بسمه‌تعالی

دانشگاه صنعتی اصفهان

دانشکده برق و کامپیوتر

کاربرد نظریه فازی در داده‌کاوی

گزارش درس سیستم های کنترل فازی

مدرس:

آقای دکتر فرید شیخ‌الاسلام

گردآورندگان:

نرجس خاتون حبیبی

زینب زالی

بهار ۸۶

فهرست

۱- مقدمه ۳

۲- داده‌کاوي چيست؟ ٤

۲-۱- سابقه داده کاوي ٤

۲-۲- مفهوم داده کاوي ٥

۲-۳- فرآيند داده‌کاوي ٦

۲-۴- مثالي از داده‌کاوي ۷

۲-٥- تفاوت داده کاوی و آنالیز های آماری ۸

۲-۶- داده‌کاوي و مديريت دانش ۹

۲-۷- روش‌هاي داده‌کاوي ۹

۳- قوانين وابستگي ۱۱

۳-۱- تعاريف ۱۲

۳-۲- جزئيات روش ۱۳

۳-۲-۱- Apriori-Gen ۱۳

۳-۲-۲- Apriori ۱٤

۳-۲-۳- AR-Gen ۱٤

۳-۳- یک روش داده کاوی فازی مبتنی بر الگوریتم Apriori، با زمان محاسباتی کاهش یافته ۱٥

۳-۳-۱- الگوریتم پیشنهادی به همراه یک مثال ۱٥

۴- درخت تصمیم‌گیری ۲٤

۴-۱- درخت تصمیم فازی و کلاس‌بندی فازی ۲٥

۴-۲ - ساخت درخت تصمیم از داده‌های پیوسته ۲٦

۴-۳ - ساخت درخت تصمیم از داده‌های پیوسته با جداسازی نرم ۲۷

۵- نتیجه گیری ۳۱

۶- مراجع ۳۲

۱- مقدمه

از هنگامی که رایانه در تحلیل و ذخیره سازی داده ها بکار رفت (۱۹۵۰) پس از حدود ۲۰ سال، حجم داده ها در پایگاه داده ها دو برابر شد. ولی پس از گذشت دو دهه و همزمان با پیشرفت فن آوری اطلاعات (IT) هر دو سال یکبار حجم داده ها، دو برابر شده است. نیز تعداد پایگاه داده ها با سرعت بیشتری رشد نمود. این در حالی است که تعداد متخصصین تحلیل داده ها و آمارشناسان با این سرعت رشد نکرد. حتی اگر چنین امری اتفاق می افتاد، بسیاری از پایگاه داده ها چنان گسترش یافته اند که شامل چندصد میلیون یا چندصد میلیارد رکورد ثبت شده هستند و امکان تحلیل و استخراج اطلاعات با روش های معمول آماری از دل انبوه داده ها مستلزم چند روز کار با رایانه های موجود است. حال با وجود سیستم های یکپارچه اطلاعاتی، سیستم های یکپارچه بانکی و تجارت الکترونیک، لحظه به لحظه به حجم داده ها در پایگاه داده های مربوط اضافه شده و باعث به وجود آمدن انبارهای عظیمی از داده ها شده است به طوری که ضرورت کشف و استخراج سریع و دقیق دانش از این پایگاه داده ها را بیش از پیش نمایان کرده است (چنان که در عصر حاضر گفته می شود « اطلاعات طلاست»).

هم اکنون در هر کشور، سازمان ها و شرکت ها برای امور بازرگانی، پرسنلی، آموزشی و آماری پایگاه داده ها ایجاد یا خریداری شده است، به طوری که این پایگاه داده ها برای مدیران، برنامه ریزان و پژوهشگران جهت تصمیم گیری های راهبردی، تهیه گزارش های مختلف و توصیف وضعیت جاری خود می تواند مفید باشد. داده کاوی[1] یا استخراج و کشف سریع و دقیق اطلاعات با ارزش و پنهان از این پایگاه داده ها از جمله اموری است که هر کشور، سازمان و شرکتی به منظور توسعه علمی، فنی و اقتصادی خود به آن نیاز دارد.

[1]Data Mining

۲- داده‌کاوی چیست؟

۲-۱- سابقه داده کاوی

داده کاوی و کشف دانش در پایگاه داده ها از جمله موضوع هایی هستند که هم زمان با ایجاد و استفاده از پایگاه داده ها در اوایل دهه ۸۰ برای جستجوی دانش در داده ها شکل گرفت.

داده کاوی پل ارتباطی میان علم آمار ، علم کامپیوتر ، هوش مصنوعی ، الگوشناسی ، فراگیری ماشین و بازنمایی بصری داده می باشد.

شاید بتوان لوول (۱۹۸۳) را اولین شخصی دانست که گزارشی در مورد داده کاوی تحت عنوان «شبیه سازی فعالیت داده کاوی» ارائه نمود. هم زمان با او پژوهشگران و متخصصان علوم رایانه، آمار، هوش مصنوعی و یادگیری ماشین نیز به پژوهش در این زمینه و زمینه های مرتبط با آن پرداخته اند.

پژوهش جدی روی موضوع داده کاوی از اوایل دهه ۹۰ شروع شد. پژوهش ها و مطالعه های زیادی در این زمینه صورت گرفته، همچنین سمینارها، دوره های آموزشی و کنفرانس هایی نیز برگزار شده است. نتایج پایه های نظری داده کاوی در تعدادی از مقاله های پژوهشی آورده شده است. مثلاً سال ۱۹۹۱ پیاتتسکی و شاپیرو [۱] « استقلال آماری قاعده ها در داده کاوی» را بررسی نموده اند. سال ۱۹۹۵ هافمن و نش استفاده از داده کاوی و داده انبار[۲] توسط بانک های آمریکا را بررسی نموده و بیان کردند که چگونه این سیستم ها برای بانک های آمریکا قدرت رقابت بیشتری ایجاد می کنند. چت فیلد مشکلات ایجاد شده توسط داده کاوی را بررسی نمود و همچنین مقاله ای تحت عنوان «مدل های خطی غیر دقیق داده کاوی و استنباط آماری» ارایه نمود. هندری نیز دیدگاه اقتصاد سنجی روی داده کاوی را تهیه کرد. در این سال انجمن داده کاوی هم زمان با اولین کنفرانس بین المللی «کشف دانش و داده کاوی» شروع به کار کرد. این کنفرانس توسعه یافته چهار دوره آموزشی بین المللی در پایگاه های داده درسال ۱۹۸۹ تا ۱۹۹۴ بود. انجمن مذکور، یک سازمان علمی به نام ACM- SIGKDD را ایجاد نمود. سال ۱۹۹۶ ایمیلنسکی[۳] و منیلا[۴] دیدگاهی از داده کاوی به عنوان «پرس و جو کننده از پایگاه های استنتاجی[۵]» را پیشنهاد کردند. فایاد، پیاتتسکی- شاپیرو، اودوراُسامی پیشرفت های کشف دانش و داده کاوی را عنوان کردند. در

---

[1] - piatetsky-shapiro
[2]- Data warehouse
[3]- Imielnski
[4] - Mannila
[5] - Inductive databases

سال ۱۹۹۷ منیلا خلاصه ای از مطالعه روی اساس داده کاوی ارایه نمود. باربارا و همکاران نیز دیدگاه کاهش داده ها روی داده کاوی را در گزارش کاهش داده های نیوجرسی ارایه نمودند. همچنین می توان برای کاربرد داده کاوی در مدیریت مالی می توان، تحلیل داده های مالی و مدل سازی مالی بنینگاه و چاچ کز و هیگینز [۱] را ملاحظه کرد. فریدمن نیز مقاله ای در ارتباط با مفهوم آمار و داده کاوی ارایه نمود. سال ۱۹۹۸ هند [۲] مقاله ای تحت عنوان « داده کاوی : آمار یا بیشتر؟» ارائه نمود. کلینبرگ[۳] پائودیمیتریو و راغان [۴] دیدگاه اقتصاد سنجی روی داده کاوی و عملکرد داده کاوی به عنوان یک مسئله بهینه را ارایه نمودند. در این سال نیز کنفرانس های ناحیه ای و بین المللی در مورد داده کاوی برگزار شد که از جمله می توان به کنفرانس آسیا و اقیانوسیه درباره کشف دانش و داده کاوی اشاره کرد. سال ۲۰۰۰ هند و همکاران و اسمیت بحث های مقایسه‌ای بین آمار و داده کاوی را ارایه کردند. سری و استاوا، کولی، رش پاند و تن استفاده از وب در کاوش داده ها و کاربردهای آن را ارایه کردند. سال ۲۰۰۲ کلادیو کانورسانو و همکاران « مدل آمیخته چندگانه جمع پذیر تعمیم یافته » برای داده کاوی را بررسی نمودند. پائلو و گیانلوکاپاسرون، « داده کاوی ساختارهای پیوند برای مدل رفتار مصرف کننده » را ارایه نمودند.

