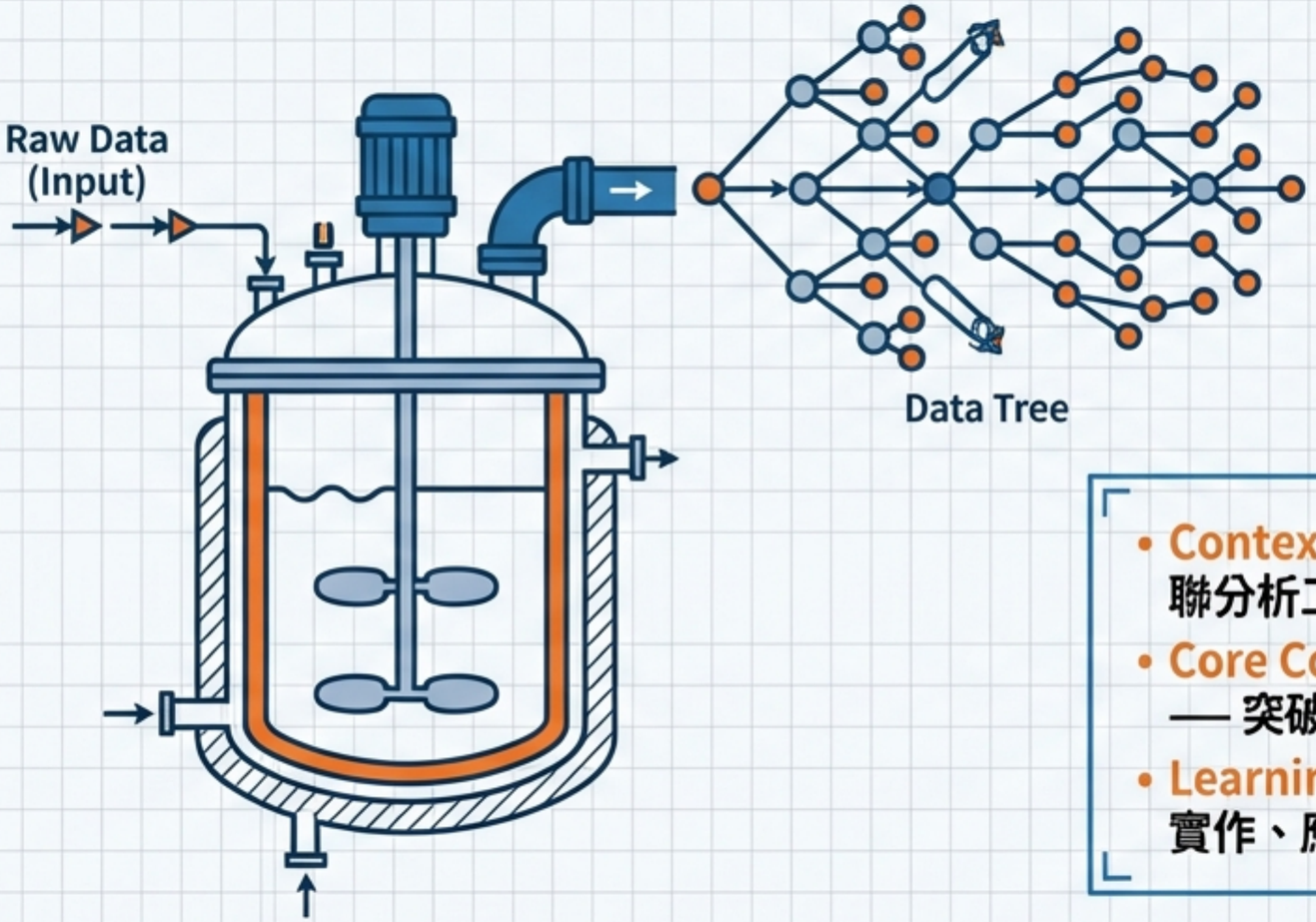
Python 程式語言基礎：建構數位化工廠的基石

Unit 08 FP-Growth 演算法：高效能配方數據挖掘反應器

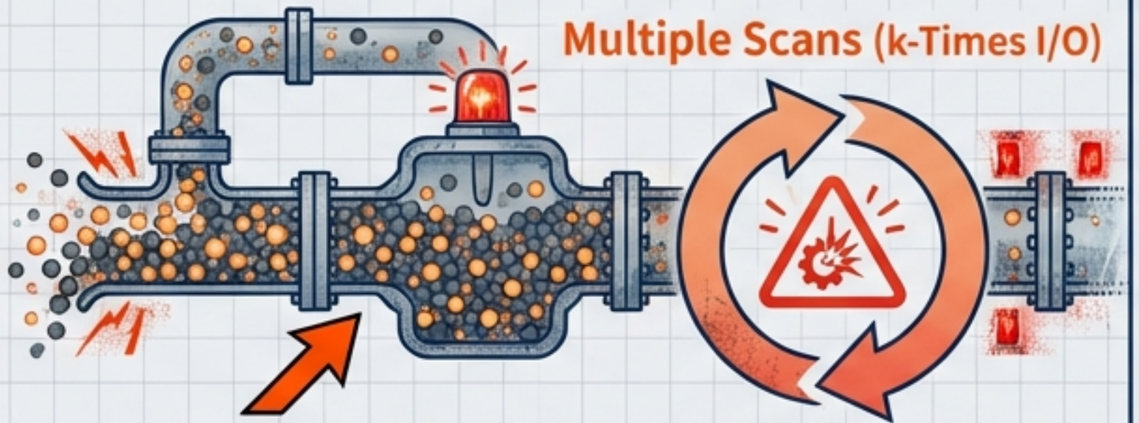


- **Context**: 針對大規模化工配方與製程數據的進階關聯分析工具。
- **Core Concept**: “Divide and Conquer” (分而治之) —— 突破 Apriori 效能瓶頸的關鍵技術。
- **Learning Goal**: 理解 FP-Tree 結構、掌握 Python 實作、應用於聚合物配方優化。

