IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 31 August 2023, accepted 20 September 2023, date of publication 22 September 2023,
date of current version 28 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3318478

RESEARCH ARTICLE

Cost Harmonization LightGBM-Based Stock Market Prediction

XIAOSONG ZHAO, (Graduate Student Member, IEEE), YONG LIU, AND QIANGFU ZHAO, (Senior Member, IEEE)
Graduate School of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu, Fukushima 965-0006, Japan
Corresponding author: Xiaosong Zhao (d8232119@u-aizu.ac.jp)

ABSTRACT Stock market prediction (SMP) is a challenging task due to its uncertainty, nonlinearity, and volatility. Machine learning models, such as artificial neural networks (ANNs) and support vector regression (SVR), have been widely used for stock market prediction and achieved high performance in the sense of ''minimum errors.'' In the context of SMP, however, it is more meaningful to measure the performance using ''minimum cost.'' For example, a false positive error (FPE) could result in a big trading loss, while a false negative error (FNE) might just miss a chance. For a ''cautious'' investor, fewer FPEs are preferable. In fact, cost-sensitive learning has been used in areas such as fraud detection and medical diagnosis. In our earlier study, we proposed a false-sensitive method called focal-loss LightBGM (FL-LightGBM) for SMP by introducing a cost-aware loss in LightGBM, which is known to be a fast and efficient gradient-boosting learning algorithm for solving large-scale problems. FL-LightBGM, however, still assumes that all false negative errors (or false positive errors) contribute equally to the final cost. Such learned trading strategies might be useful only for an investor who is always ''aggressive'' or ''cautious.'' In practice, some errors may result in irreversible loss, so it is important to measure the cost based on ''data'' rather than the investor's character. In this paper, we propose a new method called cost-harmonization loss-based LightGBM (CHL-LightGBM), in which the cost for each datum can be calculated dynamically based on the difficulty of the datum. To verify the effectiveness of CHL-LightGBM, comparisons have been made among LightGBM, XGBoost, decision trees, FL-LightGBM, and CHL-LightGBM for stock predictions on data from Shanghai, Hong Kong, and NASDAQ Stock Exchanges. The simulation results show that although there is no significant difference between CHL-LightGBM and other models on the accuracy and winning rate, CHL-LightGBM obtained the highest annual return on all the test data.

INDEX TERMS Cost-harmonization loss function, cost-sensitive learning, LightGBM, stock market prediction.

I. INTRODUCTION

Stock market prediction (SMP) is often considered a challenging problem due to its uncertainty, nonlinearity, and volatility. Making accurate trading decisions has been difficult, even for professional investors. Over the past few decades, machine learning models such as artificial neural networks (ANNs) and support vector regression (SVR) have been widely used for SMP and have achieved high predictive accuracy. Concurrently, the inherent attributes of the stock market, typified by non-linear and non-stationary fluctuations, have engendered heightened interest in deep learning methodologies [1], [2] such as Convolutional Neural Networks (CNN) [3], Long Short-Term Memory (LSTM) networks [4], [5], Graph Neural Network (GNN) [6], Graph Attention Network (GAT) [7], and Hierarchical Adaptive Temporal-Relational Network(HATR) [8]. These deep learning paradigms have demonstrated their ascendancy over alternative predictive models within this domain, but their lack of interpretability can pose challenges, which is crucial for investor confidence [9]. For SMP, providing a more interpretable method for investors and minimizing

The associate editor coordinating the review of this manuscript and approving it for publication was Alberto Cano.

the cost of obtaining a certain amount of return is often more important than maximizing the accuracy. Research on cost-sensitive learning (CSL) has attracted widespread attention in recent years. CSL is a machine learning approach that considers the cost or penalty of misclassification errors. The main idea is not to minimize the number of errors but to reduce the total cost of the errors [10]. So far, CSL has been successfully used for fraud detection and medical diagnosis. CSL can help by assigning different costs to different types of misclassification errors, such as false positives and false negatives. For example, CSL can induce decision trees by incorporating cost information into the split criteria to reduce the expected cost of misclassification. Considering that false-positive and false-negative errors can lead to investment return losses during predicting investment decisions, CSL should be a feasible solution to minimize the total cost and maximize the profits. However, there is still a lack of related exploration in SMP.

LightGBM is a gradient-boosting framework that is now widely used in machine learning for handling large-scale datasets, including financial stock data. One reason that makes LightGBM efficient in handling big data is the introduction of gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB). GOSS is a sampling technique that helps to select only the most informative samples for each tree, while EFB combines features that have similar distribution into a single feature and reduces the number of features that need to be processed. By using these techniques, LightGBM can reduce the computational requirements of the algorithm and maintain high levels of accuracy. In addition, LightGBM applies a histogram-based method to split data and group values into discrete bins to enhance processing efficiency. Compared to traditional data splitting methods, such an approach helps to lower the overall memory requirements and computation time and lets LightGBM be a good choice for large-scale datasets.

In our previous study [11], the FL-LightGBM model was developed by replacing the default loss function of LightGBM with focal loss. During the optimization of model parameters, the prediction cost evaluation metric is reduced by adjusting the class balance parameter α and example difficulty parameter γ of Focal loss. By utilizing this approach, the model should not only focus on the difficulty of the example during training but also adjust the class distribution to minimize the overall cost of classification. This method for stock investment can decrease the number of false positives, reduce the loss resulting from misjudged trading, and enhance investment returns.

Traditional stock prediction methods consider false positive and false negative errors as ''equal'' and aim to decrease the overall number of errors. In the FL-LightGBM, the importance of false positive errors is given higher weight (with a higher cost), as purchasing a stock based on a fake chance can result in actual financial loss. In a certain sense, the trained model benefits relatively conservative users who desire to minimize risks and maximize returns.

In the present study, we incorporated CSL into the model training process. During the training, the example cost was dynamically calculated based on its difficulty, and this was utilized to balance the size of its gradient, ultimately enhancing the model's focus on high-cost examples.

By applying CSL to LightGBM, a new cost-harmonization loss (CHL) is proposed to design CHL LightGBM (CHL-LightGBM) for solving SMP in this paper. CHL-LightGBM extends LightGBM's default loss function to enable example-dependent CSL and reduce the overall cost loss by combining each example's cost factor and distance factor, where distance reflects the examples' difficulty. In example-dependent CSL, each example has a specific cost weight that depends on its class and other factors such as the example's importance and data source [12]. The cost weight reflects the risk that the model incurs in making decisions so that the learned model avoids any possible risks.

While the cost weight cannot fully reflect the importance of different examples, the model could focus more on examples with larger gradients in gradient descent-based learning. CHL introduces a distance factor to combine the gradient descent direction with the example's cost weight to better reflect the examples' difficulty. Therefore, CHL-LightGBM would pay more attention to examples with high-cost weights and large gradients during the learning. The learned CHL-LightGBM could more accurately predict the example's class by reducing the overall cost loss.

In this paper, the example-dependent cost-sensitive mechanism is introduced into the model training to reflect better the influence of example cost on the model decision. To understand the decision-making mechanism of CHL-LightGBM, we present the gradient-boosting tree algorithm in detail, analyze the principle of the cost harmonization loss function in-depth, and visualize the coordination between an example's cost weight and difficulty (impact on loss and gradient). Our study focuses on the significance of transparently explaining model decisions in the financial domain and the interpretability of the proposed approach; it aims to enhance trust in financial predictions made by machine learning models.

Three stock markets have been chosen to test the proposed CHL-LightGBM: the Shanghai Stock Exchange, the Hong Kong Stock Exchange, and the NASDAQ Stock Exchange. According to the efficient market hypothesis, those different stock markets may have unique characteristics that impact the model's accuracy. Testing the model's performance by using different stock markets would help to know whether the learned model would be robust in various market conditions. The three selected markets are in different development stages. Shanghai Stock Exchange is still in its early development stage, where financial companies account for a high proportion of market value, and retail investors hold more shares than institutional investors. Hong Kong Stock Exchange is considered a more mature and sophisticated market than Shanghai Stock Exchange, with a higher level of transparency and a greater focus on

international investors. It is the only market in the world where Chinese mainland investors, overseas investors, and Hong Kong local investors can participate, and the investor structure is mainly institutional investors. NASDAQ Stock Exchange is considered one of the world's most sophisticated and mature stock markets, with a strong focus on technology, biotech, and other high-growth sectors. Institutional investors hold more shares than retail investors.

In this paper, CHL-LightGBM was compared with XGBoost, Decision Trees, LightGBM, and Focal-loss LightGBM (FL-LightGBM) in four different aspects of prediction accuracy, cost-harmonizing performance, profitability, and risk control ability.

The main contributions of this study can be summarized as follows.

1) A new loss function, Cost-Harmonization Loss (CHL), which combines the cost of misclassifying examples with the hardness of classification, is proposed for optimizing the LightGBM model to better reflect the examples' difficulty.
2) The first-order and second-order derivative of CHL is derived in detail and embedded into LightGBM.
3) The effectiveness of CHL-LightGBM is validated for investment prediction in the stock market across three stages of development.
4) By comparing with other commonly used machine learning models such as XGBoost and Decision Trees, it is demonstrated that CHL-LightGBM has the best experiment performance.

II. LightGBM WITH COST-SENSITIVE LEARNING (CSL)

Cost-sensitive learning (CSL) is an efficient learning technique designed to address the problem posed by unbalanced misclassification costs. Conventional machine learning approaches typically assume equal costs associated with misclassification errors across all classes. Nevertheless, in numerous real-world scenarios, the costs of misclassification exhibit variability contingent upon the specific class being predicted. CLS aims to address this issue by adjusting the learning process to minimize the total expected cost, rather than just the misclassification errors [13]. The idea of ''minimax'' classification was originally for minimizing the maximum expected loss over all possible classes, which was regarded as a precursor to modern cost-sensitive learning techniques [14]. Later on, researchers began to develop more sophisticated CSL algorithms. Provost and Fawcett introduced CSL in decision trees, which adjusted the tree structure to minimize the expected cost of misclassification [15]. Fumera et al. applied CSL in support vector machine, which minimized a weighted combination of the misclassification rate and the cost of misclassification [16].

In recent years, CSL has been greatly applied in LightGBM. Tang et al. proposed a cost-sensitive LightGBM-based online fault detection model to optimize the misclassification loss [17]. Liu et al. proposed a cost-sensitive LightGBM (LightGBM-focal) for the imbalanced credit scoring problem [18]. They associated the misclassification cost and classification hardness to focal loss and embedded it into LightGBM, which let LightGBM be focal-awarded and cost-sensitive. LightGBM had been used to train a model to maximize diagnostic accuracy while minimizing the cost of misclassification [19]. In medical diagnosis, misclassifying a patient's condition could have serious consequences. CSL would be useful to build a model that minimizes the cost of misclassification. For example, the cost of misclassifying a patient with a life-threatening condition as healthy could be much higher than the cost of misclassifying a healthy patient as having a minor condition. Zhao et al. used LightGBM and technical indicators to predict the investment chances and proposed a concept called cost awareness to give the model more attention to the misclassification errors [20]. CSL has also been shown to be helpful in predicting stock price movements [21].