۲-۲- مفهوم داده کاوی

عبارت داده کاوی مترادف با یکی از عبارت های استخراج دانش، برداشت اطلاعات، وارسی داده ها و حتی لایروبی کردن داده هاست که در حقیقت کشف دانش در پایگاه داده ها[۵] (**KDD**) را توصیف می کند. بنابراین ایده ای که مبنای داده کاوی است یک فرآیند با اهمیت از شناخت الگوهای بالقوه مفید، تازه و درنهایت قابل درک در داده هاست. واژه کشف دانش در پایگاه داده ها در اوایل دهه ۸۰ در مراجعه به مفهوم کلی، گسترده، سطح بالا و به دنبال جستجوی دانش در اطلاعات شکل گرفته است. داده کاوی کاربرد سطح بالای فنون و ابزار بکار برده شده برای معرفی و تحلیل داده ها ی تصمیم گیرندگان است. اصطلاح داده کاوی را آمار شناسان، تحلیل گران داده ها و انجمن سیستم های اطلاعات مدیریت به کار برده اند در حالی که پژوهشگران یادگیری ماشین و هوش مصنوعی از **KDD** بیشتر استفاده می کنند. در ادامه چند تعریف از داده کاوی ارائه می شود.

---

[1] - Benninga, Czaczkes, Higgins
[2] - Hand
[3] - Kleinberg
[4] - Paodimitriou , Raghavan
[5] - Knowledge Discovery of Database

- «داده کاوی یا به تعبیر دیگر کشف دانش در پایگاه داده ها، استخراج غیر بدیهی اطلاعات بالقوه مفید از روی داده هایی است که قبلاً، ناشناخته مانده اند. این مطلب برخی از روش های فنی مانند خوشه بندی، خلاصه سازی داده ها، فراگیری قاعده های رده بندی، یافتن ارتباط شبکه ها، تحلیل تغییرات و کشف بی قاعدگی را شامل می شود » (پیاتتسکی شاپیرو، ماتئوس کریستوفر)
- «داده کاوی در حقیقت کشف ساختارهای جالب توجه، غیر منتظره و با ارزش از داخل مجموعه وسیعی از داده ها می باشد و فعالیتی است که اساساً با آمار و تحلیل دقیق داده ها منطبق است» هند (۱۹۹۸)
- «داده کاوی فرآیند کشف رابطه ها، الگوها و روندهای جدید معنی داری است که به بررسی حجم وسیعی از اطلاعات ذخیره شده در انبارهای داده با فناوری های تشخیص الگو (مانند ریاضی و آمار) می پردازد». (سایت http://www.spss.com)

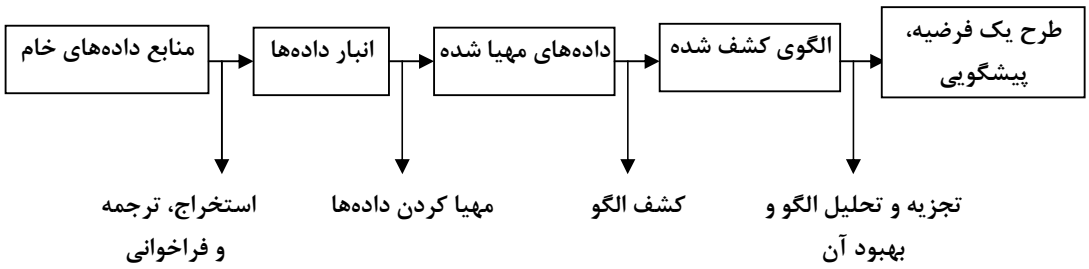
کشف دانش در پایگاه داده ها در جهت کشف اطلاعات مفید از مجموعه بزرگ داده هاست. دانش کشف شده می تواند قاعده ای باشد تا ویژگی های داده ها، الگوهایی که به طور متناسب رخ می دهند، خوشه بندی موضوع های درون پایگاه داده ها و غیره را توصیف می کند.

یک کاربر سیستم **KDD** بایستی درک بالایی از قلمرو داده ها به منظور انتخاب زیر مجموعه صحیحی از داده ها، رده مناسبی از الگوها و معیار خوبی برای الگوهای جالب داشته باشد. بنابراین سیستم **KDD** باید ابزارهایی با اثر تعاملی داشته باشد نه سیستم های تجزیه و تحلیل خودکار. لذا کشف دانش از پایگاه داده ها باید مثل یک فرآیند شامل گام های زیر باشد:

۱. درک قلمرو
۲. آماده کردن مجموعه داده ها
۳. کشف الگوها (داده کاوی)
۴. پردازش بعد از کشف الگو
۵. استفاده از نتایج

۲-۳- فرآیند داده‌کاوی

می‌توان فرآیند داده‌کاوی را طی مراحل زیر به صورت نمودار نشان داد.



شکل ۱) فرآیند داده‌کاوی

در فرآیند بالا، داده‌های خام از منابع مختلفی جمع‌آوری می‌شوند و از طریق استخراج، ترجمه و فرآیندهای بازخوانی به انبار داده‌ها وارد می‌شوند. در بخش مهیاسازی داده‌ها، داده‌ها از انبار خارج شده و به صورت یک فرمت مناسب برای داده‌کاوی درمی‌آیند. در بخش کشف الگو با روش‌های داده‌کاوی برای پاسخ به سؤال‌های خاصی که به ذهن می‌رسند، الگوریتم‌هایی را استخراج می‌کنند و از این الگوریتم‌ها برای ساخت الگو استفاده می‌شود. در بخش تجزیه و تحلیل الگو، الگوها به یک دانش مفید و قابل استفاده تبدیل می‌شوند و پس از بهبود آن‌ها، الگوهایی که کارا محسوب می‌شوند در یک سیستم اجرایی به کار گرفته خواهند شد.

۲-۴- مثالی از داده‌کاوی

یکی از نمونه های بارز داده کاوی را می توان در فروشگاه های زنجیره ای مشاهده نمود، که در آن سعی می شود ارتباط محصولات مختلف هنگام خرید مشتریان مشخص گردد. فروشگاه های زنجیره ای مشتاقند بدانند که چه محصولاتی با یکدیگر به فروش می روند.

برای مثال طی یک عملیات داده کاوی گسترده در یک فروشگاه زنجیره ای در آمریکای شمالی که بر روی حجم عظیمی از داده های فروش صورت گرفت، مشخص گردید که مردانی که برای خرید قنداق بچه به فروشگاه می روند معمولا آب جو نیز خریداری می کنند. همچنین مشخص گردید مشتریانی که تلویزیون خریداری می کنند، غالبا گلدان کریستالی نیز می خرند. نمونه مشابه عملیات داده کاوی را می توان در یک شرکت بزرگ تولید و عرضه پوشاک در اروپا مشاهده نمود، به شکلی که نتایج داده کاوی مشخص می کرد که افرادی که کراوات های ابریشمی خریداری می کنند، در همان روز یا روزهای آینده گیره کراوات مشکی رنگ نیز خریداری می کنند.

به روشنی این مطلب قابل درک است که این نوع استفاده از داده کاوی می تواند فروشگاه ها را در برگزاری هوشمندانه فستیوال های فروش و نحوه ارائه اجناس به مشتریان یاری رساند.

نمونه دیگر استفاده از داده کاوی در زمینه فروش را می توان در یک شرکت بزرگ دوبلاژ و تکثیر و عرضه فیلم های سینمایی در آمریکای شمالی مشاهده نمود که در آن عملیات داده کاوی، روابط مشتریان و هنرپیشه های سینمایی و نیز گروه های مختلف مشتریان بر اساس سبک فیلم ها (ترسناک، رمانتیک، حادثه ای) مشخص گردید. بنابراین آن شرکت به صورت کاملا هوشمندانه می توانست مشتریان بالقوه فیلم های سینمایی را بر اساس علاقه مشتریان به هنرپیشه های مختلف و سبک های سینمایی شناسایی کند.

از دیگر زمینه های به کارگیری داده کاوی، استفاده بیمارستانها و کارخانه های داروسازی جهت کشف الگوها و مدلهای ناشناخته تاثیر دارو ها بر بیماری های مختلف و نیز بیماران گروه های سنی مختلف را می توان نام برد.

استفاده از داده کاوی در زمینه های مالی و بانکداری به شناخت مشتریان پر خطر و سودجو بر اساس معیار هایی از جمله سن ، درآمد، وضعیت سکونت، تحصیلات، شغل و غیره می انجامد.

۲-۵- تفاوت داده کاوی و آنالیز های آماری

داده کاوی با آنالیز های متداول آماری متفاوت است. در زیرمی توان برخی از اصلی ترین تفاوت های داده کاوی و آنالیز آماری را مشاهده نمود:

- آنالیز آماری:

- آمار شناسان همیشه با یک فرضیه شروع به کار می کنند.
- آن ها از داده های عددی استفاده می کنند.
- آمارشناسان باید رابطه هایی را ایجاد کنند که به فرضیه آنها مربوط است.
- آن ها می توانند داده های نابجا و نادرست را در طول آنالیز مشخص کنند.
- آن ها می توانند نتایج کار خود را تفسیر و برای مدیران بیان کنند.

- داده کاوی:

- به فرضیه احتیاجی ندارد.
- ابزارهای داده کاوی از انواع مختلف داده ، نه تنها عددی می توانند استفاده کنند.

- الگوریتمهای داده کاوی به طور اتوماتیک روابط را ایجاد می کنند.
- داده کاوی به داده های صحیح و درست نیاز دارد.
- نتایج داده کاوی نسبتا پیچیده می باشد و نیاز به متخصصانی جهت بیان آنها به مدیران دارد.

