Nomura Quant Challenge 2025 - Strategy Documentation

Overview

This submission implements three sophisticated trading strategies with a focus on robustness, efficiency, and out-of-sample performance. The implementation emphasizes proper handling of transaction costs, data leakage prevention, and comprehensive performance analysis.

Task 1: Individual Strategy Implementation

Strategy 1: Average Weekly Returns

- **Methodology**: Ranks stocks based on their average returns using up to 50 weeks of data
- **Key Features**:
  - Uses all available weeks when fewer than 50 weeks of data exist
  - Takes long positions in bottom 6 stocks and short positions in top 6 stocks
  - Ensures market neutrality with equal weight distribution
- **Implementation Details**:
  - Efficient caching of weekly returns
  - Proper handling of incomplete weeks
  - Vectorized operations for performance

Strategy 2: SMA vs. LMA

- **Methodology**: Uses 30-day LMA and 5-day SMA crossover signals
- **Key Features**:
  - Takes long positions in stocks where SMA > LMA (bottom 5)
  - Takes short positions in stocks where SMA < LMA (top 5)
  - Proper handling of moving average calculations with min_periods
- **Implementation Details**:
  - Efficient caching of moving averages
  - Proper alignment of signals with returns
  - Robust handling of edge cases

Strategy 3: Rate of Change (ROC)

- **Methodology**: Uses 7-day rate of change for momentum signals
- **Key Features**:
    - Takes long positions in stocks with lowest ROC (bottom 4)
    - Takes short positions in stocks with highest ROC (top 4)
    - Proper handling of price changes
- **Implementation Details**:
    - Efficient calculation of ROC
    - Proper handling of missing data
    - Vectorized operations

Strategy 4: Support/Resistance

- **Methodology**: Uses 21-day rolling mean and standard deviation
- **Key Features**:
    - Identifies support and resistance levels using mean ± 3*std
    - Takes long positions in stocks near support (top 4)
    - Takes short positions in stocks near resistance (top 4)
- **Implementation Details**:
    - Proper min_periods handling for rolling calculations
    - Efficient caching of metrics
    - Robust handling of edge cases

Strategy 5: %K Oscillator

- **Methodology**: Uses 14-day %K oscillator for momentum signals
- **Key Features**:
    - Takes long positions in stocks with lowest %K (bottom 3)
    - Takes short positions in stocks with highest %K (top 3)
    - Proper handling of high/low calculations
- **Implementation Details**:
    - Efficient calculation of %K
    - Proper handling of missing data
    - Vectorized operations

Task 2: Strategy Selection

Methodology

- **Walk-Forward Validation**: Uses only data available up to day-1 for decisions
- **Strategy Selection**: Simple but robust Sharpe-based selector
- **Key Features**:

    - Proper handling of transaction costs
    - Efficient vectorized calculations
    - Strategy diversity monitoring
- **Implementation Details**:
    - `SimpleStrategySelector` class for strategy selection
    - Proper handling of lookback periods
    - Strategy persistence for production use

Performance Metrics

- Net Returns
- Sharpe Ratio
- Strategy Diversity Metrics
- Transaction Cost Analysis

Task 3: Ensemble Strategy

Methodology

- **Walk-Forward Validation**: Maintains temporal integrity
- **Parameter Optimization**: Optimizes lookback window using cross-validation
- **Key Features**:
    - Comprehensive performance analysis
    - Strategy diversity monitoring
    - Transaction cost consideration
- **Implementation Details**:
    - `EnsembleSelector` class for strategy selection
    - `TransactionCostCalculator` for cost analysis
    - Comprehensive performance metrics

Performance Analysis

1. **Return Metrics**:

    - Total Return
    - Annualized Return
    - Sharpe Ratio
    - Calmar Ratio

2. **Risk Metrics**:

    - Maximum Drawdown
    - Average Drawdown
    - Drawdown Duration

    - Rolling Volatility

3. **Cost Analysis**:

    - Average Turnover
    - Total Transaction Costs
    - Cost Impact on Returns

4. **Strategy Diversity**:

    - Strategy Usage Distribution
    - Dominance Ratio
    - Strategy Persistence

Overfitting Prevention

1. **Walk-Forward Validation**:

    - Uses only past data for decisions
    - Maintains temporal integrity
    - Prevents look-ahead bias

2. **Parameter Optimization**:

    - Cross-validation for lookback period
    - Multiple performance metrics
    - Strategy diversity monitoring

