

## ۱-۱) استخراج دانش

پیشرفت‌های بوجود آمده در جمع آوری داده‌ها و قابلیت‌های ذخیره‌سازی در طی دهه‌های اخیر باعث شده در بسیاری از علوم با حجم بزرگی از اطلاعات روبرو شویم. محققان در زمینه‌های مختلف مانند مهندسی، اقتصاد، ستاره‌شناسی و زیست‌شناسی هر روز با مشاهدات بیشتر و بیشتری روبرو می‌شوند. در مقایسه با بسترهاي داده‌ای قدیمی و کوچکتر، بسترهاي داده‌ای امروزی چالش‌های جدیدی در تحلیل داده‌ها بوجود آورده‌اند. روش‌های آماری سنتی به دو دلیل امروزه کارائی خود را از دست داده‌اند. علت اول افزایش تعداد مشاهدات<sup>۱</sup> است، و علت دوم که از اهمیت بالاتری برخوردار است، افزایش تعداد متغیرهای مربوط به یک مشاهده می‌باشد.

تعداد متغیرهایی که برای هر مشاهده باید اندازه‌گیری شوند، ابعاد داده‌ها نامیده می‌شود. واژه‌ی متغیر<sup>۲</sup> بیشتر در علم آمار استفاده می‌شود، در حالی که در علوم کامپیوتر و یادگیری ماشین بیشتر از واژه‌های صفت‌خاصه<sup>۳</sup> و یا ویژگی<sup>۴</sup> استفاده می‌گردد.

مراحل موجود در فرایند استخراج دانش در شکل ۱-۱ نشان داده شده است و شامل مراحل زیر است:

- جمع آوری داده‌ها<sup>۵</sup>: در این مرحله پس از پالایش داده‌ها، چندین منبع داده‌ای در یک انبار داده‌ی<sup>۶</sup> یکپارچه قرار می‌گیرند.
- انتخاب و آماده‌سازی داده‌ها: در این قسمت داده‌های مرتبط انتخاب می‌گردد و به شکل مناسبی برای داده‌کاوی تبدیل می‌شوند.
- داده‌کاوی: فرایندی که با به خدمت گرفتن روش‌های هوشمند در میان داده‌ها به دنبال الگوهای خاصی می‌گردد.
- تفسیر و ارزشیابی الگوهای از میان انبوهی از الگوها، با تعریف معیارهای متنوع الگوهای محدودی برای تفسیر و تحلیل انتخاب می‌شوند.

<sup>1</sup> Observations

<sup>2</sup> Variable

<sup>3</sup> Attribute

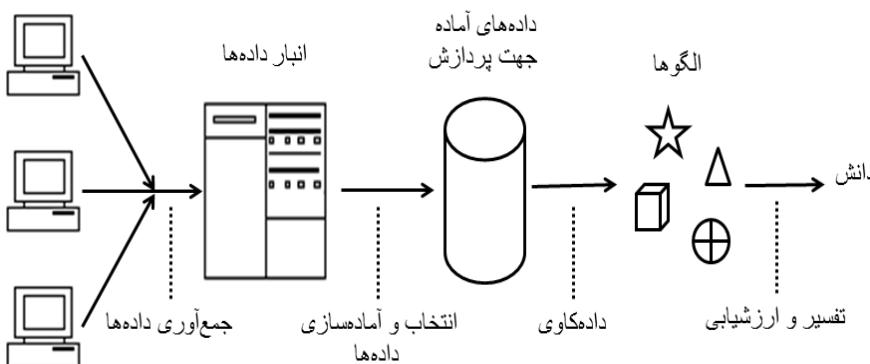
<sup>4</sup> Feature

<sup>5</sup> Integration

<sup>6</sup> Data Warehouse

- ارائه‌ی دانش: در این مرحله با کمک ابزار بصری‌سازی<sup>۱</sup> و تکنیک‌های مختلف دانش کشف شده به کاربر و یا تحلیل‌گر ارائه می‌شود.

## منابع داده‌ها



شکل ۱-۱: مراحل موجود در فرایند استخراج دانش

اغلب کمپانی‌های بزرگ دارای تعدادی شعبه هستند که هر یک از آنها حجم زیادی از داده‌ها را تولید می‌کنند. حتی برخی از سازمان‌ها با وجود تمرکز بر روی یک محل اصلی برای مستقر شدن، دارای بخش‌هایی هستند که هر یک از آنها می‌تواند دارای سیستم‌های عملیاتی مربوط به خود و در نتیجه ساختار داده‌ای خاص خود باشند. جهت تحلیل داده‌ها و در نهایت اتخاذ یک تصمیم مدیریتی لازم است اطلاعات کلیه‌ی قسمت‌ها جمع‌آوری شوند. تنظیم پرس‌وجوها بر اساس هر یک از این ساختارها کاری دشوار و ناکارآمد است. به علاوه داده‌ها عموماً توصیفی از وضعیت کنونی را در خود دارند، در حالیکه تحلیل‌گر اکثر اوقات نیاز به داده‌های قدیمی را نیز یک ضرورت می‌داند. در این وضعیت انبار داده‌ها یک راه حل مناسب تلقی می‌شود. اگرچه وجود انبار داده‌ها پیش‌نیاز داده‌کاوی نیست، ولی در کاربردهایی نظیر سازمان‌ها و شرکت‌های بزرگ با وجود داشتن یک انبار داده‌ها عمل داده‌کاوی بسیار آسان‌تر می‌شود. انتخاب مجموعه داده‌های اصلی برای تحلیل اولین ضرورت است. بسیاری از الگوریتم‌های جمع‌آوری داده‌ها فقط با پایگاه داده‌های همگن<sup>۲</sup> کار می‌کنند که این مسئله نیز در جمع‌آوری داده‌ها محدودیت محسوب می‌شود.

<sup>1</sup> Visualization<sup>2</sup> Homogeneous

## ۱-۲) انواع داده‌ها جهت داده‌کاوی

عملیات داده‌کاوی به یک نوع از داده‌ها محدود و محصور نمی‌شود و معمولاً داده‌های مختلفی توسط این سیستم‌ها پذیرفته می‌شوند. به خاطر داشته باشید تکنیک‌های متفاوتی برای نوع‌های مختلفی از داده‌ها مناسبند و یافتن یک راهکار کلی، تلاشی بیهوده به نظر

می‌رسد. تکنیک‌های داده‌کاوی را می‌توان بر روی داده‌های غیرساخت‌یافته<sup>۱</sup> (مانند متون)، نیمه‌ساخت‌یافته<sup>۲</sup> (مانند اسناد) و ساخت‌یافته<sup>۳</sup> (مانند جداول در مدل رابطه‌ای) اعمال نمود. در این قسمت به صورت خلاصه برخی از آنها را مرور می‌کنیم.

- جداول در پایگاه داده‌ای رابطه‌ای یکی از رایج‌ترین شکل‌های ورودی برای الگوریتم‌های داده‌کاوی محسوب می‌شوند. در جداول، سطراها نماینده‌ی نمونه‌ها و ستون‌ها ویژگی و صفات خاصه‌ی نمونه‌ها را تشکیل می‌دهند. اغلب روش‌های داده‌کاوی با این شکل از داده‌ها مشکلی ندارند. حتی در برخی از کاربردها کاربران ابتدا داده‌های خود را به این شکل تبدیل و پس از آن الگوریتم‌های داده‌کاوی را بر روی این شکل تبدیل یافته اجرا می‌کنند.
- اکثر روش‌های داده‌کاوی بر روی داده‌های ساخت‌یافته مانند جداول متتمرکز هستند، حال آنکه حجم وسیعی از اطلاعات در دسترس در دنیای واقعی به صورت نیمه‌ساخت‌یافته و یا غیرساخت‌یافته ذخیره شده‌اند. این پایگاه داده شامل مجموعه‌ی بزرگی از مستندات متنی مانند کتاب‌ها، مقالات و صفحات وب می‌شوند. این موضوع اهمیت استفاده از تکنیک‌های داده‌کاوی را برای این نوع از داده‌ها دوچندان کرده است. عموماً این داده‌ها نیمه‌ساخت‌یافته هستند. برای مثال یک مقاله را درنظر بگیرید. این سند شامل برخی از ویژگی‌های ساخت‌یافته مانند عنوان، نویسنده، تاریخ چاپ و... و همچنین شامل واژه‌هایی است که از هیچ ساختاری (صرف‌نظر از ساختمان یک جمله) پیروی نمی‌کنند.
- انبار داده‌ها شکل دیگری از داده‌ها تلقی می‌شوند که از آنها می‌توان به تنها یی نیز جهت تحلیل داده‌ها استفاده نمود. یک انبار داده‌ها مخزنی از اطلاعات جمع‌آوری شده از چندین منبع داده‌ای تحت یک شیمای<sup>۴</sup> واحد است. به دلیل آنکه این داده‌ها از منابع متفاوتی جمع‌آوری می‌شوند، عملیاتی چون پالایش داده‌ها، حذف نویز و داده‌های ناقص و تبدیل داده‌ها به شکل‌های مناسب برای

<sup>1</sup> Unstructured

<sup>2</sup> Semi structured

<sup>3</sup> Structured

<sup>4</sup> Schema

داده‌کاوی بر روی آن انجام می‌گردد. در مورد انبار داده‌ها در فصل سوم به تفصیل صحبت شده است.

- پایگاه داده‌ی تراکنشی<sup>۱</sup> شکل دیگری از داده‌هاست و همانطور که از نام آن مشخص است، حاوی مجموعه رکوردهایی است که هر یک از آنها دلالت بر یک تراکنش واحد همراه با اطلاعات دیگر دارد. تحلیل سبد خرید مشتریان فروشگاه‌ها نمونه‌ای بارز از این نوع از داده‌ها است که در فصل‌های چهارم و پنجم شرح داده شده است.
- بدون شک صفت‌خاصه‌ی زمان ویژگی بسیار مهمی برای مجموعه داده‌ها محسوب می‌شود. پایگاه داده‌ای که شامل صفت‌خاصه‌ی زمان است، اطلاعات مفیدتر و دقیق‌تری را تحت اختیار کاربران قرار می‌دهد. چنین پایگاه داده‌ای که حاوی رخدادهایی است که با زمان تغییر می‌کند، پایگاه داده‌ی سری‌های زمانی<sup>۲</sup> می‌نامیم. تکنیک‌های داده‌کاوی می‌توانند رفتار محتویات تراکنش‌ها را در رابطه با زمان بررسی کنند.
- امروزه وب یک مخزن داده‌ای پویا و نیز ناهمگن<sup>۳</sup> محسوب می‌شود که در آن می‌توان انواع داده‌ها از جمله متن، صدا و تصویر را یافت. وب‌کاوی پیوند تکنیک‌های داده‌کاوی با این مجموعه از داده‌ها است. کاوش در داده‌های چندرشته‌ای نیز می‌تواند به وب‌کاوی کمک کند. یک سیستم مدیریت پایگاه داده‌ی چندرشته‌ای مجموعه‌ی وسیعی از داده‌های چندرشته‌ای را ذخیره و مدیریت می‌کند. این داده‌ها می‌توانند صدا، تصویر، ویدئو، گرافیک، متن و حتی داده‌هایی مانند صفحات وب باشند. برای کاوش در میان داده‌های چندرشته‌ای، ذخیره و بازیابی موثر و سریع داده‌ها از اهمیت بالایی برخوردار است. در مورد وب‌کاوی و همچنین کاوش در میان داده‌های چند رسانه‌ای در فصل آخر اشاره‌ای خواهیم داشت.

<sup>1</sup> Transactional Database

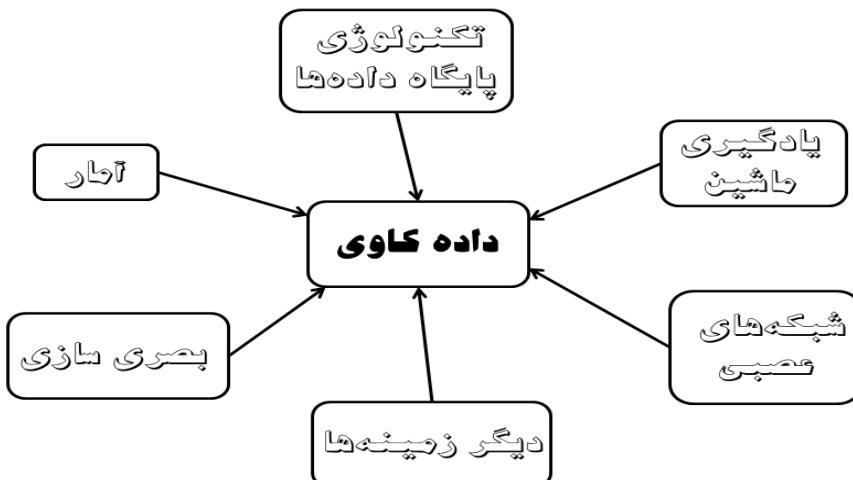
<sup>2</sup> Time Series Database

<sup>3</sup> Heterogeneous

- یک پایگاه داده‌ی مکان‌محور<sup>۱</sup> شامل مجموعه داده‌های زیادی در رابطه با مکان است. نقشه‌ها، تصاویر پزشکی و لایه‌های تراشه‌های *VLSI* نمونه‌ای از این داده‌ها به شمار می‌روند. این نوع از پایگاه داده‌ها دارای یک سری از ویژگی‌ها هستند که می‌توان آنرا از پایگاه داده‌ی نوع رابطه‌ای تشخیص داد. امروزه داده‌کاوی این نوع از پایگاه داده بطور گسترده‌ای مورد استفاده‌ی کاربران قرار می‌گیرد.

### ۱-۱) داده‌کاوی و تکنیک‌های آن

به صورت ساده اینطور می‌توان بیان کرد که داده‌کاوی به استخراج دانش از حجم انبوهی از داده‌ها اطلاق می‌شود. به همین دلیل بسیاری از افراد این واژه را مترادفی برای واژه‌ی کشف دانش می‌دانند. اما همانطور که در شکل ۱-۱ نیز مشاهده کردید، داده‌کاوی در واقع مرحله‌ای از فرایند کشف دانش تلقی می‌شود. داده‌کاوی شامل مجموعه‌ای از تکنیک‌هایی است که در حوزه‌های دیگر علمی مانند پایگاه داده‌ها، آمار، یادگیری ماشین، شبکه‌های عصبی، بازیابی اطلاعات و تشخیص الگو می‌توان آنرا یافت (شکل ۱-۲).



شکل ۱-۲: داده‌کاوی و تجمعی از زمینه‌های مختلف

<sup>۱</sup> Spatial Database

به طور معمول کلیه‌ی الگوهای تولید شده توسط الگوریتم برای کاربر مفید نیست و تنها کسر کوچکی از این الگوها می‌توانند برای تحلیل‌گر و کاربر جالب باشند و نظر آنها را جلب کنند. در این راستا سه سوال اساسی مطرح می‌شود. چه چیزی باعث آن می‌شود که ما یک الگو را جالب بدانیم؟ آیا یک سیستم داده‌کاوی قادر به تولید تمام الگوهای جالب هست؟ آیا یک سیستم داده‌کاوی می‌تواند فقط الگوهای بدردبور و جالب را تولید کند؟ برای پاسخ به سوال اول می‌توان اینطور ادعا نمود که الگوهایی جالب هستند که:

- توسط انسان به راحتی قابل فهم باشند.
- درستی آنها با درجه‌ای از قطعیت برای داده‌های جدید و آزمایشی تضمین شده باشد.
- مفید و بدیع باشند.
- برای فرضیه‌های تعریف شده توسط کاربر معتبر باشند.

اما سوال مهم این است که خصوصیاتی چون قابل فهم بودن یا سودمند و بدیع بودن الگو چگونه اندازه‌گیری می‌شود؟ در فصل‌های بعدی معیارهایی را معرفی خواهیم کرد، تا این مفاهیم را بتوان محاسبه نمود و رتبه‌ای برای الگوهای بدست آمده متصور شد.

جهت پاسخ به سوال دوم که آیا یک سیستم داده‌کاوی قادر به تولید تمام الگوها هست یا خیر، باید گفت که این موضوع که یک سیستم داده‌کاوی تمام الگوهای ممکن را تولید کند، نه کارآمد است و نه واقع بینانه. در مقابل هر کاربر به معرفی محدودیتها و معیارهایی می‌پردازد، تا الگوریتم به تولید برخی از آنها اکتفا کند. در بسیاری از موارد فضای جستجوی الگوها آنقدر وسیع است که تولید کلیه الگوها چنانچه امکان‌پذیر هم باشد، بصورت قابل توجهی زمانبند خواهد بود.

اما سوال سوم که یک مسئله‌ی بهینه‌سازی در داده‌کاوی تلقی می‌شود، که آیا سیستم می‌تواند فقط الگوهای جالب توجه کاربر را تولید کند، اگر چنین باشد که بسیار دلخواه و مطلوب است و در واقع هدف غایی کاربر این است که فقط تعداد محدود و خاصی از الگوها تحت عنوان الگوهای جالب در خروجی قرار گیرند. اما سیستم‌ها در رسیدن به این هدف با چالش‌های بسیار زیادی روبرو هستند. بدین ترتیب پس از مرحله‌ی داده‌کاوی در فرایند استخراج دانش به معیارهایی نیاز خواهیم داشت تا میان الگوهای استخراج شده یک رتبه‌بندی مناسب تشکیل دهد و از الگوهای مزاحم صرفنظر کند. اینچنین معیارهایی جهت اجرای کارآ و موثر الگوریتم‌ها نیز مفید هستند.

در فصل‌های مربوط به قوانین انجمنی (فصل‌های چهارم و پنجم) و همچنین فصل‌های مربوط به بحث خوشبندی (فصل‌های هشتم و نهم) بیشتر با این معیارها و محدودیتها آشنا می‌شویم.

تکنیک‌های متنوعی در داده‌کاوی وجود دارند که الگوهای مختلفی را تولید می‌کنند. روش‌های کشف قوانین انجمنی<sup>۱</sup>، طبقه‌بندی داده‌ها<sup>۲</sup> و خوشبندی<sup>۳</sup> از عمده‌ترین راهکارهایی محسوب می‌شوند که به تولید الگوهای خاص خود می‌پردازند. هر چند در طول کتاب به تفصیل به آنها خواهیم پرداخت، ولی در ادامه به صورت خلاصه اشاره‌ای به آنها می‌کنیم.

<sup>1</sup> Association Rules

<sup>2</sup> Classification

<sup>3</sup> Clustering

### ۱-۳-۱) قوانین انجمانی

طی سال‌های گذشته در میان تکنیک‌های داده‌کاوی توجه خاصی به الگوریتم‌های کشف الگوهای مکرر<sup>۱</sup> وجود داشته است. همانطور که از نام این الگوریتم‌ها مشخص است، به دنبال الگوهایی هستیم که به دفعات در مجموعه داده‌ها دیده می‌شوند. در این میان به الگوریتم‌های کشف مجموعه اقلام مکرر<sup>۲</sup> بیشتر پرداخته شده است که در نهایت به تولید قوانین انجمانی منجر می‌شود.

در بسیاری از کاربردها، روزانه داده‌های زیادی ذخیره می‌شود. برای مثال در یک بانک روزانه تراکنش‌های متعددی انجام می‌شود و یا اجناس خریداری شده از فروشگاه‌های زنجیره‌ای، حجم وسیعی از حافظه‌ی کامپیوتر را اشغال می‌کند. سبد خرید، مجموعه‌ای از اقلام خریداری شده بوسیله مشتری در یک تراکنش ساده است. بسیاری از مدیران فروشگاه‌ها علاقه‌مند هستند تا رفتهارهای(خریدهای) مشتریان خود را تحلیل کنند. یک تحلیل مرسوم که بر روی پایگاه داده‌ی تراکنشی انجام می‌شود، یافتن مجموعه اقلامی است که همراه با خیلی از تراکنش‌ها ظاهر می‌شوند. یک مدیر می‌تواند با اطلاع از این موضوع و با اعمال تغییراتی برای اقلام مزبور، فروش خود را بهبود بخشد.

در قوانین انجمانی وابستگی‌های مهم میان اقلام موجود در یک پایگاه داده‌ی تراکنشی را مشخص می‌کنیم، به نحوی که حضور بعضی اقلام در تراکنش‌ها بر حضور برخی اقلام دیگر در همان تراکنش‌ها دلالت دارد. برای مثال می‌خواهیم بدانیم مشتریانی که شیر می‌خرند، آیا تمایلی به خرید نان هم از خود نشان می‌دهند، یا چند درصد از مشتریان را می‌توان یافت که از دو نوع تسهیلات مانند وام مسکن و وام خرید خانه در بانک استفاده کرده‌اند. برای پاسخ به این نمونه از سوال‌ها به قوانین انجمانی نیازمندیم که یک نمونه از آن برای سوال اول بصورت زیر نمایش داده شده است:

$$\{Milk\} \rightarrow \{Bread\} \quad [Support=40\%, Confidence=66\%]$$

همانطور که مشاهده می‌کنید، مفید بودن قانون انجمانی فوق با دو معیار پشتیبان<sup>۳</sup> و اطمینان<sup>۱</sup> اندازه‌گیری شده است. تفسیر قانون انجمانی مزبور بدین گونه است که ۴۰ درصد

<sup>1</sup> Frequent Patterns Mining

<sup>2</sup> Frequent Itemsets Mining

<sup>3</sup> Support

از تراکنش‌ها حاوی شیر و نان هستند و ۶۶ درصد از مشتریانی که شیر خریده‌اند، نان را نیز در سبد خرید خود دارند.

