Enhance Pin Bar Trading Strategy with Machine Learning

Kevin Ho

2025-02-25

I) Overview

In the realm of financial trading, technical analysis is a well-known method for identifying trading opportunities that can yield better returns. This paper focuses specifically on candlestick patterns, with an emphasis on the pin bar. The analysis is grounded in a carefully defined set of trading rules regarding the pin bar, which serves as a critical indicator for potential market movements.

To support this analysis, I utilize the dataset titled "33 World-Wide Stock Market Indices Data" from Kaggle. This dataset provides comprehensive historical information on various stock market indices globally, including daily metrics such as opening price, closing price, high, low, and volume. Among the 33 indices included, I specifically focus on the Hang Seng Index (HSI), which represents the stock market of Hong Kong. The HSI is composed of the largest and most liquid companies listed on the Hong Kong Stock Exchange, making it a vital benchmark for the region's market performance.

The dataset spans multiple years, offering a robust timeline for backtesting trading strategies. I first apply the predefined rules to conduct backtesting and assess the accuracy of the trading signals generated. Subsequently, I explore the potential of machine learning to enhance profitability by analyzing patterns associated with pin bars, allowing for some flexibility beyond the rigid rules initially established.

I employ k-Nearest Neighbors (kNN) and Random Forest models to capture the complexities of the pin bar patterns. Finally, I create an ensemble model that combines the strengths of both approaches and re-evaluate the original trading rules against this refined model. Based on the findings, I draw conclusions and implications for future trading strategies.

II) Method Analysis

a) Data Exploration and Data Cleaning

The HSI dataset covers historical data from 1997 to 2022, providing a comprehensive view of the Hong Kong stock market over a significant period.

Column Descriptions

- **Date**: Represents the trading date for each entry. This is crucial for time series analysis.
- **Open**: The price at which the index opened for trading on that date. It reflects the initial market sentiment.
- **High**: The maximum price reached during the trading day. This indicates the peak market performance.
- **Low**: The minimum price recorded during the trading day. It shows the lowest market performance.
- **Close**: The price at which the index closed at the end of the trading day. This is often used to assess market performance and is critical for calculating daily returns.

Data Cleaning Requirements

To prepare the dataset for analysis, the following data cleaning steps may be necessary:

1. **Date Formatting**: Ensure that the date column is in a proper date format for time series analysis. This may involve converting from string to date types.
2. **Handling Missing Values**: Check for any missing or null values in the dataset. If any are found, decide whether to fill them (using methods like interpolation) or remove those entries.
3. **Removing Duplicates**: Verify that there are no duplicate entries for any given date.
4. **Data Type Conversion**: Ensure that numerical columns (Open, High, Low, Close) are in the appropriate numeric format, which may require converting from string representations.
5. **Outlier Detection**: Identify any outliers in the price data that could skew analysis, particularly in the Open, High, Low, and Close columns.

Example Data

Here is a snippet of the dataset for reference:

Date	Open	High	Low	Close
1/2/1997	13362.5	13200.20	13362.5	13203.40
1/3/1997	13241.20	13147.30	13173.30	13222.80
1/6/1997	13443.90	13282.30	13282.30	13443.90
1/7/1997	13564.60	13371.80	13515.60	13420.20
1/8/1997	13501.20	13432.80	13442.10	13454.90
1/9/1997	13412.00	13086.20	13412.00	13198.10
1/10/1997	13335.50	13133.10	13290.30	13191.50
1/13/1997	13310.30	13103.90	13165.20	13289.20
1/14/1997	13328.00	13215.60	13280.20	13293.90
1/15/1997	13771.50	13407.40	13407.40	13766.70
1/16/1997	14004.90	13684.30	13749.70	13830.70
1/17/1997	13977.50	13794.40	13879.20	13856.40

b) Pin Bar Trading Strategy

In this section, I will use the traditional pin bar to identify trading opportunities and evaluate the results of this trading strategy.

Definitions

1. **Bullish Pin Bar**:
    - A candlestick with a small body at the upper end of its range, a long lower wick, and little to no upper wick, indicating a potential reversal from a downtrend to an uptrend.
2. **Bearish Pin Bar (Upside-Down Pin Bar)**:
    - A candlestick with a small body at the lower end of its range, a long upper wick, and little to no lower wick, indicating a potential reversal from an uptrend to a downtrend.

