Strategy Breakdown

read the data in...Backtrader has requirements on how data is presented so may change in final code.

In [91]:

```
df_can = pd.read_csv('data/aapl_train.csv')
df_can.columns = ['open', 'high', 'low', 'close']
df_can['direction']  = 0
df_can['pivot_down'] = False
df_can['pivot_up']   = False
df_can['pl_value']   = -1.0
df_can.tail()
```

Out[91]:

	open	high	low	close	direction	pivot_down	pivot_up	pl_value
195	95.19	95.49	95.10	95.14	0	False	False	-1.0
196	95.14	95.48	94.91	95.47	0	False	False	-1.0
197	95.47	95.63	95.27	95.45	0	False	False	-1.0
198	95.79	95.97	95.65	95.82	0	False	False	-1.0
199	95.83	96.36	95.72	96.22	0	False	False	-1.0

In this example we use a very short window length to illustrate what is happening

We are setting pivot points using min/max values within a rolling window of three

for example...low_rolling_right, the first two numbers in the column come from the min number in the following

- first row: 94.01 [94.25, 94.01, 94.75]
- second row: 94.01 [94.01, 94.75, 95.04]

In [92]:

```
window_length = 3
df_can['high_rolling_left']  = df_can['high'].rolling(window_length + 1).apply(r
df_can['low_rolling_left']   = df_can['low'].rolling(window_length + 1).apply(np
df_can['high_rolling_right'] = df_can['high'][::-1].rolling(window_length + 1).a
df_can['low_rolling_right']  = df_can['low'][::-1].rolling(window_length + 1).ap

df_can.tail(10)
```

Out[92]:

	open	high	low	close	direction	pivot_down	pivot_up	pl_value	high_rolling_left	low_
190	94.59	94.61	94.25	94.40	0	False	False	-1.0	94.70	
191	94.41	94.94	94.01	94.79	0	False	False	-1.0	94.94	
192	94.81	95.40	94.75	95.22	0	False	False	-1.0	95.40	
193	95.22	95.65	95.04	95.60	0	False	False	-1.0	95.65	
194	95.61	95.65	95.10	95.16	0	False	False	-1.0	95.65	
195	95.19	95.49	95.10	95.14	0	False	False	-1.0	95.65	
196	95.14	95.48	94.91	95.47	0	False	False	-1.0	95.65	
197	95.47	95.63	95.27	95.45	0	False	False	-1.0	95.65	
198	95.79	95.97	95.65	95.82	0	False	False	-1.0	95.97	
199	95.83	96.36	95.72	96.22	0	False	False	-1.0	96.36	

Next we create two more columns. In the pivot_down col the follwing condition must be met

high = high_rolling_left = high_rolling_right

In [93]:

```
df_can['pivot_up'] = [True if x.low == x.low_rolling_left and x.low == x.low_rol
                      for x in df_can.itertuples()]
df_can['pivot_down'] = [True if x.high == x.high_rolling_left and x.high == x.hi
                        for x in df_can.itertuples()]
```

Had to set the window length much bigger to provide an example

look at line 162 to see that pivot_down set to true because it met the condition described above

- high = 96.27
- high_rolling_left = 96.27
- high_rolling_right = 96.27

In [127]:

```
windows_length = 40

for i in range(windows_length, 200):
    tmp_df = df_can[(i - windows_length):i]
    direction = 0.0
    direction_long = False
    direction_short = False
    pivot_line = -1.0

tmp_df
```

Out[127]:

	open	high	low	close	direction	pivot_down	pivot_up	pl_value	high_rolling_left	low_
159	95.31	95.72	95.22	95.31	0	False	False	-1.0	95.90	
160	95.32	95.70	95.08	95.45	0	False	False	-1.0	95.90	
161	95.45	95.88	95.35	95.80	0	False	False	-1.0	95.90	
162	95.81	96.27	95.30	95.99	0	True	False	-1.0	96.27	
163	95.99	96.13	95.35	95.80	0	False	False	-1.0	96.27	
164	95.80	96.00	95.20	95.95	0	False	False	-1.0	96.27	
165	95.93	96.20	95.63	95.65	0	False	False	-1.0	96.27	
166	95.64	95.85	95.27	95.41	0	False	False	-1.0	96.20	
167	95.41	95.73	94.88	94.89	0	False	False	-1.0	96.20	
168	94.89	95.29	94.75	94.94	0	False	False	-1.0	96.20	
169	94.97	95.04	94.43	94.69	0	False	True	-1.0	95.85	
170	94.70	94.77	94.44	94.70	0	False	False	-1.0	95.73	
171	94.94	95.37	94.81	95.04	0	False	False	-1.0	95.37	
172	95.04	95.48	95.01	95.30	0	False	False	-1.0	95.48	
173	95.30	95.89	95.22	95.52	0	True	False	-1.0	95.89	
174	95.53	95.88	95.32	95.60	0	False	False	-1.0	95.89	
175	95.60	95.87	95.00	95.01	0	False	False	-1.0	95.89	
176	95.01	95.50	95.00	95.22	0	False	False	-1.0	95.89	
177	95.24	95.47	95.04	95.40	0	False	False	-1.0	95.88	
178	95.40	95.49	94.88	94.93	0	False	False	-1.0	95.87	
179	94.93	94.94	94.55	94.66	0	False	False	-1.0	95.50	
180	93.53	93.87	93.25	93.81	0	False	True	-1.0	95.49	
181	93.81	94.12	93.52	93.73	0	False	False	-1.0	95.49	
182	93.73	94.34	93.62	94.09	0	False	False	-1.0	94.94	
183	94.08	94.48	93.98	94.29	0	True	False	-1.0	94.48	
184	94.28	94.33	93.85	93.94	0	False	False	-1.0	94.48	
185	93.93	94.14	93.62	93.73	0	False	False	-1.0	94.48	
186	93.72	93.83	93.59	93.67	0	False	True	-1.0	94.48	

