ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KINH TẾ - LUẬT
KHOA TÀI CHÍNH - NGÂN HÀNG
CHUYÊN NGÀNH CÔNG NGHỆ - TÀI CHÍNH

Đề tài cuối học kỳ môn Gói phần mềm ứng dụng trong tài chính 1

CHỦ ĐỀ: XÂY DỰNG MÔ HÌNH PHÂN TÍCH KỸ THUẬT DỰ BÁO TÍN HIỆU MUA - BÁN CỔ PHIẾU CÔNG TY GOOGLE GIAI ĐOẠN ĐẦU NĂM 2020 ĐẾN ĐẦU NĂM 2022

Thực hiện bởi:
Sinh viên: PHẠM QUỲNH HƯƠNG
MSSV: K194141724
Lớp: K19414C

Hướng dẫn bởi:
Thầy: NGÔ PHÚ THANH
Giảng viên môn Gói phần mềm ứng dụng tài chính 1

MỤC LỤC

1. Đề tài lựa chọn

- Em lựa chọn 3 đề tài bao gồm:

(1) Xây dựng technical analysic dashboard 9 chỉ số.

(2) Xây dựng mô hình dự báo điểm mua-bán và lợi nhuận bằng backtest của 2 chỉ số MACD và RSI kết hợp, trường hợp áp dụng cho dự báo điểm mua-bán và lợi nhuận của cổ phiếu GOOGL.

(3) Xây dựng mô hình dự báo điểm mua- bán và lợi nhuận bằng backtest của các chỉ số MA (4 đường MA 20, 50, 100, 200) và RSI kết hợp, trường hợp áp dụng cho dự báo điểm mua-bán và lợi nhuận của cổ phiếu GOOGL.

2. Lý do lựa chọn đề tài

(1) Em muốn xây dựng bảng điều khiển TA cho 9 chỉ số: ADX, ATR, CCI, DMI, SMA, EMA, Bollinger band, RSI, MACD để có thể kết hợp 9 chỉ số trong việc phân tích, dự báo sự biến động giá, khối lượng của các cổ phiếu; từ đó có cơ sở cho việc đầu tư cổ phiếu bằng phân tích kỹ thuật bổ trợ cho phân tích cơ bản.

(2) Chỉ số MACD và RSI là 2 chỉ báo động lượng theo xu hướng cực kỳ phổ biến trong phân tích kĩ thuật. Trong khi chỉ số MACD đo lường mối quan hệ giữa hai đường EMA, thì chỉ số RSI đo lường sự thay đổi giá liên quan đến mức cao và mức giá gần đây. Hai chỉ số này thường được sử dụng cùng nhau để cung cấp cho các nhà phân tích kỹ thuật một bức tranh kỹ thuật đầy đủ hơn về một thị trường. Việc nghiên cứu sự kết hợp giữa 2 chỉ báo này để dự báo điểm mua-bán của cổ phiếu là một điều cần thiết để tạo thêm công cụ hỗ trợ phân tích dự báo cho nhà đầu tư.

(3) Chỉ số MA ngắn hạn 20, 50 sẽ cho tín hiệu điểm vào - ra khá tốt nhưng lại có nhược điểm dễ bị nhiễu, cho các tín hiệu mua-bán liên tục. Do đó việc kết hợp đường xu hướng dài hạn MA 100, 200 là cần thiết để nhà đầu tư nắm bắt được xu hướng chính của cổ phiếu. Bên cạnh đó, mô hình kết hợp thêm chỉ báo RSI để đảm bảo rằng nhà đầu tư nắm được mức độ giao dịch, khối lượng mua-bán của cổ phiếu tại thời điểm đó là cao hay thấp, tránh việc bán quá sớm, mua quá muộn.

3. Khái niệm MA, MACD, RSI

3.1. MA

- Đường MA là chỉ báo xu hướng, mục đích chính là để giá đang vận động theo xu hướng tăng, giảm hay không có xu hướng. Nó được xem là chỉ báo chậm, nó không có tác dụng để dự báo mà chủ yếu là sẽ vận động theo diễn biến giá đã được hình

thành, nhìn chung nó có ý nghĩa tương đối. Chỉ báo này được tính dựa trên mức giá đóng cửa trung bình giá trong một khoảng thời gian.

- Có rất nhiều period khác nhau để thiết lập thông số cho đường trung bình di động (MA), việc sử dụng period nào thì đều có ý nghĩa cả.
- Đối với khung đồ thị Daily, chúng ta thường hay sử dụng các period 20, 50, 100, 200 ngày. Thông số 20 ngày MA(20) biểu thị thời gian trong vòng 1 tháng giao dịch; thông số MA(50) biểu thị thời gian giao dịch 1 quý (1 quý là kỳ công bố Báo cáo tài chính của các doanh nghiệp; thông số MA(100) đại diện cho 2 quý và thông số MA(200) đại diện cho 1 năm.

3.2. MACD

- MACD (Moving Average Convergence Divergence – Trung bình động hội tụ phân kỳ) được tính bằng độ chênh lệch giữa 2 trung bình trượt số mũ (chậm 26 ngày và nhanh 12 ngày).

- Chỉ báo MACD sẽ có 2 đường, màu xanh là đường tín hiệu, mà đỏ là đường MACD. Đường MACD giao với đường tín hiệu dự báo về xu hướng giá. Người ta sẽ dựa vào 2 đường này để phân tích kỹ thuật.

- Nếu đường MACD giao với đường tín hiệu từ dưới lên sẽ báo hiệu giá sẽ tăng hơn mức hiện tại. Đây là tín hiệu tốt để các nhà đầu tư mua vào.
- Nếu đường MACD vượt đường tín hiệu từ trên xuống báo hiệu giá đang trên đà giảm. Khi này các nhà đầu tư nên vào lệnh bán.

3.3. RSI

- RSI (Relative Strength Index) hay còn được gọi là chỉ số sức mạnh tương đối.
- Về cơ bản, chỉ số rsi được xây dựng để đo lường mức độ thay đổi giá trong khoảng thời gian gần nhất. Từ đó, giúp các nhà đầu tư xác định thời điểm quá mua, quá bán của thị trường.

- RSI được hiển thị dưới dạng một bộ dao động từ 0 đến 100 và nó là một đồ thị di chuyển giữa 2 điểm cực trị.

- RSI biểu thị vùng quá mua (overbought)

Đường RSI vượt ngưỡng 70 được coi là vùng quá mua. Khi này giá đã đến đỉnh và có xu hướng điều chỉnh giảm giá.

- RSI biểu thị vùng quá bán (oversold)

RSI xuống dưới ngưỡng 30 là vùng quá bán. Khi này giá đang trên đà chạm đáy và sẽ có đợt điều chỉnh để giá tăng trở lại.

