Portfolio Risk-Return Decision Optimization Using AI

by

Junyao Yang

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
Dec 9, 2023

Keywords: Risk-return Optimization, Portfolio Decision making, Machine Learning, Bilevel Optimization

Approved by

Aleksandr Vinel, Chair, Associate Professor of Industrial and Systems Engineering
Jeffrey Smith, Professor of Industrial and Systems Engineering
Richard Sesek, Professor of Industrial and Systems Engineering
Jia Liu, Assistant Professor of Industrial and Systems Engineering

Abstract

Academia and businesses have extensively studied portfolio decision-making under uncertainty and risk. Prediction-based portfolio models usually forecast either the portfolio's future return or risk. On the other hand, portfolio optimization models are built upon statistically defined risk measures or returns using historical scenarios. However, multiple reasons are leading even advanced models away from good performance on both topics, mainly due to the high stochastic market behaviors. However, artificial intelligence and risk management models have the potential to interact with each other and achieve better outcomes in making better decisions. In our study, we assumed that if portfolios can be optimized upon profitable prediction results, the outcome will be beneficial from the combination of machine learning prediction and risk minimization portfolio optimization. In addition, it is believed that stock movements are predictable using various data sources. We use open-source API and public financial databases to generate prediction data. All data is processed through machine learning models, including SVM, Deep Neural Network, and Long Short Term Memory. Different metrics, including accuracy, Cross Entropy Loss, etc, are used to measure the prediction performance. The prediction results are selected and applied to the risk optimization model to obtain an optimal asset allocation strategy. The portfolio risk will be optimized over the conditional value at risk, which could be calculated using either historical scenarios or predictions-based scenarios. Sets of trading simulations have been devised to evaluate the performance of the proposed models under the condition of dynamic asset allocation. In addition, a Broker-Investor competition problem is formulated using bilevel architecture. The broker and investor are aligned by returns(lower level) while competing on risk(upper level). A bilevel optimization model fits well in simulating the situation and provides a novel view of risk and return management. From our case study, we

achieve several conclusions: (1) the prediction accuracy has a significant impact on portfolio outcomes; (2) the prediction system, along with the optimization model, leads to better and more profitable portfolio outcomes. (3) bilevel portfolio risk-return framework shows its potential to model a realistic business situation and provide novel risk-return research paths in the future.

Acknowledgments

First, I would like to thank my dear wife for all her love and support. It took her incredible courage to carry and bring such an adorable girl to life during the COVID-19 pandemic. I absorbed extraordinary motivation and inspiration from this marriage in finishing my research and learning to be a better husband and father. This dissertation is completed through hard work, perseverance, and, most importantly, love from my family. In addition, I must sincerely express my gratitude to my advisor, Dr. Aleksandr Vinel. He is the most supportive and knowledgeable advisor I could have asked for. Dr. Vinel carries me through not only the difficulties in research but also the hardships in life. Dr. Vinel was dedicated to leading me on the research path and giving me much freedom to develop my research interests further. I could not finish my research without his intelligent guidance and countless support. I would also like to thank the other committee members, Dr. Jeffery Smith, Dr. Richard Sesek, and Dr. Jia Liu, for their time and invaluable advice. A special thanks goes to my former committee member, Dr. James Barth. I extend my deepest condolences for the passing of such a respectable professor. I am honored to have Dr. Barth's spirit in my mind while finishing this dissertation. Last but not least, I need to thank my parents for taking me to this beautiful and challenging world. A warm family is essential for my life journey.

Table of Contents

List of Figures

List of Tables

Chapter 1

Dissertation Introduction

Decision-making in portfolio management occurs before the market trading operations. Market uncertainty and risk-return management are the main challenges for investors. Academia and businesses have conducted numerous studies on portfolio management via different objectives. In modern portfolio theory(MPT), investors are assumed to be risk averse, meaning that a portfolio is said to dominate another if either a higher expected return or lower risk is achieved. Indeed, investors are willing to take more risk only if compensated by higher expected returns. Thus, an optimization framework can be modeled through a portfolio of assets such that the expected return is maximized under a given level of risk. One major drawback of the MPT is that it highly depends on historical data and usually produces conservative results. Besides, it does not consider any auxiliary data, even if such indicators, that are available during decision-making, may be predictive of future market movement.

Predictions are an important topic in decision-making under uncertainty. It projects future insights that will generally describe how the decision environment changes while making decisions. Predictions usually require auxiliary data and prediction models. Research has shown that considerable auxiliary data can be used in market trend prediction. And data mining techniques can extract and capture potential correlated features from various data sources. In addition, artificial intelligence provides robust prediction models to process market data. In addition, statistical models can analyze prediction results and produce mathematical impacts on final decisions. However, prediction models usually encounter accuracy issues, and prediction results can not be directly applied to decisions. Thus, a well-defined prescriptive analytic model is desired to unify prediction and decision-making. This dissertation aims to

develop a novel decision-making framework for portfolio design, that incorporates advanced machine learning for stock movement forecasting with traditional portfolio optimization.

In Chapter 2, we propose a hybrid risk optimization and machine learning model for portfolio risk-return management that takes into account binary classification on stock movement only. It is motivated by the idea that binary classification (i.e., predicting whether a particular asset will go up to down in value) is a considerably easier forecasting task (for example, compared to predicting actual future price), and hence can be expected to achieve higher accuracy. Specifically, we use machine learning algorithms to perform market trend prediction, which is then used to filter the stock selection process. Finally, the portfolio will be constructed using the conditional value at risk optimization model using machine learning-selected stocks. This model brings new insight into risk optimization and returns management. To the best of the author's knowledge, this is a novel approach to portfolio design.

In Chapter 3, we intend to use a prescriptive multi-objective approach to optimize portfolio risk and return, this time accounting for probability estimation. In the previous model, our single objective was the CVaR function, which focuses on portfolio risk minimization. Although we have a return constraint in the optimization progress, this return constraint is still built upon the historical data. In other words, our model does not optimize the returns, and the future return expectation is not guaranteed during portfolio design. To address these issues, we employ recently developed machine learning techniques for calibrating the probabilistic performance of a predictive task, that enable estimation of the probability of a particular class outcome. Although several studies have been conducted on approximating the market trend probability distribution, most have relied on a prescriptive analysis to model a stochastic problem, for example, to maximize the expected return. The authors believe there is a lack of research objectives on risk-return management. Thus, a novel prescriptive risk-return multi-objective optimization model will be proposed for portfolio design.

For the last chapter, the authors are inspired by the Principle-Agent situation that occurs in many cases. The Principle-Agent problem (Jensen and Meckling, 1976) describes a conflict between the asset owners and the authorized agents who act on the owner's behalf. The work develops a mathematical relationship between ownership and agent, which they defined as the separation of asset ownership and control. Given that the agent prefers to act in his interests, the owner would like an incentive strategy that aligns the agent's interests with his preference. This separate control appears when the owner hires an agent, for example, an investment trust. The owner and trust have exactly a Principle-Agent relationship, resulting in a bilevel optimization problem. The first level focuses on profit maximization, which is the owner's (principle) objective. The first level is constrained by the second level, an optimization sub-problem. Although the agent may want to act on his preference, he also wants to avoid extreme losses. Thus, a new optimization framework can be introduced to solve this financial problem by determining the optimal solution to this bi-level optimization. To the authors' best knowledge, this is a novel portfolio design approach.

Chapter 2

Portfolio Design Through a Hybrid Risk Optimization and Machine Learning Approach

2.1 Introduction

Portfolio optimization models aim to develop effective trading strategies for managing financial assets' risk and return. Typical approaches in the literature begin with a set of historical data on asset prices and then apply a combination of optimization and statistical techniques. Traditional portfolio optimization models rely on historical data, i.e., primarily treat the historical performance as indicative of future performance, without explicitly aiming to predict it. Instead, they employ a correlation structure as a way to enforce diversification. Therefore, their performance is limited due to a lack of ability to consider future market trends, even though some indicators of future behavior may be available when the portfolio is built. Artificial intelligence techniques are increasingly attracting attention in analyzing financial portfolio data, specifically for constructing market prediction models. While these too rely on historical data, a variety of studies employ various sources of auxiliary (forward-looking) non-traditional market-related data to forecast asset behavior.

In principle, in the presence of an accurate predictive model, no additional optimization is needed. Indeed, if an investor has access to a high-quality forecast, then the resulting decisions are straightforward (buy assets that are predicted to go up in price, and potentially short assets that are expected to lose value). At the same time, due to high stochastic and multi-factor properties, even modern prediction models are not guaranteed to obtain high accuracy. As a result, using only prediction models in trading strategy will introduce significant risk factors to portfolios. Therefore, even in the presence of advanced machine learning forecasting tools, an investor may still benefit from employing traditional risk-averse

stochastic optimization techniques. The goal of this study is to investigate the best ways to organize such an approach.

Two particular challenges can be identified. First, a significant amount of literature is dedicated to both forecasting models and stochastic optimization for portfolio risk management, yet few existing efforts are concerned with considering a combined approach. In other words, both streams of literature have seen significant advances in terms of improving forecasting accuracy and better ways to improve portfolio diversification to hedge against risks. Still, the idea of employing machine learning-based prediction within a stochastic optimization model has not been extensively studied.

Secondly, while advanced machine learning models have been successfully employed to forecast asset prices, accurate prediction remains an exceptionally challenging task, especially on a time scale measured in days. One direction that has seen somewhat more promising results is to predict asset movement rather than exact price. In this approach, a machine learning model is trained to forecast whether a particular asset will increase or decrease in value, i.e., the underlying task is a simple binary classification. Since such a task is simpler than regression-like approaches required for predicting actual price, the existing models reported in the literature have decent performance.

Therefore, the goal of the proposed approach is to construct an optimization framework that can take advantage of both modern risk-averse stochastic optimization and advanced asset movement prediction models. Note that the optimization step is intended to hedge the investor against correlations both in asset prices (this is the traditional goal of diversification) and in forecasting errors. The latter issue is unique to the proposed framework and has not been extensively studied in the literature before.

Here we propose to design a simple approach to combining prediction and optimization. Specifically, we surmise that the benefits of both can be attained if on the one hand, only assets predicted to increase in value are selected. On the other hand, we will employ a popular risk-diversification approach – mean-CVaR optimization – to hedge against prediction inaccuracy

as much as possible. While straightforward in construction, we show that this simple approach can outperform both diversification-without-predictions and prediction-only portfolios.

This research aims to address these challenges and focuses on the following proposed procedure.

1. We implement and compare three machine learning algorithms to forecast short-term movements for a collection of U.S. stocks.

2. We propose a hybrid machine learning-informed CVaR risk optimization model to hedge the risk of stock portfolios, that explicitly takes advantage of the the predictive models.

3. Our proposed risk optimization model is then compared in terms of returns with the standard CVaR optimization model and purely prediction-based model to evaluate its performance.

Note that we present the approach as explicitly aimed at financial portfolio optimization. At the same time, similar challenges arise in other similar settings, where both machine learning predictions and stochastic optimization are customarily employed.

2.2 Literature Review

2.2.1 Stock Market Prediction Models

Early stock market prediction approaches have largely relied on the Efficient Market Hypothesis (EMH) (Fama, 1965) and random walk theory (RWT) (Cootner, 1964; Fama, 1965, 1991, 1995), posing that the stock market is purely stochastic, and thus, can not be predicted with any significant accuracy (Bollen et al., 2011). However, subsequent studies challenged this perspective (e.g. Malkiel, 2003; Smith, 2003; Nofsinger, 2005; Prechter Jr and Parker, 2007; Bollen et al., 2011). These attempted to show that the stock market indeed can be predicted, at least to a certain extent. Investors may derive information from various sources to collect disparate data and correlate those with the market movements (e.g. Bollen

et al., 2011; Mittal and Goel, 2012; Jiao et al., 2020). Some results in the literature suggest that early trading indicators can be extracted from online sources, including Google Trends, Wikipedia's financially-related website traffic volume, and news sentiments (e.g. Moat et al., 2013; Li et al., 2014; Preis et al., 2013). Further, studies performed on social media, for example, X (Twitter) and financial blogs, indicate that these early trading signals can also serve as valuable predictors Nguyen et al. (2015).

From the research perspective, highly correlated features and appropriate prediction algorithms are two critical factors in stock movement prediction modeling. Most advanced statistic models and artificial intelligence approaches have been employed in stock market prediction with varied results (e.g. Weng et al., 2017; Ou and Wang, 2009; Angadi and Kulkarni, 2015; Weng et al., 2017). Recently, even modern advanced deep learning algorithms, including Deep Neural Networks, Convolutional Neural Networks (CNN), and Long Short Term Memory (LSTM) Recurrent Neural Networks, have been explored with different data sources for this purpose (e.g. Weng et al., 2018; Selvin et al., 2017). Finally, advances in Natural Language Processing (NLP) have been instrumental in enabling sentiment analysis of the auxiliary data sources, which, in turn, can then be used as early predictors (Xing et al., 2018). In the context of stock movement prediction, the prediction accuracy varies significantly across different research works. In general, the prediction accuracy has a range from 45% to 87.5% (Mittal and Goel, 2012).

Note that this study does not aim to improve the existing predictive models. Instead, our goal is to consider novel approaches to employing them in optimization. As such, for our case study, as discussed below, we rely on some of the available models. Specifically, SVM, NN, and LSTM are studied in more detail.

2.2.2 Models for Financial Portfolio Optimization

Decision-making and optimization under uncertainty have attracted significant interest in operations research and management sciences communities. One fundamental principle in

this related field is von Neumann and Morgenstern's utility theory of choice under uncertainty (Morgenstern and Von Neumann, 1953). Markowitz volatility risk portfolio model (Markowitz, 1952) represents the other cornerstone of the modern risk management theory. Based on the Markowitz model, one widely known risk model is the Value-at-Risk (VaR) measure (e.g. Jorion, 1997; Duffie and Pan, 1997). It has been adopted as the standard for measuring risk in many financial use cases. At the same time, criticisms are accompanying its growing popularity. Specifically, it does not take into account the extreme losses beyond the α-quantile, and, paradoxically, VaR can be shown to, in general, be inconsistent with the fundamental risk management principle of risk reduction via diversification (Krokhmal et al., 2013). To overcome these drawbacks, Conditional-Value-at-Risk (CVaR) was introduced to overcome the shortcomings of VaR. It mainly focuses on extreme losses by estimating the average loss function exceeding a certain threshold, and, through the carefully constructed definition, is convex and allows for efficient optimization representation (e.g. Rockafellar et al., 2000).

2.2.3 Portfolio Optimization with Predictive Models

Recent developments have seen the initiation of novel research endeavors in decision-making, focusing on the collaboration between machine learning and portfolio optimization. Newly proposed methods use auxiliary data and predictive models to turn the predicable target into prescriptive decisions. Advanced data mining and deep learning algorithms are experimented with in predicting the most profitable return on stocks. Portfolios are built upon those predicted results that indicate the most profitable allocation. In the works (e.g. Ma et al., 2020, 2021), the actual stock return is predicted, followed by a risk minimization model built by integrating the predicted returns and predictive errors. Regardless of the optimization techniques, Ma's works define the portfolio risk directly using the prediction error and optimize the risk in the allocation. Other research in the literature also considers stock return prediction(Ta et al., 2020), and the predicted return is incorporated with the simple Mean-Variance model to optimize the risk. New approaches take advantage of both return

prediction and risk optimization. Based on their experiments, better and more profitable outcomes can be achieved. However, accurately predicting actual stock returns is a formidable challenge, with no assurance of precision.

2.3 Methods

In this paper, we propose a two-phase prediction-based portfolio risk optimization model. In the first phase, we implement machine learning classification algorithms to predict one-day stock closing price movements. Binary classification results can be obtained, indicating market trends. These prediction results are introduced to the portfolio risk optimization model. Since our prediction results are on a daily basis, investors can allocate their budget through the guideline of predicted optimization results. Indeed, this two-phase model is designed to return a daily risk-optimized allocation strategy that is supposed to be profitable at the same time. Thus, our model can be summarized in Figure 2.1.

We next describe each component in detail. The main contribution of the approach is in the construction of the optimization problem in Section 2.3.3. First, though, we elaborate on the predictive models considered as well data preparation process, which serve as the basis for the proposed approach.

2.3.1 Data Acquisition and Prepossessing

In our prediction phase, stocks may have different feature dependencies and feature space needs to be carefully analyzed. Based on data availability, we chose 60 months period from 07/01/2015 to 06/30/2020 for all of the experiments. We employ four data sources, following Weng et al. (e.g. 2017, 2018). All provide free Python API access for the period in question.

First, Yahoo Finance provides us with the historical stock price, including daily Opening/Closing price, High/Low, Adjust Close, and volume of trades for each stock per trading day. In addition, the Dow Jones Industrial Average and NASDAQ composite indices are

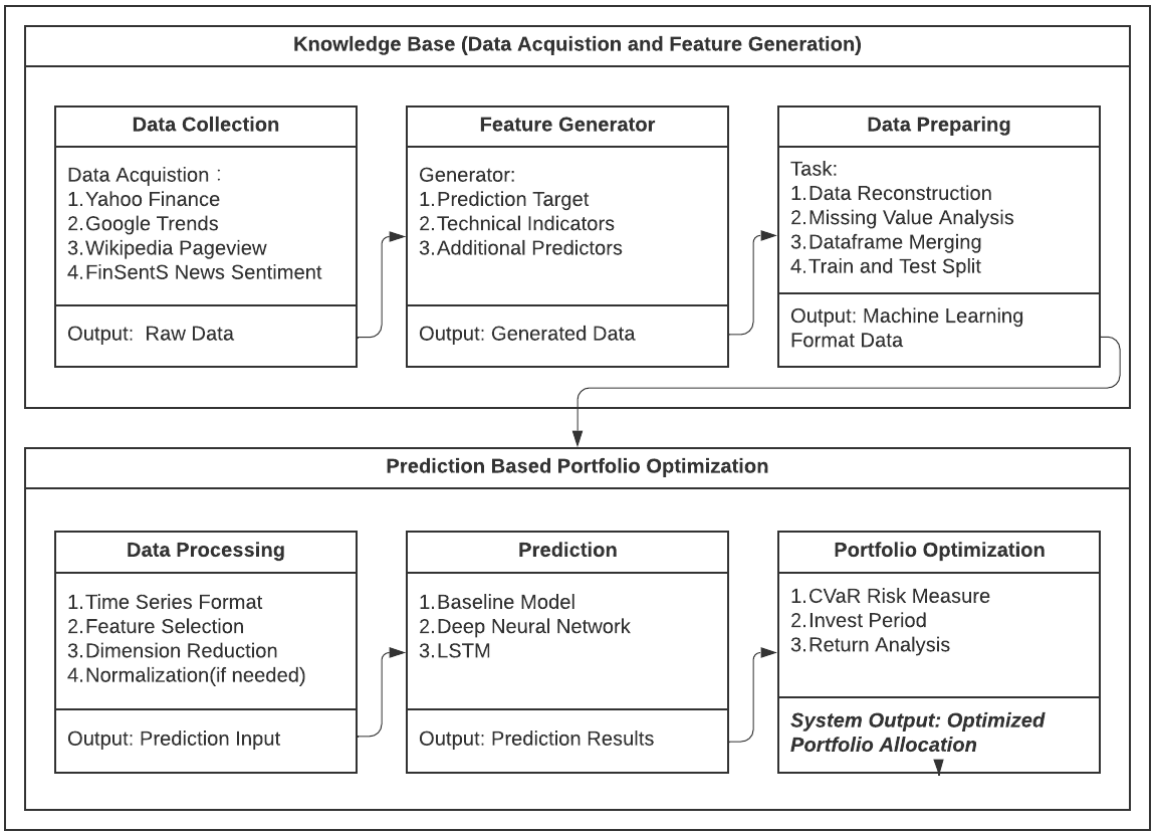


Figure 2.1: Proposed Model Overview

also included in each data set since they are common market indicators. Further, we calculate price–earnings ratio using historical quarterly Trailing Twelve Months (TTM) and daily closing price. Recall that the price–earnings ratio can be seen as an estimator for the fundamental health of the company (Gabrielsson and Johansson, 2015).

In our second and third data sources, we extract online data activity data from Google Trends and Wikipedia Pageview. Google Trends analyses how frequently a given search term is entered into Google's search engine relative to the site's total search volume over a given period. It reflects search query popularity across regions and languages. Google provides API modules to retrieve data from its server. However, Google currently sets limits on the time resolution based on queries' time frames. For example, a query for the last 7 days will return hourly search trends. The Daily data is only available with a query period between 9 months and up to 36 hours before the query. Therefore, for our purposes, only monthly

data is available. In addition, it is possible to recursively make queries in a 9-month interval until the full time range is reached. However, this recursive method will produce inconsistent results, and different data will be returned even in the same time interval. To overcome this limitation and get precise daily data, data reconstruction is an option. To perform reconstruction, we make multiple 9-month period queries with significant overlapping periods and use the overlapped periods to have consistent scaling. Detailed implementation can be found in `https://github.com/jzy0040/Dissertation`.

The third data source, Wikipedia provides website analysis and user traffic counts. We queried the daily traffic data from their website www.wikipediatrends.com. Detailed implementation can be found in the code as well. In both cases, it is assumed that the potential trading signals can be extracted from financial-related user traffic in either Wikipedia or Google. This has previously been observed in the literature, Weng et al. (e.g. 2017, 2018).

The last data source, FinSentS Web News Sentiment, offers daily media sentiment indicators for global equities. It contains five indicators. First, the sentiment score is a numeric measure of the bullishness/bearishness of news coverage of the stock. High/low sentiments are the daily highest and lowest sentiment scores. News Volume is the absolute number of news articles covering the stocks. News Buzz is a numeric measure of the change in coverage volume for the stock. Literature suggests that a correlation does exist between news sentiment and stock movements Mittal and Goel (e.g. 2012); Li et al. (e.g. 2014); Nguyen et al. (e.g. 2015). We expect that news sentiment will be beneficial to prediction accuracy. However, significant issues with missing values are present, particularly, for smaller or less popular financial assets. In order to obtain continuous time series data we, therefore, must consider missing data imputation. Implementation details are introduced later in the Experiment setup section.

A summary of the data sources is given in Table 2.1. Note that, naturally, financial trading data is only available on trading days, while the rest of the sources can be queried every day.

Data Sources	Size	Continuity	Missing Values	Reconstruct
Yahoo Finance	1259, 6	Trading days only	No	No
Google Trends	1827, 1	All	No	Yes
Wikipedia Pageview	1827, 1	All	No	No
FinSentS Sentiment	1827, 8	All	Yes	Yes
Note: all data are indexed by date and merged accordingly.				

Table 2.1: Data Sources Summary

Additional predictors are constructed by calculating some common technical indicators, following (Bao and Yang, 2008). In finance, technical indicator is a fundamental technical analysis methodology that can provide insights into historical and current price trends, volatility, and potential future price movements. A summary of indicators used in this work is given in Table 2.2.

Indicators	Yahoo Finance	Google Trends	Wiki Pageview	FinSentS
Disparity	Yes	Yes	Yes	Yes
MA	Yes	Yes	No	Yes
EMA	Yes	Yes	No	Yes
ROC	Yes	Yes	Yes	Yes
SO	Yes	No	No	Yes
Willams R	Yes	No	No	Yes
RSI	Yes	Yes	Yes	Yes
MACD	Yes	Yes	Yes	Yes
PPO	Yes	Yes	Yes	Yes
CMO	Yes	Yes	Yes	Yes
Note: all data are indexed by date and merged accordingly.				

