Enhanced Portfolio Selection through Ensemble Predictive Modeling

Project-III (MA57301)

Under the Supervision of Professor Debjani Chakraborty

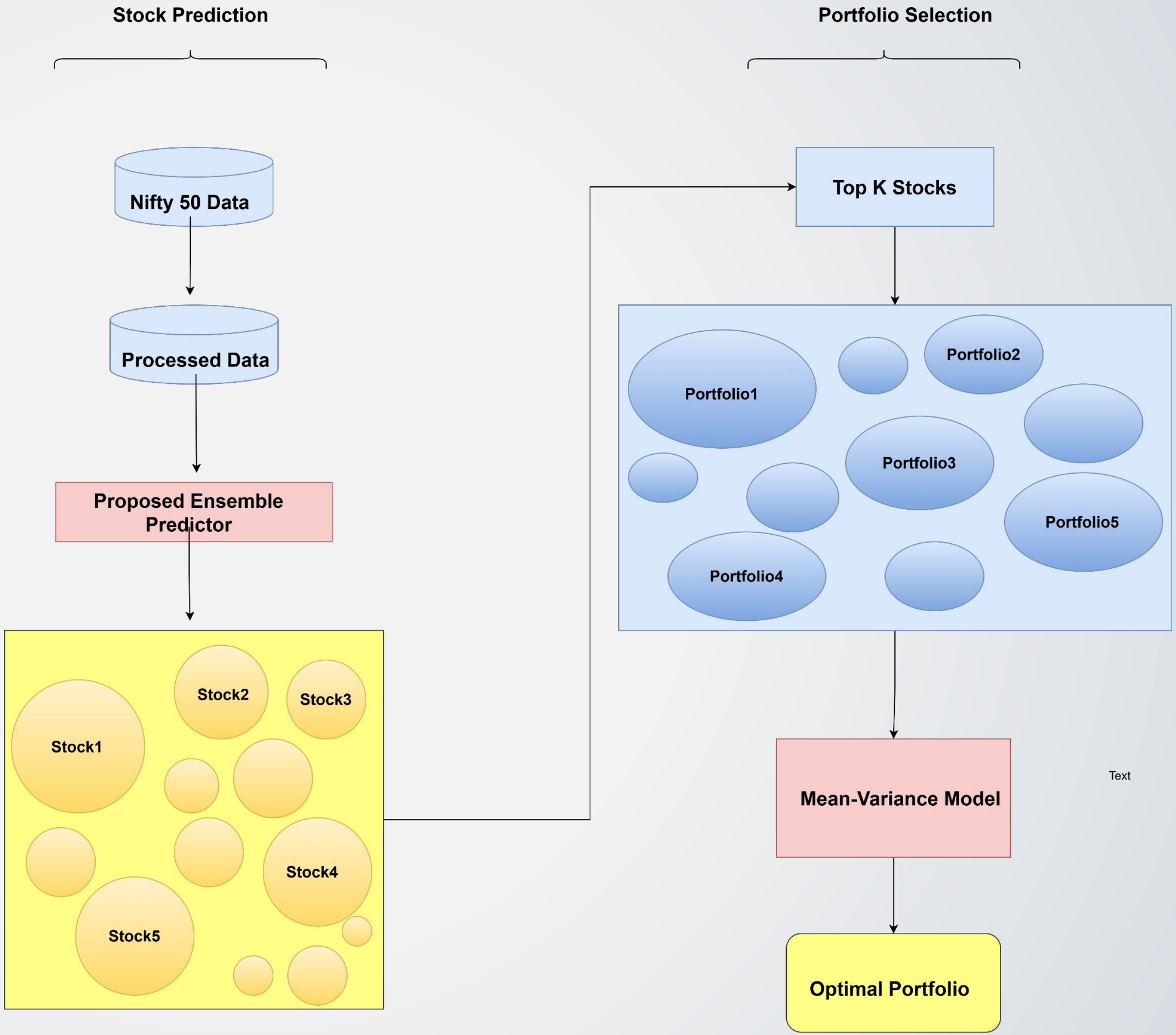
Atharv Bajaj
20MA20014
13/11/2024

Problem Definition

When investing available capital in the stock market, the primary challenges are:

- Which Stocks should we invest in?
- How should we allocate our money across these selected stocks?

Proposed Method



Prediction Methodology

Objective:

To forecast future returns of selected stocks, enabling the identification of top-performing stocks for portfolio optimization.

Existing Models for Stock Return Prediction:

Various Models Exist: like Linear Regression, Time Series Models (e.g., ARIMA), and Machine Learning Models (e.g., Random Forest, SVM) have been commonly used to predict stock returns. However, each comes with its own limitations in handling market complexities and capturing non-linear patterns.

- What if we could harness the collective power of multiple models to achieve a heightened level of accuracy?

- While current models exhibit acceptable performance levels, there's undeniable potential for enhancement. How might we push these boundaries further?

Ensemble Learning

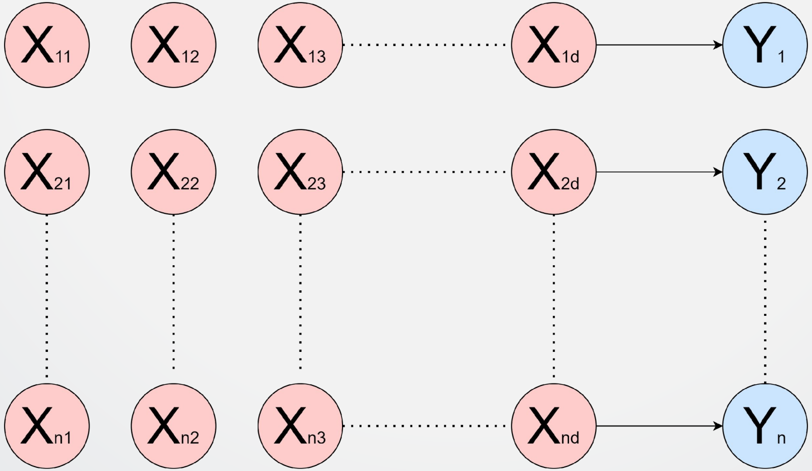
Constructing an ensemble system involves two primary challenges:

- learning a diverse collection of base models
- devising an effective technique for combining their outputs.

