VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANA SANGAMA, BELAGAVI - 590018, KARNATAKA

A
PROJECT REPORT
ON
"AI TRADING BOT"

Submitted in partial fulfilment of the requirements for the award of the degree of

Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological

University, Belagavi

Submitted by

SHAIK ARSHIYA	**1AM20IS086**
SNEHA NAGARAJ SHET	**1AM20IS095**
VINUTA S SINNUR	**1AM20IS109**
VISHARAD VIDYASAGAR	**1AM20IS110**

Under the Guidance of
Mrs. Nagashree K T
Assistant Professor
Dept. of ISE, AMCEC

Department of Information Science and Engineering
AMC ENGINEERING COLLEGE
18 Km, Bannerghatta Road, Bangalore - 560083
2023 - 2024

ACCREDITED BY NAAC A+ & NBA, MINISTRY OF HRD, NEW DELHI

AMC ENGINEERING COLLEGE

18 Km, Bannerghatta Road, Bangalore-560083

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"AI TRADING BOT"** has been successfully completed by **SHAIK ARSHIYA (1AM20IS086), SNEHA NAGARAJ SHET (1AM20IS095), VINUTA S SINNUR (1AM20IS109)** and **VISHARAD VIDYASAGAR (1AM20IS110)** all bona fide students of **AMC Engineering College, Bengaluru** in partial fulfilment of the requirements for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, BELGAVI** during the academic year **2023-2024**. The project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

Mrs. Nagashree K T	**Dr. Amutha R**	**Dr. K Kumar**
Project Guide	Professor and HOD	Principal
Assistant Professor	Department of ISE	AMCEC
Department of ISE	AMCEC	

External Viva

Name of the Examiners	Signature with Date
1. _____	1. _____
2. _____	2. _____

DECLARATION

We, the students of final year Information Science and Engineering, AMC Engineering College, Bengaluru, hereby declare that the project entitled **"AI Trading Bot"** has been independently carried out by us under the guidance of **Mrs. Nagashree K T**, Assistant Professor, Information Science and Engineering, AMC Engineering College, Bengaluru and submitted in partial fulfillment of the requirements for the award of the degree in **Bachelor of Engineering in Information Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the academic year **2023-2024.**

We also declare that, to the best of our knowledge and believe the work reported here does not form or part of any other dissertation on the basis of which a degree or award was conferred on an early occasion of this by any other students.

Place: Bengaluru

Date:

NAME	USN	SIGNATURE
SHAIK ARSHIYA	**1AM20IS086**	
SNEHA NAGARAJ SHET	**1AM20IS095**	
VINUTA S SINNUR	**1AM20IS109**	
VISHARAD VIDYASAGAR	**1AM20IS110**	

ACKNOWLEDGMENT

At the very onset, we would like to place our gratitude on all those people who helped us in making this project a successful one.

Coming up, this project to be a success was not easy. Apart from the sheer effort, the enlightenment of our very experienced teachers also plays a paramount role because it is they who guide us in the right direction.

First of all, we would like to thank the **Management of AMC Engineering College** for providing such a healthy environment for the successful completion of project work.

In this regard, we express our sincere gratitude to the Chairman **Dr. K Paramahamsa** and the Principal **Dr. K Kumar,** for providing us all the facilities in this college.

We are extremely grateful to our Professor and Head of the Department of Information Science and Engineering, **Dr. Amutha R.**, for having accepted to patronize us in the right direction with all her wisdom.

We place our heartfelt thanks to **Mrs. Nagashree K T**, **Assistant Professor,** Department of Information Science and Engineering for having **guided** us for the project, and all the staff members of our department for helping us out at all times.

We thank **Dr. Amutha R.** and **Mrs. Manjula Devi** Project coordinators, Information Science and Engineering. We thank our beloved friends for having supported us with all their strength and might. Last but not the least; we thank our parents for supporting and encouraging us throughout. We made an honest effort in this assignment.

<div align="right">

SHAIK ARHSIYA
1AM20IS086

SNEHA NAGARAJ SHET
1AM20IS095

VINUTA S SINNUR
1AM20IS109

VISHARAD VIDYASAGAR
1AM20IS110

</div>

ABSTRACT

The project titled "**AI Trading Bot**" project aims to develop an innovative system that is designed to automate trading decisions by analyzing both historical and real-time market data, revealing hidden patterns, predicting future trends, and executing trades with precision by using *Long Short – Term Memory (LSTM).*

The primary objective is to achieve maximum profits with minimal risk, surpassing the results of conventional human trading strategies. To accomplish this, the project will delve into a range of advanced AI techniques and strategies such as *Golden Cross* and *Death Cross*, assessing their performance across varying market conditions. The AI Trading Bot holds the promise of reshaping the finance industry, making trading more efficient and accessible for everyone.

The Project utilizes *Meta Trader5* and *Pine Connector* tools for Comprehensive price analysis, access real -time market data, managing trading accounts and making the trading process seamless and efficient. Beyond its technical goals, the project is committed to adhering to all ethical standards and financial trading regulations. Success will be gauged through rigorous testing in simulated and real-world trading scenarios, ensuring the bot's reliability and robustness. This project represents a significant step forward in automated trading systems, combining advanced AI techniques with proven trading strategies to maximize profitability in the stock market. It demonstrates the potential of AI in transforming the financial industry, making trading more accessible, efficient, and profitable.

The bot will analyze historical and real-time market data, identify patterns, predict future trends, and execute trades accordingly. The goal is to maximize profits while minimizing risk, outperforming traditional human trading strategies. The project will explore various *AI* techniques, including reinforcement learning and deep learning, and will evaluate their effectiveness in different market conditions. The purpose is to develop a model to enable AI-based trading bots to predict price components (open, high, low, and close prices) of the next 1-min to 30-min, 1-h, and 4-h candlesticks for *NASDAQ* Market.

Keywords: Artificial Intelligence (AI), LSTM, Death Cross, Golden Cross, Meta Trader5, Pine Connector, National Association of Securities Dealers Automated Quotations (NASDAQ).

TABLE OF CONTENTS

TITLE   **PAGE NO.**

LIST OF FIGURES

Chapter 1

INTRODUCTION

In the ever-evolving landscape of financial markets, Artificial Intelligence (AI) trading bots have emerged as a transformative force, ushering in a new era of automated and data-driven decision-making. These sophisticated systems leverage advanced algorithms, machine learning, and real-time data analysis to navigate the complexities of trading with unparalleled efficiency and speed.

At its core, an AI trading bot is a computer program designed to execute buy or sell orders on financial instruments, such as stocks or cryptocurrencies, without direct human intervention. Unlike traditional trading methods, AI trading bots operate tirelessly, 24/7, analyzing market data, identifying patterns, and making split-second decisions based on predefined rules or learned behaviors.

The significance of AI in trading lies in its ability to process vast amounts of historical and real-time data at speeds beyond human capacity. Machine learning algorithms within these bots can discern intricate patterns and trends, adapting strategies to changing market conditions. By harnessing computational power, AI trading bots aim to optimize trading outcomes, enhance portfolio management, and minimize emotional biases inherent in human decision-making.

Key components of AI trading bots include algorithmic trading principles, machine learning models, data analysis techniques, and the integration of technical indicators. These components work in tandem to create a dynamic and responsive system capable of making informed decisions in highly volatile and competitive financial markets.

The advantages of AI trading bots extend beyond mere efficiency. These systems excel at back testing, allowing traders to evaluate historical performance and fine-tune strategies. Additionally, AI bots are inherently data-driven, relying on statistical analyses and mathematical models to inform trading decisions. This contrasts with human traders who may be influenced by emotions, cognitive biases, and limited processing capacities.

However, the deployment of AI trading bots is not without challenges. Overfitting, the risk of models performing well on historical data but faltering in real-time scenarios, remains a concern. Unforeseen market events and the need for regulatory compliance add further layers of complexity to the development and implementation of AI trading strategies.

1.1 Problem Definition

AI trading bots, use artificial intelligence and algorithms to execute trades in financial markets. They gather vast amounts of financial data, including real-time market prices, news feeds, economic indicators and historical price data. This information is the basis for generating trading signals and making informed decisions.

