# Лекция 5. Основи на клъстерен анализ

## *5.1. Какво е клъстерен анализ?*

Процесът на разбиване на зададеното множество от физически или абстрактни обекти по групи от *сходни* обекти се нарича *клъстеризация*. Един клъстер представлява колекцията от обекти, които *приличат* един на друг *вътре в клъстера* и се *различават* от обекти от *други клъстери*. Клъстерният анализ е много важна част от човешката дейност. Дори още в ранното детство човекът се учи, например, как да различи котки от кучета или животни от растения чрез постоянно усъвършенстваване на свои подсъзнателни схеми за клъстеризация. В търговията клъстерният анализ може да бъде използван за създаване на индивидуализирани маркетингови стратегии, базиращи се на откриване на различни групи потребители в съществуващите бази данни от клиенти. В биологията той може да се използва за извеждането на таксономии на растения или животни, за категоризация на гени със сходна функционалност и т.н. Клъстеризацията може да подпомогне при идентификация на области от сходни по използване земи от базата данни с аеро или космически фотоснимки, за идентификация на групи сгради в един град съгласно техния тип, стойност и географско разположение. Тя също така може да подпомогне класификация на документи във Web с цел намиране на търсената информация. Като един ИЗД метод, клъстерният анализ може да бъде използван като автономното средство за анализ на разпределение на данни, за визуализация на характеристики на всеки клъстер и за фокусиране върху някой отделен клъстер с цел по-детайлния му анализ. От друга страна, той може да бъде използван като една предварителна стъпка в процеса за подготовката на данни за работа на други алгоритми, например класификация или описание на понятия.

Клъстеризацията на данни е област на непрекъснато развитие. Големи изследвания в тази област се провеждат в такива научни дисциплини като ИЗД, математическата статистика, машинното самообучение, управление на мултимедийни бази данни, биология, астрономия, маркетинг и др.

Като клон на *математическата статистика*, изследванията върху клъстерния анализ от много години се фокусират основно върху *базиран на разстояние (distance-based)* подход към тази задача. Инструментите за клъстерния анализ, използващи такива класически базирани на разстояние алгоритми за клъстеризация като k-means, k-medoids и няколко други метода, са част от големи пакети за статистически анализ, като S-Plus, SPSS и др. В *машинното самообучение* клъстеризацията е примерът на неуправлявано (без учител – unsupervised) обучение. В отличие от класификацията, клъстеризацията и самообучение без учител не използват предварително определени класове и вече класифицирани обучаващи примери. По тази причина клъстеризацията може да се разглежда по-скоро като една форма на *самообучение от наблюдения*, от колко самообучение от примери. При *концептуалната клъстеризация* една група от обекти формира

клъстер само, ако той може да бъде описан като понятие. Това се отличава от традиционната клъстеризация, която измерва сходството на база на геометричното разстояние. Концептуалната клъстеризация се състои от две компоненти: 1) откриване на подходящи клъстери и 2) формиране на описания за всеки клъстер, както при класификацията. И в този случай се прилагат критерии за силно вътре-клъстерното и слабо между-клъстерното сходство.

При ИЗД усилията са фокусирани върху намирането на методи за провеждане на ефективен клъстерен анализ върху *големи бази данни*. Активните направления в изследванията са свързани с въпроси за *разширяемостта (scaleability)* на клъстерните методи, за ефективността на методи за създаване на клъстери със *сложна форма и от различни типове данни*, за клъстерните техники за *данни с много голяма размерност*, както и за методи за клъстеризация на *смесени числови и номинални данни* в големи бази данни.

Клъстеризацията е предмет на интензивни научни изследвания, където нейните потенциални приложения налагат свои специфични изисквания. Методите за клъстеризация, които се разработват и се прилагат в ИЗД, трябва да удовлетворяват на следните изисквания:

- *Разширяемостта.* Съществуват много алгоритми за клъстеризация, които работят много добре върху малки множества от данни, съдържащи не повече от 200 обекта; обаче, една голяма база данни може да съдържа милиони обекти. Клъстеризацията, извършена върху *някоя извадка* от тези данни, може да доведе до изкривени резултати. Необходими са алгоритми за клъстеризация, които могат да работят върху много големи множества от данни.
- *Възможността за работа с различни типове атрибути.* Съществуват множество алгоритми, създадени за клъстеризация на непрекъснати (числови) данни. Обаче, някои приложения могат да изискват клъстеризация на данни от други типове, например двоични, номинални или от всички тези типове заедно.
- *Откриване на клъстери с различна форма.* Повечето от клъстерните алгоритми се базират върху Евклидово или абсолютното разстояние. Такива алгоритми имат тенденция да намират сферичните по форма клъстери с близък размер и плътност (гъстота). Обаче, един клъстер може да има произволна форма, поради това важно е да бъдат разработени алгоритми, способни да намират клъстери с произволна форма.
- *Минимизиране на изисквания към знанията за проблемната област за определяне на входни параметри.* Много от клъстерните алгоритми изискват от потребителя да въвежда определени параметри за желания клъстерен анализ (например броя на клъстери). Резултатите от клъстеризацията могат в значителна степен да зависят от тях. Определянето на подобни параметри често е доста трудна задача, особено за множества от данни, съдържащи обекти с голяма размерност. Това не само затруднява потребители, но и води до проблеми с контрола върху качеството на клъстеризацията.
- *Възможността за работа със зашумени данни.* Повечето от реални бази данни съдържат екстремни или липсващи стойности, както и сгрешени данни. Някои

от клъстерните алгоритми са чувствителни към подобни данни, което води до намаляване на качеството на клъстеризацията.

