only a 2% improvement over random guessing in accuracy, its ability to predict stock price movements at the minute level is evident. Considering the high frequency of minutes in the dataset, which provides around 85,000 potential trading opportunities per year, this 2% advantage holds the potential to translate into a profitable trading strategy. To assess the performance of the primary model at different trading thresholds (τ), we construct base strategies accordingly.

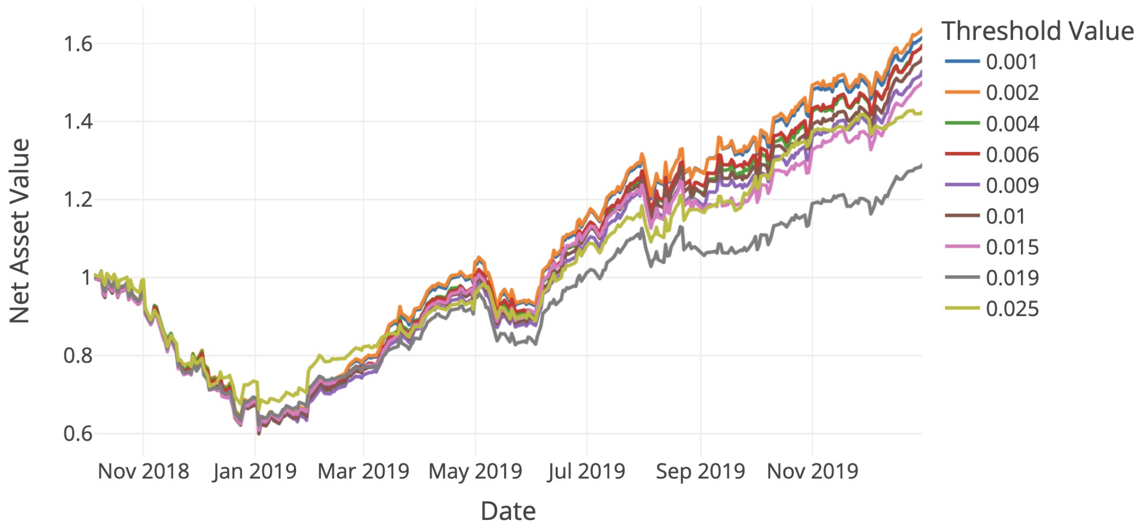
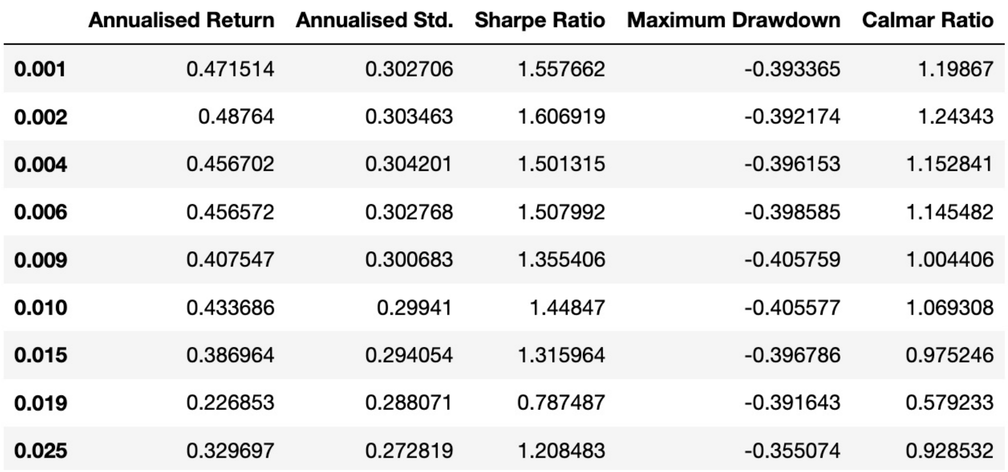


Figure 7.2: Net Asset Value Curve of Primary Model



	Annualised Return	Annualised Std.	Sharpe Ratio	Maximum Drawdown	Calmar Ratio
0.001	0.471514	0.302706	1.557662	-0.393365	1.19867
0.002	0.48764	0.303463	1.606919	-0.392174	1.24343
0.004	0.456702	0.304201	1.501315	-0.396153	1.152841
0.006	0.456572	0.302768	1.507992	-0.398585	1.145482
0.009	0.407547	0.300683	1.355406	-0.405759	1.004406
0.010	0.433686	0.29941	1.44847	-0.405577	1.069308
0.015	0.386964	0.294054	1.315964	-0.396786	0.975246
0.019	0.226853	0.288071	0.787487	-0.391643	0.579233
0.025	0.329697	0.272819	1.208483	-0.355074	0.928532

Figure 7.3: Risk-return Metrics of Primary Model

In Figure 7.2, the net asset value curves illustrate that as the threshold τ varies, the investment gains generally trend in the same direction with minor deviations. Notably, losses are observed between October 2018 and the beginning of 2019, attributed to specific market events such as Apple's new iPhone XR launch and the US-China trade war, resulting in market volatility. During these periods, the model's ability to capture changes in the time series information was temporarily affected, leading to decreased forecasting accuracy. However, after adapting to the market, the LSTM model regained its predictive ability, yielding substantial returns with a 2.5x net asset value growth during 2019.