4. Quy trình thực hiện đề tài số 1 (TA- dashboard)

4.1. Tải thư viện

Các thư viện được sử dụng:

- numpy, pandas, matplotlib, yfinance, cufflinks, plotly, datetime, ipywidgets

4.2. Nguồn lấy dữ liệu

Nguồn: yfinance

Thời gian: tùy chỉnh (ngày bắt đầu - ngày hôm nay)

4.3. Xây dựng technical analysic dashboard

- Các bước thực hiện:

(1) Cài đặt các thư viện

(2) Tải dữ liệu từ API yfinance

(3) Vẽ đồ thị nến cổ phiếu GOOGL

(4) Xây dựng bảng điều khiển kĩ thuật

+ Tải thư viện

+ Nhập 9 chỉ số

+ Xây dựng hàm tạo giao diện đồ thị cho bảng điều khiển

+ Xây dựng các nút lệnh và hộp lệnh chính

+ Xây dựng các nút lệnh cho từng chỉ số

+ Xây dựng các hộp lệnh phụ, thêm các nút lệnh của các chỉ số vào hộp lệnh phụ

+ Xây dựng các đầu ra tương tác

+ Trình bày kết quả bảng điều khiển

4.4. Code đầy đủ

(1) Cài đặt thư viện

pip install yfinance

```
pip install cufflinks
```

(2) Import thư viện

```
import matplotlib.pyplot as plt
import warnings
import datetime
import yfinance as yf
import cufflinks as cf
from plotly.offline import iplot, init_notebook_mode

%matplotlib inline
%config InlineBackend.figure_format = 'retina'
plt.rcParams['figure.figsize'] = [8, 4.5]
plt.rcParams['figure.dpi'] = 300
warnings.simplefilter(action='ignore', category=FutureWarning)
```

(3) Cài đặt thời gian (ngày bắt đầu là 01/01/2000; ngày kết thúc là ngày hôm nay)

```
start_time = datetime.datetime(2000, 1, 1)
end_time = datetime.datetime.now().date().isoformat()        # today
```

(4) Cài đặt tên cổ phiếu và tải dữ liệu từ API yfinance

```
com_name = 'GOOGL'

df_com = yf.download(com_name,
                     start=start_time,
                     end=end_time,
                     progress=False,
                     auto_adjust=True)
```

(5) Xây dựng đồ thị nến cho cổ phiếu

```
cf.go_offline()
init_notebook_mode(connected=False)

qf = cf.QuantFig(df_com, title=f"{com_name} Stock Price",
```

```
            legend='top', name=com_name)
qf.add_volume()   # Volume
```

(6) Hàm bổ sung để hiển thị đồ thị tương tác iplot() cho trình duyệt Google Colab

```
def configure_plotly_browser_state():
  import IPython
  display(IPython.core.display.HTML('''
        <script src="/static/components/requirejs/require.js"></script>
        <script>
          requirejs.config({
            paths: {
              base: '/static/base',
              plotly: 'https://cdn.plot.ly/plotly-latest.min.js?noext',
            },
          });
        </script>
        '''))
```

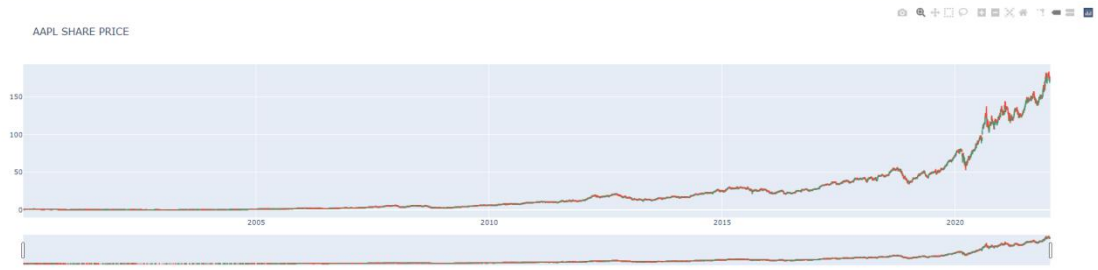
(7) Hiển thị kết quả đồ thị plotly

```
configure_plotly_browser_state()
qf.iplot()
```

(8) Hiển thị kết quả bằng plotly.graph_objects

```
import plotly.graph_objects as go

fig = go.Figure(data=[go.Candlestick(x=df_com.index,
```

```
            open=df_com['Open'],
            high=df_com['High'],
            low=df_com['Low'],
            close=df_com['Close'])])
fig.update_layout(title=f"{com_name} SHARE PRICE")
configure_plotly_browser_state()
fig.show()
```



(9) Import thư viện xây dựng bảng điều khiển

```
import ipywidgets as wd
from ipywidgets import interact, interact_manual
import cufflinks as cf
import pandas as pd
import yfinance as yf
from plotly.offline import iplot, init_notebook_mode

init_notebook_mode()
```

(10) Nhập 5 chỉ số

```
indicators = ['ADX', 'ATR', 'Bollinger Bands', 'CCI', 'DMI', 'EMA', 'MACD', 'SMA', '
RSI']
```

(11) Xây dựng hàm tạo giao diện đồ thị cho bảng điều khiển

```
def ta_dashboard(com, indicator, start_date, end_date, adx_period, atr_period, bb_k, b
b_n, cci_period, cci_upper, cci_lower, dmi_period, multi_ema, ema_periods1, ema_p
eriods2, ema_periods3, macd_fast, macd_slow, macd_signal, rsi_periods, rsi_upper, r
si_lower, multi_sma, sma_periods1, sma_periods2, sma_periods3, vol=False):

    df = yf.download(com,
                     start=start_date,
```

```
                end=end_date,
                progress=False,
                auto_adjust=True)

    qf = cf.QuantFig(df, title=f'Technical analysis dashboard - {com}',
                legend='right', name=f'{com}')

    if vol:
        qf.add_volume()
    if 'ADX' in indicator:
        qf.add_adx(periods=adx_period, color='red')
    if 'ATR' in indicator:
        qf.add_atr(periods=atr_period)
    if 'Bollinger Bands' in indicator:
        qf.add_bollinger_bands(periods=bb_n, boll_std=bb_k)
    if 'CCI' in indicator:
        qf.add_cci(periods=cci_period, cci_upper=cci_upper, cci_lower=cci_lower, sho
wbands=True)
    if 'DMI' in indicator:
        qf.add_dmi(periods=dmi_period)
    if 'EMA' in indicator:
        qf.add_ema(periods=ema_periods1, color='orange')
        if multi_ema:
            qf.add_ema(periods=ema_periods2, color='lightpurple')
            qf.add_ema(periods=ema_periods3, color='grey')
    if 'MACD' in indicator:
        qf.add_macd(fast_period=macd_fast, slow_period=macd_slow, signal_period=m
acd_signal)
    if 'RSI' in indicator:
        qf.add_rsi(periods=rsi_periods, rsi_upper=rsi_upper, rsi_lower=rsi_lower, show
bands=True)
    if 'SMA' in indicator:
        qf.add_sma(periods=sma_periods1, color='red')
        if multi_sma:
```

```
            qf.add_sma(periods=sma_periods2, color='purple')
            qf.add_sma(periods=sma_periods3, color='black')

    configure_plotly_browser_state()
    return qf.iplot()
```

