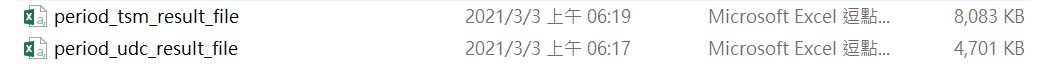
**時間序列之週期異常偵測交接文件**

* Git Project位置: <http://172.22.37.77:8055/anomaly_detection/perioddetection>
* 參考週報範圍: Tony\_Tien\_WeeklyReport\_20200908~20210225
* 交接重點:
  + 1.週期的偵測
    - 1-1 去趨勢方法
    - 1-2 autocorrelation
    - 1-3 週期判定規則
  + 2.對週期序列的特徵提取與建模
    - 2-1 開根號處理scale
    - 2-2 填值演算法
    - 2-3 FFT特徵提取
    - 2-4 OCSVM建模
  + 3.對週期序列的異常偵測
    - 3-1 OCSVM判定異常
    - 3-2 PR用指標計算
    - 3-3 畫圖功能
      * 3-2-1 特徵分布圖
      * 3-2-2 Source IP 線與異常區間圖(主圖)
      * 3-2-3 各期週期走勢圖 (子圖)
      * 3-2-4 General basis圖
* 程式運行流程與辦法:



Detect\_period\_main.py

train\_main.py

Predict\_main.py

輸出路徑: All/Data\_Anomaly\_Detection/ap/summary/

STEP1. Git clone 該專案

STEP2.可透過執行env.sh，幫助建立虛擬環境及install requirement.txt套件，以及執行config.sh對環境config進行自動設置。

STEP3. 開始執行程式。

執行指令EX: python ./perioddetection/PeriodDetect/train\_main.py

--SourceData=total\_sent\_bytes --if\_to\_current=True --if\_training=False

* 程式介紹:

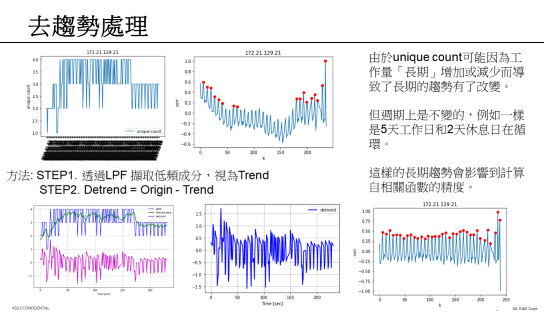
|  |  |  |
| --- | --- | --- |
| PY名稱 | 類別 | 環節 |
| Detect\_period\_main.py | 主程式 | Source Data 資料讀取，包括訓練資料的切割、符合建模的server farm，並針對每個source IP 篩選出有「週期者」。 |
| train\_main.py | 主程式 | 針對有「週期者」的Source data進行前處理、  特徵提取與模型建立等。 |
| Predict\_main.py | 主程式 | 偵測各期是否出現週期異常，計算PR用指標，並畫圖。 |
| Log.py | 紀錄 | 紀錄log作用 |
| Decorator.py | 紀錄 | 紀錄function執行時間 |
| whether\_have\_period.py | 副程式 | 周期偵測演算法(Detect\_period\_main.py會用到) |
| Calculate\_PR\_Indicator.py | 副程式 | 計算PR用指標演算法(Predict\_main.py會用到) |
| DetrendMethod.py | 副程式 | 去趨勢方法統整(whether\_have\_period.py會用到) |
| ModelTraining.py | 副程式 | OSCVM模型與其參數設置(train\_main.py會用到) |
| calculate.py | 函式型 | Autocorrelation演算法、抓取Autocorrelation peak方法、相關係數及FFT特徵計算(train\_main.py、Predict\_main.py) |
| process.py | 函式型 | 程式處理類型，如缺值處理、日期填補、型態轉換等 |
| time\_fun.py | 函式型 | 時間相關function統整 |
| Plot\_function.py | 函式型 | 畫圖相關function統整 |

|  |  |  |
| --- | --- | --- |
| PY名稱 | 類別 | 使用 |
| Schema/dm.py | 設置型 | RawData之欄位Type定義規範 |
| [config\_sample.py](http://172.22.37.77:8055/anomaly_detection/perioddetection/blob/master/PeriodDetect/config_sample.py) | 設置型 | 各項參數、路徑、設置等 |