Table 2.2: Technical Indicator Applied for Data Sources

One significant factor in our study is the prediction target. Researchers focus on different target values in predicting stock movements. As we mentioned before, we will focus on one-day stock movements, rather than price or return prediction. Equation (2.1) gives the

expression that we will interpret as the prediction target.

$$p_i = \begin{cases} 0, & \text{if } Close_{i+1} - Close_i \leq 0 \\ 1, & \text{else} \end{cases} \tag{2.1}$$

Naturally, this leads to a binary classification problem.

2.3.2 Prediction Models

In the first phase, we explore three machine learning classification algorithms. All data is prepared in time series data format and indexed by date. Meanwhile, both the non-time series model and time series model are compared by prediction accuracy. The best model for each stock is statistically evaluated and the best overall model will be used to generalize on the testing set. All prediction models are introduced in the following paragraphs below. Readers should note that we will develop all three prediction models for each stock using the same data sets, though feature spaces may be different due to normalization or time-series requirements.

In the context of stock market prediction, Support Vector Machine(SVM) is a basic and commonly used classification algorithm (e.g. Kim, 2003; Nguyen et al., 2015; Weng et al., 2017). SVM typically distinguishes the data by constructing a hyperplane in the feature space that maximizes the margins between the support vectors. It has been widely accepted that SVM can handle high-dimensional data for classification purposes. Due to the non-linearity of the feature space, kernel functions could be considered and they map the original feature space to high dimensional space, where training samples are linearly separable. As a result, SVM is capable of performing well for high-dimensional feature space regardless of linearity. Consequently, we will employ SVM as the baseline method.

Artificial Neural Network(ANN) aspires loosely to emulate the human brain and its learning procedure. It was originally developed by psychologist Frank Rosenblatt (Rosenblatt, 1958) who invented the Perceptron algorithm. ANN is a universal approximating algorithm

with many possible variations, including multi-layers Perceptron, Recurrent Neural Networks, and Convolutional Neural Networks in deep learning, etc. In stock prediction literature, ANNs have been extensively studied on different market indices (e.g. Kim and Han, 2000; Atsalakis and Valavanis, 2009; Dase and Pawar, 2010; Weng et al., 2017, 2018). Because of its universal approximating capabilities, it demonstrates satisfactory performance with both linear and nonlinear functions. Compared with SVM, it does not require a kernel function selection, which means it maintains the original feature spaces in training and testing. In this paper, our ANN models are designed to be fully connected multi-layer neural networks. We will use the Rectified Linear Unit as the activation function. Theoretically, it is non-trivial to determine how many layers and neurons are needed in the network. These decisions must be made through experiments. Note that both SVM and ANN are non-time-series models, i.e., disregard temporal dimension of the training data.

Long Short Term Memory(LSTM) is an affiliate member of Recurrent Neural Network architecture (Hochreiter and Schmidhuber, 1997). Unlike standard feed-forward neural networks, LSTM architecture takes advantage of feedback process that passes sample data to the previous or the same layer. It can process not only single sample data, but also a sequence of samples. It has been applied for both classification and regression purposes (e.g. Selvin et al., 2017; Nelson et al., 2017) and has been able to accomplish better results on time series problems. In our LSTM models, we have one LSTM and a dense layer designed as our hidden layers. The output layer which is also a dense layer uses the Sigmoid activation function. In training, we apply Binary Cross Entropy loss function. Detailed implementation is described in the Experiments section.

2.3.3 CVaR Portfolio Optimization

Conditional-value-at-risk (CVaR) for continuous distribution is defined by taking the weighted average of the extreme losses that exceed a certain quantile (i.e., Value-at-Risk, VaR) of the loss distribution. It has been designed as a remedy for the drawbacks of VaR. First,

CVaR addresses the inability of VaR to account for extreme but rare losses, by averaging the tail of the loss distribution. Secondly, unlike VaR, CVaR is coherent, i.e., is consistent with the principle of risk reduction through diversification. Further, CVaR can be defined as a solution to a linear programming problem, meaning that it allows for efficient evaluation in a portfolio optimization setting. Therefore, CVaR is widely used as the main way to account for risk in risk-reward optimization framework. See Rockafellar et al. (2000) and Krokhmal et al. (2013) for more details.

In a traditional mean-CVaR optimization problem, the goal is to select portfolio weights x_i, for each of n financial assets. It is assumed that m scenarios for the realization of asset returns are given (usually obtained from historical performance). These are defined as r_{ij}. Therefore, portfolio return under scenario j is given by $\sum_j r_{ij}x_i$. The problem in question is to select a portfolio that balances average return against the CVaR of negative return (i.e., CVaR of portfolio losses). This is a bi-objective optimization problem with the solution given by a Pareto front (efficient frontier). It can be obtained, for example, by solving the following optimization problem (see, for example, Krokhmal et al., 2013)

$$
\begin{aligned}
CVaR_\alpha(\mathbf{x}) = \min_{\eta\in\mathbb{R}} \quad & \eta + \frac{1}{m(1-\alpha)}\sum_{j=1}^{m}\Big[-\sum_{i=1}^{n} x_i r_{ij} - \eta\Big]^+ \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1 \\
& E_j\Big[\sum_i r_{ij}x_i\Big] \geq R \\
& x_i \geq 0, \quad i = 1,\ldots,n.
\end{aligned}
$$

Here, α is the confidence level of CVaR (i.e., CVaR measures the average in $1-\alpha$ worst cases), R gives a target average rate of return, $[t]^+ = \max\{0, t\}$ and η is an auxiliary variable used in the definition of CVaR. it can be reformulated as a linear program by introducing

variables w_j as follows.

$$
\begin{aligned}
CVaR_\alpha(\mathbf{x}) = \min_{\eta \in \mathbb{R}} \quad & \eta + \frac{1}{m(1-\alpha)} \sum_{i=1}^{m} w_i \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1 \\
& \sum_{i} \bar{r}_i x_i \geq R \\
& w_j \geq -\sum_{i=0}^{n} x_i r_{ij} - \eta \\
& w_j \geq 0, \quad j = 1, \ldots, m \\
& x_i \geq 0, \quad i = 1, \ldots, n
\end{aligned}
$$

where $\bar{r}_i$ is the average return on asset i.

Now, suppose that in addition to historical returns r_{ij}, asset movement prediction is available, in the form of p_i, defined above. Naturally, assets that are predicted to decrease in value should be avoided in the portfolio. Therefore, we propose to solve the following problem.

$$
\begin{aligned}
CVaR_\alpha(\mathbf{x}, \mathbf{p}) = \min_{\eta \in \mathbb{R}} \quad & \eta + \frac{1}{m(1-\alpha)} \sum_{i=1}^{m} w_j && (2.2) \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1 && (2.3) \\
& \sum_{i} \bar{r}_i x_i \geq R && (2.4) \\
& w_j \geq -\sum_{i=0}^{n} x_i r_{ij} - \eta && (2.5) \\
& x_i \leq p_i, \quad i = 1, \ldots, n && (2.6) \\
& w_j \geq 0, \quad j = 1, \ldots, m && (2.7) \\
& x_i \geq 0, \quad i = 1, \ldots, n && (2.8)
\end{aligned}
$$

Constraint 2.6 enforces that only assets predicted to increase in value are selected. On the other hand, the rest of the problem ensures that the portfolio is still diversified. The solution to an individual problem results in a single portfolio that is optimal for the prescribed average return R. The Pareto front can then be constructed by varying this value.

Observe that the problem may be infeasible in two cases. First, if the target average return is too high and is therefore unattainable. Secondly, if none of the assets are predicted to increase in value. The former can be avoided by carefully selecting target R. In the following experiments, we select $R = R_{0.8}$, which is the 80th quantile of average return among the assets. The latter case means that the decision maker is required to determine a trading policy for when none of the assets are predicted to grow. In our experiments, we assume that in this case, the decision is to withdraw from the market and hold cash.

The resulting problem is still linear, and hence can be solved efficiently with standard off-the-shelf solvers. We next evaluate its performance base in a numerical experiment with the data described earlier. In the next chapters, we will consider more advanced decision-making approaches to constructing the portfolio – a bi-objective and bi-level optimization. For the remainder of this chapter, we will concentrate on evaluating the performance of this simple approach.

2.4 Experiments

Experiments are designed to illustrate our approach by comparing 3 different portfolio allocation strategies. We use two baseline models to compare with our proposed model. The first baseline model is the standard CVaR model which is well described in (Rockafellar et al., 2000). We will refer to it as CVaR portfolio. Naturally, it ignores the predictions, and any gains are realized through historical a combination of average performance and diversification based on historical correlation structure. Our second baseline model is constructed as a pure prediction portfolio, which will be referred to as Prediction+EW. Here, we build a portfolio by assigning equal weight to each asset that is predicted to grow. This is a natural

option, that takes full advantage of the forecast, but still attempts minimal diversification by spreading the budget among the selected assets.

2.4.1 Experimental Setup

We selected 36 stocks for the experiment: 30 of the top Fortune 500 list and 6 additional famous companies. Note that since our predictive models rely on news-based predictors, more widely discussed assets can be expected to have lower forecasting errors. The four data sources discussed earlier provide the necessary time series data. Note that two issues should be addressed here: inconsistency between reporting frequency (trading days only vs once a day) and missing data (particularly for FinSentS News Sentiment index on days that a particular asset was not discssed in relevant news sources).

First, we calculate all additional features before merging different data sources. Note that sequential inputs are needed for a number of constructed features. For example, the Relative Strength Index indicator commonly uses 14 trading periods. If we merge data according to the Yahoo Finance data index, and indeed we must use it because our target value is aligned with the Yahoo Finance data index, we will lose continuity in calculating indicators for Google Trends, Wikipedia Pageview, and FinSentS News Sentiment. Second, missing values in FinSentS News Sentiment are randomly distributed and varied by amount. It is non-trivial to perform data imputation for a large missing value data set. In our implementation, missing values are imputed using a quadratic approximation function when the missing rate is lower than 0.2. For those with missing values greater than 0.2, the FinSentS News Sentiment data source and its additional generated features are ignored. Therefore, FinSentS News Sentiment does not cover all target stocks. We also ignore those stocks unavailable in FinSentS News Sentiment.

We divide our time series data into two parts: 07/01/2015 to 12/31/2019 for training containing 1101 trading dates, and 01/01/2020 to 06/30/2020 for testing containing 125 trading dates. Experiment setups are consistent among all models. We will discuss our

prediction model first and each strategy in detail in the following subsections. Readers should note that all prediction experiments are conducted on each stock and optimal settings could be different among stocks.

2.4.2 Prediction Models

SVM

Corinna Cortes and Vladimir Vapnik (Cortes and Vapnik, 1995) first introduced SVMs in 1995, which are supervised learning models that analyze data for classification and regression purposes. SVM maps training samples to feature space to maximize the width of the two support vectors. New samples are then considered in the same feature space and predicted for a specific purpose. In addition to linear separable space, SVM can also efficiently apply to non-linear space with the help of kernel function, which maps original feature space to higher dimensional feature space. In the case of non-linearity and good generalization, our experiment uses Radial Basis(RBF) as kernel function, which is defined in equation 2.9.

$$K(v_i, v_j) = \exp(-\gamma \parallel v_i - v_j \parallel^2). \tag{2.9}$$

Parameter γ is the kernel coefficient of RBF and v_i represents the training features in i dimensional spaces. The settings of our SVM model are listed in Table 2.3, where C represents the regularization parameter of SVM. The shuffle function allows samples randomly fed into the model when training SVM.

Parameters	**Values**	**Description**
Kernel	RBF	Radial basis function kernel
Gamma	$(Dim \cdot X.var())^{-1}$	Kernel Coefficient
C	$1, 2, 4, 6, 8, 16$	Regularization parameter
Shuffle	True	Sample order randomization

Table 2.3: Parameters of SVM

ANN

ANN is a widely employed classification and regression algorithm. ANN model usually consists of one input layer, multiple hidden layers, and one output layer. Theoretically, more hidden layers are aimed to achieve better learning ability. In our experiments, we consistently use 2 fully connected hidden layers. ANN has many hyperparameters to tune. In our experiments, we use parameters in table 2.4.

Parameters	Values	Description
1th Hidden nodes	$Range(10, 100)$	Neurons in first hidden layer
2th Hidden nodes	$Range(5, 50)$	Neurons in second hidden layer
Learning Rate	$0.0001, 0.001, 0.01, 0.1$	Regularization parameter
Activation Function	Relu, logistic	Hidden layers activation function
Optimizer	Adam	Optimization algorithm
Loss Function	MAE, Log	Model training loss function
Shuffle	True	Sample order randomization

Table 2.4: Parameters of ANN

LSTM

Long Short Term Memory was first introduced by Hochreiter and Schmidhuber (Hochreiter and Schmidhuber, 1997). LSTM was designed to solve time series problems since it inherited the characteristics of the Recurrent Neural Network, which extracted sequential information by internal loops in its layer. This learning ability allows the LSTM model to better approximate time-series problems like human activation recognition, language recognition, and stock movement prediction. LSTM contains similar parameters to ANN. It is the dimension of the input sequence that makes LSTM unique from other non-time series models. We have our experiment settings for LSTM presented in Table 2.5. For the LSTM model, overfitting is a potential issue that needs particular attention. Early stopping and drop-out layers are good strategies to avoid overfitting. LSTM is the only time-series model

in our experiments. We will compare it with the other two non-time series models in the results section.

Parameters	Values	Description
LSTM nodes	$Range(10, 100)$	Neurons in first hidden layer
Dense nodes	$Range(5, 50)$	Neurons in second hidden layer
Learning Rate	$0.0001, 0.001, 0.01, 0.1$	Regularization parameter
Activation Function	Relu, Logistic	Hidden layers activation function
Optimizer	Adam	Optimization algorithm
Loss Function	MAE, Log	Model training loss function
Shuffle	False	Sample order randomization

Table 2.5: Parameters of LSTM

2.4.3 Trading Simulation

To illustrate our prediction-based portfolio optimization model, we present trading simulation experiments by comparing three different portfolio allocation models over 125 trading days in a rolling horizon fashion. For the purposes of this experiments we will ignore transaction fees. For generating scenarios in the CVaR optimization problems we will use 180 previous trading days. Trading simulation experiment settings are listed in 2.6.

Parameters	Values	Description
Time Horizon	180 days	CVaR calculation historical scenarios
Test Range	125 days	Trading dates between 2020-01-01 and 2020-06-30
Target Return	0.8 percentile	Return constraint

Table 2.6: CVaR Model Settings

The trading is performed with a rolling horizon. For each day, 180 previous trading days are used as scenarios for constructing the three portfolios. Next, each portfolio value is evaluated and recorded, before moving 1 day ahead, updating the 180 scenarios by excluding the oldest trading day and including the most recent trading outcomes.

Our first trading simulation is performed on a standard CVaR portfolio optimization model using settings in Table 2.6. The second strategy is to evenly allocate budgets to those stocks that are predicted to be profitable each day. These are two benchmark models in our experiments for comparison purposes. Finally, in our purposed approach, budgets are allocated according to the result produced by model (2.2) per trading day. Readers should note that all target stocks can have decreasing movement predictions on some days. This will lead to empty inputs for Prediction+EW and Prediction+CVaR models. As mentioned earlier, in this case, we withdraw funds from the market and continue to the next period. Our trading simulation is described in the pseudo code below.

Algorithm 1 Prediction+CVaR Trading Simulation

Require: $initialFund \geq 0$
Ensure: $i = testRange$
Ensure: $j = stock_j$
  $currentFund \leftarrow initialFund$
  $p_{ij} \leftarrow predictionResults$
  $c_{ij} \leftarrow stockClosePrice$
  $profitHist = [\,]$
  while $i \geq 0$ **do**
    if p_{ij} is Not None **then** ▷ Some stocks are going up
      $weight_{ij} = CVaR(p_{ij}X)$ ▷ CVaR model in equations 2.2
      $currentFund = currentFund \cdot \sum_{j=1}^{n} weight_{ij} \cdot (1 + pct\_change(c_{i+1j}))$
    else if p_{ij} is None **then** ▷ Depression day
      $currentFund = currentFund$
    end if
    $profitHist.append(currentFund)$
    $i \leftarrow i - 1$
  end while

2.5 Results

Prediction and trading simulation results are presented in this section. We first apply evaluation metrics to measure the prediction model performance. Then, trading simulation results using all interesting stocks are presented to research our model performance under

default return constraints. Finally, random sub-selections using 10, 15, and 25 stocks are compared to evaluate the robustness of the approach.

2.5.1 Prediction Results

In the model implementation, we use the Scikit-learn machine learning library to build our SVM and Neural Network models. LSTM models are built using the TensorFlow library. We will train all models for each stock. By comparing their evaluation metrics, we found that the Deep Neural network outperforms all other models. And its results are presented in table 2.7.

Table 2.7: DNN Prediction Results

Stocks	**Precision**	**Class 0 Precision**	**Class 1 Precision**	**Support**
AAPL	70	0.65	0.73	58, 67
ABC	0.71	0.69	0.73	59, 66
AAPL	0.70	0.65	0.73	58, 67
ABC	0.71	0.69	0.73	59, 66
AMZN	0.66	0.6	0.7	51, 74
ANTM	0.64	0.62	0.66	63, 62
BAC	0.65	0.69	0.61	65, 60
BRK-A	0.7	0.75	0.65	63, 62
CAH	0.64	0.63	0.65	62, 63
CI	0.66	0.71	0.61	62, 63
CMCSA	0.67	0.60	0.72	57, 68
COST	0.67	0.66	0.69	62, 63
CVS	0.63	0.66	0.61	66, 59

Continued on next page

Table 2.7 – continued from previous page

Stocks	Precision	Class 0 Precision	Class 1 Precision	Support
CVX	0.69	0.71	0.67	67, 58
F	0.69	0.7	0.68	76, 49
FB	0.62	0.69	0.66	53, 72
FDX	0.67	0.65	0.68	61, 64
FNMA	0.69	0.7	0.67	72, 53
GM	0.67	0.74	0.59	69, 56
GOOGL	0.68	0.55	0.76	50, 75
HD	0.66	0.68	0.64	60, 65
JNJ	0.63	0.65	0.60	68, 57
JPM	0.69	0.76	0.61	67, 58
KO	0.63	0.65	0.60	64, 61
KR	0.62	0.62	0.61	60, 65
MCK	0.62	0.62	0.61	62, 63
MPC	0.64	0.67	0.61	67, 58
MSFT	0.68	0.65	0.69	52, 73
PFE	0.61	0.65	0.55	71, 54
PSX	0.63	0.70	0.55	69, 56
T	0.61	0.57	0.64	62, 63
TM	0.63	0.64	0.62	69, 56
TSLA	0.67	0.68	0.66	56, 69
UNH	0.68	0.67	0.69	62, 63
VZ	0.7	0.76	0.63	65, 60
WBA	0.62	0.65	0.59	65, 60

Continued on next page

Table 2.7 – continued from previous page

Stocks	Precision	Class 0 Precision	Class 1 Precision	Support
WMT	0.7	0.77	0.61	69, 56
XOM	0.72	0.85	0.54	71, 54

We will use our best prediction model Deep Neural Network to predict 125 test date outcomes. These outcomes will be the coefficients p_j in our CVaR portfolio optimization model. Since prediction accuracy is directly correlated with prediction-based portfolio models. The accuracy along with F1-score and confusion metrics are key indicators among all evaluation metrics. In table 2.7, the lowest accuracy in our ANN model is 0.62. This model performance is consistent with the theory that ANN is capable to approximateselected a variety of problems regardless of their complexity. SVM fails to produce decent results. Probably, it is the feature space that influences the performance. Please note that we use the RBF kernel function to map existing feature space to a high dimensional space, which will produce unpredictable complex problems for SVM. Considering the complexity of our feature space and the high volatility of the stock market itself, SVM would not be our best choice. In our LSTM studies, we use 30 days' historical data as our time-series look-back period. The time-series information has limitations for LSTM neural network to learn the correlations. If we use a longer time series period, we will lose more samples for the model to learn and it is tedious. Another possible reason for the failure of the LSTM model is that our time-series data inevitably lose some continuity when we merge different data sources. LSTM is always sensitive to the quality of its time-series data. Data continuity could not be ignored.

2.5.2 Trading Simulation Results

This section presents all trading simulations using the proposed Prediction+CVaR portfolio risk optimization model, standard CVaR, and Prediction+EW models. For the first

trading simulation, all 36 selected stocks are used across the entire simulation period. Figure 2.2 presents the portfolio value (initial value set to 100k) evolution in time, as well as daily return distributions.

Note that, by design, the testing range contains the first phase of the Covid-19 pandemic and the corresponding financial market downturn. In addition to generally resulting in losses in most individual assets, it can be expected that the different performance of the market during this period (compared to the historical experience) poses a challenge to both prediction and diversification. First, observe that the pure CVaR model does not perform well. It is still affected by the market crash at the beginning of the pandemic, and then, while recovering, does not realize potential gains. On the other hand, it does exhibit the lowest volatility.

Both prediction-based models are susceptible to the initial market crash. At the same time, both are able to quickly recover by taking advantage of the predictions. It is worth noting that this result is somewhat surprising, given the relatively low accuracy of the prediction models reported in the previous section. Despite that, even pure prediction model is able to correctly identify sufficiently many growing assets to enable overall gains.

Comparing the two prediction models, the main advantage of the proposed model is the reduced volatility (see the boxplot in Figure 2.2). It achieves significantly lower variance, particularly avoiding the most extreme losses, while displaying higher average returns, and higher resulting portfolio value.

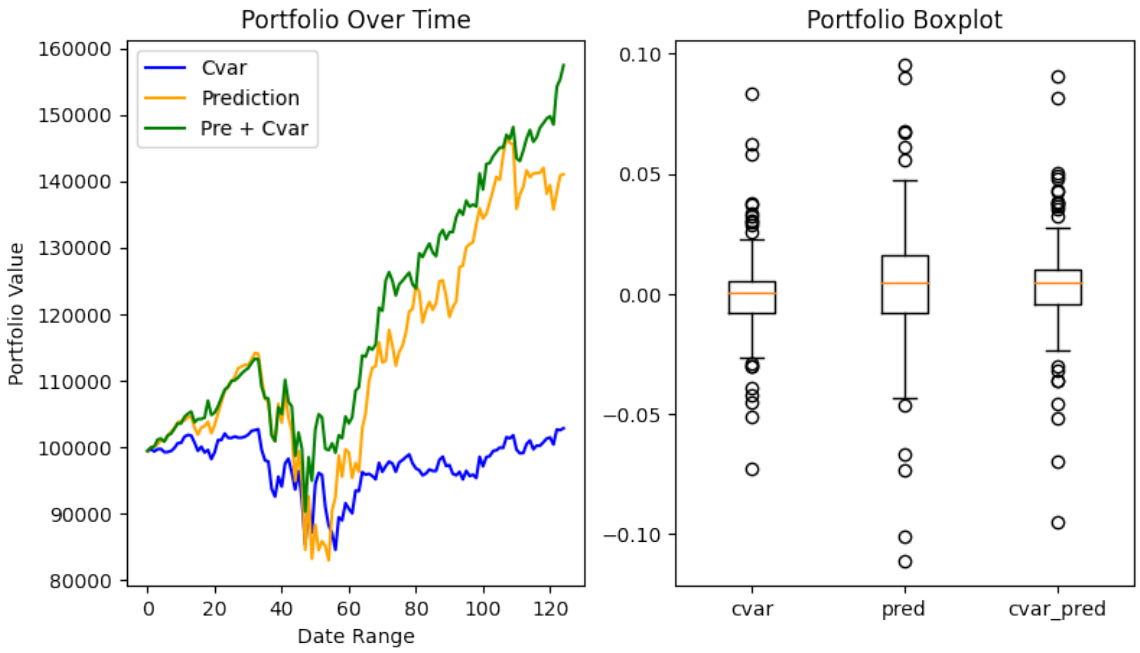


Figure 2.2: 36 Stocks Portfolio Trading Simulation

Next, we sample subsets of the assets (15, 20 or 25) in order to test model robustness. We repeat sampling 35 times for each sample size. Figures 2.3, 2.4 and 2.5 present representative examples for each sample size. Overall, in most cases, the same conclusions apply. We then conduct statistical analysis to further evaluate the conclusions.

