Descriptive Modeling

Based in part on Chapter 9 of Hand, Manilla, & Smyth
And Section 14.3 of HTF

David Madigan
Data Mining Algorithms

“A data mining algorithm is a well-defined procedure that takes data as input and produces output in the form of models or patterns”

Hand, Mannila, and Smyth

“well-defined”: can be encoded in software

“algorithm”: must terminate after some finite number of steps
Models

- Prediction
  - Linear regression
  - Piecewise linear
  - Nonparametric regression
  - Classification

- Probability Distributions
  - Parametric models
  - Mixtures of parametric models
  - Graphical Markov models (categorical, continuous, mixed)

- Structured Data
  - Time series
  - Markov models
  - Mixture Transition Distribution models
  - Hidden Markov models
  - Spatial models
Patterns

Global

- Clustering via partitioning
- Hierarchical Clustering
- Mixture Models

Local

- Outlier detection
- Changepoint detection
- Bump hunting
- Scan statistics
- Association rules
What is a descriptive model?

• “presents the main features of the data”
• “a summary of the data”
• Data randomly generated from a “good” descriptive model will have the same characteristics as the real data
• Chapter focuses on techniques and algorithms for fitting descriptive models to data
Estimating Probability Densities

- parametric versus non-parametric
- log-likelihood is a common score function:
  \[ S_L(\theta) = -\sum_{i=1}^{n} \log p(x(i);\theta) \]
- Fails to penalize complexity
- Common alternatives:
  \[ S_{\text{BIC}}(M_k) = 2S_L(\hat{\theta}_k;M_k) + d_k \log n \]
  \[ S_{\text{VL}}(M_k) = -\sum_{x \in D_v} \log \hat{p}_{M_k}(x \mid \theta) \]
Parametric Density Models

• Multivariate normal

• For large $p$, number of parameters dominated by the covariance matrix

• Assume $\Sigma = I$?

• Graphical Gaussian Models

• Graphical models for categorical data
Mixture Models

\[ f(x) = p \frac{(\lambda_1)^x e^{-\lambda_1}}{x!} + (1 - p) \frac{(\lambda_2)^{52-x} e^{-\lambda_2}}{(52-x)!} \]

“Two-stage model”

\[ f(x) = \sum_{k=1}^{K} \pi_k f_k(x; \theta_k) \]
x<-c(rpois(99000,2.2),52-rpois(1000,3))
hist(x,nclass=52)

OR
n<-100000; x <- rep(0,n)
for (i in 1:n) {
  if (runif(1) < 0.01) {
    x[i] <- 52-rpois(1,3)
  }
  else {
    x[i] <- rpois(1,2.2)
  }
}

x<-c(rpois(89000,2.2),52-rpois(1000,3),trunc(runif(10000,0,53)))
hist(x,nclass=52)
Mixture Models and EM

• No closed-form for MLE’s

• EM widely used - flip-flop between estimating parameters assuming class mixture component is known and estimating class membership given parameters.

• Time complexity $O(Kp^2n)$; space complexity $O(Kn)$

• Can be slow to converge; local maxima
Mixture-model example

Market basket: \( x_j(i) = \begin{cases} 
1, \text{if person } i \text{ purchased item } j \\
0, \text{otherwise}
\end{cases} \)

For cluster \( k \), item \( j \): \( p_k(x_j(i) = 1) = \theta_{kj} \)

Thus for person \( i \): \( p(x(i)) = \sum_{k=1}^{K} \pi_k \prod_{j} \theta_{kj}^{x_{j}(i)}(1 - \theta_{kj})^{1 - x_{j}(i)} \)

Probability that person \( i \) is in cluster \( k \): \( p(k \mid x(i)) = \frac{\pi_k \prod_{j} \theta_{kj}^{x_{j}(i)}(1 - \theta_{kj})^{1 - x_{j}(i)}}{p(x(i))} \) \( \text{E-step} \)

Update within-cluster parameters: \( \theta_{kj}^{\text{new}} = \frac{\sum_{i=1}^{n} p(k \mid x(i))x_{j}(i)}{\sum_{i=1}^{n} p(k \mid x(i))} \) \( \text{M-step} \)
from Padhraic Smyth's website…
Figure 8: Density estimation for the Lansing Woods maples. (a) BIC from model-based clustering. The maximum-BIC model is a six-component nonuniform spherical mixture. (b) Model-based classification, with circles indicating the circles defined by the estimated covariance of each of the six groups. (c) Contours of the density as determined by model-based clustering, with the location of the maples superimposed. (d) Contours of a standard Gaussian kernel density estimate with bandwidth selected by cross-validation.

