

# realtime-evolution-strategy

September 29, 2021

```
[1]: from sklearn.preprocessing import MinMaxScaler
import os
import numpy as np
import pandas as pd
import time
import random

import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

```
[2]: def softmax(z):
    assert len(z.shape) == 2
    s = np.max(z, axis=1)
    s = s[:, np.newaxis]
    e_x = np.exp(z - s)
    div = np.sum(e_x, axis=1)
    div = div[:, np.newaxis]
    return e_x / div
```

```
[3]: df = pd.read_csv('FB.csv')
df.head()
```

```
[3]:
```

|   | Date       | Open       | High       | Low        | Close      | Adj Close  | \ |
|---|------------|------------|------------|------------|------------|------------|---|
| 0 | 2018-05-23 | 182.500000 | 186.910004 | 182.179993 | 186.899994 | 186.899994 |   |
| 1 | 2018-05-24 | 185.880005 | 186.800003 | 185.029999 | 185.929993 | 185.929993 |   |
| 2 | 2018-05-25 | 186.020004 | 186.330002 | 184.449997 | 184.919998 | 184.919998 |   |
| 3 | 2018-05-29 | 184.339996 | 186.809998 | 183.710007 | 185.740005 | 185.740005 |   |
| 4 | 2018-05-30 | 186.539993 | 188.000000 | 185.250000 | 187.669998 | 187.669998 |   |

|   | Volume   |
|---|----------|
| 0 | 16628100 |
| 1 | 12354700 |
| 2 | 10965100 |
| 3 | 16398900 |
| 4 | 13736900 |

```
[4]: parameters = [df['Close'].tolist(), df['Volume'].tolist()]
```

```
[5]: def get_state(parameters, t, window_size = 20):
    outside = []
    d = t - window_size + 1
    for parameter in parameters:
        block = (
            parameter[d : t + 1]
            if d >= 0
            else -d * [parameter[0]] + parameter[0 : t + 1]
        )
    res = []
    for i in range(window_size - 1):
        res.append(block[i + 1] - block[i])
    for i in range(1, window_size, 1):
        res.append(block[i] - block[0])
    outside.append(res)
    return np.array(outside).reshape((1, -1))
```

Our output size from `get_state` is 38. In this notebook, I only use `Close` and `Volume` parameters, you can choose any parameters you want from your `DataFrame`.

After that, I want to add another parameters, my `inventory` size, mean of `inventory` and `capital`.

Let say for an example,

```
inventory_size = 1
mean_inventory = 0.5
capital = 2
last_state = 0
```

We have 3 actions,

1. 0 for do nothing.
2. 1 for buy.
3. 2 for sell.

```
[6]: inventory_size = 1
    mean_inventory = 0.5
    capital = 2
    concat_parameters = np.concatenate([get_state(parameters, 20), [[inventory_size,
                                                                    mean_inventory,
                                                                    capital]]],
    ↪axis = 1)
```

```
[7]: input_size = concat_parameters.shape[1]
    input_size
```

```
[7]: 79
```

```
[8]: class Deep_Evolution_Strategy:

    inputs = None

    def __init__(
        self, weights, reward_function, population_size, sigma, learning_rate
    ):
        self.weights = weights
        self.reward_function = reward_function
        self.population_size = population_size
        self.sigma = sigma
        self.learning_rate = learning_rate

    def _get_weight_from_population(self, weights, population):
        weights_population = []
        for index, i in enumerate(population):
            jittered = self.sigma * i
            weights_population.append(weights[index] + jittered)
        return weights_population

    def get_weights(self):
        return self.weights

    def train(self, epoch = 100, print_every = 1):
        lasttime = time.time()
        for i in range(epoch):
            population = []
            rewards = np.zeros(self.population_size)
            for k in range(self.population_size):
                x = []
                for w in self.weights:
                    x.append(np.random.randn(*w.shape))
                population.append(x)
            for k in range(self.population_size):
                weights_population = self._get_weight_from_population(
                    self.weights, population[k]
                )
                rewards[k] = self.reward_function(weights_population)
            rewards = (rewards - np.mean(rewards)) / (np.std(rewards) + 1e-7)
            for index, w in enumerate(self.weights):
                A = np.array([p[index] for p in population])
                self.weights[index] = (
                    w
                    + self.learning_rate
                    / (self.population_size * self.sigma)
                    * np.dot(A.T, rewards).T
                )
```

```

        if (i + 1) % print_every == 0:
            print(
                'iter %d. reward: %f'
                % (i + 1, self.reward_function(self.weights))
            )
        print('time taken to train:', time.time() - lasttime, 'seconds')

class Model:
    def __init__(self, input_size, layer_size, output_size):
        self.weights = [
            np.random.rand(input_size, layer_size)
            * np.sqrt(1 / (input_size + layer_size)),
            np.random.rand(layer_size, output_size)
            * np.sqrt(1 / (layer_size + output_size)),
            np.zeros((1, layer_size)),
            np.zeros((1, output_size)),
        ]

    def predict(self, inputs):
        feed = np.dot(inputs, self.weights[0]) + self.weights[-2]
        decision = np.dot(feed, self.weights[1]) + self.weights[-1]
        return decision

    def get_weights(self):
        return self.weights

    def set_weights(self, weights):
        self.weights = weights

```

```

[9]: class Agent:

    POPULATION_SIZE = 15
    SIGMA = 0.1
    LEARNING_RATE = 0.03

    def __init__(self, model, timeseries, skip, initial_money, real_trend, ↵
↵minmax):
        self.model = model
        self.timeseries = timeseries
        self.skip = skip
        self.real_trend = real_trend
        self.initial_money = initial_money
        self.es = Deep_Evolution_Strategy(
            self.model.get_weights(),
            self.get_reward,
            self.POPULATION_SIZE,
            self.SIGMA,

```

```

        self.LEARNING_RATE,
    )
    self.minmax = minmax
    self._initiate()

def _initiate(self):
    # i assume first index is the close value
    self.trend = self.timeseries[0]
    self._mean = np.mean(self.trend)
    self._std = np.std(self.trend)
    self._inventory = []
    self._capital = self.initial_money
    self._queue = []
    self._scaled_capital = self.minmax.transform([[self._capital, 2]])[0, 0]

def reset_capital(self, capital):
    if capital:
        self._capital = capital
        self._scaled_capital = self.minmax.transform([[self._capital, 2]])[0, 0]
        self._queue = []
        self._inventory = []

def trade(self, data):
    """
    you need to make sure the data is [close, volume]
    """
    scaled_data = self.minmax.transform([data])[0]
    real_close = data[0]
    close = scaled_data[0]
    if len(self._queue) >= window_size:
        self._queue.pop(0)
    self._queue.append(scaled_data)
    if len(self._queue) < window_size:
        return {
            'status': 'data not enough to trade',
            'action': 'fail',
            'balance': self._capital,
            'timestamp': str(datetime.now()),
        }
    state = self.get_state(
        window_size - 1,
        self._inventory,
        self._scaled_capital,
        timeseries = np.array(self._queue).T.tolist(),
    )
    action, prob = self.act_softmax(state)
    print(prob)

```

