NIFTY 50 OPTION TRADING USING PYTHON.

DONE BY

OMEGA S
SARANYA N
SEVI ANTO JENSIMA I

COIMBATORE INSTITUTE OF TECHNOLOGY
(A GOVERNMENT AIDED AUTONOMOUS INSTITUTION AFFILIATED TO ANNA UNIVERSITY)
COIMBATORE – 641 014
TAMILNADU, INDIA

MAY 2024

CONTENTS

1. SYNOPSIS :

This project presents a sophisticated exploration of Nifty 50 option trading, employing advanced methodologies to optimize investor returns amidst market complexities. By meticulously analyzing historical data, growth trajectories, and macroeconomic indicators, it calculates expected returns and evaluates risk using sophisticated models like the Treynor and Sharpe ratios. Integration of technical tools such as MACD enhances trend identification and facilitates precise entry/exit point determination. Future enhancements include leveraging advanced machine learning algorithms for predictive modeling, real-time data analysis for agile decision-making, and sentiment analysis for nuanced market insights. A robust risk management framework incorporating scenario analysis and stress testing ensures prudent capital allocation and resilience to market uncertainties. Enhanced visualization techniques and user interfaces provide investors with intuitive tools for better decision-making and engagement. Ultimately, this project offers investors a cutting-edge approach to navigate Nifty 50 option trading, empowering them to achieve their financial objectives with confidence in a dynamic market environment.

2. INTRODUCTION :

In the dynamic world of financial markets, Nifty 50 option trading offers investors a compelling avenue to optimize returns. This project report is dedicated to unraveling the intricacies of this domain, with a primary focus on guiding investors towards informed and profitable investment decisions. At its core lies the meticulous calculation of expected returns for all 50 companies comprising the Nifty 50 index. Through a comprehensive analysis encompassing historical performance, growth trajectories, market dynamics, and macroeconomic indicators, this report endeavors to unveil the companies poised to deliver the most promising returns. Additionally, the determination of the risk-free rate is paramount in evaluating the attractiveness of investment opportunities within the Nifty 50 universe. By examining government bond yields and other low-risk instruments, we establish a baseline against which the excess returns generated by Nifty 50 options can be assessed. Furthermore, the integration of technical analysis tools, notably the Moving Average Convergence Divergence (MACD), enhances our ability to discern market trends and identify optimal entry and exit points for option trades. As we delve deeper into this exploration, the utilization of sophisticated models like the Treynor and Sharpe ratios further refines our investment strategies. These metrics provide a quantitative framework for evaluating the risk-adjusted returns of individual companies, empowering investors to make well-informed decisions. This project report serves as a comprehensive guide, offering insights and strategies to maximize returns while effectively managing risks in the dynamic landscape of Nifty 50 option trading. Through a blend of quantitative analysis, technical expertise, and prudent risk management, investors can navigate the complexities of the market with confidence and precision, ultimately achieving their financial objectives.

3. STATEMENT OF THE PROBLEM:

Navigating Nifty 50 option trading's complexities challenges investors aiming to optimize returns. Present strategies lack structured analyses, neglecting historical performance, growth trends, and market dynamics. Challenges extend to determining the risk-free rate and integrating technical tools like MACD. This dearth of comprehensive guidance leaves investors exposed to risk and uncertainty. There's a critical need for a refined framework providing insights and strategies to effectively navigate this landscape. Addressing these challenges empowers investors to make informed decisions, maximizing returns while managing risks. A comprehensive approach incorporating historical data, growth projections, market trends, and technical analysis like MACD can enhance decision-making. By establishing a structured framework, investors can confidently navigate the intricacies of Nifty 50 option trading, ultimately achieving their financial objectives with precision and confidence.

4. SCOPE OF THE STUDY :

This study focuses on providing a comprehensive understanding of Nifty 50 option trading, aiming to equip investors with the knowledge and tools necessary to navigate this dynamic landscape effectively. The scope encompasses:

1. **Calculation of Expected Returns**: The study meticulously calculates expected returns for all 50 companies within the Nifty 50 index, considering factors such as historical performance, growth trajectories, market dynamics, and macroeconomic indicators.

2. **Evaluation of Risk-Free Rate:** It delves into determining the risk-free rate by examining government bond yields and other low-risk instruments, establishing a baseline against which the excess returns generated by Nifty 50 options can be assessed.

3. **Integration of Technical Analysis:** The study explores the integration of technical analysis tools, notably the Moving Average Convergence Divergence (MACD), to enhance the ability to discern market trends and identify optimal entry and exit points for option trades.

4. **Utilization of Sophisticated Models:** It leverages sophisticated models like the Treynor and Sharpe ratios to further refine investment strategies, providing a quantitative framework for evaluating the risk-adjusted returns of individual companies.

5. Comprehensive Guidance: The study serves as a comprehensive guide, offering insights and strategies to maximize returns while effectively managing risks in Nifty 50 option trading. It aims to empower investors with a blend of quantitative analysis, technical expertise, and prudent risk management, enabling them to achieve their financial objectives with confidence and precision.

5. OBJECTIVES OF THE STUDY:

1. **Identify Optimal Investment Opportunities:** Systematically identify and prioritize investment opportunities within the Nifty 50 index, employing rigorous quantitative and qualitative analyses to pinpoint companies poised for superior returns.
2. **Enhance Decision-Making Process:** Provide investors with a robust decision-making framework grounded in comprehensive data analysis, enabling them to make informed choices in Nifty 50 option trading that align with their financial goals and risk preferences.
3. **Mitigate Risks:** Develop risk mitigation strategies tailored to the nuances of Nifty 50 option trading, leveraging advanced risk management techniques to safeguard investment capital and optimize risk-adjusted returns.
4. **Increase Market Understanding:** Foster a deeper comprehension of Nifty 50 option trading dynamics among investors, equipping them with the knowledge and insights needed to navigate the complexities of the market landscape with proficiency and confidence.

6. LIMITATIONS OF THE STUDY:

1. **Reliance on Historical data :** The forecasts made by the study are predicated on past performance information from the Nifty 50 index businesses. Because financial markets are dynamic, previous data, while insightful, may not necessarily be a reliable indicator of future performance. Future changes in the market, improvements in technology, or adjustments in customer behavior may not be accounted for by past success. Consequently, investors should use caution when considering historical data, even though it might be informative, and they should be aware of any potential restrictions.

2. **Market Volatility :** The financial markets exhibit notable volatility, especially in the context of options trading. The performance of the investing techniques suggested in this research may be impacted by the significant degree of uncertainty brought forth by this intrinsic volatility. It might be difficult to produce consistent results when there are

significant variances from predicted returns due to rapid market volatility. When making investing selections, investors need to take this volatility into account and be ready for it.

3. **Inadequate Macroeconomic Reporting :** It is hard to take into account every probable macroeconomic event, even though the study uses a variety of macroeconomic variables to forecast market trends and company performance. Market conditions can be drastically changed by unforeseen governmental changes, natural disasters, abrupt economic shifts, or geopolitical conflicts. The unpredictability of these events can greatly affect the study's accuracy, highlighting the necessity of continuous monitoring and adaptability in investing plans.

4. **Risk-Free Variations in Rates :** The rates on government bonds and other low-risk investments are used to calculate the risk-free rate. The market's demand for government assets, shifts in governmental policies, and changes in the state of the economy can all affect these rates. These discrepancies may have an impact on the reference point used to evaluate the excess returns from Nifty 50 options, which could result in inaccurate assessments of the attractiveness of investments.

5. **Limited Company Purpose :** The top 50 Indian firms by market capitalization make up the Nifty 50 index, which is the sole subject of this study. Although this emphasis offers a thorough understanding of the main players in the market, it leaves out startups or smaller businesses that might present significant development prospects. This restriction limits the analysis's breadth and raises the possibility of missing out on profitable investing opportunities that are not included in the Nifty 50 index.

6. **Modifications to Regulations :** There could be changes to the regulatory framework controlling options trading and financial markets. Trading techniques and their profitability may be impacted by new laws, modifications to current ones, or adjustments to compliance requirements. These regulatory adjustments may result in higher expenses, different market dynamics, or new restrictions on trade. In order to modify their plans appropriately, investors must be up to date on regulatory developments

7. METHODOLOGY :

7.1 DATA SOURCES

1. **Historical Price Data:** Stock price data for the Nifty 50 companies is sourced from reputable financial data providers such as Bloomberg, Yahoo Finance, and NSE (National Stock Exchange of India) historical data.

2. **Financial Statements:** Company financial reports and statements are obtained from official company filings available on the NSE and company websites.
3. **Government Bond Yields:** Information on government bond yields is retrieved from the RBI and financial data services like Investing.com.

