13.double-duel-recurrent-q-learning-agent

September 29, 2021

[1]:
```
import numpy as np
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

[2]:
```
df = pd.read_csv('../dataset/GOOG-year.csv')
df.head()
```

[2]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2016-11-02	778.200012	781.650024	763.450012	768.700012	768.700012	1872400
1	2016-11-03	767.250000	769.950012	759.030029	762.130005	762.130005	1943200
2	2016-11-04	750.659973	770.359985	750.560974	762.020020	762.020020	2134800
3	2016-11-07	774.500000	785.190002	772.549988	782.520020	782.520020	1585100
4	2016-11-08	783.400024	795.632996	780.190002	790.510010	790.510010	1350800

[3]:
```
from collections import deque
import random

class Model:
    def __init__(self, input_size, output_size, layer_size, learning_rate,␣
 ↪name):
        with tf.variable_scope(name):
            self.X = tf.placeholder(tf.float32, (None, None, input_size))
            self.Y = tf.placeholder(tf.float32, (None, output_size))
            cell = tf.nn.rnn_cell.LSTMCell(layer_size, state_is_tuple = False)
            self.hidden_layer = tf.placeholder(tf.float32, (None, 2 *␣
 ↪layer_size))
```

```
            self.rnn,self.last_state = tf.nn.dynamic_rnn(inputs=self.
↪X,cell=cell,
                                                         dtype=tf.float32,
                                                         initial_state=self.
↪hidden_layer)
            tensor_action, tensor_validation = tf.split(self.rnn[:,-1],2,1)
            feed_action = tf.layers.dense(tensor_action, output_size)
            feed_validation = tf.layers.dense(tensor_validation, 1)
            self.logits = feed_validation + tf.subtract(feed_action,tf.
↪reduce_mean(feed_action,axis=1,keep_dims=True))
            self.cost = tf.reduce_sum(tf.square(self.Y - self.logits))
            self.optimizer = tf.train.AdamOptimizer(learning_rate =␣
↪learning_rate).minimize(self.cost)

class Agent:

    LEARNING_RATE = 0.003
    BATCH_SIZE = 32
    LAYER_SIZE = 256
    OUTPUT_SIZE = 3
    EPSILON = 0.5
    DECAY_RATE = 0.005
    MIN_EPSILON = 0.1
    GAMMA = 0.99
    MEMORIES = deque()
    COPY = 1000
    T_COPY = 0
    MEMORY_SIZE = 300

    def __init__(self, state_size, window_size, trend, skip):
        self.state_size = state_size
        self.window_size = window_size
        self.half_window = window_size // 2
        self.trend = trend
        self.skip = skip
        tf.reset_default_graph()
        self.INITIAL_FEATURES = np.zeros((4, self.state_size))
        self.model = Model(self.state_size, self.OUTPUT_SIZE, self.LAYER_SIZE,␣
↪self.LEARNING_RATE,
                           'real_model')
        self.model_negative = Model(self.state_size, self.OUTPUT_SIZE, self.
↪LAYER_SIZE, self.LEARNING_RATE,
                                    'negative_model')
        self.sess = tf.InteractiveSession()
        self.sess.run(tf.global_variables_initializer())
        self.trainable = tf.trainable_variables()
```

```
    def _assign(self, from_name, to_name):
        from_w = tf.get_collection(tf.GraphKeys.TRAINABLE_VARIABLES,␣
↪scope=from_name)
        to_w = tf.get_collection(tf.GraphKeys.TRAINABLE_VARIABLES,␣
↪scope=to_name)
        for i in range(len(from_w)):
            assign_op = to_w[i].assign(from_w[i])
            self.sess.run(assign_op)

    def _memorize(self, state, action, reward, new_state, dead, rnn_state):
        self.MEMORIES.append((state, action, reward, new_state, dead,␣
↪rnn_state))
        if len(self.MEMORIES) > self.MEMORY_SIZE:
            self.MEMORIES.popleft()

    def _select_action(self, state):
        if np.random.rand() < self.EPSILON:
            action = np.random.randint(self.OUTPUT_SIZE)
        else:
            action = self.get_predicted_action([state])
        return action