Pin Bar Identification Rules

Bullish Pin Bar Identification The function checks: - The lower wick size is greater than twice the body size. - The body size is less than half the total candle height. - The body is positioned in the upper half of the candlestick range.

```
is_bullish_pin_bar <- function(open, high, low, close) {
  body_size <- abs(close - open)
  lower_wick_size <- open - low
  upper_wick_size <- high - pmax(open, close, na.rm = TRUE)
```

```
  long_lower_wick <- lower_wick_size > 2 * body_size
  small_body <- body_size < (high - low) / 2
  body_at_top <- pmin(open, close, na.rm = TRUE) > (low + (high - low) / 2)

  return(long_lower_wick & small_body & body_at_top)
}
```

Bearish Pin Bar Identification The function checks: - The upper wick size is greater than twice the body size. - The body size is less than half the total candle height. - The body is positioned in the lower half of the candlestick range.

```
is_bearish_pin_bar <- function(open, high, low, close) {
  body_size <- abs(close - open)
  upper_wick_size <- high - pmax(open, close, na.rm = TRUE)
  lower_wick_size <- pmin(open, close) - low

  long_upper_wick <- upper_wick_size > 2 * body_size
  small_body <- body_size < (high - low) / 2
  body_at_bottom <- pmax(open, close, na.rm = TRUE) < (high - (high - low) / 2)

  return(long_upper_wick & small_body & body_at_bottom)
}
```

Entry Rules

Buy Signal (Bullish Pin Bar)

- Triggered when a bullish pin bar is identified. Enter a long position at the close of the pin bar.

Sell Signal (Bearish Pin Bar)

- Triggered when a bearish pin bar is identified. Enter a short position at the close of the pin bar.

Success and Failure Criteria

Bullish Pin Bar

- **Success**: The trade is successful if the price rises above the entry price by a specified target (e.g., +100 pips) before hitting a stop-loss (e.g., -100 pips).
- **Failure**: If the price falls below the stop-loss before reaching the profit target.

Bearish Pin Bar

- **Success**: The trade is successful if the price falls below the entry price by a specified target (e.g., -100 pips) before hitting a stop-loss (e.g., +100 pips).
- **Failure**: If the price rises above the stop-loss before reaching the profit target.

Test Results (Early 2016 - Late 2021)

- **Total Trades Recorded**: 369
- **Buy Signals Triggered**: 247
- **Successful Buy Signals**: 94
- **Sell Signals Triggered**: 122
- **Successful Sell Signals**: 34
- **Overall Success Rate**: 34.69%

The backtesting results from early 2016 to late 2021 indicated an overall success rate of 34.69%. While this demonstrates some level of predictive ability, it is not sufficient for a robust trading strategy, especially considering transaction costs and market volatility.

c) Identify Improvement Areas

Since the result from the traditional pin bar strategy is not that promising, I decided to use machine learning methods to refine the approach. My goal is to explore the potential of machine learning to analyze pin bar patterns and yield better results.

Identifying Deviations from Defined Rules

1. **Partial Pin Bars**: Look for pin bars that meet some but not all criteria. For instance, a pin bar might have a long lower wick but a body that is too large or positioned incorrectly.
2. **Variations in Wick Lengths**: Analyze pin bars with varying wick lengths. Some may have shorter lower wicks or longer upper wicks that are still indicative of potential reversals but do not fit the strict definition.
3. **Contextual Factors**: Consider the broader market context in which these pin bars occur. Patterns that deviate from the rules may still signal significant market behavior if they occur near support or resistance levels.

Feature Engineering with Pin Bar Variations

Features for Pin Bar Analysis

1. **Body Size**: This feature quantifies the size of the pin bar's body, calculated as the absolute difference between the closing and opening prices.

