Market basket analysis

Find joint values of the variables $X = (X_1, ..., X_p)$ that appear most frequently in the data base. It is most often applied to binary-valued data X_j.

- In this context the observations are sales transactions, such as those occurring at the checkout counter of a store. The variables represent all of the items sold in the store. For observation i, each variable X_j is assigned one of two values;

$$x_{ij} = \begin{cases} 1 & \text{If the jth item is purchased in ith transaction} \\ 0 & \textit{Otherwise} \end{cases}$$

- Those variables that frequently have joint values of one represent items that are frequently purchased together. This information can be quite useful for stocking shelves, cross-marketing in sales promotions, catalog design, and consumer segmentation based on buying patterns.

Market basket analysis

Assume $X_1, ..., X_p$ are all binary variables. Market basket analysis aims to find a subset of the integers $\mathcal{K} = \{1, ..., P\}$ so that the following is large:

$$P\left(\prod_{k\in\mathcal{K}}\{X_k = 1\}\right).$$

- The set $\mathcal{K}$ is called an **item set**. The number of items in $\mathcal{K}$ is called its size.
- The above probability is called the **Support or prevalence**, $T(\mathcal{K})$ of the item set $\mathcal{K}$. It is estimated by

$$\hat{P}\left(\prod_{k\in\mathcal{K}}\{X_k = 1\}\right) = \frac{1}{N}\sum_{i=1}^{N}\prod_{i\in\mathcal{K}} x_{ik}.$$

- An observation i for which $\prod_{i\in\mathcal{K}} x_{ik} = 1$ is said to **Contain** the item set $\mathcal{K}$.
- Given a lower bound t, the market basket analysis seeks all the item set $\mathcal{K}_l$ with support in the data base greater than this lower bound t, i.e., $\{\mathcal{K}_l | T(\mathcal{K}_l) > t\}$.

The Apriori algorithm

The solution to the market basket analysis can be obtained with feasible computation for very large data bases provided the threshold t is adjusted so that the solution consists of only a small fraction of all 2^p possible item sets. The **"Apriori" algorithm** (Agrawal et al., 1995) exploits several aspects of the curse of dimensionality to solve the problem with a small number of passes over the data. Specifically, for a given support threshold t:

- The cardinality $|\{\mathcal{K}|T(\mathcal{K}) > t\}|$ is relatively small.
- Any item set $\mathcal{L}$ consisting of a subset of the items in $\mathcal{K}$ must have support greater than or equal to that of $\mathcal{K}$, i.e., if $\mathcal{L} \subset \mathcal{K}$, then $T(\mathcal{L}) > T(\mathcal{K})$

The Apriori algorithm

- The first pass over the data computes the support of all single-item sets. Those whose support is less than the threshold are discarded.
- The second pass computes the support of all item sets of size two that can be formed from pairs of the single items surviving the first pass.
- Each successive pass over the data considers only those item sets that can be formed by combining those that survived the previous pass with those retained from the first pass.
- Passes over the data continue until all candidate rules from the previous pass have support less than the specified threshold
- The Apriori algorithm requires only one pass over the data for each value of $T(\mathcal{K})$, which is crucial since we assume the data cannot be fitted into a computer's main memory. If the data are sufficiently sparse (or if the threshold t is high enough), then the process will terminate in reasonable time even for huge data sets.

Association rule

Each high support item set $\mathcal{K}$ returned by the Apriori algorithm is cast into a set of "association rules." The items $X_k, k \in \mathcal{K}$, are partitioned into two disjoint subsets, $A \cup B = \mathcal{K}$, and written

$$A => B.$$

The first item subset A is called the **"antecedent"** and the second B the **"consequent."**

- The **"support" of the rule** $T(A => B)$ is the support of the item set they are derived.
- The **"confidence" or "predictability"** $C(A => B)$ of the rule is its support divided by the support of the antecedent

$$C(A => B) = \frac{T(A => B)}{T(A)}$$

which can be viewed as an estimate of $P(B|A)$

- The **"expected confidence"** is defined as the support of the consequent $T(B)$, which is an estimate of the unconditional probability $P(B)$.
- The **"lift" of the rule** is defined as the confidence divided by the expected confidence

$$L(A => B) = \frac{C(A => B)}{T(B)}$$

Association rule: example

suppose the item set $\mathcal{K} = \{butter, jelly, bread\}$ and consider the rule $\{peanutbutter, jelly\} => \{bread\}$.

