ASSOSICATION RULE MINING (B İRLİKTELİK KURAL ÇIKARIMI)

Bu bölümde genelde:

"şunu alan şunu da aldı" gibi bağlantıların kurulduğu ve kampanya, ürün tavsiyesi gibi alanlarda sıkça kullanılan algoritmaları göreceğiz.

bu bölümde, kısaca ARM (Association Rule Mining) olarak kısaltacağımız aşağıdaki algoritmaları anlatarak uygulamalarını Python dili üzerinden yapacağız.

1. Apriori

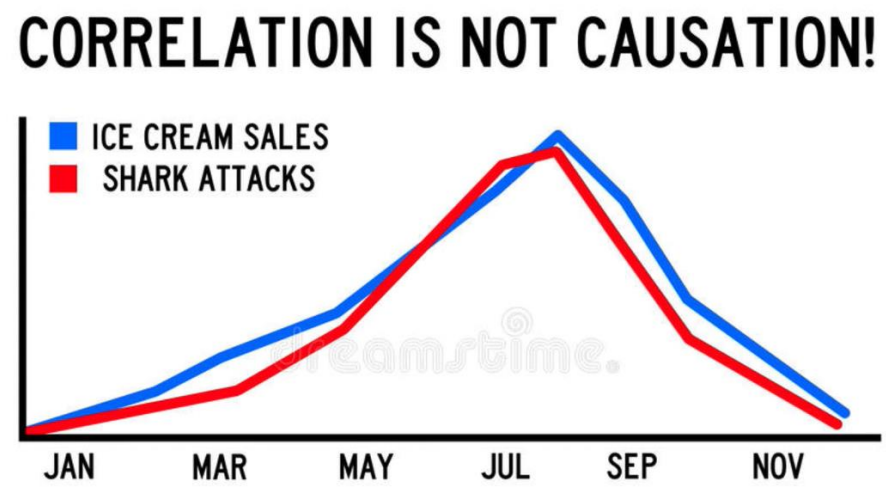
2. Eclat

APRIORI ALGORITHM

Kavram alışveriş sepetlerinden çıkıyor. “Bunu alanlar bunu da aldı”, “Bunu izleyenler bunu da izledi”, “Bunu …’lar bunu da … “ Bu ilişkiyi yakalamayı planlıyor.

Makineler neden sonucu düşünmek zorunda değil. Sadece correletion’a bakar. Bu da aslında bizim göremediğimiz veya nedenini anlayamadığımız bağlantıları kurmalarını sağlar.

Aşağıda dondurma satışı ile köpekbalığı saldırıları arasında bir ilişki var. Biz bunun nedenini anlamayabiliriz ama var. Sıcaklardan oluyor. Ama bu neden bilgisayarın umrunda değil bilgisayar correlation’a bakar.



Both ice cream sales and shark attacks increase when the weather is hot and sunny, but they are not caused by each other (they are caused by good weather, with lots of people at the beach, both eating ice cream and having a swim in the sea)

Algoritmanın anlaşılması için farlı müşterilerin yaptığı tekrarlayan eylemler olduğunu varsayalım. Müşteri1 a,b,c filmlerini izlemiş veya a,b,c ürünlerini almış veya a,b,c reklamlarına tıklamış veya herhangi bir a,b,c eventini gerçekleştirmiş. Müşteri2 başka transaction'lar gerçekleşmiş. Müşteri3 de aynı şekilde başka transactionlar … Sonuçta bu tekrarlanan olaylar arasında bir bağlantı kurma amacını güdüyoruz. Detayını örnekle açıklayacağız.

Önce bazı kavramları tanımlayalım:

Support(A): A varlığını içeren eylemler / Toplam eylem sayısı

→ A'nın supportunu verir. Mesela mağazadan 100 farklı alışveriş yapılmış 10 tanesinde A ürünü alınmış: Support(A) = %10

Confidence(A->B): A ve B varlıklarını içeren eylemler / A varlığını içeren eylemler.

