Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Detección de patrones chartistas basado en dynamic time warping y redes convolucionales

Detection of chartist patterns based on dynamic time warping and convolutional networks

José Antonio Antúnez Pulido

La Laguna, 17 de junio de 2024

D. **Cándido Caballero Gil**, profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor.

D. **Javier Giner Rubio**, profesor Titular de Universidad adscrito al Departamento de Economía, Contabilidad y Finanzas de la Universidad de La Laguna, como cotutor.

C E R T I F I C A (N)

Que la presente memoria titulada:

”*Detección de patrones chartistas basado en dynamic time warping y redes convolucionales*”

ha sido realizada bajo su dirección por D. **José Antonio Antúnez Pulido**.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 17 de junio de 2024

Agradecimientos

Me gustaría agradecer a mis tutores Cándido y Javier por su trabajo. También agradecerle a mi familia por cuidarme todos estos años, especialmente a mi madre, sin su apoyo no estaría donde estoy hoy.

Licencia

Resumen

En este trabajo se implementarán tecnologías de redes neuronales convolucionales y de dynamic time warping en PerseumAI, un programa desarrollado por exalumnos de la ULL capaz de detectar patrones del análisis chartista, y se medirá su desempeño en la detección de patrones. Las redes neuronales convolucionales (CNN) son un tipo de red neuronal muy efectiva en tareas de reconocimiento y clasificación de imágenes. Utilizan filtros para extraer y aprender características relevantes de los datos. Por otro lado, el algoritmo Dynamic Time Warping (DTW) mide la similitud entre dos series temporales que pueden tener diferentes velocidades o longitudes. Para entrenar la red convolucional se creará una base de datos con imágenes de patrones chartistas, apoyándonos en un script de creación de patrones sintéticos, para posteriormente implementarla al programa. Mientras que para el algoritmo DTW usaremos una base de datos con patrones en formato CSV de modo que podamos utilizar el algoritmo para comparar dichos patrones con la gráfica de precios que estemos estudiando y así detectar los patrones. Además, se realizarán varios estudios intentando optimizar al máximo PerseumAI, mejorando también su interfaz a una más moderna y sencilla de usar. Todo esto con el objetivo de conseguir una herramienta lo más fiable y útil posible, que sirva de apoyo a los analistas técnicos en el estudio del mercado.

Palabras clave: DTW, redes neuronales convolucionales, análisis técnico, bolsa de valores, patrones chartistas

Abstract

In this work, convolutional neural network technologies and dynamic time warping have been implemented in PerseumAI, a program developed by former ULL students capable of detecting chart pattern analysis, and to measure the performance in pattern detection. Convolutional neural networks (CNN) are a type of neural network that is very effective in image recognition and classification tasks. They use filters to extract and learn relevant features from data. On the other hand, the Dynamic Time Warping (DTW) algorithm measures the similarity between two-time series that may have different speeds or lengths. In order to train the convolutional network, a database with images of chart patterns has been created, relying on a synthetic pattern creation script, to later implement it into the program. Meanwhile, for the DTW algorithm, we will use a database with patterns in CSV format so that we can use the algorithm to compare these patterns with the price chart we are studying and thus detect the patterns. Additionally, several studies will be conducted to optimize PerseumAI to the fullest, also improving its interface to a more modern and user-friendly one. All of this is aimed at creating the most reliable and useful tool possible, to support technical analysts in market studies.

Keywords: DTW, convolutional neural network, technical analysis, stock market, chartist patterns

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

1.1. Motivo del proyecto

El presente trabajo está diseñado como una evolución de los esfuerzos previos en el desarrollo de una herramienta para la detección de patrones chartistas, PerseumAI. En una era donde la capacidad de procesamiento y análisis de datos crece exponencialmente, este trabajo busca capitalizar estos avances para proporcionar una herramienta sofisticada que asista en la toma de decisiones a inversores de todos los tipos. El mercado de valores, conocido por su complejidad y dinamismo, presenta desafíos continuos para las personas que operan en él. La habilidad para identificar patrones y tendencias en los datos históricos puede ser de gran ventaja para los inversores ya que uno de los principios del análisis técnico es que los movimientos del mercado son de caracter cíclico, por lo que conocer el pasado nos puede ayudar a prevenir el futuro, además la herramienta no solo proporcionaría información histórica si no también patrones en formación. Ante este escenario, este trabajo busca mejorar y ampliar las capacidades de PerseumAI. El proyecto no solo aborda el aspecto técnico de la implementación de algoritmos en Python (Python Software Foundation (2023)) y el manejo de grandes volúmenes de datos a través de bibliotecas especializadas como 'Pandas' (The pandas development team (2023)), sino que también se enfoca en la interpretación y presentación de los resultados de una manera accesible y útil para el usuario final. En definitiva, este trabajo pretende ser un apoyo a inversores que deseen añadir el análisis técnico a sus herramientas a la hora de operar.

1.2. Objetivos

El enfoque del trabajo está dirigido a continuar la evolución de PerseumAI, creada originalmente por Gabriel García en 2022 y mejorada por Aitor Alonso el curso anterior en 2023. Este año se pretende lo siguiente:

- Mejorar la detección de patrones históricos.
- Mejorar la detección de patrones actuales.
- Utilizar una red neuronal para la detección de patrones.
- Mejorar la interfaz.

1.3. Fases del proyecto

Las fases del proyecto son las siguientes:

- Investigación sobre el análisis técnico.
- Investigación sobre las tecnologías usadas en el proyecto (python, DTW, redes neuronales...).
- Leer y comprender el código para su posterior modificación.
- Mejora de la base de datos del programa.
- Experimentos para determinar los mejores parámetros en la búsqueda de patrones históricos y actuales en cuanto a tiempo y cantidad de patrones encontrados.
- Experimentos para determinar los mejores parámetros en cuanto a calidad de los patrones encontrados.
- Desarrollo de un nuevo algoritmo para la búsqueda de patrones históricos.
- Desarrollo de un nuevo algorimo para la búsqueda de patrones actuales.
- Experimento para determinar si las mejoras han sido satisfactorias.

- Desarrollo y entrenamiento de una red neuronal para la detección de patrones chartistas.
- Implementar la red neuronal en el programa.
- Experimento para comparar la búsqueda de patrones original mejorada con la búsqueda usando redes neuronales.
- Desarrollo de la aplicación final.

Capítulo 2

Contexto

2.1. Conceptos relevantes

En este capítulo se pretende estudiar diferentes definiciones y conceptos que hay que conocer para poder entender el proyecto, como qué es el análisis técnico, en qué consiste el algoritmo Dynamic Time Warping (DTW) y demás.

2.1.1. Análisis técnico versus análisis fundamental

Cuando hablamos de maneras en las que podemos estudiar el mercado de valores tenemos dos grandes escuelas de pensamiento con diferentes filosofías, el análisis fundamental y el análisis técnico. El análisis fundamental consiste en estudiar datos financieros y económicos con el objetivo de predecir si una empresa logrará más beneficios y por tanto subirá su valor, el inversor fundamentalista se sumerge en los informes de auditoría, estados de pérdidas y ganancias, balances trimestrales, registros de dividendos, políticas de las compañías en su radar, indicadores macroeconómicos y demás fuentes de información. Todo esto para intentar evaluar el valor “verdadero” o “intrínseco” de las acciones de una empresa a partir del rendimiento futuro que se espere de ellas, Martínez and Gallegos (1999).

Por otro lado, el análisis técnico, se refiere al estudio de una acción basado en el mercado en sí mismo frente al estudio de los bienes en los que se negocia el mercado como ocurre en el análisis fundamental (Achelis (2001)). Esto quiere decir que para nosotros realizar un análisis sobre si debemos o no invertir en una determinada acción solamente deberemos estudiar los movimien-

tos del mercado (Murphy (1999)), observando el comportamiento de variables como precio, volúmen y el interés abierto (este último se usa solo en futuros y opciones).

Este estudio del mercado se realiza mediante diferentes métodos como el análisis de indicadores técnicos o el uso de líneas de tendencia con el objetivo de predecir la tendencia futura del mercado. Dentro del análisis técnico existe el análisis chartista, que estudia la forma de los gráficos de precios para predecir la tendencia futura. Nuestro trabajo fin de grado se basa en estas técnicas, intentando la detección de patrones repetitivos, estructuras gráficas que nos permiten conocer con cierto grado de certeza los precios futuros.

Existe una gran controversia en la literatura científica sobre la validez del análisis técnico y en concreto de los patrones visuales. Farias et al. (2017) analiza 85 investigaciones científicas relevantes sobre análisis técnico para analizar las ventajas y desventajas de estas técnicas.El estudio refleja que la mayor parte de estudios encontrados se centran en métodos de análisis técnico que pueden expresarse de forma algebraica, tales como medias móviles o filtros de diferentes tipos. Sin embargo, los inversores utilizan un amplio abanico de técnicas basadas en el reconocimiento de patrones visuales. El análisis chartista se centra en la identificación de repeticiones periódicas en las series temporales, patrones no lineales que pueden ayudarnos a anticiparnos al movimiento de los mercados.

Según el estudio de Farias et al. (2017), el análisis chartista sólo fue utilizado en 3 de los 85 artículos relevantes analizados. Por ejemplo, Dawson and Steeley (2003) analiza la existencia de patrones repetitivos en el mercado de acciones de Reino Unido. De una forma más general, Wang and Chan (2007) y Friesen et al. (2009) también dedican sus esfuerzos al estudio de las técnicas chartistas.

Se podría decir que es necesaria más investigación en el campo del estudio de la existencia de patrones, pero las dificultades técnicas son evidentes. De alguna manera, el análisis chartista recae en la pericia y experiencia de los analistas que, de forma subjetiva, creen ser capaces de identificar estos patrones. Por eso

la utilidad de este trabajo fin de grado, intentar aportar algo de rigurosidad al estudio del análisis chartista.

2.1.2. Inteligencia Artificial, DTW y Redes Neuronales

La Inteligencia Artificial (IA) se refiere a la capacidad de las máquinas para simular procesos de inteligencia humana, como el aprendizaje, el razonamiento y la percepción. Se basa en algoritmos y modelos que permiten a las máquinas realizar tareas que normalmente requerirían inteligencia humana. En el contexto de este proyecto utilizaremos redes neuronales convolucionales y el algoritmo Dynamic Time Warping para el reconocimiento de los patrones del análisis técnico.

Las redes neuronales convolucionales son un tipo de red neuronal artificial usadas en el procesamiento de imágenes. Estas redes están compuestas por capas de neuronas que aprenden automáticamente características relevantes de los datos de entrada mediante el uso de filtros convolucionales, O'shea and Nash (2015).

El algoritmo DTW es una técnica utilizada para medir la similitud entre dos series temporales, incluso cuando tienen longitudes diferentes o se han desplazado en el tiempo. Funciona encontrando la correspondencia óptima entre puntos de las dos secuencias, minimizando una medida de distancia entre ellas Müller (2007). En el contexto financiero, DTW puede ser útil para comparar series temporales de precios de acciones y encontrar patrones similares a pesar de las variaciones en el tiempo. En nuestro caso en específico, DTW devuelve una distancia que representa cuanto se parecen las gráficas que se comparan, a menor distancia, más se parecen.

2.2. Antecedentes y estado actual del tema

La búsqueda de una herramienta informática capaz de predecir los precios del mercado de forma precisa no es nueva y existe desde que la potencia de las computadoras lo permite. Los estudios más antiguos relacionados con el reconocimiento de patrones

usando programas informáticos datan de la década de los noventa del siglo pasado, por ejemplo tenemos un estudio publicado en 1998 en el que utilizan una red neuronal multicapa de alimentación hacia delante para predecir los precios de la bolsa en un plazo de 4 días, De Villiers (1998).

El paso del tiempo y el aumento de la potencia junto con el auge de la Inteligencia Artificial han permitido estudios como el realizado en 2018 por Marc Velay y Fabrice Daniel, Velay and Daniel (2018), en el cual se comparan redes convolucionales, redes LSTM y detección basada en código, algo bastante parecido a lo que se hará en este TFG.

Por otro lado, el algoritmo de dynamic time warping se lleva usando desde la década de los ochenta para temas relacionados con el reconocimiento de voz y escritura. Por ejemplo, Myers and Rabiner (1981) estudian diferentes variaciones del algoritmo para el reconocimiento de palabras en un texto.

Capítulo 3

Desarrollo del proyecto

3.1. Estado previo de PerseumAI

PerseumAI es una herramienta creada originalmente por Gabriel García (García Jaubert (2022)) y mejorada por Aitor Alonso (Alonso Melián (2023)) que detecta mediante el algoritmo Dynamic Time Warping, los patrones doble techo, doble suelo, triángulo ascendente, triángulo descendente, hombro cabeza hombro y hombro cabeza hombro invertido para un periodo de fechas determinado en las empresas que se quiera, y además calcula el porcentaje de cumplimiento de los patrones encontrados.

Todo el funcionamiento que se explique en este apartado, es referente a la herramienta que dejó Aitor Alonso, sin ninguna modificación.

3.1.1. Parámetros del programa

Para funcionar, el programa necesita una serie de parámetros introducidos por el usuario y ciertos parámetros establecidos en el código. Los parámetros introducidos por el usuario son (Figura 3.1):

1. Patrón a buscar: Hay unos checkbox que permiten seleccionar los patrones que se desea analizar.

2. Tamaño de ventana: Este parámetro determina el tamaño de los subconjuntos de datos que se analizarán en cada paso al buscar patrones en la gráfica de precios de cierre de una empresa. Debido a que no es factible explorar toda la gráfica

de una vez, el tamaño de ventana define cuántos puntos de datos se incluirán en cada subconjunto que se examina.

3. Fecha inicial: Se tiene que especificar una fecha desde la cual partir para buscar los patrones.

4. Fecha final: fecha en la que finaliza la búsqueda.

5. Tickers de las empresas a analizar: El botón 'Open File' abre una ventana para seleccionar un fichero de texto que contenga los tickers de las empresas a analizar. El formato debe ser un archivo TXT que tenga los tickers de las empresas separados por saltos de líneas. Se pueden colocar comas o no al final de cada línea, ya que la herramienta detecta los tickers de igual manera.

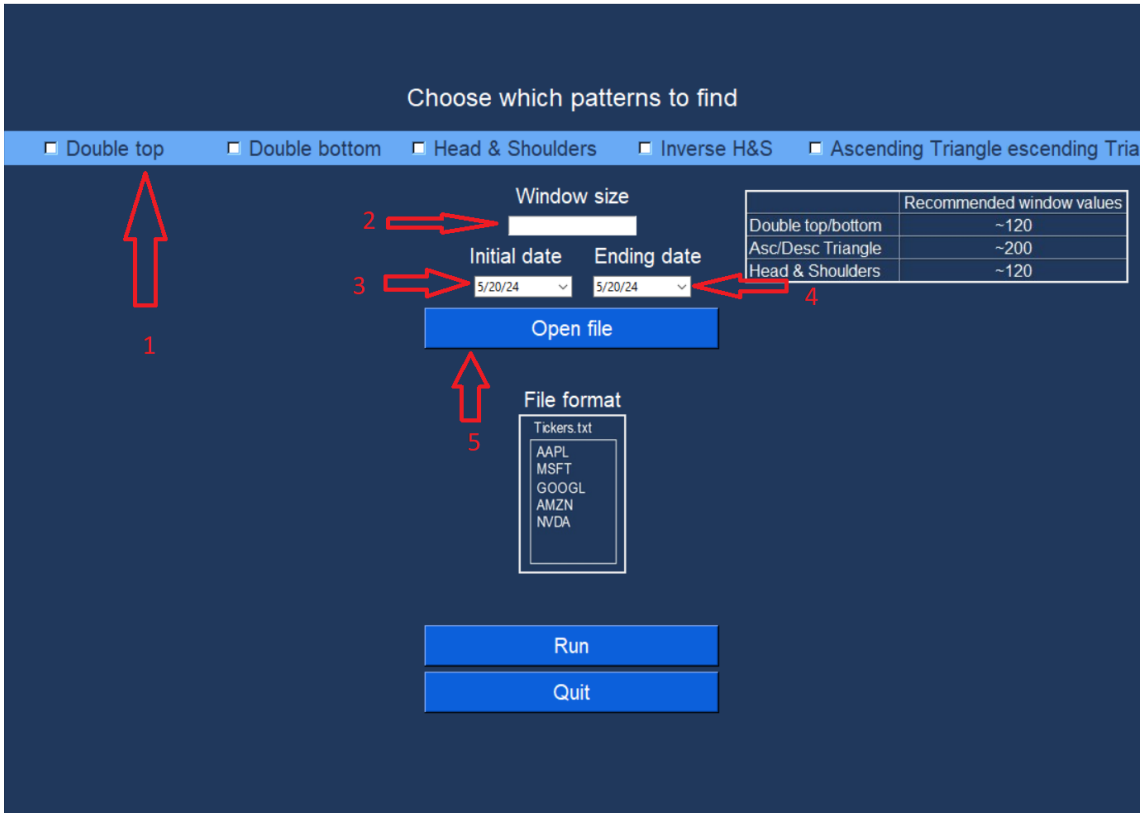


Figura 3.1: Interfaz del programa en la versión anterior

A partir de ahora cuando hablemos de "ventana", nos referiremos a un subconjunto de los datos de precio original cuya longitud es el tamaño de ventana introducido por el usuario. Los parámetros que no son accesibles al usuario son:

1. Divisiones: Cuando en una ventana se detecta un patrón esta se divide en subventanas para encontrar en qué parte de la ventana se encuentra el patrón, las divisiones es una lista de números enteros que indica en cuántas subventanas se va a dividir, por ejemplo, divisiones $[1, 2, 3, 4]$ significa que la ventana original primero se va a dividir por 1, luego en 2, luego en 3 y finalmente en 4 partes. Por ejemplo si estamos explorando una ventana de tamaño 120 con las divisiones anteriores primero dividiríamos la ventana en 1, es decir la ventana completa, luego en dos obteniendo dos subventanas de 60 puntos y así sucesivamente.

2. Número de patrones a cargar: Cuántos patrones de ejemplo se van a cargar de la base de datos para posteriormente comparar las ventanas con dichos patrones usando DTW, por defecto 15.

3. Incremento: Si en una ventana no se encuentra ningún patrón, es el valor que se le suma al índice izquierdo de la ventana actual para determinar la siguiente ventana, por defecto 25.

4. Periodicidad de la media móvil: Antes de explorar los patrones se le aplica la media móvil a la gráfica de precios, con una periodicidad de 3.

5. Distancia de aceptación de los patrones: Distancia mínima para aceptar una ventana como patrón y explorarla en busca de dicho patrón usando las divisiones.

6. Tamaño mínimo de ventana, por defecto 80.

3.1.2. Funcionamiento del programa

Búsqueda de patrones históricos

El funcionamiento, de manera resumida, una vez seleccionados los parámetros de la búsqueda de patrones históricos es el siguiente:

1. Se descargan los datos de la empresa y se iteran los datos por ventanas.

2. Si en una ventana detecta un patrón se divide la ventana en subventanas para buscar si alguna tiene mejor similitud con el patrón que la ventana original.
3. Una vez seleccionada la mejor candidata, se pasa un filtro para comprobar si cumple con las características del tipo de patrón en cuestión y si las cumpliera calcula las líneas de soporte, resistencia y la tendencia. Por ejemplo, si se tratará de un doble techo o doble suelo, el filtro detectaría los dos picos máximos o mínimos, y comprobaría si son similares, en cuyo caso se confirma que se ha encontrado el patrón.
4. Si el patrón se ha confirmado se guarda para ser mostrado más adelante, cuando todos los datos hayan sido analizados.
5. Se escoge la siguiente ventana de datos, y se aplican de nuevo los pasos 2-4.
6. Una vez se ha finalizado el análisis de todos los datos, se muestran los patrones encontrados a través de la interfaz, además de

Búsqueda de patrones actuales

La búsqueda de patrones actuales funciona de la siguiente manera:

1. Se descargan los precios de cierre tomando como fecha inicial el día actual menos el tamaño de ventana como días y como fecha final la fecha actual.
2. Se realiza la comparación usando DTW y si la distancia es menor a 30 se pasa la ventana por el filtro.
3. Si el filtro acepta el patrón se guarda para ser mostrado más adelante y se repiten los pasos 1 y 2 con la siguiente empresa.

3.2. Guardar patrones como imágenes

Se añadió una nueva funcionalidad para que si el usuario lo desea se guarden los patrones encontrados como imágenes. Estos se guardan en la carpeta saved_patterns.

3.3. Mejoras en la base de datos

La base de datos no es más que un directorio que contiene para cada tipo de patrón ejemplos en formato csv de dicho patrón, lo que se hizo fue usando un script que te permite visualizar las gráficas en un directorio, observar las instancias que tenían peor calidad para posteriormente eliminarlas, además el programa carga 15 patrones como ya se mencionó, pero para los tipos de patrón triángulo descendente y hombro cabeza hombro invertido se encontraban menos de 15 instancias (12 y 11 respectivamente).

Una vez que se sabía cuantos patrones nuevos se iba a necesitar para cada tipo de patrón se modificó un poco PerseumAI para que guardara el CSV de los patrones que encontraba y así utilizando la misma herramienta encontrar los patrones de buena calidad a mano.

Antes de modificar la base de datos, se realizó un estudio para identificar las características de los patrones de peor calidad en cada tipo de patrón. Esto permitió seleccionar patrones de alta calidad que se alejaran de estas características, con la esperanza de mejorar la calidad general de los patrones encontrados. Dichas características son:
Para doble techo y doble suelo (Figura 3.2):

- Asegurar que el valle entre los dos picos es lo suficientemente profundo y que no sobrepasa la línea de objetivo.
- Comprobar que los picos no estén ni muy juntos ni muy separados.
- Asegurar que la diferencia de altura entre los picos no sea muy notoria.

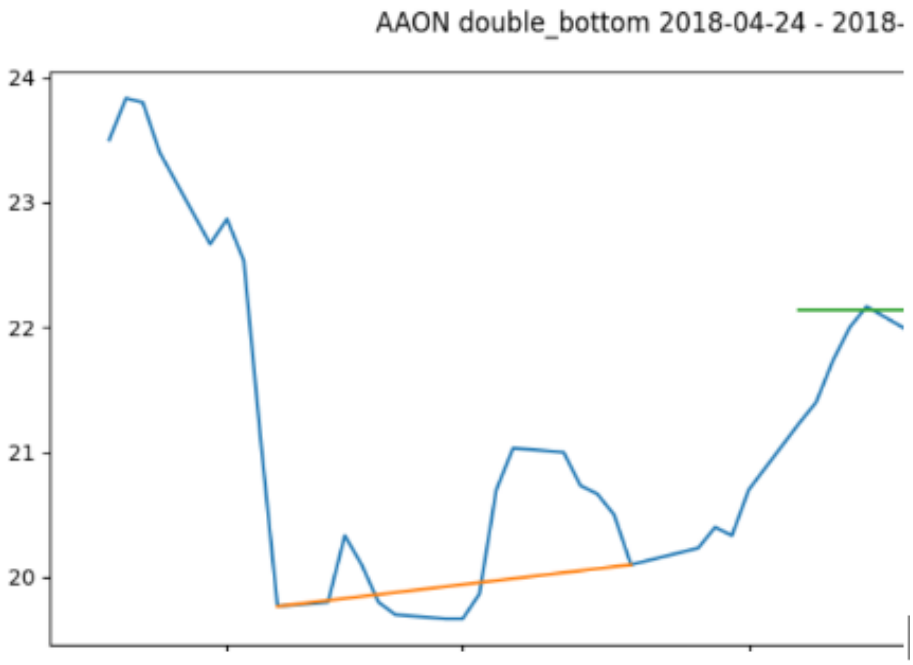


Figura 3.2: Doble suelo de mala calidad

Para hombro cabeza hombro y hombro cabeza hombro invertido (Figura 3.3):

- Comprobar que los valles son profundos y no sobrepasan la línea entre la cabeza y el hombro.
- Que la diferencia de altura entre cabezas y hombro sea notoria.
- Que la distancia entre el primer hombro y la cabeza sea parecida a la distancia entre la cabeza y el segundo hombro.

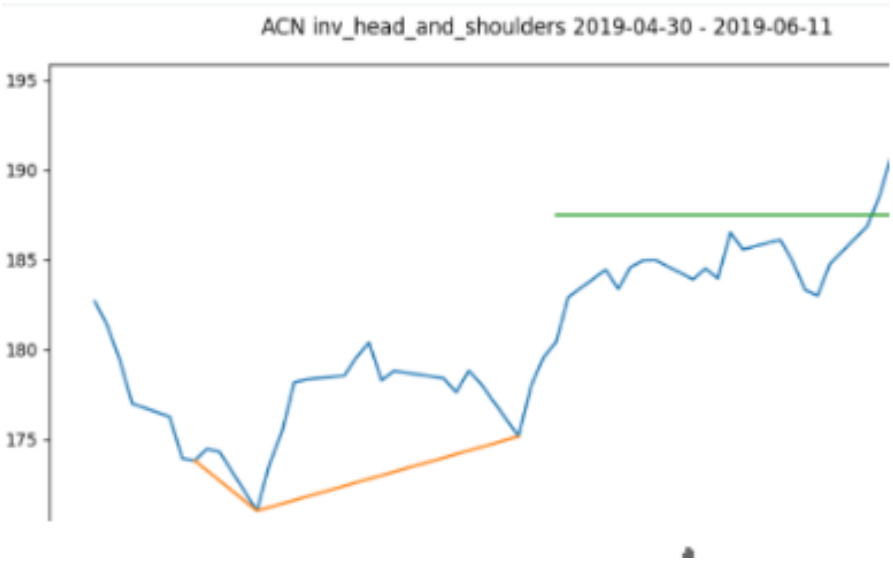


Figura 3.3: Hombro cabeza hombro invertido de mala calidad

Para triángulo ascendente y descendente (Figura 3.4):

- Asegurar que el patrón no sobrepase las líneas de resistencia y soporte.
- Que parezca un triángulo y no una cuña, este problema ya estaba comentado en el TFG de Aitor.

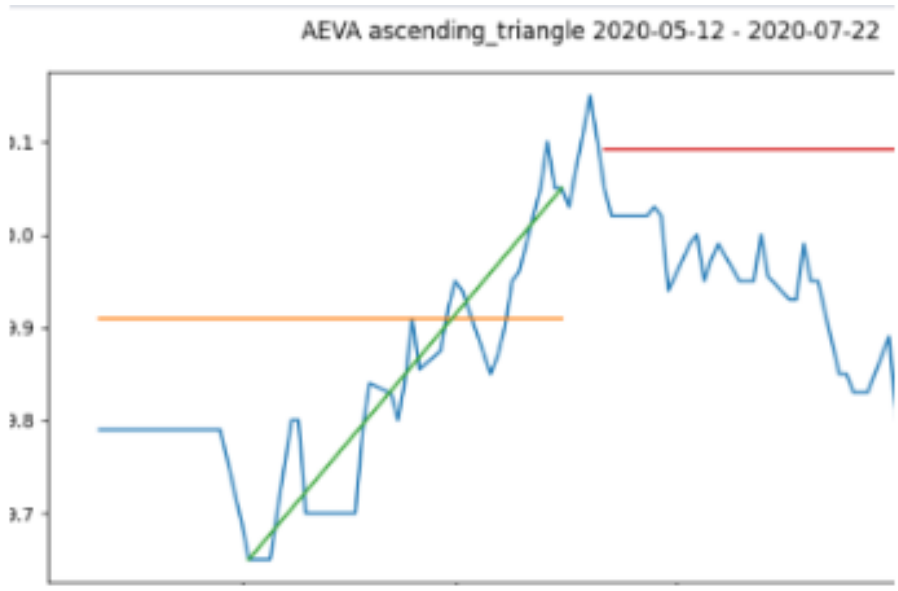


Figura 3.4: Triángulo ascendente de mala calidad

3.4. Experimento para la optimización de PerseumAI

Se ha realizado un experimento llamado "Experimento para la optimización de PerseumAI", en el cual se buscaban los valores más óptimos para los parámetros del programa, más específicamente para el número de patrones cargados y la combinación de tamaño de ventana y divisiones en relación a la duración del rango de fechas. Para ello el experimento se ha dividido en tres partes. En la primera se determinó el mejor número de patrones a cargar y se acotaron los tamaños de ventana y divisiones posibles para rangos de fechas de diferentes duraciones, en la segunda parte se determinó la calidad de los patrones encontrados para cada combinación de tamaño de ventana y divisiones, en la tercera parte se estudió para rangos de fechas de duración media y grande (que son las que más se van a usar en el programa) la mejor combinación de parámetros y como afectan a los tipos de patrones mediante distintas métricas como número de patrones encontrados, distancias de aceptación, porcentajes de aceptación...

3.4.1. Primera parte del experimento

Lo primero fue determinar el mejor valor para el número de patrones cargados de la base de datos. Para ello se compararon 10, 15, 20 y 25, como se aprecia en la Figura 3.5 apenas hay diferencia, siendo 15 patrones, el valor original, ligeramente mejor.

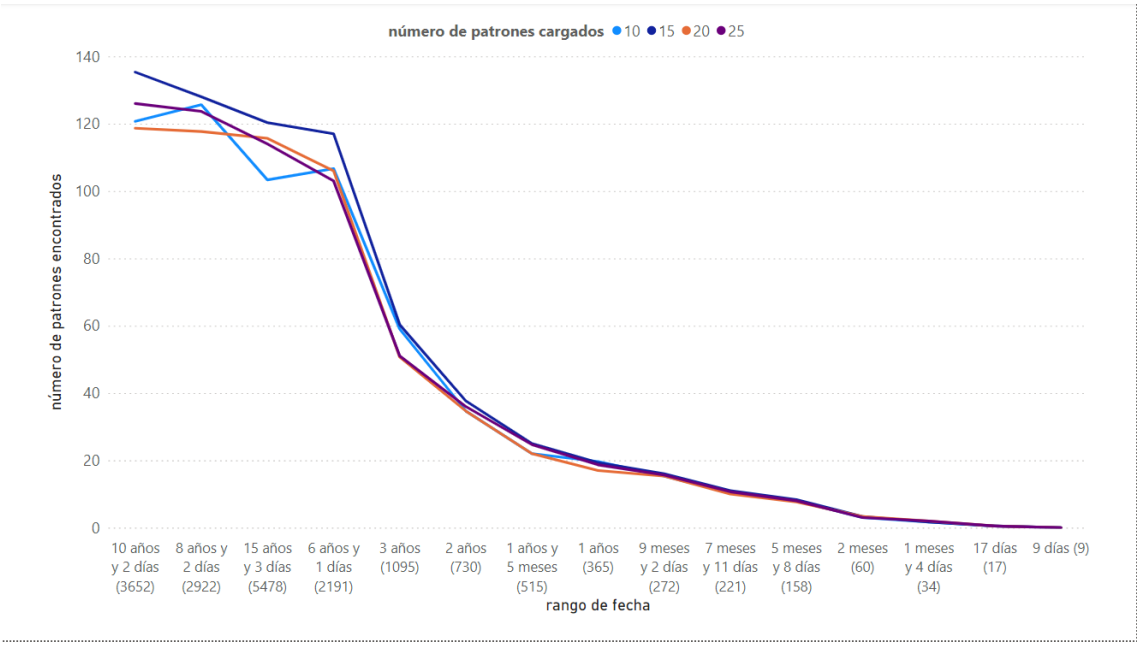


Figura 3.5: Número de patrones encontrados para diferentes patrones cargados

A partir de ahora se usará 15 como el número de patrones a cargar.

Lo siguiente fue acotar los tamaños de ventana y divisiones adaptado al rango de fechas, nos interesaba especialmente rangos pequeños (<1 o 2 años) y rangos grandes (>10 años). Se ha llegado a la conclusión que para rangos medianos y grandes la mejor división y tamaño de ventana es la misma ya que, a menor tamaño de ventana (siempre que esté por encima de 80 aproximadamente) mayor número de patrones tiende a encontrar, pero más tiempo tiende a tardar (Figura 3.6). Por lo que siempre va a ser mejor en cuanto a número de patrones encontrados tener el menor tamaño de ventana posible.

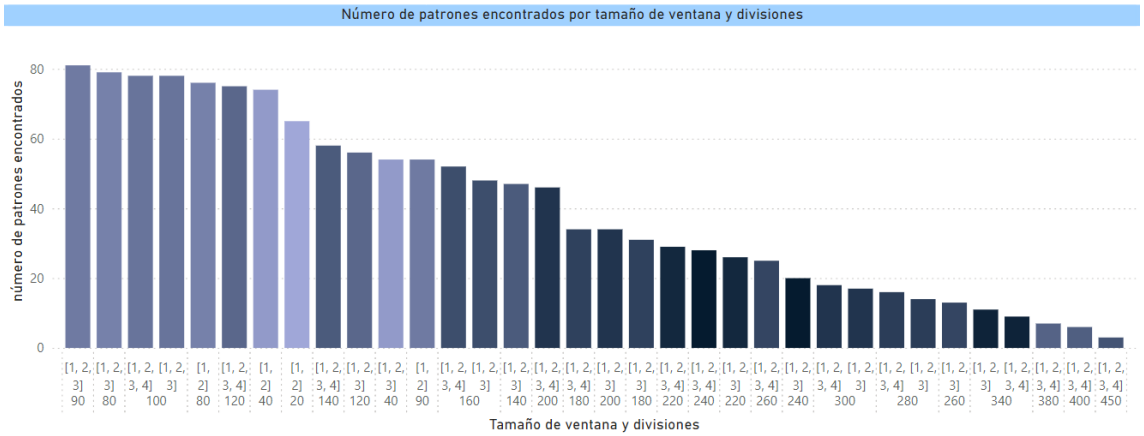


Figura 3.6: Número de patrones encontrados por división y tamaño de ventana para rangos de fecha 1 año, 1 año y medio, 2 años

El tamaño máximo de ventana que sigue encontrando patrones en un tiempo razonable es alrededor de 260. Es por eso que en la tercera parte del experimento se estudiará más en profundidad únicamente rangos de fechas de entre 1 y 15 años, ya que son donde la aplicación se va a usar mayoritariamente y la mejor combinación va a ser la misma para todos esos rangos.
En cuanto a los rangos de fechas de poca duración lo mejor parece ser pocas divisiones y tamaños de ventana entre 40 y 80, sin embargo, el problema de estos rangos de fecha es la calidad de los patrones, ya que suelen tener pocos puntos y para tipos de patrones más complejos como los triángulos es complicado encontrar patrones de calidad con tan pocos datos.

3.4.2. Segunda parte del experimento

Se estudió la calidad de los patrones, especialmente para tamaños de ventana pequeños, con el objetivo de intentar determinar cuál es el rango de fechas mínimo en el cual se encuentran patrones de calidad. Para ello, lo que se hizo fue guardar las imágenes de los patrones encontrados con cada combinación de tamaño de ventana y divisiones, para luego, manualmente, contar cuantos de los encontrados son de calidad (Tabla 3.1). Para los tamaños de ventana pequeños (<= 40) se usó un rango de fechas de tres años y para los grandes (>= 80) un rango de 7 años.

Como vemos en la tabla 3.1, se cumple que a mayor tamaño de ventana, menor número de patrones encuentra (teniendo en

cuenta que para los tamaños de ventana menor a 40 solamente se estudiaron tres años en contra de los 7 años para tamaños mayores o igual a 80). En cuanto a la calidad en rangos de fecha pequeños la mejor combinación fue un tamaño de ventana 40 con divisiones [1] con un porcentaje de calidad del 51 %, lo cual es bastante malo además teniendo en cuenta que prácticamente todos los patrones acertados eran doble techo y doble suelo, para patrones más complejos como los triángulos y hombro cabeza hombro se necesita de un mayor tamaño de ventana para encontrar patrones de calidad. Para tamaños de ventana grande el porcentaje de calidad se encuentra en un rango entre 50 y 70 %, para tamaños de ventana más pequeños funciona mejor menos divisiones mientras que tamaños más grandes, a partir de 200, suelen requerir más divisiones. No por aumentar el tamaño de ventana aumenta la calidad, además de que encuentra muchos menos patrones. La combinación que mejor calidad presentó fue un tamaño de ventana de 160 con divisiones [1, 2] sin embargo es bastante posible que si se cambian las empresas que se estudian en el experimento o el rango de fechas la mejor combinación pase a ser otra. En definitiva no hay una mejor combinación para todos los casos, para encontrar patrones de calidad el tamaño de ventana debe ser mayor de 80, y en general con menor divisiones tiene mejor calidad.

Tamaño de ventana	Divisiones	Resultado	Porcentaje Calidad
5	[[1]]	0/0	0 %
10	[[1]]	0/26	0 %
10	[[1, 2]]	0/26	0 %
20	[[1, 2]]	2/45	4 %
20	[1]	14/59	23 %
40	[[1]]	37/73	50.68 %
40	[[1, 2]]	25/56	44 %
40	[[1, 2, 3]]	20/54	37 %
80	[[1, 2]]	77/117	65 %
80	[[1, 2, 3]]	60/122	49 %
90	[[1, 2]]	64/110	58 %
90	[[1, 2, 3]]	63/126	50 %
100	[[1, 2]]	59/86	68 %
100	[[1, 2, 3]]	59/107	55 %
100	[[1, 2, 3, 4]]	61/128	47 %
100	[[1, 2, 3, 4, 5]]	51/110	46 %
120	[[1, 2]]	43/77	57 %
120	[[1, 2, 3]]	64/111	57 %
120	[[1, 2, 3, 4]]	50/106	47 %
120	[[1, 2, 3, 4, 5]]	63/120	50 %
140	[[1, 2]]	64/90	71 %
140	[[1, 2, 3]]	62/102	60 %
140	[[1, 2, 3, 4]]	53/91	58 %
140	[[1, 2, 3, 4, 5]]	55/105	52 %
160	[[1, 2]]	56/71	78 %
160	[[1, 2, 3]]	48/71	67 %
160	[[1, 2, 3, 4]]	46/82	56 %
160	[[1, 2, 3, 4, 5]]	47/86	54 %
180	[[1, 2]]	43/57	75 %
180	[[1, 2, 3]]	37/63	58 %
180	[[1, 2, 3, 4]]	37/71	52 %
180	[[1, 2, 3, 4, 5]]	45/85	52 %
200	[[1, 2]]	28/47	59 %
200	[[1, 2, 3]]	24/58	41 %
200	[[1, 2, 3, 4]]	38/68	55 %
200	[[1, 2, 3, 4, 5]]	38/77	49 %
220	[[1, 2]]	19/39	48 %
220	[[1, 2, 3]]	28/51	54 %
220	[[1, 2, 3, 4]]	28/59	47 %
220	[[1, 2, 3, 4, 5]]	46/80	57 %
240	[[1, 2]]	25/45	55 %
240	[[1, 2, 3]]	31/56	55 %
240	[[1, 2, 3, 4]]	33/52	63 %
240	[[1, 2, 3, 4, 5]]	32/60	53 %

Tabla 3.1: Resultados del experimento de calidad

3.4.3. Tercera parte del experimento

En la última parte del experimento se estudiaron combinaciones de tamaño de ventana y divisiones para rangos de fechas de duración media y grande, cada combinación se estudió en cada rango de fecha, por lo que podremos comparar más adecuadamente cuáles son las mejores combinaciones. Al contrario que en la primera parte del experimento, que cada combinación solo se estudiaba en unos rangos determinados y no se podía comparar entre diferentes rangos de fecha.

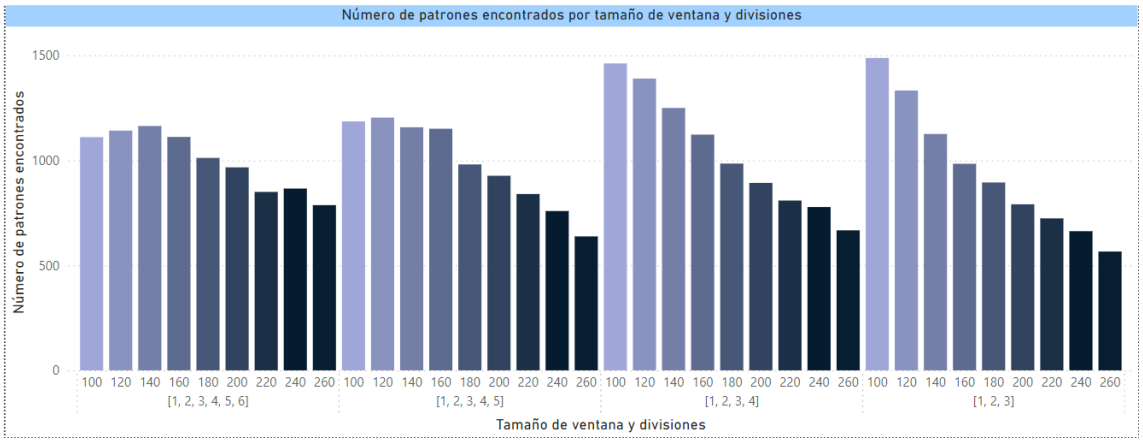


Figura 3.7: Número de patrones encontrados para cada combinación de tamaño de ventana y divisiones

Como se aprecia en la Figura 3.7 se sigue cumpliendo que a mayor tamaño de ventana, menor número de patrones encuentra. También ocurre que a mayor número de divisiones encuentra más patrones con tamaños de ventana grandes y con tamaños de ventana más pequeños encuentra más patrones con menos divisiones. Lo cual concuerda con los resultados del experimento de calidad.

También se estudiaron los tamaños de los patrones (Figura 3.8)

Figura 3.8: Tamaño medio de los patrones por tamaño de ventana y divisiones

A menos divisiones y mayor tamaño de ventana los patrones contienen más puntos, sin embargo, como ya se demostró en la segunda parte del experimento que un patrón sea más grande no implica que tenga más calidad.

Finalmente, se estudiaron las distancias con las que se aceptan los patrones en findCommon (cuando se detecta el patrón en la ventana por primera vez), distancias de aceptación final (distancia de los patrones encontrados finalmente) y porcentaje de patrones aceptados por el filtro (Figura 3.9).

Distancia por patrón

Tipo de patrón	Distancia	Distancia fc	Porcentaje aceptación
ascending_triangle	19,25	22,57	9,46
descending_triangle	21,64	27,22	7,75
double_bottom	23,00	23,82	27,39
double_top	17,20	16,68	29,40
head_and_shoulders	24,41	26,82	10,42
inv_head_and_shoulders	20,34	20,65	15,58
Total	**20,97**	**22,96**	**16,67**

Figura 3.9: Distancias y porcentaje de aceptación por tipo de patrón

La distancia media de aceptación tanto en findCommon como

en los patrones finales es bastante menor a 40, la distancia mínima para aceptar el patrón, por lo que ese parámetro se puede optimizar para que la distancia de aceptación dependa del patrón detectado, usando estos datos como base para establecer los valores. La distancia en findCommon es de media mayor a la distancia final, lo cual tiene sentido ya que de los que se acepta en findCommon, solamente los más parecidos se acaban dando por buenos. De media, solo el 16.67 % de las ventanas que se detectan en primera instancia como un patrón acaban siendo aceptadas por el filtro. Para los tipos de patrones más sencillos, doble techo y doble suelo, este porcentaje es de alrededor del 30 %, mientras que para los tipos de patrones más complejos es de alrededor del 10 %, lo cual es bastante malo ya que de cada 100 ventanas en las que se detecta un patrón, únicamente 10 tienen verdaderamente el patrón. Esto conlleva un gran gasto de recursos ya que se están explorando muchas ventanas que en primer lugar no debería haber detectado ningún patrón, por lo que con la reducción de la distancia de aceptación y adaptándola a cada tipo de patrón se espera solucionar este problema.

3.5. Nuevo algoritmo de búsqueda de patrones históricos

Usando los datos del experimento se ha diseñado un nuevo algoritmo para la búsqueda de patrones históricos que consiste en que el usuario a partir de ahora no tiene que escoger un tamaño de ventana sino un tipo de búsqueda (Figura 3.10).

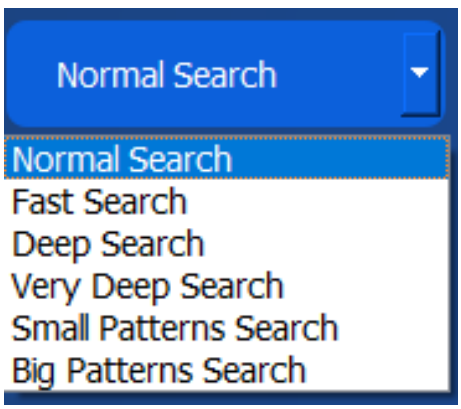


Figura 3.10: Tipos de búsqueda

Internamente lo que se selecciona es una lista con tamaños de ventana, en este nuevo algoritmo en vez de explorar una sola ventana lo que se hace es explorar una lista de ventanas, luego dependiendo del tamaño de ventana se eligen las divisiones. También se cambió la distancia de aceptación en findCommon para que dependa del tipo de patrón que se detecta, para patrones más sencillos se puso una distancia menor y para patrones más complejos una distancia mayor para que le sea más sencillo encontrarlos.

También se cambió la manera en la que se calculaba la siguiente ventana a explorar, antes lo que se hacía era sumar un valor fijo (25) al índice izquierdo de la ventana. Esto no tiene en cuenta el tamaño de la ventana, para grandes como pequeñas siempre va a sumar el mismo valor, lo que puede hacer que en ventanas pequeñas perdamos muchos datos, o en ventanas grandes que el incremento sea muy pequeño y exploremos lo mismo muchas veces. Por eso ahora el valor que se suma es el 25 % del tamaño de la ventana.

Finalmente se creo un algoritmo que dado una lista de patrones elimina los repetidos, ya que al explorar los mismos datos varias veces encuentra lo mismo frecuentemente. Funciona comprobando que los patrones sean de la misma empresa, mismo tipo de patrón, que la tendencia del patrón sea la misma y que las fechas de inicio y de fin estén en el mismo rango con un margen de 35 días.

3.6. Nuevo algoritmo de búsqueda de patrones actuales

También se ha diseñado un algoritmo más complejo para detectar los patrones actuales, en vez de explorar únicamente el tamaño de ventana que se le pasa por parámetro lo que se hace es ir reduciendo el tamaño de la ventana hasta tres veces si no se encuentra el patrón. Además si se encuentra un patrón también se disminuye el tamaño de la ventana para ver si mejora (Figura 3.11).

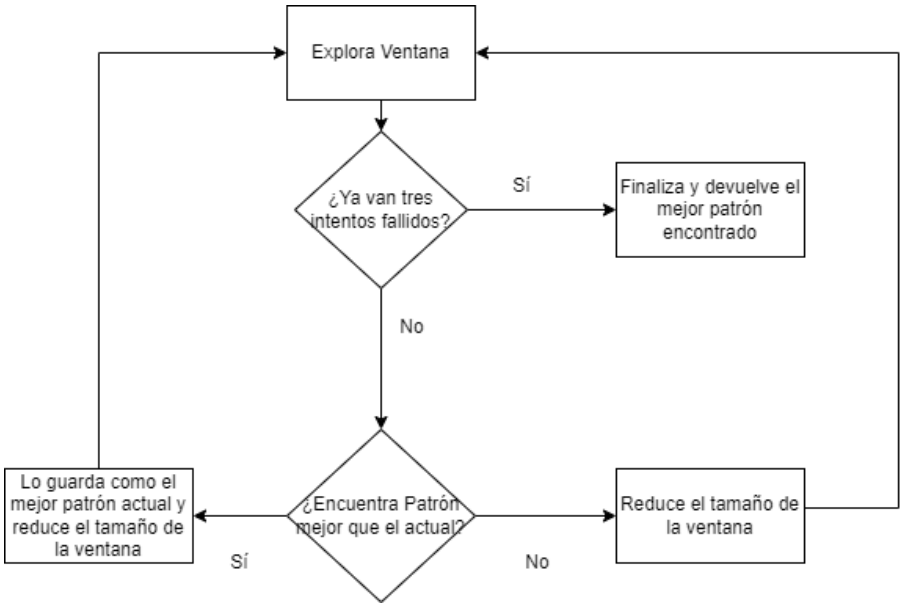


Figura 3.11: Algoritmo para la búsqueda de patrones actuales

Al igual que con la búsqueda de patrones históricos se explora una lista de ventanas, sin embargo, en este caso no hace falta eliminar patrones repetidos, desde que encuentra un patrón actual se dejan de explorar el resto de ventanas, por lo que cada vez que se llama a la función sólo devuelve un patrón.

3.7. Creación de datos para entrenamiento de la red neuronal

Una vez que hemos mejorado la búsqueda de patrones con DTW podemos empezar con la red neuronal, el primer paso es elegir la arquitectura. Se barajaron una red LSTM y una red convolucional y finalmente se decidió por una convolucional al ser más sencilla y haber más documentación. La resolución de las imágenes de entrada será de 256x64, tiene un buen balance entre calidad de la imagen y complejidad computacional (a mayor tamaño de imagen, más lenta irá la red). En la Figura 3.12 se muestra un ejemplo de la imagen de un patrón hombro cabeza hombro usado en el entrenamiento.

Figura 3.12: Imagen usada en el entrenamiento

La creación de la base de datos para el entrenamiento consistió en dos fases, primero se creo un script que buscaba patrones históricos y guardaba en un directorio la imagen y un csv con los puntos del patrón. Luego, a mano se escogieron los patrones de mayor calidad de los encontrados y se programó un script que crea patrones sintéticos basados en un patrón pasado por parámetro. Para obtener patrones sintéticos se añaden nuevos puntos al patron original, basados en la media entre los dos puntos entre los que se introduce, y se modifican puntos existentes, restándoles o sumándoles entre un 10 % y un 50 % de su valor. El número de puntos a añadir es un valor aleatorio entre el 30 % y el 50 % del número de puntos originales, mientras que el número de puntos a modificar es un valor aleatorio entre el 40 % y el 70 %. A estos porcentajes se llego mediante la práctica, si se es demasiado conservador con los porcentajes los patrones generados apenas variarán del original, pero si nos pasamos es posible que el patrón pierda su forma. En la Figura 3.13 vemos un ejemplo de ejecución en el que creamos tres patrones sintéticos basados en un hombro cabeza hombro.

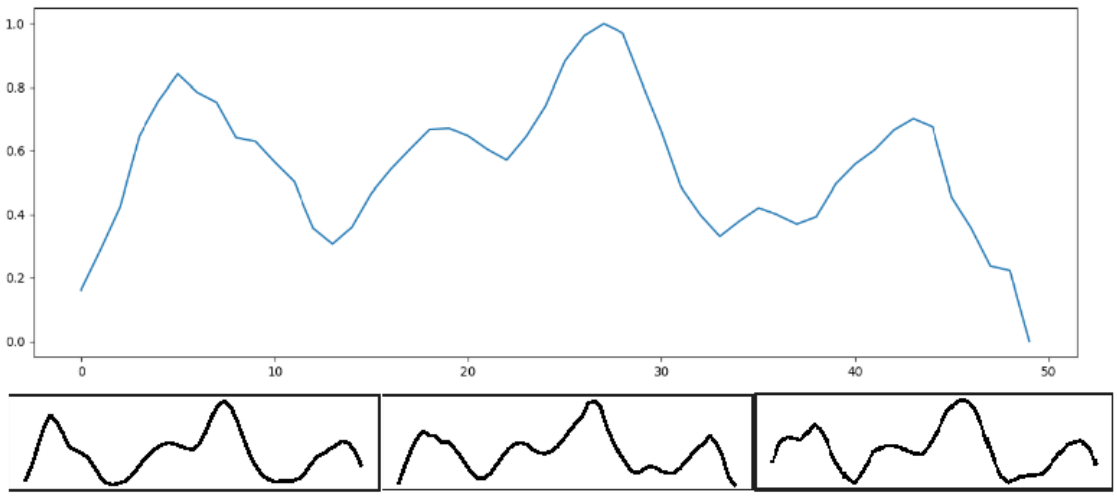


Figura 3.13: Patrón original y 3 patrones sintéticos basados en este

Usando este script creamos la base de datos de imágenes de patrones, con alrededor de 2300 instancias para cada tipo de patrón. Además de los patrones también se guardaron imágenes de gráficos de precio sin patrones.

3.8. Desarrollo y entrenamiento de la red neuronal

Una vez se dispone de datos se puede comenzar con el desarrollo y entrenamiento de la red, usando la libería PyTorch (PyTorch Contributors (2023)). Se decidió probar con una red para cada patrón (Figura 3.14) y una red general para todos los patrones, la arquitectura para ambas es la misma, solamente cambia el número de neuronas en la capa de salida, dos neuronas de salida en el caso de una red por patrón y siete neuronas en el caso de una red general (los seis tipos de patrón de la herramienta y ningún patrón).

```python
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(in_channels=3, out_channels=16, kernel_size=3, stride=1, padding=1)
        self.conv2 = nn.Conv2d(in_channels=16, out_channels=32, kernel_size=3, stride=1, padding=1)
        self.conv3 = nn.Conv2d(in_channels=32, out_channels=64, kernel_size=3, stride=1, padding=1)
        self.pool = nn.MaxPool2d(kernel_size=2, stride=2, padding=0)
        self.fc1 = nn.Linear(64 * 32 * 8, 128)
        self.fc2 = nn.Linear(128, 2) # 2 neuronas de salida

    def forward(self, x):
        x = self.pool(torch.relu(self.conv1(x)))
        x = self.pool(torch.relu(self.conv2(x)))
        x = self.pool(torch.relu(self.conv3(x)))
        x = x.view(-1, 64 * 32 * 8)
        x = torch.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

Figura 3.14: Red neuronal por patrón

La red consiste en capas convolucionales que extraen características locales de la imagen, detectando patrones como bordes, texturas y formas básicas en diferentes niveles de abstracción. Cada capa convolucional aplica filtros a la imagen de entrada y genera mapas de características que resaltan ciertas propiedades visuales. La salida de la capa convolucional pasa a una capa de pooling que reduce la dimensionalidad de los mapas de

características, disminuyendo el número de parámetros y la carga computacional, además de proporcionar cierta invariancia a la traslación (pequeñas variaciones de la imagen). Finalmente, se aplana la salida de la última capa de pooling en un vector de características y se le aplican capas lineales que realizan la combinación y transformación final de las características extraídas por las capas convolucionales, permitiendo que la red tome decisiones de clasificación basadas en las características combinadas y aprendidas de las imágenes.

Terminado el desarrollo de la red se procede al entrenamiento, se dividieron los datos en conjuntos de entrenamiento con un 80 % y testeo el 20 % restante. Es importante dividir los datos en conjuntos de entrenamiento y testeo, ya que permite evaluar el rendimiento del modelo de manera objetiva. El conjunto de entrenamiento se utiliza para ajustar los parámetros del modelo, mientras que el conjunto de testeo, que no se ha visto durante el entrenamiento, sirve para evaluar cómo generaliza el modelo a datos nuevos e independientes. Lo que nos permite detectar problemas como el sobreajuste, donde el modelo funciona bien en los datos de entrenamiento pero falla en datos no vistos.

El entrenamiento de la red consiste en ajustar los parámetros del modelo (pesos y sesgos) mediante la retropropagación y la optimización iterativa, utilizando un conjunto de datos de entrenamiento. Durante este proceso, el modelo realiza predicciones sobre los datos de entrenamiento, calcula la pérdida o error comparando las predicciones con las etiquetas reales, y luego ajusta los parámetros para minimizar esta pérdida mediante el algoritmo del descenso de gradiente estocástico.

Tras finalizar el entrenamiento se guarda un fichero con los parámetros del modelo, para poder usarlo más adelante. Tanto para las redes por patrón como para la red conjunta, el porcentaje de acierto con el conjunto de testeo ronda entre el 96 %-98 % dependiendo del patrón (Figura 3.15).

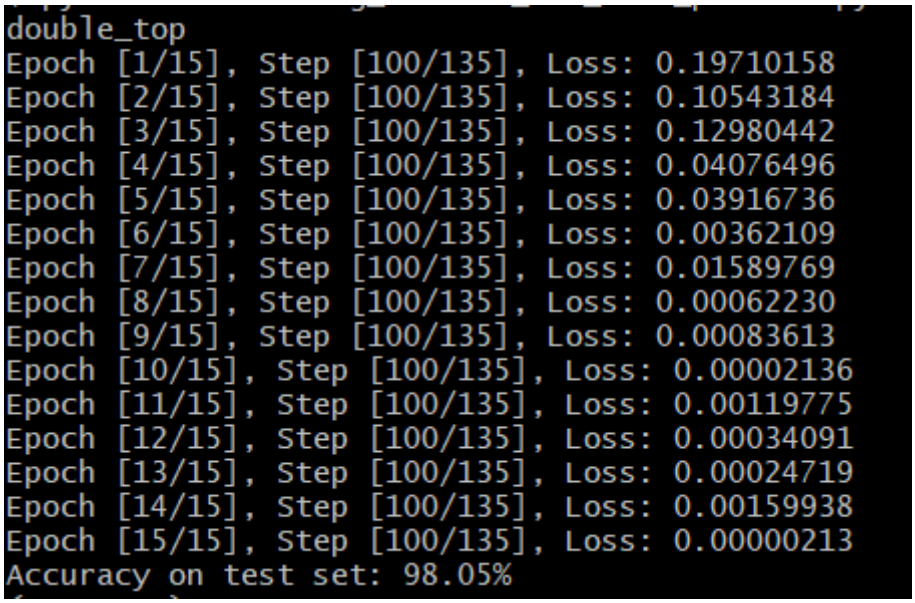


Figura 3.15: Entrenamiento de la red para el patrón doble techo

3.9. Implementación de la red en el programa

Para poder implementar la red al programa se requiere:

1. Convertir la ventana que se esta explorando a imagen.
2. Convertir la imagen de la ventana a un tensor de pytorch para poder pasársela a la red.
3. Pasarle el tensor a la red neuronal y que realice la clasificación.

El primer y segundo paso fueron automáticos, en el entrenamiento se desarrollaron funciones que convertían la ventana a imagen y la imagen a tensor. En el caso de una red por patrón la función que realiza la clasificación recibe una lista de modelos (un modelo para cada tipo de patrón) y para cada modelo realiza la clasificación, que devuelve la clase predicha (si es o no el patrón) y un valor de confianza, al final se queda con la clase con mayor confianza. Para la red conjunta es más sencillo, solamente hay que realizar la clasificación y devuelve la clase que predice.

Los algoritmos de búsqueda de patrones con red neuronal son los mismos que con DTW, la diferencia es que la función que te decía si en una ventana había un patrón y con que distancia ahora es la función de clasificación, que devuelve lo mismo (el tipo de

patrón detectado), pero en vez de distancia la confianza de la clasificación. Si se recuerda el funcionamiento del programa, una vez que se detecta el patrón y si su distancia es menor a la distancia mínima para aceptarlo (en este caso si la confianza es mayor a la confianza mínima), se explora la ventana en subventanas con las divisiones y si alguna subventana es mejor se usa esa para pasarla al filtro. Se realizó un pequeño estudio para determinar que era mejor, si no explorar la ventana con subventanas y pasársela tal cual al filtro, si explorarla con el modo clásico (usando DTW), o explorarla pero en vez de con DTW con la propia red neuronal. El resultado fue que explorando en subventanas con DTW es mejor, con red neuronal tardaba bastante más (156 segundos con DTW y 234 segundos con red) y la calidad era ligeramente peor, había patrones que no mejoraban bien y se quedaban con la ventana original, lo que daba la impresión que los patrones estaban alejados (Figura 3.16)

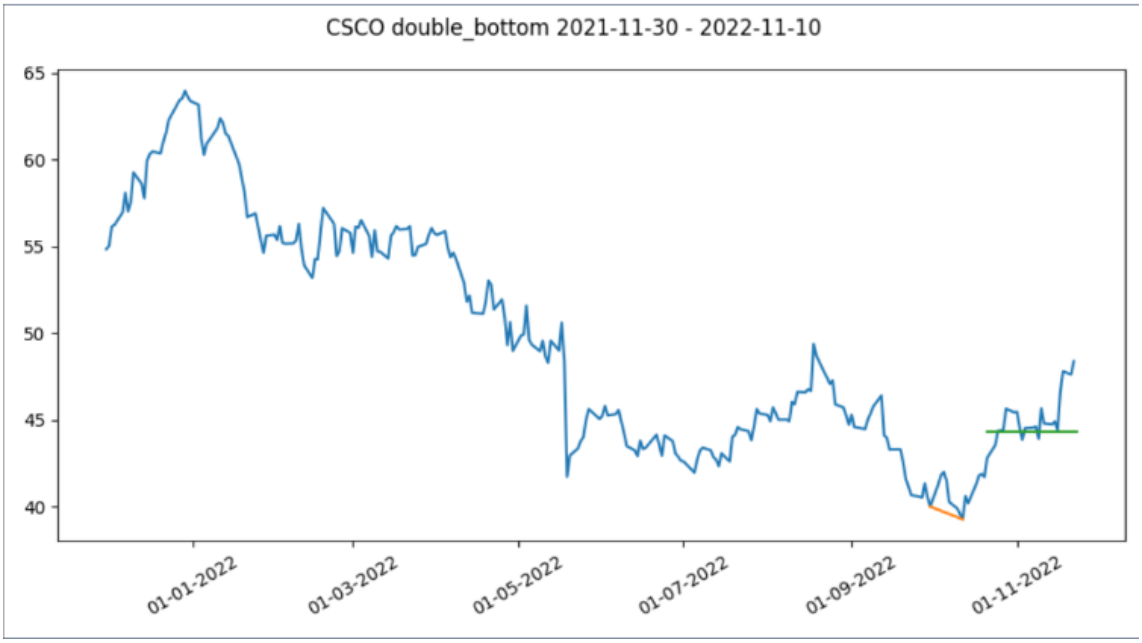


Figura 3.16: Ejemplo de patrón alejado

En cuanto al algoritmo de detección de patrones actuales, es el mismo pero se cambia la detección con DTW y distancias por clasificación con la red y confianza.

También se comparó el desempeño de la red conjunta (C) contra las redes por patrón (PP) utilizando el algoritmo de búsqueda de

patrones históricos para diferentes conjuntos de empresas y el resultado fue el siguiente (Tabla 3.2):

Conjunto	Tiempo PP (s)	Tiempo C (s)	Encontrados PP	Encontrados C
1	55.25	32.47	46	32
2	46.65	29.53	42	37
3	49.26	30.32	31	31
4	56.97	34.44	44	25

Tabla 3.2: Resultados experimento red por patrón versus red conjunta

La red por patrón tarda en promedio un 50 % más pero encuentra en promedio un 20 % más de patrones por lo que nos decantaremos por esta. Finalmente se realizó un estudio para comparar la búsqueda de patrones históricos utilizando red neuronal contra DTW, las empresas usadas en el estudio fueron 30 y se muestran en la Figura 3.17, y el rango de fechas usado fue entre el 1 de enero de 2018 y el 1 de enero de 2024.