A. LightGBM

LightGBM (Light Gradient Boosting Machine) [22] is an open-source gradient-boosting framework that uses tree-based learning algorithms. It was developed by Microsoft and is known for its high accuracy, speed, and scalability. Both LightGBM and XGBoost [23] are frameworks for implementing gradient to boost decision trees [24]. LightGBM uses a gradient-based one-side sampling (GOSS) technique and exclusive feature bundling (EFB) to reduce training time while maintaining high accuracy. It also uses histogram-based algorithms for binning data and a leaf-wise growth strategy to build trees, which results in faster convergence and better performance [25]. GOSS is a method for efficient gradient boosting with large data sets. It works by sampling the data points based on their gradients, where data points with smaller gradients have a lower probability of being selected in training. Therefore, GOSS reduces the number of data points used for training while maintaining the overall performance of the model [26].

EFB, on the other hand, is a feature bundling algorithm that combines similar features to reduce the number of features used for training. By bundling features together, EFB can help reduce the computational cost of training and improve the accuracy of the model [27]. Combined with other techniques like histogram-based binning and leaf-wise tree growth, GOSS and EFB have helped LightGBM achieve state-of-the-art performance on various machine-learning tasks, especially for large-scale and high-dimensional data sets. Histogram-based algorithms are used in LightGBM to bin data into discrete intervals, which reduces memory usage and computational cost. The leaf-wise growth strategy is used to build trees in LightGBM, where the algorithm selects the leaf with the largest gradient to split, resulting in faster convergence and better performance.

The core of the LightGBM algorithm is to obtain the optimal model by iteratively training weak classifiers. Referring to XGBoost and XGBoost Documentation, the

basic principle of the LightGBM algorithm is described as follows.

For a given dataset of N examples and m features $D = \{(\mathbf{x}_i, \mathrm{y}_i)\}(|D| = N, \mathbf{x}_i \in \mathbb{R}^m, \mathrm{y}_i \in \mathbb{R})$, a tree ensemble model uses K additive functions to predict the output.

$$\hat{\mathrm{y}}_i = \phi(\mathbf{x}_i) = \sum_{k=1}^{K} f_k(\mathbf{x}_i),\ f_k \in \mathcal{F} \tag{1}$$

where f_k is a function in the functional space $\mathcal{F}$. The function f_k is defined as

$$f_k(\mathbf{x}_i) = \mathbf{w}_{q(\mathbf{x}_i)},\ \mathbf{w} \in \mathbb{R}^T,\ q : \mathbb{R}^m \to \{1, 2, \cdots, T\}.$$

Here $\mathbf{w}$ is the vector of scores on leaves, q is a function assigning each data point to the corresponding leaf, and T is the number of leaves. $\mathcal{F}$ captures the set of all possible functions based on different combinations of q and $\mathbf{w}$. The objective function to be optimized is given by

$$O(\phi) = \sum_{i=1}^{N} L(\mathrm{y}_i, \hat{\mathrm{y}}_i) + \sum_{k=1}^{K} \omega(f_k) \tag{2}$$

where L is a loss function that measures the difference between the raw prediction $\hat{\mathrm{y}}_i$ and the target y_i. $\omega(f_k)$ is the regularization term, which is used to penalize the complexity of the model. Its definition is as follows.

$$\omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{k=1}^{T} w_k^2$$

where γ and λ are hyperparameters.

The model is trained in an additive manner, $\hat{\mathrm{y}}_i^{(t)}$ is the prediction of the i-th example at the t-th iteration,

$$\begin{aligned}
\hat{\mathrm{y}}_i^{(0)} &= 0 \\
\hat{\mathrm{y}}_i^{(1)} &= f_1(\mathbf{x}_i) = \hat{\mathrm{y}}_i^{(0)} + f_1(\mathbf{x}_i) \\
\hat{\mathrm{y}}_i^{(2)} &= f_1(\mathbf{x}_i) + f_2(\mathbf{x}_i) = \hat{\mathrm{y}}_i^{(1)} + f_2(\mathbf{x}_i) \\
&\cdots \\
\hat{\mathrm{y}}_i^{(t)} &= \sum_{k=1}^{t} f_k(\mathbf{x}_i) = \hat{\mathrm{y}}_i^{(t-1)} + f_t(\mathbf{x}_i)
\end{aligned}$$

f_t is added to minimize the following objective:

$$\begin{aligned}
O^{(t)} &= \sum_{i=1}^{N} L(\mathrm{y}_i, \hat{\mathrm{y}}_i^{(t)}) + \sum_{k=1}^{t} \omega(f_k) \\
&= \sum_{i=1}^{N} L(\mathrm{y}_i, \hat{\mathrm{y}}_i^{(t-1)} + f_t(\mathbf{x}_i)) + \sum_{k=1}^{t-1} \omega(f_k) + \omega(f_t) \\
&= \sum_{i=1}^{N} L(\mathrm{y}_i, \hat{\mathrm{y}}_i^{(t-1)} + f_t(\mathbf{x}_i)) + \omega(f_t) + \eta
\end{aligned} \tag{3}$$

where $\eta = \sum_{k=1}^{t-1} \omega(f_k)$.

The f_t that most improve the model according to Eq.(2) is greedily added. The second-order approximation can be used to quickly optimize the objective in the general case [28].

$$\begin{aligned}
O^{(t)} = \sum_{i=1}^{N} &[L(\mathrm{y}_i, \hat{\mathrm{y}}_i^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)] \\
&+ \omega(f_t) + \eta
\end{aligned} \tag{4}$$

where the g_i and h_i are defined as

$$\begin{aligned}
g_i &= \partial_{\hat{\mathrm{y}}_i^{(t-1)}} L(\mathrm{y}_i, \hat{\mathrm{y}}_i^{(t-1)}), \\
h_i &= \partial^2_{\hat{\mathrm{y}}_i^{(t-1)}} L(\mathrm{y}_i, \hat{\mathrm{y}}_i^{(t-1)}),
\end{aligned}$$

which are first-order and second-order gradient statistics on the loss function.

When training the mode at t step, $L(\mathrm{y}_i, \hat{\mathrm{y}}_i^{(t-1)})$ and η have been determined, which can be regarded as constants, and have no effect on the optimization of the objective function. Removing the constant terms obtain the following simplified objective:

$$O^{(t)} \simeq \sum_{i=1}^{N} [g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)] + \omega(f_t) \tag{5}$$

This becomes the optimization goal for the new tree. An important advantage of this definition is that the value of the objective function only depends on g_i and h_i. This is how LightGBM supports custom loss functions.

B. COST-SENSITIVE LEARNING (CSL)

CSL is a machine learning approach that considers the cost of misclassification errors when training a model. In many real-world scenarios, misclassifying certain types of examples can be more costly than others. For example, misclassifying a patient as healthy when they have a serious medical condition could have severe consequences. CSL can be divided into example-dependent and class-dependent in terms of processing methods. Example-dependent cost-sensitive learning is a technique where the cost of misclassification is calculated for each example individually based on its specific characteristics. This approach is advantageous because it allows the model to learn more about the specific types of more costly errors in a given context. In contrast, class-dependent cost-sensitive learning involves assigning a cost to each class label rather than to individual examples. This approach is less flexible than example-dependent cost-sensitive learning because it assumes that misclassification costs are the same for all examples of a given class [29].

In example-dependent cost-sensitive learning, the cost of misclassifying an example can vary depending on the particular example. The cost can be represented by a cost matrix. For binary classification problems, the cost matrix represents the costs associated with different types of errors. The matrix rows represent the true class labels (positive and negative), while the columns represent the predicted class labels. The matrix elements represent the cost of making each type of error. Table 1 shows a typical binary classification cost matrix, where C_{FP_i} (Cost of False Positive example

TABLE 1. Binary classification cost matrix.

	predict negative	predict positive
actual negative	$C(0\|0) = 0$	$C(1\|0) = C_{FP_i}$
actual positive	$C(0\|1) = C_{FN_i}$	$C(1\|1) = 0$

$\mathbf{x}_i$) is defined as the cost of misclassifying an example that truly belongs to class 0 as belonging to class 1 (i.e., a false positive). C_{FN_i} (Cost of False Negative example $\mathbf{x}_i$) is defined as the cost of misclassifying an example that does not belong to class 0 as belonging to class 0 (i.e., a false negative). By incorporating C_{FP_i} and C_{FN_i} into the learning algorithm, we can adjust the classification decisions to minimize the expected misclassification cost, while traditional classifier decisions minimize misclassification errors.

The method of calculating the cost matrix depends on the CSL application field. To predict the return on stock investment, we assign C_{FP_i} to represent investment loss, while C_{FN_i} represents profit loss due to missed investment opportunities. The investment loss equals the loss caused by the stock price decline. The profit loss equals the return generated by the rise in stock price. Each example involves the above two costs in the binary example-dependent cost matrix, and how to choose them is determined by the result of misclassification. C_{FP_i} and C_{FN_i} are constants for each example, although their values may fluctuate across examples.

Similar to traditional classification methods, to solve the problem of cost-sensitive classification, a cost-sensitive loss function can be constructed to minimize the classification cost by optimizing cost-sensitive risk [30]. Given the misclassification cost matrix C, as defined in Table 1, the prediction loss is calculated using a cost-sensitive loss function $L_C(\mathrm{y}, f(\mathbf{x}))$. The cost-sensitive decision function $\hat{F}_C$ is obtained by optimizing the cost-sensitive loss (cost-sensitive risk) $E_{\mathrm{y},\mathbf{x}}[L_C(\mathrm{y}, F(\mathbf{x}))]$ in the example space.

$$\hat{F}_C = \arg\min_{\mathrm{F}} E_{\mathrm{y},\mathbf{x}}[L_C(\mathrm{y}, F(\mathbf{x}))] \tag{6}$$

The cost-sensitive loss function can be obtained by modifying the standard loss function. According to specific requirements, modulation constraint terms and weight conditions are added to the standard loss function so that the optimization trend of the objective function is biased towards the direction designed in advance. The cost-sensitive classification is converted to minimize the optimization objective:

$$L_C = \sum_{i=1}^{N} P(\mathrm{Y} = \mathrm{y} \mid \mathbf{x}_i) C_i \tag{7}$$

where $P(\mathrm{Y} = \mathrm{y} \mid \mathbf{x}_i)$ denotes the posterior probability that example $\mathbf{x}_i$ belongs to class y. Since the loss weight bias factor C_i is added to the optimization objective, the model no longer focuses on how to obtain the maximum posterior probability of $P(\mathrm{Y} = \mathrm{y} \mid \mathbf{x}_i)$ but balances the prediction result and the cost loss caused by the misclassification.

A common way to express cost-sensitive loss is based on binary cross-entropy loss. The equation can be written as follows.

$$L_C = -\sum_{i=1}^{N} C_i \left[\mathrm{y}_i \log(\mathrm{p}_i) + (1 - \mathrm{y}_i) \log(1 - \mathrm{p}_i)\right] \tag{8}$$

where N is the number of examples in the dataset, y_i is the true label of example $\mathbf{x}_i$, p_i is the predicted probability scores, and C_i is the cost associated with misclassifying example $\mathbf{x}_i$.

III. LightGBM WITH COST-HARMONIZATION LOSS (CHL)

Instead of using the LightGBM default function, CHL-LightGBM uses a new cost-harmonization loss function to improve stock investment returns. During training, the new loss function affects the feature splitting gain and the leaf weight, which leads the training to pay more attention to misclassified examples with significant cost weight for improving the model's accurate prediction of stock investment opportunities.