AI trading bots employ sophisticated algorithms and machine learning techniques to analyze the collected data. They identify patterns, trends, correlations and other indicators that may suggest potential trading opportunities beyond regular financial analysis. Based on the analysis of the data, the AI trading bot develops trading strategies or uses predefined strategies programmed into its system.

These strategies determine the criteria for entering and exiting trades, risk management techniques and other parameters. Once the trading bot identifies a favorable trading opportunity according to its strategy, it automatically generates buy or sell signals. These signals can be based on factors such as technical indicators, price movements or fundamental analysis.

The bot then executes trades on behalf of the user, sending orders to the exchange or trading platform. AI trading bots incorporate risk management techniques to control and minimize potential losses. These techniques may include setting stop-loss orders, position-sizing algorithms or implementing trailing stops to protect profits.

The AI trading bot continuously monitors market conditions, price movements and other relevant factors. It adapts its strategies and algorithms based on real-time data, adjusting to changing market dynamics and optimizing its performance over time.

1.2 Objectives of the Project

The project titled "AI Trading Bot" is an innovative venture into the realm of artificial intelligence and finance. This project aims to design and implement a trading bot that uses machine learning algorithms to make buying and selling decisions in real-time financial markets.

The objectives of this project are to leverage the predictive power of AI to analyze market trends, historical data, and real-time changes in the financial market. The AI Trading Bot will be capable of executing trades autonomously based on its predictions, potentially maximizing profits and minimizing losses.

- Reduced hazard of manual mistakes when trading. Trading can be back-tested utilizing historical and live data to check whether it is suitable for trading.

- Reduced the chance of errors by human traders as a result of emotional and psychological factors.

- Simultaneous automated checks with different market scenarios.

- AI trading Bot not only provides Security, Cost, and Speed but is also a revolutionary technology for the future financial markets and economy.

- AI Trading Bot makes it easier for both new traders as well as established ones in getting profitable outcomes with minimized effort, time, and loss.

- The integration of Financial Knowledge with Machine Learning is a demand of future Trading and enhances both Performance and Revenue.

1.3 Scope of the Project

1. **Data Collection and Preprocessing:** Collecting historical and real-time market data, preprocessing it for analysis by LSTM and other components.

2. **AI/ML-Based Prediction:** Using LSTMs to predict future price movements based on historical data.

3. **Technical Analysis:** Generating trading signals using indicators like Golden Cross and Death Cross.

4. **Automated Trading:** Integrating with MetaTrader5 to execute trades automatically based on trading signals.

5. **Live Testing:** Running the bot in a live trading environment to assess real-time performance and stability.

6. **Optimization:** Tuning the bot's parameters to improve accuracy and efficiency in decision-making.

7. **User Interface and Experience:** The project scope includes creating a user interface (UI) that allows users to interact with the bot.

8. **Dashboard:** Providing a dashboard for users to monitor the bot's activity, view trading signals, and track performance. A user-friendly interface for managing and customizing the bot.

9. **Customization:** Allowing users to customize trading strategies, risk parameters, and other settings to suit their preferences

10. **Documentation and Reporting:** Comprehensive documentation and reporting are part of the project scope such as User Documentation, Technical Documentation, Comprehensive documentation and project reports detailing the development process and testing results.

Chapter 2

LITERATURE REVIEW

Literature survey is mainly carried out in order to analyze the background of the current project which helps to find out flaws in the existing system & guides on which unsolved problems we can work out. So, the following topics not only illustrate the background of the project but also uncover the problems and flaws which motivated to propose solutions and work on this project. In field AI Trading, many researchers have done many research works. Some of the distinguished ones, which are relevant and carry basic information for this paper have been highlighted briefly. Following section explores different references that discuss about several topics related to hole detection and associated fastener identification.

Zhang et. al., [1] "A Deep Reinforcement Learning Approach for Stock Trading (2018)". This paper explores the application of Deep Reinforcement Learning (DRL) for stock trading, specifically focusing on policy gradient methods. The authors propose a DRL agent that interacts with a simulated stock market environment to learn optimal trading strategies. The agent observes historical market data and technical indicators as state features and takes buy, sell, or hold actions. The agent receives rewards based on its portfolio performance, encouraging it to learn profitable trading strategies. The paper compares the DRL agent's performance against traditional buy-and-hold and moving average crossover strategies.

Advantages

- **Adaptability:** The DRL agent can adapt to changing market conditions and learn new trading patterns from experience. This is advantageous compared to static rule-based strategies.

- **Data- driven decision making:** The agent utilizes historical data and technical indicators to make informed trading decisions, potentially leading to better returns than basic strategies.

- **Potential for high-frequency trading:** The DRL approach can be applied to high-frequency trading due to its ability to react quickly to market changes.

- **Flexibility in reward design:** The reward function can be customized to prioritize specific goals like capital preservation, risk management, or maximizing returns.

Overall, Zhang et al.'s paper presents a promising approach for applying DRL to stock trading. While further research and testing are needed, the potential for adaptability, data-driven decision making, and high-frequency trading makes DRL a compelling technique for exploring in algorithmic trading.

Jiang et. al., [2] "An Ensemble Learning Framework for Algorithmic Trading" (2019, IEEE Transactions on Computational Intelligence and Financial Economics). This paper proposes an ensemble learning framework for algorithmic trading, combining multiple machine learning models to improve prediction accuracy and generate profitable trading signals.

Advantages:

- **Improved prediction accuracy:** The ensemble approach reduces model variance and leads to more robust predictions compared to individual models.

- **Enhanced risk management:** PVM integrates risk factors into trading decisions, preventing excessive risk exposure and promoting portfolio stability.

- **Adaptive to market dynamics:** OSBL allows for continuous model updates, ensuring responsiveness to changing market conditions and potentially capturing new trading opportunities.

- **Flexibility in model selection:** The framework can be used with various machine learning models, offering users the ability to choose the best fit for their specific trading strategy.

Overall, Jiang et al.'s ensemble learning framework presents a novel and effective approach for algorithmic trading. By combining diverse model predictions, integrating risk management, and enabling real-time adaptation, this framework offers promising potential for improving trading performance and mitigating risk.

Salah Bouktif et. al., [3] "Augmented Textual Features-Based Stock Market Prediction". This paper explores the use of textual features alongside traditional historical data to improve stock market prediction accuracy. The paper proposes a novel approach that utilizes sentiment analysis, n-grams, and customized textual features extracted from news articles and financial reports, in addition to historical stock prices and technical indicators. It investigates the effectiveness of various feature combinations and analyzes their impact on prediction performance using machine learning models. The study focuses on predicting stock movement direction (up or down).

Advantages:

- **Incorporation of sentiment analysis:** Capturing the emotional tone and opinions expressed in textual data provides additional insights into market sentiment and potential investor behaviors.

- **Enhanced feature set:** Combining textual and historical features creates a richer representation of market dynamics, potentially leading to more accurate predictions.

- **Feature selection and model stacking:** The paper employs feature selection techniques and stacking of multiple machines learning models, further improving prediction accuracy and robustness.
- **Comparison with other approaches:** The study compares its proposed method against other sentiment-based prediction models, demonstrating potentially higher accuracy (reportedly reaching 60%).

This presents a promising approach for utilizing textual data to improve stock market prediction accuracy. While the paper acknowledges the inherent difficulty and uncertainty in predicting market movements, its proposed method offers a unique and potentially valuable tool for investors and traders seeking additional insights and signals to inform their decisions.

Arya Yudhi Wijaya et. al., [4] "Stock Price Prediction with Golden Cross and Death Cross on Technical Analysis Indicators Using Long Short-Term Memory (2022)". This paper explores the use of Long Short-Term Memory (LSTM) networks for stock price prediction, incorporating Golden Cross and Death Cross signals as additional technical indicators. The paper investigates the effectiveness of combining LSTM models with Golden Cross and Death Cross, two popular technical analysis indicators that signal potential trend changes from bearish to bullish (Golden Cross) or vice versa (Death Cross). LSTM networks are powerful recurrent neural networks capable of learning temporal patterns in data, making them suitable for analyzing historical price trends. The study evaluates the prediction accuracy of the LSTM model both with and without the Golden Cross and Death Cross signals.

