## Data Science Intern Assignment: Algorithmic Trading

**Objective:**
To assess the candidate's ability to analyze financial data, develop predictive models, and implement basic algorithmic trading strategies using data science techniques.

---

## Task 1: Data Analysis and Feature Engineering

**Dataset:**
You are provided with historical stock price data for a company (or use a publicly available dataset like the Yahoo Finance API to get data for any stock of your choice).

1. **Download the data** (e.g., 3 years of daily stock prices: open, close, high, low, volume).
2. **Preprocess the data:**
    - Handle missing values, outliers, and anomalies.
    - Perform feature engineering by creating technical indicators (e.g., Moving Averages, RSI, MACD) that could be useful for algorithmic trading.
3. **Data Visualization:**
    - Plot the stock's price data with relevant technical indicators to visualize trends and potential trading signals.
    - Plot the distribution of returns and volume over time.

**Deliverables for Task 1:**

- A Python script or Jupyter notebook demonstrating the steps of data preprocessing and feature engineering.
- Data visualizations highlighting trends, patterns, and key indicators.

---

## Task 2: Model Building

Now that you have engineered some features, the next task is to develop a predictive model to forecast whether the price will go up or down the next day (binary classification).

1. **Train a predictive model** to forecast the direction of price movement (up/down) for the next trading day using the engineered features.
    - Use machine learning algorithms such as Logistic Regression, Random Forest, or XGBoost.
    - Split the data into training and testing sets.
    - Optimize the model with cross-validation and tuning parameters (if applicable).
2. **Evaluation:**
    - Evaluate the model using accuracy, precision, recall, and F1-score.

    - Also, compute the confusion matrix and ROC-AUC curve to assess performance.

**Deliverables for Task 2:**

- Code for training and evaluating the model.
- A report summarizing the model's performance, and any improvements you would suggest.

---

## Task 3: Backtesting a Simple Trading Strategy

Using the predictive model built in Task 2, you need to create a simple algorithmic trading strategy. The strategy will place a "buy" order if the model predicts the price will go up the next day, and a "sell" order if the model predicts the price will go down.

1. **Backtesting the strategy:**
    - Backtest the trading strategy on historical data and calculate its performance metrics (e.g., cumulative returns, Sharpe ratio).
    - Simulate trade execution with an assumed transaction cost (e.g., 0.1% per trade) and slippage.
2. **Evaluate the performance:**
    - Plot the strategy's performance over time, comparing it against the buy-and-hold strategy.
    - Calculate performance metrics such as cumulative returns, maximum drawdown, and Sharpe ratio.

**Deliverables for Task 3:**

- Python code for the backtesting process.
- A report with the backtest results, strategy performance, and insights.

---

## Task 4: Optimization and Refinement

The next step in improving the algorithm is to optimize it.

1. **Parameter Optimization:**
    - Use techniques like grid search or random search to optimize hyperparameters of the predictive model (e.g., number of trees in Random Forest, learning rate in XGBoost).
2. **Strategy Refinement:**
    - Modify the trading strategy to include risk management (e.g., stop-loss, take-profit).
    - Assess the effect of adding more features or improving the model on strategy performance.

**Deliverables for Task 4:**

- Code for parameter optimization and strategy refinement.
- A brief report discussing the improvements made and the final results after optimization.

---

## Bonus Task: (Optional)

**Alternative Models:**
If time permits, experiment with more advanced models (e.g., Recurrent Neural Networks or LSTM for time-series forecasting). Compare their performance with the traditional machine learning models used in Task 2.

**Deliverables:**

- Code and a comparison of results between models.

---

## Submission Guidelines:

- Submit all code in a Jupyter notebook or Python script format.
- Ensure the code is well-documented, and provide clear explanations for each step.
- Include a brief report summarizing your findings, challenges faced, and improvements you would suggest for future work.

---

## Evaluation Criteria:

1. **Technical Skills:** Ability to preprocess, analyze, and model financial data effectively.
2. **Creativity:** Innovative feature engineering and use of appropriate models and techniques.
3. **Clarity of Code and Reports:** Well-structured code with clear explanations and well-documented reports.
4. **Backtesting and Strategy Performance:** Understanding of trading strategies, and ability to backtest and assess performance metrics.
5. **Communication:** Clear and concise communication of your approach, results, and insights.

---

Good luck, and we look forward to seeing your work!