Generation of a diverse set of models can either be done :

- Using single base learning model with different parameters
- Using varied set of models

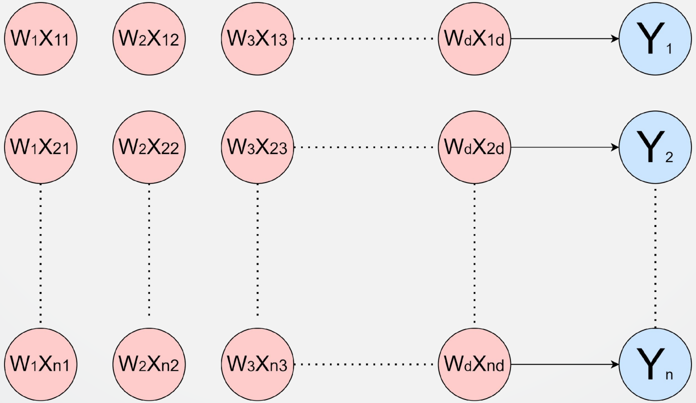
Let X denote the feature space and Y represent the label space, where each x $\in$ X is associated with a label y $\in$ Y . Consider a training dataset comprising a finite sequence of examples, S = {(x1, y1), . . . , (xN , yN)}, where yi denotes the prediction corresponding to the xi having d features. Consequently, a model, represented by the function $h: X \rightarrow Y$, aims to predict an output y from a given input x, where X represents the input space and Y represents the output space.



- This study first considers the problem of the generation of multiple models using a single base model (Neural Network).

- intuitive ideology-
  manipulate the training examples

- To manipulate the training examples effectively, weightings can be assigned to individual features before inputting them into the base learning model to get multiple model.



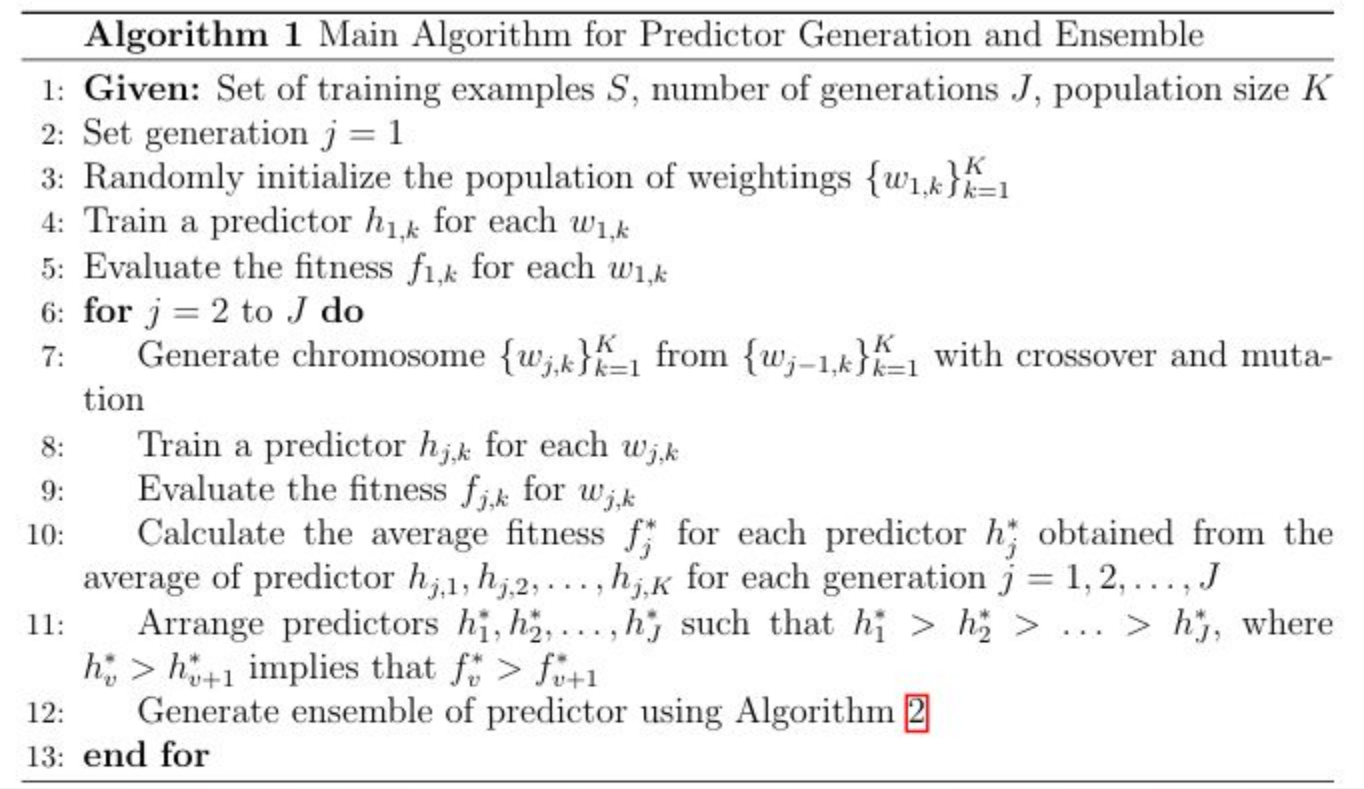
- to enhance the model's adaptability and performance, it is highly essential to assign appropriate and optimized weights.

- Proposing a methodology to assign optimal weightings to features inputted to the classifier is imperative.

- The weight space is vast, presenting a challenge in determining the most effective weightings.

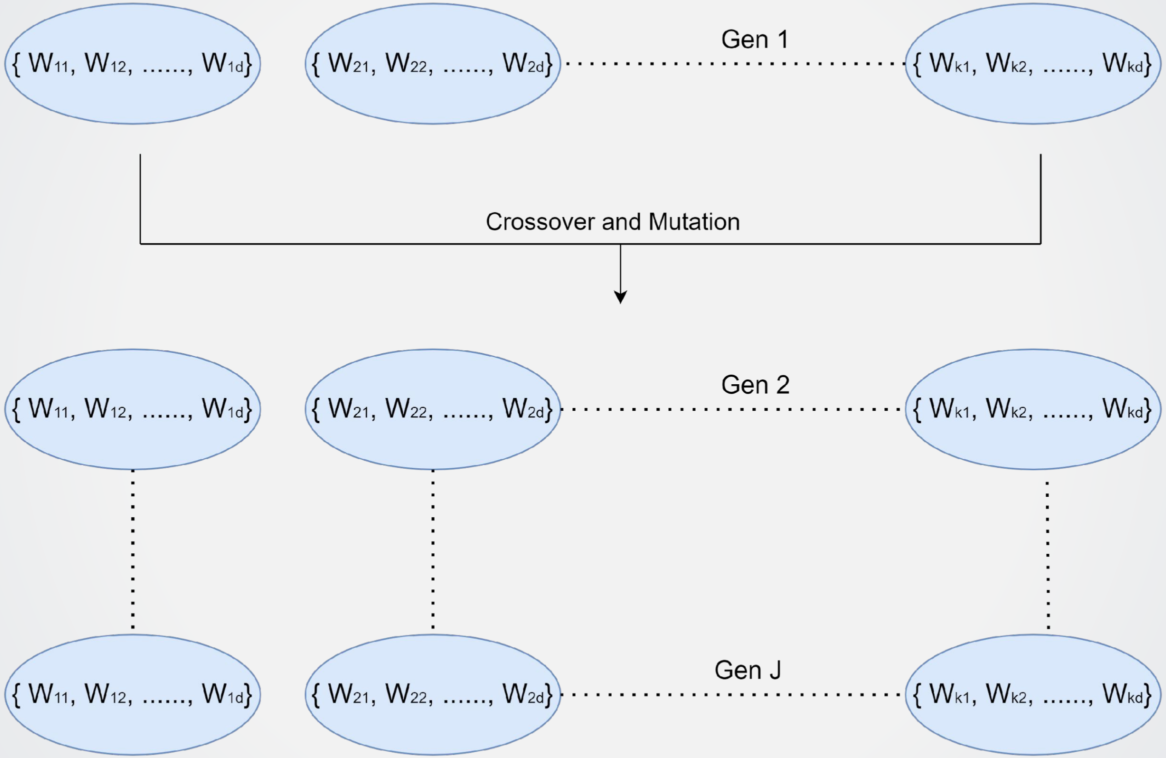
- Several methods exist to address this challenge, including
    - hill climbing,
    - simulated annealing, and
    - Evolutionary algorithms (genetic algorithms).

