ドンチャンブレイクアウトとは

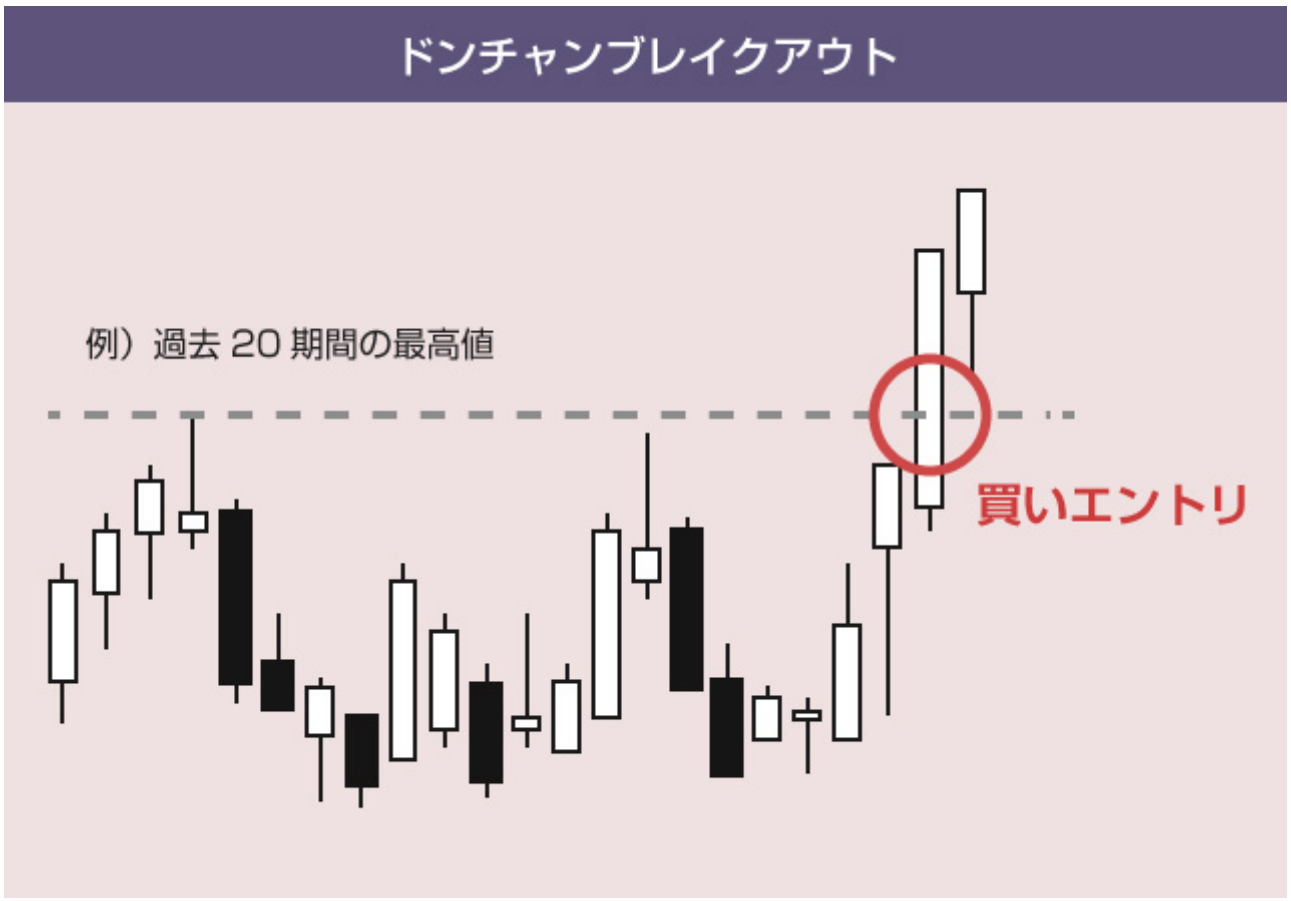
ブレイクアウト手法とは、狭い値幅での揉み合い（レンジ相場）をブレイクアウトした瞬間を捕まえてエントリーし、その後のトレンドに乗って利益を出す順張り（トレンドフォロー型）の手法のことをいいます。

ブレイクアウトを捉えるための手法はさまざまなものが開発されていますが、シストレなどの自動売買BOTの世界では、以下の２つが最も定番です。

１．ドンチャン・チャネル・ブレイクアウト

過去ｎ期間の最高値・最安値を更新したときに、その方向にエントリーする手法です。

例えば、過去20期間の最高値を更新したときに「買い」でエントリーします。次に過去20日の最安値を更新したときに決済して手仕舞いますが、このときさらに反対方向にそのままエントリー（ドテン）する場合も多いです。



