

# Penalizing LSTM Output Size for Short-Term Stock Market Analysis

Franklin Doane

*dept. of Computer Science*

*Tennessee Technological University*

*Cookeville, United States*

*doanefranklin89@gmail.edu*

**Abstract**—Time series based regressors are a common tool for stock market prediction; however, they can fail to take into account the risk increase that comes with overestimation as opposed to underestimation. This paper proposes the use of a logit size penalty loss term that can quantify this risk imbalance and increase precision of trades. The experiments show that using such penalties for an LSTM yielded a precision of !!!19.2% and average annual return of !!!90.8% when employed with a simple trading algorithm, compared to a precision of !!!10.2% and average return of !!!47.7% without the penalty. !!!A cross-dataset evaluation also demonstrated this to be similarly true over different time periods. These results suggest that penalizing predictor logits can force them to become more conservative in their price estimates, and this correlates to higher profits.

**Index Terms**—

## I. INTRODUCTION

The stock market is one of the greatest challenges for a data scientist. No where else in society does there exist such a straightforward translation from information to capital. And yet, anyone who wishes to harness information for trading must race against the speed at which the market integrates it. With the introduction of machine learning algorithms, it becomes easier to find signals that may be hiding in the data, and that may be missed by the average trader. However, there is inherent danger with every trade, no matter how strong the signal.

Trading stocks has a risk imbalance that favors harsh discrimination over careless investment. It can be thought of as a classification problem, where the goal is to classify "good" and "bad" investments at every timestep. Even when regression models are used, they are made with the purpose of combining with an algorithm that classifies any given timestep in these terms. When considering trading as a classification problem, it is worth noting that it is riskier to lose precision than it is to lose negative predictive value or even overall accuracy. Missing 100 possible opportunities is worth it for the chance to be certain that a price will go up just 1 time. This need to be correct when identifying positives more than when identifying negatives is something that would be given consideration with any classification model. However, in cases such as this where a regression model is used for a classification task, this need could be completely unquantified.

This work aims to be able to quantify that need by creating an additional penalties for over-estimating price. If a price pre-

diction model becomes more conservative with it's predictions, any trading algorithm based on its outcomes should become more precise. This paper will use LSTMs to explore a method by which to produce that result, and examine its effect on profitability.

## Contributions

- 1) This paper proposes a logit size penalty loss term to control model predictions.
- 2) This paper will demonstrate adding this term will increase trading precision with an ablation study.
- 3) This paper will analyze how this term will affect profitability in a practical application.

## II. RELATED WORK

### A. LSTMs for Market Prediction

Mei Sun, Qingtau Li, and Peiguang Lang, published work that was focused on using LSTM architecture for market price prediction [1]. Much like this work, they were aiming to use LSTM architecture for short-term price prediction; however, their focus was heavily on featuring engineering through the use of singular value composition (SVD). Qi Li, Norshaliza Kamaruddin, Siti Sophiayati Yuhani and Hamdan Amer Ali Al-Jaifi also have done work in this area [2]. Their work was on the use of LSTMs for price prediction. They aimed to augment it with the use of Symbolic Genetic Programming (SGP). While both of these use combine different methods with LSTM architecture to predict market prices, this work is distinct in its use of penalties for model logits.

### B. Penalizing Models for Market Prediction

Huifeng Jiang, Xuemei Hu, and Hong Jia explored different methods of penalizing logistic regressors in order to increase their performance as market indicators [3]. The authors explore penalties that apply to all model coefficients. In this work, the penalties being explored will apply only to model logits.

## III. SELECTING A PENALTY COEFFICIENT

### A. General Methods

**Dataset creation:** The all datasets were created by compiling data consisting of daily OHLCV values for each stock being used. The list of stocks being used was determined by picking the 1500 stocks with the largest market cap that also have data for the time periods that the datasets intend to cover.

*Training setup:* The dataset is loaded and the day of the week is 1-hot-encoded into 5 separate features that are added. The data then gets cut into time series, each with a length of 50. The data from the day following each time series is used to create the label, which consist of the relative increase in high price and low price compared to the close price of the previous day. Each training example gets all of its price data normalized together on a scale of -1 to 1. Similarly, the volume data gets normalized on a scale of -1 to 1. 500 batches are sliced off of the back of the dataset to become the validation set. The ordering of the training set gets shuffled according to a random seed. The training data gets split into whole batches of size 200. The model is initialized with random parameters. The model architecture is depicted in “Fig. 1”.

