DSA5205 Project Report

Group Name: **Confused-Matrix**

Project Title: **Automated Trading for Investors Using Machine Learning**

Members: Haidi Azaman A0216941E | SK Ruban A0253837W | Liu Tsun Hei A0275764N | Bryan Wong A0215114X

1 Executive Summary

Research Objectives: Design an automated trading algorithm. It consists of stock selection, ML-based signal-generation and trade execution based on the signals. The final result can be used by retail traders who like to invest in the stock market but do not have time to follow the market closely.

Approach Chosen: Choose the 10 stocks from the investment universe using random forest classifier; basic time series analysis on data; feature engineering based on on a grid of 72 features (derived from technical indicators such as VWAP and EMAs); train a fully connected neural network (FCN) to generate trading signals (long, neutral and short signals); rebalance the portfolio with equal-weight.

Roles:

Haidi: FCN training, Literature review

Ruban: Stock selection, Markowitz portfolio

Hei: Time series analysis, trading strategy execution

Bryan: Feature engineering, FCN training

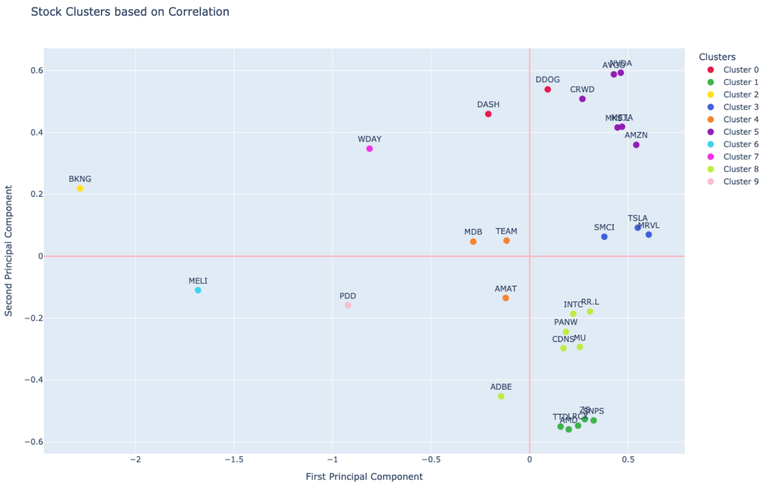
Outcomes: The FCN can predict the correct signal with around 40% accuracy (across 3 target classifications). The trading strategy based on the Neural Network model trained outperforms our baseline strategy (the equal-weighted buy-and-hold strategy for the 10 stocks that we have selected) by 3%. However the strategy did not earn a positive return in the testing period (1st - 31st July).

2 Stock Selection

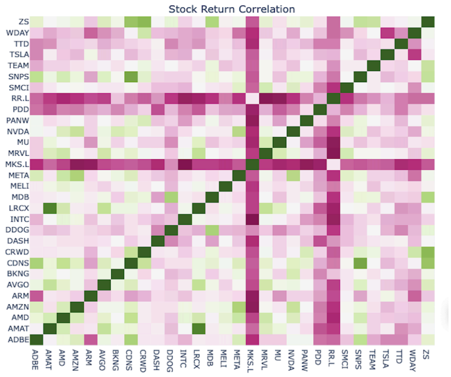
Our stock universe comprised NASDAQ 100, FTSE 100, and 20 SGX stocks, balancing liquidity, diversification and returns. Using 60-minute data from Yahoo Finance up to January 1, 2024, we employed a Random Forest Regression model with windowed metrics (detailed in **5. Feature Engineering**) to select the top 30 stocks. The model demonstrated robust performance with a test RMSE of 0.0088 and R-squared of 0.8904. Feature analysis showed 30-day momentum as the primary factor, with 90-day momentum and 30-day Sharpe ratio following.

We then employed the following approach to refine to 10 stocks:

1. Applied **PCA** to the correlation matrix of daily returns (2022 - 2023), reducing dimensionality to two components explaining 72.94% of total variance.
2. Used **K-means clustering** to group stocks into 10 clusters based on PCA coordinates.