- *Нечувствителността към реда на постъпване на входни данни.* Някои клъстерни алгоритми са чувствителни към подредба на входни данни; например, едно и също множество от данни, когато е представено на такъв алгоритъм, при различна подредба на данни може да бъде разбито на абсолютно различни клъстери. По тази причина важно е да бъдат разработвани алгоритми нечувствителни към подредбата на данни.
- *Голяма размерност.* Една база данни може да съдържа голямо количество атрибути. Много от клъстерни алгоритми са добри при работата с данни с малка размерност, например с двумерни или тримерни данни. Човешките очи са много добър инструмент за определяне на качеството на клъстеризацията в пространството с до 3 размерности. Предизвикателството е да се клъстеризират обекти в многомерното пространство, особено отчитайки, че подобни данни могат да бъдат силно разпръснати и да имат много асиметрично разпределение в пространството.
- *Клъстеризация базирана на ограничения.* Някои реални приложения могат да изискват извършване на клъстеризацията при наличието на различни видове ограничения. Да предположим, че задачата е да бъдат избрани в града места за разполагане на зададения брой банкомати. За решаването й вие можете да клъстеризирате жилищата с отчитане на такива ограничения като наличие на градски реки, пътища и т.н. Задачата за намиране на групи в данни с добро клъстеризационно поведение, които удовлетворяват зададени ограничения, е една амбициозна изследователска задача.
- *Разбираемостта и използваемостта.* Потребителите очакват резултатите от клъстеризацията да се поддават на интерпретация, да бъдат разбираеми и използваеми. С други думи, често клъстеризацията трябва да бъде свързана със семантичната интерпретация и приложения. Важно е да бъде разбрано, как една приложна цел може да повлияе на избора на конкретния метод за клъстеризация.

## 5.2. Типове данни, използвани при клъстерен анализ

В настоящия раздел ще се запознаем с типове данни, използвани при клъстерния анализ, и начини за тяхната предварителна обработка. Да предположим, че множеството от данни, подлежащи на клъстеризация, съдърже *n* обекта. Алгоритмите за клъстеризация, базирани на използване на основната памет, обикновено използват следващи две структури данни:

- Вече ни позната *матрицата на данни* (или структура *обект-по-атрибут*). Тази структура има форма на релационна таблица или *n x p* матрица:

$$\begin{bmatrix} x_{11} \ldots x_{1f} \ldots x_{1p} \\ . \quad . \quad . \quad . \quad . \\ \mathrm{x}_{\mathrm{n1}} \ldots . x_{nf} \ldots . x_{np} \end{bmatrix} \quad (5.1)$$

- *Матрица на различия* (или структура *обект-по-обект*). Тя съхранява колекция от стойностите за близостта между всички двойки обекти. Често тя се представя чрез *n x n* матрица:

$$\begin{bmatrix} 0 & & & & & \\ d(2,1) & 0 & & & & \\ d(3,1) & d(3,2) & 0 & & & \\ . & . & . & . & . & \\ d(n,1) & d(n,2) & . & . & . & 0 \end{bmatrix} \quad (5.2)$$

където *d(i, j)* е измерената *разлика* или *несходство* между обекти *i* и *j*.

По-долу, ще разгледаме накратко, как се изчислява различието между обекти, описани с атрибути от различни типове данни. Освен вече познати ни типове данни – непрекъснати и номинални, ще разгледаме и такива типове като двоични, атрибути с наредба (ordinal) и пропорционално-скалирани атрибути.

### 5.2.1. Непрекъснати атрибути

*Непрекъснатите* или *интервално-скалирани атрибути* са непрекъснати числови стойности на измервания, направени съгласно една приблизително линейна скала. Типичните примери са тегло, височина, температура и т.н. Трябва да се знае, че използваните мерни единици могат да окажат съществено влияние на резултатите от клъстерния анализ. Например, промяната от метри в инчи при измерване на височина може до доведе до получаване на съвсем различни клъстерни схеми. В общия случай, представянето на един атрибут в по-малки мерни единици води до по-голям диапазон на стойностите на този атрибут и, следователно, до по-голям ефект върху структурата на клъстери. За да избегнем влияние от избора на мерните единици, данните трябва да бъдат *стандартизирани* (макар, че в някои приложения на някои атрибути могат да бъдат целенасочено присвоени по-големи тегла).

За да бъдат стандартизирани непрекъснати атрибути, оригиналните измервания трябва да бъдат превърнати в *безразмерни единици.* В лекцията, посветена на предварителната обработка на данни, ние вече разгледахме някои методи за стандартизация или нормализация на непрекъснатите данни. В клъстерния анализ често се прилага вече познатата ни *z-нормализация,* но с малки изменения:

$$z_{if} = \frac{x_{if} - \hat{x}_f}{s_f} \quad (5.3)$$

където $\hat{x}_f$ е средната стойност на атрибута *f*, а $s_f$ е *средното абсолютно отклонение* на този атрибут, изчислявано по формула:

$$s_f = \frac{1}{n}\sum_{i=1}^{n} | x_{if} - \hat{x}_f | \quad (5.4)$$

Средното абсолютно отклонение $s_f$ е по-устойчиво към екстремните стойности от "традиционното" стандартно отклонение $\sigma_f$ , тъй като при неговото изчисляване отклоненията от средната стойност не се възвеждат на квадрат и, следователно, по този начин се намалява влиянието на екстремните стойности. От другата страна, използването на средното абсолютно отклонение не прави z-значения на екстремните стойности прекалено малки, т.е. тези крайности остават разпознаваеми, което е важно за анализа на крайностите.

Не зависимо от това, дали стойностите на непрекъснати атрибути са стандартизирани или не, различието (или сходство) между два обекта, описвани чрез такива атрибути, се измерва с помощта на някоя *мярка за разстояние*. Както вече ви е известно от курса по машинно самообучение, един общ клас от такива мерки се задава от функцията за *разстояние на Минковски*:

$$d_L(\mathbf{x},\mathbf{y}) = \left( \sum_{k=1}^{p} | x_k - y_k |^L \right)^{\frac{1}{L}}, L \geq 1 \quad (5.5)$$

където **x** и **y** са два *p*-мерни обекта.