→ Mesela 5 alışverişte A ve B birlikte alınmış olsun. 10 farklı alışverişte A almıştı. 5/10 = %50

Lift (A->B): Confidence (A->b) / Support (A)

→ Lift'in 1'den büyük olması demek A ürününün alınıyor olması B ürününün alınma ihtimalini artırıyor demek. Lift 1'in altında çıkarsa A yı alanlar B yi almamaya daha meilli demek.

Yüz kişiye sorduk

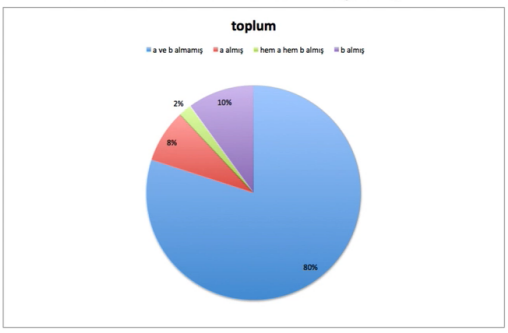
$$\text{Support(a)} = \frac{\text{a varlığını içeren eylemler}}{\text{Toplam Eylem Sayısı}}$$

a ürününü aan 100 kişiye sorduk

$$\text{Confidence(a->b)} = \frac{\text{a ve b varlığını içeren eylemler}}{\text{a varlığını içeren eylemler}}$$

B ürününü amak ne kadar artış sağlar?

$$\text{Lift(a->b)} = \frac{\text{Confidence(a->b)}}{\text{support(b)}}$$



Burada kırmızı olan kırmızı taraf %10, A ürününü alanlar. %12 mor B ürününü alanlar. %2 A ve B alanlar.

Support(A): 0.1

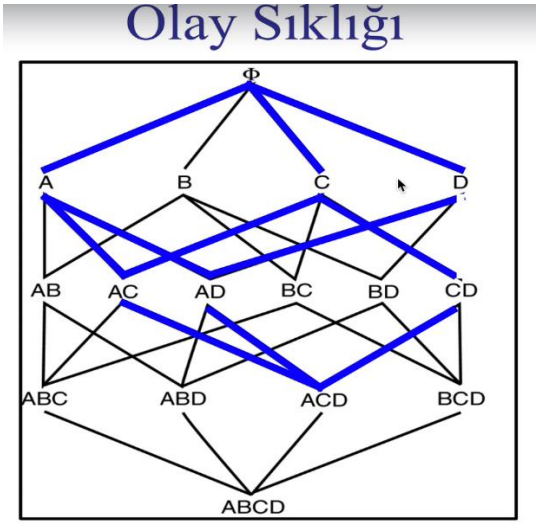
Support(B): 0.12

Confidence (A->B): 0.2

Lift (A->B): 0.2/0.12 = 1.67 . Demekki A'nin alınması B'ya lift sağlar çünkü. Tüm toplumda B nın alınması %12 iken A'yı alanlar arasında bu ihtimal daha fazla 0.2 oluyor. 0.2>0.12 !!!