Investment opportunity identification is the classification of potential stocks that may be profitable. The prediction rate given by the model decides whether to approve or decline an investment. According to the algorithm of the gradient boosting decision trees (GDBT), the hard example with a lower prediction rate is more likely to be misclassified. This type of example produces a greater gradient than others, drawing more attention from the model. Nevertheless, extremely hard examples that can be regarded as outliers may compromise the model's stability, as their gradients deviate significantly from those of other examples [31]. Due to their small gradient contribution, well-classified examples are often ignored. However, it is important to consider easy examples to prevent weakening the model's profitability in stock investment. Therefore, to solve these problems well, it is necessary to improve the traditional classification method.

In the research on stock investment, we find that the cost losses of misclassified examples are unbalanced. False positive examples bring real cost loss, while false negative examples only lose the profit opportunity; thus, the false positive costs C_{FP_i} should be greater than the false negative costs C_{FN_i}. Therefore, stock market prediction belongs to typical cost-sensitive learning.

There are many ways to solve the cost-sensitive problem in the stock investment field. The class-dependent cost-sensitive approach minimizes the total cost of misclassification by adjusting the balance boundary of positive and negative classes [11], [20]. This approach focuses on minimizing the number of false positive examples, thereby reducing failed transactions. This approach is based on the assumption that examples of a given class have the same cost, but the reality is that the cost of misclassification of each example is different.

For false positive examples, we compare two results, one with a larger proportion of high-cost examples and the other with a higher proportion of low-cost examples. The first scenario leads to more losses for investors. Therefore, the purpose of our research is to improve the classifier with example-dependent cost-sensitive learning so that the model can be sensitive to high-cost examples and minimize the loss of stock investment.

In this paper, the cost-sensitive approach is concisely and effectively integrated into the gradient-boosting algorithm by taking advantage of the excellent performance of LightGBM in ensemble learning. We propose a novel loss function inspired by example-dependent cost-sensitive learning, which uses a cost harmonization factor to balance example cost weights and difficulty. This approach forces the model to down-weight examples with low cost while focusing on examples with high cost.

A. COST-HARMONIZATION LOSS (CHL) FUNCTION

The proposed loss function is based on the binary cross-entropy loss by introducing a cost harmonization factor β_i. We name it cost-harmonization loss (CHL). The mathematical definition of CHL is given by the following equation:

$$L_{CH}(\mathrm{y}, \mathrm{p}) = -\sum_{i=1}^{N} \beta_i [\mathrm{y}_i \log(\mathrm{p}_i) + (1 - \mathrm{y}_i) \log(1 - \mathrm{p}_i)] \quad (9)$$

where β_i is the cost harmonization factor, y_i is the true label, and p_i is the predicted probability of the positive class.

We define a variable δ_i that reflects the distance between the predicted probability p_i and the ground-truth label y_i. Examples with larger distances are misclassified as difficult examples, and those with smaller distances are correctly classified as easy examples.

$$\delta_i = \mid \mathrm{y}_i - \mathrm{p}_i \mid = \begin{cases} 1 - \mathrm{p}_i & \text{if } \mathrm{y}_i = 1 \\ \mathrm{p}_i & \text{if } \mathrm{y}_i = 0 \end{cases} \quad (10)$$

It can be observed from the above formula that the distance δ_i is equivalent to the gradient of the cross-entropy loss. The distance reflects the difficulty of the examples.

The cost harmonization factor depends on a function $\theta(\mathrm{y}_i, \mathrm{p}_i, C_i)$, where C_i is the cost matrix. In this study, the example-dependent cost matrix is transformed into sample weight. For binary classification, the expression for calculating the sample weight (S_i) is as follows.

$$S_i = \begin{cases} (1 - \mathrm{p}_i) \times C_{FN_i} & \text{if } \mathrm{y}_i = 1 \\ \mathrm{p}_i \times C_{FP_i} & \text{if } \mathrm{y}_i = 0 \end{cases} \quad (11)$$

The part of the Eq.(11) related to the predicted probability p_i is equal to the distance δ_i. The other part represents the example cost weight ζ_i, which is only related to y_i and the cost matrix C_i.

$$\zeta_i = \begin{cases} C_{FN_i} & \text{if } \mathrm{y}_i = 1 \\ C_{FP_i} & \text{if } \mathrm{y}_i = 0 \end{cases} \quad (12)$$

Then the cost harmonization factor β_i is converted to the product of the example cost weight ζ_i and the distance δ_i.

$$\beta_i = \zeta_i \times \delta_i \quad (13)$$

Replacing β_i in Eq.(9) with ζ_i and δ_i, CHL is converted to the following equation:

$$L_{CH}(\mathrm{y}, \mathrm{p}) = -\sum_{i=1}^{N} \zeta_i [\mathrm{y}_i (1 - \mathrm{p}_i) \log(\mathrm{p}_i) + (1 - \mathrm{y}_i) \mathrm{p}_i \log(1 - \mathrm{p}_i)] \quad (14)$$

To simplify the notation, a variable p_i^* is introduced.

$$\mathrm{p}_i^* = \begin{cases} \mathrm{p}_i & \text{if } \mathrm{y}_i = 1 \\ 1 - \mathrm{p}_i & \text{if } \mathrm{y}_i = 0 \end{cases} \quad (15)$$

Then Eq.(14) forms can be unified as:

$$L_{CH}(\mathrm{p}^*) = -\sum_{i=1}^{N} \zeta_i (1 - \mathrm{p}_i^*) log(\mathrm{p}_i^*) \quad (16)$$

By splitting the equation Eq.(11), the example cost weight ζ_i can be calculated separately before model training. It reduces training time and simplifies the calculation of the derivative of the loss function. It provides a more direct meaning for cost weight ζ_i and distance δ_i, making the loss function more interpretable. The calculation and effect of the example cost, the cost harmonization factor β_i, are explained as follows.

In the model training stage, the example cost is determined by the target label, the predicted probability of the positive class, and the cost matrix. For binary classification, the example cost can be defined as the product of cost weight and difficulty. The cost weight is determined by the target label. It is C_{FN_i} if the example belongs to the positive class; otherwise, it is C_{FP_i}. The difficulty of an example is the absolute value of the difference between its target label and the predicted probability. In each model training step, the dynamic change of the prediction result determines the change of the difficulty, affecting the change of the example cost. The example cost balances the calculation results of the gradient, which affects the optimization of the objective function. Finally, under the interaction of the cost weight ζ_i and the difficulty δ_i, the cost-sensitive classifier enhances the attention to high-cost examples and reduces the accumulated misclassification cost.

Fig. 1 and Fig. 2 are schematic diagrams of the changes of loss and gradient with the probability of ground truth class for an example with different cost weight, which randomly select different predictive values p^* between 0.1 and 1, and observe the changes of loss L_{CH} and its gradient under different normalized ζ values. The benchmark is the binary cross-entropy loss function L_{BCE}. In Fig. 1 and Fig. 2, it can be observed that, for hard examples ($\mathrm{p}^* < 0.5$), the loss or gradient with a small cost weight is significantly reduced, while the loss or gradient with a larger cost weight is closer to L_{BCE}. In this way, the model pays more attention to hard

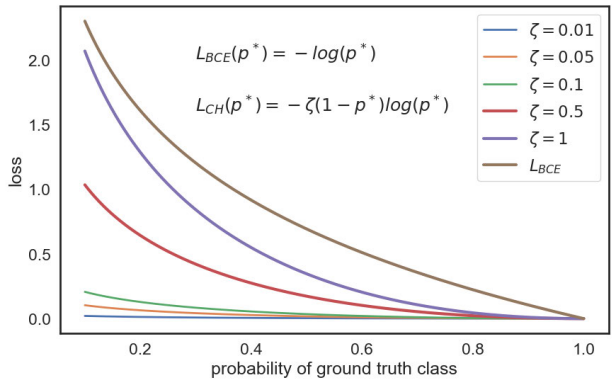

FIGURE 1. The change of loss with the probability of ground truth class for an example with different cost weights.

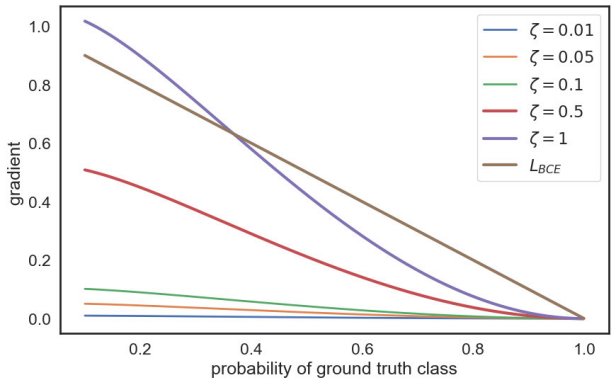

FIGURE 2. The change of gradient with the probability of ground truth class for an example with different cost weights.

and high-cost weight examples. The experiment results show that our proposed cost-sensitive model can effectively reduce the global investment cost and improve profitability.

B. LightGBM WITH CHL

According to the algorithm principle of LightGBM, as shown in Eq.(5), the optimal objective only contains the gradient and second-order gradient of the loss function. In order to achieve an algorithm for example-dependent cost-sensitive LightGBM, we substitute the gradient and second-order gradient of the cost-harmonization loss function Eq.(16).

Given the provided training data and a set of raw predictions $\hat{y}_i (i = 1, \cdots, N)$ before logistic transformation, the predicted probability scores are defined as

$$p_i = \frac{1}{1 + e^{-\hat{y}_i}} \tag{17}$$

The first-order derivative of the cost harmonization loss function is calculated following the chain rule:

$$g_i = \frac{\partial L_{CH}}{\partial \hat{y}_i} = \frac{\partial L_{CH}}{\partial p_i^*} \times \frac{\partial p_i^*}{\partial p_i} \times \frac{\partial p_i}{\partial \hat{y}_i} \tag{18}$$

The first part of the chain is the most complicated part, as it is the derivative of a product of three elements:

$$\begin{aligned} \frac{\partial L_{CH}}{\partial p_i^*} &= \zeta_i \log(p_i^*) - \frac{\zeta_i(1-p_i^*)}{p_i^*} \\ &= \zeta_i(\frac{p_i^* \log(p_i^*) + p_i^* - 1)}{p_i^*}) \end{aligned} \tag{19}$$

The second part of the chain is the derivative of p_i^* with respect to p_i. The trick is to flatten p_i^* into a differentiable formula:

$$p_i^* = \frac{p_i(y_i^* + 1)}{2} + \frac{(1-p_i)(1-y_i^*)}{2} \tag{20}$$

Note that the above equation assumes $y_i^* \in \{-1, 1\}$, here $y_i^* = 2y_i - 1$. The derivative then is deduced as:

$$\begin{aligned} \frac{\partial p_i^*}{\partial p_i} &= \frac{y_i^* + 1}{2} + \frac{y_i^* - 1}{2} \\ &= \frac{2y_i^*}{2} \\ &= y_i^* \end{aligned} \tag{21}$$

Finally, the last part of the chain is the well-known gradient of the sigmoid function, thus:

$$\frac{\partial p_i}{\partial \hat{y}_i} = p_i(1-p_i) \tag{22}$$

Because $p_i(1-p_i)$ is equal to $p_i^*(1-p_i^*)$. Therefore, once the chain parts are multiplied with each other, some terms are canceled out:

$$\begin{aligned} g_i &= \frac{\partial L_{CH}}{\partial \hat{y}_i} \\ &= \zeta_i(\frac{p_i^* \log(p_i^*) + p_i^* - 1)}{p_i^*}) \times y_i^* \times p_i^*(1-p_i^*) \\ &= \zeta_i \times y_i^* \times (1-p_i^*)(p_i^* log(p_i^*) + p_i^* - 1) \end{aligned} \tag{23}$$