7.2 DATA PRE-PROCESSING

Data pre-processing involves cleaning and organizing the raw data to ensure accuracy and reliability in analysis:

1. **Data Cleaning:** This step includes removing duplicates, correcting errors, and handling missing values in the dataset. Techniques such as mean imputation for missing values or using interpolation for time-series data are employed.
2. **Normalization:** Financial data, especially price data, is normalized to ensure consistency and to facilitate comparison across different companies.
3. **Date Alignment:** Ensuring that all datasets are aligned on the same time frame for accurate historical analysis. This involves synchronizing data points to common dates and intervals.

7.3 DESCRIPTION OF THE TOOLS USED :

1. **Python:**
   - **Pandas:** For data manipulation and analysis, providing data structures and operations for manipulating numerical tables and time series.
   - **NumPy:** For numerical computing, offering support for large multi-dimensional arrays and matrices, along with a collection of mathematical functions.
   - **SciPy:** For advanced mathematical, scientific, and technical computing.
   - **Matplotlib:** For data visualization, enabling the creation of static, animated, and interactive plots.
   - **TA-Lib:** For technical analysis, providing tools for MACD and other indicators.
   - **Statsmodels:** For statistical modeling, useful for regression analysis and statistical tests.

QUANTITATIVE ANALYSIS MODELS

1. **Expected Returns Calculation:**
   - **Historical Average Method:** Expected returns are calculated by averaging historical returns of the Nifty 50 companies over a specified period.

2. **Excess Returns Calculation:**
   - **Formula:** Excess returns are calculated as the difference between the actual return of an investment and the risk-free rate.
   - **Risk-Free Rate:** The yield on government bonds is used as the proxy for the risk-free rate.
3. **MACD (Moving Average Convergence Divergence):**
   - **Calculation:** The MACD is calculated by subtracting the 26-period Exponential Moving Average (EMA) from the 12-period EMA. A 9-day EMA of the MACD, called the "signal line," is then plotted on top of the MACD line, which can act as a trigger for buy and sell signals.
   - **Application:** The MACD is used to identify bullish and bearish trends in stock prices, assisting in the determination of optimal entry and exit points for option trades.
4. **Treynor Ratio:**
   - **Formula:** (Return of the Portfolio - Risk-Free Rate) / Beta of the Portfolio.
   - **Purpose:** This ratio measures returns earned in excess of that which could have been earned on a risk-free investment, per unit of market risk (beta). It helps in evaluating the performance of the portfolio by accounting for market risk

5. **Sharpe Ratio:**
   - **Formula:** (Return of the Portfolio - Risk-Free Rate) / Standard Deviation of the Portfolio’s Excess Return.
   - **Purpose:** This ratio is used to understand the return of an investment compared to its risk. It measures the average return earned in excess of the risk-free rate per unit of volatility or total risk. A higher Sharpe ratio indicates better risk-adjusted returns.

8. ANALYSIS

8.1 ALGORITHM USED

The analysis of Nifty 50 option trading in this project utilizes several key algorithms and statistical models to calculate expected returns, assess risk, and identify optimal investment opportunities. Here are the primary algorithms and methods used:

1. **Simple Averaging for Expected Returns:**
   - **Formula:** Expected Return=30 D %CHNG+365 D %CHNG / 2
   - **Purpose:** This method calculates the average return based on the short-term (30-day) and long-term (365-day) percentage changes in stock prices for each company in the Nifty 50 index. This approach provides a straightforward estimate of expected returns by considering both recent and annual performance.

2. **Excess Returns Calculation:**
   - **Formula:** $\text{Excess Return} = \text{Expected Return} - \text{Risk} - \text{Free Rate}$
   - **Purpose:** This calculation determines the returns earned above the risk-free rate, which is typically represented by government bond yields. It helps in assessing the attractiveness of an investment relative to a low-risk benchmark.
3. **MACD (Moving Average Convergence Divergence):**
   - **Formula:**
     - $\text{MACD} = \text{EMA12} - \text{EMA26}$
     - Signal Line=EMA9(MACD)
     - MACD Histogram=Close Price
   - **Purpose:** The MACD is used to identify trends in stock prices by comparing short-term and long-term moving averages. It helps in determining optimal entry and exit points for option trades by signaling bullish and bearish trends.
4. **Treynor Ratio:**
   - **Formula:**
   - Treynor Ratio=(Return of the Portfolio−Risk-Free Rate) / Beta of the Portfolio
   - **Purpose:** This ratio measures the returns earned in excess of the risk-free rate per unit of market risk (beta). It evaluates the performance of a portfolio considering its market risk, providing a way to compare different portfolios or investments.
5. **Sharpe Ratio:**
   - **Formula:**
   - Sharpe Ratio=(Return of the Portfolio−Risk-Free Rate) / (Standard Deviation of the Portfolio’s Excess Return)
   - **Purpose:** The Sharpe ratio assesses the return of an investment compared to its risk. It measures the average return earned in excess of the risk-free rate per unit of volatility, allowing investors to evaluate risk-adjusted returns.

8.2 EXPLORATORY DATA ANALYSIS (EDA)

Exploratory Data Analysis (EDA) involves examining the data through various statistical and graphical techniques to understand its structure, identify patterns, and extract meaningful insights. Below are key steps and visualizations used in the EDA process for this project:

1. Distribution Plots

Histogram: Plotting histograms for 30-day and 365-day percentage changes to visualize the distribution of returns.

Purpose:

- **Distribution Insight:** Histograms help in understanding the distribution of the returns over different periods, showing how often certain return values occur.

- **Outliers Identification:** They allow for the identification of outliers or extreme values that could significantly affect the analysis.

2. Time Series Analysis

Line Chart: Plotting time series data for stock prices of individual companies to identify trends and seasonal patterns.

Purpose:

- **Trend Analysis:** Line charts reveal trends over time, such as upward or downward movements in stock prices.
- **Seasonal Patterns:** They help identify any seasonal effects or recurring patterns in the data.

3. Bar Chart

Bar Chart: Creating a bar chart to display the expected returns for each company in the Nifty 50 index.

Purpose:

- **Comparison:** Bar charts allow for easy comparison of expected returns across different companies, highlighting which companies have the highest or lowest expected returns.
- **Visual Clarity:** They provide a clear visual representation of the data, making it easier to interpret and analyze.

4. Scatter Plot

Scatter Plot: Comparing the market returns and the returns of a specific company (e.g., Infosys) using a scatter plot.

Purpose:

- **Correlation Insight:** Scatter plots show the relationship between two variables. In this case, it helps in understanding how Infosys returns move in relation to market returns.
- **Trend Identification:** They can reveal whether there is a linear relationship, helping in beta calculation and understanding market sensitivity.

9. CODE

Main Frame :

```
import tkinter as tk
from tkinter import Button, Label
from PIL import Image, ImageTk
import cv2
import os

class MainApp:
    def __init__(self, master):
        self.master = master
        master.title("NIFTY50")
        master.attributes('-fullscreen', True)

        self.video_label = Label(master)
        self.video_label.pack(fill=tk.BOTH, expand=True)

        self.video_source = cv2.VideoCapture("C:/Omega/Semester 4/Financial
Engineering/Project/Mframe.mp4")

        self.update_video()

        title_label = tk.Label(master, text="NIFTY50 OPTION TRADING", font=("Algerian", 50))
        title_label.place(relx=0.5, rely=0.1, anchor="center")

        buttons_frame = tk.Frame(master, bg="#1e1e1e")
        buttons_frame.place(relx=0.7, rely=0.5, anchor="center", relwidth=0.4)

        button_texts = ["Returns", "MACD", "Model"]
        button_commands = [self.call_returns_program, self.call_macd_program,
self.call_model_program]
        button_colors = ["#FFD700", "#87CEEB", "#98FB98", "#FFA07A"]

        for text, command, color in zip(button_texts, button_commands, button_colors):
            button = Button(buttons_frame, text=text, bg=color, font=("Arial", 18), padx=20,
pady=10, command=command)
            button.pack(fill="x", padx=10, pady=5)

    def update_video(self):
        ret, frame = self.video_source.read()
        if ret:
            frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            frame_resized = cv2.resize(frame_rgb, (self.master.winfo_screenwidth(),
self.master.winfo_screenheight()))
```

```
            img = Image.fromarray(frame_resized)
            img_tk = ImageTk.PhotoImage(image=img)
            self.video_label.config(image=img_tk)
            self.video_label.image = img_tk
        self.master.after(10, self.update_video)

    def call_returns_program(self):
        file_path = ('"C:/Omega/Semester 4/Financial Engineering/Project/Returns.py"')
        os.system(f"python {file_path}")

    def call_macd_program(self):
        file_path = ('"C:/Omega/Semester 4/Financial Engineering/Project/Macd.py"')
        os.system(f"python {file_path}")

    def call_model_program(self):
        file_path = ('"C:/Omega/Semester 4/Financial Engineering/Project/Sharpes and
Treynor.py"')
        os.system(f"python {file_path}")


if __name__ == "__main__":
    root = tk.Tk()
    app = MainApp(root)
    root.mainloop()
```