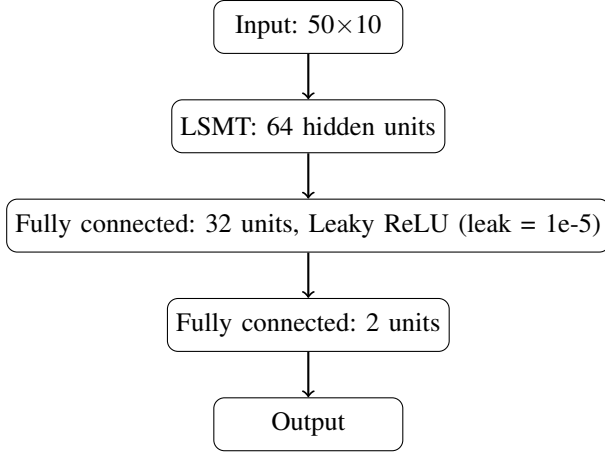


Fig. 1. Model architecture.

*Training methods:* The loss for each batch is calculated with formula “(1)”. The formula utilizes Mean Squared Error loss and a magnitude penalty for the logits.

$$Loss = mse(Y, \hat{Y}) + \lambda \sum_{i=1}^B (\hat{Y}_{i,1}) + \lambda \sum_{i=1}^B (\hat{Y}_{i,2}) \quad (1)$$

Here the logit at index 0 is the high price that the model is predicting, and at index 1 is the low price.  $B$  represents the batch size, and  $\lambda$  is the size penalty coefficient. The loss is backpropagated for each batch, and the parameters are updated with the Adam optimizer. The learning rate for the optimizer starts at  $1e-3$ , and is set by PyTorch’s LowerOnPlateau scheduler. The metric tracked by the scheduler is validation loss. At the end of each epoch, validation occurs. The model predicts all of the examples in the validation set and loss is calculated. Unlike during training, the loss in validation is calculated as a pure Mean Squared Error loss with no penalties. The loss is then used to update the learning rate scheduler. All models are trained for 30 epochs.

*Testing:* The testing dataset undergoes feature calculation and normalization in the same manner as the training set. An agent is created with a trading balance of \$100. The agent employs a simple trading algorithm that aims to only use strong positive signals. If the model predicts a high price

increase  $\geq 5\%$  and a low price increase  $> -0.1\%$ , the agent purchases the stock at the close price for that day. The agent’s balance is evenly split between all of the stocks chosen by the model, as long as each getting a minimum of \$1. For each position that the agent opened, it closes the position at a price from the following day. The agent is attempting to sell the position a  $\geq 5\%$  increase, but will stop-loss if the price is  $\leq -0.1\%$ . The outcome of the position is determined using the logic detailed in “Fig. 2”. The price that each position would’ve sold for is used to update the agent balance. Each position is checked first to see if the trading parameters would have caused it to sell at the open price. Then it is checked to see if it would sell during the day. Note the outcome where an a position value of both  $\leq -0.1\%$  and  $\geq 5\%$  occur during a day. Since it is ambiguous which would have happened first when using daily candlestick data, it is considered a stop-loss. Each position that is sold at or above the aim of a 5% increase

