Intelligent System
Assignment 1 - Option B
Project Summary Report

Xuan Tuan Minh Nguyen - 103819212

October 27, 2024

Contents

1 Introduction

This project summarizes the application of machine learning techniques for predicting stock prices, with a focus on incorporating multiple techniques to enhance prediction accuracy.

The primary objectives of this project are:

- To develop a reliable stock price prediction system using different ML models
- To execute and evaluate different machine learning techniques and models for time series prediction
- To summarizes the system's implementation and suggest potential improvements
- To extend the system by implementing macroeconomic indicators into the prediction process

This report will detail the system architecture, data processing techniques, machine learning models employed, demonstration scenarios, and provide an in-depth analysis of the implementation.

2 System Architecture

The stock price prediction model will consist of several key procedures:

2.1 Data Collection

- Stock price data: Using data from Yahoo Finance using the **yfinance** library
- Macroeconomic indicators: In addition, to obtain the macroeconomic indicators, FRED will be used as the data source.

2.2 Data Processing

- Data cleaning and normalization using **pandas** and **numpy**
- Feature engineering, including technical indicators using **TA-Lib**
- Time series alignment of stock and macroeconomic data

2.3 Machine Learning Models

- LSTM (Long Short-Term Memory) neural network adopted from **TensorFlow/Keras**
- ARIMA (AutoRegressive Integrated Moving Average) adopted from **pmdarima**
- Ensemble model combining LSTM and ARIMA predictions using **sklearn**

2.4 Prediction and Evaluation

- Model training and prediction
- Evaluating the model using RMSE (Root Mean Square Error)
- Visualization of predictions vs actual stock prices using **matplotlib** and **seaborn**

3 Implemented Data Processing Techniques

3.1 Data Collection and Preprocessing

The project uses the **yfinance** library to retrieve historical stock data for Tesla (TSLA) and, in addition, the **fredapi** library to retrieve macroeconomic indicators. The collected data undergoes several preprocessing steps:

- Clean missing values: Any missing data points are either filled or removed to ensure data consistency.
- Timestamps alignment: Stock market data and macroeconomic indicators are aligned to ensure consistency.
- Data normalization: Advanced techniques are applied to bring all features to a machine-friendly scale, resulting in better understanding for many machine learning algorithms, such as **MinMaxScaler**, etc.

3.2 Feature Engineering

Technical indicators are in charged to provide additional features for the models, which helps capturing aspects of market trends and stock price movements:

- Moving Averages (MA): Using a 5-day and 20-day period moving averages to smooth out fluctuations and highlight longer-term trends.
- Relative Strength Index (RSI): This indicator helps measuring the speed and change of price movements, helping to identify any overbought or oversold conditions.
- Moving Average Convergence Divergence (MACD): This trend-following momentum indicator shows the relationship between two moving averages of a security's price.

3.3 Time Series Preparation

The data is structured into sequences suitable for time series prediction:

- Sliding windows: Historical data is organized into sequences that overlap to capture temporal dependencies.
- Data splitting: The dataset is divided into 80 percents for training and 20 percents for testing sets.
- Feature-target alignment: Ensuring that input features and target variables are properly aligned for each time step in the sequence.

4 Experimented Machine Learning Techniques

4.1 LSTM Neural Network

A Long Short-Term Memory (LSTM) neural network is implemented from **TensorFlow/Keras**. This model is well-suited for problems that requires sequence prediction scenarios, such as stock price forecasting, thanks to the ability of capturing long-term dependencies.

4.1.1 Architecture

- Input layer: Accepts the feature sequences.
- LSTM or GRU and RNN layers: Multiple LSTM (or GRU and RNN) layers are added to capture patterns.
- Dense layers: For final predictions.
- Dropout layers: To stop the model from being overfitted.

4.1.2 Hyperparameters

- LSTM units: Around 32 to 128 units.
- Dropout rate: Usually between 0.2 to 0.5
- Learning rate: Starting at 0.001 and re-scheduled using learning rate schedules.
- Batch size: Between 32 and 128 batches.
- Number of epochs: Determined by early stopping algorithm to prevent overfitting.

4.1.3 Training

- Optimizer: Used Adam optimizer due to its efficiency in training deep networks.
- Loss function: Mean Squared Error (MSE) for regression tasks.
- Validation: Using training data for validation during training
- Early stopping: To halt training when the model has no improvements.

4.2 ARIMA Model

An AutoRegressive Integrated Moving Average (ARIMA) model is implemented using the **pmdarima** library. ARIMA is well-known as a classical statistical method for time series forecasting that uses autoregression, differencing, and moving average techniques.

4.2.1 Model Selection

- Automatic parameterized: Adopting the **auto_arima** function to find the required **p, d, q** values.
- Seasonal components: If seasonal patterns are detected then uses ARIMA models.
- Information Criteria: BIC (Bayesian Information Criterion) is used to select the best model.

4.2.2 Parameter Tuning

- Grid search: Exploring a range of **p, d, q** values (typically 0-5 for each)
- Differencing: Use the appropriate level to achieve stationarity.
- Residual analysis: Checking residuals to validate model assumptions.

4.3 Ensemble Model

An ensemble approach is used for combining predictions from LSTM and ARIMA models, empowers both neural network and statistical approaches.

4.3.1 Combination Methods

- Simple averaging: Calculating the mean of LSTM and ARIMA predictions
- Weighted averaging: Using weights based on individual model performance (RMSE)
- Stacking: Using a modern model, such as **RandomForest**, for learning learn optimal combination weights

4.3.2 Ensemble Optimization

- Cross-validation: Using time series to determine optimal ensemble weights
- Dynamic weighting: Determine the weight based on recent performance of individual models
- Diversity analysis: Making sure that both LSTM and ARIMA models capture different aspects of the data

5 Scenarios/Examples to Demonstrate System Functionality

5.1 Scenario 1: Single Stock Prediction

This scenario illustrates the system's ability to predict future stock prices for Tesla (TSLA) based on historical data with the help of macroeconomic indicators.