Figure 7.3 provides insights into the risk-return metrics of the primary model's strategy at various thresholds. The results demonstrate the strategy's positive profitability across all thresholds, indicating the feasibility of building a predictive model and a profitable trading strategy solely based on technical indicators and LSTM models. Moreover, as thresholds increase, both returns and risks tend to decrease.

This aligns with our prior observation, wherein larger thresholds imply a more cautious trading approach, leading to reduced risk. However, it also means that more trading opportunities may be missed, resulting in a final risk-adjusted return that may not be as favorable as a strategy with a smaller threshold.

7.3 Experiment 2: Meta-labeling with All-or-Nothing Trading Method

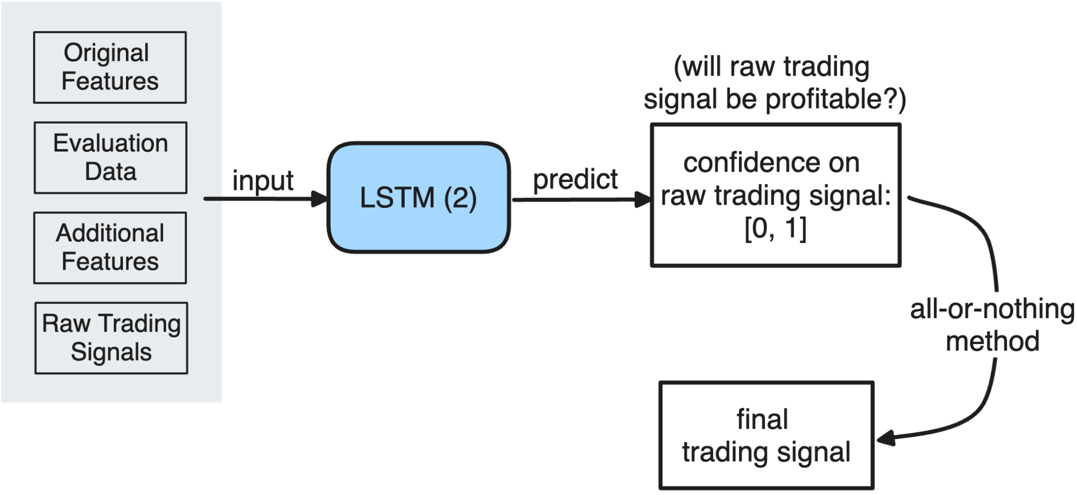
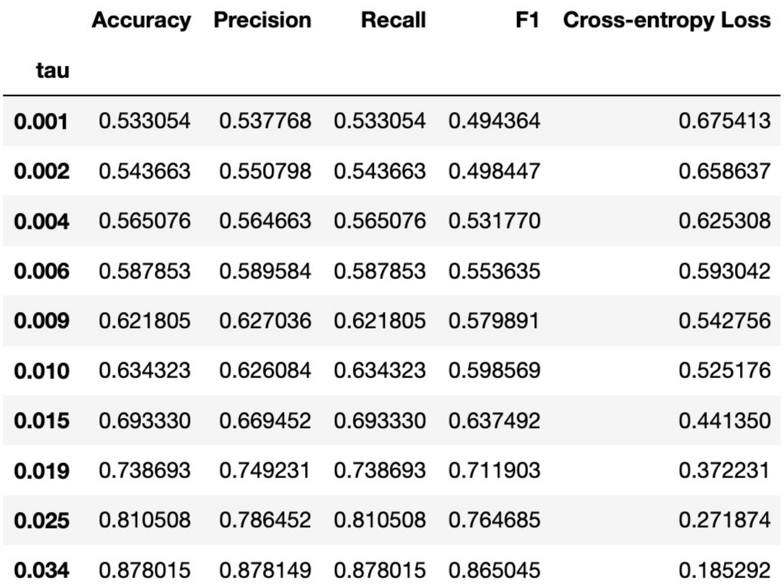


Figure 7.4: Experiment 2

In Experiment 2, we explore the impact of meta-labeling using the all-or-nothing trading method. Here, the secondary model filters out false positives from the raw trading signals, and the predicted credibility from the secondary model is employed to generate the final traded signals. We report the results in three aspects: the LSTM model's performance metrics, metrics related to the trading confusion matrix, and strategy-related metrics.