```
conjunto_empresas = [
    'EXAS', 'EXC', 'EXEL', 'EXFY', 'EXK', 'EXLS', 'EXP', 'EXPD', 'EXPE', 'EXPI',
    'EXPO', 'EXR', 'EXTO', 'EXTR', 'EYE', 'EYEN', 'EYPT', 'EZFL', 'EZGO', 'EZPW',
    'GEG', 'GEHC', 'GEL', 'GEN', 'GENC', 'GENE', 'GENI', 'GENK', 'GEO', 'GEOS'
]
```

Figura 3.17: Empresas usadas en el experimento para comparar el desempeño de la búsqueda de patrones usando red neuronal contra DTW

Los resultados se muestran en la Tabla 3.3):

Número encontrados	**Resultados modo red**
Número total de patrones	96
Número total triángulo ascendente	14
Número total triángulo descendente	34
Número total doble suelo	21
Número total doble techo	9
Número total hombro cabeza hombro	9
Número total hombro cabeza hombro invertido	9
Tiempo total	294.68 segundos
Número encontrados	**Resultados modo clásico**
Número total de patrones	516
Número total triángulo ascendente	79
Número total triángulo descendente	69
Número total doble suelo	128
Número total doble techo	137
Número total hombro cabeza hombro	37
Número total hombro cabeza hombro invertido	66
Tiempo total	842.67 segundos

Tabla 3.3: Resultados experimento red neuronal versus DTW

Como se puede ver, el modo clásico con DTW encuentra muchísimos más patrones (un 137 % más) pero también tarda muchísimo más, un 96 % más. Además la red neuronal parece encontrar más triángulos que el resto de patrones, incluso más que doble techo y doble suelo, lo cual puede indicar que estos patrones más complejos están absorbiendo a los patrones más simples. Para intentar paliar esto se aumentará la confianza necesaria para encontrar triángulos, y se disminuira para el resto de tipos de patrones. También se realizo este estudio con la búsqueda de patrones actuales

y los resultados fueron los siguientes (Tabla 3.4):

Número encontrados	**Resultados modo red**
Número total de patrones	18
Número total triángulo ascendente	0
Número total triángulo descendente	1
Número total doble suelo	7
Número total doble techo	0
Número total hombro cabeza hombro	4
Número total hombro cabeza hombro invertido	6
Tiempo total	190.3 segundos
Número encontrados	**Resultados modo clásico**
Número total de patrones	20
Número total triángulo ascendente	0
Número total triángulo descendente	0
Número total doble suelo	17
Número total doble techo	1
Número total hombro cabeza hombro	0
Número total hombro cabeza hombro invertido	2
Tiempo total	317.8 segundos

Tabla 3.4: Resultados experimento red conjunta versus red por patrón

Las conclusiones que podemos sacar son que a pesar de que en la búsqueda de patrones históricos es mucho mejor el modo clásico, ya que nos centramos en calidad y cantidad de patrones encontrados más que en tiempo de ejecución, en cuanto a la búsqueda de patrones actuales la red neuronal es mejor, encontrando más variedad de patrones.

3.10. Desarrollo de la interfaz gráfica y aplicación final

Para el desarrollo de la interfaz final se utilizó la librería PyQt5 (Qt for Python Development Team (2023)), haciendo uso de la herramienta PyQt5 designer, que nos permite crear la interfaz de manera gráfica seleccionando y arrastrando los elementos que queremos introducir (Figura 3.18)

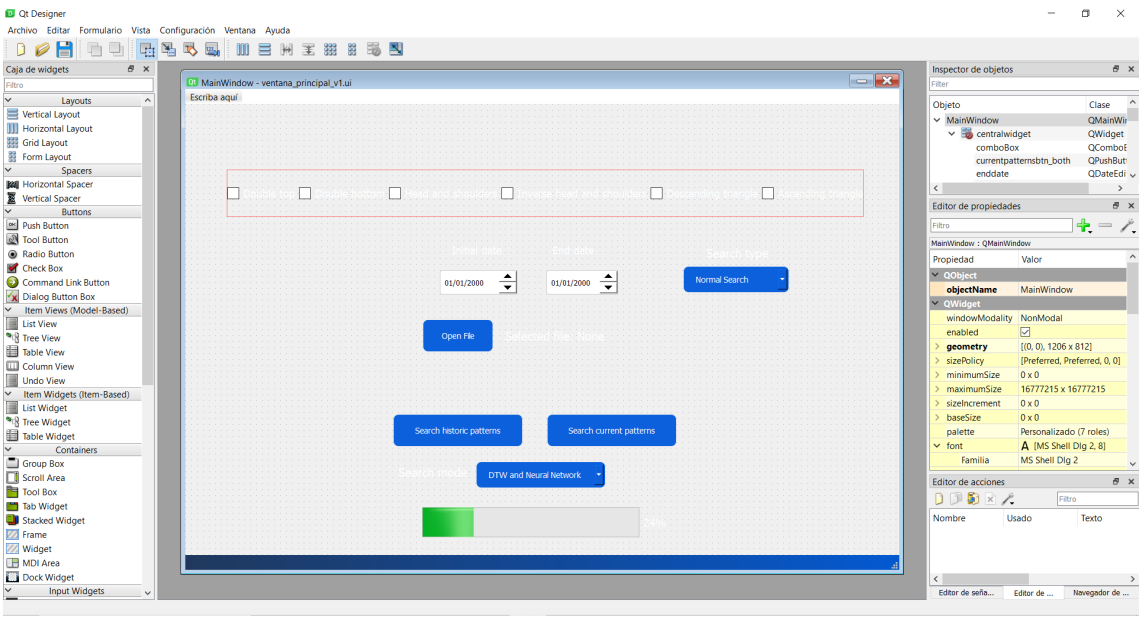


Figura 3.18: Aplicación pyqt5 designer

Una vez que se tiene la interfaz se exporta a un fichero python para poder añadirle funcionalidades. Con esta nueva interfaz se han arreglado varios problemas que tenía la versión anterior, siendo el más importante que si se encontraban muchos patrones ocurría lo que en la Figura 3.19,

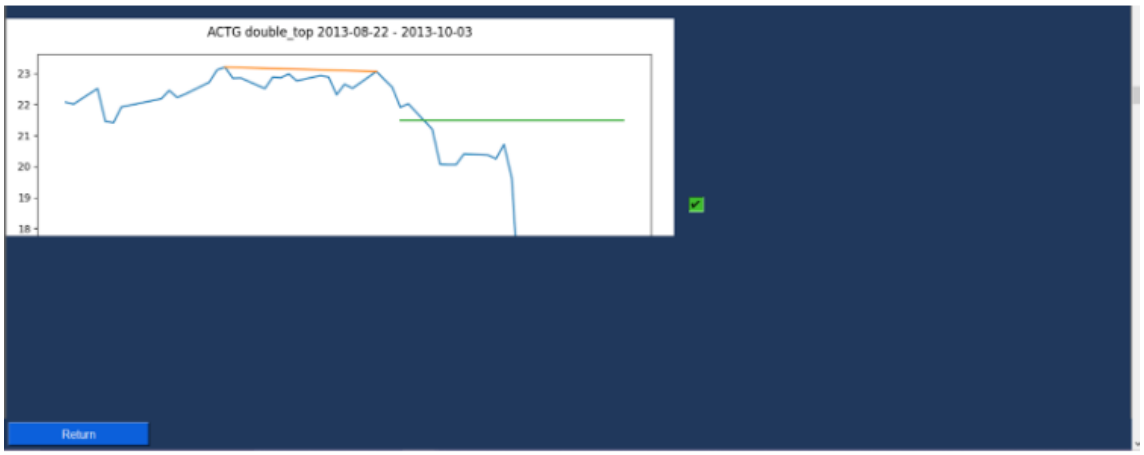


Figura 3.19: Error en la versión anterior de PerseumAI

llegaba un punto en el que no cargaba más patrones. También se solucionó el problema de la barra de carga, la cual estaba implementada en el programa original, pero a veces funcionaba y a veces no. El cambio más relevante respecto a la versión anterior es la abstracción del usuario respecto al funcionamiento del programa, antes el usuario tenía que introducir el tamaño de ventana, sin embargo a no ser que se leyera el TFG era imposible saber que era exactamente y como funcionaba, se intentaba solucionar el problema poniendo una tabla de tamaños de ventana recomendados (Figura 3.20). Para solucionarlo se hizo lo explicado en el apartado 3.4, en vez de seleccionar un tamaño de ventana, ahora el usuario elige el tipo de búsqueda.

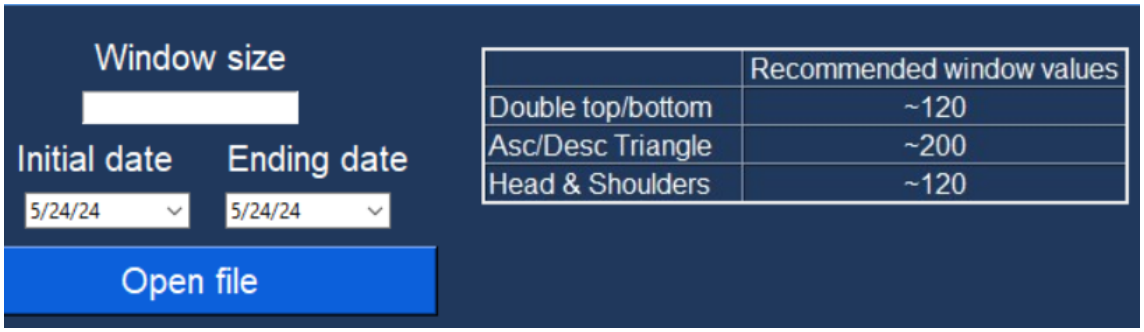


Figura 3.20: Tabla con las distancias recomendadas

También se separó la búsqueda de patrones actuales e históricos, algo que antes estaba junto en un solo botón de búsqueda de patrones y se añadieron varias opciones de búsqueda, buscar patrones con red neuronal y modo clásico, buscar patrones solo con el modo clásico o solo con la red. En definitiva, la nueva interfaz

consistiría de (Figura 3.21):

1. Patrón a buscar: Hay unos checkbox que permiten seleccionar los patrones que se desea analizar.
2. Guardar imágenes de los patrones: en el caso de activar el checkbox los patrones encontrados se guardarán como imágenes en la carpeta saved_patterns.
3. Fechas de inicio y fin: rango de fechas que se desea explorar.
4. Tickers de las empresas a analizar: El botón 'Open File' abre una ventana para seleccionar un fichero de texto que contenga los tickers de las empresas a analizar.
5. Tipo de búsqueda: te permite personalizar la búsqueda de patrones.
6. Modo de búsqueda: si queremos utilizar DTW, red neuronal o ambos para la búsqueda de patrones.
7. Botones de búsqueda de patrones históricos y actuales.

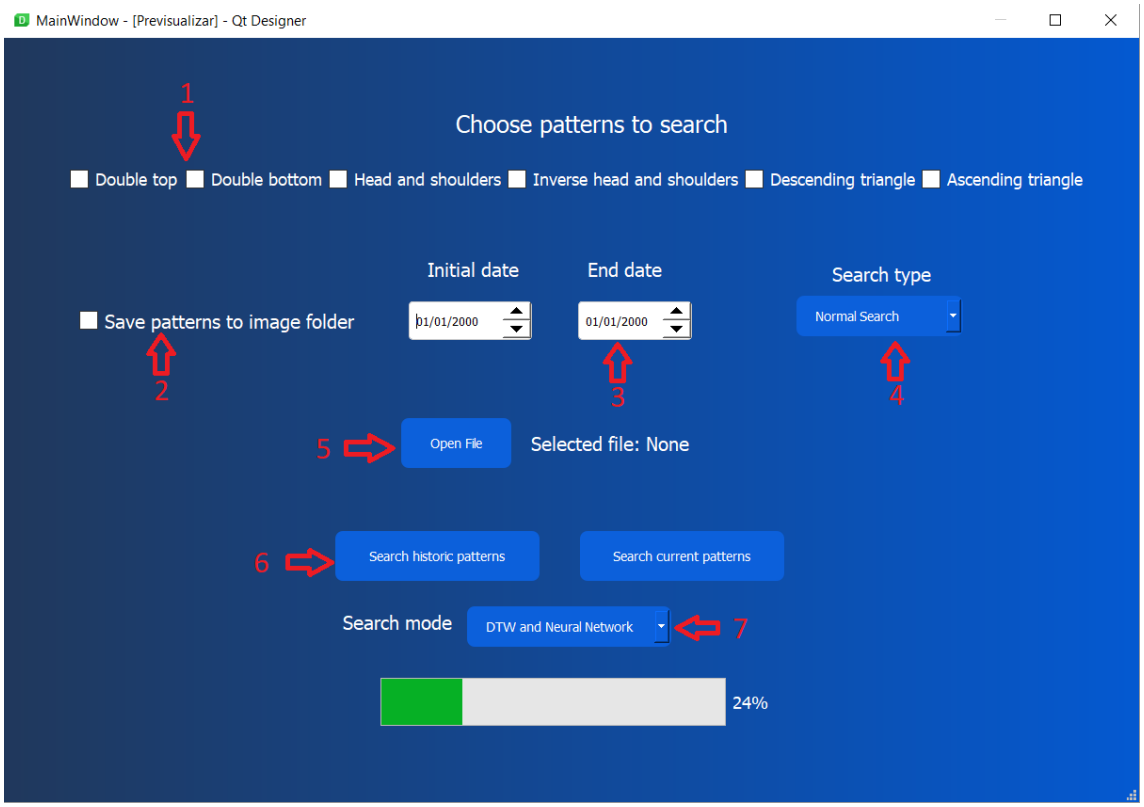


Figura 3.21: Nueva interfaz de PerseumAI

Capítulo 4

Resultados obtenidos

4.1. Desempeño de la herramienta respecto a la última versión

Para determinar si los cambios introducidos se traducen en un mejor rendimiento respecto a la última versión del programa, se va a realizar el mismo experimento de backtesting que en el TFG del año pasado y se compararán los resultados.

El experimento consistirá en estudiar los patrones encontrados para dos conjuntos de empresas, uno del sector tecnológico y el otro del sector financiero, para el periodo que comprende entre el 1 de enero de 2005 hasta el 3 de julio de 2023, y se comparán con los encontrados en la versión anterior de la herramienta. Estas fechas han sido seleccionadas coincidiendo con las fechas del trabajo anterior para poder realizar un estudio comparativo. También se estudió el origen de los patrones y se compararon todos los encontrados por DTW y por red neuronal para determinar si los encontrados con la red neuronal, al ser bastante menos, eran diferentes a los que se encuentran con la técnica DTW.

