**Finance and stock market trend prediction related**

### **1) Lob‑based deep learning models for stock price trend prediction: a benchmark study : 🔹 Innovations (What makes this paper unique)**

1. Introduces **LOBCAST**, an open-source standardized benchmarking framework.
2. Implements **15 state-of-the-art deep learning models** for stock price trend prediction (SPTP).
3. Provides **LOB data preprocessing utilities** (normalization, splitting, labeling).
4. Integrates **hyperparameter tuning with WANDB** for efficient optimization.
5. Generates **detailed reports** on both model performance (F1, Accuracy, Recall) and complexity (training time, inference time, parameters).
6. Supports **backtesting with profit analysis** via Backtesting.py.
7. Reveals that many prior works **overstated model performance** by testing under inconsistent setups.
8. Identifies **BINCTABL** as the most robust and generalizable model, thanks to its adaptive bilinear normalization layer.

### **🔹 Limitations**

1. **High sensitivity to hyperparameters** – many models diverged during training.
2. Only **raw LOB features** were used (to ensure fairness), which limited models designed for richer features.
3. **Computationally expensive grid search** for hyperparameter tuning (needed GPUs and long training time).
4. **Ensemble models** did not outperform the best single models due to high agreement among systems.
5. Dataset imbalance (stationary class dominates) biased some predictions.

### **🔹 Lessons Learned**

1. **Standardized benchmarking is essential** to fairly evaluate financial DL models.
2. **Attention-based models** (like BINCTABL) perform best on LOB data.
3. **Short horizons (k=1–2)** are noisy and hard to predict.
4. **Medium to longer horizons (k=5–10)** yield more stable and accurate predictions.
5. Using **raw LOB features** can sometimes outperform handcrafted features (e.g., CNNLSTM case).

### **🔹 Future Directions**

1. Develop models that are **less sensitive to hyperparameters**.
2. Integrate **richer market features** (e.g., order flow imbalance, trader strategies, news sentiment).
3. Explore **online learning / adaptive models** for dynamic market changes.
4. Improve **explainability (XAI) for LOB models** to build trust in real-world trading.
5. Investigate **class imbalance handling** methods (e.g., better labeling or resampling).

**2) Predicting Stock Price Movements with Combined Deep Learning Models and Two-Tier Metaheuristic Optimization Algorithm**

### **🔹 Innovations (What makes this paper unique)**

1. **Two-tier optimization approach**

* Uses **Dingo Optimizer Algorithm (DOA)** for feature selection (filtering most relevant features).
* Uses **Equilibrium Optimizer (EO)** for hyperparameter tuning of the MHA-BiGRU model.

1. **Combined deep learning model**

* Proposes **MHA-BiGRU (Multi-Head Attention + Bi-directional GRU)**, which captures both temporal dependencies and attention-based feature importance.
* Better handling of noisy, high-dimensional financial time-series data.

1. **Pre-processing improvement**

* Applies **Z-score normalization** to ensure standardized input features, improving model consistency and convergence speed.

1. **Performance achievement**

* Achieved an extremely high correlation value (**CORR = 0.9999**), indicating near-perfect predictive accuracy on experimental datasets.

### **🔹 Limitations**

1. **Overfitting risk** – Very high CORR may indicate **possible overfitting** to experimental data rather than true generalization.
2. **Data constraints** – Only relies on **historical stock and technical indicators**, ignores macroeconomic, news, and sentiment factors.
3. **Computational complexity** – Combining DOA + EO + MHA-BiGRU increases training time and resource requirements.
4. **Interpretability issue** – Deep hybrid models lack transparency, which may reduce investor trust.
5. **Generalization uncertainty** – Model tested on limited datasets; unclear if it performs well in different markets (e.g., US, EU, or emerging markets).

### **🔹 Key Lessons Learned**

1. **Hybrid DL + metaheuristic optimization improves performance** compared to single techniques.
2. **Feature selection matters** — DOA effectively reduces noise and improves computational efficiency.
3. **Attention mechanisms enhance forecasting power** by identifying relevant temporal dependencies.
4. **Hyperparameter tuning is crucial** — EO improves stability and avoids poor manual selection.
5. **Pre-processing consistency is a foundation** — simple steps like Z-score normalization significantly improve model convergence.

### **🔹 Future Research Directions**

1. **Incorporate multi-source data**

* News, social media sentiment, macroeconomic indicators, and alternative data (Google Trends, ESG metrics).

