Identify Patterns in Stock Market for Day Trading

Melvin Mathew
Shichun Wang
Rong Tai Damien Tan

Introduction

Pattern recognition in stock time series has attracted the attention of the both business and technical experts in recent years. Technical analysts and traders believe that certain stock chart patterns and shapes are signals for profitable trading opportunities. Generally, these trading strategies have been used for long term investments, but in this project, we will be applying these concepts on day trading. Day trading is speculation in securities, specifically buying and selling financial instruments within the same trading day.

Researchers have been using two main approaches to recognize patterns from time series data. These are template based approaches and rule based approaches. In both these approaches we need to fix a window size. In this project we use an novel approach of using genetic algorithm to get this window size and combine both template and rule based approaches.

Here we work with five (5) special patterns of stock time series which are used for stock trend predictions. We use only one (1) in case of rule-based approach. Our main focus is to recognize patterns and determine the neckline stock time series data. This time series data is per minute stock prices for seven (7) days. In this project we are only looking at downtrend reversal patterns, i.e, patterns which predict an uptrend.

Perceptually Important Points

Perceptually Important Points (PIP) represents the minimal set of data points which are necessary to form a pattern. To identify PIPs in a sequence P, we use the following procedure.

1. The first two PIPs will be the first and last points of P.
2. The next PIP will be the point in P with maximum distance to the two PIPs directly left and right of it.
3. Repeat step 2 until number of desired PIPs are obtained.

An example of the process of finding PIPs is given in Figure 1.

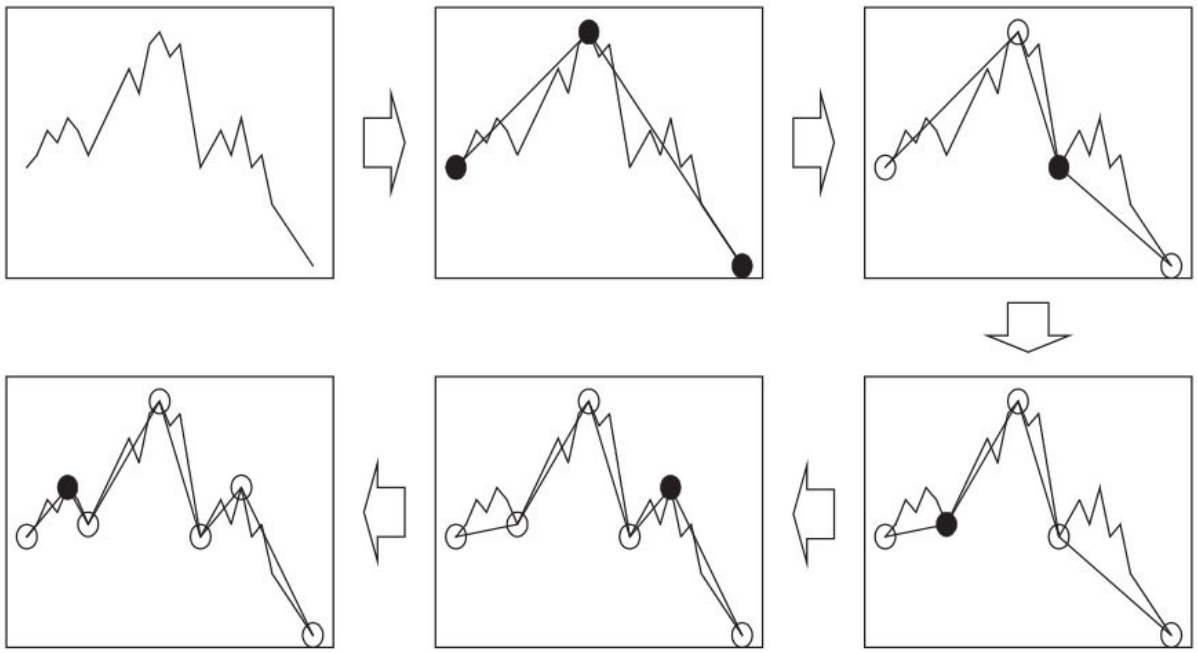


Fig 1: Identification of 7 PIPs for Head and Shoulders pattern.

To determine the maximum distance for a given point between the two adjacent PIPs, two distance measures are proposed. The first measure is the perpendicular distance (PD) between the test point p_3 and the line connecting the two adjacent PIPs as shown in Fig. 2a, i.e.,

$$Slope(p_1, p_2) = s = \frac{y_2 - y_1}{x_2 - x_1}$$

$$x_c = \frac{x_3 + (sy_3) + (s^2 x_2) - (sy_2)}{1 + s^2} - (x_3)^2$$

$$y_c = (sx_c) - (sx_2) + y_2$$

$$PD(p_3, p_c) = \sqrt{(x_c - x_3)^2 + (y_c - y_3)^2}$$

The second measure, depicted in Fig. 2b, is the vertical distance (VD) between the test point p_3 and the line connecting the two adjacent PIPs, i.e.,

$$VD(p_3, p_c) = |y_c - y_3| = \left| \left(y_1 + (y_2 - y_1) \frac{x_c - x_1}{x_2 - x_1} - y_3 \right) \right|$$

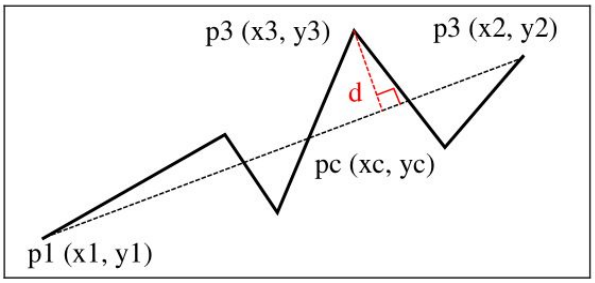


Fig 2a: Perpendicular distance

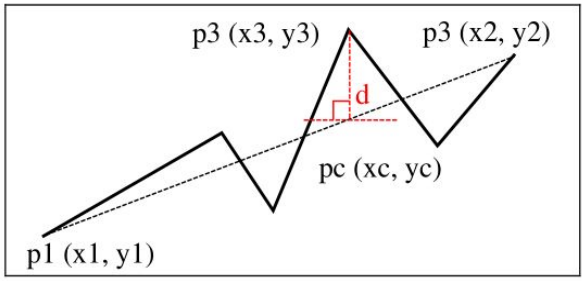


Fig 2b: Vertical distance

Template-based Matching

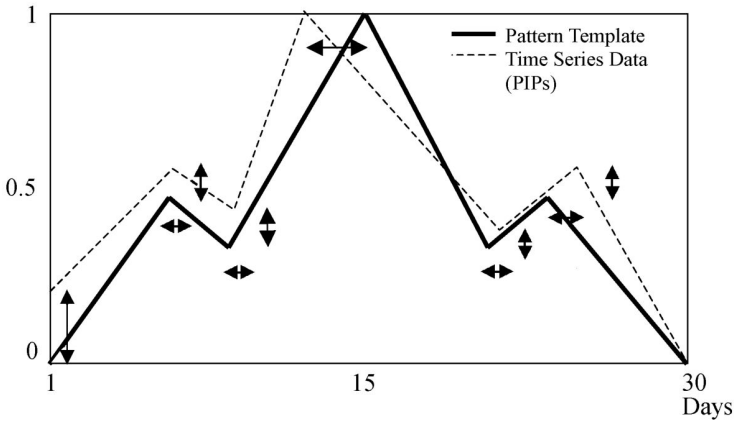


Fig 3: Template matching distance measure(DM)

A template is an instance of a time-series normalized between 0 and 1, which is meant to be compared to the corresponding PIP points obtained from the test time-series. We compute the differences in the corresponding points between the template and the PIP values, and set a threshold value to decide if the PIP points matches the template.

The algorithm is as follows:

1. Compute the PIP points in the input time-series. Number of PIP points need to be same as number of points in template.
2. For each PIP point and corresponding template point (in sequential order), sum the differences in x-axis and y-axis values. This value is the Distance Measure (DM)

Mathematically, the distance measure is defined as follows:

$$AD(P,Q) = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(SP_k - Q_k)^2}$$

$$TD(P,Q) = \sqrt{\frac{1}{n-1}\sum_{k=2}^{n}(SP_k^t - Q_k^t)^2}$$

$$DM(SP,Q) = w_1 \times AD(SP,Q) + (1 - w_1) \times TD(SP,Q)$$

AD is amplitude distance (y-axis), while TD is temporal distance (x-axis). DM is distance measure. ω is the weighting used to change emphasis on the different components, and is taken to be 0.5 in this project. SP are the PIP points, while Q are the points on the template.

The templates used in the template matching are shown below:

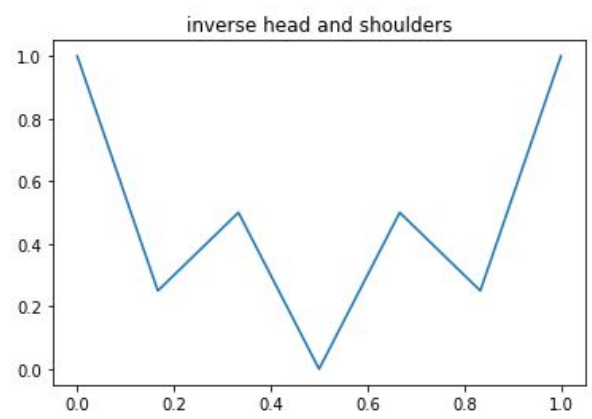


Fig 4a: Inverse head and shoulders template

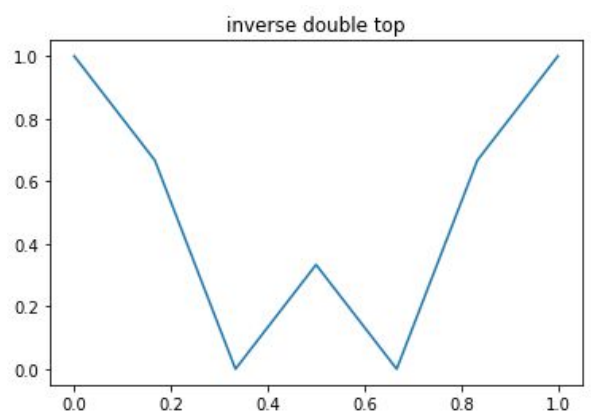


Fig 4b: Inverse double top template

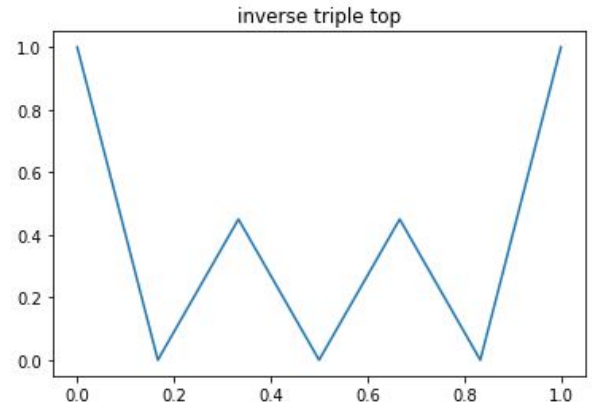


Fig 4c: Inverse triple top template

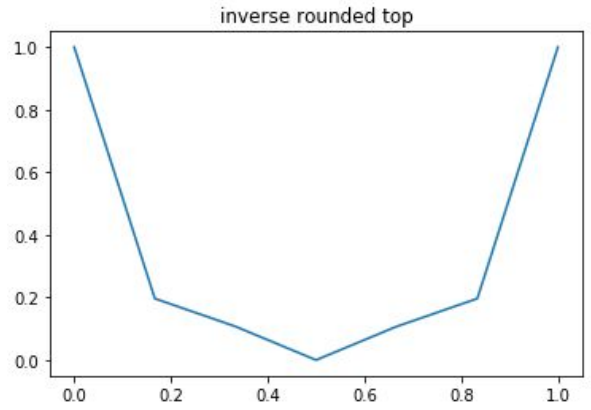


Fig 4d: Inverse rounded top template

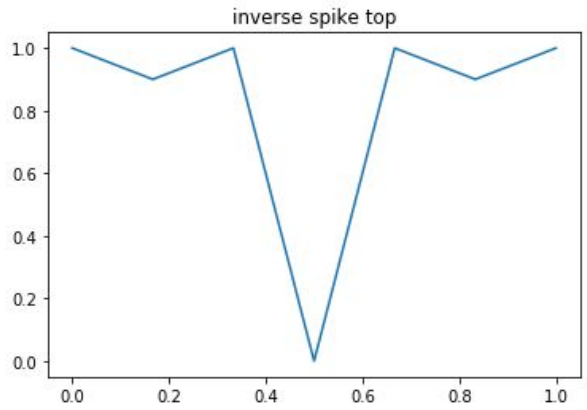


Fig 4e: Inverse spike top template

Rule-based Matching

Besides defining the preferred patterns visually as pattern templates, rules can be defined to describe the shape of the preferred patterns. One of the advantages of applying rule-based pattern matching over the template-based approach is that the relationship between the points is hard to define explicitly in the template-based approach. For example, in a head-and-shoulder pattern, the general guidelines are that the two shoulders in the pattern must be lower than the head while having a similar degree of amplitude (within 15% in average). In such a case, although we can plot a pattern template according to these requirements, such kinds of rules cannot be guaranteed during the pattern-matching process. Patterns with similar shapes as compared to the query pattern but violating certain rules may still be identified. Therefore, the rule-based approach gives yet another method to identify patterns.

