Optimal Portfolio Creation Comparison and Combination of Different Techniques for Improvement

1st Ömer Faruk Merey
Computer Engineering
TOBB University of Economics and Technology
Ankara, Turkey
o.merey@etu.edu.tr

2th Kerem İhsan Ulaşan
Computer Engineering
TOBB University of Economics and Technology
Ankara, Turkey
kulasan@etu.edu.tr

***Abstract*—This document presents a comprehensive study on optimal portfolio creation based on the stock market using various techniques, including stock price prediction models, deep learning models, genetic algorithms and Markov approach integrated with Yahoo Finance API. The aim is to create an optimal portfolio with risk and return analysis help of combining and comparing different methodologies.**

***Index Terms*—Optimal Portfolio, Stock Price Prediction, Machine Learning, Deep Learning, RNN, CNN, LSTM, XGBoost, Markowitz, Genetic Algorithms, Yahoo Finance**

I. Introduction

Creating an optimal portfolio is a quintessential task in finance, involving the selection of a combination of financial investments that collectively meet a particular set of objectives better than any other combination. These objectives typically balance the desire for maximum returns against the tolerance for risk, under constraints of budget, legality, and market conditions. As financial markets have evolved, so too have the methodologies for creating optimal portfolios, leading from traditional Markowitz mean-variance optimization to more complex techniques that incorporate advances in computing power, statistical methods, and financial theory.

Incorporation of stock price prediction into the process of creation optimal portfolios represents significant advancement. The advancement of stock price prediction techniques, fueled by breakthroughs in machine learning and data analytics, has provided investors with tools to anticipate future market movements more accurately. The integration, not only enhances the potential for improved returns but also introduces a new layer of complexity and opportunity in the design and evaluation of investment strategies. Our work also expands to examine how stock price predictions influence portfolio optimization techniques efficacy and applicability.

II. Methodologies

In the last few years, we have noticed that the stock markets have been a crucial place for investors and people who have dreams of gaining money. These investors are not always that experienced about the market or the properties of an asset. Our motivation for the project is to create a portfolio with the minimum risk and maximum return. We picked some of the top companies from S&P 500 and analyzed them with effective explanatory analysis.

For the stock price prediction models, we created features and new parameters to find the impact on price predictions. These parameters, that we will talking about later in the paper, were selected on traditional aspects and our price beliefs. We then created models and fine-tuned them with the our train sets. In our models, we aimed to create the most effective model based on experiments on parameters.

Markowitz is an investment theory developed in the 1950s by Harry Markowitz, the founder of Modern Portfolio Theory (MPT). At its core, it explains how investors should allocate their assets to maximize expected return while minimizing risk. This theory quantitatively assesses the relationship between risk and return and demonstrates how investment portfolio diversification can manage risk. The risk of a portfolio depends not only on the individual risks of the assets included but also on their correlation with each other. Investors can reduce portfolio risk by combining assets with low or negative correlations. The Markowitz theory is a powerful tool that can be used when planning and implementing investment strategies. This is why it is preferred as a benchmark

Genetic algorithms primarily operate within a specific population and lineage to search for a global maximum, distinct from the Markowitz approach which does not directly consider the correlation between portfolios.

Fundamentally, there are three different methods involved: elitism, crossover, and a simplified version of both, aiming to return a value that we strive to maximize. Through generations, we start to learn and understand better. Mutations help us escape from local maxima to reach global maxima, as these variations prevent us from narrow thinking. On the other hand, elitism directly transfers the best genes from parent to offspring, ensuring the perpetuation of good genes. This allows for more confident and rapid progress in known directions. Genetic algorithms are one of the most critical tools for any optimization problem, facilitating a comprehensive approach to finding the best solutions

III. Dataset

A. Data Collection

We collected our data from Yahoo Finance Python API. Focussing on the price feature, we took adjusted close price on the account.

B. Preprocessing and Split

1) **Minimum and Maximum Scaler:** Its purpose is to scale all features in the dataset to a specified range, usually between 0 and 1. This ensures that all features are on the same scale.
2) **Split:** We split our data in a common way for all stocks. We used 2010-01-01 as a start point and trained our models to 2020-01-01 to analyse how our work is successful.

