*Unsupervised vs Supervised Learning:*



Unsupervised Learning

* Most of this course focuses on *supervised learning* methods such as regression and classification.
* In that setting we observe both a set of features

*X*1*, X*2*, . . . , Xp* for each object, as well as a response or outcome variable *Y* . The goal is then to predict *Y* using *X*1*, X*2*, . . . , Xp*.

* Here we instead focus on *unsupervised learning*, we where observe only the features *X*1*, X*2*, . . . , Xp*. We are not interested in prediction, because we do not have an associated response variable *Y* .
* The goal is to discover interesting things about the measurements: is there an informative way to visualize the data? Can we discover subgroups among the variables or among the observations?
* We discuss two methods:
  + *principal components analysis*, a tool used for data visualization or data pre-processing before supervised techniques are applied, and
  + *clustering*, a broad class of methods for discovering unknown subgroups in data.
* Unsupervised learning is more subjective than supervised learning, as there is no simple goal for the analysis, such as prediction of a response.
* But techniques for unsupervised learning are of growing importance in a number of fields:
  + subgroups of breast cancer patients grouped by their gene expression measurements,
  + groups of shoppers characterized by their browsing and purchase histories,
  + movies grouped by the ratings assigned by movie viewers.
* It is often easier to obtain *unlabeled data* — from a lab instrument or a computer — than *labeled data*, which can require human intervention.
* For example it is difficult to automatically assess the overall sentiment of a movie review: is it favorable or not?
* PCA produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance, and are mutually uncorrelated.
* Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization.
* The *first principal component* of a set of features

*X*1*, X*2*, . . . , Xp* is the normalized linear combination of the features

*Z*1 = *φ*11*X*1 + *φ*21*X*2 + *. . .* + *φp*1*Xp*

that has the largest variance. By *normalized*, we mean that

P

*p*

*φ*

*j*1

*j*=1

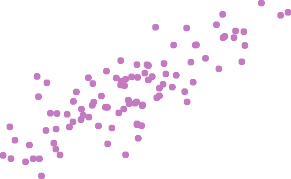
2 = 1.

* We refer to the elements *φ*11*, . . . , φp*1 as the loadings of the first principal component; together, the loadings make up

the principal component loading vector,

*φ*1 = (*φ*11 *φ*21 *. . . φp*1)*T* .

* We constrain the loadings so that their sum of squares is equal to one, since otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance.



10 20 30 40 50 60 70

Ad Spending

0 5 10 15 20 25 30 35

Population

The population size (pop) and ad spending (ad) for 100 different cities are shown as purple circles. The green solid line indicates the first principal component direction, and the blue dashed line indicates the second principal component direction.

* Suppose we have a *n* × *p* data set **X**. Since we are only interested in variance, we assume that each of the variables in **X** has been centered to have mean zero (that is, the column means of **X** are zero).
* We then look for the linear combination of the sample feature values of the form

*zi*1 = *φ*11*xi*1 + *φ*21*xi*2 + *. . .* + *φp*1*xip* (1) for *i* = 1*, . . . , n* that has largest sample variance, subject to

the constraint that L.*p* 2 = 1.

*j*=1

*φ*

*j*1

* Since each of the *xij* has mean zero, then so does *zi*1 (for

any values of *φj*1). Hence the sample variance of the *zi*1

can be written as 1 L.*n z*2 .

*n*

*i*=1

*i*1

* Plugging in [(1)](#_bookmark0) the first principal component loading vector solves the optimization problem

maximize

*φ*11*,...,φp*1

1

*n*

*i*=1

*n*

X

*p*

*j*=1

X

2

*φj*1*xij* 



subject to

X*j*=1

*φ*2 = 1*.*

*p*

*j*1

* This problem can be solved via a singular-value decomposition of the matrix **X**, a standard technique in linear algebra.
* We refer to *Z*1 as the first principal component, with realized values *z*11*, . . . , zn*1
* The loading vector *φ*1 with elements *φ*11*, φ*21*, . . . , φp*1 defines a direction in feature space along which the data vary the most.
* If we project the *n* data points *x*1*, . . . , xn* onto this direction, the projected values are the principal component scores *z*11*, . . . , zn*1 themselves.
* The second principal component is the linear combination of *X*1*, . . . , Xp* that has maximal variance among all linear combinations that are *uncorrelated* with *Z*1.
* The second principal component scores *z*12*, z*22*, . . . , zn*2

take the form

*zi*2 = *φ*12*xi*1 + *φ*22*xi*2 + *. . .* + *φp*2*xip,*

where *φ*2 is the second principal component loading vector, with elements *φ*12*, φ*22*, . . . , φp*2.