7.3.1 Performance of LSTM

We begin by analyzing the performance of the LSTM model in the secondary model. Figure 7.5 showcases the metrics for the binary classification problem, where the raw trading signal is evaluated for profitability.



tau	Accuracy	Precision	Recall	F1	Cross-entropy Loss
0.001	0.533054	0.537768	0.533054	0.494364	0.675413
0.002	0.543663	0.550798	0.543663	0.498447	0.658637
0.004	0.565076	0.564663	0.565076	0.531770	0.625308
0.006	0.587853	0.589584	0.587853	0.553635	0.593042
0.009	0.621805	0.627036	0.621805	0.579891	0.542756
0.010	0.634323	0.626084	0.634323	0.598569	0.525176
0.015	0.693330	0.669452	0.693330	0.637492	0.441350
0.019	0.738693	0.749231	0.738693	0.711903	0.372231
0.025	0.810508	0.786452	0.810508	0.764685	0.271874
0.034	0.878015	0.878149	0.878015	0.865045	0.185292

Figure 7.5: Performance Metrics of Secondary LSTM Model

As shown in Figure 7.5, the accuracy of the secondary LSTM model varies between 53.30% and 87.80% as τ changes. This demonstrates that models and additional information layered on top of the primary model can effectively exploit the information in the original features. Interestingly, we observe that the accuracy of the secondary LSTM model improves as τ increases. Two plausible explanations are put forth for this observation.

Firstly, as τ increases, the primary model incurs fewer losses. Although the overall mistake correction accuracy may be lower, the disproportionately large number of correctable mistakes leads to lower accuracy. This does not necessarily imply worse investment results, as the primary model's overall performance is less prone to losses.

Secondly, in the presence of a larger τ, the primary model only trades when the stock price trend is strong. In such strongly trending market environments, the signal-to-noise ratio in the data is higher, enabling the secondary model to better correct mistakes.

7.3.2 Metrics Related to Trading Confusion Matrix

In this subsection, we employ the all-or-nothing method to generate the final trading signal and assess its potential profitability. To evaluate the performance, we refer to the trading confusion matrix presented in Figure 3.1. It is important to note that the terms "precision" and "recall" in this context differ from their usage in the previous subsection (Section 7.1).

In the preceding subsection, the binary classification task involved determining whether the LSTM component of the secondary model correctly predicted adjustments for the raw trading signal. However, in this subsection, the binary classification task is focused on determining whether the combined trading signal from both the primary and secondary models will lead to profitable trades. To maintain clarity, we shall now refer to these metrics as ”trading precision” and ”trading recall” since they directly relate to the trading strategy.

