FP-Algorithm

DATA MINING

PC

❖ خوارزمية FP-Growth :

هي خوارزمية تستخدم في تنقيب البيانات لاستخراج الأنماط المتكررة من قاعدة البيانات. تعتمد الخوارزمية على استخدام هيكل بيانات شجرة FP (FP-Tree) لتمثيل البيانات وتسهيل استخراج الأنماط المتكررة.

❖ خطوات عمل خوارزمية:FP-Growth

1) جمع البيانات: يتم تجميع قاعدة البيانات التي تحتوي على مجموعة من المعاملات .(transactions) يتكون كل معاملة من مجموعة من العناصر (items)المرتبطة ببعضها البعض.

2) حساب الدعم الدنيا :(Minimum Support) يجب تحديد قيمة الدعم الدنيا المطلوبة لاستخراج الأنماط المتكررة. يتم استخدام قيمة الدعم الدنيا لتحديد ما إذا كانت المجموعات الفرعية من العناصر تلبي معيار الدعم المحدد أم لا

3) بناء شجرة :FP (FP-Tree) يتم بناء شجرة FP بناءً على معاملات البيانات. تبدأ عملية البناء بإنشاء عقدة جذر للشجرة. ثم يتم مسح المعاملات وترتيب العناصر في كل معاملة وفقًا لترتيب الدعم النازل. يتم إنشاء فرع لكل عنصر في المعاملة وتحديث الشجرة بناءً على العناصر الفرعية. يتم تكرار هذه العملية لجميع المعاملات في قاعدة البيانات.

4) تشكيل الأنماط المتكررة: يتم استخراج الأنماط المتكررة من الشجرة FP.يتم ذلك بتنفيذ عملية ترجيع (backtracking) على الشجرة واستخلاص المسارات الممكنة. يتم تحديد قيمة الدعم لكل نمط مستخرج ويتم التحقق مما إذا كانت تلبي معيار الدعم المحدد. يتم توليد الأنماط المتكررة النهائية.

5) توليد الأنماط المرتبطة: بعد استخراج الأنماط المتكررة، يمكن توليد الأنماط المرتبطة (association rules) من هذه الأنماط. يتم حساب قيمة الثقة (confidence) والقيمة الثقة المؤكدة (support confidence)لكل قاعدة مرتبطة. يتم استخدام الثقة لتحليل الارتباط بين العناصر وتحديد القواعد المهمة.

بهذه الطريقة، تعمل خوارزمية FP-Growthعلى استخراج الأنماط المتكررة من قاعدة البيانات.

❖ مميزات خوارزمية FP-Growth في تنقيب البيانات:

1) كفاءة عالية: تتطلب خوارزمية FP-Growth مسح قاعدة البيانات مرتين فقط، بالمقارنة مع خوارزمية Apriori التي تقوم بمسح المعاملات في كل تكرار. هذا يقلل من التكلفة الزمنية والحوسبية للخوارزمية.

2) سرعة معالجة البيانات: نظرًا لعدم الحاجة إلى إنشاء مرشحات، يتم تجنب العديد من العمليات الزمنية المكلفة في خوارزمية FP-Growth. بالإضافة إلى ذلك، يتم تخزين قاعدة البيانات بصيغة مضغوطة في الذاكرة، مما يسهل عمليات المعالجة ويزيد من سرعة الخوارزمية.

3) قابلية التوسع: يمكن استخدام خوارزمية FP-Growthلاستخراج أنماط متكررة طويلة وقصيرة، مما يعني أنها قابلة للتوسع لمجموعات البيانات الكبيرة والمعقدة.

4) تقليل تكرار البيانات: يتم تمثيل قاعدة البيانات في شجرةFP ، والتي تقلل من تكرار البيانات المكررة وتقلل من حجم البيانات المعالجة. هذا يؤدي إلى تحسين كفاءة الخوارزمية.

❖ عيوب خوارزمية FP-Growth في تنقيب البيانات:

1) تعقيد البناء: بناء شجرة FP يتطلب عملية تجزئة العناصر وإنشاء المسارات والعقد، وهذا يعني أنها أكثر تعقيدًا وصعوبة في البناء مقارنة بخوارزمية Apriori.