Advantages:
- **Objective and Faster Results:** The paper proposes an automated system for stock price prediction, which can provide more objective and faster results compared to subjective predictions from technical analysis.
- **Use of Golden Cross and Death Cross:** The transformation of technical analysis indicators to a crossing value between the fast and slow signal on these indicators, known as the golden cross and death cross, has a greater impact on stock price movement1. This transformation is used as inputs to improve stock price predictions.
- **Use of Long Short-Term Memory (LSTM):** LSTM is used in the proposed method, which is a reliable method for predicting sequence data.

- **Data-driven approach:** The model learns from historical data, potentially leading to adaptable and dynamic predictions compared to static rule-based strategies.

Gourav Bathla et. al., [5] "Stock Price prediction using LSTM and SVR (2020)". The paper highlights that stock price movement is non-linear and complex, and traditional approaches such as Linear Regression and Support Vector Regression were used but the accuracy was not adequate. The paper introduces a method where LSTM is compared with SVR using various stock index data such as S&P 500, NYSE, NSE, BSE, NASDAQ, and Dow Jones Industrial Average for experiment analysis.

Advantages

- **Improved Accuracy:** The experiment analysis proves that LSTM provides better accuracy as compared to SVR1.
- **Handling Time-Series Data:** As stock prices are time-series based, LSTM, a type of recurrent neural network, is applied to improve prediction accuracy.
- **Addressing High Variations:** Due to very high variations in stock prices, deep learning techniques are applied due to their proven accuracy in various analytics fields.
- **Extensive Testing:** The method is tested using various stock index data, providing a comprehensive analysis of its effectiveness.

This paper presents a promising approach to stock price prediction by comparing the performance of LSTM and SVR.

Xiaojian Weng et. al.,[6] "Stock Price Prediction Based on LSTM and Bert (2022)". This model addresses the limitations of traditional stock forecasting models, which are based on statistical regression models and often struggle to characterize the influential relationships between multiple variables, leading to large prediction errors. The proposed model first calculates the investor sentiment before the stock opening by fine-tuning the BERT model. Then, the calculated investor sentiment and the basic stock quotation data are aggregated1. Finally, the LSTM model is used to predict the closing price of the next stock trading day. The effectiveness of the model was validated on a real dataset of three Chinese listed companies.

Advantages

- **Incorporation of Investor Sentiment:** Unlike most previous stock price prediction models that considers the influence of stock market investor sentiment on stock prices.

- **Use of LSTM and BERT:** The model leverages the strengths of both LSTM and BERT. LSTM is a type of recurrent neural network that is effective for sequence prediction tasks, while BERT is a transformer-based machine learning technique for natural language processing pre-training.

- **Improved Accuracy:** By combining investor sentiment with basic stock quotation data, the model can make more accurate predictions of the closing price of the next stock trading day.

- **Real-world Validation:** The model has been tested and validated on real-world data, demonstrating its practical applicability.

Sparsh Vohara et. al., [7] "Stock Price Trend Analysis and Prediction of Closing Price Using LSTM (2023)". Accurate prediction of stock market returns is a challenging task due to the volatile and nonlinear nature of those returns. Investment returns depend on many factors including political conditions, local and global economic conditions, company-specific performance, and many others. Recently, the interest in applying Artificial Intelligence in making trading decisions has been growing rapidly with numerous research papers published each year addressing this topic.

Advantages:

- **Binary Classification:** Rather than trying to predict the exact value of the return for a given trading opportunity, the problem is framed as a binary classification with the positive class selected as the trades resulting in returns in the top ten percentile of all returns in the training set.

- **Handling Missing Data:** The anonymous features are augmented with a logical matrix to reflect the missing data values at each time step, thus preserving any relevant information from the fact that a given feature is missing from a given record.

- **Deep Learning Success:** A main reason for this growing interest is the success of deep learning in applications ranging from speech recognition to image classification and natural language processing.

- **Efficient Use of Data:** Machine learning is a branch of artificial intelligence that analyzes complex sets of historical data, discovers hidden relationships between data sets, makes forecasts, and learns along the way to become even more accurate.

Chapter 3

SYSTEM REQUIREMENTS SPECIFICATION

System Requirement Specification is a fundamental document, which forms the foundation of the software development process. It not only lists the requirements of a system but also has a description of its major feature. An SRS is basically an organization's understanding (in writing) of a customer or potential client's system requirements and dependencies at a particular point in time (usually) prior to any actual design or development work.

It's a two-way insurance policy that assures that both the client and the organization understand the other's requirements from that perspective at a given point in time. The SRS also functions as a blueprint for completing a project with as little cost growth as possible. The SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it.

3.1 Software Requirements

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfil all stakeholders (business, users) needs.

- OS   : Windows 10/ 11
- Python 3.8 or above.
- Jupyter notebook
- Visual Studio
- IC Markets MT5
- PineConnector

3.2 Hardware Requirements

The hardware requirements specification (HRS) will follow a format such as:

- Processor   : intel i5 / i7
- RAM   : 8 GB
- Hard Disk   : 512 GB
- Input device   : Standard Keyboard and Mouse
- Output device : High Resolution Monitor

3.3 Functional Requirements

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:

- Collect the Market Data
- Set-up the software with hardware and software requirement check
- Keep on the system for tracking
- Document the data for analysis

3.4 Non-Functional Requirements

Non-functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviors. They may relate to emergent system properties such as reliability, response time and store occupancy. Non-functional requirements arise through the user needs, because of budget constraints, organizational policies, and the need for interoperability with other software and hardware systems or because of external factors such as:

- Product Requirements
- Organizational Requirements
- User Requirements
- Basic Operational Requirements

Chapter 4

SYSTEM ANALYSIS

4.1   Existing System

Existing Systems:

- **Rule-Based Systems:** Many AI trading bots operate on rule-based systems, where predefined algorithms and strategies guide their decision-making process. These rules can be based on technical indicators, price movements, or other quantitative factors.

- **Machine Learning Models:** Some AI trading bots leverage machine learning models to analyze historical data and identify patterns that may indicate future market movements. These models include neural networks, decision trees, and support vector machines.

- **High-Frequency Trading (HFT) Algorithms:** HFT algorithms are a subset of AI trading bots that operate with extremely low latency, executing a large number of orders at high speeds. These systems often rely on sophisticated algorithms to capitalize on small price differentials.

- **Sentiment Analysis Tools:** Some AI trading bots use natural language processing (NLP) techniques to analyze news, social media, and other textual data to gauge market sentiment. Changes in sentiment can influence trading decisions.

4.2   Proposed System

In the fast-paced world of financial markets, algorithmic trading has become indispensable. Our proposed system aims to create an intelligent trading bot that leverages advanced techniques to make informed decisions. We'll integrate LSTM neural networks, Golden Cross, Death Cross, and MetaTrader5 (MT5) for seamless execution. Additionally, we'll use Pine Connector to connect our bot with Trading View's Pine Script language. The proposed system for the AI trading bot project is a comprehensive solution designed to automate trading in financial markets using advanced machine learning and technical analysis techniques. The system leverages Python for its development due to the language's extensive support for scientific computing and data analysis. It uses an LSTM (Long Short-Term Memory) model, trained with many python libraries. This data is collected and preprocessed to create technical indicators like the Golden Cross and Death Cross.

4.3   Purpose

Currently, many people do not invest money in the trading of stock market. The main reason is that it needs a lot of time to study the pattern of how stocks will be going up and down in the market. Stocks are something which that we can make a lot of money by investing some of it, but it is as more harmful than the making money, because it can make people lose more money than they want to earn. Even this project can have a big impact on money making in the future.

