



مباحثه و تکنیک‌های داده‌گاوی

نویسنده: مهدی اسماعیلی

فهرست مطالب

فهرست شکل‌ها

فهرست جدول‌ها

پیش‌گفتار

فصل اول: مقدمه

۲۶	۱-۱) استخراج دانش
۳۰	۱-۲) انواع داده‌ها جهت داده‌کاوی
۳۳	۱-۳) داده‌کاوی و تکنیک‌های آن
۳۶	۱-۳-۱) قوانین انجمنی
۳۷	۱-۳-۲) طبقه‌بندی
۳۸	۱-۳-۳) خوش‌بندی
۴۰	۱-۴) چالش‌های داده‌کاوی

۴۱	۱-۵) سازماندهی کتاب
۴۳	خلاصه فصل
۴۴	مراجع و منابع جهت مطالعه‌ی بیشتر

فصل دوم: آماده‌سازی داده‌ها

۴۸	۲-۱) انواع داده‌ها و خصوصیات آنها
۴۸	۲-۱-۱) متغیرهای کمی
۴۸	۲-۱-۲) متغیرهای کیفی
۵۰	۲-۲) ابعاد بالای داده‌ها
۵۲	۲-۳) آمار
۵۲	۲-۳-۱) شاخص‌های مرکزی
۵۲	میانگین
۵۴	میانه
۵۴	نما یا مُد
۵۵	۲-۳-۲) شاخص‌های پراکندگی
۵۵	دامنه‌ی تغییرات و میانگین انحرافات
۵۶	انحراف چارکی
۵۶	واریانس و انحراف استاندارد
۵۶	۲-۳-۳) کوواریانس و ضریب همبستگی
۵۸	۲-۳-۴) توزیع نرمال
۵۹	۲-۴) لزوم آماده‌سازی داده‌ها
۶۲	۲-۵) تکنیک‌های آماده‌سازی داده‌ها
۶۳	۲-۵-۱) پالایش داده‌ها
۶۴	تحلیل داده‌های خارج از محدوده
۶۶	دسته‌بندی داده‌ها
۶۹	خوشبندی

۷۰	رگرسیون
۷۲	جمع آوری داده‌ها (۲-۵-۲)
۷۳	تغییر شکل داده‌ها (۲-۵-۳)
۷۴	کاهش داده‌ها (۲-۵-۴)
۷۵	کاهش صفات خاصه (۲-۵-۴-۱)
۷۸	بررسی صفات خاصه بصورت مجزا برای کاهش ابعاد
۸۰	محاسبه‌ی آنتروپی برای رتبه‌بندی صفات خاصه
۸۲	تحلیل مولفه‌های اصلی (PCA)
۸۴	کاهش نمونه‌ها (۲-۵-۴-۲)
۸۴	تکنیک‌های نمونه‌گیری
۸۶	کاهش مقادیر یک صفت خاصه (۲-۵-۴-۳)
۸۷	روش آماری کای دو
۹۰	گسسته‌سازی مبتنی بر آنتروپی
۹۲	خلاصه فصل
۹۶	مراجع و منابع جهت مطالعه‌ی بیشتر

فصل سوم: انبار داده‌ها و تکنولوژی OLAP

۱۰۰	(۳-۱) سیستم‌های تصمیم‌ساز
۱۰۱	(۳-۲) تحلیل داده‌ها و OLAP
۱۰۲	OLAP (۳-۲-۱)
۱۰۹	OLAP (۳-۲-۲) پیاده‌سازی
۱۱۱	(۳-۳) انبارسازی داده‌ها
۱۱۴	(۳-۳-۱) شیماهای انبار داده‌ها
۱۱۷	خلاصه فصل
۱۱۷	مراجع و منابع جهت مطالعه‌ی بیشتر

فصل چهارم: الگوهای مکرر و قوانین انجمانی

۱	۱۲۰ ۴) تحلیل سبد خرید
۲	۱۲۲ ۴) قوانین انجمانی
۳	۱۲۵ ۴) تولید الگوهای مکرر
۴-۱	۱۲۶ ۴-۳-۱) الگوریتم <i>Apriori</i>
۴-۲	۱۳۲ بهبود کارایی الگوریتم <i>Apriori</i>
۴-۳	۱۳۴ <i>FP-Growth</i> (۴-۳-۲) الگوریتم
۴	۱۴۰ ۴) تولید قوانین انجمانی
۵	۱۴۱ ۴) پایگاهداده تراکنشی و قالب‌های متفاوت داده ها
۶	۱۴۴ ۴) مجموعه اقلام ماسکیمال و بسته
	۱۴۷ خلاصه فصل
	۱۴۸ مراجع و منابع جهت مطالعه بیشتر

فصل پنجم: مباحث پیشرفته در قوانین انجمانی

۱	۱۵۲ ۵) ارزیابی قوانین انجمانی
۲	۱۵۸ ۵) نکاتی پیرامون معیار پشتیبان
۳	۱۶۲ ۵) الگوهای غیرمکرر
۴-۱	۱۶۳ ۵-۳-۱) الگوهای منفی و الگوهایی با همبستگی منفی
۴-۲	۱۶۶ ۵-۳-۲) تکنیک‌هایی برای کاوش الگوهای غیرمکرر
	۱۶۶ تکنیک‌هایی مبتنی بر کاوش الگوهای منفی
	۱۶۸ تکنیک‌هایی مبتنی بر مقدار پشتیبان
۴	۱۶۹ ۵) انواع دیگری از قوانین انجمانی
۴-۱	۱۷۰ ۴-۴-۱) صفات گسسته و قوانین انجمانی
۴-۲	۱۷۳ ۴-۴-۲) قوانین انجمانی مقداری
۴-۳	۱۷۵ ۴-۴-۳) سلسله مراتب مفهومی و قوانین انجمانی
۵	۱۷۷ ۵) توالی‌ها

۱۸۱ ۵-۵-۱) کاوش توالی‌های مکرر
۱۸۸ خلاصه فصل
۱۸۹ مراجع و منابع جهت مطالعه‌ی بیشتر

فصل ششم: طبقه‌بندی داده‌ها

۱۹۲ ۶-۱) مفاهیم و تعاریف اولیه
۲۰۰ ۶-۲) اندازه‌گیری خطای روش تخمین
۲۰۲ ۶-۳) ارزیابی مدل
۲۰۲ ۶-۳-۱) تکنیک <i>Holdout</i>
۲۰۳ ۶-۳-۲) تکنیک <i>Random Subsampling</i>
۲۰۴ ۶-۳-۳) تکنیک <i>k-fold cross-validation</i>
۲۰۴ ۶-۳-۴) تکنیک <i>Bootstrap</i>
۲۰۵ ۶-۴) روش‌های مقایسه‌ی مدل‌ها
۲۰۶ ۶-۴-۱) تخمین فاصله‌ی اطمینان
۲۱۰ ۶-۴-۲) مقایسه‌ی عملکرد دو مدل
۲۱۱ ۶-۴-۳) روشی دیگر برای تخمین فاصله‌ی اطمینان
۲۱۲ ۶-۴-۴) منحنی <i>ROC</i>
۲۱۳ ۶-۵) استفاده از چند مدل
۲۱۳ ۶-۵-۱) <i>Bagging</i>
۲۱۴ ۶-۵-۲) <i>Boosting</i>
۲۱۵ ۶-۶) معیارهای دیگر جهت ارزشیابی روش‌ها
۲۱۶ خلاصه فصل
۲۱۷ مراجع و منابع جهت مطالعه‌ی بیشتر

فصل هفتم: روش‌های طبقه‌بندی و تخمین

۲۲۰	(۷-۱) درخت‌های تصمیم
۲۲۴	۷-۱-۱) معیارهای انتخاب صفت خاصه
۲۲۴	<i>Information Gain</i> معیار
۲۳۰	<i>Gini Index</i> معیار
۲۳۲	<i>Gain Ratio</i> معیار
۲۳۲	<i>Likelihood Ratio</i> معیار
۲۳۳	<i>DKM</i> معیار
۲۳۳	۷-۱-۲) چند موضوع دیگر در مورد درختان تصمیم
۲۳۵	۷-۱-۳) چند الگوریتم درخت تصمیم
۲۳۵	<i>ID3</i> الگوریتم
۲۳۵	<i>C4.5</i> الگوریتم
۲۳۶	<i>CART</i> الگوریتم
۲۳۶	<i>CHAID</i> الگوریتم
۲۳۶	۷-۲) طبقه‌بندی با کمک قانون بیز
۲۴۰	۷-۳) روش‌های طبقه‌بندی مبتنی بر یافتن شروط
۲۴۳	۷-۳-۱) الگوریتم‌های یادگیری قوانین
۲۴۷	۷-۳-۲) معیارهای سنجش قوانین
۲۴۸	۷-۳-۳) بهینه نمودن قوانین
۲۴۹	۷-۴) الگوریتم‌های <i>SVM</i>
۲۵۰	۷-۴-۱) تفکیک‌پذیری خطی
۲۵۴	۷-۴-۲) تفکیک‌پذیر غیرخطی
۲۵۶	۷-۵) طبقه‌بندی براساس تشابه نزدیک
۲۵۷	۷-۵-۱) <i>k</i> نزدیکترین همسایه (<i>knn</i>)
۲۵۹	۷-۶) رگرسیون
۲۶۰	۷-۶-۱) رگرسیون خطی

۷-۶-۲) رگرسیون غیرخطی و دیگر رگرسیون‌ها ۲۶۳
۷-۷) روش‌های دیگر برای طبقه‌بندی ۲۶۴
۷-۷-۱) شبکه‌های عصبی ۲۶۴
۷-۷-۲) الگوریتم‌های ژنتیک ۲۶۶
۷-۷-۳) مجموعه‌های فازی ۲۶۸
خلاصه فصل ۲۷۰
مراجع و منابع جهت مطالعه‌ی بیشتر ۲۷۲

فصل هشتم: خوشه‌بندی

۸-۱) اهمیت و انگیزه‌ی خوشه‌بندی ۲۷۶
۸-۲) الگوریتم‌های خوشه‌بندی ۲۸۰
۸-۳) معیارهای تشابه و انواع داده‌ها ۲۸۳
۸-۳-۱) معیارهای تشابه در داده‌های پیوسته ۲۸۴
۸-۳-۲) معیارهای تشابه در داده‌های دودویی(باینری) ۲۹۰
۸-۳-۳) معیارهای تشابه در داده‌های کیفی ۲۹۵
۸-۴) تکنیک‌های خوشه‌بندی مبتنی بر افزای داده‌ها ۲۹۷
۸-۴-۱) <i>k-Means</i> ۲۹۷
۸-۴-۲) <i>k-Medoids</i> ۳۰۱
۸-۴-۳) الگوریتم‌های دیگر مبتنی بر افزای داده‌ها ۳۰۳
۸-۵) تکنیک‌های خوشه‌بندی سلسله‌مراتبی ۳۰۵
۸-۵-۱) معیارهای تشابه میان خوشه‌ها ۳۰۷
انتخاب دو نمونه با بیشترین تشابه ۳۰۷
انتخاب دو نمونه با کمترین تشابه ۳۱۰
میانگین تشابه (عدم تشابه) میان زوج نمونه‌ها ۳۱۲
روش‌های مبتنی بر نمایندگان خوشه‌ها ۳۱۴
۸-۵-۲) <i>BIRCH</i> ۳۱۵

۳۱۸	<i>CURE</i> (الگوریتم ۸-۵-۳)
۳۱۹	<i>ROCK</i> (الگوریتم ۸-۵-۴)
۳۲۰	<i>Chameleon</i> (الگوریتم ۸-۵-۵)
۳۲۴	<i>DIANA</i> (الگوریتم ۸-۵-۶)
۳۲۶	خلاصه فصل
۳۲۷	مراجع و منابع جهت مطالعه‌ی بیشتر

فصل نهم: مباحث پیشرفته در خوشه‌بندی

۳۳۰	(۹-۱) تکنیک‌های مبتنی بر تراکم
۳۳۰	<i>DBSCAN</i> (الگوریتم ۹-۱-۱)
۳۳۳	<i>OPTICS</i> (الگوریتم ۹-۱-۲)
۳۳۹	<i>DENCLUE</i> (الگوریتم ۹-۱-۳)
۳۴۲	(۹-۲) تکنیک‌های مبتنی بر گردید
۳۴۳	<i>STING</i> (الگوریتم ۹-۲-۱)
۳۴۵	<i>CLIQUE</i> (الگوریتم ۹-۲-۲)
۳۴۸	(۹-۳) تکنیک‌های خوشه‌بندی مبتنی بر مدل
۳۴۹	<i>EM</i> (الگوریتم ۹-۳-۱)
۳۵۰	(۹-۳-۲) خوشه‌بندی مفهومی
۳۵۶	(۹-۴) یافتن الگوهای مکرر و خوشه‌بندی
۳۵۸	(۹-۵) استفاده از محدودیت‌ها در خوشه‌بندی
۳۶۱	(۹-۶) ارزشیابی روش‌های خوشه‌بندی
۳۶۲	(۹-۶-۱) بررسی ساختار داده‌ها
۳۶۳	(۹-۶-۲) تعداد خوشه‌ها
۳۶۵	(۹-۶-۳) سنجش کیفیت خوشه‌بندی
۳۶۹	خلاصه فصل
۳۷۱	مراجع و منابع جهت مطالعه‌ی بیشتر

فصل دهم: مباحثی چند در داده کاوی

۱۰-۱) چند مثال کاربردی در داده کاوی ۳۷۶
۱۰-۱-۱) داده کاوی و بانکداری ۳۷۶
۱۰-۱-۲) داده کاوی و فروشگاهها ۳۷۷
۱۰-۱-۳) داده کاوی و مخابرات ۳۷۹
۱۰-۱-۴) داده کاوی و بیوانفورماتیک ۳۸۰
۱۰-۲) نکاتی در مورد محصولات تجاری داده کاوی ۳۸۱
۱۰-۲-۱) موضوعاتی چند جهت چگونگی انتخاب یک سیستم ۳۸۱
۱۰-۲-۲) چند بسته نرم افزاری داده کاوی ۳۸۳
۱۰-۳) شکل های دیگری از داده کاوی ۳۸۸
۱۰-۳-۱) متن کاوی ۳۸۸
۱۰-۳-۲) وب کاوی ۳۹۰
۱۰-۳-۳) گراف کاوی ۳۹۳
۱۰-۳-۴) کاوش در داده های چندرسانه ای ۳۹۴
۱۰-۳-۵) داده کاوی مکان محور ۳۹۵
خلاصه فصل ۳۹۷
مراجع و منابع جهت مطالعه بیشتر ۳۹۸

ضمیمه الف: منابع و مراجع

فهرست شکل‌ها

شکل ۱-۱: مراحل موجود در فرایند استخراج دانش	۲۷
شکل ۱-۲: داده کاوی و تجمعی از زمینه‌های مختلف	۳۳
شکل ۲-۱: نمونه‌ای از همبستگی کامل مثبت (شکل چپ) و همبستگی کامل منفی (شکل راست)	۵۸
شکل ۲-۲: نمونه‌ای از منحنی توزیع نرمال	۵۹
شکل ۲-۳: داده‌های لازم جهت محاسبه ^۲ برای دو کلاس و دو بازه	۸۸
شکل ۲-۴: مقادیر بدست آمده جهت بررسی ادغام دو بازه	۸۹
شکل ۲-۵: مقادیر بدست آمده جهت بررسی ادغام دو بازه	۹۰
شکل ۳-۱: جدول متقاطع فروش محصولات با ترکیب‌های متفاوت از مقادیر نام محصول و مارک	۱۰۳
شکل ۳-۲: یک سه‌بعدی برای داده‌های جدول ۳-۱	۱۰۶
شکل ۳-۳: سلسله مرتب زمان (شکل راست) و سلسله مرتب مکان (شکل چپ)	۱۰۸
شکل ۳-۴: نمایش یک جدول متقاطع همراه با گروه‌بندی صفات خاصه	۱۰۹

شکل ۵-۳: نمونه‌ای از یک معماری انبار داده‌ها	۱۱۳
شکل ۶-۳: نمونه‌ای از شیمی ستاره‌ای	۱۱۵
شکل ۷-۳: نمونه‌ای از یک شیمی برفگونه	۱۱۶
شکل ۱-۴: فضای جستجو برای یک مجموعه‌ای با ۵ قلم داده	۱۲۷
شکل ۲-۴: کلیه‌ی مجموعه اقلام ۱ تابی و شناسایی الگوهای مکرر	۱۲۹
شکل ۳-۴: برخی از مجموعه اقلام ۲ تابی کاندید و شناسایی الگوهای مکرر	۱۲۹
شکل ۴-۴: برخی از مجموعه اقلام ۳ تابی کاندید و شناسایی الگوهای مکرر	۱۳۱
شکل ۵-۴: یک نمونه درخت <i>FP-tree</i>	۱۳۵
شکل ۶-۴: مراحل ساخت و تکمیل درخت <i>FP-tree</i>	۱۳۶
شکل ۷-۴: مسیرهای منتهی به هر یک از اقلام داده‌ها	۱۳۸
شکل ۸-۴: یک <i>FP-tree</i> شرطی (سمت راست) برای درختی با مسیرهای منتهی به ۱۳ (سمت چپ).....	۱۳۹
شکل ۹-۴: تبدیل یک جدول در مدل رابطه‌ای (سمت چپ) به یک پایگاه داده‌ی تراکنشی (سمت راست)	۱۴۲
شکل ۱۰-۴: نمایش متفاوتی از یک پایگاه داده‌ی تراکنشی	۱۴۲
شکل ۱۱-۴: نمونه‌ای از نمایش افقی (سمت چپ) و عمودی داده‌ها (سمت راست) ..	۱۴۳
شکل ۱۲-۴: رابطه‌ی میان الگوهای مکرر، الگوهای بسته و الگوهای مکرر ماکسیمال	۱۴۷
شکل ۱-۵: یک نمونه از جدول وابستگی	۱۵۳
شکل ۲-۵: جدول وابستگی مربوط به مصرف چای و قهوه	۱۵۳
شکل ۳-۵: دو جدول وابستگی جهت بررسی فراوانی زوج کلمات $\{P, Q\}$ و $\{R, S\}$..	۱۵۵
شکل ۴-۵: نمونه‌ای از یک جدول وابستگی	۱۵۶
شکل ۵-۵: روابط میان الگوهای غیرمکرر، الگوهای منفی و الگوهای با همیستگی منفی	۱۶۵
شکل ۶-۵: محتویات یک جدول وابستگی با توجه به مقدار پشتیبان	۱۶۵
شکل ۷-۵: تبدیل یک پایگاه داده‌ی تراکنشی (سمت چپ) به جدولی حاوی اقلام منفی (سمت راست)	۱۶۶

شکل ۸-۵: نمونه‌ای از یک سلسله‌مراتب مفهومی ۱۶۸
شکل ۹-۵: نمونه‌ای از یک سلسله‌مراتب برای دسته‌بندی اقلام یک فروشگاه ۱۷۵
شکل ۱۰-۵: جدول اقلام خریداری شده توسط مشتریان (سمت چپ) همراه با شکل تبدیل یافته (سمت راست) ۱۷۹
شکل ۱۱-۵: تولید توالی‌هایی با طول ۴ با پیوند الگوهایی با طول ۳ ۱۸۲
شکل ۱۲-۵: تولید کاندیداها در <i>GSP</i> ۱۸۳
شکل ۱۳-۵: عملکرد روش <i>PrefixSpan</i> ۱۸۵
شکل ۱۴-۵: نمونه‌ای از یک ساختمان داده برای نگهداری پایگاه داده‌ی شکسته شده ۱۸۶
شکل ۱۵-۵: تبدیل مجموعه داده‌ی افقی (سمت چپ) به عمودی در <i>SPADE</i> ۱۸۶
شکل ۱-۶: عملکرد روش‌های طبقه‌بندی و تخمین ۱۹۲
شکل ۲-۶: مراحل تولید یک مدل توسط الگوریتم یادگیری ۱۹۵
شکل ۳-۶: نمونه‌ای از یک ماتریس برای مجموعه داده‌ایی با دو کلاس ۱۹۶
شکل ۴-۶: شکل دیگری از نمایش ماتریس ارزشیابی مدل ۱۹۸
شکل ۵-۶: ماتریسی برای ۱۰۰ نمونه مثبت و ۱۰۰ نمونه منفی ۱۹۹
شکل ۶-۶: منحنی توزیع نرمال استاندارد ۲۰۸
شکل ۷-۶: نمونه‌ای از یک منحنی <i>ROC</i> ۲۱۳
شکل ۱-۷: درخت تصمیم برای داده‌های جدول ۷-۱ ۲۲۰
شکل ۲-۷: درخت حاصل از مرحله‌ی اول با انتخاب صفت‌خاصه‌ی درآمد ۲۲۷
شکل ۳-۷: درخت نهایی برای داده‌های جدول ۷-۲ ۲۲۷
شکل ۴-۷: شروط منتج از درخت رسم شده در شکل ۷-۳ ۲۲۸
شکل ۵-۷: محاسبه‌ی آتروپی برای نقاط انفصال گوناگون صفت‌خاصه‌ی سن ۲۲۹
شکل ۶-۷: دقت محاسبه شده برای حالت‌های متفاوتی از شروط ۲۴۵
شکل ۷-۷: دقت محاسبه شده برای مرحله‌ی دوم از الگوریتم ۲۴۶
شکل ۸-۷: نمایش مجموعه‌ای از داده‌های دو کلاسی در فضای دو بُعدی ۲۵۱
شکل ۹-۷: مقایسه‌ی حاشیه‌های دو ابرصفحه برای داده‌های یکسان ۲۵۲
شکل ۱۰-۷: نمونه‌ای از یک شبکه‌ی عصبی با ۴ ورودی و ۱ خروجی و ۱ لایه‌ی پنهان ۲۶۵

شکل ۸-۱: نمونه‌ای از یک ماتریس تشابه ۲۸۴
شکل ۸-۲: نمونه‌ای از یک ماتریس شمارش برای نمونه‌های O_3 و O_4 ۲۹۲
شکل ۸-۳: ماتریس تشابه برای داده‌های جدول ۸-۴ ۲۹۳
شکل ۸-۴: ماتریس تشابه برای داده‌های موجود در جدول ۸-۲ ۲۹۸
شکل ۸-۵: نمونه‌ای از یک دندروگرام جهت نمایش خوشبندی سلسله‌مراتبی ۳۰۶
شکل ۸-۶: ماتریس تشابه برای داده‌های جدول ۸-۶ ۳۰۸
شکل ۸-۷: ماتریس تشابه پس از اجرای مرحله‌ی اول ۳۰۹
شکل ۸-۸: ماتریس تشابه پس از اجرای مرحله‌ی دوم ۳۱۰
شکل ۸-۹: ماتریس تشابه پس از اجرای مرحله‌ی سوم ۳۱۰
شکل ۸-۱۰: نمودار دندروگرام برای خوشبندی داده‌های جدول ۸-۶ ۳۱۱
شکل ۸-۱۱: ماتریس تشابه پس از اجرای مرحله‌ی اول ۳۱۲
شکل ۸-۱۲: ماتریس تشابه پس از اجرای مرحله‌ی نهایی ۳۱۲
شکل ۸-۱۳: ماتریس تشابه برای ۴ نمونه ۳۱۳
شکل ۸-۱۴: مراحل اجرای الگوریتم <i>Chameleon</i> ۳۲۱
شکل ۹-۱: نمایش شعاع همسایگی برای نقطه‌ی A ۳۳۱
شکل ۹-۲: نمایش نمونه‌های مرزی با شعاع همسایگی معین ۳۳۲
شکل ۹-۳: نمایش تصویری برای مفاهیم <i>RchDis</i> و <i>CorDis</i> ۳۳۴
شکل ۹-۴: چندین مرحله از اجرای الگوریتم <i>OPTICS</i> ۳۳۸
شکل ۹-۵: مثالی از اجرای الگوریتم <i>DENCLUE</i> برای داده‌های یک بُعدی ۳۴۰
شکل ۹-۶: یک نمونه از شبکه‌ی گردید با سه سطح ۳۴۳
شکل ۹-۷: ساختار درخت پس از درج اولین نمونه ۳۵۲
شکل ۹-۸: ساختار درخت پس از درج دومین نمونه ۳۵۳
شکل ۹-۹: درج نمونه‌ی سوم همراه با خوشبندی C_1 ۳۵۴
شکل ۹-۱۰: درخت حاصل پس از درج نمونه‌ی سوم ۳۵۵
شکل ۹-۱۱: نمودار تغییرات <i>SSE</i> بر اساس تعداد خوشبندیها ۳۶۴
شکل ۱۰-۱: تصویری از برنامه‌ی کاربردی <i>WEKA</i> ۳۸۴

فهرست شکل‌ها

۱۹

-
- شکل ۱۰-۲: تصویری از نرم‌افزار *Clementine* ۳۸۶
- شکل ۱۰-۳: رابط کاربر نرم‌افزار *RapidMiner* ۳۸۷

فهرست جدول‌ها

جدول ۱-۱: داده‌های آزمایشی با دو صفت خاصه و یک برچسب کلاس ۷۸
جدول ۲-۱: داده‌های آزمایشی با دو صفت خاصه و یک برچسب کلاس ۸۸
جدول ۳-۱: بخشی از پایگاه داده‌ی یک فروشگاه لوازم الکترونیکی ۱۰۲
جدول ۳-۲: شکل دیگری از جدول متقاطع نشان داده شده در شکل ۳-۱ ۱۰۵
جدول ۱-۴: نمونه‌ای از یک پایگاه داده‌ی تراکنشی فروشگاه ۱۲۰
جدول ۲-۴: نمونه‌ای از یک پایگاه داده با ۴ تراکنش ۱۲۳
جدول ۳-۴: یک پایگاه داده‌ی تراکنشی با ۵ تراکنش و ۱۱ قلم داده ۱۲۸
جدول ۴-۴: مجموعه داده‌هایی شامل ۹ تراکنش ۱۳۵
جدول ۵-۴: الگوهای مکرر همراه با پسوند آنها ۱۳۹
جدول ۶-۴: پایگاه داده‌ی تراکنشی با ۲ تراکنش ۱۴۶
جدول ۱-۵: معیارهای ارزیابی مجموعه اقلام و قوانین انجمنی ۱۵۷
جدول ۲-۵: دسته‌بندی اقلام با توجه به مقدار پشتیبان ۱۶۰
جدول ۳-۵: مثالی از یک مجموعه داده با صفات خاصه‌ی گسسته ۱۷۰

جدول ۴-۵: تبدیل جدول ۳-۵ به داده‌هایی از نوع دودویی ۱۷۱
جدول ۵-۵: نمونه‌ای از یک پایگاه داده شامل داده‌های گستته و پیوسته ۱۷۳
جدول ۱-۶: نمونه داده‌های پزشکی ۱۹۳
جدول ۲-۶: چند معیار جهت ارزشیابی مدل ۱۹۷
جدول ۳-۶: مقادیر $Z_{\alpha/2}$ برای سطوح مختلف اطمینان ۲۰۹
جدول ۴-۶: فاصله‌ی اطمینان برای مقدار متفاوتی از داده‌ها ۲۰۹
جدول ۱-۷: مشخصات بیماران همراه با داروی تجویزی برای هر یک ۲۲۱
جدول ۲-۷: نمونه‌ای از یک داده‌ی آزمایشی ۲۲۵
جدول ۳-۷: اطلاعاتی جهت تصمیم‌گیری برای اعطای وام ۲۳۰
جدول ۴-۷: یک نمونه داده‌های آموزشی ۲۳۸
جدول ۵-۷: مجموعه داده‌ای با ۱۰ نمونه‌ی آموزشی ۲۴۴
جدول ۶-۷: مشخصات دکارتی ۶ نقطه همراه با کلاس هر یک ۲۵۹
جدول ۷-۷: سودهای حاصل از سرمایه‌گذاری یک شرکت در ۱۰ سال ۲۶۲
جدول ۱-۸: نمایش سه نمونه با چهار صفت خاصه ۲۸۵
جدول ۲-۸: مشخصات دکارتی ۶ نقطه ۲۸۹
جدول ۳-۸: مشخصات ۵ نمونه همراه با ۶ صفت خاصه برای هر یک ۲۹۱
جدول ۴-۸: نمایش جدول ۳-۸ با کمک داده‌های دودویی ۲۹۲
جدول ۵-۸: چگونگی محاسبه‌ی چند نمونه از معیارهای تشابه برای داده‌های دودویی ۲۹۴
جدول ۶-۸: داده‌های آزمایشی با تعداد ۵ نمونه و ۲ صفت خاصه ۳۰۸
جدول ۷-۸: معانی عبارت‌های به کار برده شده در محاسبه‌ی توابع RC و RI ۳۲۳
جدول ۱-۹: خوش‌های تولید شده با انتخاب متنوعی از صفات خاصه برای یک مجموعه داده ۳۴۶
جدول ۲-۹: نمونه‌ای از داده‌های آزمایشی با تعداد ۳ صفت خاصه ۳۵۱
جدول ۳-۹: نتیجه‌ی نهایی خوش‌بندی ۱۰۰۰ سند متنی ۳۶۶

در دنیا تنها یک فضیلت وجود دارد، آن دانایی است
و تنها یک گناه و آن جهل است.

پیش‌گفتار

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. داده‌کاوی کوششی برای بدست آوردن اطلاعات مفید از میان این داده‌هاست و رشد بی رویه‌ی داده‌ها در سطح جهان اهمیت داده‌کاوی را دو چندان کرده است.

کتاب حاضر حاوی مفاهیم و تکنیک‌های اساسی داده‌کاوی است و در ده فصل تنظیم شده است که در انتهای هر فصل پس از بیان خلاصه به توضیح منابع و مراجع برای مطالعه‌ی بیشتر نیز پرداخته شده است. فصل اول به مقدمه‌ای در مورد مفاهیم داده‌کاوی می‌پردازد. در واقع این فصل توضیحی مفصل در مورد مطالب موجود در کتاب است. تکنیک‌های آماده‌سازی داده‌ها موضوع فصل بعدی را تشکیل می‌دهد. در فصل سوم توضیحی مختصر در مورد انبار داده‌ها و *OLAP* داریم. دو فصل چهارم و پنجم به قوانین انجمنی و موضوعات پیشرفتی پیرامون آن تخصیص داده شده است. روش‌های طبقه‌بندی داده‌ها و همچنین چگونگی ارزیابی آنها در فصل‌های ششم و هفتم بررسی می‌شوند. الگوریتم‌های مختلف خوشه‌بندی نیز در فصل‌های هشتم و نهم بحث می‌شوند. همانطور که ملاحظه می‌فرمایید، سه موضوع قوانین انجمنی، طبقه‌بندی و خوشه‌بندی مفاهیم محوری این کتاب محسوب می‌شوند و برای هر یک دو فصل تخصیص داده شده است. در فصل پایانی به برخی از موضوعات مرتبط با داده‌کاوی به اختصار اشاره‌ای خواهیم داشت. در این فصل ابتدا برخی از کاربردهای عملی داده‌کاوی بیان شده است و سپس از چند محصول تجاری داده‌کاوی نام برده‌ایم. در انتهای فصل هم به گونه‌های دیگر داده‌کاوی مانند متن‌کاوی، وب‌کاوی و ... پرداخته‌ایم.

اینکه بعضی بر این باورند که در حوزه‌ی علم کامپیوتر نباید کتابی به فارسی تالیف و یا ترجمه شود و به همین دلیل دست روی دست گذاشته‌اند، منصفانه نیست. اما در این نکته که به دلایلی تالیف و بخصوص ترجمه‌ی کتاب در این حیطه با مشکلاتی همراه است، با آنها هم نظر هستیم. دوستانی که چنین کرده‌اند با این مصائب نیز به خوبی آشنا هستند. تبدیل و ترجمه‌ی واژگان بیگانه به صورت مستقیم به زبان فارسی یا امکان‌پذیر نیست و یا خواننده را به وادی دیگری سوق می‌دهد و برای او در بسیاری از موارد ناآشنا است. به همین دلیل در نگارش کتاب حاضر تمام سعی و تلاش خود را به کار برده‌ایم، تا واژه‌ها به جای ترجمه‌ی مستقیم تفسیر شوند. در برخی از موارد نیز عین واژه در متن کتاب استفاده شده است، چرا که حقیر کلمات و واژگان مناسبی را برای آن نیافتم. از پیچیده‌نویسی و مبههم‌گویی پرهیز شده و بر ساده‌نویسی تاکید شده است. لذا در بسیاری از موارد چنانچه سختگیرانه قضاوت کنید، شاید بتوان ادعا نمود که دستور زبان فارسی به درستی رعایت نشده است، ولی در عوض امیدواریم خواننده با یک متن گیج‌کننده مواجه نشود.

اینکه این کتاب ارزنده است یا خیر، نه اینکه موضوع مهمی نباشد، اما در درجه‌ی اول با نوشتن آن خود را افغان نموده‌ام. زیرا خواندن و نوشتن از بزرگترین شادی‌های من محسوب می‌شوند و با نوشتن بر این باورم که به نحوی در دیگران پخش می‌شوم. با وجود همه‌ی سعی و تلاشی که در تمام مراحل آماده‌سازی این کتاب انجام گرفته است، یقین دارم که عاری از اشتباه نیست، چرا که تنها مکتوب بی‌نقص همان معجزه‌ی جاوید قرآن کریم است. افسوس که نمی‌توان بازگشت و از نو ساخت، اما دست کم به آنها که در آغاز راهند می‌توان یادگاری کوچکی داد، شاید به کارشان بیاید. در آخر ضمن سپاسگزاری از همه‌ی کسانی که مرا یاری داده‌اند و با پذیرش مسئولیت هرگونه کاستی احتمالی، امیدوارم که این اندک مفید افتند.

مهدی اسماعیلی

تیرماه ۱۳۹۱

فصل اول

مقدمه

تکنولوژی مدیریت پایگاه داده‌های پیشرفته انواع مختلفی از داده‌ها را می‌تواند در خود جای دهد، در نتیجه تکنیک‌های آماری و ابزار مدیریت سنتی برای آنالیز این داده‌ها کافی نیست و استخراج دانش^۱ از این مقدار حجمی یک چالش بزرگ تلقی می‌شود. داده‌کاوی^۲ کوششی برای بدست آوردن اطلاعات مفید از میان این داده‌هاست و رشد بی‌رویه‌ی داده‌ها در سطح جهان اهمیت داده‌کاوی را دو چندان کرده است.

در این فصل ابتدا به معرفی مفاهیم اصلی داده‌کاوی می‌پردازیم و پس از آن انواع داده‌ها که می‌توان عملیات داده‌کاوی را بر روی آنها انجام داد، بیان می‌شوند. انتهای فصل به چالش‌های موجود در این حوزه می‌پردازد. در واقع این فصل توضیحی مختصر در مورد فصل‌های آتی را در خود دارد.

¹ Knowledge Discovery

² Data Mining

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^۱ 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**^۲ بیشتر در علم آمار استفاده می‌شود، در حالی که در علوم کامپیوتر و یادگیری ماشین بیشتر از واژه‌های صفت‌خاصه^۳ و یا ویژگی^۴ استفاده می‌گردد.

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

- جمع آوری داده‌ها^۵: در این مرحله پس از پایانش داده‌ها، چندین منبع داده‌ای در یک انبار داده‌ی ^۶ 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 انواعی از الگوها، با تعریف معیارهای متنوع الگوهای محدودی برای تفسیر و تحلیل انتخاب می‌شوند.

¹ Observations

² Variable

³ Attribute

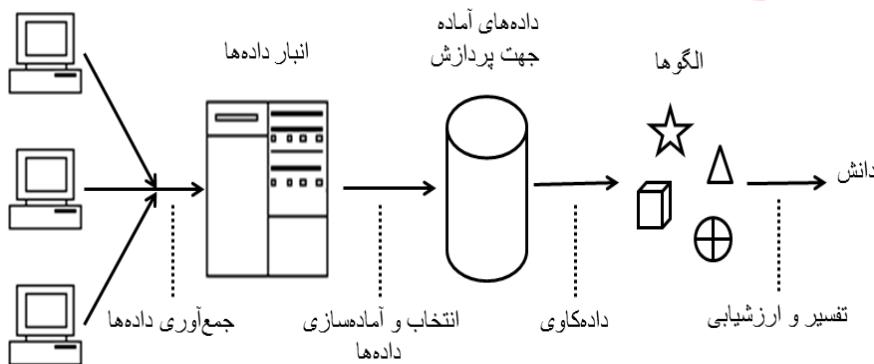
⁴ Feature

⁵ Integration

⁶ Data Warehouse

- ارائه‌ی دانش: در این مرحله با کمک ابزار بصری‌سازی^۱ 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 نظری 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^۲ کار می‌کنند که این مسئله نیز در جمع‌آوری داده‌ها محدودیت محسوب می‌شود.

¹ Visualization² Homogeneous

هر ساختار کلیدی در انبار داده‌ها به صورت تلویحی یا مستقیم شامل عنصری از جنس زمان است. بر اساس این توضیحات یک انبار داده‌ها می‌تواند به عنوان مخزن داده‌های یک سازمان در نظر گرفته شود، تا با کمک آن بتوان از تصمیم‌گیری‌های راهبردی حمایت کند. وظیفه‌ی آن ذخیره‌ی یکپارچه‌ی داده‌های سازمان است و از آنجا که عمل بهنگام‌سازی انبار داده‌ها عملی زمانبُر محسوب می‌شود، اغلب داده‌های آن بهنگام نیستند. بنابراین داده‌ها اندکی قدیمی هستند که البته این مسئله برای سیستم‌های تصمیم‌ساز مشکل بزرگی تلقی نمی‌شود. توضیحات بیشتر در مورد انبار داده‌ها را در فصل سوم مطالعه فرمایید.

به منظور تسهیل و بهبود فرایند داده‌کاوی آماده‌سازی داده‌ها^۱ یکی از مراحل اساسی تلقی می‌شود. بسیاری از کارشناسان داده‌کاوی در اینکه آماده‌سازی داده‌ها یکی از بحرانی‌ترین مراحل موجود در فرایند استخراج دانش است، اتفاق نظر دارند. نامفهوم بودن داده‌ها و استفاده‌ی نادرست از ابزار داده‌کاوی، می‌تواند این فرایند را در مسیری نادرست قرار دهد. از این‌رو می‌توان گفت داده‌کاوی فقط راهنمای استفاده از ابزاری برای مشکل مطرح شده نیست، بلکه یک فرایند بحرانی اکتشافی است و به همین دلیل داده‌ها باید برای این عمل مهم، درست و سازگار تعریف شوند. با توجه به این دلایل و همانطور که قبل از این نیز اشاره کردیم، بسیاری از کارشناسان مرحله‌ی آماده‌سازی و تغییر شکل مناسب داده‌های اولیه را در داده‌کاوی یکی از بحرانی‌ترین گام‌ها می‌دانند.

کیفیت داده‌های ورودی در ساده‌ترین تحلیل تا ساخت مدل‌های پیچیده یکی از کلیدهای موفقیت انجام یک پروژه به حساب می‌آید. با توجه به این نکته که مقدار و پیچیدگی داده‌ها روز به روز رو به افزایش است، توانایی مدل برای تولید نتایج خوب و بدردبار خور بطور کامل به داده‌های خوب و درست ورودی متکی است.

در عمل تقریباً ۸۰ درصد از سعی و تلاش کل برای مهندسی داده‌ها^۲، صرف آماده‌سازی داده‌ها می‌شود و این اهمیت موضوع را کاملاً مشخص می‌کند. به طور مثال در بسیاری از

¹ Data Preparing

² Data Engineering

شاخه‌های علم کامپیوتر نظریه تشخیص الگو^۱، بازیابی اطلاعات، یادگیری ماشین^۲ و همچنین داده‌کاوی، آماده‌سازی داده‌ها بوسیله‌ی پردازش‌های قبل از اجرا، نیاز اصلی محسوب می‌شود.

بنابراین یکی از مراحل مهم و بیچیده در فرایند کشف دانش آماده‌سازی داده‌هاست. انتخاب داده‌های مناسب به منظور بدست آوردن کارایی حداکثر و پردازش حداقل از اهداف کلی مرحله‌ی آماده‌سازی داده‌ها به شمار می‌آید. بنابراین با داده‌های درست و کامل و در عین حال کم، الگوریتم‌های داده‌کاوی سریعتر و موثرتر به الگوی مورد نظر می‌رسند و نتایج حاصل از آن قابل فهم‌تر و راحت‌تر برای کاربر خواهد بود. آماده‌سازی و پیش‌پردازش داده‌ها ممکن است به ماهیت داده‌ها و نوع تحلیل بر روی آنهاست، بنابراین با شناخت داده‌ها می‌توان روش مناسبی برای آماده‌سازی داده‌ها انتخاب کرد. لازم است مشخصات اصلی داده‌ها با انواع مختلفی از تحلیل‌های آماری بدست آید.

بسترهاي داده‌ای که دارای ابعاد زیادی هستند علیرغم فرصت‌هایی که به وجود می‌آورند، چالش‌های محاسباتی زیادی را ایجاد می‌کنند. یکی از مشکلات داده‌هایی با ابعاد زیاد این است که در بیشتر مواقع تمام ویژگی‌های داده‌ها برای یافتن دانشی که در داده‌ها نهفته است مهمن و حیاتی نیستند. به همین دلیل در بسیاری از زمینه‌ها کاهش ابعاد داده یکی از مباحث قابل توجه در زمینه‌ی آماده‌سازی داده‌ها باقی مانده است.

اکثر تحقیقات در این چند سال اخیر بر روی الگوریتم‌های داده‌کاوی صورت گرفته و متأسفانه کوشش کمی برای بدست آوردن تئوری‌های قوی برای آماده‌سازی داده‌ها به عمل آمده است. در حالیکه دستاوردهای زیادی که در مرحله‌ی داده‌کاوی بدست می‌آیند، جملگی مدیون مرحله‌ی پیش‌پردازش و آماده‌سازی داده‌ها می‌باشند.

روش‌های آماده‌سازی داده‌ها به جز در این مرحله می‌توانند در کلیه‌ی مراحل فرایند کشف دانش استفاده شوند، نه تنها در الگوریتم‌های داده‌کاوی بلکه پس از پردازش نهایی برای خلاصه‌سازی و ارزیابی نتایج، کاربرد فراوانی دارند. برای مثال قابل فهم بودن الگوهای نهایی یکی دیگر از وظایف آماده‌سازی داده‌ها محسوب می‌شود. استفاده از ابزاری برای

¹ Pattern Recognition

² Machine Learning

نمایش گرافیکی، ساختارهای شرطی، زبان‌های طبیعی و تکنیک‌های دیگر می‌توانند در این مرحله به ما کمک کنند.

تکنیک‌های مختلفی از جمله پالایش داده‌ها، جمع‌آوری و تغییر شکل داده‌ها و کاهش داده‌ها در این مرحله استفاده می‌شوند که برای هر یک الگوریتم‌های متعددی وجود دارند.

معیارهای بسیاری برای انتخاب یک الگوریتم مناسب وجود دارند، اما به جرأت می‌توان گفت که دانستن ماهیت داده‌ها در انتخاب، اهمیت زیادی دارد. آماده‌سازی داده‌ها و همچنین داده‌کاوی کاملاً به برنامه‌ی کاربردی و اهداف آن وابسته هستند. از این رو شناخت داده‌ها و خصوصیات آنها کمک زیادی به بهبود عمل آماده‌سازی می‌کند.

مرحله‌ی بعدی در فرایند استخراج دانش اعمال تکنیک‌های داده‌کاوی یافتن الگوهای مفید نهان در داده‌ها است. در این مرحله با به خدمت گرفتن الگوریتم‌های مناسب داده‌کاوی برآوردها تا دانش جدیدی از میان داده‌ها کسب کنیم. راهکارهای متنوعی جهت انجام این کار وجود دارند که ما در این کتاب روش‌های کاوش قوانین انجمنی، تکنیک‌های طبقه‌بندی و خوشه‌بندی را مورد بررسی قرار خواهیم داد.

پس از اجرای الگوریتم‌های داده‌کاوی نوبت به ارزشیابی و تفسیر نتایج خروجی این الگوریتم‌ها می‌رسد. به طور حتم می‌دانید که کلیه‌ی الگوهای بدست آمده از مرحله‌ی قبلی مفید نیستند و یا حداقل می‌توان گفت در بسیاری از موارد مقدار و تعداد این الگوها آنقدر زیاد است که امکان پذیر نیست تا کاربر همه‌ی آنها را مشاهده کند. پس از انتخاب برخی از الگوهای مورد توجه کاربر با کمک ابزارهای بصری‌سازی مناسب می‌توان این دانش را به نحو مطلوبی تحت اختیار کاربر و یا مدیران قرار داد.

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^۱ (مانند متون)، نیمه‌ساخت‌یافته^۲ (مانند اسناد) و ساخت‌یافته^۳ (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 بی نیز جهت تحلیل داده‌ها استفاده نمود. یک انبار داده‌ها مخزنی از اطلاعات جمع‌آوری شده از چندین منبع داده‌ای تحت یک شیمای^۴ 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

¹ Unstructured

² Semi structured

³ Structured

⁴ Schema

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

- پایگاه داده‌ی تراکنشی^۱ 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^۲ می‌نامیم. تکنیک‌های داده‌کاوی می‌توانند رفتار محتویات تراکنش‌ها را در رابطه با زمان بررسی کنند.
- امروزه وب یک مخزن داده‌ای پویا و نیز ناهمگن^۳ 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 چندرسانه‌ای نیز می‌تواند به وب‌کاوی کمک کند. یک سیستم مدیریت پایگاه داده‌ی چندرسانه‌ای مجموعه‌ی وسیعی از داده‌های چندرسانه‌ای 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 چندرسانه‌ای، ذخیره و بازیابی موثر و سریع داده‌ها از اهمیت بالایی برخوردار است. در مورد وب‌کاوی و همچنین کاوش در میان داده‌های چندرسانه‌ای در فصل آخر اشاره‌ای خواهیم داشت.

¹ Transactional Database

² Time Series Database

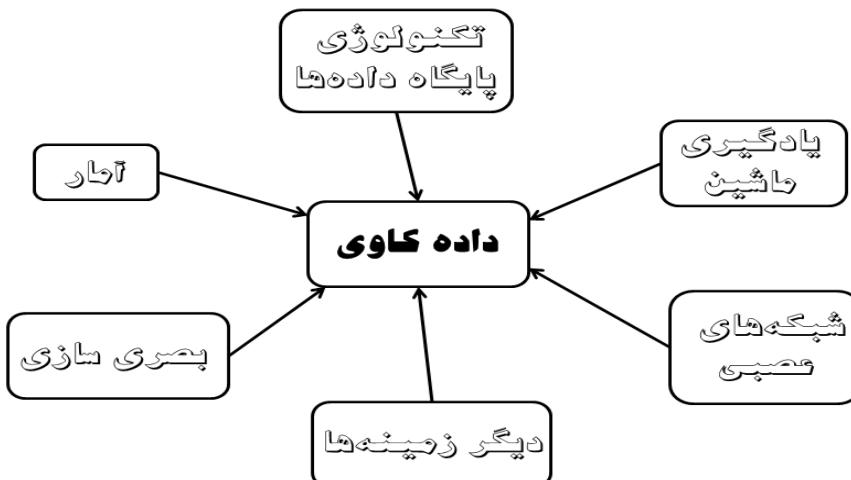
³ Heterogeneous

- یک پایگاه داده‌ی مکان‌محور^۱ 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

^۱ Spatial Database

ما در کتاب حاضر از زاویه‌ی دید پایگاه داده‌ها به آن نگاه می‌کنیم و بدین ترتیب به دنبال الگوریتم‌های مقیاس‌پذیر و کارایی از داده‌کاوی خواهیم بود. در بخش قبلی انواع داده‌هایی که می‌توانند ورودی یک الگوریتم داده‌کاوی را تشکیل دهند، بررسی شدند. اجازه دهید در این بخش به نوع الگوهای استخراج شده توسط روش‌های داده‌کاوی اشاره‌ای داشته باشیم. از یک نقطه نظر می‌توان عملیات داده‌کاوی را در دو گروه دسته‌بندی نمود، که دسته‌ی اول به توصیف خصوصیت داده‌ها در پایگاه داده‌ها می‌پردازند و عملیات دسته‌ی دوم با مدل‌سازی داده‌های در دسترس سعی می‌کنند تا یک پیش‌بینی صحیح از داده‌های آتی و آزمایشی داشته باشند. مواقیعی که کاربر هیچ‌گونه نظری در مورد نوع الگوها ندارد، می‌تواند از دو روش استفاده و نتایج را مشاهده و ارزیابی کند. به همین دلیل بسیار مهم است که یک سیستم داده‌کاوی را انتخاب کند که عملیات متنوعی را می‌توان در آن یافت (در فصل دهم پیرامون موضوع انتخاب بسته‌های نرم‌افزاری داده‌کاوی توضیحات بیشتری ارائه شده است).

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^۱، طبقه‌بندی داده‌ها^۲ و خوشبندی^۳ 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.

¹ Association Rules

² Classification

³ Clustering

۱-۳-۱) قوانین انجمانی

طی سال‌های گذشته در میان تکنیک‌های داده‌کاوی توجه خاصی به الگوریتم‌های کشف الگوهای مکرر^۱ 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^۲ 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^۳ و اطمینان^۱ اندازه‌گیری شده است. تفسیر قانون انجمانی مزبور بدین گونه است که ۴۰ درصد

¹ Frequent Patterns Mining

² Frequent Itemsets Mining

³ 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.

¹ Confidence

روش‌های ناظارت‌شده‌ای^۱ 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^۲ 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^۳ 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.

۱-۳-۳) 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 خوش شبيه بوده ولی متفاوت از نمونه‌های دیگر خوشبند است.

¹ Supervised

² Training Dataset

³ 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 کم جمعیت 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^۱، توانایی مواجهه با انواع داده‌ها، استخراج خوش‌هایی 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 خوش‌هایی مبتنی 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 خوش‌های 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.

¹ 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 چندساله‌ای، گراف‌ها و... چالش بزرگی است.

- توسعه‌ی داده‌کاوی توزیع شده^۱ و بلاذرنگ^۲. 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^۳، 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.

۱-۵) سازماندهی کتاب

بعد از معرفی مفاهیم اصلی داده‌کاوی در فصل اول، در فصل‌های باقیماندهی کتاب روش‌ها و تکنیک‌های اساسی داده‌کاوی شرح داده می‌شوند. ساختار کتاب و محتویات فصل‌ها به ترتیب زیر هستند:

- در فصل دوم تکنیک‌های آماده‌سازی داده‌ها بررسی می‌شوند. این تکنیک‌ها در گروه‌های مختلفی از جمله پالایش داده‌ها، حذف داده‌های نامربوط، کاهش ابعاد داده‌ها و تبدیل شکل داده‌ها گنجانده شده‌اند.

¹ Distributed

² Real Time

³ Visual Data Mining

- فصل سوم را تخصیص داده‌ایم به مطلب انبار داده‌ها و تکنیک‌های *OLAP*. در این فصل چگونگی ساخت یک انبار داده‌ها و همچنین عملیاتی که بر روی آن اعمال می‌شوند، توضیح داده شده است.
- قوانین انجمنی را در فصل‌های چهارم و پنجم بحث کرده‌ایم. یافتن الگوهای مکرر و الگوریتم‌های آن موضوع فصل چهارم را تشکیل داده است و در فصل پنجم تحت عنوان مفاهیم پیشرفته در قوانین انجمنی معیارهای ارزشیابی این قوانین و محدودیت‌های هر یک بیان شده‌اند.
- روش‌های طبقه‌بندی موضوع فصل‌های ششم و هفتم را تشکیل می‌دهند. ابتدا در فصل ششم مفاهیم کلی، روش‌های ارزیابی و تست تکنیک‌های طبقه‌بندی بررسی می‌شوند. پس از آن در فصل هفتم بعضی از روش‌های مرسوم جهت طبقه‌بندی بیان می‌شوند. از این میان می‌توان به درخت‌های تصمیمی، روش‌های مبتنی بر یافتن قوانین و شروط و همچنین *SVM* اشاره نمود.
- همانند قوانین انجمنی و روش‌های طبقه‌بندی دو فصل نیز به تکنیک‌های خوشه‌بندی تخصیص داده شده است. در فصل هشتم روش‌های رایج و پرکاربرد خوشه‌بندی توضیح داده شده است و در فصل نهم تکنیک‌های پیشرفته‌ی خوشه‌بندی همراه با مثال‌های متعدد قرار داده شده است. در این فصل از ارزشیابی خروجی الگوریتم‌های خوشه‌بندی نیز صحبت می‌کیم.
- فصل آخر یعنی فصل دهم کتاب در واقع اشاره‌ای به برخی از مطالب مرتبط با داده‌کاوی را در خود دارد. این فصل با چند مثال کاربردی شروع می‌شود و پس از آن با معرفی چند معیار، برخی از بسته‌های تجاری داده‌کاوی را نام می‌برد. شکل‌های دیگری از داده‌کاوی مانند متن‌کاوی، وب‌کاوی، کاوش در میان مجموعه داده‌های چندرشانه‌ای و یا کاوش میان ساختار داده‌ای مانند گراف‌ها موضوعاتی هستند که در انتهای فصل دهم به آن پرداخته‌ایم.

خلاصه فصل

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 هر روز با مشاهدات بیشتری روبرو می‌شوند. تکنولوژی مدیریت پایگاه داده‌ها انواع مختلفی از داده‌ها را در خود جای می‌دهد، در نتیجه تکنیک‌های آماری و ابزار مدیریت سنتی برای تحلیل این داده‌ها کافی نیست و استخراج داده از این مقدار حجمی یک چالش بزرگ تلقی می‌شود. داده‌کاوی کوششی برای بدست آوردن اطلاعات مفید از این داده‌هاست و رشد بی رویه‌ی داده‌ها در سطح جهان اهمیت داده‌کاوی را دو چندان کرده است.

جمع آوری داده‌ها، انتخاب و آماده‌سازی داده‌ها، داده‌کاوی، تفسیر و ارزشیابی الگوها و در نهایت ارائه‌ی دانش مراحل مختلف استخراج و کشف دانش را تشکیل می‌دهند. در هر یک از این مراحل با به خدمت گرفتن الگوریتم و یا تکنیک‌های متفاوت سعی در یافتن الگوهای جالب‌تر خواهیم داشت. الگویی جالب است که قابل فهم، مفید و بدیع و همچنین مطابق با فرضیه‌های تعریف شده توسط کاربر باشد.

به منظور تسهیل و بهبود فرایند داده‌کاوی آماده‌سازی داده‌ها یکی از مراحل اساسی تلقی می‌شود. بسیاری از کارشناسان داده‌کاوی در اینکه آماده‌سازی داده‌ها یکی از بحرانی‌ترین مراحل موجود در فرایند استخراج دانش است، اتفاق نظر دارند. نامفهوم بودن داده‌ها و استفاده‌ی نادرست از ابزار داده‌کاوی، این فرایند را در مسیری نادرست قرار می‌دهد. کیفیت داده‌های ورودی در ساده‌ترین تحلیل تا ساخت مدل‌های پیچیده یکی از کلیدهای موفقیت انجام یک پروژه به حساب می‌آید.

عملیات داده‌کاوی به یک نوع از داده‌ها محدود نمی‌شود و عموماً داده‌های مختلفی توسط این سیستم‌ها پذیرفته می‌شوند. تکنیک‌های داده‌کاوی را می‌توان بر روی داده‌های غیرساخت‌یافته، نیمه‌ساخت‌یافته و همچنین ساخت‌یافته إعمال نمود. جداول شکل رایجی از ورودی این الگوریتم‌ها تلقی می‌شوند که در آن سطرهای جدول نشان‌دهنده‌ی نمونه‌ها و یا رکوردها هستند و ستون‌ها دلالت بر ویژگی و صفات خاصه‌ی هر یک از این نمونه‌ها دارند.

تکنیک‌های متنوعی در داده‌کاوی وجود دارند که به تولید الگوهای متفاوتی می‌پردازند. روش‌های کشف قوانین انجمنی که به دنبال رابطه‌ای میان صفات می‌گردند، روش‌های طبقه‌بندی داده‌ها که به ساخت یک مدل می‌پردازند و همچنین تکنیک‌های خوشه‌بندی که نمونه‌ها را بر اساس تشابه میان آنها در گروه‌های متفاوتی قرار می‌دهند، از عمده‌ترین راهکارهایی محسوب می‌شوند که این الگوها را تولید می‌کنند.

تنوع داده‌ها و کاربردهای داده‌کاوی چالش‌های تحقیقاتی بسیاری را برای آن ایجاد نموده است. طراحی و پیاده‌سازی سیستم‌های داده‌کاوی برای کاربردهای خاص، توسعه‌ی الگوریتم‌های مقیاس‌پذیر، استانداردسازی یک زبان برای عملیات داده‌کاوی، طراحی و پیاده‌سازی روش‌های جدید برای مقابله با انواع داده‌های پیچیده مانند متن، توسعه‌ی داده‌کاوی توزیع شده و بلاذرگ و همچنین بررسی موضوعاتی مانند امنیت و مهندسی نرم‌افزار در داده‌کاوی از این چالش‌ها محسوب می‌شوند.

مراجع و منابع جهت مطالعه‌ی بیشتر

کتاب‌های بسیار زیادی را می‌توان یافت که بر روی موضوع داده‌کاوی متمرکز شده‌اند. در طول نگارش این کتاب از دو مرجع [HCP11] و [TSK05] بسیار استفاده کرده‌ایم. به همین دلیل فقط به نام آنها در این فصل اشاره خواهیم کرد. وجود مثال‌های متنوع و پوشش بسیار زیادی از مطالب پایه‌ای داده‌کاوی دلیل اصلی این انتخاب بود. ممکن است در حین طبع کتاب حاضر، نسخه‌های بعدی آنها نیز به بازار عرضه شده باشند.

اما اخیراً کتاب‌های زیادی در زمینه‌ی داده‌کاوی را می‌توانید در قفسه‌های کتابخانه‌ها پیدا کنید که صرفنظر از زاویه‌ی دید هر یک از آنها می‌توان از این بین به [WFH11] [HMS01] [Dun03] و [WI98] اشاره نمود. البته کتاب‌های دیگری نیز هستند که بر روی موضوعاتی مانند کاوش بر روی داده‌های وب و یا کاربرد داده‌کاوی در تجارت و دیگر موارد تکیه دارند، ولی در آنها می‌توان الگوریتم‌های اساسی داده‌کاوی را نیز پیدا کرد. برخی از این مراجع عبارتند از [BL99] [Cha03a] [Liu06] و [MA03].

همچنین کتاب‌هایی را می‌توان یافت که شامل مجموعه مقالات و فصل‌های گزینشی در مورد مطلب خاصی از بحث استخراج دانش و داده‌کاوی هستند، برای مثال مراجع

[YHF10] و [MH09] و [ZZ09] و [Agg06] و [CH07] و [De01] از این گونه کتب محسوب می‌شوند.

نکته‌ی حائز اهمیت وجود یک خبرنامه‌ای با نام KDNuggets است که از سال ۱۹۹۱ توسط شخصی با نام Piatetsky-Shapiro اداره می‌شود. سایت اینترنتی آن www.kdnuggets.com است و شما می‌توانید با رجوع به این سایت به مطالب علمی بسیار زیادی در مورد کشف دانش و داده‌کاوی دست پیدا کنید. در ضمن در این سایت شما می‌توانید اطلاعاتی در مورد ژورنال‌ها و کنفرانس‌های معتبر بین‌المللی در زمینه‌ی داده‌کاوی مانند SIAM و ACM-SIGKDD کسب کنید و به همین دلیل از نام بردن و توضیحاتی پیرامون این مطلب در آنجا پرهیز می‌کنیم.

در ایران نیز کنفرانس‌های متعددی در زمینه‌ی داده‌کاوی برگزار می‌شود که یکی از آنها کنفرانس بین‌المللی داده‌کاوی است که پنجمین دوره‌ی آن در سال ۱۳۹۰ برگزار شده است و برای اطلاعات بیشتر می‌توانید به سایت www.irandatamining.com رجوع کنید.

نتایج تحقیقات و پژوهش‌های انجام شده در داده‌کاوی ممکن است در کتاب‌ها، ژورنال‌ها و کنفرانس‌هایی در حوزه‌های دیگر مانند پایگاه داده‌ها، آمار، یادگیری ماشین، بازیابی اطلاعات و نیز یافت شوند و شما می‌توانید با انتخاب یک حوزه‌ی خاص به این منابع رجوع کنید. از آنجا که زاویه‌ی نگرش ما در این کتاب از منظر پایگاه داده است در انتهای بعضی از مراجع رایج پایگاه داده‌ها اشاره‌ای می‌کنیم که عبارتند از [GMUW08] و [RG03] و [EN10] و [SKS10].

فصل دوم

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^۱ 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.

¹ 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^۱ و داده‌های کیفی^۲.

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^۳ یا گسسته^۴ 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

¹ Quantitative

² Qualitative

³ Continuous

⁴ Discrete

مورد نظر است، داده‌ها در طبقات و دسته‌های مختلفی قرار می‌گیرند. گاهی این متغیرها را داده‌های طبقه‌بندی شده^۱ 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^۲ گویند. مجموعه‌ی مقادیر برای مدرک تحصیلی (دیپلم، کاردانی، 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.

¹ Categorical

² Ordinal

چنانچه دسته‌هایی که متغیر در آنها قرار می‌گیرد، دارای هیچگونه ترتیب طبیعی نباشد، هر یک از آن متغیرها را متغیر صوری یا اسمی^۱ 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 پرس‌وچهارهای پایگاه داده کاربر راحت‌تر عبارات را بیان کند و یا هر یک از کدها دارای بار معنایی مناسبی باشند.

۲-۲) ابعاد بالای داده‌ها

یکی از مشکلات فرایند داده‌کاوی از آنجا ناشی می‌شود که ما با مقدار خیلی زیاد و متنوعی از داده‌ها روبرو هستیم، که هر یک دارای رفتار متفاوتی هستند. جمع‌آوری صفات خاصه در انبارداده‌ها^۲ بطور طبیعی افزایش ابعاد داده‌ها را به دنبال دارد و این یکی از مشکلات اساسی شناخته شده در داده‌کاوی است. خصوصیات فضاهایی با ابعاد بالا عموماً برای ما غیر مشهود است، چرا که ما در طبیعت با فضاهای دو و سه بعدی مأнос هستیم. برای روشن‌تر شدن مسئله چند خصوصیت برای داده‌هایی با ابعاد بالا را بررسی می‌کنیم. به منظور تحلیل داده‌ها به مقدار داده مناسبی نیاز داریم. طبیعی است که با تنوع بیشتر در ویژگی داده‌ها که ما آنرا با ابعاد بالاتر نشان می‌دهیم، مقدار داده‌های موجود در انبار داده‌ها نیز باید بیشتر شوند. بنابراین با افزایش ابعاد بطور حتم افزایش داده‌ها را نیز به صورت تصاعدی خواهیم داشت. با یک مثال این موضوع را می‌توان به وضوح مشاهده نمود. اگر در یک فضای یک بعدی دارای ۱۰۰ نمونه داده باشیم، با حفظ چگالی تعداد این

¹ Nominal

² Data Warehouse

نمونه‌ها در یک فضای ۵ بعدی به $10^5 = 100^5$ می‌رسد. به عبارت دیگر در فضاهایی با ابعاد بالا برای جمع‌آوری کسری از داده‌ها، شاعر بیشتری از داده‌ها باید انتخاب شوند. فرمول زیر این خصوصیت را روشن‌تر بیان می‌کند:

$$E_d(p) = p^{1/d}$$

که در آن d تعداد ابعاد داده‌ها و p کسری از نمونه‌های است که مقدار آن مشخص است. برای مثال اگر بخواهیم فقط ۱۰ درصد از نمونه‌ها را جمع‌آوری کنیم، سطح‌های انتخابی در داده‌هایی با ابعاد ۲، ۳ و ۱۰ در ادامه محاسبه شده‌اند:

$$E_2(0.10) = (0.10)^{1/2} = 0.32$$

$$E_3(0.10) = (0.10)^{1/3} = 0.46$$

$$E_{10}(0.10) = (0.10)^{1/10} = 0.80$$

این محاسبات نشان می‌دهد که حجم زیادی از داده‌های کنار هم، فقط بخش کوچکی از داده‌ها را در فضاهایی با ابعاد بالاتر شامل می‌شوند.

خصوصیت دیگر داده‌های حجیم، فاصله‌ی بیشتر نمونه‌ها در ابعاد بالاتر است. برای تعداد

n نمونه داده در ابعاد d فاصله‌ی D به صورت زیر محاسبه می‌شود:

$$D(d, n) = 1/2 \times (1/n)^{1/d}$$

با داشتن ۱۰۰۰۰ نمونه این فاصله برای ابعاد ۲ و ۱۰ برابر است با:

$$D(2, 10000) = [(1/10000)^{1/2}] / 2 = 0.0005$$

$$D(10, 10000) = [(1/10000)^{1/10}] / 2 = 0.4$$

بنابراین با افزایش ابعاد و ثابت نگهداشت تعداد نمونه‌ها، فاصله‌ی میان داده‌های تخمینی نیز افزایش می‌یابد. اغلب این خصوصیات هنگامی که با تعداد محدودی از نمونه‌ها در فضاهایی با ابعاد بسیار بالا روبرو هستیم، اثر و نتیجه‌ای جدی را به دنبال دارند. تعداد نمونه‌ها، تعداد صفات خاصه و همچنین تعداد مقادیری که یک صفت خاصه می‌تواند داشته باشد (دامنه‌ی^۱ صفت خاصه)، حجم بالای داده‌ها را سبب می‌شوند. در داده‌های مربوط به یک فروشگاه زنجیره‌ای بزرگ با میلیون‌ها نمونه روبرو هستیم، یا در علم بیوانفورماتیک مجموعه داده‌ها با هزاران صفت خاصه توصیف می‌شوند. بدین علت علی‌رغم وجود تعداد

^۱ Domain

زیاد تکنیک‌های داده‌کاوی برخی از آنها قادر به رویارویی با این حجم وسیع داده را ندارند. به هر حال جهت بهبود مقیاس‌پذیری این الگوریتم‌ها نیز روش‌هایی ارائه شده‌اند، که در فصل‌های بعدی به آن خواهیم پرداخت.

۲-۳) آمار

در این بخش بصورت خلاصه به بیان برخی از شاخص‌های مرکزی^۱ و شاخص‌های پراکنده‌گی^۲ در آمار توصیفی می‌پردازیم. برای جزییات بیشتر و همچنین موضوعات مرتبط با آمار استنباطی می‌توانید به کتاب‌های آمار و احتمالات رجوع کنید.

۱-۲-۳) شاخص‌های مرکزی

این شاخص‌ها مقادیری هستند که با استفاده از آنها مجموعه‌ای از داده‌ها در یک اندازه یا عدد که نماینده‌ی آن مجموعه است، بیان می‌شود. از مهمترین و همچنین پرکاربردترین شاخص‌های مرکزی می‌توان به میانگین^۳، میانه^۴ و نما(مُد)^۵ اشاره کرد.

میانگین

یکی از معروف‌ترین شاخص‌های مرکزی میانگین است، که انواع آن عبارت است از میانگین حسابی^۶، میانگین هندسی^۷ و میانگین هارمونیک^۸. در متون علمی هر گاه نامی از نوع آن وجود ندارد، مقصود میانگین حسابی است که از تقسیم مجموع داده‌ها بر تعداد آنها بدست می‌آید.

$$\mu_A = \frac{x_1 + x_2 + \dots + x_n}{n} = 1/n \sum_{i=1}^n x_i$$

¹ Central Tendency

² Measures of Variation

³ Mean

⁴ Median

⁵ Mode

⁶ Arithmetic Mean

⁷ Geometric Mean

⁸ Harmonic Mean

چنانچه داده‌ها دارای وزن یکسانی (تأثیر یکسانی) نباشند، برای محاسبه‌ی میانگین حسابی آن باید هر یک از آنها را در وزن خود ضرب و حاصل جمع نهایی را تقسیم بر مجموع وزن‌ها کنیم. این میانگین به میانگین وزنی نیز شناخته می‌شود. توجه کنید که مقدار میانگین به شدت تحت تأثیر مقدار داده‌ها قرار دارد. به عبارت دیگر وجود داده‌های بسیار بزرگ و یا داده‌های بسیار کوچک باعث می‌شود تا مقدار میانگین به سوی این مقادیر حرکت کند. یکی دیگر از ویژگی‌های این شاخص این است که مجموع مجدورهای انحراف داده‌ها از میانگین یعنی مقدار $\sum_{i=1}^n (x_i - \mu)^2$ حداقل است.

میانگین هندسی n عدد از ریشه‌ی n حاصل ضرب آنها بدست می‌آید.

$$\mu_G = \sqrt[n]{x_1 \times x_2 \times \dots \times x_n} = \left(\prod_{i=1}^n x_i \right)^{1/n}$$

از میانگین هندسی موقعی استفاده می‌شود که صحبت از نرخ رشد یک ویژگی مطرح باشد. برای مثال جهت محاسبه‌ی نرخ رشد جمعیت، نرخ رشد سود و نرخ رشد تولید به این نوع میانگین رجوع می‌شود.

میانگین هارمونیک عبارت است از تقسیم تعداد داده‌ها بر مجموع معکوس مقادیر موجود در داده‌ها.

$$\mu_H = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \dots + \frac{1}{x_n}} = \frac{n}{\sum_{i=1}^n 1/x_i}$$

در یک مورد خاص که ما فقط دو مقدار داریم، این میانگین به صورت دیگری محاسبه می‌شود:

$$H = \frac{2 \cdot x_1 \cdot x_2}{x_1 + x_2}$$

بطور معمول هنگامی که واحد اندازه‌گیری داده‌ها ترکیبی باشد، از این نوع میانگین استفاده می‌شود. برای مثال اگر یک وسیله‌ی نقلیه فاصله‌ی میان دو شهر را در مرحله‌ی رفت با یک سرعت و در برگشت با سرعت دیگری طی کند، برای محاسبه‌ی میانگین سرعت آن از میانگین هارمونیک استفاده می‌کنیم. همانطور که می‌دانید واحد اندازه‌گیری سرعت، مسافت طی شده در واحد زمان است، که یک واحد ترکیبی است.

ثابت می‌شود که رابطه‌ی زیر برای سه نوع میانگین وجود دارد.

$$\mu_H \leq \mu_G \leq \mu_A$$

رابطه‌ی تساوی هنگامی برقرار است که مقادیر کلیه‌ی داده‌ها با یکدیگر برابر باشند.

میانه

میانه عددی است که توزیع داده‌ها را به دو قسمت مساوی تقسیم می‌کند، به نحوی که نیمی از داده‌ها بزرگتر و نیم دیگر کوچکتر از آن هستند. بنابراین در مجموعه اعداد مرتب شده، داده‌ی میانی به عنوان میانه لحاظ می‌گردد. در صورتی که تعداد داده‌ها زوج باشد، نصف مجموع دو داده‌ای که در وسط قرار دارند، میانه محسوب می‌شود. همچنین میانه برای داده‌های پیوسته‌ی دسته‌بندی شده از فرمول زیر محاسبه می‌شود:

$$m = L + \frac{(n/2 - g) \times w}{f}$$

که در آن L کران پایین دسته‌ای است که میانه در آن قرار دارد، n تعداد داده‌ها، g فراوانی تجمعی دسته‌ی قبل از دسته‌ی میانه، f فراوانی دسته‌ی میانه و w اندازه یا طول دسته را نشان می‌دهند.

توجه کنید که میانه بر خلاف میانگین تحت تأثیر داده‌های بسیار بزرگ و یا بسیار کوچک قرار نمی‌گیرد. به عبارت دیگر برای میانه تعداد داده‌ها اهمیت دارد. میانه هنگامی به کار برده می‌شود که قصد توصیف داده‌های ترتیبی و فاصله‌ای را داشته باشیم.

نما یا مُد

داده‌ای که فراوانی آن از سایر داده‌ها بیشتر باشد را نما یا مُد می‌نامیم. اگر دو داده‌ی مجاور دارای بیشترین فراوانی باشند، نصف مجموع آنها به عنوان نما لحاظ می‌گردد. در غیر این صورت اگر دو داده مجاور نباشند هر دو به عنوان نما انتخاب می‌شوند. در داده‌های دسته‌بندی شده، دسته‌ای که فراوانی آن از سایرین بیشتر است را به عنوان دسته‌ی نمایی انتخاب و نما را از فرمولی که در ادامه آمده، محاسبه می‌کنیم.

$$M = L + \left(\frac{D_{before}}{D_{before} + D_{after}} \right) \times w$$

که در آن L کران پایین دسته‌ی نمایی، D_{after} و D_{before} به ترتیب اختلاف فراوانی‌های نسبی دسته‌ی نمایی و دسته‌های قبل و بعد از آن و w نیز اندازه یا طول دسته را نشان می‌دهند.

۲-۳-۲) شاخص‌های پراکندگی

یکی از موضوعات مهم در مورد داده‌ها میزان تغییرات و پراکندگی آنها است، بدین معنی که اندازه‌گیری تا چه اندازه از داده به داده دیگر تغییر می‌کند. بدین منظور شاخص‌های پراکندگی به عنوان معیاری جهت سنجش تغییرات معرفی می‌شوند، که در این بخش به بررسی مختصر برخی از مهمترین آنها می‌پردازیم.

دامنه‌ی تغییرات و میانگین انحرافات

تفاوت میان بزرگترین و کوچکترین داده را به عنوان دامنه‌ی تغییرات^۱ داده‌ها می‌شناسیم.

$$R = \text{Max}(x_1, x_2, \dots, x_n) - \text{Min}(x_1, x_2, \dots, x_n)$$

به دلیل اینکه این شاخص تنها به بزرگترین و کوچکترین داده متکی است، نمی‌تواند شاخص مناسبی برای ارزیابی تغییرپذیری داده‌ها به شمار آید. به منظور دلالت تاثیر تمام داده‌ها می‌توان فاصله‌ی کلیه‌ی داده‌ها با میانگین را محاسبه نمود، که ما آنرا با انحراف از میانگین داده‌ها می‌شناسیم. پس از آن به طریق زیر میانگین تمام این انحرافات نیز محاسبه می‌شود:

$$D = \frac{1}{n} \sum_{i=1}^n |x_i - \mu_A|$$

هر چند میانگین انحرافات شاخص بهتری نسبت به دامنه‌ی تغییرات است، ولی محاسبه‌ی آن مشکل و به دلیل وجود قدر مطلق، ساده نمودن آن نیز دشوار است.

^۱ Range

انحراف چارکی

در آمار توصیفی به هر یک از سه مقداری که یک مجموعه داده‌ی مرتب را به چهار قسمت مساوی تقسیم می‌کنند، چارک گفته می‌شود. بدین ترتیب هر کدام از این بخش‌ها یک چهارم از جمعیت نمونه‌ها را نمایش می‌دهد. حال برای بدست آوردن انحراف چارکی کافی است نصف تفاصل بین چارک‌های سوم و اول را محاسبه کنیم. این شاخص نیز از دامنه‌ی تغییرات باشتر است، چرا که در واقع داده‌های بسیار بزرگ و بسیار کوچک به ترتیب در قسمت‌های چهارم و اول قرار می‌گیرند.

واریانس و انحراف استاندارد

میانگین مجدور انحرافات را واریانس^۱ می‌نامند.

$$S^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \mu_A)^2$$

در فرمول‌های آماری به جای تقسیم بر n واریانس را از مجموع مجدور انحرافات داده‌ها تقسیم بر $n-1$ بدست می‌آورند. با جذر گرفتن از مقدار واریانس یکی از پرکاربردترین شاخص‌های پراکندگی به نام انحراف استاندارد^۲ بدست می‌آید. انحراف استاندارد شاخص بسیار مفیدی است، زیرا می‌توان با کمک آن میزان پراکندگی هر توزیع پیوسته را بر حسب واحد اندازه‌گیری نشان داد. اما واریانس و انحراف استاندارد به واحد اندازه‌گیری داده‌ها بستگی دارند. بنابراین برای مقایسه‌ی دو مجموعه داده از شاخص دیگری مانند ضریب تغییر استفاده می‌کنیم. ضریب تغییر از تقسیم انحراف استاندارد به میانگین بدست می‌آید و به طور معمول با درصد نشان داده می‌شود.

۳-۳-۲) کوواریانس و ضریب همبستگی

در نظریه‌ی احتمالات کوواریانس^۳ اندازه‌ی تغییرات همانگ دو متغیر تصادفی است. اگر دو متغیر یکی باشند، کوواریانس برابر با واریانس خواهد بود.