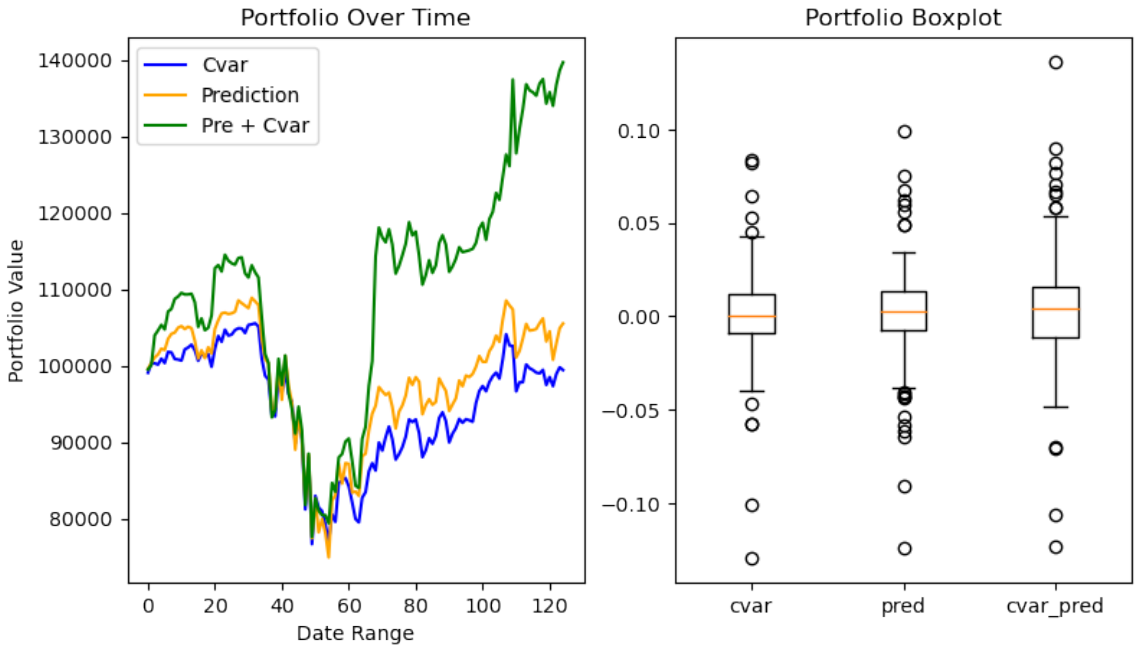


Figure 2.3: Random 15 out of 36 Stocks Portfolio Trading Simulation

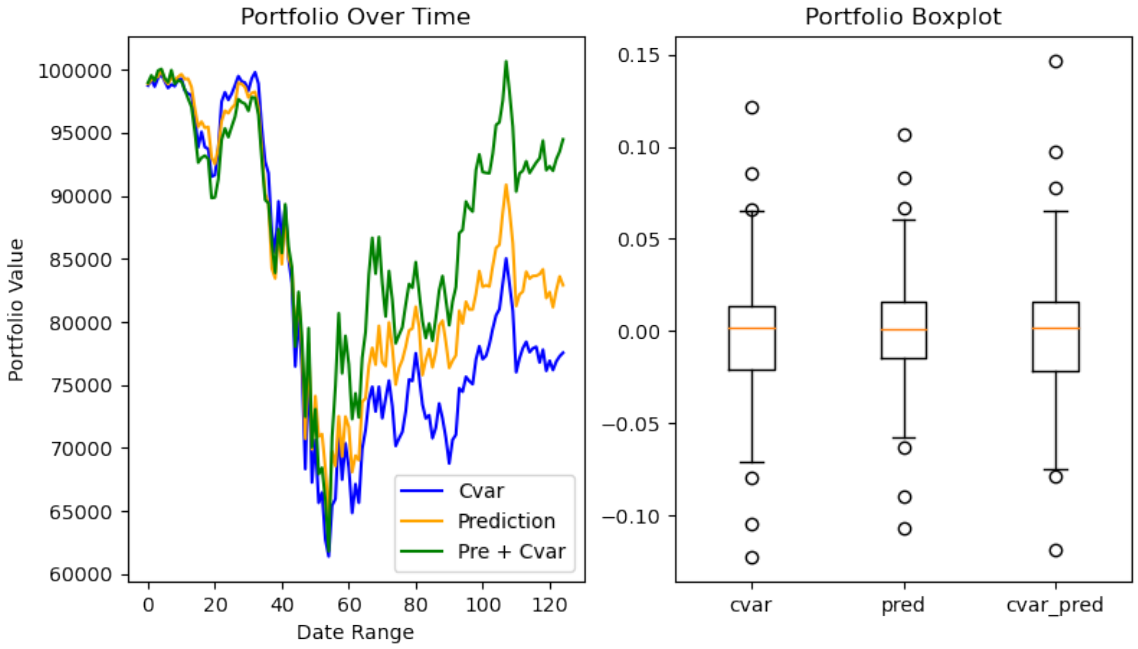


Figure 2.4: Random 20 out of 36 Stocks Portfolio Trading Simulation

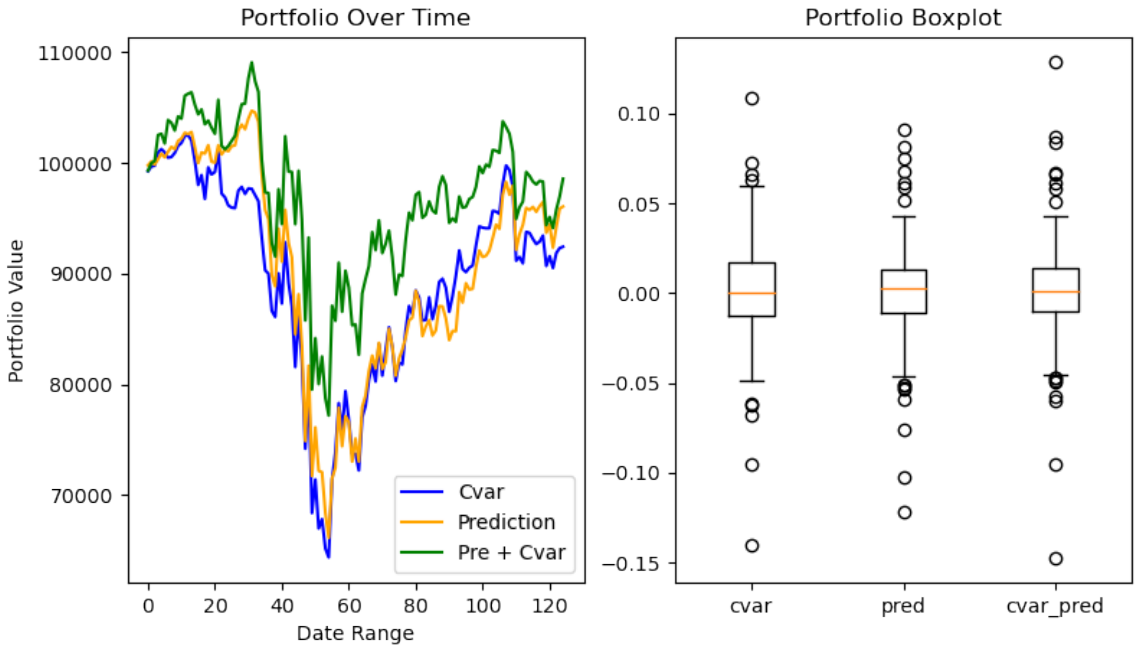


Figure 2.5: Random 25 out of 36 Stocks Portfolio Trading Simulation

The T-test is a robust technique for the assumption of normality and homogeneity of variance when the sample size is large(≥ 30). One-tailed T-tests are performed according to our alternative hypothesis H_a, which is that the expectation of the Prediction+CVaR model's return is greater than the other two baseline models. And indeed, the results in table 2.8 provide supporting evidence for our alternative hypothesis. All our t-test results support that the Prediction+CVaR model has the most profitable return expectation with P-values greater than 0.05. While reader should know that all the results produced by the CVaR model are constrained by our default return constraint in equation 2.2. Consequently, the return distribution may lose normality due to the power of return constraints. Thus, all returns from random subset selections will not be fully independent. It is possible to argue that the T-test is not appropriate for this situation. However, as we announced before, T-test studies are intended to robustly research model performance, and the results indicate our proposed model is profitable in returns.

Sample Size	Methods	T-statistic	P-Value
15	CVaR	1.7520	0.9593
	Pred	1.7757	0.9613
	Pred+CVaR	Not Apply	Not Apply
20	CVaR	-0.6828	0.2477
	Pred	-1.0477	0.1480
	Pred+CVaR	Not Apply	Not Apply
25	CVaR	-1.2544	0.1055
	Pred	-1.1912	0.1174
	Pred+CVaR	Not Apply	Not Apply
Note: confidence level = 0.95			

Table 2.8: Trading Simulation Results

Investors usually hedge portfolio risk by splitting their budget into multiple assets. To verify our model results with this strategy, we conduct another set of T-tests by comparing the daily average returns of the Prediction+CVaR model using random selection in sizes 15, 20, and 25 stocks. We assume that the model will produce more attractive results if more assets are included in building the portfolio. The results in table 2.9 are consistent with our expectation that the returns by using random 20 and 25 stocks are statistically higher than using 15 random stocks.

Sample Size	T-statistic	P-Value
15 vs 20	0.7221	0.7637
15 vs 25	-0.9089	0.1833
Note: Confident level 0.95		

Table 2.9: Pred+CVaR Model Outcomes T-test Using Multiple Sizes

2.6 Conclusions and Future Work

This paper applies three machine learning classification models to build a prediction-based portfolio risk optimization model. We collect four open-source data sets and some additional indicators to generate our feature spaces. SVM is designed as our baseline model.

We use time series and non-time series models in our prediction progress, DNN and LSTM. Both models are frequently used in deep learning neural networks to approximate stock market behaviors. All three models focus on binary classification for stock movements. And prediction outcomes are compared through three evaluation metrics. Then, prediction results are plugged into the optimization model as the coefficient described in model 2.2. To show its superiority, this proposed model is compared with two other baseline models, CVaR and Predition+EW.

In our prediction models, outputs are evaluated using accuracy F1 scores. We pick the best model to generalize our prediction outcomes. Through experiments, we find DNN has better compatibility with our collected input features. In addition, DNN achieves the best learning abilities with the lowest prediction error. Consequently, DNN prediction results are further fed into optimization models to build portfolios. In trading simulation experiments, we compared three portfolio allocation strategies, CVaR, Prediction+EW, and Prediction+CVaR. The first trading simulation uses all 36 target stocks over the entire trading simulation period. The result shows that our proposed model produces the best profit. To better study our model, we perform random subset selections in sizes 15, 20, and 25 among all 36 target stocks. Our trading simulations indicate that our model performance remains tractable and consistent in terms of risk and return management.

This research has limitations and can be extended by considering a more realistic prediction-based portfolio risk optimization model. First, investors certainly know that transaction fees can have crucial impacts on portfolios. It is more realistic if transaction fees can be involved. Second, to produce decent prediction results, powerful machine learning models and quality data sources are always highly preferred. For example, data continuity needs to be emphasized for our LSTM model. Last but not least, stock market prediction is still a challenging problem. In this paper, the prediction generalization accuracy still has research opportunities to improve. From a quantitative trading analysis perspective,

prediction accuracy can be modeled as another risk factor. Indeed, artificial intelligence and risk management models have more potential to help decision-makers to seek better solutions.

Chapter 3

Probability Estimated Portfolio Risk Optimization Through a Prescriptive Multi-Objective Approach

3.1 Introduction

Portfolio risk optimization and financial forecasting problems are two data-driven decision-making techniques that aim to address similar problems. Portfolio optimization theory heavily emphasizes the impact of historical data in defining risk and returns to enforce decisions to improve future performance, primarily through diversification. Unlike machine learning, portfolio models usually do not explicitly model actual future outcomes when making decisions. In contrast, financial forecasting studies establish approaches to learn the target patterns from historical data as a way to describe probable future outcomes. Both streams of literature have achieved significant results. However, good prediction results are different from optimal decisions and vice versa. Hence, it is important to consider the best approaches to combine the two techniques. Nowadays, data-driven decision-making based on forecasting models increasingly draws attention in the research community. The emerging field is often referred to as contextual optimization, and a thorough review can be found in (Sadana et al., 2023). A general idea of combining ML and optimization, one possible consideration is that given the prediction, an approximation of the random variable can be modeled as an optimization problem,

$$\hat{Z}_N(x) \in \arg\min_{z \in \mathcal{Z}} C(z, \hat{Y}_N(x)) \tag{3.1}$$

where $\hat{Z}_N$ is the decision vector, C is the cost function, $\mathcal{Z}$ is the feasible region and $\hat{Y}_N(x))$ represents the predicted outcome given data x.

Thus, further research can be developed by modeling risk-return as a prediction-based stochastic optimization framework. However, since ML does not address optimal decision-making under uncertainty, it is unclear how to make a good decision from an existing prediction. A good decision must consider the uncertainty while balancing it with hedging against both intrinsic uncertainty and prediction error. For example, in portfolio optimization, solving 3.1 based on historical samples $\{y_1, y_2, ..., y_n\}$ but without market data in specifying x would generally lead to conservative and inconsistent solutions. In addition, the waste of good market data is even more unacceptable.

In the first chapter, the proposed Machine Learning Informed Optimization of the CVaR model outperforms the standard CVaR and EW-prediction models. Note though, that it is constructed to rely on binary movement prediction only, i.e., at the optimization step, the only piece of prediction information used is a 0-1 forecast for each asset. This substantially limits possible ways that prediction can be incorporated into decision making.

A class of machine learning models that have been receiving an increasing amount of attention recently are models that provide both a prediction and estimation of the probability of the correct prediction. Note that, in general, it is not the same as prediction accuracy. Such models can either be trained specifically for this task, or a separate calibration procedure can be constructed.

The availability of such models opens the door to more ways to incorporate predictions in optimization. Specifically, since stochastic optimization explicitly relies on scenario probabilities, forecast probability estimates can be employed in such problems. In this chapter, we consider ways to enable such an approach.

This research targets the limitations and challenges by focusing on the following contributions.

1. We develop and compare two machine learning algorithms to estimate the probability of short-term market movements for 36 U.S. stocks, by calibrating the predicted probability to match the true probability of market behavior.

2. We propose two probability-calibrated CVaR risk optimization models to optimize the risk and return for the stock portfolio.

3. Our proposed risk optimization model is compared in terms of returns with the standard CVaR optimization model and Equal-Weight prediction model to ensure its performance.

3.2 Literature Review

3.2.1 Stock Market Prediction Models

The financial forecasting tasks represent a class of problems with large amounts of historical data that we can train, validate, and test deep learning approaches on. There has been a significant amount of literature on using deep learning models to predict stock prices. Some literature (e.g. Xu and Cohen, 2018; Chen et al., 2023) have found the existence of nonlinear correlations which explain the predictability of future stock prices. Many sophisticated data mining and deep learning models have been built combining macroeconomic data and auxiliary market information, including financial news and media sentiments (e.g. Bollen et al., 2011; Nguyen et al., 2015; Xu and Cohen, 2018). These works have indicated the path dependence between stock price movement and selected market information.

From a financial application perspective, robust and high-accuracy prediction models are crucial in financial decision-making problems. Countless efforts have been made in academia and industry to tackle this problem. Our purpose is to consider novel techniques to use less perfect prediction models in optimization, which argues in our work, that we rely on some of the existing algorithms, including DNN, LSTM, and BNN.

Instead of modeling stock movement prediction as a binary classification problem, as has been attempted in the previous chapter, a reliable probability estimation of the market behavior is crucial to many real-world financial applications with inherent uncertainty. The problem of predicting probabilities should represent the true correctness likelihood of all possible events (Niculescu-Mizil and Caruana, 2005). Good probability confidence estimation

provides more valuable information to approximate the ground truth probability using empirical data and establishing user trustworthiness.

Deep learning can estimate probabilities (Liu et al., 2021) by training models on observed outcomes of possible events. However, the ground-truth probabilities of the events are usually unknown. Therefore, the problem is analogous to the classification task with the difference that the output is the probabilities for each class rather than the specific labels. Deep networks for classification purposes often generate probabilities, such as the Softmax function outputs, which quantify the uncertainty for how likely the network is to classify the sample. This uncertainty quantification has been proven to be inaccurate (e.g. Murphy, 2012; Guo et al., 2017). For problems with high stochastic behaviors, like the stock market, it is further complicated to evaluate the model performance without access to the ground-truth probabilities.

To predict good probabilities, several methods have been developed to address the limitations, including temperature scaling for probability calibration (e.g. Guo et al., 2017; Liu et al., 2021). Temperature scaling is a robust single-parameter variant of Platt Scaling(Platt et al., 1999), which is also known as logistic calibration, a methodology employed to calibrate the output of binary classifiers with the objective of estimating probabilities. Platt scaling serves as a post-processing procedure designed to transform these decision scores into probability estimates. The basic idea behind Platt scaling is to fit a logistic regression model to the decision scores produced by a binary classifier. The logistic regression model takes the form of the sigmoid function (the logistic function), which maps values to the range 0 to 1, representing probabilities. As temperature scaling is developed upon Platt Scaling, it extends the calibration technique to Deep Neural Networks classification. One major advantage is that temperature scaling does not affect the prediction accuracy, which means the binary classification results will remain the same. In the case of calibrating the binary classification neural network, the variation of Plat Scaling is to use a single scalar parameter $T > 0$ for all classes. Given the output logits (not Softmax outputs), the new confidence prediction is

quantified in the equation 3.2.

$$\hat{q}_i = \max_k \ \sigma_{SM}(z_i/T)^k \tag{3.2}$$

T is referred to as the temperature and is optimized with respect to loss functions using validation data. Because of the setting, Temperature Scaling is a robust method that can calibrate almost all neural networks.

Another consideration in probability calibration is to measure the performance of a probability calibration model. It is crucial to assess how well it converts the model's raw scores (logits) into well-calibrated probability estimates. Several metrics and techniques can be used to evaluate the performance of a calibration model.

1. Reliability Diagram: A reliability diagram (Niculescu-Mizil and Caruana, 2005), also known as Probability Calibration Curve, is a graphical representation of observed vs. predicted probabilities. The curve shows how predicted probabilities change with the actual probabilities. A well-calibrated model will produce a curve that closely follows the diagonal line.

2. Expected Calibration Error(ECE): ECE is a scalar measure of calibration accuracy. It quantifies the difference between the observed accuracy and the predicted probability for each prediction bin. A lower ECE indicates better calibration.

3. Cross-Entropy Loss(CEL): Logistic loss, also known as log loss or cross-entropy loss, measures the divergence between predicted probabilities and the true classification labels. A well-calibrated model will have a lower log loss.

A number of studies discussing portfolio optimization in the presence of predictions exist in the literature, as reviewed in the previous chapter. Studies using machine learning techniques in particular (e.g. Freitas et al., 2009; Ma et al., 2020, 2021), are based on the underlying assumption that predicted future returns offer a more accurate estimation of expected returns compared to historical returns. Naturally, these methods neglect the

uncertain nature of the problem. This limitation is caused by the deterministic nature of the learning models, which ignores the uncertainty in prediction results. In contrast, probabilistic models (Abdar et al., 2021) provide prediction estimating the posterior probability as a form of distribution. The problem uncertainty can be signified using the predicted probabilities.

Hence, probabilistic portfolio models can optimize the risk and return objectives by considering the future risk and return simultaneously. This optimization problem can be summarized as follows:

$$\begin{aligned} \min \quad & \lambda \hat{R}(X) - (1-\lambda) \sum_{i=1}^{n} x_i \hat{E}(X) \\ \text{s.t.} \quad & \sum_{i}^{n} x_i = 1 \\ & 0 \leq x_i \leq 1 \end{aligned} \tag{3.3}$$

where x_i denotes the weights of portfolio assets in total of n portfolio assets; $\hat{R}(X)$ is some risk measure of the portfolio given X with prediction, and $\hat{E}(X)$ is the future expected return given the prediction. λ is a hyper-parameter to balance the risk and return objective. In this work, we propose a probabilistic portfolio optimization framework that focuses on using prediction to quantify both $\hat{E}(X)$ and $\hat{R}(X)$ for the optimization procedure in approaching the optimal allocation.

3.3 Method

In this work, we propose a three-phase probability-calibrated portfolio risk-return optimization model. In the first phase, a traditional (deep-)learning prediction model is constructed. Next, a calibration procedure is applied to obtain probability estimates. Finally, an optimization model is proposed to employ the resulting probabilistic prediction model to construct a portfolio. Next, we discuss each step in detail.

3.3.1 Prediction Models

We implement deep learning algorithms to produce the logits in predicting one-day stock movements. Formally speaking, in machine learning, logits refer to the raw, unnormalized predictions generated by most Neural Network classification models before applying any activation function or converting them into probabilities. Instead of constructing sophisticated prediction models to attain high accuracy, this study focuses on utilizing two well-established deep learning Neural Networks for a more straightforward approach. In the context of stock market movement predictions, DNN and LSTM models have been widely applied to the problem, with the consideration of both the non-time series and time series characteristics inherent in the problem. Logits can be easily derived from both DNN and LSTM, where the goal is to assign an input to one of several predefined classes. Logits can be interpreted as a measure of the model's confidence in each class prediction. Higher logit indicates a stronger belief in a particular class, while lower logit suggests less confidence. After obtaining logits, they are often transformed into probabilities using a Softmax function, which performs normalization and converts them into values between 0 and 1 that sum up to 1. From a probability point of view, the logits are not the predicted probability. Because of the calibration issues, for example, a model may consistently overestimate or underestimate the sample probabilities that are away from their true probabilities. Overconfidence can lead to incorrect predictions, while underconfidence can result in missed opportunities. Probability calibration aims to address these issues and improve the reliability of predicted probabilities.

