جواب تمرین‌های فصل ششم کتاب Han

امیر شبانی – ۹۴۱۳۰۸۸۰۲۱

۱. طبق تعریف (ویراست سوم، صفحه‌ی ۲۴۷) در کتاب Han، مجموعه‌ی آیتمی (همچون X) را در مجموعه‌ی داده (همچون D) **بسته** می‌نامیم، که هیچ مجموعه‌ی بزرگ‌تر (که X زیرمجموعه‌ی آن باشد) وجود نداشته‌باشد که مقدار support آن برابر X باشد. پس اگر بدانیم مجموعه‌ای **پرتکرار** و **بسته** است، می‌توانیم بگوییم تمام زیرمجموعه‌های آن نیز پرتکرار هستند. یا به عبارت دیگر، با در اختیار داشتن همه‌ی مجموعه‌ی آیتم‌های پرتکرار و بسته‌ی یک مجموعه‌ی داده، برای اینکه پرتکرار بودن یا نبودن یک مجموعه‌ی آیتم دلخواه (همچون X) را بررسی کنیم، کافیست ببینیم آیا X، زیرمجموعه‌ی حداقل یکی از مجموعه آیتم‌های پرتکرار و بسته هست یا خیر؛ اگر بود، پس X نیز پرتکرار است، در غیر این صورت، پرتکرار نیست. برای بدست آوردن مقدار support برای مجموعه‌ی آیتم X نیز باید بزرگ‌ترین مقدار support از بین مجموعه‌های پرتکرار و بسته در D که X زیرمجموعه‌ی آن‌ها هست را در نظر بگیریم. همین الگوریتم را به زبان پایتون پیاده‌سازی کردم و در گیت‌هاب آپلود کردم.

۳.
قسمت a) می‌دانیم مقدار support برای زیرمجموعه‌های یک مجموعه‌ی آیتم، از خود آن مجموعه بزرگ‌تر یا مساوی آن است. پس اگر یک مجموعه‌ی آیتم پرتکرار بود، یعنی مقدار support برای آن از min_sup بزرگ‌تر است، پس مقدار support برای زیرمجموعه‌های ناتهی آن نیز از min_sup بزرگ‌تر یا مساوی خواهد بود. پس زیرمجموعه‌های ناتهی یک مجموعه‌ی آیتم پرتکرار، هر کدام پرتکرار خواهند بود.

قسمت b) این مسئله با توجه به قسمت a بدیهی می‌باشد؛ زیرا اگر مجموعه‌ای، زیر مجموعه‌ی یک مجموعه آیتم پرتکرار باشد، مقدار support برای آن از خود مجموعه بزرگ‌تر یا مساوی آن است. پس اگر مجموعه‌ای از آیتم‌ها، شامل مجموعه‌ای دیگر باشد، مقدار support برای آن، کمتر یا مساوی خواهد بود.

قسمت c) اگر l را مجموعه‌ای پرتکرار در نظر بگیریم، در حالی که s زیرمجموعه‌ای از آن و s' نیز زیرمجموعه‌ای از s باشد، می‌توانیم بگوییم که مقدار support برای s کوچک‌تر یا مساوی همین مقدار برای s' می‌باشد. پس با توجه به تعریف confidence و توجه به این نکته که می‌توان مجموعه‌ی l را به مجموعه‌های s و s-l (یا s' و s’-l) افراض کرد به گونه‌ای که اجتماع آن‌ها برابر l باشد و با یکدیگر اشتراکی نداشته‌باشند، می‌توان نتیجه گرفت که مقدار confidence برای قانون اول کوچک‌تر یا مساوی همین مقدار برای قانون دوم است.

۶. قسمت a) به کمک الگوریتم Apriori، مجموعه‌های آیتم پرتکرار را این گونه به دست آوریم:
در مرحله‌ی اول دیتابیس را اسکن می‌کنیم و مقدار support را برای مجموعه‌های آیتم یک‌عضوی به دست می‌آوریم.

C1	
Itemset	Support count
{A}	1
{C}	2
{D}	1
{E}	4
{I}	1
{K}	5
{M}	3
{N}	2
{O}	3
{Y}	3

سپس آن دسته از مجموعه آیتم‌ها که مقدار supportشان از min_sup یعنی ۶۰ درصد کمتر هست را حذف می‌کنیم و به مجموعه‌ی L1 می‌رسیم:

L1	
Itemset	Support count
{K}	5
{M}	3
{E}	4
{O}	3
{Y}	3

سپس زیرمجموعه‌های دوعضوی قابل استخراج از L1 و مقدار supportشان را به دست می‌آوریم و همین فرآیند را تکرار می‌کنیم تا جایی که دیگر نتوانیم زیرمجموعه به دست آوریم.

C2	
Itemset	Support count
{E, K}	1
{E, M}	2
{E, O}	1
{E, Y}	4
{K, M}	1
{K, O}	5
{K, Y}	3
{M, O}	2
{M, Y}	3
{O, Y}	3

L2	
Itemset	Support count
{E, K}	1
{E, O}	1
{K, M}	1
{K, O}	5
{K, Y}	3

C3	
Itemset	Support count
{E, K, O}	3

L3	
Itemset	Support count
{E, K, O}	3

پس می‌توان گفت مجموعه‌های پرتکرار عبارت‌اند از: {E, K, M, O, Y, EK, EO, KM, KO, KY, EKO}

قسمت b) قوانین قوی:

{E, O} → K [support = 60%, confidence = 100%]
{K, O} → E [support = 60%, confidence = 100%]

۱۰. برای این کار می‌توانیم از روش Partitioning برای بهبود الگوریتم Apriori استفاده کنیم. به این گونه که مجموعه‌های آیتم پرتکرار را در یک قسمت جداگانه و داده‌هایی که جدید اضافه می‌شوند را در قسمتی دیگر ذخیره کنیم و پس از آن قوانین لازم را به دست می‌آوریم.

.۱۴
قسمت a) آری، قانون hotdogs → hamburgers یک قانون قوی به حساب می‌آید. زیرا مقدار support برای hotdog برابر ۰.۶ و برای hamburger برابر ۰.۵ می‌باشد. پس مقدار support هر کدام از آن‌ها از مقدار min_sup بزرگ‌تر است. حال باید مقدار confidence این قانون را به دست آوریم. مقدار support برای رخداد hotdog و hamburger (یعنی رخداد اجتماع آن‌ها) برابر ۰.۴ می‌باشد. اگر این عدد را به مقدار support قسمت اول قانون یعنی hotdogs تقسیم کنیم به مقدار ۲/۳ می‌رسیم که از مقدار min_conf یعنی ۵۰ درصد بیشتر است. پس این قانون را می‌توان قوی به حساب آورد.

قسمت b) خیر، خرید hotdog، مستقل از خرید hamburger نیست، بلکه رابطه‌ی بین آن‌ها از نوع همبستگی مثبت است.