Evolutionary Algorithm(genetic algorithm)



Algorithm 1 Main Algorithm for Predictor Generation and Ensemble

```
**Given:** Set of training examples $S$, number of generations $J$, population size $K$
Set generation $j = 1$
Randomly initialize the population of weightings $\{w_{1,k}\}_{k=1}^{K}$
Train a predictor $h_{1,k}$ for each $w_{1,k}$
Evaluate the fitness $f_{1,k}$ for each $w_{1,k}$
**for** $j = 2$ to $J$ **do**
    Generate chromosome $\{w_{j,k}\}_{k=1}^{K}$ from $\{w_{j-1,k}\}_{k=1}^{K}$ with crossover and mutation
    Train a predictor $h_{j,k}$ for each $w_{j,k}$
    Evaluate the fitness $f_{j,k}$ for $w_{j,k}$
    Calculate the average fitness $f_j^*$ for each predictor $h_j^*$ obtained from the average of predictor $h_{j,1}, h_{j,2}, \ldots, h_{j,K}$ for each generation $j = 1, 2, \ldots, J$
    Arrange predictors $h_1^*, h_2^*, \ldots, h_J^*$ such that $h_1^* > h_2^* > \ldots > h_J^*$, where $h_v^* > h_{v+1}^*$ implies that $f_v^* > f_{v+1}^*$
    Generate ensemble of predictor using Algorithm 2
**end for**
```