2. **Upper Wick Size**: This feature measures the length of the upper wick, determined by the difference between the high price and the higher of the opening or closing price.
3. **Lower Wick Size**: This feature captures the length of the lower wick, calculated as the difference between the lower of the opening or closing price and the low price.

```
body_size = abs(Close - Open)
upper_wick_size = High - pmax(Open, Close)
lower_wick_size = pmin(Open, Close) - Low
```

Relaxed Rules for Pin Bar Identification

Pin Bar Identification A pin bar is identified when the lower wick is greater than the body size. This allows for some flexibility in the definition.

Upside-Down Pin Bar Identification An upside-down pin bar is recognized when the upper wick is greater than the body size.

```
PinBar = (lower_wick_size > body_size)
UpsideDownPinBar = (upper_wick_size > body_size)
```

Utilizing Machine Learning Models

Model Training Use the engineered features and categorized patterns as inputs for your machine learning models (kNN and Random Forest). This allows the models to learn from both well-defined patterns and those that show potential despite not fully conforming to the rules.

Pattern Recognition Train the models to recognize which deviations from the rules are associated with successful trades. This can reveal hidden patterns and improve overall predictive performance.

d) Apply Machine Learning Model

Data Preparation: Training and Testing

1. Training Data

- The training dataset consists of historical trading data filtered for the date range from January 1, 2001, to December 31, 2015. This data is used to train the machine learning models.
- Features are extracted from this dataset to identify pin bars and upside-down pin bars, which are essential for predicting trade success.

2. Test Data

- The test dataset includes data from January 1, 2016, to December 31, 2021. This dataset is used to evaluate the performance of the trained models.
- Similar feature extraction is applied to ensure consistency with the training data.

Models Used

1. k-Nearest Neighbors (kNN)

- A simple, instance-based learning algorithm used for classification based on the proximity of data points.
- The model is trained using features like body size and wick sizes to predict the success of trades.

2. Random Forest

- An ensemble learning method that constructs multiple decision trees and merges them to improve accuracy.
- Like kNN, it uses the same features to predict trade success, benefiting from its ability to handle complex relationships in the data.

Cross-Validation

- **Time Series Cross-Validation**:
  - The code uses a single split for time series cross-validation rather than multiple folds. This approach divides the dataset into training (80%) and testing (20%) sets.
  - The model is trained on the training set and evaluated on the testing set. This method is particularly useful in time series analysis, where the order of data matters, ensuring that the model is validated on future data points that were not seen during training. This helps assess the model's performance in a realistic setting, reflecting its ability to generalize to new, unseen data.

Measuring Metrics

1. Confusion Matrix

- After making predictions on the test data, confusion matrices are generated for both models (kNN and Random Forest). A confusion matrix summarizes the correct and incorrect classifications made by the model.
- It provides four outcomes:

  - **True Positives (TP)**: Correctly predicted successes.
  - **True Negatives (TN)**: Correctly predicted failures.
  - **False Positives (FP)**: Incorrectly predicted successes.
  - **False Negatives (FN)**: Incorrectly predicted failures.

2. Performance Metrics

- The code extracts several key metrics from the confusion matrix:
  - **Accuracy**: The proportion of total correct predictions (TP + TN) to the total number of predictions.
  - **Sensitivity (Recall)**: The proportion of true positives to the actual positives (TP / (TP + FN)), indicating how well the model identifies successful trades.
  - **Specificity**: The proportion of true negatives to the actual negatives (TN / (TN + FP)), indicating how well the model identifies unsuccessful trades.
  - **Prevalence**: The proportion of actual positive cases in the dataset.

These metrics provide insights into the models' effectiveness and help determine which model performs better in predicting trade success.

III) Modelling Result and Performance

a) kNN Model Analysis

Confusion Matrix

Prediction	Reference 0	Reference 1
0	926	136
1	48	132

Metrics Calculation

- **True Positives (TP)**: 132 (successful trades predicted as successful)
- **False Positives (FP)**: 48 (unsuccessful trades incorrectly predicted as successful)

1. **Total Signal Triggers**:
   - Total Signal Triggers = TP + FP = 132 + 48 = 180
2. **Success Rate**:
   - Success Rate = TP / (TP + FP) = 132 / 180 0.7333 (or 73.33%)

Analysis:

- **Total Signal Triggers (180)**: Indicates that the kNN model suggested entering a trade 180 times.
- **Successful Trades Count (132)**: The model identified 132 profitable trades.
- **Success Rate (73.33%)**: A solid performance, reflecting that over 73% of signals led to profitable trades. However, the model also has a relatively high number of false positives, which may indicate a tendency to signal trades that are not successful.