(12) Xây dựng các nút lệnh và hộp lệnh chính

```
stocks_selector = wd.Text(
    value='GOOGL',
    description='Stock Code'
)

indicator_selector = wd.SelectMultiple(
    description='Indicator',
    options=indicators,
    value=[indicators[0]]
)

start_date_selector = wd.DatePicker(
    description='Start Date',
    value=pd.to_datetime('2019-01-01'),
    continuous_update=False
)

end_date_selector = wd.DatePicker(
    description='End Date',
    value=pd.to_datetime(end_time),
    continuous_update=False
)

vol_button = wd.Checkbox(
    description='Volume',
    value = False
)
```

```
main_selector_label = wd.Label('Main parameters',
                               layout=wd.Layout(height='45px'))

main_selector_box = wd.VBox(children=[main_selector_label,
                                      stocks_selector,
                                      vol_button,
                                      indicator_selector,
                                      start_date_selector,
                                      end_date_selector])
```

(13) Xây dựng các nút lệnh cho các chỉ số

- ***ADX indicator***

```
adx_label = wd.Label('Average Directional Index')

adx_period = wd.IntSlider(value=14, min=5, max=100, step=5,
                          description='ADX periods:',
                          continuous_update=False)
adx_period.style.handle_color = 'lightblue'
adx_box = wd.VBox(children=[adx_label, adx_period])
```

- ***ATR indicator***

```
atr_label = wd.Label('Average True Range')

atr_period = wd.IntSlider(value=14, min=5, max=200, step=5,
                          description='ATR periods:',
                          continuous_update=False)
atr_period.style.handle_color = 'lightblue'
atr_box = wd.VBox(children=[atr_label, atr_period])
```

- ***SMA indicator***

```
sma_label = wd.Label('Simple Moving Average')

multi_sma = wd.Checkbox(description='Multiple SMA periods', value = False)

sma_periods1 = wd.IntSlider(value=30, min=2, max=100, step=1,
                          description='Short-term SMA periods:',
```

```
                    continuous_update=False)

sma_periods2 = wd.IntSlider(value=100, min=2, max=200, step=2,
                    description='Medium-term SMA periods:',
                    continuous_update=False)

sma_periods3 = wd.IntSlider(value=200, min=2, max=200, step=2,
                    description='Long-term SMA periods:',
                    continuous_update=False)

sma_periods1.style.handle_color = 'lightblue'
sma_periods2.style.handle_color = 'lightblue'
sma_periods3.style.handle_color = 'lightblue'

sma_box = wd.VBox(children=[sma_label, multi_sma, sma_periods1, sma_periods2,
sma_periods3])
```

- ***Bollinger band***

```
bb_label = wd.Label('Bollinger Bands')

n_param = wd.IntSlider(value=20, min=1, max=100, step=5,
                description='N:', continuous_update=False)

k_param = wd.FloatSlider(value=2, min=0.5, max=4, step=0.5,
                description='k:', continuous_update=False)

n_param.style.handle_color = 'lightblue'
k_param.style.handle_color = 'lightblue'

bollinger_box = wd.VBox(children=[bb_label, n_param, k_param])
```

- ***CCI indicator***

```
cci_label = wd.Label('Commodity Chanel Index')

cci_period = wd.IntSlider(value=20, min=2, max=100, step=1,
                    description='CCI periods:',
```

```
                    continuous_update=False)

cci_upper = wd.IntSlider(value=100, min=50, max=200, step=5,
                    description='Upper Thr:',
                    continuous_update=False)

cci_lower = wd.IntSlider(value=-100, min=-200, max=-50, step=5,
                    description='Lower Thr:',
                    continuous_update=False)
cci_period.style.handle_color = 'lightblue'
cci_upper.style.handle_color = 'lightblue'
cci_lower.style.handle_color = 'lightblue'

cci_box = wd.VBox(children=[cci_label, cci_period,
                     cci_upper, cci_lower])
```

- ***DMI indicator***

```
dmi_label = wd.Label('Directional Movement Index')

dmi_period = wd.IntSlider(value=14, min=5, max=200, step=5,
                    description='DMI periods:',
                    continuous_update=False)
dmi_period.style.handle_color = 'lightblue'
dmi_box = wd.VBox(children=[dmi_label, dmi_period])
```

- ***EMA indicator***

```
ema_label = wd.Label('Exponential Moving Average')

multi_ema = wd.Checkbox(description='Multiple EMA periods', value = False)

ema_periods1 = wd.IntSlider(value=20, min=2, max=100, step=1,
                    description='Short-term EMA periods:',
                    continuous_update=False)

ema_periods2 = wd.IntSlider(value=50, min=2, max=200, step=2,
                    description='Medium-term EMA periods:',
```

```
                    continuous_update=False)

ema_periods3 = wd.IntSlider(value=200, min=2, max=200, step=2,
                    description='Long-term EMA periods:',
                    continuous_update=False)

ema_periods1.style.handle_color = 'lightblue'
ema_periods2.style.handle_color = 'lightblue'
ema_periods3.style.handle_color = 'lightblue'

ema_box = wd.VBox(children=[ema_label, multi_ema, ema_periods1, ema_periods2,
 ema_periods3])
```

- ***MACD indicator***

```
macd_label = wd.Label('Moving Average Convergence Divergence')

macd_fast = wd.IntSlider(value=12, min=2, max=100, step=1,
                    description='Fast avg:',
                    continuous_update=False)

macd_slow = wd.IntSlider(value=26, min=2, max=100, step=1,
                    description='Slow avg:',
                    continuous_update=False)

macd_signal = wd.IntSlider(value=9, min=2, max=100, step=1,
                     description='MACD signal:',
                     continuous_update=False)

macd_fast.style.handle_color = 'lightblue'
macd_slow.style.handle_color = 'lightblue'
macd_signal.style.handle_color = 'lightblue'

macd_box = wd.VBox(children=[macd_label, macd_fast,
                      macd_slow, macd_signal])
```

