Application of Deep Learning to Asset Return Prediction Trading System

Date: 9th Dec., 2023

Instructors: Prof. Chris Kelliher
Prof. Eugene Sorets

Members:

Ziqian Min	U20137296
Jieyuan Deng	U45663565
Yiran Song	U54976289
Shikun Sang	U78886841
Anushree Minase	U86471202
Shihan Dai	U96333132

Introduction

The financial market is inherently uncertain, predicting asset movements is an engaging endeavor. In this project, we aim to implement deep learning models, particularly LSTM-based model and KNN-based model to predict the future asset returns and construct trading strategies for each model.

Data Description -

This project report utilizes a comprehensive dataset of 30 carefully chosen stocks from the S&P 500 index, focusing on daily fundamental features extracted from Bloomberg Terminal data spanning the period from 2010 to 2023. The chosen stocks form a representative pool, allowing for a detailed examination of key financial indicators.

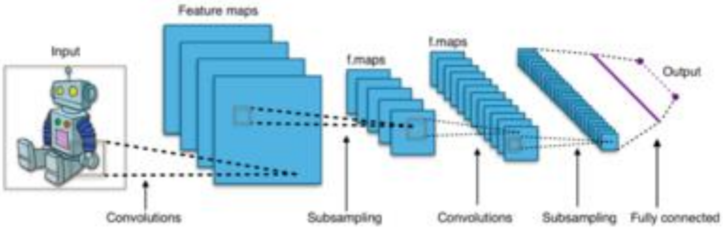
- Price Data: Open, High, Low, Close, Volume
- Stochastic Oscillator: It helps identify potential trend reversal points by comparing the closing price to the price range.
- Moving Averages: Moving averages smooth out price data, making trends more apparent. They are widely used to identify the direction of the trend and potential reversal points.
- Moving Average Convergence and Divergence (MACD): MACD is a versatile indicator providing insights into the strength and direction of a trend. Crossovers and divergences between MACD and its signal line can signal potential buying or selling opportunities.
- Momentum Indicators: Momentum indicators help identify the speed of price changes. Positive momentum suggests bullish trends, while negative momentum suggests bearish trends.
- Relative Strength Index (RSI): RSI helps assess whether a stock is overbought or oversold, indicating potential reversal points.

LSTM-based model

Methodology

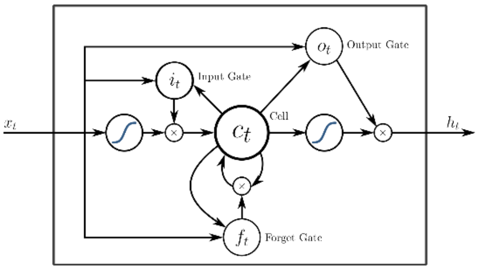
1. CNN (Convolutional Neural Network)

Convolutional Neural Network, a feed-forward neural network, can do feature engineering by its kernel optimization. It consists of input layer, hidden layers (convolutional layers, pooling layers) and output layer (fully connected layer). Though CNN is usually used in image processing, out model uses this algorithm to extracting characteristics from raw data so as to capture more important features of stocks.



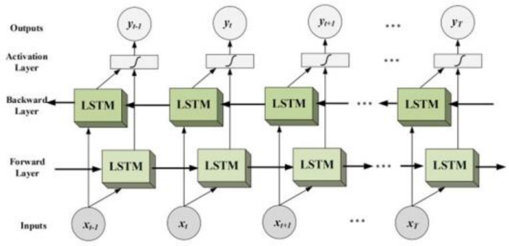
2. LSTM (Long short-term memory)

Long short-term memory network, a recurrent neural network can solve the vanishing gradient problem. It's commonly used in Time-Series problem and long-interval data. A common LSTM unit is composed of cell, input gate, output gate and forget gate.