    def _construct_memories(self, replay):
        states = np.array([a[0] for a in replay])
        new_states = np.array([a[3] for a in replay])
        init_values = np.array([a[-1] for a in replay])
        Q = self.sess.run(self.model.logits, feed_dict={self.model.X:states,
                                                        self.model.hidden_layer:
↪init_values})
        Q_new = self.sess.run(self.model.logits, feed_dict={self.model.X:
↪new_states,
                                                            self.model.hidden_layer:
↪init_values})
        Q_new_negative = self.sess.run(self.model_negative.logits,
                                       feed_dict={self.model_negative.X:new_states,
                                                  self.model_negative.hidden_layer:
↪init_values})
        replay_size = len(replay)
        X = np.empty((replay_size, 4, self.state_size))
        Y = np.empty((replay_size, self.OUTPUT_SIZE))
        INIT_VAL = np.empty((replay_size, 2 * self.LAYER_SIZE))
        for i in range(replay_size):
            state_r, action_r, reward_r, new_state_r, dead_r, rnn_memory =␣
↪replay[i]
            target = Q[i]
```

```
            target[action_r] = reward_r
            if not dead_r:
                target[action_r] += self.GAMMA * Q_new_negative[i, np.
↪argmax(Q_new[i])]
            X[i] = state_r
            Y[i] = target
            INIT_VAL[i] = rnn_memory
        return X, Y, INIT_VAL

    def get_state(self, t):
        window_size = self.window_size + 1
        d = t - window_size + 1
        block = self.trend[d : t + 1] if d >= 0 else -d * [self.trend[0]] +␣
↪self.trend[0 : t + 1]
        res = []
        for i in range(window_size - 1):
            res.append(block[i + 1] - block[i])
        return np.array(res)

    def buy(self, initial_money):
        starting_money = initial_money
        states_sell = []
        states_buy = []
        inventory = []
        state = self.get_state(0)
        init_value = np.zeros((1, 2 * self.LAYER_SIZE))
        for k in range(self.INITIAL_FEATURES.shape[0]):
            self.INITIAL_FEATURES[k,:] = state
        for t in range(0, len(self.trend) - 1, self.skip):
            action, last_state = self.sess.run([self.model.logits,self.model.
↪last_state],
                                               feed_dict={self.model.X:[self.
↪INITIAL_FEATURES],
                                                          self.model.
↪hidden_layer:init_value})
            action, init_value = np.argmax(action[0]), last_state
            next_state = self.get_state(t + 1)

            if action == 1 and initial_money >= self.trend[t]:
                inventory.append(self.trend[t])
                initial_money -= self.trend[t]
                states_buy.append(t)
                print('day %d: buy 1 unit at price %f, total balance %f'% (t,␣
↪self.trend[t], initial_money))

            elif action == 2 and len(inventory):
```

```
                bought_price = inventory.pop(0)
                initial_money += self.trend[t]
                states_sell.append(t)
                try:
                    invest = ((close[t] - bought_price) / bought_price) * 100
                except:
                    invest = 0
                print(
                    'day %d, sell 1 unit at price %f, investment %f %%, total␣
↪balance %f,'
                    % (t, close[t], invest, initial_money)
                )

            new_state = np.append([self.get_state(t + 1)], self.
↪INITIAL_FEATURES[:3, :], axis = 0)
            self.INITIAL_FEATURES = new_state
        invest = ((initial_money - starting_money) / starting_money) * 100
        total_gains = initial_money - starting_money
        return states_buy, states_sell, total_gains, invest


    def train(self, iterations, checkpoint, initial_money):
        for i in range(iterations):
            total_profit = 0
            inventory = []
            state = self.get_state(0)
            starting_money = initial_money
            init_value = np.zeros((1, 2 * self.LAYER_SIZE))
            for k in range(self.INITIAL_FEATURES.shape[0]):
                self.INITIAL_FEATURES[k,:] = state
            for t in range(0, len(self.trend) - 1, self.skip):
                if (self.T_COPY + 1) % self.COPY == 0:
                    self._assign('real_model', 'negative_model')

                if np.random.rand() < self.EPSILON:
                    action = np.random.randint(self.OUTPUT_SIZE)
                else:
                    action, last_state = self.sess.run([self.model.logits,
                                                        self.model.last_state],
                                                        feed_dict={self.model.X:[self.
↪INITIAL_FEATURES],
                                                                   self.model.
↪hidden_layer:init_value})
                    action, init_value = np.argmax(action[0]), last_state

                next_state = self.get_state(t + 1)
```

```
                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) > 0:
                    bought_price = inventory.pop(0)
                    total_profit += self.trend[t] - bought_price
                    starting_money += self.trend[t]