Най-често използваните разстояния в клъстерния анализ са *Евклидово* (при *L=2*) и *абсолютното* (при *L=1*). Претегленото разстояние на Минковски се получава когато всеки атрибут има различно тегло $w_f$:

$$d_{Lw}(\mathbf{x},\mathbf{y}) = \left( \sum_{k=1}^{p} w_k | x_k - y_k |^L \right)^{\frac{1}{L}}, \sum_{k=1}^{p} w_k = 1; L \geq 1 \quad (5.6)$$

## 5.2.2. Двоични атрибути

Един двоичен атрибут приема само два състояния: 0 или 1, където 0 означава, че атрибутът не присъства, а 1 – че присъства. Например, 1 за атрибут *Пушач* означава, че човекът пуши, а 0 - че не е.

При изчисляване на различието между двоични атрибути се използват два подхода. Първият от тях се базира на изчисляване на така наречена *таблица на съответствия (contingency table)*. Ако всички двоични атрибути се третират като имащи *еднакво тегло,* то за два обекта – **x** и **y**, описвани с *p* двоични атрибута, може да построи следната 2 x 2 таблица на съответствията:

| | | Обект **y** | | |
|---|---|---|---|---|
| | | *1* | *0* | ***Сума*** |
| Обект **x** | *1* | $f_{11}$ | $f_{10}$ | $f_{11} + f_{10}$ |
| | *0* | $f_{01}$ | $f_{00}$ | $f_{01} + f_{00}$ |
| | ***Сума*** | $f_{11} + f_{01}$ | $f_{10} + f_{00}$ | *p* |

където:

$f_{11}$ е броят на общи атрибути (т.е. имащи стойност 1 и за двата обекта);
$f_{10}$ е броят на атрибути, присъстващи в обект **x**, но липсващи в обекта **y**;
$f_{01}$ е броят на атрибути, липсващи в обект **x**, но присъстващи в обекта **y**;
$f_{00}$ е броят на атрибути, липсващи и в двата обекта.

Общият брой на атрибути е $p = f_{11} + f_{10} + f_{01} + f_{00}$.

Ще наричаме един двоичен атрибут *симетричен*, ако и двете му стойности имат еднакво тегло, т.е. няма никакво значение, кое от двата значения на такъв атрибут да бъде кодирано с 1, а кое с нула. Например, атрибутът *Пол*, приемащ състояния *мъж* или *жена*, е симетричен. Сходството, оценявано на базата на симетрични атрибути, се нарича *инвариантното сходство*, тъй като резултатът не се променя, при промяна на кодиране на някой от атрибутите. Оценката на сходство между два обекта, описвани със симетрични атрибути, най-често се прави чрез *прост коефициент на съвпадение (simple matching coefficient –SMC)* :

$$SMC(\mathbf{x},\mathbf{y}) = \frac{f_{11} + f_{00}}{f_{01} + f_{10} + f_{11} + f_{00}} = \frac{f_{11} + f_{00}}{p} \quad (5.7)$$

Друга, често използвана мярка за различието на симетрични двоични атрибути е *разстоянието по Хеминг* (Hamming distance):

$$d_{Ham}(\mathbf{x},\mathbf{y}) = p - \sum_{i=1}^{p} x_i y_i - \sum_{i=1}^{p} (1-x_i)\cdot(1-y_i), \quad (5.8)$$

където:
$\mathbf{x} = (x_1,...,x_p), \quad x_i \in \{0,1\}$
$\mathbf{y} = (y_1,...,y_p), \quad y_i \in \{0,1\}$

Един двоичен атрибут се нарича *асиметричен*, ако неговите състояния имат различна важност (тегло). Например, *положителен* и *отрицателен* резултат на атрибута *Тест за СПИН* имат съвсем различно тегло! По съглашение, за асиметричните атрибути чрез 1 се кодира състояние, което се среща по-рядко (положителен – за *Тест за СПИН*). Следователно, за асиметрични атрибути положителното съвпадение (съвпадение на 1-ци) е по-важно от отрицателното. По тази причина, асиметричните атрибути се разглеждат не като двоични, а като "единични", т.е. имащи само едно състояние. Сходството, базирано върху асиметричните атрибути, се нарича *неинвариантното сходство*. Най-известна мярка за изчисляване на сходството между асиметрични атрибути е *Жакардов коефициент*:

$$J(\mathbf{x},\mathbf{y}) = \frac{f_{11}}{f_{01} + f_{10} + f_{11}} \quad (5.9)$$

*Скаларното произведение* е един вариант на разстоянието по Хеминг, което често се прилага като мярка за различието в случая на несиметрични атрибути:

$$d_{scalar}(\mathbf{x},\mathbf{y}) = p - \sum_{i=1}^{p} x_i y_i \quad (5.10)$$

Когато атрибутните стойности се кодират само с 0 и 1, само присъстващите признаци (т.е. 1-ци) допринасят за оценяването на сходството между обекти. Обаче, когато стойностите се кодират чрез –1 и +1 (както в невронни мрежи) – несъвпадащите стойности носят "наказателни" точки за тази мярка.

*Пример 5.1*. За да илюстрираме разликата между описаните по-горе метрики за сходство и различие, да ги изчислим за следните два двоични вектора:
**x** = (1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
**y** = (0, 0, 0, 0, 0, 0, 1, 0, 0, 1)

$f_{01} = 2$ броя на атрибути, където **x** има стойност 0, а **y** – 1.
$f_{10} = 1$ броя на атрибути, където **x** има стойност 1, а **y** – 0.
$f_{00} = 7$ броя на атрибути, където **x** има стойност 0 и **y** – 0.
$f_{11} = 0$ броя на атрибути, където **x** има стойност 1 и **y** – 1.