- ***RSI indicator***

```
rsi_label = wd.Label('Relative Strength Index')

rsi_periods = wd.IntSlider(value=14, min=2, max=100, step=1,
                           description='RSI periods:',
                           continuous_update=False)

rsi_upper = wd.IntSlider(value=70, min=1, max=100, step=1,
                         description='Upper Thr:',
                         continuous_update=False)

rsi_lower = wd.IntSlider(value=30, min=1, max=100, step=1,
                         description='Lower Thr:',
                         continuous_update=False)
rsi_periods.style.handle_color = 'lightblue'
rsi_upper.style.handle_color = 'lightblue'
rsi_lower.style.handle_color = 'lightblue'

rsi_box = wd.VBox(children=[rsi_label, rsi_periods,
                            rsi_upper, rsi_lower])
```

(14) Xây dựng các hộp lệnh phụ, thêm các nút lệnh của các chỉ số vào hộp lệnh phụ

```
sec_box_1 = wd.VBox([adx_box, bollinger_box, cci_box])
sec_box_2 = wd.VBox([atr_box, macd_box, sma_box])
sec_box_3 = wd.VBox([dmi_box, rsi_box, ema_box])

secondary_selector_box = wd.HBox([sec_box_1, sec_box_2, sec_box_3])
```

(15) Xây dựng các đầu ra tương tác

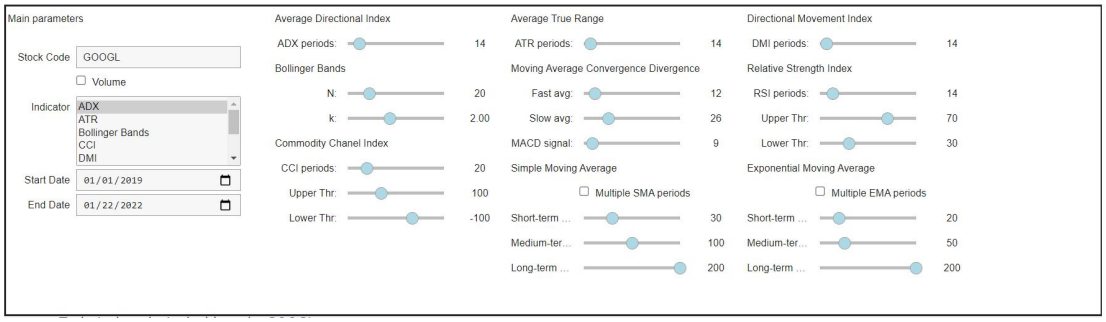
```
controls_dict = {'com':stocks_selector, 'indicator':indicator_selector, 'start_date':start_
date_selector, 'end_date':end_date_selector,
                 'adx_period': adx_period,
                 'atr_period': atr_period,
                 'bb_k':k_param, 'bb_n':n_param,
                 'cci_period': cci_period, 'cci_upper': cci_upper, 'cci_lower': cci_lower,
                 'dmi_period': dmi_period,
```

```
            'multi_ema': multi_sma, 'ema_periods1': sma_periods1, 'ema_periods2': sm
a_periods2, 'ema_periods3': sma_periods3,
            'macd_fast': macd_fast, 'macd_slow': macd_slow, 'macd_signal': macd_sig
nal,
            'rsi_periods': rsi_periods, 'rsi_upper': rsi_upper,'rsi_lower': rsi_lower,
            'multi_sma': multi_sma, 'sma_periods1': sma_periods1, 'sma_periods2': sm
a_periods2, 'sma_periods3': sma_periods3,
            'vol': vol_button}

from ipywidgets import Layout, Button, Box

items_layout = Layout( width='auto')

box_layout = Layout(display='flex', flex_flow='row', align_items='stretch', border='so
lid', width='100%', height="400px", grid_gap="40px")

ui = wd.HBox([main_selector_box, secondary_selector_box], layout=box_layout)
out = wd.interactive_output(ta_dashboard, controls_dict)
```

(16) Trình bày bảng điều khiển

```
display(ui, out)
```

4.5. Kết quả



4.6. Diễn giải, đánh giá kết quả

- Bảng điều khiển hiện thị tùy chọn 9 chỉ số (có thể chọn nhiều chỉ số cùng lúc)

    + Nút lệnh Stock Code có thể tùy chỉnh cổ phiếu mình mong muốn.

    + Volume (tick vào nếu muốn hiển thị khối lượng giao dịch)

- Đối với 2 chỉ số SMA và EMA, tick vào Multiple periods nếu muốn sử dụng các đường SMA, EMA kết hợp (Long-term, Medium-term, Short-term).

5. Quy trình thực hiện đề tài số 2 (backtest for the combination of MACD and RSI indicators)

5.1. Tải thư viện

- Các thư viện được sử dụng:

datetime, backtrader, yfinance, matplotlib.

5.2. Nguồn lấy dữ liệu

Nguồn: API yfinance

Thời gian: tùy chỉnh (từ 01-01-2020 đến hôm nay)

5.3. Thư viện backtrader là gì?

- Backtrader là một thư viện Python hỗ trợ phát triển chiến lược và thử nghiệm cho các nhà giao dịch trên thị trường tài chính.

- Nó là một khuôn khổ mã nguồn mở cho phép thử nghiệm chiến lược trên dữ liệu lịch sử. Hơn nữa, nó có thể được sử dụng để tối ưu hóa các chiến lược, tạo ra các đồ thị tín hiệu mua-bán trực quan và thậm chí có thể được sử dụng để giao dịch trực tiếp.

- Sử dụng Backtrader giúp tiết kiệm vô số giờ viết mã để thử nghiệm các chiến lược thị trường.

5.4. Chạy mô hình

- Các bước thực hiện:

+ Cài đặt thư viện

+ Import thư viện

+ Cài đặt tên cổ phiếu, thời gian, tải dữ liệu

+ Xây dựng chiến lược giao dịch

+ Nhập số dư đầu kỳ

+ Chạy mô hình

+ Trả kết quả số dư cuối kỳ, lãi / lỗ

+ Trình bày biểu đồ giá, chỉ số RSI và MACD kết hợp, có thể hiện tín hiệu mua-bán.