Expected Returns :

```
import pandas as pd
import matplotlib.pyplot as plt
from tkinter import Tk, Button, Text, Scrollbar, Frame, Label, RIGHT, Y
from PIL import Image, ImageTk
import os

df = pd.read_csv("C:/Omega/Semester 4/Financial Engineering/Project/Nifty50.csv")
df.columns = df.columns.str.strip().str.replace('\n', '')

df.rename(columns={'30 D   %CHNG': '30 D %CHNG'}, inplace=True)

numeric_cols = ['30 D %CHNG', '365 D % CHNG  12-May-2023']
df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric, errors='coerce')

df["Expected Return"] = (df["30 D %CHNG"] + df["365 D % CHNG  12-May-2023"]) / 2
```

```
root = Tk()
root.title("Expected Returns")
root.attributes('-fullscreen', True)

background_image_path = "C:/Omega/Semester 4/Financial Engineering/Project/Returns.jpg"  #
Change to your background image path
background_image = Image.open(background_image_path)
background_image = background_image.resize((root.winfo_screenwidth(),
root.winfo_screenheight()), Image.LANCZOS)
background_photo = ImageTk.PhotoImage(background_image)

background_label = Label(root, image=background_photo)
background_label.place(relwidth=1, relheight=1)

value_frame = Frame(root, bg='black')
value_frame.place(relx=0.4, rely=0.2, relwidth=0.5, relheight=0.7)

def go_back():
    root.destroy()
    os.system('C:/Omega/Semester 4/Financial Engineering/Project/Frame.py')

def display_all():
    info_text.config(state='normal')
    info_text.delete('1.0', 'end')
    for index, row in df.iterrows():
        info_text.insert('end', f"Company: {row['SYMBOL']} - Expected Return: {row['Expected
Return']:.2f}%\n")
    info_text.config(state='disabled')

def display_best():
    best_company = df.loc[df["Expected Return"].idxmax()]
    info_text.config(state='normal')
    info_text.delete('1.0', 'end')
    info_text.insert('end', f"Best Company to Invest In:\n"
                            f"Company: {best_company['SYMBOL']}\n"
                            f"Expected Return: {best_company['Expected Return']:.2f}%\n"
                            f"Excess Return over Risk-Free Rate: {best_company['Expected Return'] -
5:.2f}%")
    info_text.config(state='disabled')
```

```
def display_excess():
    df["Excess Return"] = df["Expected Return"] - 5
    info_text.config(state='normal')
    info_text.delete('1.0', 'end')
    for index, row in df.iterrows():
        info_text.insert('end', f"Company: {row['SYMBOL']} - Excess Return: {row['Excess
Return']:.2f}%\n")
    info_text.config(state='disabled')

def display_chart():
    plt.figure(figsize=(12, 6))
    plt.bar(df["SYMBOL"], df["Expected Return"], color='skyblue')
    plt.xlabel('Company Symbol')
    plt.ylabel('Expected Return (%)')
    plt.title('Expected Returns for Each Company')
    plt.xticks(rotation=90)
    plt.tight_layout()

    for i, value in enumerate(df["Expected Return"]):
        plt.text(i, value + 0.5, f"{value:.2f}%", ha='center', va='bottom', fontsize=8)

    plt.show()

title_label = Label(root, text="EXPECTED RETURNS", font=("Algerian", 45), bg='white')
title_label.place(relx=0.5, rely=0.1, anchor="center")

button_all = Button(root, text="All Companies and Expected Returns", command=display_all,
bg="#6A5ACD", font=("Arial", 14), width=30, height=2)
button_all.place(relx=0.05, rely=0.3, anchor="w")

button_best = Button(root, text="Best Company to Invest In", command=display_best,
bg="#32CD32", font=("Arial", 14), width=30, height=2)
button_best.place(relx=0.05, rely=0.4, anchor="w")

button_excess = Button(root, text="Excess Return for Each Company",
command=display_excess, bg="#FF6347", font=("Arial", 14), width=30, height=2)
button_excess.place(relx=0.05, rely=0.5, anchor="w")

button_chart = Button(root, text="Bar Chart", command=display_chart, bg="#1E90FF",
font=("Arial", 14), width=30, height=2)
```

```
button_chart.place(relx=0.05, rely=0.6, anchor="w")

info_text = Text(value_frame, height=10, width=0, bg='black', fg='white', font=("Arial", 14),
wrap='word')
info_text.pack(fill="both", expand=True, side="left")

scrollbar = Scrollbar(value_frame, command=info_text.yview, orient='vertical')
scrollbar.pack(side=RIGHT, fill=Y)
info_text.config(yscrollcommand=scrollbar.set, state='disabled')

back_button = Button(root, text="Back", command=go_back, bg="red", fg="white",
font=("Arial", 14))
back_button.place(relx=0.95, rely=0.05, anchor="ne")

root.mainloop()
```

Moving Average Convergence and Divergence :

```
import pandas as pd
import ta
import matplotlib.pyplot as plt
import tkinter as tk
from tkinter import Button, Frame, Label, Text, Scrollbar, RIGHT, Y
from PIL import Image, ImageTk
import os

class MacdApp:
    def __init__(self, master):
        self.master = master
        master.title("MOVING AVERAGE CONVERGENCE AND DIVERGENCE")
        master.attributes('-fullscreen', True)

        background_image_path = "C:/Omega/Semester 4/Financial Engineering/Project/Ratio.jpg"
# Change to your background image path
        background_image = Image.open(background_image_path)
        background_image = background_image.resize((master.winfo_screenwidth(),
master.winfo_screenheight()), Image.LANCZOS)
        self.background_photo = ImageTk.PhotoImage(background_image)

        self.background_label = Label(master, image=self.background_photo)
        self.background_label.place(relwidth=1, relheight=1)
```

```
        title_label = Label(master, text="MOVING AVERAGE CONVERGENCE AND DIVERGENCE", font=("Algerian", 45), bg='white')
        title_label.place(relx=0.5, rely=0.1, anchor="center")

        self.left_frame = Frame(master, bg='black')
        self.left_frame.place(relx=0.1, rely=0.5, anchor='w')

        self.button_values = Button(self.left_frame, text="Values", command=self.display_values, bg="#6A5ACD", fg="white", font=("Arial", 14), width=10, height=2)
        self.button_values.pack(side="top", padx=10, pady=5)

        self.button_graph = Button(self.left_frame, text="Graph", command=self.display_graph, bg="#87CEEB", fg="white", font=("Arial", 14), width=10, height=2)
        self.button_graph.pack(side="top", padx=10, pady=5)

        self.right_frame = Frame(master, bg='black')
        self.right_frame.place(relx=0.5, rely=0.6, anchor='w')

        self.info_text = Text(self.right_frame, height=30, width=50, bg='black', fg='white', font=("Arial", 12), wrap='word')
        self.info_text.pack(fill="both", expand=True, side="left")

        self.scrollbar = Scrollbar(self.right_frame, command=self.info_text.yview, orient='vertical')
        self.scrollbar.pack(side=RIGHT, fill=Y)
        self.info_text.config(yscrollcommand=self.scrollbar.set)

        self.back_button = Button(master, text="Back", command=self.go_back, bg="red", fg="white", font=("Arial", 14))
        self.back_button.place(relx=0.95, rely=0.2, anchor="ne")

    def go_back(self):
        self.master.destroy()
        os.system('C:/Omega/Semester 4/Financial Engineering/Project/Frame.py')


    def display_values(self):
        df = pd.read_csv("C:/Omega/Semester 4/Financial Engineering/Project/Nifty50.csv")
        df.columns = df.columns.str.strip().str.replace('\n', '')
```

```
        df.rename(columns={'30 D   %CHNG': 'CLOSE'}, inplace=True)

        numeric_cols = ['CLOSE']
        df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric, errors='coerce')

        df['macd'] = ta.trend.macd(df['CLOSE'], window_fast=12, window_slow=26)

        df['macd_signal'] = ta.trend.macd_signal(df['CLOSE'], window_fast=12, window_slow=26,
window_sign=9)

        df['macd_histogram'] = df['macd'] - df['macd_signal']

        self.info_text.delete(1.0, tk.END)  # Clear previous text
        self.info_text.insert(tk.END, df[['SYMBOL', 'macd', 'macd_signal', 'macd_histogram']])

    def display_graph(self):
        df = pd.read_csv("C:/Omega/Semester 4/Financial Engineering/Project/Nifty50.csv")
        df.columns = df.columns.str.strip().str.replace('\n', '')

        df.rename(columns={'30 D   %CHNG': 'CLOSE'}, inplace=True)

        numeric_cols = ['CLOSE']
        df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric, errors='coerce')

        df['macd'] = ta.trend.macd(df['CLOSE'], window_fast=12, window_slow=26)

        df['macd_signal'] = ta.trend.macd_signal(df['CLOSE'], window_fast=12, window_slow=26,
window_sign=9)

        plt.figure(figsize=(12, 6))
        plt.plot(df['macd'], color='green', label='MACD')
        plt.plot(df['macd_signal'], color='red', label='Signal Line')
        plt.bar(df.index, df['macd'] - df['macd_signal'], color='skyblue', label='Histogram')
        plt.xlabel('Index')
        plt.ylabel('Values')
        plt.title('MACD, Signal Line, and Histogram', fontdict={"fontname": "Algerian", "fontsize":
20})
        plt.legend()
        plt.grid(True)
        plt.show(block=False)
```

```
root = tk.Tk()
app = MacdApp(root)
root.mainloop()
```

