

# کاوش قوانین همبستگی فازی

مسعود فاطمی<sup>۱</sup>، بهرام سلامت روندی<sup>۲</sup>

<sup>۱</sup>دانشگاه صنعتی اصفهان، [m.fatemi@ec.iut.ac.ir](mailto:m.fatemi@ec.iut.ac.ir)

<sup>۲</sup>دانشگاه صنعتی اصفهان، [b.salamat@ec.iut.ac.ir](mailto:b.salamat@ec.iut.ac.ir)

صورت همزمان اتفاق می افتد و با یکدیگر همکاری دارند همبستگی (association) نام دارد.

الگوهای مکرر الگوهایی هستند که در یک مجموعه داده به فراوانی ظاهر می شوند. به عنوان مثال، مجموعه-ای از آیتم ها، مانند شیر و نان که به فراوانی با همدیگر در یک مجموعه داده ی تراکنشی ظاهر می شوند، یا یک توالی مانند در ابتدا خریدن یک کامپیوتر و سپس یک دوربین دیجیتال و بعد از آن یک کارت حافظه، اگر مکرر در تاریخچه پایگاه داده فروشگاه رخ داده باشند، یک الگوی ترتیبی (مکرر) می باشند.

اما روابط هم بستگی که تا اینجا به آن اشاره کردیم به صورت روابط باینری مطرح می شوند که در آن ها یک ویژگی یا یک مشخصه یا حضور دارد یا خیر. برای غلبه بر این مشکل روش تحلیل قوانین همبستگی به صورت فازی یا Fuzzy Association rule mining به وجود آمده است که در آن به آیتم ها و ویژگی ها اجازه داده می-شود متعلق به بازه هایی باشند که این بازه ها می توانند با یکدیگر نیز هم پوشانی داشته باشند که موضوع بحث ما در این پروژه می باشند. در این روش آیتم ها می توانند در بیش از یک مجموعه عضویت داشته باشند و مشکل حالات باینری را بر طرف کنند. در Fuzzy Association rule میزان درجه ی تعلق یک آیتم به وسیله ی توابع عضویت و همچنین معیارهای اندازه گیری کیفیت قوانین تولید شده به کمک تئوری مجموعه های فازی تعریف می شوند. با استفاده از این روش می توان قوانینی تولید کرد که در حالت غیر فازی ممکن است اصلاً این قوانین وجود نداشته باشند. کشف قوانین همبستگی به منزله ی یک مرحله ی بسیار مهم در فرآیند داده کاوی به شمار می روند.

## ۲- روش پیشنهادی

در قسمت قبل مفهوم قوانین همبستگی و قوانین همبستگی فازی به صورت مختصر توضیح داده شد، در این بخش قوانین همبستگی فازی را در یک دیتاست استاندارد که از سایت UCI گرفته شده با روشی که در ادامه شرح داده می شود به دست می آوریم و بعد از آن پیاده سازی روش Fuzzy Association Rule Mining در واقع به روش Fuzzy Association Rule Mining است را مشاهده می کنیم.

مراحل انجام کار در یک نگاه کلی به این صورت می باشند در ابتدا از روی داده های کمی و عددی مجموعه های فازی را به دست می آوریم

چکیده- پیدا کردن مواردی که به صورت همزمان اتفاق می افتند و با یکدیگر همکاری دارند همبستگی نام دارد. قوانین همبستگی یک کلاس بسیار مهم از قواعد در داده ها می باشند که توسط جامعه و افراد خبره در زمینه ی داده کاوی به شدت و به صورت گسترده مورد مطالعه قرار گرفته اند. هدف کلی در این زمینه پیدا کردن وقایع و آیتم های می باشد که به صورت مکرر و به صورت هم زمان در یک مجموعه انتقالات یا حالت ها رخ می دهند. اما اکثر این روابط همبستگی به صورت روابط باینری مطرح می شوند که در آن ها یک ویژگی یا مشخصه یا حضور دارد یا خیر. برای غلبه بر این مشکل روش تحلیل قوانین همبستگی به صورت فازی به وجود آمده است که در آن به آیتم ها و ویژگی ها اجازه داده می شود متعلق به بازه-هایی باشند که این بازه ها می توانند با یکدیگر نیز هم پوشانی داشته باشند.

کلمات کلیدی- درجه ی عضویت، قوانین همبستگی، قوانین همبستگی فازی، مجموعه ی فازی.