逢甲大學 化工系 智慧程序系統工程實驗室｜課程製作：莊曜禎 助理教授

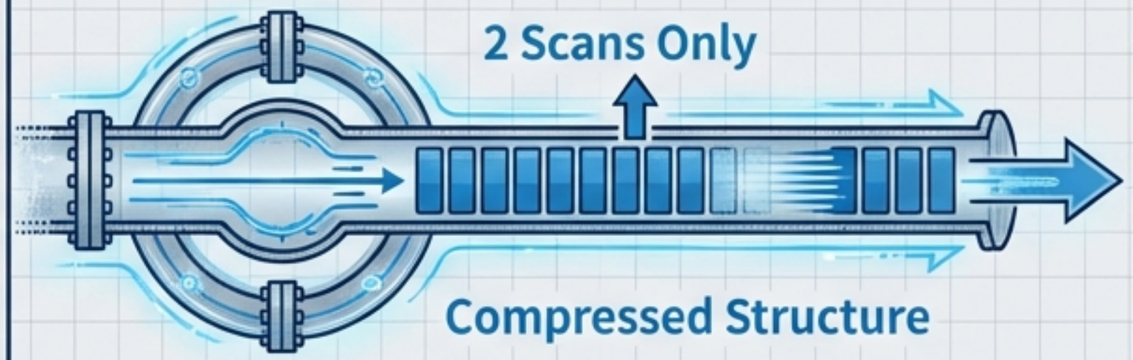
效能瓶頸分析：為什麼 Apriori 不足以應對大數據？

Apriori (Traditional)



Multiple I/O Scans = Performance Bottleneck

FP-Growth (Optimized)



Efficient Linear Data Flow = High Throughput

Engineering Metrics (Process Time)