La elección de las empresas usadas, al igual que se hizo en el TFG de Aitor, se basa en empresas que tienen más de 15 años desde su oferta pública inicial, es decir, con una gran cantidad de datos para poder analizar, y mezcladas empresas de mayor y de menor capitalización.

4.1.1. Sector tecnológico

Las empresas pertenecientes al sector tecnológico seleccionadas se muestran en la tabla 4.1:

Ticker	Compañía	Cap. de Mercado
AAPL	Apple Inc.	$2,92 Billones
MSFT	Microsoft	$3,19 Billones
NVDA	NVIDIA Corporation	$2,82 Billones
ORCL	Oracle Corporation	$340,10 mil millones
CSCO	Cisco Systems, INC	$185,65 mil millones
NOK	Nokia Oyj	$21,14 mil millones
WDC	Western Digital Corporation	$25,28 mil millones
LOGI	Logitech International S.A.	$15,07 mil millones
LPL	LG Display Co., Ltd.	$20,90 mil millones
NEON	Neonode Inc.	$51,45 millones

Tabla 4.1: Conjunto de empresas del sector tecnológico

En la tabla 4.2 se mostrarán los resultados de la búsqueda de patrones comparándola con el trabajo de Aitor. La columna ’ %’ representa el porcentaje de patrones del total que han cumplido con el objetivo, y la columna ’Nº’ el número de patrones encontrados. Las columnas que acaban en *P* representan lo mismo pero son los resultados obtenidos en el TFG de Aitor, finalmente la columna ’Dif’ representa el porcentaje de diferencia entre el número de patrones encontrados este año y el pasado.

Patrón	%	Nº	% P	Nº P	Dif
Doble techo	67 %	212	67 %	83	87 %
Doble suelo	62 %	184	69 %	81	78 %
Triángulo ascendente	51 %	110	82 %	11	164 %
Triángulo descendente	44 %	68	50 %	2	186 %
Hombro cabeza hombro	65 %	43	73 %	19	77 %
Hombro cabeza hombro invertido	69 %	70	72 %	36	64 %

Tabla 4.2: Resultados de ’PerseumAI’ con las empresas del sector tecnológico

En la tabla 4.3 se muestra cuantos patrones se encontraron por DTW, cuantos por red neuronal y las coincidencias entre ambos resultados

Nº DTW	Nº RN	Nº Repetidos	Total Final
638	170	121	687

Tabla 4.3: Patrones repetidos entre los encontrados por DTW y red neuronal sector tecnológico

4.1.2. Sector financiero

Las empresas y resultados del sector financiero fueron:

Ticker	Compañía	Cap. de Mercado
V	Visa Inc.	$537,99 mil millones
JPM	JPMorgan Chase & Co.	$568,91 mil millones
BLK	BlackRock, Inc.	$112,43 mil millones
SAN	Banco Santander, S.A.	$75,59 mil millones
BBVA	Banco Bilbao Vizcaya Argentaria, S.A.	$57,83 mil millones
BBD	Banco Bradesco S.A.	$24,90 mil millones
NDAQ	Nasdaq, Inc.	$34,51 mil millones
DB	Deutsche Bank Aktiengesellschaft	$30,82 mil millones
BBAR	Banco BBVA Argentina S.A.	$2,55 billones
FCF	First Commonwealth Financial Corporation	$1,35 mil millones

Tabla 4.4: Conjunto de empresas del sector financiero

Patrón	%	Nº	% P	Nº P	Dif
Doble techo	60 %	163	58 %	79	69 %
Doble suelo	53 %	115	56 %	79	37 %
Triángulo ascendente	43 %	118	36 %	11	166 %
Triángulo descendente	42 %	97	25 %	4	184 %
Hombro cabeza hombro invertido	69 %	62	77 %	35	55 %
Hombro cabeza hombro	69 %	35	47 %	15	80 %

Tabla 4.5: Resultados de 'PerseumAI' con las empresas del sector financiero

Nº DTW	Nº RN	Nº Repetidos	Total Final
555	92	57	590

Tabla 4.6: Patrones repetidos entre los encontrados por DTW y red neuronal sector financiero

4.1.3. Análisis de resultados

En el Apéndice A se muestran algunas imágenes de los patrones encontrados en el experimento separados por su origen (red

neuronal o DTW), y los solapados, es decir que encontro tanto la red neuronal como DTW. En el enlace del Apéndice A.1 esta subido junto al ejecutable del programa las imágenes de todos los patrones, en caso de que se quieran estudiar.

Gracias a los cambios implementados este año se encuentran muchos más patrones, de media un 103 % más. Los tipos de patrones que más se ha mejorado su detección han sido los triángulos, con un aumento de más del 150 % de media. Es interesante notar que el porcentaje de cumplimiento de los patrones, al contrario que el año pasado cuando nos encontrábamos con porcentajes muy altos o muy bajos debido a la poca cantidad de patrones, se ha estabilizado entre el 60-70 % con excepción de los triángulos que están entre 40-50 %.

Especialmente interesante es el caso de los doble techo y doble suelo, para ambos sectores, ambos tipos de patrón tienen un porcentaje de cumplimiento de alrededor del 60 %, siendo estos tipos de patrón sencillos en el que la calidad no es un problema y teniendo el mayor número de instancias encontradas. Este porcentaje de cumplimiento, junto con la abundancia de instancias detectadas, puede generar confianza entre los inversores, ya que se podría considerar como evidencia empírica de que estos patrones son indicadores fiables de tendencias futuras en el mercado.

En el caso del hombro-cabeza-hombro y su versión invertida de acuerdo con los datos obtenidos, el porcentaje de cumplimiento de estos patrones se encuentra alrededor del 70 %, lo cual es una indicación positiva de su precisión. Sin embargo, el número total de patrones encontrados es relativamente bajo en comparación con otros patrones más simples como el doble techo o el doble suelo. Esto puede atribuirse a la naturaleza intrínseca del patrón de hombro-cabeza-hombro, que requiere una serie de movimientos específicos del precio para formarse más complejos.

Respecto a los patrones triángulo ascendente y descendente, aunque se ha logrado un significativo aumento en el número de pa-

trones detectados, el porcentaje de cumplimiento se mantiene relativamente bajo, en torno al 40-45 %. Este bajo porcentaje de aceptación podría deberse a varios factores. En primer lugar, la calidad de algunas instancias detectadas puede no ser óptima, lo que afecta negativamente la tasa de éxito. Los triángulos, por su naturaleza, pueden ser confusos y difíciles de identificar con precisión, y una detección inexacta puede llevar a resultados menos fiables. En segundo lugar, persiste el problema de confundir los triángulos con cuñas, que son patrones similares pero con implicaciones opuestas para el precio objetivo. Las cuñas, al contrario de los triángulos, tienden a señalar reversiones en la tendencia en lugar de continuación, y esta confusión puede resultar en patrones identificados incorrectamente como triángulos, afectando así la tasa de cumplimiento. Además, si recordamos los resultados del experimento en el que se comparaban la red neuronal con DTW (Tabla 3.3), veíamos que al red detectaba en su mayoría triángulos, y revisando las imágenes de los patrones encontrados por la red parece que la calidad de dichos triángulos no es muy buena, lo que puede reducir aún más el porcentaje de cumplimiento.