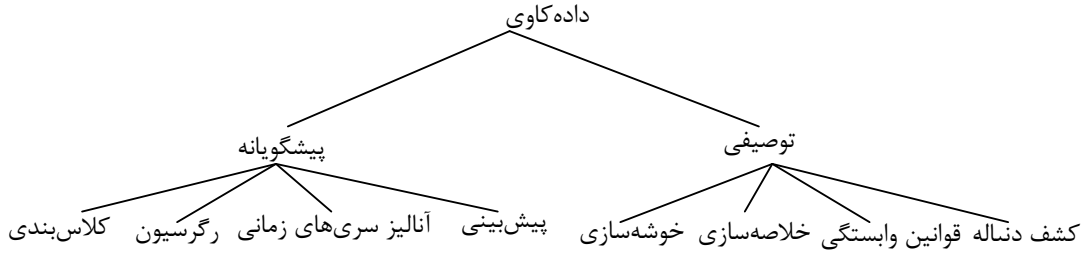
۲-۶- داده‌کاوی و مدیریت دانش

اگر چه دانش به طور انحصاری محصول فناوری اطلاعات نیست، ولی فناوری اطلاعات به طور لاینفکی در ایجاد دانش و فرآیند مدیریت دانش از سال های اول مشارکت داشته است. امروزه مدیریت دانش از مسئولیت های فناوری اطلاعات به شمار می‌رود. زیرا در جمع‌آوری، تبدیل دانش و انتقال داده‌ها، اطلاعات و دانش نقش کلیدی دارد.

از منظر مدیریت دانش، هدف داده‌کاوی، کشف دانش سازمانی پنهان در اطلاعات خام است. اینگونه نیست که هر بینش حاصل از داده‌کاوی دانش می‌سازد، بلکه در عوض بسیاری از نتایج به دست آمده، اطلاعات مدیریت، یا هوش سازمانی است. مثلاً در سازمان‌های تجاری، دانش با ارزش مورد مشتری، محصول و بازار را می‌توان از طریق داده‌کاوی به دست آورد. داده‌کاوی ابزار مفیدی برای مدیران دانش است که کشف را با تحلیل تلفیق می‌کنند. تلفیقی که اغلب منجر به ایجاد دانش می‌شود.

۲-۷- روش‌های داده‌کاوی

در این قسمت بعضی عملیات که در روند داده‌کاوی معمولا استفاده می‌شود ذکر خواهیم کرد. شکل ۲ به طور خلاصه مدل‌ها و اعمال متفاوت داده‌کاوی را نشان می‌دهد.



شکل ۲. مدل‌ها و عملیات داده‌کاوی

از مهم‌ترین و معمول‌ترین عملیات داده‌کاوی که در شکل ۲ مشاهده می‌شود، کلاس‌بندی، خوشه‌سازی و قوانین وابستگی است. کلاس‌بندی داده‌ها را به گروه‌های از پیش‌تعریف‌شده متناظر می‌کند. در خوشه‌سازی نیز داده‌ها در گروه‌هایی دسته‌بندی می‌شوند، با این تفاوت که گروه‌ها و

تعریف آن‌ها از پیش مشخص نشده است. قوانین وابستگی به منظور نشان دادن وابستگی میان آیتم های داده به کار می روند. در این تحقیق سعی داریم به شرح بعضی روش‌های فازی به‌کار رفته در کلاس‌بندی و قوانین وابستگی بپردازیم. در قسمت‌های بعد ابتدا روشی معروف و مهم در به‌دست‌آوردن قوانین وابستگی را مطرح خواهیم کرد و سپس چگونگی به کاربردن منطق و استدلال فازی در این روش را شرح می‌دهیم. پس از آن به بحث درخت‌های تصمیم‌گیری می‌پردازیم که از مهمترین روش‌ها در کلاس‌بندی داده‌هاست و روش‌های به‌کارگیری فازی در ساخت درخت‌های تصمیم‌گیری را بیان خواهیم کرد.

۳- قوانین وابستگی[1]

قوانین وابستگی به منظور نشان دادن وابستگی میان آیتم های داده به کار می روند. خرید یک جنس هنگامی که جنس دیگری خریداری شده است، نشان دهنده یک قانون وابستگی است. قوانین وابستگی به وفور در فروشگاه ها و در زمینه بازاریابی، تبلیغ، چیدمان اجناس و کنترل موجودی استفاده می شوند. این قوانین در زمینه های دیگر نیز مانند پیش بینی خطا[2] در شبکه های مخابراتی به کار گرفته می شوند. روابط کشف شده، به طور ذاتی میان آیتم ها وجود ندارد (مانند وابستگی های تابعی[3])، بلکه این قوانین استفاده رایج[4] از آیتم ها را نشان می دهند. یک مثال ساده به صورت زیر است:

یک فروشگاه مواد غذایی، تراکنش های خرید مشتریان را نگهداری می کند. هر تراکنش آیتم هایی که یک مشتری، در زمان خاص، خریداری نموده است را نشان می دهد. مدیریت فروشگاه می خواهد بداند معمولا چه آیتم هایی با یکدیگر خریداری می شوند. داده کاوی تراکنش ها نشان می دهد که در ٪۱۰۰ مواردی که کره بادام زمینی فروخته شده، نان نیز فروخته شده، و در ٪۳۳ موارد کره بادام زمینی به همراه مربا خریداری شده است.

پایگاه داده ای که قوانین وابستگی به آن اعمال می شود، به صورت مجموعه ای از رکوردهاست که هر رکورد شامل تعدادی آیتم است. جدول ۱ یک پایگاه داده را برای مثال فروشگاه مواد غذایی نشان می دهد.

---

[1] Association Rules
[2] Fault
[3] Functional Dependencies
[4] Common Usage

جدول ۱) مثالی از پایگاه داده تراکنش های یک فروشگاه

Transaction	Items
t_1	Bread, Jelly, Peanut Butter
t_2	Bread, Peanut Butter
t_3	Bread, Milk, Peanut Butter
t_4	Beer, Bread
t_5	Beer, Milk

۳-۱- تعاریف

تعدادی تعریف پایه ای در رابطه با قوانین وابستگی وجود دارند که در این بخش ارائه می شوند.

با داشتن مجموعه ای از **آیتم های** $I=\{I_1,...,I_m\}$ و یک پایگاه داده از تراکنش ها $D=\{t_1,...,t_n\}$ که $t_i=\{I_{i1},...,I_{ik}\}$ و $I_{ij}\in I$، یک **قانون وابستگی** به صورت $X\Rightarrow Y$ است که $X,Y\in I$ مجموعه ای از آیتم ها هستند که **itemset** نامیده می شوند و $X\cap Y=\varnothing$.

Support (α) برای یک قانون $X\Rightarrow Y$، درصدی از تراکنش های پایگاه داده است که شامل $X\cup Y$ باشند.

Confidence (λ) یا strength برای یک قانون $X\Rightarrow Y$، نسبت تعداد تراکنش هائیست که شامل $X\cup Y$ است، به تعداد تراکنش هایی که شامل X می باشد.

هنگام یافتن قوانین وابستگی در یک پایگاه داده، همه قوانین ممکن مد نظر نیستند، بلکه قوانین مهم. اهمیت به وسیله دو معیار support و confidence سنجیده می شود. confidence قدرت قانون را اندازه گیری می کند و support نشان می دهد که با چه نرخی قانون در پایگاه داده اتفاق می افتد. معمولا قوانین با مقدار confidence بزرگ و support کوچک مورد نظر هستند.

یک **large itemset**، itemset ایست که تعداد اتفاق افتادن آن در پایگاه داده بیشتر یا مساوی با مقدار آستانه α باشد. large itemset ها یک ویژگی مهم دارند: هر زیرمجموعه یک large itemset خود نیز large است.

رایج ترین روش یافتن قوانین وابستگی، تقسیم مساله به دو بخش است:

۱. یافتن large itemset ها (با توجه به α)
۲. تولید قوانین از large itemset های متواتر (با توجه به λ)

کارائی الگوریتم های یافتن قوانین وابستگی، با در نظر گرفتن تعداد اسکن های پایگاه داده و حداکثر تعداد itemsetهایی که بایستی شمرده شوند، ارزیابی می گردد.

۳-۲- جزئیات روش

روش یافتن قوانین وابستگی را به سه الگوریتم تقسیم می کنیم:

- Apriori-Gen
- Apriori
- AR-Gen

۳-۲-۱- Apriori-Gen

Apriori-Gen مجموعه itemset های کاندیدا- که ممکن است در مجموعه large itemset ها قرار بگیرند- را تولید می کند. روش کار بدین صورت زیر است:

- دو large itemset با اندازه i را در صورتی که i-1 آیتم آن ها مثل هم هستند با یکدیگر پیوند بده.
- large itemset های تولید شده را که زیرمجموعه هایی از آن ها large نیستند (در l_i قرار ندارند) حذف کن.

با یک مثال نحوه کار Apriori-Gen مشخص می شود. فرض کنید در مرحله چهارم اجرای الگوریتم Apriori داریم:

$$l_3 = \{abc, abd, acd, ace, bcd\}$$

l_3 را با l_3 پیوند می دهیم. ۲ itemset جدید به وجود می آیند:

- abcd از پیوند abc و abd
- acde از پیوند acd و ace

در این جا acde حذف می شود، چون ade در l_3 وجود ندارد. در نتیجه مجموعه کاندیداها در مرحله چهارم به صورت زیر است:

$$C_4 = \{abcd\}$$

۳-۲-۲- Apriori

این الگوریتم مجموعه large itemset ها را تولید می کند. مراحل الگوریتم به صورت زیر است:

- C_1 را برابر itemset های با اندازه ۱ (در واقع همان آیتم ها) قرار بده.
- تمام large itemset ها با اندازه ۱ را - به وسیله محاسبه support آن ها- تعیین کن و در l_1 قرار بده.
- تا زمانی که یک large itemset یافت می شود، انجام بده:
  - $i = i + 1$
  - $C_i = Apriori - Gen(l_{i-1})$
  - l_i را به وسیله محاسبه support آیتم های موجود در C_i تعیین کن.

