Automatizirano trgovanje financijskim instrumentima temeljeno na indeksu relativne snage

Sirovec, Borna

Sveučilište J. J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike i računarstva

Borna Sirovec

Automatizirano trgovanje financijskim instrumentima temeljeno na indeksu relativne snage

Završni rad

Osijek, 2021.

Sveučilište J. J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike i računarstva

Borna Sirovec

Automatizirano trgovanje financijskim instrumentima temeljeno na indeksu relativne snage

Završni rad

Mentor: izv. prof. dr. sc. Nenad Šuvak

Osijek, 2021.

English paper title

Sažetak

U ovom ćemo se radu baviti indeksom relativne snage (dalje u tekstu: RSI), te primjenog istog u trgovanju financijskim instrumentima s naglaskom na dionice. Rad je podijeljen na dva dijela. U prvom dijelu opisana je povijest metode za trgovanje temeljene na indeksu relativne snage, te je također objašnjen i način funkcioniranja same metode. Drugi dio sadrži implementaciju metode bazirane na indeksu relativne snage u programskom jeziku Python.

Ključne riječi

indeks relativne snage, dionica, kupovina, prodaja

Abstract

In this paper we will discuss relative strength index (hereinafter called RSI) along with its use in stock trading. The paper is divided in two sections. In the first section, we will read about history and way of functioning of method based on RSI. The second section contains the implementation of method based on RSI in Python programming langauge.

Key words

relative strength index, stock, purchase, sale

Sadržaj

Uvod

RSI je momentni oscilator koji uspoređuje veličine nedavnih gubitaka i dobitaka tokom određenog vremenskog razdoblja, kako bi se mjerila brzina i promjena kretanja cijene određene vrijednosnice. Njegova primarna upotreba je identificiranje prekupljenosti ili preprodanosti vrijednosnice. Razdobolje određivanja indeksa jest 5,7,10,14 ili čak 28 dana. Što je razdoblje kraće, oscilator postaje osjetljiviji i amplitude su veće. Za kratkoročno trgovanje informativniji su RSI temeljeni na kraćim povijesnim periodima, dok su za dugoročnija ulaganja informativniiji oni RSI temeljeni na dužim povijesnim periodima jer su amplitude ravnije.

Slika 1. Prikaz razlike u amplitudama vrijednosti indeksa petodnevnog RSI (crveno) i četrnaestodnevnog RSI (plavo)

1 Indeks relativne snage

1.1 Povijesni pregled

RSI je predstavio John Welles Wilder 1978. godine u knjizi "New Concepts in Tehnical Trading Systems". Nakon školovanja za inženjera mehanike, pokazao je zanimanje za ulaganja u mirovnske fondove i trgovanje srebrom, te se počinje baviti nekretninama. Kasnije, nakon 13 godina rada i učenja o tržištu kapitala, on postaje znalac u tom području, a njegovo ime postaje prepoznatiljivo diljem svijeta. S godinama, konstruirao je nekoliko metoda i indeksa, a najpoznatiji su: prosjek pravih vrijednosti, indeks relativne snage, indikator direktnog pokreta.

1.2 Izračun indeksa relativne snage

Indeks relativne snage mjeri se kroz period od primjerice 10 ili 14 dana te formula za izračun glasi:

$$\text{RSI} = 100 - \frac{100}{1 + RS},$$

$$\text{RS} = \frac{\bar{d}_n}{\bar{g}_n},$$

gdje je s $\bar{d}_n$ označen prosječni dobitak, a s $\bar{g}_n$ prosječni gubitak za n prethodnih dana.

U Python RSI implementaciji, definirani period za izračun indeksa jest 10 dana. Tome slijede i formule za računanje prosječnog dobitka i gubitka. Naime, kako radimo izračun za 10 dana, potrebno je prvo 10 dana promatrati cijene te izračunati:

$$\bar{d}_{10} = \frac{\sum_{i=1}^{10} UpMove_i}{10},$$

$$\bar{g}_{10} = \frac{\sum_{i=1}^{10} DownMove_i}{10},$$

gdje *UpMove* i *DownMove* označavaju porast, odnosno smanjenje cijene (*eng. adj close*) u odnosu na prethodni dan.