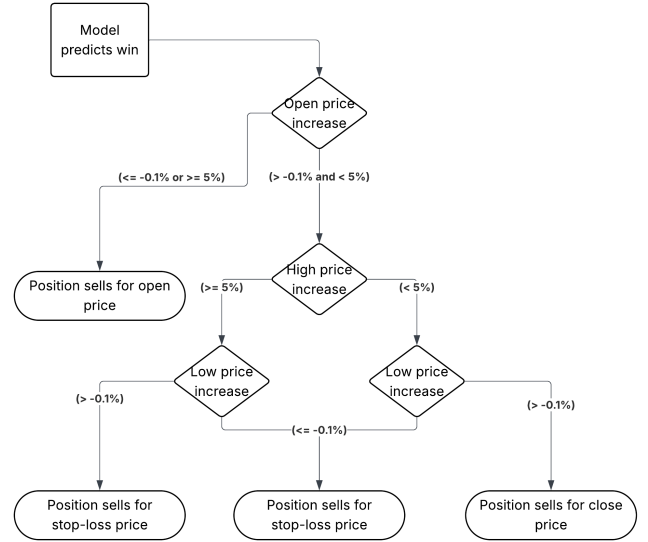


Fig. 2. Algorithm for determining price at which a position sells. The algorithm first checks to see if the position would sell at open. After this it examines mid-day prices to see if aim price or stop-loss price were hit. Note that if both of those occur, the outcome is ambiguous without more data and is given the stop-loss price in the interest of being conservative. If none of these occur, the stock sells at the close price.

is counted as a successful trade and a true positive from the agents perspective. Each position that is sold for less than or equal to  $-0.01\%$  of the purchase price, or that sells at the end of the day due to having no other opportunities to sell, is considered a false positive. After process is repeated for every timestamp, the total balance increase is calculated for the whole time span, along with the rates at which each of the outcomes occur. These rates are used to calculate classification metrics like precision.

### B. Coefficient Study

Using the general methods, 6 models were created. All were trained on data from 2013-2014, and testing on data

from 2016. Each was identical in every aspect of training and testing, except for the size penalty coefficient. The first model was given a coefficient  $\lambda = 2.5 \times 10^{-6}$ . The values for the other models increased logarithmically, ending with the largest model where  $\lambda = 7.5 \times 10^{-4}$ . The model with the value  $\lambda = 7.5 \times 10^{-6}$  performed best in testing in terms of both precision and return. 3 additional models were then created to sample more values in light of this, but it was found that  $7.5 \times 10^{-6}$  still was the best performing. Unless stated otherwise, all models in the remainder of this paper that are penalized will use this value as the penalty coefficient.

#### IV. SIZE PENALTY ABLATION TEST

##### A. Test Methods

With the size penalty coefficient  $\lambda$  selected, the model was then tested to see how it performed against an ablated model. Using the general methods, 8 models were trained with the size penalty coefficient  $\lambda = 7.5 \times 10^{-6}$ . Each one was trained on market data from all 1500 stocks from 2013 through 2014. The collection of these models will be referred to as Group A. For each of these models, a corresponding model was trained with a size penalty coefficient  $\lambda = 0$ . This is done in order to remove the term's effect on the loss entirely. Each corresponding model used the same random seed for the training data shuffle, so as to have complete parity in every part of the training process except the size penalty. This collection of models will be referred to as Group B.

##### B. Outcomes

After testing all of the models on data from the year 2016, it was found that 5 of the 8 penalized models from Group A had a higher return than their Group B counterparts. The distributions of the total return at the end of the year for both model groups can be seen in “Fig. 3”.

As indicated in “Fig. 3”, both the median and mean return were higher in Group A compared to Group B. The 2 groups had other performance differences. As intended, penalizing higher predictions lowered the positive rates for the agents using Group A models. “Fig. 4” illustrates this.

The higher selectivity exhibited in Group A is also correlated with a higher precision as demonstrated in “Fig. 5”.

These results indicate that the logit size penalty is improving model performance; however, all of these results summarize the temporal dimension rather than exploring it. It is worth examining, as seen in “Fig. 6”, how the performance changes over time.

This reveals that at many points during the evaluation period, the Group B models were performing better on average. Near the end of the year, the Group A models jump in performance may indicate that something about the market during the time period was favorable to the more selective models.

#### V. CROSS-DATASET EVALUATION

##### A. Methods

The cross-dataset evaluation was conducted using all of the methods outlined previously in Section III-A. The goal was to