$$SMC(\mathbf{x},\mathbf{y}) = \frac{f_{11} + f_{00}}{f_{01} + f_{10} + f_{11} + f_{00}} = \frac{0+7}{2+1+0+7} = 0.7$$

$$J(\mathbf{x},\mathbf{y}) = \frac{f_{11}}{f_{01} + f_{10} + f_{11}} = \frac{0}{2+1+0} = 0$$

$$d_{Ham}(\mathbf{x},\mathbf{y}) = p - \sum_{i=1}^{p} x_i y_i - \sum_{i=1}^{p} (1-x_i)\cdot(1-y_i) =$$

$$10 - (1\times 0 + 0\times 0 + 0\times 0 + 0\times 0 + 0\times 0 + 0\times 0 + 0\times 1 + 0\times 0 + 0\times 0 + 0\times 1) -$$
$$((1-1)(1-0) + (1-0)(1-0) + (1-0)(1-0) + (1-0)(1-0) + (1-0)(1-0) + (1-0)(1-0) +$$
$$(1-0)(1-1) + (1-0)(1-0) + (1-0)(1-0) + (1-0)(1-1)) = 10 - 0 - 7 = 3$$

$$d_{scalar}(\mathbf{x},\mathbf{y}) = p - \sum_{i=1}^{p} x_i y_i =$$

$$10 - (1\times 0 + 0\times 0 + 0\times 0 + 0\times 0 + 0\times 0 + 0\times 0 + 0\times 1 + 0\times 0 + 0\times 0 + 0\times 1) = 10.$$

### 5.2.3. Определяне на сходство между документи

Текстовите документи често се представят като вектори, в които всеки атрибут представлява честотата на срещане на конкретния терм (дума) в документа. На практика, представянето е по-сложно, тъй като определени общи думи се игнорират, а прилагането на различни техники позволява отчитането на различни форми на една и съща дума, както и на различни дължини на документи и т.н.

Дори документите да съдържат хиляди или десетки хиляди атрибути (терми), всеки документ остава много разпръснат, тъй като има относително малък брой ненулеви атрибути. По този начин сходството между документи (както и в случая с данните за транзакции) не трябва да зависи от броя на общи нулеви стойности, тъй като всяка двойка документи има голяма вероятност „да не съдържа" много едни и същи думи. С други думи, ако броим съвпадения от типа на 0 – 0, то получим, че всеки документ е много сходен с повечето от други документи. По тази причина една мярка за определяне на сходство между документи трябва да игнорира съвпадения от типа на 0 – 0, както това прави, например, Жакардов коефициент, но да е в състояние да работи с не-двоични вектори.

Един от възможни варианти за такава мярка е разширения Жакардов коефициент, известен още като коефициент на Танимото (Tanimoto):

$$EJ(\mathbf{x},\mathbf{y}) = \frac{\mathbf{x} \bullet y}{\|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 - \mathbf{x} \bullet \mathbf{y}} \quad (5.11)$$

където $\mathbf{x} \bullet \mathbf{y} = \sum_{i=1}^{p} x_i y_i$, а $\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{p} x_i^2} = \sqrt{\mathbf{x} \bullet \mathbf{x}}$

По-често използвана мярка за оценяване на сходство между документи е мярката на косинус:

$$\cos(\mathbf{x},\mathbf{y}) = \frac{\mathbf{x} \bullet y}{\|\mathbf{x}\| \ \|\mathbf{y}\|} \quad (5.12)$$

На практика, тази мярка измерва косинус на ъгъла между вектори **x** и **y**, така че тя е 1, когато ъгълът между тези вектори е 0°, т.е. когато **x** и **y** са едни и същи, без да отчитаме тяхната дължина. Ако стойността на мярката е 0, то ъгълът между **x** и **y** е 90° и съответните документи нямат нито една обща дума.

Уравнение (5.12) може да бъде пренаписано в следния еквивалентен вид:

$$\cos(\mathbf{x},\mathbf{y}) = \frac{\mathbf{x} \bullet y}{\|\mathbf{x}\| \ \|\mathbf{y}\|} = \frac{\mathbf{x}}{\|\mathbf{x}\|} \bullet \frac{\mathbf{y}}{\|\mathbf{y}\|} = \mathbf{x'} \bullet \mathbf{y'} \quad (5.13),$$

където $\mathbf{x'} = \mathbf{x}/\|\mathbf{x}\|, \mathbf{y'} = \mathbf{y}/\|\mathbf{y}\|$ - са нормализирани (с дължина 1) вектори. Това означава, че мярката на косинус не отчита размери на сравнявани обекти при изчисляване на тяхното сходство (в отличието от, например, Евклидовото разстояние). За вектори с дължина 1 мярката на косинус е равна на скаларното произведение на векторите, следователно при използването на тази мярка за определяне на сходство между множество от обекти, предварителната нормализация на обекти (т.е. превръщането им в обекти с дължина единица) може значително да намали необходимото време за изчисляване.

### 5.2.4. Отчитане на корелация между атрибути

Корелацията между два обекта, описани с двоични или непрекъснати атрибути е мярка за наличието на линейната зависимост между техните атрибути. Коефициент на Пиарсонова корелация между два обекта данни **x** и **y** се определя по следната формула:

$$cor(\mathbf{x},\mathbf{y}) = \frac{s_{xy}}{s_x s_y} \quad (5.14),$$

където ковариацията $s_{xy}$ и стандартните отклонения $s_x$ и $s_y$ се изчисляват по следните формули:

$$s_{xy} = \frac{1}{p-1}\sum_{i=1}^{p}(x_i - \bar{x})(y_i - \bar{y}),$$

$$s_x = \sqrt{\frac{1}{p-1}\sum_{i=1}^{p}(x_i - \bar{x})^2}, s_y = \sqrt{\frac{1}{p-1}\sum_{i=1}^{p}(y_i - \bar{y})^2} \quad (5.15)$$

$$\bar{x} = \frac{1}{p}\sum_{i=1}^{p}x_i, \bar{y} = \frac{1}{p}\sum_{i=1}^{p}y_i$$

