Project

- Project due (before) Wednesday, April 13 (5 pm)
- Two to three pages of write-up:
- introduction to data
- questions of interest
- methods and models used
- tables and figures
- conclusions
- code as Appendix

- training sample $(x_1, \ldots, x_N)$; each case has p measurements (features); no response y
- want information on the probability function (density) of $X = (X_1, \ldots, X_p)$ based on these N observations
- if $p = 1$ or 2, can use kernel density estimation as in §6.6
- we also used density estimation to construct a classifier, via Naive Bayes
- most unsupervised learning methods try to find regions of feature space (R^p) with high probability
- this is called density modelling in Roweis' CSC2515 Lecture 7
- somewhat more specialized techniques are clustering: classification with missing class variable or dimension reduction: regression with missing response variable
- no loss function to ascertain/estimate how well we're doing (exploratory data analysis)

- discover groupings among the cases; cases within clusters should be 'close' and clusters should be 'far apart' (Figure 14.5)
- many (not all) clustering methods use as input an $N \times N$ matrix D of dissimilarities
- require $D_{ii'} > 0$, $D_{ii'} = D_{i'i}$ and $D_{ii} = 0$
- sometimes the data are collected this way (see §14.3.1) but more often D needs to be constructed from the $N \times p$ data matrix
- this can be done using `dist` or `daisy` (the latter in the R library `cluster`)
- often (usually) $D_{ii'} = \sum_{j=1}^{p} d_j(x_{ij}, x_{i'j})$, where $d_j(\cdot, \cdot)$ to be chosen, e.g. $(x_{ij} - x_{i'j})^2$, $|x_{ij} - x_{i'j}|$, etc.
- sometimes $D_{ii'} = \sum_{j=1}^{p} w_j d_j(x_{ij}, x_{i'j})$, with weights to be chosen. (extensive discussion of weights on pp 457–9).

suppose number of clusters K is fixed ($K < N$), write $C(i) = k$ to mean observation i is assigned to cluster k

$$\begin{aligned} T &= \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N} D_{ii'} \\ &= \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\left(\sum_{C(i')=k} D_{ii'} + \sum_{C(i')\neq k} D_{ii'}\right) \\ &= \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')=k} D_{ii'} + \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')\neq k} D_{ii'} \\ &= W(C) + B(C) \end{aligned}$$

where $W(C)$ is a measure of within cluster dissimilarity and $B(C)$ is a measure of between cluster dissimilarity. Since T is fixed given the data, the goal of minimizing $W(C)$ is the same as that of maximizing $B(C)$

Clustering methods (§14.3)

- unsupervised learning sometimes called exploratory (multivariate) analysis, cf. VR (Ch. 11)
- well constructed pictures of the data often as informative
- other methods of unsupervised learning include projection methods
- ”classification” sometimes used to mean finding clusters; e.g. Gordon
- Ripley 96 does not recommend using clusters as a technique for classification
- partitioning methods: K-means, K-medioids, Vector Quantization
- hierarchical methods; top-down or bottom-up
- both are combinatorial methods; in contrast to model-based methods such as Gaussian mixtures.
- Note: VQ used in signal processing literature (and is the same as ?) K-means clustering (Figure 14.9)

- K-Means – uses the original data
- uses Euclidean distance $D_{ii'} = \sum_{j=1}^{p}(x_{ij} - x_{i'j})^2$
- requires a starting classification
- minimizes the within-cluster sum of squares
- maximizes the between-cluster sum of squares
- variables should be 'suitably scaled' (Ripley): no mention of this in HTF
- K-medioids: replace Euclidean by another dissimiilarity measure

$$D_{ii'} = \sum_{j=1}^{p} |x_{ij} - x_{i'j}| \quad \text{manhattan}$$

$$D_{ii'} = \sum_{j=1}^{p} \frac{|x_{ij} - x_{i'j}|}{|x_{ij} + x_{i'j}|} \quad \text{Canberra}$$

- most algorithms use a 'greedy' approach by modifying a given clustering to decrease within cluster distance: analogous to forward selection in regression
- K-means clustering is (usually) based on Euclidean distance: $D_{ii'} = ||x_i - x_{i'}||^2$, so x's should be centered and scaled (and continuous)
- Use the result

