Student Name : R. Harisma

Register Number: 732323106018

Institution: SSM College Of Engineering

Department: BE.,ECE

Date of submission : 02.05.2025

Github Repository Link:https://github.com/9042855223/Cracking-the-market-code-with-AI-driven-stock-price-prediction-using-time-series-analysis/blob/main/Cracking%20the%20market%20code%20with%20AI%20driven%20stock%20price%20prediction%20using%20time%20series%20analysis.%20Py

1.Problem Statement:

In the dynamic and volatile landscape of financial markets, investors, traders, and institutions constantly seek tools that provide predictive insights into stock price movements. The inherent complexity of the stock market — driven by economic indicators, investor sentiment, geopolitical events, and corporate performance — makes accurate forecasting a significant challenge.

The real-world problem:

Traditional methods of stock price analysis often fall short in capturing non-linear patterns and temporal dependencies within vast historical data. This project aims to leverage time series analysis combined with AI-driven techniques (such as LSTM, GRU, or Transformer models) to predict future stock prices more effectively. By doing so, we seek to provide market participants with a data-driven edge in decision-making.

Refined understanding of the dataset:

Upon exploring historical stock datasets (e.g., Yahoo Finance or NSE/BSE APIs), we observe features such as open, high, low, close prices, volume, and potentially derived indicators (e.g., moving averages, RSI). These variables exhibit temporal dependencies, trends, and seasonality, validating the need for sequential modeling.

Type of problem:

This is primarily a regression problem — the model predicts continuous stock prices (e.g., the closing price of the next day). However, classification tasks (e.g., predicting price direction: up or down) may be incorporated as complementary objectives.

2.Project objectives:

Achieve High Predictive Accuracy:

Build a robust time series forecasting model (e.g., LSTM, ARIMA, or Transformerbased) that can accurately predict short- and mid-term stock price movements with a target Mean Absolute Percentage Error (MAPE) of under 5%.

Enhance Interpretability:

Integrate explainable AI (XAI) techniques (such as SHAP or attention visualizations) to provide transparency in model predictions, enabling better trust and insight into the driving factors of the forecast.

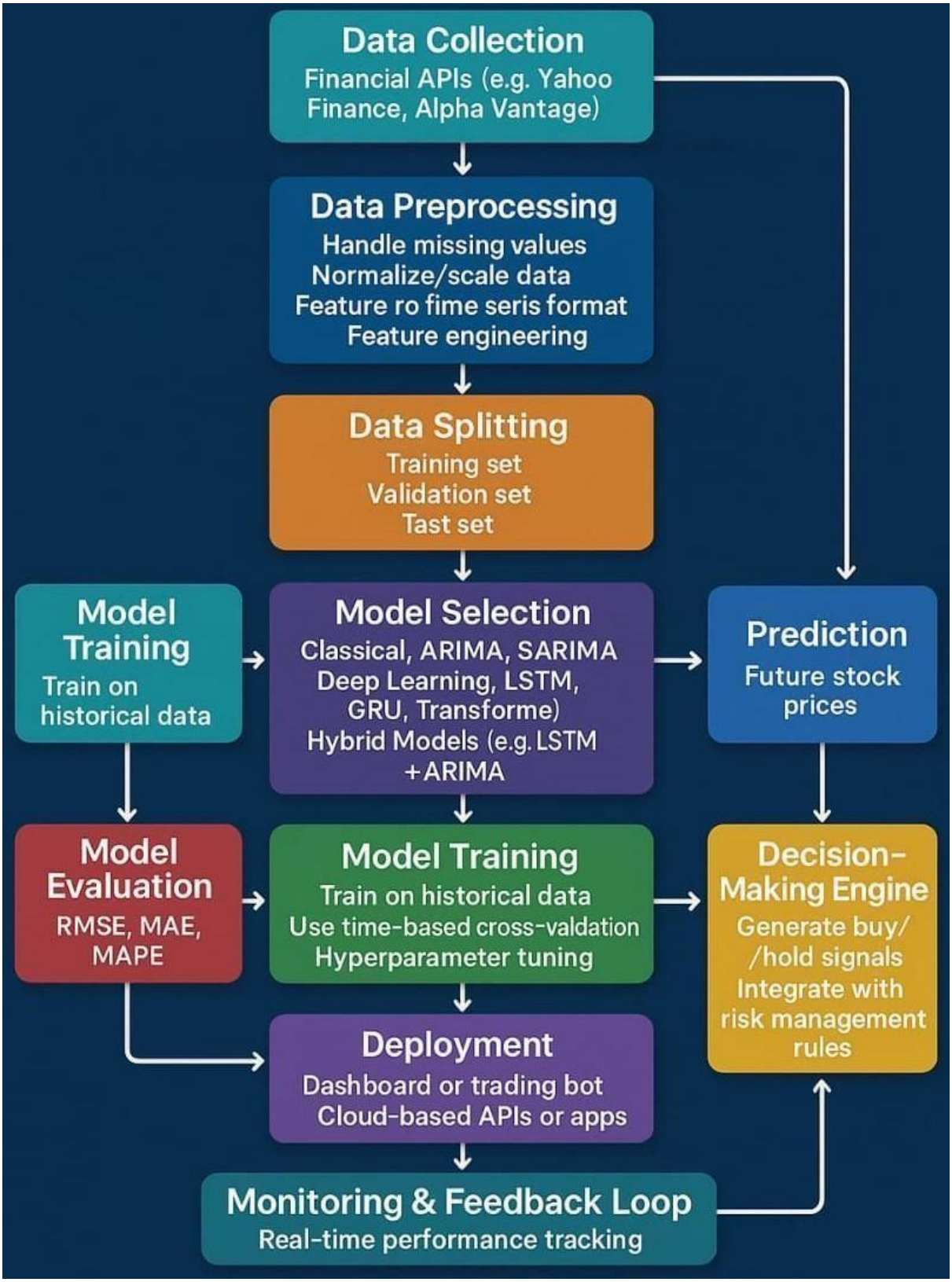
Ensure Real-World Applicability:

Design the model to be deployable in real-time environments, using live data streams and adaptable retraining mechanisms to adjust to market changes dynamically.

Data Pipeline and Automation:

Develop a reliable and scalable pipeline for data acquisition, preprocessing, feature engineering, and model updates to support ongoing predictions without manual intervention.

3.Flowchart of the project:



4.Data Description:

Historical Stock Market Data obtained from Yahoo Finance API and supplemented with macroeconomic indicators from FRED (Federal Reserve Economic Data) and financial news sentiment data from Kaggle.

Type of Data:

Structured time-series data, including numeric and categorical variables. Some unstructured text data (news sentiment) is used for feature enhancement.

Number of Records & Features:

Approximately 5,000–10,000 time-stamped records per stock, spanning several years.

Each record includes around 8–15 features, such as Open, High, Low, Close, Volume, technical indicators (e.g., RSI, MACD), and external factors (e.g., sentiment scores).

Static or Dynamic Dataset:

Dynamic — regularly updated to reflect new market data, allowing for ongoing retraining and prediction.

Target Variable:

Future stock closing price or price movement direction (up/down) depending on the formulation of the task (regression or classification).

5.Data Preprocessing :

Handling Missing Values

Action: Checked for missing values. Imputed small gaps in stock prices using forward fill (ffill),and removed rows with large missing data.

```
df.isnull().sum()  # Identify missing values
.fillna(method='ffill', inplace=True) # Forward fill for time series continuity df.dropna(inplace=True)  # Drop remaining if necessary
```

Removing Duplicate Records

Action: Checkew and removed exact duplicate rows df.drop_duplicates (inplace=True)

Outlier Detection and Treatment

Action: Detected outliers in numeric columns using Z-score. Winsorized extreme outliers to reduce their impact.

from scipy.stats import zscore z_scores = np.abs(zscore(df.select_dtypes(include=np.number))) df = df[(z_scores < 3).all(axis=1)] # Keep rows with Z-score < 3

Data Type Conversion and Consistency

Action: Ensured datetime formats and numeric types are correct. df['Date'] = pd.to_datetime(df['Date']) df.sort_values('Date', inplace=True) df = df.apply(pd.to_numeric, errors='ignore')