```

if action == 1 and self._scaled_capital >= close:
    self._inventory.append(close)
    self._scaled_capital -= close
    self._capital -= real_close
    return {
        'status': 'buy 1 unit, cost %f' % (real_close),
        'action': 'buy',
        'balance': self._capital,
        'timestamp': str(datetime.now()),
    }
elif action == 2 and len(self._inventory):
    bought_price = self._inventory.pop(0)
    self._scaled_capital += close
    self._capital += real_close
    scaled_bought_price = self.minmax.inverse_transform(
        [[bought_price, 2]]
    )[0, 0]
    try:
        invest = (
            (real_close - scaled_bought_price) / scaled_bought_price
        ) * 100
    except:
        invest = 0
    return {
        'status': 'sell 1 unit, price %f' % (real_close),
        'investment': invest,
        'gain': real_close - scaled_bought_price,
        'balance': self._capital,
        'action': 'sell',
        'timestamp': str(datetime.now()),
    }
else:
    return {
        'status': 'do nothing',
        'action': 'nothing',
        'balance': self._capital,
        'timestamp': str(datetime.now()),
    }

def change_data(self, timeseries, skip, initial_money, real_trend, minmax):
    self.timeseries = timeseries
    self.skip = skip
    self.initial_money = initial_money
    self.real_trend = real_trend
    self.minmax = minmax
    self._initiate()

```

```

def act(self, sequence):
    decision = self.model.predict(np.array(sequence))

    return np.argmax(decision[0])

def act_softmax(self, sequence):
    decision = self.model.predict(np.array(sequence))

    return np.argmax(decision[0]), softmax(decision)[0]

def get_state(self, t, inventory, capital, timeseries):
    state = get_state(timeseries, t)
    len_inventory = len(inventory)
    if len_inventory:
        mean_inventory = np.mean(inventory)
    else:
        mean_inventory = 0
    z_inventory = (mean_inventory - self._mean) / self._std
    z_capital = (capital - self._mean) / self._std
    concat_parameters = np.concatenate(
        [state, [[len_inventory, z_inventory, z_capital]]], axis = 1
    )
    return concat_parameters

def get_reward(self, weights):
    initial_money = self._scaled_capital
    starting_money = initial_money
    invests = []
    self.model.weights = weights
    inventory = []
    state = self.get_state(0, inventory, starting_money, self.timeseries)

    for t in range(0, len(self.trend) - 1, self.skip):
        action = self.act(state)
        if action == 1 and starting_money >= self.trend[t]:
            inventory.append(self.trend[t])
            starting_money -= self.trend[t]

        elif action == 2 and len(inventory):
            bought_price = inventory.pop(0)
            starting_money += self.trend[t]
            invest = ((self.trend[t] - bought_price) / bought_price) * 100
            invests.append(invest)

        state = self.get_state(
            t + 1, inventory, starting_money, self.timeseries
        )

```

```

invests = np.mean(invests)
if np.isnan(invests):
    invests = 0
score = (starting_money - initial_money) / initial_money * 100
return invests * 0.7 + score * 0.3

def fit(self, iterations, checkpoint):
    self.es.train(iterations, print_every = checkpoint)

def buy(self):
    initial_money = self._scaled_capital
    starting_money = initial_money

    real_initial_money = self.initial_money
    real_starting_money = self.initial_money
    inventory = []
    real_inventory = []
    state = self.get_state(0, inventory, starting_money, self.timeseries)
    states_sell = []
    states_buy = []

    for t in range(0, len(self.trend) - 1, self.skip):
        action, prob = self.act_softmax(state)
        print(t, prob)

        if action == 1 and starting_money >= self.trend[t] and t <
→(len(self.trend) - 1 - window_size):
            inventory.append(self.trend[t])
            real_inventory.append(self.real_trend[t])
            real_starting_money -= self.real_trend[t]
            starting_money -= self.trend[t]
            states_buy.append(t)
            print(
                'day %d: buy 1 unit at price %f, total balance %f'
                % (t, self.real_trend[t], real_starting_money)
            )

        elif action == 2 and len(inventory):
            bought_price = inventory.pop(0)
            real_bought_price = real_inventory.pop(0)
            starting_money += self.trend[t]
            real_starting_money += self.real_trend[t]
            states_sell.append(t)
            try:
                invest = (
                    (self.real_trend[t] - real_bought_price)
                    / real_bought_price

```



```

        ) * 100
    except:
        invest = 0
    print(
        'day %d, sell 1 unit at price %f, investment %f %%, total_
↪balance %f,'
        % (t, self.real_trend[t], invest, real_starting_money)
    )
    state = self.get_state(
        t + 1, inventory, starting_money, self.timeseries
    )

    invest = (
        (real_starting_money - real_initial_money) / real_initial_money
    ) * 100
    total_gains = real_starting_money - real_initial_money
    return states_buy, states_sell, total_gains, invest

```

```

[10]: stocks = [i for i in os.listdir(os.getcwd()) if '.csv' in i and not 'TWTR' in i]
stocks

```

```

[10]: ['CPRT.csv',
'AMD.csv',
'GWR.csv',
'MTDR.csv',
'GOOG.csv',
'TSLA.csv',
'FSV.csv',
'LB.csv',
'FB.csv',
'SINA.csv',
'LYFT.csv']

```

I want to train on all stocks I downloaded except for Twitter. I want to use Twitter for testing.

```

[11]: skip = 1
layer_size = 500
output_size = 3
window_size = 20

```

```

[12]: model = Model(input_size = input_size, layer_size = layer_size, output_size =
↪output_size)
agent = None

for no, stock in enumerate(stocks):
    print('training stock %s'%(stock))
    df = pd.read_csv(stock)

```

```

real_trend = df['Close'].tolist()
parameters = [df['Close'].tolist(), df['Volume'].tolist()]
minmax = MinMaxScaler(feature_range = (100, 200)).fit(np.array(parameters).
→T)
scaled_parameters = minmax.transform(np.array(parameters).T).T.tolist()
initial_money = np.max(parameters[0]) * 2

if no == 0:
    agent = Agent(model = model,
                  timeseries = scaled_parameters,
                  skip = skip,
                  initial_money = initial_money,
                  real_trend = real_trend,
                  minmax = minmax)
else:
    agent.change_data(timeseries = scaled_parameters,
                      skip = skip,
                      initial_money = initial_money,
                      real_trend = real_trend,
                      minmax = minmax)

agent.fit(iterations = 100, checkpoint = 10)
print()

```

```

training stock CPRT.csv
iter 10. reward: -1.247987
iter 20. reward: 6.464369
iter 30. reward: 20.931005
iter 40. reward: 17.746705
iter 50. reward: 18.576328
iter 60. reward: 21.053296
iter 70. reward: 19.513437
iter 80. reward: 34.468159
iter 90. reward: 18.841612
iter 100. reward: 30.144099
time taken to train: 17.304582357406616 seconds

```

