Project Report

Portfolio Optimization

SUBMITTED IN THE PARTIAL FULFILLMENT REQUIREMENT

FOR THE AWARD OF DEGREE OF

Bachelor of Technology

(COMPUTER SCIENCE and ENGINEERING)

SUBMITTED BY

AKSHIT WADHWA (230784)

JHALAK KAPILA (230576)

SAI TEJASWINI (230609)

(UNIVERSITY ROLL NO. 1232434)

UNDER THE SUPERVISION OF

DR HIRDESH KUMAR PHARASI

SCHOOL OF ENGINEERING AND TECHNOLOGY

BML MUNJAL UNIVERSITY

Gurugram, Haryana - 122413

May 2025

CANDIDATE'S DECLARATION

We hereby certify that we have worked on project entitled, "**Portfolio Optimization**", in partial fulfillment of requirements for the award of Degree of **Bachelor of Technology** in name of the department at **BML Munjal University**, having University Roll No.1232434, is an authentic record of our own work carried out during a period from January, 2025 to May, 2025 under the supervision of DR HIRDESH KUMAR PHARASI.

Akshit Wadhwa (230784)

Jhalak Kapila (230576)

Sai Tejaswini (230609)

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

DR. HIRDESH KUMAR PHARASI

Assistant Professor

ABSTRACT

This project aims to analyze the performance of N distinct stock portfolios constructed from the Nifty 50 index constituents over 10 years (2014–2024). Each portfolio consists of 10 carefully selected stocks representing diversified sectors and betas. These may or may not contain common stocks across all portfolios to observe the impact of overlapping holdings. The primary objective is to evaluate and compare the portfolios during different market conditions—classified as "normal" and "crash" periods—by breaking the entire time into 120 non-overlapping frames of 20-day epoch. The normal period is viewed as low volatility or stable returns, while the crash period is one of heightened volatility and negative returns.

The financial performance of a portfolio is tracked using cumulative return, average daily return, and volatility, among other benchmarks. These metrics form the basis for the risk returns for each portfolio that is tested under various market conditions. This project is centered on the implementation of Python programming and its extracts through frameworks such as yfinance, where data is pulled, with further processes carried out in pandas and Numpy, culminating in visualization through matplotlib and seaborn. As applied to EDA, classification of market conditions around Nifty 50 index movements and portfolios were risk-adjusted between static and dynamic phases, all hypotheses were tested using empirical statistical methods Disclaimer analysis.

Through this analysis, the study aims to provide insights into the importance of diversification, portfolio construction strategy, and the effects of market cycles on investment performance. We have also used factors such as Sharpe Ratio and Cumulative Returns. This work is relevant to individual investors, financial analysts, and researchers looking to understand how different portfolios respond to market volatility and how data-driven techniques can aid in smarter investment decisions.

ACKNOWLEDGEMENT

We are highly grateful to **DR. HIRDESH KUMAR PHARASI, ASSISTANT PROFESSOR**, BML Munjal University, Gurugram, for providing supervision to carry out the project from January-May 2025.

Dr. Hirdesh Kumar Pharasi has provided great help in carrying out our work and is acknowledged with reverential thanks. Without the wise counsel and able guidance, it would have been impossible to complete the training in this manner.

We would like to express our sincere thanks to **DR. HIRDESH KUMAR PHARASI,** for stimulating us from time to time. We would also like to thank the entire team of BML Munjal University. We would also thank our friends who devoted their valuable time and helped us in all possible ways towards successful completion.

Akshit Wadhwa (230784)
Jhalak Kapila (230576)
Sai Tejaswini (230609)

LIST OF FIGURES

Figure No.	Figure Description	Page No.
fig 1.1.	10-Year Cumulative Returns of 4 Portfolios (Equal-Weighted)	10
fig 1.2.	Stock Market Trend with Monthly Highlights (Jan–Mar 2025)	12
fig 1.3.	Volatility Comparison: INFY vs AXISBANK (Nov 2024 – Mar 2025)	16
fig 1.4.	Reliance Stock Closing Prices – February 2024	16
fig 1.4	Stock Closing Prices with Monthly Highlight & Highest Point (Jan–Mar 2025)	16
fig 1.5	This graph displays the cumulative returns of **Portfolio 4** calculated over **rolling 20-day windows** from **2014 to 2024**.	18

LIST OF TABLES

Table No.	Table Description	Page No.
1	Literature Review: Comparison	7

LIST OF ABBREVIATIONS

Abbreviation	Full Form
CAGR	Compound Annual Growth Rate
EDA	Exploratory Data Analysis
NIFTY	National Stock Exchange Fifty Index
ROI	Return on Investment
MDD	Maximum Drawdown
Std.Dev	Standard deviation
API	Application Programming Interface
CSV	Comma-Separated Values
NSE	National Stock Exchange
BSE	Bombay Stock Exchange
OLS	Ordinary Least Squares
CRSP	Center for Research in Security Prices
IR	Information Ratio
YTD	Year-To-Date
VIX	Volatility Index
NAV	Net Asset Value
DCF	Discounted Cash Flow
GDP	Gross Domestic Product

TABLE OF CONTENT

Contents **Page No.**

Candidate's Declaration **i**

Abstract **ii**

Acknowledgement **iii**

List of Figures **iv**

List of Tables **v**

List of Abbreviations **vi**

1 Introduction to Organisation **1**

2 Introduction to Project **2**

2.1 Overview 2
2.2 Existing System 2
2.3 User Requirement Analysis 2
2.4 Feasibility Study 2

3 Literature Review **3**

3.1 Comparison 4
3.2 Objectives of Project. 4

4 Exploratory Data Analysis **5**

4.1 Dataset 5
4.2 Exploratory Data Analysis and Visualisations 5
4.3 Related Sections 5

5 Methodology **6**

5.1 Introduction to Languages (Front End and Back End) 6
5.2 Any other Supporting Languages/ packages 6
5.3 User characteristics 6
5.4 Constraints 6
5.5 Starting aim 6
5.6 Database design 6
5.7 Table Structure 6
5.8 Assumptions and Dependencies 6

5.9 ML algorithm discussion 6
5.10 Implementation of Algorithm with Screen Shots/ Figures (Each Figure must be numbered and Description of Figure must be provided) 6