Based on the definitions of technical patterns by Lo et al. (2000), we described the one reversal technical patterns in rule format. According to the template-based approach, it is assumed that 7 PIPs, from sp1 to sp7, will be identified first for the pattern matching process. Therefore, the rules for describing the relationships and constraints among these 7 PIPs are defined. The corresponding definition of the inverse head-and-shoulder pattern is as following:

```
Rule set 1 (inverse head-and-shoulder)
```

- `sp4 < sp2 and sp6`
- `sp2 < sp1 and sp3`
- `sp6 < sp5 and sp7`
- `sp3 < sp1`
- `sp5 < sp7`
- `diff(sp2, sp6) < 15%`
- `diff(sp3, sp5) < 15%`

where diff(spx,spy) denotes the difference between data points spx and spy, ‘‘spx < spy and spz’’ denotes that spx must be smaller than spy and spz.

With the defined rules, the sequences can then be evaluated. First, seven PIPs are identified from the sequences. Then, those sequences which can validate all the above rules are identified as a matching result.

Optimal Segmentation of Time-Series

Given a large time-series data, we can expect that there might exist multiple patterns of different widths in the time-series. Although the previous section provides a way of matching patterns in a given time-series, we might want to find patterns of varying widths in different segments of the data. Genetic Algorithm offers a solution to this problem.

Genetic Algorithm

Genetic Algorithm (GA) is an algorithm inspired by the process of natural selection. A population of individuals are first initialized, of which each individual represents the solution to the optimization problem through a predefined encoding. Next, the individuals will go through a series of crossovers and mutations, and on each generation, the best individuals will be chosen through a process called selection. After a number of generations, the best individual will be chosen as the solution to the optimization problem.

Problem Representation in GA

Individual

Each individual is represented as a sequence of numbers, which are the splitting points in the time-series data. For example, in a time-series with 1000 data points, an individual with value {200,600,800} will have 4 segments 1-200, 201-500, 601-800 and 801-1000.

Initialization of Population

A value *dlen* (desired length) is given by the user, which is the average length of a segment that the user expects to get. The first individual will then be the time-series split by the segments of the length *dlen*.

Crossover

Crossover is the operation to exchange the characteristics of two individuals with each other. For our instance of GA, single point crossover is used.

Given two individuals A, B, where

```
A = {10,35,70,120}
B = {25,40,90,140}
```

We generate two random numbers to represent the crossover points in each individual. Assuming we generated two random numbers: (1,2), crossover point for A is 1, while crossover point for B is 2.

Therefore, after crossover, the two individuals are

```
A' = {10,90,140}
B' = {25,40,35,70,120}
```

We will then sort the two individuals to allow easier processing. Take note that it is possible for the individuals to change in size after the crossover operation.

Mutation

Mutation is used to introduce new characteristics into the population. Without mutation, it would not be possible to get out of local minima in the optimization process.

Given an individual A, mutation will either add or remove a random point in the individual. Currently, the probability of adding or removing is equal (0.5 probability). The new point generated cannot be currently in the individual. The new point to be added also must not generate a segment that is smaller than the pattern template size, or no pattern can be matched in the segment. For removal, all points can be removed, unless the individual has no splitting point, which mean that there is only one segment, which is the entire time-series.

Adding new Point

1. Generate range of allowed points that can be added (no points currently in individual, or within fixed distance from individual).
2. Randomly choose a point from the list of allowed points.
3. Add into the individual.
4. Sort the individual.

Removing Point

1. Randomly pick a point to delete, or fail if individual has no points.

2. Delete point.

Selection

Selection is the process of filtering the individuals such that only the “strongest” individuals with the best fitness scores will survive to the next generation. While we can simply pick the individuals with the best fitness scores, the individuals with low fitness scores still need to have non-zero probability of surviving to the next generation, in order to give diversity to the population.

For our optimization problem, lower fitness scores are better. Hence, we use the tournament selection scheme. The tournament selection scheme will run *k* times to select *k* individuals. On every iteration, it will randomly pick *n* individuals from the population, and select the best individual for the next generation. Hence, we can adjust *n* tournament size to be larger to give more selection pressure to choose the best individuals, or smaller *n* to give higher probability for weaker individuals to survive to the next generation.

Evaluation of Fitness

The fitness value is calculated using the template-based method mentioned in the earlier section. Rule-based matching only gives a binary match/not match result, which is not useful to the implementation of GA. Template-based method gives a distance measure (DM), in which a smaller value will represent a closer match between chosen template and PIP points. Hence, the GA algorithm is implemented with a minimization objective.

When matching with different templates for a given segment, the lowest DM value for all template matchings is used (since it is taken to be the correct matching). Since a given individual can have different number of segments, we also use the average DM value for all segments as the fitness score of the individual.

The algorithm for fitness evaluation is as follows:

For Each individual C_i in U(t)
  For Each segment $C_{i,j}$ in C_i
  Extract the PIP's from $C_{i,j}$ and store them in SP
    $fitness(C_{i,j}) = \min_{\forall Q}\{DM(SP,Q)\}$
  End For
    $fitness(C_i) = \frac{1}{n_i+1}\sum_{j=1}^{n_i+1} fitness(C_{i,j})$
End For

Setup

Python is used to implement all the algorithms mentioned in this report.

Numpy library is used to vectorize the PIP identification algorithms and the template matching algorithm. Large speedup in performance has been observed when converting from pure python to numpy code.

DEAP (Distributed Evolutionary Algorithms in Python) library is used to help in the implementation of the Genetic algorithm. In particular, the *cxMessyOnePoint()* function in the library is used to implement the crossover operation. The *selTournament()* in the library is used to implement the selection operation. Mutation and evaluation functions are implemented from scratch.

The various configuration parameters used in the program are listed below:

Parameter Name	**Value**
Population Size	50
Max Generations	1000
DLEN	29
Crossover Rate	0.5
Tournament Size	10
Minimum Segment Length	10
Mutation Rate	0.5
Mutation Add Probability	0.5

Table 1: Configuration parameters for Genetic Algorithm

The parameters are mostly obtained from the research by Chung, F.-L., et al.(2004), where they benchmarked the performance with different parameter values. Some of these parameters will be explored in the testing section.

Visualization

The visualization portion seeks to display the results of the PIP identification and pattern matching algorithms on an interactive webpage, so that users will be able to interact with the chart using a web browser over the internet.

The visualization project source files consists of DataPlot.java(Servlet) and DataPlot.jsp. The project is developed using Java EE.

DataPlot.java
As the Servlet, DataPlot could read data from front JSP page and read data from csv files, which are generated from the Python source files implementing the pattern matching and GA algorithms.

DataPlot.jsp
As the front JSP page, it allows user to enter their chosen stock symbol. It will generate two graphs. One is the trend (time-series) of the stock and one is the PIP output graph.

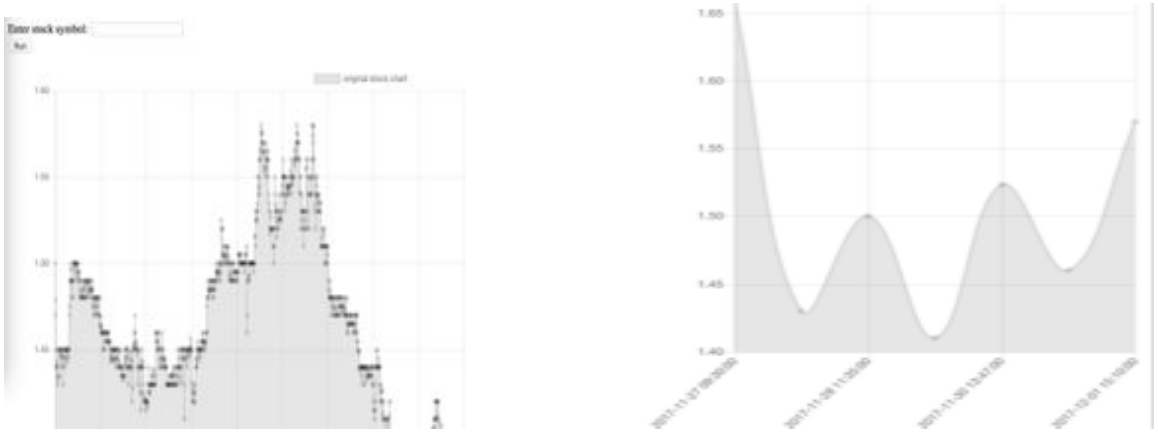


Fig 5: Visualization tool illustration

In order to make the graphs easier to view, the webpage will adjust the x-axis and y-axis according to the size of the window. When moving the mouse to any point on the graph, it will show the value of the data of that point. These two features have been developed using chart.js.

Fig 6: Visualization tool highlight point

Testing

Methodology

Two methodologies are used to test the correctness of the developed strategy. The first method used is to test the accuracy of the pattern recognition algorithm and the genetic algorithm. The second method, backtesting, is used to test the this trading strategy in the stock market. Backtesting will first run the genetic algorithm to break down the time-series into smaller segments, before running the rule-based pattern recognition algorithm to detect patterns in the time-series.

Synthetic Data

The synthetic data is generated by taking a template, scaling the template from 7 points to the desired length, then distorting the shape of the template by shifting the time values and amplitude values of individual by a random amount sampled through a normal distribution. The parameters used to generate the synthetic data are shown below. Take note that the template amplitude and time values are normalised from 0 to 1.

Noise Variance X-axis	0.06
Noise Variance Y-axis	0.05
Points Scaling	70

Table 2: Synthetic Data Parameters

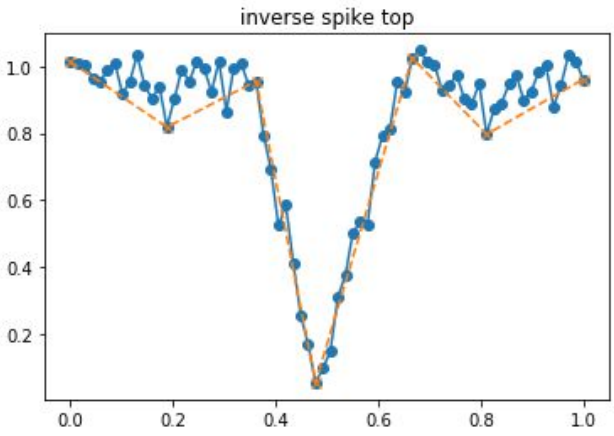


Fig 7: Example synthetic data generation

Single Pattern Test

Each pattern is generated 100 times, and the accuracy in identifying each pattern is measured.