## ۱- مقدمه

تحلیل همبستگی در واقع به معنی مطالعه و شناخته ویژگی ها، صفت ها و مشخصه هایی می باشد که با یکدیگر رخ می دهند. روش ها و متدهایی که برای تحلیل همبستگی وجود دارند به عنوان روش های تحلیل سبد بازار نیز شناخته می شوند. این روش ها و متدها به دنبال آشکار کردن و پرده برداشتن از همبستگی و ارتباط بین ویژگی های مختلف می باشند که در نهایت بتوانند با توجه به این همبستگی ها قوانینی را تولید کنند که رابطه ی بین مقدار دو متغیر و یا بیشتر را بیان کنند. قوانین همبستگی که در انتها استخراج می شوند و روابط موجود بین ویژگی های مختلف را برای ما توضیح می دهند به فرم تالی  $\rightarrow$  مقدم می باشند که همچنین همراه هر قانون دو مقدار confidence و support نیز به دست می آیند که در ادامه توضیح داده می شوند.

قوانین همبستگی یک کلاس بسیار مهم از قواعد در داده ها می باشند که توسط جامعه و افراد خبره در زمینه-ی داده کاوی به شدت و به صورت گسترده مورد مطالعه قرار گرفته اند. هدف کلی در این زمینه پیدا کردن وقایع و آیتم های می باشد که به صورت مکرر و به صورت هم زمان در یک مجموعه انتقالات یا حالت ها رخ می دهند. پیدا کردن مواردی که به

دهد. روش پیشنهادی تنها برای به بدست آوردن ۳ مجموع فازی به صورت زیر می باشد:

مجموعه Low: مرز پایین برابر با مینیم مقدار ویژگی می باشد و مرز بالا به صورت زیر محاسبه می شود:

$$hb = mean - \frac{sd}{\gamma} + mean * overlap$$

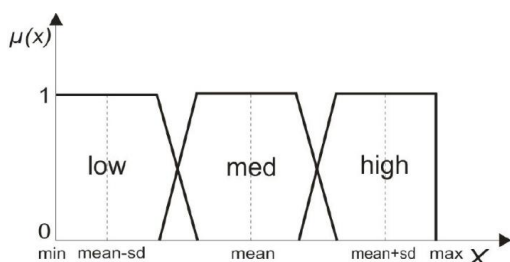
مجموعه Medium: مرز پایین و بالا صورت زیر محاسبه می شوند:

$$lb = mean - \frac{sd}{\gamma} - mean * overlap$$

$$hb = mean + \frac{sd}{\gamma} + mean * overlap$$

مجموعه High: مرز بالا برابر ماکزیمم مقدار ویژگی می باشد و مرز پایین آن به صورت زیر محاسبه می شود:

$$lb = \left( mean + \frac{sd}{\gamma} \right) - mean * overlap$$



مقدار همپوشانی (overlap) برابر هر مقداری بین ۰ تا ۱ می تواند باشد. روش ذکر شده بسیار ساده می باشد و از دقت پائینی برخوردار می باشد و تنها جهت مشاهده ی نتایج ارائه شده است و برای پروژه های واقعی نیاز به استفاده از الگوریتم های بسیار مناسب تری می باشد.

۲-۲ ساخت یک دیتاست جدید برای داده کاوی

پس از تعریف مجموعه های فازی متناظر با هر ویژگی نوبت به ساخت یک دیتاست جدید برای اجرای فرآیند Association Rule Mining از روی داده های اصلی می باشد. این مرحله به سادگی قابل اجرا و محاسبه می باشد، به این صورت که مقادیر هر ویژگی را به مجموعه های فازی متناظر با آن فیت کرده و درجه عضویت هر داده را به مجموعه ها محاسبه می کنیم، در زیر یک مثال از این مرحله را مشاهده می کنیم:

$$T = \{5, 27, 12, 17, 22, 16, 9, 14\}$$

$$A = [0, 15] \quad B = [11, 23] \quad C = [20, 30]$$

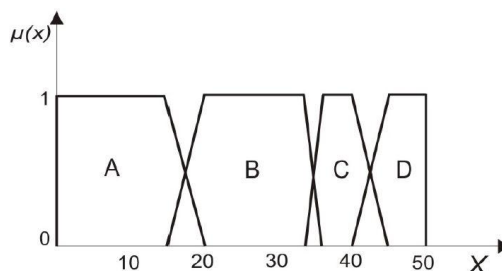
سپس یک دیتاست جهت انجام داده کاوی تولید کرده و بعد از آن عملیات فازی مورد نیاز و محاسبات فازی لازم را انجام می دهیم، در ادامه به کمک یک الگوریتم مجموعه آیتم های مکرر را تولید و به دست می آوریم و در انتها قوانین همبستگی که واقع قوانین همبستگی فازی می باشند را به دست می آوریم.