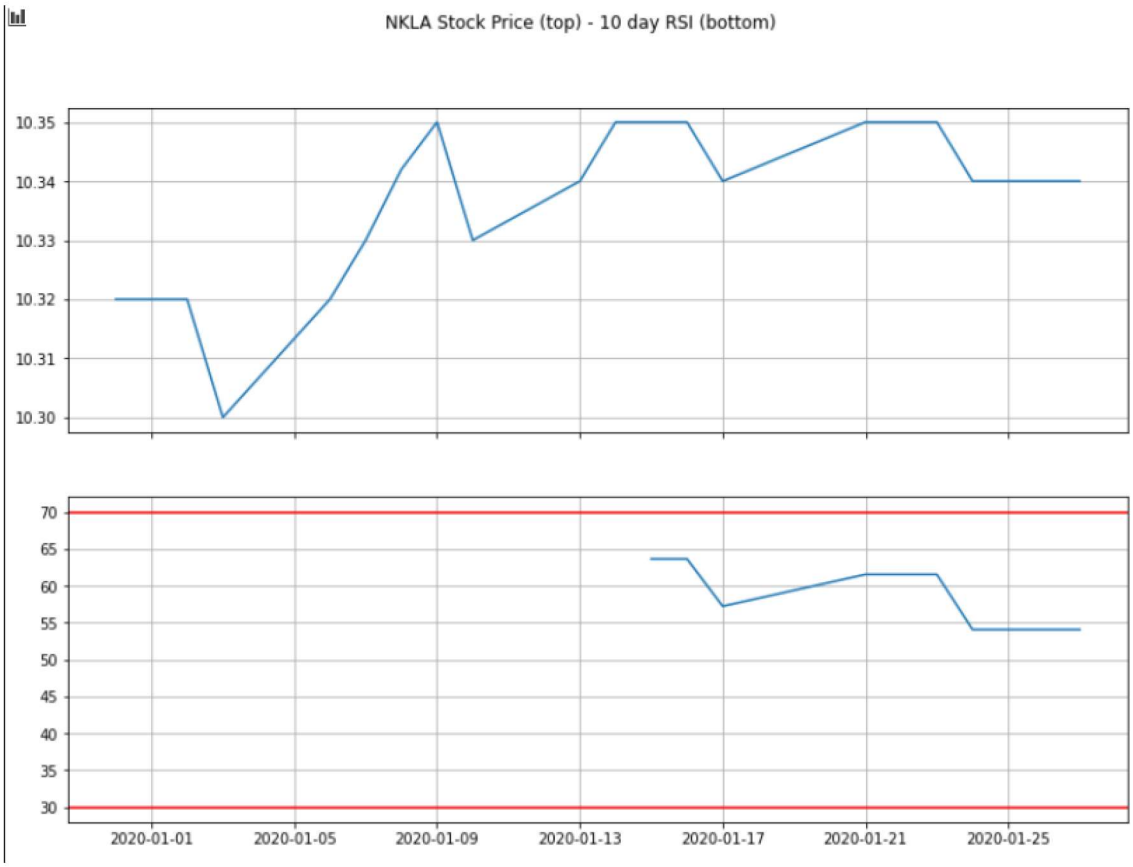
Nakon toga slijede izračuni za prosječni dobitak (*eng. average up*) i prosječni gubitak (*eng. average down*):

