Predicting Stock Movement with Neural Networks

—— A Data-Driven Approach Using AAPL Return Trends

Nanzhu Li
Stephanie Daniella Hernandez Prado
Serena Zhou

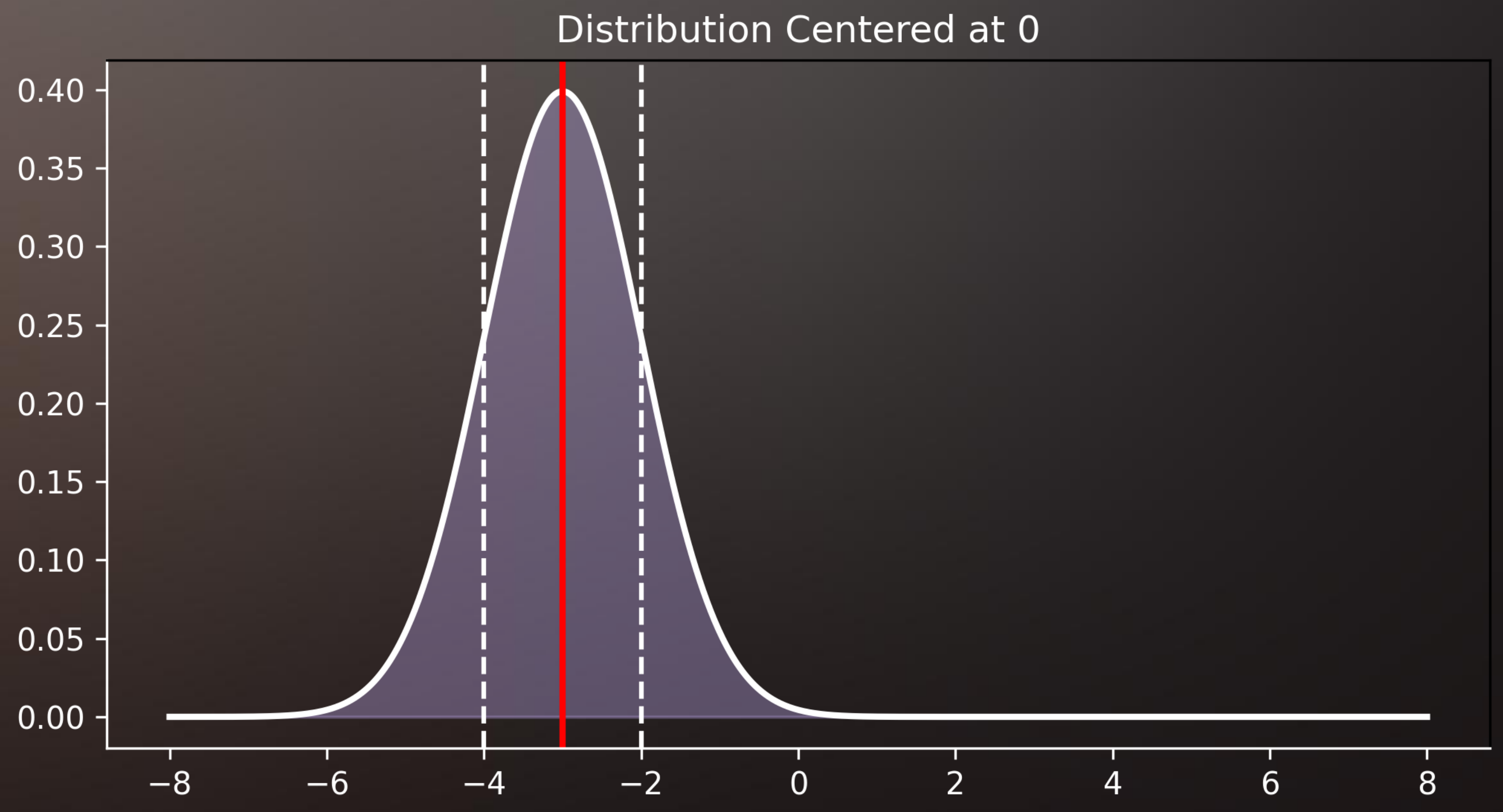
Why Predict Stock Trends?