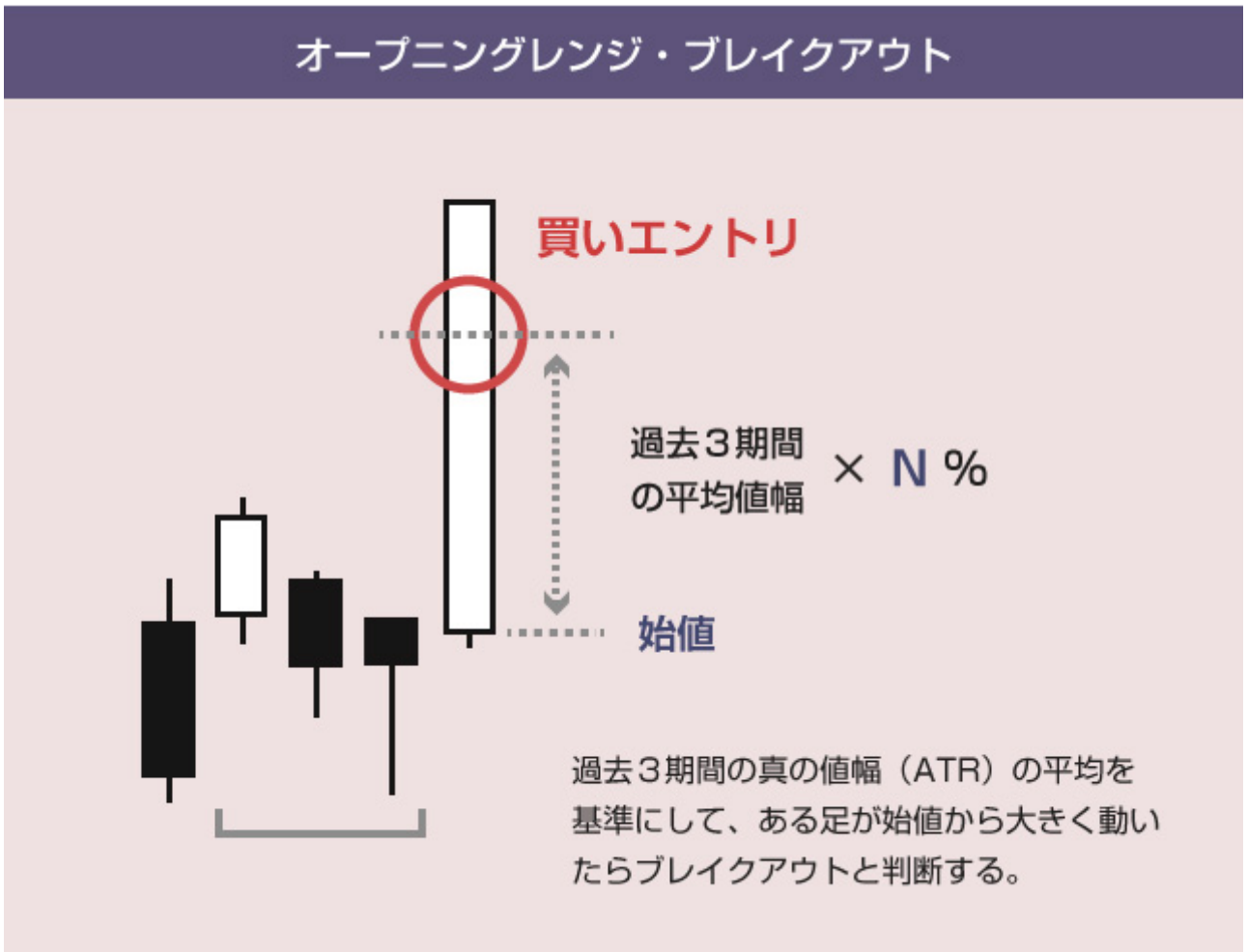
このシンプルな戦略の優位性は、トレーダーの必読本である「タートル流投資の魔術(https://www.amazon.co.jp/%E4%BC%9D%E8%AA%AC%E3%81%AE%E3%83%88%E3%83%AC%E3%83%BC%E3%83%80%E3%83%BC%E9%9B%86%E5%%E3%82%BF%E3%83%BC%E3%83%88%E3%83%AB%E6%B5%81%E6%8A%95%E8%B3%87%E3%81%AE%E9%AD%94%E8%A1%93-%E3%82%AB%E3%83%BC%E3%83%86%E3%82%A3%E3%82%B9%E3%83%BB%E3%83%95%E3%82%A7%E3%82%A4%E3%82%B9/dp/4198624267)」や、ジョン・ヒルの「究極のトレーディングガイド(https://www.amazon.co.jp/%E7%A9%B6%E6%A5%B5%E3%81%AE%E3%83%88%E3%83%AC%E3%83%BC%E3%83%87%E3%82%A3%E3%83%B3%E3%82%E5%85%A8%E7%B1%B3%E4%B8%80%E3%81%AE%E6%8A%95%E8%B3%87%E3%82%B7%E3%82%B9%E3%83%86%E3%83%A0%E5%88%86%E6%9E%E3%82%A6%E3%82%A3%E3%82%B6%E3%83%BC%E3%83%89%E3%83%96%E3%83%83%E3%82%AF%E3%82%B7%E3%83%AA%E3%83%BC%E3%82%E3%82%B8%E3%83%A7%E3%83%B3%E3%83%BBR%E3%83%BB%E3%83%92%E3%83%AB/dp/4775970151)」などの本に詳しい解説があります。

非常に単純な売買ロジックなので、プログラミングしやすく、シストレや自動売買BOTの勉強にも最適です。

２．オープニングレンジ・ブレイクアウト

オープニングレンジ・ブレイクアウトも同じく「どうやったらブレイクアウトの開始点を、シンプルな数字とロジックで発見できるか？」という視点で開発された手法です。

こちらは、過去ｎ期間の値幅の平均値（ATR）を基準値として利用し、ある足が始値から（基準値の）X%以上動いたらその方向にエントリーします。例えば、過去３期間の安値–高値の平均値が10万円でXが70%とすると、次の足で始値から７万円上に動いた時点で買いでエントリーします。



前の足が陰線か陽線かで場合分けをして、買いと売りとで別々のX%のパラメーターを使用することが多いです。

この戦略については、ラリーウィリアムズの「短期売買法(https://www.amazon.co.jp/%E3%83%A9%E3%83%AA%E3%83%BC%E3%83%BB%E3%82%A6%E3%82%A3%E3%83%AA%E3%82%A2%E3%83%A0%E3%82%E3%82%A6%E3%82%A3%E3%82%B6%E3%83%BC%E3%83%89%E3%83%96%E3%83%83%E3%82%AF-%E3%83%A9%E3%83%AA%E3%83%BC%E3%83%BB%E3%82%A6%E3%82%A3%E3%83%AA%E3%82%A2%E3%83%A0%E3%82%BA/dp/4775971603)」や、先ほどのジョンヒルの「究極のトレーディングガイド(https://www.amazon.co.jp/%E7%A9%B6%E6%A5%B5%E3%81%AE%E3%83%88%E3%83%AC%E3%83%BC%E3%83%87%E3%82%A3%E3%83%B3%E3%82%E5%85%A8%E7%B1%B3%E4%B8%80%E3%81%AE%E6%8A%95%E8%B3%87%E3%82%B7%E3%82%B9%E3%83%86%E3%83%A0%E5%88%86%E6%9E%E3%82%A6%E3%82%A3%E3%82%B6%E3%83%BC%E3%83%89%E3%83%96%E3%83%83%E3%82%AF%E3%82%B7%E3%83%AA%E3%83%BC%E3%82%E3%82%B8%E3%83%A7%E3%83%B3%E3%83%BBR%E3%83%BB%E3%83%92%E3%83%AB/dp/4775970151)」などの本に詳しい解説があります。

３．戦略の使い方

どちらも多くのトレーダーが知っている手法ですが、実際には、どの時間軸でどういうパラメーターを使うか、どういうトレンド判定の条件やフィルターと組み合わて「騙し」を除去するか、どういう手仕舞いの方法を選択するか、などによって全く違う成績になります。

これらの手法は、あくまで売買ロジックを考えるときにベースのアイデアとして使いやすいというだけで、それ自体が必勝法というわけではありません。詳しい戦略のカスタマイズ方法などは、今後、解説していく予定です。

今回はバックテストの方法の解説記事なので、まずは一番シンプルな「ドンチャン・ブレイクアウト」のロジックをpythonでコーディングするところから始めます。

ドンチャンブレイクアウトは非常に単純なロジックなので、今まで勉強した知識だけでも、pythonのプログラムコードを書くことができます。早速、やっていきましょう！

ドンチャン・ブレイクアウトのpythonコード

プログラミング初心者の方は、いきなり複雑なロジックを書こうとしてはいけません。まずは必要最小限のシンプルな骨組みの部分を作り、そこから徐々に条件などを足していくのがお勧めです。

例えば、買いと売りの両方を考えて混乱するならまずは買いエントリーだけ実装しましょう。エントリーと手仕舞いの両方を考えて混乱するなら、まずは「ｎ足後に無条件で手仕舞う」といったシンプルなロジックで実装しましょう。

ではドンチャン・ブレイクアウトの最もシンプルなロジックとは何でしょうか？ 今回は以下のように定義してみます。

１．今回実装するロジック

１）過去ｎ期間の最高値を直近の足の高値が上回ったら（その足の終値の指値で）買いエントリーする。過去ｎ期間の最安値を直近の足の安値が下回ったら売りエントリーする。
２）買いエントリーをした場合、過去ｎ期間の最安値を更新したら手仕舞い、さらに売りエントリーする
３）売りエントリーをした場合、過去ｎ期間の最高値を更新したら手仕舞い、さらに買いエントリーする

パラメーター最適化とは

パラメーターとは、ある自動売買BOTのロジックの中で自由に動かして変更できる値のことをいいます。ｎ期間ドンチャン・ブレイクアウトBOTには、以下の２つのパラメーターがありました。