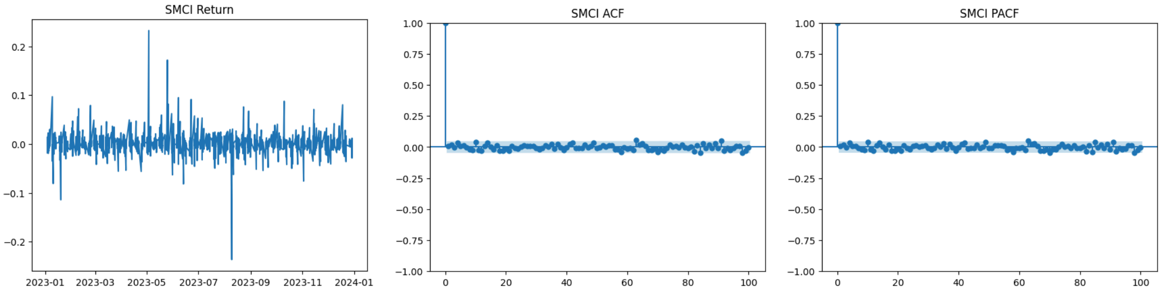
3. Selected the stock with the **highest expected annualised return** from each cluster.



Our final 10 stocks are: DDOG, SNPS, BKNG, SMCI, MDB, NVDA, MELI, WDAY, MU, PDD, with MU replacing RR.L to avoid time zone complications.

3 Dataset Analysis

We ran some analysis on the data and tried to build simple time series models such as ARMA to model the market price data. For instance, the plots below show the analysis done for the stock SMCI (Tsay, 2010). The returns reveal stationarity (confirmed by ADF test), but ACF and PACF plots indicate no significant autocorrelation. This suggests that linear time series models like ARMA or ARCH are not suitable for modelling these returns or their volatility; advanced approaches are needed to model the potential non-linearity.



4 Trading Strategy

For our trading strategy, we will build a Machine Learning model to classify 3 types of trading signals: Long, Neutral and Short. This classification will be done on each of the 10 stocks on an hourly basis. We will then allocate equal portfolio weights to the stocks with either Long or Short classifications, and adjust the portfolio weights hourly, taking equal cash positions on each of the stocks.

For this project, we have the following assumptions for our trading strategy: the market is liquid; assets are divisible; there are no transaction costs or bid-ask spreads; there is no additional margin required for short positions (in the US, the total margin is 150%); and all stocks share the same return distribution over training and testing periods.

5 Feature Engineering

To train the Neural Network model, these are the main features:

1. **Volatility**: Rolling standard deviation of Close Price
2. **EMA of Volume and Price**: divided against current Close Price/Volume

3. **SMA of Volume and Price**: divided against current Close Price/ Volume
4. **Momentum:** Current Close Price / Close Price X hours ago
5. **VWAP:** Volume Weighted Average Price across past X hours
6. **Fama French Market Returns**
7. **Open Minus Close / High Minus Low**
8. **MACD**: Dropped due to low pearson r value

These features are calculated across multiple timeframes (2 to 256 hours) to capture both short-term and long-term market dynamics. The final 20 most influential features are selected using scikit-learn's SelectKBest function with ANOVA F-value criteria, ensuring an optimised input set for the model.