* It turns out that constraining *Z*2 to be uncorrelated with *Z*1 is equivalent to constraining the direction *φ*2 to be orthogonal (perpendicular) to the direction *φ*1. And so on.
* The principal component directions *φ*1, *φ*2, *φ*3*, . . .* are the ordered sequence of right singular vectors of the matrix **X**,

and the variances of the components are 1

*n*

times the

squares of the singular values. There are at most min(*n* − 1*, p*) principal components.

* USAarrests data: For each of the fifty states in the United States, the data set contains the number of arrests per

100*,* 000 residents for each of three crimes: Assault, Murder, and Rape. We also record UrbanPop (the percent of the population in each state living in urban areas).

* The principal component score vectors have length *n* = 50, and the principal component loading vectors have length

*p* = 4.

* PCA was performed after standardizing each variable to have mean zero and standard deviation one.

−0.5 0.0 0.5

UrbanPop

RhodMe aIslUsaatnacdhuseNttesw Jersey

Hawaii

California

Connecticut

Washington

Colorado

WiscoMnsininnesota Pennsylvania

Ohio

Oregon

IllinoisArizona

New York

Nevada

Rape

Nebr KansaOs klahoDmelaaware Missouri

Texas

aska Indiana

Michigan

New Hampshire

Iowa

Idaho

Maine rth Dakota

South Dakota

Virginia Wyoming

Montana

Kentucky

New Mexico

Maryland

Assault

TennesseLeouisiana

Florida

Arkansas

Alabama

VermontWest Virginia

Alaska

Georgia

Murder

South Carolina

North Carolina Mississippi

Second Principal Component

1

2

3

0.5

−3 −2 −1 0 1 2 3

−3

−2

−1

0

−0.5

0.0

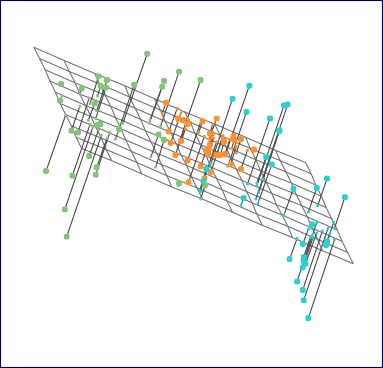
First Principal Component

The first two principal components for the USArrests data.

* The blue state names represent the scores for the first two principal components.
* The orange arrows indicate the first two principal component loading vectors (with axes on the top and right). For example, the loading for Rape on the first component is 0*.*54, and its loading on the second principal component 0*.*17 [the word Rape is centered at the point (0*.*54*,* 0*.*17)].
* This figure is known as a *biplot*, because it displays both the principal component scores and the principal component loadings.

|  |  |  |
| --- | --- | --- |
|  | PC1 | PC2 |
| Murder | 0.5358995 | -0.4181809 |
| Assault | 0.5831836 | -0.1879856 |
| UrbanPop | 0.2781909 | 0.8728062 |
| Rape | 0.5434321 | 0.1673186 |

1.0

−1.0 −0.5 0.0 0.5 1.0

• •

•

• •

•

•

• • • • ••

• • ••

• • •

•

•

• •

• •

•

• • •• • •

•

•

**• •** •

•

•

•

•

**•** •

•••

•

•

•

• •

••**•** •• •

•

••

•

• • •

• •

•

•

•

• •

• • •• •

•

•• •••

Second principal component

−1.0

−0.5

0.0

0.5

First principal component

* The first principal component loading vector has a very special property: it defines the line in *p*-dimensional space that is *closest* to the *n* observations (using average squared Euclidean distance as a measure of closeness)
* The notion of principal components as the dimensions that are closest to the *n* observations extends beyond just the first principal component.
* For instance, the first two principal components of a data set span the plane that is closest to the *n* observations, in terms of average squared Euclidean distance.
* If the variables are in different units, scaling each to have standard deviation equal to one is recommended.
* If they are in the same units, you might or might not scale the variables.

