MONASH University

Stock Portfolio Optimisation
ADS2001 2024 Group 1
Final Project Report

Zhi Xian Tan

Chung Ling Lai

Sneha Binu

Sanugi Fernando

Table of Contents

Executive Summary

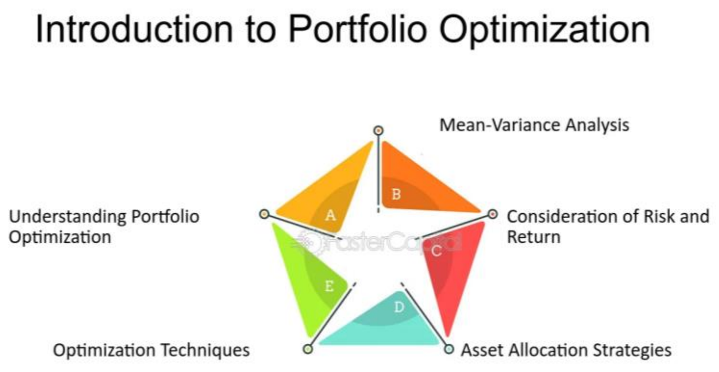

Introduction to Portfolio Optimization
Mean-Variance Analysis
Understanding Portfolio Optimization
Consideration of Risk and Return
Optimization Techniques
Asset Allocation Strategies
A
B
C
D
E

Figure 1: Diagram showing key aspects of portfolio optimisation process

In today's dynamic financial markets, effective portfolio management is crucial for maximising returns while minimising risks. Stock portfolio optimization is a systematic approach that aims to construct an efficient portfolio by allocating investments across various assets based on individuals' or audiences' expected returns, risks, and correlations. This executive summary outlines the problem of stock portfolio optimization and presents key findings and recommendations of this project.

A stock, also referred to as a share or equity, represents ownership in a company. Stocks are available for purchase by individuals, institutional investors, and corporate, and government entities. Stockholders who hold a significant share in a company typically have voting rights on the company's decisions and receive dividends based on profits. The value of a stock fluctuates based on the company's performance, market conditions, and investor sentiment. With the rise of online trading platforms, the accessibility of the stock markets has increased in past years, allowing individual investors to participate more easily. A stock portfolio is a collection of stocks from different companies owned by an individual or institution. Diversifying investments in this manner spreads risk across various companies. The goals behind curating a stock portfolio typically depend on the investor's risk tolerance, time horizon, financial objectives, and overall strategy. Stock portfolios may be used to generate income, increase the value of particular stocks over time, or preserve wealth against economic fluctuations and inflation.

Investors often face the challenge of determining the optimal allocation of their capital across different stocks to achieve their desired risk-return trade-off. The general goal of stock portfolio optimization is to create a well-diversified portfolio that maximises expected returns for a given level of risk or minimises risk for a desired level of expected return. This problem has become increasingly complex as the number of potential investments grows, making it difficult to manually analyse and optimise a portfolio.

One of the foundations of stock portfolio optimization lies in the principles of Modern Portfolio Theory. Modern Portfolio Theory allows investors to assemble an asset portfolio that maximises expected returns for a given level of risk (CFI, 2024). It provides a quantitative framework for constructing efficient portfolios by considering the expected returns, risks (measured by variance or standard deviation), and correlations among assets (See Figure 1).

The Efficient Frontier is also significant, representing the set of portfolios that offer the highest expected return for a given level of risk or the lowest risk for a given level of expected return. Portfolios on the Efficient Frontier are considered optimal as they provide the best possible risk-return trade-off. Another key aspect of stock optimization is diversification, which involves investing in a variety of assets with low or negative correlations. This approach allows investors to reduce the overall risk of their portfolio without sacrificing expected returns.

Lastly, to optimise a stock portfolio, various optimization techniques can be employed, such as mean-variance optimization, risk parity, and Black-Litterman models (See Figures 2 and 3). These techniques use mathematical algorithms and computational methods to identify the optimal asset allocation based on the investor's risk preferences and return expectations.

This investigation aims to utilize the given information detailing the prices of stocks in the S&P 500 at the time of the closing of the market to conclude the return. The large dataset provides a large amount of insight into market trends over a large period. The general strategy involved swing trading over a fixed period of 10 years. Statistical calculations were made using the information given. This allowed a statistical model to reflect the changes in stock prices over time. Then, machine learning algorithms were used to predict the output of this statistical model. The aim of this was to train a model that could effectively optimise a portfolio and calculate its return. The running times and accuracy of models using different algorithms such as KMeansClustering, RandomForestClassifier, and LightGBM were studied. The final model chosen used LightGBM which provided a balance of running time and accuracy. Two main results were created using the model; a tradebook and a summary logbook. The tradebook of stock shows details of trades made of that stock and the summary log book contains key information about the portfolio such as the top 10 stocks with the highest or lowest return.

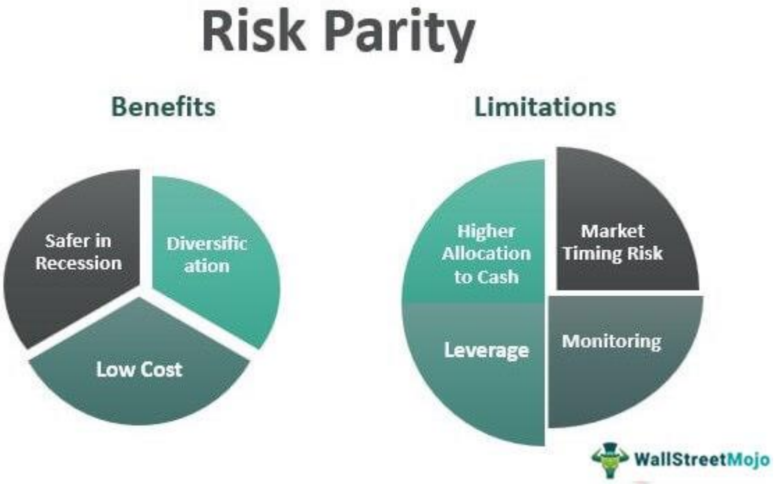


Figure 2: Diagram summarising the benefits and limitations of the Risk Parity approach

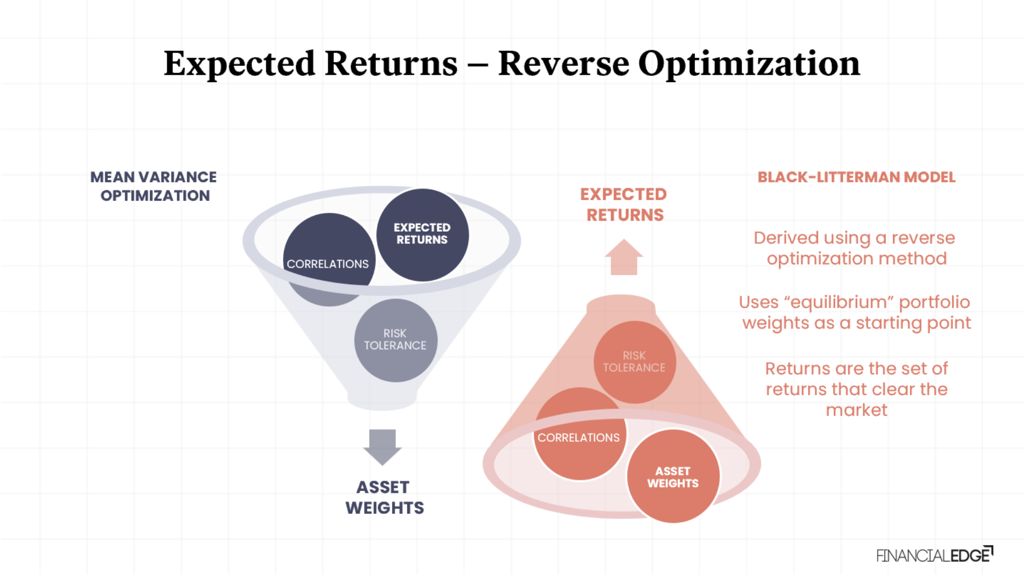
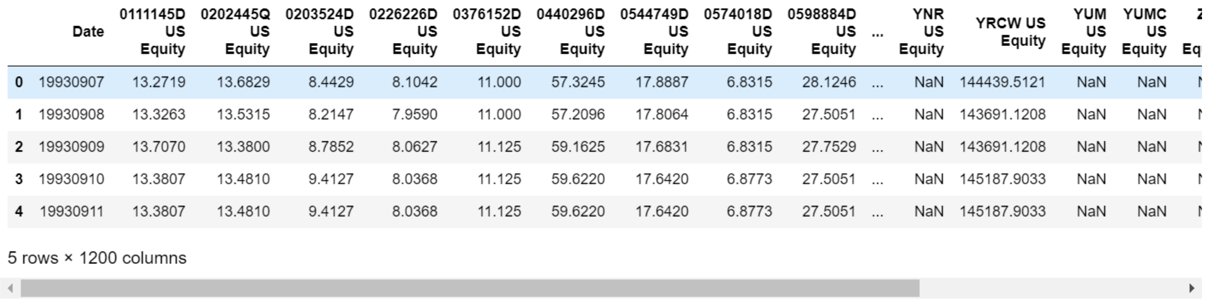

Expected Returns – Reverse Optimization
MEAN VARIANCE OPTIMIZATION
CORRELATIONS
EXPECTED RETURNS
RISK TOLERANCE
ASSET WEIGHTS
EXPECTED RETURNS
RISK TOLERANCE
CORRELATIONS
ASSET WEIGHTS
BLACK-LITTERMAN MODEL
Derived using a reverse optimization method
Uses "equilibrium" portfolio weights as a starting point
Returns are the set of returns that clear the market
FINANCIALEDGE

Figure 3: Diagram summarising derivations of Black-Litterman model

Introduction

The stock market is a complex and dynamic system where predicting future stock prices is a challenging task. This problem domain involves analysing historical stock data, identifying patterns, and developing models to forecast future price movements. The goal is to optimise investment strategies and maximise returns while effectively managing risks.

