**ІНДИВІДУАЛЬНИЙ ПРОЄКТ**

**Тема:** Індивідуальний проєкт.

**Мета:** Оптимально підібрати прибуткову торгову стратегію на основі сигналів технічних індикаторів і дати їй теоретичне обґрунтування. Створити діаграми прецедентів, класів та взаємодії. Переписати технічний індикатор мовою Python.

**Завдання 1:**

Розробити та реалізувати автоматизовану торгову стратегію, яка використовує поєднання технічних індикаторів для генерації сигналів на вхід у позиції (long та short) з чітко визначеними умовами та фіксованим ризик-менеджментом (TP/SL).

У сучасному трейдингу застосування комбінацій індикаторів дозволяє фільтрувати помилкові сигнали та підвищити точність входів у позиції.

Ця стратегія побудована на поєднанні трьох основних індикаторів:

1. EMA (Exponential Moving Average) — трендовий індикатор, який допомагає визначити загальний напрямок ринку. EMA реагує швидше за SMA і служить фільтром: ми купуємо лише тоді, коли ціна вище EMA50, і продаємо лише коли вона нижча.
2. RSI (Relative Strength Index) — осцилятор, що вимірює швидкість і зміну цінових рухів. Значення нижче 30 вказує на перепроданість (потенційний розворот вгору), вище 70 — на перекупленість (потенційний розворот вниз).
3. MACD (Moving Average Convergence Divergence) — індикатор імпульсу, який показує взаємозв’язок між двома ковзними середніми. Перетин лінії MACD і сигнальної лінії дає потужний імпульсний сигнал на вхід.

Таке поєднання дає змогу врахувати напрямок тренду, імпульс і стан ринку (перекупленість/перепроданість), що значно підвищує ефективність торгових сигналів.

Цей проєкт присвячений розробці автоматизованої торгової стратегії, яка використовує поєднання технічних індикаторів для визначення точок входу та виходу з ринку. Основна мета полягає у створенні прибуткової системи, що базується на логічному поєднанні сигналів від EMA, RSI та MACD — трьох найбільш популярних індикаторів у сучасному технічному аналізі.

Ідея стратегії полягає в тому, щоб виявляти моменти, коли ринок знаходиться у стані короткострокового перепродавання або перекупленості, при цьому зберігаючи напрям загального тренду. EMA (експоненціальне ковзне середнє) слугує фільтром, який дозволяє уникати торгівлі проти ринку. Якщо ціна знаходиться вище EMA з періодом 50, ми вважаємо, що на ринку панує висхідний тренд, і розглядаємо лише довгі угоди. Якщо нижче — лише короткі. Індикатор RSI застосовується для виявлення станів перепроданості та перекупленості: перетин рівня 30 знизу вгору розглядається як потенційний сигнал до покупки, а перетин рівня 70 зверху вниз — як сигнал до продажу. MACD у свою чергу допомагає визначити момент, коли імпульс змінюється. Перетин лінії MACD сигнальної лінії знизу вгору підтверджує готовність ринку до зростання, а перетин зверху вниз — до падіння.

Вхід у довгу позицію відбувається лише тоді, коли RSI перетинає рівень 30 вгору, MACD перетинає сигнальну лінію вгору, а ціна знаходиться вище EMA50. Аналогічно, коротка позиція відкривається, якщо RSI перетинає рівень 70 вниз, MACD сигнальну лінію вниз, а ціна закріплюється нижче EMA50. Таким чином, кожна умова доповнює іншу: RSI сигналізує про потенційний розворот, MACD підтверджує імпульс, а EMA визначає, чи варто взагалі відкривати позицію в цьому напрямі.

Вихід із ринку здійснюється за фіксованими рівнями прибутку і збитку. Для цього використовуються значення Take Profit та Stop Loss, які задаються у відсотках від ціни входу. TP встановлюється на рівні п’яти відсотків, а SL — на рівні двох. Такий ризик-менеджмент дозволяє зберігати позитивне математичне очікування стратегії, навіть якщо точність входів не перевищує п’ятдесяти відсотків.

Реалізація стратегії виконується мовою Python. Для обробки даних використовується бібліотека pandas, для обчислення технічних індикаторів — бібліотека ta. Дані подаються у вигляді CSV-файлу з історичними котируваннями (відкриття, максимум, мінімум, закриття, обсяг). У коді відбувається послідовне обчислення значень EMA, RSI, MACD та сигнальної лінії, після чого логіка перевіряє виконання умов входу. Якщо умови входу задовольняються, фіксується ціна входу, після чого контролюється, чи досягнута межа TP або SL. У кожному випадку результат записується до окремого журналу, де зберігається інформація про тип угоди, дату, ціну входу, ціну виходу, прибуток або збиток.