Gen 1

{ W_{11}, W_{12}, ..., W_{1d}}

{ W_{21}, W_{22}, ..., W_{2d}}

{ W_{k1}, W_{k2}, ..., W_{kd}}

Crossover and Mutation

Gen 2

{ W_{11}, W_{12}, ..., W_{1d}}

{ W_{21}, W_{22}, ..., W_{2d}}

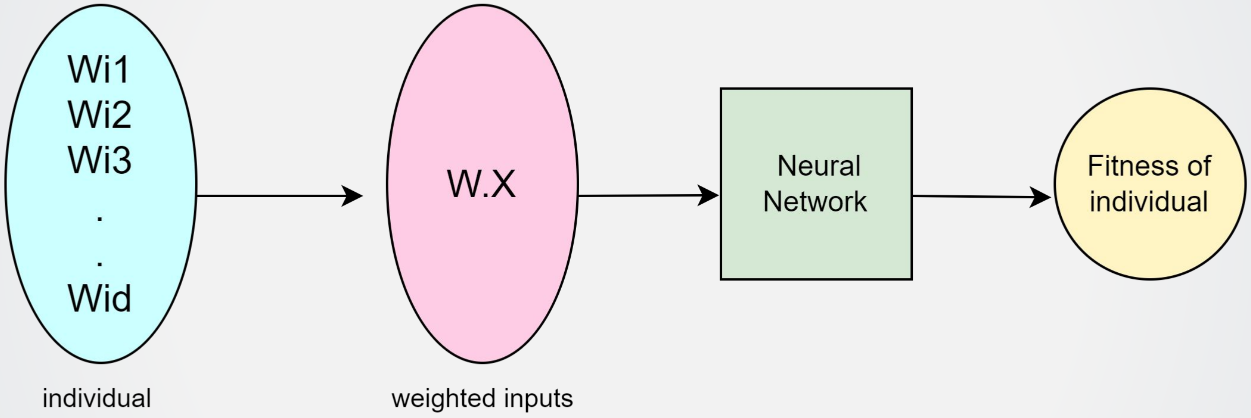
{ W_{k1}, W_{k2}, ..., W_{kd}}

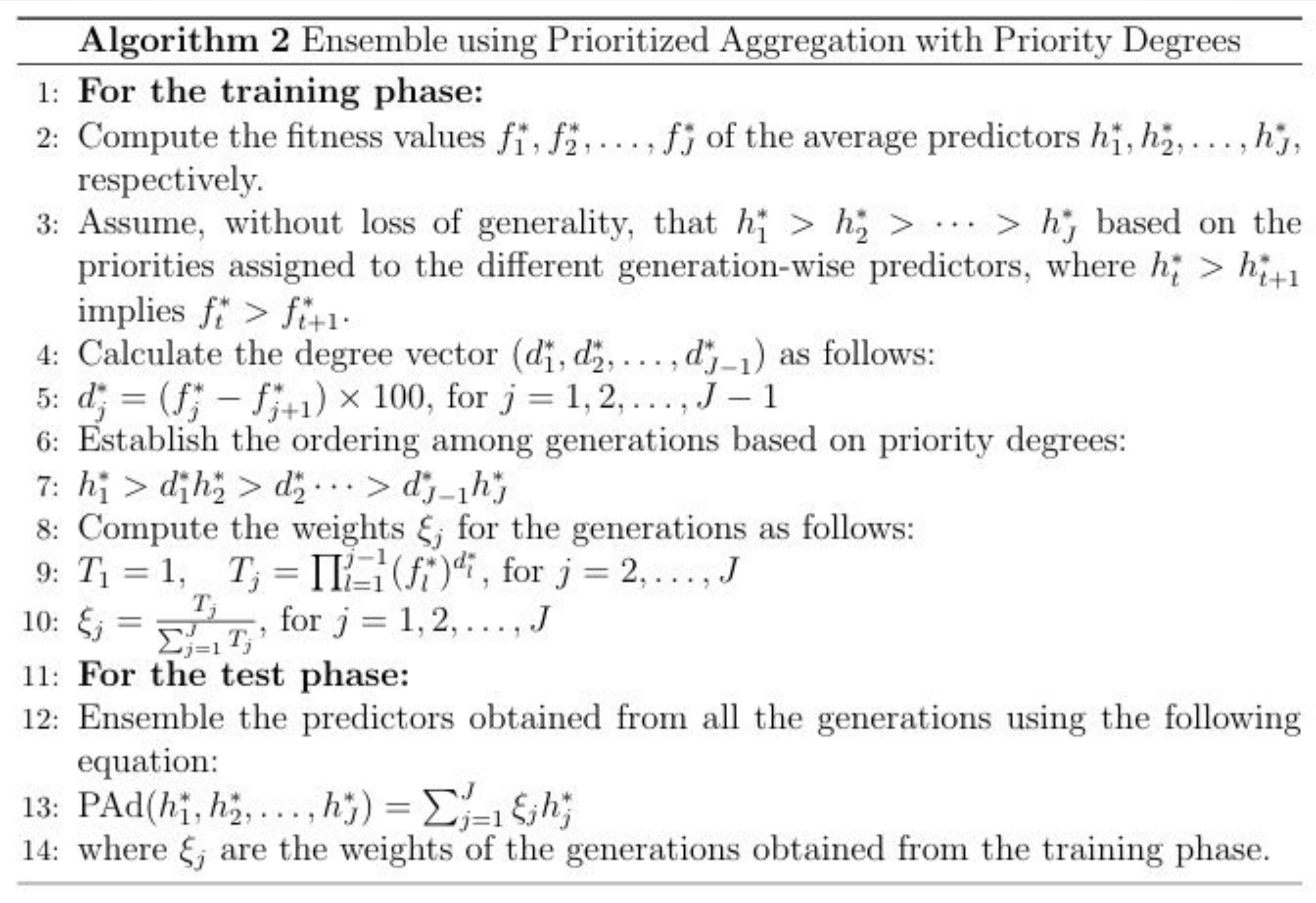
Gen J

{ W_{11}, W_{12}, ..., W_{1d}}

{ W_{21}, W_{22}, ..., W_{2d}}

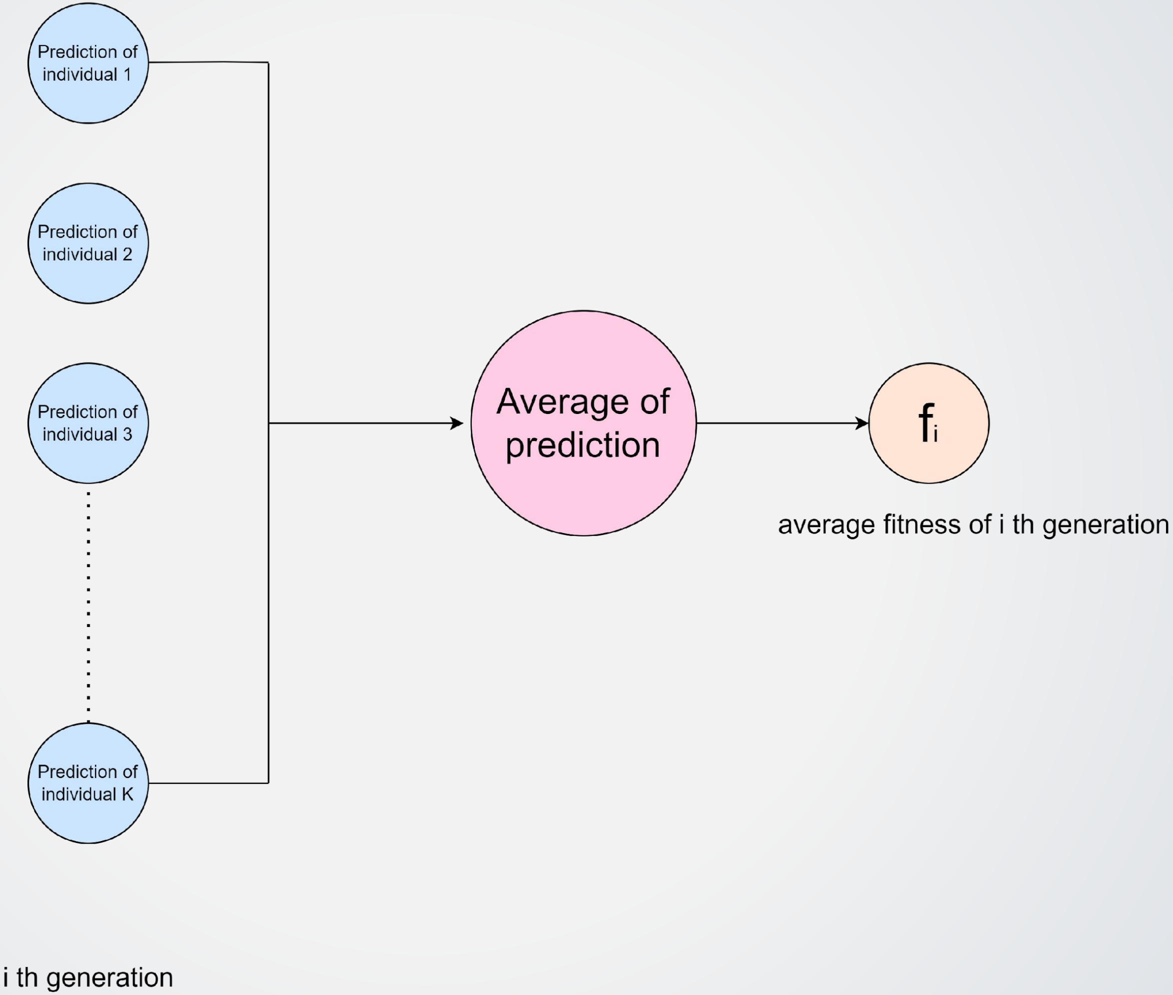
{ W_{k1}, W_{k2}, ..., W_{kd}}


Wi1
Wi2
Wi3
.
.
Wid
individual
W.X
weighted inputs
Neural Network
Fitness of individual

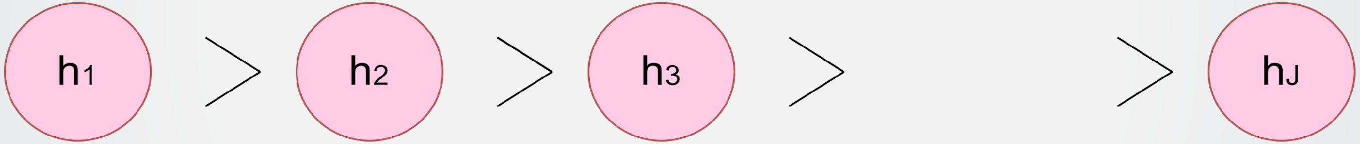


Algorithm 2 Ensemble using Prioritized Aggregation with Priority Degrees

```
For the training phase:
Compute the fitness values $f_1^*, f_2^*, \dots, f_J^*$ of the average predictors $h_1^*, h_2^*, \dots, h_J^*$, respectively.
Assume, without loss of generality, that $h_1^* > h_2^* > \cdots > h_J^*$ based on the priorities assigned to the different generation-wise predictors, where $h_t^* > h_{t+1}^*$ implies $f_t^* > f_{t+1}^*$.
Calculate the degree vector $(d_1^*, d_2^*, \dots, d_{J-1}^*)$ as follows:
$d_j^* = (f_j^* - f_{j+1}^*) \times 100$, for $j = 1, 2, \dots, J - 1$
Establish the ordering among generations based on priority degrees:
$h_1^* > d_1^* h_2^* > d_2^* \cdots > d_{J-1}^* h_J^*$
Compute the weights $\xi_j$ for the generations as follows:
$T_1 = 1, \quad T_j = \prod_{l=1}^{j-1} (f_l^*)^{d_l^*}$, for $j = 2, \dots, J$
$\xi_j = \frac{T_j}{\sum_{j=1}^{J} T_j}$, for $j = 1, 2, \dots, J$
For the test phase:
Ensemble the predictors obtained from all the generations using the following equation:
$\mathrm{PAd}(h_1^*, h_2^*, \dots, h_J^*) = \sum_{j=1}^{J} \xi_j h_j^*$
where $\xi_j$ are the weights of the generations obtained from the training phase.
```