6 Results **7**

7 Conclusion and Future Scope **8**
7.1 Conclusion 8
7.2 Future Scope 8

Chapter 1

Introduction to Organisation

In 2014, BML Munjal University (BMU) was established in Gurugram, Haryana, which aims to redefine higher education in India. Named after the founder of the Hero Group, Dr. Brijmohan Lall Munjal. BMU is a part of the Hero Group and continues in the group's tradition of valuing innovative strengths, with BMU's character being manifested in its culture of education. The university is committed to its distinctive student-centered education, which prepares students for lives of leadership, service, and a life well-lived. The courses are designed to include the theory but have an equal emphasis on the practical to make students feel confident and able to solve problems. BMU provides undergraduate, post-graduate, and doctoral programmes in streams such as engineering, management, law, and economics through a multi-disciplinary approach. The campus offers modern laboratories, technology-enabled classrooms, a contemporary library, and common areas for creativity & teamwork. Robust industry collaborations — in particular, with the Hero Group — facilitate student internships, live projects, and placements.The philosophy of learning at BMU and its experiential education approach is closely tied to programs such as the Practice School, where curricular learning and industrial learning are brought together. The university has a further emphasis on research and innovation, which is complemented by its global partnerships, particularly with Imperial College London, which maintains both staff and student opportunities for international exchange and collaboration. BMU was central in facilitating this project through providing the academic setting and resources that would keep it on course. Its emphasis on holistic education and skill development directly supports the objectives of this work, guaranteeing that the project makes a significant contribution to both academia and society at large.

Chapter 2

Introduction to Project

2.1 Overview

This project seeks to use companies within the Nifty 50 stock index, which is made up of the first 50 companies on the National Stock Exchange of India, to develop and compare the performance of a few diverse portfolios of stocks. Each of the portfolios contains ten stocks chosen from different sectors like technology, finance, energy, pharma, and consumer goods, which are used in each of the four portfolios developed. Each portfolio should have a good blend of different sectors to provide diversification and less exposure to company-specific risk in an effort to better achieve diversification because it is a means of mitigating irrational risk and improving return stability. The analysis relates to ten years because we have the ability to look back on how these portfolios performed over different periods of the Indian stock market, including stable and prosperous periods, drastic and unexpected drops, like the drops induced by the COVID-19 pandemic, the rise in global interest rates, and global military conflicts.

One of the main goals of the project is to ascertain how each portfolio performs relative to each other for various market states, specifically comparing performance from normal to crash states. Though performance is discussed heavily in the literature, there are a couple of ways to quantify performance in finance from an analytical perspective which are variability (or risk, in essence variability in return), average daily returns (something that would help measure this factor over time), and cumulative returns (i.e., the cumulative overall profit or loss on an investment). The toolset is basically Python since it can be considered fairly structured and is a data focused approach. The project carried out using Python methods, provides an organized way to analyze the data with establishing correlations and findings of meaning. Libraries like pandas allows one to format the data, yfinance allows one to source historical stock prices, and although there are other libraries, the visualizations produced are considered an appreciable learning tool from matplotlib and seaborn.

By examining the full 10-year period in 120 non-overlapping 20-day windows, we conduct granular, time-segmented evaluations of each portfolio. We have also examined the daily returns for each portfolio, and metrics such as average daily return and volatility. Each of these windows is categorized as either "normal" or "crash", depending on the Nifty 50 index performance in that window (for example, a crash is identified if the index declines more than 5% in that 20-day period). This type of categorization allows us to examine how portfolios perform differently in relatively normal versus adverse market conditions, with the objective to highlight different levels of resilience, risk, and recovery across portfolios.

2.2 Existing System

There are a myriad of financial models and frameworks for analytical systems and portfolio management, revealing stock actions, risk metrics and asset allocation strategy. Academic models, investment systems, machine learning - this is all part of the domain.

Traditional ways of managing portfolios, like Markowitz's Modern Portfolio Theory (MPT), are frequently used as foundations for portfolio decision-making. The aim of these processes is to create the best-possible portfolio, and usually map out what is termed an efficient frontier when it is plotted. An efficient frontier is a curve that contains portfolios with best-expected return for a given risk tolerance (to the right of the singularity - should you not read risk as value?). Financial analysts and experts use specialized software utilizing this model to validate their thinking when selecting assets and constructing diversified portfolios. Some well-known trading platforms like Bloomberg Terminal, Morningstar Direct, and Yahoo Finance provide sophisticated analytical tools to help capture stock performance, risk metrics, portfolios appear and past session trends in an interactive style. Providing current reporting, charting and historical analysis, they still remain focused on professional investors, and struggle with actual performance evaluation logic flexibility; such as dividing returns into user defined periods (crash OR normal) or aggregating data programmatically at a larger-scale level.

In the academic and fintech space there has been a proliferation of analytical frameworks that employ Python to analyze a portfolio. Libraries like Quantopian (now part of Robinhood), pyfolio, and bt have given users the ability to backtest investment decisions using historical data, assess and analyze their collective decisions based on a variety of risk-return measures. While these platforms offer enhanced visual displays and flexibility, many of the options focus on either long/short strategies or algorithmic trading, but do not necessarily have an explicit method of dealing with the way their portfolio behaves in falling markets.

Another system to mention is R's PerformanceAnalytics package, which encompasses calculation of an extensive set of performance and risk measures, such as rolling volatility, drawdown and return distributions. However, both R based and Python systems require a degree of technical expertise as well as lack of automatic domain-specific market state labeling systems (e.g., index thresholds for automatic crash detection), which this project deliberately operationalizes.

In short, even though most of the current systems have capabilities of satisfied portfolio monitoring and performance analysis, none of the systems offer a configurable, segment (portfolio) true evaluations of diversified portfolios, underspecified crash and normal phases, with trivial metrics.This project bridges this gap by providing a Python-driven, open, and modular system for segment-wise analysis over ten years to facilitate better insights into portfolio stability and resilience.

2.3 User Requirement Analysis

The project is designed to meet the needs of users such as retail investors, financial analysts, and academic researchers. The key user requirements include:

1. Functional Requirements.

These define what the system should do:

- **Portfolio Creation**: The System should allow users to define multiple portfolios with Nifty 50 stocks.
- **Data Retrieval**: Fetch historical stock price data from 2014 to 2024 using reliable sources like Yahoo Finance.
- **Performance Calculation**: Compute cumulative return, volatility, and average daily return for each portfolio.
- **Crash vs Normal Analysis**: Classify market windows into crash and normal periods and analyze accordingly.
- **Visualization**: Display line graphs, bar charts, and comparative plots for portfolio performance.
- **Report Generation**: Generate summary tables and exportable insights for documentation.

2. Non-Functional Requirements

These define how the system performs its functions:

- **Usability**: Interface should be simple and understandable for users with basic finance knowledge.
- **Scalability**: Should support analysis for more portfolios or longer time ranges if needed in future.
- **Performance**: Analysis and visualizations should execute within a few seconds for optimal user experience.
- **Reliability**: Ensure accurate data handling and computations even with large datasets.
- **Maintainability**: The code should be modular and well-documented for future updates.