* 週期的偵測 🡪 對應程式:Detect\_period\_main.py
  1. 去趨勢方法 🡪 對應程式:DetrendMethod

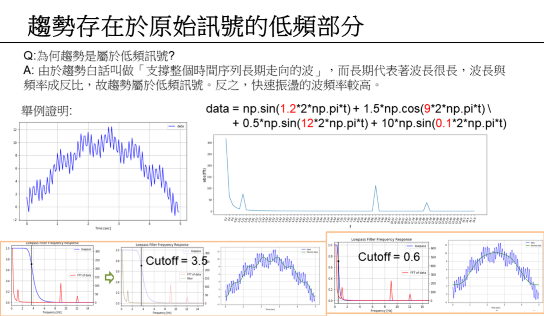
Q1:為何要去趨勢?

A1: Server farm 時常會有慢性的工作量提升或降低，經過一段時間後就會在source data形成「趨勢」。趨勢的產生導致autocorrelation也產生趨勢，較無法順利抓到peak(週期與週期重疊處)，如下圖。

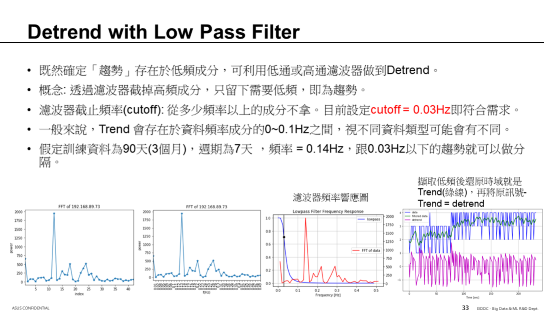


Q2:如何去趨勢?

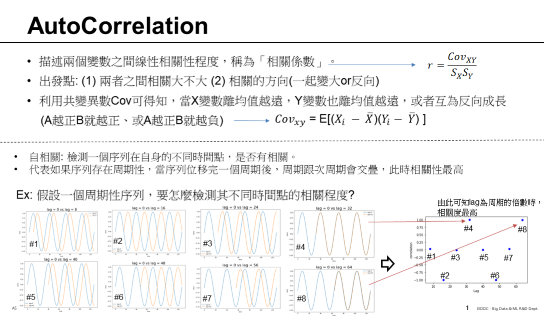
A2:透過趨勢存在於「低頻訊號」的原理，透過FFT轉換將時間序列轉換到頻域，然後砍掉低頻部分，只保留中高頻部分並利用IFFT轉換回時域訊號，即可做到「去趨勢」。

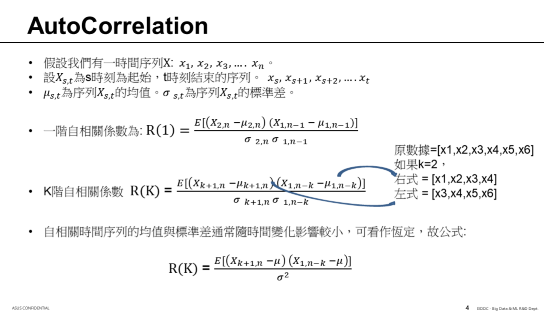


目前是截掉0.03Hz以下的頻率。



* 1. autocorrelation 🡪 對應程式: **Calculate.py** 🡪 **function: autocorrelation。**





* 1. 週期判定規則: 🡪 對應程式: **whether\_have\_period.py** 🡪

**function: detect\_have\_period。**

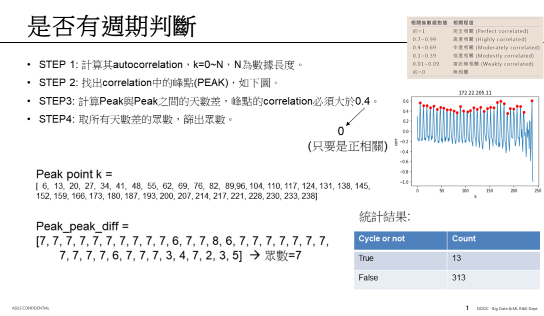
透過autocorrelation得到週期與週期之間重疊的相關係數後，透過peak演算法偵測週期。