¹ Variance

² Standard Deviation

³ Covariance

برای مقادیر صفات خاصه‌ی x و y مقدار کوواریانس برابر است با:

$$Cov(x, y) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y)$$

که در آن μ_x و μ_y به ترتیب میانگین برای دو صفت خاصه‌ی x و y را نشان می‌دهند. چنانچه این مقدار صفر باشد، میان دو صفت خاصه همبستگی وجود ندارد و دو صفت خاصه دارای رابطه‌ی خطی نیستند. مقدار مثبت آن نشان می‌دهد با افزایش یکی، دیگری نیز افزایش می‌یابد و مقدار منفی دلالت بر این دارد که افزایش یکی باعث کاهش دیگری می‌شود. اگر دو متغیر مستقل باشند، کوواریانس آنها صفر است ولی عکس این موضوع صحیح نیست. یعنی با صفر بودن کوواریانس دو متغیر تصادفی، نمی‌توان ادعا نمود آن دو مستقل هستند. برای داده‌هایی با ابعاد d مقادیر کوواریانس دو به دوی آنها در ماتریسی موسوم به ماتریس کوواریانس نگهداری می‌شود، که عنصر سطر i ام و ستون j ام آن از فرمول زیر محاسبه می‌شود:

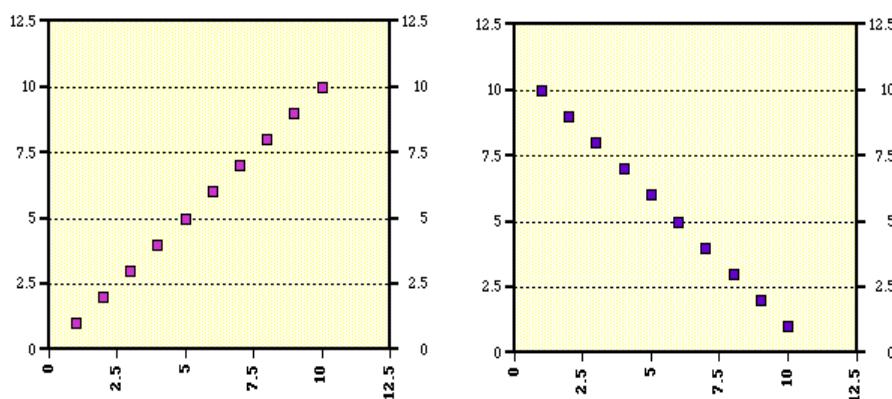
$$S_{ij} = \frac{1}{n} \sum_{i=1}^n (x_{ki} - \mu_k)(x_{kj} - \mu_j)$$

که در آن x_{ki} و x_{kj} به ترتیب مقدار i ام و j ام از k امین نمونه داده است. همبستگی به رابطه‌ی بین دو متغیر اشاره می‌کند، که می‌توان آن را با کمک نمودار پراکندگی نشان داد. ارزش مقداری آنرا ضریب همبستگی^۱ می‌نامیم و برای محاسبه‌ی آن از فرمول‌های متفاوتی استفاده می‌شود که در ادامه فرمول چگونگی محاسبه‌ی ضریب همبستگی خطی نشان داده شده است.

$$\text{Correlation}(x, y) = \frac{Cov(x, y)}{\sqrt{Cov(x, x) \cdot Cov(y, y)}} = \frac{Cov(x, y)}{\sqrt{Var(x) \cdot Var(y)}}$$

ضریب همبستگی فوق دارای دامنه‌ای بین -1 و $+1$ است. مقدار صفر عدم همبستگی را نشان می‌دهد و مقادیر $+1$ و -1 به ترتیب دلالت بر همبستگی کامل مثبت و همبستگی کامل منفی برای دو صفت خاصه‌ی مذبور را دارد (شکل ۲-۱).

^۱ Correlation



شکل ۱-۲: نمونه‌ای از همبستگی کامل مثبت (شکل چپ) و همبستگی کامل منفی (شکل راست)

۴-۳) توزیع نرمال

مهمترین توزیع پیوسته در آمار و احتمال، توزیع نرمال^۱ است که اغلب متغیرهای تصادفی پیوسته در طبیعت از آن پیروی می‌کنند. نمودار منحنی آن بستگی به دو پارامتر μ و σ^2 دارد و به صورت زیر بیان می‌شود.

$$P(x) = \frac{1}{\sqrt{2 \cdot \pi \cdot \sigma^2}} \times e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

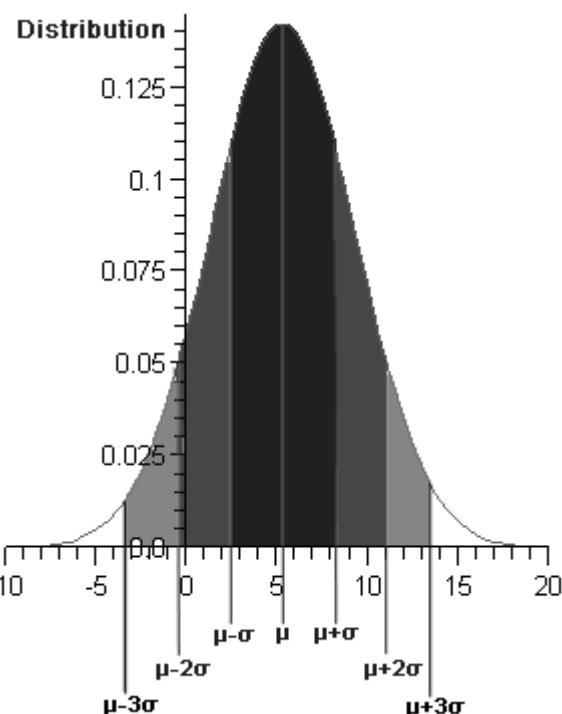
که در آن μ میانگین و σ^2 واریانس توزیع است. منحنی آن به شکل یک زنگوله و تنها دارای یک نقطه‌ی ماکزیمم در μ است (شکل ۲-۲). توزیع نرمال یک توزیع متقاض است و سطح زیر منحنی از $-\infty$ تا $+\infty$ ادامه دارد. برای تعیین سطوح مختلف زیرمنحنی از جدول توزیع نرمال استفاده می‌شود.

تحت یک توزیع نرمال نزدیک به ۶۸ درصد از داده‌ها بین دو مقدار $(\mu - \sigma)$ و $(\mu + \sigma)$ قرار می‌گیرند. درصد پوشش منحنی در شکل ۲-۲ نشان داده شده است، که برای دو و سه برابر مقدار σ نیز برابر است با:

$$(\mu \pm \sigma) = 68\% \quad , \quad (\mu \pm 2\sigma) = 95.5\% \quad , \quad (\mu \pm 3\sigma) = 99.7\%$$

¹ Normal Distribution

توزیع نرمالی که میانگین آن صفر و واریانس آن برابر با یک باشد را توزیع نرمال استاندارد^۱ می‌نامند.



شکل ۲-۲: نمونه‌ای از منحنی توزیع نرمال

۴-۲) لزوم آماده‌سازی داده‌ها

پایگاه داده‌های امروزی به دلیل حجم بسیار بالا، مستعد داده‌های نادرست و ناسازگار^۲ هستند. بطور حتم این داده‌های ناقص و افزونه می‌توانند در نتایج عملیاتی مثل داده‌کاوی خلل ایجاد کند. اصطلاح داده‌های ناقص به وضعیتی اشاره می‌کند، که داده‌های موجود حاوی اطلاعات کافی جهت استخراج دانش جدید نیستند. عدم وجود مقدار^۳ برای یک

¹ Standard

² Inconsistent

³ Missing Data

صفت خاصه نمونه‌ای از این نوع داده است. هرگاه با دو یا تعداد بیشتری از نمونه‌های یکسان برخورد کنیم، یا تعدادی از صفات خاصه با یکدیگر همبستگی قوی دارند، در واقع با داده‌های افزونه روبرو هستیم. در این شرایط ممکن است با موضوع ناسازگاری داده‌ها مواجه شویم. این خیلی مهم است که داده‌ها قبل از هرگونه اقدامی آزمایش شوند. تحلیل‌گرها قبل از اینکه مدلی از داده‌ها را درست کنند، یا از الگوریتم‌های داده‌کاوی استفاده کنند، باید خودشان با ماهیت داده‌ها آشنا باشند. اما با حجم زیاد داده‌ها این آشنایی یا بسیار دشوار است و یا امکان‌پذیر نیست. حذف داده‌های نادرست و اصلاح کردن مقادیر ناسازگار، جمع‌آوری داده‌های مورد نیاز برای تحلیل در ابزار داده‌ها و همچنین کاهش حجم داده‌ها به منظور تسريع در عملیات با داده‌ها پیش از داده‌کاوی به طور حتم موثر خواهد بود.

به منظور تسهیل و بهبود فرایند داده‌کاوی آمده‌سازی داده‌ها^۱ یکی از مراحل اساسی تلقی می‌شود. بسیاری از کارشناسان داده‌کاوی در اینکه آمده‌سازی داده‌ها یکی از بحرانی‌ترین مراحل موجود در فرایند استخراج دانش است، اتفاق نظر دارند. نامفهوم بودن داده‌ها و استفاده‌ی نادرست از ابزار داده‌کاوی، می‌تواند این فرایند را در مسیری نادرست قرار دهد. از اینرو می‌توان گفت داده‌کاوی فقط راهنمای استفاده از ابزاری برای مشکل مطرح شده نیست، بلکه یک فرایند بحرانی اکتشافی است و به همین دلیل داده‌ها باید برای این عمل مهم درست و سازگار تعریف شوند. به دلایل فوق و همانطور که قبل از این نیز اشاره کردیم، بسیاری از کارشناسان حوزه‌ی داده‌کاوی، آمده‌سازی و تغییر شکل مناسب داده‌های اولیه را یکی از بحرانی‌ترین گام‌ها می‌دانند.

این نکته درست است که این مرحله وابسته به نوع برنامه‌ی کاربردی است، اما در بسیاری از برنامه‌های کاربردی بدون در نظر گرفتن تکنیک داده‌کاوی می‌توان برخی از روش‌های آمده‌سازی داده‌ها را استفاده نمود. اینکه کامپیوتر به تنهایی و بدون کمک انسان بتواند بهترین روش آمده‌سازی داده‌ها را انتخاب کند، انتظار بیهوده‌ای است. حتی انتظار نداریم که روش‌های به کاربرده شده برای آمده‌سازی داده‌ها در یک برنامه‌ی کاربردی بهترین باشند.

¹ Data Preparing

تصور کنید شما مدیر یک سازمان هستید و تصمیم به تحلیل اطلاعاتی از سازمان مزبور را دارید. در میان داده‌ها صفات خاصه‌ای از پایگاه داده را انتخاب و در انتظار جواب می‌مانید. مقادیر برخی از این صفات خاصه در نمونه‌ها نامشخص هستند. به عبارت دیگر داده‌های مورد نظر شما برای داده‌کاوی ناتمام^۱ و یا می‌توانند ناسازگار و نادرست باشند. این مسئله می‌تواند در تصمیم‌گیری نهایی شما موثر باشد.

داده‌ها به دلایل متعددی می‌توانند ناتمام یا ناقص باشند. بطور مثال مقادیر بعضی از صفات خاصه در زمان ورود در دسترس نبوده یا حداقل در آن زمان وجودشان الزامی نبوده است. پایگاه داده‌ی آموزشی دانشگاه را تصور کنید. مشخصات دانشجو بدون مقدار برای صفت خاصه‌ی آدرس یا نام پدر می‌تواند ثبت شود. خطاهای انسانی و ماشین (مانند خطأ در انتقال داده‌ها و محدودیت‌های سخت‌افزاری) نیز می‌توانند در داده‌های ما خلل ایجاد کنند. در مرحله‌ی آماده‌سازی داده‌ها مواردی اینچنین که باعث گیج شدن فرایند داده‌کاوی می‌شود را باید رفع نمود.

برخی اوقات داده‌ها باید از چندین منبع و یا فایل جمع‌آوری شوند. این منابع می‌توانند ناهمگن^۲ یا نامتجانس و از پایگاه‌داده‌های مختلف جمع‌آوری شوند. در این صورت ممکن است با صفات خاصه‌ای با مفاهیم یکسان ولی نامهای مختلف روی رو شوید و این باعث افزونگی داده‌ها و شاید ناسازگاری آنها شود. موضوع دیگر مربوط به مقادیری است که می‌توان آنها را از داده‌های دیگر بدست آورد. وجود این مقادیر حجم داده‌های ورودی برای داده‌کاوی را افزایش می‌دهد. بطور مثال فرض کنید در پایگاه داده‌ی پزشکی یک بیمارستان وزن و یا قد بیماران در بخش‌های مختلف آن با واحدهای متفاوت اندازه‌گیری و نگهداری می‌شوند.

استفاده از روش‌هایی نظیر شبکه‌های عصبی و الگوریتم‌های ژنتیک در فرایند داده‌کاوی می‌تواند دلیلی برای تغییر شکل داده‌ها باشد. بطور مثال در پایگاه‌داده‌ی دانشگاه تغییر نمرات دانشجویان از محدوده‌ی صفر تا بیست به محدوده‌ی صفر تا یک برای ورودی به

¹ Incomplete

² Heterogeneous

لایه‌های شبکه‌ی عصبی می‌تواند مفید باشد. همچنین تبدیل مقادیر به محدوده‌ی خاص و واحد می‌تواند به یکسان‌سازی تأثیر صفات خاصه‌ی مختلف نیز کمک کند. شاید بطور کلی و خلاصه بتوان دو وظیفه‌ی زیر را برای مرحله‌ی آماده‌سازی داده‌ها در نظر گرفت:

- سازماندهی داده‌ها در یک شکل استاندارد تا برای پردازش در عمل داده‌کاوی مناسب باشند.
 - آماده‌سازی مجموعه داده‌ها تا الگوریتم‌های داده‌کاوی بتوانند با کارایی بالایی اجرا شوند.
- گاهی به تکنیک‌های آماده‌سازی داده‌ها، تکنیک‌های پیش پردازش داده‌ها^۱ نیز گفته می‌شود.

۳-۵) تکنیک‌های آماده‌سازی داده‌ها

آماده‌سازی داده‌ها به مراحل قبل از داده‌کاوی اطلاق می‌گردد، هرچند از این تکنیک‌ها می‌توان در حین اجرای الگوریتم‌های داده‌کاوی نیز استفاده نمود. همانطور که اشاره شد، این مرحله گاهی با نام مرحله‌ی پیش پردازش داده‌ها نیز شناخته می‌شود. آماده‌سازی داده‌ها یکی از مهم‌ترین گام‌ها در فرایند توسعه‌ی مدل‌ها به شمار می‌رود. کیفیت داده‌های ورودی در ساده‌ترین تحلیل تا ساخت مدل‌های پیچیده یکی از کلیدهای موفقیت انجام یک پروژه به حساب می‌آید. با توجه به این نکته که مقدار و پیچیدگی داده‌ها روز به روز رو به افزایش است، توانایی مدل جهت تولید نتایج بدربخور بطور کامل به داده‌های خوب و درست ورودی متکی است.

در این بخش به بررسی تکنیک‌های آماده‌سازی داده‌ها در قالب چهار نوع عملیات پالایش داده‌ها^۲، جمع‌آوری داده‌ها^۳، تغییر شکل داده‌ها^۴ و کاهش حجم داده‌ها^۵ می‌پردازیم. توجه

¹ Data Preprocessing

² Data Cleaning

³ Data Integration

⁴ Data Transformation

⁵ Data Reduction

کنید که برخی از روش‌های ارائه شده برای یک دسته را می‌توان برای گروه دیگر نیز به کار برد.

۱-۵-۲) پالایش داده‌ها

حتی در برنامه‌های کاربردی واقعی با مقدار بسیار زیاد داده‌ها، می‌توان نمونه‌هایی را یافت که مقداری برای صفات خاصه‌ی آنها وجود ندارد. اگر تکنیک‌ها از الگوریتم‌های قدرتمندی استفاده کنند، شاید این مقادیر ناقص در نتیجه‌ی نهایی فرایند بی‌تأثیر یا حداقل کم اثر باشند. اما تکنیک‌های داده‌کاوی کم و بیش به این مقادیر ناقص حساس هستند.

ساده‌ترین راه حل برای این مشکل صرف نظر از نمونه‌های ناقص است و این در صورتی امکان‌پذیر است که داده‌های در دسترس به اندازه‌ی کافی وجود داشته باشند و داده‌های ناقص کسر کوچکی از داده‌ها را تشکیل دهند. در مواردی که تعداد عضوهای دامنه‌ی یک متغیر کم یا تعداد داده‌های ناقص کم باشند، می‌توان به صورت دستی نمونه‌های ناقص را کامل کرد. همانطور که مشخص است، این راهکار برای پایگاه داده‌ی بسیار بزرگ زمانگیر است و از همه مهم‌تر تعیین مقدار برای داده‌های ناقص حتی به صورت دستی مشکل بزرگی است. استفاده از هر مقداری برای تکمیل داده‌های ناقص ممکن است در تحلیل نهایی موثر باشد. اما به هرحال شما مجبور هستید یا از این داده‌های ناقص صرفنظر کنید، یا مقداری را برای آنها در نظر بگیرید.

در بسیاری از سیستم‌های پایگاه داده‌ها برای جلوگیری از قرار دادن مقدار *NULL* در پایگاه داده، یک یا چند مقدار پیش فرض را برای آن صفت خاصه در نظر می‌گیرند. اما همانند دیگر روش‌ها می‌توان گفت که همیشه مناسب یا امکان‌پذیر نیست.

برای متغیرهای عددی قرار دادن مقدار میانگین صفت خاصه در نمونه‌های ناقص، روش ساده‌ی دیگری برای حل این مشکل است. روش دیگر این است که برای دسته‌های یکسانی از نمونه‌ها، مقدار میانگین محاسبه و جایگزین شود. در این روش باید ابتدا با کمک یک الگوریتم نمونه‌ها در دسته‌های مختلف قرار داده شوند. سپس از آن برای نمونه‌های موجود در هر دسته به صورت مجزا مقداری را جایگزین ویژگی‌های ناقص

خواهیم کرد. این مقدار می‌تواند میانگین مقادیر یک صفت خاصه در یک دسته یا مقدار محاسباتی آماری دیگری مانند مُد باشد.

بدون شک این روش‌ها با مشخص کردن مقدار برای داده‌های ناقص باعث سوگیری الگوریتم‌های داده‌کاوی می‌شوند. استفاده از تکنیک‌هایی موسوم به تکنیک‌های تخمين (مانند درخت تصمیم^۱) یکی از استراتژی‌های مرسوم در برنامه‌های کاربردی داده‌کاوی است، که می‌تواند در این گونه موارد به ما کمک کند.

راهکار مناسب دیگر اجرا و مقایسه‌ی نتایج چندین روش متفاوت است. برای مثال می‌توان ابتدا از داده‌های ناقص صرفنظر نموده و نتیجه‌ی داده‌کاوی را مشاهده و پس از آن با جایگزین نمودن یک مقدار دوباره نتیجه‌ی حاصل را مشاهده و تحلیل کنیم. توجه کنید در تحلیل نهایی داده‌ها لزومی به ارائه فقط یک نتیجه نیست.

تحلیل داده‌های خارج از محدوده^۲

غلب در مجموعه‌ی بزرگی از داده‌ها نمونه‌هایی وجود دارند که رفتارشان با رفتار عمومی نمونه‌ها یکسان نیست. این رفتار یا کامل مختلف است و یا با دیگر نمونه‌ها ناسازگارند. به عبارتی دیگر همیشه داده‌های ما ناقص نیستند، می‌توانند وجود داشته باشند، اما با رفتاری متفاوت از اکثر نمونه‌های موجود. وجود این نمونه‌ها می‌تواند دلایل متعددی مثل خطاهای ماشین یا خطاهای انسانی یا وجود انحرافی در یک متغیر اندازه‌گیری شده باشد.

اکثر پایگاه داده‌هایی که در جهان خارج قرار دارند، شامل داده‌هایی گمشده و کم، نامعلوم و ناقص یا اشتباه هستند. در یک سیستم پایگاه داده‌ها وجود شروط جامعیت^۳ مناسب می‌تواند از وقوع درصدی از این موارد جلوگیری کند. اما با این حال مقادیری هستند که وجودشان مجاز است ولی با دیگر نمونه‌ها متفاوت هستند.

برخی از الگوریتم‌های داده‌کاوی تأثیر این داده‌های خارج از محدوده را نادیده می‌گیرند و یا با کمک برخی از الگوریتم‌های مرحله‌ی آماده‌سازی داده‌ها آنرا حذف می‌کنند. حذف

¹ Decision Tree

² Outlier

³ Integrity Rules

این نمونه‌ها در صورت درست بودن آنها بطور حتم در نتیجه‌ی نهایی موثر است. به علاوه قابلیت تشخیص آنها برای حذف نیز خود چالش دیگری است.

شاید بتوان به صورت خوش‌بینانه این طور بیان کرد که در اکثر برنامه‌های کاربردی این داده‌ها خیلی مفید نیستند و نتیجه خطاهایی هستند که در جمع‌آوری داده‌ها بوجود آمده‌اند، اما همیشه این طور نیست. نمرات درسی یک کلاس ۵۰ نفره را در نظر بگیرید. اینکه تعداد کمی از دانشجوها (برای مثال ۲ نفر) نمره‌ی بسیار بالا و یا بسیار پایینی کسب کرده باشند و مابقی نمراتی مانند ۱۲ یا ۱۳ و یا ۱۴ داشته باشند دور از ذهن نیست. حداقل نمرات برای دانشجویان (۲ دانشجوی مذکور) وجود دارند و می‌تواند ما را گمراه کند. برای مثال برای بیان میانگین نمرات این کلاس بهتر است میانگین بدون احتساب آن ۲ نمره محاسبه شود. چرا که میانگین بسیار تحت تأثیر مقادیر بزرگ یا کوچک (نسبت به اکثر داده‌ها) قرار می‌گیرد. راه‌های مختلفی برای تشخیص این نمونه‌های خارج از محدوده وجود دارند، که در ادامه به بعضی از آنها اشاره می‌کنیم.

در فضای یک تا سه بعدی می‌توانیم از نمایش و تجسم‌سازی^۱ داده‌ها برای تشخیص استفاده کنیم. عدم وجود روش‌های مناسب تجسم‌سازی برای ابعادی بالاتر از سه از جمله محدودیت‌های این روش به شمار می‌رود.

یکی از ساده‌ترین راهکارها برای نمونه‌هایی در فضای یک بعدی استفاده از توابع آماری است. محاسبه‌ی میانگین و انحراف استاندارد و تعریف تابعی از این دو مقدار برای سطح آستانه داده‌ها می‌تواند در تشخیص داده‌های خارج از محدوده به ما کمک کند. این روش برای داده‌هایی که توزیع آنها شناخته شده باشد، مناسب است. برای مثال تابع زیر که از میانگین (*Mean*) و انحراف استاندارد (*St_Dev*) استفاده می‌کند، می‌تواند در شناسایی نمونه‌های مزبور کمک کند.

$$\text{Threshold} = \text{Mean} \pm 2 \times \text{St_Dev}$$

داده‌هایی که در محدوده دو مقدار *Threshold* قرار نمی‌گیرند، به عنوان داده‌های خارج از محدوده شناخته می‌شوند.

¹ Visualization

از میان روش‌های دیگر که محدودیت‌های استفاده از توابع آماری و پارامترها در آن کمتر باشد می‌توان به روش‌های تشخیص مبتنی بر فاصله^۱ اشاره کرد. همانطور که از نام روش مشخص است در این راهکار فاصله‌ی میان نمونه‌ها محاسبه می‌شود. پس از آن هر نمونه که از سایرین به اندازه‌ی کافی دور باشد و یا به اندازه‌ی کافی دارای همسایه‌هایی با فاصله‌ی مشخص نباشد، خارج از محدوده تشخیص داده می‌شود. برای مثال در یک روش اگر فاصله‌ی یک نمونه با حداقل تعداد p نمونه‌ی دیگر بیشتر از مقدار معین d باشد، این نمونه خارج از محدوده تشخیص داده می‌شود. این روش برای داده‌هایی با ابعاد بزرگ نیز عملکرد مناسبی دارد، ولی مقدار پارامترهای p و d باید به دقت انتخاب شوند. در ضمن مفهوم کلیدی فاصله نیز باید به روشنی تعریف شود. در فصلی که در آن روش‌های خوشه‌بندی^۲ توضیح داده می‌شوند، بیشتر به مفهوم فاصله می‌پردازیم. روش‌هایی مانند دسته‌بندی^۳، خوشه‌بندی و رگرسیون^۴ روش‌های رایجی هستند که نه تنها در بسیاری از تکنیک‌های آماده‌سازی داده‌ها که در سراسر فرایند داده‌کاوی نیز از آنها استفاده می‌شود. این روش‌ها می‌توانند جهت پالایش داده‌ها نیز به ما کمک کنند.

دسته‌بندی داده‌ها

داده‌های کمی را می‌توان دسته‌بندی کرد و این تکنیک هم می‌تواند به منظور تشخیص و حذف داده‌های نویز^۵ و مزاحم استفاده شوند، هم اینکه برای کاهش حجم داده‌ها نیز روش مفیدی است. فرض کنید که صفت خاصه‌ای شامل یک سری داده‌های کمی با محدوده‌ی مشخص و قابل شمارش است. در صورت کاهش تعداد داده‌ها می‌توانیم امیدوار باشیم تا روش‌های داده‌کاوی کارایی بهتری از خود نشان می‌دهند.

داده‌ها می‌توانند به طرق مختلفی دسته‌بندی و پس از آن داده‌های هر دسته با یک مفهوم کلی‌تر دیگری نمایش داده شوند. اغلب این مفهوم کلی‌تر یک شاخص مرکزی مانند مقدار

¹ Distance-Based Outlier Detection

² Clustering

³ Binning Methods

⁴ Regression

⁵ Noise

میانگین، نما و یا میانه در هر دسته است. اگر تعداد داده‌ها خیلی زیاد نباشد، برای پیدا کردن معیارهای مرکزی و پراکندگی و بررسی خواص داده‌ها احتیاج به دسته‌بندی آنها نیست، ولی اگر تعداد داده‌ها زیاد باشد، مفید خواهد بود اگر آنها را دسته‌بندی نماییم. برای داده‌هایی با حجم متوسط و زیاد مقدار میانگین یا مُد و برای موافقی که حجم داده‌ها محدود و کم هستند، مقادیر کناری دسته‌ها می‌تواند انتخاب مناسبی باشد.

برای مثال مجموعه‌ی ۱۰ عضوی $\{3, 2, 1, 5, 4, 3, 1, 7, 5, 3\}$ را در نظر بگیرید (با اغماس نسبت به تعریف مجموعه‌ها در ریاضی، که در آن عضو تکراری وجود ندارد). ابتدا این مجموعه را مرتب می‌کنیم. مجموعه مرتب شده‌ی $\{1, 1, 2, 3, 3, 4, 5, 5, 7\}$ را در سه گروه دسته‌بندی می‌کنیم.

$$\{1, 1, 2\}, \quad \{3, 3, 3\}, \quad \{4, 5, 5, 7\}$$

با استفاده از مقدار میانگین، سه دسته به صورت زیر نمایش داده می‌شوند:

$$\{1/33, 1/33, 1/33\}, \quad \{3, 3, 3\}, \quad \{5/25, 5/25, 5/25, 5/25\}$$

با محاسبه مقدار مُد در هر دسته نمایش به صورت زیر است:

$$\{1, 1, 1\}, \quad \{3, 3, 3\}, \quad \{5, 5, 5, 5\}$$

و بالاخره می‌توانیم از مقادیر کناری دسته‌ها برای نمایش آنها استفاده کنیم:

$$\{1, 1, 2\}, \quad \{3, 3, 3\}, \quad \{4, 4, 4, 7\}$$

همانطور که از مطالب بالا استنباط می‌شود، انتخاب تعداد و اندازه‌ی دسته‌ها خیلی مهم است. تعیین نمودن تعداد دسته‌های مناسب در مورد هر مجموعه از داده‌ها بستگی به تشخیص و ابتکار و تجربه‌ی شخص تحلیل‌گر دارد. اما نوع داده‌ها، تعداد و پراکندگی آنها در تعیین تعداد دسته‌ها نمی‌تواند بی‌تأثیر باشد. در تعیین تعداد دسته‌ها باید دقت کرد، زیرا اگر تعداد آنها کم انتخاب شوند، اندازه‌های کمی بصورت فشرده درمی‌آیند و اطلاعات اساسی ظاهر نمی‌گردد و دقت کم می‌شود. بر عکس هرقدر تعداد دسته‌ها زیادتر باشد، جزئیات داده‌ها بهتر مشخص می‌گردد، ولی گسترش دسته‌ها، محاسبه و نتیجه‌گیری را مشکل می‌سازد. از این رو بهتر است تعداد گروه‌ها با تعداد داده‌ها متناسب باشد.

یکی از روش‌های ساده‌ی تعیین تعداد دسته‌ها برای تعداد n داده استفاده از فرمول زیر است:

$$K=1+3.3 \times \log n$$

در این فرمول عدد محاسبه شده برای K در نهایت گرد می‌شود.

تعیین فاصله‌ی دسته‌ها با تعداد آنها مرتبط است. بیشتر اوقات دسته‌ها دارای فواصل مساوی هستند، ولی ممکن است فاصله‌ی دسته‌ها را از فرمول زیر محاسبه کنیم:

$$D=R/K$$

که در آن R دامنه تغییرات و K تعداد دسته‌ها است، که می‌توان با راهکار ذکر شده در بالا آن را بدست آورد. طبیعی است حاصل تقسیم بالا نیز در نهایت گرد می‌شود.

روش دیگر برای مشخص کدن تعداد عضوهای هر دسته به صورتی است که میانگین فواصل مقادیر یک دسته از میانگین یا میانه‌ی دسته‌ی خودش به حداقل برسد. به طور معمول در اجرای الگوریتم برای فاصله‌ی میانگین‌ها محدود و برای فاصله‌ی میانه‌ها قدر مطلق لحاظ می‌شود. روش را با یک مثال دنبال می‌کنیم.

مجموعه‌ی $\{1, 5, 1, 8, 2, 2, 9, 2, 8, 6\}$ را پس از مرتب کردن اعضاء به سه دسته‌ی زیر گروه‌بندی کرده‌ایم:

$$\{1, 1, 2\} \quad , \quad \{2, 2, 5\} \quad , \quad \{6, 8, 8, 9\}$$

برای هر دسته مُد را محاسبه می‌کنیم.

$$\{1, 2, 8\}$$

فاصله‌ی عضوهای هر دسته با مُد همان دسته را به عنوان خطای هر عضو محاسبه و با یکدیگر جمع خواهیم کرد. عدد حاصل می‌تواند ما را در انتخاب تعداد اعضاء گروه کمک کند. در مورد مثال نظر این عدد برابر است با:

$$\begin{aligned} Error &= |1-1| + |1-1| + |2-1| + |2-2| + |2-2| + |5-2| + |6-8| + |8-8| + |8-8| + |9-8| \\ &= 7 \end{aligned}$$

با انتقال عناصر کناری هر دسته به دسته‌ی همسایه روش را تکرار و مقدار خطای حاصل محاسبه می‌شود. این کار تا جایی ادامه پیدا می‌کند تا مقدار خطای حاصل خود برسد.

برای مثال مزبور با انتقال ۲ از دسته‌ی دوم به اول و همچنین انتقال ۶ از دسته‌ی سوم به دوم گروه‌ها به صورت زیر اصلاح می‌شوند:

$$\{1,1,2,2,2\} , \{5,6\} , \{8,8,9\}$$

مقدار مُد برای ۳ دسته برابر است با:

$$\{2,5,8\}$$

و مقدار خطای بدست آمده به عدد ۴ کاهش می‌یابد.

$$Error=1+1+0+0+0+1+0+0+1=4$$

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 شبیه بوده ولی متفاوت از نمونه‌های دیگر خوشبندی است.

تحلیل خوشبندی 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 خوشبندی کاربردهای بسیار مختلفی از جمله تشخیص الگو^۱، تحلیل داده‌ها، پردازش تصاویر^۲ و تحلیل‌های تجاری دارد. با این شیوه می‌توان مناطق پر جمعیت و کم جمعیت را شناسایی نمود و بدین ترتیب پراکندگی و همبستگی‌های جالب میان خصوصیات داده‌ها را کشف نمود.

در عالم تجارت، خوشبندی به بازاریان کمک می‌کند که در میان مشتریان خود گروه‌های متمایز و متفاوتی را پیدا کنند و یا گروه‌های مشتریان را بر اساس الگوهای خرید مشخص سازند. خوشبندی همچنین می‌تواند به ما کمک کند تا استناد موجود در اینترنت را برای کشف اطلاعات، طبقه‌بندی کنیم. تحلیل خوشبندی به عنوان عملکردی برای داده‌کاوی را می‌توان ابزاری مستقل برای کسب آگاهی از پراکندگی داده‌ها دانست. همچنین مشاهده‌ی خصوصیات هر خوشبندی و تمرکز بر مجموعه‌ی خاصی از خوشبندی‌ها برای انجام تحلیل‌های

¹ Pattern Recognition

² Image Processing

بیشتر مناسب است. خوشبندی می‌تواند یک عمل پیش‌پردازش برای الگوریتم‌های دیگر که در خوشبندی شناسایی شده به کار می‌روند، باشد.

خوشبندی یکی از شاخه‌های دشوار تحقیقاتی است که کاربردهای گوناگون آن شرایط خاصی را نیز می‌طلبد. به دلیل حجم زیاد اطلاعات جمع‌آوری شده در پایگاه‌داده‌ها، خوشبندی به موضوعی بسیار جدی در تحلیل درباره‌ی داده‌ها تبدیل شده است. از تکنیک‌های خوشبندی برای تشخیص داده‌های خارج از محدوده و یا داده‌های ناقص می‌توان استفاده کرد. توضیحات مفصل‌تر در مورد خوشبندی و الگوریتم‌های مربوط به آن در فصل‌های هشتم و نهم ارایه شده‌اند.

رگرسیون

این تکنیک آماری قدرتمند می‌تواند در کلیه‌ی مراحل فرایند داده‌کاوی داده‌ها استفاده شود. رگرسیون خطی^۱، رگرسیون چندگانه^۲ و رگرسیون غیرخطی^۳ از انواع آن به شمار می‌روند.

در رگرسیون خطی دو نمونه‌ی مختلف از داده‌ها به صورت یک خط راست مدل‌سازی می‌شوند. خط رگرسیون ابزاری است برای پیش‌بینی مقدار یک متغیر بر حسب متغیری که به آن وابسته است. در واقع برای مدل‌سازی مقادیر دو صفت‌خاصه، خطی را می‌یابیم که به همه‌ی زوج مقادیر این دو صفت‌خاصه نزدیک باشد. چنانچه زوج‌های مرتب بر روی یک خط راست نباشند، برای هر زوج و خط رگرسیون خطایی وجود دارد. این خط به نحوی انتخاب خواهد شد تا خطا به حداقل خود برسد. به حداقل رساندن مجموع مربعات خط‌ها^۴ روش مرسومی است که در بیشتر موارد از آن استفاده می‌شود.

برای دو متغیر X و Y معادله‌ی خط زیر را در نظر بگیرید:

$$Y = \alpha + \beta X$$

¹ Linear Regression

² Multiple Regression

³ Nonlinear Regression

⁴ Sum of Square Errors

مقادیر α و β از فرمول‌های زیر بدست می‌آیند و با نام ضریب همبستگی^۱ رگرسیون شناخته می‌شوند.

$$\alpha = \bar{Y} - \beta \bar{X}$$

$$\beta = \frac{\sum_{i=1}^n (x_i - \bar{X})(y_i - \bar{Y})}{\sum_{i=1}^n (x_i - \bar{X})^2}$$

که در آن \bar{X} و \bar{Y} به ترتیب میانگین زوج‌های مرتب هستند و x_i ها و y_i ها نیز مقادیر نمونه‌های موجودند که تعدادشان به n می‌رسد.

رگرسیون چندگانه حالت توسعه‌یافته‌ای از رگرسیون خطی است، با این تفاوت که در آن بیش از دو متغیر وجود دارند و مقدار یکی با توجه به دیگر متغیرها تخمین زده می‌شود. صورت کلی آن به شکل زیر است:

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2$$

در اینجا نیز می‌توانیم از روش حداقل کردن مجموع مربعات خطاهای استفاده و مقادیر ضرایب یعنی α و β_1 و β_2 را محاسبه کنیم. بسیاری از مسائل با رگرسیون خطی حل می‌شوند.

در رگرسیون خطی همان طور که از نام آن مشخص است، داده‌ها را با معادله‌ی یک خط مدل‌سازی می‌کنیم. حال آنکه در بعضی از موقع لازم است ارتباط بین متغیرها بوسیله‌ی یک تابع چندجمله‌ای^۲ مدل‌سازی شود. خیلی از مشکلات غیرخطی را می‌توان به خطی تبدیل کرد، اما این تبدیل گاهی به سادگی امکان‌پذیر نیست. لذا برای بدست آوردن حداقل مربعات باید از فرمول‌های پیچیده‌ای استفاده کرد.

علیرغم استفاده‌ی گسترده از رگرسیون خطی مدل‌های دیگری از رگرسیون نیز وجود دارند که برای موارد خاصی مفیدند. در رگرسیون خطی عملکرد مقادیر پیوسته که یکی از انواع مرسوم از داده‌های کمی است، مدل‌سازی می‌شود.

¹ Coefficient

² Polynomial

با تعمیم مدل‌های خطی می‌توانیم برای متغیرهای طبقه‌بندی شده نیز تخمینی داشته باشیم. رگرسیون لجستیک^۱ و رگرسیون پویسون^۲ انواع دیگری از رگرسیون‌ها هستند که این کار را انجام می‌دهند. امروزه نرم‌افزارهای بسیاری مانند SAS, SPSS, S-Plus وجود دارند، که می‌توان از آنها برای حل مسائل رگرسیون کمک گرفت.

۲-۵-۲) جمع‌آوری داده‌ها

جمع‌آوری و ترکیب داده‌ها از چندین منبع به درون یک محل منسجم به نام انبارداده‌ها^۳ یکی از وظایف اولیه در فرایند داده‌کاوی محسوب می‌شود. این منابع می‌تواند شامل چند پایگاهداده‌ی ناهمگن^۴ باشد. افزونگی و در نتیجه ناسازگاری داده‌ها یکی از مسائل مهمی هستند، که در جمع‌آوری داده‌ها باید به آن توجه کرد.

صفات خاصه‌ی یکسانی که از منابع مختلف جمع‌آوری می‌شوند را می‌توان توسط داده‌های میتا^۵ که معمولاً هر پایگاهداده یا انبارداده شامل آن است، تشخیص داد و بدین ترتیب از افزونگی جلوگیری نمود.

نگهداری و یا حذف صفات خاصه‌ای که مقادیر آنها از مقدار دیگر صفات مشتق می‌شوند، به دلیل استفاده از فرایند داده‌کاوی بر می‌گردد. چنانچه بدست آوردن این صفات مشتق شده از پیچیدگی زمانی بالایی برخوردار باشد، بهتر آن است که این صفات نگهداری شوند. در غیراینصورت با تشخیص مناسب، آنها را حذف می‌کنیم. با محاسبه‌ی همبستگی میان صفات خاصه می‌توان این نوع افزونگی را کنترل نمود. در هر حال تشخیص افزونگی بین صفات خاصه می‌تواند به تشخیص در سطح نمونه‌ها نیز منجر شود.

تفاوت در طریقه‌ی نمایش داده‌ها، مقیاس و واحدهای مختلف اندازه‌گیری و همچنین کدگذاری ویژگی‌ها می‌تواند مشکلاتی را در این مرحله برای ما ایجاد کند. بطور مسلم این ناهمگونی معنایی داده‌ها چالش بزرگی است که به راحتی و فقط با محاسبه‌ی فرمول و به

¹ Logistic Regression

² Poisson Regression

³ Data Warehouse

⁴ Heterogeneous

⁵ Metadata

صورت خودکار نمی‌توان بر آن فائق شد. لذا در این مرحله دانش تحلیل‌گر و طراحی خوب پایگاه‌داده‌ها می‌تواند کمک بزرگی محسوب شود.

۲-۵-۳) تغییر شکل داده‌ها

شکل مناسب داده‌ها به عنوان ورودی الگوریتم‌های داده‌کاوی نقش به سزایی در این فرایند بازی می‌کند و در مرحله‌ی آماده‌سازی داده‌ها این نقش پررنگ است. تکنیک‌های تغییر شکل داده‌ها متنکی به مشکل نیستند و اغلب در اجرا نتایج بهتری از داده‌کاوی را سبب می‌شوند.

استفاده از توابع تجمعی^۱، نرمال‌سازی^۲، خوشبندی و رگرسیون روش‌های مرسومی هستند که برای تغییر و تبدیل شکل داده‌ها می‌توان از آنها استفاده نمود. فراموش نکنید انتخاب روش مناسب به ماهیت داده‌ها، مقدار آن و اهداف داده‌کاوی بستگی دارد. در جمع آوری داده‌ها و ساخت انبارداده‌ها از توابع تجمعی نظیر ماکریم، مینیمم، میانگین و مجموع استفاده می‌شود. این توابع علاوه بر تغییر شکل داده‌ها می‌توانند از حجم آن نیز بکاهند. در فصل بعدی در مورد انبار داده‌ها و ابزارهایی که می‌توان با کمک آنها داده‌های این مخازن داده‌ها را تحلیل نمود، شرح خواهیم داد.

روش‌های نرمال‌سازی معمولاً برای مواردی که محاسبه‌ی فواصل بین نمونه‌ها در داده کاوی مطرح است، مفیدند. در این روش مقادیر اندازه‌گیری شده به محدوده‌ی جدیدی نگاشت می‌شوند. این محدوده‌ی جدید با توجه به الگوریتم‌های موجود بیشتر بین ۰ تا ۱ یا بین -۱ تا ۱ انتخاب می‌شود. سه روش مرسوم برای نرمال‌سازی در ادامه توضیح داده شده‌است.

- روش اول بوسیله‌ی حرکت نقطه‌ی اعشار در داده‌ها، محدوده‌ی جدید را درست می‌کند. برای تبدیل مقدار یک داده در این روش فقط کافی است آن را برعددی که توانی از ده است، تقسیم کنیم. این عدد با توجه به حداقل قدر مطلق مقادیر موجود تعیین می‌شود. بطور مثال فرض کنید

¹ Aggregation Functions

² Normalization

محدوده‌ی مقادیر موجود بین ۳۷۶ تا ۸۰۰ باشد. بزرگترین قدر مطلق در این مجموعه عدد ۸۰۰ است. برای اینکه این اعداد در محدوده‌ی ۱ تا ۱ قرار گیرند، فقط کافی است آنرا بر ۱۰۰۰ (ده به توان سه) تقسیم کنیم. همین کار را با مابقی مقادیر تکرار می‌کنیم.

- با داشتن حداکثر و حداقل مقادیر موجود می‌توانیم با فرمول زیر آنها را به هر محدوده‌ی جدید دلخواهی نرمال‌سازی کنیم.

$$v = \frac{v - Min}{Max - Min} (NewMax - NewMin) + NewMin$$

که در آن Max و Min به ترتیب مقدار حداقل و حداکثر موجود برای صفت خاصه‌ی مورد نظر و $NewMax$ و $NewMin$ محدوده‌ی جدید است. این روش یک تغییر شکل خطی بر روی داده‌های اولیه را اجرا می‌کند.

- نرمال‌سازی نوع سوم برای موقعی مناسب است، که در نمونه‌های صفت خاصه مقدار خارج از محدوده وجود داشته باشد و همینطور مقدار حداکثر و حداقل داده‌ها مشخص نیستند. این روش از میانگین و انحراف استاندارد صفات خاصه استفاده می‌کند. برای تبدیل مقدار V_{old} به مقدار V_{new} از فرمول زیر استفاده می‌کنیم.

$$V_{new} = \frac{V_{old} - Mean}{St_Dev}$$

که در آن $Mean$ میانگین مقادیر صفت خاصه مورد نظر و St_Dev انحراف استاندارد همان مقادیر است.

۴-۵) کاهش داده‌ها

پالایش داده‌ها و تغییر شکل آنها و همچنین تکنیک‌هایی که در مرحله‌ی جمع آوری داده‌ها استفاده می‌شوند، برای داده‌هایی با حجم پایین و حتی متوسط مناسب هستند. استخراج دانش از داده‌هایی با حجم بسیار بالا مستلزم صرف زمان زیادی است. بنابراین منطقی به نظر می‌رسد که ما روش‌هایی را برای کاهش اندازه‌ی داده‌ها به کار ببریم.

شاید با مقدار زیاد داده‌ها نتایج بهتری را بتوانیم بدست آوریم، ولی نمی‌توان به جرأت گفت داده‌های کم، دارای بار اطلاعاتی کمی هستند. تکنیک‌های کاهش داده‌ها می‌توانند بدون از دست دادن درستی داده‌ها و بدون به مخاطره اندختن نتایج نهایی داده‌کاوی وارد عمل شوند. کاوش بر روی داده‌های کمتر هم سریع‌تر و هم کارآثر است. بطور حتم با کاهش داده‌ها در مراحل مختلف داده‌کاوی، سادگی ارائه و نمایش داده‌ها را نیز به همراه خواهیم داشت، به نحوی که مدل قابل فهم‌تر خواهد بود.

تسريع در فرایند داده‌کاوی، افزایش دقت و کارایی و ساده‌سازی داده‌ها از اهداف کلی روش‌های کاهش داده‌ها تلقی می‌شوند. اما دستیابی همزمان به تمام این اهداف شدنی نیست. لذا نگهداری این اهداف در سطح مطلوب و برقراری توازن بین آنها روشی مناسب و امکان‌پذیر به نظر می‌رسد. تصمیم‌گیری در مورد روش انتخابی کاهش داده‌ها به دانشی که در مورد برنامه‌ی کاربردی داریم، برمی‌گردد و روشن است که یک روش و تکنیک شناخته شده‌ی بهینه برای کلیه‌ی کاربردها وجود ندارد.

در تکنیک‌های کاهش داده‌ها عمل کاهش می‌تواند در سه سطح انجام شود. کاهش صفات‌خاصه^(ستون)^۱، کاهش نمونه‌ها(سطر)^۲ و کاهش در مقادیر یک صفت‌خاصه^۳. الگوریتم‌هایی که از هر سه عمل فوق پشتیبانی می‌کنند، ساده نیستند، به خصوص هنگامی که باید بر روی حجم زیادی از داده‌ها اجرا شوند. به کار بردن حداقل تلاش همراه با حداقل کارایی در انتخاب داده‌ها برای اجرای فرایند داده‌کاوی از اهداف مهم تکنیک‌های کاهش داده‌هاست. همان هدفی که مرحله‌ی آماده‌سازی داده‌ها دنبال می‌کند.

۱-۴-۵) کاهش صفات‌خاصه

تکنیک‌های کاهش صفات‌خاصه و یا به عبارت دیگر انتخاب صفات‌خاصه^۴ مناسب بر اساس دانش برنامه‌ی کاربردی و تعریف یک معیار، می‌توانند زیرمجموعه‌ای مناسب از صفات‌خاصه‌ی اولیه را انتخاب کنند.

¹ Columns Reduction

² Rows Reduction

³ Value Reduction

⁴ Attribute Selection

چنانچه داده‌های اولیه حاوی تعداد صفات خاصه‌ی کمی باشد، ممکن است کلیه‌ی زیرمجموعه‌های صفات خاصه را جهت یافتن صفات خاصه بهینه بررسی کنیم. در این حالت که تعداد صفات خاصه کم است، با تعریف پارامتر یا تابعی برای ارزیابی، کلیه‌ی حالات می‌توانند کنترل شوند. پارامترهای ارزیابی بر اساس سازگاری نمونه‌ها، محتویات آنها، فاصله‌ی بین نمونه‌ها و یا وابستگی‌های آماری تعریف می‌شوند.

مجموعه داده‌ها می‌تواند شامل صدھا یا بیشتر صفت خاصه باشد، که خیلی از آنها نامربوط یا افزونه هستند. به طور مثال در پایگاه داده‌ی دانشگاه برای ارزیابی سطح دانش دانشجویان کلیه‌ی نمرات درس‌ها جمع‌آوری می‌شوند. در حالیکه می‌توان با جمع‌آوری نمرات دروس مشترک در رشته‌های مختلف مثل دروس ریاضی، فیزیک، زبان و حتی مبانی کامپیوتر تحلیل خوبی در زمینه‌ی سطح دانش دانشجویان ارائه کرد. توجه داشته باشید، صفات خاصه‌ی نامربوط مثل شماره‌ی دانشجویی و آدرس در مرحله‌ی جمع‌آوری داده‌ها یا در دیگر مراحل کنار گذاشته می‌شوند.

به هر حال انتخاب صفات خاصه مفید می‌تواند مشکل و زمانبر باشد، به خصوص هنگامی که رفتار داده‌ها خوب شناخته شده نیست. نکته‌ی حائز اهمیت اینکه کاهش صفات خاصه مرتبط و نگهداری صفات خاصه نامربوط باعث گیجی الگوریتم‌های داده‌کاوی و منجر به استخراج الگویی ضعیف یا نادرست می‌شود. عمل کاهش ابعاد در شیماهی پایگاه داده با حذف صفات خاصه از آن آغاز می‌شود و مطمئن باشید این کاهش، الگوهای استخراج شده را قابل فهم‌تر می‌سازد.

از یک نقطه نظر می‌توان انتخاب یک زیرمجموعه‌ی مناسب را همانند الگوریتم‌های جستجو فرض کرد. باید از یک مجموعه‌ی n عضوی صفات خاصه زیرمجموعه‌ای مناسب را مشخص کنیم. طبیعی است در صورت بررسی کل حالات باید $2^n - 1$ حالت بررسی شوند. به طور معمول با مقدار n بزرگ فضای جستجو بسیار گسترده است. در حالی که با n کوچک می‌توان همه‌ی فضا را برای انتخاب بهتر جستجو کرد. به هر حال با داشتن تعداد زیاد صفات خاصه، جستجوی جامع در این فضا برای پیدا کردن زیرمجموعه‌ی مناسب هزینه‌بر خواهد بود و شاید امکان‌پذیر نباشد. استفاده از الگوریتم‌های حریصانه و الگوریتم‌های ژنتیک می‌تواند به ما کمک کند.

می‌توان با مجموعه‌ی کاملی از صفات خاصه شروع و با اجرای یک الگوریتم در هر مرحله صفات خاصه‌ای را حذف کنیم تا به مجموعه‌ی بهینه برسیم. یا اینکه با مجموعه‌ی تهی شروع و با اجرای الگوریتم در هر مرحله صفات خاصه‌ی مناسبی را به این مجموعه اضافه کنیم تا به مجموعه جواب نزدیک شویم. به عبارت دیگر روش می‌تواند از بالا به پایین یا بالعکس، از پایین به بالا عمل کند. تقریباً اکثر الگوریتم‌های موجود برای بررسی و انتخاب صفات خاصه‌ی مناسب به وسیله‌ی یکی از این دو روش عمل می‌کنند.

گاهی الگوریتم‌های طبقه‌بندی^۱ نیز می‌تواند در کاهش ابعاد استفاده شوند. الگوریتم‌های درخت‌های تصمیم^۲ مثل *ID3* و *C4.5* بطور طبیعی نامزد خوبی برای این کار هستند. این درخت‌ها ساختاری همانند فلوچارت‌ها دارند، که هر گرهی غیربرگ شرطی بر روی یک صفت خاصه و هر شاخه با توجه به نتیجه‌ی شرط به گرهی بعدی اشاره می‌کند. در گره‌های برگ یا گره‌های پایانی، برچسب کلاسی مشخص می‌شود و در گره‌های میانی بهترین صفات خاصه به منظور طبقه‌بندی داده‌ها انتخاب می‌شوند. هر درخت تصمیم معمولاً شامل تمام صفات خاصه نیست و صفات خاصه‌ی ظاهر شده در درخت حداقل صفات خاصه‌ای هستند که می‌توان برای تجزیه و تحلیل از آنها استفاده نمود. بنابراین درخت‌های تصمیم می‌توانند به عنوان روشی برای کاهش ابعاد داده‌ها استفاده شوند. الگوریتم‌های دیگر طبقه‌بندی نیز به همین طریق می‌توانند مفید باشند. البته فراموش نکنید داده‌های ورودی این روش‌ها دارای برچسب کلاسی برای هر نمونه است. اطلاعات بیشتر در مورد الگوریتم‌های طبقه‌بندی را می‌توانید در فصل‌های ششم و هفتم جستجو کنید.

روش‌های کاهش ابعاد داده به دو دسته‌ی روش‌های مبتنی بر استخراج ویژگی و همچنین روش‌های مبتنی بر انتخاب صفت خاصه تقسیم می‌شوند. در روش‌های اول یک فضای چند بعدی به یک فضایی با ابعاد کمتر نگاشت می‌شود. در واقع با ترکیب صفات خاصه‌ی موجود، تعداد کمتری ویژگی بوجود می‌آورند، بطوریکه این ویژگی‌ها دارای تمام یا بخش اعظمی از اطلاعات موجود در صفات خاصه‌ی اولیه باشند. این روش‌ها به دو دسته‌ی خطی

¹ Classification Algorithms

² Decision Tree

و غیرخطی تقسیم می‌شوند و تکنیک *PCA* که در ادامه به توضیح آن می‌پردازیم، بهترین روش برای کاهش ابعاد داده‌ها به صورت خطی است. در روش‌های مبتنی بر انتخاب صفات خاصه سعی و تلاش می‌شود با انتخاب زیرمجموعه‌ای از ویژگی‌های اولیه، ابعاد داده‌ها کاهش داده شود.

روش‌های بسیار زیادی وجود دارند که می‌توانند جهت کاهش ابعاد داده‌ها، صفات خاصه‌ی مناسب را انتخاب کنند. در این بخش فقط برای فهم بهتر چند گونه از آنها توضیح داده می‌شود.

بررسی صفات خاصه بصورت مجزا برای کاهش ابعاد

یکی از تکنیک‌های موجود برای انتخاب صفات خاصه‌ی مناسب، استفاده از میانگین و واریانس داده‌های است. چنانچه توزیع داده‌ها نرمال باشد، این تکنیک به خوبی جواب‌گو است. هنگام استفاده از این تکنیک‌ها داده‌ها دارای برچسب کلاس هستند. به عبارت دیگر نمونه‌ها دسته‌بندی شده‌اند. مقایسه کردن مقادیر میانگین صفات خاصه که با مقدار واریانس نرمال‌سازی شده‌اند، یکی از تکنیک‌های رایج آماری مورد استفاده است. محاسبه‌ی همبستگی میان داده‌ها نیز از دیگر تکنیک‌های شناخته شده‌ی آماری در این زمینه است.

داده‌های جدول ۲-۱ و صفات خاصه‌ی *X* و *Y* را برای انتخاب یک زیرمجموعه‌ی مناسب در نظر بگیرید. در این جدول کلاس داده‌ها نیز با نام *Class* مشخص شده است.

جدول ۲-۱: داده‌های آزمایشی با دو صفت خاصه و یک برچسب کلاس

X	Y	Class
3	7	A
2	9	B
6	6	A
5	5	A
8	7	B
4	9	A

از عبارت زیر می‌توان جهت انتخاب صفات خاصه‌ی مناسب هنگامی که داده‌ها در دو کلاس A و B قرار دارند (مانند جدول ۱-۱)، استفاده کرد. مقدار آستانه‌ی $TherVal$ توسط کاربر تعیین می‌شود.

$$\frac{|Mean(A) - Mean(B)|}{\sqrt{\frac{Var(A)}{N1} + \frac{Var(B)}{N2}}} > TherVal$$

که در آن Var و $Mean$ به ترتیب توابعی هستند که مقدار میانگین و واریانس را برای کلاس‌های مربوطه محاسبه می‌کنند. تعداد نمونه‌ها نیز در هر کلاس با $N1$ و $N2$ نمایش داده شده است. این عبارت برای هر یک از صفات خاصه محاسبه می‌شود. سپس کاربر با تعیین یک حد آستانه می‌تواند صفات خاصه مناسب را مشخص سازد. اجازه دهید برای فهم بهتر عبارت فوق را برای داده‌های جدول ۱-۱ محاسبه کنیم. داده‌ها در دو کلاس A و B قرار دارند که سهم کلاس A چهار نمونه از کل داده‌ها و سهم کلاس B برابر با دو نمونه است.

$$\begin{array}{ll} X_A = \{3, 6, 5, 4\} & X_B = \{2, 8\} \\ Y_A = \{7, 6, 5, 9\} & Y_B = \{9, 7\} \end{array}$$

مقدار میانگین و واریانس برای داده‌ها به صورت مجزا برای هر کلاس برابر است با:

$$\begin{array}{llll} Mean(X_A) = 4.50 & Mean(X_B) = 5.00 & Mean(Y_A) = 6.75 & Mean(Y_B) = 8.00 \\ Var(X_A) = 1.25 & Var(X_B) = 9.00 & Var(Y_A) = 2.20 & Var(Y_B) = 1.00 \end{array}$$

پس از جایگذاری مقادیر در عبارت نتایج بدست می‌آیند.

$$S_x = \sqrt{(Var(X_A)/N1 + Var(X_B)/N2)} = \sqrt{(1.25/4 + 9/2)} = 2.1937$$

$$S_y = \sqrt{(Var(Y_A)/N1 + Var(Y_B)/N2)} = \sqrt{(2.20/4 + 1/2)} = 1.0247$$

$$|Mean(X_A) - Mean(X_B)| / S_x = |4.5 - 5| / 2.1937 = 0.2279$$

$$|Mean(Y_A) - Mean(Y_B)| / S_y = |6.75 - 8| / 1.0247 = 1.2198$$

با توجه به مقدار در نظر گرفته شده برای آستانه، صفت خاصه‌ی کاندید برای حذف از مجموعه را می‌توان بدست آورد. به طور مثال با تنظیم مقداری برابر با 0.5 برای آستانه،

صفت خاصه‌ی X را می‌توان از مجموعه‌ی انتخابی حذف کرد و دیگر صفت خاصه‌ی Y می‌تواند برای تشخیص کلاس‌های متفاوت از داده‌ها در مجموعه‌ی حداقل باقی بماند. مشخص کردن مقدار سطح آستانه یکی از مسائلی است که در این روش و روش‌های مشابه ما را با مشکل رو برو می‌کند.

در این روش صفات خاصه به صورت مجزا مورد آزمایش قرار می‌گیرند. روش‌های دیگری وجود دارند که در آنها کلیه‌ی صفات خاصه با یکدیگر در انتخاب موثرند. بدین ترتیب احتمال وجود افزونگی در داده‌ها کاهش می‌یابد. استفاده از ماتریس کوواریانس می‌تواند مفید باشد. مشکل اصلی هنگامی است که حجم داده‌ها بسیار زیاد است. بنابراین فضای جستجو نیز به صورت توانی رشد می‌کند و در صورت استفاده از این گونه الگوریتم‌ها، باید از قوانین اکشافی^۱ یا هیوریستیک مناسبی بهره برد تا سرعت اجرا بصورت قابل ملاحظه‌ای کاهش یابد.

محاسبه‌ی آنتروپی^۲ برای رتبه‌بندی صفات خاصه

ایده‌ی اصلی در این روش بدین صورت است که ما صفت خاصه‌ای را حذف می‌کنیم که حذف آن لطمehای به مشخصات کلی داده‌ها وارد نمی‌سازد. به عبارتی دیگر با نبود آن باز هم می‌توانیم بین نمونه‌ها تمایز قابل شویم و در درجه‌ی نظم داده‌ها تغییرات محسوسی حاصل نمی‌شود.

برای این کار الگوریتم از فرمولی جهت یافتن شباهت میان نمونه‌ها استفاده می‌کند. در فصل مربوط به خوشه‌بندی فرمول‌های زیادی جهت محاسبه‌ی شباهت ارائه شده است. یکی از آنها و طریقه‌ی محاسبه‌ی آن در زیر نشان داده شده است.

$$S_{ij} = e^{-\alpha D_{ij}}$$

$$D_{ij} = \left[\sum_{k=1}^n ((x_{ik} - x_{jk}) / (Max_k - Min_k))^2 \right]^{1/2}$$

¹ Heuristic

² Entropy Measure

که در آن S_{ij} و D_{ij} به ترتیب معرف تشابه و فاصله میان نمونه‌های i ام و j ام هستند. مقدار n نیز تعداد صفات خاصه و متغیرهای Min_k و Max_k به ترتیب نشان‌دهنده بیشترین و کمترین مقدار صفت خاصه k ام است. مقدار k امین صفت خاصه از n نمونه در فرمول با x_{ik} نشان داده شده است. پارامتر α را می‌توان از فرمول‌های متفاوتی بدست آورد. اما تجربه در برنامه‌های کاربردی مقدار $5/0$ را برای α انتخاب خوبی می‌داند. چون ماهیت کلیه‌ی داده‌ها عددی نیست، برای محاسبه‌ی شباهت میان داده‌های غیر عددی می‌توانیم از فرمول زیر استفاده کنیم.

$$S_{ij} = \left(\sum_{k=1}^n |x_{ik} - x_{jk}| \right) / n$$

مقایسه‌ی داخل قدر مطلق در صورت مساوی بودن مقادیر دو صفت خاصه برابر با یک و در غیراینصورت صفر لحاظ می‌گردد. در فرمول تعداد کل صفات خاصه با n نشان داده شده است.

حال نوبت به معیار سنجش آنتروپی می‌رسد که برای مجموعه داده‌ای با تعداد N نمونه به صورت زیر بیان می‌شود:

$$Entropy = - \sum_{i=1}^{N-1} \sum_{j=i+1}^N [S_{ij} \times \text{Log} S_{ij} + (1 - S_{ij}) \times \text{Log} (1 - S_{ij})]$$

الگوریتم را می‌توان در چند مرحله به قرار زیر شرح داد:

- با مجموعه‌ی کاملی از صفات خاصه شروع می‌کنیم. این مجموعه را Set می‌نامیم.
- آنتروپی را برای Set محاسبه می‌کنیم.
- با حذف یک صفت خاصه از Set آنتروپی مجموعه‌ی کاهش یافته را محاسبه و این کار را برای کلیه‌ی صفات خاصه تکرار می‌کنیم.
- فاصله‌ی آنتروپی مجموعه‌های جدید در مرحله‌ی قبل را با آنتروپی Set بدست آورده و از بین آنها کمترین را انتخاب می‌کنیم.
- مجموعه‌ای که کمترین فاصله‌ی آنتروپی با Set را دارد، مجموعه کاهش یافته‌ی جدید است. مراحل قبل را برای این مجموعه جدید تا محقق شدن یک شرط

پایانی ادامه می‌دهیم. پس از آن میان مجموعه صفات خاصه، ترتیبی از اهمیت آنها به دست آورده‌ایم.

این روش یک تکنیک بالا به پایین تلقی می‌شود. بدین ترتیب که با مجموعه‌ی کاملی از صفات خاصه شروع می‌کند. سپس در هر مرحله با حذف یک صفت خاصه به ارزیابی نتیجه می‌پردازد تا به یک زیرمجموعه‌ی مناسب از صفات خاصه دست یابد. از معایب این الگوریتم می‌توان به پیچیدگی محاسباتی الگوریتم اشاره کرد که با پیاده‌سازی موازی آن، این مسئله تا حدودی بهبود می‌یابد.

تحلیل مولفه‌های اصلی^۱ (PCA)

یکی از عمومی‌ترین روش‌های آماری به منظور کاهش ابعاد داده‌ها روش تحلیل مولفه‌های اصلی است. در این روش واریانس کل صفات خاصه موجود تحلیل می‌شود. مولفه‌ها طوری برآورد می‌گردند تا واریانس صفات خاصه را در کمترین ابعاد نشان دهند. در واقع مولفه‌های اصلی مجموع موزون صفات خاصه است. این روش می‌تواند در برنامه‌های کاربردی با زمینه‌های تشخیص الگو و فشرده‌سازی تصاویر مفید باشد. همچنین این روش تکنیکی برای یافتن الگویی در داده‌ها با ابعاد بالا محسوب می‌شود.

این روش از مقادیر ویژه^۲ و بردارهای ویژه^۳ ماتریس کوواریانس^۴ داده‌ها استفاده می‌کند. ماتریس کوواریانس و فرمول محاسبه‌ی آن در بخش‌های قبلی توضیح داده شده است. ماتریس کوواریانس یک ماتریس متقارن است و عناصر قطر اصلی این ماتریس مقادیر واریانس‌های صفات خاصه هستند. بنابراین با فرض اینکه ابعاد داده‌های ما n باشد ما به یک ماتریس $n \times n$ که ماتریسی مربع است، دست خواهیم یافت.

مرحله‌ی بعدی بدست آوردن بردارها و مقادیر ویژه برای این ماتریس است. معادله‌ی زیر برای بدست آوردن آنها استفاده می‌شود.

$$C \times E_{vector} = \lambda \times E_{vector}$$

¹ Principal Component Analysis

² Eigenvalues

³ Eigenvectors

⁴ Covariance Matrix

در فرمول C ماتریس کوواریانس، E_{vector} بردار ویژه و λ مقدار ویژه هستند. باید بدانیم بردارهای ویژه برای ماتریس‌های مریع وجود دارند ولی هر ماتریس مربعی دارای بردار ویژه نیست. نکته‌ی دیگر اینکه در صورت وجود بردار ویژه تعداد آن برابر با ابعاد ماتریس است. بنابراین یک ماتریس $n \times n$ دارای n بردار ویژه خواهد بود. نکته‌ی آخر اینکه بردارهای ویژه متعامدند^۱. این نکته موضوع مهمی است، زیرا می‌توانیم داده‌ها را با این بردارهای ویژه توصیف کنیم، به مثال زیر توجه کنید.

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 3 \\ 2 \end{bmatrix} = 4 \times \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

در این مثال برای یک ماتریس نمونه 2×2 یک مقدار ویژه 4 و یک بردار ویژه نشان داده شده است.

پس از محاسبه‌ی ماتریس کوواریانس برای داده‌ها، مقادیر و بردارهای ویژه را می‌توان محاسبه نمود. سپس مقادیر ویژه را به صورت نزولی مرتب می‌کنیم. در حالتی که ما دارای n صفت خاصه هستیم، این ترتیب به صورت زیر نشان داده می‌شود.

$$\lambda_1 > \lambda_2 > \lambda_3 > \dots > \lambda_{n-1} > \lambda_n$$

بالاخره می‌خواهیم با انتخاب k ابعاد را از n به k کاهش دهیم. انتخاب k با کمک یک سطح آستانه‌ای که می‌تواند توسط کاربر مشخص شود، تعیین شود. اگر مقدار $95/0$ برای حد آستانه مشخص شود، مقدار k را می‌توانیم از عبارت زیر بدست آوریم.

$$\frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^n \lambda_i} \geq 0.95$$

این بدین معنی است که با ابعاد k ما 95 درصد از خصوصیات داده‌های اصلی را حفظ کرده‌ایم. مقدار k کوچکترین مقداری است که عبارت نامساوی مذبور را برقرار می‌کند. مرحله‌ی پایانی در الگوریتم بدست آوردن داده‌های جدید است. این داده‌ها از فرمول زیر بدست می‌آیند.

$$Data_{new} = Vector^T \times Data_{original}^T$$

^۱ Orthogonal

که در آن *Vector* ماتریس بردارهای ویژه است که در عبارت فوق از ترانهاده آن استفاده شده است. در این ماتریس مهمترین بردارهای ویژه در سطرهای بالایی قرار دارند. مهمترین یعنی بردارهای ویژه‌ای که دارای بزرگترین مقدار ویژه هستند. ماتریس $Data_{original}^T$ نیز داده‌های اولیه‌ی ما را نشان می‌دهد. در این دهنه‌ی ابعاد است و و ستون‌ها نمونه‌ها را نشان می‌دهند. فراموش نکنید که در این روش بردارهای ویژه‌ای که دارای مقادیر ویژه بزرگتری هستند، نگهداری می‌شوند.

۲-۴-۵) کاهش نمونه‌ها

بدون شک تعداد نمونه‌ها بیشترین مقدار داده‌ای (نسبت به تعداد صفات خاصه) هستند که ما در حجم بسیار داده‌ها با آن روبرو هستیم و باید با استفاده از تکنیک‌های کاهش داده‌ها مقدار آن را در حد مطلوبی قرار دهیم. کاهش نمونه‌ها یکی از پیچیده‌ترین وظایف در روش‌های کاهش داده‌ها محسوب می‌شود. اما با این وجود راههایی هستند که می‌توانیم با کمک آنها حجم نمونه‌ها را کم کنیم. یکی از ساده‌ترین و معمولی‌ترین آنها استفاده از روش‌های نمونه‌گیری^۱ است.

تکنیک‌های نمونه‌گیری

اگر بخواهیم از نمونه‌گیری نتیجه‌ی رضایت‌بخشی بدست آوریم، باید به مراحلی که نمونه‌ها انتخاب می‌شوند آگاهی داشته باشیم. عمده‌ترین موضوع نمونه‌گیری انتخاب واقعی نمونه است. جلوگیری از جهت‌گیری در فرایند انتخاب و افزایش دقت در تمام مراحل، ما را به انتخاب واقعی نزدیک می‌کند. اندازه‌ی نمونه یعنی تعداد نمونه‌های انتخابی یکی از پارامترهای مهم در روش‌های نمونه‌گیری است که متناسبانه راهکار قاطعی برای یافتن آن وجود ندارد. اما در برخی از کتاب‌ها حداقل حجم نمونه‌ها را با توجه به کاربردهای مختلف پیشنهاد می‌کنند. در صورتیکه حجم نمونه بزرگ باشد، شباهت نمونه‌ها با داده‌های اصلی افزایش خواهد یافت، اما کاهش چشمگیری در حجم داده‌ها

¹ Sampling

دیده نمی‌شود. از طرفی دیگر با تعداد کم نمونه‌ها برخی از مشخصات داده‌های اولیه در نمونه‌ها وجود نخواهد داشت و نتیجه‌ی داده‌کاوی بر روی داده‌ها آن چیزی نیست که به دنبال آن هستیم. در ادامه به ذکر برخی از روش‌های نمونه‌گیری می‌پردازیم.

نمونه‌گیری تصادفی از ساده‌ترین تکنیک‌های نمونه‌گیری به شمار می‌رود که در دو صورت باجایگزینی^۱ و بدون جایگزینی^۲ انجام می‌شود. در این گونه روش‌ها هر یک از نمونه‌ها دارای شانسی مساوی و مستقل برای انتخاب هستند. در نوع بدون جایگزینی هر نمونه فقط یک بار شанс انتخاب شدن را دارد، درصورتی که در نمونه‌گیری تصادفی باجایگزینی هر نمونه بیش از یک بار می‌تواند انتخاب شود. اغلب روش‌های آماری از نوع دوم یعنی باجایگذاری استفاده می‌کنند.

نمونه‌گیری منظم نوع دیگری از نمونه‌گیری است که در آن تمام نمونه‌ها بدون هرگونه ترتیبی قرار گرفته و سپس با یک نظم معین نمونه مورد نظر انتخاب می‌شود. برای مثال مشخصات دانشجویان را بدون هیچگونه ترتیبی قرار می‌دهیم و سپس برای نمونه‌گیری از هر ۱۰ دانشجو یکی را انتخاب می‌کنیم. بدین ترتیب پس از نمونه‌گیری اول ۱۰ دانشجوی اول در انتخاب بعدی شانسی برای گزینش ندارند.

نوع دیگر نمونه‌گیری طبقه‌ای است. از این روش هنگامی استفاده می‌شود که مایلیم نمونه‌هایی از هر طبقه در نتیجه‌ی نهایی وجود داشته باشند. برای مثال فرض کنید مایلیم از میان دانشجویان یک دانشگاه افرادی را انتخاب کنیم. از آنجا که می‌خواهیم تا در نتیجه‌ی نمونه‌گیری ما هم خانم‌ها و هم آقایان شرکت داشته باشند، می‌توان نمونه‌گیری طبقه‌ای را برای دختران و پسران بصورت مستقل انجام داد. این روش تضادی با اصل تصادفی بودن ندارد. چرا که ابتدا نمونه‌ها به طبقه‌های مختلف تقسیم و سپس هر نمونه به صورت تصادفی از طبقه‌ها برگزیده می‌شود.

نمونه‌گیری خوش‌های نوع دیگری از روش‌های نمونه‌گیری است که در آن به جای نمونه، گروه‌ها (خوش‌های) به صورت تصادفی انتخاب می‌شوند. این روش هنگامی مناسب است که انتخاب نمونه مشکل یا غیرممکن باشد. اگر عمل نمونه‌گیری با استفاده از خوش‌های

¹ With Replacment

² Without Replacment

یک بار انجام شود، به آن نمونه‌گیری خوش‌های تک مرحله‌ای گفته می‌شود. چنانچه این عمل بیش از یک بار صورت پذیرد به نمونه‌گیری خوش‌های چندمرحله‌ای موسوم است. در این روش می‌توانید پس از نمونه‌گیری خوش‌ها، نمونه‌ها را از خوش‌های انتخاب شده گزینش کنید.

۳-۴-۵) کاهش مقادیر یک صفت خاصه

مقادیر موجود برای یک صفت خاصه محدوده‌ای را تشکیل می‌دهند. بدون شک این مقادیر زیرمجموعه‌ای از دامنه‌ی صفت خاصه‌ی مذبور است. برای مثال صفت خاصه سن را در نظر بگیرید. فرض کنید دامنه‌ی آن از ۱۵۰ تا ۱ باشد و مقادیر موجود در داده‌ها از ۱۰ شروع و به ۱۳۰ ختم شود. در واقع حداقل تعداد اعداد متمایز موجود در پایگاه داده برای صفت سن حدود ۱۲۱ عدد خواهد بود. اغلب هدف اصلی تکنیک‌های کاهش مقادیر صفات خاصه، کاهش این تعداد است. در این مثال شما می‌توانید با تعیین محدوده‌های خاص این ۱۲۱ عدد متمایز را در ۵ گروه سنی کودک، نوجوان، جوان، میانسال و پیر قرار دهید.

تکنیک‌های مجزاسازی^۱ یا گسسته‌سازی می‌توانند برای کاهش مقادیر در پایگاه داده استفاده شوند. وظیفه‌ی این تکنیک‌ها قرار دادن داده‌های پیوسته در تعدادی فاصله است. این تکنیک‌ها عموماً در توصیف ساده‌تر داده‌ها و فهم راحت‌تر و در نهایت نتایج پایانی فرایند داده‌کاوی تأثیر بسزایی دارند. برخی از این تکنیک‌ها می‌توانند توسط انسان بصورت دستی یا در تعریف اولیه‌ی شمای، پایگاه داده‌ها تعریف شوند. همانند مثال قبلی که این کار در مورد صفت خاصه‌ی سن انجام شد.

تکنیک‌های دسته‌بندی^۲ که در بخش‌های قبلی توضیح داده شدند، نیز روش‌های مناسبی هستند. روش آماری کای مربع (χ^2) یکی از روش‌های پرکاربرد است که می‌توان از آن برای گسسته‌سازی نیز استفاده کرد.

¹ Discretization Methods

² Binning

روش آماری کای دو^۱

تکنیک کای دو یک الگوریتم خودکار گسسته‌سازی است که با استفاده از آماره‌ی χ^2 کارش را انجام می‌دهد. این روش با کمک کلاس‌های نمونه‌ها، شباهت‌های بین توزیع داده‌ها در دو بازه‌ی^۲ همسایه را تعیین می‌کند. چنانچه نتیجه‌ی آزمون χ^2 این باشد که برچسب کلاس مستقل از فواصل تعريف شده در بازه‌ها است، آنگاه بازه‌ها ادغام خواهند شد. در غیر این صورت تفاوت میان بازه‌ها از نظر آماری قابل توجه بوده و ادغامی صورت نمی‌گیرد.

این تکنیک دارای سه مرحله‌ی اساسی برای گسسته‌سازی است. ابتدا مقادیر صفت‌خاصه مذبور به صورت صعودی مرتب می‌شوند. پس از آن هریک از مقادیر داده‌ها را به عنوان یک بازه‌(فاسله) در نظر می‌گیریم. در مرحله‌ی سوم با محاسبه‌ی χ^2 برای دو بازه‌ی مجاور و مقایسه‌ی آن با یک حد آستانه تصمیم برای ادغام گرفته می‌شود. این مرحله تا هنگامی تکرار خواهد شد که هیچ دو بازه‌ی مجاوری قابلیت ادغام نداشته باشند. به عبارت دیگر پس از هر ادغام آزمون χ^2 را برای بازه‌های باقیمانده محاسبه می‌کنیم. برای χ^2 ‌هایی که مقدار آنها کمتر از حد آستانه است، بازه‌ها ادغام می‌شوند. فرمول زیر مقدار کای دو برای دو بازه‌ی مجاور محاسبه می‌کند:

$$\chi^2 = \sum_{i=1}^{k-1} \sum_{j=1}^{k-i} |A_{ij} - E_{ij}|^2 / E_{ij}$$

که در آن k تعداد کلاس‌ها را بیان می‌کند و A_{ij} تعداد نمونه‌ها در i امین بازه و j امین کلاس است.

فرآوانی موردنظر برای E_{ij} با A_{ij} نشان داده می‌شود و از فرمول زیر می‌توان بدست آورد:

$$E_{ij} = (R_i \times C_j) / N$$

که در آن R_i و C_j به ترتیب تعداد نمونه‌ها در i امین بازه و تعداد نمونه‌ها در کلاس j است. تعداد کل نمونه‌ها نیز با N نشان داده شده است. اگر مقدار R_i یا C_j صفر باشند، فرمول زیر مقدار کوچک مثل $1/0$ مقداردهی می‌شود.

¹ ChiMerge

² Interval

برای مجموعه داده‌هایی با دو کلاس، شکل ۲-۳ نشان دهندهٔ متغیرهای لازم جهت محاسبه χ^2 است.