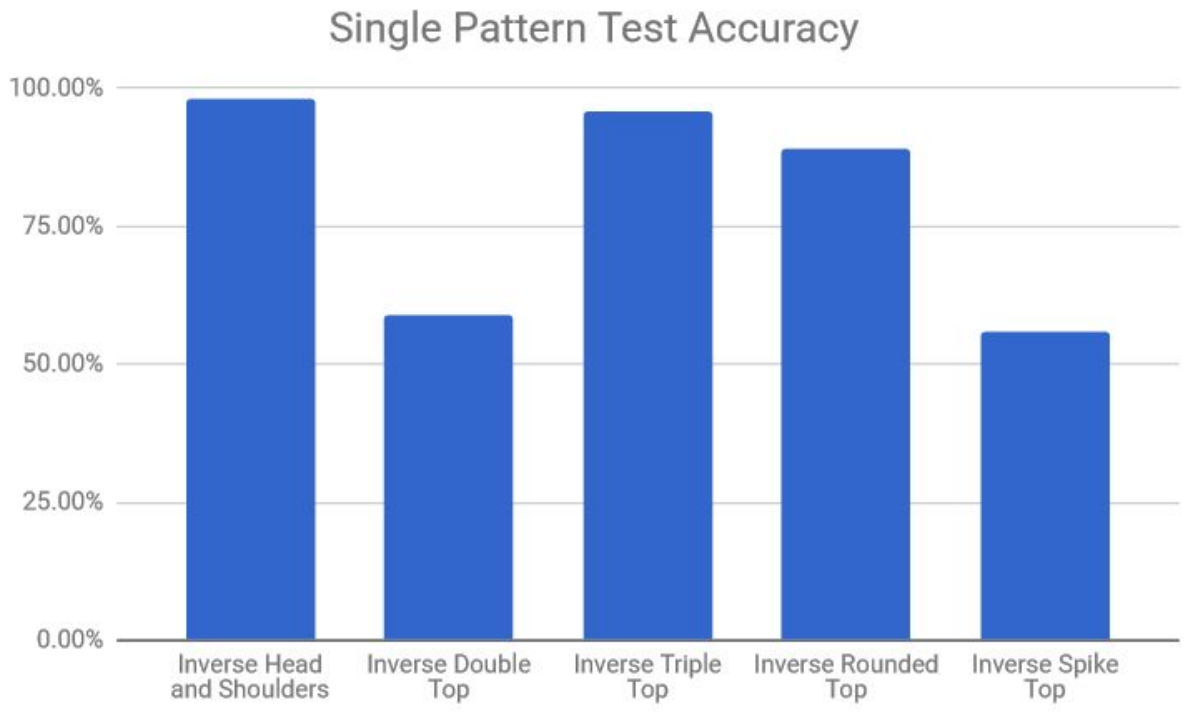


Fig 8: Single pattern test accuracy graph

Pattern Name	Accuracy (100 runs)

Inverse Head and Shoulders	98%
Inverse Double Top	59%
Inverse Triple Top	96%
Inverse Rounded Top	89%
Inverse Spike Top	56%

Table 3: Single pattern test accuracy data

We can observe that three of the patterns perform well in the detection tests, while two of the patterns (Inverse Double Top and Inverse Spike Top), have relatively bad results.

After analysing the results, it was discovered that that inverse spike top was being misclassified as inverse double top. If we are to remove inverse double top from the list of patterns to classify, inverse spike top's accuracy will increase to 96%. However, the converse is not true. Removing inverse spike top from the list does not increase the accuracy of inverse double top. The poor results for inverse double top is corroborated in the paper by Fu, Tak-Chung, et al. (2007).

Multiple Pattern GA Test

To test the genetic algorithm, we generate a series of synthetic patterns and concatenate them into a time-series data. The lengths of the patterns are preset, and are different for each pattern.

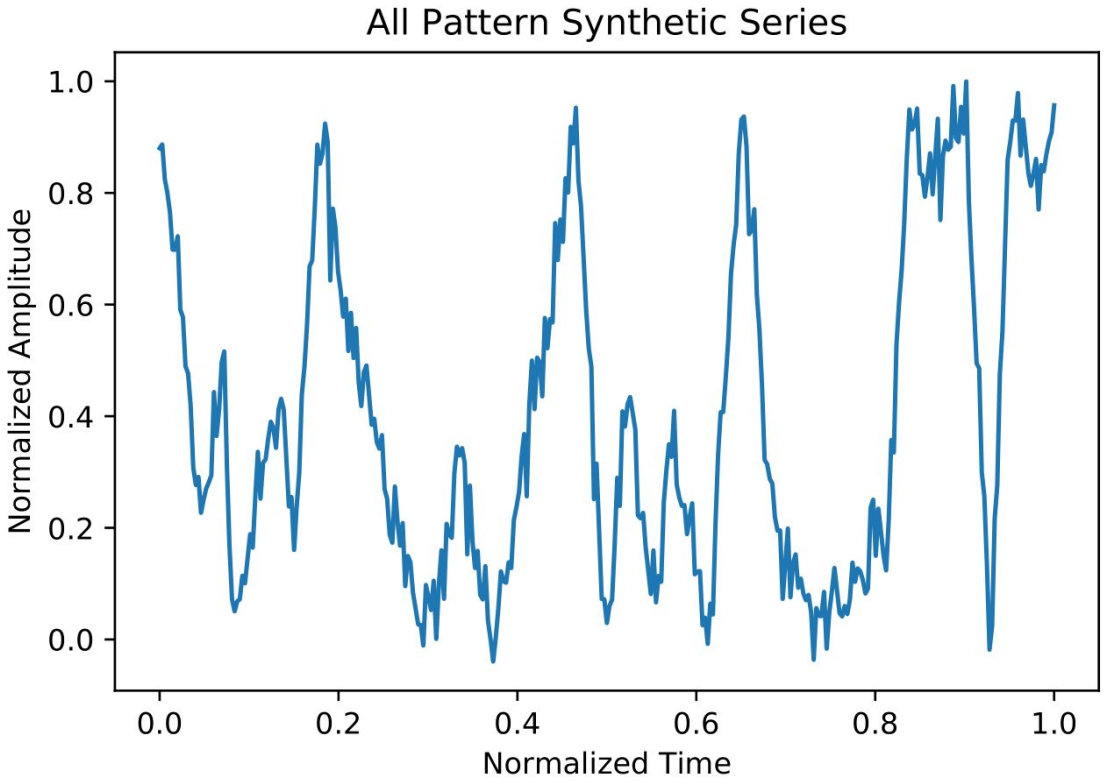
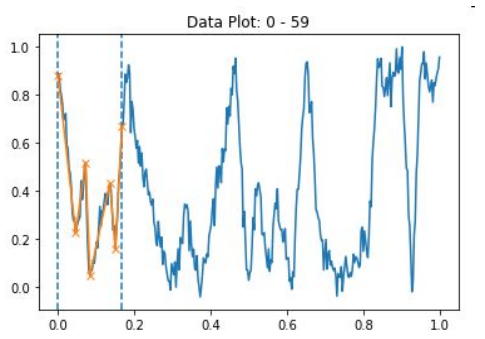
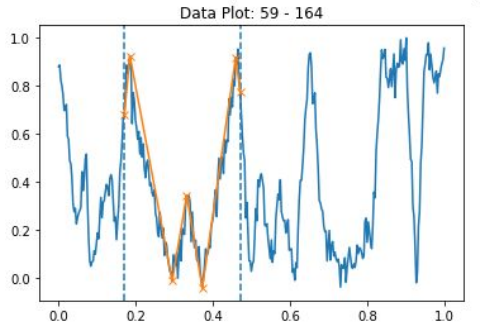
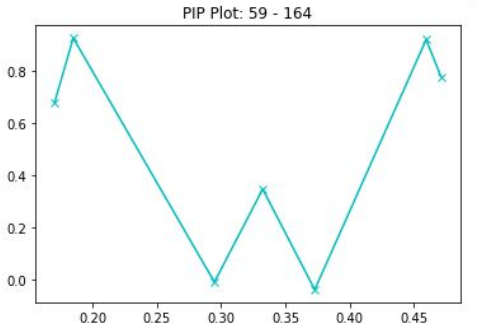
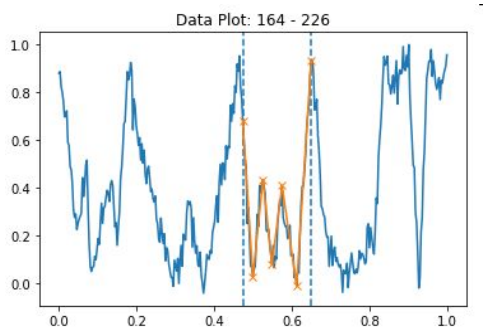
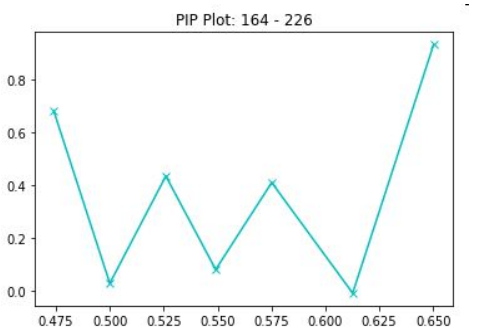
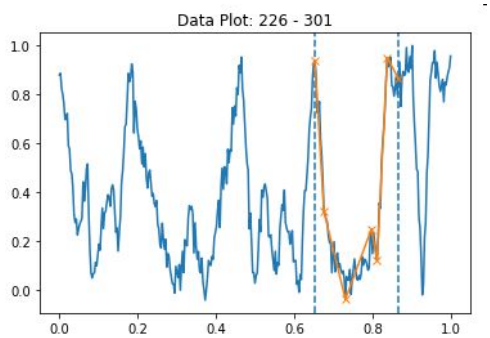
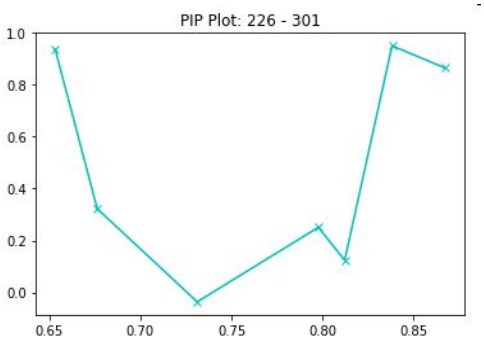


Fig 9: Example of synthetic time-series with five different inverse patterns

Pattern	**Length**
Inverse head and shoulders	63
Inverse double top	98
Inverse triple top	65
Inverse rounded top	65
Inverse spike top	56

Table 4: Multiple pattern test generated length

An example of a successful run of genetic algorithm which correctly classifies all patterns in shown below.

		Inverse Head and Shoulders
		Inverse Double Top
		Inverse Triple Top
		Inverse Rounded Top

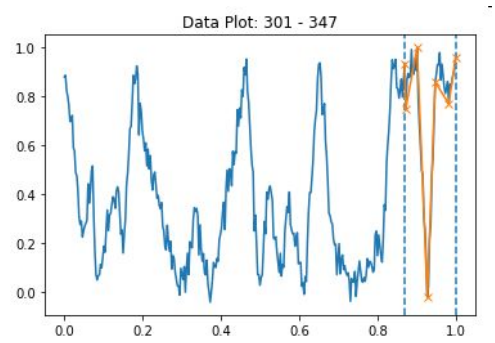
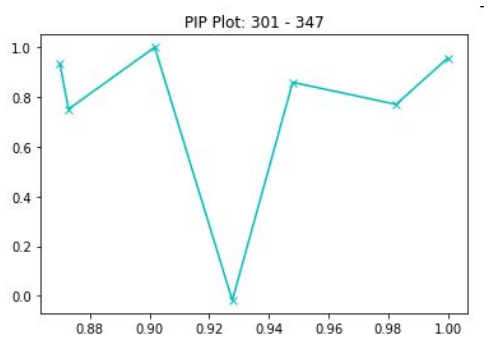
		Inverse Spike Top

Table 5: Example of successful GA run

For this test, it has been discovered that the GA algorithm can sometimes find a lower optimal value for the fitness function by combining the last three patterns into a segment. The PIP algorithm tends to erase the features of the time-series when computing the PIPs, and in that sense the last three patterns can easily be considered as an inverse triple top. A solution that contains less segments also tend to have a higher chance of giving better fitness values. While such an interpretation might not be considered wrong in the case of real world data, it is not the ground truth in our synthetic experiment.

For our synthetic data generation, it has been discovered that 100 generations is enough to get the correct optimal value. However, we still stick to the 1000 generations that was mentioned in the paper by Chung, F.-L., et al.(2004) to use in the backtesting of real world data.

Backtesting

Backtesting is the process of testing a trading strategy on relevant historical data to ensure its viability before the trader risks any actual capital.

We first run the genetic algorithm to break down the time-series into smaller segments that should have higher probabilities of containing patterns, before running the rule-based pattern recognition algorithm to detect patterns in the time-series. For this testing, we only run the rule-based pattern for detecting inverse head and shoulder pattern.