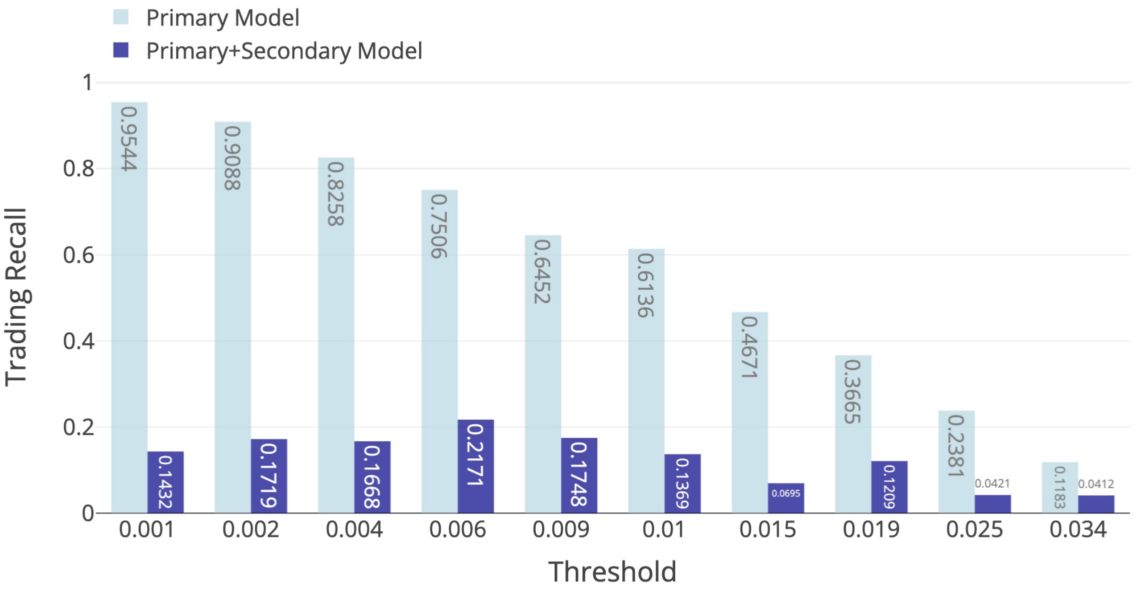


Figure 7.6: Trading Recall

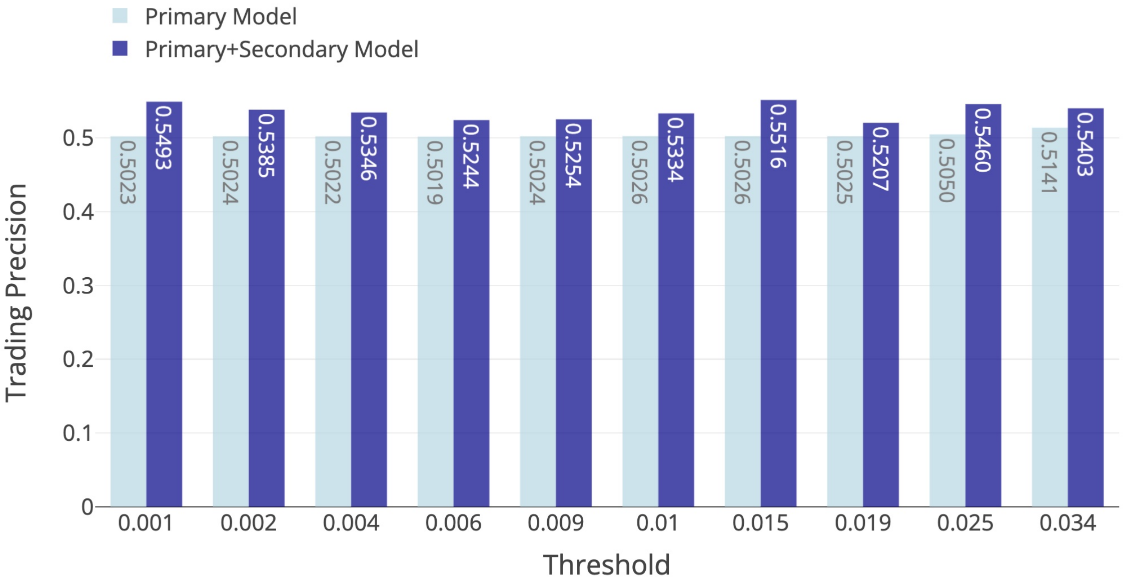


Figure 7.7: Trading Precision

Analyzing Figure 7.6, we observe that when the secondary model rejects certain raw trading signals, the trading recall naturally decreases, resulting in missed opportunities. On the other hand, Figure 7.7 shows a significant increase in trading precision. Even with the simplicity of the all-or-nothing logic, the final strategy’s probability of profit per trade improves. While meta-labelling does appear to manage the trade-off between precision and recall, its precise impact on risk-return metrics requires further investigation in subsequent analyses.

7.3.3 Strategy-related Metrics

In this section, we conduct a backtest on the final trading signal to generate the NAV curve and evaluate various strategy-related metrics.

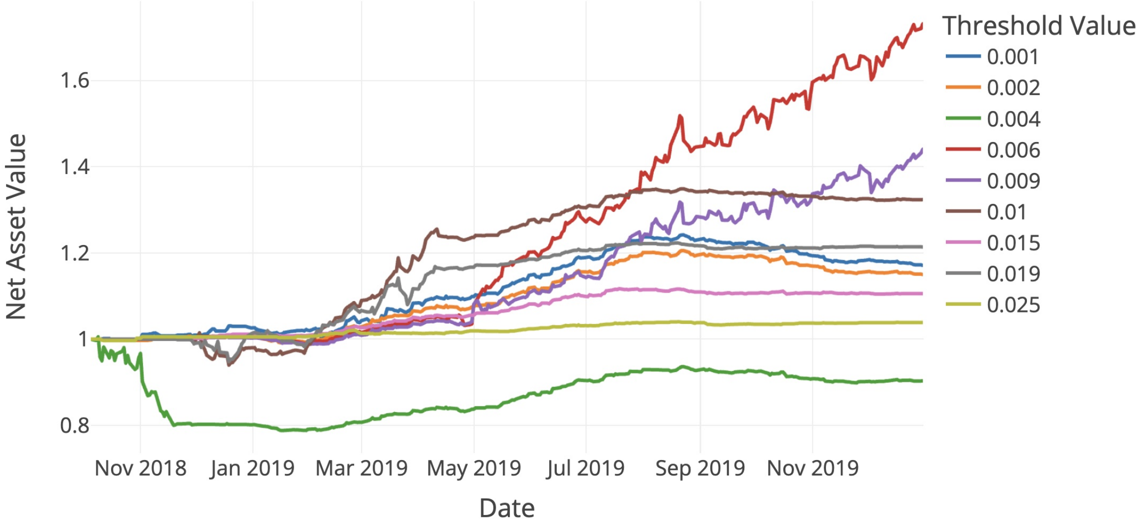
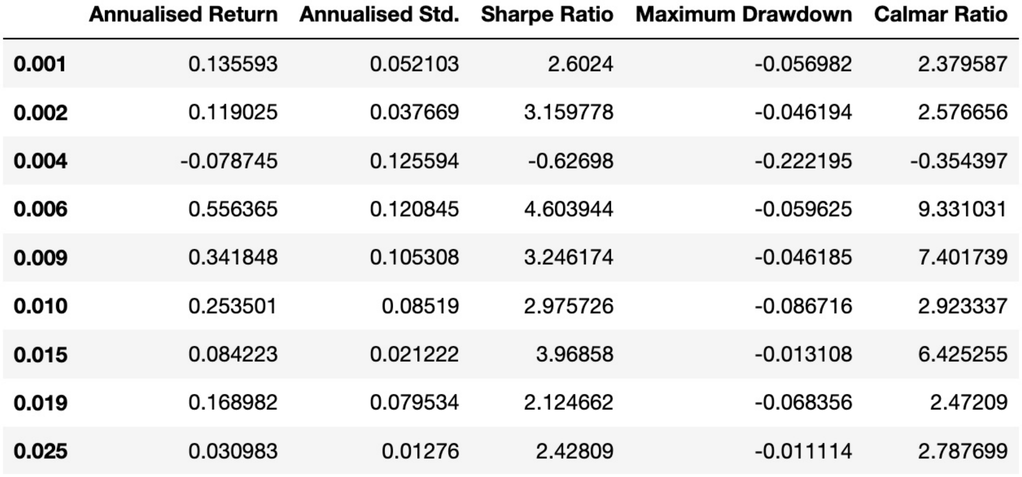


