Fuzzy Moving Average Trading System With Genetic Algorithm Optimization

EN.525.770.81 - Intelligent Algorithms
Liam Ventura
Original Authors: Tan Chee Wei, Laxman Singh,
Ramasamy Muthuraman, and Isaac Varun Kumar

Motivation

- The Moving Average Strategy (MAS) is one of the most popular technical indicators that informs traders of existing trends and helps identify upcoming reversals
    - Generates buy or sell signals by seeing if the difference between the long and short-term MA is positive or negative [1][2]
- Fuzzy Logic integration allows to describe the strength of the signal to generate controller actions (i.e. buy or sell)
- Genetic Algorithm implementation will optimized the trading strategy parameters in order to maximize returns

Simple Moving Average

Simple Moving Average (SMA)

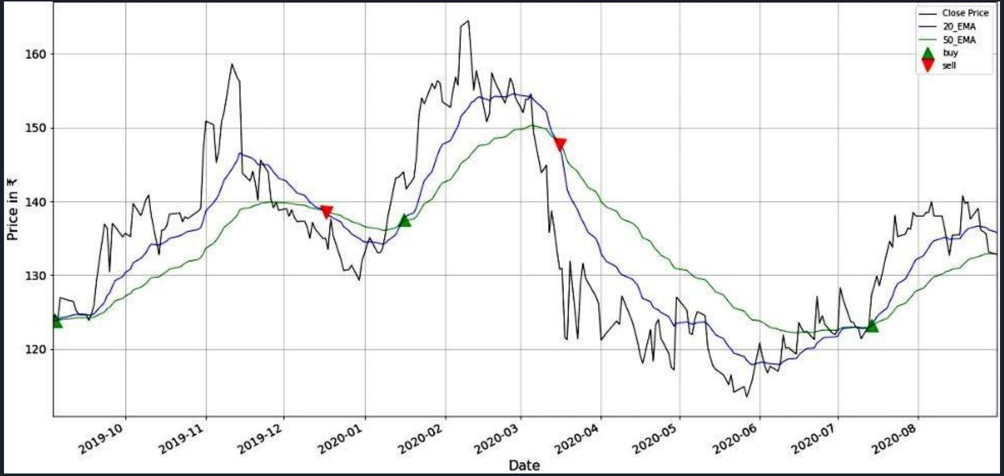
A = Average in period n

n = Number of time periods (window)

$$SMA = \frac{A_1 + A_2 + ... + A_n}{n}$$

Simple Moving Average Strategy

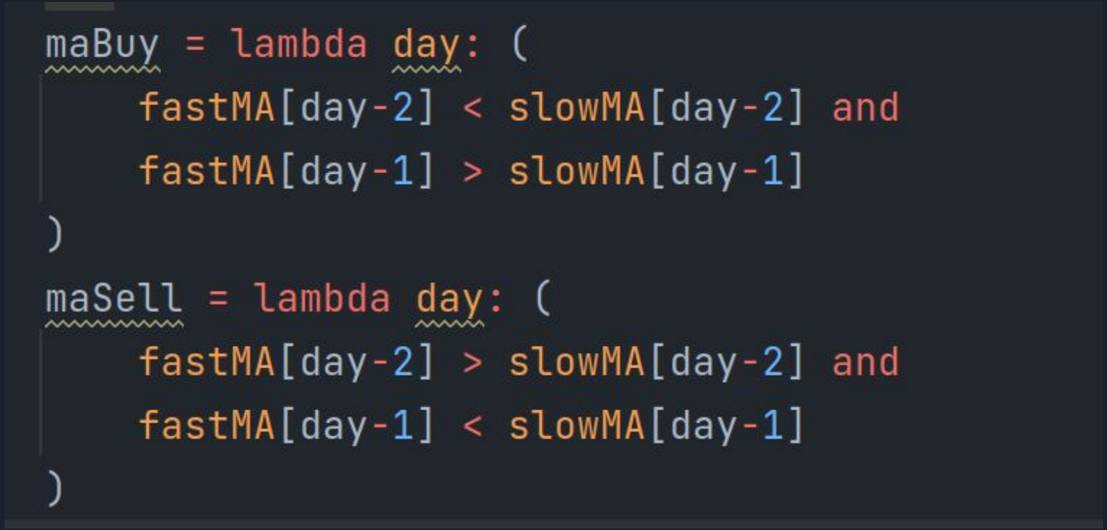
- When the Short-term (Fast) Moving Average crosses above the Long-term (Slow) Moving Average -> Indicates a ***Buy*** Signal
- When the Short-term (Fast) Moving Average crosses below the Long-term (Slow) Moving Average -> Indicates a ***Sell Signal***



Source: [Linkedin - Patrick Nabriya]

Simple Moving Average Strategy

- When the Short-term (Fast) Moving Average crosses above the Long-term (Slow) Moving Average -> Indicates a ***Buy*** Signal
- When the Short-term (Fast) Moving Average crosses below the Long-term (Slow) Moving Average -> Indicates a ***Sell Signal***



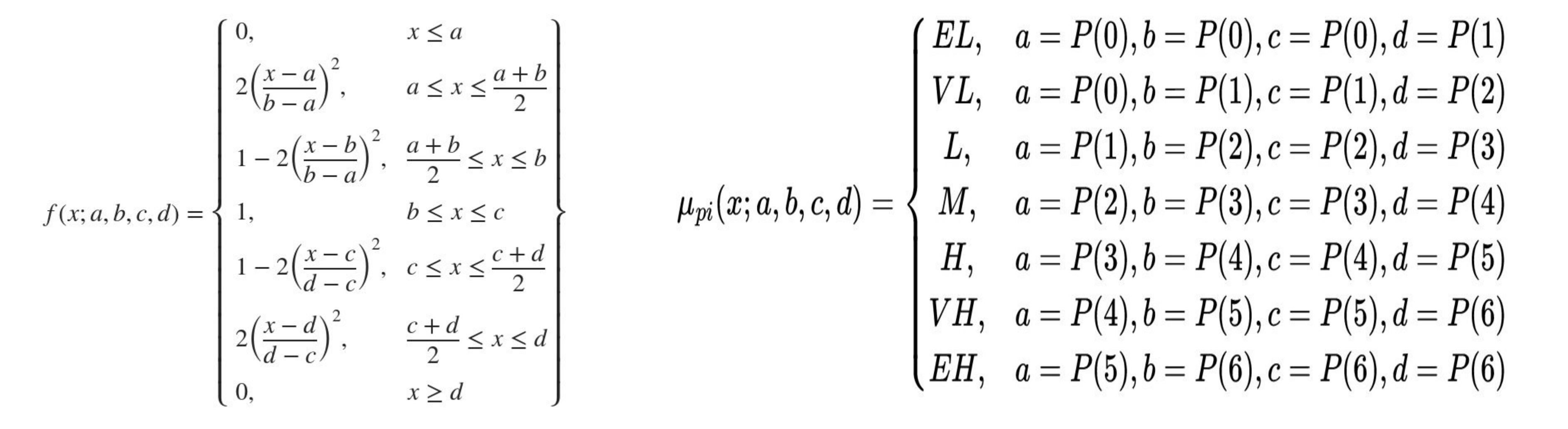
```
maBuy = lambda day: (
    fastMA[day-2] < slowMA[day-2] and
    fastMA[day-1] > slowMA[day-1]
)
maSell = lambda day: (
    fastMA[day-2] > slowMA[day-2] and
    fastMA[day-1] < slowMA[day-1]
)
```

Fuzzy System - Concept

- Moving Average provides buy and sell signals, if using the simple MA strategy then must immediate take action when signals cross over. This can cause extra costs or losses.
- Using Fuzzy Logic we can wait to make decisions until a certain threshold is exceeded
- Difference Between two Moving Averages is divided into 7 extents [1][2]:
    - Extremely Low (EL), Very Low (VL), Low (L), Middle (M), High (H), Very High (VH), Extremely High (EH)
    - Moving Average Differences acts as centroids for Pi Shaped Input Membership Function
    - Fuzzy Input Memberships then get mapped to Fuzzy Controller Based on Rules
- Output Membership Functions have a range of [-1,1] mapped to Negative, Medium, Positive
    - Y > (0.5) indicates a strong buy (bullish trend) -> Results in Buy Action
    - Y < (- 0.5) indicate a strong sell (bearish trend) -> Results in Sell Action
- Rule 1: If x is EL or x is VL or x is L then y is Negative
- Rule 2: If x is M then y is Medium
- Rule 3: If x is EL or x is VH or x is H then y is Positive