2) تكلفة الموارد: على الرغم من كفاءتها في معالجة البيانات، إلا أن خوارزمية FP-Growth قد تكون مكلفة من حيث استهلاك الموارد مثل الذاكرة ووحدة المعالجة المركزية، خاصة عندما تكون قاعدة البيانات كبيرة جدًا.

3) تحدي التوافق مع الذاكرة: عندما تكون قاعدة البيانات كبيرة جدًا، قد تواجه خوارزمية FP-Growth تحديات فيما يتعلق بتناسبها في الذاكرة المشتركة. قد يتطلب ذلك استخدام تقنيات تخزين مؤقت أو تقسيم قاعدة البيانات إلى أجزاء صغيرة لتجنب هذه المشكلة.

4) قابلية التفاقم: قد يزداد زمن تشغيل خوارزمية FP-Growth بشكل متسارع مع زيادة حجم قاعدة البيانات وعدمعلوماتك. وبالتالي، قد يكون من الصعب استخدامها في بعض الحالات عند التعامل مع بيانات ضخمة.

Example Of FP-Growth Algorithm

Support threshold=50%, Confidence= 60%

Table 1

T1	I1,I2,I3
T2	I2,I3,I4
T3	I4,I5
T4	I1,I2,I4
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

Solution:

Support threshold=50% => 0.5*6= 3 => min_sup=3

1. Count of each item

Table 2

Item	Count
I1	4
I2	5
I3	4
I4	4
I5	2

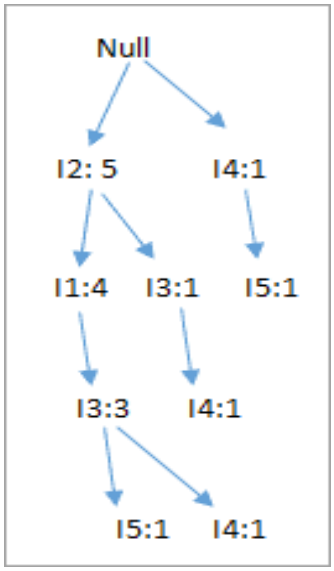
2. Sort the itemset in descending order.

Table 3

Item	Count
I2	5
I1	4
I3	4
I4	4

3. Build FP Tree

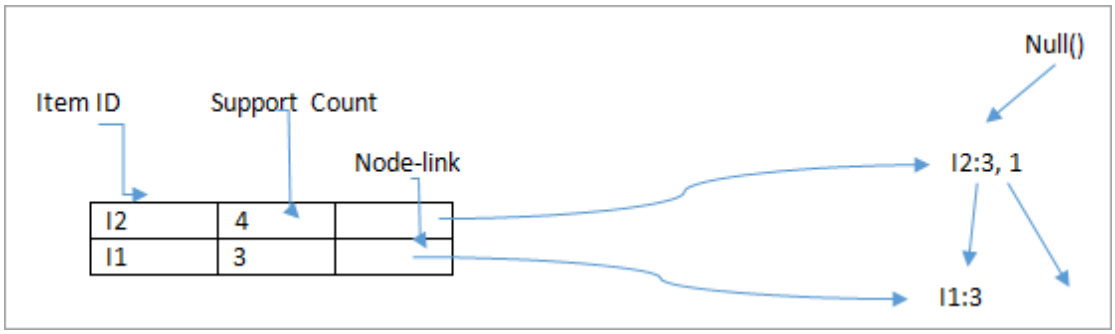
1. Considering the root node null.
2. The first scan of Transaction T1: I1, I2, I3 contains three items {I1:1}, {I2:1}, {I3:1}, where I2 is linked as a child to root, I1 is linked to I2 and I3 is linked to I1.
3. T2: I2, I3, I4 contains I2, I3, and I4, where I2 is linked to root, I3 is linked to I2 and I4 is linked to I3. But this branch would share I2 node as common as it is already used in T1.
4. Increment the count of I2 by 1 and I3 is linked as a child to I2, I4 is linked as a child to I3. The count is {I2:2}, {I3:1}, {I4:1}.
5. T3: I4, I5. Similarly, a new branch with I5 is linked to I4 as a child is created.
6. T4: I1, I2, I4. The sequence will be I2, I1, and I4. I2 is already linked to the root node, hence it will be incremented by 1. Similarly I1 will be incremented by 1 as it is already linked with I2 in T1, thus {I2:3}, {I1:2}, {I4:1}.
7. T5:I1, I2, I3, I5. The sequence will be I2, I1, I3, and I5. Thus {I2:4}, {I1:3}, {I3:2}, {I5:1}.
8. T6: I1, I2, I3, I4. The sequence will be I2, I1, I3, and I4. Thus {I2:5}, {I1:4}, {I3:3}, {I4 1}.