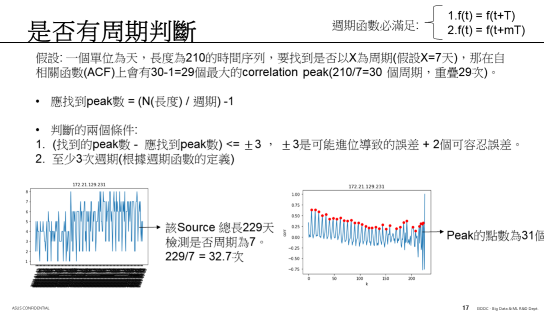
* 判定有週期的條件:

1. 峰點們的correlation > 0.4
2. (找到的peak數 - 應找到peak數) <= 3 (後來改為4)

* 判定週期為多少:

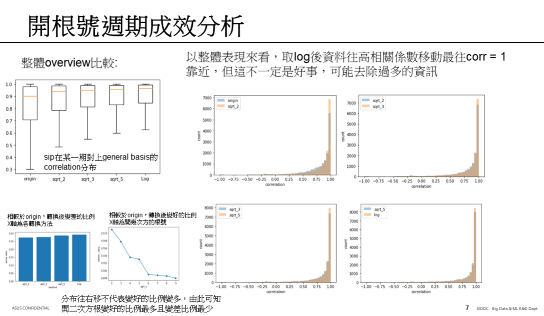
1. 取峰點間的間格天數，可能會有間格亂跳的情況，故取眾數，如果該眾數>5次，則確定週期。





2-1 開根號處理scale 🡪 對應程式: **train\_main.py**

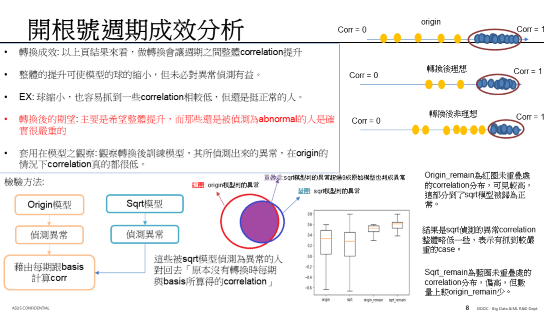
這頁顯示出開越高次方根號，整體計算出來的correlation越容易往1靠近，但實際以比例來說，越高次方根號變差的比例越多。



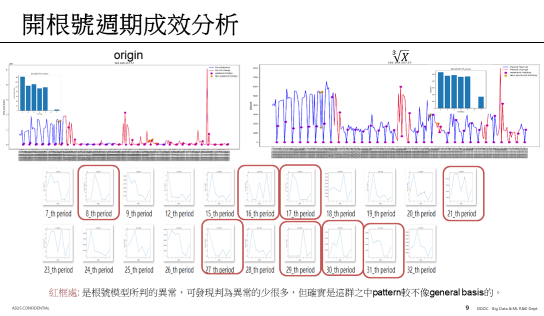
模型球縮小未必對異常偵測有益，原本界在模糊地帶的正常人也可能被歸類為異常。

這頁投影片紅圈為原始不做任何轉換的模型，藍圈為取三次方開根號後的模型結果，而紫色交集處為這兩種模型同時判為異常的部分，藍色剩餘為原本模型判正常但轉換後判異常的比例，紅色剩餘為原本模型判異常但轉換後判正常的比例。

原則上，我們做轉換的目標是為了提升整體correlation使OCSVM訓練的球縮小，進而抓到的異常才是真的非常嚴重的異常，比較原本與開根號模型後，發現開根號模型所抓到的異常相較原本模型抓到的，correlation還要更低。