Given the inverse head and shoulders pattern, we can draw a neckline (line connecting the highest price and the second highest price in the pattern). The theory states that the price should rise for some time after the neckline has been crossed (Lo, Andrew, et al., 2000).

The period for testing is from 24 November to 8 December 2017. The test is conducted on 1951 stocks from Nasdaq stock market. The stock data is obtained from the Alpha Vantage web API[1]. We get the optimal segments from the genetic algorithm and only look for inverse head and shoulder pattern. We use three (3) different investment strategies. In all these strategies, a stock is purchased as soon as the current price rises above the neckline. We will invest $150 in each stock.

Out of the 1951 stocks, we detected inverse head and shoulder pattern in the following 9 stocks.

Strategy 1

Purchase stock when price above neckline and sell at the end of the day on the purchase date (on stock market closing time). See table S1 for results. We made a loss of $13.48 (1.1%).

	Stock	Purchase Price	# Stocks	Total Purchase Cost	Purchase Date	Sell Price	Total Sell Price	Sell Date	Profit/ Loss
1	ARLZ	1.58	94	148.52	2017-12-01 15:14:00	1.52	142.88	2017-12-01 16:00:00	-5.64
2	APDN	2.46	60	147.60	2017-12-06 15:16:00	2.38	142.80	2017-12-06 16:00:00	-4.80
3	VREX	37.62	3	112.86	2017-12-08 15:35:00	37.55	112.65	2017-12-08 16:00:00	-0.21
4	PULM	1.64	91	149.24	2017-12-01 12:32:00	1.63	148.33	2017-12-01 16:00:00	-0.91
5	AROW	36.75	4	147.00	2017-12-06 9:50:00	36.1	144.40	2017-12-06 16:00:00	-2.60

[1]Alpha Vantage Inc. 2017. https://www.alphavantage.co/

6	ACAD	30.38	4	121.52	2017-12-08 10:43:00	30.49	121.96	2017-12-08 16:00:00	0.44
7	WVE	37.25	4	149.00	2017-11-30 15:18:00	37.2	148.80	2017-11-30 16:00:00	-0.20
8	AVXS	100.87	1	100.87	2017-12-08 9:50:00	99.83	99.83	2017-12-08 16:00:00	-1.04
9	MARA	5.059	29	146.71	2017-12-05 14:51:00	5.11	148.19	2017-12-05 16:00:00	1.48
Total				**1,223.32**			**1,209.84**		**-13.48**

Table 6: Backtesting data using strategy 1

Strategy 2

Purchase 1 stock when price above neckline and sell at the end of purchase day or if the price of the stock falls below the highest price recorded. See table S2 for results. We made a profit of $7.33 (0.6%).

	Stock	Purchase Price	# Stocks	Total Purchase Cost	Purchase Date	Sell Price	Total Sell Price	Sell Date	Profit/ Loss
1	ARLZ	1.58	94	148.52	2017-12-01 15:14:00	1.5766	148.20	2017-12-01 15:19:00	-0.32
2	APDN	2.46	60	147.60	2017-12-06 15:16:00	2.45	147.00	2017-12-06 15:32:00	-0.60
3	VREX	37.62	3	112.86	2017-12-08 15:35:00	37.58	112.74	2017-12-08 15:36:00	-0.12
4	PULM	1.64	91	149.24	2017-12-01 12:32:00	1.64	149.24	2017-12-01 14:23:00	0.00
5	AROW	36.75	4	147.00	2017-12-06 9:50:00	36.5	146.00	2017-12-06 14:23:00	-1.00
6	ACAD	30.38	4	121.52	2017-12-08 10:43:00	30.635	122.54	2017-12-08 11:18:00	1.02
7	WVE	37.25	4	149.00	2017-11-30 15:18:00	37.375	149.50	2017-11-30 15:26:00	0.50
8	AVXS	100.87	1	100.87	2017-12-08 9:50:00	101.58	101.58	2017-12-08 12:35:00	0.71
9	MARA	5.059	29	146.71	2017-12-05 14:51:00	5.3051	153.85	2017-12-05 15:11:00	7.14
Total				**1,223.32**			**1,230.65**		7.33

Table 7: Backtesting data using strategy 2

Strategy 3

Purchase stock when price is above neckline, and sell at the end of the test period or if the price of the stock falls below the highest price recorded. See table S3 for results. We made a profit of $32.88 (2.7%).

	Stock	Purchase Price	# Stocks	Total Purchase Cost	Purchase Date	Sell Price	Total Sell Price	Sell Date	Profit/ Loss
1	ARLZ	1.58	94	148.52	2017-12-01 15:14:00	1.58	148.52	2017-12-04 9:55:00	0.00
2	APDN	2.46	60	147.60	2017-12-06 15:16:00	2.5	150.00	2017-12-07 9:30:00	2.40
3	VREX	37.62	3	112.86	2017-12-08 15:35:00	37.58	112.74	2017-12-08 15:36:00	-0.12
4	PULM	1.64	91	149.24	2017-12-01 12:32:00	1.72	156.52	2017-12-04 9:57:00	7.28
5	AROW	36.75	4	147.00	2017-12-06 9:50:00	36.5	146.00	2017-12-06 14:23:00	-1.00
6	ACAD	30.38	4	121.52	2017-12-08 10:43:00	30.635	122.54	2017-12-08 11:18:00	1.02
7	WVE	37.25	4	149.00	2017-11-30 15:18:00	39.7	158.80	2017-12-05 9:30:00	9.80
8	AVXS	100.87	1	100.87	2017-12-08 9:50:00	101.58	101.58	2017-12-08 12:35:00	0.71
9	MARA	5.059	29	146.71	2017-12-05 14:51:00	5.5	159.50	2017-12-06 9:41:00	12.79
Total				**1,223.32**			**1,256.20**		32.88

Table 8: Backtesting data using strategy 3

Improvement

A further improvement to the algorithm would be to use other reversal patterns like inverse triple top. We can also design different trading strategies for different sectors, and that might give better results.

Work Done

Melvin Mathew:	PIP algorithm implementation, rule-based matching, backtesting
Shichun Wang:	Visualization, data collection using Alpha Vantage
Rong Tai Damien Tan:	Genetic algorithm, template-based matching, synthetic tests

References

Alpha Vantage. “Alpha Vantage - Free APIs for Realtime and Historical Financial Data, Technical Analysis, Charting, and More!”, 11 Dec. 2017, www.alphavantage.co/.

Chung, F.-L., et al. “An Evolutionary Approach to Pattern-Based Time Series Segmentation.” *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 5, 2004, pp. 471–489., doi:10.1109/tevc.2004.832863.

F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné. DEAP: Evolutionary Algorithms Made Easy. *Journal of Machine Learning Research*, 13:2171--2175, 2012.

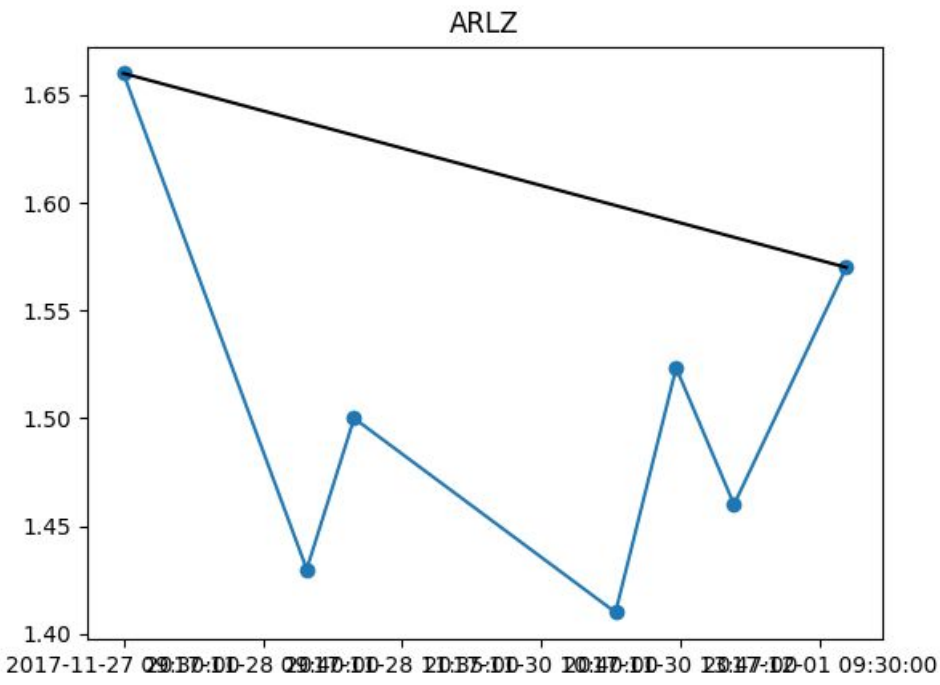
Fu, Tak-Chung, et al. “Stock Time Series Visualization Based on Data Point Importance.” *Engineering Applications of Artificial Intelligence*, vol. 21, no. 8, 2008, pp. 1217–1232., doi:10.1016/j.engappai.2008.01.005.

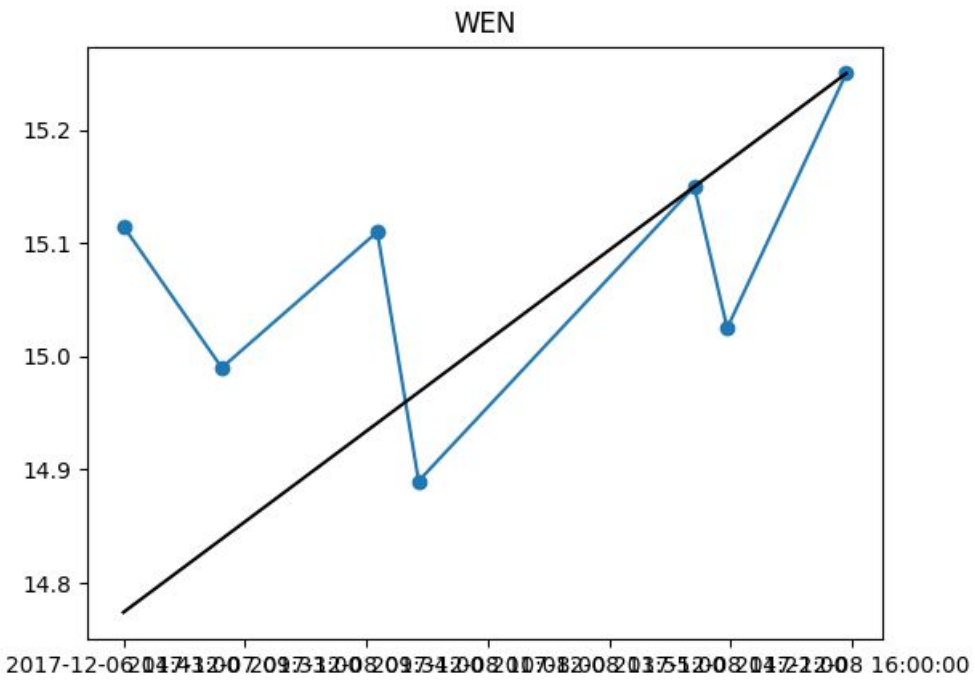
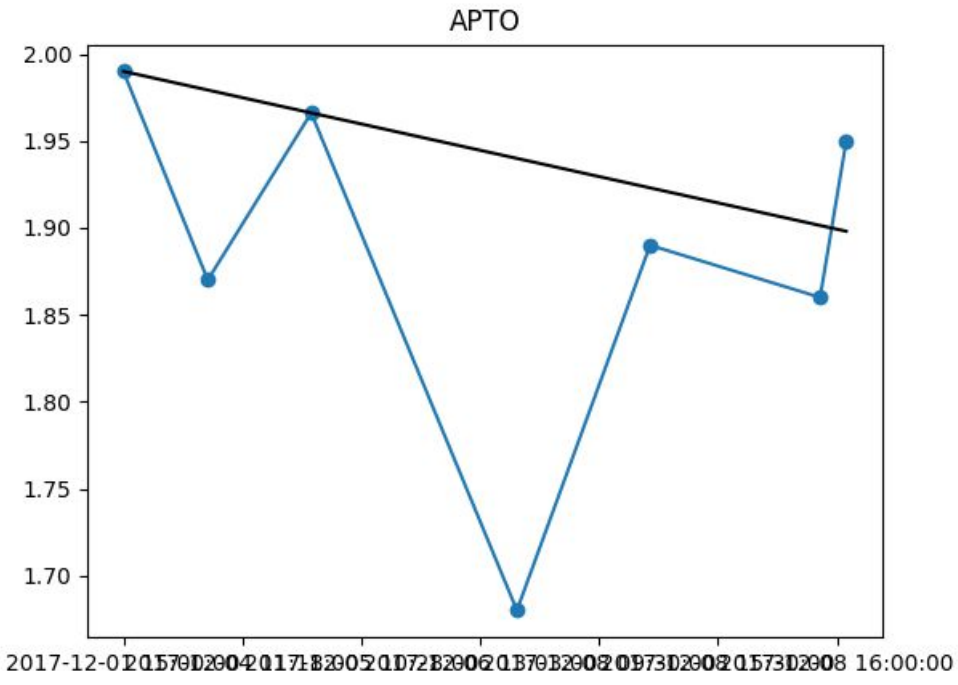
Fu, Tak-Chung, et al. “Stock Time Series Pattern Matching: Template-Based vs. Rule-Based Approaches.” *Engineering Applications of Artificial Intelligence*, vol. 20, no. 3, 2007, pp. 347–364., doi:10.1016/j.engappai.2006.07.003.

