Algorithmic Trading Project Report

Introduction

Algorithmic trading involves using automated, pre-programmed trading instructions to execute orders in financial markets at high speed and frequency. The goal of this project was to design and implement two core trading strategies—mean-reversion and trend-following—and to integrate these strategies with a broker API to enable real-time trading. Additionally, the project emphasized robust risk management techniques, including the use of Value-at-Risk (VaR), to ensure the trading system could operate effectively in a live environment.

This report is structured into three main sections:

1. **Development of Core Trading Strategies:**
    - Implementation and backtesting of mean-reversion and trend-following strategies using historical data.
2. **Integration with Broker API:**
    - Real-time data streaming, decision-making, and order execution using the Alpaca API.
3. **Risk Management and Advanced Strategy Enhancements:**
    - Incorporating Value-at-Risk (VaR) calculations, order handling, position tracking, and market data validation to improve the system's robustness.

Part 1: Development of Core Trading Strategies

Strategy 1: EMA-ADX (Trend-Following)

Overview

The EMA-ADX strategy is designed to capture sustained trends in the market by combining the Exponential Moving Average (EMA) and the Average Directional Index (ADX).

- **Exponential Moving Average (EMA):** This indicator gives more weight to recent price data, allowing it to react more quickly to price changes compared to a simple moving average. It is used to determine the direction of the trend.
- **Average Directional Index (ADX):** The ADX measures the strength of the trend. A high ADX value indicates a strong trend, while a low ADX value suggests that the market is ranging or the trend is weak.

Implementation

- **Buy Signal:** A buy signal is generated when the ADX value exceeds a specified threshold, indicating a strong trend, and the current price is above the EMA, indicating an upward trend.
- **Sell Signal:** A sell signal is triggered when the ADX is above the threshold and the current price is below the EMA, indicating a downward trend.

- **EMA Calculation:** The EMA is calculated as:

$$EMA_t = \frac{P_t \times (2/(n+1)) + EMA_{t-1} \times (1 - (2/(n+1)))}{n}$$

where P_t is the current price, n is the number of periods, and EMA_{t-1} is the previous EMA value.

- **ADX Calculation:** The ADX is derived from the directional movement indicators (DM+ and DM-) and the true range (TR). It is calculated as:

$$ADX = 100 \times \frac{EMA(DX)}{n}$$

where DX is the directional movement index.

Strategy 2: Bollinger Bands (Mean-Reversion)

Overview

The Bollinger Bands strategy is based on the mean-reversion theory, which posits that prices tend to return to their average over time.

- **Bollinger Bands:** This indicator consists of a middle band, which is typically a 20-period simple moving average (SMA), and two outer bands set at a specified number of standard deviations away from the SMA. The bands expand and contract based on market volatility.

Implementation

- **Buy Signal:** A buy signal is generated when the current price falls below the lower Bollinger Band, suggesting the asset is oversold and may revert upwards.
- **Sell Signal:** A sell signal is generated when the current price rises above the upper Bollinger Band, suggesting the asset is overbought and may revert downwards.

Mathematical Description

- **Bollinger Bands Calculation:**

Mathematical Description

- Bollinger Bands Calculation:

$$\text{Upper Band} = \text{SMA} + (k \times \text{Standard Deviation})$$

$$\text{Lower Band} = \text{SMA} - (k \times \text{Standard Deviation})$$

where k is the number of standard deviations, and SMA is the simple moving average of the closing prices over n periods.

Backtesting Results

We conducted backtesting on three stocks: McDonald's (MCD), PepsiCo (PEP), and Coca-Cola (KO), using 15-minute historical data over a period of approximately three months. The results of the backtesting are summarized below:

Results for McDonald's (MCD):

- **Optimized EMA-ADX Strategy:**
  - **Parameters:** `{'ema_window': 40, 'adx_window': 10, 'adx_threshold': 25}`
  - **Return [%]:** `4.56`
  - **Sharpe Ratio:** `1.31`
  - **Max. Drawdown [%]:** `-6.07`
  - **Win Rate [%]:** `75.0`
- **Optimized Bollinger Bands Strategy:**
  - **Parameters:** `{'window': 25, 'num_std_dev': 3}`
  - **Return [%]:** `2.55`
  - **Sharpe Ratio:** `0.84`
  - **Max. Drawdown [%]:** `-6.01`
  - **Win Rate [%]:** `100.0`

Results for PepsiCo (PEP):

- **Optimized EMA-ADX Strategy:**
  - **Parameters:** `{'ema_window': 15, 'adx_window': 10, 'adx_threshold': 35}`
  - **Return [%]:** `9.77`
  - **Sharpe Ratio:** `1.94`
  - **Max. Drawdown [%]:** `-7.09`
  - **Win Rate [%]:** `100.0`
- **Optimized Bollinger Bands Strategy:**
  - **Parameters:** `{'window': 15, 'num_std_dev': 3}`
  - **Return [%]:** `9.49`
  - **Sharpe Ratio:** `2.11`
  - **Max. Drawdown [%]:** `-6.05`
  - **Win Rate [%]:** `100.0`

Results for Coca-Cola (KO):

- **Optimized EMA-ADX Strategy:**
  - **Parameters:** `{'ema_window': 25, 'adx_window': 20, 'adx_threshold': 35}`
  - **Return [%]:** `5.15`
  - **Sharpe Ratio:** `1.46`
  - **Max. Drawdown [%]:** `-7.55`
  - **Win Rate [%]:** `76.47`
- **Optimized Bollinger Bands Strategy:**
  - **Parameters:** `{'window': 10, 'num_std_dev': 3}`
  - **Result:** No trades executed (possibly due to the conditions never being met).