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

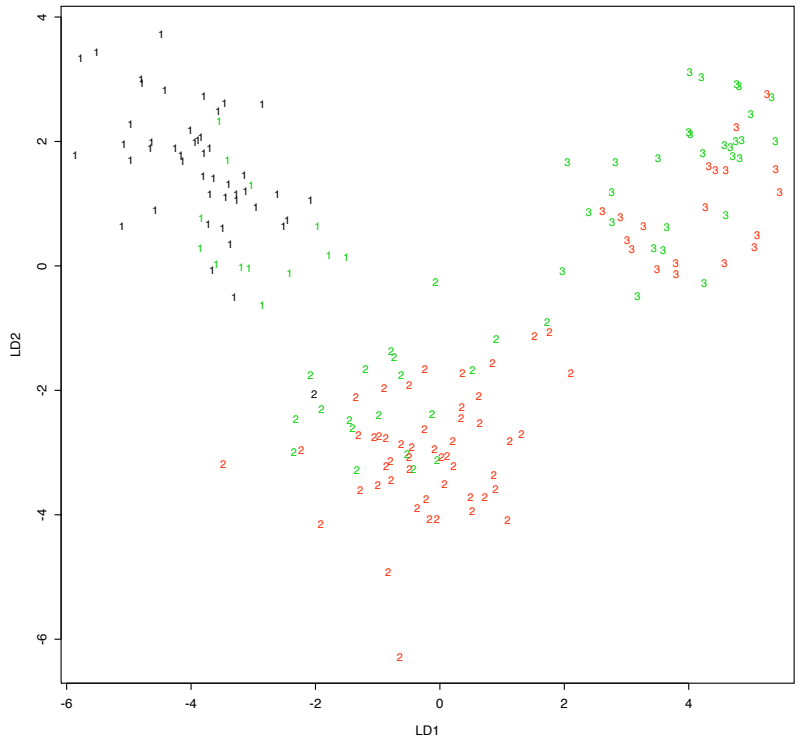
  where N_k is the number of observations in cluster k and $\bar{x}_k = (\bar{x}_{1k}, \ldots, \bar{x}_{pk})$ is the mean in the kth cluster
- The algorithm starts with a current set of clusters, and computes the cluster means. Then assign observations to clusters by finding the cluster whose mean is closest. Recompute the cluster means and continue.

- sometimes require cluster center to be one of the data values (means that algorithm can be applied to dissimilarity matrices directly)
- choose K by possibly plotting the total within cluster dissimilarity vs. K; it is always decreasing but a 'kink' may be evident (see §14.3.11).
- hard to describe the results of partitioning methods of clustering, although see Figure 14.6.
- Algorithm 14.1:
  - for a given cluster assignment, minimize the total cluster variance $\sum_{k=1}^{K} N_k \sum_{C(i)=k} ||x_i - m_k||^2$ with respect to $\{m_1, \ldots, m_K\}$; this is easily achieved by taking each m_k to be the sample mean of the kth cluster
  - For a given set of $\{m_k\}$, minimize distance by letting $C(i) = \text{argmin}_{1 \leq k \leq K} ||x_i - m_k||^2$

Example: wine data

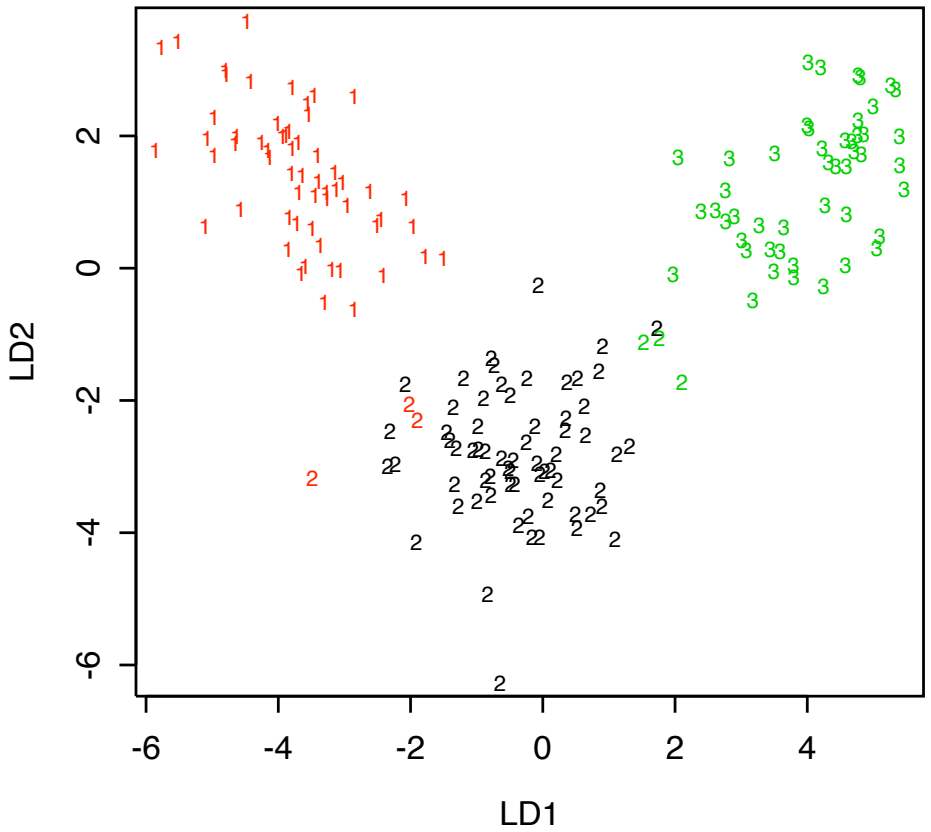
Example: wine data

linear discriminant analysis showed a good separation of the 3 classes. I ran K-means with a random choice of initial cluster