Prediction of individual 1
Prediction of individual 2
Prediction of individual 3
Prediction of individual K
Average of prediction
f_i
average fitness of i th generation
i th generation

Arrange the predictors such that



- Next, we combine all the predictors
- We will demonstrate how to accomplish this using a technique known as the prioritized averaging operator with priority degree.
- evaluate the priority degrees as follows:

$$d_1^* = \frac{f_1^* - f_2^*}{f_2^*} \times 100$$

$$d_2^* = \frac{f_2^* - f_3^*}{f_3^*} \times 100$$

$$\vdots$$

$$d_{\mathrm{J}-1}^* = \frac{f_{\mathrm{J}-1}^* - f_{\mathrm{J}}^*}{f_{\mathrm{J}}^*} \times 100$$

where, indicates by how much percent the fitness of h_{t}^* is better than the fitness of $h_{\mathrm{t}+1}^*$

- Thus we have

$$h_1^* > d_1^* h_2^* > d_2^* h_3^* > \ \dots\dots\dots . . > d_{\mathrm{j}-1}^* h_{\mathrm{j}}^*$$

- These fitness values and priority degree are used to calculate the weights ξ_j for each generation j=1,2…,J The formula of ξ_j is given by:

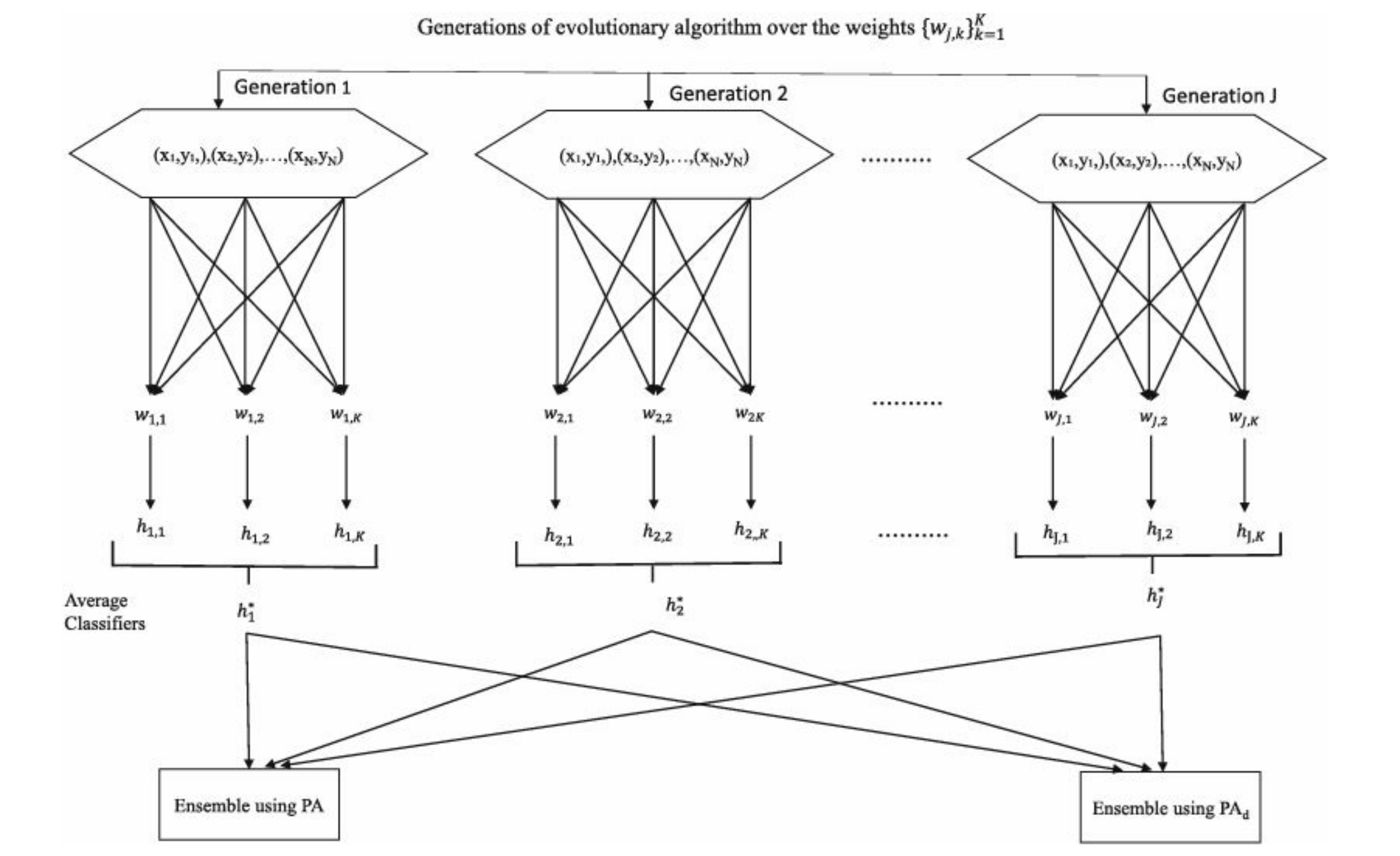
$$\xi_j = \frac{T_j}{\sum_{j=1}^{J} T_j} \qquad \text{for j = 1, 2………, J}$$

$$\text{Where } T_1 \text{=1 and } T_j = \prod_{l=1}^{j-1} (f_l^*)^{d_l^*} \quad \text{for j=2,3,......,J}$$

- Finally, we ensemble all the predictions obtain from all the generations by applying the PAd operator as follows:

$$p_{Ad}\left(h_1^*, h_2^*, \dots, h_j^*\right) = \sum_{j=1}^{J} \xi_j h_j^*$$

- During testing phase we apply ensemble of predictions obtained from all the generations, using the weights obtained during the training phase.