En cuanto a si la red neuronal encuentra patrones diferentes respecto a los de DTW, para el sector tecnológico tenemos que de los 170 que encontró con red neuronal, 121 los encontró también DTW, es decir un 71 %, mientras que para el sector financiero de los 92 encontrados por la red 57 eran repetidos, el 62 %. Podemos llegar a la conclusión que para estos datos la red encuentra algunos patrones que no encuentra con DTW, pero teniendo él cuenta el gran número de patrones encontrados con DTW respecto a la red son muy pocos.

Un aspecto a aclarar es que debido a la naturaleza del programa cuando se usa DTW estos resultados no son replicables de forma idéntica, si volviéramos a ejecutar el experimento veríamos que el número de patrones encontrados y porcentajes de cumplimiento no serían exactamente los mismos, aunque generalmente seguirían en la misma línea, con una variación máxima en el porcentaje de cumplimiento de alrededor del 8-10 puntos porcentuales. Estos

valores son basados en mi experiencia con la herramienta.

4.2. Selección de parámetros

En esta sección se van a comentar los diferentes parámetros del programa que se han usado para el backtesting, tanto los que selecciona el usuario como los internos.

Para empezar con los parámetros introducidos en la herramienta tenemos los ya comentados, seleccionamos todos los tipos de patrones, un periodo de fechas entre el 1 de enero 2005 y el 7 de julio 2023 y los ficheros de texto con las empresas por sectores. Además, si recordamos la nueva interfaz del programa (Figura 3.18) todavía no se ha discutido la elección del tipo y modo de búsqueda. Para el tipo de búsqueda se usó la búsqueda muy profunda o "Very deep search", como su nombre indica realiza una búsqueda exahustiva examinando tamaños de ventana entre 80 y 260 para asegurar que se obtienen la mayor cantidad de patrones posibles. En el modo de búsqueda se seleccionó DTW y Red neuronal por el mismo motivo.

Por otro lado tenemos los parámetros internos del programa, ya explicados en el capítulo 3, como la distancia para filtrar que ahora depende del tipo de patrón o el número de patrones a cargar, que por defecto es 15 como se explica en la sección 3.4.1.

4.3. Análisis de errores, limitaciones y posibles mejoras

4.3.1. Calidad de los triángulos

Un problema que se arrastra de la versión anterior y que no se ha conseguido mejorar del todo es la calidad de los triángulos ascendentes y descendentes, no los detecta bien tanto con DTW como con la red neuronal, hay veces que se detectan como triángulos patrones más parecidos a cuñas cuya tendencia es la contraria. Una solución podría ser revisar el código del filtro para estos tipos de patrón para comprobar si se puede mejorar, o para el caso de la red neuronal revisar la base de datos y volver a

entrenar los modelos de triángulos.

4.3.2. Programa no responde

Otro problema es que si la búsqueda de patrones es muy compleja (gran periodo de fechas, muchas empresas...) la aplicación sigue funcionando, pero aparece como que no responde, lo cual puede ser confunso para el usuario final y es posible que cierre la pestaña antes de que se termine la búsqueda. Esto se debe a que la aplicación hace la búsqueda de patrones en el hilo principal del programa, y no se ha solucionado por falta de tiempo y por no ser uno de los objetivos principales de este TFG.

Capítulo 5

Conclusiones y líneas futuras

5.1. Conclusiones

En conclusión, este trabajo ha consistido en utilizar diferentes estrategias para mejorar la aplicación PerseumAI y que encuentre la mayor cantidad de patrones posibles con la mejor calidad. Además de mejoras en la interfaz y la experiencia del usuario con la aplicación, eliminando la necesidad de introducir parámetros que podían resultar confusos.

Se ha introducido un nuevo modo para detectar patrones, redes neuronales convolucionales, una arquitectura de red neuronal especializada en el reconocimiento de imágenes, y se comparó con el método de búsqueda usado hasta ahora basado en DTW. Para la detección de patrones históricos, DTW es mejor que la red neuronal, encontrando muchos más patrones. Para la detección de patrones actuales sin embargo la red neuronal parece ser mejor, encontrando más variedad de patrones.

Además se realizaron varios estudios para determinar los parámetros más óptimos de la herramienta. Estos experimentos consistían en ejecutar PerseumAI con diferentes valores, almacenar los resultados y compararlos. Se realizó tanto un estudio para determinar los mejores parámetros en cuanto a número de patrones encontrados y en cuanto a calidad de los patrones encontrados.

Se programaron nuevos algoritmos para la búsqueda de patrones

históricos y actuales. El nuevo algoritmo para patrones históricos se basa en explorar la misma gráfica de precios con diferentes tamaños de ventana, para asegurarnos que obtenemos todos los patrones. Mientras que el algoritmo para patrones actuales se basa en estudiar una ventana de tiempo que acaba en el día actual e irla reduciendo progresivamente para encontrar el patrón o el mismo, pero con mayor precisión.

5.2. Líneas de trabajo futuras

Posibles líneas de trabajo futuras para mejorar la herramienta pueden ser:

- Mejorar la calidad de los triángulos, Para abordar este problema se propone revisar el código del filtro encargado de este tipo de patrón.
- Añadir funcionalidades como porcentaje de ganacias de patrones actuales y stop loss adecuados.
- Probar diferentes modelos de deep learning como arquitecturas LSTM o MLP para determinar si mejora la detección de patrones.
- Mejorar la visualización de los patrones y en general que la interfaz sea más profesional.
- Mejorar la base de datos de entrenamiento de la red neuronal y volver a entrenar los modelos.

Capítulo 6

Summary and Conclusions

6.1. Conclusions

In conclusion, this work has involved using different strategies to improve the PerseumAI application to find as many patterns as possible with the best quality. In addition to improvements in the interface and user experience with the application, eliminating the need to enter parameters that could be confusing.