and got the following: The numbers show the classes in the original data, and the colors show the K-means clusters. The overlap is not very good. Then I standardized each column of the wine matrix to have mean zero and variance 1, and ran K-means again.

Example: wine data



Here clustering has been quite effective.

- binary: simple matching uses

$$D_{ii'} = (\#\{(1,0) \text{ or } (0,1) \text{ pairs }\})/p$$

  Jacard coefficient uses

$$D_{ii'} = (\#\{(1,0) or (0,1) \text{ pairs }\})/(\#\{(1,0),(0,1) \text{ or } (1,1) \text{ pairs }\})$$

- ordered categories – use ranks as continuous data (see eq. (14.23))
- unordered categories – create binary dummy variables and use matching
- mixed categories – Gower's 'general dissimilarity coefficient' – see Gordon

```
dist(x, method = c("euclidean", "maximum",
"manhattan", "canberra", "binary"))
```

where `maximum` is $\max_{1 \leq j \leq p}(x_{ij} - x_{i'j})$ and `binary` is Jacard coefficient.

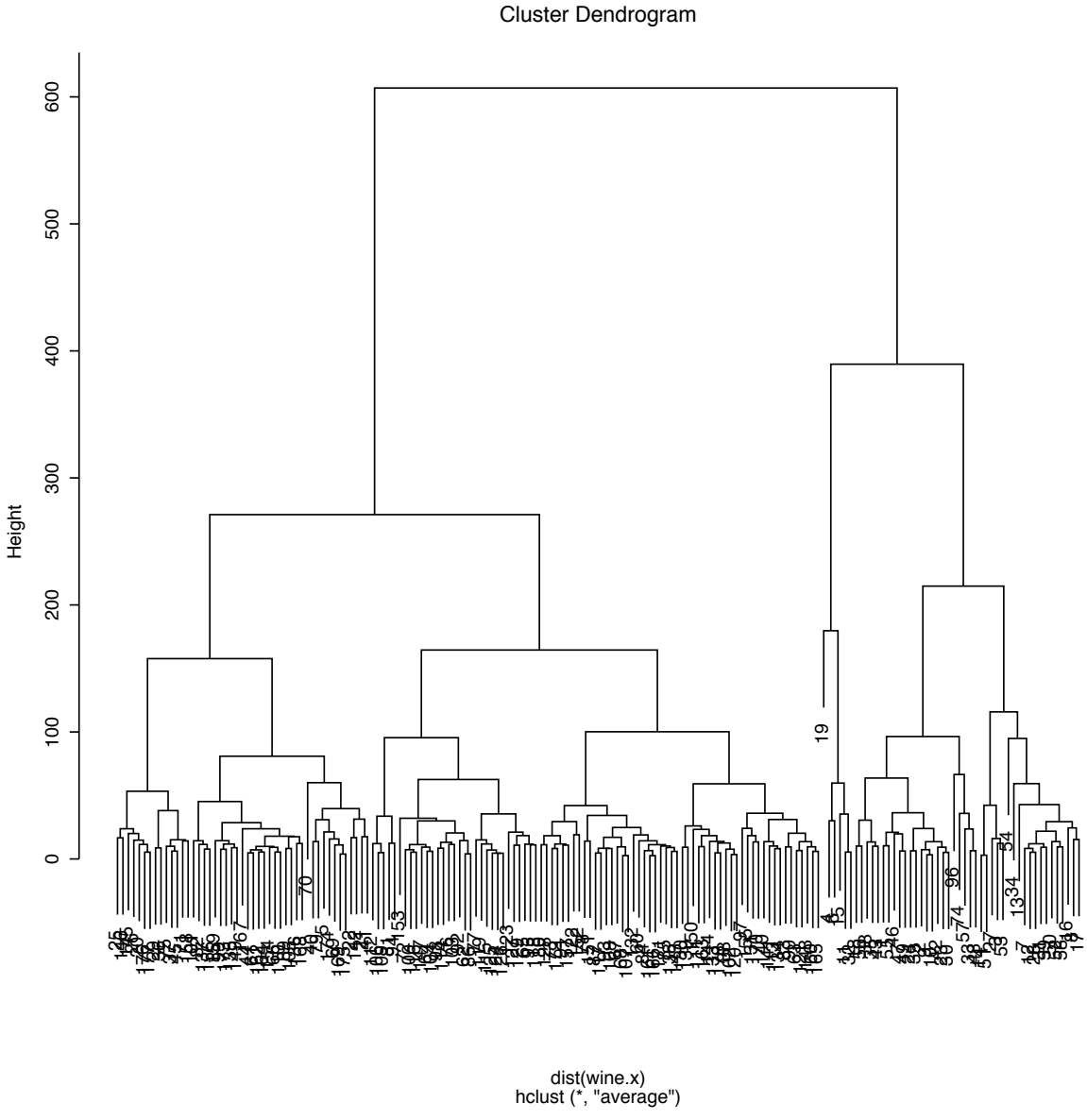
```
daisy(x, metric=c("euclidean", "manhattan",
standardize=F, type=c("ordratio","logratio","asymm"
```

