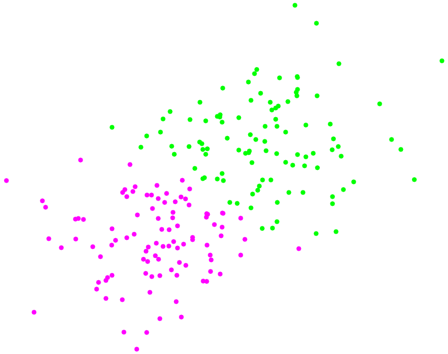
there are often latent variables that identity subsets in a collection of documents

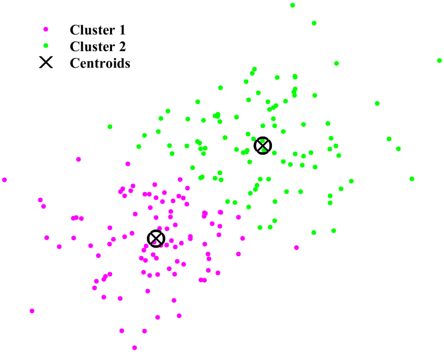
technique that can identify corpus subsets based on document (dis-)similarity

preferably, the model can be used for both utility and understanding

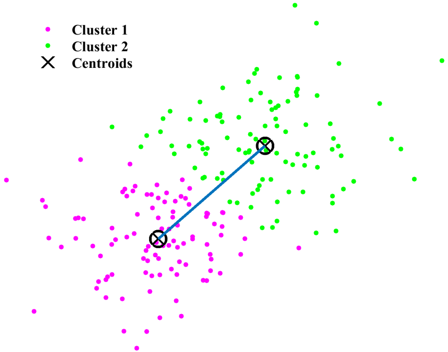
implicit assumption that we study differences in variables (e.g., terms) between homogeneous objects (e.g., documents)



systematic differences between objects result in non-random subsets that are often ignored



Cluster analysis: partitions data into homogeneous subsets using inter-object similarity/distance measures



minimize distance between the centroid and points within each cluster
maximize distance centroids and points between clusters

	t_1	t_2	...	t_k
d_1	f_{d_1,t_1}			
d_2		f_{d_2,t_2}		
...			...	
d_n				f_{d_n,t_n}

$\Rightarrow$

	d_1	d_2	...	d_n
d_1	0			
d_2		0		
...			...	
d_n				0

$$C = \{d_{1,C_1}, d_{2,C_2}, ..., d_{n,C_k}\} \text{ where } k \leq n$$

convert a matrix of n documents measured on k terms to a matrix of inter-document similarity and then apply a clustering method to the similarity/distance matrix

either because we want conceptually meaningful groups of documents (or terms) that share common characteristics *or* because we want useful groups that abstract from the individual documents (summarization or compression)

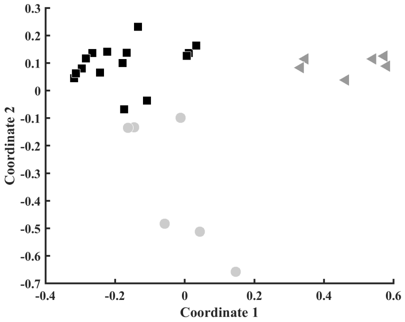
clustering for understanding or utility

k-means is a widely used clustering technique that partitions n documents (or terms) in k clusters

clusters are non-overlapping, so a document belong exclusively to one cluster

```
select k points as initial centroids
repeat
form k clusters by assigning each point to its closest centroid
recompute the centroid of each cluster
until centroid do not change
```

k-means is a prototyped-based clustering method that finds a centroid (mean) of all the points in a cluster and minimizes the distance (within-cluster sum of squares) of each point to centroid



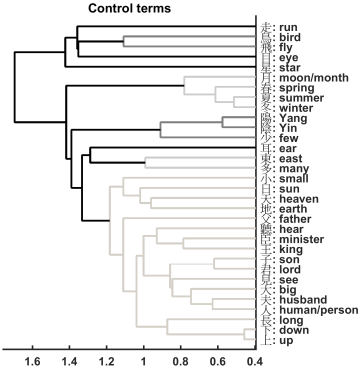
projection is aoften applied to the document matrix for visualization purposes

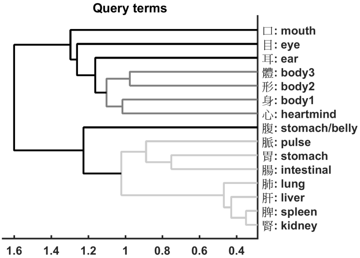
agglomerative hierarchical clustering is a set of clustering methods that starts with each document as a single cluster and then repeatedly merge the two closest clusters until a single, all encompassing cluster remains (alternate methods use divisive clustering)

hierarchical clustering produce nested clusters that are organized in a tree-like structure (visualized with a dendrogram)

1.	compute proximity matrix
2.	**repeat**
3.	merge the closest two clusters
4.	update the proximity matrix to reflect the distance between the new clusters and the original clusters
5.	**until** only on cluster remains

to compute the proximity between groups of data points a particular technique is chosen (e.g. min, max, group average)





with hierarchical clustering you cut or prune the tree at some level to define clusters.

k-means with scikit-learn

```
docs = vanilla_folder(datapath)

## kmeans partioning
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans

# build vector space
vectorizer = TfidfVectorizer(stop_words='english')
vectspc = vectorizer.fit_transform(docs)

# train model
k = 5
mdl = KMeans(n_clusters = k, init='k-means++', max_iter=100, n_init=1, random_state = 1234)
mdl.fit(vectspc)

print("Top features per cluster:")
order_centroids = mdl.cluster_centers_.argsort()[:, ::-1]
features = vectorizer.get_feature_names()
for i in range(k):
    print "Cluster %d:" % i,
    for ind in order_centroids[i, :10]:
        print ' %s' % features[ind],
    print
```

k-means in R

```
books.mat <- as.matrix(books.dtm)

## kmeans
# length normalize
books.mat <- norm_eucl(books.mat)

# graphical approach to determining k
wssplot <- function(data, nc=15, seed=1234){
  wss <- (nrow(data)-1)*sum(apply(data,2,var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
  plot(1:nc, wss, type="b", xlab="Number of Clusters",
       ylab="Within groups sum of squares")}
max_k = 10
dev.new()
wssplot(books.mat,nc = max_k)

# 3 sub-groups or clusters
k = 3
books.cl <- kmeans(books.mat, k)
# classification
books.cl$cluster
x <- prcomp(books.mat)$x[,1]; y <- prcomp(books.mat)$x[,2]; names <- capname(rownames(books.mat))
cols = as.double(books.cl$cluster)
dev.new()
plot(x, y, type='p', pch=20, col=cols, cex = 2,xlab='Comp.1',
        ylab='Comp.2',xlim = c(-.4,.7),ylim = c(-.7,.3))
text(x, y, class.v, col=cols, cex=.8, pos=4)
```