```

training stock AMD.csv
iter 10. reward: 36.286963
iter 20. reward: 43.994438
iter 30. reward: 36.426547
iter 40. reward: 40.952264
iter 50. reward: 42.701169
iter 60. reward: 46.964220
iter 70. reward: 50.213138
iter 80. reward: 47.108041
iter 90. reward: 52.502404

```

iter 100. reward: 55.356372  
time taken to train: 18.030934810638428 seconds

training stock GWR.csv  
iter 10. reward: 21.827143  
iter 20. reward: 26.996406  
iter 30. reward: 28.478715  
iter 40. reward: 24.769332  
iter 50. reward: 32.276361  
iter 60. reward: 29.583129  
iter 70. reward: 32.912309  
iter 80. reward: 33.685202  
iter 90. reward: 35.657235  
iter 100. reward: 33.971033  
time taken to train: 17.878694772720337 seconds

training stock MTDR.csv  
iter 10. reward: 6.588120  
iter 20. reward: 9.963038  
iter 30. reward: 6.995521  
iter 40. reward: 11.924601  
iter 50. reward: 12.691655  
iter 60. reward: 13.768439  
iter 70. reward: 13.757468  
iter 80. reward: 17.086748  
iter 90. reward: 18.127015  
iter 100. reward: 17.695863  
time taken to train: 16.625855445861816 seconds

training stock GOOG.csv  
iter 10. reward: 14.863618  
iter 20. reward: 17.537896  
iter 30. reward: 18.676808  
iter 40. reward: 22.530143  
iter 50. reward: 24.339954  
iter 60. reward: 25.471286  
iter 70. reward: 27.379953  
iter 80. reward: 26.651974  
iter 90. reward: 26.503188  
iter 100. reward: 28.608727  
time taken to train: 18.292449712753296 seconds

training stock TSLA.csv  
iter 10. reward: -1.988759  
iter 20. reward: 3.583594  
iter 30. reward: 6.227101  
iter 40. reward: 7.969605  
iter 50. reward: 13.373963

iter 60. reward: 18.205339  
iter 70. reward: 19.686566  
iter 80. reward: 19.667429  
iter 90. reward: 20.270016  
iter 100. reward: 22.427184  
time taken to train: 17.09872031211853 seconds

training stock FSV.csv  
iter 10. reward: 18.520624  
iter 20. reward: 22.422610  
iter 30. reward: 23.608830  
iter 40. reward: 23.608830  
iter 50. reward: 23.507973  
iter 60. reward: 24.019357  
iter 70. reward: 24.429757  
iter 80. reward: 25.059607  
iter 90. reward: 25.248546  
iter 100. reward: 25.237088  
time taken to train: 17.251641750335693 seconds

training stock LB.csv  
iter 10. reward: -4.582982  
iter 20. reward: -4.592372  
iter 30. reward: -3.883796  
iter 40. reward: -3.076274  
iter 50. reward: -2.990688  
iter 60. reward: -0.984809  
iter 70. reward: -0.301800  
iter 80. reward: 1.037189  
iter 90. reward: 1.035139  
iter 100. reward: 1.982279  
time taken to train: 17.05810546875 seconds

training stock FB.csv  
iter 10. reward: -28.704470  
iter 20. reward: -17.753076  
iter 30. reward: -18.305663  
iter 40. reward: -17.788208  
iter 50. reward: -17.444941  
iter 60. reward: -17.783883  
iter 70. reward: -18.965700  
iter 80. reward: -18.308042  
iter 90. reward: -17.689998  
iter 100. reward: -17.506858  
time taken to train: 16.90356683731079 seconds

training stock SINA.csv  
iter 10. reward: -4.434994

```

iter 20. reward: 1.403997
iter 30. reward: 2.220121
iter 40. reward: 4.055608
iter 50. reward: 3.661211
iter 60. reward: 4.540796
iter 70. reward: 6.477745
iter 80. reward: 6.597851
iter 90. reward: 6.701629
iter 100. reward: 6.760706
time taken to train: 16.092115879058838 seconds

```

```

training stock LYFT.csv
iter 10. reward: 3.374927
iter 20. reward: 10.456719
iter 30. reward: 10.456719
iter 40. reward: 10.456719
iter 50. reward: 10.456719
iter 60. reward: 10.456719
iter 70. reward: 10.456719
iter 80. reward: 10.456719
iter 90. reward: 10.456719
iter 100. reward: 10.456719
time taken to train: 4.763254642486572 seconds

```

**0.0.1** If you saw the whole training session on certain stocks are negatives (like FB), means that, that stock markets are losing very bad

```

[13]: df = pd.read_csv('GOOG.csv')
real_trend = df['Close'].tolist()
parameters = [df['Close'].tolist(), df['Volume'].tolist()]
minmax = MinMaxScaler(feature_range = (100, 200)).fit(np.array(parameters).T)
scaled_parameters = minmax.transform(np.array(parameters).T).T.tolist()
initial_money = np.max(parameters[0]) * 2

agent.change_data(timeseries = scaled_parameters,
                  skip = skip,
                  initial_money = initial_money,
                  real_trend = real_trend,
                  minmax = minmax)

```