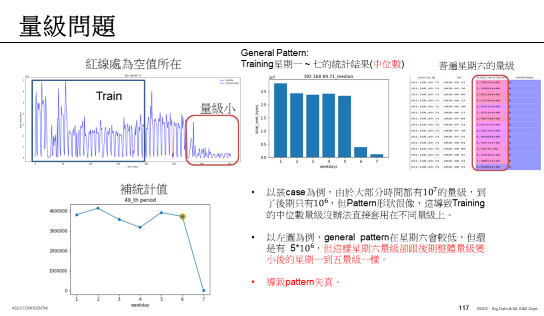
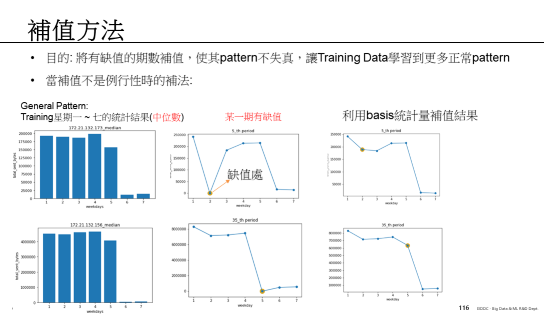


以一個case比較原始模型與開三次方根號模型的比較，異常變少了，留下來的都是真的很嚴重的case。



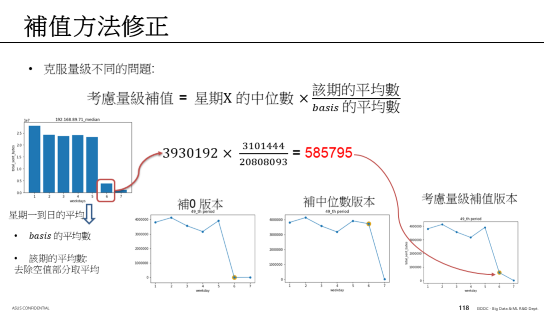
2-2 填值演算法

對應程式: **process.py 🡪** **OnePeriodFill、 fill\_value\_method、fill\_value\_job**



**補值基本上是用General basis去補，何謂general basis其實就是把Training 範圍的每一個週期在對應的時間上做中位數統計，可見一個該server的「大宗」週期走勢。利用到General basis拿對應時間下的中位數補值。**

**然而這會遇到如右邊投影片的「量級問題」。**

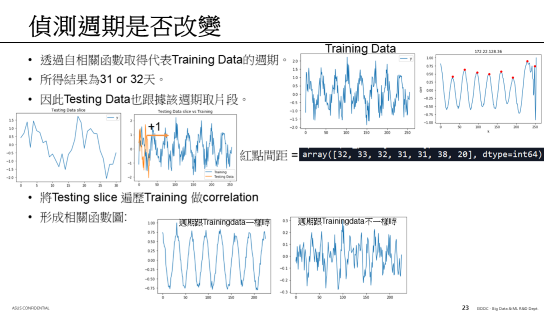


為解決量級改變的問題，必須乘上該其相對應的權重上去，如上投影片的公式。

2-3 FFT與相關係數特徵提取

對應程式: calculate.py **🡪** calculate\_overall\_feature、calculate\_fft\_amp\_with\_correlation、calculate\_fft\_amp\_with\_correlation