The primary dataset used in this study is the S&P 500 index, which comprises 500 large-cap companies listed on the U.S. stock exchanges. The data spans from September 1993 to July 2019 and contains the adjusted close prices for a large number of stocks over this period. Each column represents a different stock or security, identified by its ticker symbol, and the rows correspond to different dates, as shown in *Figure 4*.



	Date	0111145D US Equity	0202445Q US Equity	0203524D US Equity	0226226D US Equity	0376152D US Equity	0440296D US Equity	0544749D US Equity	0574018D US Equity	0598884D US Equity	...	YNR US Equity	YRCW US Equity	YUM US Equity	YUMC US Equity	Z Eq
0	19930907	13.2719	13.6829	8.4429	8.1042	11.000	57.3245	17.8887	6.8315	28.1246	...	NaN	144439.5121	NaN	NaN	N
1	19930908	13.3263	13.5315	8.2147	7.9590	11.000	57.2096	17.8064	6.8315	27.5051	...	NaN	143691.1208	NaN	NaN	N
2	19930909	13.7070	13.3800	8.7852	8.0627	11.125	59.1625	17.6831	6.8315	27.7529	...	NaN	143691.1208	NaN	NaN	N
3	19930910	13.3807	13.4810	9.4127	8.0368	11.125	59.6220	17.6420	6.8773	27.5051	...	NaN	145187.9033	NaN	NaN	N
4	19930911	13.3807	13.4810	9.4127	8.0368	11.125	59.6220	17.6420	6.8773	27.5051	...	NaN	145187.9033	NaN	NaN	N

5 rows × 1200 columns

Figure 4: Screenshot of first 5 rows of the given dataset

What is the S&P 500 index?

The S&P 500 index is a market capitalization-weighted index of the 500 leading publicly traded companies in the United States (Kenton, 2023). It is seen as one of the best indicators of overall U.S. stock performance. The S&P 500 is a common benchmark against which portfolio performance can be evaluated. Each listed stock does not simply represent 1/500th of the index. Massive companies such as Apple (AAPL 1.22%) and Amazon (AMZN -0.58%) have a greater impact on the S&P 500 index than relatively smaller companies like General Motors (GM 1.02%) (Frankel, 2024).

In the stock market, various factors influence stock prices, including company fundamentals, macroeconomic conditions, investor sentiment, and market trends. The relationships between these factors and stock prices are often complex and non-linear, making it difficult to develop accurate predictive models. One of the key challenges in stock price prediction is the presence of noise and volatility in the data. Stock prices can fluctuate rapidly due to various external factors such as COVID-19 and economic recessions, making it difficult to identify clear patterns and trends. Additionally, the non-stationarity of financial time series data, where statistical properties change over time, poses another challenge for traditional modelling techniques.

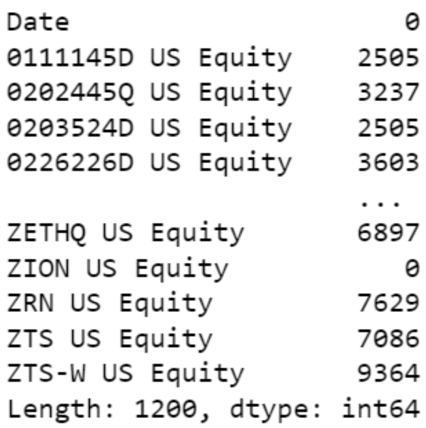
This project aims to perform an in-sample optimization of a trading portfolio. We prefer focusing on the best bull market in the time frame. It is observed that 2010 to 2019 was the best bull market (Duggan, 2023). The 10-year period from 2010 to 2019 could avoid large anomalies in the data analysis, such as COVID-19 and the economic recession.

We implemented strategies such as swing trading, stop earnings, and stop loss criteria to benefit our investors potentially. The goal of our strategy is to make a 3% profit in total. In our project, we first cleaned the data to improve its quality. We then built a statistical model to generate buy and sell signals. Subsequently, we developed trading metrics such as MACD, RSI, EMA, and SMA, and implemented these into our algorithm. Using these metrics, we applied a machine-learning model to classify the signals. After implementing the machine learning model on each stock, we built a trade book to evaluate the strategy's performance on each stock and determine whether it yielded a positive or negative return. Finally, we created a summary logbook to assess the overall performance.

Data Quality

Data preprocessing is a crucial step in preparing data for machine learning models, including those used for classifying the S&P 500 index performance. This process also helps us understand the dataset in more depth. Data quality is essential because the model’s performance heavily depends on the quality of the input data. The presence of missing values and potential data quality issues could affect the model’s accuracy.

To assess the potential quality issues with the provided dataset, we examined some basic descriptive statistics and visualisations. Firstly, we checked the number of missing values in each column, as shown in Figure 4.1.



```
Date                      0
0111145D US Equity     2505
0202445Q US Equity     3237
0203524D US Equity     2505
0226226D US Equity     3603
                       ...
ZETHQ US Equity        6897
ZION US Equity            0
ZRN US Equity          7629
ZTS US Equity          7086
ZTS-W US Equity        9364
Length: 1200, dtype: int64
```

Figure 4.1: Screenshot showing the number of missing values in each column of dataset

Many columns had a significant number of missing values, with some columns having over 7,000 missing entries out of around 9500 rows. Missing values represent times when the stock price was not recorded as that particular stock was not part of the top 500 of all stocks in the S&P 500. This high proportion of missing values could introduce bias and affect the accuracy of any models trained on this data. First, we converted the dates in the 'Date' column to a datetime format that could be interpreted as passing time by models. The graph below, in Figure 4.2, shows an example of how a date and a specific stock dataset should look over the years.

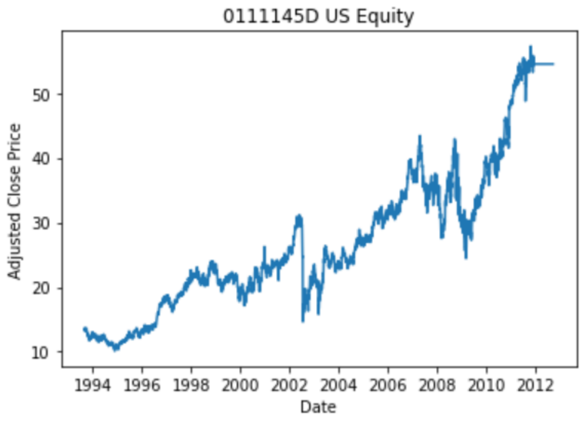


Figure 4.2: Graph of Close Price of Stock 0111145D US Equity from 1994-2012

As mentioned in our executive summary and introduction, we decided to use the dataset from 2010 to 2019, which is 10 years of the S&P 500 data. We chose this period to avoid the economic recession and COVID-19, making 2010 a suitable starting point for our analysis. For the missing values (NaN) in these 10 years, we filled them with the first or last valid observation in each column using forward fill and backward fill methods. This approach, rather than imputing zeroes, provided a more accurate representation of the closing values.