1. **Model interpretability (XAI)**

* Apply SHAP, LIME, or attention visualization to improve transparency for investors.

1. **Robustness under extreme conditions**

* Test on Black Swan events (e.g., COVID crash, geopolitical crises).

1. **Scalability and real-time prediction**

* Adapt PSPMCDL-TTMO for high-frequency trading and real-time updates.

1. **Portfolio-level optimization**

* Extend from single-stock prediction to multi-stock **portfolio selection and risk management**.

1. **Comparative studies with transformers**

* Benchmark against newer architectures like Informer, Temporal Fusion Transformer (TFT), or FinanceBERT.

**3) Title: Stock Price Prediction in the Financial Market Using Machine Learning Models**

# **📘 Paper Overview** **Authors:** Diogo M. Teixeira and Ramiro S. Barbosa **Published in:** *Computation*, 2025, Volume 13, Issue 1, Article 3 **DOI:** [10.3390/computation13010003](https://doi.org/10.3390/computation13010003)

## **1. 🎯 Research Goal**

The paper investigates **how different ML/DL models perform in predicting stock prices**, focusing on:

* **Recurrent Neural Network (RNN)**
* **Long Short-Term Memory (LSTM)**
* **Gated Recurrent Unit (GRU)**
* **Convolutional Neural Network (CNN)**
* **XGBoost**
* **Hybrid models** (e.g., LSTM+GRU, CNN+LSTM, GRU+RNN).

The aim is to identify **which models and feature combinations are best suited for stock price forecasting**.

## **2. 📊 Dataset & Features**

* **Dataset:** Apple Inc. stock prices (1980–2024) from Yahoo Finance.
* **Features:**
* Raw prices (Open, High, Low, Close, Adj Close, Volume).
* **43 technical indicators & economic indices**: SMA, EMA, MACD, RSI, Treasury yields, Oil price, NASDAQ, S&P 500, Dow Jones, NYSE, etc.
* **Feature selection:** Used correlation analysis and SelectKBest → Top **20 most correlated features** were chosen.

## **3. ⚙️ Methodology**

* **Target variable:** Next-day Adjusted Close price.
* **Preprocessing:** Normalization (0–1 scaling), input window size = 100 days (optimized).
* **Validation:** **Time Series Cross Validation (10 folds)** (instead of random splits).
* **Optimization:** Bayesian optimization for hyperparameters.
* **Metrics:** MAE, MSE, RMSE, MAPE, R².

## **4. 🔬 Models Implemented**

* **LSTM** (2–5 layers tested; 2-layer best).
* **GRU** (2-layer best).
* **CNN** (alone + hybrid with LSTM/GRU).
* **RNN** (alone + hybrid with GRU/LSTM).
* **XGBoost** (optimized via GridSearchCV).
* **Total:** 44 model variations tested.

## **5. 📈 Key Results**

* **Best performers:**
* **GRU (2 layers)** → Lowest error on MSE & MAPE.
* **XGBoost** → Best on MAE, RMSE, R².
* **Moderate performer:** LSTM (good but weaker than GRU/XGBoost).
* **Poor performers:** RNN and LSTM+RNN (struggled with volatility).
* **Hybrid models:** Some improved accuracy (CNN+GRU, CNN+LSTM), others worsened performance.

## **6. 🏆 Innovations**

1. **Comprehensive comparison** of deep learning (LSTM, GRU, CNN, RNN) and boosting (XGBoost) models in one framework.
2. **Hybrid architectures tested extensively** (44 variations, unusual in scope).
3. **Feature engineering** combining stock prices, technical indicators, and macroeconomic indices.
4. **Time Series Cross Validation (10 folds)** → More realistic than random splits.
5. **Bayesian hyperparameter optimization** for fairness across models.

## **7. ⚠️ Limitations**

1. **Single dataset (Apple Inc.)** → results may not generalize to all stocks.
2. **Short-term forecast only (next-day prediction)** → no long-term horizon explored.
3. **Limited external signals** → no news, sentiment, or macroeconomic shocks considered.
4. **Hybrid models not always beneficial** (some combos degraded performance).
5. **High computational cost** (training 44 deep models + tuning).

## **8. 🔮 Future Work**

1. **Explore new algorithms** → Transformers, GANs, advanced hybrid deep learning.
2. **Apply to multiple companies/markets** for broader generalization.
3. **Include external features** (financial news, social media sentiment, economic indicators).
4. **Try classification models** (up/down movement) instead of only regression.
5. **Develop ensemble approaches** (mix boosting + DL).
6. **Optimize input window & architecture selection** with smarter validation.

