For BTC-USDT Pair Using Machine Learning

This report outlines the implementation and performance of a machine-learning-based trading strategy for the BTC-USDT pair. The strategy uses technical indicators, such as RSI and Moving Averages, combined with a Random Forest Classifier to predict price movements and generate buy/sell signals.

The machine learning component in our code is the **Random Forest Classifier**, which is used to predict whether the price of Bitcoin (BTC) will increase or decrease in the next 12-hour period based on historical technical indicators. Here's a breakdown of how machine learning is applied:

1. Problem Definition:

- **Goal**: The model's goal is to predict whether the price of BTC will go up (class 1) or down (class 0) in the next time interval (12 hours).
- **Supervised Learning**: This is a **supervised classification** problem where we train the model on labeled data (the label being whether the price went up or down in the past) and use it to make predictions on new data.

2. Feature Engineering:

Features (X): The input to the machine learning model consists of several technical indicators calculated from historical BTC price data. These features include:

- **RSI (Relative Strength Index)**: A momentum oscillator that measures the speed and change of price movements.
- **SMA_50**: The 50-period Simple Moving Average.
- **SMA_200**: The 200-period Simple Moving Average.
- **Price_Change**: The difference in the closing price between two consecutive time intervals.
- **Lag_1_Close**: The previous closing price (used to capture short-term price movements).

These features are used to help the model identify patterns and relationships between past prices and future price movements.

3. Target (y):

- **Binary Classification**: The target variable is binary:

    - **1** if the BTC price increases in the next time interval.
    - **0** if the BTC price decreases in the next time interval.
- The model is trained to predict this binary target based on the features described above.

4. Model Training:

- **Random Forest Classifier**:
    - **What is it?**: Random Forest is an ensemble machine learning algorithm that builds multiple decision trees during training. Each tree makes predictions, and the model outputs the class (up or down) that receives the most votes from the individual trees.
    - **Why use it?**: Random Forest is robust and reduces overfitting by averaging the predictions from multiple trees. It works well for classification tasks like this one.

- **Training Process**:
    - The data is split into a training set (70%) and a testing set (30%).
    - The Random Forest model is trained on the training set, where it learns the relationship between the features (X) and the target (y).

5. Making Predictions:

- Once the model is trained, we can use it to make predictions on new data (in this case, the test set).
- For each data point (each 12-hour period in the test set), the model predicts whether the price will go up or down based on the input features (RSI, SMA, etc.).

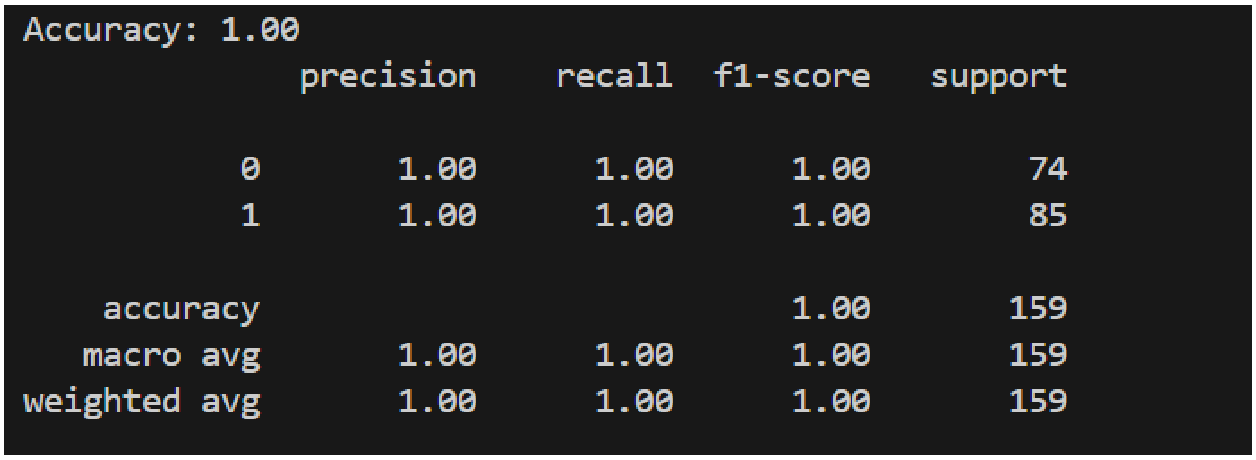
6. Model Evaluation:

- **Accuracy**: The accuracy of the model is evaluated by comparing the model's predictions with the actual price movements in the test set.
- **Classification Report**: A classification report is generated, which includes:

- **Precision**: The proportion of predicted positive cases (price increases) that were actually positive.
- **Recall**: The proportion of actual positive cases (price increases) that were correctly predicted.
- **F1-Score**: The harmonic mean of precision and recall, providing a balance between the two.