To calibrate the probabilities, we consider Temperature Scaling(Guo et al., 2017), a universal probability calibration technique for Neural Networks. In our classification model, the Softmax function is used to convert the model's logits into probabilities. The Softmax function applies the exponential function to each score and then normalizes them by dividing by the sum of all exponentiated scores. The Softmax function is defined as follows:

$$P(class_i) = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}}, \tag{3.4}$$

where $P(class_i)$ is the probability of $classi$, z_i is the logit for class i, T is the temperature parameter, and the sum is taken over all classes. When the temperature T is set to 1, the Softmax function behaves normally. However, by increasing or decreasing the temperature, we can control the concentration or spread of the probabilities. Higher temperatures(> 1) cause the probabilities to be more uniform, while lower temperatures(< 1) make the probabilities more peaky, with the highest probability dominating. To calibrate the probabilities output by a model, temperature parameter T is optimized with respect to Negative Log Loss to match the model's confidence with the observed accuracy on a validation set. This technique assumes that the model's confidence, as reflected in the probabilities, should be well-calibrated and correspond to its accuracy. In other words, Temperature Scaling will calibrate the Neural Network to be less confident without changing the prediction accuracy regarding classification. To evaluate the calibration performance, we mainly rely on two metrics, Cross Entropy Loss and Expected Calibration Error. They can be expressed in equation 3.5:

$$\begin{aligned} ECE &= \sum_{k=1}^{M} \frac{B_k}{N} |acc(B_k) - conf(B_k)| \\ CEL &= -\frac{1}{N} \sum_{i=1}^{N} [y_i log(\hat{P}(y_i)) + (1 - y_i) log(1 - \hat{P}(y_i))] \end{aligned} \tag{3.5}$$

where M is the number of bins or intervals. $B_k$is the set of samples within the $k - th$ bin. N is the total number of samples.$acc(B_k)$ is the accuracy within the $k - th$ bin. $conf(B_k)$ is the confidence (average predicted probability) within the $k - th$ bin. N is the number of labels/classes. y_i is the true labels. $\hat{P}y_i$ is the predicted probability for class y_i. We expect both loss values will be decreased after the calibration.

3.3.2 CVaR Bi-objective Portfolio Optimization

As before, in the optimization step, the goal is to select portfolio weights x_i such that the expected return is maximized, while CVaR of the negative returns is minimized. In the traditional setting, both of these values are evaluated based on historical values. Now,

suppose that in addition to historic returns r_{ij}, we also have access to values p_j^+ and p_j^- that give the probabilities that asset j increases or decreases in value during the next time period respectively. Accordingly, define as r_{ij}^+ and r_{ij}^- the historic (actually realized, not predicted) scenarios corresponding to positive and negative returns.

Consequently, we can interpret the predicted probabilities as weights that can be applied to the historical scenarios. In other words, each scenario r_{ij}^+ can be interpreted to occur with probability $\frac{p^+}{m}$, and each scenario in r_{ij}^- with probability $\frac{p^+}{m}$, where m is the total number of scenarios. This allows us to evaluate either of the two objectives of interest based on purely historic realizations (equal scenario probabilities), or according to predicted values (weighted scenario probabilities). This leads to two optimization problems.(3.6) and (3.7).

First formulation:

$$\begin{aligned}
\min \quad & \eta + \frac{1}{m(1-\alpha)} \sum_{j=1}^{m} \Big[- \sum_{i=1}^{n} x_i r_{ij} - \eta \Big]^+ \\
\max \quad & \sum_{i=1}^{n} \Big(p_i^+ \sum_{j=1}^{m} x_i r_{ij}^+ - p_i^- \sum_{j=1}^{m} x_i r_{ij}^- \Big) \\
\text{s.t.} \quad & \sum_{j=1}^{n} x_j = 1, \\
& 0 \le x_i \le 1, \quad i = 1, \ldots, n.
\end{aligned} \tag{3.6}$$

Second formulation:

$$\begin{aligned}
\min \quad & \eta + \frac{1}{m(1-\alpha)} \sum_{j=1}^{m} \Big[- p_i^+ \sum_{i=1}^{n} x_i r_{ij}^+ + p_i^- \sum_{i=1}^{n} x_i r_{ij}^- - \eta \Big]^+ \\
\max \quad & \sum_{i=1}^{n} \Big(p_i^+ \sum_{j=1}^{m} x_i r_{ij}^+ - p_i^- \sum_{j=1}^{m} x_i r_{ij}^- \Big) \\
\text{subject to} \quad & \sum_{j=1}^{n} x_j = 1, \\
& 0 \le x_i \le 1, \quad i = 1, \ldots, n.
\end{aligned} \tag{3.7}$$

In both formulations we consider the expected value objective to be evaluated based on the predicted probabilities. In the first formulation CVaR is evaluated traditionally, while in the second formulation, the predicted probabilities are used again. Note that the case where both objectives are evaluated traditionally is the ordinary mean-CVaR problem which will be used as the benchmark. Finally, we do not consider the remaining case (traditional expectation and predicted CVaR), since we surmise that the expected value objective can most benefit from the prediction, and the main purpose of the CVaR component is to ensure diversification.

Note that both formulations can be converted to a collection of single-objective linear programs in the same way as the similar problem in Chapter 2, and hence, the resulting problems can be solved with standard off-the-shelf linear solvers.

3.4 Experiments and Results

The experiments are designed to illustrate and verify the feasibility and performance of the proposed models. The new proposed model has challenging tasks that need to be addressed in the experiments. The market trend is a less challenging target than the actual price prediction, but it still requires careful consideration. Some possible factors for the challenge include data availability, data quality, model availability, highly stochastic market behavior, etc. To begin with our experiments, we first introduce the data acquisition process.

3.4.1 Data Specification

We prepare daily trading data for the same 36 US stocks as in the previous chapter and download all stock prices from Yahoo Finance from 2015/07/01 to 2023/5/19. In all 36 assets, 30 stocks are picked from the top of the Fortune 100 company list. The other 6 stocks are picked based on the authors' interests. As suggested by the literature (Weng et al., 2017), we adopt the approach of using the closing price of the next day relative to the close price the day before as the target label for each asset. To build our model, we utilize the

adjusted closing prices up until the current date as input features. Additionally, we calculate several widely used stock technical indicators that reflect price changes over time using open, high, low price, and volume. Literature suggests these stock indicators are informative and can benefit prediction accuracy. In this study, we focus on eight selected technical indicators, which are different from the features in Chapter 2. The indicators considered in this work are mainly concentrated on stock trends, momentum, volatility, and volume. In particular, these indicators are suggested to be informative to stock movement, which is consistent with the purpose of the prediction classification. A detailed summary is in Table 3.1.

Indicators	**Category**	**Description**
MACD	Trend	Reveals price change in strength and trends
PSAR	Trend	Indicates parabolic stop or reverse of current trend
Bollinger Bands	Volatility	Forms a range of prices for trading decisions
Stochastic Oscillator	Momentum	Overbought and oversold signals
Rate of Change	Momentum	Measures the percent change of the prices
On-Balance Volume	Volume	Volume flow to predict changes in stock price
Force Index	Volume	Indicates bull or bear market
MACD: Moving average convergence or divergence.		
PSAR: Parabolic Stop and Reverse.		

Table 3.1: Description of technical indicators

Due to the time series data quality and data availability consideration, we will only retrieve from Yahoo Finance in this work. We removed Google trends, Wikipedia Pageview, and Sentiment news from our consideration. Please note that the features can be either categorical or continuous. For categorical features, we employ embedding techniques to convert them into dense numeric vectors. On the other hand, we rescale all continuous features to a range between 0 and 1. This scaling is necessary because neural networks are generally challenging to train and can be sensitive to input scale (Glorot and Bengio, 2010). Furthermore, it is important to note that data preparation is performed individually for each stock since the prediction models are applied sequentially to one stock at a time in the experiments.

3.4.2 Prediction and Calibration Results

We first present prediction model configurations for both Deep Neural Network (DNN) and Long Short-Term Memory (LSTM) models, and the specific parameters utilized in the implementation are provided for reference. One of the model setting differences between DNN and LSTM is that we select validation sets for DNN randomly to ensure model performance. While using simple cross-validation would introduce bias in LSTM because of the missing values, one-step-ahead cross-validation instead overcomes the problem. Please note that each model is trained using the same model parameters for all stocks. It is possible to tune the parameters for each individual stock to achieve better prediction performance, but we choose not to consider this, since the primary objective of this work does not revolve around maximizing the prediction performance of the models employed. In order to optimize computational efficiency, uniform model settings are applied across all stocks in this study. The model settings are specified in table 3.2.

As mentioned before, all stocks are processed by each prediction model individually, and the models are evaluated by the CrossEntropy loss function. The best models are saved and calibrated using Temperature Scaling to obtain the probability estimates. As required for Temperature Scaling, all probabilities are calibrated using the same validation set as used in the training processes. It is crucial to note that the calibration assumption for this setting is the training, validation, and test sets are drawn from the same distribution. Similar to the prediction model settings, we apply universal calibration model settings presented in Table 3.3. Because our prediction model outputs the binary classification logits, we take only the positive class confidence in the following discussions for simplicity. Given a test sample x_i, the prediction model outputs a non-probabilistic output p_i, or the logits, the calibrated probability of corresponding positive class ($y_i = 1$) in terms of binary classification can be denoted as p_i^+. To illustrate the calibration results and performance, we compare the CrossEntropy loss and ECEloss (Guo et al., 2017) values before and after calibration in Tables 3.4 and 3.5.

Models	Parameters	Values
DNN	Hidden Nodes	32, 16, 32
	Learning Rate	0.001
	Activation Function	ReLU, Softmax
	Optimizer	Adam
	Loss Function	CrossEntropy
	Train Size	0.6
	Validation Size	0.2
	Test Size	0.2
	Batch Size	10
	Epoch	300
	Data Shuffle	True
LSTM	Hidden Nodes	16, 32
	Learning Rate	0.001
	Activation Function	ReLU, Softmax
	Optimizer	Adam
	Loss Function	CrossEntropy
	Train Size	0.6
	Validation Size	0.2
	Test Size	0.2
	Batch Size	10
	Window Size	10
	Epoch	200
	Data Shuffle	False

Table 3.2: Prediction Model Settings

Reliability diagram is the conventional way to evaluate probabilistic predictions. The calibration curve is essential for quantitative analysis, as it helps visualize the calibration of a model's predicted probabilities. To construct a calibration curve, a series of standard samples (fraction of positive class in each bin) with known concentrations are prepared, and their responses (mean predicted probability in each bin) are measured using the analytical instrument or method. In other words, each tested sample is placed into a bin based on predicted probability, and then for each bin the proportion of the samples that are correctly classified is calculated. The result is then plotted in the predicted vs realized probability coordinates. An ideal calibration curve is a straigh line from (0,0) to (1,1). It is important to note that the accuracy and reliability of the calibration curve depend on the quality of the

Parameters	Values
Number of Bins	15
Learning Rate	0.01
Epoch	50
Loss Function	CE, ECE
Initial Temperature	1.5

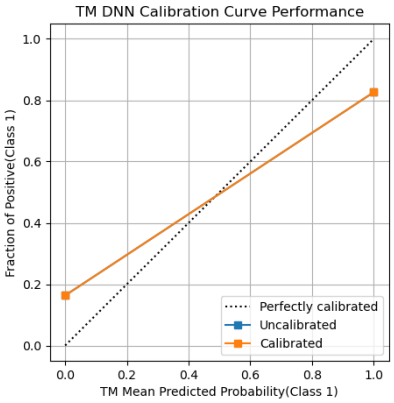
Table 3.3: Temperature Scaling Model Settings

standard solutions, the precision of the prediction model, and the proper statistical analysis of the data. Regularly performing calibration checks and including quality control samples can help ensure the accuracy of the results obtained from the calibration curve. Based on the validation accuracy, we illustrate the calibration performance by presenting the good, fair, and poor cases in Figure 3.1. More reliability diagrams can be found in Appendix B.

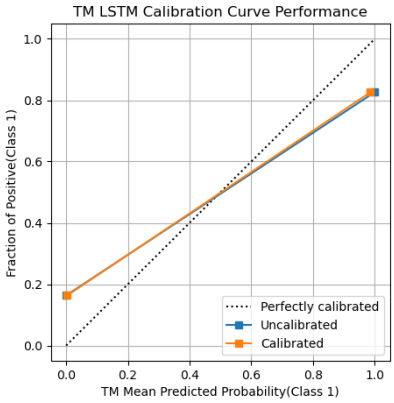
Based on our results, Temperature scaling is a powerful tool for improving the reliability and accuracy of predicted probabilities as both CrossEntropy Loss and Expected Calibration Error are decreased after calibration. From the reliability diagrams, we conclude that the calibrated probabilities are less confident as the line fits more closely to the diagonal line and narrows the range on the axis. Beyond the obvious loss decrease and improvement shown by reliability diagrams, the calibration performance is consistent and corresponds to prediction accuracy. Generally, we conclude that the predicted probabilities are improved with calibration and represent the best achievable true probability that will proceed to the optimization problem.

3.4.3 Trading Simulations: Case Study for The US Stocks

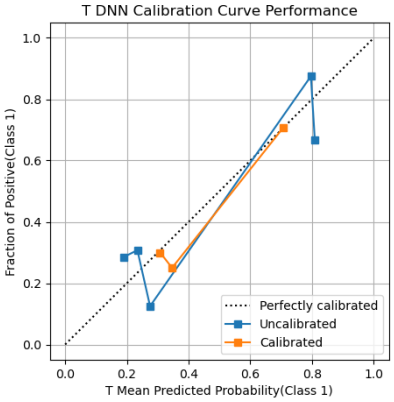
We present the trading simulation to verify the performance of the proposed models. Four predict-then-optimize models are compared in different settings. In the case of solving the bi-objective optimization problem where both objective functions are linear, we introduce a weight parameter λ. It is the parameter for a linear combination of both objective functions, expressed in Equation 3.8, where Obj_1 and Obj_2 represent the risk and return objective in



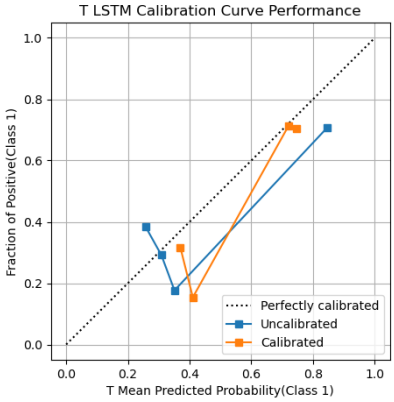
(a) Good Performance 1



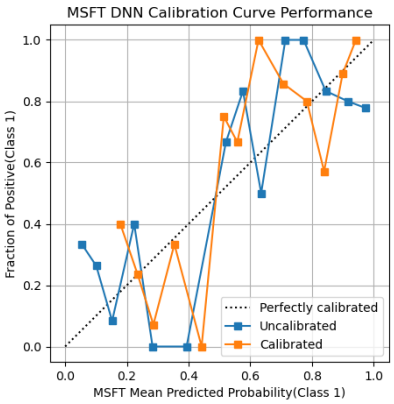
(b) Good Performance 2



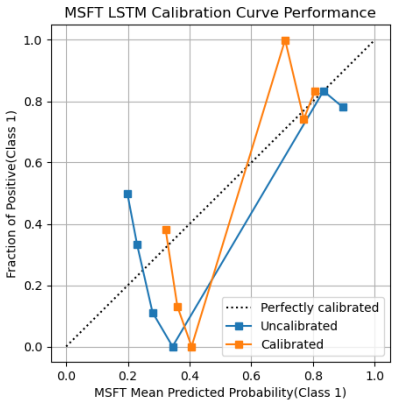
(c) Fair Performance 1



(d) Fair Performance 2



(e) Poor Performance 1



(f) Poor Performance 2

Figure 3.1: Reliability Diagram Examples of Calibration Performance

Stocks	CE Loss		ECE Loss		Validation Accuracy
	Before	After	Before	After	
AAPL	0.727	0.657	0.152	0.067	0.515
ABC	0.87	0.723	0.211	0.131	0.499
AMZN	1.006	0.75	0.232	0.142	0.493
BRK-A	0.762	0.637	0.188	0.082	0.562
CAH	0.759	0.686	0.175	0.105	0.534
CI	0.992	0.793	0.239	0.159	0.519
CMCSA	0.824	0.712	0.186	0.096	0.470
CNC	0.77	0.711	0.137	0.093	0.493
COST	0.864	0.739	0.224	0.155	0.538
CVS	0.898	0.745	0.23	0.13	0.485
CVX	0.894	0.703	0.198	0.101	0.510
ELV	1.83	1.163	0.284	0.238	0.501
F	0.897	0.703	0.21	0.141	0.501
GM	0.8	0.689	0.158	0.101	0.535
GOOGL	0.986	0.758	0.266	0.157	0.478
HD	1.308	0.875	0.286	0.206	0.530
JNJ	0.826	0.702	0.183	0.093	0.513
JPM	0.833	0.694	0.179	0.101	0.494
KO	2.295	1.349	0.33	0.27	0.493
KR	0.689	0.676	0.075	0.048	0.574
MCK	0.753	0.667	0.146	0.059	0.488
META	0.723	0.651	0.161	0.068	0.549
MPC	0.706	0.632	0.156	0.061	0.531
MSFT	1.814	1.119	0.274	0.226	0.509
NKE	0.858	0.706	0.155	0.135	0.507
PFE	0.865	0.728	0.197	0.127	0.485
PSX	0.902	0.721	0.224	0.123	0.533
T	0.68	0.643	0.112	0.057	0.449
TM	2.576	1.394	0.195	0.178	0.499
TSLA	1.383	0.956	0.211	0.166	0.503
UNH	0.742	0.691	0.151	0.074	0.480
VLO	0.852	0.715	0.168	0.101	0.476
VZ	0.695	0.652	0.116	0.085	0.520
WBA	3.241	1.803	0.377	0.316	0.488
WMT	0.693	0.656	0.118	0.061	0.511
XOM	7.163	3.767	0.36	0.341	0.535

Table 3.4: DNN Calibration Performance

the formulation, respectively. The linear combination converts the bi-objective optimization problem to a single-objective optimization problem such that the problem can be solved as

Stocks	CE Loss		ECE Loss		Validation Accuracy
	Before	After	Before	After	
AAPL	0.681	0.64	0.123	0.031	0.665
ABC	0.674	0.663	0.105	0.065	0.641
AMZN	0.741	0.676	0.161	0.061	0.624
BRK-A	0.734	0.659	0.163	0.065	0.657
CAH	0.711	0.683	0.116	0.079	0.603
CI	0.681	0.666	0.09	0.047	0.627
CMCSA	0.715	0.673	0.112	0.066	0.641
CNC	0.676	0.647	0.104	0.03	0.638
COST	0.699	0.673	0.095	0.035	0.611
CVS	0.69	0.668	0.108	0.065	0.635
CVX	0.758	0.632	0.169	0.095	0.719
ELV	0.662	0.645	0.089	0.02	0.654
F	0.722	0.623	0.161	0.083	0.684
GM	0.678	0.607	0.139	0.055	0.705
GOOGL	0.752	0.653	0.168	0.098	0.686
HD	0.701	0.674	0.113	0.036	0.597
JNJ	0.741	0.673	0.17	0.084	0.624
JPM	0.761	0.644	0.199	0.09	0.678
KO	0.682	0.639	0.132	0.037	0.659
KR	0.729	0.682	0.115	0.055	0.576
MCK	0.684	0.656	0.106	0.054	0.641
META	0.708	0.643	0.142	0.06	0.673
MPC	0.915	0.704	0.237	0.147	0.646
MSFT	0.768	0.672	0.164	0.089	0.659
NKE	0.717	0.621	0.154	0.064	0.705
PFE	0.768	0.66	0.17	0.092	0.670
PSX	0.714	0.648	0.159	0.056	0.657
T	0.7	0.655	0.135	0.068	0.659
TM	1.451	0.797	0.17	0.153	0.830
TSLA	0.697	0.645	0.132	0.059	0.662
UNH	0.695	0.66	0.109	0.037	0.619
VLO	0.675	0.64	0.123	0.064	0.670
VZ	0.738	0.689	0.157	0.07	0.576
WBA	0.641	0.625	0.082	0.013	0.676
WMT	1.07	0.778	0.231	0.152	0.624
XOM	0.863	0.654	0.185	0.091	0.727

Table 3.5: LSTM Calibration Performance

a single-objective LP. In general, λ can be interpreted as a representation of the investor's preference. If the investor is extremely risk averse, the λ value of 1 should be chosen.

Consequently, the model is only minimizing the risk, Obj_1 in Equation 3.8, to be conservative. On the contrary, if the decision maker is seeking more returns, the model should select a value close to 1 such that the return objective, Obj_2 in Equation 3.8, is more weighted in optimization. On the other hand, λ, a key parameter in the context of this study, may also be construed as an integral trustworthiness parameter, signifying the degree of confidence reposed by the decision maker towards the expected predicted returns. In other words, if the decision maker highly trusts the prediction, less weight shall be applied to the risk minimization objective, and more weight will be concentrated on the expected predicted returns maximization. Many research efforts have been conducted on the topic of machine learning trustworthiness, but this topic is beyond the scope of our work. Instead, we consider a set of 10 values for λ evenly distributed from 0 to 1.

$$\lambda \, Obj_1 + (1 - \lambda) \, Obj_2, \tag{3.8}$$

We will have four combinations of the proposed models: DNN with First Formulation (DNN+First), DNN with Second Formulation (DNN+Second), LSTM with First Formulation (LSTM+First), and LSTM with Second Formulation (LSTM+Second). Trading simulation settings are summarized in Table 3.6.

The trading simulation return trend lines with different lambda settings are shown in Figure 3.2 and Table 3.7. We found that the combination of LSTM and the second optimization formulation is generally more aggressive in seeking return and outperforms the other with the same value of λ. The second formulation trends to return higher profits across all λ. Regarding binary classification accuracy, LSTM dominates the DNN model in returns since LSTM has a better prediction performance reflected by the validation accuracy. From Figure 3.3, the first formulation turns out to be more conservative, i.e., results in lower volatility, which can be preferable. This result is in accordance with our expectations. Indeed, since we only use the prediction in the first formulation, reflected in the second objective function, to maximize the return, while in the second formulation, we define both risk and

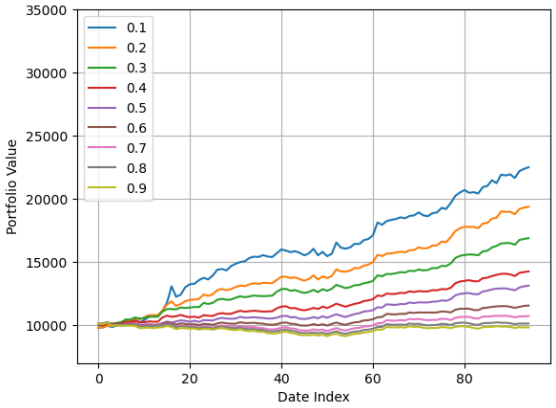
return using prediction. As we expect, the more prediction gets involved in the decision making, the more aggressive the model will tend to be when seeking returns. And the results unsurprisingly match our expectations.

Parameters	**Values**	**Description**
Time Horizon	180 days	CVaR calculation historical scenarios
Test Range	95 days	Trading dates between 2023-01-01 and 2020-05-19
Benchmarks	2	CVaR and CVaR + Binary Prediction
Proposed Models	4	First+DNN, First+LSTM, Sec+DNN, Sec+LSTM
λ	range(0, 1)	Weight parameter for objective functions

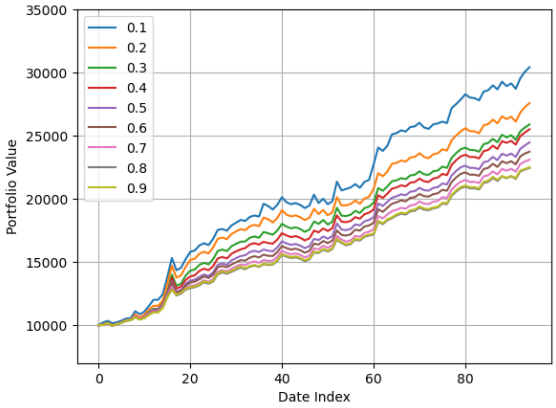
Table 3.6: Trading Simulation Model Settings

λ	**DNN+First**		**DNN+Second**		**LSTM+First**		**LSTM+Second**	
	CVaR	Return	CVaR	Return	CVaR	Return	CVaR	Return
0.1	4.290	0.911	0.393	1.222	4.047	0.957	0.670	1.295
0.2	3.131	0.735	0.255	1.114	2.925	0.698	0.335	1.169
0.3	2.573	0.564	0.158	1.041	2.470	0.532	0.206	1.080
0.4	2.161	0.377	0.105	1.022	2.094	0.383	0.125	0.990
0.5	1.923	0.284	0.073	0.975	1.878	0.256	0.078	0.964
0.6	1.777	0.147	0.048	0.941	1.767	0.127	0.053	0.968
0.7	1.708	0.068	0.035	0.910	1.707	0.054	0.040	0.952
0.8	1.663	0.000	0.025	0.878	1.664	0.013	0.032	0.937
0.9	1.649	-0.027	0.02	0.880	1.648	-0.031	0.029	0.924
Note: CVaR and Return are the average values in percentage.								

