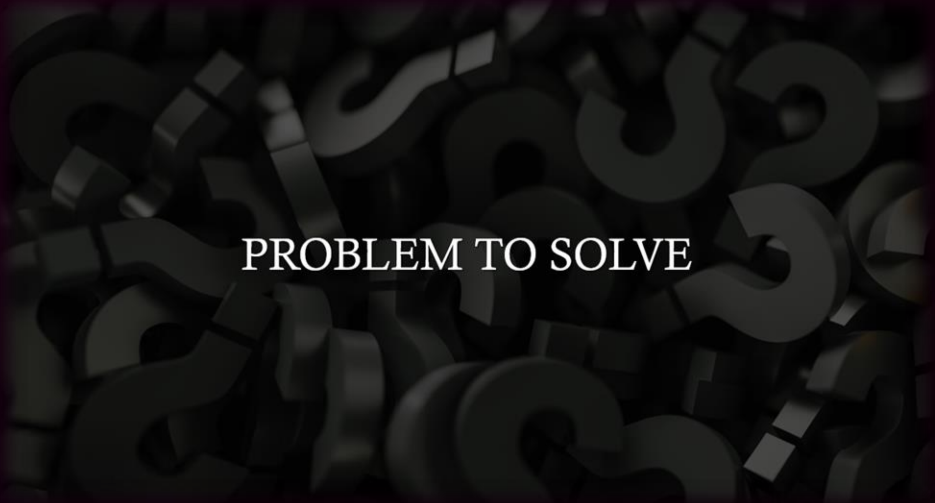
ALGO-TRADING

Emanuele Regnani



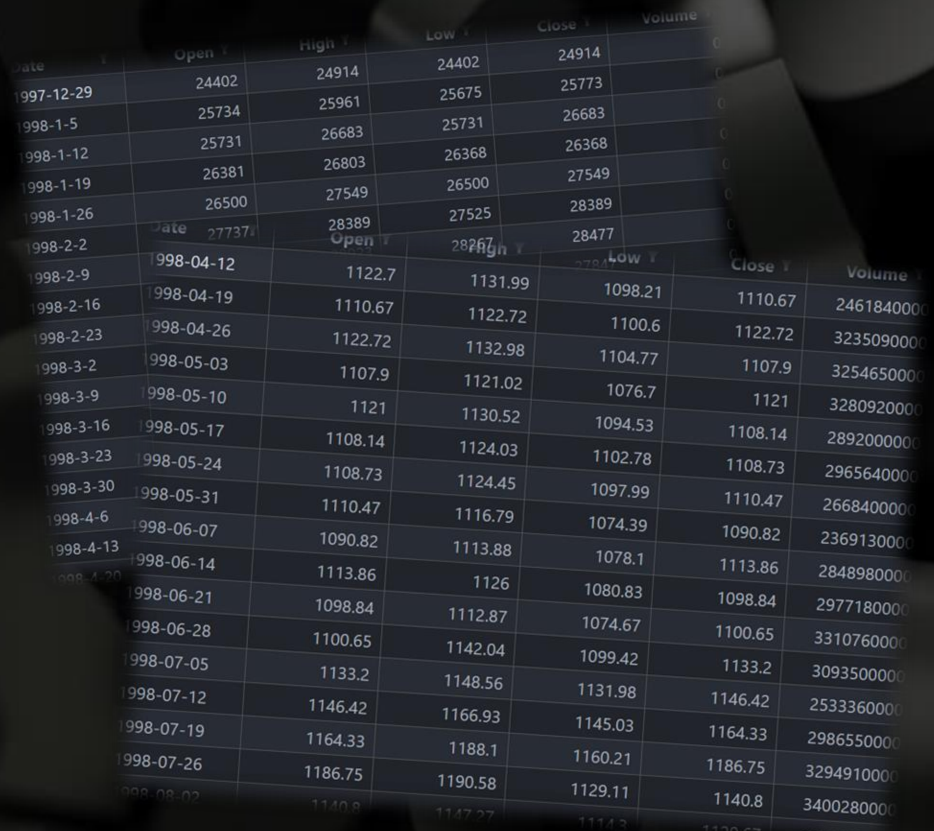
PROBLEM TO SOLVE

METHODOLOGY

SOFTWARE PROGRAM

RESULTS & LIMITATIONS

PROBLEM TO SOLVE



Date	Open	High	Low	Close	Volume
1997-12-29	24402	24914	24402	24914	
1998-1-5	25734	25961	25675	25773	
1998-1-12	25731	26683	25731	26683	
1998-1-19	26381	26803	26368	26368	
1998-1-26	26500	27549	26500	27549	
1998-2-2	27737	28389	27525	28389	
1998-2-9			28267	28477	
1998-2-16					
1998-2-23					
1998-3-2					
1998-3-9					
1998-3-16					
1998-3-23					
1998-3-30					
1998-4-6					
1998-4-13					
1998-4-20					

Date	Open	High	Low	Close	Volume
1998-04-12	1122.7	1131.99	1098.21	1110.67	246184000
1998-04-19	1110.67	1122.72	1100.6	1122.72	323509000
1998-04-26	1122.72	1132.98	1104.77	1107.9	325465000
1998-05-03	1107.9	1121.02	1076.7	1121	328092000
1998-05-10	1121	1130.52	1094.53	1108.14	289200000
1998-05-17	1108.14	1124.03	1102.78	1108.73	296564000
1998-05-24	1108.73	1124.45	1097.99	1110.47	266840000
1998-05-31	1110.47	1116.79	1074.39	1090.82	236913000
1998-06-07	1090.82	1113.88	1078.1	1113.86	284898000
1998-06-14	1113.86	1126	1080.83	1098.84	297718000
1998-06-21	1098.84	1112.87	1074.67	1100.65	331076000
1998-06-28	1100.65	1142.04	1099.42	1133.2	309350000
1998-07-05	1133.2	1148.56	1131.98	1146.42	253336000
1998-07-12	1146.42	1166.93	1145.03	1164.33	298655000
1998-07-19	1164.33	1188.1	1160.21	1186.75	329491000
1998-07-26	1186.75	1190.58	1129.11	1140.8	340028000
1998-08-02	1140.8	1147.27	1114.3	1120.67	330125000