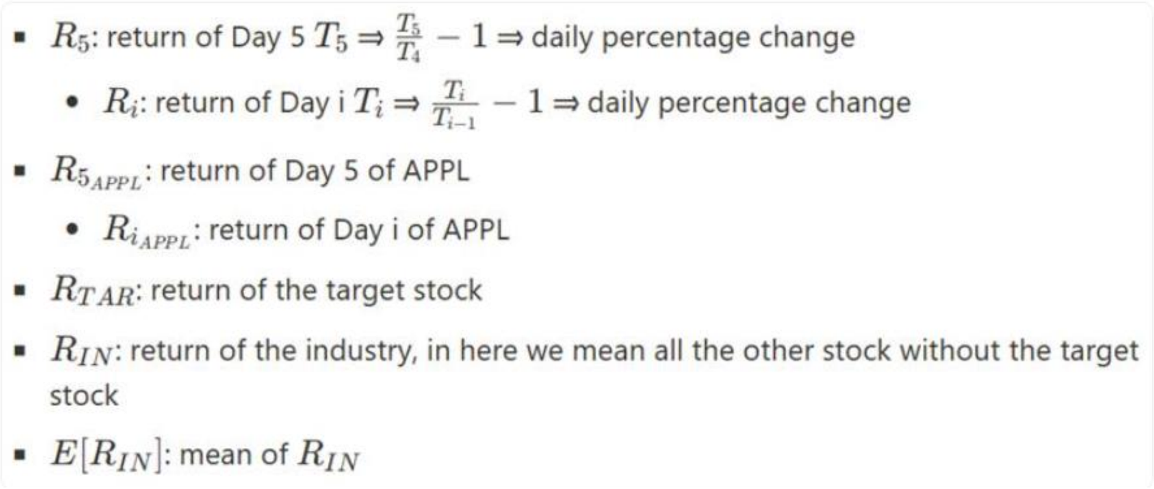
We also checked for duplicate columns and inconsistencies that needed to be addressed through data-cleaning techniques. We identified 1,089 duplications and decided to remove them for more accurate findings and to keep the dataset neat.

Due to the large size of this dataset, it was deemed unreasonable to check the accuracy of the recorded stock prices. As the dataset was used only to train the model, whether the data is precise or not is redundant. However, we did find some discrepancies in the results section of the project (See Page 16).

Strategy

We established clear buying and selling criteria to guide our trading decisions. Our buying criterion was straightforward: initiate a purchase when a buying signal is generated. The selling criteria were designed to manage risk and maximise profits through Stop Loss and Take Profit strategies. We would sell if a sell signal was triggered, if the holding period exceeded 120 trading days, if the closing price dropped 10% from the entry price, or if the closing price fell 10% from the trade's peak value. These criteria ensured that we systematically captured gains while minimising potential losses, thereby optimising our stock portfolio's performance.

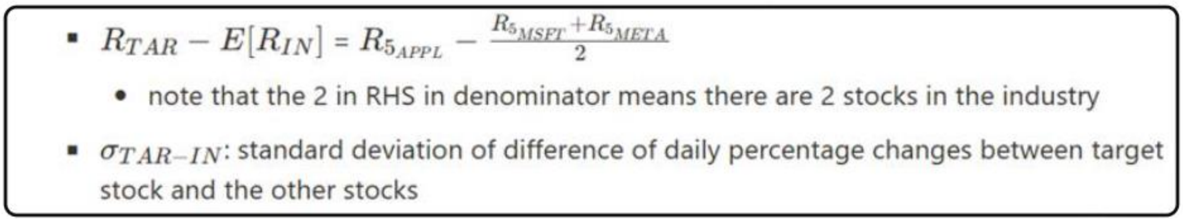
Modelling



- R_5: return of Day 5 $T_5 \Rightarrow \frac{T_5}{T_4} - 1 \Rightarrow$ daily percentage change
  - R_i: return of Day i $T_i \Rightarrow \frac{T_i}{T_{i-1}} - 1 \Rightarrow$ daily percentage change
- $R_{5_{APPL}}$: return of Day 5 of APPL
  - $R_{i_{APPL}}$: return of Day i of APPL
- R_{TAR}: return of the target stock
- R_{IN}: return of the industry, in here we mean all the other stock without the target stock
- $E[R_{IN}]$: mean of R_{IN}

Figure 5.1: Variables used for statistical model and their significance

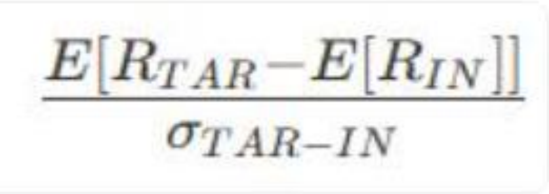
We created a statistical model that could provide us with the most accurate returns using many measures of value (See Figure 5.1). In the first model assumption, we used R_i to get the return of Day_i T_i => T_i / T_{i-1} to get the daily percentage change. R_{TAR} tells us the return of the target stock. E[RIN] gives us the mean average of RIN. These 2 variables are important as they help us calculate our target stock.



- $R_{TAR} - E[R_{IN}] = R_{5_{APPL}} - \frac{R_{5_{MSFT}} + R_{5_{META}}}{2}$
  - note that the 2 in RHS in denominator means there are 2 stocks in the industry
- σ_{TAR-IN}: standard deviation of difference of daily percentage changes between target stock and the other stocks

Figure 5.2 : First part of the formula that forms the foundation of model to be built

The model aims to find the difference between the target stock and the industry by using the mean return of the target stock and reducing it by the return of the industry of all the other stocks excluding the target stock (See Figure 5.2). Then, we divide this by the standard deviation of the difference in daily percentage changes between the target stock and the other stocks. This is shown in Figure 5.3. The model presents the gap between the target stock and other stocks in the industry, as shown in Figure 5.4. The concept of the model is slightly similar to the concept of beta value, which is a measure of a stock's volatility concerning the overall market (McClure, 2020).



$$\frac{E[R_{TAR} - E[R_{IN}]]}{\sigma_{TAR-IN}}$$

Figure 5.3: Formula of value to be calculated

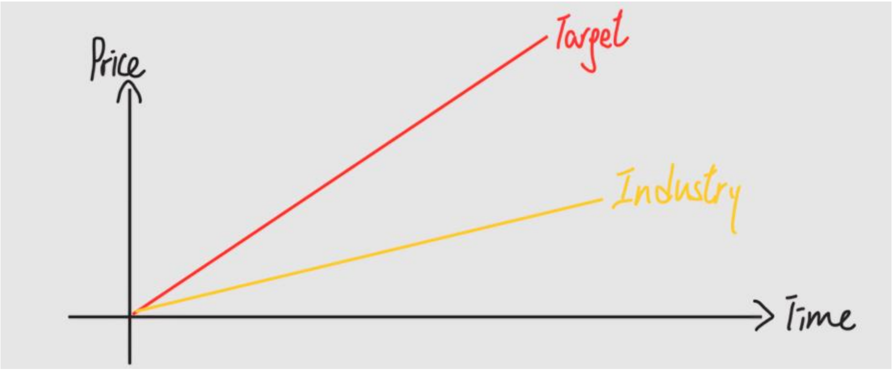
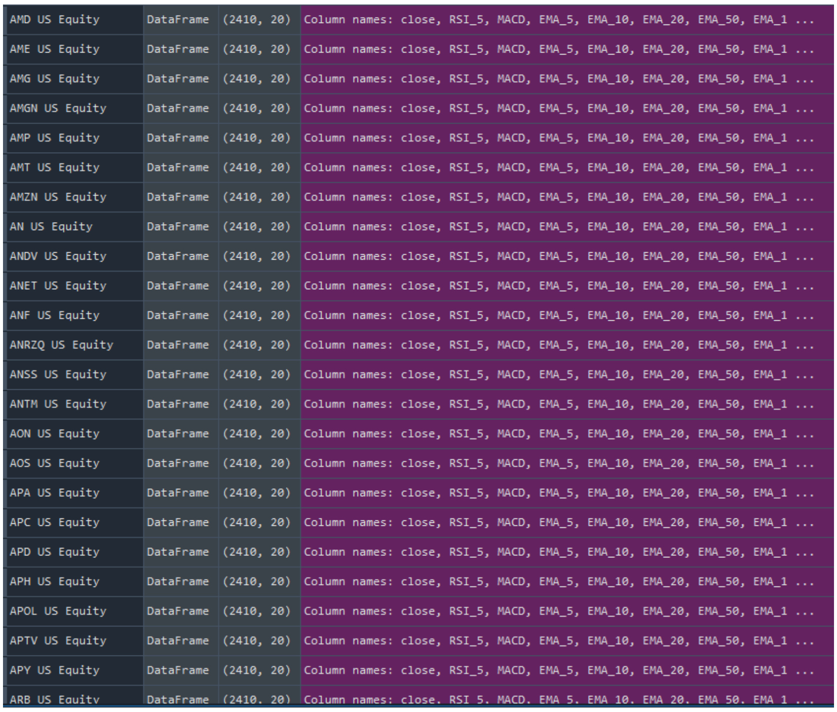


Figure 5.4: Graph showing the gap between Target stock and other stocks in the industry

