

卒 業 論 文

テクニカル指標の組み合わせによる トレーディングアルゴリズムの提案

九州工業大学 情報工学部
情報・通信工学科

オンコン マハルク ラフマン

2023 年度

指導教員：藤原 暁宏

概 要

近年，金融分野で数学や情報科学を応用したフィナンシャルテクノロジーが注目を浴びている．その中でも，株取引においては，アルゴリズムを用いて自動的に注文を判断し実行するアルゴリズム取引が活発に行われている．

このアルゴリズム取引においては，ゴールデンクロス・デッドクロスや MACD などのテクニカル指標を用いた取引が多く用いられており，利益やリスクの少なさの点でも優れているとされている．また，損切りや防衛損切り，利益確定を行うことで最終利益が向上する傾向があることも一般的に知られている．

本研究では複数のテクニカル指標を組み合わせたアルゴリズムを提案するとともに，トレーディングツールを用いて検証し，テクニカル指標の有効な組み合わせを検証することを目指す．テクニカル指標は MACD，ボリンジャーバンド，60 日移動平均線，日経平均株価の 4 つを用いて，MACD とボリンジャーバンドを主体に既存も含めて 13 通りのアルゴリズムを提案する．

また，提案アルゴリズムについては実際の東証の株価データを用いたバックテストを通じて，有効性の検証を行う．検証結果より，4 つの提案したアルゴリズムにより，10 年間の期間において 100 % を超える利益が得られることを示す．

目次

第1章	はじめに	1
第2章	準備	3
2.1	株について	3
2.1.1	株	3
2.1.2	株価	3
2.1.3	株価データ	3
2.1.4	株価チャート	3
2.1.5	移動平均線	4
2.1.6	ローソク足チャート	4
2.1.7	日経平均株価	4
2.1.8	利益確定	4
2.1.9	損切り	5
2.2	テクニカル指標	5
2.2.1	MACD(Moving Average Convergence and Divergence)	5
2.2.2	ボリンジャーバンド (BB: Bollinger Band)	6
2.3	Python によるバックテスト環境	7
2.3.1	Python の概要	7
2.3.2	研究で使用するライブラリ	8
第3章	トレーディングアルゴリズム	16
3.1	MACD	16
3.1.1	MACD のみ	16
3.1.2	MACD と 60 日移動平均線	16
3.1.3	MACD と日経株価平均	18
3.1.4	MACD と日経株価平均と長期線	19
3.2	BB	21
3.2.1	BB のみ	22
3.2.2	BB と 60 日移動平均線	23
3.2.3	BB と日経平均株価	24
3.2.4	BB と日経株価平均と長期線	25

3.3	MACD と BB	27
3.3.1	MACD と BB	27
3.3.2	MACD と BB と 60 日移動平均線	32
第 4 章	実験結果	37
4.1	実験環境	37
4.2	上位 95 社の株価データ	38
4.3	トレーディングアルゴリズムの検証	39
4.4	実験結果と考察	47
第 5 章	まとめ	48
	謝辞	49
	参考文献	50

第1章 はじめに

近年、金融分野に対して数学や情報科学を活用したファイナンシャルテクノロジーが注目を集めている。このファイナンシャルテクノロジーの一分野として、資産運用の一つである株取引に対して、あらかじめ定めておいた手順に従い自動的に注文の数量やタイミングを判断して取引を実行するアルゴリズム取引が盛んに行われている。また、様々なトレーディングアルゴリズムを過去のビッグデータを用いて検証可能なトレーディングツールも多数提案されている。

このトレーディングアルゴリズムに対するこれまでの研究例として、森本の研究 [1] がある。森本は該当研究において、一定期間における安値の最低値を計算し、その変動を折れ線グラフで表現した Low グラフと、一定期間における高値の最高値を計算し、その変動を折れ線グラフで表現した High グラフに着目し、売買タイミングを判断するアルゴリズムの考察と提案をしている。該当研究では考察・提案した複数のアルゴリズムで既存の取引手法である、ゴールデンクロス・デッドクロスを用いたアルゴリズムが最終利益、リスクの少なさの点で最も優れることを示している。また、損切りや防衛損切り、利益確定を行うことで最終利益が向上する傾向があることも示されている。

本研究では、様々なテクニカル指標を組み合わせて有効なトレーディングアルゴリズムの提案を目標とする。また、提案アルゴリズムについては、トレーディングツールを用いて有効性の検証を行う。用いる指標は MACD、ボリンジャーバンド、60 日移動平均線、日経平均株価の 4 つであり、MACD とボリンジャーバンドを主体に既存アルゴリズムも含めて 13 通りのアルゴリズムを提案する。

また、提案アルゴリズムは、株価データに対してバックテストを行うためのツールの一つである `backtesting.py` を用いて実装するとともに、実際の東証の上位の株価データに対する適用を行い、提案アルゴリズムの有効性の検証を行う。検証結果より、既存の 2 つのアルゴリズムと、4 つの提案したアルゴリズムにより、10 年間の期間において 100 % を超える利益が得られることを示す。また、利益、取引数は低い平均勝率が約 67 % となる提案アルゴリズムも示す。

本論文は、以下のように構築される。第 2 章では、Python 言語の概要、本研究で使った Python 言語のライブラリ、及び、株の基本知識と株取引で使われているテクニカル分析の指標について説明する。第 3 章では、本研究のテーマである Python 言語を用いたトレーディングアルゴリズムについて述べる。第 4 章では、アルゴリズムの実行結果について述べる。第 5 章では、本研究のまとめを行い今後の課

題について言及する .

第2章 準備

本章では，Python 言語の概要，本研究で使った Python 言語のライブラリ，及び，株の基本知識と株取引で使われているテクニカル分析の指標について述べる．

2.1 株について

本節では，株に関する基礎知識と専門用語について簡単に説明する．

2.1.1 株

株とは企業が資金調達して事業行うために発行しているものである．株を買って保有することで出資者となり，会社のオーナーの一人になることができる [15]．

2.1.2 株価

株価とは，株式市場において売買が成立したときの株の価格のことである [2]．

2.1.3 株価データ

株価データとはある一定期間の株価の始値，高値，終値をまとめたものである．期間は分，日，週，月，年等がある．短期の株価を参照する場合は3分，5分などの分単位，長期的な場合は週や月単位の株価データを用いる．よく使われているのは日単位が多く，本研究も日単位を用いる [2]．

2.1.4 株価チャート

株価チャートとは株価データをグラフ化し，見やすくしたものである．主に移動平均線とローソク足チャートが取引に使われている [2]．

2.1.5 移動平均線

移動平均線とは，一定期間の株価の終値より平均値を計算し，その値を元に折れ線グラフで表したものである．毎日の株価の平均を計算するため，平均値が移動していくことから移動平均または Moving average(MA) と呼ばれる．

n 日間の移動平均線のことを n 日移動平均線と呼ぶ [2]．本論文では提案アルゴリズムにおいて 60 日移動平均線を用いる．

2.1.6 ローソク足チャート

ローソク足とは，一定時間内の始値，高値，安値，終値を一本の蠟燭型のグラフに表したものである．終値が始値より高い場合を株価分析ではローソク足が用いられることが多い．本論文では，株価チャートとして用いる [2]．

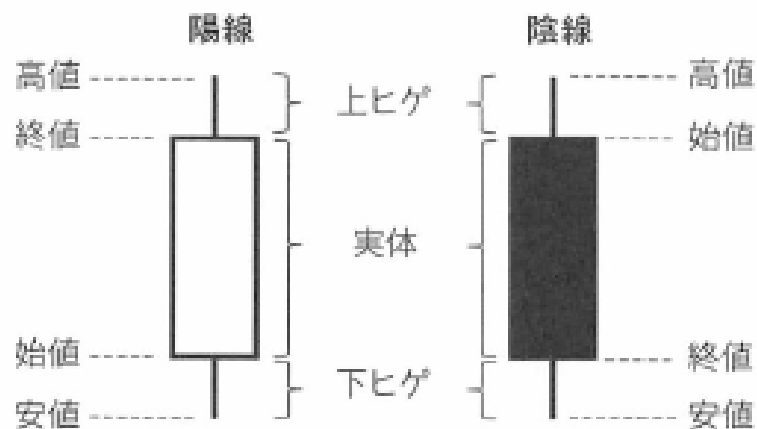


図 2.1 ローソク足 (引用元 [2])

2.1.7 日経平均株価

日経平均株価とは，日本経済新聞社が東京証券取引所プライム市場上場銘柄から選定した 225 銘柄の株価の平均ことである [16]．

2.1.8 利益確定

利益確定とは，買った株の値段が上がったときに売却し，利益を確定することである．「利食い売り」ともいわれる．ある程度利益を確定して売却することで損失に転換するリスクを可能性を回避できる [12]．

2.1.9 損切り

損切りとは、損失を抱えている状態で抱えている株を売却して損失を確定させることである。ロスカットやストップロスなどとも呼ばれている。価格が下落し続けている株を保有しておくことで損害が大きくなるため、損切りで損失額を抑えることができる [13]。

2.2 テクニカル指標

2.2.1 MACD(Moving Average Convergence and Divergence)

短期と中期の移動平均線がどちらも横ばいか上向きであり、短期移動平均線を中期移動平均線が下から上に突き抜ける買いのタイミングをゴールデンクロスと呼ぶ。MACD[2] は、1970 年代後半に米国のシグナラート・コーポレーション社の経営者であったジェラルド・アペル氏によって考案された指標であり、このゴールデンクロスをもっと汎用的にしたテクニカル指標である。MACD では移動平均として、単純移動平均ではなく、直近の価格に比重をおく平滑移動平均を用いる。MACD は以下の 3 つの要素で構成されており、各要素の求め方は以下の通りである。

- $\text{MACD 線} = X \text{ 期間の短期移動平均線 (EMA)} - Y \text{ 期間の長期移動平均線 (EMA)}$
- $\text{シグナル線} = \text{MACD の } Z \text{ 期間の単純移動平均線 (EMA)}$
- $\text{ヒストグラム (OSCI)} = \text{MACD 線} - \text{シグナル線}$

X, Y, Z はパラメータで、本研究では $X=5$, $Y=20$, $Z=9$ を用いる。MACD とシグナルは折れ線グラフで表し、ヒストグラムは棒グラフで表す [2]。MACD の買いシグナルは、MACD 線がシグナル線を下から上に突き抜ける場合で、売りシグナルは、シグナル線が MACD 線を下から上に突き抜ける場合である。