                invest = ((starting_money - initial_money) / initial_money)
                new_state = np.append([self.get_state(t + 1)], self.
↪INITIAL_FEATURES[:3, :], axis = 0)

                self._memorize(self.INITIAL_FEATURES, action, invest, new_state,
                               starting_money < initial_money, init_value[0])
                self.INITIAL_FEATURES = new_state
                batch_size = min(len(self.MEMORIES), self.BATCH_SIZE)
                replay = random.sample(self.MEMORIES, batch_size)
                X, Y, INIT_VAL = self._construct_memories(replay)

                cost, _ = self.sess.run([self.model.cost, self.model.optimizer],
                                        feed_dict={self.model.X: X, self.model.
↪Y:Y,
                                                   self.model.hidden_layer:␣
↪INIT_VAL})
                self.T_COPY += 1
                self.EPSILON = self.MIN_EPSILON + (1.0 - self.MIN_EPSILON) * np.
↪exp(-self.DECAY_RATE * i)
            if (i+1) % checkpoint == 0:
                print('epoch: %d, total rewards: %f.3, cost: %f, total money:␣
↪%f'%(i + 1, total_profit, cost,
                                                                              ␣
↪  starting_money))
```

[4]:
```
close = df.Close.values.tolist()
initial_money = 10000
window_size = 30
skip = 1
batch_size = 32
agent = Agent(state_size = window_size,
              window_size = window_size,
              trend = close,
              skip = skip)
agent.train(iterations = 200, checkpoint = 10, initial_money = initial_money)
```

```
WARNING:tensorflow:<tensorflow.python.ops.rnn_cell_impl.LSTMCell object at
0x7f39ffaed7b8>: Using a concatenated state is slower and will soon be
```

```
deprecated.  Use state_is_tuple=True.
WARNING:tensorflow:From <ipython-input-3-401815182242>:17: calling reduce_mean
(from tensorflow.python.ops.math_ops) with keep_dims is deprecated and will be
removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
WARNING:tensorflow:<tensorflow.python.ops.rnn_cell_impl.LSTMCell object at
0x7f39ffaede80>: Using a concatenated state is slower and will soon be
deprecated.  Use state_is_tuple=True.
epoch: 10, total rewards: 328.014401.3, cost: 0.233912, total money: 2446.714413
epoch: 20, total rewards: 629.485052.3, cost: 0.592428, total money: 5723.605047
epoch: 30, total rewards: 1222.065245.3, cost: 0.182284, total money:
7288.965209
epoch: 40, total rewards: 719.309753.3, cost: 0.690094, total money: 3739.159728
epoch: 50, total rewards: 328.994876.3, cost: 0.918951, total money: 2756.724856
epoch: 60, total rewards: 1518.540281.3, cost: 0.226017, total money:
10545.210264
epoch: 70, total rewards: 440.315127.3, cost: 0.145386, total money: 7494.335086
epoch: 80, total rewards: 656.779966.3, cost: 0.113699, total money: 6666.949948
epoch: 90, total rewards: 846.820129.3, cost: 0.444679, total money: 6860.080139
epoch: 100, total rewards: 1044.679930.3, cost: 0.240218, total money:
9067.419920
epoch: 110, total rewards: 207.934935.3, cost: 0.236219, total money:
10207.934935
epoch: 120, total rewards: 6.745002.3, cost: 1.133358, total money: 10006.745002
epoch: 130, total rewards: 586.910091.3, cost: 0.162622, total money:
4665.650081
epoch: 140, total rewards: 1084.244877.3, cost: 0.630996, total money:
6178.484867
epoch: 150, total rewards: 991.774842.3, cost: 1.439193, total money: 420.904786
epoch: 160, total rewards: 714.735100.3, cost: 0.337296, total money:
5744.735038
epoch: 170, total rewards: 1158.574706.3, cost: 0.186633, total money:
10185.244689
epoch: 180, total rewards: 1120.314817.3, cost: 0.539594, total money:
7186.704770
epoch: 190, total rewards: 230.760193.3, cost: 0.110742, total money:
4290.020202
epoch: 200, total rewards: 218.420047.3, cost: 0.125164, total money:
10218.420047
```

[5]:
```
states_buy, states_sell, total_gains, invest = agent.buy(initial_money =␣
↪initial_money)
```