3. Bi-LSTM

Bi-LSTM is an extension of classical LSTM model. Unlike capturing relationship from past data to next data, Bi-LSTM process sequences in both directions. It contains two separate LSTM with two separate sets of hidden states. With the help of Backward LSTM, the Bi-LSTM model may give us a better fitted model.



4. Attention mechanism

The Attention Mechanism is a selective mechanism in machine learning in order to imitate human vision, which can help the algorithm focus more on useful information.

$$Attention(Q, K, V) = \sum a_i V_i$$

5. Regression Moel & Classification Model

Based on the model output, the model can be categorized into regression model and classification model. In our project for the LSTM-based model, we initially attempted to set the model output as the stock return values with is a regression model. After obtaining unsatisfactory results from the regression model, we transform the output of the LSTM-based model form exact value of stock returns into three categories based on stock returns.

Result

- **Stock Rating System: LSTM rating, financial indicator rating**

In our rating system, there are six factors: one LSTM signal, and five financial indicators: MOM, RSI, SOD, MA, EMA. Each of the six factors independently assigns score to thirty individual stocks. The available values of the score are -1, 0, 1. The probability of giving a score of 0 is from 16.5% to 95%, differing from factors and stocks. The total score of one individual stock is the sum of the product of the weights and the separate scores. LSTM signal has the largest weight. We will select fifteen stocks with highest score and add them to our portfolio construction. The total scores of each individual stock and selection of fifteen stocks differ from day to day.

- **Portfolio Optimization by Markowitz model**

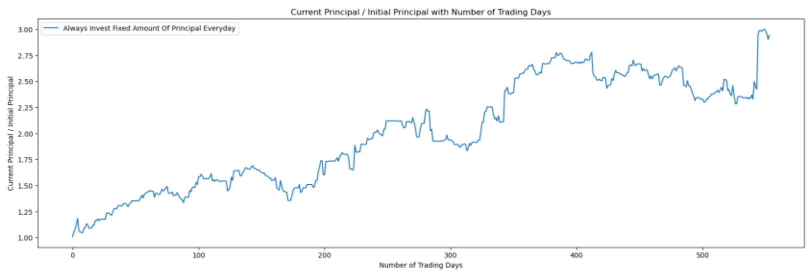
After obtaining the fifteen most trade-worthy stocks by rating system, we construct a portfolio by using Markowitz portfolio optimization model. We adjust the weights of the stocks daily based on the strategies.

There are two strategies for portfolio construction. The one is to prevent large volatility; the other is to combine the market situation and decide trading direction. For the first strategy, we set -3 and 3 as the lower and upper bounds for the weight of single stock. If Markowitz model shows us the forming the optimal portfolio requires to overbuy or oversell one particular stock by more than 3 times the principal, we will take no action on that trading day. For the second strategy, we analyze the market sentiment by using main indexes' price data and decide to long or short in the portfolio. If the sentiment reflected from the main indexes' information is optimistic, we long in our optimized portfolio; if the sentiment of market is pessimistic, we short in our optimized portfolio.

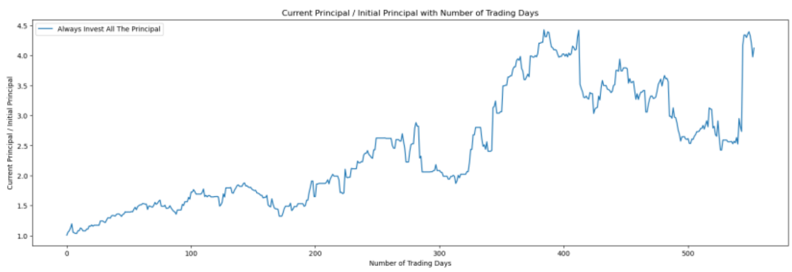
- **Portfolio Returns**

We have two methods to invest. The first method is to invest fixed amount of money every day, the second method is to always invest all the principal. The plots depict the change of the ratio of current principal to initial principal over time.