$$\bar{d}_n = \frac{\bar{d}_{n-1} \cdot 9 + UpMove_n}{10}$$

$$\bar{g}_n = \frac{\bar{g}_{n-1} \cdot 9 + DownMove_n}{10}.$$

1	Date	Adj Close	Down Move	Average Up	Average Down	RS	RSI
2	2019-12-31 00:00:00	10,3199997					
3	2020-01-02 00:00:00	10,3199997	0				
4	2020-01-03 00:00:00	10,3000002	0,019999504				
5	2020-01-06 00:00:00	10,3199997	0				
6	2020-01-07 00:00:00	10,3299999	0				
7	2020-01-08 00:00:00	10,342	0				
8	2020-01-09 00:00:00	10,3500004	0				
9	2020-01-10 00:00:00	10,3299999	0,020000458				
10	2020-01-13 00:00:00	10,3400002	0				
11	2020-01-14 00:00:00	10,3500004	0				
12	2020-01-15 00:00:00	10,3500004	0	0,00700006	0,003999996	1,7500179	63,63660008
13	2020-01-16 00:00:00	10,3500004	0	0,00630006	0,003599997	1,7500179	63,63660008
14	2020-01-17 00:00:00	10,3400002	0,010000229	0,00567005	0,00424002	1,3372703	57,21504689
15	2020-01-21 00:00:00	10,3500004	0	0,00610307	0,003816018	1,5993296	61,52854148
16	2020-01-22 00:00:00	10,3500004	0	0,00549276	0,003434416	1,5993296	61,52854148
17	2020-01-23 00:00:00	10,3500004	0	0,00494349	0,003090974	1,5993296	61,52854148
18	2020-01-24 00:00:00	10,3400002	0,010000229	0,00444914	0,0037819	1,1764294	54,05318424
19	2020-01-27 00:00:00	10,3400002	0	0,00400422	0,00340371	1,1764294	54,05318424
20	2020-01-28 00:00:00	10,3450003	0	0,00410381	0,003063339	1,3396537	57,25863222

Slika 2. Prikaz Excel tablice sa izračunatim indeksima za 20 dana



Slika 3. Grafički prikaz vrijednosti RSI za prvih 20 dana

1.3 Obrazac neuspješnog zamaha

RSI postiže vrijednosti od 0 do 100. Tržište s pretjeranim kupovinama označava indeks veći od 70 (*eng.* overbought), dok ono s pretjeranim prodajama označava indeks manji od 30 (*eng.* oversold). Obrazac "neuspješnog zamaha" događa se kada je RSI iznad 70, odnosno ispod 30.

1.3.1 Bearish divergence

Bearish divergence postiže se kada u porastu cijene, lokalni maksimum RSI-a (preko 70 bodova) ne uspije premašiti prethodni maksimum, pa kasnije slijedi pad vrijednosti RSI. Sljedeća slika prikazuje neuspješan zamah na vrhu te linije koje upućuju na divergenciju cijene i RSI:

Slika 4. Divergencija cijene(gore) i RSI(dolje) pri neuspješnom zamahu na vrhu

1.3.2 Bullish divergence

Bullish divergence postiže se kada u padu cijene, lokalni minimum RSI-a (ispod 30 bodova) ne uspije pasti ispod prethodnog minimuma, pa slijedi porast vrijednosti RSI. Sljedeća slika prikazuje neuspješan zamah na dnu te linije koje upućuju na divergenciju cijene i indeksa relativne snage:

Slika 5. Divergencija cijene(gore) i RSI(dolje) pri neuspješnom zamahu na dnu

2 Implementacija RSI u programskom jeziku Python

Programski kod sačinjen je od dvije datoteke. Glavna datoteka jest Python bilježnica imena "rsi.ipynb", a druga, ona pomoćna, jest "rsi_help.py". U njoj se nalaze funkcije za izračun vrijednosti prosječnih dobitaka i gubitaka, izračun relativnih snaga i samog RSI. Također, nalaze se i funkcije za popunjavanje stupaca izračunatim podacima, te funkcije za crtanje grafova.