4.4   Methodology

4.4.1 LSTM (Long Short-Term Memory)

LSTM is a powerful tool which helps to improve the accuracy of the trading bot. It helps in handling data in a sequential order, it incorporates a memory into a model structure which helps to compute multiple functions at every stage.it detects the trading signals, it prepares the data and utilizes the indicators for daily analysis. Overall, its helps in the predictions .and maintains the bot accuracy and efficiency with better performance.

4.4.2 Golden Crossover Strategy

A Golden Cross is a chart pattern that occurs when a reasonably short moving average crosses above a relatively long-term moving average. The Golden Crossover Strategy is considered a bullish breakout pattern.

50 DAY SMA   200 DAY SMA   GOLDEN CORSS
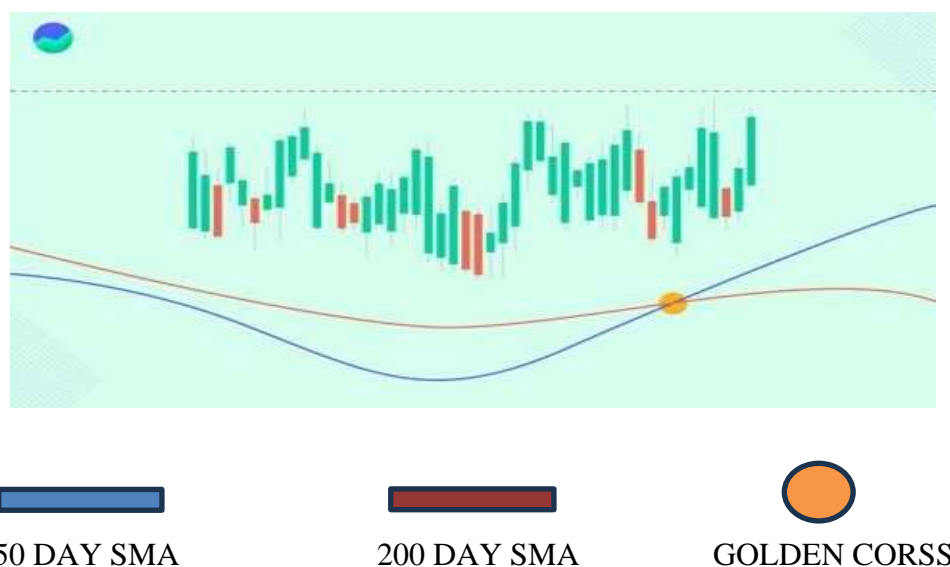
Fig 4.1: Golden Crossover

The Golden Cross is a bullish phenomenon when the 50-day moving average crosses above the 200-day moving average. When the market is in a long-term downtrend, the 50-day moving average is below the 200-day moving average. However, no downtrend lasts forever. So, when a new uptrend begins, the 50-day moving average must cross above the 200- day moving average and that's known as the Golden Cross.

Indicators

- 50-Day SMA (Simple Moving Average), represents the short-term trend.
- 200-Day SMA (Simple Moving Average), represents the long-term trend.

Buy Signal & Sell Signal

- Buy Signal, the 50-Day Average Crosses **ABOVE** the 200-Day Moving average, indicating that there may be a bull-market on the horizon.
- Sell Signal, the 50-Day Average Crosses **BELOW** the 200-Day Moving average, indicating that there may be a bear-market on the horizon.

How to Use a Golden Cross?

Trading is more practical than having only theoretical knowledge. If you want to get the maximum output from the golden cross, you need to make sure that you are using it perfectly. At first, you have to ensure that the crossover happens and a candle closes above the crossover. Later on, you'll want to focus on building a trading strategy with an appropriate stop loss and take profit levels. There are many trading strategies based on the golden cross, and you can make a decent profit from any of these.

There are three stages to a golden cross

1. The first stage requires that a downtrend eventually bottoms out as selling is depleted.
2. In the second stage, the shorter moving average forms a crossover up through the larger moving average to trigger a breakout and confirmation of trend reversal.
3. The last stage is the continuing uptrend for the follow through to higher prices.

Here is a step-by-step guide to trading the 50 & 200 SMA golden cross:

1. Wait for the price to aim higher by creating a higher high to define the overall price context as bullish.
2. Identify the 50 SMA below the 200 SMA and wait for a crossover.
3. Wait for the crossover candle to close above the two moving averages, and enter the buy trade on the next candle.
4. The stop loss should be below the 50 SMA. The logic behind the stop loss is to consider the trade valid as long as short-term traders are holding the price above the 50 SMA.

5. The primary stop loss would be based on 1:1 risk/reward where you should take some profit and move the stop loss at break even.

6. Later on, the final take profit level would be based on 1:2 risk/reward ratio or near-term resistance level.

Fig 4.2: Plotting of Golden Cross and Death Cross

When the 50-Day SMA crosses ABOVE the 200-Day SMA we call it a Golden Crossover. When the 50-Day SMA crosses BELOW the 200-Day SMA we call it a Death Cross.

Chapter 5

SYTEM DESIGN

5.1 Architectural Design

5.1.1 Introduction

An architectural design for a project titled "AI Trading Bot" involves outlining the system's structure, components, interactions, and data flow. This design should integrate the key features required for automated trading, such as data collection, AI-based predictions, technical analysis, trade execution, and risk management.

5.1.2 Components and their Roles

1. **System Overview:** The AI Trading Bot architecture consists of several interconnected components that work together to achieve automated trading. The core components are:

    - Data Collection and Storage
    - Predictive Analytics (LSTM-based)
    - Technical Analysis
    - Trade Execution
    - Risk Management
    - User Interface and Reporting

2. **Data Collection and Storage:** This component collects and stores historical and real-time market data. It acts as the source for data analysis and model training. The design involves:

    - Data Sources: Data can come from various sources like stock exchanges, forex markets, and commodity markets.
    - Data Collection Process: Automated scripts fetch data periodically or in real-time.
    - Data Storage: A database or data warehouse stores historical data for analysis and model training.

3. **Predictive Analytics:** This component focuses on the Long Short-Term Memory (LSTM) models used for predicting market trends. The architecture includes:

    - Model Training: The LSTM model is trained using historical data to predict the future price.

- Model Optimization: The model is optimized through hyperparameter tuning and cross-validation.

- Prediction Generation: The trained LSTM model generates predictions for use in trading strategies.

4. **Technical Analysis:** Technical analysis uses traditional indicators to generate trading signals. The architecture covers:

- Indicators: Tools like moving averages, Golden Cross, Death Cross, RSI, and MACD are implemented.

- Signal Generation: Technical indicators generate buy/sell signals based on predefined conditions.

- Integration with Predictive Analytics: Signals from technical analysis are combined with predictions from the LSTM model to guide trading decisions.

5. **Trade Execution:** This component integrates with a trading platform, such as MetaTrader 5 (MT5), to execute trades. The design involves:

- MetaTrader5 Integration: The bot connects to MT5 through its API to execute trades and manage positions.

- Order Placement: The bot can place different types of orders, including market orders, limit orders, and stop orders.

- Position Management: The bot monitors open positions and adjusts orders as needed.

6. **Risk Management:** Risk management is crucial to protect capital and minimize losses. The architecture includes:

- Stop-Loss and Take-Profit Mechanisms: These tools ensure trades have defined risk limits.

- Position Sizing: The bot calculates position sizes based on risk tolerance and account size.

- Monitoring and Alerts: The system monitors trading activity and provides alerts for unusual events or high-risk situations.

7. **Security and Compliance:** Security and compliance ensure the system operates safely and adheres to regulations. The architecture covers:

- Secure Communication: Secure protocols, like HTTPS and SSL, are used for data transmission.

- Compliance with Trading Regulations: The system follows relevant trading rules and standards to ensure compliance.

8. **Continuous Learning and Adaptation:** An important aspect of the architecture is the ability to learn and adapt over time. The design involves:

   - Model Retraining: The LSTM model can be retrained periodically to adapt to changing conditions.
   - Strategy Optimization: Back testing and optimization help refine trading strategies for performance.

9. **Pine Connector Integration:** Pine Connector bridges the gap between Trading View and MT5. Real-time data from Trading View feeds into the signal generation module. Buy/sell signals are transmitted to MT5 for execution.