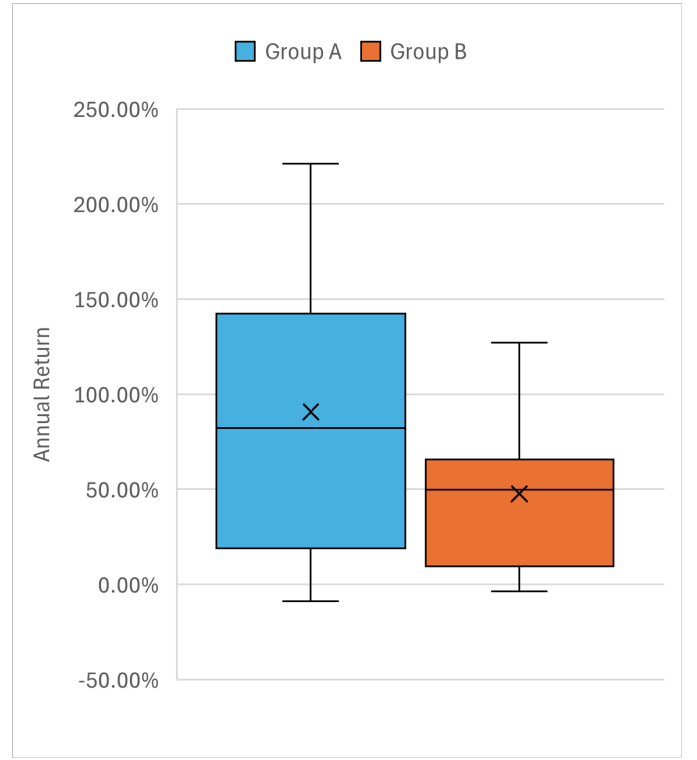


Fig. 3. Distributions of annual return from the year 2016 are shown for agents using penalized and un-penalized models in their evaluations on the data from the year 2016. Mean values are indicated with an X.

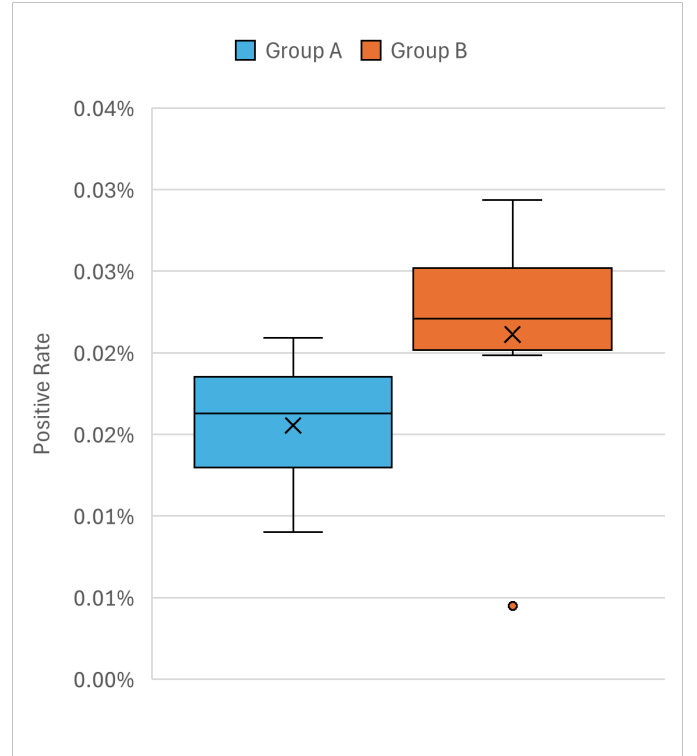


Fig. 4. Distributions of positive rates are shown for agents using penalized and un-penalized models in their evaluations on the data from the year 2016. Mean values are indicated with an X.

