LSTM-Based Trading Strategy Analysis Report

Application for Trexquant

Sparsh Gupta

June 2025

Abstract

This report presents a comprehensive analysis of a Long Short-Term Memory (LSTM) neural network-based trading strategy applied to Pfizer Inc. (PFE) stock from 2010 to 2025. The strategy employs technical indicators and machine learning techniques to generate buy/sell signals for systematic trading decisions.conditions.

Contents

1 Data Extraction

1.1 Data Source and Collection Process

The strategy begins with systematic data extraction from Yahoo Finance using the yfinance Python library. This process involves:

- **Source:** Yahoo Finance API through yfinance library
- **Security Symbol:** PFE (Pfizer Inc.)
- **Date Range:** January 1, 2010 to April 17, 2025
- **Data Fields:** Complete OHLCV dataset (Open, High, Low, Close, Volume, Adjusted Close)

2 Data Cleaning

2.1 Data Quality Assurance Process

Systematic data cleaning ensures model reliability through multiple validation steps:

2.1.1 Missing Value Treatment

- **Method:** Complete case analysis using dropna() function
- **Reason:** Removes any trading days with incomplete price or volume data

2.1.2 Feature Selection for Analysis

The cleaning process deliberately focuses on two primary variables:

- **Close Prices:** End-of-day settlement prices for trend analysis
- **Volume:** Trading activity levels for market participation insights

3 Handling of Non-Numeric Features

3.1 Data Type Management

While this particular strategy focuses on numeric time series data, the implementation demonstrates proper handling of different data types:

3.1.1 Date Index Processing

- **Original Format:** String-based date information from Yahoo Finance
- **Conversion Process:** Automatic conversion to pandas DateTime index
- **Temporal Functionality:** Enables time-based filtering and chronological operations

3.1.2 Numeric Data Validation

- **Price Data:** Verified as float64 format for mathematical operations
- **Volume Data:** Confirmed as integer/float format for technical calculations
- **Missing Value Encoding:** Proper handling of NaN values before model input

3.1.3 Binary Target Creation

The strategy creates a binary classification target by converting continuous price movements into discrete up/down signals, demonstrating categorical variable creation from numeric inputs.

4 Feature Engineering

4.1 Technical Indicator Development Process

The feature engineering process transforms raw price and volume data into meaningful predictive variables through systematic technical analysis:

4.1.1 Momentum-Based Features

- **Daily Returns (Return_1d):** Calculates percentage change between consecutive closing prices using pct_change() method, capturing short-term price velocity
- **10-Day Momentum (Momentum_10):** Measures relative price performance by comparing current price to price 10 trading days ago, identifying medium-term trends

4.1.2 Moving Average Features

- **Short-Term Average (MA_10):** 10-day simple moving average using rolling window calculations, smoothing recent price action
- **Long-Term Average (MA_50):** 50-day simple moving average providing broader trend context
- **Moving Average Ratio (MA_ratio):** Normalized relationship between short and long-term averages, indicating trend strength and direction

4.1.3 Technical Oscillators

RSI Calculation Process:

1. Computes daily price changes and separates gains from losses
2. Applies 14-period exponential smoothing to average gains and losses
3. Calculates Relative Strength (RS) ratio and converts to 0-100 RSI scale
4. Identifies overbought (> 70) and oversold (< 30) market conditions

MACD Development:

1. Creates 12-period and 26-period exponential moving averages
2. Calculates difference (MACD line) to identify trend momentum changes
3. Provides early signals for potential trend reversals

4.1.4 Target Variable Creation

- **Binary Classification Target:** Converts continuous price movements into discrete signals by comparing next-day closing price to current closing price
- **Signal Logic:** Creates boolean values (True/False) then converts to integer format (1/0) for model compatibility

5 Modelling

5.1 Model Architecture Design Process

5.1.1 Sequential Model Construction

The LSTM model follows a systematic architecture designed for time series prediction:
Input Layer Configuration:

- **LSTM Layer:** 64 memory units configured for sequence processing
- **Input Shape:** Automatically determined from training data dimensions (sequence_length, features)
- **Return Sequences:** Set to False for final prediction output
- **Memory Capability:** Handles long-term dependencies in price patterns

Regularization Implementation:

- **Dropout Layer:** 20
- **Purpose:** Prevents overfitting by reducing model complexity
- **Placement:** Applied after LSTM layer before dense processing

Dense Layer Processing:

- **Hidden Layer:** 32 neurons with ReLU activation function
- **Function:** Non-linear transformation of LSTM output features
- **Activation Choice:** ReLU prevents vanishing gradient problems

Output Layer Design:

- **Single Neuron:** Binary classification output
- **Sigmoid Activation:** Converts output to probability between 0 and 1
- **Interpretation:** Values > 0.5 indicate upward price movement probability