3. Validation Requirements

These ensure that the system fulfills the intended use correctly:

- **Accuracy Check**: Verify computed metrics (returns, volatility) with known financial formulas.
- **Data Integrity**: Ensure no missing or corrupted values in fetched stock data before processing.
- **Visualization Validation**: Cross-check graph outputs with underlying data.
- **Unit Testing**: Write tests for core functions like returns calculation, volatility estimation, and classification logic.
- **User Feedback**: Collect feedback from at least one domain expert or end-user to validate usability and correctness.

2.4 Feasibility Study

1. Technical Feasibility

The project is technically feasible using tools like Python, Pandas, NumPy, Matplotlib, and yfinance for data analysis. These tools are well-suited for financial time-series data, portfolio simulations, and visualizations.

2. Economic Feasibility

The project uses open-source tools and publicly available data, making it cost-effective with no need for premium software or APIs.

3. Operational Feasibility

The system can be used by users with basic financial knowledge. With proper documentation and UI enhancements (e.g., through a web dashboard), it can be operationally viable for broader audiences.

4. Legal and Ethical Feasibility

The project does not violate any data privacy laws as it uses publicly accessible stock market data and focuses purely on analysis.

Chapter 3

Literature Review

Understanding the dynamics of stock markets and portfolio performance across different time periods—especially during volatile or crash phases—has been a critical task for financial analysts and researchers. Identifying historical patterns and drawing insights for optimized investment strategies requires not only domain expertise but also computational support. Given the vast amount of financial data generated daily, manually analyzing and drawing conclusions has become a labor-intensive and complex task.

With the advent of advanced computational tools and financial modeling techniques, various methods have been developed for evaluating portfolios using both historical and real-time data. Historical stock prices, sector-wise indices, and market benchmarks like the Nifty 50 serve as rich sources of time-series data.

Techniques such as rolling window analysis and time-series segmentation help in understanding how a portfolio's risk and return metrics change over time [6]. Objective metrics like cumulative return, average daily return [7], and standard deviation (volatility) [8] are commonly used to assess investment efficiency. [5]

Clustering methods help group stocks by behavior while developing rolling periods for performance tracking allows investors to obtain a more real-time view of changing behavior.

Unlike traditional static analysis, this approach allows assigning a portfolio to multiple performance states over time, similar to how topic modeling allows documents to align with multiple topics. This flexibility enhances the understanding of investment behavior across changing market scenarios.

3.1 Comparison

Author(s)	Year	Focus Area	Methodology Used	Findings / Limitations
Markowitz	1952	Portfolio Theory	Mean-Variance Optimization	Introduced concept of efficient frontier but assumes normality [1]
Fama & French	1993	Asset Pricing	Multi-Factor Model	Better explains stock returns but needs more data inputs [2]
DeMiguel et al.	2009	Portfolio Diversification	Out-of-sample performance analysis	Simple equal-weighted portfolios can outperform optimized ones [3]
Choudhury et al.	2020	Indian Stock Market Analysis	Rolling Window Volatility Analysis	Lacked granularity in crash period behavior [4]
Sharma & Mehta	2022	NIFTY 50 Portfolio Optimization	Sharpe Ratio, VaR, Rolling Metrics	Focused only on Sharpe ratio, ignored sectoral diversification [5]

3.2 Objectives of Project

The primary objective of this project is to construct and evaluate **four distinct stock portfolios** from the **NIFTY 50 index**, each consisting of **10 stocks** diversified across sectors. These portfolios are analyzed over 10 years (2014–2024), segmented into **normal** and **crash** periods. The aim is to address gaps observed in existing literature, especially the **lack of temporal behavior analysis**,

rolling window evaluations, and **multi-metric assessment** of performance.

Key Objectives:

1. **To construct sector-diversified portfolios** using NIFTY 50 stocks with two common stocks across all portfolios.
2. **To analyze portfolio performance** using cumulative return, volatility, and average daily return.
3. **To classify market periods** into normal and crash based on NIFTY 50 volatility.
4. **To apply rolling window analysis** (20-day non-overlapping windows) for better temporal resolution.
5. **To compare performance trends** across all portfolios during crash vs. normal periods.
6. **To visualize portfolio behavior** using line graphs, bar charts, and comparative return plots.
7. **To provide data-driven insights** that may guide investors toward better portfolio construction strategies.

Chapter 4:

Exploratory Data Analysis

4.1 Dataset

The dataset used in this Colab notebook is historical stock price information for the components of the Nifty 50 index that was obtained from Yahoo Finance using the yfinance library.It is not a named csv dataset as the prices of the stocks keep on varying with time.The time range we have taken spans ten years, from January 1, 2014, to January 1, 2024. The initial dataset for the specified stocks is processed to remove columns with a significant amount of missing values. Subsequently, daily percentage returns are calculated from the adjusted closing prices of the remaining stocks, forming the basis for constructing and analyzing the performance and volatility of randomly selected portfolios over defined time windows.

1.Data Collection Process:

- **Identification of Assets:** A list of 50 stock tickers belonging to the Nifty 50 index is defined.
- **Specification of Timeframe:** The data collection period is defined by the start date ("2014-01-01") and end date ("2024-01-01").
- **Data Download:** The yfinance.download() function is used to retrieve historical stock data for the specified list of tickers over the defined period. This function typically downloads various data points like Open,High, Low, Close, Adjusted Close, and Volume.

2.Data Preprocessing Pipeline:

- **Handle Missing Data:** Remove columns with more than 5% missing data.
- **Calculate Daily Returns:** Compute daily percentage changes and drop the initial NaN row.
- **Construct Portfolios:** Randomly select and group stocks into four portfolios.
- **Calculate Portfolio Returns:** Determine the daily mean return for each portfolio.
- **Prepare for Window Analysis:** Organize portfolio returns for processing in time windows.

