A SYNOPSIS ON

STOCK PREDICTION

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

BACHELOR OF TECHNOLOGY

In

Computer Science & Engineering

Submitted by:

Divyarth Sah 2261639

Priyanshu Bhatt 2261441

Paras Punetha 2261411

Lalit Tiwari 2261333

Under the Guidance of

Mr. Prince Kumar

Professor

Project Team ID: 77 ID No.

Department of Computer Science & Engineering

Graphic Era Hill University, Bhimtal, Uttarakhand

March-2025

Graphic Era Hill University
BHIMTAL CAMPUS

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the Synopsis entitled **"Stock Prediction** in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science & Engineering of the Graphic Era Hill University, Bhimtal campus and shall be carried out by the undersigned under the supervision of **Mr. Prince Kumar, Professor**, Department of Computer Science & Engineering, Graphic Era Hill University, Bhimtal.

Divyarth Sah	2261639
Priyanshu Bhatt	2261441
Paras Punetha	2261411
Lalit Tiwari	2261333

The above mentioned students shall be working under the supervision of the undersigned on the **"Stock Prediction"**

Signature
Supervisor

Signature
Head of the Department

Internal Evaluation (By DPRC Committee)

Status of the Synopsis: Accepted / Rejected

Any Comments:

Name of the Committee Members: **Signature with Date**

1.

2.

Table of Contents

Chapter 1

Introduction and Problem Statement

1. Introduction

The stock market, known for its dynamic behavior, has intrigued economists, investors, and researchers for decades. Predicting stock prices is a formidable task due to the involvement of numerous volatile and unpredictable factors such as global economic indicators, political events, company performance, and market sentiment. Traditional prediction methods often rely on linear models or statistical techniques that struggle to capture the complexities of such a non-linear, multidimensional problem.

Recent advances in Machine Learning (ML) have provided more robust approaches to model these complexities. Simultaneously, developments in system-level software engineering have introduced new ways to optimize computational workloads. Our project sits at the confluence of these two domains—Operating Systems (OS) and Machine Learning—by building a high-performance forecasting system that utilizes multiprocessing and intelligent scheduling.

This project leverages multiprocessing to train and execute multiple ML models in parallel, thereby reducing runtime and increasing throughput. We also design a priority-based scheduler inspired by OS principles, ensuring efficient and fair resource utilization. The system aims to be modular, extendable, and suitable for integration with real-time data pipelines and GUI interfaces, offering practical utility for both academic research and real-world financial applications.

2. Problem Statement

While machine learning models have shown promise in predicting stock market trends, most current systems suffer from inefficiencies due to sequential execution. The lack of parallel processing and intelligent scheduling mechanisms results in longer processing times, limited scalability, and poor utilization of computational resources.

Furthermore, existing systems are often rigid, not easily modifiable, and difficult to integrate with advanced analytics or visualization tools. There is also an absence of a systematic approach to manage multiple prediction tasks, each with varying importance and urgency.

Our project proposes a multiprocessing-enabled, priority-scheduled stock prediction framework that overcomes these limitations. By running multiple ML models simultaneously and managing them through a simulated OS-style scheduler, the system ensures faster processing, better resource management, and a foundation for scalable enhancements.

.

Chapter 2

Background/ Literature Survey

Stock price prediction is a long-standing challenge in financial analytics. Traditional techniques like moving averages, autoregressive models, and ARIMA are limited in their capacity to model nonlinear relationships. With the advent of ML, techniques like Support Vector Machines (SVM), Random Forests, and Neural Networks have gained popularity due to their adaptability and improved accuracy.

Simultaneously, Operating Systems (OS) play a crucial role in efficient program execution through features like multiprocessing, process scheduling, and memory management. The multiprocessing module in Python allows parallel execution of tasks, which is highly advantageous in computationally heavy tasks like ML training and prediction.