یک قانون انجمنی با عبارت  $X \rightarrow Y$  بیان می‌شود که در آن  $X$  و  $Y$  مجموعه اقلام غیرتپهی هستند که هیچگونه اشتراکی ندارند ( $X \cap Y = \emptyset$ ). دو معیار پشتیبان و اطمینان به منظور ارزیابی قوانین انجمنی استفاده می‌شوند، هرچند معیارها فقط به این دو ختم نمی‌شوند. مقدار پشتیبان نشان می‌دهد که در چند درصد از تراکنش‌های پایگاه داده‌ها می‌توان مجموعه اقلام  $X$  و  $Y$  را همراه یکدیگر پیدا کرد و مقدار اطمینان در میان تراکنش‌هایی که مجموعه اقلام  $X$  را در خود دارند، بدبال مجموعه اقلام  $Y$  می‌گردد. این نکته ساده را فراموش نکنید که تراکنش‌های حاوی  $X$  می‌تواند شامل  $Y$  نباشد و بالعکس.

الگوریتم‌های کشف قوانین انجمنی، مستعد تولید تعداد بسیار زیادی از قوانین هستند. حتی با تعداد کم اقلام داده‌ها نیز با حجم وسیعی از قوانین روبرو هستیم. چنانچه فرض کنیم کلیه‌ی الگوها مفید هستند، برای کاربر امکان‌پذیر نیست تا قضاوت مناسبی میان آنها داشته باشد. بدین علت نیاز به الگوریتم‌های موثر جهت محدود نمودن این فضای وسیع و همچنین معیارهایی ارزیابی قوانین انجمنی به خوبی احساس می‌شود. در فصل‌های چهارم و پنجم کتاب به تفصیل در این مورد صحبت شده است.

### ۲-۳-۱) طبقه‌بندی

پایگاه داده‌ها منبع بسیار غنی از اطلاعات پنهان است که می‌توان به کمک این اطلاعات تصمیمات هوشمندی را اتخاذ نمود. در این میان طبقه‌بندی و تخمین دو شکل از تحلیل داده‌ها محسوب می‌شوند که می‌توان به کمک آنها مدلی جهت توصیف داده‌ها استخراج کرد و یا برای داده‌های بعدی جهتی متصور شد. بدین وسیله داده‌هایی با حجم بالا نیز بهتر فهمیده می‌شوند.

---

<sup>1</sup> Confidence

روش‌های نظارت شده‌ای<sup>۱</sup> مانند طبقه‌بندی و تخمین تلاش می‌کنند تا رابطه‌ی میان صفات خاصه‌ی ورودی (که گاه متغیرهای مستقل نامیده می‌شوند) را با یک یا چندین صفت خاصه‌ی هدف (که گاه متغیر وابسته نامیده می‌شوند) کشف کنند. در نهایت این رابطه با یک ساختار به عنوان مدل نمایش داده می‌شود.

با کمک این مدل و با شرط داشتن صفات خاصه‌ی ورودی می‌توانیم مقدار صفت خاصه هدف را تخمین بزیم. به عبارت دیگر با کمک مدل قادر هستیم نمونه‌ها را به یکی از چندین طبقه‌ی تعریف شده مناسب و یا مقدار تعیین شده‌ای را برای صفت خاصه هدف تعیین کنیم.

فرایند ساخت مدل یک فرایند دو مرحله‌ای است، که در مرحله‌ی اول با کمک مجموعه داده‌های آموزشی<sup>۲</sup> که برچسب کلاس تمام نمونه‌های آن مشخص است، مدل ساخته می‌شود. این مرحله به نام مرحله‌ی یادگیری شناخته می‌شود. در مرحله‌ی دوم با کمک مجموعه داده‌های آزمایشی<sup>۳</sup> که در آن معمولاً برچسب کلاس‌ها نامعلوم است، مدل بدست آمده اعتبارسنجی می‌شود. در واقع ارزشیابی مدل با توجه به اینکه کلاس چه تعداد از نمونه داده‌های آزمایشی درست تخمین زده شده است، محاسبه می‌شود.

مفاهیم اولیه موضوع طبقه‌بندی در فصل ششم بررسی می‌شوند و الگوریتم‌های طبقه‌بندی بحث فصل هفتم را تشکیل می‌دهد. در فصل ششم می‌بینیم که تکنیک‌های طبقه‌بندی چگونه ارزشیابی می‌شوند و در فصل هفتم با کمک همان روش‌ها، مزایا و محدودیت‌های الگوریتم‌های متنوع طبقه‌بندی مانند درخت‌های تصمیم را بررسی می‌کنیم.

### ۳-۱-۳) خوشبندی

فرایند گروه‌بندی مجموعه‌ای از داده‌ها و قرار دادن آنها در طبقاتی از نمونه‌های مشابه خوشبندی نام دارد. یک خوشبندی مجموعه‌ای از داده‌های است که نسبت به دیگر داده‌های همان خوشبندی شایعه بوده ولی متفاوت از نمونه‌های دیگر خوشبندی است.

<sup>1</sup> Supervised

<sup>2</sup> Training Dataset

<sup>3</sup> Test Dataset

تحلیل خوشها یکی از فعالیت‌های مهم انسانی است. در واقع انسان در کودکی می‌آموزد که چگونه بین اشیاء مختلف فرق بگذارد. این امر به دلیل افزایش مستمر طرح‌های ناخودآگاه دسته‌بندی اشیاء در ذهن اوست. تحلیل خوشها کاربردهای بسیار مختلفی از جمله تشخیص الگو، تحلیل داده‌ها، پردازش تصاویر و تحلیل‌های تجاری دارد. با این شیوه می‌توان مناطق پر جمعیت و کم‌جمعیت را شناسایی نمود و بدین ترتیب پراکنده‌گی و همبستگی‌های جالب میان خصوصیات داده‌ها را کشف نمود.

در مراحل کشف و استخراج دانش، خوشبندی می‌تواند برای پیش‌پردازش داده‌ها و یا آماده‌سازی آن نیز بکار برده شود. در ضمن بسته‌های نرم‌افزاری بسیاری همچون SPSS و SAS دارای روش‌های کلاسیکی از خوشبندی هستند که این موضوع اهمیت مسئله را در تحلیل داده‌ها بیش از پیش مشخص می‌سازد.

یک الگوریتم خوشبندی می‌تواند دارای مشخصات مطلوبی نظیر قابلیت مقیاس‌پذیری<sup>۱</sup>، توانایی مواجهه با انواع داده‌ها، استخراج خوشها به هر شکل دلخواه، توانایی مقابله با داده‌های نویز، عدم حساسیت به ترتیب ورود داده‌ها، عدم نیاز به پارامترهای ورودی، پذیرش داده‌هایی با ابعاد بالا، قابلیت یافتن خوشها به مبنای بر محدودیت و همچنین قابل فهم بودن نتایج نهایی الگوریتم باشد.

موضوع اصلی در تکنیک‌های خوشبندی تشابه و عدم تشابه دو نمونه داده است. در هر خوش نمونه‌هایی که تشابه بیشتری دارند، قرار می‌گیرند. به عبارتی دیگر قرار است تا نمونه‌های مشابه در یک خوش و نمونه‌های غیرمشابه در خوشها متفاوت گروه‌بندی شوند. بنابراین به منظور ارزیابی تشابه نیاز به یک مقیاس و یا یک معیار ضروری است. از آنجا که هر نمونه می‌تواند شامل صفات خاصه متعددی باشد و هر یک از این صفات خاصه یک نوع داده تلقی می‌شود، لذا در محاسبه یا تحلیل تشابه دو نمونه باید معیارهای تشابه برای انواع داده‌ها تعریف شوند.

در فصل‌های هشتم و نهم ما به موضوع خوشبندی می‌پردازیم. با کمک مثال‌های متنوع سعی شده تا راهکارهای مختلف خوشبندی نیز بررسی شوند، تا کاربر با توجه به کاربرد عملی خود قادر به انتخاب یکی از آنها باشد.

---

<sup>1</sup> Scalability

#### ۴-۱) چالش‌های داده‌کاوی

تنوع داده‌ها و عملیات و تکنیک‌های داده‌کاوی چالش‌های تحقیقاتی بسیاری را برای داده‌کاوی ایجاد نموده است. توسعه‌ی روش‌ها و سیستم‌های داده‌کاوی کارآ، ساخت محیط‌های داده‌کاوی مجتمع و تعاملی، طراحی زبان‌های داده‌کاوی و به کار بستن تکنیک‌های داده‌کاوی جهت حل برنامه‌های کاربردی عظیم از مهم‌ترین وظایف پژوهشگران و توسعه‌دهندگان این حوزه‌ی کاری محسوب می‌شوند. در ادامه برخی از گرایش‌های داده‌کاوی که منعکس کننده‌ی این چالش‌ها است را بررسی می‌کنیم.

- ایجاد سیستم‌های داده‌کاوی خاص یا تک منظوره. عموماً برنامه‌های موجود داده‌کاوی برای رقابت در حوزه‌ی تجارت طراحی و پیاده‌سازی شده‌اند. بدون شک حوزه‌های دیگری مانند بیوانفورماتیک بسیار مستعد استفاده از راهکارهای داده‌کاوی هستند. به همین دلیل توسعه‌ی سیستم‌های خاص جهت این کاربردها یک نیاز محسوب می‌شود.
- توسعه‌ی روش‌های مقیاس‌پذیر و تعاملی. حجم بسیار زیاد داده‌ها و همچنین نرخ رشد بالای آن باعث شده است تا خصوصیت مقیاس‌پذیر بودن الگوریتم‌ها از توجه بیشتری میان محققان برخوردار باشد. تعامل بیشتر کاربران با سیستم‌های داده‌کاوی نیز از چالش‌های این حوزه محسوب می‌شود، هر چند برخی از سیستم‌ها با پذیرفتن محدودیت‌های کاربران، خروجی الگوریتم انتخابی را به سمت دلخواه کاربران هدایت می‌کنند.
- استانداردسازی زبان‌های داده‌کاوی. وجود یک زبان استاندارد برای داده‌کاوی توسعه‌ی سیستماتیک این‌گونه سیستم‌ها را تسهیل می‌کند. در ضمن آموزش ساده‌تر و ارتباط میان چند سیستم داده‌کاوی را نیز بهبود می‌بخشد. کوشنش‌هایی در این زمینه انجام شده است که به برخی از آنها در فصل دهم اشاره‌ای شده است.
- طراحی و پیاده‌سازی روش‌های جدید برای انواع داده‌های پیچیده. مجموعه داده‌های ورودی در الگوریتم‌های حال حاضر عموماً از ساختار ساده‌ای

برخوردارند. جداول در مدل رابطه‌ای، انبار داده‌ها و پایگاه داده‌ی تراکنشی از این ساختارها به شمار می‌روند. رشد بی‌رویه‌ی داده‌ها در شکل‌ها و ساختارهای پیچیده‌تر، روش‌های داده‌کاوی را نیز تحت تأثیر خود قرار داده است. چگونگی برخورد تکنیک‌های داده‌کاوی با این نوع از داده‌های خاص مانند متون، داده‌های چندرسانه‌ای، گراف‌ها و... چالش بزرگی است.

- توسعه‌ی داده‌کاوی توزیع شده<sup>۱</sup> و بلاذرنگ<sup>۲</sup>. بسیاری از الگوریتم‌های موجود داده‌کاوی برای محیط‌های توزیع شده مناسب نیستند. این در حالی است که امروزه سیستم‌های توزیع شده بسیار محبوب و رایج هستند. داده‌کاوی پویا نیز یکی از بایدهایی است که راهکارهایی را جهت استفاده در این گونه محیط‌ها می‌طلبد.

چالش‌ها و موضوعات دیگری مانند امنیت و داده‌کاوی، داده‌کاوی بصری<sup>۳</sup>، داده‌کاوی و مهندسی نرم‌افزار و... وجود دارند که هر یک به طیف وسیعی از مفاهیم و تعاریف نیاز دارند و می‌توانند موضوع پژوهشی مناسبی برای محققین به شمار آیند.

---

<sup>1</sup> Distributed

<sup>2</sup> Real Time

<sup>3</sup> Visual Data Mining

# آماده‌سازی داده‌ها

کتابخانه‌های دیجیتال، آرشیوی از تصاویر، اطلاعات پزشکی بیماران، مجموعه داده‌های مربوط به تجارت و خرید و فروش و همچنین داده‌های علمی نمونه‌های بارزی از داده‌ها هستند که استخراج دانش از آنها بدون شک مهم است. وقتی مقیاس داده‌ها و کار بر روی آنها بالاتر از قابلیت‌های انسانی قرار می‌گیرند، نیاز به تکنولوژی‌های محاسباتی به جای تحلیل دستی و سنتی بیشتر احساس می‌شود. نکته‌ی حائز اهمیت در این میان آماده‌سازی داده‌ها<sup>۱</sup> برای یک تحلیل هوشمند است.

در این فصل قبل از هر چیز، کمی در مورد انواع داده‌ها و خصوصیت داده‌های بسیار بزرگ صحبت می‌کنیم. به صورت خلاصه، گذری بر آمار خواهیم داشت و پس از آن لزوم آماده‌سازی داده‌ها و همچنین تکنیک‌های آن بررسی می‌شوند. بدون شک عملیات مربوط به آماده‌سازی داده‌ها به تکنیک‌های شرح داده شده در این فصل محدود نمی‌شوند.

---

<sup>1</sup> Data Preparing

## ۱) انواع داده‌ها و خصوصیات آنها

بطور حتم نوع داده‌ها می‌تواند ما را در انتخاب تکنیک‌های داده‌کاوی کمک کند. صرفنظر از اینکه داده‌ها ممکن است ساخت‌یافته، نیمه‌ساخت‌یافته و یا غیرساخت‌یافته باشند، ممکن است دو گونه از داده‌ها داشته باشیم؛ داده‌های کمی<sup>۱</sup> و داده‌های کیفی<sup>۲</sup>.

### ۱-۱) متغیرهای کمی

هر گاه صفت‌خاصه و داده‌ی مورد نظر را بتوان شمارش و یا اندازه‌گیری کرد و سپس آن را به صورت عدد بیان نمود، یک متغیر کمی خواهیم داشت که به آن متغیر عددی نیز می‌گویند. در این نوع از داده‌ها ما دارای دو خاصیت ترتیب و فاصله هستیم. این متغیرها می‌توانند پیوسته<sup>۳</sup> یا گسسته<sup>۴</sup> باشند.

متغیرهای پیوسته، متغیرهایی هستند که می‌توانند کلیه‌ی مقادیر حقیقی بین محدوده‌ای را داشته باشند. وزن و قد نمونه‌هایی از این نوع هستند. کلیه‌ی عملیات ریاضی می‌تواند بر روی آنها انجام شود. نوعی از این داده‌ها مقدارشان با زمان تعییر نمی‌کند. این داده‌ها با نام داده‌های ایستا شناخته می‌شوند و از طرفی در مقابل داده‌های پویا قرار دارند که مقدارشان با زمان به روز می‌شود. اکثر روش‌های داده‌کاوی برای داده‌های ایستا مناسبند و برای داده‌های پویا روش‌های محدود و خاصی وجود دارند.

از طرفی متغیرهایی مثل تعداد فرزندان یک خانواده و یا تعداد دروس انتخاب شده توسط یک دانشجو که در اثر شمارش بدست می‌آیند، مثال‌هایی از متغیرهای گسسته یا جدا هستند. حوزه‌ی مقادیر متغیرهای گسسته مجموعه‌ای قابل شمارش و محدود است.

### ۱-۲) متغیرهای کیفی

متغیرهایی مانند جنسیت اشخاص، محل تولد، آدرس و رنگ چشم را متغیرهای کیفی می‌نامیم. حاصل متغیرهای کیفی را نمی‌توان با عدد نشان داد، بلکه بر اساس خاصیتی که

<sup>1</sup> Quantitative

<sup>2</sup> Qualitative

<sup>3</sup> Continuous

<sup>4</sup> Discrete

مورد نظر است، داده‌ها در طبقات و دسته‌های مختلفی قرار می‌گیرند. گاهی این متغیرها را داده‌های طبقه‌بندی شده<sup>۱</sup> و گسسته نیز می‌نامند. فراموش نکنید در متغیرهای کمی گسسته، ترتیب و فاصله معنی دارند، در حالیکه این دو خصوصیت برای متغیرهای کمی گسسته قابل تعریف نیست یا حداقل خصوصیت ترتیب را می‌توان به سختی و برای برخی از موارد خاص تعریف نمود. رابطه‌ای که در این نوع از داده‌ها وجود دارد، مساوی یا نامساوی بودن است. یعنی دو مقدار در این نوع از داده‌ها یا مساوی هستند و یا دارای مقادیر یکسان نیستند. برای مثال محل تولد یک شخص یا مساوی محل تولد شخص دیگری است یا خیر.

چنانچه دسته‌هایی که متغیر در آنها قرار می‌گیرد، دارای یک نوع ترتیب طبیعی باشد، آن متغیر کیفی را متغیر کیفی ترتیبی<sup>۲</sup> گویند. مجموعه‌ی مقادیر برای مدرک تحصیلی (دیپلم، کارданی، کارشناسی، کارشناسی ارشد و دکترا) نمونه‌ی بارزی از این نوع است. در داده‌های نوع ترتیبی همانطور که از نام آن مشخص است، ترتیب رعایت می‌شود ولی فاصله‌ی میان آنها معیار دقیقی نیست. رتبه‌بندی دانشجویان یک کلاس مثال مناسب دیگری است. توجه کنید که فاصله‌ی رتبه‌ی دوم با سوم بطور دقیق برابر با فاصله‌ی رتبه‌ی دهم با یازدهم نیست، هر چند این دو فقط در یک رتبه متفاوتند.

در داده‌های کیفی ترتیبی به جز رابطه‌ی مساوی بودن، رابطه‌ی کوچکتر و بزرگتر نیز معنی پیدا می‌کند. به وسیله‌ی این روابط می‌توان داده‌ها را دسته‌بندی نمود. دسته‌بندی و گروه‌بندی این نوع از داده‌ها رابطه‌ی نزدیکی با تئوری منطق فازی و دیدگاه نزدیکی با زبان محاوره دارد، لذا به عنوان روشی که ارائه‌ی آن قابل فهم و ساده برای کاربر باشد، استفاده می‌شود. بطور مثال صفت‌خاصه‌ی سن که می‌توان آنرا به رده‌های جوان، میانسال و پیر تقسیم کرد، از زاویه‌ی دید کاربر، کار بر روی آنرا از برخی جهات تسهیل می‌کند. برای برخی از داده‌ها می‌توان با تمهیداتی فاصله‌ها را دقیق‌تر بیان نمود که در بخش‌های بعدی به آن اشاره می‌شود.

---

<sup>1</sup> Categorical

<sup>2</sup> Ordinal

چنانچه دسته‌هایی که متغیر در آنها قرار می‌گیرد، دارای هیچگونه ترتیب طبیعی نباشد، هر یک از آن متغیرها را متغیر صوری یا اسمی<sup>۱</sup> کیفی می‌نامند.

در محاسبات و تجزیه و تحلیل داده‌هایی که متغیرهای کیفی در آن دخالت دارند، گاهی اوقات ساده‌تر به نظر می‌رسد که در ابتدا متغیرها کدگذاری شوند و سپس از مقادیر معرفی شده برای کدها به جای بکار بردن متغیرهای اصلی استفاده نمود. به عبارت دیگر با نسبت دادن اعداد دلخواه و مناسب به داده‌های کیفی، به گونه‌ای این داده‌ها به صورت کمی نشان داده می‌شوند. فراموش نکنید همچنان فاصله‌ی میان آنها بی‌معنی خواهد بود، غیر از اینکه کاربر با دانش کافی بر روی ماهیت داده‌ها، کدگذاری هوشمندی را تنظیم کند.

ممکن است کدگذاری به نحوی انجام شود که در پرس‌وجوهای پایگاه داده کاربر راحت‌تر عبارات را بیان کند و یا هر یک از کدها دارای بار معنایی مناسبی باشند.

---

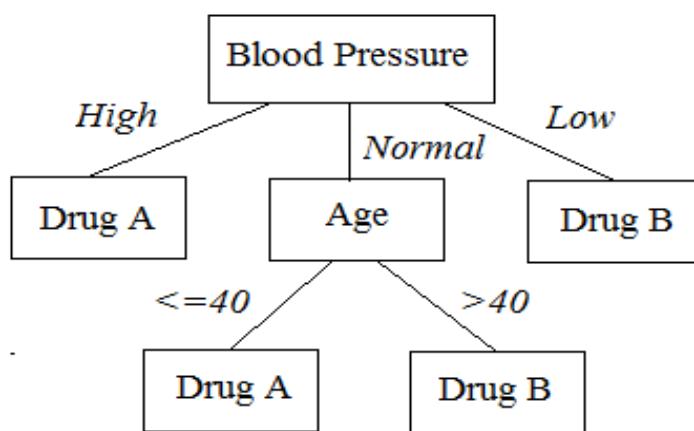
<sup>1</sup> Nominal

<sup>2</sup> Data Warehouse

# روش‌های طبقه‌بندی و تخمین

### ۷-۱) درخت‌های تصمیم

درخت تصمیم<sup>۱</sup> در داده‌کاوی مدلی است که جهت نمایش طبقه‌کننده‌ها<sup>۲</sup> و رگرسیون‌ها استفاده می‌شود. همانطور که از نام آن مشخص است، این درخت از تعدادی گره و شاخه تشکیل شده است. در درخت تصمیمی که عمل طبقه‌بندی را انجام می‌دهد، برگ‌ها بیانگر کلاس‌ها هستند. در هر یک از گره‌های دیگر(گره‌های غیربرگ) با توجه به یک یا چند صفت خاصه تصمیم‌گیری صورت می‌گیرد. شکل ۱-۷ نمونه‌ای از یک درخت تصمیم را برای مثالی از داده‌های پزشکی نشان می‌دهد.