Після симуляції всіх угод на історичних даних проводиться оцінка ефективності стратегії, зокрема визначається загальний прибуток, кількість виграшних і програшних угод, середній прибуток на угоду, співвідношення ризику до винагороди. Це дозволяє зробити висновки щодо надійності і прибутковості стратегії на конкретному активі та таймфреймі.

У перспективі проєкт може бути розширений за рахунок додавання нових фільтрів, наприклад, обсягу або індикатора волатильності, таких як ATR. Крім того, замість фіксованих рівнів TP і SL можна використовувати динамічні, що адаптуються до поточної ринкової ситуації. Це відкриває можливості для автоматичної оптимізації параметрів або навіть застосування методів машинного навчання для адаптації стратегії до нових умов.

У підсумку, ця стратегія поєднує простоту реалізації з ефективністю, надаючи трейдеру гнучкий інструмент для ухвалення обґрунтованих торгових рішень на основі технічного аналізу.

Лістинг програми:

//@version=5

strategy("EMA9/EMA21 + RSI Strategy", overlay=true, default\_qty\_type=strategy.percent\_of\_equity, default\_qty\_value=100)

// === Налаштування ===

ema\_fast\_len = 9

ema\_slow\_len = 21

rsi\_len = 14

tp\_pct = 2.0 // Тейк-профіт у %

sl\_pct = 1.0 // Стоп-лосс у %

price = close

ema\_fast = ta.ema(price, ema\_fast\_len)

ema\_slow = ta.ema(price, ema\_slow\_len)

rsi\_val = ta.rsi(price, rsi\_len)

// === Умови входу ===

longCondition = ta.crossover(ema\_fast, ema\_slow) and rsi\_val > 50

shortCondition = ta.crossunder(ema\_fast, ema\_slow) and rsi\_val < 50

// === Розрахунок TP/SL у пунктах ===

long\_tp = price \* (1 + tp\_pct / 100)

long\_sl = price \* (1 - sl\_pct / 100)

short\_tp = price \* (1 - tp\_pct / 100)

short\_sl = price \* (1 + sl\_pct / 100)

// === Входи ===

if (longCondition)

strategy.entry("Long", strategy.long)

strategy.exit("Long TP/SL", from\_entry="Long", limit=long\_tp, stop=long\_sl)

if (shortCondition)

strategy.entry("Short", strategy.short)

strategy.exit("Short TP/SL", from\_entry="Short", limit=short\_tp, stop=short\_sl)

// === Графік індикаторів ===

plot(ema\_fast, color=color.orange, title="EMA 9")

plot(ema\_slow, color=color.blue, title="EMA 21")

hline(50, "RSI 50", color=color.gray)

plot(rsi\_val, title="RSI", color=color.purple, linewidth=1, display=display.none) // сховано з графіка

Результат виконання:



**Завдання 2:**

Лістинг програми:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

# Ініціалізація випадкових цін

np.random.seed(42)

timeline = pd.date\_range("2022-01-01", periods=200)

base\_price = np.cumsum(np.random.randn(200)) + 100

peak = base\_price + np.random.rand(200) \* 2

bottom = base\_price - np.random.rand(200) \* 2

market = pd.DataFrame({

    'timestamp': timeline,

    'open': base\_price,

    'high': peak,

    'low': bottom,

    'close': base\_price

})

# Технічні індикатори

def calc\_ema(values, period):

    return values.ewm(span=period, adjust=False).mean()

def calc\_rsi(series, period=14):

    delta = series.diff()

    up = delta.clip(lower=0).rolling(period).mean()

    down = -delta.clip(upper=0).rolling(period).mean()

    rs = up / down

    return 100 - (100 / (1 + rs))

market['EMAfast'] = calc\_ema(market['close'], 9)

market['EMAslow'] = calc\_ema(market['close'], 21)

market['RSI14'] = calc\_rsi(market['close'])

# Клас стратегії

class StrategySimulator:

    def \_\_init\_\_(self, dataset, capital=1000, take=0.02, stop=0.01):

        self.chart = dataset.copy()

        self.equity = capital

        self.take\_profit = take

        self.stop\_loss = stop

        self.orders = []

        self.flow = []

    def simulate(self):

        i = 21

        while i < len(self.chart):

            curr = self.chart.iloc[i]

            prev = self.chart.iloc[i - 1]

            order\_type = None

            if prev['EMAfast'] < prev['EMAslow'] and curr['EMAfast'] > curr['EMAslow'] and curr['RSI14'] > 50:

                order\_type = 'long'

            elif prev['EMAfast'] > prev['EMAslow'] and curr['EMAfast'] < curr['EMAslow'] and curr['RSI14'] < 50:

                order\_type = 'short'

            if order\_type:

                entry = curr['close']

                goal = entry \* (1 + self.take\_profit) if order\_type == 'long' else entry \* (1 - self.take\_profit)

                barrier = entry \* (1 - self.stop\_loss) if order\_type == 'long' else entry \* (1 + self.stop\_loss)

                for fwd in range(i + 1, len(self.chart)):

                    next\_candle = self.chart.iloc[fwd]

                    if order\_type == 'long':