Scaled

−0.5 0.0 0.5

UrbanPop

\*

\*\* \*\* \*

\*

\*

\* \*

\* \*

\*

\*

\*

\*

\*

\* R\*ape

\*

\*

\*

Assault

\* \*

\*

\*

\*

\*

\*

\* \* \* \* \*

\*

\*

\*

\*

\*

\* \*

\*

\*

\*

\*

\*

M\*urder

\*

\*\*

Second Principal Component

1

2

3

Unscaled

−0.5 0.0 0.5 1.0

UrbanPop

Rape

\*

\*

\*

\* \*

\*\* \* \* \* \*\*

**\***

\* **\***

\*\*\*\* \* er

\***\***

\*

\*

\*

M\* urd\* \* \*

\* \*\* \*

\*

\*\* **\*** \*

\*

\* \* \*

\*

\* Assa

\*

0.5

Second Principal Component

150

0.5

1.0

−3 −2 −1 0 1 2 3 −100 −50 0 50 100 150

−3

−2 −1

0

−0.5

0.0

−100 −50 0 50 100

−0.5

0.0

First Principal Component First Principal Component

* To understand the strength of each component, we are interested in knowing the proportion of variance explained (PVE) by each one.



Proportion Variance Explained

* The *total variance* present in a data set (assuming that the variables have been centered to have mean zero) is defined as

*p p n*

L Var(*X* ) = L 1 L *x*2 *,*

*j*

*n*

*ij*

*j*=1

*j*=1

*i*=1

and the variance explained by the *m*th principal component is

1 L

Var(*Z* ) = *z*2 *.*

*m*

*n*

*im*

*n*

*i*=1

*p j*=1

* It can be shown that L.

*m*=1

with *M* = min(*n* − 1*, p*).

Var(*Xj*) = L.*M*

Var(*Zm*),

Prop. Variance Explained

0.0 0.2 0.4 0.6 0.8 1.0

Cumulative Prop. Variance Explained

0.0 0.2 0.4 0.6 0.8 1.0

* Therefore, the PVE of the *m*th principal component is given by the positive quantity between 0 and 1



Proportion Variance Explained: continued

L.*n z*2

*p*

*j*=1

*n*

*i*=1

*x*

L.

L.

*i*=1 *im*

2 *.*

*ij*

* The PVEs sum to one. We sometimes display the cumulative PVEs.

1.0 1.5 2.0 2.5 3.0 3.5 4.0 1.0 1.5 2.0 2.5 3.0 3.5 4.0

Principal Component Principal Component

If we use principal components as a summary of our data, how many components are sufficient?

* No simple answer to this question, as cross-validation is not available for this purpose.
  + *Why not?*

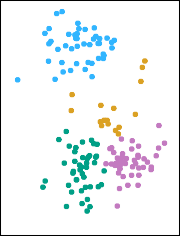
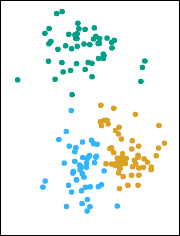
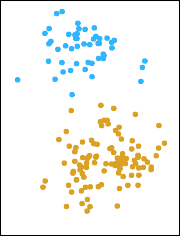
If we use principal components as a summary of our data, how many components are sufficient?

* No simple answer to this question, as cross-validation is not available for this purpose.
  + *Why not?*
  + When could we use cross-validation to select the number of components?

If we use principal components as a summary of our data, how many components are sufficient?