Fraley and Raftery (2000)
Non-parametric density estimation

• Doesn’t scale very well - Silverman’s example

• Note that for Gaussian-type kernels estimating \( f(x) \) for some \( x \) involves summing over contributions from all \( n \) points in the dataset
The Curse of Dimensionality

\[ X \sim \text{MVN}_p (\mathbf{0}, \mathbf{I}) \]

- Gaussian kernel density estimation
- Bandwidth chosen to minimize MSE at the mean
- Suppose want:

\[
E[(\hat{p}(x) - p(x))^2] \quad \text{subject to} \quad x = 0
\]

<table>
<thead>
<tr>
<th>Dimension</th>
<th># data points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>67</td>
</tr>
<tr>
<td>6</td>
<td>2,790</td>
</tr>
<tr>
<td>10</td>
<td>842,000</td>
</tr>
</tbody>
</table>
What is Cluster Analysis?

• Cluster: a collection of data objects
  – Similar to one another within the same cluster
  – Dissimilar to the objects in other clusters

• Cluster analysis
  – Grouping a set of data objects into clusters

• Clustering is unsupervised classification: no predefined classes

• Typical applications
  – As a stand-alone tool to get insight into data distribution
  – As a preprocessing step for other algorithms
General Applications of Clustering

• Pattern Recognition

• Spatial Data Analysis
  – create thematic maps in GIS by clustering feature spaces
  – detect spatial clusters and explain them in spatial data mining

• Image Processing

• Economic Science (especially market research)

• WWW
  – Document classification
  – Cluster Weblog data to discover groups of similar access patterns
Examples of Clustering Applications

- **Marketing**: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs.
- **Land use**: Identification of areas of similar land use in an earth observation database.
- **Insurance**: Identifying groups of motor insurance policy holders with a high average claim cost.
- **City-planning**: Identifying groups of houses according to their house type, value, and geographical location.
- **Earthquake studies**: Observed earthquake epicenters should be clustered along continent faults.
What Is Good Clustering?

• A good clustering method will produce high quality clusters with
  – high intra-class similarity
  – low inter-class similarity
• The quality of a clustering result depends on both the similarity measure used by the method and its implementation.
• The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns.
Major Clustering Approaches

- **Partitioning algorithms**: Construct various partitions and then evaluate them by some criterion.
- **Hierarchy algorithms**: Create a hierarchical decomposition of the set of data (or objects) using some criterion.
- **Density-based**: based on connectivity and density functions.
- **Grid-based**: based on a multiple-level granularity structure.
- **Model-based**: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other.
Partitioning Algorithms: Basic Concept

• **Partitioning method**: Construct a partition of a database $D$ of $n$ objects into a set of $k$ clusters

• Given a $k$, find a partition of $k$ clusters that optimizes the chosen partitioning criterion
  
  – Global optimal: exhaustively enumerate all partitions
  
  – Heuristic methods: $k$-means and $k$-medoids algorithms
  
  – $k$-means (MacQueen’67): Each cluster is represented by the center of the cluster
  
  – $k$-medoids or PAM (Partition around medoids) (Kaufman & Rousseeuw’87): Each cluster is represented by one of the objects in the cluster
The K-Means Algorithm

for \( k = 1, \ldots, K \) let \( r(k) \) be a randomly chosen point from \( D \);
while changes in clusters \( C_k \) happen do

form clusters:
for \( k = 1, \ldots, K \) do

\[ C_k = \{ x \in D \mid d(r_k, x) \leq d(r_j, x) \text{ for all } j = 1, \ldots, K, j \neq k \} \]
end;

compute new cluster centers:
for \( k = 1, \ldots, K \) do

\( r_k = \text{the vector mean of the points in } C_k \)
end;
end;
The *K-Means* Clustering Method

