پروپوزال

۱– مقدمه :

در این پروژه قصد پیاده سازی برنامه ای به منظور استخراج قواعد انجمنی (association rule mining) از دیتاست های کاربران را داریم. برنامه فوق به زبان پایتون و با استفاده از پلتفرم Qt Designer به منظور طراحی رابط کاربری گرافیکی ساده جهت سهولت در استفاده از برنامه برای کاربران پیاده سازی شده است. مجموعه داده استفاده شده در پروژه Market_Basket_Optimisation است. الگوریتم های استفاده شده در برنامه به همراه شبه کد های آنها درادامه به اختصار توضیح داده شده اند. کاربر به دلخواه خود و باتوجه به مجموعه داده های خود و شرایط آن امکان استفاده از ۳ الگوریتم Apriori و FP-growth و Eclat را به همراه پارامتر های مورد نظرمثل Support و Confidence و سایر آرگومان های ورودی الگوریتم های موجود را دارد. خروجی برنامه شامل یک فایل متنی txt. حاوی قواعد انجمنی کشف شده توسط الگوریتم های موجود است که میتواند به عنوان پایه‌هایی برای تصمیم‌سازی درباره فعالیت‌هایی مانند تحلیل سبد خرید مورد استفاده قرارگیرد.

۲– کاربردها :

استخراج قوانین انجمنی بیشتر برای تصمیم گیری استفاده می شود. قوانین انجمنی را میتوان برای یافتن الگوها در بسیاری از داده ها استفاده می کرد. قوانین انجمنی در کسب و کار به تصمیم گیری در زمینه بازاریابی و سایر زمینه ها کمک می کند. می توان از آن برای بهبود تصمیم گیری در طیف گسترده ای از برنامه ها مانند: تشخیص پزشکی ، GIS ، پایگاه داده رابطه ای و پایگاه داده توزیع شده و غیره استفاده کرد.

در پایگاه داده های بزرگ مثال هایی از کاربرد قوانین انجمنی در تجزیه و تحلیل سبد بازار به شرح زیر است:

- تجزیه و تحلیل نقطه معامله فروش.
- از اطلاعات مربوط به آنچه مشتریان خرید می کنند استفاده می کند تا بینش خود را در مورد اینکه چه کسانی هستند و چرا خریدهای خاصی انجام می دهند ارائه دهد.
- از کدام محصولات با هم خریداری می شود و کدامیک بیشتر مایل به پشتیبانی هستند.

۳– توضیحات :

اغلب الگوریتم‌های یادگیری ماشین در داده‌کاوی با داده‌های عددی کار می‌کنند و در پیاده‌سازی و نحوه کار آن‌ها گرایش به ریاضیات محض وجود دارد. اما، کاوش قواعد وابستگی (association rule mining) که از آن با عنوان

«کاوش قواعد وابستگی» نیز یاد می‌شود، برای داده‌های دسته‌ای مناسب و محاسبات آن نسبت به بسیاری از دیگر الگوریتم‌ها ساده‌تر است. این روش، یکی از راهکارهای مبتنی بر قواعد(rules) ، برای کشف روابط جالب بین متغیرها در پایگاه داده‌های بزرگ محسوب می‌شود. در کاوش قواعد وابستگی، قواعد قوی با استفاده از سنجه جذابیت (interestingness) شناسایی می‌شوند.

به عنوان مثال، قانون وابستگی {onions, potato} ⟶ {burger}

در داده‌های فروش یک سوپرمارکت، نشان می‌دهد در صورتی که یک مشتری پیاز (onions) و سیب زمینی (potatoes)را در سبد خرید خود قرار داده است، احتمالاً او مایل به خرید گوشت همبرگر نیز خواهد بود. چنین اطلاعاتی می‌تواند به عنوان مبنای تصمیماتی برای برخی از فعالیت‌های فروشگاهی همچون ارائه مناسب تخفیف برای محصولات یا قراردادن مناسب محصولات در کنار هم، مورد استفاده قرار گیرد. علاوه بر مثال فوق که در مورد تحلیل سبد خرید مطرح شد، امروزه یادگیری قانون وابستگی در کاربردهای متفاوت همچون مصرف کاوی وب، تشخیص نفوذ، و بیوانفورماتیک مورد استفاده قرار می‌گیرد.