１）X分足の時間軸を使う
２）ｎ期間の最高値（最安値）ブレイクアウトで買う

同じドンチャンブレイクアウトでも、15分足で10期間のドンチャン・ブレイクアウト戦略を使うのと、1時間足で30期間のドンチャン・ブレイクアウト戦略を使うのとでは、全く成績が異なります。

基本的には、数千通りくらいのパターンであれば、ただの「総当たり」で全パターンを計算すれば十分です。全パターンを試しても数十秒～１分もあれば終わりますので、複雑な機械学習や最適化アルゴリズムは必要ありません。

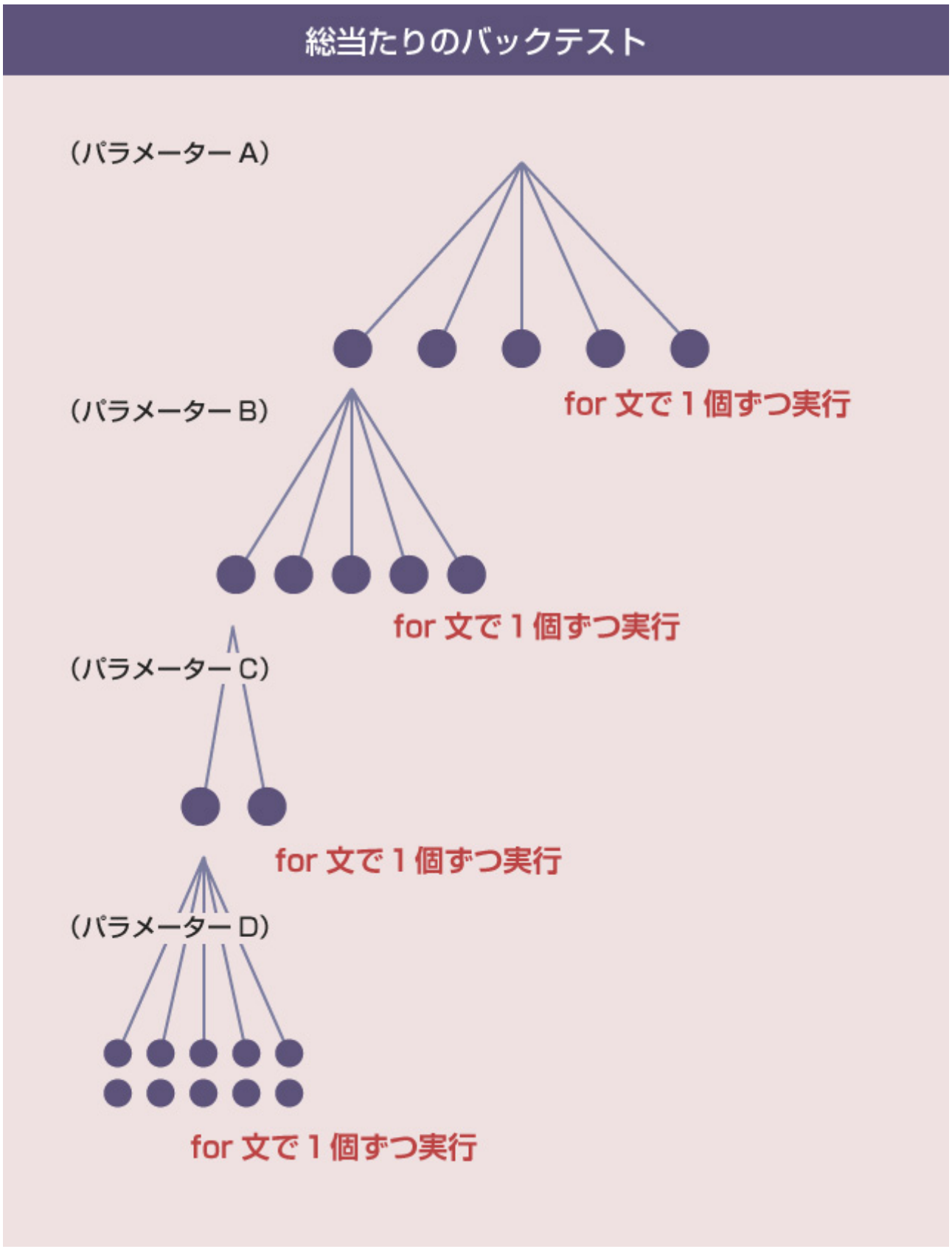
例えば、以下のようなパターンの組み合わせをすべて試したいとしましょう。

```
# ４種類のパラメーター
paramA = [ 10,20,30,40,50 ]
paramB = [ 10,20,30,40,50 ]
paramC = [ "A" , "B" ]
paramD = [ 1,2,3,4,5,6,7,8,9,10 ]
```

この場合、以下のようにfor文を書くことで全パターンの組み合わせを試すことができます。

```
# for文で総当たりのパターンを記述
for a in paramA:
    for b in paramB:
        for c in paramC:
            for d in paramD:

                # 各組合せで実行したい処理
```

パラメーターAのfor文処理をするなかで、１つの要素についてさらにパラメーターBのfor文処理をし、パラメーターBのfor文処理のなかの１つの要素に対してさらにパラメーターCのfor文処理をし....という入れ子構造にしていくわけですね。



これを実行することで、5 × 5 × 2 × 10 = 500通りの組み合わせについて、全てのパターンでバックテストを実行することができます。

シンプルな記述方法

なお、pythonでは上記のコードをもっとシンプルにして以下のように書くことができます。

```
    combinations = [(a, b, c, d)
        for a in paramA
        for b in paramB
        for c in paramC
        for d in paramD]

    for a,b,c,d in combinations:
        # 各組合せで実行したい処理
```

試すパラメーターの組み合わせ

なお、今回のドンチャンブレイクアウトのパラメーター最適化では、以下の組み合わせを総当たりで全て試してみることにします。

１）使用する時間軸
⇒　30分足／１時間足／２時間足／６時間足／12時間足／１日足

２）上値ブレイクアウトの判定期間
⇒　10/15/20/25/30/35/40/45

３）下値ブレイクアウトの判定期間
⇒　　10/15/20/25/30/35/40/45

４）判定に使用する価格
⇒　　高値・安値／終値・終値

時間足が６通り、判定期間が上値ブレイクアウト・下値ブレイクアウトでそれぞれ８通りずつ、判定に使用する価格が２通りで、合計 768通りのパターンをテストしてみましょう。

これをfor文の総当たりでテストするためには、以下のように書けばOKです。

```
# バックテストのパラメーター設定

chart_sec_list  = [ 1800, 3600, 7200, 21600, 43200, 86400 ]  # 時間足
buy_term_list   = [ 10,15,20,25,30,35,40,45 ] # 上値ブレイクの判断期間
sell_term_list  = [ 10,15,20,25,30,35,40,45 ] # 下値ブレイクの判断期間
judge_price_list = [
    {"BUY":"close_price","SELL":"close_price"}, # 終値/終値 と 高値/安値
    {"BUY":"high_price","SELL":"low_price"}
]

# for文の記述方法

combinations = [(chart_sec, buy_term, sell_term, judge_price)
    for chart_sec in chart_sec_list
    for buy_term  in buy_term_list
    for sell_term in sell_term_list
    for judge_price in judge_price_list]

for chart_sec, buy_term, sell_term,judge_price in combinations:
    # 各回のバックテスト処理を記述
    # （今までの記事で作成したもの）
```

