END-TERM EVALUATION

ALGORISK INSIGHTS

Mentors:

Moni Dutt
Rachit Choudhary
Prakhar Upadhyay

Introduction

AlgoRisk Insights

WEEK - 1

NUMPY, PANDAS, YFINANCE, SHARPE RATIO, MAXIMUM DRAWDOWN,

WEEK - 2

TECHNICAL INDICATORS AND SIGNAL GENERATION

WEEK - 3

MEAN REVERSION, STOP LOSS AND DENOISING

WEEK -4

PAIRS TRADING AND COINTEGRATION

WEEK - 1

Numpy:

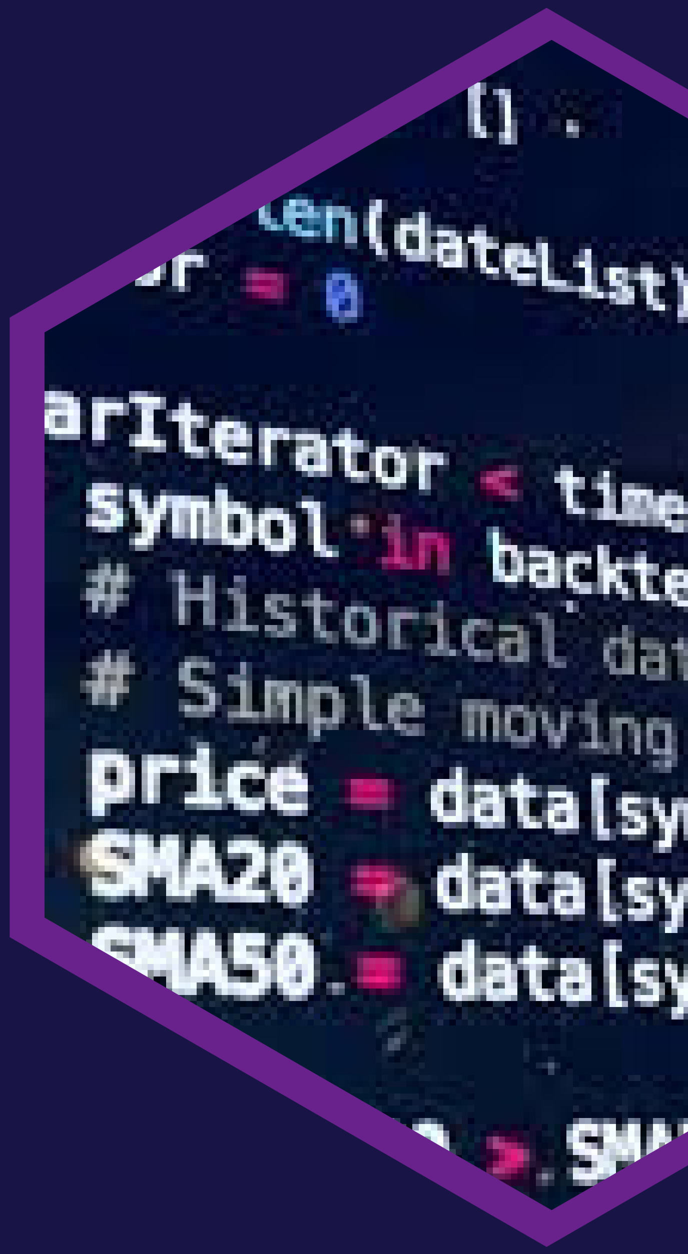
- NumPy is a Python library used for working with arrays.
- It also has functions for working in domain of linear algebra, and matrices.

Pandas:

- Pandas is a Python library used for working with data sets.
- It has functions for analyzing, cleaning, exploring, and manipulating data.

Yfinance:

- yfinance is a popular Python library that provides a simple and convenient way to download historical market data using Yahoo Finance.
- It allows users to retrieve and analyze financial data for stocks, currencies, and more.



1 Returns

Return on investment is basically the gain or loss on an investment during a period of time. For calculating the daily returns we can use the following

```
def Returns(df):

    df["Returns"] = (df["Close"] - df["Open"])/df["Open"]
```

2 Sharpe Ratio

Sharpe Ratio is the measure of investment's risk adjusted performance ,it is calculated by comparing the returns of investment to the returns of a risk free asset The Sharpe ratio divides a portfolio's excess returns by a measure of its volatility to assess risk-adjusted performance

```
SharpeRatio(df):

rfr = 0
return  (252 * (df["Returns"].mean()) - rfr )/(np.sqrt(252) * df["Returns"].std())
```

3 Max drawdown

Max drawdown is the largest loss from a peak to a trough of a portfolio, before a new peak is reached.

```
max_peak_till_now = data['daily_capital'].cummax()
drawdown = (data['daily_capital'] - max_peak_till_now)/max_peak_till_now
max_drawdown=drawdown.min()
```

WEEK - 2

TECHNICAL ANALYSIS AND INDICATORS-

TECHNICAL ANALYSIS :

Technical analysis is a tool, or method, used to predict the probable future price movement of a security – such as a stock or currency pair – based on market data, namely price and volume. It runs on the belief that the past trading activities and price changes of a stock are valuable indicators of the stock's future price movements.

TECHNICAL INDICATORS :

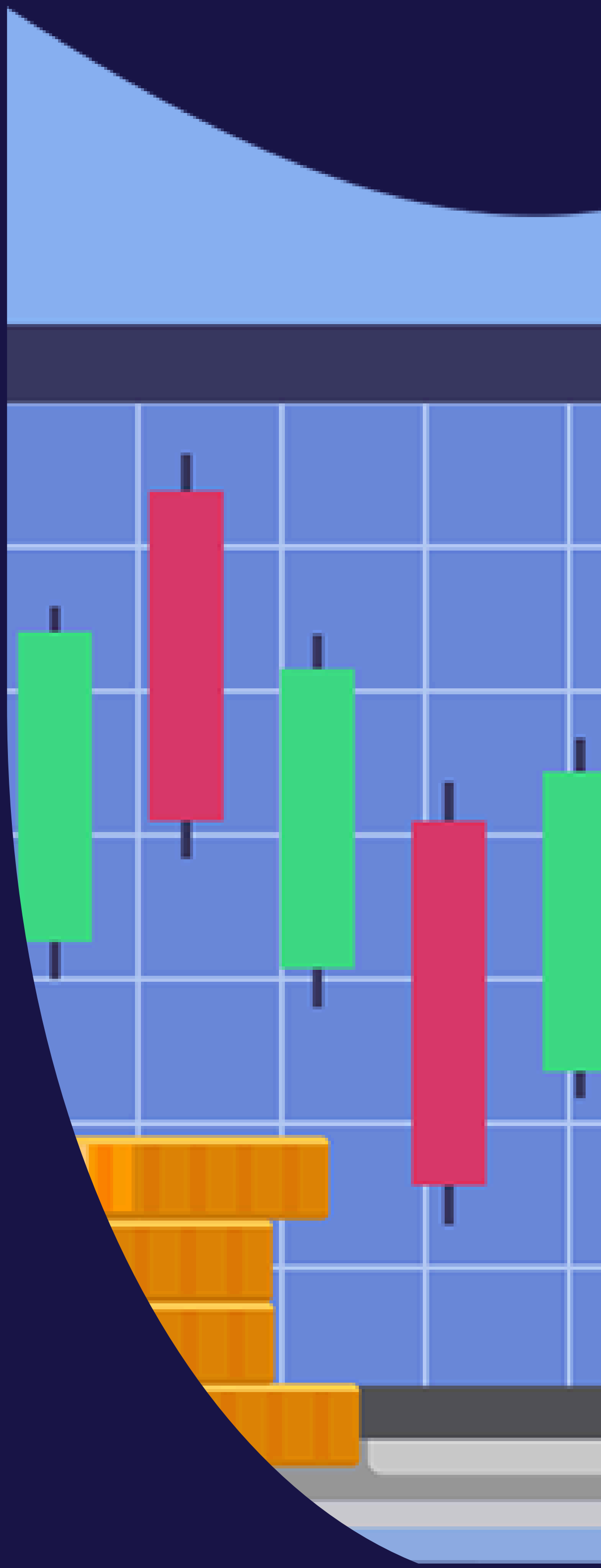
Technical indicators are mathematical pattern based signals based on price and volume of a stock used by a trader to follow technical analysis. These are used to predict future data using the analysis of historical data movements.