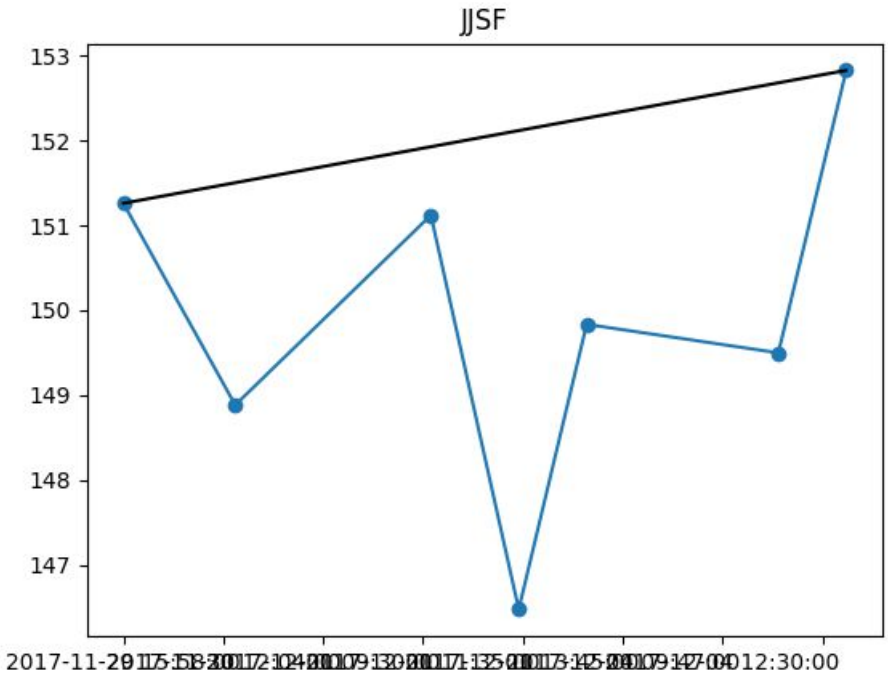
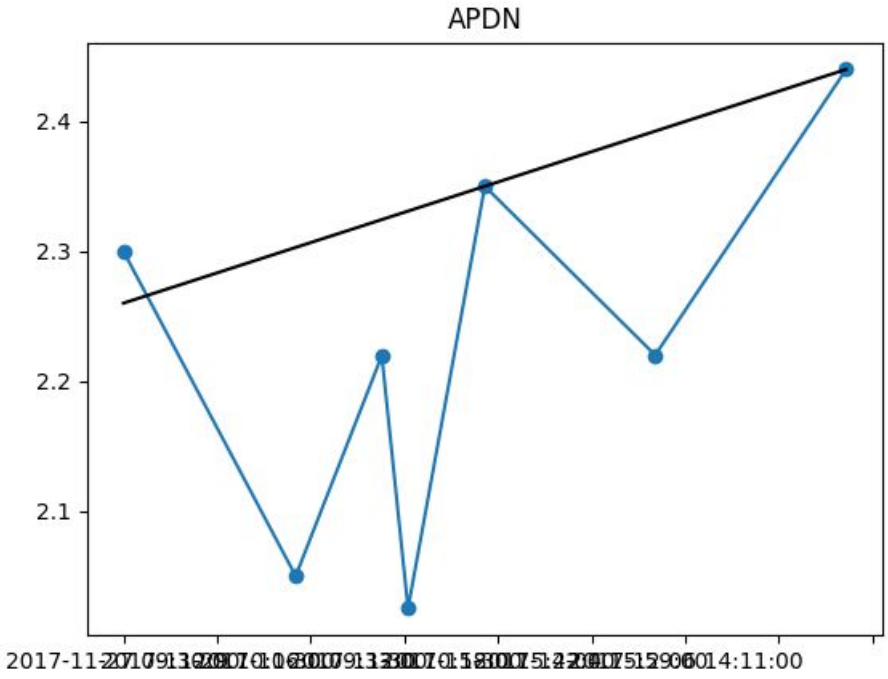
Lo, Andrew, et al. “Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation.” 2000, doi:10.3386/w7613.

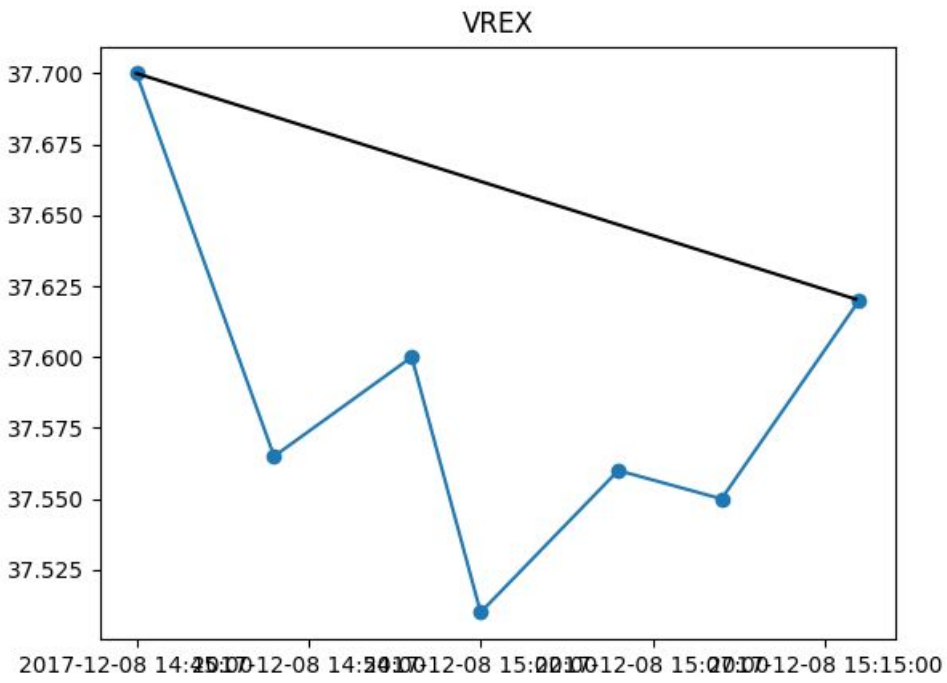
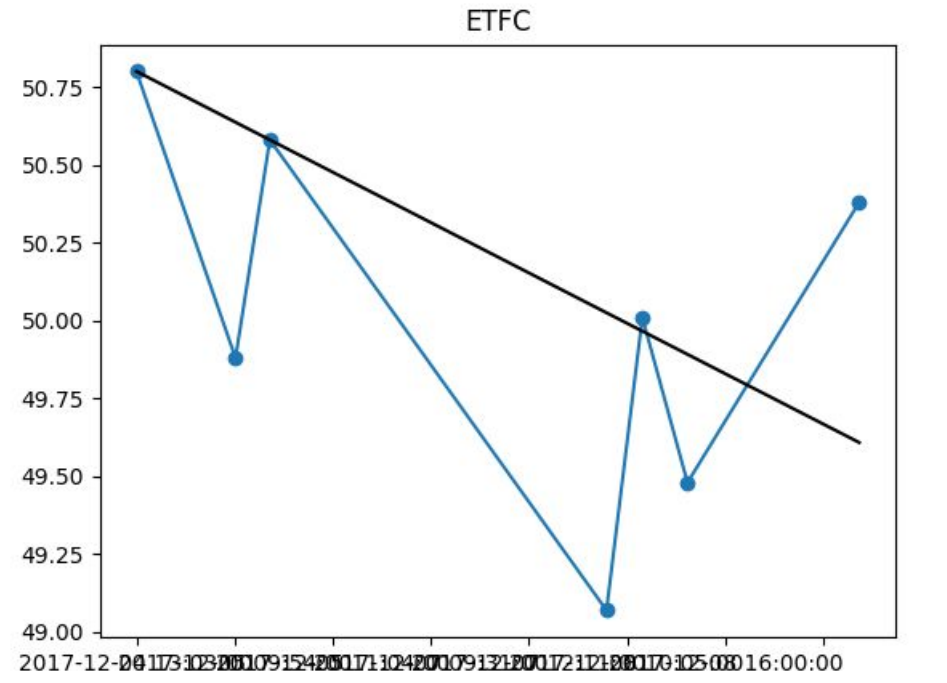
Appendix

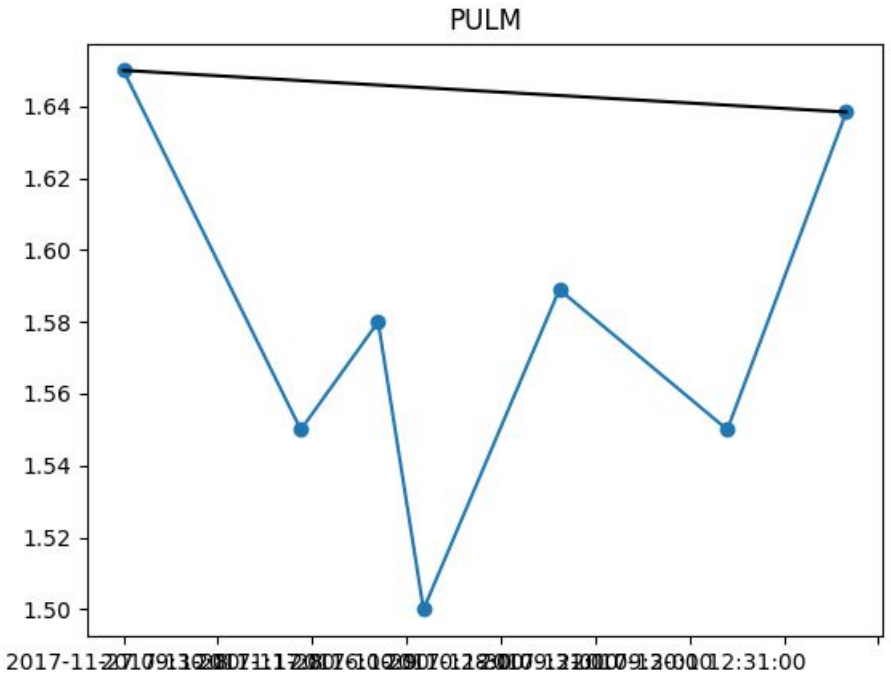
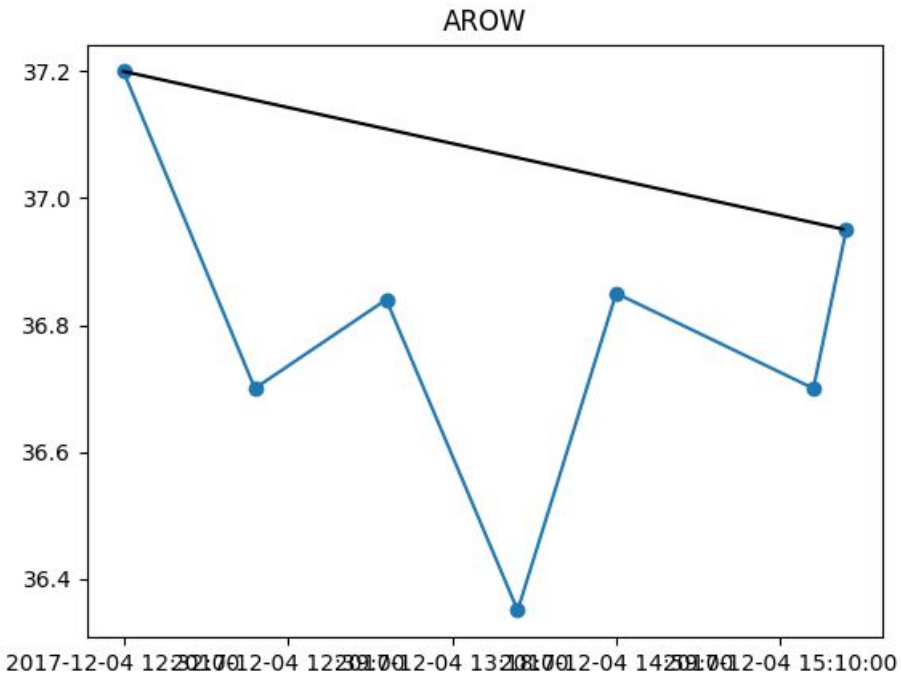
Below are the detected inverse head and shoulder pattern in the 26 stocks out of the 1951 stocks that have been analysed.