これで総当たりのfor文の準備は完了です！

最大化したい指標を決める

何のためにパラメーター最適化を実施するのかといえば、リターンを極限まで高くしたいからですよね。なので「総利益が最大になるようなパラメーターを探す」というのも１つの方法です。

しかし自動売買BOTの評価を最終損益だけで判断するのは不十分です。より厳密にいえば、私たちが探している理想の売買ロジックは、「できるだけリスクを抑えながら、できるだけ大きなリターンを得ること」にあるはずです。そのため、リスクとリターンの比率を１つの数字で表せるような指標を「最大化したい数字」に設定すべきです。

このような指標には、シャープレシオ、MARレシオなど、いくつかの指標がありますが、ここでは一番オーソドックスな「プロフィットファクター」を使うことにします。

プロフィットファクター（PF）とは

同じ最終利益100万円でも、「利益110万円 / 損失10万円」の売買システムと、「利益200万円 / 損失100万円」の売買システムとでは、かなり意味が違ってきます。

前者は、損失が利益のおよそ1/10で済んでいるのに対し、後者は、なんと利益の半分もの損失を出してしまっています。当然、前者の方が安定した売買システムで、後者のほうがより不安定な（リスクの高い）売買システムということになります。

このような、「 総利益 / 総損失 」で表される指標のことをプロフィットファクター（PF）といいます。PFが大きければ大きいほど、そのシステムは安定していてリスクが小さいと評価できます。

バックテストに必要な時間軸のチャートをすべて取得

price_list = {} for chart_sec in chart_sec_list: price_list[chart_sec] = get_price(chart_sec,after=1451606400) print("-----{}分軸の価格データをCryptowatchから取得中-----".format(int(chart_sec/60))) time.sleep(10)

テストごとの各パラメーターの組み合わせと結果を記録する配列を準備

param_buy_term = [] param_sell_term = [] param_chart_sec = [] param_judge_price = []

result_count = [] result_winRate = [] result_returnRate = [] result_drawdown = [] result_profitFactor = [] result_gross = []

総当たりのためのfor文の準備

combinations = [(chart_sec, buy_term, sell_term, judge_price) for chart_sec in chart_sec_list for buy_term in buy_term_list for sell_term in sell_term_list for judge_price in judge_price_list]

for chart_sec, buy_term, sell_term,judge_price in combinations:

```
price = price_list[ chart_sec ]
last_data = []
i = 0

# フラッグ変数の初期化
flag = {
    "order":{
        "exist" : False,
        "side" : "",
        "price" : 0,
        "count" : 0
    },
    "position":{
        "exist" : False,
        "side" : "",
        "price": 0,
        "count":0
    },
    "records":{
        "date":[],
        "profit":[],
        "return":[],
        "side":[],
        "holding-periods":[],
        "slippage":[]
    }
}

while i < len(price):

    # ドンチャンの判定に使う期間分の安値・高値データを準備する
    if len(last_data) < buy_term or len(last_data) < sell_term:
        last_data.append(price[i])
        time.sleep(wait)
        i += 1
        continue

    data = price[i]

    if flag["order"]["exist"]:
        flag = check_order( flag )
    elif flag["position"]["exist"]:
        flag = close_position( data,last_data,flag )
    else:
        flag = entry_signal( data,last_data,flag )

    last_data.append( data )
    i += 1
    time.sleep(wait)


print("--------------------------")
print("テスト期間　:")
print("開始時点　　: " + str(price[0]["close_time_dt"]))
print("終了時点　　: " + str(price[-1]["close_time_dt"]))
print("時間軸　　: " + str(int(chart_sec/60)) + "分足で検証")
print("パラメータ１ : " + str(buy_term)  + "期間 / 買い" )
print("パラメータ２ : " + str(sell_term) + "期間 / 売り" )
print(str(len(price)) + "件のローソク足データで検証")
print("--------------------------")


result = backtest( flag )


# 今回のループで使ったパラメータの組み合わせを配列に記録する
param_buy_term.append( buy_term )
param_sell_term.append( sell_term )
param_chart_sec.append( chart_sec )
if judge_price["BUY"] == "high_price":
    param_judge_price.append( "高値/安値" )
else:
    param_judge_price.append( "終値/終値" )
```

```
# 今回のループのバックテスト結果を配列に記録する
result_count.append( result["トレード回数"] )
result_winRate.append( result["勝率"] )
result_returnRate.append( result["平均リターン"] )
result_drawdown.append( result["最大ドローダウン"] )
result_profitFactor.append( result["プロフィットファクター"] )
result_gross.append( result["最終損益"] )
```

全てのパラメータによるバックテスト結果をPandasで１つの表にする

df = pd.DataFrame({ "時間軸" : param_chart_sec, "買い期間" : param_buy_term, "売り期間" : param_sell_term, "判定基準" : param_judge_price, "トレード回数" : result_count, "勝率" : result_winRate, "平均リターン" : result_returnRate, "ドローダウン" : result_drawdown, "PF" : result_profitFactor, "最終損益" : result_gross })

列の順番を固定する

df = df[["時間軸","買い期間","売り期間","判定基準","トレード回数","勝率","平均リターン","ドローダウン","PF","最終損益"]]

トレード回数が100に満たない記録は消す

df.drop(df[df["トレード回数"] < 100].index, inplace=True)

最終結果をcsvファイルに出力

df.to_csv("result-{}.csv".format(datetime.now().strftime("%Y-%m-%d-%H-%M")))

</pre>

実行結果

これを実行すると以下のようなCSVファイルが出力されます。

無題 1 - OpenOffice Calc

ファイル(F) 編集(E) 表示(V) 挿入(I) 書式(O) ツール(T) データ(D) ウィンドウ(W) ヘルプ(H)