* No simple answer to this question, as cross-validation is not available for this purpose.
  + *Why not?*
  + When could we use cross-validation to select the number of components?
* the “scree plot” on the previous slide can be used as a guide: we look for an “elbow”.
* *Clustering* refers to a very broad set of techniques for finding *subgroups*, or *clusters*, in a data set.
* We seek a partition of the data into distinct groups so that the observations within each group are quite similar to each other,
* It make this concrete, we must define what it means for two or more observations to be *similar* or *different*.
* Indeed, this is often a domain-specific consideration that must be made based on knowledge of the data being studied.
* PCA looks for a low-dimensional representation of the observations that explains a good fraction of the variance.
* Clustering looks for homogeneous subgroups among the observations.
* Suppose we have access to a large number of measurements (e.g. median household income, occupation, distance from nearest urban area, and so forth) for a large number of people.
* Our goal is to perform *market segmentation* by identifying subgroups of people who might be more receptive to a particular form of advertising, or more likely to purchase a particular product.
* The task of performing market segmentation amounts to clustering the people in the data set.
* In *K-means clustering*, we seek to partition the observations into a pre-specified number of clusters.
* In *hierarchical clustering*, we do not know in advance how many clusters we want; in fact, we end up with a tree-like visual representation of the observations, called a *dendrogram*, that allows us to view at once the clusterings obtained for each possible number of clusters, from 1 to *n*.

**K=2 K=3 K=4**



A simulated data set with 150 observations in 2-dimensional space. Panels show the results of applying *K*-means clustering with different values of *K*, the number of clusters. The color of each observation indicates the cluster to which it was assigned using the *K*-means clustering algorithm. Note that there is no ordering of the clusters, so the cluster coloring is arbitrary.

These cluster labels were not used in clustering; instead, they are the outputs of the clustering procedure.

Let *C*1*, . . . , CK* denote sets containing the indices of the observations in each cluster. These sets satisfy two properties:

1. *C*1 ∪ *C*2 ∪ *. . .* ∪ *CK* = {1*, . . . , n*}. In other words, each observation belongs to at least one of the *K* clusters.
2. *Ck* ∩ *Ckt* = ∅ for all *k* I= *k*t. In other words, the clusters are non-overlapping: no observation belongs to more than one cluster.

For instance, if the *i*th observation is in the *k*th cluster, then

*i* ∈ *Ck*.

* The idea behind *K*-means clustering is that a *good* clustering is one for which the *within-cluster variation* is as small as possible.
* The within-cluster variation for cluster *Ck* is a measure WCV(*Ck*) of the amount by which the observations within a cluster differ from each other.
* Hence we want to solve the problem

minimize

*C*1*,...,CK*

*K*

*k*=1

(L

WCV(*Ck*))

*.* (2)

* In words, this formula says that we want to partition the observations into *K* clusters such that the total

within-cluster variation, summed over all *K* clusters, is as small as possible.

* Typically we use Euclidean distance

WCV(*Ck*

1

) =

|*Ck*|

L L(*xij*

− *xitj* ) *,* (3)

2

*i,it*∈*Ck j*=1

*p*

where |*Ck*| denotes the number of observations in the *k*th cluster.

* Combining [(2)](#_bookmark1) and [(3)](#_bookmark2) gives the optimization problem that defines *K*-means clustering,

minimize

L*K* 1

L L(*xij* − *xitj* )2

*.* (4)

*p*

*C*1*,...,CK*

*k*

*k*=1 |*Ck* | *i,it*∈*C*

*j*=1 

1. Randomly assign a number, from 1 to *K*, to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
   1. For each of the *K* clusters, compute the cluster *centroid*. The *k*th cluster centroid is the vector of the *p* feature means for the observations in the *k*th cluster.
   2. Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

* This algorithm is guaranteed to decrease the value of the objective [(4)](#_bookmark3) at each step. *Why?*
* This algorithm is guaranteed to decrease the value of the objective [(4)](#_bookmark3) at each step. *Why?* Note that

1

|*Ck*|

L L(*xij*

− *xitj* )2 = 2 L L(*xij*

− *x*¯*kj*

)2*,*

*i,it*∈*Ck j*=1 *i*∈*Ck j*=1

*p*

*p*

where *x*¯*kj* = 1 L.*i*∈*C xij* is the mean for feature *j* in

|*Ck*|

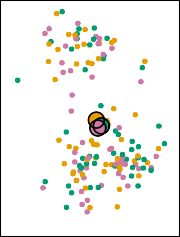
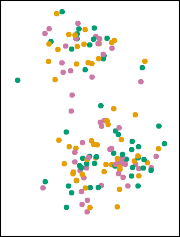
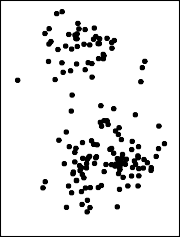
*k*

cluster *Ck*.