Fuzzy System - Implementation

- Difference Between two Moving Averages is divided into 7 extents array P:
  - Extremely Low (EL), Very Low (VL), Low (L), Middle (M), High (H), Very High (VH), Extremely High (EH) [1][2]
  - These values serve as the parameters for the input membership function as defined below:



$$f(x;a,b,c,d)=\left\{\begin{array}{ll} 0, & x\le a \\ 2\left(\frac{x-a}{b-a}\right)^2, & a\le x\le \frac{a+b}{2} \\ 1-2\left(\frac{x-b}{b-a}\right)^2, & \frac{a+b}{2}\le x\le b \\ 1, & b\le x\le c \\ 1-2\left(\frac{x-c}{d-c}\right)^2, & c\le x\le \frac{c+d}{2} \\ 2\left(\frac{x-d}{d-c}\right)^2, & \frac{c+d}{2}\le x\le d \\ 0, & x\ge d \end{array}\right\}$$

$$\mu_{pi}(x;a,b,c,d)=\begin{cases} EL, & a=P(0),b=P(0),c=P(0),d=P(1) \\ VL, & a=P(0),b=P(1),c=P(1),d=P(2) \\ L, & a=P(1),b=P(2),c=P(2),d=P(3) \\ M, & a=P(2),b=P(3),c=P(3),d=P(4) \\ H, & a=P(3),b=P(4),c=P(4),d=P(5) \\ VH, & a=P(4),b=P(5),c=P(5),d=P(6) \\ EH, & a=P(5),b=P(6),c=P(6),d=P(6) \end{cases}$$

Source: Matlab - Pi MF

Fuzzy System - Implementation

- The Output Membership Function is defined as such
  - Negative (Neg), Medium (Med), Positive (Pos)

$$f(x;a,b,c,d)=\left\{\begin{array}{ll}0, & x\le a\\ 2\left(\frac{x-a}{b-a}\right)^2, & a\le x\le\frac{a+b}{2}\\ 1-2\left(\frac{x-b}{b-a}\right)^2, & \frac{a+b}{2}\le x\le b\\ 1, & b\le x\le c\\ 1-2\left(\frac{x-c}{d-c}\right)^2, & c\le x\le\frac{c+d}{2}\\ 2\left(\frac{x-d}{d-c}\right)^2, & \frac{c+d}{2}\le x\le d\\ 0, & x\ge d\end{array}\right\}$$

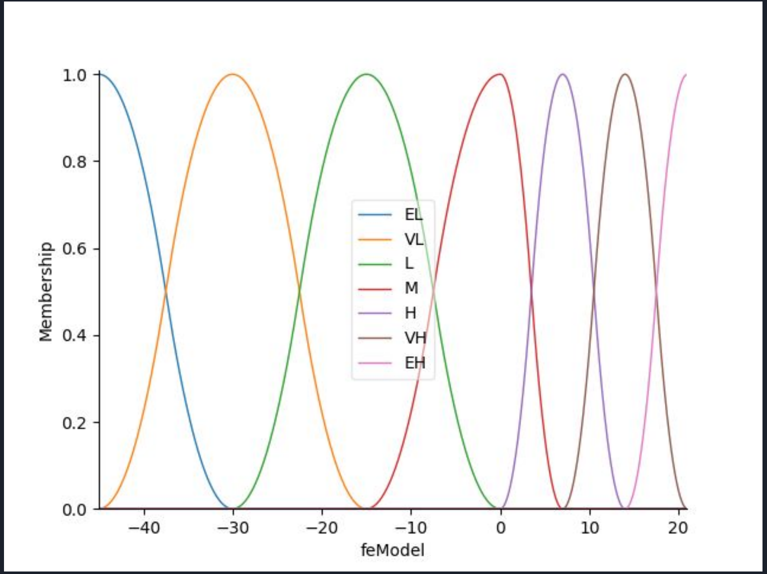
$$\mu_{pi}(x;a,b,c,d)=\begin{cases}Neg, & a=-1,b=-1,c=-1,d=0\\ Med, & a=-1,b=0,c=0,d=1\\ Pos, & a=0,b=1,c=1,d=1\end{cases}$$

Source: Matlab - Pi MF

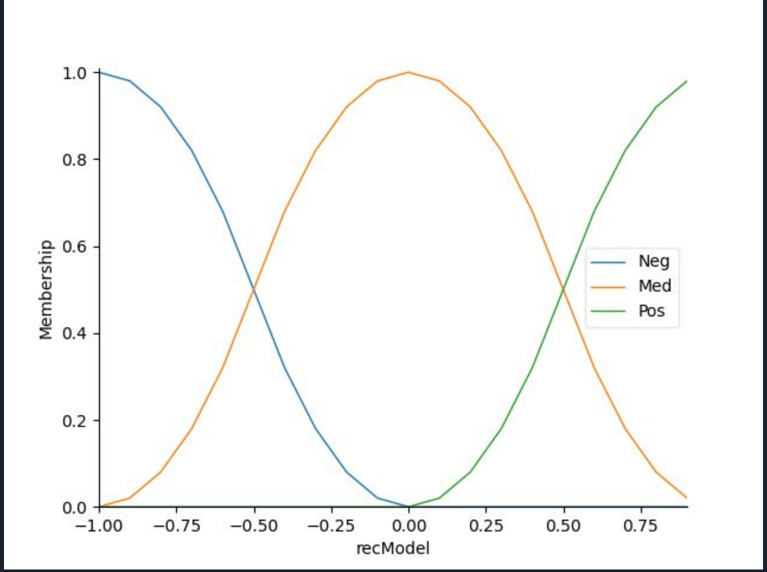
Fuzzy System - Implementation

- Difference Between two Moving Averages is divided into 7 extents:
  - Extremely Low (EL), Very Low (VL), Low (L), Middle (M), High (H), Very High (VH), Extremely High (EH) [1][2]
  - Fuzzy Input Memberships then get mapped to Fuzzy Controller Based on Rules