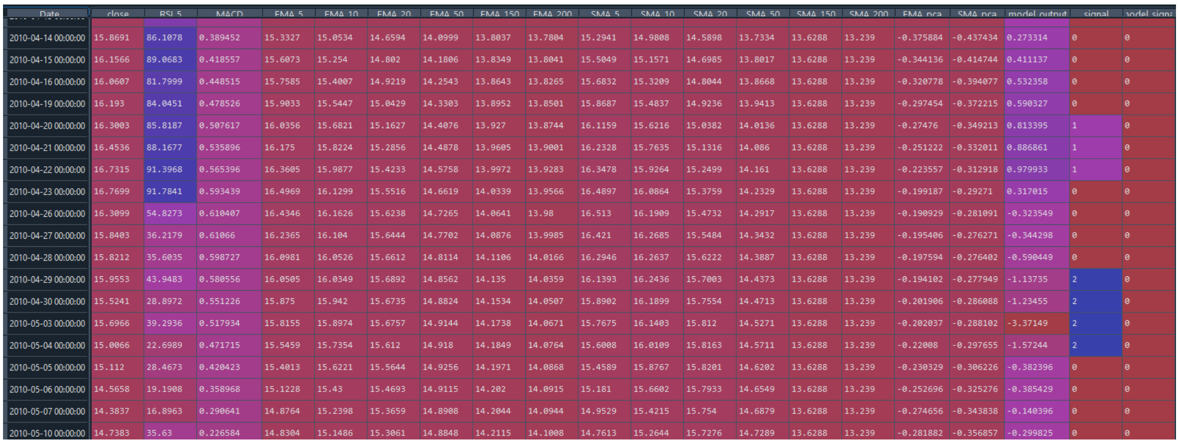
The mean and standard deviation are rolling. Regarding the beta value, the window was initially considered as a magnitude of 1.5. Through trial and error, 0.8 was found to be the best magnitude. For instance, if the model output is greater than or equal to 0.8, it generates a buy signal (1). It generates a sell signal (2) if it is less than or equal to -0.8. Otherwise, it returns 0, indicating no action is recommended.



AMD US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AME US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AMG US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AMGN US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AMP US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AMT US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AMZN US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AN US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
ANDV US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
ANET US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
ANF US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
ANRZQ US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
ANSS US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
ANTM US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AON US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
AOS US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
APA US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
APC US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
APD US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
APH US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
APOL US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
APTV US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
APY US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...
ARB US Equity	DataFrame	(2410, 20)	Column names: close, RSI_5, MACD, EMA_5, EMA_10, EMA_20, EMA_50, EMA_1 ...

Figure 6: Screenshot of contents of ticker dictionary

Given the time-series nature of stock price data, we decided to enhance stock portfolio optimization with additional informative columns for each stock. In Figures 5 and 6, we can see that the additional informative columns consist of Close, RSI, MACD, EMA, SMA, EMA_PCA, SMA_PCA, Model Output, Signal, and Predicted Signal. All stocks are entered into a ticker dictionary, as viewed in Figure 6.



Date	close	RSI_5	MACD	EMA_5	EMA_10	EMA_20	EMA_50	EMA_150	EMA_200	SMA_5	SMA_10	SMA_20	SMA_50	SMA_150	SMA_200	EMA_pca	SMA_pca	model output	signal	odel signa
2010-04-14 00:00:00	15.8691	86.1078	0.389452	15.3327	15.0534	14.6594	14.0999	13.8037	13.7804	15.2941	14.9808	14.5898	13.7334	13.6288	13.239	-0.375884	-0.437434	0.273314	0	0
2010-04-15 00:00:00	16.1566	89.0683	0.418557	15.6073	15.254	14.802	14.1806	13.8349	13.8041	15.5049	15.1571	14.6985	13.8017	13.6288	13.239	-0.344136	-0.414744	0.411137	0	0
2010-04-16 00:00:00	16.0607	81.7999	0.448515	15.7585	15.4007	14.9219	14.2543	13.8643	13.8265	15.6832	15.3209	14.8044	13.8668	13.6288	13.239	-0.320778	-0.394077	0.532358	0	0
2010-04-19 00:00:00	16.193	84.0451	0.478526	15.9033	15.5447	15.0429	14.3303	13.8952	13.8501	15.8687	15.4837	14.9236	13.9413	13.6288	13.239	-0.297454	-0.372215	0.590327	0	0
2010-04-20 00:00:00	16.3003	85.8187	0.507617	16.0356	15.6821	15.1627	14.4076	13.927	13.8744	16.1159	15.6216	15.0382	14.0136	13.6288	13.239	-0.27476	-0.349213	0.813395	1	0
2010-04-21 00:00:00	16.4536	88.1677	0.535896	16.175	15.8224	15.2856	14.4878	13.9605	13.9001	16.2328	15.7635	15.1316	14.086	13.6288	13.239	-0.251222	-0.332011	0.886861	1	0
2010-04-22 00:00:00	16.7315	91.3968	0.565396	16.3605	15.9877	15.4233	14.5758	13.9972	13.9283	16.3478	15.9264	15.2499	14.161	13.6288	13.239	-0.223557	-0.312918	0.979933	1	0
2010-04-23 00:00:00	16.7699	91.7841	0.593439	16.4969	16.1299	15.5516	14.6619	14.0339	13.9566	16.4897	16.0864	15.3759	14.2329	13.6288	13.239	-0.199187	-0.29271	0.317015	0	0
2010-04-26 00:00:00	16.3099	54.8273	0.610407	16.4346	16.1626	15.6238	14.7265	14.0641	13.98	16.513	16.1909	15.4732	14.2917	13.6288	13.239	-0.190929	-0.281091	-0.323549	0	0
2010-04-27 00:00:00	15.8403	36.2179	0.61066	16.2365	16.104	15.6444	14.7702	14.0876	13.9985	16.421	16.2685	15.5484	14.3432	13.6288	13.239	-0.195406	-0.276271	-0.344298	0	0
2010-04-28 00:00:00	15.8212	35.6035	0.598727	16.0981	16.0526	15.6612	14.8114	14.1106	14.0166	16.2946	16.2637	15.6222	14.3887	13.6288	13.239	-0.197594	-0.276402	-0.590449	0	0
2010-04-29 00:00:00	15.9553	43.9483	0.580556	16.0505	16.0349	15.6892	14.8562	14.135	14.0359	16.1393	16.2436	15.7003	14.4373	13.6288	13.239	-0.194102	-0.277949	-1.13735	2	0
2010-04-30 00:00:00	15.5241	28.8972	0.551226	15.875	15.942	15.6735	14.8824	14.1534	14.0507	15.8902	16.1899	15.7554	14.4713	13.6288	13.239	-0.201906	-0.286088	-1.23455	2	0
2010-05-03 00:00:00	15.6966	39.2936	0.517934	15.8155	15.8974	15.6757	14.9144	14.1738	14.0671	15.7675	16.1403	15.812	14.5271	13.6288	13.239	-0.202037	-0.288102	-3.37149	2	0
2010-05-04 00:00:00	15.0066	22.6989	0.471715	15.5459	15.7354	15.612	14.918	14.1849	14.0764	15.6008	16.0109	15.8163	14.5711	13.6288	13.239	-0.22008	-0.297655	-1.57244	2	0
2010-05-05 00:00:00	15.112	28.4673	0.420423	15.4013	15.6221	15.5644	14.9256	14.1971	14.0868	15.4589	15.8767	15.8201	14.6202	13.6288	13.239	-0.230329	-0.306226	-0.382396	0	0
2010-05-06 00:00:00	14.5658	19.1908	0.358968	15.1228	15.43	15.4693	14.9115	14.202	14.0915	15.181	15.6602	15.7933	14.6549	13.6288	13.239	-0.252696	-0.325276	-0.385429	0	0
2010-05-07 00:00:00	14.3837	16.8963	0.290641	14.8764	15.2398	15.3659	14.8908	14.2044	14.0944	14.9529	15.4215	15.754	14.6879	13.6288	13.239	-0.274656	-0.343838	-0.140396	0	0
2010-05-10 00:00:00	14.7383	35.63	0.226584	14.8304	15.1486	15.3061	14.8848	14.2115	14.1008	14.7613	15.2644	15.7276	14.7289	13.6288	13.239	-0.281882	-0.356857	-0.299825	0	0

Figure 7: Screenshot of contents of ticker dictionary entry of DELL stock

The closing price is the last price at which a security is traded during the regular trading day (Hayes, 2023). Taking DELL on 2010-04-14 in Figure 7 as an example, we can see that the closing price is 15.8691. In the next column, we have the Relative Strength Index (RSI), which is a technical momentum indicator used in technical analysis. The value normally oscillates between 0 and 100, with readings below 30 considered oversold and above 70 considered overbought (Fernando, 2023b). As we can see in the example, the value is 86.1087, which means it is considered overbought if we were to buy.