5.1.1 Input

- Historical stock data with **Open, High, Low, Close, Volume** from 2015-01-01 to 2023-08-25
- Macroeconomic indicators **(GDP growth rate, inflation rate, unemployment rate)**
- Technical indicators calculated from the stock data **(RSI, MACD)**

5.1.2 Process

- Data preprocessing: Cleaning, normalizing, and feature engineering
- Model training: Using LSTM and ARIMA to train based on the processed data
- Prediction: Generate predictions for the next period of days

5.1.3 Output

- Predicted stock prices for TSLA for the next period of days
- Performance metrics **(RMSE, MAE)** for the prediction period

5.1.4 Visualization

- Line plot for displaying the actual historical prices and predicted future prices
- Shaded area representing the uncertainty of predictions
- Overlay of key macroeconomic events or indicators

5.2 Scenario 2: Model Comparison

This scenario compares the performance of different models to highlight their strengths and weaknesses in predicting TSLA stock prices.

5.2.1 Input

- Processed data for TSLA stock, including all features and indicators
- Holdout test set for final evaluation

5.2.2 Process

- Train LSTM, ARIMA, and Ensemble models using the same training data
- Generate predictions for the test period using each model
- Calculate performance metrics (RMSE, MAE, R-squared) for each model

5.2.3 Output

- RMSE scores for LSTM, ARIMA, and Ensemble models
- MAE and R-squared values for each model
- Prediction series from each model for the test period

5.2.4 Visualization

- Bar chart comparing RMSE, MAE, and R-squared across models
- Line plot showing predictions from each model against actual prices
- Residual plots for each model to visualize prediction errors

5.3 Scenario 3: Macroeconomic Impact Analysis

This scenario demonstrates the impact of incorporating macroeconomic indicators on prediction accuracy.

5.3.1 Input

- Dataset 1: Stock data with technical indicators only
- Dataset 2: Stock data with technical indicators and macroeconomic features

5.3.2 Process

- Train models on both datasets separately
- Generate predictions using both sets of models
- Compare prediction accuracy between models with and without macroeconomic data

5.3.3 Output

- Prediction accuracy metrics for models with and without macroeconomic data
- Importance scores for macroeconomic features (if applicable)
- Analysis of which macroeconomic indicators have the most significant impact

5.3.4 Visualization

- Line chart showing prediction improvements with macroeconomic data
- Heatmap of feature importance, highlighting key macroeconomic indicators
- Scatter plot of prediction errors vs. specific macroeconomic variables

6 Critical Analysis of the Implementation

6.1 Strengths

- Macroeconomic indicators integration: The integration of global economic context enhances the model's ability to capture beyond stock-specific factors.
- Ensemble approach: Combining LSTM and ARIMA models increases the strengths of both neural network and statistical approaches, leading to more efficient predictions.
- Comprehensive data preprocessing: The data cleaning, normalizing, and feature engineering processes ensures high-quality input for the models.
- Evaluation metrics: RMSE, MAE, and R-squared provides a well evaluation of model performance.
- Scenario-based demonstration: Empowers the specific scenarios helps summarize the system's functionality and potential real-world applications.

6.2 Limitations

- Linear ensemble combination: The current ensemble method uses simple or weighted averaging, which may not capture complex interactions between model predictions.
- Potential overfitting in LSTM: Deep learning models like LSTM can be prone to overfitting, especially with limited data or complex architectures.
- Assumption of consistent relationships: The model assumes that the relationships between indicators and stock prices remain consistent over time, which may not always hold true in dynamic markets.
- Limited to single stock prediction: The current implementation focuses on Tesla stock, which may not generalize well to other stocks or market sectors.
- Lack of real-time adaptation: The models are trained on historical data and may not quickly adapt to sudden market changes or new economic conditions.

6.3 Potential Improvements

- Modern ensemble techniques: Implement more sophisticated ensemble methods such as stacking or boosting to better combine individual model strengths.
- Using alternative architectures: Explore other deep learning models like Transformer networks or Temporal Convolutional Networks (TCN) for time series prediction.
- Real-time data integration: Develop a system for continuous data fetching and model updating to adapt to changing market conditions.
- Incorporate sentiment analysis: Integrate natural language processing techniques to analyze news articles, social media, and financial reports for additional predictive signals.
- Expand to multi-stock prediction: Extend the system to handle multiple stocks simultaneously, potentially capturing inter-stock relationships and sector-wide trends.
- Adaptive learning rate and regularization: Implement techniques like cyclical learning rates and adaptive regularization to improve model training and generalization.
- Optimize hyperparameters using Bayesian optimization: Use Bayesian optimization techniques for more efficient hyperparameter tuning across all models.

7 Summary/Conclusion

This project demonstrates the combination of traditional time series analysis (ARIMA) with modern deep learning techniques (LSTM) for stock price prediction. The use of macroeconomic indicators provides a more comprehensive approach to capturing outside factors.

In conclusion, while the developed system shows promise in predicting stock prices, it's critical to note that financial markets are influenced by multiple factors, which could be unpredictable. This system should be used as one of many inputs in making investment decisions, rather than as a sole determinant. The project demonstrates the power of machine learning in financial analysis but also points out the complexity and challenges involved in stock price prediction.

The use of macroeconomic indicators and ensemble methods represent a revolutionary step forward in creating more efficient prediction models. Hence, adaptation to market conditions, and careful interpretation of results are essential for practical application in real-world stock trading.