4.2 Exploratory Data Analysis and Visualisations

We have first divided the Nifty Fifty into 4 different virtual portfolios, where each portfolio can may or may not contain repeated stocks.

```
# Create a DataFrame to display the selected portfolios
portfolio_df = pd.DataFrame(dict([(k, pd.Series(v)) for k, v in portfolios.items()]))

print("Selected Stocks in Each Portfolio:\n")
print(portfolio_df)
```

```
Selected Stocks in Each Portfolio:

                  Portfolio_1               Portfolio_2               Portfolio_3  \
0          (Volume, M&M.NS)     (Open, BAJAJ-AUTO.NS)  (Volume, ASIANPAINT.NS)
1         (High, HDFCBANK.NS)   (Open, ASIANPAINT.NS)          (Open, SBIN.NS)
2       (Low, ULTRACEMCO.NS)   (Close, EICHERMOT.NS)       (Low, RELIANCE.NS)
3           (Open, MARUTI.NS)          (Low, WIPRO.NS)       (Low, JSWSTEEL.NS)
4        (Open, SUNPHARMA.NS)       (Low, ADANIENT.NS)         (Volume, UPL.NS)
5       (Open, HINDUNILVR.NS)           (Low, NTPC.NS)       (High, HCLTECH.NS)
6   (Volume, BAJFINANCE.NS)  (Volume, HEROMOTOCO.NS)     (High, ICICIBANK.NS)
7       (Volume, ADANIENT.NS)   (High, TATACONSUM.NS)  (Volume, COALINDIA.NS)
8            (Close, ONGC.NS)   (Volume, AXISBANK.NS)     (Close, SHREECEM.NS)
9   (Volume, HINDUNILVR.NS)          (Open, BPCL.NS)         (High, CIPLA.NS)

               Portfolio_4
0       (Low, COALINDIA.NS)
1          (Open, GRASIM.NS)
2      (Low, INDUSINDBK.NS)
3     (High, BAJAJ-AUTO.NS)
4           (High, BPCL.NS)
5     (High, ASIANPAINT.NS)
6              (Low, LT.NS)
7     (Open, HEROMOTOCO.NS)
8   (Volume, EICHERMOT.NS)
9          (Low, MARUTI.NS)
```

After that we have formed the time series graph of each portfolio and see the cumulative Returns over the period of 10 series.

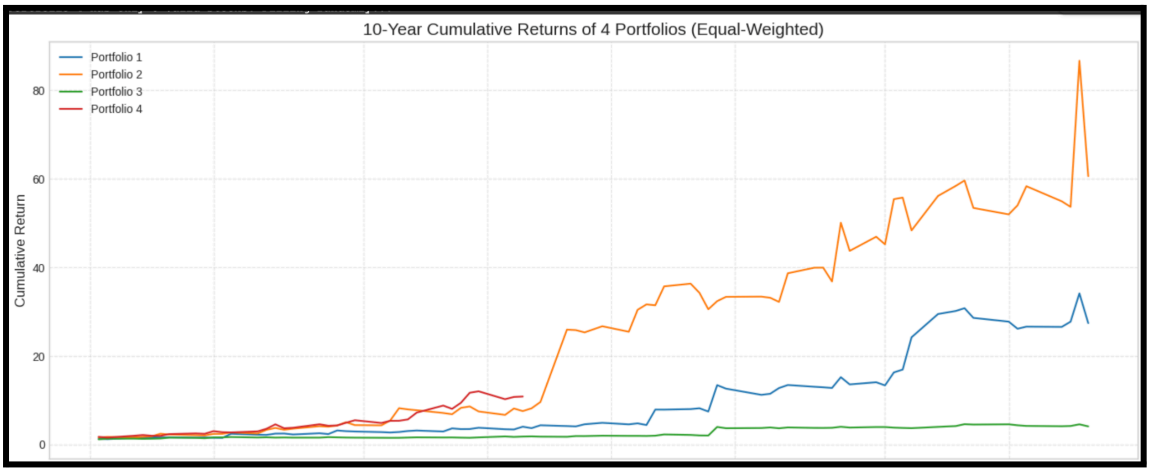


fig 1.1. This graph compares the cumulative return growth of four different portfolios over 10 years.

Chapter 5

Methodology

The technique adopted in this project is gathering the past stock prices of Nifty 50 components over ten years, data cleansing by handling the stocks with zero value for the time duration we took into consideration, and calculating the daily return and estimated return. Then, four randomly chosen, distinct portfolios of 10 stocks each are constructed, and their average daily return is determined. We employed the understanding of MPT and its impact on the strategy of risk-return and diversification by investors. The project then analyzes these returns on the portfolios by partitioning the data into non-overlapping 20-day intervals. Inside each window, it calculates cumulative portfolio return, estimates market volatility in the form of standard deviation of daily mean returns over portfolios, and decides the label for the market regime as 'Crash' or 'Normal' based on thresholds given for volatility and total market return. The final result is a nicely formatted summary of these measurements and regimes of markets for each time window, which can be used for examination of portfolio behavior for varying market environments.

5.1 Introduction to Languages (Front End and Back End)

- **Python:** supported by several powerful libraries. Python is the core language orchestrating data downloading, manipulation, calculations, and analysis. The code also implicitly relies on the functionality provided by specialized Python libraries like pandas for data handling, numpy for numerical operations, and yfinance for accessing financial data, enabling complex financial computations and data processing tasks to be performed efficiently within the Python environment.

5.2 Any other Supporting packages

- **Python:** The primary programming language used for the entire project.
- **pandas:** A powerful library for data manipulation and analysis is used to process stock data, calculate returns and create data frames.
- **numpy**: A fundamental library for numerical operations, likely used implicitly by pandas and potentially for array manipulations.
- **yfinance**: A specific library for downloading financial data from Yahoo Finance.
- matplotlib.pyplot: A plotting library, though no plots are generated in the provided code snippet, it is imported and suggests potential for visualization.
- **cvxpy:** A library for convex optimization, imported but not used in the provided code snippet, indicating a potential future step for optimal portfolio allocation.
- **random:** Used for shuffling the list of stocks to create random portfolios.
- **IPython and IPython.display:** Libraries related to the Jupyter/Colab environment for displaying outputs and working with cells.

5.3 User characteristics

- **Interest in Stock Market and Finance:** The project's topic strongly suggests the user is interested in financial analysis, particularly portfolio management and market regimes.
- **Python familiarity:** The code is provided in Python so we can assume that the reader has somewhat of some familiarity with the language.
- **Financial concepts familiarity:** The reader likely has at least a minimal understanding of financial concepts, such as: market crashes, diversification, volatility, stock returns, etc.
- **Using Google Colab or Jupyter Notebooks**: The code contains %% [markdown] cells and IPython imports suggesting that the user is working in an interactive notebook environment.