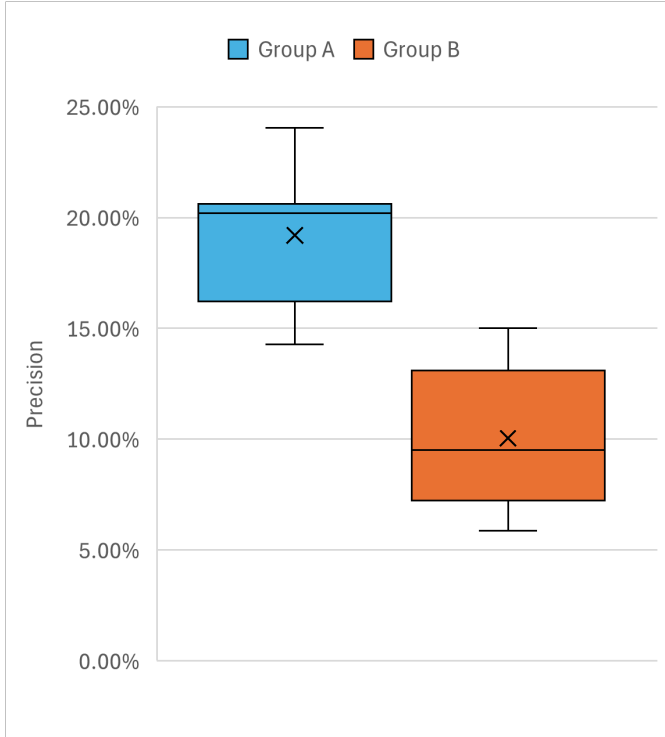


Fig. 5. Distributions of precision are shown for agents using penalized and un-penalized models in their evaluations on the data from the year 2016. Mean values are indicated with an X.

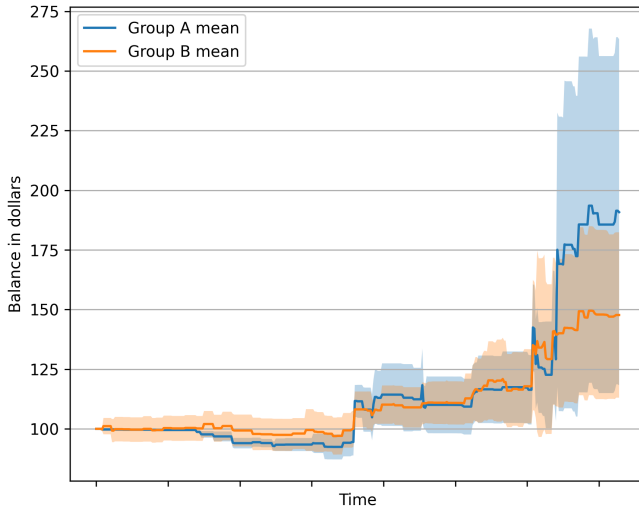


Fig. 6. The mean balances of the agents using Group A and Group B models are shown over the course of the evaluation year, 2016. The shaded area around each line represents the 95% confidence interval for the balance at that point.

conduct the same type of ablation test as in Section IV-A, but to test on multiple datasets that span multiple time periods. For every consecutive 2 year period from 2013-2014 through 2020-2021, a new model was trained on that data using the size penalty as described in previous sections. The collection of these models will be referenced as Group C. For each of these models, an ablated counterpart was created with identical training, except that the size penalty coefficient was set  $\lambda = 0$ . These models will be collectively referred to as Group D. All of the models in Group C and Group D had the random elements of their training seeded with the singular value. This value was chosen to be the seed used for training for the upper median models in both Group A and Group B in terms of annual return and precision. The models were tested on the 2nd year after the end of their training data, so as to avoid data leakage.

### B. Outcomes

Similar to the results in Section IV-B, both the median and mean annual return were higher for the agents who used who were trained with penalties. This is shown in “Fig .7”. Also similar to the previous results, the size-penalized models in Group C have a lower positive rate and a higher precision on average. “Fig .8” and “Fig .9” demonstrate this.