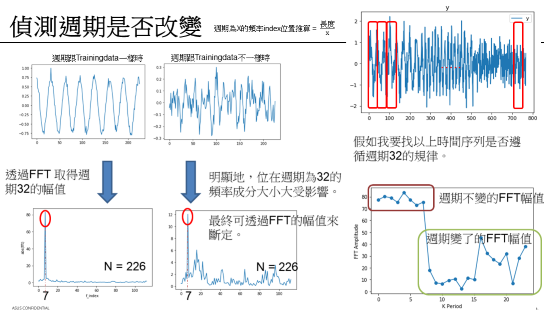




上面投影片模擬一個情境，假設訓練資料是前段的32為週期的時序，當週期改變(中段)或者沒有有週期(後段)了，要能偵測出來。

透過將一期週期遍歷整個Training資料的方式計算出autocorrelation，

由第二張投影片可發現到當週期一致時所得到的autocorrelation較高且較穩定，反之則因為週期無法重疊導致autocorrelation雜訊很高。



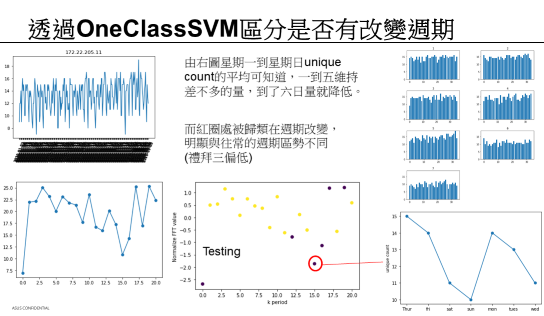
故根據以上實驗，autocorrelation可以拿來當作檢驗「是否週期改變」的一個依據，而該如何將autotcorrelation的穩定程度量化出來，以上有兩種方法可做到，1. 將autocorrelation 時間序列進行FFT轉換 2. 直接抓出autocorrelation每個時間點的峰值(代表不斷位移，週期重疊的時刻)，去計算峰值的相關係數平均。

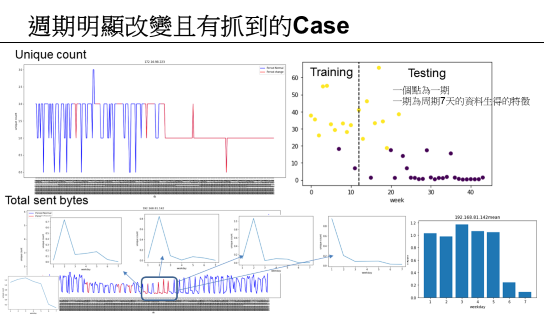
在FFT上的呈現，如果該期週期與Training不一致，所做出來的頻譜在該週期上的頻率成分會降低，反之，頻率成分則會非常明顯，藉此就可透過頻率成分的多寡去判斷。

而計算相關係數平均則是直接反映在該期週期與Training的一致性上，算是很直觀的方法。

2-4 OCSVM建模

對應程式: train\_main.py 🡪 ModelTraining.py 🡪 Period\_modeltraining



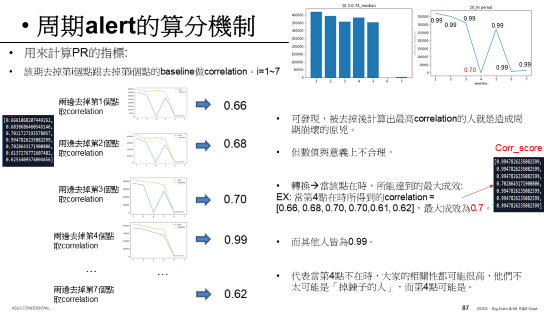


3-1 OCSVM判定異常

對應程式: Predict\_main.py

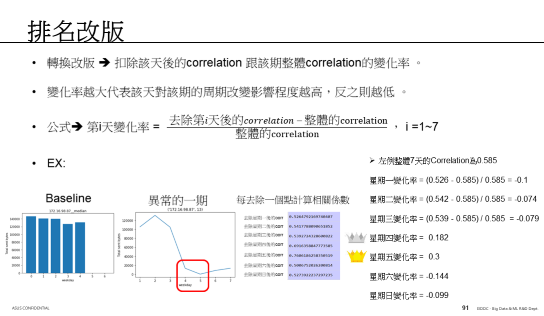
3-2 PR用指標計算

對應程式: **Calculate\_PR\_Indicator.py  🡪** Period\_calcu\_PR\_indicator



該期去掉第i個點跟去掉第i個點的baseline做correlation，可得到當該點被去掉之後的影響，被去掉後計算出最高correlation的人就是造成週期崩壞的原兇。

只憑這數值沒有比較意味，故下頁將該correlation值做了變化率的轉換。



變化率的物理意義是 扣除該天後相關程度如果往下掉，那該點不太可能是影響周期改變的人。

反之，如果扣除該點相關程度往上升，那該點可能就是使周期改變的人，因為相關程度會因為他下降。

實質是計算往下降 還是往上升? 然後下降或上升多少?