	open	high	low	close	direction	pivot_down	pivot_up	pl_value	high_rolling_left	low_
187	93.67	94.37	93.65	94.13	0	False	False	-1.0	94.37	
188	94.12	94.34	93.97	94.07	0	False	False	-1.0	94.37	
189	94.31	94.70	94.31	94.60	0	False	False	-1.0	94.70	
190	94.59	94.61	94.25	94.40	0	False	False	-1.0	94.70	
191	94.41	94.94	94.01	94.79	0	False	False	-1.0	94.94	
192	94.81	95.40	94.75	95.22	0	False	False	-1.0	95.40	
193	95.22	95.65	95.04	95.60	0	True	False	-1.0	95.65	
194	95.61	95.65	95.10	95.16	0	True	False	-1.0	95.65	
195	95.19	95.49	95.10	95.14	0	False	False	-1.0	95.65	
196	95.14	95.48	94.91	95.47	0	False	True	-1.0	95.65	
197	95.47	95.63	95.27	95.45	0	False	False	-1.0	95.65	
198	95.79	95.97	95.65	95.82	0	False	False	-1.0	95.97	

In [128]:

```
np.where(tmp_df['pivot_down'][:(-window_length)] == True)[0]
```

Out[128]:

```
array([ 3, 14, 24, 34, 35])
```

In [129]:

```
tmp_df['pivot_down'].iloc[5]
```

Out[129]:

```
False
```

```
window_length = 3 (for example)

Loop through a range from window_length to length of dataframe

slice df_can into temp dataframe that is

(10-3):10 i.e. df_can(7:10) or df_can(8:11) etc

set the default variables
```

In [131]:

```
for i in range(window_length, df_can.shape[0]):
    tmp_df = candles[(i - window_length):i]
    direction = 0.0
    direction_long = False
    direction_short = False
    pivot_line = -1.0
```

Check Long position

check that dataframe has entries

set ind_current_pivot_down to -1

return the position in the tmp_df['pivot_down'] array where the values are true

In [132]:

```
if df_can.shape[0] > 0:
    ind_current_pivot_down = -1
    potential_pivots_down  = np.where(tmp_df['pivot_down'][:(-window_length)] ==
    print(potential_pivots_down)
```

```
[ 3 14 24 34 35]
```

if there are pivot points reset ind_current_pivot_down (i.e. add one)...

In [133]:

```
if len(potential_pivots_down) > 0:
    ind_current_pivot_down = potential_pivots_down[-1] + 1
```

The next if statement broken down...

In [144]:

```
if ind_current_pivot_down > -1:
    high_price_current_pivot_down = tmp_df['high'].iloc[ind_current_pivot_down -
    prev_highs = np.where((tmp_df['high'][:(ind_current_pivot_down - 1)] > high_
                tmp_df['pivot_down'][:(ind_current_pivot_down - 1)] == True))[0]
```

...get the value of the last value in the potentential_pivots_down array...

In [145]:

```
high_price_current_pivot_down = tmp_df['high'].iloc[ind_current_pivot_down - 1]
high_price_current_pivot_down
```

Out[145]:

```
95.65
```

...get satisfied bars (price level and pivot point status)

In [141]:

```
np.where((tmp_df['high'][:(ind_current_pivot_down - 1)] > high_price_current_piv
       & (tmp_df['pivot_down'][:(ind_current_pivot_down - 1)] == True))[0]
```

Out[141]:

```
array([ 3, 14])
```

If we have these bars then get additional info...

In [ ]:

```
if len(prev_highs) > 0:
    open_price_now  = tmp_df['open'].iat[-1]
    close_price_now = tmp_df['close'].iat[-1]

    high_price_prev_good_pivot_down = tmp_df['high'].iloc[prev_highs[-1]]
    ind_prev_good_pivot_down        = prev_highs[-1] + 1

    num_bars_between_pivot_down_points      = ind_current_pivot_down - ind_prev_
    num_bars_between_current_pivot_down_now = history_bars_length - ind_current_

```

Printed out to make it easier to follow...

In [153]:

```
print(open_price_now)
print(close_price_now)
print('-----')
print(high_price_prev_good_pivot_down)
print(ind_prev_good_pivot_down)
print('-----')
print(num_bars_between_pivot_down_points)
print(num_bars_between_current_pivot_down_now)
```

```
95.79
95.82
-----
95.89
15
-----
21
4
```

Calculate trend line and make the decision

In [159]:

```
dydx_ratio = (high_price_current_pivot_down - high_price_prev_good_pivot_down) /
pivot_line = high_price_current_pivot_down + dydx_ratio * num_bars_between_curre
if dydx_ratio < 0 and close_price_now > pivot_line > open_price_now:
    direction_long = True
    direction = 1.0

dydx_ratio
pivot_line
```

Out[159]:

```
95.60428571428572
```

In [156]:

```
1 direction_long
```

Out[156]:

```
False
```

In [157]:

```
1 direction
```

Out[157]:

```
0.0
```

In [ ]:

```
1
```