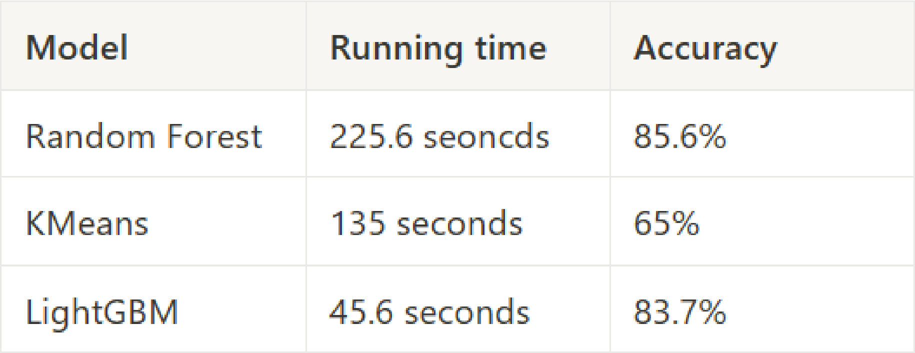
Following RSI, we have MACD (Moving Average Convergence Divergence). It is known as a trend-following momentum indicator that shows the relationship between two moving averages of prices (Fernando, 2023a). The value is calculated by subtracting the 26-period Exponential Moving Average (EMA) from the 12-period EMA (Fernando, 2023a). Hence, MACD is an indicator of trends, with a 5-day window. Next, we have EMA, also known as Exponential Moving Average, which is a technical chart indicator and a type of weighted moving average that gives more weight to recent price data, making it more responsive to recent price changes than a simple moving average (SMA) (Maverick, 2022). For example, a 20-period EMA would be calculated by applying more weight to the most recent 20 periods' worth of price data. As we can see in DELL's example, we created an algorithm that provides us with 5, 10, 20, 50, 150, and 200 trading days as the windows of EMA and SMA in different columns.

SMA calculates the average of a selected range of prices by the number of periods in that range (Hayes, 2019). For example, a 50-day SMA would be calculated by summing the closing prices for the last 50 days and dividing it by 50. As you can see from Dell's output, we applied a similar algorithm to EMA, creating different columns stating SMA_5, SMA_10, SMA_20, SMA_50, SMA_150, and SMA_200, where the numbers represent the trading days.

EMA_PCA and SMA_PCA are the Exponential Moving Average and Simple Moving Average indicators calculated using Principal Component Analysis (PCA), a dimensionality reduction technique. PCA can be applied to extract the most important components or features from a dataset before calculating technical indicators like EMA and SMA. In Dell's example, we can see that EMA_PCA is -0.47913 and SMA_PCA is -0.520452.

Model Output refers to the output from the statistical model mentioned earlier. The remaining two columns are Signal and Predicted Signal. Signal refers to an indication or trigger to enter or exit a position based on certain predefined rules or conditions. A trading signal may be generated when the 50-day SMA crosses above the 200-day SMA, indicating a potential buy signal. In our algorithm, if the number hits 0, it indicates no action should be taken during this period. If the signal hits 1, it indicates a potential entry point, while 2 indicates a potential selling point.

Predicted Signal refers to the signal classified by the machine learning algorithm. A model trained on historical stock data and past trading signals outputs predicted buy or sell signals for future periods. In our case, it follows the signals algorithm: print 0 if no action is required, 1 if it advises investors to buy, and 2 if it's the best time to sell.



Model	Running time	Accuracy
Random Forest	225.6 seoncds	85.6%
KMeans	135 seconds	65%
LightGBM	45.6 seconds	83.7%

Figure 8.1: Table of running time and accuracy of Random Forest, KMeans Clustering and LightGBM models

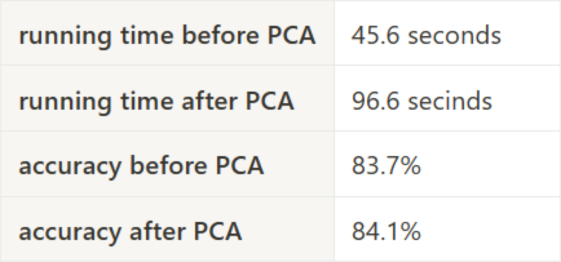
As shown in Figure 8.1, we tried using the Random Forest Classifier. Random Forest operates by constructing multiple decision trees during training and outputting the classification or mean prediction through regression of the individual trees. It is effective as it can capture complex patterns in the data while being resistant to overfitting. After running the Random Forest model, we received a runtime of 225.6 seconds but a higher accuracy of approximately 85.6%. The construction of multiple decision trees and the complexity may cause the long runtime.

Another model we tried was KMeans Clustering, an unsupervised machine learning algorithm used for clustering data points into K distinct clusters. It provided us with a runtime of 135 seconds. Although its runtime is better than the Random Forest model, it provided a much lower accuracy of 65% compared to PCA and the Random Forest model.

Another model we tried is LightGBM, a gradient-boosting algorithm that uses tree-based learning algorithms (Geeksforgeeks, 2024). It is designed to be highly efficient and can handle large-scale data and high-dimensional features. LightGBM is known for its speed and accuracy in classification, regression, ranking, and many other machine-learning competitions. After using the LightGBM algorithm for our dataset, we received an execution time of a stunning 45.6 seconds, which is by far the fastest among all the machine learning algorithms we used. It also provides an accuracy of 83.7%, very similar to the accuracy achieved with PCA.

After trying out various machine learning models to find the best runtime and accuracy, we ended up employing the LightGBM machine learning algorithm as our classification model since it provides the best efficiency in terms of runtime and accuracy. Although the highest accuracy was achieved with the Random Forest algorithm, as shown in Figure 7, the execution time was 225.6 seconds, which is five times longer than the LightGBM algorithm.

To reduce complexity, we applied PCA to all EMA and SMA features. PCA is a statistical technique used for dimensional reduction and feature extraction in machine learning, simplifying a large dataset into a smaller set while still maintaining significant patterns and trends (Jaadi, 2024). After using PCA, the runtime was 120 seconds, and the accuracy of LightGBM increased by 0.4% to 84.1%, as shown in Figure 8.2.



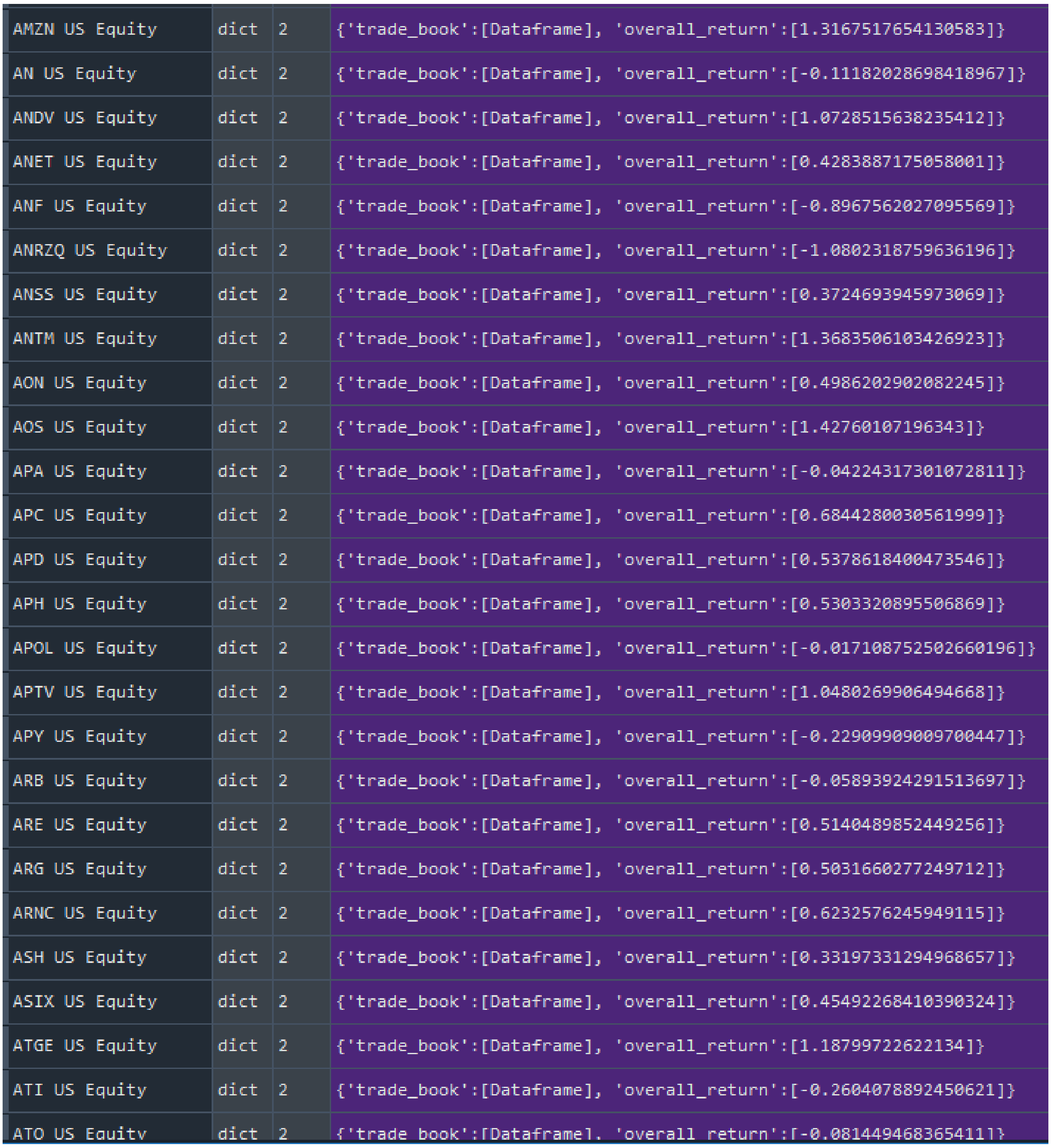
running time before PCA	45.6 seconds
running time after PCA	96.6 secinds
accuracy before PCA	83.7%
accuracy after PCA	84.1%

Figure 8.2: Table of running times and accuracy of LightGBM model before and after applying PCA

Although the accuracy of LightGBM is high, it couldn’t classify the signals at all, given that the majority of the signals are 0.