**Extra**

**1.A Review of Stock Price Prediction Techniques using Machine Learning** International Journal for Research in Applied Science & Engineering Technology (IJRASET)

## **📌 Procedure (Step-by-Step)**

1. **Data Collection**

* Stock price data collected from Yahoo Finance (yfinance API).
* Structured dataset (historical stock price, technical indicators).

1. **Feature Engineering**

* Calculation of external indicators:
* Technical: SMA, EMA, RSI, MACD, Bollinger Bands.
* Macroeconomic: GDP, interest rates, unemployment.
* Market sentiment: VIX, trading volume, institutional buy/sell.
* News sentiment (financial news & social media).

1. **Model Design – Multi-Input LSTM**

* LSTM network designed with **gates (input, forget, output)** as shown in the diagram.
* Each input stream (prices + indicators) processed separately.
* Features merged before final prediction.

1. **Training & Evaluation**

* Models trained on historical data.
* Metrics: RMSE, MSE, MAE, MAPE, R², accuracy.
* Compared with other ML/DL models (RF, SVR, CEEMDAN-LSTM, CNN, etc.).

1. **Analysis & Visualization**

* Performance comparison graphs.
* Table summarizing advantages and challenges of past models.

1. **Conclusion**

* Multi-input LSTM provides better accuracy and robustness.

## **📌 Novelty**

* **Multi-Input LSTM**: Unlike normal LSTM models, this approach integrates multiple data streams (stock prices + technical + macroeconomic + sentiment indicators).
* **Structured Dataset Focus**: Prioritizes clean structured stock data over noisy unstructured data (like raw news).
* **Comparison with Literature**: Systematic review of hybrid models (CEEMDAN, EMD, ARIMAX) to identify gaps.
* **Enhanced Feature Context**: External indicators add context, reducing error in predictions.

## **📌 Innovations**

1. **Combination of Traditional & Advanced Models** – A blend of ML regressors, deep learning, and hybrid decomposition methods.
2. **Use of External Indicators in LSTM** – Expands the scope beyond historical prices.
3. **Improved Forecasting Accuracy** – Achieves lower RMSE and MSE compared to classical ML models.
4. **Framework for Multi-Feature Inputs** – Stock data, technical indicators, sentiment, and macroeconomic variables combined in one predictive model.

## **📌 Limitations**

* **High Computational Complexity** – Training LSTM with multiple inputs is resource-heavy.
* **Parameter Sensitivity** – Requires fine-tuning hyperparameters (hidden units, dropout, learning rate).
* **Market Unpredictability** – Sudden crashes, political or global events limit predictive power.
* **Data Constraints** – Mostly structured data used, missing real-time multimodal (text, images, social media).
* **Overfitting Risk** – LSTM may memorize noise if not carefully regularized.

## **📌 Future Integrations**

1. **Hyperparameter Tuning** – Use Bayesian optimization or Grid/Random Search.
2. **Feature Selection/Reduction** – PCA, Autoencoders, or RBM for dimensionality reduction.
3. **Hybrid Architectures** –

* CNN-LSTM (convolutions for feature extraction + LSTM for sequence learning).
* Transformer models (attention for long-term dependencies).

1. **Multimodal Learning** – Integrate structured stock data + financial news sentiment + social media data.
2. **Explainable AI (XAI)** – Use SHAP/LIME to make predictions interpretable for investors.
3. **Cross-Market Expansion** – Apply the model to commodities, forex, and cryptocurrencies.

✅ So, in short:

* **Procedure** → Collect data → Add external features → Build Multi-Input LSTM → Train & Evaluate → Compare.
* **Novelty** → Multi-input approach with structured + external indicators.
* **Innovations** → Combination of ML/DL, hybrid decomposition, and external indicators.
* **Limitations** → High complexity, parameter tuning, unpredictability of markets.
* **Future Integrations** → CNN-LSTM, transformers, XAI, multimodal sentiment analysis.