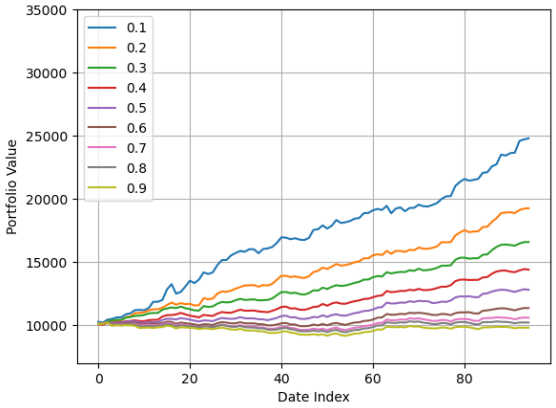
Table 3.7: Trading Simulation Numerical Results by λ



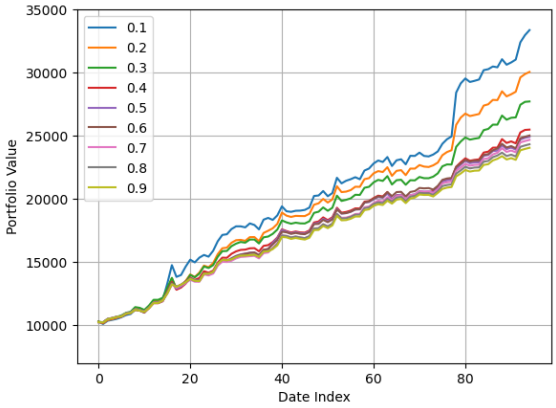
(a) DNN First Return Trend



(b) DNN Second Return Trend

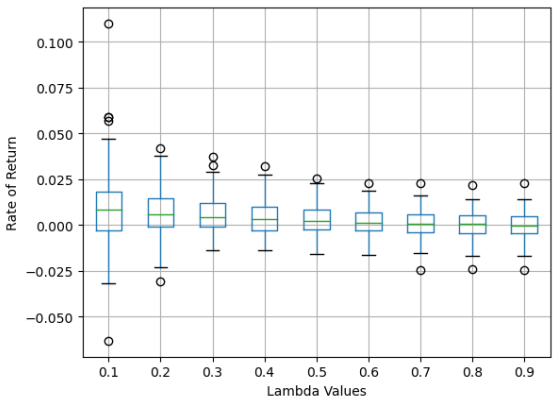


(c) LSTM First Return Trend

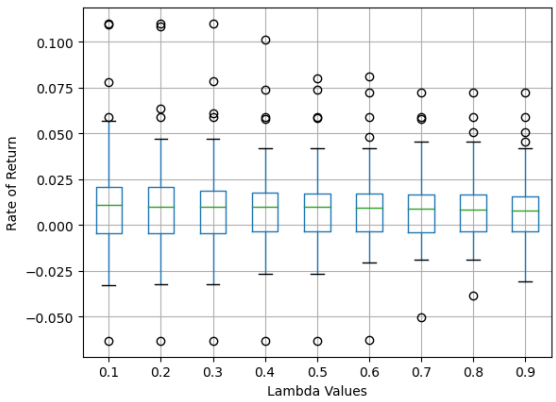


(d) LSTM Second Return Trend

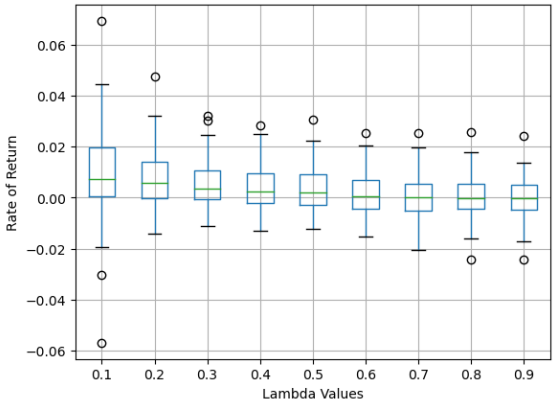
Figure 3.2: Trading Simulation Return Trends



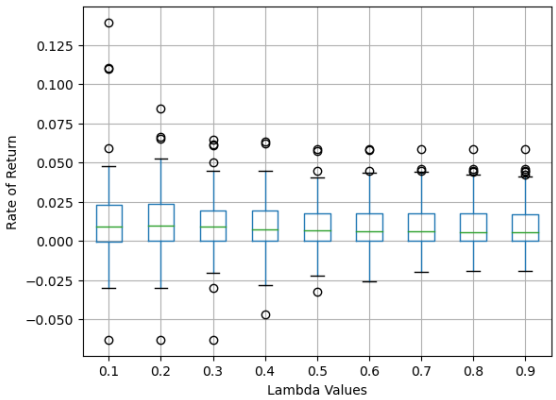
(a) DNN First Return Box Plot



(b) DNN Second Return Box Plot



(c) LSTM First Return Box Plot



(d) LSTM Second Return Box Plot

Figure 3.3: Trading Simulation Return Box Plot

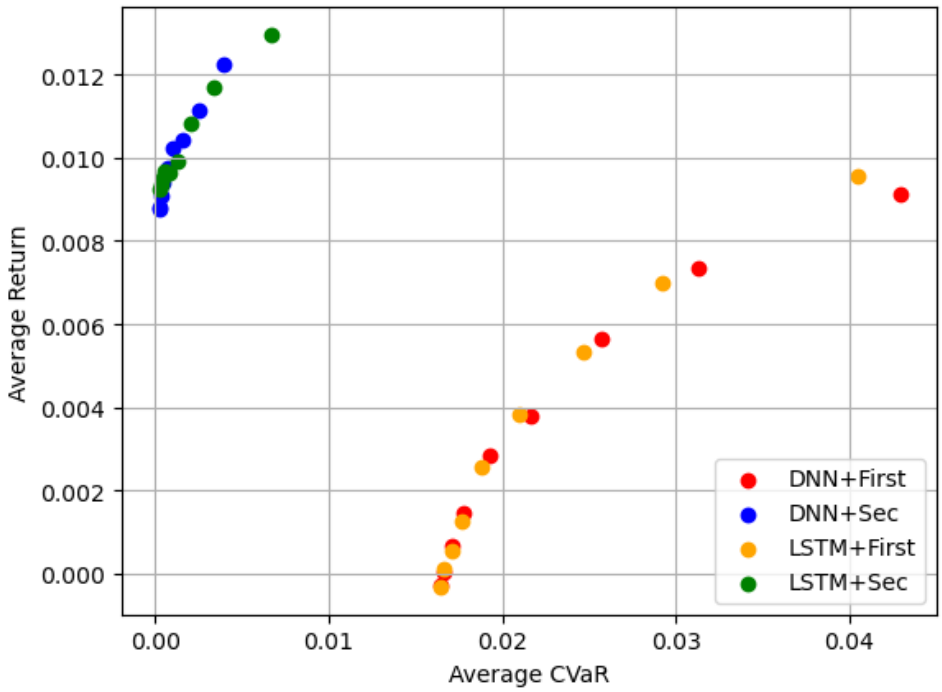


Figure 3.4: Frontiers for All Models

Further, in Modern Portfolio Theory, the Efficient Frontiers is a set of optimal solutions that find the highest possible return given a certain level of risk. The Efficient Frontiers for all our models are shown in Figure 3.4. Technically, we cannot directly compare the risk, measured by CVaR values, across two formulations since the risk is defined using different CVaR calculations. The result, shown in our efficient frontier, further illustrates how the decisions were made in accordance with risk and return trade-offs. Generally, the decision maker will encounter more risk when seeking more returns. Consistent with Table 3.7, the second formulation is more aggressive in seeking return, and the LSTM+Second formulation model outperforms the others.

Next, we consider two benchmarks to make cross-comparisons with the proposed models, including the classic CVaR optimization model and the model we presented in Chapter 2, referred to as CVaR with Binary Prediction. All four combinations of the proposed models are compared with the benchmarks. In our experiments, we choose the validation accuracy as a suitable choice for λ value. We train and select the best model according to the validation

accuracy. It already represents the confidence of trustworthiness. In this work, we consider the average validation accuracy of DNN and LSTM as reliable values for λ in solving the biobjective optimization problem. The trading simulations with benchmark comparison are shown in Figures 3.5 – 3.8.

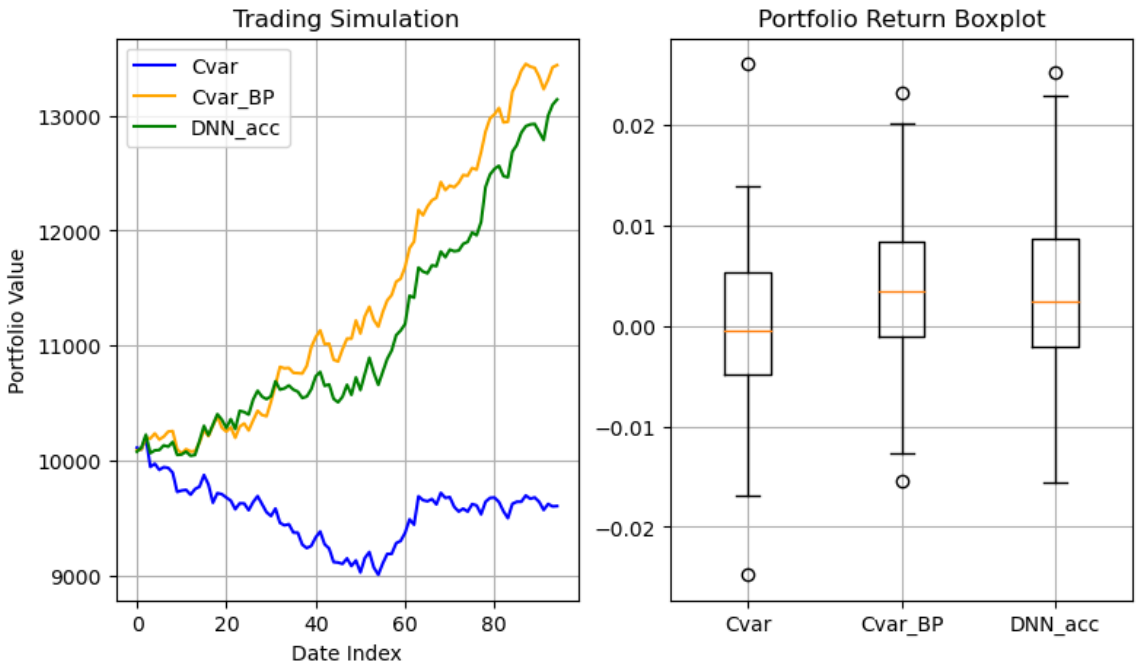


Figure 3.5: NN and First Formulation with Benchmarks

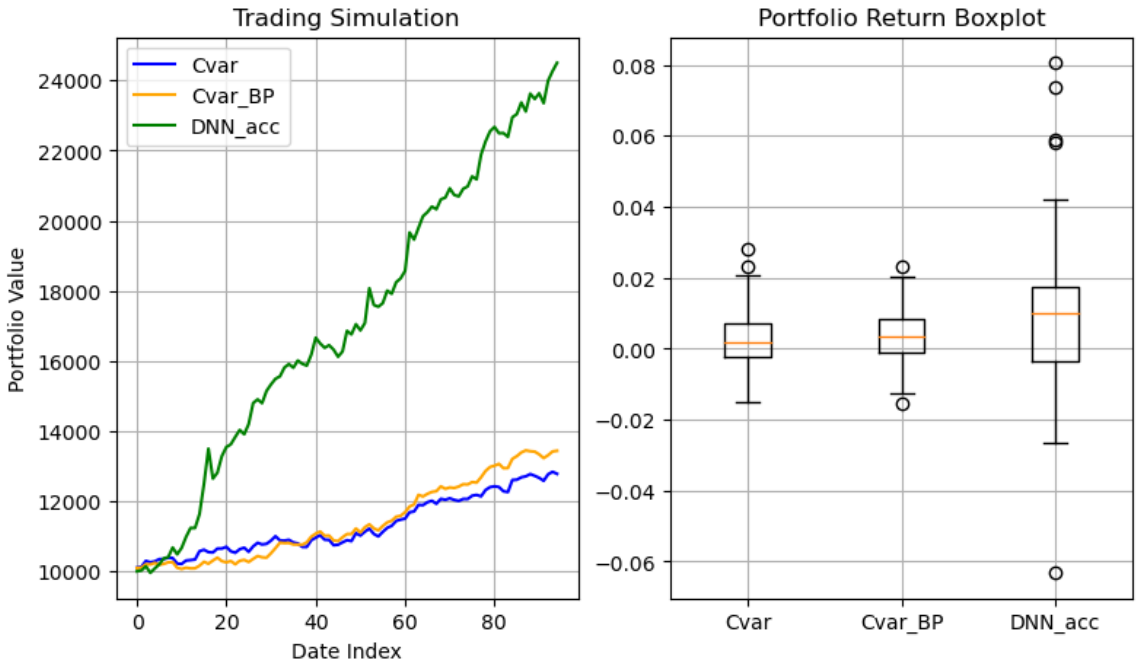


Figure 3.6: NN and Second Formulation with Benchmarks

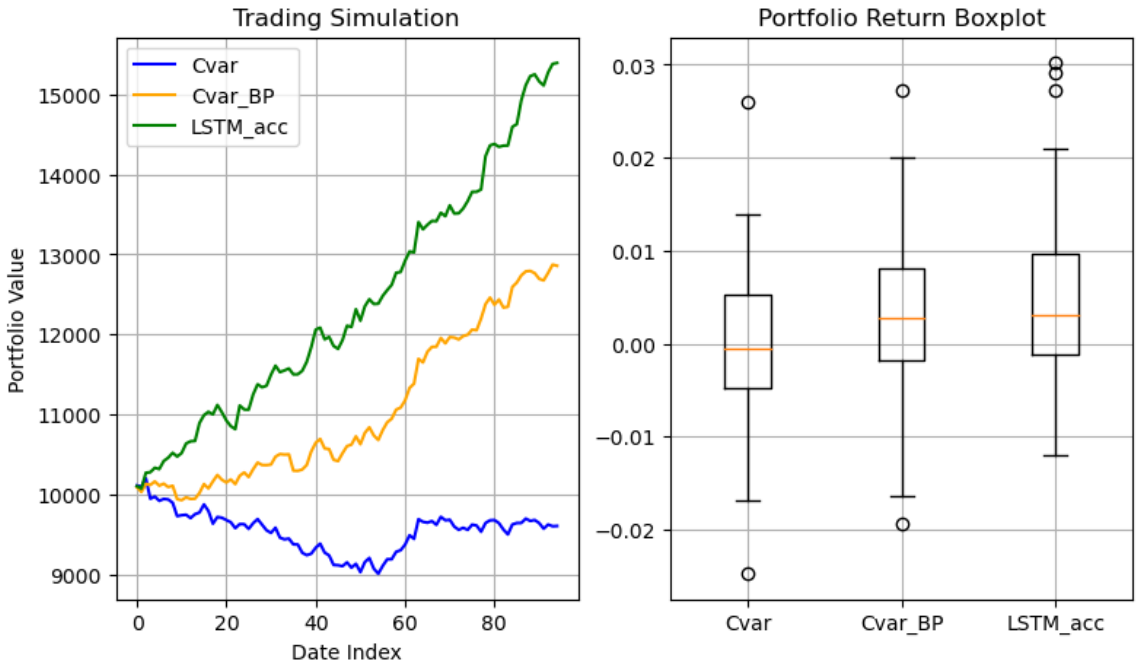


Figure 3.7: LSTM and First Formulation with Benchmarks

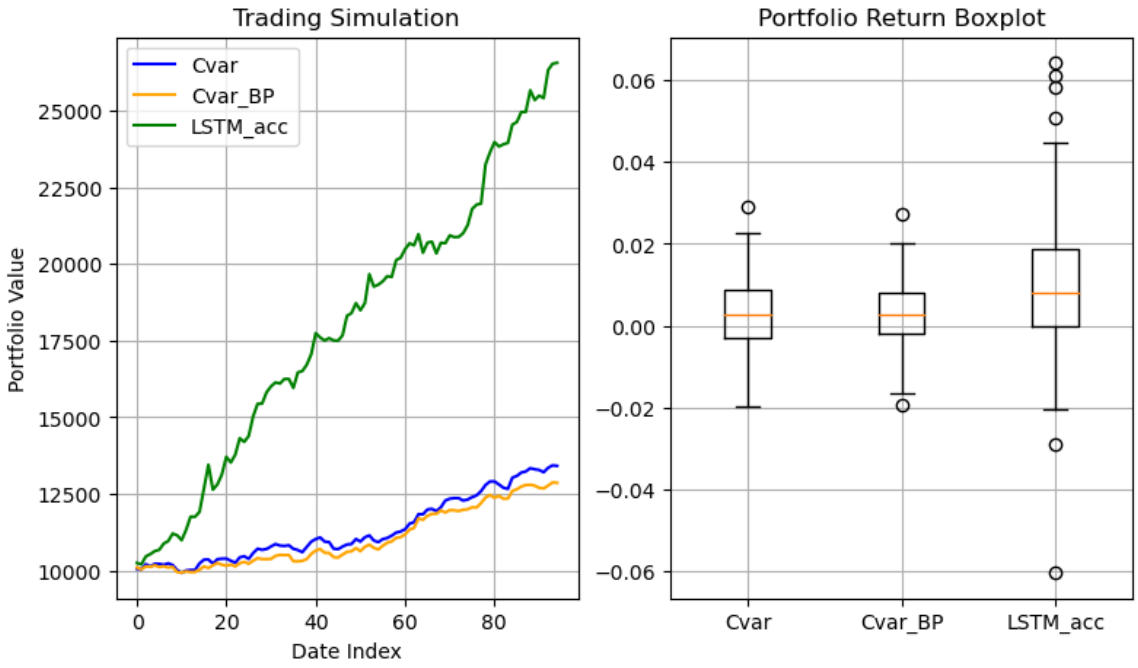


Figure 3.8: LSTM and Second Formulation with Benchmarks

Models	CVaR	Average Return	Return Std	Final Value
CVaR	1.645	0.05154	0.7808	$9,608.44
CVaR+DNN	1.98	0.3076	0.74417	$13,436.34
CVaR+LSTM	2.017	0.2608	0.762	$12,858.22
DNN+First	1.928	0.2854	0.8158	$13,138.19
DNN+Second	0.074	0.977	1.9760	$24,501.41
LSTM+First	2.298	0.4524	0.87545	$15,392.16
LSTM+Second	0.1690	1.037	1.9294	$26,568.40
Note: CVaR, Average Return, and Return Std are average values in percentage.				

Table 3.8: Trading Simulation with Benchmarks Comparison

Naturally, it cannot be considered a promising solution to our model, despite performing well on average, since it also displays very significant volatility. Based on the results we presented, the proposed models generally have good performance on both risk hedging and seeking return tasks. Since we define and calculate the risk differently, the CVaR value cannot be directly compared over all models. We evaluate the proposed model by concentrating on the average return and final portfolio value. Obviously, the combination of LSTM and the

second formulation with λ equals LSTM validation accuracy constantly produces the best profits.

3.5 Conclusions and Future Work

In this work, we implement two machine learning models and one calibration technique to estimate the probability of asset movements. We retrieve market data and construct a feature space to maintain a suitable time series format. We followed the general rule of thumb to use DNN as the baseline model to perform the prediction. Furthermore, LSTM is applied to get more prediction power and match the time series property of stock movement prediction. We selected the best model according to the validation accuracy. Then, the prediction models proceeded with the temperature scaling calibration neural network. The calibration results showed a consistent decrease in both CrossEntropy loss and Expected Calibration Error. The main contribution of this work is that we establish ties among probability prediction, calibration, and decision making. In the decision making process, the calibrated probability was introduced to the multi-objective optimization problem with the recommended weight parameter for solving it. It is a leap of faith to weight the objective functions using the validation accuracy from prediction. The results verified the performance of the proposed models in both risk hedging and return seeking tasks.

The proposed models have the potential to be applied in other applications, including but not limited to Supply Chain Management, Power Generation, and sustainable and resilient decision making applications – wherever both risk-averse optimization and machine learning predictions have been considered.

Meanwhile, we observe limitations in this research mainly in three aspects. First, we encounter many challenges in building a decent prediction model, including data availability, data quality, prediction accuracy issues, etc. Although many efforts have been put into improving the prediction accuracy, we could not get good prediction results compared to other machine learning prediction tasks, like image recognition. Stock market prediction is still

one of the challenging problems. Further, in the case of a financial application, the proposed models do not consider the tax and other trading fees associated with the Constant-Weighting Asset Allocation. In reality, continuously rebalancing a portfolio could lead to non-negligible tax and transaction fees, which may cause unexpected losses. Other research efforts have investigated these topics, including Reinforcement Learning, Bilevel Optimization, and other financial asset allocation strategies. Finally, as we used the validation accuracy to weight the objective function, we don't have scientific evidence to support this implementation. However, trustworthiness AI is attractive nowadays. We can extend the work to build a reliable prediction-then-optimization decision making framework.

Chapter 4

Prescriptive Bi-level Approach for Portfolio Risk-return Management

4.1 Introduction

Artificial Intelligence has revolutionized the field of finance by enabling businesses to make more informed and accurate decisions. One way AI is used in financial decision-making is through predictive analytics, which allows financial institutions to forecast future market trends more accurately and seek more returns. The previous two chapters of this dissertation presented approaches to incorporate predictive machine learning modeling into classic portfolio optimization problems. Next, in this chapter, we will consider a bi-level variation on portfolio optimization and how it can interact with machine learning-based forecasts.

Consider Hedge Fund Management (Fung et al., 2008), a type of business known for using performance-based fees. Typically, a hedge fund investment professional charges a management fee, which is a percentage of assets under management, as well as a performance fee, which is a percentage of the investment returns that exceed a certain benchmark. It is important to note that performance-based fees can create an alignment of interests between the investment professional (broker) and the client (investor) because the broker earns more when the investor's portfolio performs well. However, the business can also introduce conflicts of interest, as the broker may be motivated to take on more risk to boost returns and, consequently, their commission. Obviously, the financial broker makes decisions based on his interests in investing in a wide range of securities, maximizing its benefits. However, with limited market information, the investor has limited choices of securities to minimize risk and ensure a given expected return. Hence, the market information of securities becomes a critical business factor when a broker provides financial services to clients. Since two levels of decision-makers are involved in this problem – the broker and the investor – who have

imperfectly aligned incentives, it may be important to explicitly consider the two levels of decision-making. Such hierarchical decision-making is usually modeled by multi-level (in our case, bi-level) optimization, which is the topic of this chapter.