Performance Analysis

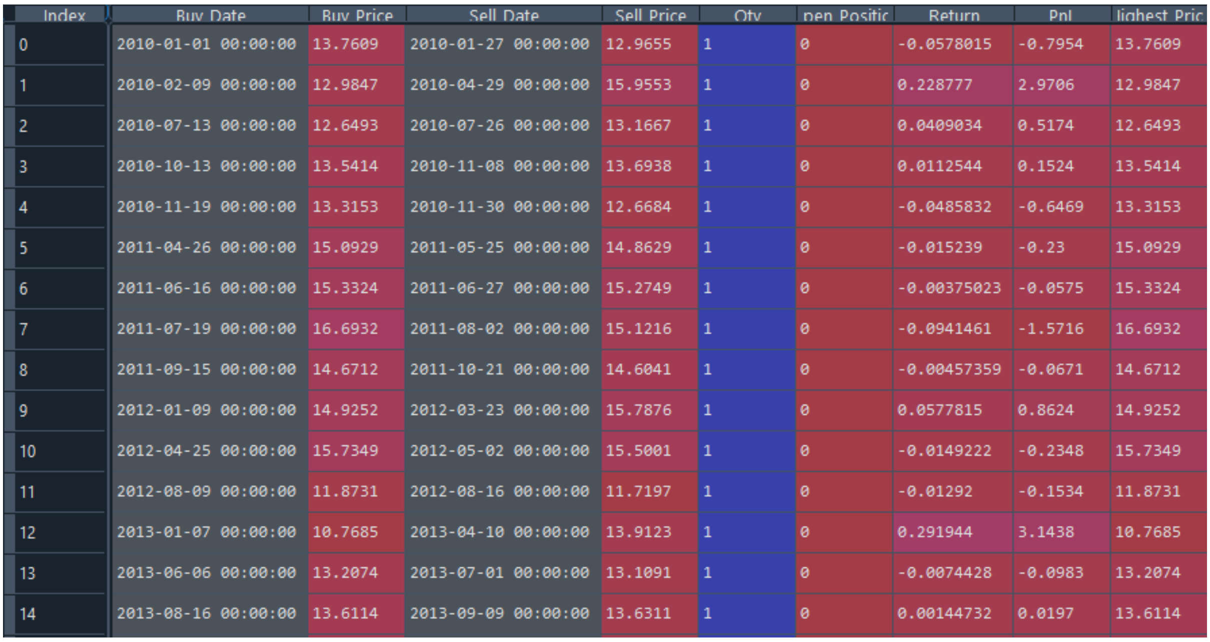
Tradebook



AMZN US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[1.3167517654130583]}
AN US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-0.11182028698418967]}
ANDV US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[1.0728515638235412]}
ANET US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.4283887175058001]}
ANF US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-0.8967562027095569]}
ANRZQ US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-1.0802318759636196]}
ANSS US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.3724693945973069]}
ANTM US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[1.3683506103426923]}
AON US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.4986202902082245]}
AOS US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[1.42760107196343]}
APA US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-0.04224317301072811]}
APC US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.6844280030561999]}
APD US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.5378618400473546]}
APH US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.5303320895506869]}
APOL US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-0.017108752502660196]}
APTV US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[1.0480269906494668]}
APY US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-0.22909909009700447]}
ARB US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-0.05893924291513697]}
ARE US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.5140489852449256]}
ARG US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.5031660277249712]}
ARNC US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.6232576245949115]}
ASH US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.33197331294968657]}
ASIX US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[0.45492268410390324]}
ATGE US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[1.18799722622134]}
ATI US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-0.26040788924506 21]}
ATO US Equity	dict	2	{'trade_book':[Dataframe], 'overall_return':[-0.08144946836541l]}

Figure 9.1: Screenshot of tradebooks generated

Figure 9.1 shows trade books that we have created to store each company's overall return by accumulating each trading day's returns over 10 years. A trade book provides a centralised record of all the stocks/securities an investor currently holds, along with details like purchase prices, quantities, and entry/exit points. This is beneficial as investors can easily monitor the performance of their portfolio, calculate gains/losses, and evaluate the effectiveness of their investment strategies. Additionally, it serves as a valuable resource for investors to review their past trading performance, making it easier to identify patterns and learn from their successes and failures.

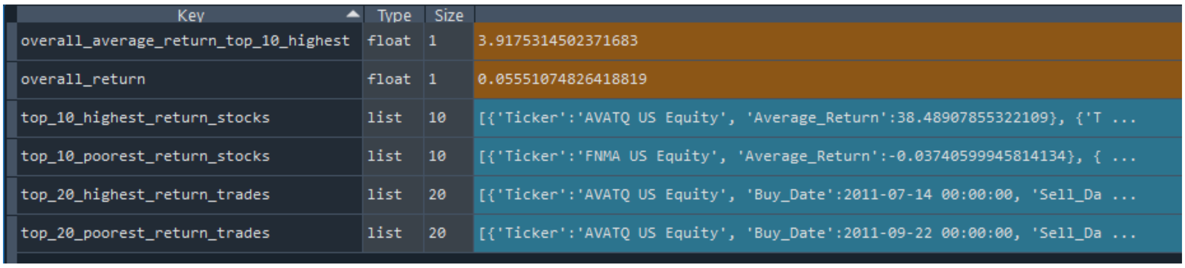


Index	Buy Date	Buy Price	Sell Date	Sell Price	Qty	pen Positic	Return	Pnl	lighest Pric
0	2010-01-01 00:00:00	13.7609	2010-01-27 00:00:00	12.9655	1	0	-0.0578015	-0.7954	13.7609
1	2010-02-09 00:00:00	12.9847	2010-04-29 00:00:00	15.9553	1	0	0.228777	2.9706	12.9847
2	2010-07-13 00:00:00	12.6493	2010-07-26 00:00:00	13.1667	1	0	0.0409034	0.5174	12.6493
3	2010-10-13 00:00:00	13.5414	2010-11-08 00:00:00	13.6938	1	0	0.0112544	0.1524	13.5414
4	2010-11-19 00:00:00	13.3153	2010-11-30 00:00:00	12.6684	1	0	-0.0485832	-0.6469	13.3153
5	2011-04-26 00:00:00	15.0929	2011-05-25 00:00:00	14.8629	1	0	-0.015239	-0.23	15.0929
6	2011-06-16 00:00:00	15.3324	2011-06-27 00:00:00	15.2749	1	0	-0.00375023	-0.0575	15.3324
7	2011-07-19 00:00:00	16.6932	2011-08-02 00:00:00	15.1216	1	0	-0.0941461	-1.5716	16.6932
8	2011-09-15 00:00:00	14.6712	2011-10-21 00:00:00	14.6041	1	0	-0.00457359	-0.0671	14.6712
9	2012-01-09 00:00:00	14.9252	2012-03-23 00:00:00	15.7876	1	0	0.0577815	0.8624	14.9252
10	2012-04-25 00:00:00	15.7349	2012-05-02 00:00:00	15.5001	1	0	-0.0149222	-0.2348	15.7349
11	2012-08-09 00:00:00	11.8731	2012-08-16 00:00:00	11.7197	1	0	-0.01292	-0.1534	11.8731
12	2013-01-07 00:00:00	10.7685	2013-04-10 00:00:00	13.9123	1	0	0.291944	3.1438	10.7685
13	2013-06-06 00:00:00	13.2074	2013-07-01 00:00:00	13.1091	1	0	-0.0074428	-0.0983	13.2074
14	2013-08-16 00:00:00	13.6114	2013-09-09 00:00:00	13.6311	1	0	0.00144732	0.0197	13.6114

Figure 9.2: Screenshot of contents of tradebook of DELL stock

For example, if we click on a stock in the trade book, such as Dell shown in Figure 9.2, the trade book acts as a broker. When the closing price satisfies any buying or selling criteria, it starts or ends a trade. It only initiates another trade if the last trade is closed. The trade book provides information on the buy date and sell date along with the buy price and sell price. To provide insights about every trade, it calculates the return rate in percentage for the transactions. We also implemented a column that indicates the profit and loss of the trade in dollars, showing positive values for profits and negative values for losses.