Sharpe's ratio and Treynor's Model :

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import yfinance as yf
import tkinter as tk
from tkinter import Button, Frame, Label, Text, Scrollbar, RIGHT, Y
from PIL import Image, ImageTk

class StockApp:
    def __init__(self, master):
        self.master = master
        master.title("Stock Analysis")
        master.attributes('-fullscreen', True)

        background_image_path = "C:/Omega/Semester 4/Financial Engineering/Project/Ratio.jpg"
        background_image = Image.open(background_image_path)
        background_image = background_image.resize((master.winfo_screenwidth(),
master.winfo_screenheight()), Image.LANCZOS)
        self.background_photo = ImageTk.PhotoImage(background_image)

        self.background_label = Label(master, image=self.background_photo)
        self.background_label.place(relwidth=1, relheight=1)

        self.left_frame = Frame(master)
        self.left_frame.pack(side="left", padx=10, pady=10)

        self.button_avg_market = Button(master, text="Average and Standard Deviation",
command=self.display_market_metrics, bg="#6A5ACD", font=("Arial", 14), width=30,
height=2)
        self.button_avg_market.place(relx=0.05, rely=0.3, anchor="w")

        self.button_infy_graph = Button(master, text="Infosys", command=self.display_infy_graph,
bg="#87CEEB", font=("Arial", 14), width=30, height=2)
        self.button_infy_graph.place(relx=0.05, rely=0.4, anchor="w")
```

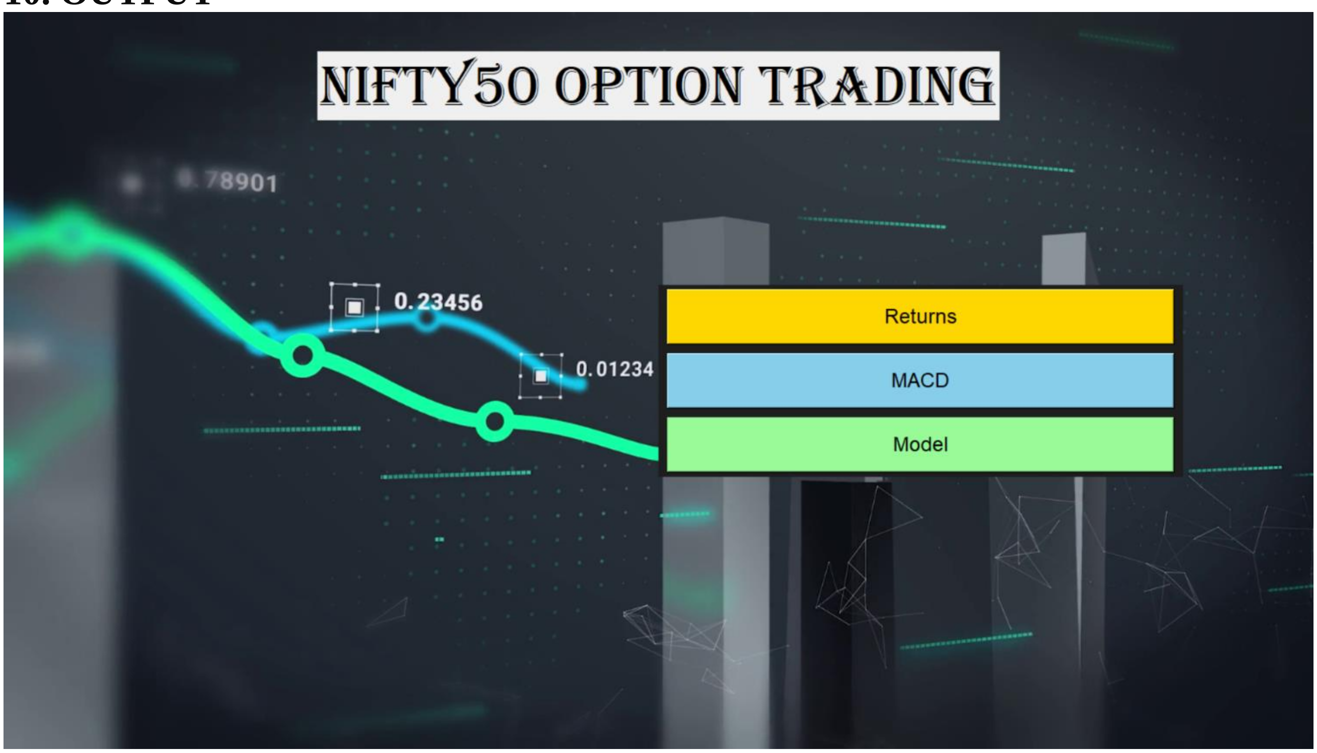
```
        self.button_mm_graph = Button(master, text="Mahindra",
command=self.display_mm_graph, bg="#90EE90", font=("Arial", 14), width=30, height=2)
        self.button_mm_graph.place(relx=0.05, rely=0.5, anchor="w")

        self.button_hindunilvr_graph = Button(master, text="Hindustan",
command=self.display_hindunilvr_graph, bg="#FFA500", font=("Arial", 14), width=30,
height=2)
        self.button_hindunilvr_graph.place(relx=0.05, rely=0.6, anchor="w")

        self.right_frame = Frame(master)
        self.right_frame.pack(side="right", padx=20, pady=20)

        self.metrics_label = Label(self.right_frame, text="", justify="left")
        self.metrics_label.pack()

        self.major_title = Label(master, text="MODELS", font=("Algerian", 45))
        self.major_title.pack(pady=20)


        self.results_frame = Frame(master)
        self.results_frame.pack(side="right", padx=20, pady=20)

        self.info_label = Label(self.results_frame, text="", bg='black', fg='white', font=("Arial", 14),
wraplength=400, justify="right")
        self.info_label.pack(fill="both", expand=True, side="right")

        self.download_data()

    def download_data(self):
        self.nifty_data = yf.download('^NSEI', start='2023-05-15', end='2024-05-15')['Adj Close']

        self.infy_data = yf.download('INFY.NS', start='2023-05-15', end='2024-05-15')['Adj Close']
        self.mm_data = yf.download('M&M.NS', start='2023-05-15', end='2024-05-15')['Adj
Close']
        self.hindunilvr_data = yf.download('HINDUNILVR.NS', start='2023-05-15', end='2024-05-
15')['Adj Close']

    def display_market_metrics(self):
        market_avg_return = self.nifty_data.pct_change().mean()
```

```
        market_std_dev = self.nifty_data.pct_change().std()

        self.info_label.config(text=f"Average Market Return: {market_avg_return}\n"
                                    f"Market Standard Deviation: {market_std_dev}")

    def display_infy_graph(self):
        plt.figure(figsize=(10, 6))
        plt.plot(self.infy_data.index, self.infy_data)
        plt.xlabel('Date')
        plt.ylabel('Infosys Price')
        plt.title('Infosys Stock Prices')
        plt.legend()
        plt.show()

        infy_returns = self.infy_data.pct_change().dropna()
        nifty_returns = self.nifty_data.pct_change().dropna()

        covariance_infy = np.cov(infy_returns, nifty_returns)[0, 1]
        market_var = np.var(nifty_returns)

        beta_infy = covariance_infy / market_var

        risk_free_rate = 0.05
        excess_returns_infy = infy_returns - risk_free_rate

        sharpe_ratio_infy = excess_returns_infy.mean() / excess_returns_infy.std()

        treynor_ratio_infy = excess_returns_infy.mean() / beta_infy

        self.info_label.config(text=f"Infosys Metrics:\n"
                                    f"Beta: {beta_infy}\n"
                                    f"Sharpe's Ratio: {sharpe_ratio_infy}\n"
                                    f"Treynor's Ratio: {treynor_ratio_infy}")