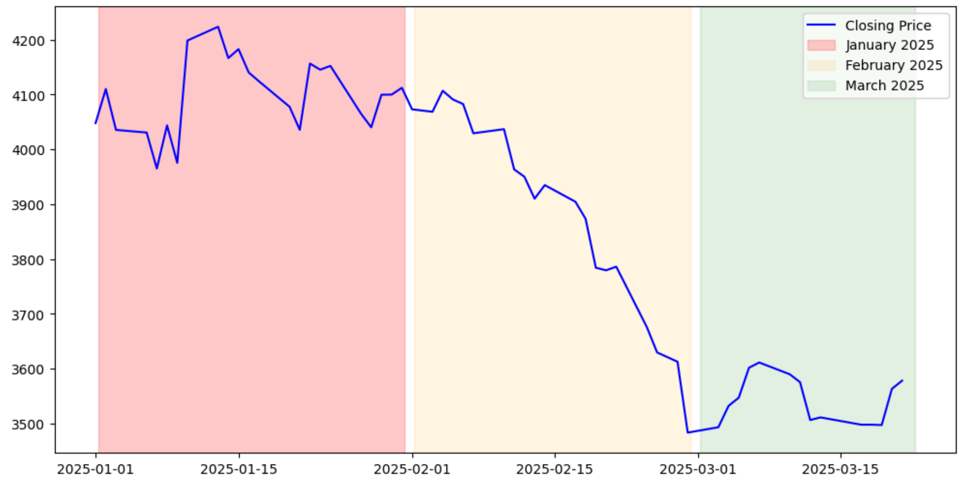
5.4 Constraints

- **Data Availability:** We used the **yfinance** library which gave us the overall data of all the stocks present in the National Stock exchange(NSE).
- **Fixed Time Window:** The analysis is conducted using a fixed, non-overlapping 20-day window size.
- **Market Regime Thresholds:** The classification of Crash' and 'Normal' market regimes is based on specific, predefined volatility and return thresholds.
- **Random Portfolio Composition:** The portfolios are created through random selection, meaning the specific results are dependent on the particular random sets of stocks chosen.

5.5 Starting Aim

Our starting goal was to create efficient limits with the highest expected return or lowest risk for a given portfolio.
Getting stock prices for certain stock groups was the beginning of how we did this project by studying terms such as historical price, closing price and finding out about different trends in the stock market.

fig 1.2.



stock/index from January to March 2025, with each month shaded in a different color.
Then we had moved on to make portfolios which consisted of 6 stocks, which contained 2 same stocks in all the 4 portfolios. We calculated the mean return of a stock as well as the mean return of a portfolio for better understanding.

1.2 What is the mean return for a stock vs a mean return of a portfolio.

–The mean return of stock is the average return of a single stock over time

–It can also be called as how much a stock returns on a monthly basis.

1.3 A mean return of a portfolio can be calculated using the mean return on an investment given the historical returns or the probable rates of returns under giving return scenarios.

– The tool given and used by Investors to weigh investment decsions is known as mean varience analysis.

– The Varience shows how spread out the returns of a specific security on a daily or weekly basis.

– The expected return is a probability expressing the estimated return of the investment in the security.

The optimal portfolio will have high return and low variance

5.6 Database design

- In our database, the main **entities** we took were the stock, name, and the symbol it will be represented by. Since the stocks are all from the Nifty Fifty stocks of India, an important aspect is that the country remains the same for all. We have also used the expected return as an entity that tells us the approximate daily return of a portfolio.
- Next was the user elements of who will be the one acting or changing the element of the database, for that we have taken characters such as email, username and the user_id.
- To represent each separate portfolio, we have taken the entities such as portfolio_id, optimal return, mean, volatility, etc as the main points. The parameters, such as expected output, cumulative returns, etc.

5.7 Assumptions and Dependencies

- **Assumptions:**
    - Complete and accurate historical stock data from yfinance.
    - The selected 20-day window size and market regime thresholds (volatility> 0.015 and return < -0.03) are appropriate for the analysis.
    - Random portfolio selection gives a representative method of investigating diversification.
- **Dependencies:**
    - A working Python environment with the required libraries (pandas, numpy, yfinance, etc.) installed.
    - A stable internet connection to download data, as we can select the stocks we need, and depending on the stock size, the time to download increases.
    - The availability and functionality of the Yahoo Finance data source.

5.8 ML algorithm discussion

The ml algorithm we had used is firstly segregation the stocks in each portfolio based on randomness.

```
[2] # Firstly we are going to load all the libraries which we need
    import pandas as pd
    import numpy as np
    import yfinance as yf
    import matplotlib.pyplot as plt
    import cvxpy as cp
```

– Now we are going to load the set of the nifty fifty stocks

```
#importing the nifty fifty stocks
nifty50_Stocks = [
    "RELIANCE.NS", "TCS.NS", "INFY.NS", "HDFCBANK.NS", "ICICIBANK.NS", "KOTAKBANK.NS",
    "HINDUNILVR.NS", "SBIN.NS", "LT.NS", "ITC.NS", "AXISBANK.NS", "BAJFINANCE.NS",
    "ASIANPAINT.NS", "MARUTI.NS", "SUNPHARMA.NS", "ULTRACEMCO.NS", "NESTLEIND.NS", "WIPRO.NS",
    "NTPC.NS", "POWERGRID.NS", "COALINDIA.NS", "TITAN.NS", "TECHM.NS", "GRASIM.NS",
    "BHARTIARTL.NS", "HCLTECH.NS", "DRREDDY.NS", "ADANIENT.NS", "BAJAJFINSV.NS", "TATASTEEL.NS",
    "HDFCLIFE.NS", "SBILIFE.NS", "HEROMOTOCO.NS", "DIVISLAB.NS", "INDUSINDBK.NS", "JSWSTEEL.NS",
    "CIPLA.NS", "EICHERMOT.NS", "UPL.NS", "BPCL.NS", "BRITANNIA.NS", "SHREECEM.NS",
    "ONGC.NS", "APOLLOHOSP.NS", "BAJAJ-AUTO.NS", "M&M.NS", "HINDALCO.NS", "TATACONSUM.NS",
    "TATAMOTORS.NS", "SBICARD.NS"
]

# Defining the time for which we need the data for

data_stocks = yf.download(nifty50_Stocks, start="2014-01-01", end="2024-01-01")


print(data_stocks.head())
```

```
YF.download() has changed argument auto_adjust default to True
[*********************100%***********************]  50 of 50 completedPrice           Close
Ticker     ADANIENT.NS APOLLOHOSP.NS ASIANPAINT.NS AXISBANK.NS BAJAJ-AUTO.NS
Date
2014-01-01   37.933197    927.892883    459.297363  247.552414   1448.022827
2014-01-02   36.041824    890.798279    446.246796  243.740112   1434.919922
2014-01-03   35.053795    911.284058    449.877075  241.556168   1431.521606
2014-01-06   35.576031    905.301147    450.934021  240.674911   1426.046631
2014-01-07   34.150455    892.281921    450.290741  236.450699   1427.292480

Price                                                                         \
Ticker     BAJAJFINSV.NS BAJFINANCE.NS BHARTIARTL.NS    BPCL.NS BRITANNIA.NS
Date
2014-01-01     75.461983    151.039139    286.472443  31.349638   399.551544
2014-01-02     72.309883    150.933365    278.796448  30.474327   395.527496
2014-01-03     72.404205    149.615616    279.347778  29.563105   398.855469
```

– Sometimes in some portfolios there are no null values present so we go on and clean those values.