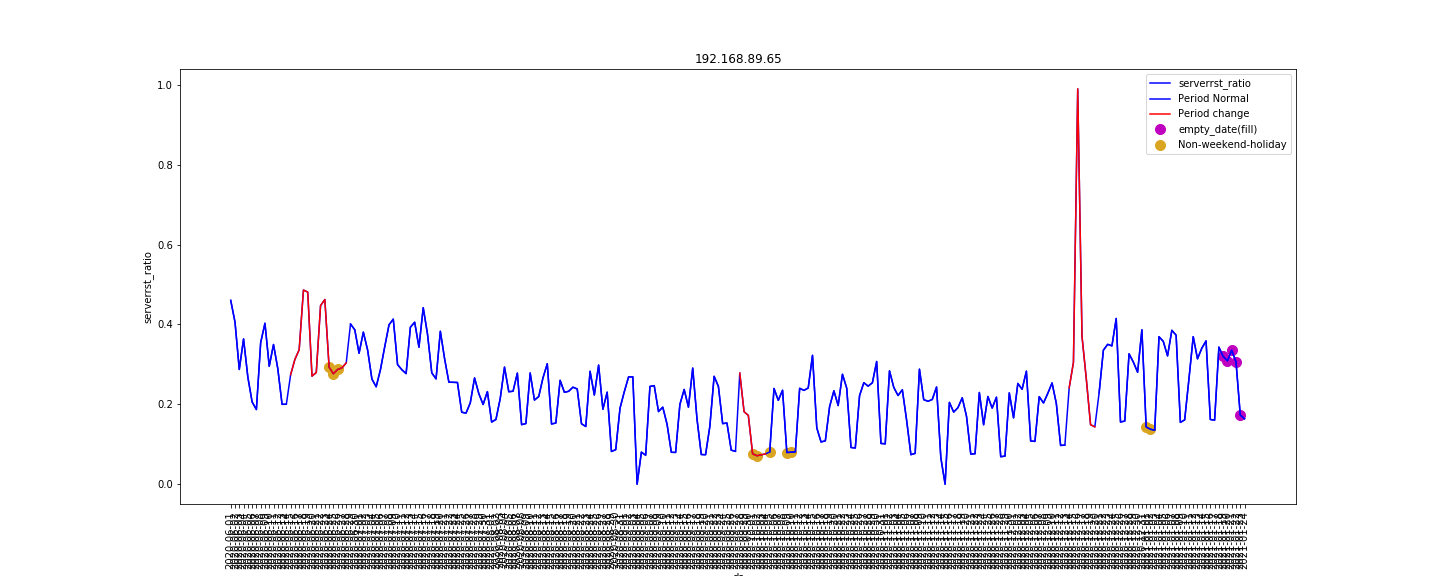
當數值是往下降時，代表去掉該點會造成相關程度降低，反之，當數值是往上升時，代表去掉該點會造成相關程度提升，此點為異常點。