2.1 Dohvaćanje i priprema podataka

Podaci koji se obrađuju i na kojima se izvodi računanje indeksa relativne snage dohvaćaju se funkcijom yf.download() koja je dio "yfinance" API-ja tvtke Yahoo!. Uz yfinance, korištene biblioteke su NumPy te Pandas. Konkretno, gledati ćemo cijene dionica tvrtke Nikola Corporation čija je skraćenica "NKLA", u razdoblju od 01.01.2020. do 20.08.2021.

```
import numpy as np
import pandas as pd
import numpy as np
import yfinance as yf
import datetime as dt
import rsi_help as rh

name = "NKLA"
start = "2020-01-01"
end = "2021-08-20"

data = yf.download(name,start,end)
```

Primijetimo, tip podatka **data** jest pd.dataframe. Sada nad dohvaćenim podacima vršimo obradu te postojećim stupcima Date, Open, High, Low, Close, Adj Close te Volume, dodajemo stupce nama potrebne za izračun indeksa relativne snage: Up Move, Down Move, Average Up, Average Down, RS, RSI. Također, prethodno navedene stupce inicijalno postavljamo na tip podatka np.nan budući da predstavljaju podatke koje tek trebamo izračunati.

```
data['Up Move'] = np.nan
data['Down Move'] = np.nan
data['Average Up'] = np.nan
data['Average Down'] = np.nan

    # Relative Strength
data['RS'] = np.nan

    # Relative Strength Index
data['RSI'] = np.nan
```

2.2 Obrada podataka i računanje RSI

Nakon što su svi stupci inicijalizirani i imamo sve potrebno za izračun Indeksa relativne snage, u glavnoj datoteci definiramo i na skup podataka pozivamo funkciju calculateRsi(data) koja računa gore navedene vrijednosti te ih upisuje u odgovarajuće stupce.

```
def calculateRsi(data):
    data = rh.fillMoves(data)
    for _day in range (1,len(data)):
        if(~(data['Down Move'][_day]>0)):
            data['Down Move'][_day]=0
    data = rh.fillRsi(data)
    return data

data = calculateRsi(data)
```

2.2.1 Funkcija fillMoves(data)

Funkcija fillMoves(data) za svaki redak tablice, tj. jedan dan računa vrijednost stupaca Up Move i Down Move. Primijetimo da nakon poziva funkcije, rh.fillMoves(data) na našim podacima, naknadno iteriramo stupcem Down Move kako bi osigurali da nam tip podatka ne ostane np.nan jer takav tip podatka kasnije ne možemo upotrijebiti u računanju.

```
def fillMoves(data):
    for _day in range(1, len(data)):
        _previousDay = _day-1
        data['Down Move'][_day] = 0
        data['Up Move'][_day] = 0

        if data['Adj Close'][_day] > data['Adj Close'][_previousDay]:
            data['Up Move'][_day] = data['Adj Close'][_day] - data['Adj
   Close'][_previousDay]


        if data['Adj Close'][_day] < data['Adj Close'][_previousDay]:
            data['Down Move'][_day] = abs(data['Adj Close'][_day] - data['
   Adj Close'][_previousDay])

    return data
```

2.2.2 Funkcija fillRsi(data)

Ova je funkcija skup ugnježdenih funkcija fillAverages(data) i fillRelativeStrength(data). Po formulama navedenim u poglavlju 1.2 one računaju prosječne dobitke i gubitke, relativnu snagu za pojedine dane, te na koncu izračunavaju i indeks relativne snage za odgovarajući dan.

```
def fillAverages(data):

    _data = data
    ## First 10days Avg
    _data['Average Up'][10] = _data['Up Move'][1:11].mean()
    _data['Average Down'][10] = _data['Down Move'][1:11].mean()

    ## Rest Avgs
    for _day in range(11,len(_data)):
        _previousDay = _day-1

        _data['Average Up'][_day] = (_data['Average Up'][_previousDay]*9 +
    _data['Up Move'][_day])/10
        _data['Average Down'][_day] = (_data['Average Down'][_previousDay
   ]*9 + _data['Down Move'][_day])/10

    return _data


def fillRelativeStrength(data):
    _data = fillAverages(data)

    ## First 10days RS
    _data['RS'][10] = _data['Average Up'][10] / _data['Average Down'][10]

    ## Rest RelativeStrengths
    for _day in range(11,len(_data)):
        _data['RS'][_day] = _data['Average Up'][_day] / _data['Average
   Down'][_day]

    return _data

def fillRsi(data):
    _data = fillRelativeStrength(data)

    # First 10Days RSI
    _data['RSI'][10] = 100 - (100 / (1 + _data['RS'][10]))

    # Rest RSIs
    for _day in range(11,len(_data)):
        _data['RSI'][_day] = 100 - (100/ (1+_data['RS'][_day]))
    return _data
```