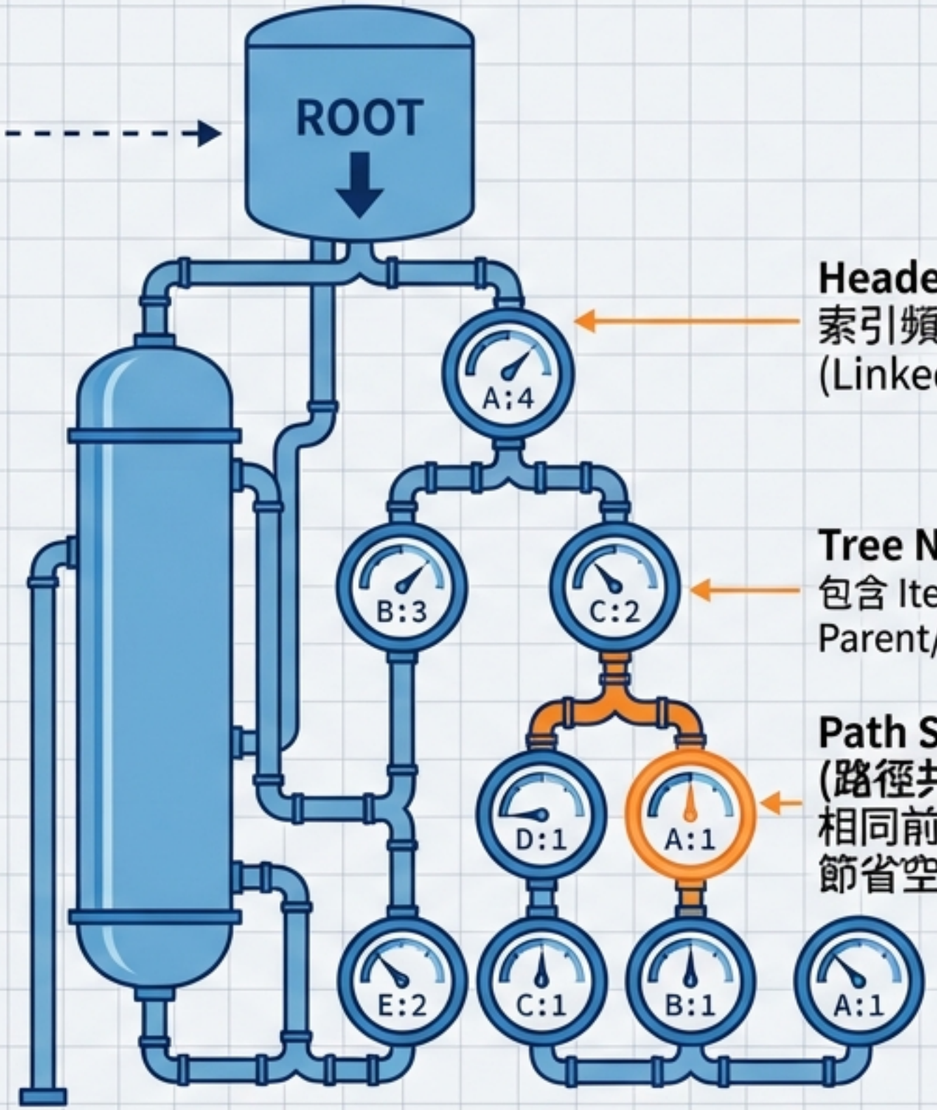
Recipes	Apriori	FP-Growth	Speedup
1,000 Recipes	Apriori: 2.3s	FP-Growth: 0.8s	Speedup: 2.9x
10,000 Recipes	Apriori: 45.6s	FP-Growth: 5.2s	Speedup: 8.8x
100,000 Recipes	Apriori: >30 min	FP-Growth: 78s	Speedup: >23x !