    def display_mm_graph(self):
        plt.figure(figsize=(10, 6))
        plt.plot(self.mm_data.index, self.mm_data, label='M&M', color='green')
        plt.xlabel('Date')
        plt.ylabel('M&M Price')
```

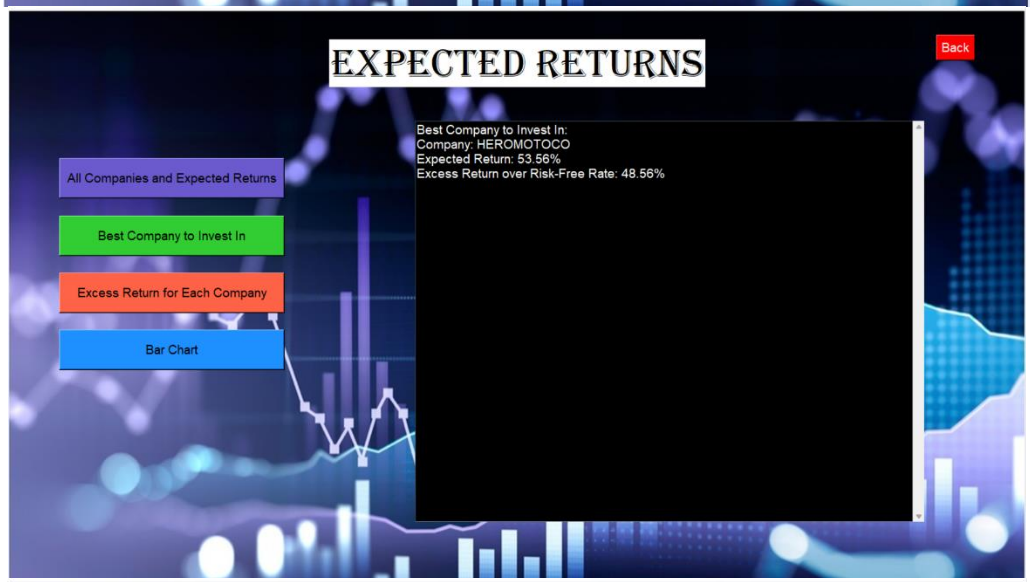
```
        plt.title('M&M Stock Prices')
        plt.legend()
        plt.show()

        mm_returns = self.mm_data.pct_change().dropna()
        nifty_returns = self.nifty_data.pct_change().dropna()

        covariance_mm = np.cov(mm_returns, nifty_returns)[0, 1]
        market_var = np.var(nifty_returns)

        beta_mm = covariance_mm / market_var

        risk_free_rate = 0.05
        excess_returns_mm = mm_returns - risk_free_rate

        sharpe_ratio_mm = excess_returns_mm.mean() / excess_returns_mm.std()

        treynor_ratio_mm = excess_returns_mm.mean() / beta_mm

        self.info_label.config(text=f"M&M Metrics:\n"
                                f"Beta: {beta_mm}\n"
                                f"Sharpe's Ratio: {sharpe_ratio_mm}\n"
                                f"Treynor's Ratio: {treynor_ratio_mm}")

    def display_hindunilvr_graph(self):
        plt.figure(figsize=(10, 6))
        plt.plot(self.hindunilvr_data.index, self.hindunilvr_data, label='HINDUNILVR',
color='orange')
        plt.xlabel('Date')
        plt.ylabel('HINDUNILVR Price')
        plt.title('HINDUNILVR Stock Prices')
        plt.legend()
        plt.show()

        hindunilvr_returns = self.hindunilvr_data.pct_change().dropna()
        nifty_returns = self.nifty_data.pct_change().dropna()

        covariance_hindunilvr = np.cov(hindunilvr_returns, nifty_returns)[0, 1]
        market_var = np.var(nifty_returns)
```

```
        beta_hindunilvr = covariance_hindunilvr / market_var

        risk_free_rate = 0.05
        excess_returns_hindunilvr = hindunilvr_returns - risk_free_rate

        sharpe_ratio_hindunilvr = excess_returns_hindunilvr.mean() /
excess_returns_hindunilvr.std()

        treynor_ratio_hindunilvr = excess_returns_hindunilvr.mean() / beta_hindunilvr

        self.info_label.config(text=f"HINDUNILVR Metrics:\n"
                                    f"Beta: {beta_hindunilvr}\n"
                                    f"Sharpe's Ratio: {sharpe_ratio_hindunilvr}\n"
                                    f"Treynor's Ratio: {treynor_ratio_hindunilvr}")


