Gene Expression Profiles

• we’ll assume we have a 2D matrix of gene expression measurements
  – rows represent genes
  – columns represent different experiments, time points, individuals etc. (what we can measured using one* microarray)
• we’ll refer to individual rows or columns as profiles
  – a row is a profile for a gene

* Depending on the number of genes being considered, we might actually use several arrays per experiment, time point, individual.
Expression Profile Example

- rows represent genes
- columns represent people with leukemia

Expression Profile Example

- rows represent genes
- columns represent time points in a given experiment
Task Definition: Clustering Gene Expression Profiles

- given: expression profiles for a set of genes or experiments/individuals/time points (whatever columns represent)
- do: organize profiles into clusters such that
  - instances in the same cluster are highly similar to each other
  - instances from different clusters have low similarity to each other

Motivation for Clustering

- exploratory data analysis
  - understanding general characteristics of data
  - visualizing data
- generalization
  - infer something about an instance (e.g. a gene) based on how it relates to other instances
- everyone else is doing it
The Clustering Landscape

- there are many different clustering algorithms
- they differ along several dimensions
  - hierarchical vs. partitional (flat)
  - hard (no uncertainty about which instances belong to a cluster) vs. soft clusters
  - disjunctive (an instance can belong to multiple clusters) vs. non-disjunctive
  - deterministic (same clusters produced every time for a given data set) vs. stochastic
  - distance (similarity) measure used

Distance/Similarity Measures

- many clustering methods employ a distance (similarity) measure to assess the distance between
  - a pair of instances
  - a cluster and an instance
  - a pair of clusters
- given a distance value, it is straightforward to convert it into a similarity value
  \[ \text{sim}(x, y) = \frac{1}{1 + \text{dist}(x, y)} \]
- not necessarily straightforward to go the other way
- we’ll describe our algorithms in terms of distances
Distance Metrics

- properties of metrics
  \[ \text{dist}(x_i, x_j) \geq 0 \]
  \[ \text{dist}(x_i, x_i) = 0 \]
  \[ \text{dist}(x_i, x_j) = \text{dist}(x_j, x_i) \]
  \[ \text{dist}(x_i, x_j) \leq \text{dist}(x_i, x_k) + \text{dist}(x_k, x_j) \]

- some distance metrics
  - Manhattan
    \[ \text{dist}(x_i, x_j) = \sum_e |x_{i,e} - x_{j,e}| \]
  - Euclidean
    \[ \text{dist}(x_i, x_j) = \sqrt{\sum_e (x_{i,e} - x_{j,e})^2} \]

\( e \) ranges over the individual measurements for \( x_i \) and \( x_j \)

Hierarchical Clustering:
A Dendogram

leaves represent instances (e.g. genes)

height of bar indicates degree of distance within cluster
Hierarchical Clustering of Expression Data

Hierarchical Clustering

- can do top-down (divisive) or bottom-up (agglomerative)
- in either case, we maintain a matrix of distance (or similarity) scores for all pairs of
  - instances
  - clusters (formed so far)
  - instances and clusters
Bottom-Up Hierarchical Clustering

given: a set \( X = \{x_1, \ldots, x_n\} \) of instances

for \( i := 1 \) to \( n \) do

\( c_i := \{x_i\} \) /* each object is initially its own cluster, and a leaf in tree */

\( C := \{c_1, \ldots, c_n\} \)

\( j := n \)

while \( |C| > 1 \)

\( j := j + 1 \)

\( (c_a, c_b) := \arg\min_{(c_u, c_v)} \text{dist}(c_u, c_v) \) /* find least distant pair in \( C \) */

\( c_j = c_a \cup c_b \) /* create a new cluster for pair */

add a new node \( j \) to the tree joining \( a \) and \( b \)

\( C := C \setminus \{c_a, c_b\} \cup \{c_j\} \)

return tree with root node \( j \)

---

Haven’t We Already Seen This?

