Assignment (Python + ML)

Problem:

You are provided **1 year of NIFTY intraday historical data** (inside the data/ folder). Your task is to use Python and Machine Learning to predict whether the **next candle's closing price** will go **up or down**, and evaluate the model performance.

Data Set Details:

ColumName Description

Timestamp   Date & time of the price candle

Open   Price at start of the candle

High   Highest price of the candle

Low   Lowest price of the candle

Close   Price at the end of the candle

Task:

- **Label Generation**

    Create a new column:

    - target = 1 → if **next row closing price is higher**

    - target = 0 → if **next row closing price is lower**

- **Model Building**

    - Use at least **two different ML approaches** (example: Logistic Regression, Random Forest, Gradient Boosting, SVM, etc.)

    - Compare models and choose the one with the **highest accuracy.**

- **Train/Test Split**

    - Split the dataset into training and testing sets (for example: 70% train / 30% test or time-based split).

- **Model Evaluation**

    After training, evaluate the chosen model using:

    - **Accuracy**
    - **Precision**
    - **Recall**

- **Model Signal Generation**

  - Using the **testing dataset**, add a new column model_call
    - "buy" if model predicts 1 (price up), else "sell"

- Add another column: model_pnl

  - If model_call is 'buy' → Subtract current close price from model_pnl
  - If model_call is 'sell' → Add current close price to model_pn
  - Iterate row by row in timestamp ascending order
  - Keep updating the cumulative running PnL value

Final Expected Output:

Timestamp	Close	Predicted	model_call	model_pnl
...	19850.2	1	buy	-19850.2
...	19860.1	0	sell	9.9

Submission Guidelines:

- Create a well-structured Python project and push it to a GitHub repository.
- On completion, Provide the repository access to **tradiohub-app**
- Ensure the repository includes runnable Python scripts along with a clear and detailed README file outlining setup steps, execution instructions, and a brief (2–4 lines) explanation of which model performed best and the reasoning behind it.
- Include the final testing dataset in the repository with all additional generated columns as part of the assignment output.