ROBERT GORDON UNIVERSITY ABERDEEN TEF Gold

INFORMATICS INSTITUTE OF TECHNOLOGY

BSc (Hons) Artificial Intelligence And Data Science
Level 04

CM 4601
Computational Intelligence

Optimized Asset Allocation Using Genetic Algorithms and Machine Learning

CW Report

NADUN SHAMIKA SENARATHNE
IIT ID: 20210488
RGU ID: 2117538

Table of Contents

Task 1: Portfolio Optimization Problem Identification

1. Problem Identification:

- **Industry:** Finance
- **Application:** Optimize asset allocation in a financial portfolio to maximize returns while minimizing risks.
- **Complexity:**
  - There are numerous asset combinations conceivable in a portfolio, especially with a high number of accessible assets.
  - Each asset's risk and return profile fluctuates over time owing to market circumstances, creating uncertainty.
  - Multiple competing objectives, such as maximising profits and minimising risk (mean-variance optimization).
  - A Constraints such as budget, industry diversity, and risk tolerance further complicate the situation.

2. Why Traditional IT or Numerical Optimization Fails:

- **Exponential Growth of Combinations:** As the number of assets rises, the number of alternative portfolio configurations expands exponentially, making thorough searches impossible.
- **Non-Convex Objective Function:** Portfolio optimization includes non-linear interactions between assets, creating a non-convex objective function with several local optima. This makes classic gradient-based optimization approaches challenging to apply.
- **Uncertainty:** Traditional numerical optimization methodologies are ineffective in capturing market uncertainty due to their deterministic nature and continuous change.

3. Why Genetic Algorithms (GA) Are Suitable for This Problem:

Evolutionary techniques, such as Genetic techniques (GA), are ideal for this sort of problem because they can effectively traverse a wide, complicated search domain. They are capable of managing several conflicting objectives and providing strong answers in the face of ambiguity. Unlike classic optimization approaches, GAs may avoid being stuck in local optima and adapt effectively to the changing nature of financial markets.

Task 2: Justification for Using Evolutionary Algorithms (GA)

1. Complexity of Search Space:

- Traditional search methods become ineffective when the quantity of assets increases dramatically.
- The present implementation of the GA utilizes tournament selection, uniform crossover, and random mutations to effectively traverse the huge space.

2. Handling Multiple Objectives:

- Optimizing a portfolio requires balancing many objectives, including maximising return, minimising risk, and ensuring diversification.
- The fitness function in the current implementation combines the Sharpe ratio, risk penalties, and incentives for balanced asset allocations to optimise several objectives.

3. Non-Convex and Non-Smooth Objective Functions:

- A portfolio's return-risk landscape is non-linear and influenced by complicated inter-asset correlations.
- The GA's stochastic evolution process avoids local optima in the present implementation of covariance matrix-based risk computation, which is not convex.

4. Stochastic Nature and Risk Management:

- Deterministic optimization is ineffective in financial markets due to its uncertainty and volatility.
- In the current implementation:
  - GA's mutation and crossover operators use randomization to prevent premature convergence.
  - The Random Forest regression model predicts future asset returns by using previous volatility to direct GA development.

5. Constraint Handling:

- Portfolio optimization involves both hard limitations (e.g., budget limits, minimum asset weights) and soft constraints (e.g., risk tolerance, diversification).
- In the current implementation:
  - The current method enforces hard limitations through penalties in the fitness function for weight deviations or high risk.
  - To promote diversification, asset weights with significant variance are penalised while balanced allocations are rewarded.

6. Robustness and Multi-Run Evaluation:

- Financial optimization demands resilience in many circumstances.
- In the current implementation:
  - GA is performed numerous times to ensure convergence and consistency.
  - This multi-run technique optimises the GA without relying on a single solution, providing a comprehensive perspective of its performance.