```

[14]: states_buy, states_sell, total_gains, invest = agent.buy()

```

```

0 [0. 0. 1.]
1 [0. 0. 1.]
2 [0. 0. 1.]
3 [1. 0. 0.]

```

4 [0. 0. 1.]  
 5 [2.88041485e-42 1.00000000e+00 0.00000000e+00]  
 day 5: buy 1 unit at price 1084.989990, total balance 1490.169922  
 6 [1. 0. 0.]  
 7 [0. 0. 1.]  
 day 7, sell 1 unit at price 1139.290039, investment 5.004659 %, total balance  
 2629.459961,  
 8 [0. 0. 1.]  
 9 [0. 0. 1.]  
 10 [0. 0. 1.]  
 11 [0. 0. 1.]  
 12 [0. 0. 1.]  
 13 [0. 0. 1.]  
 14 [0.00000000e+00 1.00000000e+00 1.7139084e-34]  
 day 14: buy 1 unit at price 1134.790039, total balance 1494.669922  
 15 [0. 0. 1.]  
 day 15, sell 1 unit at price 1152.119995, investment 1.527151 %, total balance  
 2646.789917,  
 16 [0.00000000e+000 3.38086219e-109 1.00000000e+000]  
 17 [0. 0. 1.]  
 18 [0. 1. 0.]  
 day 18: buy 1 unit at price 1168.060059, total balance 1478.729858  
 19 [0. 0. 1.]  
 day 19, sell 1 unit at price 1169.839966, investment 0.152381 %, total balance  
 2648.569824,  
 20 [0. 1. 0.]  
 day 20: buy 1 unit at price 1157.660034, total balance 1490.909790  
 21 [0. 1. 0.]  
 day 21: buy 1 unit at price 1155.479980, total balance 335.429810  
 22 [0. 1. 0.]  
 day 22: buy 1 unit at price 1124.810059, total balance -789.380249  
 23 [0. 1. 0.]  
 day 23: buy 1 unit at price 1118.459961, total balance -1907.840210  
 24 [7.52059788e-254 1.00000000e+000 0.00000000e+000]  
 25 [1. 0. 0.]  
 26 [1. 0. 0.]  
 27 [1. 0. 0.]  
 28 [0. 1. 0.]  
 29 [0. 1. 0.]  
 30 [0. 1. 0.]  
 31 [0.00000000e+00 1.00000000e+00 2.51869099e-15]  
 32 [0. 0. 1.]  
 day 32, sell 1 unit at price 1152.839966, investment -0.416363 %, total balance  
 -755.000244,  
 33 [1. 0. 0.]  
 34 [0. 0. 1.]  
 day 34, sell 1 unit at price 1183.479980, investment 2.423235 %, total balance  
 428.479736,

35 [1. 0. 0.]  
 36 [1. 0. 0.]  
 37 [1. 0. 0.]  
 38 [1. 0. 0.]  
 39 [1. 0. 0.]  
 40 [1. 0. 0.]  
 41 [1. 0. 0.]  
 42 [1. 0. 0.]  
 43 [0. 1. 0.]  
 day 43: buy 1 unit at price 1263.699951, total balance -835.220215  
 44 [0. 1. 0.]  
 45 [0. 1. 0.]  
 46 [0. 1. 0.]  
 47 [0. 1. 0.]  
 48 [0. 1. 0.]  
 49 [0. 1. 0.]  
 50 [0. 1. 0.]  
 51 [0. 1. 0.]  
 52 [0. 1. 0.]  
 53 [0. 1. 0.]  
 54 [0. 1. 0.]  
 55 [0. 0. 1.]  
 day 55, sell 1 unit at price 1237.609985, investment 10.028353 %, total balance 402.389770,  
 56 [1. 0. 0.]  
 57 [0. 1. 0.]  
 day 57: buy 1 unit at price 1242.099976, total balance -839.710206  
 58 [0. 1. 0.]  
 59 [0. 1. 0.]  
 60 [1. 0. 0.]  
 61 [1. 0. 0.]  
 62 [1. 0. 0.]  
 63 [1. 0. 0.]  
 64 [1. 0. 0.]  
 65 [0. 1. 0.]  
 66 [0. 1. 0.]  
 67 [1. 0. 0.]  
 68 [1. 0. 0.]  
 69 [1. 0. 0.]  
 70 [0. 0. 1.]  
 day 70, sell 1 unit at price 1218.189941, investment 8.916723 %, total balance 378.479735,  
 71 [1. 0. 0.]  
 72 [1. 0. 0.]  
 73 [1. 0. 0.]  
 74 [1. 0. 0.]  
 75 [0. 0. 1.]  
 day 75, sell 1 unit at price 1164.640015, investment -7.838881 %, total balance

1543.119750,  
 76 [0. 0. 1.]  
 day 76, sell 1 unit at price 1177.359985, investment -5.212140 %, total balance  
 2720.479735,  
 77 [1.00000000e+000 1.71349693e-284 0.00000000e+000]  
 78 [0. 0. 1.]  
 79 [0. 1. 0.]  
 day 79: buy 1 unit at price 1172.530029, total balance 1547.949706  
 80 [0. 0. 1.]  
 day 80, sell 1 unit at price 1156.050049, investment -1.405506 %, total balance  
 2703.999755,  
 81 [0. 0. 1.]  
 82 [0. 0. 1.]  
 83 [0. 0. 1.]  
 84 [0. 0. 1.]  
 85 [0. 0. 1.]  
 86 [0. 0. 1.]  
 87 [1. 0. 0.]  
 88 [0. 0. 1.]  
 89 [1. 0. 0.]  
 90 [0. 0. 1.]  
 91 [0. 0. 1.]  
 92 [0. 0. 1.]  
 93 [0. 1. 0.]  
 day 93: buy 1 unit at price 1168.189941, total balance 1535.809814  
 94 [1. 0. 0.]  
 95 [0. 1. 0.]  
 day 95: buy 1 unit at price 1148.969971, total balance 386.839843  
 96 [0. 1. 0.]  
 day 96: buy 1 unit at price 1138.819946, total balance -751.980103  
 97 [0. 1. 0.]  
 day 97: buy 1 unit at price 1081.219971, total balance -1833.200074  
 98 [2.58002336e-293 0.00000000e+000 1.00000000e+000]  
 day 98, sell 1 unit at price 1079.319946, investment -7.607495 %, total balance  
 -753.880128,  
 99 [0. 1. 0.]  
 day 99: buy 1 unit at price 1110.079956, total balance -1863.960084  
 100 [0. 1. 0.]  
 101 [0. 0. 1.]  
 day 101, sell 1 unit at price 1121.280029, investment -2.409980 %, total balance  
 -742.680055,  
 102 [0. 0. 1.]  
 day 102, sell 1 unit at price 1115.689941, investment -2.031050 %, total balance  
 373.009886,  
 103 [1. 0. 0.]  
 104 [0. 0. 1.]  
 day 104, sell 1 unit at price 1096.459961, investment 1.409518 %, total balance  
 1469.469847,



105 [0. 0. 1.]  
 day 105, sell 1 unit at price 1101.160034, investment -0.803539 %, total balance  
 2570.629881,  
 106 [0. 0. 1.]  
 107 [0. 0. 1.]  
 108 [0. 0. 1.]  
 109 [0. 0. 1.]  
 110 [1. 0. 0.]  
 111 [0. 0. 1.]  
 112 [1. 0. 0.]  
 113 [0. 0. 1.]  
 114 [0. 0. 1.]  
 115 [0. 0. 1.]  
 116 [0. 1. 0.]  
 day 116: buy 1 unit at price 1055.810059, total balance 1514.819822  
 117 [0. 1. 0.]  