- Example

K=2

Arbitrarily choose K object as initial cluster center

Assign each objects to most similar center

Update the cluster means

Update the cluster means

reassign

reassign
Comments on the $K$-Means Method

- **Strength:** Relatively efficient: $O(tkn)$, where $n$ is # objects, $k$ is # clusters, and $t$ is # iterations. Normally, $k$, $t << n$.
  - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$

- **Comment:** Often terminates at a local optimum. The global optimum may be found using techniques such as: deterministic annealing and genetic algorithms

- **Weakness**
  - Applicable only when mean is defined, then what about categorical data?
  - Need to specify $k$, the number of clusters, in advance
  - Unable to handle noisy data and outliers
  - Not suitable to discover clusters with non-convex shapes
Variations of the $K$-Means Method

- A few variants of the $k$-means which differ in
  - Selection of the initial $k$ means
  - Dissimilarity calculations
  - Strategies to calculate cluster means

- Handling categorical data: $k$-modes (Huang’98)
  - Replacing means of clusters with modes
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters
  - A mixture of categorical and numerical data: $k$-prototype method
What *K-Means* is doing

- *k-means* attempts to minimize the within-cluster point scatter:

\[
W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} \sum d(x_i, x_{i'})
\]

- Note the total point scatter $T$ is constant. Thus, minimizing $W(C)$ is the same as maximizing $B(C)$:

\[
B(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i') \neq k} \sum d(x_i, x_{i'})
\]

since $T = W(C) + B(C)$
K-Means Clustering in R

kmeans(x, centers, iter.max=10)

x  A numeric matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns).

centers
Either the number of clusters or a set of initial cluster centers. If the first, a random set of rows in x are chosen as the initial centers.

iter.max
Hartigan’s Rule

When deciding on the number of clusters, Hartigan (1975, pp 90-91) suggests the following rough rule of thumb. If \( k \) is the result of \( k \)-means with \( k \) groups and \( k+1 \) is the result with \( k+1 \) groups, then it is justifiable to add the extra group when:

\[
\frac{\text{sum}(k\text{-withinss})}{\text{sum}(k+1\text{-withinss})-1} \times (\text{nrow}(x)-k-1)
\]

is greater than 10.
Example Data Generation

```r
library(MASS)
x1 <- mvrnorm(100, mu = c(2, 2), Sigma = matrix(c(1, 0, 0, 1), 2))
x2 <- mvrnorm(100, mu = c(-2, -2), Sigma = matrix(c(1, 0, 0, 1), 2))
x <- matrix(nrow = 200, ncol = 2)
x[1:100,] <- x1
x[101:200,] <- x2
plot(x[,1], x[,2])
```
\textbf{$k$-means Applied to our Data Set}

\begin{verbatim}
#Here we perform k=means clustering for a sequence of model sizes
x.km2<-kmeans(x,2)
x.km3<-kmeans(x,3)
x.km4<-kmeans(x,4)

plot(x[,1],x[,2],type="n")
text(x[,1],x[,2],labels=as.character(x.km2$cluster))
\end{verbatim}
The 3 term $k$-means solution
The 4 term $k$-means Solution
Determination of the Number of Clusters Using the Hartigan Criteria

\[
\begin{align*}
\text{> (sum(x.km3$withinss)/sum(x.km4$withinss)-1)*(200-3-1)} & \Rightarrow 23.08519 \\
\text{> (sum(x.km4$withinss)/sum(x.km5$withinss)-1)*(200-4-1)} & \Rightarrow 75.10246 \\
\text{> (sum(x.km5$withinss)/sum(x.km6$withinss)-1)*(200-5-1)} & \Rightarrow -6.55368 \\
\text{> plot(x[,1],x[,2],type="n")} \\
\text{> text(x[,1],x[,2],labels=as.character(x.km5$cluster))}
\end{align*}
\]
$k=5$ Solution
What is the problem with the k-Means Method?

• The k-means algorithm is sensitive to outliers!
  – Since an object with an extremely large value may substantially distort the distribution of the data.