```
day 17: buy 1 unit at price 768.239990, total balance 9231.760010
day 18, sell 1 unit at price 770.840027, investment 0.338441 %, total balance
10002.600037,
day 20: buy 1 unit at price 747.919983, total balance 9254.680054
```

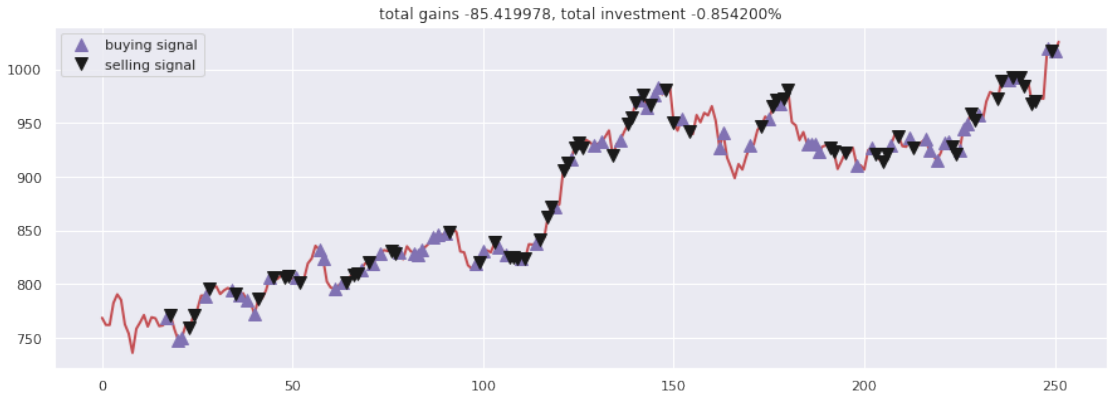
day 21: buy 1 unit at price 750.500000, total balance 8504.180054
day 23, sell 1 unit at price 759.109985, investment 1.496150 %, total balance 9263.290039,
day 24, sell 1 unit at price 771.190002, investment 2.756829 %, total balance 10034.480041,
day 27: buy 1 unit at price 789.270020, total balance 9245.210021
day 28, sell 1 unit at price 796.099976, investment 0.865351 %, total balance 10041.309997,
day 34: buy 1 unit at price 794.559998, total balance 9246.749999
day 35, sell 1 unit at price 791.260010, investment -0.415323 %, total balance 10038.010009,
day 36: buy 1 unit at price 789.909973, total balance 9248.100036
day 38: buy 1 unit at price 785.049988, total balance 8463.050048
day 40: buy 1 unit at price 771.820007, total balance 7691.230041
day 41, sell 1 unit at price 786.140015, investment -0.477264 %, total balance 8477.370056,
day 44: buy 1 unit at price 806.150024, total balance 7671.220032
day 45, sell 1 unit at price 806.650024, investment 2.751422 %, total balance 8477.870056,
day 48, sell 1 unit at price 806.359985, investment 4.475134 %, total balance 9284.230041,
day 49, sell 1 unit at price 807.880005, investment 0.214598 %, total balance 10092.110046,
day 51: buy 1 unit at price 806.070007, total balance 9286.040039
day 52, sell 1 unit at price 802.174988, investment -0.483211 %, total balance 10088.215027,
day 57: buy 1 unit at price 832.150024, total balance 9256.065003
day 58: buy 1 unit at price 823.309998, total balance 8432.755005
day 61: buy 1 unit at price 795.695007, total balance 7637.059998
day 63: buy 1 unit at price 801.489990, total balance 6835.570008
day 64, sell 1 unit at price 801.340027, investment -3.702457 %, total balance 7636.910035,
day 66, sell 1 unit at price 808.380005, investment -1.813411 %, total balance 8445.290040,
day 67, sell 1 unit at price 809.559998, investment 1.742501 %, total balance 9254.850038,
day 68: buy 1 unit at price 813.669983, total balance 8441.180055
day 70, sell 1 unit at price 820.450012, investment 2.365597 %, total balance 9261.630067,
day 71: buy 1 unit at price 818.979980, total balance 8442.650087
day 73: buy 1 unit at price 828.070007, total balance 7614.580080
day 76, sell 1 unit at price 831.330017, investment 2.170417 %, total balance 8445.910097,
day 77, sell 1 unit at price 828.640015, investment 1.179520 %, total balance 9274.550112,
day 78: buy 1 unit at price 829.280029, total balance 8445.270083
day 82: buy 1 unit at price 829.080017, total balance 7616.190066
day 83: buy 1 unit at price 827.780029, total balance 6788.410037