図 2.2 で MACD の例を示す。この図では MACD の MACD 線、シグナル線、ヒストグラムが示されている。MACD 線がシグナル線が上から下に突き抜けた時に売りシグナルが出ており、売りのタイミングであることがわかる。同様に、MACD 線がシグナル線を下から上に突き抜けたときにシグナルが出ており、買いのタイミングであることがわかる。ヒストグラムは、売りシグナルのとき正から負になっており、買いシグナルのときは負から正になっていることがわかる。

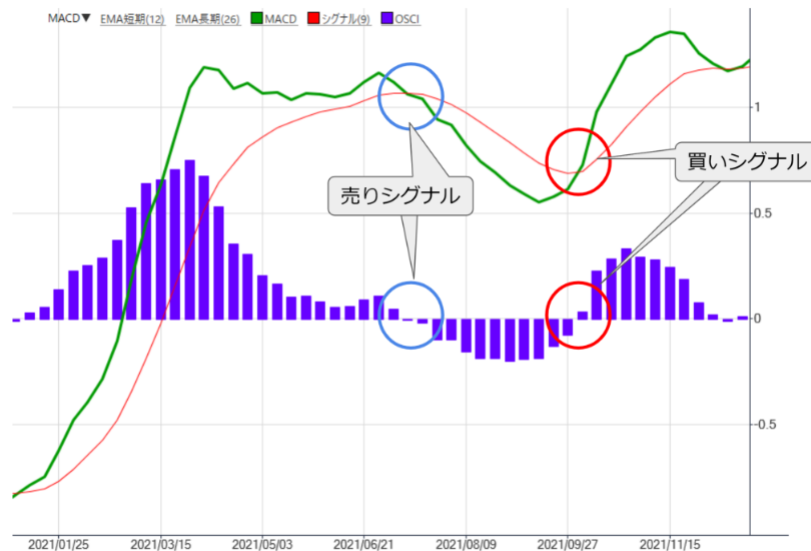


図 2.2 MACD グラフ (引用元 [14])

2.2.2 ボリンジャーバンド (BB: Bollinger Band)

ボリンジャーバンド [2] とは移動平均線の上下に標準偏差を元に計算された 3 つのバンドという線を用いるテクニカル指標である．ボリンジャーバンドの定義式は以下になる．

- 中央のバンド = n 日の移動平均線
- 上部バンド = 中央のバンド + (n 日の標準偏差 $\times X$)
- 下部バンド = 中央のバンド - (n 日の標準偏差 $\times X$)

n は期間のことであり，本研究では 25 日を用いている． X は標準偏差の倍数で 1 から 3 までであり，単位は で扱う．バンドは +1 (アッパーバンド 1)，+2 (アッパーバンド 2)，+3 (アッパーバンド 3)，-1 (ロワーバンド 1)，-2 (ロワーバンド 2)，-3 (ロワーバンド 3) の名前で扱う．株価の変動はバンドの範囲内に収まることが多いと考えられており，ばらつきが多いほど標準偏差は大きくなるため，バンドの幅が広くなる方に値動きが大きくなると判断できる．株価が収まる確率は以下の通りだとされている [2]．

- ± 1 : 68.2 %
- ± 2 : 95.4 %
- ± 3 : 99.7 %

BB はローソク足とともに表示されることが多く、ローソク足の位置で株価の上がり下りの余地がどれくらいあるかを判断する． ± 3 は ± 2 との差が 4 % しかないため指標としてはあまり重視されない．本論でも ± 2 をつかって検証する．

図 2.3 は BB の例を示している．移動平均線を基準に ± 3 , ± 2 , ± 1 のバンドを示している．BB の買い売りシグナルはいくつかあるが本論では、終値が ± 2 を下回ったときに買いシグナルを出すアルゴリズムを提案している．

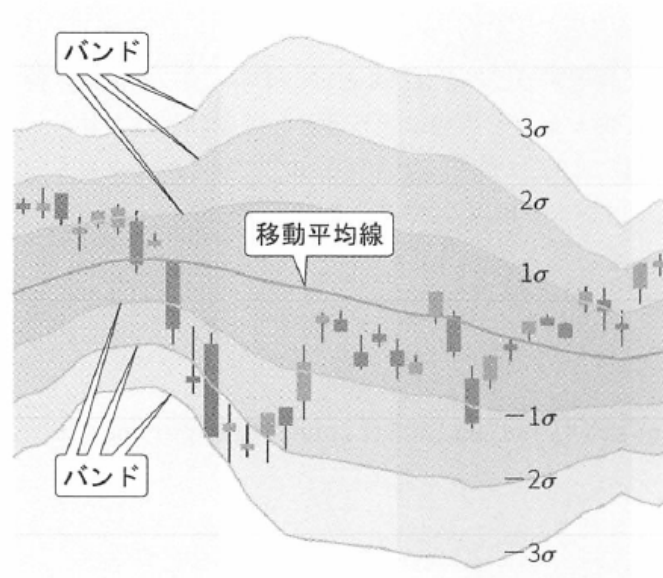


図 2.3 BB グラフ (引用元 [2])

2.3 Python によるバックテスト環境

2.3.1 Python の概要

Python[2, 4, 17] は高水準のプログラミング言語であり、直感的で読みやすい構文を持つことが特徴である．1991 年に Guido van Rossum によって設計され、その後コミュニティによって発展していった．Python は汎用性が高く、ウェブ開発、データ解析、機械学習、人工知能などさまざまな分野で広く利用されている．

Python の特徴的な構文として、インデント（行頭の空白文字）を使ったブロック構造があり、これにより可読性が向上し、他のプログラミング言語よりもコードが書きやすい．また、動的型付けを採用しており、変数の型を事前に宣言する必要がないため柔軟性がある．標準ライブラリが豊富であり、多くの便利なモジュールが組み込まれている．これにより、ファイル操作、ネットワーク通信、データ処理などの様々なタスクを容易に実行することができる．

Python はオープンソースで、コミュニティによって広くサポートされており、さらに豊富なドキュメンテーションやオンラインリソースが利用可能であるため、初学者から経験豊富な開発者まで幅広い層に利用されている。簡潔かつ表現力豊かなコードを書くことができ、初学者にも優しい言語であるため、プログラミングの入門に適している。Python は今日では非常に人気があり、多くの大規模なプロジェクトや企業で採用されている。

2.3.2 研究で使用するライブラリ

ここでは研究で用いたライブラリについて説明していく。

`backtesting[8]` : `backtesting` は過去の株価データから取引戦略の実行可能性を推測するためのライブラリである。エコシステムライブラリ (Pandas, Numpy など) の上に構築されているため使いやすく高速に実行可能である。

図 2.4 の `Bactesting` に含まれている `self.I()` メソッドについて説明する。`self.I()` メソッドはインディケータを作成するメソッドで、指定されたインディケータ (ここでは *SMA*) を呼び出し、その結果を取得する。*SMA* は単純移動平均を計算する関数である。`self.data` はデータフレームで、Date: 日付, Open: 始値, High: 高値, Low: 安値, Close: 終値, Volume: 出来高を格納している。ここでは終値の `self.data["Close"]` を用いている。`self.n` は移動平均の期間を指定するパラメータである。つまりこのメソッドの引数には、*n* 期間の終値の単純移動平均がか与えられている。

図 2.5 の `crossover` メソッドについて説明する。このメソッドは交差するかどうかを判定する関数である。この場合、`self.b_lower` が、`self.data["Close"]` より下から上に交差した場合、`True` を返す関数となっている。

図 2.6 について説明する。`class bb` は `Bactesting` の `Strategy` クラスを継承して、独自の売買戦略を定義している。パラメータや変数は提案するアルゴリズムによって定義する。ここでは *n*, *n1*, *n2*, *n3*, *nb*, *upper_sigma*, *lower_sigma*, *slinit*, *tpinit* を定義している。

`def init(self)` は、初期化メソッドで、図 2.4 と同じように *MACD*, *BB* を取得し、`self.macd`, `self.macdsignal`, `self.b_upper`, `self.b_lower` に格納している。また、損切りのパラメータの *slinit* と利確のパラメータの *tpinit* を初期化している。

`def next(self)` メソッドは、時間単位で呼び出され、条件で売買を行う。ここでは図 2.5 で説明した `crossover` メソッドを用いて真偽値を出している。真である場合、`self.buy` メソッドを呼び出して買い注文をしている。*slinit* と *tpinit* をもちいて売却の条件も設定している。

```
self.I(SMA, self.data["Close"], self.n)
```

図 2.4 インディケーターメソッド, SMA, self.data

```
(crossover(self.b_lower, self.data["Close"]))
```

図 2.5 クロスオーバーメソッド

```
class bb(Strategy): # 独自の売買戦略
    n = 60 # 移動平均線日数
    n1 = 5 # 短期日数
    n2 = 20 # 長期日数
    n3 = 9 # シグナル日数
    nb=25
    upper_sigma = 2 # ボリンジャーバンド+2σ
    lower_sigma = 2 # ボリンジャーバンド-2σ
    slinit = 95 #利確
    tpinit =105 #損切り
    def init(self):
        close_prices = self.data["Close"].astype(float)
        self.macd, self.macdsignal = self.I(MACD, close_prices, 10, self.n2, self.n3) # MACD
        self.b_upper, self.b_lower = self.I(BB, close_prices, self.nb, self.upper_sigma, self.lower_sigma) # BB
        self.slinit=self.slinit
        self.tpinit=self.tpinit

    def next(self):
        if crossover(self.b_lower, self.data["Close"]): # -2σ > 終値 で買う
            price = self.data["Close"]
            self.buy(
                sl=price * (0.01*self.slinit),
                tp=price * (0.01*self.tpinit)
            )
```

図 2.6 売買戦略

図 2.7 について説明する．*s_code* は銘柄コードを格納しており，*df* は独自の *get_stock_data_from_file()* から取得した株価データを格納している．*data* は取得した株価データの 2013 年 1 月 1 日から 2022 年 12 月 31 日までの株価データを保持している．

bt は Bactesting の Backtest クラスを用いてバックテストを初期化する．ここで使用しているのは先ほど説明した *data*，図 2.6 で説明した戦略と取引をクローズ時に行うかどうかの設定の *trade_on_close* で，True の場合はクローズ時に取引が行われる．

result は *optimize* メソッドで *bt* を最適化している．ここで最適化するパラメータは *slinit* と *tpinit* で，*slinit* は 90 から 100 まで 1 単位，*tpinit* は 101 から 110 まで 1 単位で試行する．このとき *Return*[%]，つまり利益率を最大化するように設定する．

```
s_code=6920
df = get_stock_data_from_file(s_code)
data = df[dt.datetime(2013,1,1):dt.datetime(2022,12,31)]
bt = Backtest(data, bb, trade_on_close=True)
result=bt.optimize(slimit = range(90, 100, 1), tpinit = range(101, 111, 1), maximize = "Return [%]")
```