• K-Medoids: Instead of taking the mean value of the object in a cluster as a reference point, medoids can be used, which is the most centrally located object in a cluster.
The *K-Medoids* Clustering Method

- Find *representative* objects, called *medoids*, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the quality of the resulting clustering
  - *PAM* works effectively for small data sets, but does not scale well for large data sets
- *CLARA* (Kaufmann & Rousseeuw, 1990)
- *CLARANS* (Ng & Han, 1994): Randomized sampling
- Focusing + spatial data structure (Ester et al., 1995)
PAM (Partitioning Around Medoids) (1987)

• PAM (Kaufman and Rousseeuw, 1987), built in Splus

• Use real object to represent the cluster
  
  – Select \(k\) representative objects arbitrarily
  
  – For each pair of non-selected object \(b\) and selected object \(i\), calculate the total swapping cost \(TC_{ib}\)
  
  – For each pair of \(i\) and \(b\),
    
    • If \(TC_{ib} < 0\), \(i\) is replaced by \(b\)
    
    • Then assign each non-selected object to the most similar representative object
  
  – repeat steps 2-3 until there is no change
PAM Clustering: Total swapping cost $TC_{ih} = \sum_j C_{jih}$

$i$ & $t$ are the current mediods

$C_{jih} = d(j, h) - d(j, i)$

$C_{jih} = 0$

$C_{jih} = d(j, t) - d(j, i)$

$C_{jih} = d(j, h) - d(j, t)$
What is the problem with PAM?

• Pam is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean.

• Pam works efficiently for small data sets but does not scale well for large data sets.
  – \( O(k(n-k)^2) \) for each iteration

  where \( n \) is # of data, \( k \) is # of clusters

♫ Sampling based method,

CLARA(Clustering LARge Applications)
CLARA (Clustering Large Applications) (1990)

- CLARA (Kaufmann and Rousseeuw in 1990)
  - Built in statistical analysis packages, such as R
- It draws multiple samples of the data set, applies PAM on each sample, and gives the best clustering as the output
- **Strength**: deals with larger data sets than PAM
- **Weakness:**
  - Efficiency depends on the sample size
  - A good clustering based on samples will not necessarily represent a good clustering of the whole data set if the sample is biased
K-Means Example

- Given: \{2,4,10,12,3,20,30,11,25\}, k=2
- Randomly assign means: \(m_1=3, m_2=4\)
- Solve for the rest ....
- Similarly try for k-medoids
K-Means Example

• Given: \{2,4,10,12,3,20,30,11,25\}, k=2
• Randomly assign means: \(m_1=3, m_2=4\)
• \(K_1=\{2,3\}, K_2=\{4,10,12,20,30,11,25\}\), \(m_1=2.5, m_2=16\)
• \(K_1=\{2,3,4\}, K_2=\{10,12,20,30,11,25\}\), \(m_1=3, m_2=18\)
• \(K_1=\{2,3,4,10\}, K_2=\{12,20,30,11,25\}\), \(m_1=4.75, m_2=19.6\)
• \(K_1=\{2,3,4,10,11,12\}, K_2=\{20,30,25\}\), \(m_1=7, m_2=25\)
• Stop as the clusters with these means are the same.
Hierarchical Clustering

• Agglomerative versus divisive

• Generic Agglomerative Algorithm:

  
  for \( i = 1, \ldots, n \) let \( C_i = \{x(i)\} \);
  while there is more than one cluster left do
    let \( C_i \) and \( C_j \) be the clusters
    minimizing the distance \( D(C_k, C_h) \) between any two clusters;
    \( C_i = C_i \cup C_j \);
    remove cluster \( C_j \);
  end;

• Computing complexity \( O(n^2) \)
Distance Between Clusters

Between-cluster dissimilarity measures
(Linkage for hierarchical clustering)

- Single (minimum): Elongated clusters, sensitive to outliers
- Complete (maximum): Compact clusters, sensitive to outliers
- Distance between centroids
- Average (Mean) linkage: In between, less sensitive to outliers
Cluster Summary Parameters

\[
\text{centroid } = C_m = \frac{\sum_{i=1}^{N} (t_{mi})}{N}
\]

\[
\text{radius } = R_m = \sqrt{\frac{\sum_{i=1}^{N} (t_{mi} - C_m)^2}{N}}
\]

\[
\text{diameter } = D_m = \sqrt{\frac{\sum_{i=1}^{N} \sum_{j=1}^{N} (t_{mi} - t_{mj})^2}{(N)(N - 1)}}
\]
Height of the cross-bar shows the change in within-cluster SS.