Invest Fixed Amount Everyday



Always Invest All The Principal



Irrelevant with the amount of money we invest, the Sharpe ratio of the portfolio is 1.2367.

KNN-based model

1. The main idea of the strategy

The basic idea of our group is that if we can find several days in the past that have similar features to today, we think that the future return of these past days will also be similar to today.

That's why we use KNN. For a given time t, we look into the past days to find some similar days as time t using KNN. Essentially, what our strategy do is just predict future returns.

2. General Methods

We use the k-nearest neighbor (KNN) algorithm to predict future stock returns (the target variable) based on a set of predictor (feature) variables, which can be based on technical, fundamental and some other data. Since we mainly use yahoo finance to gather data, the features we use are mainly price and volume related data. The strategy we research is a single-stock strategy, i.e., for each stock the target variable is predicted using the price and volume data only for this stock (but no cross-sectional data, i.e., no data for other stocks). The target variable $Y(t)$ is defined as the cumulative return over the next T trading dates.

$$Y(t) = \frac{P(t+T)}{P(t)} - 1 \tag{2.1}$$

The predictor variable $X_a(t), a = 1, \dots, m$, are defined using prices $P(t')$ and volumes $V(t')$ at times t' before t (i.e., $t' < t$).

We also set a time gap between the KNN sample and time t since the feature we selected includes the price. Without the gap, the k nearest neighbors found will just be $t-1, t-2, \dots, t-k$.

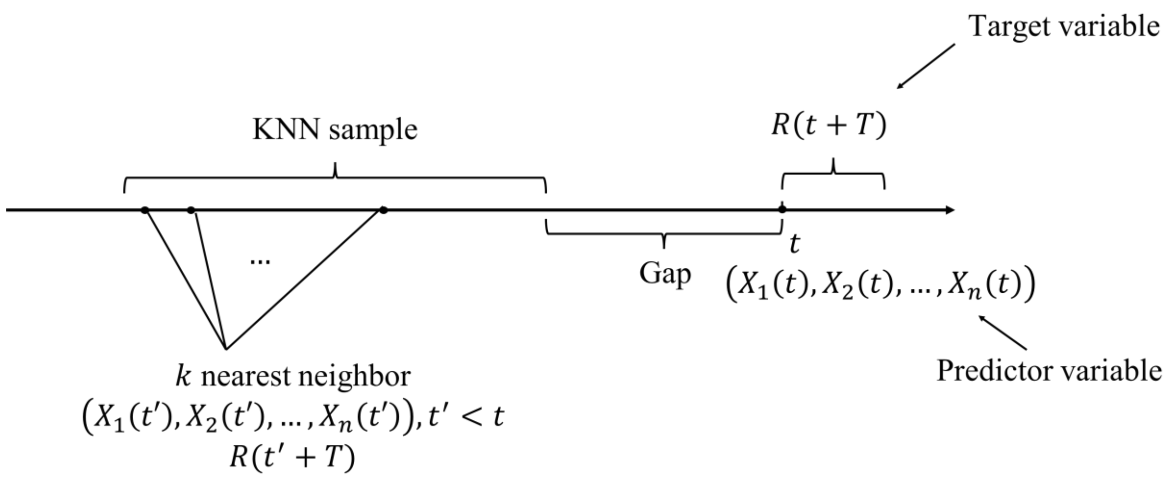


Figure 1

The predictor variables are further normalized to lie between 0 and 1:

$$\tilde{X}_a(t) = \frac{X_a(t) - X_a^-}{X_a^+ - X_a^-} \tag{2.2}$$

Where X_a^+ and X_a^- are the maximum and minimum values of $X_a(t)$ over the training period. The final ingredient is the number k of the nearest neighbors. For a given value of t we can take k nearest neighbors of the m-vector $\tilde{X}_a(t)$ among the m-vectors $\tilde{X}_a(t')$, $t' = t-1, t-2, \ldots, t-T_*$, using the KNN algorithm (here T_* is the sample size). For KNN we can use the Euclidean distance $D(t,t')$ between $\tilde{X}_a(t)$ and $\tilde{X}_a(t')$ defined as

$$[D(t,t')]^2 = \sum_{a=1}^{m} \left(\tilde{X}_a(t) - \tilde{X}_a(t')\right)^2 \tag{2.3}$$