図 2.7 バックテスト

図 2.8 に付いて説明する．この図はバックテストを行って最適化実行時の結果である．`pandas.DataFrame` 型のオブジェクトである．ここでは本研究で使用する実行結果のデータは，以下の通りである．

Return[%] : 10 年間の利益率

Trade : 10 年間の取引数

Win Rate[%] : 10 年間の勝率

datetime : `datetime` は日時に関するデータを操作するためのクラスを提供している [7]．図 2.9 と図 2.10 に簡単な使用例と実行結果を示す．`datetime.now()` メソッドを使って，現在の日付と時刻と取得し，`current_datetime` に格納している．そのあと `print()` 関数で結果を出力する．実行結果は図 2.10 のようになる．

pandas : `pandas` はデータを簡単かつ直感的に扱えるように設計されたデータ構造を提供している [11]．図 2.11 と図 2.12 に簡単な使用例と実行結果を示す．辞書型 (dictionary) としてデータを定義する．キーとして 'Name', 'Age', 'City' があり，それぞれのキーに対応する値がリストとして格納されている．辞書型のデータを用いて `Pandas` の `DataFrame` を作成する．`DataFrame` は表形式のデータ構造で，ここでは 'Name', 'Age', 'City' の列を持つ 3 つの行が作成される．そのあと `print()` 関数で結果を出力する．実行結果は図 2.12 のようになる．

numpy : `numPy` は，`Python` を使用した科学技術コンピューティング用のライブラリである [10]．図 2.13 と図 2.14 に簡単な使用例と実行結果を示す．`NumPy` の `array` 関数を用いて与えられたリスト `[[1, 2, 3], [4, 5, 6]]` より 2 次元の `NumPy` 配列を作成する．そのあと `print()` 関数で結果を出力する．実行結果は図 2.14 のようになる．

```

Start                2004-08-19 00:00:00
End                  2013-03-01 00:00:00
Duration              3116 days 00:00:00
Exposure Time [%]    93.9944
Equity Final [$]     51959.9
Equity Peak [$]      75787.4
Return [%]           419.599
Buy & Hold Return [%] 703.458
Return (Ann.) [%]    21.328
Volatility (Ann.) [%] 36.5383
Sharpe Ratio         0.583718
Sortino Ratio        1.09239
Calmar Ratio         0.444518
Max. Drawdown [%]    -47.9801
Avg. Drawdown [%]    -5.92585
Max. Drawdown Duration 584 days 00:00:00
Avg. Drawdown Duration 41 days 00:00:00
# Trades              65
Win Rate [%]         46.1538
Best Trade [%]        53.596
Worst Trade [%]       -18.3989
Avg. Trade [%]        2.35371
Max. Trade Duration   183 days 00:00:00
Avg. Trade Duration   46 days 00:00:00
Profit Factor         2.08802
Expectancy [%]        8.79171
SQN                   0.916893
_strategy             SmaCross
_equity_curve          Eq...
_trades                Size EntryB...
dtype: object

```

図 2.8 バックテスト実行時の結果

```
from datetime import datetime

# 現在の日付と時刻を取得
current_datetime = datetime.now()
print("現在の日付と時刻:", current_datetime)
```

図 2.9 datetime の使用例

```
現在の日付と時刻: 2024-02-08 04:28:34.689703
```

図 2.10 図 2.9 実行時の結果

```
import pandas as pd
data = {'Name': ['Alice', 'Bob', 'Charlie'],
        'Age': [25, 30, 35],
        'City': ['New York', 'San Francisco', 'Los Angeles']}

df = pd.DataFrame(data)
print(df)
```

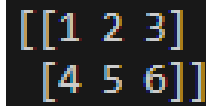
図 2.11 pandas の使用例

	Name	Age	City
0	Alice	25	New York
1	Bob	30	San Francisco
2	Charlie	35	Los Angeles

図 2.12 図 2.11 実行時の結果

```
arr_2d = np.array([[1, 2, 3], [4, 5, 6]])
print(arr_2d)
```

図 2.13 numpy の使用例



```
[[1 2 3]
 [4 5 6]]
```

図 2.14 図 2.13 実行時の結果

talib : talib は金融取引で使われている指標のライブラリを提供している [5] . 図 2.15 と図 2.16 に簡単な使用例と実行結果を示す . NumPy の random 関数を用いて , 0 から 1 までのランダムな数値を 100 個生成し , それぞれに 100 を掛けて仮想の株価データを制作する . Ta-Lib の SMA 関数を用いて , 与えられた株価データに対して 20 期間の単純移動平均を計算する . そのあと `print()` 関数で結果を出力する . 実行結果は図 2.15 のようになる .

matplotlib : matplotlib はアニメーションやグラフを作成するために使われるライブラリである [9] . 図 2.17 と図 2.18 に簡単な使用例と実行結果を示す . NumPy の linspace 関数を用いて , 0 から 10 までの範囲を 100 点で均等に区切ったデータを生成する . NumPy の sin 関数を用いて , 各 x に対する $\sin(x)$ の値を計算する . `plt.plot()` メソッドで折れ線グラフをプロットする . `plt.title()` メソッドでグラフにタイトルを追加する . `plt.xlabel()` メソッドと `plt.ylabel()` メソッドで x 軸と y 軸にラベルを追加する . `plt.legend()` メソッドでグラフにを表示する . `plt.show()` メソッドで , 作成したグラフを表示する . 実行結果は図 2.18 のようになる .

```
import talib
import numpy as np

# 価格データの作成
close_prices = np.random.random(100) * 100 # 仮想の株価データ

# SMA (単純移動平均) の計算
sma = talib.SMA(close_prices, timeperiod=20)

print("SMA:", sma)
```

図 2.15 talib の使用例

```

SMA: [      nan      nan      nan      nan      nan      nan      nan      nan
      nan      nan      nan      nan      nan      nan      nan      nan
      nan 46.89872159 48.16766702 45.62484269 43.69898179 47.17230934
48.40436552 50.12107499 49.73598585 51.28702272 49.48072183 49.24436691
47.67165122 48.88673728 52.66988728 53.92105318 50.82720274 52.04059869
53.12500838 49.66047036 49.34998327 49.58800331 52.4977614 52.03087272
53.86558386 53.68307857 52.3631216 48.79958457 50.80628082 51.43172527
52.68086539 53.33914497 53.06482635 53.2489389 49.58504579 50.14858376
51.5944838 48.93416877 48.44030438 51.56804295 52.53015446 55.32341548
53.83861901 54.69936405 52.17097431 52.14963561 50.96886874 51.49495531
48.17918774 46.10446898 44.50855899 46.81610176 47.23412558 46.74600997
46.89358976 47.24949349 47.77390353 50.51176193 51.74744412 48.7413544
48.0307519 47.2468511 46.60627312 45.41489241 46.54394607 43.18694342
45.77338541 47.98765526 49.6388623 51.13498361 53.15232626 53.9309145
56.13453811 57.32919816 60.17042561 60.52666967 58.34635016 55.4473217
53.52535791 54.66360339 55.12701062 56.34709566]

```

図 2.16 図 2.15 実行時の結果

```

import matplotlib.pyplot as plt
import numpy as np

# データの生成
x = np.linspace(0, 10, 100)
y = np.sin(x)

# 折れ線グラフのプロット
plt.plot(x, y, label='sin(x)')

# グラフにタイトルと軸ラベルを追加
plt.title('Sin Function')
plt.xlabel('x')
plt.ylabel('sin(x)')

# 凡例を表示
plt.legend()

# グラフを表示
plt.show()

```

図 2.17 matplotlib の使用例

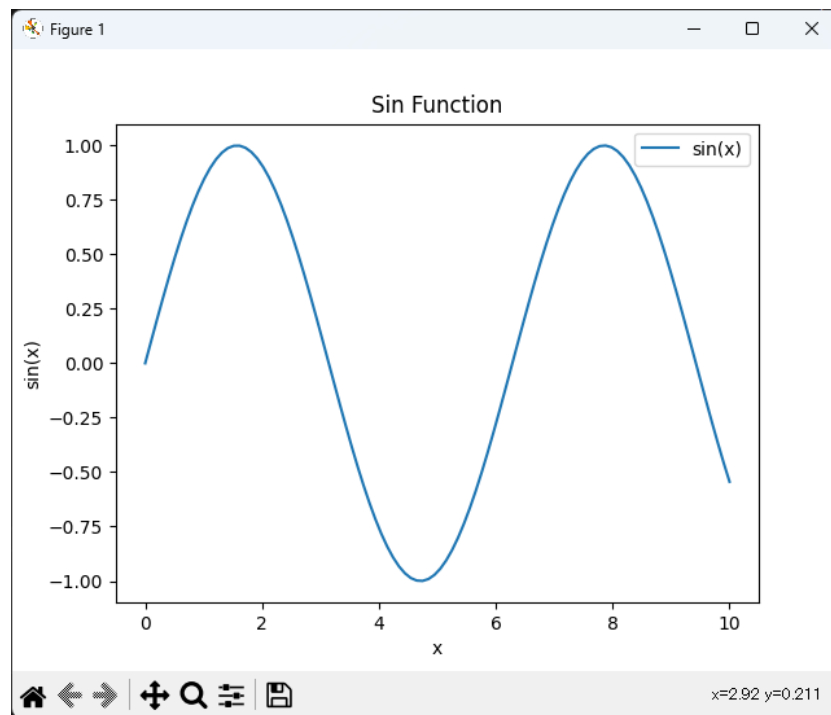


図 2.18 図 2.17 実行時の結果

第3章 トレーディングアルゴリズム

本章では、実際に使うトレーディングアルゴリズムを順に説明する。

3.1 MACD

ここでは MACD を主に使うトレーディングアルゴリズムの説明をする。利確時の変数 tp および損切り時の変数 sl はアルゴリズム実行時、変更可能なパラメータである。

3.1.1 MACD のみ

MACD のみを使う場合のトレーディングアルゴリズムの説明をする。ここでは (B1) とする。

Step 1 : MACD がシグナルを下から上へ突き抜けたときに翌日の始値で株を購入する。

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する。

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る。

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る。

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す。

図 3.1 に MACD による買いシグナルの例を示す。図 3.1 の黒丸が MACD の買いシグナルで、その後 tp % 以上で売っている。

3.1.2 MACD と 60 日移動平均線

MACD と 60 日移動平均線を使う場合のトレーディングアルゴリズムを説明する。長期の移動平均線の傾きが正の場合、株価も上昇する傾向にあるため、MACD と組み合わせることで取引の勝率が上がると考えられ、それにより最終利益が増加すると想定している。このアルゴリズムでは、60 日平均線を長期線とする。ここでは (B2) とする。



図 3.1 (B1) アルゴリズムの買いシグナル

Step 1 : 以下の (1-1) ~ (1-2) のすべての条件が満たされれば翌日購入するする .

