MSA220 - Statistical Learning for Big Data

Lecture 8b

Rebecca Jörnsten

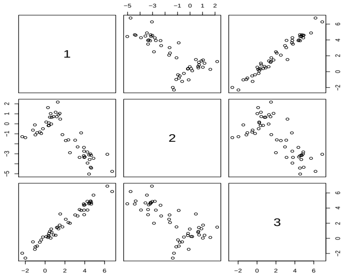
Mathematical Sciences
University of Gothenburg and Chalmers University of Technology

Feature selection

Once we move to a modelbased clustering procedure we can use BIC to select features as well. An elegant approach to this, which is an extension of Mclust, is the following (implemented in the clustvarsel() package.
Consider the following; maybe not all features are relevant for clustering, either directly or indirectly.

Feature selection

Here's an example where $x1$ and $x2$ are relevant for clustering (having means $0, 0$ and $4, -3$ for the two clusters respectively, and correlation 0.6 and -0.6 between features 1 and 2 in cluster 1 and 2 respectively). Feature $x3$ is related to the clustering indirectly as $x3 = x1 + e$, $e \sim N(0, .5)$.



We can of course also have features that are completely unrelated to the clustering, i.e. not differ in mean for the different clusters and not correlated with any feature that is mean-shifted between clusters.

ClustVarSel

ClustVarSel is a procedure that decomposes the likelihood as follows. Let Z be the cluster labels for the data set:

$$p(x_c, x_{no-c} \mid Z) = p(x_c \mid Z)p(x_{no-c} \mid Z, x_c) = p(x_c \mid Z)p(x_{no-c} \mid x_c)$$

The x-variables are thus partitioned into a set x_c that is dependent on the clustering label, i.e. the multivariate normal distribution for the x_c variables have mean and possibly covariance parameters that are cluster specific. The variable set x_{no-c} are conditionally (on x_c) independent of the cluster labels. That is, if we know x_c then $x_{no-c} \mid x_c$ distributions are not cluster specific.

ClustVarSel

If x_{no-c} has a distribution that cannot be simplified to remove cluster-specific distribution parameters by conditioning on x_c, then x_{no-c} is directly related to the clustering.
The ClustVarSel procedure searches for variables to add in either the c (cluster related) or $no - c$ (not cluster related) set. The partitioning that is optimal is determined via the BIC.

ClustVarSel

- If x_1 is in the cluster relevant set already, consider adding x_2
- Fit the model where both x_1, x_2 are in the set c.
- Fit the model where x_1 is in the set c and x_2 is not. This is done via a mixture model fit with x_1 and a regression model for $x_2 \mid x_1$.
- Compute BIC for the two alternatives and pick the alternative that has the smallest BIC.
- Considering adding or removing variables from the set c until no move can be accepted (no smaller BIC alternative).

The search can be done forward (where no variable is in set c to start with) or backward (where all are in the set c initially).

ClustVarSel

Running clustvarsel on the simulation from figure above:

```
'clustvarsel' model object:

Stepwise (forward) greedy search:
  Var.proposed  BIC        BIC diff.   TypeStep  Decision
1 2            -211.3307  5.255851     Add        Accepted
2 1            -377.4739  38.873054    Add        Accepted
3 3            -377.4739  -14.769282   Add        Rejected
4 1            -377.4739  38.873083    Remove     Rejected

Selected subset: 2, 1 }
```

Clustvarsel picks variables 2 and 1 to be cluster related (correctly) and does not add variable 3 (also correct decision).