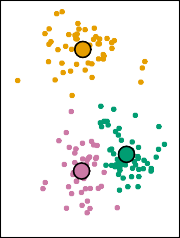
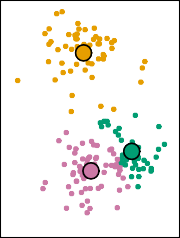
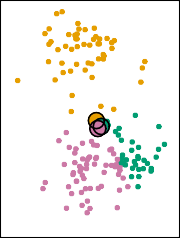
* however it is not guaranteed to give the global minimum.

*Why not?*

**Data Step 1 Iteration 1, Step 2a**



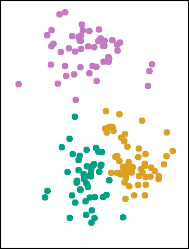
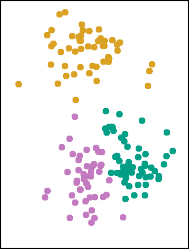
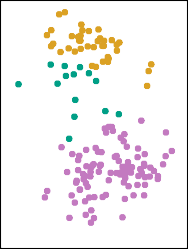
**Iteration 1, Step 2b Iteration 2, Step 2a Final Results**



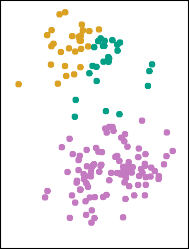
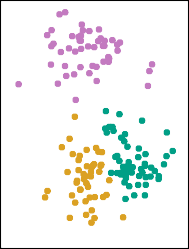
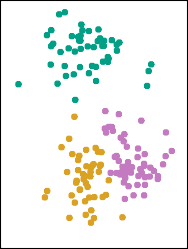
The progress of the K-means algorithm with *K*=3.

* *Top left:* The observations are shown.
* *Top center:* In Step 1 of the algorithm, each observation is randomly assigned to a cluster.
* *Top right:* In Step 2(a), the cluster centroids are computed. These are shown as large colored disks. Initially the centroids are almost completely overlapping because the initial cluster assignments were chosen at random.
* *Bottom left:* In Step 2(b), each observation is assigned to the nearest centroid.
* *Bottom center:* Step 2(a) is once again performed, leading to new cluster centroids.
* *Bottom right:* The results obtained after 10 iterations.

**320.9 235.8 235.8**



**235.8 235.8 310.9**



*K*-means clustering performed six times on the data from previous figure with *K* = 3, each time with a different random assignment of the observations in Step 1 of the *K*-means algorithm.

Above each plot is the value of the objective [(4).](#_bookmark3)

Three different local optima were obtained, one of which resulted in a smaller value of the objective and provides better separation between the clusters.

Those labeled in red all achieved the same best solution, with an objective value of 235.8

* *K*-means clustering requires us to pre-specify the number of clusters *K*. This can be a disadvantage (later we discuss strategies for choosing *K*)
* *Hierarchical clustering* is an alternative approach which does not require that we commit to a particular choice of *K*.
* In this section, we describe *bottom-up* or *agglomerative* clustering. This is the most common type of hierarchical clustering, and refers to the fact that a dendrogram is built starting from the leaves and combining clusters up to the trunk.

D

E

A

B

C

D

E

A

B

C

D

E

A

B

C

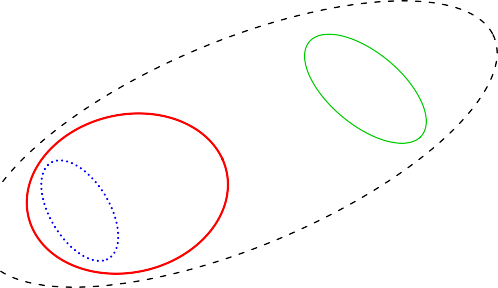
D

E

A

B

C



D

E

A

B

C

* Start with each point in its own cluster.

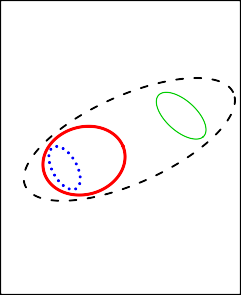
0

D E B A

C

* Identify the closest two clusters and merge them.
* Repeat.
* Ends when all points are in a single cluster.