(1-1): MACD がシグナルを下から上へ突き抜けたとき .

(1-2): 長期線の傾きが 0 以上のとき .

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する .

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る .

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る .

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す .

図 3.2 に MACD と長期線による買いシグナルの例を示す . 図 3.2 の黒丸が MACD の買いシグナルで , その時の長期線の傾きも 0 以上なので購入している . その後 tp % 以上で売っている .



図 3.2 (B2) アルゴリズムの買いシグナル

3.1.3 MACD と日経株価平均

MACD と日経平均株価を使う場合のトレーディングアルゴリズムを説明する．日経平均株価は日本の主要な 225 銘柄を対象としており，日経平均株価が上昇しているということは市場全体が安定していると考えられ，投資家の信頼があることを示唆している．つまり東証上位 100 社にもこの安定性が影響を及ぼすと考えられ，それにより最終利益が増加すると想定している．ここでは (B3) とする．

Step 1：以下の (1-1) ~ (1-2) のすべての条件が満たされれば翌日購入するする．

(1-1): 注目する日の MACD がシグナルを下から上へ突き抜けたとき．

(1-2): 注目する日の日経平均株価の長期線の傾きが 0 以上のとき．

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する .

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る .

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る .

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す .

図 3.3 に MACD と日経株価平均による買いシグナルの例を示す . 図 3.3 の黒丸が MACD の買いシグナルで , その時の日経株価平均の傾きも 0 以上なので購入している . その後 tp % 以上で売っている .

3.1.4 MACD と日経株価平均と長期線

MACD と日経平均株価と取引している株価データの長期線を使う場合のトレーディングアルゴリズムを説明する . 日経平均株価と取引している株価データの長期線を組み合わせることで , より勝率が上がると考えられ , (B2) , (B3) よりも最終利益が増加すると想定している . ここでは二通りの方法を説明する . 最初に説明するアルゴリズムを (B4) , その後に説明するアルゴリズムを (B5) とする .

(B4):

Step 1 : 以下の (1-1) ~ (1-3) のすべての条件が満たされれば翌日購入する .

(1-1): 注目する日の MACD がシグナルを下から上へ突き抜けたとき .

(1-2): 注目する日の日経平均株価の長期線の傾きが 0 以上のとき .

(1-3): 取引している株価データの長期線の傾きが 0 以上のとき .

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する .

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る .

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る .

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す .

図 3.4 に MACD , 長期線 , 日経株価平均による買いシグナルの例を示す . 図 3.4 の黒丸が MACD の買いシグナルで , その上に表示されているその時の長期線の傾きと更に上の日経株価平均の傾きも 0 以上なので購入している . その後 tp % 以上で売っている .

(B5):



図 3.3 (B3) アルゴリズムの買いシグナル

Step 1 : 以下の (1-1) の条件と (1-2) , (1-3) のどちらかの条件が満たされれば翌日購入するする .

(1-1): 注目する日の MACD がシグナルを下から上へ突き抜けたとき .

(1-2): 注目する日の日経平均株価の長期線の傾きが 0 以上のとき .

(1-3): 取引している株価データの長期線の傾きが 0 以上のとき .

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する .

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る .

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る .



図 3.4 (B4) アルゴリズムの買いシグナル

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す .

図 3.5 , 図 3.6 に (B5) アルゴリズムの買いシグナルの例を示す . 図 3.5 の黒丸が MACD の買いシグナルで , その時の赤丸の長期線の傾きは 0 以下だが青丸の日経株価平均の傾きが 0 以上なので購入している . その後 tp % 以上で売っている . また図 3.6 の黒丸が MACD の買いシグナルで , その時の青丸の日経株価平均の傾きは 0 以下だが赤丸の長期線の傾きが 0 以上なので購入している . その後 tp % 以上で売っている .

3.2 BB

ここでは BB を主に使うトレーディングアルゴリズムの説明をする . 利確時の変数 tp および損切り時の変数 sl はアルゴリズム実行時 , 変更可能なパラメータである .



図 3.5 (B5) アルゴリズムの (1-1) と (1-2) の買いシグナル

3.2.1 BB のみ

BB のみを使う場合のトレーディングアルゴリズムの説明をする．ここでは (B6) とする．

Step 1 : 注目する日のローソクの終値が -2 の BB を上から下へ突き抜けたときに終値で株を購入する．

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する．

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る．

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る．

Step 3 : 用意した株価データの最終日に来るまで Step 1 から Step 3 まで繰り返す．



図 3.6 (B5) アルゴリズムの (1-1) と (1-3) の買いシグナル

図 3.7 に BB による買いシグナルの例を示す．図 3.7 の黒丸が BB の買いシグナルで，その後 tp % 以上で売っている．

3.2.2 BB と 60 移動平均線

BB と移動平均線を使う場合のトレーディングアルゴリズムの説明をする．60 移動平均線を組み合わせる理由として (B2) と同じように，取引の勝率が上がると考えられ，それにより最終利益が増加すると想定している．ここでは (B7) とする．

Step 1：以下の (1-1) ~ (1-2) のすべての条件が満たされれば終値で購入する．

(1-1): 注目する日のローソクの終値が -2 の BB を上から下へ突き抜けたときに終値で株を購入する．

(1-2): 長期線の傾きが 0 以上のとき．



図 3.7 (B6) アルゴリズムの買いシグナル

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する .

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る .

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る .

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す .

図 3.8 に BB と長期線による買いシグナルの例を示す . 図 3.8 の黒丸が BB の買いシグナルで , その時の赤丸の長期線の傾きは 0 以上なので購入している . その後 tp % 以上で売っている .

3.2.3 BB と日経平均株価

BB と日経平均株価を使う場合のトレーディングアルゴリズムの説明をする . 日経平均株価を組み合わせる理由として (B3) と同じように , 取引の勝率が上がると考えられ , それにより最終利益が増加すると想定している . ここでは (B8) とする .

Step 1 : 以下の (1-1) ~ (1-2) のすべての条件が満たされれば翌日購入する .

(1-1): 注目する日のローソクの終値が -2 の BB を上から下へ突き抜けたときに終値で株を購入する .



図 3.8 (B7) アルゴリズムの買いシグナル

(1-2): 注目する日の日経平均株価の長期線の傾きが 0 以上のとき .

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する .

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る .

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る .

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す .

図 3.9 に BB と日経株価平均による買いシグナルの例を示す . 図 3.9 の黒丸が BB の買いシグナルで , その時の赤丸の日経株価平均の傾きは 0 以上なので購入している . その後 tp % 以上で売っている .

3.2.4 BB と日経株価平均と長期線

BB と日経平均株価と取引している株価データの長期線を使う場合のトレーディングアルゴリズムを説明する . (B4) , (B5) 同じように , 取引の勝率が上がると考えられ , それにより最終利益が増加すると想定している . ここでは二通りの方法を



図 3.9 (B8) アルゴリズムの買いシグナル

説明する．最初に説明するアルゴリズムを (B9) , その後に説明するアルゴリズムを (B10) とする．

(B9):

Step 1 : 以下の (1-1) ~ (1-3) のすべての条件が満たされれば翌日購入するする．

(1-1): 注目する日のローソクの終値が-2 の BB を上から下へ突き抜けたときに終値で株を購入する．

(1-2): 注目する日の日経平均株価の長期線の傾きが0 以上のとき．

(1-3): 取引している株価データの長期線の傾きが0 以上のとき．

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する．

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る．

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る．

Step 3 : 用意した株価データの最終日に来るまで Step 1 から Step 3 まで繰り返す．

図 3.10 に BB，長期線，日経株価平均による買いシグナルの例を示す．図 3.10 の黒丸が BB の買いシグナルで，その時の長期線の傾きと日経株価平均の傾きも 0 以上なので購入している．その後 tp % 以上で売っている．

(B10):

Step 1：以下の (1-1) の条件と (1-2)，(1-3) のどちらかの条件が満たされれば終値で購入する．

c: 注目する日のローソクの終値が -2 の BB を上から下へ突き抜けたときに終値で株を購入する．

(1-2): 注目する日の日経平均株価の長期線の傾きが 0 以上のとき．

(1-3): 取引している株価データの長期線の傾きが 0 以上のとき．

Step 2：以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する．

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る．

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る．

Step 3：用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す．

図 3.11，及び，図 3.12 に，(B10) アルゴリズムによる買いシグナルの例を示す．図 3.11 の黒丸が BB の買いシグナルで，その時の赤丸の長期線の傾きは 0 以下だが青丸の日経株価平均の傾きが 0 以上なので購入している．その後 tp % 以上で売っている．また図 3.12 の黒丸が BB の買いシグナルで，その時の青丸の日経株価平均の傾きは 0 以下だが赤丸の長期線の傾きが 0 以上なので購入している．その後 tp % 以上で売っている．

3.3 MACD と BB

ここでは MACD と BB を主に使うトレーディングアルゴリズムの説明をする．利確時の変数 tp および損切り時の変数 sl はアルゴリズム実行時，変更可能なパラメータである．

3.3.1 MACD と BB

MACD と BB を使う場合のトレーディングアルゴリズムを説明する．2 つを組み合わせることでより良い効果が得られるのではないかと想定している．ここでは (B11) とする．