۳-۲-۳- AR-Gen

AR-Gen در پایان قوانین را از large itemset های تولید شده در مرحله قبل، تولید می کند.

ورودی:

- D // پایگاه داده از تراکنش ها
- I // آیتم ها
- L // مجموعه large itemset
- α // support
- λ // confidence

خروجی:

- R // مجموعه قوانینی که α و λ را ارضا می کنند.

مراحل:

- $R = \varnothing$
- برای هر $l \in L$ انجام بده:
  - برای هر $x \subset l$ که $x \neq \varnothing$ انجام بده:
    - اگر $\frac{\text{support}(l)}{\text{support}(x)} \geq \alpha$ است، قانون $x \Rightarrow (1-x)$ را به R اضافه کن.

در بخش بعد یک روش یافتن قوانین وابستگی فازی براساس الگوریتم Apriori ارائه شده است.

۳-۳- یک روش داده کاوی فازی مبتنی بر الگوریتم Apriori، با زمان محاسباتی کاهش یافته

نظریه فازی، به دلیل سادگی و شباهتش به استدلال انسانی، هر چه بیشتر در سیستم های هوشمند استفاده می شود. الگوریتم های یادگیری فازی متعددی برای استنتاج قوانین از مجموعه داده ها طراحی شده اند. استفاده از مجموعه های فازی در داده کاوی نیز در سال های اخیر مورد توجه قرار گرفته است.

در این بخش روش پیشنهادی در [۳] به عنوان یک الگوریتم فازی یافتن قوانین وابستگی در تراکنش ها، با مقادیر کمی[1]، ارائه شده است. این الگوریتم مقادیر کمی را به متغیرهای زبانی[2] تبدیل کرده، تعدادی از این متغیرها را حذف می کند و نهایتا قوانین وابستگی را می یابد. الگوریتم بر روی مهم ترین متغیرهای زبانی متمرکز می شود و به این ترتیب پیچیدگی زمانی را کاهش می دهد.

۳-۳-۱- الگوریتم پیشنهادی به همراه یک مثال

نمادهای استفاده شده در الگوریتم، به شرح زیر هستند:

- n: تعداد تراکنش ها
- m: تعداد صفات
- $D^{(i)}$: i-امین تراکنش
- A_j: j-امین صفت
- $|A_j|$: تعداد نواحی فازی برای صفت A_j
- R_{jk}: k-امین ناحیه فازی برای صفت A_j
- $v_j^{(i)}$: مقدار صفت A_j برای تراکنش $D^{(i)}$
- $f_j^{(i)}$: مجموعه فازی تبدیل شده از $v_j^{(i)}$
- $f_{jk}^{(i)}$: مقدار عضویت $v_j^{(i)}$ در ناحیه R_{jk}
- $count_{jk}$: مجموع $f_{jk}^{(i)}$ در تمام تراکنش ها

[1] Quantitative
[2] Linguistic Terms

- $count_j^{\max}$: حداکثر مقدار در میان $count_{jk}$ها، k=1 تا $|A_j|$
- $R_j^{\max}$: ناحیه فازی متناظر با $count_j^{\max}$ برای صفت A_j
- α: مقدار support حداقل از پیش تعریف شده
- λ: مقدار confidence حداقل از پیش تعریف شده
- C_r: مجموعه آیتم های کاندیدا با r صفت (آیتم)
- $\overline{C}_r$: مجموعه موقتی برای ذخیره مقادیر فازی آیتم های C_r
- L_r: مجموعه large itemset های بزرگ با r صفت.

الگوریتم ابتدا هر مقدار کمی را به وسیله توابع عضویت[1] متغیرهای زبانی به مجموعه های فازی تبدیل می کند. سپس اندازه[2] هر متغیر را بر روی داده های تراکنش محاسبه و در جدولی ذخیره می کند. برای هر صفت تنها متغیر زبانی با اندازه حداکثر در مراحل بعدی پردازش در نظر گرفته می شود و تعداد نواحی فازی[3] به اندازه تعداد آیتم های اولیه باقی بماند. در نهایت پروسه داده کاوی فازی بر اساس شمارش های فازی[4] انجام گرفته و قوانین وابستگی فازی تولید می شوند. الگوریتم دارای **۱۲** مرحله است و ورودی و خروجی آن عبارتند از:

ورودی: n تراکنش، هر یک با m صفت، مجموعه ای از توابع عضویت، حداقل مقدار support α، مقدار confidence λ.

خروجی: مجموعه ای از قوانین وابستگی فازی.

در این قسمت مثالی را در نظر گرفته و هر مرحله الگوریتم بر روی مثال توضیح داده می شود.

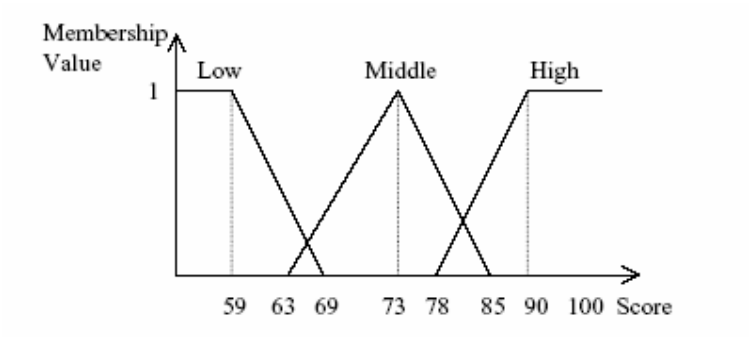
مجموعه داده ای با ۱۰ تراکنش در نظر می گیریم. هر تراکنش شامل نمرات یک دانشجو در دروس آمار (ST)، پایگاه داده (DB)، برنامه نویسی شی گرا (OOP)، ساختمان داده (DS) و مدیریت سیستم های اطلاعاتی (MIS) است (جدول ۲).

[1]Membership Functions
[2]Scalar Cardinality
[3]Fuzzy Regions
[4]Fuzzy Counts

جدول۲) پایگاه داده نمرات دانشجویان در ۱۰ درس

Case no.	ST	DB	OOP	DS	MIS
1	86	77	86	71	68
2	61	79	89	77	80
3	84	89	86	79	89
4	73	86	79	84	62
5	70	89	87	72	79
6	65	77	86	61	87
7	67	87	75	71	80
8	86	63	64	84	86
9	75	65	79	87	88
10	79	63	63	85	89

هر درس به عنوان یک صفت در نظر گرفته می شود. هدف یافتن قوانین وابستگی بین نمرات دروس است. توابع عضویت برای نمرات دروس در شکل ۳ نمایش داده شده اند.



شکل ۳) توابع عضویت مثلثی برای نمرات دروس

در این مثال از توابع عضویت مثلی به دلیل سادگی، درک آسان و کارایی محاسباتی استفاده می شود. همان طور که مشاهده می شود، هر صفت شامل سه ناحیه فازی است: Low، Middle، High. برای هر نمره سه مقدار عضویت فازی بر اساس توابع ذکر شده تولید می شود.

مراحل الگوریتم به صورت زیر است:

مرحله ۱. مقادیر موجود در هر تراکنش را به مجموعه های فازی تبدیل کن. به عنوان مثال نمره ST در تراکنش ۱ را در نظر می گیریم. مقدار ۸۶ برای این نمره به مجموعه فازی (0.0/Low + 0.0/Middle + 0.7/High) با استفاده از توابع عضویت داده شده، تبدیل می شود. نتیجه برای تمام تراکنش ها و درس ها در جدول ۳ آورده شده است.

جدول ۳) تبدیل مقادیر به مجموعه های فازی

Case no.	ST			DB			OOP			DS			MIS		
	L	M	H	L	M	H	L	M	H	L	M	H	L	M	H
1	0.0	0.0	0.7	0.0	0.7	0.0	0.0	0.0	0.7	0.0	0.8	0.0	0.1	0.5	0.0
2	0.8	0.0	0.0	0.0	0.5	0.1	0.0	0.0	0.9	0.0	0.7	0.0	0.0	0.4	0.2
3	0.0	0.1	0.5	0.0	0.0	0.9	0.0	0.0	0.7	0.0	0.5	0.1	0.0	0.0	0.9
4	0.0	1.0	0.0	0.0	0.0	0.7	0.0	0.5	0.1	0.0	0.1	0.5	0.7	0.0	0.0
5	0.0	0.7	0.0	0.0	0.0	0.9	0.0	0.0	0.8	0.0	0.9	0.0	0.0	0.5	0.1
6	0.4	0.2	0.0	0.0	0.7	0.0	0.0	0.0	0.7	0.8	0.0	0.0	0.0	0.0	0.8
7	0.2	0.4	0.0	0.0	0.0	0.8	0.0	0.8	0.0	0.0	0.8	0.0	0.0	0.4	0.2
8	0.0	0.0	0.7	0.6	0.0	0.0	0.5	0.1	0.0	0.0	0.1	0.5	0.0	0.0	0.7
9	0.0	0.8	0.0	0.4	0.2	0.0	0.0	0.5	0.1	0.0	0.0	0.8	0.0	0.0	0.8
10	0.0	0.5	0.1	0.6	0.0	0.0	0.6	0.0	0.0	0.0	0.0	0.6	0.0	0.0	0.9

مرحله ۲. یک جدول موقتی $\overline{C}_1$، شامل تمام دوتائی های $(R_{jk}\ ,\ f_{jk}^{(i)})$ تشکیل بده. نتیجه برای مثال در جدول ۴ آمده است.