+ Trình bày biểu đồ tỷ suất sinh lợi đầu tư hàng ngày

5.5. Code đầy đủ

(1) Cài đặt thư viện

```
pip install yfinance
pip install backtrader
```

(2) Import thư viện

```
from datetime import datetime
import backtrader as bt
import yfinance as yf
import matplotlib.pyplot as plt
```

(3) Cài đặt tên cổ phiếu, thời gian, tải dữ liệu từ API yfinance

```
name = 'GOOGL'
start_date = '2020-01-01'
end_date = datetime.now().date().isoformat()  # Today
data = bt.feeds.PandasData(dataname=yf.download(name,
                        start=start_date,
                        end=end_date,
                        progress=False,
                        auto_adjust=True))
```

(4) Xây dựng chiến lược giao dịch kết hợp 2 chỉ số MACD và RSI

```
class TheStrategy(bt.Strategy):
    params = (
```

```
    # Standard MACD Parameters
    ('macd1', 12),
    ('macd2', 26),
    ('macdsig', 9),
    ('rsiperiod', 14),  # ATR Period (standard)
)
def __init__(self):
    # keep track of close price in the series
    self.data_close = self.datas[0].close
    self.data_open = self.datas[0].open
    # keep track of pending orders/buy price/buy commission
    self.order = None
    self.price = None
    self.comm = None
    # add MACD indicator
    self.macd = bt.indicators.MACD(self.data,
                                   period_me1=self.p.macd1,
                                   period_me2=self.p.macd2,
                                   period_signal=self.p.macdsig)
    # Cross of macd.macd and macd.signal
    self.mcross = bt.indicators.CrossOver(self.macd.macd, self.macd.signal)
    # add RSI indicator
    self.rsi = bt.indicators.RSI_SMA(self.data_close, period=14)

def log(self, txt):
  dt = self.datas[0].datetime.date(0).isoformat()
  print(f'{dt}, {txt}')

def notify_order(self, order):
    if order.status in [order.Submitted, order.Accepted]:
        # order already submitted/accepted - no action required
        return
    # report executed order
    if order.status in [order.Completed]:
```

```
            if order.isbuy():
                self.log(
                    f'BUY EXECUTED ---
 Price: {order.executed.price:.2f}, Cost: {order.executed.value:.2f}, Commission: {or
der.executed.comm:.2f}'
                )
                self.price = order.executed.price
                self.comm = order.executed.comm
            else:
                self.log(
                    f'SELL EXECUTED ---
 Price: {order.executed.price:.2f}, Cost: {order.executed.value:.2f}, Commission: {or
der.executed.comm:.2f}'
                )

        # report failed order
        elif order.status in [order.Canceled, order.Margin, order.Rejected]:
            self.log('Order Failed')
        # set no pending order
        self.order = None

    def notify_trade(self, trade):
        if not trade.isclosed:
            return
        self.log(f'OPERATION RESULT ---
 Gross: {trade.pnl:.2f}, Net: {trade.pnlcomm:.2f}')

    def next(self):
        if not self.position:  # not in the market
            if self.mcross[0] == 1.0 and self.rsi < 70:
                # calculate the max number of shares ('all-in')
                size = int(self.broker.getcash() / self.datas[0].open)
                # buy order
```

```
            self.log(f'BUY CREATED ---
 Size: {size}, Cash: {self.broker.getcash():.2f}, Open: {self.data_open[0]}, Close: {sel
f.data_close[0]}')
            self.order = self.buy(size=size)

        else:  # in the market
            if self.mcross[0] == -1.0 and self.rsi > 60:
                # sell order
                self.log(f'SELL CREATED --- Size: {self.position.size}')
                self.order = self.sell(size=self.position.size)
```

(5) Nhập số dư đầu kỳ và chạy mô hình

```
cash = 10000.0
cerebro = bt.Cerebro(stdstats = False)
cerebro.adddata(data)
cerebro.broker.setcash(cash)
cerebro.broker.setcommission(commission=0.001)
cerebro.addstrategy(TheStrategy)
cerebro.addobserver(bt.observers.BuySell)
cerebro.addobserver(bt.observers.Value)
cerebro.addanalyzer(bt.analyzers.Returns, _name='returns')
cerebro.addanalyzer(bt.analyzers.TimeReturn, _name='time_return')
```

(6) Trả kết quả số dư cuối kỳ, lãi/ lỗ

```
start_portfolio_value = cerebro.broker.getvalue()
print(f'Starting Portfolio Value: {cerebro.broker.getvalue():.2f}')
backtest_result = cerebro.run()
end_portfolio_value = cerebro.broker.getvalue()
print(f'Final Portfolio Value: {cerebro.broker.getvalue():.2f}')
pnl = end_portfolio_value - start_portfolio_value
print(f'Profit and Loss: {pnl:.2f}')
```

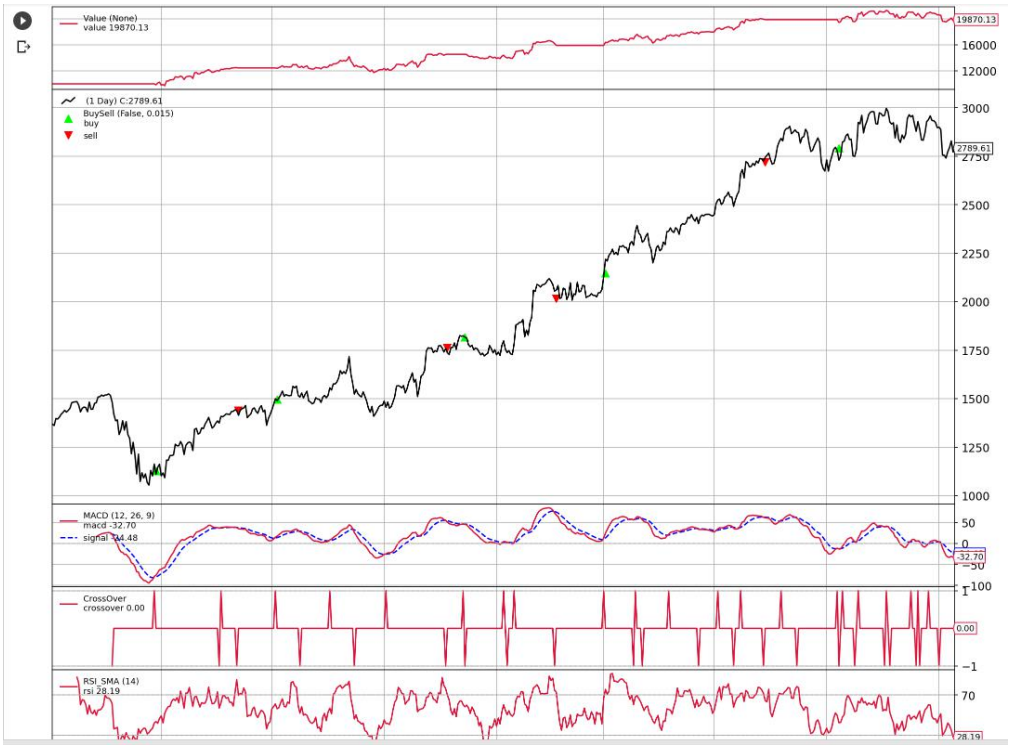
(7) Trình bày biểu đồ giá, chỉ số MACD, RSI có thể hiện tín hiệu mua-bán

```
%matplotlib inline
plt.rcParams['figure.figsize'] = [15, 12]
```

```
plt.rcParams.update({'font.size': 12})
cerebro.plot(iplot=False, volume=False)
```

```
Starting Portfolio Value: 10000.00
2020-03-26, BUY CREATED --- Size: 8, Cash: 10000.00, Open: 1114.719970703125, Close: 1162.9200439453125
2020-03-27, BUY EXECUTED --- Price: 1127.47, Cost: 9019.76, Commission: 9.02
2020-06-03, SELL CREATED --- Size: 8
2020-06-04, SELL EXECUTED --- Price: 1436.78, Cost: 9019.76, Commission: 11.49
2020-06-04, OPERATION RESULT --- Gross: 2474.48, Net: 2453.97
2020-07-06, BUY CREATED --- Size: 8, Cash: 12453.97, Open: 1488.1500244140625, Close: 1499.6500244140625
2020-07-07, BUY EXECUTED --- Price: 1496.13, Cost: 11969.04, Commission: 11.97
2020-11-19, SELL CREATED --- Size: 8
2020-11-20, SELL EXECUTED --- Price: 1762.00, Cost: 11969.04, Commission: 14.10
2020-11-20, OPERATION RESULT --- Gross: 2126.96, Net: 2100.89
2020-12-04, BUY CREATED --- Size: 7, Cash: 14554.86, Open: 1820.219970703125, Close: 1823.760009765625
2020-12-07, BUY EXECUTED --- Price: 1815.55, Cost: 12708.85, Commission: 12.71
2021-02-22, SELL CREATED --- Size: 7
2021-02-23, SELL EXECUTED --- Price: 2013.99, Cost: 12708.85, Commission: 14.10
2021-02-23, OPERATION RESULT --- Gross: 1389.08, Net: 1362.27
2021-04-01, BUY CREATED --- Size: 7, Cash: 15917.13, Open: 2092.25, Close: 2129.780029296875
2021-04-05, BUY EXECUTED --- Price: 2147.15, Cost: 15030.05, Commission: 15.03
2021-08-11, SELL CREATED --- Size: 7
2021-08-12, SELL EXECUTED --- Price: 2719.51, Cost: 15030.05, Commission: 19.04
2021-08-12, OPERATION RESULT --- Gross: 4006.52, Net: 3972.45
2021-10-11, BUY CREATED --- Size: 7, Cash: 19889.59, Open: 2785.840087890625, Close: 2778.280029296875
2021-10-12, BUY EXECUTED --- Price: 2789.60, Cost: 19527.20, Commission: 19.53
Final Portfolio Value: 19870.13
Profit and Loss: 9870.13
```