MACD:

- Moving average convergence/divergence (MACD) is a technical indicator to help investors identify market entry points for buying or selling.
- The MACD line is calculated by subtracting the 26-period exponential moving average (EMA) from the 12-period EMA.
- The signal line is a nine-period EMA of the MACD line.
- MACD is best used with daily periods, where the traditional settings of 26/12/9 days is the default.

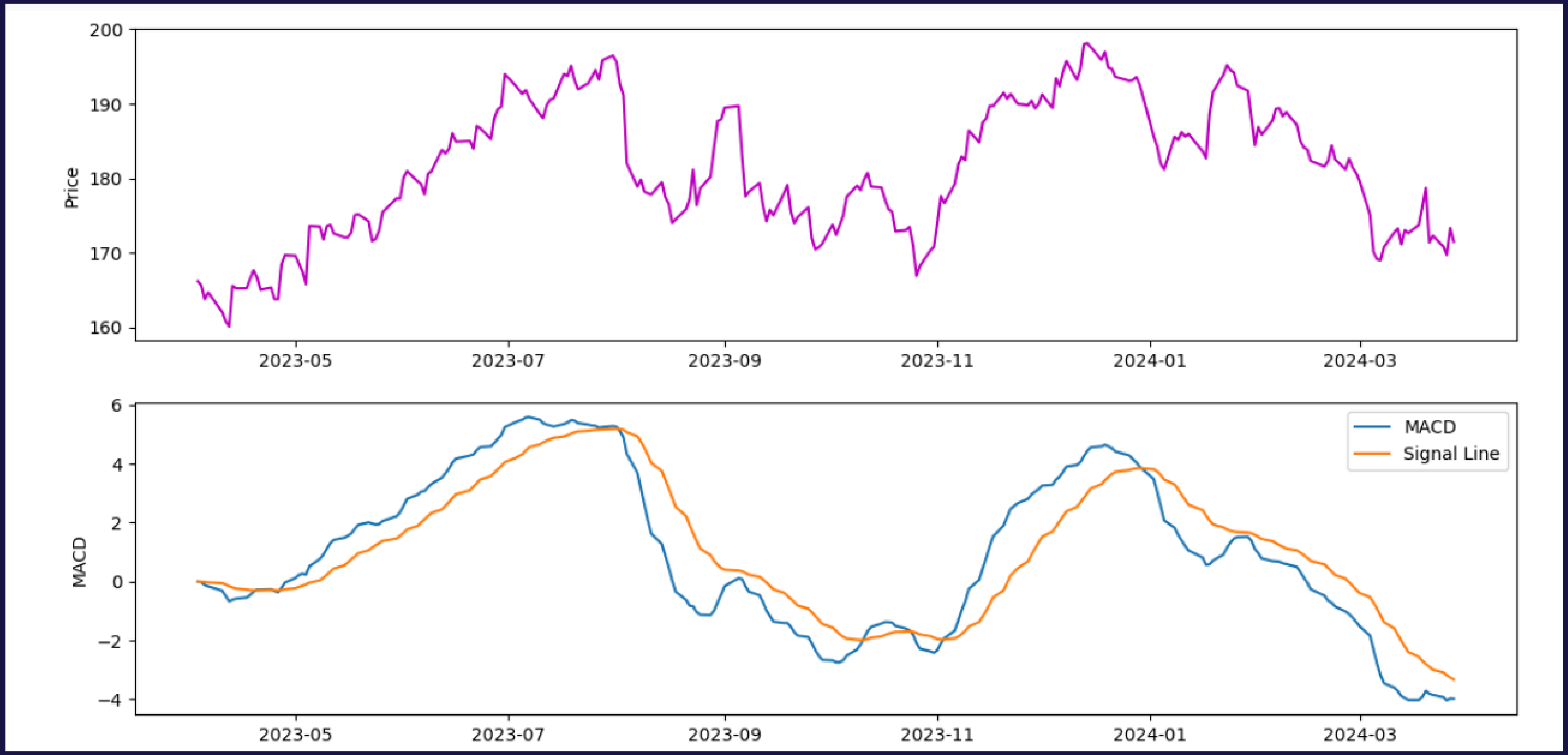
```python
def macd(data):
    ema_12=data['Close'].ewm(12,adjust=False).mean()
    ema_26=data['Close'].ewm(26,adjust=False).mean()
    data['md']=ema_12-ema_26
    data['signal_line']=data['md'].ewm(9,adjust=False).mean()

    return data['md'],data['signal_line']
```



Generating signals from MACD indicator:

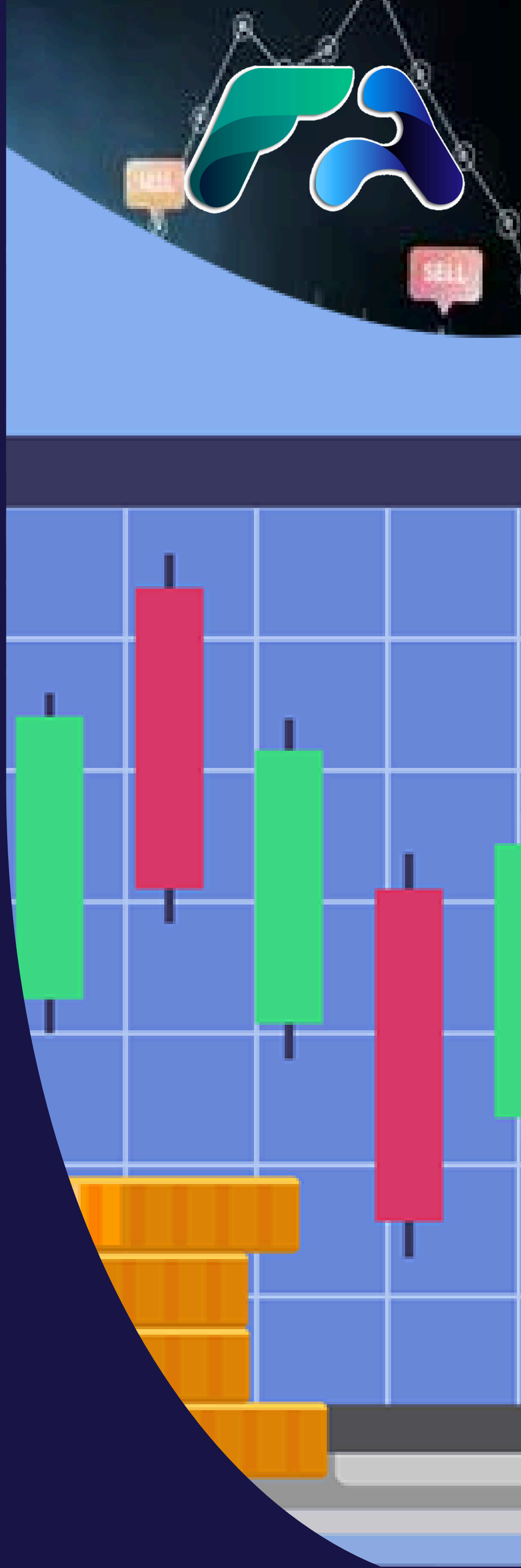
- Short-term buy-and-sell signals are generated by the MACD line and the signal line.
- If the MACD line crosses above the signal line, this may be interpreted as a buy signal.
- If the MACD line crosses below the signal line, this may be interpreted as a sell signal.