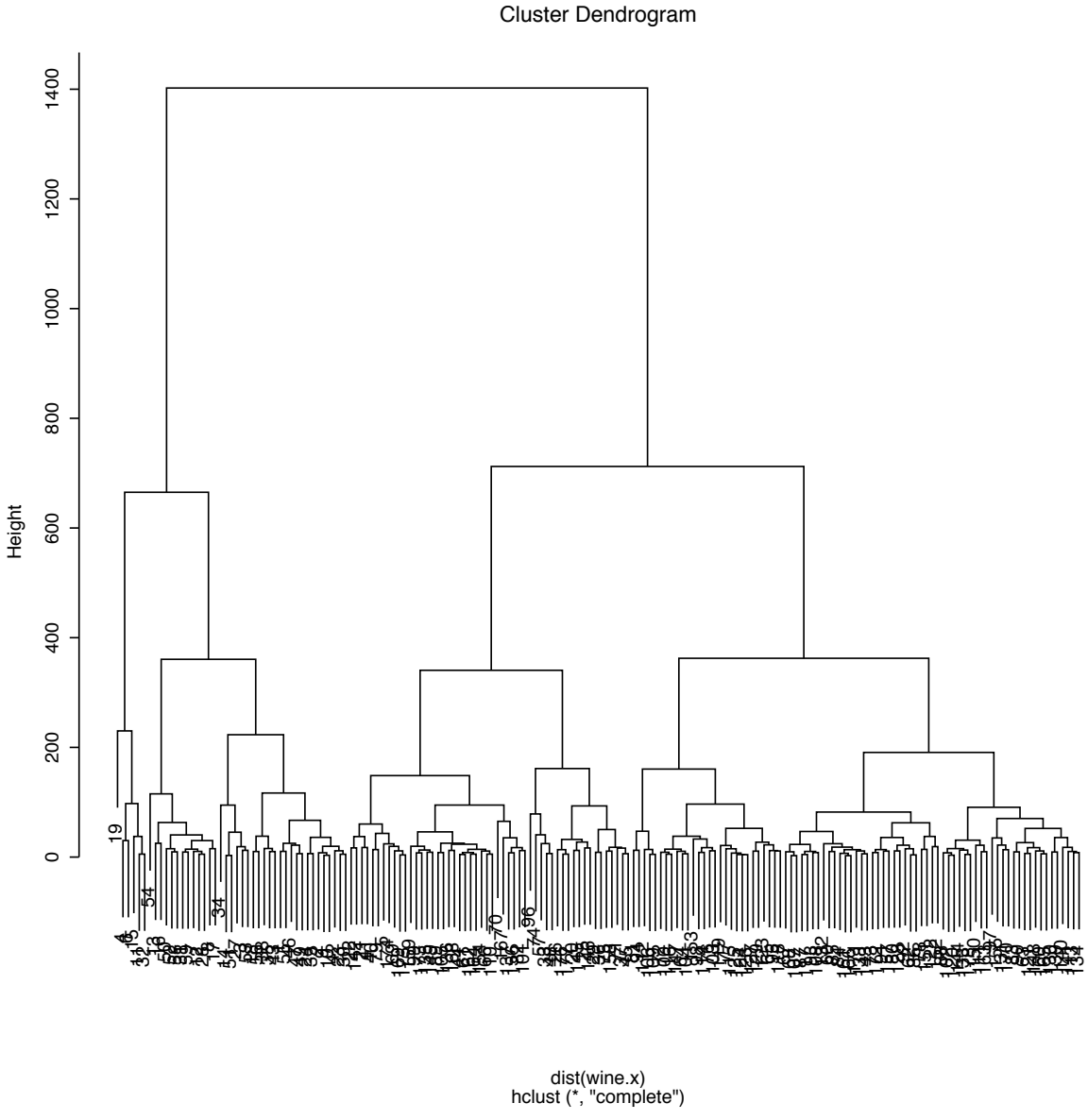
(see the help files)