FP-Growth 優勢：避免產生海量候選項目集 (No Candidate Generation)，僅需 2 次 I/O 掃描。

核心技術圖解：FP-Tree (Frequent Pattern Tree)

HEADER TABLE (頭表)

Item Name	Count	Link (鏈接)
A	4	
B	3	
C	2	



Header Table (頭表):
索引頻繁項目與鏈接 (Linked List)

Tree Node (節點):
包含 Item Name, Count, Parent/Children Links

Path Sharing (路徑共享):
相同前綴合併以節省空間

ENGINEERING ANALOGY

- **Function:** 高度壓縮的前綴樹 (Compressed Prefix Tree)
- **Efficiency:** 壓縮至原始大小的 1/10 ~ 1/100
- **Integrity:** 無資訊損失 (Lossless)

演算法流程圖：兩階段分治策略

Phase 1: Construction (建構階段)

Scan 1
Roboto Mono

計算頻率 &
建立 F-List

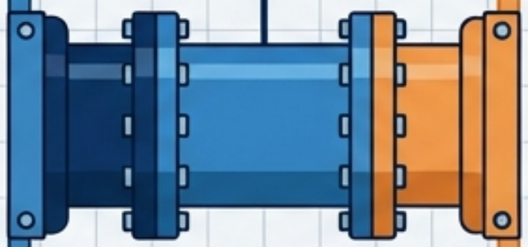
Scan 2
Roboto Mono

重新排序交易

插入樹結構
(Insert into Tree)

更新計數

Compressed
Data Structure



Phase 2: Mining (挖掘階段)

Bottom-Up
Strategy

從頭表底部
項目開始

Conditional
Pattern Base

找出前綴路徑

Recursive
Mining

構建條件
FP-Tree

遞歸運算

核心概念：分而治之 (Divide-and-Conquer) — 將大任務分解為獨立的小路徑運算。

藍圖實作：FP-Tree 建構範例 (T1-T5)