3. **Robust Implementation**:

    - Proper handling of edge cases
    - Transaction cost consideration
    - Comprehensive error handling

Code Structure

- **Main Components**:

    - Strategy implementations (Task 1)
    - Strategy selection (Task 2)
    - Ensemble implementation (Task 3)
    - Performance analysis
    - Visualization tools

- **Key Classes**:

    - `SimpleStrategySelector`
    - `EnsembleSelector`

    - `TransactionCostCalculator`

- **Helper Functions**:

    - `normalize_weights`
    - `calculate_drawdown`
    - `calculate_rolling_metrics`
    - `calculate_performance_metrics`

Results and Visualizations

1. **Performance Plots**:

    - Cumulative Returns
    - Rolling Volatility
    - Drawdown Analysis
    - Strategy Diversity

2. **Output Files**:

    - task1.csv: Individual strategy performance
    - task2_weights.csv: Selected strategy weights
    - task3_weights.csv: Ensemble strategy weights
    - task_2.csv: Task 2 performance metrics
    - task_3.csv: Task 3 performance metrics
    - task3_performance_analysis.png: Performance visualization
    - task3_strategy_diversity.png: Strategy usage visualization

Future Improvements

1. **Strategy Enhancement**:

    - Additional risk management features
    - More sophisticated parameter optimization
    - Enhanced strategy diversity metrics

2. **Performance Analysis**:

    - Regime analysis
    - Correlation analysis
    - More detailed cost analysis

3. **Implementation**:

    - Parallel processing for optimization
    - Enhanced error handling

    - More comprehensive logging

Ensemble Strategy Documentation

Methodology and Approach

Overview

The ensemble strategy combines multiple trading strategies to improve performance and reduce risk. The approach involves:

- **Strategy Selection**: Using a combination of mean reversion, momentum, volatility, volume, and price level strategies.
- **Weight Calculation**: Dynamically adjusting weights based on recent performance and market regimes.
- **Risk Management**: Implementing volatility targeting and drawdown control to manage risk.

Detailed Approach

1. **Data Preprocessing**: Load and preprocess data from `train_data.csv` and `crossval_data.csv`.
2. **Strategy Implementation**: Implement five distinct strategies:
    - **Strategy 1**: Mean reversion based on average weekly returns.
    - **Strategy 2**: Momentum based on short-term and long-term moving averages.
    - **Strategy 3**: Volatility-based strategy using rate of change.
    - **Strategy 4**: Volume-based strategy identifying support/resistance levels.
    - **Strategy 5**: Price level strategy using the %K oscillator.
3. **Ensemble Selection**: Use a strategy selector to choose the best strategy based on historical performance.
4. **Weight Calculation**: Calculate weights for each strategy using exponential weighted Sharpe ratios.
5. **Risk Management**: Apply volatility targeting and drawdown control to adjust weights dynamically.

Performance Metrics

Key Metrics

- **Total Return**: Measures the overall performance of the strategy.
- **Annualized Return**: Annualized version of the total return.
- **Annualized Volatility**: Measures the risk of the strategy.
- **Sharpe Ratio**: Indicates the risk-adjusted return.
- **Max Drawdown**: Measures the largest drop from peak to trough.

- **Average Turnover**: Indicates the frequency of trading.
- **Win Rate**: Percentage of profitable trades.
- **Profit Factor**: Ratio of gross profit to gross loss.

Overfitting Avoidance

- **Cross-Validation**: Use cross-validation data to validate the strategy's performance.
- **Out-of-Sample Testing**: Test the strategy on unseen data to ensure robustness.
- **Parameter Optimization**: Use grid search to optimize strategy parameters without overfitting.

Code and Visualizations

Code

The code is well-commented and reproducible, ensuring clarity and ease of understanding. Key functions include:

- `backtester_without_TC`: Backtests the strategy without transaction costs.
- `task1_Strategy1`, `task1_Strategy2`, etc.: Implement individual strategies.
- `task3`: Combines strategies and calculates performance metrics.

Visualizations

Visualizations support the analysis by showing:

- Performance metrics over time.
- Strategy weights and their changes.
- Drawdown and volatility targeting effects.

Conclusion

The ensemble strategy demonstrates robust performance through careful strategy selection, dynamic weight adjustment, and risk management. The approach avoids overfitting by using cross-validation and out-of-sample testing, ensuring the strategy's effectiveness on unseen data.