 day 117: buy 1 unit at price 1093.390015, total balance 421.429807  
 118 [0. 0. 1.]  
 day 118, sell 1 unit at price 1082.400024, investment 2.518442 %, total balance  
 1503.829831,  
 119 [0. 0. 1.]  
 day 119, sell 1 unit at price 1066.150024, investment -2.491333 %, total balance  
 2569.979855,  
 120 [1. 0. 0.]  
 121 [0. 0. 1.]  
 122 [0. 0. 1.]  
 123 [0. 0. 1.]  
 124 [1. 0. 0.]  
 125 [1. 0. 0.]  
 126 [1. 0. 0.]  
 127 [1. 0. 0.]  
 128 [1. 0. 0.]  
 129 [1. 0. 0.]  
 130 [1. 0. 0.]  
 131 [1. 0. 0.]  
 132 [0. 0. 1.]  
 133 [0. 0. 1.]  
 134 [1. 0. 0.]  
 135 [0. 1. 0.]  
 day 135: buy 1 unit at price 1050.819946, total balance 1519.159909  
 136 [1. 0. 0.]  
 137 [0. 0. 1.]  
 day 137, sell 1 unit at price 1036.579956, investment -1.355131 %, total balance  
 2555.739865,  
 138 [0. 0. 1.]  
 139 [0. 0. 1.]  
 140 [0. 1. 0.]  
 day 140: buy 1 unit at price 1063.680054, total balance 1492.059811

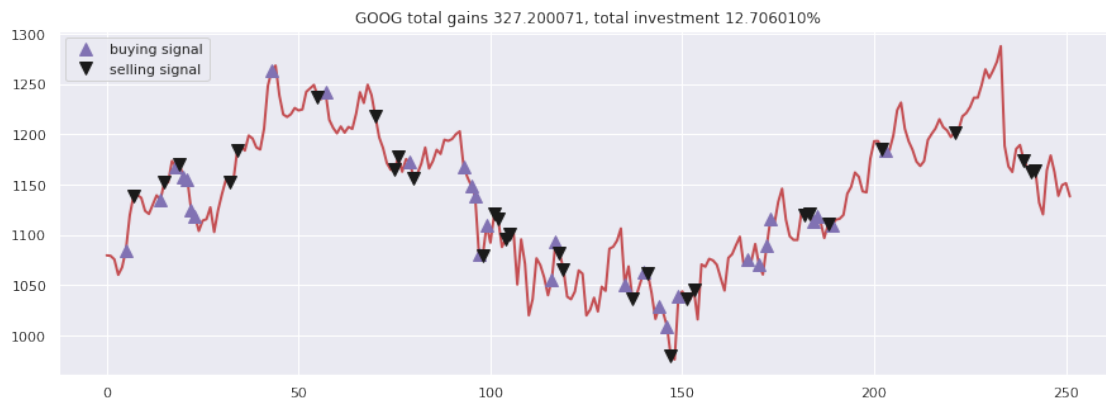
141 [0. 0. 1.]  
 day 141, sell 1 unit at price 1061.900024, investment -0.167346 %, total balance  
 2553.959835,  
 142 [0. 0. 1.]  
 143 [0. 0. 1.]  
 144 [0. 1. 0.]  
 day 144: buy 1 unit at price 1028.709961, total balance 1525.249874  
 145 [1. 0. 0.]  
 146 [0. 1. 0.]  
 day 146: buy 1 unit at price 1009.409973, total balance 515.839901  
 147 [0. 0. 1.]  
 day 147, sell 1 unit at price 979.539978, investment -4.779771 %, total balance  
 1495.379879,  
 148 [1. 0. 0.]  
 149 [0. 1. 0.]  
 day 149: buy 1 unit at price 1039.459961, total balance 455.919918  
 150 [1. 0. 0.]  
 151 [0. 0. 1.]  
 day 151, sell 1 unit at price 1037.079956, investment 2.741204 %, total balance  
 1492.999874,  
 152 [1. 0. 0.]  
 153 [0. 0. 1.]  
 day 153, sell 1 unit at price 1045.849976, investment 0.614744 %, total balance  
 2538.849850,  
 154 [0. 0. 1.]  
 155 [0. 0. 1.]  
 156 [1. 0. 0.]  
 157 [0. 0. 1.]  
 158 [0. 0. 1.]  
 159 [0. 0. 1.]  
 160 [0. 0. 1.]  
 161 [1. 0. 0.]  
 162 [1. 0. 0.]  
 163 [1. 0. 0.]  
 164 [1. 0. 0.]  
 165 [1. 0. 0.]  
 166 [1. 0. 0.]  
 167 [0. 1. 0.]  
 day 167: buy 1 unit at price 1075.569946, total balance 1463.279904  
 168 [1. 0. 0.]  
 169 [1. 0. 0.]  
 170 [0. 1. 0.]  
 day 170: buy 1 unit at price 1070.079956, total balance 393.199948  
 171 [1. 0. 0.]  
 172 [0. 1. 0.]  
 day 172: buy 1 unit at price 1089.060059, total balance -695.860111  
 173 [0. 1. 0.]  
 day 173: buy 1 unit at price 1116.369995, total balance -1812.230106

174 [1. 0. 0.]  
 175 [1. 0. 0.]  
 176 [1. 0. 0.]  
 177 [1. 0. 0.]  
 178 [0. 1. 0.]  
 179 [0. 1. 0.]  
 180 [0. 1. 0.]  
 181 [0. 1. 0.]  
 182 [0. 0. 1.]  
 day 182, sell 1 unit at price 1120.160034, investment 4.145717 %, total balance  
 -692.070072,  
 183 [0. 0. 1.]  
 day 183, sell 1 unit at price 1121.670044, investment 4.821143 %, total balance  
 429.599972,  
 184 [0. 1. 0.]  
 day 184: buy 1 unit at price 1113.650024, total balance -684.050052  
 185 [0. 1. 0.]  
 day 185: buy 1 unit at price 1118.560059, total balance -1802.610111  
 186 [0.0000000e+00 1.0000000e+00 3.9040329e-80]  
 187 [0. 1. 0.]  
 188 [0. 0. 1.]  
 day 188, sell 1 unit at price 1110.369995, investment 1.956727 %, total balance  
 -692.240116,  
 189 [0. 1. 0.]  
 day 189: buy 1 unit at price 1109.400024, total balance -1801.640140  
 190 [0. 1. 0.]  
 191 [1. 0. 0.]  
 192 [1. 0. 0.]  
 193 [0. 1. 0.]  
 194 [1. 0. 0.]  
 195 [1. 0. 0.]  
 196 [1. 0. 0.]  
 197 [1. 0. 0.]  
 198 [0. 1. 0.]  
 199 [0. 1. 0.]  
 200 [1. 0. 0.]  
 201 [1.00000000e+00 0.00000000e+00 1.11009789e-17]  
 202 [0. 0. 1.]  
 day 202, sell 1 unit at price 1185.550049, investment 6.196875 %, total balance  
 -616.090091,  
 203 [0. 1. 0.]  
 day 203: buy 1 unit at price 1184.459961, total balance -1800.550052  
 204 [0. 1. 0.]  
 205 [0. 1. 0.]  
 206 [0. 1. 0.]  
 207 [0. 1. 0.]  
 208 [0. 1. 0.]  
 209 [0. 1. 0.]

210 [0. 1. 0.]  
 211 [0. 1. 0.]  
 212 [0. 1. 0.]  
 213 [0. 1. 0.]  
 214 [0. 1. 0.]  
 215 [0. 1. 0.]  
 216 [0. 1. 0.]  
 217 [0. 1. 0.]  
 218 [0. 1. 0.]  
 219 [1. 0. 0.]  
 220 [1. 0. 0.]  
 221 [0. 0. 1.]  
 day 221, sell 1 unit at price 1202.160034, investment 7.947740 %, total balance  
 -598.390018,  
 222 [1. 0. 0.]  
 223 [1. 0. 0.]  
 224 [1. 0. 0.]  
 225 [1. 0. 0.]  
 226 [1. 0. 0.]  
 227 [1. 0. 0.]  
 228 [1. 0. 0.]  
 229 [1. 0. 0.]  
 230 [1. 0. 0.]  
 231 [0. 1. 0.]  
 232 [0. 1. 0.]  
 233 [0. 1. 0.]  
 234 [1. 0. 0.]  
 235 [1. 0. 0.]  
 236 [0. 1. 0.]  
 237 [0. 1. 0.]  
 238 [0. 1. 0.]  
 239 [0. 0. 1.]  
 day 239, sell 1 unit at price 1174.099976, investment 4.965305 %, total balance  
 575.709958,  
 240 [0. 1. 0.]  
 241 [0. 0. 1.]  
 day 241, sell 1 unit at price 1162.380005, investment 4.775553 %, total balance  
 1738.089963,  
 242 [0. 0. 1.]  
 day 242, sell 1 unit at price 1164.270020, investment -1.704569 %, total balance  
 2902.359983,  
 243 [0. 1. 0.]  
 244 [0. 0. 1.]  
 245 [0. 0. 1.]  
 246 [0. 0. 1.]  
 247 [0. 0. 1.]  
 248 [0. 0. 1.]  
 249 [1. 0. 0.]

250 [0. 0. 1.]