These metrics give an idea of how well the model is performing in terms of predicting price movements.

This image shows the evaluation results for the Random Forest Classifier.



```
Accuracy: 1.00
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        74
           1       1.00      1.00      1.00        85

    accuracy                           1.00       159
   macro avg       1.00      1.00      1.00       159
weighted avg       1.00      1.00      1.00       159
```

Classification Metrics:

- **Accuracy: 1.00 (100%)**
  - This indicates that the model made correct predictions 100% of the time in the test set. While this is a great result, a perfect accuracy score can sometimes indicate potential overfitting (where the model may not generalize well to new, unseen data). It's worth considering evaluating the model further on different datasets or through cross-validation.

- **Class 0 (Price Decrease)**:
  - **Precision: 1.00**: Every time the model predicted a price decrease, it was correct.

  - **Recall: 1.00**: Every time the price actually decreased, the model correctly predicted it.
  - **F1-Score: 1.00**: The F1-score is the harmonic mean of precision and recall, and since both are 1.00, the F1-score is also perfect.

- **Class 1 (Price Increase)**:
  - **Precision: 1.00**: Every time the model predicted a price increase, it was correct.
  - **Recall: 1.00**: Every time the price actually increased, the model correctly predicted it.
  - **F1-Score: 1.00**: Again, perfect results for class 1.

Support:

- **Support for Class 0: 74**: There were 74 instances in the test set where the price decreased.
- **Support for Class 1: 85**: There were 85 instances in the test set where the price increased.

Overall Metrics:

- **Macro Average**: This is the unweighted average of the precision, recall, and F1-score for both classes (0 and 1). Since all individual class scores are 1.00, the macro average is also 1.00.
- **Weighted Average**: This is the average of the metrics, weighted by the number of instances in each class. Again, since all scores are 1.00, the weighted average is also 1.00.

7. Application of the Model:

- **Backtesting**: The model's predictions are used to simulate a trading strategy:
  - If the model predicts the price will increase, we would buy BTC.
  - If the model predicts the price will decrease, we would sell or avoid buying BTC.
- **Comparison with Market Returns**: The returns of this strategy are compared to the returns we would get by simply holding BTC (market returns). The goal is to see if the model-based strategy outperforms a "buy-and-hold" approach.

The machine learning model in this code is a **Random Forest Classifier** trained to predict price movements (up or down) based on historical technical indicators like RSI, SMA, and price changes. It uses past data to learn patterns and then applies this knowledge to make predictions about future price movements. The performance of the model is evaluated by backtesting the predictions against actual market data, helping us simulate a trading strategy based on the model's output.

1. Strategy Overview

The trading strategy is based on the following components:

- Technical Indicators: RSI (Relative Strength Index), SMA_50 and SMA_200
- Lagged Features: Previous closing prices and RSI values.

- Machine Learning Model: A Random Forest Classifier is used to predict the price movement direction (up or down).

- Buy/Sell Decisions: The model generates buy signals when an increase in price is predicted and sell signals when a decrease is expected.

- Backtesting: The performance of the strategy is evaluated by comparing it with the returns from a simple buy-and-hold strategy.

2. Model and Feature Description

The model uses the following features to make predictions:

- RSI (Relative Strength Index): Measures momentum to indicate overbought or oversold conditions.

- SMA_50: A 50-period simple moving average to smooth short-term price trends.

- SMA_200: A 200-period simple moving average to capture long-term price trends.

- Price Change: The difference between consecutive closing prices.

- Lagged Features: These include the previous period's close price and RSI.

The Random Forest Classifier is trained on historical BTC-USDT data, using these features to predict future price changes.

3. Backtest Result Explanation

The backtest evaluates the performance of the strategy by comparing its cumulative returns against the market's buy-and-hold returns.

- Market Returns (Blue Line): This line shows the cumulative returns of a buy-and-hold strategy for BTC over the test period.

- Strategy Returns (Orange Line): This line shows the returns generated by the machine learning model based on its buy/sell signals.