C. Feature Engineering

1) **Moving Average:** Moving Averages Percentage adds the average change value of N days to each row in the dataset. It can be used to smooth out short-term fluctuations and highlight longer-term trends in the data. The formula for the Moving Averages Percentage is given by:

$$\text{Moving Average}_N = \frac{1}{N}\sum_{i=0}^{N-1} X_{t-i} \qquad (1)$$

2) **Volatility:** Volatility represents the degree of variation of a trading price series over a certain period of time. It is often measured by the standard deviation of the returns. Adding the average standard deviation value of N days to each row in the dataset can be useful for understanding price dynamics.

$$\text{Volatility}_N = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(r_i - \overline{r})^2} \qquad (2)$$

3) **Bollinger Bands:** Used to analyze the volatility and potential price movements of a financial asset. They consist of three lines: the middle band, the upper band, and the lower band. The middle band is typically a Simple Moving Average (SMA) calculated over a specific period. The upper band is calculated by adding a multiple (usually 2 or 2.5) of the standard deviation of the price series to the middle band. Similarly, the lower band is calculated by subtracting a multiple of the standard deviation from the middle band.
4) **Moving Average Convergence Divergence:** Used to generate trading signals. MACD provides information about the momentum and trend direction, these play a role in indicating the trading points.
5) **Relative Strength Index (RSI):** Used in technical analysis to assess the strength and speed of price movements in a financial asset. RSI is typically calculated using the average gain and average loss over a specified period, often 14 periods by default. Traditionally, RSI readings above 70 are considered overbought, indicating that the asset may be due for a downward correction, while readings below 30 are considered oversold, suggesting a potential upward reversal.
6) **The Ichimoku Cloud:** Versatile technical analysis indicator that provides insights into the direction, momentum, and support/resistance levels of a financial asset. The technique was experimented but was not useful on the model and feature selection/importance.

D. Feature Importance and Selection

At the selection point of our preparation, we analyzed the created features, created correlation matrix to determine which features are necessary and also used feature importance to indicate which features have more impact on the adjusted close.

As you can see from the correlation matrix for one of our stocks. For price prediction models, our features was better at less. After our work we reduced the features and computed feature importance. (See example figures Fig. 1, 2 and 3.)

Another important aspect to price prediction models in our feature selection was that different stocks were correlated differently to the features we created. There were not dramatic changes but the difference really shows that we should not generalize stock price prediction models. As we will see in the upcoming parts in our paper, some models will do better on some stocks which also gives us a peak at this stage of our work. This also gives clue about the portfolio creation that some of the stocks may be more selective from some models.

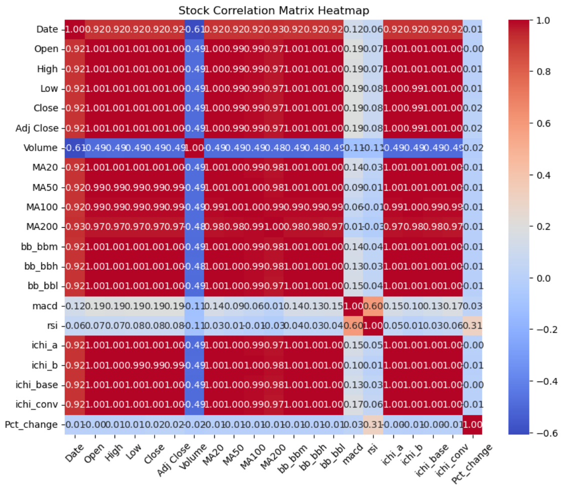

Fig. 1. Correlation of all features

IV. Evaluation

1) **Mean Absolute Error (MAE):** It is the average of the absolute differences between the predicted and actual