– Time series analysis involves the study of the collected data points which were recorded over time. In financial terms it is also known as the historical price and the performance data of the financial assets.

One of the important feature of time series is trend analysis which helps in identifying long term movements or the patterns in the dataAnother thing the trend series helps us understand is the performance metrics

5.9 Implementation of Algorithm with Screen Shots/ Figures

From that we are going to Break the period into non-overlapping 20-day windows

```
window_results = []

for i in range(0, len(portfolio_returns) - 20 + 1, 20):
    window = portfolio_returns.iloc[i:i+20]

    # after creating a loop we have made

    cumulative = (1 + window).prod() - 1

    daily_mean = window.mean(axis=1)

    # Volatility of the window
    volatility = daily_mean.std()

    # Total return of the market (mean return of all portfolios)
    total_return = (1 + daily_mean).prod() - 1

    # Label as Crash if volatility is high and returns are sharply negative
    label = 'Crash' if (volatility > 0.015 and total_return < -0.03) else 'Normal'

    result = {
        'Start_Date': window.index[0],
        'Portfolio_1': round(cumulative['Portfolio_1'], 4),
        'Portfolio_2': round(cumulative['Portfolio_2'], 4),
        'Portfolio_3': round(cumulative['Portfolio_3'], 4),
        'Portfolio_4': round(cumulative['Portfolio_4'], 4),
        'Volatility': round(volatility, 4),
        'Market_Regime': label
    }

    window_results.append(result)

# Create DataFrame
windows_df = pd.DataFrame(window_results)
print(windows_df.head(10))

#plot graph
```

```
   Start_Date  Portfolio_1  Portfolio_2  Portfolio_3  Portfolio_4  Volatility  \
0 2014-01-02      -0.0176       1.1447      -0.0346       2.4261      0.0791
```

```
[ ] mean_returns = supposed_return.mean()
    variances = supposed_return.var()
```

```
score = mean_returns / variances

score
```

		0
Price	**Ticker**	
Close	**ADANIENT.NS**	1.447990
	APOLLOHOSP.NS	2.291628

4 Here we have made random portfolios from the nifty fifty

```
] from scipy.optimize import minimize

  # Optimizes portfolio weights to minimize risk for different target returns and plots the efficient frontier.
  def plot_efficient_frontier(mean_returns, cov_matrix, portfolio_name):
      target_returns = np.linspace(mean_returns.min(), mean_returns.max(), 50)
      risks = []

      for target in target_returns:
          def minimize_volatility(weights):
              return np.sqrt(np.dot(weights.T, np.dot(cov_matrix, weights)))

          constraints = [
              {'type': 'eq', 'fun': lambda x: np.dot(x, mean_returns) - target},
              {'type': 'eq', 'fun': lambda x: np.sum(x) - 1}
          ]
          bounds = [(0, 1)] * len(mean_returns)
          init_guess = [1 / len(mean_returns)] * len(mean_returns)

          result = minimize(minimize_volatility, init_guess, bounds=bounds, constraints=constraints)
          risks.append(result.fun)

      plt.plot(risks, target_returns, label=f"{portfolio_name}")
```

To clean the missing values–

```
[ ] # Now we need to remove the missing data in the stocks
    data_stocks = data_stocks.dropna(axis=1, thresh=len(data_stocks)*0.95)
    import random
```

Chapter 6

Result

1. This project was carried out in parts by building upon one goal after another. Firstly, during the first week we were able to initialise and understand the importance of historical data during a given time by importing the yfinance library.

```
[ ] # Fetch historical data for the last month
    last_month = tcs.history(start="2024-02-01", end="2024-03-01")
    print(last_month.head(5))

                                  Open         High          Low        Close  \
    Date
    2024-02-01 00:00:00+05:30  3715.216283  3797.787356  3700.676413  3748.429443
    2024-02-02 00:00:00+05:30  3768.756429  3875.106788  3765.790044  3857.503418
    2024-02-05 00:00:00+05:30  3873.744913  3911.675130  3853.418303  3864.311035
    2024-02-06 00:00:00+05:30  3887.361385  4036.067209  3880.553397  4022.548584
    2024-02-07 00:00:00+05:30  4039.082061  4041.027200  3962.200377  3971.391113

                            Volume  Dividends  Stock Splits
    Date
    2024-02-01 00:00:00+05:30  2363107        0.0           0.0
    2024-02-02 00:00:00+05:30  2826510        0.0           0.0
    2024-02-05 00:00:00+05:30  1691523        0.0           0.0
    2024-02-06 00:00:00+05:30  4474396        0.0           0.0
    2024-02-07 00:00:00+05:30  2124267        0.0           0.0
```

2. Next, we created plots for the selected plots for the selected stocks and analysed it over time

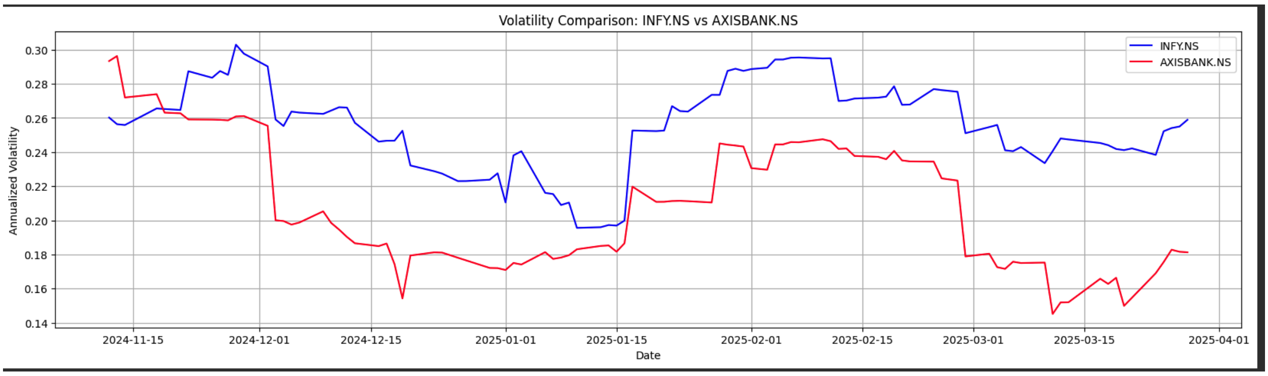


fig 1.3. This line chart compares the rolling annualized volatility of Infosys (INFY) and Axis Bank since (Nov 2024 – Mar 2025)