Buy/Hold/Sell Recommendations

METHODOLOGY

TRADING INDICATORS

FUZZY LOGIC

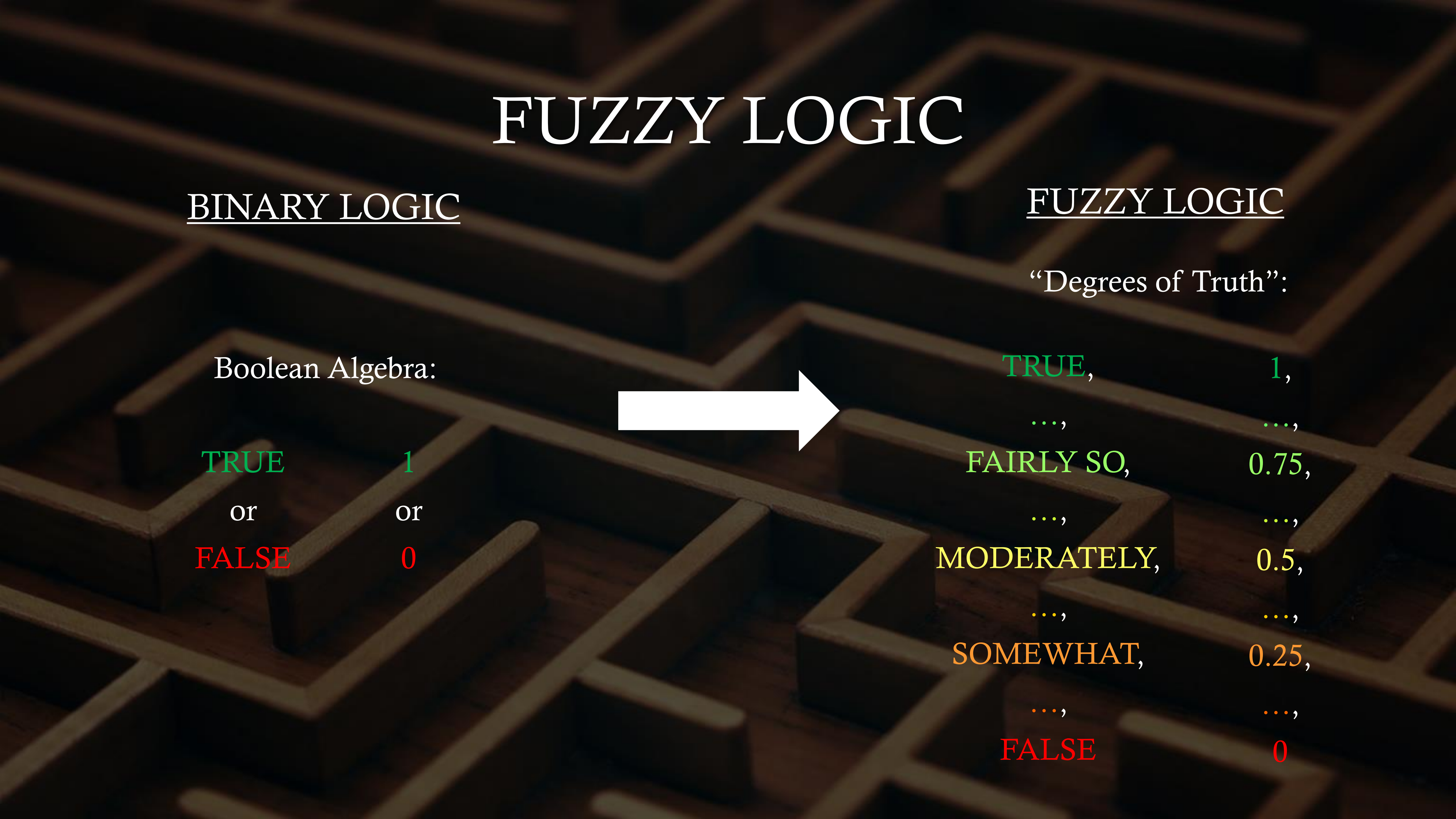
GENETIC ALGORITHM

TRADING INDICATORS

Visual / Analytical tools that help analyze financial markets and make trading decisions by helping to spot trends, patterns, momentum, volatility, …

Example:

The average closing price of the last n days



FUZZY LOGIC

BINARY LOGIC

Boolean Algebra:

TRUE	1
or	or
FALSE	0

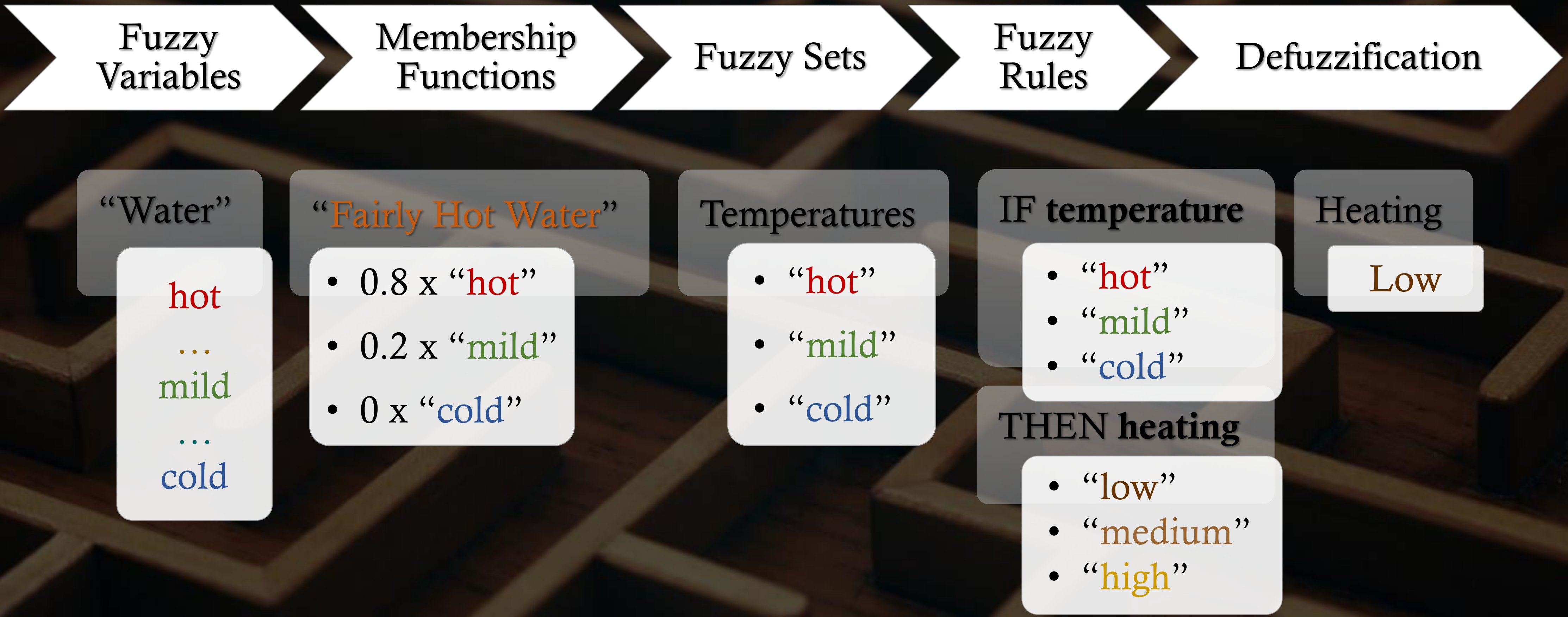
→

FUZZY LOGIC

"Degrees of Truth":

TRUE,	1,
…,	…,
FAIRLY SO,	0.75,
…,	…,
MODERATELY,	0.5,
…,	…,
SOMEWHAT,	0.25,
…,	…,
FALSE	0

FUZZY LOGIC



Fuzzy Variables

“Water”

hot
…
mild
…
cold

Membership Functions

“Fairly Hot Water”

- 0.8 x “hot”
- 0.2 x “mild”
- 0 x “cold”

Fuzzy Sets

Temperatures

- “hot”
- “mild”
- “cold”

Fuzzy Rules

IF **temperature**

- “hot”
- “mild”
- “cold”

THEN **heating**

- “low”
- “medium”
- “high”

Defuzzification

Heating

Low

GENETIC ALGORITHMS

Optimization algorithms:

- Inspired by biological evolution

- Heuristic (vs “Exact”):
    - faster and more efficient solution needed, despite not necessarily the best one (e.g.: great number of parameters)

GENETIC ALGORITHMS

FITNESS FUNCTION

A function to be minimized or maximized, determining each individual’s ability to compete in its environment.

It is evaluated at each iteration of the optimization and used to select the best individuals to survive to the next generation.

SOFTWARE PROGRAM

TRADING INDICATORS

FUZZY LOGIC

GENETIC ALGORITHM

TRADING INDICATORS

- RSI (Relative Strength Index):

