PHASE-1 PROJECT DOCUMENT SUBMISSION

TOPIC : CRACKING THE MARKET CODE WITH AI-DRIVEN STOCK PRICE PREDICTION USING TIME SERIES ANALYSIS.

Student Name: KEERTHIVERAJ P

Register Number: 620123106054

Institution: AVS ENGINNERING COLLEGE, AMMAPET, SALEM.

Department: ELECTRONICS AND COMMUNICATION ENGINEERING

Date Of Submission: 30.04.2025

---

1. Problem Statement

The stock market is characterized by high volatility and complexity, making accurate price forecasting a significant challenge. Market prices are influenced by a wide range of dynamic factors—including economic indicators, company performance, geopolitical events, and investor sentiment—that can lead to sudden and unpredictable movements. Traditional forecasting methods (such as simple statistical models or linear regression) often struggle to capture the non-linear and time-dependent patterns present in stock price data. By leveraging advanced AI-driven techniques and time series analysis, this project aims to improve predictive accuracy and provide better tools for informed investment decisions and risk management.

2. Objectives of the Project

- **Predict future stock prices using AI-driven models:** Develop models that learn from historical data to forecast future stock prices (e.g., closing prices) with improved accuracy.

- **Evaluate and compare multiple forecasting models:** Implement and test a variety of approaches (such as ARIMA, LSTM, Prophet, and XGBoost) to determine which model best predicts stock price movements.

- **Identify influential features:** Analyze which input features or technical indicators (such as lagged prices, moving averages, RSI, MACD, etc.) most significantly impact the model's predictions.

- **Generate actionable insights:** Interpret model results to provide practical insights for investors or stakeholders, highlighting trends and key factors driving price changes.

3. Scope of the Project

- The project will focus on **analyzing historical stock price data** for selected publicly traded companies. Key data fields will include **open, high, low, close prices, and trading volume**.

- All data will be sourced from **publicly available datasets** (e.g., Yahoo Finance). The project will not use any proprietary or confidential data, ensuring the work is reproducible and transparent.

- The analysis will be **limited to offline data exploration and modeling**. It will **not** involve actual trading, real-time data feeds, or live execution of trades. As a result, the study will not provide direct financial advice or investment recommendations.

- The project's scope is confined to **time series forecasting and analysis**; emphasis will be on methodology, model development, and evaluation rather than deployment in a production trading environment.

4. Data Sources

- **Yahoo Finance API:** Historical daily stock data (prices and volume) will be collected using Python libraries like `yfinance`. This source provides up-to-date market information for many publicly traded companies.

- **Kaggle Datasets:** Publicly available CSV datasets from Kaggle (or similar platforms) will be used as an alternative or supplementary data source. These datasets typically offer historical price snapshots that can be manually updated.

- Both sources are **public and free to use**, ensuring transparency. Data integrity and version control will be maintained to keep track of updates (for example, noting the date of data download or refresh).

5. High-Level Methodology

- **Data Collection** – Gather historical stock price data (including date, open, high, low, close, and volume) for the target stocks over a significant period. This can be done via the Yahoo Finance API (using tools like yfinance) or by downloading relevant datasets from Kaggle. Ensure that the dataset covers various market conditions (e.g., several years of daily data) to support robust modeling.

- **Data Cleaning** – Preprocess the collected data to ensure quality and consistency. This includes handling missing values (for example, by interpolation or removal of incomplete records), removing any duplicate entries, and correcting format inconsistencies (such as ensuring that date fields are properly formatted and numerical fields are correctly typed). This step ensures the dataset is ready for analysis and modeling.

- **Exploratory Data Analysis (EDA)** – Conduct EDA to uncover underlying patterns and trends in the data. Create visualizations such as line charts of historical prices, moving average overlays to smooth trends, and volatility plots. Use statistical tools like autocorrelation (ACF) and partial autocorrelation (PACF) plots to detect serial correlations and help determine appropriate lag orders for modeling.

- **Feature Engineering** – Generate new features that may improve model predictions. This includes constructing lagged price values (e.g., previous day's closing price), rolling statistics (such as moving averages or rolling standard deviations), and technical indicators (e.g., Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD)). These features help capture market momentum, trend strength, and cyclical behaviors in the data.

- **Model Building** – Develop and train multiple predictive models to forecast future stock prices. This includes:

  ARIMA (Auto Regressive Integrated Moving Average): A classical statistical model suitable for univariate time series forecasting.

  LSTM (Long Short-Term Memory networks): A type of recurrent neural network that can learn long-term dependencies in sequential data.

  Prophet: A time series forecasting library developed by Facebook (Meta), robust to missing data and trend changes, often used for business forecasting.

  XGBoost: An ensemble gradient boosting framework that can be applied to time series data by treating it as a regression problem with engineered features.
  Each model will be trained on the historical dataset (using appropriate train-test splits, such as training on past data and validating on a hold-out period) to learn from past price patterns

- **Model Evaluation** – Evaluate each model's forecasting accuracy using appropriate metrics. Common evaluation metrics include Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). Compare the models on these metrics using a test set or cross-validation to determine which approach performs best. Analyze prediction errors over time to assess models' reliability in different market conditions.

- **Visualization & Interpretation** – Present and interpret the results through visualizations and analysis. Plot actual vs. predicted price series for each model to visually assess accuracy and trends. Use feature importance charts (for models that support it) to highlight which inputs most influence predictions. Optionally, create interactive charts or dashboards (for example, using Plotly or Streamlit) to allow stakeholders to explore the forecasts and insights. Summarize key findings (such as notable prediction errors or periods of high volatility) in clear visual and textual formats to convey actionable insights.

- **Deployment** – As an optional extension, develop a simple user interface using Streamlit (or a similar framework) to demonstrate the forecasting model. The application could allow a user to select a stock symbol and date range, then display historical data alongside model predictions. This deployment will showcase the model's practical application, but actual execution of trades or real-time integration is outside the scope of this project.

6. Tools and Technologies

- **Programming Language:** Python (chosen for its extensive libraries and community support in data science).

- **Environment/IDE:** Google Colab or Jupyter Notebook (for interactive development, visualization, and sharing).

- **Data Manipulation & Analysis:** `pandas`, `numpy` (for data handling and numerical computations).

- **Visualization:** `matplotlib`, `seaborn`, or `plotly` (for creating charts and graphs during EDA and results presentation).

- **Machine Learning & Modeling:** `scikit-learn` (utilities and metrics), `statsmodels` (for ARIMA modeling), `TensorFlow/Keras` (for building LSTM networks), `prophet` (for the Prophet model), and `xgboost` (for tree-based modeling).

- **Data Access:** `yfinance` or `pandas_datareader` (to fetch historical stock data from Yahoo Finance).

- **Optional Deployment: Streamlit (for building an interactive web dashboard), GitHub (for version control and code sharing).**

7. Team Members and Roles

- **Gobinath P** contributes across all project stages, from data collection and cleaning to model development (ARIMA, LSTM), evaluation, visualization, and documentation.
- **Keerthiveraj P** shares equal responsibility in data preprocessing, feature creation, model training, performance analysis, visualization, and report preparation.
- **Arun S** collaborates fully in every aspect of the project, including data gathering, preprocessing, modeling, evaluation, and documentation.
- **Kavin D** actively engages in all phases, including data handling, model building (ARIMA, LSTM), performance evaluation, visualization, and documentation.