شکل ۱-۷: درخت تصمیم برای داده‌های جدول ۱-۷

در درخت شکل ۱-۷ می‌بینیم که چگونه یک پزشک می‌تواند بر اساس صفات خاصه‌ی فشارخون و سن بیمار داروی مناسب جهت مداوای او را تجوییز کند. این مثال به خوبی نشان می‌دهد که چگونه یک درخت تصمیم برای نمایش یک مدل طبقه‌بندی استفاده می‌شود.

درخت تصمیم به دلیل سادگی و قابل فهم بودن تکنیک محبوبی در داده‌کاوی محسوب می‌شود. به عبارت دیگر درخت تصمیم خود به تنها ی همه‌ی مطالب را توصیف می‌کند و نیاز به فرد خبره‌ای نیست تا خروجی را تفسیر کند. در واقع این یک روش گرافیکی است

<sup>1</sup> Decision Tree

<sup>2</sup> Classifier

و بدین دلیل تفسیر آن شاید ساده‌تر از تکنیک‌های دیگر طبقه‌بندی باشد. اما به خاطر داشته باشید که داشتن تعداد گره‌های زیاد در درخت می‌تواند نمایش گرافیکی درخت تصمیم را با مشکل رو برو سازد. به سراغ مثالی که در شکل ۷-۱ درخت تصمیم آن رسم شده است، بر می‌گردیم. این درخت تصمیم با توجه به داده‌های جدول ۷-۱ رسم شده است. در این داده‌ها برای هر بیمار مشخصاتی چون جنسیت، سن، فشارخون و داروی تجویزی نگهداری می‌شوند.

جدول ۷-۱: مشخصات بیماران همراه با داروی تجویزی برای هر یک

| ID | Sex    | Age | Blood  | Drug |
|----|--------|-----|--------|------|
|    |        |     | P.     |      |
| 1  | Male   | 20  | Normal | A    |
| 2  | Female | 73  | Normal | B    |
| 3  | Male   | 37  | High   | A    |
| 4  | Male   | 33  | Low    | B    |
| 5  | Female | 48  | High   | A    |
| 6  | Male   | 29  | Normal | A    |
| 7  | Female | 52  | Normal | B    |
| 8  | Male   | 42  | Low    | B    |
| 9  | Male   | 61  | Normal | B    |
| 10 | Female | 30  | Normal | A    |
| 11 | Female | 26  | Low    | B    |
| 12 | Male   | 54  | High   | A    |

ریشه‌ی درخت شامل تمام ۱۲ نمونه‌ی آموزشی است که باید به کلاس‌های مختلف تقسیم شوند. هر یک از گره‌های داخلی (گره‌ای غیر از برگ) فضای نمونه را براساس یک یا چند صفت‌خاصه به چند قسمت منشعب می‌کند. در ساده‌ترین شکل ممکن گره‌های داخلی یک درخت شرطی را بر روی یک صفت‌خاصه انجام می‌دهند. چنانچه صفت‌خاصه انتخاب شده از نوع پیوسته و یا عددی باشد، شاخه‌های منشعب شده محدوده‌ای از مقدار این صفت‌خاصه را برای هر شاخه نشان می‌دهند. نمونه‌ها با طی مسیری از ریشه به برگ‌ها طبقه‌بندی می‌شوند. در شکل ۷-۱ هر گره‌ی داخلی درخت با صفت‌خاصه‌ای که

بر روی آن تست انجام می‌شود، برچسب خورده و شاخه‌های منشعب شده از آن با مقادیر خاصی از همان صفت خاصه نشان داده شده است.

برای مثال ریشه‌ی درخت صفت خاصه‌ی فشارخون را نشان می‌دهد و سه شاخه‌ای که از آن خارج می‌شود دارای سه برچسب بالا، طبیعی و پایین است. برای شاخه‌های خارج شده از گرهی داخلی سن که صفت خاصه‌ی عددی است دو محدوده‌ی کوچکتر مساوی ۴۰ سال و بزرگتر از ۴۰ سال در نظر گرفته شده است.

کاربران به صورت طبیعی ترجیح می‌دهند با درخت تصمیم کوچکی روبرو شوند، چرا که اینچنین درختی قابل فهم‌تر خواهد بود. مجموع کل گره‌ها، تعداد برگ‌ها، عمق درخت و همچنین تعداد صفات خاصه‌ای که استفاده شده‌اند، از عواملی است که برای محاسبه‌ی پیچیدگی درخت از آنها استفاده می‌شود.

ساختن یک درخت تصمیم بهینه از روی داده‌های آموزشی وظیفه‌ی ساده‌ای نیست. در برخی از مراجع نشان داده شده است که یافتن یک درخت تصمیم مینیمال سازگار با مجموعه داده‌های آموزشی یک مسئله‌ی  $NP$  سخت<sup>۱</sup> است. همچنین ابراز می‌شود که ساخت یک درخت دودوبی مینیمال با تعداد تست‌های قابل انتظار برای طبقه‌بندی یک نمونه‌ی جدید یک مسئله‌ی  $NP$  کامل<sup>۲</sup> است. حتی یافتن درخت تصمیمی مینیمال برای یک درخت تصمیم موجود و درخت تصمیم بهینه از جداول تصمیم یک مسئله‌ی  $NP$  سخت شناخته می‌شود. به این علت الگوریتم‌هایی که درخت بهینه را استفاده می‌کنند فقط برای مشکلات کوچک امکان‌پذیر خواهند بود. در نتیجه روش‌هایی که از هیوریستیک مناسبی استفاده می‌کنند، برای حل مسئله لازم به نظر می‌رسند.

برای مثال می‌توان ساخت درخت را از بالا به طرف پایین انجام داد و یا ممکن است برای داده‌هایی خاص این عمل بر عکس انجام شود. الگوریتم‌های متعددی وجود دارند که برای ساخت درخت تصمیم از روش بالا به پایین استفاده می‌کنند. برخی از روش‌ها دارای دو مرحله‌ی رشد و هرس کردن<sup>۳</sup> هستند، درحالی که برخی دیگر فقط شامل مرحله‌ی رشد هستند. فرض کنید می‌خواهیم یک درخت تصمیم با کمک روش بالا به پایین ایجاد

<sup>1</sup> Hard

<sup>2</sup> Complete

<sup>3</sup> Prunning

کنیم. دو موضوع اساسی در تولید این درخت مطرح هستند. اول اینکه چگونه مناسب‌ترین صفت خاصه برای هر گره انتخاب شود و مسئله‌ی دوم اینکه شرط پایان الگوریتم چیست. برای مثال به درخت تصمیم ساده‌ای که در شکل ۱-۷ نشان داده شده است و همچنین داده‌های آموزشی مربوط به این درخت (جدول ۷-۱) توجه کنید.

به نظر می‌رسد که از میان جنسیت، فشار خون و سن بیمار در ابتدا و در ریشه‌ی درخت فشار خون به عنوان با اهمیت‌ترین صفت خاصه انتخاب شده است. حتی با نگاهی دقیق‌تر متوجه می‌شویم که جنسیت بیمار در تجویز دارو نقشی بازی نمی‌کند؛ چون در درخت شما نامی از جنسیت در گره‌های داخلی نمی‌بینید. حداقل می‌توان به این شکل بیان کرد که ارزش فشار خون و سن برای تشخیص نوع داروی  $A$  و  $B$  بیشتر از جنسیت بیمار است. اینکه چه صفت خاصه‌ای انتخاب می‌شود در بخش بعدی چندین معیار جهت انجام این کار پیشنهاد می‌کنیم. اما در مورد شرط پایان الگوریتم می‌توان گفت که رشد درخت ادامه پیدا می‌کند تا یکی از شروط توقف زیر محقق شوند:

- همه‌ی نمونه‌های باقیمانده از مجموعه‌ی آموزشی متعلق به یک کلاس باشند.
- به حداقل عمق درخت رسیده باشیم. این حداقل توسط کاربر مشخص می‌شود.
- تعداد نمونه‌های گره از حداقل تعدادی که کاربر مشخص کرده است، کمتر باشد.
- در صورت انشعاب، تعداد نمونه‌ها در یک یا چند گره‌ی فرزند کمتر از حداقل نمونه‌هایی است که برای هر گره (فرزنده) تعریف شده است.
- مقادیر محاسبه شده برای انتخاب صفت خاصه برای هیچیک از صفات خاصه از مقدار آستانه‌ی آن بیشتر نیست.

در نهایت درخت تصمیم با معیارهایی که در فصل قبل آنرا شرح دادیم همانند دقت یا نرخ خطای مدل ارزشیابی می‌شود. فراموش نمی‌کنیم که برای هر روش طبقه‌بندی معیارهای ارزشیابی مطابق با مفاهیم موجود در آن روش نیز وجود دارند. برای مثال در یک درخت تصمیم به دلیل ماهیت درخت بودن معیارهایی چون تعداد گره‌ها و عمق درخت نیز ممکن است به عنوان معیارهایی جهت ارزیابی استفاده شوند.

### ۷-۱) معیارهای انتخاب صفت خاصه

اگل هر گرهی داخلی در درخت تصمیم بر اساس مقدار یک صفت خاصه منشعب می‌شود، در نتیجه الگوریتم به دنبال بهترین انتخاب خود در میان صفات خاصه می‌گردد. البته روش‌هایی نیز وجود دارند که با کمک آنها در گره‌های داخلی درخت می‌توانیم چندین شرط بر روی چندین صفت خاصه داشته باشیم که به دلیل پیچیدگی در این مجال بررسی نمی‌شوند. در ادامه چند معیار رایج برای انتخاب صفت خاصه برتر بیان شده است.

### معیار *Information Gain*

این معیار یکی از معروف‌ترین معیارهایی است که برای ساخت درخت تصمیم از آن استفاده می‌شود و خود از معیار دیگری به نام آنتروپی<sup>۱</sup> استفاده می‌کند.

$$\text{InformationGain}(A) = \text{Entropy}(D) - \text{Entropy}_A(D)$$

این فرمول *InformationGain* را برای صفت خاصه‌ی  $A$  محاسبه می‌کند که در آن  $D$  دلالت بر مجموعه داده‌های آموزشی دارد و داریم:

$$\text{Entropy}(D) = -\sum_{i=1}^c P_i \times \log_2(P_i)$$

$$\text{Entropy}_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times \text{Entropy}(D_j)$$

که در آن  $c$  تعداد برچسب کلاس‌های موجود در داده‌های آموزشی،  $P_i$  احتمال اینکه نمونه‌ای از داده‌ها متعلق به کلاس  $i$  نام باشد،  $v$  تعداد اعضای دامنه‌ی صفت خاصه‌ی  $A$  و  $D_j$  قسمتی از داده‌های اولیه که مقدار صفت خاصه‌ی آنها  $v_j$  است را نشان می‌دهند. در ضمن  $|D_j|/|D|$  دلالت بر اندازه‌ی داده‌های  $D$  دارد.

جدول ۷-۲ را در نظر بگیرید. در این جدول مشخصات ۱۰ نفر درج شده است که بعضی از آنها دارای کامپیوتر هستند.

---

<sup>۱</sup> Entropy

جدول ۷-۲: نمونه‌ای از یک داده‌ی آزمایشی

| ID | Age    | Income | Job     | Computer |
|----|--------|--------|---------|----------|
| 1  | Old    | Medium | Student | No       |
| 2  | Middle | High   | Teacher | No       |
| 3  | Old    | Low    | Teacher | No       |
| 4  | Young  | Medium | Teacher | Yes      |
| 5  | Young  | Low    | Teacher | Yes      |
| 6  | Old    | Medium | Student | Yes      |
| 7  | Middle | Medium | Student | Yes      |
| 8  | Young  | High   | Teacher | No       |
| 9  | Old    | High   | Student | No       |
| 10 | Middle | High   | Student | No       |

سن، درآمد و شغل افراد در داشتن کامپیوتر شخصی موثر است. با فرض اینکه بدانیم صفت خاصه‌ی *Computer* برچسب کلاس را تشکیل می‌دهد، می‌خواهیم درخت تصمیمی را برای داده‌های مذبور ایجاد کنیم.  
از آنجا که از ۱۰ نمونه‌ی موجود در داده‌ها ۴ نمونه دارای برچسب *Yes* و ۶ نمونه‌ی دیگر دارای برچسب *No* هستند پس داریم:

$$\text{Entropy}(D) = -\frac{4}{10} \text{Log}_2(\frac{4}{10}) - \frac{6}{10} \text{Log}_2(\frac{6}{10}) = 0.970$$

دامنه‌ی (در واقع مقادیر موجود جاری) هر یک از ۳ صفت خاصه‌ی سن، درآمد و شغل عبارت است از:

$$\text{Domain}(Age) = \{\text{Old}, \text{Middle}, \text{Young}\}$$

$$\text{Domain}(Income) = \{\text{High}, \text{Medium}, \text{Low}\}$$

$$\text{Domain}(Job) = \{\text{Teacher}, \text{Student}\}$$

سپس باید برای ۳ صفت خاصه‌ی سن، درآمد و شغل مقدار آنتروپی را محاسبه کنیم.

$$\begin{aligned} Entropy_{Age}(D) &= \frac{4}{10} \times \left( -\frac{3}{4} \log_2(\frac{3}{4}) - \frac{1}{4} \log_2(\frac{1}{4}) \right) + \\ &\quad \frac{3}{10} \times \left( -\frac{2}{3} \log_2(\frac{2}{3}) - \frac{1}{3} \log_2(\frac{1}{3}) \right) + \\ &\quad \frac{3}{10} \times \left( -\frac{1}{3} \log_2(\frac{1}{3}) - \frac{2}{3} \log_2(\frac{2}{3}) \right) = 0.875 \end{aligned}$$

$$\begin{aligned} Entropy_{Income}(D) &= \frac{4}{10} \times \left( -\frac{4}{4} \log_2(\frac{4}{4}) - \frac{0}{4} \log_2(\frac{0}{4}) \right) + \\ &\quad \frac{4}{10} \times \left( -\frac{1}{4} \log_2(\frac{1}{4}) - \frac{3}{4} \log_2(\frac{3}{4}) \right) + \\ &\quad \frac{2}{10} \times \left( -\frac{1}{2} \log_2(\frac{1}{2}) - \frac{1}{2} \log_2(\frac{1}{2}) \right) = 0.524 \end{aligned}$$

$$\begin{aligned} Entropy_{Job}(D) &= \frac{5}{10} \times \left( -\frac{3}{5} \log_2(\frac{3}{5}) - \frac{2}{5} \log_2(\frac{2}{5}) \right) + \\ &\quad \frac{5}{10} \times \left( -\frac{3}{5} \log_2(\frac{3}{5}) - \frac{2}{5} \log_2(\frac{2}{5}) \right) = 0.970 \end{aligned}$$

پس از آن مقدار *InformationGain* را برای کلیه‌ی صفات خاصه محاسبه می‌کنیم.

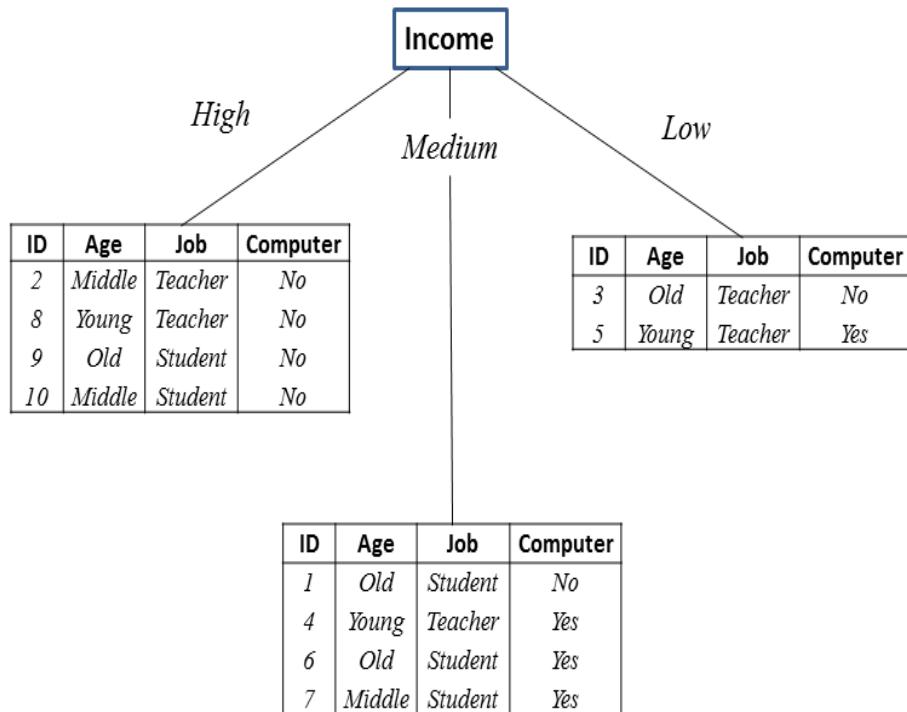
$$InformationGain(Age) = 0.970 - 0.875 = 0.095$$

$$InformationGain(Income) = 0.970 - 0.524 = 0.446$$

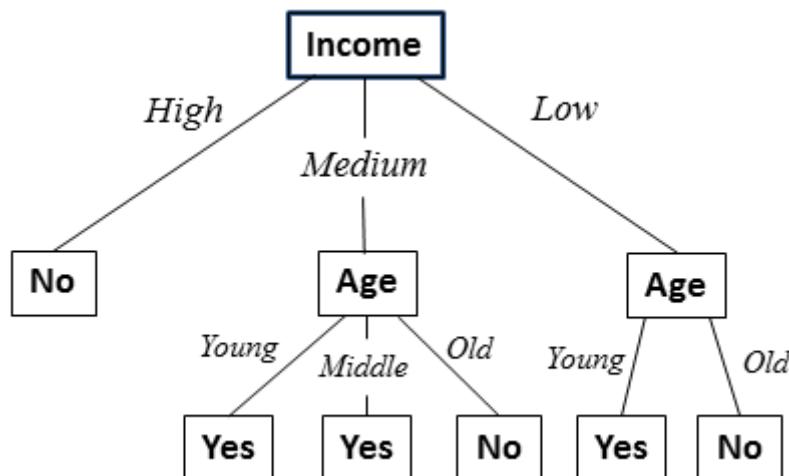
$$InformationGain(Job) = 0.970 - 0.970 = 0$$

به دلیل اینکه صفت خاصه‌ی درآمد دارای بیشترین مقدار است، برای ریشه‌ی درخت انتخاب می‌شود. چون در دامنه‌ی این صفت خاصه می‌توان ۳ مقدار متمایز یافت، بنابراین سه شاخه از این گره منشعب می‌شود که هر یک با مقادیر سه گانه‌ی این صفت خاصه برچسب خورده‌اند (شکل ۷-۲).

درواقع داده‌های آموزشی به ۳ زیرمجموعه افزایش می‌شوند. برای هر یک از این زیرمجموعه‌ها همانند قبل بهترین صفت خاصه جهت انشعاب محاسبه می‌شود. این بار صفت خاصه‌ی درآمد در محاسبات شرکت نمی‌کند و از میان دیگر صفات یکی انتخاب می‌شود. این کار تا هنگامی که یکی از شروط توقف الگوریتم محقق شود، ادامه می‌یابد. درخت تصمیم نهایی برای داده‌های جدول ۷-۲ در شکل ۷-۳ نشان داده شده است.



شکل ۷-۲: درخت حاصل از مرحله‌ی اول با انتخاب صفت خاصه‌ی درآمد



شکل ۷-۳: درخت نهایی برای داده‌های جدول ۷-۲

درخت تصمیم رسم شده در شکل ۷-۳ برای ۹ نمونه از ۱۰ نمونه‌ی موجود در داده‌های آموزشی تخمینی صحیح دارد. تنها برای نمونه‌ی شماره‌ی ۶ این مدل به نادرست کلاسی معادل *No* خواهد داشت، در صورتی که می‌بایست در طبقه‌بندی *Yes* قرار گیرد. بنابراین دقت این مدل برابر با  $\frac{8}{10} = 0.8$  است و نرخ خطای آن برابر با  $\frac{1}{10} = 0.1$  خواهد بود. توجه کنید که این درخت برای فرد میانسالی (*Age=Middle*) که درآمد کمی (Income=Low) دارد، هیچ کلاسی را تخمین نمی‌زند.