**2) An Interpretable Framework for Stock Market Forecasting Using Long Short-Term Memory (LSTM) Networks with SHAP-Explainable AI (XAI) Method**

|  |  |
| --- | --- |
| **Aspect** | **Details** |
| **Innovations** | - Integration of LSTM networks (for sequential/time-series prediction) with SHAP (Explainable AI) to enhance interpretability.- Provides feature-level explanations for stock price predictions, addressing the “black box” problem of deep learning.- Combines predictive power and transparency in a single framework. |
| **Limitations** | - Complexity of financial markets may still affect prediction accuracy despite interpretability.- Model trained only on HDFC Bank stock data from 2020–2025; limited generalizability to other stocks or markets.- Uses only historical closing prices; additional features like volume, news sentiment, or macroeconomic indicators not considered.- Computational overhead of SHAP analysis not quantified. |
| **Novelty** | - Directly combines LSTM forecasting with SHAP-based explainability in financial markets.- Provides actionable insights into which historical data points most influence predictions.- Enhances trust and understanding for analysts using AI-based forecasts. |
| **Gap Addressed** | - Lack of interpretability in traditional deep learning models for stock market forecasting.- Difficulty in understanding the contribution of individual historical data points to predictions. |
| **Future Work** | - Incorporate additional input features (volume, news sentiment, macroeconomic indicators, financial ratios).- Test framework across multiple stocks, markets, and longer time periods.- Improve visualization of SHAP explanations for analysts.- Explore real-time application and computational efficiency.- **Compare SHAP with other XAI methods for interpretability insights.** |
| **Dataset** | - Historical closing prices of **HDFC Bank** from **2020 to 2025**, sourced from **Yahoo Finance**.- Preprocessed (normalized) for LSTM training.- Exact size/frequency of dataset not specified. |

### **3) Comparative Analysis of Machine Learning Algorithms in Stock Price Prediction** <https://doi.org/10.54047/bibted.1406867>

|  |  |
| --- | --- |
| **Aspect** | **Details** |
| **Dataset Used** | - **Source**: borsaistanbul.com, investing.com, isyatirim.com.  - **Scope**: Daily data (2016–2020, ~1070 working days) for 5 BIST 100 stocks: QNBFB (QNB Finansbank), ENKAI (Enka İnşaat), FROTO (Ford Otomotiv), EREGL (Ereğli Demir Çelik), KCHOL (Koç Holding).  - **Features**: 22 total (3 internal: Net Profit, Resources, Dividend Income; 19 external: Open/High/Low/Close/Volume/%Change, BIST metrics, Dollar/Euro-TL, Gold (XAU-USD), Brent Oil, S&P 500, Euro Stoxx 50, Interest, Inflation). Data leakage  - **Target**: Daily closing price.  - **Split**: 80% train, 20% test; tested on unseen 2021 data (104 days). |
| **Methodology** | - **Type**: Supervised regression (predict continuous closing prices).  - **Algorithms**: MLR, SVR (linear kernel), DTR, RFR (300 estimators), BRR, ANN (MLP: 4 hidden layers, logistic activation, 1000 epochs, lr=0.0001).  - **Preprocessing**: Cleaning (numeric-only, no empties), Normalization (Min-Max or Standard Scaler), CV (k-fold).  - **Optimization**: Parameter tuning (e.g., epochs for ANN), Feature Selection (forward/backward elimination, heatmap-based; reduced to 7–18 features).  - **Evaluation**: R², MAE, MSE on test/unseen data. |
| **Innovations/Novelty** | - **Unique Combo**: Integrates rare internal balance-sheet features (Net Profit, Dividends) with standard external macros—unlike most studies using only prices/indices.  - **Comparative Depth**: Tests 6 algos with full pipeline (preprocess + select + tune + CV); shows ANN's edge on Turkish market (BIST).  - **Realism Tweaks**: Excludes leaky features (Open/High/Low) in some tests for practical prediction.  - **Hybrid Impact**: Proves feature selection + normalization boosts all algos (e.g., +14% R² for SVR). |
| **Limitations** | - **Data Leakage**: High accuracy partly from same-day Open/High/Low (not truly predictive for end-of-day).  - **Scope**: Only 5 stocks, 5 years; no real-time/news sentiment; ignores unpredictable events (politics/disasters).  - **No Causality**: Correlations (heatmap) don't prove causation.  - **Overfitting Risk**: Despite CV, high R² on test data may not hold in volatile markets. |
| **Gaps** | - **Broader Data**: Limited to BIST; no global/diverse markets or longer periods (e.g., post-2020 COVID effects).  - **Advanced ML**: No deep learning variants (e.g., LSTM for time-series) or ensembles beyond RFR.  - **External Validation**: No comparison to baselines like ARIMA; lacks statistical significance tests (e.g., t-tests on errors).  - **Interpretability**: Black-box algos like ANN hard to explain "why" a price is predicted. |
| **Future Works** | - **More Features**: Expand internals (e.g., board changes); add sentiment from news/social media.  - **Advanced Models**: Hybrid (e.g., ANN + LSTM); time-series specifics.  - **Real-World**: Test on more stocks/years; integrate trading simulations for ROI.  - **Ethics/Scale**: Address biases in macro data; scale to real-time APIs. |