Literature reveals that most ML-based stock prediction systems are built as monolithic scripts lacking concurrency or scheduling. Researchers have demonstrated that multiprocessing can reduce training time significantly, especially when dealing with large datasets and multiple models. Meanwhile, OS-level scheduling algorithms like Round Robin, Priority Scheduling, and Shortest Job First (SJF) are well-established in optimizing CPU utilization.

Our system brings these domains together by using OS principles to design a custom scheduler that executes prediction tasks concurrently. It incorporates priority levels to reflect task importance and uses an aging mechanism to prevent indefinite postponement, ensuring fairness. This integration enhances performance and offers a scalable, adaptable solution for predictive analytics**.**

Chapter 3

Objectives

The objectives of the project are as follows:

1. To develop a stock prediction system that integrates Operating System concepts with machine learning techniques.
2. To implement multiprocessing that allows concurrent execution of multiple prediction models.
3. To design a task scheduler inspired by OS principles like priority scheduling and aging.
4. To use Linear Regression, Random Forest, and Support Vector Regression (SVR) as core predictive models.
5. To collect, preprocess, and clean historical stock market data using Python libraries.
6. To visualize prediction outputs through graphs and comparative analytics.
7. To ensure modularity and scalability for future enhancements such as GUI integration, LSTM-based forecasting, and real-time data ingestion.
8. To evaluate system performance improvements in terms of execution time, accuracy, and resource utilization.

Chapter 4

Hardware and Software Requirements

4.1 Hardware Requirements

Sl. No	**Name of the Hardware**	**Specification**
1	Processor	Intel i5 or above
2	RAM	8 GB minimum
3	Storage	256 GB SSD or higher
4	GPU (optional)	NVIDIA GTX 1050 or better

These specifications are adequate for executing multiprocessing tasks and handling data preprocessing and visualization efficiently.

4.2 Software Requirements

Sl. No	Name of the Software	Specification
1	Operating System	Windows 10 / Linux Ubuntu
2	Programming Language	Python 3.9 or later
3	Libraries	scikit-learn, pandas, matplotlib, numpy
4	IDE	VS Code / Jupyter Notebook

The software stack is selected to ensure compatibility with open-source tools, enabling easy maintenance, extension, and community support.

The software stack is selected to ensure compatibility with open-source tools, enabling easy maintenance, extension, and community support.

Chapter 5

Possible Approach/ Algorithms

System Architecture:

The system consists of the following modular components:

- User Input Module: Captures stock symbol, forecast horizon, and preferred models.
- Data Handler: Downloads historical stock prices, handles missing values, and formats data for modeling.
- Scheduler: Implements a custom algorithm to assign priorities to tasks and applies an aging mechanism.
- Multiprocessing Engine: Launches each model in an isolated process for concurrent execution.
- Machine Learning Models:
    - *Linear Regression*: Serves as a baseline due to its simplicity and speed.
    - *Random Forest*: Uses ensemble learning to provide robust and stable predictions.
    - *Support Vector Regression (SVR)*: Suitable for nonlinear data patterns.
- Result Aggregator: Gathers outputs from all processes and compares model performance using graphs and metrics.

Scheduling Algorithm:

Initialize task queue with processes and priorities

While task queue is not empty:

Sort tasks by priority (higher first)

For each task:

If task waiting time exceeds threshold:

Increase task priority (aging)

Execute the highest-priority task using multiprocessing

Retrieve and store prediction results

References

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 2nd ed., MIT Press.

[2] Open Source Computer Vision Library (OpenCV). [Online]. Available: https://opencv.org/

[3] Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/

[4] J. Brownlee, Machine Learning Mastery with Python, Machine Learning Mastery, 2016.

[5] Real Python Tutorials – Multiprocessing in Python. [Online]. https://realpython.com/

[6] Python multiprocessing official documentation. [Online]. https://docs.python.org/3/library/multiprocessing.html

[7] Ensemble Methods in Machine Learning, Springer.

[8] P. J. Rousseeuw and A. M. Leroy, Robust Regression and Outlier Detection, Wiley.