```
day 84: buy 1 unit at price 831.909973, total balance 5956.500064
day 87: buy 1 unit at price 843.250000, total balance 5113.250064
day 88: buy 1 unit at price 845.539978, total balance 4267.710086
day 90: buy 1 unit at price 847.200012, total balance 3420.510074
day 91, sell 1 unit at price 848.780029, investment 2.500999 %, total balance
4269.290103,
day 98: buy 1 unit at price 819.510010, total balance 3449.780093
day 99, sell 1 unit at price 820.919983, investment -1.008109 %, total balance
4270.700076,
day 100: buy 1 unit at price 831.409973, total balance 3439.290103
day 103, sell 1 unit at price 838.549988, investment 1.142226 %, total balance
4277.840091,
day 104: buy 1 unit at price 834.570007, total balance 3443.270084
day 106: buy 1 unit at price 827.880005, total balance 2615.390079
day 107, sell 1 unit at price 824.669983, investment -0.375709 %, total balance
3440.060062,
day 108, sell 1 unit at price 824.729980, investment -0.863073 %, total balance
4264.790042,
day 109, sell 1 unit at price 823.349976, investment -2.359920 %, total balance
5088.140018,
day 110: buy 1 unit at price 824.320007, total balance 4263.820011
day 111, sell 1 unit at price 823.559998, investment -2.599520 %, total balance
5087.380009,
day 114: buy 1 unit at price 838.210022, total balance 4249.169987
day 115, sell 1 unit at price 841.650024, investment -0.655098 %, total balance
5090.820011,
day 117, sell 1 unit at price 862.760010, investment 5.277544 %, total balance
5953.580021,
day 118, sell 1 unit at price 872.299988, investment 4.918153 %, total balance
6825.880009,
day 119: buy 1 unit at price 871.729980, total balance 5954.150029
day 121, sell 1 unit at price 905.960022, investment 8.554107 %, total balance
6860.110051,
day 122, sell 1 unit at price 912.570007, investment 10.229744 %, total balance
7772.680058,
day 123: buy 1 unit at price 916.440002, total balance 6856.240056
day 124, sell 1 unit at price 927.039978, investment 12.461177 %, total balance
7783.280034,
day 125, sell 1 unit at price 931.659973, investment 11.148751 %, total balance
8714.940007,
day 126, sell 1 unit at price 927.130005, investment 6.355182 %, total balance
9642.070012,
day 129: buy 1 unit at price 928.780029, total balance 8713.289983
day 131: buy 1 unit at price 932.219971, total balance 7781.070012
day 134, sell 1 unit at price 919.619995, investment 0.346994 %, total balance
8700.690007,
day 136: buy 1 unit at price 934.010010, total balance 7766.679997
day 138, sell 1 unit at price 948.820007, investment 2.157667 %, total balance
```