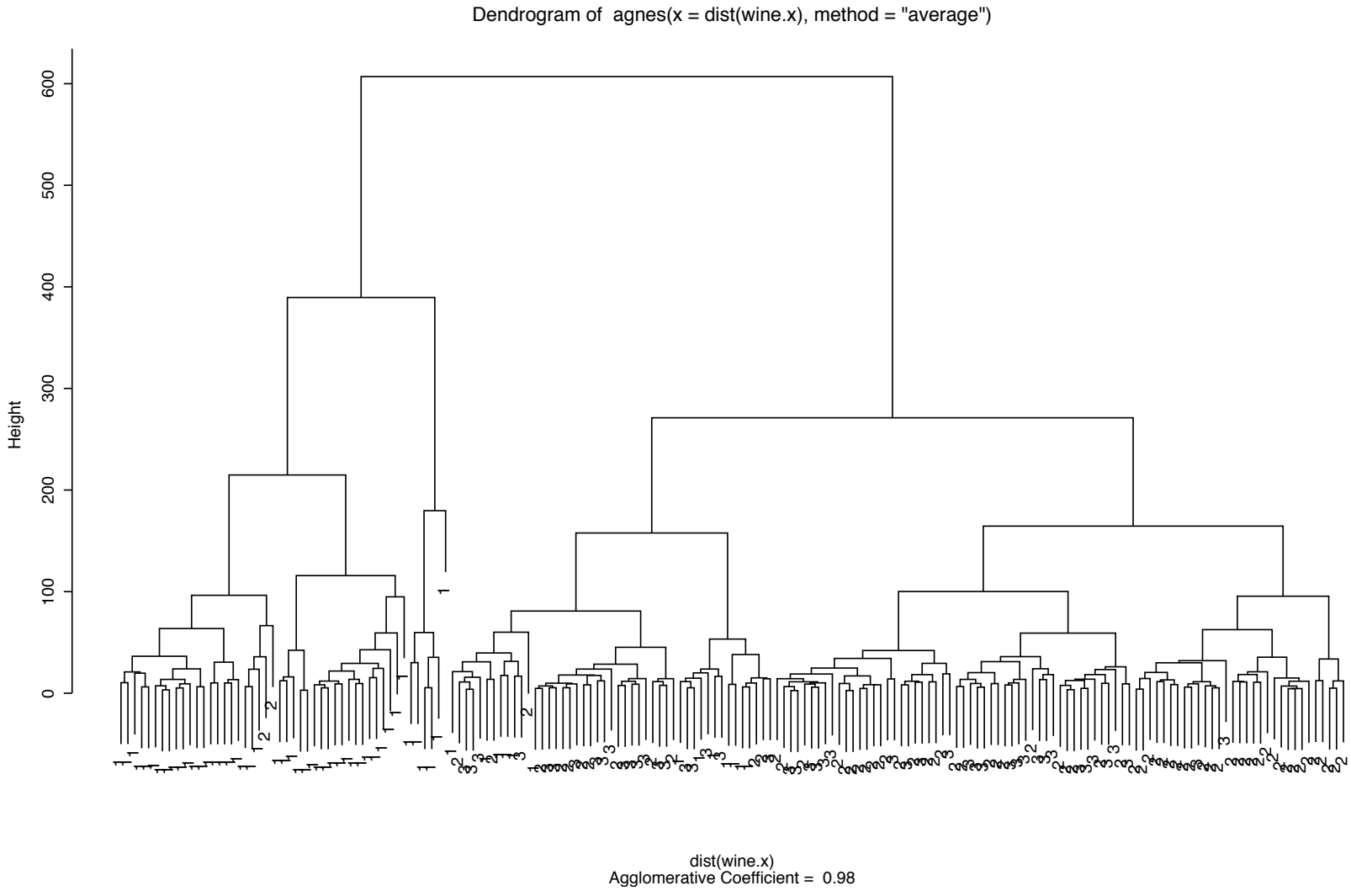
- bottom up: each value is a cluster, iterate: find the 'closest' pair of clusters C_i and $C_{i'}$ merge them
- need a measure for distance between points and between clusters (the clusters needn' tbe vectors)
- single link clustering measures the distance between clusters by the minimum distance $d(C_1, C_2) = \min_{i \in C_1, i' \in C_2} D_{ii'}$
- susceptible to 'chaining'; long strings of points assigned to the same cluster
- sensitive to outliers
- not useful for segmentation