MS Pゴシック 10

Q34

	A	B	C	D	E	F	G	H	I	J	K	L
1		時間軸	買い期間	売り期間	判定基準	トレード回数	勝率	平均リターン	ドローダウン	PF	最終損益	
2	0	1800	10	10	終値/終値	234	32.5	-0.05	-680428	1.01	44928	
3	1	1800	10	10	高値/安値	350	30.6	-0.23	-1154065	0.87	-878576	
4	2	1800	10	15	終値/終値	190	37.4	0.21	-575837	1.06	296892	
5	3	1800	10	15	高値/安値	280	31.8	-0.42	-1701425	0.75	-1654008	
6	4	1800	10	20	終値/終値	160	38.8	0.12	-1012123	0.93	-326938	
7	5	1800	10	20	高値/安値	232	34.1	-0.24	-1046779	0.81	-1062268	
8	6	1800	10	25	終値/終値	145	40.7	0.02	-1226056	0.87	-589540	
9	7	1800	10	25	高値/安値	199	37.2	-0.24	-1033715	0.82	-948945	
10	8	1800	10	30	終値/終値	124	44.4	0.53	-1220123	1.02	80195	
11	9	1800	10	30	高値/安値	173	39.3	-0.01	-805782	0.9	-440610	
12	10	1800	10	35	終値/終値	120	41.7	0.32	-1751886	0.89	-450309	
13	11	1800	10	35	高値/安値	160	40	0.2	-620085	0.99	-46347	
14	12	1800	10	40	終値/終値	110	40.9	0.57	-1728088	0.95	-187252	
15	13	1800	10	40	高値/安値	154	40.3	0.26	-698797	1.01	27238	
16	14	1800	10	45	終値/終値	102	42.2	0.72	-1547382	0.99	-28204	
17	15	1800	10	45	高値/安値	150	38.7	0.18	-769042	0.96	-183208	
18	16	1800	15	10	終値/終値	187	31	-0.27	-910468	0.97	-132297	
19	17	1800	15	10	高値/安値	286	28	-0.16	-992848	0.96	-215250	
20	18	1800	15	15	終値/終値	155	36.8	0.05	-549243	1.04	190785	
21	19	1800	15	15	高値/安値	234	29.9	-0.29	-1059905	0.84	-960953	
22	20	1800	15	20	終値/終値	131	41.2	0.19	-703872	1.03	124863	
23	21	1800	15	20	高値/安値	192	36.5	0.06	-667903	0.96	-200442	
24	22	1800	15	25	終値/終値	120	40.8	0.01	-932900	0.93	-274823	
25	23	1800	15	25	高値/安値	167	38.3	0.07	-589558	0.98	-86877	
26	24	1800	15	30	終値/終値	104	43.3	0.7	-921778	1.15	498019	
27	25	1800	15	30	高値/安値	145	42.1	0.39	-547153	1.09	363609	
28	26	1800	15	35	終値/終値	100	41	0.47	-1323579	0.99	-18436	
29	27	1800	15	35	高値/安値	136	43.4	0.46	-649064	1.09	325350	
30	29	1800	15	40	高値/安値	130	43.8	0.55	-785528	1.11	407874	
31	31	1800	15	45	高値/安値	126	41.3	0.47	-858891	1.05	200131	
32	32	1800	20	10	終値/終値	161	32.9	-0.23	-885539	1.04	164522	
33	33	1800	20	10	高値/安値	236	28.8	-0.1	-1053964	1.07	381376	
34	34	1800	20	15	終値/終値	137	38.7	0.09	-575494	1.09	358483	
35	35	1800	20	15	高値/安値	196	32.7	-0.15	-796514	0.99	-66834	
36	36	1800	20	20	終値/終値	113	46	0.51	-714032	1.25	822761	
37	37	1800	20	20	高値/安値	164	39	0.21	-794044	1.1	441603	
38	38	1800	20	25	終値/終値	104	45.2	0.32	-611853	1.14	442366	
39	39	1800	20	25	高値/安値	139	44.6	0.31	-751704	1.17	658027	
40	41	1800	20	30	高値/安値	121	47.1	0.7	-601869	1.33	1068545	
41	43	1800	20	35	高値/安値	112	50	0.96	-410342	1.43	1242356	
42	45	1800	20	40	高値/安値	106	50.9	1.1	-482652	1.48	1324880	
43	47	1800	20	45	高値/安値	104	47.1	0.99	-537216	1.4	1143835	
44	48	1800	25	10	終値/終値	134	36.6	0.13	-677346	1.2	732619	
45	49	1800	25	10	高値/安値	201	32.3	0.01	-751467	1.16	695677	
46	50	1800	25	15	終値/終値	114	43.9	0.6	-530604	1.28	954849	
47	51	1800	25	15	高値/安値	169	36.1	-0.1	-483407	1.01	57062	
48	53	1800	25	20	高値/安値	139	43.2	0.26	-588000	1.11	423818	
49	55	1800	25	25	高値/安値	119	46.2	0.41	-475283	1.18	623633	

Sheet1 Sheet2 Sheet3

プロフィットファクター（PF）のJ列を「降順」で並び変えてみると、ドンチャン・ブレイクアウトの有望なパラメーターの組み合わせがわかります。

▽ PFで並び替え後の成績ベスト30

	A	B	C	D	E	F	G	H	I	J
1	時間軸	買い期間	売り期間	判定基準	トレード回数	勝率	平均リターン	ドローダウン	PF	最終損益
2	21600	10	10	終値/終値	107	50.5	3.97	-383405	3.15	3002469
3	7200	20	20	高値/安値	105	52.4	3.97	-290334	2.66	3310836
4	21600	25	10	高値/安値	106	49.1	3.17	-496919	2.54	2356237
5	7200	15	10	終値/終値	121	50.4	3.29	-429929	2.4	3043208
6	7200	15	15	終値/終値	107	46.7	3.62	-403706	2.37	2971022
7	7200	20	10	終値/終値	109	47.7	3.19	-287252	2.37	2923670
8	7200	25	10	高値/安値	147	45.6	2.32	-299767	2.31	3010128
9	7200	25	15	高値/安値	119	49.6	3.12	-262814	2.3	2778213
10	7200	20	15	高値/安値	133	49.6	3.03	-267366	2.27	2967961
11	7200	15	20	高値/安値	117	49.6	3.7	-290684	2.25	2994735
12	3600	40	20	高値/安値	104	44.2	1.72	-304034	2.22	2887108
13	7200	20	10	高値/安値	169	45	2.05	-270037	2.22	3130358
14	3600	40	15	高値/安値	116	42.2	1.47	-386911	2.19	2883571
15	21600	10	15	高値/安値	131	46.6	3.4	-485734	2.19	2270360
16	21600	10	20	高値/安値	107	47.7	4.87	-688114	2.19	2059774
17	21600	15	15	高値/安値	107	50.5	4.19	-591672	2.15	2246105
18	7200	10	15	終値/終値	117	46.2	3.57	-640494	2.12	2862356
19	21600	20	10	高値/安値	125	47.2	2.68	-496919	2.12	2108214
20	3600	45	15	高値/安値	110	42.7	1.51	-424205	2.11	2674865
21	7200	10	10	終値/終値	133	48.9	3.26	-880596	2.1	2920644
22	7200	15	25	高値/安値	101	50.5	3.93	-496916	2.06	2499706
23	7200	10	20	高値/安値	135	44.4	3.34	-490852	2	2849792
24	7200	15	15	高値/安値	145	48.3	3	-364595	2	2725920
25	7200	30	10	高値/安値	137	44.5	1.97	-384330	1.97	2388849
26	3600	25	20	高値/安値	124	42.7	1.73	-401759	1.96	2731032
27	7200	10	10	高値/安値	209	45	2.04	-332361	1.94	3112856
28	7200	15	10	高値/安値	187	42.8	1.99	-290736	1.93	2828758
29	7200	30	15	高値/安値	113	46	2.61	-272143	1.91	2159787
30	7200	10	15	高値/安値	167	46.7	2.72	-375357	1.88	2767009
31	3600	25	30	高値/安値	102	47.1	2.22	-463274	1.87	2273264

上記のコードではトレード回数が100回に満たなかった結果データを全て捨てているので、どのパラメーターの組み合わせも最低限のサンプル数を確保できています。上位30の成績のパラメーターをみると、ほとんどが 2 時間足以上の時間軸なのがわかります。