5.2 Model Training and Optimization Process

5.2.1 Compilation Configuration

- **Loss Function:** Binary cross-entropy for classification tasks
- **Optimizer:** Adam algorithm with 0.001 learning rate for adaptive gradient descent
- **Metrics:** Accuracy tracking for model performance monitoring

5.2.2 Training Data Preparation

- **Sequence Creation:** Transforms time series into overlapping 20-day windows
- **Feature Standardization:** StandardScaler normalization ensures consistent input ranges
- **Target Alignment:** Ensures proper temporal alignment between features and prediction targets

5.2.3 Training Process Execution

- **Epoch Configuration:** 10 complete dataset iterations
- **Batch Processing:** 32-sample mini-batches for efficient gradient updates
- **Validation Monitoring:** Real-time performance tracking on separate validation dataset
- **Convergence:** Model learns patterns through iterative weight adjustments

6 Data Partitioning Strategy

6.1 Temporal Split Methodology

- **Training Period:** 2010-2020 (10 years)
- **Validation Period:** 2020-2022 (2 years)
- **Testing Period:** 2022-2025 (3 years)

This chronological partitioning ensures realistic backtesting conditions by preventing look-ahead bias and maintaining temporal sequence integrity.

6.2 Feature Standardization

All technical indicators undergo StandardScaler normalization to ensure consistent model input distributions and optimal LSTM performance.

7 Trading Signal Generation

7.1 Probability Threshold Framework

- **Long Signals:** Probability > 0.55 (Signal = +1)
- **Short Signals:** Probability < 0.45 (Signal = -1)
- **Neutral Zone:** $0.45 \leq$ Probability ≤ 0.55 (Signal = 0)

7.2 Risk Management

The symmetric threshold approach (± 0.05 from neutral) provides built-in risk management by avoiding low-confidence predictions and reducing transaction frequency.

8 Performance Metrics and Backtesting

8.1 Portfolio Construction

- **Initial Capital:** $1,000,000$**Position Sizing:**$Fullcapitalallocationpersignal$
- **Return Calculation:** Strategy returns = Daily returns $\times$ Position signals
- **Performance Tracking:** Cumulative profit/loss monitoring

8.2 Risk-Adjusted Performance

Sharpe Ratio Calculation:

$$\text{Sharpe Ratio} = \frac{\text{Mean(Strategy Returns)}}{\text{Standard Deviation(Strategy Returns)}} \tag{1}$$

This metric evaluates risk-adjusted performance by measuring excess returns per unit of volatility.

9 Technical Implementation Details

9.1 Sequence Generation Process

```
def create_sequences(data, features, target, seq_len=20):
# Creates overlapping 20-day windows for LSTM input
# Ensures temporal consistency in feature-target relationships
```

9.2 Model Training Validation

- **Validation Split:** Independent 2020-2022 dataset
- **Early Stopping:** Implicit through epoch limitation
- **Overfitting Prevention:** Dropout regularization and validation monitoring

10 Results Visualization and Analysis

10.1 Performance Visualization

The strategy generates a comprehensive performance chart displaying:

- Cumulative profit/loss trajectory over the testing period
- Visual identification of winning and losing periods
- Overall strategy performance trends

10.2 Statistical Summary

Key performance indicators include:

- **Total Profit/Loss:** Absolute dollar returns over testing period
- **Sharpe Ratio:** Risk-adjusted performance measurement
- **Strategy Consistency:** Cumulative return pattern analysis

11 Model Limitations and Considerations

11.1 Market Assumptions

- **Transaction Costs:** Not explicitly modeled
- **Market Impact:** Perfect liquidity assumption
- **Slippage:** Not incorporated in return calculations

11.2 Model Constraints

- **Feature Selection:** Limited to technical indicators only
- **Market Regime Changes:** No adaptive mechanism for changing market conditions
- **Overfitting Risk:** Relatively short validation period

12 Recommendations and Future Enhancements

12.1 Model Improvements

1. **Feature Expansion:** Incorporate fundamental analysis metrics
2. **Ensemble Methods:** Combine multiple ML algorithms
3. **Dynamic Thresholds:** Adaptive signal generation based on market volatility
4. **Transaction Cost Integration:** Realistic trading cost modeling

12.2 Risk Management Enhancements

1. **Position Sizing:** Implement Kelly Criterion or volatility-based sizing
2. **Stop-Loss Mechanisms:** Incorporate maximum drawdown limits
3. **Regime Detection:** Add market condition classification

13 Conclusion

This LSTM made 327,502 dollars over the testing period with a sharpe of 0.05,I tried applying PCA to find the best features and solving but profit was lower in such techniques while the sharpe remained the same.
I have tried using models like CNN,Random Forest but LSTM out performed the other methods by a good margin hence the final submission.// SVM had a close enough profit margin and could outperform my current technique on some other stock, hence included in my submission.// Thank you for giving the time to read my submission and hoping for getting into the program hosted by trexquant.