Input Data & F-List

T1: {A, X, High Temp}

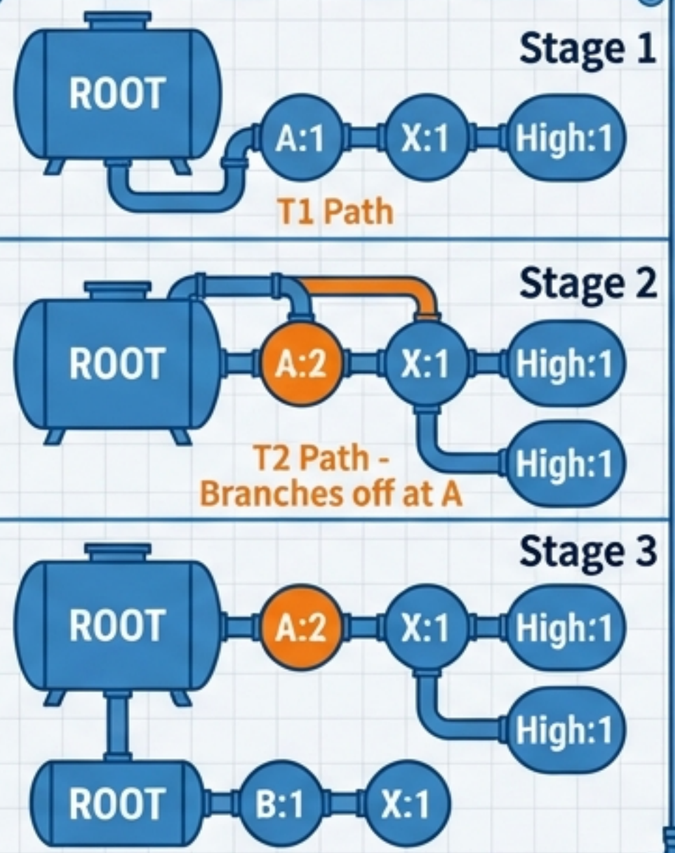
T2: {A, High Temp, Y}

T3: {B, X, Low Temp}

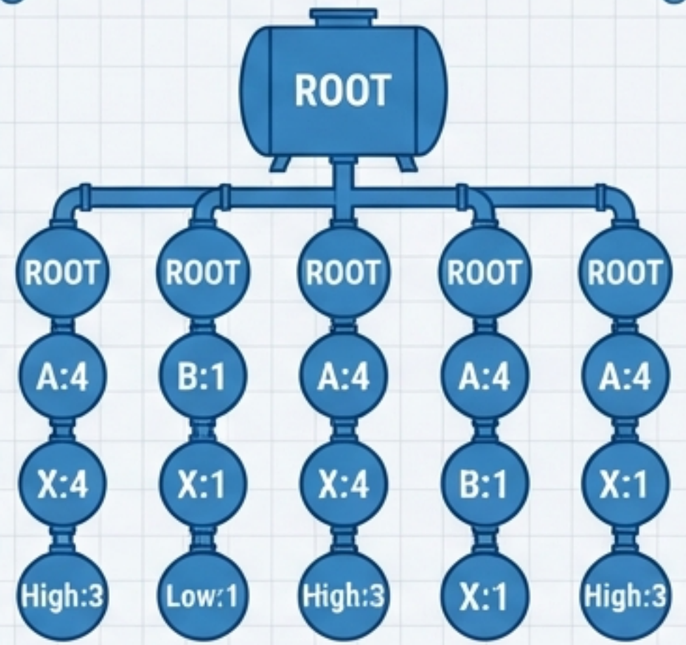
F-List (Sorted by Freq):

A(4) -> X(4) -> High Temp(3)

Tree Construction (Step-by-Step)

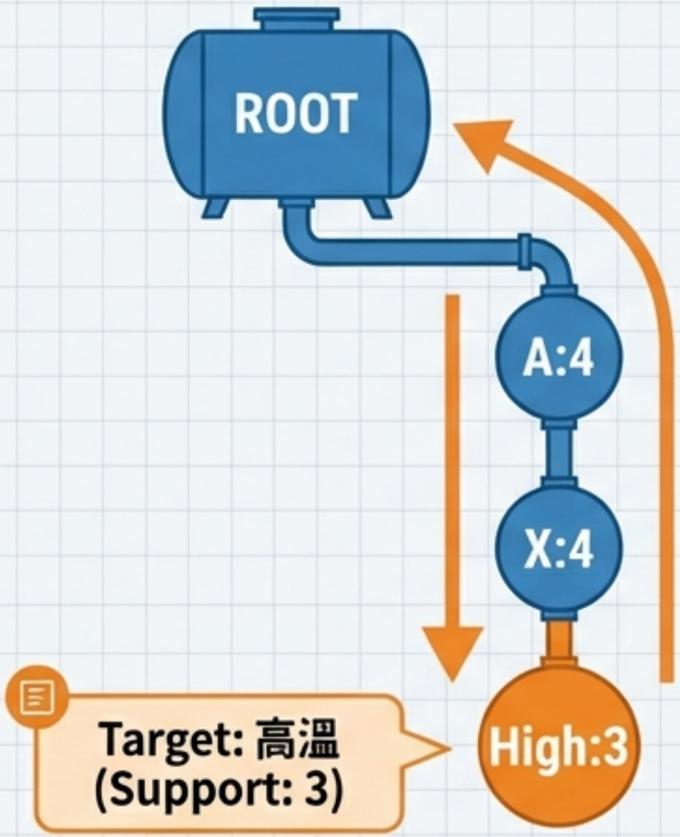


Final State



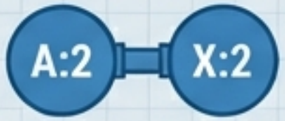
相同前綴路徑共享節點，實現數據壓縮。

數據萃取：從 FP-Tree 挖掘頻繁模式

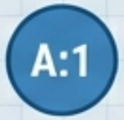


Conditional Pattern Base (條件模式基)

Prefix Path 1: {Catalyst A, Solvent X} (Count: 2)



Prefix Path 2: {Catalyst A} (Count: 1)



Generated Patterns (規則產生):