図 3.10 (B9) アルゴリズムの買いシグナル

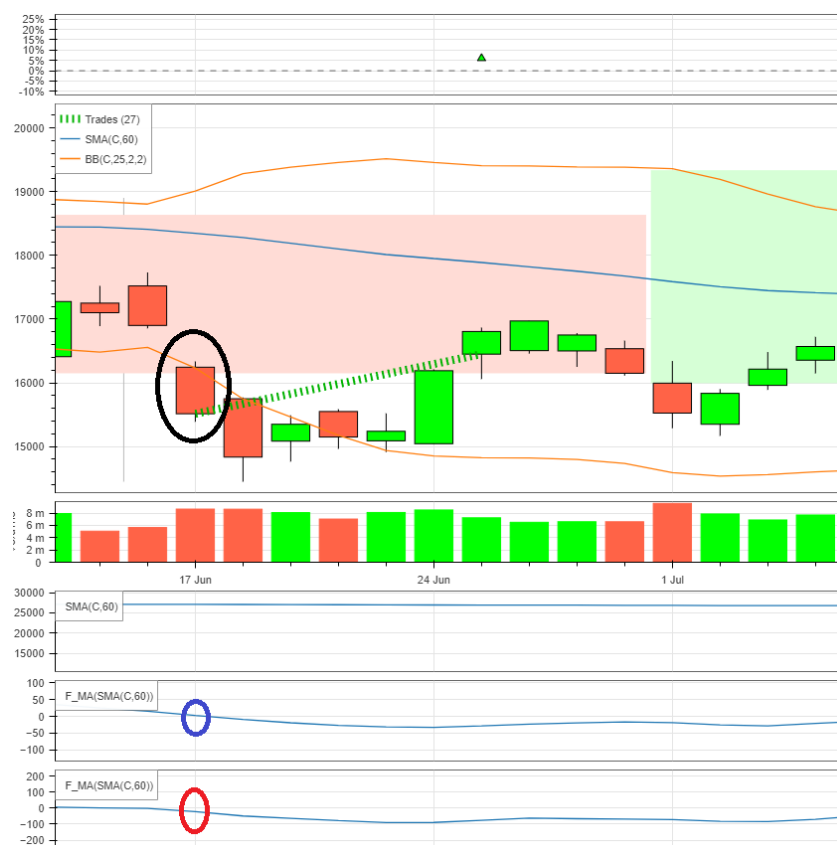


図 3.11 (B10) アルゴリズムの (1-1) と (1-2) の買いシグナル

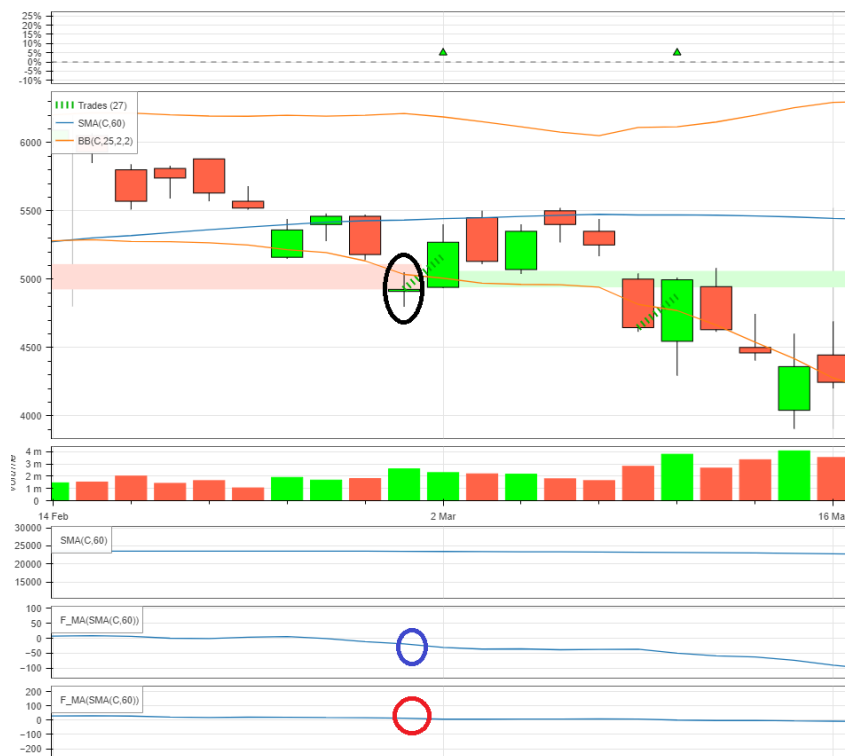


図 3.12 (B10) アルゴリズムの (1-1) と (1-3) の買いシグナル

Step 1 : 以下の (1-1) ~ (1-2) のどちらかの条件が満たされれば購入する .

(1-1): MACD がシグナルを下から上へ突き抜けたときに翌日の始値で株を購入する .

(1-2): 注目する日のローソクの終値が -2 の BB を上から下へ突き抜けたときに終値で株を購入する .

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する .

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る .

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る .

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す .

図 3.13 , 図 3.14 に MACD と B よる買いシグナルの例を示す . 図 3.13 の黒丸の BB はなにも起きていないが , 赤丸が MACD の買いシグナルになっているので購入している . その後 tp % 以上で売っている . また , 図 3.14 の赤丸の MACD はなにも起きていないが , 黒丸が BB の買いシグナルになっているので購入している . その後 tp % 以上で売っている .



図 3.13 (B11) アルゴリズムの (1-2) の買いシグナル

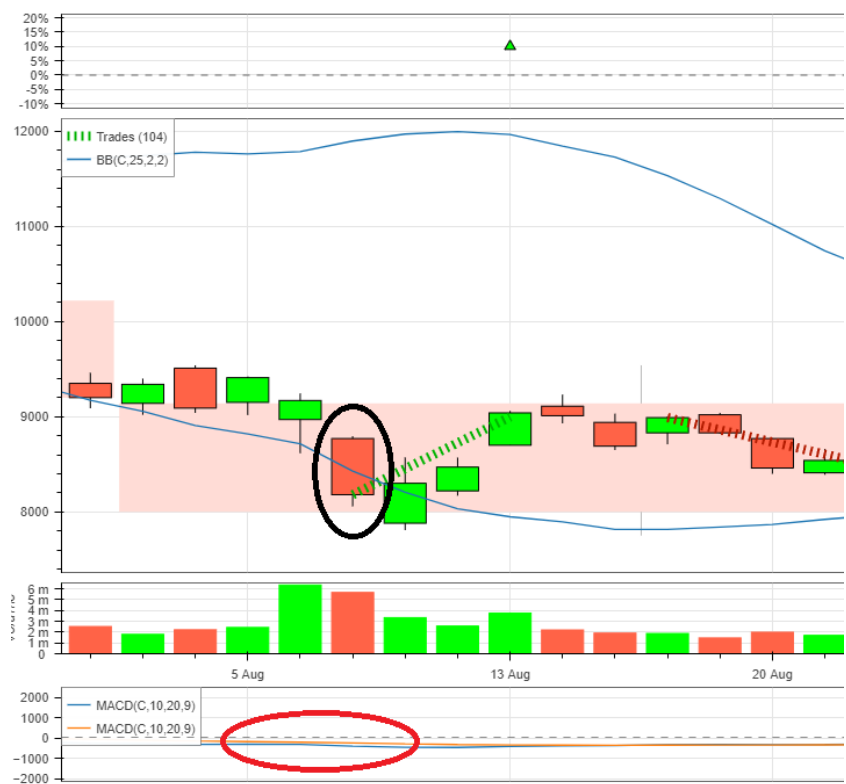


図 3.14 (B11) アルゴリズムの (1-1) の買いシグナル

3.3.2 MACD と BB と 60 日移動平均線

MACD と BB と長期線を使う場合のトレーディングアルゴリズムを説明する．ここでは二通りの方法を説明する．(B11) に移動平均線を加えることで安定して勝率が上がるのではないかと想定している．ここでは最初に説明するアルゴリズムを (B12) ，その後に説明するアルゴリズムを (B13) とする．

(B12):

Step 1 : 以下の (1-1) ~ (1-2) のどちらかと (1-3) の条件が満たされれば翌日購入するする．

(1-1): 注目する日の MACD がシグナルを下から上へ突き抜けたとき．

(1-2): 注目する日のローソクの終値が -2 の BB を上から下へ突き抜けたときに翌日の始値で株を購入する．

(1-3): 取引している株価データの長期線の傾きが 0 以上のとき．

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する．

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る．

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る．

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す．

図 3.15 , 図 3.16 に B12 アルゴリズムの買いシグナルの例を示す．図 3.15 の青丸の BB はなにも起きていないが，黒丸が MACD の買いシグナルになっているのと，赤丸の長期線の傾きも 0 以上なので購入している．その後 tp % 以上で売っている．また，図 3.16 の青丸の MACD はなにも起きていないが，黒丸が BB の買いシグナルになっているのと，赤丸の長期線の傾きも 0 以上なので購入している．その後 tp % 以上で売っている．

(B13):

Step 1 : 以下の (1-1) の条件または (1-2) , (1-3) のすべての条件が満たされれば翌日購入する．

(1-1): 注目する日の MACD がシグナルを下から上へ突き抜けたとき．

(1-2): 注目する日のローソクの終値が -2 の BB を上から下へ突き抜けたときに翌日の始値で株を購入する．

(1-3): 取引している株価データの長期線の傾きが 0 以上のとき．



図 3.15 (B12) アルゴリズムの (1-1) と (1-3) の買いシグナル



図 3.16 (B12) アルゴリズムの (1-2) と (1-3) の買いシグナル

Step 2 : 以下の (2-1) ~ (2-2) のどちらかの条件が満たされれば売却する .

(2-1): 株の始値が購入した株価より tp % 以上になった時点で売る .

(2-2): 株の始値が購入した株価より sl % 以下になった時点で売る .

Step 3 : 用意した株価データの最終日が来るまで Step 1 から Step 3 まで繰り返す .