A new mode for pattern detection has been introduced, convolutional neural networks, a neural network architecture specialized in image recognition, and it was compared with the previously used search method based on DTW. For historical pattern detection, DTW is better than the neural network, finding many more patterns. For current pattern detection, however, the neural network seems to be better, finding a greater variety of patterns.

Additionally, several studies were conducted to determine the most optimal parameters for the tool. These experiments consisted of running PerseumAI with different values, storing the results, and comparing them. Both a study to determine the best parameters in terms of the number of patterns found and in terms of the quality of the patterns found were conducted.

New algorithms were programmed for the search of historical and current patterns. The new algorithm for historical patterns is based on exploring the same price chart with different window sizes, to ensure that we obtain all patterns. Meanwhile, the algo-

rithm for current patterns is based on studying a time window that ends on the current day and progressively reducing it to find the pattern or the same pattern but with greater precision.

Capítulo 7

Presupuesto

7.1. Costes materiales

Tipos	Descripción	Coste
Equipo informático	Portátil (GF63 thin 11uc)	950 €

Tabla 7.1: Costes materiales

Tipos	Horas	Coste/Hora	Total
Investigación	20	30 €	600 €
Familiarización con el código existente	10	35 €	350 €
Mejora de la búsqueda de patrones históricos y actuales	140	40 €	5600 €
Desarrollo e implementación de la red neuronal	30	40 €	1200 €
Desarrollo de la aplicación final	25	40 €	1000 €
Desarrollo de la memoria	20	35 €	700 €
Total	245	~36.6 €	9450 €

Tabla 7.2: Tabla de costes de trabajo

Apéndice A

Elementos adicionales

A.1. Enlaces al programa

Repositorio Github (Código)
Google Drive (Aplicación).

A.2. Fuentes de datos de las tablas del capítulo 4

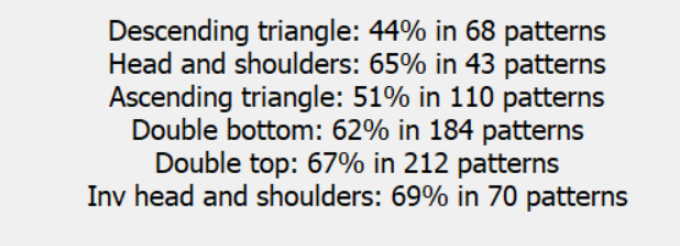


Figura A.1: Patrones encontrados con datos tecnológicos

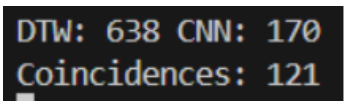


Figura A.2: Coincidencias datos tecnológicos

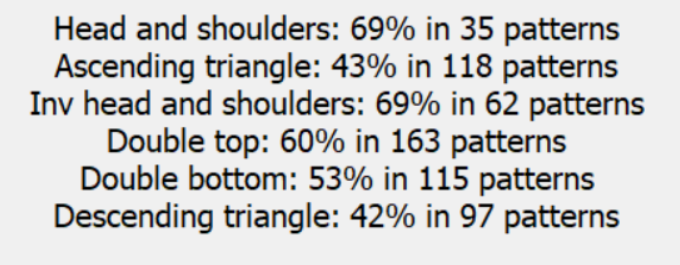


Figura A.3: Patrones encontrados con datos financieros

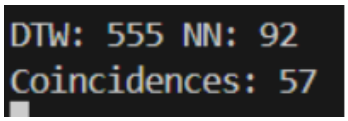
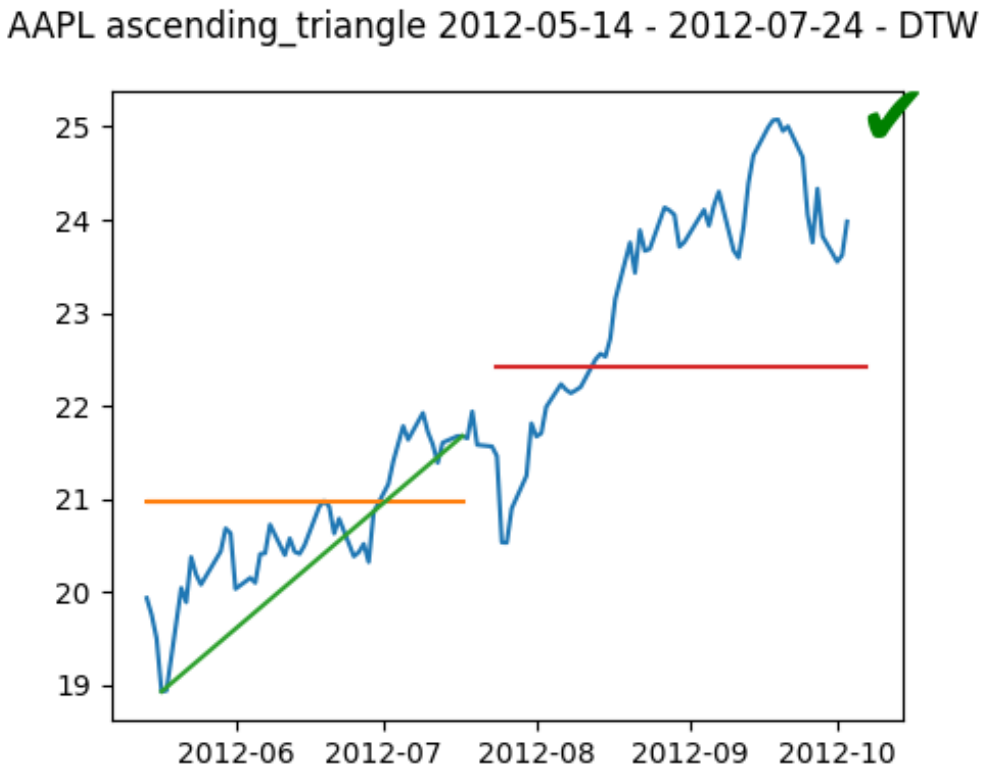
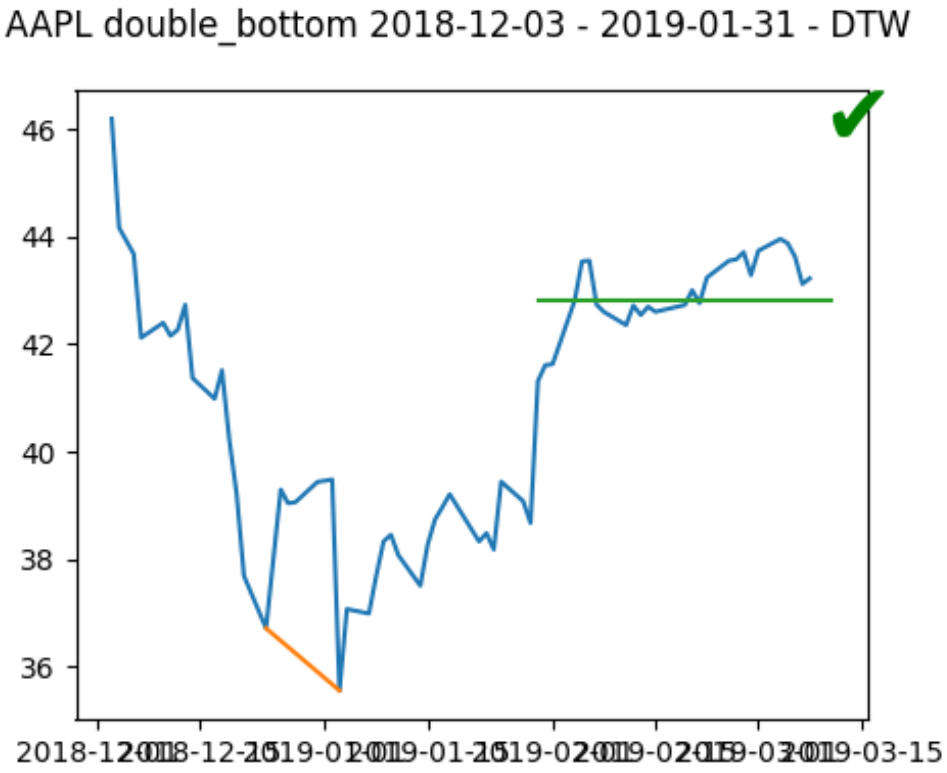
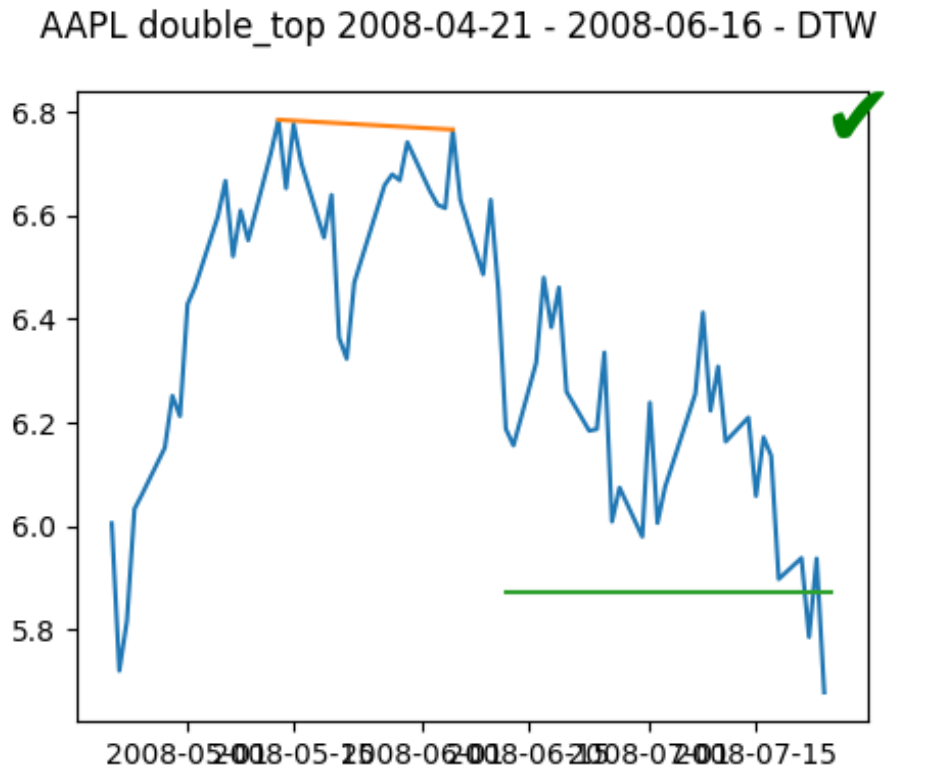


Figura A.4: Coincidencias datos financieros

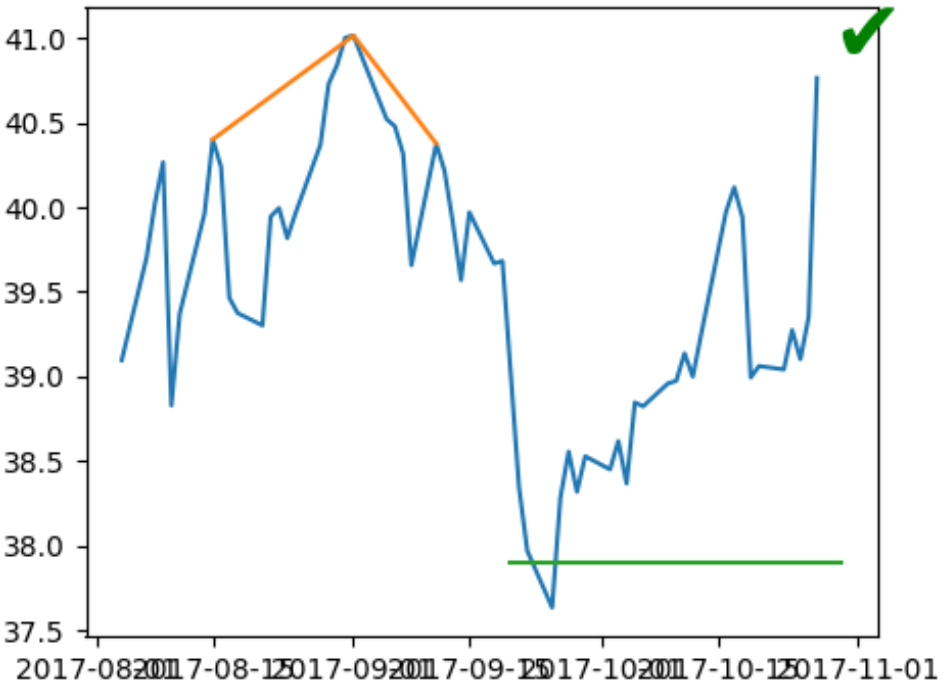
A.3. Resultados DTW sector tecnológico



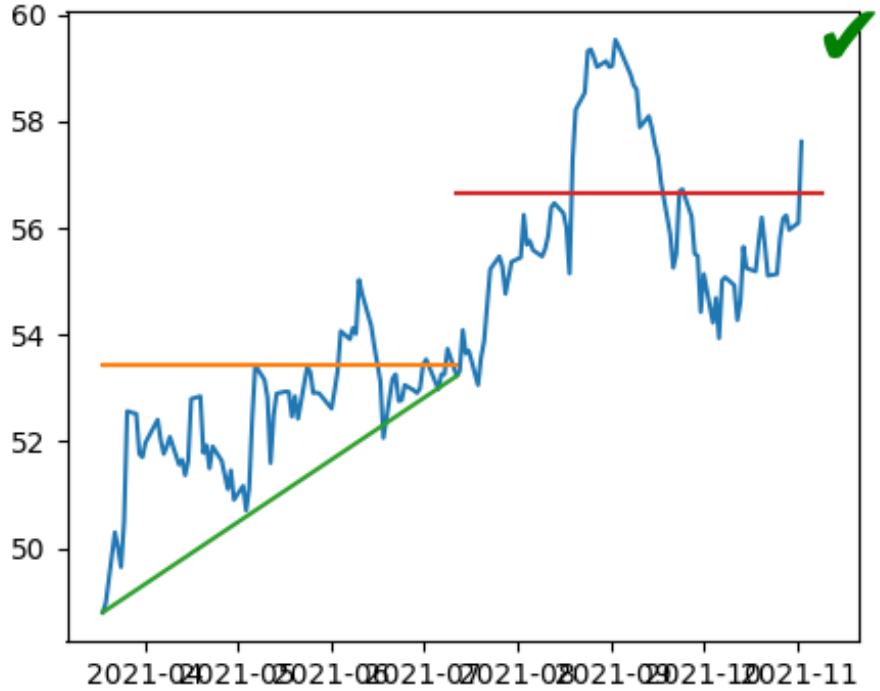

AAPL double_bottom 2018-12-03 - 2019-01-31 - DTW
46
44
42
40
38
36
2018-12-01
2018-12-15
2019-01-01
2019-01-15
2019-02-01
2019-02-15
2019-03-01
2019-03-15

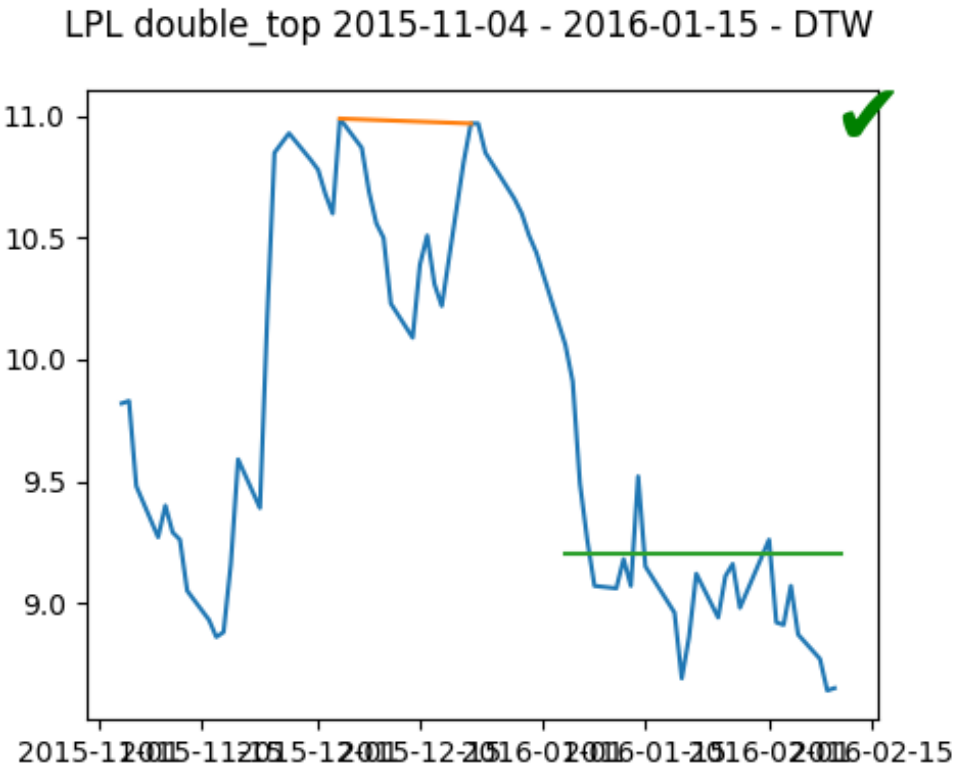

AAPL double_top 2008-04-21 - 2008-06-16 - DTW
6.8
6.6
6.4
6.2
6.0
5.8
2008-05-01
2008-05-15
2008-06-01
2008-06-15
2008-07-01
2008-07-15

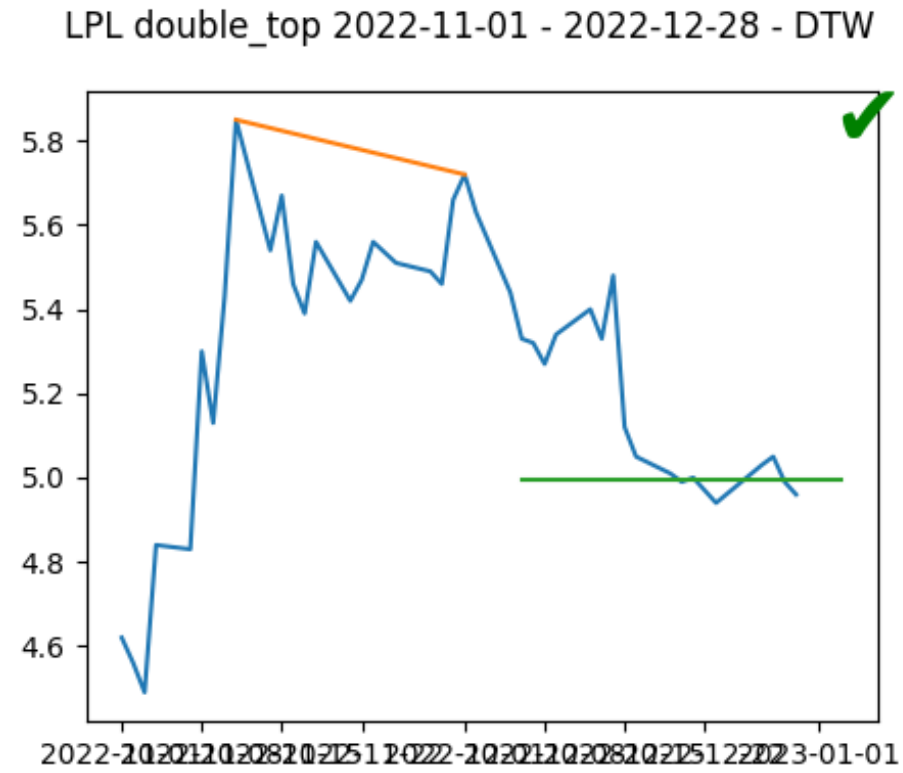
AAPL head_and_shoulders 2017-08-04 - 2017-09-29 - DTW

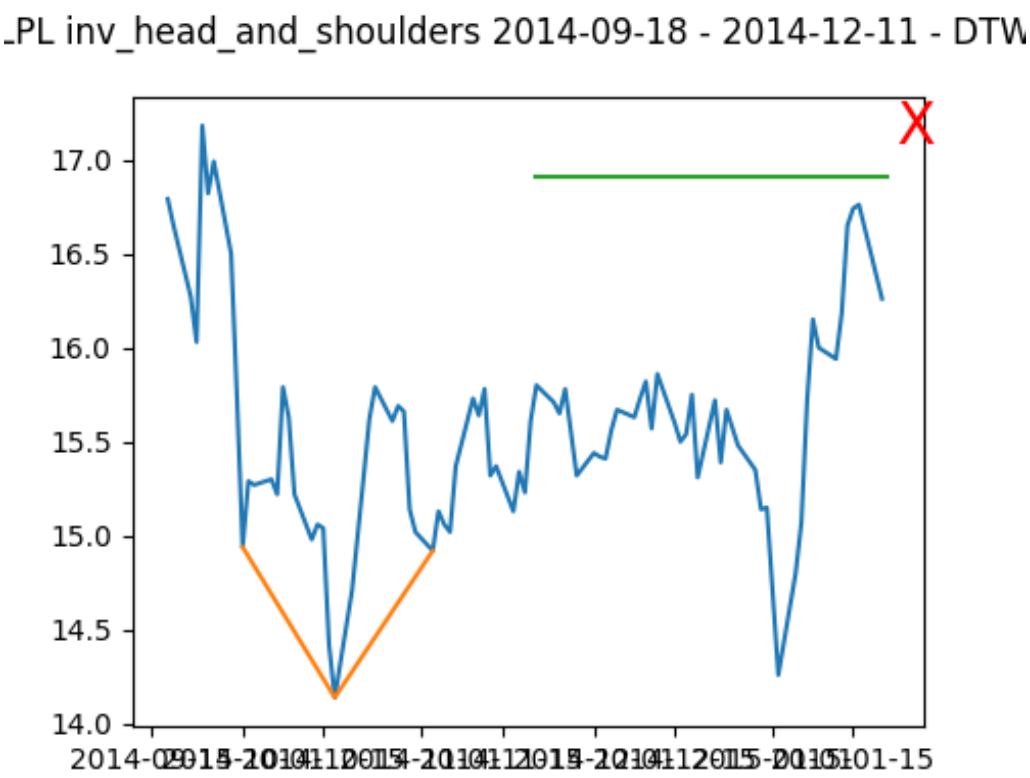


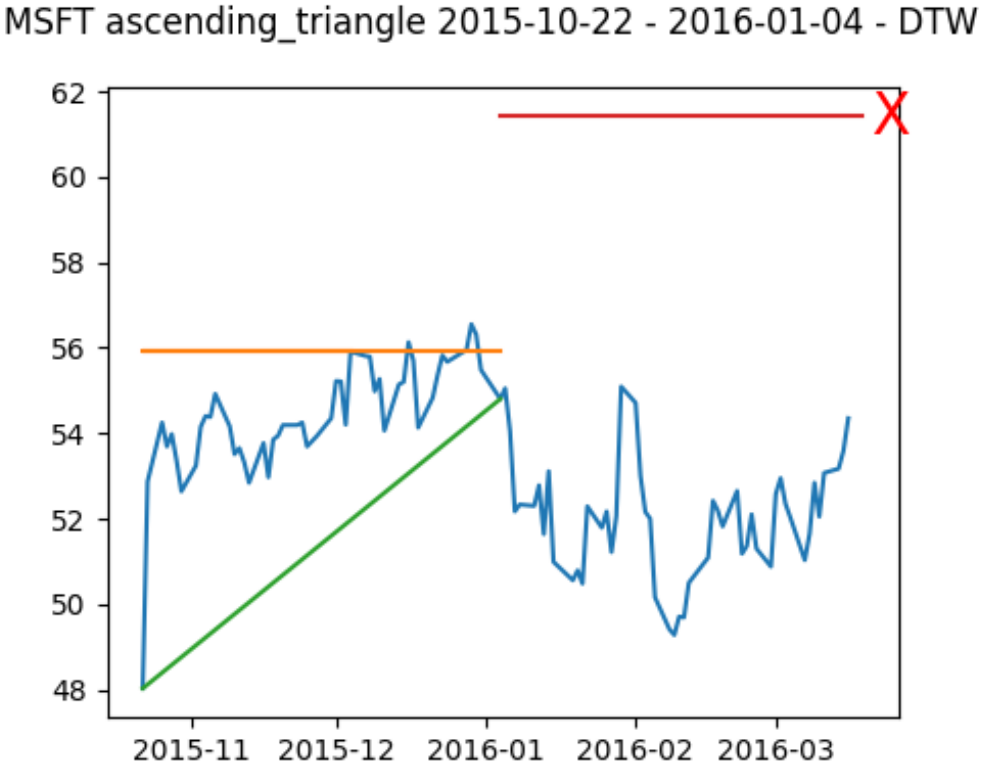
CSCO ascending_triangle 2021-03-18 - 2021-07-12 - DTW



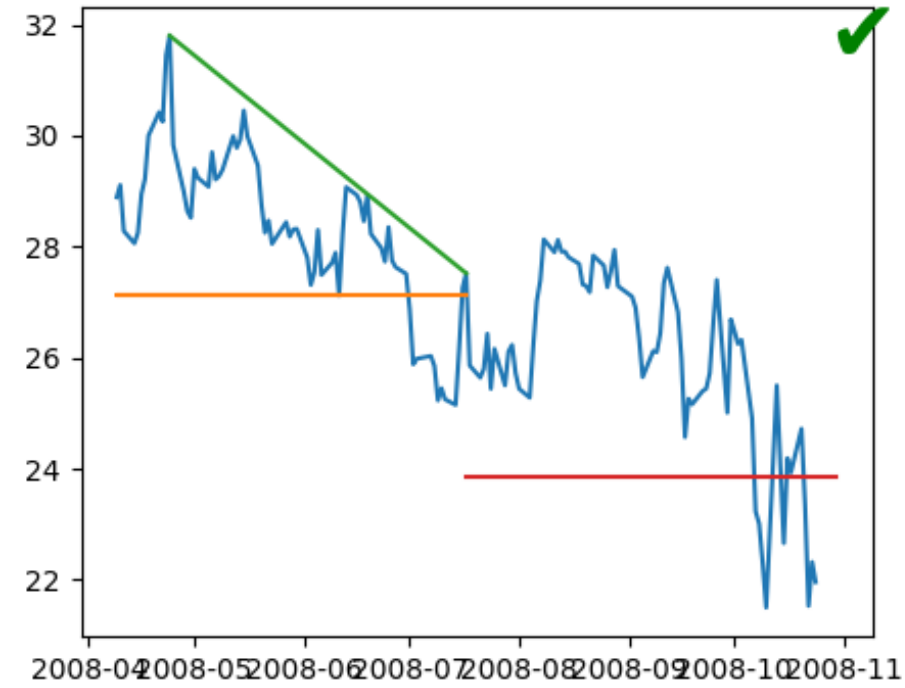

LPL double_top 2015-11-04 - 2016-01-15 - DTW
11.0
10.5
10.0
9.5
9.0


LPL double_top 2022-11-01 - 2022-12-28 - DTW
5.8
5.6
5.4
5.2
5.0
4.8
4.6

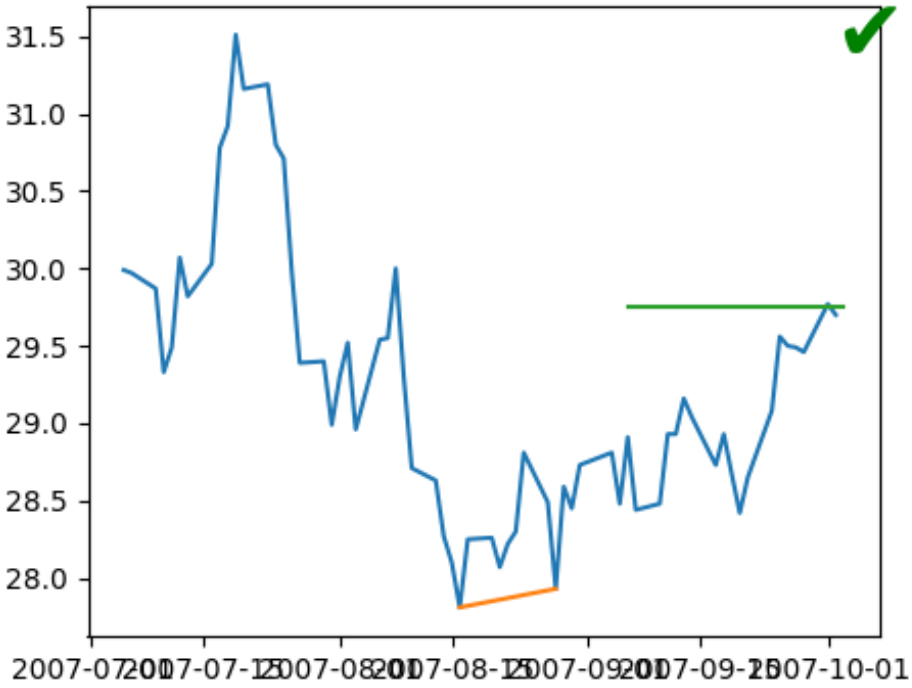

_PL inv_head_and_shoulders 2014-09-18 - 2014-12-11 - DTW
17.0
16.5
16.0
15.5
15.0
14.5
14.0

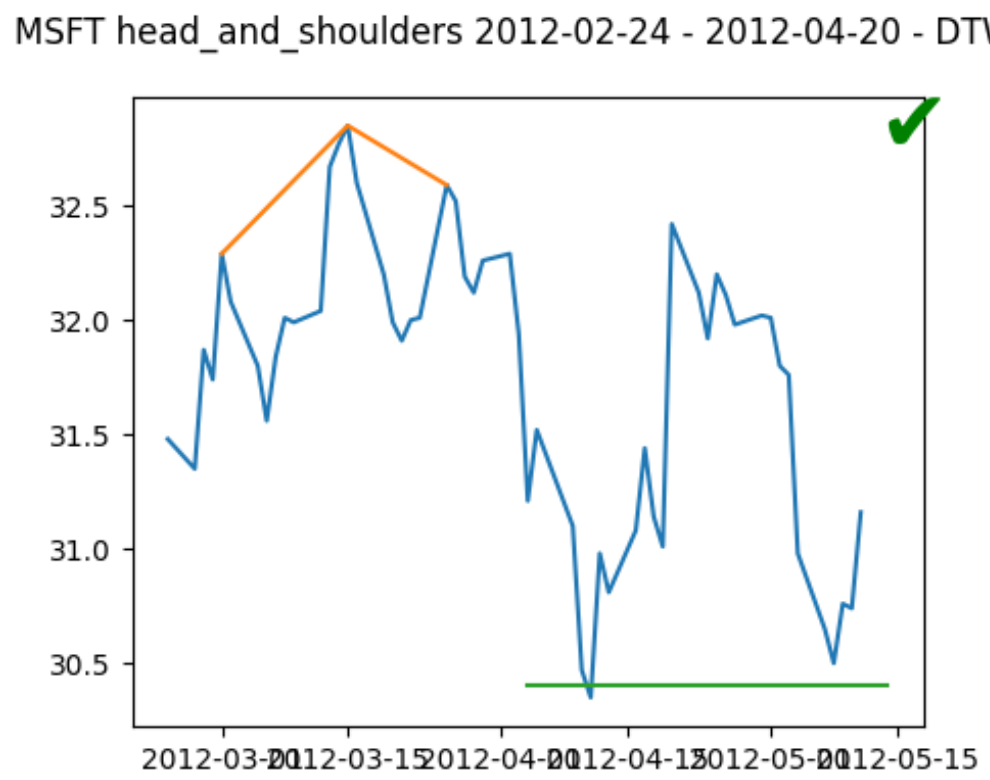

MSFT ascending_triangle 2015-10-22 - 2016-01-04 - DTW
62
60
58
56
54
52
50
48
2015-11
2015-12
2016-01
2016-02
2016-03

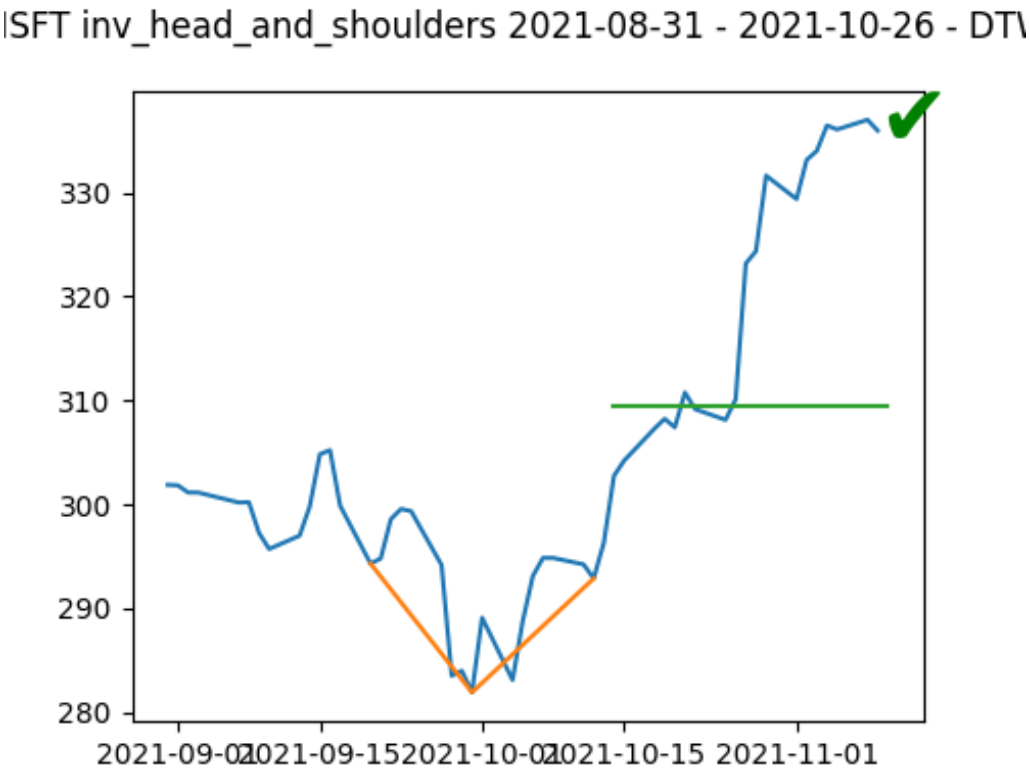
MSFT descending_triangle 2008-04-09 - 2008-07-17 - DTW

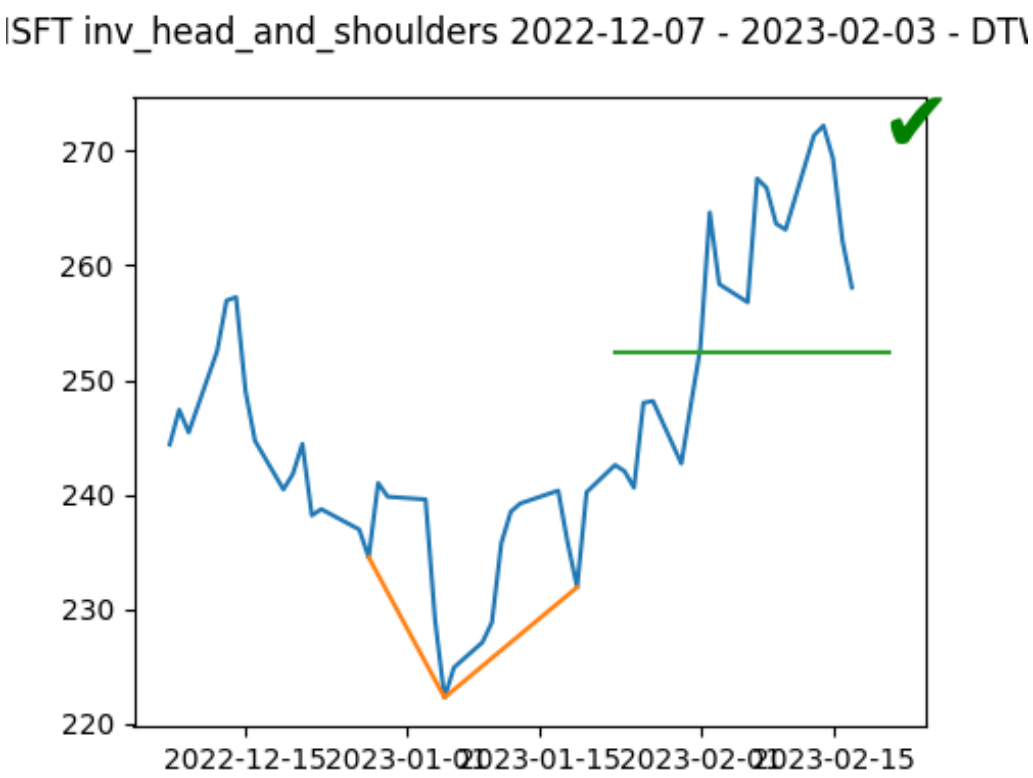


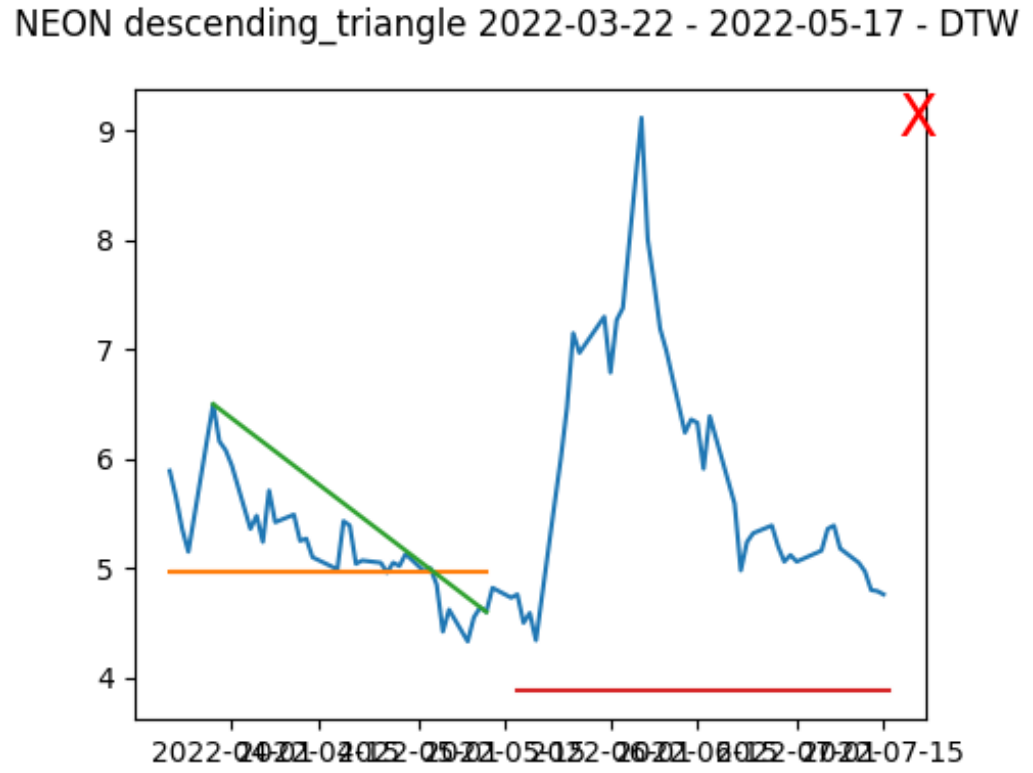
MSFT double_bottom 2007-07-05 - 2007-09-13 - DTW

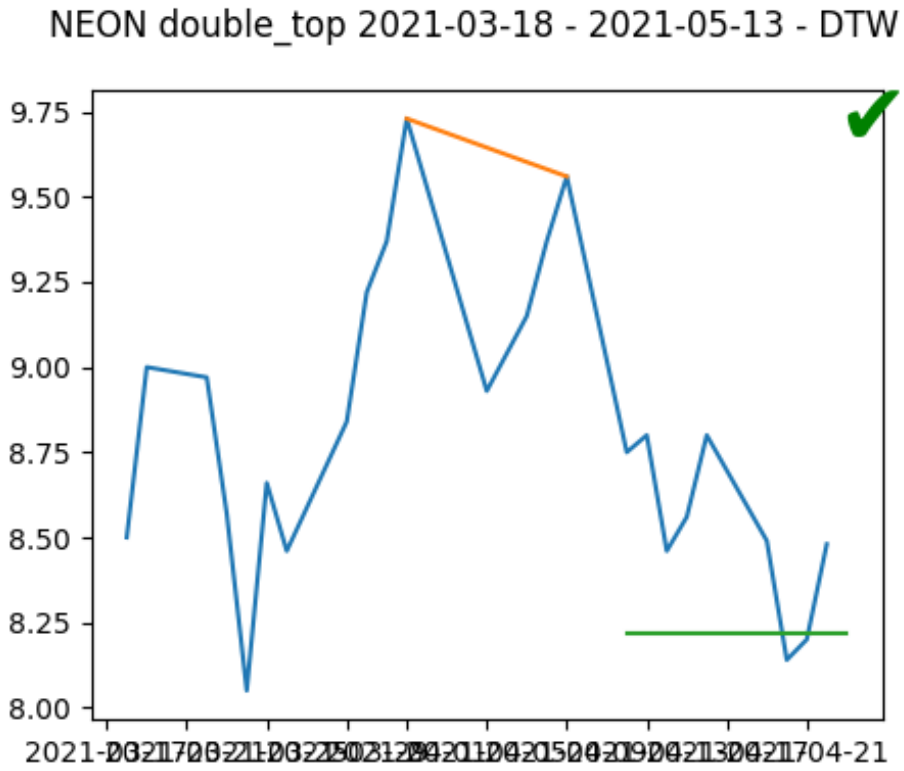


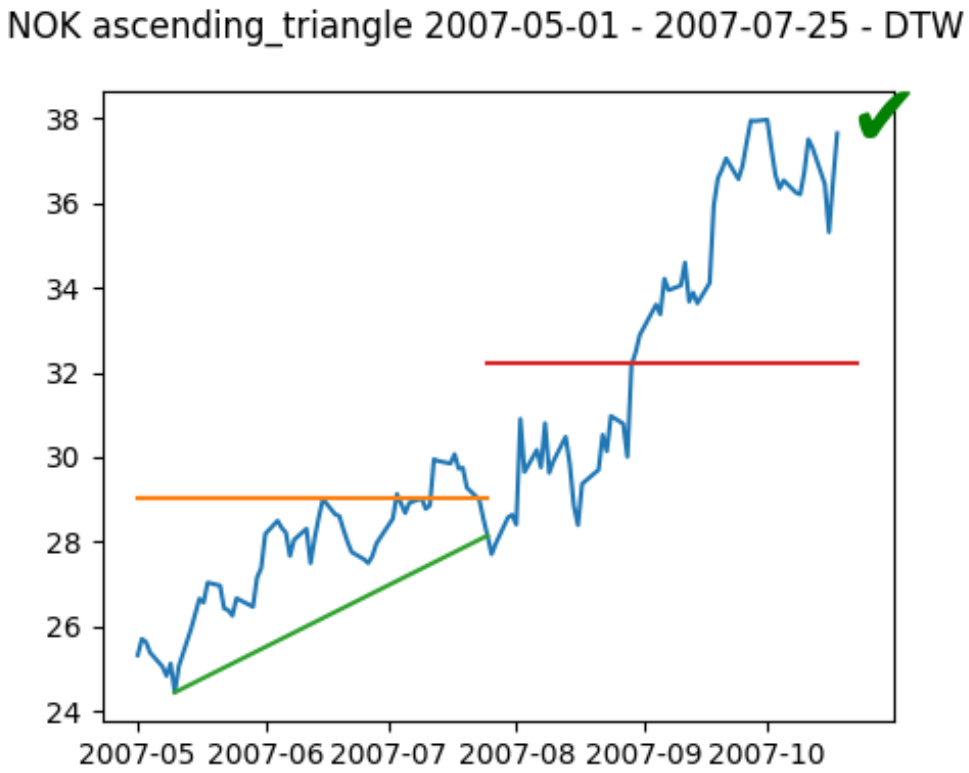

MSFT head_and_shoulders 2012-02-24 - 2012-04-20 - DTW
32.5
32.0
31.5
31.0
30.5
2012-03-01
2012-03-15
2012-04-01
2012-04-15
2012-05-01
2012-05-15


ISFT inv_head_and_shoulders 2021-08-31 - 2021-10-26 - DT\
330
320
310
300
290
280
2021-09-01
2021-09-15
2021-10-01
2021-10-15
2021-11-01

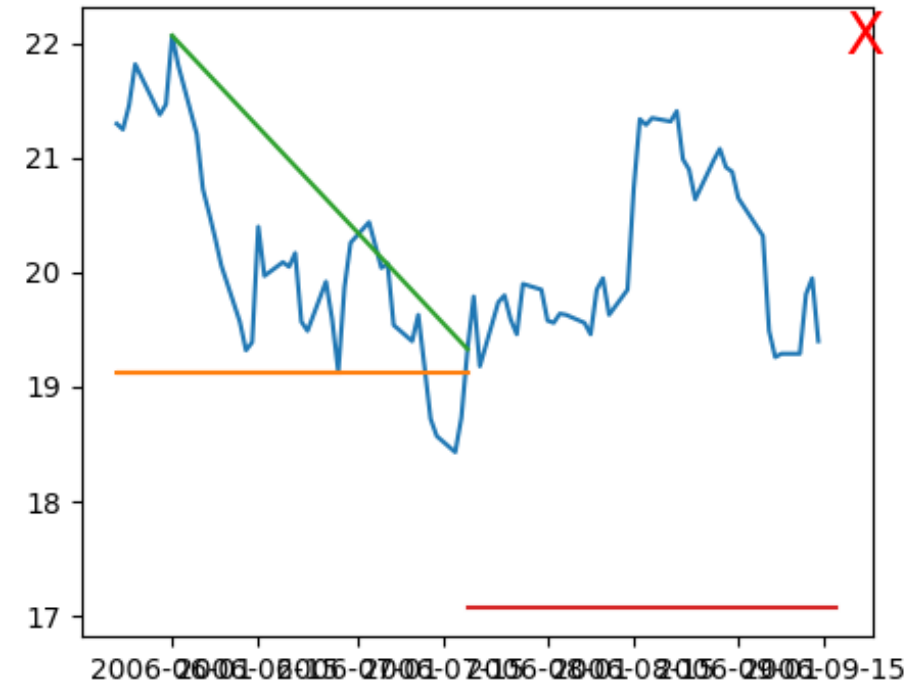

ISFT inv_head_and_shoulders 2022-12-07 - 2023-02-03 - DT\
270
260
250
240
230
220
2022-12-15
2023-01-01
2023-01-15
2023-02-01
2023-02-15


NEON descending_triangle 2022-03-22 - 2022-05-17 - DTW
9
8
7
6
5
4
2022-07-15

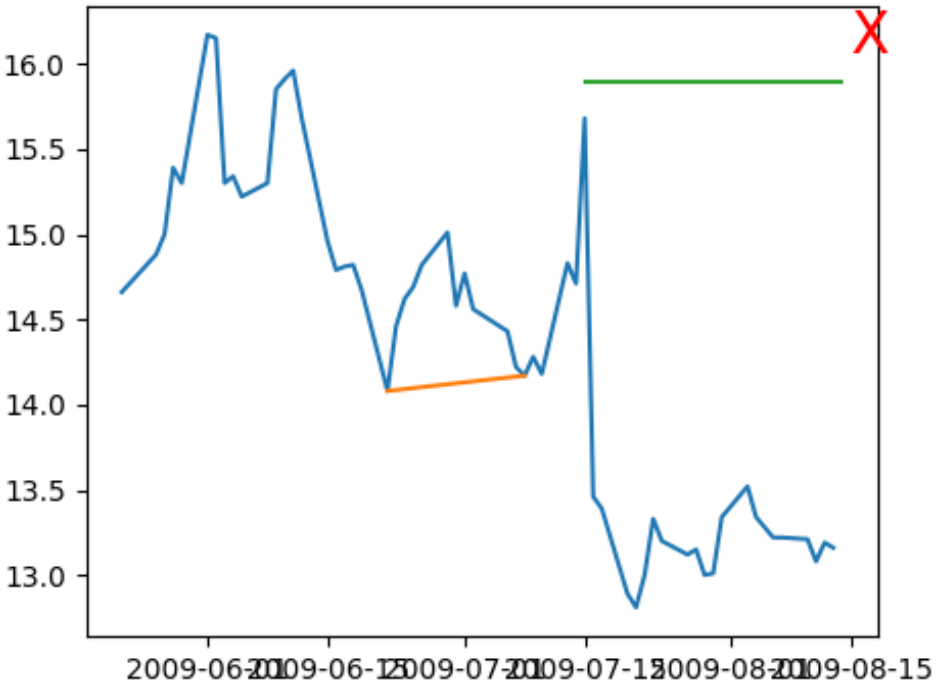

NEON double_top 2021-03-18 - 2021-05-13 - DTW
9.75
9.50
9.25
9.00
8.75
8.50
8.25
8.00

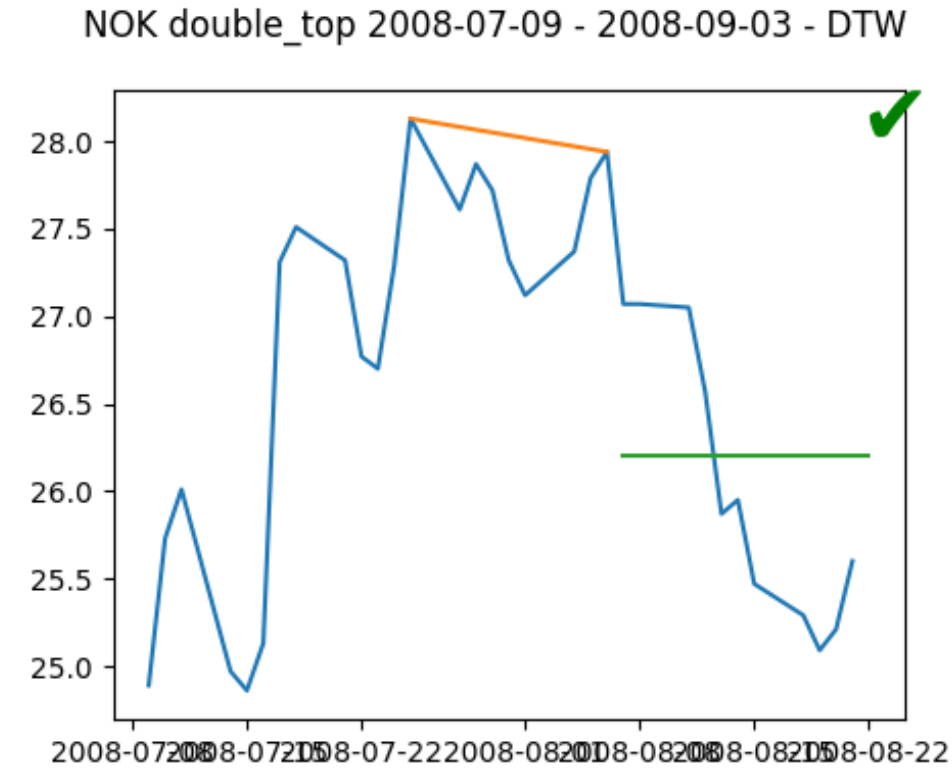

NOK ascending_triangle 2007-05-01 - 2007-07-25 - DTW
38
36
34
32
30
28
26
24
2007-05
2007-06
2007-07
2007-08
2007-09
2007-10

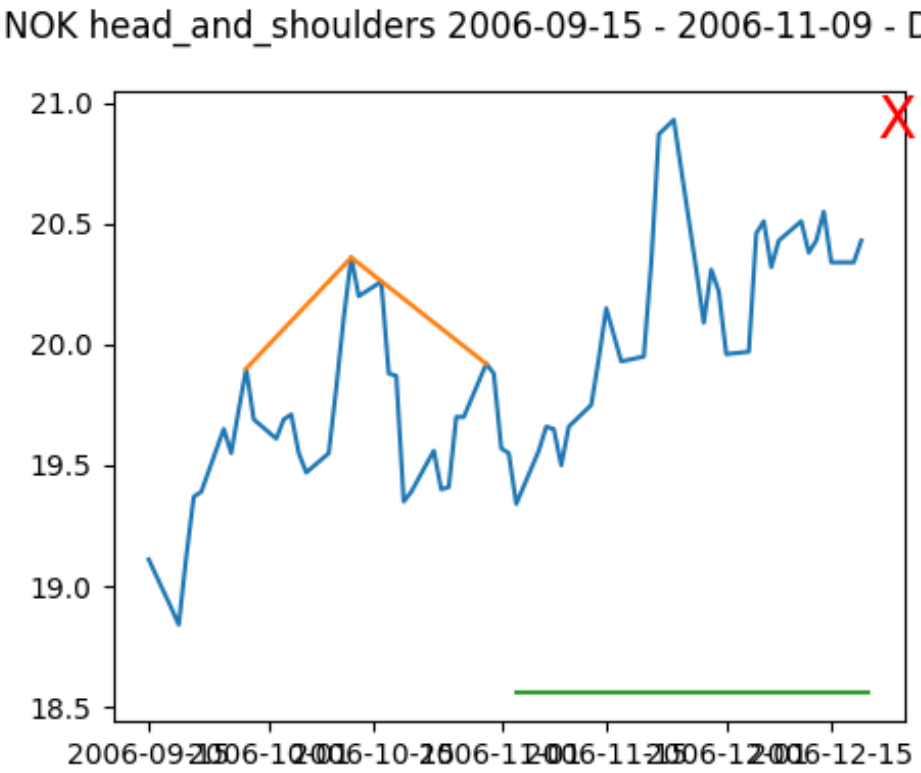
NOK descending_triangle 2006-05-23 - 2006-07-19 - DTW



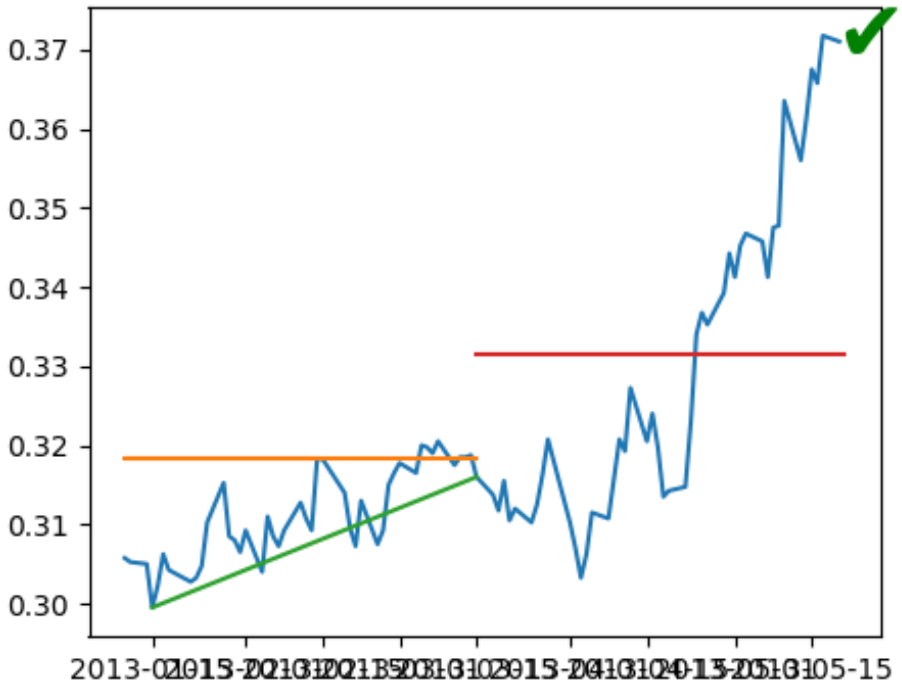
NOK double_bottom 2009-05-22 - 2009-07-20 - DTW



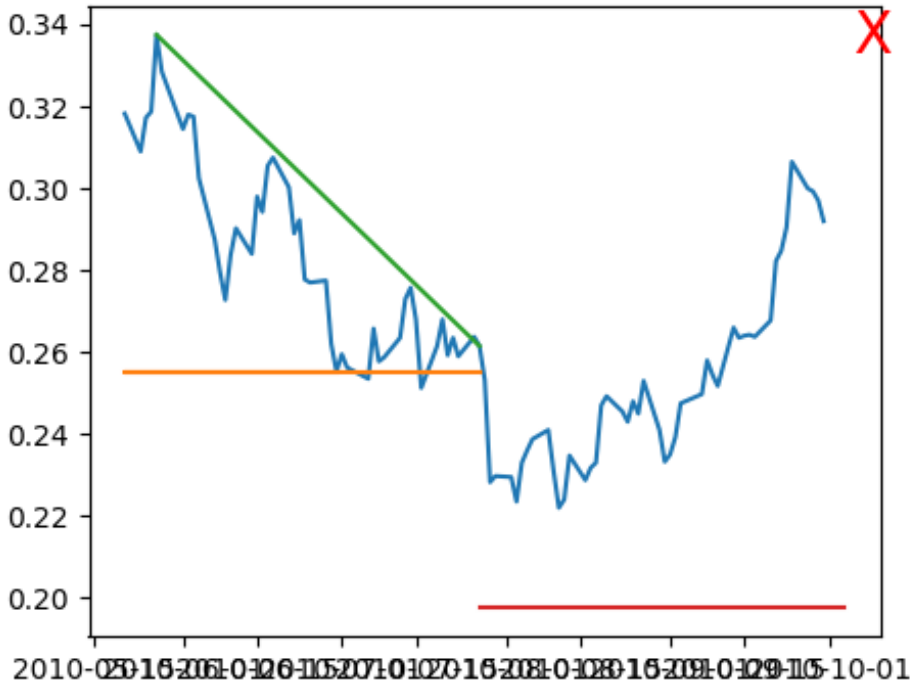

NOK double_top 2008-07-09 - 2008-09-03 - DTW
28.0
27.5
27.0
26.5
26.0
25.5
25.0
2008-07-22
2008-08-01
2008-08-22

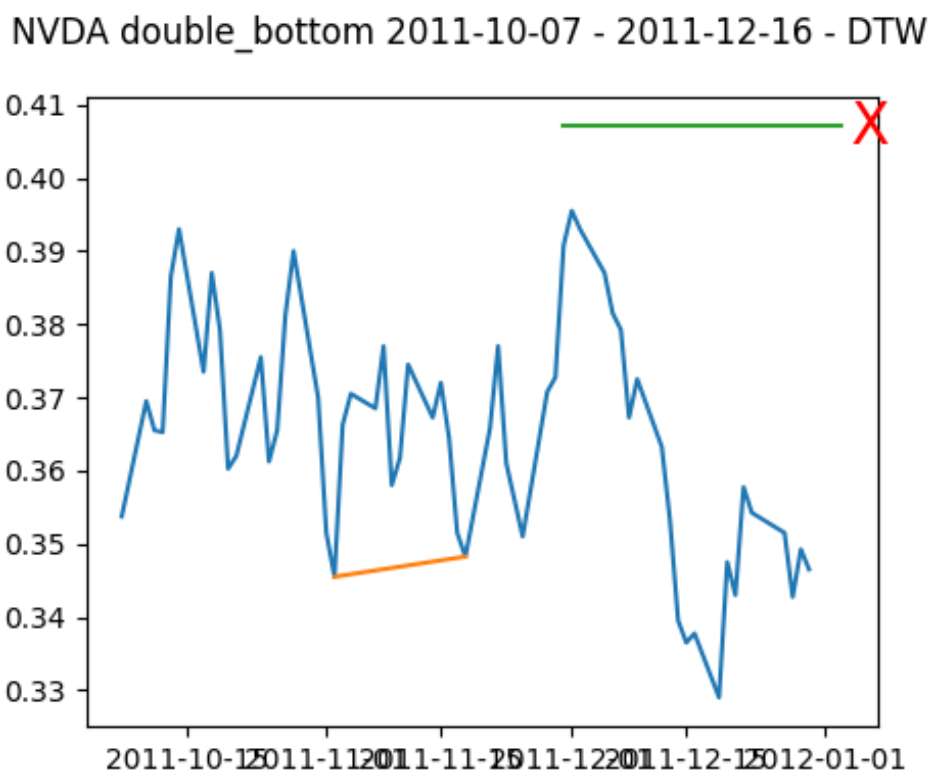

NOK head_and_shoulders 2006-09-15 - 2006-11-09 - DTW
21.0
20.5
20.0
19.5
19.0
18.5
2006-10-16
2006-11-15
2006-12-15

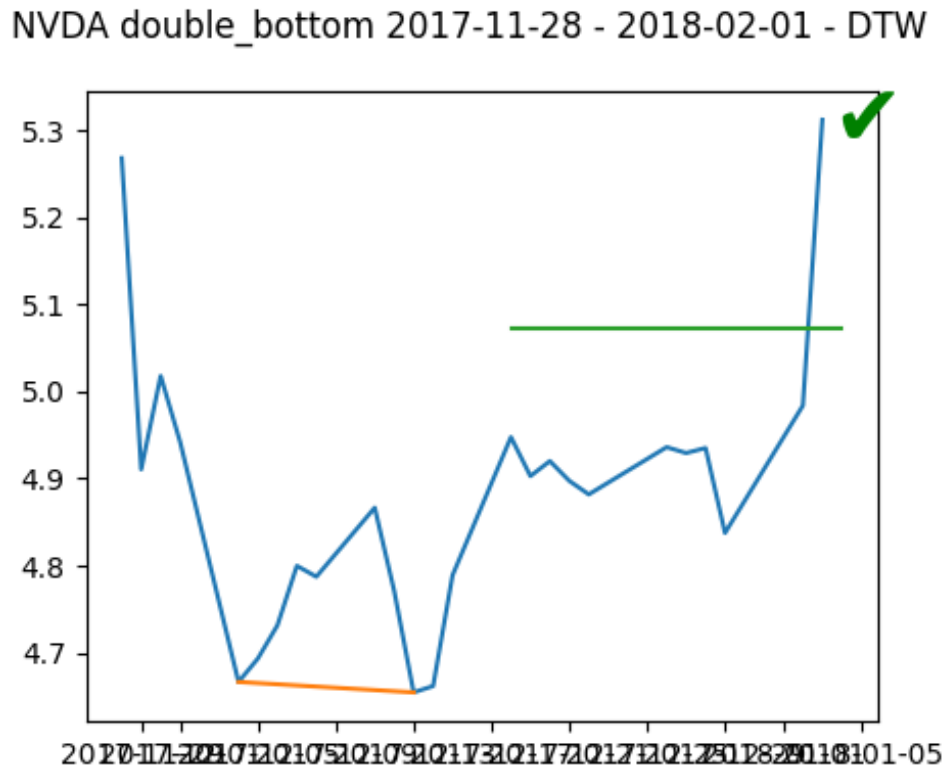
NVDA ascending_triangle 2013-01-10 - 2013-03-15 - DTW



NVDA descending_triangle 2010-05-21 - 2010-07-27 - DTW

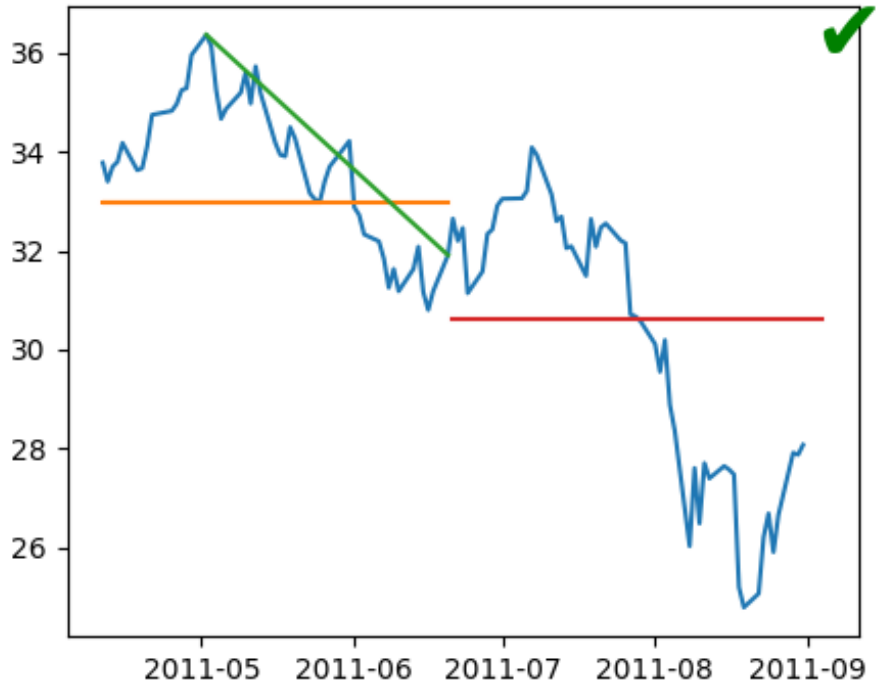



NVDA double_bottom 2011-10-07 - 2011-12-16 - DTW
0.41
0.40
0.39
0.38
0.37
0.36
0.35
0.34
0.33
2012-01-01

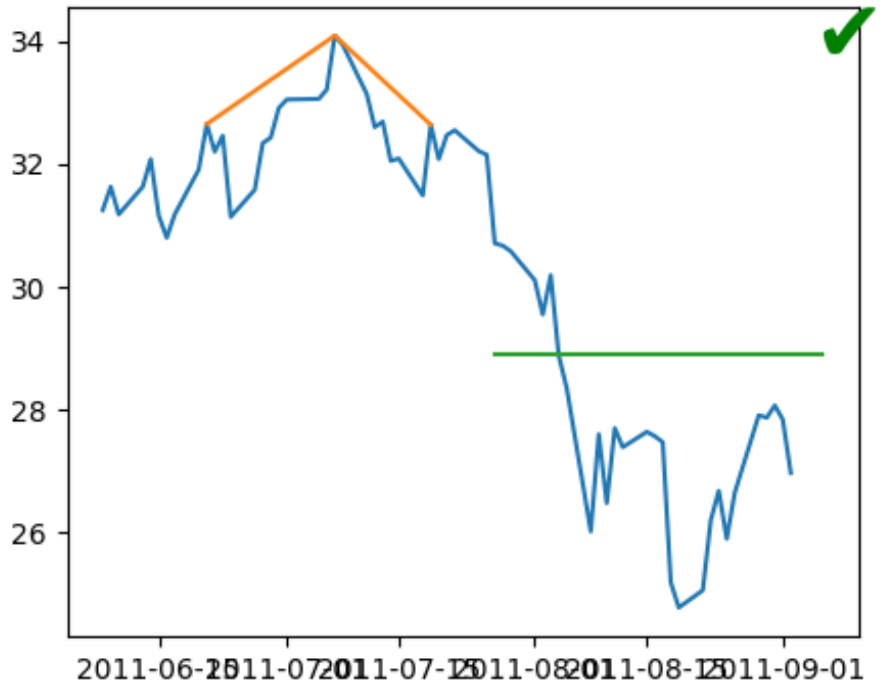

NVDA double_bottom 2017-11-28 - 2018-02-01 - DTW
5.3
5.2
5.1
5.0
4.9
4.8
4.7

VDA inv_head_and_shoulders 2009-09-16 - 2009-11-17 - DT

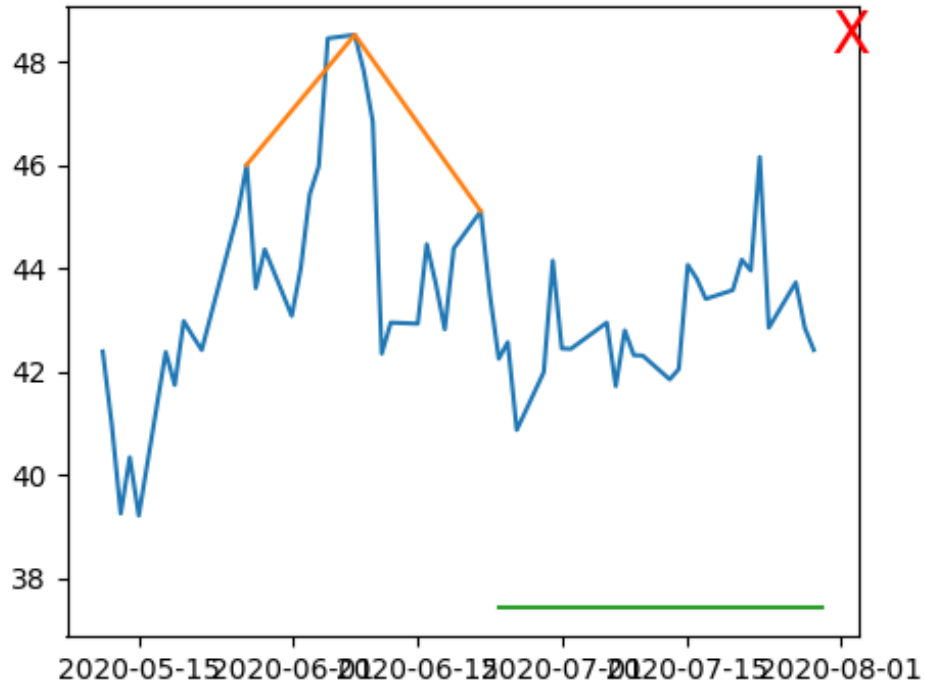
ORCL descending_triangle 2011-04-11 - 2011-06-21 - DTW

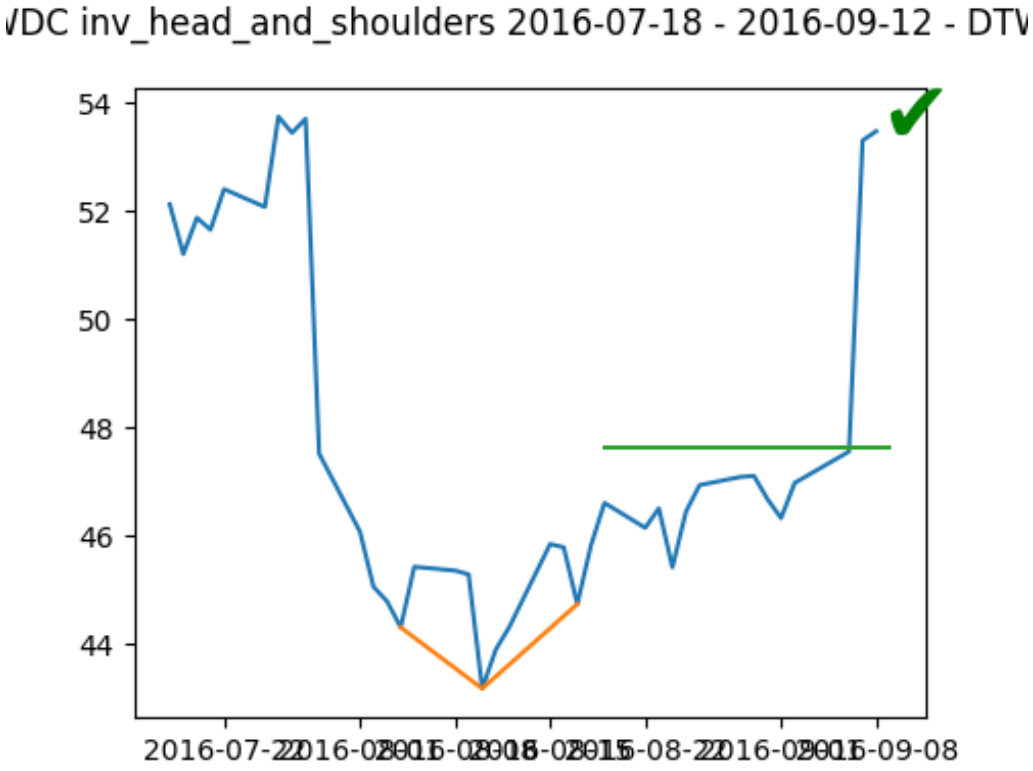


ORCL head_and_shoulders 2011-06-08 - 2011-08-03 - DTW

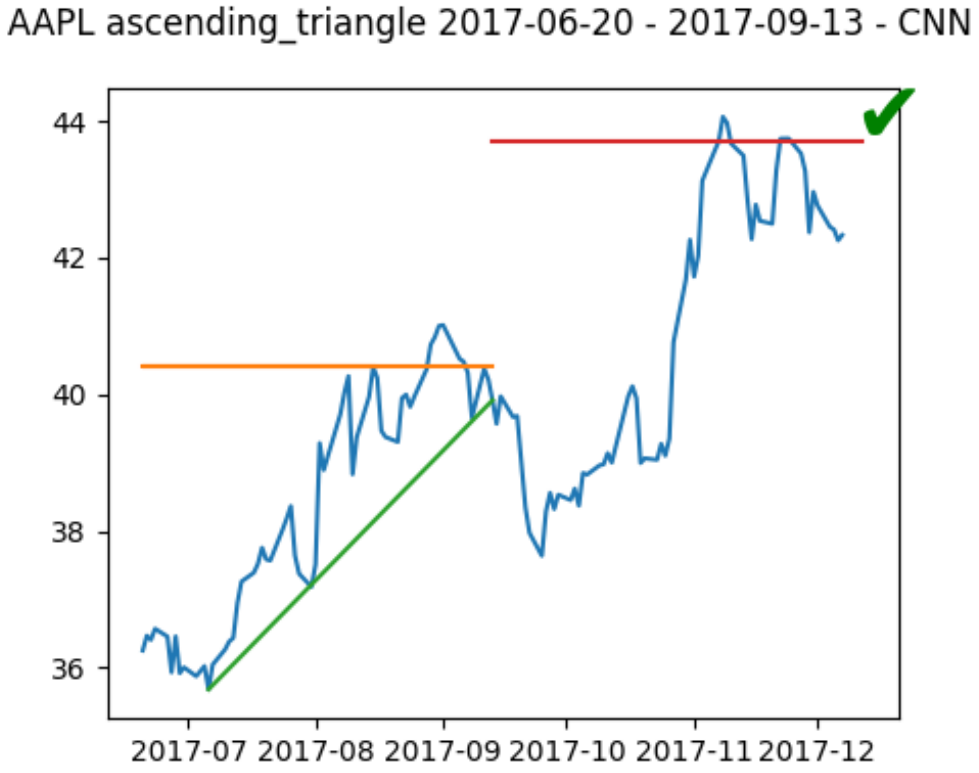


WDC head_and_shoulders 2020-05-11 - 2020-07-07 - DTW

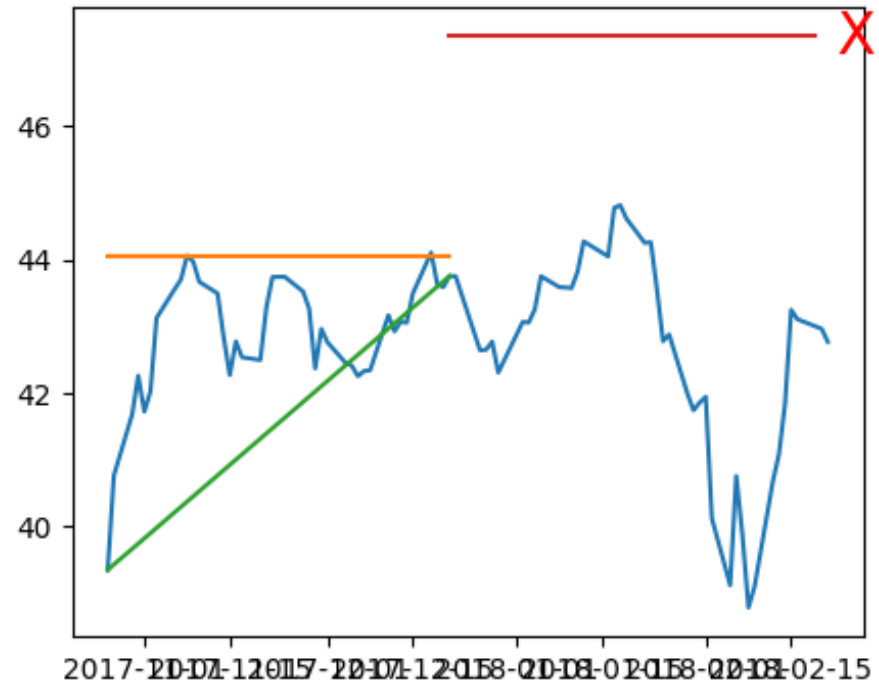




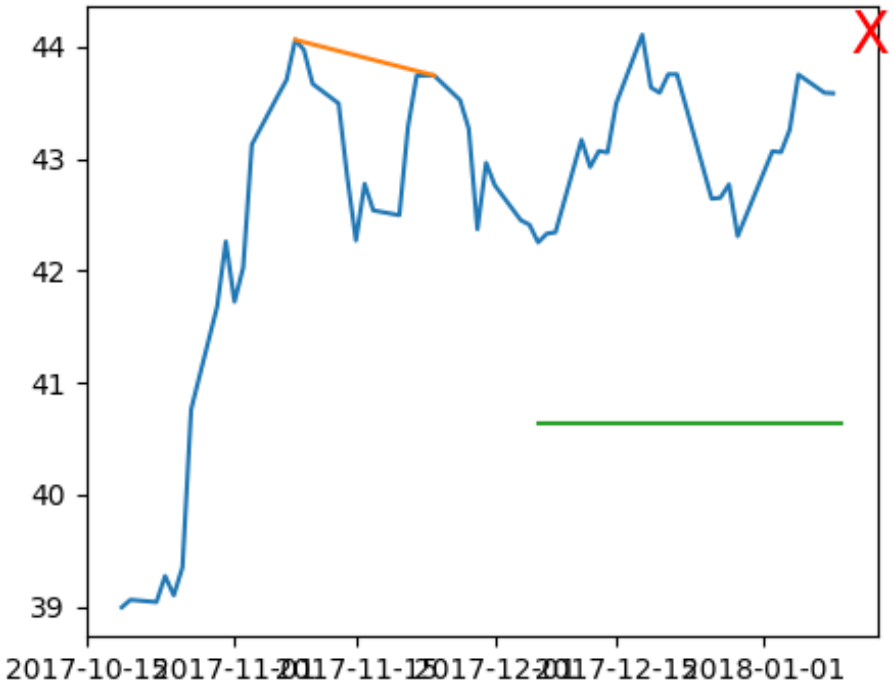
A.4. Resultados red neuronal sector tecnológico



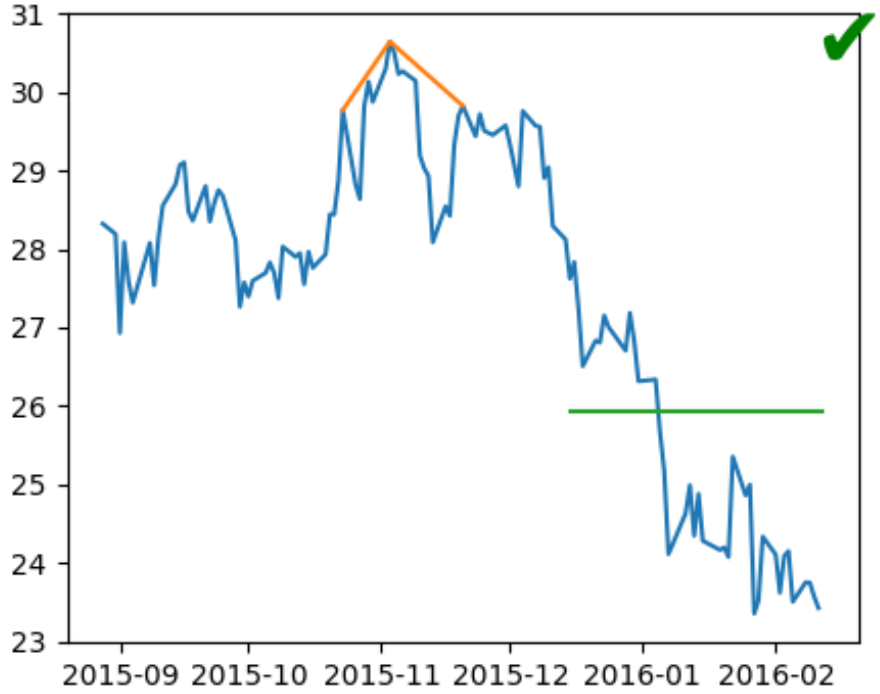
AAPL ascending_triangle 2017-10-26 - 2017-12-21 - CNN



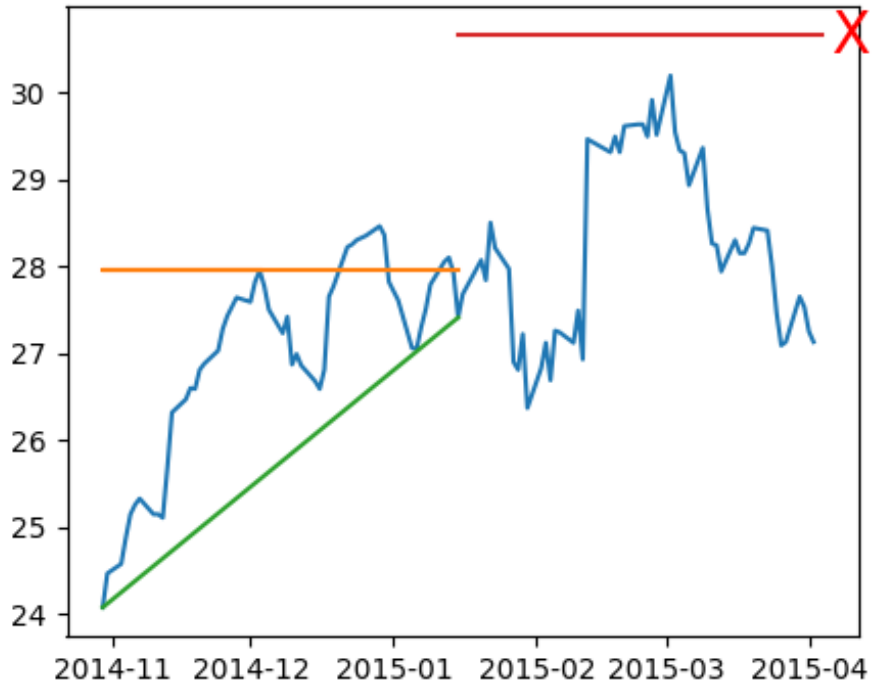
AAPL double_top 2017-10-19 - 2017-12-14 - CNN



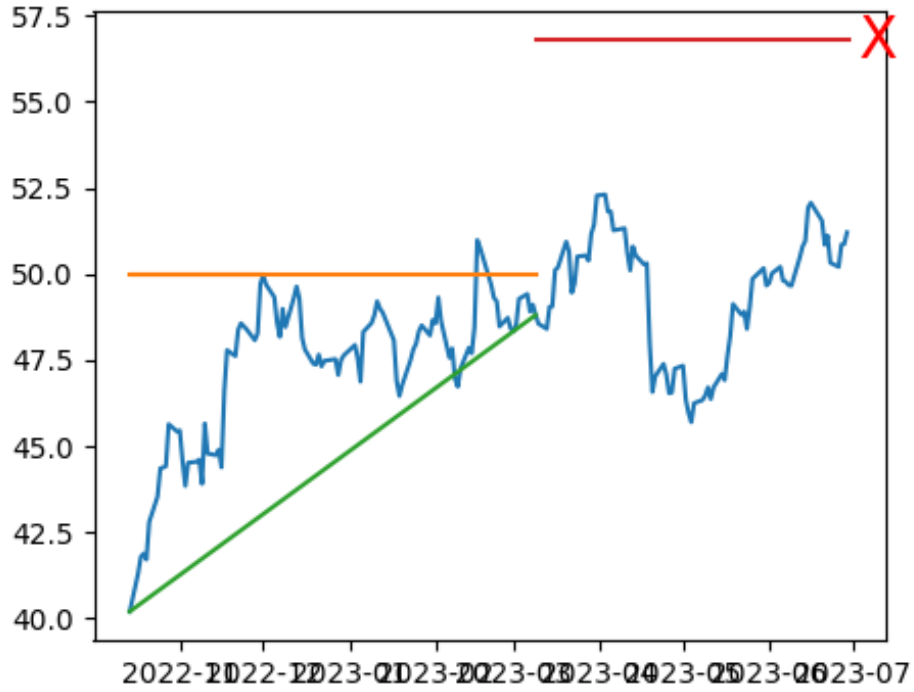
AAPL head_and_shoulders 2015-08-28 - 2015-12-21 - CNN



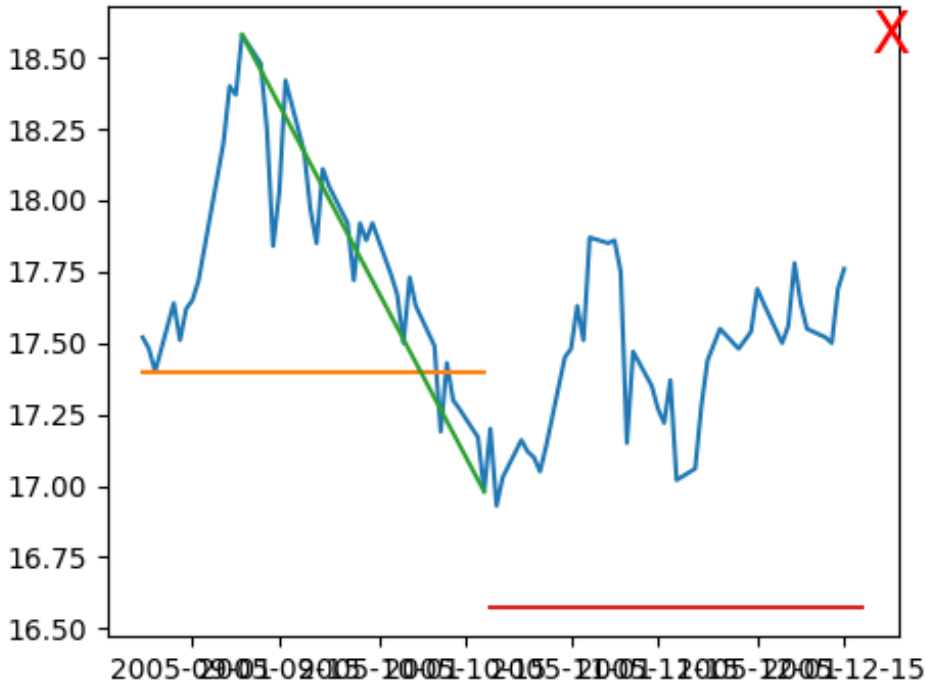
CSCO ascending_triangle 2014-10-30 - 2015-01-15 - CNN