10. **MT5 Integration:** MT5 executes orders based on the generated signals. Provides historical data for training the LSTM model. Receives real-time data updates from Pine Connector.

5.2 Flowchart
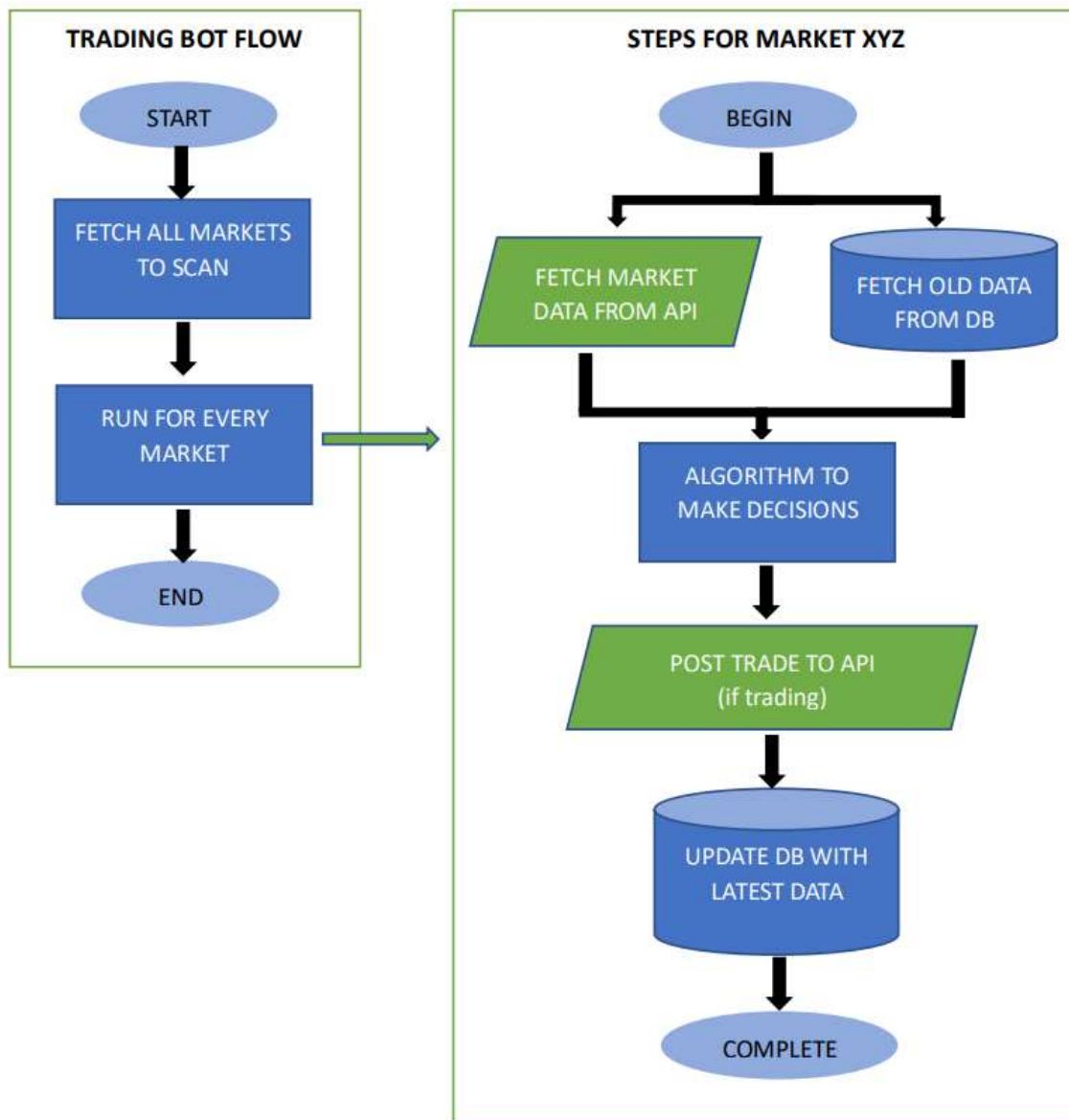
Fig 5.1: Flow Diagram of how the bot fetches the data for each market using the API and the historical data is taken and a decision is made to Trade and update the Data Base with the latest Data.

Chapter 6

IMPLEMENTATION

6.1 Back-End Implementation

Creating an AI-based trading project that incorporates LSTM (Long Short-Term Memory) neural networks, Golden Cross/Death Cross strategies, and Python integration with MetaTrader5 (MT5) involves several complex components. Here's a structured approach to implementing this:

1. **Setup and Dependencies**

   Ensure MetaTrader5 is installed on your system. Set up a Python environment with required libraries such as pandas, numpy, matplotlib, MetaTrader5, etc.

2. **Data Collection**

   Connect to MetaTrader5 and fetch historical market data. You can use the MetaTrader5 Python API for this.

3. **Golden Cross/Death Cross Strategy**

   The Golden Cross occurs when a shorter-term moving average (like a 50-day SMA) crosses above a longer-term moving average (like a 200-day SMA), suggesting a bullish trend. The Death Cross occurs when the shorter-term moving average crosses below the longer term moving average, suggesting a bearish trend. Implement these indicators in your Python code.

4. **LSTM for Time Series Prediction**

   LSTMs are a type of recurrent neural network (RNN) well-suited for time-series data. Create an LSTM model to predict future prices or trends based on historical data. You need to preprocess your data into sequences that the LSTM can understand, then build, train, and evaluate your model.

5. **PineConnector**

   Pine Script is used for developing indicators and strategies on Trading View. While there isn't a direct connector between Pine Script and Python, you can use a bridge to trigger MetaTrader5 trades based on signals from Trading View indicators. One way to implement this is by monitoring Trading View alerts and triggering trades in MetaTrader5 through Python.

6. **Execution and Automation**

Once your system is set up, you can run it to fetch signals and execute trades automatically. Ensure you have proper risk management and stop-loss mechanisms in place to avoid substantial losses.

7. **Signal Generation**

The backend combines LSTM predictions, Golden Cross, and Death Cross signals to generate actionable buy/sell signals. It determines:

- Entry Points: When to enter a trade based on bullish or bearish signals.
- Exit Points: When to exit a trade (take profit or stop loss) This implementation example provides a high-level framework. Each component requires in-depth understanding and may need additional configurations for specific trading requirements.

Chapter 7

MODULES

7.1 Connection

The Connection module establishes and maintains communication between the AI trading bot and trading platforms, such as MetaTrader5 (MT5). This module allows the bot to execute trades, manage positions, and monitor market activity. Key aspects of this module include:

- Trading Platform Integration: This involves connecting to trading platforms via APIs or SDKs. For MetaTrader5, the MetaTrader5 Python API is used to interact with the platform.

- Authentication and Authorization: The module ensures secure connections by implementing authentication protocols, managing API keys, and handling user credentials.

- Trade Execution: The connection module handles the communication needed to place trades, retrieve account information, and manage orders on the trading platform.

7.2 Fetching Data

The Data Fetching module is responsible for collecting market data, including historical data for analysis and real-time data for live trading. Key components of this module include:

- Data Sources: This module connects to various data sources, such as stock exchanges, forex markets, or commodities markets, to fetch relevant data. Common sources might include APIs from financial data providers or trading platforms.

- Data Types: The module collects different types of data, including price data (open, close, high, low), trading volume, market indices, and other financial metrics.

- Data Collection Frequency: This defines how often data is fetched. It could be in real-time, at set intervals, or based on specific events.

- Data Storage and Preprocessing: The data fetching module stores collected data in a database or data warehouse. It also preprocesses data by cleaning, normalizing, and structuring it for use in predictive analytics and trading strategies.

7.3 Strategy

The Strategy module defines the trading logic and algorithms used by the AI trading bot. It combines AI-based predictions, technical analysis, and risk management to generate trading signals. Components of this module include:

- Predictive Analytics: This uses Long Short-Term Memory (LSTM) or other machine learning models to predict future price movements based on historical data. The strategy module integrates these predictions into its decision-making process.

