時間序列降維與符號化：PAA 與 SAX 演算法

投影片 1：封面與議程

專題：時間序列的降維與符號化

(PAA 與 SAX (策略：'Normal') 演算法詳解)

內容大綱

1. SAX 演算法回顧與 Breakpoint 策略
2. `strategy='normal'` (傳統 SAX) 的斷點計算原理
3. PAA (分段聚合近似) 演算法詳解
4. PAA 與 SAX 的整合與距離下界特性

投影片 2：SAX 演算法概覽 (Symbolic Aggregate ApproXimation)

SAX 的核心目標：

將複雜的數值時間序列轉換成簡短的**符號字串**，以便進行更高效的索引和分析。

SAX 三步驟流程 (經典實作)

1. **標準化 (Z-Normalization)：** 調整時間序列，使其平均值 $\mu = 0$，標準差 $\sigma = 1$。
2. **降維 (PAA - 可選)：** 將長度 n 的序列縮短為長度 w 的數值向量。(稍後詳述 PAA)
3. **符號化 (Symbolization)：** 根據預先定義的**斷點 (Breakpoints)**，將 w 維的數值轉換成 α 個符號（例如：a, b, c, d）。

投影片 3：SAX 斷點策略比較 (`n_bins = \alpha`)

SAX 演算法如何劃分數值到符號區間，取決於選擇的策略 (`strategy`)。

策略	原理	數據分佈特性	符號化適用性
'uniform' (等寬)	在數值範圍 (Max - Min) 上平均劃分 α 個等寬區間。	假設數據在原始範圍內均勻分佈。	適用於分佈極端不均的數據。
'quantile' (等頻)	讓每個區間包含**相同數量**的數據點。	不依賴特定分佈，僅依賴數據自身的百分位數。	適用於所有數據集，是穩健的選擇。
'normal' (常態)	根據**標準常態分佈**的等概率面積來劃分區間。	假設數據在 Z-標準化後近似服從 $N(0, 1)$。	**傳統 SAX** 使用，確保每個符號在理論上具有相同發生機率。

投影片 4： `strategy='normal'` 的核心原理

傳統 SAX (Normal Strategy) 的定義

- **目標：** 在**標準常態分佈曲線**下，劃分出 α 個**等概率 (Equal-Probable)** 的區域。

- **機率值：** 每個區域的面積（機率） $P = \frac{1}{\alpha}$。
- **斷點數量：** $\alpha - 1$ 個斷點 ($\beta_1, \beta_2, \ldots, \beta_{\alpha-1}$)。

斷點的意義

斷點 β_j 是 Z 軸上的數值，使得從 $-\infty$ 到 β_j 的累積機率面積正好是 $\frac{j}{\alpha}$。

公式：

$$\Phi(\beta_j) = P(Z \leq \beta_j) = \frac{j}{\alpha}$$

其中 Φ 是標準常態分佈的累積機率分佈函式 (CDF)。

投影片 5：斷點計算：數學依據

1. 關係：PDF 與 CDF

- **機率密度函式 (PDF, $\phi(z)$)：** 曲線下的面積即為機率。
- **累積機率函式 (CDF, $\Phi(z)$)：** 是 PDF 的積分，表示 $P(Z \leq z)$。

2. 斷點的計算

由於我們知道目標機率 ($\frac{j}{\alpha}$)，需要反推對應的 Z 值 (β_j)，這是一個**逆運算**。

$$\beta_j = \Phi^{-1}\left(\frac{j}{\alpha}\right)$$

- Φ^{-1} 被稱為**標準常態分佈分位數函式 (Inverse CDF / Quantile Function)**。
- 計算需要查閱 Z 表或使用統計軟體庫 (例如 Python `scipy.stats.norm.ppf`)。