root = tk.Tk()
app = StockApp(root)
root.mainloop()
```

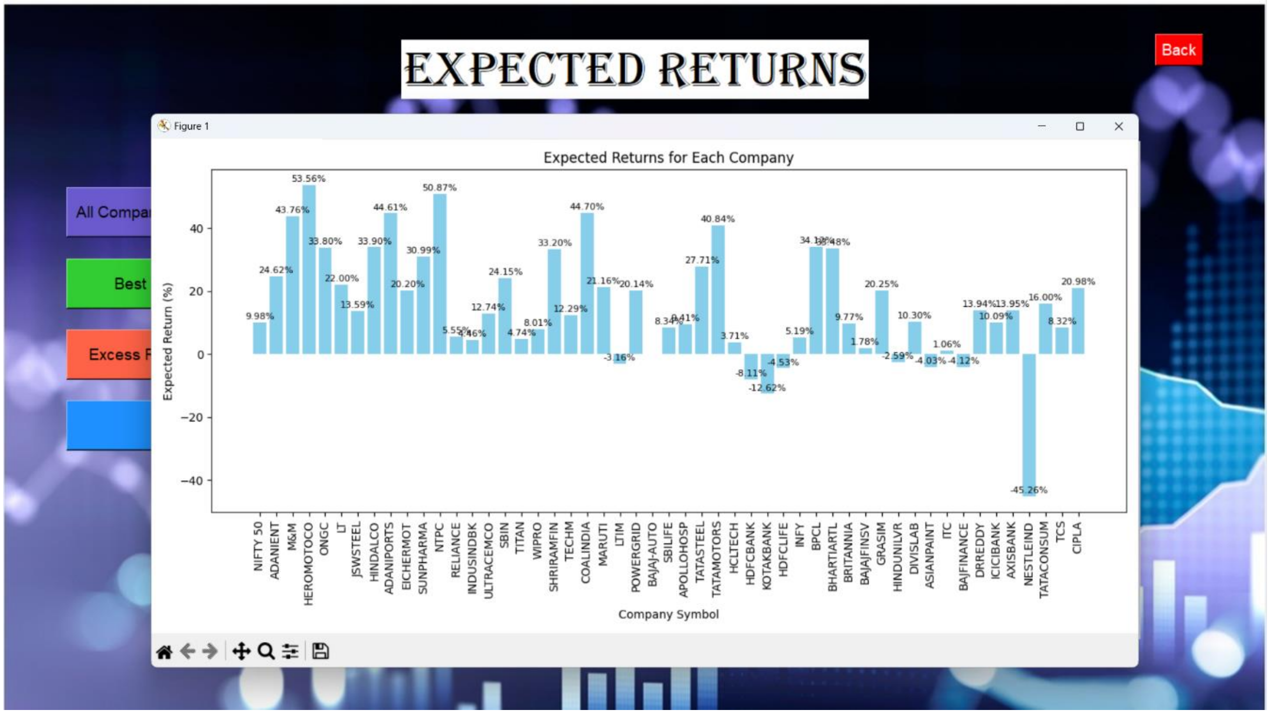
10. OUTPUT

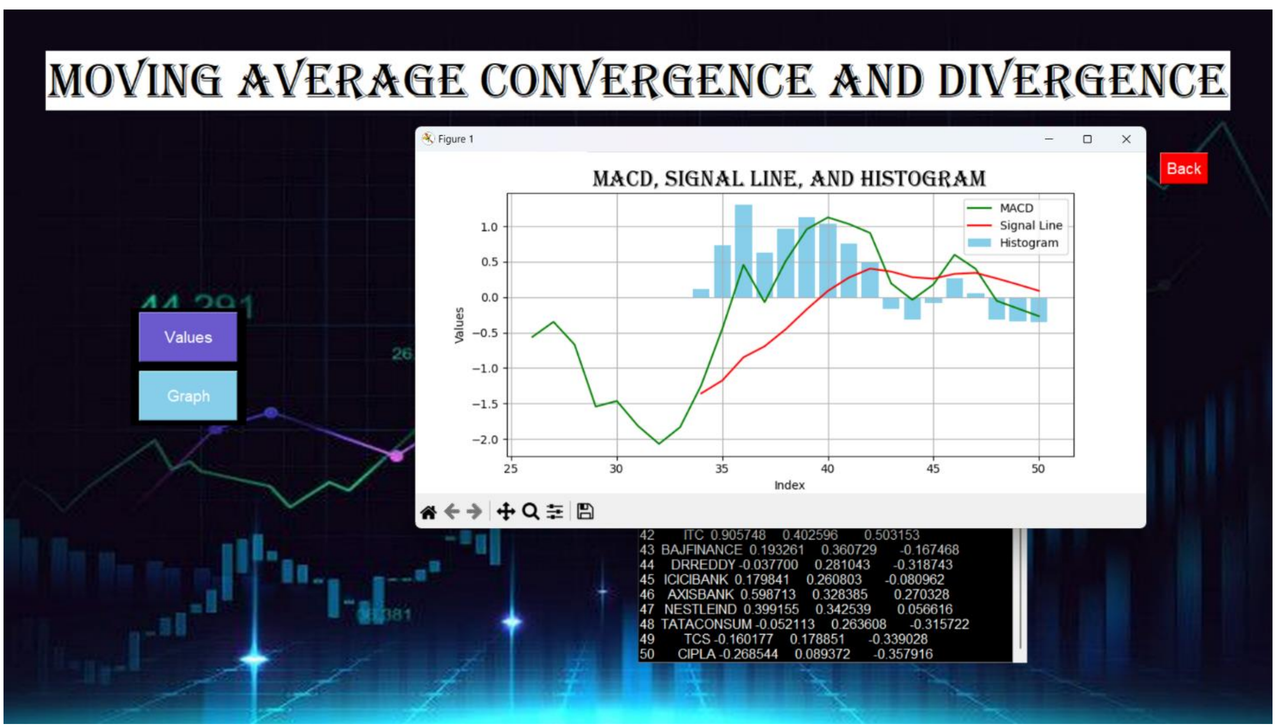



EXPECTED RETURNS
Back
All Companies and Expected Returns
Best Company to Invest In
Excess Return for Each Company
Bar Chart
Company: NIFTY 50 - Expected Return: 9.98%
Company: ADANIENT - Expected Return: 24.62%
Company: M&M - Expected Return: 43.76%
Company: HEROMOTOCO - Expected Return: 53.56%
Company: ONGC - Expected Return: 33.80%
Company: LT - Expected Return: 22.00%
Company: JSWSTEEL - Expected Return: 13.59%
Company: HINDALCO - Expected Return: 33.90%
Company: ADANIPORTS - Expected Return: 44.61%
Company: EICHERMOT - Expected Return: 20.20%
Company: SUNPHARMA - Expected Return: 30.99%
Company: NTPC - Expected Return: 50.87%
Company: RELIANCE - Expected Return: 5.55%
Company: INDUSINDBK - Expected Return: 4.46%
Company: ULTRACEMCO - Expected Return: 12.74%
Company: SBIN - Expected Return: 24.15%
Company: TITAN - Expected Return: 4.74%
Company: WIPRO - Expected Return: 8.01%
Company: SHRIRAMFIN - Expected Return: 33.20%
Company: TECHM - Expected Return: 12.29%
Company: COALINDIA - Expected Return: 44.70%
Company: MARUTI - Expected Return: 21.16%
Company: LTIM - Expected Return: -3.16%
Company: POWERGRID - Expected Return: 20.14%
Company: BAJAJ-AUTO - Expected Return: nan%
Company: SBILIFE - Expected Return: 8.34%
Company: APOLLOHOSP - Expected Return: 9.41%

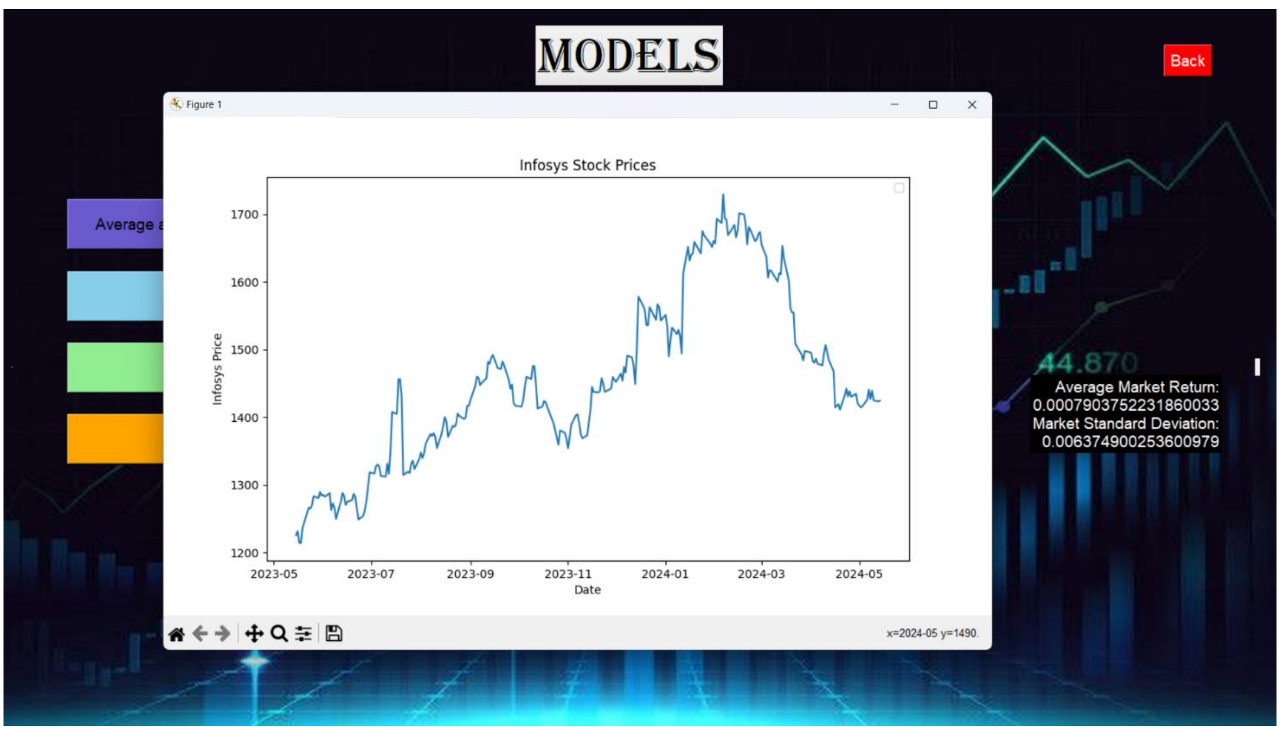

EXPECTED RETURNS
Back
All Companies and Expected Returns
Best Company to Invest In
Excess Return for Each Company
Bar Chart
Best Company to Invest In:
Company: HEROMOTOCO
Expected Return: 53.56%
Excess Return over Risk-Free Rate: 48.56%

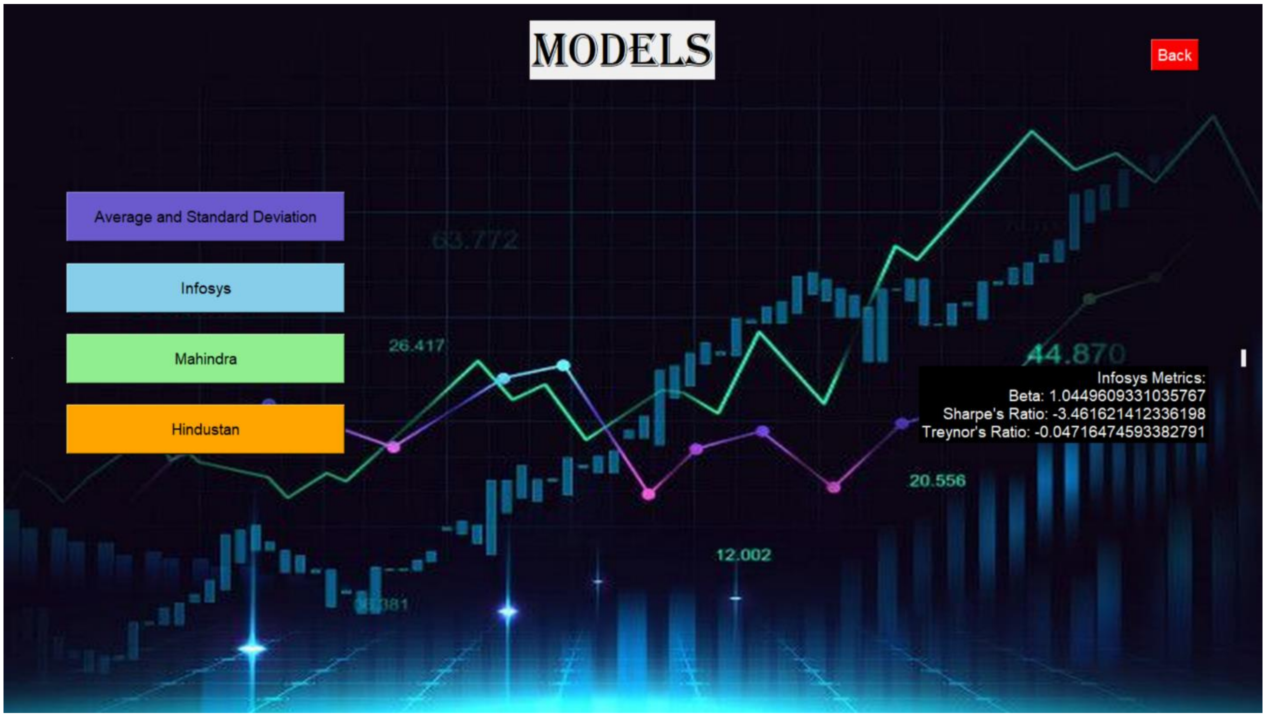

EXPECTED RETURNS
Back
All Companies and Expected Returns
Best Company to Invest In
Excess Return for Each Company
Bar Chart
Company: NIFTY 50 - Excess Return: 4.98%
Company: ADANIENT - Excess Return: 19.62%
Company: M&M - Excess Return: 38.76%
Company: HEROMOTOCO - Excess Return: 48.56%
Company: ONGC - Excess Return: 28.80%
Company: LT - Excess Return: 17.00%
Company: JSWSTEEL - Excess Return: 8.59%
Company: HINDALCO - Excess Return: 28.90%
Company: ADANIPORTS - Excess Return: 39.61%
Company: EICHERMOT - Excess Return: 15.20%
Company: SUNPHARMA - Excess Return: 25.99%
Company: NTPC - Excess Return: 45.87%
Company: RELIANCE - Excess Return: 0.55%
Company: INDUSINDBK - Excess Return: -0.54%
Company: ULTRACEMCO - Excess Return: 7.74%
Company: SBIN - Excess Return: 19.15%
Company: TITAN - Excess Return: -0.26%
Company: WIPRO - Excess Return: 3.01%
Company: SHRIRAMFIN - Excess Return: 28.20%
Company: TECHM - Excess Return: 7.29%
Company: COALINDIA - Excess Return: 39.70%
Company: MARUTI - Excess Return: 16.16%
Company: LTIM - Excess Return: -8.16%
Company: POWERGRID - Excess Return: 15.14%
Company: BAJAJ-AUTO - Excess Return: nan%
Company: SBILIFE - Excess Return: 3.34%
Company: APOLLOHOSP - Excess Return: 4.41%

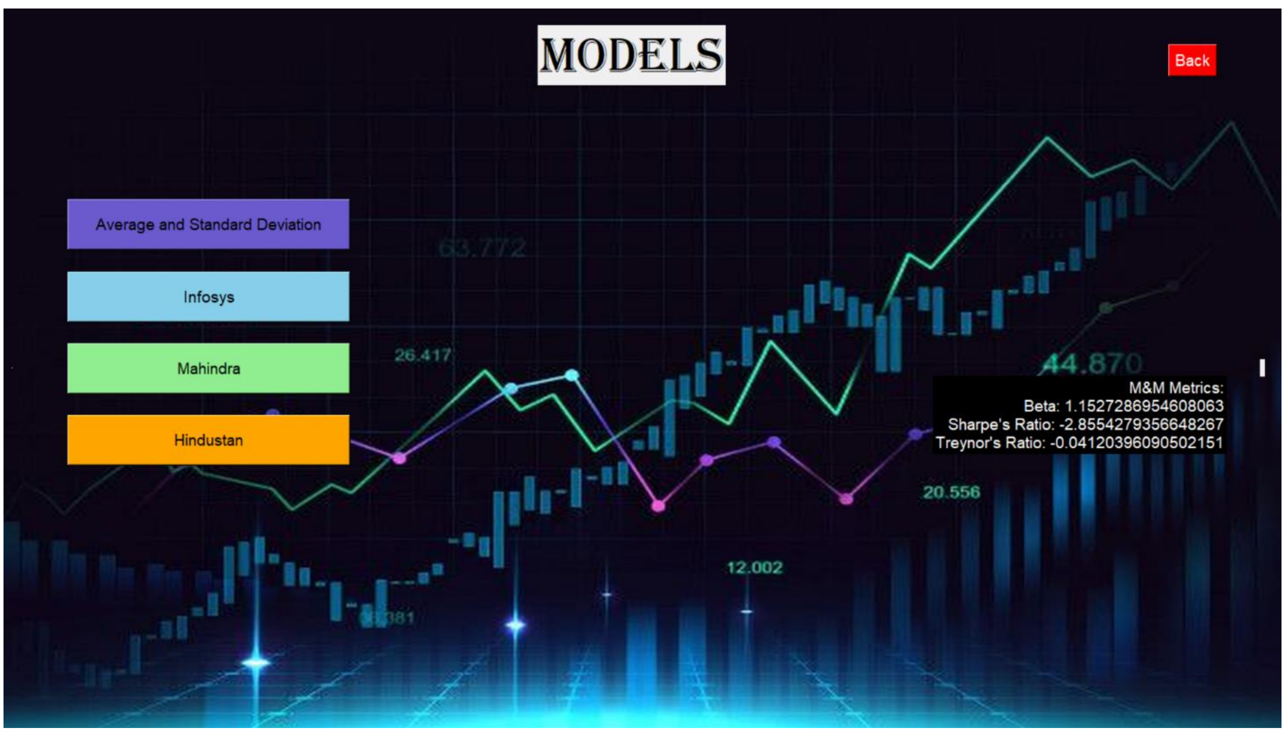

EXPECTED RETURNS
Back
Figure 1
Expected Returns for Each Company
Expected Return (%)
Company Symbol
All Compa
Best
Excess R
NIFTY 50
ADANIENT
M&M
HEROMOTOCO
ONGC
LT
JSWSTEEL
HINDALCO
ADANIPORTS
EICHERMOT
SUNPHARMA
NTPC