Generations of evolutionary algorithm over the weights $\{w_{j,k}\}_{k=1}^{K}$
Generation 1
Generation 2
Generation J
$(x_1,y_1,),(x_2,y_2),\ldots,(x_N,y_N)$
$(x_1,y_1,),(x_2,y_2),\ldots,(x_N,y_N)$
$(x_1,y_1,),(x_2,y_2),\ldots,(x_N,y_N)$
$w_{1,1}$
$w_{1,2}$
$w_{1,K}$
$w_{2,1}$
$w_{2,2}$
w_{2K}
$w_{J,1}$
$w_{J,2}$
$w_{J,K}$
$h_{1,1}$
$h_{1,2}$
$h_{1,K}$
$h_{2,1}$
$h_{2,2}$
$h_{2,,K}$
$h_{J,1}$
$h_{J,2}$
$h_{J,K}$
Average Classifiers
h_1^*
h_2^*
h_j^*
Ensemble using PA
Ensemble using PA_d

Data Preprocessing

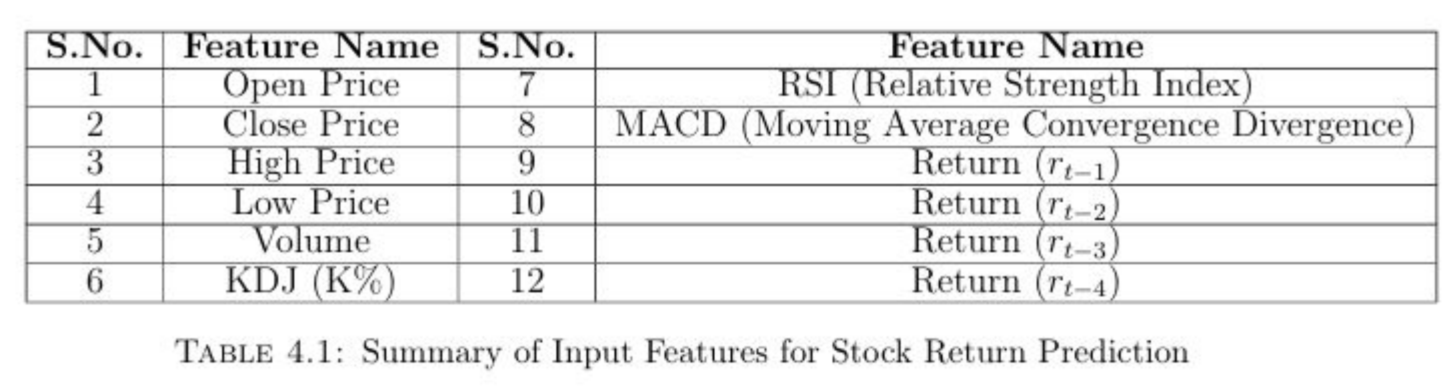
Objective:
To prepare the stock data with relevant features that enhance the accuracy and reliability of return predictions.

Original Features

Collected historical stock data, including:

- **Price** (Open, High, Low, Close, Adjusted Close)
- **Volume**: Number of shares traded
- **Date**: Timestamp for tracking changes over time

To enrich the dataset with more informative features, we added the following:



S.No.	Feature Name	S.No.	Feature Name
1	Open Price	7	RSI (Relative Strength Index)
2	Close Price	8	MACD (Moving Average Convergence Divergence)
3	High Price	9	Return (r_{t-1})
4	Low Price	10	Return (r_{t-2})
5	Volume	11	Return (r_{t-3})
6	KDJ (K%)	12	Return (r_{t-4})