- A **support** value of 0.03 for this rule means that peanut butter, jelly, and bread appeared together in 3% of the market baskets.
- A **confidence** of 0.82 for this rule implies that when peanut butter and jelly were purchased, 82% of the time bread was also purchased.
- If bread appeared in 43% of all market baskets then the rule $\{peanutbutter, jelly\} => \{bread\}$ would have a lift of 1.95.

Association rule

- Sometimes, the desired output of the entire analysis is a collection of association rules that satisfy the constraints

$$T(A => B) > t \text{ and } C(A => B) > c$$

for some threshold t and c. For example,
Display all transactions in which ice skates are the consequent that have confidence over 80% and support of more than 2%.
Efficient algorithms based on the apriori algorithm have been developed.

- Association rules have become a popular tool for analyzing very large commercial data bases in settings where market basket is relevant. That is, when the data can be cast in the form of a multidimensional contingency table. The output is in the form of conjunctive rules that are easily understood and interpreted.

- The Apriori algorithm allows this analysis to be applied to huge data bases, much larger that are amenable to other types of analyses. Association rules are among data mining's biggest successes.
- The number of solution item sets, their size, and the number of passes required over the data can grow exponentially with decreasing size of this lower bound. Thus, rules with high confidence or lift, but low support, will not be discovered. For example, a high confi- dence rule such as *vodka* $=>$ *caviar* will not be uncovered owing to the low sales volume of the consequent *caviar*.

We illustrate the use of Apriori on a moderately sized demographics data base. This data set consists of $N = 9409$ questionnaires filled out by shopping mall customers in the San Francisco Bay Area (Impact Resources, Inc., Columbus OH, 1987). Here we use answers to the first 14 questions, relating to demographics, for illustration.

TABLE 14.1. *Inputs for the demographic data.*

Feature	Demographic	# Values	Type
1	Sex	2	Categorical
2	Marital status	5	Categorical
3	Age	7	Ordinal
4	Education	6	Ordinal
5	Occupation	9	Categorical
6	Income	9	Ordinal
7	Years in Bay Area	5	Ordinal
8	Dual incomes	3	Categorical
9	Number in household	9	Ordinal
10	Number of children	9	Ordinal
11	Householder status	3	Categorical
12	Type of home	5	Categorical
13	Ethnic classification	8	Categorical
14	Language in home	3	Categorical

A freeware implementation of the Apriori algorithm due to Christian Borgelt is used.

- After removing observations with missing values, each ordinal predictor was cut at its median and coded by two dummy variables; each categorical predictor with k categories was coded by k dummy variables.
- This resulted in a 6876 Œ 50 matrix of 6876 observations on 50 dummy variables.
- The algorithm found a total of 6288 association rules, involving ??? 5 predictors, with support of at least 10%. Understanding this large set of rules is itself a challenging data analysis task.

Here are three examples of association rules found by the Apriori algorithm:

- Association rule 1: Support 25%, confidence 99.7% and lift 1.03.

$$\left[\begin{array}{rcl} \text{number in household} & = & 1 \\ \text{number of children} & = & 0 \end{array}\right] \Downarrow \text{language in home} = English$$

- Association rule 2: Support 13.4%, confidence 80.8%, and lift 2.13.

$$\left[\begin{array}{rcl} \text{language in home} & = & \textit{English} \\ \text{householder status} & = & \textit{own} \\ \text{occupation} & = & \{\textit{professional/managerial}\} \end{array}\right] \Downarrow \text{income} \geq \$40{,}000$$

- Association rule 3: Support 26.5%, confidence 82.8% and lift 2.15.

$$\left[\begin{array}{rcl} \text{language in home} & = & \textit{English} \\ \text{income} & < & \textit{\$40,000} \\ \text{marital status} & = & \textit{not married} \\ \text{number of children} & = & \textit{0} \end{array}\right]$$

$$\Downarrow$$

$$\text{education} \notin \{\textit{college graduate}, \textit{graduate study}\}$$

Cluster anlaysis

- Group or segment a collection of objects into subsets or "clusters," such that those within each cluster are more closely related to one another than objects assigned to different clusters.
- Central to all of the goals of cluster analysis is the notion of the degree of similarity (or dissimilarity) between the individual objects being clustered. A clustering method attempts to group the objects based on the definition of similarity supplied to it.
- Definition of similarity can only come from subject matter considerations. The situation is somewhat similar to the specification of a loss or cost function in prediction problems (supervised learning). There the cost associated with an inaccurate prediction depends on considerations outside the data.