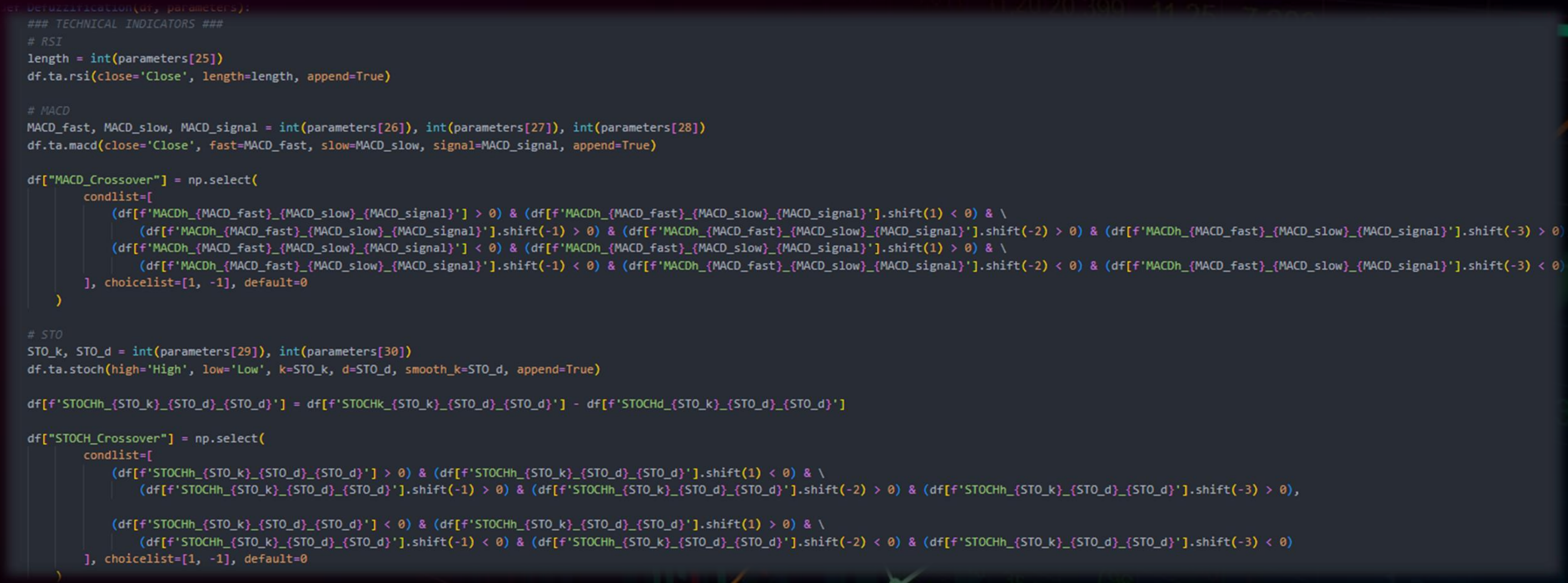
  momentum indicator measuring the strength of a security’s price action
  “oversold” – “risky trading” – “overbought”

- MACD:

  momentum indicator also showing trend changes
  by making use of different weighted averages of the prices and combining them

- Stochastic Oscillator:

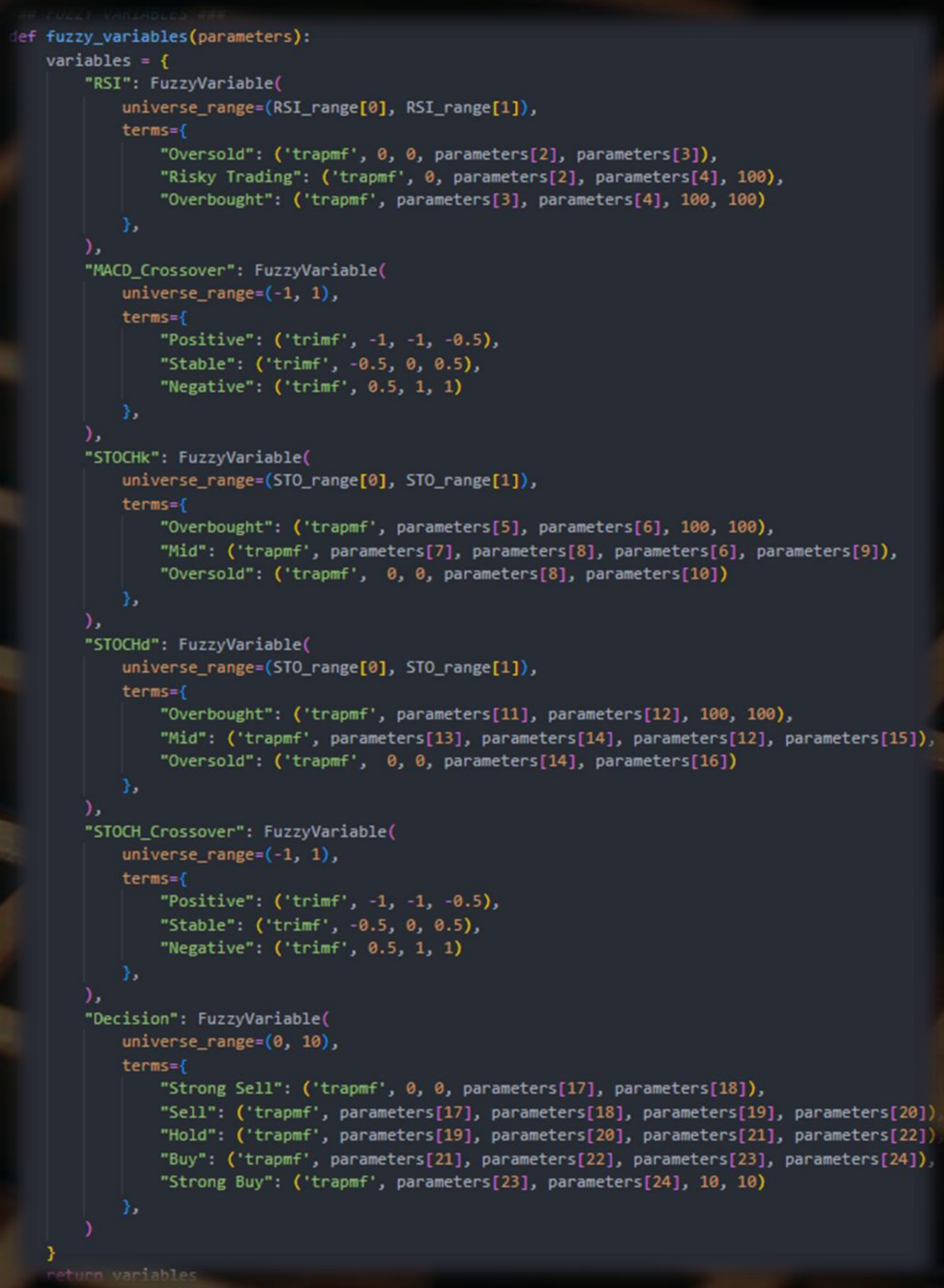
  similar to RSI



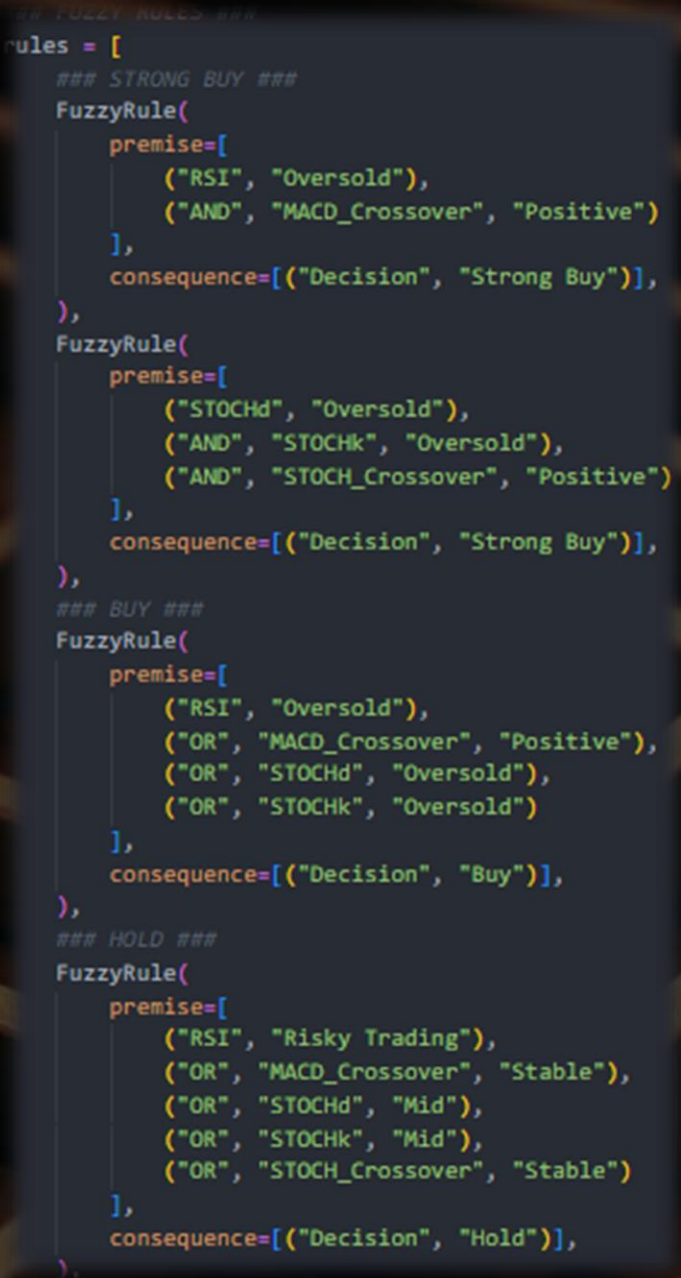
```python
def Defuzzification(df, parameters):
    ### TECHNICAL INDICATORS ###
    # RSI
    length = int(parameters[25])
    df.ta.rsi(close='Close', length=length, append=True)

    # MACD
    MACD_fast, MACD_slow, MACD_signal = int(parameters[26]), int(parameters[27]), int(parameters[28])
    df.ta.macd(close='Close', fast=MACD_fast, slow=MACD_slow, signal=MACD_signal, append=True)

    df["MACD_Crossover"] = np.select(
            condlist=[
                (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'] > 0) & (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'].shift(1) < 0) & \
                    (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'].shift(-1) > 0) & (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'].shift(-2) > 0) & (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'].shift(-3) > 0)
                (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'] < 0) & (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'].shift(1) > 0) & \
                    (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'].shift(-1) < 0) & (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'].shift(-2) < 0) & (df[f'MACDh_{MACD_fast}_{MACD_slow}_{MACD_signal}'].shift(-3) < 0)
            ], choicelist=[1, -1], default=0
        )

    # STO
    STO_k, STO_d = int(parameters[29]), int(parameters[30])
    df.ta.stoch(high='High', low='Low', k=STO_k, d=STO_d, smooth_k=STO_d, append=True)

    df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'] = df[f'STOCHk_{STO_k}_{STO_d}_{STO_d}'] - df[f'STOCHd_{STO_k}_{STO_d}_{STO_d}']

    df["STOCH_Crossover"] = np.select(
            condlist=[
                (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'] > 0) & (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'].shift(1) < 0) & \
                    (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'].shift(-1) > 0) & (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'].shift(-2) > 0) & (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'].shift(-3) > 0),

                (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'] < 0) & (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'].shift(1) > 0) & \
                    (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'].shift(-1) < 0) & (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'].shift(-2) < 0) & (df[f'STOCHh_{STO_k}_{STO_d}_{STO_d}'].shift(-3) < 0)
            ], choicelist=[1, -1], default=0
        )
```