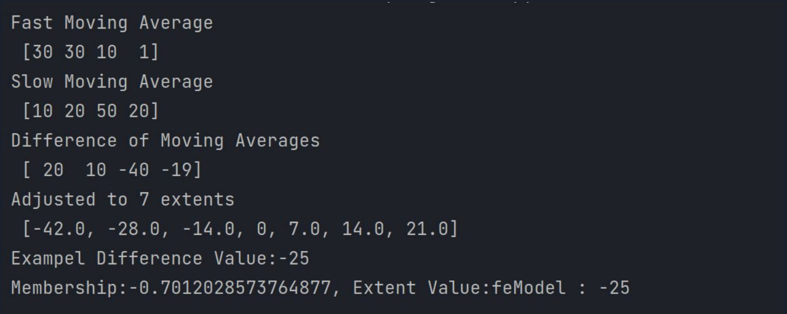
Input Membership Function



Output Membership Function

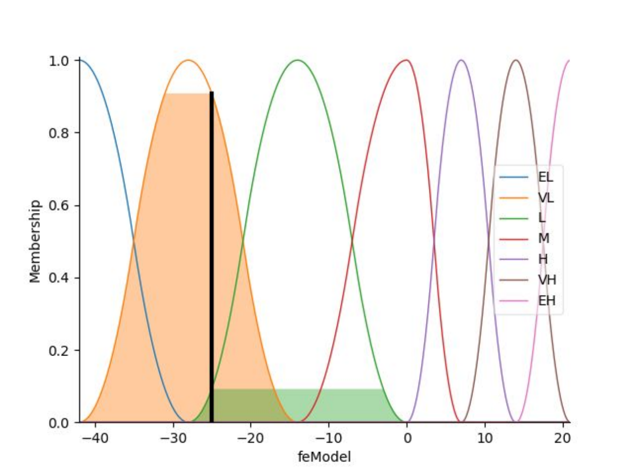


Fuzzy System - Implementation Testing Sell Signal

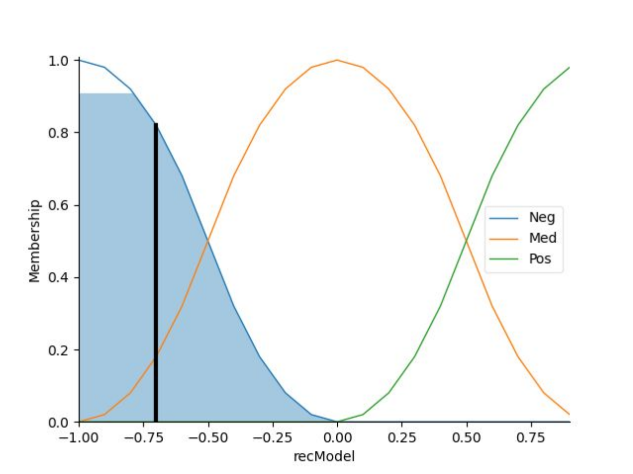


```
Fast Moving Average
 [30 30 10  1]
Slow Moving Average
 [10 20 50 20]
Difference of Moving Averages
 [ 20  10 -40 -19]
Adjusted to 7 extents
 [-42.0, -28.0, -14.0, 0, 7.0, 14.0, 21.0]
Exampel Difference Value:-25
Membership:-0.7012028573764877, Extent Value:feModel : -25
```

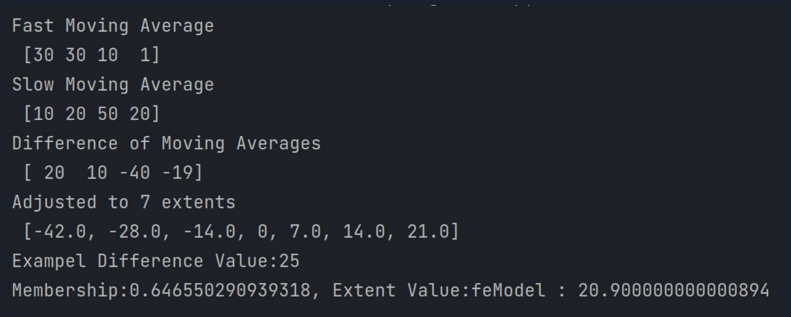
Input Membership Function



Output Membership Function

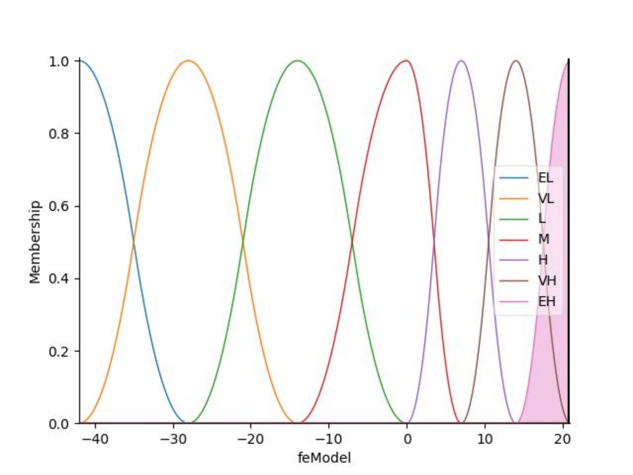


Fuzzy System - Implementation Testing Buy Signal

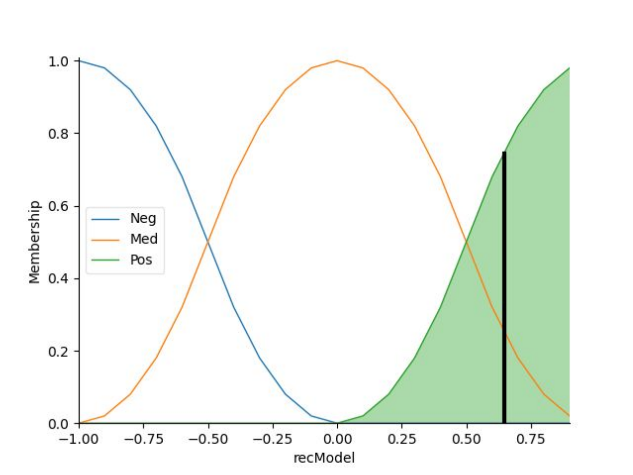


```
Fast Moving Average
 [30 30 10  1]
Slow Moving Average
 [10 20 50 20]
Difference of Moving Averages
 [ 20  10 -40 -19]
Adjusted to 7 extents
 [-42.0, -28.0, -14.0, 0, 7.0, 14.0, 21.0]
Exampel Difference Value:25
Membership:0.646550290939318, Extent Value:feModel : 20.900000000000894
```

Input Membership Function

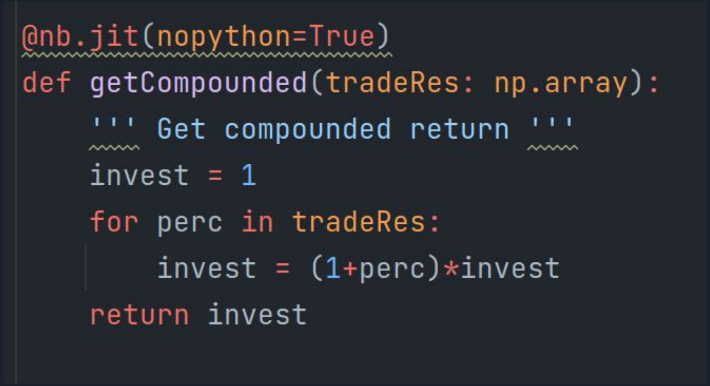


Output Membership Function



Genetic Algorithm - Implementation

1. Initialize Population of n Strategies (Individuals)
    - a. Calculate Fitness of for each Strategy
    - b. Generation = 0
2. For generation in Generations(size):
    - a. Generation = Generation + 1
        - i. Keep Best 30% of Strategies (Elitism)
    - b. Selection: (Tournament Style)
    - c. Generate offsprings of Best Strategies
    - d. Mutate offsprings
    - e. Introduce a subset of new random strategies
    - f. Evaluate Fitness of Population
    - g. End
3. Output the best strategy



```python
@nb.jit(nopython=True)
def getCompounded(tradeRes: np.array):
    ''' Get compounded return '''
    invest = 1
    for perc in tradeRes:
        invest = (1+perc)*invest
    return invest
```

Maximization of fitness function: Compounded Profit Returns from trades executed

Implementation - Changes from Authors

- Authors used Adaptive Moving Average, Typical Moving Average, and Triangular Moving Average in addition to Simple Moving Average
    - Using Exponential Moving Average and Weighted Moving Average
- Authors had a fixed long (m) and short (n) period lengths [1][2]
    - m = {10, 20, 50, 100, 150, 200} ; n = {1, 3, 5, 10, 15, 20} and generated 32 combinations
    - Changed to let the GA iterate through a maximum and minimum window range
- Authors used *Backtrader* to implement backtest [2]
    - Wrote own method of backtesting
- Authors examined Crude Oil Futures [1][2]
    - This implementation will take a look at Blue Chip Tech Stocks

Genetic Algorithm - Details of Changes

- Optimizable Parameters -
  - Moving Average Type: Simple Exponential, Weighted
  - Moving Average Price Field: (Open, Low, High, Close, Adj Close)
  - Moving Average Length: range [3 - 300]
- What can be optimized:
  - Mean: Mean Percentage of gain/loss over all trades made
  - Median: Median Percentage of gain/loss over all trades made
  - Compounded: Multiples of Initial Investment from applying strategy
- Since Optimizing Strategy Across Multiple Tickers will end up with fitness values for each ticker, to combine fitness value across all tickers can implement
  - Min: the minimum of all Fitness Values
  - Mean: the mean of all Fitness Values
  - Median: the median of all Fitness Values
  - Max: the max of all Fitness Values