درخت تصمیم شکل ۷-۳ را می‌توان با کمک مجموعه‌ای از شروط یا قوانین نیز نمایش داد. تعداد این شروط برابر با تعداد مسیرهایی است که از ریشه تا گره‌های برگ درخت طی می‌شوند. شکل ۷-۴ این مجموعه شروط را برای درخت تصمیم مذبور نشان می‌دهد. ترتیب اجرای قوانین در این مجموعه و در این حالت خاص اهمیت ندارد. این نکته به این دلیل حائز اهمیت است که در برخی از روش‌ها این ترتیب مهم است.

*If Income=high Then No*  
*If Income=medium and Age=young Then Yes*  
*If Income=medium and Age=middle Then Yes*  
*If Income=medium and Age=old Then No*  
*If Income=low and Age=young Then Yes*  
*If Income=low and Age=old Then No*

شكل ۷-۴: شروط منتج از درخت رسم شده در شکل ۷-۳

برای تخمین نمونه‌ی جدید کافی است از ریشه‌ی درخت مسیری را با توجه به مقادیر این نمونه دنبال کنیم و یا اینکه با کمک یکی از قوانین *If* در شکل ۷-۴ برچسب کلاس تخمین زده شود.

اما چگونه مقدار *InformationGain* برای یک صفت‌خاصه‌ی عددی (پیوسته) محاسبه می‌شود؟ برای مثال فرض کنید به جای سه مقدار جوان، میانسال و پیر برای صفت‌خاصه‌ی سن، مقدار عددی سن هر شخص در پایگاه داده‌ها نگهداری می‌شد. در

چنین وضعیتی یکی از راهکارها پیدا کردن حداقل یک نقطه‌ی انفصال برای جداسازی مقادیر صفت‌خاصه‌ی عددی است.

فرض کنید صفت‌خاصه‌ی  $Att$  عددی (پیوسته) است و دارای  $n$  مقدار متفاوت در مجموعه داده‌های آموزشی است. پس از مرتب‌سازی این  $n$  مقدار، برای ( $n-1$ ) حالت جداسازی آنتروپی صفت‌خاصه محاسبه می‌شود. توجه کنید که در هر حالت داده‌ها به دو بخش تقسیم می‌شوند. حالتی که کوچکترین مقدار آنتروپی را دارد، به عنوان نقطه‌ی انفصال برگزیده می‌شود. بنابراین در حین ساخت درخت تصمیم در صورت انتخاب صفت‌خاصه‌ی عددی، در ساده‌ترین حالت دو شاخه از گره منشعب خواهد شد. چنانچه مایل باشید صفت‌خاصه‌ی مزبور به گروه‌های زیادتری شکسته شود، می‌توانید هر گروه را با همین روش به دسته‌های کوچکتر منشعب کنید.

شكل ۷-۵ چگونگی یافتن نقطه‌ی انفصال را برای صفت‌خاصه‌ی  $Age$  نشان می‌دهد. با توجه به اعداد نتیجه می‌شود که صفت‌خاصه‌ی سن بهتر است به دو دسته‌ی بالای ۱۵ سال و کمتر مساوی ۱۵ سال تقسیم شود. بدین ترتیب متغیر عددی سن به داده‌ی گسسته‌ای با دو مقدار تبدیل خواهد شد.

| Age | Class |                            |                 |  |
|-----|-------|----------------------------|-----------------|--|
| 15  | A     |                            |                 |  |
| 20  | B     | $Age=\{12,15,20,27\}$      |                 |  |
| 20  | B     | $Age \leq 12$ , $Age > 12$ | $Entropy=0.755$ |  |
| 12  | A     | $Age \leq 15$ , $Age > 15$ | $Entropy=0.405$ |  |
| 27  | B     | $Age \leq 20$ , $Age > 20$ | $Entropy=0.862$ |  |
| 15  | B     |                            |                 |  |
| 20  | B     |                            |                 |  |
| 15  | A     |                            |                 |  |

شكل ۷-۵: محاسبه‌ی آنتروپی برای نقاط انفصال گوناگون صفت‌خاصه‌ی سن

روش‌های متعدد دیگری نیز وجود دارند که می‌توان با کمک آنها متغیرهای پیوسته را به نوع گسسته تبدیل نمود که در فصل مربوط به پیش‌پردازش داده‌ها برخی از آنها توضیح داده شده است.

### معیار *Gini Index*

جهت محاسبه‌ی این معیار برای داده‌های  $D$  از فرمول زیر استفاده می‌کنیم.

$$Gini(D) = 1 - \sum_{i=1}^c P_i^2$$

که در آن  $c$  تعداد کلاس‌های موجود در داده‌ها و  $P_i$  احتمال تعلق نمونه‌ای از داده‌ها به کلاس  $i$  را نشان می‌دهند. این معیار در درخت یک انشعاب دودویی را برای هر یک از صفات خاصه ایجاد می‌کند. اگر مجموعه داده‌ی  $D$  برای صفت خاصه‌ی  $A$  به دو

زیرمجموعه‌ی  $D_1$  و  $D_2$  تقسیم شود، داریم:

$$Gini_A(D) = \frac{|D_1|}{|D|} \times Gini(D_1) + \frac{|D_2|}{|D|} \times Gini(D_2)$$

برای هر یک از صفات خاصه، تمام حالت‌های دسته‌بندی دودویی در نظر گرفته می‌شوند. پس از محاسبه‌ی *Gini Index* برای همه‌ی حالات، مقدار حداقل انتخاب می‌شود. به عبارت دیگر در نهایت از میان صفات خاصه، هر کدام که مقدار *Gini Index* آن کوچک‌تر است، برای گرهی جاری درخت تصمیم در نظر گرفته می‌شود. می‌توان درجه‌ی ناخالصی را از فرمول زیر محاسبه و هر صفتی که آن را به حداقل برساند، برگزید.

$$Gini(A) = Gini(D) - Gini_A(D)$$

جدول ۷-۳ اطلاعات مشتریانی که درخواست وام داشتند را نشان می‌دهد.

جدول ۷-۴: اطلاعاتی جهت تصمیم‌گیری برای اعطای وام

| ID | Age    | Job   | House | Credit    | Class |
|----|--------|-------|-------|-----------|-------|
| 1  | Old    | False | True  | Excellent | Yes   |
| 2  | Old    | False | True  | Good      | Yes   |
| 3  | Middle | False | False | Fair      | No    |
| 4  | Middle | True  | True  | Good      | Yes   |
| 5  | Young  | False | False | Fair      | No    |
| 6  | Old    | False | False | Fair      | No    |
| 7  | Middle | False | True  | Excellent | Yes   |
| 8  | Young  | True  | False | Good      | Yes   |
| 9  | Young  | True  | True  | Fair      | Yes   |
| 10 | Middle | False | False | Good      | No    |

سن، داشتن شغل و منزل مسکونی و همچنین اعتبار مشتریان در تصمیم‌گیری برای اعطای وام موثر هستند. از آنجا که ۱۰ نمونه داده به دو کلاس *No* و *Yes* به ترتیب به نسبت ۶ و ۴ توزیع شده است، داریم:

$$Gini(D) = 1 - \left(\frac{6}{10}\right)^2 - \left(\frac{4}{10}\right)^2 = 0.48$$

جهت یافتن بهترین صفت خاصه، معیار *Gini Index* برای کلیه‌ی صفات خاصه محاسبه می‌شود. در مجموعه دامنه‌ی دو صفت خاصه‌ی شغل و منزل مسکونی می‌توان دو مقدار *True* و *False* را یافت و به همین دلیل محاسبه‌ی معیار خیلی سخت نیست.

$$Gini_{Job}(D) = \frac{7}{10} \times \left(1 - \left(\frac{4}{7}\right)^2 - \left(\frac{3}{7}\right)^2\right) + \frac{3}{10} \times \left(1 - \left(\frac{0}{3}\right)^2 - \left(\frac{3}{3}\right)^2\right) = 0.343$$

$$Gini_{House}(D) = \frac{5}{10} \times \left(1 - \left(\frac{4}{5}\right)^2 - \left(\frac{1}{5}\right)^2\right) + \frac{5}{10} \times \left(1 - \left(\frac{0}{5}\right)^2 - \left(\frac{5}{5}\right)^2\right) = 0.160$$

اما صفات خاصه‌ی سن و اعتبار هر یک دارای سه مقدار هستند و از آنجا که معیار *Gini Index* یک انشعاب دودویی را برای هر یک از این صفات خاصه می‌سازد، باید کلیه‌ی حالات تقسیم مقادیر برای این صفات خاصه بررسی شوند.

$$Gini_{Age}(D)=0.476 \quad \text{وقتی} \quad \{Old\}, \{Middle, Young\}$$

$$Gini_{Age}(D)=\underline{0.467} \quad \text{وقتی} \quad \{Middle\}, \{Old, Young\}$$

$$Gini_{Age}(D)=0.476 \quad \text{وقتی} \quad \{Young\}, \{Middle, Old\}$$

$$Gini_{Credit}(D)=0.400 \quad \text{وقتی} \quad \{Excellent\}, \{Good, Fair\}$$

$$Gini_{Credit}(D)=0.450 \quad \text{وقتی} \quad \{Good\}, \{Excellent, Fair\}$$

$$Gini_{Credit}(D)=\underline{0.317} \quad \text{وقتی} \quad \{Fair\}, \{Excellent, Good\}$$

همانطور که ملاحظه می‌کنید بهترین انشعاب برای صفت خاصه‌ی سن هنگامی است که مقادیر پیر و جوان در یک گروه و میانسال در گروه دیگر قرار می‌گیرند و مقدار *Gini Index* برابر با  $\frac{467}{467} = 1$  است. بطور مشابه مقدار این معیار برای صفت خاصه‌ی اعتبار برابر با  $\frac{317}{317} = 1$  می‌باشد. در مرحله‌ی اول از میان صفات خاصه، منزل مسکونی با مقدار

انتخاب می‌شود. با تکرار عملیات فوق در نهایت ما با یک درخت تصمیم دودویی روبرو خواهیم بود.

### **معیار *Gain Ratio***

این معیار در واقع معیار *Information Gain* را نرمال‌سازی می‌کند و به صورت زیر بیان می‌شود:

$$GainRatio_A(D) = \frac{InformationGain(A)}{Entropy_A(D)}$$

در صورتی که مخرج کسر مقدار صفر داشته باشد، این معیار قابل تعریف نیست. معیارهای قبلی به طرف صفات خاصه‌ای با مقادیر دامنه‌ی بزرگتر گرایش دارند. به عبارت دیگر این معیارها صفات خاصه‌ای با مقدار زیاد را به صفات خاصه با مقدار کم ترجیح خواهند داد. به همین دلیل نرمال‌سازی این معیارها مفید به نظر می‌رسد. می‌توان نشان داد *Information Gain* در مقایسه با *Gain Ratio* عملکرد بهتری را در دقت و پیچیدگی مدل از خود نشان می‌دهد. یافتن نقطه‌ی انفصل برای مجموعه داده‌های پیوسته‌ای (عددی) که تعداد زیادی مقادیر مجزا دارند، یکی از نقطه‌های تاریک این معیار به حساب می‌آید که این مسئله در محاسبه‌ی *Information Gain* نیز بی‌تأثیر نیست.

## ۷-۱) چند الگوریتم درخت تصمیم

الگوریتم‌های متعددی برای ساخت درخت تصمیم وجود دارند که در این بخش به برخی از معروف‌ترین آنها اشاره می‌شود.

### الگوریتم ID3

این الگوریتم یکی از ساده‌ترین الگوریتم‌های درخت تصمیم است که از معیار استفاده می‌کند. در اجرای این الگوریتم دو شرط توقف وجود دارد. یکی این که کلیه نمونه‌های باقیمانده متعلق به یک کلاس باشند و یا اینکه پس از محاسبه‌ی مقدار معیار *Information Gain* بهترین آن بزرگ‌تر از صفر نباشد. هیچگونه روش هرس کردنی در آن موجود نیست و می‌تواند صفات خاصه‌ی عددی و داده‌های ناقص را به عنوان ورودی بپذیرد.

### الگوریتم C4.5

این الگوریتم یکی از تعمیم‌های الگوریتم ID3 است که از معیار *Gain Ratio* جهت انتخاب صفت خاصه استفاده می‌کند. الگوریتم هنگامی متوقف می‌شود که تعداد نمونه‌ها کمتر از مقدار مشخص شده‌ای باشد. این الگوریتم از تکنیک پس هرس استفاده می‌کند و همانند الگوریتم قبلی داده‌های عددی را نیز می‌پذیرد. با کمی تغییر هم می‌توان برای داده‌های ناقص از آن استفاده کرد.

### CART<sup>۱</sup> الگوریتم

نتیجه این الگوریتم یک درخت تصمیم دودویی است. بدین معنی که هر گرهی داخلی بطور دقیق دارای دو انشعاب است. از معیار *Twoing* استفاده می‌کند و روشی را نیز برای هرس کردن دارد. یکی از ویژگی‌های مهم *CART* توانایی تولید درختان رگرسیون است. برگ‌ها در چنین درختی یک عدد واقعی را به جای برچسب کلاس تخمین می‌زنند.

### ۷- طبقه‌بندی با کمک قانون بیز

یکی از فرمول‌های مهم احتمال، فرمول احتمال بیز<sup>۲</sup> است که می‌توانیم به کمک آن احتمال برچسب کلاس یک نمونه از داده‌ها را تخمین بزنیم. استفاده از این قانون برای طبقه‌بندی، دقت و سرعت خوبی را در پایگاه داده‌های بزرگ به همراه دارد. در این روش فرض بر این است که تأثیر مقدار یک صفت‌خاصه بر روی برچسب کلاس مستقل از مقادیر دیگر صفات‌خاصه است و این موضوع استقلال شرطی<sup>۳</sup> کلاس نامیده می‌شود. این فرض همانطور که در ادامه خواهیم دید باعث ساده‌تر شدن محاسبات می‌شود.

<sup>1</sup> Classification Regression Trees

<sup>2</sup> Chi-squared Automatic Interaction Detection

<sup>3</sup> Bayesian Theorem

<sup>4</sup> Conditional Independence

فرض کنید  $C$  نام صفت خاصه‌ی کلاسی با  $m$  مقدار متمایز در مجموعه داده‌های آموزشی  $D$  باشد. جهت تخمین برچسب نمونه‌ای مانند  $d$  کلیه‌ی احتمالات شرطی  $P(C=c_i/d)$  محاسبه و بیشترین احتمال، برچسب کلاس  $d$  را تعیین می‌کند. اگر مجموعه داده‌ی  $D$  دارای  $n$  صفت خاصه باشد،  $d$  می‌تواند به صورت زیر بیان شود:

$$d = \langle A_1 = a_1, A_2 = a_2, \dots, A_n = a_n \rangle$$

و قانون بیز می‌تواند به صورت زیر نشان داده شود:

$$\begin{aligned} P(C = c_i | A_1 = a_1, \dots, A_n = a_n) &= \frac{P(A_1 = a_1, \dots, A_n = a_n | C = c_i) \times P(C = c_i)}{P(A_1 = a_1, \dots, A_n = a_n)} \\ &= \frac{P(A_1 = a_1, \dots, A_n = a_n | C = c_i) \times P(C = c_i)}{\sum_{k=1}^m P(A_1 = a_1, \dots, A_n = a_n | C = c_k) \times P(C = c_k)} \end{aligned}$$

مقدار  $P(C = c_i)$  را می‌توان از روی مجموعه داده‌های آموزشی  $D$  بمحاسبه کسری از داده‌ها که دارای برچسب کلاس  $c_i$  هستند، محاسبه نمود. مقدار احتمال زیر نیز برای همه کلاس‌ها یکسان است و بنابراین تأثیری در تصمیم‌گیری ندارد.

$$P(A_1 = a_1, \dots, A_n = a_n)$$

و برای احتمالی که در صورت دو کسر در فرمول بیز قرار دارد، می‌توان نوشت:

$$P(A_1 = a_1, \dots, A_n = a_n | C = c_i) =$$

$$P(A_1 = a_1 | A_2 = a_2, \dots, A_n = a_n, C = c_i) \times P(A_2 = a_2, \dots, A_n = a_n | C = c_i)$$

و به دلیل استقلال شرطی کلاس داریم:

$$P(A_1 = a_1 | A_2 = a_2, \dots, A_n = a_n, C = c_i) = P(A_1 = a_1 | C = c_i)$$

با جایگذاری بازگشتی برای فرمول و با آگاهی به اینکه فرمول برای همه‌ی صفات خاصه برقرار است، داریم:

$$P(A_1 = a_1, \dots, A_n = a_n | C = c_i) = \prod_{j=1}^n P(A_j = a_j | C = c_i)$$

که در آن  $P(A_j = a_j | C = c_j)$  از تقسیم تعداد نمونه‌هایی که هر دو شرط تساوی  $C = c_j$  و  $A_j = a_j$  را برآورده می‌کنند بر تعداد کل نمونه‌هایی که دارای برچسب کلاس  $c_j$  هستند، بدست می‌آید.

بالاخره با ترکیب فرمول‌های مذبور عبارت نهایی حاصل می‌شود. اما از آنجا که مایلیم محتمل‌ترین کلاس (بزرگ‌ترین مقدار برای فرمول از میان کلاس‌های موجود) را برای نمونه‌ای از داده‌های آزمایشی تخمین بزنیم و با توجه به اینکه مقدار مخرج فرمول بیز برای همه‌ی کلاس‌ها یکسان است، فقط کافی است صورت کسر محاسبه شود. بنابراین کافی است برای هر یک از کلاس‌های موجود فرمول زیر محاسبه و با توجه به بیشترین مقدار، کلاس نمونه‌ی آزمایشی تخمین زده شود.

$$P(C = c_i) \times \prod_{j=1}^n P(A_j = a_j | C = c_i)$$

اجازه دهید برای فهم بهتر موضوع کارمان را با یک مثال ادامه دهیم. جدول ۷-۴ یک نمونه داده‌های آموزشی را نشان می‌دهد. این جدول دارای دو صفت‌خاصه‌ی  $A$  و  $B$  و برچسب کلاس  $C$  است. در ادامه نشان می‌دهیم که با کمک فرمول احتمال بیز چگونه می‌توانیم کلاس نمونه‌ای را تخمین بزنیم که مقدار صفت‌خاصه‌ی  $A$  آن  $a$  و مقدار صفت‌خاصه‌ی  $B$  آن برابر با  $f$  باشد.

جدول ۷-۴: یک نمونه داده‌های آموزشی

| ID | A   | B   | C   |
|----|-----|-----|-----|
| 1  | $a$ | $d$ | $y$ |
| 2  | $a$ | $e$ | $y$ |
| 3  | $b$ | $f$ | $y$ |
| 4  | $c$ | $e$ | $y$ |
| 5  | $b$ | $f$ | $y$ |
| 6  | $b$ | $f$ | $n$ |
| 7  | $b$ | $e$ | $n$ |
| 8  | $c$ | $d$ | $n$ |
| 9  | $c$ | $f$ | $n$ |
| 10 | $a$ | $d$ | $n$ |

به دنبال کلاس نمونه‌ای هستیم که مقدار صفت خاصه‌ی  $A$  آن  $a$  و مقدار صفت خاصه‌ی  $B$  آن برابر با  $f$  باشد.

$$\begin{array}{lll} P(C=y)=5/10=1/2 & P(C=n)=5/10=1/2 \\ P(A=a/C=y)=2/5 & P(A=b/C=y)=2/5 & P(A=c/C=y)=2/5 \\ P(A=a/C=n)=1/5 & P(A=b/C=n)=2/5 & P(A=c/C=n)=2/5 \\ P(B=d/C=y)=1/5 & P(B=e/C=y)=2/5 & P(B=f/C=y)=2/5 \\ P(B=d/C=n)=2/5 & P(B=e/C=n)=1/5 & P(B=f/C=n)=2/5 \end{array}$$

حال با کمک فرمول احتمال تعلق نمونه‌ی مورد نظر را برای هر دو کلاس  $y$  و  $n$  محاسبه می‌کنیم.

$$P(C = y) \times \prod_{j=1}^2 P(A_j = a_j | C = y) = \frac{1}{2} \times \frac{2}{5} \times \frac{2}{5} = \frac{4}{50} = 0.08$$

$$P(C = n) \times \prod_{j=1}^2 P(A_j = a_j | C = n) = \frac{1}{2} \times \frac{1}{5} \times \frac{2}{5} = \frac{2}{50} = 0.04$$

از آن جا که مقدار محاسبه شده برای کلاس  $y$  بزرگتر است، بنابراین این کلاس برای نمونه‌ی آزمایشی برگزیده می‌شود.

به آسانی می‌توان یافت که برای انجام محاسبات مجبور فقط کافی است داده‌ها یک بار پیمایش شوند و خطی بودن الگوریتم یکی از نقاط قوت آن محسوب می‌شود. هرچند روش با فرض محکم استقلال شرطی کلاس پیش می‌رود، اما تحقیقات نشان می‌دهند که دقیق به دست آمده از آن در مقایسه با تکنیک‌های دیگر نیز بسیار مناسب است. برای کنترل صفات خاصه‌ی عددی می‌توانیم از تکنیک‌های گسترش‌سازی استفاده کنیم، که یک نمونه‌ی آن برای درختان تصمیم و با کمک آنتروپی بیان شد. به طور معمول در مواجهه با مقادیر ناقص هم در محاسبه‌ی احتمالات و هم در نمونه‌ی آزمایشی از آن صرفظیر می‌شود.