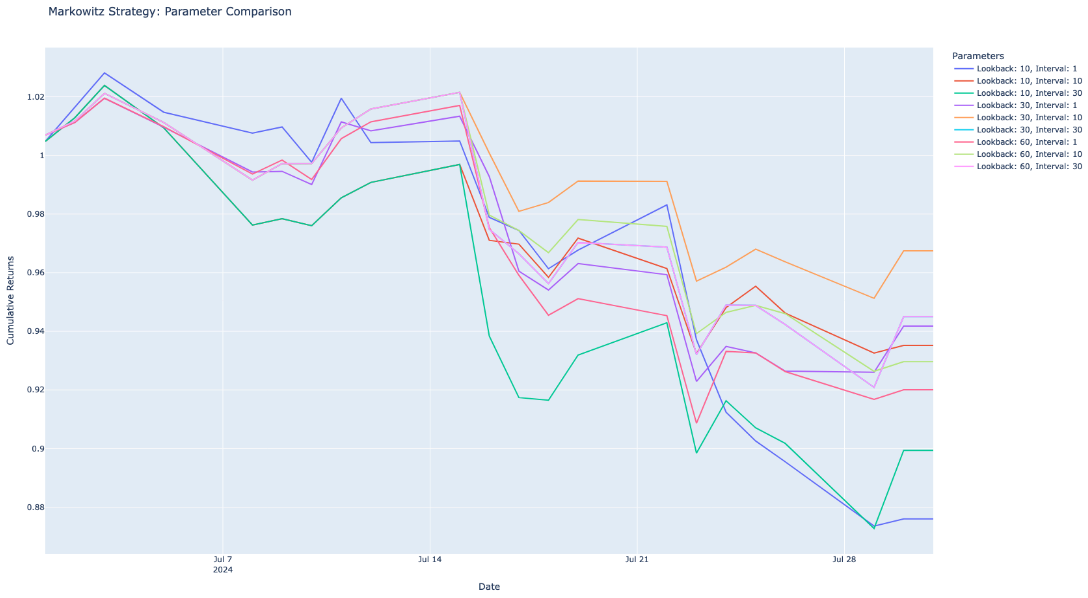
6 Methods

6.1 Markowitz Portfolio Optimisation

Our first approach leverages Markowitz's Modern Portfolio Theory (Jansen, 2020) to optimise stock allocations. This method aims to maximise the portfolio's **Sharpe ratio**, balancing return and risk:

1. Rebalance every 10 days using a 30-day lookback period
2. Optimization constraints: no short-selling, full investment

Through extensive testing of various parameters, we determined that a 10-day lookback with a 30-day rebalancing interval produced optimal results. While this strategy underperformed the S&P 500 in July 2024 (see **7 Findings**), it demonstrated significant outperformance over the January-July 2024 period, achieving a cumulative return of 1.96 and a Sharpe ratio of 2.71, surpassing the benchmark.



<u>6.2 Fully-connected Neural Networks (FCN)</u>

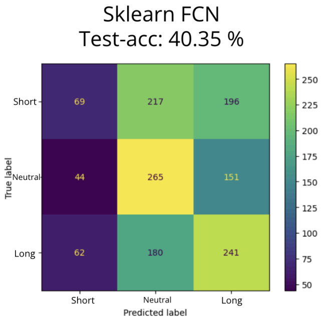
Expanding on Lee & Chen's (2007) work demonstrating the effectiveness of neural networks with simple technical indicators (Oğuz et al., 2019) for stock prediction, we developed a more sophisticated model using complex indicators and hourly data.

The target classes are derived from log returns for the next timestamp, referred to as log_return_shifts. They are classified according to percentiles: 0th-33rd as **Short**, 33rd-66th as **Neutral**, and 66th-100th as **Long**. Using sklearn, we initialised a `param_grid` and employed `ParameterGrid` to explore various configurations of an `MLPClassifier`. While this approach yielded decent results, it had limitations in diagnosing and improving model performance due to the inability to observe train and validation loss curves in real-time, making it challenging to identify overfitting or underfitting. We have also experimented with different Machine Learning models, such as XGBoost, but our Neural Networks model yields the best results.

7 Findings

Our final FCN model achieved a test accuracy of 40.35%, This is a positive outcome compared to the random guess baseline of 33%. Assuming $100 initial capital, the table below summarises the performance of all our strategies. The FCN model is shown to outperform the equal-weighted buy-and-hold strategy for the 10 stocks that we have selected.

Jul 2024	S&P 500	Equal-Weighted	Markowitz	FCN
PnL (in dollar)	0.86	-2.79	-2.84	-0.38



8 Conclusion

Contributions: Our method is designed for individuals without a finance background who want to trade. The entire process, from stock selection to signal generation and trade execution, is fully automated, allowing anyone to engage in trading without requiring in-depth financial knowledge.

Future Work: This project achieved decent results and shows the feasibility of this strategy with more fine-tuning, improved models and more data for training. Future work includes applying the same strategy to a more granular timeframe, e.g. price data in minutes, or even seconds. We could also apply the strategy to different markets and asset classes, such as cryptocurrency or commodities. Lastly, we could improve our Machine Learning models by trying out different models and architectures, such as XGBoost, and better features could be engineered to model the market trends better, which could greatly improve the model performance.

References

- Jansen, S. (2020). Machine learning for algorithmic trading. Packt Publishing.
- Lee, C. T., & Chen, Y. P. (2007, November). The efficacy of neural networks and simple technical indicators in predicting stock markets. *(pp. 2292-2297)*. IEEE.
- Tsay, R.S. (2010) Analysis of Financial Time Series. 3rd Edition
- R. F. Oğuz, Y. Uygun, M. S. Aktaş and İ. Aykurt, On the Use of Technical Analysis Indicators for Stock Market Price Movement Direction Prediction, 2019