Implementation - Technical Indicators

<u>Simple Moving Average (SMA)</u>

A = Average in period n

n = Number of time periods (window)

$$SMA = \frac{A_1 + A_2 + ... + A_n}{n}$$

<u>Exponential Moving Average (EMA)</u>

P = Price ;K = 2 / (N+1)

N = Number of time period (window)

$$EMA_n = P_n * k + EMA_{n-1} * (1 - k)$$

<u>Weighted Moving Average (WMA)</u>

P = Price; W = Weight ; n= number of periods in weight group

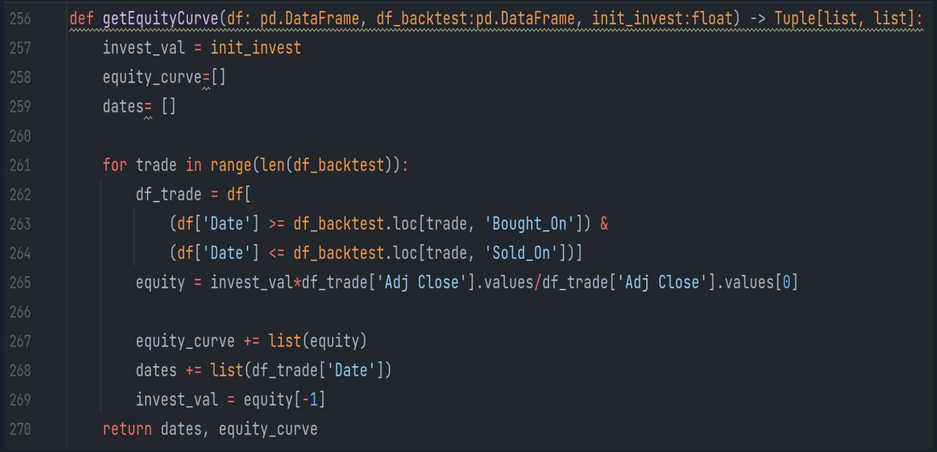
- More weight assigned to earlier prices

$$WMA = \frac{\sum_{t=1}^{n} W_t * P_t}{\sum_{t=1}^{n} W_t}$$

Implementation - Backtesting Strategy

- Looks the original price dataframe and the dataframe that contains trades executed
- Starts with initial investment and based on strategy evaluation (in this case compounded) returns the equity curve for the optimized strategy and buy and hold strategy

Backtesting Algorithm

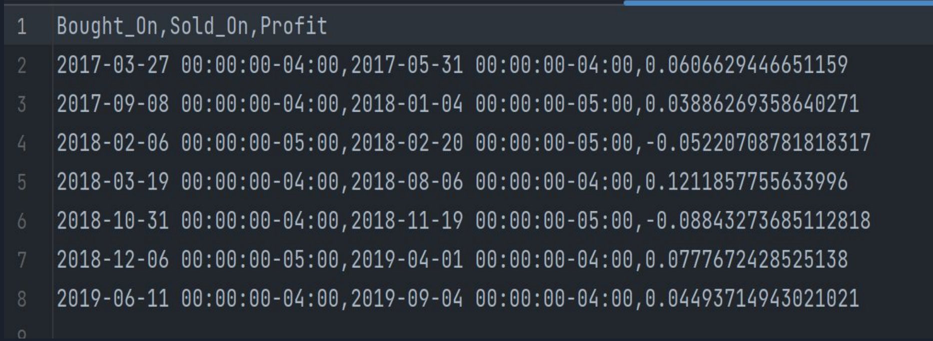


```python
def getEquityCurve(df: pd.DataFrame, df_backtest:pd.DataFrame, init_invest:float) -> Tuple[list, list]:
    invest_val = init_invest
    equity_curve=[]
    dates= []

    for trade in range(len(df_backtest)):
        df_trade = df[
            (df['Date'] >= df_backtest.loc[trade, 'Bought_On']) &
            (df['Date'] <= df_backtest.loc[trade, 'Sold_On'])]
        equity = invest_val*df_trade['Adj Close'].values/df_trade['Adj Close'].values[0]

        equity_curve += list(equity)
        dates += list(df_trade['Date'])
        invest_val = equity[-1]
    return dates, equity_curve
```

Example of Trades Executed for Google



```
Bought_On,Sold_On,Profit
2017-03-27 00:00:00-04:00,2017-05-31 00:00:00-04:00,0.0606629446651159
2017-09-08 00:00:00-04:00,2018-01-04 00:00:00-05:00,0.03886269358640271
2018-02-06 00:00:00-05:00,2018-02-20 00:00:00-05:00,-0.05220708781818317
2018-03-19 00:00:00-04:00,2018-08-06 00:00:00-04:00,0.1211857755633996
2018-10-31 00:00:00-04:00,2018-11-19 00:00:00-05:00,-0.08843273685112818
2018-12-06 00:00:00-05:00,2019-04-01 00:00:00-04:00,0.0777672428525138
2019-06-11 00:00:00-04:00,2019-09-04 00:00:00-04:00,0.04493714943021021
```

Model - Parameters

Training Ticker = [NVDA, AMZN, GOOG, TSLA, TSM]

Testing Tickers = [AAPL, GOOGL, META]

Number of strategies = 40

Keep Best (Elitism) = 30%

Number of Generations = 20

Moving Average Types = [Simple, Exponential, Weighted]

Moving Average Fields = [Open, Low, High, Close, Adj Close]

Minimum Moving Average Window (Days) = 3

Maximum Moving Average Window (Days) = 200

Max Window Frame Mutation Increment = +/- Range(-10,10)