یکی از مسائلی که شاید در استفاده از این روش با آن روبرو شویم، مقدار احتمال صفر است. امکان دارد که مقدار صفت خاصه‌ای در مجموعه‌ی آزمایشی هرگز با برچسبی از یک

کلاس اتفاق نیفتاده باشد. نتیجه اینکه احتمال مورد نظر صفر خواهد بود و با جایگذاری در فرمول مقدار به دست آمده برابر با صفر خواهد شد. جهت مقابله با این مشکل راه حل کاربردی ساده‌ای وجود دارد که در ادامه توضیح داده شده است.

فرض کنید  $n_{ij}$  تعداد نمونه‌هایی است که صفت خاصه‌ی  $A_i$  آنها مقداری برابر با  $a_i$  و دارای برچسب کلاس  $C_j$  هستند و  $n_j$  مجموع نمونه‌هایی در داده‌های آموزشی است که برچسب کلاس آنها  $c_j$  است. قبل از این احتمال به صورت زیر محاسبه می‌شود:

$$P(A_i = a_i \mid C = c_j) = \frac{n_{ij}}{n_j}$$

اما با کمی تغییر احتمال را به صورت زیر محاسبه می‌کنیم:

$$P(A_i = a_i \mid C = c_j) = \frac{n_{ij} + \lambda}{n_j + \lambda \cdot d_i}$$

که در آن  $d_i$  تعداد اعضای دامنه‌ی صفت خاصه‌ی  $A_i$  است و  $\lambda$  ضریبی است که به طور معمول یکی از دو مقدار ۱ یا  $1/n$  را به خود می‌گیرد (تعداد نمونه‌های موجود در مجموعه داده‌های آموزشی است).

برای مثال با قرار دادن  $1/n$  برای ضریب  $\lambda$  دو احتمال از احتمالات قبلی به صورت زیر محاسبه می‌شوند:

$$P(A=a_i/C=y) = (2 + 1/10) / (5 + 3 \times 1/10) = 2.1 / 5.3 = 0.396$$

$$P(B=d_i/C=y) = (1 + 1/10) / (5 + 3 \times 1/10) = 1.1 / 5.3 = 0.208$$

## ۴-۷) الگوریتم‌های *SVM*

به جرأت می‌توان گفت الگوریتم‌های *SVM*<sup>۱</sup> از دقیق‌ترین و نیرومند‌ترین الگوریتم‌های داده‌کاوی بشمار می‌رond. این شیوه‌ی جدید می‌تواند برای طبقه‌بندی داده‌های خطی و غیرخطی استفاده شود. در سالهای اخیر به دلیل ارائه‌ی نتایج خوب، این الگوریتم‌ها به یک تکنیک متداول برای طبقه‌بندی تبدیل شده‌اند. با وجود آنکه استفاده از الگوریتم‌های *SVM* در مقایسه با برخی از روش‌های دیگر مثل شبکه‌های عصبی راحت‌تر است، اما به دلیل عدم آشنایی کاربران با جزئیات آن، استفاده‌کنندگان از آن نتایج مناسبی را بدست نمی‌آورند.

چنانچه بخواهیم بطور خلاصه بیان کنیم، الگوریتم‌های *SVM* با کمک یک نگاشت غیرخطی فضای داده‌های آموزشی را به یک بعد بالاتر تبدیل می‌کند و سپس در این بعد جدید به دنبال آبرصفحه‌ای<sup>۲</sup> است که نمونه‌های یک کلاس را از کلاس‌های دیگر جدا کند. با یک نگاشت غیرخطی مناسب، مجموعه داده‌های دو کلاسی می‌توانند توسط یک آبرصفحه جدا شوند.

الگوریتم‌های *SVM* جهت یافتن این آبرصفحه از مفاهیمی چون بردارهای پشتیبان<sup>۳</sup> و حاشیه‌ها<sup>۴</sup> استفاده می‌کنند که در ادامه توضیح خواهیم داد. توجه کنید چنانچه مجموعه داده‌های آموزشی دارای دو صفت خاصه باشند می‌توان تصور نمود که داده‌ها در یک فضای دو بعدی قرار دارند و بنابراین ما به دنبال خطی جهت جداسازی کلاس‌ها هستیم. هنگامی که فضا سه بعدی می‌شود، جداکننده یک صفحه است و بطور عام چون تعداد صفات خاصه در مجموعه داده‌های اولیه بیش از سه عدد است، از کلمه‌ی آبرصفحه برای

---

<sup>1</sup> Support Vector Machine

<sup>2</sup> Hyperplane

<sup>3</sup> Support Vectors

<sup>4</sup> Margins

جداساز میان کلاس‌ها استفاده می‌کنیم. در این بخش ما نیز صرفنظر از تعداد ابعاد ورودی از کلمه‌ی آبرصفحه استفاده خواهیم کرد.

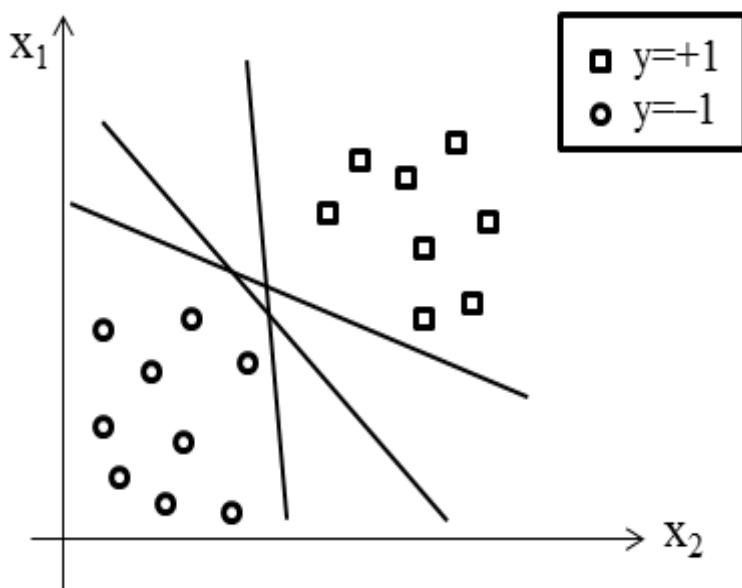
الگوریتم‌های *SVM* به محاسبات پیچیده نیاز دارند و به همین دلیل سریع‌ترین آنها بسیار کند عمل می‌کنند. اما پیچیدگی محاسباتی این الگوریتم‌ها به ابعاد فضای ورودی بستگی ندارد و نتیجه‌ی نهایی از دقت بسیار بالایی برخوردار است. این الگوریتم‌ها بطور خودکار اندازه‌ی مدل را انتخاب می‌کنند و همچنین برای مدل یادگیری شده توصیف فشرده‌ای را ارائه می‌دهند. از این الگوریتم‌ها علاوه بر طبقه‌بندی همچنین می‌توان برای تخمین نیز استفاده نمود. منظور از تخمین استفاده‌ی آنها در مواقعي است که برچسب کلاس گستته نیست. از کاربردهای عملی این الگوریتم‌ها می‌توان به تشخیص الگو، پردازش تصویر، متن کاوی و کاربردهای پزشکی اشاره نمود.

#### ۱-۴-۷) تفکیک‌پذیری خطی

بدون از دست دادن کلیات فرض کنید مجموعه داده‌های آموزشی دارای دو برچسب کلاس هستند و همچنین می‌توان کلاس‌ها را بصورت خطی از یکدیگر جدا ساخت. مجموعه صفات خاصه توسط بردار  $X$  و برچسب کلاس با متغیر  $Y$  نشان داده می‌شود. بدون از دست دادن کلیات مسئله فرض می‌کنیم مقادیر  $+1$  و  $-1$  دو برچسب کلاس را تشکیل می‌دهند. اجازه دهید جهت فهم بهتر و توانایی نمایش، فضای داده‌ها را دو بعدی فرض کنیم. بدین ترتیب هر بردار  $X_i$  برابر با دو مقدار خواهد بود و می‌توان مجموعه داده‌ها را در یک سیستم دکارتی نمایش داد. شکل ۷-۸ نمونه‌ای از این داده‌ها را نمایش می‌دهد.

با مشاهده به راحتی قابل تشخیص است که تعداد آبرصفحه‌هایی (در این مثال خطوط) که می‌توان نمونه‌ها را بر اساس برچسب کلاس‌شان تفکیک نمود، محدود نیست. در شکل

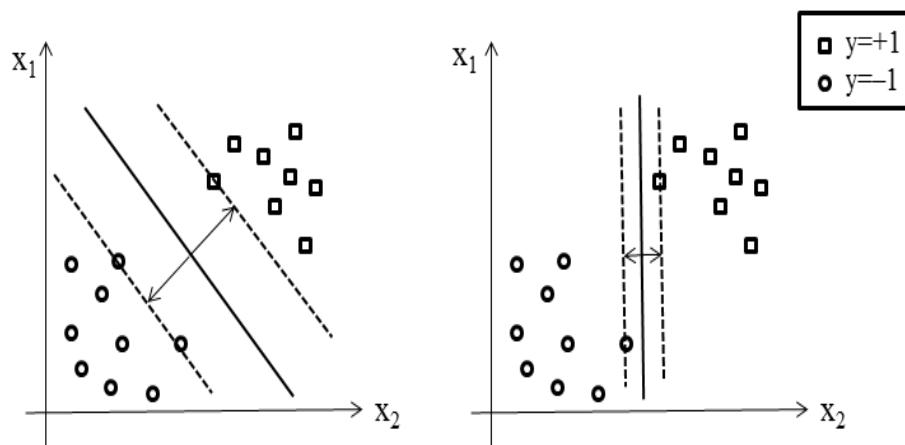
۷-۸ فقط ۳ نمونه از این خطوط رسم شده است. اما کدامیک از آنها بهترین انتخاب برای جداسازی نمونه‌های متفاوت است؟



شکل ۷-۸: نمایش مجموعه‌ای از داده‌های دو کلاسی در فضای دو بعدی

یک الگوریتم *SVM* به دنبال آبرصفحه‌ای با حداقل حاشیه می‌گردد. انتظار می‌رود آبرصفحه‌ای با حاشیه‌ی بیشتر دقت بیشتری را نیز در طبقه‌بندی داده‌های آزمایشی داشته باشد.

شکل ۷-۹ دو آبرصفحه (خط) را برای جداسازی داده‌های یکسانی نشان می‌دهد که در آنها حاشیه‌ها یکسان نیستند. در این شکل حاشیه‌ها با خطوط پیکاندار مشخص شده‌اند. از نقطه نظر هندسی حاشیه از فاصله‌ی موجود بین آبرصفحه و نزدیک‌ترین نمونه‌های آموزشی محاسبه می‌شود. کوتاهترین فاصله از یک آبرصفحه تا نمونه‌ای (نمونه‌هایی) با برچسب  $+1$  برابر با کوتاهترین فاصله از آن آبرصفحه تا نمونه‌ای (یا نمونه‌هایی) با برچسب  $-1$  است. در واقع حاشیه از دو برابر این کوتاهترین فاصله بدست می‌آید.



شکل ۷-۹: مقایسه‌ی حاشیه‌های دو آبرصفحه برای داده‌های یکسان

## ۷-۵) طبقه‌بندی براساس تشابه نزدیک

همه‌ی روش‌های طبقه‌بندی که تاکنون توضیح دادیم، برای تخمین کلاس یک نمونه‌ی آزمایشی ابتدا مدلی را با کمک داده‌ها طراحی می‌کنند و پس از آن با استفاده از مدل، برچسب کلاس نمونه‌ی خواسته شده را تخمین می‌زنند. تصور کنید بدون ساختن مدل، روش با مقایسه‌ی نمونه‌ی آزمایشی و مجموعه داده‌ها و یافتن مشابه‌ترین نمونه قادر باشد کلاس داده‌ی آزمایشی را تخمین بزند. این روش الگوریتم‌هایی موسوم به یادگیرنده‌های تنبیل<sup>۱</sup> است. کلمه‌ی تنبیل به این جهت انتخاب شده است که روش تا ورود داده‌ی آزمایشی صبر می‌کند و به ساخت مدلی جهت طبقه‌بندی نمی‌پردازد. این دسته از الگوریتم‌ها به تکنیک‌های کارایی جهت ذخیره‌سازی و بازیابی نیاز دارند و مناسب برای پیاده‌سازی در محیط‌های موازی هستند. در ضمن این روش‌ها بر روی داده‌هایی که از مدل پیچیده‌ای برخوردارند نیز عملکرد خوبی دارند. چرا که راهکارهای دیگر برای ساختن مدل برای چنین داده‌هایی با مشکلاتی روبرو می‌شوند. یکی از معروف‌ترین این روش‌ها  $k$  نزدیکترین همسایه<sup>۲</sup> نام دارد که در ادامه توضیحاتی پیرامون این روش بیان شده است.

---

<sup>1</sup> Lazy Learner

<sup>2</sup> K Nearest Neighbor

### ۱-۵-۷) $k$ نزدیکترین همسایه ( $knn$ )

الگوریتم  $knn$  هنگامی که قدرت محاسباتی کامپیوترها افزایش یافت، محبوب شد و یکی از کاربردهای رایج آن تشخیص الگو است. برای یک داده‌ی آزمایشی الگوریتم به دنبال  $k$  نمونه از نزدیکترین نمونه‌ها می‌گردد ( $k$  نمونه‌ی مشابه). نزدیکی دو نمونه با بدست آوردن تشابه و یا فاصله‌ی میان این دو نمونه محاسبه می‌شود. هر نمونه می‌تواند از انواع داده‌ها تشکیل شده باشد که باید تشابه میان آنها بررسی شود. در فصل مربوط به خوشبندی روش‌های متعددی جهت محاسبه‌ی تشابه یا فاصله شرح داده شده است که از هر یک می‌توان برای الگوریتم  $knn$  استفاده نمود.

پس از یافتن این  $k$  داده‌ی مشابه با نمونه‌ی آزمایشی، با رأی اکثريت برچسب کلاس داده‌ی آزمایشی انتخاب می‌شود. چنانچه مقدار ۱ برای  $k$  تنظيم شود، در اين صورت کلاس نزدیکترین داده به نمونه‌ی آزمایشی، به عنوان کلاس تخمینی ارائه می‌شود. اما به دليل وجود داده‌های نويز و خارج از محدوده مقدار ۱ عدد مناسبی برای  $k$  نیست. می‌توان مقدار مناسب را به صورت تجربی به دست آورد. برای مثال با ۱ شروع و برای مجموعه‌ای آزمایشی نرخ خطای افزايش مقدار  $k$  اين کار را تكرار می‌كنيم. مقداری از  $k$  که باعث حداقل نرخ خطای می‌شود، انتخاب مناسبی است. در كاربرد مقادير ۳ و ۵ برای  $k$  نتایج خوبی را به دنبال داشته است.

این الگوریتم همچنین می‌تواند برای داده‌هایی که برچسب کلاس آنها از نوع پيوسته (عددی) است نيز استفاده شود. در اين صورت پس از یافتن  $k$  همسایه، ميانگين مقادير حاصل از کلاس اين  $k$  نمونه به عنوان برچسب کلاس نمونه‌ی آزمایشی برگزide می‌شود. هرگاه مقدار یا مقاديری از صفات خاصه در مجموعه داده‌های اصلی یا آزمایشی ناقص باشند در محاسبه‌ی تشابه کمترین و در محاسبه‌ی فاصله بيشترین فاصله برای اين صفت خاصه در نظر گرفته می‌شود. فرض کنيد مقادير يك صفت خاصه عددی به فاصله‌ی صفر تا يك نگاشت می‌شوند. می‌دانيد که اين عمل می‌تواند با يك نرمال‌سازی ساده انجام شود. برای محاسبه‌ی فاصله‌ی میان دو نمونه و برای اين صفت خاصه اگر هر دو مقدار ناقص باشند فاصله برابر يك (تشابه برابر با صفر) در نظر گرفته می‌شود. اما چنانچه

یکی از این مقادیر ناقص باشد و دیگری دارای ارزشی برابر با ۰، فاصله و تشابه از فرمول‌های زیر بدست می‌آیند:

$$Dis(A_i, A_j) = \text{Max}(|I-v|, |O-v|)$$

$$Sim(A_i, A_j) = \text{Min}(|I-v|, |O-v|)$$

برای صفات خاصه غیر عددی کافی است حداقل یکی از آنها ناقص باشد، تا فاصله برابر با یک و تشابه برابر با صفر تنظیم شود.

با انتساب وزن به هر یک از صفات خاصه درصد مشارکت صفات خاصه در محاسبه‌ی تشابه و یا فاصله‌ی میان نمونه‌ها را کمتر یا بیشتر می‌کنیم. بدین ترتیب صفات خاصه‌ی نامربوط و یا داده‌های نویز تأثیر کمتری در فرایند خواهد داشت.

بدلیل اینکه الگوریتم  $knn$  برای یافتن برچسب کلاس داده‌های آزمایشی باید کلیه‌ی داده‌ها را پیمایش کند، این عمل در داده‌هایی با حجم بسیار بالا می‌تواند به شدت از کارایی الگوریتم بکاهد. تمهیداتی وجود دارند که پیچیدگی الگوریتم را بهبود می‌بخشند. ذخیره‌ی داده‌های اولیه در یک ساختار درختی می‌تواند پیچیدگی جستجو و پیمایش را لگاریتمی کند و یا با پیاده‌سازی موازی الگوریتم انتظار داریم سرعت بهتری از الگوریتم داشته باشیم.

روش کاربردی دیگر جهت بهبود الگوریتم اولیه، محاسبه‌ی فاصله و یا تشابه میان زیرمجموعه‌های از صفات خاصه به جای فضای کل است. بدین ترتیب که در ابتدا فاصله‌ی میان نمونه‌ی آزمایشی و داده‌های ذخیره شده با توجه به زیرمجموعه‌های از صفات خاصه به جای کلیه‌ی آنها محاسبه می‌شود. در این صورت چنانچه فاصله (تشابه) از مقدار تعریف شده‌ای بیشتر (کمتر) باشد، بدون محاسبه‌ی کامل فاصله یا تشابه به سراغ داده‌ی بعدی خواهیم رفت. به علاوه اگر به هر نحوی قادر باشیم برخی از نمونه‌ها را از فضای جستجو خارج کنیم بدون شک سرعت الگوریتم افزایش می‌یابد. در آخر با یک مثال ساده بحث در مورد الگوریتم  $knn$  را به پایان می‌بریم.

جدول ۷-۶ حاوی مشخصات ۶ نقطه است که در دو کلاس  $A$  و  $B$  طبقه‌بندی شده‌اند.

جدول ۷-۶: مشخصات دکارتی ۶ نقطه همراه با کلاس هر یک

| ID | X | Y | Class | Distance |  |
|----|---|---|-------|----------|--|
|    |   |   |       | to P     |  |
| 1  | 2 | 5 | A     | 3.00     |  |
| 2  | 6 | 3 | B     | 4.12     |  |
| 3  | 4 | 4 | A     | 2.83     |  |
| 4  | 1 | 2 | B     | 1.00     |  |
| 5  | 1 | 6 | B     | 4.12     |  |
| 6  | 3 | 2 | A     | 1.00     |  |

با تنظیم عدد ۳ برای مقدار  $k$  می‌خواهیم کلاس نقطه  $P(2,2)$  را با روش  $knn$  بدست آوریم. در ستون آخر جدول فاصله‌ی نقطه‌ی  $P$  با هر یک از نمونه‌ها نیز محاسبه شده است(فاصله‌ی اقلیدسی).

از آنجا که مقدار  $k$  برابر با ۳ است، الگوریتم ۳ نمونه‌ی نزدیک به نقطه  $P$  را انتخاب خواهد کرد. نمونه‌های سوم، چهارم و ششم انتخاب می‌شوند. در میان این ۳ نمونه، دو برچسب  $A$  و یک برچسب  $B$  قرار دارد که بدین ترتیب کلاس نقطه‌ی  $P$  برابر با  $A$  تخمین زده می‌شود.

# خوشه‌بندی

خوشه‌بندی<sup>۱</sup> یک تابع کاوشی ناظارت‌نشده‌ی<sup>۲</sup> داده‌کاوی به منظور کشف گروه‌بندی طبیعی درون داده‌هاست. یک خوشه به مجموعه‌ای از داده‌ها اطلاق می‌شود که از جهاتی شبیه به هم دیگر هستند. الگوریتم‌های خوشه‌بندی به طور خودکار ویژگی‌های متمایز کننده‌ی زیر گروه‌ها را تعریف می‌کنند و آنها را سازماندهی می‌نمایند و مدل را منحصرًا از روی روابطی که در داده‌ها وجود دارند و همچنین از روی خوشه‌هایی که الگوریتم شناسایی می‌نماید، آموزش می‌دهند.