Agglomerative
Hierarchical Clustering

Single link/Nearest neighbor ("chaining")

\[ D_{sl}(C_i, C_j) = \min_{x,y} \{d(x, y) \mid x \in C_i, y \in C_j \} \]

Complete link/Furthest neighbor (~clusters of equal vol.)

\[ D_{fl}(C_i, C_j) = \max_{x,y} \{d(x, y) \mid x \in C_i, y \in C_j \} \]

- centroid measure (distance between centroids)
- group average measure (average of pairwise distances)
- Ward’s \([SS(C_i) + SS(C_j) - SS(C_{i+j})]\)
Figure 9.9: Dendrogram of the single link method applied to the data in figure 9.7.
Hierarchical Clustering in R

- Assuming that you have read your data into a matrix called `data.mat` then first you must compute the interpoint distance matrix using the `dist` function

```r
library(mva)
data.dist<- dist(data.mat)
```

- Next hierarchical clustering is accomplished with a call to `hclust`
hclust

• It computes complete linkage clustering by default

• Using the `method="connected"` we obtain single linkage clustering

• Using the `method = "average"` we obtain average clustering
plclust and cutree

• **plot** is used to plot our dendrogram

• **cutree** is used to examine the groups that are given at a given cut level
Computing the Distance Matrix

dist(x, metric = "euclidean")

metric = character string specifying the distance metric to be used.

The currently available options are "euclidean", "maximum", "manhattan", and "binary". Euclidean distances are root sum-of-squares of differences, "maximum" is the maximum difference, "manhattan" is the of absolute differences, and "binary" is the proportion of non-that two vectors do not have in common (the number of occurrences of a zero and a one, or a one and a zero divided by the number of times at least one vector has a one).
Example Distance Matrix Computation

> x.dist<-dist(x)

> length(x.dist)

[1] 19900
hclust

hclust(d, method = "complete", members=NULL)

d  a dissimilarity structure as produced by dist.

method  the agglomeration method to be used. This should be (an unambiguous abbreviation of) one of "ward", "single", "complete", "average", "median" or "centroid".
Values Returned by `hclust`

- **merge**: an \( n-1 \) by 2 matrix. Row \( i \) of `merge` describes the merging of clusters at step \( i \) of the clustering. If an element \( j \) in the row is negative, then observation \( -j \) was merged at this stage. If \( j \) is positive then the merge was with the cluster formed at the (earlier) stage \( j \) of the algorithm. Thus negative entries in `merge` indicate agglomerations of singletons, and positive entries indicate agglomerations of non-singletons.

- **height**: a set of \( n-1 \) non-decreasing real values. The clustering `height`: that is, the value of the criterion associated with the clustering `method` for the particular agglomeration.

- **order**: a vector giving the permutation of the original observations suitable for plotting, in the sense that a cluster plot using this ordering and matrix `merge` will not have crossings of the branches.

- **labels**: labels for each of the objects being clustered.

- **call**: the call which produced the result.

- **method**: the cluster method that has been used.

- **dist.method**: the distance that has been used to create \( d \) (only returned if the distance object has a "method" attribute).
Complete Linkage Clustering with \texttt{hclust}

> \texttt{plot(hclust(x.dist))}
Single Linkage Clustering with `hclust`

```r
> plot(hclust(x.dist, method="single"))
```
Average Linkage Clustering with `hclust`

```r
plot(hclust(x.dist, method="average"))
```
Pruning Our Tree

cutree(tree, k = NULL, h = NULL)

tree a tree as produced by \texttt{hclust}. \texttt{cutree()} only
  expects a list with components \texttt{merge}, \texttt{height},
  and \texttt{labels}, of
  appropriate content each.

k an integer scalar or vector with the desired number of groups

h numeric scalar or vector with heights where the tree should be
cut.

At least one of k or h must be specified, k overrides h if
both are given.