**Dendrogram**



D

E

A B

C

1

2

3

4

0

D E B A

C

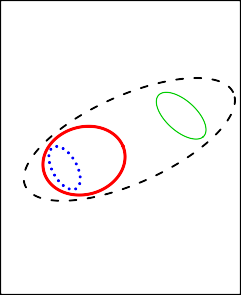
The approach in words:



Hierarchical Clustering Algorithm

* Start with each point in its own cluster.
* Identify the closest two clusters and merge them.
* Repeat.
* Ends when all points are in a single cluster.

**Dendrogram**



D

E

A B

C

1

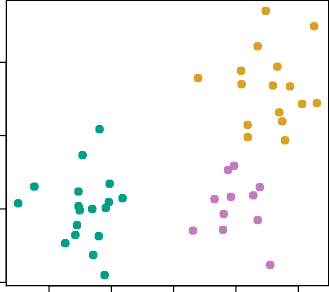
2

3

4



An Example

−6 −4 −2 0 2

*X*2

2

−2

0

4

*X*1

45 observations generated in 2-dimensional space. In reality there are three distinct classes, shown in separate colors.

However, we will treat these class labels as unknown and will seek to cluster the observations in order to discover the classes from the data.



Application of hierarchical clustering

0

2

4

6

8

10

0

2

4

6

8

10

0 2 4 6 8 10

* *Left:* Dendrogram obtained from hierarchically clustering the data from previous slide, with complete linkage and Euclidean distance.



Details of previous figure

* *Center:* The dendrogram from the left-hand panel, cut at a height of 9 (indicated by the dashed line). This cut results in two distinct clusters, shown in different colors.
* *Right:* The dendrogram from the left-hand panel, now cut at a height of 5. This cut results in three distinct clusters, shown in different colors. Note that the colors were not used in clustering, but are simply used for display purposes in this figure

9

7

8 5

3

2

1

6

4

−1.5 −1.0 −0.5 0.0 0.5 1.0



Another Example

0.0 0.5 1.0 1.5 2.0 2.5 3.0

3

4

1

6

9

2

8

5

7

*X*2

−1.5 −1.0 −0.5 0.0 0.5

*X*1

* An illustration of how to properly interpret a dendrogram with nine observations in two-dimensional space. The raw data on the right was used to generate the dendrogram on the left.
* Observations 5 and 7 are quite similar to each other, as are observations 1 and 6.
* However, observation 9 is *no more similar to* observation 2 than it is to observations 8*,* 5*,* and 7, even though observations 9 and 2 are close together in terms of horizontal distance.
* This is because observations 2*,* 8*,* 5*,* and 7 all fuse with observation 9 at the same height, approximately 1*.*8.

−1.5 −1.0 −0.5 0.0 0.5 1.0

9

7

8

3

5

2

1

6

4

*X*2

−0.5 0.0

−1.5

−1.0

0.5

*X*1

9

8

3

2

1

6

4

7

5

0.5

−1.5 −1.0 −0.5 0.0 0.5 1.0

9

8

3

2

1

6

4

7

5

*X*2

−1.5 −1.0 −0.5 0.0 0.5

*X*1

9

7

8

3

5

2

1

6

4



Merges in previous example

−1.5 −1.0 −0.5 0.0 0.5 1.0

*X*2

−1.5 −1.0 −0.5 0.0

*X*1

−1.5 −1.0 −0.5 0.0 0.5 1.0

*X*2

−1.5 −1.0 −0.5 0.0 0.5

*X*1



Types of Linkage

|  |  |
| --- | --- |
| *Linkage* | *Description* |
| Complete | Maximal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *largest* of these dissimilarities. |
| Single | Minimal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *smallest* of these dissimilarities. |
| Average | Mean inter-cluster dissimilarity. Compute all pairwise  dissimilarities between the observations in cluster A and the observations in cluster B, and record the *average* of these dissimilarities. |
| Centroid | Dissimilarity between the centroid for cluster A (a mean vector of length *p*) and the centroid for cluster B. Cen- troid linkage can result in undesirable *inversions*. |

* So far have used Euclidean distance.
* An alternative is *correlation-based distance* which considers

two observations to be similar if their features are highly correlated.

* This is an unusual use of correlation, which is normally

computed between variables; here it is computed between the observation profiles for each pair of observations.