Proximity matrices

- Most algorithms presume a matrix of dissimilarities with nonnegative entries and zero diagonal elements: $d_{ii} = 0, i = 1, 2, ..., N$.
- If the original data were collected as similarities, a suitable monotone-decreasing function can be used to convert them to dissimilarities.
- most algorithms assume symmetric dissimilarity matrices, so if the original matrix D is not symmetric it must be replaced by $(D + D^T)/2$

Attribute dissimilarity

For the jth attribute of objects x_{ij} and $x_{i'j}$, let $d_j(x_{ij}, x_{i'j})$ be the dissimilarity between them on the jth attribute.

- **Quantitative variables**: $d_j(x_{ij}, x_{i'j}) = (x_{ij} - x_{i'j})^2$
- **Ordinal variables**: Error measures for ordinal variables are generally defined by replacing their M original values with

$$\frac{i - 1/2}{M}, i = 1, ..., M$$

  in the prescribed order of their original values. They are then treated as quantitative variables on this scale.

- **Nominal variables**:

$$d_j(x_{ij}, x_{ij'}) = \begin{cases} 1 & \text{if } x_{ij} = x_{i'j} \\ 0 & \text{otherwise} \end{cases}$$

Object Dissimilarity

Combining the p-individual attribute dissimilarities $d_j(x_{ij}, x_{i'j}), j = 1, ..., p$ into a single overall measure of dissimilarity $D(x_i, x_{i'})$ between two objects or observations, is usually done through convex combination:

$$D(x_i, x_{i'}) = \sum_{j=1}^{p} w_j d_j(x_{ij}, x_{i'j}), \qquad \sum_{j=1}^{p} w_j = 1.$$

- It is important to realize that setting the weight w_j to the same value for each variable does not necessarily give all attributes equal influence. When the **squared error distance** is used, the relative importance of each variable is proportional to its **variance** over the data.
- With the squared error distance, setting the weight to be the **inverse of the vairance** leads to equal influence of all attributes in the overall dissimilarity between objects.

Standardization in clustering?

- If the goal is to discover natural groupings in the data, some attributes may exhibit more of a grouping tendency than others. Variables that are more relevant in separating the groups should be assigned a higher influence in defining object dissimilarity. Giving all attributes equal influence in this case will tend to obscure the groups to the point where a clustering algorithm cannot uncover them.
- Although simple generic prescriptions for choosing the individual attribute dissimilarities $d_j(x_{ij}, x_{i'j})$ and their weights w_j can be comforting, there is no substitute for careful thought in the context of each individual problem. Specifying an appropriate dissimilarity measure is far more important in obtaining success with clustering than choice of clustering algorithm. This aspect of the problem is emphasized less in the clustering literature than the algorithms themselves, since it depends on domain knowledge specifics and is less amenable to general research.

Standardization in clustering?