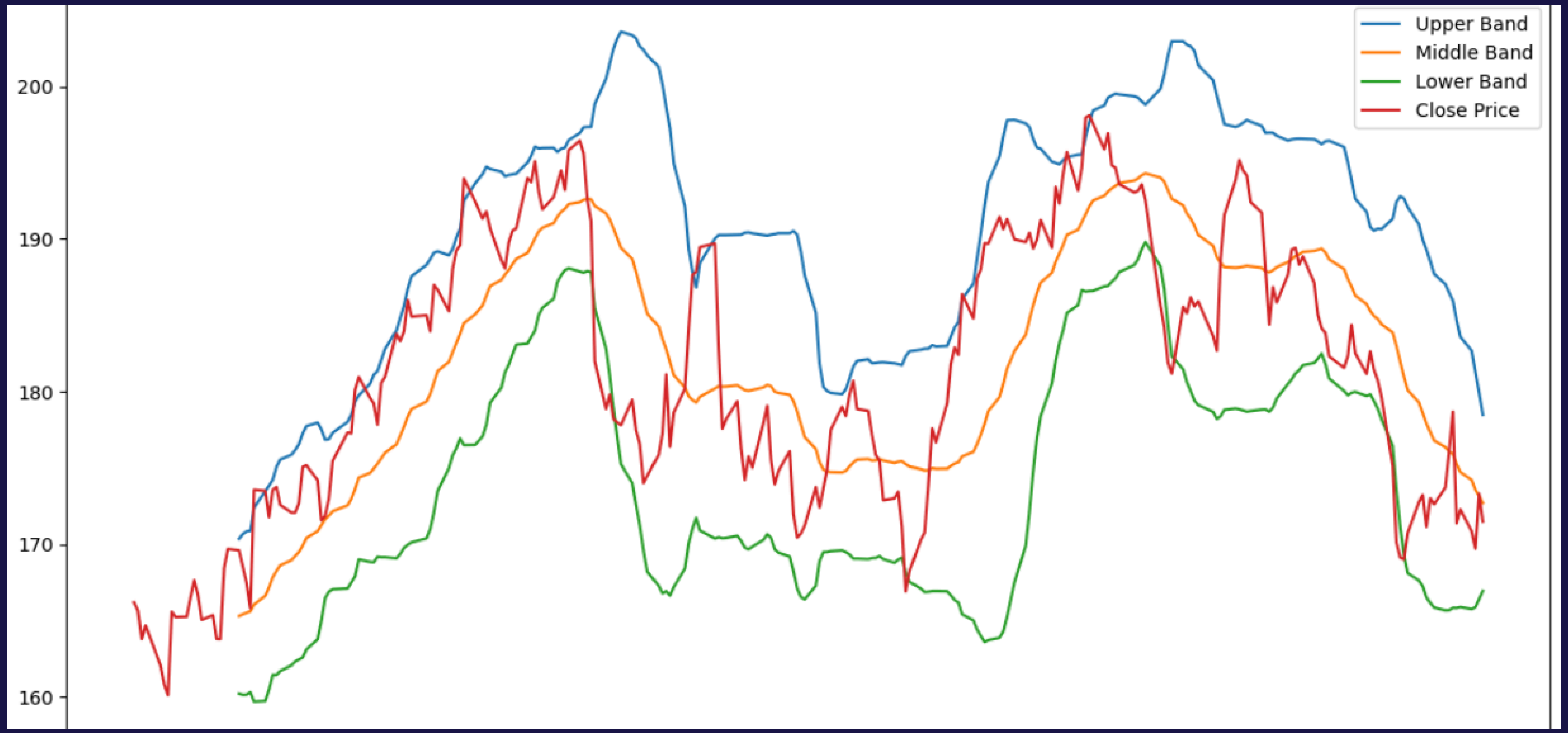
BOLLINGER BANDS:

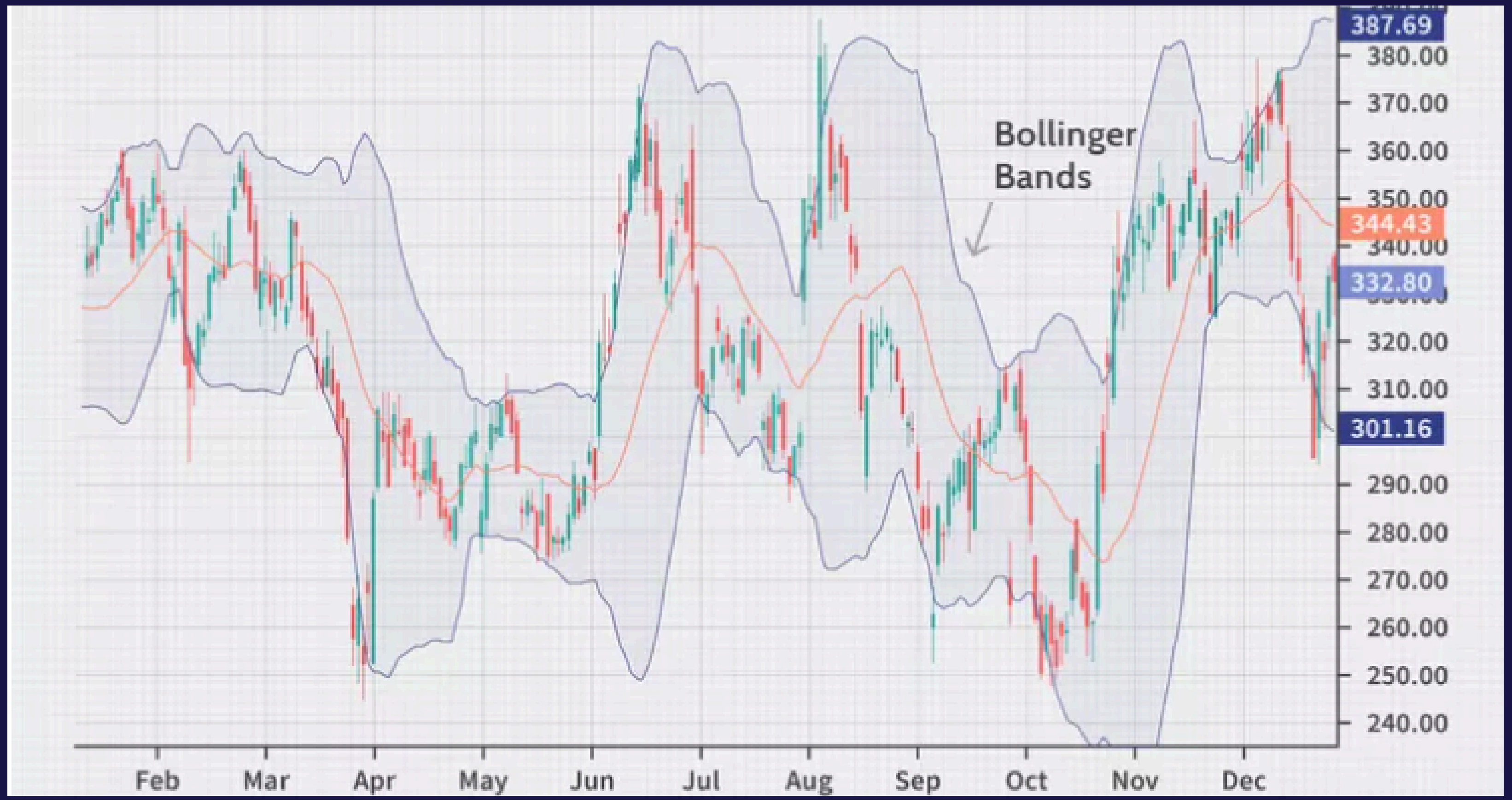
- Bollinger Bands is a technical analysis tool used to determine where prices are high and low relative to each other.
- These bands are composed of three lines: a simple moving average (the middle band) and an upper and lower band.
- The upper and lower bands are typically two standard deviations above or below a 20-period simple moving average (SMA).
- The bands widen and narrow as the volatility of the underlying asset changes.

```python
def bollinger_bands(multiplier):
    SMA=data['Close'].rolling(20).mean()
    STD=data['Close'].rolling(20).std()
    data['upper_band']=SMA+(STD*multiplier)
    data['middle_band']=SMA
    data['lower_band']=SMA-(STD*multiplier)
    return data['upper_band'],data['middle_band'],data['lower_band']
```



Generating signals from Bollinger bands :





When the opening price is greater than the upper band then we sell a stock whereas if a stock's closing price is lower than the lower band then we buy a stock.