	<i>Class1</i>	<i>Class2</i>	
<i>Interval1</i>	A_{11}	A_{12}	R_1
<i>Interval2</i>	A_{21}	A_{22}	R_2
	C_1	C_2	N

شکل ۲-۳: داده‌های لازم جهت محاسبه χ^2 برای دو کلاس و دو بازه

اجازه دهید برای فهم بهتر، روش را با یک مثال ساده دنبال کنیم. جدول ۲-۲ تعداد ۱۲ نمونه را همراه با مقادیر صفت‌خاصه‌ی پیوستهٔ *Att* و کلاس هر نمونه نشان می‌دهد. بدون از دست دادن کلیات مسئله، مقادیر برای این صفت‌خاصه مرتب شده هستند. بازه‌های اولیه به گونه‌ای تعریف می‌شوند که یک نمونه از داده‌ها در هر بازه قرار گیرد.

$$[0,2) \quad [2,5) \quad [5,7.5) \quad [7.5,8.5) \quad [8.5,10) \quad [10,17) \quad [17,30) \dots$$

همانطور که می‌بینید عدد میان دو مقدار به عنوان مرز فواصل تعریف شده است. حال برای دو بازه‌ی مجاور مقدار χ^2 محاسبه می‌شود. کوچکترین مقدار محاسبه شده را با حد آستانه تعریف شده توسط کاربر مقایسه می‌کنیم. چنانچه این مقدار کمتر از حد آستانه باشد، دو بازه با یکدیگر ادغام می‌شوند.

جدول ۲-۲: داده‌های آزمایشی با دو صفت‌خاصه و یک برچسب کلاس

ID	Att	Class
1	1	A
2	3	B
3	7	A
4	8	A
5	9	A
6	11	B
7	23	B
8	37	A
9	39	B
10	45	A
11	46	A
12	59	A

جدول شکل ۲-۴ و محتوای آن برای ادغام دو بازه‌ی $(7.5, 8.5)$ و $(8.5, 10)$ تشکیل شده است.

	Class A	Class B	
$[7.5, 8.5)$	$A_{11}=1$	$A_{12}=0$	$R_1=1$
$[8.5, 10)$	$A_{21}=1$	$A_{22}=0$	$R_2=1$
	$C_1=2$	$C_2=0$	$N=2$

شکل ۲-۴: مقادیر بدست آمده جهت بررسی ادغام دو بازه

با توجه به مقادیر موجود در جدول داریم:

$$E_{11}=2/2=1$$

$$E_{12}=0/2 \approx 0.1$$

$$E_{21}=2/2=1$$

$$E_{22}=0/2 \approx 0.1$$

$$\chi^2 = (1-1)^2/1 + (0-0.1)^2/0.1 + (1-1)^2/1 + (0-0.1)^2/0.1 = 0.2 < 2.706$$

درجه‌ی آزادی آزمون χ^2 برای یک مجموعه داده یکی کمتر از تعداد کلاس‌ها در نظر گرفته می‌شود. چون در مثال دو کلاس وجود دارد، لذا مقدار χ^2 برای درجه‌ی آزادی یک از جدول توزیع بدست می‌آید. این مقدار برابر با $2/706$ است، که بسیار بزرگتر از مقدار $2/0$ است. می‌توان اینگونه نتیجه گرفت که تفاوت‌های قابل توجهی در فراوانی‌های نسبی دو بازه وجود ندارد و در نتیجه بازه‌های انتخاب شده می‌توانند ادغام شوند. پس از ادغام χ^2 همسایه‌های جدید محاسبه و مقایسه می‌شوند. این کار را ادامه می‌دهیم تا دیگر ادغامی صورت نگیرد. اگر در پایان الگوریتم تعداد بازه‌های نهایی بیشتر از حد تعریف شده توسط کاربر باشد، می‌توان حد آستانه را کاهش و الگوریتم را تکرار نمود.

ما در اینجا یک مرحله‌ی دیگر از آزمون را تحلیل می‌کنیم. مرحله‌ای که نمونه‌های اول و دوم و سوم در یک بازه ادغام شده‌اند و نمونه‌های چهارم و پنجم نیز با یکدیگر در یک بازه قرار گرفته‌اند. می‌خواهیم با محاسبه نشان دهیم آیا این دو بازه مستعد ادغام شدن هستند یا خیر.

شکل ۲-۵ مقادیر لازم برای محاسبه χ^2 را نشان می‌دهد.

	Class A	Class B	
[0,7.5)	$A_{11}=2$	$A_{12}=1$	$R_1=3$
[7.5,10)	$A_{21}=2$	$A_{22}=0$	$R_2=2$
	$C_1=4$	$C_2=1$	$N=5$

شکل ۲-۵: مقادیر بدست آمده جهت بررسی ادغام دو بازه

با توجه به محتوای جدول داریم:

$$E_{11}=12/5=2.4$$

$$E_{12}=3/5=0.6$$

$$E_{21}=8/5=1.6$$

$$E_{22}=2/5=0.4$$

$$\chi^2=(2-2.4)^2/2.4+(1-0.6)^2/0.6+(2-1.6)^2/1.6+(0-0.4)^2/0.4=0.834 < 2.706$$

مشاهده می‌کنید که این دو بازه نیز می‌توانند ادغام شوند. پس از اجرای مکرر الگوریتم سه بازه‌ی زیر به عنوان نتیجه‌ی نهایی پیشنهاد می‌شوند.

$$[0,10) , [10,42) , [42,60]$$

این سه بازه می‌تواند با اعداد یا کلمات کدگذاری شوند. بدین ترتیب ۱۲ مقدار متمایز صفت خاصه‌ی Att به تعداد ۳ مقدار کاهش می‌یابد.

آزمون کای دو یکی از روش‌های آماری غیرپارامتریک است، که دارای کاربردهای فراوانی است. در به کارگیری این آزمون رعایت فرض‌های دشوار، نظیر آنچه با روش‌های پارامتریک به کار برده می‌شود، الزامی نیست. از این آزمون برای تعیین رابطه بین متغیرهای گسسته نیز می‌توان استفاده کرد.

گسسته‌سازی مبتنی بر آنتروپی

آنتروپی یکی از معیارهای رایج جهت گسسته‌سازی به شمار می‌رود. گسسته‌سازی مبتنی بر آنتروپی یک تکنیک جداسازی بالا به پایین است، که بر روی داده‌هایی با برچسب

کلاس انجام می‌شود. این بدین معنی است که روش با کمک اطلاعات توزیع کلاس، نقاط انفصال را برای داده‌ها محاسبه می‌کند. نقاط انفصال آنها بی‌هستند که مقادیر داده‌ها بر مبنای آنها افزایش می‌شوند. برای گسسته‌سازی صفت‌خاصه‌ای مانند A ، روش مزبور مقداری از A را به عنوان نقطه‌ی انفصال انتخاب می‌کند که مقدار آنتروپی را حداقل کند و این کار به صورت بازگشتی جهت یافتن نقاط انفصال دیگر ادامه می‌یابد. چنانچه مجموعه داده‌های D برای هر یک از نمونه‌ها دارای برچسب کلاس باشند، روش گسسته‌سازی صفت‌خاصه‌ی پیوسته‌ی A شامل سه مرحله است.

- می‌دانیم که هر مقدار صفت‌خاصه‌ی A می‌تواند بالقوه یک نقطه‌ی انفصال باشد. پس از مرتب‌سازی مقادیر A به بررسی تک تک حالت‌ها می‌پردازیم. با انتخاب یک نقطه‌ی انفصال SP ، مجموعه داده‌های D به دو زیرمجموعه‌ی D_2 و D_1 شکسته می‌شود که به ترتیب هر یک از این زیرمجموعه‌ها شرط $A \leq SP$ و $A > SP$ را ارضاء می‌کنند.
- برای هر یک از نقاط انفصال تعریف شده در مرحله‌ی قبل فرمول زیر محاسبه می‌شود.

$$Info_A(D) = \frac{|D_1|}{|D|} \times Entropy(D_1) + \frac{|D_2|}{|D|} \times Entropy(D_2)$$

که در آن $|D| / |D_1|$ تعداد نمونه‌ها در مجموعه‌ی D را نشان می‌دهد و برای آنتروپی داریم:

$$Entropy(D_1) = -\sum_{i=1}^k p_i \times \log_2(p_i)$$

که در آن p_i احتمال کلاس C_i در D_1 را نشان می‌دهد.

- در نهایت از میان نقاط انفصال، نقطه‌ای انتخاب می‌شود که مقدار $Info$ محاسبه شده‌ی آن کوچکترین باشد. به عبارت دیگر این نقطه، داده‌ها را با درجه‌ی خلوص بهتری طبقه‌بندی می‌کند، یعنی تنوع مقدار کلاس در هر طبقه به حداقل می‌رسد.

پس از طی مراحل فوق داده‌های اصلی با توجه به نقطه‌ی انفصل به دو گروه تقسیم خواهد شد. چنانچه مایلید تعداد گروه‌های بیشتری داشته باشید، می‌توان این مراحل را برای هر زیرمجموعه تکرار کنید. این کار را تا به دست آوردن تعداد گروه‌های دلخواه ادامه می‌دهیم.

خلاصه فصل

یکی از مراحل مهم و پیچیده در فرایند کشف و استخراج دانش آماده‌سازی داده‌هاست. انتخاب داده‌های مناسب به منظور بدست آوردن کارایی حداکثر و پردازش حداقل از اهداف کلی مرحله‌ی آماده‌سازی داده‌ها به شمار می‌آید. بنابراین با داده‌های درست و کامل و در عین حال کم، الگوریتم‌های داده‌کاوی سریعتر و موثرتر به الگوی مورد نظر می‌رسند و نتایج حاصل از آن قابل فهم تر و راحت‌تر برای کاربر خواهد بود.

آماده‌سازی داده‌ها متکی به ماهیت داده‌ها و نوع تحلیل بر روی آنهاست، بنابراین با شناخت داده‌ها می‌توان روش مناسبی برای آماده‌سازی داده‌ها انتخاب کرد. لازم است مشخصات اصلی داده‌ها با انواع مختلفی از تحلیل‌های آماری شامل جدول‌بندی ساده و جدول‌بندی متقطع^۱ داده‌ها و محاسبه‌ی پارامترهای آماری مهم بدست آید.

اکثر تحقیقات در این چند سال اخیر بر روی الگوریتم‌های داده‌کاوی صورت گرفته و متأسفانه کوشش کمی برای بدست آوردن تئوری‌های قوی برای آماده‌سازی داده‌ها به عمل آمده است. در حالیکه دستاوردهای زیادی که در مرحله‌ی داده‌کاوی بدست می‌آیند، جملگی مدیون مرحله‌ی پیش‌پردازش و آماده‌سازی داده‌ها می‌باشد.

روش‌های آماده‌سازی داده‌ها به جز در این مرحله می‌توانند در کلیه‌ی مراحل فرایند کشف دانش استفاده شوند، نه تنها در الگوریتم‌های داده‌کاوی بلکه پس از پردازش نهایی برای خلاصه‌سازی و ارزیابی نتایج، کاربرد فراوانی دارند. برای مثال قابل فهم بودن الگوهای نهایی یکی دیگر از وظایف آماده‌سازی داده‌ها محسوب می‌شود. استفاده از ابزاری برای

^۱ Cross Tabulation

نمایش گرافیکی، ساختارهای شرطی، زبان‌های طبیعی و تکنیک‌های دیگر می‌توانند در این مرحله به ما کمک کنند.

تکنیک‌های داده‌کاوی به اندازه‌ی کافی پیشرفت داشته‌اند و این با پیشرفت در مرحله‌ی آماده‌سازی داده‌ها مطابقت نمی‌کند و به عبارتی همگام نیست. بنابراین نیاز ما به روش‌های مکانیزه و قابل اعتماد و درست برای آماده‌سازی داده‌ها بیش از پیش احساس می‌شود.

تکنیک‌های مختلفی از جمله پالایش داده‌ها، جمع‌آوری و تغییر شکل داده‌ها و کاهش داده‌ها در این مرحله استفاده می‌شوند که برای هر یک الگوریتم‌های متعددی وجود دارد. معیارهای بسیاری برای انتخاب یک الگوریتم مناسب وجود دارند، اما به جرأت می‌توان گفت که دانستن ماهیت داده‌ها در انتخاب، اهمیت زیادی دارد. آماده‌سازی داده‌ها و همچنین داده‌کاوی یک فرایند کاملاً کاربردگرا^۱ و به برنامه‌ی کاربردی و اهداف آن وابسته هستند. از این رو شناخت داده‌ها و خصوصیات آنها کمک زیادی به بهبود عمل آماده‌سازی می‌کند.

دسته‌بندی و طبقه‌بندی داده‌ها و همچنین بدست آوردن پارامترهای آماری از جمله راهکارهای رایجی هستند که می‌توانند ما را در این فرایند یعنی شناخت داده‌ها و روابط بین آنها یاری کنند. اما تعیین برخی از پارامترهای ورودی (همانند سطح آستانه) توسط تحلیل‌گر در این روش‌ها می‌تواند ما را با مشکل بزرگی روبرو سازد. بنابراین یا باید این مقادیر، بهینه انتخاب شوند و یا باید از روش‌هایی استفاده شود که دارای حداقل پارامتر هستند.

پس از بررسی روش‌های متعدد برای آماده‌سازی داده‌ها(مرحله‌ی آغازین کشف دانش) می‌توانیم آنها را در انواع زیر قرار دهیم.

- الگوریتم‌هایی که در جمع‌آوری داده‌ها استفاده می‌شوند. با دارا بودن هدف کلی در مطالعه، انتخاب مجموعه داده‌های اصلی برای تحلیل اولین ضرورت است. بسیاری از الگوریتم‌های جمع‌آوری داده‌ها فقط با پایگاه داده‌های همگن^۲ کار

¹ Application-Oriented

² Homogeneous

می‌کند که این مسئله نیز در جمع‌آوری داده‌ها محدودیت محسوب می‌شود. بدون شک افزونگی و ناسازگاری دو مشکل عمدہ‌ای هستند که در جمع‌آوری داده‌ها به چشم می‌خورند. پیشنهاد ما تصویب یک استاندارد برای ذخیره و نمایش داده‌های است. این استاندارد می‌تواند برای انتقال داده‌ها نیز به کار گرفته شود. در این صورت در تشخیص افزونگی و ناسازگاری راحت عمل خواهیم کرد. استفاده از داده‌های متأ در تشخیص افزونگی راهکار مناسبی است. توابع تجمعی در این مرحله حجم داده‌ها را کاهش می‌دهند.

- روش‌هایی برای مقابله با داده‌های نادرست و ناقص. نمونه‌های جمع‌آوری شده از انبار داده‌ها یا بانک اطلاعاتی اغلب از آنچه آلدگی داده‌ها نام‌گذاری شده است، رنج می‌برند و بنابراین لازم است پاک‌سازی و پالایش شوند تا از یک دست بودن شکل آنها اطمینان حاصل شود. باید امیدوار باشیم تا حجم کمی از داده‌ها نادرست یا ناقص باشند. چون اکثر روش‌ها به منظور رفع این مشکل، داده‌های نادرست و ناقص را با توجه به مقادیر موجود در پایگاه داده جایگزین می‌کند و چنانچه درصد بالایی از آنها نادرست یا ناقص باشند، تخمین‌ها نیز باعث جهت‌دار شدن الگوریتم‌ها می‌شوند و در نتیجه الگوهای تولیدی نامناسبند. در کاربردهای تجاری داده‌های نادرست و ناقص مشکلات حادتری ایجاد می‌کنند. تعیین مقدار پیش‌فرض برای داده‌هایی که امکان دارد در زمان ورود اطلاعات اجباری نباشند، می‌تواند راه حل مناسبی برای داده‌های ناقص تلقی شود. اما همیشه داده‌های ناقص از عملکرد بد کاربر ناشی نمی‌شود. گروه‌بندی داده‌ها با استفاده از استراتژی‌های پیچیده‌ی آماری جهت معرفی مقداری برای صفات خاصه ناقص و تبیین وابستگی بین آنها نیز راه خوبی به نظر می‌رسد.

- دسته‌بندی و طبقه‌بندی هوشمندانه داده‌ها. به نظر می‌رسد الگوریتم‌هایی که وظیفه‌ی دسته‌بندی داده‌ها را به عهده دارند، مهمترین روش‌ها در آماده‌سازی داده‌ها به شمار می‌روند. این الگوریتم‌ها در کلیه‌ی مراحل آماده‌سازی داده‌ها استفاده می‌شوند. بطور مثال برای مشخص کردن داده‌های ناقص و نادرست و حتی به منظور کاهش اندازه‌ی داده‌ها. این الگوریتم‌ها داده‌ها را در گروه‌های

مختلف قرار می‌دهند و مشخصات هر یک از گروه‌ها را تحلیل می‌کنند. تکنیک‌های خوشبندی و درخت‌های تصمیم از راهکارهای مناسب برای این کار به حساب می‌آیند. در سیستم‌های تجاری که اکثر داده‌ها از نوع کمی هستند، این دو دسته از تکنیک‌ها می‌توانند به خوبی عمل کنند. نمایش داده‌ها در قالب نمودار مانند نمودار پراکندگی و هیستوگرام‌ها، گروه‌بندی در خوشبندی متناسب را تسهیل می‌کند. استنباط عمیق‌تر ممکن است با به کارگیری تکنیک‌های گرافیکی پیشرفته حاصل شود.

- انتخاب یک زیرمجموعه‌ی مناسب از داده‌ها(کاهش داده‌ها). در پایگاه داده تعداد زیاد نمونه‌ها و صفات خاصه یکی از مشکلات بزرگ است. این داده‌های زیاد فضای جستجو را افزایش و شناس رساندن به یک الگوی نادرست و بدردنخور را افزایش می‌دهند. بسیاری از الگوریتم‌های موجود مثل الگوریتم‌های خوشبندی با مجموعه داده‌های زیاد خوب عمل نمی‌کنند. کاهش حجم داده‌ها یکی از پیچیده‌ترین و در عین حال مهمترین مراحل در آماده‌سازی داده‌ها به شمار می‌رود. به دلیل حجم بالای داده‌ها، در اجرای الگوریتم با فضای بسیار زیادی از انتخاب‌ها روبرو هستیم که انتخاب بهترین راه حل زمانگیر و یا گاهی غیرممکن است. استفاده از یک قانون اکتشافی برای به دست آوردن بهترین زیرمجموعه از صفات خاصه روش خوبی است. شرط اصلی در گزینش یک زیرمجموعه‌ی مناسب از داده‌هایی با ابعاد بالا این است که این زیرمجموعه علیرغم کوچک بودن دارای خصوصیات داده‌های اصلی نیز باشد. الگوریتم‌های ژنتیک می‌توانند از این فضای جستجوی وسیع انتخاب‌های بهینه‌ای را به ما ارائه دهند. یکی از تکنیک‌های معمول برای تلخیص داده‌ها روش تحلیل مولفه‌های اصلی است که به تفصیل به آن پرداختیم. الگوریتم‌هایی که برای کاهش ابعاد داده‌ها استفاده می‌شوند، اغلب دارای پیچیدگی محاسباتی بالایی هستند، لذا پیشنهاد می‌گردد تا در صورت امکان این الگوریتم‌ها بصورت موازی پیاده‌سازی شوند. استفاده از الگوریتم‌های خاص فشرده‌سازی راهکار دیگری برای کاهش داده‌ها تلقی می‌شود. در واقع این الگوریتم‌ها همان فرمول‌های پیچیده‌ی آماری هستند، که

برای ذخیره‌ی داده‌ها، با به دست آوردن الگوهای نهفته در داده‌ها، حجم کمتری را برای نگهداری آنها مصرف می‌کنند.

مراجع و منابع جهت مطالعه‌ی بیشتر

کتاب‌های متعددی را می‌توان یافت که در زمینه‌ی پیش‌پردازش داده‌ها صحبت می‌کنند. منابعی چون [Los01] [DJ03] [Red01] و [Py199] نمونه‌هایی از این کتب هستند. اما منابعی را نیز می‌توان یافت که در راستای تکنیک‌های خاص برای آماده‌سازی داده‌ها نوشته شده‌اند. مراجع [BFOS84] [Qui89] و [HP07] در مورد کنترل داده‌های ناقص و مفقود بحث می‌کنند. این مراجع بیشتر حاوی مقالاتی هستند که در کنفرانس‌ها و یا ژورنال‌های معتبر ارائه شده‌اند. منابعی را نیز می‌توان یافت که در مورد دسته‌بندی داده‌ها و نرمال‌سازی آنها مطالبی را تشریح کرده‌اند. برخی از آنها عبارتند از [Py199] [WI98] [KLV⁺⁹⁸] و [BDF⁺⁹⁷].

گزارش فنی [BDF⁺⁹⁷] منبع بسیار مناسبی برای توضیح روش‌های کاهش داده‌ها تلقی می‌شود. چگونگی انتخاب زیرمجموعه‌ای از صفات خاصه در [NKNW96] [LM98a] [DL97] و [LM98b] بررسی شده‌اند. همانطور که می‌دانید برای انتخاب زیرمجموعه‌ای از صفات خاصه می‌توان با مجموعه‌ی کاملی از صفات خاصه شروع و در هر مرحله صفات خاصه‌ای را حذف کنیم یا اینکه با مجموعه‌ی تهی شروع و در هر مرحله از اجرا صفات خاصه‌ی مناسبی را به این مجموعه اضافه کنیم. در [SS88] ترکیبی از این دو روش بررسی شده است.

در اکثر بسته‌های نرم‌افزاری آماری مانند SAS روتین‌هایی برای PCA یافت می‌شوند ولی [PTVF07] نیز کتاب مناسبی است که مفاهیم اولیه‌ی PCA در آن بحث شده است.

برای بحث رگرسیون‌ها نیز منابع زیادی وجود دارند که در این میان می‌توان به [NKNW96] [Pea88] [Dev95] [JW92] [Dob90] [Jam85] اشاره نمود.

برای موضوع نمونه‌گیری و داده‌کاوی دو منبع [KM94] و [JL96] را جستجو کنید. منبع [LHTD02] مرور کاملی بر روی روش‌های گسسته‌سازی داده‌ها دارد. اما مراجع دیگری نیز هستند که در آنها از راهکارهای خودکار گسسته‌سازی داده‌های پیوسته یا عددی صحبت شده است. بعضی از آنها عبارتند از [Qui93] [Ker92] [LS95] و [FI93]. سلسله‌مراتب مفهومی و تولید خودکار آن از داده‌های نوع طبقه‌بندی شده نیز در [HF94] توضیح داده شده است.

تکنیک‌های تجسم‌سازی یا به عبارت بهتر بصری‌سازی از روش‌های پرکاربرد و مورد علاقه کاربران و تحلیل‌گران است که هم در مرحله‌ی پیش‌پردازش داده‌ها می‌توان از آن استفاده کرد و هم برای ارائه نتایج پایانی فرایند کشف دانش. کتب و مقالات متعددی یافت می‌شوند که در این زمینه بحث کرده‌اند و [Tuf97] [Tuf90] [Tuf83] [Cle93] [FGW01] و [Ber81] از این جمله هستند.

فصل سوم

انبار داده‌ها و تکنولوژی *OLAP*

رشد چشمگیر داده‌ها مدیران را مجبور ساخته تا از ابزاری جهت تصمیم‌گیری‌های مناسب استفاده کنند. خواسته‌های آنها بسیار پیچیده‌تر از آن است که بتوان با کمک یک زبان مانند *SQL* پاسخی مناسب برای آن پیدا کرد. به همین دلیل تکنیک‌ها و ابزارهای متعددی در دسترس قرار گرفته‌اند که در اخذ تصمیمی مناسب به کمک مدیران می‌آیند. بعضی از این ابزارها به تحلیل‌گر اجازه می‌دهند تا داده‌ها را از زوایای متفاوت بررسی و مشاهده کند و برخی دیگر به خلاصه‌سازی حجم بالای این داده‌ها می‌پردازند، تا بتوان پاسخی سریع برای پرس‌وجوی کاربر یافت. برخی از نسخه‌های *SQL* شامل مفاهیم و ساختارهای جدیدی هستند که در تحلیل داده‌ها جهت تصمیم‌گیری به کمک کاربران می‌آیند. تکنیک‌ها و الگوریتم‌های داده‌کاوی نیز که هر یک موضوع فصل‌های بعدی را تشکیل می‌دهند، با هدف تشخیص الگوهای متفاوت در داده‌ها به کمک تحلیل‌گر خواهند آمد. در این فصل به بررسی مفاهیم *OLAP* و همچنین انبار داده‌ها می‌پردازیم.

۱-۳) سیستم‌های تصمیم‌ساز^۱

برنامه‌های کاربردی پایگاهداده‌ها را می‌توان در دو گروه دسته‌بندی نمود. دسته‌ی اول سیستم‌های پردازش تراکنش‌ها هستند و همانطور که از نام آن مشخص است، به ضبط اطلاعاتی در مورد تراکنش‌های پایگاه داده‌ها می‌پردازن. نمونه تراکنش‌هایی که مربوط به فروش محصولات یک کمپانی می‌شوند و یا اطلاعاتی در مورد دانشجویان یک دانشگاه از مثال‌هایی هستند که در این دسته از سیستم‌ها گنجانده می‌شوند. امروزه از این گونه سیستم‌ها به وفور استفاده می‌شود و با کمک آنها داده‌های حجمی تولید شده است.

هدف دسته‌ی دوم که به سیستم‌های تصمیم‌ساز شناخته می‌شوند، کسب اطلاعات در سطوح بالاتری از داده‌های جمع‌آوری شده توسط سیستم‌های پردازش تراکنش‌ها است. یک سیستم تصمیم‌ساز با شناسایی نشانه‌های کلیدی به مدیران کمک می‌کند تا تصمیمات مهم و حیاتی را اتخاذ نمایند.

برای مثال پایگاهداده‌های یک فروشگاه زنجیره‌ای را در نظر بگیرید. این پایگاه داده شامل اطلاعات زیادی در مورد مشتریان و تراکنش‌های آنها است. اطلاعات مشتریان می‌تواند شامل نام هر یک از آنها، اقلامی که خریده‌اند، مبلغ پرداختی و تاریخ خرید باشد. همچنین برای کالاها و اجنباس اطلاعاتی مانند قیمت، مدل و کارخانه‌ی سازنده می‌تواند نگهداری شود. حجم بالای این داده‌ها می‌تواند گنجی از اطلاعات مفید برای تصمیم‌گیری‌های مناسب باشد. تصمیم‌گیری‌هایی نظیر مقدار تخفیف‌های لازم برای برخی از کالاها، آگاهی دادن به بخشی از مشتریان برای اجنباس مخصوص و یا خرید محصولات مشخص در زمان‌های معین، در روند رو به رشد این کمپانی یا فروشگاه بی‌تأثیر نیست. مثال‌های مختلف فراوانی را می‌توان یافت که وضعیتی مشابه با مثال ذکر شده دارند. در واقع در اکثر موارد با یافتن و بررسی الگوهایی در داده‌ها، تصمیماتی گرفته می‌شوند. برای ذخیره و بازیابی این داده‌های حجمی جهت تصمیم‌گیری، موضوعاتی هستند که باید آنها را بررسی نمود. برای مثال اگر چه بسیاری از پرسش‌هایی که در تصمیم‌گیری به ما کمک می‌کنند را می‌توان در SQL نوشت، اما برخی از آنها را یا اصلاً نمی‌توان با این زبان پیاده‌سازی نمود و یا حداقل به راحتی امکان‌پذیر نیست. هرچند نسخه‌های تعمیم یافته‌ی SQL

^۱ Decision Support Systems

تسهیلاتی را جهت این هدف در خود تعییه نموده است. در حوزه‌ی **OLAP**^۱ ابزارها و تکنیک‌هایی یافت می‌شوند که جهت تحلیل داده‌ها می‌توان پاسخی فوری را برای خواسته‌هایی چون خلاصه‌سازی داده‌ها پیدا نمود. در ضمن زبان‌های پرس‌وجوهی پایگاه داده‌ها برای تحلیل‌های آماری پیشرفته‌ی داده‌ها مناسب به نظر نمی‌رسند.

بسته‌های نرم‌افزاری گوناگونی مانند **SAS** و **S++** موجودند که این امکانات را فراهم می‌کنند. تنوع داده‌های یک سازمان نیز از موضوعاتی است که در ذخیره و بازیابی این نوع اطلاعات چالش‌هایی را ایجاد می‌کند. جهت اجرای موثر پرس‌وجوهای در چنین محیطی، معمولاً^۲ کمپانی‌ها انبارهای داده‌ای را از اطلاعات خود می‌سازند. هدف اصلی یک انبار داده^۳ افزایش هوشمندی در یک فرایند تصمیم‌گیری و افزایش دانش افراد درگیر در این فرایند است. در بخش‌های بعدی توضیحاتی پیرامون **OLAP** و انبار داده‌ها داده شده است.

۳-۲) تحلیل داده‌ها و **OLAP**

اگر چه تحلیل‌های پیچیده‌ی آماری به عهده‌ی نرم‌افزارهای آماری گذاشته می‌شود، اما پایگاه داده‌ها نیز باید از شکل‌های ساده‌ی تحلیل‌های داده‌ها پشتیبانی کند. از آنجا که معمولاً^۴ پایگاه داده‌ها دارای حجم بالای داده‌های است، بهتر است که جهت استخراج اطلاعات مفید توسط انسان این داده‌ها خلاصه‌سازی شوند. در حالیکه^۵ **OLTP** بطور مشخص در مورد داده‌های تراکنشی مورد استفاده قرار می‌گیرد، ابزارهای **OLAP** برای سیستم‌های تصمیم‌ساز در تحلیل اطلاعات تجمعی شده‌ی برنامه‌های کاربردی بسیار سودمند است. سیستم **OLTP** به دلایل متعددی برای پرس‌وجوهای سیستم‌های تصمیم‌ساز مناسب نیستند. چرا که بدست آوردن اطلاعات خلاصه، نیازمند پرس‌وجوهایی است که از داده‌های تراکنشی بسیار زیاد با ارتباط میان جداول مختلف تشکیل شده است. به علاوه طراحی بانک اطلاعاتی در سیستم **OLTP** به گونه‌ای است که برای

¹ Online Analytical Processing

² Data Warehouse

³ Online Transaction Processing

پاسخگویی به سوالات تصمیم‌ساز خوب نیست. جهت اثبات این ادعا وظایف متعدد و رایجی را می‌توان پیدا کرد که با کمک توابع تجمعی و گروه‌بندی *SQL* قابل انجام نیستند. اخیراً امکاناتی در *SQL* پیشنهاد شده‌اند که این وظایف را پشتیبانی می‌کنند و در محصولاتی نظیر *IBM DB2* و *Oracle* پیاده‌سازی شده‌اند.

OLAP (۳-۲-۱)

در تحلیل‌های آماری اغلب عمل گروه‌بندی بر روی چندین صفت خاصه و ویژگی لازم به نظر می‌رسد. برای فهم بهتر این موضوع اطلاعات فروش یک مرکز فروش الکترونیکی را در نظر بگیرید. جدول ۱-۳ بخشی از این داده‌ها را نشان می‌دهد.

جدول ۱-۳: بخشی از پایگاه داده‌ی یک فروشگاه لوازم الکترونیکی

ID	Name	Mark	Color	Number
1	Computer	Sony	Black	7
2	Computer	Samsung	Black	8
3	Computer	Samsung	Silver	10
4	Computer	LG	White	5
5	Notebook	Sony	Black	15
6	Notebook	Sony	Silver	20
7	Notebook	Samsung	White	6
8	Notebook	LG	Black	4
9	Tablet	Sony	Black	3
10	Tablet	Samsung	Black	12
11	Tablet	Samsung	Silver	11
12	Tablet	Samsung	White	9
13	Mobile	Samsung	Black	16
14	Mobile	Samsung	White	13
15	Mobile	LG	Silver	14
16	Mobile	LG	White	17

این جدول اطلاعاتی نظیر نام محصول، مارک، رنگ و تعداد فروخته شده را نشان می‌دهد.

هرچند در کاربردهای واقعی صفات خاصه‌ی دیگری را نیز می‌توان برای این جدول متصور شد. حال با کمک این داده‌ها، برخی از مفاهیم را در ادامه‌ی بحث توضیح می‌دهیم.

معمولًاً برای تحلیل داده‌ها بر روی موضوعی مرکز می‌شویم و با توجه به این موضوع محوری برخی از صفات خاصه به عنوان ویژگی‌های *measure* شناخته می‌شوند. در مثال ذکر شده قصد داریم وضعیت فروش محصولات را به عنوان محور تعیین کنیم، لذا صفت خاصه‌ی *Number* را که دلالت بر تعداد فروخته شده‌ی هر محصول دارد، به عنوان ویژگی یا صفت خاصه‌ی *measure* در نظر می‌گیریم. در واقع تابع تجمعی^۱ بر روی این ویژگی اعمال خواهد شد. تعدادی (یا همه) از صفات خاصه‌ی دیگر در جدول به عنوان صفات خاصه‌ی *dimension* معرفی می‌شوند. این صفات خاصه ابعاد را بر روی ویژگی‌های *measure* مشخص و آنها را تلخیص (یا تجمعی) می‌کنند. در مثال ذکر شده نام محصول، مارک و رنگ آن از این نوع ویژگی‌ها به شمار می‌روند. حال با کمک این دو نوع صفت خاصه و یا ویژگی، داده‌ها می‌توانند مدل‌سازی و به عنوان یک داده‌ی چندبعدی^۲ شناخته شوند. جهت تحلیل داده‌های چندبعدی، یک مدیر شاید مایل باشد داده‌ها را همانند شکل ۱-۳ ببیند.

	Sony	Samsung	LG	Total
Computer	7	18	5	30
Notebook	35	6	4	45
Tablet	3	32	0	35
Mobile	0	29	31	60
Total	45	85	40	170

شکل ۱-۳: جدول متقطع فروش محصولات با ترکیب‌های مختلف از مقادیر نام محصول و مارک

این جدول متقطع مقدار فروش محصولات فروشگاه را با ترکیب‌های مختلفی از مقادیر صفات خاصه‌ی نام محصول و مارک آن نشان می‌دهد. سطرها نام محصول و ستون‌ها

¹ Aggregation Function

² Multidimensional Data

مارک آن را نشان می‌دهند. در این جدول از تابع تجمعی مجموع استفاده شده است، اما با توجه به نیاز کاربر یا تحلیل‌گر می‌توان از تابع دیگری مانند میانگین نیز استفاده نمود. همانطور که مشاهده می‌کنید، ویژگی رنگ در این جدول وجود ندارد، که معنی آن این است که برای هر محصول کلیه‌ی رنگ‌ها در نظر گرفته شده‌اند. این جدول می‌تواند برای ترکیب‌های دو تایی دیگری از صفات خاصه نیز ساخته شود. شکل ۳-۱ مثالی از یک ساخت این گونه از جداول دو صفت‌خاصه‌ی *dimension* انتخاب می‌شوند. برای ساخت این گونه از جداول دو صفت‌خاصه‌ی *dimension* انتخاب می‌شوند، که در مثال ما نام محصول و مارک آن انتخاب شده‌اند. مقادیر موجود این صفات خاصه تعداد سطرها و ستون‌های جدول را تشکیل می‌دهند. محتوای این جدول با اعمال یک تابع تجمعی (در مثال مجموع) بر روی مقادیر صفت‌خاصه *measure* و با توجه به سطر و ستون قرار گرفته شده، محاسبه می‌شود. برای مثال در شکل ۳-۱ به سطحی که به محصول *Notebook* و ستونی که مارک *Sony* اشاره می‌کند نگاه کنید. مقدار عددی ۳۵ را مشاهده می‌کنید. این مقدار از مجموع مقادیر صفت‌خاصه *Number* برای سطرهای پنجم و ششم از جدول اصلی (جدول ۱-۳) که مقدار ویژگی *Name* آنها برابر با *Notebook* و مقدار صفت‌خاصه *Mark* آنها برابر با *Sony* است، بدست آمده است. این مقادیر به ترتیب ۱۵ و ۲۰ هستند که پس از جمع مقدار ۳۵ در جدول نگهداری می‌شود. این مقدار صرف نظر از مقادیر موجود برای رنگ این رکوردها محاسبه شده است. به علاوه در کاربردهای واقعی شما با صدها و یا هزاران رکورد موافق خواهید بود و همانطور که قبلاً از این نیز به آن اشاره کردیم می‌توان به جای مجموع از تابع تجمعی دیگری مانند میانگین، حداکثر و حداقل بسته به کاربرد استفاده نمود. در شکل ۳-۱ یک سطر و یک ستون اضافی برای نمایش مجموع محتوای جدول نیز نمایش داده شده است، تا اطلاعات بیشتری در اختیار تحلیل‌گر قرار گیرد. محتوای شکل ۱-۳ را می‌توان به شکل یک جدول نیز نمایش داد (جدول ۲-۳).

یک *cross-tabulation* با جداول ذخیره شده در یک پایگاه داده‌ی رابطه‌ای متفاوت است. تعداد سطرها و ستون‌ها در این نوع جداول به مقادیر موجود در داده‌ها بستگی دارند و با اضافه شدن یک سطر در جدول داده‌های اصلی، ممکن است سطر یا ستونی به یک

جدول ۳-۲: شکل دیگری از جدول متقاطع نشان داده شده در شکل ۳-۱

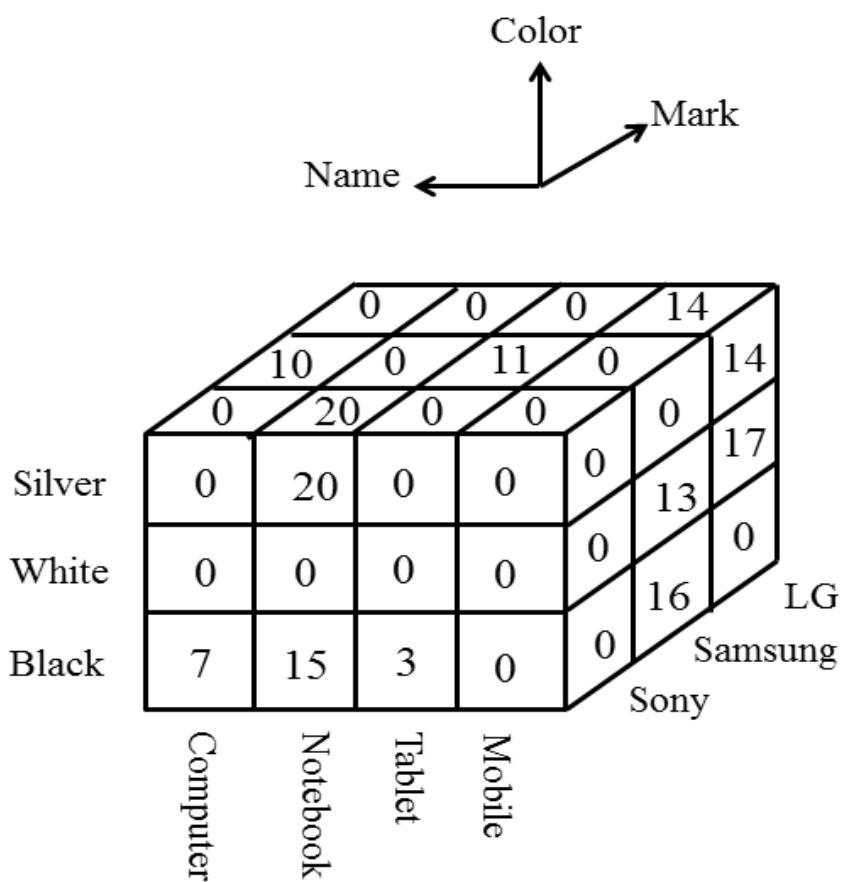
Name	Mark	Color	Number
Computer	Sony	All	7
Computer	Samsung	All	18
Computer	LG	All	5
Computer	All	All	30
Notebook	Sony	All	35
Notebook	Samsung	All	6
Notebook	LG	All	4
Notebook	All	All	45
Tablet	Sony	All	3
Tablet	Samsung	All	32
Tablet	LG	All	0
Tablet	All	All	35
Mobile	Sony	All	0
Mobile	Samsung	All	29
Mobile	LG	All	31
Mobile	All	All	60
All	Sony	All	45
All	Samsung	All	85
All	LG	All	40
All	All	All	170

افزوده شود. سطرهای نیز شامل مقدار **All** هستند. برای مثال سطر دوم در جدول ۳-۲ بین معنی است که از کامپیوتر مارک سامسونگ صرفنظر از رنگ آن تعداد ۱۸ دستگاه به فروش رسیده است.

حال نوبت به آن رسیده تا یک *cross-tabulation* که دو بعدی است را به داده‌های چندبعدی تبدیل کنیم. به عبارت دیگر می‌خواهیم داده‌ها را با بیش از دو بعد نمایش دهیم. این شکل از داده‌ها با نام مکعب یا *cube* چندبعدی^۱ شناخته می‌شوند. توجه کنید، بر خلاف مرسوم میان عام که کلمه‌ی مکعب شکلی سه بعدی را در ذهن متبلور می‌کند،

^۱ Multidimensional Cube

در مبحث ما یک مکعب داده‌ها می‌تواند دارای ابعاد بالاتر از سه نیز باشد. به همین دلیل در ادامه آن را با واژه‌ی انگلیسی *cube* یا *data cube* می‌شناسیم. برای مثال شکل ۳-۲ یک *cube* سه بعدی برای داده‌هایی که بخشی از آن در جدول ۱-۱ آمده است را نشان می‌دهد. ابعاد آن را نام محصول، مارک و رنگ آن مشخص نموده است و ویژگی آن تعداد فروخته شده از هر جنس است که با صفت‌خاصه‌ی *measure* در جدول معرفی می‌شود. توجه کنید که بعضی از سلول‌ها و محتویات آنها در شکل دیده نمی‌شوند.



شکل ۳-۲: یک *cube* سه‌بعدی برای داده‌های جدول ۱-۱

هر سلول در *cube* دارای یک مقدار است که با توجه به ابعاد و تابع تجمعی که بر روی مقادیر صفت‌خاصه‌ی *measure* اعمال می‌شود، محاسبه خواهد شد. آدرس هر سلول در واقع مقدار صفات‌خاصه‌ی *dimension* را برای آن سلول نشان می‌دهد. می‌توانیم برای

هر بُعد از *cube* علاوه بر مقادیر موجود برای هر صفت خاصه مقدار *All* را نیز اضافه کنیم، معنی این مقدار همانی است که برای *cross-tabulation* توضیح داده شد.

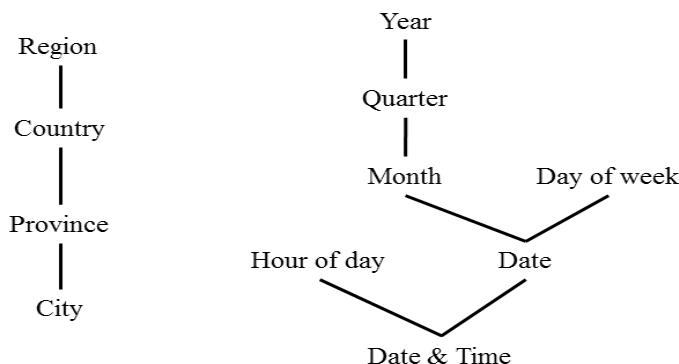
یک سیستم *OLAP* با تحلیل‌گر به صورت تعاملی کار می‌کند و اجزه می‌دهد تا او زاویه‌ی دیدهای متفاوتی از داده‌های چندبعدی داشته باشد. تحلیل‌گر همچنین قادر است تا درخواست جدیدی از خلاصه‌سازی داده‌ها ارائه کند و ابزار *OLAP* در زمانی کوتاه پاسخ را مهیا می‌کند و لازم نیست تا برای دیدن نتایج یک پرس‌وجو تحلیل‌گر برای مدتی طولانی انتظار بکشد.

با کمک یک سیستم *OLAP* تحلیل‌گر داده‌ها می‌تواند برای داده‌های یکسان جداول متقطع یا *cross-tabulation* متفاوتی را مشاهده کند. همانطور که می‌دانیم هر *cube* یک نوع مشاهده‌ی دو بُعدی بر روی *cross-tabulation* شما ارائه می‌کند. برای مثال شاید تحلیل‌گر زوج صفات خاصه‌ی نام محصول و مارک آنرا برای یک *cross-tabulation* انتخاب کند و یا صفات خاصه‌ی رنگ و مارک را برای تحلیل خود مناسب‌تر بداند. عملیات تغییر و انتخاب ابعاد استفاده شده در یک جدول *pivoting* را با نام *cross-tabulation* می‌شناسیم.

یک سیستم *OLAP* عملیات دیگری را نیز در خود دارد. برای مثال تحلیل‌گر ممکن است مایل باشد تا یک جدول *cross-tabulation* بر روی نام محصول و مارک و برای مقدار ثابتی از رنگ (نه همه‌ی رنگ‌ها) را مشاهده کند. به این عمل *slicing* اطلاق می‌شود، چرا که با این عمل شما بخشی از داده‌های *cube* را برش داده‌اید. گاهی اوقات به این عمل *dicing* نیز گفته می‌شود، به خصوص هنگامی که مقادیر چندین صفت خاصه‌ی *dimension* در آن ثابت فرض شوند.

هنگامی که یک *cross-tabulation* دارید، برخی از صفات خاصه در انتخاب شما قرار نمی‌گیرند و شما دو *dimension* را انتخاب می‌کنید. بدین سبب معمولاً در بالای این صفات خاصه انتخاب نشده همراه با مقدار آنها نمایش داده می‌شوند. همانند قبل این مقدار می‌تواند *All* باشد و یا مقداری مشخص که عمل *dicing* بر روی داده‌های *cube* شما انجام می‌دهد.

موضوع دیگر اینکه سیستم‌های *OLAP* به کاربران اجازه می‌دهند تا داده‌ها را در هر سطحی از دانه‌بندی^۱ مشاهده کنند. برای نمونه به داده‌های مثال می‌توانید ستون یا ویژگی تاریخ و زمان فروش را نیز اضافه کنید. بدین ترتیب زمان فروش هر محصول را نیز می‌دانید. حال به دلخواه تحلیل‌گر، بُعد مربوط به زمان می‌تواند بر حسب روز، ماه، فصل و یا حتی سال باشد که نام آنها از دانه ریز^۲ به دانه درشت^۳ بیان شده است. عملیات *rollup* به عملیاتی اطلاق می‌شود که طریقه‌ی نمایش داده‌ها را از دانه ریز به دانه درشت نشان می‌دهد. برای مثال اگر واحد صفت‌خاصه‌ی زمان روز است و برای نمایش *cube* آن را به ماه یا فصل تبدیل می‌کنید، در واقع *rollup* کرده‌اید. عملیات بر عکس که در آن از دانه درشت به دانه ریز حرکت می‌کنیم با نام عمل *drill down* شناخته می‌شود. توجه کنید که داده‌های مربوط به دانه‌ریزها نمی‌توانند از داده‌های دانه درشت تر محاسبه شوند. در این حالت یا باید به داده‌های اصلی رجوع کنید و یا از داده‌هایی با درجه‌ی ریزی بیشتر باید استفاده شود. برای مثال با داشتن مقادیر فروش ماهیانه، می‌توان به فروش فصلی یا سالانه دسترسی داشت، اما نمی‌توان با کمک آن به فروش روزانه رسید. سطوح متفاوتی از جزئیات یک صفت‌خاصه را می‌توان در یک سلسله‌مراتب سازماندهی نمود. شکل ۳-۳ دو سلسله‌مراتب یکی برای مکان و دیگری برای زمان را نشان می‌دهد.



شکل ۳-۳: سلسله‌مراتب زمان (شکل راست) و سلسله‌مراتب مکان (شکل چپ)

¹ Granularity

² Fine Granularity

³ Coarse Granularity

به هر حال یک تحلیل‌گر ممکن است شکل‌های متفاوتی از داده‌ها را برای تحلیل مناسب بداند و با عملیات ذکر شده این شکل از داده‌ها را از داده‌های اصلی یا *cube* چندبعدی تشکیل شده، محاسبه کند. برای مثال در شکل ۳-۴ مخصوصات در دو گروه تک هسته‌ای و چندهسته‌ای با یک *cross-tabulation* واحد نمایش داده شده است.

۳-۲-۲ پیاده‌سازی ***OLAP***

در ابتدا سیستم‌های *OLAP* برای ذخیره‌ی *cubes* از آرایه‌های چندبعدی^۱ استفاده می‌کردند و به همین دلیل با نام سیستم‌های *OLAP* چندبعدی^۲ یا به صورت خلاصه *MOLAP* شناخته شده‌اند. در این روش ذخیره‌سازی داده‌های هر بخش در ساختار ویژه‌ای نگهداری می‌شوند و این موضوع امکان بازیابی موثری از داده‌های چندبعدی را فراهم می‌سازد. پس از آن قابلیت‌های *OLAP* در کنار سیستم‌های رابطه‌ای قرار گرفت تا داده‌ها در یک پایگاه داده‌ی رابطه‌ای نگهداری شوند.

		<i>Sony</i>	<i>Samsung</i>	<i>LG</i>	<i>Total</i>
<i>Single Core</i>	<i>Computer</i>	1	13	5	19
	<i>Notebook</i>	5	2	2	9
	<i>Tablet</i>	2	12	0	14
	<i>Mobile</i>	0	20	25	45
	<u><i>Subtotal</i></u>	<u>8</u>	<u>47</u>	<u>32</u>	<u>87</u>
<i>Multicore</i>	<i>Computer</i>	6	5	0	11
	<i>Notebook</i>	30	4	2	36
	<i>Tablet</i>	1	20	0	21
	<i>Mobile</i>	0	9	6	15
	<u><i>Subtotal</i></u>	<u>37</u>	<u>38</u>	<u>8</u>	<u>83</u>
	<i>Total</i>	45	85	40	170

شکل ۳-۴: نمایش یک جدول متقارع همراه با گروه‌بندی صفات خاصه

¹ Multidimensional Arrays

² Multidimensional OLAP

در این سیستم‌ها که به *OLAP* رابطه‌ای^۱ یا *ROLAP* معروف هستند، تمام داده‌ها در یک سیستم پایگاه داده‌ی رابطه‌ای ذخیره می‌شوند. این ساختار در مقایسه با *MOLAP* از مقیاس‌پذیری^۲ بهتری برخوردار است، ولی در عوض برای پاسخ به برخی از پرسش‌ها از کارایی کمتری بهره می‌برد و جهت دسترسی به داده‌ها تنها به ابزار *SQL* نیاز دارد. در ضمن استفاده از مدل رابطه‌ای باعث شده است تا افزونگی به حداقل برسد. سیستم‌های *OLAP* تلفیقی^۳ یا *HOLAP* همانطور که از نامشان مشخص است، ترکیبی از دو سیستم فوق هستند. در این سیستم‌ها معمولاً داده‌های اصلی در قالب رابطه‌ای ذخیره و دیگر داده‌های تجمیعی در حافظه نگهداری می‌شوند.

با افزایش ابعاد داده‌ها تنوع *cube*‌هایی که می‌توان محاسبه و پردازش نمود نیز افزایش می‌یابد. برای ساخت همه‌ی آنها استفاده از یک الگوریتم ساده و بدون هیچگونه هیوریستیکی بسیار مشکل است. چرا که باید پیمایش‌های زیادی را بر روی داده‌هایی با حجم بالا داشته باشیم. ساده‌ترین نکته در بهینه‌سازی این عملیات ساخت *cube*‌ها از *cubes*‌های موجود در سطوح دیگر است. برای مثال جهت ساخت (*Name,Mark*) به جای استفاده از داده‌های اولیه می‌توان با کمک توابع تجمعی از *cube* ساخته شده‌ی (*Name,Mark,Color*) بهره گرفت. البته این عمل همیشه شدنی نیست و استفاده از برخی توابع تجمعی مانند میانه باعث می‌شود تا به دنبال راهکار دیگری باشیم. استفاده از چنین هیوریستیک‌هایی باعث می‌شود تا برای محاسبه‌ی *cube* با حجم کمتری از داده‌ها نسبت به استفاده از داده‌های اولیه روبرو باشیم. موضوعات دیگری نیز وجود دارند تا در بهبود ساخت یک *cube* به شما کمک کنند. برای مثال در حین یک پیمایش ممکن است چندین گروه‌بندی صفات خاصه را یکجا محاسبه نمود. در حال حاضر یافتن الگوریتم‌های کارا جهت محاسبه و ساخت *cube* موضوع بسیاری از طرح‌های پژوهشی محققین را تشکیل می‌دهد.

¹ Relational OLAP

² Scalability

³ Hybrid OLAP

پیاده‌سازی‌های اولیه در *OLAP* به صورتی بود که در آن تمام *cubes* از قبل محاسبه می‌شدند و بدین ترتیب پاسخ پرس‌وجوها به سرعت آماده بود. حتی زمان پاسخ به این پرس‌وجوها برای داده‌هایی با میلیون‌ها رکورد (دها یا صدها گیگابایت) قابل قبول و سریع است. اما می‌دانیم که با تعداد n صفت خاصه‌ی *dimension* در واقع با 2^n حالت گروه‌بندی این صفات خاصه روبرو هستیم و حال سلسله‌مراتب صفات خاصه را نیز به موضوع اضافه کنید. در نتیجه حجم کلیه‌ی *cubes* از حجم داده‌های اولیه به مراتب بیشتر خواهد بود و در بسیاری از موارد ذخیره‌ی این داده‌ها اصلاً امکان‌پذیر نیست. بنابراین به جای اینکه همه‌ی *cubes* از قبل محاسبه شوند، عاقلانه به نظر می‌رسد که فقط برخی از آنها پردازش شوند و مابقی در صورت تقاضای کاربر یا تحلیل‌گر ساخته شوند. حال برای ساخت برخی از *cubes* می‌توانیم از داده‌های تجمعی موجود به جای داده‌های اولیه استفاده کنیم. توجه کنید که پیش‌ساخت کدامیک از *cubes* در اولویت قرار دارند. این بدین معنی است که انتخاب *cubes* ای که از قبل به محاسبه و ساخت آنها می‌پردازید، می‌تواند در کارایی و پاسخ پرس‌وجوهای آتی به صورت چشمگیری موثر باشد.

۳-۳) انبارسازی داده‌ها^۱

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

^۱ Data Warehousing

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 داشتن یک انبار داده‌ها عمل داده‌کاوی بسیار آسان‌تر می‌شود. هدف اصلی یک انبار داده‌ها تسهیل در فرایند تصمیم‌گیری و افزایش دانش افراد درگیر در این فرایند است. یک انبار داده‌ها مجموعه‌ای از پایگاه داده‌های یکپارچه^۱ و موضوع‌گرا^۲ است به نحوی که هر واحد از داده‌ها در آن وابسته به زمان می‌باشد. انبار داده‌ها مخزنی از اطلاعات است که از منابع مختلفی جمع‌آوری و تحت یک شیوهٔ یکدست ذخیره می‌شود. به عبارت دیگر داده‌ها در انبار داده‌ها از تمامی منابع داده‌ای یکپارچه می‌شوند، بطوريکه این منابع دربرگیرندهٔ اقلام داده‌های مرتبط با موضوع انبار داده‌ها باشند. موضوع‌گرا بودن به این معنی است که یک انبار داده‌ها حول موضوعات خاصی سازماندهی می‌شود. این بدین معنی است که به جای تمرکز بر روی عملیات روزانه و پردازش تراکنش‌های یک سازمان، انبار داده‌ها بر روی مدل‌سازی و تحلیل داده‌ها جهت اخذ تصمیم بهتر متمرکز می‌شود.

از آنجا که معمولاً یک انبار داده‌ها از روی چندین منبع ناهمگن^۳ ساخته می‌شود، لازم است عملیاتی چون پالایش داده‌ها و برخی از تکنیک‌های جمع‌آوری و تبدیل داده‌ها بر روی آن إعمال شود. این تکنیک‌ها از نوع همان‌هایی هستند که در فصل قبل تحت عنوان آماده‌سازی و یا پیش‌پردازش داده‌ها به آنها پرداختیم.

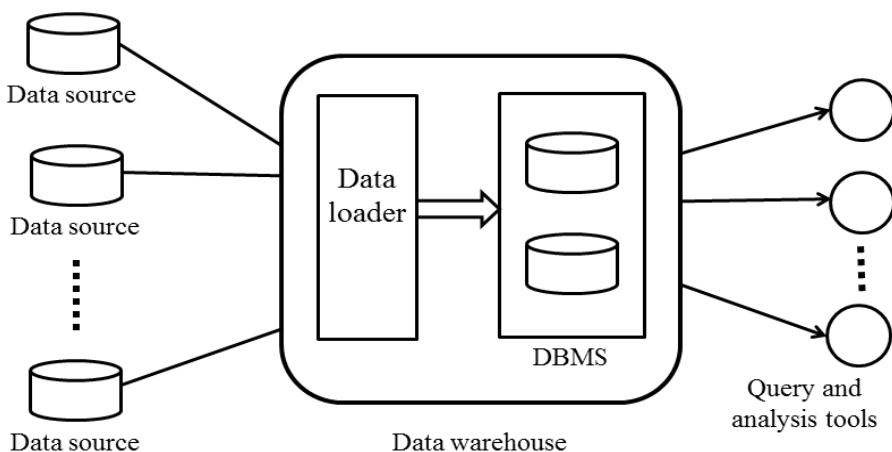
هر ساختار کلیدی در انبار داده‌ها به صورت تلویحی یا مستقیم شامل عنصری از جنس زمان است. بر اساس این توضیحات یک انبار داده‌ها می‌تواند به عنوان مخزن داده‌های یک سازمان در نظر گرفته شود، تا از تصمیم‌گیری‌های راهبردی حمایت کند. وظیفه‌ی آن ذخیره‌ی یکپارچه‌ی داده‌های سازمان است و اغلب در آن داده‌ها بهنگام نمی‌شوند. از آنجا که عمل بهنگام‌سازی انبار داده‌ها عملی زمانبر محسوب می‌شود، داده‌ها اندکی قدیمی هستند که البته این مسئله برای سیستم‌های تصمیم‌ساز مشکل بزرگی تلقی نمی‌شود.

¹ Integrated

² Subject Oriented

³ Heterogeneous

اگل ب یک سازمان ممکن است دارای چندین انبار داده‌ها باشد که به آنها *data mart* گفته می‌شود. در تعریف یک *data mart* انبار داده‌ای است که برای برآورد کردن نیازهای یک گروه خاص از کاربران طراحی شده است و بسته به زمینه‌ی کاری ممکن است بزرگ یا کوچک باشد. به علاوه با داشتن این خصوصیات برای انبار داده‌ها پی خواهیم برد که عملیاتی نظیر کنترل همروندي^۱ و ترمیم^۲ که در مبحث پایگاه داده‌ها مطرح هستند، لازم نیست برای یک انبار داده‌ها در نظر گرفته شوند. شکل ۳-۵ نمونه‌ای ساده از یک معماری را برای انبار داده‌ها نشان می‌دهد.



شکل ۳-۵: نمونه‌ای از یک معماری انبار داده‌ها

اگر چه ایجاد یک انبار داده‌ها کار ساده‌ای نیست و در بسیاری از متون با جزئیات به آن پرداخته شده است، اما مراحل ساخت آن را می‌توان در سه مرحله‌ی استخراج^۳ داده‌ها، تغییر شکل^۴ آن و بارگذاری^۵ که تحت نام خلاصه شده‌ی *ETL* شناخته می‌شوند، خلاصه نمود.

¹ Concurrency Control

² Recovery

³ Extract

⁴ Transform

⁵ Load

۱-۳-۳) شِماهای انبار داده‌ها

هر انبار داده‌ها دارای یک شِما^۱ است که برای تحلیل داده‌ها با کمک ابزاری نظری *OLAP* طراحی می‌شود. معمولاً^۲ یک مدل داده‌ای چندبعدی برای طراحی انبار داده‌ها استفاده می‌شود. از معروف‌ترین شِماها در این مدل داده‌ای می‌توان به شِمای ستاره‌ای^۳ و شِمای برفگونه^۴ اشاره کرد.

شِمای ستاره‌ای از متداول‌ترین شِماهای بانک‌های اطلاعاتی تحلیلی به شمار می‌رود و همانطور که از نامش مشخص است، شکل ستاره‌ای دارد. در این شِما انبار داده‌ها شامل یک جدول مرکزی بزرگ است که به نام جدول *fact* شناخته می‌شود. در این جدول که حاوی تعداد زیادی رکورد است، جزئیات هر رکورد تراکنشی بدون افزونگی ثبت می‌گردد. به همین علت جدول *fact* بزرگ‌ترین جدول در پایگاه داده‌های تحلیلی در این شِما محسوب می‌شود. در اطراف آن، جداول مربوط به هر ویژگی *dimension* قرار می‌گیرد. جدول *fact* نرمال‌سازی شده است و به سایر جداول *dimension* از طریق کلیدهای خارجی^۵ مرتبط می‌شود. در شکل ۳-۶ شِمای ستاره‌ای یک انبار داده را برای یک فروشگاه مشاهده می‌کنید.

جدول *Sales* که در مرکز قرار دارد به عنوان جدول *fact* شناخته می‌شود و حاوی کلیه‌ی تراکنش‌های فروش مربوط به این فروشگاه است. اطلاعاتی مانند مشخصه‌ی محصول، زمان فروش کالا، مشخصه‌ی خریدار و ... در آن ذخیره می‌شوند. در این مثال ۵ جدول *dimension* مشاهده می‌شوند که هر یک از آنها فقط شامل یک جدول و تعدادی صفات خاصه هستند. هر یک از جداول *dimension* با یک (یا چند) صفت‌خاصه در جدول *fact* ارتباط برقرار می‌کند. برای مثال جدول *Customer* که مشخصات کامل مشتریان را درخود دارد با کمک صفت‌خاصه‌ی *Customer-id* در جدول *fact* به این جدول پیوند خورده است. جداول *dimension* می‌توانند دارای افزونگی باشند. برای مثال تصور کنید که برخی از مشتریان ممکن است دارای چندین حساب بانکی باشند و هر

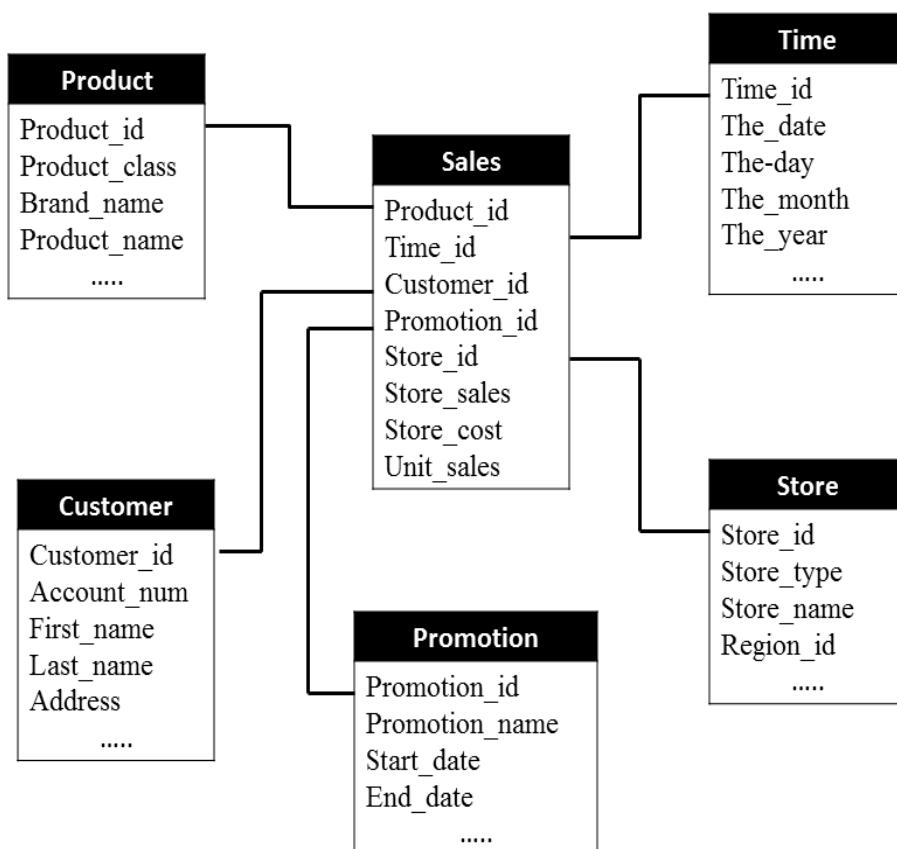
¹ Schema

² Star Schema

³ Snowflake Schema

⁴ Foreign Keys

بار در خرید مایل باشند تا از آنها برای پرداخت مبلغ خریدشان استفاده کنند. در این صورت برای اینچنین مشتریانی در جدول *Customer* با افزونگی روبرو خواهیم بود. اما همه‌ی صفات خاصه موجود در جدول *fact* دارای یک جدول *dimension* نیستند. این صفات خاصه همان ویژگی‌های قبلی این فصل در مورد این نوع از ویژگی‌ها صحبت شد.

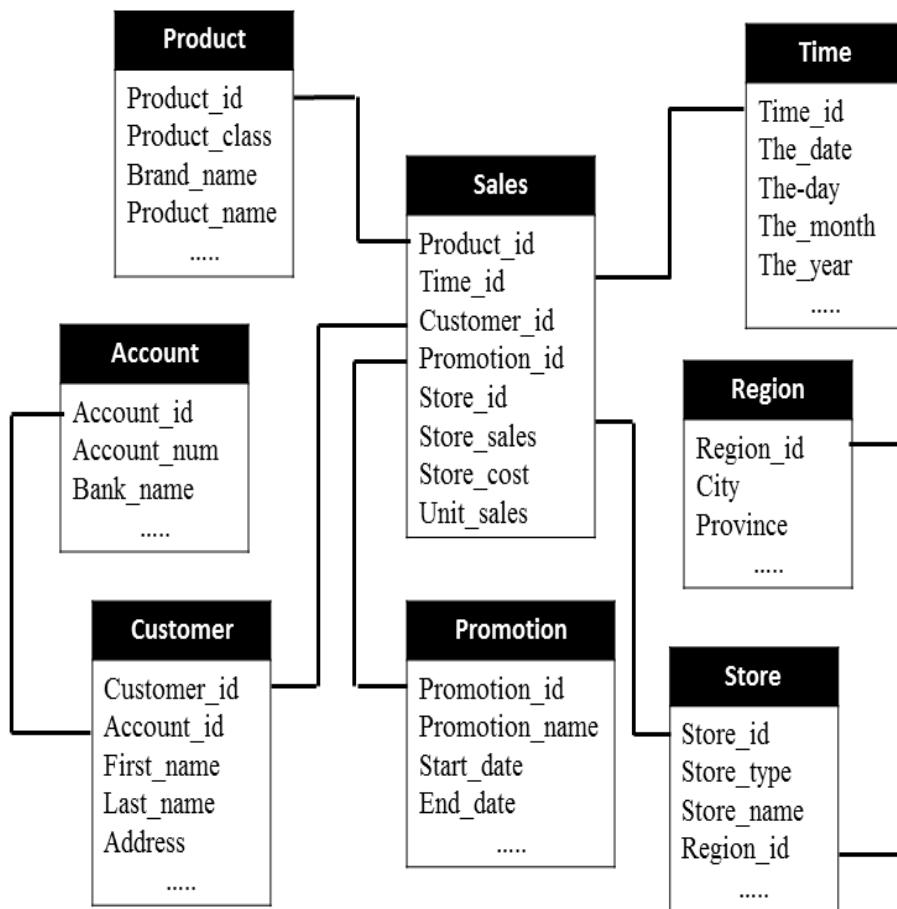


شکل ۳-۶: نمونه‌ای از شیمای ستاره‌ای

شیمای ستاره‌ای گونه‌ای از شیمای دیگری به نام شیمای برفگونه است. در شیمای برفگونه برخی از جداول *dimension* نرمال‌سازی شده‌اند تا افزونگی موجود در شیمای ستاره‌ای در این ساختار وجود نداشته باشد (شکل ۳-۷). برای مثال جدول *Region* به *Store* و *Customer* متصل شده است. این جداول فضای کمتری را نسبت به حالت قبل که دارای افزونگی بود، اشغال می‌کنند. به هر حال فضای صرفه‌جویی

شده در مقابل حجم بالای جدول *fact* ناچیز است. به علاوه به دلیل نرمال بودن جداول، برای برخی از پرس‌وجوها عمل پیوند^۱ بیشتری نیاز خواهد بود. از این رو اگر چه شیمای برفگونه افزونگی را کاهش می‌دهد، اما در طراحی انبار داده‌ها شیمای ستاره‌ای محبوب‌تر و رایج‌تر است.

نوع دیگری از طراحی برای شیمای انبار داده‌ها وجود دارد که در آن می‌توان بیش از یک جدول *fact constellation* پیدا کرد. در این شیما که با نام *fact dimension* شناخته می‌شود، هر یک از جداول *fact* دارای جداول *dimension* مربوط به خود هستند که می‌تواند بین آنها به اشتراک گذاشته شود.



شکل ۷-۳: نمونه‌ای از یک شیمای برفگونه

^۱ Join

خلاصه فصل

در سال‌های اخیر ابزارها و روش‌های *OLAP* بسیار معمول و متداول شده‌اند، به طوری که به کاربران اجازه می‌دهند تا با ایجاد دیدگاه‌های چندگانه از داده‌ها که با نمایش‌های گرافیکی پیشرفته حمایت می‌شود، داده‌های لازم در یک انبار داده را تحلیل و بررسی کنند. استخراج داده‌ها و تحلیل نتایج در *OLAP* شبیه محاسبات در نرم‌افزارهای نوع صفحه‌گسترده هستند. همچنین ابزار *OLAP* از داده‌ها چیزی نمی‌آموزند و دانش جدیدی را ایجاد نمی‌کنند. اغلب آنها ابزار تصویرسازی با اهداف خاصی هستند که قادرند به کاربران نهایی در تصمیم‌گیری بر اساس داده‌های تجمعی کمک کنند. این ابزار برای فرایند داده‌کاوی سودمند هستند، اما نمی‌توانند جانشینی برای داده‌کاوی محسوب شوند. نمایش مجموعه داده‌ها در *cube* چندبعدی و استفاده از عملیاتی مانند *rollup* و *drill down* که با توجه به سلسله‌مراتب صفات خاصه، داده‌ها را به شکل‌های متفاوتی خلاصه‌سازی می‌کنند، باعث می‌شود تا تحلیل‌گر، تفسیر و توجه بهتری به داده‌ها داشته باشد. پیاده‌سازی این داده‌های چندبعدی به سه صورت *ROLAP*، *MOLAP* و *HOLAP* انجام می‌شود که هر یک دارای مزایا و معایب مربوط به خود هستند.

انبار داده‌ها راهکاری جهت یکپارچه کردن داده‌های مرتبط به هم محسوب می‌شود. از آنجا که انبار داده‌ها معمولاً از روی چندین منبع ساخته می‌شوند، لازم است عملیاتی چون پاکسازی داده‌ها و تبدیل آنها به شکلی مناسب بر روی آنها اعمال گردد تا از ناسازگاری و ناقص بودن داده‌ها در آن جلوگیری شود. شیوه‌های ستاره‌ای و برفگونه از معروف‌ترین شیوه‌هایی هستند که در ساخت انبار داده‌ها از آنها استفاده می‌شود و از این میان شیوه ستاره‌ای علیرغم داشتن افزونگی محبوب‌تر و رایج‌تر از نوع برفگونه است.

مراجع و منابع جهت مطالعه‌ی بیشتر

کتاب‌های زیادی را می‌توان یافت که توضیحی مقدماتی در مورد انبار داده‌ها و تکنولوژی *OLAP* را در خود جای داده‌اند و بعضی از معروف‌ترین آنها عبارتند از: [KRTM08]، [GM99] و [CD97]. در [Inm96] و [IGG03] نیز با مجموعه‌ای از مقالات که چگونگی پیاده‌سازی انبار داده‌ها را بررسی کرده‌اند، روبرو هستید.

تاریخچه‌ی سیستم‌های تصمیم‌ساز به سال‌های ۱۹۶۰ برمی‌گردد. پیشنهاد ساخت انبار داده‌های بزرگ جهت تحلیل داده‌های چند بعدی توسط Codd در [CCS93] داده شد، کسی که واژه‌ی OLAP را نیز برای اولین بار استفاده کرد. در همین راستا انجمن OLAP در سال ۱۹۹۵ راهاندازی شد.

مروری بر کاستی‌های SQL در رابطه با توانایی پشتیبانی و مقایسه در دنیای تجارت و همچنین ارائه‌ی مجموعه برنامه‌های کاربردی که به ساختن انبار داده‌ها و تکنولوژی OLAP نیاز دارند را می‌توانید در [KR02] پیدا کنید. مروری بر سیستم‌های OLAP در مقابل پایگاه داده‌ی آماری در [Sho97] بحث شده است.

در [HRU96] یک الگوریتم حریصانه برای محاسبه‌ی یک Cube پیشنهاد شده است. مطالعات متعددی جهت یافتن راهکارهایی برای محاسبه‌ی Cube‌ها انجام شده است که از مهم‌ترین آنها می‌توان به [RS97] [ZDN97] [AAD⁺96] [SS94] [BR99] [HPDW01] و [XHLW03] اشاره کرد.

منابع دیگری نیز وجود دارند که روش‌ها و عملیات گوناگونی را بر روی Cube کندوکاو می‌کنند. برای مثال روش‌هایی برای تخمین اندازه‌ی Cube‌ها را می‌توان در [RS97] [DNR⁺97] [BR99] و [AGS97] جستجو کرد و در [HAC⁺99] ارائه مدل‌سازی پایگاه داده‌ی چند بعدی پیشنهاد می‌شود. همچنین روش‌های دیگری برای پاسخ‌گویی سریع به پرس‌وپرس‌های توسعه تجمعی در [HHW97] و [ZZH09] ارائه شده‌اند.

برای ساخت Cube از روی انواع داده‌های دیگر نیز منابعی یافت می‌شوند. برای داده‌های چند رسانه‌ای به [ZHL⁺98] و برای داده‌های مکان محور به [SHK00] [PKZT01] و [SLT⁺01] و برای تحلیل داده‌های چند بعدی از جنس متن می‌توانید به [LDH⁺08] و [ZZH09] رجوع کنید.

فصل چهارم

الگوهای مکرر و قوانین انجمنی

طی سال‌های گذشته در میان تکنیک‌های داده‌کاوی توجه خاصی به الگوریتم‌های کشف الگوهای مکرر^۱ وجود داشته است. همانطور که از نام آن می‌توان حدس زد، این گونه الگوها به دفعات در مجموعه داده‌ها دیده می‌شوند. در این میان به الگوریتم‌های کشف مجموعه اقلام مکرر^۲ بیشتر پرداخته شده است که در نهایت به تولید قوانین انجمنی^۳ منجر می‌شود.

در این فصل پس از بیان مفاهیم اولیه، الگوریتم‌هایی را بررسی می‌کنیم که به کشف و استخراج الگوهای مکرر می‌پردازند. چگونگی ساخت قوانین انجمنی با کمک الگوهای مکرر موضوع بخش‌های بعدی را تشکیل می‌دهد و در نهایت دو مفهوم الگوهای ماکسیمال و بسته توضیح داده می‌شوند.

¹ Frequent Patterns Mining

² Frequent Itemsets Mining

³ Association Rules

۱-۴) تحلیل سبد خرید^۱

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 حافظه کامپیوتر را اشغال می‌کند. سبد خرید مجموعه‌ای از اقلام خریداری شده بوسیله مشتری در یک تراکنش ساده است. محتویات جدول ۱-۴ شکل ساده‌ای از این داده‌ها را نشان می‌دهد که بطور معمول در هر سطر خصیصه‌ای منحصر به فرد برای شناختن هر تراکنش (مشخصه‌ای مانند شماره تراکنش) همراه با مجموعه‌ای از اقلام خریداری شده توسط مشتری نگهداری می‌شود. در واقع این جدول مدلی برای یک نوع بانک اطلاعاتی است که آن را پایگاه داده‌ی تراکنشی^۲ می‌نامیم.

جدول ۱-۴: نمونه‌ای از یک پایگاه داده‌ی تراکنشی فروشگاه

TID	Items
1	{Bread, Milk}
2	{Bread, Egg, Butter, Cheese}
3	{Milk, Butter, Cheese, Chicken}
4	{Bread, Cheese, Chicken}
5	{Bread, Milk, Butter, Cheese}

در جدول ۱-۴ خرید ۵ مشتری نشان داده شده است. برای مثال مشتری که با مشخصه‌ی تراکنش ۱ شناخته می‌شود، نان و شیر را خریده است، یا به عبارتی دیگر این دو قلم جنس را در سبد خرید خود دارد. بسیاری از مدیران فروشگاه‌ها علاقه‌مند هستند تا رفتارهای (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 إعمال تغییراتی برای اقلام مزبور، فروش خود را بهبود بخشد. مجموعه اقلامی که شامل k قلم داده

¹ Market Basket Analysis

² Transactional Database

است را مجموعه اقلام k -تایی^۱ می‌نامند. برای مثال با توجه به محتويات جدول ۴-۱ مجموعه $\{Milk, Egg, Butter\}$ یک مجموعه اقلام ۳-تایی است. یک خصوصیت مهم برای مجموعه اقلام، پشتیبان^۲ نامیده می‌شود که برابر است با تعداد تراکنش‌هایی که شامل مجموعه اقلام مذبور است و بطور معمول بصورت درصدی از تراکنش‌ها مجموعه اقلام را شامل می‌شوند، بیان می‌گردد. برای مثال مقدار پشتیبان برای مجموعه اقلام ۲-تایی $\{Butter, Cheese\}$ برابر با ۳ است. زیرا فقط ۳ تراکنش از میان ۵ تراکنش هستند که شما می‌توانید این دو قلم داده را با یکدیگر در آنها ببینید. بنابراین می‌توان گفت پشتیبان این مجموعه اقلام ۲-تایی برابر با ۶۰ درصد است.