U ovome trenutku, naši su podaci obrađeni. Za zadano vremensko razdobolje izračunat je desetodnevni RSI, te su naši podaci spremni za grafički prikaz.
Primijetimo da se vrijednosti RSI računaju tek 11. dana, jer koristimo desetodnevni RSI.

Date	Open	High	Low	Close	Adj Close	Volume	Up Move	Down Move	Average Up	Average Down	RS	RSI
2019-12-31	10.317	10.320	10.315	10.320	10.320	6500	NaN	NaN	NaN	NaN	NaN	NaN
2020-01-02	10.310	10.320	10.310	10.320	10.320	60300	0.000	0.00	NaN	NaN	NaN	NaN
2020-01-03	10.300	10.300	10.300	10.300	10.300	15000	0.000	0.02	NaN	NaN	NaN	NaN
2020-01-06	10.325	10.350	10.320	10.320	10.320	68700	0.020	0.00	NaN	NaN	NaN	NaN
2020-01-07	10.440	10.440	10.330	10.330	10.330	42900	0.010	0.00	NaN	NaN	NaN	NaN
2020-01-08	10.330	10.350	10.300	10.342	10.342	1247500	0.012	0.00	NaN	NaN	NaN	NaN
2020-01-09	10.350	10.350	10.350	10.350	10.350	168700	0.008	0.00	NaN	NaN	NaN	NaN
2020-01-10	10.350	10.350	10.330	10.330	10.330	370000	0.000	0.02	NaN	NaN	NaN	NaN
2020-01-13	10.350	10.350	10.340	10.340	10.340	25600	0.010	0.00	NaN	NaN	NaN	NaN
2020-01-14	10.350	10.350	10.350	10.350	10.350	10000	0.010	0.00	NaN	NaN	NaN	NaN
2020-01-15	10.350	10.350	10.350	10.350	10.350	0	0.000	0.00	0.007000	0.004000	1.750018	63.636600
2020-01-16	10.348	10.360	10.345	10.350	10.350	672400	0.000	0.00	0.006300	0.003600	1.750018	63.636600
2020-01-17	10.350	10.360	10.340	10.340	10.340	667000	0.000	0.01	0.005670	0.004240	1.337270	57.215047
2020-01-21	10.340	10.360	10.340	10.350	10.350	356700	0.010	0.00	0.006103	0.003816	1.599330	61.528541
2020-01-22	10.350	10.350	10.350	10.350	10.350	33400	0.000	0.00	0.005493	0.003434	1.599330	61.528541
2020-01-23	10.360	10.360	10.350	10.350	10.350	180000	0.000	0.00	0.004943	0.003091	1.599330	61.528541
2020-01-24	10.350	10.350	10.340	10.340	10.340	350000	0.000	0.01	0.004449	0.003782	1.176429	54.053184
2020-01-27	10.340	10.340	10.340	10.340	10.340	100000	0.000	0.00	0.004004	0.003404	1.176429	54.053184
2020-01-28	10.345	10.345	10.345	10.345	10.345	100000	0.005	0.00	0.004104	0.003063	1.339654	57.258632
2020-01-29	10.340	10.350	10.340	10.350	10.350	118000	0.005	0.00	0.004193	0.002757	1.521014	60.333423