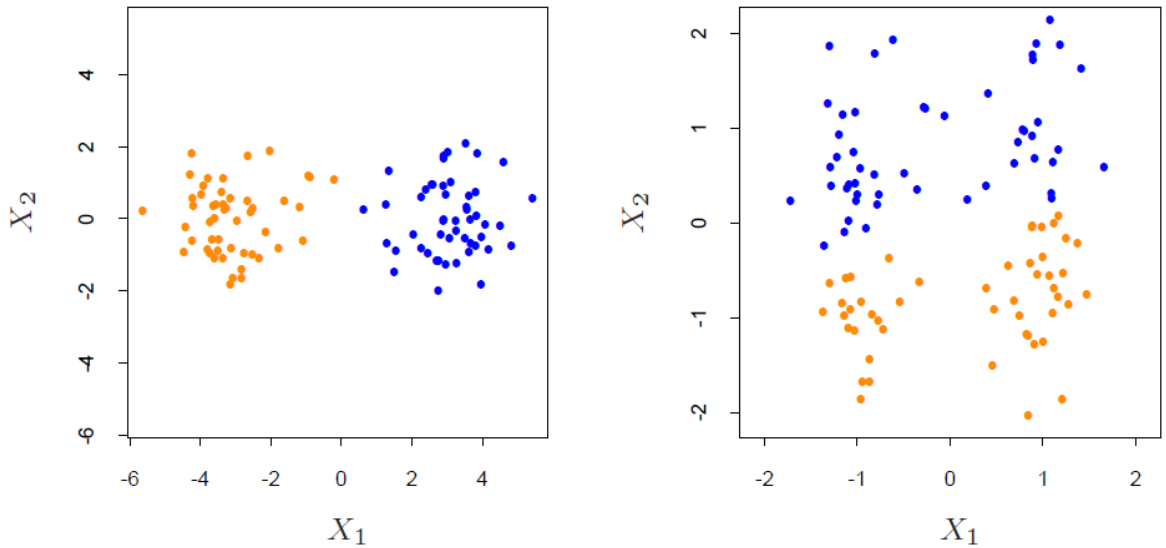

Figure: Simulated data: on the left, K-means clustering (with $K = 2$) has been applied to the raw data. The two colors indicate the cluster memberships. On the right, the features were first standardized before clustering. This is equivalent to using feature weights $1/[2var(X_j)]$. The standardization has obscured the two well-separated groups. Note that each plot uses the same units in the horizontal and vertical axes.

K-means clustering

The K-means algorithm is one of the most popular iterative descent clustering methods. It is intended for situations in which **all variables are of the quantitative type, and squared Euclidean distance is chosen as the dissimilarity measure**. The **within-cluster point scatter** is

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')=k}\|x_i - x_i'\|^2 = \sum_{k=1}^{K} N_k \sum_{C(i)=k}\|x_i - \bar{x}_k\|^2$$

where $\bar{x}_k$ is the mean vector associated with the kth cluster under the clustering rule $C(i)$, and N_k is the number of observations belonging to cluster k.

The K-means clustering algorithm aims find a clustering rule C^* such that

$$C^* = \min_C \sum_{k=1}^{K} \sum_{k=1}^{K} N_k \sum_{C(i)=k} \|x_i - \bar{x}_k\|^2.$$

Notice that

$$\bar{x}_S = \text{argmin}_m \sum_{i \in S} \|x_i - m\|^2$$

Hence we can obtain C^* by solving the enlarged optimization problem

$$\min_{C, m_1, m_2, \ldots, m_K} \sum_{k=1}^{K} N_k \sum_{C(i)=k} \|x_i - m_k\|^2$$

This can be minimized by an alternatiing optimization procedure given in the next algorithm.

Algorithm 14.1 *K-means Clustering.*

1. For a given cluster assignment C, the total cluster variance (14.33) is minimized with respect to $\{m_1, \ldots, m_K\}$ yielding the means of the currently assigned clusters (14.32).

2. Given a current set of means $\{m_1, \ldots, m_K\}$, (14.33) is minimized by assigning each observation to the closest (current) cluster mean. That is,

$$C(i) = \underset{1 \le k \le K}{\operatorname{argmin}} ||x_i - m_k||^2. \tag{14.34}$$

3. Steps 1 and 2 are iterated until the assignments do not change.

- The K-means is guaranteed to converge. However, the result may represent a suboptimal local minimum.
- one should start the algorithm with many different random choices for the starting means, and choose the solution having smallest value of the objective function.
- K-means clustering has shortcomings. For one, it does not give a linear ordering of objects within a cluster: we have simply listed them in alphabetic order above.
- Secondly, as the number of clusters K is changed, the cluster memberships can change in arbitrary ways. That is, with say four clusters, the clusters need not be nested within the three clusters above. For these reasons, hierarchical clustering, is probably preferable for this application.

