Breakdown of the code:

Libraries Used

- yfinance: Used to download historical stock price data from Yahoo Finance.
- numpy: Used for numerical computations, especially for handling arrays.
- pandas: Used for data manipulation and analysis, particularly with DataFrames.
- sklearn.preprocessing.MinMaxScaler: Used to scale the stock price data. Scaling is important for neural networks.
- tensorflow.keras.models.Sequential: Used to create the LSTM neural network model.
- tensorflow.keras.layers.LSTM, tensorflow.keras.layers.Dense: These are the building blocks of the neural network. LSTM layers are designed to process sequential data like stock prices, and Dense layers are standard fully connected layers.
- matplotlib.pyplot: Used for plotting the results.

Functions

1. fetch_stock_data(symbol, start_date, end_date)
    - Fetches stock price data from Yahoo Finance using the stock symbol, start date, and end date.
    - It handles potential errors during data retrieval and returns None if there's an issue.
    - If successful, it returns a pandas DataFrame containing the stock price data.
2. preprocess_data(df)
    - Preprocesses the stock price data to make it suitable for the LSTM model.
    - It extracts the 'Close' price, which is the target variable we're trying to predict.
    - It scales the data using MinMaxScaler to ensure all values are between 0 and 1. This helps the neural network learn more effectively.
    - It creates sequences of data for the LSTM model. The LSTM model doesn't just look at one day's price; it looks at a sequence of past prices to predict the future price. The sequence_length variable (set to 60) determines how many past days are used in each sequence.
    - It returns the scaled data, the scaler object (which is needed to reverse the scaling later), and the input sequences (X, y). X is the input to the LSTM (sequences of past prices), and y is the target variable (the price after the sequence).
3. build_lstm_model(input_shape)
    - Defines the LSTM neural network model.
    - It's a sequential model, meaning the layers are stacked one after the other.

    - It consists of two LSTM layers and one Dense output layer.
        - The first LSTM layer has 50 units and returns sequences to the next LSTM layer.
        - The second LSTM layer also has 50 units.
        - The Dense layer has 1 unit, which represents the predicted stock price.
    - The model is compiled with the 'adam' optimizer and 'mean_squared_error' loss function, which are commonly used for regression problems.
4. train_and_validate_model(model, X_train, y_train, epochs=25, batch_size=32, validation_split=0.2)
    - Trains the LSTM model using the provided training data (X_train, y_train).
    - It also performs validation during training to monitor the model's performance on unseen data.
    - Key parameters:
        - epochs: The number of times the model goes through the entire training dataset.
        - batch_size: The number of samples per gradient update.
        - validation_split: The fraction of the training data used for validation.
    - It returns the training history, which contains information about the loss and other metrics during training.
5. plot_predictions(model, X_test, y_test, scaler, scaled_data, df, symbol)
    - Generates a plot comparing the model's predictions with the actual stock prices.
    - It takes the trained model, test data (X_test, y_test), the scaler object, the original dataframe df and stock symbol as input.
    - It makes predictions on the test data, inverse transforms the scaled predictions and actual values to their original scale.
    - It plots the actual and predicted prices over time using matplotlib.
6. calculate_rsi(data, window=14)
    - Calculates the Relative Strength Index (RSI), a momentum indicator used in technical analysis.
    - It measures the magnitude of recent price changes to evaluate overbought or oversold conditions in the market.
    - The function takes price data (pandas Series) and a window length (default is 14 days) as input.
    - It returns the RSI values as a pandas Series.
7. plot_rsi(df, symbol)
    - Plots the calculated RSI along with the stock's closing prices.
    - It creates a two-panel plot: the top panel shows the closing price, and the bottom panel shows the RSI.

    - Horizontal lines at RSI 70 and 30 are added to indicate overbought and oversold levels, respectively.
8. calculate_moving_average(data, window=20)
    - Calculates the moving average of the stock price data.
    - The moving average smooths out price fluctuations and helps to identify trends.
    - The function takes price data (pandas Series) and a window length (default is 20 days) as input.
    - It returns the moving average values as a pandas Series
9. plot_moving_average(df, symbol)
    - Plots the moving average along with the actual stock prices.
    - It plots the closing price and the moving average over time, allowing for visual analysis of the trend.
10. main(symbol='AAPL', start_date='2012-01-01', end_date='2024-01-01', epochs=25, batch_size=32)
    - The main function that orchestrates the entire process.
    - It calls the other functions to:
        - Fetch the stock data.
        - Preprocess the data.
        - Split the data into training and testing sets.
        - Build, train, and validate the LSTM model.
        - Plot the predictions.
        - Plot the RSI.
        - Plot the Moving Average
    - It defines the stock symbol, date range, and training parameters.
    - It handles potential errors during the process and prints informative messages.

Key Concepts

- LSTM (Long Short-Term Memory): A type of recurrent neural network (RNN) that is well-suited for time series data like stock prices. LSTMs can learn long-term dependencies, which is important for predicting future prices based on past trends.
- Time Series Data: Data that is ordered sequentially in time, such as stock prices, weather data, and sensor readings.
- Feature Scaling: The process of scaling the data to a specific range, such as 0 to 1. This is important for neural networks to ensure that all features contribute equally to the learning process.
- Training and Testing Data: The data is split into two sets: training data, which is

used to train the model, and testing data, which is used to evaluate the model's performance on unseen data.

- Overfitting: A situation where the model performs well on the training data but poorly on the testing data. This means the model has learned the noise in the training data rather than the underlying pattern.
- Technical Indicators: RSI and Moving Averages are technical indicators used in stock market analysis to identify trends and potential trading opportunities.