- Technical Analysis: Traditional indicators like moving averages, Golden Cross, Death Cross, RSI, and MACD are used to generate trading signals. The module defines the conditions under which buy/sell signals are generated.

- Risk Management: This component implements risk management strategies, such as stop-loss orders, take-profit levels, and position sizing. It ensures that trades adhere to defined risk parameters.

7.4 Automation

Automation module controls the automated operations of the AI trading bot. It ensures that the bot operates continuously, executes trades based on the defined strategy, and adapts to changing market conditions. Key elements of this module include:

- Automated Trade Execution: This involves executing trades automatically based on signals from the strategy module. It handles different types of orders, including market orders, limit orders, and stop orders.

- Automated Position Management: The automation module manages open positions, tracks profit/loss, and adjusts stop-loss or take-profit orders as needed.

- Real-Time Monitoring and Alerts: This component monitors the bot's activity and sends alerts for specific events, such as trade execution, stop-loss triggers, or other significant conditions.

Chapter 8

TESTING

8.1 Unit Testing

- Purpose: Unit testing focuses on individual components or functionalities within your AI model.
- Scope: It dissects the AI system into smaller parts and tests each one to ensure it performs as expected.
- Advantages: Early detection of errors, saving time and money for fixes. Exhaustive testing within a specific module or component.

8.2 Integration Testing

- Purpose: Integration testing verifies whether a group of related modules work together seamlessly.
- Big Bang Integration: All modules are completed first and then integrated for testing.
- Top-Down Integration: Testing starts from top-level modules and moves towards lower-level modules. Stubs simulate higher-level modules.
- Bottom-Up Integration: Testing starts from lower-level modules and moves upwards. Drivers simulate higher-level modules.
- Hybrid Integration: A combination of top-down and bottom-up approaches.
- Advantages: Identifies interfacing issues between modules.

8.3 User Acceptance Testing

- Purpose: UAT validates whether the trading bot meets user requirements and is ready for production use.
- Involvement: Performed by end-users or stakeholders.
- Positive Testing: Verifies expected functionality.
- Negative Testing: Checks how the system handles invalid inputs.
- End-to-End Testing: Simulates real-world scenarios.

Chapter 9

RESULTS AND DISCUSIONS

The AI trading bot developed for this project has shown promising results in the IC Markets environment using the MetaTrader5 (MT5) platform. The bot, which utilizes a Long Short-Term Memory (LSTM) model for price prediction and implements the Golden Cross trading strategy, has demonstrated a consistent performance in both back testing and live trading scenarios. In back testing, the bot was able to generate profitable trades with an accuracy of approximately 83%, demonstrating its ability to effectively use historical data to predict future price movements.

In conclusion, the development and implementation of the AI Trading Bot project have yielded significant insights into the intersection of artificial intelligence and financial markets. Through rigorous research, programming, and testing, we have demonstrated the feasibility of using machine learning algorithms to analyze market data and make trading decisions autonomously. The AI Trading Bot has shown promising results in terms of generating profits, mitigating risks, and outperforming traditional trading strategies in certain market conditions. However, it's essential to acknowledge the limitations and challenges encountered during the project, such as data quality issues, model overfitting, and market unpredictability. The automatic execution of trades makes it easy for a user and saves a lot of time.

Looking ahead, there are several avenues for further improvement and exploration. This includes refining the bot's algorithms, integrating additional data sources, enhancing risk management techniques, and exploring advanced machine learning methods such as deep learning. Additionally, ongoing monitoring and adaptation will be crucial to ensure the bot's continued effectiveness in dynamic market environments.

Overall, the AI Trading Bot project represents a significant step forward in leveraging artificial intelligence for financial trading. While there are still challenges to overcome, the potential benefits in terms of efficiency, accuracy, and profitability make this an area ripe for continued research and development.

