THUẬT TOÁN APRIORI VÀ CÁC THUẬT TOÁN XUẤT PHÁT TỪ APRIORI (APRIORI TID )

* + 1. NGUYÊN TẮC APRIORI
* Đếm số lượng của từng Item, tìm các Item xuất hiện nhiều nhất.
* Tìm các cặp ứng viên: Đếm các cặp => cặp item xuất hiện nhiều nhất.
* Tìm các bộ ba ứng viên: Đếm các bộ ba => bộ ba item xuất hiện nhiều nhất. Và tiếp tục với bộ 4, bộ 5, …
* Nguyên tắc chủ yếu: Mọi tập con của tập phổ biến phải là tập con phổ biến.
  + 1. MÔ TẢ THUẬT TOÁN APRIORI
* **Bước 1**: Đếm số support cho mỗi tập gồm một phần tử và xem chúng như một Large itemset. Support của chúng là minsup.
* **Bước 2**: Với mỗi tập Large item bổ sung các item vào và tạo một Large itemset mới, tập này được gọi là tập ứng viên (Candidate itemset - C). Đếm số support cho mỗi tập C trên cơ sở dữ liệu, từ đó quyết định tập C nào là *Large Item* thực sự, và ta dùng làm hạt giống cho bước kế tiếp.
* **Bước 3**: Lặp lại bước 2 cho đến khi không còn tìm thấy thêm, một tập Large itemset nữa.
  + 1. NỘI DUNG THUẬT TOÁN APRIORI: (IMPLEMENT WITH PSEUDOCODE)

|  |  |
| --- | --- |
|  | **Input:** Tập các giao dịch D, ngưỡng support tối thiểu minsup |
|  | **Output:** L- tập mục phổ biến trong D |
|  | **Method:** |
| 1. | L1=***Large\_1\_ItemSets***() |
| 2. | **for** (k=2; Lk-1 ≠ ∅; k++) **do** |
| 3. | **begin** |
| 4. | Ck=apriori-gen(Lk-1); |
| 5. | **for** (mỗi một giao dịch TD) **do** |
| 6. | **begin** |
| 7. | CT = subset(Ck, T); |
| 8. | **for** (mỗi một ứng cử viên c CT) **do** |
| 9. | c.count++; |
| 10. | **end;** |
| 11. | Lk = {c ∈ Ck| c.count ≥ minsup} |
| 12. | **end;** |
| 13. | return ∪kLk |

* Hàm ***Large\_1\_ItemSets***() trả về các *Item* có số *support* lớn hơn hay bằng minsup.

|  |  |
| --- | --- |
| 1. | ***for all*** *transaction* t ∈ D ***do*** |
| 2. | ***for all*** *item* i ∈ t ***do*** |
| 3. | i.*count* ++; |
| 4. | L1={i | i.*count* ≥ *minsup*}; |

* Hàm ***Apriori\_Gen*** (Lk-1) thực hiện việc kết các cặp (*k-1*) *ItemSet* để phát sinh các tập *k\_ItemSet* mới. Tham số của hàm là Lk-1 – tập tất cả các (*k-1*)-*ItemSet* và kết quả trả về của hàm là tập các *k-ItemSet*.

|  |  |
| --- | --- |
| 1. | ***Join*** Lk-1 ***with*** Lk-1; |
| 2. | ***Insert into*** Ck |
| 3. | ***select*** p.item1,p.item2, . . .p.itemk-1, q.itemk-1 |
| 4. | ***from*** Lk-1 ***as*** p, Lk-1 ***as*** q; |
| 5. | ***where*** (p.item1= q.item1)∧...∧(p.itemk-2 = q.item k-2)∧(p.item k-1<q.item k-1);  Điều kiện (p.item k-1<q.item k-1) sẽ bảo đảm không phát sinh các bộ trùng nhau. |

**Minh họa 1:** Cho một ví dụ tập các giao dịch từ các hóa đơn mua hàng như sau:

|  |  |
| --- | --- |
| **TID** | **Các món hàng được mua (Item)** |
| 1 | { b, m, t, y } |
| 2 | { b, m } |
| 3 | { p, s, t } |
| 4 | { a, b, c, d } |
| 5 | { a, b } |
| 6 | { e, t, y } |
| 7 | { a, b, m } |

Cho *Min Support = 30%,* *Min Confidence = 60%*

**Tính tập Large 1-item, ta có F1:**

|  |  |
| --- | --- |
| **Tập Item** | **Số lần xuất hiện** |
| {a} | 3 |
| {b} | 5 |
| {m} | 3 |
| {t} | 3 |