جدول٤) مقادیر تبدیل شده به همراه درجه عضویت آن ها

Case	Set-of-itemsets
1	{(ST.High, 0.7), (DB.Middle, 0.7), (OOP.High, 0.7), (DS.Middle, 0.8), (MIS.Low, 0.1), (MIS.Middle, 0.5)}
2	{(ST.Low, 0.8), (DB.Middle, 0.5), (DB.High, 0.1), (OOP.High, 0.9), (DS.Middle, 0.7), (MIS.Middle, 0.4), (MIS.High, 0.2)}
3	{(ST.Middle, 0.1), (ST.High, 0.5), (DB.High, 0.9), (OOP.High, 0.7), (DS.Middle, 0.5), (DS.High, 0.1), (MIS.High, 0.9)}
4	{(ST.Middle, 1.0), (DB.High, 0.7), (OOP.Middle, 0.5), (OOP.High, 0.1), (DS.Middle, 0.1), (DS.High, 0.5),(MIS.Low, 0.7)}
5	{(ST.Middle, 0.7), (DB.High, 0.9), (OOP.High, 0.8), (DS.Middle, 0.9), (MIS.Middle, 0.5), (MIS.High, 0.1)}
6	{(ST.Low, 0.4), (ST.Middle, 0.2), (DB.Middle, 0.7), (OOP.High, 0.7), (DS.Low, 0.8), (MIS.High, 0.8)}
7	{(ST.Low, 0.2), (ST.Middle, 0.4), (DB.High, 0.8), (OOP.Middle, 0.8), (DS.Middle, 0.8), (MIS.Middle, 0.4), (MIS.High, 0.2)}
8	{(ST.High, 0.7), (DB.Low, 0.6), (OOP.Low, 0.5), (OOP.Middle, 0.1), (DS.Middle, 0.1), (DS.High, 0.5), (MIS.High, 0.7)}
9	{(ST.Middle, 0.8), (DB.Low, 0.4), (DB.Middle, 0.2), (OOP.Middle, 0.5), (OOP.High, 0.1), (DS.High, 0.8), (MIS.High, 0.8)}
10	{(ST.Middle, 0.5), (ST.High, 0.1), (DB.Low, 0.6), (OOP.Low, 0.6), (DS.High, 0.6), (MIS.High, 0.9)}

مرحله ۳. برای هر ناحیه R_{jk} ذخیره شده در $\overline{C}_1$، مقدار اندازه آن از $\overline{C}_1$ محاسبه کن:

$$count_{jk} = \sum_{i=1}^{n} f_{jk}^{(i)}$$

به عنوان مثال ناحیه ST.Low را در نظر می گیریم. اندازه این ناحیه مساوی (۰,۸+۰,۴+۰,۲)=۱,۴ خواهد بود. نتیجه برای تمام نواحی در جدول ۵ نشان داده شده است.

جدول۵) اندازه هر ناحیه فازی برای تمام صفات

Itemset	Count
ST.Low	1.4
ST.Middle	3.7
ST.High	2.0
DB.Low	1.6
DB.Middle	2.1
DB.High	3.4
OOP.Low	1.1
OOP.Middle	1.9
OOP.High	4.0
...	...
MIS.High	4.6

مرحله ۴. برای هر صفت، در میان نواحی موجود، ناحیه $R_j^{\max}$ با بالاترین مقدار count را پیدا کن. $R_j^{\max}$ ناحیه متناظر با مقدار زیر است:

$$count_j^{\max} = Max_{k=1}^{|A_j|}(count_{jk})$$

به عنوان مثال ST را در نظر می گیریم. مقدار ناحیه ها ۱,۴ برای Low، ۳,۷ برای Middle و ۲,۰ برای High است. بنابراین Middle به عنوان درس ST در مراحل بعدی پردازش استفاده می شود. به همین ترتیب High برای صفات DB، OOP، MIS و Middle برای ST و DS انتخاب می شوند.

مرحله ۵. برای هر ناحیه انتخاب شده در مرحله ۴، بررسی کن که آیا مقدار count آن بزرگتر یا مساوی مقدار support حداقل α هست یا نه. اگر برای $R_j^{\max}$ این شرط برقرار است، آن را در مجموعه ۱-large itemset های بزرگ قرار بده:

$$L_1 = \{R_j^{\max} \mid count_j^{\max} \geq \alpha, 1 \leq j \leq m\}$$

در این مثال $\alpha = 2.0$ فرض می کنیم. 1-large itemset ها در جدول ۶ نشان داده شده اند.

جدول٦) آیتم ست های بزرگ با اندازه ۱

Itemset	Count
ST.Middle	3.7
DB.High	3.4
OOP.High	4.0
DS.Middle	3.9
MIS.High	4.6

مرحله ۶. r را مساوی ۱ قرار بده. r نشان دهنده تعداد آیتم های موجود در مجموعه itemset های فعلی است.

مرحله ۷. مجموعه کاندیداهای C_{r+1} را از مجموعه L_r ایجاد کن. الگوریتم دو آیتم را در صورتی ادغام می کند که r-1 آیتم در دو itemset یکسان و آیتم دیگر متفاوت باشد. در C_{r+1} itemset هایی که تمام زیر مجموعه های rتایی آن ها در L_rاست، را ذخیره کن.

در این مثال در ابتدا C_2از L_1به صورت زیر تولید می شود:

(ST.Middle, DB.High), (ST.Middle, OOP.High), (ST.Middle, DS.Middle), (ST.Middle, MIS.High), (DB.High, OOP.High), (DB.High, DS.Middle), (DB.High, MIS.High), (OOP.High, DS.Middle), (OOP.High, MIS.High), (DS.Middle, MIS.High).

مرحله ۸. یک مجموعه موقتی $\bar{C}_{r+1}$ایجاد کن. در مثال، $\bar{C}_2$ساخته می شود.

مرحله ۹. برای هر (r+1)-itemset جدید در C_{r+1}،

الف. مقدار فازی آن را برابر کمینه مقدار فازی آیتم های موجود در آن قرار بده.

ب. مقادیر به دست آمده را در $\bar{C}_{r+1}$ ذخیره کن.

ج. با استفاده از $\bar{C}_{r+1}$ مجموع count را برای هر itemset محاسبه کن.

د. اگر count بزرگتر یا مساوی α است، itemset را در مجموعه itemsetهای بزرگ L_{r+1} قرار بده.

برای مثال مورد نظر، دراین مرحله برای هر itemset s در C_2گام های زیر انجام می شود:

الف. در هر تراکنش، مقدار عضویت فازی را از $\bar{C}_1$ محاسبه کن. **عملگر کمینه** به عنوان اشتراک استفاده می شود. به عنوان مثال itemset کاندیدای (ST.Middle, DB.High) را در نظر می گیریم. فقط در موارد ۳، ۴، ۵ و ۷ هر دو آیتم ST.Middle و DB.High در $\bar{C}_1$ وجود دارد. نتیجه در جدول ۷ آورده شده است. مقادیر برای دیگر ۲-itemsetها به طور مشابه محاسبه می شود.

جدول۷) مقدار فازی برای دو آیتم ست خاص در هر تراکنش

Case	ST.Middle	DB.High	(ST.Middle, DB.High)
1	0.0	0.0	0.0
2	0.0	0.1	0.0
3	0.1	0.9	0.1
4	1.0	0.7	0.7
5	0.7	0.9	0.7
6	0.2	0.0	0.0
7	0.4	0.8	0.4
8	0.0	0.0	0.0
9	0.8	0.0	0.0
10	0.5	0.0	0.0

ب. دوتائی های $(s, f_s^{(i)})$ برای سطر iام را، وقتی که $f_s^{(i)} \neq 0$، در $\overline{C}_2$ ذخیره کن. نتایج در جدول ۸ نشان داده شده است.

جدول۸) مقدار فازی برای آیتم ست های با اندازه ۲ در هر تراکنش

Case	Set-of-itemsets
1	{(OOP.High, DS.Middle, 0.7)}
2	{(DB.High, OOP.High, 0.1), (DB.High, DS.Middle, 0.1), (DB.High, MIS.High, 0.1), (OOP.High, DS.Middle, 0.7), (OOP.High, MIS.High, 0.2), (DS.Middle, MIS.High, 0.2)}
3	{(ST.Middle, DB.High, 0.1), (ST.Middle, OOP.High, 0.1), (ST.Middle, DS.Middle, 0.1), (ST.Middle, MIS.High, 0.1), (DB.High, OOP.High, 0.7), (DB.High, DS.Middle, 0.5), (DB.High, MIS.High, 0.9), (OOP.High, DS.Middle, 0.5), (OOP.High, MIS.High, 0.7), (DS.Middle, MIS.High, 0.5)}
4	{(ST.Middle, DB.High, 0.7), (ST.Middle, OOP.High, 0.1), (ST.Middle, DS.Middle, 0.1), (DB.High, OOP.High, 0.1), (DB.High, DS.Middle, 0.1), (OOP.High, DS.Middle, 0.1)}
5	{(ST.Middle, DB.High, 0.7), (ST.Middle, OOP.High, 0.7), (ST.Middle, DS.Middle, 0.7), (ST.Middle, MIS.High, 0.1), (DB.High, OOP.High, 0.8), (DB.High, DS.Middle, 0.9), (DB.High, MIS.High, 0.1), (OOP.High, DS.Middle, 0.8), (OOP.High, MIS.High, 0.1), (DS.Middle, MIS.High, 0.1)}
6	{(ST.Middle, OOP.High, 0.2), (ST.Middle, MIS.High, 0.2), (OOP.High, MIS.High, 0.7)}
7	{(ST.Middle, DB.High, 0.4), (ST.Middle, DS.Middle, 0.4), (ST.Middle, MIS.High, 0.2), (DB.High, DS.Middle, 0.8), (DB.High, MIS.High, 0.2), (DS.Middle, MIS.High, 0.2)}
8	{(DS.Middle, MIS.High, 0.1)}
9	{(ST.Middle, OOP.High, 0.1), (ST.Middle, MIS.High, 0.8), (OOP.High, MIS.High, 0.1)}
10	{(ST.Middle, MIS.High, 0.5)}

ج. مقادیر $counts = \sum_{i=1}^{n} f_s^{(i)}$ را با استفاده از $\overline{C}_2$ محاسبه کن. به این ترتیب اندازه هر itemset کاندیدا در C_2 مشخص می شود. نتایج در جدول ۹ نمایش داده شده اند.