- Investors aim to anticipate market movements to reduce risk and improve timing.
- Can past returns help us forecast short-term trends?
- **Our goal : We aim to develop a model that classifies short-term stock movement int**

  Uptrend, Downtrend, and Neutral movement

Assumptions

- **Future trends depend on past 15-day return patterns.**
- **Stock returns are assumed to follow a relatively stable distribution over time.**
- **These assumptions form the basis of our feature-label construction.**



Data Processing

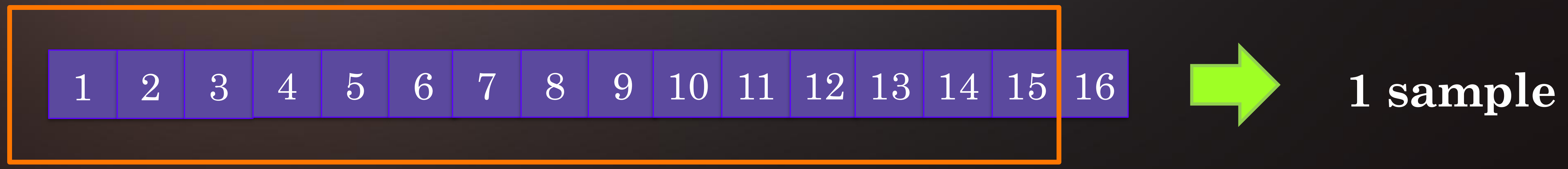
- **Data Sources**

We collected historical stock price data for Apple Inc. (AAPL) from python library yfinance, covering the period from 2012 to 2025.

- **Daily Return Calculation**

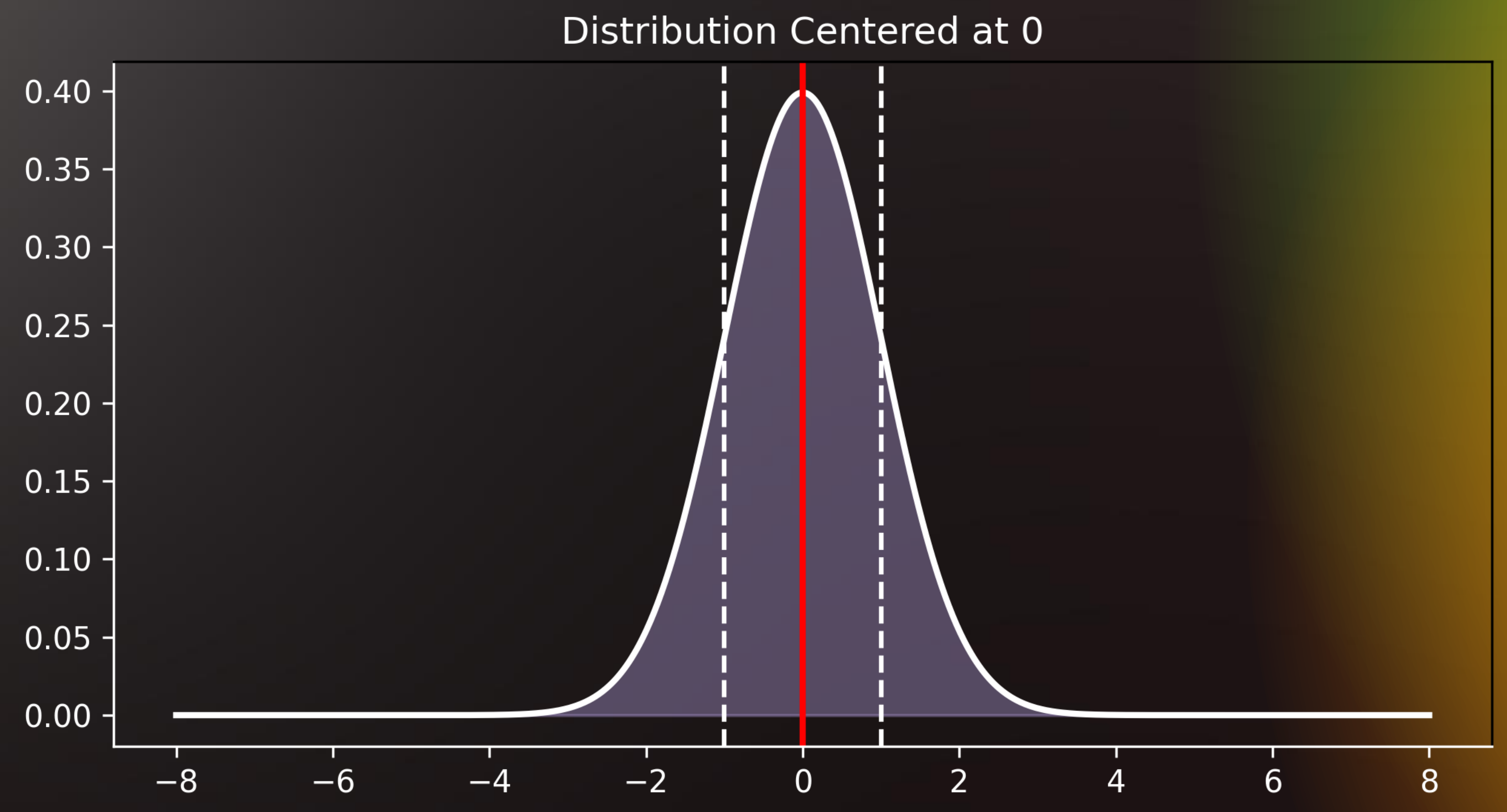
$$\text{Daily return} = (P_t - P_{i-1}) / P_{i-1}$$