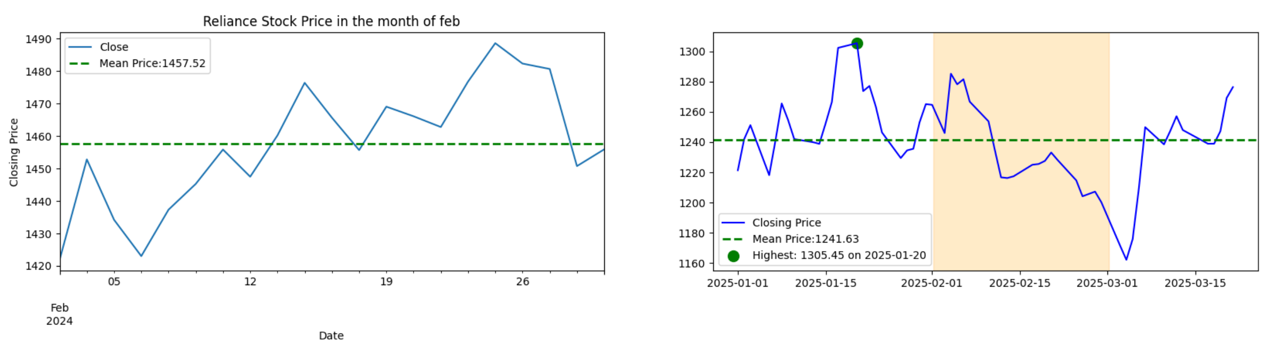


fig 1.4. The left panel shows Reliance's daily closing prices in February 2024 with the average market. The right panel displays closing prices with the highest point highlighted and February 2025 shaded

3. Then we had created the covariance matrix for taking stocks randomly vs when we had taken 2 stocks same in all the fields who had the highest score. Score is the mean returns divided by the variance.

2.1 After finding out the score of the all the nifty fifty , we see the top two score

```
[ ] top_2_stocks = score.sort_values(ascending=False).head(2).index.tolist()
    print(top_2_stocks)
```

```
[('Close', 'BRITANNIA.NS'), ('High', 'HINDUNILVR.NS')]
```

2.2 Here we have identified the top 2 stocks which have the highest score

```
📊 Portfolio 1 Covariance Matrix:
Price                         Close          High     Volume          Low  \
Ticker                 BRITANNIA.NS HINDUNILVR.NS HCLTECH.NS NESTLEIND.NS
Price  Ticker
Close  BRITANNIA.NS        0.000256      0.000059        NaN     0.000051
High   HINDUNILVR.NS       0.000059      0.000182        NaN     0.000046
Volume HCLTECH.NS               NaN           NaN        NaN          NaN
Low    NESTLEIND.NS        0.000051      0.000046        NaN     0.000194
       UPL.NS              0.000069      0.000030        NaN     0.000069
Close  HEROMOTOCO.NS       0.000069      0.000036        NaN     0.000031

Price                                 Close
Ticker                  UPL.NS HEROMOTOCO.NS
Price  Ticker
Close  BRITANNIA.NS   0.000069      0.000069
High   HINDUNILVR.NS  0.000030      0.000036
Volume HCLTECH.NS          NaN           NaN
Low    NESTLEIND.NS   0.000069      0.000031
       UPL.NS         0.000507      0.000075
Close  HEROMOTOCO.NS  0.000075      0.000327

📊 Portfolio 2 Covariance Matrix:
Price                        Close          High      Low          Open  \
Ticker                BRITANNIA.NS HINDUNILVR.NS   BPCL.NS ASIANPAINT.NS
Price Ticker
Close BRITANNIA.NS        0.000256      0.000059  0.000056      0.000029
High  HINDUNILVR.NS       0.000059      0.000182  0.000034      0.000050
Low   BPCL.NS             0.000056      0.000034  0.000427      0.000070
Open  ASIANPAINT.NS       0.000029      0.000050  0.000070      0.000275
High  UPL.NS              0.000052      0.000037  0.000088      0.000058
Low   LT.NS               0.000062      0.000039  0.000138      0.000077

Price                    High       Low
Ticker                 UPL.NS     LT.NS
Price Ticker
Close BRITANNIA.NS   0.000052  0.000062
High  HINDUNILVR.NS  0.000037  0.000039
Low   BPCL.NS        0.000088  0.000138
Open  ASIANPAINT.NS  0.000058  0.000077
High  UPL.NS         0.000397  0.000105
Low   LT.NS          0.000105  0.000303
```

Here we have the covariance matrix between these portfolio when 2 stocks were taken common in each one.

```
📊 Portfolio 1 (Portfolio 1) Covariance Matrix:
Price                          Low      Volume           Open  \
Ticker                COALINDIA.NS HCLTECH.NS INFY.NS INDUSINDBK.NS
Price  Ticker
Low    COALINDIA.NS       0.000319   0.000071     NaN      0.000119
       HCLTECH.NS         0.000071   0.000268     NaN      0.000069
Volume INFY.NS                 NaN        NaN     NaN           NaN
Open   INDUSINDBK.NS      0.000119   0.000069     NaN      0.000686
High   INDUSINDBK.NS      0.000075   0.000054     NaN      0.000391
Open   WIPRO.NS           0.000048   0.000089     NaN      0.000091
Close  BAJAJ-AUTO.NS      0.000049   0.000037     NaN      0.000038
Volume ADANIPORTS.NS           NaN        NaN     NaN           NaN
       ASIANPAINT.NS           NaN        NaN     NaN           NaN
       EICHERMOT.NS            NaN        NaN     NaN           NaN

Price                           High      Open         Close        Volume  \
Ticker                 INDUSINDBK.NS  WIPRO.NS BAJAJ-AUTO.NS ADANIPORTS.NS
Price  Ticker
Low    COALINDIA.NS         0.000075  0.000048      0.000049           NaN
       HCLTECH.NS           0.000054  0.000089      0.000037           NaN
Volume INFY.NS                   NaN       NaN           NaN           NaN
Open   INDUSINDBK.NS        0.000391  0.000091      0.000038           NaN
High   INDUSINDBK.NS        0.000525  0.000055      0.000069           NaN
Open   WIPRO.NS             0.000055  0.000277      0.000017           NaN
Close  BAJAJ-AUTO.NS        0.000069  0.000017      0.000255           NaN
Volume ADANIPORTS.NS             NaN       NaN           NaN           NaN
       ASIANPAINT.NS             NaN       NaN           NaN           NaN
       EICHERMOT.NS              NaN       NaN           NaN           NaN
```