جدول۹) مقدار فازی برای آیتم ست های با اندازه ۲

Itemset	Count
(ST.Middle, DB.High)	1.9
(ST.Middle, OOP.High)	1.2
(ST.Middle, DS.Middle)	1.3
(ST.Middle, MIS.High)	1.9
(DB.High, OOP.High)	1.7
(DB.High, DS.Middle)	2.4
(DB.High, MIS.High)	1.3
(OOP.High, DS.Middle)	2.8
(OOP.High, MIS.High)	1.8
(DS.Middle, MIS.High)	1.1

د. بررسی کن که آیا مقادیر count ها بزرگتر یا مساوی ۲= α هستند یا خیر. تنها دو itemset (DB.High, DS.Middle) و (OOP.High, DS.Middle) در L_2 وارد می شوند. (جدول ۱۰)

جدول۱۰) آیتم ست های بزرگ با اندازه ۲

Itemset	Count
(DB.High, DS.Middle)	2.4
(OOP.High, DS.Middle)	2.8

مرحله ۱۰. اگر مجموعه L_{r+1} تهی است، به مرحله بعد برو. در غیر این صورت قرار بده $r = r+1$ و مراحل ۷ تا ۱۰ را مجددا تکرار کن. چون در این مثال L_2 تهی نیست، $r = 2$ قرار می گیرد. سپس مراحل ۷ تا ۱۰ برای یافتن l_3 تکرار می گردد. ابتدا C_3 از L_2 تولید می شود:

$C3 = \{(\text{DB.High, OOP.High, DS.Middle})\}$

Count تنها itemset موجود ۱,۵ و کوچکتر از ۲,۰ است. پس در مجموعه l_3 قرار نمی گیرد. چون l_3 تهی است، به مرحله ۱۱ می رویم.

مرحله ۱۱. قوانین وابستگی را برای تمام q-itemset های بزرگ، در دو گام ایجاد کن:

الف. تمام قوانین ممکن را تشکیل بده:

$$s_1 \Lambda ... s_{k-1} \Lambda s_{k+1} \Lambda ... \Lambda s_q \rightarrow s_k, k = 1-q$$

در این مثال، ۴ قانون زیر از 2-itemsetها بزرگ حاصل می شود:

1- If DB = High, then DS=Middle;
2- If DS=Middle, then DB=High;
3- If OOP=High, then DS=Middle;
4- If DS=Middle, then OOP=High;

ب. مقدار confidence را برای هر قانون محاسبه کن:

$$\frac{\sum_{i=1}^{n} f_s^{(i)}}{\sum_{i=1}^{n} f_{s_1}^{(i)} \wedge \ldots \wedge f_{s_{k-1}}^{(i)} \wedge f_{s_{k+1}}^{(i)} \wedge \ldots \wedge f_{s_q}^{(i)}}$$

در مثال فرض می کنیم که مقدار confidence حداقل ۰٫۷= λ است. به عنوان مثال اولین قانون تولید شده در بالا را در نظر می گیریم:

$$\frac{\sum_{i=1}^{10} (DB.High \cap DS.Middle)}{\sum_{i=1}^{10} (DB.High)} = \frac{2.4}{3.4} = 0.71$$

به طور مشابه مقدار confidence برای قانون دوم ۰٫۶۲، برای قانون سوم ۰٫۷۰ و برای قانون چهارم ۰٫۷۲ به دست می آید.

مرحله ۱۲. قوانین با مقدار confidence بزرگتر یا مساوی با λ را به عنوان خروجی الگوریتم مشخص کن.

در مثال مورد نظر با توجه به مقدار ۰٫۷ = λ ، قوانین زیر به دست می آید:

۱. اگر نمره پایگاه داده بالا است، آن گاه نمره ساختمان داده متوسط است، با مقدار confidence ۰٫۷۱.
۲. اگر نمره برنامه نویسی شی گرا بالا است، آن گاه نمره ساختمان داده متوسط است، با مقدار confidence ۰٫۷۰.
۳. اگر نمره ساختمان داده متوسط است، آن گاه نمره برنامه نویسی شی گرا بالا است، با مقدار confidence ۰٫۷۲

این سه قانون به عنوان یک فوق- دانش[1] برای مجموعه داده مثال بیان شده، در نظر گرفته می شود. نتایج حاصل از اجرای الگوریتم بر روی داده یک سوپرمارکت در [۳] کارائی الگوریتم را نشان می دهد.

[1]meta-knowledge

۴- درخت تصمیم‌گیری

همان‌طور که بیان شد کلاس‌بندی از مباحث اصلی در داده‌کاوی است. درخت تصمیم‌گیری به طور گسترده در کلاس‌بندی داده‌ها استفاده می‌شود. الگوریتم‌های متفاوتی برای تولید درخت تصمیم‌گیری به کار رفته است، از جمله ID3، C4.5 و CART. این الگوریتم‌ها با تقسیم‌بندی متوالی فضای ویژگی‌ها، به صورت بازگشتی ساختار درخت را می‌سازند. درخت نهایی که به دست می‌آید فضای تصمیم را به طور کامل به چندین ناحیه مجزا تقسیم‌بندی می‌کند.

درخت تصمیم‌گیری اولیه که پایه‌گذاری شد برای ویژگی‌های با مقادیر گسسته به کار می‌رود. اگر ویژگی‌های با مقادیر پیوسته داشته باشیم، باید آن‌ها را به طرز مناسبی به چندین بازه بشکنیم. یک راه، تقسیم‌بندی دامنه مقادیر ویژگی به دو زیربازه با استفاده از یک آستانه است. راه دیگر تقسیم‌بندی دامنه مقادیر ویژگی به بیش از دو زیربازه است، این کار با استفاده از مجموعه‌ای از نقاط جداکننده انجام می‌شود. در هر دو روش ذکرشده، نقاط جداکننده به طرز شدید و تندی کلاس‌ها را از هم جدا می‌کنند. این روش در مورد فضاهایی که به طور واضح و مشخص به چندین کلاس تقسیم می‌شوند و مرز کلاس‌ها کاملا مشخص است، مفید می‌باشد. اما باید توجه داشت که در مسائلی که در دنیای واقع وجود دارند، مرز کلاس‌ها به صورت واضح مشخص نیست، در این صورت درخت تصمیم‌گیری خطای زیادی روی داده‌های تست خواهد داشت، هرچند داده‌های آموزشی را به درستی کلاس‌بندی کند. برای حل این مشکل راه‌های زیادی که بر پایه احتمال و امکان است، ارائه شده است.

در درخت تصمیم‌گیری کلاسیک هنگام کلاس‌بندی یک نمونه جدید، فقط یک شاخه از درخت که یک قانون خاص را تولید خواهد کرد قادر به کلاس‌بندی آن نمونه است. در این صورت ممکن است وقتی ورودی به نقطه جداکننده بسیار نزدیک باشدکلاس‌بندی اشتباه صورت گیرد. این مشکل می‌تواند بدین صورت حل شود که در مورد یک نمونه جدید چندین شاخه تصمیم‌گیری کند، ولی با احتمال‌های متفاوت.

این روش با استفاده از مفاهیم مجموعه‌های فازی قابل پیاده‌سازی است. در درخت تصمیم‌گیری فازی، فرآیند منطق استلزام فازی این امکان را فراهم می‌آورد که یک نمونه توسط بیش از یک قانون کلاس‌بندی شود و نتیجه نهایی از ترکیب چند نتیجه به‌دست آید. در قسمت‌های بعد روش‌های فازی ساخت درخت‌ها که در [۴] آمده است، شرح داده می‌شود.