BUILD A COHESIVE TEAM

In our team task, we collaborated to develop a trading strategy using technical indicators like Moving Averages and Bollinger Bands. We generated buy/sell signals and implemented risk management techniques, including stop loss and take profit. We also explored denoising methods like Heikin-Ashi to smooth data and reduce noise, enhancing our strategy's effectiveness. Additionally, we learned about Mean Reversion and Pairs Trading to understand advanced trading concepts and their applications. Finally, we applied the ADF Test to ensure our data was stationary and suitable for analysis.

"Talent wins games, but teamwork and intelligence win championships."
– Michael Jordan

WEEK - 3

STOP LOSS:

- A stop-loss order instructs that a stock be bought or sold when it reaches a specified price known as the stop price.
- Once the stop price is met, the stop order becomes a market order and is executed at the next available opportunity.
- Stop-loss orders are used to limit loss or lock in profit on existing positions.
- They can protect investors with either long or short positions.

TAKE PROFIT:

- Take-profit (T/P) orders are limit orders that are closed when a specified profit level is reached.
- Limit prices for T/P orders are placed using either fundamental or technical analysis.
- Take-profit orders are beneficial for short-term traders interested in profiting from a quick bump in the security costs.

IMPLEMENTATION OF STOP LOSS AND TAKE PROFIT:

```
def trading_strategy(data,stop_loss_level,take_profit_level):
    entry_point=[]
    exit_point=[]
    Net_Return=0
    signal=False
    for i in range(len(data)):
        if not signal:
            current=data['Close'].iloc[i]
            entry_point.append(data['Date'].iloc[i])
            Stop_Loss=(1-stop_loss_level)*current
            Take_Profit=(1+take_profit_level)*current
            signal=True
        else:
            if(data['Close'].iloc[i]<=Stop_Loss or data['Close'].iloc[i]>=Take_Profit):
                exit_point.append(data['Date'].iloc[i])
                Net_Return+=(data['Close'].iloc[i]-current)
                signal=False
    if signal:
        exit_point.append(data['Date'].iloc[i])
        Net_Return+=(data['Close'].iloc[i]-current)
    return Net_Return,entry_point,exit_point
```

Mean Reversion and Technical Analysis-

• What is Mean Reversion?

Mean reversion is a theory that says asset prices will tend to revert to their historical mean or average over time. It serves as the backbone for various trading strategies across multiple asset classes, including stocks, forex, and commodities.It helps traders identify overbought or oversold conditions, thereby providing potential entry and exit points.

• What Is Z-Score?

The Z-score is a way to figure out how far away a piece of data is from the average of a group, measured in standard deviations.Z-score is measured in terms of standard deviations from the mean. In investing and trading, Z-scores are measures of an instrument's variability and can be used by traders to help determine volatility.

Use of Mean Reversion in Trading-

Investors employ mean reversion strategies to capitalize on asset prices that have deviated significantly from their historical mean. The underlying assumption is that prices eventually will revert to their long-term average. Investors typically use mean reversion in the following ways:

1. Statistical Analysis: Investors use statistical tools like Z-scores to measure how far an asset price has deviated from its mean. A Z-score above 1.5 or below -1.5 might signal a trading opportunity.

2. Pairs Trading: Here, investors and traders identify two correlated assets. When the price ratio between them deviates from its mean, they go long on the undervalued asset and short the overvalued one.

WEEK - 4

Pairs Trading Strategy and Correlation-

What is Correlation?

The correlation between two stocks is a statistical measure ranging from -1 to +1, representing the degree of co-movement in their prices.

-1 : Perfect Negative correlation.

+1 : Perfect Positive correlation.

Values close to 0: Weak or no correlation.

What is Pairs Trading?

Pairs trading is a trading strategy that involves simultaneously buying one stock and selling another stock in the same sector or with a high historical correlation. The idea is to profit from the relative price movements of the two stocks.

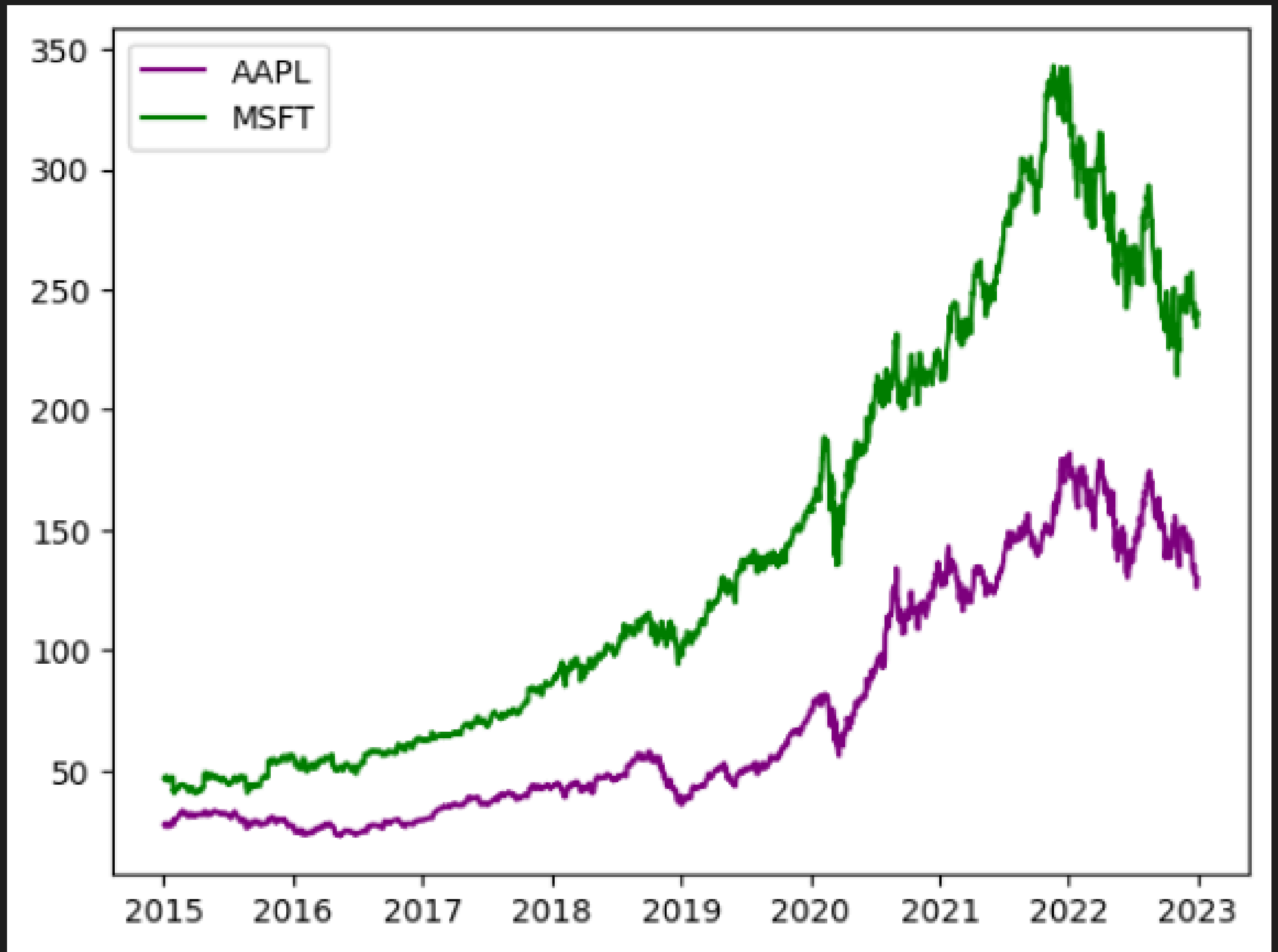
Analyzing Pair Trading-

The image on the right is a code implementation of calculation of correlation value between closing prices of two stocks. Also, plotting closing prices of both the stocks gives an overview about such high correlation. From graph, we can see that they follow similar trend in price movement. The reason for the same can be accounted to the similar nature of Apple and microsoft.

```
correlation=apple['Close'].corr(msft['Close'])
plt.plot(apple ['Close'],color="Purple", label="AAPL")
plt.plot(msft['Close'],color="Green", label="MSFT")
plt.legend()
```