7. Simulation of Future Scenarios:

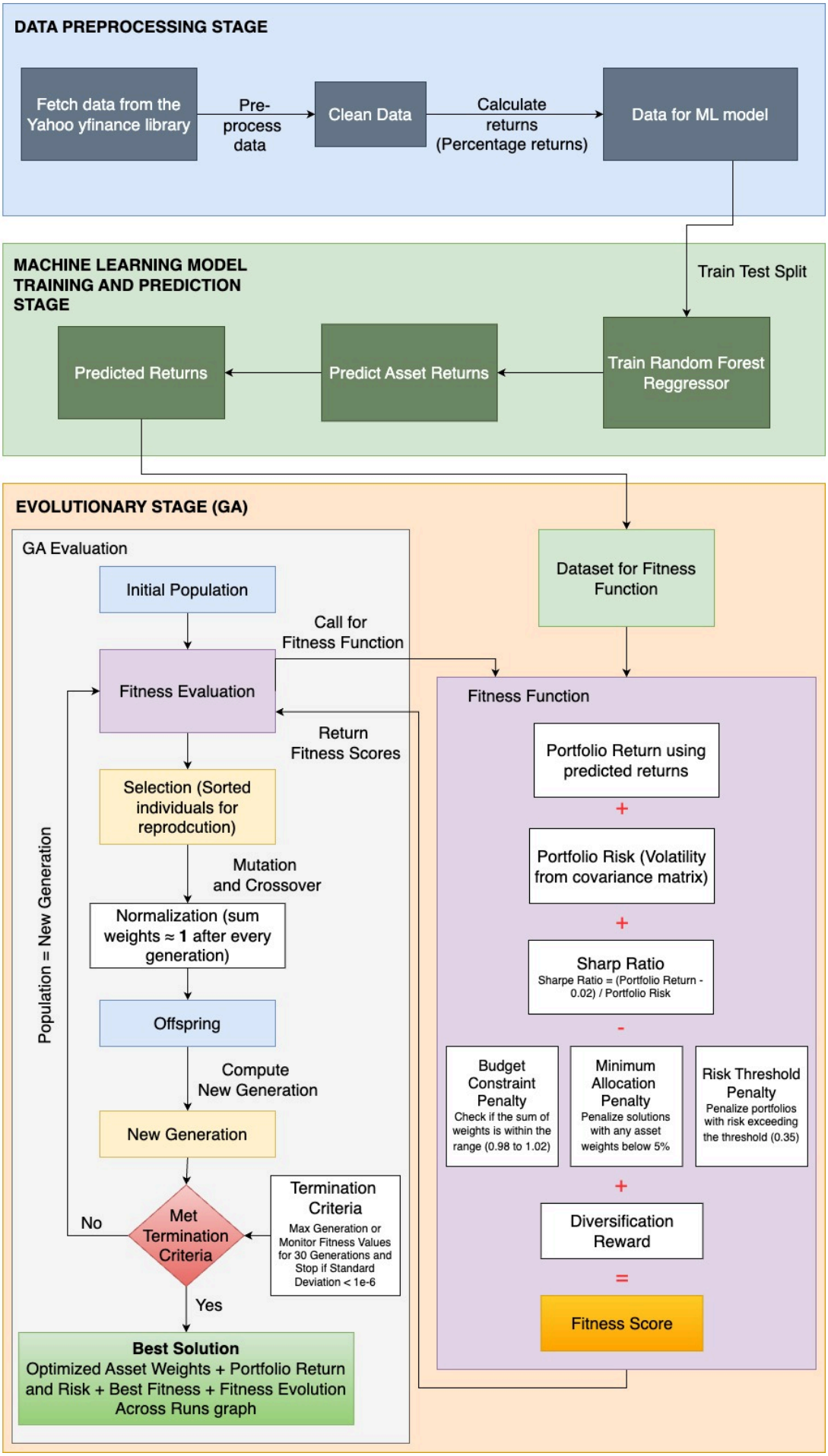
- Optimizing portfolios requires considering future market situations.
- The present implementation uses the Random Forest model to estimate predicted returns based on current data, simulate future scenarios, and guide optimization.

The code improvements, such as machine learning integration for return prediction, multi-run evaluation for robustness, and sophisticated constraint handling in the fitness function, show the versatility and power of Genetic Algorithms in solving complex, multi-objective, and non-linear portfolio optimisation problems. These enhancements make GAs suited for real-world financial optimisation tasks.

Task 3: Holistic View and Modularized Diagram

The figure illustrates the modularized architecture of the portfolio optimization process, which employs Genetic Algorithms (GA) and machine learning. It starts with the Data Preprocessing Stage, which involves retrieving and cleaning data, followed by return calculation for machine learning input. In the Machine Learning Stage, asset returns are predicted using a Random Forest Regressor. The Evolutionary Stage uses GA to optimise the portfolio by evaluating its fitness, which takes into account portfolio return, risk, Sharpe Ratio, constraints (budget, allocation, risk), and diversification rewards. Termination criteria include terminating early when fitness levels reach a plateau. The approach generates optimal asset weights, portfolio metrics, and fitness evolution visualizations.

Here’s a holistic view of the system:


DATA PREPROCESSING STAGE
Fetch data from the Yahoo yfinance library
Pre-process data
Clean Data
Calculate returns (Percentage returns)
Data for ML model
Train Test Split
MACHINE LEARNING MODEL TRAINING AND PREDICTION STAGE
Train Random Forest Reggressor
Predict Asset Returns
Predicted Returns
EVOLUTIONARY STAGE (GA)
GA Evaluation
Initial Population
Call for Fitness Function
Fitness Evaluation
Return Fitness Scores
Selection (Sorted individuals for reprodcution)
Mutation and Crossover
Normalization (sum weights ≈ 1 after every generation)
Offspring
Compute New Generation
New Generation
Met Termination Criteria
No
Yes
Termination Criteria
Max Generation or Monitor Fitness Values for 30 Generations and Stop if Standard Deviation < 1e-6
Population = New Generation
Best Solution
Optimized Asset Weights + Portfolio Return and Risk + Best Fitness + Fitness Evolution Across Runs graph
Dataset for Fitness Function
Fitness Function
Portfolio Return using predicted returns
+
Portfolio Risk (Volatility from covariance matrix)
+
Sharp Ratio
Sharpe Ratio = (Portfolio Return - 0.02) / Portfolio Risk
-
Budget Constraint Penalty
Check if the sum of weights is within the range (0.98 to 1.02)
Minimum Allocation Penalty
Penalize solutions with any asset weights below 5%
Risk Threshold Penalty
Penalize portfolios with risk exceeding the threshold (0.35)
+
Diversification Reward
=
Fitness Score

Task 4: Define the Methodology

1. Chromosome Structure

- **Chromosome Representation**: Each chromosome represents a potential solution, such as a financial portfolio.
- **Representation in Code**: Each chromosome is a vector of real integers, with each value representing the capital allocation to an asset. For example, a chromosome representing five assets (AAPL, MSFT, GOOGL, AMZN, and TSLA) may look like [0.20, 0.25, 0.15, 0.30, 0.10].
- **Gene**: Each gene represents the weight of a certain asset in the portfolio.
- **Constraint**: All asset weights must be substantially equal to one to ensure 100% portfolio allocation. The fitness function penalises weights that stray from the 98% to 102% range.

2. Fitness Function

The fitness function evaluates a prospective portfolio's performance by balancing risk, return, and diversification. The aim is to maximize the Sharpe Ratio, which is a measure of return adjusted for risk.

Components of the Fitness Function:

1. **Portfolio Return**: The portfolio's return is determined as the weighted sum of expected asset returns. The anticipated returns are calculated using a Random Forest Regressor trained on historical asset returns.
2. **Portfolio Risk**: The covariance matrix of asset returns is used to determine the portfolio's risk or volatility. This matrix tracks how the returns of several assets change together.
3. **Sharpe Ratio**: The Sharpe Ratio serves as the major fitness metric. It calculates the return-risk trade-off by dividing the difference between the portfolio's projected return and a risk-free rate by the volatility of the portfolio. The code assumes a risk-free rate of 2%.
4. **Diversification Reward**: Portfolios with non-zero asset weights receive an additional benefit. This fosters diversity rather than focussing just on a few items.
5. **Balance Penalty**: Portfolios with large volatility in asset weights are penalised to ensure a more balanced allocation. Lower weight standard deviations are rewarded, encouraging evenly dispersed portfolios.

Constraint Handling in the Fitness Function:

- A penalty is given if the total of weights is not between 98% and 102%.
- Penalties are enforced for asset weights below 5% to ensure meaningful allocations.
- If the portfolio's risk exceeds 0.35, the solution is penalised.

3. Constraints

The following constraints ensure that the portfolio solutions are realistic and feasible:

1. **Budget Constraint**: Total portfolio allocation should be about 100%.
2. **Minimum Allocation**: Allocate at least 5% of each asset to avoid negligible contributions.
3. **Risk Constraint**: The portfolio risk (volatility) cannot exceed a specified threshold of 0.35.
4. **Diversification**: To decrease concentration risk, portfolios are rewarded for distributing allocations across numerous assets.

4. Genetic Algorithm (GA) Operations

The Genetic Algorithm is used to iteratively evolve portfolios toward optimal solutions.

- **Selection**: The tournament selection technique is used to choose parent portfolios. Higher fitness portfolios have a better probability of being chosen for reproduction.
- **Crossover**: The algorithm employs uniform crossover, combining asset weights from two parent portfolios to produce child portfolios. This strategy provides a diverse population.
- **Mutation**: Random mutation is used to increase variety and prevent premature convergence. The mutation rate in the code is set to 25%, causing asset weights to be adjusted at random within legal ranges.
- **Normalization**: After mutation and crossover, portfolio weights are normalised to add up to 1 while adhering to the budget restriction.

5. Algorithm Flow

1. **Initialize Population**: Generate a random population of portfolios with asset weights between 0 and 1.
2. **Evaluate Fitness**: The fitness function takes into account return, risk, diversity, and restrictions to determine each portfolio's fitness level.
3. **Selection**: Parent portfolios are chosen based on fitness scores through tournament selection.
4. **Crossover and Mutation**: New offspring portfolios are created by mixing the weights of parent portfolios (crossover) and making random modifications.
5. **Normalization**: Asset weights are normalised to add up to 100%.
6. **Constraint Handling**: Portfolios that violate limitations (e.g. risk or minimum allocation) receive poorer fitness ratings.
7. **Termination Criteria**: Run max number of generations or if the fitness values do not improve significantly over 30 generations, the algorithm stops early.
8. **Output the Best Portfolio**: Output the optimal portfolio, including predicted return, risk, and fitness score (Sharpe Ratio), at the end of the evolution process.

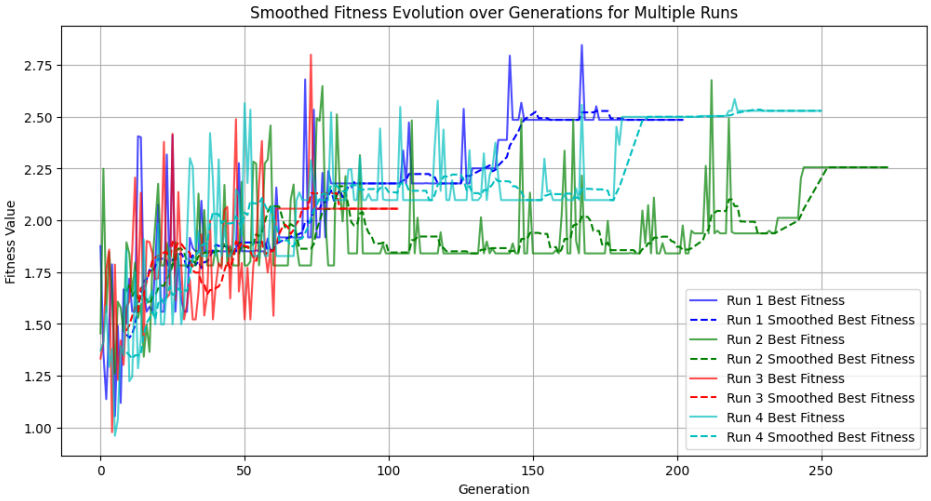
This methodology optimizes financial portfolios by combining machine learning forecasts with Genetic Algorithms. This technique enforces realistic portfolio limits, including total allocation, minimum asset weight, and risk constraints.

- The fitness function incorporates Sharpe Ratio, diversification incentives, and penalties to ensure robust evaluation.
- Genetic Algorithms use selection, uniform crossover, mutation, and normalization to effectively explore solution space while balancing return and risk.

The end result is a strong, diverse, and optimized portfolio that satisfies predetermined goals and restrictions while reacting to market fluctuations.

Evaluation Results

The evaluation results of the Genetic Algorithm (GA) runs show the stochastic character of the optimization process, which causes modest differences in outcomes over several iterations. Figures 1 and 2 show how fitness evolves across generations for both the Standard Execution and Batch-wise Execution techniques.



