Stacked LSTM for Stock Price Prediction

Team 3: Yitian Wang, Zhixuan Li, Jason Liao, Sihan Zhou, Mia Kwon

Introduction

Problem Statement

This project focuses on predicting stock prices, a quintessential problem in financial markets. Accurate stock price prediction is crucial for investors and financial analysts, as it guides investment strategies, risk management, and portfolio optimization.

Data Overview

The dataset used consists of historical stock prices of Goldman Sachs (GS) from 2011 to 2023. Key features include Open, High, Low, and Close prices, and Volume of stocks traded. The chronological nature of stock market data, coupled with its inherent volatility and trend patterns, makes it a compelling case for applying deep learning models.

Relevance and Business Impact

The implementation of a Stacked LSTM model for stock price prediction holds significant potential to revolutionize various aspects of the financial sector. By accurately forecasting market trends, it enables the execution of more informed and profitable investment strategies, contributing to enhanced financial stability and investor confidence. Its applications extend to algorithmic trading, where rapid, data-driven trades can capitalize on market opportunities, and to risk management, assisting institutions in mitigating potential losses. Additionally, the model

can be pivotal in portfolio optimization, providing insights for asset diversification to balance returns and risks. Its utility also stretches to financial advisory services, where it can augment the quality of advice provided to clients, and in regulatory compliance, aiding in the detection of market anomalies. The business impact can be summarized by the following points

Improved Decision Making: The predictive model will provide a more data-driven approach to stock market investment, reducing reliance on intuition and speculative methods.

Increased Return on Investments (ROI): By accurately predicting market trends, the model can help users enter and exit positions at more optimal times, potentially increasing their ROI.

Risk Mitigation: Early identification of negative trends allows for quicker response, thereby reducing potential losses. This is particularly valuable for risk-averse investors and those managing large portfolios.

Competitive Advantage: Investment firms and hedge funds using the model could gain an edge over competitors not utilizing similar advanced predictive tools.

However, the application of such advanced predictive models is not without challenges. Stock markets are inherently volatile and influenced by a multitude of unpredictable factors, including political events, economic shifts, and global incidents, which may limit the model's accuracy. The risk of overfitting historical data poses another significant concern, potentially reducing the model's efficacy in predicting future market movements. Furthermore, an overreliance on data-driven models may lead to the undervaluing of qualitative factors like management quality or industry trends.

Data Preparation

Data Preprocessing

The raw data was subjected to several preprocessing steps to make it suitable for feeding into a deep learning model:

Normalization

In this project, normalization was performed using the MinMaxScaler, which scales each feature to a range between 0 and 1. This step is crucial for multiple reasons:

Uniformity of Features: Stock price data typically involves various metrics like Open, High, Low, Close, and Volume, each with different scales and ranges. Normalization brings these diverse features onto a common scale, eliminating the dominance of one feature over others due to its scale.

Improved Model Training: Neural networks, including LSTMs, generally perform better and converge faster when the input data is normalized. This is because large input values (stock prices, in this case) can lead to larger error gradients, causing the learning process to oscillate wildly. Normalization helps in moderating the gradients, leading to a smoother and faster convergence.

Consistency Across Time: The stock market is dynamic, with prices changing drastically over time. Normalization helps in maintaining consistency in the input data over different time periods, making the model more robust to fluctuations in the scale of the data.

Sequence Formatting

The raw time series data was transformed into a format suitable for supervised learning. A key aspect of this transformation is the creation of sequences using a defined 'lookback' period:

Lookback Period: Each input sequence comprises data from a set number of previous time steps (the lookback period). This period is crucial as it determines how much historical data the model will consider when making a prediction. In our case, a lookback period of 30 days means the model will use data from the past 30 days to predict the next day's stock price.

Input-Target Pairing: For each sequence, the final day’s stock price is set as the target variable, and the preceding days' data as the input features. This structure is fundamental for the LSTM to learn the temporal dependencies within the data.

Data Integration

The data was divided into training and testing sets. The training set was used to train the model, and the testing set was used to evaluate its performance.

Modeling

Model Choice: Stacked LSTM over Standard LSTM

Learning Hierarchical Representations: Financial markets are influenced by a multitude of factors, including macroeconomic indicators, company news, and market sentiment. A Stacked

LSTM can capture these complex relationships at different levels of abstraction. Each layer in the stack can be thought of as learning different features of the data, with higher layers learning more abstract representations.

Capturing Long-Term Dependencies: Stock prices are not only affected by recent trends but also by events that occurred in the more distant past. Standard LSTMs may struggle to remember information for long periods, but Stacked LSTMs, with their increased depth, can potentially capture long-term dependencies better, which is crucial for stock price prediction.

Robustness to Noise: Financial data can be noisy and volatile. Stacked LSTMs, with their multiple layers, can be more robust to noise, learning to ignore irrelevant fluctuations and focusing on underlying trends in the data.

Model Architecture

The model architecture includes:

- Multiple LSTM layers, enabling the model to learn higher-level temporal representations.
- A fully connected layer that maps the output of the LSTM layers to the desired output shape.

Hyperparameter Tuning

Key hyperparameters tuned include:

- **Hidden Dimensions**: Size of the LSTM hidden layers.
- **Number of Layers**: Depth of the stacked LSTM.
- **Learning Rate**: Step size at each iteration of model weight updates.
- **Dropout Rate**: Fraction of neurons dropped out to prevent overfitting.