```
8715.500004,
day 139, sell 1 unit at price 954.960022, investment 2.439344 %, total balance
9670.460026,
day 140, sell 1 unit at price 969.539978, investment 3.804024 %, total balance
10640.000004,
day 141: buy 1 unit at price 971.469971, total balance 9668.530033
day 142, sell 1 unit at price 975.880005, investment 0.453955 %, total balance
10644.410038,
day 143: buy 1 unit at price 964.859985, total balance 9679.550053
day 144, sell 1 unit at price 966.950012, investment 0.216615 %, total balance
10646.500065,
day 145: buy 1 unit at price 975.599976, total balance 9670.900089
day 146: buy 1 unit at price 983.679993, total balance 8687.220096
day 148, sell 1 unit at price 980.940002, investment 0.547358 %, total balance
9668.160098,
day 150, sell 1 unit at price 949.830017, investment -3.441157 %, total balance
10617.990115,
day 152: buy 1 unit at price 953.400024, total balance 9664.590091
day 154, sell 1 unit at price 942.309998, investment -1.163208 %, total balance
10606.900089,
day 162: buy 1 unit at price 927.330017, total balance 9679.570072
day 163: buy 1 unit at price 940.489990, total balance 8739.080082
day 170: buy 1 unit at price 928.799988, total balance 7810.280094
day 173, sell 1 unit at price 947.159973, investment 2.138393 %, total balance
8757.440067,
day 175: buy 1 unit at price 953.419983, total balance 7804.020084
day 176, sell 1 unit at price 965.400024, investment 2.648623 %, total balance
8769.420108,
day 177, sell 1 unit at price 970.890015, investment 4.531657 %, total balance
9740.310123,
day 178: buy 1 unit at price 968.150024, total balance 8772.160099
day 179, sell 1 unit at price 972.919983, investment 2.045269 %, total balance
9745.080082,
day 180, sell 1 unit at price 980.340027, investment 1.259103 %, total balance
10725.420109,
day 185: buy 1 unit at price 930.500000, total balance 9794.920109
day 186: buy 1 unit at price 930.830017, total balance 8864.090092
day 187: buy 1 unit at price 930.390015, total balance 7933.700077
day 188: buy 1 unit at price 923.650024, total balance 7010.050053
day 191, sell 1 unit at price 926.789978, investment -0.398713 %, total balance
7936.840031,
day 192, sell 1 unit at price 922.900024, investment -0.851927 %, total balance
8859.740055,
day 195, sell 1 unit at price 922.669983, investment -0.829763 %, total balance
9782.410038,
day 198: buy 1 unit at price 910.979980, total balance 8871.430058
day 202: buy 1 unit at price 927.000000, total balance 7944.430058
day 203, sell 1 unit at price 921.280029, investment -0.256590 %, total balance
```

8865.710087,
day 205, sell 1 unit at price 913.809998, investment 0.310656 %, total balance 9779.520085,
day 206, sell 1 unit at price 921.289978, investment -0.615968 %, total balance 10700.810063,
day 207: buy 1 unit at price 929.570007, total balance 9771.240056
day 209, sell 1 unit at price 937.340027, investment 0.835872 %, total balance 10708.580083,
day 212: buy 1 unit at price 935.950012, total balance 9772.630071
day 213, sell 1 unit at price 926.500000, investment -1.009671 %, total balance 10699.130071,
day 216: buy 1 unit at price 935.090027, total balance 9764.040044
day 217: buy 1 unit at price 925.109985, total balance 8838.930059
day 219: buy 1 unit at price 915.000000, total balance 7923.930059
day 221: buy 1 unit at price 931.580017, total balance 6992.350042
day 222: buy 1 unit at price 932.450012, total balance 6059.900030
day 223, sell 1 unit at price 928.530029, investment -0.701537 %, total balance 6988.430059,
day 224, sell 1 unit at price 920.969971, investment -0.447516 %, total balance 7909.400030,
day 225: buy 1 unit at price 924.859985, total balance 6984.540045
day 226: buy 1 unit at price 944.489990, total balance 6040.050055
day 227: buy 1 unit at price 949.500000, total balance 5090.550055
day 228, sell 1 unit at price 959.109985, investment 4.820763 %, total balance 6049.660040,
day 229, sell 1 unit at price 953.270020, investment 2.328303 %, total balance 7002.930060,
day 230: buy 1 unit at price 957.789978, total balance 6045.140082
day 235, sell 1 unit at price 972.599976, investment 4.305857 %, total balance 7017.740058,
day 236, sell 1 unit at price 989.250000, investment 6.962137 %, total balance 8006.990058,
day 238: buy 1 unit at price 989.679993, total balance 7017.310065
day 239, sell 1 unit at price 992.000000, investment 5.030229 %, total balance 8009.310065,
day 240: buy 1 unit at price 992.179993, total balance 7017.130072
day 241, sell 1 unit at price 992.809998, investment 4.561348 %, total balance 8009.940070,
day 242, sell 1 unit at price 984.450012, investment 2.783495 %, total balance 8994.390082,
day 244, sell 1 unit at price 968.450012, investment -2.145136 %, total balance 9962.840094,
day 245, sell 1 unit at price 970.539978, investment -2.181057 %, total balance 10933.380072,
day 248: buy 1 unit at price 1019.270020, total balance 9914.110052
day 249, sell 1 unit at price 1017.109985, investment -0.211920 %, total balance 10931.220037,
day 250: buy 1 unit at price 1016.640015, total balance 9914.580022

[6]:
```
fig = plt.figure(figsize = (15,5))
plt.plot(close, color='r', lw=2.)
plt.plot(close, '^', markersize=10, color='m', label = 'buying signal',␣
 ↪markevery = states_buy)
plt.plot(close, 'v', markersize=10, color='k', label = 'selling signal',␣
 ↪markevery = states_sell)
plt.title('total gains %f, total investment %f%%'%(total_gains, invest))
plt.legend()
plt.show()
```



[ ]:
```

```