Bi-level optimization (Sinha et al., 2017), a class of mathematical programming, follows a two-level hierarchical architecture, where one optimization problem is nested within the other. It has gained significant attention due to its applications in various fields, including finance. The upper (leader) optimization problem is commonly constrained by the lower (follower) problem. Correspondingly, both upper and lower problems are optimized over two classes of decisions. The lower-level problem is a parametric optimization that is solved with respect to its feasible spaces. The lower-level problem serves as a constraint to the upper-level optimization problem. Consequently, the overall optimizations are considered in the feasible spaces that are lower-level optimal and also satisfy the upper-level constraints.

We consider the following version of the problem. An investor is interested in contracting with a broker, who will, in turn, manage a financial portfolio on behalf of the investor. We will assume that the broker acts according to modern portfolio optimization, for example, following the model considered in Chapter 3. Naturally, the broker can still select model hyper-parameters, most importantly selecting the securities and CVaR tail parameter (α), to adjust to their own risk preferences and incentive structure. The investor may have a limited view of these choices. On the other hand, due to the incentive structure, the investor can be reasonably assured that the broker will maximize the portfolio return. What may be missing is a more concerted effort to minimize risk, due to the mismatch in risk-aversion between the parties.

In the absence of predictive models, this mismatch can, in principle, be overcome by enforcing the same risk measure. The presence of the forecasts, and particularly if different forecasting models are available to different parties, makes modeling the decision-making even more challenging. Therefore, in this chapter, we consider this challenge and provide some modeling and computational insights.

The main contributions of this work are as follows: 1) we incorporate a hierarchical bi-level approach with two-level decision-making on portfolio optimization problems (the broker maximizes the returns, whereas the investor minimizes risk while ensuring a given expected return; 2) the bi-level optimization model considers modern ML technology in making decisions while performing risk hedging and seeking returns; 3) we develop a new bi-level optimization formulation to obtain optimal solutions for the considered problems.

4.2 Review of Relevant Literature

Bi-level optimization has applications in various fields, including economics (e.g., Stackelberg games), transportation (e.g., traffic network design and pricing), engineering (e.g., supply chain optimization), and machine learning (e.g., hyperparameter tuning) (e.g. Yue and You, 2017; Franceschi et al., 2018). Only a limited number of existing works considered this framework for the risk-averse stochastic optimization setting. We next review two of the most relevant papers.

In traditional portfolio optimization research (Li et al., 2019), there is only one variable vector involved in the decisions. Recent research endeavors (Leal et al., 2020) have considered innovative bi-level leader-follower portfolio selection problems aimed at optimizing the broker's (leader's) benefits while taking into account the selection of unit transaction costs and concurrently incorporating the definition of portfolio risk within the follower's framework. Hence, transaction costs become decision variables in the leader problem, and the follower problem optimizes the risk over the portfolio allocation. Then, a general MILP can be adapted to solve the problem with this new hierarchical structure of objectives.

With more twofold portfolio problems being studied, the broker-investor problems are further explored in (González-Díaz et al., 2021). In Gonz's work, the dealer exercises authority over the determination of transaction costs associated with diverse securities, pursuing the objective of profit maximization, while the investor confronts the conventional portfolio risk minimization problem concerning the allocation of capital across a spectrum of financial

instruments. The distinguishing feature of Gonz's work relies on the study of multiple follower cases as the problem complexity can be exponentially increased.

Both mentioned works consider Conditional-Value-at-Risk to measure and quantify the risk in the investor's problem. However, in the financial industry, brokers usually have the authority of their clients to decide the allocation. In addition, the non-alignment of interests can also weaken the relationship between the two parties from persisting in the long run. In this work, we study bi-level optimization modeling and machine learning techniques to address the hierarchical broker and investor problem we previously discussed. We ensure that the interests of both parties are aligned by the returns, given that the broker receives only the performance fee. The conflict of interests between the two parties exists in their utility theory towards the risk.

4.3 Methodology

Let us consider an investment over $N = \{1, 2, 3, ..., n\}$ securities. There exists a subset of $B \subseteq N$ securities that are predicted by some reliable (but not perfectly reliable) AI technology to be bullish during the next trading cycle. In most cases, $B \neq \emptyset$ and $B \subsetneqq N$. First, we assume that the broker has the full authority of his client in allocating all the capital to any securities in N, and the broker's goal is to maximize the portfolio returns with respect to a certain level of risk constraint, so that their commission, denoted by R^l, is maximized. We will denote the broker's decision variables as x_i^l, where $i = 1, 2, ..., N$. In other words, the broker does not explicitly restrict the portfolio to bullish assets. On the other hand, the market prediction results performed by the AI are shared with the investor. We assume that the investor is more risk-averse. Denote the investor's risk loss funciton $R^u = F^u(x^l, x^u)$. To balance the incentives, suppose that the investor authorizes the broker to proceed with an investment under the agreement that the risk of the broker's portfolio should not exceed the portfolio that consists only of the bullish securities, that was separately optimized by the investor according to their preferences. Hence, the investor's decision variables, denoted by

x_i^u, are a subset of securities, where $i \in B$. In general, this optimization problem can be formulated as follows,

$$\text{Min} \quad R^u = F(x^u, x^l) \tag{4.1}$$

$$\text{s.t.} \quad x^l \in \arg\max\{f(x^l) \,|\, g(x^l) \geq 0\} \tag{4.2}$$

$$G(x_u, x_l) \geq 0, \tag{4.3}$$

where $G(x_u, x_l)$ and $g(x_l)$ denotes the upper- and lower-level constraints, respectively.

In the formulation, we intend to divide the decision space using prediction to introduce a novel view of risk. Specifically, we use machine learning to predict market movement. The prediction results can split the decision space apart into two sub-spaces, bullish and bearish. We assume that the investor will focus only on the bullish set of the securities, which formulates an upper-level risk minimization problem. The broker at the lower level is guided by the investor in seeking return, with the constraint of risk ensuring that the lower level risk remains within acceptable bounds. Most importantly, the upper level will have a different decision space from the lower level because the upper level uses the subset of the entire securities set while the broker makes decisions on the entire security set. Therefore, we propose a prediction-based bi-level optimization model below,

Upper Level:

$$g = \min \eta + \frac{1}{m(1-\beta)} \sum_{j=1}^{m} \Big[- \sum_{i \in B} x_i^u r_{ij} - \eta \Big]^+ \tag{4.4}$$

$$\text{s.t.} \quad \sum_{i \in B} x_i^u = 1 \tag{4.5}$$

$$x_i^u \geq 0, \quad i \in B \tag{4.6}$$

Lower Level: (4.7)

$$\max \sum_{i=1}^{n} \left(p_i^{l+} \sum_{j=1}^{m} x_{ij}^{l} r_{ij}^{l+} - p_i^{l-} \sum_{j=1}^{m} x_{ij}^{l} r_{ij}^{l-} \right) \tag{4.8}$$

$$\text{s.t. } CVaR(x^l) <= g \tag{4.9}$$

$$\sum_{i=1}^{n} x_i^l = 1 \tag{4.10}$$

$$x_i^l \geq 0, \quad i \in N. \tag{4.11}$$

Here $(p_i^{l+}$ and $(p_i^{l-}$ are the probabilities predicted and calibrated by a machine learning model (for example, the LSTM+TemperatureScaling Neural Network) analogously with Chapter 3.

Both the broker and the investor measure the risk using CVaR to avoid extreme losses. At the lower level, the predicted expected return is maximized, which is consistent with the broker's interests, i.e., the performance-based fees. The broker is motivated to be risk-seeking and maximize their interests. Note that the broker's objective is evaluated according to the advanced probabilistic ML model since it is the most likely to generate maximum rewards. However, the investor at the upper level is ensuring the portfolio to be more conservative, as it minimizes the extreme loss and constrains the risk in the lower level. In addition, the CVaR risk objective at the upper level is aimed at enhancing diversification. Otherwise, the model could produce some results that all capital is allocated to one security.

The proposed bi-level problem relies on the formulations from both Chapters 2 and 3. Specifically, the upper-level problem is the model presented in Chapter 2. The lower-level problem adopts the return objective, identified as the second objective in the bi-objective models discussed in Chapter 3, along with an additional constraint linking the two levels.

4.3.1 Feasibility and Solution Technique

Feasibility is a fundamental consideration in optimization and is crucial for ensuring that the problem can be solved to achieve meaningful results. A feasible solution to a bilevel optimization problem must satisfy all the constraints of both the upper level and lower level problems. The feasibility of a bilevel optimization problem can be assessed in three ways: upper level feasibility, lower level feasibility, and overall feasibility. The upper and lower level feasibility refers to a set of decision variables that satisfy its own level without considering the other level. The overall feasibility means that there is a combination of decision variables for both levels' problems that jointly satisfy all constraints and objectives. This is a more stringent condition than just having each level individually feasible. Obviously, we have shown the feasibility of the upper level problem in Chapter 2. The feasibility of the proposed model mainly relies on constraint 4.9. Since the CVaR risk measure is sub-additive and convex, the inequality in 4.12 holds,

$$\min CVaR(x^l) <= \min CVaR(x^u), \tag{4.12}$$

where $CVaR(x^u)$ and $CVaR(x^l)$ represent the upper and lower level risk quantified by CVaR, and g is the lower level objective value. Thus, the proposed model is guaranteed to be feasible.

Solving bilevel optimization problems can be challenging due to their hierarchical structure, where an upper-level problem depends on the solution of a lower-level problem. Various solution techniques and approaches are employed to address bilevel optimization problems, including Penalty method (White and Anandalingam, 1993), Heuristic method (Angelo and Barbosa, 2015), Transformation using Karush-Kuhn-Tucker(KKT) condition (e.g. Sinha and Sinha, 2002; Allende and Still, 2013), etc.

In this work, we consider a reformulation using KKT condition for our bilevel optimization problem. The KKT transformation unifies the upper-level and lower-level problems into a single-level problem while considering the interactions between the two levels through

Lagrange multipliers and dual variables. This transformation simplifies the problem and enables the application of standard optimization solvers and algorithms to find optimal solutions.

4.4 Trading Simulations: Case Study for The US Stocks

This section is dedicated to presenting the numerical results associated with a case study based on real data extracted from the US stock market, with the aim of suggesting potential economic insights that verify the performance of the proposed model. Although further investigation into advanced data sources and models is valuable for enhancing the reliability of prediction results and improving the final portfolio's performance, it is essential to know that enhancing prediction performance would entail separate research work.

For this case study, we use the same simulation setup as given in Chapter 3, including selection of assets, time period, predictive models, etc. Trading settings given in Table 3.6 are used. In the meantime, we consider two benchmarks to compare the performance with the proposed model: traditional CVaR and CVaR+BinaryPrediction (model presented in Chapter 2).

Models	**CVaR**	**Average Return**	**Return Std**	**Final Value**
CVaR	1.645	0.05154	0.7808	\$9,608.44
CVaR+LSTM	2.017	0.2608	0.762	\$12,858.22
LSTM+Bilevel	3.064	0.1371	1.40	\$11,516.77
Note: CVaR, Average Return, and Return Std are average values in percentage.				

Table 4.1: Bilevel Trading Simulation with Benchmarks Comparison

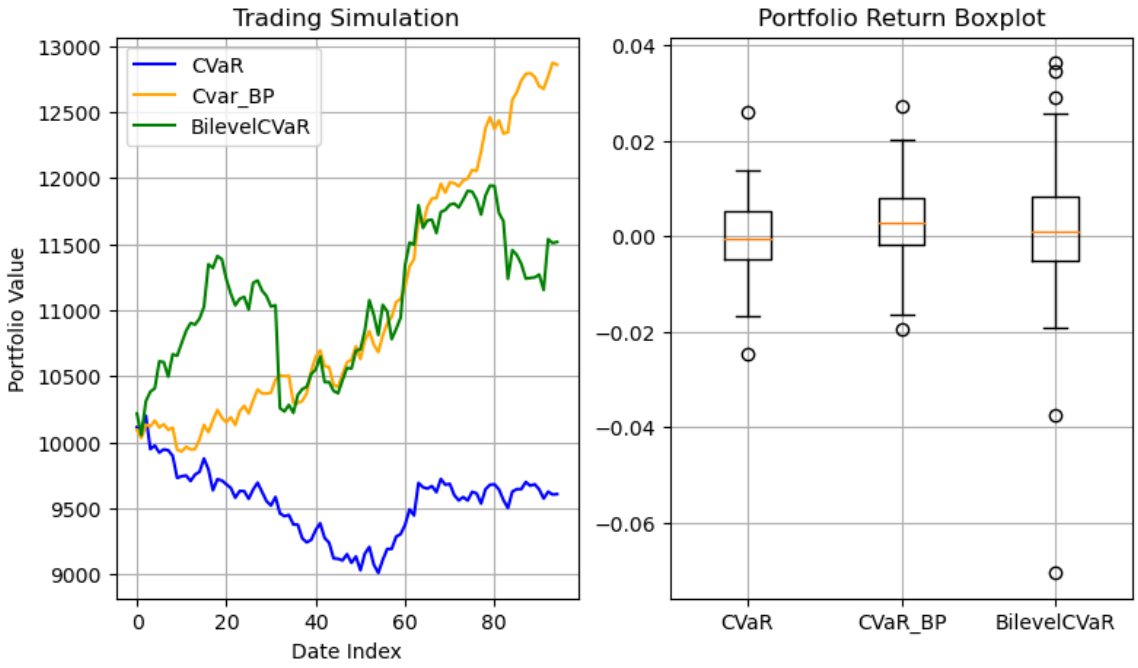


Figure 4.1: Bilevel Model Trading Simulation with Benchmarks

The results are summarized in Table 4.1 and Figure 4.1. There are two major observations we made through the trading simulation. The bilevel programming framework failed to produce attractive returns as the bi-objective and CVaR+Binary Prediction formulation. Certainly, the broker's problem is well constrained by the investor's risk averse utility reflected in the lower level objective. It further limited the broker's motivation to seek risk. From the upper level perspective, the portfolio generated acceptable profit, more than the standard CVaR model, and the risk is hedged over a hierarchical setting. The model also encounters a risk increase, compared to CVaR+Binary Prediction formulation, since return seeking at the lower level is involved in the decision-making process. However, the bilevel programming framework takes advantage of balancing the risk and return from the hierarchical settings and plunder returns by having machine learning prediction involved in the decision making process. The bilevel formulation simulates a real competing game between broker and investor, and the solution can represent optimal decisions of both competing parties. In addition,

the effective utilization of machine learning and bilevel programming in decision-making constitutes a significant contribution within the context of this research.

4.5 Conclusion

This chapter proposed a single-period portfolio selection problem with two levels of decision-making on risk and return. On the one hand, there is a broker who controls the portfolio allocation to the different securities in order to maximize his benefit, and there is an investor who chooses risk preference while ensuring a certain expected return. This framework extends this topic to an implicit competition in order to boost the decision of both parties. To verify the model performance and get close to economic insights that can be achieved by the model, we have developed a case study based on real data from the US stock market with a constant allocation trading simulation during 95 trading days. These results report a comparison between the models and the benchmarks. One of the most interesting findings of our analysis is that there seems to be some form of advantage of the Bilevel model over the traditional CVaR model in balancing the risk and return obtained by the two decision makers. Bilevel programming exhibits favorable characteristics in the modeling and resolution of the competing problem. Furthermore, this study lays the groundwork for potential future research paths, including extensions to multiple brokers and investors models, the inclusion of discrete trading cycle scenarios, and the investigation of economic insights by using Bilevel programming for risk-return management. It is not only the problem addressed in this paper but also for other potential extensions. We provide our code to allow researchers to replicate and extend our work at `https://github.com/jzy0040/Dissertation`.

Chapter 5

Conclusions, Limitations and Future Work

In this dissertation, we have studied a series of prediction-then-optimization frameworks for portfolio decision making problems. In Chapter 2, we propose a hybrid machine learning-informed CVaR risk optimization model to hedge against risk, while simultaneously taking advantage for binary stock price classification in order to improve return-seeking. We first focus on building the stock movement classification model with market data and predicting the bullish and bearish stocks. The binary classification results are introduced to the CVaR risk optimization model and force the model to avoid the bearish predicted stocks, as zero weight is allocated to those stocks. The primary focus of this study is not attempting to improve the accuracy of the stock movement prediction. Instead, we are interested in leveraging the performance of decision making models by combining the two disciplines. The results provide evidence that the proposed model can benefit from the prediction in seeking returns and outperforms the traditional CVaR risk model. This groundwork leads the future research path in this dissertation, proving that the prediction-then-optimization framework can produce attractive and meaningful results in risk hedging and return seeking. However, the model causes issues due to its limitations. The major limitation we found is the binary classification prediction setting. Considering the prediction accuracy, our results further proved that the task of stock market prediction is still challenging. The average prediction accuracy is significantly lower than the other machine learning applications, like image and speech recolonization, E-commerce Product Recommendations, etc. Intuitively, wrong prediction results could mislead the optimization model and cause it to make bad decisions. In an extreme situation, if all the bullish predicted results are incorrect, the binary classification results will force the model to make a decision among all bearish assets. By

and large, we consider the limitation was caused by the deterministic nature of the binary classification results, which ignore the uncertainty in prediction results.

To address this issue, we proposed a bi-objective probabilistic portfolio decision making model in Chapter 3. In this probabilistic model, instead of binary classification on market movement, we estimate the probability of asset movement. The proposed probabilistic portfolio models optimize the risk and return objectives simultaneously. The risk and return are defined and calculated using the predicted probability. Thus, the prediction uncertainty can be signified using predicted probability. In this work, a primary challenge involves the estimation of the true probability of stock movement based on market data. In recent machine learning studies, probability calibration was further developed for modern neural network classification models. In machine learning, probability calibration is a data-driven technique that involves adjusting the output probabilities of a classification model to match the true likelihood of the predicted outcomes. Calibrating the output probabilities of a model is critical when deploying machine learning models in scenarios where well-calibrated probabilities are critical for decision-making. Specifically, we apply the Temperature Scaling probability calibration technique to calibrate the output of our stock movement classification models. We use two metrics to measure the calibration performance, the Cross-Entropy Loss and Expected Calibration Error. Based on our results, both metrics decrease after calibration for NN and LSTM across 36 US stocks. Consequently, we consider our probability calibration to be successful. In optimization modeling, we first consider maximizing the expected future return, which was defined using predicted probability. Besides, the model can be varied by using predicted probability to define the risk. In that case, the risk measure, conditional Value at Risk, will be performed using positive and negative return scenarios corresponding to the predicted probabilities. According to the trading simulation, we found that the second formulation outperforms in seeking return. The first formulation also shows its advantage in balancing the risk and return. However, the calibration performance has significant variations across different assets. The temperature scaling techniques will calibrate the probability with

respect to the prediction accuracy in the case of binary classification, but the calibration cannot help with the accuracy. Obviously, the low prediction accuracy assets will surely have a poor calibration performance, which means the probability will be less reliable. Thus, a more accurate prediction model with a robust calibration technique will always be preferred. Besides, as we mentioned before, we have less evidence in choosing λ.

In Chapter 4, we introduce a realistic broker-investor competing problem. In this problem, the broker and investor are aligned by the portfolio returns. However, since the commission was based on portfolio performance, the broker will tend to take more risk in seeking returns. Thus, the investor can set a risk tolerance for the broker such that the broker's decision problem (lower level) follows the investor's risk optimization problem(upper level). The hierarchical nature of this problem must be reflected in the modeling, which indicates a bilevel modeling framework. According to our literature review, this is a novel attempt to use bilevel optimization to construct a portfolio where machine learning is considered in the decision making process. Our result shows the research potential on this topic and possible extension to multiple brokers and investor problems.

In this dissertation, we demonstrate the efficiency and effectiveness of the predict-then-optimization framework in portfolio decision making. We discussed multiple machine learning techniques, including classification algorithms and probability calibration techniques. Still, there are several directions in which we can extend our research. First, when we build the prediction model, the data quality and availability are constantly causing issues to keep the prediction results less reliable. We would investigate more data sources to build more informative and reliable data for market prediction, including Twitter, financial news, etc. In addition, the evolution of machine learning algorithms is dramatically affecting academia and industry. The Large Language Model, like ChatGPT 4.0, nowadays are widely recognized for both research and application potentials. New attempts can be applied to market prediction in financial applications. Meanwhile, with the development of AI trustworthiness, new techniques can be used to explain how much we should trust the AI prediction results. Thus, the study

will have a significant impact on decision making using the prediction-then-optimization framework. Moreover, the bilevel multi-broker and multi-investor problem is a novel research path, which may have modeling and computational research involved. Last but not least, the proposed model can also be applied to other applications wherever risk is considered, including Supply Chain, Power Management, Marketing, etc. We believe that future research on these topics can have a significant potential.

Bibliography

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information fusion*, 76:243–297, 2021.

Gemayqzel Bouza Allende and Georg Still. Solving bilevel programs with the kkt-approach. *Mathematical programming*, 138:309–332, 2013.

Mahantesh C Angadi and Amogh P Kulkarni. Time series data analysis for stock market prediction using data mining techniques with r. *International Journal of Advanced Research in Computer Science*, 6(6), 2015.

Jaqueline S Angelo and Helio JC Barbosa. A study on the use of heuristics to solve a bilevel programming problem. *International Transactions in Operational Research*, 22(5):861–882, 2015.

George S Atsalakis and Kimon P Valavanis. Surveying stock market forecasting techniques–part ii: Soft computing methods. *Expert systems with applications*, 36(3):5932–5941, 2009.

Depei Bao and Zehong Yang. Intelligent stock trading system by turning point confirming and probabilistic reasoning. *Expert Systems with Applications*, 34(1):620–627, 2008.

Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of computational science*, 2(1):1–8, 2011.

Luyang Chen, Markus Pelger, and Jason Zhu. Deep learning in asset pricing. *Management Science*, 2023.

Paul H.. Cootner. *The Random Character of Stock Market Prices. Paul H. Cootner.* MITP, 1964.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

RK Dase and DD Pawar. Application of artificial neural network for stock market predictions: A review of literature. *International Journal of Machine Intelligence*, 2(2):14–17, 2010.

D Duffie and J Pan. An overview of value-at-risk." journal of derivatives. *Journal of Derivatives 4*, 1997.

Eugene F. Fama. The behavior of stock-market prices. *The Journal of Business*, 38:34, 1965.

Eugene F Fama. Efficient capital markets: Ii. *The journal of finance*, 46(5):1575–1617, 1991.

Eugene F Fama. Random walks in stock market prices. *Financial analysts journal*, 51(1): 75–80, 1995.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International conference on machine learning*, pages 1568–1577. PMLR, 2018.

Fabio D Freitas, Alberto F De Souza, and Ailson R De Almeida. Prediction-based portfolio optimization model using neural networks. *Neurocomputing*, 72(10-12):2155–2170, 2009.

William Fung, David A Hsieh, Narayan Y Naik, and Tarun Ramadorai. Hedge funds: Performance, risk, and capital formation. *The Journal of Finance*, 63(4):1777–1803, 2008.

Patrick Gabrielsson and Ulf Johansson. High-frequency equity index futures trading using recurrent reinforcement learning with candlesticks. In *2015 IEEE Symposium Series on Computational Intelligence*, pages 734–741. IEEE, 2015.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

Julio González-Díaz, Brais González-Rodríguez, Marina Leal, and Justo Puerto. Global optimization for bilevel portfolio design: Economic insights from the dow jones index. *Omega*, 102:102353, 2021.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

Michael C. Jensen and William H. Meckling. Theory of the firm: Managerial behavior, agency costs and ownership structure. *Journal of Financial Economics*, 3(4):305–360, 1976. ISSN 0304-405X. doi: https://doi.org/10.1016/0304-405X(76)90026-X. URL `https://www.sciencedirect.com/science/article/pii/0304405X7690026X`.