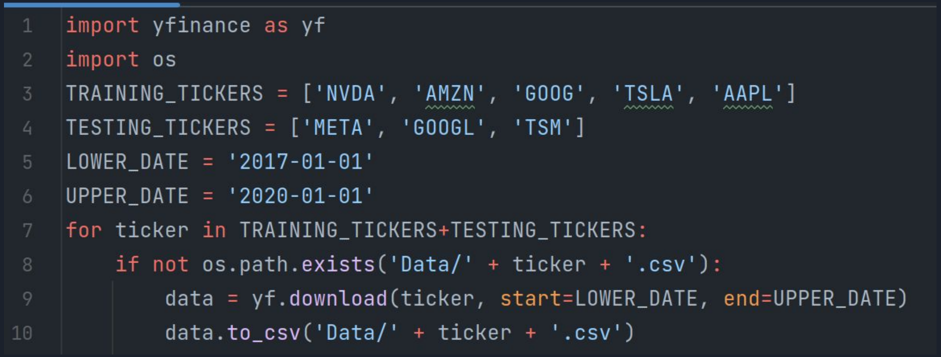
Mutation of Strategy = 20 %

Percentage of Crossover = 33%

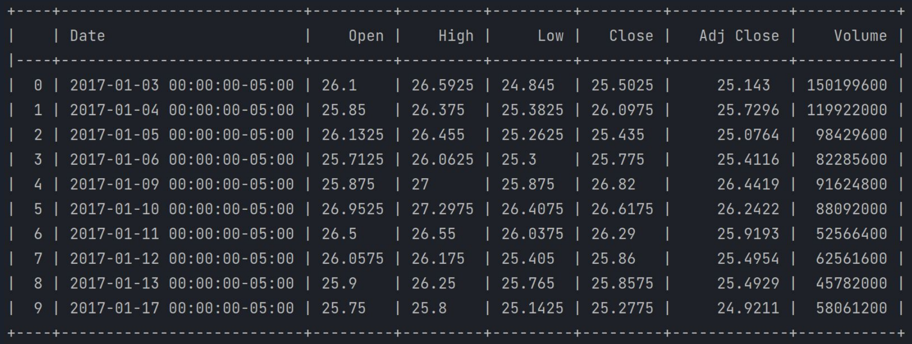
Strategy Evaluation = Compounded

Model - Data

- Data is retrieved from Yahoo Finance Using *yfinance* module
- Starting Date = 1/1/2017
- End Date = 1 /1/2020
  - Selected this Data Frame to not take into account of volatility during covid years



```python
import yfinance as yf
import os
TRAINING_TICKERS = ['NVDA', 'AMZN', 'GOOG', 'TSLA', 'AAPL']
TESTING_TICKERS = ['META', 'GOOGL', 'TSM']
LOWER_DATE = '2017-01-01'
UPPER_DATE = '2020-01-01'
for ticker in TRAINING_TICKERS+TESTING_TICKERS:
    if not os.path.exists('Data/' + ticker + '.csv'):
        data = yf.download(ticker, start=LOWER_DATE, end=UPPER_DATE)
        data.to_csv('Data/' + ticker + '.csv')
```



```
+----+---------------------------+---------+---------+---------+---------+-------------+-----------+
|    | Date                      |    Open |    High |     Low |   Close |   Adj Close |    Volume |
|----+---------------------------+---------+---------+---------+---------+-------------+-----------|
|  0 | 2017-01-03 00:00:00-05:00 | 26.1    | 26.5925 | 24.845  | 25.5025 |     25.143  | 150199600 |
|  1 | 2017-01-04 00:00:00-05:00 | 25.85   | 26.375  | 25.3825 | 26.0975 |     25.7296 | 119922000 |
|  2 | 2017-01-05 00:00:00-05:00 | 26.1325 | 26.455  | 25.2625 | 25.435  |     25.0764 |  98429600 |
|  3 | 2017-01-06 00:00:00-05:00 | 25.7125 | 26.0625 | 25.3    | 25.775  |     25.4116 |  82285600 |
|  4 | 2017-01-09 00:00:00-05:00 | 25.875  | 27      | 25.875  | 26.82   |     26.4419 |  91624800 |
|  5 | 2017-01-10 00:00:00-05:00 | 26.9525 | 27.2975 | 26.4075 | 26.6175 |     26.2422 |  88092000 |
|  6 | 2017-01-11 00:00:00-05:00 | 26.5    | 26.55   | 26.0375 | 26.29   |     25.9193 |  52566400 |
|  7 | 2017-01-12 00:00:00-05:00 | 26.0575 | 26.175  | 25.405  | 25.86   |     25.4954 |  62561600 |
|  8 | 2017-01-13 00:00:00-05:00 | 25.9    | 26.25   | 25.765  | 25.8575 |     25.4929 |  45782000 |
|  9 | 2017-01-17 00:00:00-05:00 | 25.75   | 25.8    | 25.1425 | 25.2775 |     24.9211 |  58061200 |
+----+---------------------------+---------+---------+---------+---------+-------------+-----------+
```

Model - Initial Baseline Strategy

```
STARTING_STRAT = {
    'fast_ma_type': 'simple',
    'slow_ma_type': 'simple',
    'fast_ma_field': 'Adj Close',
    'slow_ma_field': 'Adj Close',
    'fast_ma_period': 5,
    'slow_ma_period': 20}
```

Simple MA Model - Optimized Strategies

```
Optimization time: 196.72148323059082
```

```
Optimal Mean Strategy Parameters: {'fast ma type': 'weighted', 'slow ma type':
'exponential', 'fast ma field': 'Adj Close', 'slow_ma_field': 'Adj Close',
'fast_ma_period': 13, 'slow_ma_period': 18}
```

```
Optimal Max Strategy Parameters: {'fast ma type': 'weighted', 'slow_ma_type':
'exponential', 'fast ma field': 'Close', 'slow_ma_field': 'Low',
'fast_ma_period': 14, 'slow_ma_period': 16}
```

```
Optimal Median Strategy Parameters: {'fast ma type': 'weighted', 'slow_ma_type':
'exponential', 'fast ma field': 'Close', 'slow_ma_field': 'Close',
'fast_ma_period': 13, 'slow_ma_period': 16}
```

```
Optimal Min Strategy Parameters: {'fast ma type': 'weighted', 'slow ma type':
'simple', 'fast ma field': 'Open', 'slow_ma_field': 'Low', 'fast_ma_period': 5,
'slow_ma_period': 6}
```

Results - Author’s Result for Max Fitness Function Optimization

Max Fitness = 1.36

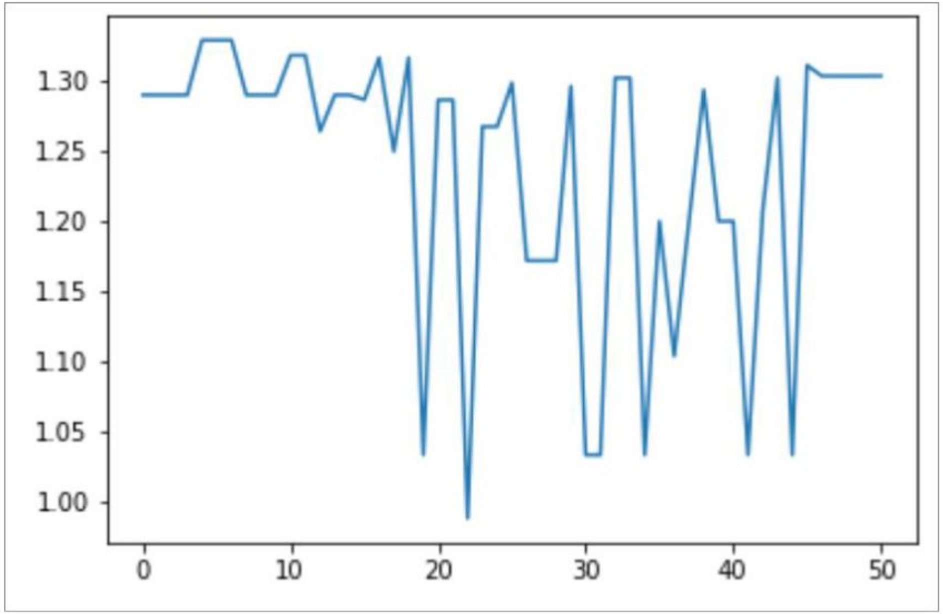


Fig. 6. Convergence plot

Results - Author’s Backtesting Results (Crude Oil)

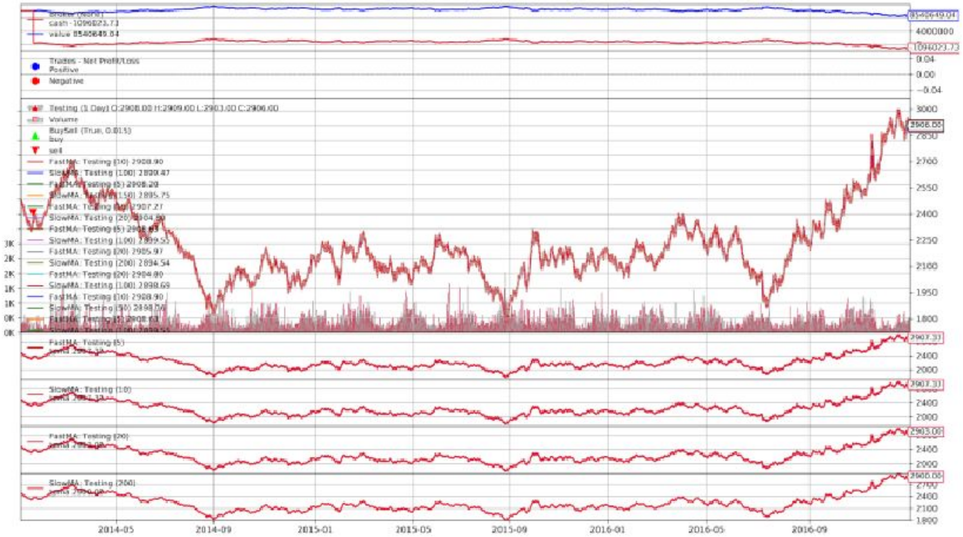
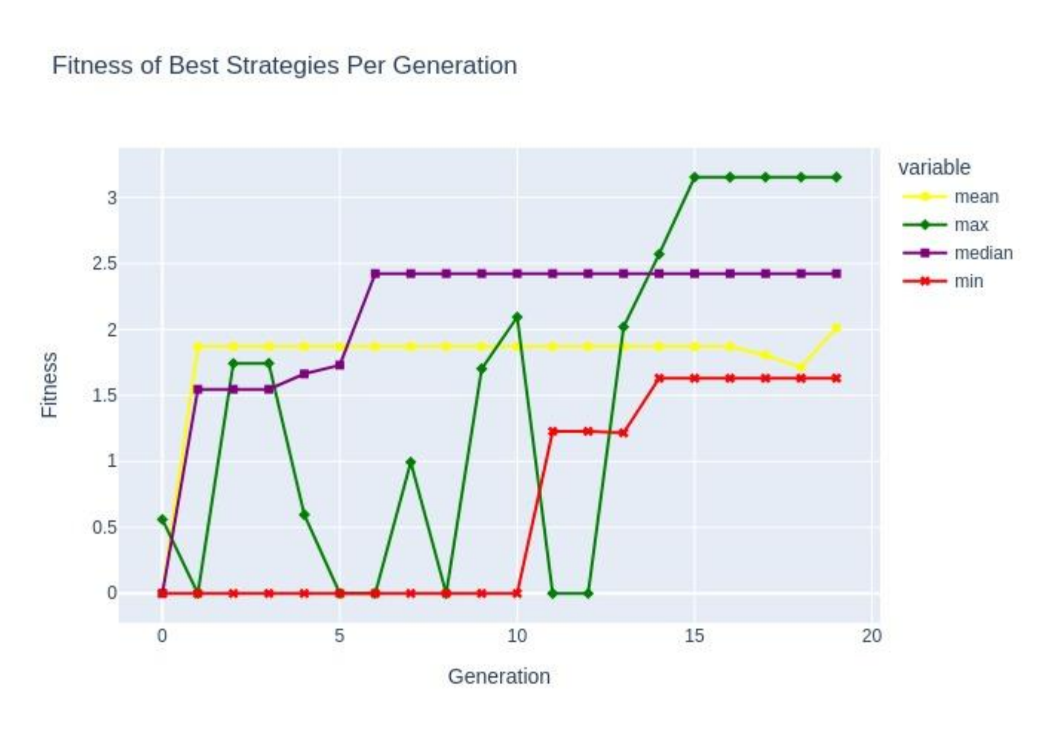