- Sliding window : 15 consecutive daily returns



Labeling the Trend

- **Look ahead 5 days after each window**
- **Using the sum of next 5 days to determine label:**
  - If the rolling sum $> 0.3 \times$ std Uptrend
  - If the rolling sum $< -0.3 \times$ std Downtrend
  - Otherwise, Neutral



Labels in Database

Change_1	Change_2	Change_3	Change_4	Change_5	Change_6	Change_7	Change_8	Change_9	Change_10	Change_11	Change_12	Change_13	Change_14	Change_15	Trend
0.005374	0.011102	0.010454	-0.001586	0.003581	-0.001630	-0.002745	-0.003749	0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	2.0
0.011102	0.010454	-0.001586	0.003581	-0.001630	-0.002745	-0.003749	0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	2.0
0.010454	-0.001586	0.003581	-0.001630	-0.002745	-0.003749	0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	1.0
-0.001586	0.003581	-0.001630	-0.002745	-0.003749	0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	0.012811	2.0
0.003581	-0.001630	-0.002745	-0.003749	0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	0.012811	0.007660	1.0
-0.001630	-0.002745	-0.003749	0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	0.012811	0.007660	-0.000635	2.0
-0.002745	-0.003749	0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	0.012811	0.007660	-0.000635	-0.002345	2.0
-0.003749	0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	0.012811	0.007660	-0.000635	-0.002345	0.010019	2.0
0.011649	0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	0.012811	0.007660	-0.000635	-0.002345	0.010019	0.009332	2.0
0.010383	-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	0.012811	0.007660	-0.000635	-0.002345	0.010019	0.009332	0.010475	2.0
-0.003169	-0.017417	0.016917	-0.016378	0.062439	-0.004545	0.005960	0.012811	0.007660	-0.000635	-0.002345	0.010019	0.009332	0.010475	0.016744	2.0

This process generated 3,304 labeled samples, each with 15 input features.