- {A, High Temp}
- {X, High Temp}
- {A, X, High Temp}

優勢：不需產生候選集，直接通過路徑回溯 (Backtracking) 組合出規則。

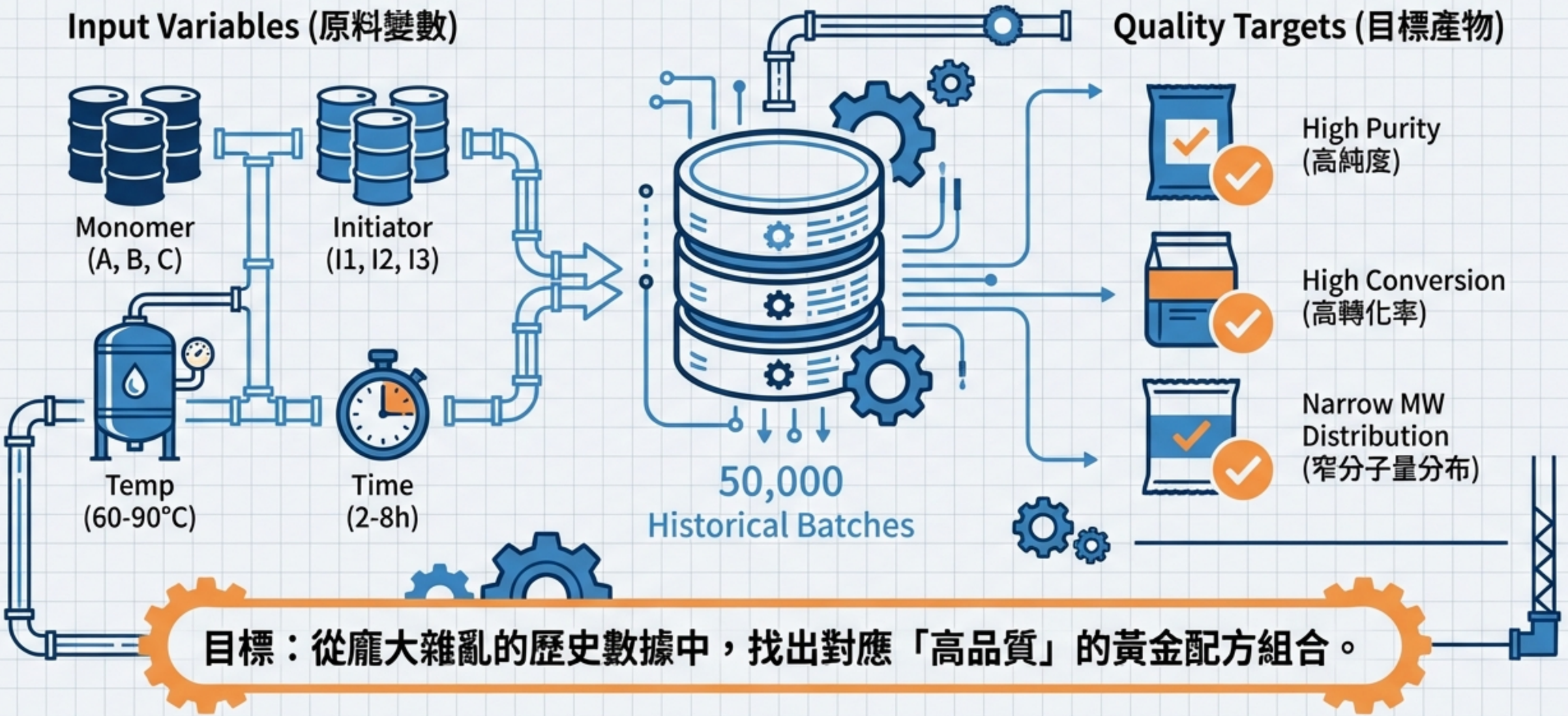
控制核心：Python mlxtend 實作指南

```python
import pandas as pd
from mlxtend.frequent_patterns import fpgrowth

# 1. 數據編碼 (One-Hot Encoding)
# df_encoded = ... (True/False format)

# 2. 執行 FP-Growth
frequent_itemsets = fpgrowth(
    df_encoded,
    min_support=0.02,  # 最小支持度 2%
    use_colnames=True  # 保持原始名稱
)

# 3. 結果輸出
print(frequent_itemsets.sort_values(by='support', ascending=False))
```

決定過濾雜訊的閥值

保持配方名稱，便於工程解讀

應用案例：50,000 筆聚合物配方優化



目標：從龐大雜亂的歷史數據中，找出對應「高品質」的黃金配方組合。

運算效能驗證：FP-Growth vs Apriori

3.00x
FASTER (加速)