RELIANCE
INDUSINDBK
ULTRACEMCO
SBIN
TITAN
WIPRO
SHRIRAMFIN
TECHM
COALINDIA
MARUTI
LTIM
POWERGRID
BAJAJ-AUTO
SBILIFE
APOLLOHOSP
TATASTEEL
TATAMOTORS
HCLTECH
HDFCBANK
KOTAKBANK
HDFCLIFE
INFY
BPCL
BHARTIARTL
BRITANNIA
BAJAJFINSV
GRASIM
HINDUNILVR
DIVISLAB
ASIANPAINT
ITC
BAJFINANCE
DRREDDY
ICICIBANK
AXISBANK
NESTLEIND
TATACONSUM
TCS
CIPLA

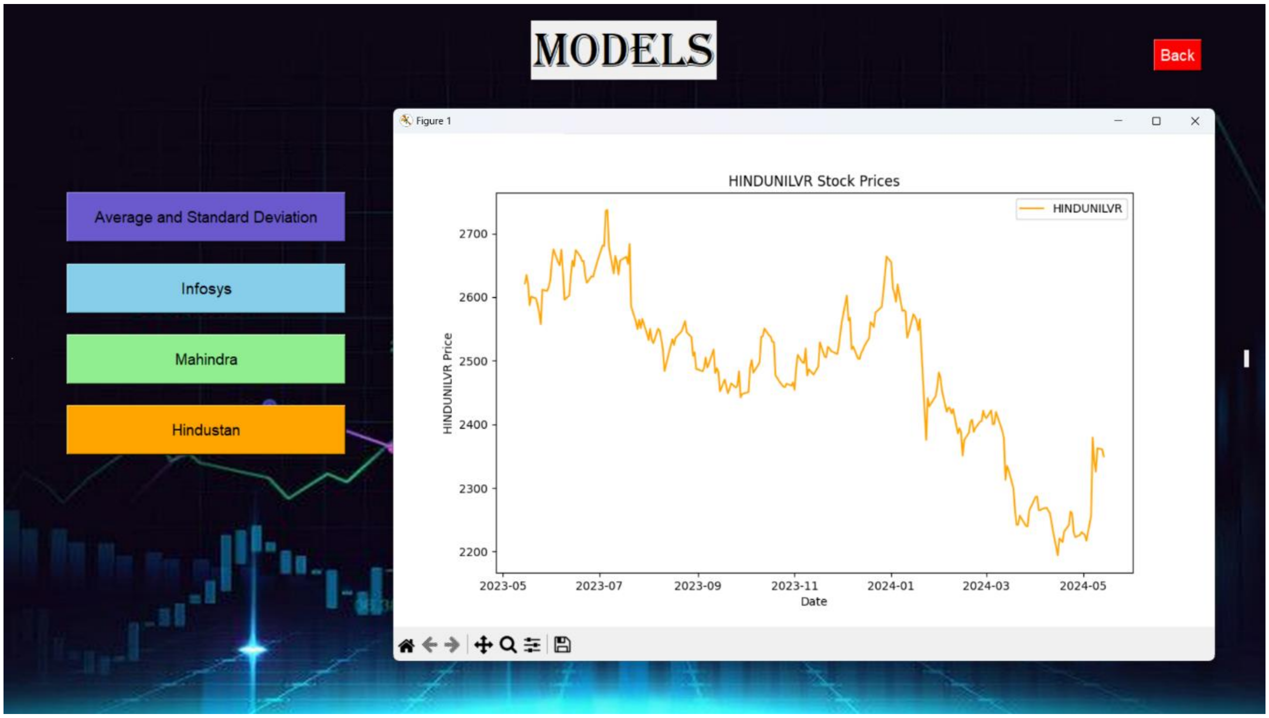

MOVING AVERAGE CONVERGENCE AND DIVERGENCE
Back
Values
Graph
Figure 1
MACD, SIGNAL LINE, AND HISTOGRAM
MACD
Signal Line
Histogram
Values
Index
42 ITC 0.905748 0.402596 0.503153
43 BAJFINANCE 0.193261 0.360729 -0.167468
44 DRREDDY -0.037700 0.281043 -0.318743
45 ICICIBANK 0.179841 0.260803 -0.080962
46 AXISBANK 0.598713 0.328385 0.270328
47 NESTLEIND 0.399155 0.342539 0.056616
48 TATACONSUM -0.052113 0.263608 -0.315722
49 TCS -0.160177 0.178851 -0.339028
50 CIPLA -0.268544 0.089372 -0.357916

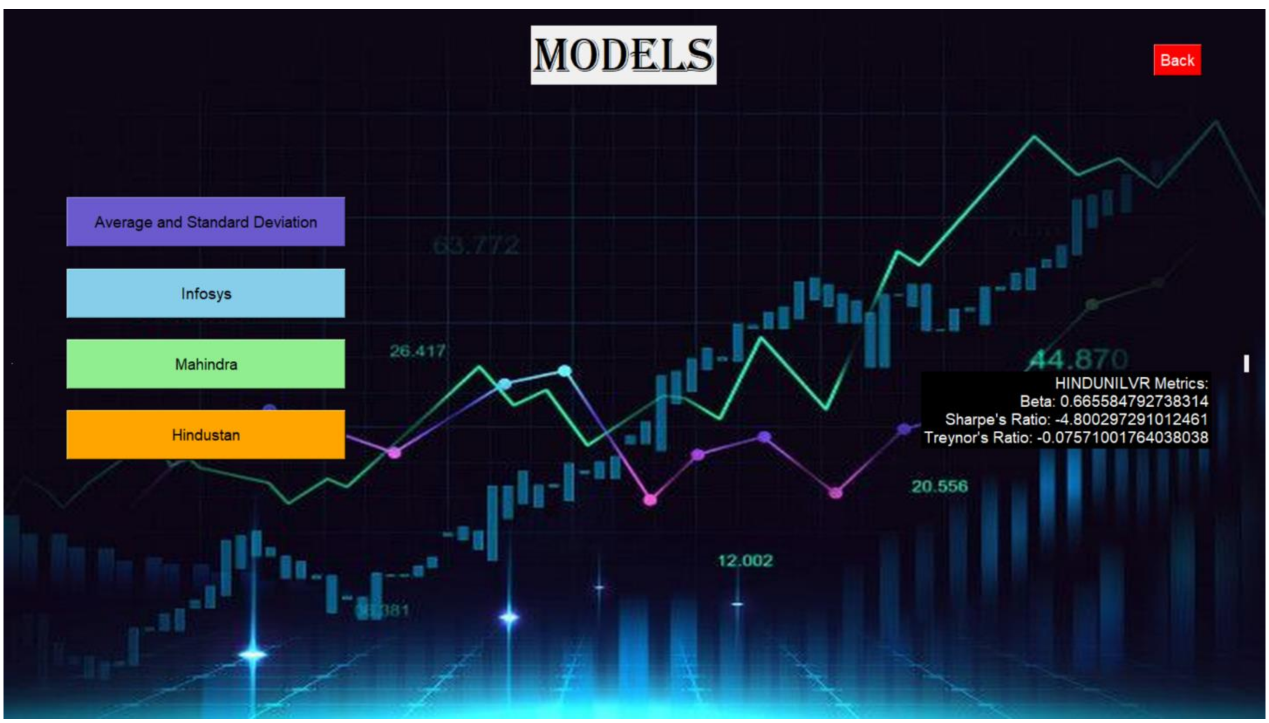

MODELS
Back
Average and Standard Deviation
Infosys
Mahindra
Hindustan
Average Market Return:
0.0007903752231860033
Market Standard Deviation:
0.006374900253600979


MODELS
Back
Figure 1
Infosys Stock Prices
Infosys Price
Date
x=2024-05 y=1490.
Average Market Return:
0.0007903752231860033
Market Standard Deviation:
0.006374900253600979


MODELS
Back
Average and Standard Deviation
Infosys
Mahindra
Hindustan
Infosys Metrics:
Beta: 1.0449609331035767
Sharpe's Ratio: -3.461621412336198
Treynor's Ratio: -0.04716474593382791


MODELS
Back
Average and Standard Deviation
Infosys
Mahindra
Hindustan
M&M Metrics:
Beta: 1.1527286954608063
Sharpe's Ratio: -2.8554279356648267
Treynor's Ratio: -0.04120396090502151


MODELS
Back
Average and Standard Deviation
Infosys
Mahindra
Hindustan
Figure 1
HINDUNILVR Stock Prices
HINDUNILVR
HINDUNILVR Price
Date
2700
2600
2500
2400
2300
2200
2023-05
2023-07
2023-09
2023-11
2024-01
2024-03
2024-05


MODELS
Back
Average and Standard Deviation
Infosys
Mahindra
Hindustan
HINDUNILVR Metrics:
Beta: 0.665584792738314
Sharpe's Ratio: -4.800297291012461
Treynor's Ratio: -0.07571001764038038

11. FINDINGS

1. Expected Returns Analysis

- The calculation of expected returns based on a blend of short-term (30-day) and long-term (365-day) performance provides valuable insights into potential investment opportunities within the Nifty 50 index.
- Companies exhibiting consistent positive returns over both short and long periods may indicate stability and growth potential, making them attractive investment prospects.
- Variations in expected returns across different companies highlight disparities in performance and market sentiment, allowing investors to prioritize investments based on their risk-return preferences.

2. Risk Assessment and Mitigation

- The determination of the risk-free rate serves as a critical benchmark for evaluating the attractiveness of investment opportunities within the Nifty 50 universe.
- Calculating excess returns relative to the risk-free rate enables investors to assess the premium earned for undertaking market risk, aiding in risk-adjusted decision-making.
- Integration of sophisticated models like the Treynor and Sharpe ratios provides a quantitative framework for evaluating the risk-adjusted returns of individual companies, guiding investors towards optimal investment choices.