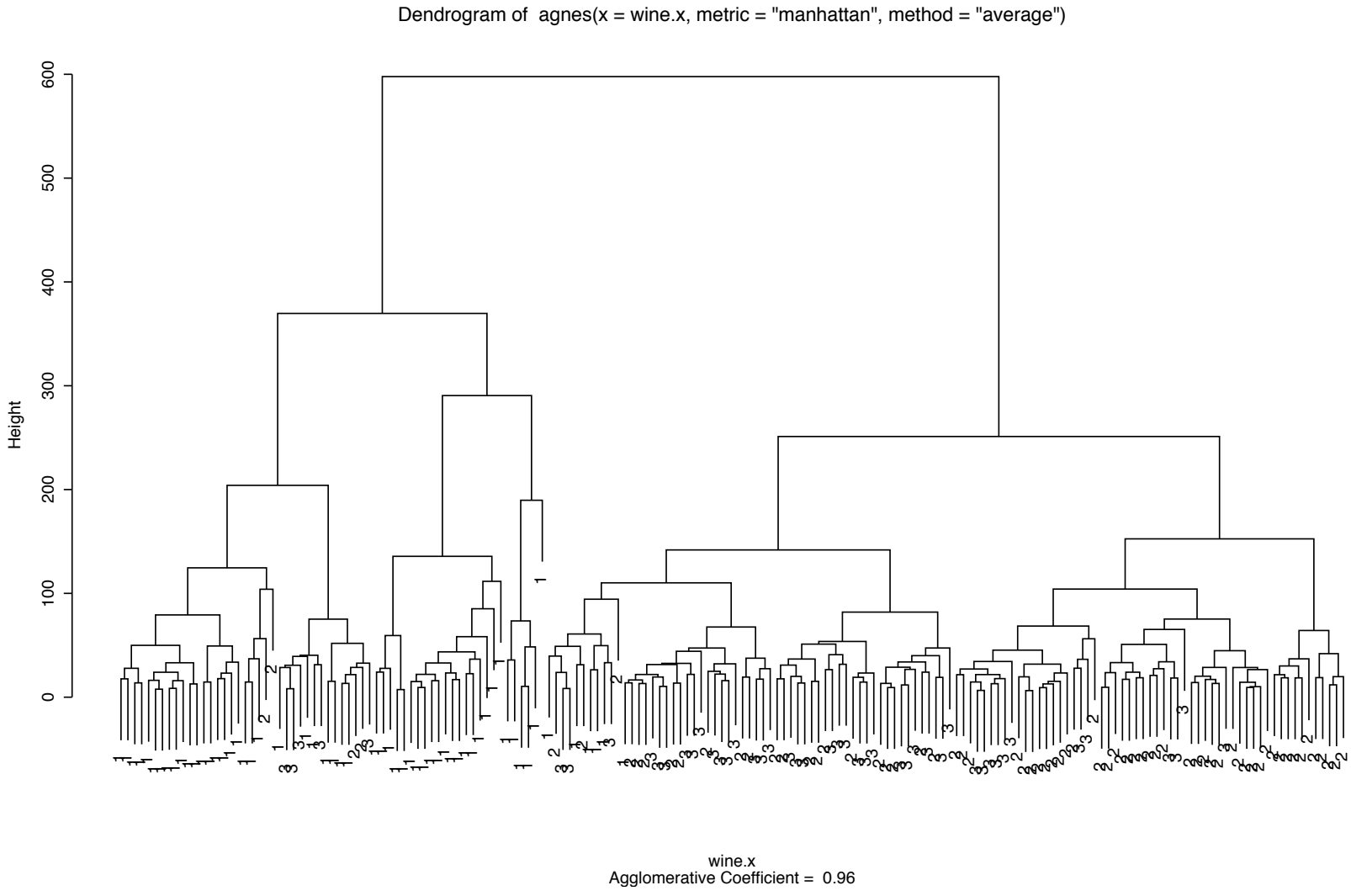
- has an invariance property: if two pairs of clusters are equidistant it doesn't matter which pair is merged first
- complete linkage $d(C_1, C_2) = \max_{i \in C_1, i' \in C_2} D_{ii'}$
- clusters then to be of equal size in 'volume of space' occupied
- useful for segmentation
- group average intermediate between complete and single linkage.
- Hierarchical clustering is easily pictured in a dendogram, see Figs 14.12 and 14.13. Note that the 'look' is quite different for different linkages. Implemented in R in `hclust` and `agnes`.