Data Split Strategy

Training 80%

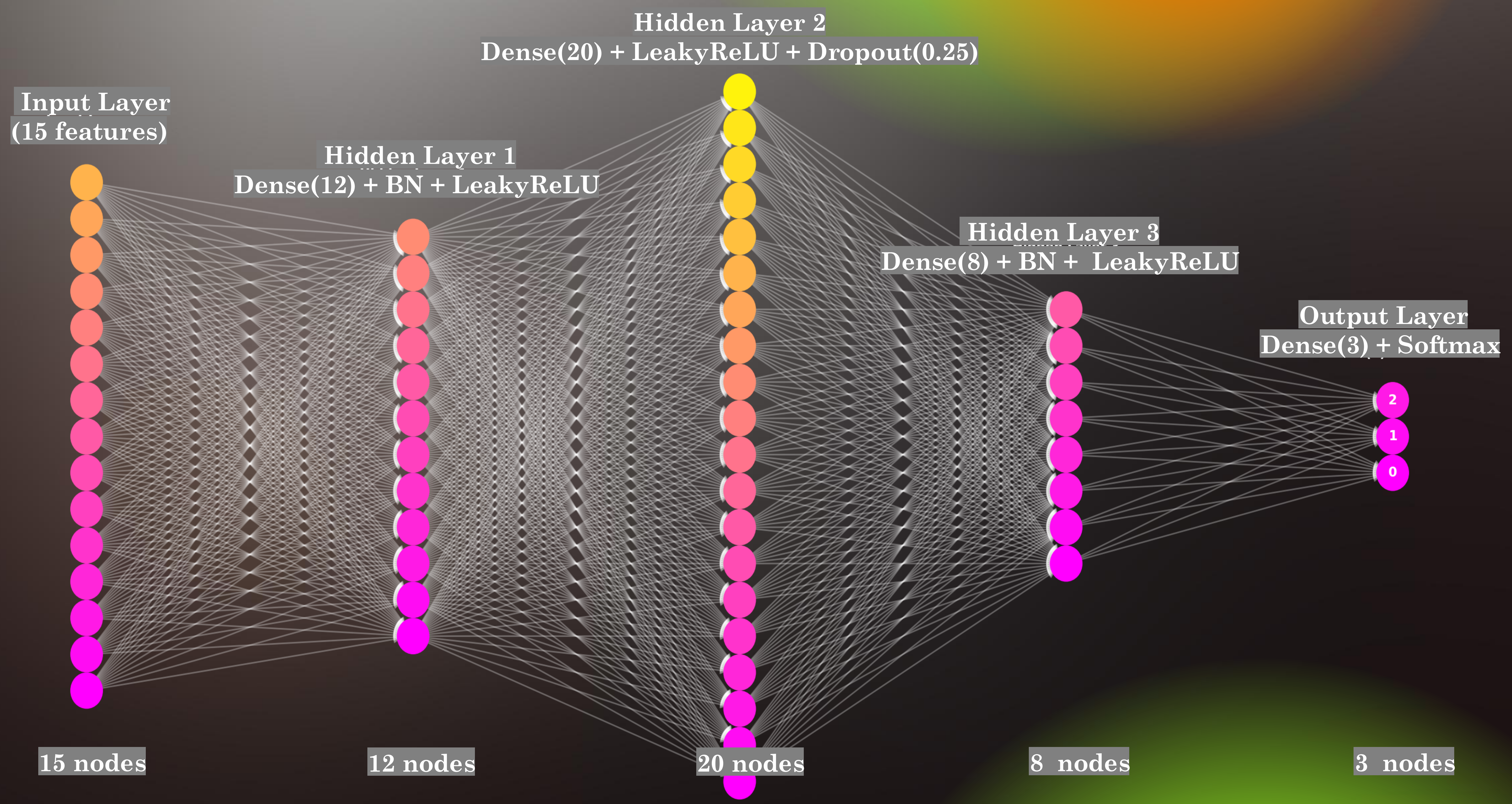
Testing 20%

We split the dataset **chronologically**:

- **80% of the oldest data** was used for training
- **20% of the newest data** was used for testing

Reflects real-world forecasting: learning from the past, testing on the future

Neural Networks



Functions

Sparse Categorical Cross-Entropy

$$L = -\sum_{i=1}^{N} y_i \log(p_i)$$

- N is the number of samples (in a batch)
- y_i is the true class index for sample i
- p_i is the predicted probability for the true class corresponding to sample i