FUZZY VARIABLES

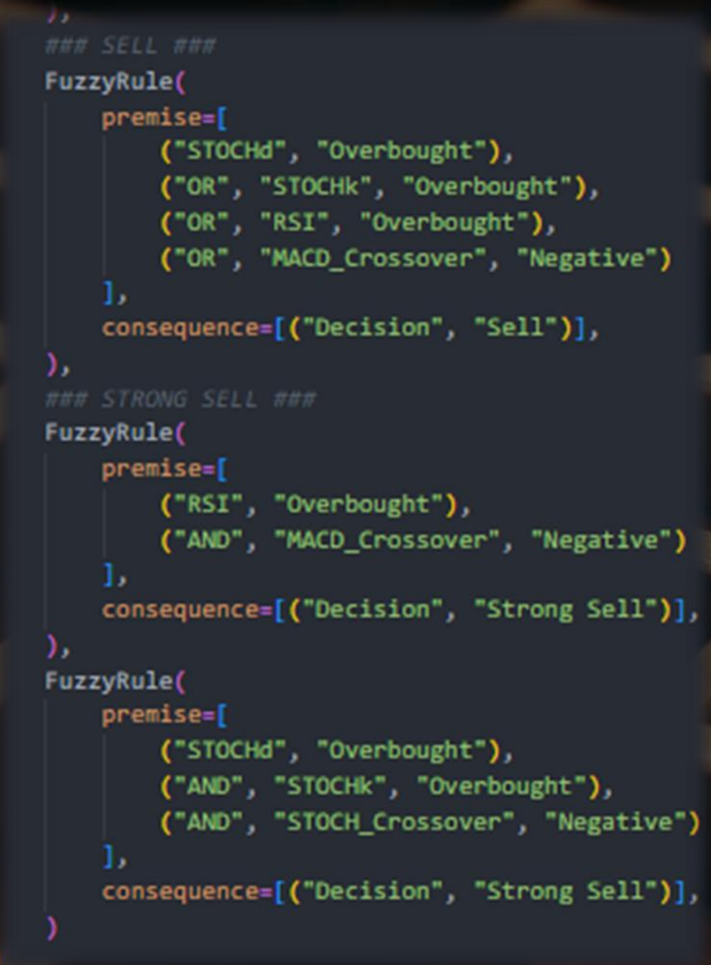
- For each indicator
- For particular combinations of them
- For the trading decision to be made



```python
### FUZZY VARIABLES ###
def fuzzy_variables(parameters):
    variables = {
        "RSI": FuzzyVariable(
            universe_range=(RSI_range[0], RSI_range[1]),
            terms={
                "Oversold": ('trapmf', 0, 0, parameters[2], parameters[3]),
                "Risky Trading": ('trapmf', 0, parameters[2], parameters[4], 100),
                "Overbought": ('trapmf', parameters[3], parameters[4], 100, 100)
            },
        ),
        "MACD_Crossover": FuzzyVariable(
            universe_range=(-1, 1),
            terms={
                "Positive": ('trimf', -1, -1, -0.5),
                "Stable": ('trimf', -0.5, 0, 0.5),
                "Negative": ('trimf', 0.5, 1, 1)
            },
        ),
        "STOCHk": FuzzyVariable(
            universe_range=(STO_range[0], STO_range[1]),
            terms={
                "Overbought": ('trapmf', parameters[5], parameters[6], 100, 100),
                "Mid": ('trapmf', parameters[7], parameters[8], parameters[6], parameters[9]),
                "Oversold": ('trapmf',  0, 0, parameters[8], parameters[10])
            },
        ),
        "STOCHd": FuzzyVariable(
            universe_range=(STO_range[0], STO_range[1]),
            terms={
                "Overbought": ('trapmf', parameters[11], parameters[12], 100, 100),
                "Mid": ('trapmf', parameters[13], parameters[14], parameters[12], parameters[15]),
                "Oversold": ('trapmf',  0, 0, parameters[14], parameters[16])
            },
        ),
        "STOCH_Crossover": FuzzyVariable(
            universe_range=(-1, 1),
            terms={
                "Positive": ('trimf', -1, -1, -0.5),
                "Stable": ('trimf', -0.5, 0, 0.5),
                "Negative": ('trimf', 0.5, 1, 1)
            },
        ),
        "Decision": FuzzyVariable(
            universe_range=(0, 10),
            terms={
                "Strong Sell": ('trapmf', 0, 0, parameters[17], parameters[18]),
                "Sell": ('trapmf', parameters[17], parameters[18], parameters[19], parameters[20]),
                "Hold": ('trapmf', parameters[19], parameters[20], parameters[21], parameters[22]),
                "Buy": ('trapmf', parameters[21], parameters[22], parameters[23], parameters[24]),
                "Strong Buy": ('trapmf', parameters[23], parameters[24], 10, 10)
            },
        )
    }
    return variables
```



```
## FUZZY RULES ###
rules = [
    ### STRONG BUY ###
    FuzzyRule(
        premise=[
            ("RSI", "Oversold"),
            ("AND", "MACD_Crossover", "Positive")
        ],
        consequence=[("Decision", "Strong Buy")],
    ),
    FuzzyRule(
        premise=[
            ("STOCHd", "Oversold"),
            ("AND", "STOCHk", "Oversold"),
            ("AND", "STOCH_Crossover", "Positive")
        ],
        consequence=[("Decision", "Strong Buy")],
    ),
    ### BUY ###
    FuzzyRule(
        premise=[
            ("RSI", "Oversold"),
            ("OR", "MACD_Crossover", "Positive"),
            ("OR", "STOCHd", "Oversold"),
            ("OR", "STOCHk", "Oversold")
        ],
        consequence=[("Decision", "Buy")],
    ),
    ### HOLD ###
    FuzzyRule(
        premise=[
            ("RSI", "Risky Trading"),
            ("OR", "MACD_Crossover", "Stable"),
            ("OR", "STOCHd", "Mid"),
            ("OR", "STOCHk", "Mid"),
            ("OR", "STOCH_Crossover", "Stable")
        ],
        consequence=[("Decision", "Hold")],
    ),
```