Ở bước kết Từ F1 trên ta có tập C2 gồm các cặp 2-item:

{{a, b}, {a, m}, {a,t}, {b,m}, {b,t}, {m,t}}

**Tính tập Large 2-item, ta có F2:**

|  |  |
| --- | --- |
| **Tập Item** | **Số lần xuất hiện** |
| {a, b} | 3 |
| {a, m} | 1 |
| {a, t} | 0 |
| {b, m} | 3 |
| {b, t} | 1 |
| {m, t} | 1 |

Chỉ lấy các cặp 2-items có Support > Min Support ( = 30% ) gồm: {a, b} và {b, m}

**Phát sinh luật:**

a → b có độ Confidence 3/3 = 100%

b → a có độ Confidence 3/5 = 60%

b → m có độ Confidence 3/5 = 60%

m → b có độ Confidence 3/3 = 100%

Ở bước lược bỏ ta có F2 = {{a, b}, {b,m}}

Ở bước kết Từ F2 ta có tập C3 gồm các cặp 3-item là {∅}

### Thuật toán kết thúc.

Những cái chưa làm được:

* Code bằng C++
* Vẽ biểu đồ
* Phân tích độ phức tạp của thuật toán và thời gian chạy
* Dịch sang Tiếng Anh

Vẽ biểu đồ bằng python:

Thuật toán Apriori áp dụng để phân tích thị trường. Đây là một ví dụ:

Cho một file chứa các sản phẩm được bán từ tháng 1 đến tháng 12 sau đó ta sẽ tìm ra được sản phẩm ( ứng viên như trên thuật toán Apriori )

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

import seaborn as sns

df = pd.read\_csv("bread basket.csv")

plt.figure(*figsize*=(15,5))

sns.barplot(*x* = df.Item.value\_counts().head(20).index, *y* = df.Item.value\_counts().head(20).values, *palette* = 'gnuplot')

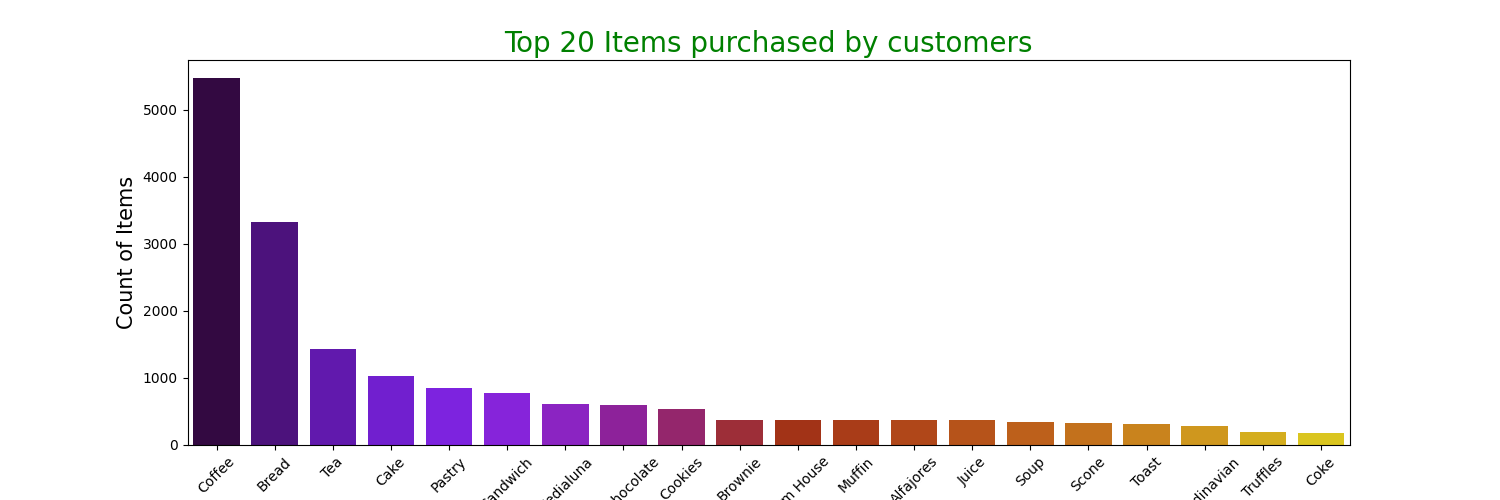
plt.xlabel('Items', *size* = 15)

plt.xticks(*rotation*=45)

plt.ylabel('Count of Items', *size* = 15)

plt.title('Top 20 Items purchased by customers', *color* = 'green', *size* = 20)

plt.show()