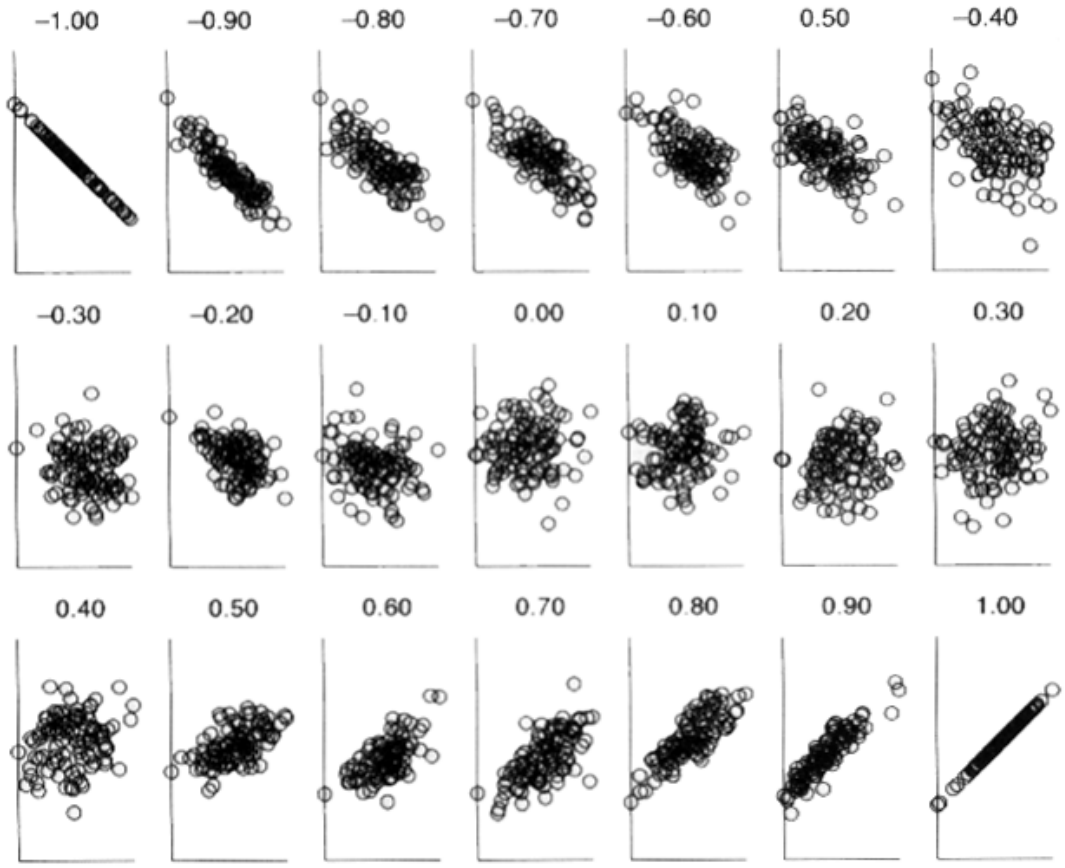
*Пример 5.2 (Перфектна корелация).* Стойността на коефициента на корелация е между -1 и 1, като стойностите 1 (-1) означават, че между **x** и **y** съществува перфектна линейна зависимост, т.е. $x_i = ay_i + b$, където $a$ и $b$ са константи. По-долу са показани две двойки обекти с корелацията -1 и 1 съответно:

**x** = (-3, 6, 0, 3, -6) **x** = (3, 6, 0, 3, 6)
**y** = (1, -2, 0, -1, 2) **y** = (1, 2, 0, 1, 2)

*Пример 5.3 (Нелинейна зависимост).* Стойността на коефициента на корелация равна на 0 означава липсва на линейната зависимост между атрибутите на сравняваните обекти. Обаче, в този случай може да съществува някаква нелинейна зависимост между тях. Например, между показаните по-долу обекти съществува нелинейната зависимост $y_i = x_i^2$, макар че тяхната корелация е равна на 0:

**x** = (-3, -2, -1, 0, 1, 2, 3) **y** = (9, 4, 1, 0, 1, 4, 9)

*Пример 5.3 (Визуализация на корелация).* Наличието на корелацията между двойка обекти **x** и **y** може лесно да се установи визуално чрез графично изобразяване на съответните двойки на атрибутните стойности. На фиг. 5.1 са показани няколко примера на такива графики за двойка обекти **x** и **y,** описвани чрез 30 атрибута. Стойностите на тези атрибути са генерирани по случаен начин (с нормално разпределение) така, че корелацията между **x** и **y** да варира от -1 до 1. Всяко кръгче от един такъв график представлява един от 30 атрибута; неговата $x$ координата отговаря на стойността на съответния атрибут в обекта **x**, а неговата $y$ координата – на стойността на същия атрибут в обекта **y**.




*Фигура 5.1. Графики, илюстриращи корелацията между -1 и 1.*

Повечето от досега определени метрики за изчисляване на разстояние (например Евклидова) отчитали разликата в диапазона на стойностите на атрибутите, но не отчитали възможност за наличието на корелацията между атрибутите. Едно обобщение на Евклидовото разстояние – т.н. *разстояние по Махаланобис* (Mahalanobis distance) е удобно да се ползва в случаите, когато атрибутите са корелират, имат различни диапазони на стойностите (т.е. различна вариабелност) и разпределението на данни е приблизително нормално (Гаусово).