But here we have the comparison when the stocks of each of the portfolio were taken in random from the y finance library

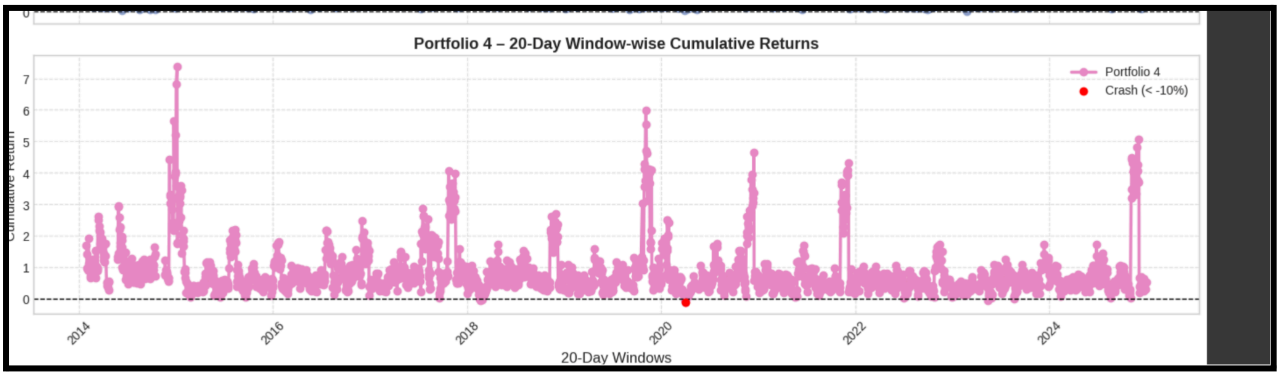


fig 1.5. This graph displays the cumulative returns of **Portfolio 4** calculated over **rolling 20-day windows** from **2014 to 2024**.

Chapter 7

Conclusion and Future Scope

7.1 Conclusion

Our project can effectively demonstrate how diversified and well-structured portfolios, developed from the study, vividly illustrate its long-term performance of well-structured and diversified portfolios made up of Nifty 50 constituents across different market conditions. By partitioning the ten-year time frame (2014–2024) into 120 20-day periods which do not overlap with each other and by distinguishing.
The research provides a proper insight into the performance of stock combinations in stable and unstable conditions in the time of a crash and also during normal periods.

Volatility, average daily return, and cumulative return were the three prime indicators used by the research to analyze the portfolios. It was noted that:

1. Sectorally diversified portfolios overall were more stable.
2. Overlapping common stocks had an influence on the stability of performance between the portfolios.
3. Although crash times had a material effect on returns, the composition in some portfolios allowed them to manage risk more effectively.

Python because of its large collections, yfinance, pandas, numpy, and matplotlib to mention a few, could efficiently pull, manipulate, and graph data. Increased availability and replicability of investment analysis may be realized through segmenting market states and evaluating performance using rolling windows.
This study highlights the value of evidence-based portfolio decisions and demonstrates the applicability of quantitative techniques in calculating investment return over different market conditions.

7.2 Future Scope

The present work demonstrates a basic methodology for analyzing stock portfolio performance under healthy and crash market conditions employing diversified NIFTY 50 stocks. Nevertheless, there are some areas where this study can be developed and enhanced in the future:
Employing More Sophisticated Metrics
Further research can encompass more performance metrics like Alpha/Beta, Sortino Ratio, Maximum Drawdown, and Sharpe Ratio to gain a wider insight.

- **Portfolio Optimization in Real Time**

Portfolio optimization based on dynamic real-time information could be assisted through the implementation of machine learning models like reinforcement learning or genetic algorithms.

- **Incorporating Macroeconomic Indicators**

One can look into the impact of such extrinsic elements like GDP growth, interest rate, and inflation on portfolio performance.

- **Extension to Global Indices**

The method can be further applied to consider portfolios by international indices such as the S&P 500, FTSE 100, or Nikkei 225 for comparison across the globe.

- **Backtesting with Transaction Costs**

Future models may incorporate transaction costs, taxes, and slippage to provide more realistic investment conditions.

- **Creation of an Online Tool**

To allow users to see and interact with customized portfolios based on their interests, an application or dashboard may be developed.

- **ESG and Ethical Investing Filters**

Investing can be aligned with socially responsible objectives by considering Environment, Social, and Governance (ESG) factors.

- **Predictive Modeling to Detect Crash Events**

Time-series prediction or classification models are applied to predict upcoming crash periods and rebalance portfolios accordingly.

Bibliography

1. Markowitz, H. (1952). *Portfolio Selection*. The Journal of Finance, 7(1), 77–91. https://www.math.hkust.edu.hk/~maykwok/courses/ma362/07F/markowitz_JF.pdf
2. Elton, E. J., & Gruber, M. J. (1995). *Modern Portfolio Theory and Investment Analysis*. Wiley.https://archive.org/details/modernportfoliot0000elto_z6s2
3. Reilly, F. K., & Brown, K. C. (2012). *Investment Analysis and Portfolio Management*. Cengage Learning. https://archive.org/details/investmentanalys0000reil_p1a6
4. Ang, A., & Bekaert, G. (2002). *International asset allocation with regime shifts*. Review of Financial Studies, 15(4), 1137–1187. https://academic.oup.com/rfs/article-abstract/15/4/1137/1568247
5. Campbell, J. Y., Lettau, M., Malkiel, B. G., & Xu, Y. (2001). *Have individual stocks become more volatile? An empirical exploration of idiosyncratic risk*. Journal of Finance, 56(1), 1–43. https://www.nber.org/papers/w7590
6. Lo, A. W. (2002). *The adaptive markets hypothesis: Market efficiency from an evolutionary perspective*. Journal of Portfolio Management, 30(5), 15–29.https://web.mit.edu/Alo/www/Papers/JPM2004.html
7. Tsay, R. S. (2010). *Analysis of Financial Time Series*. Wiley.https://archive.org/download/econometrics_books/Analysis%20of%20Financial%20Time%20Series%20-%20R.%20S.%20Tsay.pdf
8. Bali, T. G., Cakici, N., & Whitelaw, R. F. (2005). *Maxing out: Stocks as lotteries and the cross-section of expected returns*. Journal of Financial Economics, 99(2), 427–446.https://pages.stern.nyu.edu/~rwhitela/papers/max%20jfe11.pdf
9. https://www.pyquantnews.com/free-python-resources/portfolio-optimization-with-mpt-and-python