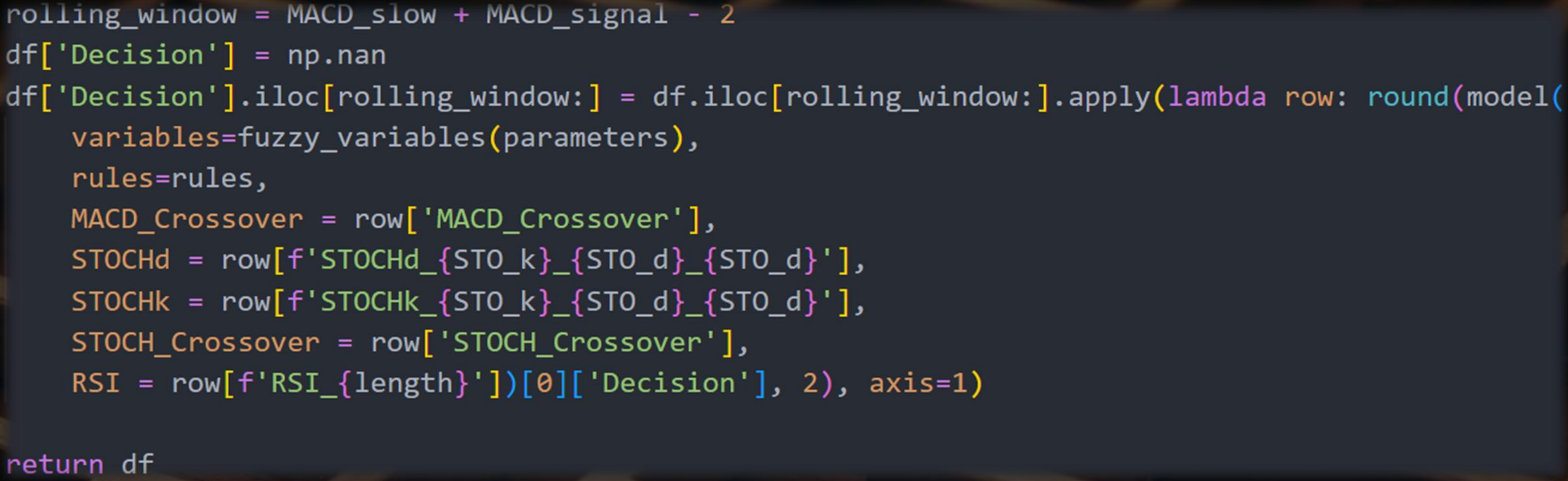


```
    ),
    ### SELL ###
    FuzzyRule(
        premise=[
            ("STOCHd", "Overbought"),
            ("OR", "STOCHk", "Overbought"),
            ("OR", "RSI", "Overbought"),
            ("OR", "MACD_Crossover", "Negative")
        ],
        consequence=[("Decision", "Sell")],
    ),
    ### STRONG SELL ###
    FuzzyRule(
        premise=[
            ("RSI", "Overbought"),
            ("AND", "MACD_Crossover", "Negative")
        ],
        consequence=[("Decision", "Strong Sell")],
    ),
    FuzzyRule(
        premise=[
            ("STOCHd", "Overbought"),
            ("AND", "STOCHk", "Overbought"),
            ("AND", "STOCH_Crossover", "Negative")
        ],
        consequence=[("Decision", "Strong Sell")],
    )
```

FUZZY RULES

- “Strong Buy”
- “Buy”
- “Hold”
- “Sell”
- “Strong Sell”

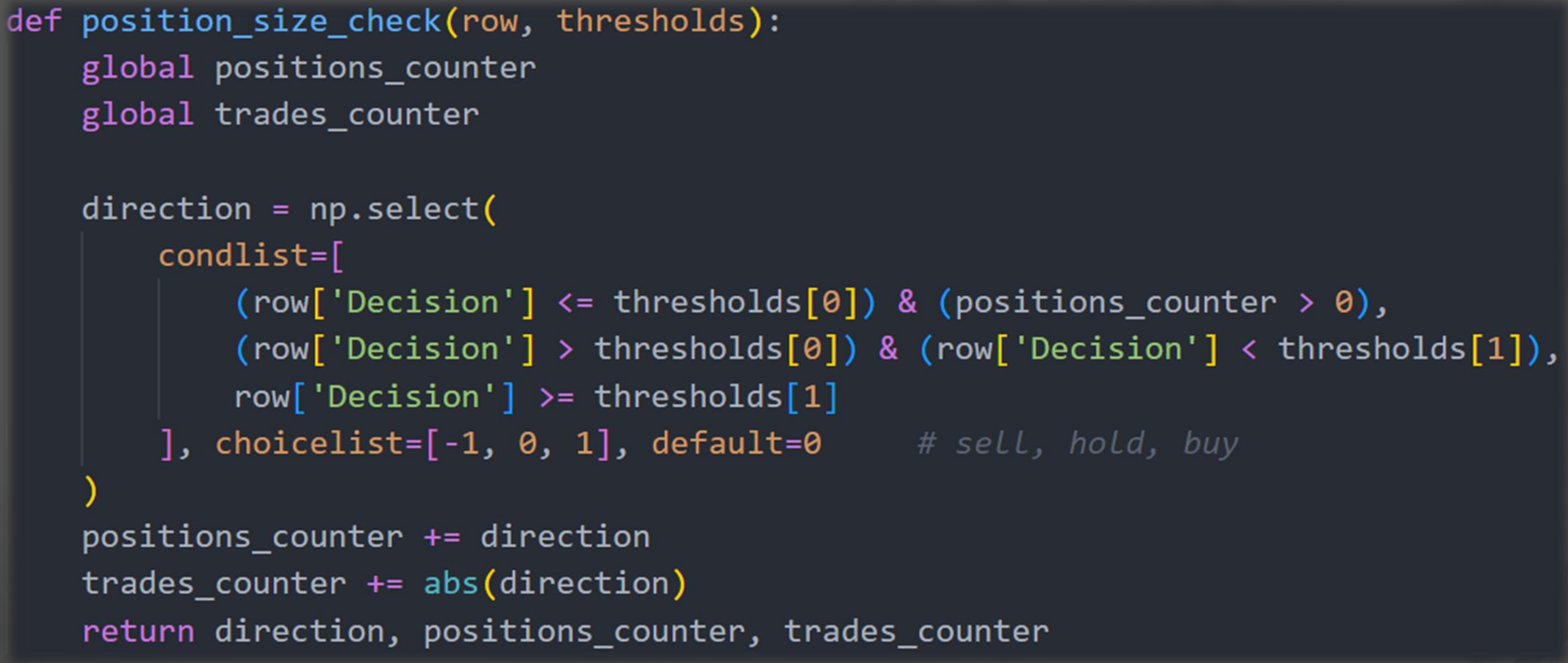
DEFUZZIFICATION



```
rolling_window = MACD_slow + MACD_signal - 2
df['Decision'] = np.nan
df['Decision'].iloc[rolling_window:] = df.iloc[rolling_window:].apply(lambda row: round(model(
    variables=fuzzy_variables(parameters),
    rules=rules,
    MACD_Crossover = row['MACD_Crossover'],
    STOCHd = row[f'STOCHd_{STO_k}_{STO_d}_{STO_d}'],
    STOCHk = row[f'STOCHk_{STO_k}_{STO_d}_{STO_d}'],
    STOCH_Crossover = row['STOCH_Crossover'],
    RSI = row[f'RSI_{length}'])[0]['Decision'], 2), axis=1)

return df
```