▽ PFで並び替え後の成績ワースト30

	A	B	C	D	E	F	G	H	I	J
1	時間軸	買い期間	売り期間	判定基準	トレード回数	勝率	平均リターン	ドローダウン	PF	最終損益
2	1800	10	15	高値/安値	280	31.8	-0.42	-1701425	0.75	-1654008
3	1800	10	20	高値/安値	232	34.1	-0.24	-1046779	0.81	-1062268
4	1800	15	15	高値/安値	234	29.9	-0.29	-1059905	0.84	-960953
5	1800	10	25	高値/安値	199	37.2	-0.24	-1033715	0.82	-948945
6	1800	10	10	高値/安値	350	30.6	-0.23	-1154065	0.87	-878576
7	1800	10	25	終値/終値	145	40.7	0.02	-1226056	0.87	-589540
8	1800	10	35	終値/終値	120	41.7	0.32	-1751886	0.89	-450309
9	1800	10	30	高値/安値	173	39.3	-0.01	-805782	0.9	-440610
10	1800	10	20	終値/終値	160	38.8	0.12	-1012123	0.93	-326938
11	1800	15	25	終値/終値	120	40.8	0.01	-932900	0.93	-274823
12	1800	15	10	高値/安値	286	28	-0.16	-992848	0.96	-215250
13	1800	15	20	高値/安値	192	36.5	0.06	-667903	0.96	-200442
14	1800	10	40	終値/終値	110	40.9	0.57	-1728088	0.95	-187252
15	1800	10	45	高値/安値	150	38.7	0.18	-769042	0.96	-183208
16	1800	15	10	終値/終値	187	31	-0.27	-910468	0.97	-132297
17	1800	15	25	高値/安値	167	38.3	0.07	-589558	0.98	-86877
18	1800	20	15	高値/安値	196	32.7	-0.15	-796514	0.99	-66834
19	1800	10	35	高値/安値	160	40	0.2	-620085	0.99	-46347
20	1800	10	45	終値/終値	102	42.2	0.72	-1547382	0.99	-28204
21	1800	15	35	終値/終値	100	41	0.47	-1323579	0.99	-18436
22	1800	10	40	高値/安値	154	40.3	0.26	-698797	1.01	27238
23	1800	10	10	終値/終値	234	32.5	-0.05	-680428	1.01	44928
24	1800	25	15	高値/安値	169	36.1	-0.1	-483407	1.01	57062
25	1800	10	30	終値/終値	124	44.4	0.53	-1220123	1.02	80195
26	1800	15	20	終値/終値	131	41.2	0.19	-703872	1.03	124863
27	1800	20	10	終値/終値	161	32.9	-0.23	-885539	1.04	164522
28	1800	15	15	終値/終値	155	36.8	0.05	-549243	1.04	190785
29	1800	15	45	高値/安値	126	41.3	0.47	-858891	1.05	200131
30	3600	10	10	高値/安値	311	36.3	0.07	-1104916	1.03	203135
31	1800	10	15	終値/終値	190	37.4	0.21	-575837	1.06	296892

逆に最もプロフィットファクターの低かった（悪かった成績）30コをみると、短い時間軸（30分足）に集中しているのがわかります。ドンチアンブレイクアウトは、あまり短い時間軸には適さない戦略の可能性がありそうです。

（ただし時間軸によってテストに使用しているローソク足の期間が違うので、厳密には比較できない点に注意してください。）

実行結果 2

次はさらに範囲を絞ってテストをしてみましょう！

今度は時間軸を 1 時間足・2 時間足だけに絞り、上値ブレイクアウトの期間・下値ブレイクアウトの期間を10～55期間の範囲にして 1 期間刻みでテストしています。以下のようにパラメーターを設定すればOKですね。

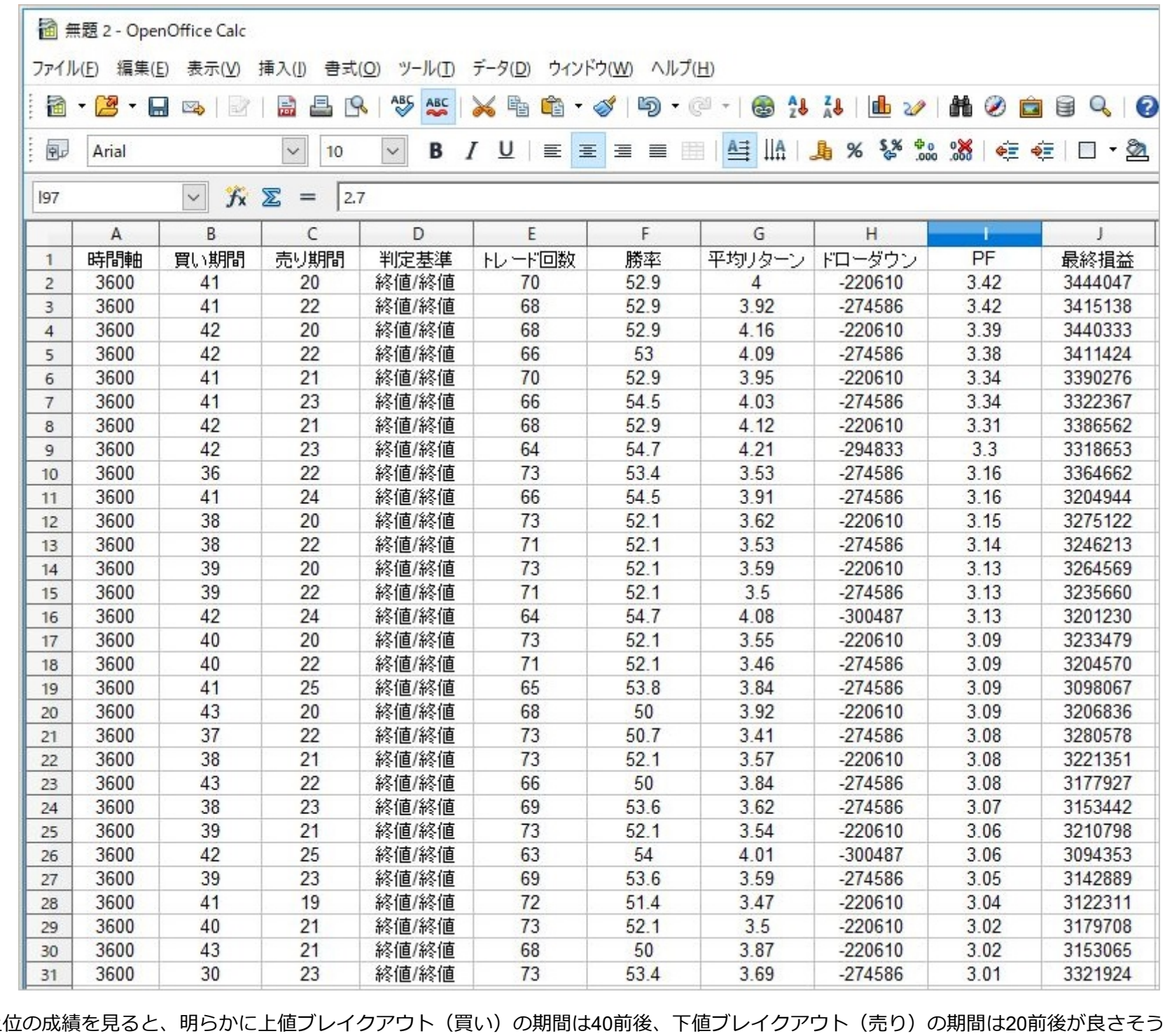
```
# バックテストのパラメーター設定

chart_sec_list  = [ 3600,7200 ]
buy_term_list   = np.arange(10,56) # 10～55までの連番の配列を作る
sell_term_list  = np.arange(10,56)
```