Figure 7.8: Net Asset Value Curve of Meta-labeling with All-or-Nothing Method



	Annualised Return	Annualised Std.	Sharpe Ratio	Maximum Drawdown	Calmar Ratio
0.001	0.135593	0.052103	2.6024	-0.056982	2.379587
0.002	0.119025	0.037669	3.159778	-0.046194	2.576656
0.004	-0.078745	0.125594	-0.62698	-0.222195	-0.354397
0.006	0.556365	0.120845	4.603944	-0.059625	9.331031
0.009	0.341848	0.105308	3.246174	-0.046185	7.401739
0.010	0.253501	0.08519	2.975726	-0.086716	2.923337
0.015	0.084223	0.021222	3.96858	-0.013108	6.425255
0.019	0.168982	0.079534	2.124662	-0.068356	2.47209
0.025	0.030983	0.01276	2.42809	-0.011114	2.787699

Figure 7.9: Risk-return Metrics of Meta-labeling with All-or-Nothing Method

Analyzing Figure 7.8, we observe that the impact of the all-or-nothing method of meta-labeling varies at different thresholds (τ), resulting in distinct trends in the net asset value curves. Examining the investment metrics in Figure 7.9, we find that the annualized return becomes negative when τ equals 0.004. This indicates that, in this particular case, the application of meta-labeling leads to worse investments. However, in all other cases, the inclusion of meta-labeling enhances the Sharpe ratio, thereby improving the overall investment performance.

In conclusion, meta-labeling generally improves risk-return metrics by filtering out false positive investment signals and managing the precision-recall trade-off. Nonetheless, it's important to note that in specific scenarios, the all-or-nothing method of meta-labeling may lead to suboptimal investments. Furthermore, this approach results in significant variations in investment styles across different thresholds, which can pose challenges in selecting appropriate investment styles.

7.4 Experiment 3: Meta-labeling with ECDF Method

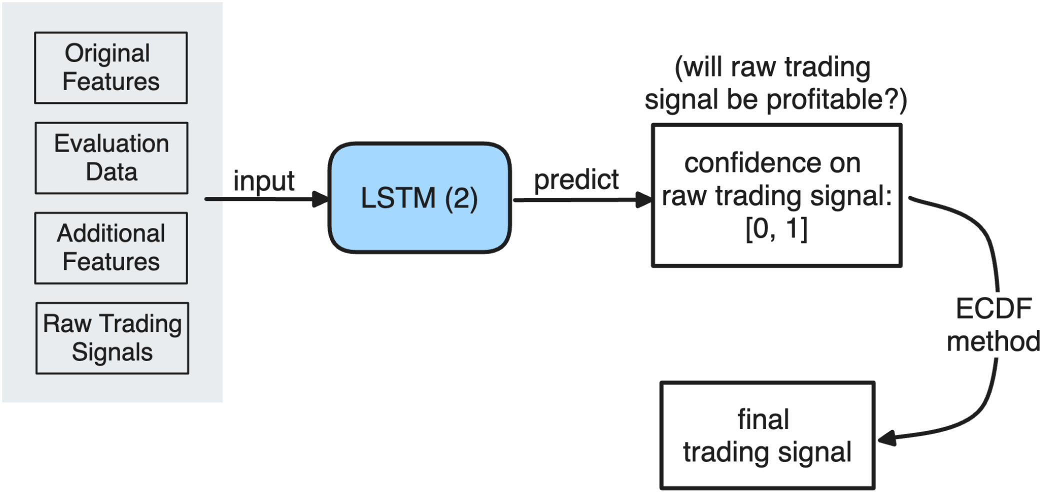


Figure 7.10: Experiment 3

In Experiment 3, we investigate the impact of using the ECDF method to generate final trading signals, leveraging the LSTM outputs in the secondary model to make credibility judgments at a finer granularity.