We may use some other distance in our research. Let the k nearest neighbors of $\tilde{X}_a(t)$ be $\tilde{X}\left(t'_\alpha(t)\right), \alpha = 1, \ldots, k$. Then we can define the predicted value $\mathcal{Y}(t)$ as an average of the corresponding realized values $Y\left(t'_\alpha(t)\right)$:

$$\mathcal{Y}(t) = \frac{1}{k} \sum_{\alpha=1}^{k} Y\left(t'_\alpha(t)\right) \tag{2.4}$$

3. Decide Signal

The signal at $t = 0$ can be defined using the predicted value $\mathcal{Y} = \mathcal{Y}(0)$, which is the expected return for the next T days. For single-stock trading we define thresholds for establishing long and short trades, and liquidating existing positions, e.g., as follows:

$$\text{Signal} = \begin{cases} \text{Establish long position if } \mathcal{Y} > z_1 \\ \text{Liquidate long position if } \mathcal{Y} \le z_2 \\ \text{Establish short position if } \mathcal{Y} < -z_1 \\ \text{Liquidate short position if } \mathcal{Y} \ge -z_2 \end{cases}$$

Since we calculate $\mathcal{Y}$ using the average of the k nearest neighbors, it will be convenient for us if the returns are normal distributed. Then the mean $\mathcal{Y}$ is also normal distributed. As a result, we can determine the z values using z-score of normal distribution. Therefore, we first test the normality of the returns of the stock. And we choose the stocks that have returns closest to normal distribution. And they are the following 7 stocks.

Stock Name	P Value
ALLE	0.067495
CMG	0.000655
EQT	0.000077
GPC	0.017991
HSY	0.000902
NKE	0.00002
NUE	0.000038

Although our strategy is a single stock strategy, we apply our strategy to multiple to reduce volatility of the strategy.

4. Model Optimization

4.1 Parameter Optimization

There are several parameters to decide:

Parameters	
KNN sample size:	T_*
Gap size:	T_g
Features we use to calculate distance	$\tilde{X}_a(t)$
The n day return in target variable:	T

Here we do not conduct gradient search to find the optimal value for each value. We test several values of the parameters that we think make sense. And our final choice of parameter value is:

$$T_* = 252, T_g = 252, T = 20$$

And we include all of our features.

4.2 Calculation of Predicted Value

$$\mathcal{Y}(t) = \sum_{\alpha=1}^{k} w_\alpha Y\big(t'_\alpha(t)\big) \tag{4.1}$$

Instead of using the simple average of the corresponding realized values $Y\big(t'_\alpha(t)\big)$, we can use a weighted average. To get the value of w_α, we can divide training and testing sample and run a regression to find the weight on the kth nearest sample.

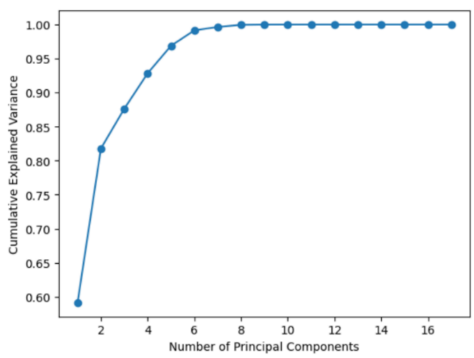
4.3 Deciding the Value of z

For the trading signal, the choice of the value of z does not affect the strategy performance a lot. Therefore, we simply set z_1 as 2 times the standard deviation $\hat{s}$ of the corresponding realized values $Y\big(t'_\alpha(t)\big)$ times square root of k and z_2 as 1 time the standard deviation $\hat{s}$ of the corresponding realized values $Y\big(t'_\alpha(t)\big)$ times square root of k. That is,

$$\mathcal{Y}(t) > 2 \cdot \frac{\hat{s}}{\sqrt{k}} = z_1, \mathcal{Y}(t) < 1 \cdot \frac{\hat{s}}{\sqrt{k}} = z_2 \tag{4.2}$$