Разстоянието по Махаланобис между два обекта (вектора) **x** и **y** се дефинира като:

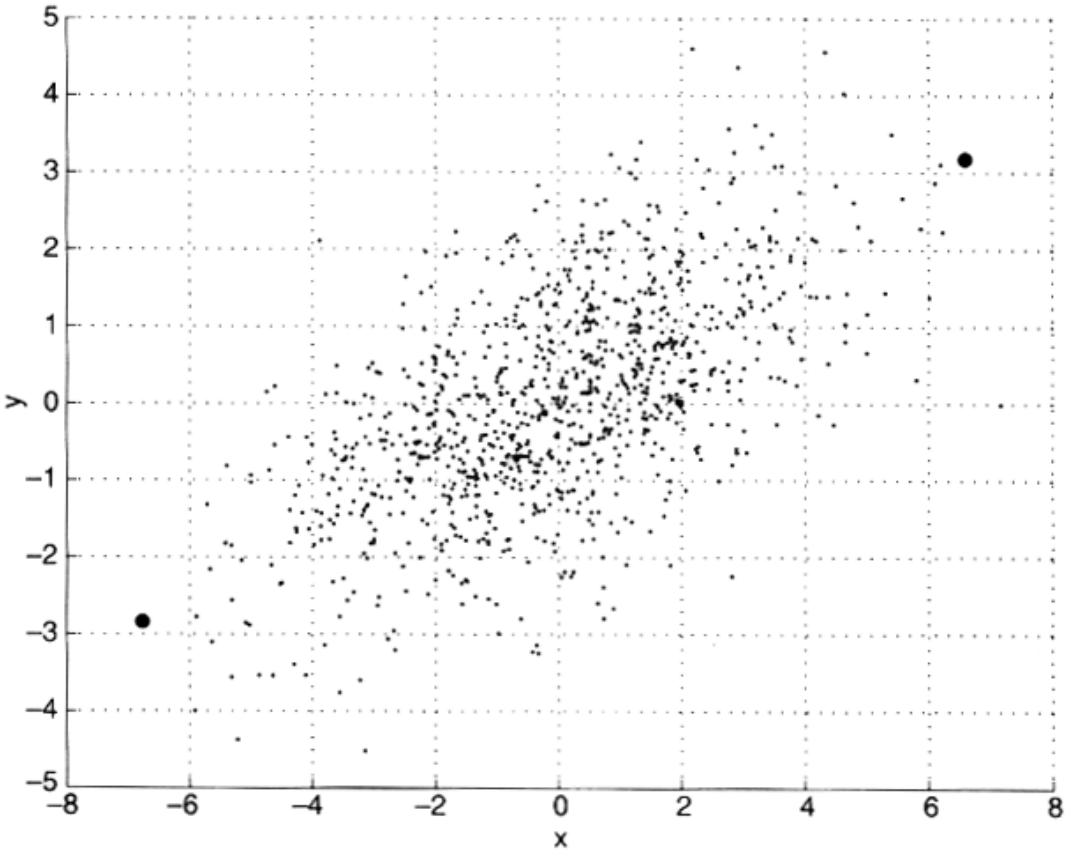
$$d_{Mahal}(\mathbf{x},\mathbf{y}) = (\mathbf{x}-\mathbf{y})\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{y})^{\mathrm{T}} \quad (5.16),$$

където матрицата $\boldsymbol{\Sigma}^{-1}$ е обратната на ковариационната матрица на данните. Ковариационната матрица $\boldsymbol{\Sigma}$ е такава матрица, в която елементът *ij* представлява ковариацията между *i*-я и *j*-я атрибути, изчислен по формула (5.15).

*Пример 5.4.* На фиг. 5.2 са показан график на 1000 двумерни точки, чиито атрибути *x* и *y* имат корелация 0.6. Разстоянието между две големи точки, разположени в два противоположни краища на дългата ос на елипсата е равно на 14.7 единици в

терминитe на Евклидовото разстояние, но само на 6 единици в терминитe на разстоянието по Махаланобис.

Практическото прилагане на разстоянието по Махаланобис и много скъпо от изчислителната гледна точка, но има смисъл когато атрибутите корелират. Ако атрибутите корелират слабо, но имат различни диапазони на стойностите, е достатъчно да се използват процедури по предварителната нормализация на атрибутите.




*Фиг. 5.2. Примерът с 1000 двумерни точки*

### 5.2.3. Номинални атрибути

Номиналният атрибут е едно обобщение на двоичния, при което атрибутът може да приема повече от едно състояние. Типичният пример на номиналния атрибут е *Цвят*, който приема, например, състояния *червен, жълт, зелен, розов* и *син*. Често *за удобство M* състояния на един номинален атрибут се кодират чрез цели числа – от *1* до *M*, като не се предполага *никаква наредба* между тези числови стойности.

Разликата между два обекта, описвани с номинални атрибути, може да се оцени чрез *прост коефициент на съвпадение*:

$$d(\mathbf{x}, \mathbf{y}) = \frac{p - m}{p} \quad (5.17)$$

където $m$ е броят на съвпадения – т.е. броят на атрибути, които приемат една и съща стойност (състояние) и за двата обекта **x** и **y**, а $p$ – общият брой на атрибутите. За да се увеличи ефекта от съвпадение е възможно на съвпадащите стойности да бъдат назначени различни тегла, или да бъдат назначени по-големи тегла на атрибутите, които могат да приемат по-голям брой възможни състояния.

Номиналните атрибути могат да бъдат кодирани като *асиметрични* двоични атрибути чрез прилагане на *процедурата за бинаризация*. При тази процедура всеки номинален атрибут, приемащ $M$ състояния, се заменя с $M$ асиметрични двоични атрибути, приемащи стойност 1 за съответното състояние. Например, значение на номиналния атрибут *Цвят = червен*, се кодира със следните пет двоични атрибута: *Цвят_червен = 1*; *Цвят_жълт = 0; Цвят_зелен = 0; Цвят_розов = 0; Цвят_син = 0.* При използване на подобна бинаризация, различието между обекти, описвани с номинални атрибути, може да бъде оценено чрез вече ни познатия Жакардов коефициент (5.9).

За други по-сложни метрики за оценяване на разликата между номинални атрибути (например MVDM) – вижте раздел за самообучение чрез запомняне, от лекциите по машинно самообучение.

