**Stock Price Prediction and Backtesting with Machine Learning Models**

This script fetches historical stock data, processes it, applies machine learning models for prediction, and generates a PDF report with visualizations and backtesting results. It provides a comprehensive workflow for stock price prediction and backtesting with machine learning.

**1. Importing Libraries**

The script imports necessary libraries for data processing (pandas, numpy), plotting (matplotlib), machine learning (sklearn, XGBoost), and PDF file handling (pypdf).

**2. Fetching Stock Data**

Options for Data Sources:

Yahoo Finance (yfinance) (commented out)

CSV File (utils.get\_csv) (commented out)

Stooq (via pandas\_datareader) (currently used)

After fetching the data, the date index is reset to make it a column.

**3. Data Preparation**

Technical Indicators Calculation

The script calls:

df = utils.add\_technical\_indicators(stock\_data, "Close")

This function adds technical indicators like RSI, MACD, Bollinger Bands, etc.

Sorting and Feature Scaling

The data is sorted by date.

It is scaled for machine learning models:

df\_scaled, scaler = utils.prepare\_features(df)

Creating Sequences for Time Series Modeling

A sequence length of 15 days is used to generate input (X) and output (y) sequences for machine learning models:

X, y = utils.create\_sequences(df\_scaled[19:], sequence\_length)

Splitting Data into Train and Test Sets

80% of the data is used for training, and 20% for testing:

train\_size = int(len(X) \* 0.8)

X\_train, X\_test = X[:train\_size], X[train\_size:]

y\_train, y\_test = y[:train\_size], y[train\_size:]

**4. Creating and Saving the PDF Report**

A PDF file is created to store visualizations:

**5. Data Visualization**

Stock Price Plot

A StockPlotter object is used to visualize the stock price movement and save the plot to the PDF:

**6. Backtesting Trading Strategy**

A backtest strategy is initialized and executed to evaluate trading performance:

A backtest visualization is generated and added to the PDF:

**7. Machine Learning Models for Stock Price Prediction**

LSTM Model

A Long Short-Term Memory (LSTM) neural network is created to predict stock prices.

The model undergoes hyperparameter tuning and training:

best\_params = lstm\_model.tune\_hyperparameters(X\_train, y\_train)

history\_lstm = lstm\_model.fit(X\_train, y\_train, epochs=40, batch\_size=32, validation\_split=0.1)

The training loss is plotted and saved in the PDF:

XGBoost Model

A gradient boosting model (XGBoost) is trained:

xgb\_model = XGBoostModel(n\_estimators=200, max\_depth=5, learning\_rate=0.05)

xgb\_model.fit(X\_train, y\_train, verbose=False)

The training history is plotted:

XGBoost Model with Validation

A validation split (70% train, 15% validation, 15% test) is created for better visualisation:

X\_train\_val, X\_temp, y\_train\_val, y\_temp = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

X\_val, X\_test\_val, y\_val, y\_test\_val = train\_test\_split(X\_temp, y\_temp, test\_size=0.5, random\_state=42)

The model is trained with early stopping:

xgb\_model\_val = XGBoostModelVal(n\_estimators=200, max\_depth=5, learning\_rate=0.05, early\_stopping\_rounds=10)

xgb\_model\_val.fit(X\_train\_val, y\_train\_val, X\_val=X\_val, y\_val=y\_val, verbose=True)

The training history is saved to the PDF:

Gaussian Process Model

A Gaussian Process model (GP) is trained:

gp\_model = Models.GP\_Model()

gp\_model.fit(X\_train, y\_train)

The validation plot is saved to the PDF:

**8. Model Evaluation**

A ModelEvaluator object compares the LSTM, XGBoost, and Gaussian Process models:

models = {

'LSTM': lstm\_model,

'XGBoost': xgb\_model,

'Gaussian Process': y\_pred\_gp

}

evaluator = ModelEvaluator(models, X\_test, y\_test, ticker)

Evaluation results are printed, and performance metrics are visualized and saved:

**9. Time Series Visualization**

The actual vs. predicted values are plotted with uncertainty bands (standard deviation) for the Gaussian Process model:

visualizer = TimeSeriesVisualizer(y\_test, predictions, ticker, std\_devs)

The PDF is finalized and closed:

**10. Merging the PDF with Explanatory Text**

Additional explanatory text (text\_fig.pdf) is merged into the final results PDF:

**Summary of Features**

✅ Fetches stock data (Stooq, Yahoo Finance, or CSV)

✅ Calculates technical indicators

✅ Prepares data for machine learning models

✅ Trains and evaluates three models (LSTM, XGBoost, Gaussian Process)

✅ Performs backtesting of trading strategy

✅ Generates a detailed PDF report