**4) Title : Deep Learning Approaches for Stock Price Prediction: A Comparative Study on Nifty 50 Dataset (pdf downloaded from Researchgate )**

## **🔹 Dataset Used**

* **Dataset:** **Nifty 50 index data (from Kaggle). -> we can use for dhaka stock**
* **Features included:** strike price, call/put open, high, low, close, volume, open interest, and market OHLC (open, high, low, close, volume).
* **Target variable:** Spot Price.
* **Size:** 2,350 training data points and 235 testing data points.

## **🔹 Methodology**

The paper compares **multiple models**:

1. **Baseline:** Linear Regression
2. **Deep Learning Models:** LSTM, GRU, CNN, RNN, TCN
3. **Hybrid Models:** LSTM+GRU, CNN+RNN, CNN+TCN, LSTM+TCN

### **Training & Optimization:**

* **Optimizers tested:** Adadelta, Adagrad, Adamax, Nadam, Adam, SGD, RMSprop.
* **Evaluation Metrics:** MSE, RMSE, MAE, MAPE, R² score.
* **Iterations:** 50 training iterations per model.

## **🔹 Innovations & Novelty**

* **Comprehensive Comparison**: Unlike many works focusing on a single model, this paper **compares multiple deep learning architectures** + their **hybrid combinations**.
* **Optimizer Impact Analysis**: Systematic evaluation of different optimizers (Adamax, Nadam, RMSprop performed best).
* **Hybrid models** explored (e.g., LSTM+GRU, TCN+LSTM) to combine strengths of sequential & convolutional learning.

## **🔹 Key Results**

* **LSTM & GRU performed best** in capturing temporal dependencies.
* **Hybrid models** (like LSTM+GRU, CNN+TCN) sometimes improved results, but depended heavily on optimizer choice.
* **TCN** showed potential but underperformed compared to LSTM/GRU.
* **Best optimizers**: Adamax, Nadam, RMSprop.
* **Worst optimizer**: Adadelta.

## **🔹 Limitations & Gaps**

1. **Single Dataset**: Only Nifty 50 dataset used (no cross-market validation like S&P500, NASDAQ).
2. **Limited Feature Space**: Mostly price & option data; **no sentiment/news data** considered.
3. **Short Training Data**: Only ~2.5k records — deep learning usually benefits from larger datasets.
4. **No Hyperparameter Tuning** details (e.g., number of layers, neurons not deeply optimized).-> wandb used
5. **No Real-Time Testing**: Only historical backtesting, no simulation of real-world trading.

## **🔹 Future Works Suggested**

* Extend models to **other datasets** (global indices, sector-wise stocks).
* Incorporate **sentiment analysis** (news, social media, investor mood).
* Apply **transfer learning** across markets.
* Use **advanced hybrid models** (e.g., Transformer + LSTM, Attention-based TCN).
* Explore **real-time trading strategies** with these models.
* Perform **hyperparameter optimization** (grid search, Bayesian optimization).
* XAI implementation

## **📌 Quick Takeaway**

* **Dataset:** Nifty 50 (Kaggle).
* **Methodology:** Compare regression, deep learning (LSTM, GRU, CNN, RNN, TCN), hybrids.
* **Novelty:** Comparative + optimizer-focused + hybrid testing.
* **Best Models:** LSTM & GRU (with Adamax/Nadam/RMSprop).
* **Limitations:** Small dataset, no sentiment/news, only Indian market.
* **Future Work:** Multi-market datasets, hybrid deep learning + NLP, real-time trading.