Cluster Dendrogram
Height
0
20
40
60
80
100
120
dist(wine.x)
hclust (*, "single")

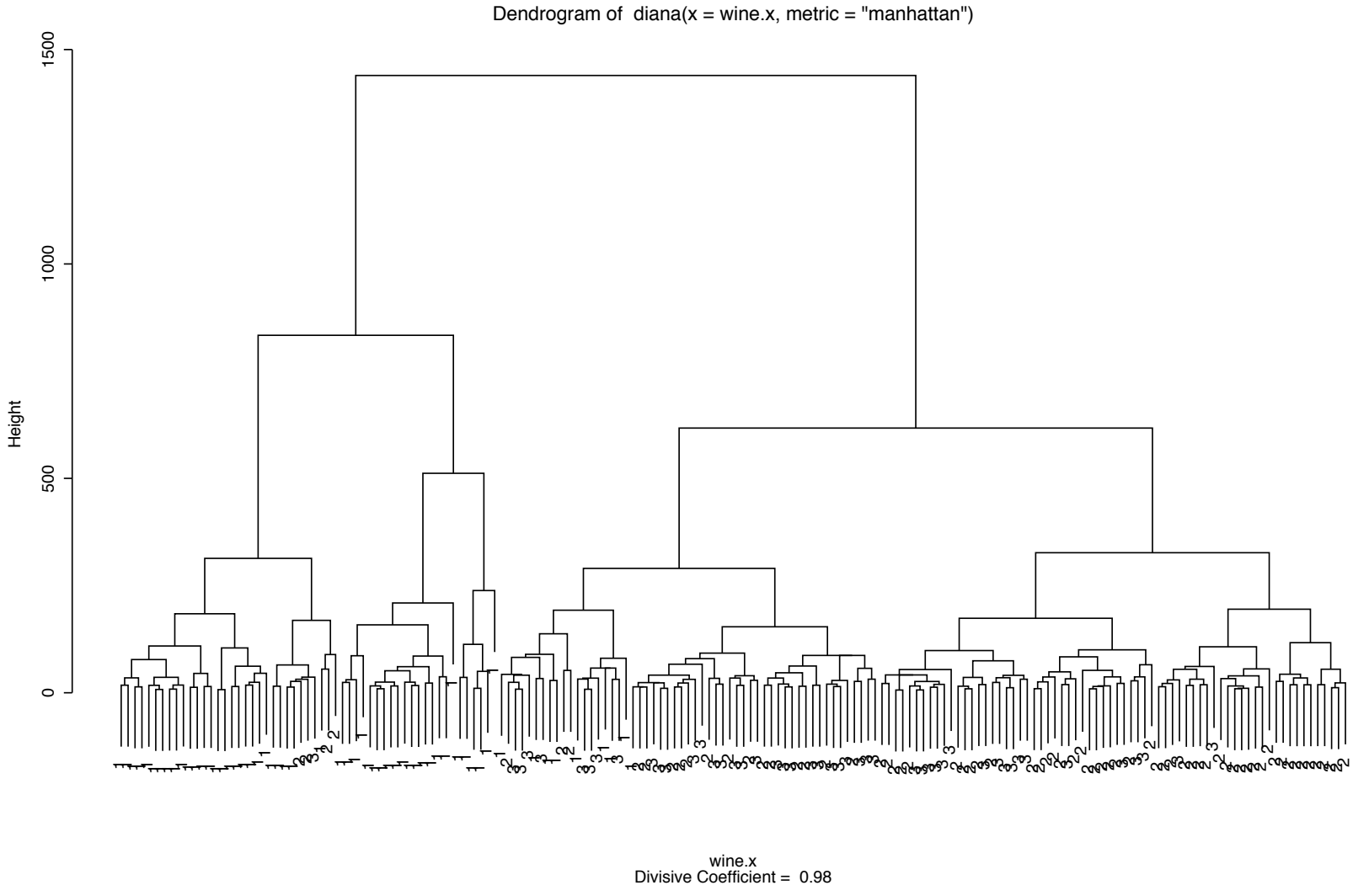

Cluster Dendrogram
Height
0
100
200
300
400
500
600
dist(wine.x)
hclust (*, "average")


Cluster Dendrogram
Height
0
200
400
600
800
1000
1200
1400
dist(wine.x)
hclust (*, "complete")


Dendrogram of agnes(x = dist(wine.x), method = "average")
Height
0
100
200
300
400
500
600
dist(wine.x)
Agglomerative Coefficient = 0.98


Dendrogram of agnes(x = wine.x, metric = "manhattan", method = "average")
Height
0
100
200
300
400
500
600
wine.x
Agglomerative Coefficient = 0.96

– top down starts with all observations in one cluster, and amalgamate. Computationally more intensive, harder to find optimal clusters. Implemented in R in `diana`.