محاسبه‌ی مقدار پشتیبان برای کلیه‌ی اقلام داده‌ها، فضای جستجوی بسیار گسترده‌ای را ایجاد می‌کند و در نتیجه این عمل مستلزم محاسبات سنگینی خواهد بود. در جدول ۴-۱ که فقط از ۶ قلم داده استفاده شده است، مقدار پشتیبان برای کلیه‌ی زیرمجموعه‌های این ۶ داده به جز تهی یعنی $=6^3=63$ ^۳ باید محاسبه شود. حال تصور کنید در یک فروشگاه صدها یا هزاران قلم داده (جنس) وجود دارد. بدون شک در چنین موقعیتی بررسی بر روی تمام مجموعه اقلام داده‌ها کار چندان ساده‌ای نیست. نکته‌ی مهم دیگر اینکه بسیاری از این مجموعه اقلام برای کاربر مفید نیستند. بدین علت ما به جستجوی مجموعه اقلامی هستیم که برای کاربر جالب باشند. اما جالب یا مفید بودن چگونه تعریف می‌شود. اولین معیار همانطور که قبل از این نیز نامی از آن برده شد، معیار پشتیبان است. در میان این فضای جستجوی وسیع، بدنیال مجموعه اقلامی خواهیم بود که مقدار پشتیبان آنها از مقدار حداقلی ($minsup$) که توسط کاربر تعریف شده است، بیشتر باشد. چنین مجموعه اقلامی، مجموعه اقلام مکرر نامیده می‌شوند که در ادامه از آنها به عنوان الگوهای مکرر^۴ نیز نام می‌بریم. بنابراین در اینجا بدنیال الگوریتم‌های مقیاس‌پذیری^۴ هستیم که به شناسایی مجموعه اقلام مکرر می‌پردازنند. برای مثال اگر مقدار $minsup$ برابر با ۶۰ درصد (یا مقدار ۳) در نظر گرفته شود، مجموعه اقلام ۳-تایی $\{Bread, Milk\}$ در

¹ K-itemsets

² Support

³ Frequent Pattern

⁴ Scalable

جدول ۴-۱ یک الگوی مکرر نیست، در حالیکه مجموعه اقلام ۲تاًی کرده و پنیر یعنی $\{Butter, Cheese\}$ یک مجموعه اقلام مکرر محسوب می‌شود.

۴-۲) قوانین انجمنی

می‌خواهیم وابستگی‌های مهم میان اقلام موجود در یک پایگاه داده‌ی تراکنشی را مشخص کنیم، 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 پنیر و کره را در سبد خرید خود دارند. برای پاسخ به این نمونه از سوال‌ها ابتدا به تعریف قوانین انجمنی می‌پردازیم 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 انجمنی فوق با دو معیار پشتیبانی و اطمینان^۱ اندازه‌گیری شده‌است که در ادامه آنها را شرح خواهیم داد. اما برای شروع بهتر است بدانید تفسیر قانون انجمنی مذبور بدین گونه است که ۴۰ درصد از تراکنش‌ها حاوی دو قلم جنس شیر و نان هستند و ۶۶ درصد از مشتریانی که شیر خریده‌اند، نان را نیز در سبد خرید خود دارند. همانند مجموعه اقلام مکرر، قوانین انجمنی مفید به آن دسته از قوانین اطلاق می‌شود که مقدار پشتیبان و همچنین اطمینان آن از حداقل مقدار تعریف شده توسط کاربر بیشتر باشد.

یک قانون انجمنی با عبارت $X \rightarrow Y$ بیان می‌شود که در آن X و Y مجموعه اقلام غیرتهی هستند که هیچ‌گونه اشتراکی ندارند ($X \cap Y = \emptyset$). دو معیار پشتیبانی 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

^۱ Confidence

تراکنش‌هایی که مجموعه اقلام X را در خود دارند، بدنبال مجموعه اقلام Y می‌گردد. این نکته ساده را فراموش نکنید که تراکنش‌های حاوی X می‌تواند شامل Y نباشد و بالعکس. بدین سبب دو قانون انجمنی $Y \rightarrow X$ و $X \rightarrow Y$ یکسان نیستند. از زاویه‌ی دید تئوری احتمالات برای پشتیبان و اطمینان داریم:

$$\text{Support } (X \rightarrow Y) = P(X \cap Y)$$

$$\text{Confidence } (X \rightarrow Y) = P(Y/X) = P(X \cap Y)/P(X)$$

به جدول ۴-۲ توجه کنید. برای سادگی هر قلم داده با حرف I همراه با یک زیرنویس نشان داده شده است. مجموعه اقلام نیز از مجموعه‌ی $\{I1, I2, I3, I4, I5\}$ انتخاب شده‌اند.

جدول ۴-۲: نمونه‌ای از یک پایگاه داده با ۴ تراکنش

TID	Items
1	$\{I1, I3, I4\}$
2	$\{I2, I3, I5\}$
3	$\{I1, I2, I3, I5\}$
4	$\{I2, I5\}$

چند قانون انجمنی همراه با مقادیر پشتیبان و اطمینان آنها عبارتند از:

$$R1: \{I1\} \rightarrow \{I3\} \quad [\text{Support}=50\%, \text{Confidence}=100\%]$$

$$R2: \{I1\} \rightarrow \{I4\} \quad [\text{Support}=25\%, \text{Confidence}=50\%]$$

$$R3: \{I2, I3\} \rightarrow \{I5\} \quad [\text{Support}=50\%, \text{Confidence}=100\%]$$

$$R4: \{I2\} \rightarrow \{I3, I5\} \quad [\text{Support}=50\%, \text{Confidence}=66\%]$$

چنانچه مقدار $minsup$ برابر با ۵۰ درصد و حداقل مقدار برای اطمینان ($minconf$) ۷۵ درصد تنظیم شوند، فقط قوانین $R1$ و $R3$ به عنوان قوانین قوی^۱ انتخاب می‌شوند.

¹ Strong Rules

قانونی با مقدار پشتیبان پایین، کمتر مورد توجه مدیر یک فروشگاه قرار می‌گیرد. زیرا مدیر علاقه‌ای به تمرکز بر روی اجنبایی که به ندرت به فروش رسیده‌اند را ندارد (البته استثنایی هستند که در فصل بعدی به آن می‌پردازیم). از طرف دیگر مقدار اطمینان نشان دهنده‌ی درصد واستگی دو مجموعه اقلام در دو طرف قانون انجمنی است. بدین سبب جهت انتخاب و ارزشیابی قوانین از این دو معیار استفاده می‌کنیم. اما تعیین یک مقدار مناسب برای $minconf$ و همچنین $minsup$ ساده نیست. انتخاب مقدار کوچک برای این دو پارامتر باعث می‌شود که تعداد قوانین بدست آمده بسیار زیاد و در نتیجه تعداد قوانین غیرمفید نیز افزایش یابد. در مقابل با یک مقدار بالا برای آنها، امکان از دست دادن قوانین مفید وجود خواهد داشت.

حال فرض کنید می‌خواهیم به جستجوی کلیه‌ی قوانین انجمنی یک پایگاه داده و همچنین محاسبه‌ی مقدار پشتیبان و اطمینان آنها بپردازیم. بدلیل وجود تعداد بسیار زیاد این قوانین، عدم ارائه یک الگوریتم مناسب و کارا می‌تواند بصورت کمرشکنی برای ما گران تمام شود. اگر در پایگاه‌داده‌ها از k قلم داده استفاده شود، تعداد قوانین انجمنی در این پایگاه داده برابر است با:

$$N_{Rule} = 3^k - 2^{k+1} + 1$$

حتی برای مقادیر کوچک k تعداد قانون بصورت توانی رشد می‌کند. برای مثال فقط با ۵ قلم داده، ۱۸۰ قانون انجمنی تولید می‌شود که با تنظیم مقدار $minconf$ و $minsup$ اغلب آنها می‌توانند حذف شوند. به همین دلیل برای جلوگیری از محاسبات بیهوده بهتر است برخی از قوانین بدون محاسبه‌ی مقدار پشتیبان و اطمینان هرس شوند. برای مثال قانون انجمنی زیر را در نظر بگیرید:

$$\{I1\} \rightarrow \{I2, I3\}$$

قوانين دیگری نیز وجود دارند که همانند قانون مذبور فقط از اقلام $I2$, $I1$ و $I3$ استفاده می‌کنند، مانند:

$$\{I1, I2\} \rightarrow \{I3\}, \{I1, I3\} \rightarrow \{I2\}, \{I2\} \rightarrow \{I1, I3\}, \{I3\} \rightarrow \{I1, I2\}$$

محاسبه‌ی مقدار پشتیبان برای تمام قوانین فوق یکسان است. فقط کافی است درصد تراکنش‌هایی که شامل مجموعه اقلام ۳تایی $\{I1, I2, I3\}$ است را بدست آورید.

نکته حائز اهمیت اینجاست که اگر مقدار پشتیبان محاسبه شده برای فقط یکی از قوانین فوق کمتر از $minsup$ باشد، دیگر حتی لازم نیست قوانین دیگر تولید شوند. این موضوع می‌تواند به عنوان یک قانون ساده جهت هرس کردن قوانین انجمنی و همچنین کاهش محاسبات استفاده شود. راه حل‌های دیگری نیز وجود دارند که در بخش‌های بعدی برخی از آنها بررسی می‌شوند.

از دیدگاه کلی کاوش قوانین انجمنی را می‌توان یک فرایند دو مرحله‌ای در نظر گرفت:

- یافتن کلیه‌ی مجموعه اقلام مکرر
- تولید قوانین انجمنی حائز شرایط (قوی) با کمک مجموعه اقلام مکرر پیدا شده در مرحله‌ی قبل

به دلیل هزینه‌ی بالای محاسباتی در مرحله‌ی اول، بطور معمول الگوریتم‌ها بر روی بهینه‌سازی عملیات این مرحله متمرکز می‌شوند. چرا که کارایی الگوریتم با توجه به پیچیدگی این مرحله سنجیده می‌شود.

۳-۴) تولید الگوهای مکرر

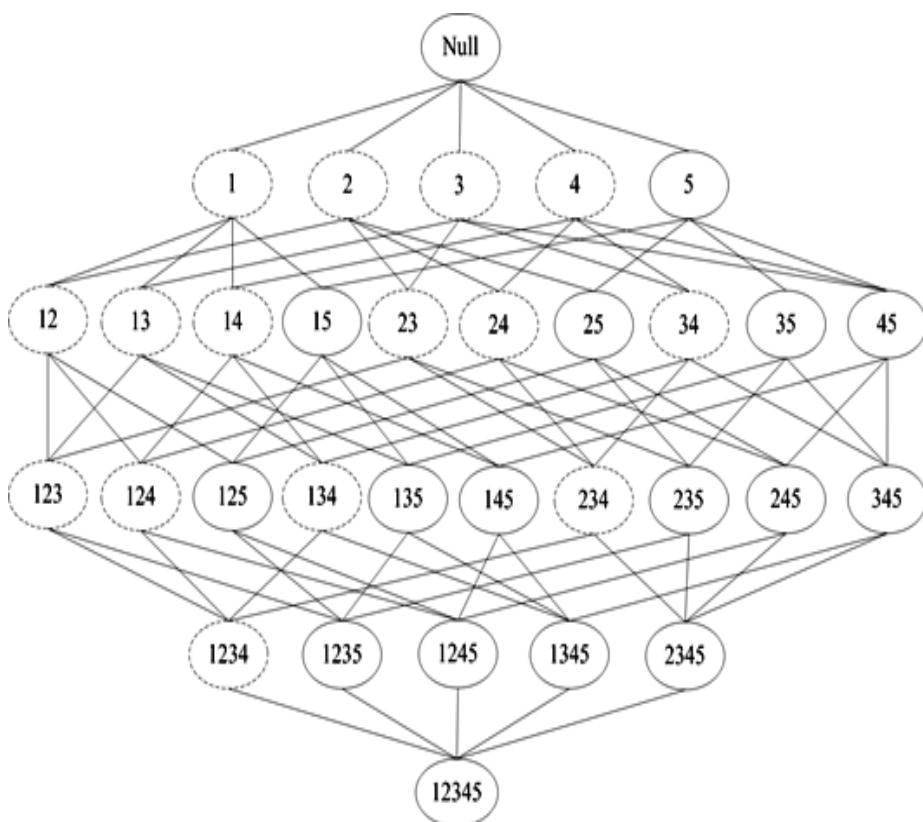
همانطور که قبل از این نیز اشاره شد، تولید مجموعه اقلام مکرر مرحله‌ی مهمی از کشف قوانین انجمنی را تشکیل می‌دهد. مجموعه داده‌هایی که شامل k قلم داده باشد، می‌تواند تعداد $2^k - 1$ الگو تولید کند که هر یک از آنها می‌تواند یک الگوی مکرر باشد. با افزایش مقدار k فضای جستجو نیز بصورت نمایی رشد می‌کند. راهکاری که از هیچ قانون میانبر و یا هیوریستیکی استفاده نمی‌کند، نمی‌تواند در زمان قابل قبولی کار خود را در این فضای جستجو به پایان برساند. زیرا پس از تولید این $2^k - 1$ مجموعه اقلام، برای محاسبه مقدار پشتیبان هر یک از آنها پایگاه داده‌ها پیمایش و برای هر سطر آن وجود مجموعه اقلام بررسی می‌شوند. بدین ترتیب اگر تعداد تراکنش‌ها N باشد و بطور میانگین هر تراکنش دارای فقط تعداد A قلم داده باشد، الگوریتم با تعداد $(2^k - 1) \times N \times A$ مقایسه روبرو خواهد بود. بنابراین نیاز به روش‌هایی که از حجم این محاسبات بکاهد به وضوح احساس می‌شود. در ادامه برخی از الگوریتم‌های تولید مجموعه اقلام مکرر بیان می‌شوند.

۱-۳-۴) الگوریتم *Apriori*

این الگوریتم از اولین الگوریتم‌هایی است که جهت یافتن مجموعه اقلام مکرر از آن استفاده می‌شود. نام آن برگرفته از شیوه‌ای است که از آن استفاده می‌کند، یعنی استفاده از دانش مرحله‌ی قبل که در ادامه آن را شرح می‌دهیم. الگوریتم *Apriori* یک الگوریتم جستجوی سطحی است، که با پایان کاوش در سطح (مرحله‌ی) k ام به مرحله‌ی بعدی یعنی $k+1$ می‌رود. این عمل تا محقق شدن شرط یا شروط پایانی تکرار می‌شود. در مرحله‌ی k ام مجموعه اقلام k تایی تولید خواهند شد. پس از محاسبه‌ی مقدار پشتیبان برای هر کدام و مقایسه‌ی آن با مقدار $minsup$ الگوهای مکرر k تایی شناسایی می‌شوند. در مرحله‌ی بعدی الگوریتم با کمک الگوهای مکرر k تایی، مجموعه اقلام $(k+1)$ تایی کاندید که بالقوه می‌توانند مکرر باشند را تولید می‌کند. به همین ترتیب با توجه به مقدار برخی حذف شده و مجموعه اقلام مکرر $(k+1)$ تایی تشکیل خواهند شد. این عمل تا یافتن آخرین مجموعه قلم مکرر ادامه پیدا می‌کند.

این الگوریتم در حین اجرا از قاعده‌ای موسوم به قاعده‌ی *Apriori* استفاده می‌کند که بدین صورت بیان می‌شود: "اگر یک الگوی مکرر داشته باشیم، کلیه‌ی زیرمجموعه‌های آن نیز مکرر هستند." به عبارت دیگر اگر مجموعه اقلام I مکرر نباشد، هر مجموعه‌ای که شامل I است نیز نمی‌تواند مکرر باشد.

با کمک قاعده‌ی *Apriori* فضای جستجو کاهش می‌یابد. شکل ۱-۴ کل فضای جستجو را برای مجموعه اقلامی که از $\{1,2,3,4,5\}$ استفاده کرده است، نشان می‌دهد. برای سادگی الگوها به شکل یک عدد نمایش داده شده است. برای مثال الگوی $\{1,2,3\}$ را بصورت عدد 123 نشان داده‌ایم. دقت کنید که در شکل الگوهای مکرر با دایره‌های خطچین مشخص شده‌اند. با نگاه به الگوی مکرر $\{1,2,3,4\}$ که در شکل 1234 نشان داده‌ایم، متوجه خواهید شد که همه‌ی زیرمجموعه‌های آن نیز مکرر هستند. یا اینکه کلیه‌ی مجموعه اقلامی که شامل $\{5\}$ هستند، نمی‌توانند مکرر باشند. چون $\{5\}$ مکرر نیست. بدین ترتیب استفاده از این استراتژی باعث کاهش فضای جستجو در تولید اقلام مکرر می‌شود.



شکل ۱-۴: فضای جستجو برای یک مجموعه‌ای با ۵ قلم داده

قاعده‌ی *Apriori* به گروه خاصی از قواعد که دارای خاصیت پادیکنواختی^۱ هستند، تعلق دارد. این خاصیت بصورت خلاصه بدین صورت بیان می‌شود که اگر مجموعه‌ای نتواند در یک آزمون موفق باشد، کلیه‌ی آبرمجموعه‌های^۲ آن نیز در همان آزمون با شکست مواجه می‌شوند.

فرض کنید J می‌تواند هر نمونه‌ای از مجموعه اقلامی باشد که از مجموعه‌ی I منتج می‌شود. یک مقیاس f دارای خاصیت یکنواختی^۳ است اگر:

$$\forall X, Y \in J : (X \subseteq Y) \Rightarrow f(X) \leq f(Y)$$

¹ Anti-monotone

² Supersets

³ Monotone

که نشان می‌دهد اگر X زیرمجموعه‌ی Y باشد، بنابراین $f(X)$ باید بزرگتر از $f(Y)$ باشد.
در مقابل f دارای خاصیت پادیکنواختی است اگر:

$$\forall X, Y \in J : (X \subseteq Y) \Rightarrow f(Y) \leq f(X)$$

هر مقیاس و قاعده‌ای همانند *Apriori* که دارای خاصیت پادیکنواختی است، می‌تواند برای الگوریتم‌های داده‌کاوی مثل تولید مجموعه اقلام مکرر موثر باشد.

جدول ۴-۳ یک پایگاه داده تراکنشی با ۵ تراکنش و ۱۱ قلم داده را نشان می‌دهد. با تنظیم مقدار *minsup* برابر با ۶۰ درصد و با کمک قاعده *Apriori* می‌خواهیم الگوهای مکرر را در این پایگاه داده تولید کنیم.

جدول ۴-۳: یک پایگاه داده‌ی تراکنشی با ۵ تراکنش و ۱۱ قلم داده

TID	Items
1	{I1,I2,I3,I4,I5,I6}
2	{I2,I3,I4,I5,I6,I7}
3	{I1,I4,I5,I8}
4	{I1,I4,I6,I9,I10}
5	{I2,I4,I5,I10,I11}

با پیمایش پایگاه‌داده‌ها و با توجه به مقدار *minsup* که برابر با ۶۰ درصد (معادل ۳ تکرار از میان ۵ تراکنش) است، الگوهای مکرر اتایی یا یک عضوی بدست می‌آیند (شکل ۴-۲).

همانطور که از شکل ۴-۲ مشخص است از میان ۱۱ قلم داده فقط ۵ قلم مکرر هستند، که در ستون سوم جدول علامت خورده‌اند. در مرحله‌ی دوم با کمک این مجموعه اقلام مکرر و الحقیقی^۱ آنها مجموعه اقلام کاندیدی تولید خواهند شد که می‌توانند بالقوه مکرر باشند (شکل ۴-۳).

^۱ Join

1-itemsets	Support	Frequent 1-itemset
$I1$	3 (60%)	✓
$I2$	3 (60%)	✓
$I3$	2 (40%)	—
$I4$	5 (100%)	✓
$I5$	4 (80%)	✓
$I6$	3 (60%)	✓
$I7$	1 (20%)	—
$I8$	1 (20%)	—
$I9$	1 (20%)	—
$I10$	2 (40%)	—
$I11$	1 (20%)	—

شکل ۲-۴: کلیه‌ی مجموعه اقلام ۱تایی و شناسایی الگوهای مکرر

چنانچه الگوریتم بدون توجه به اقلام مکرر ۱تایی قصد تولید اقلام ۲تایی را داشت، باید ۵۵ مجموعه قلم ۲تایی را ایجاد می‌کرد.

2-itemsets	Support	Frequent 2-itemset
$\{I1, I2\}$	1 (20%)	—
$\{I1, I4\}$	3 (60%)	✓
$\{I1, I5\}$	2 (40%)	—
$\{I1, I6\}$	2 (40%)	—
$\{I2, I4\}$	3 (60%)	✓
$\{I2, I5\}$	3 (60%)	✓
$\{I2, I6\}$	2 (40%)	—
$\{I4, I5\}$	4 (80%)	✓
$\{I4, I6\}$	3 (60%)	✓
$\{I5, I6\}$	2 (40%)	—

شکل ۳-۴: برخی از مجموعه اقلام ۲تایی کандید و شناسایی الگوهای مکرر

این در حالی است که با کمک الگوهای مکرر ۱تایی فقط تعداد ۱۰ مجموعه اقلام ۲تایی ساخته شده است و این نکته کاهش فضای جستجو را نشان می‌دهد.

در این مرحله نیز پایگاهداده برای محاسبه‌ی مقدار پشتیبان مجموعه اقلام ۲تایی موجود در شکل ۴-۳ پیمایش می‌شود. پس از حذف مجموعه اقلام ۲تایی که مقدار پشتیبان آنها از حد آستانه (مقدار ۳) کمتر است، مجموعه اقلام مکرر شناسایی می‌شوند. بعد از این با الحاق الگوهای مکرر ۲تایی باید مجموعه اقلام ۳تایی که مستعد مکرر بودن هستند، تولید شوند. دو نکته‌ی مهم در پیاده‌سازی این مرحله به بعد باید رعایت شود. در این مرحله شما مجاز به الحاق دو الگوی ۲تایی هستید که نتیجه یک مجموعه اقلام ۳تایی باشد. برای مثال با پیوند الگوهای مکرر $\{I1, I4\}$ و $\{I2, I4\}$ به مجموعه اقلام ۳تایی $\{I1, I2, I4\}$ می‌رسیم. در حالیکه با پیوند دادن $\{I1, I4\}$ و $\{I2, I5\}$ یک مجموعه قلم ۴تایی $\{I1, I2, I4, I5\}$ تولید می‌شود. فراموش نکنید که ترتیب قرار گرفتن اقلام در الگو مهم نیستند. نکته‌ی دوم اینکه در حین ایجاد مجموعه اقلام ۳تایی از قانون استفاده می‌کنیم. باید مجموعه اقلامی تولید شود که تمام زیرمجموعه‌های آن مکرر هستند. برای مثال از الحاق $\{I2, I4\}$ و $\{I4, I6\}$ که هر دوی آنها مکرر هستند، مجموعه $\{I2, I4, I6\}$ بدست می‌آید. اما از آنجا که الگوی $\{I2, I6\}$ مکرر نیست، بدون محاسبه‌ی مقدار پشتیبان می‌توان فهمید که $\{I2, I4, I6\}$ نیز نمی‌تواند مکرر باشد.

راه حل ساده‌ی دیگر برای تولید مجموعه اقلام کاندید با طول ۳، الحاق الگوهای مکرر ۲تایی و الگوهای مکرر ۱تایی است. این کار می‌تواند برای ساخت کاندیداهایی با طول بالاتر نیز استفاده شود، به نحوی که جهت ساخت کاندیدی با طول n کافی است الگوهای مکرر $(n-1)$ تایی با الگوهای مکرر ۱تایی ترکیب شوند. شکل ۴-۴ نتایج مرحله‌ی سوم الگوریتم را نشان می‌دهد. در شکل کلیه‌ی مجموعه اقلام ۳تایی که از پیوند الگوهای مکرر ۲تایی بدست آمده‌اند، نشان داده شده است. به جز برای $\{I2, I4, I5\}$ لازم نیست الگوریتم مقدار پشتیبان را برای دیگر مجموعه‌ها محاسبه کند. زیرا برای هر یک از آنها حداقل یک زیرمجموعه وجود دارد که مکرر نیست. به همین دلیل لزومی ندارد مقدار پشتیبان برای آنها محاسبه شود و با استناد به قاعده‌ی *Apriori* کنار گذارده می‌شوند و نیازی به پیمایش داده‌ها نیست.

پس از این مرحله مجموعه اقلام بزرگتری یافت نمی‌شود تا الگوریتم به کار خود ادامه دهد، لذا الگوریتم متوقف می‌شود. بنابراین بزرگترین الگوی مکرر برای مثال مذبور برابر با ۳ خواهد بود.

3-itemsets	Support	Frequent 3-itemset
$\{I1, I2, I4\}$	—	—
$\{I1, I4, I5\}$	—	—
$\{I1, I4, I6\}$	—	—
$\{I2, I4, I5\}$	3 (60%)	✓
$\{I2, I4, I6\}$	—	—
$\{I4, I5, I6\}$	—	—

شکل ۴-۴: برخی از مجموعه اقلام ۳تایی کاندید و شناسایی الگوهای مکرر

توجه کنید در این مثال ساده با کمک قانون *Apriori* فقط تعداد $(11+10+6)=27$ مجموعه اقلام تولید شد. در حالیکه اگر از این قانون استفاده نمی‌شد، مجبور به تولید

$$\binom{11}{1} + \binom{11}{2} + \binom{11}{3} = 231$$

مثال ساده نشان می‌دهد که چگونه فضای جستجو با کمک این قانون می‌تواند بطور قابل ملاحظه‌ای کاهش یابد.

الگوریتم *Apriori* در هر مرحله دو عملیات انجام می‌دهد. ابتدا الگوریتم با الحاق الگوهای مکرر k تایی، به تولید مجموعه اقلام $(k+1)$ تایی می‌پردازد. همانطور که قبل از این نیز گفته شد، در این مرحله می‌توان برای ساخت مجموعه اقلام $(k+1)$ تایی، هر الگوی مکرر k تایی را با الگوهای مکرر 1 تایی الحاق نمود. سپس با کمک قاعده‌ی *Apriori* برخی از مجموعه اقلام حذف و با محاسبه‌ی مقدار پشتیبان برای مابقی، الگوهای مکرر تشخیص داده می‌شوند.

بهبود کارایی الگوریتم *Apriori*

هرچند فضای جستجو در الگوریتم *Apriori* بسیار کمتر از کاوش در میان کلیه‌ی حالات است، اما در برنامه‌های کاربردی از کارایی خوبی برخوردار نیست. پیشنهادهای زیادی جهت بهبود این الگوریتم ارائه شده است که در ادامه به صورت کلی و مختصر برخی از آنها بررسی می‌شوند.

- بسیاری از روش‌ها با کمک یک ساختمان داده‌های مناسب عملکرد بهتری از خود نشان می‌دهند. استفاده از یک تابع درهم‌ساز^۱ و ذخیره‌ی مجموعه اقلام k -تایی ($k > 1$) در باکت‌های^۲ یک فایل با ساختار مستقیم^۳ می‌تواند سرعت دسترسی به داده‌ها را افزایش دهد. در ضمن با نگهداری مقدار پشتیبان برای هر باکت، می‌توان شناسایی الگوهای مکرر را تسهیل نمود.
- بدون شک تراکنشی که حاوی هیچ الگوی مکرر k -تایی نیست، نمی‌تواند شامل الگوی مکرر بزرگتر از k باشد. بدین ترتیب نیازی نیست در مراحل بعدی اجرای الگوریتم این تراکنش بازبینی شود. لذا با علامت زدن و یا حذف این تراکنش برای مراحل بعدی، می‌توان سرعت جستجو را افزایش داد.
- تقسیم داده‌ها به قسمت‌های مجزای کوچک‌تر تکنیک دیگری است که می‌تواند در بهبود الگوریتم *Apriori* نقش مهمی ایفا کند. این عمل در دو مرحله انجام می‌شود. در مرحله‌ی اول داده‌ها به n قسمت مجزا و بدون همپوشانی داده‌ها تقسیم می‌شوند. تعداد و یا اندازه‌ی بخش‌ها می‌توانند با توجه به ظرفیت حافظه‌ی اصلی کامپیوتر مشخص شوند. الگوهای مکرر برای هر قسمت بصورت مجزا استخراج می‌شوند. اگر مقدار پشتیبان برای داده‌های اصلی برابر با $minsup$ تنظیم شده باشد، مقدار پشتیبان برای هر قسمت باید برابر با $minsup/n$ در نظر گرفته شود. بنابراین در ابتدا الگوهای مکرر محلی^۴ با مقدار جدید $minsup$ شناسایی می‌شوند. سپس در مرحله‌ی بعدی با جمع‌آوری کلیه‌ی الگوهای مکرر

¹ Hashing Function

² Buckets

³ Direct

⁴ Local

محلی و با یک پیمایش از کل پایگاه داده‌ها الگوهای مکرر در داده‌های اولیه بدست می‌آیند. از آنجا که با مقدار جدید پشتیبان هر الگوی مکرر در کل داده‌ها، باید حداقل در یکی از قسمت‌ها به عنوان الگوی مکرر محلی شناخته شده باشد، لذا عملکرد صحیح روش تضمین می‌شود. در ضمن این روش بسیار مناسب برای پیاده‌سازی در یک محیط موازی است.

- با افزایش حجم پایگاه‌داده‌ها تکنیک‌های نمونه‌گیری^۱ می‌تواند روش مناسبی برای کاهش ابعاد داده‌ها و در نتیجه بهبود کارایی الگوریتم باشد. روش‌های نمونه‌گیری که در فصل دوم نیز اشاره‌ای به آنها شد، از تنوع زیادی برخوردارند و هر یک با درجه‌ی دقت متفاوتی برای کاربردهای مختلف عمل می‌کنند. همانطور که قبلاً از این نیز بیان شد، استفاده از این تکنیک‌های نمونه‌گیری محدود به الگوریتم *Apriori* نیست. تعداد نمونه‌ها و چگونگی انتخاب آنها از پارامترهای مهم در این روش‌ها به شمار می‌روند.
- در الگوریتم *Apriori* اولیه، از یک استراتژی سطح به سطح^۲ استفاده می‌شود. به عبارت دیگر در مرحله *l*ام با توجه به داشتن مرحله‌ی قبلی مجموعه اقلام *i*تایی تولید و برای مکرر بودن بررسی می‌شوند. امروزه روش‌های پویایی جهت تعمیم الگوریتم ارائه شده است که در آنها حین پیمایش پایگاه‌داده‌ها مجموعه اقلام مستعدی(با هر اندازه) را به مجموعه جواب اضافه می‌کنند. بدین ترتیب در هر مرحله یا در هر نقطه‌ای از اجرای الگوریتم، اگر مکرر بودن یک مجموعه قلم مشخص شود، بدون توجه به طول آن، الگوریتم آنرا انتخاب می‌کند. بنابراین انتظار می‌رود که با استفاده از این روش، پیمایش کمتری از داده‌ها را به دنبال داشته باشیم.

¹ Sampling

² Level by Level

۴-۳-۲ الگوریتم FP-Growth

هر چند الگوریتم *Apriori* یک هیوریستیک مناسب برای تولید مجموعه اقلام ارائه می‌کند، اما بهر حال در شرایطی خاص با پیچیدگی محاسباتی بالایی روبروست. برای مثال وقتی که تعداد اقلام زیاد و بسیاری از آنها نیز مکرر باشند، الگوریتم نیاز به تولید تعداد بسیار زیادی از مجموعه اقلام ۲تایی یا بالاتر خواهد داشت. یا اینکه با وجود فقط تعداد کمی از الگوهای مکرر با طول کوتاه، تعداد کاندیداهای بسیار زیادی باید تولید شود. برای مثال اگر مجموعه‌ی داده‌ها شامل فقط یک الگوی مکرر با طول ۱۰۰ باشد، الگوریتم حداقل باید به تولید $2^{100} \approx 10^30$ کاندید بپردازد. در برخی کاربردها نیز الگوریتم چندین بار پایگاه داده‌ها را پیمایش می‌کند که این عمل بسیار زمانبر خواهد بود و به ورودی و خروجی زیادی نیاز دارد. در این بخش به توضیح در مورد الگوریتمی می‌پردازیم که بدون تولید مجموعه اقلام کاندید و با کمک یک ساختمان داده‌های درختی الگوهای مکرر را شناسایی می‌کند. این الگوریتم که استراتژی تقسیم و غلبه^۱ را دنبال می‌کند، در ابتدا پایگاه داده‌ها را بصورت یک درخت موسوم به *FP-tree* تبدیل می‌کند. پس از آن بصورت مستقیم به استخراج الگوهای مکرر از این درخت می‌پردازد.

به طور خلاصه می‌توان گفت یک *FP-tree* نمایش فشرده‌ای از داده‌های پایگاه داده است. برای ساخت این درخت الگوریتم، اقلام تراکنش‌ها را یک به یک خوانده و بصورت یک مسیر بر روی درخت نگاشت می‌کند. از آنجا که معمولاً تراکنش‌ها از اقلام مشترکی استفاده می‌کنند، لذا مسیرهای تراکنش‌ها بر روی درخت همپوشانی دارند و به همین دلیل درخت نسبت به داده‌های اولیه فشرده‌تر خواهد بود. اجازه دهید روش را همراه با یک مثال شرح دهیم. جدول ۴-۴ مجموعه داده‌ایی مشتمل بر ۹ تراکنش و ۵ قلم داده را نشان می‌دهد.

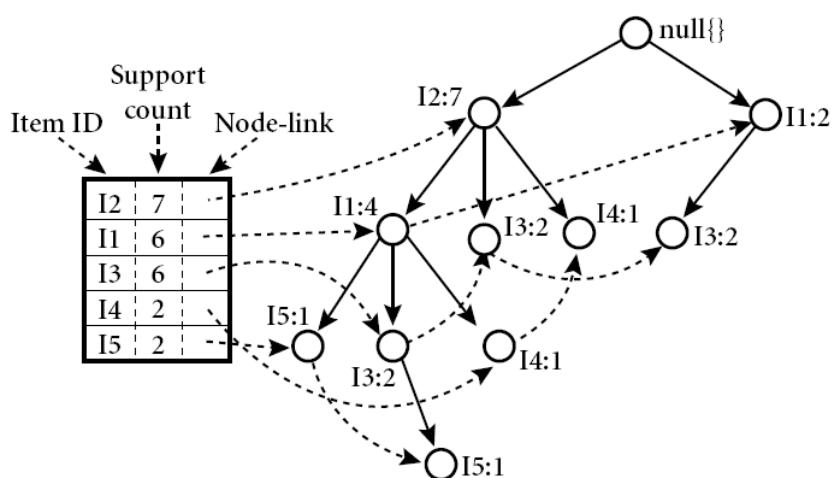
شکل نهایی درخت *FP-tree* با توجه به محتويات این جدول در شکل ۴-۵ نشان داده شده است.

^۱ Divide and Conquer

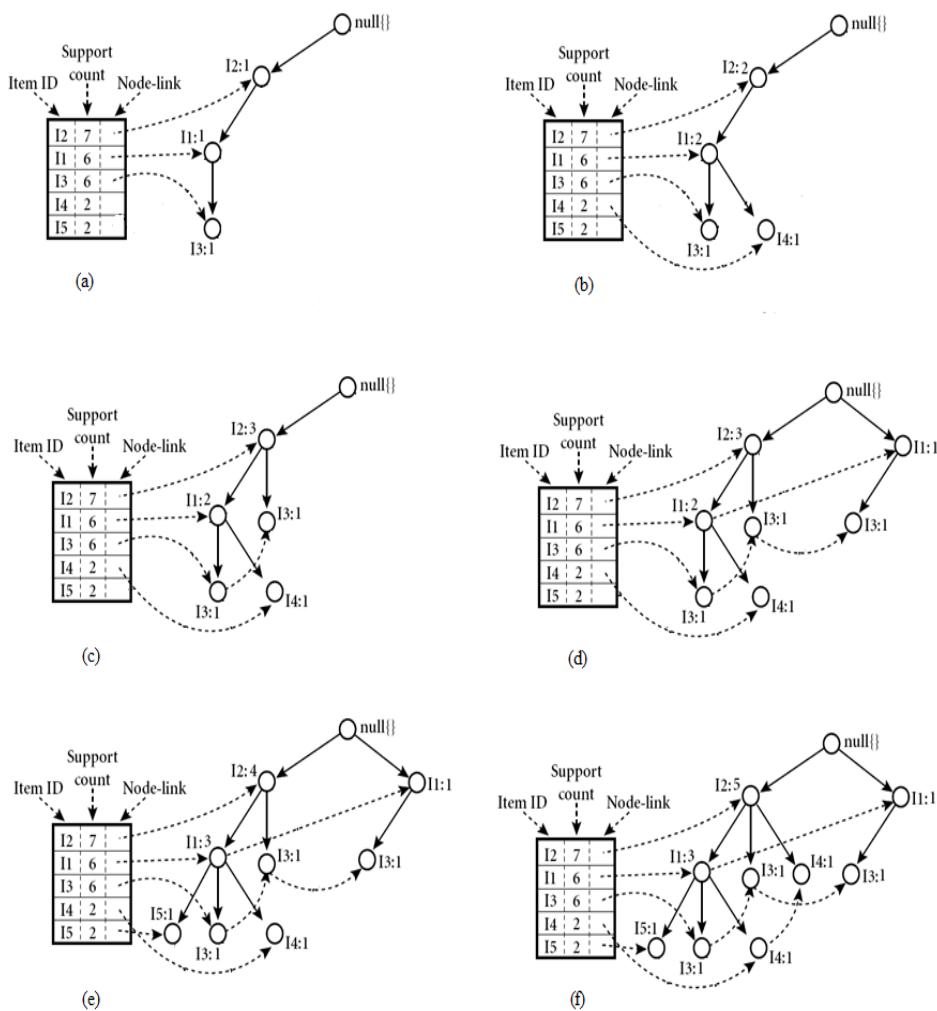
جدول ۴-۴: مجموعه داده‌های شامل ۹ تراکنش

TID	Items
1	$I1, I2, I3$
2	$I1, I2, I4$
3	$I2, I3$
4	$I1, I3$
5	$I1, I2, I5$
6	$I2, I4$
7	$I2, I3$
8	$I1, I3$
9	$I1, I2, I3, I5$

اما چگونه این ساختار درختی ساخته می‌شود؟ در ابتدا همانند الگوریتم *Apriori* بیمایش پایگاهداده‌ها مقدار پشتیبان را برای مجموعه اقلام ۱ تایی بدست می‌آوریم و از میان آنها الگوهای مکرر انتخاب می‌شوند. با مقدار ۲ برای $minsup$ هر ۵ قلم داده مکرر هستند و همانطور که در شکل ۴-۵ می‌بینید، هر یک از آنها همراه با مقدار پشتیبان و یک اشاره‌گر در جدولی (به ترتیب نزولی مقدار پشتیبان) قرار داده شده‌اند.

شکل ۴-۵: یک نمونه درخت *FP-tree*

اشاره‌گری که در جدول برای هر یک از اقلام وجود دارد، جهت پیمایش سریع‌تر و آسان‌تر درخت تعبیه شده است. هر اشاره‌گر شروع لیستی است که برای هر یک از اقلام به صورت مجزا تشکیل می‌شود. برای مثال در درخت نهایی تعداد ۳ گره را می‌توان یافت که دارای برچسب $I3$ هستند و هر سه توسط یک لیست پیوندی (تصویر خطچین) به یکدیگر متصل شده‌اند. شکل ۴-۶ شش مرحله از ساخت FP -tree مذکور را با خواندن ۶ تراکنش اول از جدول ۴ نشان می‌دهد.

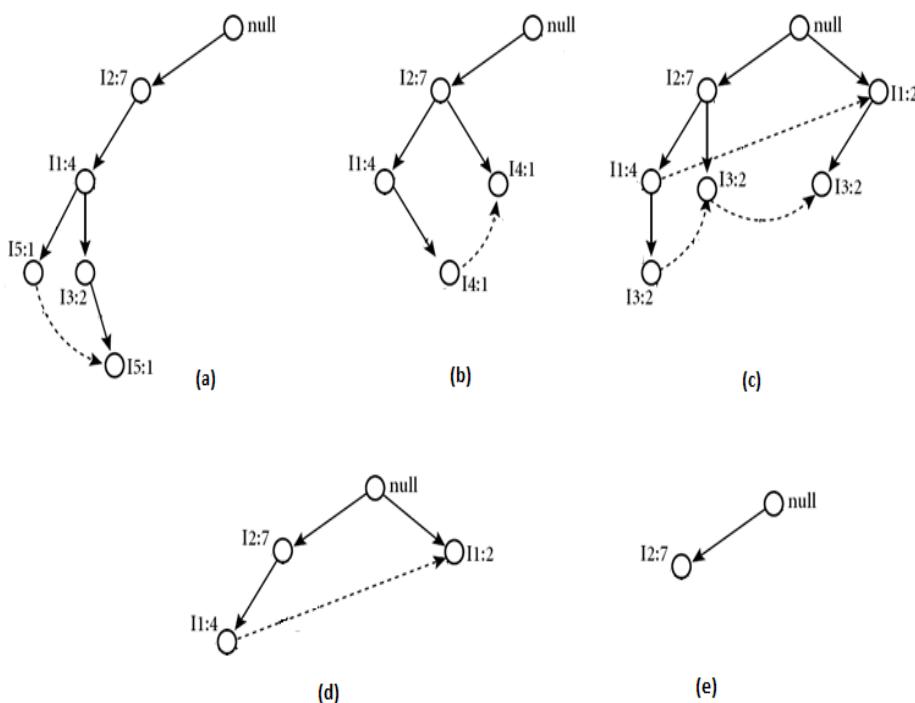


شکل ۴-۶: مراحل ساخت و تکمیل درخت FP -tree

در هر مرحله با خواندن اقلام هر یک از تراکنش‌ها، درخت تکمیل می‌شود. توجه کنید قبل از ساختن یک مسیر بر روی درخت، باید اقلام موجود در هر تراکنش را بر اساس مقدار پشتیبان، به ترتیب نزولی قرار دهید. برای مثال اقلام تراکنش اول که $\{I1, I2, I3\}$ هستند را بصورت $\{I1, I2, I3\}$ مرتب و به همین ترتیب بر روی درخت نگاشت می‌کنیم.

اغلب اندازه‌ی *FP-tree* ساخته شده از اندازه‌ی پایگاهداده‌ی اولیه کوچک‌تر است. در خوش‌بینانه‌ترین حالت که همه‌ی تراکنش‌ها از مجموعه‌ی یکسانی از اقلام استفاده کرده باشند، درخت دارای فقط یک شاخه است. بدترین حالت هم هنگامی رخ می‌دهد که تراکنش‌ها از اقلام متفاوت و منحصر به فردی تشکیل شده باشند. اندازه‌ی درخت در این حالت (با اغماص نسبت به حجم اشاره‌گرها) برابر با حجم داده‌های اولیه است. اندازه‌ی درخت به عواملی از جمله ترتیب قرار گرفتن اقلام (نزولی یا صعودی) بستگی دارد. البته نمی‌توان ادعا نمود که کدام ترتیب، درخت کوچک‌تری را ایجاد می‌کند.

حال نوبت به مرحله‌ای می‌رسد که الگوریتم با کمک درخت *FP-tree* الگوهای مکرر را استخراج کند. الگوهای مکرر اتایی که در مرحله‌ی اول الگوریتم بدست آمدند. در این مرحله برخلاف مرحله‌ی اول که مجموعه اقلام اتایی بصورت نزولی مرتب شدند، ما از قلم انتهایی لیست با کمترین مقدار پشتیبان (یعنی $I5$) شروع و به طرف اقلام بالای جدول حرکت می‌کنیم. از آنجا که هر تراکنش بصورت یک مسیر بر روی درخت نگاشت می‌شود، می‌توانیم به الگوهای مکرری که با قلم خاصی (مثل $I5$) ختم می‌شوند، دسترسی داشته باشیم. این عمل با دنبال کردن مسیرهایی که شامل گره‌ای با برچسب قلم داده‌ی خاص هست، میسر می‌شود. یادآوری می‌کنیم که با کمک اشاره‌گرها سریع‌تر به این مسیرها دسترسی دارید. مسیرهای استخراج شده از درخت برای هر یک از اقلام در شکل ۴-۷ نشان داده شده است. برای مثال قسمت a در شکل ۴-۷ مسیرهای منتهی به قلم $I5$ و قسمت d در همین شکل مسیرهای منتهی به $I1$ را نشان می‌دهد.



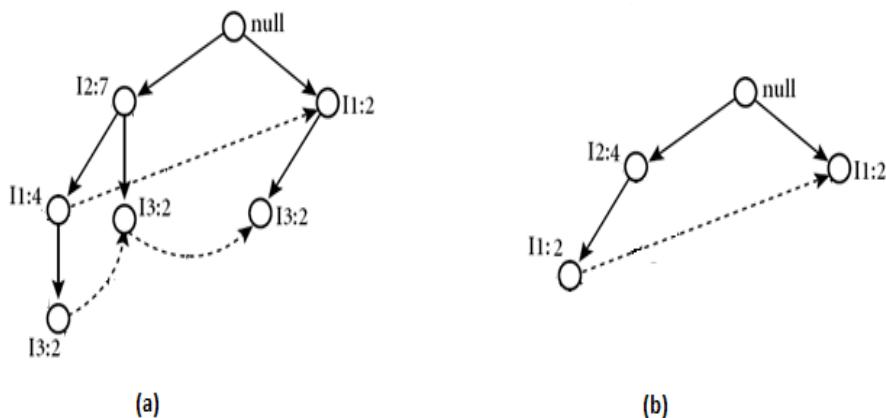
شکل ۴-۷: مسیرهای منتهی به هر یک اقلام داده‌ها

بدین ترتیب مسئله‌ی شناسایی الگوهای مکرر به چندین مسئله‌ی کوچکتر (۵ مسئله) تجزیه می‌شود. به صورتی که هر مسئله‌ی کوچکتر برای یافتن الگوهای مکرری که به I2, I1, I3, I4, I5 ختم می‌شوند، کوشش می‌کند. بدین ترتیب الگوریتم پس از یافتن الگوهای مکرر منتهی به I5 به سراغ مجموعه اقلام منتهی به I4 می‌رود. این کار همانند قبل با توجه به مسیرهای استخراج شده در شکل ۴-۷ برای اقلام دیگر نیز انجام می‌شود. حال فرض کنید I5 مکرر تشخیص داده شد. پس از آن الگوریتم به دنبال مجموعه اقلام مکرری است که در آن I5 به عنوان یک پسوند^۱ شناخته می‌شود. الگوریتم با پیروی از تکنیک تقسیم و غلبه این مسئله را به ۴ مسئله‌ی کوچکتر می‌شکند و به دنبال الگوهای مکرری خواهد گشت که دارای پسوندهای $\{I2, I5\}$, $\{I3, I5\}$, $\{I4, I5\}$ و $\{I1, I5\}$ هستند. برای این منظور الگوریتم از درختی موسوم به $FP\text{-}tree$ شرطی^۲ که بسیار شبیه به یک $FP\text{-}tree$ است، استفاده می‌کند. برای مثال از این درخت برای یافتن الگوهای

¹ Suffix² Conditional FP-tree

مکرری که دارای پسوند $\{I4, I5\}$ هستند، استفاده می‌شود. این کار برای مجموعه اقلام دیگر نیز تکرار می‌شود. شکل ۴-۸ یک $FP-tree$ شرطی برای درختی با مسیرهای منتهی به $I3$ را نشان می‌دهد.

در این قسمت اشاره به دو نکته ضروری است. اول اینکه درخت $FP-tree$ شرطی برای الگوهای مکرر ساخته می‌شوند و روشن است که اگر الگویی مکرر نباشد، آبرمجموعه‌های آن نیز مکرر نیستند. نکته‌ی دیگر اینکه پس از یافتن الگوهای مکرر منتهی به $\{I4, I5\}$ در واقع الگوهای مکرر منتهی به $\{I5, I4\}$ نیز بدست آمده‌اند. زیرا ترتیب قرار گرفتن اقلام در یک الگوی مکرر دارای اهمیت نیست.



شکل ۴-۸: یک $FP-tree$ شرطی (سمت راست) برای درختی با مسیرهای منتهی به $I3$ (سمت چپ) جدول ۴-۵ لیستی از الگوهای مکرر همراه با پسوندشان را نشان می‌دهد.

جدول ۴-۴: الگوهای مکرر همراه با پسوند آنها

Suffix	Frequent Itemsets
$I5$	$\{I5\}\{I2, I5\}\{I1, I5\}\{I2, I1, I5\}$
$I4$	$\{I4\}\{I2, I4\}$
$I3$	$\{I3\}\{I2, I3\}\{I1, I3\}\{I2, I1, I3\}$
$I1$	$\{I1\}\{I2, I1\}$
$I2$	$\{I2\}$

این مثال نشان می‌دهد الگوریتم *FP-Growth* که از روش تقسیم وغلبه در آن استفاده می‌شود، بدلیل آنکه مسائل کوچک‌تر بطور کامل مجزا هستند، نمی‌تواند مجموعه اقلام را بصورت تکراری تولید کند. به علاوه اعداد نگهداری شده در گره‌های درخت به ما کمک می‌کند که در حین تولید مجموعه اقلامی با پسوند یکسان، مقدار پشتیبان آنها نیز شمارش شوند. تحقیقات نشان داده است که این روش در شرایط مشخصی عملکرد قابل ملاحظه‌ای نسبت به روش *Apriori* اولیه دارد. شرایط مشخص به این معناست که پس از اجرای مرحله‌ای اول که ساختمان داده‌ی *FP-tree* ساخته می‌شود، درجه‌ی فشردگی این ساختار نسبت به داده‌های اولیه چه اندازه است. هر چه درخت‌های تولید شده توسط الگوریتم دارای شاخ و برگ بیشتری باشد، بنابراین مسائل به تعداد بیشتری از مسئله‌های کوچک‌تر شکسته خواهند شد و در نتیجه زمان اجرای الگوریتم افزایش می‌یابد.

۴-۴) تولید قوانین انجمنی

پس از اینکه با هر الگوریتم و یا هر روشی الگوهای مکرر بدست آمدند، نوبت به تولید قوانین انجمنی می‌رسد. برای مثال اگر $\{I1, I2, I3\}$ یک الگوی مکرر 3 تایی باشد، ۶ قانون انجمنی زیر می‌تواند از این الگوی مکرر بدست آید. مقدار پشتیبان برای همه آنها برابر است، اما باید مقدار اطمینان برای هر یک محاسبه شود.

$$\begin{array}{lll} \{I1\} \rightarrow \{I2, I3\} & , & \{I2\} \rightarrow \{I1, I3\} \\ \{I1, I2\} \rightarrow \{I3\} & , & \{I1, I3\} \rightarrow \{I2\} \end{array} \quad , \quad \begin{array}{lll} \{I3\} \rightarrow \{I1, I2\} \\ \{I2, I3\} \rightarrow \{I1\} \end{array}$$

برای هر الگوی k تایی مکرر Y می‌توان 2^{k-2} قانون انجمنی تولید کرد. از قوانینی که یک طرف آن تهی است صرفنظر می‌شود. یک قانون انجمنی با تقسیم Y به دو زیرمجموعه‌ی غیرتهی بدست می‌آید. به عبارت دیگر برای هر زیرمجموعه‌ی غیرتهی X از Y قانون انجمنی $X \rightarrow Y-X$ که مقدار اطمینان آن از حداقل تعریف شده بیشتر باشد، پذیرفته می‌شود. برای محاسبه‌ی مقدار اطمینان این قانون نیازی به پیمایش مجدد پایگاه داده نیست. برای مثال قانون انجمنی $\{I1, I2, I3, I4\} \rightarrow \{I1, I2\}$ که با توجه به الگوی مکرر $\{I1, I2, I3, I4\}$ تشکیل شده است را در نظر بگیرید. مقدار اطمینان این قانون از

تقسیم مقدار پشتیبان $\{I1, I2, I3, I4\}$ بر مقدار پشتیبان $\{II, I2\}$ بدست می‌آید. از آنجا که هر دوی آنها الگوهای مکرر هستند، پس مقادیر پشتیبان برای هر یک از آنها قبلاً محاسبه شده است.

برخلاف مقیاس پشتیبان، معیار اطمینان دارای خاصیت یکنواختی نیست. برای مثال اگر $X1$ زیرمجموعه‌ی X و $Y1$ زیرمجموعه‌ی Y باشد، هیچگونه ارتباطی میان مقادیر اطمینان دو قانون $X \rightarrow Y$ و $X1 \rightarrow Y1$ نمی‌توان یافت. با این وجود با کمی دقت به قوانین انجمنی تولید شده از یک الگوی مکرر Y و با توجه به مقدار اطمینان می‌توان منطقی جهت هرس کردن قوانین بدست آورد. چنانچه قانون انجمنی $X \rightarrow Y - X$ مقدار اطمینانی کمتر از $minconf$ داشته باشد، هر قانون انجمنی دیگر به شکل $X1 \rightarrow Y - X1$ که در آن $X1$ زیرمجموعه‌ی X باشد نیز نمی‌تواند مقدار اطمینانی بیشتر از $minconf$ داشته باشد. اثبات این موضوع بسیار ساده است. مقدار اطمینان برای هر دو قانون بصورت زیر محاسبه می‌شود:

$$\text{Confidence } (X \rightarrow Y - X) = \text{Support}(Y) / \text{Support}(X)$$

$$\text{Confidence } (X1 \rightarrow Y - X1) = \text{Support}(Y) / \text{Support}(X1)$$

صورت کسرها برابرند و بدون شک با توجه به اینکه $X1$ زیرمجموعه‌ی X است، داریم:
 $\text{Support}(X1) \geq \text{Support}(X)$

بنابراین مقدار اطمینان قانون دوم نمی‌تواند بیشتر از قانون انجمنی اول باشد. این موضوع می‌تواند از صرف زمان جهت آزمایش بیهوده‌ی برخی از قوانین انجمنی جلوگیری کند.

۵-۴) پایگاهداده تراکنشی و قالب‌های متفاوت داده‌ها

تاکنون جهت کاوش قوانین انجمنی از پایگاه داده‌ی تراکنشی استفاده می‌کردیم که مجموعه اقلام موجود در سبدهای خرید مشتریان از بارزترین مثال‌هایی است که در این زمینه می‌توان یافت. اما نکته‌ی مهم این است که می‌توان با کمی تغییر از شکل‌های دیگر داده‌ها به عنوان ورودی الگوریتم‌های کاوش قوانین انجمنی استفاده نمود.

مجموعه‌ای از استاد متنی مانند لیستی از ایمیل‌ها را در نظر بگیرید. هر یک از این اسناد می‌تواند متناظر با یک پایگاه داده‌ی تراکنشی در نظر گرفته شود. هر واژه‌ای که در سند وجود دارد به عنوان یک قلم داده تلقی می‌شود. پس از حذف کلمات تکراری در یک سند، داده‌ها به شکل پایگاه داده‌ی تراکنشی در می‌آیند. مشابه عملکرد فوق را می‌توانید بر روی یک جدول رابطه‌ای انجام دهید. شکل ۴-۹ تبدیل یک جدول با تعداد ۳ صفت خاصه را به یک مجموعه داده‌ی تراکنشی نشان می‌دهد.

Att1	Att2	Att3	TID	Items
<i>a</i>	<i>a</i>	<i>b</i>	1	(Att1, <i>a</i>),(Att2, <i>a</i>),(Att3, <i>b</i>)
<i>c</i>	<i>d</i>	<i>e</i>	2	(Att1, <i>c</i>),(Att2, <i>d</i>),(Att3, <i>e</i>)

شکل ۴-۹: تبدیل یک جدول در مدل رابطه‌ای (چپ) به یک پایگاه داده‌ی تراکنشی (راست) در پایگاه داده‌ی تراکنشی شکل ۴-۹ برای هر صفت خاصه نام و مقدار آن ذکر شده است. استفاده از مقدار بدون ذکر نام کافی نیست. زیرا صفات خاصه‌ی متفاوت می‌توانند دارای مقدار یکسانی باشند. برای مثال بدون ذکر نام صفات خاصه *Att1* و *Att2* در تراکنش شماره‌ی ۱ تشخیص مقدار *a* برای آنها مشکل است.

روش‌های بسیاری نیز وجود دارند که جهت بهبود عملکرد الگوریتم‌های کاوش الگوهای مکرر یا قوانین انجمنی، پایگاه داده‌ی تراکنشی را به شکل‌های دیگری تبدیل می‌کنند. یکی از روش‌های ساده، تبدیل پایگاه داده‌ی تراکنشی به رشته‌ای از صفر و یک است که در شکل ۴-۱۰ یک نمونه از آن نشان داده شده است.

TID	Items	TID	a	b	c	d	e
10	{ <i>a,b,c</i> }	10	1	1	1	0	0
20	{ <i>b,d</i> }	20	0	1	0	1	0
30	{ <i>a,c,d</i> }	30	1	0	1	1	0
40	{ <i>b,d,e</i> }	40	0	1	0	1	1
50	{ <i>a,c,d,e</i> }	50	1	0	1	1	1

شکل ۴-۱۰: نمایش متفاوتی از یک پایگاه داده‌ی تراکنشی

همانطور که مشاهده می‌کنید داده‌های سبد خرید می‌توانند به شکل دودویی نمایش داده شوند که هر سطر نشان‌دهنده‌ی یک تراکنش و هر ستون مخصوص یک قلم داده است. وجود و عدم وجود قلم داده به ترتیب با مقدار یک و صفر تنظیم می‌شوند. روش‌های متعددی برای نمایش پایگاه داده‌ی تراکنشی وجود دارند. انتخاب یک روش می‌تواند بر روی هزینه‌ی خواندن و نوشتمن داده‌ها بر روی حافظه‌ی اصلی و یا حافظه‌ی ثانویه تأثیر به سزایی داشته باشد. شکل ۱۱-۴ یک نمونه‌ی دیگری از تبدیل یک پایگاه داده‌ی تراکنشی را نشان می‌دهد.

TID	Items		Items	TID's
10	{a,b,c}		a	10 20 30
20	{b,d}		b	10 20 40
30	{a,c,d}		c	10 30 50
40	{b,d,e}		d	20 30 40 50
50	{a,c,d,e}		e	40 50

شکل ۱۱-۴: نمونه‌ای از نمایش افقی (سمت چپ) و عمودی داده‌ها (سمت راست)

نمایش اولیه‌ی پایگاهداده برای سبد خرید به عنوان شکل افقی^۱ داده‌ها شناخته می‌شود که بسیاری از الگوریتم‌ها از این شکل به عنوان ورودی خودشان استفاده می‌کنند. امکان دیگر ذخیره داده‌های تراکنشی که جدول سمت راست شکل ۱۱-۴ آن را نمایش می‌دهد، موسوم به شکل عمودی^۲ داده‌هاست. در این شکل جدید، شماره تراکنش‌ها اقلام داده‌ها را تشکیل می‌دهند. در این وضعیت جهت محاسبه‌ی مقدار پشتیبان برای هر مجموعه قلم کافی است تا اشتراکی از لیست تراکنش‌های اقلام تشکیل دهنده محاسبه شود. برای مثال مقدار پشتیبان مجموعه اقلام ۲تاًی {b,d} بصورت زیر محاسبه می‌شود:

$$\{10,20,40\} \cap \{20,30,40,50\} = \{20,40\}$$

¹ Horizontal

² Vertical

این بدین معنی است که ۲ تراکنش با مشخصه‌ی ۲۰ و ۴۰ حاوی مجموعه اقلام ۲تایی $\{b,d\}$ هستند. بنابراین مقدار پشتیبان آن برابر با ۲ یا ۴۰ درصد بیان می‌شود. بررسی‌ها نشان می‌دهند در شرایط مشخصی مانند وجود الگوهای مکرر با طول بزرگ، این تکنیک می‌تواند سرعت بهتری نسبت به هنگامی که از شکل استاندارد پایگاهداده‌ی تراکنشی استفاده می‌کنیم، داشته باشد.

۶-۴) مجموعه اقلام ماکسیمال و بسته

چالش بزرگی که در کاوش الگوهای مکرر وجود دارد این است که الگوریتم‌ها عموماً تعداد بسیار زیادی مجموعه اقلام تولید می‌کنند که مقدار پشتیبان اغلب آنها نیز از $minsup$ بیشتر است. به ویژه هنگامی که مقدار $minsup$ کوچک و یا الگوهای مکرر دارای طول بزرگی باشند. این بدین سبب است که هر یک از الگوهای مکرر دارای زیرمجموعه‌هایی است که آنها نیز مکررند. بدین ترتیب اگر در پایگاه داده یک الگوی مکرر با طول بالا (بطور مثال ۱۰۰) یافت شود، فقط برای نگهداری زیرمجموعه‌های این الگوی مکرر نیاز به فضای زیادی از حافظه داریم. جهت غلبه براین مشکل به معرفی دو مفهوم ماکسیمال^۱ و بسته^۲ برای الگوهای مکرر می‌پردازیم. در واقع این دو مفهوم راهی جهت نمایش فشرده‌تر الگوهای مکرر هستند. به عبارت دیگر با ذخیره‌ی اطلاعات کمتر می‌توانیم به همه‌ی الگوهای مکرر دسترسی داشته باشیم.

یک الگوی مکرر هنگامی ماکسیمال است که هیچیک از آبرمجموعه‌های آن مکرر نباشد. برای مثال چنانچه بخواهید بدانید الگوی مکرر ۲تایی ab در یک پایگاهداده‌هایی که از اقلام داده‌ی $\{a,b,c,d,e\}$ استفاده می‌کند، ماکسیمال هست یا خیر، باید کلیه‌ی آبرمجموعه‌های بلافصل ab یعنی (abc, abd, abe) را بررسی کنید. اگر هیچکدام از آنها مکرر نباشند، الگوی مکرر ab ماکسیمال است. همان طور که ملاحظه می‌کنید لازم نیست تا آبرمجموعه‌های ۴ یا ۵ عضوی مانند $abcd$ یا $abcde$ بررسی شوند. زیرا اگر abc مکرر نباشد هیچیک از آبرمجموعه‌های آن نیز مکرر نیستند. بنابراین بررسی

¹ Maximal

² Closed

آبرمجموعه‌های بلافصل یک الگوی مکرر برای تشخیص ماکسیمال بودن یا نبودن آن کافی هستند. با در دسترس بودن الگوهای مکرر ماکسیمال نیازی به نگهداری کلیه‌ی الگوهای مکرر نیست. چرا که می‌توان با کمک آنها همه‌ی الگوهای مکرر را بدست آورد. بنابراین برای پایگاهداده‌ای که شامل الگوهایی با طول بالا هستند، بهتر است الگوهای مکرر ماکسیمال نگهداری شوند. با این وجود این راهکار هنگامی عملی و کارا خواهد بود که بتوان الگوریتم مناسبی جهت یافتن الگوهای مکرر ماکسیمال طراحی نمود. علیرغم اشغال فضای بسیار کم، الگوهای مکرر ماکسیمال حاوی هیچ اطلاعاتی درباره‌ی مقدار پشتیبان زیرمجموعه‌های خود نیستند. آنها فقط مقدار پشتیبان را برای الگوهای مکرر ماکسیمال نگهداری می‌کنند. هرچند با یک پیمایش دیگر پایگاهداده‌ها می‌توان مقدار پشتیبان را برای الگوهای مکرری که ماکسیمال نیستند، بدست آورد.

مجموعه اقلام بسته بدون از دست دادن اطلاعاتی در مورد مقدار پشتیبان، شکل فشرده و کاهش یافته‌ای از مجموعه اقلام را تهییه می‌کند. یک مجموعه اقلام X هنگامی بسته نامیده می‌شود که هیچیک از آبرمجموعه‌های بلافصل آن مقدار پشتیبانی دقیقاً برابر با پشتیبان X نداشته باشد. به عبارت دیگر مجموعه اقلام X بسته نیست، اگر حداقل یک آبرمجموعه بلافصل او مقدار پشتیبانی برابر با پشتیبان X داشته باشد. حال اگر X مکرر نیز باشد آنگاه X به عنوان یک الگوی مکرر بسته شناخته می‌شود. برای مثال چنانچه در پایگاهداده‌ی تراکنشی مثال قبلی که از ۵ قلم $\{a,b,c,d,e\}$ استفاده می‌کرد، مقدار پشتیبان برای دو مجموعه اقلام ab و abc برابر باشد، بنابراین ab نمی‌تواند به عنوان یک الگوی بسته بیان شود. زیرا حداقل یک آبرمجموعه بلافصل ab یعنی ab مقدار پشتیبانی برابر با ab دارد. الگوریتم‌های متعددی وجود دارند که می‌توانند بطور مستقیم الگوهای مکرر بسته را از پایگاهداده‌ها استخراج کنند. با کمک الگوهای مکرر بسته قادر به مقدار پشتیبان الگوهای مکرری که بسته نیستند را نیز محاسبه کنیم. فراموش نکنید به منظور یافتن مقدار پشتیبان برای الگوهایی که بسته نیستند، مقدار پشتیبان کلیه‌ی آبرمجموعه‌های آن باید شناخته شده باشند.

فرض کنید یک پایگاه داده تراکنشی دارای ۲ تراکنش است که یکی دارای ۲۰ قلم و دیگری دارای ۱۰ قلم داده است (جدول ۶-۴).

جدول ۶-۴: پایگاه داده‌ی تراکنشی با ۲ تراکنش

TID	Items
10	{I1,I2,...,I19,I20}
20	{I1,I2,...,I9,I10}

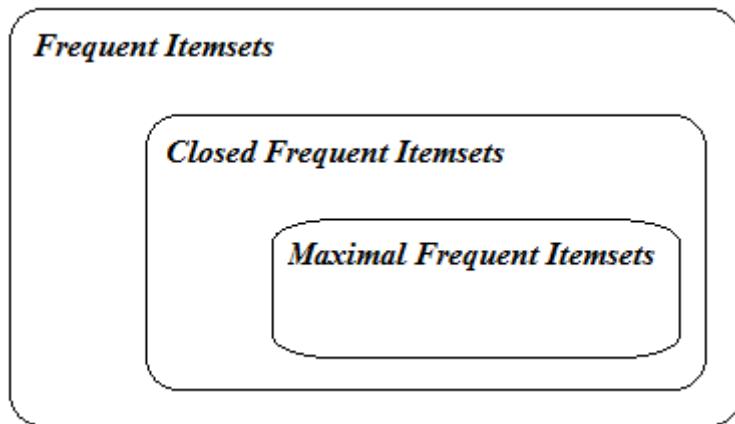
مقدار $minsup$ نیز برابر با ۱ (۵۰ درصد) تنظیم شده است. با این فرض تعداد الگوهای مکرر این پایگاه داده‌ی کوچک به عدد $10^{\approx 20}$ -۳ می‌رسد. در صورتی که الگوهای مکرر ماکسیمال و بسته برای این پایگاه داده به ترتیب عبارتند از:

$$Maximal = (\{I1,I2,...,I20\})$$

$$Closed = (\{I1,I2,...,I20\}, \{I1,I2,...,I10\})$$

این مثال مزیت استفاده از الگوهای مکرر ماکسیمال و بسته را نسبت به نگهداری همه‌ی الگوهای مکرر به خوبی نشان می‌دهد. به جای نگهداری مجموعه‌ای با $10^{\approx 20}$ عضو، کافی است فقط مجموعه‌ای با ۲ یا ۱ عضو نگهداری شوند. توجه کنید حتی با مقدار ۲ برای $minsup$ هنوز تعداد مجموعه اقلام مکرر برای این داده‌ها کم نیست ($10^{\approx 30}$ -۳). اغلب برای تحلیل داده‌ها کافی است الگوهای مکرر بسته نگهداری شوند.

شکل ۶-۱۲ رابطه‌ی میان الگوهای مکرر، الگوهای مکرر بسته و همچنین ماکسیمال را نشان می‌دهد. در این شکل نشان داده شده است که کلیه‌ی الگوهای مکرر ماکسیمال، بسته نیز هستند. زیرا هیچیک از آنها نمی‌توانند مقدار پشتیبان یکسانی با آبرمجموعه‌های بالاصل خود داشته باشند. از الگوریتم‌های رایج برای کشف الگوهای مکرر بسته می‌توان به *Charm*, *Charm+* و *Closet*, *Closet+* اشاره نمود و دو الگوریتم مرسوم جهت یافتن الگوهای مکرر ماکسیمال *GenMax* و *Mafia* هستند.



شکل ۱۲-۴: رابطه‌ی میان الگوهای مکرر، الگوهای مکرر بسته و الگوهای مکرر ماکسیمال

خلاصه فصل

استخراج الگوهای مکرر و همچنین روابط موجود میان داده‌ها ممکن است بتواند در تصمیم‌گیری‌های مدیریتی کلان بسیار مفید باشد. تحلیل سبد خرید نمونه‌ای بارز از اینگونه کاربردهاست و همانطور که از نام آن مشخص است، در آن به دنبال اقلامی هستیم که به صورت مکرر توسط مشتریان در سبد خریدشان قرار می‌گیرد.

فضای جستجوی بسیار بزرگ برای یافتن اقلام مکرر باعث شده است تا از الگوریتم‌هایی استفاده کنیم که این فضا را کاهش می‌دهند. اولین الگوریتم از خاصیتی موسوم به *Apriori* بهره می‌برد که بدین صورت بیان می‌شود: کلیه زیرمجموعه‌های هر مجموعه‌ی مکرر خود مکرر هستند. علیرغم کاهش فضای جستجو با کمک گرفتن از *Apriori* همچنان در شرائطی خاص با مشکل محاسباتی روبرو هستیم.

الگوریتم *FP-Growth* که از یک ساختار درختی استفاده می‌کند، در مرحله‌ی اول به ساخت این درخت می‌پردازد و پس از آن با کمک درخت تولیدشده به دنبال مجموعه اقلام مکرر می‌گردد. هر چند عوامل مختلفی می‌توانند بر روی اجرای این الگوریتم تاثیر بگذارند، اما در مجموع این الگوریتم کارایی بهتری نسبت به الگوریتم *Apriori* دارد. پس از یافتن الگوهای مکرر تولید قوانین انجمنی شروع می‌شود. قوانین انجمنی قوی و جالب آنهایی هستند که معیارهایی نظیر پشتیبان و اطمینان را ارضا می‌کنند. جهت یافتن

قوانين انجمانی جالب بهتر است از قوانین اکتشافی و هیوریستیک‌های مناسب استفاده شود تا فضای جستجو کاهش یابد.

الگوهای مکرر ماسیمال و بسته راهی جهت نمایش فشرده‌تر الگوهای مکرر هستند. یک الگوی مکرر هنگامی ماسیمال است که هیچیک از آبرمجموعه‌های آن مکرر نباشد و آبرمجموعه‌های یک الگوی مکرر بسته مقدار پشتیبانی برابر با خود آن ندارند. تکنیک و الگوریتم‌های متعددی جهت یافتن این الگوهای مکرر پیشنهاد شده‌اند.

مراجع و منابع جهت مطالعه‌ی بیشتر

کشف قوانین انجمانی برای اولین بار در [AIS93] ارائه شد و پس از آن الگوریتم Apriori در [AS94b] پیشنهاد و بررسی شد. نسخه‌ی متفاوتی از الگوریتم با هیوریستیکی مشابه در [MTV94] آمده است و پس از آن ترکیبی از دو کار انجام شده نوشته شد [AMS⁺96]. در [AS94a] روشی برای یافتن قوانین انجمانی از روی مجموعه اقلام مکرر نوشته شده است.

پیشنهادهای زیادی جهت بهبود الگوریتم Apriori ارائه شده است که در ادامه به برخی از منابع آنها اشاره می‌کنیم. استفاده از توابع درهم‌ساز در [PCY95a]، پیشنهاد برای تکنیک‌های افزایش داده‌ها در [SON95]، کمک گرفتن از راهکارهای نمونه‌گیری در [Toi96]، شمارش پویای اقلام داده‌ها در [BMUT97] و کاوش قوانین انجمانی به صورت موازی و توزیع شده در [ZPOL97] [CHN⁺96] [PCY95b] و [CHNW96] بررسی شده‌اند. در ضمن در [CHNW96] روشی جهت بهینگام‌سازی الگوهای استخراج شده به صورت افزایشی پیشنهاد شده است.

الگوریتم‌های مقیاس‌پذیر دیگری غیر از الگوریتم Apriori را نیز می‌توان یافت که دارای منابع خوبی هستند. روش FP-Growth که بدون تولید اقلام مکرر کاندید کار خود را انجام می‌دهد، در [HPY00] پیشنهاد شد. روشی در [LPWH02] وجود دارد که دو گونه پیمایش بالا به پایین و پایین به بالا را برای درختان FP-tree در آن دیده می‌شود. یک پیاده‌سازی مبتنی بر آرایه و کارا برای ساختار prefix-tree را در [GZ03] جستجو کنید. کاوش اقلام مکرر در شکل عمودی پایگاه داده‌ها در

[Zak00] پیشنهاد شد و کنار هم قرار دادن دو مفهوم کاوش قوانین انجمنی و همچنین سیستم‌های پایگاه داده‌ای رابطه‌ای را نیز در منبع [STA98] می‌توانید پیدا کنید. الگوریتم‌های متعددی برای کاوش اقلام مکرر بسته پیشنهاد شده‌اند که هر کدام از هیوریستیک‌های خاصی بهره می‌برند. شاید بتوان گفت اولین بار در [PBTL99] به این نوع اقلام اشاره شد. به نام و عملکرد بعضی از آنها در این فصل اشاره کردیم. در اینجا برخی از منابعی را که می‌توانید با رجوع به آنها اطلاعات بیشتری در زمینه‌ی کاوش اقلام مکرر بسته کسب کنید، آورده شده‌اند. این مراجع عبارتند از [ZH02] [PHM00] [GZ03] [PCT⁺03] [LLL03] [WHP03]. کاوش الگوهای ماکسیمال هم ابتدا در [Bay98] مطالعه شد و سپس روش و الگوریتم موثر MAFIA برای این نوع از الگوها در [BCG01] بررسی گردید.

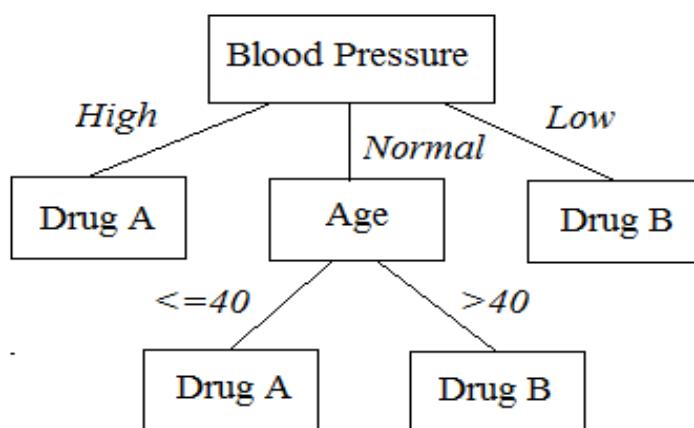
فصل هفتم

روش‌های طبقه‌بندی و تخمین

بدون شک اگر بخواهیم فقط شرح مختصری از تکنیک‌های طبقه‌بندی را ذکر کنیم، با حجم انبوهی از مطالب روبرو خواهیم بود. به همین علت فقط برخی از آنها انتخاب شده‌اند و بطور مختصر در مورد آنها در این فصل نکاتی بیان شده است. درخت‌های تصمیم از محبوب‌ترین روش‌های طبقه‌بندی محسوب می‌شوند که در این فصل چگونگی ساخت آنها را بررسی می‌کنیم. طبقه‌بندی با کمک قانون بیز، طبقه‌بندی مبتنی بر یافتن شروط، الگوریتم‌های *SVM*، طبقه‌بندی بر اساس تشابه، رگرسیون‌ها و همچنین مختصری در مورد شبکه‌های عصبی، الگوریتم ژنتیک و منطق فازی مطالبی هستند که در ادامه‌ی این فصل به شرح آنها می‌پردازیم.

۷-۱) درخت‌های تصمیم

درخت تصمیم^۱ در داده‌کاوی مدلی است که جهت نمایش طبقه‌کننده‌ها^۲ و رگرسیون‌ها استفاده می‌شود. همانطور که از نام آن مشخص است، این درخت از تعدادی گره و شاخه تشکیل شده است. 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

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

¹ Decision Tree

² 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 روپرتو سازد. به سراغ مثالی که در شکل ۷-۱ درخت تصمیم آن رسم شده است، برمی‌گردیم. 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 P.	Drug
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^۱ است. همچنین ابراز می‌شود که ساخت یک درخت دودوبی مینیمال با تعداد تست‌های قابل انتظار برای طبقه‌بندی یک نمونه‌ی جدید یک مسئله‌ی NP کامل^۲ 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^۳ هستند، درحالی که برخی دیگر فقط شامل مرحله‌ی رشد هستند. فرض کنید می‌خواهیم یک درخت تصمیم با کمک روش بالا به پایین ایجاد

¹ Hard

² Complete

³ 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 فشارخون 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 فشارخون 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 حداکثر عمق درخت رسیده باشیم. این حداکثر 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^۱ استفاده می‌کند.

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

این فرمول *InformationGain* را برای صفت خاصه 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 خاصه A و D_j قسمتی از داده‌های اولیه که مقدار صفت خاصه آنها v_j است را نشان می‌دهند. در ضمن $|D_j|/|D|$ دلالت بر اندازه 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.

^۱ 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 مزبور 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.