Peiran Jiao, Andre Veiga, and Ansgar Walther. Social media, news media and the stock market. *Journal of Economic Behavior & Organization*, 176:63–90, 2020.

Philippe Jorion. *Value at risk: the new benchmark for controlling market risk.* Irwin Professional Pub., 1997.

Kyoung-jae Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

Kyoung-jae Kim and Ingoo Han. Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert systems with Applications*, 19(2):125–132, 2000.

Pavlo Krokhmal, Michael Zabarankin, and Stan Uryasev. Modeling and optimization of risk. *Handbook of the fundamentals of financial decision making: Part II*, pages 555–600, 2013.

Marina Leal, Diego Ponce, and Justo Puerto. Portfolio problems with two levels decision-makers: Optimal portfolio selection with pricing decisions on transaction costs. *European Journal of Operational Research*, 284(2):712–727, 2020.

Xiaodong Li, Haoran Xie, Li Chen, Jianping Wang, and Xiaotie Deng. News impact on stock price return via sentiment analysis. *Knowledge-Based Systems*, 69:14–23, 2014.

Zongxin Li, Zhiping Chen, and Yongchang Hui. Portfolio selection through maslow's need hierarchy theory. *Applied Economics*, 51(4):364–372, 2019.

Sheng Liu, Aakash Kaku, Weicheng Zhu, Matan Leibovich, Sreyas Mohan, Boyang Yu, Haoxiang Huang, Laure Zanna, Narges Razavian, Jonathan Niles-Weed, et al. Deep probability estimation. *arXiv preprint arXiv:2111.10734*, 2021.

Yilin Ma, Ruizhu Han, and Weizhong Wang. Prediction-based portfolio optimization models using deep neural networks. *IEEE Access*, 8:115393–115405, 2020.

Yilin Ma, Ruizhu Han, and Weizhong Wang. Portfolio optimization with return prediction using deep learning and machine learning. *Expert Systems with Applications*, 165:113973, 2021.

Burton G Malkiel. The efficient market hypothesis and its critics. *Journal of economic perspectives*, 17(1):59–82, 2003.

HM Markowitz. Portfolio selection. *Journal of Finance, March*, 1952.

Anshul Mittal and Arpit Goel. Stock prediction using twitter sentiment analysis. *Standford University, CS229 (2011 http://cs229. stanford. edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis. pdf)*, 15:2352, 2012.

Helen Susannah Moat, Chester Curme, Adam Avakian, Dror Y Kenett, H Eugene Stanley, and Tobias Preis. Quantifying wikipedia usage patterns before stock market moves. *Scientific reports*, 3(1):1–5, 2013.

Oskar Morgenstern and John Von Neumann. *Theory of games and economic behavior*. Princeton university press, 1953.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

David MQ Nelson, Adriano CM Pereira, and Renato A De Oliveira. Stock market's price movement prediction with lstm neural networks. In *2017 International joint conference on neural networks (IJCNN)*, pages 1419–1426. IEEE, 2017.

Thien Hai Nguyen, Kiyoaki Shirai, and Julien Velcin. Sentiment analysis on social media for stock movement prediction. *Expert Systems with Applications*, 42(24):9603–9611, 2015.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 625–632, 2005.

John R Nofsinger. Social mood and financial economics. *The Journal of Behavioral Finance*, 6(3):144–160, 2005.

Phichhang Ou and Hengshan Wang. Prediction of stock market index movement by ten data mining techniques. *Modern Applied Science*, 3(12):28–42, 2009.

John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

Robert R Prechter Jr and Wayne D Parker. The financial/economic dichotomy in social behavioral dynamics: the socionomic perspective. *The Journal of Behavioral Finance*, 8 (2):84–108, 2007.

Tobias Preis, Helen Susannah Moat, and H Eugene Stanley. Quantifying trading behavior in financial markets using google trends. *Scientific reports*, 3(1):1–6, 2013.

R Tyrrell Rockafellar, Stanislav Uryasev, et al. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

Utsav Sadana, Abhilash Chenreddy, Erick Delage, Alexandre Forel, Emma Frejinger, and Thibaut Vidal. A survey of contextual optimization methods for decision making under uncertainty. *arXiv preprint arXiv:2306.10374*, 2023.

Sreelekshmy Selvin, R Vinayakumar, EA Gopalakrishnan, Vijay Krishna Menon, and KP Soman. Stock price prediction using lstm, rnn and cnn-sliding window model. In *2017 international conference on advances in computing, communications and informatics (icacci)*, pages 1643–1647. IEEE, 2017.

Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. A review on bilevel optimization: from classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2017.

Surabhi Sinha and SB Sinha. Kkt transformation approach for multi-objective multi-level linear programming problems. *European Journal of Operational Research*, 143(1):19–31, 2002.

Vernon L Smith. Constructivist and ecological rationality in economics. *American economic review*, 93(3):465–508, 2003.

Van-Dai Ta, CHUAN-MING Liu, and Direselign Addis Tadesse. Portfolio optimization-based stock prediction using long-short term memory network in quantitative trading. *Applied Sciences*, 10(2):437, 2020.

Bin Weng, Mohamed A Ahmed, and Fadel M Megahed. Stock market one-day ahead movement prediction using disparate data sources. *Expert Systems with Applications*, 79: 153–163, 2017.

Bin Weng, Lin Lu, Xing Wang, Fadel M Megahed, and Waldyn Martinez. Predicting short-term stock prices using ensemble methods and online data sources. *Expert Systems with Applications*, 112:258–273, 2018.

Douglas J White and G Anandalingam. A penalty function approach for solving bi-level linear programs. *Journal of Global Optimization*, 3:397–419, 1993.

Frank Z Xing, Erik Cambria, and Roy E Welsch. Natural language based financial forecasting: a survey. *Artificial Intelligence Review*, 50(1):49–73, 2018.

Yumo Xu and Shay B Cohen. Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1970–1979, 2018.

Dajun Yue and Fengqi You. Stackelberg-game-based modeling and optimization for supply chain design and operations: A mixed integer bilevel programming framework. *Computers & Chemical Engineering*, 102:81–95, 2017.

Appendix A

Missing Data Summary

FinSentS missing values and infinite values are summarized in the table below.

Table A.1: FinSentS Missing Values

Stocks	**Total Missing**	**Missing Rate**	**Infinite Value**
AAPL	72	0.0358	False
ABC	1708	0.8493	False
AMZN	76	0.0378	False
ANTM	1389	0.6907	False
BAC	622	0.3093	False
BRK-A	not available	not available	False
CAH	1498	0.7449	False
CI	1198	0.5957	False
CMCSA	567	0.2819	False
COST	648	0.3222	False
CVS	846	0.4207	False
CVX	494	0.2456	False
F	233	0.1159	False
FB	83	0.0413	False
FDX	789	0.3923	False
FNMA	496	0.7403	True

Continued on next page

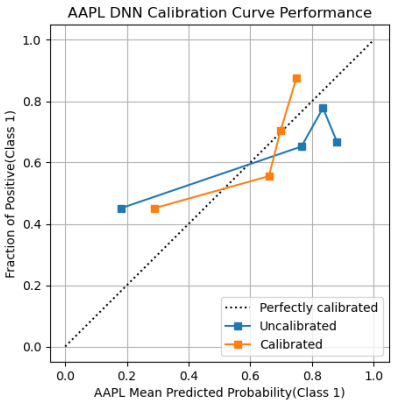
Table A.1 – continued from previous page

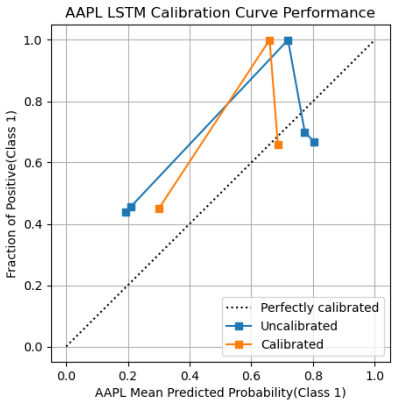
Stocks	Precision	Class 0 Precision	Class 1 Precision
GM	339	0.1685	True
GOOGL	196	0.0975	False
HD	721	0.3585	False
JNJ	594	0.2953	False
JPM	451	0.2243	False
KO	534	0.2655	False
KR	938	0.4664	False
MCK	1379	0.6857	False
MPC	1447	0.7195	False
MSFT	126	0.0627	True
PFE	555	0.2760	False
PSX	1639	0.8150	False
T	367	0.1825	False
TM	164	0.0816	False
TSLA	122	0.0607	False
UNH	1037	0.5157	False
VZ	515	0.2561	False
WBA	972	0.4833	False
WMT	171	0.0850	False
XOM	324	0.1611	True

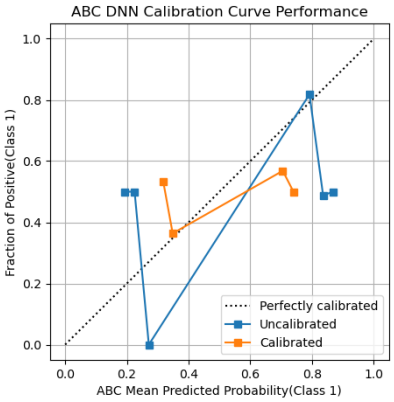
Note: missing rates are calculated based on all dates.

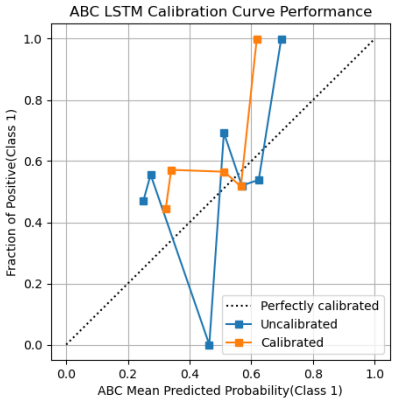
Appendix B

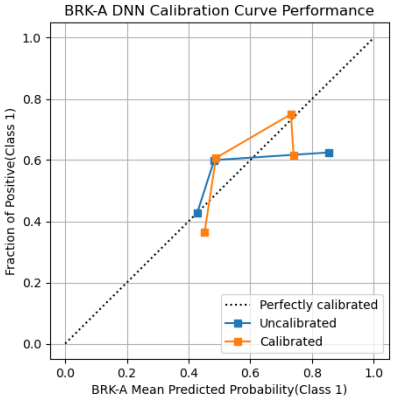
Reliability Diagram of Calibration Results

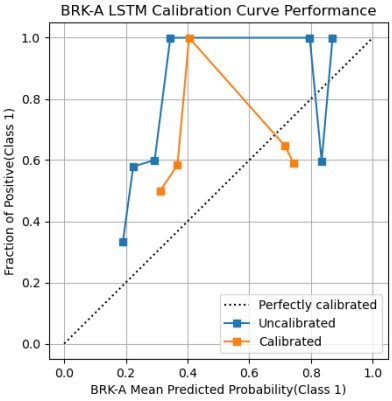


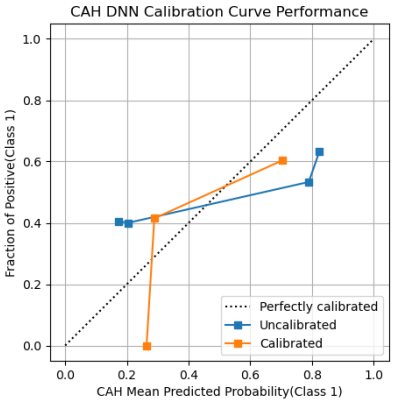


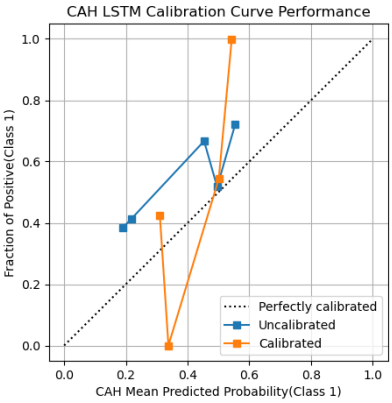


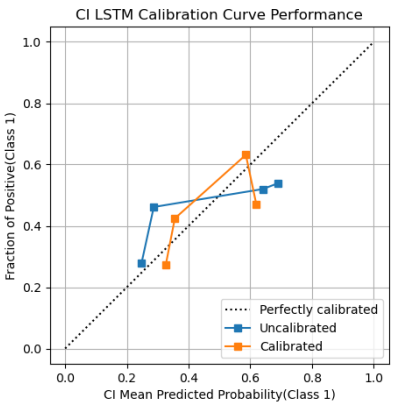


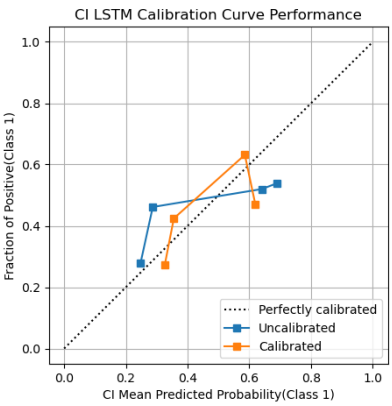

BRK-A DNN Calibration Curve Performance
Fraction of Positive(Class 1)
BRK-A Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

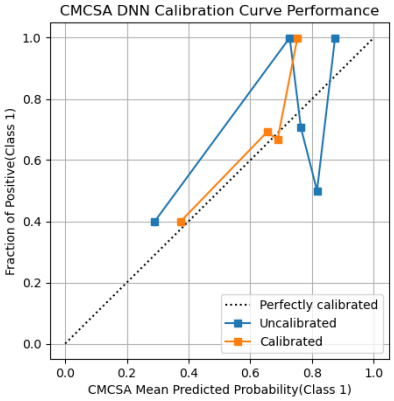

BRK-A LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
BRK-A Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

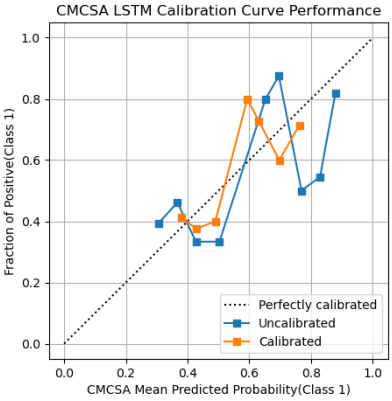

CAH DNN Calibration Curve Performance
Fraction of Positive(Class 1)
CAH Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

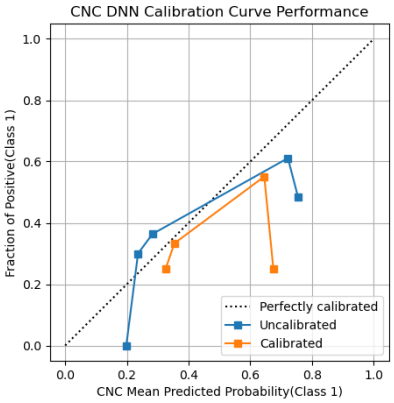

CAH LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
CAH Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

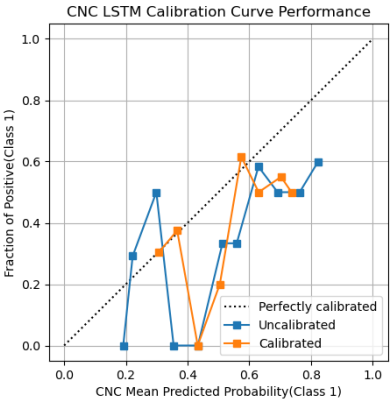

CI LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
CI Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

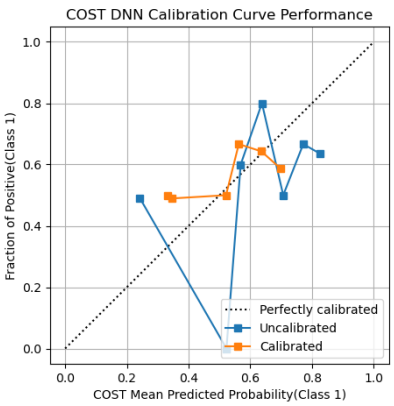

CI LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
CI Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

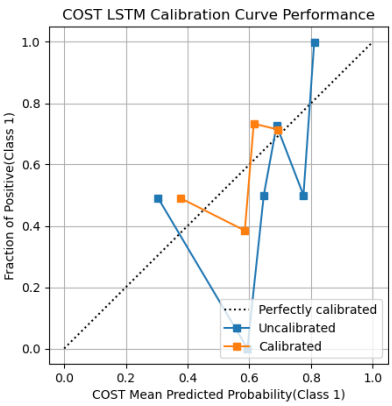

CMCSA DNN Calibration Curve Performance
Fraction of Positive(Class 1)
CMCSA Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

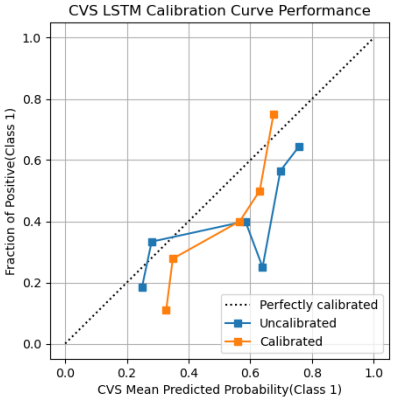

CMCSA LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
CMCSA Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

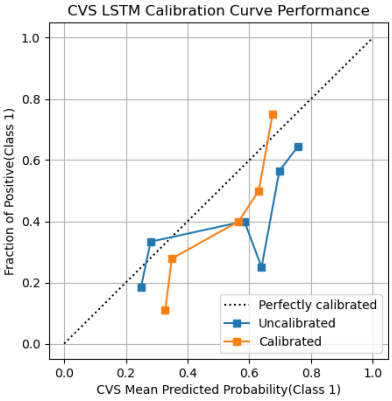

CNC DNN Calibration Curve Performance
Fraction of Positive(Class 1)
CNC Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

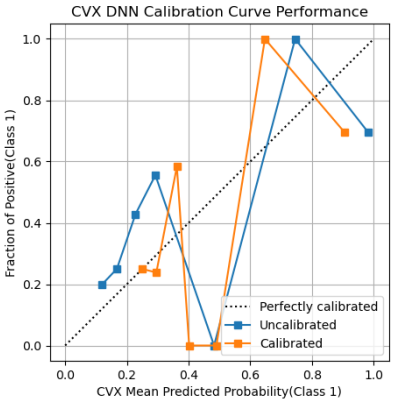

CNC LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
CNC Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

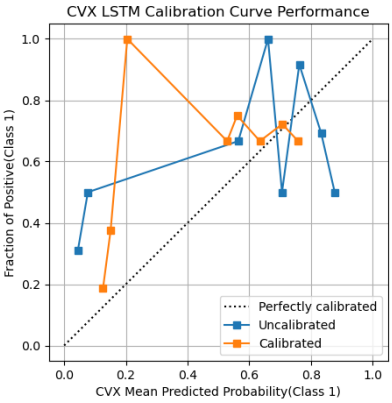

COST DNN Calibration Curve Performance
Fraction of Positive(Class 1)
COST Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

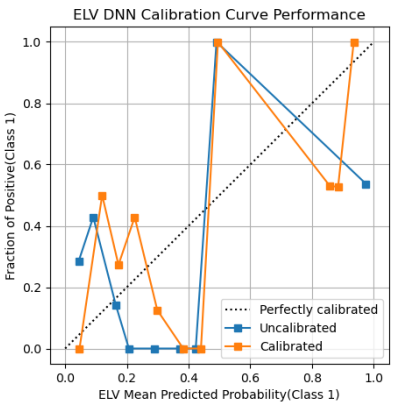

COST LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
COST Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

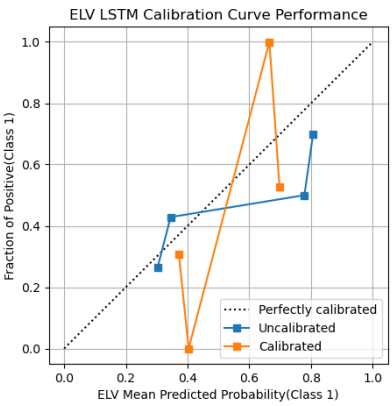

CVS LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
CVS Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

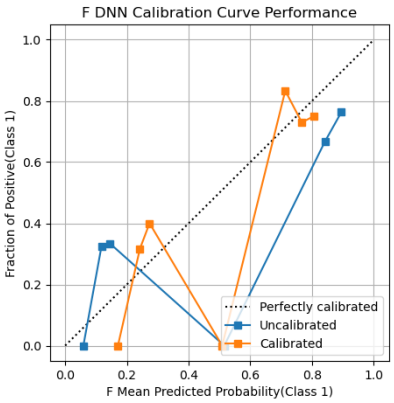

CVS LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
CVS Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

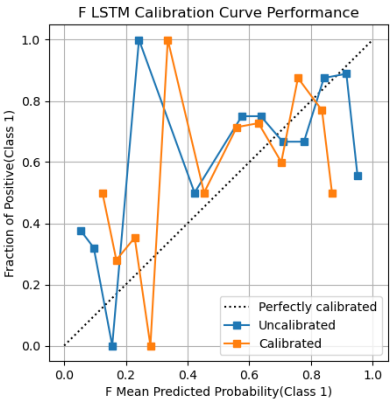

CVX DNN Calibration Curve Performance
Fraction of Positive(Class 1)
CVX Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

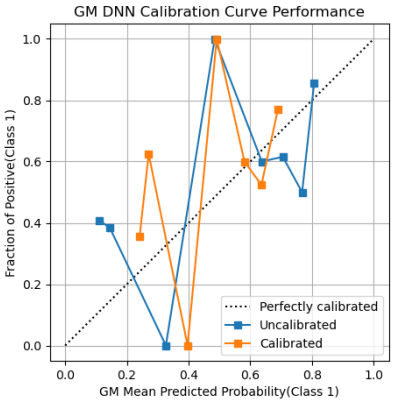

CVX LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
CVX Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

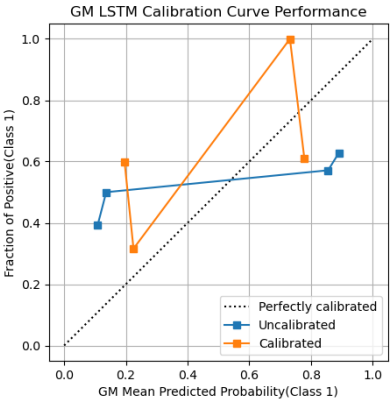

ELV DNN Calibration Curve Performance
Fraction of Positive(Class 1)
ELV Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

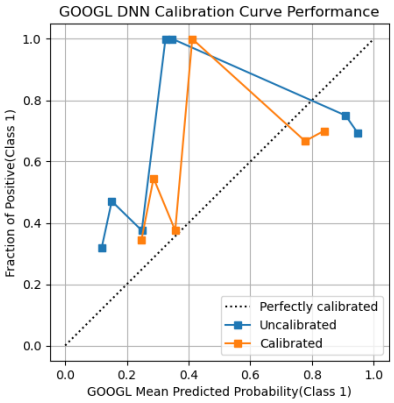

ELV LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
ELV Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