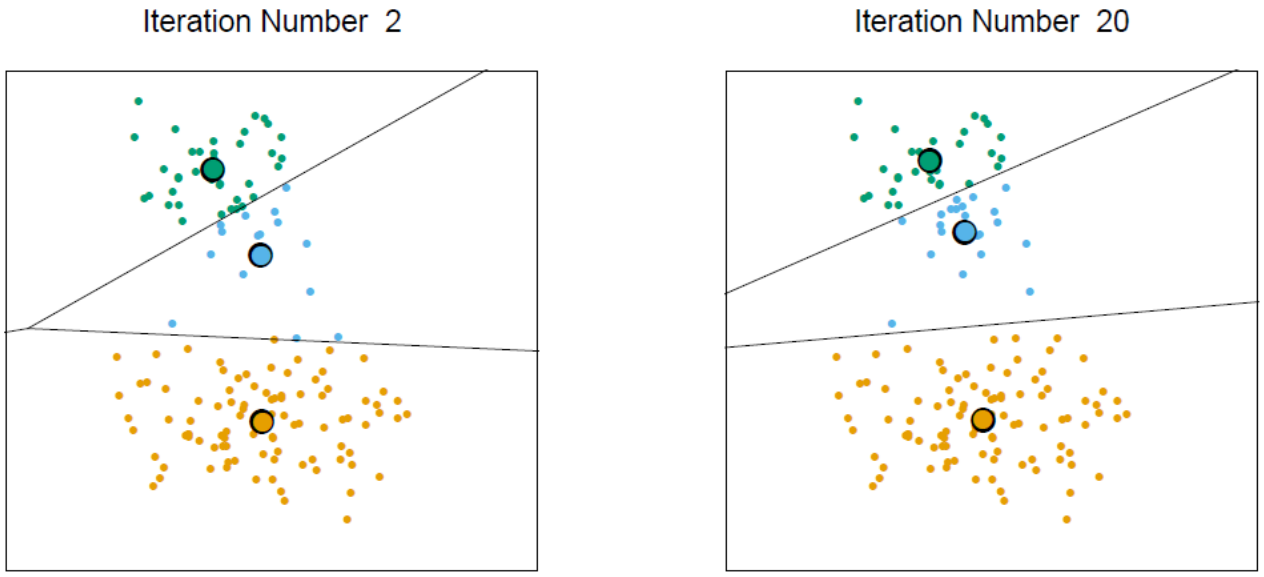

Figure: Successive iterations of the K-means clustering algorithm for a simulated data.

K=medoids clustering

Algorithm 14.2 *K-medoids Clustering.*

1. For a given cluster assignment C find the observation in the cluster minimizing total distance to other points in that cluster:

$$i_k^* = \underset{\{i:C(i)=k\}}{\operatorname{argmin}} \sum_{C(i')=k} D(x_i, x_{i'}). \tag{14.35}$$

   Then $m_k = x_{i_k^*}$, $k = 1, 2, \ldots, K$ are the current estimates of the cluster centers.

2. Given a current set of cluster centers $\{m_1, \ldots, m_K\}$, minimize the total error by assigning each observation to the closest (current) cluster center:

$$C(i) = \underset{1 \le k \le K}{\operatorname{argmin}} D(x_i, m_k). \tag{14.36}$$

3. Iterate steps 1 and 2 until the assignments do not change.

K medoids clustering

- Medoids clustering does not require all variables to be of the quantitative type.
- squared Euclidean distance can be replaced with distances robust to outliers.
- Finding the center of each cluster with Medoids clustering costs $O(N_k^2)$ flops, while with K-means, it costs $O(N_k)$. Thus, K-medoids is far more computationally intensive than K-means.

Initialization of K centers

- Choose one center uniformly at random from among the data points.
- For each data point x, compute $D(x)$, the distance between x and the nearest center that has already been chosen.
- Choose one new data point at random as a new center, using a weighted probability distribution where a point x is chosen with probability proportional to $D(x)^2$.
- Repeat Steps 2 and 3 until k centers have been chosen.
- Now that the initial centers have been chosen, proceed using standard K-means clustering.

This seeding method yields considerable improvement in the final error of K-means. Although the initial selection in the algorithm takes extra time, the K-means part itself converges very quickly after this seeding and thus the algorithm actually lowers the computation time.