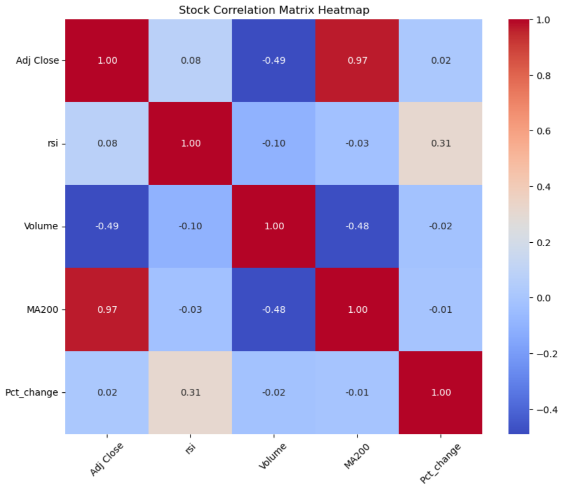

Fig. 2. Correlation after future selection

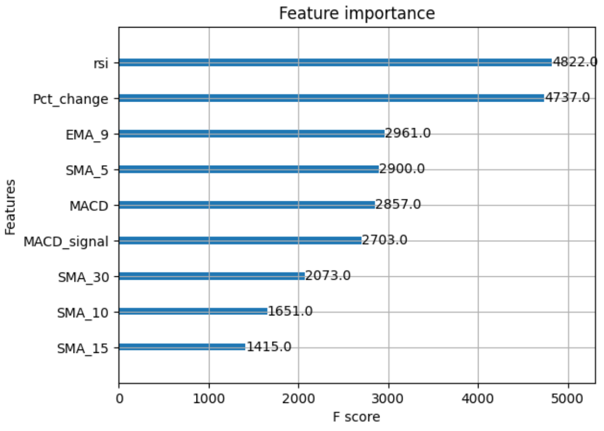

Fig. 3. Feature importance after future selection for XGBoost Model

values. The formula for MAE is:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3}$$

2) **Mean Absolute Error (MAE):**It is the average of the squared differences between the predicted and actual values. The formula for MSE is:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4}$$

3) **Rolling Volatility:** Statistical measure used in finance to assess the variability of returns for a financial asset over a specific time period. Instead of calculating the volatility over the entire historical dataset, rolling volatility computes volatility over a moving window of data.
4) **Rolling Sharpe:** Dynamic measure of risk-adjusted return calculated over a rolling period of time. It is an extension of the traditional Sharpe ratio, which evaluates the performance of an investment by comparing its average return to its volatility.
5) **Rolling Sortino:** Dynamic measure of risk-adjusted return similar to the rolling Sharpe ratio but using downside deviation instead of total volatility.

The evaluation metrics MAE and MSE were mostly used for stock price prediction model. There were not the only metrics used but they did play a big role when we were picking which model to choose for which stock. (See Table IV)

We used daily, weekly, monthly, yearly returns to analyze our portfolio throughout the process. (See Fig 4.) We used 6 month as the time period for rolling metrics.

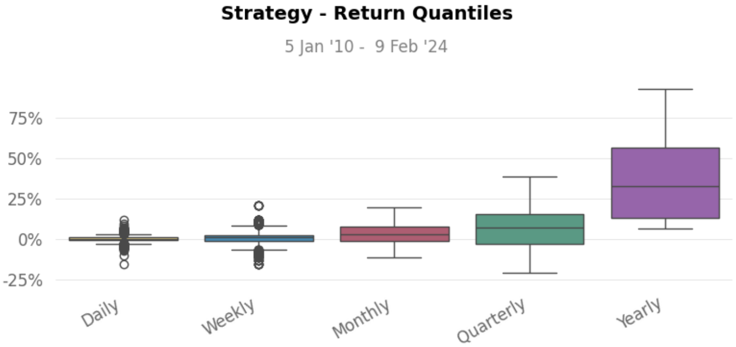

Fig. 4. Time Periods

In our evaluations, there were other metrics such as monthly active returns, drawdown periods etc. to analyze and compare between our models. (See Fig. 5)

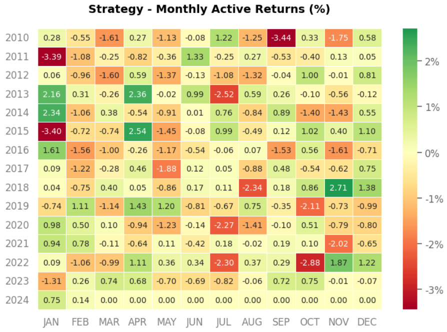

Fig. 5. Monthly Active Returns

V. Models

A. Stock Price Prediction

Our main focus was to predict the upcoming stock prices and using the predictions we can create a weighted portfolio. This although seemed like a good idea, it had downfalls. Seasonal speculations of the prices, had a dramatic effect on our predictions. (See Fig. 7)

As we will mention in the evaluation section, the evaluation metrics was another problem when the prices were having significant changes. However, even though modifying the changes seemed a reasonable task we were not receiving the optimal weights thus possibly resulting in a worse portfolio. (See Table I)

Model Name	MSE	MAE
CNN-LSTM	0.036	0.081
LSTM	0.045	0.095
RNN	0.92	0.41
XGBoost	0.037	0.97

TABLE I
Performance of Stock Prediction Models (Avg)

Name	Score
MSE	0.036
MAE	0.081
Variance	0.942612
R2 Score	0.9521
Max Error	0.182930

TABLE II
Performance of CNN-LSTM Model (Avg)

One of the and most experimented deep learning model was with combination of CNN layers along with LSTM layers. The results was very good and promising for the model as expected. The power of CNN layers was also once again was proven within our work.