この条件でテストしてみると以下のようなCSVファイルが出力されます。

・CSV結果出力ファイル１ (https://ryota-trade.com/wp-content/uploads/csv/result-2018-04-23-10-11.csv)
・CSV結果出力ファイル２ (https://ryota-trade.com/wp-content/uploads/csv/result-2018-04-23-11-12.csv)

ファイル１は先ほどと同じくトレード回数が100回に満たなかったテスト結果を削除したもの、ファイル２は削除せずに全ての結果を残したものです。今回はファイル２を使ってみましょう。

▽ PFで並び替え後の成績ベスト30（１時間足）



	A	B	C	D	E	F	G	H	I	J
1	時間軸	買い期間	売り期間	判定基準	トレード回数	勝率	平均リターン	ドローダウン	PF	最終損益
2	3600	41	20	終値/終値	70	52.9	4	-220610	3.42	3444047
3	3600	41	22	終値/終値	68	52.9	3.92	-274586	3.42	3415138
4	3600	42	20	終値/終値	68	52.9	4.16	-220610	3.39	3440333
5	3600	42	22	終値/終値	66	53	4.09	-274586	3.38	3411424
6	3600	41	21	終値/終値	70	52.9	3.95	-220610	3.34	3390276
7	3600	41	23	終値/終値	66	54.5	4.03	-274586	3.34	3322367
8	3600	42	21	終値/終値	68	52.9	4.12	-220610	3.31	3386562
9	3600	42	23	終値/終値	64	54.7	4.21	-294833	3.3	3318653
10	3600	36	22	終値/終値	73	53.4	3.53	-274586	3.16	3364662
11	3600	41	24	終値/終値	66	54.5	3.91	-274586	3.16	3204944
12	3600	38	20	終値/終値	73	52.1	3.62	-220610	3.15	3275122
13	3600	38	22	終値/終値	71	52.1	3.53	-274586	3.14	3246213
14	3600	39	20	終値/終値	73	52.1	3.59	-220610	3.13	3264569
15	3600	39	22	終値/終値	71	52.1	3.5	-274586	3.13	3235660
16	3600	42	24	終値/終値	64	54.7	4.08	-300487	3.13	3201230
17	3600	40	20	終値/終値	73	52.1	3.55	-220610	3.09	3233479
18	3600	40	22	終値/終値	71	52.1	3.46	-274586	3.09	3204570
19	3600	41	25	終値/終値	65	53.8	3.84	-274586	3.09	3098067
20	3600	43	20	終値/終値	68	50	3.92	-220610	3.09	3206836
21	3600	37	22	終値/終値	73	50.7	3.41	-274586	3.08	3280578
22	3600	38	21	終値/終値	73	52.1	3.57	-220610	3.08	3221351
23	3600	43	22	終値/終値	66	50	3.84	-274586	3.08	3177927
24	3600	38	23	終値/終値	69	53.6	3.62	-274586	3.07	3153442
25	3600	39	21	終値/終値	73	52.1	3.54	-220610	3.06	3210798
26	3600	42	25	終値/終値	63	54	4.01	-300487	3.06	3094353
27	3600	39	23	終値/終値	69	53.6	3.59	-274586	3.05	3142889
28	3600	41	19	終値/終値	72	51.4	3.47	-220610	3.04	3122311
29	3600	40	21	終値/終値	73	52.1	3.5	-220610	3.02	3179708
30	3600	43	21	終値/終値	68	50	3.87	-220610	3.02	3153065
31	3600	30	23	終値/終値	73	53.4	3.69	-274586	3.01	3321924

１時間足の上位の成績を見ると、明らかに上値ブレイクアウト（買い）の期間は40前後、下値ブレイクアウト（売り）の期間は20前後が良さそう、という傾向が見えます。また判定基準は「終値」が上位を独占していますね。

▽ PFで並び替え後の成績ベスト30（２時間足）

無題 1 - OpenOffice Calc

ファイル(F) 編集(E) 表示(V) 挿入(I) 書式(O) ツール(T) データ(D) ウィンドウ(W) ヘルプ(H)

Arial 10

I63 4.72

	A	B	C	D	E	F	G	H	I	J
1	時間軸	買い期間	売り期間	判定基準	トレード回数	勝率	平均リターン	ドローダウン	PF	最終損益
2	7200	52	34	終値/終値	41	51.2	8.36	-230673	5.85	3621579
3	7200	53	34	終値/終値	41	51.2	8.36	-230673	5.85	3621579
4	7200	54	34	終値/終値	41	51.2	8.36	-230673	5.85	3621579
5	7200	55	34	終値/終値	41	51.2	8.36	-230673	5.85	3621579
6	7200	52	21	終値/終値	49	44.9	7.58	-211929	5.78	3851489
7	7200	53	21	終値/終値	49	44.9	7.58	-211929	5.78	3851489
8	7200	54	21	終値/終値	49	44.9	7.58	-211929	5.78	3851489
9	7200	55	21	終値/終値	49	44.9	7.58	-211929	5.78	3851489
10	7200	52	35	終値/終値	41	48.8	8.15	-278627	5.49	3572294
11	7200	53	35	終値/終値	41	48.8	8.15	-278627	5.49	3572294
12	7200	54	35	終値/終値	41	48.8	8.15	-278627	5.49	3572294
13	7200	55	35	終値/終値	41	48.8	8.15	-278627	5.49	3572294
14	7200	52	36	終値/終値	41	48.8	7.98	-278627	5.44	3564794
15	7200	53	36	終値/終値	41	48.8	7.98	-278627	5.44	3564794
16	7200	54	36	終値/終値	41	48.8	7.98	-278627	5.44	3564794
17	7200	55	36	終値/終値	41	48.8	7.98	-278627	5.44	3564794
18	7200	52	22	終値/終値	49	44.9	7.28	-211929	5.42	3709340
19	7200	53	22	終値/終値	49	44.9	7.28	-211929	5.42	3709340
20	7200	54	22	終値/終値	49	44.9	7.28	-211929	5.42	3709340
21	7200	55	22	終値/終値	49	44.9	7.28	-211929	5.42	3709340
22	7200	52	26	終値/終値	45	48.9	8.24	-233560	5.4	3671695
23	7200	53	26	終値/終値	45	48.9	8.24	-233560	5.4	3671695
24	7200	54	26	終値/終値	45	48.9	8.24	-233560	5.4	3671695
25	7200	55	26	終値/終値	45	48.9	8.24	-233560	5.4	3671695
26	7200	52	33	終値/終値	43	51.2	7.05	-230673	5.35	3503147
27	7200	53	33	終値/終値	43	51.2	7.05	-230673	5.35	3503147
28	7200	54	33	終値/終値	43	51.2	7.05	-230673	5.35	3503147
29	7200	55	33	終値/終値	43	51.2	7.05	-230673	5.35	3503147
30	7200	52	23	終値/終値	49	44.9	7.13	-233560	5.2	3657939
31	7200	53	23	終値/終値	49	44.9	7.13	-233560	5.2	3657939

２時間足の上位の成績を見ると、上値ブレイクアウト（買い）の期間は50以上、下値ブレイクアウト（売り）の期間は20～30期間がいいようですね。またやはり「終値」でブレイクアウトの判定をした方が、トレード回数は減るものの成績が良くなる傾向にありそうです。