投影片 6：範例計算：$\alpha = 4$

假設符號數量 $\alpha = 4$，需要 3 個斷點。每個區域的機率為 $1/4 = 0.25$。

斷點編號 j	目標累積機率 $\frac{j}{4}$	斷點 $\beta_j = \Phi^{-1}(\frac{j}{4})$
$j = 1$	0.25	$\beta_1 \approx **-0.6745**$
$j = 2$	0.50	$\beta_2 = **0.0000**$ (中位數/平均值)
$j = 3$	0.75	$\beta_3 \approx **0.6745**$ (與 β_1 對稱)

符號區間劃分

區間	Z-Score 範圍	累積機率面積	符號
A	$(-\infty, -0.67]$	0.25	a

B	$(-0.67, 0.00]$	0.25	b
C	$(0.00, 0.67]$	0.25	c
D	$(0.67, \infty)$	0.25	d

投影片 7：PAA 演算法介紹 (Piecewise Aggregate Approximation)

PAA 的目的

PAA 是 SAX 之前的**降維 (Dimensionality Reduction)** 步驟。它用分段的平均值來近似原始時間序列，達到**資料壓縮**的效果。

核心思想

用一個長度較短的向量來代表原始序列的**主要形狀**和**趨勢**。

參數

- n: 原始時間序列長度。
- w: 降維後的目標長度 ($w < n$)。

降維效果展示

投影片 8：PAA 演算法操作步驟

假設原始序列 C 長度 $n = 8$，目標長度 $w = 4$。

步驟 1: 確定分段長度

- 分段長度 $L = \frac{n}{w}$
- 範例中：$L = \frac{8}{4} = 2$。

步驟 2: 劃分與計算平均值

將原始序列劃分成 w 個不重疊的等長片段。

- **第 i 個片段的數值 $\bar{c}_i$：**

$$\bar{c}_i = \frac{1}{L} \sum_{k=1}^{L} c_{(i-1)L+k}$$

步驟 3: 構建 PAA 向量

將計算出的 $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_w$ 組合起來形成降維後的 PAA 向量 $\bar{C}$。

投影片 9：PAA 實例演示

原始序列 (n=8):

$$C = (10, 12, 14, 16, 5, 8, 11, 14)$$

片段 i	原始數據點	計算 $\bar{c}_i$ (平均值)	PAA 向量 $\bar{C}$
1	c_1, c_2 (10, 12)	$\frac{10+12}{2} = 11.0$	**11.0**
2	c_3, c_4 (14, 16)	$\frac{14+16}{2} = 15.0$	**15.0**
3	c_5, c_6 (5, 8)	$\frac{5+8}{2} = 6.5$	**6.5**
4	c_7, c_8 (11, 14)	$\frac{11+14}{2} = 12.5$	**12.5**

PAA 降維結果 (w=4):

$$\bar{C} = (11.0, 15.0, 6.5, 12.5)$$

原始的 8 點序列被成功壓縮為 4 點序列。

投影片 10：PAA 與 SAX 的整合優勢

PAA 的距離下界 (Lower Bounding) 特性

PAA 演算法的關鍵優勢在於，降維後的 PAA 距離是原始歐氏距離的**有效下界**。

數學表達：

$$\text{PAA 距離}(\bar{C}, \bar{Q}) \leq \text{歐氏距離}(C, Q)$$

$$\sqrt{\frac{n}{w}\sum_{i=1}^{w}(\bar{c}_i - \bar{q}_i)^2} \leq \sqrt{\sum_{j=1}^{n}(c_j - q_j)^2}$$

整合 SAX 的優勢 (SAX Lower Bounding)

SAX 在 PAA 基礎上進一步符號化，同時也保留了距離下界特性（SAX 距離也是 PAA 距離的下界）。

實際應用：

1. **高效查詢：** 在大型資料庫中，可以先用 PAA/SAX 快速篩選掉與查詢序列距離過遠的候選者。
2. **減少計算：** 只對少數被選中的候選序列進行原始、昂貴的歐氏距離計算。