۴-۱- درخت تصمیم فازی و کلاس‌بندی فازی

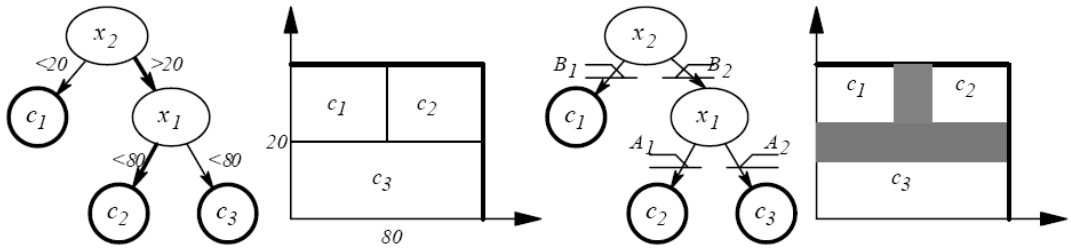
درخت تصمیم‌گیری فازی با استفاده از منطق استلزام فازی، کلاس‌بندی را عام‌تر و قدرتمندتر می‌کند. شکل ۴ تفاوت درخت تصمیم‌گیری فازی و کلاسیک را نشان می‌دهد. شکل ۴الف درخت تصمیم‌گیری کلاسیک را نشان می‌دهد که بر اساس جداسازی سخت[1] کار می‌کند، در صورتی که درخت شکل ۴ب از جداسازی نرم[2] استفاده می‌کند. در هر دو درخت هر مسیر از ریشه به برگ یک قانون را مشخص می‌کند.

با استفاده از جداسازی سخت فضای تصمیم‌گیری به کلاس‌هایی که کاملا جدا هستند و اشتراکی ندارند تقسیم می‌شوند، همانطور که در سمت راست شکل ۴الف مشاهده می‌کنید که هر داده‌ای به یک کلاس منسوب می‌شود. در مقابل ، درخت تصمیم‌گیری فازی نتیجه را با امکانی در بازه [0 1] بیان می‌کند، همان‌طور که در سمت راست شکل ۴ب مشاهده می‌کنید. درخت تصمیم‌گیری فازی روش قوی‌تری را برای جلوگیری از کلاس‌بندی اشتباه فراهم می‌کند.

برای مثال درخت تصمیم‌گیری کلاسیک نمونه $(x_1=81,x_2=25)$ را در کلاس c_3 کلاس‌بندی می‌کند. حال اگر نمونه به علت نویز اندکی $(x_1=79,x_2=25)$ داده شود، درخت تصمیم‌گیری کلاسیک به اشتباه، نمونه را در c_2 کلاس‌بندی می‌کند. اما اگر همین نمونه را به درخت تصمیم‌گیری فازی دهیم چه نتیجه‌ای حاصل می‌شود؟ درخت تصمیم‌گیری فازی نمونه را با امکان‌هایی در هر یک از کلاس‌های c_1، c_2 و c_3 قرار می‌دهد، برای این نمونه به ترتیب امکان‌های $\pi_1=0$، $\pi_2=0.25$ و $\pi_3=0.48$ به دست می‌آید. با توجه به این نتیجه‌ها، می‌توان تصمیم‌گیری نهایی را کرد یا پیش‌بینی‌های دیگری انجام داد یا یک یادگیر سطح بالاتری برای تصمیم‌گیری به کار برد.

هر شاخه درخت تصمیم‌گیری فازی از ریشه تا برگ، یک قانون تصمیم‌گیری تولید می‌کند که می تواند به فرم "if x_1 is A_1 and … x_i is A_i … and x_m is A_m then class is c_j" باشد، 'x_1 is A_1'، 'x_i is A_i' و 'x_m is A_m' تصمیم‌های فازی در نودها و شاخه‌های مربوطه هستند و c_j کلاس در برگ است. یک مثال از قانونی که از شکل ۴ب به‌دست می‌آید، قانون "if x_2 is B_2 and x_1 is A_1 then class is c_2" است.

[1] hard Discretization
[2] Soft Discretization



الف) درخت تصمیم‌گیری کلاسیک

ب) درخت تصمیم‌گیری فازی

شکل ۴. درخت تصمیم‌گیری و فضای تصمیم‌گیری

کلاس‌بندی برای نمونه جدید داده‌شده از درجه‌های تطبیق نمونه به هر نود، از ریشه تا برگ، به‌دست می‌آید.در مثال قبل، امکان تعلق یک نمونه به کلاس c_2 از رابطه ۱ به‌دست می‌آید:

$$\pi_2 = \otimes[B_2(x_2), A_1(x_1)] \qquad (۱)$$

علامت $\otimes$ عملگر فازی ضرب است (معمولا مینیمم یا میانگین وزن‌دارشده استفاده می‌شود) و $B_2(x_2)$ و $A_1(x_1)$ به ترتیب درجه عضویت x_2 به B_2 و x_1 به A_1 است. به روش مشابه امکان این‌که نمونه‌ای به هر کلاس دیگر تعلق داشته باشد محاسبه می‌شود. اگر بیشتر از یک برگ به کلاس c_i تعلق داشت، می‌گوییم مقدار $\pi_I = \oplus(\pi_j)$ امکان تعلق به کلاس مربوطه است. $\oplus$ عملگر فازی جمع است. در آخر اگر یک مقدار امکان مانند π_k خیلی بالاتر از بقیه بود، یعنی $\pi_k >> \pi_{i\neq k}$ آن‌گاه کلاس c_k به آن نمونه منسوب می‌شود.

۴-۲ - ساخت درخت تصمیم از داده‌های پیوسته

در ساخت درخت تصمیم از داده‌های با مقادیر پیوسته، یک آستانه مناسب T که ویژگی پیوسته A را به دو زیربازه $A_1=[\min(A),T)]$ و $A_2=(T,\max(A)]$ جدا کند مشخص می‌شود. T با استفاده از میزان بهره اطلاعات که با جداسازی توسط این آستانه به‌دست می‌آید انتخاب می‌شود. با در دست‌داشتن آستانه، $A \leq T$ به شاخه سمت چپ نود تصمیم و $A > T$ به شاخه سمت راست منسوب می‌شود.

فرض کنید که می‌خواهیم یک ویژگی را برای نودی که در آن مجموعه S از N نمونه قرار دارد، انتخاب کرده‌ایم. این نمونه‌ها را بر اساس مقادیر ویژگی پیوسته A مرتب می‌کنیم، فرض کنید این دنباله مرتب شده $a_1, a_2, ..., a_N$ باشد. روی هر جفت از مقادیر همسایه در این دنباله می‌توان

یک آستانه $T=(a_i+a_{i+1})/2$ حدس زد. هر یک از این Tها یک کاندید برای آستانه نهایی است. فرض کنید که k کلاس $c_1,c_2,...c_k$ وجود دارد و $p(c_j,S)$ نسبتی از نمونه‌ها ی S باشد که متعلق به کلاس c_j هستند. آنتروپی با رابطه (۲) تعریف می‌شود:

$$E(S)=-\sum_{j=1}^{k}p(c_j,S)\log(p(c_j,S)) \qquad (۲)$$

بعد از این که S با استفاده از آستانه T به دو زیرمجموعه S_1 و S_2 تقسیم شد، آنتروپی اطلاعات توسط رابطه (۳) تعریف می‌شود:

$$E(A,T;S)=\frac{N_1}{N}E(S_1)+\frac{N_2}{N}E(S_2) \qquad (۳)$$

$$E(S_1)=-\sum_{j=1}^{k}p(c_j,S_1)\log(p(c_j,S_1)) \qquad (۴)$$

$$E(S_2)=-\sum_{j=1}^{k}p(c_j,S_2)\log(p(c_j,S_2)) \qquad (۵)$$

که در این رابطه‌ها $N_1=|S_1|$، $N_2=|S_2|$ و $|N|=S$ تعداد نمونه‌های S_1، S_2 و S_3 هستند و $p(c_j,S_1)$ و $p(c_j,S_2)$ نسبتی از نمونه‌های کلاس c_j در S_1 و S_2 است. برای تمام آستانه‌های کاندید مقدار آنتروپی اطلاعات را از رابطه (۳) بدست می‌آوریم. آستانه‌ای که مقدار کمینه را برای این کمیت حاصل کند به عنوان آستانه بهینه انتخاب می‌شود.

همچنین برای یک نود از آن‌جایی که انتخاب‌های زیادی برای ویژگی داریم، ویژگی A_j که برای آن مقدار $E(A_j,T^j_A;S)$ کمینه باشد یا بهره اطلاعات $E(S)-E(A_j,T^j_A;S)$ بیشینه باشد انتخاب می‌شود.

بدین‌ترتیب با شروع از ریشه برای هر نود ویژگی بهینه و آستانه بهینه برای آن ویژگی به‌دست می‌آید و این کار ادامه پیدا می‌کند تا تمام درخت ساخته شود. در قسمت بعد با چگونگی استفاده از فازی در روند ایجاد این درخت آشنا خواهید شد.

۴-۳ - ساخت درخت تصمیم از داده‌های پیوسته با جداسازی نرم

جداسازی نرم را می‌توان توسعه‌ای از جداسازی سخت بیان کرد. تمام معیارهایی که برای انتخاب ویژگی و آستانه با استفاده از مفاهیم احتمال تعریف شد، در جداسازی نرم به طور مشابه با استفاده از مفاهیم امکان در منطق فازی تعریف می‌شوند.

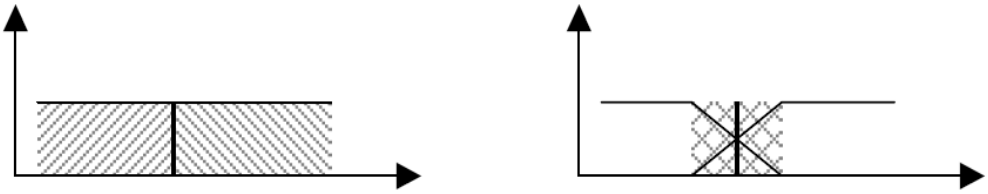
مجموعه کلاسیک A_c با تابع سخت $A_c(a):\Omega \to \{0,1\}; a \in \Omega$ بیان می‌شود، همچنین مجموعه فازی A با تابع عضویت $A_c(a):\Omega \to [0,1]; a \in \Omega$ مشخص می‌شود. عضویت $A(a)$ امکان این که A مقدار a را بگیرد است.