✓ 0.3s

```
<matplotlib.legend.Legend at 0x1ad8377c690>
```




```
correlation
```

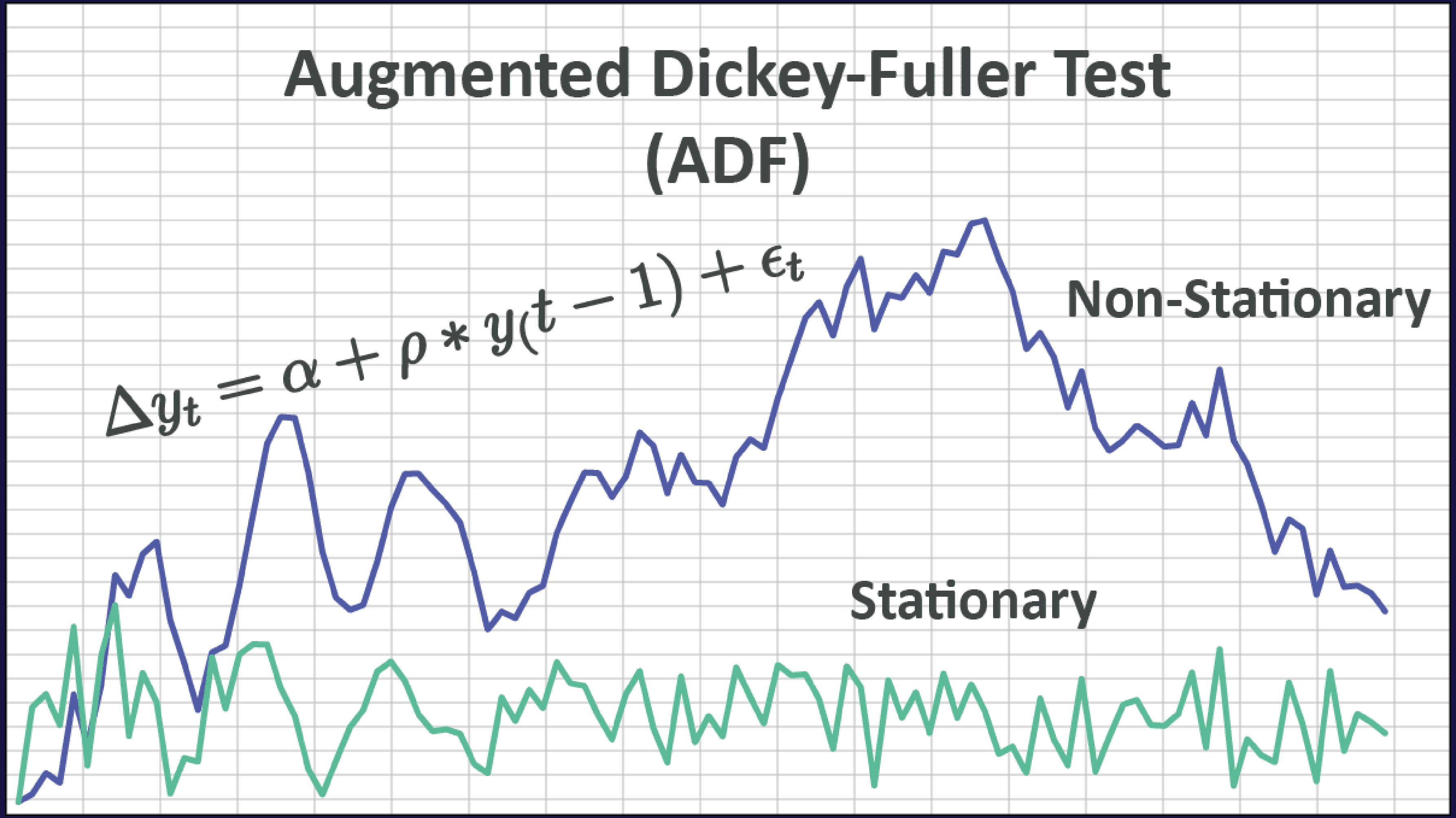
✓ 0.0s

```
0.9780244611941278
```

ADF test-

Augmented Dickey Fuller test (ADF Test) is a common statistical test used to test whether a given Time series is stationary or not. It is one of the most commonly used statistical test when it comes to analyzing the stationary of a series.

This Python code uses statsmodels to perform the ADF test on the price_ratio time series, printing the ADF test statistic which helps to assess whether the price_ratio data is stationary or not.



The Augmented Dickey-Fuller test is based on two hypothesis:

- The null hypothesis states that the time series and is non-stationary.
- The alternative hypothesis states that the time series and is stationary or trend stationary.

Conditions to Reject Null Hypothesis

- p-value < alpha
- ADF Statistic <≈ Critical Value

If these conditions are satisfied then reject Null Hypothesis, i.e. time series does not have a unit root, meaning it is stationary. It does not have a time-dependent structure.

ADF Statistic value is generally a negative number.
More negative is the ADF statistic, more is the reason to reject the null hypothesis, especially when it is less than any of the critical value.

```python
#ADF Test
import statsmodels.api as sm
from statsmodels.tsa.stattools import adfuller
adf_test = adfuller(data)
print('ADF Statistic:', adf_test[0])
print('p-value:', adf_test[1])
print('Critical Values:')
for key, value in adf_test[4].items():
    print(f'   {key}: {value}')


alpha = 0.05
if adf_test[1] < alpha:
    print("Reject the null hypothesis - the time series is stationary.")
else:
    print("Fail to reject the null hypothesis - the time series is non-stationary.")
```

```
ADF Statistic: -1.2034960245313127
p-value: 0.6720900052009388
Critical Values:
   1%: -3.434776133889219
   5%: -2.8634948507368767
   10%: -2.5678107418736302
Fail to reject the null hypothesis - the time series is non-stationary.
```

Data Denoising

Data denoising is a process where we remove the fluctuations called as noise in the chart pattern to make it look more easier to understand and analyse

1. **Heiken Ashi**-Heikin-Ashi, meaning "average bar" in Japanese, is a charting technique used in financial trading to make candlestick charts more readable and to better identify market trends. Unlike traditional candlestick charts, which use open, high, low, and close prices, Heikin-Ashi charts apply a modified formula that smooths price data, reducing market noise and highlighting trends.

Formula:

- HA_CLOSE = (OPEN + HIGH + LOW + CLOSE)/4
- HA_OPEN = (OPEN OF PREV_BAR + CLOSE OF PREV_BAR)/2
- HA_HIGH = MAX(HIGH, HA_CLOSE, HA_OPEN)
- HA_LOW = MIN(LOW, HA_OPEN, HA_CLOSE)

WEEK - 5

Portfolio Optimization:-

Portfolio optimization is the process of choosing the proportions of various assets to include in a portfolio, in such a way that the portfolio has the best possible expected level of return for its level of risk.

Key Concepts:

- Expected Return: The average return that is anticipated on an investment or portfolio.
- Risk(Variance/Standard Deviation): The measure of the dispersion of returns from the expected return.
- Covariance: Measures how two stocks move together.
- Correlation: A standardized measure of covariance.