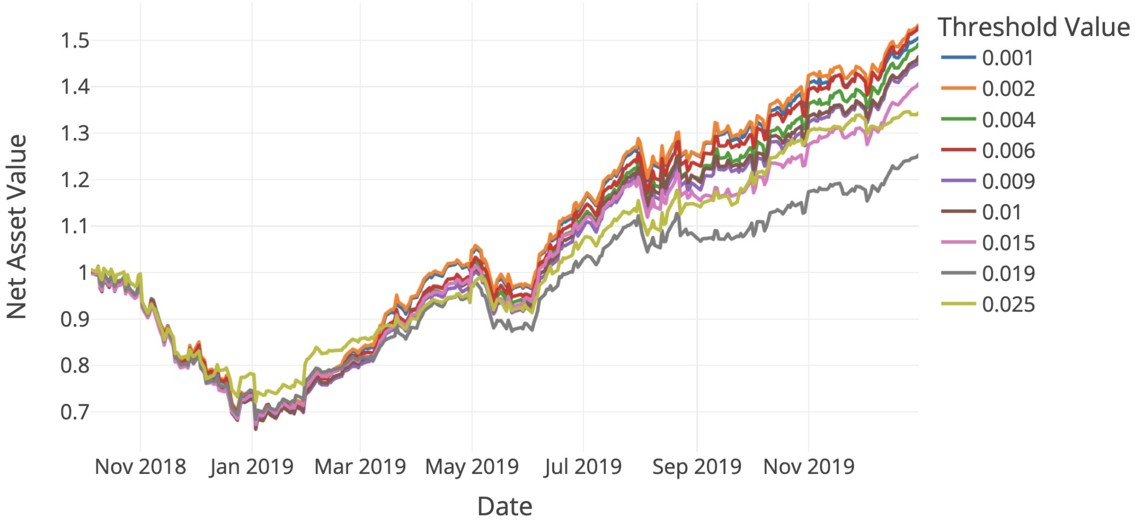
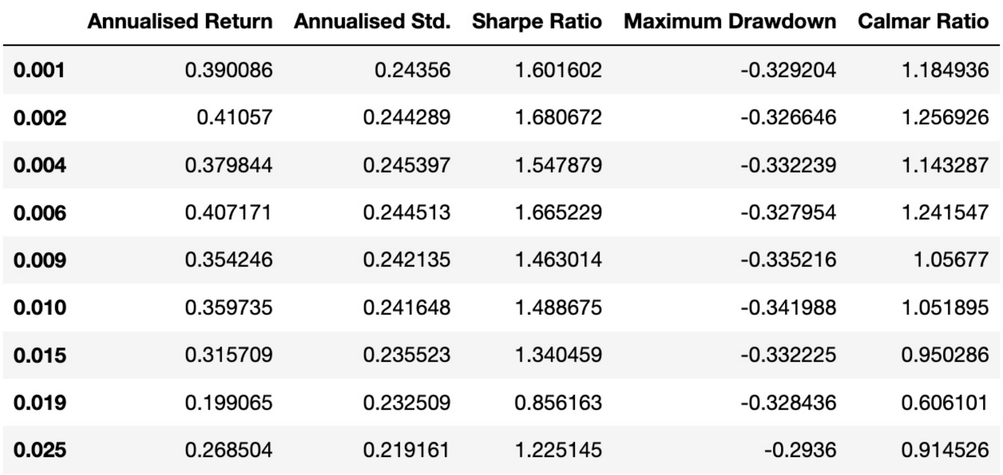


Figure 7.11: Net Asset Value Curve of Meta-labeling with ECDF Method



	Annualised Return	Annualised Std.	Sharpe Ratio	Maximum Drawdown	Calmar Ratio
0.001	0.390086	0.24356	1.601602	-0.329204	1.184936
0.002	0.41057	0.244289	1.680672	-0.326646	1.256926
0.004	0.379844	0.245397	1.547879	-0.332239	1.143287
0.006	0.407171	0.244513	1.665229	-0.327954	1.241547
0.009	0.354246	0.242135	1.463014	-0.335216	1.05677
0.010	0.359735	0.241648	1.488675	-0.341988	1.051895
0.015	0.315709	0.235523	1.340459	-0.332225	0.950286
0.019	0.199065	0.232509	0.856163	-0.328436	0.606101
0.025	0.268504	0.219161	1.225145	-0.2936	0.914526

Figure 7.12: Risk-return Metrics of Meta-labeling with ECDF Method

Comparing the NAV curves in Figure 7.11 with those in Experiment 1, we observe that the overall trend remains largely consistent with the strategy generated by raw trading signals, with no significant changes in style. Analyzing the various strategy metrics in Figure 7.12 and comparing them with those from Experiment 1, we find an improvement in the Sharpe ratio and maximum drawdown for all values of τ. However, there is a decrease in annualized returns, leading to mixed results in the Calmar ratio for different τ values.