Observation 1

Observation 2

Observation 3

2

3

1

5 10 15 20

0 5 10 15 20

Variable Index

46 / 52

Socks Computers



0

2

4

6

8

10

0.0

0.2

0.4 0.6 0.8

1.0

1.2

0

500

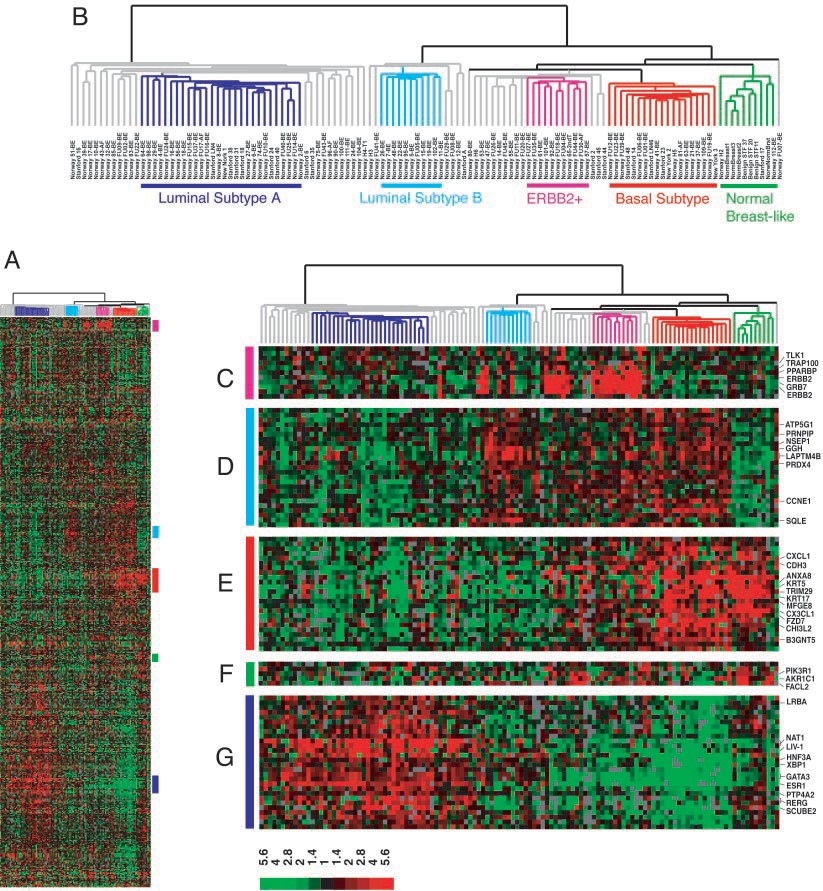
1000

1500

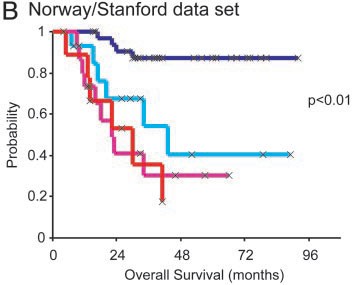
Socks Computers

Socks Computers

* Should the observations or features first be standardized in some way? For instance, maybe the variables should be centered to have mean zero and scaled to have standard deviation one.
* In the case of hierarchical clustering,
  + What dissimilarity measure should be used?
  + What type of linkage should be used?
* How many clusters to choose? (in both *K*-means or hierarchical clustering). Difficult problem. No agreed-upon method. See Elements of Statistical Learning, chapter 13 for more details.
* “Repeated observation of breast tumor subtypes in independent gene expression data sets;” Sorlie at el, PNAS 2003
* Average linkage, correlation metric
* Clustered samples using 500 *intrinsic genes:* each woman was measured before and after chemotherapy. Intrinsic genes have smallest within/between variation.







* *Unsupervised learning* is important for understanding the variation and grouping structure of a set of unlabeled data, and can be a useful pre-processor for supervised learning



Conclusions

* It is intrinsically more difficult than *supervised learning* because there is no gold standard (like an outcome variable) and no single objective (like test set accuracy)
* It is an active field of research, with many recently developed tools such as *self-organizing maps*, *independent components analysis* and *spectral clustering*.

See *The Elements of Statistical Learning*, chapter 14.