پایه سازی مورد نظر در محیط R صورت گرفته که یک محیط رایگان و بسیار مناسب برای محاسبات آماری و عملیات داده کاوی می باشد. پایه سازی صورت گرفته قابل اجرا بر روی هر پلت فرمی که زبان R بر روی آن نصب شده باشد می باشد.

۱-۲ ساخت مجموعه های فازی

در بیشتر روش های بدست آوردن قوانین همبستگی فازی که تا کنون به وجود آمده است از یک فرد خبره در زمینه ی مورد نظر برای تعریف و مشخص کردن مجموعه های فازی استفاده شده است که در آنها فرد خبره متناسب با هر ویژگی کمی و عددی مجموعه فازی آن را تعریف کرده است. در تعدادی از روش های موجود نیز تلاش شده تا با استفاده از برخی روش ها مانند خوشه بندی مجموعه های فازی متناسب با هر ویژگی به صورت اتوماتیک کشف و ساخته شوند. در پایه سازی صورت گرفته ما به دنبال این هستیم که هر دو روش را برای کاربر ممکن سازیم تا کاربر تصمیم بگیرد که آیا به صورت دستی خواهان ساخت مجموعه های فازی می باشد و یا می خواهد ساخت این مجموعه ها را به برنامه واگذار کند. در حالتی که خود کاربر تصمیم به تعریف مجموعه های فازی بگیرد او باید برای هر مجموعه مرزهایی مناسب تعریف کند. علاوه بر تعیین دقیقه محل مرزها کاربر باید تعداد مجموعه های فازی برای هر ویژگی را به دقت مشخص کند، چرا که این مجموعه فازی ها و مرزهای آنها باید به گونه ای انتخاب شود که تمام مقادیر یک ویژگی را پوشش دهند. در زیر یک نمونه از تعریف ۴ مجموعه فازی برای یک ویژگی را مشاهده می کنیم:

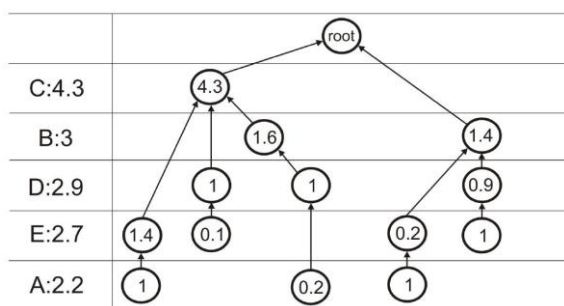
$$A = [0, 20], \quad B = [15, 36], \quad C = [34, 45], \quad D = [40, 50]$$



اگر کاربر تصمیم به تعریف مجموعه فازی ها نداشته باشد سیستم با روش پیشنهادی که در ادامه توضیح داده می شود آنها را به دست می آورد، اما این روش تضمینی برای صحیح عمل کردن به ما نمی -

درخت سخته شده را مشاهده می‌کنید. حال از این درخت برای اجرای الگوریتم FP-Growth استفاده می‌شود که در زیر به صورت مختصر آن را توضیح می‌دهیم.

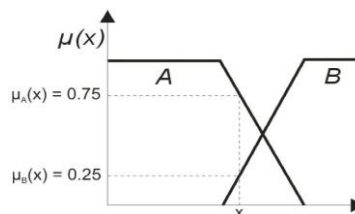
A	B	C	D	E
1	0	0.3	0	1
0	0.6	1	0	0
0.2	1	1	1	0
0	0.4	0	0.9	1
0	0	1	0	0.4
0	0	1	1	0.1
1	1	0	0	0.2



FP-growth که در اینجا استفاده می‌شود بسیار مشابه مدل حالت باینری آن می‌باشد. تفاوت آنها در این است که در اینجا برای محاسبه معیار Support، ساخت الگوهای شرطی پایه و FP-Tree شرطی جدید از مقادیر فازی استفاده می‌کند. مقدار مینیمم Support می‌تواند با مقایسه مینیمم یک مسیر در درخت با مینیمم Supportی که توسط کاربر تعریف شده است چک شود. در شکل زیر الگوهای شرطی پای برای آیتم‌های درخت بالا نشان داده شده‌اند.