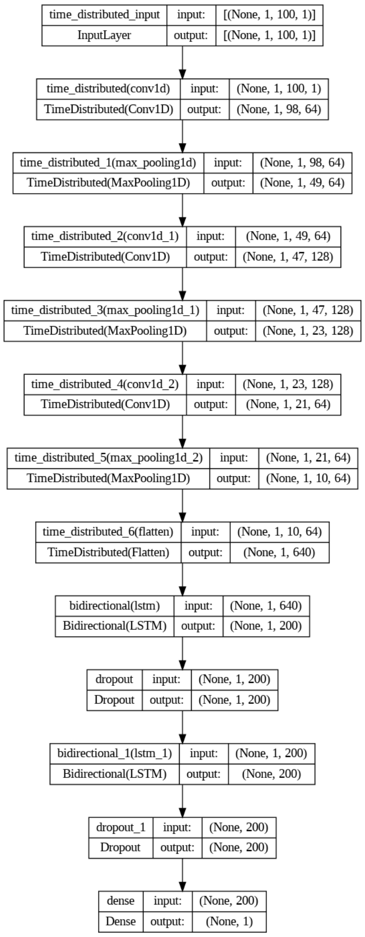
time_distributed_input | input: | [(None, 1, 100, 1)]
InputLayer | output: | [(None, 1, 100, 1)]
time_distributed(conv1d) | input: | (None, 1, 100, 1)
TimeDistributed(Conv1D) | output: | (None, 1, 98, 64)
time_distributed_1(max_pooling1d) | input: | (None, 1, 98, 64)
TimeDistributed(MaxPooling1D) | output: | (None, 1, 49, 64)
time_distributed_2(conv1d_1) | input: | (None, 1, 49, 64)
TimeDistributed(Conv1D) | output: | (None, 1, 47, 128)
time_distributed_3(max_pooling1d_1) | input: | (None, 1, 47, 128)
TimeDistributed(MaxPooling1D) | output: | (None, 1, 23, 128)
time_distributed_4(conv1d_2) | input: | (None, 1, 23, 128)
TimeDistributed(Conv1D) | output: | (None, 1, 21, 64)
time_distributed_5(max_pooling1d_2) | input: | (None, 1, 21, 64)
TimeDistributed(MaxPooling1D) | output: | (None, 1, 10, 64)
time_distributed_6(flatten) | input: | (None, 1, 10, 64)
TimeDistributed(Flatten) | output: | (None, 1, 640)
bidirectional(lstm) | input: | (None, 1, 640)
Bidirectional(LSTM) | output: | (None, 1, 200)
dropout | input: | (None, 1, 200)
Dropout | output: | (None, 1, 200)
bidirectional_1(lstm_1) | input: | (None, 1, 200)
Bidirectional(LSTM) | output: | (None, 200)
dropout_1 | input: | (None, 200)
Dropout | output: | (None, 200)
dense | input: | (None, 200)
Dense | output: | (None, 1)

Fig. 6. CNN-LSTM Architecture

Recurrent Neural Networks (RNNs) on the other hand, was very prone to overfitting. We often had to change the epochs and change the dropout rate for different stocks. It mostly resulted in an overfit position but for some iterations the results were not bad. Suprisingly, out ouf every stock and model combinations, RNNs were never the best model to choose.

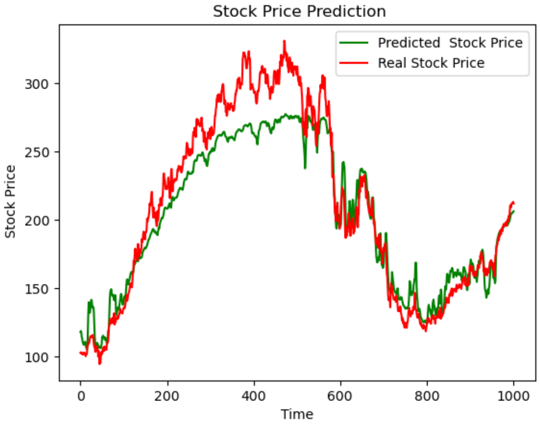

Fig. 7. Prediction comparison on dates with seasonal specs

B. Markowitz

In our study, we implemented a simple version of Markowitz's portfolio optimization using Python. For this implementation, we utilized the PyPortfolioOpt library. These tools helped us to apply the principles of modern portfolio theory in our analysis.