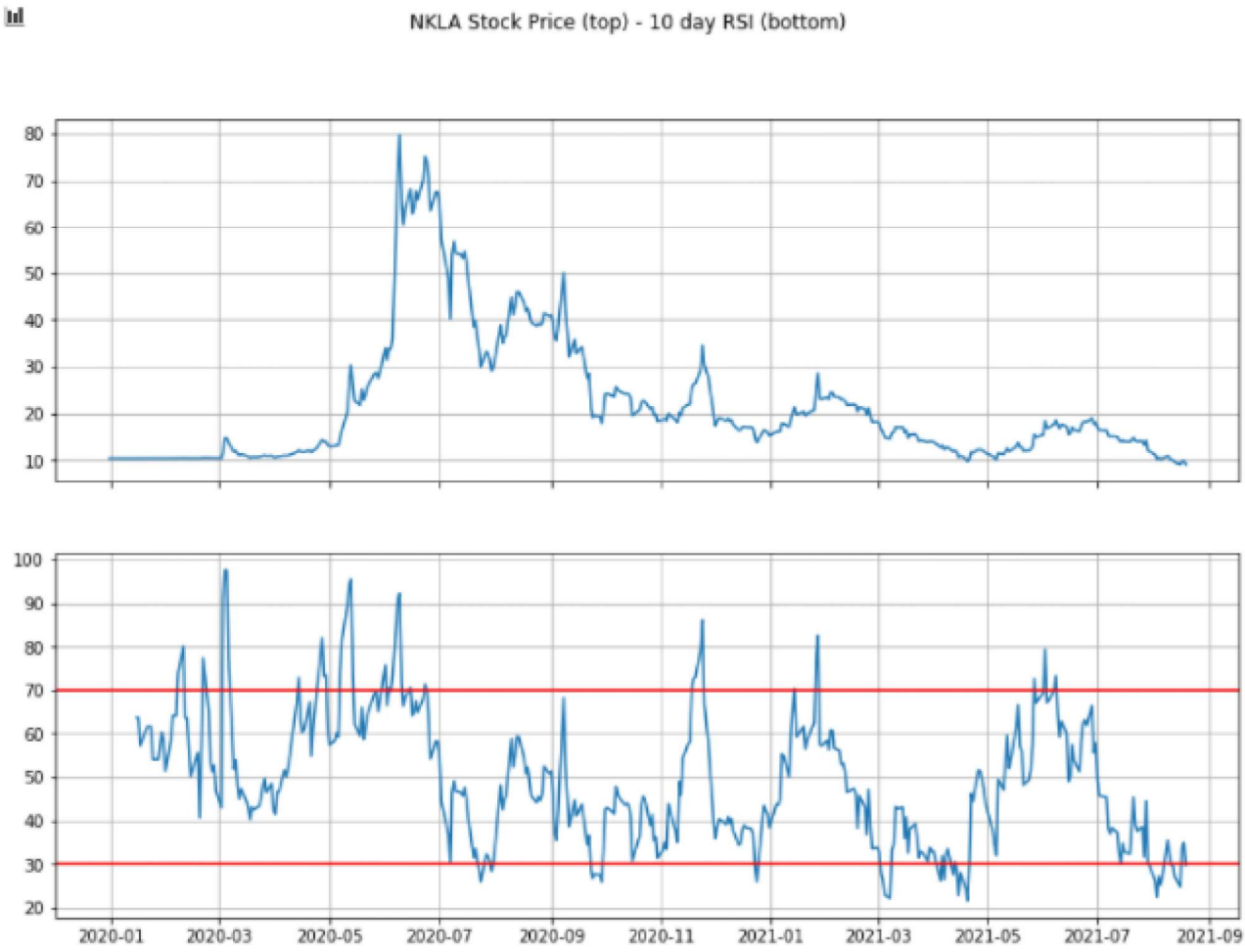
Slika 6. Prikaz podataka nakon poziva funkcije calculateRsi(data)