• this algorithm is very similar to UPGMA and neighbor joining; there are some differences

• what tree represents
  – phylogenetic inference: tree represents hypothesized sequence of evolutionary events; internal nodes represent hypothetical ancestors
  – clustering: inferred tree represents similarity of instances; internal nodes don’t represent ancestors

• form of tree
  – UPGMA: rooted tree
  – neighbor joining: unrooted
  – hierarchical clustering: rooted tree

• how distances among clusters are calculated
  – UPGMA: average link
  – neighbor joining: based on additivity
  – hierarchical clustering: various
Distance Between Two Clusters

- the distance between two clusters can be determined in several ways
  - single link: distance of two most similar instances
    \[ \text{dist}(c_u, c_v) = \min \{ \text{dist}(a, b) \mid a \in c_u, b \in c_v \} \]
  - complete link: distance of two least similar instances
    \[ \text{dist}(c_u, c_v) = \max \{ \text{dist}(a, b) \mid a \in c_u, b \in c_v \} \]
  - average link: average distance between instances
    \[ \text{dist}(c_u, c_v) = \frac{1}{|c_u| \cdot |c_v|} \sum_{a \in c_u} \sum_{b \in c_v} \text{dist}(a, b) \]

Complete-Link vs. Single-Link Distances

complete link

single link
Computational Complexity

- The naïve implementation of hierarchical clustering has $O(n^3)$ time complexity, where $n$ is the number of instances
  - Computing the initial distance matrix takes $O(n^2)$ time
  - There are $O(n)$ merging steps
  - On each step, we have to update the distance matrix $O(n)$ and select the next pair of clusters to merge $O(n^2)$

- For single-link clustering, we can update and pick the next pair in $O(n)$ time, resulting in an $O(n^2)$ algorithm
- For complete-link and average-link, we can do these steps in $O(n \log n)$ time resulting in an $O(n^2 \log n)$ method

- See http://www-csli.stanford.edu/~schuetze/completelink.html for an improved and corrected discussion of the computational complexity of hierarchical clustering
Updating Distances Efficiently

- if we just merged $C_u$ and $C_v$ into $C_j$, we can determine distance to each other cluster $C_k$ as follows
  - single link:
    $$\text{dist}(c_j, c_k) = \min\{\text{dist}(c_u, c_k), \text{dist}(c_v, c_k)\}$$
  - complete link:
    $$\text{dist}(c_j, c_k) = \max\{\text{dist}(c_u, c_k), \text{dist}(c_v, c_k)\}$$
  - average link:
    $$\text{dist}(c_j, c_k) = \frac{|c_u| \times \text{dist}(c_u, c_k) + |c_v| \times \text{dist}(c_v, c_k)}{|c_u| + |c_v|}$$

Partitional Clustering

- divide instances into disjoint clusters
  - flat vs. tree structure
- key issues
  - how many clusters should there be?
  - how should clusters be represented?
Partitional Clustering Example

- we can always generate a partitional clustering from a hierarchical clustering by “cutting” the tree at some level.
**K-Means Clustering**

- assume our instances are represented by vectors of real values
- put $k$ cluster centers in same space as instances
- each cluster is represented by a vector $\tilde{f}_j$
- consider an example in which our vectors have 2 dimensions

![Diagram of K-Means Clustering]

**K-Means Clustering**

- each iteration involves two steps
  - assignment of instances to clusters
  - re-computation of the means

![Assignment and re-computation of means diagrams]
**K-Means Clustering: Updating the Means**

- for a set of instances that have been assigned to a cluster $c_j$, we re-compute the mean of the cluster as follows

$$
\mu(c_j) = \frac{\sum_{\bar{x}_i \in c_j} \bar{x}_i}{|c_j|}
$$

---

**K-Means Clustering**

given: a set $X = \{\bar{x}_1,...,\bar{x}_n\}$ of instances

select $k$ initial cluster centers $\bar{f}_1,...,\bar{f}_k$

while stopping criterion not true do
  for all clusters $c_j$ do
    // determine which instances are assigned to this cluster
    $$
c_j = \{\bar{x}_i | \forall \bar{f}_i \text{ dist}(\bar{x}_i, \bar{f}_j) < \text{dist}(\bar{x}_i, \bar{f}_i)\}
$$
  for all means $\bar{f}_j$ do
    // update the cluster center
    $$
    \bar{f}_j = \mu(c_j)
    $$
Given the following 4 instances and 2 clusters initialized as shown. Assume the distance function is \( \text{dist}(x_i, x_j) = \sum_{e} |x_{i,e} - x_{j,e}| \)