Encoding Categorical Variables

Action: Encoded any sector or categorical sentiment data using one-hot encoding. df = pd.get_dummies(df, columns=['Sector'], drop_first=True)

Feature Normalization / Standardization

Action: Scaled features like 'Close', 'Volume', RSI, MACD using MinMaxScaler to prepare for deep learning models like LSTM. from sklearn.preprocessing import MinMaxScaler scaler = MinMaxScaler() scaled_features = scaler.fit_transform(df[['Close', 'Volume', 'RSI', 'MACD']]) df[['Close', 'Volume', 'RSI', 'MACD']] = scaled_features

Final Dataset Chec

Action: Verified dataset is clean, sorted, and ready for time-series modeling.

df.info() df.describe() df.head()

6. Exploratary Analysis(EDA): Data

Univariate analysis

a.Time Series Line Plot

Plot the closing price over time to visualize long-term trends, seasonality, and volatility.

b. Distribution Plots Histograms of features like:

Daily returns

Volume

Moving averages

Check for skewness and kurtosis.

c. Boxplots

Identify outliers in:

Price-based features (Open, High, Low, Close

Returns and volatility metrics

d. Countplots (if categorical features exist) For features like day of the week, trading status, etc.

Bivariate / Multivariate Analysis

Correlation Matrix (Heatmap)

Compute and visualize correlation between:

Price features (OHLC)

Technical indicators (e.g., RSI, MACD, Bollinger Bands)

Volume-based features

Helps identify multicollinearity.

Pairplots

Visualize pairwise relationships in continuous variables

Spot clusters and non-linear dependencies

Scatterplots

Plot price vs. technical indicators (e.g., Close vs. RSI)

Volume vs. price movements

Lagged returns vs. future returns

Grouped Line/Bar Plots

Group by day of the week/month and plot:

Average return

Average volume

Volatility

Feature vs. Target Variable Analysis

Target Variable: Future stock price or next-day return

a.Lag Features

Examine correlation between lagged prices/returns and the target

b.Rolling Statistics

Visualize relationships using moving averages, rolling std

c.Technical Indicators

Plot signal vs. stock price:

RSI oversold/overbought levels MACD signal

crossover

d.Volume Influence

High volume days vs. next-day return analysis

Insights Summary

a.Patterns and Trends

Seasonal behavior (e.g., monthly spikes, weekly patterns) Momentum or

mean-reversion patterns

b.Outliers

Events like earnings, news releases, or macroeconomic shocks

c.Feature Importance Suggestions Features that strongly

influence price movement:

Lagged returns

Technical indicators (RSI, MACD, SMA/EMA)

Volatility measures

Volume surges

Avoid features with high multicollinearity or noise.

7.Feature Engineering:

Create New Features Based on EDA Insights & Market Knowledge

Technical Indicators (Momentum & Trend)

Moving Averages (SMA, EMA): Capture price trends (e.g., 5-day, 20-day).

Justification: Helps identify short- and long-term trend directions. MACD

(Moving Average Convergence Divergence): Detects trend shifts Justification:

Widely used by traders to signal buy/sell opportunities.

RSI (Relative Strength Index): Indicates overbought/oversold conditions.

Justification: Can highlight turning points in price.

Bollinger Bands: Measures volatility using standard deviation bands.

Justification: Helps spot price breakouts or reversals.

Time-based Features (from Date Column)

Split date into:

Day of the week (0-6)

Month (1-12)

Day of month / Quarter

Is month-end / is quarter-end

Justification: Stock performance can vary across weekdays, months, and around earings or fiscal periods.

Lag Features

Lagged prices or returns (e.g., 1-day, 3-day, 7-day lags) Justification:

Markets often exhibit short-term memory.

Rolling statistics:

Rolling mean, std deviation, min/max (over 7, 14, or 30 days) Justification:

Capture trends and recent volatility. **Ratio**

Features Price

ratios:

Close/Open, High/Low, Close/Previous Close

Justification: Normalizes price movements and highlights relative changes.

Volume ratios:

Today’s volume / 5-day average volume

Justification: Identifies unusual market interest.