Coffee has the highest transactions. ( Coffee là sản phẩm được bán nhiều nhất)

Coke is the 20th most buyed product. ( Coke là sản phẩm được mua nhiều thứ 20)

* + 1. THUẬT TOÁN APRIORITID:
* Thuật toán *AprioriTID* sử dụng hàm *Apriori\_Gen* để tạo các tập *ItemSet* ứng viên. Thuật toán này không dùng cơ sở dữ liệu D để đếm *support* kể từ bước thứ hai, thay vào đó là sử dụng tập Ck cho mục đích này. Mỗi thành viên của tập Ck có dạng <TID, {Xk}> với Xk là tập *k-ItemSet* thể hiện một phần giao tác t có mã là TID, hay ta có thể viết <t.TID, {c∈Ck | c có trong t}>.
* Nếu một giao tác không chứa bất kỳ một tập *k-ItemSet* ứng viên nào, thì giao tác này không được đưa vào . Do đó, số lượng ứng viên được đưa vào  có thể nhỏ hơn số lượng các giao tác trong cơ sở dữ liệu.
  + 1. MÔ PHỎNG THUẬT TOÁN APRIORI-TID
* **Bước 1**: Quét tất cả các giao dịch để tìm tất cả các item có độ Support lớn hơn Min Support và đưa tập Large 1-Item vào F1
* **Bước 2**: Đưa toàn bộ các Tid của giao dịch cùng các Items vào C’1 dưới dạng <Tid,{X1}>
* **Bước 3**: Xây dựng các cặp 2-items từ F1 đưa vào tập ứng viên C2. Quét tất cả các giao dịch trong C’1 để tìm tất cả các tập Large 2-Item từ C2 đưa vào C’2 dưới dạng <Tid,{X2}>, đồng thời đưa các tập Large 2-Item ứng viên vào F2.
* **Bước 4**: Phát sinh Luật. Xây dựng các cặp k items từ Fk-1 đưa vào tập ứng viên Ck. Quét tất cả các giao dịch trong C’k-1 để tìm tất cả các tập Large k-Item từ Ck và đưa vào C’k dưới dạng <Tid,{Xk}>, đồng thời đưa các tập Large k-Item vào Fk. Lặp lại Bước 4 cho đến khi hết ứng viên mới.
  + 1. NỘI DUNG TỐI ƯU THUẬT TOÁN APRIORI-TID

|  |  |
| --- | --- |
| 1. | L1 = ***Large\_1\_ItemSets*** (); |
| 2. | = Database D; |
| 3. | ***for*** (k=2; Lk-1 ≠ ∅ ; k++) ***do begin*** |
| 4. | Ck = ***Apriori\_Gen***(Lk-1); |
| 5. | = ∅; |
| 6. | ***for all*** t∈  ***do begin*** |
| 7. | Ct = {c ∈ Ck | (c-c[k]) ∈ t.Set\_of\_ItemSets ^ |
| 8. | (c-c[k-1] ∈ t.Set\_of\_ItemSets}; |
| 9. | ***for all*** candidate c ∈ Ct ***do*** |
| 10. | c.count ++; |
| 11. | ***if*** (Ct≠∅) ***then*** += < t.TID, Ct >; |
| 12. | ***End*** |
| 13. | Lk = {c ∈| c.count ≥ minsup} |
| 14. | ***End*** |
| 15. | Answer = ∪kLk; |

* + 1. CẤU TRÚC LƯU TRỮ:
* Mỗi tập *ItemSet* ứng viên sẽ được gán cho một mã số duy nhất, gọi là ***ID***. Mỗi tập *ItemSet* Ck được lưu trong một mảng. Một thành viên của  bây giờ có dạng <TID, ID >, mỗi  được lưu trong một cấu trúc tuần tự.
* Hàm ***Apriori\_Gen*** phát sinh một tập các *k-ItemSet* ứng viên Ck bằng cách kết hai tập *Large (k-1)-ItemSets*. Mỗi *ItemSet* ứng viên ta thêm hai trường:

*(i) generators.*

*(ii) extensions.*

* Trường *generators* của tập *ItemSet* ck lưu các ID của hai tập *Large (k-1)-ItemSet* kết với nhau để phát sinh ck.
* Trường *extensions* của tập *ItemSet* ck lưu những ID của các tập *Large (k+1)-ItemSet* kết với nhau để phát sinh ck.
* Khi một *ItemSet* ck ứng viên được phát sinh bằng cách kết 11k-1 và 12k-1, thì các ID của 11k-1 vaø 12k-1 sẽ được lưu vào trường *generators* của ck, đồng thời ID của ck được lưu vào trường *extension* của 11k-1.
* Với cấu trúc lưu trữ này thì câu lệnh