---

b) Random Forest Model Analysis

Confusion Matrix

Prediction	Reference 0	Reference 1
0	919	340
1	55	128

Prediction	Reference 0	Reference 1

Metrics Calculation

- **True Positives (TP)**: 128 (successful trades predicted as successful)
- **False Positives (FP)**: 55 (unsuccessful trades incorrectly predicted as successful)

1. **Total Signal Triggers**:
   - Total Signal Triggers = TP + FP = 128 + 55 = 183
2. **Success Rate**:
   - Success Rate = TP / (TP + FP) = 128 / 183 0.6989 (or 69.89%)

Analysis:

- **Total Signal Triggers (183)**: Indicates that the Random Forest model suggested entering a trade 183 times.
- **Successful Trades Count (128)**: The model identified 128 profitable trades.
- **Success Rate (69.89%)**: While still a respectable success rate, it is slightly lower than that of the kNN model, indicating that the Random Forest model may be a bit more conservative in its predictions, with a somewhat higher number of false positives than successful trades.

Comparative Summary

Metric	kNN Model	Random Forest Model
True Positives (TP)	132	128
False Positives (FP)	48	55
Total Signal Triggers	180	183
Success Rate	73.33%	69.89%

---

c) Ensemble Model Analysis

Confusion Matrix

Prediction	Reference 0	Reference 1
0	902	281
1	72	187

Metrics Calculation

- **True Positives (TP)**: 187 (successful trades predicted as successful)
- **False Positives (FP)**: 72 (unsuccessful trades incorrectly predicted as successful)
- **True Negatives (TN)**: 902 (unsuccessful trades correctly predicted as unsuccessful)
- **False Negatives (FN)**: 281 (successful trades incorrectly predicted as unsuccessful)

1. **Total Signal Triggers**:
   - Total Signal Triggers = TP + FP = 187 + 72 = 259
2. **Success Rate**:
   - Success Rate = TP / (TP + FP) = 187 / 259 0.7224 (or 72.24%)

Key Metrics

- **Overall Accuracy**: 75.52%
- **Sensitivity**: 92.61%
- **Specificity**: 39.96%
- **Prevalence**: 32.45%

Comparative Summary of All Models

Metric	kNN Model	Random Forest Model	Ensemble Model
True Positives (TP)	132	128	187
False Positives (FP)	48	55	72
True Negatives (TN)	926	919	902
False Negatives (FN)	336	340	281
Total Signal Triggers	180	183	259
Success Rate	73.33%	69.89%	72.24%

Metric	kNN Model	Random Forest Model	Ensemble Model
Overall Accuracy	73.37%	72.61%	75.52%
Sensitivity	95.07%	94.35%	92.61%
Specificity	28.21%	27.35%	39.96%

Analysis

1. **Success Rate**:
    - The success rate of approximately 72.24% indicates that out of 259 signals triggered, around 72% resulted in profitable trades. While this is slightly lower than the kNN model's success rate, it reflects a balanced approach that also improves the identification of unsuccessful trades.
2. **Sensitivity**:
    - The sensitivity of the ensemble model is 92.61%, which is slightly lower than the kNN model's sensitivity. This means it still effectively identifies a high proportion of successful trades but with a slight trade-off in capturing all successful trades.
3. **Specificity**:
    - The specificity of the ensemble model (39.96%) is significantly better than both individual models. This indicates that it has improved its ability to correctly identify unsuccessful trades, reducing the number of false positives.

---

d) Compare Rigid Trading Pin Bar Strategy with Ensemble Model

Let's compare the results of the ensemble model with the rigid trading pin bar rule based on the metrics you've provided. This comparison will help highlight the effectiveness of the ensemble approach versus the traditional trading rule.