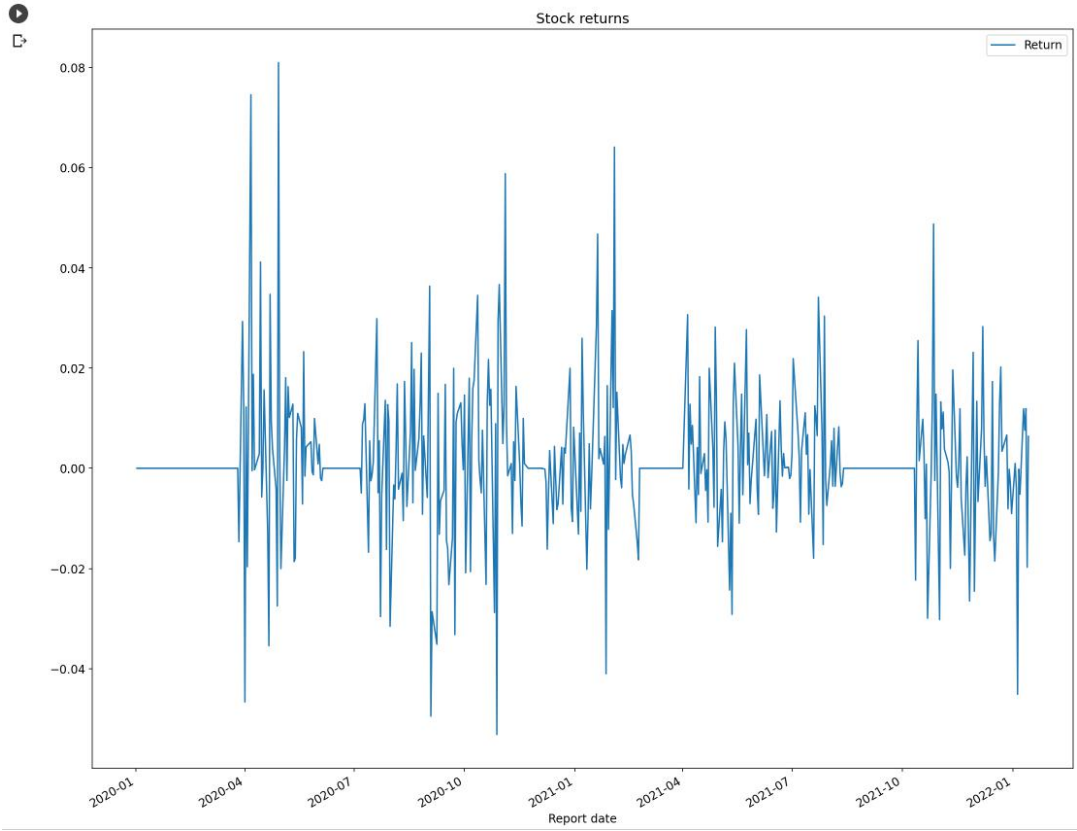
5.6. Kết quả



5.7. Tỷ suất sinh lợi hàng ngày

```
returns_dict = backtest_result[0].analyzers.time_return.get_analysis()
returns_df = pd.DataFrame(list(returns_dict.items()),
                          columns = ['Report date', 'Return']) \
             .set_index('Report date')
```

```
returns_df.plot(title='Stock returns')

plt.tight_layout()
plt.show()
```



5.8. Diễn giải , đánh giá kết quả

- MACD cho tín hiệu xu hướng của thị trường khá tốt nhưng bị 1 nhược điểm là có độ trễ so với thị trường, tín hiệu phát chậm hơn. Do đó sử dụng riêng 1 chỉ số MACD dễ bị nhiễu và không phù hợp với thị trường có xu hướng.

- Vì vậy, sự kết hợp 2 chỉ số MACD và RSI cho tín hiệu mua-bán khá tốt đối với cổ phiếu GOOGL trong giai đoạn từ năm 2020 đến nay. Trong khi MACD sẽ phát tín hiệu mua-bán khi đường MACD cắt đường tín hiệu thì RSI lại đặt điều kiện để nhận biết điểm quá mua-quá bán.

+ Nếu thị trường quá mua trong khi MACD cho tín hiệu bán thì lệnh bán lập tức được thực hiện.

+ Ngược lại, nếu thị trường quá bán trong khi MACD cho tín hiệu mua thì lệnh mua lập tức được thực hiện.