Values Returned

cutree returns a vector with group memberships if k or h are scalar,
otherwise a matrix with group memberships is returned where
each column
 corresponds to the elements of k or h, respectively (which are also
 used as column names).
Example Pruning

> x.cl2 <- cutree(hclust(x.dist), k=2)

> x.cl2[1:10]
  [1] 1 1 1 1 1 1 1 1 1 1

> x.cl2[190:200]
  [1] 2 2 2 2 2 2 2 2 2 2
Identifying the Number of Clusters

• As indicated previously we really have no way of identify the true cluster structure unless we have divine intervention

• In the next several slides we present some well-known methods
Method of Mojena

• Select the number of groups based on the first stage of the dendogram that satisfies

\[ \alpha_{j+1} > \bar{\alpha} + k s_{\alpha} \]

• The \( \alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_{n-1} \) are the fusion levels corresponding to stages with \( n, n-1, \ldots, 1 \) clusters. \( \bar{\alpha} \) and \( s_{\alpha} \) are the mean and unbiased standard deviation of these fusion levels and \( k \) is a constant.

• Mojena (1977) \( 2.75 < k < 3.5 \)

• Milligan and Cooper (1985) \( k=1.25 \)
Method of Mojena Applied to Our Data Set - I

```r
> x.clfl <- hclust(x.dist)$height
# assign the fusion levels

> x.clm <- mean(x.clfl)
# compute the means

> x.cls <- sqrt(var(x.clfl))
# compute the standard deviation

> print((x.clfl - x.clm) / x.cls)
# output the results for comparison with k
```
Method of Mojena Applied to Our Data Set - II

> print((x.clfl-x.clm)/x.cls)

[1] -0.609317763 -0.595451243 -0.591760600 -0.590785339 -0.590132779
[6] -0.587620192 -0.574381404 -0.570288225 -0.560984067 -0.559183861
.
.
.
[186] 1.189406923 1.391764160 1.582611713 1.731697165 1.817821995
[191] 2.056156268 2.057782017 2.534517541 2.606030029 3.157604485
[196] 3.473036668 4.028366785 4.385419127 8.368682725
> print(x.clfl[196])
[1] 5.131528
Visualizing Our Cluster Structure

```r
> x.clmojena <- cutree(hclust(x.dist), h = x.clfl[196])
> plot(x[,1], x[,2], type = "n")
> text(x[,1], x[,2], labels = as.character(x.clmojena))
```
Visualizing Our Cluster Structure
(Cutting the Tree Higher)

> x.cllastsplit<-cutree(hclust(x.dist), h=x.clfl[198])
## Single-Link Agglomerative Example

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

![Diagram of single-link agglomerative clustering]

Threshold of 1 2 3 4 5
Clustering Example

a) Single Link

b) Complete Link

b) Average Link
AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Use the Single-Link method and the dissimilarity matrix.
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion

![Diagram](attachment:image.png)
DIANA (Divisive Analysis)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Inverse order of AGNES
- Eventually each node forms a cluster on its own
Clustering Market Basket Data: ROCK

- **ROCK**: Robust Clustering using links, by S. Guha, R. Rastogi, K. Shim (ICDE’99).
  - Use links to measure similarity/proximity
  - Not distance based
  - Computational complexity: \( O(n^2 + nm_m m_a + n^2 \log n) \)

- Basic ideas:
  - Similarity function and neighbors:
    \[
    Sim(T_1, T_2) = \frac{|T_1 \cap T_2|}{|T_1 \cup T_2|}
    \]
  - Let \( T_1 = \{1,2,3\} \), \( T_2 = \{3,4,5\} \)
    \[
    Sim(T_1, T_2) = \frac{|\{3\}|}{|\{1,2,3,4,5\}|} = \frac{1}{5} = 0.2
    \]
Rock: Algorithm

• Links: The number of common neighbours for the two points.

\{1,2,3\}, \{1,2,4\}, \{1,2,5\}, \{1,3,4\}, \{1,3,5\}, \{1,4,5\}, \{2,3,4\}, \{2,3,5\}, \{2,4,5\}, \{3,4,5\}

\{1,2,3\} \leftrightarrow 3 \leftrightarrow \{1,2,4\}

• Algorithm
  – Draw random sample
  – Cluster with links
  – Label data in disk

Nbrs have sim > threshold
CLIQUE (Clustering In QUEst)

- Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD’98).
- Automatically identifying subspaces of a high dimensional data space that allow better clustering than original space.
- CLIQUE can be considered as both density-based and grid-based.
  - It partitions each dimension into the same number of equal length interval.
  - It partitions an m-dimensional data space into non-overlapping rectangular units.
  - A unit is dense if the fraction of total data points contained in the unit exceeds the input model parameter.
  - A cluster is a maximal set of connected dense units within a subspace.
CLIQUE: The Major Steps

- Partition the data space and find the number of points that lie inside each unit of the partition.