```
[15]: fig = plt.figure(figsize = (15, 5))
plt.plot(df['Close'], color='r', lw=2.)
plt.plot(df['Close'], '^', markersize=10, color='m', label = 'buying signal',
↳markevery = states_buy)
plt.plot(df['Close'], 'v', markersize=10, color='k', label = 'selling signal',
↳markevery = states_sell)
plt.title('GOOG total gains %f, total investment %f%%'%(total_gains, invest))
plt.legend()
plt.show()
```



```
[16]: df = pd.read_csv('TWTR.csv')
real_trend = df['Close'].tolist()
parameters = [df['Close'].tolist(), df['Volume'].tolist()]
minmax = MinMaxScaler(feature_range = (100, 200)).fit(np.array(parameters).T)
scaled_parameters = minmax.transform(np.array(parameters).T).T.tolist()
initial_money = np.max(parameters[0]) * 2

agent.change_data(timeseries = scaled_parameters,
                  skip = skip,
                  initial_money = initial_money,
                  real_trend = real_trend,
                  minmax = minmax)

states_buy, states_sell, total_gains, invest = agent.buy()
```

0 [0. 0. 1.]  
1 [0. 0. 1.]  
2 [0. 0. 1.]  
3 [0. 0. 1.]  
4 [0. 0. 1.]  
5 [0. 0. 1.]

6 [0. 0. 1.]  
 7 [0. 0. 1.]  
 8 [0. 0. 1.]  
 9 [1. 0. 0.]  
 10 [1. 0. 0.]  
 11 [0. 1. 0.]  
 day 11: buy 1 unit at price 41.209999, total balance 52.309997  
 12 [0. 1. 0.]  
 day 12: buy 1 unit at price 41.419998, total balance 10.889999  
 13 [0. 0. 1.]  
 day 13, sell 1 unit at price 43.490002, investment 5.532645 %, total balance 54.380001,  
 14 [0. 0. 1.]  
 day 14, sell 1 unit at price 44.070000, investment 6.397881 %, total balance 98.450001,  
 15 [0. 1. 0.]  
 day 15: buy 1 unit at price 46.759998, total balance 51.690003  
 16 [0. 0. 1.]  
 day 16, sell 1 unit at price 45.799999, investment -2.053035 %, total balance 97.490002,  
 17 [0. 0. 1.]  
 18 [0. 1. 0.]  
 day 18: buy 1 unit at price 44.950001, total balance 52.540001  
 19 [0. 1. 0.]  
 day 19: buy 1 unit at price 46.130001, total balance 6.410000  
 20 [0. 1. 0.]  
 21 [0. 1. 0.]  
 22 [0. 1. 0.]  
 23 [0. 1. 0.]  
 24 [0. 1. 0.]  
 25 [0. 1. 0.]  
 26 [0. 1. 0.]  
 27 [1. 0. 0.]  
 28 [1. 0. 0.]  
 29 [0. 1. 0.]  
 30 [0. 1. 0.]  
 31 [4.84761617e-08 9.99999952e-01 0.00000000e+00]  
 32 [1. 0. 0.]  
 33 [0. 1. 0.]  
 34 [1. 0. 0.]  
 35 [1. 0. 0.]  
 36 [0. 0. 1.]  
 day 36, sell 1 unit at price 44.259998, investment -1.535046 %, total balance 50.669998,  
 37 [0. 1. 0.]  
 day 37: buy 1 unit at price 44.709999, total balance 5.959999  
 38 [0. 0. 1.]  
 day 38, sell 1 unit at price 43.340000, investment -6.048127 %, total balance

49.299999,  
 39 [0. 0. 1.]  
 day 39, sell 1 unit at price 43.439999, investment -2.840528 %, total balance  
 92.739998,  
 40 [0. 1. 0.]  
 day 40: buy 1 unit at price 43.419998, total balance 49.320000  
 41 [1. 0. 0.]  
 42 [1. 0. 0.]  
 43 [0. 0. 1.]  
 day 43, sell 1 unit at price 44.220001, investment 1.842476 %, total balance  
 93.540001,  
 44 [0. 0. 1.]  
 45 [1. 0. 0.]  
 46 [1. 0. 0.]  
 47 [0. 1. 0.]  
 day 47: buy 1 unit at price 31.870001, total balance 61.670000  
 48 [1. 0. 0.]  
 49 [0. 0. 1.]  
 day 49, sell 1 unit at price 32.820000, investment 2.980857 %, total balance  
 94.490000,  
 50 [1. 0. 0.]  
 51 [0. 0. 1.]  
 52 [0. 0. 1.]  
 53 [0. 0. 1.]  
 54 [0. 0. 1.]  
 55 [0. 0. 1.]  
 56 [0. 0. 1.]  
 57 [0. 0. 1.]  
 58 [0. 0. 1.]  
 59 [0. 0. 1.]  
 60 [0. 0. 1.]  
 61 [0. 0. 1.]  
 62 [1. 0. 0.]  
 63 [0. 0. 1.]  
 64 [1. 0. 0.]  
 65 [1. 0. 0.]  
 66 [1. 0. 0.]  
 67 [1. 0. 0.]  
 68 [1.00000000e+000 0.00000000e+000 5.88236875e-246]  
 69 [0. 0.30331368 0.69668632]  
 70 [1. 0. 0.]  
 71 [1. 0. 0.]  
 72 [1. 0. 0.]  
 73 [0. 1. 0.]  
 day 73: buy 1 unit at price 30.809999, total balance 63.680001  
 74 [0. 1. 0.]  
 day 74: buy 1 unit at price 30.490000, total balance 33.190001  
 75 [1. 0. 0.]

76 [0. 1. 0.]  
 day 76: buy 1 unit at price 30.889999, total balance 2.300002  
 77 [0. 1. 0.]  
 78 [0. 1. 0.]  
 79 [0. 1. 0.]  
 80 [0. 0. 1.]  
 day 80, sell 1 unit at price 28.860001, investment -6.329108 %, total balance 31.160003,  
 81 [0.00000000e+000 1.00000000e+000 8.40150266e-172]  
 day 81: buy 1 unit at price 29.219999, total balance 1.940004  
 82 [0. 0. 1.]  
 day 82, sell 1 unit at price 29.520000, investment -3.181371 %, total balance 31.460004,  
 83 [1.00000000e+000 0.00000000e+000 1.68749158e-251]  
 84 [0. 0. 1.]  
 day 84, sell 1 unit at price 28.500000, investment -7.737129 %, total balance 59.960004,  
 85 [0. 0. 1.]  
 day 85, sell 1 unit at price 28.600000, investment -2.121831 %, total balance 88.560004,  
 86 [0. 0. 1.]  
 87 [0. 0. 1.]  
 88 [0. 0. 1.]  
 89 [1. 0. 0.]  
 90 [0. 0. 1.]  
 91 [1. 0. 0.]  
 92 [1. 0. 0.]  
 93 [1. 0. 0.]  
 94 [1. 0. 0.]  
 95 [0. 0. 1.]  
 96 [1.49596746e-208 0.00000000e+000 1.00000000e+000]  
 97 [0. 0. 1.]  
 98 [0. 0. 1.]  
 99 [1. 0. 0.]  
 100 [0. 0. 1.]  
 101 [0. 0. 1.]  
 102 [0. 0. 1.]  
 103 [1. 0. 0.]  
 104 [0. 0. 1.]  
 105 [0. 0. 1.]  
 106 [0. 0. 1.]  
 107 [0. 0. 1.]  
 108 [1. 0. 0.]  
 109 [1. 0. 0.]  
 110 [0. 0. 1.]  
 111 [0. 0. 1.]  
 112 [0. 1. 0.]  
 day 112: buy 1 unit at price 34.750000, total balance 53.810004



113 [0. 1. 0.]  
 day 113: buy 1 unit at price 34.619999, total balance 19.190005  
 114 [0. 0. 1.]  
 day 114, sell 1 unit at price 34.299999, investment -1.294967 %, total balance 53.490004,  
 115 [0. 0. 1.]  
 day 115, sell 1 unit at price 34.020000, investment -1.733099 %, total balance 87.510004,  
 116 [0. 1. 0.]  
 day 116: buy 1 unit at price 34.419998, total balance 53.090006  
 117 [0. 1. 0.]  
 day 117: buy 1 unit at price 34.990002, total balance 18.100004  
 118 [0. 1. 0.]  
 119 [0. 1. 0.]  
 120 [0. 1. 0.]  
 121 [0. 1. 0.]  
 122 [0. 1. 0.]  
 123 [0. 1. 0.]  
 124 [0. 1. 0.]  
 125 [0. 1. 0.]  
 126 [0. 1. 0.]  
 127 [1. 0. 0.]  
 128 [1. 0. 0.]  
 129 [1.000000000e+000 3.87913053e-131 0.000000000e+000]  
 130 [1. 0. 0.]  
 131 [1. 0. 0.]  
 132 [1. 0. 0.]  
 133 [1. 0. 0.]  
 134 [0. 0. 1.]  
 day 134, sell 1 unit at price 33.660000, investment -2.208013 %, total balance 51.760004,  
 135 [1. 0. 0.]  
 136 [0. 0. 1.]  
 day 136, sell 1 unit at price 32.959999, investment -5.801666 %, total balance 84.720003,  
 137 [0. 0. 1.]  
 138 [0. 0. 1.]  
 139 [0. 0. 1.]  
 140 [0. 0. 1.]  
 141 [1.000000000e+000 2.08795633e-032 2.67176100e-180]  
 142 [0. 0. 1.]  
 143 [0. 0. 1.]  
 144 [0. 0. 1.]  
 145 [0. 1. 0.]  
 day 145: buy 1 unit at price 32.930000, total balance 51.790003  
 146 [1. 0. 0.]  
 147 [0. 1. 0.]  
 day 147: buy 1 unit at price 27.309999, total balance 24.480004

148 [0. 1. 0.]  
 day 148: buy 1 unit at price 26.450001, total balance -1.969997  
 149 [0. 1. 0.]  
 150 [0. 0. 1.]  
 day 150, sell 1 unit at price 28.680000, investment -12.906165 %, total balance  
 26.710003,  
 151 [0. 1. 0.]  
 day 151: buy 1 unit at price 28.430000, total balance -1.719997  
 152 [0. 1. 0.]  
 153 [0. 0. 1.]  
 day 153, sell 1 unit at price 28.809999, investment 5.492494 %, total balance  
 27.090002,  
 154 [0. 0. 1.]  
 day 154, sell 1 unit at price 27.990000, investment 5.822302 %, total balance  
 55.080002,  
 155 [0. 0. 1.]  
 day 155, sell 1 unit at price 29.950001, investment 5.346469 %, total balance  
 85.030003,  
 156 [0. 0. 1.]  
 157 [0. 0. 1.]  
 158 [0. 0. 1.]  
 159 [0. 0. 1.]  
 160 [1. 0. 0.]  
 161 [1. 0. 0.]  
 162 [1.00000000e+00 0.00000000e+00 1.18347919e-79]  
 163 [0. 0. 1.]  
 164 [1. 0. 0.]  
 165 [1. 0. 0.]  
 166 [0. 1. 0.]  
 day 166: buy 1 unit at price 32.250000, total balance 52.780003  
 167 [0. 1. 0.]  
 day 167: buy 1 unit at price 30.969999, total balance 21.810004  
 168 [1.00000000e+000 3.75718032e-277 0.00000000e+000]  
 169 [0. 1. 0.]  
 day 169: buy 1 unit at price 32.900002, total balance -11.089998  
 170 [0. 1. 0.]  
 171 [0. 1. 0.]  
 172 [0. 1. 0.]  
 173 [0. 1. 0.]  
 174 [0. 1. 0.]  
 175 [0. 1. 0.]  
 176 [0. 1. 0.]  
 177 [0. 0. 1.]  
 day 177, sell 1 unit at price 34.160000, investment 5.922481 %, total balance  
 23.070002,  
 178 [1. 0. 0.]  
 179 [1. 0. 0.]  
 180 [0. 1. 0.]

day 180: buy 1 unit at price 30.230000, total balance -7.159998  
 181 [0. 1. 0.]  
 182 [0. 0. 1.]  
 day 182, sell 1 unit at price 31.120001, investment 0.484346 %, total balance 23.960003,  
 183 [0. 0. 1.]  
 day 183, sell 1 unit at price 30.959999, investment -5.896665 %, total balance 54.920002,  
 184 [0. 0. 1.]  
 day 184, sell 1 unit at price 31.230000, investment 3.307972 %, total balance 86.150002,  
 185 [0. 0. 1.]  
 186 [0. 0. 1.]  
 187 [0. 0. 1.]  
 188 [0. 0. 1.]  
 189 [0. 0. 1.]  
 190 [0. 0. 1.]  
 191 [0. 0. 1.]  
 192 [0. 0. 1.]  
 193 [1. 0. 0.]  
 194 [0. 0. 1.]  
 195 [0. 0. 1.]  
 196 [1. 0. 0.]  
 197 [1. 0. 0.]  
 198 [1. 0. 0.]  
 199 [0. 1. 0.]  
 day 199: buy 1 unit at price 30.870001, total balance 55.280001  
 200 [1. 0. 0.]  
 201 [1. 0. 0.]  
 202 [0. 0. 1.]  
 day 202, sell 1 unit at price 31.030001, investment 0.518303 %, total balance 86.310002,  
 203 [0. 0. 1.]  
 204 [0. 0. 1.]  
 205 [0. 0. 1.]  
 206 [1.000000000e+000 3.78767981e-200 0.000000000e+000]  
 207 [1. 0. 0.]  
 208 [0. 0. 1.]  
 209 [1. 0. 0.]  
 210 [0. 1. 0.]  
 day 210: buy 1 unit at price 33.060001, total balance 53.250001  
 211 [1. 0. 0.]  
 212 [0. 1. 0.]  
 day 212: buy 1 unit at price 32.869999, total balance 20.380002  
 213 [1. 0. 0.]  
 214 [0. 1. 0.]  
 215 [0. 0. 1.]  
 day 215, sell 1 unit at price 33.750000, investment 2.087111 %, total balance