- Tuy nhiên đối với cổ phiếu GOOGL, không có quá nhiều giai đoạn quá bán nên chiến lược giao dịch này sẽ được điều chỉnh như sau:

+ Nếu MACD cho tín hiệu mua và rsi < 70 thì thực hiện lệnh mua.

+ Nếu MACD cho tín hiệu bán và rsi > 70 thì thực hiện lệnh bán.

- Tuy theo từng cổ phiếu và thị trường giao dịch có các đặc trưng riêng biệt mà nhà đầu tư cần tùy chỉnh các chiến lược giao dịch, chỉ số, tham số đầu vào,… cho tối ưu để tìm ra mô hình dự báo phù hợp nhất đối với cổ phiếu mình lựa chọn.

6. Quy trình thực hiện đề tài số 3 (backtest for the combination of 4 MA lines and RSI indicators)

6.1. Tải thư viện

- Các thư viện được sử dụng:

datetime, backtrader, yfinance, matplotlib, pandas.

6.2. Nguồn lấy dữ liệu

Nguồn: API yfinance

Thời gian: tùy chỉnh (từ 01-01-2019 đến hôm nay)

6.3. Chạy mô hình

- Các bước thực hiện:

+ Cài đặt thư viện

+ Import thư viện

+ Cài đặt tên cổ phiếu, thời gian, tải dữ liệu

+ Xây dựng chiến lược giao dịch

+ Nhập số dư đầu kỳ

+ Chạy mô hình

+ Trả kết quả số dư cuối kỳ, lãi / lỗ

+ Trình bày biểu đồ giá, chỉ số RSI và MACD kết hợp, có thể hiện tín hiệu mua-bán.

+ Trình bày biểu đồ tỷ suất sinh lợi đầu tư hàng ngày

6.4. Code đầy đủ

(1) Cài đặt tên cổ phiếu, thời gian, tải dữ liệu từ API yfinance

```
name = 'GOOGL'
start_date1 = '2019-01-01'
end_date = datetime.now().date().isoformat()  # Today
data1 = bt.feeds.PandasData(dataname=yf.download(name,
```

```
                    start=start_date1,
                    end=end_date,
                    progress=False,
                    auto_adjust=True))
```

(2) Xây dựng chiến lược giao dịch kết hợp 2 chỉ số MACD và RSI

```
class MAcrossover(bt.Strategy):
    # Moving average parameters
    params = (('sma20',20),('sma50',50), ('sma100', 100), ('sma200', 200), ('rsiperiod', 1
4), )

    def log(self, txt, dt=None):
        dt = dt or self.datas[0].datetime.date(0)
        print(f'{dt.isoformat()} {txt}') # Comment this line when running optimization

    def __init__(self):
        self.data_close = self.datas[0].close
        self.data_open = self.datas[0].open

        # Order variable will contain ongoing order details/status
        self.order = None
        self.price = None
        self.comm = None

        # Instantiate moving averages
        self.sma20 = bt.ind.SMA(self.datas[0], period=self.params.sma20)
        self.sma50 = bt.ind.SMA(self.datas[0], period=self.params.sma50)
        self.sma100 = bt.ind.SMA(self.datas[0], period=self.params.sma100)
        self.sma200 = bt.ind.SMA(self.datas[0], period=self.params.sma200)

        # add RSI indicator
        self.rsi = bt.indicators.RSI_SMA(self.data_close, period=14)

    def notify_order(self, order):
        if order.status in [order.Submitted, order.Accepted]:
```

```
        # An active Buy/Sell order has been submitted/accepted - Nothing to do
            return

        # Check if an order has been completed
        # Attention: broker could reject order if not enough cash
        if order.status in [order.Completed]:
            if order.isbuy():
                self.log(f'BUY EXECUTED ---
 Price: {order.executed.price:.2f}, Cost: {order.executed.value:.2f}, Commission: {or
der.executed.comm:.2f}')
            elif order.issell():
                self.log(f'SELL EXECUTED ---
 Price: {order.executed.price:.2f}, Cost: {order.executed.value:.2f}, Commission: {or
der.executed.comm:.2f}')
            self.bar_executed = len(self)

        elif order.status in [order.Canceled, order.Margin, order.Rejected]:
            self.log('Failed Order')

        # Reset orders
        self.order = None

    def notify_trade(self, trade):
        if not trade.isclosed:
            return

        self.log(f'OPERATION RESULT ---
 Gross: {trade.pnl:.2f}, Net: {trade.pnlcomm:.2f}')

    def next(self):
        # Check for open orders
        if self.order:
            return
```

```
        # Check if we are in the market
        if not self.position:
            #If the 20 SMA is above the 50 SMA
            if (self.sma20[0] > self.sma50[0]) and (self.sma20[-1] < self.sma50[-
1]) and (self.sma100 > self.sma200) and (self.rsi < 70):
                self.log(f'BUY CREATED ---
 Cash: {self.broker.getcash():.2f}, Open: {self.data_open[0]}, Close: {self.data_close[
0]}')
                # Keep track of the created order to avoid a 2nd order
                self.order = self.buy(size = int(self.broker.getcash() / self.datas[0].open))


        else:
                # We are already in the market
            if (self.sma20[0] < self.sma50[0]) and (self.sma20[-1] > self.sma50[-
1]) and (self.rsi > 70):
                self.log(f'SELL CREATED')
                # Keep track of the created order to avoid a 2nd order
                self.order = self.sell(size = self.position.size)
```

(3) Nhập số dư đầu kỳ và chạy mô hình

```
cash = 10000.0
cerebro1 = bt.Cerebro(stdstats = False)
cerebro1.adddata(data1)
cerebro1.broker.setcash(cash)
cerebro1.broker.setcommission(commission=0.001)
cerebro1.addstrategy(MAcrossover)
cerebro1.addobserver(bt.observers.BuySell)
cerebro1.addobserver(bt.observers.Value)
cerebro1.addanalyzer(bt.analyzers.Returns, _name='returns by 4 MA lines and RSI an
alysis')
cerebro1.addanalyzer(bt.analyzers.TimeReturn, _name='time_return')
```

(8) Trả kết quả số dư cuối kỳ, lãi/ lỗ

```
start_portfolio_value1 = cerebro1.broker.getvalue()
print(f'Starting Portfolio Value: {cerebro1.broker.getvalue():.2f}')
```

```
backtest_result1 = cerebro1.run()
end_portfolio_value1 = cerebro1.broker.getvalue()
print(f'Final Portfolio Value: {cerebro1.broker.getvalue():.2f}')
pnl1 = end_portfolio_value1 - start_portfolio_value1
print(f'Profit and Loss: {pnl1:.2f}')
```

```
Starting Portfolio Value: 10000.00
2020-04-30 BUY CREATED --- Cash: 10000.00, Open: 1331.3599853515625, Close: 1346.699951171875
2020-05-01 BUY EXECUTED --- Price: 1324.09, Cost: 9268.63, Commission: 9.27
Final Portfolio Value: 20249.37
Profit and Loss: 10249.37
```