- **Weight Decay**: L2 regularization parameter.

Alternative models like single-layer LSTM, ARIMA, and Prophet were considered. Each model has its strengths, but stacked LSTMs were chosen for their ability to model complex patterns in time series data.

Implementation

Challenges and Solutions

- **Model Complexity**: Managing a multi-layered LSTM architecture was challenging due to the increased complexity. Systematic coding and thorough testing helped in mitigating this.
- **GPU Utilization**: Leveraging CUDA for GPU-accelerated training posed challenges in data and model management, which were addressed by ensuring proper tensor and model device assignments.

Code Implementation

The project's implementation, documented in a Jupyter notebook, involves:

- Data loading and preprocessing.
- Defining the LSTM model structure.
- Training the model on the processed data.
- Hyperparameter tuning using Hyperopt.

Results and Evaluation

Performance Metrics

- **RMSE (Root Mean Square Error)**: Used to measure the differences between predicted and actual stock prices.
- **Visual Analysis**: Plots comparing the model's predictions against actual stock prices provided visual insights into its accuracy.

Model Performance

The Stacked LSTM model achieved a training RMSE of 5.00 and a testing RMSE of 10.90, outperforming the XGBoost benchmark, which had an RMSE of 18.07.

Additional testing also show that the model has a superior result in a more volatile time. In this additional evaluation, we use Morgan Stanley's historical prices in this additional result since MS has a higher volatility compared to GS in 08' recession. The model improved in a more volatile time. It has a RMSE of 1.52 for training and 2.36 RMSE. Compared to the benchmark XGBoost model that has 10.21 RMSE, it produces a significant improvement. XGBoost seems unable to predict a value that it has not seen. The stacked LSTM does not have this limitation

Benchmarking

The model's performance was benchmarked against an XGBoost model, highlighting LSTM's superior capability in handling temporal dependencies. XGBoost, short for Extreme Gradient Boosting, is a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework. It's widely recognized for its performance in predictive modeling competitions and its ability to handle a variety of data types, distributions, and the presence of missing values. When it comes to stock price prediction, XGBoost can capture nonlinear relationships and interactions between features, which are common in financial datasets. Benchmarking against XGBoost allows us to demonstrate the Stacked LSTM's ability to

outperform a strong baseline that doesn't specialize in time-series data. The comparison can solidify the argument for using more complex, sequential models like LSTMs in scenarios where temporal relationships are key. If the Stacked LSTM shows significantly better performance than XGBoost, it would validate the choice of using a more complex and computationally intensive model for the task at hand.

By setting XGBoost as a benchmark, we establish a performance baseline grounded in a different class of machine learning algorithms, thus providing a broader perspective on the model's capabilities and helping to underline the specific conditions under which deep learning models are more suitable for stock price prediction tasks.

Business Application

The Stacked LSTM model developed for stock price prediction can serve as a cornerstone for multiple business applications in the financial sector, capitalizing on its ability to understand and forecast market dynamics effectively.

AI-Driven Trading Systems

The model's predictive capabilities can be integrated into AI-driven trading systems that operate on algorithmic strategies. These systems can execute trades based on the predictive signals generated by the model, optimizing for maximum returns or other specific financial goals. By automating trade decisions based on learned patterns in historical data, these systems aim to outperform manual trading strategies, both in terms of speed and consistency.

Investment Strategy Formulation

Investment firms and hedge funds can leverage the model's insights to craft sophisticated investment strategies. By predicting future price movements, analysts can identify potential buy/sell signals, adjust their asset allocations, and manage entry and exit points in their investment portfolios. This can lead to enhanced portfolio yields and better alignment with risk appetite and investment horizons.

Risk Management

The ability to forecast stock prices is also crucial for risk management. Financial institutions can use model predictions to anticipate market downturns and take preemptive actions to hedge their portfolios against potential losses. This proactive approach to risk management is essential for maintaining financial stability and can help firms navigate through volatile market conditions.

Deployment

Deployment Strategy

The deployment of the Stacked LSTM model can be executed within a cloud-based framework or on local servers, depending on the specific needs and infrastructure of the organization. Utilizing a cloud-based environment offers scalability and accessibility, allowing financial analysts and traders to tap into the model's capabilities from anywhere, at any time, and with the potential to handle vast amounts of data that financial markets generate daily. On the other hand, an on-premise deployment can offer enhanced security and control, which can be critical for sensitive financial data. Integration into existing trading platforms would enable real-time analysis and decision-making, allowing for instantaneous market actions based on the model's predictions. This could be achieved through APIs that feed the model's insights directly into the

trading system, thereby streamlining operations and facilitating a swift response to dynamic market conditions. Whether cloud-based or on-premise, the key to successful deployment lies in ensuring the model's seamless integration, maintaining data integrity, and meeting the rigorous performance and security standards required in financial applications.

Ethical and Risk Considerations

- **Ethical Transparency**: Ensuring clarity on how predictions are made is critical.
- **Market Risk**: The inherent unpredictability of stock markets should be acknowledged.
- **Model Misuse**: Potential misuse for market manipulation needs to be safeguarded against.

Risk Mitigation

Risks can be mitigated through robust testing, clear communication of model limitations, and adherence to ethical guidelines.

Team Member Contribution:

Tasks were equally divided among all team members