4.4 Dimension Reduction

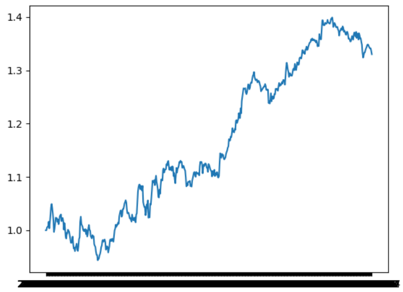
We also use PCA to reduce the noise interference. Conducting PCA to our features and significantly reduce the volatility of the strategy.



We finally reduce the dimension of our features into 3.

5. Performance

The back test period is from 9/7/2021 to 11/27/2023. The PNL of the strategy is as follows:



And the following table shows some analysis:

begin	end	slippage	initCap	stddev(annual)	sharpe
9/7/2021	11/27/2023	0	10000000	0.120908	1.228665
calmar	absReturn	annualReturn	maxDrawdown	win	winLoss
1.408149	0.330078	0.148555	0.105497	0.53046595	1.088916

We notice that the volatility of the first half of the back test period is relatively high and the back test performance on the most recent data is not good. This means that we need to observe for more time to judge whether this strategy is profitable in the future.

6. Future Optimizations

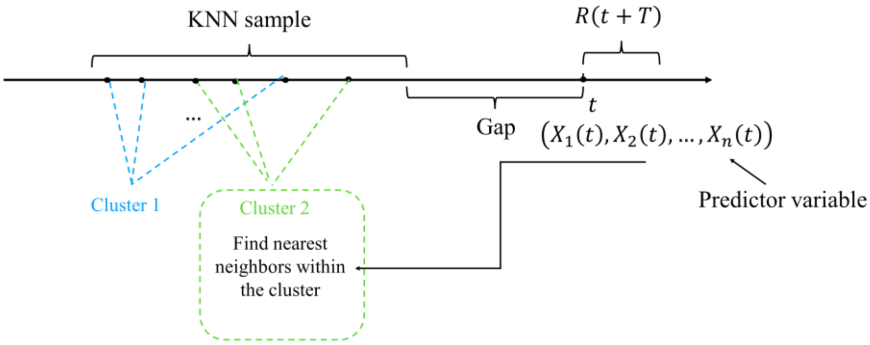
6.1 Feature importance

Some features may be more important, we may do some analysis and put more weight on some features when calculating neighbors. That is, when calculating the distance, we can add a term w_α based on the importance of different features.

$$[D(t,\ t')]^2 = \sum_{a=1}^{m} w_\alpha \left(\tilde{X}_a(t) - \tilde{X}_a(t')\right)^2$$

6.2 Combine with clustering

For the past samples to calculate the nearest neighbors, we do not include the influence of future return, because we do not know the future return at time t. We can first do a supervised clustering on the past sample. Determine which cluster the predictor belongs to and find nearest neighbors with in the cluster.