Pin Bar Rule Results

Summary of Results - **Total Trades Recorded**: 369 - **Buy Signals Triggered**: 247 - **Successful Buy Signals**: 94 - **Sell Signals Triggered**: 122 - **Successful Sell Signals**: 34 - **Overall Success Rate**: 34.69%

Metrics Calculation for Pin Bar Rule

1. **Total Successful Trades**:
   - Successful Buy Signals: 94
   - Successful Sell Signals: 34
   - Total Successful Trades: 94 + 34 = 128
2. **Total Signals Triggered**:
   - Total Signals: Buy + Sell = 247 + 122 = 369
3. **Success Rate**:
   - The overall success rate is already provided as 34.69%.

Ensemble Model Results Recap

Ensemble Model Metrics - **Total Signal Triggers**: 259 - **Successful Trades (TP)**: 187 - **Overall Success Rate**: 72.24% (derived from the successful trades compared to total signals triggered)

Comparative Summary

Metric	Pin Bar Rule	Ensemble Model
Total Trades Recorded	369	259
Total Successful Trades	128	187
Total Signals Triggered	369	259
Overall Success Rate	34.69%	72.24%

Analysis

1. **Total Trades Recorded**:
   - The ensemble model processed fewer total trades (259) compared to the rigid trading rule (369). This could indicate a more selective trading strategy focusing on higher-quality signals.

2. **Successful Trades**:

    - The ensemble model achieved 187 successful trades, significantly higher than the 128 successful trades from the pin bar rule. This suggests that the ensemble approach better identifies profitable opportunities.

3. **Overall Success Rate**:

    - The ensemble model's success rate of 72.24% far surpasses the pin bar rule's success rate of 34.69%. This indicates that the ensemble model is much more effective at generating profitable trades relative to the number of signals it triggers.

IV) Conclusion

The comparison shows that the ensemble model outperforms the traditional pin bar trading rule in several key areas:

- Higher overall success rate
- Greater number of successful trades
- More selective trading with fewer signals triggered

This analysis illustrates the potential benefits of using an ensemble approach in trading strategies, especially in terms of profitability and efficiency.

Limitations

1. **Complexity**:
    - Ensemble models are generally more complex and may require more computational resources compared to simple rules. This can pose challenges for traders with limited technical expertise.
2. **Data Dependency**:
    - The performance of machine learning models, including ensembles, is heavily dependent on the quality and quantity of data. Poor-quality data can lead to inaccurate predictions.
3. **Overfitting Risk**:
    - There is a risk of overfitting, where the model performs well on training data but poorly on unseen data. Careful validation and testing are necessary to ensure generalizability.
4. **Market Changes**:
    - Markets can change rapidly due to economic, political, or social factors. A model that performs well in one period may not necessarily perform well in another, necessitating continuous monitoring and adjustments.

Future Work

1. **Model Refinement**:
    - Further tuning of the ensemble model's parameters and exploring additional algorithms can enhance performance. Techniques such as hyperparameter optimization and feature selection should be considered.
2. **Incorporating More Features**:

- Future iterations of the model could benefit from incorporating more diverse features, such as sentiment analysis, macroeconomic indicators, or technical indicators, to improve predictive accuracy.

3. **Real-Time Implementation**:
    - Implementing the ensemble model in a real-time trading environment will provide insights into its practical viability and performance. This will also help in assessing the impact of transaction costs on profitability.
4. **Comparative Studies**:
    - Conducting comparative studies between different ensemble techniques (e.g., bagging, boosting) and other machine learning models could yield insights into which methods work best under varying market conditions.
5. **User-Friendly Interfaces**:
    - Developing user-friendly interfaces for traders to utilize these models can increase accessibility. Providing visualizations and insights can help traders make informed decisions without needing deep technical knowledge.
6. **Continuous Learning**:
    - Implementing mechanisms for the model to continuously learn from new data and adapt over time can enhance its effectiveness and resilience against changing market dynamics.

The ensemble model demonstrates substantial advantages over the rigid trading pin bar rule, particularly in terms of profitability and decision-making. However, its complexity and data dependency pose challenges that must be addressed. Future work focused on refining the model, incorporating additional features, and facilitating real-time implementation will be crucial for maximizing its potential impact in trading strategies.

V)References

Datasets

- **Kaggle**: A platform that hosts various datasets related to finance and trading, including stock market data and trading strategies.
  - Kaggle Datasets