4. Mining of FP-tree is summarized below:

1. The lowest node item I5 is not considered as it does not have a min support count, hence it is deleted.
2. The next lower node is I4. I4 occurs in 2 branches , {I2,I1,I3:,I41},{I2,I3,I4:1}. Therefore considering I4 as suffix the prefix paths will be {I2, I1, I3:1}, {I2, I3: 1}. This forms the conditional pattern base.
3. The conditional pattern base is considered a transaction database, an FP-tree is constructed. This will contain {I2:2, I3:2}, I1 is not considered as it does not meet the min support count.
4. This path will generate all combinations of frequent patterns : {I2,I4:2},{I3,I4:2},{I2,I3,I4:2}
5. For I3, the prefix path would be: {I2,I1:3},{I2:1}, this will generate a 2 node FP-tree : {I2:4, I1:3} and frequent patterns are generated: {I2,I3:4}, {I1:I3:3}, {I2,I1,I3:3}.
6. For I1, the prefix path would be: {I2:4} this will generate a single node FP-tree: {I2:4} and frequent patterns are generated: {I2, I1:4}..

Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I4	{I2,I1,I3:1},{I2,I3:1}	{I2:2, I3:2}	{I2,I4:2},{I3,I4:2},{I2,I3,I4:2}
I3	{I2,I1:3},{I2:1}	{I2:4, I1:3}	{I2,I3:4}, {I1:I3:3}, {I2,I1,I3:3}
I1	{I2:4}	{I2:4}	{I2,I1:4}



ECLAT:

-هو طريقة لاستخراج مجموعات عناصر متكررة باستخدام تنسيق البيانات العمودي. ECLAT

-يقوم ECLAT بتحويل البيانات من تنسيق البيانات الأفقي إلى التنسيق العمودي

.

For Example, Apriori and FP growth use:

Transaction	List of items
T1	I1,I2,I3
T2	I2,I3,I4
T3	I4,I5
T4	I1,I2,I4
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

The ECLAT will have the format of the table as:

Item	Transaction Set
I1	{T1,T4,T5,T6}
I2	{T1,T2,T4,T5,T6}
I3	{T1,T2,T5,T6}
I4	{T2,T3,T4,T5}
I5	{T3,T5}

-يشكل مجموعات عناصر مكونة من عنصرين أو ثلاثة عناصر أو أكثر في التنسيق العمودي.
-يتم زيادة قيمة k بمقدار واحد حتى لا يتم العثور على أي مجموعات عناصر مرشحة.
-يستخدم بعض تقنيات التحسين مثل diffset مع.Apriori

الاستنتاج:
-يُستخدم خوارزمية Apriori لتنقيب قواعد الارتباط. يعمل على المبدأ "يجب أن تكون الأجزاء الفرعية غير الفارغة لمجموعات العناصر المتكررة متكررة أيضًا". يقوم بتشكيل مرشحات مجموعات العناصر k من مجموعات العناصر (k-1) ويمسح قاعدة البيانات لإيجاد مجموعات العناصر المتكررة.
-خوارزمية FP Growth هي طريقة للعثور على أنماط متكررة بدون إنتاج مرشحات. يقوم ببناء شجرة FP بدلاً من استخدام استراتيجية الإنشاء والاختبار في .Apriori يركز خوارزمية FP Growth على تجزئة مسارات العناصر واستخراج أنماط متكررة.