### 5.2.4. Атрибути с наредба

Един *дискретен атрибут с наредба (ordinal)* е такъв номинален атрибут, в който неговите $M$ състояния са подредени в определена, имаща смисъл наредба. Атрибутите с наредба се използват за описание на субективната оценка на такива величини, които не могат да бъдат измерени обективно. Например, професионалните звания често се представят като една наредба, от типа на *асистент, главен асистент, доцент, професор*. *Непрекъснатият атрибут с наредба* представлява един непрекъснат атрибут, измерен по неизвестна скала, т.е. при този атрибут важна не конкретната му стойност, а *относителния порядък* на неговите стойности. Например, относителното ранжиране в спорта (т.е. златен, сребърен, бронзов медал) често по-важно от конкретния резултат. Атрибутите с наредба могат да бъдат получени чрез дискретизацията на непрекъснатия атрибут. Значенията на атрибута с наредба могат да бъдат превърнати в *рангове*. Например, ако един атрибут с наредба $f$ има $M_f$ наредени състояния, то те могат да бъдат превърнати в рангове $1, ..., M_f$.

При изчисляването на различието между обекти, третирането на атрибутите с наредба е почти същото, както и при непрекъснати атрибути. Ако $f$ е атрибут с

наредба от $M_f$ наредени състояния, то различието между два обекта по отношение на този атрибут се изчислява следните 3 стъпки:

1. За всеки обект $i$ стойността $x_{if}$ на атрибута с наредба $f$ се заменя със съответния му ранг: $r_{if} \in \{1,...,M_f\}$.
2. Тъй като различни атрибути с наредба могат да имат различен брой състояния, за да имат всички те едно и също тегло трябва техните стойности да бъдат стандартизирани, например в интервал [0.0, 1.0]. Това може да бъде постигнато чрез замяна на рангове със техни нормализирани стойности:
$$z_{if} = \frac{r_{if} - 1}{M_f - 1} \quad (5.18)$$
3. Различието между обекти се изчислява чрез използване на някоя от мерки за разстояние (5.5) или (5.6), като атрибут $f$ вече се третира като *непрекъснат* със стойностите $z_{if}$.

## 5.2.5. Пропорционално-скалирани атрибути

*Пропорционално-скалираният атрибут* представлява някакво положително измерване, направено по една нелинейна скала, която приблизително може да бъде описана чрез формулата:

$$Ae^{Bt} \text{ или } Ae^{-Bt} \quad (5.19)$$

където $A$ и $B$ са положителни константи. Типичните примери на подобни величини са ръст на популацията на бактерии или периодът за полуразпада на един радиоактивен елемент.

Различието между обекти, описвани чрез пропорционално-скалираните атрибути, може да бъде изчислено по следните три метода:

1. Всички пропорционално-скалирани атрибути да се третират като *обикновени непрекъснати* атрибути. Обаче, това не е добър подход при силно нелинейни скали.
2. Прилагане на *логаритмичната трансформация*: $y_{if} = log(x_{if})$ и третирането трансформираният атрибут като непрекъснат със стойностите $y_{if}$. Могат да бъдат приложени и други трансформации (например, log-log, взимане на корен, степенуване или $y_{if} = \frac{x_{if}}{1 - x_{if}}$ за данни, изразени като пропорция) в зависимост от известна дефиниция на измерваната стойност и конкретното приложение.
3. Третиране на $x_{if}$ като един *непрекъснат атрибут с наредба*, ранжиране на съответните стойности и последващото им третиране като стойностите на непрекъснатия атрибут.

Последните два метода се смятат за най-ефективни, макар че изборът на конкретния метод може да зависи от конкретното приложение.

### 5.2.6. Атрибути от смесени типове

В общия случай базата данни може да съдържа *p* различни атрибута от всички описани по-горе типове. В този случай за получаване на добри резултати от клъстерния анализ е необходимо всички те да бъдат комбинирани в една обща матрица на различия, като всички атрибути да бъдат приведени към една обща скала с диапазон [0.0, 1.0]. Различието между два обекта от такава база данни се дефинира като:

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{f} d^{f}(i,j)}{\sum_{f=1}^{p} \delta_{ij}^{f}} \quad (5.20)$$

където индикатор

$$\delta_{ij}^{f} = \begin{cases} 0, \text{ако } x_{if} \text{ или } x_{jf} \textit{ липсват} \\ 0, \text{ако } x_{if} = x_{jf} \text{ и } f \text{ е асиметричен двоичен атрибут} \\ 1, \text{във всички останали случаи} \end{cases}$$

Атрибутното разстояние $d^{f}(j, j)$ между обекти *i* и *j* се изчислява в зависимост от типа на атрибута *f*:

- Ако *f* е двоичен или номинален атрибут, то:
  $$d^{f}(i,j) = \begin{cases} 0, \text{ако } x_{if} = x_{jf} \\ 1, \text{в противен случай} \end{cases}$$
- Ако *f* е непрекъснат атрибут, то:
  $$d^{f}(i,j) = \frac{|x_{if} - x_{jf}|}{|\max_{k}(x_{kf}) - \min_{k}(x_{kf})|}, \text{k} \in [1, \text{n}]$$
- Ако $f$ е атрибутът с наредба или пропорционално-скалиран, то: изчисли неговите рангове $r_{if}$ и ги нормализирай - $z_{if} = \frac{r_{if} - 1}{M_f - 1}$, след което третирай $z_{if}$ като стойности на непрекъснатия атрибут.

## 5.3. Категоризация на основни методи за клъстъризация

В литература е описано огромно количество различни алгоритми за клъстеризация. Изборът на конкретен алгоритъм зависи както от типа на налични данни, така и от конкретната цел на анализа. Ако клъстерният анализ се използва като инструмент за описание и изследване на данни, то е възможно да се опита няколко различни алгоритъма върху едни и същи данни, за да се види, какви зависимости могат да бъдат намерени.