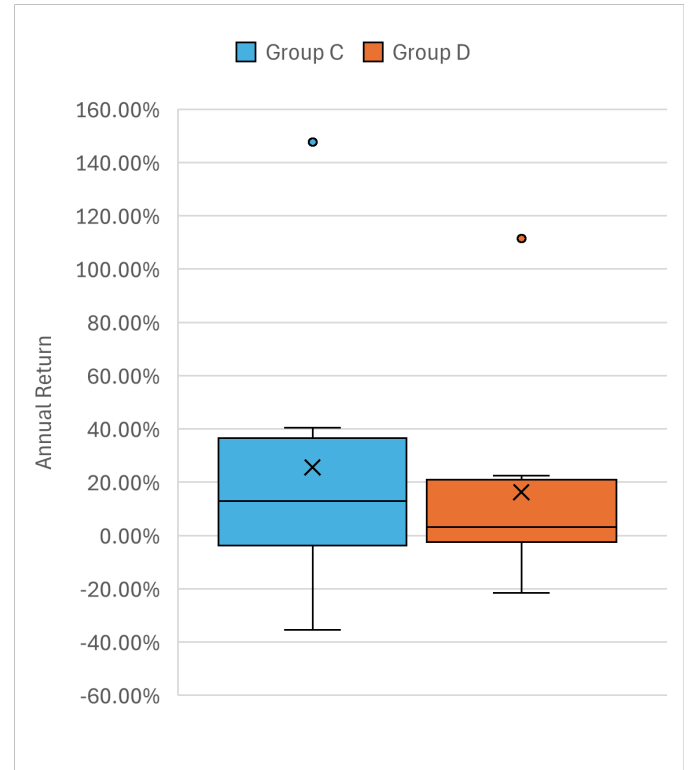


Fig. 7. Distributions of annual returns are shown for agents using the penalized and un-penalized models. Results are from evaluations ranging from 2016 through 2023. Mean values are indicated with an X.

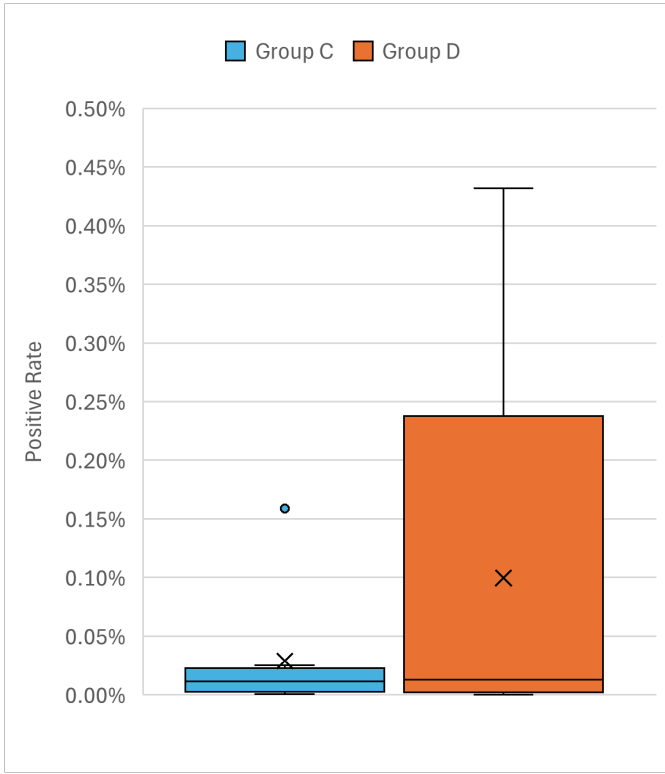


Fig. 8. Distributions of positive rates are shown for agents using penalized and un-penalized models. Results are from evaluations ranging from 2016 through 2023. Mean values are indicated with an X.

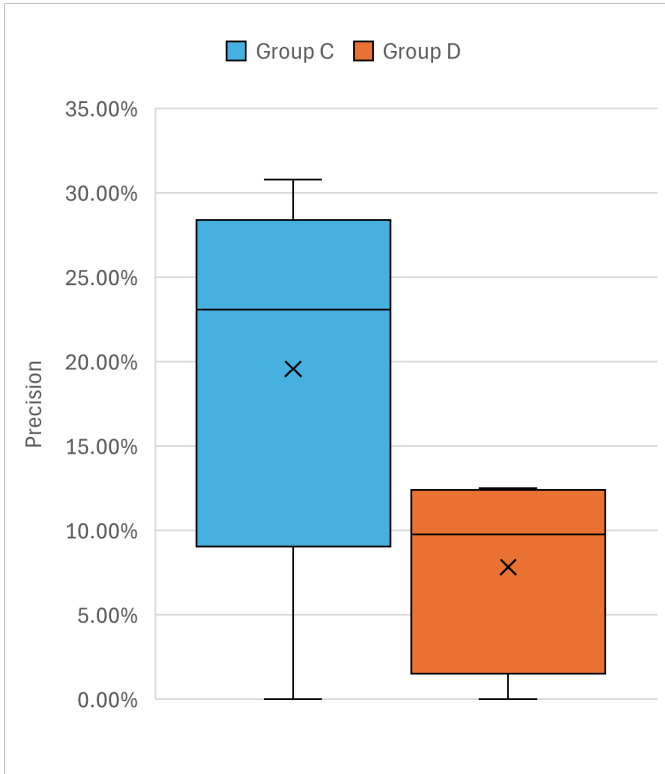


Fig. 9. Distributions of precision are shown for agents using penalized and un-penalized models. Results are from evaluations ranging from 2016 through 2023. Mean values are indicated with an X.