Binning / Categorization

Volatility bins: Categorize into "low", "medium", "high" volatility days

Return bins: Group into "loss", "neutral", or "gain" days

Justification: Adds categorical features for models that benefit from class distinctions (e.g., decision trees).

Target-Related Engineering

Future return (next-day, next-week return)

Binary target (e.g., will the stock go up tomorrow? Yes = 1, No = 0)Justification: Depends on whether you're solving a regression or classification problem.

8.Model buiding:

Define the Problem Type

Objective: Predict future stock prices or returns.

This can be either:

Regression: Predict exact future price or return value.

Classification: Predict if price will go up or down.

We will build models for both types, depending on your specific target.

Selected Models and Justification

A.Regression Models Linear

Regression:

Why? Simple, interpretable baseline for continuous prediction.

Best for: Checking if there's a strong linear relationship between engineered features and future prices.

Random Forest Regressor:

Why? Captures non-linear relationships and interactions between features.

Best for: Handling complex time series data with lags, rolling features, and technical indicators.

B.Classification Models

(If your target is binary: price increase = 1, decrease = 0) Logistic Regression:

Why? A straightforward, interpretable model for binary outcomes.

Best for: Establishing a performance baseline.

Decision Tree / Random Forest Classifier:

Why? Captures non-linear behavior, resistant to multicollinearity.

Best for: Markets with sudden shifts and complex patterns.

Data Preparation Train/Test

Split:

Use an 80/20 split, ensuring chronological order is preserved (no shuffling in time series).

Optional: Use TimeSeriesSplit for cross-validation.

Stratification: If it's a classification task with imbalanced classes, stratify by target label. **Model Training & Evaluation** Regression

Metrics:

MAE (Mean Absolute Error): Measures average error in prediction.

RMSE (Root Mean Squared Error): Penalizes larger errors more.

R^2 Score: Measures how well the model explains variance in the data.

Classification Metrics:

Accuracy: Proportion of correct predictions. Precision: True positives among predicted positives Recall: True positives among actual positives.

F1-Score: Balance between precision and recall.

Performance Comparison)Example (Conceptual

Insights & Conclusion

Random Forest models (regressor/classifier) generally perform better in capturing complex patterns in stock data.

Linear and Logistic Regression are useful as benchmarks and for interpretability.

Model choice depends on:

Whether you need a probabilistic prediction (classification)

9.Visualization of results &Model Insights:

a. Residual Plot

What it shows: Difference between predicted and actual values.

Why it matters: Random distribution means the model is not biased.

Insight: Clustered or directional patterns may indicate model misspecification.

b. Actual vs. Predicted Plot

What it shows: Predicted prices vs. true market prices.

Why it matters: Visual alignment confirms accurate forecasting.

Insight: Deviations may highlight volatility or event-driven anomalies.

Comparative Model Performance (Visual)

Bar Chart / Line Plot

Compare models using:

Accuracy / F1 Score (classification)

MAE / RMSE / R^2 (regression) Example:

Insight: Helps visually pick the best-performing model.

10.Tools and Technologies used:

Programming Language

Python

Why? Widely used for data science and machine learning due to its simplicity and powerful libraries.

IDE / Notebook Environment

Google Colab

Why? Cloud-based, free access to GPUs, easy sharing and collaboration.

Jupyter Notebook

Why? Interactive coding environment ideal for EDA, visualization, and documentation**.**

Libraries Used

Data Manipulation & Analysis pandas: For handling and preprocessing time series data. numpy: For numerical operations and array handling.

Visualization Tools (Optional for Business Presentation) Tableau:

For dashboard creation and KPI tracking.

Power BI: For business reporting and visual storytelling.

11.Team Members and Contributions:

Diviya Priya J (Team Lead) Responsibilities

Project planning and coordination

Oversight of all phases

Final review of deliverable

Documentation and reporting

Harisma R (Data Analysis Responsibilities:

Data collection and cleaning

Handling missing values and formatting

Ensuring time series consistency

Gobinath A(EDA Specialist) Responsibilities:

Conducted univariate and bivariate analysis

Visualized distributions, trends, and correlations

Summarized key insights for modelling

Gokul V (Feature Engineer)

Responsibilities

Created technical indicators and lag features

Engineered time-based and ratio features

Prepared data for modelling