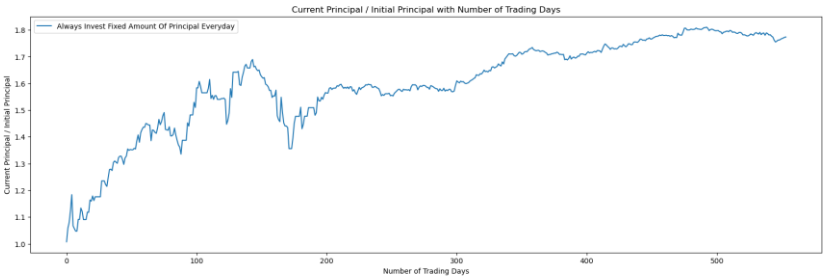
Strategy Rotation & Portfolio Reconstruction

- **Strategy Rotation**

We have two deep learning models and strategies. We will switch to use these two models to pursue the optimal portfolio. We simulate the returns before entering the market. If one model performs better in simulation phase (has higher Sharpe ratio), it will be selected as prediction model for the first phase. If it still outperforms another model in the first phase, it will be the prediction model for the second phase; otherwise, another model will replace this model and so forth.

We divide the whole test set totally 555 days into 5 subsets, the first phase has 74 returns and other four phases have 120 returns for each of them. The result is that the earliest 195 trading days are with LSTM-based model and later 360 trading days are with KNN-based model. The final Sharpe ratio is 1.2270. The cumulative returns are shown below.

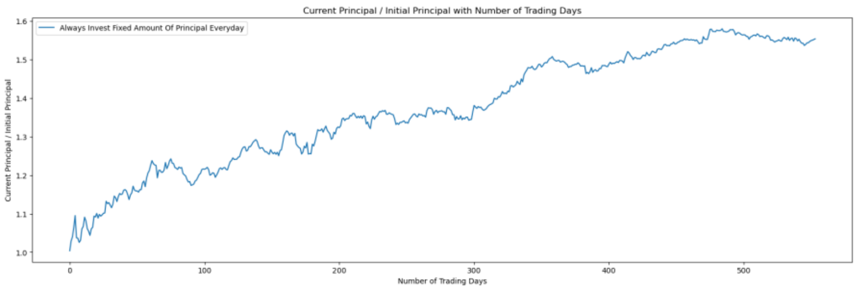
Invest Fixed Amount Everyday



- **Portfolio Reconstruction**

Instead of switching to use these two models, we try combining two portfolios gained by two different models. We use Minimum Variance Portfolio Formula to reconstruct a new portfolio from the old two portfolios. We calculate the covariance matrix of these two portfolios of early 120 trading days and gain the weight of each portfolio based on minimum variance portfolio formula. We adjust the weights every 120 trading days. The Sharpe ratio surprisingly rises to 1.9622. The cumulative returns are shown below.

Invest Fixed Amount Everyday



Conclusion

We present two deep learning-based trading strategies: LSTM-Based and KNN-Based. Employing a rolling back-testing approach every 120 days, we dynamically select the optimal strategy based on cumulative returns and Sharpe ratio. This alternating process ensures adaptability to evolve market conditions, enhance robustness in real-time trading

Reference

[1] Kakushadze, Z., & Serur, J. A. (2018). RETRACTED CHAPTER: Introduction and Summary. In *RETRACTED BOOK: 151 Trading Strategies* (pp. 1-1). Palgrave Macmillan, Cham.

[2] Jiang, W. (2021). Applications of deep learning in stock market prediction: recent progress. *Expert Systems with Applications*, *184*, 115537.

[3] Xiong, Z., Liu, X. Y., Zhong, S., Yang, H., & Walid, A. (2018). Practical deep reinforcement learning approach for stock trading. *arXiv preprint arXiv:1811.07522*, 1-7.

[4] Lin, H., Zhou, D., Liu, W., & Bian, J. (2021, August). Learning multiple stock trading patterns with temporal routing adaptor and optimal transport. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (pp. 1017-1026).

[5] Zhang, X., & Tan, Y. (2018). Deep stock ranker: A LSTM neural network model for stock selection. In *Data Mining and Big Data: Third International Conference, DMBD 2018, Shanghai, China, June 17–22, 2018, Proceedings 3* (pp. 614-623). Springer International Publishing.

[6] Prosky, J., Song, X., Tan, A., & Zhao, M. (2017). Sentiment predictability for stocks. *arXiv preprint arXiv:1712.05785*.