## VI. RESULTS

The results from both tests indicate a performative advantage with using the size penalty loss terms during training. !!!The mean annual return from using the models in Group A is 90.8% as opposed to the 47.7% from Group B. !!!The values for both of these are larger and farther apart than the means for Groups C and D, which are 25.6% and 16.3% respectively. !!!The mean precision from trading with the models in Group A was 19.2% on average, as opposed to the 10.1% from Group B. !!!A similar pattern was seen in the cross-dataset evaluation, where Group C and D produced precisions of 19.6% and 7.8% respectively. !!!In both tests, the agents using size penalized models were seen to have a higher increase on average for each true positive. These findings are illustrated in “Fig. 10” and “Fig. 11”.

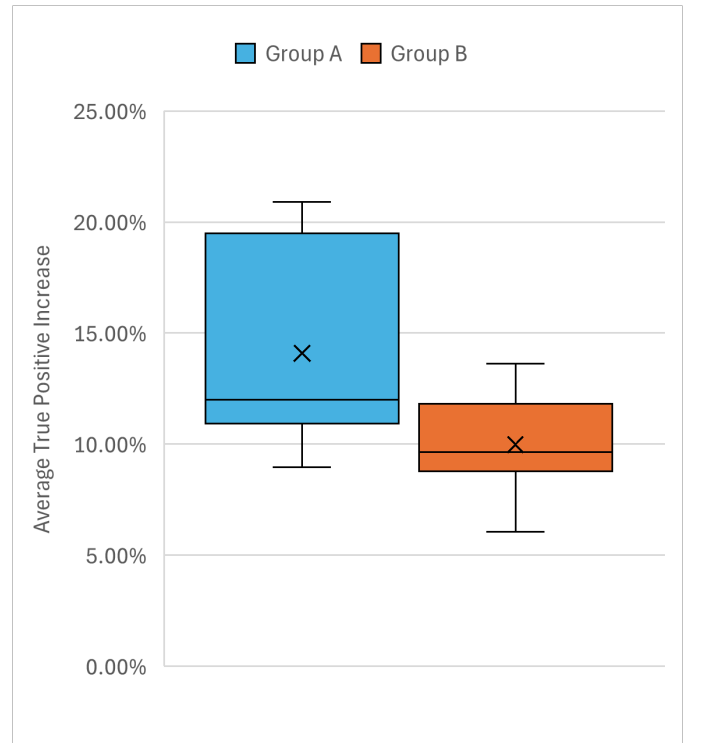


Fig. 10. Distributions of precision are shown for agents using penalized and un-penalized models from their evaluation on data from the year 2016. Results are from evaluations ranging from 2016 through 2023. Mean values are indicated with an X.

!!!The fact that the average true positive increases became more similar between the 2 groups in the temporal generalization test could help to explain why the average annual return became more similar for them as well. !!!The overall magnitude of these trade increases also suggests that the agents are making most of their profits when they sell stocks for an opening price which exceeds their 5% goal. !!!Otherwise the average increase for a successful trade would be expected to be much closer to 5%. “Fig. 12” shows the average performance over time for the models apart of the temporal generalization test in groups C and D.