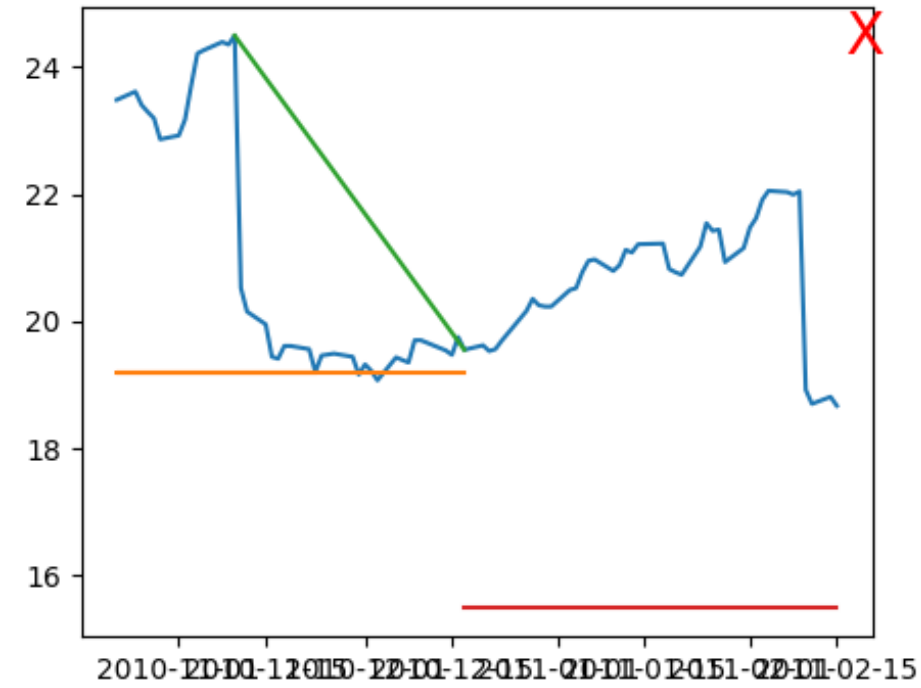
CSCO ascending_triangle 2022-10-14 - 2023-03-09 - CNN



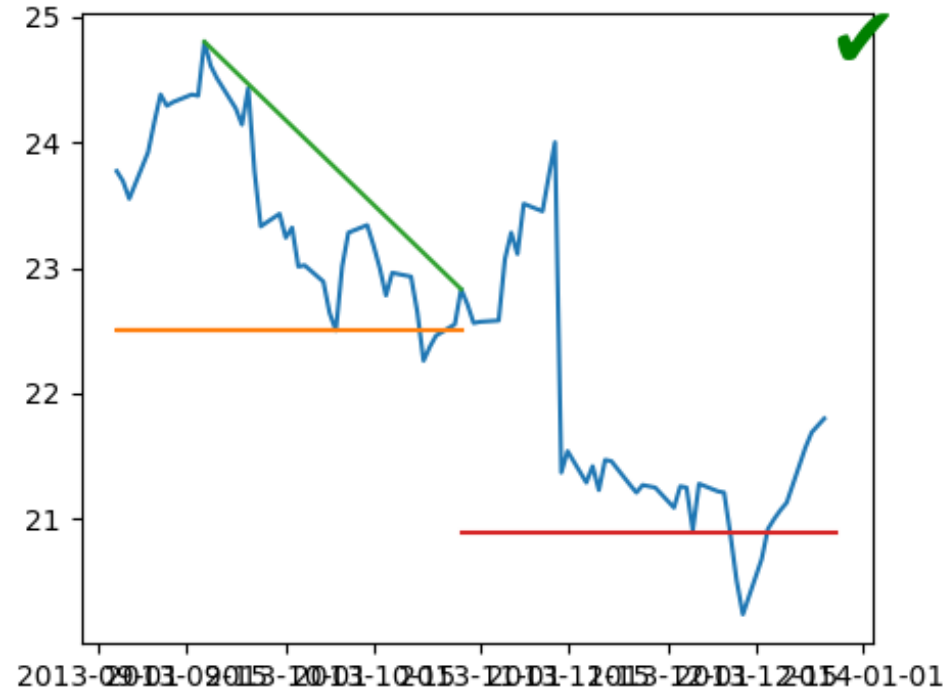
CSCO descending_triangle 2005-08-24 - 2005-10-19 - CNN



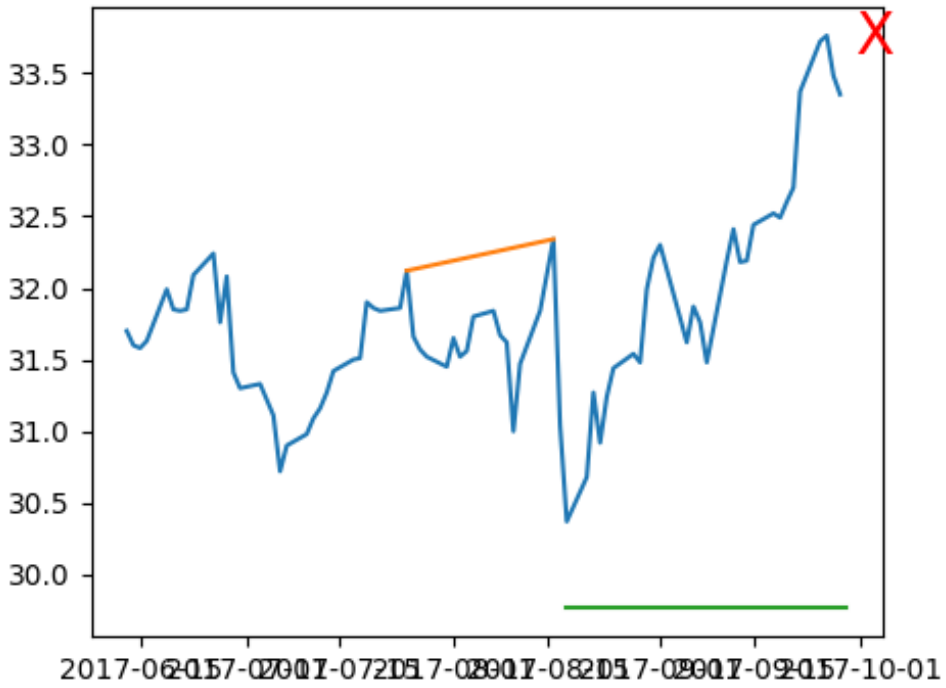
CSCO descending_triangle 2010-10-22 - 2010-12-17 - CNN



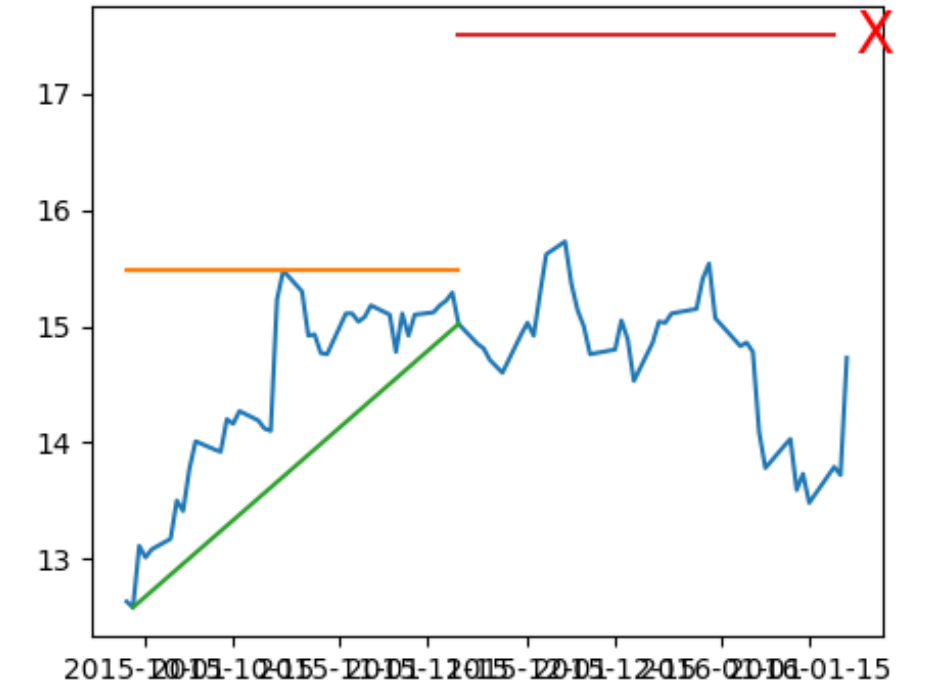
CSCO descending_triangle 2013-09-04 - 2013-10-29 - CNN



CSCO double_top 2017-06-13 - 2017-08-22 - CNN

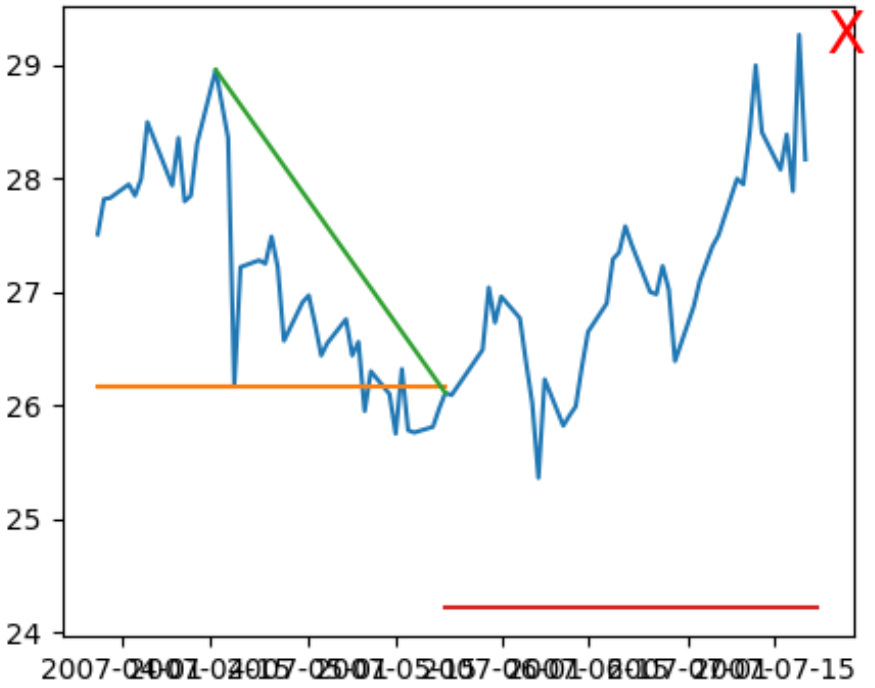


LOGI ascending_triangle 2015-09-28 - 2015-11-20 - CNN

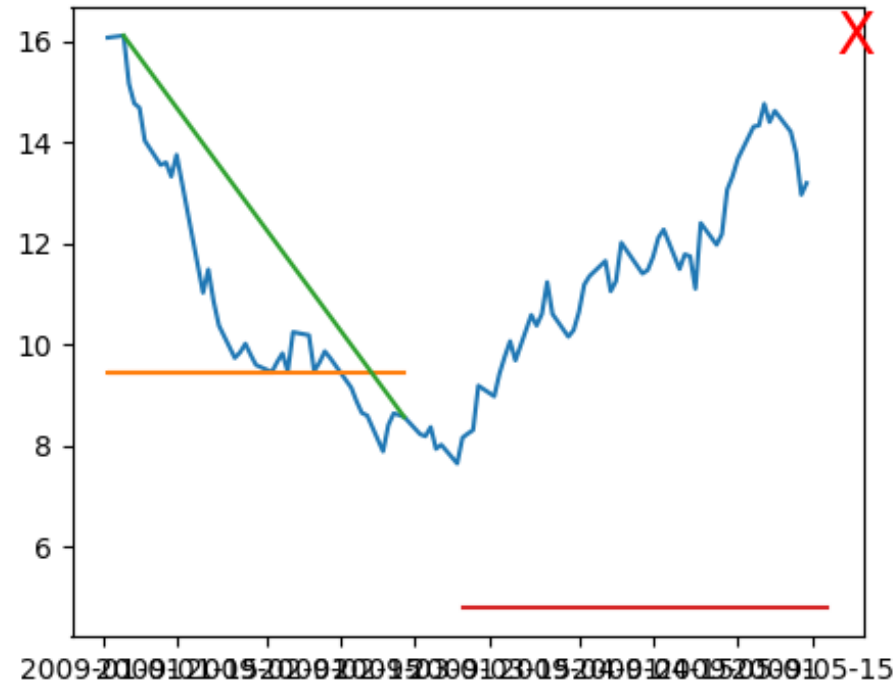


LOGI ascending_triangle 2017-05-01 - 2017-06-26 - CNN

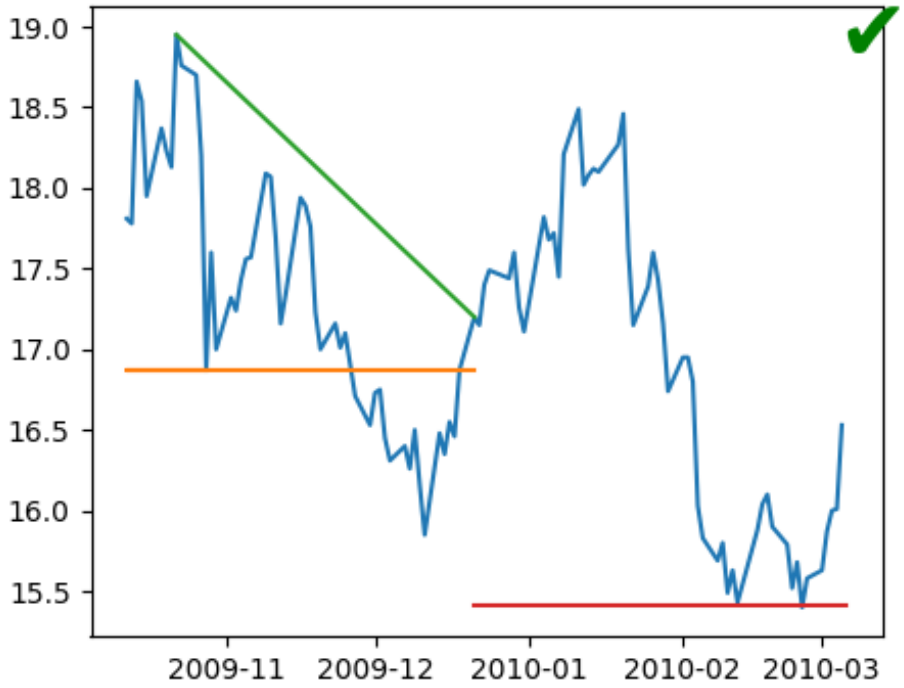
LOGI descending_triangle 2007-03-28 - 2007-05-23 - CNN



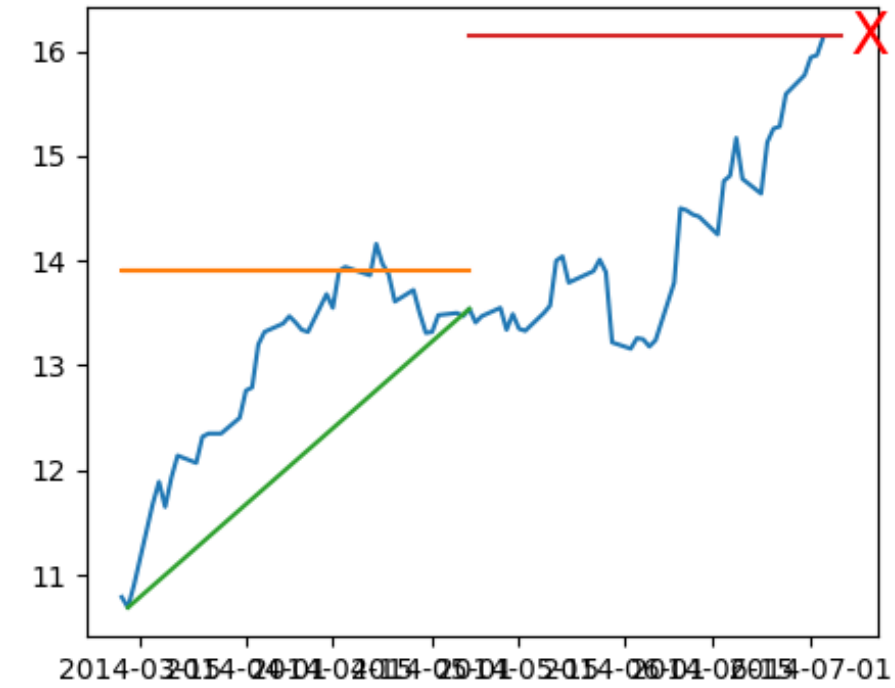
LOGI descending_triangle 2009-01-02 - 2009-03-10 - CNN



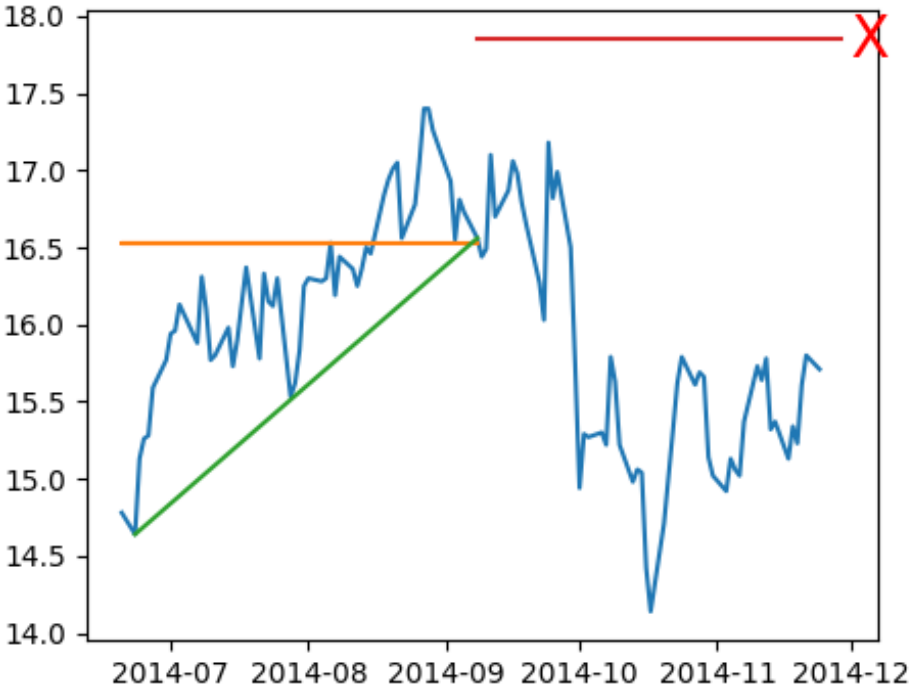
LOGI descending_triangle 2009-10-12 - 2009-12-21 - CNN



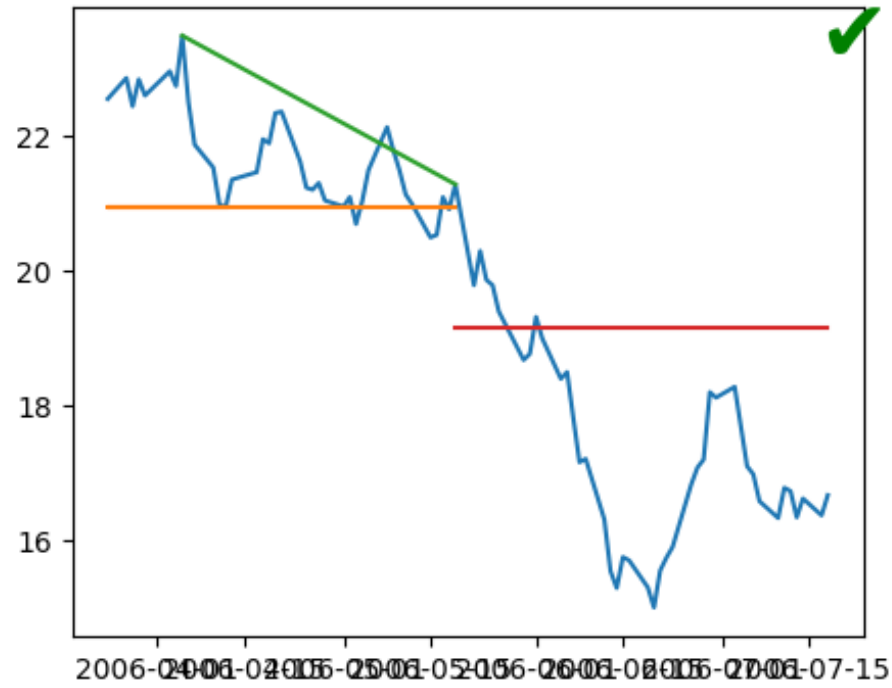
LPL ascending_triangle 2014-03-12 - 2014-05-07 - CNN



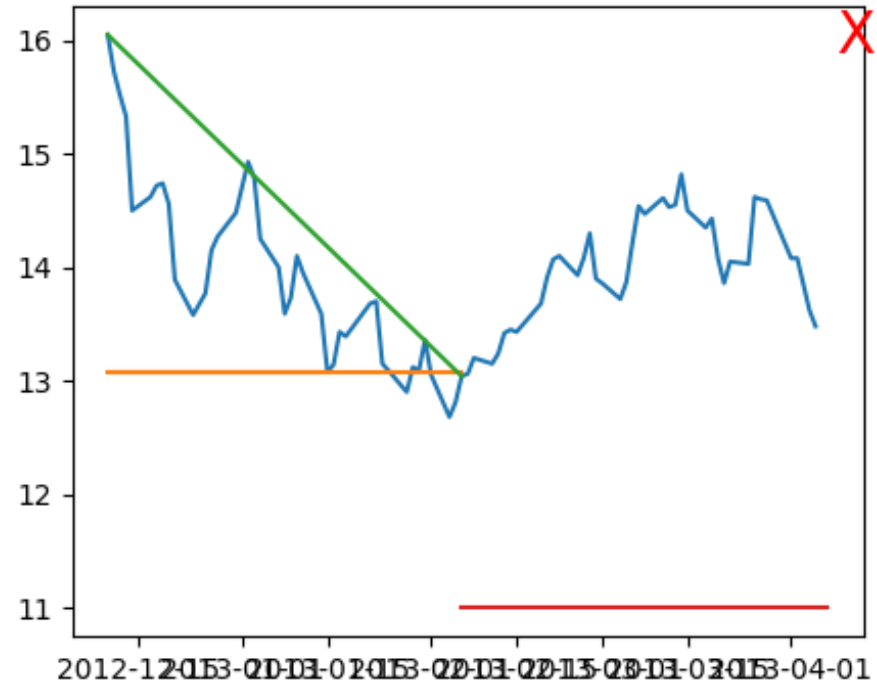
LPL ascending_triangle 2014-06-20 - 2014-09-08 - CNN

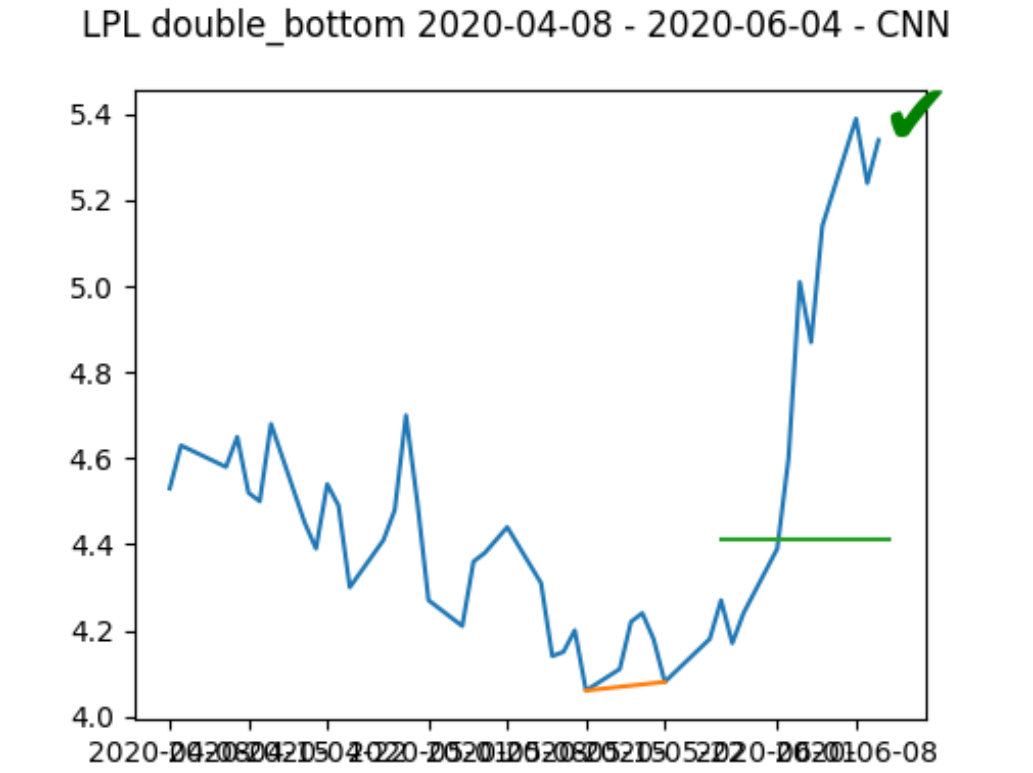


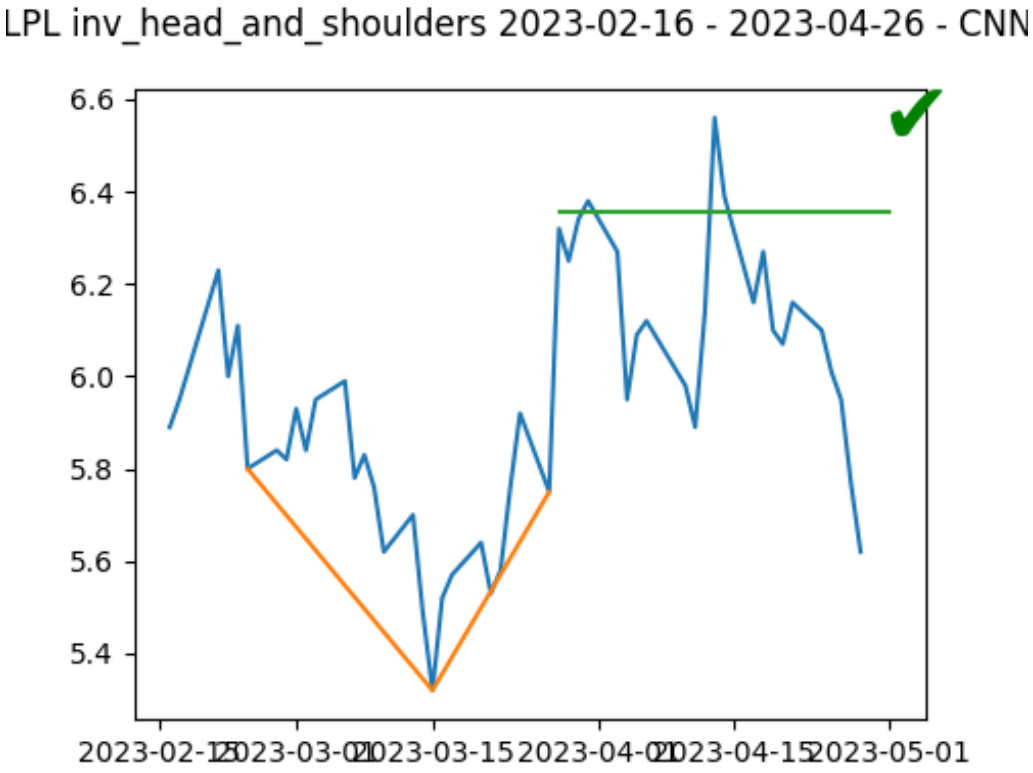
LPL descending_triangle 2006-03-24 - 2006-05-19 - CNN



LPL descending_triangle 2012-12-10 - 2013-02-06 - CNN

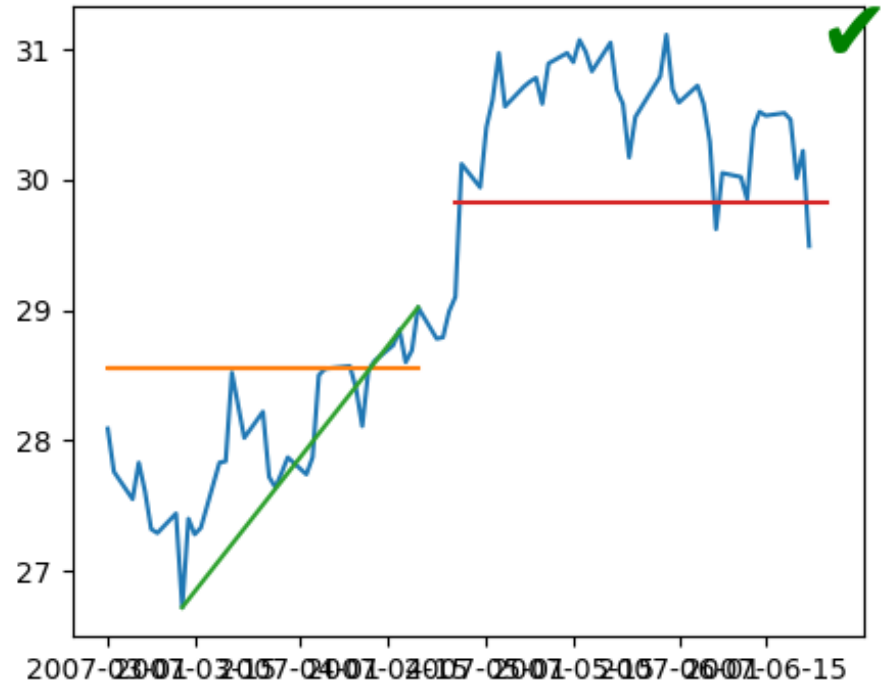



LPL double_bottom 2020-04-08 - 2020-06-04 - CNN
5.4
5.2
5.0
4.8
4.6
4.4
4.2
4.0

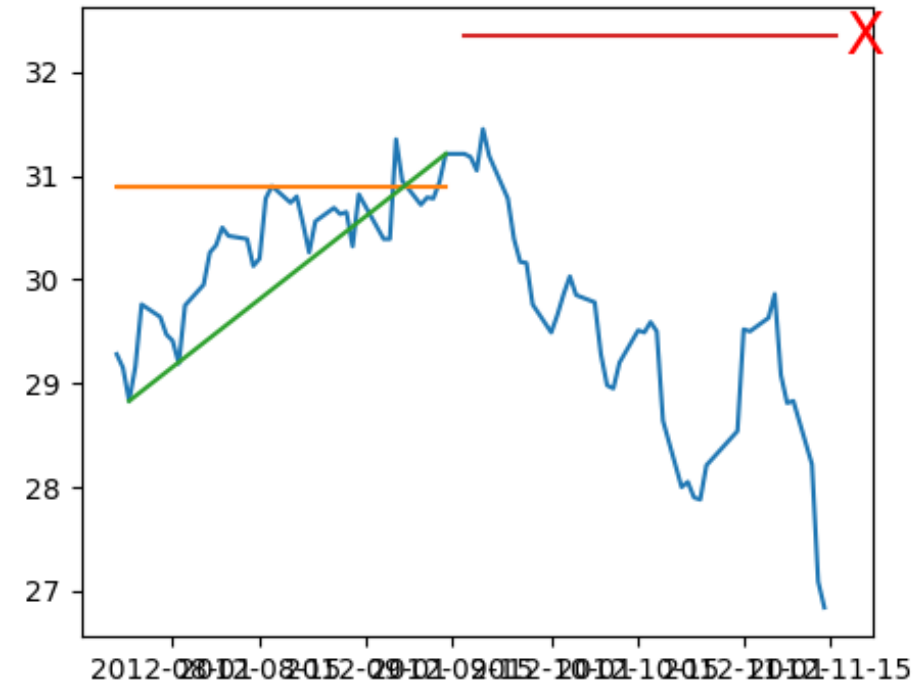

LPL inv_head_and_shoulders 2023-02-16 - 2023-04-26 - CNN
6.6
6.4
6.2
6.0
5.8
5.6
5.4
2023-03-01
2023-03-15
2023-04-01
2023-04-15
2023-05-01

MSFT ascending_triangle 2006-09-13 - 2007-02-06 - CNN

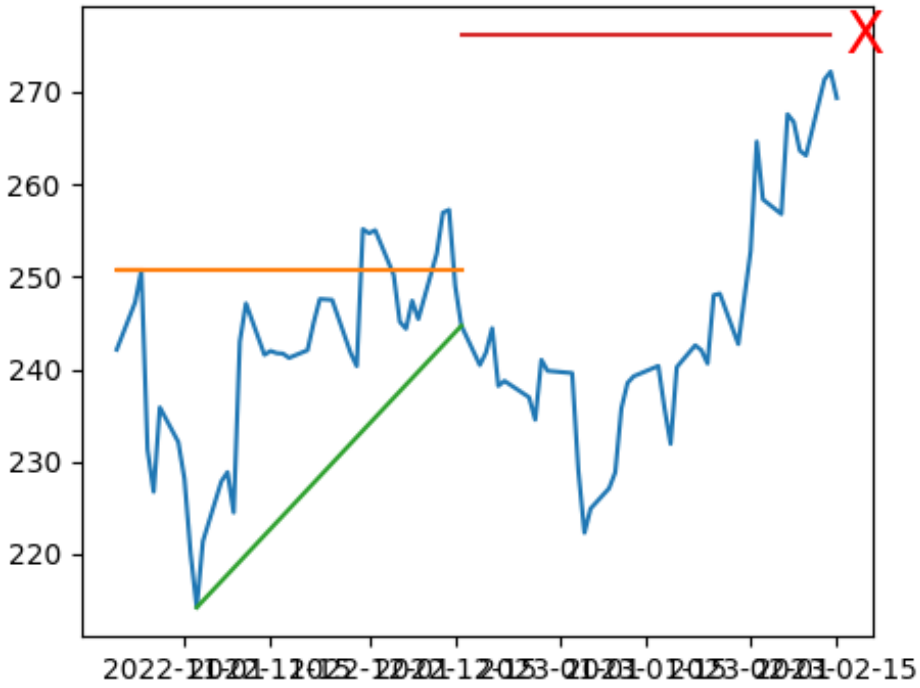
MSFT ascending_triangle 2007-03-01 - 2007-04-26 - CNN



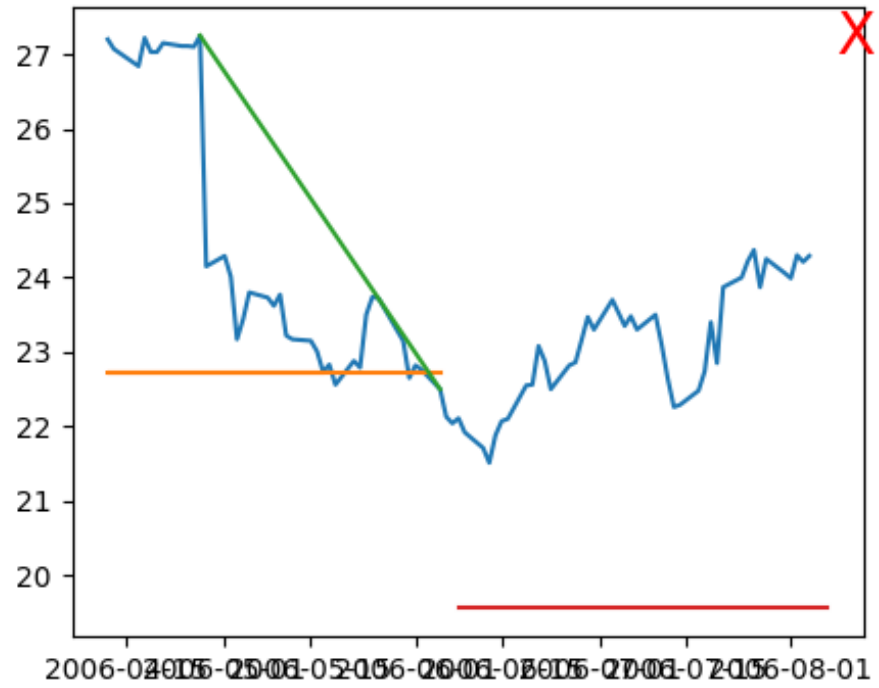
MSFT ascending_triangle 2012-07-23 - 2012-09-17 - CNN



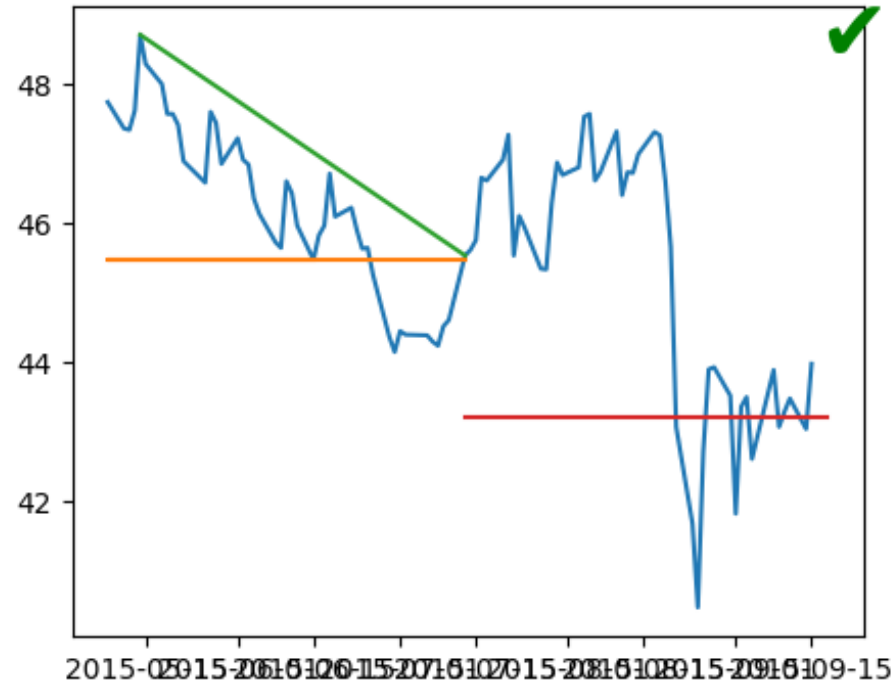
MSFT ascending_triangle 2022-10-21 - 2022-12-16 - CNN



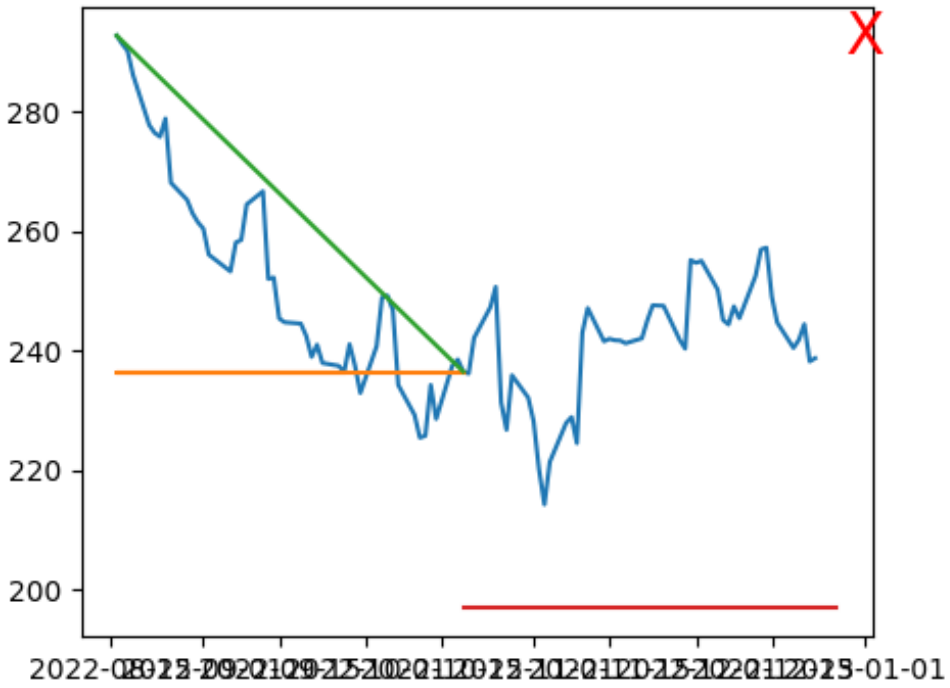
MSFT descending_triangle 2006-04-12 - 2006-06-08 - CNN



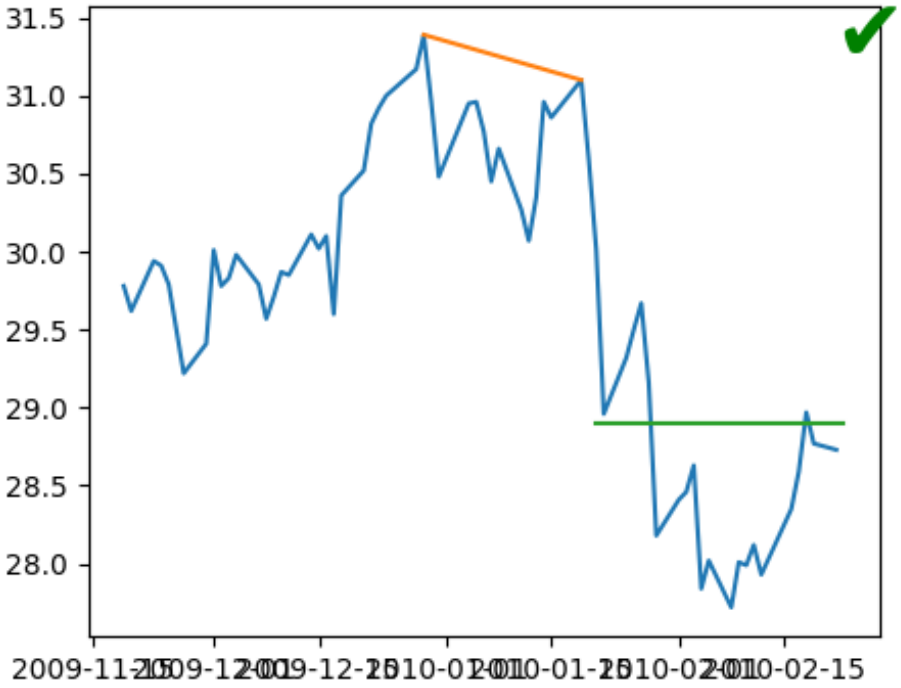
MSFT descending_triangle 2015-05-08 - 2015-07-13 - CNN



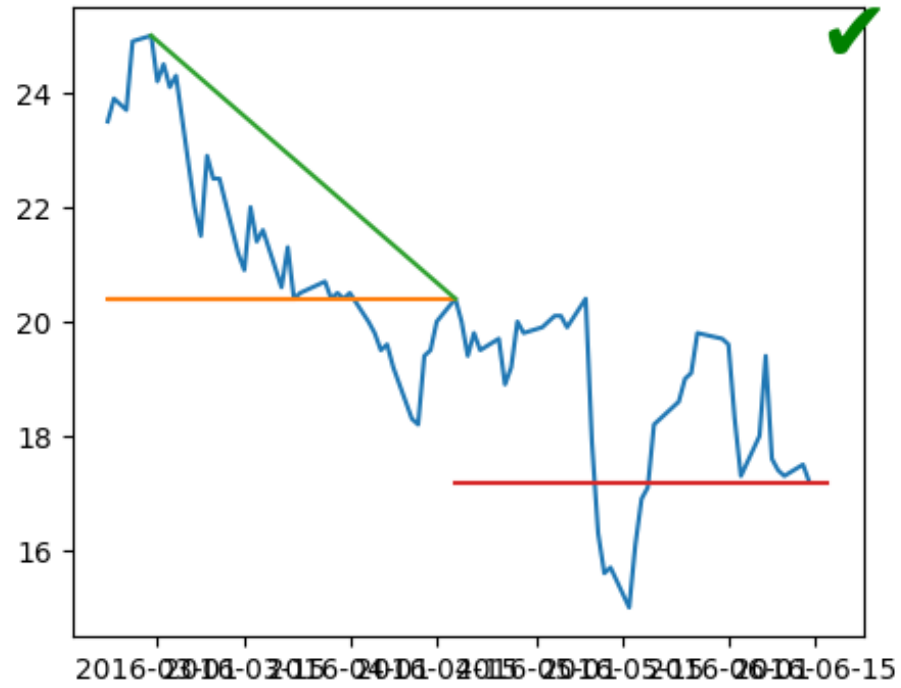
MSFT descending_triangle 2022-08-16 - 2022-10-19 - CNN



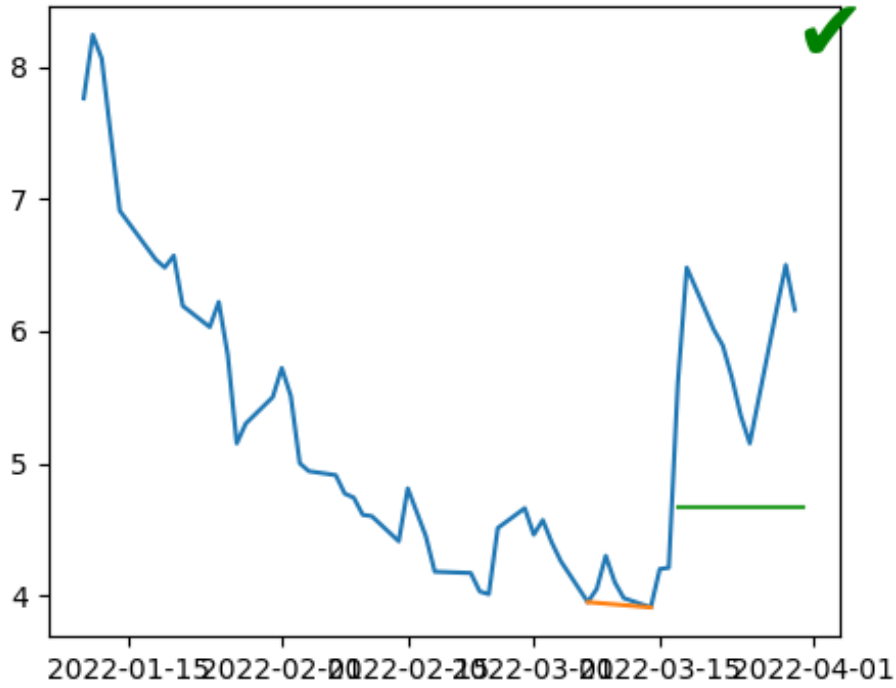
MSFT double_top 2009-11-19 - 2010-02-02 - CNN



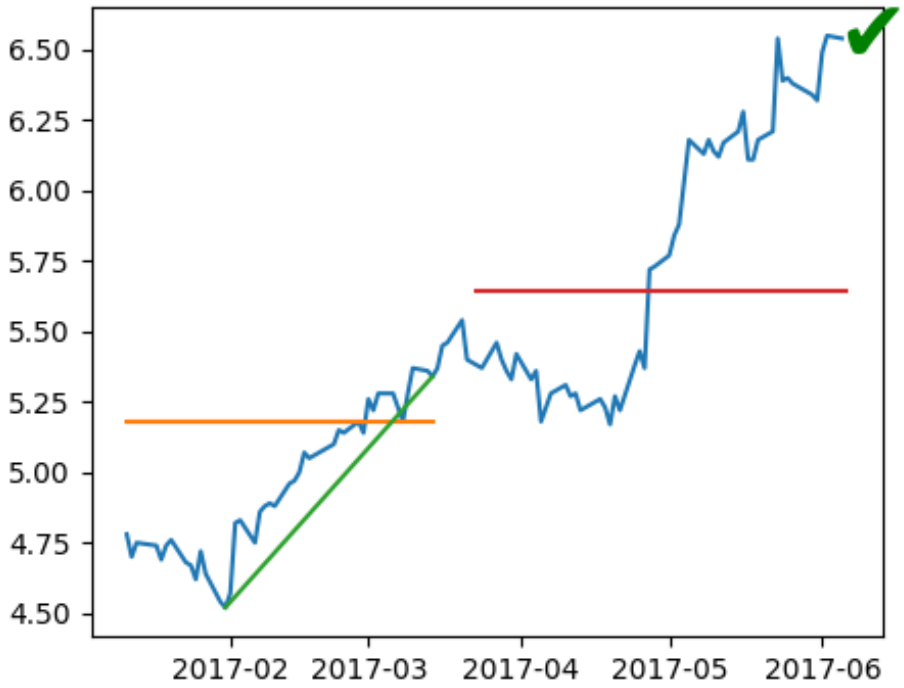
NEON descending_triangle 2016-02-22 - 2016-04-18 - CNN



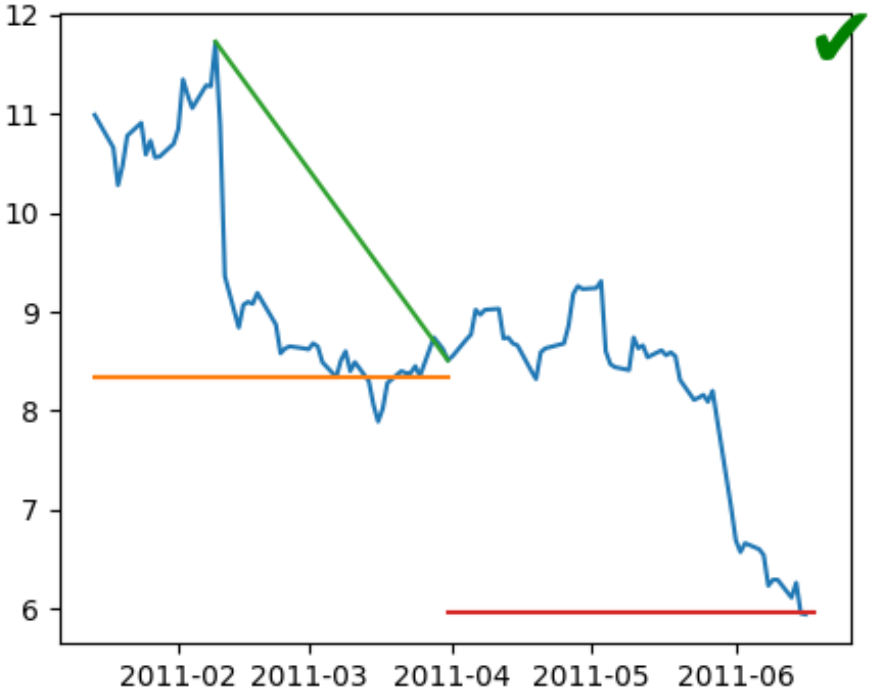
NEON double_bottom 2022-01-10 - 2022-03-24 - CNN



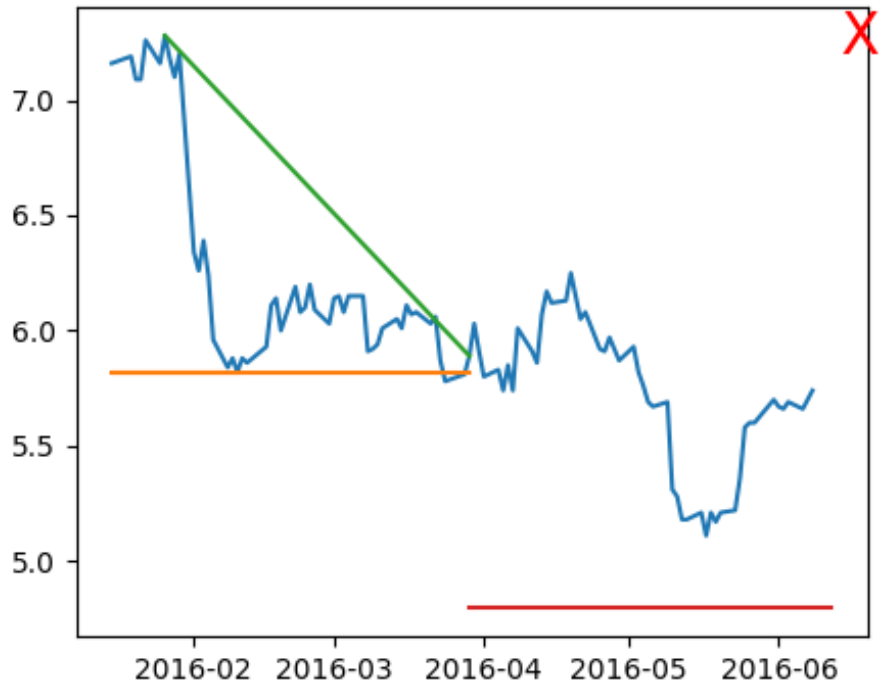
NOK ascending_triangle 2017-01-11 - 2017-03-23 - CNN



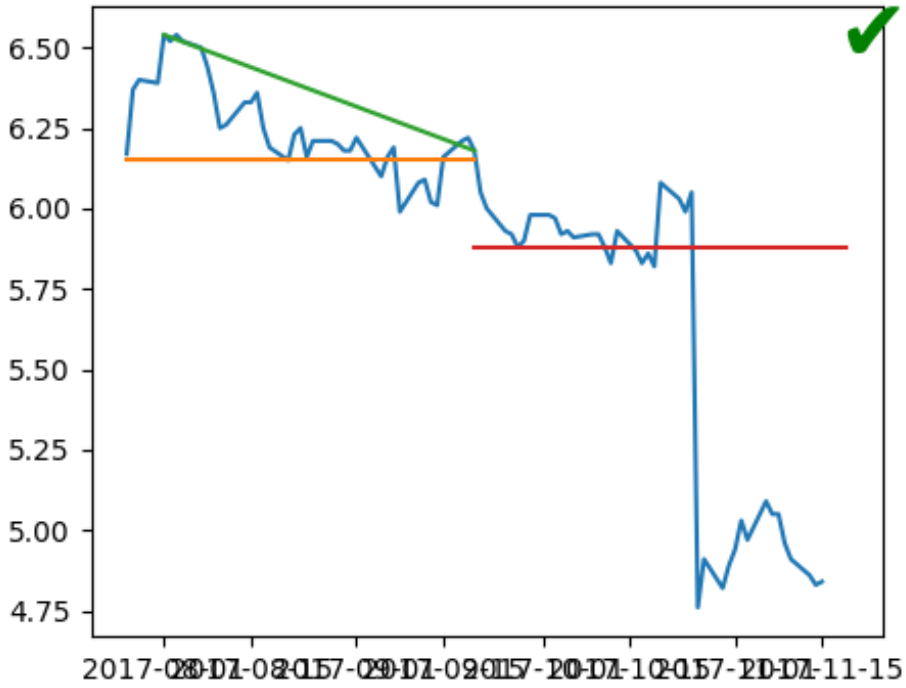
NOK descending_triangle 2011-01-14 - 2011-03-31 - CNN



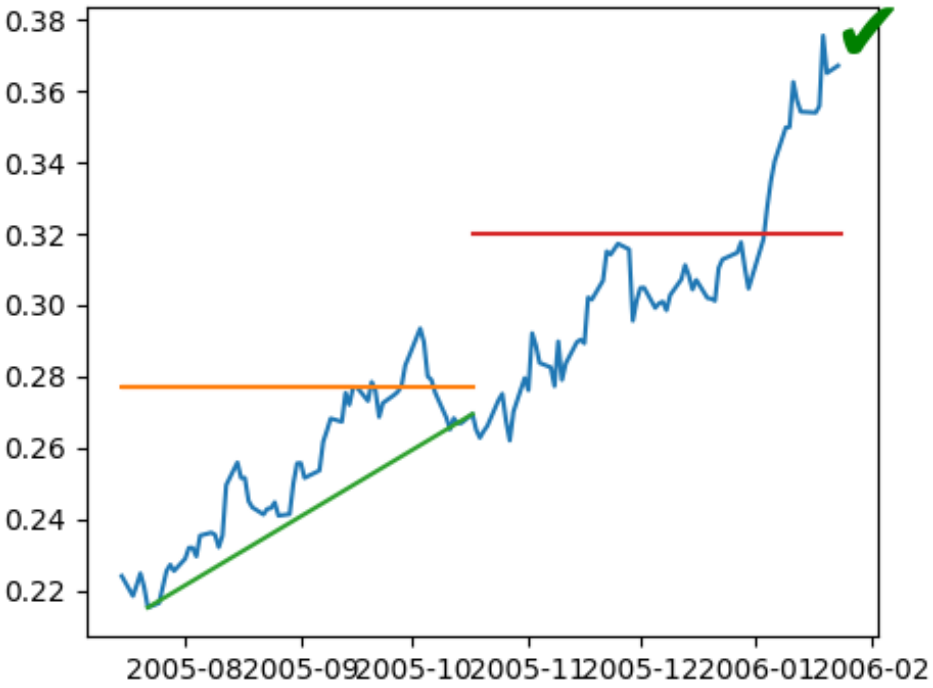
NOK descending_triangle 2016-01-15 - 2016-03-29 - CNN



NOK descending_triangle 2017-07-26 - 2017-09-20 - CNN

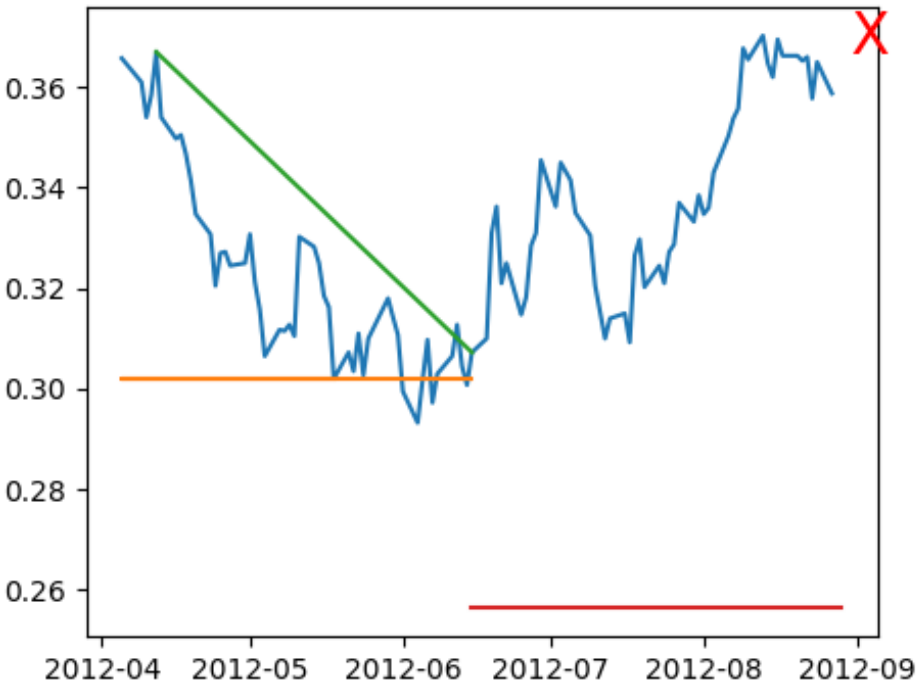


NVDA ascending_triangle 2005-07-15 - 2005-10-17 - CNN

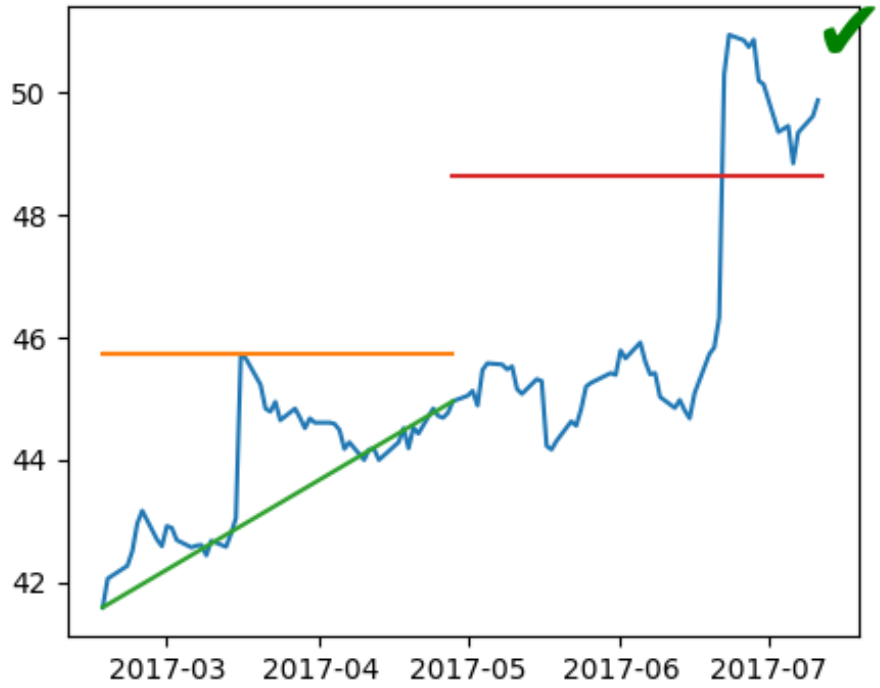


NVDA descending_triangle 2011-05-27 - 2011-07-25 - CNN

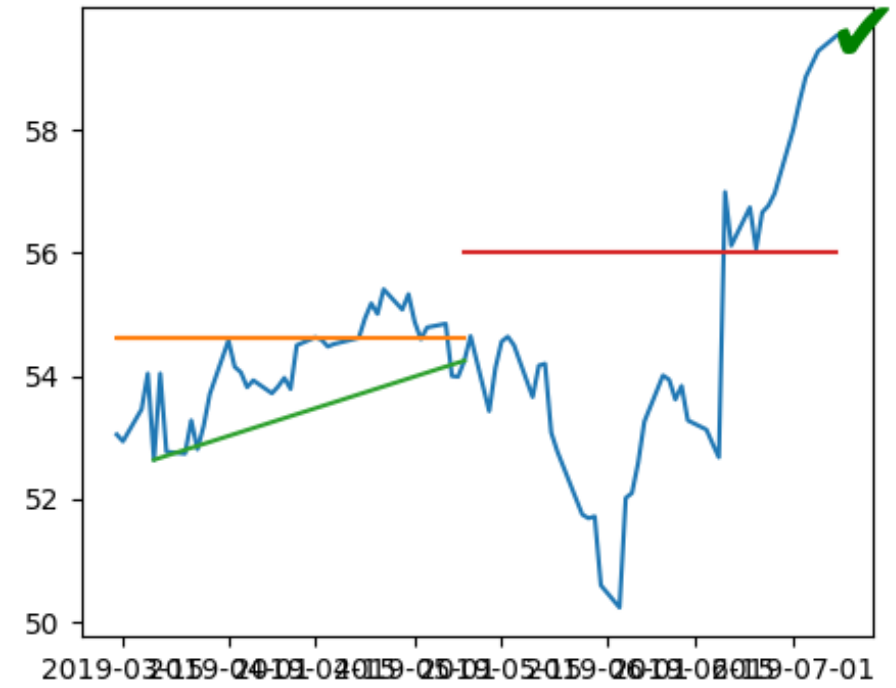
NVDA descending_triangle 2012-04-05 - 2012-06-15 - CNN



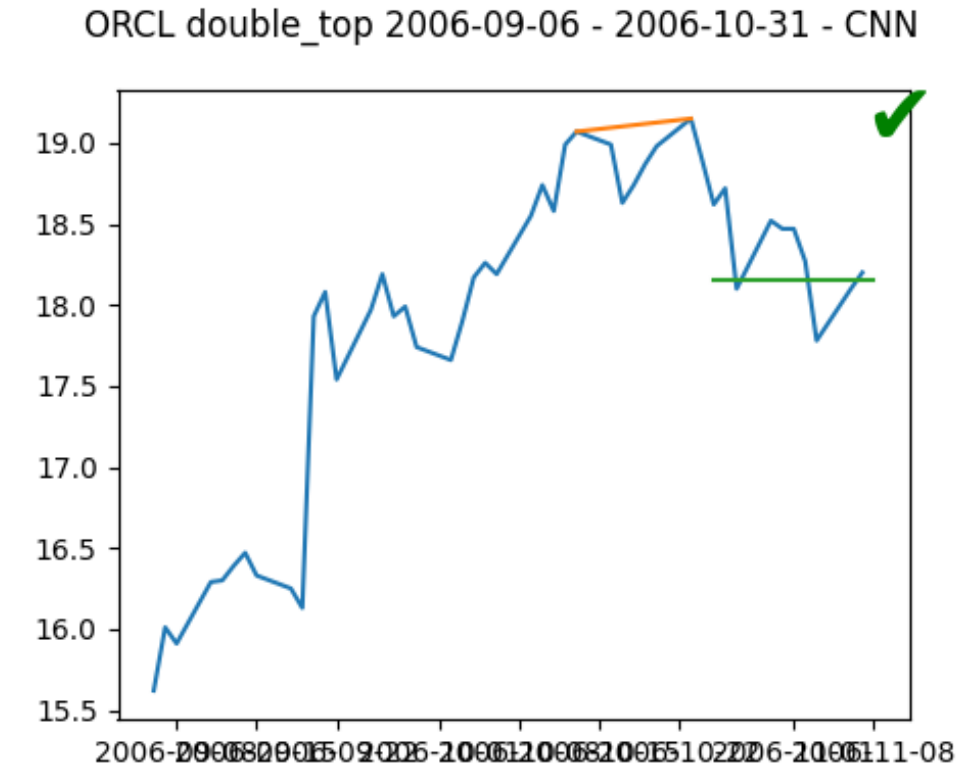
ORCL ascending_triangle 2017-02-16 - 2017-04-28 - CNN

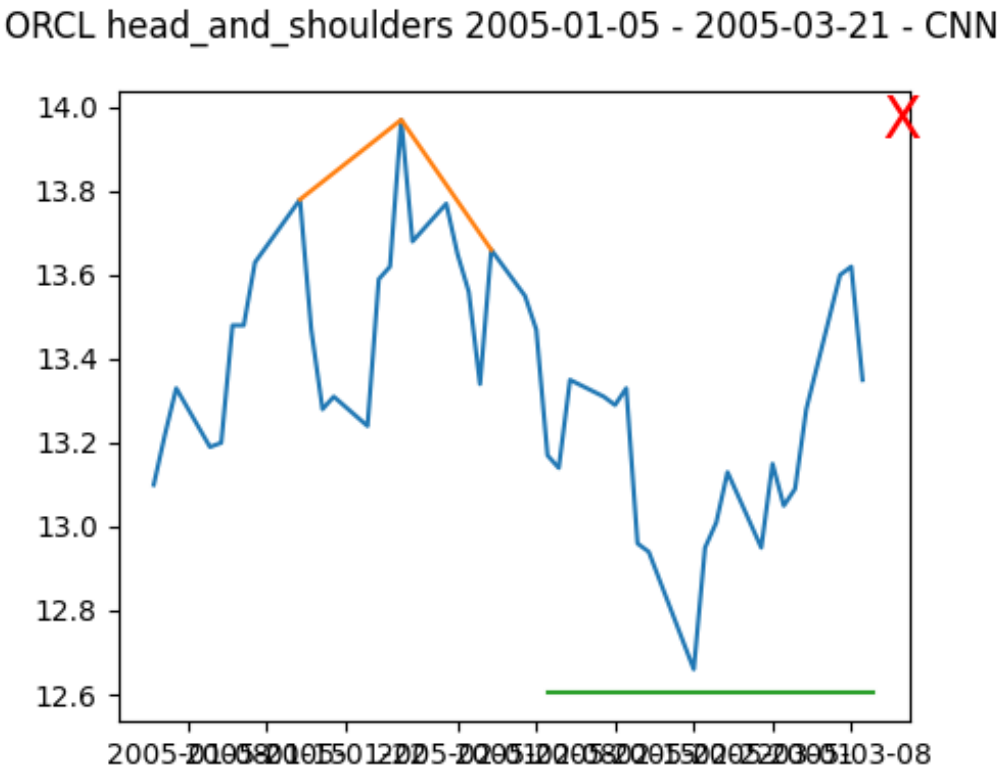


ORCL ascending_triangle 2019-03-14 - 2019-05-09 - CNN

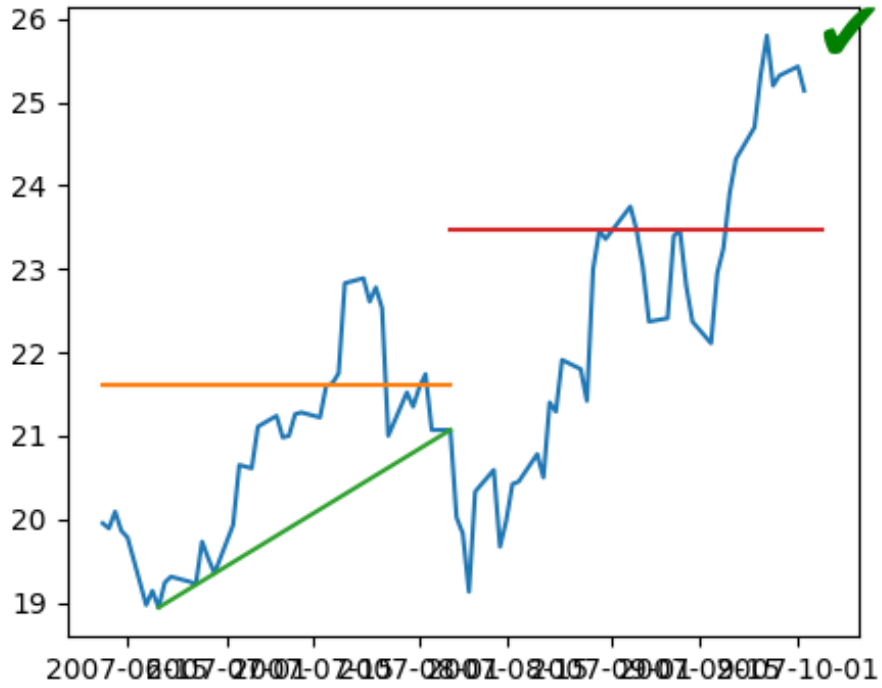


ORCL descending_triangle 2021-12-07 - 2022-03-03 - CNN

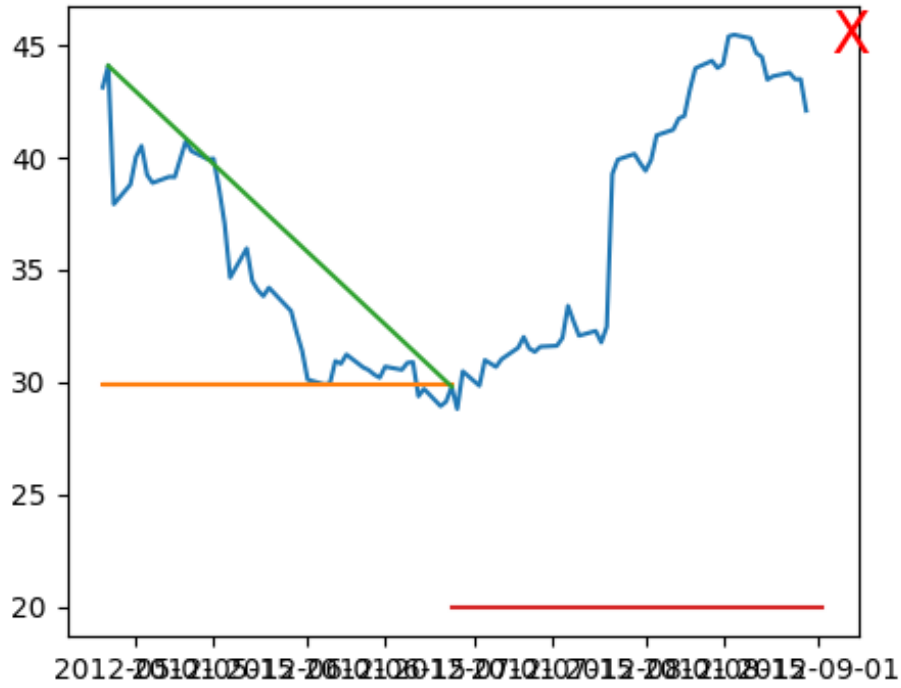

ORCL double_top 2006-09-06 - 2006-10-31 - CNN
19.0
18.5
18.0
17.5
17.0
16.5
16.0
15.5


ORCL head_and_shoulders 2005-01-05 - 2005-03-21 - CNN
14.0
13.8
13.6
13.4
13.2
13.0
12.8
12.6

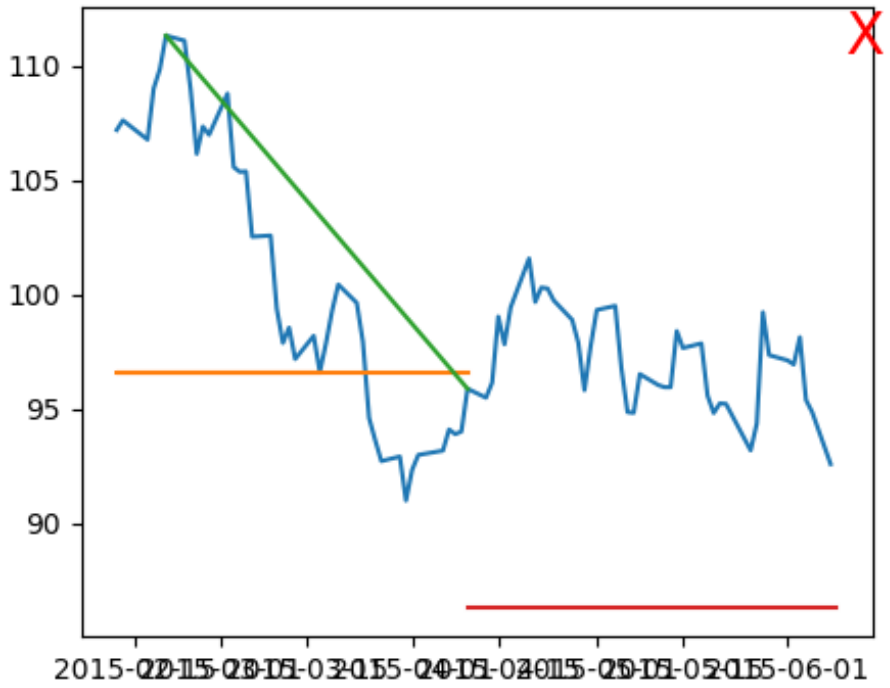
WDC ascending_triangle 2007-06-11 - 2007-08-06 - CNN



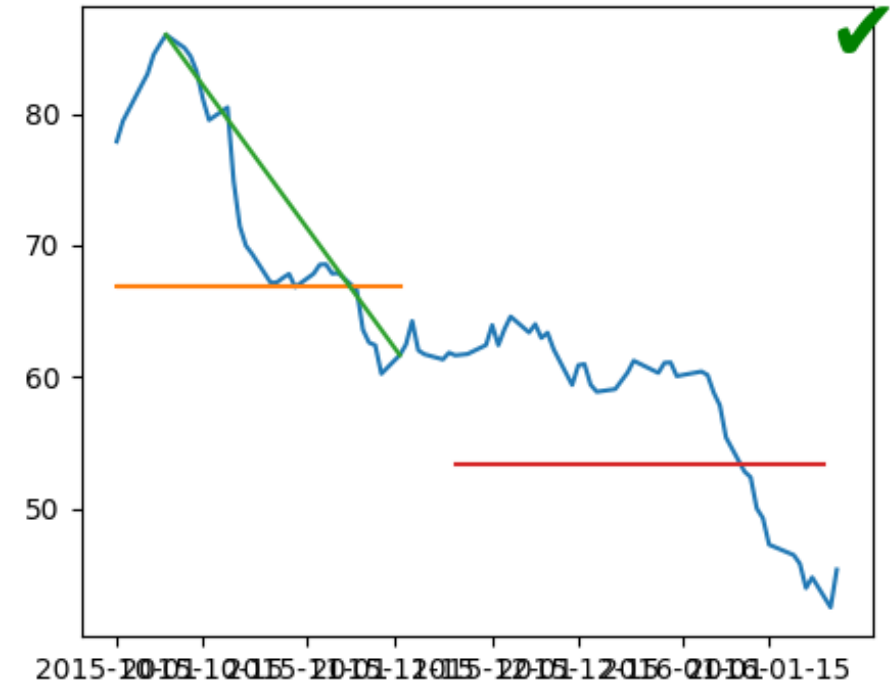
WDC descending_triangle 2012-04-25 - 2012-06-27 - CNN

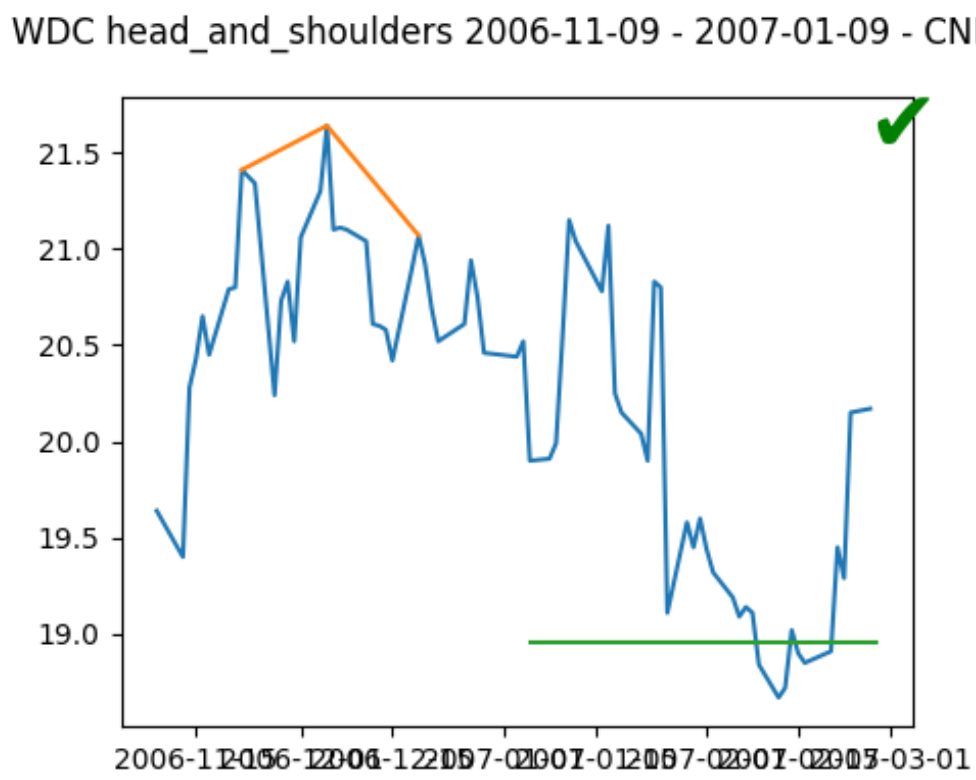


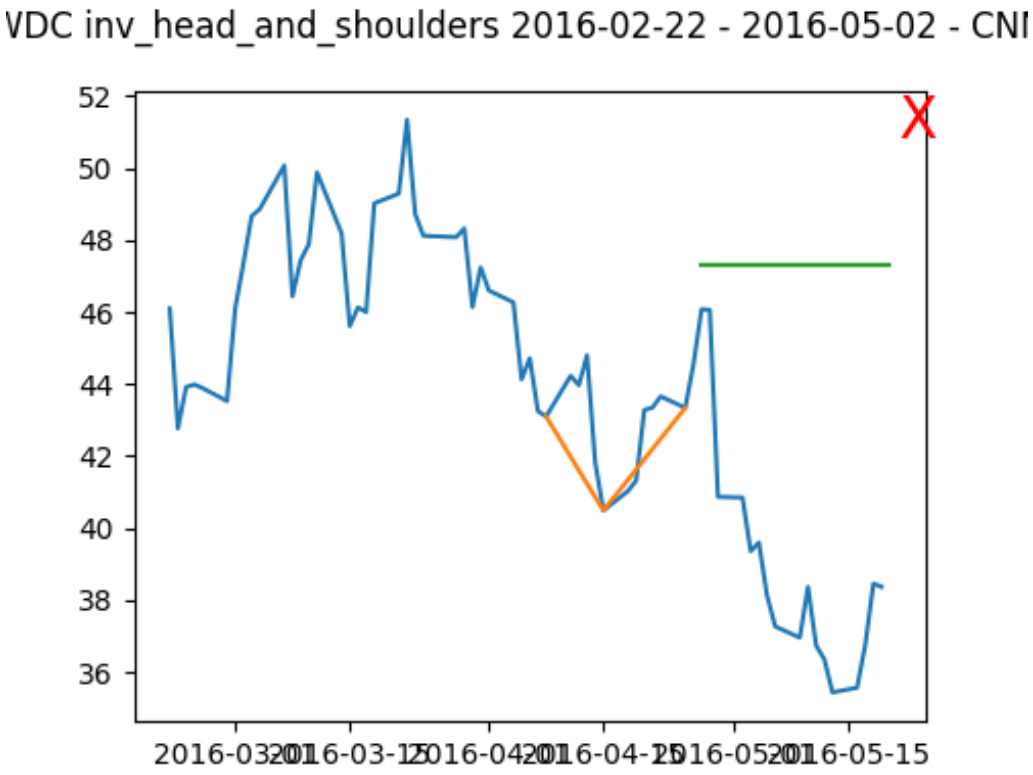
WDC descending_triangle 2015-02-12 - 2015-04-10 - CNN



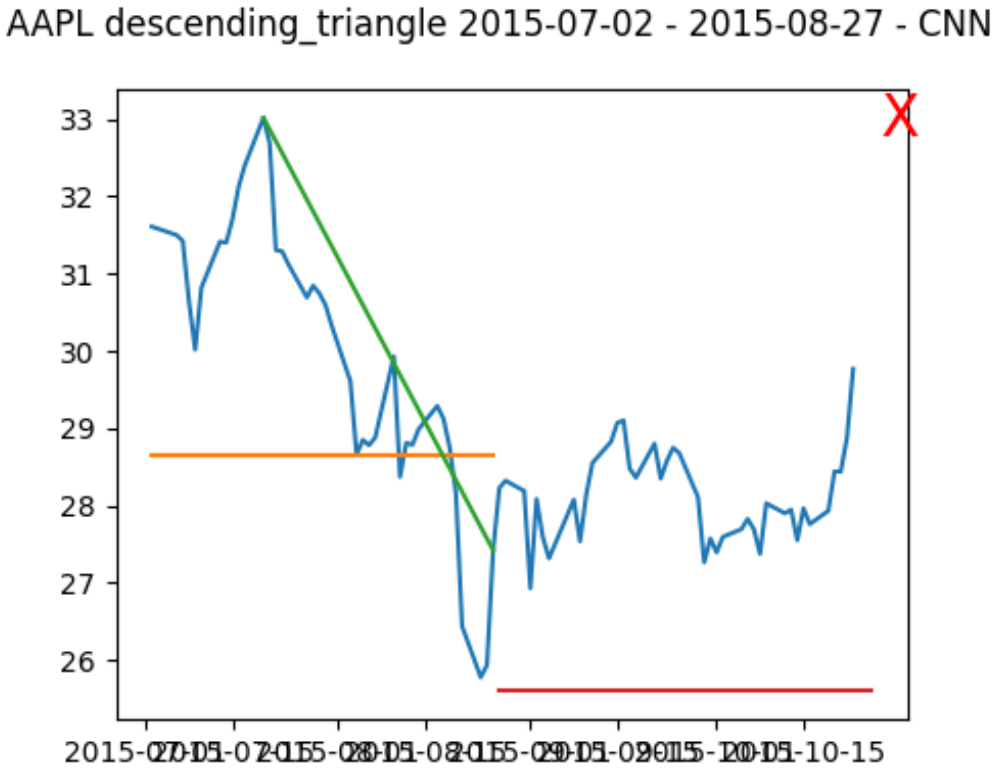
WDC descending_triangle 2015-10-01 - 2015-11-25 - CNN

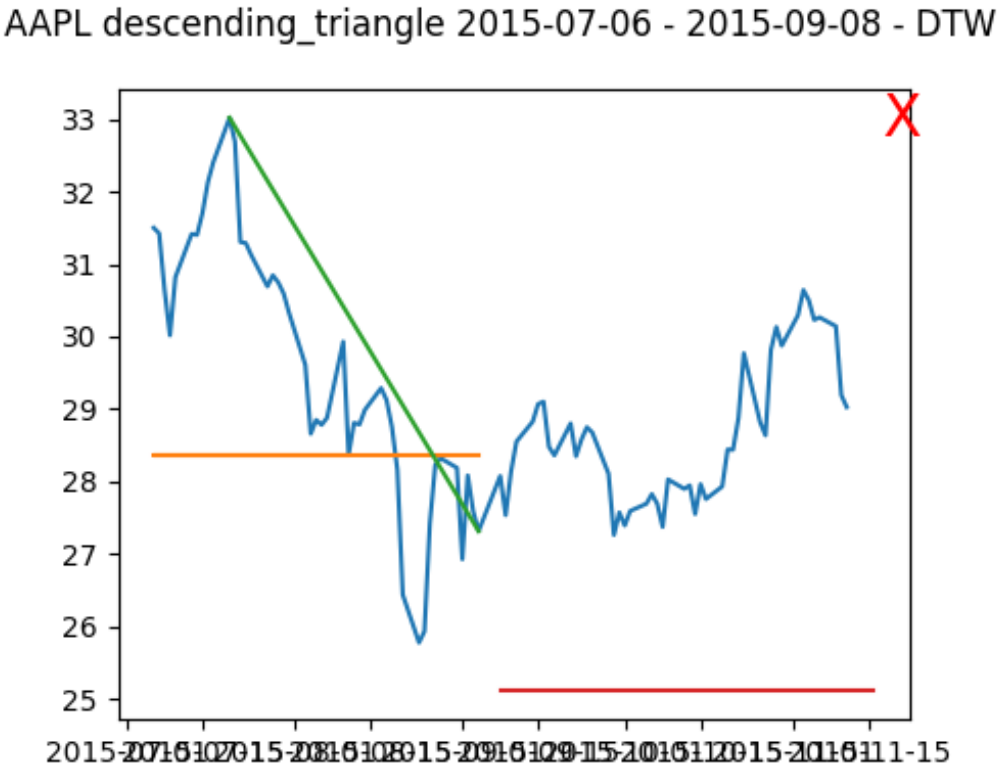



WDC head_and_shoulders 2006-11-09 - 2007-01-09 - CNN
21.5
21.0
20.5
20.0
19.5
19.0

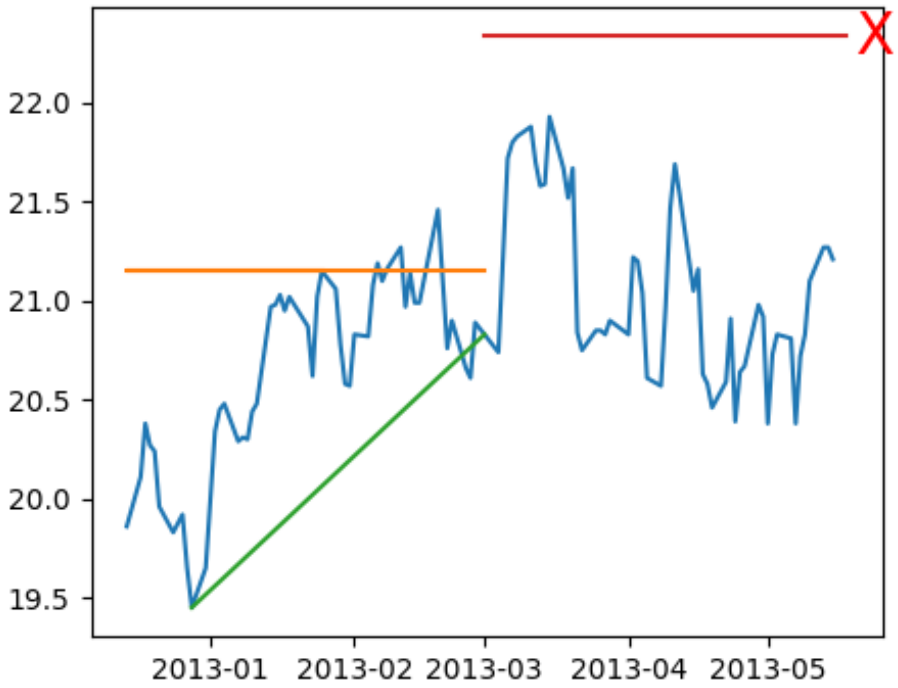

VDC inv_head_and_shoulders 2016-02-22 - 2016-05-02 - CNI
52
50
48
46
44
42
40
38
36

A.5. Patrones solapados sector tecnológico

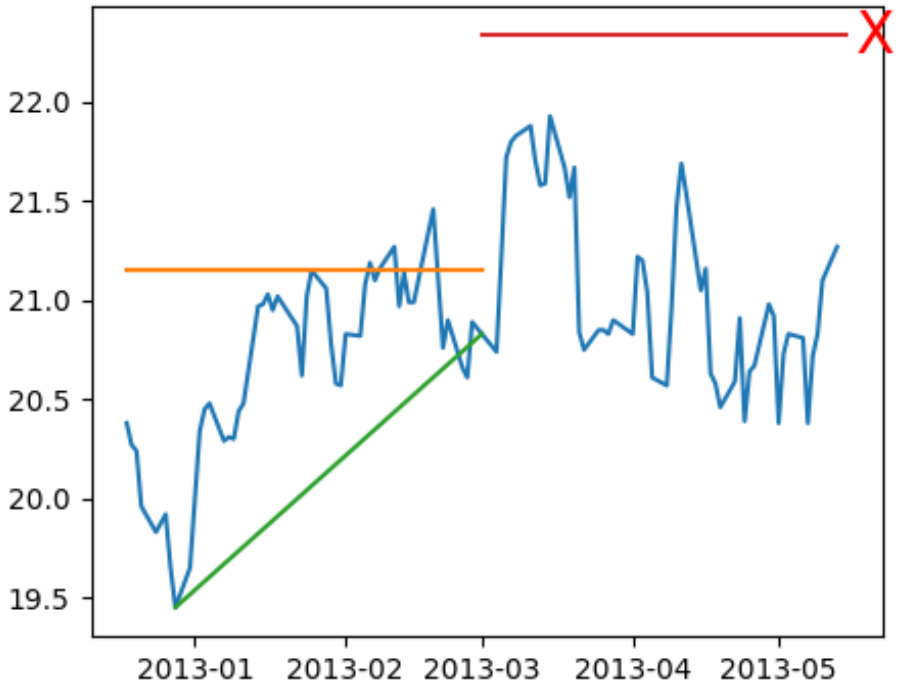




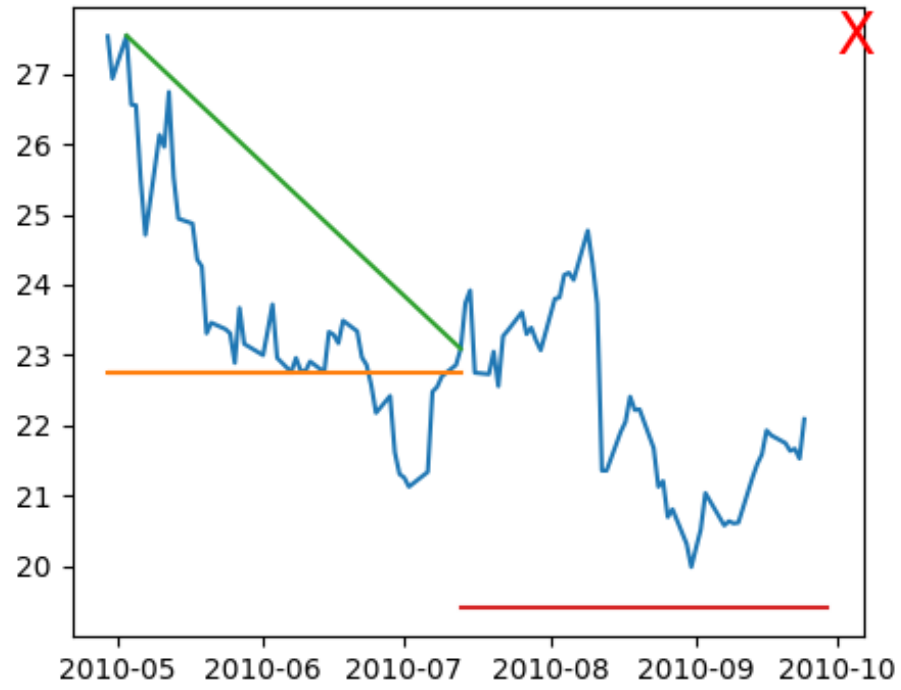
CSCO ascending_triangle 2012-12-14 - 2013-03-01 - CNN



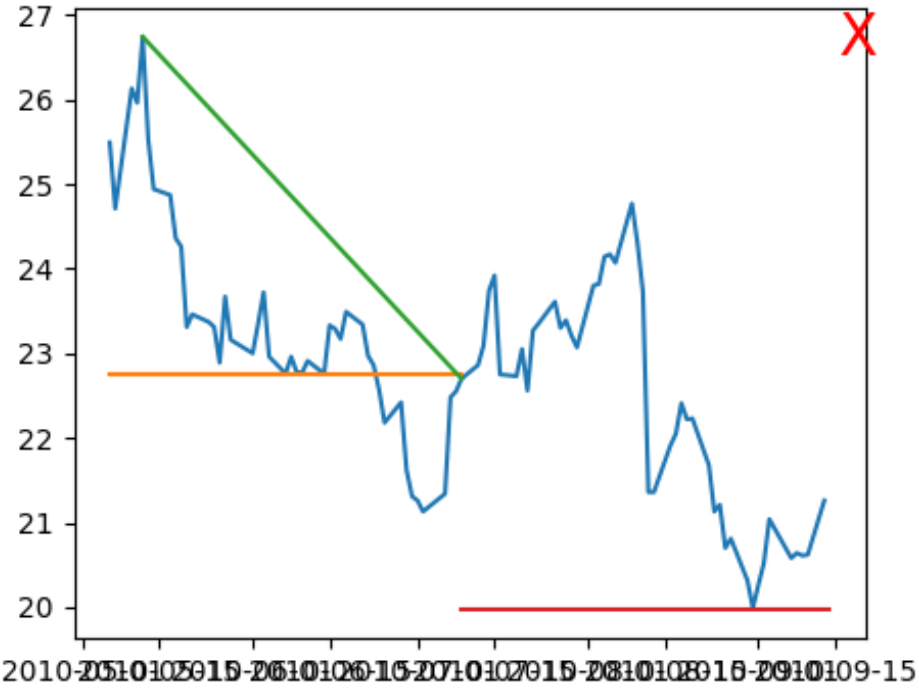
CSCO ascending_triangle 2012-12-18 - 2013-03-01 - DTW

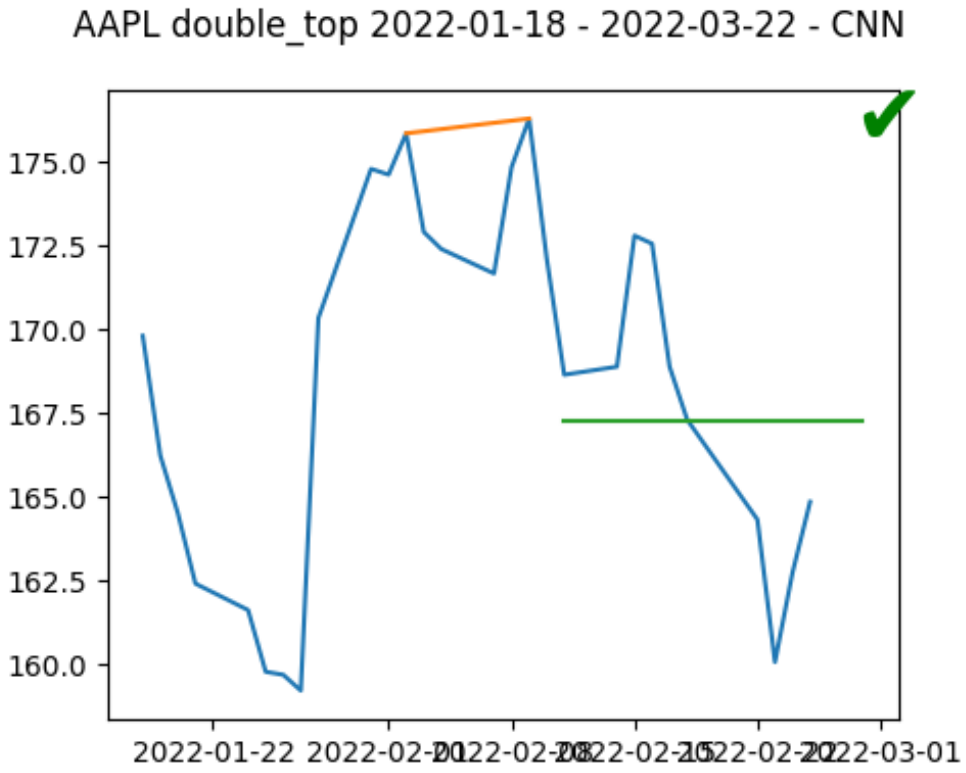


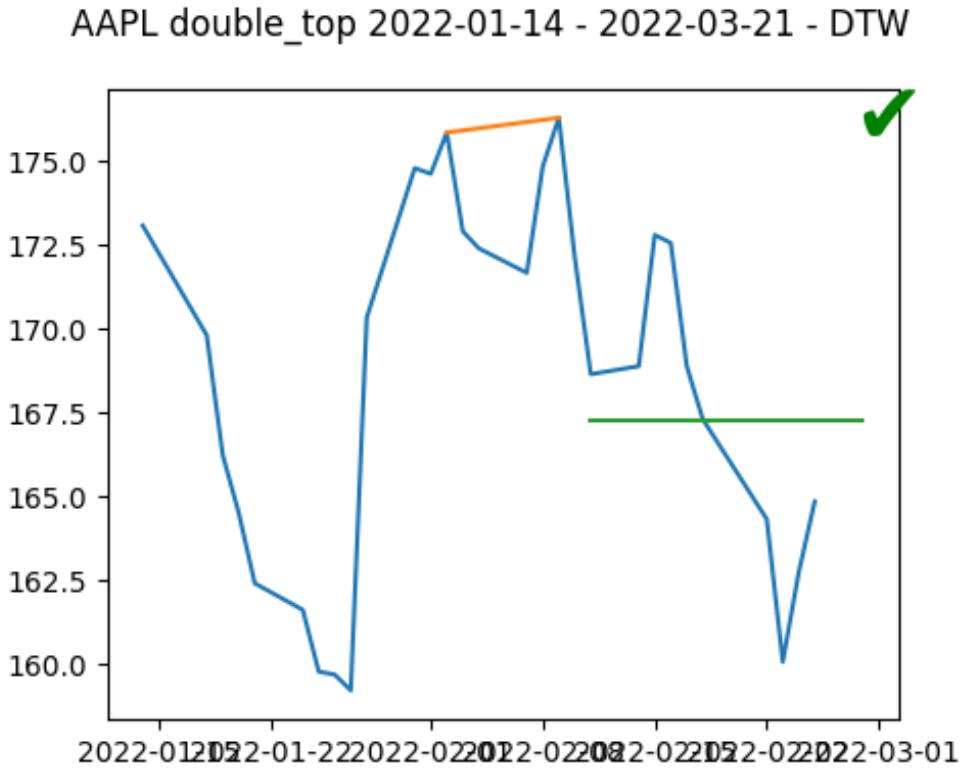
CSCO descending_triangle 2010-04-29 - 2010-07-13 - CNN



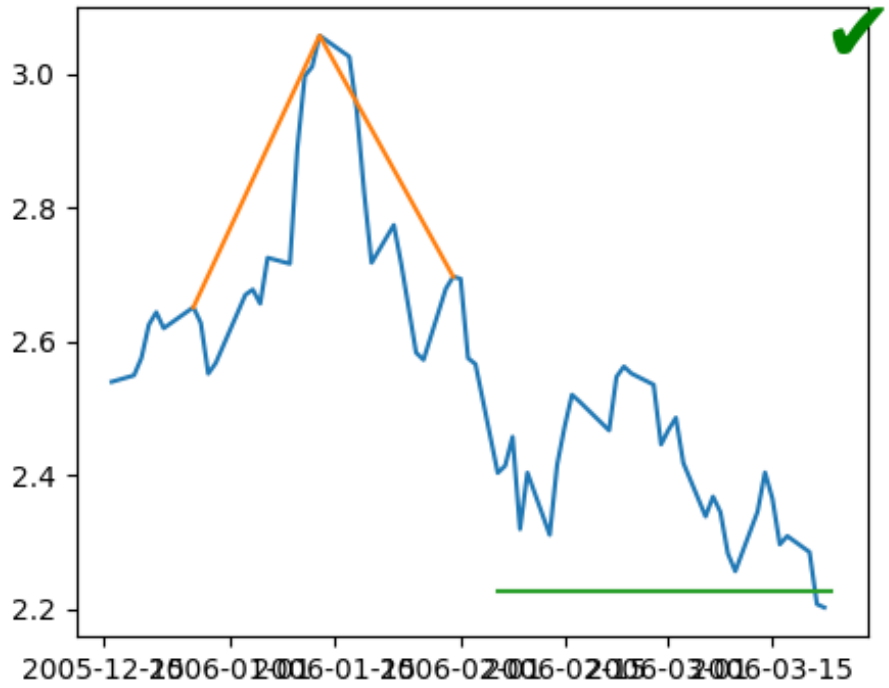
CSCO descending_triangle 2010-05-06 - 2010-07-09 - DTW



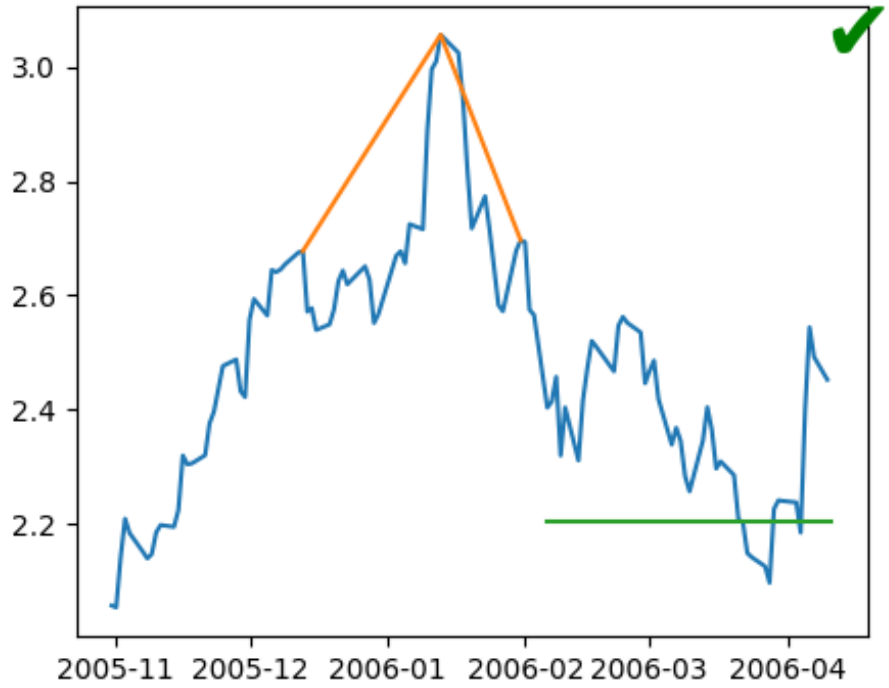

AAPL double_top 2022-01-18 - 2022-03-22 - CNN
175.0
172.5
170.0
167.5
165.0
162.5
160.0
2022-01-22
2022-03-01


AAPL double_top 2022-01-14 - 2022-03-21 - DTW
175.0
172.5
170.0
167.5
165.0
162.5
160.0
2022-01-22
2022-03-01

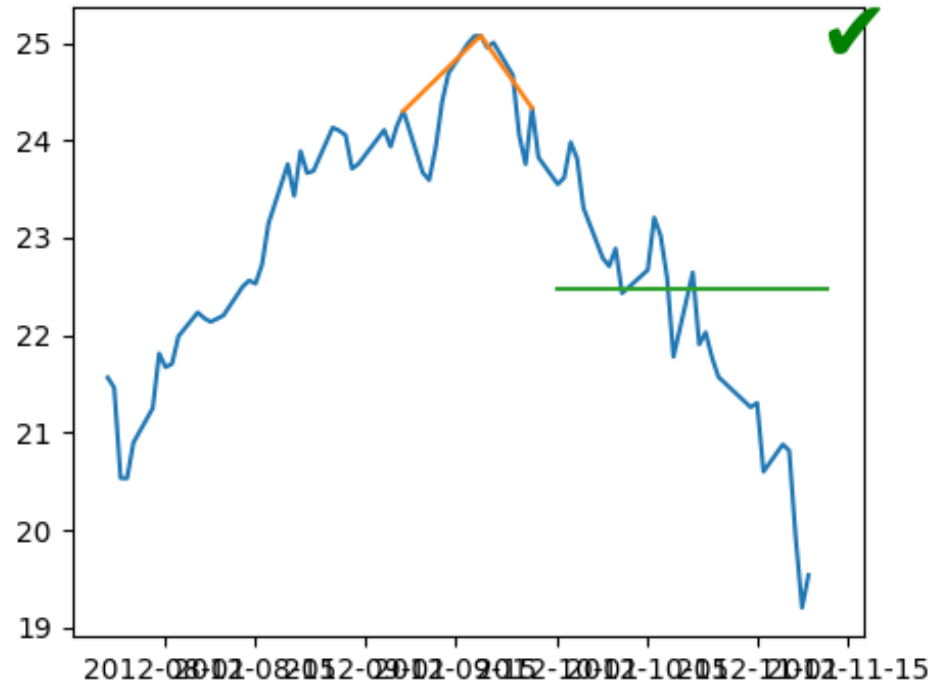
AAPL head_and_shoulders 2005-12-16 - 2006-02-14 - CNN



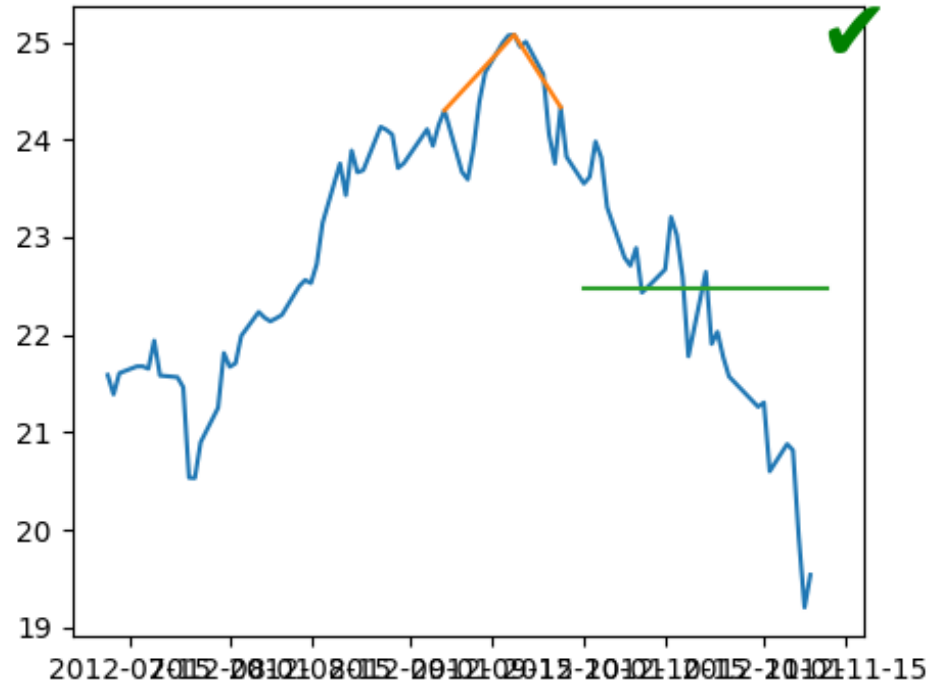
AAPL head_and_shoulders 2005-10-31 - 2006-03-10 - DTW

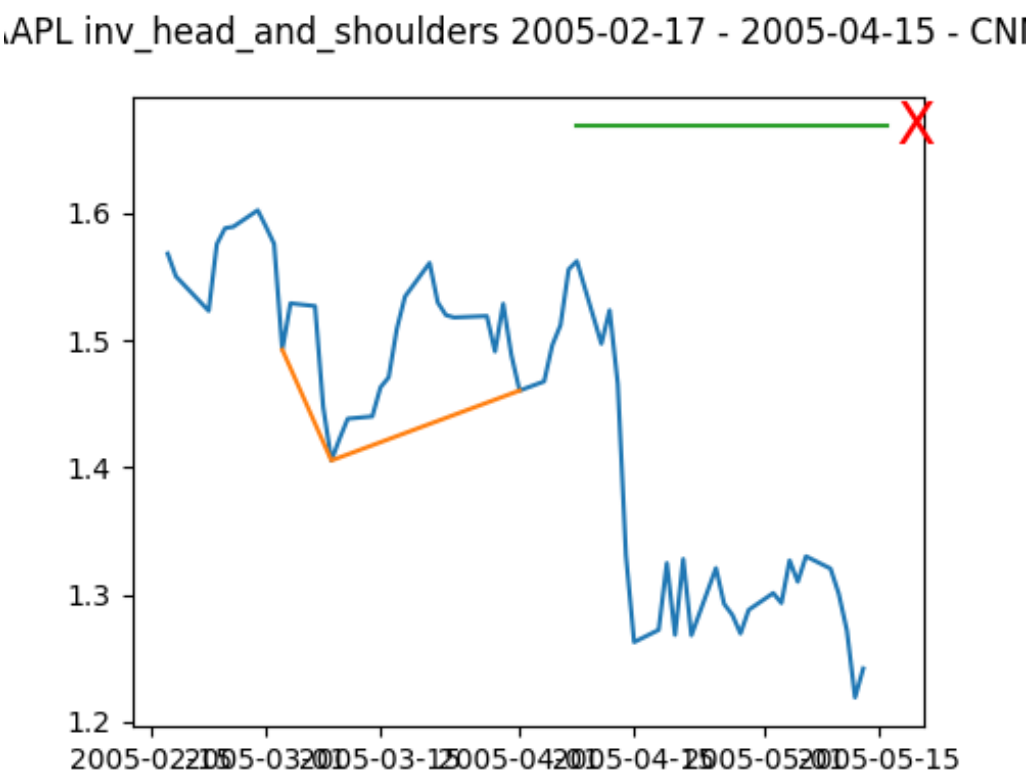


AAPL head_and_shoulders 2012-07-23 - 2012-11-14 - CNN

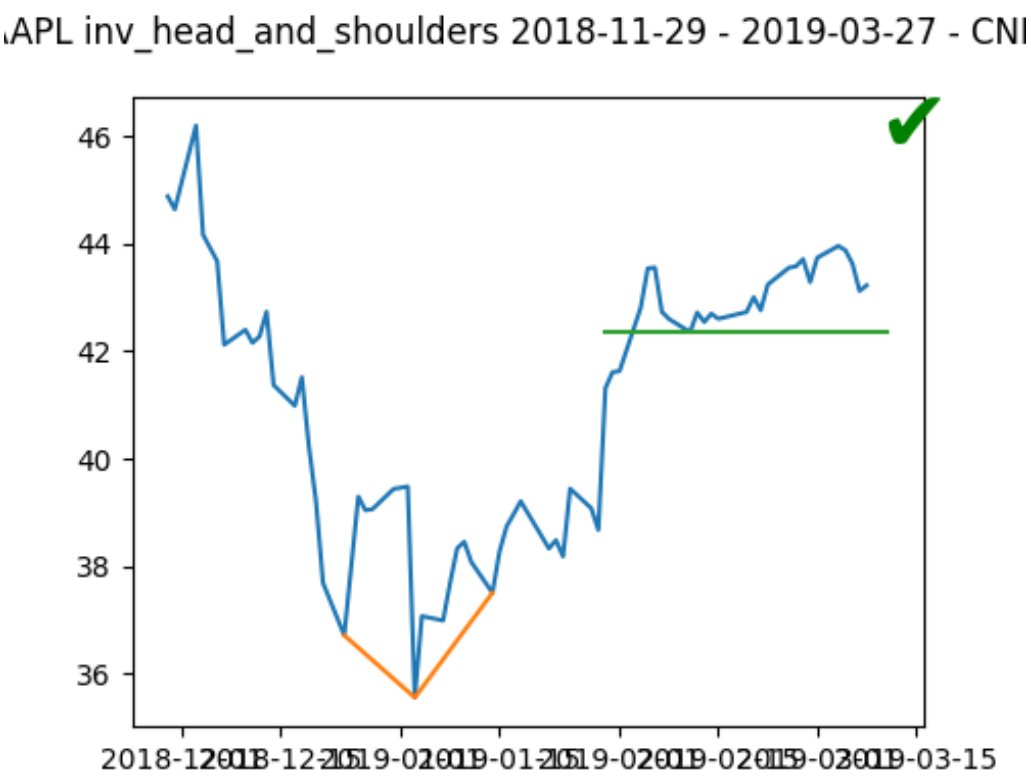


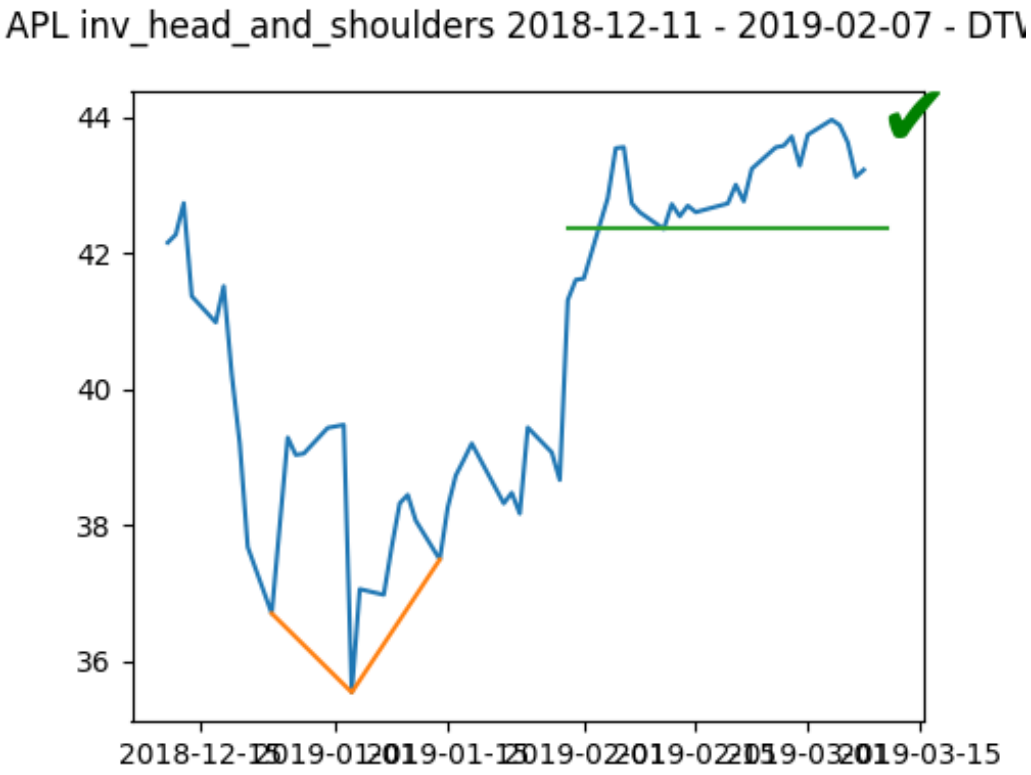
AAPL head_and_shoulders 2012-07-11 - 2012-11-02 - DTW

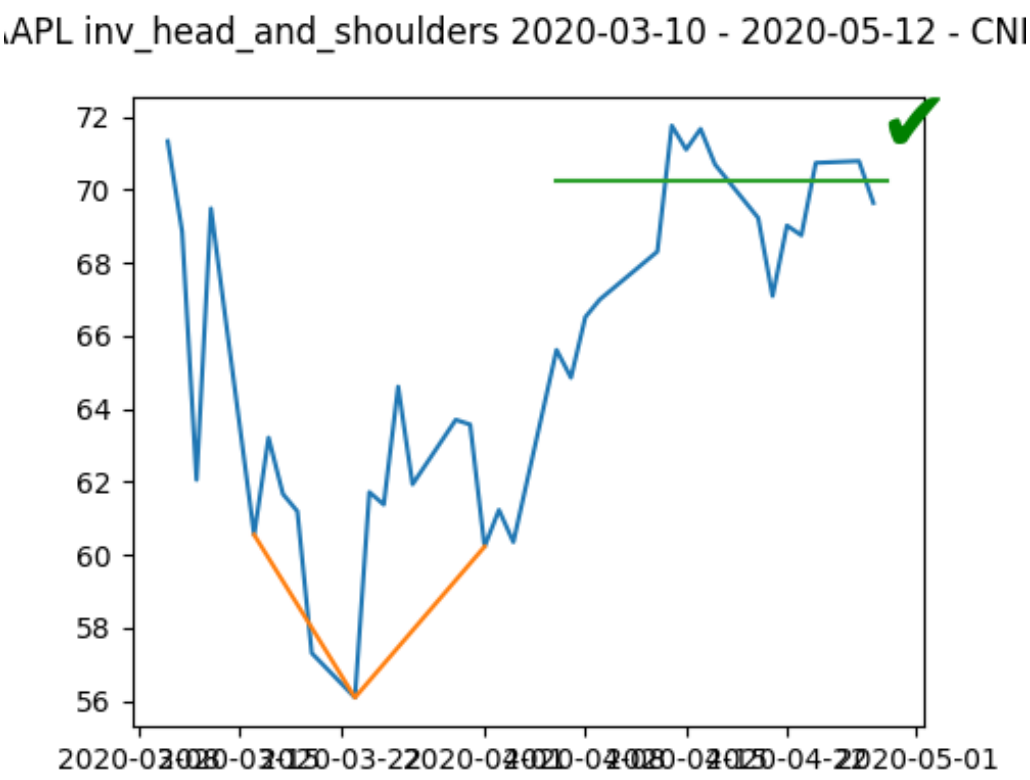


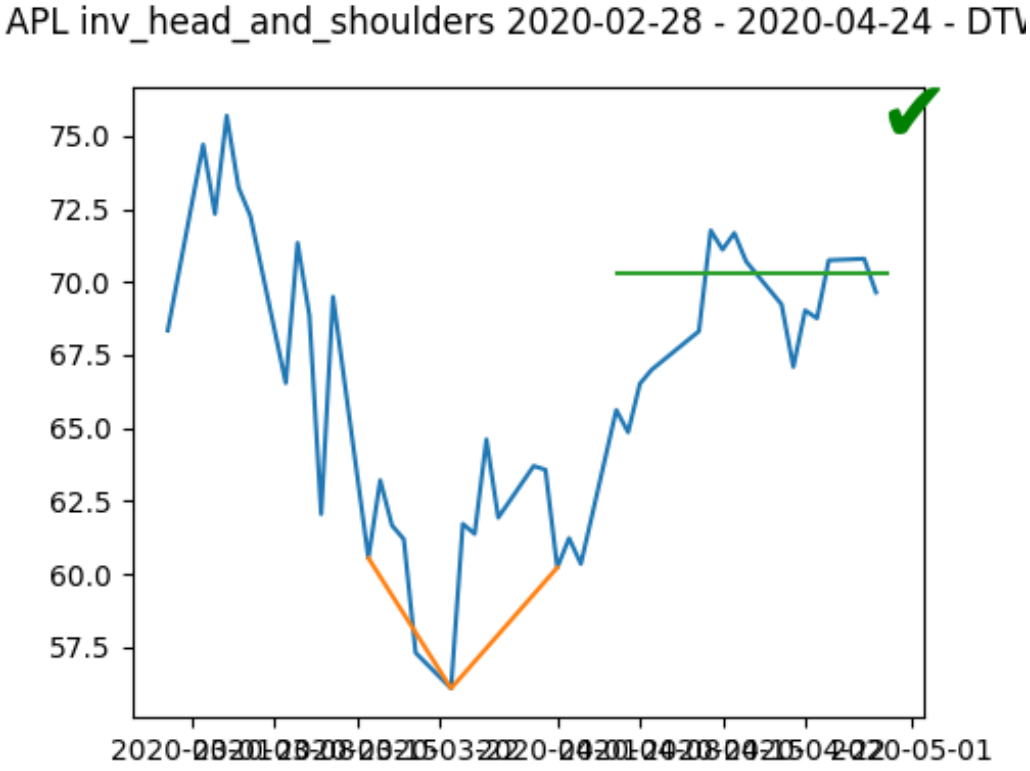

APL inv_head_and_shoulders 2005-02-17 - 2005-04-15 - CNI
1.6
1.5
1.4
1.3
1.2
2005-02-15
2005-03-01
2005-03-15
2005-04-01
2005-04-15
2005-05-01
2005-05-15


APL inv_head_and_shoulders 2005-02-14 - 2005-04-19 - DTV
1.6
1.5
1.4
1.3
1.2
2005-02-15
2005-03-01
2005-03-15
2005-04-01
2005-04-15
2005-05-01
2005-05-15

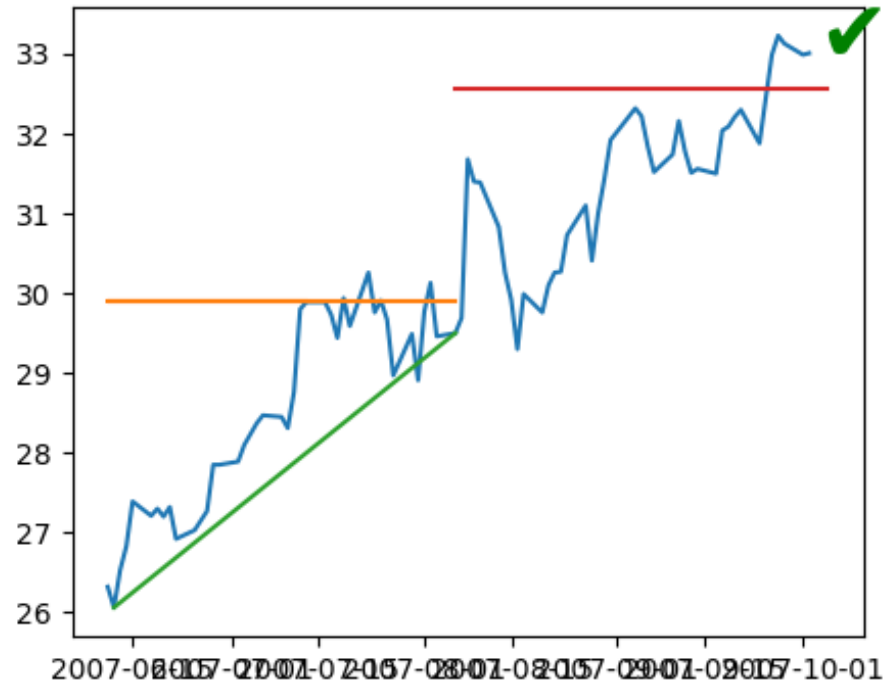

APL inv_head_and_shoulders 2018-11-29 - 2019-03-27 - CNI
46
44
42
40
38
36
2018-12-26
2019-01-26
2019-02-15
2019-03-15


APL inv_head_and_shoulders 2018-12-11 - 2019-02-07 - DTV
44
42
40
38
36
2018-12-25
2019-01-25
2019-02-15
2019-03-15

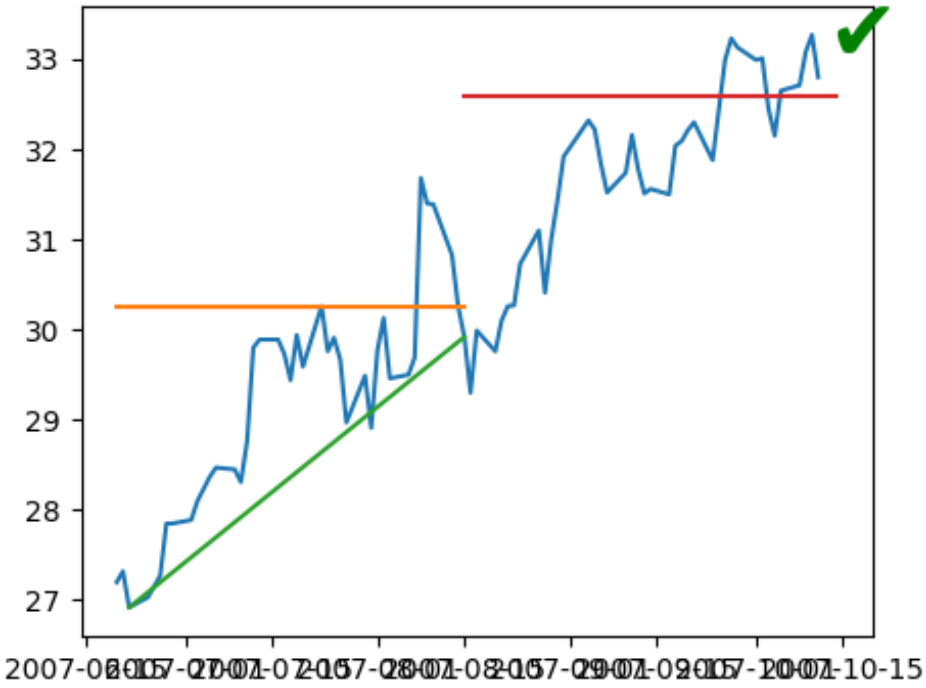

APL inv_head_and_shoulders 2020-03-10 - 2020-05-12 - CNI
72
70
68
66
64
62
60
58
56
2020-05-01


APL inv_head_and_shoulders 2020-02-28 - 2020-04-24 - DTW