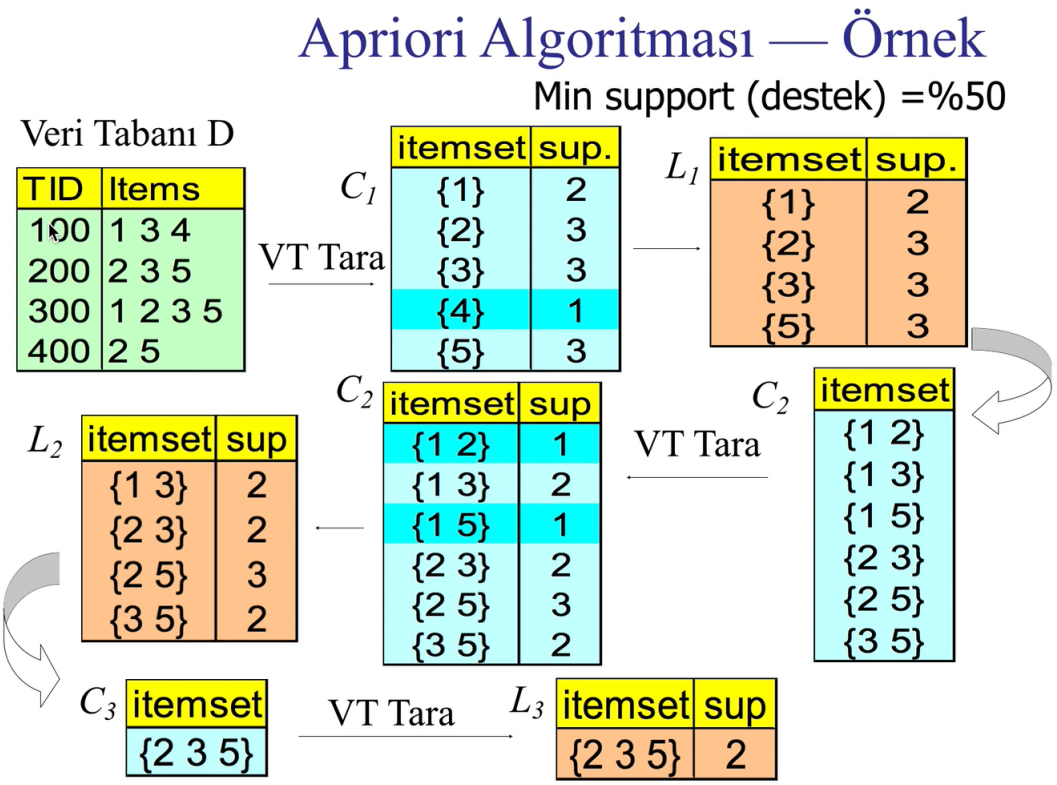
Çalışma Prensibine Geçelim:

Mesela 4 farklı ürünümüz var A,B,C,D bu ürünlerin tek tek support değerlerine bakıyoruz. Belirlenen treshold supportu geçemeyen ürünleri baştan eleriz. Daha sonra onun bağlantıları da elenmiş olur. Çünkü zaten B ürünü %1 alınıyorsa bunun kombinasyonları da max %1 alınacaktır.



Daha sonra elenmeyen ürünlerin ikili olarak ne sıklıkla alındığına bakıyoruz. Yine düşük sıklıkları olanları eliyoruz.

Bir örnek üzerinden gidersek:



Mesela veritabanında fiş id'ler ve o alışverişte alınan ürünler tutulmuş. 100 id'li alışverişte 1,3,4 ürünleri alınmış gibi.

Daha sonra bu veritabanından hangi ürünlerin hangi sıklıkla alındığının tablosu çıkarılıyor. Mesela ürün 1, 100 ve 300 nolu alışverişlerde alınmış yani toplam 2 kez alınmış gibi.

Max sıklık 3 olarak görünüyor o zaman bunun %50'si 1.5 sıklığının altındakileri eleyelim. Yani 4 numaralı ürünü bundan sonra hesaba katımıyoruz. Bunun alıcısı yok zaten.

Geriye 1,2,3,5 itemları kaldı. Şimdi bunların kombinasyonlarına bakacağız. Bunun için kalan ürünlerin tüm kombinasyonlarını yazacağız ve bunların kaç alışverişte birlikte alındıklarına bakacağız. Mesela 1,2 kombinasyonu yalnızca 300 id'li alışverişte birlikte alınmış sıklık 1.

Yine support değerimiz 1.5 , bu sıklığın altındakileri eliyoruz. Geriye (1,3), (2,3), (2,5), (3,5) kalıyor.

Şimdi kalan tabloya bakarsak, mesela 2 numaralı ürünün yanına rafta hangi ürünü koyalım deseler neye koyarız? 5'i koyarız.

Şimdi 3 lü gruplara bakıyoruz zaten elimizde 1,2,3,5 var veritabanında zaten sadece 2 tane 3 elamanı geçen alışveriş var 2 3 5 ve 1 2 3 5 . Son listede 1 2 olmadığına göre 1 2 3 veya 1 2 5 olamaz. (2,3,5) in sıklığı 2 (1 3 5) in sıklığı 1.