In conclusion, meta-labeling with the ECDF method enhances the strategy metrics without introducing drastic changes to the trading style. The improvements achieved with the ECDF method are more subtle and exhibit greater stability compared to the all-or-nothing method used in Experiment 2.

7.5 Analysis Summary

In summary of the four experiments, the following conclusions can be drawn:

First, the LSTM model demonstrates the capability to predict stock price movements at the minute level, enabling the development of profitable trading strategies.

Second, the application of meta-labelling proves effective in filtering false positive trading signals and managing the trade-off between trading precision and trading recall. Consequently, this leads to improved model performance metrics and risk-return metrics.

Third, within the context of meta-labeling, the ECDF method offers a more stable increase in investment metrics when compared to the simpler all-or-nothing method.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

Throughout this research, we have undertaken an extensive exploration of predicting stock price movements and developing trading strategies using LSTM models and meta-labeling, providing novel insights into the intersection of machine learning and financial markets. We designed several experiments to demonstrate the predictive power of LSTM models for stock movements and the impact of meta-labeling on trading strategies.

Our results from the primary model confirm that LSTM models can effectively predict stock price movements at the minute level. This outcome presents potential for building profitable strategies.

Furthermore, the application of meta-labeling proved to be an efficient tool for the enhancement of the trading strategy's performance. Through the secondary model and the use of additional features, the primary model is filtered for false positives, improving the risk-return metrics. Our findings suggest that meta-labelling can generally lead to improvements in investment metrics.

However, the outcomes also indicated that the benefits of meta-labeling can vary significantly depending on the trading method used, thereby suggesting the importance of carefully calibrating these components based on the unique characteristics of each trading strategy and investment style.

8.2 Future Work

While our research has shed light on the potential of LSTM models for predicting stock price movements and the benefits of meta-labeling, it also highlights several opportunities for future work.

Firstly, the utilization of only minute-level price and volume data for a single stock (Apple) somewhat limits the generalizability of our findings. Future research can benefit from applying the same methodological approach to a broader range of stocks and asset classes. Additionally, expanding to higher-level time frames, such as hourly or daily data, may provide insights into the applicability of our model in different trading contexts.

Secondly, in our experiments, the selection of technical indicators can be more scientific by using the techniques such as feature importance. Future studies could explore the efficacy of different technical indicators and feature engineering techniques, which might lead to improvements in the prediction accuracy of the LSTM model and the overall performance of the trading strategy.

Thirdly, in order to conserve computational resources, there are compromises in the training of the LSTM in this paper, although the impact on the overall results is within manageable limits. Future research could search for a larger hyperparameter space to find the optimal hyperparameters during the cross-validation of LSTMs. And cross-validation could be performed individually for each LSTM model used.

Lastly, more sophisticated meta-labelling frameworks can be investigated. This includes: developing different secondary models for long and short signal respectively, calibrating the output of the secondary model to more closely match realistic probabilities, and exploring trading methods other than all-or-nothing and ECDF.

In conclusion, the results of our research provide a promising foundation for future explorations in the field of machine learning applications in financial markets. We hope that our findings will encourage further investigations that will continue to push the boundaries of what is possible in algorithmic trading.

Bibliography

Bollerslev, Tim (1986). “Generalized autoregressive conditional heteroskedasticity”. In: *Journal of Econometrics* 31.3, pp. 307–327. ISSN: 0304-4076. DOI: `10.1016/0304-4076(86)90063-1`.

Cont, R. (2001). “Empirical properties of asset returns: stylized facts and statistical issues”. In: *Quantitative Finance* 1.2, pp. 223–236. DOI: `10.1080/713665670`.

Elman, Jeffrey L. (1990). “Finding structure in time”. In: *Cognitive Science* 14.2, pp. 179–211. ISSN: 0364-0213. DOI: `10.1016/0364-0213(90)90002-E`.

Engle, Robert F. (1982). “Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation”. In: *Econometrica* 50.4, pp. 987–1007. ISSN: 00129682, 14680262.

Fischer, Thomas and Krauss, Christopher (2018). “Deep learning with long short-term memory networks for financial market predictions”. In: *European Journal of Operational Research* 270.2, pp. 654–669. ISSN: 0377-2217. DOI: `10.1016/j.ejor.2017.11.054`.

Gandhmal, Dattatray P. and Kumar, K. (2019). “Systematic analysis and review of stock market prediction techniques”. In: *Computer Science Review* 34, p. 100190. ISSN: 1574-0137. DOI: `10.1016/j.cosrev.2019.08.001`.