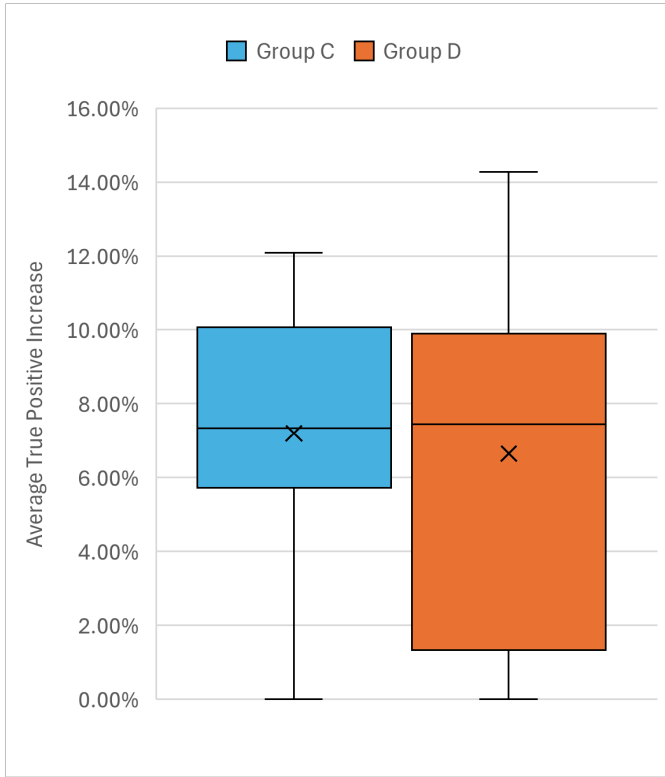


Fig. 11. Distributions of precision are shown for agents using penalized and un-penalized models. Results are from evaluations ranging from 2016 through 2023. Mean values are indicated with an X.

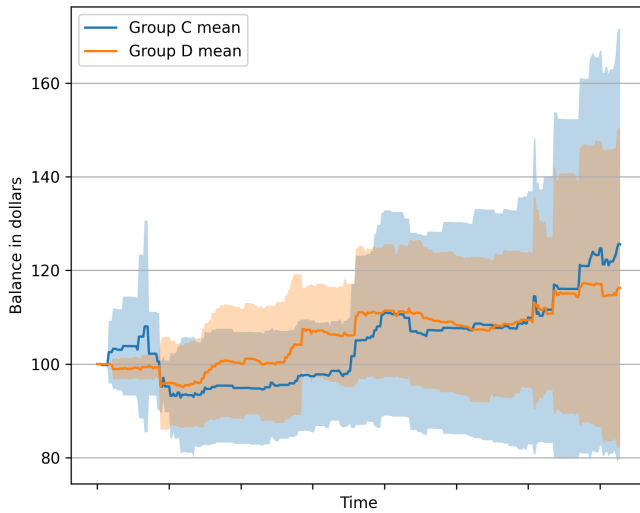


Fig. 12. The mean balances of the agents using Group C and Group D models are shown over the course of the year. For each group, every model was trained and evaluated on a different timeframe. This means that the timeframe represented is not a specific year, but just shows the average of how each model performs when different portions of its evaluation year have elapsed. The shaded area around each line represents the 95% confidence interval for the balance at that point.

## VII. CONCLUSION

This paper investigated using logit size penalties to make price predictors less prone to over-estimation and increase precision of trading algorithms. !!!The conducted ablation study demonstrated that models with this penalty did produce trading algorithms with consistently higher precision than those without it. !!!This paper demonstrated that this result could also be repeated with data from different timeframes. !!!Along with the increase in precision, trading with the penalized models also yielded an increased return. !!!This indicates that quantifying the risk of over-estimation did correlate with higher profitability.

Future work in this area should explore conducting studies that test different trading strategies in conjunction with penalized models. It is worth investigating how these results may vary when used with different price aims and less strict stop-losses. Also worth investigating is if these results could be replicated purely by changing the strategy with which the predictor is used. For example, one might ask if using the penalized regressor with a %5 increase goal is similar to using an un-penalized regressor with an even higher aim.

## REFERENCES

- [1] M. Sun, Q. Li, and P. Lin, "Short-term stock price forecasting based on an SVD-LSTM model," *Intelligent Automation & Soft Computing*, vol. 28, no. 2, pp. 369–378, Feb. 2021, doi: <https://doi.org/10.32604/iasc.2021.014962>.
- [2] Q. Li, N. Kamaruddin, S. S. Yuhaziz, and H. A. A. Al-Jaifi, "Forecasting stock prices changes using long-short term memory neural network with symbolic genetic programming," *Scientific Reports*, vol. 14, no. 1, p. 422, Jan. 2024, doi: <https://doi.org/10.1038/s41598-023-50783-0>.
- [3] H. Jiang, X. Hu, and H. Jia, "Penalized logistic regressions with technical indicators predict up and down trends," *Soft Computing*, Aug. 2022, doi: <https://doi.org/10.1007/s00500-022-07404-1>.