### K-means Clustering Example

(Continued)

\[
f_1 = \left\{ \frac{4+4+6}{3}, \frac{1+3+2}{3} \right\} = \{4.67, 2\}
\]

\[
f_2 = \left\{ \frac{8}{1}, \frac{8}{1} \right\} = \{8, 8\}
\]

Assignments remain the same, so the procedure has converged.
EM Clustering

• in k-means as just described, instances are assigned to one and only one cluster
• we can do “soft” k-means clustering via an *Expectation Maximization* (EM) algorithm
  – each cluster represented by a distribution (e.g. a Gaussian)
  – E step: determine how likely is it that each cluster “generated” each instance
  – M step: adjust cluster parameters to maximize likelihood of instances

Representation of Clusters

• in the EM approach, we’ll represent each cluster using an \(m\)-dimensional multivariate Gaussian

\[
N_j(\bar{x}_i) = \frac{1}{\sqrt{(2\pi)^m |\Sigma_j|}} \exp\left[ -\frac{1}{2}(\bar{x}_i - \bar{\mu}_j)^T \Sigma_j^{-1}(\bar{x}_i - \bar{\mu}_j) \right]
\]

where

\(\bar{\mu}_j\) is the mean of the Gaussian
\(\Sigma_j\) is the covariance matrix

this is a representation of a Gaussian in a 2-D space
EM Clustering

- the EM algorithm will try to set the parameters of the Gaussians, $\Theta$, to maximize the log likelihood of the data, $X$

$$\log \text{likelihood}(X \mid \Theta) = \log \prod_{i=1}^{n} \Pr(\tilde{x}_i)$$

$$= \log \prod_{i=1}^{n} \sum_{j=1}^{k} N_j(\tilde{x}_i)$$

$$= \sum_{i=1}^{n} \log \sum_{j=1}^{k} N_j(\tilde{x}_i)$$

EM Clustering

- the parameters of the model, $\Theta$, include the means, the covariance matrix and sometimes prior weights for each Gaussian
- here, we’ll assume that the covariance matrix and the prior weights are fixed; we’ll focus just on setting the means
EM Clustering: Hidden Variables

• on each iteration of $k$-means clustering, we had to assign each instance to a cluster
• in the EM approach, we’ll use hidden variables to represent this idea
• for each instance $\tilde{x}_i$ we have a set of hidden variables $z_{ij}, \ldots, z_{ik}$
• we can think of $z_{ij}$ as being 1 if $\tilde{x}_i$ is a member of cluster $j$ and 0 otherwise

EM Clustering: the E-step

• recall that $z_{ij}$ is a hidden variable which is 1 if $N_j$ generated $\tilde{x}_i$ and 0 otherwise
• in the E-step, we compute $h_{ij}$, the expected value of this hidden variable

$$h_{ij} = E(z_{ij} \mid \tilde{x}_i) = \frac{N_j(\tilde{x}_i)}{\sum_{l=1}^{k} N_l(\tilde{x}_i)}$$
EM Clustering: the M-step

• given the expected values $h_{ij}$, we re-estimate the means of the Gaussians

$$\bar{\mu}_j' = \frac{\sum_{i=1}^{n} h_{ij} \bar{x}_i}{\sum_{i=1}^{n} h_{ij}}$$

• can also re-estimate the covariance matrix and prior weights, if we’re varying them

EM Clustering Example

Consider a one-dimensional clustering problem in which the data given are:

$x_1 = -4$
$x_2 = -3$
$x_3 = -1$
$x_4 = 3$
$x_5 = 5$

The initial mean of the first Gaussian is 0 and the initial mean of the second is 2. The Gaussians have fixed width; their density function is:

$$f(x, \mu) = \frac{1}{\sqrt{8\pi}} e^{-\frac{1}{2} \left( \frac{x-\mu}{\sigma} \right)^2}$$

where $\mu$ denotes the mean (center) of the Gaussian.
EM Clustering Example

\[ f(x, \mu) = \frac{1}{\sqrt{8\pi}} e^{-\frac{1}{2} \left( \frac{x-\mu}{2} \right)^2} \]

- \( f(-4, \mu_1) = \frac{1}{\sqrt{8\pi}} e^{-\frac{1}{2} \left( \frac{-4-0}{2} \right)^2} = .0269 \)
- \( f(-4, \mu_2) = .0022 \)
- \( f(-3, \mu_1) = .0646 \)
- \( f(-3, \mu_2) = .00874 \)
- \( f(-1, \mu_1) = .176 \)
- \( f(-1, \mu_2) = .0646 \)
- \( f(3, \mu_1) = .0646 \)
- \( f(3, \mu_2) = .176 \)
- \( f(5, \mu_1) = .00874 \)
- \( f(5, \mu_2) = .0646 \)

EM Clustering Example: E Step

\[
\begin{align*}
    h_1 &= \frac{f(x_1, \mu_1)}{f(x_1, \mu_1) + f(x_1, \mu_2)} = \frac{.0269}{.0269 + .0022} = .0269 \\
    h_2 &= \frac{f(x_1, \mu_2)}{f(x_1, \mu_1) + f(x_1, \mu_2)} = \frac{.00874}{.0269 + .00874} = .00874 \\
    h_3 &= \frac{f(x_2, \mu_1)}{f(x_2, \mu_1) + f(x_2, \mu_2)} = \frac{.176}{.176 + .0646} = .176 \\
    h_4 &= \frac{f(x_2, \mu_2)}{f(x_2, \mu_1) + f(x_2, \mu_2)} = \frac{.0646}{.176 + .0646} = .0646 \\
    h_5 &= \frac{f(x_3, \mu_1)}{f(x_3, \mu_1) + f(x_3, \mu_2)} = \frac{.0646}{.0646 + .176} = .0646 \\
    h_6 &= \frac{f(x_3, \mu_2)}{f(x_3, \mu_1) + f(x_3, \mu_2)} = \frac{.176}{.0646 + .176} = .176 \\
    h_7 &= \frac{f(x_4, \mu_1)}{f(x_4, \mu_1) + f(x_4, \mu_2)} = \frac{.00874}{.00874 + .0646} = .00874 \\
    h_8 &= \frac{f(x_4, \mu_2)}{f(x_4, \mu_1) + f(x_4, \mu_2)} = \frac{.0646}{.00874 + .0646} = .0646 \\
\end{align*}
\]
EM Clustering Example: M-step

\[ \mu_1 = \frac{\sum h_{1i} \times x_i}{\sum h_{1i}} = \frac{-4 \times .924 + -3 \times .881 + -1 \times .732 + 3 \times .268 + 5 \times 1.119}{.924 + .881 + .732 + .268 + 1.119} = -1.94 \]

\[ \mu_2 = \frac{\sum h_{12} \times x_i}{\sum h_{12}} = \frac{-4 \times .076 + -3 \times 1.119 + -1 \times .268 + 3 \times 3.732 + 5 \times .191}{.076 + 1.119 + .268 + 3.732 + .191} = 3.39 \]

- here we’ve shown just one step of the EM procedure

- we would continue the E- and M-steps until convergence
EM and $K$-Means Clustering

- both will converge to a local maximum
- both are sensitive to initial positions (means) of clusters
- have to choose value of $k$ for both

The CLICK Algorithm

- Sharan & Shamir, ISMB 2000
- instances to be clustered (e.g. genes) represented as vertices in a graph
- weighted, undirected edges represent similarity of instances
CLICK: How Do We Get Graph?