Choice of K

- Cross-validation chooses large K, because the within cluster dissimilarity W_K decreases as K increases, even for test data!
- An estimate K^* for the optimal K^* can be obtained by identifying a "kink" in the plot of W_k as a function of K, that is, a sharp decrease of W_k followed by a slight decrease.
- Uses the **Gap statistic** - it chooses the K^* where the data look most clustered when compared to uniformly-distributed data.
  - For each K, compute the within cluster dissimilarity $\tilde{W}_K$ and its standard deviation s_K, for m sets of ranomly generated uniformly-distributed data.
- Choose K such that $G(K) \geq G(K+1) - s_K\sqrt{1+1/m}$ with $G(K) = |\log W_k - \log \tilde{W}_K|$.

Gap statistic

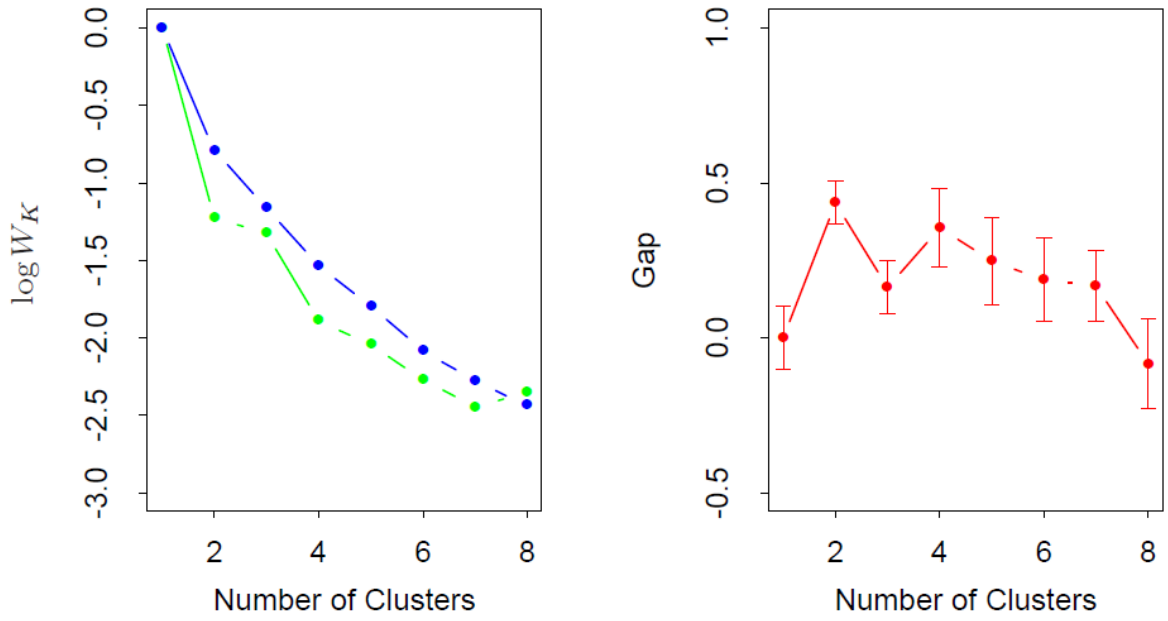

FIGURE 14.11. *(Left panel): observed (green) and expected (blue) values of* $\log W_K$ *for the simulated data of Figure 14.4. Both curves have been translated to equal zero at one cluster. (Right panel): Gap curve, equal to the difference between the observed and expected values of* $\log W_K$*. The Gap estimate* K^* *is the smallest* K *producing a gap within one standard deviation of the gap at* $K+1$*; here* $K^* = 2$*.*

Hierarchical clustering (Agglomerative)

- Produce hierarchical representations in which the clusters at each level of the hierarchy are created by merging clusters at the next lower level.
- Do not require initial configuration assignment and initial choice of the number of clusters
- Do require the user to specify a **measure of dissimilarity between (disjoint) groups of observations**, based on the pairwise dissimilarities among the observations in the two groups.

Agglomerative clustering

- Begin with every observation representing a singleton cluster.
- At each of the $N-1$ steps the closest two (least dissimilar) clusters are merged into a single cluster, producing one less cluster at the next higher level, according to a measure of dissimilarity between two clusters. Let G and H represent two groups.