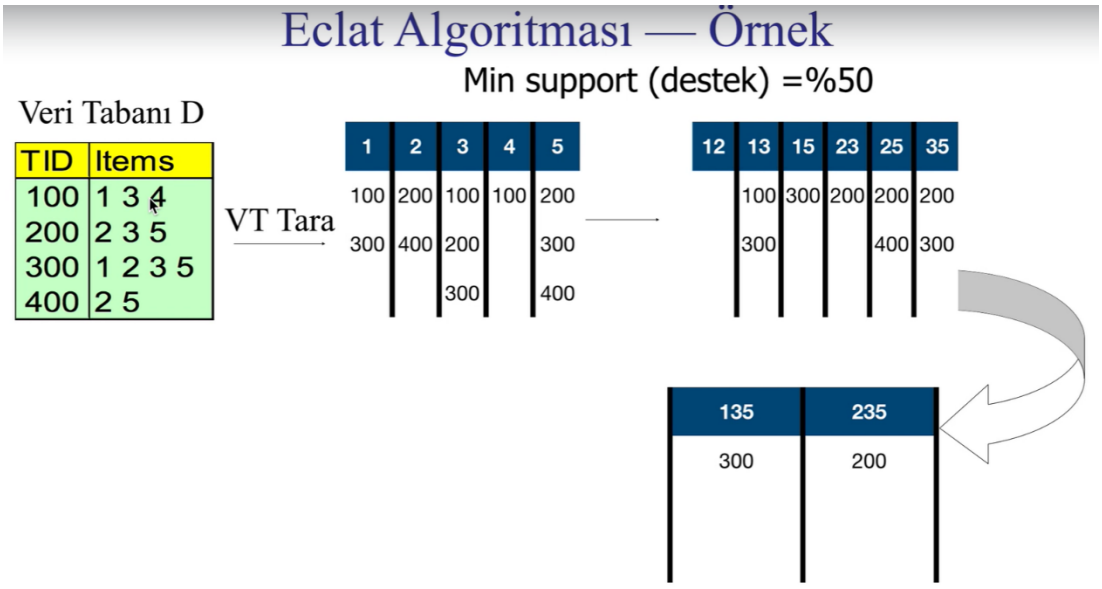
Yani artık 2,3,5 ‘ten birini alanlara kalanları da almak isteyip istemediğini soracağız. Çünkü bir çok müşteri bunu yapmış.

Sonuç olarak bu eylem alışveriş sepetinden çıkmış olsa da , şuan çok farklı alanlarda kullanılabilmekte. İşte A havayolu ile B şehrine uçan insanların C kafesinde kahve içme sıklığı yüksek se o zaman A ve B yi yapanlara C yi önerebilirsin. Tabi arka arkaya gelen olaylar için yön var yani C’yi yapana A ve B yi öneremezsin saçma olur. Bu concepte yönlendirilmiş ARM deniyor.

ECLAT ALGORİTMASI

Aslında Apriori genelde yeterli oluyor ve Eclat’a göre big data ile daha iyi çalışıyor ama farklı yönlerini görelim diye ECLAT dan da bahsedeceğiz.

ECLAT: Equivalance Class Transformation sözcüklerinden gelir.



Veritabanında yine farklı eventler tutuluyor bunlar ürün olabilir veya tıklamalar olabilir. Web sitesine girince tıklanılan ürünler mesela. Yeni gelen müşteriye 1 – 2 tıklamadan sonra diğer tıklamaları tavsiye etmek ARM içindeydi.

Eclat daha hızlı çalışır.

Eclatta yukarıda görüldüğü gibi her ürünün hangi transtactionslarda geçtiğini yazıyoruz. Mesela 1 item’ı 100 ve 300 nolu transactionslarda var gibi.

Bu alternatif yapılanma 1 ve 2 itemları hazırlandıktan sonra kalanlara bakmadan 12'ye bakmamıza olanak sağlıyor Apriori de bu mümkün değildi. Önce tüm tekillerin bitmesi gerekiyorki ikili kombinasyonlara geçeriz.

Min support yine elenebilir. Sonuçta 135 ve 235 ürünleri çıkıyor.

Sonuçta Apriori için itemları odaklanırken burada transcationslara odaklanıyoruz.