خوشه‌بندی در واقع یافتن ساختار در مجموعه‌ای از داده‌هایی است که طبقه‌بندی نشده‌اند. به بیان دیگر می‌توان گفت که خوشه‌بندی قراردادن داده‌ها در گروه‌هایی است که اعضای هر گروه از زاویه‌ی خاصی به یکدیگر شباهت دارند و با اعضای خوشه‌های دیگر هیچ شباهتی ندارند یا حداقل نسبت به اعضای خوشه‌ی خود از شباهت بسیار کمتری با

---

<sup>1</sup> Clustering

<sup>2</sup> Unsupervised

اعضای دیگر خوشه‌ها برخوردارند. معیار شباهت در اینجا فاصله بوده و نحوه محاسبه‌ی این فاصله در خوشه‌بندی بسیار مهم است. فاصله که همان معرف عدم تجانس است به ما کمک می‌کند در فضای داده‌ای حرکت کنیم و خوشه‌ها را تشکیل دهیم. با محاسبه‌ی فاصله‌ی بین دو داده می‌توان فهمید تا چه اندازه این دو داده به هم نزدیک هستند و بر این اساس می‌توان آنها را در یک خوشه قرار داد. خوشه‌بندی نوعی تحلیل اطلاعات نظرارت نشده است. به این معنی که هیچگونه نشانه‌گذاری، مرتب‌سازی و یا برچسب‌دهی اولیه بر روی اطلاعات انجام نشده است. در واقع مسئله‌ی اصلی، دسته‌بندی همین نوع اطلاعات به خوشه‌های معنی‌دار است.

<sup>1</sup> Classification

## ۸-۲) الگوریتم‌های خوشه‌بندی

تکنیک‌های خوشه‌بندی را می‌توان از زوایای متفاوت به دسته‌های مختلفی گروه‌بندی نمود. در این بخش ما این تکنیک‌ها را در پنج گروه اصلی دسته‌بندی کرده‌ایم، هرچند برخی از آنها را نمی‌توان به صورت دقیق فقط به یکی از این دسته‌ها تخصیص داد.

- خوشه‌بندی مبتنی بر افزار<sup>۱</sup>
- خوشه‌بندی سلسله مراتبی<sup>۲</sup>
- خوشه‌بندی مبتنی بر تراکم یا چگالی داده‌ها<sup>۳</sup>
- خوشه‌بندی مبتنی بر شبکه‌های شطرنجی گردید<sup>۴</sup>
- خوشه‌بندی مبتنی بر مدل<sup>۵</sup>

مبنای هر یک از گروه‌های بالا با دیگر گروه‌ها بسیار متفاوت است و هر یک از گروه‌های فوق دارای فرایند اجرا و همچنین قالب‌های متفاوت در نتیجه‌ی نهایی خود هستند.

در الگوریتم‌هایی که خوشه‌بندی مبتنی بر افزار داده‌ها انجام می‌شود، همانطور که از نام آن مشخص است داده‌ها به درون چند خوشه‌ی جدا افزار می‌شوند. این بدین معنی است که تعداد خوشه‌ها معمولاً<sup>۶</sup> در این الگوریتم‌ها به عنوان ورودی الگوریتم مشخص می‌شود و هر نمونه داده فقط عضو یک خوشه می‌تواند باشد. قرارگرفتن هر نمونه در یک خوشه توسط معیارهای تشابهی که الگوریتم مشخص می‌کند، ارزیابی می‌شود.

همانطور که می‌دانید این معیارهای تشابه بر روی عملکرد الگوریتم‌های خوشه‌بندی تأثیر به سزاوی دارند. به همین دلیل تکنیک‌های خوشه‌بندی مبتنی بر افزار تاکید بسیار زیادی بر روی این معیارها دارند. در صورتی که تکنیک‌های دیگر خوشه‌بندی نگاهی به مدل داده‌ها نیز می‌کنند و به تنها‌ی با توابع و معیارهای تشابه به عمل یافتن خوشه‌ها ادامه نمی‌دهند. در ادامه با بررسی چند نمونه از این الگوریتم‌ها به این نکته بیشتر و بهتر اشاره می‌شود.

<sup>1</sup> Partitioning Method

<sup>2</sup> Hierarchical

<sup>3</sup> Density- based

<sup>4</sup> Grid- based

<sup>5</sup> Model- based

در حالی که تکنیک‌های خوشبندی مبتنی بر افزار با یک ورودی که مشخص کننده تعداد خوشبند است شروع می‌کند و هر نمونه را با کمک معیارهای معرفی شده در الگوریتم به خوشبند تخصیص می‌دهد، در روش‌ها و تکنیک‌های خوشبندی سلسله‌مراتبی ما با یکی از دو عمل ادغام<sup>۱</sup> و یا تقسیم<sup>۲</sup> روبرو هستیم. در حقیقت دو نوع خوشبندی سلسله‌مراتبی وجود دارد. الگوریتم‌های نوع اول در ابتدا هر یک از نمونه‌ها را در یک خوشبند قرار می‌دهند. به عبارتی دیگر در شروع اجرای الگوریتم، تعداد خوشبند برابر با تعداد نمونه‌ها است و الگوریتم در حین اجرا سعی بر ادغام نمودن خوشبندی دارد که در ارزیابی معیار تشابه موفق باشند، یعنی شباهت چشمگیری با یکدیگر داشته باشند. این در حالی است که در الگوریتم‌های نوع دوم تمام نمونه‌ها در یک خوشبند قرار داده می‌شوند و در هر مرحله از الگوریتم، نمونه‌هایی که کمترین شباهت را دارند در خوشبندی متفاوت قرار می‌گیرند. نوع اول از الگوریتم‌های خوشبندی سلسله‌مراتبی بصورت گستردگی نسبت به نوع دوم استفاده می‌شوند و با بررسی چند الگوریتم از این نمونه به خوبی به این نکته پی خواهیم برد.

از آنجا که تکنیک‌های خوشبندی مبتنی بر افزار محدود به یافتن خوشبندی به شکل کروی (شکل قرار گرفتن داده‌ها در فضای  $n$  بعدی) می‌شوند، باید تکنیک‌هایی وجود داشته باشد تا این عیب را پوشش دهند. بدین ترتیب تکنیک‌های خوشبندی مبتنی بر تراکم توسعه داده شدند. در این تکنیک‌ها یا خوشبندی‌ها بر اساس تراکم نمونه‌های همسایه رشد می‌یابند و یا بر طبق تحلیل یک تابع تراکم که از قبل تعریف شده است. به هر حال فضای کل نمونه‌ها به نواحی با تراکم بالا تقسیم می‌شوند. این نکته باعث می‌شود تا تکنیک خوشبندی محدود به خوشبندی‌ای با شکل خاص (مثلاً کروی) نباشد.

نوع چهارم از تکنیک‌های خوشبندی موسوم به تکنیک‌های مبتنی بر شبکه‌ی گرید هستند. در این گونه تکنیک‌ها فضای نمونه‌ها بر روی تعداد محدودی از سلول‌ها با ساختار شبکه‌ی شطرنجی قرار می‌گیرند و یا در واقع نقش می‌بندند. کلیه‌ی عملیات خوشبندی بر روی این ساختار اجرا می‌شود. زمان پردازش سریع در این روش از مزیت اصلی آن به

<sup>1</sup> Merge

<sup>2</sup> Split

شمار می‌رود. معمولاً زمان اجرای تکنیک‌های خوشه‌بندی مبتنی بر گردید وابسته به تعداد سلول‌هایی است که نمونه‌ها بر روی آنها نقش می‌بندند، نه به تعداد نمونه داده‌هایی که ورودی الگوریتم را تشکیل می‌دهند.

در تکنیک‌های مبتنی بر مدل همانطور که از نام آن مشخص است، برای هر یک از خوشه‌ها یک مدل پیش فرض در نظر گرفته می‌شود و در ادامه به یافتن نمونه‌هایی که در این مدل گنجانده می‌شوند، می‌پردازد. بطور معمول در این نوع تکنیک خوشه‌بندی از راهکارهای آماری و یا شبکه‌های عصبی استفاده می‌شود.

### ۳-۸) معیارهای تشابه و انواع داده‌ها

موضوع اصلی در تکنیک‌های خوشبندی تشابه<sup>۱</sup> و عدم تشابه<sup>۲</sup> دو نمونه داده است. در هر خوشبندی‌هایی که تشابه بیشتری دارند، قرار می‌گیرند. به عبارتی دیگر قرار است تا نمونه‌های مشابه به یکدیگر در یک خوش و نمونه‌های غیرمشابه در خوشبندی متفاوت گروه‌بندی شوند. بنابراین به منظور ارزیابی تشابه نیاز به مقیاس و یا معیاری ضروری است. از آنجا که هر نمونه می‌تواند شامل صفات خاصه متعددی باشد و هر یک از این صفات خاصه یک نوع داده تلقی می‌شود، لذا در محاسبه یا تحلیل تشابه دو نمونه باید معیارهای تشابه برای انواع داده‌ها تعریف شوند.

داده‌ها می‌توانند توسط یک ماتریس یا یک جدول نشان داده شوند. تعداد ستون‌ها در این ساختار نشان‌دهنده‌ی ویژگی‌ها و صفات خاصه‌ی هر نمونه و تعداد سطرها معرف تعداد داده‌ها یا نمونه‌ها هستند. این روش نمایش داده‌ها کاملاً شبیه یک جدول یا رابطه در مدل رابطه‌ای است. به عبارتی دیگر یک ماتریس  $n \times m$  حاوی مقادیری است که  $n$  نمونه داده را نمایش می‌دهد و هر یک از این  $n$  نمونه دارای  $m$  ویژگی هستند. برای مثال اگر نمونه‌ها موجودیت دانشجو را نشان می‌دهند، ویژگی‌ها یا صفات خاصه‌ای نظیر شماره دانشجویی، نام و نام خانوادگی را برای هر دانشجو ثبت کرده‌ایم. همان طور که قبل از این نیز به آن اشاره شد، نوع هر صفت خاصه یا ویژگی می‌تواند با نوع ویژگی دیگر متفاوت باشد. شماره دانشجویی یک نوع داده‌ی عددی است در حالی که نام و نام خانوادگی اینطور نیست. ما در این کتاب از بحث در مورد مجموعه داده‌های خاص مانند داده‌های چندرسانه‌ای<sup>۳</sup> پرهیز می‌کنیم، هر چند در بسیاری از برنامه‌های کاربردی می‌توان در ابتدا داده‌ها را به شکل داده‌های جدولی تبدیل نمود و از تکنیک‌های ارائه شده در این کتاب استفاده کرد.

ماتریس دیگری در تحلیل خوشبندی وجود دارد که محتويات آن تشابه یا عدم تشابه (فاصله) بین نمونه‌ها را نشان می‌دهد. معمولاً هر عضو این ماتریس عدد مثبتی است که

<sup>1</sup> Similarity

<sup>2</sup> Dissimilarity (Distance)

<sup>3</sup> Multimedia

به نزدیکی (یا دور بودن) دو نمونه دلالت دارد و از آنجا که در اکثر موقع و نه همه زمان‌ها، تشابه نمونه‌های  $a$  و  $b$  در معنی تفاوتی با تشابه میان نمونه‌های  $a$  و  $b$  ندارد، این ماتریس متقارن است. لذا فقط کافی است اعضای بالای قطر اصلی و یا پایین آن نگهداری شوند. ماتریس شکل ۸-۱ عدمتشابه (فاصله) میان  $n$  نمونه را نشان می‌دهد. در این ماتریس ( $d_{i,j}$ ) فاصله‌ی دو نمونه‌ی  $i$ ام و  $j$ ام را بیان می‌کند. عدد کوچکتر نشان دهنده‌ی تشابه بالای دو موجودیت یا نمونه است.

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \dots & \dots & \dots & \dots & \dots \\ d(n,1) & d(n,2) & d(n,3) & \dots & 0 \end{bmatrix}$$

شکل ۸-۱: نمونه‌ای از یک ماتریس تشابه

ما این ماتریس را ماتریس تشابه<sup>۱</sup> می‌نامیم. نکته مهمی که باید به آن اشاره شود اینکه مقادیر موجود در این ماتریس هم می‌تواند تشابه میان نمونه‌ها و هم می‌تواند عدمتشابه (فاصله) میان آنها را نشان دهد. هر چند در کاربرد باید این مسئله به روشنی مشخص شود، اما در طول کتاب هر دوی آنها را با نام ماتریس تشابه می‌شناسیم و در هر لحظه مشخص خواهیم نمود که در ماتریس اعداد نشان‌دهنده‌ی تشابه هستند و یا فاصله. به عبارت دیگر روشن خواهد شد که مقدار بزرگتر هر عضو ماتریس نشان تشابه بیشتر است یا تشابه کمتر. در ادامه روش‌هایی برای ارزیابی تشابه میان نمونه‌هایی بیان می‌شوند که صفات خاصه آنها می‌تواند از انواع داده‌ای (عددی، غیر عددی، دودویی،...) تشکیل شده باشد.

### ۸-۳-۱) معیارهای تشابه در داده‌های پیوسته

مرسوم‌ترین نوع داده که می‌تواند مقدار صفت‌خاصه‌ی یک نمونه را نشان دهد، داده‌های عددی هستند. منظور از داده‌های عددی مقادیر پیوسته مانند اعداد حقیقی است. مقادیر

<sup>۱</sup> Similarity Matrix

صفات خاصه‌ای مانند قد و وزن مثال‌های بارزی از این نوع داده هستند. قبل از بیان چند معیار تشابه میان این نوع از داده ذکر یک نکته‌ی مهم ضروری است و آن هم واحد اندازه‌گیری این ویژگی‌هاست. همانطور که می‌دانید واحد اندازه‌گیری برای یک صفت خاصه می‌تواند متفاوت باشد. برای مثال قد یک شخص می‌تواند با واحد سانتی‌متر و یا اینچ سنجیده شود. حداقل اینکه مطمئن هستیم واحد اندازه‌گیری صفات خاصه مختلف متفاوت هستند. هر مقدار با واحدهای اندازه‌گیری متفاوت ممکن است تأثیر مختلفی بر روی روش‌های خوشبندی داشته باشد. بیان ویژگی قد در واحد کوچکتر (مقدار بزرگتر) باعث می‌شود تا شما عدد بزرگتری (کوچکتری) را در تحلیل معیار تشابه به کار ببرید. به عبارت دیگر تأثیر بیشتر این ویژگی را در الگوریتم خوشبندی باعث می‌شود. لذا قبل از هر چیز و به منظور مستقل بودن نتیجه‌ی خوشبندی و همچنین واحدهای اندازه‌گیری صفات خاصه، این مقادیر نرمال‌سازی می‌شوند. از آنجا که معمولاً این گونه عملیات در مرحله‌ی پیش‌پردازش و آماده‌سازی داده‌ها انجام می‌شود، شما می‌توانید اطلاعات بیشتر در این زمینه را در فصل دوم مطالعه فرمایید.

یکی از پرکاربردترین معیار ارزیابی تشابه میان ویژگی‌هایی که مقادیر آنها پیوسته است، فاصله‌ی اقلیدسی<sup>۱</sup> است. فاصله‌ی هندسی که در یک فضای چند بعدی برای دو نمونه از فرمول زیر محاسبه می‌شود:

$$\text{Distance}(O_i, O_j) = \sqrt{\sum_{k=1}^m (x_{ik} - x_{jk})^2}$$

در فرمول فوق  $m$  تعداد ویژگی‌ها و  $\lambda$ ها مقادیر صفات خاصه برای نمونه داده‌ها هستند. برای مثال جدول ۱-۸ که برای ۳ نمونه که با ۴ ویژگی توصیف شده‌اند را در نظر بگیرید.

جدول ۱-۸: نمایش سه نمونه با چهار صفت خاصه

| Object | Att <sub>1</sub> | Att <sub>2</sub> | Att <sub>3</sub> | Att <sub>4</sub> |
|--------|------------------|------------------|------------------|------------------|
| $O_1$  | 4                | 5                | 4                | 8                |
| $O_2$  | 7                | 8                | 3                | 1                |
| $O_3$  | 3                | 4                | 5                | 3                |

<sup>1</sup> Euclidean Distance

با محاسبه‌ی فاصله‌ی اقلیدسی میان زوج نمونه‌های جدول ۱-۸ داریم:

$$\text{Distance}(O_1, O_2) = \sqrt{(4-7)^2 + (5-8)^2 + (4-3)^2 + (8-1)^2} = 8.25$$

$$\text{Distance}(O_2, O_3) = \sqrt{(7-3)^2 + (8-4)^2 + (3-5)^2 + (1-3)^2} = 6.32$$

$$\text{Distance}(O_1, O_3) = \sqrt{(4-3)^2 + (5-4)^2 + (4-5)^2 + (8-3)^2} = 5.29$$

عدد کوچکتر نشان دهنده‌ی فاصله‌ی کمتر و در نتیجه تشابه بیشتر است. فاصله‌ی اقلیدسی یک معیار و مقیاسی است که فاصله را نشان می‌دهد. بدین معنی که مقدار بزرگتر در این معیار به مفهوم فاصله‌ی بیشتر و تشابه کمتر دو نمونه است. در برخی از موقع معيارهایی که تشابه را بجای عدم تشابه نشان می‌دهند به کار برده می‌شود. همانطور که متوجه شدید عدم تشابه تحت عنوان فاصله معرفی می‌گردد. در این مثال نمونه‌ی  $O_1$  به  $O_3$  نزدیکتر است تا به نمونه‌ی  $O_2$ .

معیار معروف دیگری که برای محاسبه‌ی فاصله میان داده‌های پیوسته استفاده می‌شود با نام فاصله‌ی مانهاتان<sup>۱</sup> (بلوک شهری<sup>۲</sup>) شناخته و با فرمول زیر بیان می‌شود:

$$\text{Distance}(O_i, O_j) = \sum_{k=1}^m |x_{ik} - x_{jk}|$$

برای سه نمونه موجود در جدول ۱-۸، این مقیاس بصورت زیر محاسبه می‌شود:

$$\text{Distance}(O_1, O_2) = |4-7| + |5-8| + |4-3| + |8-1| = 14$$

$$\text{Distance}(O_2, O_3) = |7-3| + |8-4| + |3-5| + |1-3| = 12$$

$$\text{Distance}(O_1, O_3) = |4-3| + |5-4| + |4-5| + |8-3| = 8$$

همانند فاصله‌ی اقلیدسی با محاسبه‌ی این معیار نیز متوجه می‌شویم که تشابه نمونه‌های  $O_2$  و  $O_3$  بیشتر از تشابه میان نمونه‌های دیگر است، اما توجه کنید که همیشه اینطور نیست.

<sup>1</sup> Manhattan

<sup>2</sup> City Block

فاصله‌ی اقلیدسی و فاصله‌ی مانهاتان (بلوک شهری) شکل خاصی از معیار مینکوفسکی<sup>۱</sup> هستند که برای  $p$ ‌های بزرگتر از صفر به صورت کلی زیر بیان می‌شود:

$$\text{Distance}(O_i, O_j) = \left( \sum_{k=1}^m |x_{ik} - x_{jk}|^p \right)^{1/p}$$

با قرار دادن مقادیر ۱ و ۲ برای  $p$  در این فرمول به ترتیب فاصله‌های مانهاتان (بلوک شهری) و اقلیدسی را بدست خواهیم آورد.

برای این دو معیار و همچنین بسیاری از معیارهای فاصله، خصوصیات زیر را می‌توان تعریف کرد:

- فاصله یک معیار عددی غیر منفی است یعنی  $\text{Distance}(O_i, O_j) \geq 0$
- فاصله یک تابع متقارن است یعنی به عبارتی دیگر خواهیم داشت:  $\text{Distance}(O_i, O_j) = \text{Distance}(O_j, O_i)$
- فاصله‌ی هر نمونه با خودش برابر با صفر است یعنی  $\text{Distance}(O_i, O_i) = 0$
- میان سه نمونه، نامعادله‌ی مثلثی زیر وجود دارد:  $\text{Distance}(O_i, O_j) \leq \text{Distance}(O_i, O_k) + \text{Distance}(O_k, O_j)$

---

<sup>1</sup> Minkowski

## ۴-۸) تکنیک‌های خوشبندی مبتنی بر افزایش داده‌ها

در این نوع از تکنیک‌های خوشبندی با فرض وجود  $n$  نمونه و ورودی  $k$  به عنوان تعداد خوشبندی‌های نهایی، نمونه‌ها در  $k$  خوشبندی می‌شوند. معیاری جهت ارزیابی تشابه میان نمونه‌های یک خوشبندی و عدم تشابه میان خوشبندی‌ها برای شکل‌دهی آنها در حالت بهینه استفاده می‌شود. از شناخته شده‌ترین الگوریتم‌های خوشبندی مبتنی بر افزایش داده‌ها می‌توان به الگوریتم‌های مانند  $k$ -*Medoids* و  $k$ -*Means* اشاره کرد. معمولاً الگوریتم‌های دیگر، تعمیم‌یافته‌ی این دو الگوریتم یا بهتر بیان کنیم تعمیم‌یافته‌ی الگوریتم  $k$ -*Means* هستند. در این بخش به بررسی چند الگوریتم که از تکنیک افزایش داده‌ها استفاده می‌کنند، می‌پردازیم.