```

54.130002,
216 [0. 1. 0.]
day 216: buy 1 unit at price 34.380001, total balance 19.750001
217 [0. 1. 0.]
218 [0. 1. 0.]
219 [0. 1. 0.]
220 [0. 1. 0.]
221 [0. 1. 0.]
222 [0. 1. 0.]
223 [0. 1. 0.]
224 [0. 1. 0.]
225 [1. 0. 0.]
226 [1.00000000e+000 1.79166683e-113 0.00000000e+000]
227 [1. 0. 0.]
228 [0. 1. 0.]
229 [1. 0. 0.]
230 [1. 0. 0.]
231 [0. 1. 0.]
232 [0. 1. 0.]
233 [0. 1. 0.]
234 [0. 1. 0.]
235 [0. 1. 0.]
236 [0. 0. 1.]
day 236, sell 1 unit at price 39.950001, investment 21.539404 %, total balance
59.700002,
237 [0.00000000e+000 3.52341905e-242 1.00000000e+000]
day 237, sell 1 unit at price 40.799999, investment 18.673641 %, total balance
100.500001,
238 [0. 1. 0.]
239 [1. 0. 0.]
240 [0. 1. 0.]
241 [0. 1. 0.]
242 [0. 1. 0.]
243 [0. 1. 0.]
244 [0. 1. 0.]
245 [0. 1. 0.]
246 [0. 1. 0.]
247 [0. 1. 0.]
248 [1. 0. 0.]
249 [1. 0. 0.]
250 [0. 1. 0.]

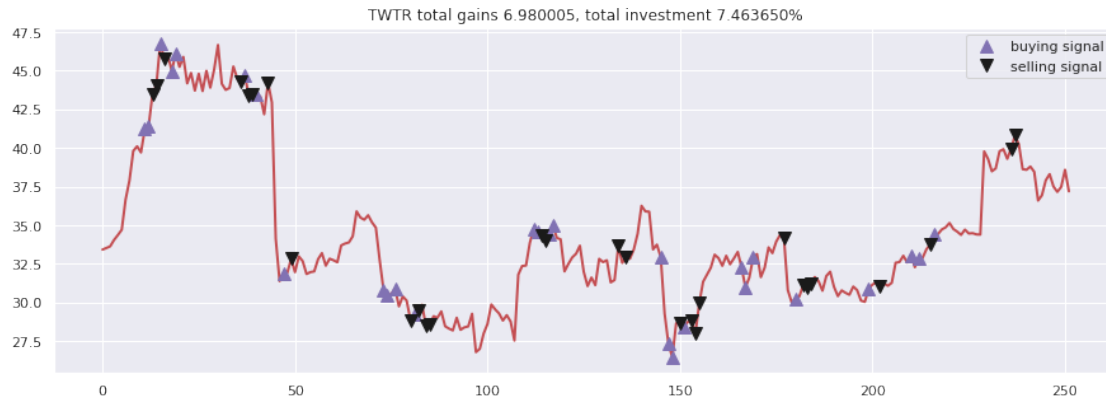
```

```

[17]: fig = plt.figure(figsize = (15, 5))
plt.plot(df['Close'], color='r', lw=2.)
plt.plot(df['Close'], '^', markersize=10, color='m', label = 'buying signal',
↪markevery = states_buy)

```

```
plt.plot(df['Close'], 'v', markersize=10, color='k', label = 'selling signal',
        ⇨markevery = states_sell)
plt.title('TWTR total gains %f, total investment %f%%'%(total_gains, invest))
plt.legend()
plt.show()
```