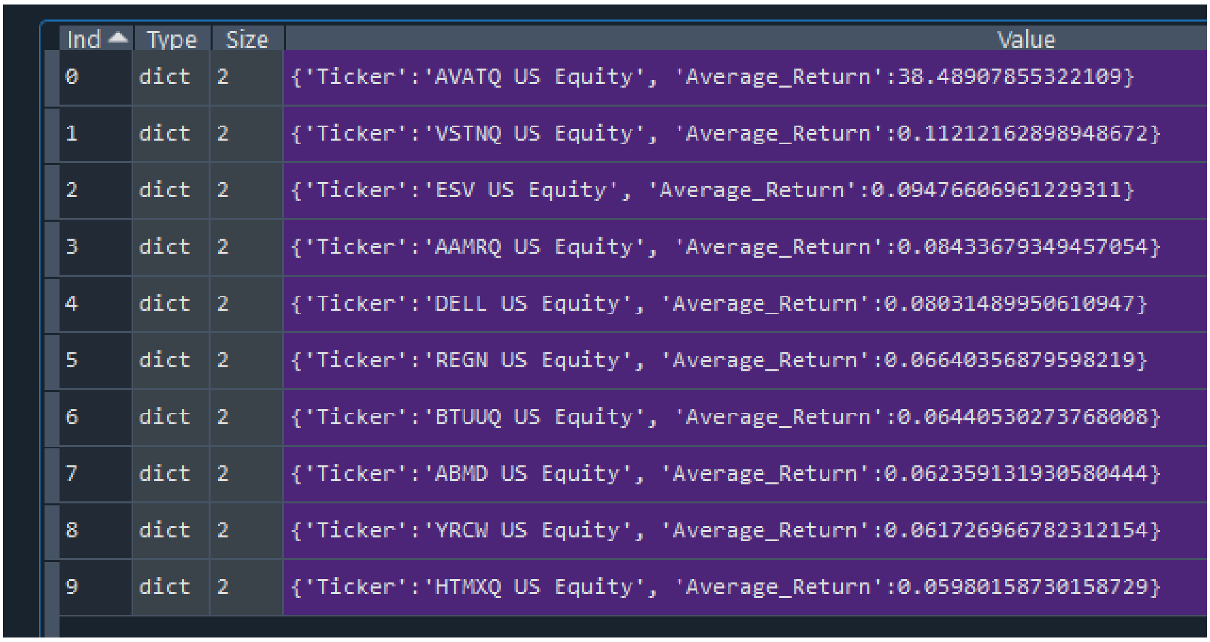
Summary Logbook



Key	Type	Size	
overall_average_return_top_10_highest	float	1	3.9175314502371683
overall_return	float	1	0.05551074826418819
top_10_highest_return_stocks	list	10	[{'Ticker':'AVATQ US Equity', 'Average_Return':38.48907855322109}, {'T ...
top_10_poorest_return_stocks	list	10	[{'Ticker':'FNMA US Equity', 'Average_Return':-0.03740599945814134}, { ...
top_20_highest_return_trades	list	20	[{'Ticker':'AVATQ US Equity', 'Buy_Date':2011-07-14 00:00:00, 'Sell_Da ...
top_20_poorest_return_trades	list	20	[{'Ticker':'AVATQ US Equity', 'Buy_Date':2011-09-22 00:00:00, 'Sell_Da ...

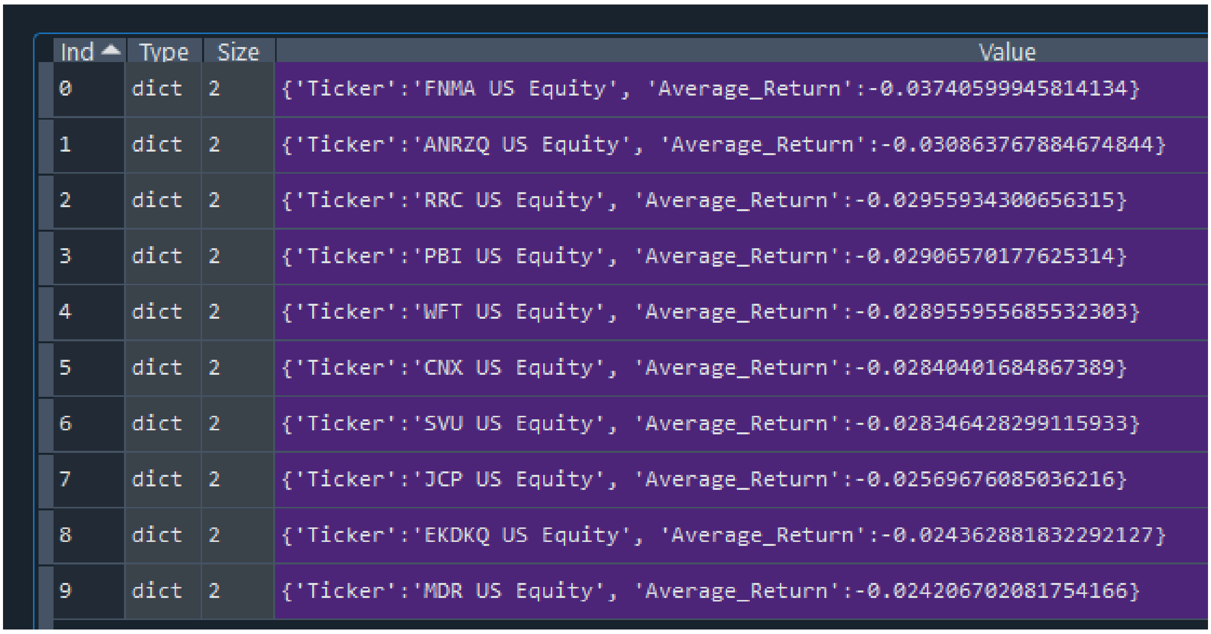
Figure 10.1: Screenshot of contents of Summary Logbook

Figure 10 presents our summarised logbook highlighting the six most important aspects of stock portfolio optimization. The overall average return among all stocks is 5.55%, demonstrating that the statistical model is performing better than expected, as we were aiming for a return of 3%. If we invest in only the top 10 performing stocks, we could achieve an average return of 392%. The logbook also identifies the most and least worthwhile stocks to invest in over the 10-year period, as well as the 10 best and worst-performing stocks (See Figures 10.2 and 10.3). We found that the highest return stock is ‘AVATQ US Equity’ with an average return of 38.489 but ‘AVATQ US Equity’ also has the lowest return compared to all the other companies in the S&P 500 (See Figures 10.4 and 10.5). Research did not show any real record of this stock.



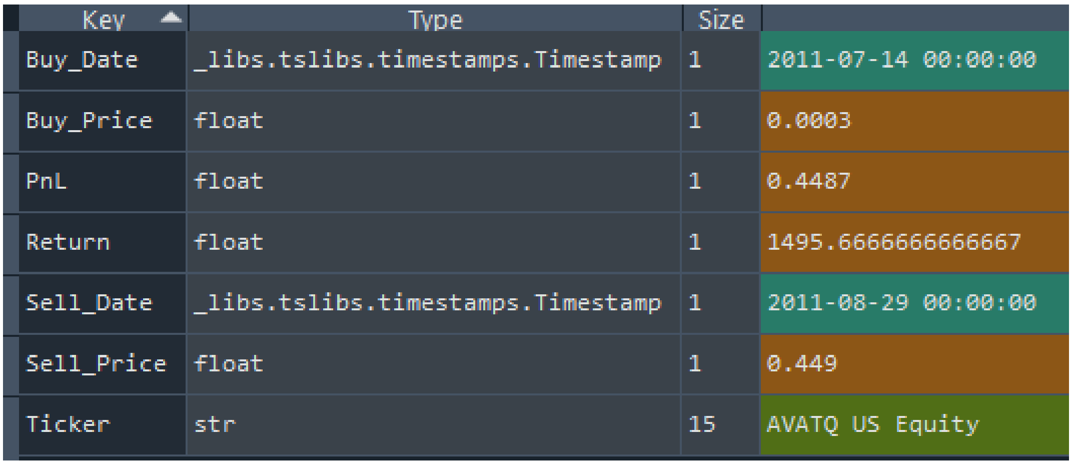
Ind ▲	Type	Size	Value
0	dict	2	{'Ticker':'AVATQ US Equity', 'Average_Return':38.48907855322109}
1	dict	2	{'Ticker':'VSTNQ US Equity', 'Average_Return':0.11212162898948672}
2	dict	2	{'Ticker':'ESV US Equity', 'Average_Return':0.09476606961229311}
3	dict	2	{'Ticker':'AAMRQ US Equity', 'Average_Return':0.08433679349457054}
4	dict	2	{'Ticker':'DELL US Equity', 'Average_Return':0.08031489950610947}
5	dict	2	{'Ticker':'REGN US Equity', 'Average_Return':0.06640356879598219}
6	dict	2	{'Ticker':'BTUUQ US Equity', 'Average_Return':0.06440530273768008}
7	dict	2	{'Ticker':'ABMD US Equity', 'Average_Return':0.06235913193058044}
8	dict	2	{'Ticker':'YRCW US Equity', 'Average_Return':0.06172696678231215}
9	dict	2	{'Ticker':'HTMXQ US Equity', 'Average_Return':0.05980158730158729}