75.0
72.5
70.0
67.5
65.0
62.5
60.0
57.5
2020-05-01

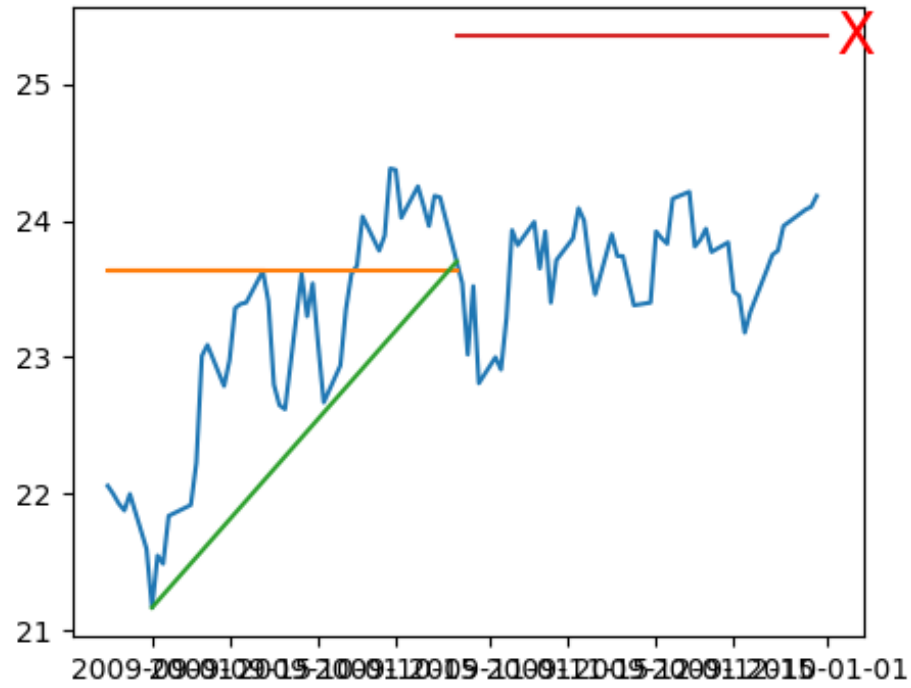
CSCO ascending_triangle 2007-06-11 - 2007-08-06 - CNN



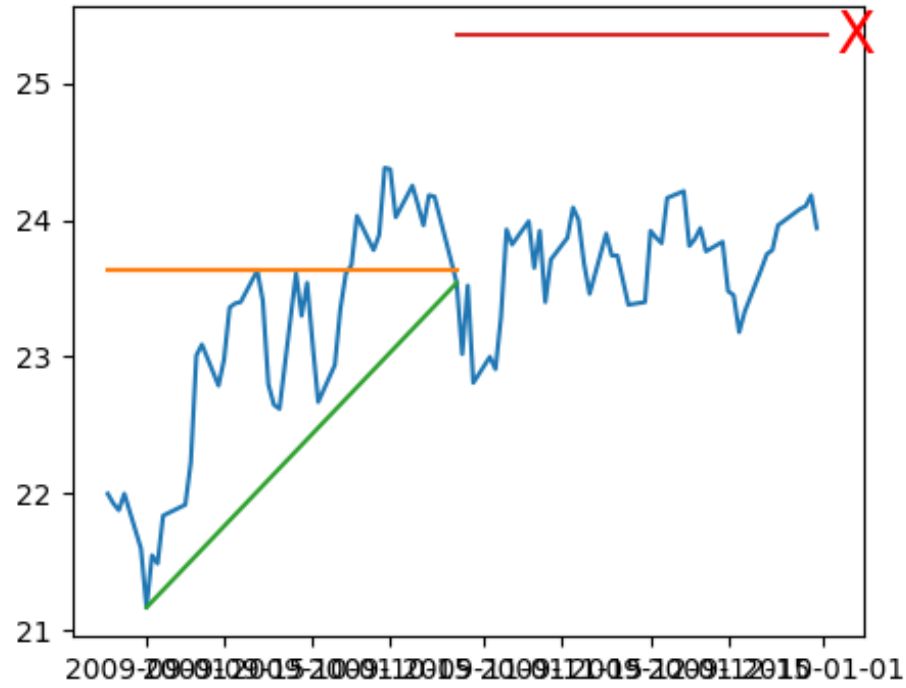
CSCO ascending_triangle 2007-06-20 - 2007-08-15 - DTW



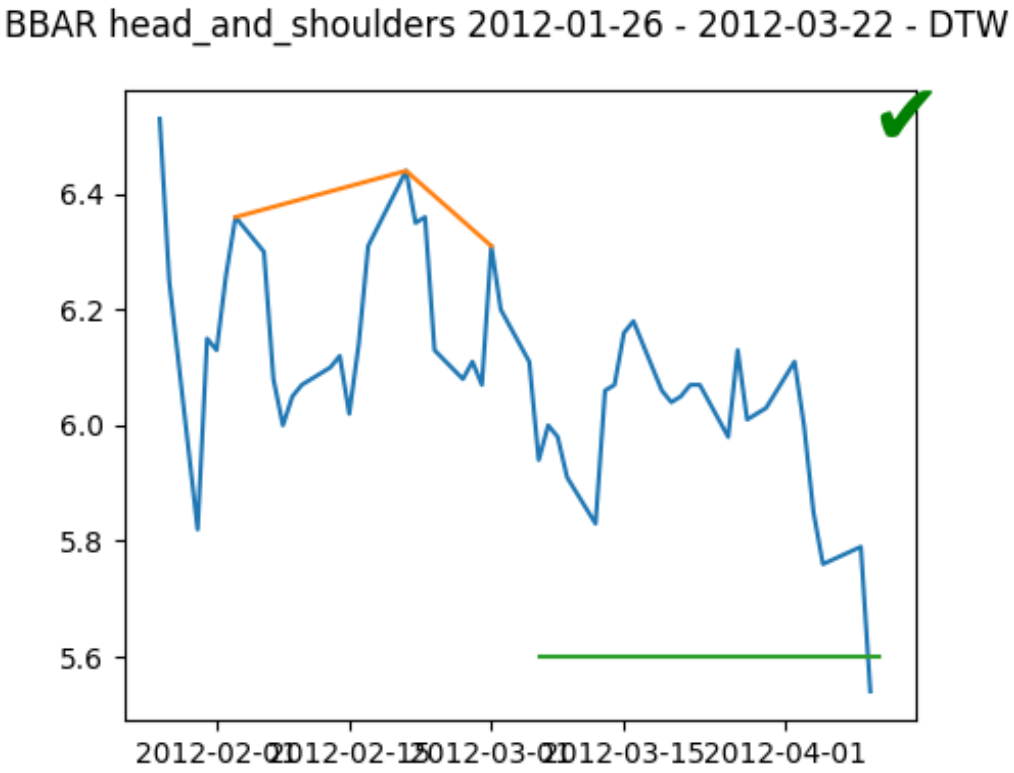
CSCO ascending_triangle 2009-08-24 - 2009-10-26 - CNN

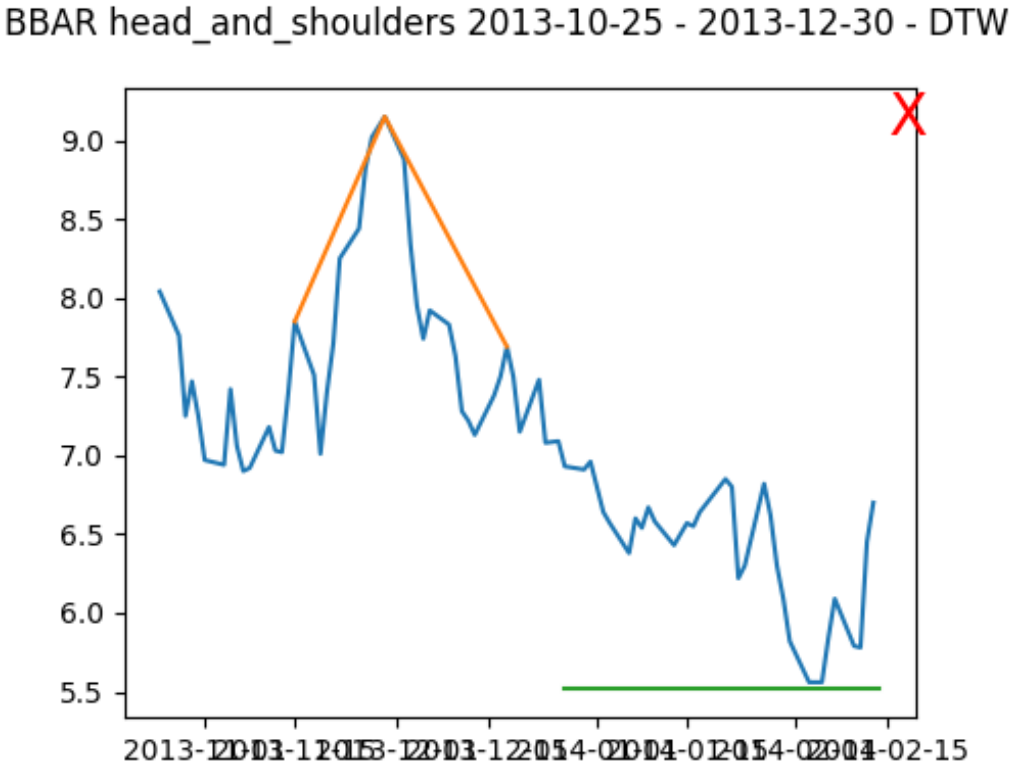


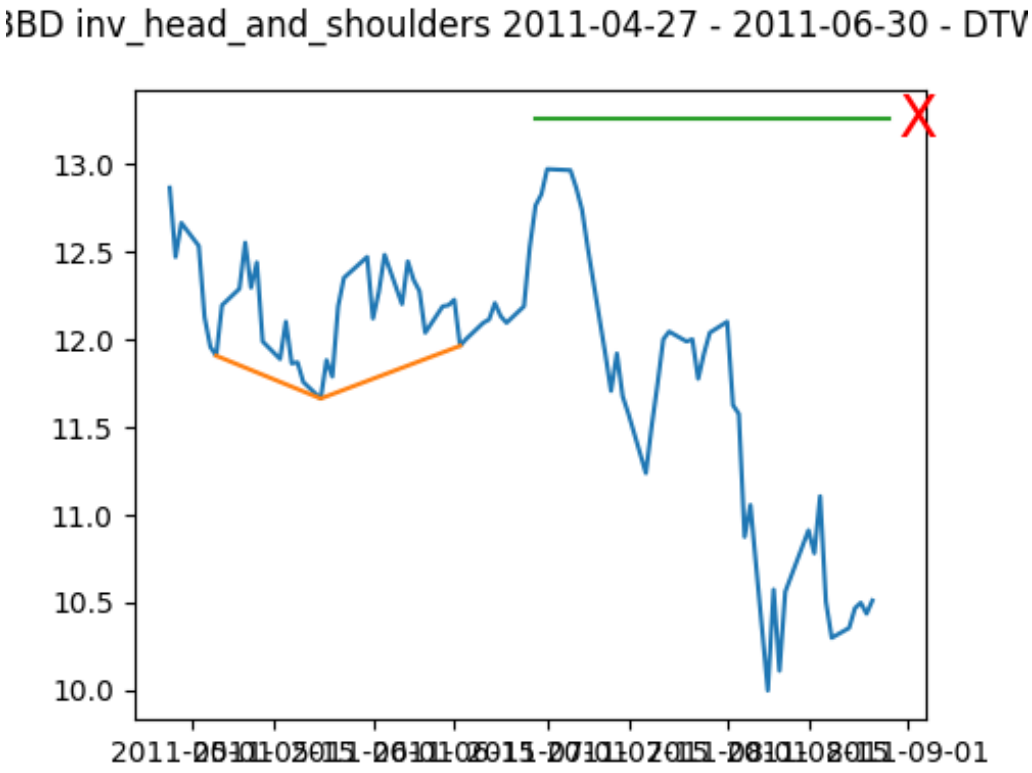
CSCO ascending_triangle 2009-08-25 - 2009-10-27 - DTW

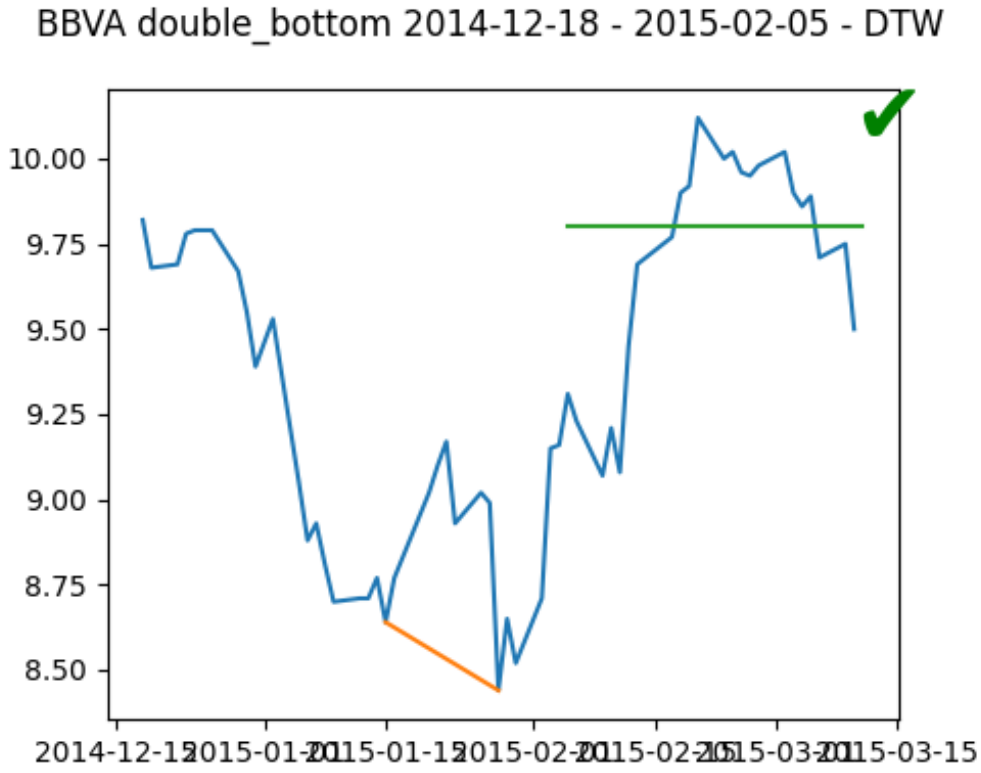


A.6. Resultados DTW sector financiero

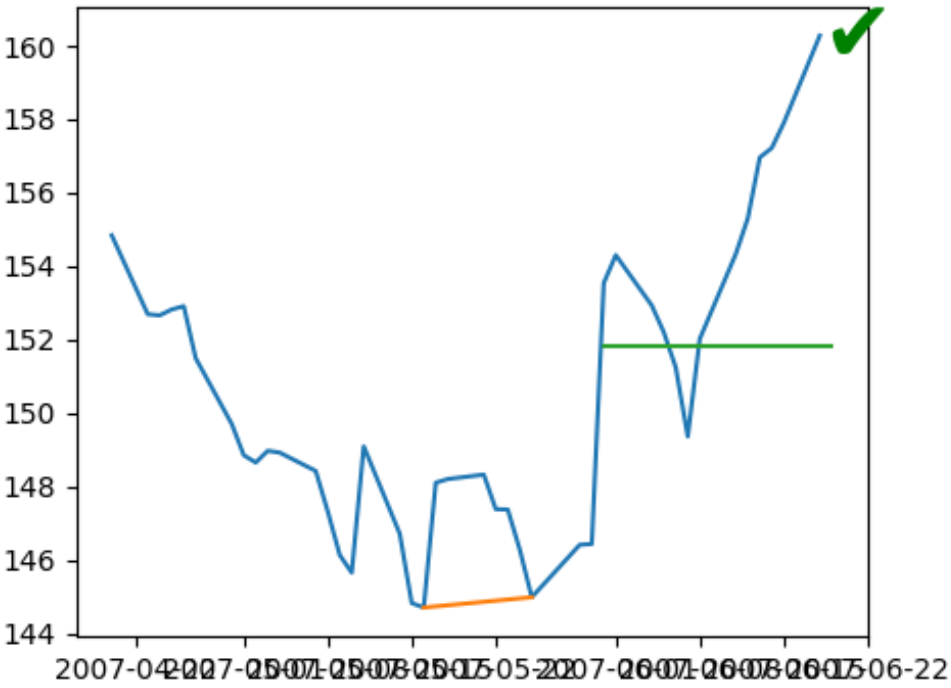




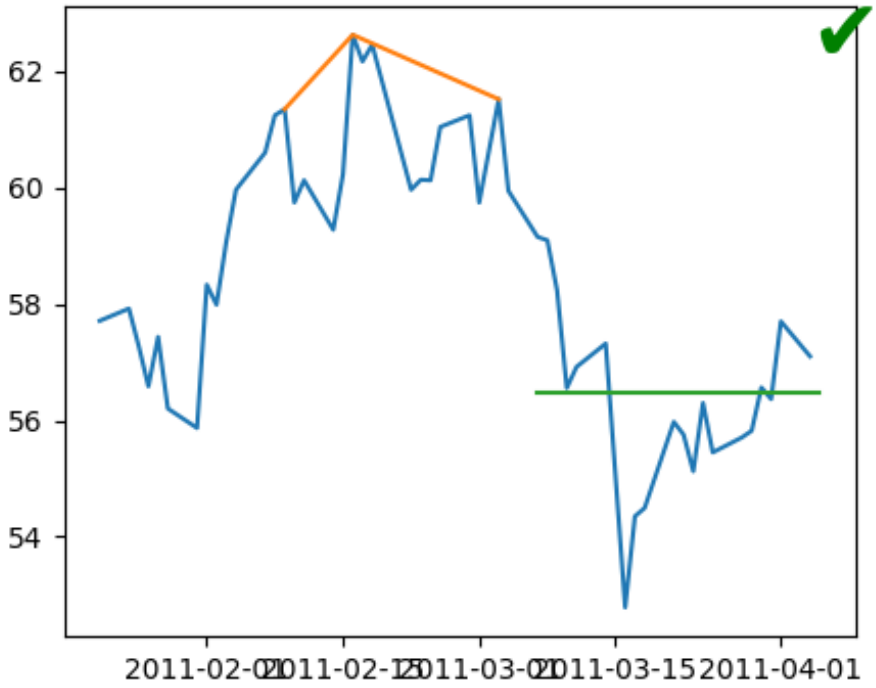

BD inv_head_and_shoulders 2011-04-27 - 2011-06-30 - DTV
13.0
12.5
12.0
11.5
11.0
10.5
10.0

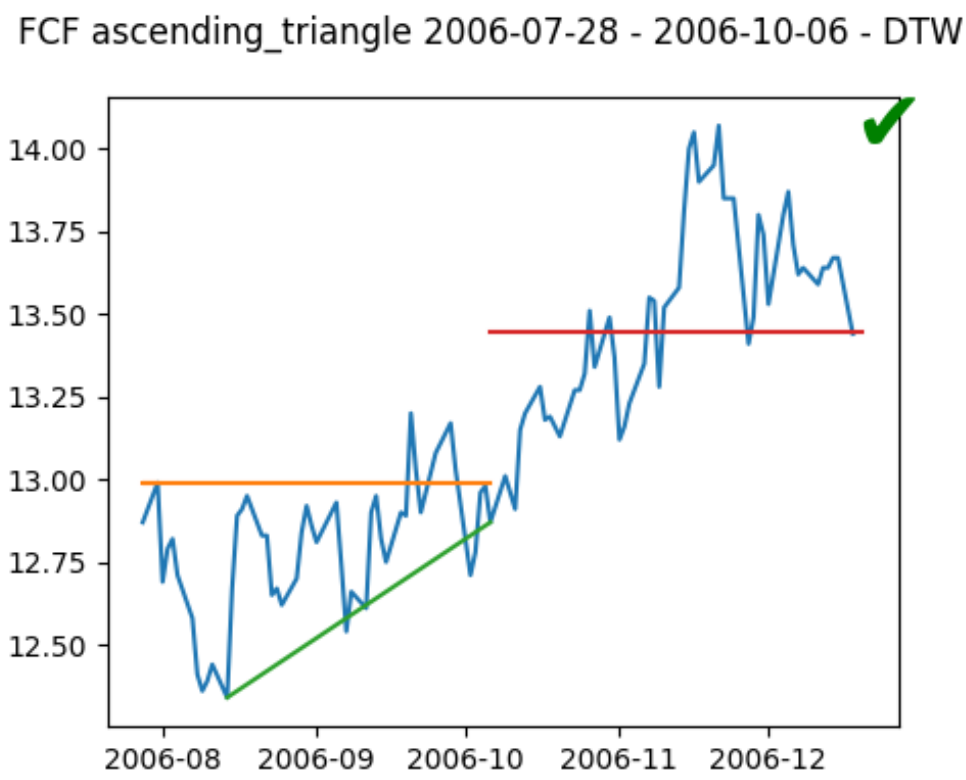

BBVA double_bottom 2014-12-18 - 2015-02-05 - DTW
10.00
9.75
9.50
9.25
9.00
8.75
8.50

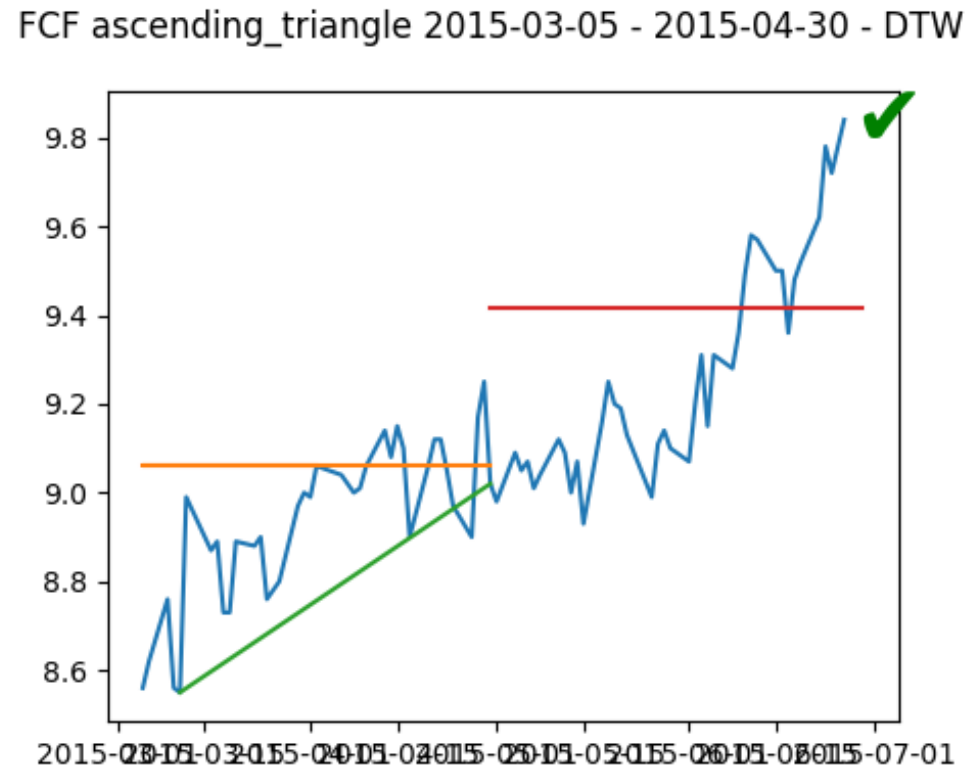
BLK double_bottom 2007-04-20 - 2007-06-06 - DTW



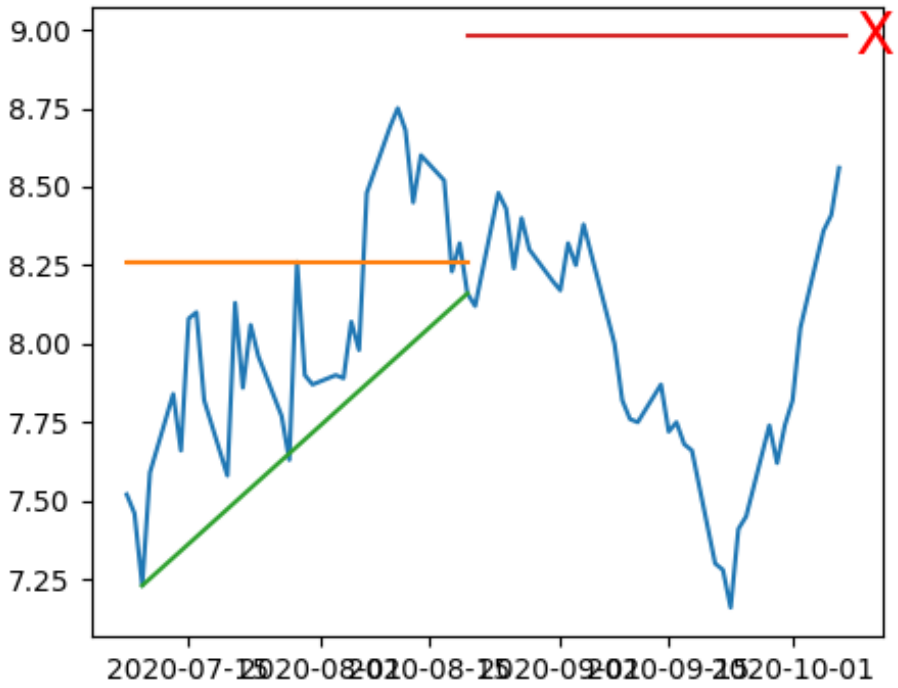
DB head_and_shoulders 2011-01-21 - 2011-03-18 - DTW



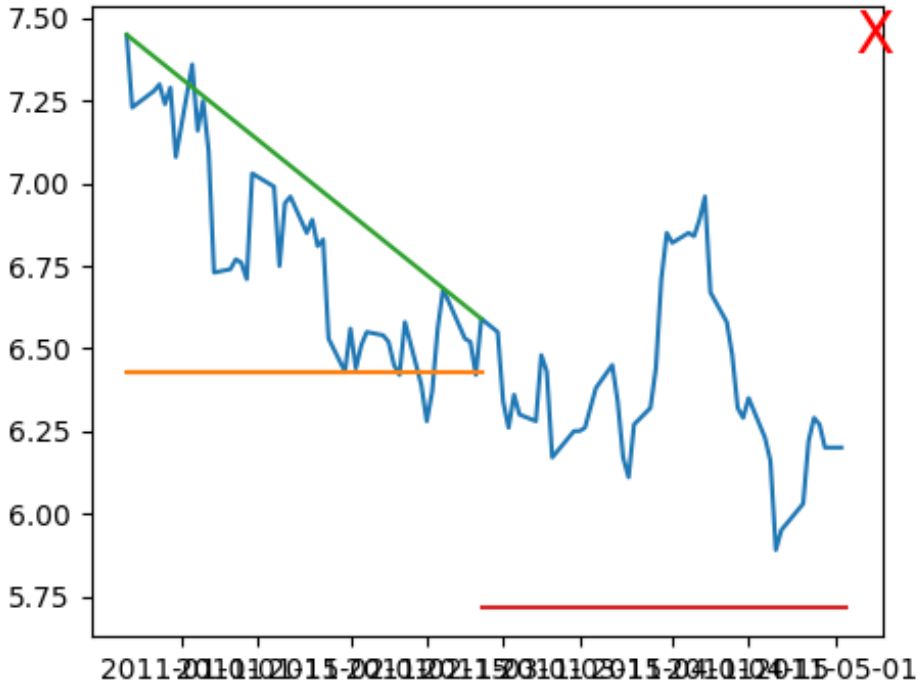

FCF ascending_triangle 2006-07-28 - 2006-10-06 - DTW
14.00
13.75
13.50
13.25
13.00
12.75
12.50
2006-08
2006-09
2006-10
2006-11
2006-12


FCF ascending_triangle 2015-03-05 - 2015-04-30 - DTW
9.8
9.6
9.4
9.2
9.0
8.8
8.6

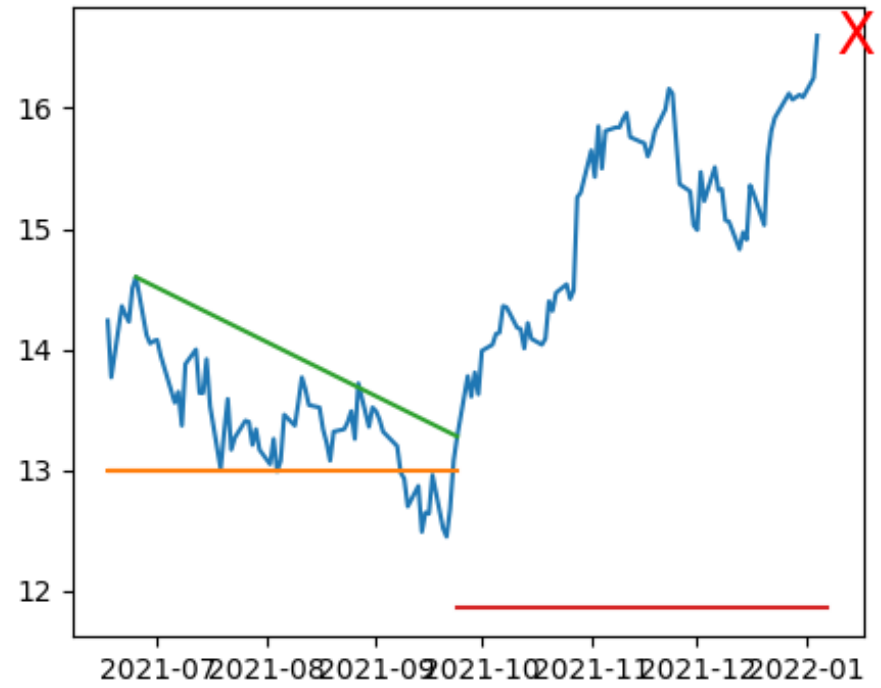
FCF ascending_triangle 2020-07-07 - 2020-08-20 - DTW



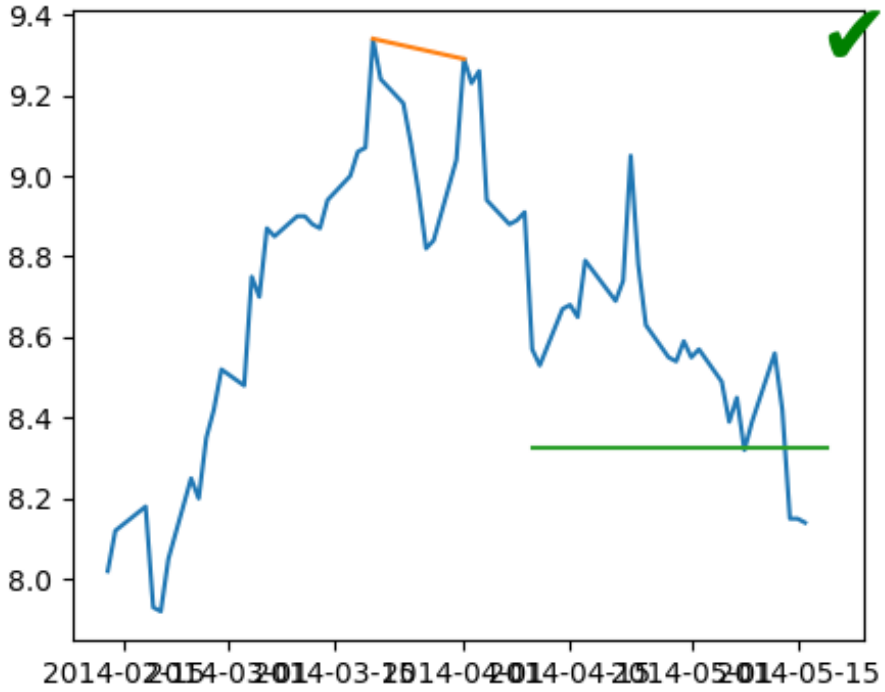
FCF descending_triangle 2010-12-22 - 2011-02-25 - DTW

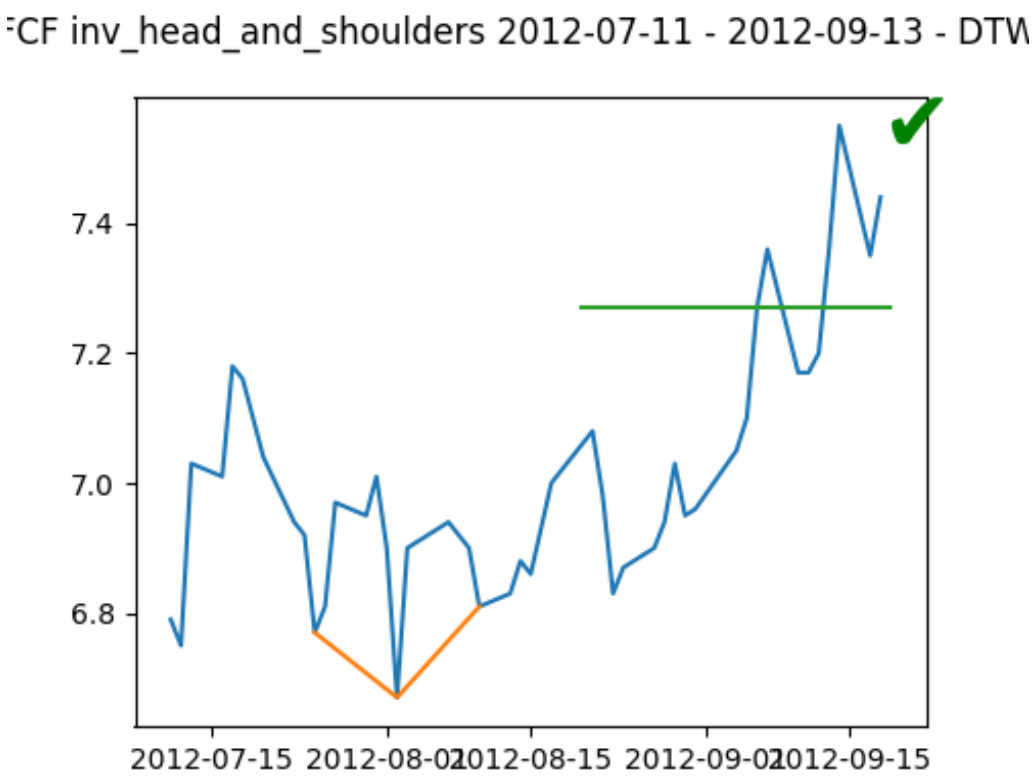


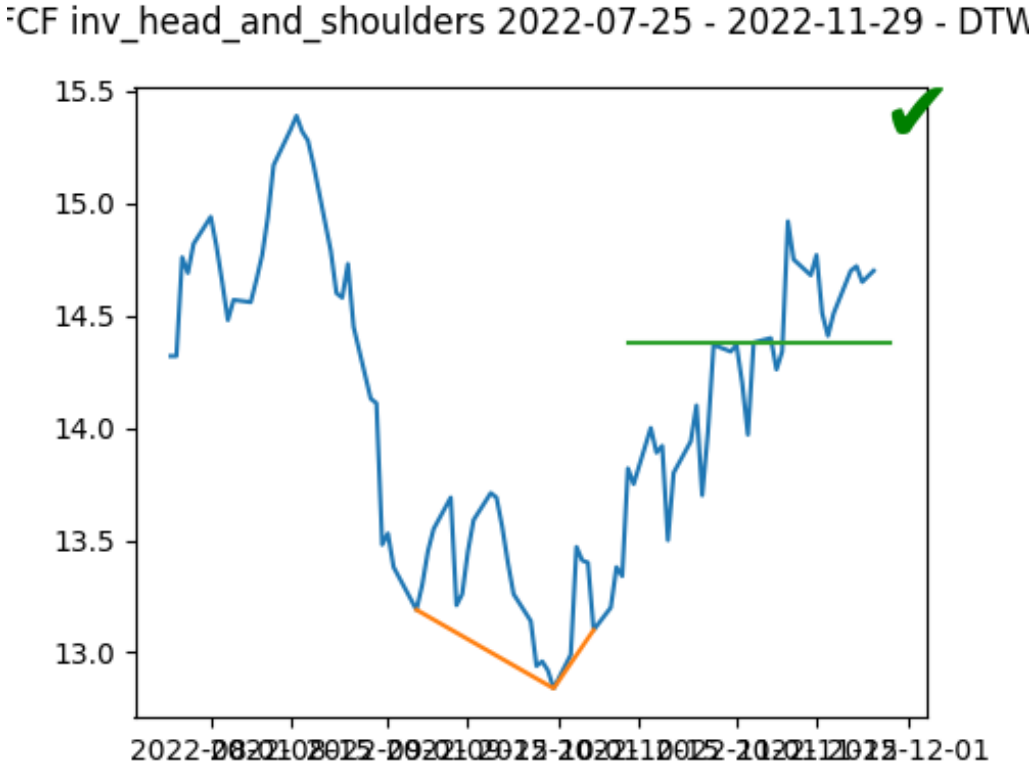
FCF descending_triangle 2021-06-17 - 2021-09-24 - DTW



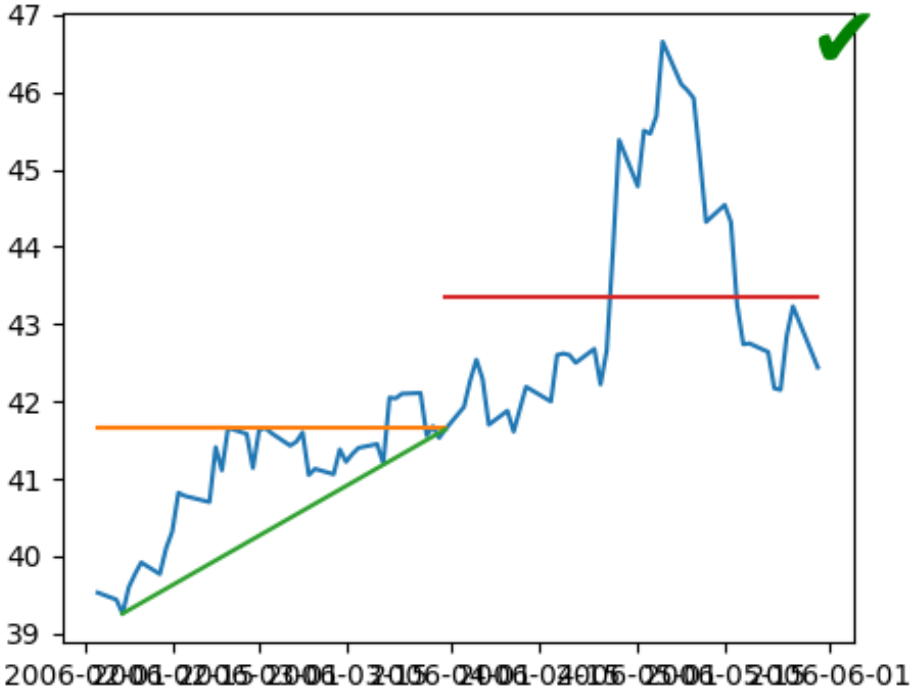
FCF double_top 2014-02-13 - 2014-04-17 - DTW



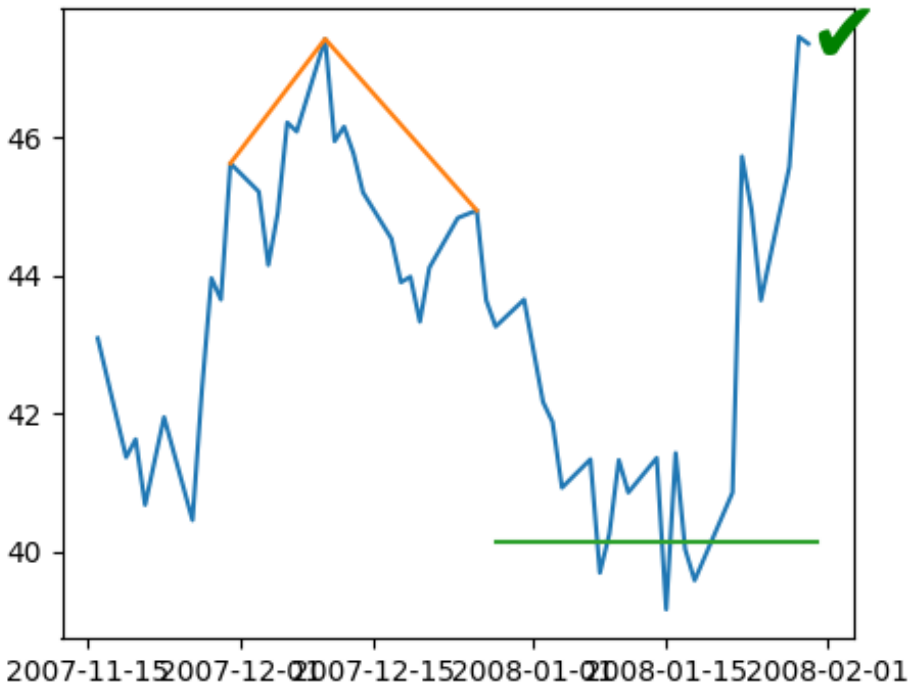

FCF inv_head_and_shoulders 2012-07-11 - 2012-09-13 - DTW
7.4
7.2
7.0
6.8
2012-07-15
2012-08-01
2012-08-15
2012-09-01
2012-09-15


FCF inv_head_and_shoulders 2022-07-25 - 2022-11-29 - DTW
15.5
15.0
14.5
14.0
13.5
13.0

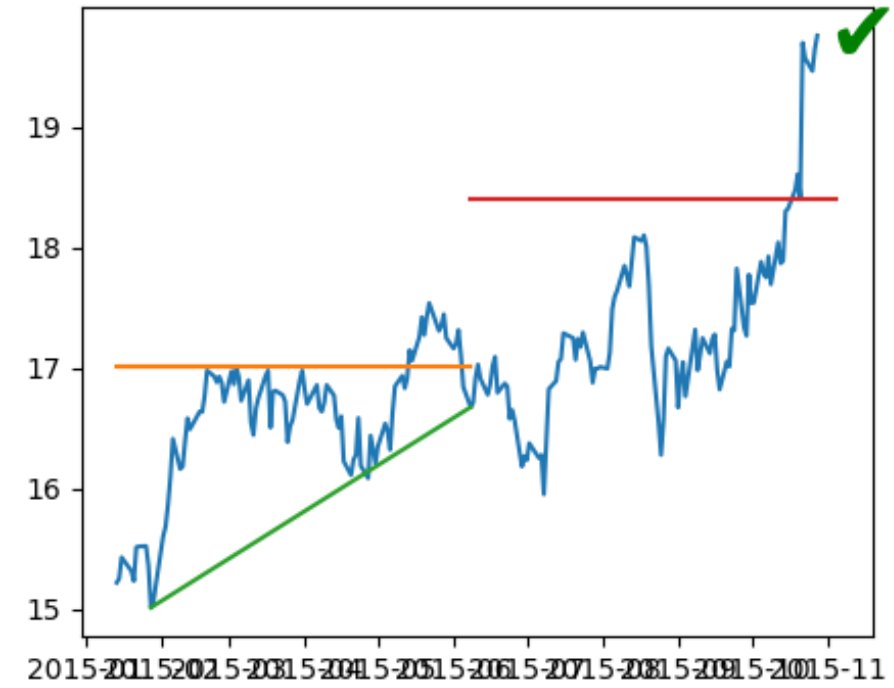
JPM ascending_triangle 2006-02-03 - 2006-03-31 - DTW



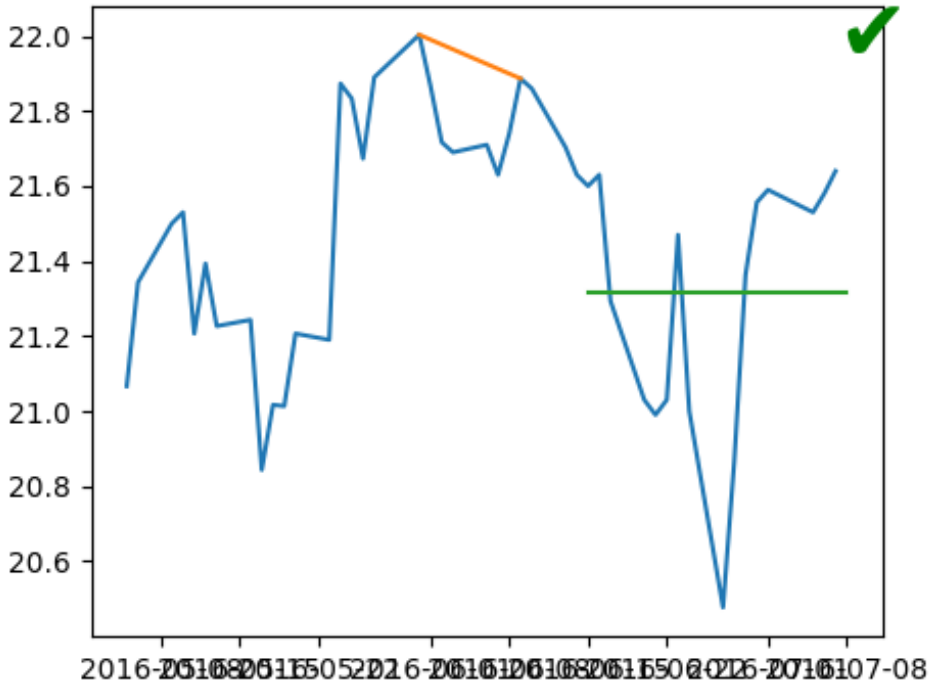
JPM head_and_shoulders 2007-11-16 - 2008-01-23 - DTW



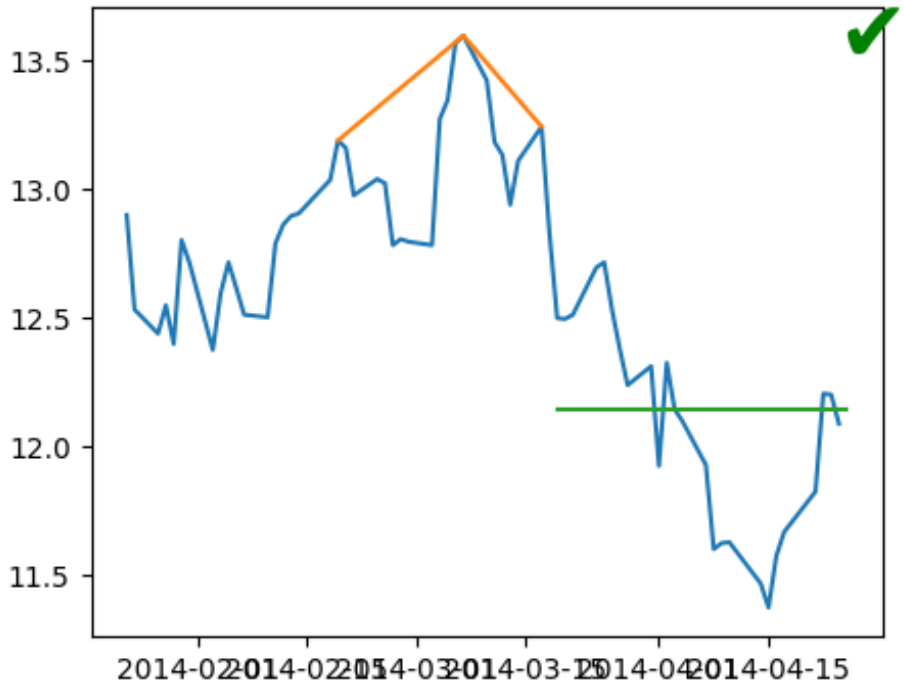
NDAQ ascending_triangle 2015-01-14 - 2015-06-08 - DTW



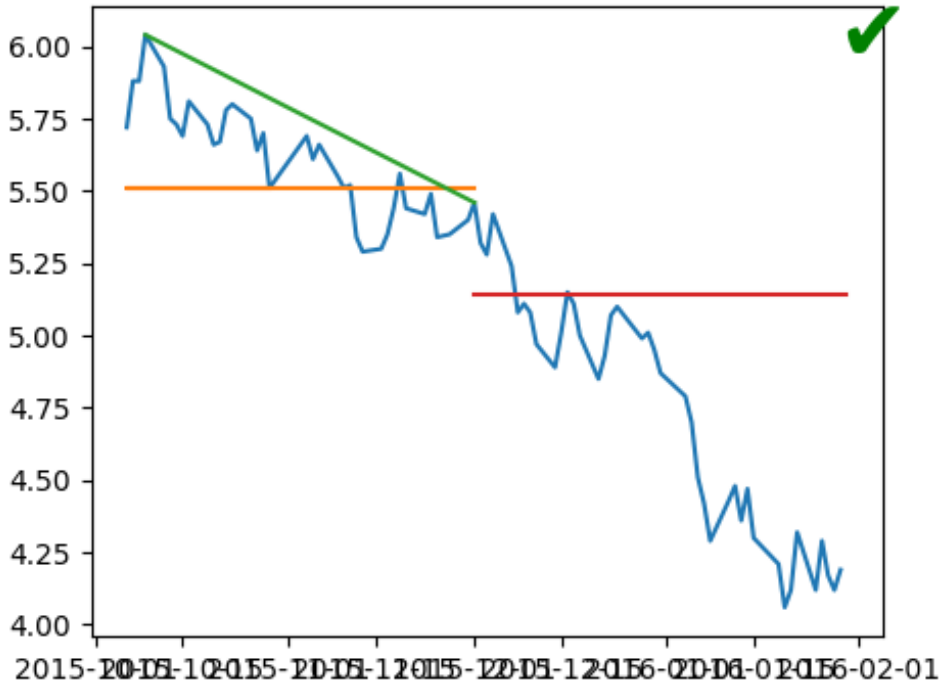
NDAQ double_top 2016-05-05 - 2016-06-30 - DTW

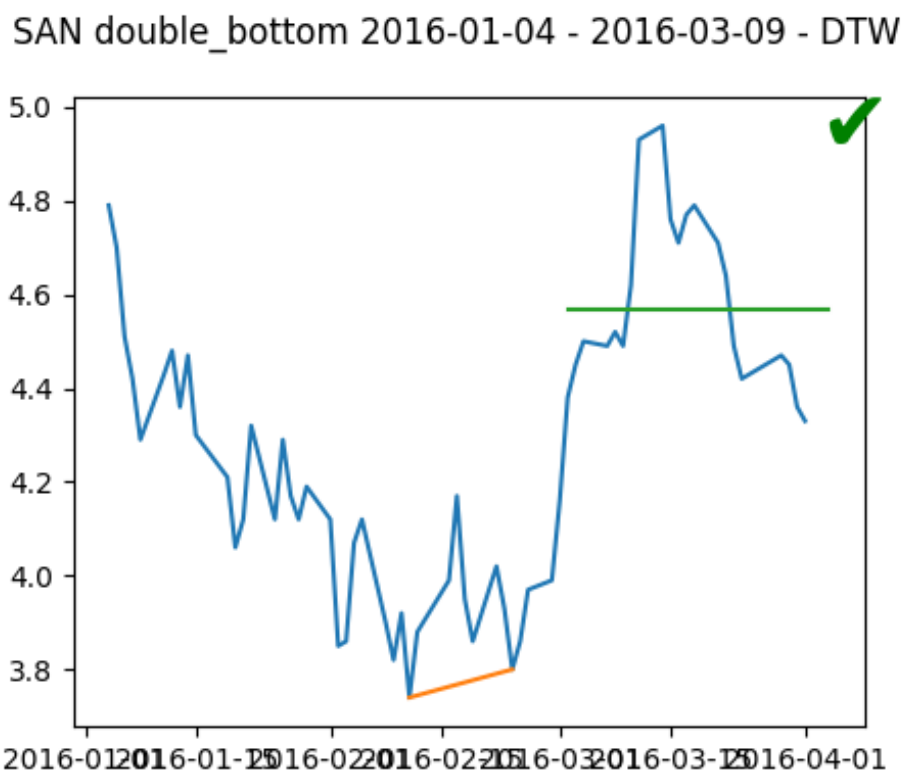


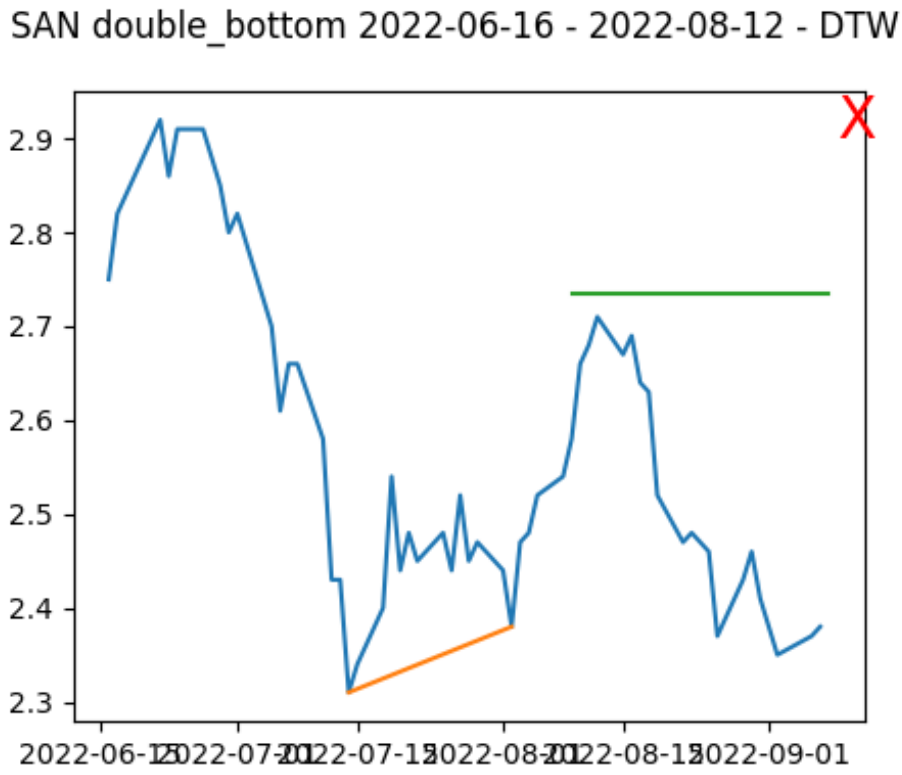
NDAQ head_and_shoulders 2014-01-23 - 2014-03-27 - DTW

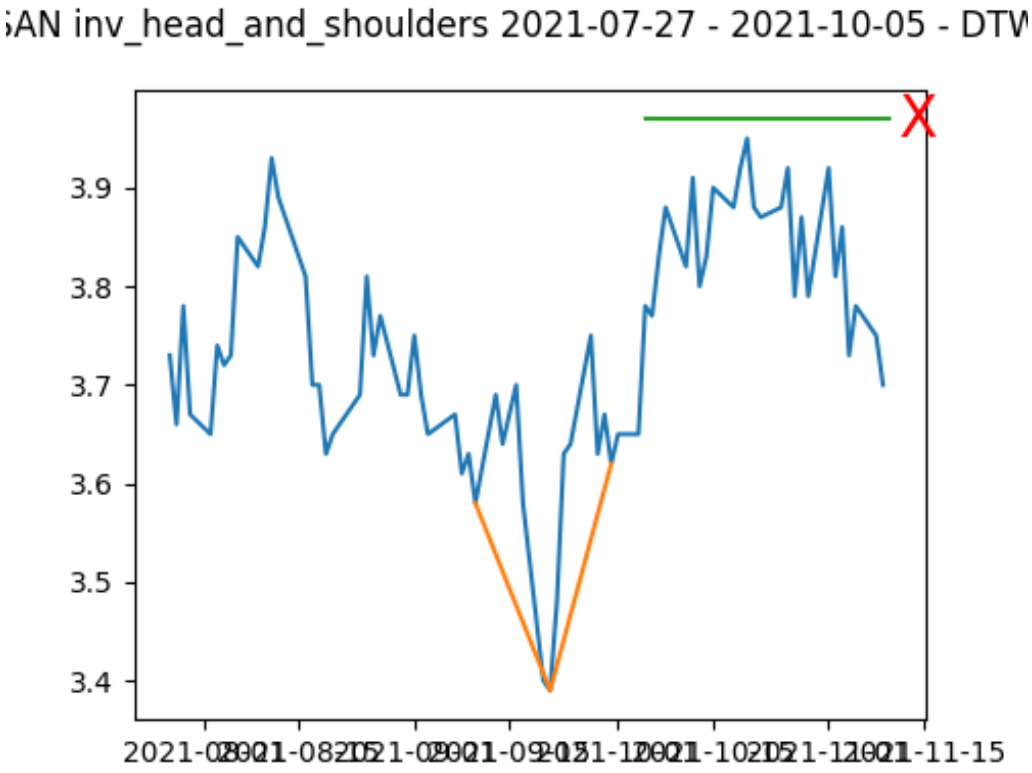


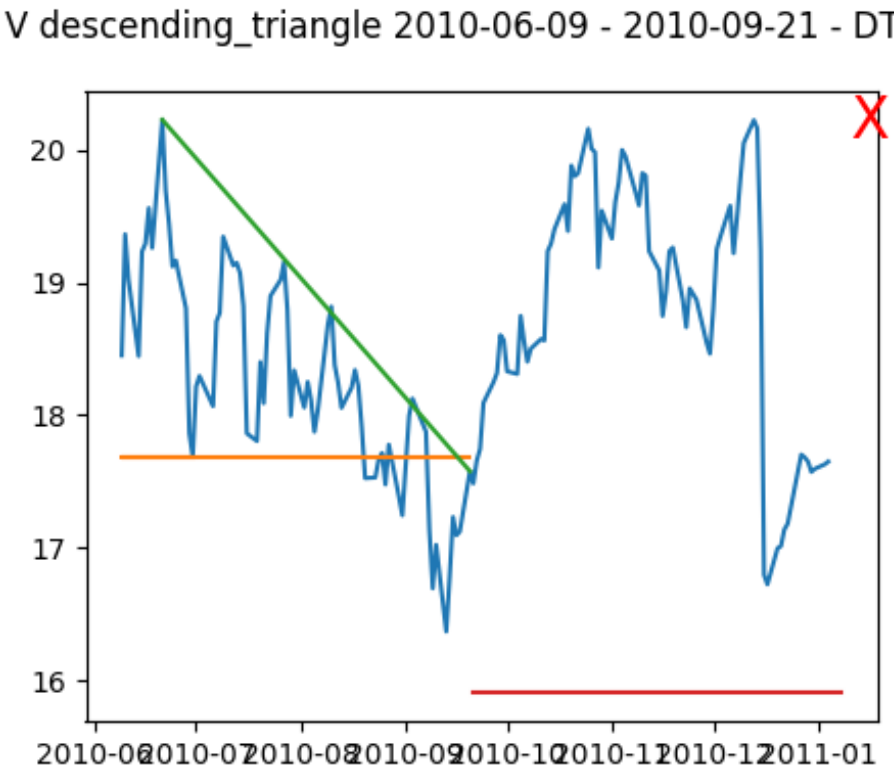
SAN descending_triangle 2015-10-06 - 2015-12-01 - DTW

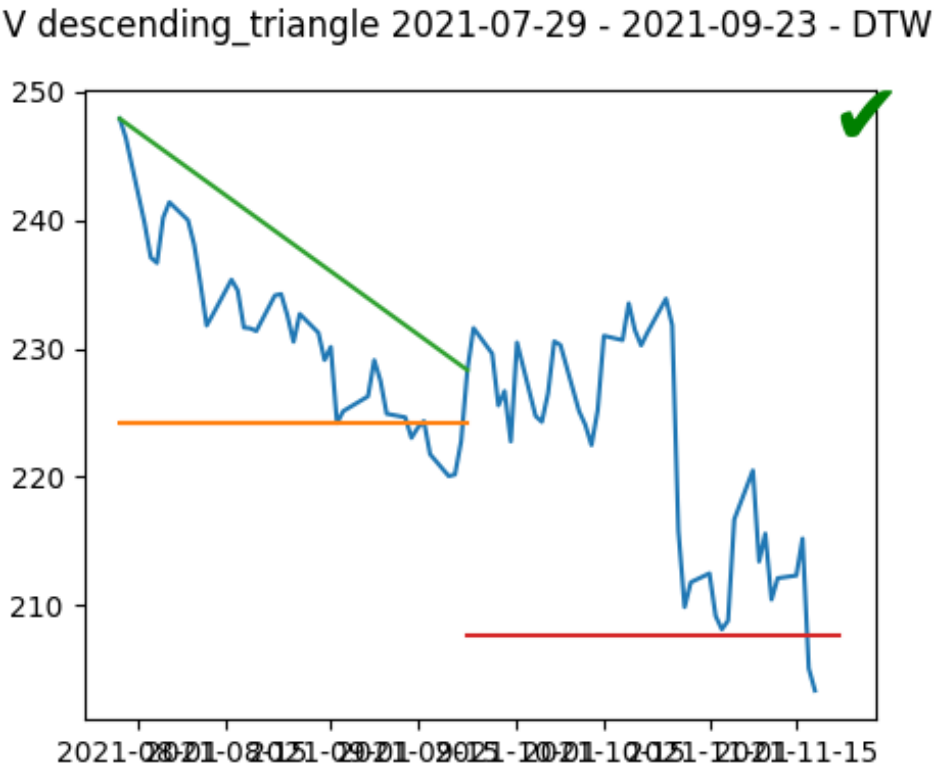


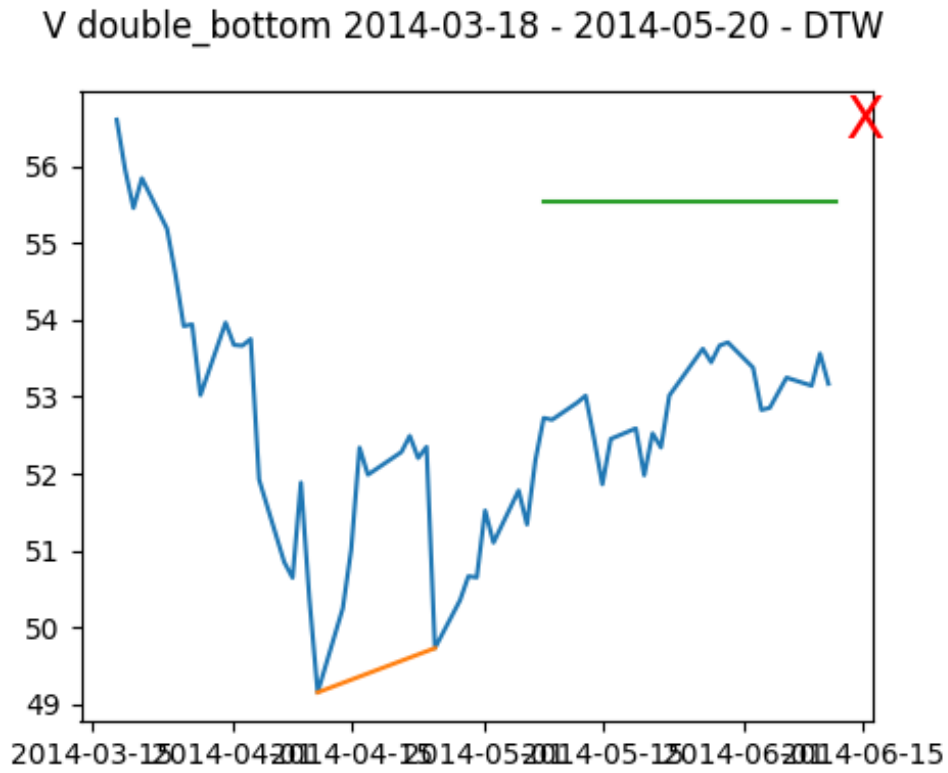

SAN double_bottom 2016-01-04 - 2016-03-09 - DTW
5.0
4.8
4.6
4.4
4.2
4.0
3.8
2016-04-01

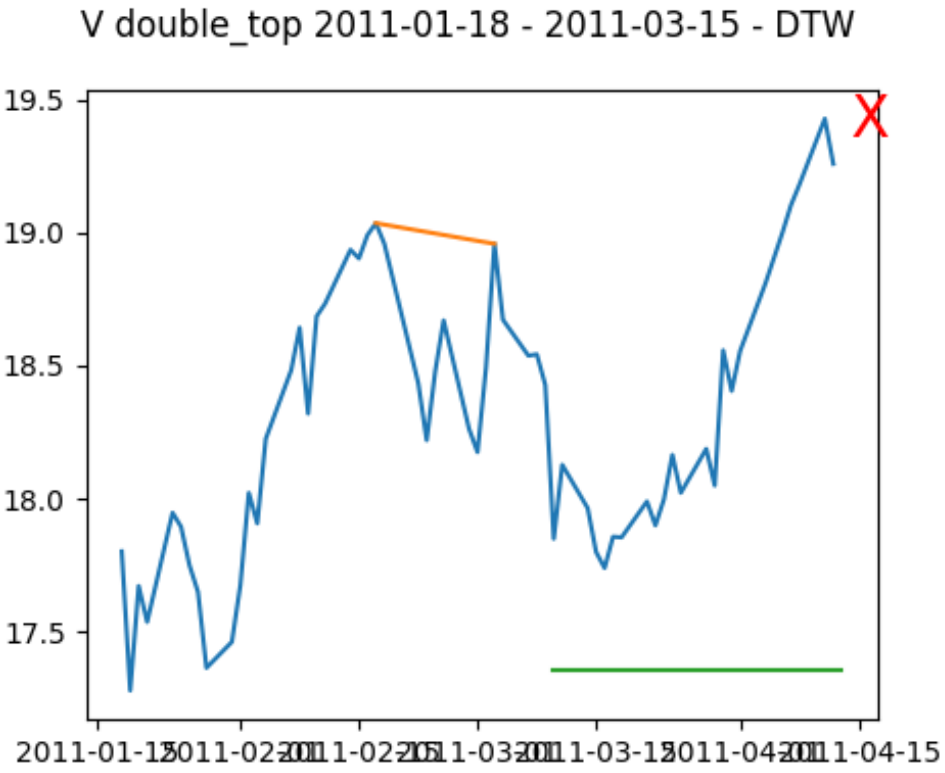

SAN double_bottom 2022-06-16 - 2022-08-12 - DTW
2.9
2.8
2.7
2.6
2.5
2.4
2.3
2022-09-01

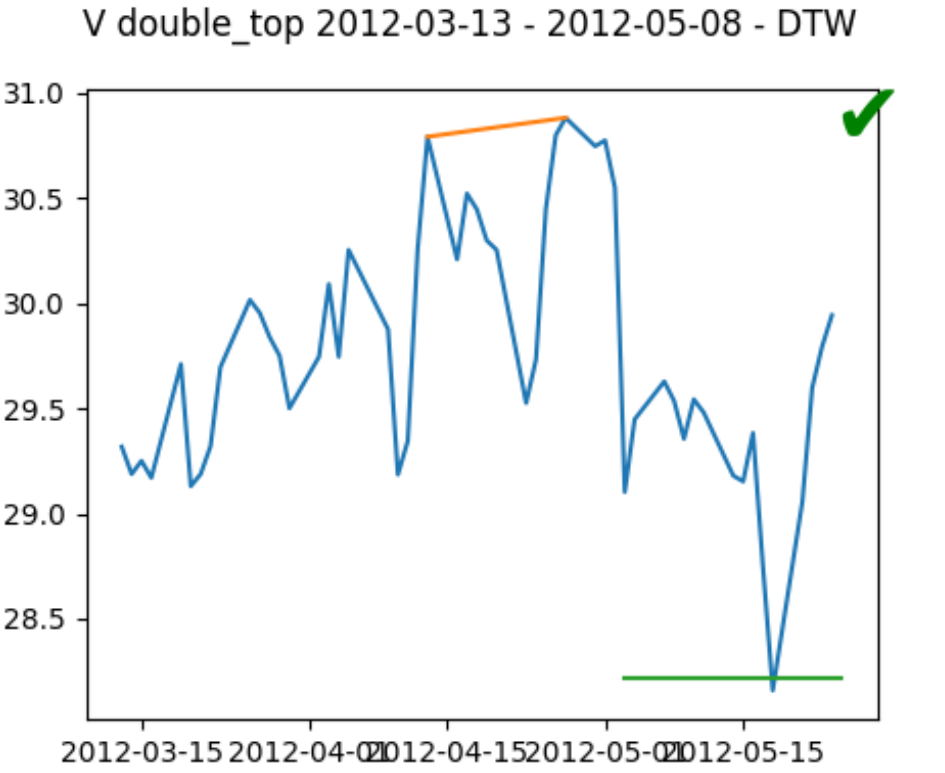

SAN inv_head_and_shoulders 2021-07-27 - 2021-10-05 - DTW
3.9
3.8
3.7
3.6
3.5
3.4
2021-11-15

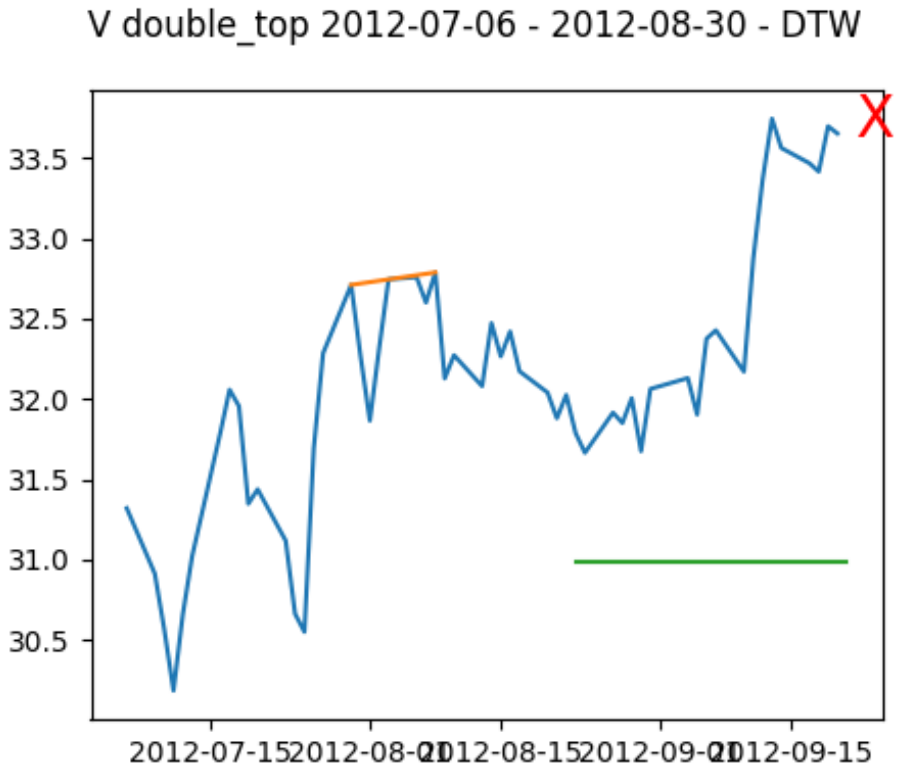

V descending_triangle 2010-06-09 - 2010-09-21 - DTW
20
19
18
17
16
2010-06
2010-07
2010-08
2010-09
2010-10
2010-11
2010-12
2011-01

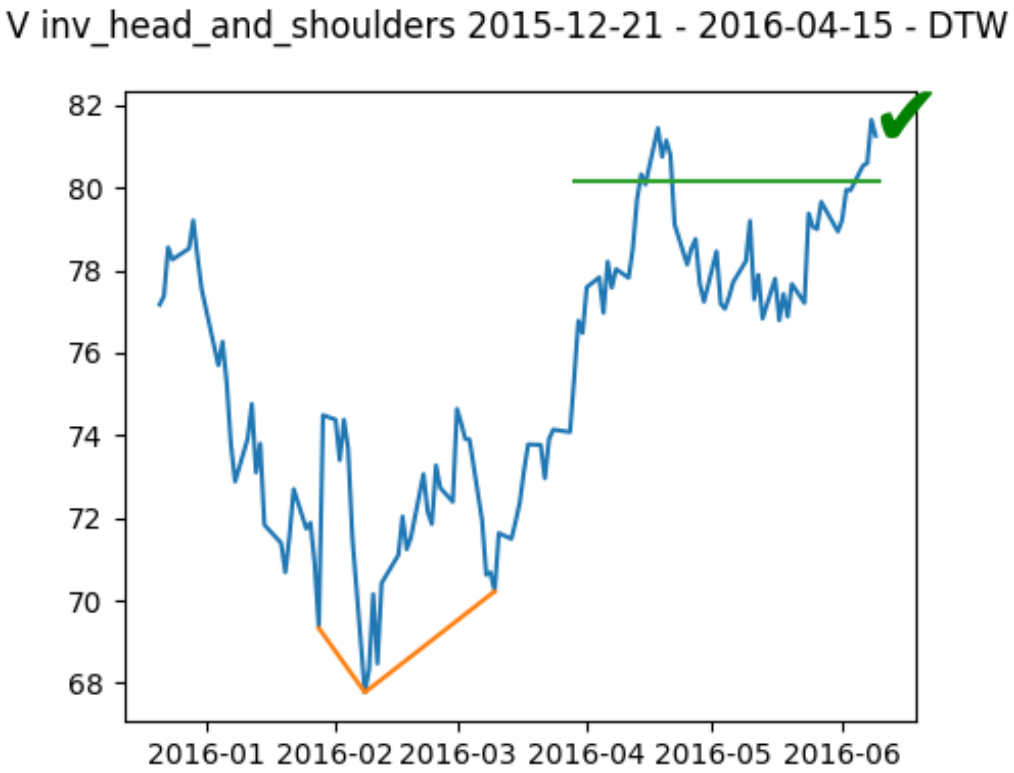

V descending_triangle 2021-07-29 - 2021-09-23 - DTW
250
240
230
220
210
2021-11-15


V double_bottom 2014-03-18 - 2014-05-20 - DTW
56
55
54
53
52
51
50
49
2014-06-15

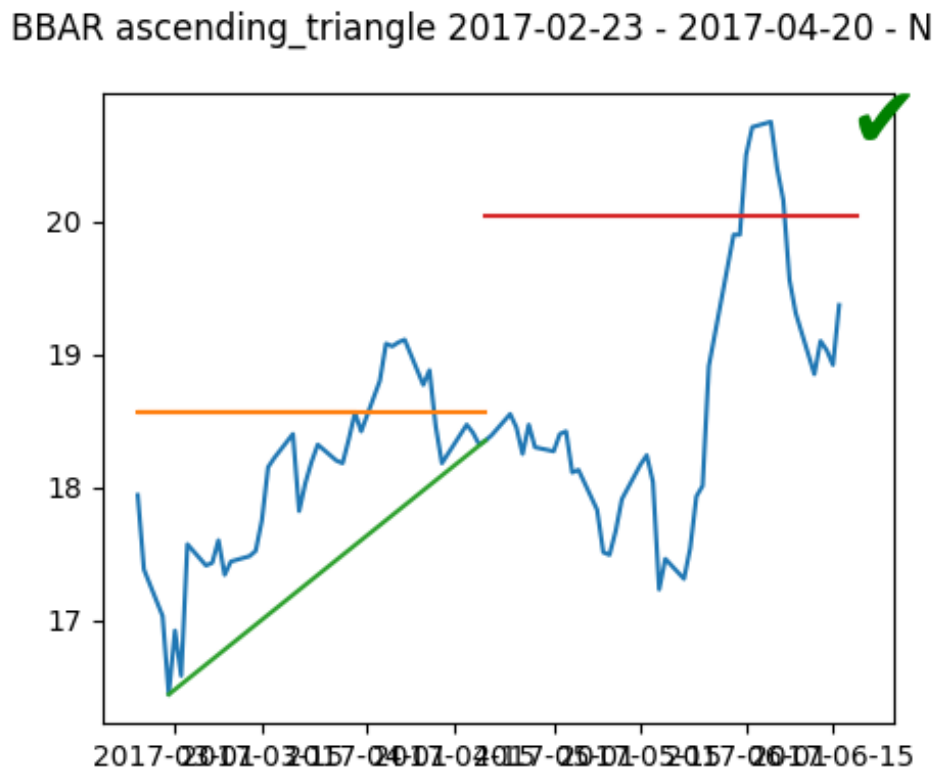

V double_top 2011-01-18 - 2011-03-15 - DTW
19.5
19.0
18.5
18.0
17.5
X
2011-01-15
2011-02-01
2011-02-15
2011-03-01
2011-03-15
2011-04-01
2011-04-15

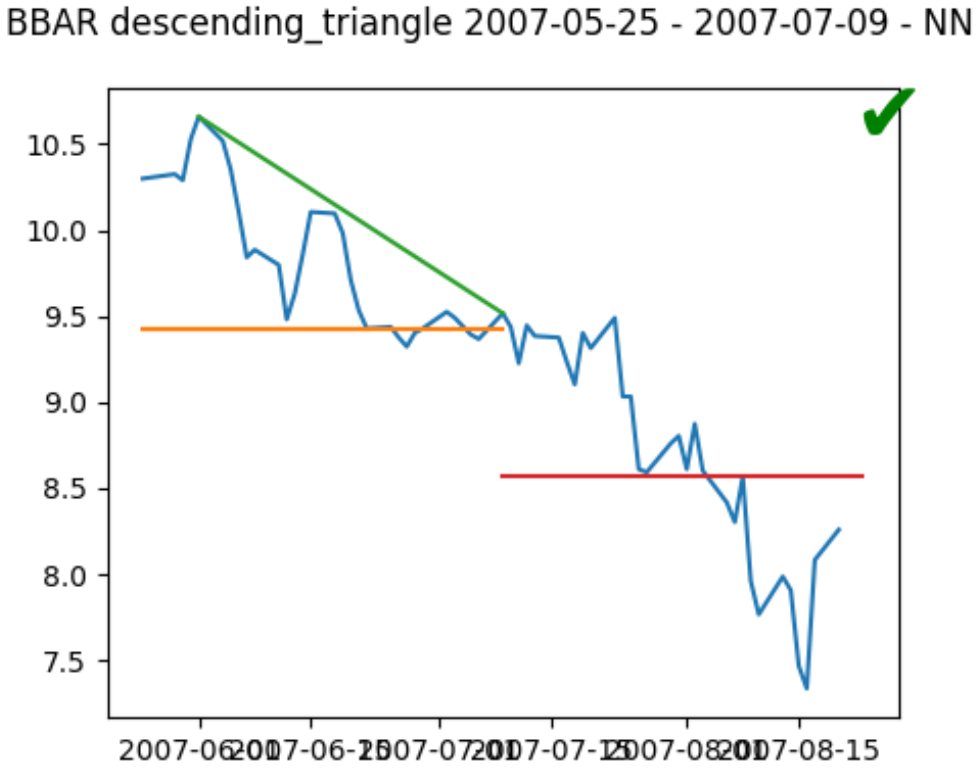

V double_top 2012-03-13 - 2012-05-08 - DTW
31.0
30.5
30.0
29.5
29.0
28.5
2012-03-15
2012-04-01
2012-04-15
2012-05-01
2012-05-15

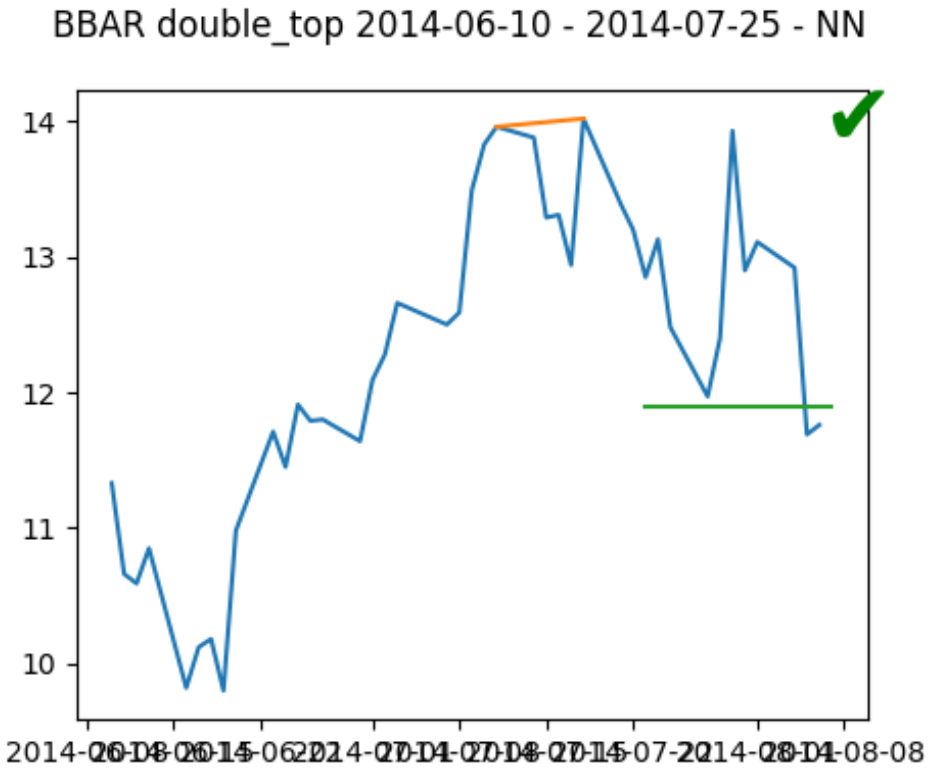

V double_top 2012-07-06 - 2012-08-30 - DTW
33.5
33.0
32.5
32.0
31.5
31.0
30.5
2012-07-15
2012-08-01
2012-08-15
2012-09-01
2012-09-15

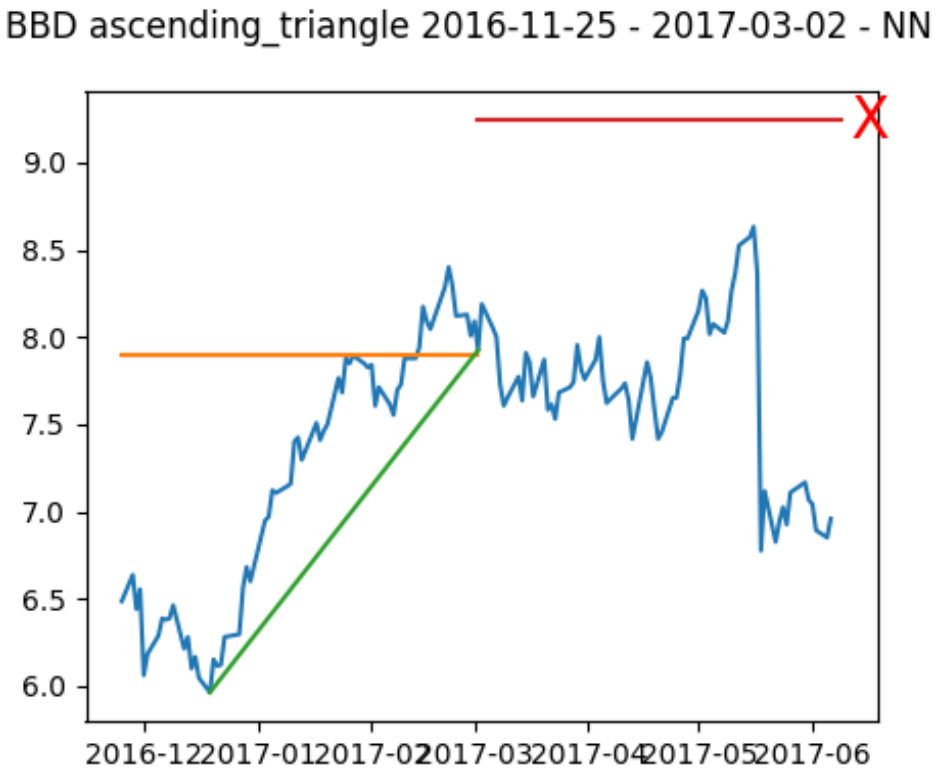

V inv_head_and_shoulders 2015-12-21 - 2016-04-15 - DTW
82
80
78
76
74
72
70
68
2016-01
2016-02
2016-03
2016-04
2016-05
2016-06

A.7. Resultados red neuronal sector financiero

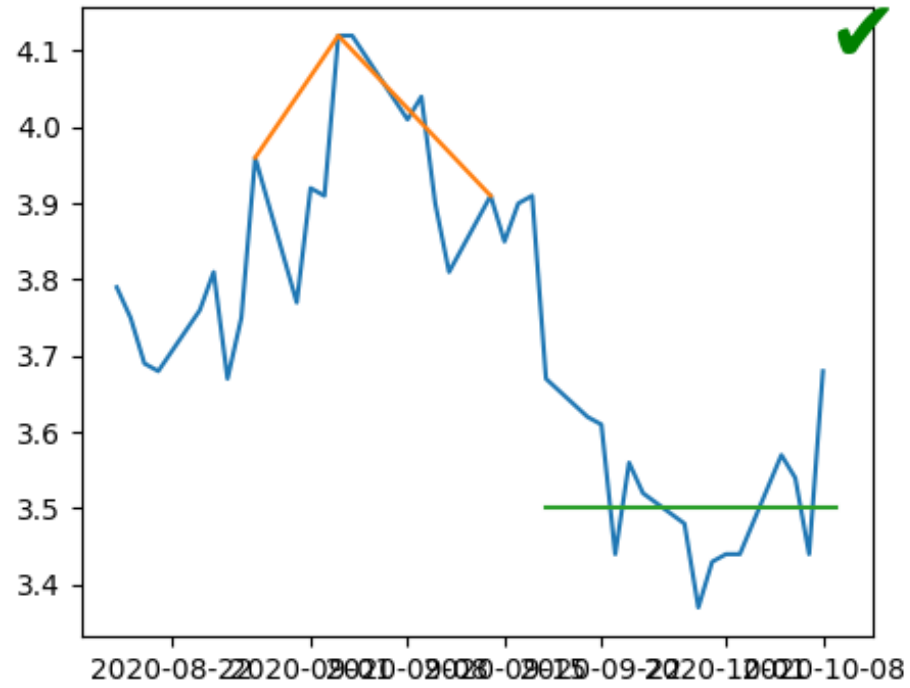




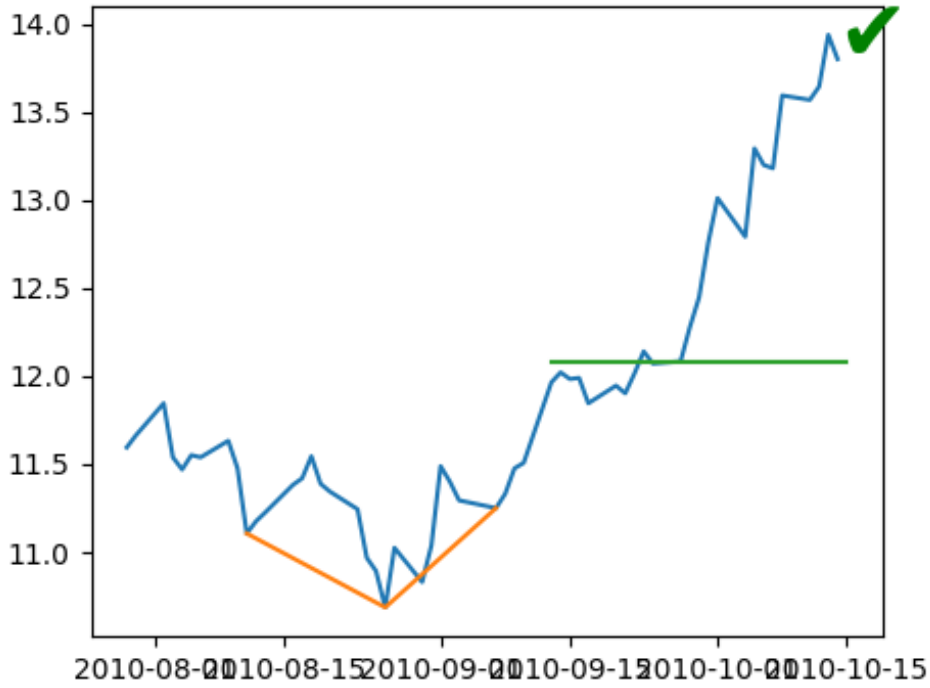

BBAR double_top 2014-06-10 - 2014-07-25 - NN
14
13
12
11
10


BBD ascending_triangle 2016-11-25 - 2017-03-02 - NN
9.0
8.5
8.0
7.5
7.0
6.5
6.0
2016-12
2017-01
2017-02
2017-03
2017-04
2017-05
2017-06

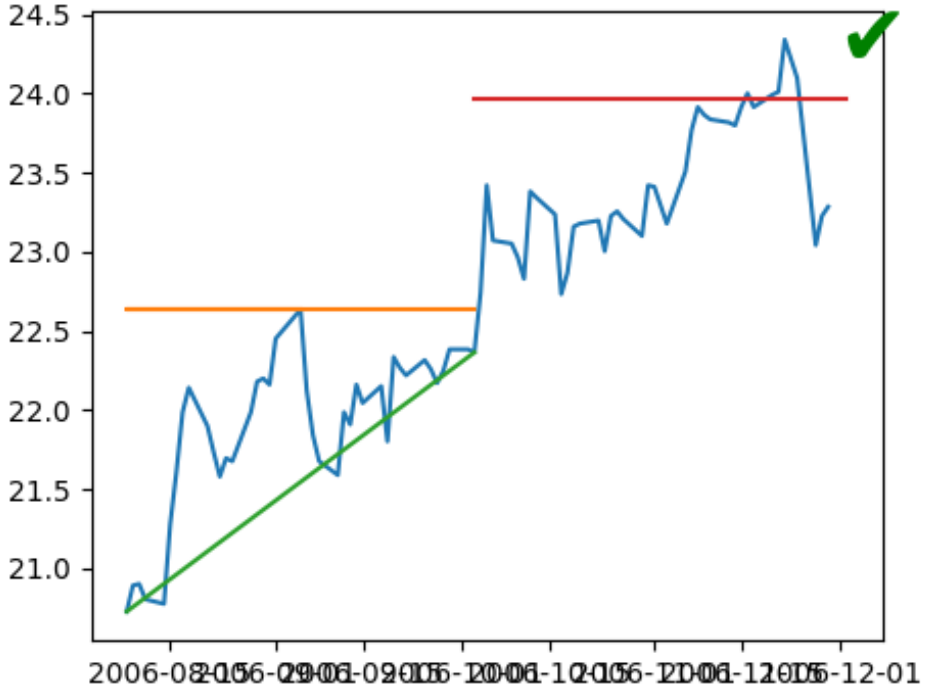
BBD head_and_shoulders 2020-08-18 - 2020-10-13 - NN



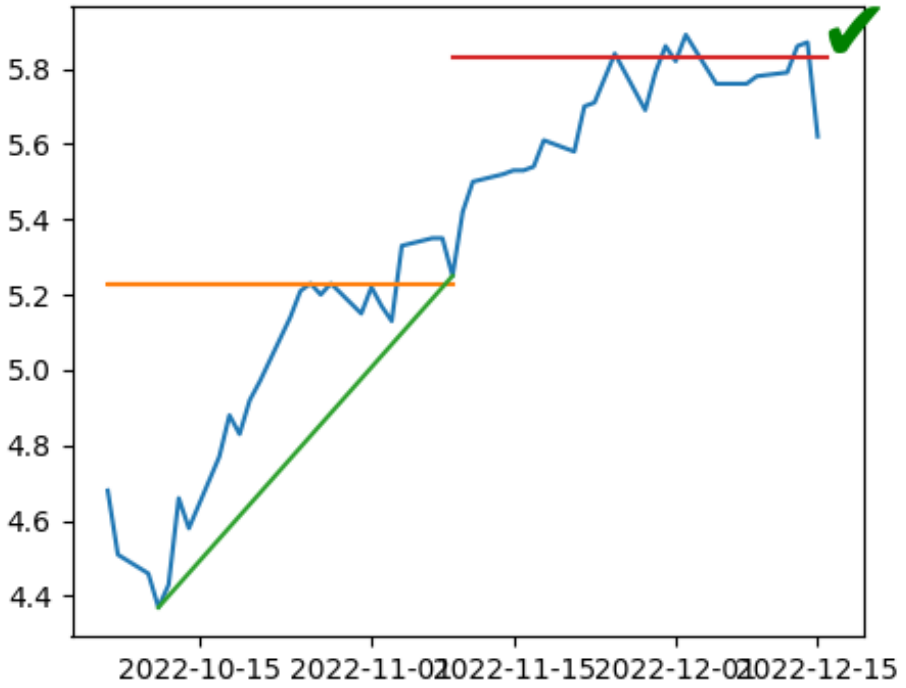
BBD inv_head_and_shoulders 2010-07-29 - 2010-09-23 - NN



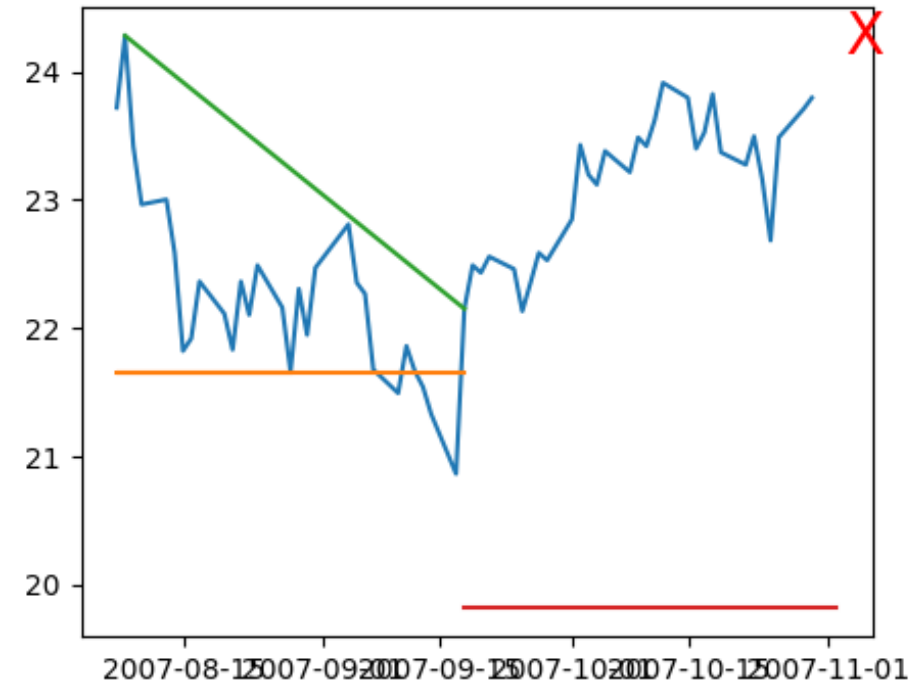
BBVA ascending_triangle 2006-08-08 - 2006-10-03 - NN



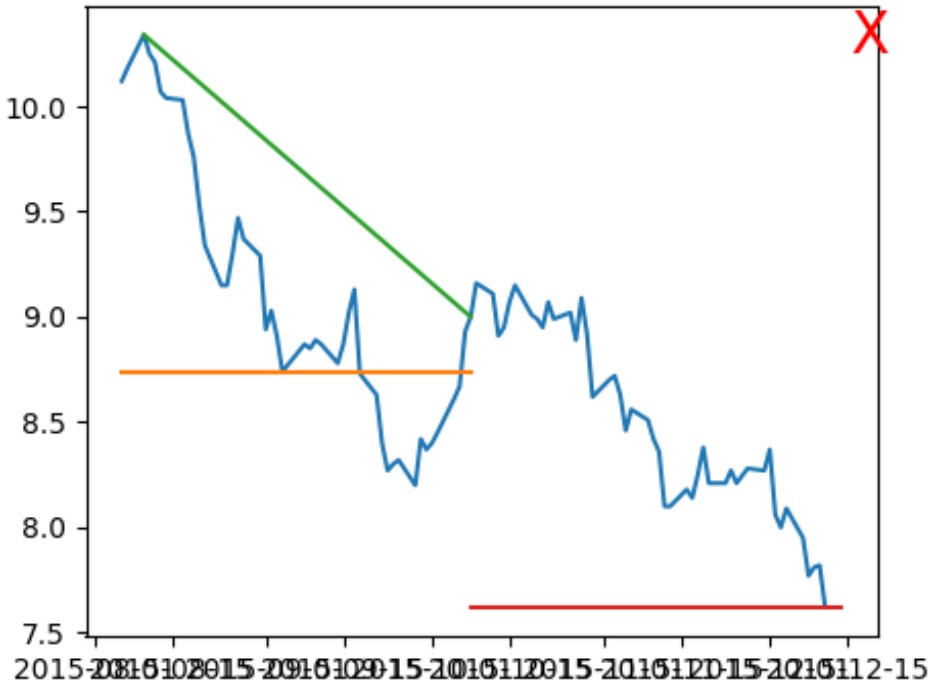
BBVA ascending_triangle 2022-10-06 - 2022-11-09 - NN



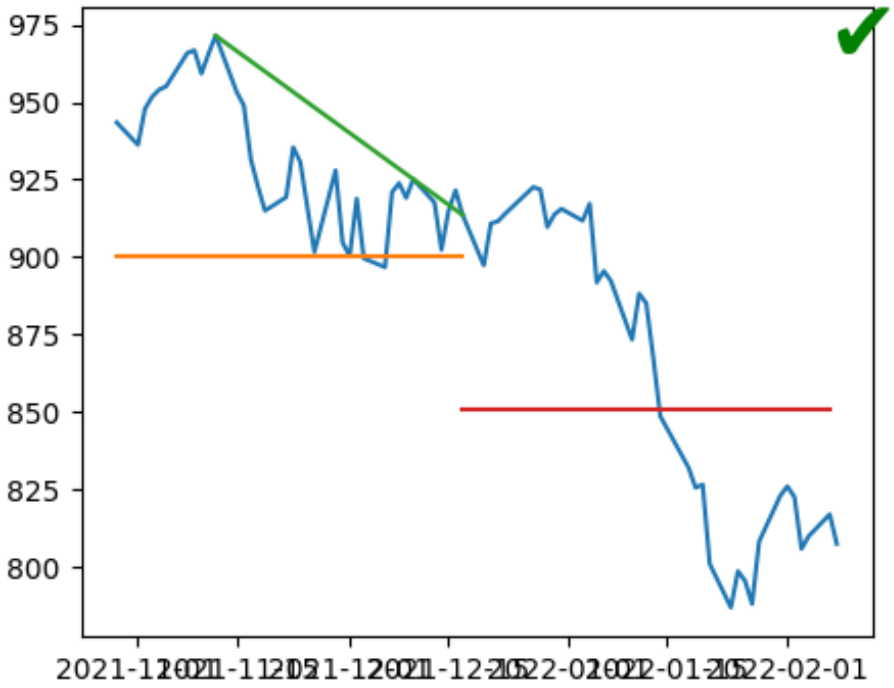
BBVA descending_triangle 2007-08-07 - 2007-09-18 - NN



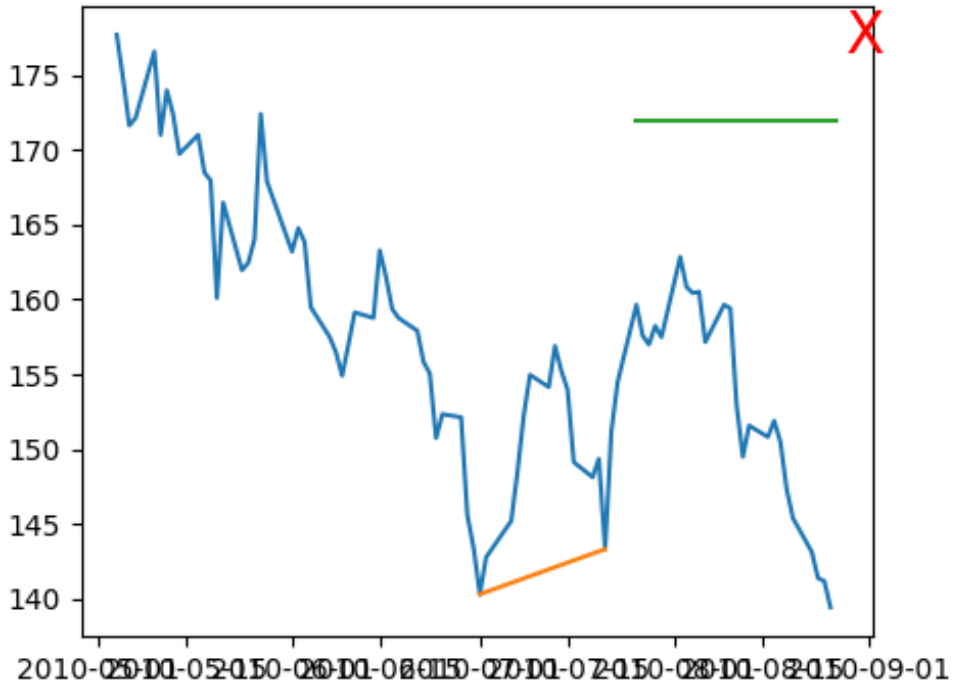
BBVA descending_triangle 2015-08-06 - 2015-10-08 - NN



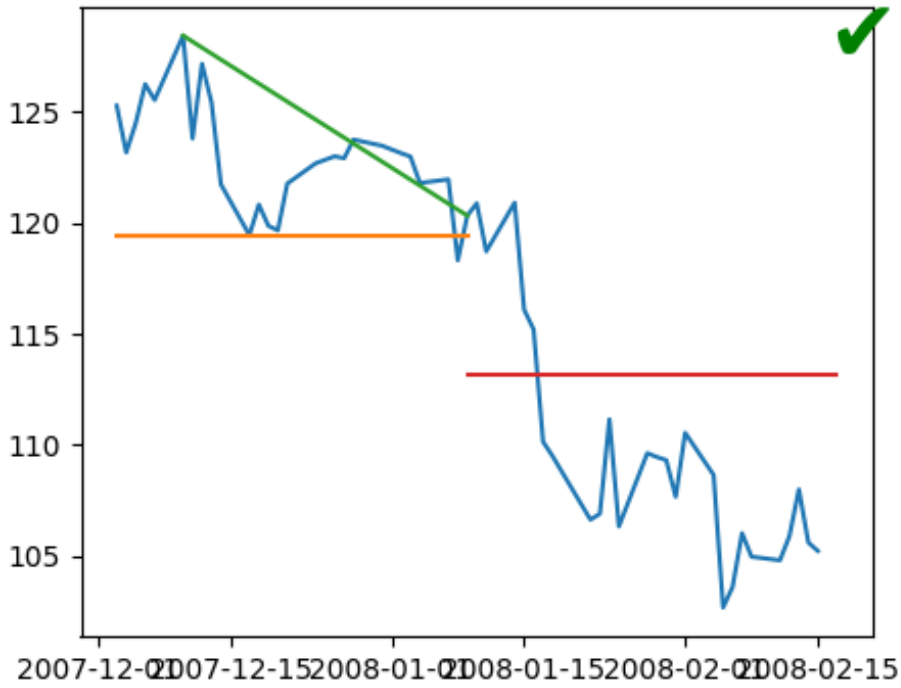
BLK descending_triangle 2021-10-29 - 2021-12-17 - NN



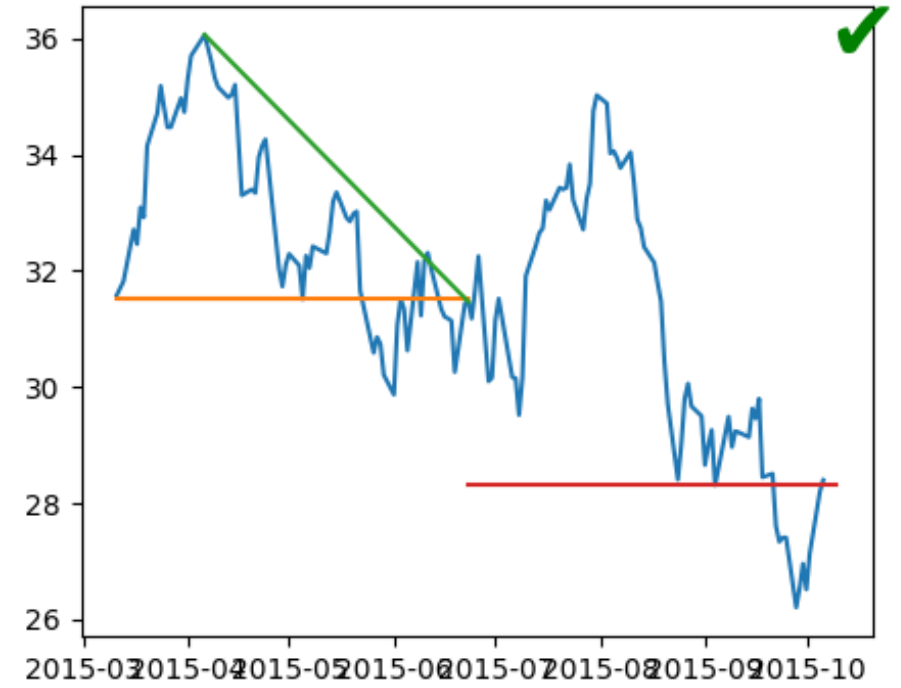
BLK double_bottom 2010-05-04 - 2010-08-04 - NN

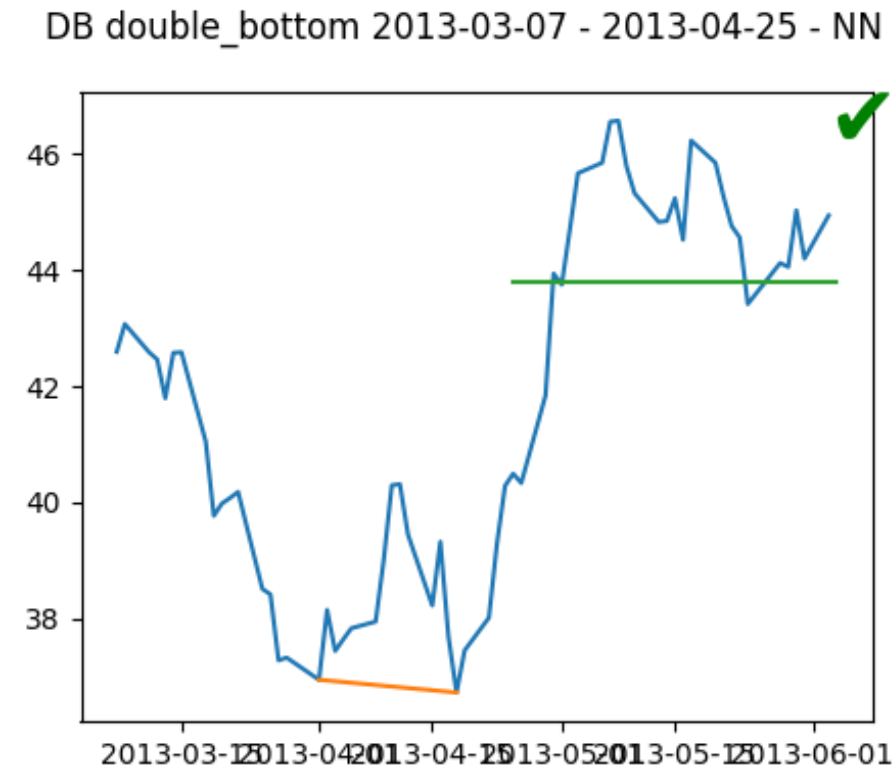


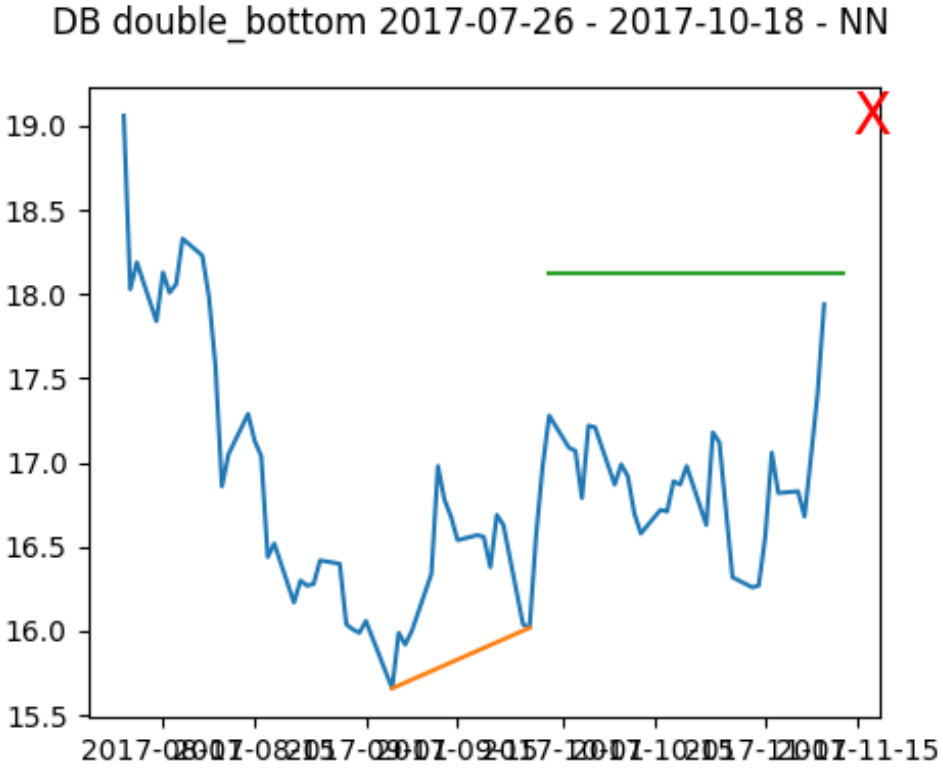
DB descending_triangle 2007-12-03 - 2008-01-09 - NN



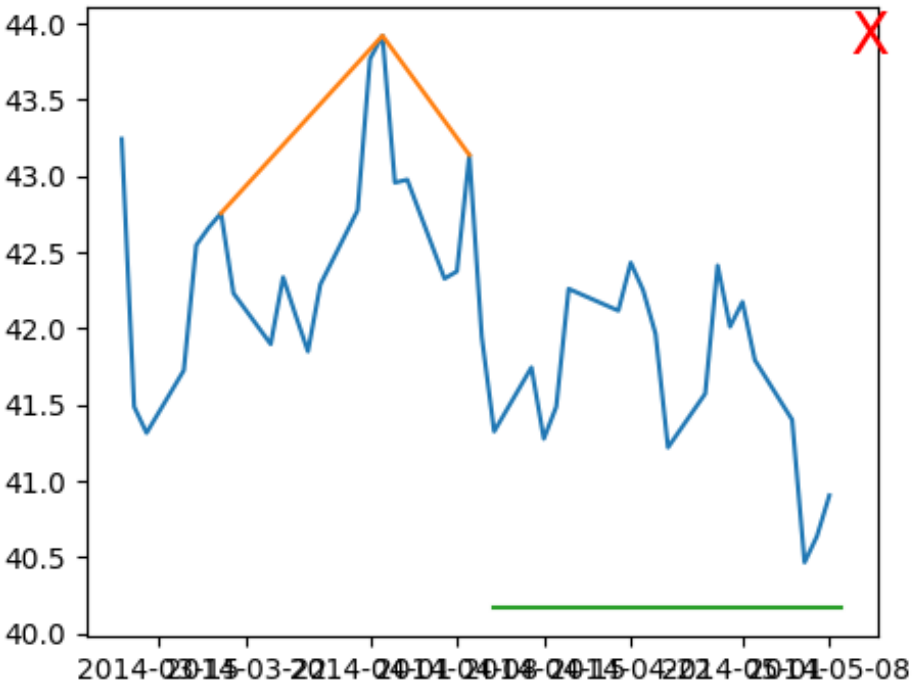
DB descending_triangle 2015-03-11 - 2015-06-23 - NN



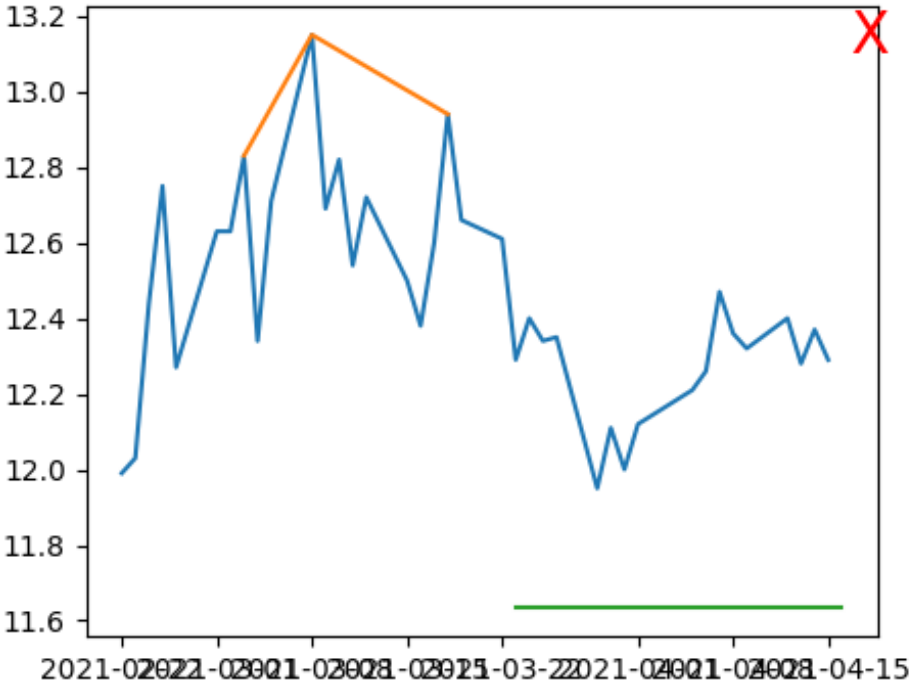

DB double_bottom 2013-03-07 - 2013-04-25 - NN
46
44
42
40
38
2013-03-15
2013-04-01
2013-04-15
2013-05-01
2013-05-15
2013-06-01


DB double_bottom 2017-07-26 - 2017-10-18 - NN
19.0
18.5
18.0
17.5
17.0
16.5
16.0
15.5

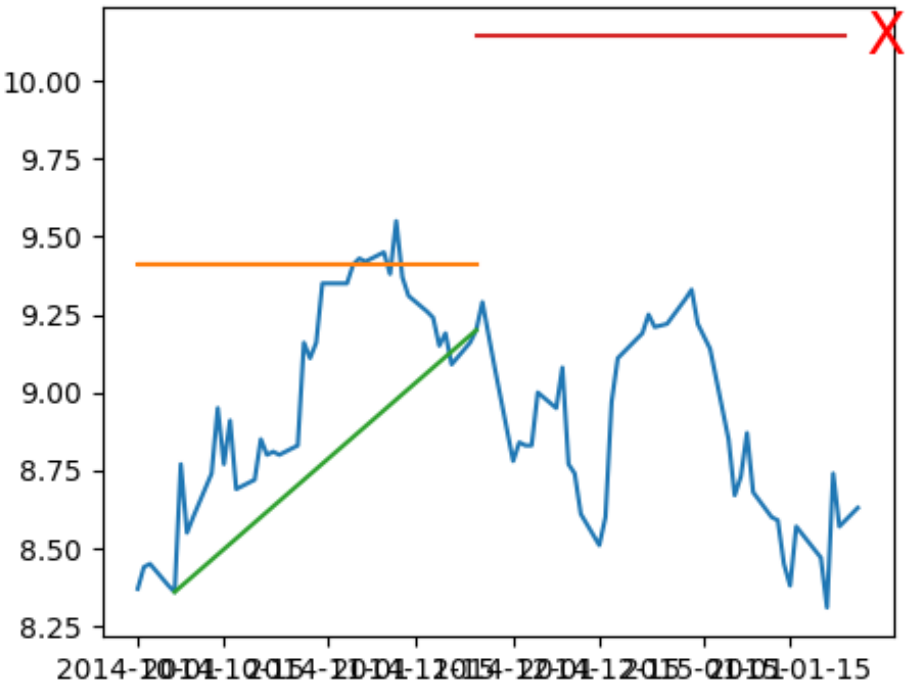
DB head_and_shoulders 2014-03-12 - 2014-04-23 - NN



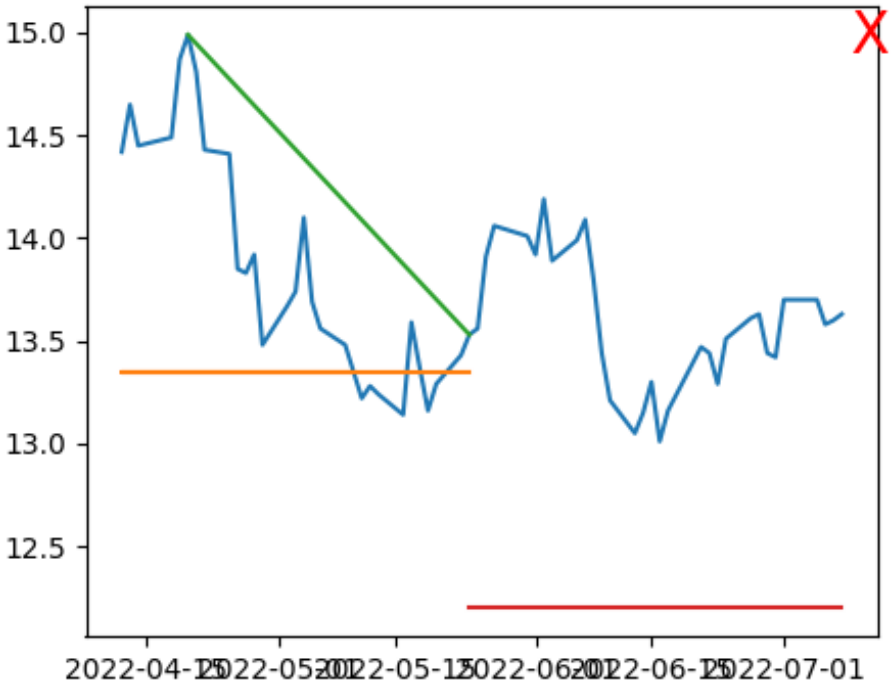
DB head_and_shoulders 2021-02-22 - 2021-04-26 - NN



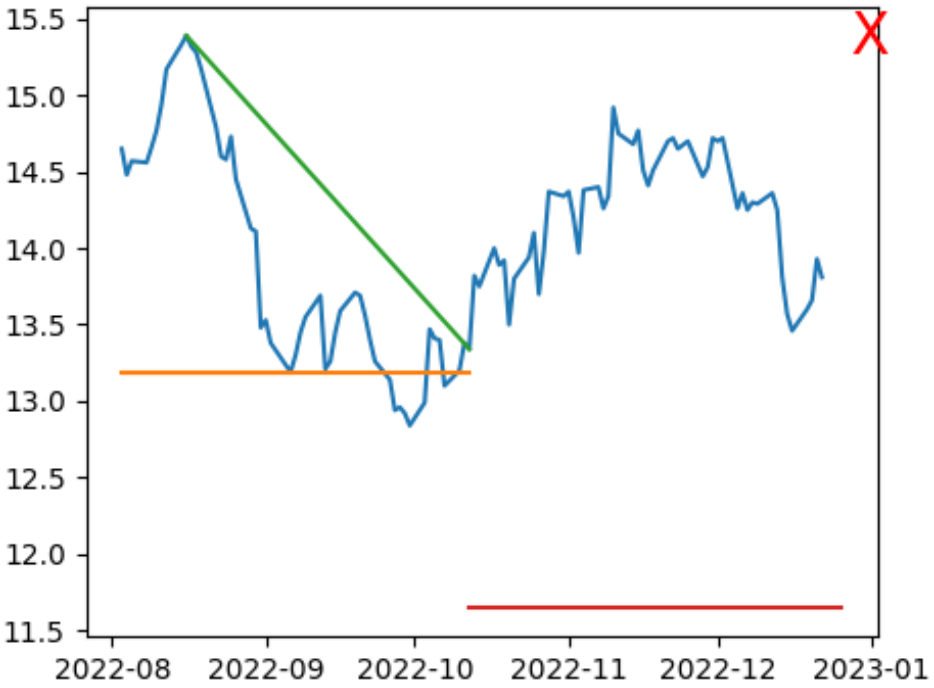
FCF ascending_triangle 2014-10-01 - 2014-11-25 - NN



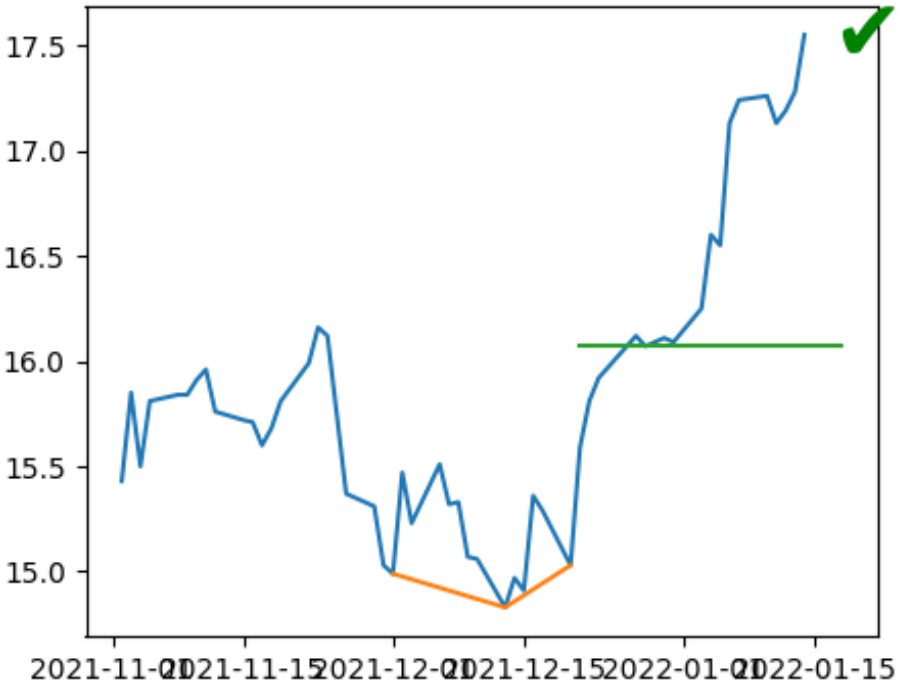
FCF descending_triangle 2022-04-12 - 2022-05-24 - NN



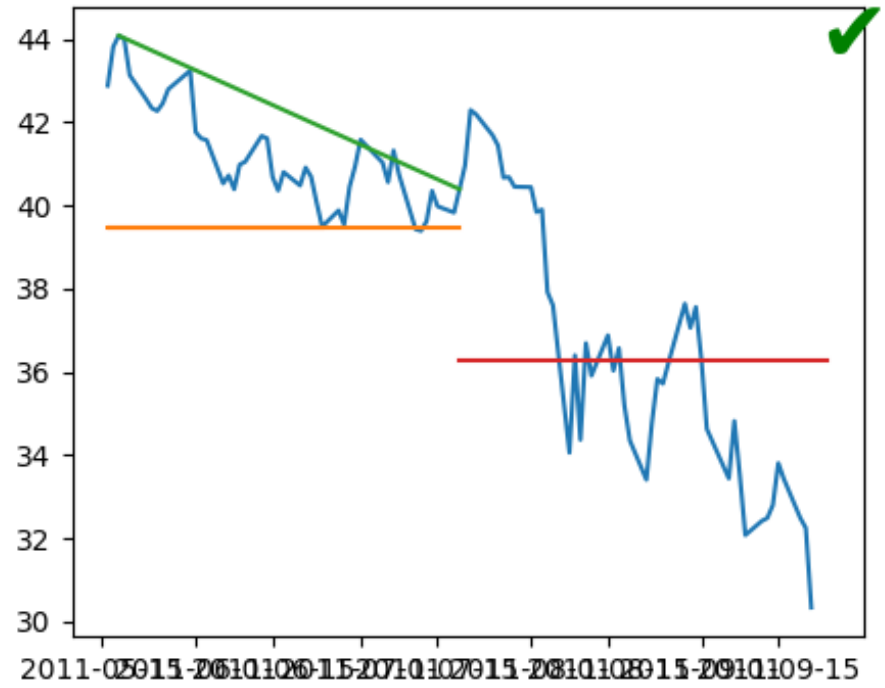
FCF descending_triangle 2022-08-03 - 2022-10-12 - NN



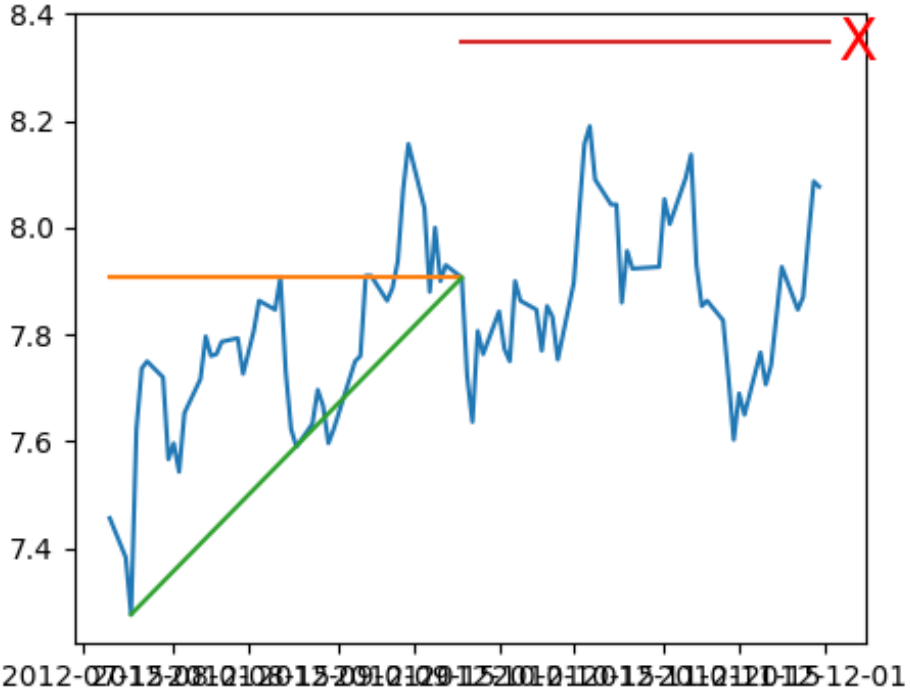
FCF inv_head_and_shoulders 2021-11-02 - 2021-12-29 - NN



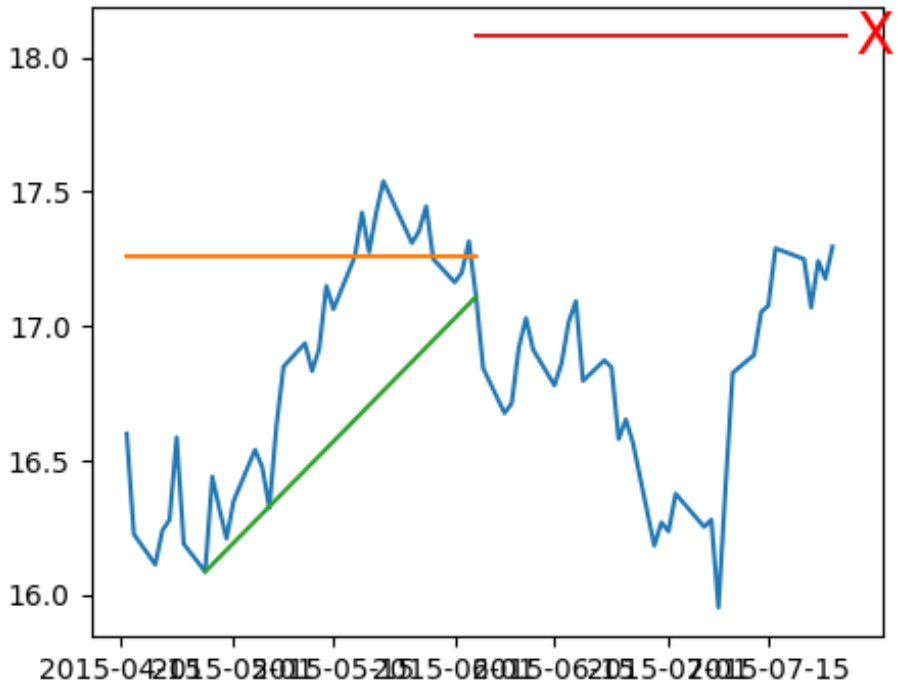
JPM descending_triangle 2011-05-16 - 2011-07-19 - NN