item	conditional pattern base
A	$\{\langle C, E:1 \rangle, \langle C, B, D:0.2 \rangle, \langle B, E:0.2 \rangle\}$
E	$\{\langle C:1.4 \rangle, \langle C, D:0.1 \rangle, \langle B:0.2 \rangle, \langle B, D:0.9 \rangle\}$
D	$\{\langle C:1 \rangle, \langle C, B:1 \rangle, \langle B:0.9 \rangle\}$
B	$\{\langle C:1.6 \rangle\}$
C	$\emptyset$

توجه به این نکته مهم است که بر خلاف روش کلاسیک مقادیر کمتر از نودهای برگ می‌توانند در یک مسیر در درخت ظاهر شوند و این دلیل این است که ما نمی‌توانیم از مقادیر برگ‌ها برای ساخت الگوهای شرطی استفاده کنیم و به جای آن باید از مقدار مینیمم یک مسیر استفاده کنیم.



A	B	C
1	0	0
0	0	1
0.75	0.25	0
0	1	0
0	0.33	0.67
0	1	0
1	0	0
0.25	0.75	0

## ۲-۳ محاسبه ی عملگرهای فازی

در این مرحله در ابتدا باید تصمیم گرفت که از کدام t-norm برای محاسبه معیارهای فازی Confidence و Support می‌خواهیم استفاده کنیم. در روش پیشنهادی از t-norm Lukasiewicz برای این مقادیر استفاده می‌شود که فرمول آن را در زیر ارائه می‌کنیم:

$$supp(A \rightarrow B) = \sum_{(x,y) \in D} \min(A(x), B(y))$$

$$conf(A \rightarrow B) = \frac{\sum_{(x,y) \in D} \min(A(x), B(y))}{\sum_{(x,y) \in D} A(x)}$$

## ۲-۴ تولید مجموعه آیتم های مکرر

برای تولید مجموعه آیتم‌های مکرر از پایگاه داده، الگوریتم FP-Growth انتخاب شده است. تنها مشکلی که در اینجا ظاهر می‌شود تغییر الگوریتم به نحوی می‌باشد که توانایی عمل کردن بر روی پایگاه داده ی فازی را داشته باشد. خوش بختانه این تغییر برای الگوریتم مورد نظر به سادگی قابل اجرا می‌باشد چرا که داده‌های فازی به آسانی توانایی تشکیل یک FP-Tree را دارند و کاوش روی درخت ساخته شده تفاوت چندانی با الگوریتم اصلی ندارد. ساخت یک FP-Tree از روی یک پایگاه داده ی فازی کار سر راستی به حساب می‌آید. تنها تفاوت این قسمت با الگوریتم اصلی این می‌باشد که علاوه بر به حساب آوردن آیتم‌هایی که به صورت هم زمان رخ می‌دهند باید مقادیر درجه‌های عضویت نیز مد نظر قرار بگیرند. این مقادیر درجه ی عضویت در نتیجه به آسانی به مقدار کلی نودهای درخت اضافه می‌شوند. برای مثل پایگاه داده ی فازی زیر را در نظر بگیرید که از آن برای ساخت یک FP-Tree استفاده می‌شود. ساخت یک FP-Tree از این پایگاه داده منجر به درخت زیر می‌شود که شامل مجموع مقادیر فازی در ندهایش می‌باشد. در شکل زیر

تابع بعدی تابع generateMinMtrx می باشد که تمام توابع قبل را با یکدیگر ترکیب می کند و ماتریس نهایی که داده کاوی را برای ما ممکن می سازد را تولید می کند.

مرحله ی مهم بعدی ذخیره ی دیتاست جدید و مجموعه های اصلی می باشد که با استفاده از تابع MatrixIndex انجام می شود.

در این مرحله برای محاسبه ی support ما به یک مجموعه از آیت ها احتیاج داریم. تابع genSupp مقادیر مینیمم مجموعه آیت های مشخص شده در هر ردیف از پایگاه داده ی فازی را برای ما حساب می کند و همانطور که پیش از این توضیح داده شد این مقدار محاسبه شده برابر support این مجموعه آیت می باشد.