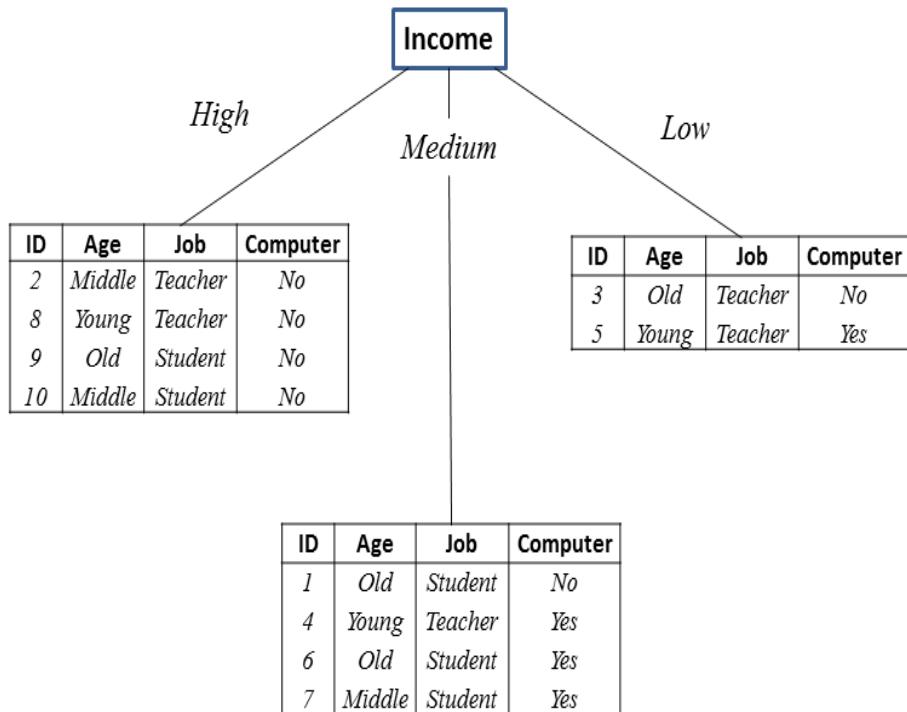
$$InformationGain(Age) = 0.970 - 0.875 = 0.095$$

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

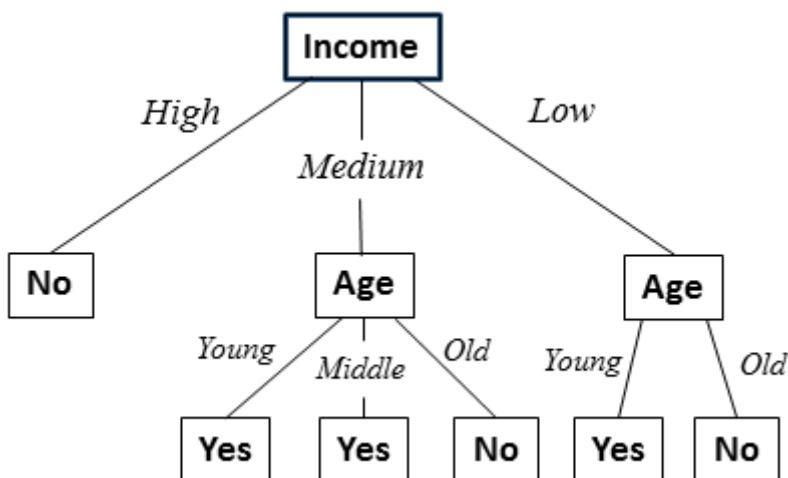
$$InformationGain(Job) = 0.970 - 0.970 = 0$$

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 دقیق این مدل برابر با $9/10 = 0.9$ است و نرخ خطای آن برابر با $1 - 0.9 = 0.1$ 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 انفال 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 انفال 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.

شکل ۷-۵ چگونگی یافتن نقطه‌ی انفال 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		

شکل ۷-۵: محاسبه‌ی آنتروپی برای نقاط انفال 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 $\frac{16}{16} = 1$

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.

معیار *Likelihood Ratio*

این معیار را می‌توان به صورت زیر معرفی کرد:

$$G_A^2(D) = 2 \times \ln(2) \times |D| \times InformationGain(A)$$

معیار مذبور برای اندازه‌گیری اهمیت آماری معیار *Information Gain* مناسب است. چنانچه فرض صفر استقلال شرطی صفات خاصه و برچسب کلاس تعیین شود، آزمون آماری بر اساس χ^2 با درجه‌ی آزادی زیر توزیع می‌شود:

$$(Domain(A)-1) \times (C-1)$$

که در آن ($Domain(A)$) به معنی تعداد اعضای دامنه‌ی صفت‌خاصه‌ی A و C نشان دهنده‌ی تعداد برچسب‌های کلاس هستند.

معیار DKM

این معیار برای مجموعه داده‌های آموزشی دو کلاسی طراحی شده است و به صورت زیر بیان می‌شود.

$$DKM(D) = 2 \times \sqrt{\frac{|D_1|}{|D|} \times \frac{|D_2|}{|D|}}$$

استفاده از این معیار در مقایسه با *Gini Index* و *Information Gain* با خطای معین، باعث ایجاد درخت تصمیم کوچکتری می‌شود.

۷-۱-۲) چند موضوع دیگر در مورد درختان تصمیم

در این بخش به بررسی چند موضوع در مورد درختان تصمیم می‌پردازیم. هرس کردن درخت، چگونگی مواجهه الگوریتم‌ها با داده‌های ناقص یا مفقود و همچنین داده‌های نویز از موضوعاتی هستند که روش‌های متعددی از ساخت درختان تصمیم با آن دست و پنجه نرم می‌کنند.

به طور معمول یک الگوریتم ساخت درخت تصمیم به صورت بازگشتی مجموعه داده‌های آموزشی را به بخش‌های کوچکتری تقسیم می‌کند و این کار را تا آخرین صفت‌خاصه و یا آخرین نمونه ادامه می‌دهد. نتیجه‌ی این فرایند درخت عمیقی است با تعداد زیادی برگ که هر یک از این برگ‌ها تعداد نمونه‌های کمی را پوشش می‌دهند. دقت تخمین برای برچسب کلاس یک نمونه از داده‌های آموزشی که با کمک این درخت تصمیم ساخته شده، بسیار عالی است. اما متأسفانه هنگامی که قصد طبقه‌بندی نمونه‌ای جدید از داده‌ها را داشته باشیم، با دقت پایینی روبرو خواهیم بود. این پدیده با نام *Overfitting* شناخته می‌شود و می‌تواند ناشی از داده‌های نویز باشد. برچسب نادرست کلاس و یا مقدار اشتباه یک صفت‌خاصه و یا حتی پیچیدگی و تنوع دامنه‌های کاربردی نیز می‌تواند از عوامل ایجاد این پدیده باشد. جهت کاهش این مشکل می‌توانیم درخت تصمیم خود را هرس

کنیم. درواقع با حذف تعدادی از شاخه‌ها و یا زیردرخت‌ها و جایگزینی آنها با برگ‌هایی که نظر اکثریت را نشان می‌دهند، درخت را هرس کرده‌ایم.

برای انجام این کار می‌توان از دو راهکار پیش هرس^۱ و پس هرس^۲ استفاده نمود. همانطور که از نام هر یک می‌توان عملکرد آنها را حدس زد، در روش‌های پیش هرس در حین ساخت درخت تصمیم عمل هرس کردن انجام می‌شود، در حالی که در الگوریتم‌های پس هرس این عمل بعد از ساخت کامل درخت تصمیم صورت می‌پذیرد. نشان داده شده است که الگوریتم‌های پس هرس کارتر هستند. زیرا پس از ساخت درخت، دید روشن‌تر و کامل‌تری از شاخه‌ها و یا زیردرخت‌های غیرمفید داریم. الگوریتم‌های متعددی هم برای پیش هرس و هم برای پس هرس وجود دارند.

کنترل کردن مقادیر ناقص یا مفقود نیز در درخت‌های تصمیم می‌تواند چالش بزرگی محسوب شود. همانطور که قبل از این نیز در فصل‌های قبل ذکر شد، داده‌های دنیای واقعی می‌توانند قادر باشند و برای تحلیل این نوع داده‌ها راهکارهای متفاوتی نیز بیان شده‌اند. در برخی از الگوریتم‌های ساخت درخت تصمیم نیز تمهیداتی جهت مقابله با این نوع داده وجود دارد.

شاید به جرأت می‌توان گفت که روش‌های موجود جهت ساخت یک درخت تصمیم برای داده‌های بسیار حجیم کارایی خود را از دست می‌دهند. اکثر این روش‌ها با این فرض اجرا می‌شوند که مجموعه داده‌های آموزشی در حافظه‌ی اصلی قرار می‌گیرند. در صورتی که می‌دانیم تعداد زیاد نمونه‌ها امری طبیعی در دنیای واقعی است و معمولاً تمام این داده‌ها نمی‌توانند در حافظه‌ی اصلی کامپیوتر قرار بگیرند. بنابراین راهکارهای مقیاس‌بذیرتری مورد نیاز است که از جایه‌جایی مکرر داده‌ها میان حافظه اصلی و محل ذخیره‌ی داده‌های آموزشی جلوگیری کند.

الگوریتم‌های مقیاس‌بذیر متعددی مثل *SPRINT* و *SLIQ* از جمله الگوریتم‌هایی هستند که درخت تصمیم را از داده‌های بسیار بزرگ ایجاد می‌کنند. هر دوی این الگوریتم‌ها داده‌های عددی و غیرعددی را پشتیبانی می‌کنند. ایده‌ی اصلی آنها استفاده از

¹ Preprunning

² Postprunning

یک ساختمان داده‌ی جدیدتر و موثرتر جهت ساخت ساده‌تر درخت تصمیم است. دو روش *BOAT* و *RainForest* نیز الگوریتم‌های ساخت درخت تصمیمی هستند که جهت مقیاس‌پذیری بهتر از راهکاری متفاوت از دو روش قبلی بهره می‌برند. منابع بیشتر برای مطالعه‌ی این روش‌ها در انتهای فصل ذکر شده‌اند.

۷-۱-۳) 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 *Information Gain* استفاده می‌کند. در اجرای این الگوریتم دو شرط توقف وجود دارد. یکی این که کلیه‌ی 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 پذیرد.

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.

الگوریتم^۱ *CART*

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.

الگوریتم^۲ *CHAID*

پس از سال‌های ۱۹۷۰ محققان آمار کاربردی الگوریتم‌هایی را جهت تولید و ساخت درخت تصمیم توسعه دادند. از میان این الگوریتم‌ها می‌توان به *MAID*^۳ و *AID* و *CHAID* اشاره کرد. الگوریتم *CHAID* در ابتدا برای متغیرهای اسمی طراحی شده بود. این الگوریتم با توجه به نوع برچسب کلاس از آزمون‌های مختلف آماری استفاده می‌کند. این الگوریتم هرگاه به حداقل عمق تعریف شده‌ای برسد و یا تعداد نمونه‌ها در گرهی جاری از مقدار تعریف شده‌ای کمتر باشد، متوقف می‌شود. الگوریتم *CHAID* هیچ گونه روش هرسی را اجرا نمی‌کند و می‌تواند مقادیر ناقص را نیز کنترل کند.

۷-۲) طبقه‌بندی با کمک قانون بیز

یکی از فرمول‌های مهم احتمال، فرمول احتمال بیز^۴ 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^۵ کلاس نامیده می‌شود. این فرض همانطور که در ادامه خواهیم دید باعث ساده‌تر شدن محاسبات می‌شود.

¹ Classification Regression Trees

² Chi-squared Automatic Interaction Detection

³ Bayesian Theorem

⁴ 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 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:

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

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

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

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

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

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

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

۷-۳) روش‌های طبقه‌بندی مبتنی بر یافتن شروط

روشی دیگری که برای طبقه‌بندی داده‌ها استفاده می‌شود، استخراج مجموعه‌ای از قوانین^۱ به صورت شرط است. قبل از این نیز نشان دادیم که هر درخت تصمیم می‌تواند توسط مجموعه‌ای از این شروط نمایش داده شود (در شکل ۷-۴). در این بخش برآئیم تا نشان دهیم که می‌توانیم این شروط را به طور مستقیم از مجموعه داده‌های آموزشی استخراج کنیم.

^۱ Rules

یک قانون به صورت کلی زیر نمایش داده می‌شود:

If ($A_1 \text{ op } v_1$) *and* ($A_2 \text{ op } v_2$) *and ... Then Class* = c_i

که در آن A_i نشانده‌نده‌ی نام یک صفت‌خاصه، v_i یک مقدار مشخص، متغیر $Class$ به صفت‌خاصه‌ی کلاس و c_i به یکی از مقادیر کلاس‌ها اشاره می‌کنند. متغیر op از میان عملگرهای مقایسه‌ای انتخاب می‌شود.

$$op = \{=, <>, <, >, \leq, \geq\}$$

در این بخش سمت چپ هر قانون را با واژه‌ی شرط یا شروط می‌شناسیم و هر شرط کامل (همراه با تخمین کلاس) را به عنوان یک قانون بیان می‌کنیم. همانطور که در شکل کلی هر قانون مشاهده می‌کنید، قسمت ابتدایی و سمت چپ هر قانون مجموعه‌ای از تست‌هایی است که بر روی صفات خاصه انجام شده است. میان این شروط از عملگر منطقی *and* استفاده می‌شود. قسمت نتیجه‌گیری قانون (سمت راست) همیشه تعیین یک کلاس از مجموعه برچسب‌های موجود در داده‌های اصلی است.

چنانچه مجموعه شروط سمت چپ یک قانون برای نمونه‌ای از داده‌ها صادق باشد، در واقع قانون مذبور، آن نمونه را پوشش می‌دهد. بنابراین می‌توان گفت یکی از اهداف ما یافتن قوانینی است که تعداد نمونه‌های بیشتری از مجموعه داده‌های آموزشی را پوشش دهد. بدین ترتیب امیدوار خواهیم بود که مدل با تعداد قوانین کمتری قابل توصیف است. علاوه بر این امکان دارد نمونه یا نمونه‌هایی موجود باشند که شروط سمت چپ قانون را ارضاء کنند، اما کلاسی متفاوت از کلاس تخمین زده شده توسط قانون داشته باشند. در این مورد دقت مدل نهایی کمی کاهش می‌یابد. در برخی از منابع می‌توان فرمول‌هایی را جهت ارزشیابی یک قانون پیدا کرد که دو فرمول زیر از رایج‌ترین آنها به شمار می‌رود.

$$\text{Coverage}(\text{Rule}) = \frac{N_{Cover}}{|D|}$$

$$\text{Accuracy}(\text{Rule}) = \frac{N_{Correct}}{N_{Cover}}$$

فرمول اول درصد پوشش قانون $Rule$ را نشان می‌دهد که در آن متغیر N_{Cover} تعداد نمونه‌هایی است که قانون مذبور پوشش داده است و $|D|$ دلالت بر تعداد کل نمونه‌ها در مجموعه داده‌های آموزشی دارد. فرمول دیگر دقت قانون $Rule$ را محاسبه می‌کند که در

آن $N_{Correct}$ تعداد نمونه‌های پوشش داده شده توسط قانون *Rule* است که برچسب کلاس آنها درست تخمین زده است.

در نهایت پس از یافتن مجموعه‌ای از قوانین، مدل حاصل باید توانایی تخمین داده‌های جدید را نیز داشته باشد. برای یافتن برچسب کلاس یک نمونه داده باید به دنبال قانونی باشیم که این نمونه را پوشش می‌دهد. به عبارتی دیگر مقادیر نمونه‌ی جدید، سمت چپ قانون را ارضا کند. پس از یافتن چنین قانونی کافی است کلاس تخمین زده شده توسط این قانون به عنوان کلاس نمونه‌ی آزمایشی در نظر گرفته شود. حال فرض کنید چنانچه از میان مجموعه قوانین، چنین قانونی یافت نشود و یا اینکه مقادیر نمونه جدید بیش از یک قانون را ارضا کند، روش چگونه باید عمل کند؟

برای پاسخ دادن به سوال اول یعنی حالتی که هیچ قانونی یافت نشود، بطور معمول الگوریتم‌ها قانون پیش فرضی را در انتهای قوانین اضافه می‌کنند. این قانون فقط برچسب کلاس را هنگامی تخمین می‌زند که نمونه‌ی آزمایشی توسط هیچ یک از قوانین پوشش داده نشود. اینکه چه برچسب کلاسی برای این قانون پیش فرض درنظر گرفته می‌شود، الگوریتم‌ها روش‌های متفاوتی را استفاده می‌کنند. در ضمن توجه کنید که هیچگونه شرطی در این قانون پیش فرض گنجانده نمی‌شود.

حال فرض کنید برای یک نمونه‌ی آزمایشی بیش از یک قانون وجود دارد که مقادیر صفات خاصه‌ی نمونه‌ی آزمایشی، شروط این قوانین را ارضا می‌کند. استراتژی‌های متفاوتی وجود دارند که می‌توان با کمک آنها این مشکل را حل نمود. ولی به طور کلی می‌توان گفت در همه‌ی این راه حل‌ها یک نوع اولویت‌گذاری میان قوانین مطرح است. بدین معنی که برای یک نمونه‌ی آزمایشی، بایستی قوانین به ترتیب خاصی تست شوند. بدین ترتیب اولین قانونی که نمونه‌ی آزمایشی را پوشش دهد، به عنوان قانون اصلی جهت تخمین برچسب کلاس نمونه انتخاب می‌شود. ترتیب قرارگیری این قوانین می‌تواند بر اساس اولویت کلاس‌ها تعیین شود. برای مثال ابتدا قوانین مربوط به کلاس‌هایی با اولویت بالاتر آزمایش شوند و به همین ترتیب تا کلاس‌هایی با اولویت پایین‌تر ادامه پیدا می‌کند. راه حل دیگر استفاده از قانونی با اندازه بزرگتر است. اندازه‌ی یک قانون بر اساس

تعداد شروطی مشخص خواهد شد که در سمت چپ آن قرار دارد. برای مثال دو قانون زیر را در نظر بگیرید.

R1: If A>5 and B='red' Then Class=c1

R2: If A<>12 and C>1 and D='yes' Then Class=c2

قانون R2 از نظر اندازه بزرگتر از R1 است. چرا که سمت چپ قانون R2 دارای سه شرط است، در حالیکه در R1 دو شرط وجود دارد. در این حالت الگوریتم انتخاب نهایی خود را به تخمین قانون بزرگ‌تر یعنی R2 می‌سپارد. روش رأی‌گیری راهکار دیگری است که ممکن است هنگامی که یک نمونه‌ی آزمایشی جدید بیش از یک شرط را ارضا می‌کند، استفاده شود. کلاس تخمینی نهایی کلاسی با رأی بالاتر خواهد بود. در این راهکار می‌توان برای هر رأی وزنی نیز متصور شد. به صورتی که برای مثال قانونی با دقت بالاتر رأی سنگین‌تری(با ارزش بالاتری) داشته باشد. این روش مزايا و معایب مربوط به خود را دارد اما به دلیل عدم ترتیب میان قوانین مدل سریع‌تر ساخته می‌شود و به نظر می‌رسد در مجموع دقت بهتری در تخمین زدن نسبت به حالتی که ترتیبی میان قوانین است، داشته باشد. اما فراموش نکنید که برای یک نمونه‌ی آزمایشی کلیه‌ی قوانین باید بررسی شوند.

۱-۳-۷) الگوریتم‌های یادگیری قوانین

یک دسته از الگوریتم‌هایی که قوانین را مستقیم از مجموعه داده‌های آموزشی استخراج می‌کند موسوم به الگوریتم‌های پوشش متوالی^۱ هستند. ایده‌ی اصلی این الگوریتم‌ها بدین صورت است که به دنبال قانونی با کیفیت می‌گردند و پس از یافتن آن و با حذف نمونه‌هایی که این قانون آنها را پوشش می‌دهد، این کار را تا تحقق یک شرط پایانی تکرار می‌کند. در واقع قوانین یکی یکی ایجاد می‌شوند.

الگوریتم‌های متعددی وجود دارند که به این شیوه عمل می‌کنند و از محبوب‌ترین آنها می‌توان به RIPPER، REP، I-REP، FOIL، AQ، CN2 اشاره نمود. برخی

¹ Sequential Covering

از این الگوریتم‌ها پس از یافتن کلیه‌ی قوانین مربوط به یک کلاس خاص به سراغ قوانین کلاس بعدی می‌روند. بدین ترتیب در لیست نهایی مجموعه قوانین، قوانین هر کلاس در کنار یکدیگر قرار می‌گیرند. ترتیب قوانین برای یک کلاس خاص مهم نیست، در حالیکه ترتیب میان قوانین کلاس‌های متفاوت باید حفظ شود. بطور معمول در ابتدا قوانین مربوط به کلاسی که کمترین نمونه را دارد، تولید می‌شوند. با این کار الگوریتم مطمئن خواهد بود که قوانینی را برای کلاسی با کمترین تکرار تولید کرده است. بعضی دیگر از الگوریتم‌ها بدون درنظر گرفتن کلاس خاصی، در هر مرحله قانونی از یک کلاس را تولید می‌کنند. بنابراین در لیست نهایی لزومی ندارد قوانین مربوط به یک کلاس خاص کنار یکدیگر قرار گیرند. رعایت ترتیب در اجرای قوانین نیز از اهمیت خاصی برخوردار است. اما چگونه یک قانون جدید تولید می‌شود؟ هرچند جزئیات الگوریتم‌های نام برده شده در بالا با یکدیگر متفاوت هستند، اما در ادامه با یک مثال ساده استراتژی کلی برای یافتن یک قانون را بیان می‌کنیم.

جدول ۷-۵ را که در آن ۳ صفت‌خاصه‌ی A و B و C همراه با برچسب کلاس Y نمایش داده شده است را در نظر بگیرید.

جدول ۷-۵: مجموعه داده‌ای با ۱۰ نمونه‌ی آموزشی

ID	A	B	C	Y
1	a_1	b_1	c_1	y_1
2	a_1	b_2	c_1	y_1
3	a_1	b_2	c_2	y_2
4	a_2	b_1	c_1	y_2
5	a_2	b_2	c_3	y_2
6	a_3	b_1	c_2	y_2
7	a_3	b_1	c_1	y_1
8	a_3	b_2	c_1	y_2
9	a_3	b_2	c_3	y_2
10	a_3	b_1	c_3	y_1

این مجموعه دارای ۱۰ نمونه‌ی آموزشی است که در دو کلاس با برچسب y_1 و y_2 طبقه‌بندی شده‌اند. الگوریتم در پی یافتن قوانینی جهت مدل‌سازی این داده‌ها است. برای شروع به سراغ کلاس y_1 و قانونی که هیچ گونه شرطی در آن نیست، می‌رویم.

If ? Then $Y=y_1$

توجه کنید که ما به دنبال شرط یا شروطی هستیم تا جایگزین ؟ شود. برای اولین شرط ۸ گزینه با توجه به مقادیر ۳ صفت‌خاصه وجود دارد که همراه با دقت حاصل از هر یک در شکل ۷-۶ نشان داده شده است.

$A=a_1$	$Accuracy(A=a_1, Y=y_1) = n_{correct}/n_{cover} = 2/3$
$A=a_2$	$Accuracy(A=a_2, Y=y_1) = n_{correct}/n_{cover} = 0/2$
$A=a_3$	$Accuracy(A=a_3, Y=y_1) = n_{correct}/n_{cover} = 2/5$
$B=b_1$	$Accuracy(B=b_1, Y=y_1) = n_{correct}/n_{cover} = 3/5$
$B=b_2$	$Accuracy(B=b_2, Y=y_1) = n_{correct}/n_{cover} = 1/5$
$C=c_1$	$Accuracy(C=c_1, Y=y_1) = n_{correct}/n_{cover} = 3/5$
$C=c_2$	$Accuracy(C=c_2, Y=y_1) = n_{correct}/n_{cover} = 0/2$
$C=c_3$	$Accuracy(C=c_3, Y=y_1) = n_{correct}/n_{cover} = 1/3$

شکل ۷-۶: دقت محاسبه شده برای حالت‌های متفاوتی از شروط

اعداد سمت راست نشان دهنده‌ی دقت شرط ساخته شده است. به طور مثال مقدار $2/3$ را برای $A=a_1$ در نظر بگیرید. این عدد به این معنی خواهد بود که در مجموعه داده‌ها شما می‌توانید ۳ نمونه را پیدا کنید که مقدار صفت‌خاصه A آنها برابر با a_1 است و از میان این ۳ نمونه، برچسب ۲ تای آنها برابر با y_1 درج شده است. از میان شروط فوق شرطی با بزرگترین مقدار دقت انتخاب می‌شود. بنابراین داریم:

If $A=a_1$ Then $Y=y_1$

همان طور که بیان شد شرط موجود در این قانون ۳ نمونه را پوشش می‌دهد، اما یکی از نمونه‌ها درست طبقه‌بندی نمی‌شود. لذا در ادامه برای کامل شدن می‌توان به دنبال عبارت شرطی دیگری باشیم تا در کنار $A=a_1$ قرار دهیم، به نحوی که این نمونه نیز پوشش داده شود.

با حذف صفت خاصه‌ی A تعداد ۵ گزینه را می‌توان انتخاب نمود که در شکل ۷-۷ نشان داده شده‌اند.

$B=b_1$	$Accuracy(A=a_1 \text{ and } B=b_1, Y=y_1) = n_{correct}/n_{cover} = 1/1$
$B=b_2$	$Accuracy(A=a_1 \text{ and } B=b_2, Y=y_1) = n_{correct}/n_{cover} = 1/2$
$C=c_1$	$Accuracy(A=a_1 \text{ and } C=c_1, Y=y_1) = n_{correct}/n_{cover} = 2/2$
$C=c_2$	$Accuracy(A=a_1 \text{ and } C=c_2, Y=y_1) = n_{correct}/n_{cover} = 0/1$
$C=c_3$	$Accuracy(A=a_1 \text{ and } C=c_3, Y=y_1) = n_{correct}/n_{cover} = 0/0$

شکل ۷-۷: دقت محاسبه شده برای مرحله‌ی دوم از الگوریتم

در این لیست دو مقدار $1/1$ و $2/2$ بزرگترین مقدار را دارند. اما از آنجا که به ازای $C=c_1$ تعداد نمونه‌های بیشتری (۲ نمونه به جای ۱ نمونه) پوشش داده می‌شود، این شرط را انتخاب می‌کنیم.

If ($A=a_1$ and $C=c_1$) Then $Y=y_1$

لازم نیست قانون بدست آمده با شرط دیگری توسعه داده شود، چرا که دقتی برابر با 100 درصد دارد. اما این قانون فقط دو نمونه از چهار نمونه‌ای که دارای برچسب کلاس y_1 هستند را پوشش می‌دهد. لذا پس از حذف نمونه‌های ۱ و ۲ که توسط این قانون پوشش داده می‌شوند، به سراغ ساخت قانون بعدی می‌رویم. در واقع مراحل قبلی تکرار می‌شوند ولی این بار با ۸ نمونه‌ی باقیمانده‌ای که قانون (قوانين) قبلی آنها را پوشش نداده است. این الگوریتم از تکنیک حریصانه پیروی می‌کند و در این تکنیک‌ها هیچگونه عقبگردی در الگوریتم مشاهده نمی‌شود. در هر مرحله با کمک یک هیوریستیک (دراینجا دقت قانون) بهترین انتخاب صورت می‌گیرد. اما بدلیل اینکه در حین اجرای الگوریتم با نتایج ضعیفی روبرو نشویم، پیشنهاد می‌شود به جای انتخاب یک شرط بهینه، چندین (برای مثال k) شرط بهینه انتخاب شود.

۷-۳-۲) معیارهای سنجش قوانین

بدون شک هر الگوریتم جهت سنجش قوانینی که تولید می‌کند، به معیارهایی نیاز دارد. به عبارت دیگر افزودن یک شرط به قانون در حال ساخت، با توجه به همین معیار امکان‌پذیر است.

دقت و پوشش در نظر اول معیارهای خوبی به نظر می‌رسند، اما در بسیاری از موقعیت‌ها دو معیار محک مناسبی نیستند. برای مثال دقت قانونی که ۴۸ نمونه از ۵۰ مورد را درست طبقه‌بندی می‌کند برابر با ۹۶ درصد و دقت قانونی که همه‌ی ۲ نمونه‌ی خود را بطور صحیح طبقه‌بندی کرده است برابر با ۱۰۰ درصد است. آیا در این مثال دقت بالاتر نشان از قانون قوی‌تری است. همینطور در نظر بگیرید قانونی را که تعداد نمونه‌های بیشتری نسبت به قانون دیگر پوشش می‌دهد، اما اکثر نمونه‌ها مربوط به برچسب کلاس تخمینی نیستند. بنابراین این دو معیار حداقل به تنها یکی قابل اعتماد نیستند و ما نیاز به معیارهای دیگری جهت سنجش قوانین داریم.

یکی از معیارها آنتروپی است که پیش‌تر توضیحات آن در بخش مربوط به ساخت درخت تصمیم ارائه شد. همانطور که می‌دانید این معیار در شرایطی یک صفت خاصه را انتخاب می‌کرد که با آن انتخاب، تعداد زیادی از نمونه‌های یک کلاس و همچنین تعداد کمی از کلاس‌های دیگر در انشعاب درخت گنجانده می‌شوند. همین عمل می‌تواند برای افزودن شرط جدید به یک قانون نیز بررسی شود.

معیار دیگری که اولین بار در الگوریتم *FOIL* پیشنهاد شد و در *RIPPER* از آن استفاده می‌شود و بر اساس پارامتر *Information Gain* بدست می‌آید، برای دو قانون R و R' به صورت زیر بیان می‌شود:

$$Gain(R, R') = P' \times \left(\log_2 \frac{P'}{P' + N'} - \log_2 \frac{P}{P + N} \right)$$

که در آن P (تعداد نمونه‌های مثبتی (منفی)) است که توسط قانون R پوشش داده می‌شوند و همینطور P' (تعداد نمونه‌های مثبتی (منفی)) است که توسط قانون R' پوشش داده می‌شوند. این معیار باعث می‌شود تا قوانینی با دقت بالا و همچنین درصد پوشش زیاد نمونه‌های مثبت انتخاب شوند.

معیار سوم یک آزمون آماری است که به مقایسه‌ی توزیع مشاهده شده و مورد انتظار کلاس در میان مجموعه نمونه‌های تحت پوشش قانون می‌پردازد. می‌توانیم از فرمول زیر برای محاسبه‌ی آن استفاده کنیم.

$$\text{Likelihood - Ratio} = 2 \sum_{i=1}^c O_i \times \text{Log}\left(\frac{O_i}{E_i}\right)$$

که در آن c تعداد کلاس‌ها را نشان می‌دهد و در میان مجموعه داده‌هایی که شروط قانون را ارضا می‌کنند، O_i تعداد تکرارهای کلاس i ام و E_i تعداد تکرارهای مورد انتظار کلاس i ام است برای وقتی که قانون بصورت اتفاقی کلاسی را تخمین می‌زند. این آزمون توزیع χ^2 را با درجه‌ی آزادی ($c-1$) دنبال می‌کند. الگوریتم CN2 از این آزمون همراه با آنتروپی استفاده می‌کند.

۳-۳-۷) بهینه نمودن قوانین

از آنجا که قوانین نهایی بر اساس مجموعه داده‌های آموزشی تولید می‌شوند و هیچگونه داده‌ی آزمایشی جهت ارزیابی قوانین تولید شده وجود ندارد، دقت بالای آنها نمی‌تواند کیفیت بالای این قوانین را تضمین کند. عواملی نظیر وجود داده‌های نویز می‌تواند باعث این پدیده باشد که اجرای قوانین بر روی داده‌های آموزشی خوب اما برای تخمین از دقت مناسبی برخوردار نباشند.

همانند الگوریتم‌های ساخت درخت تصمیم می‌توانیم برای بهتر شدن قوانین آنها را هرس کنیم. استراتژی‌های متفاوتی جهت هرس کردن قوانین وجود دارند که یکی از ساده‌ترین و مؤثرترین آنها را می‌توان در الگوریتم FOIL یافت که از فرمول زیر استفاده می‌کند:

$$\text{Prune}(Rule) = \frac{P - N}{P + N}$$

در آن P و N به ترتیب تعداد نمونه‌های مثبت و منفی پوشش داده شده توسط قانون $Rule$ هستند. مقدار بزرگتر برای جواب نهایی نشان‌دهنده‌ی بهتر بودن قانون است. به عبارت دیگر پس از تغییرات إعمال شده برای قانون خاص، فرمول فوق محاسبه می‌شود و

در صورتی که با افزایش مقدار آن روبرو شویم، قانون جدید به عنوان قانون هرس شده پذیرفته خواهد شد.

۴-۷) الگوریتم‌های *SVM*

به جرأت می‌توان گفت الگوریتم‌های *SVM*^۱ 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^۲ 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^۳ و حاشیه‌ها^۴ 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

¹ Support Vector Machine

² Hyperplane

³ Support Vectors

⁴ 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 تخمين نیز استفاده نمود. منظور از تخمين استفاده‌ی آنها در موقعی است که برچسب کلاس گستته نیست. از کاربردهای عملی این الگوریتم‌ها می‌توان به تشخیص الگو، پردازش تصویر، متن کاوی و کاربردهای پزشکی اشاره نمود.

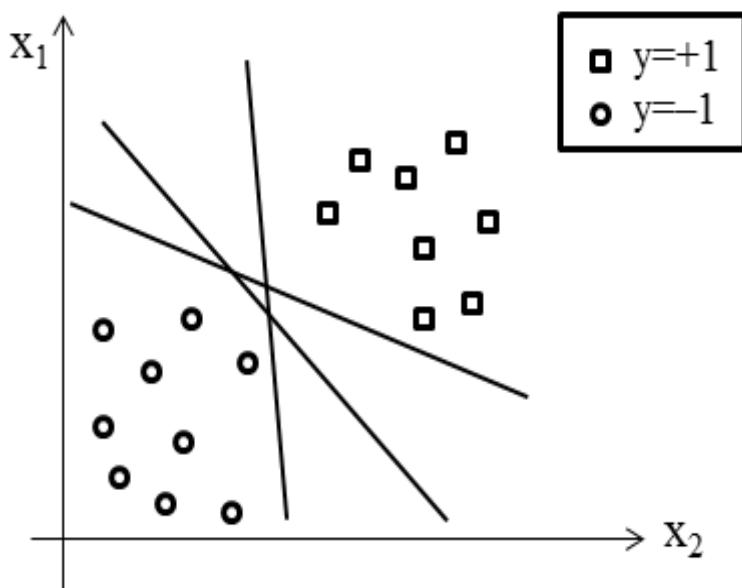
در دو بخش بعدی بصورت خلاصه دو حالت استفاده از *SVM* ها بررسی می‌شوند. حالت اول هنگامی است که می‌توانیم داده‌ها را به صورت خطی و با توجه به کلاس آنها جدا کنیم و حالت دیگر مربوط به موقعی است که نمی‌توان داده‌ها را به صورت خطی تفکیک نمود.

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

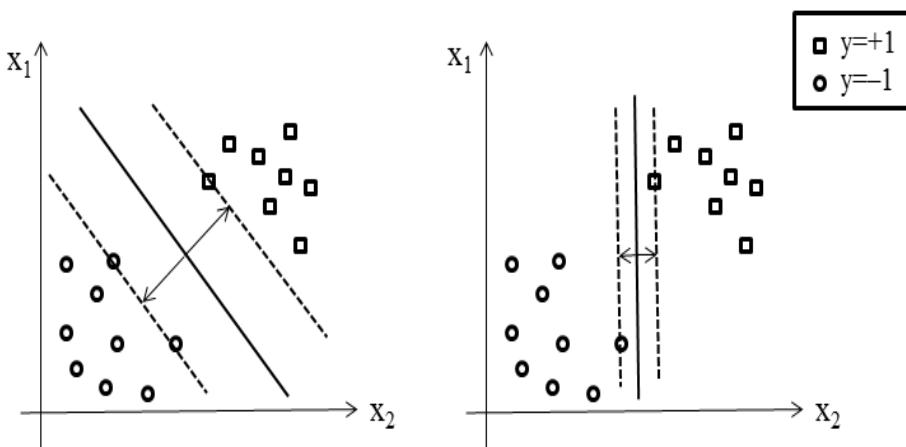
۷-۸ فقط ۳ نمونه از این خطوط رسم شده است. اما کدامیک از آنها بهترین انتخاب برای جداسازی نمونه‌های متفاوت است؟



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 +۱ برابر با کوتاهترین فاصله از آن آبرصفحه تا نمونه‌ای (یا نمونه‌هایی) با برچسب -۱ است. در واقع حاشیه از دو برابر این کوتاهترین فاصله بدست می‌آید.



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

یک آبرصفحه‌ی جداکننده را می‌توان به صورت زیر معرفی کرد:

$$W \cdot X + b = 0$$

که در آن $\{w_1, w_2, \dots, w_n\}$ برداری است که تعداد عضوهای موجود در آن برابر با صفات خاصه است و b مقدار ثابتی فرض می‌شود. در یک فضای دو بعدی که مجموعه داده‌ها با ۲ صفت خاصه و یک برچسب کلاس توصیف می‌شوند و با فرض $b=w_0$ معادله‌ی فوق بصورت زیر بازنویسی می‌شود:

$$w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 = 0$$

بدین ترتیب نمونه‌هایی (نقاطی) که در فضای بالایی این آبرصفحه (خط) قرار می‌گیرند نامعادله‌ی

$$w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 > 0$$

و نمونه‌هایی (نقاطی) که در فضای پایینی این آبرصفحه (خط) قرار می‌گیرند نامعادله‌ی

$$w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 < 0$$

را ارضاء می‌کنند.

یک نمونه (نقطه) از کلاس +1 و یک نمونه (نقطه) از کلاس -1 را که به آبرصفحه‌ی (خط) ما نزدیکترین هستند در نظر بگیرید. می‌توان دو آبرصفحه (خط) H_1 و H_2 را که مماس بر نمونه‌های است، رسم کرد. این آبرصفحه‌ها (خطوط) موازی آبرصفحه‌ی (خط) ما خواهند بود.

با تنظیم بردار W و مقدار b خواهیم داشت:

$$H_1: w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 \geq 1 \quad \text{If } y_i = +1$$

$$H_2: w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 \leq -1 \quad \text{If } y_i = -1$$

این بدین معنی است که هر نمونه‌ای بالا یا روی H_1 قرار گیرد، متعلق به کلاس +1 و هر نمونه‌ای که پایین و روی H_2 قرار گیرد متعلق به کلاس -1 - خواهد بود. هر یک از نمونه‌هایی که دقیقاً بر روی آبرصفحه‌ی H_1 و یا H_2 قرار دارند را بردار پشتیبان می‌نامیم. حداقل دو نمونه (از هر کلاس یک نمونه) وجود خواهند داشت که بردار پشتیبان را تشکیل می‌دهند. در واقع بردارهای پشتیبان مشکل‌ترین نمونه‌ها جهت طبقه‌بندی داده‌ها به حساب می‌آیند و اطلاعات زیادی را برای طبقه‌بندی در اختیار ما قرار می‌دهند. اکنون می‌توانیم اندازه‌ی حاشیه را محاسبه کنیم. فاصله‌ی موجود بین آبرصفحه‌ی

جداکننده و هر نمونه‌ی موجود بر روی H_1 برابر با $\frac{1}{\|w\|}$ است، که مخرج کسر نرم

اقلیدسی^۱ را نشان می‌دهد و برابر است با:

$$\sqrt{W \cdot W} = \sqrt{w_1^2 + w_2^2 + \dots + w_n^2}$$

چون این فاصله برابر با فاصله‌ی هر نمونه‌ی موجود بر روی H_2 تا آبرصفحه نیز هست، اندازه‌ی حاشیه برابر با $\frac{2}{\|w\|}$ خواهد بود. حال الگوریتم به دنبال آبرصفحه‌ای است که

اندازه‌ی این حاشیه را به حداقل برساند. در واقع ما با یک مسئله‌ی بهینه‌سازی^۲ درجه‌ی دوم روبرو هستیم که توضیح آن از حدود این کتاب خارج است و فقط به ذکر برخی از نتایج بسنده می‌کنیم. از آنجا کهتابع از نوع درجه‌دو و خطی و محدب^۳ است، می‌توان برای حل آن از ضرایب لاغرانژ^۴ و همچنین شروط *Kuhn-Tucker* استفاده نمود. محاسبات از پیچیدگی بالایی برخوردار است و برای مجموعه داده‌هایی با حجم بالا بهتر است از الگوریتم‌های کارایی استفاده شود. پس از یافتن بردارهای پشتیبان و در نتیجه

¹ Euclidean Norm

² Optimization

³ Convex

⁴ Lagrange Multipliers

آبرصفحه‌ای با حداکثر حاشیه، ما دارای یک SVM آموزش دیده هستیم. با کمک این آبرصفحه می‌توان داده‌هایی را که بصورت خطی تفکیک‌پذیرند، طبقه‌بندی نمود. به همین دلیل می‌توان آنرا خطی نامید. پس از ساخت SVM ، با کمک فرمول زیر می‌توانیم جهت تخمین برچسب کلاس یک نمونه‌ی آزمایشی مانند X^T استفاده کرد.

$$\sum_{i=1}^s y_i \cdot \alpha_i \cdot X_i \cdot X^T + b$$

که در آن i ‌ها بردارهای پشتیبان ما هستند همراه با برچسب کلاس i ، تعداد این بردارها با s نشان داده شده است و α_i و b نیز مقادیری هستند که توسط الگوریتم‌های بهینه‌سازی یا SVM محاسبه می‌شوند. پس از قرار دادن مقادیر نمونه‌ی آزمایشی X^T در عبارت فوق، علامت حاصل از نتیجه‌ی محاسبات بررسی خواهد شد. چنانچه علامت مثبت باشد بدین معنی است که نمونه‌ی X^T بر روی یا بالای آبرصفحه‌ی ما قرار دارد، بنابراین کلاس آن $+1$ است و بطور مشابه اگر علامت منفی باشد این نمونه پایین یا روی آبرصفحه واقع شده است که معنی آن این است که کلاس آن -1 خواهد بود.

همانطور که مشاهده می‌کنید پیچیدگی محاسبات مبتنی بر تعداد بردارهای پشتیبان است و بعد داده‌ها در آن تأثیر ندارد. از آنجا که بردارهای پشتیبان اساسی‌ترین نمونه‌ها برای یافتن آبرصفحه‌ی مورد نظر هستند، با حذف بقیه‌ی نمونه‌ها و تکرار الگوریتم می‌توان انتظار داشت نتیجه‌ی مشابهی حاصل می‌شود. به علاوه تعداد بردارهای پشتیبان می‌تواند در محاسبه‌ی یک محدوده‌ی نرخ خطای مورد انتظار به ما کمک کند.

۷-۴-۲) تفکیک‌پذیر غیرخطی

الگوریتم‌های SVM خطی که در بخش قبلی کلیات آن توضیح داده شد، قادر نیستند جهت داده‌های غیرخطی استفاده شوند. اما نکته‌ی مهم اینکه می‌توانیم آنها را به گونه‌ای تعمیم دهیم تا برای طبقه‌بندی داده‌هایی که نمی‌توان کلاس آنها را بصورت خطی جدا کرد، از آنها استفاده کنیم. بدین منظور آنها را الگوریتم‌های SVM غیرخطی می‌نامیم. در واقع این الگوریتم‌ها قادرند که در فضای ورودی به دنبال آبرسطح‌های^۱ غیرخطی باشند.

^۱ Hypersurface

الگوریتم‌های SVM غیرخطی دارای دو مرحله‌ی اصلی هستند. در مرحله‌ی اول با کمک یک نگاشت غیرخطی مجموعه داده‌های اولیه به فضایی با ابعاد بالاتر تبدیل می‌شود. چندین نگاشت غیرخطی وجود دارند که می‌توان از آنها استفاده نمود. پس از تبدیل داده‌ها به یک فضای جدید با ابعاد بالاتر، الگوریتم در مرحله‌ی دوم به دنبال یک آبرصفحه‌ی جداکننده‌ی خطی در فضای جدید می‌گردد. در این مرحله همانند قبل با یک مسئله‌ی بهینه‌سازی درجه دوم روبرو خواهیم بود. آبرصفحه‌ای با حداقل حاشیه که در فضای جدید داده‌ای پیدا شده است، نظیر آبرسطح جداکننده‌ی غیرخطی در داده‌های اصلی است. با نگاه مجدد به فرمول‌های بخش قبلی درمی‌یابیم که محاسبات زمانبند و سنگین است. برای تخمین کلاس یک نمونه‌ی آزمایشی باید ضرب برداری میان هر بردار پشتیبان و نمونه‌ی مزبور محاسبه شود. حتی در آموزش (اجرای الگوریتم) این عمل چندین بار جهت یافتن آبرصفحه‌ای با حداقل حاشیه تکرار می‌شود. خوشبختانه در عمل برای حل یک مسئله‌ی بهینه‌سازی درجه دوم مربوط به SVM خطی بسیار اتفاق می‌افتد که نمونه‌های داده‌های آموزشی به شکل ضرب برداری $\Phi(X_i) \cdot \Phi(X_j)$ بیان می‌شوند. که در آن $\Phi(X)$ یکتابع نگاشت غیرخطی است که برای تبدیل داده‌ها انتخاب شده است.

به جای محاسبه‌ی ضرب برداری بر روی داده‌های تبدیل شده‌ی جدید می‌توان از یک تابعی موسوم به تابع کرnel¹ برای داده‌های اولیه استفاده کرد. بدین ترتیب هرگاه حین اجرای الگوریتم یادگیری به $\Phi(X_i) \cdot \Phi(X_j)$ برخورد کردیم، می‌توان آنرا با تابع کرnel $K(X_i, X_j)$ جایگزین نمود. از آنجا که تابع کرnel بر روی داده‌های اولیه اجرا می‌شود و ابعاد این داده‌ها کمتر از داده‌های تبدیل شده‌ی جدید است، محاسبات سریع‌تر انجام خواهد شد. از یک نقطه نظر انگار نگاشتی صورت نگرفته است. پس از آن مشابه روشی که در بخش قبل توضیح داده شد به دنبال آبرصفحه‌ای با حداقل حاشیه می‌گردیم.

نکته‌ی قابل توجه اینکه توابع کرnel زیادی وجود دارند که هر یک دارای خصوصیات مربوط به خود هستند و شما می‌توانید با رجوع به منابع خاص به بررسی عملکرد هر یک از آنها بپردازید. هیچگونه شرط پیش فرضی برای انتخاب تابع کرnel بهتر وجود ندارد و به

¹ Kernel Function

دلیل مجزا بودن الگوریتم یادگیری و توابع کرنل، می‌توانیم مستقل از الگوریتم یادگیری به مطالعه‌ی این توابع پردازیم.

الگوریتم‌های *SVM* علیرغم مزایای فراوانی که دارند دارای محدودیت‌هایی نیز هستند. این الگوریتم‌ها فقط بر روی داده‌هایی با مقدار واقعی کار می‌کنند و انواع دیگر داده‌ها باید به داده‌های عددی تبدیل شوند. این الگوریتم‌ها اجازه‌ی فقط دو کلاس را می‌دهند. برای مسائلی که چندین برچسب کلاس در آن وجود دارد، چندین استراتژی هست که می‌توان مسئله را حل نمود. آبرصفحه‌ی تولید شده توسط الگوریتم به سختی برای کاربر قابل فهم است. مسئله در مورد توابع کرنل بدتر است. بنابراین پیشنهاد می‌شود هنگامی از این روش استفاده شود که لازم نیست خروجی‌ها به صورت مستقیم تحت اختیار کاربران قرار گیرد.

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^۱ 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^۲ نام دارد که در ادامه توضیحاتی پیرامون این روش بیان شده است.

¹ Lazy Learner

² 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 خوشه‌بندی 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 كمترین و در محاسبه‌ی فاصله بيشترین فاصله برای اين صفت خاصه در نظر گرفته می‌شود. فرض كنيد 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.

۷-۶) رگرسیون

در روش‌های طبقه‌بندی که قبل از این در مورد آنها بحث شد، اغلب برچسب کلاس از نوع داده‌ی گسسته (غیرعددی) بود. اگر چه در برخی از آنها با کمی تغییر می‌توان روش را برای تخمین کلاس‌های پیوسته (عددی) توسعه داد، اما روش‌های رگرسیون یکی از معروف‌ترین تکنیک‌های آماری به حساب می‌آیند که برای این کار بسیار مناسب هستند. تا جایی که در متون داده‌کاوی دو کلمه‌ی رگرسیون و تخمین کلماتی متراffد یکدیگر در نظر گرفته و استفاده می‌شوند. هدف تحلیل رگرسیون تعیین بهترین مدلی است که چگونگی ارتباط یک متغیر را با یک یا چند متغیر دیگر تعیین می‌کند. در بسیاری از

کاربردهای عملی نیاز به پیش‌بینی مقدار یک متغیر (وابسته) از روی مقادیر چند متغیر (مستقل) بسیار رایج است. برای مثال پیش‌بینی معدل ترم جاری یک دانشجو با توجه به نمرات چندین درس در ترم‌های قبل و یا سود حاصل از فروش محصولات یک شرکت با توجه به فروش و سود سال‌های گذشته از این نوع مسائل می‌باشند. همانطور که می‌توانید حدس بزنید در کاربردهای داده‌کاوی متغیر وابسته منظور برچسب کلاسی است که برای نمونه‌ها تعیین شده است. بدین ترتیب رابطه‌ی منتج از تحلیل رگرسیون رابطه‌ی میان صفات خاصه و کلاس را مشخص خواهد کرد. چندین بسته‌ی نرم‌افزاری وجود دارند که می‌توان به کمک آنها تحلیل رگرسیون را انجام داد و از میان آنها می‌توان به SAS، SPSS و S-Plus اشاره نمود. به دلیل گسترده‌ی بحث رگرسیون‌ها نمی‌توان در این بخش کوچک جزئیات آنها را بررسی کرد و به این دلیل فقط توضیح مختصری در بخش بعدی بیان می‌شود.

۱-۶-۷) رگرسیون خطی

معمولًاً بیشترین تکنیک‌های آماری مورد استفاده مدل‌های خطی هستند. چنانچه تعداد صفات خاصه در مجموعه داده‌ها n باشد، معادله‌ی زیر که به معادله‌ی رگرسیون موسوم است، می‌تواند رابطه‌ای که بر این مجموعه داده‌ها حاکم است را نشان دهد.

$$y = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n$$

که در آن x_i ها مقادیر صفات خاصه در مجموعه داده‌ها هستند، y برچسب کلاس و w_i ها که به ضرایب رگرسیون شناخته می‌شوند، پارامترهای نامعلومی هستند که بایستی برآورده شوند. منظور از رگرسیون خطی این است که میانگین y به طور خطی با x در ارتباط است.

ساده‌ترین شکل رگرسیون برای معادله‌ی رگرسیون هنگامی است که مجموعه داده‌های آموزشی دارای یک صفت خاصه و یک برچسب کلاس باشند. این رگرسیون که به نام رگرسیون ساده‌ی خطی شناخته می‌شود، y را همانند یکتابع خطی از x مدل می‌کند.

$$y = w_0 + w_1x$$

چنانچه واریانس σ^2 ثابت فرض شود، می‌توانیم ضرایب خط رگرسیون فوق را (w_0, w_1) با روش حداقل مربعات بدست آوریم، به طوری که خطای میان مجموعه داده‌ها و خط رگرسیون تخمین زده شده به حداقل خود برسد. در روش حداقل مربعات، مجموع مربعات باقیمانده را معمولاً مجموع مربعات خطاهای حول خط رگرسیون گویند و با SSE نمایش می‌دهند.

$$SSE = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \sum_{i=1}^n (y_i - w_0 + w_1 x_i)^2$$

که در آن e_i مقدار خطای مشاهده شده و \hat{y}_i و y_i به ترتیب جواب حاصل از مدل و مقدار واقعی خروجی هستند. به دنبال مقادیری از ضرایب رگرسیون (w_0, w_1) هستیم که SSE را به حداقل برساند. بنابراین با مشتق‌گیری از SSE نسبت به w_0 و w_1 و معادل صفر قرار دادن این مشتقات جزیی، معادلاتی به دست می‌آید که پس از حل آنها ضرایب به ترتیب زیر محاسبه می‌شوند:

$$w_1 = \frac{\sum_{i=1}^n (x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$w_0 = \bar{y} - w_1 \cdot \bar{x}$$

که در آن \bar{x} و \bar{y} به ترتیب میانگین مقادیر x و y هستند. با روشی مشابه می‌توان برای رگرسیون خطی چندگانه نیز ضرایب را تعیین نمود. رگرسیون خطی چندگانه مواقعي است که به جای یک متغیر مستقل چندین متغیر مستقل (چندین صفت‌خاصه) داشته باشیم. جدول ۷-۷ سود حاصل از سرمایه‌گذاری ۱۰ سال یک شرکت را نشان می‌دهد. پس از مشخص نمودن معادله رگرسیون، مایلیم سود حاصل از مقدار سرمایه‌گذاری ۱۰۰۰۰ را بدست آوریم. از محتویات جدول ۷-۷ مقادیر زیر حاصل می‌شوند که در آن x مقادیر سرمایه‌گذاری و y مقادیر سود را نشان می‌دهند.

$$\sum_{i=1}^{10} x_i = 91 \quad \sum_{i=1}^{10} y_i = 554 \quad \bar{x} = 9.1 \quad \bar{y} = 55.4$$

$$\sum_{i=1}^{10} x_i^2 = 1187 \quad \sum_{i=1}^{10} y_i^2 = 35444 \quad \sum_{i=1}^{10} x_i \cdot y_i = 6311$$

$$S_{xy} = \sum x_i \cdot y_i - [(\sum x_i \times \sum y_i) / n] = 6311 - [(91) \times (554) / 10] = 1269.6$$

جدول ۷-۷: سودهای حاصل از سرمایه‌گذاری یک شرکت در ۱۰ سال

ID	Cost (×1000)	Benefit (×10)
1	3	30
2	8	57
3	9	64
4	13	72
5	3	36
6	6	43
7	11	59
8	21	90
9	1	20
10	16	83

$$S_{xx} = \sum x_i^2 - [(\sum x_i)^2 / n] = 1187 - [(91)^2 / 10] = 358.9$$

$$w_1 = \frac{S_{xy}}{S_{xx}} = 1269.6 / 358.9 = 3.54$$

$$w_0 = \bar{y} - w_1 \cdot \bar{x} = 55.4 - (3.54) \cdot (9.1) = 23.18$$

در نتیجه داریم:

$$y = 23.18 + 3.54x$$

حال با جایگذاری مقدار ۱۰ به جای x در این معادله، سود حاصل از مقدار سرمایه‌گذاری ۱۰۰۰۰ برابر با ۵۸۵ بدست می‌آید:

$$y = 23.18 + (3.54) \cdot (10) = 58.58$$

روش‌های رگرسیون موقعی مناسب هستند که مقادیر صفات خاصه در مجموعه داده‌ها به برچسب کلاس بستگی داشته باشند. برای سنجش میزان وابستگی دو صفت خاصه (متغیر) از معیاری به نام ضریب همبستگی خطی استفاده می‌شود که می‌توان به صورت زیر تعریف کرد:

$$\rho(x, y) = \frac{Cov(x, y)}{\sqrt{Var(x) \times Var(y)}} = \frac{\sigma_{xy}}{\sigma_x \times \sigma_y}$$

این ضریب به واحد اندازه‌گیری داده‌ها بستگی ندارد و همواره مقداری بین -1 و $+1$ به خود می‌گیرد. در عبارت، صورت کسر کواریانس x و y و مخرج آن کواریانس x و کواریانس y را نشان داده شده است.

۷-۶-۲) رگرسیون غیرخطی و دیگر رگرسیون‌ها

بسیاری از مسائل رگرسیون وجود دارند که در آغاز غیرخطی هستند، ولی می‌توانند به شکل رگرسیون خطی چندگانه تبدیل شوند. برای مثال یک رابطه‌ی چند جمله‌ای نظیر:

$$y = w_0 + w_1x_1 + w_2x_2 + w_3x_1x_3 + w_4x_2x_3$$

می‌تواند به وسیله‌ی تنظیم مقادیر جدید $x_4 = x_1x_3$ و $x_5 = x_2x_3$ به شکل خطی تبدیل شود.

یا منحنی یک جمله‌ای درجه‌ی سوم زیر را در نظر بگیرید.

$$y = w_0 + w_1x_1 + w_2x^2 + w_3x^3$$

با جایگزینی $x_1 = x$ و $x_2 = x^2$ و $x_3 = x^3$ می‌توان آنرا خطی و تبدیل به رگرسیون چندگانه نمود تا با روش مربعات قابل حل باشد.

مدل رگرسیون تعمیم یافته نشان می‌دهد که تئوری رگرسیون خطی می‌تواند برای برچسب کلاس‌های گستته نیز استفاده شود. یکی از مدل‌های خطی تعمیم یافته رگرسیون لجستیک نام دارد. رگرسیون لجستیک احتمال وقایع اتفاق افتاده‌ای که می‌توان آنها را بصورت یک تابع خطی از مجموعه‌ی صفات خاصه بیان کرد، مدل می‌کند. در این مدل به جای تخمین مقادیر برچسب برای کلاس، سعی می‌شود احتمال آن پیش‌گویی شود. چنانچه احتمال تخمین زده شده مقداری بیشتر از 50% داشته باشد، جواب پیش‌گویی مثبت و در غیر این صورت منفی است. به همین دلیل هنگامی از این رگرسیون استفاده می‌شود که برچسب کلاس دارای دو مقدار است.

همچنین می‌توانیم با تغییراتی در ساخت الگوریتم‌های درخت تصمیم آنها را به درخت رگرسیون تبدیل کنیم. در هر درخت رگرسیون برگ‌ها به جای تخمین برچسب کلاس، مقدار پیوسته‌ای (عددی) را تخمین می‌زنند. در واقع این مقدار برابر با میانگین مقدار صفت خاصه‌ی نمونه‌های تخمین‌زده شده‌ای است که به برگ درخت می‌رسند.

۷-۷) روش‌های دیگر برای طبقه‌بندی

در این بخش درباره‌ی چندین روش طبقه‌بندی بصورت خلاصه بحث می‌شود. اینکه این روش‌ها بطور خلاصه بررسی می‌شوند به این دلیل نیست که از اهمیتی برخوردار نیستند، بلکه هر یک از این مباحثت می‌تواند عنوان یا عنوانی چندین کتاب را تشکیل دهدن. به همین دلیل و عدم فضای کافی در کتاب حاضر فقط به اشاره‌ای بسنده می‌کنیم. انتهای فصل چندین منبع جهت مطالعه‌ی بیشتر معرفی شده است.

۷-۷-۱) شبکه‌های عصبی

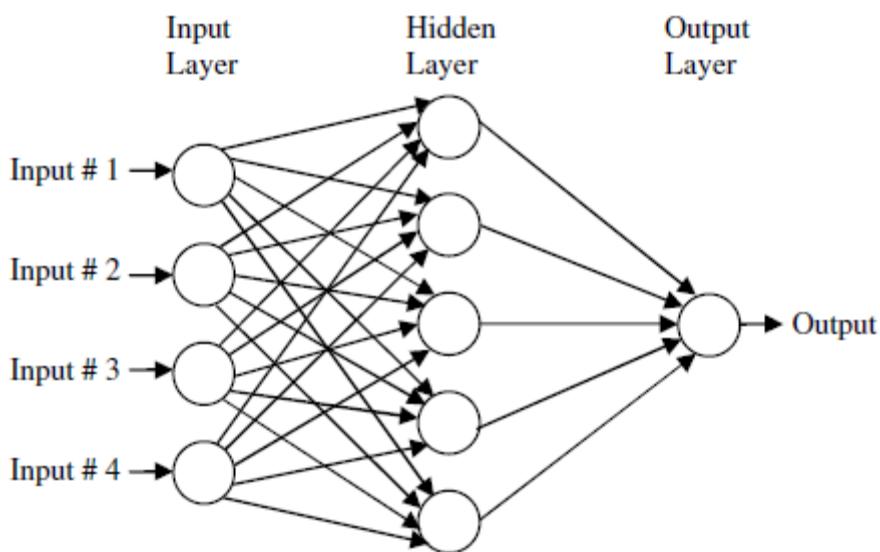
شبکه‌های عصبی نوعی مدل‌سازی ساده از سیستم‌های عصبی واقعی هستند. حوزه‌ی کاربرد این شبکه‌ها آنچنان گسترده است که از کاربردهای طبقه‌بندی گرفته تا کاربردهایی نظیر تخمین و آشکارسازی و ... را شامل می‌شود. در حقیقت یک شبکه مجموعه‌ای از ساختارهای ساده است که در کنار یکدیگر سیستم پیچیده‌ی نهایی را توصیف می‌کنند. مجموعه‌ای از گره‌ها و اتصالات بین آنها دو مولفه‌ی اصلی تشکیل دهنده‌ی شبکه‌ها هستند. هر گره واحد محاسباتی شبکه محسوب می‌شود که می‌تواند پردازشی ساده و یا پیچیده را بر روی ورودی‌ها انجام دهد. در شبکه‌های عصبی مصنوعی^۱ هر گره به عنوان یک نرون مصنوعی در نظر گرفته می‌شود. هر چند می‌توان گفت نحوه‌ی مدل کردن نرون از اساسی‌ترین نکات کلیدی در کارایی شبکه‌ی عصبی است اما نحوه‌ی برقراری اتصالات شبکه نیز فاکتور مهمی است.

یکی از ساده‌ترین و در عین حال کارآمدترین اتصال پیشنهادی، مدل پرسپترون چند لایه^۲ یا به اختصار *MLP* است که از یک لایه‌ی ورودی، یک یا چند لایه‌ی پنهان^۳ و یک لایه‌ی خروجی تشکیل یافته است (شکل ۷-۱۰). تعداد نرون‌های هر لایه مستقل از تعداد نرون‌های لایه‌های دیگر است.

¹ Artificial Neural Networks

² Perceptron Multilayer

³ Hidden Layer



شکل ۷-۱۰: نمونه‌ای از یک شبکه‌ی عصبی با ۴ ورودی و ۱ خروجی و ۱ لایه‌ی پنهان

بطور کلی شبکه‌های عصبی از لحاظ یادگیری در دو دسته‌ی شبکه‌های وزن ثابت و شبکه‌هایی با وزن متغیر تقسیم می‌شوند. شبکه‌هایی با وزن متغیر که به نام شبکه‌های یادگیرنده نیز شناخته می‌شوند، خود می‌توانند به دو دسته‌ی هدایت شده^۱ و هدایت نشده^۲ تقسیم می‌شوند. به منظور طبقه‌بندی می‌توانیم از شبکه‌های عصبی هدایت شده استفاده کنیم که همانند روش‌های طبقه‌بندی در داده‌کاوی مرحله‌ای به نام آموزش را با مجموعه نمونه‌های برچسب‌دار (برچسب کلاس) می‌گذرانند. هدف از فرآیند آموزش رسیدن به خروجی مطلوب است. در حالیکه در شبکه‌های هدایت نشده بر اساس یک معیار (بطور مثال فاصله) و بر اساس نوعی رقابت، خروجی مورد نظر در کلاس جداگانه قرار می‌گیرد. به منظور آموزش شبکه روش‌های بسیار زیادی وجود دارند که یکی از مشهورترین آن الگوریتم پس انتشار خط‌ها^۳ است که از روش‌های هدایت شده تلقی می‌شود. واژه‌ی پس انتشار به معنای این است که خطاهای به سمت عقب در شبکه تغذیه می‌شوند تا وزن‌ها را اصلاح کنند. در این الگوریتم ابتدا فرض بر این است که وزن‌های شبکه بطور تصادفی انتخاب شده‌اند. در هر مرحله خروجی شبکه محاسبه می‌شود و بر حسب میزان اختلاف آن

¹ Supervised

² Unsupervised

³ Error Back Propagation Algorithm

با خروجی مطلوب وزن‌ها به گونه‌ای تصحیح می‌شوند تا در نهایت خطأ به حداقل می‌رسد. خطای کل شبکه به شکل مجموع خطاهای تک تک عصب‌های لایه‌ی خارجی نوشته می‌شود و از مربع تفاضل بین خروجی واقعی و خروجی مطلوب استفاده می‌شود. برای بررسی ارتباط خطأ با عواملی چون ورودی‌ها، وزن‌ها و خروجی‌ها روش‌های متفاوتی وجود دارند که روش گرادیان شیب^۱ از ساده‌ترین و پرکاربردترین آنها به شمار می‌رود. در عمل می‌توان نشان داد که یک شبکه‌ی عصبی سه لایه می‌تواند راندمانی مشابه با شبکه‌هایی با لایه‌های بیشتر داشته باشد. بنابراین از آنجا که افزایش تعداد لایه‌ها، الگوریتم یادگیری را پیچیده‌تر می‌کند، مرسوم است که از شبکه‌های عصبی سه لایه استفاده گردد.

برای شبیه‌سازی یک روش طبقه‌بندی بر روی شبکه‌های عصبی در لایه‌ی ورودی باید به تعداد صفات خاصه نرون قرار دهیم. در لایه‌ی خروجی نیز باید به تعداد کلاس‌ها نرون داشته باشیم. در حالت ایده‌آل برای هر نمونه داده، نرون مربوط به آن کلاس مقدار ۱ و بقیه‌ی نرون‌ها مقدار صفر به خود می‌گیرند. اما در عمل نرونی از لایه‌ی خروجی که مقدار بزرگتری دارد کلاس مربوط به آن داده را مشخص خواهد کرد.

شبکه‌های عصبی معمولاً زمان زیادی را برای آموزش نیاز دارند و برای کار کردن با آنها چندین پارامتر مثل ساختار شبکه هست که بصورت تجربی بهترین آنها انتخاب می‌شوند. در ضمن تفسیر و فهم آنها توسط کاربران ساده نیست. یکی از مزایای آنها قابلیت تحمل بالای شبکه‌های عصبی در مقابل با داده‌های نویز است و هنگامی که دانش کمی از ارتباط میان صفات خاصه و کلاس‌ها وجود دارد، روش مناسبی به نظر می‌رسند. قابلیت پیاده‌سازی موازی خوبی دارند و همچنین برای ورودی و خروجی‌های نوع عددی بسیار مناسبند.

۷-۷-۲) الگوریتم‌های ژنتیک

الگوریتم‌های ژنتیک که بر اساس مکانیسم علم ژنتیک طبیعت بنا نهاده شده است، بدليل توانایی در حل مسایل بهینه‌سازی سخت مورد توجه قرار گرفته است. فرایند کلی این الگوریتم‌ها به ترتیبی است که در ابتدا کار خود را با جمعیت اولیه‌ای از نمونه‌ها بصورت

^۱ Gradient Descent

تصادفی آغاز می‌کند. هر یک از نمونه‌های موجود در جمعیت نشان‌دهنده‌ی یک جواب بالقوه مسئله مورد بحث است. نمونه‌ها طی تکرارهای متوالی به نام نسل تکامل یافته و هر نسل با معیارهایی ارزیابی می‌شود. نسل‌های بعدی توسط عملگرهای ژنتیکی(مانند جهش^۱) ساخته شده و فرآیند تکرار تا رسیدن به شرط پایانی ادامه می‌یابد. پس از گذشت تعدادی نسل، الگوریتم به سمت بهترین یا تقریباً بهترین راه حل همگرا می‌شود. بهر حال جهت استفاده از الگوریتم‌های ژنتیک لازم است به جزئیات بیشتری توجه شود. برای مثال روش ساختن جمعیت اولیه، فرآیند ارزیابی هر نسل، عملگرهای ژنتیکی و تنظیم پارامترهای مختلف در این الگوریتم‌ها از جمله جزئیاتی هستند که برای استفاده باید به آنها توجه شود.

الگوریتم‌های ژنتیک به جای آنکه عملیاتی را بروی راه حل‌ها انجام دهند، با یک شکل کدگذاری شده از آنها درگیر هستند. برای مثال فرض کنید در مجموعه داده‌هایی با دو صفت خاصه A_1 و A_2 که دامنه‌ی هر یک دارای دو مقدار هست و یک برچسب کلاس که می‌تواند دو مقدار c_1 و c_2 را به خود بگیرد، وجود داشته باشد. جهت کدگذاری شروط طبقه‌بندی مانند شرط زیر:

If A_1 and A_2 Then c_1

می‌توان از یک رشته‌ی سه بیتی استفاده نمود که هر یک از بیت‌ها نشان دهنده‌ی صفات خاصه و برچسب کلاس هستند. برای مثال رشته بیت 111 ممکن است دلالت بر شرط فوق داشته باشد و 100 می‌تواند نشان دهنده‌ی شرط زیر باشد.

If A_1 and not A_2 Then c_2

بدین ترتیب و با تعیین یکتابع ارزشیابی^۲ مناسب در واقع اصول اولیه‌ی الگوریتم ژنتیک جهت اجرا آمده شده‌اند. اگر صفت خاصه‌ای دارای بیش از دو مقدار باشد، تعداد بیت‌های بیشتری نیز برای کدگذاری نیاز است. در این مثال تابع ارزشیابی می‌تواند دقت طبقه‌بندی مجموعه داده‌های آموزشی لحاظ گردد.

¹ Mutation

² Fitness Function

الگوریتم‌های ژنتیک همچنین مناسب برای موازی‌سازی هستند و قابلِ اعمال بر روی مسائل بهینه‌سازی پیوسته و نیز گستته می‌باشند. در کاربردهای داده‌کاوی این الگوریتم‌ها می‌توانند جهت ارزشیابی دیگر تکنیک‌های داده‌کاوی نیز استفاده شوند. عدم توانایی الگوریتم‌های ژنتیک در پیدا کردن راه حل بهینه یا حتی نزدیک به راه حل بهینه در شرایط خاص یکی از معایب این روش محسوب می‌شود. چنین عدم موفقیت‌هایی توسط همگرایی‌های زودرس نسبت به یک جواب محلی صورت می‌گیرد. البته مسئله‌ی فوق نه تنها برای دیگر الگوریتم‌های بهینه‌سازی یک امر متداول محسوب می‌شود، بلکه برای برخی از تکنیک‌های داده‌کاوی نیز مرسوم است.

۳-۷-۷) مجموعه‌های فازی

مفاهیم فازی^۱ از پدیده‌های فازی که معمولاً در دنیای واقعی روی می‌دهند، سرچشممه می‌گیرند. در اغلب تجزیه و تحلیل‌ها فرض می‌شود که ما با معیارهای دقیق سروکار داریم و این دقت توسط بیان پدیده‌ها بصورت مقادیر عددی یا طبقه‌بندی شده بدست می‌آید. اگر چه استفاده از مقادیر ثابت شده کاربردهای زیادی دارند، اما طبیعت مفاهیم بشری را منعکس نمی‌کنند، زیرا این مفاهیم به انتزاعی و غیر دقیق بودن تمایل بیشتری دارند. برای مثال فرض کنید در یک مجموعه داده صفت‌خاصه‌ی x قد افراد را نشان می‌دهد و عبارت $180 < x \leq$ به منظور تعیین افراد بلند قد استفاده می‌شود. این بین معنی است که افرادی با قد بیشتر از 180 سانتی‌متر بلند قد تلقی می‌شوند. توجه کنید شخصی با مقدار قد $180/001$ بلندقد در نظر گرفته می‌شود در حالیکه یک شخص 180 سانتی‌متری با عبارت شرطی فوق در مجموعه‌ی بلندقدها قرار نمی‌گیرد. این تمایز با توجه به شرط درست است اما از نظر عقلانی و درک مستقیم غیرمنطقی است. ایراد از تبدیل صریح بین عضو بودن یا نبودن در مجموعه‌ی بلندقدها ناشی می‌گردد. یک مجموعه‌ی فازی، مجموعه‌ای بدون مرز مشخص است. یعنی تبدیل از متعلق بودن به یک مجموعه به متعلق نبودن به مجموعه تدریجی است و با کمک توابع توصیف می‌شود.

^۱ Fuzzy

بنابراین برخلاف تئوری مجموعه‌ها در ریاضی که یک عضو یا متعلق به یک مجموعه است یا خیر، در تئوری مجموعه‌ی فازی اعضاء می‌توانند به بیش از یک مجموعه‌ی فازی متعلق باشند. برای مثال شخص مذکور با قد ۱۸۰/۰۰۱ هم متعلق به مجموعه‌ی فازی اشخاص بلند قد می‌شود و هم متعلق به کوتاه قدها، اما با درجات مختلف. عبارت زیر این مسئله را با مفاهیم مجموعه‌ی فازی نشان می‌دهد.

$$M_{tall}(180.001)=0.95 \text{ and } M_{short}(180.001)=0.16$$

که در آن M تابع عضویت برای مجموعه‌ی فازی نامیده می‌شود و هر کدام از عناصر مجموعه را به یک درجه‌ی عضویت بین صفر و یک نگاشت می‌کند. برخلاف تئوری احتمالات که در آن مجموع احتمالات برابر با یک است، مجموع مقادیر عضوها (درجه‌ی اعضاء) در مجموعه‌ی فازی لازم نیست برابر با یک باشد. با عبارت فوق شخصی با قد ۱۸۰/۰۰۱ به احتمال ۹۵ درصد در اعضاء بلندقدها و به احتمال ۱۶ درصد میان کوتاه قدها قرار می‌گیرد. تئوری مجموعه‌ی فازی می‌تواند همراه با تکنیک‌های داده‌کاوی استفاده شود. روش‌های طبقه‌بندی مبتنی بر یافتن قوانین را در نظر بگیرید. می‌توان عملیاتی را جهت ترکیب قوانین و مجموعه‌ی فازی استخراج نمود. اجتماع و اشتراک از اصلی‌ترین عملیات بر روی مجموعه‌های کلاسیک هستند که متناظر آنها نیز در مجموعه‌های فازی قابل تعریف است. فرض کنید مجموعه داده‌ها دارای دو صفت‌خاصه قد و سن افراد است، که هر یک به ترتیب دامنه‌ای برابر با $\{\text{بلند، کوتاه}\}$ و $\{\text{پیر، جوان}\}$ دارند. قانونی را در نظر بگیرید که در قسمت شرط آن بلند قد بودن را همراه با جوان بودن برای نمونه‌ی x آزمایش می‌کند. چنانچه این دو شرط با عملگر منطقی and به یکدیگر متصل شوند، حداقل مقدار میان دو تابع عضویت آنها به عنوان نتیجه برگردانده می‌شود.

$$M_{(tall \text{ and } young)}(x)=\text{Min}(M_{tall}(x), M_{young}(x))$$

به عبارت دیگر اشتراک دو مجموعه‌ی فازی، مجموعه‌ی فازی دیگری است که تابع عضویت آن بر اساس عبارت فوق محاسبه می‌شود. اجتماع دو مجموعه‌ی فازی نیز به طور مشابه تعریف می‌شود. آن هنگامی است که دو شرط قانون ما با عملگر منطقی or

با یکدیگر پیوند خورده باشند. بدین ترتیب حداکثر مقدار میان دوتابع عضویت برگزیده می‌شود.

$$M_{(tall \text{ or } young)}(x) = \text{Max}(M_{tall}(x), M_{young}(x))$$

عملیات ساده و مفید دیگری را نیز می‌توان بر روی مجموعه‌های فازی انجام داد. علیرغم اینکه منطق فازی بر پایه‌ی ریاضیات پیشرفت‌ته بنا نهاده شده است، اما یادگیری آن بسیار آسان است و در کاربردهای متفاوتی می‌توان از آن استفاده نمود.

خلاصه فصل

پس از آنکه در فصل قبل به مفاهیم اولیه‌ی طبقه‌بندی پرداختیم، در این فصل به شرح برخی از الگوریتم‌های آن رسیدیم. درخت‌های تصمیم از رایج‌ترین روش‌های طبقه‌بندی محسوب می‌شود، چرا که قابل فهم بودن خروجی آن توسط کاربران غیرحرفه‌ای از مزیت اصلی آن به شمار می‌رود. تنوع معیارهای انتخاب صفات خاصه‌ی بهتر الگوریتم‌های مختلفی را باعث شده است. پس از ساخت هر درخت تصمیم مشابه هر روش طبقه‌بندی نوبت به ارزشیابی آن می‌رسد که با کمک معیارهایی نظیر دقت می‌توان این کار را انجام داد. حذف تعدادی از شاخه‌ها و زیردرخت‌ها و جایگزینی آنها با برگ‌هایی که نظر اکثربت را نشان می‌دهند، هرس کردن درخت نام دارد که به دو صورت پیش‌هرس و پس‌هرس انجام می‌شود. کنترل مقادیر ناقص و مفقود نیز چالش دیگری در الگوریتم‌های ساخت درخت تصمیم است.

یکی از فرمول‌های مهم احتمال، فرمول احتمال بیز 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. روش دیگری که برای طبقه‌بندی داده‌ها استفاده می‌شود، استخراج مجموعه‌ای از قوانین به صورت شرط است. هر درخت تصمیم می‌تواند توسط مجموعه‌ای از این شروط نمایش داده شود، اما می‌توانیم این شروط را به طور مستقیم از مجموعه داده‌های آموزشی استخراج کنیم. برای یافتن بهترین قوانین یا شروط از دو معیار پوشش و دقت استفاده کردیم.

به جرأت می‌توان گفت الگوریتم‌های *SVM* از دقیق‌ترین و نیرومند‌ترین الگوریتم‌های داده‌کاوی به شمار می‌روند. 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 داده‌ها تبدیل شده‌اند. چنانچه بخواهیم خلاصه بیان کنیم، الگوریتم‌های *SVM* با کمک یک نگاشت غیرخطی فضای مجموعه داده‌های آموزشی را به یک بعد بالاتر تبدیل می‌کنند و سپس در این بعد به دنبال آبرصفحه‌ای هستند که نمونه‌های یک کلاس را از کلاس دیگر جدا کند. الگوریتم‌های *SVM* جهت یافتن این آبرصفحه از مفاهیمی چون بردارهای پشتیبان و حاشیه‌ها استفاده می‌کنند.