図 3.17 , 図 3.18 に (B13) の買いシグナルの例を示す . 図 3.17 の黒丸が MACD の買いシグナルになっているのと , 赤丸の長期線の傾きも 0 以上だが青丸の BB はなにも起きていたため , MACD の買いシグナルのみで購入している . その後 tp % 以上

で売っている．また，図 3.18 の青丸の MACD はなにも起きていないが，黒丸が BB の買いシグナルになっているのと，赤丸の長期線の傾きも 0 以上なので購入している．その後 tp % 以上で売っている．

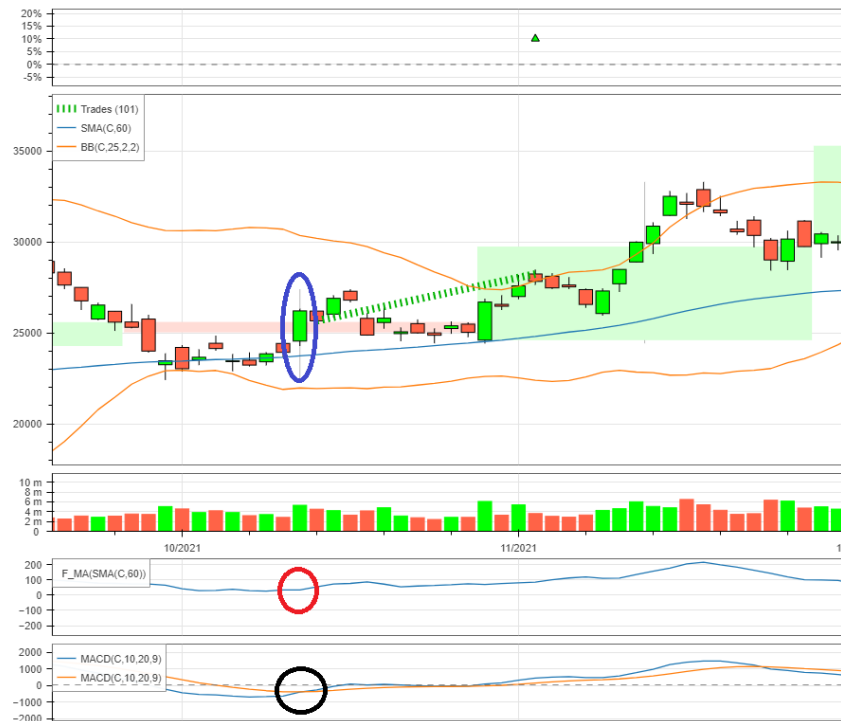


図 3.17 (B13) アルゴリズムの (1-1) の買いシグナル

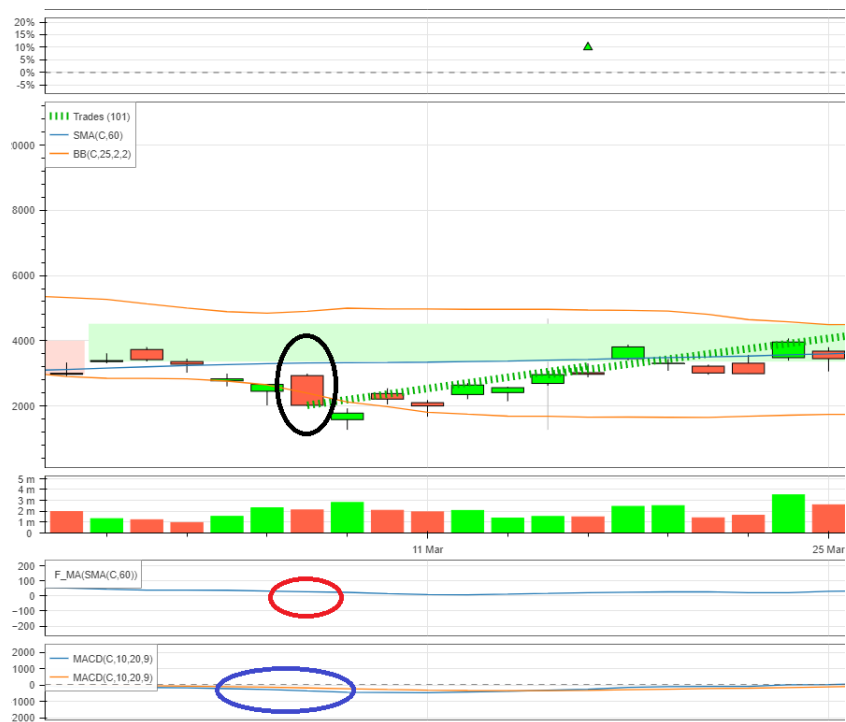


図 3.18 (B13) アルゴリズムの (1-2) と (1-3) の買いシグナル

第4章 実験結果

本章では，前章で説明したトレーディングアルゴリズムを使って上位東証上位100社のうち2013年からのデータがある95株の平均利益，平均勝率，最適化した平均 tp ， ls ，平均トレード数を評価する．

4.1 実験環境

本研究における実験環境を表4.1と4.2に表す．

表 4.1 実験環境

OS	Ubuntu 22.04.3 LTS
CPU	Intel Core i9-9900K
メモリ	128GB
言語	Python 3.10.12

表 4.2 ライブラリのバージョン

ライブラリ	バージョン
Backtesting	0.3.3
ta-lib-bin	0.4.26
numpy	1.26.0
pandas	1.5.3
matplotlib	3.8.0

4.2 上位 95 社の株価データ

検証で使った株価データのコードを表 4.3 に表す．上位東証上位 100 社のうち 2013 年からのデータがある 95 株を使用する．

表 4.3 株コード

株コード				
1605	1925	1928	2502	2503
2802	2914	3382	4063	4307
4452	4502	4503	4507	4519
4523	4543	4568	4578	4612
4661	4684	4689	4901	4911
5108	5401	6146	6201	6273
6301	6326	6367	6501	6502
6503	6594	6701	6702	6723
6752	6758	6762	6857	6861
6902	6920	6954	6971	6981
7011	7201	7203	7267	7269
7270	7532	7733	7741	7751
7832	7974	8001	8002	8015
8031	8035	8053	8058	8113
8267	8306	8308	8309	8316
8411	8591	8604	8630	8725
8750	8766	8801	8802	9020
9022	9101	9432	9433	9503
9613	9735	9843	9983	9984

これらの株価データの入手方法を以下に示す．

1. まずはじめに <https://finance.yahoo.com> に行く．
2. 次に検索バーに株コードを入れると候補欄が出てくるのでそこで選択する．
3. 出てきたページの Historical Data を選択する．
4. Time Period の項目があるので選択して株価データの使う期間を選択する．本論文では 2013 年 1 月 1 日から 2022 年 12 月 31 日のデータを使う．
5. Apply ボタンがあるので押してその下にある Download ボタンを押せば，株価データの csv ファイルを取得することができる．

4.3 トレーディングアルゴリズムの検証

第3章で説明した tp, sl を最適化してそれぞれのアルゴリズムを実行し、結果を以下の表 4.4 に表す。 tp, sl はそれぞれ 101 ~ 110 % , -99 ~ -90 % で最適化する。図 4.1 は、表 4.4 の平均利益、平均取引数、平均勝率をグラフ化したもので、図 4.2 は、表 4.4 の平均 TP、平均 SL をグラフ化したものである。

また、図 4.3 から図 4.15 までのグラフは、アルゴリズム (B1) から (B13) によって得られた 95 社の株の利益である。

表 4.4 各戦略の比較

戦略	平均利益	平均 TP	平均 SL	平均勝率%	平均取引数
(B1)	212.02144	107.95789	93.48421	53.80%	67.32
(B2)	95.86136	107.47368	93.34737	56.36%	35.80
(B3)	80.29327	107.63158	93.29474	56.22%	34.51
(B4)	63.71441	107.52632	93.46316	57.08%	24.82
(B5)	122.89338	107.37895	93.35789	56.44%	45.95
(B6)	125.71012	107.68421	93.00000	58.99%	40.18
(B7)	57.27753	106.90526	92.76842	64.45%	19.48
(B8)	51.61579	106.50526	92.62105	67.32%	19.21
(B9)	39.39328	106.83158	93.27368	66.21%	12.36
(B10)	69.63080	106.66316	92.47368	64.45%	26.14
(B11)	305.81290	107.41053	93.57895	53.82%	98.56
(B12)	123.13827	107.14737	93.65263	56.25%	50.98
(B13)	238.55610	107.80000	93.57895	53.21%	78.83

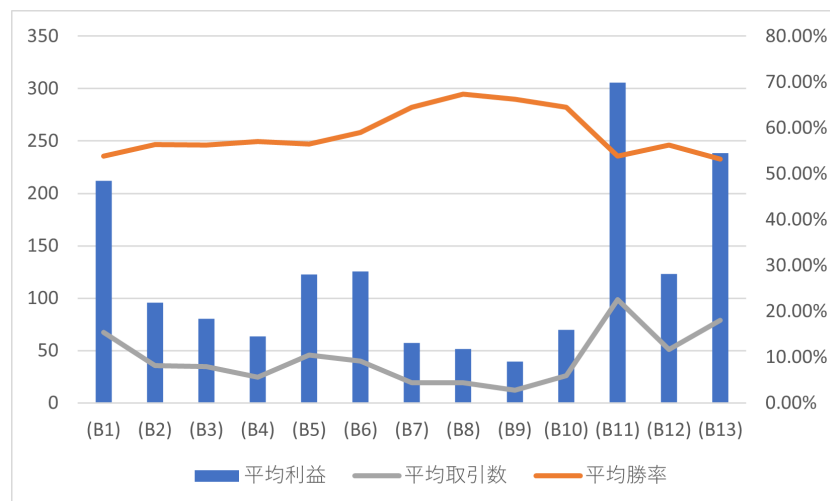


図 4.1 平均利益、平均取引数、平均勝率のグラフ

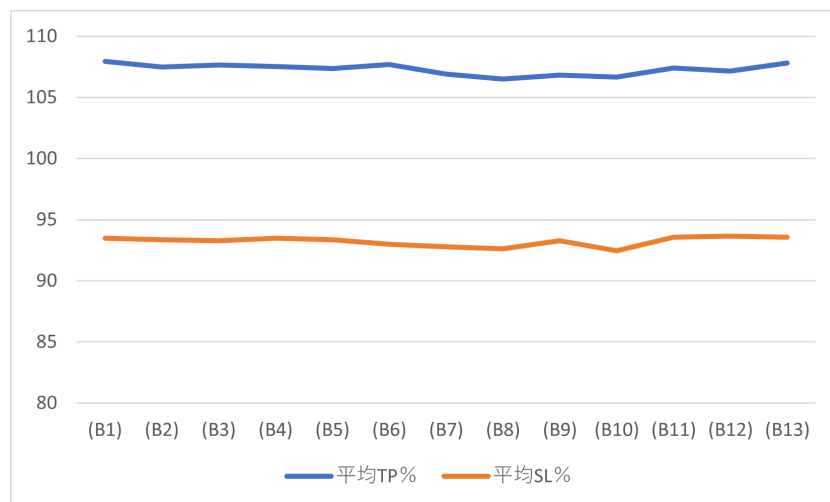


図 4.2 平均 SL , 平均 TP のグラフ

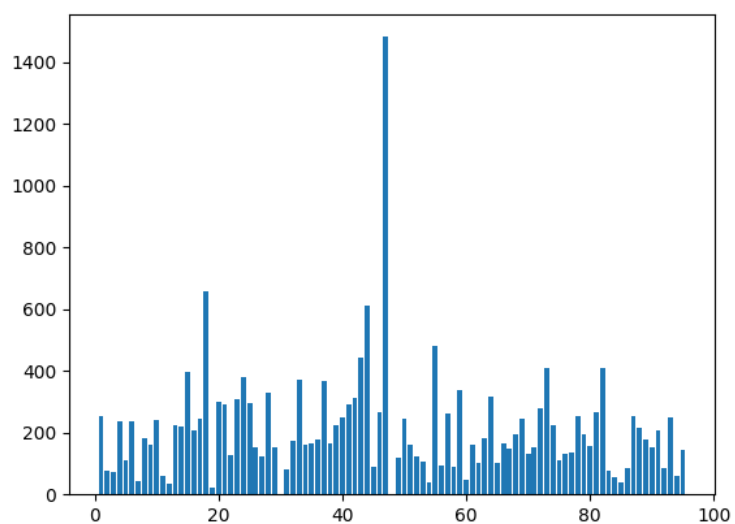


図 4.3 アルゴリズムによる利益 (B1)