```
[18]: from datetime import datetime

volume = df['Volume'].tolist()

for i in range(100):
    print(agent.trade([real_trend[i], volume[i]]))
```

```
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247020'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247175'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247287'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247427'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247529'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247611'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247683'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247790'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247877'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.247947'}
```

```

{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248107'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248180'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248251'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248322'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248392'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248463'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248535'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248605'}
{'status': 'data not enough to trade', 'action': 'fail', 'balance':
93.51999599999999, 'timestamp': '2019-08-31 02:39:42.248674'}
[0. 1. 0.]
{'status': 'buy 1 unit, cost 46.130001', 'action': 'buy', 'balance':
47.38999499999999, 'timestamp': '2019-08-31 02:39:42.251303'}
[0. 1. 0.]
{'status': 'buy 1 unit, cost 45.240002', 'action': 'buy', 'balance':
2.1499929999999988, 'timestamp': '2019-08-31 02:39:42.251959'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.252573'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.253071'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.253555'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.254131'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.254743'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.255300'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.255858'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.256398'}

```

```

[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.256922'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.257477'}
[1.51020718e-59 1.00000000e+00 0.00000000e+00]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.258040'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.258505'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.259010'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.259559'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 2.1499929999999988,
'timestamp': '2019-08-31 02:39:42.260065'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 44.259998', 'investment': -4.053767525389831,
'gain': -1.8700030000000004, 'balance': 46.4099909999999984, 'action': 'sell',
'timestamp': '2019-08-31 02:39:42.260639'}
[0. 1. 0.]
{'status': 'buy 1 unit, cost 44.709999', 'action': 'buy', 'balance':
1.69999199999999875, 'timestamp': '2019-08-31 02:39:42.261195'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 43.340000', 'investment': -4.199827400538136,
'gain': -1.9000020000000006, 'balance': 45.039991999999999, 'action': 'sell',
'timestamp': '2019-08-31 02:39:42.261827'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 43.439999', 'investment': -2.8405279096517004,
'gain': -1.2699999999999996, 'balance': 88.479990999999998, 'action': 'sell',
'timestamp': '2019-08-31 02:39:42.262354'}
[0. 1. 0.]
{'status': 'buy 1 unit, cost 43.419998', 'action': 'buy', 'balance':
45.0599929999999984, 'timestamp': '2019-08-31 02:39:42.262827'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 45.0599929999999984,
'timestamp': '2019-08-31 02:39:42.263297'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 45.0599929999999984,
'timestamp': '2019-08-31 02:39:42.263789'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 44.220001', 'investment': 1.8424759024632011,
'gain': 0.80000300000000038, 'balance': 89.279993999999999, 'action': 'sell',

```

```

'timestamp': '2019-08-31 02:39:42.264333'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 89.27999399999999,
'timestamp': '2019-08-31 02:39:42.264848'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 89.27999399999999,
'timestamp': '2019-08-31 02:39:42.265265'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 89.27999399999999,
'timestamp': '2019-08-31 02:39:42.265720'}
[0. 1. 0.]
{'status': 'buy 1 unit, cost 31.870001', 'action': 'buy', 'balance':
57.409992999999986, 'timestamp': '2019-08-31 02:39:42.266187'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 57.409992999999986,
'timestamp': '2019-08-31 02:39:42.266665'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 32.820000', 'investment': 2.980856511425896,
'gain': 0.9499989999999983, 'balance': 90.22999299999998, 'action': 'sell',
'timestamp': '2019-08-31 02:39:42.267213'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.267673'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.268178'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.268595'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.269065'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.269536'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.269989'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.270436'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.270887'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.22999299999998,
'timestamp': '2019-08-31 02:39:42.271333'}
[0. 0. 1.]

```



```

{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.271783'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.272248'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.272705'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.273157'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.273620'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.274058'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.274518'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.274966'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.275422'}
[1.00000000e+000 0.00000000e+000 7.84116119e-283]
{'status': 'do nothing', 'action': 'nothing', 'balance': 90.229992999999998,
'timestamp': '2019-08-31 02:39:42.275891'}
[0.00000000e+00 1.00000000e+00 1.39371786e-93]
{'status': 'buy 1 unit, cost 35.639999', 'action': 'buy', 'balance':
54.589993999999998, 'timestamp': '2019-08-31 02:39:42.276366'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 54.589993999999998,
'timestamp': '2019-08-31 02:39:42.276836'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 54.589993999999998,
'timestamp': '2019-08-31 02:39:42.277325'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 54.589993999999998,
'timestamp': '2019-08-31 02:39:42.277798'}
[0. 1. 0.]
{'status': 'buy 1 unit, cost 30.809999', 'action': 'buy', 'balance':
23.7799949999999985, 'timestamp': '2019-08-31 02:39:42.278268'}
[0. 1. 0.]
{'status': 'buy 1 unit, cost 30.490000', 'action': 'buy', 'balance':
-6.7100050000000013, 'timestamp': '2019-08-31 02:39:42.278739'}
[1. 0. 0.]

```

```

{'status': 'do nothing', 'action': 'nothing', 'balance': -6.7100050000000013,
'timestamp': '2019-08-31 02:39:42.279207'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': -6.7100050000000013,
'timestamp': '2019-08-31 02:39:42.279678'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': -6.7100050000000013,
'timestamp': '2019-08-31 02:39:42.280149'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': -6.7100050000000013,
'timestamp': '2019-08-31 02:39:42.280618'}
[0. 1. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': -6.7100050000000013,
'timestamp': '2019-08-31 02:39:42.281083'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 28.860001', 'investment': -19.023563945666766,
'gain': -6.7799979999999955, 'balance': 22.149995999999987, 'action': 'sell',
'timestamp': '2019-08-31 02:39:42.281636'}
[0. 1. 0.]
{'status': 'buy 1 unit, cost 29.219999', 'action': 'buy', 'balance':
-7.07000300000000105, 'timestamp': '2019-08-31 02:39:42.282125'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 29.520000', 'investment': -4.186949178414432,
'gain': -1.2899989999999946, 'balance': 22.449996999999999, 'action': 'sell',
'timestamp': '2019-08-31 02:39:42.282660'}
[1.0e+000 0.0e+000 2.5e-323]
{'status': 'do nothing', 'action': 'nothing', 'balance': 22.449996999999999,
'timestamp': '2019-08-31 02:39:42.283158'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 28.500000', 'investment': -6.526730075434576,
'gain': -1.9900000000000002, 'balance': 50.949996999999999, 'action': 'sell',
'timestamp': '2019-08-31 02:39:42.283696'}
[0. 0. 1.]
{'status': 'sell 1 unit, price 28.600000', 'investment': -2.121831010329591,
'gain': -0.6199989999999964, 'balance': 79.549996999999999, 'action': 'sell',
'timestamp': '2019-08-31 02:39:42.284236'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.549996999999999,
'timestamp': '2019-08-31 02:39:42.284700'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.549996999999999,
'timestamp': '2019-08-31 02:39:42.285168'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.549996999999999,
'timestamp': '2019-08-31 02:39:42.285621'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.549996999999999,
'timestamp': '2019-08-31 02:39:42.286073'}

```

```

[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.286537'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.287003'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.287465'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.287914'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.288377'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.288825'}
[1.4679413e-130 0.0000000e+000 1.0000000e+000]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.289297'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.289754'}
[0. 0. 1.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.290210'}
[1. 0. 0.]
{'status': 'do nothing', 'action': 'nothing', 'balance': 79.54999699999999,
'timestamp': '2019-08-31 02:39:42.290666'}

```

```

[19]: import copy
import pickle

copy_model = copy.deepcopy(agent.model)

with open('model.pkl', 'wb') as fopen:
    pickle.dump(copy_model, fopen)

```

```
[ ]:
```