The calculation of the second-order derivative of the cost-harmonization loss function still follows the rule:

$$\begin{aligned} h_i &= \frac{\partial^2 L_{CH}}{\partial \hat{y}_i^2} \\ &= \frac{\partial}{\partial \hat{y}_i}(\frac{\partial L_{CH}}{\partial \hat{y}_i}) \\ &= \frac{\partial}{\partial p_i^*}(\frac{\partial L_{CH}}{\partial \hat{y}_i}) \times \frac{\partial p_i^*}{\partial p_i} \times \frac{\partial p_i}{\partial \hat{y}_i} \end{aligned} \tag{24}$$

After computing the chain's last two parts, which are equal to y_i^* and $p_i(1-p_i)$. To make the calculation easier, We defined u_i and v_i based on Eq.(23).

$$\frac{\partial L_{CH}}{\partial \hat{y}_i} = u_i \times v_i \tag{25}$$

$$u_i = \zeta_i \times y_i^* \times (1-p_i^*) \tag{26}$$

$$v_i = p_i^* \times log(p_i^*) + p_i^* - 1 \tag{27}$$

The derivatives of u_i and v_i with respect to p_i^* are quite straightforward:

$$\frac{\partial u_i}{\partial p_i^*} = -\zeta_i \times y_i^* \tag{28}$$

$$\frac{\partial v_i}{\partial p_i^*} = log(p_i^*) + 2 \tag{29}$$

The second-order derivative is thus:

$$\begin{aligned} h_i &= \frac{\partial^2 L_{CH}}{\partial \hat{y}_i^2} \\ &= (\frac{\partial u_i}{\partial p_i^*} \times v_i + u_i \times \frac{\partial v_i}{\partial p_i^*}) \times \frac{\partial p_i^*}{\partial p_i} \times \frac{\partial p_i}{\partial \hat{y}_i} \\ &= [-v_i \times \zeta_i \times y_i^* + u_i \times log(p_i^*) + 2u_i] \\ &\quad \times y_i^* \times p_i^* \times (1 - p_i^*) \end{aligned} \tag{30}$$

Algorithm 1 Cost Harmonization Loss Function for LightGBM

Input:
y_pred: prediction
y_true: correct labels
cost_mat: cost matrix
Output:
grad: first-order derivative
hess: second-order derivative
function ch_loss(*y_pred*, *y_true*, *cost_mat*)
$p \leftarrow 1/(1 + np.exp(-y\_pred))$
$y \leftarrow y\_true$
$C_{fp} \leftarrow cost\_mat[:, 0]$
$C_{fn} \leftarrow cost\_mat[:, 1]$
$\zeta \leftarrow C_{fn} \times y + C_{fp} \times (1 - y)$
$y^* \leftarrow 2 \times y - 1$
$p^* = p \times (y^* + 1)/2 + (1 - p) \times (1 - y^*)/2$

$grad \leftarrow \zeta \times y^* \times (1 - p^*) \times [p^* \times log(p^*) + p^* - 1]$
$u \leftarrow \zeta \times y^* \times (1 - p^*)$
$v \leftarrow y^* \times p^* \times log(p^*) + p^* - 1$
$hess \leftarrow [-v \times \zeta \times y^* + u \times log(p^*) + 2u]$
$\times y^* \times p^* \times (1 - p^*)$
return *grad*, *hess*
end function

The LightGBM framework provides an interface specification for custom loss functions. Using Algorithm 1, it is convenient to replace the default function of LightGBM with the cost-harmonization loss function and construct the cost-harmonization loss-based LightGBM.

IV. PREPROCESSING FOR DATA FROM THREE STOCK MARKETS

Data used in the simulation were selected from three stock markets, including the Shanghai Stock Exchange, the Hong Kong Stock Exchange, and the NASDAQ Stock Exchange. The effectiveness of the prediction model may be affected by the state of the market [32]. To address this, incorporating data from various market conditions could resolve this issue. The data used in this thesis is taken from the Tushare data platform (https://tushare.pro). Tushare provides a free data interface, including stock, financial statement data, indexes, funds, futures, macroeconomic data, etc.

We obtained the daily historical data of individual stocks in three stock markets from the platform, which spans from 2013 to 2021, including the historical data of 1,665 stocks (main board) in the Shanghai market, 2,065 stocks in the Hong Kong market, and 2,131 stocks in the NASDAQ market. We also selected the historical data of the Shanghai Stock Exchange Composite Index (SSE Composite Index), Hang Seng Index (HSI), and NASDAQ Composite Index (IXIC).

By testing models on various markets, we can evaluate their performance across various market conditions and investor behaviors. Each market possesses distinct features and factors influencing stock prices. The use of multiple markets in the assessment of prediction models strengthens their applicability in real-world scenarios. Stock markets are interconnected and influenced by global factors, making it essential for models to demonstrate robustness across various market settings. Models that exhibit consistent performance across multiple markets are more likely to be reliable in practical investment decision-making processes. This robustness ensures that the models can effectively capture and analyze market dynamics, enhancing their practical utility and credibility.

Fig. 3 shows the experiment process of the proposed model CHL-LightGBM.

A. DATA PREPROCESSING

Financial data has been proven as noisy and contain a lot of outliers and missing values, which can negatively affect the performance of machine learning models. However, machine learning algorithms require high-quality, relevant, well-structured data to produce accurate predictions. That means the data processing section is an essential step for this research [33]. To complete data processing, we removed the missing value, and then z-score was implemented to detect the outliers and standardization.

The z-score is a statistical measuring method that indicates how many standard deviations a data point is from the mean of a dataset. It is calculated by subtracting the mean of the dataset from the data point and then dividing it by the standard deviation of the dataset. In the outlier detection part, z-score has been applied by considering any data point with a z-score greater than a certain threshold to be an outlier. If the certain z-score is greater than the threshold, it indicates that the data point is significantly far from the mean of the dataset and is, therefore, likely to be an anomaly. At the same time, using the z-score method to complete data standardization. The whole approach can be described as transforming data with a mean of zero and a standard deviation of one. This transformation is achieved by subtracting the dataset's mean from each

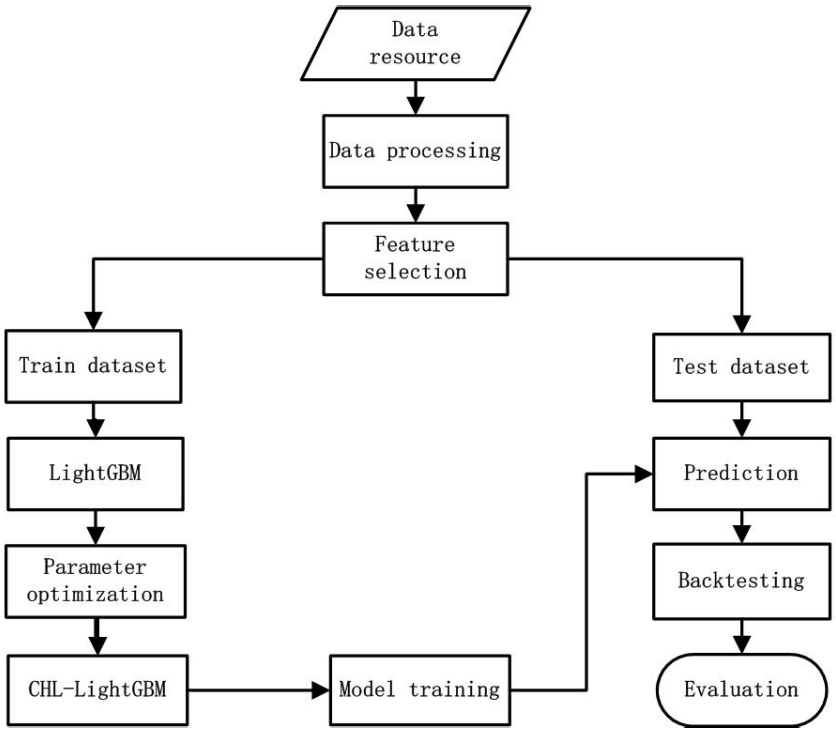

FIGURE 3. The experiment process of the proposed model CHL-LightGBM is generally divided into multiple steps, including data processing, feature selection, dataset splitting, parameter optimization, model training, prediction, backtesting, and evaluation.

data point and dividing it by the standard deviation of the dataset. The standardization of data confers the advantage of facilitating the comparison of data points measured on different scales, making it easier to identify patterns and relationships between variables. the formula for computing the z-score for a data point x in a dataset with mean μ and standard deviation σ is as follows.

$$z = \frac{\mathbf{x} - \mu}{\sigma}$$

B. FEATURE SELECTION

Feature engineering is the process of creating new features from raw data to improve the model's predictive performance. This process includes feature extraction, feature selection, and data denoising. We selected 3 different sets of features, which are OHLC indicators, financial indicators, and technical indicators [34]. OHLC indicators are the daily open price, high price, low price, and close price.

Technical indicators are based on historical price and volume data and are used to analyze patterns and trends in the market. We use the TA-Lib tool (https://ta-lib.github.io/ta-lib-python/index.html) to extract technical indicators related to short-term investment, including overlap studies, momentum, volume, cycle, and volatility indicators [35]. For indicators with time parameters, we set the ''timeperiod'' to 5 days and 10 days, respectively. In this way, we extracted a total of more than 40 technical indicators. It has been proved that these features effectively identify patterns and relationships that may influence future stock prices [36]. Part of the technical indicators that have been put into use are shown in Table 2. We also selected market indices and financial indicators. The financial indicators include the p/e ratio, p/b ratio, and price sales ratio.

Four filtering methods have been used to remove irrelevant features:

1) filter the features with missing values [37];
2) filter the features with unique value [38];
3) filter the features with high correlation with other features [39];
4) filter the features with low importance [40].

During the process of filtering the highly correlated features, we implemented the computation of Pearson Correlation coefficient to determine the interrelationships between the features [41]. It was observed that highly correlated features were associated with computational inefficiencies and a propensity for overfitting. Consequently, variables displaying a correlation of 90% were eliminated from the dataset. Finally, features demonstrating minor importance were eliminated as they were deemed to have no discernible impact on the model. After data processing and feature selection, we generated 38 features as input variables.

C. DATA LABELING

In the domain of stock market prediction, a common approach for classification involves assigning labels to examples based on the changes in the closing price of stocks. Specifically,

TABLE 2. Description of technical indicators.

Name	Definition
CMO	Chande Momentum Oscillator (CMO) is an indicator that calculates the difference between the sum of recent gains and the sum of recent losses and then divides the result by the sum of all price movements over the same period. CMO = 100 × ((Su - Sd) ÷ (Su + Sd)). where: Su = Sum of the difference between the current close and previous close on up days for the specified period. Up days are days when the current close is greater than the previous close. Sd = Sum of the absolute value of the difference between the current close and the previous close on down days for the specified period. Down days are days when the current close is less than the previous close.
CCI	The Commodity Channel Index (CCI) is an indicator that measures the difference between the current price and the historical average price. CCI = (Typical Price - SMATP) ÷ (0.015 × Mean Deviation). where: SMATP = Simple MA(20) applied to the Typical Price.
SAR	As known as the 'stop and reversal system,' SAR is an indicator that detects the price direction of an asset and decides how to distribute the attention to the signal of price direction changes.
KAMA	Kaufman's Adaptive Moving Average (KAMA) is a moving average designed to account for market noise or volatility.
ADX	Average directional index(ADX), an indicator representing the strength of a price trend.
MOM	Momentum can be regarded as the ratio of stock price changes over a period of time. MOM = Price - Price of n periods ago
ROC	As known as price rate of change, measure the scale of change in price between the current price and the price of a certain historical price. ROC = $((\text{Closing Price}_p - \text{Closing Price}_{p-n}) \div \text{Closing Price}_{p-n}) \times 100$. where: Closing Price_p = Closing price of most recent period. $\text{Closing Price}_{p-n}$ = Closing price n periods before most recent period.

two classes are typically employed: "up" and "down." An example is classified as "up" if the closing price of a stock rises the following day, while it is labeled as "down" if the price decreases. However, this conventional labeling method overlooks the impact of transaction charges associated with market activities. Consequently, examples that incur losses may be mistakenly labeled as profitable if the investment gains fail to cover the transaction costs, such as stamp duty.

To address this limitation, this research paper introduces a novel labeling scheme incorporating a threshold value denoted as T_R. This threshold is determined based on the stock's return rate (R). Specifically, if the return rate R exceeds the predefined threshold T_R, the corresponding example is categorized as belonging to the profit class. Conversely, the example is classified as a loss if R is less than or equal to T_R. Notably, throughout the experiment analysis conducted in this study, a uniform value of $T_R = 0.3\%$ is employed as the threshold.

$$Label = \begin{cases} 1 & \text{if } R > T_R \\ 0 & \text{if } R <= T_R \end{cases}$$

The return rate of the stock is calculated according to the following formula:

$$R = \frac{next\_price}{close\_price} - 1$$

where *close_price* is the stock closing price, and *next_price* is the stock closing price the next day.

By adopting the return label strategy, we can ensure that the labeling strategy does not influence the potential loss incurred through stock trading. This separation of losses from labeling decisions maintains the integrity of the evaluation process.

TABLE 3. Dataset splitting.

	Training dataset	Test dataset
Group 1	2013-01-01 to 2019-12-31	2020-01-01 to 2020-12-31
Group 2	2014-01-01 to 2020-12-31	2021-01-01 to 2021-12-31

D. DATASET SPLITTING

Table 3 presents the dataset splitting for the stock market prediction analysis. The table outlines data division into training and test datasets for two groups. In Group 1, the training dataset contains the period from January 1, 2013, to December 31, 2019, representing seven years of historical data. This dataset is utilized for training the prediction model and optimizing its parameters. The corresponding test dataset for Group 1 covers the year 2020, from January 1 to December 31. It is an independent data set to evaluate the model's performance and assess its ability to make correct investment decisions for that particular year.

Similarly, in Group 2, the training dataset travels from January 1, 2014, to December 31, 2020, providing seven years of historical data. This dataset is employed for training the model and fine-tuning its parameters. The test dataset for Group 2 consists of the year 2021, from January 1 to December 31. It is an unseen dataset for evaluating the model's predictions and measuring its performance in

capturing stock market movements and trends for that year. The training and test datasets for each stock market contain all the stocks we selected. This dataset-splitting strategy allows for assessing the model's performance across different time periods, ensuring a comprehensive evaluation of its predictive capabilities in varying market conditions [42].

V. EXPERIMENT SETTING

This section presents details about the model frameworks, hyperparameter optimization, backtesting, and performance measurement.

A. MODEL FRAMEWORKS

The experiments in this paper involve three model frameworks, LightGBM [22], XGBoost [23], and Scikit-learn [43]. LightGBM and XGBoost are gradient-boosting frameworks that utilize tree-based learning algorithms. Scikit-learn is a Python module that integrates various cutting-edge machine-learning algorithms for medium-scale supervised and unsupervised problems. Five models are constructed based on the above framework, namely LightGBM, CHL-LightGBM, FL-LightGBM [11], XGBoost, and Scikit-learn decision tree [44].

In this study, we considered the following factors when selecting the benchmark models:

1) To observe the difference in performance when transitioning from the decision tree algorithm to the gradient boosting decision tree algorithm, we selected the Decision Tree model and XGBoost model as the benchmarks. The XGBoost model possesses the same principle as LightGBM, and it is also possible to compare the performance difference between the two models in predicting stock returns.
2) To more accurately compare the effectiveness of the proposed model, we selected the original LightGBM and FL-LightGBM models as the benchmarks. By comparing the proposed model with the original LightGBM, we can thoroughly analyze the cost-balancing mechanism's action process and the proposed model's prediction effect. By comparing the proposed model with FL-LightGBM, we can observe the difference in prediction effect between example-dependent and class-dependent cost-sensitive learning.

B. HYPERPARAMETER OPTIMIZATION

To further optimize the model, we use the Optuna [45] framework to complete the steps of hyperparameter optimization. Optuna is a hyperparameter optimization framework that can automatically search for the best hyperparameters of a given machine learning model. Optuna uses a technique called Bayesian optimization, which sequentially selects hyperparameters based on their expected improvement on the objective function. Generally speaking, Optuna is efficient, flexible, easy to use, and parallelizable.

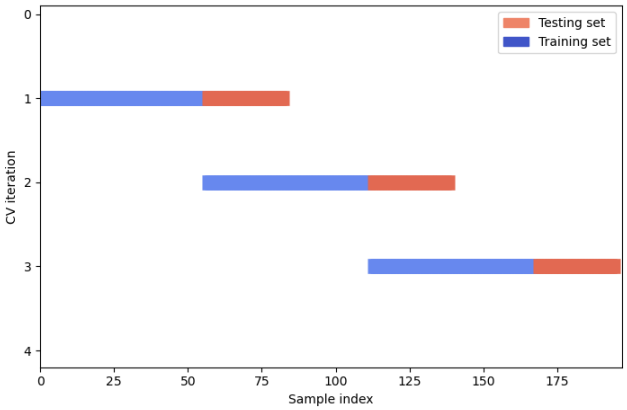

FIGURE 4. Blocking time series cross-validation for the entire training dataset.

TABLE 4. Hyperparameter search space of LightGBM.

Parameter	Data type	Search space
objective		binary
metric		auc
boosting_type		gbdt
num_boost_round		1000
learning_rate	float	0.01, 0.1
lambda_l1	float	1e-8, 10.0, log=True
lambda_l2	float	1e-8, 10.0, log=True
num_leaves	int	31, 256
feature_fraction	float	0.4, 1.0
bagging_fraction	float	0.4, 1.0
bagging_freq	int	1, 7
min_child_samples	int	20, 100

We optimize model hyperparameters based on blocking time series cross-validation [11] to improve model accuracy and generalization. During cross-validation, we split the entire training dataset into three blocks chronologically. The division rule is that each block contains a training set and a testing set, and from the second block, the start date of the training set of the block is the same as the start date of the testing set of the previous block, as shown in Fig. 4.

The hyperparameter search spaces of each model are defined as shown in Table 4, Table 5, Table 6, and Table 7. CHL-LightGBM is constructed based on parameter-tuned LightGBM and only needs to fine-tune its learning_rate.

C. BACKTESTING

Backtesting involves building models that simulate trading strategies using historical data aimed to evaluate the performance of the model [46]. In this research, the backtesting is based on the prediction results from the classifier; the stocks predicted to be profitable for the next day can participate in simulated trading. The specific steps of the backtesting are as follows.

We first determined the total investment amount and the investment ratio of one stock. Secondly, set the classification threshold. To improve the precision, the threshold is generally

TABLE 5. Hyperparameter search space of FL-LightGBM.

Parameter	Data type	Search space
objective		Focal_loss
metric		costloss
boosting_type		gbdt
num_boost_round		1000
alpha	float	0.4, 0.9
gamma	float	1.0, 3.0
learning_rate	float	0.01, 0.1
lambda_l1	float	1e-8, 10.0, log=True
lambda_l2	float	1e-8, 10.0, log=True
num_leaves	int	31, 256
feature_fraction	float	0.4, 1.0
bagging_fraction	float	0.4, 1.0
bagging_freq	int	1, 7
min_child_samples	int	20, 100

TABLE 6. Hyperparameter search space of XGBoost.

Parameter	Data type	Search space
objective		binary:logistic
metric		auc
booster		gbtree
tree_method		exact
n_estimators		1000
eta	float	0.01, 0.1
alpha	float	1e-8, 10.0, log=True
lambda	float	1e-8, 10.0, log=True
max_depth	int	3, 15
subsample	float	0.4, 1.0
colsample_bytree	float	0.4, 1.0
gamma	int	0, 5
min_child_weight	int	20, 100

selected in the range of 0.6 to 0.7. Stocks classified as profitable are sorted in descending order of their predicted probabilities. Finally, the selected stocks are simulated until the total investment is reduced to the limit or all the selected stocks are traded.

Backtesting follows a defined investment strategy. This study focuses on investing over a short period, aiming to generate investment income while maintaining asset liquidity. Drawing on the short-term investment methodology used in [20], our longest holding period is five trading days. Therefore, all inventory will be sold within this time frame.

D. PERFORMANCE MEASUREMENT

We discuss the measurement of the model in four different aspects: predictive accuracy, cost-harmonizing performance, profitability, and risk control ability.

1) METRICS OF PREDICTIVE ACCURACY

To measure the predictive accuracy of our model, we employed several metrics, including AUC, Accuracy, Precision, and F2.

AUC (Area Under Curve) is the area under the receiver operating characteristic curve and is an important measure of classifier performance. Its calculation formula is as follows.

$$\text{AUC} = \int_0^1 ROC \times \text{d}FPR$$

where FPR is the false positive rate.

Accuracy measures the proportion of correctly classified examples by the classifier, and its calculation formula is as follows.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

Precision measures the ability of the classifier to predict positive examples correctly, and its calculation formula is as follows.

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

F2 score is a weighted average of Precision and Recall, with a weight biased towards Recall, making it suitable for imbalanced datasets. Its calculation formula is as follows.

$$\text{F2} = \left(1 + \beta^2\right) \times \frac{\text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}}$$

where $\text{Recall} = TP/(TP + FN)$, β is typically set to 2 to emphasize the importance of Recall.

TABLE 7. Hyperparameter search space of decision trees.

Parameter	Data type	Search space
criterion		gini
metric		auc
splitter	category	best, random
max_depth	int	10, 100
min_samples_split	int	2, 10
min_samples_leaf	int	2, 10
max_leaf_nodes	int	30, 256

2) METRIC OF COST-HARMONIZING PERFORMANCE

To better capture the effectiveness of cost harmonization, we defined a cost-weight positive rate (CPR) as:

$$\text{CPR} = \frac{W_{TP}}{W_{FP} + W_{FN}}$$

where W_{TP} is the total cost weight from true positive examples, W_{FP} is the total cost weight from false positive examples, and W_{FN} is the total cost weight from false negative examples.

CPR can be used to evaluate the performance of a binary classification model by considering both the profits and costs associated with the classification decisions. The numerator represents the total profits earned from correctly identifying true positive examples. The denominator represents the total costs incurred from misclassifying false positive and false negative examples. A higher rate indicates better performance of the model, as it implies higher profits relative to the costs incurred.

TABLE 8. Classification results of different models.

	Stock Market	Shanghai		Hongkong		NASDAQ	
	Test Dataset (year)[a]	2020	2021	2020	2021	2020	2021
AUC	LightGBM	0.5768	**0.5555**	0.5454	0.5522	0.5401	0.5192
	CHL-LightGBM	**0.5798**	0.5553	**0.5496**	0.5539	0.5355	**0.5205**
	FL-LightGBM	0.5790	0.5555	0.5460	**0.5562**	0.5340	0.5199
	DT	0.5703	0.5368	0.5206	0.5112	0.4997	0.5061
	XGBoost	0.5738	0.5545	0.5428	0.5528	**0.5427**	0.5031
Accuracy	LightGBM	0.5774	0.5722	0.5541	0.5623	0.5344	0.5417
	CHL-LightGBM	0.5777	0.5721	**0.5562**	0.5637	0.5315	0.5393
	FL-LightGBM	**0.5826**	0.5726	0.5535	**0.5673**	0.5261	0.5383
	DT	0.5731	0.5722	0.5549	0.5440	0.5068	0.5545
	XGBoost	0.5720	**0.5736**	0.5544	0.5615	**0.5379**	**0.5472**
Precision	LightGBM	0.5372	0.5097	0.4890	0.5298	0.4998	0.4621
	CHL-LightGBM	0.5349	0.5085	**0.4943**	0.5277	0.4964	0.4643
	FL-LightGBM	0.5556	0.5107	0.4884	**0.5464**	0.4913	0.4564
	DT	**0.6855**	**0.6226**	0.4833	0.4221	0.4637	**0.4713**
	XGBoost	0.5372	0.5225	0.4794	0.5418	**0.5038**	0.4503
F2	LightGBM	0.5549	0.5368	0.5370	0.5292	0.5325	0.5196
	CHL-LightGBM	**0.5580**	**0.5399**	**0.5450**	**0.5383**	0.5296	**0.5247**
	FL-LightGBM	0.5562	0.5389	0.5384	0.5340	0.5262	0.5180
	DT	0.5077	0.5014	0.5172	0.4849	0.5034	0.4971
	XGBoost	0.5344	0.5269	0.5136	0.5157	**0.5374**	0.4602
CPR	LightGBM	0.3322	0.2039	0.3351	0.2390	0.5798	0.2549
	CHL-LightGBM	**0.3762**	**0.2447**	**0.4009**	**0.3169**	**0.6782**	**0.3345**
	FL-LightGBM	0.3201	0.2158	0.3486	0.2464	0.6551	0.2103
	DT	0.0748	0.0185	0.1874	0.0562	0.4786	0.0813
	XGBoost	0.2115	0.1276	0.1697	0.1488	0.6231	0.1205

[a] The training data-set corresponding to test data-set2020 is the stock data from 2013 to 2019; the training data-set corresponding to test data-set2021 is the stock data from 2014 to 2020

3) METRICS OF PROFITABILITY

To describe the profitability of a stock prediction model, we applied various metrics such as the rate of return (ROR), winning rate (WR), and annualized return (AR). These metrics can provide insights into the model's ability to generate profits.

Rate Of Return (ROR) measures the percentage change in the value of an investment over a specific period. It can be calculated using the following formula:

$$\mathrm{ROR} = \frac{V_f - V_i}{V_i} \times 100\%$$

where V_i is the initial value of the investment, V_f is the final value of the investment.

Winning Rate (WR) measures the percentage of successful trades from all trades made by the model. It can be calculated using the following formula:

$$\mathrm{WR} = \frac{N_{st}}{N_t} \times 100\%$$

where N_{st} is the number of successful trades, N_t is the total number of trades.

Annualized Return (AR) is the average rate of return per year over a specific period. It can be calculated using the following formula:

$$\mathrm{AR} = (1 + ROR)^{1/n} - 1$$

where ROR is the rate of return, n is the number years held.

4) METRICS OF RISK CONTROL ABILITY

Evaluating the risk-controlling ability of a stock prediction model is important because stock market prediction involves risks, and it is necessary to manage and control these risks to minimize losses and increase profits. A model with good risk-controlling ability can protect investors' capital and increase their returns in the long run. To describe the risk control ability of a stock market prediction model, we can use various indicators such as the Sharpe ratio (SR), the Sortino ratio (SOR), and annual volatility (AV).

Sharpe Ratio (SR) measures the excess return per unit of risk in an investment. It can be calculated using the following formula:

$$\mathrm{SR} = \frac{R_p - R_f}{\sigma_p}$$

where R_p is the portfolio's expected return, R_f is the risk-free rate, and σ_p is the portfolio's standard deviation.

Sortino Ratio (SOR) measures the risk-adjusted return of an investment. It takes into account only the downside risk.

It can be calculated using the following formula:

$$\text{SOR} = \frac{R_p - R_f}{\sigma_d}$$

where R_p is the portfolio's expected return, R_f is the risk-free rate, and σ_d is the downside deviation.

Annual Volatility (AV) measures the variability of a portfolio's returns over a year. It can be calculated using the following formula:

$$\text{AV} = \sigma_p \times \sqrt{N}$$

where σ_p is the portfolio's standard deviation, and N is the number of trading days in a year.

VI. EXPERIMENT RESULTS

The training datasets are selected from 2013 to 2019 and 2014 to 2020, respectively, and the prediction results for 2020 and 2021 are evaluated. The model's performance has been evaluated in four categories: predictive accuracy, cost-harmonization, profitability, and risk control ability.

A. PREDICTIVE ACCURACY

Table 8 presents the performance comparison of different models, including CHL-LightGBM, on predicting stock market trends. The evaluation metrics used in the analysis are AUC, Accuracy, Precision, and F2. The test dataset includes data from 2020 and 2021 for Shanghai, Hong Kong, and NASDAQ stock markets.

The results show that the CHL-LightGBM model outperformed the other models in terms of F2 scores. Specifically, CHL-LightGBM achieved an F2 score of 0.5580 and 0.5399 for Shanghai's 2020 and 2021 test datasets, respectively. Similarly, the F2 score of CHL-LightGBM for other stock markets, except for NASDAQ 2020, is also the highest among all models.

In terms of AUC, CHL-LightGBM outperforms other models overall. For Shanghai 2020, Hong Kong 2020, and NASDAQ 2021, CHL-LightGBM achieved the highest score. The rest is also comparable to other models.

Regarding accuracy and precision, CHL-LightGBM showed comparable performance with the other models. Except for the outstanding Precision of the Decision Trees, the scores of the other models are not significantly different. The results demonstrate that CHL-LightGBM is a competitive model for predicting stock market trends, with superior performance in AUC and F2 scores and comparable performance in other evaluation metrics. In addition, the performance of all models is relatively comparable, with AUC values ranging between 0.4997 and 0.5798. This finding implies that the models can reasonably distinguish between positive and negative examples, demonstrating their effectiveness in predicting stock market outcomes.

Comparing the predictive accuracy of the models, it can be seen that there are slight differences in their predictive accuracy across different years and stock markets. However, the overall differences in predictive accuracy are relatively small. In most cases, CHL-LightGBM has slightly better predictive accuracy than the other two models, especially in predicting the values of Shanghai and Hong Kong in 2020 and 2021. However, the differences in predictive accuracy between the models are not significant enough to draw firm conclusions. It is important to note that other factors, such as the quality and quantity of data used and the modeling techniques, can also impact predictive accuracy. Therefore, a more detailed analysis would be necessary to draw more conclusive comparisons between these models. It appears that CHL-LightGBM exhibits a marginal improvement in predictive accuracy compared to the other two models across most cases, particularly in predicting the values of stocks in the Shanghai and Hong Kong Stock Exchanges during 2020 and 2021. Nevertheless, the observed disparities in predictive accuracy between the models lack statistical significance and, thus, warrant caution in drawing definitive conclusions.

According to Table 8, CHL-LightGBM performs slightly better than the other two models, LightGBM and FL-LightGBM, in terms of AUC and F2 score for all three stock markets and across different years. Specifically, CHL-LightGBM shows slightly better prediction accuracy than other models in predicting the values of stocks in the Shanghai and Hong Kong Stock Exchanges during 2020 and 2021. However, the differences in accuracy between the models are not significant enough to draw firm conclusions. In comparison with Decision Trees (DT) and XGBoost, all three LightGBM models perform better in terms of AUC and F2 scores.

B. COST HARMONIZATION

The analysis of the CPR values is presented in Table 8 indicates that CHL-LightGBM outperforms the other models, with the highest CPR values for all columns. In addition, LightGBM, CHL-LightGBM, and FL-LightGBM show superior performance compared to DT and XGBoost.

C. PROFITABILITY

Table 9 presents a comparison of profitability among different models for three stock markets, Shanghai, Hong Kong, and NASDAQ, using test datasets from 2020 and 2021. The measures of profitability include the rate of return, annualized return, and winning ratio. Results indicate that CHL-LightGBM has the highest rate of return and annualized return compared to other models for all three markets, representing that it is the most profitable model. Moreover, CHL-LightGBM has a higher winning ratio than LightGBM and FL-LightGBM but lower than DT and XGBoost for all three markets.

Fig. 5, Fig. 6, and Fig. 7 provide the comparative analysis of the models CHL-LightGBM, FL-LightGBM, and LightGBM in different markets during 2020 and 2021. Indicates that CHL-LightGBM consistently displays outstanding performance in terms of ROR. This observation is particularly noticeable in the case of the 2020 Shanghai Stock Exchange and the 2021 NASDAQ market, where

TABLE 9. Profit results of different models.

	Stock Market	Shanghai		Hongkong		NASDAQ	
	Test Dataset (year)	2020	2021	2020	2021	2020	2021
Rate Of Return	LightGBM	0.5678	0.1258	3.0071	3.1909	5.3254	0.9660
	CHL-LightGBM	**0.6367**	**0.2104**	**3.7313**	**4.1944**	**6.6450**	**1.4733**
	FL-LightGBM	0.6174	0.1620	3.3526	3.9349	6.5295	1.0771
	DT	0.1918	0.0081	0.6964	0.0070	−0.1924	−0.0510
	XGBoost	0.3996	0.0310	2.0382	2.7061	4.9351	0.1901
	Benchmark[b]	0.1387	0.0421	0.0650	−0.0389	0.4364	0.2214
Annualized Return	LightGBM	0.5702	0.1277	3.0262	3.2968	5.3656	0.9855
	CHL-LightGBM	**0.6395**	**0.2138**	**3.7566**	**4.3456**	**6.6986**	**1.5062**
	FL-LightGBM	0.6201	0.1645	3.3746	4.0739	6.5819	1.0994
	DT	0.1925	0.0081	0.6995	0.0071	−0.1930	−0.0517
	XGBoost	0.4012	0.0315	2.0498	2.7916	4.9715	0.1931
Winning Rate	LightGBM	0.5584	0.5161	0.5850	**0.6064**	**0.5587**	0.5288
	CHL-LightGBM	0.5540	0.5171	0.5700	0.5791	0.5305	0.5315
	FL-LightGBM	0.5651	0.5172	0.5779	0.6053	0.5556	**0.5325**
	DT	**0.6505**	**0.5754**	0.5420	0.4655	0.4609	0.4769
	XGBoost	0.5652	0.5168	**0.6006**	0.5936	0.5582	0.5040

[b] ROR benchmark for the stock indexes.

TABLE 10. Risk metrics results of different models.

	Stock Market	Shanghai		Hongkong		NASDAQ	
	Test Dataset (year)	2020	2021	2020	2021	2020	2021
Sharpe Ratio	LightGBM	2.81	0.58	10.48	13.56	11.90	4.06
	CHL-LightGBM	**2.88**	**0.97**	**12.32**	16.42	10.44	**5.26**
	FL-LightGBM	3.16	0.75	11.21	**17.44**	**15.47**	4.53
	DT	2.10	−0.49	1.99	−0.20	−0.60	−0.58
	XGBoost	2.61	0.06	7.03	13.03	9.92	0.87
Sortino Ratio	LightGBM	75.83	15.10	243.00	529.74	334.62	109.08
	CHL-LightGBM	74.65	**23.82**	**299.36**	588.29	363.44	**139.62**
	FL-LightGBM	**89.05**	18.44	293.23	**623.73**	**431.49**	123.08
	DT	−12.55	−29.97	−17.42	−27.00	6.12	−16.33
	XGBoost	84.55	10.42	187.71	436.00	286.03	24.18
Annual Volatility	LightGBM	101.54	330.90	39.64	32.27	58.23	84.35
	CHL-LightGBM	101.56	**230.81**	**37.44**	30.56	72.66	**74.65**
	FL-LightGBM	**92.61**	278.81	39.00	**27.92**	**50.68**	78.30
	DT	111.70	−183.54	117.91	1453.53	−597.95	−688.75
	XGBoost	102.53	945.03	49.59	31.14	66.12	256.92
	Benchmark[c]	334.11	668.37	528.69	−1619.71	210.34	208.50

[c] Annual volatility benchmark for the stock indexes.

CHL-LightGBM displays the best performance as compared to the other models. Such findings provide compelling evidence of the efficacy of CHL in enhancing the predictive power of LightGBM for stock returns.

Although there is no significant difference between CHL-LightGBM and other models on the accuracy and winning rate, CHL-LightGBM obtained the highest annualized return on all the test data. This result further reveals that the proposed model selects trades that are not winning by quantity but are more focused on cost. In this study, we are not trying to optimize the winning rate (or the error rate). Instead, we optimize the return by reducing costs (avoiding significant loss). Compared with other models, the cost weight positive rate (*CPR*) of the proposed model is the highest, which reflects that in the prediction results, the true positive (*TP*) examples have a large proportion of high-cost weights, and the false positive (*FP*) examples have a small ratio of high-cost weights. In this case, obtaining a high annualized return is possible even if the number of *TP* is less than the number of *FP*.

D. RISK CONTROL ABILITY

The assessment of a model's risk control ability involves the evaluation of various risk measures, including the Sharpe ratio, Sortino ratio, and annual volatility. A model with lower annual volatility is considered to have better risk control ability, while a higher Sharpe and Sortino ratio indicates better risk-adjusted returns.

Based on Table 10, the LightGBM, CHL-LightGBM, and FL-LightGBM models demonstrate better risk control ability

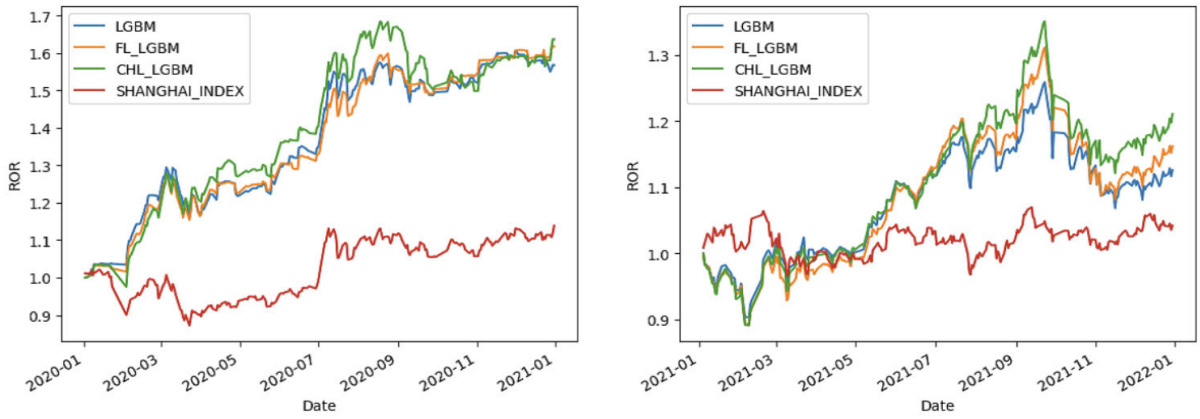

FIGURE 5. ROR comparison of different models of the Shanghai stock market. SHANGHAI_INDEX is the ROR of the Shanghai Index, which is used as the benchmark to compare the performance of the model with the market as a whole.

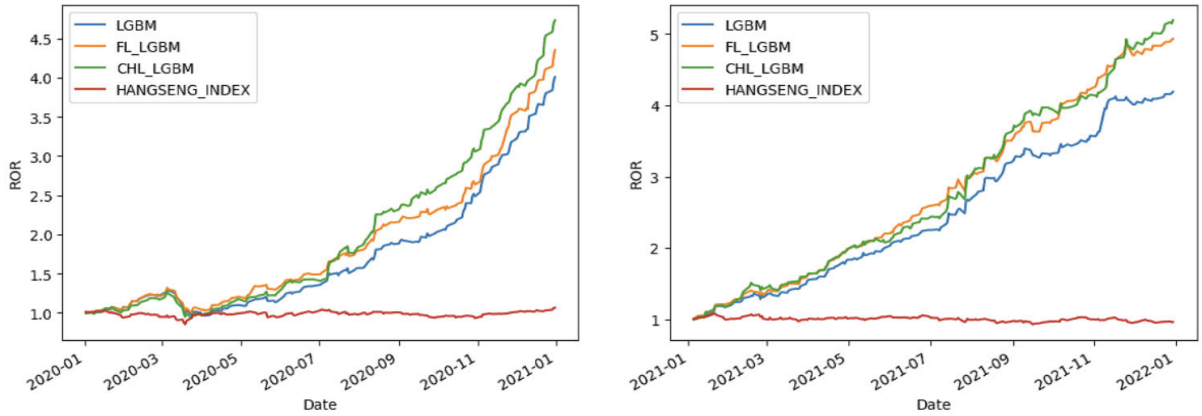

FIGURE 6. ROR comparison of different models of the Hong Kong stock market. HANGSENG_INDEX is the ROR of the Hang Seng Index, which is used as the benchmark to compare the performance of the model with the market as a whole.

than the DT and XGBoost models, as they have lower annual volatility. Regarding risk-adjusted returns, the Sharpe and Sortino ratios are utilized. The Sharpe ratio is higher when the return is higher relative to the risk-free rate and the volatility of returns, while the Sortino ratio considers only the downside volatility (volatility of negative returns). All models demonstrate positive Sharpe ratios, indicating positive risk-adjusted returns. Among the three models, the FL-LightGBM model exhibits the highest Sharpe ratio in the Shanghai and Hong Kong stock markets for both test years, showing a better risk-adjusted return. However, in the NASDAQ stock market, the LightGBM model exhibited the highest Sharpe ratio in 2020, while the CHL-LightGBM model exhibited the highest Sharpe ratio in 2021. In contrast, the Sortino ratio is higher for all models compared to the Sharpe ratio. Among the three models, the CHL-LightGBM model exhibits the highest Sortino ratio for all three stock markets and both test years, indicating better risk-adjusted returns.

In summary, the CHL-LightGBM model demonstrates good risk control ability, as evidenced by its low annual volatility. It also shows good risk-adjusted returns with high Sharpe and Sortino ratios. On the other hand, the FL-LightGBM model exhibits the best risk control ability in terms of a good Sharpe ratio for the Shanghai and Hong Kong stock markets. Therefore, the selection of a model eventually depends on the investor's specific risk tolerance and investment objectives.

The CHL-LightGBM outperforms all other models in terms of both profit and risk metrics in different periods across the three stock markets. Still, the experiment results show significant differences in the performance of the proposed model in different stock markets. From the analysis of the transverse experiment results of three stock markets, for the proposed model, the Shanghai stock market has the highest annual volatility (AV) and the lowest annualized return (AR); The Hong Kong stock market has the least AV and the most stable AR. Its AR is higher than that of Shanghai but lower than the highest value of the NASDAQ; The NASDAQ stock market has a medium AV and the highest AR of the 2020 test set. It can be observed from the aforementioned analysis that the proposed model demonstrates a remarkable level of stability and a high rate of return in the mature stock market.

Through the above analysis, it can be shown that the profitability of CHL-LightGBM is related to the volatility of the stock market, which imposes certain limitations

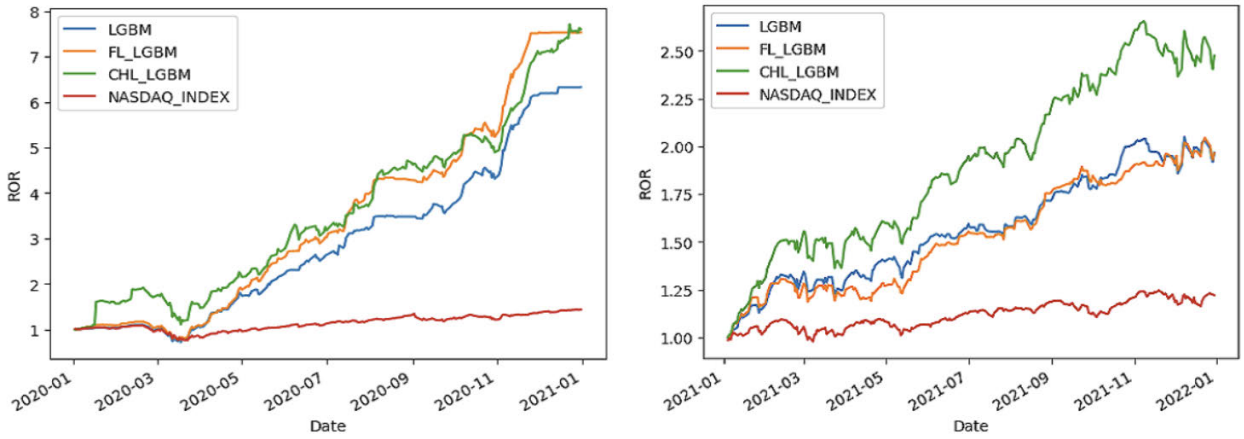

FIGURE 7. ROR comparison of different models of the NASDAQ stock market. NASDAQ_INDEX is the ROR of the NASDAQ Index, which is used as the benchmark to compare the performance of the model with the market as a whole.

on the performance of the proposed model. One possible explanation could be that the model's cost factor primarily depends on the cost matrix, which is calculated from the stock's price changes. Suppose the volatility of the stock market is excessively high. In that case, the cost matrix of the training set cannot be effectively adapted to the test set, which consequently diminishes the predictive efficacy of the model.

VII. CONCLUSION

Based on CSL, a new CHL was proposed to combine the cost of misclassifying examples with their hardness of classification for stock predictions. By embedding CHL into LightGBM, the derivatives of CHL could adaptively balance the cost and the hardness among the examples in the training process. The effectiveness of CHL-LightGBM has been validated for investment prediction in the stock market across three stages of development and different periods. experiment results demonstrate that CHL-LightGBM outperformed the original LightGBM model in terms of profitability, risk control, and predictive accuracy. CHL-LightGBM also performed much better than LightGBM, XGBoost, and decision trees for predicting the three stock markets.

CHL-LightGBM suits cost-sensitive classification applications, particularly in large-scale and structured data domains. CHL-LightGBM has practical applications in anomaly detection (e.g., malware detection and intrusion detection for network security), defect detection (e.g., industrial product inspection), medical diagnosis, credit card fraud detection, etc. Meanwhile, the proposed cost harmonization loss function would also be tested in training other deep-learning models.

Therefore, this study provides a new method to address the problem of imbalanced difficulty levels of examples and achieves good results in predicting the stock market, which has certain reference significance and value for research and practice in relevant fields.

In our future research, we will delve into the interpretability of LightGBM by techniques such as SHAP values [47] and individual conditional expectation (ICE) diagrams. These techniques can aid in revealing the correlation between features and predictions, thereby unveiling the impact of individual variables on the model's output.

REFERENCES

[1] A. Thakkar and K. Chaudhari, "A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions," *Expert Syst. Appl.*, vol. 177, Sep. 2021, Art. no. 114800.

[2] J. Zou, Q. Zhao, Y. Jiao, H. Cao, Y. Liu, Q. Yan, E. Abbasnejad, L. Liu, and J. Q. Shi, "Stock market prediction via deep learning techniques: A survey," 2022, *arXiv:2212.12717*.

[3] A. Rostamian and J. G. O'Hara, "Event prediction within directional change framework using a CNN-LSTM model," *Neural Comput. Appl.*, vol. 34, no. 20, pp. 17193–17205, Oct. 2022.

[4] P. S. Sisodia, A. Gupta, Y. Kumar, and G. K. Ameta, "Stock market analysis and prediction for Nifty50 using LSTM deep learning approach," in *Proc. 2nd Int. Conf. Innov. Practices Technol. Manage. (ICIPTM)*, vol. 2, Feb. 2022, pp. 156–161.

[5] P. R. Singh, N. Manohar, and R. Mahesh, "Stock price prediction system with improved LSTM," in *Proc. IEEE North Karnataka Subsection Flagship Int. Conf. (NKCon)*, Nov. 2022, pp. 1–6.

[6] D. Cheng, F. Yang, S. Xiang, and J. Liu, "Financial time series forecasting with multi-modality graph neural network," *Pattern Recognit.*, vol. 121, Jan. 2022, Art. no. 108218.

[7] K. Huang, X. Li, F. Liu, X. Yang, and W. Yu, "ML-GAT: A multilevel graph attention model for stock prediction," *IEEE Access*, vol. 10, pp. 86408–86422, 2022.

[8] H. Wang, S. Li, T. Wang, and J. Zheng, "Hierarchical adaptive temporal-relational modeling for stock trend prediction," in *Proc. 13th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 3691–3698.

[9] Y. Li, Z. Simon, and D. Turkington, "Investable and interpretable machine learning for equities," *SSRN Electron. J.*, Jan. 2020.

[10] Y. Sahin, S. Bulkan, and E. Duman, "A cost-sensitive decision tree approach for fraud detection," *Expert Syst. Appl.*, vol. 40, no. 15, pp. 5916–5923, Nov. 2013.

[11] X. Zhao and Y. Liu, "False awareness stock market prediction by LightGBM with focal loss," in *Proc. Conf. Res. Adapt. Convergent Syst.*, Oct. 2022, pp. 147–152.

[12] D. Almhaithawi, A. Jafar, and M. Aljnidi, "Example-dependent cost-sensitive credit cards fraud detection using SMOTE and Bayes minimum risk," *Social Netw. Appl. Sci.*, vol. 2, no. 9, pp. 1–12, Sep. 2020.

[13] C. Elkan, "The foundations of cost-sensitive learning," in *Proc. Int. Joint Conf. Artif.*, vol. 17, no. 1, 2001, pp. 973–978.

[14] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification Scene Analysis*, vol. 3. New York, NY, USA: Wiley, 1973.

[15] F. Provost and T. Fawcett, "Robust classification for imprecise environments," *Mach. Learn.*, vol. 42, no. 3, pp. 203–231, 2001.

[16] G. Fumera et al., "Cost-sensitive learning in support vector machines," in *Proc. VIII Convegno Associazione Italiana per L'Intelligenza Artificiale*, 2002.
[17] M. Tang, Q. Zhao, H. Wu, and Z. Wang, "Cost-sensitive LightGBM-based online fault detection method for wind turbine gearboxes," *Frontiers Energy Res.*, vol. 9, Aug. 2021, Art. no. 701574.
[18] W. Liu, H. Fan, M. Xia, and M. Xia, "A focal-aware cost-sensitive boosted tree for imbalanced credit scoring," *Expert Syst. Appl.*, vol. 208, Dec. 2022, Art. no. 118158.
[19] K. Zheng, G. Jia, L. Yang, and C. Liu, "A cost-sensitive diagnosis method based on the operation and maintenance data of UAV," *Appl. Sci.*, vol. 11, no. 23, Nov. 2021, Art. no. 11116.
[20] X. Zhao and Q. Zhao, "Stock prediction using optimized LightGBM based on cost awareness," in *Proc. 5th IEEE Int. Conf. Cybern. (CYBCONF)*, Jun. 2021, pp. 107–113.
[21] G. Shaoyun, L. Jifeng, L. Hong, H. Xingtang, S. Kaiyue, and Z. Yue, "Distributed day-ahead peer-to-peer trade for multimicrogrid integration of smart buildings in active distribution networks," in *Proc. IEEE Power Energy Soc. Gen. Meeting (PESGM)*, Aug. 2020, pp. 1–5.
[22] D. Wang, Y. Zhang, and Y. Zhao, "LightGBM: An effective miRNA classification method in breast cancer patients," in *Proc. Int. Conf. Comput. Biol. Bioinf.*, Oct. 2017, pp. 7–11.
[23] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.
[24] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001, doi: 10.1214/aos/1013203451.
[25] J. Yan, Y. Xu, Q. Cheng, S. Jiang, Q. Wang, Y. Xiao, C. Ma, J. Yan, and X. Wang, "LightGBM: Accelerated genomically designed crop breeding through ensemble learning," *Genome Biol.*, vol. 22, pp. 1–24, Dec. 2021.
[26] K. Li, H. Xu, and X. Liu, "Analysis and visualization of accidents severity based on LightGBM-TPE," *Chaos, Solitons Fractals*, vol. 157, Apr. 2022, Art. no. 111987.
[27] B. Shaker, M.-S. Yu, J. S. Song, S. Ahn, J. Y. Ryu, K.-S. Oh, and D. Na, "LightBBB: Computational prediction model of blood–brain-barrier penetration based on LightGBM," *Bioinformatics*, vol. 37, no. 8, pp. 1135–1139, May 2021.
[28] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Ann. Statist.*, vol. 28, no. 2, pp. 337–407, 2000.
[29] Y. Zelenkov, "Example-dependent cost-sensitive adaptive boosting," *Expert Syst. Appl.*, vol. 135, pp. 71–82, Nov. 2019.
[30] K. Li, B. Wang, Y. Tian, and Z. Qi, "Fast and accurate road crack detection based on adaptive cost-sensitive loss function," *IEEE Trans. Cybern.*, vol. 53, no. 2, pp. 1051–1062, Feb. 2023.
[31] B. Li, Y. Liu, and X. Wang, "Gradient harmonized single-stage detector," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 8577–8584.
[32] S. Mehtab and J. Sen, "Analysis and forecasting of financial time series using CNN and LSTM-based deep learning models," in *Advances in Distributed Computing and Machine Learning*. Cham, Switzerland: Springer, 2022, pp. 405–423.
[33] A. U. Haq, A. Zeb, Z. Lei, and D. Zhang, "Forecasting daily stock trend using multi-filter feature selection and deep learning," *Expert Syst. Appl.*, vol. 168, Apr. 2021, Art. no. 114444.
[34] X. Zhang, Y. Hu, K. Xie, S. Wang, E. W. T. Ngai, and M. Liu, "A causal feature selection algorithm for stock prediction modeling," *Neurocomputing*, vol. 142, pp. 48–59, Oct. 2014.
[35] M. R. Vargas, C. E. M. dos Anjos, G. L. G. Bichara, and A. G. Evsukoff, "Deep leaming for stock market prediction using technical indicators and financial news articles," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.
[36] Y. Zhai, A. Hsu, and S. K. Halgamuge, "Combining news and technical indicators in daily stock price trends prediction," in *Proc. Int. Symp. Neural Netw.* Nanjing, China: Springer, Jun. 2007, pp. 1087–1096.
[37] E. Acuna and C. Rodriguez, "The treatment of missing values and its effect on classifier accuracy," in *Classification, Clustering, and Data Mining Applications*. Chicago, IL, USA: Springer, Jul. 2004, pp. 639–647.
[38] C. K. Prahalad and V. Ramaswamy, *The Future Competition: Co-Creating Unique Value With Customers*. Brighton, MA, USA: Harvard Business Press, 2004.
[39] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," in *Data Classification: Algorithms and Applications*, Jan. 2014, pp. 37–64.
[40] M. B. Kursa, A. Jankowski, and W. R. Rudnicki, "Boruta—A system for feature selection," *Fundamenta Informaticae*, vol. 101, no. 4, pp. 271–285, 2010.
[41] Y. Liu, Y. Mu, K. Chen, Y. Li, and J. Guo, "Daily activity feature selection in smart homes based on Pearson correlation coefficient," *Neural Process. Lett.*, vol. 51, no. 2, pp. 1771–1787, Apr. 2020.
[42] I. E. Tampu, A. Eklund, and N. Haj-Hosseini, "Inflation of test accuracy due to data leakage in deep learning-based classification of OCT images," *Sci. Data*, vol. 9, no. 1, p. 580, Sep. 2022.
[43] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12 no. 10, pp. 2825–2830, 2012.
[44] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: Experiences from the scikit-learn project," in *Proc. ECML/PKDD Workshop, Lang. Data Mining Mach. Learn.*, 2013, pp. 108–122.
[45] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 2623–2631.
[46] K. Olorunnimbe and H. Viktor, "Deep learning in the stock market—A systematic survey of practice, backtesting, and applications," *Artif. Intell. Rev.*, vol. 56, no. 3, pp. 2057–2109, Mar. 2023.
[47] L. Li, J. Qiao, G. Yu, L. Wang, H.-Y. Li, C. Liao, and Z. Zhu, "Interpretable tree-based ensemble model for predicting beach water quality," *Water Res.*, vol. 211, Mar. 2022, Art. no. 118078.

XIAOSONG ZHAO (Graduate Student Member, IEEE) received the master's degree, in 2021. He is currently pursuing the Ph.D. degree with the Graduate School of Computer Science and Engineering, The University of Aizu, Japan. His current research interests include machine learning and financial information engineering.

YONG LIU was a Lecturer with the State Key Laboratory of Software Engineering, Wuhan University, China, in 1994, and a Research Fellow with the AIST Tsukuba Central 2, National Institute of Advanced Industrial Science and Technology, Japan, in 1999. He was a Guest Professor with the School of Computer Science, China University of Geosciences, China, in 2010. He is currently a Professor with The University of Aizu, Japan. His research interests include evolutionary computation and neural networks.

QIANGFU ZHAO (Senior Member, IEEE) received the Ph.D. degree from Tohoku University, Japan, in 1988. He joined the Department of Electronic Engineering, Beijing Institute of Technology, China, in 1988, first as a Postdoctoral Fellow and then as an Associate Professor. Since October 1993, he has been an Associate Professor with the Department of Electronic Engineering, Tohoku University. In April 1995, he joined The University of Aizu, Japan, as an Associate Professor, where he became a tenure Full Professor, in April 1999. His research interests include image processing, pattern recognition, machine learning, and awareness computing.