$$\begin{aligned} &P : \text{Lower bound on the portfolio return} \\ &\mu_j : \text{Mean return for security } j \\ \text{Minimize Var} : &\sum_{j=1}^{n} v_j x_j^2 + 2 \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} q_{ij} x_i x_j = x^T Q x \\ \text{Subject to:} & \\ &\sum_{j=1}^{n} x_j = 1 \\ &\sum_{j=1}^{n} \mu_j x_j \geq P \\ &0 \leq x_j \leq 1 \text{ for } j = 1 \ldots n \end{aligned} \tag{5}$$

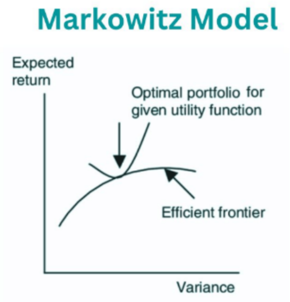

Fig. 8. Expected Return - Variance Graph

Shares	Markowitz
TSLA	0.10011
AAPL	0.21433
GOOG	-
AMZN	0.06883
MSFT	-
META	-
NVDA	0.21118
INTC	-
BRK-B	-
UNH	0.40554

TABLE III
Portfolio Rate of Markowitz Model

C. Genetic Algortihms

In our study, we have explored three distinct models through the application of genetic algorithms:

Normal Model: The first model is a straightforward one, possessing a specific mutation rate. It serves as the base model, characterized by its simplicity and a set mutation ratio that governs the genetic variation introduced during the simulation process.

Increased Mutation and Crossover Model: The second model features an increased mutation rate and employs a more extensive crossover mechanism. Unlike the standard approach where two individuals are crossed, this model utilizes six individuals for the crossover process. Despite its complexity, it can still be described in simple terms. The mutation rate has been elevated to enhance genetic diversity and exploration capabilities within the population.

Elitism Model: For the third model, elitism has been implemented. This approach selects certain individuals within the population as 'elite', and these elite individuals receive the best genes directly from their parents. This method aims for elite members to reach local maxima within their genetic landscape, with the hope that one of them will represent the global maximum. While increased crossover can facilitate easier access to the global maximum by exploring a broader genetic space, the presence of noise makes it challenging to directly achieve the global maximum without a more nuanced approach

Models	population	gen	mutation	elit chromosome
normal	1000	90	0.10	0
normal2	200	50	0.05	0
mutation	1000	90	0.20	0
mutation2	200	50	0.25	0
elite	1000	90	0.01	50
elite2	200	50	0.10	20

TABLE IV
Parameters of Genetic Algorithm Models

D. Combination of Models (Our Approach)

At the end of our work we decided to go beyond comparing our models and may be come up with a better one. We thought of a scenario where an investor let's say, wants to invest in some of the stocks but wants to get the most out of it. So we trained our models stock price prediction models to 2020-01-01 and predicted the future prices from 2020-01-01 to 2024-01-01. Then we concatenated the predicted prices and append it to the data frame.

Shares	normal	normal2	elite	elite2	mutation	mutation2
TSLA	0.1035	0.092	0.105	0.128	0.129	0.135
AAPL	0.1153	0.126	0.117	0.137	0.155	0.129
GOOG	0.003	0.0	0.0	0.001	0.007	0.004
AMZN	0.081	0.062	0.090	0.099	0.111	0.101
MSFT	0.1515	0.157	0.170	0.184	0.197	0.036
META	0.022	0.014	0.030	0.026	0.027	0.036
NVDA	0.172	0.186	0.189	0.204	0.203	0.213
INTC	0.0007	0.0	0.0	0.004	0.002	0.0
BRK-B	0.001	0.0	0.001	0.007	0.0001	0.0001
UNH	0.348	0.349	0.402	0.385	0.446	0.426

TABLE V
Portfolio Rate of Genetic Algorithm Models

We picked our outputs from all of the price prediction models and compared them for better results. We mostly got results from XGBoost and CNN-LSTM models.