Fig. 8. Training and testing trading plot

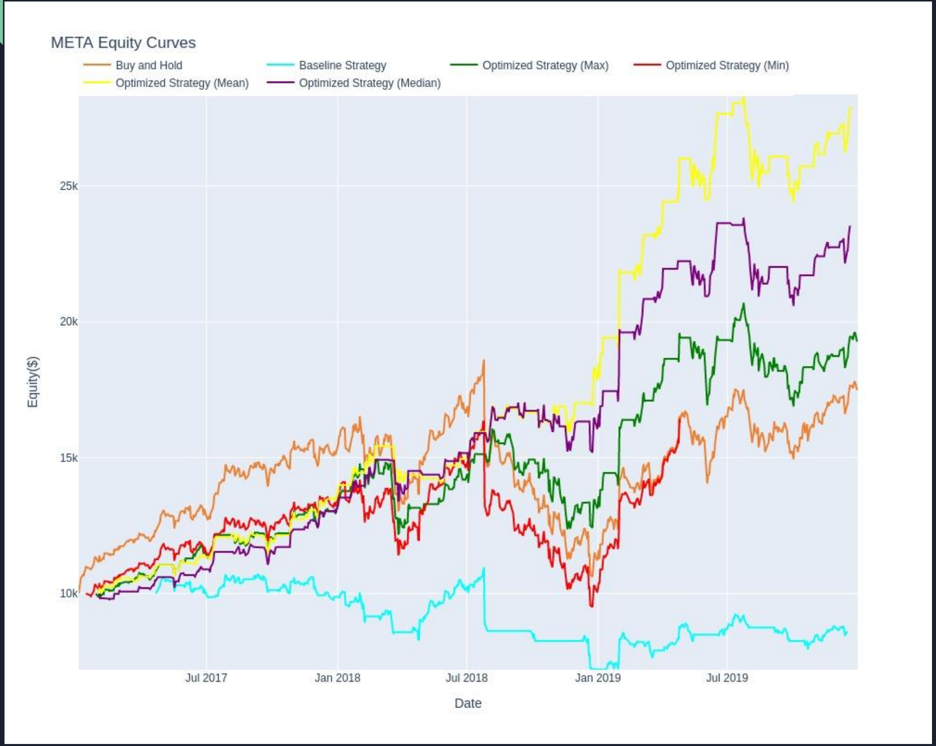
Initial Investment	$10,000,000
ROI = (Net Profit)/Cost * 100	%
Author’s Max Strategy	-17.05983

Results - Simple MA Parameters Training Fitness



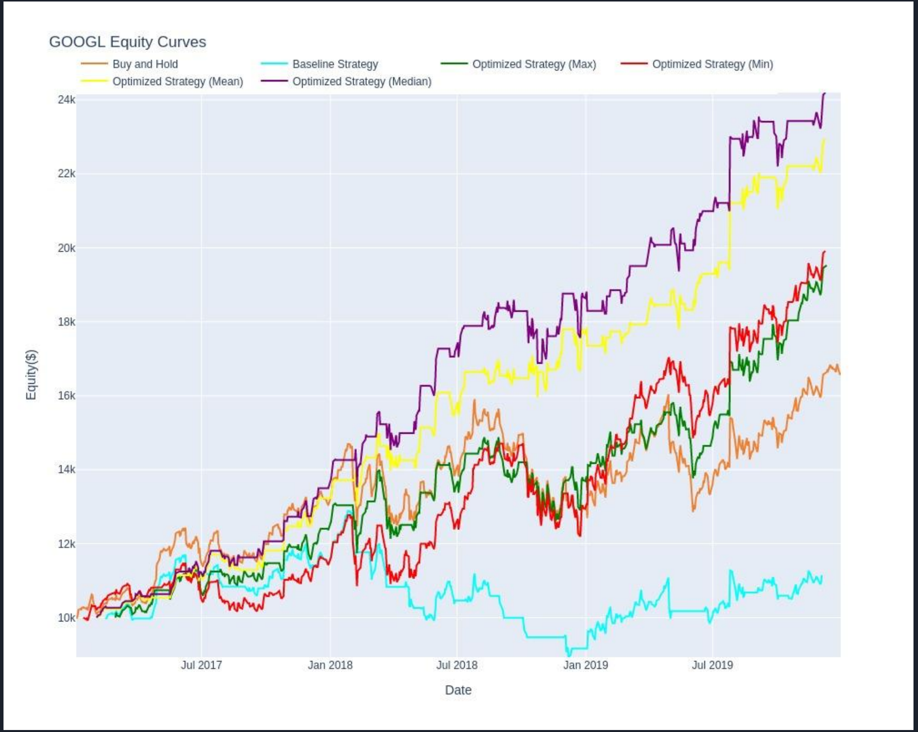
Strategy	Fitness
Mean	2.013362
Max	3.152996
Median	2.423178
Min	1.629903

Results - Simple MA Backtesting Results (META)