2.3 Grafički prikaz podataka

Kako smo izračunali sve potrebne podatke, slijedi grafički prikaz cijena NKLA dionica i vrijednosti RSI za prethodno navedeno razdoblje. Grafički prikaz napravljen je pomoću pltRsi(data), a na slici 7. imamo njen ishod.

```
def pltRsi(data):
    fig,axs = plt.subplots(2, sharex=True, figsize=(13,9))
    fig.suptitle('NKLA Stock Price (top) - 10 day RSI (bottom)')
    axs[0].plot(data['Adj Close'])
    axs[1].plot(data['RSI'])
    axs[1].axhline(y=70,color='r',linestyle='-')
    axs[1].axhline(y=30,color='r',linestyle='-')

    axs[0].grid()
    axs[1].grid()
```



Slika 7. Cijena dionica (gore) i desetodnevni RSI (dolje)

2.4 Računanje signala za kupnju i prodaju

Prije nego krenemo sa izračunom signala za kupnju i prodaju, potrebno je podacima dodati sljedeće stupce: Long Tomorrow, Buy Signal, Sell Signal, Buy RSI, Sell RSI te im vrijednost postaviti na np.nan. Stupac Long Tommorow sadrži boolean vrijednosti koje govore hoće li na određeni dan postojati kupnja ili prodaja, dok stupci Buy Signal, Sell Signal te Buy RSI, Sell RSI imaju vrijednost različitu od nule u danima kada se dogodila kupovina odnosno prodaja. Vrijednosti prethodno nabrojanih stupaca koristimo za grafički prikaz signala za kupnju i prodaju na grafovima cijene dionica i vrijednosti RSI.

```
# Calculate the buy & sell signals
## Initialize the columns that we need
data['Long Tomorrow'] = np.nan
data['Buy Signal'] = np.nan
data['Sell Signal'] = np.nan
data['Buy RSI'] = np.nan
data['Sell RSI'] = np.nan
data['Strategy'] = np.nan
```

Nakon ovog neophodnog koraka, na podatke pozivamo funkciju calculateSignals(data) koja sadrži potrebnu logiku za izračun i kreiranje signala za kupovinu, odnosno prodaju. Kada imamo izračunate sve signale, na podatke pozivamo funkciju pltSignals(data) koja navedene signale ucrtava na prikaz sa Slike 7.

```
def calculateSignals(data):
    _data = data
    for _day in range(11,len(_data)):
        _previousDay = _day - 1
        # Calculate "Long Tomorrow" column
        if ((_data['RSI'][_day]<=30) & (_data['RSI'][_previousDay]>30)):
            _data['Long Tomorrow'][_day] = True
        elif ((_data['Long Tomorrow'][_previousDay]==True) & (_data['RSI'
    ][_day]<=70)):
            _data['Long Tomorrow'][_day] = True
        else:
            _data['Long Tomorrow'][_day] = False

        # Calculate "Buy Signal" column
        if ((_data['Long Tomorrow'][_day]==True) & (_data['Long Tomorrow'
    ][_previousDay]==False)):
            _data['Buy Signal'][_day] = _data['Adj Close'][_day]
            _data['Buy RSI'][_day] = _data['RSI'][_day]

        # Calculate "Sell Signal" column
        if((_data['Long Tomorrow'][_day]==False) & (_data['Long Tomorrow'
    ][_previousDay]==True)):
            _data['Sell Signal'][_day] = _data['Adj Close'][_day]
            _data['Sell RSI'][_day] = _data['RSI'][_day]


    return _data
```

```
def pltSignals(data):
    _data = data
    fig, axs = plt.subplots(2, sharex=True, figsize=(13,9))
    fig.suptitle('Stock price(top) & 10day RSI(bottom)')

    ## Chart the stock close price & buy/sell signals:
    axs[0].scatter(_data.index, _data['Buy Signal'], color='green', marker
   ='^', alpha=1)
    axs[0].scatter(_data.index, _data['Sell Signal'], color='red', marker=
   'v', alpha=1)

    axs[0].plot(_data['Adj Close'], alpha = 0.8)
    axs[0].grid()

    ## Chart RSI & buy/sell signals:
    axs[1].scatter(_data.index, _data['Buy RSI'], color='green', marker='^
   ', alpha=1)
    axs[1].scatter(_data.index, _data['Sell RSI'], color='red', marker='v'
   , alpha=1)
    axs[1].axhline(y=70,color='r',linestyle='-')
    axs[1].axhline(y=30,color='r',linestyle='-')

    axs[1].plot(_data['RSI'], alpha = 0.8)
    axs[1].grid()
    return _data
```