- Identify the dense units using the Apriori principle.

- Determine connected dense units in all subspaces of interests.

- Generate minimal description for the clusters
  - Determine maximal regions that cover a cluster of connected dense units for each cluster
  - Determination of minimal cover for each cluster
Example

Figure 1: Identification of clusters in subspaces (projections) of the original data space.
\[ \tau = 2 \]
Strength andWeakness of

**CLIQUE**

- **Strength**
  - It *automatically* finds subspaces of the highest dimensionality such that high density clusters exist in those subspaces
  - It is *insensitive* to the order of records in input and does not presume some canonical data distribution
  - It scales *linearly* with the size of input and has good scalability as the number of dimensions in the data increases

- **Weakness**
  - The accuracy of the clustering result may be degraded at the expense of simplicity of the method
Model-based Clustering

\[ f(x) = \sum_{k=1}^{K} \pi_k f_k(x; \theta_k) \]
Advantages of the Probabilistic Approach

• Provides a distributional description for each component
• For each observation, provides a K-component vector of probabilities of class membership
• Method can be extended to data that are not in the form of p-dimensional vectors, e.g., mixtures of Markov models
• Can find clusters-within-clusters
• Can make inference about the number of clusters
• But... its computationally somewhat costly
Mixtures of \{Sequences, Curves, \ldots\}

\[
p(D_i) = \sum_{k=1}^{K} p(D_i | c_k) \alpha_k
\]

**Generative Model**

- select a component $c_k$ for individual $i$
- generate data according to $p(D_i | c_k)$
  - $p(D_i | c_k)$ can be very general
  - e.g., sets of sequences, spatial patterns, etc

[Note: given $p(D_i | c_k)$, we can define an EM algorithm]
Application 1: Web Log Visualization

(Cadez, Heckerman, Meek, Smyth, KDD 2000)

• MSNBC Web logs
  – 2 million individuals per day
  – different session lengths per individual
  – difficult visualization and clustering problem

• WebCanvas
  – uses mixtures of SFSMs to cluster individuals based on their observed sequences
  – software tool: EM mixture modeling + visualization
Example: Mixtures of SFSMs

Simple model for traversal on a Web site
(equivalent to first-order Markov with end-state)

Generative model for large sets of Web users
- different behaviors $\Leftrightarrow$ mixture of SFSMs

EM algorithm is quite simple: weighted counts
Figure 3: Pairs plot showing the clinical classification of the diabetes data. The symbols have the following interpretation: squares – normal; circles – chemical diabetes; triangles – overt diabetes.

- glucose: plasma glucose response to oral glucose,
- insulin: plasma insulin response to oral glucose,
- sspg: degree of insulin resistance.
Figure 1: A projection of the three-group classification of the diabetes data from Reaven and Miller [56] using single link or nearest neighbor, standard k-means, and the unconstrained model-based approach. Filled symbols represent misclassified observations.
Figure 4: The plot on the left shows the Bayesian Information Criterion (BIC) for model-based methods applied to the diabetes data. The first local (also global) maximum occurs for the unconstrained model with three clusters. The plot on the right depicts the uncertainty of the classification produced by the best model (unconstrained, 3 clusters) indicated by the BIC. The symbols have the following interpretation: dots $< 0.1$; open circles $\geq 0.1$ and $< 0.2$; filled circles $\geq 0.2$. 
BIRCH (1996)

- Birch: Balanced Iterative Reducing and Clustering using Hierarchies, by Zhang, Ramakrishnan, Livny (SIGMOD’96)

- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering
  - Phase 1: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)
  - Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree

- Scales linearly: finds a good clustering with a single scan and improves the quality with a few additional scans

- Weakness: handles only numeric data, and sensitive to the order of the data record.
Clustering Feature Vector

Clustering Feature: $CF = (N, LS, SS)$

$N$: Number of data points

$LS$: $\sum_{i=1}^{N} = \overrightarrow{X_i}$

$SS$: $\sum_{i=1}^{N} = \overrightarrow{X_i}^2$

$R = \left( \frac{\sum_i (X_i - \overline{X})^2}{N} \right)^{\frac{1}{2}}$ (3,4)