3-3 畫圖功能

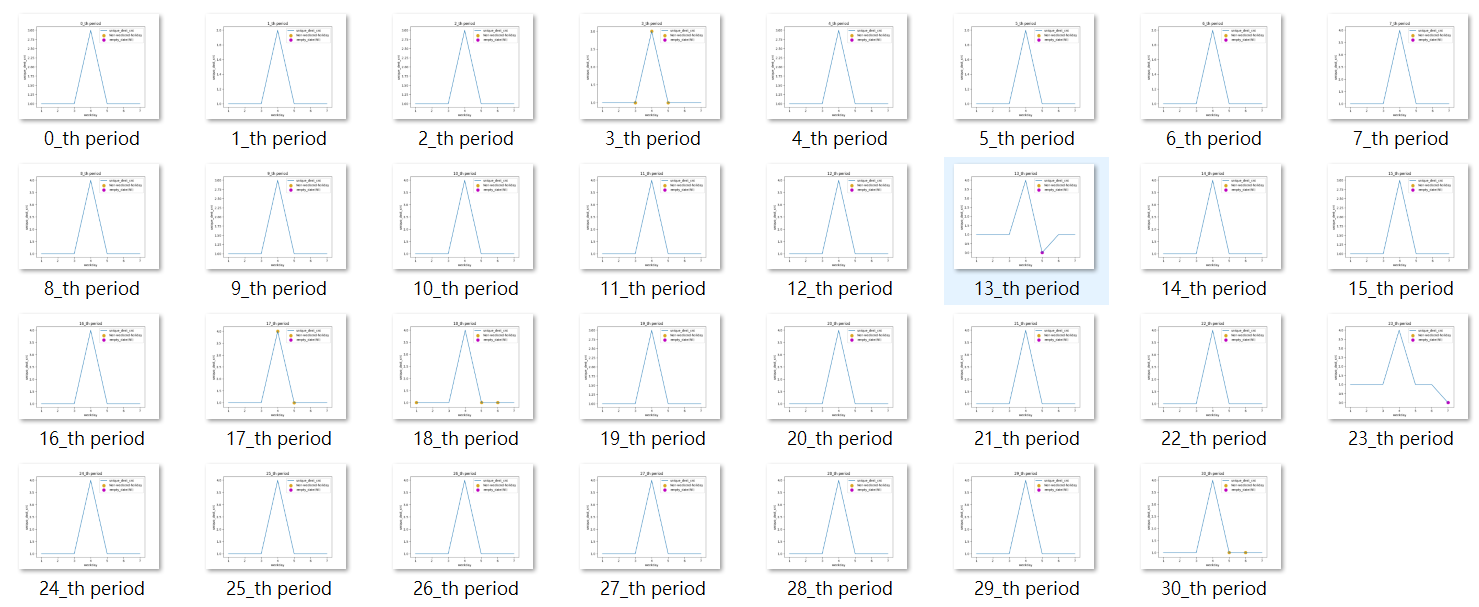
對應程式: plot\_function.py**🡪** plot\_normal\_abnormal\_in\_data、plot\_normal\_abnormal\_period、plot\_feature\_scatter、plot\_cycle\_general

* 周期異常偵測的四種圖呈現:

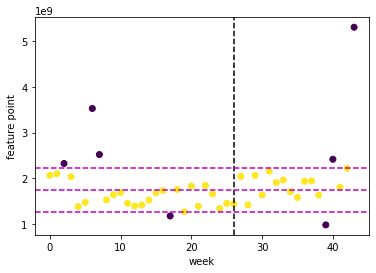
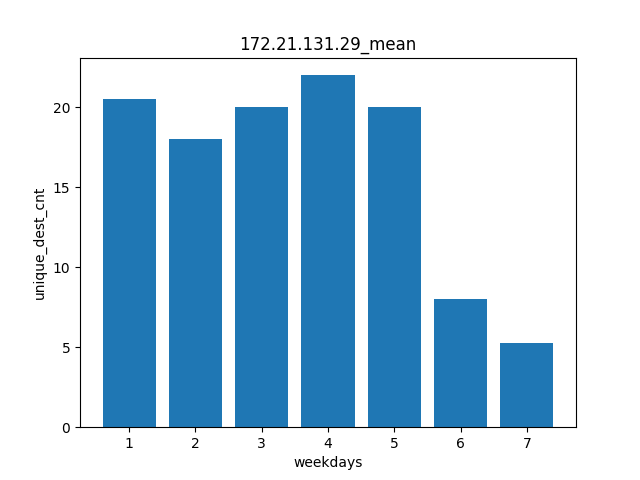
plot\_normal\_abnormal\_in\_data:



plot\_normal\_abnormal\_period:



plot\_feature\_scatter: plot\_cycle\_general:



* 畫圖之開關設置:

請在Predict\_main.py中的plot\_job進行設置，default為不畫圖。