- assume each X_j takes values in a set S_j
- let $s_j \subseteq S_j$ be a subset of these values
- example: age classes (0–14, 15–24, ...)
- example: employment status (working full-time, working part-time, seeking work, ...)
- Goal: find $s_1, s_2, \ldots s_p$ so that

$$\Pr(X_j \in s_j, j = 1, \ldots, p) = \Pr\{\cap_{j=1}^{p}(X_j \in s_j)\}$$

  relatively large
- Note if $s_j = S_j$ then $\Pr(X_j \in s_j) = 1$, i.e. X_j "does not appear"
- Simplification: s_j either S_j or a single value (called v_{0j} on p.440)
- Then want to find subsets $\mathcal{J} \subset \{1, \ldots, p\}$ and values $v_{0j}, j \in \mathcal{J}$ so that $\Pr(\cap_{\mathcal{J}} S_j = v_{0j})$ is large

- Special case: each $X_j = 0, 1$ (binary features) then $v_{0j} = 1$ and $\cap_{j \in \mathcal{J}}(X_j = 1) \Rightarrow \prod_{j \in \mathcal{J}} X_j = 1$
- If X_j takes a finite number of values, $v_{j1}, \ldots v_{jn_j}$, say, then create n_j dummy variables $Z_{j1}, Z_{j2}, \ldots, Z_{jn_j}$ that are binary
- Renumber these to $Z_1, \ldots, Z_K$; goal is now to find a subset $\mathcal{K} \subset \{1, \ldots K\}$ to give a large value of

$$\Pr(\prod_{k \in \mathcal{K}} Z_k = 1)$$

- This is estimated by

$$\frac{1}{N} \sum_{i=1}^{N} \prod_{k \in \mathcal{K}} z_{ik} = \widehat{\Pr}(\prod_{\mathcal{K}} Z_k = 1) \equiv T(\mathcal{K})$$

- Implementation: Find all sets $\mathcal{K}_\ell$ so that $T(\mathcal{K}_\ell) > t$: this reduces the number of possible item sets.

- $\mathcal{K}$ is an item set.
- $$T(\mathcal{K}) = \frac{1}{N}\sum_{i=1}^{N}\prod_{k\in\mathcal{K}} z_{ik}$$
- prevalence of the item set $\mathcal{K}$.
- §14.2.2 describes the APriori algorithm for implementing this
- The item sets $\mathcal{K}_\ell$ are described by a set of association rules $A \Rightarrow B$ example $\{\text{peanut butter, jelly}\} \Rightarrow \{\text{bread}\}$
- and summarized by estimates of

$$\begin{array}{lll} T(A \Rightarrow B) & \Pr(A \cap B) & \text{“support”} \\ C(A \Rightarrow B) & \Pr(B \mid A) & \text{“confidence”} \\ & \dfrac{\Pr(A \cap B)}{\Pr(A)\Pr(B)} & \text{“lift”} \end{array}$$

- See §14.2.3 for an example (that gave 6288 rules!)

Association Rules (§14.2)

- $\mathcal{K}$ is an item set.
- $$T(\mathcal{K}) = \frac{1}{N}\sum_{i=1}^{N}\prod_{k\in\mathcal{K}} z_{ik}$$
- prevalence of the item set $\mathcal{K}$.
- §14.2.2 describes the APriori algorithm for implementing this
- The item sets $\mathcal{K}_\ell$ are described by a set of association rules $A \Rightarrow B$ example $\{\text{peanut butter, jelly}\} \Rightarrow \{\text{bread}\}$
- and summarized by estimates of

$T(A \Rightarrow B)$	$\Pr(A \cap B)$	"support"
$C(A \Rightarrow B)$	$\Pr(B \mid A)$	"confidence"
	$\frac{\Pr(A \cap B)}{\Pr(A)\Pr(B)}$	"lift"

- See §14.2.3 for an example (that gave 6288 rules!)

If we are interested in a particular consequence, $P(B \mid A)$, we could create a 'response' variable $y = 1\{x \in B\}$ and use methods for supervised learning such as logistic regression, classification, etc.

A more clever use of supervised learning for association rules is described in §14.2.4 and §14.2.5, suggestion in §14.2.6 to use CART

References

- Pattern Recognition and Neural Networks. B.D. Ripley (1996), Cambridge University Press. *Good discussion of many machine learning methods.*
- Classification (2nd ed.), A. D. Gordon (1999), Chapman & Hall/CRC Press. *Unsupervised learning/clustering; see Ch. 2 for good description of dissimilarity measures.*
- Finding Groups in Data: An Introduction to Cluster Analysis, L. Kaufman and P.J. Rousseeuw, (1990) Wiley. *Learn all about daisy, agnes, and many other of R's clustering methods.*
- Modern Applied Statistics with S (4th Ed.), W.N. Venables and B.D. Ripley (2002), Springer-Verlag. *The bible for computing with Splus and R; Ch. 11 covers unsupervised learning, Chs. 8,9 and 12 cover supervised learning.*
- Principles of Data Mining. D. Hand, H. Mannila, P. Smyth (2001) MIT Press. *Nice blend of computer science and statistical methods. Clustering covered in Ch. 9*