• assume pairwise similarity values are normally distributed

\[
N(\mu_T, \sigma_T^2) \quad \text{for mates (instances in same “true” cluster)}
\]

\[
N(\mu_F, \sigma_F^2) \quad \text{for non-mates}
\]

• estimate the parameters of these distributions and \( \Pr(mates) \) (the prob that two randomly chosen instances are mates) from the data

CLICK: How Do We Get Graph?

• let \( f(S_{ij} \mid i, j \text{ are mates}) \) be the probability density function for similarity values when \( i \) and \( j \) are mates

• then set the weight of an edge by:

\[
w_{ij} = \log \frac{\Pr(mates) f(S_{ij} \mid i, j \text{ are mates})}{(1 - \Pr(mates)) f(S_{ij} \mid i, j \text{ are non-mates})}
\]

• prune edges with weights < specified non-negative threshold \( t \)
The Basic CLICK Algorithm

BasicCLICK (G) :
if $V(G) = \{v\}$ then /* does graph have just one vertex? */
    move $v$ to singleton set $R$
else if $G$ is a kernel /* does graph satisfy stopping criterion? */
    return $V(G)$
else /* partition graph, call recursively */
    $(H, \overline{H}) \leftarrow \text{MinWeightCut}(G)$
    BasicCLICK ($H$)
    BasicCLICK ($\overline{H}$)

Minimum Weight Cuts

- a cut of a graph is a subset of edges whose removal disconnects the graph
- a minimum weight cut is the cut with the smallest sum of edge weights
- can be found efficiently
Deciding When a Subgraph Represents a Kernel

- we can test a cut \( C \) against two hypotheses
  \( H_0^C : C \) contains only edges between non-mates
  \( H_1^C : C \) contains only edges between mates

- we can then score \( C \) by

\[
\log \frac{\Pr(H_1^C \mid C)}{\Pr(H_0^C \mid C)}
\]

- if we assume a complete graph, the minimum weight cut algorithm finds a cut that minimizes this ratio, i.e.

\[
\text{weight}(C) = \log \frac{\Pr(H_1^C \mid C)}{\Pr(H_0^C \mid C)}
\]

- thus, we accept \( H_1^C \) and call \( G \) a kernel iff

\[
\text{weight}(C) > 0
\]
Deciding When a Subgraph Represents a Kernel

- but we don’t have a complete graph
- we call $G$ a kernel iff $\text{weight}(C) + \text{weight}^{\prime}(C) > 0$
  where $\text{weight}^{\prime}(C)$ approximates the contribution of missing edges

The Full CLICK Algorithm

- the basic CLICK algorithm produces kernels of clusters
- add two more operations
  - adoption: find singletons that are similar, and hence can be adopted by kernels
  - merge: merge similar clusters
The Full CLICK Algorithm

CLICK($G_N$):

\[ R \leftarrow N \]

while some change occurs do

BasicCLICK ($G_R$)

let $L$ be the set of kernels produced

let $R$ be the set of singletons produced

Adoption($L, R$)

Merge($L$)

Adoption($R$)

CLICK Experiment:
Fibroblast Serum Response Data

figure from: Sharan & Shamir, ISMB 2000
Evaluating Clustering Results

- given random data without any “structure”, clustering algorithms will still return clusters
- the gold standard: do clusters correspond to natural categories?
- do clusters correspond to categories we care about? (there are lots of ways to partition the world)

Evaluating Clustering Results

- some approaches
  - external validation
    - E.g. do genes clustered together have some common function?
  - internal validation
    - How well does clustering optimize intra-cluster similarity and inter-cluster dissimilarity?
  - relative validation
    - How does it compare to other clusterings using these criteria?
    - E.g. with a probabilistic method (such as EM) we can ask: how probable does held-aside data look as we vary the number of clusters.
Comments on Clustering

• there many different ways to do clustering; we ‘ve discussed just a few methods
• hierarchical clusters may be more informative, but they’re more expensive to compute
• clusterings are hard to evaluate in many cases