$D = \left( \frac{\sum_i \sum_j (X_i - X_j)^2}{N(N-1)} \right)^{\frac{1}{2}}$ (2,6)

$\overrightarrow{CF} = (5, (16,30),(54,190))$ (4,5)

(4,7)

(3,8)
Han & Kamber

**CF Tree**

Branching Factor (B) = 7
Max Leaf Size (L) = 6

Root

<table>
<thead>
<tr>
<th>CF₁</th>
<th>CF₂</th>
<th>CF₃</th>
<th>......</th>
<th>CF₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>child₁</td>
<td>child₂</td>
<td>child₃</td>
<td>......</td>
<td>child₆</td>
</tr>
</tbody>
</table>

Non-leaf node

<table>
<thead>
<tr>
<th>CF₁</th>
<th>CF₂</th>
<th>CF₃</th>
<th>......</th>
<th>CF₇</th>
</tr>
</thead>
<tbody>
<tr>
<td>child₁</td>
<td>child₂</td>
<td>child₃</td>
<td>......</td>
<td>child₇</td>
</tr>
</tbody>
</table>

Leaf node

<table>
<thead>
<tr>
<th>prev</th>
<th>CF₁</th>
<th>CF₂</th>
<th>......</th>
<th>CF₆</th>
<th>next</th>
</tr>
</thead>
</table>

Leaf node

<table>
<thead>
<tr>
<th>prev</th>
<th>CF₁</th>
<th>CF₂</th>
<th>......</th>
<th>CF₄</th>
<th>next</th>
</tr>
</thead>
</table>
Insertion Into the CF Tree

- Start from the root and recursively descend the tree choosing closest child node at each step.
- If some leaf node entry can absorb the entry (i.e. $T_{\text{new}} < T$), do it
- Else, if space on leaf, add new entry to leaf
- Else, split leaf using farthest pair as seeds and redistributing remaining entries (may need to split parents)
- Also include a merge step
The \( k \)-modes algorithm consists of the following steps (refer to (Huang 1997) for the detailed description of the algorithm):

1. Select \( k \) initial modes, one for each cluster.
2. Allocate an object to the cluster whose mode is the nearest to it according to \( d \). Update the mode of the cluster after each allocation according to the Theorem.
3. After all objects have been allocated to clusters, retest the dissimilarity of objects against the current modes. If an object is found such that its nearest mode belongs to another cluster rather than its current one, reallocate the object to that cluster and update the modes of both clusters.
4. Repeat 3 until no object has changed clusters after a full cycle test of the whole data set.
Let $X$, $Y$ be two categorical objects described by $m$ categorical attributes. The dissimilarity measure between $X$ and $Y$ can be defined by the total mismatches of the corresponding attribute categories of the two objects. The smaller the number of mismatches is, the more similar the two objects. Formally,

$$d(X, Y) = \sum_{j=1}^{m} \delta(x_j, y_j)$$

(2)

where

$$\delta(x_j, y_j) = \begin{cases} 0 & (x_j = y_j) \\ 1 & (x_j \neq y_j) \end{cases}$$

(3)
\[ \tau = 3 \]
**Typical k-medoids algorithm (PAM)**

- **Arbitrary choose** $k$ object as initial medoids
- **Assign each** remaining object to nearest medoids
- **Randomly select a** nonmedoid object $O_{\text{random}}$
- **Do loop**
  - **Until no** change
  - Swapping $O$ and $O_{\text{random}}$
    - If quality is improved.
  - **Compute total cost of swapping**
  - **Total Cost = 20**

**K=2**
Desiderata for Clustering in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- High dimensionality
- Interpretability and usability
Measure the Quality of Clustering

- Dissimilarity/Similarity metric: Similarity is expressed in terms of a distance function, which is typically metric: \( d(i, j) \)
- There is a separate “quality” function that measures the “goodness” of a cluster.
- The definitions of distance functions are usually very different for interval-scaled, boolean, categorical, and ordinal variables.
- Weights can be associated with different variables based on applications and data semantics.
- It is hard to define “similar enough” or “good enough”
  - the answer is typically highly subjective.
Distance Between Clusters

- **Single Link**: smallest distance between points
- **Complete Link**: largest distance between points
- **Average Link**: average distance between points
- **Centroid**: distance between centroids