High-dimensional Classification and Clustering

- Reduce the number of parameters in the mixture model
- Assume classes/clusters live in a lower dimensional space (intrinsic number of dimensions)
- How? Generalize QDA/Mixture model to only utilize the leading PC components of the class/cluster-specific Σ_k

High-dimensional Classification and Clustering

- Assume Q_k are the leading d_k components of the $p \times p$-dimensional Σ_k
- Assume the corresponding leading eigenvalues are $a_{jk}, j = 1, \cdots, d_k$ and the remaining eigenvalues are small and equal b_k
- Think of the $p - d_k$ dimensions corresponding to the small eigenvalues as noise
- Estimate parameters under these restrictions - save a lot of parameters!
- Choose class/cluster-specific complexity (d_k) via BIC
- R-package HDclassif

Consensus clustering

- Any method that comprises many steps is subject to instability since each step is a source of error
- How many features, how many eigenvalues?
- In addition, many clustering methods are quite sensitive to small data perturbations

Consensus clustering

- If you can do things once, you can do it 100 times!
- Add some randomness to the procedure and run it many times
- Retain clusters that are *stable* across multiple runs!

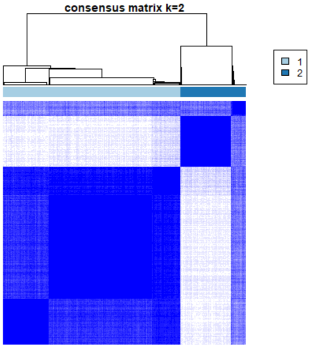
Consensus clustering

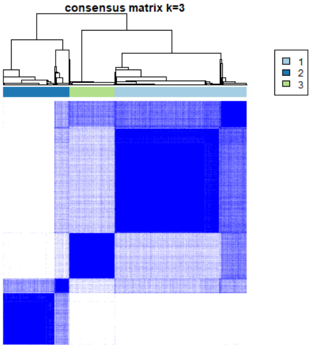
- How add randomness?
- Resampling of observations... but also
- Subset of features
- Subset of features + PCA
- Random projections
- ....

Consensus clustering

- Each run produces a clustering result
- How do we combine these?
- Some methods compare the clusters in terms of overlap
- Other methods use a similar idea to RF clustering: for each pair of objects, count how many times they appear in a cluster together. Use this is a new similarity metric and use e.g. hierarchical clustering to produce a final result.
- I like the latter approach because it gives you a lot of flexibility in which clustering procedures to compare across runs.

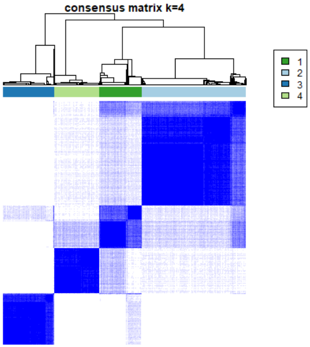
Example of results

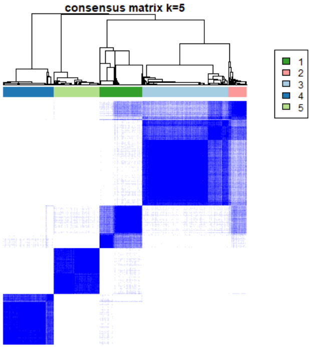




Consensus matrices (i.e. proportion of times clusters are in agreement for all pairs of observations.

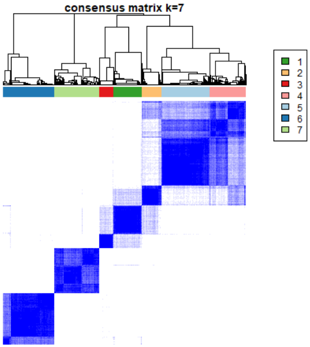
Example of results

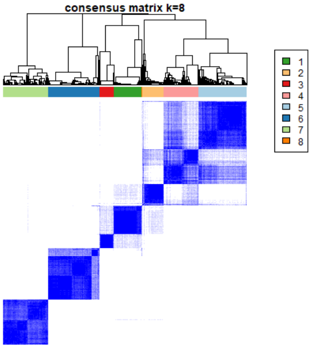




Consensus matrices (i.e. proportion of times clusters are in agreement for all pairs of observations.