Stock	Best Future Price Predictor
APPLE	XGBOOST
TESLA	CNN-LSTM
GOOGLE	XGBOOST
AMAZON	XGBOOST
MICROSOFT	CNN-LSTM
META	CNN-LSTM
NVIDIA	XGBOOST
INTC	RNN
BRK-B	XGBOOST
UNH	RNN

TABLE VI
Which stock was predicted best

We then used our newly generated data frame and retrained new genetic algorithms to see if we can beat the Markowitz. We trained 6 different genetic algorithm and beat the Markowitz in two of our models by not only returns but also with the volatility of our portfolio! (See Fig 9. and 10.)

Our approach (mutation and elitism2) beat the Markowitz model because of how it is constructed. Especially the idea behind elitism, where the parents may be in global max and this way we may have best portfolio constructed in our hands. Also where we mutate, we may have come across the global max.

The Markowitz model was not the only beaten model, we also compared it with our plain genetic algorithms and we came across %40 increase in total (cumulative) return and gave a lower volatile (risk) portfolio at the end.

VI. Acknowledgement

In modern days, an investor who wants to create the optimal portfolio for his/hers investment should use the help of these methods. Our study has shown that in order to create the optimal portfolio or weights for a given subset of assets we should not only rely on the models but also use stock prediction models and combine them to create an extra efficient with more optimized returns and reduced risks. This paper

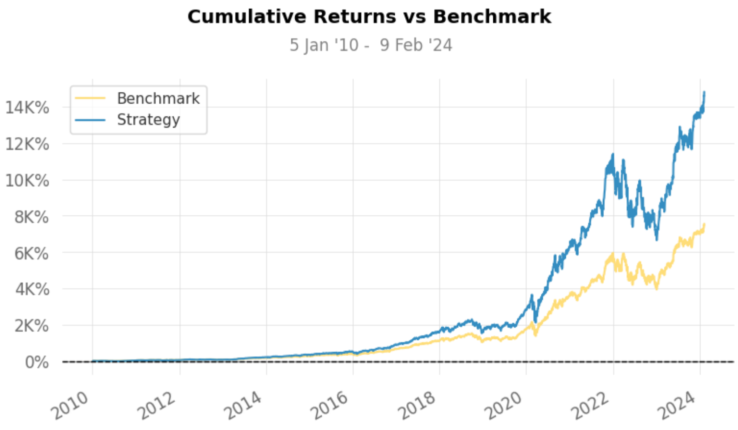

Fig. 9. Price Prediction + Genetic Algorithm vs Markowitz

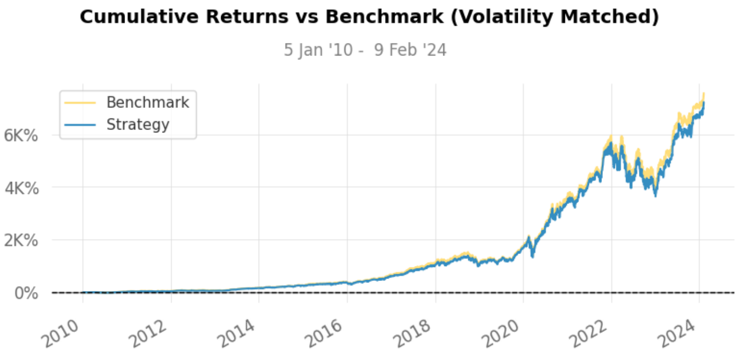

Fig. 10. Price Prediction + Genetic Algorithm vs Markowitz

indicates that, without the knowledge of the real future prices, we can construct models to predict the futures and rely on these futures to create effective portfolios.

VII. Experiments

Our experiments and work are publicly avaliable at here

References

[1] https://finance.yahoo.com/

[2] https://iaeme.com/MasterAdmin/Journal _uploads/IJCET/VOLUME_10_ISSUE_3/IJCET_10_03_003.pdf

[3] https://ieeexplore.ieee.org/document/9752248

[4] https://arxiv.org/pdf/2305.14378.pdf

[5] https://www.nature.com/articles/s41599-024-02807-x

[6] https://towardsdatascience.com/predicting-stock-prices-using-a-keras-lstm-model-4225457f0233

[7] https://medium.com/@sohelrana.aiubPro/mastering-stock-price-prediction-using-deep-learning-models-a-comprehensive-guide-8884df010030

[8] Gurav, U., & Kotrappa, D. S. (2020). Impact of COVID-19 on stock market performance using efficient and predictive LBL-LSTM based mathematical model. International Journal on Emerging Technologies11 (4), 108-115.

[9] https://github.com/ranaroussi/quantstats

[10] https://medium.com/latinxinai/portfolio-optimization-the-markowitz-mean-variance-model-c07a80056b8a