Softmax Function $\sigma(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$

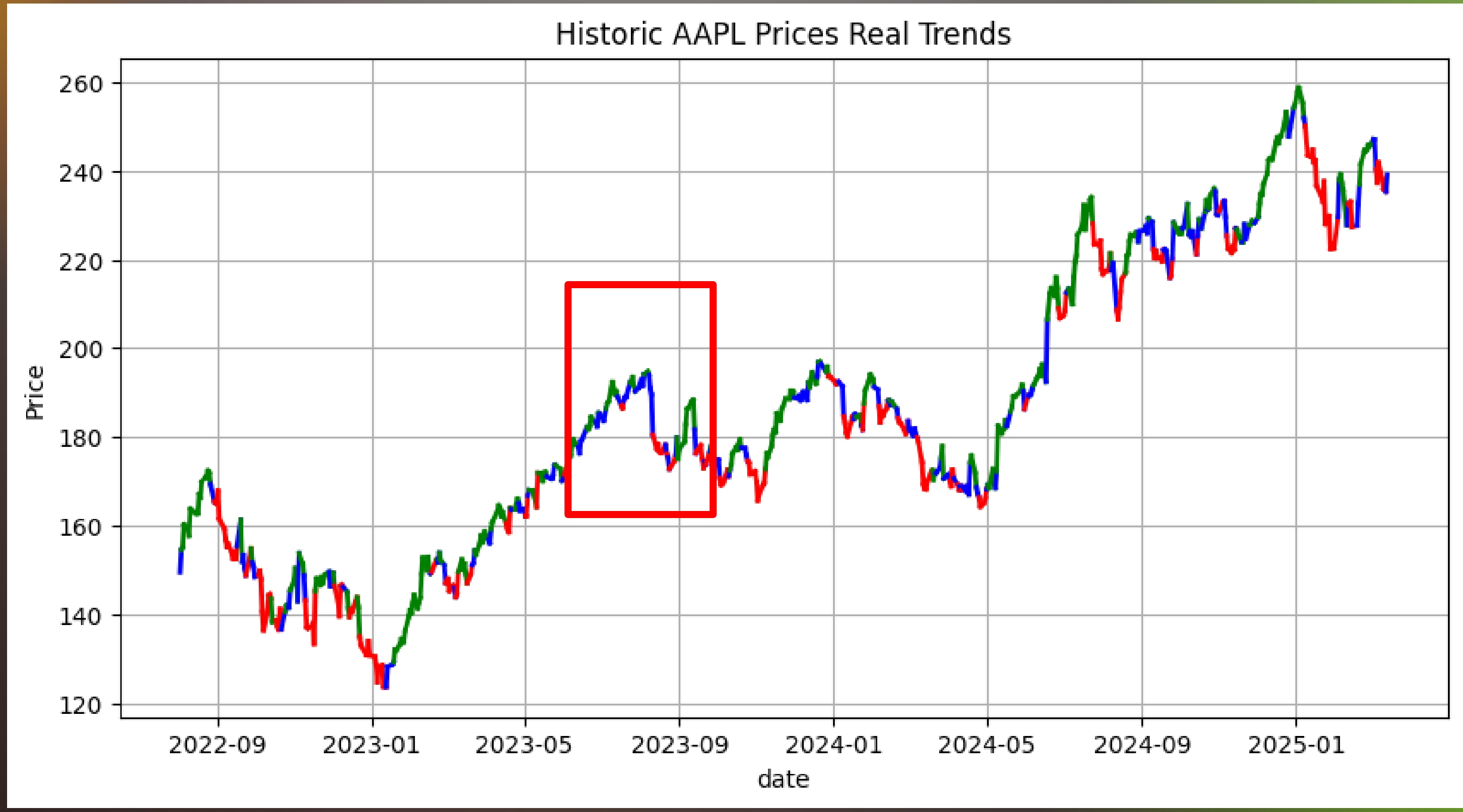
Leaky ReLU $f(x) = \begin{cases} x & if\ x > 0 \\ \alpha x & if\ x \leq 0 \end{cases}$