Steps:

1. **Calculate Expected Returns:** Use historical data to estimate future returns.
2. **Calculate Covariance Matrix:** Measure how each asset moves relative to others.
3. **Optimize Portfolio:** Use mathematical methods to find the best mix of assets.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import yfinance as yf
from scipy.optimize import minimize

# Define stocks and download data
stocks = ["AAPL", "MSFT", "GOOGL", "AMZN"]
data = yf.download(stocks, start="2018-01-01", end="2023-01-01")["Adj Close"]
data.head()
```

[3] ✓ 2.2s

```
[*********************100%%**********************]  4 of 4 completed
```

Ticker	AAPL	AMZN	GOOGL	MSFT
Date				
2018-01-02	40.615883	59.450500	53.598984	79.936737
2018-01-03	40.608807	60.209999	54.513435	80.308746
2018-01-04	40.797447	60.479500	54.725189	81.015572
2018-01-05	41.261944	61.457001	55.450859	82.020012
2018-01-08	41.108669	62.343498	55.646633	82.103722

```python
# Calculate daily returns
returns = data.pct_change().dropna()

# Calculate expected returns and covariance matrix
expected_returns = returns.mean()
cov_matrix = returns.cov()
```

[4] ✓ 0.0s

```python
# Define portfolio performance function
def portfolio_performance(weights, expected_returns, cov_matrix):
    returns = np.dot(weights, expected_returns)
    volatility = np.sqrt(np.dot(weights.T, np.dot(cov_matrix, weights)))
    return returns, volatility

# Define negative Sharpe ratio function
def negative_sharpe_ratio(weights, expected_returns, cov_matrix, risk_free_rate=0.01):
    returns, volatility = portfolio_performance(weights, expected_returns, cov_matrix)
    return -(returns - risk_free_rate) / volatility

# Constraints and bounds
constraints = ({'type': 'eq', 'fun': lambda x: np.sum(x) - 1})
bounds = tuple((0, 1) for _ in range(len(stocks)))

# Initial guess (equal distribution)
initial_guess = len(stocks) * [1. / len(stocks)]

# Optimization
result = minimize(negative_sharpe_ratio, initial_guess, args=(expected_returns, cov_matrix), method='SLSQP', bounds=bounds, constraints=constraints)

# Optimal weights
optimal_weights = result.x
```

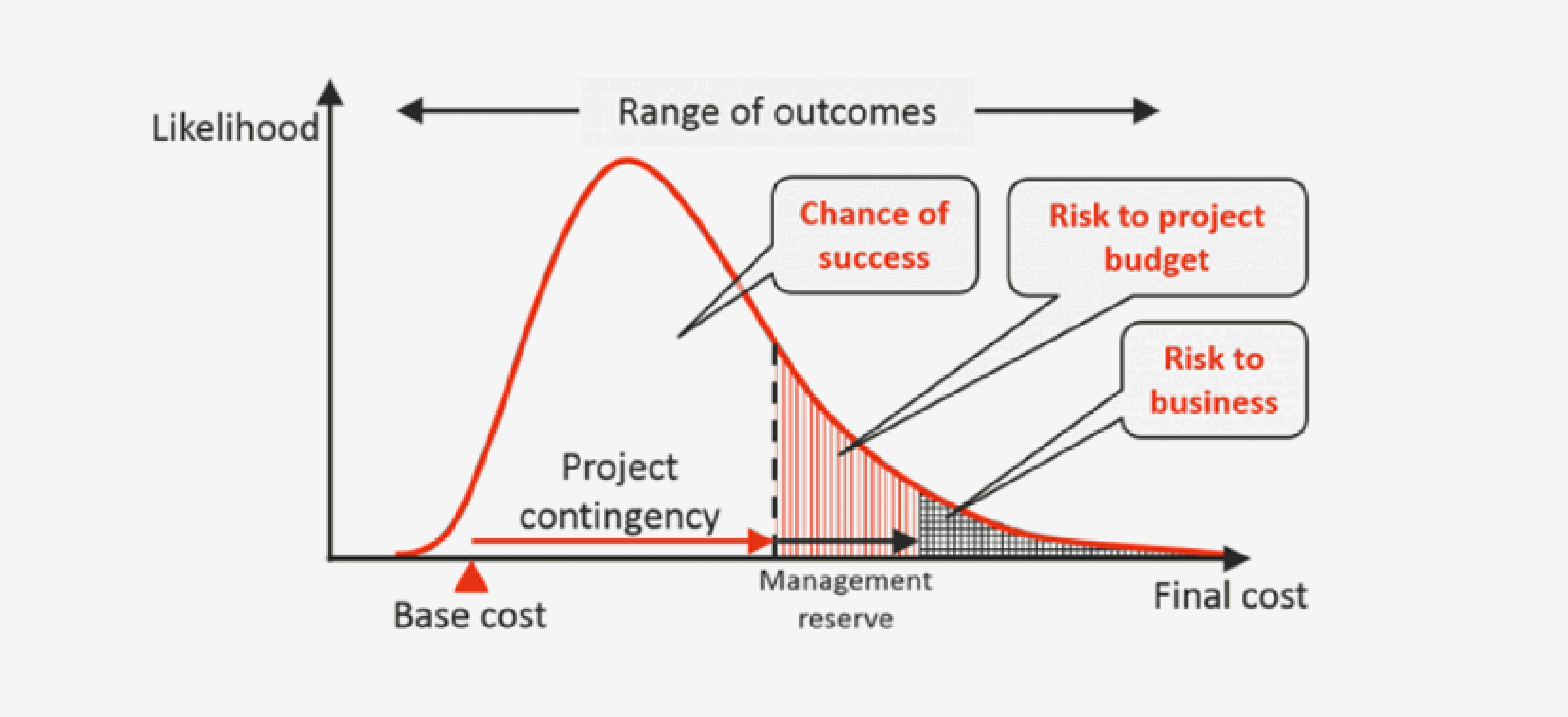
✓ 0.0s

```python
# Display optimal weights
optimal_weights_df = pd.DataFrame(optimal_weights, index=stocks, columns=["Weight"])
optimal_weights_df
```

✓ 0.0s

Monte Carlo Simulation

Monte Carlo simulation is a technique that uses random sampling to predict the possible outcomes of a process. By running many simulations with random inputs, it helps to understand the range of possible results and their probabilities. In finance, it's used to estimate risks, price options, and optimize investment strategies by modeling different future scenarios.



Calculation of π using Monte Carlo Simulation

Monte Carlo simulation can be used to estimate π by randomly generating points within a square and counting how many fall inside a circle inscribed in that square.

Steps:

Randomly generate points (x, y) within a square of side length 2 centered at the origin.

Count the points that fall inside the circle of radius 1.

The ratio of points inside the circle to the total points approximates the area ratio (π/4).

<u>**Formula:**</u>

$\pi \approx 4 \times$Number of points inside the circle/(Total number of points)

```
# import libraries
import numpy as np

# initialize variables
n_simulations = 100000
n_points_circle = 0
n_points_square = 0

# create lists to store x and y values
l_xs = []
l_ys = []

# loop n_simulations times
for _ in range(n_simulations):

    # x is randomly drawn from a continuous uniform distritbuion
    x = np.random.uniform(-1, 1)
    # store x in the list
    l_xs.append(x)

    # y is randomly drawn from a continuous uniform distribution
    y = np.random.uniform(-1, 1)
    # store y in the list
    l_ys.append(y)

# loop n_simulations times
for i in range(n_simulations):

    # calculate the distance between the point and the origin
    dist_from_origin = l_xs[i] ** 2 + l_ys[i] ** 2

    # if the distance is smaller than or equal to 1, the point is in the circle
    if dist_from_origin <= 1:
        n_points_circle += 1

    # by definition of the uniform distribution, the point is in the square
    n_points_square += 1

# estimate the value of pi
pi = 4 * n_points_circle / n_points_square
print(pi)
```

```
3.13836
```

Jarque-Bera test :

The Jarque-Bera test is used to test if a sample of data has the skewness and kurtosis matching a normal distribution. It combines the measures of skewness and kurtosis to quantify the deviation from normality.

Key Concepts:

1)Skewness:

Measures the asymmetry of the data distribution.

A normal distribution has a skewness of 0.

Positive skewness indicates a longer right tail; negative skewness indicates a longer left tail.

2)Kurtosis:

Measures the "tailedness" of the data distribution.

A normal distribution has a kurtosis of 3.

Higher kurtosis indicates more data in the tails and a sharper peak; lower kurtosis indicates less data in the tails and a flatter peak.

- n is the sample size.
- S is the sample skewness.
- K is the sample kurtosis.

Jarque-Bera Statistic (JB):

$$JB = \frac{n}{6}\left(S^2 + \frac{(K-3)^2}{4}\right)$$

Null Hypothesis (H0):

The data follows a normal distribution.

Alternative Hypothesis (H1):

The data does not follow a normal distribution.

P-value:

If the p-value is below a certain threshold (commonly 0.05), we reject the null hypothesis, indicating that the data does not follow a normal distribution.

Code for Jarque-Bera Test:

```
import pandas as pd
import numpy as np
import zipfile
from scipy.stats import jarque_bera

# Step 2: Load the data
data = pd.read_csv('all_stocks_5yr.csv')

# Step 3: Clean and preprocess the data
data['date'] = pd.to_datetime(data['date'])
data = data.sort_values('date')
data['a']= data['close']-data['open']
data['Returns'] = data['a']

# Step 4: Conduct the Jarque-Bera Test
jb_test_stat, jb_p_value = jarque_bera(data['Returns'].dropna())

# Output the results
print(f"Jarque-Bera Test Statistic: {jb_test_stat}")
print(f"p-value: {jb_p_value}")
if jb_p_value <0.05:
 print('We reject null hypothesis and the data is different from normal distribution')
else:
    print('The data follow normal distribution')
```

```
Jarque-Bera Test Statistic: 996762373.2063034
p-value: 0.0
We reject null hypothesis and the data is different from normal distribution
```

WEEK - 6

What is VaR?

"VaR" stands for "Value at Risk." It is a statistical measure used to assess the risk of loss on a specific portfolio of financial assets. VaR estimates the maximum potential loss over a given time period with a certain confidence level, under normal market conditions.

For example, if a portfolio has a one-day VaR of $1 million at a 95% confidence level, it means there is a 95% chance that the portfolio will not lose more than $1 million in a single day.

VaR is widely used by financial institutions for risk management, regulatory compliance, and capital allocation.

Risk Management and Assessment

Risk management is the process of identifying, assessing, and controlling threats to an organization's capital and earnings. These threats could stem from a variety of sources, including financial uncertainties, legal liabilities, and strategic management errors.

Key Steps in Risk Management:

1.Risk Identification:-

Internal and External Risks:-Identify risks originating within the organization (e.g., operational inefficiencies) and external risks (e.g., market volatility).

2.Risk Assessment:-

Qualitative Analysis:-Assess risks based on their likelihood and potential impact. This involves expert judgment and experience.

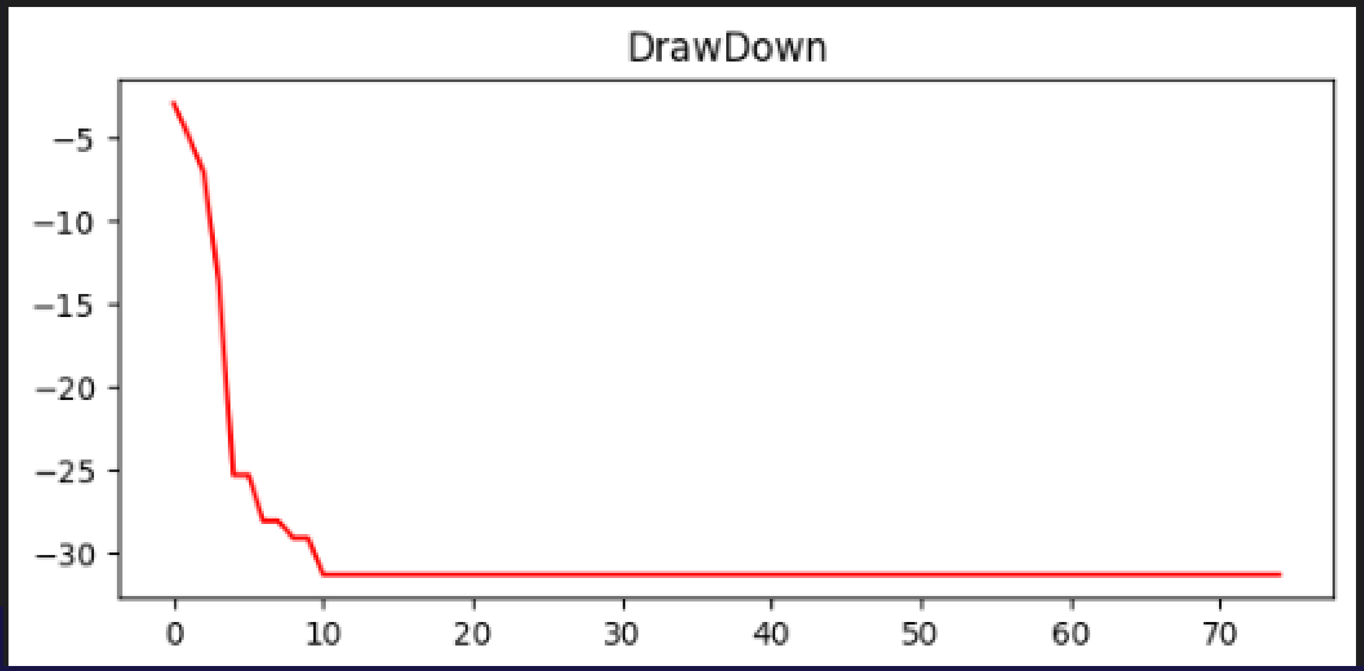
Development of Backtesting Engine

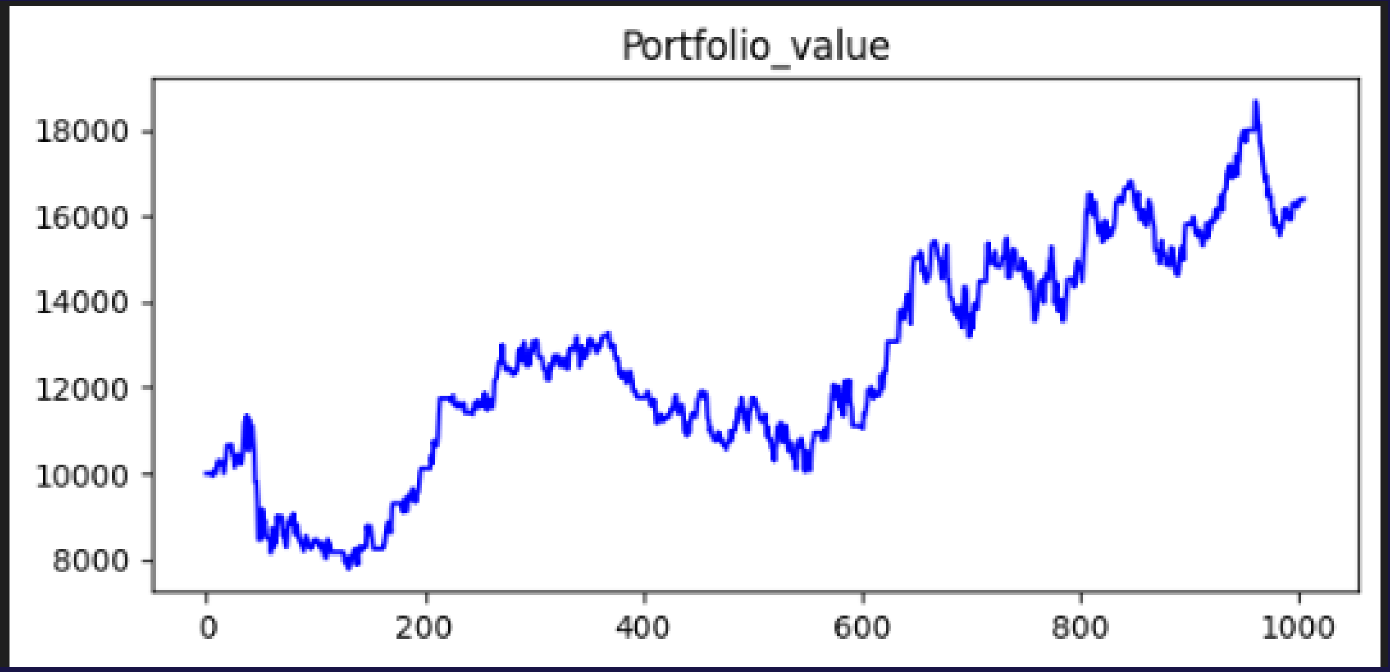
As part of a collaborative effort, our group successfully developed a backtesting code designed to calculate profits, returns, Sharpe ratio and drawdown based on trading signals generated by technical indicators. This tool evaluates the effectiveness of various trading strategies, providing valuable insights into potential market performance. Our framework supports key technical indicators such as the RSI, Bollinger Bands, and MACD, enabling comprehensive analysis and optimization of trading parameters. Our Backtesting code also include risk-optimization using Stop-loss, Take profit and Trailing Stop loss which helps in reducing risk which in turn reduces loss while trading. This effort represents a significant step forward in our ability to systematically and accurately assess trading strategies over historical data

Implementation of Backtesting Engine

These are results obtained by using our backtesting engine of a trading strategy applied to Microsoft's historical data. The strategy utilized the MACD (Moving Average Convergence Divergence) and RSI (Relative Strength Index) indicators to generate buy and sell signals. We also incorporated risk management techniques, including stop loss, take profit, and trailing stop loss.