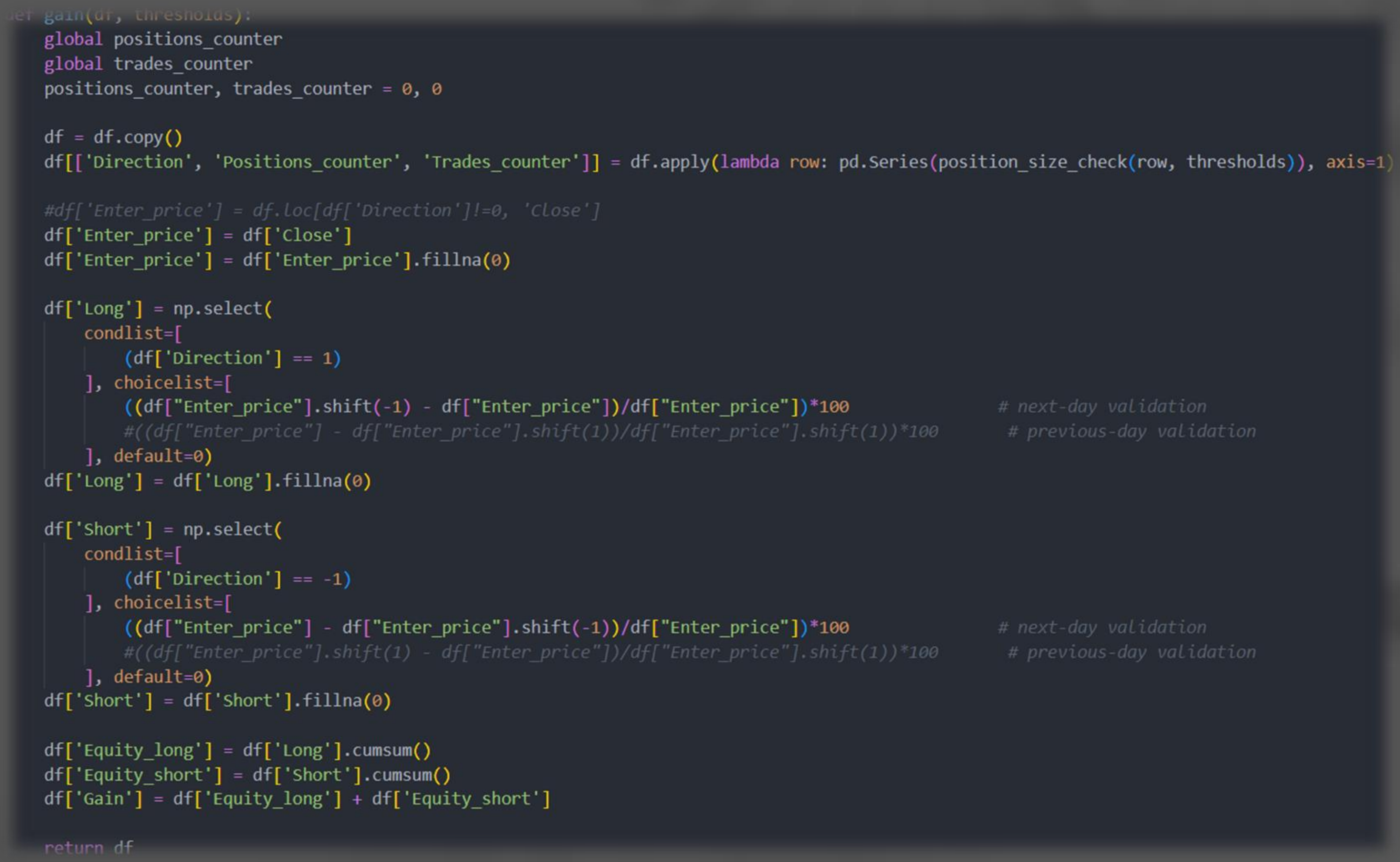
FITNESS FUNCTION:

- Positions Counter
- Trades Counter



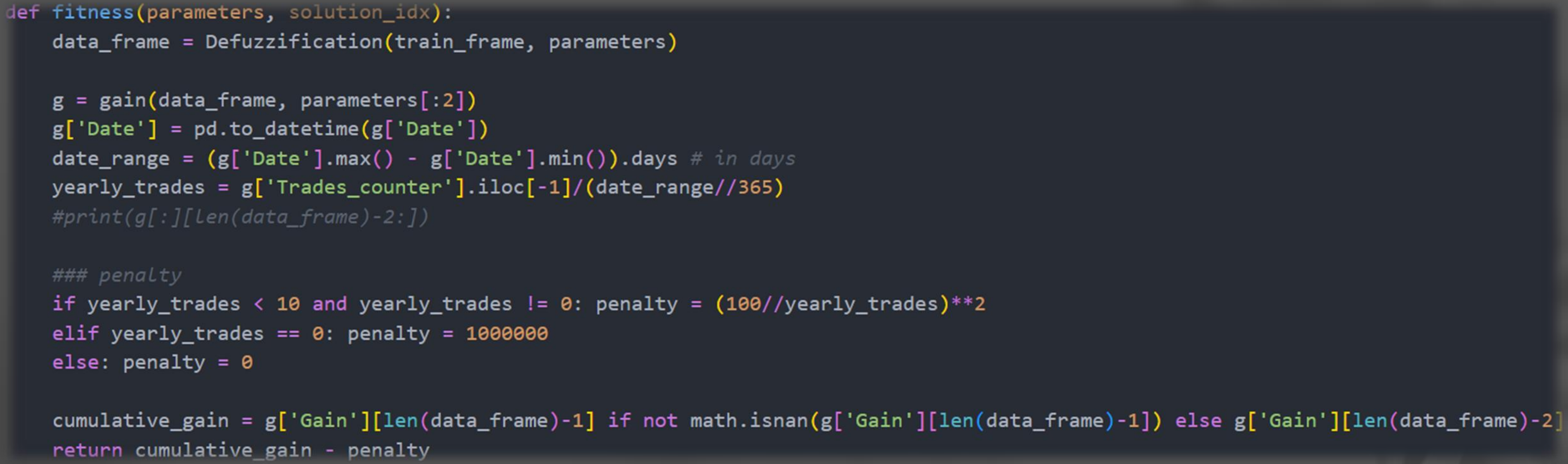
```python
def position_size_check(row, thresholds):
    global positions_counter
    global trades_counter

    direction = np.select(
        condlist=[
            (row['Decision'] <= thresholds[0]) & (positions_counter > 0),
            (row['Decision'] > thresholds[0]) & (row['Decision'] < thresholds[1]),
            row['Decision'] >= thresholds[1]
        ], choicelist=[-1, 0, 1], default=0     # sell, hold, buy
    )
    positions_counter += direction
    trades_counter += abs(direction)
    return direction, positions_counter, trades_counter
```