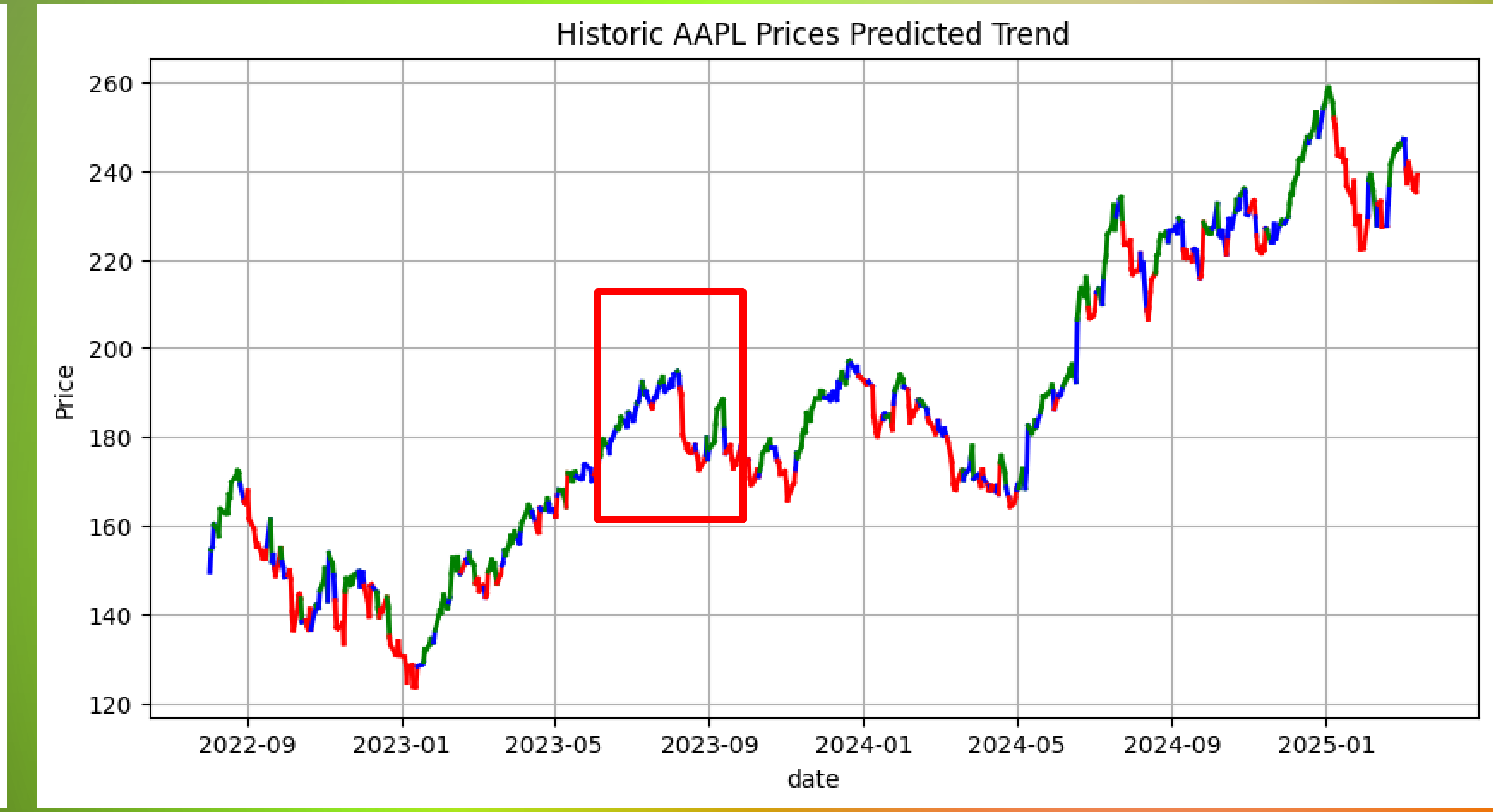
Optimizer: Adam

Batch size: 16

Epochs: 50

Visualizing Trends





How does this model useful in the real life?

- **Neural network can detect short-term trend signals in return data**
- **Labeling method using thresholds is simple but powerful**
- **Model generalizes well to unseen data despite volatility**