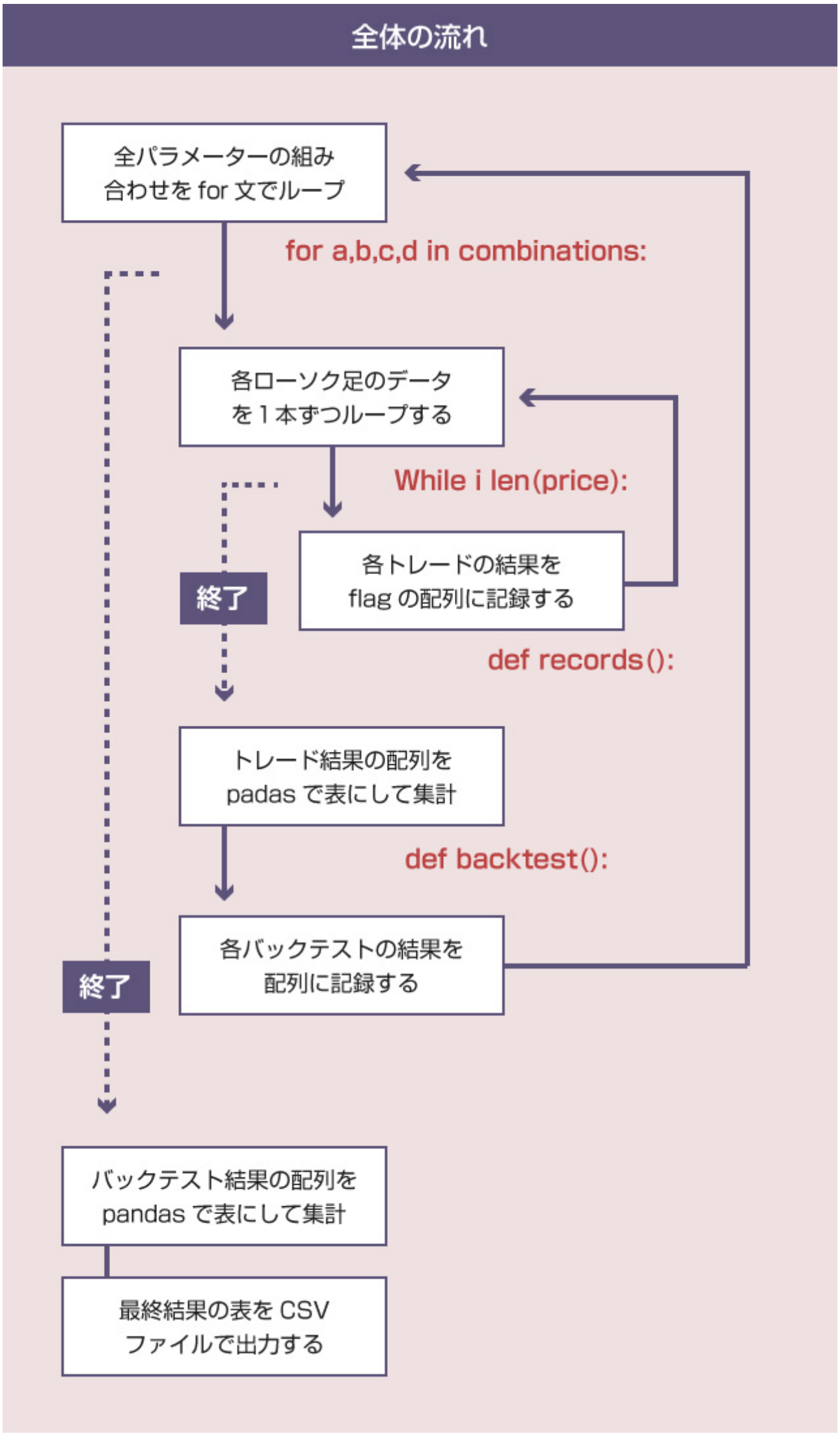
より詳しく分析したい方は、上位のパラメーターを前回までの記事で学習した「月別集計 (https://ryota-trade.com/?p=2113)」「資産曲線 (https://ryota-trade.com/?p=2023)」などを使って調べてみるといいでしょう。

Pythonコードの解説

さて、それでは新しく変更した部分のpythonコードを解説しておきましょう！

基本的なコードは前回まで勉強した内容とほとんど変わっていないことに気付いたと思います。要するに、今まで作成してきた「While文でローソク足を回してバックテストし、最終成績をbacktest() で集計するコード」を、まるまる上記で説明した「総当たりのfor文」の中に入れるだけですね。

図にすると以下のような感じです。



こののような入れ子構造になっている点だけ混乱しなければ、上記のコードで難しい箇所はなかったと思います。

今までの記事では、バックテスト集計用の関数 backtest() は、単に集計・計算した結果を表示（print）するだけでしたが、今回のパラメーター最適化のコードでは、各バックテストの結果を返すように変更する必要があります。

そのためバックテスト集計用の関数に以下の部分を足しています。

```
    # バックテストの計算結果を返す
    result = {
        "トレード回数"      : len(records),
        "勝率"              : round(len(records[records.Profit>0]) / len(records) * 100,1),
        "平均リターン"      : round(records.Rate.mean(),2),
        "最大ドローダウン" : -1 * records.Drawdown.max(),
        "最終損益"          : records.Profit.sum(),
        "プロフィットファクター" : round( -1 * (records[records.Profit>0].Profit.sum() / records[records.Profit<0].Profit.sum()) ,2)
    }

    return result
```

各バックテストの結果を配列に記録して、それを最後にpandasで１つの表にする方法は、前回までの記事の「各トレード結果を配列に記録して最後に集計する方法」と全く同じなので、説明は省きます。

drop() で不要な行を削除する

唯一新しく登場したのは、最後のdrop()の箇所でしょう。

今回のコードでは、トレード回数の少ないパラメーターの組み合わせを除外できるようにしています。それが以下の部分のコードです。

```
# トレード回数が100に満たない記録は消す
df.drop( df[ df["トレード回数"] < 100].index, inplace=True )
```

表名.drop()は、行のindexを指定することで、その行を表から削除することのできる関数です。これを使って以下のような仕組みで、トレード回数の少なすぎるバックテスト結果を削除しています。

```
# トレード回数が100未満のデータを抽出
df[df["トレード回数"] < 100]

# トレード回数が100未満のデータの行のindexを取得
df[df["トレード回数"] < 100].index

# トレード回数が100未満のデータを行ごと削除
df.drop( df[df["トレード回数"] < 100].index )
```

なお、inplce=Trueは、現在の表データに結果をそのまま上書きする、という意味です。以上でコードの解説はおしまいです！

まとめ

この章の前の方の記事でも解説したように、このようなパラメーター調整には常にカーブフィッティングの問題がつきまといます。以下のような点に注意しておきましょう。

（1）パラメーターの数を増やし過ぎないようにする
（2）おおまかな傾向を捉えることを目的にする
（3）トレード数（サンプル）が少なくならないようにする

一般論としていえば、今回のドンチャン・チャネル・ブレイクアウトのように広範なパラメーターのどの値を使ってもプラスの期待値が出る、という場合で、おおまかな傾向を捉える目的（例えば、20期間あたりが一番良さそう）でパラメーターを最適化するのは問題ありません。

一番問題があるのは、例えば、「6」という数字を使えば利益が出る、「8」という数字を使うとマイナスに転落する、「13」という数字を使えばプラスになる、といった全く連続性も傾向もないパラメーターを恣意的に調整することです。これをすると、過去データでしか利益の出せないパラメーター調整になります。

In [ ]: