Chandigarh Engineering College Jhanjeri
Mohali-140307
Department of Computer Science & Engineering

Smart Stock Portfolio Optimizer

Project-I

BACHELOR OF TECHNOLOGY
(Computer Science and Engineering)

SUBMITTED BY:
Vikash Kumar, 2420367
Ishu Barman, 2420258
Dharmu Kumar, 2420353
Shubham Tandon, 2320207
Jan 2025

Under the Guidance of
Dr. Ishu Sharma
Associate Professor

Department of Computer Science & Engineering
Chandigarh Engineering College Jhanjeri
Mohali - 140307

Table of Contents

1.Introduction

In today's financial markets, investors have to choose from a variety of stocks, but not everyone has substantial funds to invest. They often have a limited amount of money, meaning they can only buy a few stocks that fit within their budget.

The goal of this project is to develop a tool that helps investors choose the best stocks within their budget. This tool will utilize the Knapsack Dynamic Problem approach, a common optimization technique. It involves selecting stocks within a limited budget to maximize value or returns while balancing risk-adjusted returns.

In this project, the objective is to create a tool that enables investors to select stocks such that the total cost does not exceed the given budget while maximizing the total expected return. The challenge is to identify the best stocks from a given list that fit within financial constraints while also offering the highest return based on risk-adjusted values. This tool aims to provide investors with an optimal strategy to make informed financial decisions and maximize profits.

1.1 Problem Context:

Stocks investments are risky, but they gives the potentials for returns. The main challenge is to select the right stock within the budget that gives the best return. To do this, each stock has a cost per share (similar to weight in the knapsack problem) and an expected return (similar to value in Knapsack problem).

For Example :

Stock A: Cost = Rs. 1500, Expected Return = Rs. 300

Stock B: Cost = Rs. 900, Expected Return = Rs. 100

Stock C: Cost = Rs 2000, Expected Return = Rs. 500

Stock D: Cost = Rs. 2500, Expected Return = Rs. 800

Budget = Rs 4300

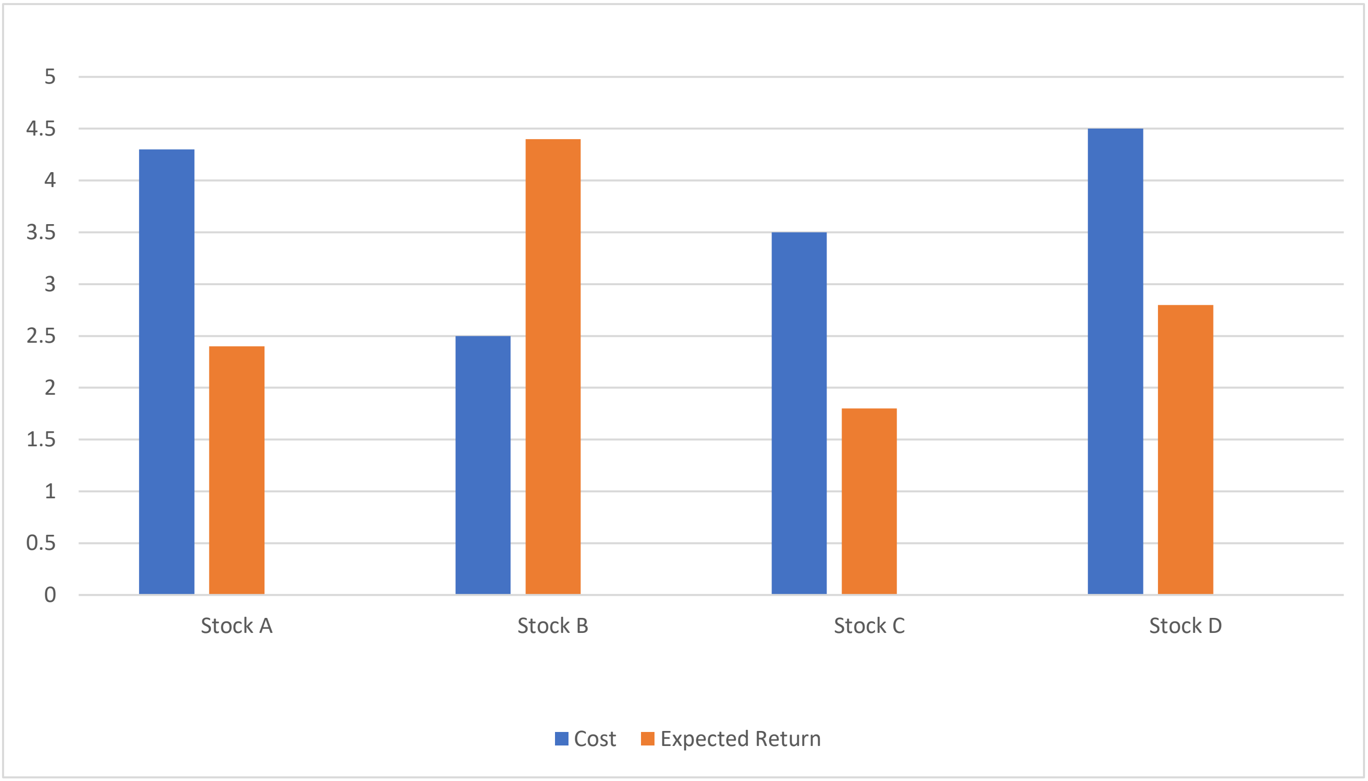

Figure - 1.1

The optimal Selection for this would be to choose **Stock D** and **Stock A** for a total cost of Rs. 4300 and a **return of Rs. 1100.**

1.2 Problem Statement:

In the context of this Project, the objective is to develop a tool that allows an investor to select a stock in such a way that the total cost does not exceeds a given total budget, and at that same time total expected return is maximized.

1.3 Scope of the Project

This project will focus on developing a web-based platform that will :

- Allow users to input their available budget.
- Let users define the stocks they want to consider.
- Allow users to input the costs and expected returns of each stock.
- Provide an optimal solution based on the available budget.

2.Literature survey

Portfolio optimization is a crucial aspect of financial and investment decision-making, and over the years, various techniques have been explored to enhance its effectiveness. The primary methods include mean-variance optimization, dynamic programming, and linear programming. Each of these methods has its own strengths and weaknesses, making them suitable for different types of problems depending on their complexity and nature. While mean-variance optimization is widely used for balancing returns and risks, dynamic programming provides a more structured approach to solving multi-stage decision problems. Linear programming, on the other hand, is useful for problems with linear constraints and objectives.

Dynamic programming is particularly effective in solving problems like the Knapsack problem, where decisions must be made based on an available budget and the objective of maximizing returns. This method breaks down the problem into smaller, manageable subproblems, solving each optimally and using these solutions to build up to the overall solution. The advantage of dynamic programming lies in its ability to handle complex decision-making scenarios efficiently, ensuring optimal allocation of resources in various fields.

In the realm of financial portfolio optimization, considering risk factors is essential for constructing robust investment strategies. Traditional models often focus primarily on returns, which can lead to the creation of unbalanced or excessively risky portfolios. While high returns are attractive, they often come with increased risk, which may not align with every investor’s objectives and risk tolerance levels. By incorporating risk as a fundamental factor in decision-making, investors can strike a balance between potential gains and the level of risk they are willing to accept.

In summary, portfolio optimization techniques continue to evolve, integrating more sophisticated methods to improve decision-making processes. Dynamic programming, along with other optimization techniques, plays a crucial role in addressing complex financial problems. By incorporating risk factors, investors can develop strategies that not only maximize returns but also align with their financial goals and risk tolerance. As research progresses, the field of portfolio optimization will continue to expand, offering more refined solutions to meet the diverse needs of investors and financial institutions.

3.Problem Formulation

Investing in the stock market is a challenge that involves balancing high returns with budget constraints. A key decision for investors is selecting stocks to maximize portfolio return while staying within a set budget. This becomes complex when considering a variety of stocks, each with unique prices and projected returns. The uncertainty of future performance adds another layer of difficulty. Investors must weigh potential gains against risks through careful research and analysis.

The objective of this project is to develop a tool that helps investors make informed stock choices while ensuring the total cost does not exceed their budget. Simultaneously, it aims to maximize total expected returns. This optimization problem closely resembles the "Knapsack Problem" in computer science, where the goal is to select the most valuable items within a limited capacity. In this scenario, stocks represent items, expected return corresponds to value, and the investor's budget serves as the constraint.

Each stock has two important factors:

1. Cost:
   - The price of buying one share of Stock.
2. Expected Return:
   - The profit an investor expects to make from that stock.

The challenge is to choose the right mix of stocks that fits within the budget limit but still provides the best return.

The problem involves these key steps:

1. Input: A list of Stocks, where each stock having a cost and an expected return.
2. Constraints:
   - The total cost of selected stocks should not exceed the available budget.
   - Include risk factors to ensure that the selected stocks provide the best return.

4.Objective

The objectives of this project are:

- **To create a tool for optimizing stock portfolios** based on cost, expected return, and risk.
- **To implement a Dynamic Programming approach** to solve the portfolio selection problem.
- **To develop a user-friendly interface** for inputting stock data, budget, and risk preferences.
- **To analyze and present the results** of portfolio optimization in a clear, understandable format.
- **To ensure that the solution can handle multiple stock options** and different budget constraints.

5.Methodology / Planning of work

1. **Stock Selection:**
    - Gather data on different stocks, including their cost and expected return.
    - Store this data in a proper format for easy access.
2. **Dynamic Programming Algorithm:**
    - Implement a dynamic approach algorithm that solves the portfolio optimization problem.
    - The algorithm will iterate through possible stock selection.
    - The objective is to maximize the total expected return while staying within the budget.

3. **Web Interface:**
    - Develop a simple web interface where users can input their budget, Stock value and their expected return.
    - Display the optimized result after running the dynamic approach algorithm.
4. **Testing and Evaluation:**
    - Test the tool with different stock combinations and their budgets.
    - Evaluate the performance and accuracy of the portfolio optimization solution.
5. **Documentation:**
    - Prepare a detailed documentation for the system, including how to use the tool and a description of the algorithm used.

6.Facilities Required for the Proposed Work

To develop and run the Smart Stock Portfolio Optimizer tool, several resources are required. These include software, hardware, and additional tool that help in development, deployment and testing. Below is the detailed explanation of the required facilities.

1. Software Requirements

These are the essential software components needed to build and run the project:

Frontend Technologies (For user Interface Development)

- **HTML***(Hyper Text Markup Language): Used to structure the web pages and define the content layout.
- **CSS*(**Cascading Style System): Helps in designing the look and feel of the webpage, including colors, fonts, and layouts.

Backend Technology (For Handling Logic and Data Processing)

- Used for interactivity, handling user inputs and communicating with the backend using APIs.
- **JavaScript*** for the backend code of Dynamic Programming to solve the knapsack

Development Environment & Tool

- Visual Studio Code **(VS Code)** *: A lightweight but powerful code editor for writing and debugging the frontend and backend code.

2. Hardware Requirements

A well-equipped system is necessary to run and develop the project efficiently. The minimum hardware requirements are:

- **Computer/Laptop**: A system with a minimum of 4GB RAM (recommended 8GB or more) for smooth development and execution.
- **Processor**: At least an Intel i3 or AMD Ryzen 3 (higher is better) to handle computations efficiently.
- **Storage**: At least 20GB of free space to accommodate the project files, database, and development tools.
- **Internet Connection**: Required for downloading dependencies, fetching external stock data (if applicable), and deploying the project.
- **Web Browser**(Google Chrome, Edge,etc.): To access and test the web application.

3. Additional Resources (Optional)

Version Control and Collaboration

- **Git:** Used for tracking changes in the project code and collaborating with team members.
- **GitHub:** A cloud-based repository where the project's source code can be stored, shared, and managed.

7.References

Web Technologies:

HTML: MDN Web Docs. (n.d.). *HTML: Hypertext Markup Language*. Mozilla Developer Network. https://developer.mozilla.org/en-US/docs/Web/HTML

CSS: MDN Web Docs. (n.d.). *CSS: Cascading Style Sheets*. Mozilla Developer Network. https://developer.mozilla.org/en-US/docs/Web/CSS

JavaScript: MDN Web Docs. (n.d.). *JavaScript: The Programming Language of the Web*. Mozilla Developer Network. https://developer.mozilla.org/en-US/docs/Web/JavaScript

Development Environment & Tool:

Visual Studio Code (VS Code): Official site of VS code (n.d.). Visual Studio Code documentation: https://code.visualstudio.com/docs

Version Control and Collaboration:

Git: Official site of Git (n.d.). Git Documentation: https://git-scm.com/doc

GitHub: Official site of GitHub (n.d.). GitHub Documentation (GitHub Docs): https://docs.github.com/en

Others:

GeeksForGeeks: Geeksforgeeks webpage (n.d.). 0/1 Knapsack Problem – GeeksforGeeks https://www.geeksforgeeks.org/0-1-knapsack-problem-dp-10/