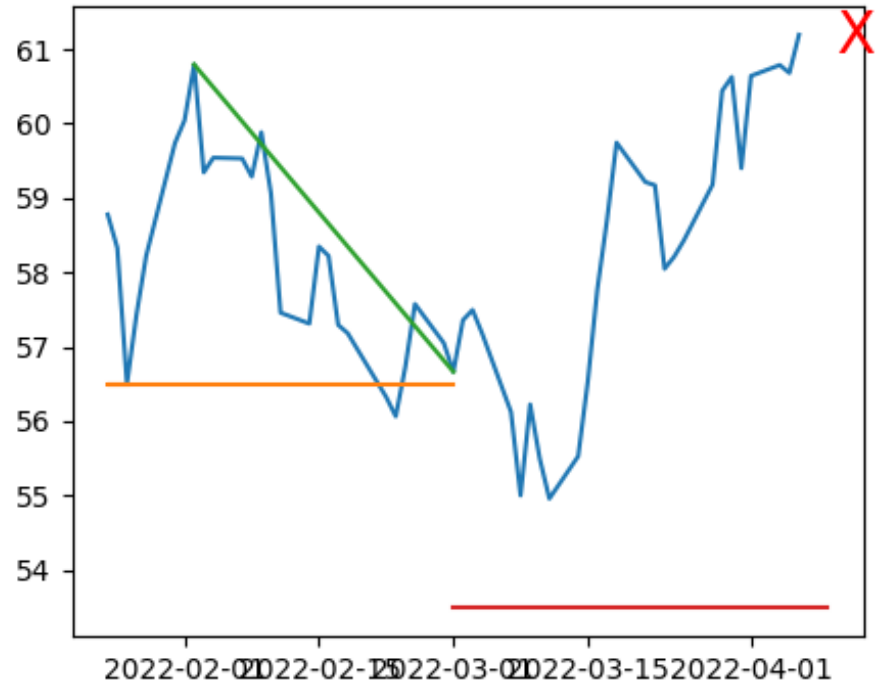
NDAQ ascending_triangle 2012-07-20 - 2012-09-24 - NN



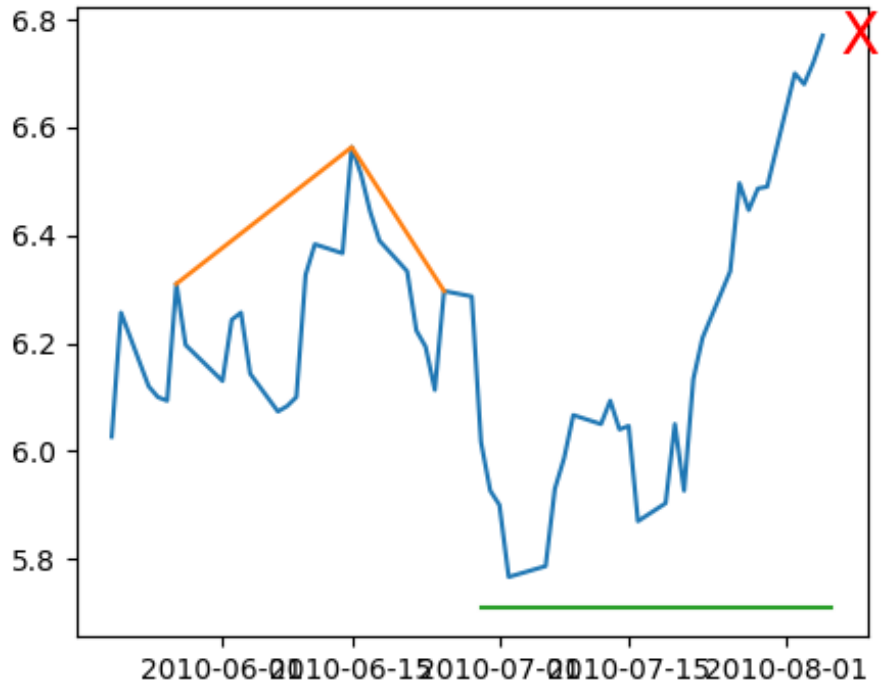
NDAQ ascending_triangle 2015-04-16 - 2015-06-04 - NN



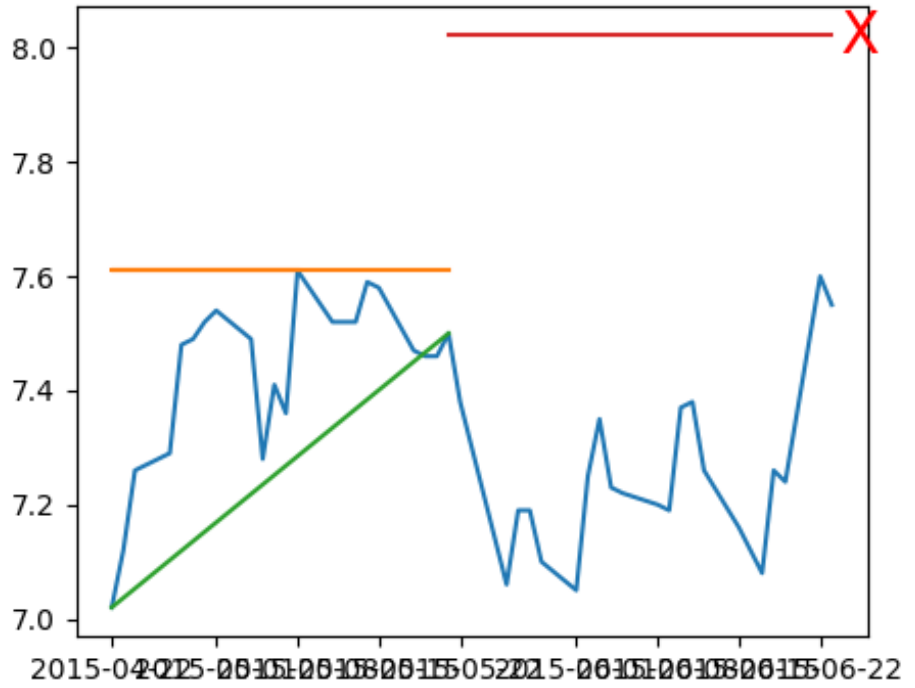
NDAQ descending_triangle 2022-01-24 - 2022-03-01 - NN



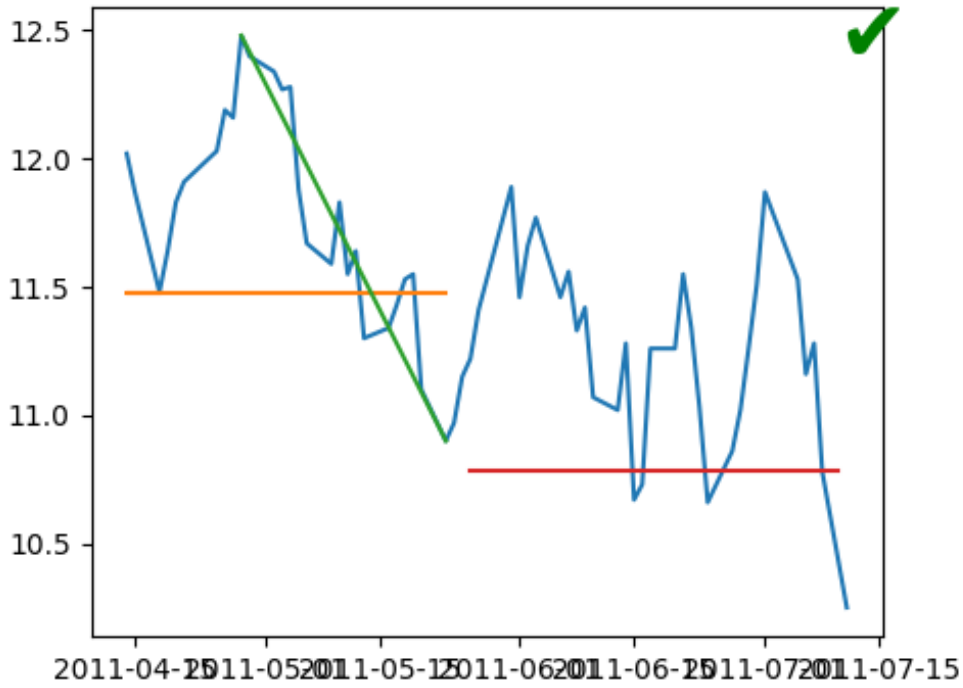
NDAQ head_and_shoulders 2010-05-20 - 2010-07-09 - NN



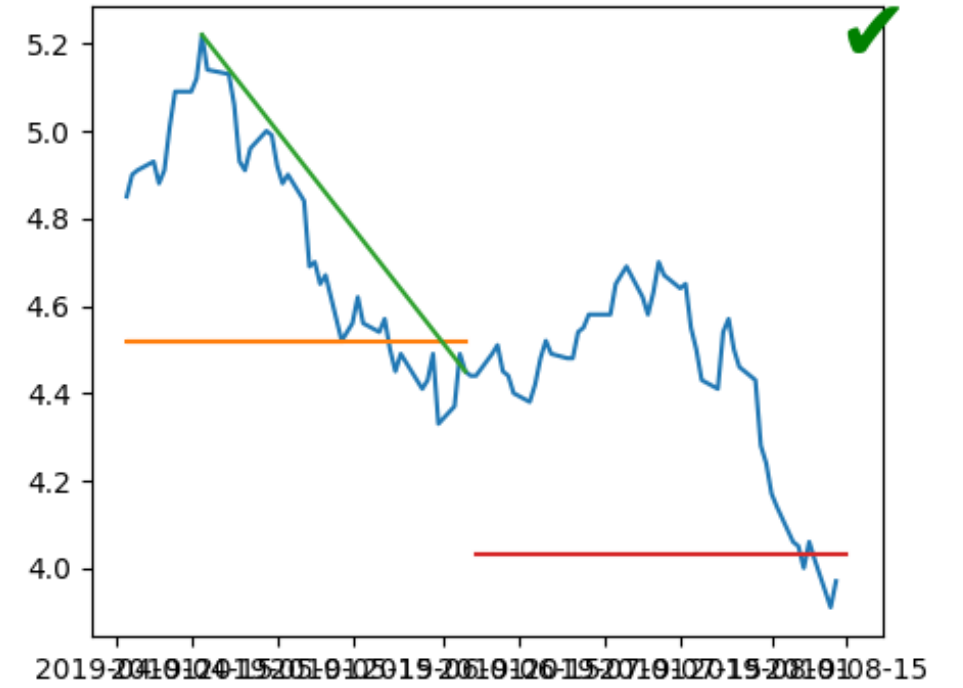
SAN ascending_triangle 2015-04-22 - 2015-05-21 - NN

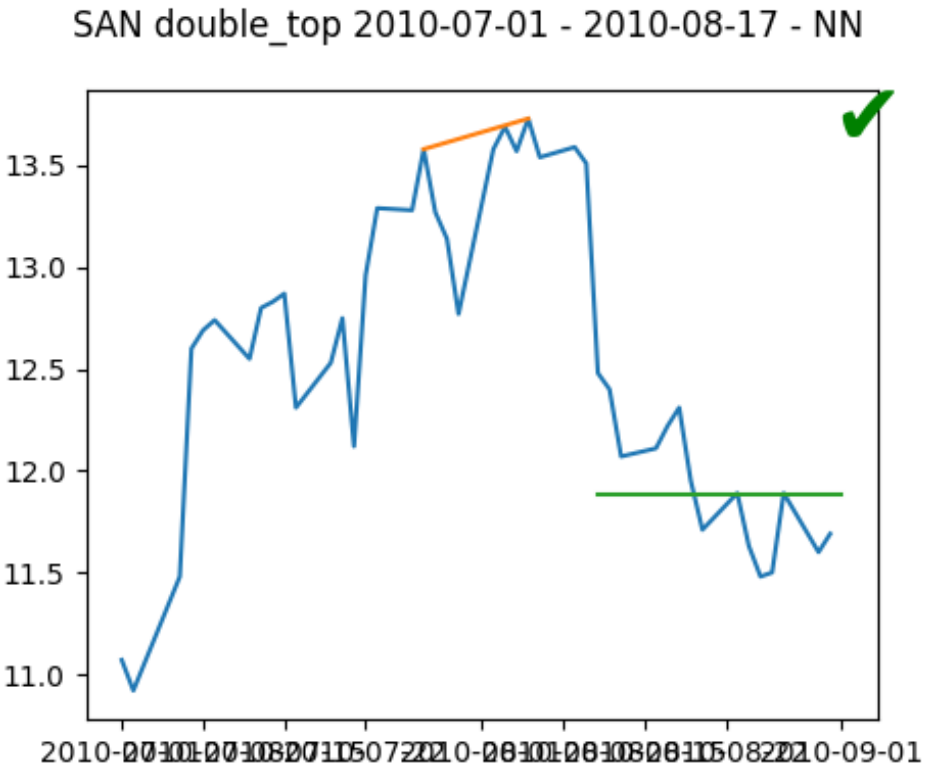


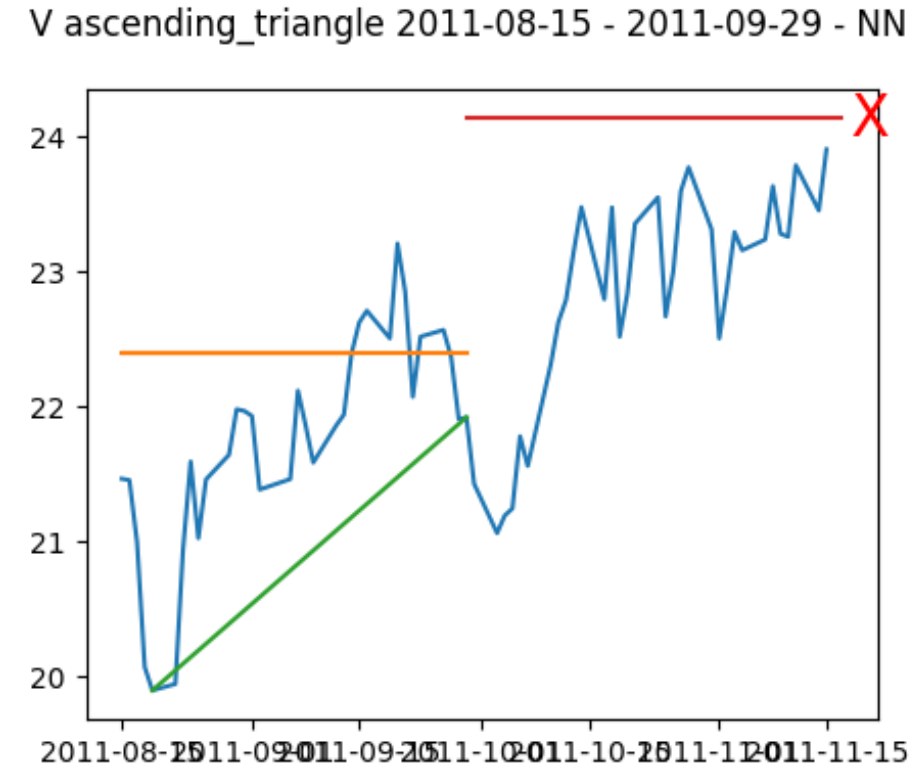
SAN descending_triangle 2011-04-14 - 2011-05-26 - NN



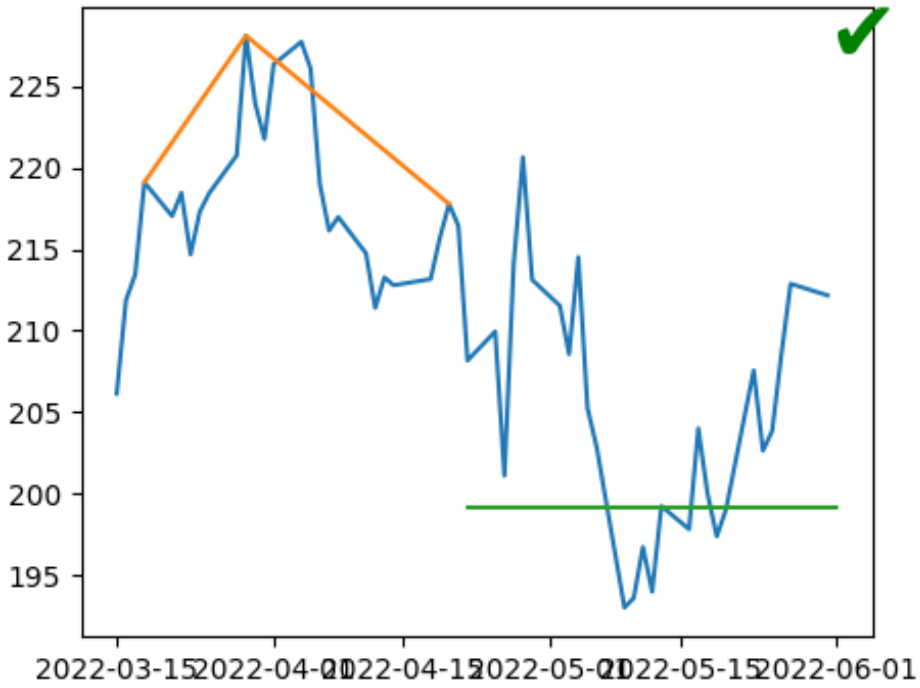
SAN descending_triangle 2019-04-03 - 2019-06-07 - NN



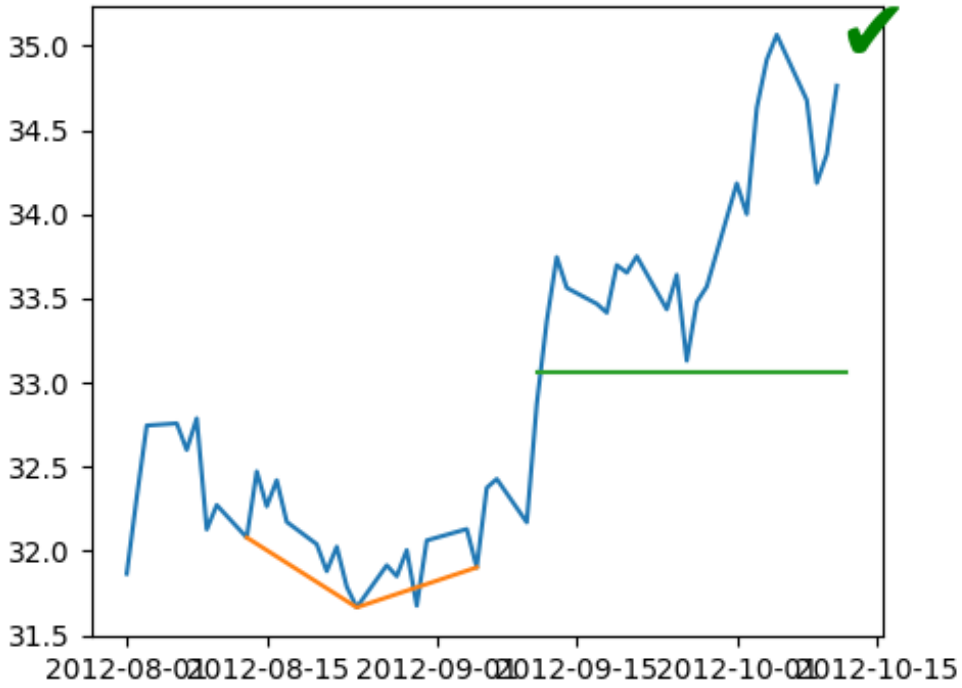

SAN double_top 2010-07-01 - 2010-08-17 - NN
13.5
13.0
12.5
12.0
11.5
11.0
2010-09-01

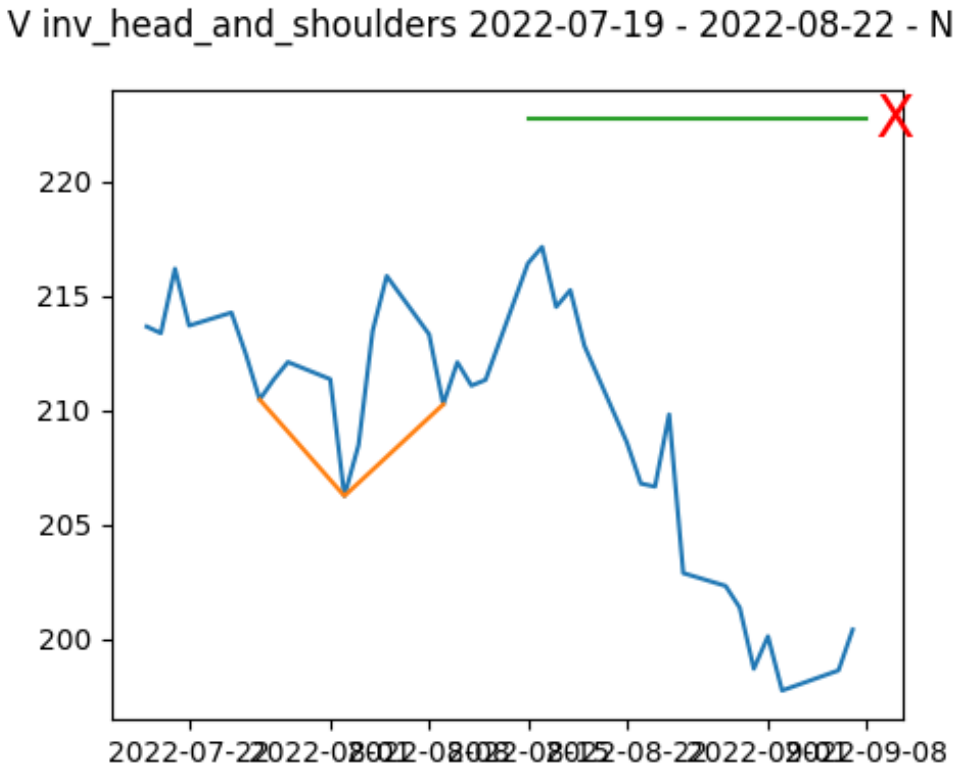

V ascending_triangle 2011-08-15 - 2011-09-29 - NN
24
23
22
21
20
2011-11-15

V head_and_shoulders 2022-03-15 - 2022-04-26 - NN

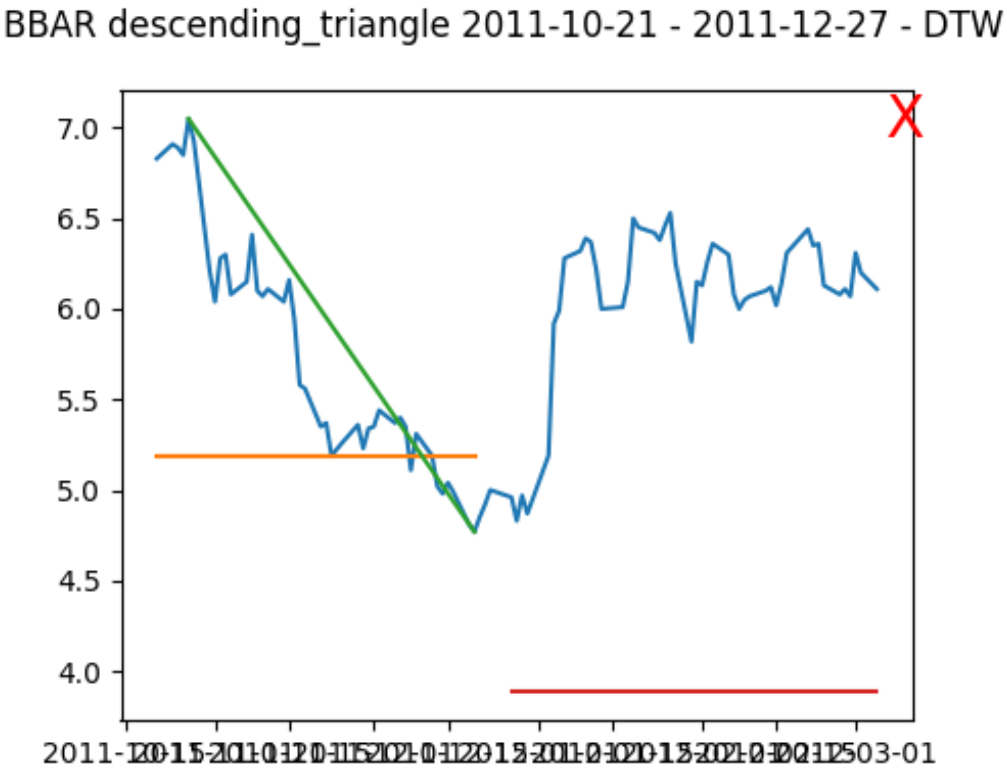


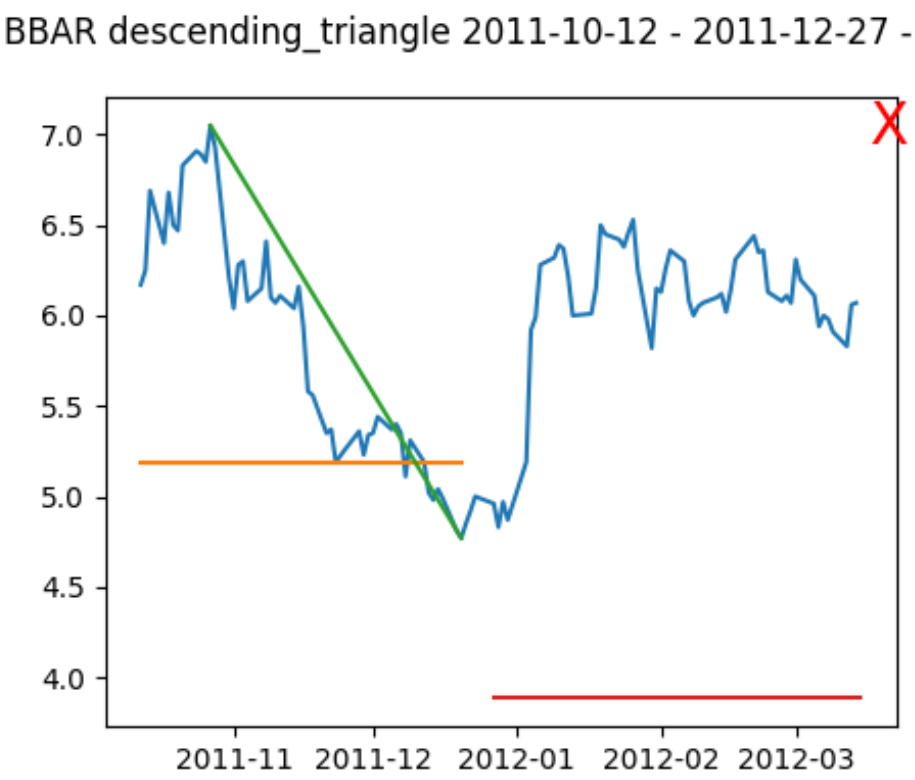
V inv_head_and_shoulders 2012-08-01 - 2012-09-12 - NN

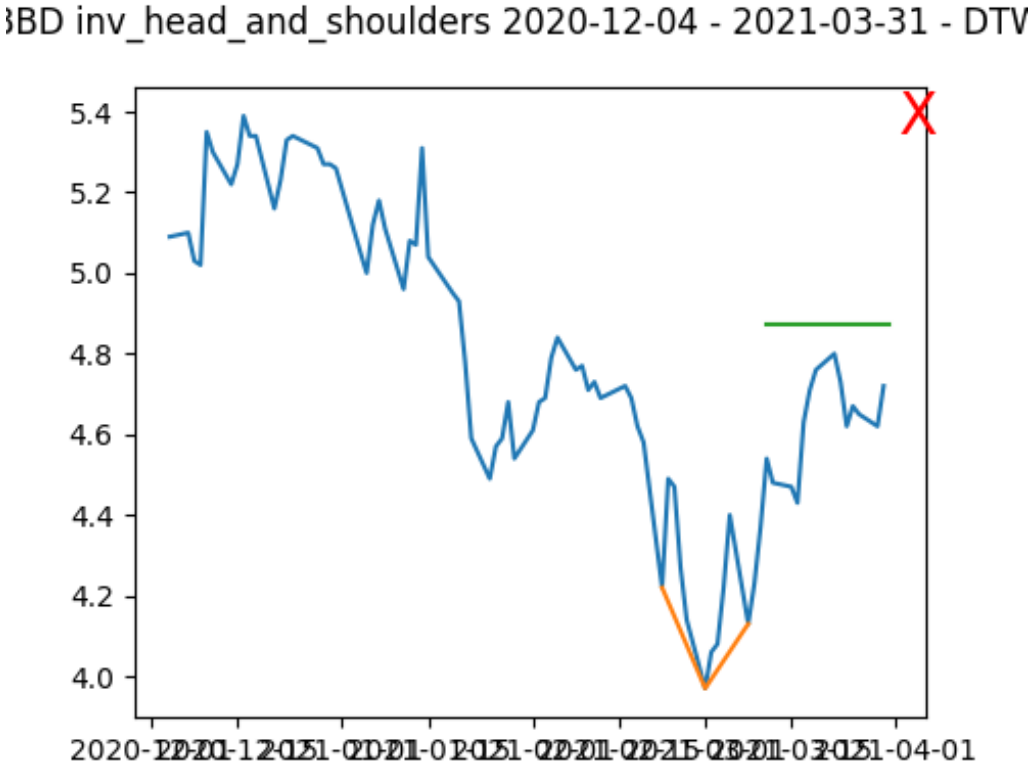


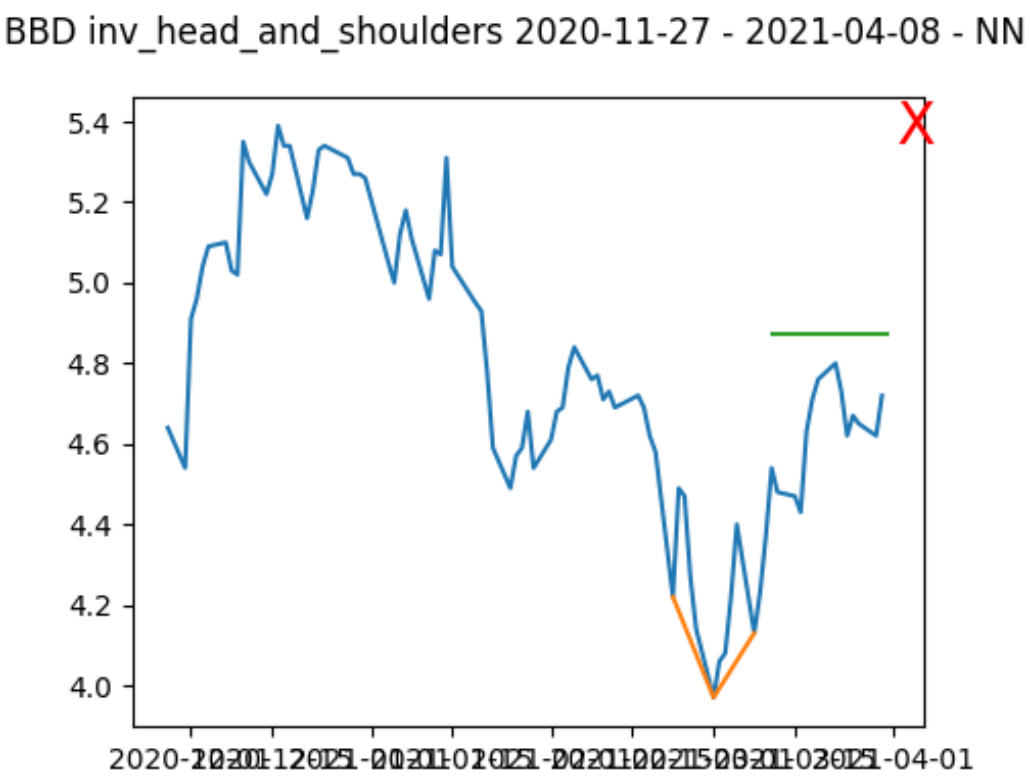


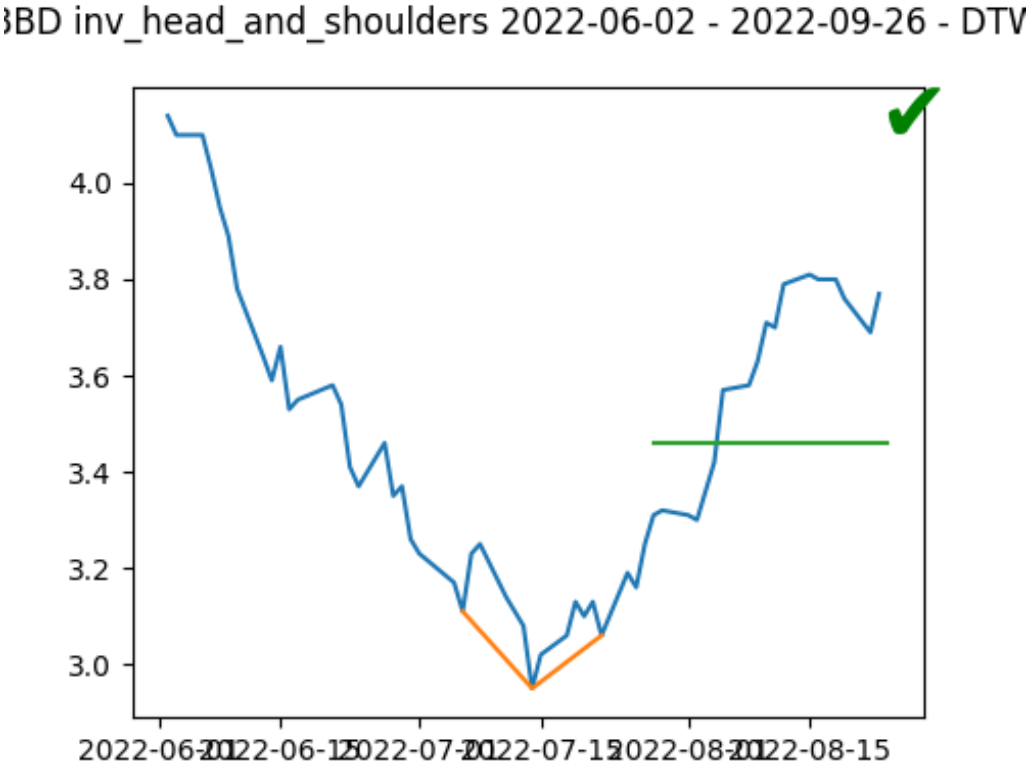
A.8. Patrones solapados sector financiero



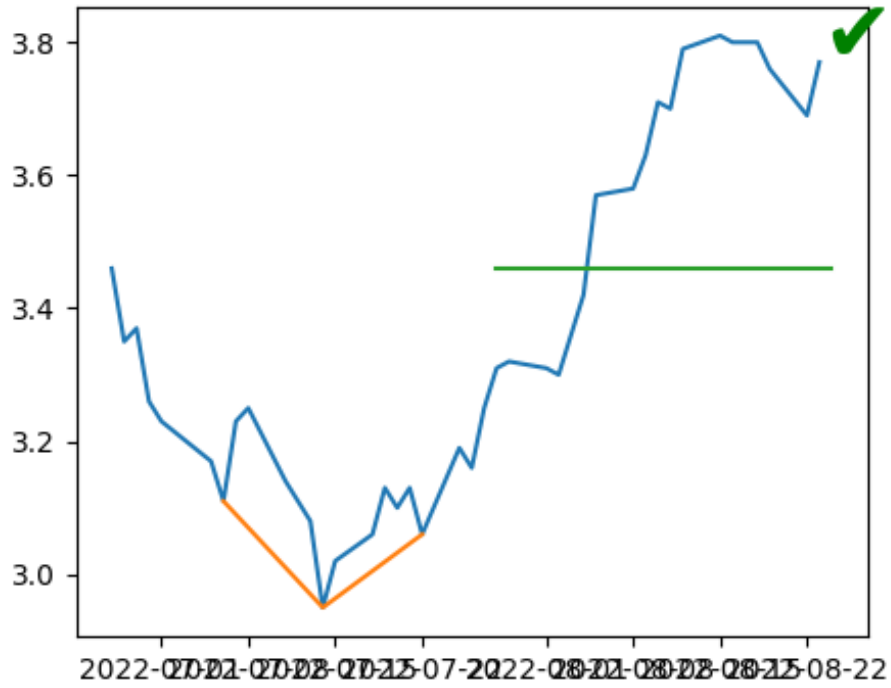

BBAR descending_triangle 2011-10-12 - 2011-12-27 - NN
7.0
6.5
6.0
5.5
5.0
4.5
4.0
2011-11
2011-12
2012-01
2012-02
2012-03


BD inv_head_and_shoulders 2020-12-04 - 2021-03-31 - DTV
5.4
5.2
5.0
4.8
4.6
4.4
4.2
4.0

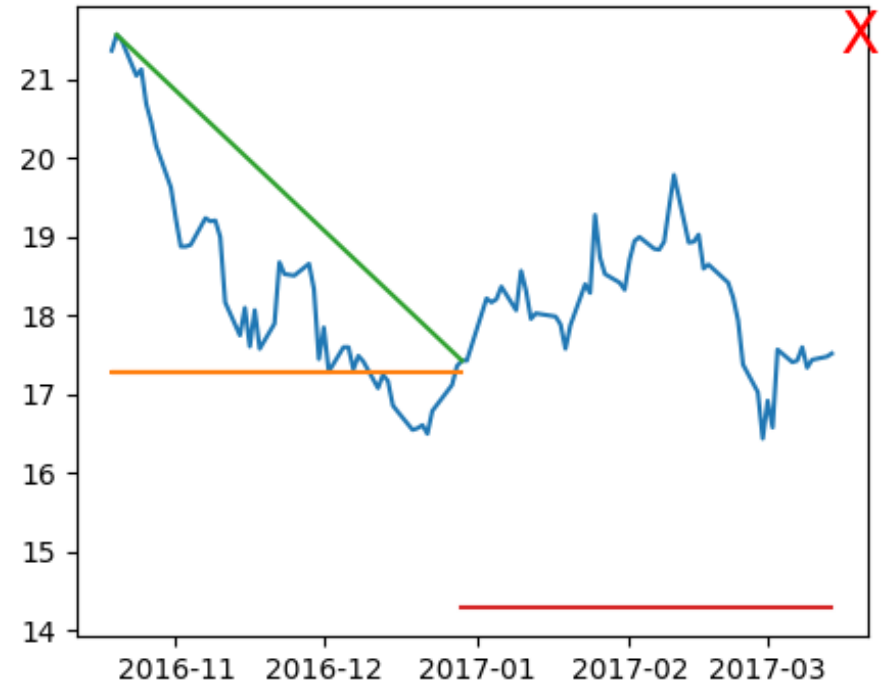

BBD inv_head_and_shoulders 2020-11-27 - 2021-04-08 - NN
5.4
5.2
5.0
4.8
4.6
4.4
4.2
4.0
X


BBD inv_head_and_shoulders 2022-06-02 - 2022-09-26 - DTV
4.0
3.8
3.6
3.4
3.2
3.0
2022-06-01
2022-06-15
2022-07-01
2022-07-15
2022-08-01
2022-08-15

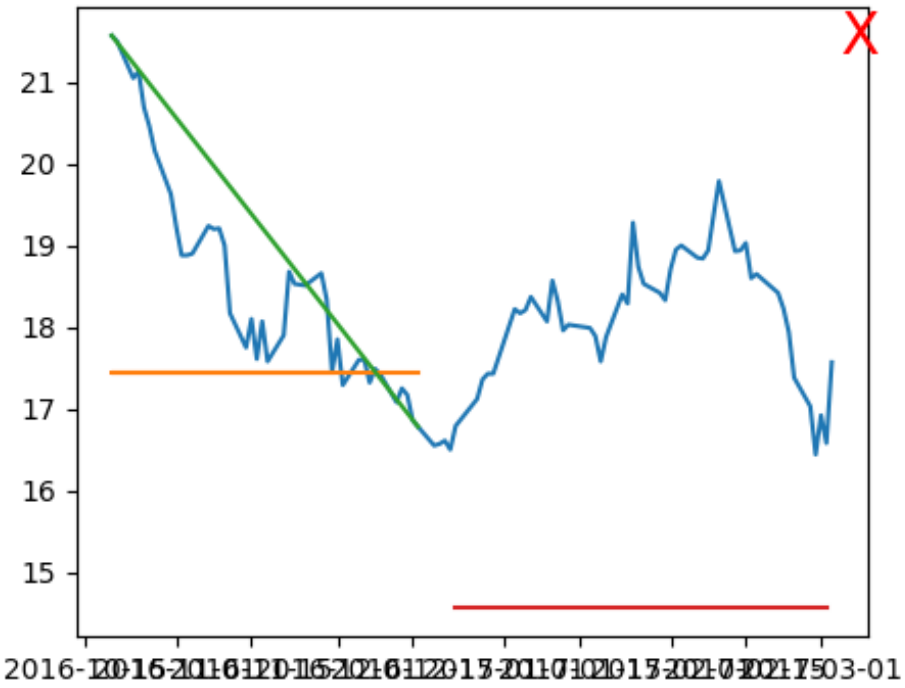
BBD inv_head_and_shoulders 2022-06-27 - 2022-08-08 - NN



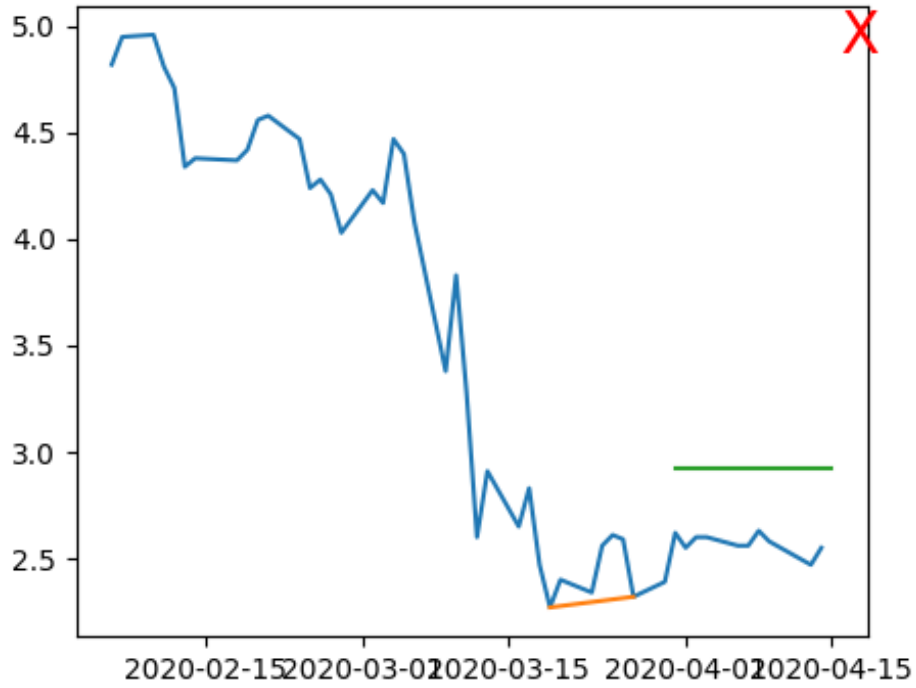
BBAR descending_triangle 2016-10-19 - 2016-12-29 - DTW

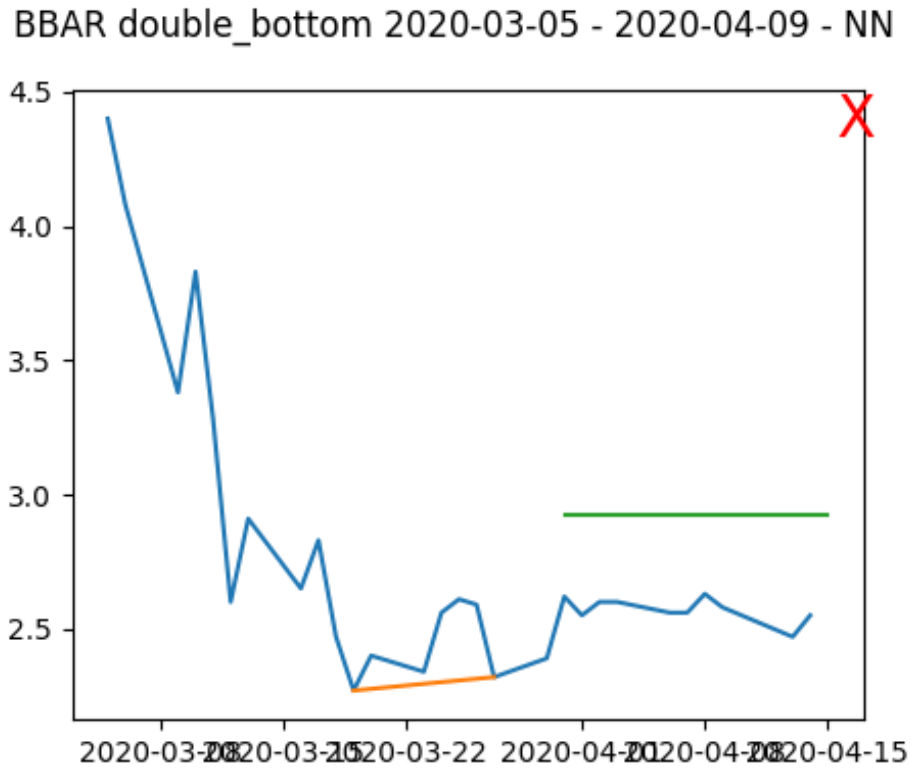


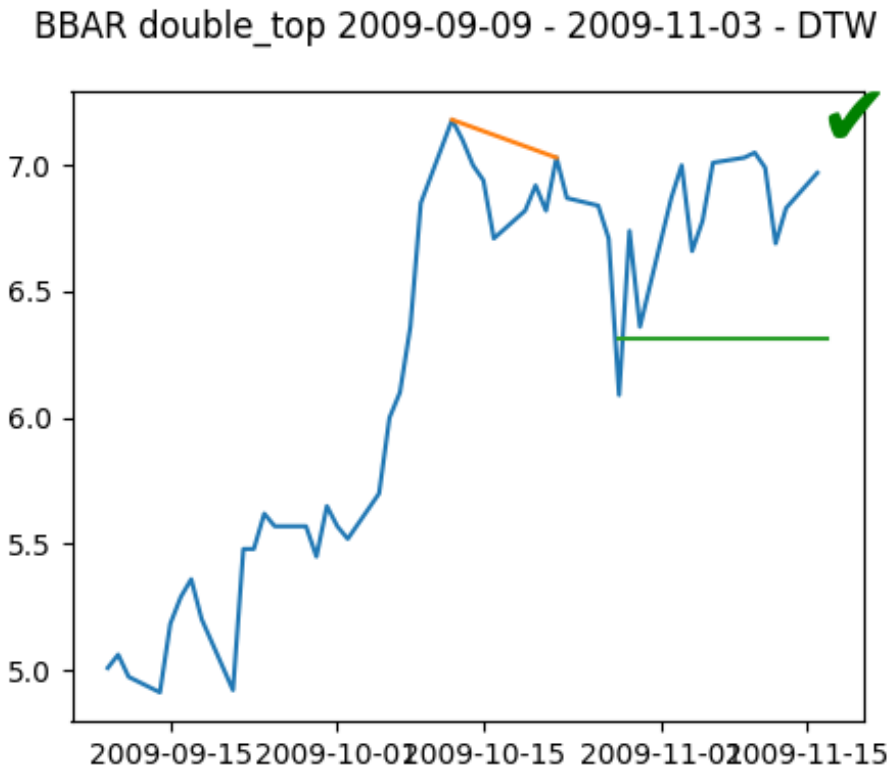
BBAR descending_triangle 2016-10-20 - 2016-12-23 - NN

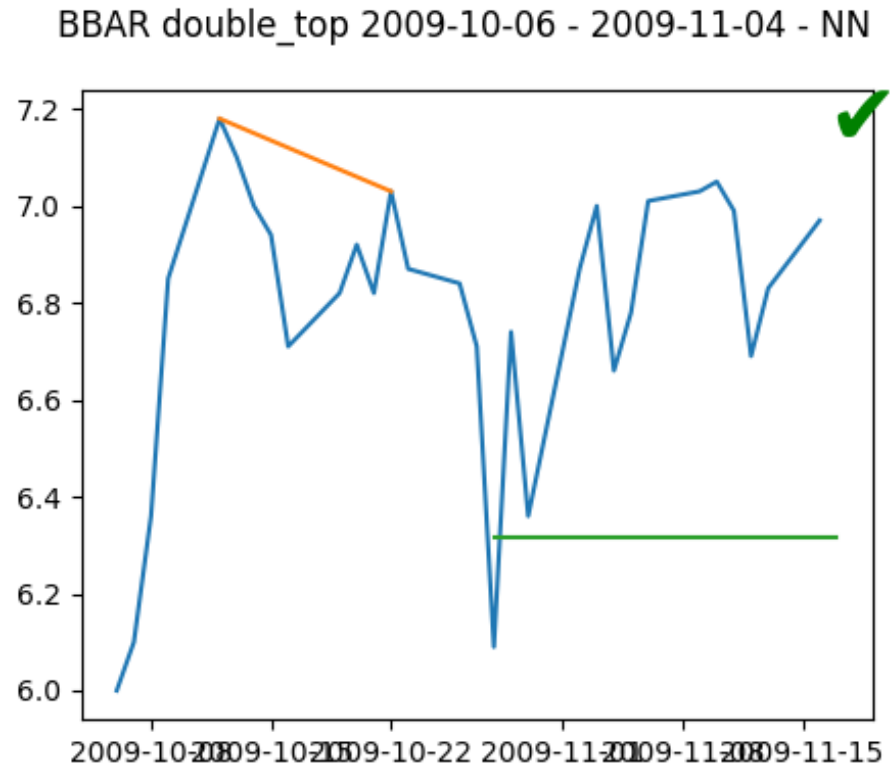


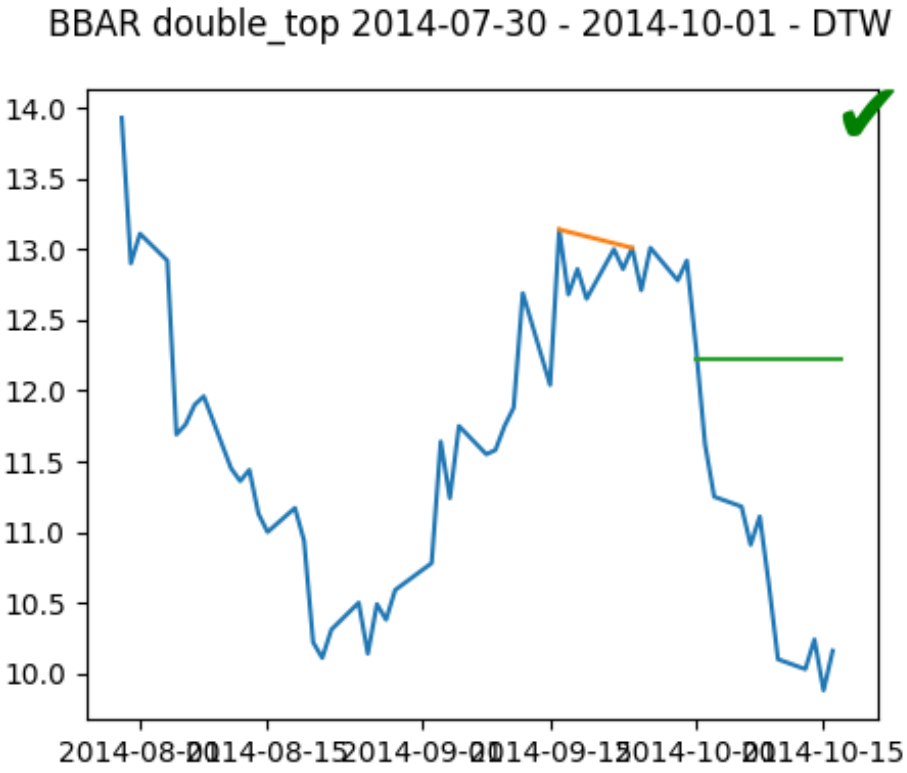
BBAR double_bottom 2020-02-06 - 2020-04-02 - DTW

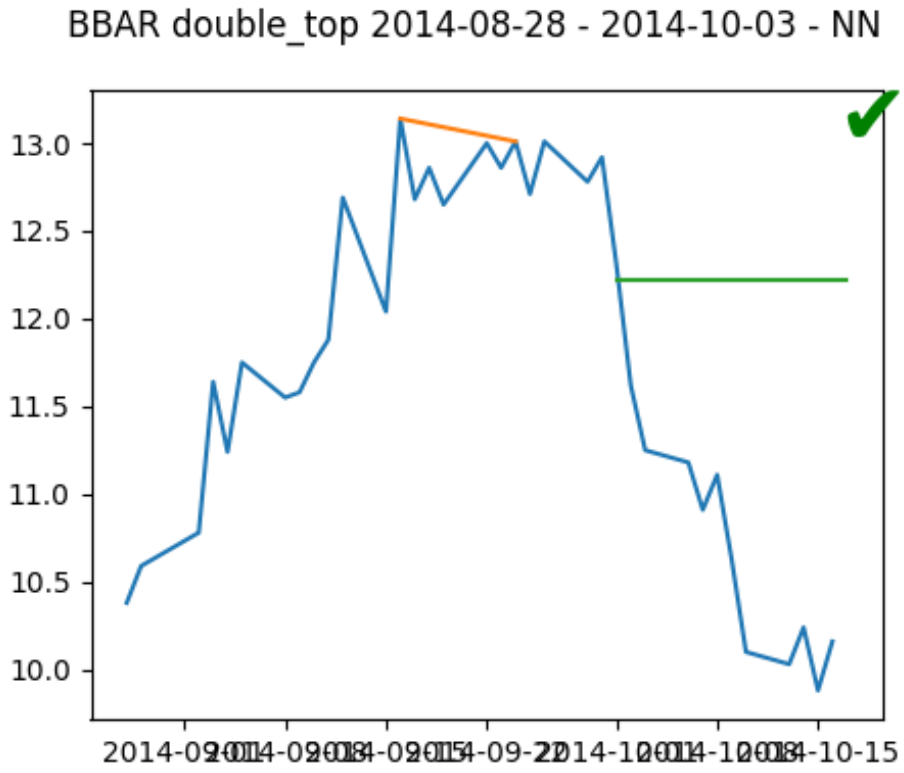


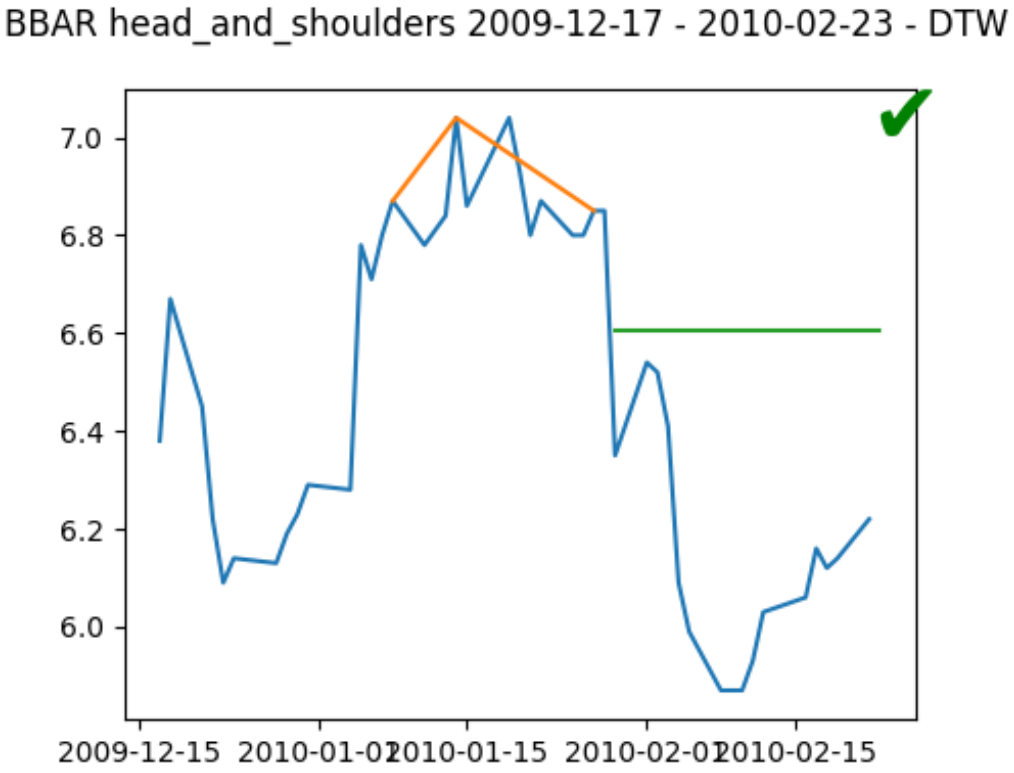

BBAR double_bottom 2020-03-05 - 2020-04-09 - NN
4.5
4.0
3.5
3.0
2.5
2020-03-22
2020-04-15


BBAR double_top 2009-09-09 - 2009-11-03 - DTW
7.0
6.5
6.0
5.5
5.0
2009-09-15
2009-10-15
2009-11-15

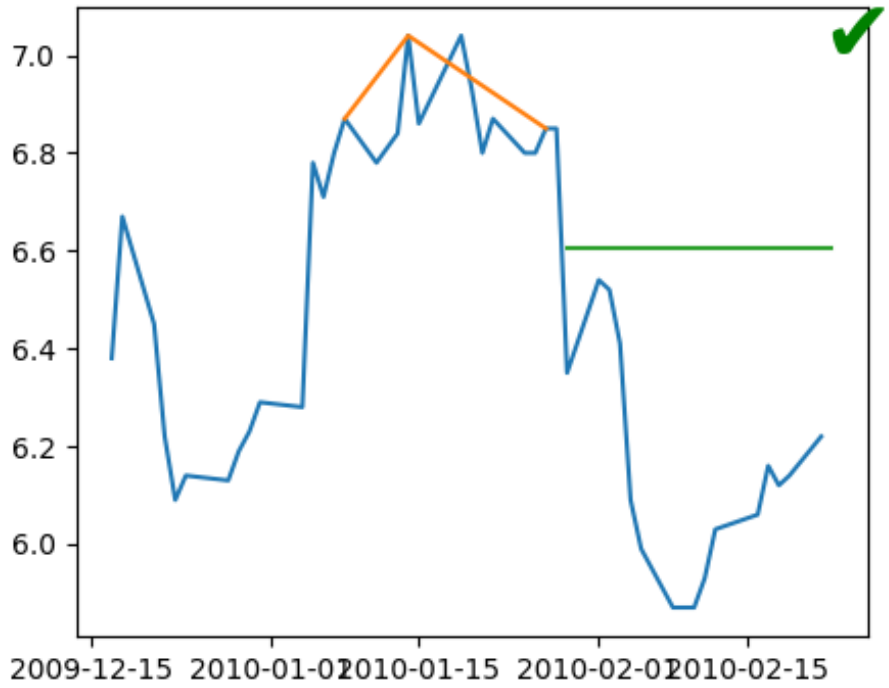

BBAR double_top 2009-10-06 - 2009-11-04 - NN
7.2
7.0
6.8
6.6
6.4
6.2
6.0
2009-10-22
2009-11-15


BBAR double_top 2014-07-30 - 2014-10-01 - DTW
14.0
13.5
13.0
12.5
12.0
11.5
11.0
10.5
10.0
2014-08-15
2014-09-15
2014-10-15

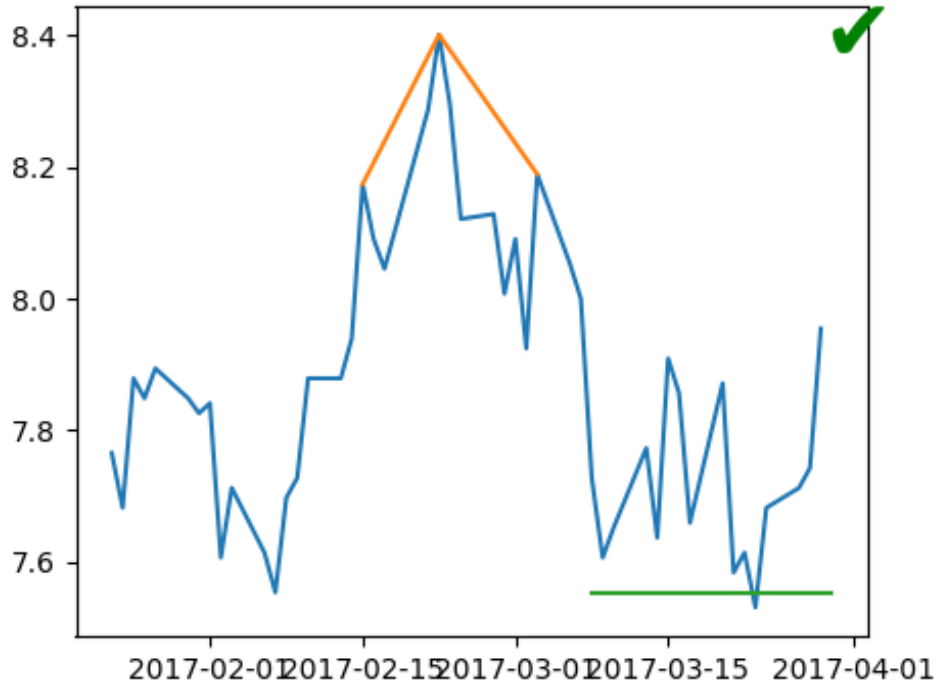

BBAR double_top 2014-08-28 - 2014-10-03 - NN
13.0
12.5
12.0
11.5
11.0
10.5
10.0
2014-10-15


BBAR head_and_shoulders 2009-12-17 - 2010-02-23 - DTW
7.0
6.8
6.6
6.4
6.2
6.0
2009-12-15
2010-01-15
2010-02-15

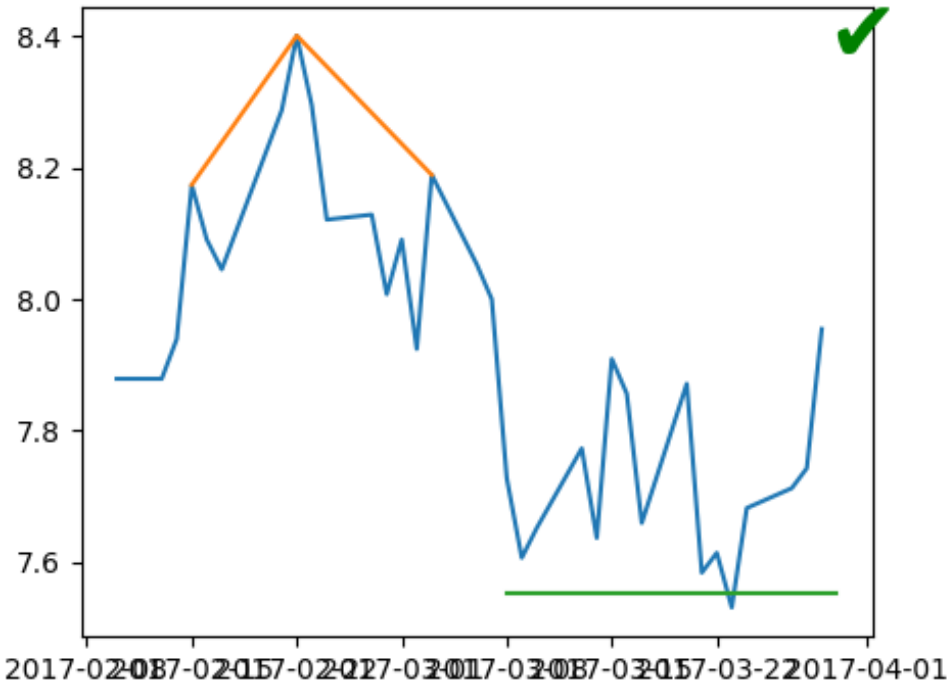
BBAR head_and_shoulders 2009-12-17 - 2010-02-08 - NN



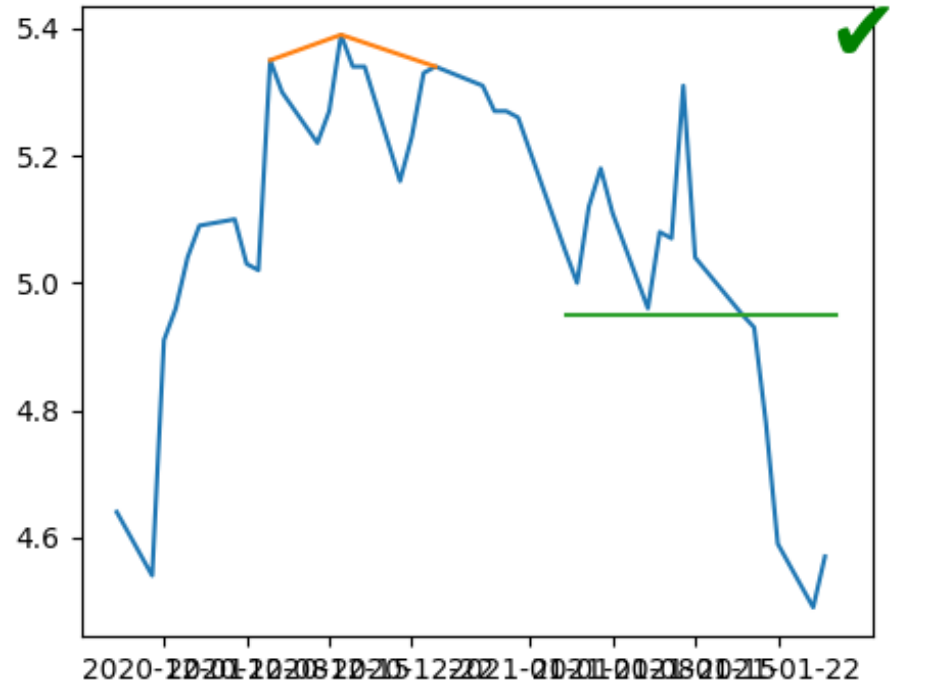
BBD head_and_shoulders 2017-01-23 - 2017-03-20 - DTW



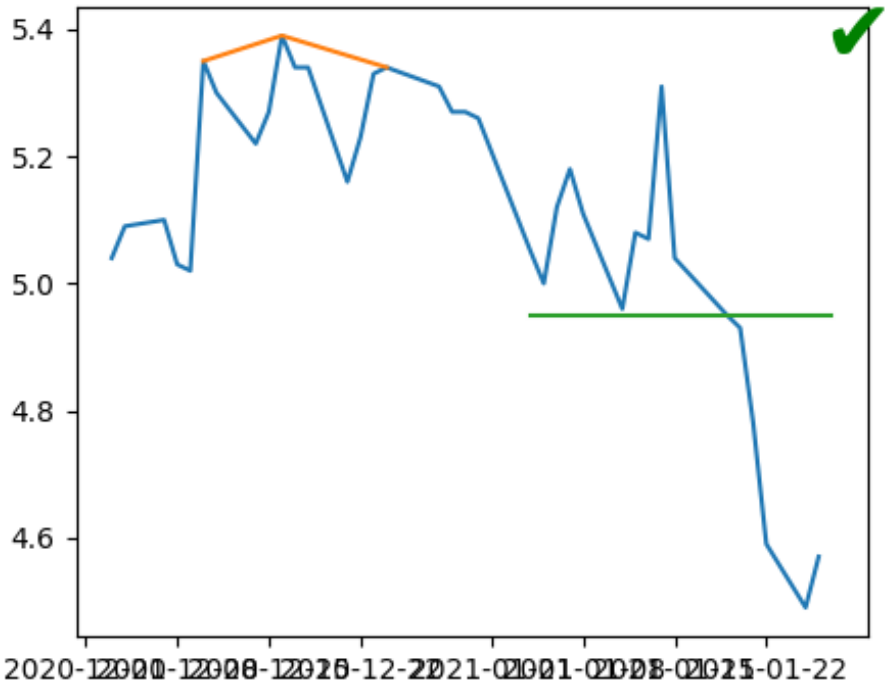
BBD head_and_shoulders 2017-02-10 - 2017-04-24 - NN



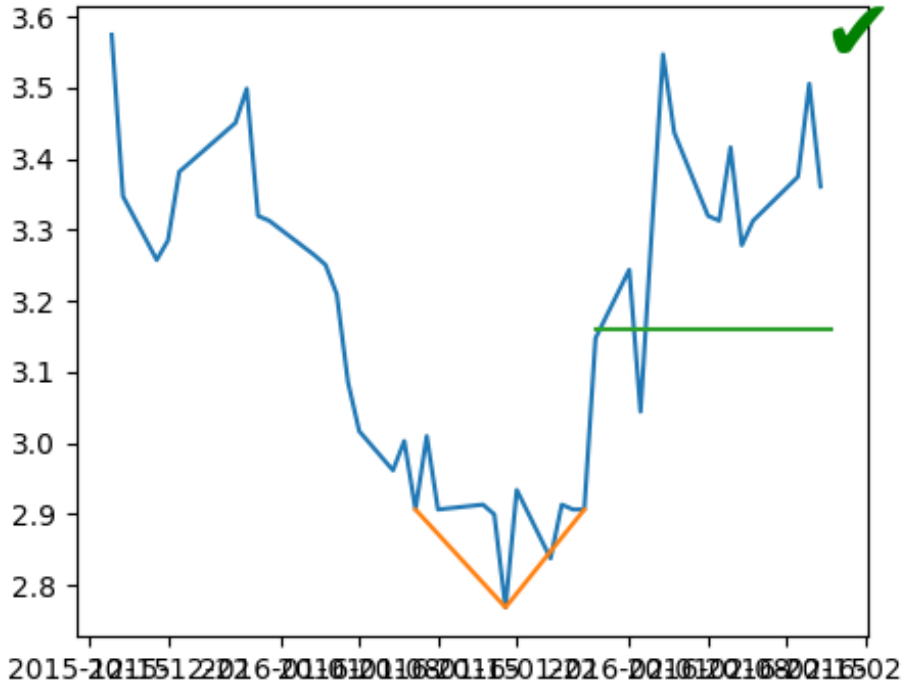
BBD head_and_shoulders 2020-11-27 - 2021-01-26 - DTW

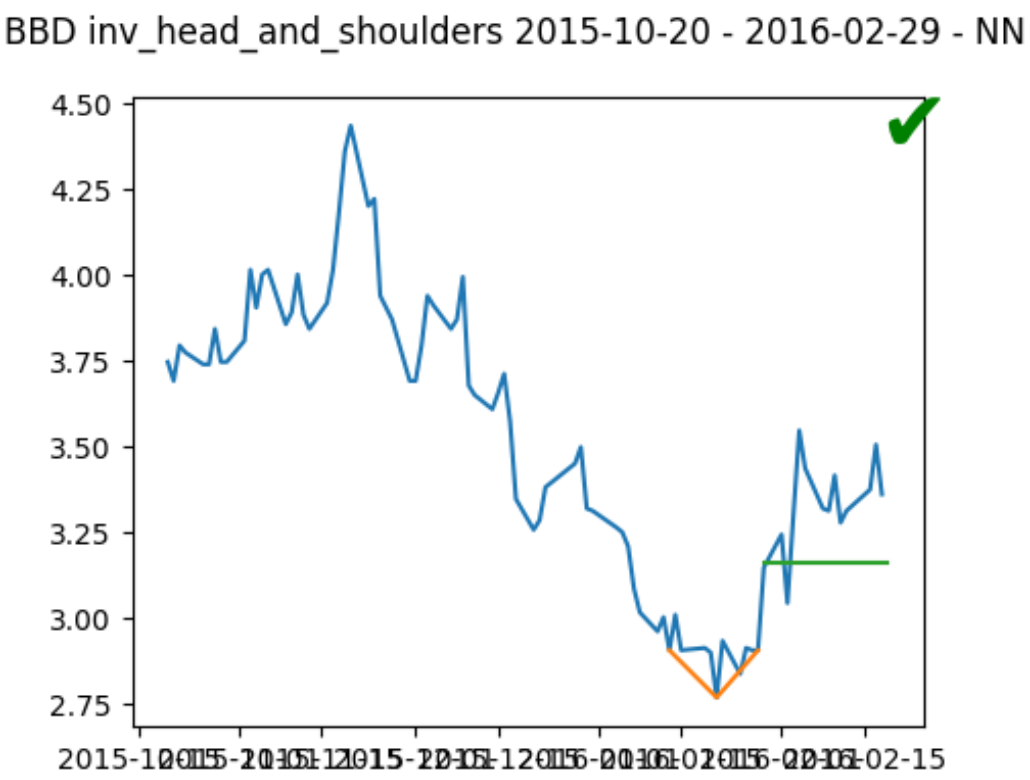


BBD head_and_shoulders 2020-12-03 - 2021-01-11 - NN



BBD inv_head_and_shoulders 2015-12-17 - 2016-02-16 - DTV




BBD inv_head_and_shoulders 2015-10-20 - 2016-02-29 - NN
4.50
4.25
4.00
3.75
3.50
3.25
3.00
2.75

Bibliografía

Steven B Achelis. 2001. Technical Analysis from A to Z.

Aitor Alonso Melián. 2023. Estrategias de inversión en bolsa basadas en la detección de patrones mediante análisis técnico. `http://riull.ull.es/xmlui/handle/915/33353` Repositorio institucional de la Universidad de La Laguna.

E. R. Dawson and J. M. Steeley. 2003. On the existence of visual technical patterns in the UK stock market. *Journal of Business Finance and Accounting* 30 (2003), 263–293.

J De Villiers. 1998. *The use of neural networks to predict share prices*. Master's thesis. University of Johannesburg (South Africa).

R.T. Farias, J.L. Silva, V.A. Sobreiro, and H. Kimura. 2017. A literature review of technical analysis on stock markets. *Quarterly Review of Economics and Finance* 66 (2017), 115–126.

G.C. Friesen, P.A. Weller, and L.M. Dunham. 2009. Price trends and patterns in technical analysis: A theoretical and empirical examination. *Journal of Banking and Finance* 33 (2009), 1089–1100.

Gabriel García Jaubert. 2022. Herramienta de inteligencia artificial para detectar patrones en bolsa. `http://riull.ull.es/xmlui/handle/915/29420` Repositorio institucional de la Universidad de La Laguna.

Ramón Jesús Ruiz Martínez and Antonio de la Torre Gallegos. 1999. El papel del análisis técnico en la filosofía del inversor medio. *Boletín de Estudios Económicos* 54 (1999), 361.

Meinard Müller. 2007. Dynamic time warping. *Information retrieval for music and motion* (2007), 69–84.

John J Murphy. 1999. *Technical analysis of the financial markets: A comprehensive guide to trading methods and applications*. Penguin. 1–2 pages.

Cory S Myers and Lawrence R Rabiner. 1981. A comparative study of several dynamic time-warping algorithms for connected-word recognition. *Bell System Technical Journal* 60, 7 (1981), 1389–1409.

Keiron O'shea and Ryan Nash. 2015. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458* (2015).

Python Software Foundation. 2023. *Python Documentation*. `https://docs.python.org/3.11/` Accessed: 2024-06-15.

PyTorch Contributors. 2023. *PyTorch Documentation*. `https://pytorch.org/docs/stable/index.html` Accessed: 2024-06-15.

Qt for Python Development Team. 2023. *Qt for Python Documentation*. `https://doc.qt.io/qtforpython-6/` Accessed: 2024-06-15.

The pandas development team. 2023. *pandas: Python Data Analysis Library*. `https://pandas.pydata.org/docs/` Accessed: 2024-06-15.

Marc Velay and Fabrice Daniel. 2018. Stock chart pattern recognition with deep learning. *arXiv preprint arXiv:1808.00418* (2018).

J.L. Wang and S.H. Chan. 2007. Stock market trading rule discovery using pattern recognition and technical analysis. *Expert Systems with Applications* 33 (2007), 304–315.