FITNESS FUNCTION: Gain Calculation



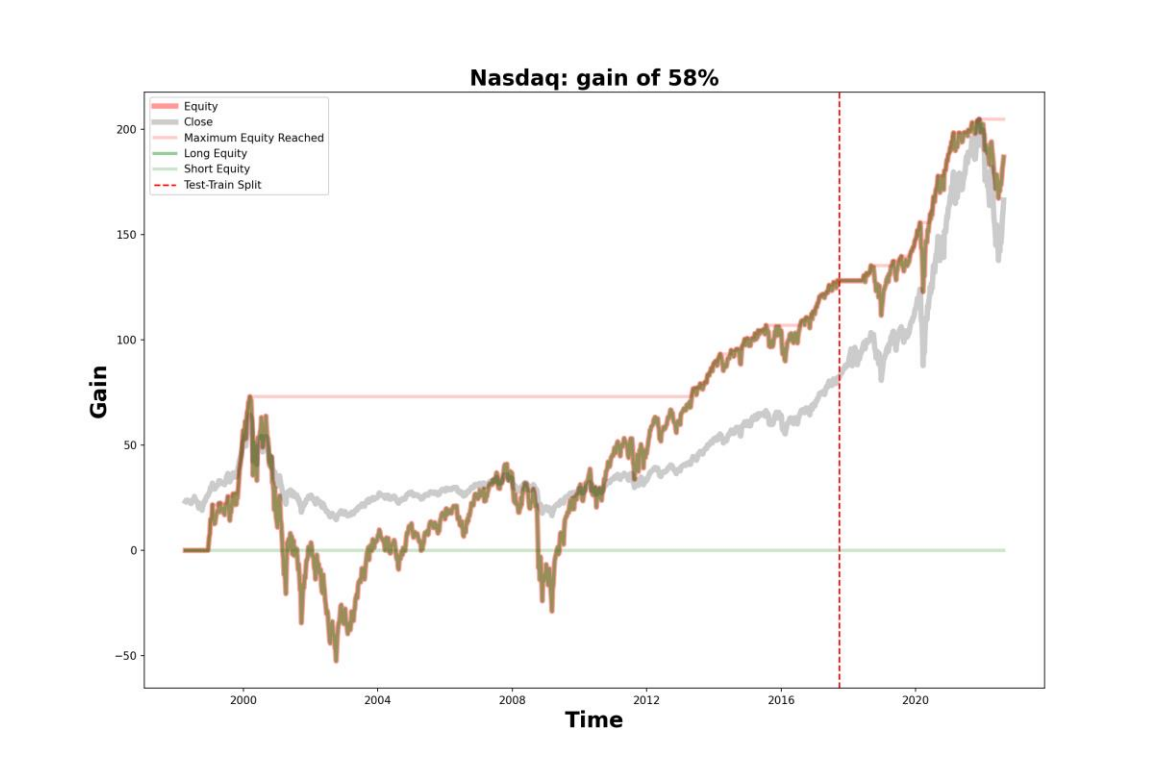
```python
def gain(df, thresholds):
    global positions_counter
    global trades_counter
    positions_counter, trades_counter = 0, 0

    df = df.copy()
    df[['Direction', 'Positions_counter', 'Trades_counter']] = df.apply(lambda row: pd.Series(position_size_check(row, thresholds)), axis=1)

    #df['Enter_price'] = df.loc[df['Direction']!=0, 'Close']
    df['Enter_price'] = df['Close']
    df['Enter_price'] = df['Enter_price'].fillna(0)

    df['Long'] = np.select(
        condlist=[
            (df['Direction'] == 1)
        ], choicelist=[
            ((df["Enter_price"].shift(-1) - df["Enter_price"])/df["Enter_price"])*100               # next-day validation
            #((df["Enter_price"] - df["Enter_price"].shift(1))/df["Enter_price"].shift(1))*100       # previous-day validation
        ], default=0)
    df['Long'] = df['Long'].fillna(0)

    df['Short'] = np.select(
        condlist=[
            (df['Direction'] == -1)
        ], choicelist=[
            ((df["Enter_price"] - df["Enter_price"].shift(-1))/df["Enter_price"])*100               # next-day validation
            #((df["Enter_price"].shift(1) - df["Enter_price"])/df["Enter_price"].shift(1))*100       # previous-day validation
        ], default=0)
    df['Short'] = df['Short'].fillna(0)

    df['Equity_long'] = df['Long'].cumsum()
    df['Equity_short'] = df['Short'].cumsum()
    df['Gain'] = df['Equity_long'] + df['Equity_short']

    return df
```

FITNESS FUNCTION

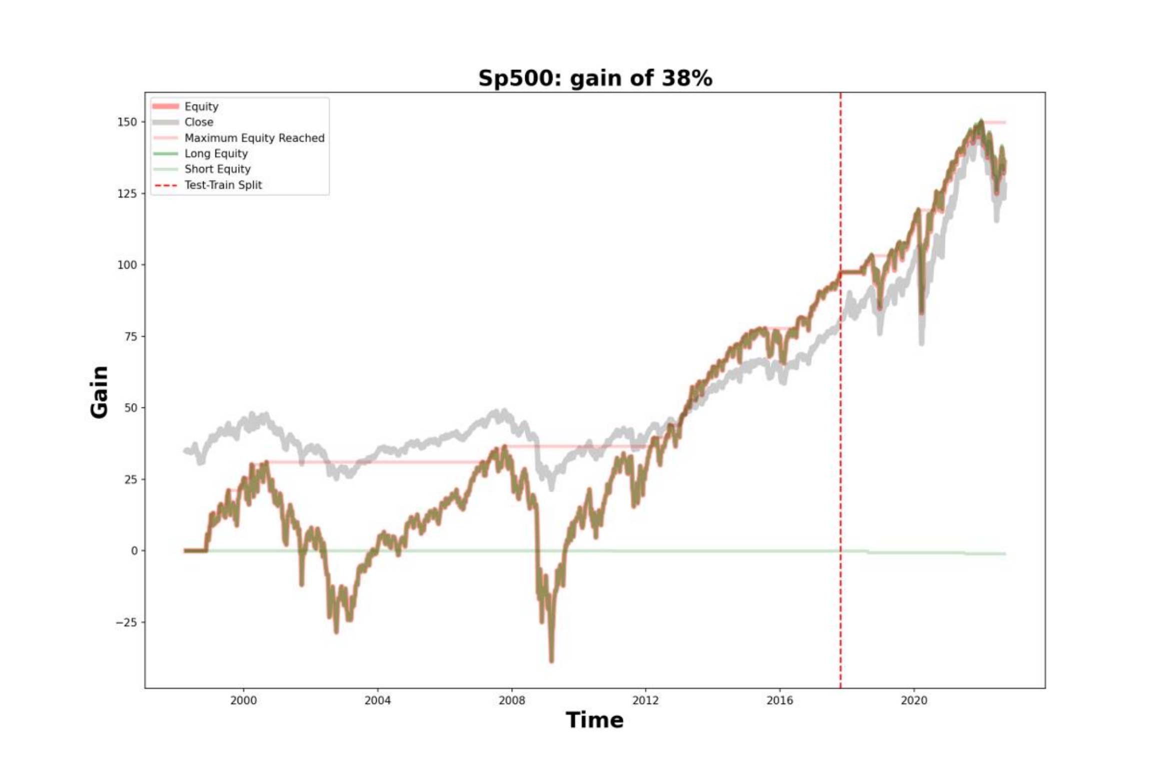


```
def fitness(parameters, solution_idx):
    data_frame = Defuzzification(train_frame, parameters)

    g = gain(data_frame, parameters[:2])
    g['Date'] = pd.to_datetime(g['Date'])
    date_range = (g['Date'].max() - g['Date'].min()).days # in days
    yearly_trades = g['Trades_counter'].iloc[-1]/(date_range//365)
    #print(g[:][len(data_frame)-2:])

    ### penalty
    if yearly_trades < 10 and yearly_trades != 0: penalty = (100//yearly_trades)**2
    elif yearly_trades == 0: penalty = 1000000
    else: penalty = 0

    cumulative_gain = g['Gain'][len(data_frame)-1] if not math.isnan(g['Gain'][len(data_frame)-1]) else g['Gain'][len(data_frame)-2]
    return cumulative_gain - penalty
```

GAIN OR LOSS?