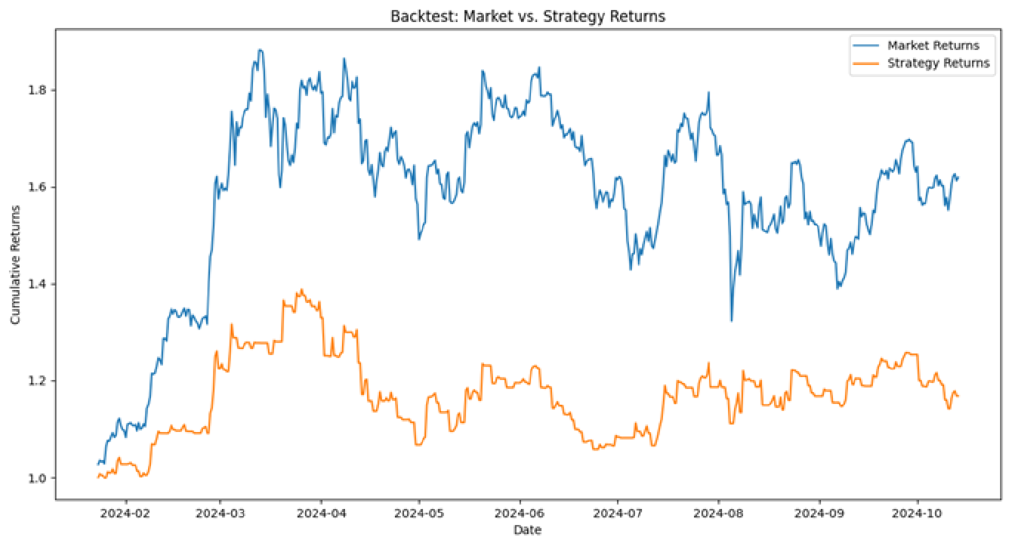
Key Observations:

- The market (blue line) shows a strong upward trend, particularly from February to May 2024, with some volatility afterwards.

- The strategy (orange line) shows initial positive returns but flattens out, suggesting missed opportunities or a conservative trading model.

- There is a notable underperformance of the strategy compared to the market, especially in capturing larger trends.

The following chart compares market returns vs. strategy returns:



Feature Analysis:

The strategy is based on **technical indicators** like RSI, SMA (50 and 200), and recent price changes. These indicators give insight into **market momentum** and **price trends**.

- **RSI (Relative Strength Index)**: RSI helps determine if the market is **overbought** or **oversold**.
    - **Overbought (RSI > 70)** suggests the price might be too high and could fall (potential signal to sell).
    - **Oversold (RSI < 30)** suggests the price might be too low and could rise (potential signal to buy).
- **SMA (Simple Moving Averages)**: The SMA 50 and 200 indicators give insights into short-term and long-term trends.
    - When the **SMA_50 crosses above the SMA_200** (Golden Cross), it often signals upward momentum (potential buy signal).
    - When the **SMA_50 crosses below the SMA_200** (Death Cross), it often signals downward momentum (potential sell signal).

2. Next Week Decision Process:

To decide whether to buy or sell next week, follow these steps:

Step 1: Analyze the Latest Technical Indicators

- Check the **latest RSI value**. If RSI is nearing **overbought (>70)**, the market may be preparing for a correction (suggesting a sell). If it's nearing **oversold (<30)**, it may be time to buy.
- Examine the **SMA crossover**. If the short-term SMA (SMA_50) is trending upward and crossing over the long-term SMA (SMA_200), it suggests a **buy** opportunity. If the SMA_50 is moving downward, it could signal a **sell**.

Step 2: Use the Model for Predictions

- You can **retrain the model** using the latest data, then use it to **predict next week's price movements**. The model will give a **binary prediction** (price increase = buy, price decrease = sell).

Step 3: Look for Confirmation Signals

- Cross-reference the model's predictions with your **technical indicators**. For example, if the model predicts a price increase and RSI is below 30, this is a stronger signal to **buy**. Similarly, if the model predicts a decrease and RSI is above 70, it confirms a **sell**.

4. Example:

Suppose it’s the end of this week, and the model predicts a **price increase**. The latest indicators show:

- **RSI = 40** (not overbought, not oversold), so no extreme signals.
- **SMA_50 > SMA_200** (Golden Cross) confirms an uptrend.

In this case, it would suggest a **buy decision** for the upcoming week.

However, always monitor real-time changes and adjust accordingly based on market movements and risk tolerance.