### ۱-۴-۸) الگوریتم $k$ -*Means*

همانطور که قبل از این نیز به آن اشاره شد، ورودی این الگوریتم  $n$  نمونه داده و مقدار  $k$  که تعداد خوشبندی‌های خروجی را مشخص می‌کند، می‌باشد. در ابتدا تعداد  $k$  نمونه به صورت اتفاقی از میان کل نمونه‌ها انتخاب می‌شوند. این نمونه‌ها به عنوان نماینده‌ی<sup>۱</sup> خوشبندی شناخته خواهند شد. گاهی به آنها مرکز ثقل<sup>۲</sup> یا مرکز خوشبندی نیز اطلاق می‌شود. هر یک از نمونه‌های باقیمانده عضوی از خوشبندی خواهند بود که یکی از این نماینده‌ها ( $k$  نمونه) متعلق به آن است. به عبارت دیگر با کمک معیارهایی همچون فاصله‌ی اقلیدسی تشابه هر یک از نمونه‌های باقیمانده را با  $k$  نماینده محاسبه می‌کنیم و نمونه‌ی مورد نظر به هر یک نزدیکتر بود، به عضویت آن خوشبندی در می‌آید. پس از آن برای هر خوشبندی، با محاسبه میانگین میان اعضای خوشبندی جدیدی انتخاب می‌گردد. این فرآیند تا پوشش

---

<sup>1</sup> Centroid

<sup>2</sup> Center of Gravity

معیاری جهت خاتمه‌ی کار تکرار می‌شود. برای مثال این فرآیند می‌تواند تا هنگامی که دیگر هیچ یک از نمونه‌ها خوش‌های خود را تغییر ندهند، ادامه پیدا کند. معمولاً به حداقل رساندن درصد خطأ که از فرمول زیر محاسبه می‌شود، معیار خوبی برای شرط پایانی است.

$$E = \sum_{i=1}^k \sum_{p \in c_i} (p - m_i)^2$$

در عبارت فوق  $E$  مجموع مجذور خطأ برای کلیه نمونه‌ها،  $p$  یک نمونه از داده‌ها و  $m_i$  نماینده یا مرکز ثقل خوش  $c_i$  است. به عبارت دیگر فاصله‌ی هر نمونه با نماینده‌ی خوش‌های مذکور به توان دو می‌رسد و مقادیر با یکدیگر جمع می‌شوند. با حداقل کردن مقدار این خطأ امیدواریم خوش‌های بدست آمده تا حد ممکن فشرده و جدا از یکدیگر باشند. توجه کنید در این الگوریتم نمونه‌ها فقط به یک خوش‌ه تعقیل دارند و نمی‌توانند هم‌زمان عضو چند خوش‌ه باشند.

اجازه دهید مراحل اجرای الگوریتم  $k$ -Means را بر پایه‌ی مجموعه داده‌های ساده‌ای که در جدول ۸-۲ ارائه شده است، تجزیه و تحلیل کنیم. برای شروع ماتریس تشابه را برای این داده‌ها بدست می‌آوریم (شکل ۸-۴). توجه کنید محتوای این ماتریس می‌تواند با کمک هر معیار تشابه‌ی بدست آید. ما از فاصله‌ی اقلیدسی استفاده کرده‌ایم.

|     | $A$  | $B$  | $C$  | $D$  | $E$  | $F$ |
|-----|------|------|------|------|------|-----|
| $A$ | 0    |      |      |      |      |     |
| $B$ | 2.80 | 0    |      |      |      |     |
| $C$ | 7.00 | 4.47 | 0    |      |      |     |
| $D$ | 1.00 | 3.60 | 7.80 | 0    |      |     |
| $E$ | 1.40 | 4.24 | 8.60 | 1.00 | 0    |     |
| $F$ | 2.00 | 4.47 | 8.48 | 1.00 | 1.40 | 0   |

شکل ۸-۴: ماتریس تشابه برای داده‌های موجود در جدول ۸-۲

می‌خواهیم این ۶ نمونه را در دو خوش‌ه قرار دهیم ( $k=2$ ). بنابراین ابتدا از ۶ نمونه‌ی موجود بصورت اتفاقی ۲ نمونه را انتخاب می‌کنیم. در این مثال ما نمونه‌های  $A$  و  $B$  را به

عنوان اولین مرکز ثقل‌های ۲ خوشی مورد نظر فرض می‌کنیم. در واقع ابتدا این ۲ نمونه به عنوان نمایندگان ۲ خوشی در نظر گرفته می‌شوند. با توجه به ماتریس تشابه می‌توانیم اعضاء ۲ خوشی را محاسبه کنیم. تک تک نمونه‌ها امتحان خواهند شد. برای مثال نمونه‌ی از میان دو نمونه‌ی  $A$  و  $B$  به نمونه‌ی  $B$  نزدیکتر است، لذا  $C$  و  $B$  در یک خوشی قرار خواهند گرفت. بدین ترتیب پس از خوشبندی مرحله‌ی اول داریم:

$$Cluster_1 = \{A, D, E, F\} \quad , \quad Cluster_2 = \{B, C\}$$

پس از آن میانگین نمونه‌ها در هر خوش محاسبه می‌شوند.

$$Mean_I = [(4+3+3+2)/4, (2+2+1+2)/4] = (3, 1.75)$$

$$Mean_2 = [(6+8)/2, (4+8)/2] = (7, 6)$$

نمونه‌های  $Mean_1$  و  $Mean_2$  نمایندگان جدید خوش‌ها محسوب می‌شوند. توجه کنید که این نمونه‌ها در جدول اولیه به عنوان داده‌های اصلی (جدول ۸-۲) وجود ندارند. در این مرحله باید تشابه هر ۶ نمونه را با این دو نماینده‌ی جدید اندازه‌گیری کنیم، تا در صورت نیاز خوش‌های  $Cluster_1$  و  $Cluster_2$  اصلاح شوند.

$$Distance(Mean_1, A) = 1.03, Distance(Mean_2, A) = 5.00 \mapsto A \in Cluster_1$$

$$Distance(Mean_1, B) = 3.75, Distance(Mean_2, B) = 2.23 \mapsto B \in Cluster_2$$

$$Distance(Mean_1, C) = 8.00, Distance(Mean_2, C) = 2.23 \mapsto C \in Cluster_2$$

$$Distance(Mean_1, D) = 0.25, Distance(Mean_2, D) = 5.65 \mapsto D \in Cluster_1$$

$$Distance(Mean_1, E) = 0.75, Distance(Mean_2, E) = 6.40 \mapsto E \in Cluster_1$$

$$Distance(Mean_1, F) = 1.03, Distance(Mean_2, F) = 6.40 \mapsto F \in Cluster_1$$

در این مثال و در این مرحله اعضای خوشها هیچ تغییری نمی‌کنند. به عبارت دیگر هیچ‌گونه انتساب مجدد و متفاوتی صورت نخواهد گرفت و لذا الگوریتم خاتمه می‌یابد. چنانچه نمونه‌ای از یک خوش به خوشی دیگر منتقل می‌شده، با است محاسبه‌ی میانگین

و فاصله‌ی میان نمونه‌ها و آنها از سر گرفته می‌شد. همانطور که قبل از این نیز به آن اشاره شد، این عمل تا محقق شدن یک شرط پایانی ادامه دارد.

با کمی دقت می‌توان متوجه شد که پیچیدگی زمانی این الگوریتم برابر است با  $O(nkt)$ ، که در آن  $n$  معرف تعداد نمونه‌ها،  $k$  تعداد خوش‌ها و  $t$  تعداد تکرارهایی است که الگوریتم پس از آن خاتمه می‌یابد. روشن است که تعداد خوش‌ها و تعداد تکرارها بسیار کوچکتر از تعداد نمونه‌ها هستند. لذا می‌توان ادعا نمود که این الگوریتم در کار با حجم بالای داده‌ها نسبتاً مقیاس‌پذیر و کارآ است.

ضرورت مشخص نمودن تعداد خوش‌ها در ابتدای الگوریتم هم عیب و هم حسن آن به شمار می‌رود. در برخی از کاربردهای عملی مایلیم تا تعداد خوش‌های نهایی مشخص باشند، ولی اغلب کاربر دید روشنی از تعداد خوش‌ها ندارد و پسند کاربر این است که الگوریتم خود به صورت خودکار این تعداد را بدست آورد. از آنجا که در هر مرحله با محاسبه‌ی میانگین نمونه‌ها سروکار داریم، بنابراین برای ویژگی‌هایی که محاسبه‌ی میانگین برای آنها دارای مفهومی نیست، این الگوریتم مناسب نیست. به علاوه از آنجا که داده‌های نویز و خارج از محدوده مانند داده‌هایی با مقدار خیلی کم و یا زیاد بر روی مقدار میانگین اثرگذار است، این الگوریتم در مواجهه با این نوع داده‌ها عملکرد مناسبی از خود نشان نمی‌دهد و به عبارتی دیگر مقاوم نیست. این روش برای کشف خوش‌هایی که دارای شکل‌های غیرکروی و در اندازه‌هایی بسیار متفاوت هستند نیز مناسب نیست. پیچیدگی فضای مورد نیاز آن  $O(k+n)$  می‌باشد و چنانچه ذخیره‌سازی تمام نمونه‌ها در حافظه‌ی اصلی امکان‌پذیر باشد، زمان دسترسی به تمام نمونه‌ها خیلی سریع بوده و الگوریتم بسیار کارآمد است.

به منظور بهبود الگوریتم  $k$ -Means پیشنهادهای متعددی وجود دارند. یکی از این پیشنهادها که معمولاً با نتیجه‌ی قابل قبولی نیز روبروست، استفاده از الگوریتم‌های خوش‌بندی سلسله‌مراتبی برای بدست آوردن مقدار  $k$  است و پس از به دست آمدن خوش‌بندی ابتدایی، تکرارها در خوش‌های بدست آمده می‌شود.

پیشنهاد دیگر استفاده از مُد و میانه به جای محاسبه‌ی میانگین بین نمونه‌های موجود در داده‌ها است. این روش‌ها که به ترتیب با نام‌های *k-Medians* و *k-Modes* شناخته می‌شوند و همانطور که از نامشان مشخص است با محاسبه‌ی مُد و میانه‌ی نمونه‌ها مرکز ثقل خوش‌ها را بدست می‌آورند. استفاده از مُد باعث می‌شود داده‌ها به نوع داده‌های عددی محدود نباشند و بتوانیم ویژگی‌های طبقه‌بندی شده‌ی اسمی و ترتیبی را نیز در روش مزبور استفاده کنیم.

#### ۲-۴-۸) الگوریتم *k-Medoids*

الگوریتم *k-Medoids* عملکردی بسیار شبیه به الگوریتم *k-Means* دارد، با این تفاوت که در الگوریتم *k-Medoids* به جای استفاده از میانگین، از خود نمونه‌ها برای مرکز‌ثقل و نمایندگی خوش‌ها استفاده می‌شود. با انتخاب نمونه‌های واقعی جهت نمایش یک خوش، حساسیت روش نسبت به نمونه‌های نویز و خارج از محدوده کاهش می‌یابد. فراموش نکنید که الگوریتم *k-Means* به دلیل اینکه حتی تعداد کمی از این داده‌ها می‌تواند در مقدار میانگین تأثیر بگذارد، الگوریتم به این گونه از داده‌ها بسیار حساس است. بنابراین روش *k-Medoids* برخلاف *k-Means* به جای اینکه مقادیر میانگین از نمونه‌ها را دریافت کند، از مرکزی‌ترین نمونه‌ی موجود در خوش به عنوان نمایش و نماینده‌ی خوش استفاده می‌کند. به همین دلیل این الگوریتم حساسیت کمی نسبت به داده‌های خارج از محدوده از خود نشان می‌دهد.

اجازه دهید با توضیحات بیشتر با این الگوریتم آشنا شویم، همانند *k-Means* در ابتدا باید مقدار  $k$  را مشخص کنید. پس از آن تعداد  $k$  نمونه به عنوان نماینده‌های اولیه‌ی  $k$  خوش به صورت اتفاقی انتخاب می‌شوند. پس از تشکیل ماتریس تشابه، هر یک از نمونه‌های باقیمانده( $n-k$  نمونه) باید در یکی از این  $k$  خوش قرار گیرند. توجه کنید که می‌توانیم به جای تشکیل ماتریس تشابه، فاصله‌ی هر یک از نمونه‌های باقیمانده را با نمونه‌ی اولیه محاسبه کنیم. هر نمونه به نزدیکترین نماینده تعلق دارد. تا این مرحله از الگوریتم تفاوتی میان *k-Medoids* و *k-Means* مشاهده نمی‌شود.

پس از این با جایگزینی یک نمونه از داده‌ها با یکی از  $k$  نمونه‌ی نماینده، کیفیت و مناسب بودن خوش‌های بدست آمده از این جایگزینی بررسی می‌شوند. در صورت بهبود در نتایج، مجاز به جایگزینی نماینده‌ی مزبور خواهیم بود. اما این بهبود چگونه ارزیابی و محاسبه می‌شود؟ یکی از معیارهای مناسب خطای مطلق است، که با فرمول زیر محاسبه می‌شود.

$$E = \sum_{i=1}^k \sum_{p \in c_i} |p - O_i|$$

در فرمول  $E$  مجموع خطای مطلق برای کلیه‌ی نمونه‌ها،  $p$  یک نمونه از داده‌ها و  $O_i$  نماینده‌ی خوش‌ی  $c_i$  است.

برای تعیین یک نماینده‌ی جدید  $O_{new}$  به جای نماینده‌ی کنونی  $O_{old}$  یک خوش‌ه، عملیات زیر برای هر نمونه‌ی غیر نماینده مثل  $p$  باید انجام شود.

- اگر  $p$  متعلق به خوش‌های است که  $O_{old}$  نماینده‌ی آن خوش‌ه باشد:

چنانچه با جایگزینی  $O_{new}$  به جای  $O_{old}$  فاصله‌ی  $p$  به نماینده‌ی جدید یعنی  $O_{new}$  نزدیکتر باشد،  $p$  را در خوش‌های به نماینده‌ی  $O_{new}$  قرار دهید. در غیر اینصورت  $p$  را در خوش‌ی  $i$  قرار دهید، به نحوی که فاصله‌ی نماینده‌ی آن خوش‌ه با  $p$  حداقل و  $old \neq i$  باشد.

- اگر  $p$  متعلق به خوش‌هی  $i$  است که  $O_i$  نماینده‌ی آن خوش‌ه باشد:

چنانچه با جایگزینی  $O_{new}$  به جای  $O_{old}$  همچنان فاصله‌ی  $p$  با نماینده‌ی خوش‌هی خود یعنی  $O_i$  حداقل است، پس هیچگونه انتساب جدیدی صورت نمی‌پذیرد. در غیر اینصورت  $p$  در خوش‌های قرار می‌گیرد که نماینده‌ی آن  $O_{new}$  است. فراموش نکنید در اینصورت نیز  $old \neq i$  خواهد بود.

هر زمان که با جایگزینی  $O_{new}$  به جای  $O_{old}$  نمونه‌های  $p$  به خوش‌هی دیگری منتقل می‌شوند، مقدار خطای مطلق نیز تغییر می‌کند. واضح است که کاهش در مقدار خطای مطلق، جایگزینی نماینده‌ی جدید  $O_{new}$  را تأیید می‌کند. این عمل برای  $k$  نماینده انجام می‌شود تا جایی که هیچ یک از نمونه‌ها خوش‌هی خود را با جایگزینی نماینده تغییر ندهند.

یکی از اولین الگوریتم‌های  $k$ -Medoids با نام  $PAM$  معرفی شد. در این الگوریتم پس از انتخاب  $k$  نماینده به صورت اتفاقی از میان نمونه‌ها، با تکرار سعی می‌شود تا نمایندگان بهتری برای خوشها انتخاب شوند. جایگزینی نمونه‌ها و نماینده‌ها در صورتی انجام می‌شود که بیشترین کاهش را در مقدار خطا داشته باشیم. در این الگوریتم کلیه‌ی ترکیبات دوتایی از نمونه‌ها ارزیابی می‌شوند، به صورتی که یکی از نمونه‌ها نماینده یا مرکز ثقل باشد. مجموعه‌ی بهترین نمونه‌های هر خوش در یک تکرار، نمایندگان خوشها برای تکرار بعدی را شکل می‌دهند. پیچیدگی زمانی هر تکرار در این الگوریتم برابر با  $O(k(n-k)^2)$  خواهد بود که برای مقادیر بزرگ  $n$  و  $k$  می‌تواند پرهزینه باشد.

## ۵-۸) تکنیک‌های خوشه‌بندی سلسله‌مراتبی

برخلاف تکنیک‌های خوشه‌بندی مبتنی بر افزار که تعداد خوشه‌ها به عنوان پارامتر ورودی توسط کاربر مشخص می‌شود، در تکنیک‌های سلسله‌مراتبی مجموعه داده‌ها و معیاری جهت ارزیابی تشابه به عنوان ورودی معین می‌شوند. بسته به اینکه تحلیل این ساختار سلسله‌مراتبی از پایین به بالا<sup>۱</sup> و یا بر عکس از بالا به پایین<sup>۲</sup> انجام شود، عملیات اصلی این الگوریتم‌ها را می‌توان در دو دسته‌ی ادغام<sup>۳</sup> و تقسیم<sup>۴</sup> قرار داد. فرایند ادغام و یا تقسیم چند خوشه در این تکنیک‌ها نقش مهمی را ایفا می‌کند. چرا که در صورت اخذ تصمیمی ضعیف جهت ادغام و تقسیم خوشه‌ها، تکنیک توانایی برگشت و اصلاح آن را ندارد و این عمل باعث کاهش کیفیت در خوشه‌های تولید شده‌ی نهایی خواهد شد.

---

<sup>1</sup> Bottom-up

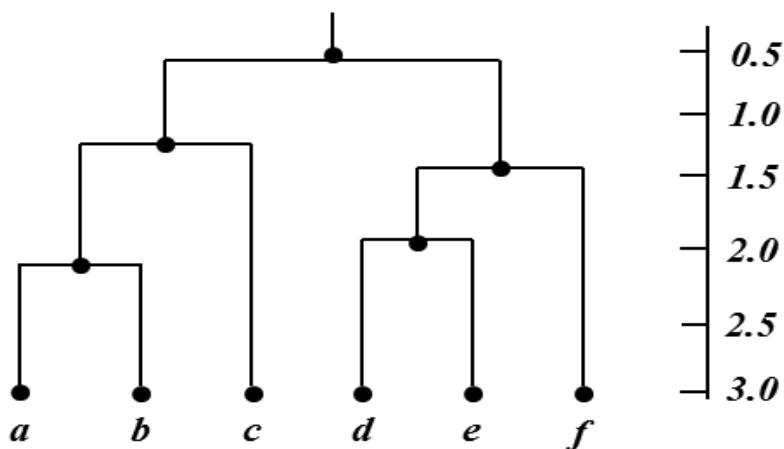
<sup>2</sup> Top-down

<sup>3</sup> Agglomerative

<sup>4</sup> Divisive

الگوریتم‌های خوشبندی سلسله مراتبی که یک فرایند پایین به بالا را طی می‌کنند، کارشان با قراردادن هر نمونه داده در یک خوشبندی مجزا شروع می‌شود. با ادغام خوشبندی‌ها الگوریتم تا جایی پیش می‌رود که یا کلیه‌ی نمونه‌ها در یک خوشبندی قرار گیرند و یا شرط از پیش تعیین شده‌ای به عنوان پایان اجرا مشخص شده باشد. اغلب تکنیک‌های موجود سلسله‌مراتبی متعلق به این دسته هستند.

در طرف دیگر الگوریتم‌های سلسله مراتبی قرار دارند که یک استراتژی بالا به پایین را دنبال می‌کنند. در این روش‌ها بر عکس تکنیک‌های قبلی در ابتدا کلیه‌ی نمونه‌ها در یک خوشبندی قرار می‌گیرند. پس از آن با کمک از یک معیار تشابه در چند مرحله بصورت سلسله‌مراتبی این خوشبندی به خوشبندی‌های کوچکتر تقسیم می‌شود. در این الگوریتم‌ها نیز می‌توان کار را تا جایی ادامه داد تا هر نمونه در یک خوشبندی قرار بگیرد و یا اینکه شرطی را جهت پایان اجرای الگوریتم معین نمود. در هر دو دسته از الگوریتم‌های سلسله‌مراتبی کاربر می‌تواند تعداد خوشبندی‌های تولید شده‌ی نهایی را به عنوان یک شرط پایانی مشخص کند. معمولاً<sup>۱</sup> فرایند خوشبندی سلسله‌مراتبی توسط یک نمودار با نام دنдрوگرام<sup>۱</sup> نمایش داده می‌شود که مثالی از آن را در شکل ۸-۵ ملاحظه می‌کنید.



شکل ۸-۵: نمونه‌ای از یک دندروگرام جهت نمایش خوشبندی سلسله‌مراتبی

<sup>۱</sup> Dendrogram

این نمودار ادغام و تقسیم خوشها را در هر مرحله نمایش می‌دهد. روش‌های مختلف دیگری نیز وجود دارند که می‌توان با کمک آنها فرایند خوشبندی سلسله‌مراتبی را نمایش داد.

محور عمودی کنار نمودار، مقادیر مقیاس تشابه میان خوشها را نشان می‌دهد. برای مثال همانطور که مشاهده می‌کنید هنگامی که تشابه میان خوشها  $\{d,e,f\}$  و  $\{a,b,c\}$  تقریباً برابر با  $5/0$  است، این دو خوش با یکدیگر ادغام می‌شوند.