(4) Trình bày biểu đồ giá, chỉ số MA và RSI có tín hiệu mua-bán

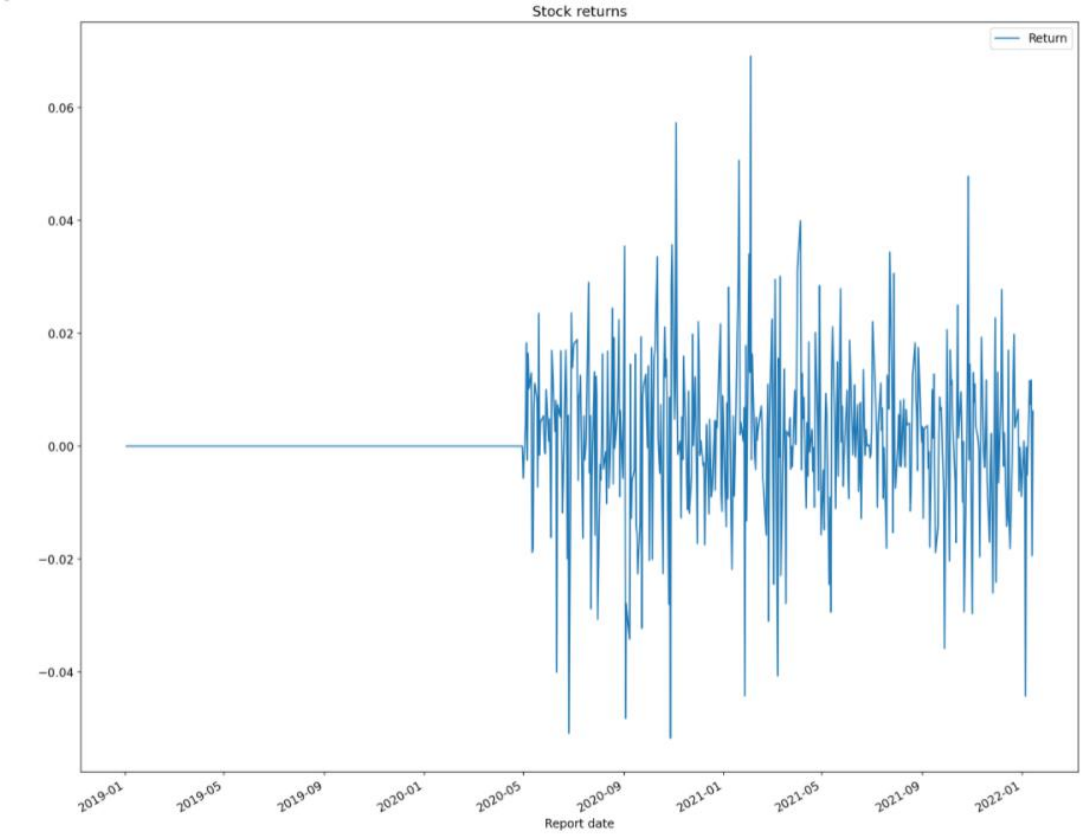
```
%matplotlib inline
plt.rcParams['figure.figsize'] = [15, 12]
plt.rcParams.update({'font.size': 12})

cerebro1.plot(iplot=False, volume=False)
```

6.5. Kết quả

6.6. Tỷ suất sinh lợi hằng ngày

```
returns_dict1 = backtest_result1[0].analyzers.time_return.get_analysis()
returns_df1 = pd.DataFrame(list(returns_dict1.items()),
```

```
                    columns = ['Report date', 'Return']) \
            .set_index('Report date')
returns_df1.plot(title='Stock returns')

plt.tight_layout()
plt.show()
```



6.7. Diễn giải, đánh giá kết quả

- MA cho tín hiệu xu hướng của thị trường khá tốt nhưng bị 1 nhược điểm là có độ trễ so với thị trường, tín hiệu phát chậm hơn. Do đó sử dụng riêng 1 chỉ số MA dễ bị nhiễu và không phù hợp với thị trường có xu hướng.

- Vì vậy, sự kết hợp 2 chỉ số MA và RSI cho tín hiệu mua-bán khá tốt đối với cổ phiếu GOOGL trong giai đoạn từ năm 2020 (vì MA200 cần dữ liệu xa hơn so với các MA ngắn hạn, nên ta sẽ lấy dữ liệu từ 01-01-2019) đến nay. Trong khi MA sẽ phát tín hiệu mua-bán khi đường MA20 cắt lên đường MA50 thì RSI lại đặt điều kiện để nhận biết điểm quá mua-quá bán. Bên cạnh đó, MA100 và MA200 giúp nhận diện xu hướng chính của cổ phiếu, tránh hiện tượng nhiều lệnh mua-bán lặp lại liên tục cho tín hiệu nheiễu của MA ngắn hạn).

+ Nếu thị trường quá mua trong khi MA cho tín hiệu bán thì lệnh bán lập tức được thực hiện.

+ Ngược lại, nếu thị trường quá bán trong khi MA cho tín hiệu mua thì lệnh mua lập tức được thực hiện.

- Tuy nhiên đối với cổ phiếu GOOGL, không có quá nhiều giai đoạn quá bán nên chiến lược giao dịch này sẽ được điều chỉnh như sau:

+ Nếu MA cho tín hiệu mua và rsi < 70 thì thực hiện lệnh mua.

+ Nếu MA cho tín hiệu bán và rsi > 70 thì thực hiện lệnh bán.

- Tuy theo từng cổ phiếu và thị trường giao dịch có các đặc trưng riêng biệt mà nhà đầu tư cần tùy chỉnh các chiến lược giao dịch, chỉ số, tham số đầu vào,… cho tối ưu để tìm ra mô hình dự báo phù hợp nhất đối với cổ phiếu mình lựa chọn.

7. Nguồn tham khảo

1) https://ipywidgets.readthedocs.io/en/latest/examples/Widget%20Styling.html

2) https://ipywidgets.readthedocs.io/en/latest/examples/Layout%20Templates.html

3) https://www.backtrader.com/blog/posts/2016-07-30-macd-settings/macd-settings/

4) https://jpoles1.github.io/cufflinks/html/_modules/cufflinks/colors.html

5) https://github.com/santosjorge/cufflinks/blob/master/cufflinks/quant_figure.py

6) Jignesh Davda (2021), Backtrader for Backtesting (Python) – A Complete Guide, https://algotrading101.com/learn/backtrader-for-backtesting/