اغلب روش‌های طبقه‌بندی برای تخمین کلاس یک نمونه‌ی آزمایشی ابتدا مدلی را طراحی می‌کنند و پس از آن با استفاده از مدل، برچسب کلاس نمونه‌ی خواسته شده را تخمین می‌زنند. اما در یادگیرنده‌های تنبیل بدون ساختن مدل، روش با مقایسه‌ی نمونه‌ی آزمایشی و مجموعه داده‌ها و یافتن مشابه‌ترین نمونه قادر است کلاس داده‌ی آزمایشی را تخمین بزند. این دسته از الگوریتم‌ها به تکنیک‌های کارایی جهت ذخیره‌سازی و بازیابی داده‌ها 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 از خود نشان می‌دهند. یکی از معروف‌ترین این روش‌ها k نزدیکترین همسایه نام دارد.

در بیشتر روش‌های طبقه‌بندی برچسب کلاس از نوع داده‌ی گسسته (غیر عددی) هست. اگر چه در برخی از آنها با کمی تغییر می‌توان روش را برای تخمین کلاس‌های پیوسته (عددی) توسعه داد، اما روش‌های رگرسیون یکی از معروف‌ترین تکنیک‌های آماری به حساب می‌آیند که برای این کار بسیار مناسب به نظر می‌رسند. هدف تحلیل رگرسیون تعیین بهترین مدلی است که چگونگی ارتباط یک متغیر را با یک یا چند متغیر دیگر تعیین می‌کند. در بسیاری از کاربردهای عملی نیاز به پیش‌بینی مقدار یک متغیر (وابسته) از روی مقادیر چند متغیر (مستقل) بسیار رایج است.

استفاده از شبکه‌های عصبی، الگوریتم‌های ژنتیک و همچنین منطق فازی می‌تواند در روش‌های داده‌کاوی از جمله طبقه‌بندی مفید باشد. هر کدام از آنها دارای مزایا و

محدودیت‌هایی هستند که تحلیل‌گر با آگاهی از آن می‌تواند راهکار مناسب را جهت تحلیل داده‌ها انتخاب کند.

مراجع و منابع جهت مطالعه‌ی بیشتر

در دو کتاب [BFOS84] و [Qui93] به ترتیب می‌توانید توضیحات مفصلی در مورد الگوریتم‌های CART و C4.5 پیدا کنید. در ضمن این دو کتاب مطالب بسیار زیادی در مورد چگونگی ساخت درخت‌های تصمیم دارند. برای هر یک از الگوریتم‌های موجود، جهت ساخت درخت تصمیم منابعی را می‌توان یافت که در ادامه برای هر یک از آنها مراجع اصلی ذکر می‌شوند. برای الگوریتم FACT به [LV88]، برای الگوریتم QUEST به [LS97]، برای الگوریتم PUBLIC به [RS98]، برای الگوریتم INFERULE به [Kas80] [Mag94]، برای الگوریتم CHAID به [UFS91]، برای الگوریتم KATE به [MK91]، برای نسخه‌های افزایشی الگوریتم‌های ID3، ID4 و ID5 به [UBC97] [Utg88] [SF86]، برای نمونه نسخه‌ی افزایشی الگوریتم CART به [Cra89]، برای الگوریتم BOAT به [GGRL99]، برای الگوریتم مقیاس‌پذیر SLIQ به [MAR96]، برای الگوریتم RainForest به [SAM96] و برای الگوریتم مقیاس‌پذیر SPRINT به [GRG98] می‌توانید رجوع کنید.

در [KH97] به صورت مفصل در مورد انتخاب صفات خاصه جهت ساخت درخت تصمیم صحبت شده است، اما منابع دیگری نیز وجود دارند که به صورت خاص به معرفی و بررسی برخی از معیارها جهت انتخاب صفات خاصه می‌پردازند که [BFOS84] [Kon95] [Shi99] [Qui93] و [FI92] از آن جمله هستند.

الگوریتم‌های متعددی نیز برای هرس کردن درخت‌های تصمیم وجود دارند که هر کدام عملکرد مربوط به خود را دارند. از منابع موجود در این زمینه می‌توان به [RS98] [MFS95] [RS98] [MAR96] [QR89] [BFOS84] و [BA97] اشاره نمود. در ضمن [Mur98] مروری کلی و کامل بر روی مسائل ساخت درخت تصمیم دارد.

طبقه‌بندی با کمک قانون بیز و موضوعات مرتبط با آن نیز در [Mit97] [DHS01] و [WK91] و [DP96] بررسی شده‌اند. شاید بتوان ادعا نمود در فصل جاری جای بحث در مورد شبکه‌های بیز خالی است. منابع [Dar10] و [Hec96] توضیحی مقدماتی در این باب دارند و منابع [RN95] [HGC95] [Bun94] [CH92] [KF09] و [Jen96] هم در این مورد مطالعاتی را نشان می‌دهند.

روش‌های مبتنی بر تولید شروط راهکار دیگری برای طبقه‌بندی هستند و در این زمینه نیز الگوریتم‌های متفاوتی پیشنهاد شده است. برای اطلاع در مورد روش AQ15 به [SG92]، روش ITRULE به [CN89]، روش CN2 به [HMM86]، روش RISE به [FW94]، روش RIPPER به [Dom94]، روش FOIL به [Coh95] و روش Swap-1 به [Qui90] و [WI98] رجوع کنید. در [MM95] نیز معیارهایی جهت انتخاب قوانین قوی از میان مجموعه قوانین ارائه شده است.

اولین مقاله در مورد الگوریتم‌های SVM در سال ۱۹۹۲ به چاپ رسید [BGV92]. کتاب‌های [Vap95] و [Vap98] جزئیات بیشتری را در این مورد ارائه می‌دهند. مرجع [Bur98] با تشریح مفاهیم این روش می‌تواند شروع خوبی را برای علاقه‌مندان به دنبال داشته باشد، اما کتاب‌های [Hay08] [Kec01] و [CS-T00] نیز در این زمینه جزء منابع خوب به حساب می‌آیند. کتابخانه‌ای از نرم‌افزارهای مرتبط با SVM را می‌توانید در آدرس www.csie.ntu.edu.tw/~cjlin/libsvm جستجو کنید.

کاوش الگوهای مکرر نیز می‌تواند به روش‌های طبقه‌بندی کمک کند. یک قانون انجمانی را در نظر بگیرید که سمت راست آن برچسب کلاس را در خود دارد. الگوریتم‌های زیادی در این زمینه پیشنهاد شده‌اند. در [LHM98] الگوریتم CBA، در [LHP01] الگوریتم CPAR، در [YH03b] الگوریتم CMAR و در [CTTX05] الگوریتم RCBT ارائه می‌شود. منابع [CYHH07] و [CYHH08] روش‌های طبقه‌بندی با کمک یافتن الگوهای مکرر را بررسی نموده‌اند.

روش نزدیکترین همسایه در سال ۱۹۵۱ در [FH51] معرفی شد. مجموعه‌ی کاملی از مقالات در این زمینه را می‌توانید در [Das91] پیدا کنید. مراجع [DHS01] و [Aha92] نیز درباره‌ی این روش مطالبی را در خود دارند.

چندین کتاب در مورد شبکه‌های عصبی نوشته شده است و بعضی از کتاب‌ها نیز فصل کاملی را برای این موضوع تخصیص داده‌اند. برخی از آنها عبارتند از [RM86] [Hay99] [Rip96] [Bis95] [CR95] [HKP91] [HN90] و [Bis95]. مقلاطی نیز هستند که حگمنگ استخراج شمطا از شبکه‌های عصبی دارند که دهانه

برای اطلاعات درباره الگوریتم‌های ژنتیک به [Mit96] [Mic92] و [Gol89] هستند که چگونگی استخراج شروط از شبکه‌های عصبی را بررسی کرده‌اند. رجوع کنید. تئوری منطق فازی نیز توسط لطفی‌زاده در [Zad65] و [Zad83] معرفی شد و منابع [YZ94] و [Kec01] نیز در این زمینه بحث کرده‌اند.

همانطور که می‌دانید روشی ارجح را نمی‌توان در مجموعه الگوریتم‌های طبقه‌بندی یافت و هر یک از آنها دارای مزایا و همچنین محدودیت‌هایی هستند. نتایج تجربی حاصل از مقایسه‌ی روش‌های طبقه‌بندی در [MST94] [CM94] [BCP93] [SMT91] [LLS00] و [BU95] مطالعه شده است.

مطالب وعناوین پیشرفتی مانند طبقه‌بندی داده‌های چند کلاسه و یا طبقه‌بندی نیمه نظارت شده و... را می‌توانید در منابع مختلف و جدیدی پیدا کنید، اما [TD02] [HT98] [DB95] [CSZ06] [ASS00] [Zhu05] [PY10] [Set10] و [DYXY07] منابع خوبی برای تحقیق در این زمینه به شمار می‌روند.

فصل هشتم

خوشه‌بندی

خوشه‌بندی^۱ یک تابع کاوشی ناظارت‌نشده‌ی^۲ 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 منحصرً 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

¹ Clustering

² Unsupervised

اعضای دیگر خوش‌ها برخوردارند. معیار شباهت در اینجا فاصله بوده و نحوه محاسبه‌ی این فاصله در خوش‌بندی بسیار مهم است. فاصله که همان معرف عدم تجانس است به ما کمک می‌کند در فضای داده‌ای حرکت کنیم و خوش‌ها 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 خوش قرار داد. خوش‌بندی 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 خوش‌های معنی دارد.

در این فصل ابتدا به اهمیت و انگیزه‌ی خوش‌بندی می‌پردازیم. معیارهای تشابه میان داده‌های متفاوت بحث بعدی را تشکیل می‌دهد. در انتهای چندین الگوریتم خوش‌بندی مبتنی بر افزار و سلسله‌مراتبی بررسی می‌شوند. الگوریتم‌های مبتنی بر تراکم، مبتنی بر مدل و همچنین مبتنی بر شبکه‌های شترنجی گردید موضوع‌های بحث فصل بعدی هستند که تحت عنوان مباحث پیشرفته در خوش‌بندی مطرح می‌شوند.

۱-۸) اهمیت و انگیزه‌ی خوش‌بندی

خوش‌بندی می‌تواند در کاربردهای زیادی مورد استفاده قرار گیرد. شاخه‌هایی همچون هوش مصنوعی، آمار، زیست‌شناسی، یادگیری ماشین، تشخیص الگو... را می‌توان نام برد. اگر چه تکنیک‌های طبقه‌بندی^۱ به منظور تشخیص گروه‌ها یا کلاس‌های هر نمونه روش موثری است، اما فراموش نکنید در این گونه روش‌ها نمونه‌های ما دارای یک برچسب کلاس هستند، تا الگوریتم بتواند با کمک آن مدلی را طراحی کند.

در خوش‌بندی روش به طریقه‌ی دیگری عمل می‌کند. ابتدا مجموعه‌ی داده‌ها با کمک معیارهای ارزیابی شباهت در گروه‌هایی قرار گرفته و سپس به هر گروه برچسبی به عنوان کلاس زده می‌شود. با کمک خوش‌بندی خودکار، مناطق متراکم و غیرمتراکم از فضای نمونه داده‌های ما مشخص می‌شوند و همچنین الگوی توزیع نمونه‌ها و همبستگی میان صفات خاصه‌ی هر نمونه هویدا می‌شود. خوش‌بندی به منظور کشف داده‌های خارج از

^۱ Classification

محدوده^۱ نیز مورد استفاده قرار می‌گیرد. برای مثال با خوشبندی داده‌های کارت‌های اعتباری و تراکنش‌های آن می‌توان خریدهای مکرر و گران را تشخیص و پس از آن برای کشف موارد خاص (احتمال وقوع جرم) بررسی 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 کلاسیک خوشبندی هستند که این موضوع اهمیت مسئله را در تحلیل داده‌ها بیش از پیش مشخص می‌سازد.

یک الگوریتم خوشبندی می‌تواند دارای مشخصات مطلوب زیر باشد:

- قابلیت مقیاس‌پذیری:^۲ بسیاری از الگوریتم‌های خوشبندی عملکردی مناسبی بر روی داده‌هایی با حجم کم دارند، در صورتی که امروزه حجم بالای داده‌ها معمولاً از پیش‌فرض‌های موجود در تکنیک‌های داده‌کاوی است. استفاده از نمونه‌های کمتر داده‌های حجیم نیز باعث می‌شود تا ما به نتایج جهت‌گیرانه گرایش داشته باشیم. به همین دلیل در الگوریتم‌های خوشبندی به قابلیت مقیاس‌پذیری بالایی نیاز داریم.
- توانایی مواجهه با انواع داده‌ها: اکثر الگوریتم‌های خوشبندی به منظور کار با داده‌های عددی مناسبند. چرا که معنی و مفهوم فاصله و تشابه در این نوع از داده‌ها به خوبی قابل تعریف است. اما همانطور که می‌دانید نمونه‌ها فقط با این نوع از داده‌ها توصیف نمی‌شوند و الگوریتم خوشبندی باید توچایی مواجهه با انواع دیگر داده‌ها همچون داده‌های اسمی را نیز داشته باشد.
- استخراج خوشبندی به هر شکل دلخواه: استفاده از پارامترهای اندازه‌گیری فاصله در تعیین خوشبندی نقش مهمی را ایفا می‌کند. برخی از این معیارهای اندازه‌گیری مانند فاصله‌ی اقلیدسی ما را در یافتن خوشبندی با شکل خاص (بطور مثال کروی) سوق می‌دهد. حال آنکه بسیاری از داده‌ها می‌توانند در خوشبندی با اندازه‌ها و تراکم‌های نامساوی قرار بگیرند و این خیلی مهم به نظر

¹ Outlier

² Scalability

می‌رسد که الگوریتم خوشبندی توانایی تشخیص این خوشبها را نیز داشته باشد.

- توانایی مقابله با داده‌های نویز، نادرست و ناقص: هر چند در مرحله‌ی آماده‌سازی و پیش‌پردازش داده‌ها سعی شده است تا داده‌های ناقص و نویز و خارج از محدوده تشخیص و به مراحل بعدی وارد نشوند. اما می‌دانیم که این کار با قطعیت و صحت صدرصد انجام نمی‌شود. بنابراین حساسیت بیش از حد الگوریتم‌های خوشبندی به این نوع از داده‌ها از معایب الگوریتم به شمار خواهد رفت.

- عدم حساسیت به ترتیب ورود داده‌ها: اینکه یک الگوریتم واحد با ترتیب‌های مختلفی از داده‌های یکسان نتایج به شدت متفاوتی تولید کند، قابل قبول به نظر نمی‌رسد. از طرفی برخی از الگوریتم‌ها پس از اجرا و به دست آوردن خوشبها قابلیت افزایش داده‌های جدید را ندارند. بدین معنی که با ورود داده‌های جدید، الگوریتم می‌بایست از ابتدا و برای کل داده‌ها اجرا گردد. توانایی اجرای الگوریتم فقط برای داده‌های جدید می‌تواند از مزایای الگوریتم‌های خوشبندی محسوب گردد. این دسته از الگوریتم‌ها با نام خوشبندی افزایشی^۱ شناخته می‌شوند.

- عدم نیاز به پارامترهای ورودی: بسیاری از الگوریتم‌های خوشبندی نیاز به پارامترهایی برای شروع کار خود دارند که این پارامترها توسط کاربر مشخص می‌گردد. تعداد خوشبها یکی از رایج‌ترین این پارامترهاست. نتایج بدست آمده از اینگونه الگوریتم‌ها می‌تواند به این پارامترها وابسته باشد و این از ضعف این دسته از الگوریتم‌ها سرچشم می‌گیرد. از طرفی دیگر تعیین این پارامترها توسط کاربر آن هم برای داده‌های بسیار حجمی نه تنها دشوار بلکه گاهی غیرممکن به نظر می‌رسد.

- پذیرش داده‌هایی با بعد بالا^۲: یک پایگاه داده یا انبار داده‌ها می‌تواند شامل چندین صفت خاصه باشد. معمولاً انبار داده‌ها به نام داده‌هایی با بعد بالا شناخته

¹ Incremental Clustering

² High Dimension

می‌شود. بسیاری از الگوریتم‌های خوشبندی فقط در مواجهه با تعداد کم ابعاد داده‌ها نتایج و عملکرد مناسبی از خود نشان می‌دهند. یافتن خوشبندی در داده‌هایی با ابعاد بالا به عنوان چالشی در الگوریتم‌های خوشبندی مطرح است.

- قابلیت یافتن خوشبندی مبتنی بر محدودیت¹: در بسیاری از کاربردهای واقعی، پیدا کردن گروه‌هایی از داده‌ها همراه با رفتار خوشبندی مناسب که بتواند محدودیت‌های مشخصی را برآورده سازد، ممکن است نتایج مطلوبی را به همراه داشته باشد. فراموش نکنید این محدودیت‌ها می‌توانند با پارامترهایی (نظیر تعداد خوشبندی) که جهت اجرای الگوریتم‌های خوشبندی استفاده می‌شود، بسیار متفاوت باشد.
- قابل فهم بودن نتایج الگوریتم: کاربران برنامه‌های کاربردی همیشه انتظار دارند تا خروجی‌های الگوریتم‌های آنها فهمیدنی و قابل درک و قابل اجرا باشد. شاید نیاز باشد تا خروجی این الگوریتم‌ها به طریقی مناسب و با کمک برنامه‌های کاربردی مناسب تفسیر شوند.

استراتژی‌های خوشبندی بسیار زیاد هستند و راهکارهای بسیار زیادی برای محاسبه شباهت میان نمونه داده‌ها در خوشبندی وجود دارد. برای روشن شدن موضوع فرض کنید شما با n نمونه از داده‌ها روبرو هستید و قصد دارید این نمونه‌ها را در تعداد c خوشبندی قرار دهید. تعداد حالت‌های ممکن برای خوشبندی را می‌توانید از فرمول زیر محاسبه کنید:

$$\frac{1}{c!} \sum_{i=1}^c (-1)^{c-i} \times \binom{c}{i} \times i^n$$

به سادگی و با جایگزینی اعدادی نظیر ۱۰۰ و ۵ به ترتیب برای n و c می‌توانید محاسبه کنید که این ۱۰۰ نمونه داده را قادر خواهد بود به 10^{67} طریق در ۵ خوشبندی قرار دهید. مشاهده می‌کنید حتی با تعداد نمونه‌های کم، تنوع روش‌های خوشبندی یک عدد نجومی است. بنابراین نیاز به استفاده از الگوریتم‌های خوشبندی به خوبی احساس می‌شود، چرا که فضای جستجویی نظیر مثال بالا به ما نشان می‌دهد که ارزیابی حالت‌های خوشبندی برای یک مسئله‌ی کوچک نه تنها زمانگیر که غیرممکن است.

¹ Constraint- based

۸-۲) الگوریتم‌های خوشبندی

تکنیک‌های خوشبندی 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.

- خوشبندی مبتنی بر افزار^۱
- خوشبندی سلسله مراتبی^۲
- خوشبندی مبتنی بر تراکم یا چگالی داده‌ها^۳
- خوشبندی مبتنی بر شبکه‌های شطرنجی گردید^۴
- خوشبندی مبتنی بر مدل^۵

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 نهای خود هستند.

در الگوریتم‌هایی که خوشبندی مبتنی بر افزار داده‌ها انجام می‌شود، همانطور که از نام آن مشخص است داده‌ها به درون چند خوشی جدا افزار می‌شوند. این بدین معنی است که تعداد خوش‌ها معمولاً^۶ در این الگوریتم‌ها به عنوان ورودی الگوریتم مشخص می‌شود و هر نمونه داده فقط عضو یک خوش می‌تواند باشد. قرارگرفتن هر نمونه در یک خوش توسط معیارهای تشابهی که الگوریتم مشخص می‌کند، ارزیابی می‌شود.

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

¹ Partitioning Method

² Hierarchical

³ Density- based

⁴ Grid- based

⁵ 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^۱ و یا تقسیم^۲ 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

¹ Merge

² Split

شمار می‌رود. معمولاً زمان اجرای تکنیک‌های خوشبندی 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 خوش‌ها 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 خوشبندی از راهکارهای آماری و یا شبکه‌های عصبی استفاده می‌شود.

همانطور که قبل از این به آن اشاره کردیم، برخی از تکنیک‌های خوشبندی از ایده‌های موجود در پنج تکنیک فوق بهره می‌برند و ما به سختی می‌توانیم آنها را در گروه‌بندی خاصی قرار دهیم. به طور مثال تکنیک‌هایی وجود دارند که قبل از خوشبندی، کاربر قادر است محدودیت‌هایی را تعریف و نتیجه‌ی نهایی را در راستای این محدودیت‌ها قرار دهد. یا بعضی از تکنیک‌های خاص خوشبندی کارایی بهتری در مواجهه با داده‌هایی با ابعاد بالا از خود نشان می‌دهند. این موضوع از اهمیت زیادی برخوردار است، چرا که برنامه‌های کاربردی زیادی می‌توان یافت که تعداد ابعاد و ویژگی‌های نمونه‌ها در آنها کم نیست. هر چند بسیاری از این صفات خاصه یا دارای مقدار نیستند و یا در تحلیل ما نقش کمتری بازی می‌کنند، اما بدون شک نیاز به یک تکنیک خاص خوشبندی در این مورد به خوبی احساس می‌شود.

انتخاب یک تکنیک مناسب خوشبندی متکی به نوع داده‌ها و همچنین هدف اصلی برنامه‌ی کاربردی شما دارد. به نظر می‌رسد استفاده از چند تکنیک مختلف و مشاهده و مقایسه‌ی نتایج آنها می‌تواند مفید باشد.

قبل از بررسی دقیق چند تکنیک خوشبندی توضیحی مختصر در مورد معیارهای تشابه داریم.

۳-۸) معیارهای تشابه و انواع داده‌ها

موضوع اصلی در تکنیک‌های خوشبندی تشابه^۱ و عدم تشابه^۲ 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 خوش‌های 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^۳ 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

¹ Similarity

² Dissimilarity (Distance)

³ 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^۱ 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 صفت خاصه‌ی یک نمونه را نشان دهد، داده‌های عددی هستند. منظور از داده‌های عددی مقادیر پیوسته مانند اعداد حقیقی است. مقادیر

^۱ 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^۱ 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 λ 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 ₁	Att ₂	Att ₃	Att ₄
O_1	4	5	4	8
O_2	7	8	3	1
O_3	3	4	5	3

¹ 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 مانهاتن^۱ (بلوک شهری^۲) 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_1 و O_3 بیشتر از تشابه میان نمونه‌های دیگر است، اما توجه کنید که همیشه اینطور نیست.

¹ Manhattan

² City Block

فاصله‌ی اقلیدسی و فاصله‌ی مانهاتان (بلوک شهری) شکل خاصی از معیار مینکوفسکی^۱ 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:

$$\text{Distance}(O_i, O_j) \leq \text{Distance}(O_i, O_k) + \text{Distance}(O_k, O_j)$$

فاصله‌ی چبی‌چف^۲ معیار دیگری است که فرمول آن همراه با محاسبه برای داده‌های

جدول ۱-۸ در زیر آمده است:

$$\text{Distance}(O_i, O_j) = \text{Max}_{k=1, 2, \dots, m} /|x_{ik} - x_{jk}|/$$

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

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

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

توجه کنید با انتخاب این معیار نمونه‌های O_2 و O_3 تشابه بیشتری دارند، در صورتی که در محاسبه‌ی دو معیار قبلی اینچنین نبود. در معیار چبی‌چف عدم تشابه بالا در یکی از

¹ Minkowski

² Chebychev

صفات خاصه کافی است تا ما ۲ نمونه‌ی تحت مقایسه را بسیار متفاوت بدانیم، حتی اگر در دیگر صفات خاصه تشابهاتی نزدیک دیده شود. معیارهای دیگری نیز وجود دارند که هر کدام تحت شرایط خاصی می‌توانند مفید باشند. به دو فرمول زیر برای محاسبه فاصله میان دو نمونه توجه کنید.

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

$$\text{Distance}(O_i, O_j) = \frac{\sum_{k=1}^m (x_{ik} \times x_{jk})}{[\sum_{k=1}^m x_{ik}^2 \times \sum_{k=1}^m x_{jk}^2]^{1/2}}$$

اولی به نام فاصله‌ی کانبرا^۱ و دومی تحت عنوان همبستگی کسینوسی شناخته می‌شود. فاصله‌ی کانبرا برای مقادیر مثبت محاسبه می‌شود. دقت کنید که به جز پیچیدگی محاسباتی تفاوت‌های دیگری نیز وجود دارد که این دو معیار را با معیارهای قبلی از همدیگر متمایز می‌سازد. برای مثال در همبستگی کسینوسی فاصله‌ی هر نمونه با خودش برابر با صفر نیست. یا مقدار فاصله در این معیار همیشه یک عدد مثبت نیست. می‌توانید نامعادله‌ی مثلثی را نیز برای معیار همبستگی کسینوسی بررسی کنید. اعداد زیر مقادیر معیار همبستگی کسینوسی را برای ۳ نمونه داده‌ی جدول ۱-۸ نشان می‌دهد.

$$\text{Distance}(O_1, O_2) = 0.72$$

$$\text{Distance}(O_2, O_3) = 0.83$$

$$\text{Distance}(O_1, O_3) = 0.90$$

برخی از مقیاس‌ها برای فاصله وجود دارند که تأثیر نقاط مجاور در فضاهای چند بعدی نمونه‌ها را نیز در نظر می‌گیرند. برای مثال تشابه بین دو نمونه‌ی O_i و O_j با استفاده از

^۱ Canberra

فاصله‌ی همسایگی متقابل^۱ (MND) که با کمک فرمول زیر می‌توان اندازه‌گیری کرد، از این جمله مقیاس‌ها محسوب می‌شود.

$$MND(O_i, O_j) = NN(O_i, O_j) + NN(O_j, O_i)$$

به نحوی که $NN(O_i, O_j)$ رتبه‌ی همسایگی O_j نسبت به O_i است. بدین ترتیب که اگر O_i نزدیکترین نمونه به O_j باشد، پس مقدار $NN(O_i, O_j)$ برابر با ۱ و اگر O_i دومین نمونه‌ی نزدیک به O_j باشد مقدار $NN(O_i, O_j)$ برابر با ۲ خواهد بود و الی آخر. با کمک جدول ۸-۲ در ادامه یک مثال ساده از طریقه‌ی محاسبه‌ی روش MND را توضیح می‌دهیم. در این مثال از فضای دو بعدی و فاصله‌ی اقلیدسی استفاده کرده‌ایم.

جدول ۸-۲: مشخصات دکارتی ۶ نقطه

Point	X	Y
A	4	2
B	6	4
C	8	8
D	3	2
E	3	1
F	2	2

به منظور محاسبه‌ی $MND(A, C)$ داریم:

$$MND(A, C) = NN(A, C) + NN(C, A)$$

برای بدست آوردن مقدار $NN(A, C)$ باید بدانیم نقطه‌ی A چندمین نقطه‌ی نزدیک به C است. بدین ترتیب بایستی فاصله‌ی C با کلیه‌ی نقاط را بدست آوریم. لذا داریم:

$$Distance(C, A) = 7.21$$

$$Distance(C, B) = 4.47$$

$$Distance(C, D) = 7.81$$

$$Distance(C, E) = 8.60$$

$$Distance(C, F) = 8.48$$

^۱ Mutual Neighbor Distance

بنابراین A دومین نقطه‌ی نزدیک به C بعد از B است و با توجه به این نکته مقدار ۲ را برای $NN(A,C)$ لحاظ می‌کنیم.

برای بدست آوردن مقدار $NN(C,A)$ باید بدانیم نقطه‌ی C چندمین نقطه‌ی نزدیک به A است. مقادیر بدست آمده در زیر به ما کمک می‌کند تا به این مطلب پی ببریم.

$$Distance(A,C)=7.21$$

$$Distance(A,B)=2.83$$

$$Distance(A,D)=1.00$$

$$Distance(A,E)=1.41$$

$$Distance(A,F)=2.00$$

چهار نقطه‌ی B و E و D و F نزدیکتر از نقطه‌ی C به A هستند. بنابراین نقطه‌ی از C از نظر نزدیکی پنجمین نقطه به A خواهد بود و داریم:

$$NN(C,A)=5$$

با بدست آمدن دو مقدار فوق داریم:

$$MND(A,C)=NN(A,C)+NN(C,A)=2+5=7$$

با افزودن نقاط جدید و نزدیک به A فاصله‌ی همسایگی متقابل بین A و دیگر نقاط افزایش می‌باید، حتی اگر A و نقاط قبلی جایه‌جا نشده باشند. مقدار کوچکتر نشانه‌ی تشابه بهتر است. در ضمن نابرابری مثلثی برای این پارامتر صادق نیست. فاصله‌ی همسایگی متقابل به طور موققت‌آمیزی در چندین تکنیک خوشبندی استفاده می‌شود. همانند قبل ذکر این نکته ضروری است که انتخاب یکی از این معیارها وابسته به ماهیت داده‌ها و کاربردی دارد که تکنیک‌های داده‌کاوی بر روی آنها اجرا می‌شود. همانطور که ملاحظه کردید نتایج هر یک از آنها بر روی داده‌های یکسان نتایج متفاوتی را به همراه دارد.

۲-۳-۸) معیارهای تشابه در داده‌های دودویی(باینری)

شاید بتوان داده‌های دودویی را دسته‌ای از داده‌های طبقه‌بندی شده با دو گروه تلقی نمود. اما به دلیل اهمیت و کاربرد وسیع آن، بررسی معیارهای تشابه در اینگونه از داده‌ها را در یک بخش مجزا قرار داده‌ایم. بسیاری از ویژگی‌ها در دنیای واقعی دارای دو مقدار هستند،

مانند جنسیت افراد که با دو مقدار مرد و زن شناخته می‌شود. مقادیر بسیاری از ویژگی‌ها را می‌توان در دو محدوده‌ی گسسته قرار داد، مانند صفت‌خاصه‌ی قد که می‌تواند به صورت دو گروه کوتاهتر از ۱۷۰ سانتی‌متر و بلندتر از ۱۷۰ سانتی‌متر گسسته‌سازی شود. مثال دیگر آزمایش‌هایی هستند که در علم پزشکی وجود یا عدم وجود یک بیماری خاص را نشان می‌دهند. وجود و عدم وجود بیماری می‌تواند به صورت ویژگی‌هایی با مقادیر دودویی نمایش داده شوند. با این توضیح می‌توان به خوبی متوجه شد که تعریف معیارهای گوناگون تشابه در ویژگی‌هایی با مقادیر دودویی می‌تواند نقش مهمی را در تکنیک‌های خوشبندی بازی کند.

جدول ۸-۳ پنج نمونه داده را که هر یک دارای شش صفت‌خاصه هستند، توصیف می‌کند.

جدول ۸-۴ مشخصات ۵ نمونه همراه با ۶ صفت‌خاصه برای هر یک

Object	Height	Weight	Lips	Hair	Hand	Gender
O_1	155	65	Thin	Curly	Right	Female
O_2	172	75	Thick	Straight	Right	Male
O_3	162	68	Thin	Curly	Right	Male
O_4	168	62	Thick	Curly	Right	Female
O_5	175	80	Thick	Curly	Left	Male

با معرفی شش صفت‌خاصه‌ی A_1 تا A_6 به صورت زیر که به ترتیب متناظر با ویژگی‌های بیان شده در جدول ۸-۳ هستند، می‌توانیم آنرا تبدیل به جدولی با ویژگی‌های دودویی کنیم.

- اگر $Height \geq 165$ باشد مقدار A_1 برابر با یک درغیراینصورت برابر با صفر در نظر گرفته می‌شود.
- اگر $Weight \geq 70$ باشد مقدار A_2 برابر با یک درغیراینصورت برابر با صفر در نظر گرفته می‌شود.
- اگر $Lips = Thick$ باشد مقدار A_3 برابر با یک درغیراینصورت برابر با صفر در نظر گرفته می‌شود.

- اگر $Hair=Curly$ باشد مقدار A_4 برابر با یک درغیراينصورت برابر با صفر در نظرگرفته می‌شود.
 - اگر $Hand=Right$ باشد مقدار A_5 برابر با یک درغیراينصورت برابر با صفر در نظرگرفته می‌شود.
 - اگر $Gender=Male$ باشد مقدار A_6 برابر با یک درغیراينصورت برابر با صفر درنظرگرفته می‌شود.
- رعایت قواعد بالا جدول ۸-۳ را به جدول ۸-۴ تبدیل می‌کند.
- جدول ۸-۴: نمایش جدول ۸-۳ با کمک داده‌های دودویی

Object	Height	Weight	Lips	Hair	Hand	Gender
O_1	0	0	0	1	1	0
O_2	1	1	1	0	1	1
O_3	0	0	0	1	1	1
O_4	1	0	1	1	1	0
O_5	1	1	1	1	0	1

روش‌هایی که برای بدست آوردن مقادیر معیارهای تشابه بین دو نمونه‌ی نمایش داده شده با ویژگی‌های دودویی مانند جدول ۸-۴ استفاده می‌شود، محتویات یک ماتریس 2×2 است که یکسان بودن یا نبودن مقادیر این ویژگی‌ها را شمارش می‌کند. ماتریس شکل ۸-۲ پس از شمارش مقادیر یکسان و غیریکسان برای نمونه‌های O_3 و O_4 موجود در جدول ۸-۴ تشکیل شده است.

		O_3		
		1	0	
O_4	1	$a=2$	$b=2$	4
0		$c=1$	$d=1$	2
		3	3	6

شکل ۸-۲: نمونه‌ای از یک ماتریس شمارش برای نمونه‌های O_3 و O_4

با نگاه به ماتریس به سادگی می‌توانید مفهوم محتوای آن را دریابید. مقدار a برابر است با تعداد حالت‌هایی که مقادیر ویژگی‌های O_3 و O_4 مساوی و برابر با یک باشند. مقدار b به تعداد حالت‌هایی اطلاق می‌شود که مقدار ویژگی O_3 برابر با صفر و مقدار همان ویژگی در O_4 برابر با یک باشد. مقدار c از تعداد صفات خاصه‌ی دودویی که مقادیر آن در O_3 برابر با یک و در O_4 برابر با صفر است، بدست می‌آید. در نهایت مقدار d تعداد حالت‌هایی را نشان می‌دهد که در آن مقدار ویژگی‌ها در O_3 و O_4 مساوی و برابر با صفر می‌باشد.

هنوز هیچگونه پارامتر و معیاری جهت ارزیابی تشابه میان نمونه‌ها تعریف نشده است. چندین مقیاس براساس محتوای ماتریس شکل ۸-۲ می‌توان یافت تا با کمک آن معیاری برای سنجش تشابه داشته باشیم. یکی از معیارها تاکید بر تعداد حالت‌های یکسان بر روی صفات خاصه دارد که با توجه به محتویات ماتریس از فرمول زیر محاسبه می‌شود:

$$\frac{a+d}{a+b+c+d}$$

ماتریس شکل ۸-۳ تشابه زوج نمونه‌های جدول ۸-۴ را هنگامی که از این مقیاس استفاده شده است، نشان می‌دهد. در ماتریس مذبور عنصر سطر i ام و ستون j ام دلالت بر تشابه بین نمونه‌های O_i و O_j دارد.

$$\begin{bmatrix} 1 & & & & \\ 1/6 & 1 & & & \\ 5/6 & 2/6 & 1 & & \\ 4/6 & 3/6 & 3/6 & 1 & \\ 1/6 & 4/6 & 2/6 & 3/6 & 1 \end{bmatrix}$$

شکل ۸-۳: ماتریس تشابه برای داده‌های جدول ۸-۴

جدول ۸-۵ چند نمونه از مقیاس‌های دیگر را همراه با ایده‌ی اصلی آن نشان می‌دهد.

جدول ۸-۵: چگونگی محاسبه‌ی چند نمونه از معیارهای تشابه برای داده‌های دودویی

	Coefficient	Rationale
1	$\frac{a+d}{a+b+c+d}$	تاكيد بر تعداد حالت‌های يكسان صفات خاصه
2	$\frac{2(a+d)}{2(a+d)+(b+c)}$	تاكيد مضاعف بر روی تعداد حالت‌های يكسان صفات خاصه
3	$\frac{a+d}{(a+d)+2(b+c)}$	تاكيد بر تعداد حالت‌های غيرهمسان
4	$\frac{a}{a+b+c+d}$	لحوظ نکردن تعداد حالت‌های صفر-صفر در صورت کسر
5	$\frac{a}{a+b+c}$	غيرمرتبط تلقی شدن حالت صفر-صفر
6	$\frac{2a}{2a+b+c}$	لحوظ نکردن حالت‌های صفر-صفر و تاكيد مضاعف حالت‌های يك-يک
7	$\frac{a}{a+2(b+c)}$	غيرمرتبط تلقی شدن حالت صفر-صفر و تاكيد بر روی حالت‌های غيرهمسان
8	$\frac{a}{b+c}$	محاسبه‌ی نرخ حالت‌های يك-يک به تعداد حالت‌های غيرهمسان

همانطور که از محتوای جدول ۸-۵ روشن است، هر یک از مقیاس‌ها جهت کاربردی خاص مناسب به نظر می‌رسند. ماهیت داده‌ها در تعیین نوع مقیاس نقش مهمی را بازی می‌کند. نکته‌ی دیگر اینکه مقیاس‌ها تشابه را ارزیابی می‌کنند. این بدین معنی است که مقدار بزرگتر نشان دهنده‌ی تشابه بیشتر خواهد بود. نکته‌ی دیگر وجود روش‌هایی است که با داشتن یکی از دو مقیاس تشابه یا فاصله قابلیت با کمک آن راهکار، دیگری را محاسبه کنید.

برای مثال فرمول زیر با فرض وجود مقدار فاصله به نحوی تشابه بین دو نمونه را محاسبه می‌کند.

$$Sim(O_i, O_j) = \frac{1}{1 + Distance(O_i, O_j)}$$

در این فرمول با وجود مقداری برای عدم تشابه برای دو نمونه، می‌توانیم مقداری برای ارزیابی تشابه آنها بدست آوریم. اگر فرض کنیم که فاصله‌ی میان ۲ نمونه عددی بین صفر تا یک باشد، این فرمول این مقادیر را به اعدادی بین ۰/۵ تا یک نگاشت می‌کند. با فرض اینکه حداکثر تشابه مقداری برابر با یک باشد و تشابه با عددی غیرمنفی نشان داده شود، می‌توانیم با کمک فرمول زیر نیز تبدیل تشابه و فاصله را داشته باشیم.

$$Distance(O_i, O_j) = \sqrt{2 \times (1 - Sim(O_i, O_j))}$$

راه حل ساده‌ی دیگر اینکه چنانچه تشابه و عدم تشابه در محدوده‌ی صفر تا یک قرار داشته باشند، از فرمول بسیار ساده‌ی زیر می‌توان آنها را به یکدیگر تبدیل نمود.

$$Sim(O_i, O_j) = 1 - Distance(O_i, O_j)$$

۳-۳-۸) معیارهای تشابه در داده‌های کیفی

داده‌های کیفی که گاه به آنها داده‌های طبقه‌بندی شده نیز اطلاق می‌شود، می‌توانند به دو صورت اسمی و ترتیبی دسته‌بندی شوند. ویژگی‌هایی مانند جنسیت (با مقادیر مرد و زن)، محل تولد و رنگ نوعی از داده‌های کیفی هستند. از این میان، داده‌های طبقه‌بندی شده‌ی ترتیبی به داده‌هایی گفته می‌شود که می‌توان بین آنها یک نظم و ترتیب مفهومی یافت. برای مثال مدارک تحصیلی را در نظر بگیرید. ممکن است مجموعه‌ی {دیپلم، فوق دیپلم، لیسانس، فوق لیسانس و دکتری} دامنه‌ی این ویژگی را تشکیل دهند. این صفت‌خاصه از نوع داده‌های طبقه‌بندی شده است، اما یک ترتیب از نظر مفهوم در میان آنها می‌توان یافت. اما برای سه مقدار آبی، سفید و قرمز که دامنه‌ی صفت‌خاصه‌ی رنگ را تشکیل می‌دهند، هیچگونه ترتیبی نهفته نیست و این ویژگی از نوع داده‌های طبقه‌بندی شده‌ی اسمی است. در این نوع از داده‌ها فقط عمل مساوی و نامساوی معنی دارد، در صورتی که عملیات کوچکتر و بزرگتر و یا بهتر بیان کنیم بعد و قبل میان داده‌های نوع ترتیبی علاوه

بر مساوی و نامساوی نیز وجود دارد. اما معیارهای تشابه میان نمونه‌هایی که از این نوع صفات خاصه استفاده می‌کنند چگونه محاسبه می‌شود؟ مرسوم‌ترین روش برای داده‌های اسمی محاسبه‌ی تعداد صفات خاصه است که مقدار آنها مساوی نیستند.

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

عبارت نامساوی در صورت کسر هنگامی که مقدار دو صفت خاصه در ۲ نمونه برابر باشند، مقدار صفر و درغیر این صورت مقدار یک را برمی‌گرداند. برای مثال چنانچه دو نمونه‌ی O_1 و O_2 دارای شش ویژگی از نوع اسمی باشند و چهار مقدار از این ویژگی‌ها در هر دو نمونه یکسان باشد، فاصله این ۲ نمونه از فرمول زیر محاسبه می‌شود:

$$\text{Distance}(O_1, O_2) = \frac{\sum_{k=1}^6 (x_{ik} \neq x_{jk})}{6} = \frac{2}{6}$$

برای معرفی معیارهای تشابه میان ویژگی‌های ترتیبی می‌توانیم از روش‌هایی که برای صفات خاصه‌ی عددی توضیح داده شد، استفاده کنیم. اما قبل از آن می‌بایست دو عملیات زیر انجام شود:

- انتساب عدد r به عنوان رتبه به جای مقدار صفت خاصه‌ی ترتیبی بصورتی که:

$$r \in \{1, 2, \dots, s\}$$

که در آن s تعداد حالت‌هایی است که ویژگی ترتیبی می‌تواند داشته باشد.

- از آنجا که هر صفت خاصه‌ی ترتیبی می‌تواند دارای تعداد حالت‌های متفاوتی برای مقداردهی باشد، اغلب مجموعه مقادیر صحیح r به محدوده‌ی صفر تا یک نگاشت می‌شود، تا هر صفت خاصه دارای وزن یکسانی با دیگر صفات خاصه داشته باشد. این کار را می‌توان با جایگزینی r با ζ و با کمک فرمول زیر انجام داد.

$$\zeta = \frac{r-1}{s-1}$$

پس از انجام عملیات فوق می‌توانیم هر یک از معیارهای تشابه برای ویژگی‌های عددی را برای صفت‌خاصه‌ی ترتیبی مورد استفاده قرار دهیم.

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 شوند. 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 الگوریتم‌هایی 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^۱ خوشبندی شناخته خواهند شد. گاهی به آنها مرکز ثقل^۲ 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 محاسبه‌ی میانگین میان اعضای خوشبندی جدیدی انتخاب می‌گردد. این فرآیند تا پوشش

¹ Centroid

² 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 خوشی محاسبه می‌شوند.

$$Mean_1 = [(4+3+3+2)/4, (2+2+1+2)/4] = (3, 1.75)$$

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

نمونه‌های $Mean_1$ و $Mean_2$ نمایندگان جدید خوشی‌ها محاسبه 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 خوشی‌های $Cluster_2$ و $Cluster_1$ اصلاح شوند.

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

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

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

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

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

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

در این مثال و در این مرحله اعضای خوشی‌ها هیچ تغییری نمی‌کنند. به عبارت دیگر هیچگونه انتساب مجدد و متفاوتی صورت نخواهد گرفت و لذا الگوریتم خاتمه می‌یابد. چنانچه نمونه‌ای از یک خوشی به خوشی دیگر منتقل می‌شد، بایست 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 دارد. 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 *PAM*^۱ با نام *k-Medoids* 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.

۳-۴-۸) الگوریتم‌های دیگر مبتنی بر افزای داده‌ها

یک الگوریتم خوشبندی مبتنی بر افزای داده‌ها نظیر *PAM* می‌تواند بر روی حجم کمی از مجموعه داده‌های ورودی مناسب باشد، اما در برخورد با داده‌های حجمی این روش معمولاً^۲ کارآ نیست. روشی موسوم به *CLARA*^۳ این چالش را با کمک شیوه‌های مبتنی بر نمونه‌گیری^۳ تا حدودی حل کرده است. ایده‌ی اصلی این روش همانند شیوه‌های نمونه‌گیری کار با بخش کوچکی از داده‌ها به جای کل داده‌هاست. نماینده‌ها و مراکز ثقل خوش‌ها نیز از میان رکوردهایی که در نمونه‌گیری قرار دارند، انتخاب می‌شوند. به جهت آنکه نمونه‌گیری از مجموعه داده‌ها جهت‌گیرانه نباشد، الگوریتم چندین بار و بر روی نمونه‌گیری‌های متفاوت اجرا و بهترین نتیجه‌ی خوشبندی بازگردانده می‌شود. پیچیدگی زمانی هر تکرار به $O(ks^2+k(n-k))$ می‌رسد. که در آن همانند قبل n و k به ترتیب تعداد کل نمونه‌ها و تعداد خوش‌ها هستند و s نماینده‌ی تعداد رکوردهایی است که با روش نمونه‌گیری انتخاب شده‌اند. کارایی این الگوریتم بطور کامل وابسته به اندازه‌ی نمونه‌گیری است.

¹ Partitioning Around Medoids

² Clustering Large Applications

³ Sampling- based Methods

در k -الگوریتم در میان کلیه‌ی نمونه‌ها به دنبال نمایندگان مناسبی می‌گشت در حالی که در *CLARA* فقط در میان برخی از نمونه‌های مجموعه داده‌ها به دنبال بهترین‌ها می‌گردد. اگر نمونه‌ی مناسبی در میان رکوردهای انتخاب شده توسط روش‌های نمونه‌گیری نباشد، بهترین خروجی را برای این الگوریتم خوشبندی خواهیم داشت. بدون شک با در اختیار داشتن کل داده‌ها نتیجه‌ی بهتری خواهد داشت، اما این عمل با پیچیدگی زمانی بالاتری نیز انجام خواهد شد.

به منظور بهبود کیفیت و قابلیت مقیاس‌پذیری روش *CLARA* الگوریتم دیگری به نام *CLARANS*^۱ پیشنهاد شد که ترکیب تکنیک نمونه‌گیری و الگوریتم *PAM* بود. این الگوریتم برخلاف *CLARA* خود را محدود به نمونه‌های ثابتی که با یک بار نمونه‌گیری بدهست می‌آید، نمی‌کند.

از نظر مفهومی فرایند خوشبندی را می‌توان همانند جستجو در گراف فرض کرد، که هر گره در این گراف یک راه حل بالقوه‌ی درست تلقی می‌شود. دو گره در این گراف موقعی می‌توانند همسایه باشند که مجموعه داده‌های آنها فقط در موقعیت یک نمونه متفاوت باشد. در هر گره همچنین مقداری نگهداری می‌شود که نشان‌دهنده‌ی مجموع فواصل بین نمونه‌های هر خوشه با نماینده‌ی همان خوشه است. در هر مرحله از الگوریتم *PAM* کلیه‌ی همسایه‌های گرهی جاری جهت یافتن حداقل خطای جستجو می‌شوند. گره یا حالت جاری با گره و حالتی جایگزین می‌شود که بیشترین کاهش را در مقدار این خطای دارد. چون در الگوریتم *CLARA* با نمونه‌های کمتری از داده‌ها که با روش‌های نمونه‌گیری انتخاب شده‌اند سروکار داریم، گرهی جاری دارای همسایه‌های کمتری است و می‌توان گفت در این الگوریتم فضای جستجوی کوچکتری نسبت به الگوریتم *PAM* خواهیم داشت.

محدود بودن فضای جستجوی الگوریتم *CLARA* می‌تواند از معایب این روش به شمار رود و این در حالی است که الگوریتم *CLARANS* یک نمونه‌ی اتفاقی از گره‌های همسایه را بصورت پویا در هر مرحله انتخاب می‌کند. با کمک پارامتری که توسط کاربر مشخص می‌شود، تعداد این گره‌های همسایه می‌تواند معین گردد. بدین ترتیب جستجو در

^۱ Clustering Large Applications based upon Randomized Search

این الگوریتم به یک ناحیه‌ی محلی محدود نمی‌شود. با یافتن حالت و گرهی همسایه‌ای که خطای کمتری دارد الگوریتم به طرف این حالت و گره حرکت می‌کند و فرایند ادامه می‌یابد، در غیراینصورت خوشبندی به یک بهینه‌ی محلی دست یافته است. این بدین معنی است که حالت (گرهی) انتخابی قبلی ممکن است مناسب نباشد. لذا الگوریتم *CLARANS* بصورت اتفاقی گرهی دیگری را انتخاب می‌کند. تعداد دفعاتی که الگوریتم می‌تواند شروع مجدد با حالت(گرهی) جدید را تجربه کند، ممکن است توسط *CLARANS* کاربر مشخص شود. نتایج تجربی بدست آمده نشان می‌دهد که الگوریتم *CLARANS* کارایی بهتری نسبت به الگوریتم‌های *PAM* و *CLARA* از خود نشان می‌دهد. این روش جهت کشف داده‌های خارج از محدوده نیز می‌تواند مفید باشد. به هر حال پیچیدگی زمانی این الگوریتم $O(n^2)$ است که در آن n تعداد نمونه‌هاست. فراموش نکنید که کیفیت خوشبندی در این روش به شیوه‌های نمونه‌گیری واپسی است و استفاده از ساختمان داده‌هایی نظیر *R*-tree* برای ذخیره نمونه‌هایی که بر روی دیسک قرار دارند، به بهبود کارایی در این الگوریتم کمک می‌کند.

۵-۸) تکنیک‌های خوشبندی سلسله‌مراتبی

برخلاف تکنیک‌های خوشبندی مبتنی بر افزار که تعداد خوشه‌ها به عنوان پارامتر ورودی توسط کاربر مشخص می‌شده، 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^۱ و یا بر عکس از بالا به پایین^۲ انجام شود، عملیات اصلی این الگوریتم‌ها را می‌توان در دو دسته‌ی ادغام^۳ و تقسیم^۴ 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.

¹ Bottom-up

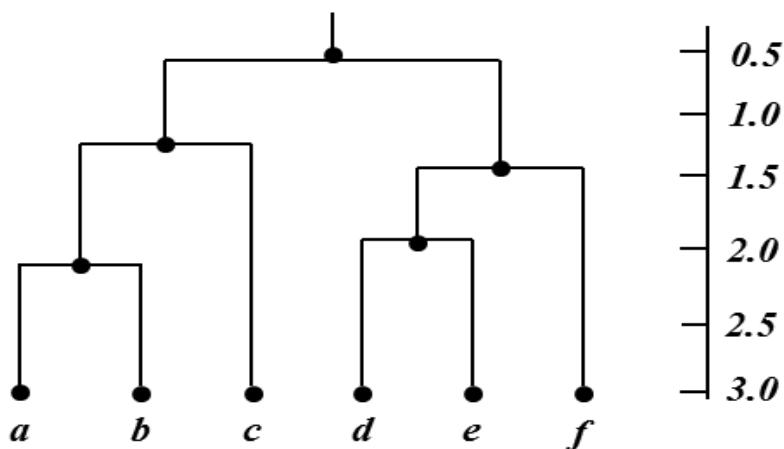
² Top-down

³ Agglomerative

⁴ 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^۱ فرایند خوشبندی سلسله‌مراتبی توسط یک نمودار با نام دنдрوگرام^۱ 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

^۱ 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^۱ 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.

^۱ 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 ۶-۸ مشخص شده است، دنبال کنیم.

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

Attribute	O ₁	O ₂	O ₃	O ₄	O ₅
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 ۶-۸).

	O ₁	O ₂	O ₃	O ₄	O ₅
O ₁	0				
O ₂	4.0	0			
O ₃	5.1	1.4	0		
O ₄	1.0	4.1	5.0	0	
O ₅	3.1	1.4	2.0	3.0	0

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

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

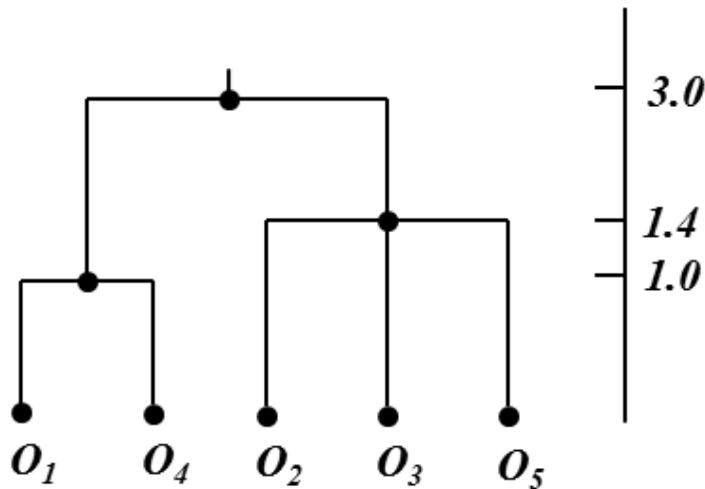
شکل ۸-۹: ماتریس تشابه پس از اجرای مرحله‌ی سوم

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^۱ و دورترین همسایه^۱ نیز شناخته می‌شود.

^۱ 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.

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

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

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

^۱ 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^۱ *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.

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

^۱ 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^۱ *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^۲ *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.

¹ Weighted Pair Group Method using Arithmetic Averages

² Unweighted Pair Group Method using the Centroids

روشی مبتنی بر میانه با نام مختصر^۱ *WPGMC*^۲ وجود دارد که همانند روش بالا عمل می‌کند، به جز اینکه با فرض ورود وزن خوشها ممکن است عملکرد بهتری نسبت به روش بالا داشته باشد. در این روش فاصله‌ی میان خوشها ادغام شده و دیگر خوشها 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 خوشها اشاره می‌شود که در سلسله مراتب خوشها هم‌سطح هستند.

BIRCH (۸-۵-۲)

معمولًاً ورودی الگوریتم‌های داده‌کاوی را حجم انبوهی از داده‌ها تشکیل می‌دهد. از این جهت الگوریتم‌های مقیاس‌پذیری که در مواجهه با این نمونه‌های حجیم می‌توانند عملکرد مناسبی از خود نشان دهند، مورد استقبال کاربران قرار می‌گیرند. الگوریتم خوشبندی *BIRCH*^۳ از این دسته از الگوریتم‌ها محسوب می‌شود. این روش با کمک تکنیک‌های خوشبندی سلسله‌مراتبی و تکنیک‌های دیگر برای کار بر روی داده‌های عددی با حجم بالا طراحی شده است. این الگوریتم همچنین از اولین الگوریتم‌های خوشبندی است که برای تشخیص داده‌های نویز و مزاحم می‌توان از آن استفاده کرد.

در این الگوریتم مفهومی به نام ویژگی خوشه^۳ یا به اختصار *CF* معرفی می‌شود که طریقه‌ای جهت نمایش خلاصه شده خوشها به شمار می‌رود. استفاده از این مفهوم *CF* باعث می‌شود تا الگوریتم از سرعت مناسب و قابلیت مقیاس‌پذیری خوبی در مواجهه با داده‌های بسیار حجیم از خود نشان دهد.

¹ Weighted Pair Group Method using the Centroids(Median)

² Balanced Iterative Reducing and Clustering using Hierarchies

³ Clustering Feature

یک ویژگی خوش‌با سه‌گانه‌ی زیر تعریف می‌شود:

$$CF = [n, LS, SS]$$

که در آن n تعداد نمونه‌های موجود در خوش‌با LS مجموع خطی این نمونه‌ها و SS مجموع مربع نمونه‌های موجود در خوش‌با است. برای مثال فرض کنید در خوش‌با C سه نمونه‌ی $(1, 2, 3)$ ، $(4, 5, 6)$ وجود دارند. محاسبه‌ی CF برای این خوش‌با بصورت زیر انجام می‌شود:

$$CF(C) = [3, (1+2+5, 3+6+4), (1^2+2^2+5^2, 3^2+6^2+4^2)] = [3, (8, 13), (30, 61)]$$

در واقع به جای نگهداری مشخصات نمونه‌های موجود در یک خوش‌با می‌توان CF مربوط به خوش‌با استفاده کرد. نکته‌ی مهم اینکه CF ‌ها برای انجام خوش‌بندی داده‌ها در الگوریتم $BIRCH$ اطلاعات کافی جهت محاسبات را دارا هستند. بدین طریق با نمایش ساده و خلاصه شده‌ی خوش‌باها می‌توانیم از حافظه به صورت مؤثر و کارا بهره ببریم. از طرفی دیگر با ادغام دو خوش‌با فقط کافی است که سه مؤلفه‌ی موجود در ویژگی خوش‌با را با یکدیگر جمع کنیم. به مثال زیر توجه نمایید:

$$CF(C_1) = [3, (8, 13), (30, 61)] \quad , \quad CF(C_2) = [4, (4, 12), (4, 46)]$$

$$CF(C_1 \cup C_2) = [3+4, (8+4, 13+12), (30+4, 61+46)] = [7, (12, 25), (34, 107)]$$

مفهوم دیگری که در الگوریتم $BIRCH$ از آن استفاده می‌شود، درخت متوازنی^۱ است که CF ‌های خوش‌باها را جهت اجرای الگوریتم ذخیره می‌کند. هر گره‌ای از این درخت که دارای گرهی فرزند است، در خود مجموع CF ‌های فرزندانش را نگهداری می‌کند. بدین ترتیب اطلاعات خلاصه شده‌ای از فرزندان خود را خواهد داشت.

درخت دارای دو پارامتر به نام‌های فاکتور شاخه‌بندی^۲ B و حد آستانه‌ی^۳ T است. حداکثر تعداد فرزندان گره‌های غیرپایانی با فاکتور شاخه‌بندی B مشخص می‌شوند و حد آستانه‌ی T حداکثر قطر(فاصله) خوش‌باها ذخیره شده در گره‌های پایانی(برگ‌ها) را معین می‌سازد. قطر به حداکثر فاصله‌ای که میان دو نمونه از خوش‌باها برگ قرار دارد، اطلاق می‌شود.

¹ Balanced Tree

² Branching Factor

³ Thersholt

پس از درج و یا حذف یک CF در درخت و یا در صورت افزایش مقادیر B و T از مقدار تعیین شده توسط کاربر، گره‌های درخت متوازن تقسیم و یا ادغام می‌شوند. واضح است که مقادیر B و T در اندازه‌ی درخت تولید شده‌ی نهایی نقش بسزایی ایفا می‌کنند. برای مثال اگر حافظه‌ی کافی جهت ذخیره‌ی درخت CF وجود نداشته باشد، می‌توانیم با مقدار جدید T (بزرگتر از قبل) درخت را بازسازی و درخت کوچکتری تولید کنیم. بنابراین می‌توان ادعا کرد، الگوریتم با منابع در دسترس بهترین خوشبندی را انجام می‌دهد. به منظور بازسازی درخت نیز لازم نیست تمام نمونه‌ها بازخوانی شوند، فقط کافی است با کمک گره‌های پایانی درخت قدیمی، درخت جدید را تولید کرد.

الگوریتم $BIRCH$ یک الگوریتم خوشبندی افزایشی^۱ به شمار می‌رود. این بدین معنی است که با ورود هر نمونه، درخت به صورت پویا ساخته می‌شود و لازم نیست الگوریتم از ابتدا بر روی کلیه‌ی داده‌ها اجرا شود. نمونه‌ی جدید به نزدیکترین مدخل برگ اضافه خواهد شد و چنانچه قطر(فاصله) خوشبندی ذخیره شده در برگ مزبور از حد آستانه‌ی T تجاوز کند، برگ و شاید دیگر گره‌ها شکسته شوند. پس از درج نمونه‌ی جدید اطلاعات آن به گره‌های بالاتر(تا ریشه‌ی درخت) منتقل می‌شود.

پس از ساخت درخت، الگوریتم $BIRCH$ با به خدمت گرفتن یک الگوریتم خوشبندی، کار خود را ادامه می‌دهد. با کمک از گره‌های پایانی (برگ‌ها) خوشبندی‌های پراکنده به عنوان داده‌های خارج از محدوده حذف و همچنین خوشبندی‌های متراکم در خوشبندی بزرگتر گروه‌بندی می‌شوند.

پیچیدگی زمانی الگوریتم برابر با $O(n)$ است که در آن n تعداد کل نمونه‌هایی است که

خوشبندی می‌شوند. روش‌ها و هیوریستیک‌هایی ارائه شده‌اند که کیفیت خوشبندی تولید شده را بهبود می‌بخشند، ولی در مقابل پیمایش مجدد داده‌ها را به دنبال دارند.

به هر حال از آنجا که هر گره ظرفیت محدودی برای نگهداری CF ‌ها دارد، خوشبندی تولید شده توسط الگوریتم می‌تواند دلخواه کاربر نباشد. به علاوه چون الگوریتم از مفهومی مثل قطر جهت کنترل محدوده خوشبندی استفاده می‌کند، در مواجهه با خوشبندی که کروی شکل نیستند، اجرای خوبی را به دنبال نخواهد داشت.

¹ Incremental Clustering

اما همانطور که قبل از این به آن اشاره شد، قابلیت مقایس‌پذیری مناسب و سرعت اجرای بالا با وجود هر مقدار حافظه، از مزایای اصلی این الگوریتم به شمار می‌روند.

۳-۵-۸) الگوریتم CURE

یکی از تکنیک‌های خوشبندی سلسله‌مراتبی که توانایی تشخیص خوش‌هایی متفاوت در اندازه و غیرکروی شکل را در پایگاه‌داده‌های بزرگ دارد، به اختصار^۱ CURE نامیده می‌شود. در این الگوریتم هر خوش با تعداد معینی از نمونه‌ها که به طرز مناسبی در فضای خوش پراکنده شده‌اند، نمایش داده می‌شود. استفاده از روشهای نمونه‌گیری تصادفی و افزار داده‌ها در این الگوریتم باعث شده است تا عملکرد مناسبی در برابر پایگاه داده‌های بزرگ داشته باشد. این الگوریتم شامل ۶ مرحله‌ی اصلی است که به اختصار هر یک از مراحل در ادامه توضیح داده شده است.

- در مرحله‌ی اول با کمک روشهای نمونه‌گیری برخی از داده‌ها انتخاب می‌شوند. همان طور که می‌دانیم، انتخاب تعداد مناسب نمونه‌ها در نتایج بدست آمده در مراحل بعدی موثر است.
- در این مرحله جهت افزایش سرعت الگوریتم در مقابله با حجم ورودی بالا، از یک روش ساده برای افزار داده‌ها استفاده می‌شود. چنانچه حجم نمونه‌گیری در مرحله اول Δ و تعداد افزارها برابر با p باشد، هر دسته دارای $\frac{s}{p}$ نمونه خواهد بود.
- در مرحله‌ی سوم هر یک از این p دسته خوشبندی می‌شوند، به نحوی که تعداد خوش‌های هر دسته به $\frac{s}{pq} > I$ است، کاهش پیدا کند.
- از آنجا که همیشه نمونه‌ها می‌توانند دارای داده‌های خارج از محدوده باشند، مرحله‌ی چهارم این الگوریتم به حذف این نمونه‌ها تخصیص داده شده است.

¹ Clustering Using Representatives

- در مرحله‌ی پنجم با کمک از یک الگوریتم خوشبندی، خوشبندی‌های تولید شده در هر دسته با یکدیگر ترکیب و خوشبندی‌های نهایی تولید می‌شوند.
 - به دلیل آنکه اجرای الگوریتم بر روی نمونه‌های محدودی انجام شده است(به دلیل نمونه‌گیری اول)، باید برای نمونه‌های باقیمانده برچسب خوشبندی در نظر گرفته شود. بنابراین در این مرحله، نمونه‌های باقیمانده برچسبی معادل خوشبندی نزدیکتر را به خود خواهند گرفت.
- در بدترین حالت، پیچیدگی زمانی این الگوریتم برابر با $O(n^2 \times \text{Log}(n))$ است. در صورتی که با ابعاد پایین داده‌ها این روش اجرا شود، این مقدار به $O(n^2)$ نیز کاهش می‌یابد.

۴-۵) الگوریتم *ROCK*

در بیشتر الگوریتم‌های خوشبندی تشابه میان نمونه‌ها معیار ارزیابی است و در هر مرحله نمونه‌های مشابه در یک خوشبندی قرار می‌گیرند. اما با این شرط الگوریتم مستعد خطاست. برای مثال دو خوشبندی را در نظر بگیرید که فقط چند نمونه از هر یک(شاید نمونه‌های خارج از محدوده) به هم نزدیک و مشابه باشند. با معیار مزبور دو خوشبندی مستعد خطاست. شوند در حالیکه این عمل مناسب نیست. الگوریتم *ROCK*¹ که برای خوشبندی بر روی داده‌هایی با صفات خاصه‌ی طبقه‌بندی شده و دودویی طراحی شده است، با کمک محاسبه‌ی تعداد همسایه‌های مشترک میان نمونه‌ها راهکار مناسبی ارائه می‌کند. بطور معمول الگوریتم‌های خوشبندی از معیارهایی که در بخش‌های اولیه این فصل ارائه شدند، جهت ارزیابی تشابه یا فاصله میان نمونه‌ها استفاده می‌کنند. نتایج تجربی نشان می‌دهند این معیارها نمی‌توانند با نوع ویژگی طبقه‌بندی شده خوشبندی را در خروجی تولید کنند. در فصل بعد در مورد مفهوم همسایگی که در تکنیک‌های خوشبندی مبتنی بر تراکم از آن استفاده می‌کنیم، توضیح داده شده است، اما بطور رسمی و خلاصه دو نمونه‌ی O_i و O_j را هنگامی همسایه می‌نامیم که داشته باشیم:

$$\text{Similarity}(O_i, O_j) \geq T$$

¹ Robust Clustering using Links

که در آن *Similarity* تابع معین و T حد آستانه‌ای است که مقدار آنها توسط کاربر مشخص می‌شود.

تعداد پیوندهای میان O_i و O_j به عنوان تعداد همسایه‌های مشترک میان این دو نمونه تلقی می‌شود. اگر این تعداد زیاد باشد، امکان بکی بودن خوش‌های این دو نمونه بیشتر است. با در نظر گرفتن تعداد همسایه‌های مشترک می‌توان گفت الگوریتم *ROCK* در مقابل داده‌های نویز و خارج از محدوده مقاوم‌تر از الگوریتم‌های خوش‌بندی است که فقط بر روی تشابه میان نمونه‌ها متتمرکز می‌شوند.

بر اساس توضیح مختصر بالا در می‌یابیم که الگوریتم *ROCK* با کمک از تابع ارزیابی تشابه، حد آستانه‌ی آن و ماتریس تشابه در مرحله‌ی اول به ساخت یک گراف پراکنده^۱ می‌پردازد. در گراف مفهوم همسایه‌های مشترک نیز لحاظ شده است. مرحله‌ی بعدی اجرای یک الگوریتم خوش‌بندی سلسله‌مراتبی بر روی این گراف است که استفاده از یک معیار ارزیابی مناسب جهت تولید خوش‌های خروجی می‌تواند آنرا بهبود بخشد. برای مجموعه داده‌هایی با حجم بالا می‌توان از روش‌های نمونه‌گیری نیز استفاده نمود.

در بدترین حالت پیچیدگی محاسباتی این الگوریتم برابر با $O(n^2 + nma + n^2 \log(n))$ است، که در آن n تعداد کل نمونه‌ها، m حداکثر تعداد همسایه‌ها و a میانگین تعداد همسایه‌ها را نشان می‌دهد.

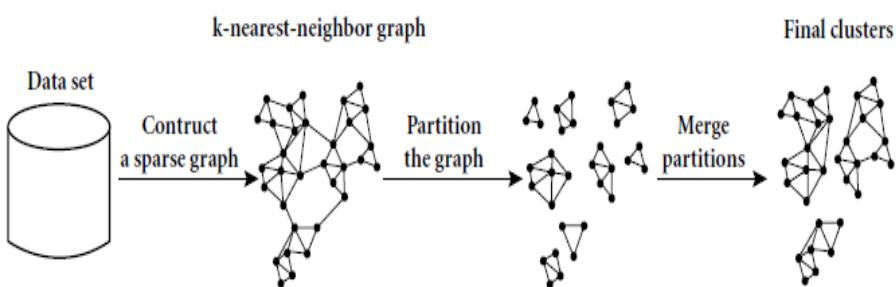
۵-۸-۵) الگوریتم *Chameleon*

از یک زاویه تکنیک‌های خوش‌بندی می‌توانند به عنوان مسئله‌ای تحت عنوان افزار گراف‌ها^۲ مطرح شوند. بسیاری از الگوریتم‌های خوش‌بندی در مراحل ابتدایی خود مجموعه داده‌ها را با یک گراف نمایش می‌دهند و ادامه‌ی کار خود را بر روی این گراف انجام می‌دهند. بطور معمول گره‌های گراف نمونه‌ها را نشان می‌دهند و یال‌ها که دو نمونه را به یکدیگر متصل می‌کند، وزن دار هستند. مقدار این وزن به نوع روش یا تکنیکی که استفاده می‌شود، بستگی خواهد داشت. برای مثال تکنیک‌های اولیه‌ای که در این

¹ Sparse Graph

² Graph Partitioning

فصل به عنوان تکنیک‌های خوشبندی سلسله‌مراتبی بررسی شدند، همانند روش انتخاب دو نمونه با بیشترین تشابه، بر روی گراف کاملی کار می‌کردند که یال‌های گراف معرف شباهت میان دو نمونه بود. به همین دلیل در برخی از منابع داده‌کاوی از این نوع روش‌ها به عنوان تکنیک‌های مبتنی بر گراف^۱ نیز یاد می‌کنند. الگوریتم Chameleon یکی از تکنیک‌های خوشبندی سلسله‌مراتبی است که مفهوم گراف در آن نقش اساسی را بازی می‌کند. مراحل انجام این الگوریتم در شکل ۸-۱۴ به صورت تصویری نمایش داده شده است.



شکل ۸-۱۴: مراحل اجرای الگوریتم Chameleon

پس از تشکیل ماتریس تشابه، الگوریتم برای هر نمونه k همسایه با کوچکترین مقادیر فاصله (بیشترین تشابه) را انتخاب و کار خود را شروع می‌کند. حاصل این مرحله گرافی است که گره‌های آن نماینده‌ی نمونه‌ها و k اتصال هر گره نشان دهنده‌ی k نمونه‌ای است که در میان داده‌ها نزدیکترین نمونه به گرهی مورد نظر هستند. به عبارتی دیگر هر گره (نمونه) حداقل به k گرهی (نمونه‌ی) دیگر متصل می‌شود.

توجه داشته باشید که مفهوم همسایگی در این گراف کاملاً یک مفهوم پویاست و این بدین معنی است که شعاع همسایگی با توجه به تراکم داده‌ها و نمونه‌ها در آن منطقه تعریف می‌شود. در مناطق متراکم گره‌های همسایه نزدیکتر هستند و در مناطق غیرمتراکم این همسایگی با فاصله‌ی بیشتری مشخص می‌شود. در مقایسه با روش‌هایی که در آن یک شعاع همسایگی ثابت تعریف می‌شود، این نکته می‌تواند از مزایای این الگوریتم به شمار رود. تراکم بیشتر مناطق، وزن بیشتر یال‌های گراف را باعث می‌شود، چرا که وزن

^۱ Graph-based Techniques

یال‌ها نشان دهندهٔ تشابه میان نمونه‌هایی است که توسط این یال‌ها به یکدیگر متصل شده‌اند.

در مرحله‌ی دوم گراف تولید شده در مرحله‌ی قبل به بخش‌های کوچکی افزار می‌شود. تقسیم گراف به نحوی است که وزن یال‌های حذف شده در این افزار به حداقل می‌رسد. این عمل تا رسیدن به یک معیار مشخص تکرار می‌شود. الگوریتم‌های متعددی جهت افزار گراف وجود دارند که از کارایی خوبی برخوردارند و می‌توان از آنها استفاده نمود. در ادامه مجموع وزن یال‌هایی که با حذف آنها خوش‌ی C به دو خوش‌ی C_i و C_j شکسته می‌شود، به صورت $EC(C_i, C_j)$ نمایش می‌دهیم.

در مرحله‌ی پایانی الگوریتم *Chameleon* با بهره از تکنیک خوش‌بندی سلسله‌مراتبی و ادغام خوش‌های تولید شده در مرحله‌ی قبل، کار خود را به اتمام می‌رساند. این الگوریتم جهت ادغام دو خوش از ترکیب دو معیار تشابه استفاده می‌کند. شباهت بین یک جفت از خوش‌های C_i و C_j بر طبق به هم پیوستگی ارتباط^۱ آنها ($RI(C_i, C_j)$ و نزدیکی نسبی^۲ آنها ($RC(C_i, C_j)$ ارزیابی می‌شود. با در نظر گرفتن این دو معیار ضعف الگوریتم‌هایی مانند *CURE* و *ROCK* که تأکید آنها فقط بر روی یکی از این پارامترهاست، مرتفع خواهد شد.

فرمول‌های زیر طریقه‌ی محاسبه‌ی هر یک از این دو معیار را نشان می‌دهند.

$$RI(C_i, C_j) = \frac{|EC(C_i, C_j)|}{\frac{1}{2}[|EC(C_i)| + |EC(C_j)|]}$$

$$RC(C_i, C_j) = \frac{\overline{EC}(C_i, C_j)}{\frac{|C_i|}{|C_i| + |C_j|} \overline{EC}(C_i) + \frac{|C_j|}{|C_i| + |C_j|} \overline{EC}(C_j)}$$

جدول ۸-۷ عبارت‌های به کار برده شده در فرمول‌ها را توضیح می‌دهد.

^۱ Relative Interconnectivity

^۲ Relative Closeness

جدول ۷-۸: معانی عبارت‌های به کار برد شده در محاسبه‌ی توابع RC و RI

Component	Meaning
$EC(C_i)$	مجموع وزن یال‌هایی که خوشبندی C_i را به دو قسمت تقریباً هم اندازه می‌شکند. یال‌های انتخابی دارای مجموع وزنی حداقل هستند.
$EC(C_i, C_j)$	مجموع وزن یال‌هایی که گره‌های خوشبندی C_i را به گره‌های خوشبندی C_j متصل می‌کند.
$\overline{EC}(C_i)$	میانگین وزن یال‌هایی در خوشبندی C_i که با حذف آنها خوشبندی مزبور به دو قسمت تقریباً هم اندازه شکسته می‌شود. یال‌های انتخابی دارای میانگین وزنی حداقل هستند.
$\overline{EC}(C_i, C_j)$	میانگین وزن یال‌هایی که گره‌های خوشبندی C_i را به گره‌های خوشبندی C_j متصل می‌کند.

با محاسبه‌ی دو تابع RC و RI برای دو خوشبندی C_i و C_j می‌توان معین کرد که این خوشبندی‌ها مستعد ادغام هستند یا خیر. روش اول تعریف دو حد آستانه‌ی T_{RC} و T_{RI} برای مقدار دو تابع فوق است، به ترتیبی که هر گاه مقادیر محاسبه شده بزرگتر از حد آستانه باشند، این دو خوشبندی ادغام می‌شوند. یعنی با درست بودن شرط زیر:

$$RI(C_i, C_j) > T_{RI} \quad \text{and} \quad RC(C_i, C_j) > T_{RC}$$

راه حل دیگر یافتن تابعی است که در آن دو تابع RC و RI نقش مستقیمی را در آن بازی می‌کنند. برای مثال در بسیاری از مواقع ارزیابی تشابه میان دو خوشبندی با حاصل ضرب زیر محاسبه می‌شود:

$$RC(C_i, C_j) \times RI(C_i, C_j)^{\alpha}$$

در نظر گرفتن مقادیر مختلفی برای α تأکید غیریکسان دوتابع فوق را باعث می‌شود. مقدار یک برای α وزن یکسانی برای اندازه‌گیری هر دوی آنها ارائه می‌دهد. از مزایای الگوریتم *Chameleon* می‌توان به توانایی در کشف خوشه‌هایی با اشکال دلخواه و چگالی گوناگون اشاره نمود. پیچیدگی زمانی آن برای n نمونه برابر با $O(n^2)$ است.

۶-۵) الگوریتم *DIANA*

همانطور که قبلاً نیز به آن اشاره کردیم، تکنیک‌های خوشه‌بندی سلسله مراتبی در دو دسته‌ی پایین به بالا و بالا به پایین گروه‌بندی شده‌اند. الگوریتم‌هایی که تاکنون به بررسی آنها پرداخته‌ایم، در دسته‌ی اول گنجانده می‌شوند. این بدین معنی است که با قرار دادن هر نمونه در خوشه‌ای مجزا شروع و پس از معرفی معیاری جهت ارزیابی تشابه، آنها را ادغام می‌کنیم تا جایی که کلیه‌ی نمونه‌ها در یک خوشه قرار می‌گیرند. علیرغم محبوبیت و عمومیت داشتن این دسته، برخی از الگوریتم‌ها نظیر^۱ *DIANA* و *DISMEA* عملکردی بر عکس روش‌های فوق را دنبال می‌کنند. به این صورت که در ابتدا کلیه‌ی نمونه‌ها در یک خوشه قرار می‌گیرند و سپس در هر مرحله خوشه‌ها به خوشه‌های کوچکتر تقسیم می‌شوند، تا جایی که هر نمونه در یک خوشه قرار می‌گیرد. به منظور آشنایی بیشتر در ادامه الگوریتم *DIANA* به اختصار توضیح داده می‌شود. در هر مرحله از الگوریتم *DIANA* بزرگترین خوشه تقسیم می‌شود، تا در مرحله $(n-1)$ ام هر نمونه در یک خوشه‌ی مجزا قرار می‌گیرد. یکی از مقیاس‌هایی که برای سنجش بزرگ بودن خوشه استفاده می‌شود، قطر هر خوشه است که با فرمول زیر محاسبه می‌شود:

$$\text{Diameter}(C) = \max_{x,y \in C} \{\text{Distance}(x, y)\}$$

این مقدار همچنین می‌تواند بر روی محور عمودی در کنار نمودار دندروگرام که مقیاس تشابه (فاصله) را نشان می‌دهد، نوشته شود.