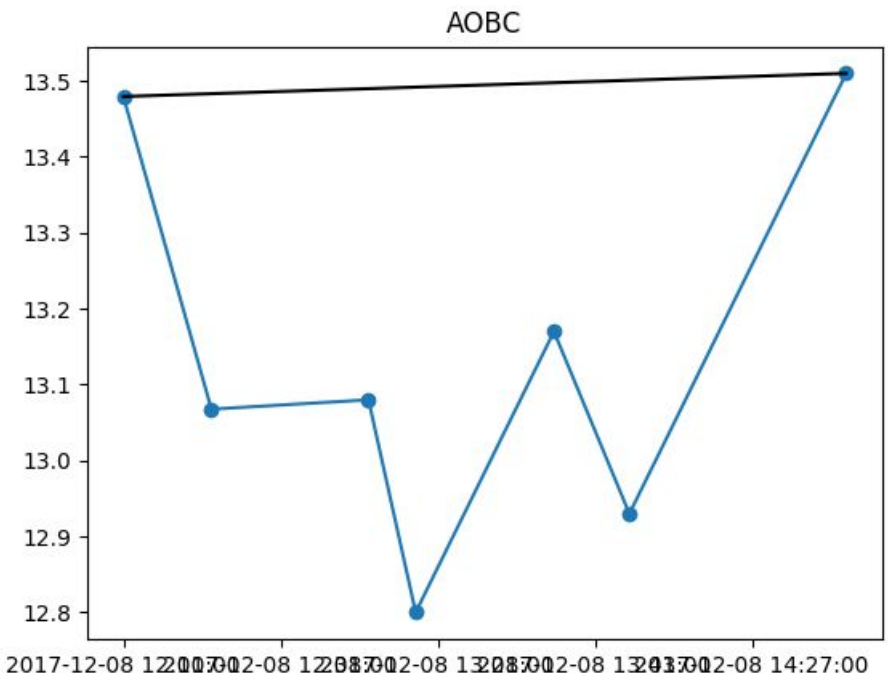
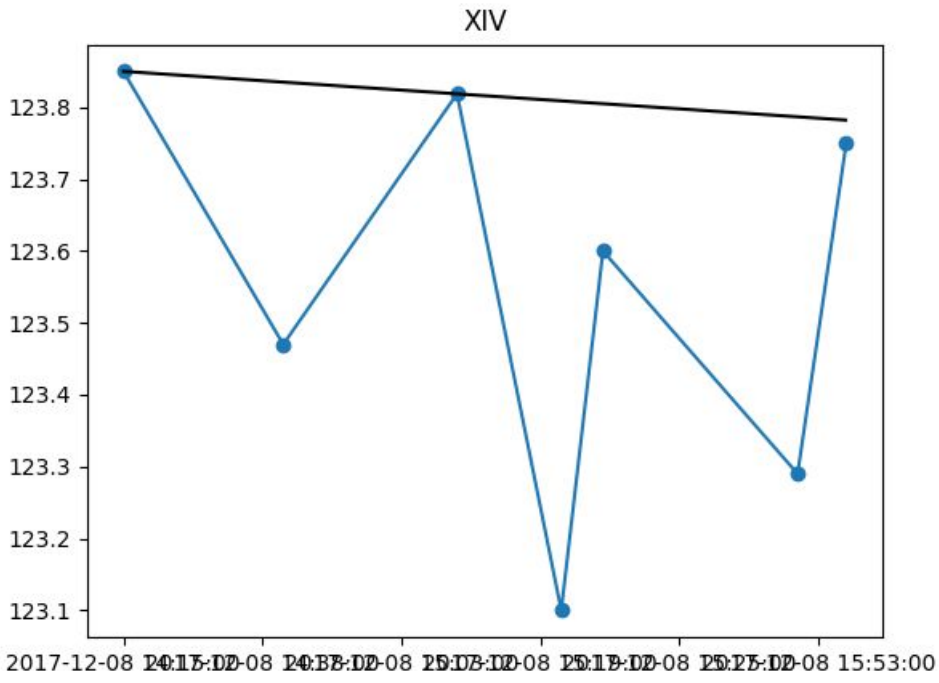
	Stock	**Neckline (in black)**
1	ARLZ	

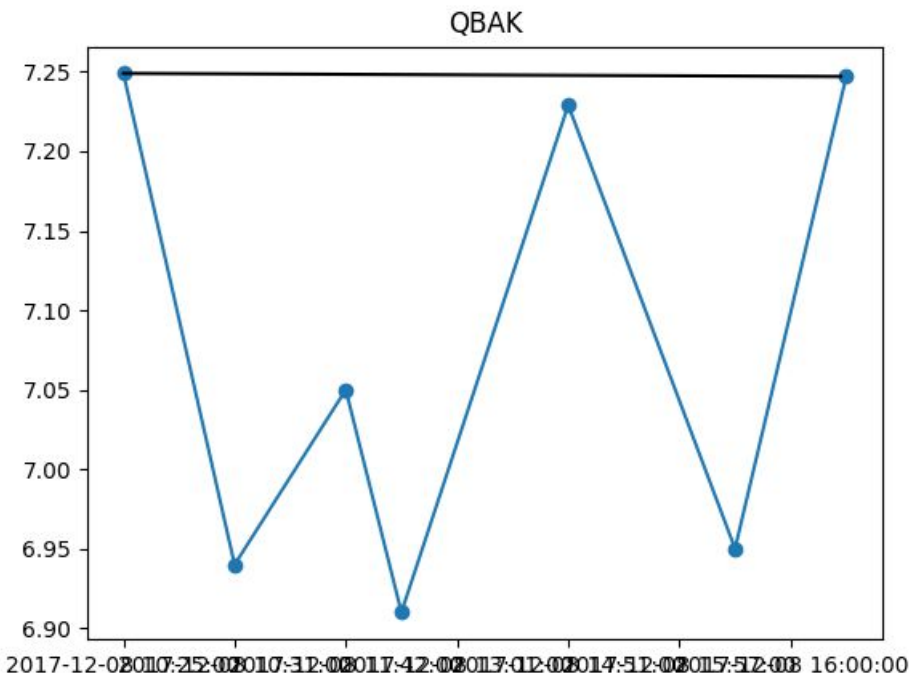
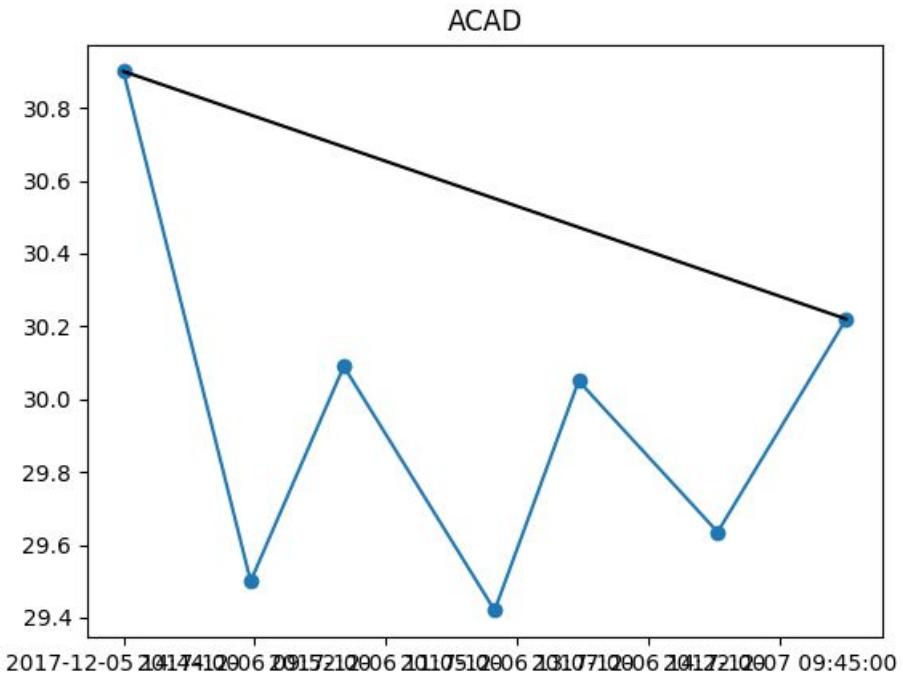
2	WEN	 WEN 15.2 15.1 15.0 14.9 14.8 2017-12-06 16:00:00
3	APTO	 APTO 2.00 1.95 1.90 1.85 1.80 1.75 1.70 2017-12-01 16:00:00

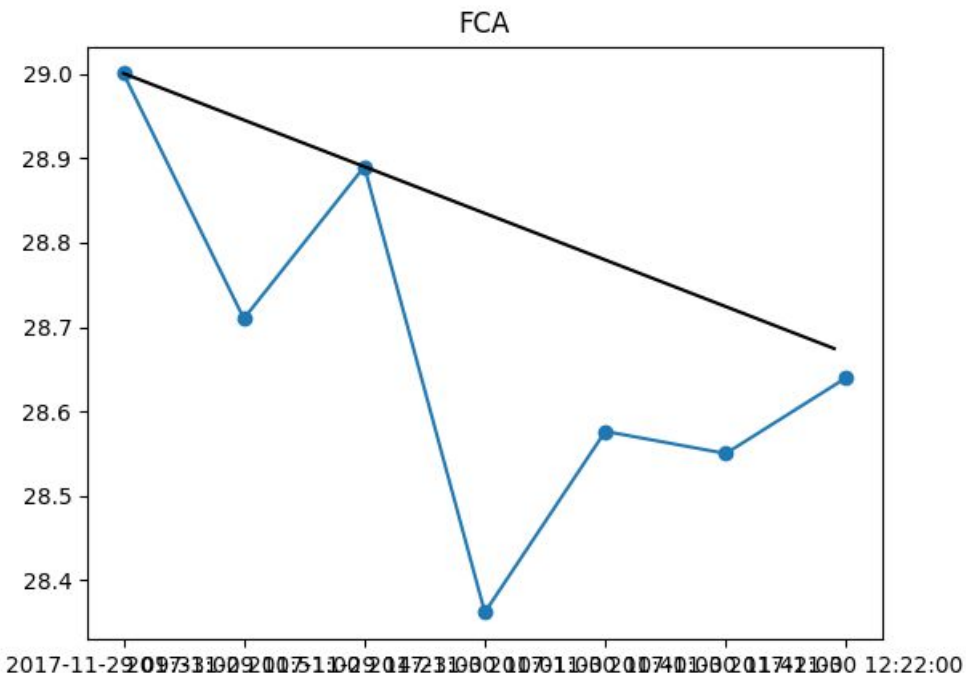
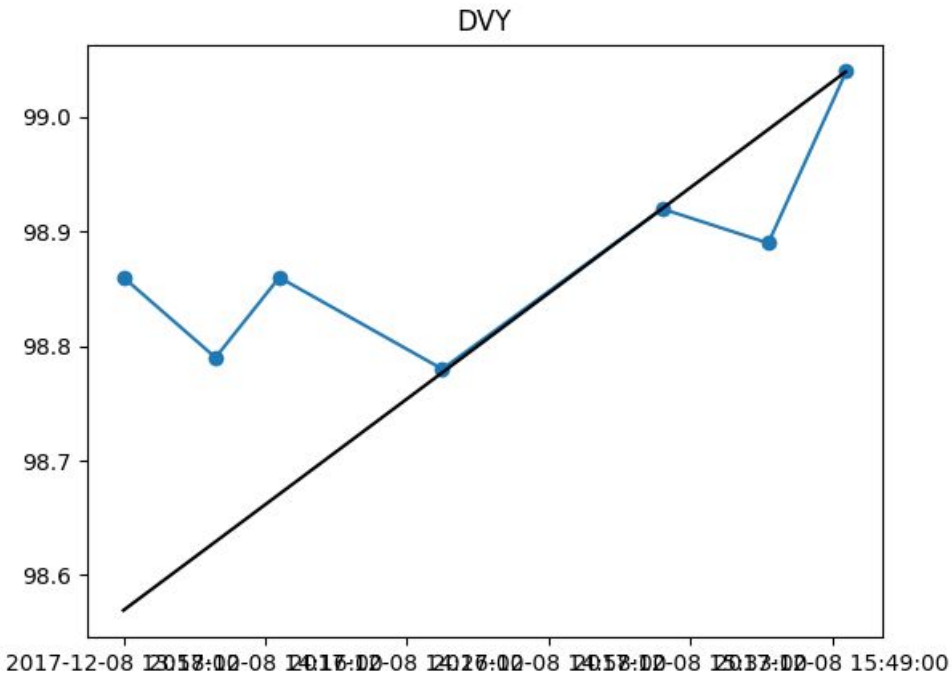
4	JJSF	 JJSF 153 152 151 150 149 148 147 12:30:00
5	APDN	 APDN 2.4 2.3 2.2 2.1 14:11:00