Greff, Klaus, Srivastava, Rupesh K., Koutník, Jan, Steunebrink, Bas R., and Schmidhuber, Jürgen (2017). “LSTM: A Search Space Odyssey”. In: *IEEE Transactions on Neural Networks and Learning Systems* 28.10, pp. 2222–2232. DOI: `10.1109/TNNLS.2016.2582924`.

Hochreiter, Sepp and Schmidhuber, Jürgen (1997). “Long Short-Term Memory”. In: *Neural Computation* 9.8, pp. 1735–1780. DOI: `10.1162/neco.1997.9.8.1735`.

Joubert, Jacques Francois (Aug. 2022). "Meta-Labeling: Theory and Framework". en. In: *The Journal of Financial Data Science* 4.3. Company: Institutional Investor Journals Distributor: Institutional Investor Journals Institution: Institutional Investor Journals Label: Institutional Investor Journals Publisher: Portfolio Management Research Section: Quantitative Finance, pp. 31–44. ISSN: 2640-3943. DOI: 10.3905/jfds.2022.1.098.

Li, Audeliano Wolian and Bastos, Guilherme Sousa (2020). "Stock Market Forecasting Using Deep Learning and Technical Analysis: A Systematic Review". In: *IEEE Access* 8, pp. 185232–185242. DOI: 10.1109/ACCESS.2020.3030226.

Lopez de Prado, Marcos (2018). *Advances in financial machine learning.* en. Nashville, TN: John Wiley Sons. ISBN: 9781119482086.

Meyer, Michael, Barziy, Illya, and Joubert, Jacques Francois (n.d.). "Meta-Labeling: Calibration and Position Sizing". en. In: *The Journal of Financial Data Science* (). Company: Institutional Investor Journals Distributor: Institutional Investor Journals Institution: Institutional Investor Journals Label: Institutional Investor Journals Publisher: Portfolio Management Research Section: Quantitative Finance. ISSN: 2640-3943. DOI: 10.3905/jfds.2023.1.119.

Meyer, Michael, Joubert, Jacques Francois, and Alfeus, Mesias (n.d.). "Meta-Labeling Architecture". en. In: *The Journal of Financial Data Science* (). Company: Institutional Investor Journals Distributor: Institutional Investor Journals Institution: Institutional Investor Journals Label: Institutional Investor Journals Publisher: Portfolio Management Research Section: Quantitative Finance. ISSN: 2640-3943. DOI: 10.3905/jfds.2022.1.108.

Thumm, Dennis, Barucca, Paolo, and Joubert, Jacques Francois (Dec. 2022). "Ensemble Meta-Labeling". en. In: *The Journal of Financial Data Science.* Company: Institutional Investor Journals Distributor: Institutional Investor Journals Institution: Institutional Investor Journals Label: Institutional Investor Journals Publisher: Portfolio Management Research Section: Quantitative Finance. ISSN: 2640-3943. DOI: 10.3905/jfds.2022.1.114.

White, Halbert L. (1988). "Economic prediction using neural networks: the case of IBM daily stock returns". In: *IEEE 1988 International Conference on Neural Networks*, 451–458 vol.2.

Appendix

Appendix A: Calculation of Technical Indicators

Exponential Moving Average:

$$EMA(Close,n)_t = \frac{1}{n}\sum_{i=1}^{n} 0.9^i Close_{t-i}$$

Simple Moving Average:

$$SMA(Close,n)_t = \frac{1}{n}\sum_{i=1}^{n} Close_{t-i}$$

Commodity Channel Index:

$$CCI = (TypicalPrice - SMA(TypicalPrice, N))/(0.015 * MeanDeviation)$$

$$MeanDeviation = \frac{1}{15}\sum_{i=1}^{15} |TypicalPrice - SMA|$$

$$TypicalPrice = \frac{High + Low + Close}{3}$$

Moving Average Convergence/Divergence:

$$MACD = EMA(10) - EMA(30)$$

Stochastic Relative Strength Index:

$$STOCHRSI = (RSI - RSILowestLow)/(RSIHighestHigh - RSILowestLow)$$

$$RSI = 100 - (100/(1 + RS))$$

$$RS = AvgUp/AvgDown$$

Williams' %R:

$$WILLR = \frac{(HighestHigh - Close)}{(HighestHigh - LowestLow)} * (-100)$$

Chaikin A/D Line:

$$AD_t = AD_{t-1} + MFV$$

$$MFV = MF * Volume$$

$$MF = \frac{Close - Open}{High - Low}$$

Chaikin A/D Oscillator:

$$ADOSC = EMA(AD, 3) - EMA(AD, 10)$$

Average True Range:

$$ATR = \frac{1}{n} \sum_{i=1}^{15} TR_i$$

$$TR = max(High, Close_{prev}) - min(Low, Close_{prev})$$