Github Link:
https://github.com/ananthi678/stock-price

Project Title: Cracking the Market Code with AI-Driven Stock Price Prediction Using Time Series Analysis

PHASE-2

1. Problem Statement

Predicting stock prices is a longstanding challenge in financial markets due to the inherent volatility and non-linear patterns in market behavior. This project aims to harness artificial intelligence, particularly time series analysis models, to predict future stock prices based on histoReal financial data. Accurate stock forecasting can aid investors in making informed decisions, enhancing portfolio performance, and managing financial risk.

The project employs deep learning and statistical models to build a robust predictive system using real-world stock market data. This is a regression problem, with the target variable being the stock's closing price over time.

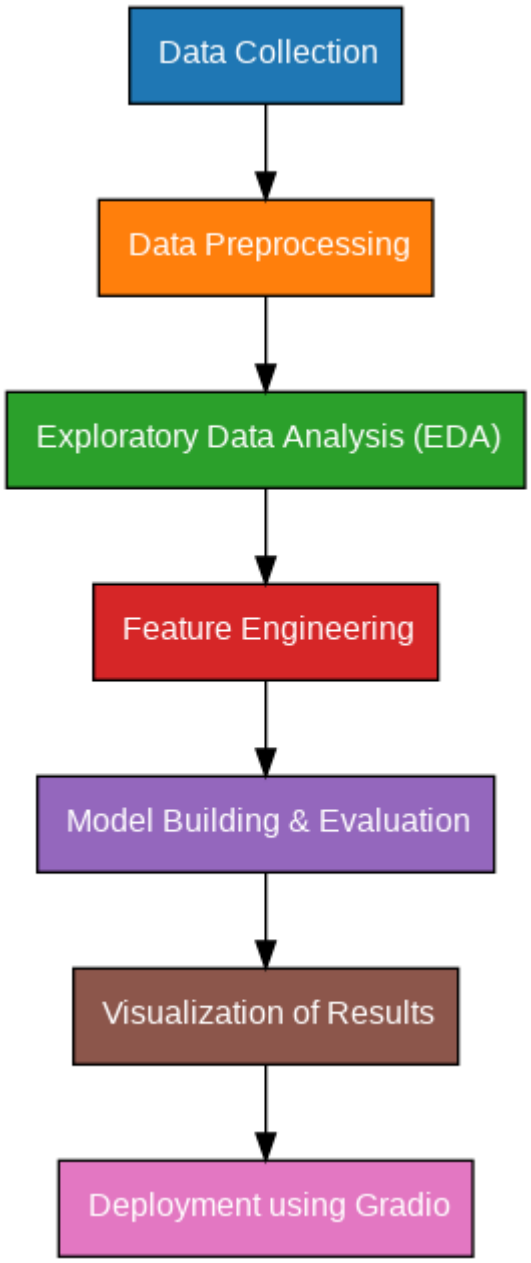
The significance of solving this problem lies in its potential application in algorithmic trading, risk management, and financial advisory services.

2. Project Objectives

- Develop machine learning models to forecast future stock prices using historical data.
- Apply time series models like ARIMA, LSTM, and Prophet for comparative analysis.
- Engineer relevant features like technical indicators (e.g., moving averages, RSI).

- Visualize trends, seasonality, and residuals for model interpretability.
- Deploy a Gradio-based interface for interactive prediction.
- Identify factors contributing to stock price movements

3. Flowchart of the Project Workflow



4. Data Description

- Dataset Source: Yahoo Finance / Alpha Vantage / Kaggle Stock Market Datasets
- Type of Data: Time series (temporal, sequential)
- Attributes: Date, Open, High, Low, Close, Volume; derived indicators like SMA, EMA, MACD, RS

- Target Variable: Closing Price (numerical, continuous)
- Static or Dynamic: Dynamic dataset (can be updated in real-time)
- Dataset Link: https://www.kaggle.com/datasets/jvanark/nvidia-daily-stock-pricedata

5. Data Preprocessing

- Converted date column to datetime type.
- Handled missing values using forward fill and interpolation.
- Scaled features using MinMaxScaler or StandardScaler.
- Created lag features and rolling window statistics.
- Ensured stationarity using differencing (for ARIMA).
- Split data into training and test sets chronologically.

6. Exploratory Data Analysis (EDA)

- Trend Analysis: Line plots of closing prices over time
- Volatility Visualization: Bollinger bands and standard deviation windows
- Correlation Heatmaps: To explore interdependence among indicators
- Seasonal Decomposition: Using STL to observe trend, seasonality, and residual

7. Feature Engineering

- Added lag features (t-1, t-2, …, t-n)
- Calculated moving averages (SMA, EMA)
- Generated technical indicators: MACD, RSI, Bollinger Bands

- Extracted temporal features: day, month, weekday
- Differenced data to remove trends for ARIMA

8. Model Building

Models Used:

ARIMA: Baseline traditional model

LSTM (Long Short-Term Memory): Captures long-range dependencies

Facebook Prophet: For handling seasonality and holidays

Model Selection Rationale:

LSTM: Suitable for sequential data with memory

ARIMA: Strong for univariate forecasting

Prophet: Simple and effective for business forecasting needs

Train-Test Split:

Chronologically split (e.g., 80% training, 20% test)

Evaluation Metrics:

Mean Absolute Error (MAE)

Root Mean Squared Error (RMSE)

Mean Absolute Percentage Error (MAPE)

9. Visualization of Results & Model Insights

- Forecast Plots: Actual vs Predicted closing prices

- Residual Diagnostics: Plots of error over time

- Feature Importance: For tree-based or hybrid models

- LSTM Training Curves: Loss vs Epoch plots

- User Testing: Interactive Gradio interface for inputting parameters and viewing predictions

10. Tools and Technologies Used

- Language: Python 3

Notebook Environment: Google Colab / Jupyter Libraries:

Pandas, numpy: Data manipulation

Matplotlib, seaborn, plotly: Visualization

Scikit-learn: Preprocessing and evaluation

Statsmodels, fbprophet, keras/tensorflow: Modeling

Gradio: Interface deployment

11. Team Members and Contributions

S. Nasreen Farzana: Data Collection & Cleaning

V. Anitha: Exploratory Data Analysis

D. Keerthana: Feature Engineering & Model Building

R. Ananthi: Visualization & Deployment

K. Divya: Documentation & Reporting