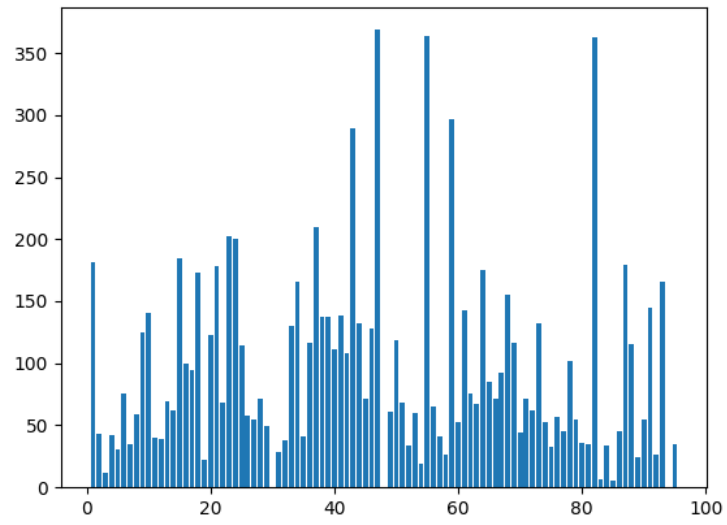


図 4.4 アルゴリズムによる利益 (B2)

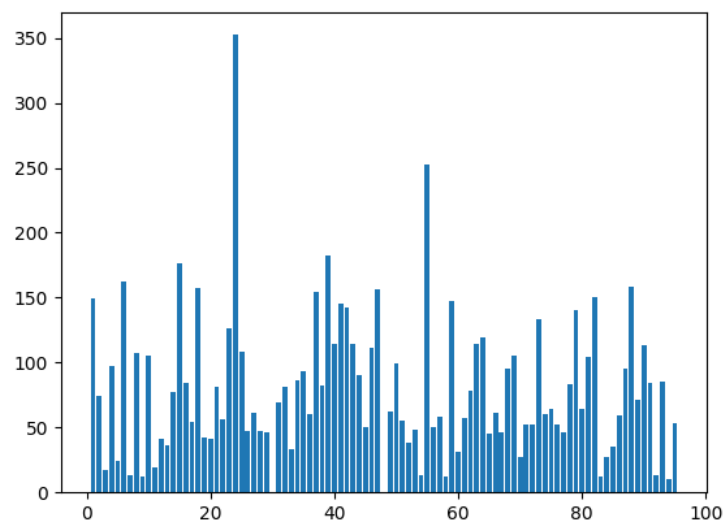


図 4.5 アルゴリズムによる利益 (B3)

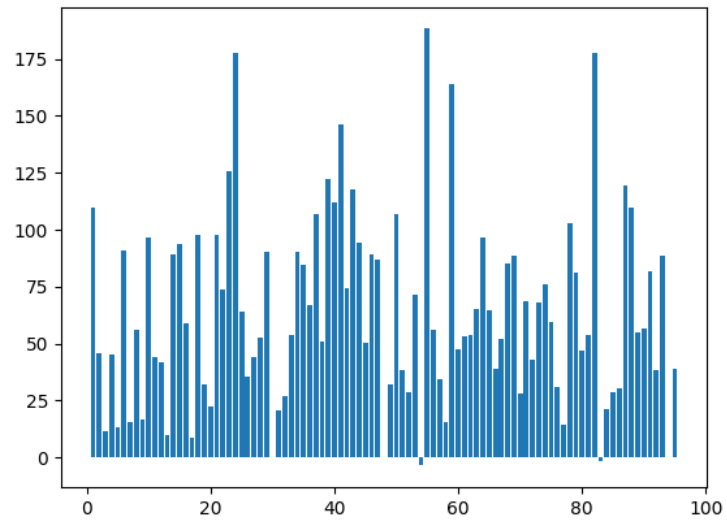


図 4.6 アルゴリズムによる利益 (B4)

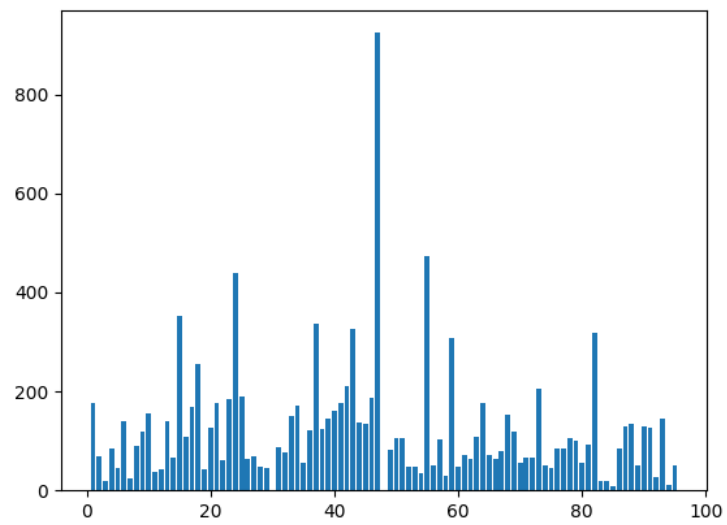


図 4.7 アルゴリズムによる利益 (B5)

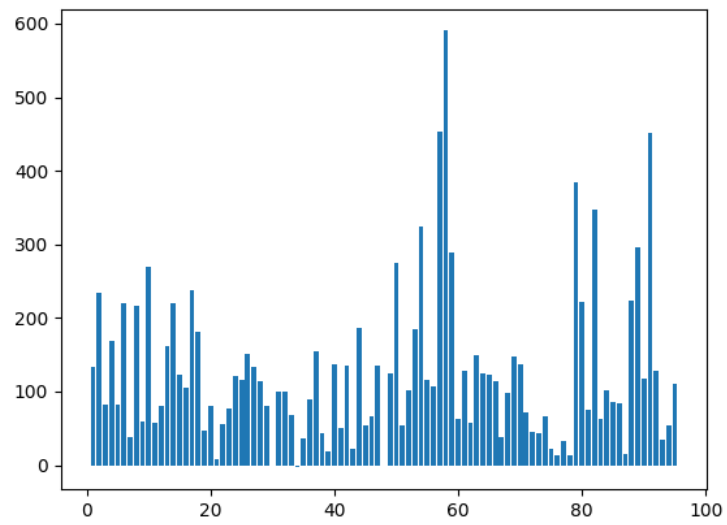


図 4.8 アルゴリズムによる利益 (B6)

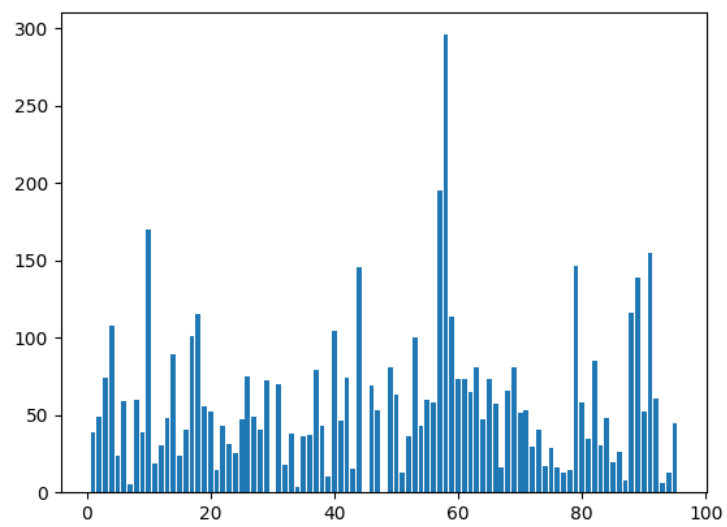


図 4.9 アルゴリズムによる利益 (B7)

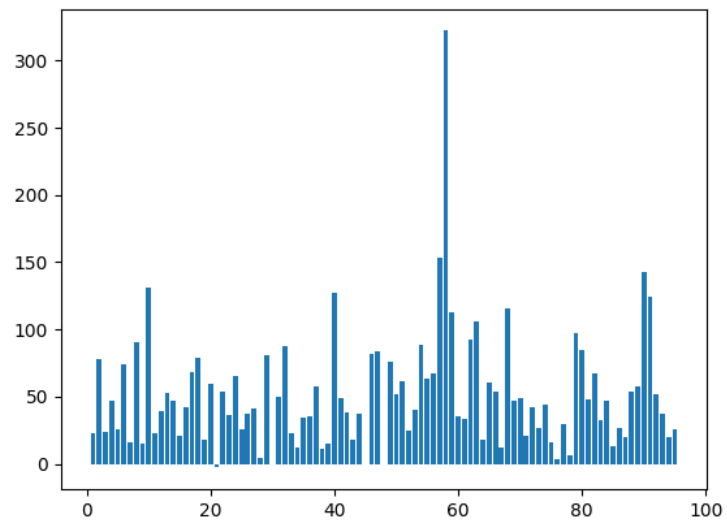


図 4.10 アルゴリズムによる利益 (B8)