#### ۱-۵-۸) معیارهای تشابه میان خوشها

در تکنیک‌های مبتنی بر افزای داده‌ها معیارهای تشابه میان دو نمونه از داده‌ها محاسبه می‌شد و این در حالی است که در تکنیک‌های سلسله‌مراتبی معیارهایی باید وجود داشته باشند تا تشابه و نزدیکی میان خوشها را نیز ارزیابی کنند. اغلب الگوریتم‌های خوشبندی سلسله‌مراتبی دارای تنوع زیادی از این معیارها هستند که در ادامه به ذکر بعضی از آنها می‌پردازیم.

#### انتخاب دو نمونه با بیشترین تشابه

یکی از ساده‌ترین روش‌های محاسبه‌ی تشابه میان دو خوش در نظر گرفتن دو نمونه با بیشترین تشابه (کمترین فاصله) است به شرطی که هر دوی آنها متعلق به یک خوش نباشند. در این روش که برخی موقع با نام‌های دیگری از جمله تکنیک پیوند منفرد<sup>۱</sup> شناخته می‌شود، حداقل فاصله بین همه‌ی زوج نمونه‌های دو خوشه (یک نمونه از یک خوشه و نمونه‌ی دوم از خوشه‌ی دیگر) محاسبه می‌شود. توجه داشته باشید در این حالت نمونه‌های خوشها همپوشانی ندارند. این بدین معنی است که هر نمونه فقط متعلق به یک خوشه است و نمونه‌ای که بتواند همزمان در دو یا چند خوشه قرار داشته باشد، وجود ندارد.

---

<sup>۱</sup> Single-link

فرمول‌های زیر این روش را به خوبی بیان می‌کند.

$$\text{Distance}(C_i, C_j) = \min_{O_i \in C_i, O_j \in C_j} \{\text{Distance}(O_i, O_j)\} = \max_{O_i \in C_i, O_j \in C_j} \{\text{Sim}(O_i, O_j)\}$$

$$\text{Distance}(C_k, C_i \cup C_j) = \min\{\text{Distance}(C_k, C_i), \text{Distance}(C_k, C_j)\}$$

اجازه دهید اجرای این تکنیک خوشبندی سلسله‌مراتبی را با مثالی که داده‌های آن در جدول ۸-۶ مشخص شده است، دنبال کنیم.

جدول ۸-۶: داده‌های آزمایشی با تعداد ۵ نمونه و ۲ صفت خاصه

| Attribute | O <sub>1</sub> | O <sub>2</sub> | O <sub>3</sub> | O <sub>4</sub> | O <sub>5</sub> |
|-----------|----------------|----------------|----------------|----------------|----------------|
| X         | 1              | 5              | 6              | 1              | 4              |
| Y         | 2              | 2              | 1              | 1              | 1              |

همانطور که از محتوای جدول روشن است، ما دارای ۵ نمونه داده هستیم که هر یک از آنها با ۲ ویژگی X و Y توصیف می‌شوند(مشخصات ۵ نقطه در فضای دو بعدی). در ابتدا هر یک از این نمونه‌ها معرف یک خوش هستند و الگوریتم کار خود را با ۵ خوش شروع می‌کند. به منظور تعیین این مطلب که کدامیک از خوش‌ها شرایط ادغام را دارند، ماتریس تشابه را تشکیل می‌دهیم (شکل ۸-۶).

|                | O <sub>1</sub> | O <sub>2</sub> | O <sub>3</sub> | O <sub>4</sub> | O <sub>5</sub> |
|----------------|----------------|----------------|----------------|----------------|----------------|
| O <sub>1</sub> | 0              |                |                |                |                |
| O <sub>2</sub> | 4.0            | 0              |                |                |                |
| O <sub>3</sub> | 5.1            | 1.4            | 0              |                |                |
| O <sub>4</sub> | 1.0            | 4.1            | 5.0            | 0              |                |
| O <sub>5</sub> | 3.1            | 1.4            | 2.0            | 3.0            | 0              |

شکل ۸-۶: ماتریس تشابه برای داده‌های جدول ۸-۶

معیار ارزیابی تشابه فاصله‌ی اقلیدسی است. بنابراین مقادیر ماتریس تشابه در واقع فاصله و یا عدم تشابه زوج نمونه‌ها را نشان می‌دهد. این بدین معنی است که عدد کوچکتر نشان دهنده‌ی تشابه بالای میان ۲ نمونه است.

در میان مقادیر موجود در ماتریس تشابه کوچکترین عدد یک است که نشان از تشابه بالای نمونه‌های  $O_1$  و  $O_4$  می‌باشد. بنابراین در این مرحله این دو نمونه می‌توانند در یک خوشه قرار گیرند و به عبارتی دیگر ادغام می‌شوند. جهت اصلاح ماتریس تشابه محاسبات زیر انجام می‌شود.

$$\text{Distance}(\{O_1, O_4\}, O_2) = \text{Min}\{d(O_1, O_2), d(O_4, O_2)\} = \text{Min}\{4.0, 4.1\} = 4$$

$$\text{Distance}(\{O_1, O_4\}, O_3) = \text{Min}\{d(O_1, O_3), d(O_4, O_3)\} = \text{Min}\{5.1, 5.0\} = 5$$

$$\text{Distance}(\{O_1, O_4\}, O_5) = \text{Min}\{d(O_1, O_5), d(O_4, O_5)\} = \text{Min}\{3.1, 3.0\} = 3$$

با توجه به مقادیر بدست آمده از محاسبات فوق ماتریس تشابه را بازسازی می‌کنیم (شکل ۸-۷).

|                |                | $\{O_1, O_4\}$ | $O_2$ | $O_3$ | $O_5$ |
|----------------|----------------|----------------|-------|-------|-------|
| $\{O_1, O_4\}$ | $\{O_1, O_4\}$ | 0              |       |       |       |
|                | $O_2$          | 4.0            | 0     |       |       |
| $O_3$          | 5.0            | 1.4            | 0     |       |       |
| $O_5$          | 3.0            | 1.4            | 2.0   | 0     |       |

شکل ۸-۷: ماتریس تشابه پس از اجرای مرحله‌ی اول

در این مرحله دو مقدار  $1/4$  در ماتریس نشان دهنده‌ی نزدیکی نمونه‌های  $O_3$  و  $O_5$  به نمونه‌ی  $O_2$  است. می‌توان این سه نمونه را در یک خوشه ادغام کرد و کار را ادامه داد. اما اگر شرط الگوریتم این است که در هر مرحله فقط یکی از نمونه‌ها با نمونه یا خوشه دیگر ادغام شود، مجبور خواهیم بود تا نمونه‌ی  $O_2$  را با یکی از نمونه‌های  $O_3$  یا  $O_5$  ادغام کنیم.

$$\text{Distance}(\{O_1, O_4\}, \{O_2, O_3\}) = \text{Min}\{d(O_1, O_2), d(O_1, O_3), d(O_4, O_2), d(O_4, O_3)\} = 4$$

$$\text{Distance}(\{O_1, O_4\}, O_5) = \text{Min}\{d(O_1, O_5), d(O_4, O_5)\} = 3$$

$$\text{Distance}(\{O_2, O_3\}, O_5) = \text{Min}\{d(O_2, O_5), d(O_3, O_5)\} = 1.4$$

ما نمونه‌ی  $O_2$  را با  $O_3$  ادغام و پس از آن ماتریس تشابه را بروزرسانی می‌کنیم (شکل ۸-۸).

|                | $\{O_1, O_4\}$ | $\{O_2, O_3\}$ | $O_5$ |
|----------------|----------------|----------------|-------|
| $\{O_1, O_4\}$ | 0              |                |       |
| $\{O_2, O_3\}$ | 4.0            | 0              |       |
| $O_5$          | 3.0            | 1.4            | 0     |

شکل ۸-۸: ماتریس تشابه پس از اجرای مرحله‌ی دوم

با توجه به محتوای ماتریس تشابه در این مرحله نمونه‌ی  $O_5$  با خوشبختی  $\{O_2, O_3\}$  ادغام خواهد شد و ماتریس تشابه پس از این ادغام اصلاح می‌شود (شکل ۸-۹). اما فاصله همان مقدار  $1/4$  را نشان می‌دهد.

|                     | $\{O_1, O_4\}$ | $\{O_2, O_3, O_5\}$ |
|---------------------|----------------|---------------------|
| $\{O_1, O_4\}$      | 0              |                     |
| $\{O_2, O_3, O_5\}$ | 3.0            | 0                   |

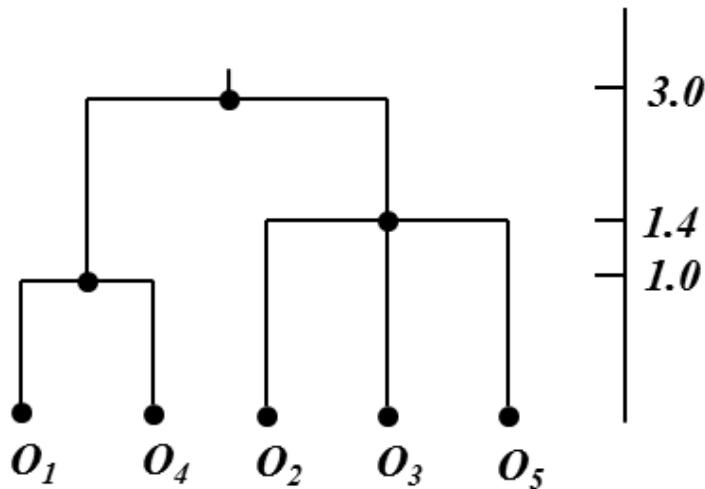
شکل ۸-۹: ماتریس تشابه پس از اجرای مرحله‌ی سوم

بالاخره در مرحله‌ی نهایی دو خوشبختی ادغام می‌شوند. در واقع کلیه‌ی نمونه‌ها در یک خوشبختی قرار می‌گیرند. نمودار دندرگرام این مثال در شکل ۸-۱۰ نشان داده شده است.

### انتخاب دو نمونه با کمترین تشابه

برخلاف روش قبلی که به منظور تصمیم‌گیری در مورد ادغام دو خوشبختی، دو نمونه با بیشترین تشابه را معیار خود قرار می‌داد، در این روش ملاک ارزیابی ادغام دو خوشبختی، دو نمونه‌ای هستند که بیشترین فاصله و یا به عبارتی کمترین تشابه را دارند. فراموش نکنید که منظور دو نمونه‌ای هستند که هر یک متعلق به خوشبختی‌های متفاوت است. این تکنیک با نام‌های پیوند کامل<sup>۱</sup> و دورترین همسایه<sup>۱</sup> نیز شناخته می‌شود.

<sup>۱</sup> Complete-link



شکل ۸-۱۰: نمودار دندروگرام برای خوشبندی داده‌های جدول ۸-۶

فرمول‌های زیر عملکرد این روش را نشان می‌دهند.

$$\text{Distance}(C_i, C_j) = \max_{O_i \in C_i, O_j \in C_j} \{\text{Distance}(O_i, O_j)\} = \min_{O_i \in C_i, O_j \in C_j} \{\text{Sim}(O_i, O_j)\}$$

$$\text{Distance}(C_k, C_i \cup C_j) = \max\{\text{Distance}(C_k, C_i), \text{Distance}(C_k, C_j)\}$$

فراموش نکنید که فرمول‌های فوق جهت ادغام خوشبندی استفاده می‌شوند.

اجرای این تکنیک را با کمک داده‌های جدول ۸-۶ و ماتریس تشابه شکل ۸-۶ دنبال می‌کنیم. مرحله‌ی اول الگوریتم همانند روش قبلی ادغام دو نمونه‌ی  $O_1$  و  $O_4$  با  $O_4$  کمترین فاصله و یا با بیشترین تشابه است. پس از آن برای محاسبه‌ی فاصله میان خوشه حاوی نمونه‌های فوق و ۳ نمونه‌ی دیگر محاسباتی لازم است که در ادامه نوشته شده‌اند.

$$\text{Distance}(\{O_1, O_4\}, O_2) = \max\{d(O_1, O_2), d(O_4, O_2)\} = \max\{4.0, 4.1\} = 4.1$$

$$\text{Distance}(\{O_1, O_4\}, O_3) = \max\{d(O_1, O_3), d(O_4, O_3)\} = \max\{5.1, 5.0\} = 5.1$$

$$\text{Distance}(\{O_1, O_4\}, O_5) = \max\{d(O_1, O_5), d(O_4, O_5)\} = \max\{3.1, 3.0\} = 3.1$$

---

<sup>۱</sup> Furthest Neighbor

در این مرحله ماتریس تشابه اصلاح می‌شود (شکل ۸-۱۱). می‌توانید تفاوت‌های این ماتریس و شکل ۸-۷ را مشاهده کنید.

|                | $\{O_1, O_4\}$ | $O_2$ | $O_3$ | $O_5$ |
|----------------|----------------|-------|-------|-------|
| $\{O_1, O_4\}$ | 0              |       |       |       |
| $O_2$          | 4.1            | 0     |       |       |
| $O_3$          | 5.1            | 1.4   | 0     |       |
| $O_5$          | 3.1            | 1.4   | 2.0   | 0     |

شکل ۸-۱۱: ماتریس تشابه پس از اجرای مرحله‌ی اول

در ماتریس کوچکترین عدد  $1/4$  است که مربوط به تشابه میان نمونه  $O_2$  با نمونه‌های  $O_3$  و  $O_5$  است. همانند مثال روش قبل در این لحظه بدلیل تشابه یکسان با دو انتخاب روبرو هستیم. بدلیل آنکه نشان دهیم الگوریتم می‌تواند در هر مرحله چندین نمونه را در یک خوشی واحد قرار دهد، سه نمونه را در این مرحله در یک خوشی ادغام می‌کنیم. ماتریس تشابه نهایی نیز در شکل ۸-۱۲ آمده است.

|                     | $\{O_1, O_4\}$ | $\{O_2, O_3, O_5\}$ |
|---------------------|----------------|---------------------|
| $\{O_1, O_4\}$      | 0              |                     |
| $\{O_2, O_3, O_5\}$ | 5.1            | 0                   |

شکل ۸-۱۲: ماتریس تشابه پس از اجرای مرحله‌ی نهایی

در مرحله پایانی همانند الگوریتم قبل اعضاء دو خوشی باقیمانده در یک خوشی ادغام می‌شوند. نمودار دندروگرام این مثال مانند شکل ۸-۱۰ است، به جز اینکه اعدادی که بر روی محور عمودی کنار نمودار نوشته شده است، متفاوت خواهد بود.

### میانگین تشابه (عدم تشابه) میان زوج نمونه‌ها

در دو روش قبل به منظور محاسبه‌ی تشابه میان دو خوشی، تشابه میان یک نمونه از هر خوشی را ملاک ارزیابی قرار دادیم. در روش اول دو نمونه‌ای که نزدیکترین فاصله (بیشترین تشابه) و در روش دوم دو نمونه‌ای که دورترین فاصله (کمترین تشابه) را

داشتند انتخاب و با محاسبه‌ی معیار ارزیابی تشابه، الگوریتم را اجرا کردیم. در روش میانگین که با نام مختصر<sup>۱</sup> *UPGMA* نیز شناخته می‌شود، باید میانگین کلیه‌ی فواصل زوج نمونه‌ها را طوری که هر نمونه متعلق به یک خوش است را محاسبه کنیم. فرمول-های زیر این مسئله را به خوبی نشان می‌دهند.

$$Distance(C_i, C_j) = \frac{1}{|C_i| \times |C_j|} \times \sum_{O_i \in C_i, O_j \in C_j} Distance(O_i, O_j)$$

$$Distance(C_k, C_i \cup C_j) = \frac{|C_i|}{|C_i| \times |C_j|} \times Distance(C_k, C_i) + \frac{|C_j|}{|C_i| \times |C_j|} \times Distance(C_k, C_j)$$

در فرمول‌های فوق منظور از  $C_i /$  تعداد نمونه‌هایی است که در خوشی  $C_i$  قرار دارند. با فرض وجود ماتریس تشابه زیر میان ۴ نمونه این روش را دنبال می‌کنیم (شکل ۸-۱۳).

|       | $O_1$ | $O_2$ | $O_3$ | $O_4$ |
|-------|-------|-------|-------|-------|
| $O_1$ | 0     |       |       |       |
| $O_2$ | 1     | 0     |       |       |
| $O_3$ | 11    | 2     | 0     |       |
| $O_4$ | 5     | 3     | 4     | 0     |

شکل ۸-۱۳: ماتریس تشابه برای ۴ نمونه

با توجه به محتوای ماتریس نمونه‌های  $O_1$  و  $O_2$  در مرحله‌ی اول ادغام می‌شوند. پس از آن چنانچه ماتریس تشابه را تشکیل دهید، متوجه خواهید شد که نمونه‌های  $O_3$  و  $O_4$  می‌توانند در مرحله‌ی بعدی ادغام شوند. در نهایت دو خوشی  $\{O_1, O_2\}$  و  $\{O_3, O_4\}$  در یک خوش قرار می‌گیرند. جهت فهم بهتر در ادامه طریقه‌ی محاسبه‌ی فاصله میان این دو خوشی پایانی بیان شده است.

$$\begin{aligned} Distance(\{O_1, O_2\}, \{O_3, O_4\}) &= (1/4) \times (d(O_1, O_3) + d(O_1, O_4) \\ &\quad + d(O_2, O_3) + d(O_2, O_4)) = (1/4) \times (11 + 5 + 2 + 3) = 5.25 \end{aligned}$$

<sup>۱</sup> Unweighted Pair Group Method using Arithmetic Averages

اگر حدس می‌زنید اندازه‌ی خوش‌های تولید شده توسط تکنیک خوش‌بندی به شدت نابرابر است، می‌توانید از روشی موسوم به<sup>۱</sup> *WPGMA* استفاده کنید که کاملاً شبیه به *UPGMA* است، با این تفاوت که در الگوریتم *WPGMA* می‌توانیم برای هر خوش وزنی را در محاسبه‌ی میانگین متصور شویم. تعداد نمونه‌های موجود در هر خوش می‌تواند مشخص کننده‌ی وزن هر خوش باشد. فرمول زیر فاصله‌ی میان خوش‌ها را در حالی نشان می‌دهد که خوش‌های  $C_i$  و  $C_j$  در یک سطح از تکنیک خوش‌بندی (در مرحله‌ی یکسانی از اجرای الگوریتم) قرار دارند.

$$\text{Distance}(C_k, C_i \cup C_j) = \frac{1}{2} \times \text{Distance}(C_k, C_i) + \frac{1}{2} \times \text{Distance}(C_k, C_j)$$

### روش‌های مبتنی بر نمایندگان خوش‌ها

در این الگوریتم‌ها با محاسبه‌ی فاصله‌ی میان مراکز ثقل و نمایندگان دو خوش، تشابه میان دو خوش بررسی می‌شود. همانطور که در روش خوش‌بندی *k-Medoids* توضیح داده شد، از مرکزی‌ترین نمونه‌ی موجود در خوش به عنوان نماینده‌ی خوش استفاده کردیم. حال به منظور مقایسه‌ی دو خوش می‌توانیم از این نمونه‌ها استفاده کنیم. در واقع فاصله‌ی میان این نمونه‌ها معرف عدم تشابه و یا تشابه دو خوش خواهد بود. این روش با نام مختصر<sup>۲</sup> *UPGMC* نیز شناخته می‌شود.

$$\begin{aligned} \text{Distance}(C_i, C_j) &= \frac{1}{|C_i| \times |C_j|} \times \sum_{O_i \in C_i, O_j \in C_j} \text{Distance}(O_i, O_j) \\ &\quad - \frac{1}{2|C_i|^2} \sum_{O_i, O_j \in C_i} \text{Distance}(O_i, O_j) \\ &\quad - \frac{1}{2|C_j|^2} \sum_{O_i, O_j \in C_j} \text{Distance}(O_i, O_j) \end{aligned}$$

در این الگوریتم چنانچه دو خوش‌ای که ادغام می‌شوند، دارای اندازه‌ی بسیار متفاوتی باشند، مرکز ثقل خوش‌ی جدید بسیار به خوش‌های با اندازه‌ی بزرگتر نزدیک خواهد بود و شاید همان نماینده بدون تغییر باقی بماند.

<sup>1</sup> Weighted Pair Group Method using Arithmetic Averages

<sup>2</sup> Unweighted Pair Group Method using the Centroids

روشی مبتنی بر میانه با نام مختصر<sup>۱</sup>  $WPGMC$  وجود دارد که همانند روش بالا عمل می‌کند، به جز اینکه با فرض ورود وزن خوشه‌ها ممکن است عملکرد بهتری نسبت به روش بالا داشته باشد. در این روش فاصله‌ی میان خوشه‌های ادغام شده و دیگر خوشه‌ها از فرمول زیر محاسبه می‌شود:

$$\begin{aligned} Distance(C_k, C_i \cup C_j) = & \frac{1}{2} Distance(C_k, C_i) + \frac{1}{2} Distance(C_k, C_j) \\ & - \frac{1}{4} Distance(C_i, C_j) \end{aligned}$$

در فرمول فوق به سه خوشه‌ای اشاره می‌شود که در سلسله مراتب خوشه‌ها هم‌سطح هستند.

---

<sup>۱</sup> Weighted Pair Group Method using the Centroids(Median)

<sup>۲</sup> Balanced Iterative Reducing and Clustering using Hierarchies