¹ Divisive Analysis

فرض کنید خوشبندی C خوشبندی مورد نظری است که باید به دو خوشبندی دیگر یعنی A و B تقسیم شود. تعداد نمونه‌های موجود در خوشبندی C بیشتر از دو نمونه است و پس از عمل تقسیم خوشبندی A و B دارای نمونه‌ی مشترکی نیستند. الگوریتم *DIANA* در ابتدا کلیه‌ی نمونه‌های خوشبندی C را در خوشبندی A قرار می‌دهد و خوشبندی B در ابتدا خالی است.

در مرحله‌ی اول یک نمونه از خوشبندی A در صورتی به خوشبندی B منتقل می‌شود که تابع زیر را به حداقل برساند.

$$D(O_i, A - \{O_i\}) = \frac{1}{|A|-1} \times \sum_{O_j \in A, O_j \neq O_i} d(O_i, O_j)$$

در فرمول O_i نمونه‌ی مورد نظر برای انتقال و $|A|$ /اندازه‌ی خوشبندی A و $d()$ تابعی جهت ارزیابی فاصله‌ی میان نمونه‌ها است. پس از انتقال O_i به خوشبندی B خوشبندی B بروزرسانی می‌شوند. یعنی:

$$A_{new} = A_{old} - \{O_i\}, \quad B_{new} = B_{old} \cup \{O_i\}$$

در مراحل بعدی الگوریتم جستجوی خود را برای انتقال دیگر نمونه‌ها از A به خوشبندی B ادامه می‌یابد و در این مراحل به حداقل رساندن تابع زیر برای انتقال i بررسی می‌شود.

$$D(O_i, A - \{O_i\}) - D(O_i, B) = \frac{1}{|A|-1} \times \sum_{O_j \in A, O_j \neq O_i} d(O_i, O_j) - \frac{1}{|B|} \times \sum_{O_k \in B} d(O_i, O_k)$$

چنانچه نمونه‌ی O_i تابع فوق را به حداقل برساند و این مقدار عددی مثبت باشد، از خوشبندی A به خوشبندی B منتقل می‌شود. در غیراینصورت با منفی یا صفر بودن این مقدار فرایند تقسیم متوقف و خوشبندی A و B جایگزین خوشبندی C خواهند شد. توابع متعدد دیگری نیز می‌توانند در فرایند تقسیم به دو خوشبندی استفاده شوند.

از آنجا که الگوریتم *DIANA* از بزرگترین فاصله میان نمونه‌های یک خوشبندی (قطر خوشبندی) به عنوان معیار تشابه استفاده می‌کند، می‌توان حدس زد که این روش نسبت به نمونه‌های خارج از محدوده عکس العمل مناسبی از خود نشان نمی‌دهد.

خلاصه فصل

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 اعضای دیگر خوشه‌ها برخوردارند. خوشه‌بندی می‌تواند در کاربردهای زیادی مورد استفاده قرار گیرد و شاخه‌هایی همچون هوش مصنوعی، آمار، زیست‌شناسی، پزشکی، یادگیری ماشین، تشخیص الگو... را می‌توان نام برد.

یک الگوریتم خوشه‌بندی می‌تواند دارای مشخصات مطلوبی نظیر قابلیت مقیاس‌پذیری، توانایی مواجهه با انواع داده‌ها، استخراج خوشه‌هایی 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 خوشه‌هایی مبتنی بر محدودیت و همچنین قابل فهم بودن نتایج نهایی الگوریتم باشد.

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

هر چند بسیاری از تکنیک‌های خوشه‌بندی را نمی‌توان در یک گروه خاص قرار داد اما در این فصل ما این تکنیک‌ها را در پنج گروه خوشه‌بندی مبتنی بر افزار، خوشه‌بندی سلسه مراتبی، خوشه‌بندی مبتنی بر تراکم یا چگالی داده‌ها، خوشه‌بندی مبتنی بر شبکه‌های شطرنجی گردید و خوشه‌بندی مبتنی بر مدل قرار داده‌ایم.

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 خوشها معمولاً در این الگوریتم‌ها به عنوان ورودی الگوریتم مشخص می‌شود و هر نمونه داده فقط عضو یک خوش می‌تواند باشد.

از شناخته شده‌ترین الگوریتم‌های خوشبندی مبتنی بر افزار می‌توان به الگوریتم‌هایی 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 حالی که تکنیک‌های خوشبندی مبتنی بر افزار با یک ورودی که مشخص کننده‌ی تعداد خوشهاست شروع می‌کند و هر نمونه را با کمک معیارهای معرفی شده در الگوریتم به خوشها تخصیص می‌دهد، در روش‌ها و تکنیک‌های خوشبندی سلسله‌مراتبی ما با یکی از دو عمل ادغام و یا تقسیم روبرو هستیم. در حقیقت دو نوع خوشبندی سلسله‌مراتبی وجود دارد. الگوریتم‌های نوع اول در ابتدا هر یک از نمونه‌ها را در یک خوش قرار می‌دهند و الگوریتم در حین اجرا سعی بر ادغام نمودن خوشهاستی 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 خوش قرار داده می‌شوند و در هر مرحله، نمونه‌هایی که کمترین تشابه را دارند در خوشهاستی متفاوت قرار می‌گیرند. نوع اول از الگوریتم‌های خوشبندی سلسله‌مراتبی بصورت گستردگی تری نسبت به نوع دوم استفاده می‌شوند. در این فصل به بررسی الگوریتم‌هایی نظیر *CURE*, *BIRCH*, *DIANA* و *Chameleon*, *ROCK* پرداختیم.

مراجع و منابع جهت مطالعه‌ی بیشتر

موضوع خوشبندی برای بیش از ۴۰ سال به صورت گستردگی مورد مطالعه قرار گرفته و در شاخه‌های گوناگون علمی از آن استفاده شده است. کتاب‌هایی که پیرامون داده‌کاوی به رشتہ‌ی تحریر درمی‌آیند، به طور حتم حداقل فصلی را به صورت کامل به این مبحث اختصاص می‌دهند. کتاب‌های [Har75], [JD88], [KR90] و [AHS96] از منابعی هستند که فقط به این مبحث می‌پردازنند. مقالاتی نیز وجود دارند که مروری بر جنبه‌های مختلف روش‌های خوشبندی دارند که [Jai10], [PHL04] و [JMF99] از آن جمله محسوب می‌شوند.

الگوریتم k-Means ابتدا در [Llo57] ارائه و پس از آن نیز در [Mac67] مطالعه شد. تعمیمی از این الگوریتم رایج را می‌توانید در [AV07] و [KMN⁺02] پیدا کنید. یک الگوریتم مقیاس‌پذیر مبتنی بر k-Means در [BFR98] بحث شده است. الگوریتم‌های PAM و CLARA در [KR90] پیشنهاد شدند. اطلاعاتی درباره‌ی الگوریتم k-Modes را در [Hua98] [CGC94] و [CGC01] جستجو کنید. الگوریتم CLARANS در [NH94] ارائه و پس از آن در [EKX95] پیشنهاداتی جهت بهبود آن داده شد.

مروری بر روش‌های خوشه‌بندی سلسله‌مراتبی که به صورت پایین به بالا عمل می‌کنند در [DE84] آمده است و در [KR90] می‌توانید نکاتی را برای دو روش سلسله‌مراتبی ادغام و تقسیم پیدا کنید. یک روش جهت بهبود کیفیت در خوشه‌بندی سلسله‌مراتبی کار هم قرار دادن این روش‌ها با تکنیک‌های دیگر خوشه‌بندی است. در مورد الگوریتم BIRCH به [ZRL96]، در مورد الگوریتم CURE به [GRS98]، در مورد الگوریتم ROCK به [GRS99] و در مورد الگوریتم Chameleon به [HG05] نیز منبع دیگری است که به مطالعه‌ی یک رجوع کنید. مرجع [KHK99] رجوع کنید. مرتع [HG05] نیز منبع دیگری است که به فرمول احتمال بیز می‌پردازد.

فصل دهم

مباحثی چند در داده‌کاوی

با آگاهی به این نکته که داده‌کاوی کار خود را اوایل ۱۹۸۰ آغاز نموده، به عنوان یک شاخه‌ی تحقیقاتی جدید پیشرفت قابل توجهی را از خود نشان داده است. امروزه داده‌کاوی در حوزه‌های متعددی استفاده می‌شود و محصولات تجاری متفاوتی برای آن وجود دارد. با این وجود چالش‌های زیادی هنوز باقی مانده است.

در این فصل ابتدا اشاره‌ای به برخی از کاربردهای عملی داده‌کاوی خواهیم داشت، هر چند حوزه‌ی عملی داده‌کاوی به این نمونه‌ها محدود نمی‌شود. سپس موضوعاتی که در چگونگی انتخاب یک سیستم نرم‌افزاری داده‌کاوی مطرح هستند همراه با چند نمونه از سیستم‌های تجاری موجود بیان می‌شوند. شکل‌های دیگری از داده‌کاوی مانند متن‌کاوی،

وب‌کاوی، استخراج الگوها از میان گراف‌ها و کاوش در میان داده‌های مکانی و همچنین چندرسانه‌ای بحث‌هایی هستند که در بخش‌های انتهایی به آن می‌پردازیم.

۱۰-۱) چند مثال کاربردی در داده‌کاوی

در فصل‌های قبل به مطالعه‌ی مفاهیم و روش‌های داده‌کاوی پرداختیم، اما به دلیل آنکه داده‌کاوی نسبتاً یک شاخه‌ی جوان با کاربردهای متنوع را در بر می‌گیرد، هنوز فاصله‌ی زیادی میان این مفاهیم و ابزارهای موثر تولید شده برای داده‌کاوی وجود دارد. در این بخش به برخی از کاربردهای داده‌کاوی اشاره می‌شود و برای هر یک شرح مختصراً در مورد چگونگی انتخاب الگوریتم‌های داده‌کاوی برای استفاده در زمینه‌ی مورد نظر داده خواهد شد.

۱۰-۱-۱) داده‌کاوی و بانکداری

شاید استفاده از داده‌کاوی در مسائل مالی اولین کاربردی باشد که بتوان در این حوزه از آن نام برد. بانک‌ها دارای طیف وسیعی از سرویس‌های مالی هستند و بسیاری از داده‌های جمع‌آوری شده در این موسسات مالی برای داده‌کاوی کامل و قابل اعتمادند. کیفیت خوب داده‌ها باعث می‌شود تا تحلیل داده‌ها و عملیات داده‌کاوی تسهیل شوند. در اینجا به برخی از مواردی که با کمک تکنیک‌های داده‌کاوی می‌توان به کمک صنعت بانکداری آمد بررسی می‌شوند.

- همانند بسیاری از کاربردها، نیاز جهت ساخت انبار داده‌ها برای داده‌های بانک‌ها و موسسات مالی می‌تواند بسیار مفید باشد. پس از آن برای تحلیل خصوصیات داده‌ها به روش‌های تحلیل داده‌های چندبعدی نیاز داریم. ابعاد در این داده‌ها می‌تواند هر یک از صفات خاصه‌ای باشد که در پایگاه‌داده‌های بانک می‌توان آنرا یافت.

- یکی از خدماتی که در موسسات مالی مانند بانک‌ها وجود دارد، اعطای تسهیلات مالی از جمله وام‌ها است. تحلیل‌های متفاوتی را می‌توان با کمک داده‌کاوی برای این سرویس بانک‌ها طراحی نمود. برای مثال می‌توانیم با کمک روش‌های

رتبه‌بندی صفات خاصه، فاکتورهای مهم در اعطای وام را بررسی کنیم. نمونه‌ی دیگر اینکه با الگوریتم‌های طبقه‌بندی می‌توانیم ریسک اعطای وام به مشتریان بعدی (مانند مشتریان بدحساب) را کمتر کنیم. همچنین با کمک روش‌های خوشبندی مشتریان در گروه‌های متفاوت قرار می‌گیرند که هر گروه دارای مشتریانی با رفتار مشابه هستند. در این صورت مدیر می‌تواند تصمیم مناسب و متفاوتی را برای هر خوشی یا گروه اتخاذ کند.

- یکی از موضوعاتی که در موسسات مالی از اهمیت بالایی برخوردار است، مواجهه با افراد سودجو و مجرم است. پول‌شویی و انواع خسارت‌های مالی که توسط این افراد به این موسسات وارد می‌شود، می‌تواند ضربه‌ی جبران ناپذیری را به دنبال داشته باشد. با کمک روش‌هایی مانند خوشبندی و طبقه‌بندی و همچنین ابزارهای بصری‌سازی می‌توان الگوهای خاصی را در میان داده‌ها یافت که با کمک آنها روابط مشکوک بین داده‌ها تشخیص داده شوند.

(۱۰-۱-۲) داده‌کاوی و فروشگاه‌ها

از آنجا که فروشگاه‌ها معمولاً مقدار داده‌های حجمی در مورد خرید مشتریان و اطلاعاتی در مورد اجناس را در خود نگهداری می‌کنند، حوزه‌ی مناسبی برای تحلیل این داده‌ها با کمک روش‌های داده‌کاوی تلقی می‌شوند. این داده‌ها به شدت در حال رشد هستند، به خصوص اینکه امروزه اینترنت و تجارت الکترونیکی کار را راحت‌تر و حجم اطلاعات را دوچندان نموده است. بسیاری از این مراکز علاوه بر مکان‌های خاص خرید و فروش از طریق سایتهايی که راهاندازی کرده‌اند نیز فعالیت می‌کنند. حتی برخی از آنها (مانند سایت معروف *AMAZON*) سرویس‌های خود را فقط از طریق سایت دنبال می‌کنند، بدون آنکه دارای مکانی (فیزیکی) جهت انجام این کار داشته باشند. به همین دلیل این داده‌ها منبع بسیار غنی برای تکنیک‌های داده‌کاوی به شمار می‌روند.

معرفی رفتار مشتریان در رابطه با خرید اجناس، کشف الگوهای خرید آنها، بهبود در کیفیت سرویس‌دهی به خریداران، کسب رضایت مشتری، افزایش نرخ مصرف اجناس، طراحی یک سیستم کارا جهت توزیع و انتقال کالاها و کاهش هزینه‌ها از جمله اهدافی

هستند که می‌توان با کمک داده‌کاوی حاصل شوند. در ادامه به برخی از این کاربردهای داده‌کاوی در صنعت خردفروشی اشاره می‌شود.

- چون داده‌های این صنعت بطور معمول از طیف وسیعی از انواع اطلاعات تشکیل می‌شود، روش‌های متعددی نیز برای طراحی انبار داده‌ها وجود دارد. اینکه ابعاد انبار داده‌ها کدامیک از این اطلاعات انتخاب شوند، به نقطه نظر کاربر حرفاء بر می‌گردد. اما توصیه می‌شود چندین انبار داده‌ها با تنوع ابعاد و سطوح امتحان شوند. پس از آن بسیار مهم است که از ابزار قدرتمندی جهت بصری‌سازی و تحلیل این داده‌ها استفاده شوند.
- فروشگاه‌ها تلاش بسیار زیادی برای فروش و در نتیجه کسب سود بیشتر اجناس خود دارند. تبلیغات و ارائه‌ی انواع مختلف تخفیف‌ها می‌تواند مشتری‌های بیشتری را جذب کنند. تحلیل محتاطانه از داده‌ها می‌تواند در جهت منافع بیشتر سازمان یا شرکت انجام شود. یافتن وابستگی‌ها میان اقلام داده‌ها و یا بدست آوردن قوانین انجمنی یکی از عملیات رایج برای این نوع از کارها بشمار می‌رود. تحلیل مسائلی چون تعداد تراکنش‌های حاوی برخی از اقلام داده‌ها قبل و بعد از تبلیغ یک جنس، می‌تواند تأثیر اینگونه عملیات را بررسی کند. یافتن اقلامی که معمولاً با یکدیگر در سبد خرید مشتریان قرار می‌گیرند نیز کاربرد رایج دیگری است.
- امروزه خرید به صورت نقدی در فروشگاه‌ها روز به روز کاهش می‌یابد و مشتریان از کارت‌های اعتباری خود جهت انجام این کار استفاده می‌کنند. تحلیل بر روی داده‌های کارت‌ها و یافتن مشتریان وفادار (مشتریان همیشگی) می‌تواند مدیر را در تصمیم‌گیری‌های مناسب کمک کند. تحلیل سبد خرید این مشتریان و همچنین خرید آنها در محدوده‌های زمانی مشخص می‌تواند کمک کند تا قیمت این اجناس تنظیم و پیشنهادهای دیگری به آنها داده شود. تصور کنید که پس از اعمال روش‌های داده‌کاوی به این الگو دست می‌یابیم " ۷۰ درصد از مشتریانی که کالای X را می‌خرند پس از آن کالای Y را نیز ابیاع می‌کنند." این الگو کمک می‌کند تا تمهداتی را جهت چیدمان این دو نوع کالا به کار ببریم

و یا به ۳۰ درصد از مشتریانی که X را خریده‌اند اما تاکنون موفق به خرید Y نشده‌اند، اطلاع دهیم که این نوع کالا نیز وجود دارد.

(۱۰-۱) داده‌کاوی و مخابرات

در این چند سال اخیر صنعت مخابرات رشد چشمگیری داشته است. از سرویس‌های محلی و راه دور گرفته تا سرویس‌های ارتباطی پیشرفته مانند تلفن‌های سولی، انتقال تصاویر و سرویس‌های متنوع اینترنتی را می‌توان امروزه در همه جای دنیا پیدا کرد. به علاوه با رفع برخی از ممنوعیت‌های قانونی در این حوزه و در بعضی از کشورها و همچنین توسعه‌ی تکنولوژی کامپیوترهای جدید، رقابت در بازار این صنعت به شدت گرم شده است. این مسئله باعث ایجاد تقاضا جهت تحلیل داده‌های جمع‌آوری شده است. بهبود کیفیت سرویس‌ها، استفاده‌ی بهینه از منابع، جلوگیری از فعالیتهای مجرمانه و تقلب و شناسایی الگوهای ارتباطی می‌تواند از اهداف اصلی استفاده از تکنیک‌های داده‌کاوی باشند. به برخی از کاربردهای داده‌کاوی در این حوزه به صورت مختصر در ادامه اشاره می‌کنیم.

- داده‌های مخابراتی با صفات خاصه و ابعادی نظیر زمان مکالمه، محل جغرافیایی طرفین مکالمه، نوع مکالمه و تاریخ آن یک داده‌ی چندبعدی تلقی می‌شود. استفاده از ابزاری جهت تحلیل و بررسی این داده‌ها می‌تواند نتایج مناسبی را به همراه داشته باشد. کشف مشارکت و الگوهای مکرر در این داده‌ها باعث ارتقای سرویس‌های مخابراتی خواهد شد. با یافتن الگوهای مکرر قادریم سرویس‌های خاصی مانند تلفن‌های همراه را در زمان‌ها و مکان‌های مشخص ارتقا دهیم.
- هر ساله حملات مختلف سایبری که به موسسات مخابراتی می‌شود، برای آنها هزینه‌های زیادی را به دنبال دارد. این مسئله بسیار مهم است که قادر باشیم کاربران فریبکار را از دیگران تشخیص دهیم. الگوریتم‌هایی در داده‌کاوی که می‌توانند این الگوهای خارج از محدوده را تشخیص دهند، در این نوع از مسائل قادر است به کاربران و مدیران کمک کند.
- از آنجا که امروزه سرویس‌های مخابراتی فقط به سرویس‌های تلفنی محدود نمی‌شوند و سرویس‌هایی نظیر اینترنت را نیز در دستور کار خود دارند، تحلیل این

داده‌ها نیز مسیر را برای سرویس‌دهی بهتر آماده می‌کند. برخی از این سرویس‌ها دارای تنوع داده‌ای هستند و برای تحلیل آنها به تکنیک‌های خاص نیاز است. برای مثال مجموعه داده‌هایی که از یک دستگاه *GPS* جمع‌آوری می‌شوند، داده‌های مکان‌محور^۱ هستند و اطلاعات مکانی را در خود ذخیره می‌کنند و ساختار مخصوص به خود را دارند.

۴-۱-۱۰) داده‌کاوی و بیوانفورماتیک

بیوانفورماتیک^۲ شاخه‌ای از علم زیست‌شناسی است که به ذخیره و تحلیل و تفسیر اطلاعات آزمایشگاهی می‌پردازد. هدف اصلی بیوانفورماتیک استخراج اطلاعات و شناسایی روابط بین مجموعه اطلاعات انبویی است که در علم زیست‌شناسی به دست آمده است. اغلب داده‌ها شامل توالی اسیدهای نوکلئیک و پروتئین، ساختارهای پروتئینی، پروفایل‌های بیان ژن، مسیرهای متابولیکی یا بیوشیمیابی هستند. از نقطه نظر علم زیست‌شناسی می‌توان گفت که بیوانفورماتیک عبارت است از ذخیره‌سازی و تجزیه و تحلیل و بازیابی داده‌های زیستی، توالی اسیدهای نوکلئیک، توالی‌های پروتئینی و اطلاعات ساختاری است.

چندین پایگاه داده‌ی معروف در زیست‌شناسی مولکولی وجود دارند که در آن توالی‌های زیادی را می‌توان یافت. تعداد توالی‌های موجود در این پایگاه داده‌ها به قدری زیاد است که نیاز به الگوریتم‌های کامپیوتربهای تجزیه و تحلیل آنها به خوبی احساس می‌شود. در ضمن تعداد این پایگاه‌داده‌ها و محتویات آنها به سرعت در حال رشد است. بنابراین با توجه به حجم بسیار بالای داده‌ها، علم بیوانفورماتیک در صدد برآمده است تا با کمک روش‌هایی چون داده‌کاوی به بررسی کارآمد و مدیریت بهتر اطلاعات مختلف بتواند تجزیه و تحلیل انواع مختلف داده‌ها (مانند توالی‌های *DNA* و پروتئین، تعیین و پیش‌بینی ساختارهای پروتئینی و چارچوب عمل ژن) را به درستی انجام دهد.

¹ Spatial Dataset

² Bioinformatic

حوزه‌ی عمل در علم بیوانفورماتیک به طور عمده شامل سه حوزه‌ی تجزیه و تحلیل توالی‌های مولکولی، ساختارهای مولکولی و تجزیه و تحلیل عملکردهای مولکولی است. تکنیک‌های مختلف داده‌کاوی مانند طبقه‌بندی، خوشه‌بندی و قوانین انجمنی را می‌توان برای تحلیل داده‌ها در این حوزه به کار برد. اما یافتن توالی‌های مکرر از رایج‌ترین عملیاتی هستند که در آن داده‌کاوی به کمک علم بیوانفورماتیک آمده است. منصفانه نیست که بخش کوچکی از یک کتاب را به کاربرد داده‌کاوی در علم بیوانفورماتیک تخصیص دهیم، چرا که مطالب در این حوزه بسیار وسیع است و ما این کار را به عهده متخصصین خودش واگذار می‌کنیم.

۲-۱۰) نکاتی در مورد محصولات تجاری داده‌کاوی

اگر چه داده‌کاوی یک حوزه‌ی علمی تقریباً جدید تلقی می‌شود و هنوز کارهای انجام نشده در آن بسیار است، اما می‌توان محصولات و برنامه‌های کاربردی متنوعی را در این حوزه در بازار پیدا کرد. بنابراین با این تاریخچه‌ی کوتاه، افزایش ویژگی‌های متنوع و یافتن یک زبان استاندارد و بررسی بسیاری از موارد دیگر در دستور کار محققان این زمینه قرار دارند. در این بخش نکاتی پیرامون چگونگی انتخاب یک محصول تجاری داده‌کاوی و همچنین اشاره‌ای کوتاه به چند برنامه‌ی کاربردی موجود در این زمینه خواهیم داشت. شاید ذکر مکرر این نکته ملال آور باشد که یادآوری کنیم بررسی هر یک از محصولات تجاری داده‌کاوی نیاز به فرصت و فضای مناسب و بیشتری جهت نمایش جزئیات هر یک از آنها دارد.

۱-۱۰) موضوعاتی چند جهت چگونگی انتخاب یک سیستم

بدون شک با تنوع نرم‌افزارهای موجود در زمینه‌های مختلف، یکی از دغدغه‌های بزرگ کاربران انتخاب بهترین از میان آنها است. برخی از سیستم‌های نرم‌افزاری کنونی مانند سیستم‌های مدیریت پایگاه داده‌ی رابطه‌ای، پیشرفتهای شایانی را داشته‌اند و بسیاری از آنها دارای ویژگی‌های مشترک و یکسانی هستند. اما نوپا بودن سیستم‌های داده‌کاوی

- باعث شده است که اشتراک‌های زیادی میان آنها مشاهده نشود. از این رو معیارها و شرایطی جهت انتخاب یکی از آنها وجود دارند که در ادامه به برخی از آنها اشاره می‌شود.
- اکثر سیستم‌های داده‌کاوی در دسترس با داده‌های مبتنی بر رکورد و جدولی کار می‌کنند. عملکرد بسیاری از روش‌ها و الگوریتم‌های داده‌کاوی به نوع داده‌های ورودی این تکنیک‌ها بستگی دارد. بنابراین شاید مجبور باشیم از یک سیستم داده‌کاوی مشخص جهت کار بر روی داده‌ها استفاده کنیم که داده‌هایی همچون متن، چندرسانه‌ای و ... را پشتیبانی می‌کند. هرچند محصولات کمپانی‌ها عموماً به صورتی است که ترجیح می‌دهند بسیاری از روش‌های داده‌کاوی را در خود جای دهند، اما نوع داده‌هایی که کاربر با آنها سروکار دارد در انتخاب بسته‌ی نرم‌افزاری بی‌تأثیر نیست.
 - امکانات سخت‌افزاری و نرم‌افزاری نیز در انتخاب یک محصول موثر است. اغلب به دلیل پیشرفت خوب و سریع در سخت‌افزارها در این مورد محدودیتی دیده نمی‌شود. اما سیستم عامل‌ها می‌توانند انتخاب شما را در محصول نرم‌افزاری تحت الشعاع خود قرار دهند. اکثر سیستم عامل‌های محبوب و رایج مانند یونیکس و ویندوز پذیرای اغلب نرم‌افزارهای داده‌کاوی هستند، هر چند برخی از آنها سیستم عامل‌های دیگر را نیز پشتیبانی می‌کنند.
 - شاید بدون اغراق بتوان ادعا نمود که تنوع الگوریتم‌های موجود در یک بسته‌ی نرم‌افزاری داده‌کاوی اولین عاملی است که در انتخاب آن توسط کاربر بررسی می‌شود. برخی از این بسته‌ها فقط تعداد محدودی از تکنیک‌ها و الگوریتم‌ها را پشتیبانی می‌کنند. بعضی دیگر طیف وسیعی از روش‌ها را در خود دارند، ولی برای هر یک از روش‌ها دارای الگوریتم‌های کم و محدودی هستند. بدون شک سیستم‌هایی که از تکنیک‌های مختلفی مانند طبقه‌بندی، خوشه‌بندی و قوانین انجمنی پشتیبانی می‌کنند و برای هر یک از آنها الگوریتم‌های متعددی را تحت اختیار کاربر قرار می‌دهند از انعطاف‌پذیری بالاتری برخوردارند. برای بسیاری از مسائل شاید بهتر باشد کاربران نتایج خروجی چندین الگوریتم مختلف داده‌کاوی را مشاهده و تحلیل کنند.

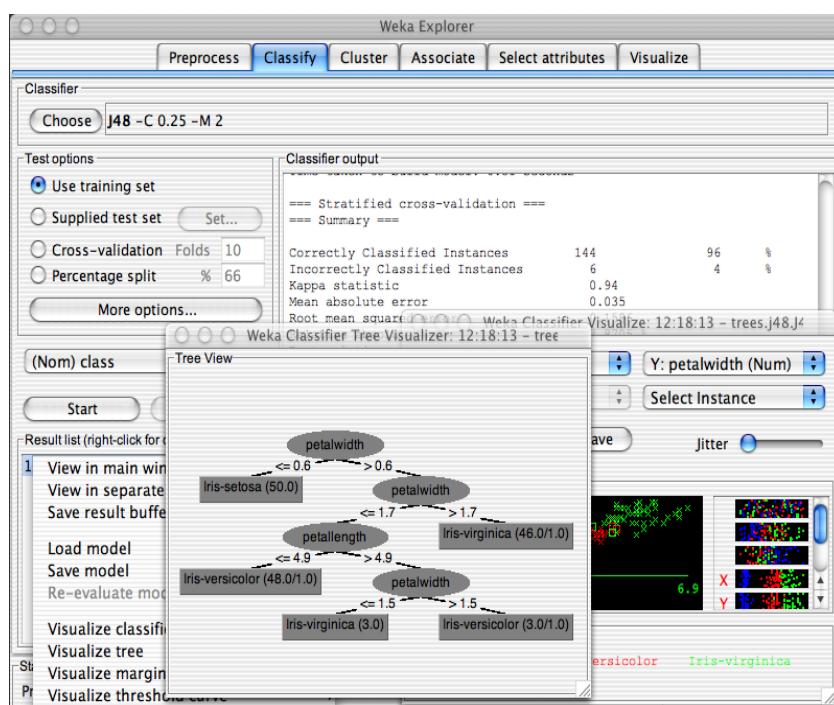
- همان‌طور که در فصل‌های قبلی نیز به آن اشاره شد، قابلیت مقیاس‌پذیری الگوریتم‌ها یکی از مزایای مهم آنها محسوب می‌شود. یک الگوریتم هنگامی مقیاس‌پذیر خواهد بود که با افزایش حجم داده‌ها سرعت اجرای الگوریتم به صورت تقریباً خطی رشد کند. یافتن نرم‌افزارهایی که با حجم بالای داده‌ها (چه از نظر تعداد نمونه‌ها و چه از نظر ابعاد) به خوبی کنار می‌آیند، چالش بزرگی است که محققان حوزه‌ی داده‌کاوی را به تلاش جهت یافتن الگوریتم‌های مناسب وادار نموده است.
- وجود ابزارهای بصری‌سازی مناسب در سیستم‌های داده‌کاوی، کاربران را در تحلیل راحت‌تر و دقیق‌تر کمک می‌کند. اغلب نمایش یک عکس و یا تصویر ارزشی برابر با صدها کلمه در بر دارد. ابزار تجسم‌سازی و نمایش می‌تواند جهت نمایش مناسب داده‌ها، نتایج و یا فرایند داده‌کاوی استفاده شود. تنوع، کیفیت و انعطاف‌پذیری این گونه ابزارها می‌تواند به شدت جذابیت و قابلیت تفسیر ساده‌تر نتایج داده‌کاوی را به دنبال داشته باشد.
- عدم وجود یک واسط کاربر مناسب می‌تواند شکست یک بسته‌ی نرم‌افزاری قدرتمند را به دنبال داشته باشد. واسطی که می‌توان از آن به راحتی استفاده نمود و دارای گرافیک مناسبی است یک مزیت مهم برای سیستم‌های داده‌کاوی تلقی می‌شود. فقدان یک زبان استاندارد برای داده‌کاوی نیز یکی از مشکلاتی است که شرکت‌های تولید نرم‌افزار را درگیر کرده است. البته اخیراً تلاش‌هایی برای معرفی و استانداردسازی زبان‌های پرس‌وجوی داده‌کاوی انجام شده است که می‌توان از آن بین به موردهایی نظیر *OLE DB* مایکروسافت و *CRISP-DM* و *PMML* همچنین *PMMML* اشاره نمود.

(۱۰-۲) چند بسته‌ی نرم‌افزاری داده‌کاوی

در این بخش بطور خلاصه به نام برخی از سیستم‌های تجاری داده‌کاوی اشاره‌ای می‌کنیم تا خواننده‌ی کتاب بداند در زمان طبع آن چه نرم‌افزارهایی در دسترس بودند. البته بسیاری از این نرم‌افزارها هر ساله نسخه‌ها و ویرایش‌های جدیدتری را به بازار عرضه می‌کنند که

دارای امکانات بهتری نسبت به نسخه‌های قبلی آنها است. می‌توان این نرم‌افزارها را با دیدگاه‌های متفاوتی دسته‌بندی نمود. برای مثال بعضی از آنها نرم‌افزارهای آماری شناخته می‌شوند و بعضی دیگر از زاویه‌ی دید پایگاه‌داده‌ها طراحی شده‌اند. اما از آنجا که قصد نداریم به جزیيات آنها بپردازیم، این کار را به عهده‌ی خواننده گذاشته تا اگر مایل باشد بر رفتن به سایت مخصوص مورد نظر اطلاعات بیشتری کسب کند.

- نرم‌افزار **WEKA** یک بسته‌ی نرم‌افزاری منبع‌باز است که دانشگاهی در نیوزلند آن را با جاوا طراحی و پیاده‌سازی نموده است. این نرم‌افزار شامل مجموعه‌ای از الگوریتم‌های داده‌کاوی مانند طبقه‌بندی، خوشه‌بندی، رگرسیون، عملیاتی جهت پیش‌پردازش داده‌ها، یافتن وابستگی‌ها و ابزاری برای بصری‌سازی است. شکل ۱۰-۱ تصویری از محیط این برنامه را نشان می‌دهد، که در آن یک درخت تصمیم‌رسم شده‌است.

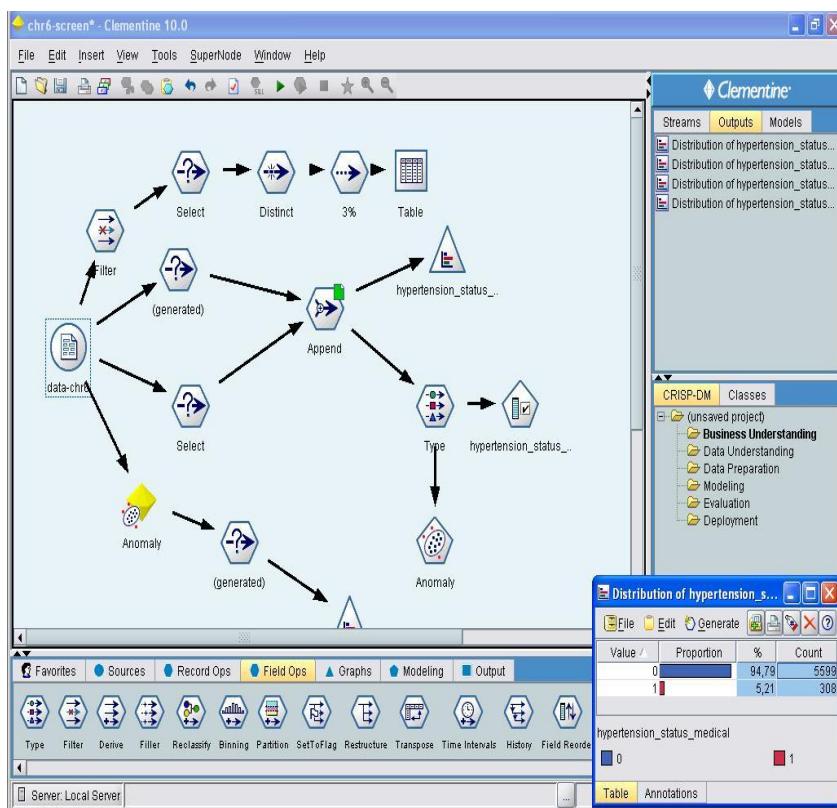


شکل ۱۰-۱: تصویری از برنامه‌ی کاربردی WEKA

- کمپانی *IBM* محصول داده‌کاوی *Intelligent Miner* را همراه با طیف وسیعی از تکنیک‌های داده‌کاوی شامل طبقه‌بندی، رگرسیون، مدل‌سازی جهت تخمین، خوشه‌بندی، کاوش وابستگی‌ها و تحلیل الگوهای مکرر ارائه کرده است. جعبه ابزارهایی برای الگوریتم‌های شبکه‌های عصبی، روش‌های آماری، ابزار آماده‌سازی داده‌ها و همچنین بصری‌سازی داده‌ها را می‌توانید به این نرم‌افزار اضافه کنید. ویژگی شاخص این نرم‌افزار قابلیت مقیاس‌پذیری الگوریتم‌های داده‌کاوی موجود در آن و نزدیکی آن با سیستم پایگاه داده‌ی رابطه‌ای *DB2* است.
- نرم‌افزار *SAS Enterprise Miner* توسط شرکت *SAS* توسعه داده شده است و همانند دیگر محصولات شامل تکنیک‌های طبقه‌بندی، رگرسیون، خوشه‌بندی، کاوش وابستگی‌ها، تحلیل داده‌های نوع سری‌های زمانی و همچنین بسته‌ی نرم‌افزاری جهت تحلیل‌های آماری است. تنوع ابزارهایی که برای تحلیل‌های آماری در این نرم‌افزار وجود دارد، مشخصه‌ی بارز آن است که البته قدمت شرکت *SAS* در این حوزه آن را باعث شده است.
- *SQL Server 2005* از مایکروسافت یک سیستم مدیریت پایگاه داده است که چندین تابع داده‌کاوی را در سیستم پایگاه داده‌ی رابطه‌ای خود و محیط‌های سیستم انبارداده‌ها گنجانده است. کاوش وابستگی‌ها، طبقه‌بندی (درخت تصمیم، بیز و الگوریتم‌های شبکه عصبی)، درختان رگرسیون، خوشه‌بندی و تحلیل سری‌های زمانی را می‌توان در آن یافت. با توجه به اینکه این بانک اطلاعاتی به عنوان یکی از قویترین سیستم‌های مدیریت پایگاه داده‌ی رابطه‌ای مخصوصاً با حجم بالای داده شناخته می‌شود، استفاده از آن در داده‌کاوی جهت کار با حجم وسیع داده‌ها می‌تواند انتخاب خوبی باشد.
- شرکت *SPSS* نرم‌افزار *Clementine* را برای کاربران آمار و داده‌کاوی تهیه نموده است. در این نرم‌افزار می‌توانید مجموعه‌ای از توابع را برای طبقه‌بندی، خوشه‌بندی، تخمین و کاوش وابستگی‌ها پیدا کنید. یکی از ویژگی‌های مهم این نرم‌افزار واسط کاربری است که به صورت شئی‌گرا توسعه یافته و به کاربران

الگوریتم‌ها اجازه می‌دهد تا به محیط برنامه‌نویسی بصری اضافه شوند. در شکل

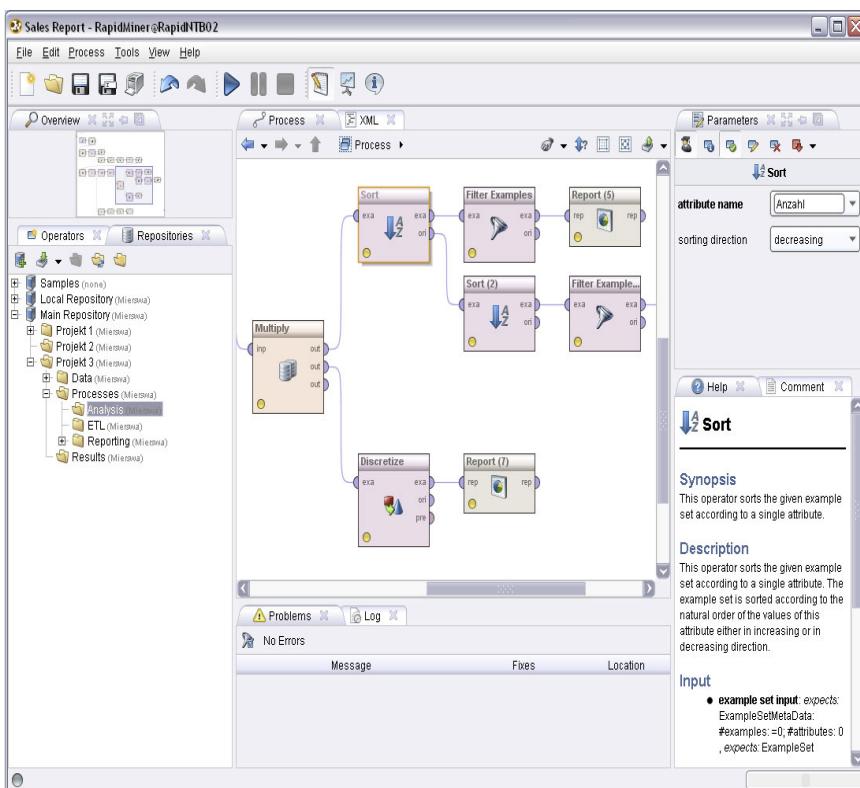
۱۰-۲ تصویری از یک مثال در محیط این نرم‌افزار نشان داده شده است.



شکل ۱۰-۲: تصویری از نرم‌افزار Clementine

- نرم‌افزار *MineSet* در سال ۱۹۹۹ توسط *SGI* معرفی شد و شامل کاوش قوانین وابستگی، طبقه‌بندی و همچنین ابزارهای پیشرفته‌ی آماری و قدرتمند بصری‌سازی است. ویژگی ممیزه‌ی آن ابزار قدرتمند گرافیکی آن است که این نرم‌افزار را به نحوی از دیگر محصولات جدا می‌سازد.
- نرم‌افزار *Insightful Miner* از شرکت *Insightful* هم دارای چندین روش داده‌کاوی شامل پالایش داده‌ها، طبقه‌بندی، تخمین، خوشبندی، بسته‌ی تحلیل آماری همراه با ابزار بصری‌سازی است. ویژگی خاصی که در آن می‌توان مشاهده کرد، واسط بصری آن است که کاربر را قادر می‌سازد با اتصال اجزاء و بخش‌ها مستندسازی کند.

- نرم افزار منبع باز *Rapid-I* از گروه *RapidMiner* محصولی است که دارای امکاناتی جهت پیش پردازش داده ها و تکنیک های متنوع داده کاوی از جمله طبقه بندی، خوشه بندی، تخمین و کاوش وابستگی ها است. این نرم افزار با مجموعه بسته هایی که به آن افزوده می شود، می تواند با داده های حجمی نیز به خوبی عمل کند. این بسته ها به قدرت نرم افزار می افزایند. برای مثال بسته متن کاوی باعث می شود تا کاربر بتواند الگوریتم ها را بر روی مجموعه داده های نیمه ساخت یافته اجرا کند. در ضمن می توان الگوریتم های *WEKA* را وارد این نرم افزار نمود و از آنها نیز استفاده کرد. در شکل ۱۰-۳ می توانید رابط کاربر این نرم افزار را مشاهده کنید.



شکل ۱۰-۳: رابط کاربر نرم افزار *RapidMiner*

بدون شک بسته‌های نرم‌افزاری دیگری را نیز می‌توان یافت که شاید به صورت خاص برای تکنیک‌های مشخص داده‌کاوی پیاده‌سازی شده‌اند. برای مثال *CART* نرم‌افزاری است که از نامش نیز می‌توان حدس زد، جهت ساخت درختان تصمیم و رگرسیون استفاده می‌شود.

۱۰-۳) شکل‌های دیگری از داده‌کاوی

در فصل‌های قبلی به معرفی مفاهیم اساسی و تکنیک‌های داده‌کاوی پرداختیم. مجموعه داده‌های ورودی در الگوریتم‌های مطالعه شده معمولاً از ساختار ساده‌ای برخوردار بودند. جداول در مدل رابطه‌ای، پایگاه داده‌ی تراکنشی و انبار داده‌ها از این ساختارها به شمار می‌روند. رشد بی‌رویه‌ی داده‌ها در شکل‌ها و ساختارهای پیچیده‌تر روش‌های داده‌کاوی را نیز تحت تأثیر خود قرار داده است. در این بخش بصورت خلاصه چگونگی برخورد تکنیک‌های داده‌کاوی را با این نوع از داده‌های خاص بررسی می‌کنیم. کاوش در میان داده‌های متنی یا بصورت خلاصه متن‌کاوی، یافتن الگوهای مفید در مجموعه داده‌های چندرشته‌ای، کار بر روی گراف‌ها، جستجو در داده‌های وب یا وب‌کاوی و استخراج الگوها از میان داده‌های مکانی موضوعات اصلی این بخش را تشکیل می‌دهند. هر یک از این موضوعات نیاز به بحث گسترده‌ای دارد که از حدود کتاب حاضر خارج است.

۱-۳-۱) متن‌کاوی^۱

اکثر روش‌های داده‌کاوی بر روی داده‌های ساخت‌یافته مانند جداول مرکز هستند، حال آنکه حجم وسیعی از اطلاعات در دسترس در دنیای بیرون در پایگاه داده‌ی متنی ذخیره شده‌اند. این پایگاه داده شامل مجموعه‌ی بزرگی از مستندات متنی مانند کتاب‌ها، مقالات، کتابخانه‌های دیجیتال و صفحات وب می‌شوند. امروزه بسیاری از سازمان‌ها اطلاعات خود را در قالب متن نگهداری می‌کنند و این موضوع اهمیت استفاده از تکنیک‌های داده‌کاوی را برای این نوع از داده‌ها دوچندان کرده است. عموماً این داده‌ها نیمه‌ساخت‌یافته هستند.

^۱ Text Mining

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. تحقیقات فراوانی بر روی داده‌های نیمه‌ساخت‌یافته صورت گرفته است و به علاوه تکنیک‌های بازیابی اطلاعات نیز جهت کار بر روی اسناد غیرساخت‌یافته توسعه داده شده است. تکنیک‌های سنتی قبلی جهت بازیابی اطلاعات در میان حجم وسیعی از داده‌ها کافی نیستند. کاربران به ابزاری نیاز دارند تا اسناد مختلف را مقایسه کنند و یا یک رتبه‌بندی بین آنها داشته باشند. یافتن الگوها در این متون می‌تواند برای کاربر مفید باشد. به همین دلیل متن کاوی به یک شاخه‌ی محظوظ در موضوع داده‌کاوی تبدیل شده است.

برای سال‌ها حوزه‌ی بازیابی اطلاعات همراه با سیستم‌های پایگاه داده‌ها به صورت موازی توسعه یافته‌اند. برخلاف سیستم‌های پایگاه داده‌ها که بر روی پرس‌وجو و پردازش تراکنش‌های داده‌های ساخت‌یافته مرکز شده است، دغدغه‌ی حوزه‌ی بازیابی اطلاعات، سازماندهی و بازیابی اطلاعات از حجم وسیعی از مستندات متنی است. از آنجا که هر یک از این دو بر روی نوع داده‌های خاص خود کار می‌کنند، در برخی از ویژگی‌ها با یکدیگر متفاوتند. برای مثال لازم نیست به مفاهیمی چون کنترل همرونندی^۱، ترمیم^۲ و مدیریت تراکنش‌ها در بازیابی اطلاعات توجه شود و در این شاخه با مسائلی چون کار بر روی مستندات غیرساخت‌یافته و یا جستجوی تقریبی بر اساس کلمات کلیدی مواجه می‌شویم. قرار دادن اسناد مرتبط با یکدیگر در یک گروه که بر اساس پرس‌وجوی کاربر جمع‌آوری شده است، از نمونه مسائلی است که در بازیابی اطلاعات مطرح است. این مجموعه اسناد که در یک گروه قرار گرفته‌اند، می‌تواند شامل مجموعه‌ای از کلمات کلیدی باشند که اطلاعات لازم را در خود نگهداری می‌کنند. روش‌های متعددی برای بازیابی متن و همچنین ارزشیابی دقت و صحت خروجی آنها وجود دارند. موضوع دیگری که می‌توان در مورد پایگاه داده‌های متنی ابراز نمود، حجم بالای تعداد اسناد و همینطور واژه‌های موجود در هر یک از آنها است. ابعاد بالا در این نوع از داده‌ها نیز باعث ناکارآمد بودن محاسبات

¹Concurrency Control

²Recovery

بر روی آن خواهد شد. به علاوه در تبدیل داده‌های متنی به شکل جدول با بردارها و ماتریس‌های خالی روبرو هستیم که در آن تشخیص ارتباط میان واژه‌ها ما را با چالش دیگری روبرو می‌کند. جهت غلبه بر این مشکلات می‌توان از تکنیک‌های کاهش ابعاد برای متون استفاده نمود.

بازیابی مبتنی بر شباهت و یا مبتنی بر کلمات کلیدی کمک شایانی به ما می‌کنند، اما داده‌کاوی یک قدم فراتر می‌رود و با کمک طبقه‌بندی یا خوشبندی اسناد به استخراج دانش از داده‌های متنی نیمه‌ساخت‌یافته می‌پردازد. جهت اطلاع بیشتر و دقیق‌تر در مورد متن کاوی به مراجعی که در انتهای فصل معرفی شده‌اند، می‌توانید رجوع کنید.

۱-۳-۲) وب کاوی^۱

با افزایش چشمگیر حجم داده‌ها و همچنین توسعه‌ی وب، نیاز به روش‌ها و تکنیک‌هایی جهت استخراج دانش از این حجم انبوه بیش از پیش احساس می‌شود. در وب کاوی با به کارگیری تکنیک‌های داده‌کاوی به کشف خودکار اطلاعات از اسناد و سرویس‌های وب می‌پردازیم. وب دارای خصوصیات منحصر به فردی است که کاوش و استخراج اطلاعات مفید از داده‌هایی با این مشخصات چالش بزرگی تلقی می‌شود. اجازه دهید برخی از آنها را در ادامه مرور کنیم.

- حجم داده‌ها و اطلاعات بر روی وب بسیار زیاد است و هر روزه نیز رو به افزایش است. تقریباً هر چیزی را بر روی وب می‌توان یافت. به عبارت دیگر داده‌ها می‌توانند از هر نوعی باشند. برای مثال جداول که ساخت‌یافته هستند و یا صفحات وب و متون که به ترتیب در داده‌های نیمه‌ساخت‌یافته و غیرساخت‌یافته گنجانده می‌شوند. فایل‌های چندسانه‌ای نیز مانند تصاویر و صداها از جایگاه ویژه‌ای بر روی وب برخوردارند.

- داده‌ها بر روی وب معمولاً ناهمگن^۲ هستند. بدین ترتیب، در هنگام جمع‌آوری اطلاعات از منابع مختلف با چالش بزرگی روبرو خواهیم بود.

¹ Web Mining

² Heterogeneous

- حجم وسیعی از اطلاعات بر روی وب به یکدیگر لینک (پیوند) شده‌اند. این لینک‌ها در واقع صفحات وب را به یکدیگر پیوند می‌زنند. صفحاتی که توسط بسیاری از صفحات وب دیگر لینک می‌خورند، صفحات پراهمیتی هستند. چرا که این موضوع نشان می‌دهد که کاربران به این صفحات بیشتر اعتماد دارند و به آن رجوع می‌کنند.
- داده‌های نامربوط و مزاحم بر روی وب یافت می‌شوند. یک صفحه‌ی وب شامل اطلاعاتی مانند تبلیغات است که ممکن است کاربر به این اطلاعات نیازی نداشته باشد. بخشی از آنها مفیدند و لذا قبل از اعمال داده‌کاوی شاید بهتر باشد تا داده‌های مربوط جداسازی شوند. در ضمن به دلیل عدم هیچگونه کنترل محتوایی، هر شخص می‌تواند هر چه را که مایل باشد در وب قرار دهد. به همین دلیل حجم وسیعی از اطلاعات موجود در وب دارای کیفیت مناسب نیستند و حتی می‌توانند گمراه کننده نیز باشند.
- می‌توان گفت وب یک محیط پویاست. اطلاعات بر روی آن مدام در حال تغییر است. اجرای الگوریتم‌های داده‌کاوی همراه با تغییر مدام داده‌ها بر روی وب موضوع بسیار مهمی در بسیاری از کاربردهای عملی محسوب می‌شود. وب همانند یک جامعه‌ی مجازی است. به این معنی که فقط شامل داده‌ها و اطلاعات نیست، بلکه تعامل میان مردم، سازمان‌ها و سیستم‌های خودکار را نیز شامل می‌شود. هر شخص یا سازمان قادر است آزادانه نظرات و دیدگاه‌های خود را برای نمایش دیگران بر روی وب قرار دهد.
- تمام خصوصیات اشاره شده در بالا و همچنین بسیاری از خصوصیات دیگر چالش‌ها و فرصت‌هایی را برای کشف و استخراج دانش از وب ایجاد می‌کند. اگر چه در وب کاوی بسیاری از الگوریتم‌های داده‌کاوی استفاده می‌شوند، اما به دلیل ناهمگن بودن داده‌ها و همچنین ماهیت غیرساخت‌یافته و نیمه‌ساخت‌یافته‌ی داده‌های وب، این الگوریتم‌ها باید به طرز قابل ملاحظه‌ای تعمیم داده شوند. در این چند سال اخیر الگوریتم‌های جدیدی نیز برای وب کاوی طراحی و ارائه شده‌اند.

مجموعه وظایف وب‌کاوی را می‌توان در سه دسته‌ی کاوش ساختاری وب^۱، کاوش محتوایی وب^۲ و کاوش استفاده از وب^۳ گروه‌بندی نمود.

نمایش ساختار وب به عهده‌ی لینک‌ها است و کاوش ساختاری وب به استخراج داشت مفید از میان این لینک‌ها می‌پردازد. اهمیت یک صفحه‌ی وب به این طریق مشخص می‌شود که می‌تواند برای موتورهای جستجو مفید باشد. تکنیک‌های داده‌کاوی سنتی قادر

روش‌هایی هستند که ساختار پیوند خورده شده میان چند نمونه داده را بررسی کنند.

محتويات صفحات وب نیز دارای اطلاعات مفیدی است که با طبقه‌بندی و خوشبندی آنها می‌توان به نتایج مناسبی دست یافت. این تکنیک‌ها مشابه همان‌هایی هستند که در فصل‌های قبل توضیح داده شدند. اما الگوهایی نیز ممکن است در محتويات صفحات وب وجود داشته باشند که الگوریتم‌های قبلی قادر به یافتن آنها نباشند.

بدون شک بررسی و ردیابی مسیرهایی از لینک‌ها که کاربران دنبال می‌کنند، یکی از اهداف اصلی در تکنیک‌های وب‌کاوی است. جهت کاوش استفاده از وب می‌توان بسیاری از الگوریتم‌های داده‌کاوی را استفاده نمود. برای مثال یافتن توالی‌های مکرر عالیق کاربران را هدف قرار می‌دهد.

فرایند وب‌کاوی مشابه فرایندی است که در داده‌کاوی انجام می‌شود و تفاوت اصلی در جمع‌آوری داده‌ها خلاصه می‌شود. در حالی که داده‌کاوی سنتی با داده‌های جمع‌آوری شده در انبار داده‌ها سروکار دارد، برای وب‌کاوی جمع‌آوری داده‌ها مرحله‌ی مهمی از این فرایند محسوب می‌شود، به خصوص در کاوش ساختاری و محتوایی وب که مجبور خواهید بود تعداد صفحات وب زیادی را پیمایش کنید.

به هر حال به دلیل تنوع و غنی بودن اطلاعات موجود بر روی وب، تکنیک‌های مختلفی برای وب‌کاوی وجود دارند که حتی در کتاب‌های وب‌کاوی نیز همه‌ی آنها پوشش داده نمی‌شوند. در انتهای فصل به برخی از منابع وب‌کاوی اشاره‌ای شده است.

¹ Web Structure Mining

² Content Mining

³ Web Usage Mining

۱۰-۳) گراف‌کاوی^۱

شاید این ادعا گزارف نباشد که در میان انواع ساختمان داده‌ها گراف‌ها در مدل‌سازی ساختارهای پیچیده از اهمیت بالایی برخوردارند. استفاده‌ی کاربردی آنرا می‌توان در طراحی مدارات الکترونیکی، تصاویر، ساختمان پروتئین‌ها، شبکه‌های زیست‌شناسی، وب و حتی اسناد XML مشاهده نمود. بسیاری از الگوریتم‌های جستجوی گراف‌ها برای کاربردهایی چون بینایی ماشین، شاخص‌بندی ویدئو و بازیابی متون توسعه یافته‌اند. با افزایش تقاضا جهت داده‌های ساخت‌یافته، گراف‌کاوی نیز به عنوان یکی از موضوعات مورد بحث و مهم در داده‌کاوی مطرح شده است.

از میان عملیاتی که بر روی گراف‌ها انجام می‌شوند، یافتن الگوهای مکرر (زیرگراف‌ها) از اساسی‌ترین و رایج‌ترین آن به شمار می‌رود. توصیف مجموعه گراف‌ها، تشخیص تمایز میان آنها، طبقه‌بندی و خوشه‌بندی گراف‌ها، ساخت شاخص برای آنها و همچنین تسهیلاتی جهت جستجو در گراف‌ها از مزایای یافتن الگوهای مکرر در گراف‌ها محسوب می‌شوند. به همین دلیل تکنیک‌های متعددی برای یافتن الگوهای مکرر ارائه شده‌اند که هر یک در شرایط مشخصی به صورت بهینه عمل می‌کنند. الگوریتم‌هایی که در فصل قوانین انجمنی توضیح داده شدند را می‌توان با تغییراتی برای گراف‌ها نیز استفاده کرد. برای مثال استفاده از الگوریتمی موسوم به *Apriori* و همچنین روش کاراتری به نام FP-Growth را می‌توان برای گراف‌ها استفاده کرد. محدودیت‌هایی که توسط کاربران مشخص می‌شوند به الگوریتم‌های گراف‌کاوی کمک خواهد کرد تا الگوهای مناسب و جالبی را تحت اختیار کاربر قرار دهند.

گسترده‌گی استفاده از گراف‌ها باعث شده است تا گراف‌کاوی به حوزه‌ی محبوبی میان محققان تبدیل شود. برای مثال تحلیل شبکه‌های اجتماعی یکی از کاربردهایی است که مفهوم گراف در آن نقش به سزاوی دارد. در این گراف گره‌ها نماینده‌ی نمونه‌ها و یال‌ها روابط میان آنها را معکس می‌کنند. روش‌های داده‌کاوی برای تحلیل شبکه‌های اجتماعی به دنبال بررسی یال‌ها و پیوندهای میان گره‌های این گراف می‌پردازنند. منابع بسیاری

^۱ Graph Mining

وجود دارند که به بررسی تکنیک‌های داده‌کاوی در گراف‌ها توجه می‌کنند. اسامی برخی از آنها را می‌توان در انتهای فصل جستجو کنید.

۴-۳) کاوش در داده‌های چندرسانه‌ای^۱

یک سیستم پایگاه داده‌ی چندرسانه‌ای مجموعه‌ی وسیعی از داده‌های چندرسانه‌ای را ذخیره و مدیریت می‌کند. این داده‌ها می‌توانند صدا، تصویر، ویدئو، گرافیک، متن و حتی داده‌هایی مانند صفحات وب باشند. امروزه ادواتی که این داده‌ها را تولید می‌کنند مانند دوربین‌های دیجیتال را می‌توانیم هر جایی پیدا کنیم و در میان مردم رایج است. لذا پایگاه داده‌ی چندرسانه‌ای از محبوبیت بالایی برخوردار است. از آنجا که واژه‌ی چندرسانه‌ای طیف وسیعی از انواع داده‌ها را تشکیل می‌دهد، اغلب برای بیان روش‌های داده‌کاوی بر روی داده‌های چندرسانه‌ای نوع داده (تصویر، صدا، متن،...) مشخص می‌شود. این بدین معنی است که تعمیم تکنیک‌ها به ماهیت داده بستگی خواهد داشت. برخی از موضوعات قابل بررسی و مهم در این حوزه به قرار زیر هستند:

- بررسی شباهت میان داده‌های چندرسانه‌ای از مسائلی است که اغلب به آن پرداخته می‌شود. این عمل می‌تواند با دو دیدگاه و یا با دو صورت انجام شود. یک نوع آن بر اساس توصیف داده‌ها مانند کلمات کلیدی، اندازه و زمان ایجاد داده‌های چندرسانه‌ای اجرا می‌شود. دیدگاه دیگر مربوط به سیستم‌هایی است که بر اساس محتوا عمل می‌کنند. الگوهای و شکل‌هایی را که در مجموعه داده‌های چندرسانه‌ای (به خصوص تصاویر) می‌توان یافت، پایه‌ی اصلی معیار شباهت قرار می‌دهند.
- جهت تسهیل در تحلیل و بررسی چندبعدی داده‌های چندرسانه‌ای، مانند دیگر داده‌ها می‌توان انبار داده‌هایی را طراحی نمود. ابعاد این انبار داده‌ها می‌تواند شامل ویژگی‌هایی که در داده‌های چندرسانه‌ای هست، باشد. هر چند این کار مناسب است، اما فراموش نکنید که ساخت یک انبار داده‌ها با ابعاد بسیار زیاد

^۱ Multimedia Mining

آن‌هم برای داده‌های چندرسانه‌ای کار ساده‌ای نیست. بنابراین تحقیقات بسیاری نیاز است تا بر روی چگونگی طراحی یک انبار داده‌های چندرسانه‌ای متتمرکز شود و توازنی میان کارایی روش و قدرت نمایش برقرار کند.

- استفاده از الگوریتم‌های طبقه‌بندی جهت کاوش در داده‌های چندرسانه‌ای نیز می‌تواند برای کاربردهایی نظیر نجوم، زلزله و زمین‌شناسی مناسب باشد. تکنیک‌هایی که در فصل‌های قبلی برای مدل‌سازی و طبقه‌بندی ارائه شدند را می‌توان برای این نوع از داده‌ها نیز استفاده نمود.
- یافتن قوانین انجمنی در گیر میان داده‌های چندرسانه‌ای می‌تواند جالب باشد. برای مثال قانونی شبیه "اگر در قسمت بالای تصویر حداقل ۷۰ درصد رنگ آبی مشاهده شد، این رنگ نشان‌دهندهی آسمان است." و یا اینکه "چنانچه یک تصویر دارای دو مربع آبی هماندازه باشد، به احتمال معینی دارای یک دایره‌ی قرمز نیز هست." برای کاوش قوانین انجمنی، داده‌های چندرسانه‌ای مانند تصاویر تراکنش‌های ما را تشکیل می‌دهند و الگوهای موجود در این داده‌ها اقلام داده‌ی ما هستند.

۵-۳-۱۰) داده‌کاوی مکان محور^۱

یک پایگاه داده‌ی مکان‌محور 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 بتوان آنرا از پایگاه داده‌ی نوع رابطه‌ای تشخیص داد.

امروزه داده‌کاوی این نوع پایگاه داده بطور گسترده‌ای مورد استفاده قرار می‌گیرد. در تکنیک‌های داده‌کاوی مذبور الگوریتم‌ها و روش‌هایی هستند که داده‌هایی دارای مشخصات مکانی را تحلیل می‌کنند. با این کار الگوها و وابستگی‌های مکانی در میان

^۱ Spatial Mining

پدیده‌ها استخراج می‌شوند. در واقع هدف از تحلیل داده‌های مکان‌محور پاسخ دادن به پرسش‌هایی است که در آن مکان و یا وابستگی‌های مکانی مطرح باشند. بر اساس نیازهای اطلاعاتی می‌توان تحلیل‌های مکان‌محور را در پنج نوع زیر جستجو کرد:

- تحلیل الگوی نقطه‌ای^۱
- تحلیل شبکه^۲
- مدل‌سازی مکان‌محور^۳
- تحلیل سطح^۴
- تحلیل شبکه‌ی شترنجی گرید^۵

نقاط، خطوط و چندضلعی‌ها از عناصر اصلی برای نمایش داده‌های مکانی هستند و اطلاعاتی نظیر موقعیت^۶، صفات خاصه‌ی داده‌ها، توپولوژی^۷، مجاورت و همسایگی^۸، محدوده^۹، اتصال^{۱۰} و تقاطع در تحلیل مکان‌محور از مهمترین اطلاعات جهت مکان‌یابی به شمار می‌روند.

مدل‌های داده‌ای گوناگونی پیشنهاد و طراحی شده‌اند که برای نگهداری این نوع از اطلاعات مناسبند. چالش اساسی در کار با این نوع از داده‌ها، طراحی تکنیک‌های کارا جهت داده‌کاوی است. این موضوع با توجه به حجم وسیع داده‌ها، پیچیدگی و همچنین روش‌های دسترسی به داده‌های مکان‌محور، حساسیت بیشتری به خود می‌گیرد. به منظور کسب اطلاعات بیشتر می‌توانید به منابع ذکر شده در پایان فصل رجوع کنید.

¹ Point Pattern Analysis

² Network Analysis

³ Spatial Modeling

⁴ Surface Analysis

⁵ Grid Analysis

⁶ Location

⁷ Topology

⁸ Adjacency

⁹ Containment

¹⁰ Connectivity

خلاصه فصل

به دلیل آنکه داده‌کاوی نسبتاً یک شاخه‌ی جوان با کاربردهای متتنوع را در بر می‌گیرد، هنوز فاصله‌ی زیادی میان مفاهیم و روش‌های داده‌کاوی و ابزارهای موثر تولید شده برای داده‌کاوی وجود دارد. در این میان موسسات و سازمان‌هایی چون بانک‌ها، فروشگاه‌ها و شرکت‌های مخابراتی به دلیل سرویس‌های متعدد و حجم انبوهی از داده‌ها، محیط مطلوبی جهت استفاده از تکنیک‌های داده‌کاوی تلقی می‌شوند. حجم بسیاری از داده‌های جمع‌آوری شده در این موسسات و کمپانی‌ها برای داده‌کاوی کامل و قابل اعتمادند.

کیفیت خوب داده‌ها باعث می‌شود تا تحلیل داده‌ها و عملیات داده‌کاوی تسهیل شوند.

با توجه به حجم بسیار بالای داده‌ها، علم بیوانفورماتیک در صدد برآمده است تا با کمک روش‌هایی چون داده‌کاوی به بررسی کارآمد و مدیریت بهتر اطلاعات مختلف بتواند تجزیه و تحلیل انواع مختلف داده‌ها (مانند توالی‌های *DNA* و پروتئین، تعیین و پیش‌بینی ساختارهای پروتئینی و چارچوب عمل ژن) را به درستی انجام دهد.

اگر چه داده‌کاوی یک حوزه‌ی علمی تقریباً جدید تلقی می‌شود و هنوز کارهای انجام نشده در آن بسیار است، اما می‌توان محصولات و برنامه‌های کاربردی متتنوعی را در این حوزه در بازار پیدا کرد. پشتیبانی از تنوع داده‌ها، امکان اجرا بر روی نرم‌افزار و سخت‌افزارهای مختلف، وجود الگوریتم‌های متتنوع داده‌کاوی، قابلیت مقیاس‌پذیری الگوریتم‌ها و وجود ابزارهای بصری‌سازی مناسب از جمله مواردی هستند که در انتخاب یک محصول تجاری داده‌کاوی بایست به آنها توجه نمود.

امروزه بسته‌های نرم‌افزاری مختلفی را می‌توان یافت که در زمینه‌ی داده‌کاوی امکاناتی را تحت اختیار کاربران قرار می‌دهند. از این میان می‌توان از *Intelligent WEKA*, *MineSet*, *Clementine*, *SQL Server Enterprise Miner*, *Miner*, *RapidMiner* و *Insightful Miner* نام برد.

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 روش‌های جدیدی را نیز می‌طلبد. کاوش در میان داده‌های متñی یا بصورت خلاصه متن‌کاوی، یافتن الگوهای مفید در داده‌های چندساله‌ای، کار بر روی گراف‌ها، جستجو در

داده‌های وب یا وب‌کاوی و استخراج الگوها از میان داده‌های مکانی موضوعاتی هستند که امروزه دغدغه‌ی زیادی را برای محققین ایجاد نموده‌اند.

مراجع و منابع جهت مطالعه‌ی بیشتر

رشد بی‌رویه‌ی داده‌ها در شکل‌ها و ساختارهای پیچیده روش‌های داده‌کاوی را نیز تحت تأثیر خود قرار داده است. چگونگی برخورد تکنیک‌های داده‌کاوی با این نوع از داده‌های خاص در کتاب‌ها و مقالات بسیاری بررسی شده‌اند که در ادامه به بعضی از آنها اشاره‌ای خواهیم داشت.

تحلیل داده‌های متنی به صورت گسترده با شاخه‌ی بازیابی اطلاعات عجین شده است. برای کسب اطلاعات در این زمینه می‌توانید به [BCC10] [CMS09] [Zha08] [MRS08] [GR04] [FS06] [Ber03] [BYRN11] [PL07] [MZ06] [BL09] [WIZD04] و [Liu06] [Cha03a] [Ran03] [Sil10] رجوع کنید.

وب‌کاوی نیز امروزه نظر محققین را به خود جلب نموده و کتاب‌هایی مانند [Ber03] و [Liu06] مراجع مناسبی در این زمینه هستند. همچنین تکنیک‌های وب‌کاوی باعث بهبود موتورهای جستجو در اینترنت شده‌اند. برای کاوش در داده‌های وب منابع [MLSZ06] [CWL⁺08] [CDK⁺99] [KT99] [Kle99] [BP98] و [Sil10] را می‌توانید استفاده کنید.

یافتن الگوهایی در گراف‌ها نیز جنبه‌ای دیگر از استفاده‌ی تکنیک‌های داده‌کاوی با انواع داده‌های است. کتاب‌ها و مجموعه مقالات بسیاری در این زمینه وجود دارند که [HCD94] و [HWB⁺04] [BB02] [YH03a] [KK01] [YH02] [IWM98] و [NK04] از آن جمله هستند. پژوهش‌هایی نیز در زمینه‌ی تحلیل شبکه‌های اطلاعاتی و اجتماعی انجام شده‌اند [Wat03] [YHF10] [WF94] [EK10] [New10] و [Wat03] [FFF99] [AB99] [NBW06]. مدل‌سازی آماری شبکه‌ها نیز در [LKF05] [KRR⁺00] و [GFKT01] [Kle99] [BP98] [CDI98] طبقه‌بندی در شبکه‌ها را نیز می‌توانید

¹ Ranking

[NGE- [SHZ⁺09] [KBDM09] [YHY05] [YHYY04] [NG04] R09 و [JSD⁺10] پیدا کنید. جستجو بر مبنای تشابه و OLAP در شبکه‌های اطلاعاتی مطالب منابع [CYZ⁺08] و [THP08] را تشکیل می‌دهند. مطالب دیگر در مورد شبکه‌های اطلاعاتی و اجتماعی را در [KH09] و [STH⁺10] جستجو کنید. کاوش در میان داده‌های چندرسانه‌ای ارتباط نزدیکی با حوزه‌های پردازش تصویر و تشخیص الگو دارد و به همین دلیل کتب زیادی از جمله [Rus06] [GW07] [DHS01] و [ZZ09] در این باره به رشتۀ تحریر درآمده است. جستجو و کاوش داده‌های چند رسانه‌ای در [NRS99] [FL95] [FS93] و [ZHZ00] بررسی شده‌اند و در [HLZ02] یک مرور بر روی روش‌های کاوش داده‌های تصویری انجام شده است.

کاوش در میان داده‌های مکان محور در مقالات متعددی مثل [MH09] و در کتاب‌های زیادی مانند [SC03] و [HLW07] بررسی شده است. منابع [MCK⁺04] نیز در این زمینه مطالبی را بیان [LHKG07] [TFPL04] [VGK02] در [LDH⁺10] [LHW07] نموده‌اند. کاوش بر روی اشیاء متحرک^۱ در [BJR08] کتاب‌های مختلفی درباره داده‌ی سری‌های زمانی نوشته شده است [SS05] [Ham94] [Cha03b] [BD02] [FRM94] [ALSS95] و [SZ04] جستجو کنید.

داده‌های رشتۀ‌ای^۲ نیز از دسترس روش‌های داده‌کاوی دور نمانند و مدل‌سازی Cube در این نوع داده‌ها در [CDH⁺02]، کاوش الگوهای مکرر در [MM02] و [DH00] [WFYH03] ، طبقه‌بندی داده‌های رشتۀ‌ای در [KPS03] و [GMMO00] و خوشه‌بندی داده‌های رشتۀ‌ای در [AHWY04b] [AHWY03] مطالعه شده‌اند.

¹ Mining Moving Object Data

² Stream Data

برای داده‌کاوی بصری^۱ کتاب‌های رایج [Tuf90] و [Tuf97] [Tuf01] و [SD02] هستند. خلاصه‌ای از تکنیک‌های بصری‌سازی در [Cle93] بررسی شده‌اند و کتاب [FGW01] شامل مجموعه مقالاتی در این زمینه است.

بحث امنیت در داده‌کاوی نیز از حوزه‌هایی است که امروزه با حجم بسیار زیاد داده‌ها و در دسترس بودن آنها مطرح است. کتاب‌های [Thu04] [AY08] [VCZ10] و [VC03] در این زمینه به [ESAG02] [AS00] و مقالات [FWFY10] رشته‌ی تحریر درآمده‌اند.

^۱ Visual Data Mining

ضمیمه الف

منابع و مراجع

- [AAD⁺96] Agarwal, S.; Agrawal, R.; Deshpande, P.M.; Gupta, A.; Naughton, J.F.; Ramakrishnan, R.; Sarawagi, S., On the computation of multidimensional aggregates, In: *Proc. 1996 Int. Conf. Very Large Data Bases (VLDB'96) Bombay, India.* (Sept. 1996), pp. 506–521.
- [AB99] Albert, R.; Barabasi, A.L., Emergence of scaling in random networks, *Science* 286 (1999) 509–512.
- [ABKS99] Ankerst, M.; Breunig, M.; Kriegel, H.-P.; Sander, J., OPTICS: Ordering points to identify the clustering structure, In: *Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'99) Philadelphia, PA.* (June 1999), pp. 49–60.

- [AEMT00] Ahmed, K.M.; El-Makky, N.M.; Taha, Y., A note on “beyond market basket: Generalizing association rules to correlations.”, *SIGKDD Explorations* 1 (2000) 46–48.
- [AGAV09] Amigó, E.; Gonzalo, J.; Artiles, J.; Verdejo, F., A comparison of extrinsic clustering evaluation metrics based on formal constraints, *Information Retrieval* 12 (4) (2009) 461–486.
- [Agg06] Aggarwal, C.C., *Data Streams: Models and Algorithms*. (2006) Kluwer Academic.
- [AGGR98] Agrawal, R.; Gehrke, J.; Gunopulos, D.; Raghavan, P., Automatic subspace clustering of high dimensional data for data mining applications, In: *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98) Seattle, WA*. (June 1998), pp. 94–105.
- [AGS97] Agrawal, R.; Gupta, A.; Sarawagi, S., Modeling multidimensional databases, In: *Proc. 1997 Int. Conf. Data Engineering (ICDE'97) Birmingham, England*. (Apr. 1997), pp. 232–243.
- [Aha92] Aha, D., Tolerating noisy, irrelevant, and novel attributes in instance-based learning algorithms, *Int. J. Man-Machine Studies* 36 (1992) 267–287.
- [AHS96] Arabie, P.; Hubert, L.J.; De Soete, G., *Clustering and Classification*. (1996) World Scientific.
- [AHWY03] Aggarwal, C.C.; Han, J.; Wang, J.; Yu, P.S., A framework for clustering evolving data streams, In: *Proc. 2003 Int. Conf. Very Large Data Bases (VLDB'03) Berlin, Germany*. (Sept. 2003), pp. 81–92.
- [AHWY04a] Aggarwal, C.C.; Han, J.; Wang, J.; Yu, P.S., A framework for projected clustering of high dimensional data streams, In: *Proc. 2004 Int. Conf. Very Large Data Bases (VLDB'04) Toronto, Ontario, Canada*. (Aug. 2004), pp. 852–863.
- [AHWY04b] Aggarwal, C.C.; Han, J.; Wang, J.; Yu, P.S., On demand classification of data streams, In: *Proc. 2004 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'04) Seattle, WA*. (Aug. 2004), pp. 503–508.

-
- [AIS93] Agrawal, R.; Imielinski, T.; Swami, A., Mining association rules between sets of items in large databases, In: *Proc. 1993 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'93) Washington, DC.* (May 1993), pp. 207–216.
- [AK93] Anand, T.; Kahn, G., Opportunity explorer: Navigating large databases using knowledge discovery templates, In: *Proc. AAAI-93 Workshop Knowledge Discovery in Databases Washington, DC.* (July 1993), pp. 45–51.
- [AL99] Aumann, Y.; Lindell, Y., A statistical theory for quantitative association rules, In: *Proc. 1999 Int. Conf. Knowledge Discovery and Data Mining (KDD'99) San Diego, CA.* (Aug. 1999), pp. 261–270.
- [Alp11] Alpaydin, E., *Introduction to Machine Learning*. 2nd ed. (2011) MIT Press, Cambridge, MA.
- [ALSS95] Agrawal, R.; Lin, K.-I.; Sawhney, H.S.; Shim, K., Fast similarity search in the presence of noise, scaling, and translation in time-series databases, In: *Proc. 1995 Int. Conf. Very Large Data Bases (VLDB'95) Zurich, Switzerland.* (Sept. 1995), pp. 490–501.
- [AMS⁺96] Agrawal, R.; Mehta, M.; Shafer, J.; Srikant, R.; Arning, A.; Bollinger, T., The Quest data mining system, In: *Proc. 1996 Int. Conf. Data Mining and Knowledge Discovery (KDD'96) Portland, OR.* (Aug. 1996), pp. 244–249.
- [APW⁺99] Aggarwal, C.C.; Procopiuc, C.; Wolf, J.; Yu, P.S.; Park, J.-S., Fast algorithms for projected clustering, In: *Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'99) Philadelphia, PA.* (June 1999), pp. 61–72.
- [ARV09] Arora, S.; Rao, S.; Vazirani, U., Expander flows, geometric embeddings and graph partitioning, *J. ACM* 56 (2) (2009) 1–37.
- [AS94a] Agrawal, R.; Srikant, R., Fast algorithm for mining association rules in large databases, In: *Research Report RJ 9839* (June 1994) IBM Almaden Research Center, San Jose, CA.
- [AS94b] Agrawal, R.; Srikant, R., Fast algorithms for mining association rules, In: *Proc. 1994 Int. Conf. Very Large Data Bases (VLDB'94) Santiago, Chile.* (Sept. 1994), pp. 487–499.