Съществуващите методи за клъстерния анализ могат да бъдат групирани по следните категории:

- *Методи за разделяне (partitioning)*. При зададена база от данни, съдържаща $n$ обекта, един метод за разделяне разбива данни на $k$ групи – клъстери, като $k \leq n$. Всички групи заедно трябва да удовлетворяват на следните изисквания: 1) всяка група трябва да съдържа най-малко един обект, и 2) всеки обект трябва да принадлежи точно към една група. (Това изискване може да бъде ослабено в някои методи за размито разделяне – за това, ще говорим по-късно в други лекции). При зададения брой на групи $k$, подлежащи на конструиране, методът за разделяне създава някое първоначално разбиване. След това той прилага една избрана *итеративна техниката за преместване* на обекти, опитвайки се да подобри качеството на разбиване чрез преместване на обекти от една група в друга. Общият критерий за качеството на разделянето е "близостта" на обекти, намиращи се в един и същ клъстер, и "отдалечеността" на обекти, намиращи се в различни клъстери. Съществуват различни видове други критерии за оценка на качеството на разделянето.

  За да бъде получено едно *глобално оптимално* решение при клъстеризацията чрез разделяне, е необходимо изчерпващото претърсване на всички възможни разбивания. Вместо това в повечето случаи се използват един от следните два популярни *евристични метода*: 1) алгоритъм *k-means* (т.е. *k-средни*), в който всеки клъстер се представя чрез средното на всички обекти от клъстера, и 2) алгоритъм *k-medoids*, в който клъстерът се представя чрез един от обекти, разположен близо до центъра на клъстера. Тези евристични методи за клъстеризация дават много добри резултати при намиране на сферични по форма клъстери в бази от данни с малки или средни размери. За намирането на клъстери със сложна форма или за клъстеризация на много големи бази данни, методите, базирани на разделяне, се нуждаят от определено разширение.

- *Йерархични методи*. Един йерархичен метод създава йерархичната декомпозиция на зададеното множество от обекти (данни). Йерархичните методи за клъстеризация могат да бъдат разбити на *агломеративни (обединяващи)* и *разделящи (devisive)* в зависимост от начина за формиране на йерархичната декомпозиция. *Агломеративният подход* (който често се нарича подходът *отдолу - нагоре*) започва работата си третирайки всеки обект като отделна група – клъстер. След това той последователно слива групите, които са

"близки" помежду си, докато или всички групи не се слеят в една (най-горното ниво в йерархията), или докато не бъде удовлетворен зададения критерий за прекратяване на сливането. Разделящият подход (който още се нарича подходът *отгоре - надолу*) започва работата си с поместване на всички обекти в един клъстер. След това той итеративно разбива всеки клъстер на по-малки клъстери, докато или всеки обект не бъде поместен в отделен клъстер, или докато не бъде удовлетворен зададения критерий за прекратяване на разделянето.

Основният недостатък на йерархичните методи се състои в това, че след извършването на някоя стъпка (сливане или разделяне), тя *никога не може да бъде преразгледана*, т.е. грешните решения никога не могат да бъдат поправени. Това е следствието от локалния, евристичен метод за търсене, използван, за да бъде избегната комбинаторната експлозия при пълното претърсване на всички възможни избори за намиране на глобалното оптимално решение. Съществуват два подхода за подобряване на качеството на йерархичната клъстеризация: 1) при всяко йерархично разделяне да бъде направен щателен анализ на всички "свързвания" на обекти, както това се прави в такива алгоритми като CURE и Chameleon, или 2) да се интегрира йерархичната агломерация с итеративно преместване на обекти чрез първоначално използване на определен алгоритъм за йерархична агломеративна клъстеризация, а след това – уточняване на резултата чрез прилагане на метода за итеративното преместване, както това е направено в алгоритъма BIRCH.

- *Методите, базирани на плътността.* Повечето от клъстерните методи за разбиване се базират върху изчисляване на разстояние между обектите. Такива методи могат да намират само сферични по форма клъстери и срещат големи затруднения при намирането на клъстери с произволна форма. Съществуват други клъстерни методи, базиращи се на понятие *плътност.* Основната идея е да се продължава разрастване на дадения клъстер до тогава, докато неговата плътност (т.е. брой обекти или точки от данни в него) остава над определен праг. С други думи, за всеки обект, намиращ се в дадения клъстер, неговата околност със зададения радиус трябва да съдържа най-малко определен минимум от обекти. Подобният метод може да се използва за филтриране на шума (крайности), както и за откриване на клъстери с произволна форма.

  Един типичен метод, създаващ клъстери в съответствие с прага на плътността, е алгоритъм DBSCAN. Алгоритъм OPTICS е един базиран на плътността метод, който изчислява определена клъстерна наредба с цел провеждане на автоматичен и интерактивен клъстерен анализ.

- *Решетъчни методи*. Решетъчните методи "дискретизират" цялото пространство от обекти на едно крайно множество от клетки, които формират решетъчната структура. Всички операции по клъстеризация се изпълняват върху тази структура (т.е. върху дискретизираното пространство). Основното предимство на този подход е бързото време за обработка, което обикновено не зависи от

броя на обекти, а само от броя на клетки във всяка от размерности на това дискретизирано пространство.

Типичният пример на такива алгоритми е STING. Алгоритмите CLIQUE и Wave-Cluster обединяват решетъчния подход с използването на плътността.

- *Методи, базирани на модели.* Методите, базирани на модели, предполагат, че всеки клъстер може да бъде описан с помощта на определен модел, и опитват да "нагласят" по най-добрия начин наличните данни към тези модели. Един алгоритъм, базиран на модели, може да локализира клъстерните центрове чрез конструиране на определена функция за вероятностната плътност, описваща пространственото разпределение на обекти. Този подход позволява автоматично да се намира броят на клъстери, базирайки се на стандартните статистики, както и да се определят шум и екстремните обекти.

Някои от клъстерните алгоритми интегрират идеи от различни методи за клъстеризация, което води до определени затруднения с тяхното класифициране по категориите. Освен това, определени приложения могат да имат свои специфични критерии за клъстеризация, което изисква интегриране на няколко клъстеризационни методи.