برای انتخاب قوانین جذاب از بین مجموعه قوانین ممکن، محدودیت‌های مختلف روی معیارهای سنجش اهمیت و جذابیت به کار می‌رود. معروف‌ترین محدودیت‌ها شامل آستانه کمینه برای پشتیبان و اطمینان است.

پشتیبان : نشانگر آن است که یک مجموعه اقلام چند بار در پایگاه تکرار شده. این مقدار به عنوان کسر رکوردهای شامل XUY بر کل تعداد رکوردها در پایگاه داده است. برای مثال اگر پشتیبان یک مورد ۰٫۱% باشد، بدین معناست که تنها ۰٫۱% از تراکنش‌ها حاوی آن مورد هستند.

Support (XY) = Support count of (XY) / Total number of transactions in D

اطمینان :کسر تعداد تراکنش‌های حاوی XUY به کل تعداد رکوردهای شامل x است. این مقدار، مقیاسی از استحکام قواعد وابستگی است. برای مثال اگر اطمینان یک قاعده وابستگی X⇒Y is 80% باشد، بدین معناست که ٪۸۰ از تراکنش‌هایی که حاوی X هستند، شامل Y نیز می‌شوند.

Confidence (X|Y) = Support (XY) / Support (X)

۴– الگوریتم ها :

در طول زمان، الگوریتم‌های متعددی برای تولید قوانین وابستگی پیشنهاد شده‌اند.بعضی الگوریتم‌های معروف در این زمینه عبارتند از: آپریوری(Apriori)، اکلات (Eclat) و FP-Growth. تمامی این الگوریتم‌ها تنها انجام دهنده نیمی از مسیر تولید قوانین وابستگی هستند. چرا که این الگوریتم‌ها برای کاوش مجموعه آیتم‌های مکرر (Frequent item-set mining) ساخته شده‌اند و پروسه دیگری روی مجموعه آیتم‌های مکرر باید انجام شود تا منتهی به قوانین وابستگی شوند.

۱-۴ الگوریتم Apriori :

الگوریتم اپریوری(Apriori) ، روشی قابل اعمال روی رکوردهای پایگاه داده و به ویژه پایگاه داده تراکنشی یا رکوردهای حاوی تعداد مشخصی فیلد یا آیتم است. اپریوری یکی از الگوریتم‌های دارای رویکرد «پایین به بالا» است که به تدریج رکوردهای پیچیده را با یکدیگر مقایسه می‌کنند. این الگوریتم یکی از روش‌های کارآمد برای حل مسائل پیچیده کنونی موجود در داده‌کاوی و یادگیری ماشین است.اساسا، الگوریتم اپریوری بخش‌هایی از یک پایگاه داده بزرگ‌تر را دریافت کرده و به آن‌ها «امتیازدهی» کرده و یا آن بخش‌ها را با دیگر مجموعه‌ها به شیوه مرتب شده‌ای مقایسه می‌کند. از نتایج خروجی، برای تولید مجموعه‌هایی استفاده می‌شود که مکررا در پایگاه داده اصلی به وقوع پیوسته‌اند.

Algorithm 3: Apriori algorithm

F_1 = {frequent items of size 1};
for (k = 1; F_k !=ϕ; k++) do begin
  C_{k+1} = apriori-gen (F_k); // New candidates generated from F_k
  for all transactions t in database do begin
    C'_t = subset (C_{k+1},t); // Candidates contained in t
    for all candidate $c \in C'_t$ do
      $c.count$++; // Increment the count of all candidates
      in C_{k+1} that are contained in t
    end
    F_{k+1} = {$C \in C_{k+1} | c.count \geq$ minimum suport}
    //Candidates in C_{k+1} with minimum support
  end
end
Answer $\bigcup_k F_k$;

۲-۴ الگوریتم Fp-Growth :

حجم داده ورودی FP-growth اصولا بسیار زیاد است روش جالبی که مجموعه اقلام پرتکرار را بدون تولید مجموعه اقلام کاندید به دست می‌آورد، الگوریتم FP-Growth است که از یک استراتژی تقسیم وحل استفاده می‌کند. این روش پایگاه داده را به مجموعه ای از پایگاه‌داده ها که هرکدام یک قلم پرتکرار دارند، تقسیم می‌کند و هر پایگاه داده را جداگانه کاوش می‌کند.