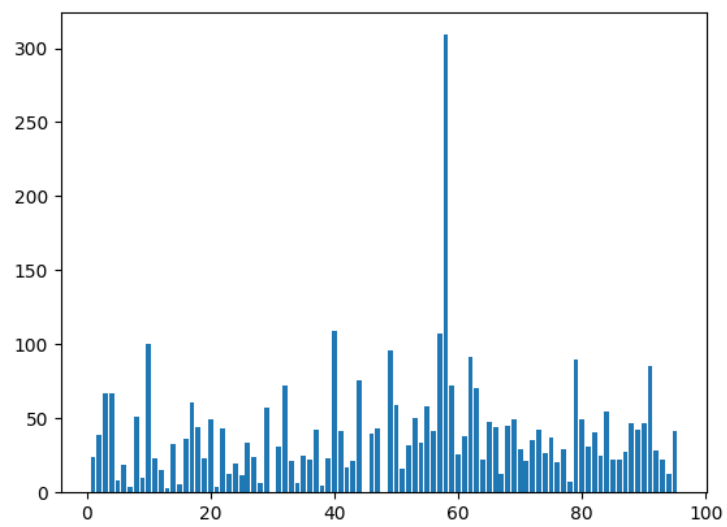


図 4.11 アルゴリズムによる利益 (B9)

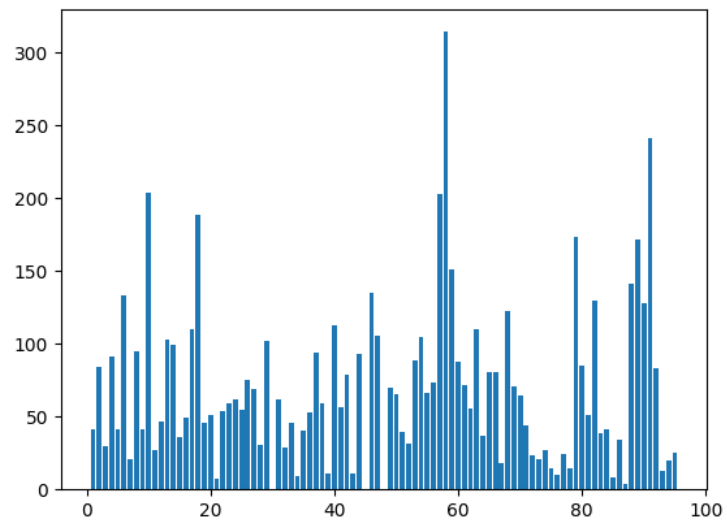


図 4.12 アルゴリズムによる利益 (B10)

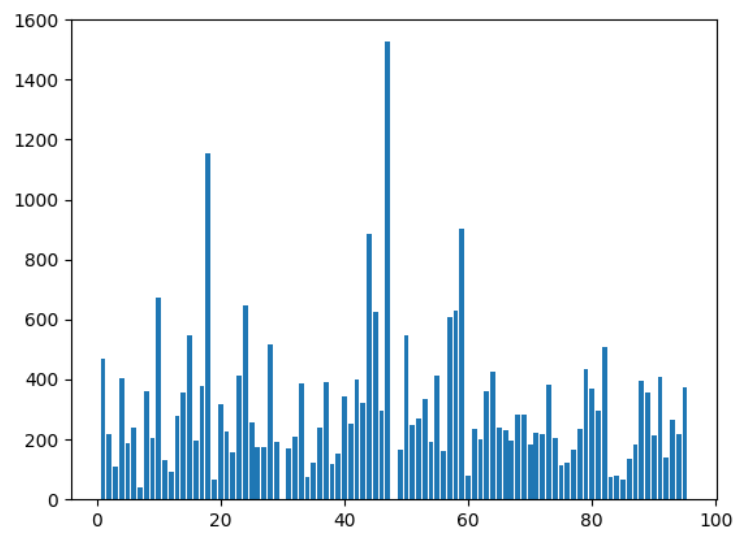


図 4.13 アルゴリズムによる利益 (B11)

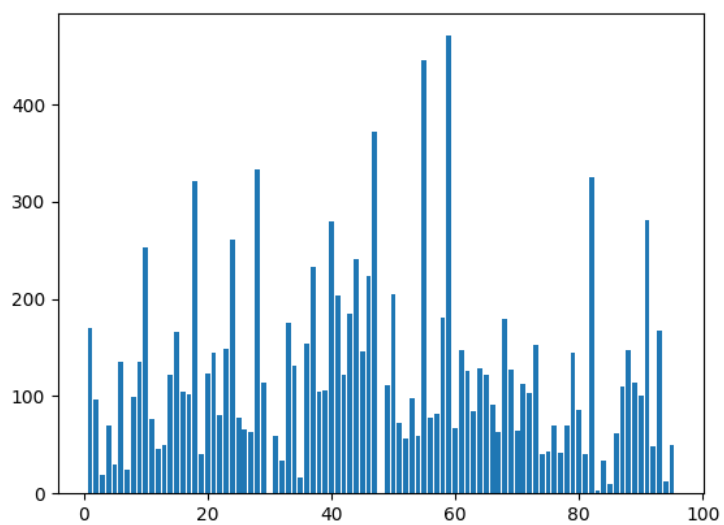


図 4.14 アルゴリズムによる利益 (B12)

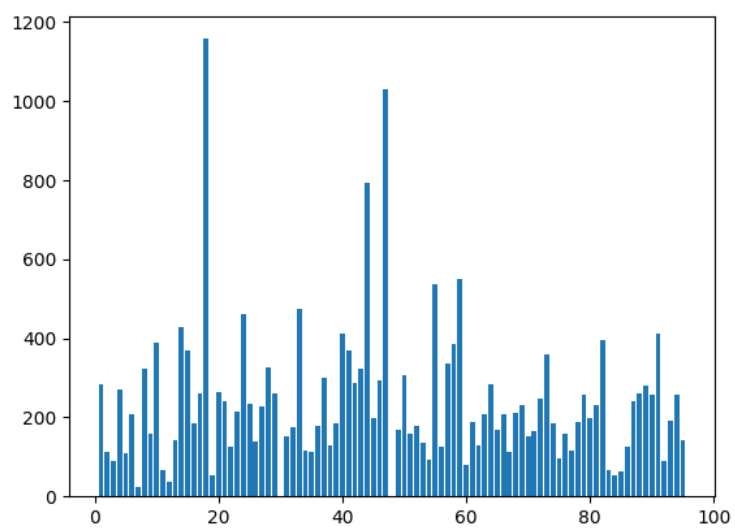


図 4.15 アルゴリズムによる利益 (B13)

4.4 実験結果と考察

表 4.4 より，利確に適している値は約 107 % で，損切りに適している値は約 93 % . 勝率は高く 67 % , 低くても 53 % であることがわかる . また , 利益が一番高いのは (B11) であることがわかる . これは既存のよく使われている 2 つ指標を組み合わせているためだと考えられる . BB と MACD それぞれ同じ条件の場合だと MACD の利益が高く BB の勝率が高い . (B9) の場合はすべてが同時になるという条件が厳しいため , 利益と取引が一番低い勝率が高くなりやすいことがわかる .

利益だけを追い求めるならば MACD 主体のトレーディングアルゴリズムか (B11) を選択すればいいが , リスクが少ない方法を選びのならば BB 主体のトレーディングアルゴリズム選択すればよいことがわかる .

第5章 まとめ

本研究では，MACD と BB を用いていくつかのトレーディングアルゴリズムの提案を行った．既存の MACD と BB それぞれ単体の場合よりも多い利益を出すことを示した．また利益重視で取引するならば MACD を主体とした組み合わせ，利益と取引数は下がるが勝率重視で取引するならば BB を主体とした組み合わせのトレーディングアルゴリズムが良いことがわかった．最大の利益を出したければ，利確を 107 %，損切りを 93 % にすれば良いという結果が得られた．

今後の課題として実際の取引でも使用されている他のテクニカル指標を用いたトレーディングアルゴリズムの提案と，AI を用いた場合の結果を比較することが挙げられる．

謝辞

本研究で知識不足な私に様々な助言を与えてくださった藤原暁宏教授に感謝します。また、ともに頑張ってきた同期のメンバーやわからないことなどにたいして助言を与えてくださった、研究室の先輩に感謝します。株などの経済の知識を教えてくれて自分に興味を持たせてくれた従兄に感謝します。最後に、何不自由なく大学で勉学に励めたのは、色々と助けてくれた両親のおかげです。心より感謝いたします。

参考文献

- [1] 森本叡多郎，長期投資におけるトレーディングアルゴリズムの検証と提案，九州工業大学卒業論文，2022
- [2] 片淵彼富，“Python ができる！株価データの分析”，森北出版株式会社，2023
- [3] Python Software Foundation，What is Python? Executive Summary，<https://www.python.org/doc/essays/blurb/>
- [4] Python Software Foundation，Applications for Python，<https://www.python.org/about/apps/>
- [5] TA-Lib - Technical Analysis Library，TA-Lib，<https://ta-lib.org/>
- [6] Yahoo!finace，Yahoo!finace，<https://finance.yahoo.com/>
- [7] Python Software Foundation，datetime — 基本的な日付型および時間型，<https://docs.python.org/ja/3/library/datetime.html>
- [8] github，backtesting.py，<https://github.com/kernc/backtesting.py>
- [9] PyPI，matplotlib 3.8.2，<https://pypi.org/project/matplotlib/>
- [10] PyPI，Numpy1.26.3，<https://pypi.org/project/numpy/>
- [11] PyPI，pandas 2.2.0，<https://pypi.org/project/pandas/>
- [12] 日本取引所グループ，利益確定売り，<https://www.jpx.co.jp/glossary/ra/460.html>
- [13] SMBC 日興証券，損切り，<https://www.smbcnikko.co.jp/terms/japan/so/J0295.html>
- [14] QUICK Money World，MACD とは何か 見方や計算式，ゴールデンクロス・ダイバージェンスについても解説，https://moneyworld.jp/news/05_00053723_news

- [15] 楽天証券, 株とは ,
<https://www.rakuten-sec.co.jp/web/domestic/special/beginner/stock01.html>
- [16] SMBC 日興証券, 日経平均株価とは? , <https://www.smbc.co.jp/kojin/toushin/gimon/start12/>
- [17] Vlogger , The History of Python , <https://python-history.blogspot.com/>