3. Technical Analysis and Market Trends

- Incorporating technical analysis tools such as the Moving Average Convergence Divergence (MACD) enhances the ability to identify market trends and optimal entry and exit points for option trades.
- The MACD histogram, in conjunction with the signal line, offers valuable insights into bullish and bearish trends, facilitating timely decision-making in option trading strategies.
- Utilizing MACD signals in combination with other technical indicators can further refine trading strategies and improve overall profitability in Nifty 50 option trading.

4. Exploratory Data Analysis (EDA) Insights

- Distribution plots, time series analysis, and bar charts of expected returns provide comprehensive insights into the structure and patterns within the Nifty 50 dataset.
- EDA techniques aid in identifying outliers, trends, and seasonal patterns, enabling investors to make informed decisions based on a thorough understanding of the underlying data.
- Scatter plots offer valuable insights into the correlation between market returns and individual company returns, facilitating risk assessment and portfolio diversification strategies.

12. FUTURE ENHANCEMENTS

1. Advanced Machine Learning Models

- Implementation of advanced machine learning algorithms can enhance predictive modeling capabilities, enabling more accurate forecasts of expected returns and risk assessment.
- Utilizing techniques such as ensemble learning, neural networks, and deep learning can capture complex relationships and non-linear patterns in financial data, improving decision-making processes.

2. Real-Time Data Analysis

- Integration of real-time data feeds and analytics platforms allows for continuous monitoring of market trends and timely adjustment of trading strategies.
- Leveraging cloud computing and big data technologies enables scalable and efficient processing of large volumes of financial data, enhancing responsiveness and agility in decision-making.

3. Sentiment Analysis and News Mining

- Incorporating sentiment analysis and news mining techniques can provide insights into market sentiment and investor behavior, augmenting traditional financial analysis with qualitative data.
- Natural language processing (NLP) algorithms can extract sentiment from news articles, social media feeds, and other textual sources, offering valuable context for interpreting market dynamics.

4. Robust Risk Management Framework

- Development of a robust risk management framework tailored to the complexities of Nifty 50 option trading ensures prudent capital allocation and mitigation of downside risks.
- Integration of scenario analysis, stress testing, and portfolio optimization techniques enables proactive risk management and resilience to market uncertainties.

5. Enhanced Visualization and User Experience

- Improving visualization techniques and user interfaces enhances accessibility and usability of analytical tools for investors, fostering better decision-making and engagement.

- Interactive dashboards, customizable reports, and intuitive data visualization libraries offer a more immersive and personalized experience, empowering investors with actionable insights.