Slika 8. Prikaz signala za kupnju (zeleno) i prodaju (crveno) na grafovima cijene dionice (gore) i vrijednosti RSI (dolje)

2.5 Učinak Indeksa relativne snage

Za kraj, analizirajmo RSI. Funkcija analyseRSI(data) računa stanje nakon provedenih kupnji i prodaja.

```
def analyseRSI(data):
    _data = data
    initial_balance = 1000
    balance = initial_balance
    print('Initial budget is', balance, '$ \n' )
    trade_count = _data['Sell Signal'].count()
    print(trade_count, 'trades have been made \n')

    buy_prices = []
    sell_prices = []
    for day in range(11,len(_data)):
        if (_data['Buy Signal'][day] > 0):
            buy_prices.append(_data['Buy Signal'][day])

        if (_data['Sell Signal'][day]> 0):
            sell_prices.append(_data['Sell Signal'][day])

    for i in range(0,trade_count):
        print('**************************************')
        print('-------BUY-------')
        print('BUY PRICE: ', buy_prices[i])
        amount = balance/buy_prices[i]
        balance = 0
        print('STOCKS BOUGHT: ' , amount, 'CURRENT BALANCE: ', balance)


        print("-------SELL-------")
        print('SELL PRICE: ', sell_prices[i])
        balance = amount*sell_prices[i]
        print('STOCKS SOLD: ', amount, 'CURRENT BALANCE', balance)
        amount = 0
        print('**************************************')


    print("Balance after trades were made: ", balance)
    print("RSI lost: ", initial_balance-balance, '$')

    return
```

Nakon poziva analyseRSI(data) na naše podatke, saznajemo kako je RSI zapravo izgubio otprilike 0.6% inicijalnog stanja:

```
Initial budget is 1000 $

3 trades have been made

**************************************
-------BUY-------
BUY PRICE:  33.939998626708984
STOCKS BOUGHT:  29.46376076789387 CURRENT BALANCE:  0
-------SELL-------
SELL PRICE:  25.420000076293945
STOCKS SOLD:  29.46376076789387 CURRENT BALANCE 748.9688009677687
**************************************
**************************************
-------BUY-------
BUY PRICE:  15.029999732971191
STOCKS BOUGHT:  49.831591102743786 CURRENT BALANCE:  0
-------SELL-------
SELL PRICE:  21.309999465942383
STOCKS SOLD:  49.831591102743786 CURRENT BALANCE 1061.9111797865294
**************************************
**************************************
-------BUY-------
BUY PRICE:  16.540000915527344
STOCKS BOUGHT:  64.20260707420115 CURRENT BALANCE:  0
-------SELL-------
SELL PRICE:  15.489999771118164
STOCKS SOLD:  64.20260707420115 CURRENT BALANCE 994.4983688845653
**************************************
Balance after trades were made:  994.4983688845653
RSI lost:  5.501631115434748 $
```

Literatura

[1] Aleksandra Knežević, Članak *Relative Strength Index*. 2017.
https://hrportfolio.hr/analize/view-vijest-relative-strength-index-rsi-46581

[2] Web stranica Investopedia.com:
https://www.investopedia.com/terms/r/rsi.asp

[3] Pandas dokumentacija:
https://pandas.pydata.org/docs/

[4] J.Wilder:
https://en.wikipedia.org/wiki/J._Welles_Wilder_Jr.

[5] Web stranica Stockcharts.com:
https://school.stockcharts.com/doku.php?id=technical_indicators:relative_strength_index_rsi

[6] Web stranica Medium.com:
https://medium.com/analytics-vidhya/momentum-trading-with-macd-and-rsi-yfinance-python-e5203d2e1a8a