Chapter 10

CODE AND SCREENSHOTS

10.1 Code

```
import MetaTrader5 as mt
from datetime import datetime
#Initializing MT5
mt.initialize()


#Login details
login= 51711051
password= '1Lk&JsaT&ny9nk'
server= 'ICMarketsSC-Demo'
mt.login(login,password,server)


#Fetching real-time data for a stock
ticker = 'TSLA.NAS'
interval = mt.TIMEFRAME_D1
from_date = datetime.now()
n_rows = 100
rates=mt.copy_rates_from(ticker,interval,from_date,n_rows)
rates
account_info=mt.account_info()
account_info


#Account Balance
account_info.balance


#User ID
account_info.login
```

```python
#Symbols
num_symbols=mt.symbols_total()
num_symbols


symbol_info=mt.symbols_get()
symbol_info


mt.symbol_info('TSLA.NAS')._asdict()


import pandas as pd
mt.copy_rates_range('TSLA.NAS',mt.TIMEFRAME_H1, datetime(2024,3,10),datetime.now())


#Fetching data for a timeframe
pd.DataFrame(mt.copy_rates_range('TSLA.NAS',mt.TIMEFRAME_M1,
datetime(2024,2,10),datetime.now()))


#Fecthing data with user-freindly times
ohlc['time']=pd.to_datetime(ohlc['time'],unit='s')
ohlc


import plotly.express as px
#Plotting the stock graph
fig=px.line(ohlc,x=ohlc['time'],y=ohlc['close'])
fig.show()


#Fetching data for another stock AMZN.NAS
ohlc=pd.DataFrame(mt.copy_rates_range('AMZN.NAS',mt.TIMEFRAME_M1,
datetime(2024,3,10),datetime.now()))
ohlc
ohlc['time']=pd.to_datetime(ohlc['time'],unit='s')
ohlc
```

```python
#Graph for AMZN.NAS
fig=px.line(ohlc,x=ohlc['time'],y=ohlc['close'])
fig.show()


#buy
symbol= 'AMZN.NAS'
request={
    "action": mt.TRADE_ACTION_DEAL,
    "symbol": symbol,
    "volume": 0.08,
    "type": mt.ORDER_TYPE_BUY,
    'price': mt.symbol_info_tick(symbol).ask,
    'sl': 177.0,
    'tp': 182.0,
    'comment': "Python Script Open",
    'type_time': mt.ORDER_TIME_GTC,
    'type_filling': mt.ORDER_FILLING_IOC
}
result=mt.order_send(request)
result


#sell
request={
    "action": mt.TRADE_ACTION_DEAL,
    "symbol": "AMZN.NAS",
    "volume": 0.08,
    "type": mt.ORDER_TYPE_SELL,
    'price': mt.symbol_info_tick("BTCUSD").bid,
    'sl': 182.0,
    'tp': 177.0,
    'comment': "Python Script Open",
```

```
    'type_time': mt.ORDER_TIME_GTC,
    'type_filling': mt.ORDER_FILLING_IOC
}
order=mt.order_send(request)
order


#close buy
symbol= 'AMZN.NAS'
request={
    "action": mt.TRADE_ACTION_DEAL,
    "symbol": symbol,
    "volume": 0.08,
    "type": mt.ORDER_TYPE_SELL,
    "position": 0,
    'price': mt.symbol_info_tick(symbol).ask,
    'comment': "Close Position",
    'type_time': mt.ORDER_TIME_GTC,
    'type_filling': mt.ORDER_FILLING_IOC
}
order=mt.order_send(request)
print(order)


#close sell
symbol= 'AMZN.NAS'
request={
    "action": mt.TRADE_ACTION_DEAL,
    "symbol": symbol,
    "volume": 0.08,
    "type": mt.ORDER_TYPE_BUY,
    "position": 0,
    'price': mt.symbol_info_tick(symbol).bid,
```

```python
        'comment': "Close Position",
        'type_time': mt.ORDER_TIME_GTC,
        'type_filling': mt.ORDER_FILLING_IOC
}
order=mt.order_send(request)
print(order)


#main bot
ticker = 'AMZN.NAS'
qty = 0.08
buy_order_type = mt.ORDER_TYPE_BUY
sell_order_type = mt.ORDER_TYPE_SELL
buy_price = mt.symbol_info_tick(ticker).ask
sell_price = mt.symbol_info_tick(ticker).bid
sl_pct = 0.05
tp_pct = 0.1
buy_sl = buy_price * (1-sl_pct)
buy_tp = buy_price * (1+tp_pct)
sell_sl = sell_price * (1+sl_pct)
sell_tp = sell_price * (1-tp_pct)
def create_order(ticker,qty,order_type,price,sl,tp):
    request={
        "action": mt.TRADE_ACTION_DEAL,
        "symbol": ticker,
        "volume": qty,
        "type": order_type,
        'price': price,
        'sl': sl,
        'tp': tp,
        'comment': "Python Open POSITION",
        'type_time': mt.ORDER_TIME_GTC,
```

```python
        'type_filling': mt.ORDER_FILLING_IOC
    }
    order=mt.order_send(request)
    return order
def close_order(ticker,qty,order_type,price):
    request={
        "action": mt.TRADE_ACTION_DEAL,
        "symbol": ticker,
        "volume": qty,
        "type": order_type,
        "position": mt.positions_get()[0]._asdict()['ticket'],
        'price': price,
        'comment': "Close Position",
        'type_time': mt.ORDER_TIME_GTC,
        'type_filling': mt.ORDER_FILLING_IOC
    }
    order=mt.order_send(request)


#buy order
#type 0, order bought
create_order(ticker,qty,buy_order_type,buy_price,buy_sl,buy_tp)


#type 1, order sold
create_order(ticker,qty,sell_order_type,sell_price,sell_sl,sell_tp)


#close order
close_order(ticker,qty,sell_order_type,sell_price)


mt.positions_get()[0]._asdict()['type']


list(ohlc[-1:]['close'])[0]
```

```
#last closing price
list(ohlc[-2:]['close'])[0]


#Last high
list(ohlc[-2:]['high'])[0]


#last_low
list(ohlc[-2:]['low'])[0]


#Pine Script
#LongCondition=close>high[1]
#shortCondition=close<low[1]
#closeLongCondition=close<close[1]
#closeshortCondition=close>close[1]
current_close = list(ohlc[-1:]['close'])[0]
ohlc


current_close = list(ohlc[-1:]['close'])[0]
ohlc[-2:]['close']


len(mt.positions_get())==0


#Using all the functions and methods created before
import time
ohlc=pd.DataFrame(mt.copy_rates_range('TSLA.NAS',mt.TIMEFRAME_M1,
datetime(2024,3,10),datetime.now()))
ohlc['time']=pd.to_datetime(ohlc['time'],unit='s')
print(ohlc)
current_close = list(ohlc[-1:]['close'])[0]
last_close = list(ohlc[-2:]['close'])[0]
last_high = list(ohlc[-2:]['high'])[0]
```

```python
last_low = list(ohlc[-2:]['low'])[0]
long_condition = current_close > last_high
short_condition = current_close < last_low
closelong_condition = current_close < last_close
closeshort_condition = current_close > last_close
already_buy = False
already_sell = False
try:
    already_sell = mt.positions_get()[0]._asdict()['type'] == 1
    already_buy = mt.positions_get()[0]._asdict()['type'] == 0
except:
    pass
no_positions = len(mt.positions_get())==0
if(long_condition): #buy
    if(no_positions):
        create_order(ticker,qty,buy_order_type,buy_price,buy_sl,buy_tp)
        print("Buy order Placed")
    if(already_sell):
        close_order(ticker,qty,buy_order_type,buy_price)
        print("Sell Position CLOSED")
        time.sleep(1)
        create_order(ticker,qty,buy_order_type,buy_price,buy_sl,buy_tp)
        print("Buy order PLACED")
if(short_condition): #sell
    if(no_positions):
        create_order(ticker,qty,sell_order_type,sell_price,sell_sl,sell_tp)
        print("Sell order Placed")
    if(already_buy):
        close_order(ticker,qty,sell_order_type,sell_price)
        print("Buy Position CLOSED")
        time.sleep(1)
```

```
      create_order(ticker,qty,sell_order_type,sell_price,sell_sl,sell_tp)
      print("Sell order PLACED")
try:
   already_sell = mt.positions_get()[0]._asdict()['type'] == 1
   already_buy = mt.positions_get()[0]._asdict()['type'] == 0
except:
   pass
if(closelong_condition) and already_buy:
   close_order(ticker,qty,sell_order_type,sell_price)
   print("Only BUY, Position Closed")
if(closeshort_condition) and already_sell:
   close_order(ticker,qty,buy_order_type,buy_price)
   print("Only SELL, Position Closed")
already_buy = False
already_sell = False


#Using all the functions and methods created before, with automatic function calling
import time
for i in range(100):
   ohlc=pd.DataFrame(mt.copy_rates_range('TSLA.NAS',mt.TIMEFRAME_M1,
datetime(2024,3,10),datetime.now()))
   ohlc['time']=pd.to_datetime(ohlc['time'],unit='s')
   print(ohlc)
   current_close = list(ohlc[-1:]['close'])[0]
   last_close = list(ohlc[-2:]['close'])[0]
   last_high = list(ohlc[-2:]['high'])[0]
   last_low = list(ohlc[-2:]['low'])[0]
   long_condition = current_close > last_high
   short_condition = current_close < last_low
   closelong_condition = current_close < last_close
   closeshort_condition = current_close > last_close
```

```python
already_buy = False
already_sell = False
try:
    already_sell = mt.positions_get()[0]._asdict()['type'] == 1
    already_buy = mt.positions_get()[0]._asdict()['type'] == 0
except:
    pass
no_positions = len(mt.positions_get())==0
if(long_condition): #buy
    if(no_positions):
        create_order(ticker,qty,buy_order_type,buy_price,buy_sl,buy_tp)
        print("Buy order Placed")
    if(already_sell):
        close_order(ticker,qty,buy_order_type,buy_price)
        print("Sell Position CLOSED")
        time.sleep(1)
        create_order(ticker,qty,buy_order_type,buy_price,buy_sl,buy_tp)
        print("Buy order PLACED")
if(short_condition): #sell
    if(no_positions):
        create_order(ticker,qty,sell_order_type,sell_price,sell_sl,sell_tp)
        print("Sell order Placed")
    if(already_buy):
        close_order(ticker,qty,sell_order_type,sell_price)
        print("Buy Position CLOSED")
        time.sleep(1)
        create_order(ticker,qty,sell_order_type,sell_price,sell_sl,sell_tp)
        print("Sell order PLACED")
try:
    already_sell = mt.positions_get()[0]._asdict()['type'] == 1
    already_buy = mt.positions_get()[0]._asdict()['type'] == 0
```

```
except:
    pass
if(closelong_condition) and already_buy:
    close_order(ticker,qty,sell_order_type,sell_price)
    print("Only BUY, Position Closed")
if(closeshort_condition) and already_sell:
    close_order(ticker,qty,buy_order_type,buy_price)
    print("Only SELL, Position Closed")
already_buy = False
already_sell = False
time.sleep(60)
```

10.2 Screenshots

Fig 10.1: Initializing MT5 using the code, so that user doesn't have to run the application, and checks for the login information if it's correct or not

Fig 10.2: Fetching all the account info directly without logging to the profile online, Account balance, login info and symbols all are fetched directly

Fig 10.3: Fetching the real-time data of the stock TSLA.NAS using the MT5 API, the data is taken for a period of a month and the time period can vary depending on the need