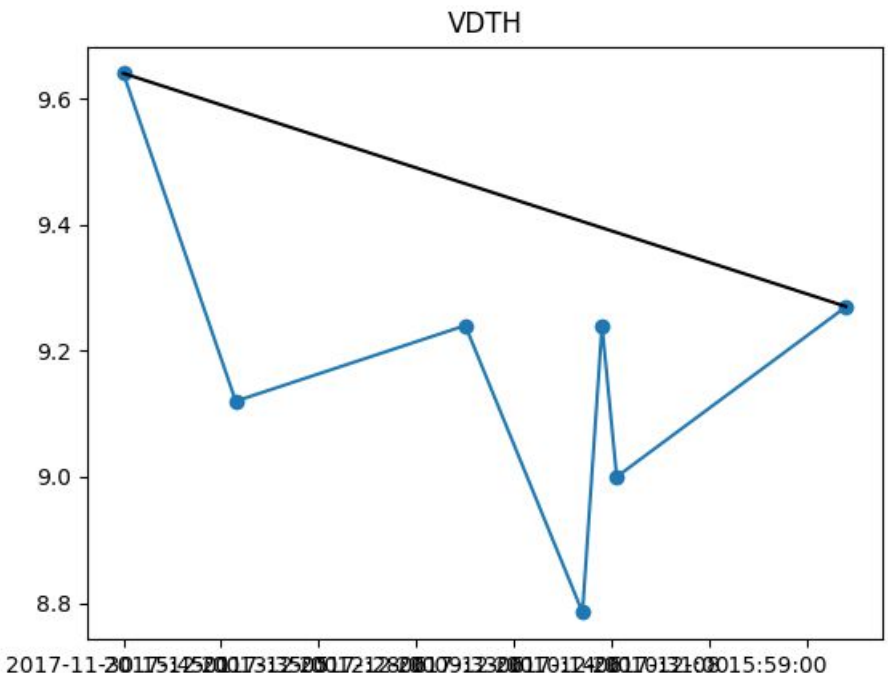
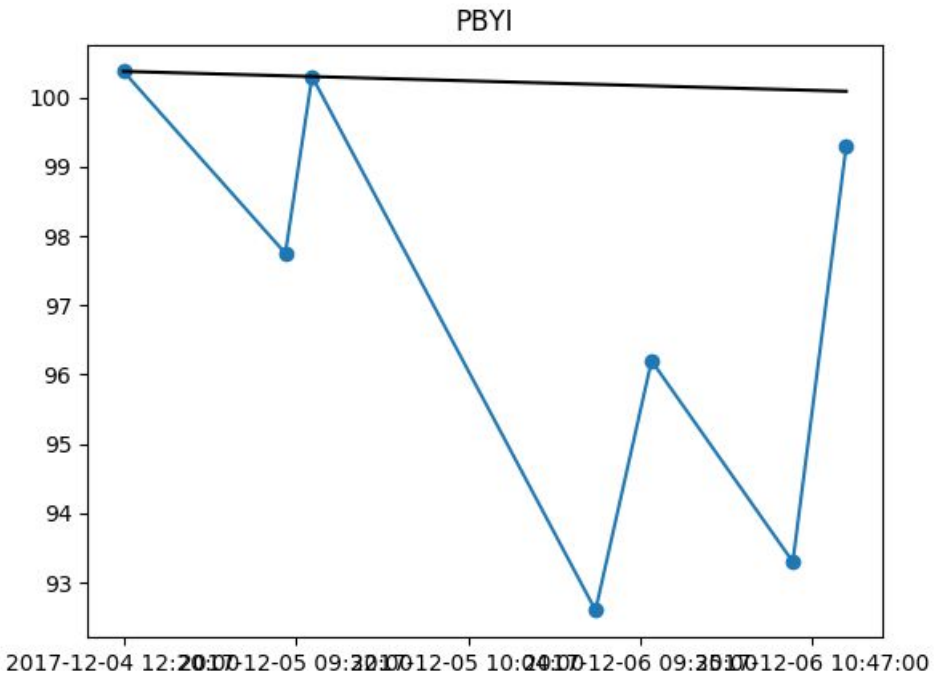
6	VREX	 VREX 37.700 37.675 37.650 37.625 37.600 37.575 37.550 37.525 2017-12-08 15:15:00
7	ETFC	 ETFC 50.75 50.50 50.25 50.00 49.75 49.50 49.25 49.00 16:00:00

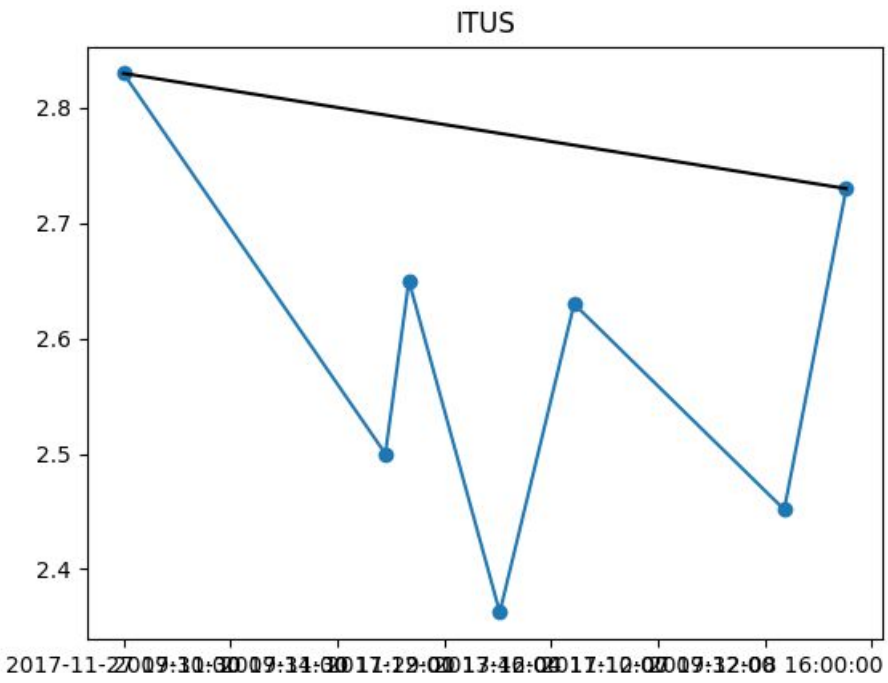
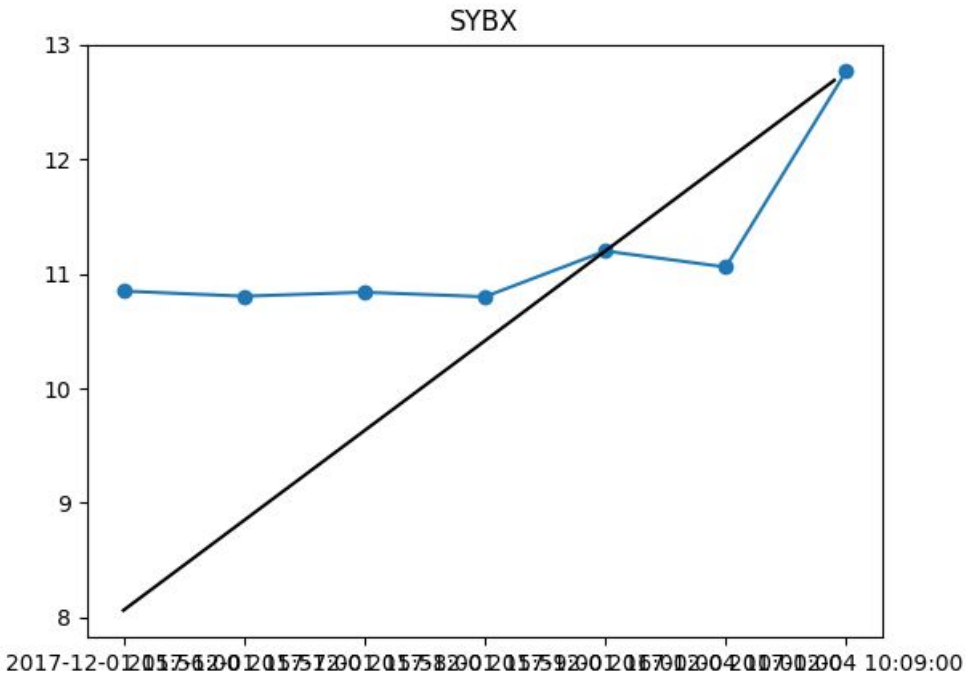
8	PULM	 PULM 1.64 1.62 1.60 1.58 1.56 1.54 1.52 1.50 12:31:00
9	AROW	 AROW 37.2 37.0 36.8 36.6 36.4 15:10:00