TABLE 4.1: Summary of Input Features for Stock Return Prediction

Prediction Results

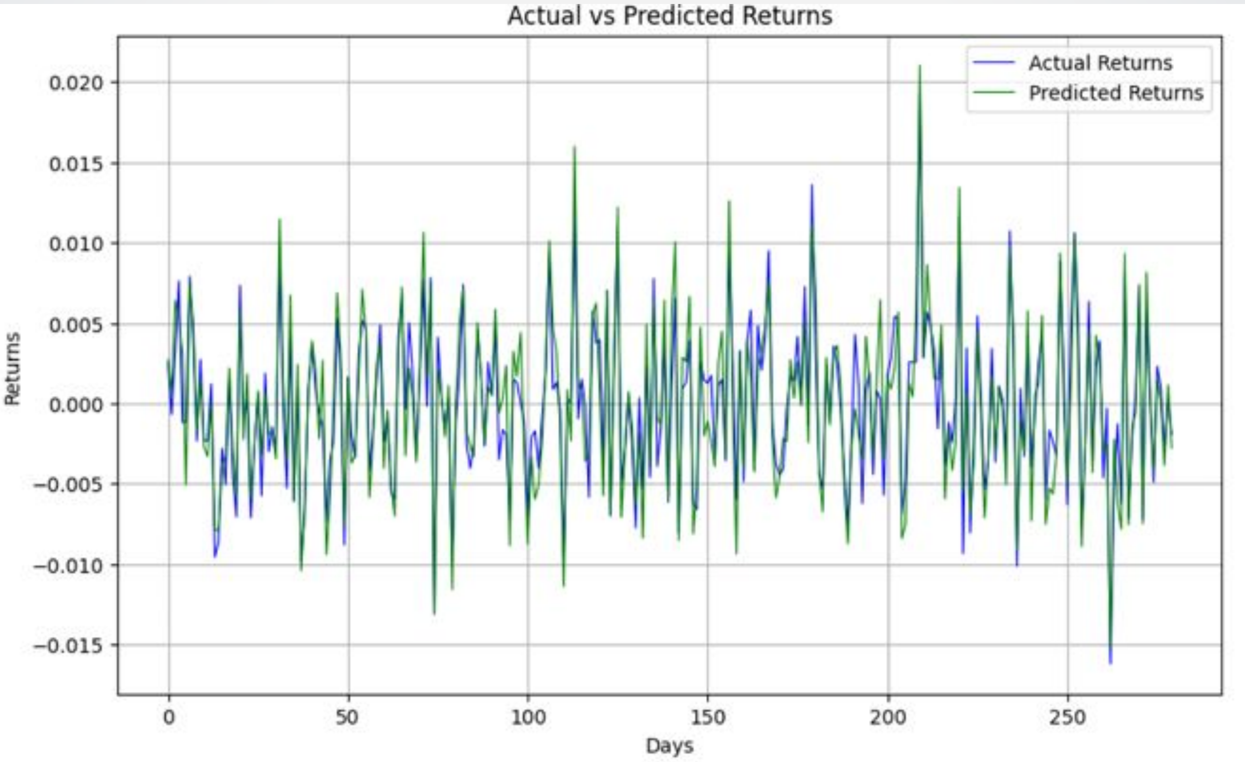


FIGURE 4.1: Actual vs Predicted Returns for NTPC Stock

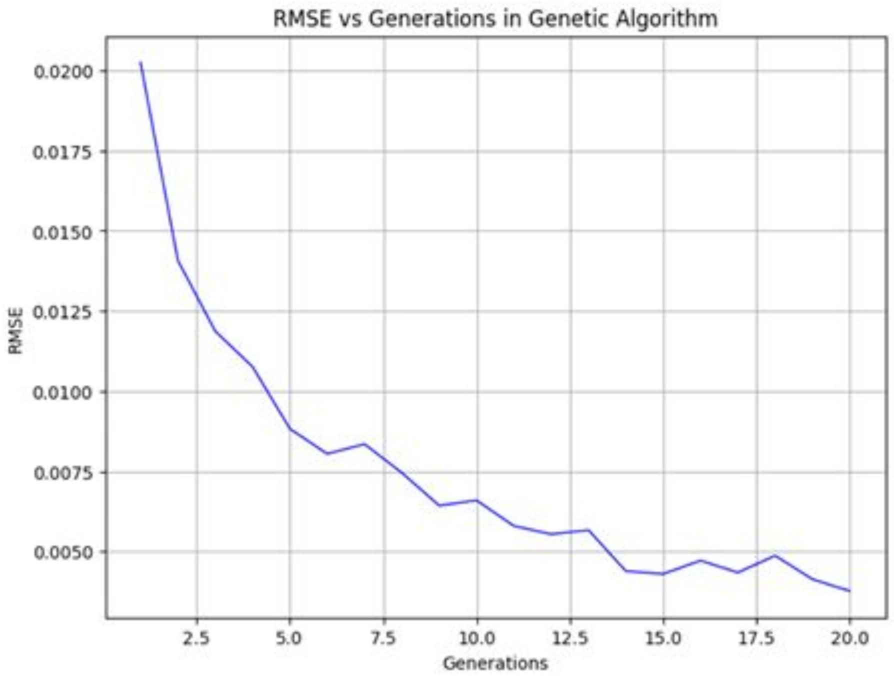


FIGURE 4.2: RMSE Scores Across Generations for Model Training

The average RMSE across all stocks is **0.01277**, indicating the overall prediction accuracy of the model.

Stock Preselection

Objective:
To identify a manageable number of high-performing stocks, balancing risk diversification with practical manageability for individual investors.

- **Portfolio Diversification**:
  Increasing the number of stocks reduces risk but can become less practical for individual investors.
- **Research Findings**:
  - Studies suggest that **5-10 stocks** offer a balance between risk management and ease of handling.
  - **Wang et al. (2019)**: Portfolios with **10 or fewer assets** are realistic for individual investors, with best performance found at **6 stocks**.
  - Similar insights by **Chaweewanchon & Chaysiri (2022)**, **Chen et al. (2021)**, and **Almahdi & Yang (2017)**.

Building on this research, we focus on the top 6 stocks with the highest predicted average returns for portfolio optimization.

Portfolio Optimization through Monte Carlo Simulation

- 50,000 portfolio simulations were conducted with randomized asset weights.
- Each simulation generated different portfolio returns, risks, and Sharpe ratios, which formed the **efficient frontier**.
- A **9% risk-free rate** was applied, reflecting current financial conditions in India.

Optimum Portfolio Characteristics

Sharpe Ratio: 3.0368

Annualized Expected Return: 0.4548

Annualized Volatility: 0.1182

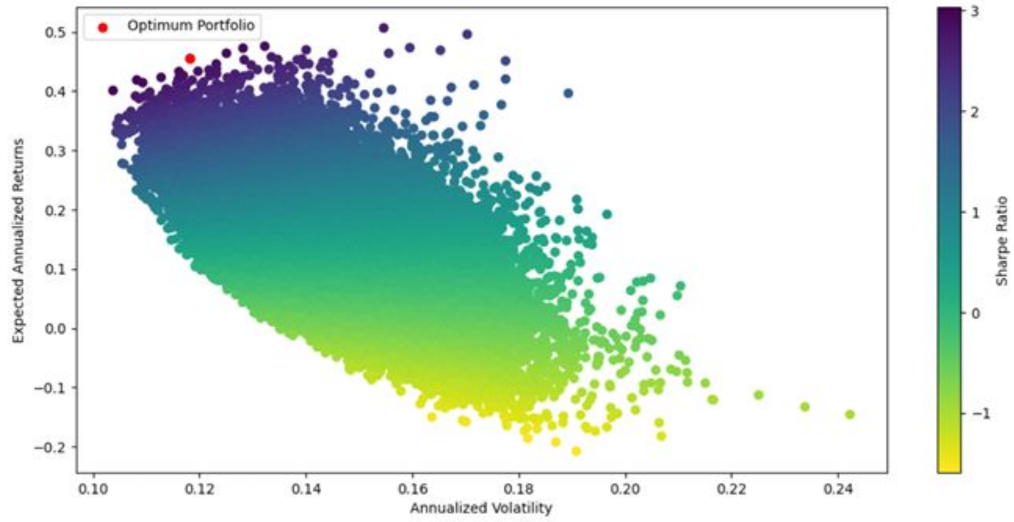


FIGURE 4.3: Efficient frontier

Portfolio Optimization through Mean Variance Model

Objective:
To construct the Efficient Frontier (Markowitz Curve) for the top 6 stocks, balancing risk and return

Bi-Objective Optimization Function:
A function combining risk and return into a single objective:

$$\lambda \cdot \text{Risk} + (1 - \lambda) \cdot (-\text{Return})$$

- **Risk**: Calculated as portfolio variance.
- **Return**: Expected return of the portfolio.
- **λ (Lambda)**: A trade-off parameter(risk aversion parameter) controlling the balance between risk and return.

Constraints:

- Portfolio weights must sum to 1 (full investment).
- Non-negative weights (no short selling).

Process and Methodology

1. **Varying λ**:
   - λ is varied from 0 to 1 in 100 steps, representing different risk-return preferences.
2. **Optimization Solver**:
   - The optimization problem is solved using the **CVXPY solver**, producing a series of portfolios.
3. **Portfolio Metrics**:
   - Portfolio risk and return are computed for each λ value.
   - **Sharpe ratio** is used to assess the risk-adjusted return of each portfolio.

Efficient Frontier Construction:

- Portfolio returns are plotted against portfolio risks for each λ value, resulting in the **Efficient Frontier**

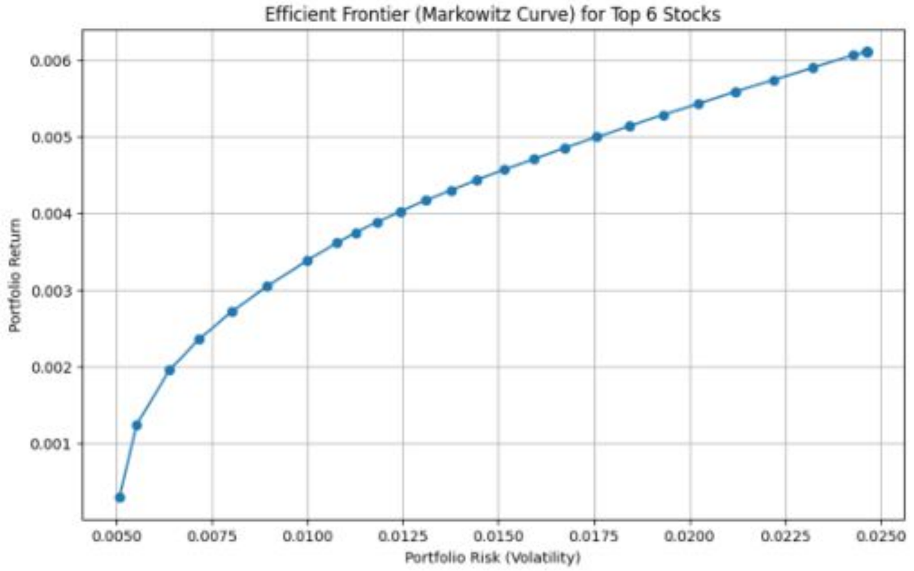


FIGURE 4.4: Efficient Frontier for Top 6 Stocks

The plot demonstrates the **best possible portfolios** for each level of risk, showing the optimal trade-off between risk and return.