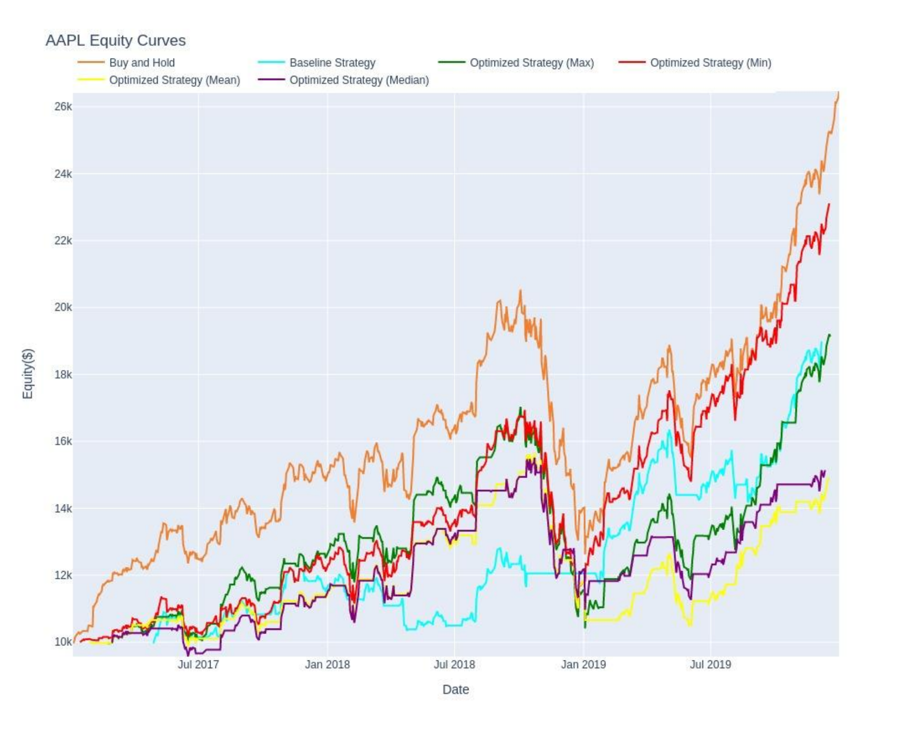
Initial Investment	$10,000
ROI = (Net Profit)/Cost * 100	Percent (%)
Buy and Hold	75.63
Baseline	-15.01
Optimized (Max)	92.7
Optimized (Min)	62.84
Optimized (Mean)	178.57
Optimized (Median)	135.54

Results - Simple MA Backtesting Results (GOOGL)



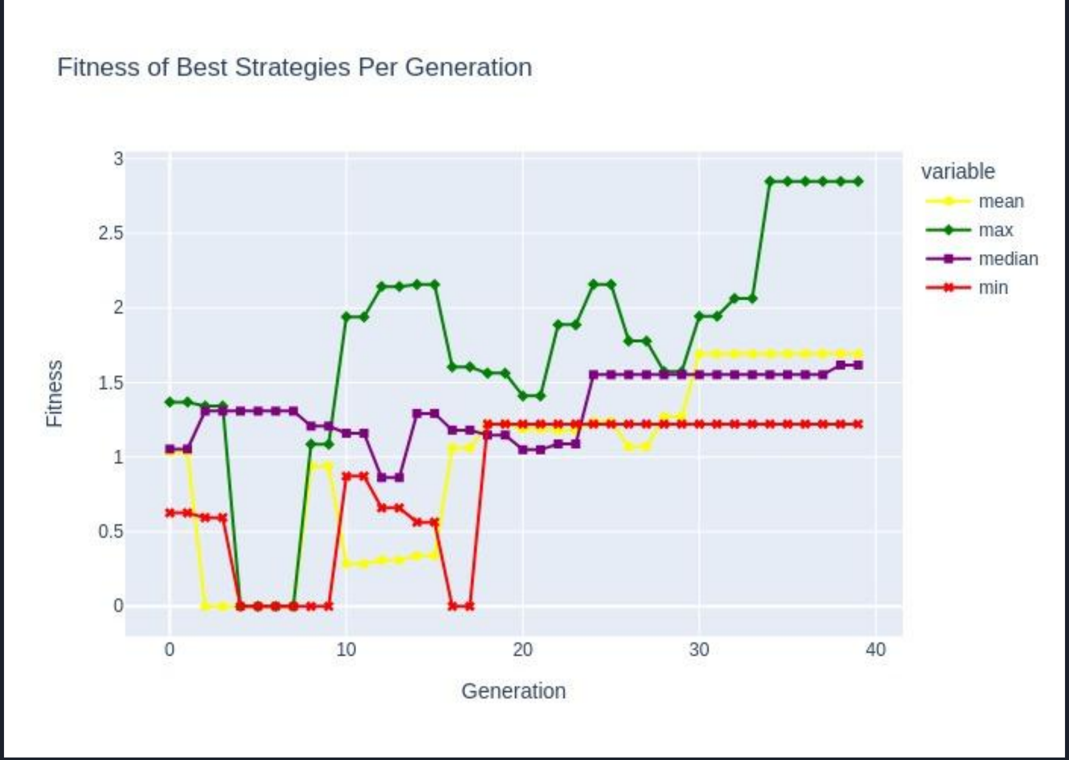
Initial Investment	$10,000
ROI = (Net Profit)/Cost * 100	Percent (%)
Buy and Hold	65.80
Baseline	11.52
Optimized (Max)	95.26
Optimized (Min)	99.14
Optimized (Mean)	129.39
Optimized (Median)	141.98

Results - Simple MA Backtesting Results (AAPL)



Initial Investment	$10,000
ROI = (Net Profit)/Cost * 100	Percent (%)
Buy and Hold	164.66
Baseline	89.86
Optimized (Max)	91.78
Optimized (Min)	131.13
Optimized (Mean)	49.22
Optimized (Median)	51.40

Results - Fuzzy Parameters Training Fitness



Strategy	Fitness
Mean	1.692051
Max	2.848038
Median	1.617651
Min	1.221721

Fuzzy Model - Optimized Strategies

```
Optimization time: 2446.594869852066
```

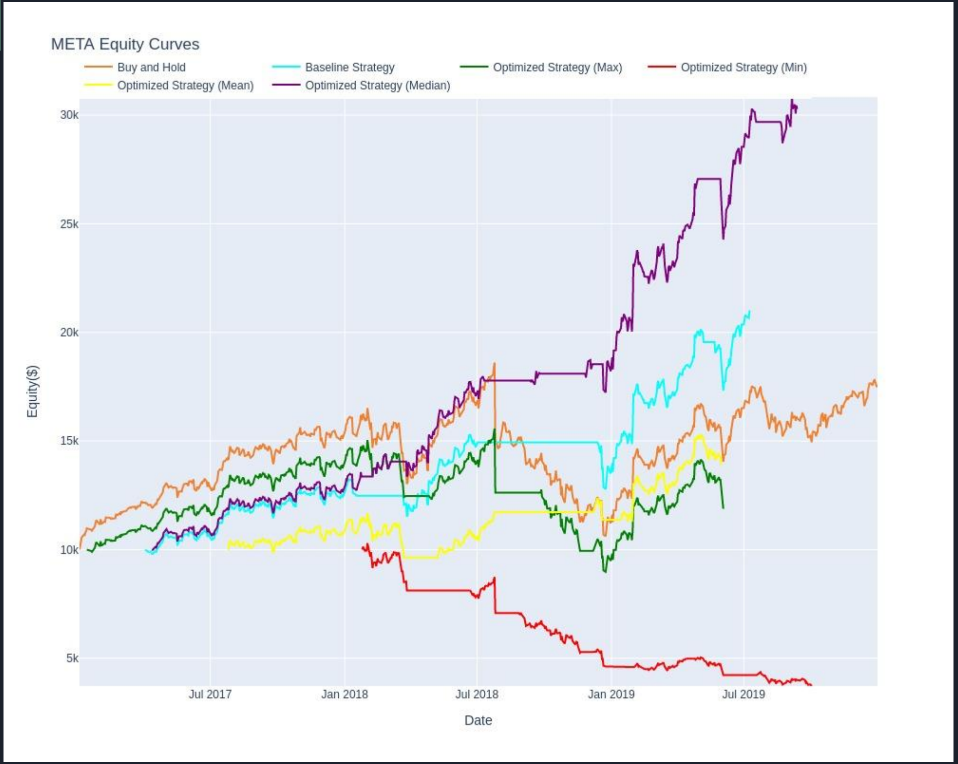
```
Optimal Mean Strategy Parameters: {'fast_ma_type': 'weighted', 'slow_ma_type':
'exponential', 'fast_ma_field': 'Open', 'slow_ma_field': 'Low',
'fast_ma_period': 11, 'slow_ma_period': 21}
```

```
Optimal Max Strategy Parameters: {'fast_ma_type': 'weighted', 'slow_ma_type':
'exponential', 'fast_ma_field': 'High', 'slow_ma_field': 'Low',
'fast_ma_period': 11, 'slow_ma_period': 28}
```

```
Optimal Median Strategy Parameters: {'fast_ma_type': 'exponential',
'slow_ma_type': 'simple', 'fast_ma_field': 'Close', 'slow_ma_field': 'Open',
'fast_ma_period': 3, 'slow_ma_period': 12}
```