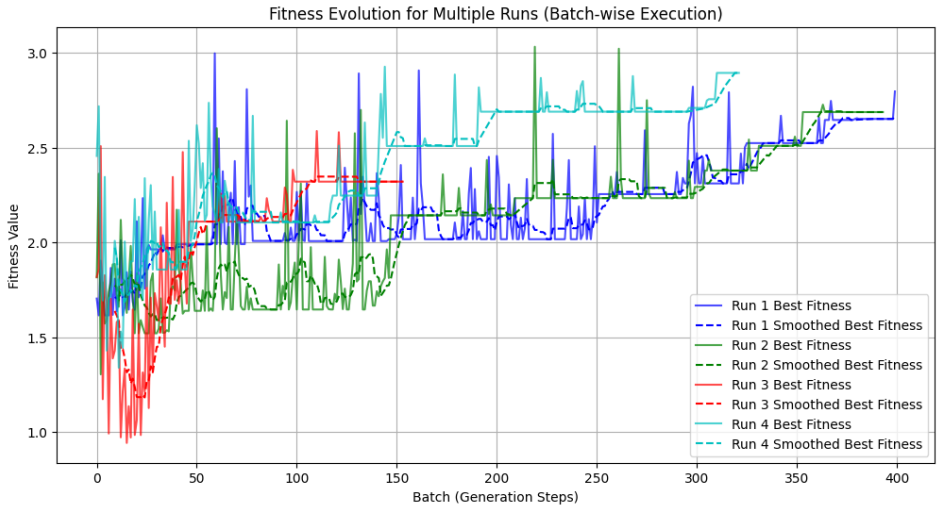

```
Run 1 Best Portfolio: [0.11610261 0.24078415 0.         0.49901869 0.14409455]
Run 1 Fitness (Sharpe Ratio + Diversification Reward): 2.8462371462341225
Run 1 Stopped at Generation: 203

Run 2 Best Portfolio: [0.21157237 0.02163707 0.         0.08776126 0.67902929]
Run 2 Fitness (Sharpe Ratio + Diversification Reward): 2.6769475392898414
Run 2 Stopped at Generation: 274

Run 3 Best Portfolio: [0.39699543 0.         0.35986113 0.24314344 0.        ]
Run 3 Fitness (Sharpe Ratio + Diversification Reward): 2.799686940303556
Run 3 Stopped at Generation: 104

Run 4 Best Portfolio: [0.11233559 0.24985767 0.1257401  0.06590313 0.44616351]
Run 4 Fitness (Sharpe Ratio + Diversification Reward): 2.5855766139011083
Run 4 Stopped at Generation: 251
```

Figure 1




```
Run 1 Best Portfolio: [0.02872333 0.32122436 0.55301074 0.09704157 0.        ]
Run 1 Fitness (Sharpe Ratio + Diversification Reward): 2.996521520768235
Run 1 Stopped at Generation: 394

Run 2 Best Portfolio: [0.04220028 0.43507855 0.0365985  0.08858423 0.39753845]
Run 2 Fitness (Sharpe Ratio + Diversification Reward): 3.031279934290999
Run 2 Stopped at Generation: 152

Run 3 Best Portfolio: [0.39743003 0.00068463 0.55170576 0.03164108 0.0185385 ]
Run 3 Fitness (Sharpe Ratio + Diversification Reward): 2.587198942231943
Run 3 Stopped at Generation: 153

Run 4 Best Portfolio: [0.25515814 0.25893085 0.02423201 0.         0.46167901]
Run 4 Fitness (Sharpe Ratio + Diversification Reward): 2.926064403810668
Run 4 Stopped at Generation: 154
```

Figure 2

Fitness Scores:

- Both approaches show improved fitness scores across generations.
- Fig. 1 (Standard Execution) shows that fitness values grow quickly in early generations, while some runs plateau or stall due to premature convergence.
- Fig. 2 (Batch-wise Execution) shows that fitness improves gradually and consistently in each batch. This minimises the likelihood of early convergence and results in a more consistent fitness value.

Fitness Evolution:

- Fig. 1 demonstrates significant fluctuations in fitness scores between generations as the GA explores the search space. However, fitness levels tend to stabilise after a number of generations.
- Fig. 2 shows a smoother fitness development with distinct leaps between batches, indicating that the GA refines solutions in organized phases. The incremental progress suggests greater investigation of the solution space over time.

Portfolio Allocations:

- While both procedures conform to the established restrictions (budget, minimum allocation, and risk threshold), the Batch-wise Execution in Fig. 2 results in portfolios with higher diversification and more stable fitness values than the Standard Execution in Fig. 1.
- The balanced solutions in Fig. 2 demonstrate how Batch-wise Execution may prevent premature convergence and increase solution quality.

Consistency:

- As seen in Fig. 1, Standard Execution results in variable fitness scores, with some runs plateauing early. This paradox derives from the GA's stochastic behaviour.
- In contrast, Fig. 2 demonstrates that Batch-wise Execution gives more consistent outcomes over numerous runs, since fitness improves progressively and consistently in each batch.

Figs. 1 and 2 show the performance differences between Standard Execution and Batch-wise Execution of the Genetic Algorithm. While Standard Execution (Fig. 1) yields quick but uneven fitness gains, Batch-wise Execution (Fig. 2) offers a more consistent, progressive, and robust optimisation procedure. This makes Batch-wise Execution the optimal method for increasing fitness values and portfolio diversity over time.