Markowitz Model 1

Trade-off Parameter λ:

When **λ = 1**, the model prioritizes **risk minimization** and disregards return, focusing solely on minimizing portfolio volatility.

The optimal portfolio weights under **Markowitz Model 1** are calculated using the formula:

$$w^* = \frac{\Omega^{-1}e}{e^T\Omega^{-1}e}$$

Where: **Ω**: Covariance matrix of asset returns

e: A vector of ones (indicating all assets are considered)

Portfolio Attributes

Using the formula, the optimal portfolio with λ = 1 results in the following attributes:

- **Portfolio Return**: 0.0020, **Portfolio Volatility**: 0.0061, **Sharpe Ratio**: 0.320

References:

- Wang, X. et al. (2019). Portfolio performance in financial markets: A compar ative analysis of return predictions using lstm. Journal of Financial Studies, 36(7):10529–10537
- Chen, Y. et al. (2021). Return prediction for stock portfolios: A deep learning approach. International Journal of Financial Engineering, 12(8):123–150.
- Chaweewanchon, J. and Chaysiri, K. (2022). Optimal asset allocation for individ ual investors: A machine learning approach. Expert Systems with Applications, 50(6):123–145.
- Chang, T., Yang, S., and Chang, K. (2009). Portfolio optimization problems in different risk measures using genetic algorithm. Expert Systems with Applications, 36(7):10529–10537.
- Li, B. and Xu, Z. (2019). Prioritized aggregation operators based on the prior ity degrees in multicriteria decision-making. International Journal of Intelligent Systems, 34:1985– 2018
- Markowitz, H. (1952). Portfolio selection. The Journal of Finance, 7(1):77–91.
- Almahdi, S. and Yang, S. (2017). Portfolio optimization and risk management in stock markets using genetic algorithms. Journal of Computational Finance, 8(2):101–120.

Limitations -

- The predictive model’s accuracy depends on historical stock data, which may not account for sudden market shifts or external factors like economic events or geopolitical risks.

- Machine learning algorithms and ensemble methods can be computationally expensive, requiring significant time and processing power, limiting real-time applicability.

- The model assumes returns follow a normal distribution and that the covariance matrix is stable, which may not hold in real-world markets.

- The study is based on a single market index and specific stocks, and further testing across diverse asset classes and market conditions is needed.

Thanks

Open: The price at which the stock or asset started trading at the beginning of the day, giving an initial value for that day’s trading activity.

High: The highest price reached by the stock or asset during the trading day, reflecting the peak price investors were willing to pay.

Low: The lowest price reached during the trading day, showing the minimum price investors were willing to accept.

Prev. Close: The price at which the stock or asset closed on the previous trading day, serving as a benchmark for daily price change calculations.

LTP (Last Traded Price): The last price at which the stock was traded at the end of the trading day, providing an immediate value reflecting the most recent transaction.

Close: The official closing price of the stock or asset, usually calculated as the average of the last few trades at the day’s end. This is often used for technical analysis and portfolio valuation.

Volume: The total number of shares or contracts traded during the day, indicating the level of trading activity and investor interest in the stock.

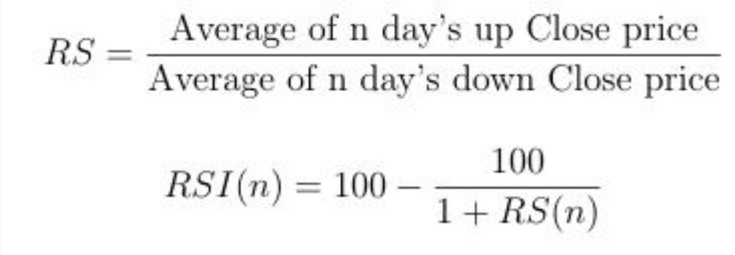
Value: The total monetary value of shares or contracts traded during the day, calculated as the volume multiplied by the price. This represents the market's total capital flow for the stock on that day.

No of Trades: The total number of individual trades or transactions made during the day, indicating market participation and liquidity for the stock.

- KDJ - The KDJ indicator provides information on the strength and direction of price movements, which is valuable for predicting returns. The K and D lines help detect bullish or bearish trends.
  When the J line rises above 100 or falls below 0, it suggests the stock is either overbought or oversold, potentially signaling a reversal. This can help forecast when the stock's return might change direction.
  The formula for calculating K% is as follows:

  K = (Close price−Lowest price(in n period))/ (Highest price (in n period)− Lowest price(in n period))

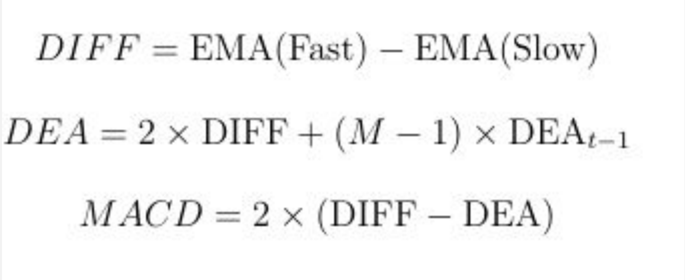
- The **Relative Strength Index (RSI)** is a momentum indicator that measures the speed and change of price movements, commonly used to identify overbought or oversold conditions in a stock. The RSI is displayed as a line graph on a scale of 0 to 100.The formula for calculating RSI (n = 14) is:



$$RS = \frac{\text{Average of n day's up Close price}}{\text{Average of n day's down Close price}}$$

$$RSI(n) = 100 - \frac{100}{1 + RS(n)}$$

The **Moving Average Convergence Divergence (MACD)** is a popular technical indicator in stock market analysis, designed to reveal changes in the strength, direction, momentum, and duration of a trend in a stock's price. It is calculated by taking the difference between two exponential moving averages (EMAs)—typically, the 12-day and 26-day EMAs.



$$DIFF = \mathrm{EMA(Fast)} - \mathrm{EMA(Slow)}$$

$$DEA = 2 \times \mathrm{DIFF} + (M - 1) \times \mathrm{DEA}_{t-1}$$

$$MACD = 2 \times (\mathrm{DIFF} - \mathrm{DEA})$$

where Fast = 12, Slow = 26, and M = 9.