اگر $Q=\{A_1,\ldots,A_k\}$ یک خانواده از مجموعه‌های فازی روی مجموعه مرجع $\Omega=\{a_1,\ldots,a_i,\ldots,a_m\}$ باشد، Q یک تقسیم پارتیشن فازی از Ω است، هرگاه:

$$\sum_{\gamma=1}^{k} A_\gamma(a) = 1, \forall a \in \Omega \qquad (۶)$$

یک جداسازی سخت، همانطور که در شکل ۵الف نشان داده شده است، با استفاده از یک آستانه تعریف می‌شود که مرزی بین دو مجموعه کلاسیک ایجاد می‌کند. در مقابل یک جداسازی نرم با یک جفت مجموعه فازی که یک پارتیشن فازی ایجاد می‌کند، تعریف می‌شود، همان‌طور که در شکل ۵ب مشاهده می‌شود.

برخلاف روش کلاسیک کلاس‌بندی، در روش نرم کلاس‌ها می‌توانند هم‌پوشانی داشته باشند. جداسازی نرم با سه پارامتر یا تابع تعریف می‌شود، نقطه تقاطع T و دو تابع عضویت مجموعه‌های فازی A_1 و A_2 به طوری که $A_1(a)+A_2(a)=1$. نقطه تقاطع T نقطه‌ای انتخاب می‌شود که بهره اطلاعات را در کلاس‌بندی بیشینه کند. توابع عضویت زوج مجموعه فازی با توجه به مشخصات ویژگی ، مانند نامعلومی[1] ویژگی مربوطه مشخص می‌شوند. معمولا هم‌پوشانی زیاد برای یک ویژگی بسیار نامعلوم به کار می‌رود، برای مثال می‌توان از میانگین فاصله نمونه‌ها به عنوان عرض هم‌پوشانی استفاده کرد.



a) جداسازی سخت با استفاده از

b) جداسازی نرم با استفاده از پارتیشن فازی

شکل۵. جداسازی سخت و جداسازی نرم

آنتروپی کلاس فازی در S با رابطه (۷) تعریف می‌شود:

[1] Uncertainty

$$E_F(S)=\sum_{j=1}^{k} p(c_j,S)\log p(c_j,S) \quad (۷)$$

$$p(c_j,S)=\sum_{a_i\in c_j}(A_1(a_i)+A_2(a_i)) \quad (۸)$$

رابطه (۸) نسبت فازی نمونه‌ها در S است. بعد از جداسازی نرم آنتروپی اطلاعات کلاس ، توسط رابطه (۹) به‌دست می‌آید.

$$E_F(A,T;S)=\frac{N_F^{S_1}}{N_F^{S}}E_F(S_1)+\frac{N_F^{S_1}}{N_F^{S}}E_F(S_2) \quad (۹)$$

$$E_F(S_1)=-\sum_{j=1}^{k} p(c_j,S_1)\log p(c_j,S_1) \quad (۱۰)$$

$$E_F(S_2)=-\sum_{j=1}^{k} p(c_j,S_2)\log p(c_j,S_2) \quad (۱۱)$$

$$P(c_j,S_k)=N_F^{S_k c_j}/N_F^{S_k}, k=1,2 \quad (۱۲)$$

که در این رابطه‌ها $N_F^{S}=\sum_{i=1}^{|s|}(A_1(a_i)+A_2(a_i))$، $N_F^{S_1}=\sum_{i=1}^{|S|}A_1(a_i)$، $N_F^{S2}=\sum_{i=1}^{|S|}A_2(a_i)$ و $N_F^{S_k c_j}=\sum_{a_i\in c_j}A_k(a_i),(k=1,2)$. مانند کلاس‌بندی کلاسیک بهره اطلاعات، $E_F(S)-E_F(A,T;S)$، برای تولید بهترین جداسازی برای صفت مربوطه استفاده می‌شود.

جزئیات کار ۸ مرحله مانند زیر است:

۱. فرض کنید نود حاضر شامل N نمونه است. مجموعه نمونه‌های این نود را S می‌نامیم. نمونه‌ها را بر اساس مقدار ویژگی A مرتب کنید، حال دنباله مرتب $a_1,a_2,\ldots,a_N$ را از نمونه‌ها داریم.

۲. نقطه‌های جداکننده کاندید را تولید کن، $T=(a_i+a_{i+1})/2$.

۳. با استفاده از زوج مجموعه‌های فازی A_1 و A_2 که یک پارتیشن فازی روی نقاط جداکننده تشکیل می‌دهند، نقطه‌های کاندید را فازی‌سازی کن تا کاندیدهای جداکننده نرم را به‌دست آوری.

۴. با استفاده از رابطه (۵) هر کاندید برای جداکنندگی نرم را ارزیابی کن.

۵. جداکننده نرمی که کمینه مقدار E_F را برای ویژگی A دارد انتخاب کن.

۶. مراحل ۱ تا ۵ را تکرار کن تا جداککننده نرم برای بقیه ویژگی‌ها را هم داشته باشیم.

۷. ویژگی A_j که جداسازی آن کمینه مقدار $E_F(A,T;S)$ را دارد انتخاب کن تا دو نود و شاخه فرزند ایجاد شوند.

۸. برای هر شاخه بدست‌آمده سطح درستی را با رابطه (۱۳) به‌دست آور:

$$\eta_1 = \frac{N_F^{S_1}}{N_F^S}, \eta_2 = \frac{N_F^{S_2}}{N_F^S} \qquad (۱۳)$$

اگر $\eta_1 \le \alpha$ یا $\eta_2 \le \alpha$، شاخه‌های مربوطه را حذف کن. اگر $\eta_1 > \alpha$ یا $\eta_2 > \alpha$ سطح درستی هر شاخه که به کلاس jام تعلق دارد از رابطه‌های (۱۴) به‌دست می‌آید:

$$\mu_{1j} = \frac{\sum_{a_i \in c_j} A_1(a_i)}{N_F^{S_1}}, \quad \mu_{2j} = \frac{\sum_{a_i \in c_j} A_2(a_i)}{N_F^{S_2}} \qquad (۱۴)$$

اگر $\max_{j=1}^{k}(\mu_{1j}) \ge \beta$ یا $\max_{j=1}^{k}(\mu_{2j}) \ge \beta$ شاخه مربوطه به عنوان برگ ختم می‌شود و این برگ به کلاس c_j منسوب می‌شود. درغیر این صورت S به S_1 و S_2 تقسیم می‌شود:

$$S_1 = \{S \mid A_1(a_i) \ge \lambda, a_i \in S \qquad (۱۵)$$

$$S_2 = \{S \mid A_2(a_i) \ge \lambda, a_i \in S$$

مراحل بالا برای هر نود فرزند تکرار می‌شود تا جایی که شرط بالا (رابطه‌های ۱۳ و ۱۴) برقرار شود.

معمولا α=0.1~0.2، β=0.8~0.9 و λ=0.5 انتخاب می‌شوند. α کمتر و β بیشتر درخت بزرگتری با دقت بالاتر در آموزش تولید می‌کند. وقتی داده نامعلوم یا نویزی باشد، α بسیار کمتر و β بسیار بیشتر باعث می‌شود درخت تصمیم تولیدشده بسیار بیش‌پوشش شود.

۵- نتیجه گیری

امروزه اهمیت و نقش داده‌کاوی در تمام زمینه‌ها بسیار روشن است. در روش‌های کلاسیک داده‌کاوی با مشکل جداسازی سخت داده‌ها، نمونه‌های آموزشی نویزی، همچنین بیش‌پوشش شدن نمونه‌های آموزشی و نیز نمونه‌های دور مواجه هستیم. با به کارگیری مجموعه‌های فازی سعی می‌کنیم اثر این مشکلات را کمتر کنیم.

الگوریتم بیان‌شده در روش Apriori در کشف قوانین وابستگی مقادیر کمی را به متغیرهای زبانی[1] تبدیل کرده، تعدادی از این متغیرها را حذف می کند و نهایتا قوانین وابستگی را می یابد. درخت تصمیم‌گیری فازی نیز سعی در نرم‌کردن جداسازی کلاس‌ها دارد. در عمل به این نتیجه می‌رسیم که استفاده از روش‌های فازی در هر یک از این مسائل بسیار کارا خواهد بود. همواره مطالعاتی در زمینه چگونگی به‌کاربردن روش‌های فازی در دیگر روش‌های داده‌کاوی صورت می‌گیرد.

[1]Linguistic Terms

۶- مراجع

[۱] احمد سعیدی، "داده کاوی، مفهوم و کاربرد آن در آموزش عالی"، نامه آموزش عالی، شماره۱۸، اسفند ۸۴.

[2] M. H. Dunham, “Data Mining Introductory and Advanced Topics”, Prentice Hall, 2002.

[3] T. Hong, C. Kuo, S. Wang, “A fuzzy AprioriTid mining algorithm with reduced computational time”, Elsevier, Applied Soft Computing 5, pp. 1–10, March 2004.

[4] Peng and P. Flach. “Soft Discretization to Enhance the Continuous Decision Tree Induction” pages 109--118. ECML/PKDD'01 workshop notes, September 2001.

[۵] وب سایت شرکت IranSQL: http://www.sqliran.com