Strategy Selection

Based on the backtesting results, the EMA-ADX strategy was selected for McDonald's (MCD) and Coca-Cola (KO) due to its higher Sharpe Ratio, lower drawdown, and better win rate. The Bollinger Bands strategy was selected for PepsiCo (PEP) due to its strong performance with a high Sharpe Ratio and consistent win rate.

Part 2: Integration with Broker API

Real-Time Data Streaming and Order Execution

Overview

In this phase, we integrated the trading strategies with the Alpaca API to enable real-time trading. The system was designed to receive live market data, process it to generate trading signals, and execute trades automatically based on these signals.

Key Components

- **Alpaca API Integration:** The Alpaca API was used to stream real-time market data and execute trades. The API provides access to market data, order execution, and account management functionalities.
- **Real-Time Decision Making:** The system continuously processes incoming trade data, calculates the relevant technical indicators (EMA, ADX, Bollinger Bands), and executes trades according to the predefined strategies.
- **Order Handling:** The system includes routines to handle various order events, such as order confirmations, cancellations, and partial fills. This ensures that the trading system can handle real-world scenarios where orders may not be filled immediately or fully.

Challenges

- **Market Timing:** Since the strategies rely on real-time market data, they can only be executed during market hours. This required the system to handle cases where real-time data might be delayed or unavailable, especially outside trading hours.
- **API Rate Limits:** Alpaca imposes rate limits on API calls, so the system needed to be designed to minimize unnecessary API requests and operate efficiently within these limits.

Stream Processing Logic

The stream processing logic was designed to receive trade updates in real-time and trigger the corresponding trading strategy based on the incoming data. Each time a trade update is received, the system calculates the relevant indicators and decides whether to place a buy or sell order based on the strategy's logic.

Part 3: Risk Management and Advanced Enhancements

Value-at-Risk (VaR) Integration

Overview

Value-at-Risk (VaR) is a widely used risk management tool that quantifies the potential loss in value of a portfolio over a specified time period for a given confidence level. By integrating VaR into our trading strategies, we aimed to ensure that trades were only executed when the potential losses were within acceptable limits.

Implementation

- **Historical VaR Calculation:** The VaR was calculated using historical returns data. The process involved:
    - **Step 1:** Collecting a sufficient amount of historical price data for each asset.
    - **Step 2:** Calculating the logarithmic returns of these prices.
    - **Step 3:** Computing the VaR by determining the percentile of the returns distribution at the specified confidence level (e.g., 95%).
- **Risk Threshold:** The calculated VaR was compared against a predefined risk threshold. If the potential loss (VaR) exceeded this threshold, the system refrained from placing the trade.

Integration with Order Execution

- The `place_order_with_var` function was introduced to handle order placement with VaR considerations. This function checks the calculated VaR against the risk threshold and only places an order if the VaR is within acceptable limits.

Order Handling and Position Tracking

- **Reconfirmation of Trades:** After placing an order, the system checks the account to confirm whether the trade has been successfully entered or exited. This step helps ensure that the trade has been executed as intended.
- **Position Management:** The system tracks open positions in real-time and updates them whenever a trade is executed. This ensures that the system maintains an accurate view of the current portfolio and its associated risk.

Market Data Validation

- **Data Quality Checks:** To ensure the reliability of the market data used for decision-making, the system includes basic validation checks. These checks help prevent the execution of trades based on erroneous or incomplete data, such as negative or zero prices.

Performance Evaluation

- **Metrics Computation:** The system calculates key performance metrics such as Drawdowns, Sharpe Ratio, and Value-at-Risk (VaR) to evaluate the effectiveness and risk of the strategies. These metrics provide insights into the system's performance and help identify areas for further improvement.

Conclusion and Future Work

This project successfully demonstrated the development and implementation of two algorithmic trading strategies—EMA-ADX and Bollinger Bands—using Python and the Alpaca API. Through rigorous backtesting, strategy optimization, and the integration of advanced risk management techniques such as VaR, we developed a robust trading system capable of executing trades in real-time.

Key Takeaways

- **EMA-ADX Strategy:** This strategy proved effective for trend-following in stocks like McDonald's (MCD) and Coca-Cola (KO), showing strong performance with acceptable risk.
- **Bollinger Bands Strategy:** This strategy was particularly effective for mean-reversion, especially in PepsiCo (PEP), where it delivered strong returns with a high win rate and effective risk management.
- **Risk Management:** The integration of VaR and real-time position tracking ensures that the system is not only profitable but also safe, minimizing potential losses and protecting trading capital.

Future Work

- **Strategy Diversification:** Expanding the system to include additional strategies and asset classes to further diversify risk and improve returns.
- **Machine Learning Integration:** Incorporating machine learning models to enhance predictive capabilities and adapt the strategies to changing market conditions.
- **Optimization and Scaling:** Refining the strategies and the system architecture to handle higher trading volumes and more complex scenarios, such as multi-asset portfolios.

This report provides a comprehensive summary of the work completed in this project, detailing the development of the trading strategies, their integration with a broker API, and the implementation of advanced risk management techniques. It serves as a solid foundation for further exploration and development in the field of algorithmic trading.