حال برای محاسبه ی confidence دیگر تنها وجود یک مجموعه داده کافی نیست و ما به یک قانون نیز نیاز داریم که شامل مقدم و تالی باشد. برای تولید یک از تابع getRule استفاده می شود. قانونی که در این مرحله تولید می شود به عنوان ورودی برای تابع genConf فرستاده می شود تا میزان confidence آن محاسبه شود. مقدار confidence با توجه به فرمولی که در بخش قبل ذکر شد برای هر قانون محاسبه می شود.

مراحل به دست آوردن مجموعه آیت های مکرر و توابع وابسته به آن به دلیل حجم زیاد و محدودیت نوشتاری در اینجا توضیح داده نمی شوند و توضیح آن ها را به صورت کامل و جامعه و شفاف به ارایه ی شفاهی واگذار می کنیم. برای به دست آوردن تمام قانون های اولیه و کاندید از مجموعه آیت های مکرر از تابع makeRules استفاده می کنیم. این تابع برای هر مجموعه آیت مکرر تمام قوانین ممکن با یک آیت تنها در قسمت تالی را تولید کرده و یک لیست برای ما برمی گرداند. لیست به دست آماده در این مرحله را به عنوان ورودی برای تابع evaluateRules در نظر می گیریم و این تابع تمام قوانینی که مقدار confidence آنها از مینیمم مقدار مشخص شده توسط کاربر بیشتر است را برای ما برمی گرداند.

و در نهایت نوبت به این می رسد که مشخص شود کدام ستون ها در پایگاه داده ی اصلی متعلق به قوانین کشف شده می باشند. این کار به وسیله ی تابع trace انجام می شود. این مرحله قوانینی تولید می کند که توسط کاربر به راحتی قابل فهم می باشند و درواقع قوانین نهایی و خروجی سیستم می باشند. (برای اجرا ی پیاده سازی صورت گرفته ابتدا باید توابع مورد نظر را در محیط R بار گذاری کنید که در سه فایل FP.r، function.r و trace.r می باشند، سپس کدهای project.r را به ترتیب برای مشاهده نتایج اجرا کنید.)

تولید قوانین همبستگی در این مرحله همان روش معمول تولید این قوانین است که به صورت کامل در کلاس شرح داده شده است. قسمت مقدم هر قانون می تواند به تعداد دلخواه آیت داشته باشد اما در قسمت تالی تنها یک آیت باید وجود داشته باشد. برای هر مجموعه آیت تک تک آیت های آن باید در قسمت تالی قرار گیرند و میزان Confidence قانون مورد نظر با مینیمم Confidence که از پیش تعیین شده مقایسه شود و در صورت بیشتر بودن Confidence قانون مورد نظر آن را در سیستم ذخیره می کنیم.

### ۳- پیاده سازی

در این بخش عملکرد پیاده سازی صورت گرفته به صورت مرحله به مرحله همراه با جزئیات شرح داده می - شود. پایگاه داده ای که در این بخش از آن استفاده می کنیم یکی از پایگاه داده های سایت UCI تحت عنوان Adultuci می باشد که شامل سه فایل عددی می باشد. قبل از اجرای پیاده سازی صورت گرفته بر روی این پایگاه داده نیاز به مقداری پیش پردازش می باشد چرا که بسیار مهم است که پایگاه داده ای که از آن برای بد- دست آوردن قوانین همبستگی فازی استفاده می کنیم فقط شامل ویژگی های عددی باشد.

اولین تابعی که در اینجا از آن استفاده می کنیم و آن را معرفی می کنیم تابع makeFuzzySets می باشد که شامل چند زیر تابع تحت عناوین getLow، getMedium، getHigh و getCenters می باشد، که همانطور که در بخش قبل ذکر شد برای هر ویژگی سه مجموعه فازی به همراه بازه ی مورد نظر برای هر مجموعه را حساب می کند و نتایج حاصل را نشان می دهد. اما برای اینکه برای تمام ویژگی ها این مجموعه فازی ها را حساب کنیم از getAllSetsgenerate استفاده می کنیم.

computeIndividualMem: این تابع مقدار درجه ی عضویت برای یک داده از دیتاست اصلی به مجموعه فازی های به دست آماده در مرحله قبل را برای ما محاسبه می کند. این تابع مقدار داده ی مورد نظر و مجموعه فازی ها را به عنوان ورودی دریافت می کند. تا اینجا ما درجه عضویت برای یک داده را محاسبه کردیم، سپس با استفاده از GetMem تمام این مقادیر درجه های عضویت محاسبه شده در مرحله ی قبل را در کنار یکدیگر می گذاریم.