- [AS95] Agrawal, R.; Srikant, R., Mining sequential patterns, In: *Proc. 1995 Int. Conf. Data Engineering (ICDE'95) Taipei, Taiwan.* (Mar. 1995), pp.3–14.
- [AS96] Agrawal, R.; Shafer, J.C., Parallel mining of association rules: Design, implementation, and experience, *IEEE Trans. Knowledge and Data Engineering* 8 (1996) 962–969.
- [AS00] Agrawal, R.; Srikant, R., Privacy-preserving data mining, In: *Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'00) Dallas, TX.* (May 2000), pp. 439–450.
- [ASS00] Allwein, E.; Shapire, R.; Singer, Y., Reducing multiclass to binary: A unifying approach for margin classifiers, *Journal of Machine Learning Research* 1 (2000) 113–141.
- [AV07] Arthur, D.; Vassilvitskii, S., K-means++: The advantages of careful seeding, In: *Proc. 2007 ACM-SIAM Symp. on Discrete Algorithms (SODA'07) Tokyo.* (2007), pp. 1027–1035.
- [AY99] Aggarwal, C.C.; Yu, P.S., A new framework for itemset generation, In: *Proc. 1998 ACM Symp. Principles of Database Systems (PODS'98) Seattle, WA.* (June 1999), pp. 18–24.
- [AY08] Aggarwal, C.C.; Yu, P.S., *Privacy-Preserving Data Mining: Models and Algorithms.* (2008) Springer, New York.
- [BA97] Breslow, L.A.; Aha, D.W., Simplifying decision trees: A survey, *Knowledge Engineering Rev.* 12 (1997) 1–40.
- [Bay98] Bayardo, R.J., Efficiently mining long patterns from databases, In: *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98) Seattle, WA.* (June 1998), pp. 85–93.
- [BB98] Bagga, A.; Baldwin, B., Entity-based cross-document coreferencing using the vector space model, In: *Proc. 1998 Annual Meeting of the Association for Computational Linguistics and Int. Conf. Computational Linguistics (COLING-ACL'98) Montreal, Quebec, Canada.* (Aug. 1998).
- [BB02] Borgelt, C.; Berthold, M.R., Mining molecular fragments: Finding relevant substructures of molecules, In: *Proc. 2002 Int. Conf. Data Mining (ICDM'02) Maebashi, Japan.* (Dec. 2002), pp. 211–218.
- [BBD⁺02] Babcock, B.; Babu, S.; Datar, M.; Motwani, R.; Widom, J., Models and issues in data stream systems, In: *Proc. 2002*

- ACM Symp. Principles of Database Systems (PODS'02)*
Madison, WI. (June 2002), pp.1–16.
- [BCC10] Buettcher, S.; Clarke, C.L.A.; Cormack, G.V., *Information Retrieval: Implementing and Evaluating Search Engines*. (2010) MIT Press, Cambridge, MA.
- [BCG01] Burdick, D.; Calimlim, M.; Gehrke., J., MAFIA: A maximal frequent itemset algorithm for transactional databases, In: *Proc. 2001 Int. Conf. Data Engineering (ICDE'01)* Heidelberg, Germany. (Apr. 2001), pp. 443–452.
- [BCP93] Brown, D.E.; Corruble, V.; Pittard, C.L., A comparison of decision tree classifiers with backpropagation neural networks for multimodal classification problems, *Pattern Recognition* 26 (1993) 953–961.
- [BD02] Brockwell, P.J.; Davis, R.A., *Introduction to Time Series and Forecasting*. 2nd ed. (2002) Springer, New York.
- [BDF⁺97] Barbará, D.; DuMouchel, W.; Faloutsos, C.; Haas, P.J.; Hellerstein, J.H.; Ioannidis, Y.; Jagadish, H.V.; Johnson, T.; Ng, R.; Poosala, V.; Ross, K.A.; Servcik, K.C., The New Jersey data reduction report, *Bull. Technical Committee on Data Engineering* 20 (Dec. 1997)3–45.
- [Ber81] Bertin, J., *Graphics and Graphic Information Processing*. (1981) Walter de Gruyter, Berlin.
- [Ber03] Berry, M.W., *Survey of Text Mining: Clustering, Classification, and Retrieval*. (2003) Springer, New York.
- [Bez81] Bezdek, J.C., *Pattern Recognition with Fuzzy Objective Function Algorithms*. (1981) Plenum Press.
- [BFOS84] Breiman, L.; Friedman, J.; Olshen, R.; Stone, C., *Classification and Regression Trees*. (1984) Wadsworth International Group.
- [BFR98] Bradley, P.; Fayyad, U.; Reina, C., Scaling clustering algorithms to large databases, In: *Proc. 1998 Int. Conf. Knowledge Discovery and Data Mining (KDD'98)* New York. (Aug. 1998), pp. 9–15.
- [BGMP03] Bonchi, F.; Giannotti, F.; Mazzanti, A.; Pedreschi, D., ExAnte: Anticipated data reduction in constrained pattern mining, In: *Proc. 7th European Conf. Principles and Practice*

- of Knowledge Discovery in Databases (PKDD'03)*, Vol. 2838/2003 (Sept. 2003) Cavtat-Dubrovnik, Croatia, pp. 59–70.
- [BGV92] Boser, B.; Guyon, I.; Vapnik, V.N., A training algorithm for optimal margin classifiers, In: *Proc. Fifth Annual Workshop on Computational Learning Theory* (1992) ACM Press, San Mateo, CA, pp.144–152.
- [Bis95] Bishop, C.M., *Neural Networks for Pattern Recognition*. (1995) Oxford University Press.
- [Bis06] Bishop, C.M., *Pattern Recognition and Machine Learning*. (2006) Springer, New York.
- [BJR08] Box, G.E.P.; Jenkins, G.M.; Reinsel, G.C., *Time Series Analysis: Forecasting and Control*. 4th ed. (2008) Prentice-Hall.
- [BL99] Berry, M.J.A.; Linoff, G., *Mastering Data Mining: The Art and Science of Customer Relationship Management*. (1999) John Wiley & Sons.
- [BL09] Blei, D.; Lafferty, J., Topic models, In: (Editors: Srivastava, A.; Sahami, M.) *Text Mining: Theory and Applications Taylor and Francis*. (2009).
- [BMS97] Brin, S.; Motwani, R.; Silverstein, C., Beyond market basket: Generalizing association rules to correlations, In: *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'97) Tucson, AZ*. (May 1997), pp. 265–276.
- [BMUT97] Brin, S.; Motwani, R.; Ullman, J.D.; Tsur, S., Dynamic itemset counting and implication rules for market basket analysis, In: *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'97) Tucson, AZ*. (May 1997), pp. 255–264.
- [BP92] Bezdek, J.C.; Pal, S.K., *Fuzzy Models for Pattern Recognition: Methods That Search for Structures in Data*. (1992) IEEE Press.
- [BP98] Brin, S.; Page, L., The anatomy of a large-scale hypertextual web search engine, In: *Proc. 7th Int. World Wide Web Conf. (WWW'98) Brisbane, Australia*. (Apr. 1998), pp. 107–117.
- [BR99] Beyer, K.; Ramakrishnan, R., Bottom-up computation of sparse and iceberg cubes, In: *Proc. 1999 ACM-SIGMOD Int.*

- Conf. Management of Data (SIGMOD'99) Philadelphia, PA.* (June 1999), pp. 359–370.
- [BU95] Brodley, C.E.; Utgoff, P.E., Multivariate decision trees, *Machine Learning* 19 (1995) 45–77.
- [Bun94] Buntine, W.L., Operations for learning with graphical models, *J. Artificial Intelligence Research* 2 (1994) 159–225.
- [Bur98] Burges, C.J.C., A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery* 2 (1998) 121–168.
- [Bre96] Breiman, L., Bagging predictors, *Machine Learning* 24 (1996) 123–140.
- [BYRN11] Baeza-Yates, R.A.; Ribeiro-Neto, B.A., *Modern Information Retrieval*. 2nd ed. (2011) Addison-Wesley, Boston.
- [CCS93] Codd, E.F.; Codd, S.B.; Salley, C.T., Beyond decision support, *Computer World* 27 (30) (July 1993) 5–12.
- [CD97] Chaudhuri, S.; Dayal, U., An overview of data warehousing and OLAP technology, *SIGMOD Record* 26 (1997) 65–74.
- [CDH⁺02] Chen, Y.; Dong, G.; Han, J.; Wah, B.W.; Wang, J., Multidimensional regression analysis of time-series data streams, In: *Proc. 2002 Int. Conf. Very Large Data Bases (VLDB'02) Hong Kong, China*. (Aug. 2002), pp. 323–334.
- [CDI98] Chakrabarti, S.; Dom, B.E.; Indyk, P., Enhanced hypertext classification using hyper-links, In: *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98) Seattle, WA*. (June 1998), pp. 307–318.
- [CDK⁺99] Chakrabarti, S.; Dom, B.E.; Kumar, S.R.; Raghavan, P.; Rajagopalan, S.; Tomkins, A.; Gibson, D.; Kleinberg, J.M., Mining the web's link structure, *COMPUTER* 32 (1999) 60–67.
- [CGC94] Chaturvedi, A.; Green, P.; Carroll, J., k -means, k -medians and k -modes: Special cases of partitioning multiway data, In: *The Classification Society of North America (CSNA) Meeting Presentation Houston, TX*. (1994).
- [CGC01] Chaturvedi, A.; Green, P.; Carroll, J., k -modes clustering, *J. Classification* 18 (2001) 35–55.

-
- [CH92] Cooper, G.; Herskovits, E., A Bayesian method for the induction of probabilistic networks from data, *Machine Learning* 9 (1992) 309–347.
- [CH07] Cook, D.J.; Holder, L.B., *Mining Graph Data*. (2007) John Wiley & Sons.
- [Cha03a] Chakrabarti, S., *Mining the Web: Discovering Knowledge from Hypertext Data*. (2003) Morgan Kaufmann.
- [Cha03b] Chatfield, C., *The Analysis of Time Series: An Introduction*. 6th ed. (2003) Chapman & Hall.
- [CHN⁺96] Cheung, D.W.; Han, J.; Ng, V.; Fu, A.; Fu, Y., A fast distributed algorithm for mining association rules, In: *Proc. 1996 Int. Conf. Parallel and Distributed Information Systems Miami Beach, FL*. (Dec. 1996), pp.31–44.
- [CHNW96] Cheung, D.W.; Han, J.; Ng, V.; Wong, C.Y., Maintenance of discovered association rules in large databases: An incremental updating technique, In: *Proc. 1996 Int. Conf. Data Engineering (ICDE'96) New Orleans, LA*. (Feb. 1996), pp. 106–114.
- [CHY96] Chen, M.S.; Han, J.; Yu., P.S., Data mining: An overview from a database perspective, *IEEE Trans. Knowledge and Data Engineering* 8(1996) 866–883.
- [Cle93] Cleveland, W., *Visualizing Data*. (1993) Hobart Press.
- [CM94] Curram, S.P.; Mingers, J., Neural networks, decision tree induction and discriminant analysis: An empirical comparison, *J. Operational Research Society* 45 (1994) 440–450.
- [CMS09] Croft, B.; Metzler, D.; Strohman, T., *Search Engines: Information Retrieval in Practice*. (2009) Addison-Wesley, Boston.
- [CN89] Clark, P.; Niblett, T., The CN2 induction algorithm, *Machine Learning* 3 (1989) 261–283.
- [Coh95] Cohen, W., Fast effective rule induction, In: *Proc. 1995 Int. Conf. Machine Learning (ICML'95) Tahoe City, CA*. (July 1995), pp. 115–123.
- [CR95] Chauvin, Y.; Rumelhart, D., *Backpropagation: Theory, Architectures, and Applications*. (1995) Lawrence Erlbaum.

-
- [Cra89] Crawford, S.L., Extensions to the CART algorithm, *Int. J. Man-Machine Studies* 31 (Aug. 1989) 197–217.
- [CS-T00] Cristianini, N.; Shawe-Taylor, J., *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. (2000) Cambridge University Press.
- [CSZ06] Chapelle, O.; Schölkopf, B.; Zien, A., *Semi-supervised Learning*. (2006) MIT Press, Cambridge, MA.
- [CTTX05] Cong, G.; Tan, K.-Lee; Tung, A.K.H.; Xu, X., Mining top- k covering rule groups for gene expression data, In: *Proc. 2005 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'05) Baltimore, MD*. (June 2005), pp. 670–681.
- [CWL⁺08] Cong, G.; Wang, L.; Lin, C.-Y.; Song, Y.-I.; Sun, Y., Finding question-answer pairs from online forums, In: *Proc. 2008 Int. ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR'08) Singapore*. (July 2008), pp. 467–474.
- [CYHH07] Cheng, H.; Yan, X.; Han, J.; Hsu, C.-W., Discriminative frequent pattern analysis for effective classification, In: *Proc. 2007 Int. Conf. Data Engineering (ICDE'07) Istanbul, Turkey*. (Apr. 2007), pp. 716–725.
- [CYHY08] Cheng, H.; Yan, X.; Han, J.; Yu, P.S., Direct discriminative pattern mining for effective classification, In: *Proc. 2008 Int. Conf. Data Engineering (ICDE'08) Cancun, Mexico*. (Apr. 2008), pp. 169–178.
- [CYZ⁺08] Chen, C.; Yan, X.; Zhu, F.; Han, J.; Yu, P.S., Graph OLAP: Towards online analytical processing on graphs, In: *Proc. 2008 Int. Conf. Data Mining (ICDM'08) Pisa, Italy*. (Dec. 2008), pp. 103–112.
- [CMC05] Cao, H.; Mamoulis, N.; Cheung, D.W., Mining frequent spatio-temporal sequential patterns, In: *Proc. 2005 Int. Conf. Data Mining (ICDM'05) Houston, TX*. (Nov. 2005), pp. 82–89.
- [Dar10] Darwiche, A., Bayesian networks, *Communications of the ACM* 53(2010) 80–90.
- [Das91] Dasarathy, B.V., *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. (1991) IEEE Computer Society Press.

-
- [DB95] Dietterich, T.G.; Bakiri, G., Solving multiclass learning problems via error-correcting output codes, *J. Artificial Intelligence Research* 2 (1995)263–286.
- [DE84] Day, W.H.E.; Edelsbrunner, H., Efficient algorithms for agglomerative hierarchical clustering methods, *J. Classification* 1 (1984)7–24.
- [De01] In: (Editors: Dzeroski, S.; Lavrac, N.) *Relational Data Mining* (2001) Springer, New York.
- [Dev95] Devore, J.L., *Probability and Statistics for Engineering and the Sciences*. 4th ed. (1995) Duxbury Press.
- [DH00] Domingos, P.; Hulten, G., Mining high-speed data streams, In: *Proc. 2000 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'00)* Boston, MA. (Aug. 2000), pp. 71–80.
- [DHS01] Duda, R.O.; Hart, P.E.; Stork, D.G., *Pattern Classification*. 2nd ed. (2001) John Wiley & Sons.
- [DJ03] Dasu, T.; Johnson, T., *Exploratory Data Mining and Data Cleaning*. (2003) John Wiley & Sons.
- [DL97] Dash, M.; Liu, H., Feature selection methods for classification, *Intelligent Data Analysis* 1 (1997) 131–156.
- [DNR⁺97] Deshpande, P.; Naughton, J.; Ramasamy, K.; Shukla, A.; Tufte, K.; Zhao, Y., Cubing algorithms, storage estimation, and storage and processing alternatives for OLAP, *Bull. Technical Committee on Data Engineering* 20 (1997) 3–11.
- [Dob90] Dobson, A.J., *An Introduction to Generalized Linear Models*. (1990) Chapman & Hall.
- [Dom94] Domingos, P., The RISE system: Conquering without separating, In: *Proc. 1994 IEEE Int. Conf. Tools with Artificial Intelligence (TAI'94)* New Orleans, LA. (1994), pp. 704–707.
- [DP96] Domingos, P.; Pazzani, M., Beyond independence: Conditions for the optimality of the simple Bayesian classifier, In: *Proc. 1996 Int. Conf. Machine Learning (ML'96)* Bari, Italy. (July 1996), pp. 105–112.
- [DR05] Davidson, I.; Ravi, S.S., Clustering with constraints: Feasibility issues and the k -means algorithm, In: *Proc. 2005 SIAM Int. Conf. Data Mining (SDM'05)* Newport Beach, CA. (Apr. 2005).

-
- [DT93] Dhar, V.; Tuzhilin, A., Abstract-driven pattern discovery in databases, *IEEE Trans. Knowledge and Data Engineering* 5 (1993)926–938.
- [Dun03] Dunham, M., *Data Mining: Introductory and Advanced Topics*. (2003) Prentice-Hall.
- [DWB06] Davidson, I.; Wagstaff, K.L.; Basu, S., Measuring constraint-set utility for partitional clustering algorithms, In: *Proc. 10th European Conf. Principles and Practice of Knowledge Discovery in Databases (PKDD'06)* Berlin, Germany. (Sept. 2006), pp. 115–126.
- [DYXY07] Dai, W.; Yang, Q.; Xue, G.; Yu, Y., Boosting for transfer learning, In: *Proc. 24th Intl. Conf. Machine Learning Corvallis, OR*. (June 2007), pp. 193–200.
- [Ega75] Egan, J.P., *Signal Detection Theory and ROC Analysis*. (1975) Academic Press.
- [EK10] Easley, D.; Kleinberg, J., *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*. (2010) Cambridge University Press.
- [EKSX96] Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X., A density-based algorithm for discovering clusters in large spatial databases, In: *Proc. 1996 Int. Conf. Knowledge Discovery and Data Mining (KDD'96)* Portland, OR. (Aug. 1996), pp. 226–231.
- [EKX95] Ester, M.; Kriegel, H.-P.; Xu, X., Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification, In: *Proc. 1995 Int. Symp. Large Spatial Databases (SSD'95)* Portland, ME. (Aug. 1995), pp. 67–82.
- [EN10] Elmasri, R.; Navathe, S.B., *Fundamentals of Database Systems*. 6th ed. (2010) Addison-Wesley, Boston.
- [ESAG02] Evfimievski, A.; Srikant, R.; Agrawal, R.; Gehrke, J., Privacy preserving mining of association rules, In: *Proc. 2002 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'02)* Edmonton, Alberta, Canada. (July 2002), pp. 217–228.
- [ET93] Efron, B.; Tibshirani, R., *An Introduction to the Bootstrap*. (1993) Chapman & Hall.

- [FB08] Friedman, J.; Bogdan, E.P., Predictive learning via rule ensembles, *Ann. Applied Statistics* 2 (2008) 916–954.
- [FFF99] Faloutsos, M.; Faloutsos, P.; Faloutsos, C., On power-law relationships of the internet topology, In: *Proc. ACM SIGCOMM'99 Conf. Applications, Technologies, Architectures, and Protocols for Computer Communication Cambridge, MA*. (Aug. 1999), pp. 251–262.
- [FGW01] Fayyad, U.; Grinstein, G.; Wierse, A., *Information Visualization in Data Mining and Knowledge Discovery*. (2001) Morgan Kaufmann.
- [FH51] Fix, E.; Hodges Jr., J.L., Discriminatory analysis, non-parametric discrimination: Consistency properties, In: *Technical Report 21-49-004(4)* (1951) USAF School of Aviation Medicine, Randolph Field, Texas.
- [FI92] Fayyad, U.M.; Irani, K.B., The attribute selection problem in decision tree generation, In: *Proc. 1992 Nat. Conf. Artificial Intelligence (AAAI'92) San Jose, CA*. (1992), pp. 104–110.
- [FI93] Fayyad, U.; Irani, K., Multi-interval discretization of continuous-valued attributes for classification learning, In: *Proc. 1993 Int. Joint Conf. Artificial Intelligence (IJCAI'93) Chambery, France*. (1993), pp. 1022–1029.
- [FL95] Faloutsos, C.; Lin, K.-I., FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets, In: *Proc. 1995 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'95) San Jose, CA*. (May 1995), pp. 163–174.
- [FMMT96] Fukuda, T.; Morimoto, Y.; Morishita, S.; Tokuyama, T., Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization, In: *Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'96) ontreal, Quebec, Canada*. (June 1996), pp. 13–23.
- [FPP07] Freedman, D.; Pisani, R.; Purves, R., *Statistics*. 4th ed. (2007) W. W. Norton & Co. .
- [FR02] Fraley, C.; Raftery, A.E., Model-based clustering, discriminant analysis, and density estimation, *J. American Statistical Association* 97(2002) 611–631.

-
- [FRM94] Faloutsos, C.; Ranganathan, M.; Manolopoulos, Y., Fast subsequence matching in time-series databases, In: *Proc. 1994 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'94 Minneapolis, MN)*. (May 1994), pp. 419–429.
- [FS93] Fayyad, U.; Smyth, P., Image database exploration: Progress and challenges, In: *Proc. AAAI'93 Workshop Knowledge Discovery in Databases (KDD'93) Washington, DC*. (July 1993), pp. 14–27.
- [FS97] Freund, Y.; Schapire, R.E., A decision-theoretic generalization of on-line learning and an application to boosting, *J. Computer and System Sciences* 55 (1997) 119–139.
- [FS06] Feldman, R.; Sanger, J., *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. (2006) Cambridge University Press.
- [FW94] Furnkranz, J.; Widmer, G., Incremental reduced error pruning, In: *Proc. 1994 Int. Conf. Machine Learning (ICML'94) New Brunswick, NJ*. (1994), pp. 70–77.
- [FWFY10] Fung, B.C.M.; Wang, K.; Fu, A.W.-C.; Yu, P.S., *Introduction to Privacy-Preserving Data Publishing: Concepts and Techniques*. (2010) Chapman & Hall/CRC.
- [GFKT01] Getoor, L.; Friedman, N.; Koller, D.; Taskar, B., Learning probabilistic models of relational structure, In: *Proc. 2001 Int. Conf. Machine Learning (ICML'01) Williamstown, MA*. (2001), pp. 170–177.
- [GGR99] Ganti, V.; Gehrke, J.E.; Ramakrishnan, R., CACTUS—clustering categorical data using summaries, In: *Proc. 1999 Int. Conf. Knowledge Discovery and Data Mining (KDD'99) San Diego, CA*. (1999), pp. 73–83.
- [GGRL99] Gehrke, J.; Ganti, V.; Ramakrishnan, R.; Loh, W.-Y., BOAT—optimistic decision tree construction, In: *Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'99) Philadelphia, PA*. (June 1999), pp. 169–180.
- [GKR98] Gibson, D.; Kleinberg, J.M.; Raghavan, P., Clustering categorical data: An approach based on dynamical systems, In: *Proc. 1998 Int. Conf. Very Large Data Bases (VLDB'98) New York, NY*. (Aug. 1998), pp. 311–323.

-
- [GM99] Gupta, A.; Mumick, I.S., *Materialized Views: Techniques, Implementations, and Applications.* (1999) MIT Press, Cambridge, MA.
- [GMMO00] Guha, S.; Mishra, N.; Motwani, R.; O'Callaghan, L., Clustering data streams, In: *Proc. 2000 Symp. Foundations of Computer Science (FOCS'00) Redondo Beach, CA.* (2000), pp. 359–366.
- [GMUW08] Garcia-Molina, H.; Ullman, J.D.; Widom, J., *Database Systems: The Complete Book.* 2nd ed. (2008) Prentice Hall.
- [Gol89] Goldberg, D., *Genetic Algorithms in Search, Optimization, and Machine Learning.* (1989) Addison-Wesley, Reading, MA.
- [GR04] Grossman, D.A.; Frieder, O., *Information Retrieval: Algorithms and Heuristics.* (2004) Springer, New York.
- [GRG98] Gehrke, J.; Ramakrishnan, R.; Ganti, V., RainForest: A framework for fast decision tree construction of large datasets, In: *Proc. 1998 Int. Conf. Very Large Data Bases (VLDB'98) New York, NY.* (Aug. 1998), pp.416–427.
- [GRS98] Guha, S.; Rastogi, R.; Shim, K., CURE: An efficient clustering algorithm for large databases, In: *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98) Seattle, WA.* (June 1998), pp. 73–84.
- [GRS99] Guha, S.; Rastogi, R.; Shim, K., ROCK: A robust clustering algorithm for categorical attributes, In: *Proc. 1999 Int. Conf. Data Engineering (ICDE'99) Sydney, Australia.* (Mar. 1999), pp. 512–521.
- [GW07] Gonzalez, R.C.; Woods, R.E., *Digital Image Processing.* 3rd ed. (2007) Prentice Hall.
- [GZ03] Grahne, G.; Zhu, J., Efficiently using prefix-trees in mining frequent itemsets, In: *Proc. ICDM'03 Int. Workshop on Frequent Itemset Mining Implementations (FIMI'03) Melbourne, FL.* (Nov. 2003).
- [HAC⁺99] Hellerstein, J.M.; Avnur, R.; Chou, A.; Hidber, C.; Olston, C.; Raman, V.; Roth, T.; Haas, P.J., Interactive data analysis: The control project, *IEEE Computer* 32 (1999) 51–59.
- [Ham94] Hamilton, J., *Time Series Analysis.* (1994) Princeton University Press.

-
- [Har75] Hartigan, J.A., *Clustering Algorithms*. (1975) John Wiley & Sons.
- [Hay99] Haykin, S.S., *Neural Networks: A Comprehensive Foundation*. (1999) Prentice-Hall.
- [Hay08] Haykin, S., *Neural Networks and Learning Machines*. (2008) Prentice-Hall.
- [HBV01] Halkidi, M.; Batistakis, Y.; Vazirgiannis, M., On clustering validation techniques, *J. Intelligent Information Systems* 17 (2001)107–145.
- [HCD94] Holder, L.B.; Cook, D.J.; Djoko, S., Substructure discovery in the subdue system, In: *Proc. AAAI'94 Workshop on Knowledge Discovery in Databases (KDD'94) Seattle, WA*. (July 1994), pp. 169–180.
- [Hec96] Heckerman, D., Bayesian networks for knowledge discovery, In: (Editors: Fayyad, U.M.; Piatetsky-Shapiro, G.; Smyth, P.; Uthurusamy, R.)*Advances in Knowledge Discovery and Data Mining* (1996) MIT Press, Cambridge, MA, pp. 273–305.
- [HF94] Han, J.; Fu, Y., Dynamic generation and refinement of concept hierarchies for knowledge discovery in databases, In: *Proc. AAAI'94 Workshop Knowledge Discovery in Databases (KDD'94) Seattle, WA*. (July 1994), pp. 157–168.
- [HF95] Han, J.; Fu, Y., Discovery of multiple-level association rules from large databases, In: *Proc. 1995 Int. Conf. Very Large Data Bases (VLDB'95) Zurich, Switzerland*. (Sept. 1995), pp. 420–431.
- [HG05] Heller, K.A.; Ghahramani, Z., Bayesian hierarchical clustering, In: *Proc. 22nd Int. Conf. Machine Learning (ICML'05) Bonn, Germany*. (2005), pp. 297–304.
- [HG07] Hinneburg, A.; Gabriel, H.-H., DENCLUE 2.0: Fast clustering based on kernel density estimation, In: *Proc. 2007 Int. Conf. Intelligent Data Analysis (IDA'07) Ljubljana, Slovenia*. (2007), pp. 70–80.
- [HGC95] Heckerman, D.; Geiger, D.; Chickering, D.M., Learning Bayesian networks: The combination of knowledge and statistical data, *Machine Learning* 20 (1995) 197–243.

- [HH01] Hilderman, R.J.; Hamilton, H.J., *Knowledge Discovery and Measures of Interest*. (2001) Kluwer Academic.
- [HHW97] Hellerstein, J.; Haas, P.; Wang, H., Online aggregation, In: *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'97) Tucson, AZ*. (May 1997), pp. 171–182.
- [HK98] Hinneburg, A.; Keim, D.A., An efficient approach to clustering in large multimedia databases with noise, In: *Proc. 1998 Int. Conf. Knowledge Discovery and Data Mining (KDD'98) New York, NY*. (Aug. 1998), pp. 58–65.
- [HKGTO3] Hadjieleftheriou, M.; Kollios, G.; Gunopulos, D.; Tsotras, V.J., Online discovery of dense areas in spatio-temporal databases, In: *Proc. 2003 Int. Symp. Spatial and Temporal Databases (SSTD'03) Santorini Island, Greece*. (July 2003), pp. 306–324.
- [HKKR99] Höppner, F.; Klawonn, F.; Kruse, R.; Runkler, T., *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition*. (1999) Wiley.
- [HKP91] Hertz, J.; Krogh, A.; Palmer, R.G., *Introduction to the Theory of Neural Computation*. (1991) Addison-Wesley, Reading, MA.
- [HKP11] Han, J.; Kamber, M.; Pei, J., *Data Mining Concepts and Techniques*. 3rd ed. (2011) Morgan Kaufmann.
- [HLW07] Hsu, W.; Lee, M.L.; Wang, J., *Temporal and Spatio-Temporal Data Mining*. (2007) IGI Publishing.
- [HLZ02] Hsu, W.; Lee, M.L.; Zhang, J., Image mining: Trends and developments, *J. Intelligent Information Systems* 19 (2002) 7–23.
- [HMM86] Hong, J.; Mozetic, I.; Michalski, R.S., Incremental learning of attribute-based descriptions from examples, the method and user's guide, In: *Report ISG 85-5, UIUCDCS-F-86-949 Department of Computer Science, University of Illinois at Urbana-Champaign*. (1986).
- [HMS01] Hand, D.J.; Mannila, H.; Smyth, P., *Principles of Data Mining (Adaptive Computation and Machine Learning)*. (2001) MIT Press, Cambridge, MA.
- [HN90] Hecht-Nielsen, R., *Neurocomputing*. (1990) Addison-Wesley, Reading, MA.

-
- [HP07] Hua, M.; Pei, J., Cleaning disguised missing data: A heuristic approach, In: *Proc. 2007 ACM SIGKDD Intl. Conf. Knowledge Discovery and Data Mining (KDD'07) San Jose, CA.* (Aug. 2007), pp. 950–958.
- [HPDW01] Han, J.; Pei, J.; Dong, G.; Wang, K., Efficient computation of iceberg cubes with complex measures, In: *Proc. 2001 ACM-SIGMOD Int. Conf. Management of Data(SIGMOD'01) Santa Barbara, CA.* (May 2001), pp. 1–12.
- [HPY00] Han, J.; Pei, J.; Yin, Y., Mining frequent patterns without candidate generation, In: *Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'00) Dallas, TX.* (May 2000), pp. 1–12.
- [HRU96] Harinarayan, V.; Rajaraman, A.; Ullman, J.D., Implementing data cubes efficiently, In: *Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'96) Montreal, Quebec, Canada.* (June 1996), pp.205–216.
- [HT98] Hastie, T.; Tibshirani, R., Classification by pairwise coupling, *Ann. Statistics* 26 (1998) 451–471.
- [HTF09] Hastie, T.; Tibshirani, R.; Friedman, J., *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. (2009) Springer Verlag .
- [Hua98] Huang, Z., Extensions to the k -means algorithm for clustering large data sets with categorical values, *Data Mining and Knowledge Discovery* 2(1998) 283–304.
- [HWB⁺04] Huan, J.; Wang, W.; Bandyopadhyay, D.; Snoeyink, J.; Prins, J.; Tropsha, A., Mining spatial motifs from protein structure graphs, In: *Proc. 8th Int. Conf. Research in Computational Molecular Biology (RECOMB)San Diego, CA.* (Mar. 2004), pp. 308–315.
- [IGG03] Imhoff, C.; Galembo, N.; Geiger, J.G., *Mastering Data Warehouse Design: Relational and Dimensional Techniques.* (2003) John Wiley & Sons.
- [Inm96] Inmon, W.H., *Building the Data Warehouse.* (1996) John Wiley & Sons.
- [IWM98] Inokuchi, A.; Washio, T.; Motoda, H., An apriori-based algorithm for mining frequent substructures from graph data, In: *Proc. 2000 European Symp. Principles of Data*

- Mining and Knowledge Discovery (PKDD'00) Lyon, France.* (Sept. 1998), pp. 13–23.
- [Jai10] Jain, A.K., Data clustering: 50 years beyond k -means, *Pattern Recognition Lett.* 31 (8) (2010) 651–666.
- [Jam85] James, M., *Classification Algorithms*. (1985) John Wiley & Sons.
- [JD88] Jain, A.K.; Dubes, R.C., *Algorithms for Clustering Data*. (1988) Prentice-Hall.
- [Jen96] Jensen, F.V., *An Introduction to Bayesian Networks*. (1996) Springer Verlag.
- [JL96] John, G.H.; Langley, P., Static versus dynamic sampling for data mining, In: *Proc. 1996 Int. Conf. Knowledge Discovery and Data Mining (KDD'96) Portland, OR*. (Aug. 1996), pp. 367–370.
- [JMF99] Jain, A.K.; Murty, M.N.; Flynn, P.J., Data clustering: A survey, *ACM Computing Surveys* 31 (1999) 264–323.
- [JSD⁺10] Ji, M.; Sun, Y.; Danilevsky, M.; Han, J.; Gao, J., Graph regularized transductive classification on heterogeneous information networks, In: *Proc. 2010 European Conf. Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECMLPKDD'10) Barcelona, Spain*. (Sept. 2010), pp. 570–586.
- [JW92] Johnson, R.A.; Wichern, D.A., *Applied Multivariate Statistical Analysis*. 3rd ed. (1992) Prentice-Hall.
- [JW02] Jeh, G.; Widom, J., SimRank: A measure of structural-context similarity, In: *Proc. 2002 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'02) Edmonton, Alberta, Canada*. (July 2002), pp. 538–543.
- [Kas80] Kass, G.V., An exploratory technique for investigating large quantities of categorical data, *Applied Statistics* 29 (1980) 119–127.
- [KBDM09] Kulis, B.; Basu, S.; Dhillon, I.; Mooney, R., Semi-supervised graph clustering: A kernel approach, *Machine Learning* 74 (2009) 1–22.
- [Kec01] Kecman, V., *Learning and Soft Computing*. (2001) MIT Press, Cambridge, MA.

-
- [Ker92] Kerber, R., ChiMerge: Discretization of numeric attributes, In: *Proc. 1992 Nat. Conf. Artificial Intelligence (AAAI'92) San Jose, CA.* (1992), pp. 123–128.
- [KF09] Koller, D.; Friedman, N., *Probabilistic Graphical Models: Principles and Techniques*. (2009) MIT Press, Cambridge, MA.
- [KH95] Koperski, K.; Han, J., Discovery of spatial association rules in geographic information databases, In: *Proc. 1995 Int. Symp. Large Spatial Databases (SSD'95) Portland, ME.* (Aug. 1995), pp. 47–66.
- [KH97] Kononenko, I.; Hong, S.J., Attribute selection for modeling, *Future Generation Computer Systems* 13 (1997) 181–195.
- [KH09] Kim, M.-S.; Han, J., A particle-and-density based evolutionary clustering method for dynamic networks, In: *Proc. 2009 Int. Conf. Very Large Data Bases (VLDB'09) Lyon, France.* (Aug. 2009).
- [KHC97] Kamber, M.; Han, J.; Chiang, J.Y., Metarule-guided mining of multi-dimensional association rules using data cubes, In: *Proc. 1997 Int. Conf. Knowledge Discovery and Data Mining (KDD'97) Newport Beach, CA.* (Aug. 1997), pp. 207–210.
- [KHK99] Karypis, G.; Han, E.-H.; Kumar, V., CHAMELEON: A hierarchical clustering algorithm using dynamic modeling, *COMPUTER* 32(1999) 68–75.
- [KK01] Kuramochi, M.; Karypis, G., Frequent subgraph discovery, In: *Proc. 2001 Int. Conf. Data Mining (ICDM'01) San Jose, CA.* (Nov. 2001), pp. 313–320.
- [KKZ09] Kriegel, H.-P.; Kroeger, P.; Zimek, A., Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering, *ACM Trans. Knowledge Discovery from Data (TKDD)* 3 (1) (2009) 1–58.
- [Kle99] Kleinberg, J.M., Authoritative sources in a hyperlinked environment, *J. ACM* 46 (1999) 604–632.
- [KLV⁺98] Kennedy, R.L.; Lee, Y.; Van Roy, B.; Reed, C.D.; Lippman, R.P., *Solving Data Mining Problems Through Pattern Recognition*. (1998) Prentice-Hall.

-
- [KM94] Kivinen, J.; Mannila, H., The power of sampling in knowledge discovery, In: *Proc. 13th ACM Symp. Principles of Database Systems Minneapolis, MN.* (May 1994), pp. 77–85.
- [KMN[†]02] Kanungo, T.; Mount, D.M.; Netanyahu, N.S.; Piatko, C.D.; Silverman, R.; Wu, A.Y., An efficient k -means clustering algorithm: Analysis and implementation, *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)* 24 (2002) 881–892.
- [KMR[†]94] Klemettinen, M.; Mannila, H.; Ronkainen, P.; Toivonen, H.; Verkamo, A.I., Finding interesting rules from large sets of discovered association rules, In: *Proc. 3rd Int. Conf. Information and Knowledge Management Gaithersburg, MD.* (Nov. 1994), pp. 401–408.
- [Koh95] Kohavi, R., A study of cross-validation and bootstrap for accuracy estimation and model selection, In: *Proc. 14th Joint Int. Conf. Artificial Intelligence (IJCAI'95)*, Vol. 2 (Aug. 1995) Montreal, Quebec, Canada, pp. 1137–1143.
- [Kon95] Kononenko, I., On biases in estimating multi-valued attributes, In: *Proc. 14th Joint Int. Conf. Artificial Intelligence (IJCAI'95)*, Vol. 2 (Aug. 1995) Montreal, Quebec, Canada, pp. 1034–1040.
- [KPS03] Karp, R.M.; Papadimitriou, C.H.; Shenker, S., A simple algorithm for finding frequent elements in streams and bags, *ACM Trans. Database Systems* 28 (2003) 51–55.
- [KR90] Kaufman, L.; Rousseeuw, P.J., *Finding Groups in Data: An Introduction to Cluster Analysis.* (1990) John Wiley & Sons.
- [KR02] Kimball, R.; Ross, M., *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling.* 2nd ed. (2002) John Wiley & Sons.
- [KRR[†]00] Kumar, R.; Raghavan, P.; Rajagopalan, S.; Sivakumar, D.; Tomkins, A.; Upfal, E., Stochastic models for the web graph, In: *Proc. 2000 IEEE Symp. Foundations of Computer Science (FOCS'00) Redondo Beach, CA.* (Nov. 2000), pp. 57–65.
- [KRTM08] Kimball, R.; Ross, M.; Thorntwaite, W.; Mundy, J., *The Data Warehouse Lifecycle Toolkit.* (2008) John Wiley & Sons, Hoboken, NJ.

-
- [KT99] Kleinberg, J.M.; Tomkins, A., Application of linear algebra in information retrieval and hypertext analysis, In: *Proc. 18th ACM Symp. Principles of Database Systems (PODS'99) Philadelphia, PA.* (May 1999), pp. 185–193.
- [LDH⁺08] Lin, C.X.; Ding, B.; Han, J.; Zhu, F.; Zhao, B., Text cube: Computing IR measures for multidimensional text database analysis, In: *Proc. 2008 Int. Conf. Data Mining (ICDM'08) Pisa, Italy.* (Dec. 2008), pp. 905–910.
- [LDH⁺10] Li, Z.; Ding, B.; Han, J.; Kays, R.; Nye, P., Mining periodic behaviors for moving objects, In: *Proc. 2010 ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD'10) Washington, DC.* (July 2010), pp. 1099–1108.
- [LHC97] Liu, B.; Hsu, W.; Chen, S., Using general impressions to analyze discovered classification rules, In: *Proc. 1997 Int. Conf. Knowledge Discovery and Data Mining (KDD'97) Newport Beach, CA.* (Aug. 1997), pp. 31–36.
- [LHKG07] Li, X.; Han, J.; Kim, S.; Gonzalez, H., Roam: Rule- and motif-based anomaly detection in massive moving object data sets, In: *Proc. 2007 SIAM Int. Conf. Data Mining (SDM'07) Minneapolis, MN.* (Apr. 2007).
- [LHM98] Liu, B.; Hsu, W.; Ma, Y., Integrating classification and association rule mining, In: *Proc. 1998 Int. Conf. Knowledge Discovery and Data Mining (KDD'98) New York.* (Aug. 1998), pp. 80–86.
- [LHP01] Li, W.; Han, J.; Pei, J., CMAR: Accurate and efficient classification based on multiple class-association rules, In: *Proc. 2001 Int. Conf. Data Mining (ICDM'01) San Jose, CA.* (Nov. 2001), pp. 369–376.
- [LHTD02] Liu, H.; Hussain, F.; Tan, C.L.; Dash, M., Discretization: An enabling technique, *Data Mining and Knowledge Discovery* 6 (2002)393–423.
- [LHW07] Lee, J.-G.; Han, J.; Whang, K., Clustering trajectory data, In: *Proc. 2007 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'07) Beijing, China.* (June 2007).
- [Liu06] Liu, B., *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data.* (2006) Springer, New York.

- [LKCH03] Lee, Y.-K.; Kim, W.-Y.; Cai, Y.D.; Han, J., CoMine: Efficient mining of correlated patterns, In: *Proc. 2003 Int. Conf. Data Mining (ICDM'03) Melbourne, FL.* (Nov. 2003), pp. 581–584.
- [LKF05] Leskovec, J.; Kleinberg, J.; Faloutsos, C., Graphs over time: Densification laws, shrinking diameters and possible explanations, In: *Proc. 2005 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'05) Chicago, IL.* (Aug. 2005), pp. 177–187.
- [LLLY03] Liu, G.; Lu, H.; Lou, W.; Yu, J.X., On computing, storing and querying frequent patterns, In: *Proc. 2003 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'03) Washington, DC.* (Aug. 2003), pp. 607–612.
- [Llo57] Lloyd, S.P., Least squares quantization in PCM, *IEEE Trans. Information Theory* 28 (1982) 128–137; (original version: Technical Report, Bell Labs, 1957).
- [LLS00] Lim, T.-S.; Loh, W.-Y.; Shih, Y.-S., A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms, *Machine Learning* 40 (2000) 203–228.
- [LM98a] Liu, H.; Motoda, H., *Feature Selection for Knowledge Discovery and Data Mining.* (1998) Kluwer Academic.
- [LM98b] In: (Editors: Liu, H.; Motoda, H.) *Feature Extraction, Construction, and Selection: A Data Mining Perspective* (1998) Kluwer Academic.
- [LNHP99] Lakshmanan, L.V.S.; Ng, R.; Han, J.; Pang, A., Optimization of constrained frequent set queries with 2-variable constraints, In: *Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'99) Philadelphia, PA.* (June 1999), pp. 157–168.
- [Los01] Loshin, D., *Enterprise Knowledge Management: The Data Quality Approach.* (2001) Morgan Kaufmann.
- [LPWH02] Liu, J.; Pan, Y.; Wang, K.; Han, J., Mining frequent itemsets by opportunistic projection, In: *Proc. 2002 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'02) Edmonton, Alberta, Canada.* (July 2002), pp. 239–248.

-
- [LS95] Liu, H.; Setiono, R., Chi2: Feature selection and discretization of numeric attributes, In: *Proc. 1995 IEEE Int. Conf. Tools with AI (ICTAI'95) Washington, DC.* (Nov. 1995), pp. 388–391.
- [LS97] Loh, W.Y.; Shih, Y.S., Split selection methods for classification trees, *Statistica Sinica* 7 (1997) 815–840.
- [LSW97] Lent, B.; Swami, A.; Widom, J., Clustering association rules, In: *Proc. 1997 Int. Conf. Data Engineering (ICDE'97) Birmingham, England.* (Apr. 1997), pp. 220–231.
- [LV88] Loh, W.Y.; Vanichsetakul, N., Tree-structured classificaiton via generalized discriminant analysis, *J. American Statistical Association* 83(1988) 715–728.
- [MA03] Mitra, S.; Acharya, T., *Data Mining: Multimedia, Soft Computing, and Bioinformatics.* (2003) John Wiley & Sons.
- [Mac67] MacQueen, J., Some methods for classification and analysis of multivariate observations, *Berkeley, CA. Proc. 5th Berkeley Symp. Math. Stat. Prob.* 1 (1967) 281–297.
- [Mag94] Magidson, J., The CHAID approach to segmentation modeling: CHI-squared automatic interaction detection, In: (Editor: Bagozzi, R.P.) *Advanced Methods of Marketing Research* (1994) Blackwell Business, pp.118–159.
- [MAR96] Mehta, M.; Agrawal, R.; Rissanen, J., SLIQ: A fast scalable classifier for data mining, In: *Proc. 1996 Int. Conf. Extending Database Technology (EDBT'96) Avignon, France.* (Mar. 1996), pp. 18–32.
- [Mar09] Marsland, S., *Machine Learning: An Algorithmic Perspective.* (2009) Chapman & Hall/CRC.
- [MB88] McLachlan, G.J.; Basford, K.E., *Mixture Models: Inference and Applications to Clustering.* (1988) John Wiley & Sons.
- [MCK⁺04] Mamoulis, N.; Cao, H.; Kolios, G.; Hadjieleftheriou, M.; Tao, Y.; Cheung, D., Mining, indexing, and querying historical spatiotemporal data, In: *Proc. 2004 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'04) Seattle, WA.* (Aug. 2004), pp. 236–245.
- [Mei03] Meilă, M., Comparing clusterings by the variation of information, In: *Proc. 16th Annual Conf. Computational*

- Learning Theory (COLT'03) Washington, DC.* (Aug. 2003), pp. 173–187.
- [Mei05] Meilă, M., Comparing clusterings: An axiomatic view, In: *Proc. 22nd Int. Conf. Machine Learning (ICML'05) Bonn, Germany.* (2005), pp. 577–584.
- [MFS95] Malerba, D.; Floriana, E.; Semeraro, G., A further comparison of simplification methods for decision tree induction, In: (Editors: Fisher, D.; Lenz, H.) *Learning from Data: AI and Statistics* (1995) Springer Verlag.
- [MH09] Miller, H.; Han, J., *Geographic Data Mining and Knowledge Discovery*. 2nd ed. (2009) Chapman & Hall/CRC.
- [Mic92] Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*. (1992) Springer Verlag.
- [Mit96] Mitchell, M., *An Introduction to Genetic Algorithms*. (1996) MIT Press, Cambridge, MA.
- [Mit97] Mitchell, T.M., *Machine Learning*. (1997) McGraw-Hill.
- [MK91] Manago, M.; Kodratoff, Y., Induction of decision trees from complex structured data, In: (Editors: Piatetsky-Shapiro, G.; Frawley, W.J.) *Knowledge Discovery in Databases* (1991) AAAI/MIT Press, pp.289–306.
- [MLSZ06] Mei, Q.; Liu, C.; Su, H.; Zhai, C., A probabilistic approach to spatiotemporal theme pattern mining on weblogs, In: *Proc. 15th Int. Conf. World Wide Web (WWW'06) Edinburgh, Scotland.* (May 2006), pp.533–542.
- [MM95] Major, J.; Mangano, J., Selecting among rules induced from a hurricane database, *J. Intelligent Information Systems* 4 (1995) 39–52.
- [MM02] Manku, G.; Motwani, R., Approximate frequency counts over data streams, In: *Proc. 2002 Int. Conf. Very Large Data Bases (VLDB'02) Hong Kong, China.* (Aug. 2002), pp. 346–357.
- [MRS08] Manning, C.D.; Raghavan, P.; Schütze, H., *Introduction to Information Retrieval*. (2008) Cambridge University Press.
- [MST94] Michie, D.; Spiegelhalter, D.J.; Taylor, C.C., *Machine Learning, Neural and Statistical Classification*. (1994) Ellis Horwood, Chichester, England.

-
- [MTV94] Mannila, H.; Toivonen, H.; Verkamo, A.I., Efficient algorithms for discovering association rules, In: *Proc. AAAI'94 Workshop Knowledge Discovery in Databases (KDD'94) Seattle, WA.* (July 1994), pp. 181–192.
- [MTV97] Mannila, H.; Toivonen, H.; Verkamo, A.I., Discovery of frequent episodes in event sequences, *Data Mining and Knowledge Discovery* 1(1997) 259–289.
- [Mur98] Murthy, S.K., Automatic construction of decision trees from data: A multi-disciplinary survey, *Data Mining and Knowledge Discovery* 2 (1998) 345–389.
- [MY97] Miller, R.J.; Yang, Y., Association rules over interval data, In: *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'97) Tucson, AZ.* (May 1997), pp. 452–461.
- [MZ06] Mei, Q.; Zhai, C., A mixture model for contextual text mining, In: *Proc. 2006 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'06) Philadelphia, PA.* (Aug. 2006), pp. 649–655.
- [NBW06] Newman, M.; Barabasi, A.-L.; Watts, D.J., *The Structure and Dynamics of Networks.* (2006) Princeton University Press.
- [New10] Newman, M., *Networks: An Introduction.* (2010) Oxford University Press.
- [NG04] Newman, M.E.J.; Girvan, M., Finding and evaluating community structure in networks, *Physical Rev. E* 69 (2004) 113–128.
- [NGE-R09] Neville, J.; Gallaher, B.; Eliassi-Rad, T., Evaluating statistical tests for within-network classifiers of relational data, In: *Proc. 2009 Int. Conf. Data Mining (ICDM'09) Miami, FL.* (Dec. 2009), pp. 397–406.
- [NH94] Ng, R.; Han, J., Efficient and effective clustering method for spatial data mining, In: *Proc. 1994 Int. Conf. Very Large Data Bases (VLDB'94) Santiago, Chile.* (Sept. 1994), pp. 144–155.
- [NK04] Nijssen, S.; Kok, J., A quick start in frequent structure mining can make a difference, In: *Proc. 2004 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'04) Seattle, WA.* (Aug. 2004), pp. 647–652.
- [NKNW96] Neter, J.; Kutner, M.H.; Nachtsheim, C.J.; Wasserman, L., *Applied Linear Statistical Models.* 4th ed. (1996) Irwin.

- [NLHP98] Ng, R.; Lakshmanan, L.V.S.; Han, J.; Pang, A., Exploratory mining and pruning optimizations of constrained associations rules, In: *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98) Seattle, WA.* (June 1998), pp. 13–24.
- [NRS99] Natsev, A.; Rastogi, R.; Shim, K., Walrus: A similarity retrieval algorithm for image databases, In: *Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'99) Philadelphia, PA.* (June 1999), pp. 395–406.
- [Omi03] Omiecinski, E., Alternative interest measures for mining associations, *IEEE Trans. Knowledge and Data Engineering* 15 (2003) 57–69.
- [OMM⁺02] O'Callaghan, L.; Meyerson, A.; Motwani, R.; Mishra, N.; Guha, S., Streaming-data algorithms for high-quality clustering, In: *Proc. 2002 Int. Conf. Data Engineering (ICDE'02) San Francisco, CA.* (Apr. 2002), pp. 685–696.
- [P-SF91] Piatetsky-Shapiro, G.; Frawley, W.J., *Knowledge Discovery in Databases.* (1991) AAAI/MIT Press.
- [PBTL99] Pasquier, N.; Bastide, Y.; Taouil, R.; Lakhal, L., Discovering frequent closed itemsets for association rules, In: *Proc. 7th Int. Conf. Database Theory (ICDT'99) Jerusalem, Israel.* (Jan. 1999), pp. 398–416.
- [Pea88] Pearl, J., *Probabilistic Reasoning in Intelligent Systems.* (1988) Morgan Kaufmann.
- [PCT⁺03] Pan, F.; Cong, G.; Tung, A.K.H.; Yang, J.; Zaki, M., CARPENTER: Finding closed patterns in long biological datasets, In: *Proc. 2003 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'03) Washington, DC.* (Aug. 2003), pp. 637–642.
- [PCY95a] Park, J.S.; Chen, M.S.; Yu, P.S., An effective hash-based algorithm for mining association rules, In: *Proc. 1995 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'95) San Jose, CA.* (May 1995), pp. 175–186.
- [PCY95b] Park, J.S.; Chen, M.S.; Yu, P.S., Efficient parallel mining for association rules, In: *Proc. 4th Int. Conf. Information and Knowledge Management Baltimore, MD.* (Nov. 1995), pp. 31–36.

-
- [PHL01] Pei, J.; Han, J.; Lakshmanan, L.V.S., Mining frequent itemsets with convertible constraints, In: *Proc. 2001 Int. Conf. Data Engineering (ICDE'01) Heidelberg, Germany.* (Apr. 2001), pp. 433–442.
- [PHL04] Parsons, L.; Haque, E.; Liu, H., Subspace clustering for high dimensional data: A review, *SIGKDD Explorations* 6 (2004) 90–105.
- [PHM00] Pei, J.; Han, J.; Mao, R., CLOSET: An efficient algorithm for mining frequent closed itemsets, In: *Proc. 2000 ACM-SIGMOD Int. Workshop Data Mining and Knowledge Discovery (DMKD'00) Dallas, TX.* (May 2000), pp. 11–20.
- [PHM-A⁺01] Pei, J.; Han, J.; Mortazavi-Asl, B.; Pinto, H.; Chen, Q.; Dayal, U.; Hsu, M.-C., PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth, In: *Proc. 2001 Int. Conf. Data Engineering (ICDE'01) Heidelberg, Germany.* (Apr. 2001), pp. 215–224.
- [PHM-A⁺04] Pei, J.; Han, J.; Mortazavi-Asl, B.; Wang, J.; Pinto, H.; Chen, Q.; Dayal, U.; Hsu, M.-C., Mining sequential patterns by pattern-growth: The prefixSpan approach, *IEEE Trans. Knowledge and Data Engineering* 16 (2004) 1424–1440.
- [PKZT01] Papadias, D.; Kalnis, P.; Zhang, J.; Tao, Y., Efficient OLAP operations in spatial data warehouses, In: *Proc. 2001 Int. Symp. Spatial and Temporal Databases (SSTD'01) Redondo Beach, CA.* (July 2001), pp. 443–459.
- [PL07] Pang, B.; Lee, L., Opinion mining and sentiment analysis, *Foundations and Trends in Information Retrieval* 2 (2007) 1–135.
- [PS85] Preparata, F.P.; Shamos, M.I., *Computational Geometry: An Introduction.* (1985) Springer Verlag.
- [PTVF07] Press, W.H.; Teukolosky, S.A.; Vetterling, W.T.; Flannery, B.P., *Numerical Recipes: The Art of Scientific Computing.* (2007) Cambridge University Press, Cambridge.
- [PY10] Pan, S.J.; Yang, Q., A survey on transfer learning, *IEEE Trans. Knowledge and Data Engineering* 22 (2010) 1345–1359.
- [Pyl99] Pyle, D., *Data Preparation for Data Mining.* (1999) Morgan Kaufmann.

- [QR89] Quinlan, J.R.; Rivest, R.L., Inferring decision trees using the minimum description length principle, *Information and Computation* 80(Mar. 1989) 227–248.
- [Qui89] Quinlan, J.R., Unknown attribute values in induction, In: *Proc. 1989 Int. Conf. Machine Learning (ICML '89)* Ithaca, NY. (June 1989), pp.164–168.
- [Qui90] Quinlan, J.R., Learning logic definitions from relations, *Machine Learning* 5 (1990) 139–166.
- [Qui93] Quinlan, J.R., *C4.5: Programs for Machine Learning*. (1993) Morgan Kaufmann.
- [Red01] Redman, T., *Data Quality: The Field Guide*. (2001) Digital Press (Elsevier).
- [RG03] Ramakrishnan, R.; Gehrke, J., *Database Management Systems*. 3rd ed. (2003) McGraw-Hill.
- [RH07] Rosenberg, A.; Hirschberg, J., V-measure: A conditional entropy-based external cluster evaluation measure, In: *Proc. 2007 Joint Conf. Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '07)* Prague, Czech Republic. (June 2007), pp. 410–420.
- [Rip96] Ripley, B.D., *Pattern Recognition and Neural Networks*. (1996) Cambridge University Press.
- [RM86] Rumelhart, D.E.; McClelland, J.L., *Parallel Distributed Processing*. (1986) MIT Press, Cambridge, MA.
- [RMS98] Ramaswamy, S.; Mahajan, S.; Silberschatz, A., On the discovery of interesting patterns in association rules, In: *Proc. 1998 Int. Conf. Very Large Data Bases (VLDB '98)* New York. (Aug. 1998), pp. 368–379.
- [RN95] Russell, S.; Norvig, P., *Artificial Intelligence: A Modern Approach*. (1995) Prentice-Hall.
- [RS97] Ross, K.; Srivastava, D., Fast computation of sparse datacubes, In: *Proc. 1997 Int. Conf. Very Large Data Bases (VLDB '97)* Athens, Greece. (Aug. 1997), pp. 116–125.
- [RS98] Rastogi, R.; Shim, K., Public: A decision tree classifier that integrates building and pruning, In: *Proc. 1998 Int. Conf. Very Large Data Bases (VLDB '98)* New York. (Aug. 1998), pp. 404–415.

-
- [Rus06] Russ, J.C., *The Image Processing Handbook*. 5th ed. (2006) CRC Press.
 - [SA95] Srikant, R.; Agrawal, R., Mining generalized association rules, In:*Proc. 1995 Int. Conf. Very Large Data Bases (VLDB'95)* Zurich, Switzerland. (Sept. 1995), pp. 407–419.
 - [SA96] Srikant, R.; Agrawal, R., Mining sequential patterns: Generalizations and performance improvements, In: *Proc. 5th Int. Conf. Extending Database Technology (EDBT'96)* Avignon, France. (Mar. 1996), pp.3–17.
 - [SAM96] Shafer, J.; Agrawal, R.; Mehta, M., SPRINT: A scalable parallel classifier for data mining, In: *Proc. 1996 Int. Conf. Very Large Data Bases (VLDB'96)* Bombay, India. (Sept. 1996), pp. 544–555.
 - [SC03] Shekhar, S.; Chawla, S., *Spatial Databases: A Tour*. (2003) Prentice-Hall.
 - [Sch07] Schaeffer, S.E., Graph clustering, *Computer Science Rev.* 1 (2007) 27–64.
 - [SD02] Soukup, T.; Davidson, I., *Visual Data Mining: Techniques and Tools for Data Visualization and Mining*. (2002) Wiley.
 - [SE10] Seni, G.; Elder, J.F., *Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions*. (2010) Morgan and Claypool.
 - [Set10] Settles, B., Active learning literature survey, In: *Computer Sciences Technical Report 1648* (2010) University of Wisconsin–Madison.
 - [SF86] Schlimmer, J.C.; Fisher, D., A case study of incremental concept induction, In: *Proc. 1986 Nat. Conf. Artificial Intelligence (AAAI'86)* Philadelphia, PA. (1986), pp. 496–501.
 - [SG92] Smyth, P.; Goodman, R.M., An information theoretic approach to rule induction, *IEEE Trans. Knowledge and Data Engineering* 4 (1992)301–316.
 - [Shi99] Shih, Y.-S., Families of splitting criteria for classification trees, *Statistics and Computing* 9 (1999) 309–315.
 - [SHK00] Stefanovic, N.; Han, J.; Koperski, K., Object-based selective materialization for efficient implementation of spatial data cubes, *IEEE Trans. Knowledge and Data Engineering* 12 (2000) 938–958.

- [Sho97] Shoshani, A., OLAP and statistical databases: Similarities and differences, In: *Proc. 16th ACM Symp. Principles of Database Systems Tucson, AZ.* (May 1997), pp. 185–196.
- [SHZ⁺09] Sun, Y.; Han, J.; Zhao, P.; Yin, Z.; Cheng, H.; Wu, T., RankClus: Integrating clustering with ranking for heterogeneous information network analysis, In: *Proc. 2009 Int. Conf. Extending Data Base Technology (EDBT'09) Saint Petersburg, Russia.* (Mar. 2009), pp. 565–576.
- [Sil10] Silvestri, F., Mining query logs: Turning search usage data into knowledge, *Foundations and Trends in Information Retrieval* 4 (2010)1–174.
- [SK08] Shieh, J.; Keogh, E., iSAX: Indexing and mining terabyte sized time series, In: *Proc. 2008 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'08) Las Vegas, NV.* (Aug. 2008), pp. 623–631.
- [SKS10] Silberschatz, A.; Korth, H.F.; Sudarshan, S., *Database System Concepts*. 6th ed. (2010) McGraw-Hill.
- [SLT⁺01] Shekhar, S.; Lu, C.-T.; Tan, X.; Chawla, S.; Vatsavai, R.R., Map cube: A visualization tool for spatial data warehouses, In: (Editors: Miller, H.J.; Han, J.) *Geographic Data Mining and Knowledge Discovery* (2001) Taylor and Francis, pp. 73–108.
- [SMT91] Shavlik, J.W.; Mooney, R.J.; Towell, G.G., Symbolic and neural learning algorithms: An experimental comparison, *Machine Learning* 6(1991) 111–144.
- [SON95] Savasere, A.; Omiecinski, E.; Navathe, S., An efficient algorithm for mining association rules in large databases, In: *Proc. 1995 Int. Conf. Very Large Data Bases (VLDB'95) Zurich, Switzerland.* (Sept. 1995), pp.432–443.
- [SON98] Savasere, A.; Omiecinski, E.; Navathe, S., Mining for strong negative associations in a large database of customer transactions, In: *Proc. 1998 Int. Conf. Data Engineering (ICDE'98) Orlando, FL.* (Feb. 1998), pp. 494–502.
- [SS88] Siedlecki, W.; Sklansky, J., On automatic feature selection, *Int. J. Pattern Recognition and Artificial Intelligence* 2 (1988) 197–220.

-
- [SS94] Sarawagi, S.; Stonebraker, M., Efficient organization of large multidimensional arrays, In: *Proc. 1994 Int. Conf. Data Engineering (ICDE'94) Houston, TX.* (Feb. 1994), pp. 328–336.
- [SS05] Shumway, R.H.; Stoffer, D.S., *Time Series Analysis and Its Applications.* (2005) Springer, New York.
- [ST96] Silberschatz, A.; Tuzhilin, A., What makes patterns interesting in knowledge discovery systems, *IEEE Trans. Knowledge and Data Engineering* 8 (Dec. 1996) 970–974.
- [STA98] Sarawagi, S.; Thomas, S.; Agrawal, R., Integrating association rule mining with relational database systems: Alternatives and implications, In: *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98) Seattle, WA.* (June 1998), pp. 343–354.
- [STH⁺10] Sun, Y.; Tang, J.; Han, J.; Gupta, M.; Zhao, B., Community evolution detection in dynamic heterogeneous information networks, In: *Proc. 2010 KDD Workshop Mining and Learning with Graphs (MLG'10) Washington, DC.* (July 2010).
- [Sto74] Stone, M., Cross-validatory choice and assessment of statistical predictions, *J. Royal Statistical Society* 36 (1974) 111–147.
- [SVA97] Srikant, R.; Vu, Q.; Agrawal, R., Mining association rules with item constraints, In: *Proc. 1997 Int. Conf. Knowledge Discovery and Data Mining (KDD'97) Newport Beach, CA.* (Aug. 1997), pp. 67–73.
- [Swe88] Swets, J., Measuring the accuracy of diagnostic systems, *Science* 240 (1988) 1285–1293.
- [SZ04] Shasha, D.; Zhu, Y., *High Performance Discovery in Time Series: Techniques and Case Studies.* (2004) Springer, New York.
- [TD02] Tax, D.M.J.; Duin, R.P.W., Using two-class classifiers for multiclass classification, In: *Proc. 16th Intl. Conf. Pattern Recognition (ICPR'2002) Montreal, Quebec, Canada.* (2002), pp. 124–127.
- [TFPL04] Tao, Y.; Faloutsos, C.; Papadias, D.; Liu, B., Prediction and indexing of moving objects with unknown motion

-
- patterns, In: *Proc. 2004 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'04) Paris, France.* (June 2004), pp. 611–622.
- [TG01] Tsoukatos, I.; Gunopulos, D., Efficient mining of spatiotemporal patterns, In: *Proc. 2001 Int. Symp. Spatial and Temporal Databases (SSTD'01) Redondo Beach, CA.* (July 2001), pp. 425–442.
- [THH01] Tung, A.K.H.; Hou, J.; Han, J., Spatial clustering in the presence of obstacles, In: *Proc. 2001 Int. Conf. Data Engineering (ICDE'01) Heidelberg, Germany.* (Apr. 2001), pp. 359–367.
- [THLN01] Tung, A.K.H.; Han, J.; Lakshmanan, L.V.S.; Ng, R.T., Constraint-based clustering in large databases, In: *Proc. 2001 Int. Conf. Database Theory (ICDT'01) London.* (Jan. 2001), pp. 405–419.
- [THP08] Tian, Y.; Hankins, R.A.; Patel, J.M., Efficient aggregation for graph summarization, In: *Proc. 2008 ACM SIGMOD Int. Conf. Management of Data (SIGMOD'08) Vancouver, British Columbia, Canada.* (June 2008), pp. 567–580.
- [Thu04] Thuraisingham, B., Data mining for counterterrorism, In: (Editors: Kargupta, H.; Joshi, A.; Sivakumar, K.; Yesha, Y.) *Data Mining: Next Generation Challenges and Future Directions* (2004) AAAI/MIT Press, pp. 157–183.
- [TK08] Theodoridis, S.; Koutroumbas, K., *Pattern Recognition*. 4th ed. (2008) Academic Press.
- [TKS02] Tan, P.-N.; Kumar, V.; Srivastava, J., Selecting the right interestingness measure for association patterns, In: *Proc. 2002 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'02) Edmonton, Alberta, Canada.* (July 2002), pp. 32–41.
- [Toi96] Toivonen, H., Sampling large databases for association rules, In: *Proc. 1996 Int. Conf. Very Large Data Bases (VLDB'96) Bombay, India.* (Sept. 1996), pp. 134–145.
- [TSK05] Tan, P.N.; Steinbach, M.; Kumar, V., *Introduction to Data Mining*. (2005) Addison-Wesley, Boston.
- [Tuf83] Tufte, E.R., *The Visual Display of Quantitative Information*. (1983) Graphics Press.

-
- [Tuf90] Tufte, E.R., *Envisioning Information*. (1990) Graphics Press.
- [Tuf97] Tufte, E.R., *Visual Explanations: Images and Quantities, Evidence and Narrative*. (1997) Graphics Press.
- [Tuf01] Tufte, E.R., *The Visual Display of Quantitative Information*. 2nd ed. (2001) Graphics Press.
- [UBC97] Utgoff, P.E.; Berkman, N.C.; Clouse, J.A., Decision tree induction based on efficient tree restructuring, *Machine Learning* 29 (1997) 5–44.
- [UFS91] Uthurusamy, R.; Fayyad, U.M.; Spangler, S., Learning useful rules from inconclusive data, In: (Editors: Piatetsky-Shapiro, G.; Frawley, W.J.) *Knowledge Discovery in Databases* (1991) AAAI/MIT Press, pp.141–157.
- [Utg88] Utgoff, P.E., An incremental ID3, In: *Proc. Fifth Int. Conf. Machine Learning (ICML '88) San Mateo, CA.* (1988), pp. 107–120.
- [Vap95] Vapnik, V.N., *The Nature of Statistical Learning Theory*. (1995) Springer Verlag.
- [Vap98] Vapnik, V.N., *Statistical Learning Theory*. (1998) John Wiley & Sons.
- [VC03] Vaidya, J.; Clifton, C., Privacy-preserving k -means clustering over vertically partitioned data, In: *Proc. 2003 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD '03) Washington, DC.* (Aug 2003).
- [VC06] Vuk, M.; Cerk, T., ROC curve, lift chart and calibration plot, *Metodološki zvezki* 3 (2006) 89–108.
- [VCZ10] Vaidya, J.; Clifton, C.W.; Zhu, Y.M., *Privacy Preserving Data Mining*. (2010) Springer, New York.
- [VGK02] Vlachos, M.; Gunopoulos, D.; Kollios, G., Discovering similar multidimensional trajectories, In: *Proc. 2002 Int. Conf. Data Engineering (ICDE '02) San Francisco, CA.* (Apr. 2002), pp. 673–684.
- [VR90] Van Rijsbergen, C.J., *Information Retrieval*. (1990) Butterworth.
- [Wat03] Watts, D.J., *Six Degrees: The Science of a Connected Age*. (2003) W. W. Norton & Company.
- [WCH10] Wu, T.; Chen, Y.; Han, J., Re-examination of interestingness measures in pattern mining: A unified

- framework, *Data Mining and Knowledge Discovery* 21 (3) (2010) 371–397.
- [WCRS01] Wagstaff, K.; Cardie, C.; Rogers, S.; Schrödl, S., Constrained k -means clustering with background knowledge, In: *Proc. 2001 Int. Conf. Machine Learning (ICML'01)* Williamstown, MA. (June 2001), pp.577–584.
- [WF94] Wasserman, S.; Faust, K., *Social Network Analysis: Methods and Applications*. (1994) Cambridge University Press.
- [WF05] Witten, I.H.; Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques*. 2nd ed. (2005) Morgan Kaufmann.
- [WFH11] Witten, I.H.; Frank, E.; Hall, M.A., *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. 3rd ed. (2011) Morgan Kaufmann, Boston.
- [WFYH03] Wang, H.; Fan, W.; Yu, P.S.; Han, J., Mining concept-drifting data streams using ensemble classifiers, In: *Proc. 2003 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'03)* Washington, DC. (Aug. 2003), pp. 226–235.
- [WHH00] Wang, K.; He, Y.; Han, J., Mining frequent itemsets using support constraints, In: *Proc. 2000 Int. Conf. Very Large Data Bases (VLDB'00)* Cairo, Egypt. (Sept. 2000), pp. 43–52.
- [WHP03] Wang, J.; Han, J.; Pei, J., CLOSET+: Searching for the best strategies for mining frequent closed itemsets, In: *Proc. 2003 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'03)* Washington, DC. (Aug. 2003), pp. 236–245.
- [WI98] Weiss, S.M.; Indurkhy, N., *Predictive Data Mining*. (1998) Morgan Kaufmann.
- [WIZD04] Weiss, S.; Indurkhy, N.; Zhang, T.; Damerau, F., *Text Mining: Predictive Methods for Analyzing Unstructured Information*. (2004) Springer, New York.
- [WK91] Weiss, S.M.; Kulikowski, C.A., *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. (1991) Morgan Kaufmann.
- [Wu83] Wu, C.F.J., On the convergence properties of the EM algorithm, *Ann. Statistics* 11 (1983) 95–103.

- [WYM97] Wang, W.; Yang, J.; Muntz, R., STING: A statistical information grid approach to spatial data mining, In: *Proc. 1997 Int. Conf. Very Large Data Bases (VLDB'97) Athens, Greece.* (Aug. 1997), pp. 186–195.
- [XHLW03] Xin, D.; Han, J.; Li, X.; Wah, B.W., Star-cubing: Computing iceberg cubes by top-down and bottom-up integration, In: *Proc. 2003 Int. Conf. Very Large Data Bases (VLDB'03) Berlin, Germany.* (Sept. 2003), pp. 476–487.
- [XSH⁺04] Xiong, H.; Shekhar, S.; Huang, Y.; Kumar, V.; Ma, X.; Yoo, J.S., A framework for discovering co-location patterns in data sets with extended spatial objects, In: *Proc. 2004 SIAM Int. Conf. Data Mining (SDM'04) Lake Buena Vista, FL.* (Apr. 2004).
- [XYFS07] Xu, X.; Yuruk, N.; Feng, Z.; Schweiger, T.A.J., SCAN: A structural clustering algorithm for networks, In: *Proc. 2007 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'07) San Jose, CA.* (Aug. 2007), pp. 824–833.
- [YFM⁺97] Yoda, K.; Fukuda, T.; Morimoto, Y.; Morishita, S.; Tokuyama, T., Computing optimized rectilinear regions for association rules, In: *Proc. 1997 Int. Conf. Knowledge Discovery and Data Mining (KDD'97) Newport Beach, CA.* (Aug. 1997), pp. 96–103.
- [YK09] Ye, L.; Keogh, E., Time series shapelets: A new primitive for data mining, In: *Proc. 2009 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'09) Paris, France.* (June 2009), pp. 947–956.
- [YH02] Yan, X.; Han, J., gSpan: Graph-based substructure pattern mining, In: *Proc. 2002 Int. Conf. Data Mining (ICDM'02) Maebashi, Japan.* (Dec. 2002), pp. 721–724.
- [YH03a] Yan, X.; Han, J., CloseGraph: Mining closed frequent graph patterns, In: *Proc. 2003 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'03) Washington, DC.* (Aug. 2003), pp. 286–295.
- [YH03b] Yin, X.; Han, J., CPAR: Classification based on predictive association rules, In: *Proc. 2003 SIAM Int. Conf. Data Mining (SDM'03) San Francisco, CA.* (May 2003), pp. 331–335.

- [YHF10] Yu, P.S.; Han, J.; Faloutsos, C., *Link Mining: Models, Algorithms and Applications*. (2010) Springer, New York.
- [YHY05] Yin, X.; Han, J.; Yu, P.S., Cross-relational clustering with user's guidance, In: *Proc. 2005 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'05) Chicago, IL*. (Aug. 2005), pp. 344–353.
- [YHYY04] Yin, X.; Han, J.; Yang, J.; Yu, P.S., CrossMine: Efficient classification across multiple database relations, In: *Proc. 2004 Int. Conf. Data Engineering (ICDE'04) Boston, MA*. (Mar. 2004), pp. 399–410.
- [YWY07] Yuan, J.; Wu, Y.; Yang, M., Discovery of collocation patterns: From visual words to visual phrases, In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR'07) Minneapolis, MN*. (June 2007), pp. 1–8.
- [YZ94] Yager, R.R.; Zadeh, L.A., *Fuzzy Sets, Neural Networks and Soft Computing*. (1994) Van Nostrand Reinhold.
- [Zad65] Zadeh, L.A., Fuzzy sets, *Information and Control* 8 (1965) 338–353.
- [Zad83] Zadeh, L., Commonsense knowledge representation based on fuzzy logic, *Computer* 16 (1983) 61–65.
- [Zak00] Zaki, M.J., Scalable algorithms for association mining, *IEEE Trans. Knowledge and Data Engineering* 12 (2000) 372–390.
- [Zak01] Zaki, M., SPADE: An efficient algorithm for mining frequent sequences, *Machine Learning* 40 (2001) 31–60.
- [ZDN97] Zhao, Y.; Deshpande, P.M.; Naughton, J.F., An array-based algorithm for simultaneous multidimensional aggregates, In: *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'97) Tucson, AZ*. (May 1997), pp. 159–170.
- [ZH02] Zaki, M.J.; Hsiao, C.J., CHARM: An efficient algorithm for closed itemset mining, In: *Proc. 2002 SIAM Int. Conf. Data Mining (SDM'02) Arlington, VA*. (Apr. 2002), pp. 457–473.
- [Zha08] Zhai, C., *Statistical Language Models for Information Retrieval*. (2008) Morgan and Claypool.
- [ZHL⁺98] Zaïane, O.R.; Han, J.; Li, Z.N.; Chiang, J.Y.; Chee, S., MultiMedia-Miner: A system prototype for multimedia data mining, In: *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98) Seattle, WA*. (June 1998), pp. 581–583.

-
- [Zhu05] Zhu, X., Semi-supervised learning literature survey, In: *Computer Sciences Technical Report 1530* (2005) University of Wisconsin–Madison.
 - [ZHJ00] Zaïane, O.R.; Han, J.; Zhu, H., Mining recurrent items in multimedia with progressive resolution refinement, In: *Proc. 2000 Int. Conf. Data Engineering (ICDE'00) San Diego, CA.* (Feb. 2000), pp.461–470.
 - [ZPOL97] Zaki, M.J.; Parthasarathy, S.; Ogihara, M.; Li, W., Parallel algorithm for discovery of association rules, *Data Mining and Knowledge Discovery* 1 (1997) 343–374.
 - [ZRL96] Zhang, T.; Ramakrishnan, R.; Livny, M., BIRCH: An efficient data clustering method for very large databases, In: *Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'96) Montreal, Quebec, Canada.* (June 1996), pp. 103–114.
 - [ZYHY07] Zhu, F.; Yan, X.; Han, J.; Yu, P.S., gPrune: A constraint pushing framework for graph pattern mining, In: *Proc. 2007 Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD'07) Nanjing, China.* (May 2007), pp. 388–400.
 - [ZZ09] Zhang, Z.; Zhang, R., *Multimedia Data Mining: A Systematic Introduction to Concepts and Theory.* (2009) Chapman & Hall.
 - [ZZH09] Zhang, D.; Zhai, C.; Han, J., Topic cube: Topic modeling for OLAP on multidimensional text databases, In: *Proc. 2009 SIAM Int. Conf. Data Mining (SDM'09) Sparks, NV.* (Apr. 2009), pp. 1123–1134.