- **Single linkage(SL) clustering**:

$$d_{SL}(G,H) = \min_{i \in G, i' \in H} d_{ii'}$$

- **Complete linkage (CL) clustering**:

$$d_{CL}(G, H) = \max_{i \in G, i' \in H} d_{ii'}$$

- **Group average clustering**:

$$d_{GA}(G, H) = \frac{1}{N_G N_H} \sum_{i \in G} \sum_{i' \in H} d_{ii'}$$

where N_G and N_H are the respective number of observations in each group.

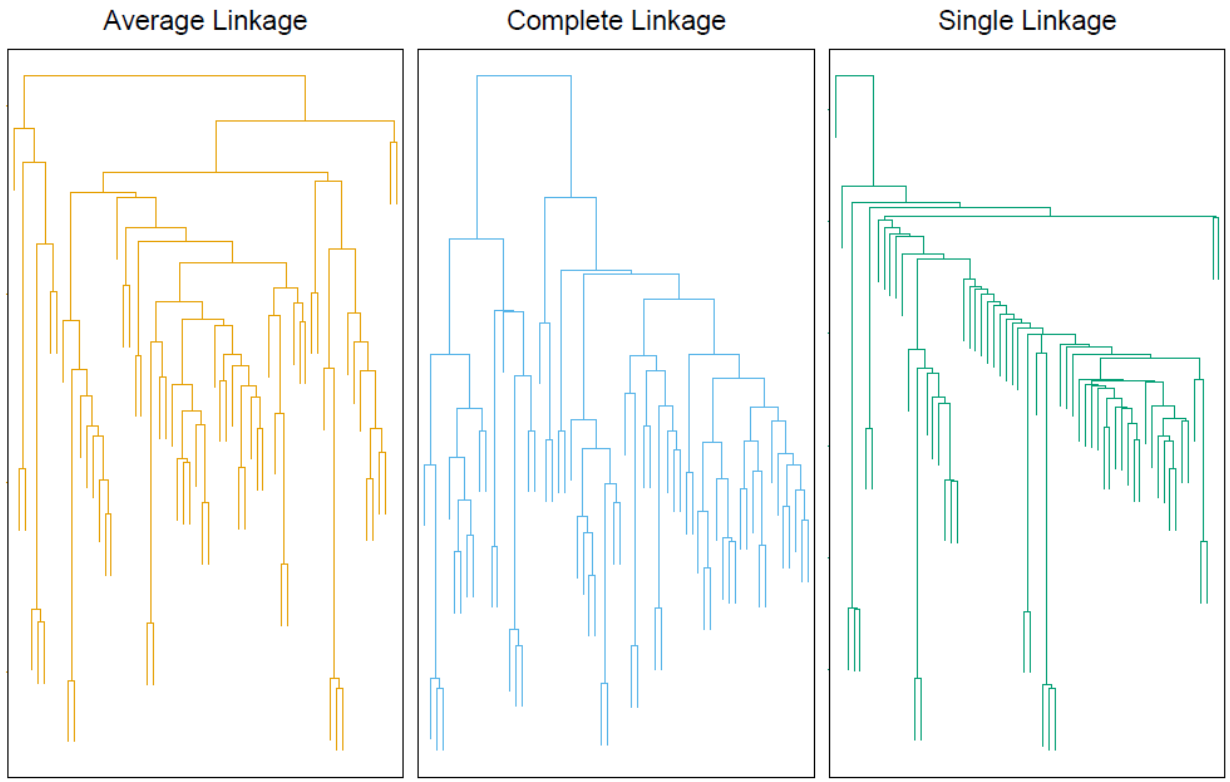

FIGURE 14.13. *Dendrograms from agglomerative hierarchical clustering of human tumor microarray data.*

Comparison of different cluster dissimilarity measures

- If the data exhibit a strong clustering tendency, with each of the clusters being compact and well separated from others, then all three methods produce similar results. Clusters are compact if all of the observations within them are relatively close together (small dissimilarities) as compared with observations in different clusters.
- Single linkage has a tendency to combine, at relatively low thresholds, observations linked by a series of close intermediate observations. This phenomenon, referred to as **chaining**, is often considered a defect of the method. The clusters produced by single linkage can violate the "compactness" property
- Complete linkage will tend to produce compact clusters. However, it can produce clusters that violate the "closeness" property. That is, observations assigned to a cluster can be much closer to members of other clusters than they are to some members of their own cluster.
- Group average represents a compromise between the two.