در اوّلین اسکن پایگاه داده همانند اپریوری مجموعه آیتم‌های یک عضوی و پشتیبانی آنها مشخص می‌شود. مجموعه اقلام پرتکرار به ترتیب نزولی پشتیبانی‌شان مرتّب می‌شوند.

سپس یک درخت به این صورت ساخته می‌شود که: اوّل ریشه درخت با برچسب null ساخته می‌شود. بعد از آن پایگاه داده برای بار دوم اسکن می‌شود. اقلام هر تراکنش به ترتیب L پردازش می‌شوند و یک شاخه برای هر تراکنش ایجاد می‌شود. به منظور تسهیل پیمایش درخت، یک جدول ساخته می‌شود که هر قلم درآن به محلّ خودش در درخت اشاره می‌کند. درخت پس از اسکن همه تراکنش ها کامل می‌شود.

Procedure *FPgrowth*(T)*
Input: A conditional FP-tree T
Output: The complete set of all FI's corresponding to T.
Method:
1. **if** T only contains a single branch B
2. **for each** subset Y of the set of items in B
3. output itemset $Y \cup T.base$ with count = smallest count of nodes in Y;
4. **else for each** i in $T.header$ **do begin**
5. output $Y = T.base \cup \{i\}$ with $i.count$;
6. **if** $T.FP\text{-}array$ is defined
7. construct a new header table for Y's FP-tree from $T.FP\text{-}array$
8. **else** construct a new header table from T;
9. construct Y's conditional FP-tree T_Y and possibly its FP-array A_Y;
10. **if** $T_Y \neq \emptyset$
11. call *FPgrowth*(T_Y)*;
12. **end**

۳-۴ الگوریتم Eclat:

الگوریتم Eclat به عنوان خوشه بندی کلاس Equivalence و الگوریتم عرضی شبکه از پایین به بالا خلاصه می شود. این یک الگوریتم برای پیدا کردن مجموعه موارد مکرر در یک معامله یا پایگاه داده است. این یکی از بهترین روش های یادگیری قانون انجمن است. این بدان معناست که الگوریتم Eclat برای ایجاد مجموعه های مکرر موارد در پایگاه داده استفاده می شود. الگوریتم Eclat از اولین جستجوی عمق اول برای کشف مجموعه های مکرر موارد استفاده می کند ، در حالی که الگوریتم Apriori از جستجوی اول سطح استفاده می کند. برخلاف الگوریتم Apriori که داده ها را به صورت افقی نشان می دهد ، داده ها را به صورت عمودی نشان می دهد. این الگوی عمودی الگوریتم Eclat در مقایسه با الگوریتم Apriori به الگوریتم سریع تری تبدیل می شود. از این رو ، الگوریتم Eclat نسخه کارآمدتر و مقیاس پذیرتر از یادگیری قانون انجمن است.

TABLE I. PSEUDO CODE FOR ECLAT ALGORITHM

Input: dataset, support_threshold
Output: all frequent itemsets

```
1.  freq_set = build_invert_list(dataset)
2.  item_num  = 2
3.  all_freq_set.append(freq_set)
4.  while (len(freq_set) > 0):
5.      cand_set = []
6.      num_freq = 0
7.      for freq in freq_set:
8.          for i in range(0, len(freq)):
9.              cand = combine_list(i, freq, support_threshold)
```

علی سید مومنی

سباقنبری

نوید افضلی

فخرالسادات میرشریفی