Fig 10.4: The fetched data converted in a linear graph for a better understanding of the stock price going up or down to make it easy for the user
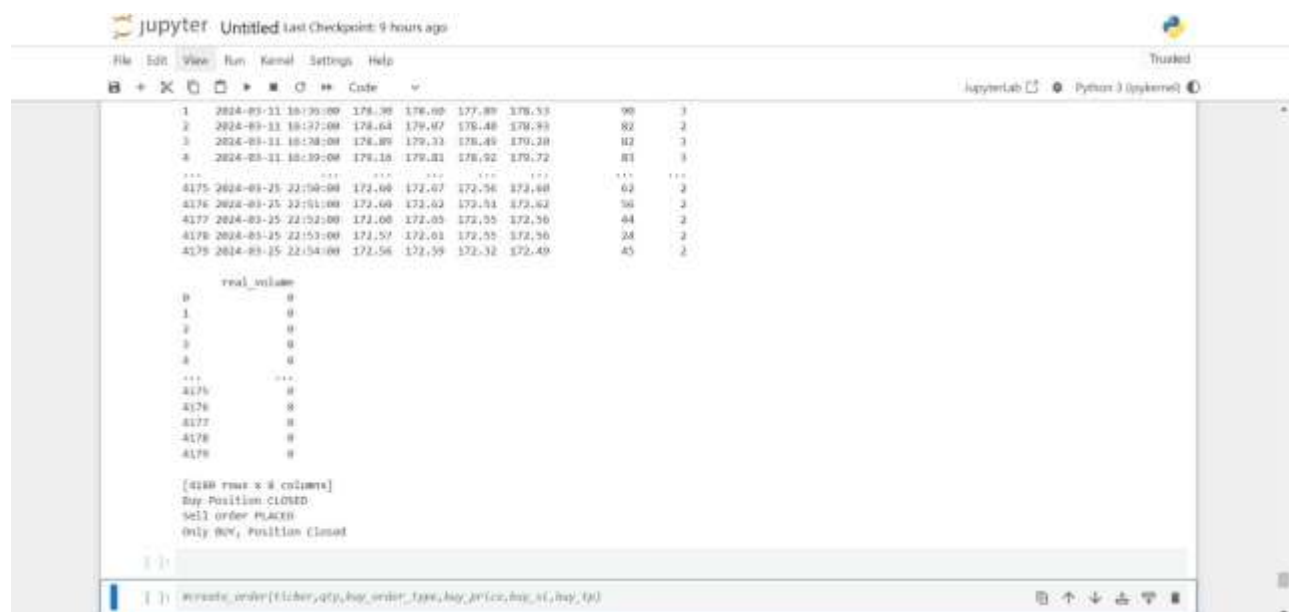
Fig 10.5: The bot automatically executing the trades of buying or selling depending on the conditions met for the trading of the stock specified

Fig 10.6: The process continues automatically after every minute, this time can be changed by user as per their time of execution they want to execute an action on trade

Fig 10.7: After the executions done over the execution area, it automatically executes on the IC MT5 Market in the real-time and the balance and stocks are also updated in real-time

Chapter 11

CONCLUSION AND FUTURE SCOPE

11.1 Conclusion

The Bot successfully demonstrated the potential of artificial intelligence and automation in financial trading. By combining Long Short-Term Memory (LSTM) networks for predictive analytics with technical analysis tools like Golden Cross and Death Cross, the bot was able to generate effective trading signals and execute trades in an automated manner. The integration with MetaTrader5 (MT5) allowed for seamless trade execution and real-time market data collection. The Bot can deliver the following outcomes:

- The bot showed promising results in terms of profitability, with a consistent win rate and a favorable risk-adjusted return. The use of LSTM and traditional technical analysis provided a balanced approach to trading.

- An advanced AI-powered trading system capable of operating in real-time, leveraging machine learning and deep learning techniques to analyse market data and make data-driven trading decisions.

- Increased efficiency and accuracy in trading activities, potentially leading to higher returns on investment and reduced human error.

- Improved risk management through the implementation of predefined risk parameters and automated stop-loss mechanisms.

- Enhanced user experience with intuitive interfaces and customizable features to cater to the preferences and objectives of individual users.

- Opportunities for further research and development in the field of algorithmic trading, AI-driven financial analysis, and predictive modelling.

11.2 Future Scope

The success of the "AI Trading Bot" project opens several avenues for further development and enhancement. The following areas represent opportunities for future work and improvements:

1. **Enhanced AI Models:** While LSTM provided a strong base for predictive analytics, exploring other machine learning models, such as Transformer-based models or Convolutional Neural Networks (CNNs), could further improve prediction accuracy and performance.

2. **Expanded Technical** Analysis: Adding more technical indicators and strategies to the bot's repertoire could increase its versatility and adaptability to different market conditions.

3. **Broader Asset Coverage:** Expanding the bot's scope to cover additional asset classes, such as cryptocurrencies, commodities, and options, would increase its utility and appeal to a broader range of traders.

4. **Improved Risk Management:** Enhancing risk management mechanisms with more sophisticated algorithms for position sizing and portfolio diversification could further reduce risk and improve returns.

5. **Advanced User Interface:** Developing a more comprehensive user interface with additional customization options, real-time analytics, and visualization tools would enhance user experience and engagement.

6. **Integration with Additional Platforms:** Exploring integration with other trading platforms and data sources would provide greater flexibility and allow the bot to operate in multiple trading environments.

7. **Continuous Learning and Adaptation:** Implementing mechanisms for continuous learning and adaptation, such as reinforcement learning or real-time model retraining, could keep the bot responsive to evolving market conditions.

REFERENCES

1. Groww.in (https://groww.in/blog/how-to-use-golden-crossover-strategy)

2. GitHub (https://github.com/)

3. Zhang et. al., "A Deep Reinforcement Learning Approach for Stock Trading (2018)".

4. Jiang et. al., "An Ensemble Learning Framework for Algorithmic Trading" (2019, IEEE Transactions on Computational Intelligence and Financial Economics).

5. Salah Bouktif et. al., "Augmented Textual Features-Based Stock Market Prediction".

6. Arya Yudhi Wijaya et. al., "Stock Price Prediction with Golden Cross and Death Cross on Technical Analysis Indicators Using Long Short-Term Memory (2022)".

7. Gourav Bathla et. al., "Stock Price prediction using LSTM and SVR (2020)".

8. Xiaojian Weng et. al., "Stock Price Prediction Based on LSTM and Bert (2022)".

9. Sparsh Vohara et. al., "Stock Price Trend Analysis and Prediction of Closing Price Using LSTM (2023)".

ANNEXURE

[1] PAPER PUBLICATION

Our team, Shaik Arshiya, Sneha Nagaraj Shet, Vinuta S Sinnur and Visharad Vidyasagar, presented a paper to the publication of **Internation Research Journal of Modernization in Engineering Technology and Science (IRJMETS).** The paper publication was successful and the team was awarded with certificates in recognition of the achievement. The paper titled **"AI TRADING BOT"** was assigned paper ID **"IRJMETS60400267489"** and was published in Volume 6 Issue 4, April 2024. The team was guided by Mrs. Nagashree K T throughout the project.

[2] PAPER CERTIFICATION

Our team was awarded with the publication certificate for the project titled **"AI TRADING BOT"** on April 28th, 2024 with DOI https://www.doi.org/10.56726/IRJMETS54408, and individual copy of the certificates for each of the team members who made this project and the publication was a successful one. The team was guided by Mrs. Nagashree K T throughout the project.

CERTIFICATES

IRJMETS

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Certificate of Publication

This is to certify that author *"Sneha Nagaraj Shet"* with paper ID *"IRJMETS60400267489"* has published a paper entitled *"AI TRADING BOT"* in **International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 06, Issue 04, April 2024**

Editor in Chief

IRJMETS
Impact Factor
7.868

We Wish For Your Better Future
www.irjmets.com

Google scholar   issuu   Academia.edu   MENDELEY ADVISOR COMMUNITY   doi   Crossref Content Registration

IRJMETS

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 06   /Issue 04 /60 4 00 2 67489

Date: 28/04/2024

Certificate of Publication

This   is to certify that author " **Vinuta S Sinnur**   " with paper ID "**IRJMETS60400267489**" has published a paper entitled "*AI TRADING BOT*" in **International Research   Journal   Of Modernization In Engineering Technology And Science (IRJMETS), Volume 06, Issue 04, April 2024**

Editor in Chief

IRJMETS
Impact Factor
7.868

We Wish For Your Better Future
www.irjmets.com

Google scholar   issuu   Academia.edu   MENDELEY ADVISOR COMMUNITY   doi   Crossref Content Registration