                        if next\_candle['low'] <= barrier:

                            self.equity -= (entry - barrier)

                            self.orders.append(('loss', entry, barrier))

                            break

                        elif next\_candle['high'] >= goal:

                            self.equity += (goal - entry)

                            self.orders.append(('win', entry, goal))

                            break

                    else:

                        if next\_candle['high'] >= barrier:

                            self.equity -= (barrier - entry)

                            self.orders.append(('loss', entry, barrier))

                            break

                        elif next\_candle['low'] <= goal:

                            self.equity += (entry - goal)

                            self.orders.append(('win', entry, goal))

                            break

                self.flow.append(self.equity)

            i += 1

    def stats(self):

        successes = [deal for deal in self.orders if deal[0] == 'win']

        failures = [deal for deal in self.orders if deal[0] == 'loss']

        success\_rate = (len(successes) / len(self.orders) \* 100) if self.orders else 0

        gross\_gain = sum(exit\_price - entry\_price for \_, entry\_price, exit\_price in successes)

        gross\_loss = sum(entry\_price - stop\_price for \_, entry\_price, stop\_price in failures)

        net = gross\_gain - gross\_loss

        return {

            'Trades executed': len(self.orders),

            'Winning ratio (%)': round(success\_rate, 2),

            'Net P&L': round(net, 2),

            'Final capital': round(self.equity, 2)

        }

    def plot\_equity(self):

        plt.figure(figsize=(10, 4))

        plt.plot(self.flow, color='green', label='Capital')

        plt.title("Capital Evolution")

        plt.xlabel("Trade count")

        plt.ylabel("Equity")

        plt.grid(True)

        plt.legend()

        plt.tight\_layout()

        plt.show()

# Запуск

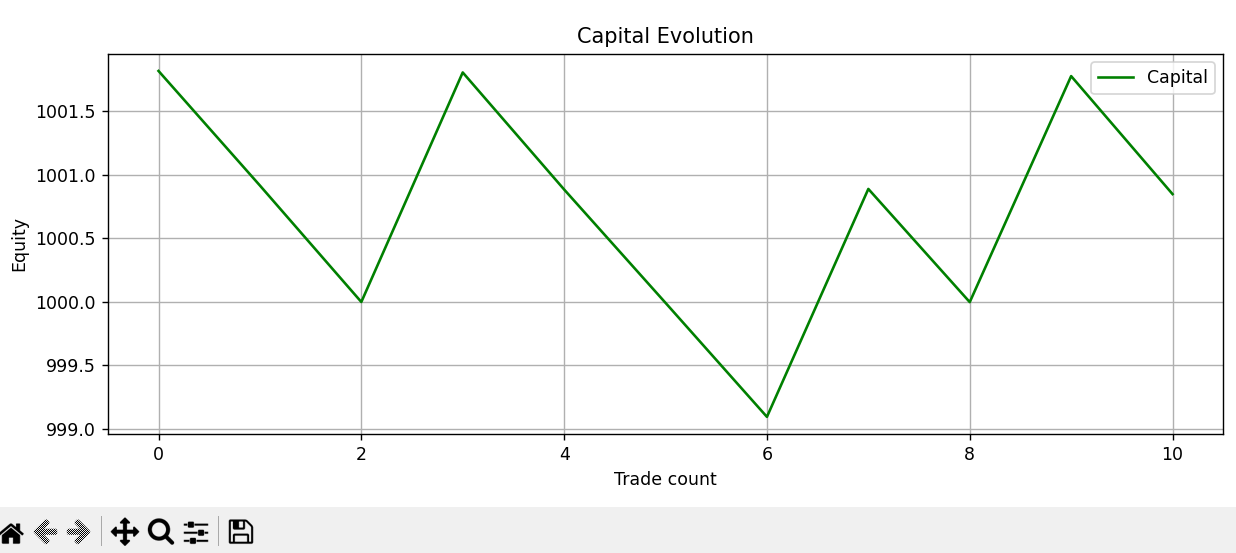
bot = StrategySimulator(market)

bot.simulate()

print(bot.stats())

bot.plot\_equity()