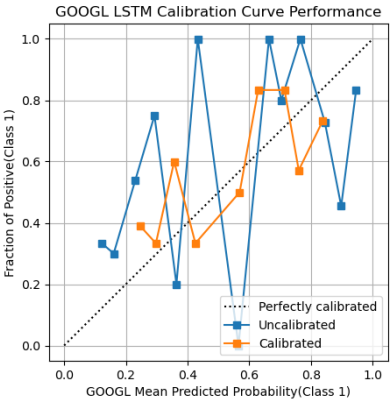

F DNN Calibration Curve Performance
Fraction of Positive(Class 1)
F Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

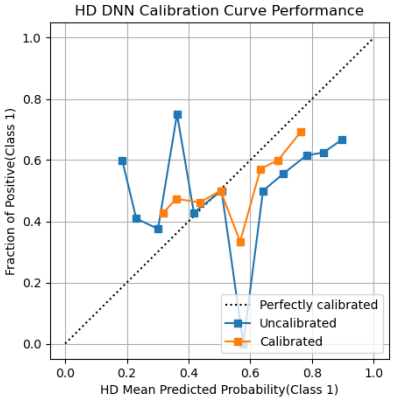

F LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
F Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

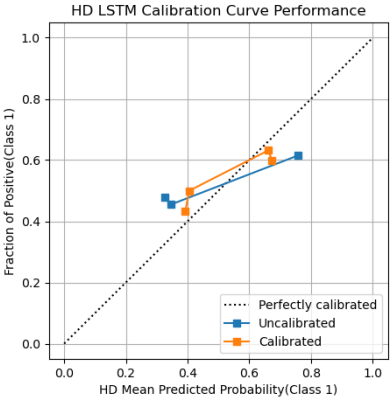

GM DNN Calibration Curve Performance
Fraction of Positive(Class 1)
GM Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

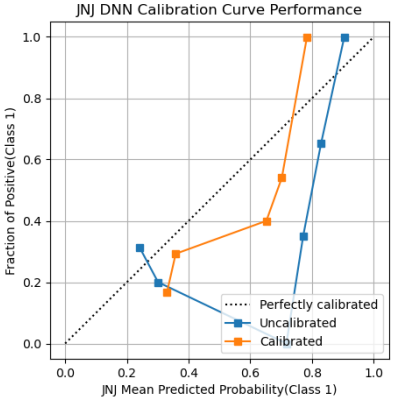

GM LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
GM Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

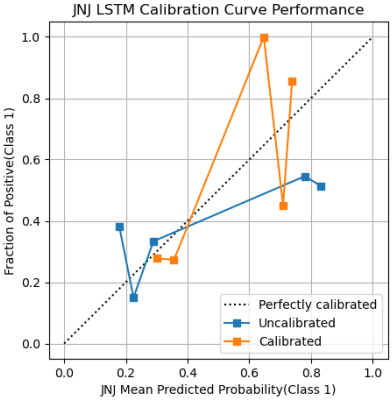

GOOGL DNN Calibration Curve Performance
Fraction of Positive(Class 1)
GOOGL Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

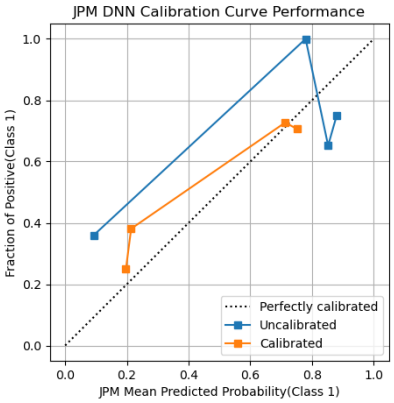

GOOGL LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
GOOGL Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

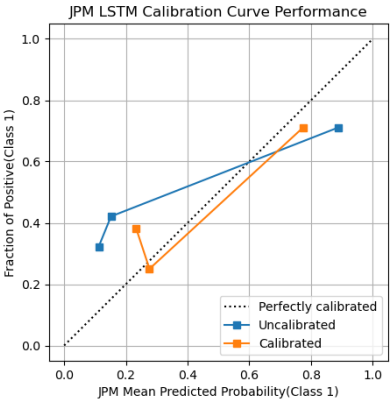

HD DNN Calibration Curve Performance
Fraction of Positive(Class 1)
HD Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

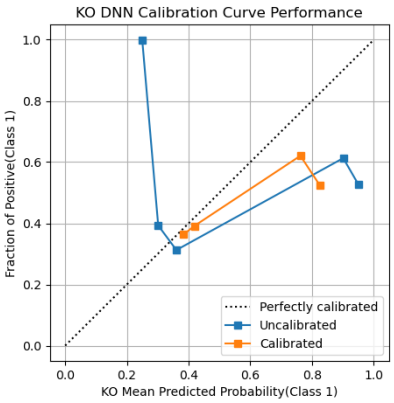

HD LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
HD Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

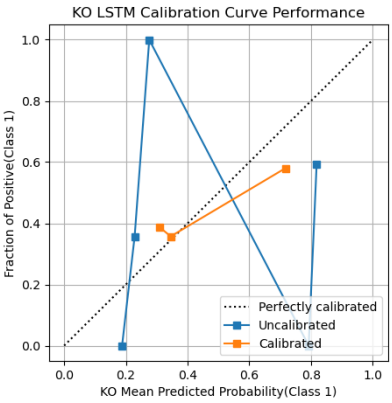

JNJ DNN Calibration Curve Performance
Fraction of Positive(Class 1)
JNJ Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

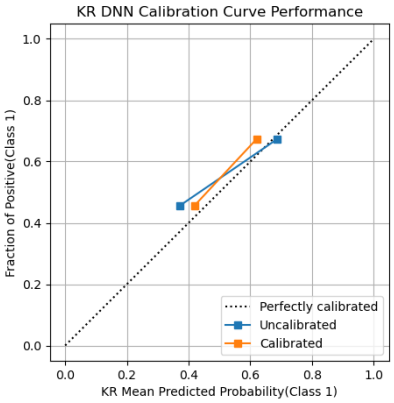

JNJ LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
JNJ Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

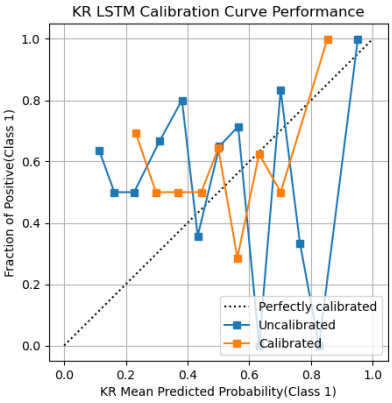

JPM DNN Calibration Curve Performance
Fraction of Positive(Class 1)
JPM Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

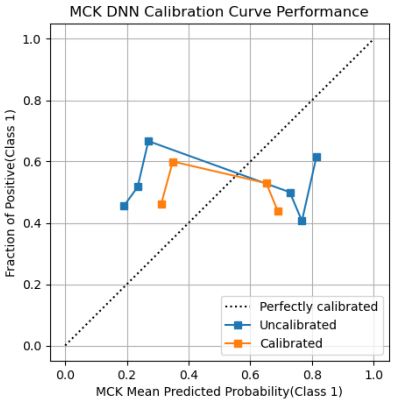

JPM LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
JPM Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

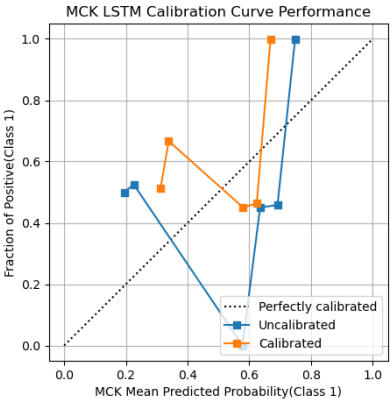

KO DNN Calibration Curve Performance
Fraction of Positive(Class 1)
KO Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

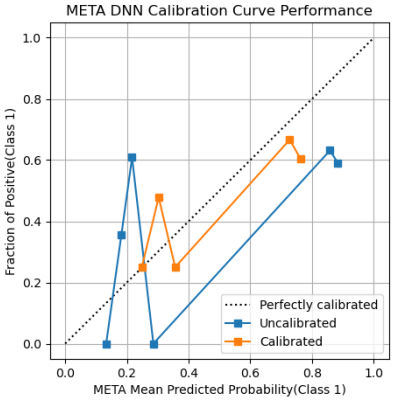

KO LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
KO Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

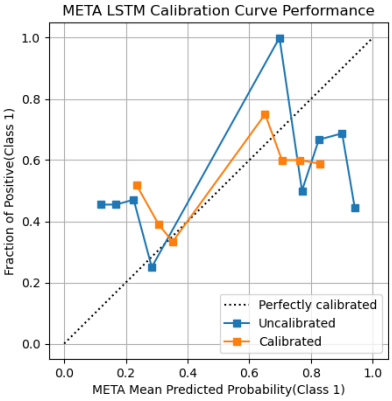

KR DNN Calibration Curve Performance
Fraction of Positive(Class 1)
KR Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

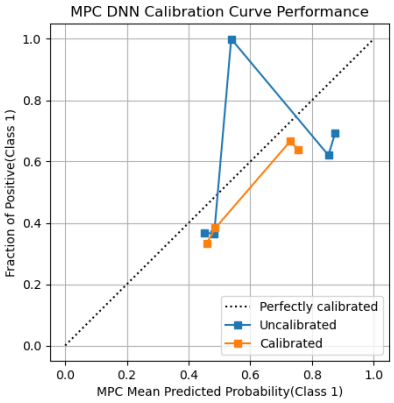

KR LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
KR Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

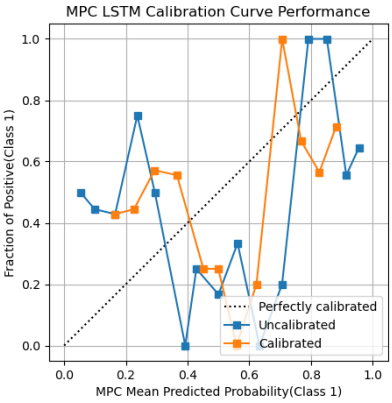

MCK DNN Calibration Curve Performance
Fraction of Positive(Class 1)
MCK Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

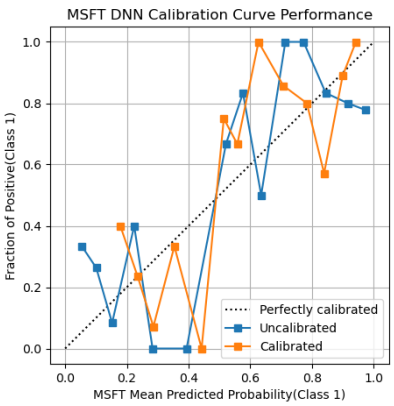

MCK LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
MCK Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

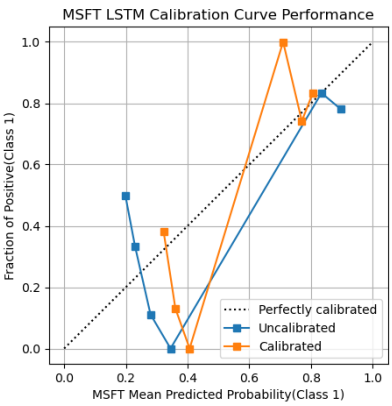

META DNN Calibration Curve Performance
Fraction of Positive(Class 1)
META Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

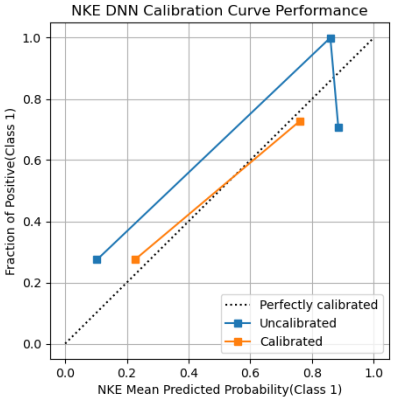

META LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
META Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

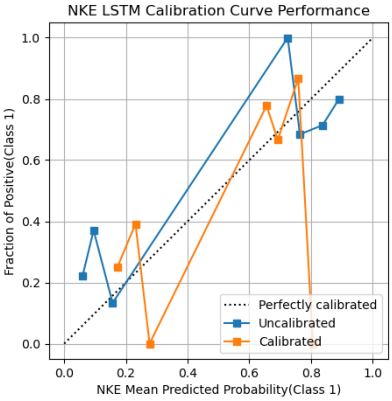

MPC DNN Calibration Curve Performance
Fraction of Positive(Class 1)
MPC Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

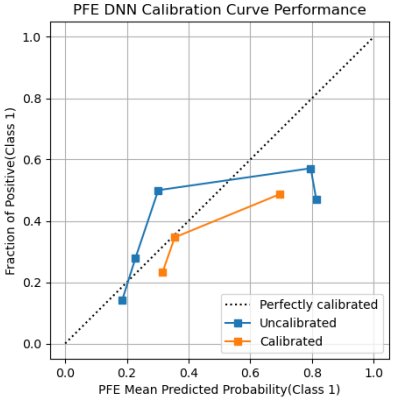

MPC LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
MPC Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

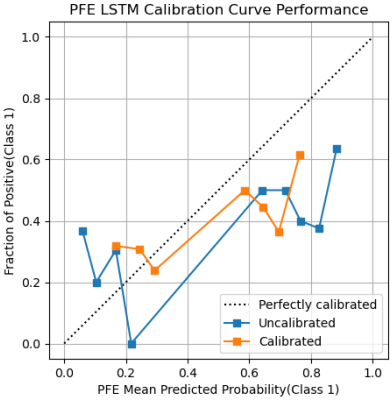

MSFT DNN Calibration Curve Performance
Fraction of Positive(Class 1)
MSFT Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

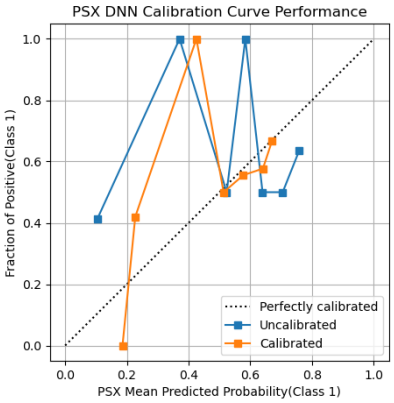

MSFT LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
MSFT Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

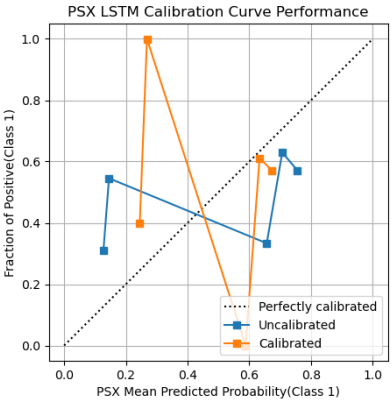

NKE DNN Calibration Curve Performance
Fraction of Positive(Class 1)
NKE Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

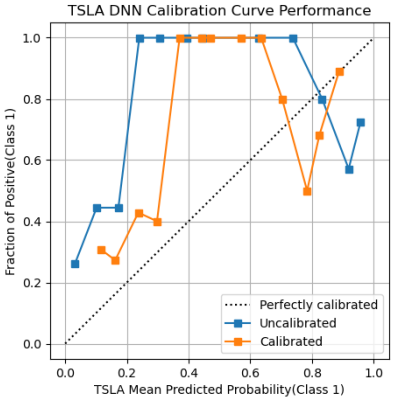

NKE LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
NKE Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

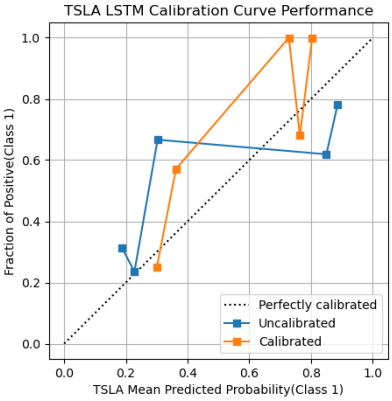

PFE DNN Calibration Curve Performance
Fraction of Positive(Class 1)
PFE Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

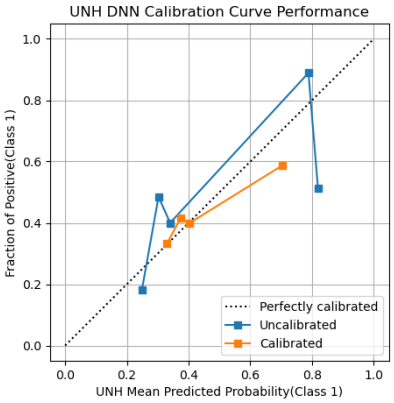

PFE LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
PFE Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

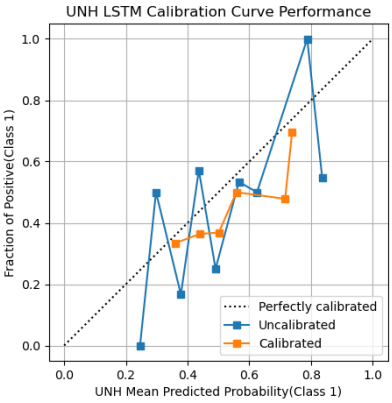

PSX DNN Calibration Curve Performance
Fraction of Positive(Class 1)
PSX Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

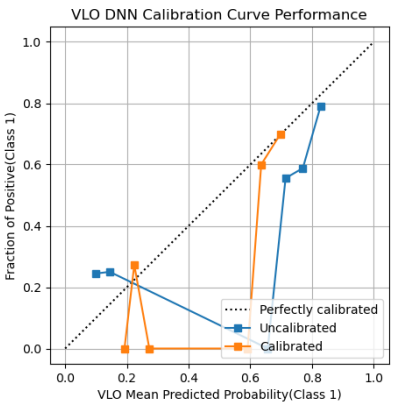

PSX LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
PSX Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

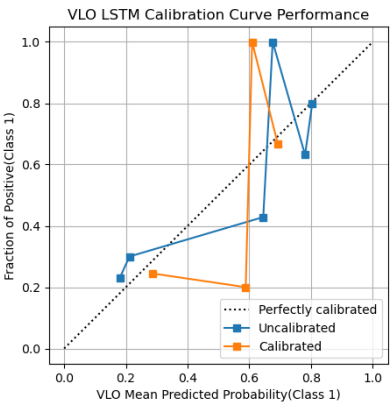

TSLA DNN Calibration Curve Performance
Fraction of Positive(Class 1)
TSLA Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

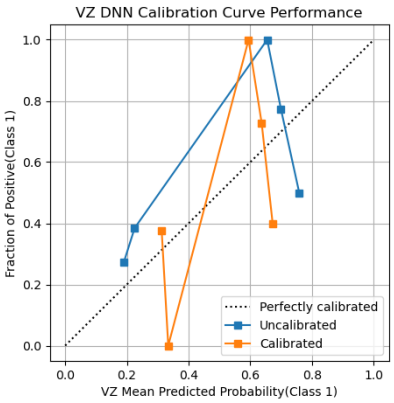

TSLA LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
TSLA Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

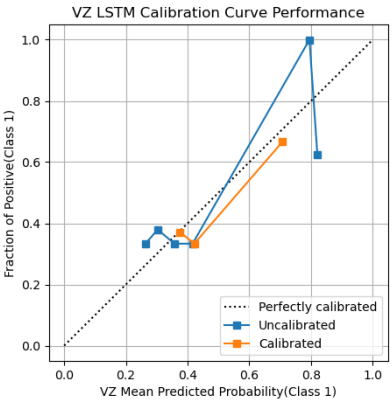

UNH DNN Calibration Curve Performance
Fraction of Positive(Class 1)
UNH Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

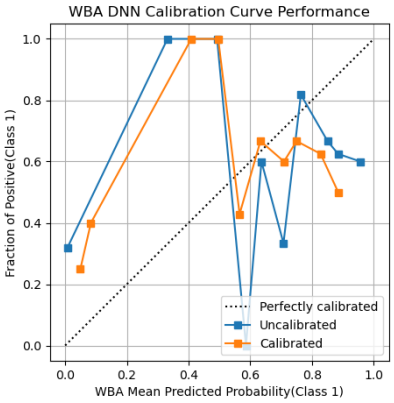

UNH LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
UNH Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

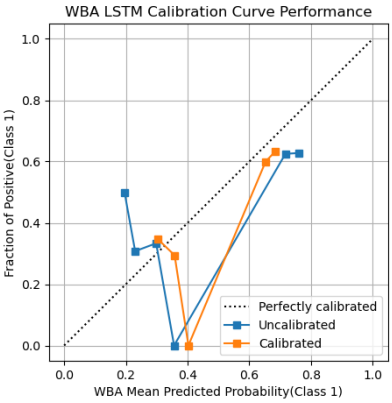

VLO DNN Calibration Curve Performance
Fraction of Positive(Class 1)
VLO Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

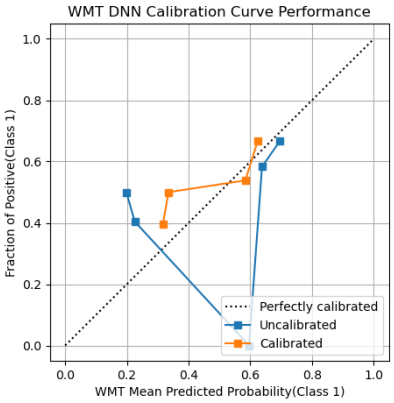

VLO LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
VLO Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

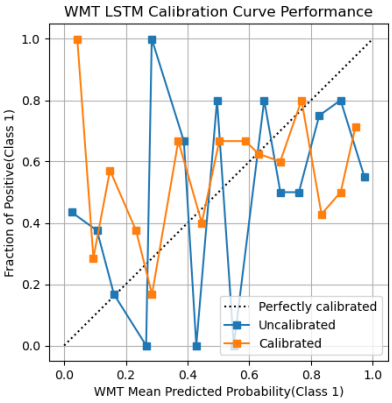

VZ DNN Calibration Curve Performance
Fraction of Positive(Class 1)
VZ Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

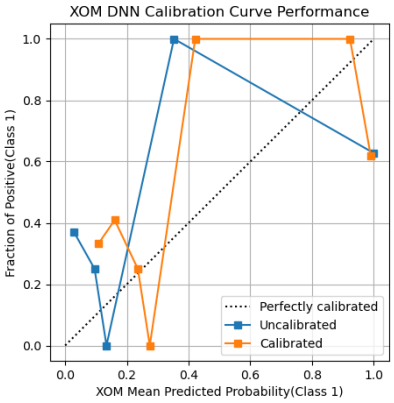

VZ LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
VZ Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated

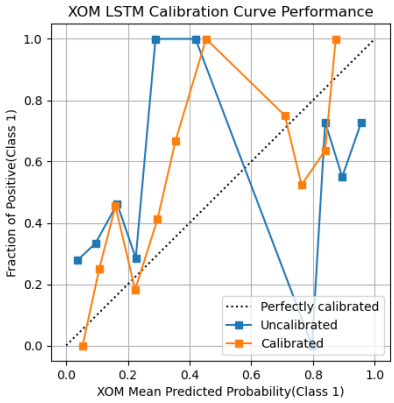

WBA DNN Calibration Curve Performance
Fraction of Positive(Class 1)
WBA Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated


WBA LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
WBA Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated


WMT DNN Calibration Curve Performance
Fraction of Positive(Class 1)
WMT Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated


WMT LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
WMT Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated


XOM DNN Calibration Curve Performance
Fraction of Positive(Class 1)
XOM Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated
0.0
0.2
0.4
0.6
0.8
1.0


XOM LSTM Calibration Curve Performance
Fraction of Positive(Class 1)
XOM Mean Predicted Probability(Class 1)
Perfectly calibrated
Uncalibrated
Calibrated
0.0
0.2
0.4
0.6
0.8
1.0