Figure 10.2: Screenshot showing the top 10 stocks with the highest return



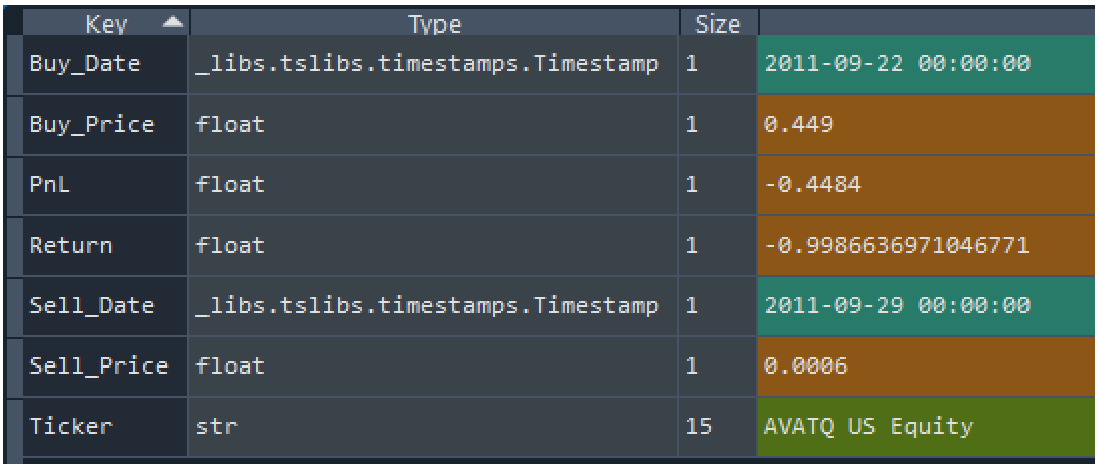
Ind	Type	Size	Value
0	dict	2	{'Ticker':'FNMA US Equity', 'Average_Return':-0.03740599945814134}
1	dict	2	{'Ticker':'ANRZQ US Equity', 'Average_Return':-0.030863767884674844}
2	dict	2	{'Ticker':'RRC US Equity', 'Average_Return':-0.02955934300656315}
3	dict	2	{'Ticker':'PBI US Equity', 'Average_Return':-0.02906570177625314}
4	dict	2	{'Ticker':'WFT US Equity', 'Average_Return':-0.028955955685532303}
5	dict	2	{'Ticker':'CNX US Equity', 'Average_Return':-0.02840401684867389}
6	dict	2	{'Ticker':'SVU US Equity', 'Average_Return':-0.028346428299115933}
7	dict	2	{'Ticker':'JCP US Equity', 'Average_Return':-0.02569676085036216}
8	dict	2	{'Ticker':'EKDKQ US Equity', 'Average_Return':-0.024362881832292127}
9	dict	2	{'Ticker':'MDR US Equity', 'Average_Return':-0.024206702081754166}

Figure 10.3: Screenshot showing the top 10 stocks with the lowest return



Key	Type	Size	
Buy_Date	_libs.tslibs.timestamps.Timestamp	1	2011-07-14 00:00:00
Buy_Price	float	1	0.0003
PnL	float	1	0.4487
Return	float	1	1495.6666666666667
Sell_Date	_libs.tslibs.timestamps.Timestamp	1	2011-08-29 00:00:00
Sell_Price	float	1	0.449
Ticker	str	15	AVATQ US Equity

Figure 10.4: Screenshot of details of trade with the highest return



Key	Type	Size	
Buy_Date	_libs.tslibs.timestamps.Timestamp	1	2011-09-22 00:00:00
Buy_Price	float	1	0.449
PnL	float	1	-0.4484
Return	float	1	-0.9986636971046771
Sell_Date	_libs.tslibs.timestamps.Timestamp	1	2011-09-29 00:00:00
Sell_Price	float	1	0.0006
Ticker	str	15	AVATQ US Equity

Figure 10.5: Screenshot of details of trade with the lowest return

Conclusion

This project aimed to optimise a stock portfolio by developing and implementing a trading strategy using a dataset spanning from 2010 to 2019. The goal was to achieve a 3% profit on each trade while navigating the complexities of the stock market.

Our approach began with data preprocessing, addressing issues such as missing values and duplications to ensure high data quality. By focusing on the period from 2010 to 2019, we excluded significant anomalies such as the economic recession and COVID-19, providing a stable dataset for our analysis. We then developed a statistical model to generate buy and sell signals based on our historical stock data. This model incorporated essential technical indicators such as RSI, MACD, EMA, and SMA to enhance the accuracy of our trading decisions. While we experimented with multiple machine learning models, we ultimately chose LightGBM for its optimal balance of accuracy (84.1%) and execution time (96.6 seconds).

In conclusion, our trading strategy yielded a positive average return of 3.91753% across the top 10 highest trading stocks, as shown in the summary logbook. This was achieved by using a combination of SMA and EMA, alongside their PCA-reduced versions, to smooth out price data and reduce noise. Moreover, the trade book provided a centralised record of all trades, allowing for easy portfolio performance monitoring, calculating gains/losses, and evaluating investment strategies. Each stock's trade details, such as buy and sell dates and prices, were recorded to assist in pattern recognition and strategy refinement. These results highlight the significant potential of combining traditional financial analysis with modern machine learning techniques, offering a more comprehensive view of the stock market for investors that can ultimately aid in decision-making.

References

CFI Team. (2024). Modern Portfolio Theory(MPT). https://corporatefinanceinstitute.com/resources/career-map/sell-side/capital-markets/modern-portfolio-theory-mpt/#:~:text=The%20Modern%20Portfolio%20Theory%20(MPT)%20refers%20to%20an%20investment%20theory,prefer%20the%20less%20risky%20portfolio.

Duggan, W. (2023). *A History Of U.S. Bull Markets, 1957 to 2022 – Forbes Advisor*. www.forbes.com. https://www.forbes.com/advisor/investing/bull-market-history/

Fernando, J. (2023a). *Moving Average Convergence Divergence – MACD Definition*. Investopedia. https://www.investopedia.com/terms/m/macd.asp

Fernando, J. (2023b). *Relative Strength Index – RSI*. Investopedia. https://www.investopedia.com/terms/r/rsi.asp

GeeksforGeeks. (2024). LightGBM (Light Gradient Boosting Machine). https://www.geeksforgeeks.org/lightgbm-light-gradient-boosting-machine/

Hayes, A. (2019). *Simple Moving Average - SMA*. Investopedia. https://www.investopedia.com/terms/s/sma.asp

Hayes, A. (2023). What Is Closing Price? Definition, How It's Used, and Example.

Investopedia. https://www.investopedia.com/terms/c/closingprice.asp#:~:text=Key%20Takeaways

Kenton, W. (2021). *S&P 500 Index: What It's for and Why It's Important in Investing*. Investopedia. https://www.investopedia.com/terms/s/sp500.asp

Matthew Frankel. (2024). What is the S&P 500 Index? https://www.fool.com/investing/stock-market/indexes/sp-500/

Maverick, J. B. (2022). *How Is the Exponential Moving Average (EMA) Formula Calculated?* Investopedia. https://www.investopedia.com/ask/answers/122314/what-exponential-moving-average-ema-formula-and-how-ema-calculated.asp#:~:text=The%20exponential%20moving%20average%20(EMA)%20is%20a%20technical%20chart%20indicator

McClure, B. (2020). *What Beta Means When Considering a Stock's Risk*. Investopedia. https://www.investopedia.com/investing/beta-know-risk/#:~:text=Beta%20is%20a%20measure%20of

Jaadi, Z. (2024). A Step-by-Step Explanation of Principal Component Analysis (PCA). https://builtin.com/data-science/step-step-explanation-principal-component-analysis