**Результат виконання:**

****

**Завдання 3:**

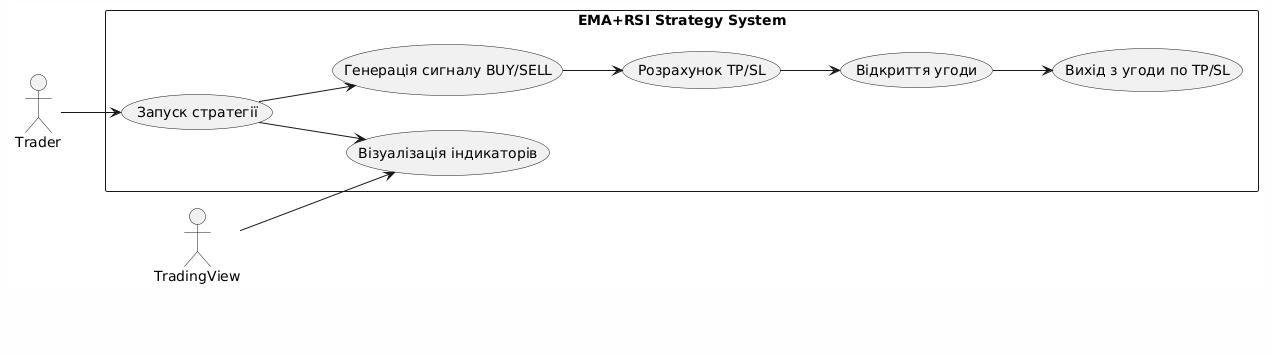


Рис. 1. Діаграма прецедентів

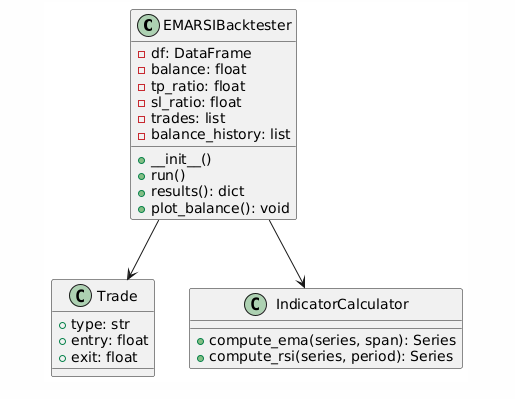


Рис. 2. Діаграма класів

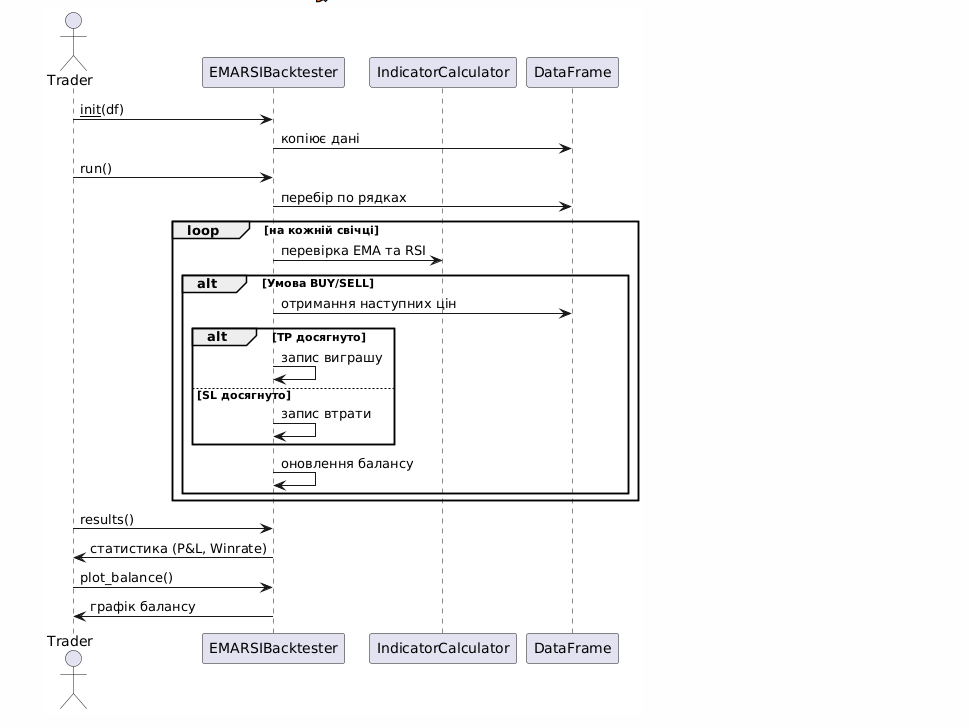


Рис. 3. Діаграма взаємодії