Comparison of different cluster dissimilarity measures

- Single linkage and complete linkage clustering are invariant to monotone transformation of the distance function, but the group average clustering is not.
- The group average dissimilarity is an estimate of

$$\int \int d(x, x') p_G(x) p_H(x') dx dx'$$

which is a kind of distance between the two densities p_G for group G and p_H for group H. On the other hand, single linkage dissimilarity approaches zero and complete linkage dissimilarity approaches infinity as N. Thus, it is not clear what aspects of the population distribution are being estimated by the two group dissimilarities.

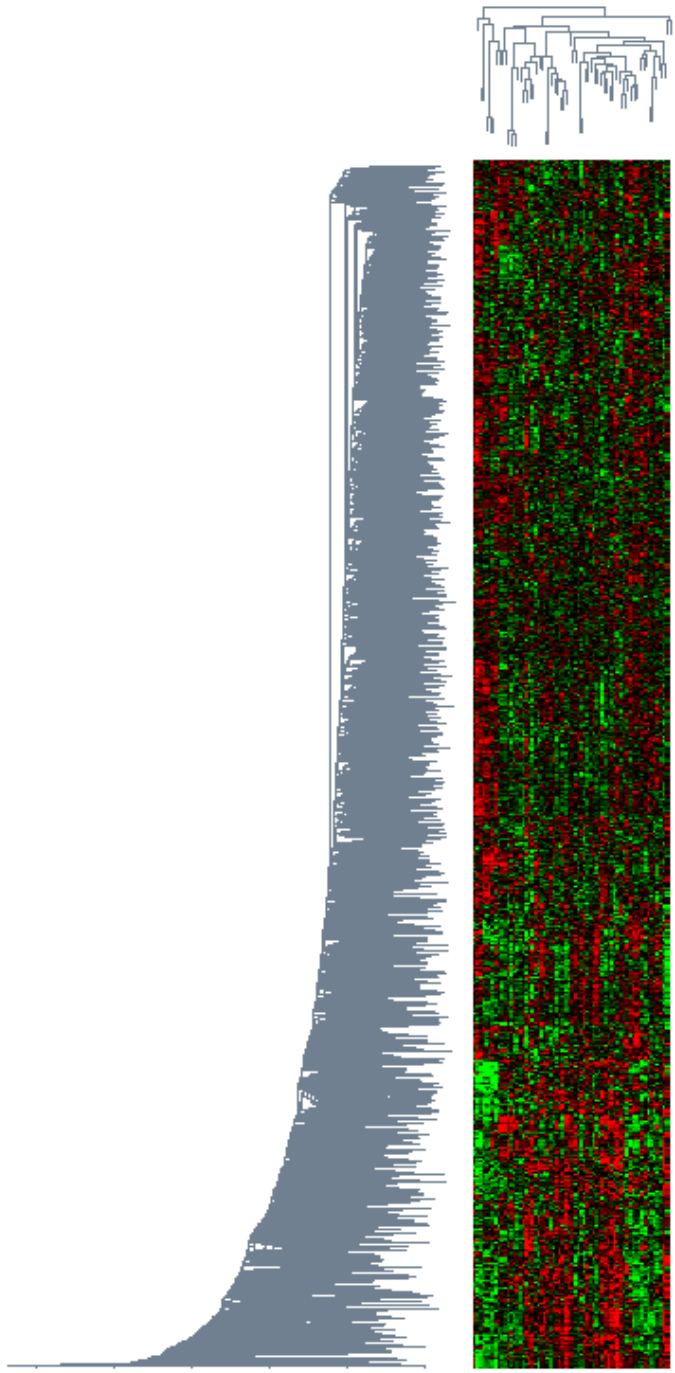

FIGURE 14.14. *DNA microarray data: average linkage hierarchical clustering has been applied independently to the rows (genes) and columns (samples), determining the ordering of the rows and columns (see text). The colors range from bright green (negative, under-expressed) to bright red (positive, over-expressed).*