Accuracy Check (準確度驗證):

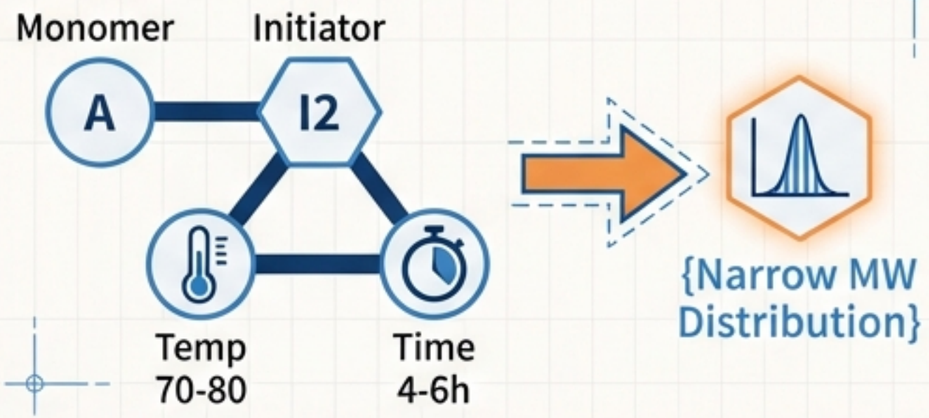
- Frequent Itemsets Found: 1,175 (Both Algorithms)
- Status: 結果完全一致，效率顯著提升。

結論：對於大規模化工數據 (>10,000 筆)，FP-Growth 是標準選擇。

工程洞察：工程洞察：挖掘出的黃金配方規則

RULE ID: 1043

Recipe



Metrics

Lift:
17.36
(High Impact)

Confidence:
85.68%
85%

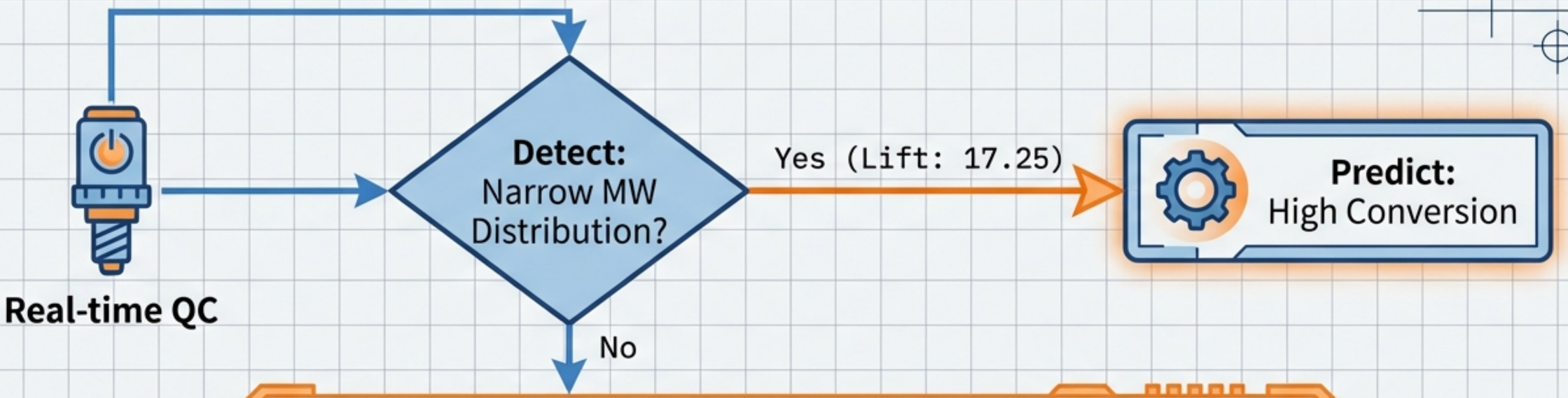
RULE ID: 763

Recipe

Insight

揭示了鏈轉移劑與特定溶劑的化學協同效應 (Synergy)。

製程監控指標：分子量分布的預測價值



Rule 990 Insight:
「窄分子量分布」是「高轉化率」的強預測指標。

Action Item: 將 MW Distribution 納入即時預警系統。
若變寬，預警轉化率即將下降。

實際應用：新世代高效能配方設計

New Product Development: High Purity & Conversion

1. Base Formulation (Data-Driven)

- Core Ingredients:
  Monomer A + Initiator I_2

- Process Conditions:

Temp
70-80°C

+

Time
4-6h

2. Modifiers (Optimization)

- Additives:
  Solvent S2 + CT1
  (Boost Purity)

+

3. Predicted Performance

- High Purity Probability:

~78%

- Narrow MW
  Probability: ~85%

Value: 減少盲目試錯 (Trial-and-Error)，實驗開發成本降低 50%+

決策矩陣：工程師該如何選擇演算法？

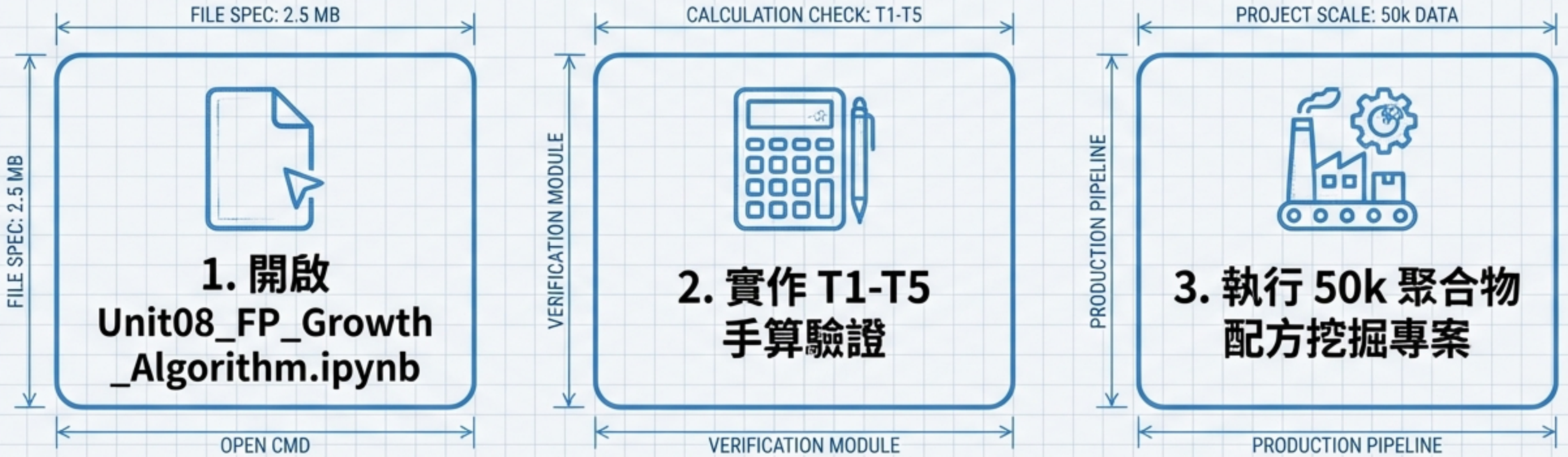
Scenario (應用場景)	Select (建議選擇)	Reason (決策理由)
大規模歷史挖掘 (>10k Data)	FP-Growth	速度優先
實時推薦系統 (Real-time)	FP-Growth	響應快
低支持度/異常偵測	FP-Growth	無候選集瓶頸
教學演示/極小數據	Apriori	直觀、易解釋

總結：選擇取決於數據規模與即時性需求。

實務操作檢查表 (Best Practices Checklist)

"Follow the process, trust the data, and validate the results."

下一步：啟動您的 Jupyter Notebook



前往實作：理論是不夠的，您需要親自寫 Code。
將數據轉化為工廠產能的關鍵就在您手中。