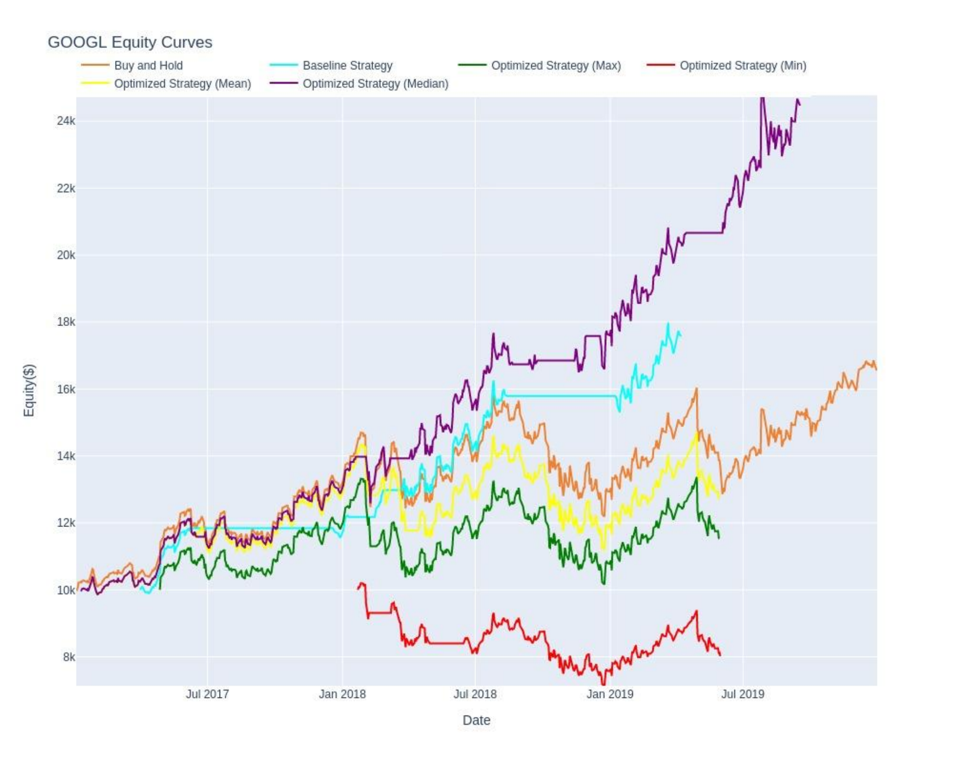
```
Optimal Min Strategy Parameters: {'fast_ma_type': 'weighted', 'slow_ma_type':
'exponential', 'fast_ma_field': 'High', 'slow_ma_field': 'Open',
'fast_ma_period': 12, 'slow_ma_period': 17}
```

Results - Fuzzy Backtesting (META)



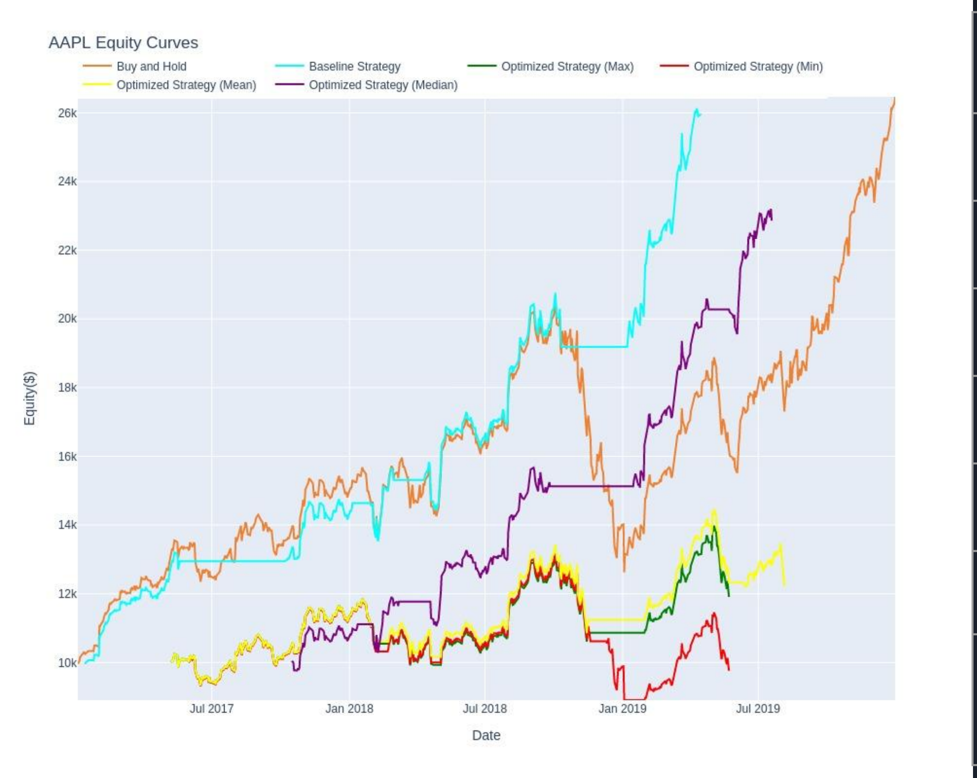
Initial Investment	$10,000
ROI = (Net Profit)/Cost * 100	Percent (%)
Buy and Hold	75.6375
Baseline	110.1757
Optimized (Max)	18.8014
Optimized (Min)	-62.74246
Optimized (Mean)	38.9113
Optimized (Median)	202.7797

Results - Fuzzy Backtesting (GOOGL)



Initial Investment	$10,000
ROI = (Net Profit)/Cost * 100	Percent (%)
Buy and Hold	65.8036
Baseline	75.8266
Optimized (Max)	15.5649
Optimized (Min)	-19.81175
Optimized (Mean)	27.2372
Optimized (Median)	144.706

Results - Fuzzy Backtesting (AAPL)



Initial Investment	$10,000
ROI = (Net Profit)/Cost * 100	Percent (%)
Buy and Hold	164.6595
Baseline	159.523
Optimized (Max)	19.087
Optimized (Min)	-2.41
Optimized (Mean)	22.3
Optimized (Median)	132.08

Conclusion

- Combination of Fuzzy Logic Rule and Moving Average Strategy can form a Fuzzy Moving Average System (FMAS) to generate trade signals
- Genetic algorithm methods can be used for better strategy optimization
- Time Complexity using Python is Exponential when comparing the time to Optimize a Simple MA Strategy vs. Fuzzy Moving Average System
    - `196.72148323059082 vs 2446.594869852066`
- Author concluded that trend over training years show a downward or sideways price movement
    - Explains why author's model sells most of the time in test period even though market is bullish
    - Training data not being a combination of upward and downward trends leads to model biased towards downward trends
- Adjusted implementation of model across Blue Chip Tech Stocks showed that the model was able to optimize strategy parameters and increased performance for majority of testing Tech Stocks as opposed to Buy and Hold Strategy

References

1. Liu, X., An, H., Wang, L., & Guan, Q. (2016). Quantified moving average strategy of crude oil futures market based on fuzzy logic rules and genetic algorithms. *Physica A-statistical Mechanics and Its Applications, 482*, 444-457.
2. Wei, T. C., Singh, L., Muthuraman, R., & Kumar, I. V. (2018, December 3). *Fuzzy Moving Average System with Genetic Algorithm Optimisation for Trading Crude Palm Oil Futures*. GitHub. Retrieved April 25, 2023, from https://github.com/telescopeuser/Workshop-Project-Submission-Template-Trading
3. Hajimiri, H. (2022). Use of Genetic Algorithm in Algorithmic Trading to Optimize Technical Analysis in the International Stock Market (Forex). *Journal of Cyberspace Studies*, *6*(1), 21-29. doi: 10.22059/jcss.2021.334193.1067
4. Fernández-Blanco, P., Bodas-Sagi, D. J., Soltero, F. J., & Hidalgo, J. I. (2008, July). Technical market indicators optimization using evolutionary algorithms. In *Proceedings of the 10th annual conference companion on Genetic and evolutionary computation* (pp. 1851-1858).