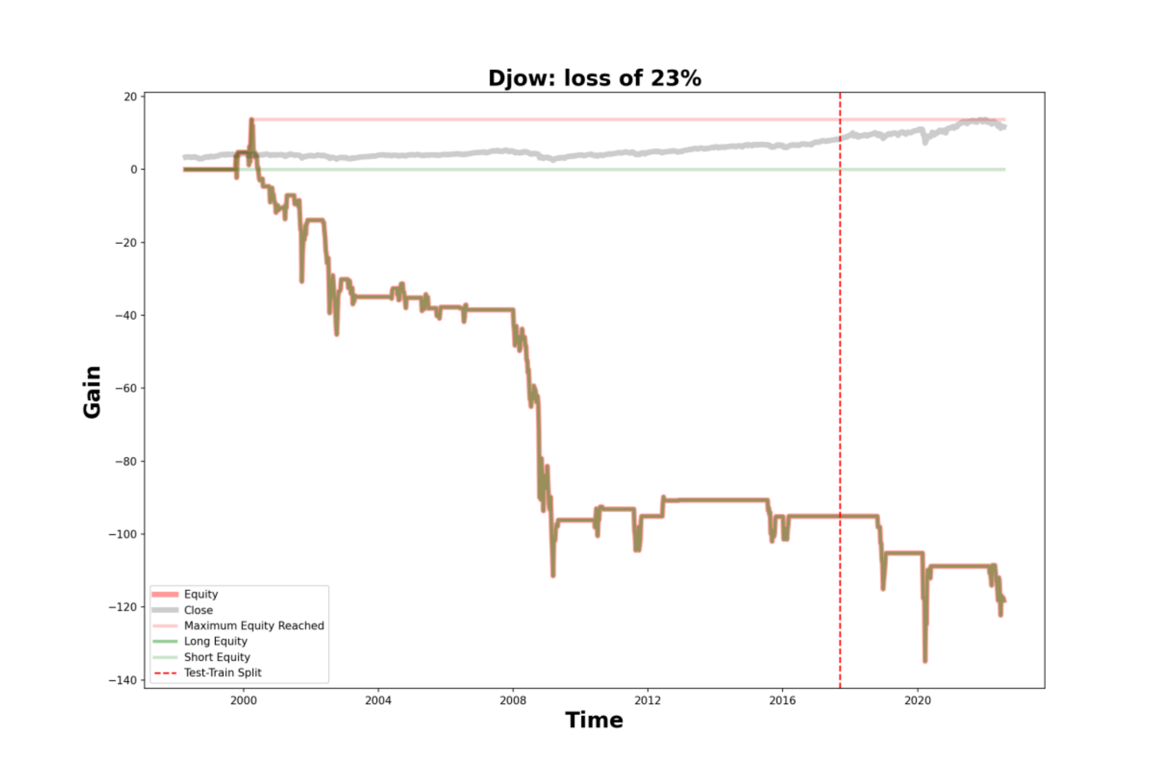


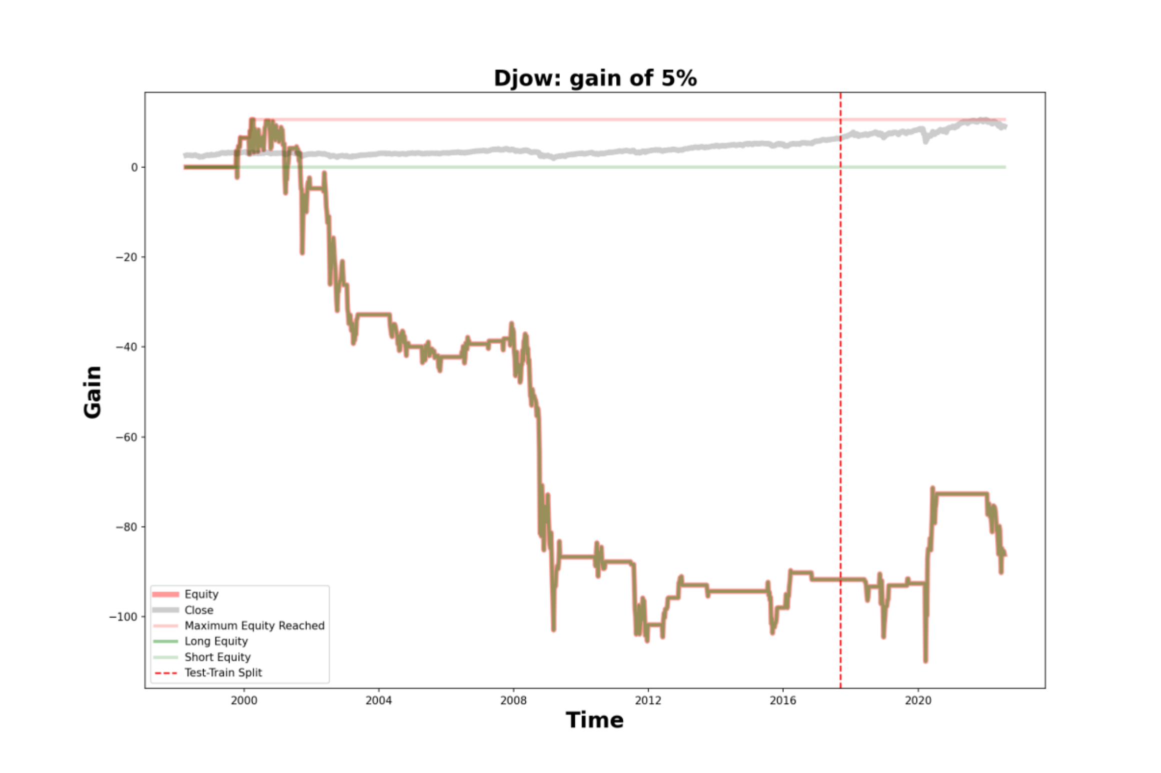
Nasdaq: gain of 58%
Equity
Close
Maximum Equity Reached
Long Equity
Short Equity
Test-Train Split
200
150
100
50
0
−50
Gain
2000
2004
2008
2012
2016
2020
Time



Sp500: gain of 38%
Equity
Close
Maximum Equity Reached
Long Equity
Short Equity
Test-Train Split
150
125
100
75
50
25
0
−25
Gain
2000
2004
2008
2012
2016
2020
Time

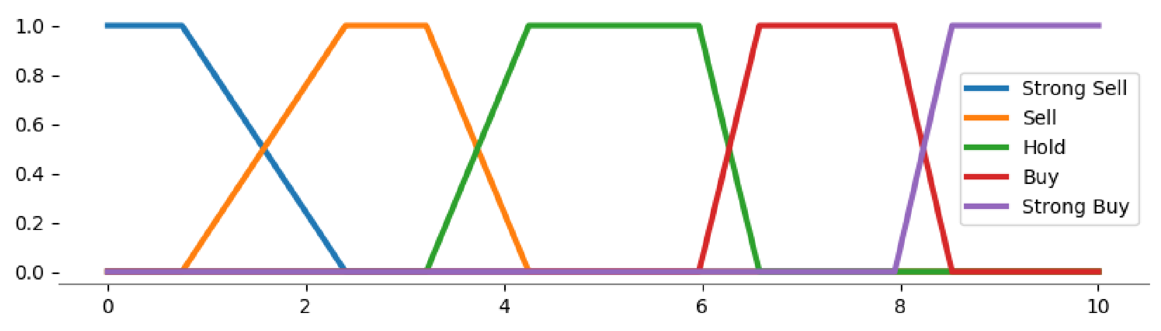
HEURISTIC NATURE!



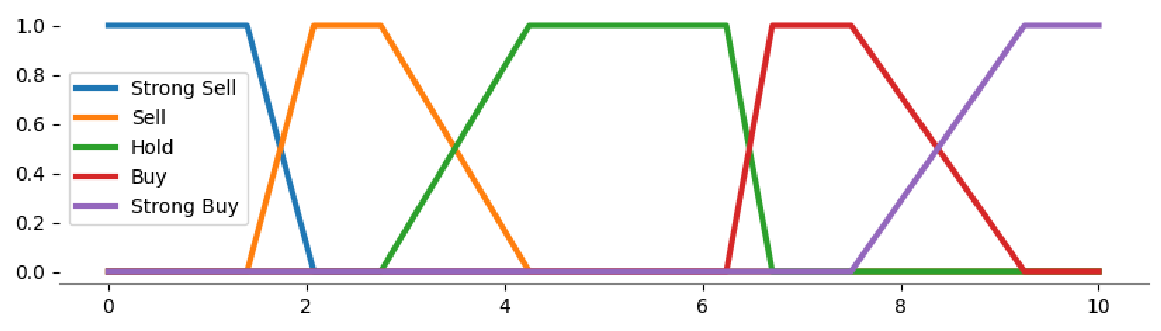


CONSIDERATIONS

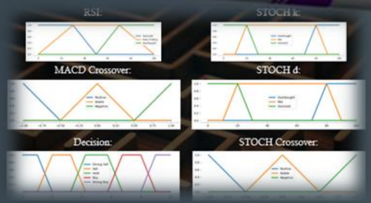
- Weekly Data Used
- Optimal parameters tend to approach values similar to the ones commonly suggested
- Interesting adaptations and peculiarities, however more analysis needed:



Losing



Winning



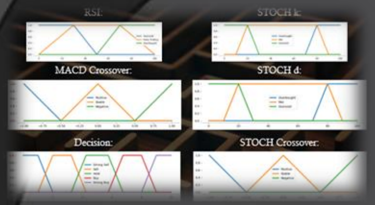
LIMITATIONS

CODING

- Speed & Efficiency: takes minutes to run
- Data & Testing: it has only been tested on weekly data

STRATEGY:

- Membership Functions: sometimes overly simplistic
- Technical Indicators: more and more varied (mostly momentum ones for now)



OTHER:

- More analysis of parameters found could reveal interesting insights

THANK YOU!