```
Win:=44
Loss=31
Stop-Loss hit=16
Take-Profit hit=9
Benchmark_Return=13411.780598708465
Gross-Profit=6399.238972969974
Maximum Drawdown=-31.328738599119404
Returns=259.6818102974858
Sharpe_Ratio=0.560099004637375
Sortino_Ratio=0.7573740286241141
```





WEEK - 7

Financial Models

CAPM-

The Capital Asset Pricing Model (CAPM) is a financial model that helps in determining the expected return on an investment based on its risk and the overall market's risk. It provides a framework for evaluating the relationship between expected return and systematic risk, which is represented by beta.

Definitions for a few terms associated with the CAPM:

- Expected Return: The CAPM starts with the concept of expected return, which is the anticipated gain or loss on an investment. It is a measure of the average return an investor expects to receive in exchange for taking on a certain level of risk.
- Risk-Free Rate: The model considers the risk-free rate, which is the return on an investment that is considered to have no risk, such as a government bond. The risk-free rate represents the time value of money and serves as a benchmark for comparing the expected returns of other investments.

- Market Risk Premium: The CAPM takes into account the market risk premium, which is the additional return that investors demand for bearing the risk of investing in the overall market. It is the difference between the expected return on the market as a whole and the risk-free rate.
- Beta: Beta is a measure of an investment's systematic risk or volatility in relation to the overall market. It represents the sensitivity of an investment's returns to changes in the market. A beta of 1 indicates that the investment's returns move in line with the market, while a beta greater than 1 suggests higher volatility, and a beta less than 1 indicates lower volatility

CAPM Formula:

Expected Return = Risk-Free Rate + Beta × (Market Risk Premium)

This formula implies that the expected return of an investment is equal to the risk-free rate plus a risk premium, which is determined by multiplying the investment's beta by the market risk premium.

WEEK - 8

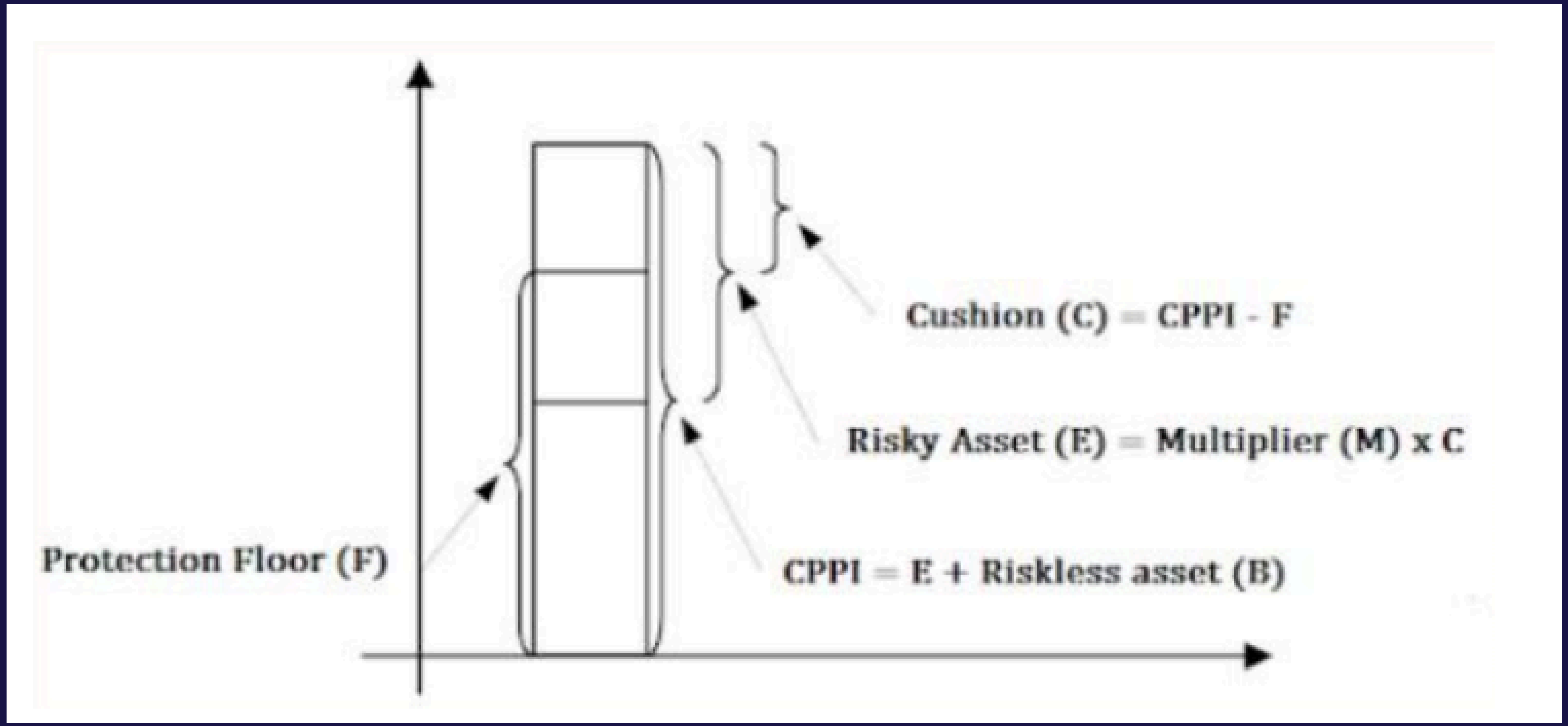
CPPI:

- CPPI (Constant Proportion Portfolio Insurance) is a strategy that allows an investor to keep exposure to a risky asset's upside potential while providing a guarantee against the downside risk by dynamically scaling the exposure.
- Imagine a portfolio consisting of two types of assets, "safe" (with a given yield y) and "risky". Firstly, choose the protection level of the portfolio, i.e. the minimum value you want to protect (for example its starting value, s) and the protection period in years, T. It's then easy to calculate how much may the portfolio value decline at each point in time t, so that the safe asset will be able to recover the loss back to the protection level.

$$F_t = \frac{S}{(1 + y_t)^{(T-t)}}$$

This is the floor (the value, you don't want to go below). The difference between the current portfolio value and the floor is then called cushion; it is the value you can put at risk.

At every point in time, you can allocate multiplier M of the cushion to the risky assets



SCENARIO ANALYSIS:

- Scenario analysis is the process of estimating the expected value of a portfolio after a given period of time, assuming specific changes in the values of the portfolio's securities .These assessments can be used to examine the amount of risk present within a given investment as related to a variety of potential events, ranging from highly probable to highly improbable. Depending on the results of the analysis, an investor can determine if the level of risk present falls within their comfort zone.

STRESS TESTING:

- Stress Testing is a computer simulation technique used to test the resilience of institutions and investment portfolios against possible future financial situations. Such testing is customarily used by the financial industry to help gauge investment risk and the adequacy of assets and help evaluate internal processes and controls. In recent years, regulators have also required financial institutions to carry out stress tests to ensure their capital holdings and other assets are adequate.

Thank you!

-Team AlgoRisk Insights