***5) An explainable deep learning approach for stock market trend Prediction (Heliyon )***

| **Aspect** | **Details (Important Info)** |
| --- | --- |
| **Dataset** | - **Source:** Yahoo Finance (1990–2022)  - **Markets:** S&P500 (US), DAX30 (Germany), FTSE100 (UK), Nikkei225 (Japan)  - **Data type:** Daily adjusted closing prices  - **Coverage:** 32 years (includes crises: dot-com bubble, 2008 crash, COVID-19)  - **Size:** ~600+ labeled samples per index (after preprocessing)  - **Features:** 270 per sample (15-day window × 18 technical indicators) |
| **Methodology** | - **Trend labeling:** 5 classes (Upward, Downward, Double Top, Rounded Bottom, Rounded Top)  - **Window:** 15-day sliding window (best trade-off vs. 10/20 days)  - **Class balancing:** SMOTE (synthetic oversampling)  - **Feature engineering:** Returns, MA, Volatility, RSI, Momentum + Raw price = 18/day  - **Model:** 6-layer DNN (270 → 135 → 67 → 405 → 200 → 5 classes)  - **Training:** Adam optimizer + categorical cross-entropy, dropout to avoid overfitting  - **Benchmark models:** Random Forest, SVM, Logistic Regression  - **Validation:** 10-fold cross-validation  - **XAI tools:** SHAP (global importance), LIME (local explanations)  - **Experiments:** Full feature set vs. SHAP Top-10 feature set |
| **Innovation** | - First **multi-class trend classification** (5 patterns) instead of price or binary trends  - Combines **deep learning + explainability (SHAP & LIME)** for financial prediction  - Demonstrates trade-off: reduced feature set (Top-10) increases precision/recall while slightly lowering accuracy |
| **Novelty** | - Moves from **binary (up/down)** to **multi-class technical pattern recognition**  - Introduces **XAI transparency** (global + local) in stock-trend prediction  - Uses a **tailored DNN** architecture optimized with grid search |
| **Research Gap Addressed** | - Lack of **multi-class trend models** (prior works mostly regression or binary)  - Lack of **interpretability** in deep models for finance (black-box issue)  - Provides **granular chart-pattern prediction** (not just scalar price prediction) |
| **Limitations** | - Tested only on **historical data**, not real-time  - Uses only **technical indicators** (no sentiment, macroeconomic, or news data)  - SHAP & LIME may **oversimplify feature interactions**  - Deep model may be **data-hungry** (though mitigated by long time series) |
| **Future Work** | - Incorporate **external data** (news, social media, macroeconomic indicators)  - Extend to **real-time / high-frequency trading** scenarios  - Improve **XAI methods** to capture complex feature interactions- Explore hybrid models (DL + other architectures) for robustness |

# **15-Day Sliding Window Approach Explained**

## **Basic Concept**

A **sliding window** is a technique where you take a fixed-size "window" of consecutive data points and move it through your time series data, one step at a time.

Think of it like looking at stock prices through a camera viewfinder that only shows 15 days at once, then you slide forward one day and look at the next 15-day period.

## **Visual Example**

Let's say you have stock closing prices for 30 days:

Day: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...

Price: 100 102 101 105 107 104 108 110 109 112 115 113 116 118 120 119 122 125 123 126 ...

### **Window 1 (Days 1-15):**

[100, 102, 101, 105, 107, 104, 108, 110, 109, 112, 115, 113, 116, 118, 120]

→ Predict trend at Day 16

### **Window 2 (Days 2-16) - Slide forward by 1 day:**

[102, 101, 105, 107, 104, 108, 110, 109, 112, 115, 113, 116, 118, 120, 119]

→ Predict trend at Day 17

### **Window 3 (Days 3-17):**

[101, 105, 107, 104, 108, 110, 109, 112, 115, 113, 116, 118, 120, 119, 122]

→ Predict trend at Day 18

And so on...

## **How It Works in This Paper**

### **Step-by-Step Process**

1. **Identify a Trend**:

* Researchers manually identify where trends occur (e.g., "Upward trend detected on Day 30")

1. **Look Backward 15 Days**:

* Use Days 15-29 (the 15 days before Day 30) as input features

1. **Calculate 18 Features for Each Day**:

* For each of those 15 days, calculate:
* Closing price (1 value)
* Returns (4 values: 1-day, 5-day, 10-day, 15-day)
* Moving average (3 values: 5-day, 10-day, 15-day)
* Volatility (3 values)
* RSI (3 values)
* Momentum (4 values)

1. **Create Input Vector**:

* 15 days × 18 features = **270 values** fed into the neural network

1. **Label**:

* The trend type at Day 30 becomes the label (e.g., "Upward")

## **Sliding Window in Action**

Let's see how many training samples this creates:

If you have **1000 days** of data:

* Possible windows: 1000 - 15 = **985 samples**
* Each sample: 270 input features → 1 trend label

This is why sliding windows are powerful: they **maximize training data** from limited time series.