Ct = {c ∈ Ck | (c-c[k]) ∈ t.*Set\_of\_ItemSets* ∧(c-c[k-1] ∈ t.*Set\_of\_ItemSets*};

sẽ được thực hiện như sau: trường t.*Set-of-ItemSets* của bản ghi *t* thuộc  lưu các ID của tập ứng viên (*k-1*)-*ItemSet* chứa trong giao tác t.TID. Với mỗi ck-1, trường *extensions* chứa tập Tk là tập các ID của tất cả các tập *k-ItemSet* ứng viên được mở rộng từ ck-1. Mỗi ck trong Tk, trường *generators* chứa các ID của hai tập *ItemSet* dùng để phát sinh ra ck. Nếu những tập *itemSet* này nằm trong danh sách các tập *ItemSet* của bản ghi *t*, thì có thể kết luận ck thuộc giao tác t.TID, và ck được thêm vào tập Ct.

* + 1. MINH HỌA THUẬT TOÁN APRIORI-TID:

Cho một ví dụ tập các giao dịch Tid với các Items như sau:

|  |  |
| --- | --- |
| **Tid** | **Items** |
| 100 | {1, 3, 4} |
| 200 | {2, 3, 5} |
| 300 | {1, 2, 3, 5} |
| 400 | {2, 5} |

Cho *Min Support* = 50%, *Min Confidence* = 60%

**Tính tập Large 1-item, ta có F1:**

|  |  |
| --- | --- |
| **Tập 1-item** | **Số lần xuất hiện** |
| { 1 } | 2 |
| { 2 } | 3 |
| { 3 } | 3 |
| { 5 } | 3 |
|  |  |

**Lấy toàn bộ <Tid,{X1}> đưa vào C’1**

|  |  |
| --- | --- |
| **Tid** | **Tập 1-Item** |
| 100 | {{1 }, {3}, {4}} |
| 200 | {{2}, {3}, {5}} |
| 300 | {{1}, {2}, {3}, {5}} |
| 400 | {{2}, {5}} |

Ở bước kết Từ F1 trên ta có tập C2 gồm các cặp 2-item:

{{1,2}, {1,3}, {1,5}, {2,3}, {2,5}, {3,5}}.

**Xác định ứng viên từ C2 khi duyệt Tid trong C’1 và đưa vào C’2**

|  |  |
| --- | --- |
| **Tid** | **Tập 2-Item** |
| 100 | {{1,3}} |
| 200 | {{2,3}, {2,5}, {3,5}} |
| 300 | {{1,2}, {1,3}, {1,5},  {2,3}, {2,5}, {3,5}} |
| 400 | {{2,5}} |

Ở bước kết Từ F1 trên ta có tập C2 gồm các cặp 2-item:

{{1,2}, {1,3}, {1,5}, {2,3}, {2,5}, {3,5}}.

**Tính tập Large 2-Item, ta có F2**

|  |  |
| --- | --- |
| **Tập 2-Item** | **Số lần xuất hiện** |
| {1,3} | 2 |
| {2,3} | 2 |
| {2,5} | 3 |
| {3,5} | 2 |

Ở bước kết Từ F2 ta có tập C3 gồm cặp 3-item

{{2,3,5}}.

**Xác định ứng viên từ C3 khi duyệt Tid trong C’2 và đưa vào C’3**

|  |  |
| --- | --- |
| **Tid** | **Tâp 3-Itims** |
| 200 | {{2, 3, 5}} |
| 300 | {{2, 3, 5}} |

**Tính tập Large 3-Item, ta có F3:**

|  |  |
| --- | --- |
| **Tập 3- Item** | **Số lần xuất hiện** |
| {{2, 3, 5}} | 2 |

**Phát sinh luật:**

2,3 🡪 5 có độ Confidence 2/2 = 100%

2,5 🡪3 có độ Confidence 2/3 = 66,66%

3,5 🡪 2 có độ Confidence 2/2 = 100%

**Ở bước kết Từ F3 ta có tập C4 gồm các cặp 4-item là {∅}**

**Thuật toán kết thúc.**

Những cái chưa làm được:

* Code bằng C++
* Vẽ biểu đồ
* Phân tích độ phức tạp của thuật toán và thời gian chạy
* Dịch sang Tiếng Anh