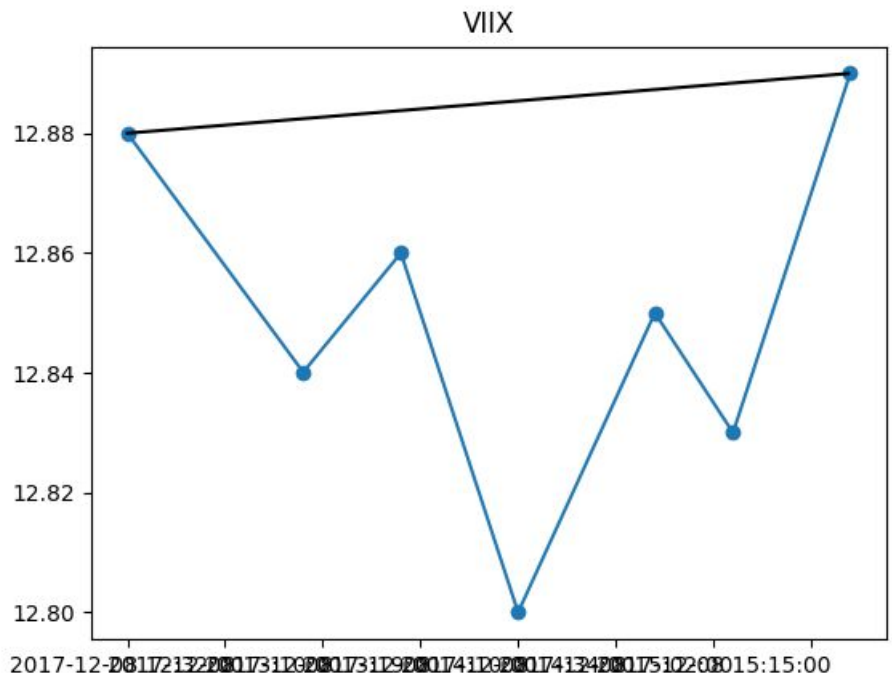
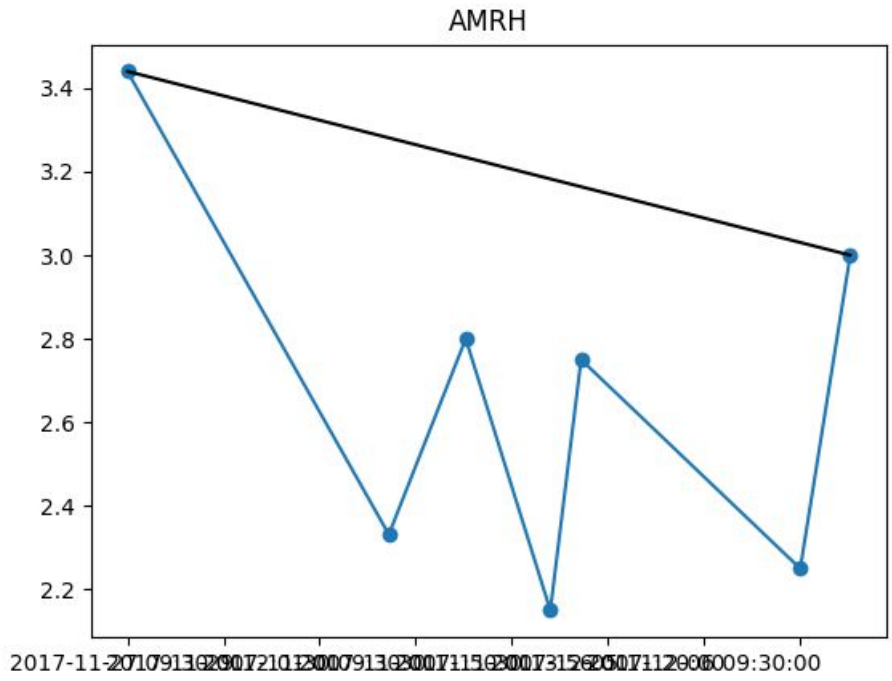
10	AOBC	
11	XIV	

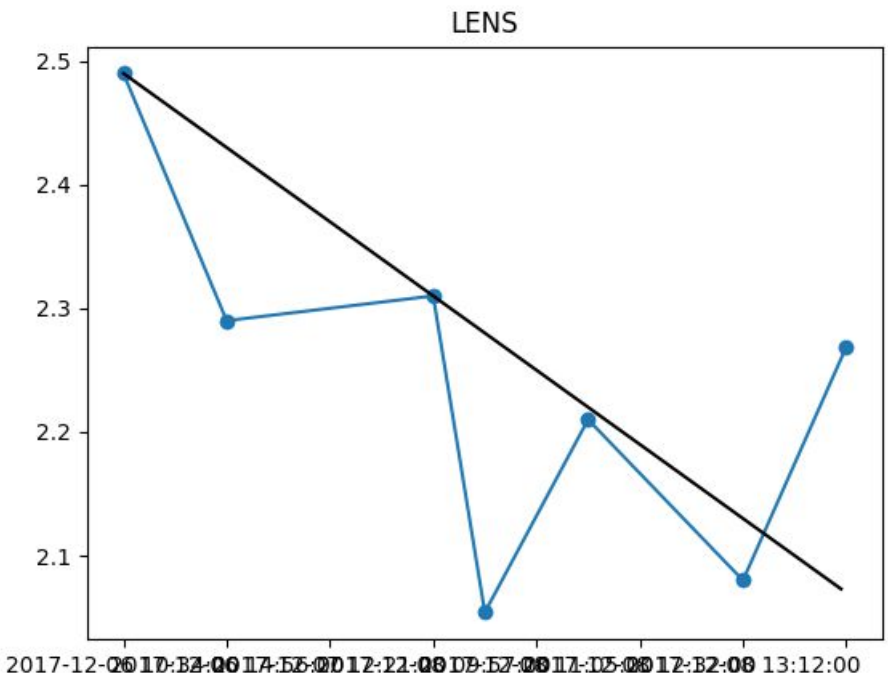
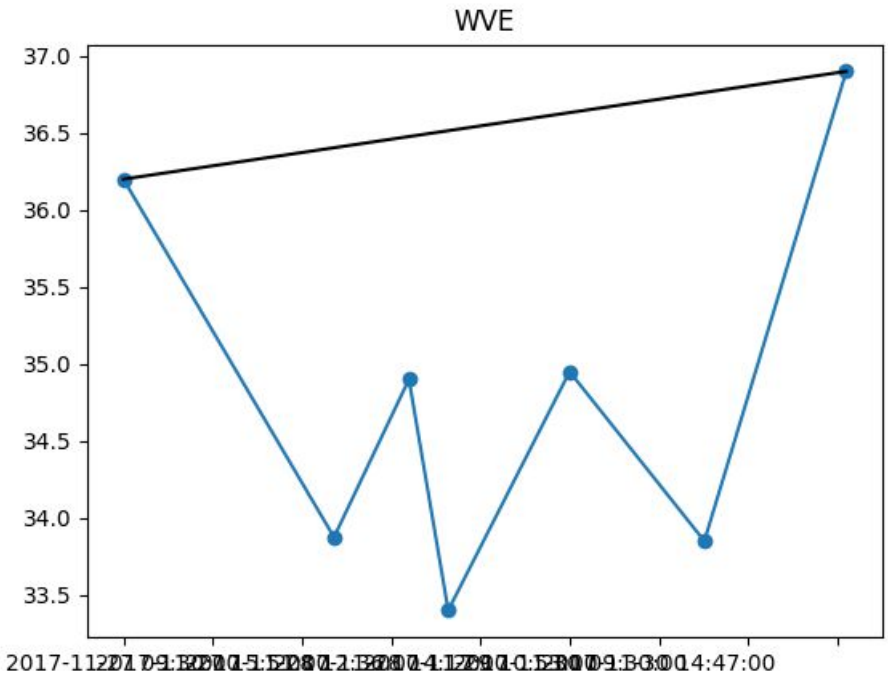
12	QBAK	
13	ACAD	

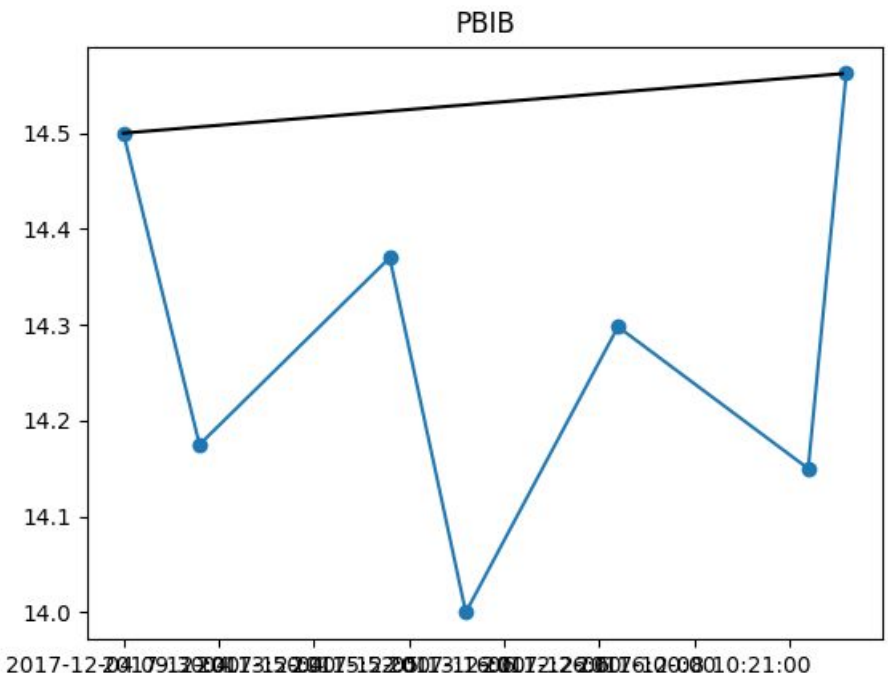
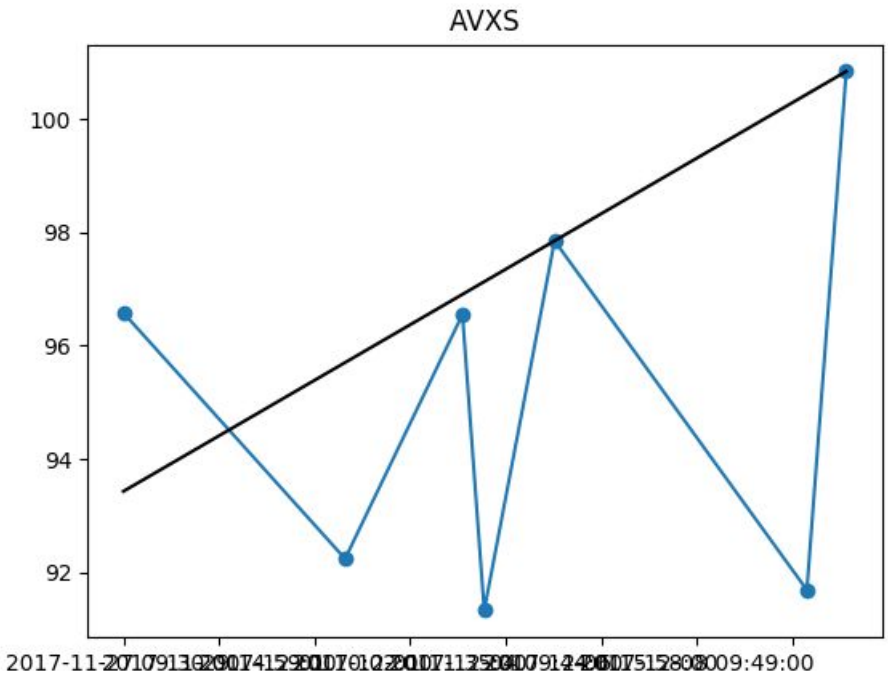
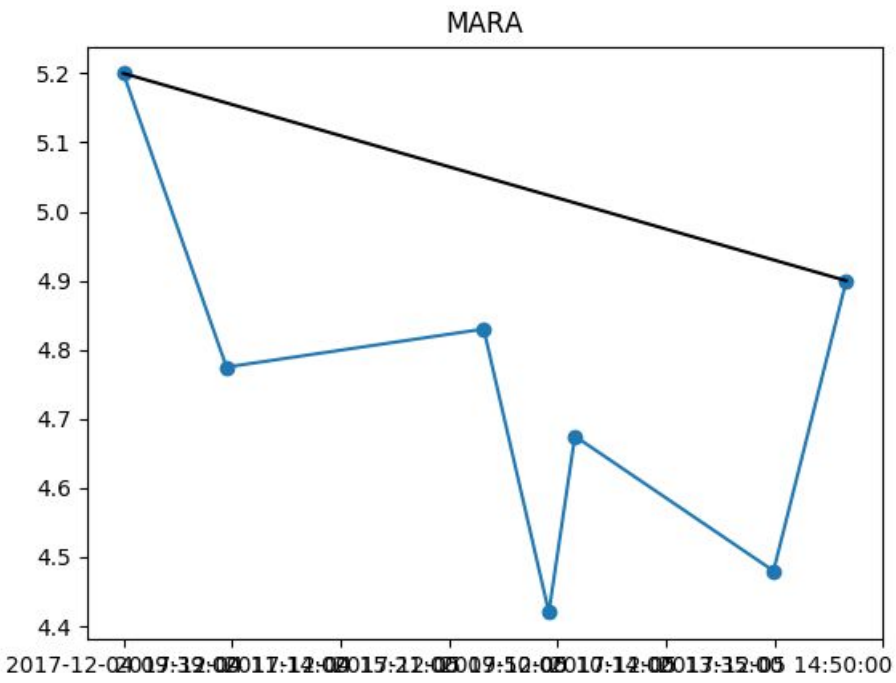
14	FCA	 FCA 29.0 28.9 28.8 28.7 28.6 28.5 28.4 12:22:00
15	DVY	 DVY 99.0 98.9 98.8 98.7 98.6 15:49:00

16	VDTH	
17	PBYI	

18	ITUS	
19	SYBX	

20	VIIX	
21	AMRH	

22	LENS	
23	WVE	

24	PBIB	
25	AVXS	

26	MARA