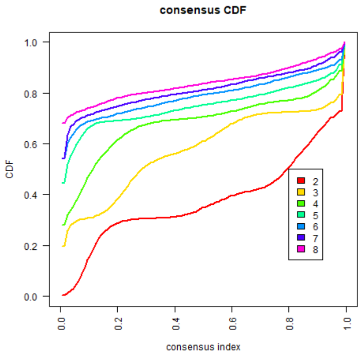
Example of results

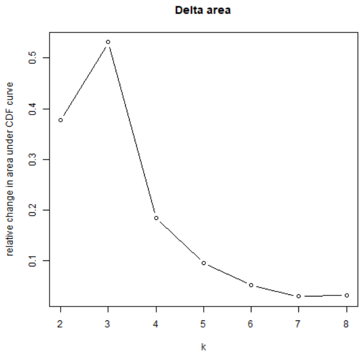




Consensus matrices (i.e. proportion of times clusters are in agreement for all pairs of observations.

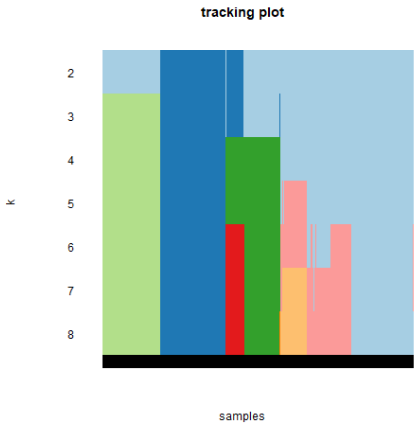
Example of results





CDF of consensus matrices can be used to choose the number of clusters. Check where adding clusters "stops paying off" - can be assessed by comparing the CDF differences (right panel).

Example of results



It can be interesting to look at how clusters are formed as you increase the number of clusters.

Graphical Lasso

- Remember our discussion about the covariance matrix and the inverse covariance matrix in class
- The *sparse* inverse covariance matrix has non-zero entries where there is a direct correlation between items
- That is, if there is a partial correlation remaining once we account for all other dependencies.
- Can also utilize this for *network modeling*
- Nice visualizations of complex data!
- Related to clustering in the sense that....
- ... observations are represented in a network - neighbors are more similar.
- But also a more complex question - neighbors are close once dependency on other observations taken into account

Graphical Lasso

- Lots of methods for network modeling (Bayesian networks, information theoretic, directed/mechanistic,...)
- Here we will focus on *sparse modeling*
- Assume data comes from a multivariate normal model $N(\mu, \Sigma)$
- The inverse of the covariance matrix Σ, Θ, is called the *precision matrix*

Graphical Lasso

- The inverse of the covariance matrix Σ, Θ, is called the *precision matrix*
- Fact: The precision matrix is non-zero for entry i, j only if the *partial correlation* between i, j is non-zero
- Partial correlation $=$ correlation between i, j once dependency on all other observations accounted for
- $\theta_{i,j} = Cov(X_i, X_j \mid X_k, k \neq i, j)$
- Can compute the partial correlation from residual correlation from regression of i on all other variables and j on all other variables

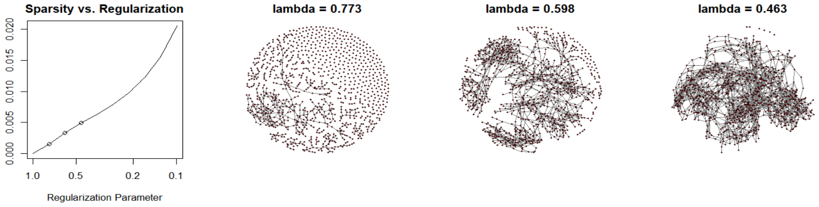
Graphical Lasso

- In practice, can't compute the inverse $\hat{\Theta}$ of the $p \times p$ $\hat{\Sigma}$ if $p > n$
- Sparse modeling to the rescue - which we will learn more about after the easter break.
- However, what we do is essentially regularize the inverse estimates, shrinking some elements toward 0.
- Specifically: We maximize the gaussian log-likeihood with penalty $\lambda \sum_{j<i} |\theta_{i,j}|$
- Methods: gradient based glasso, lasso-regression based neighborhood selection.
- Packages glasso and huge

Graphical Lasso

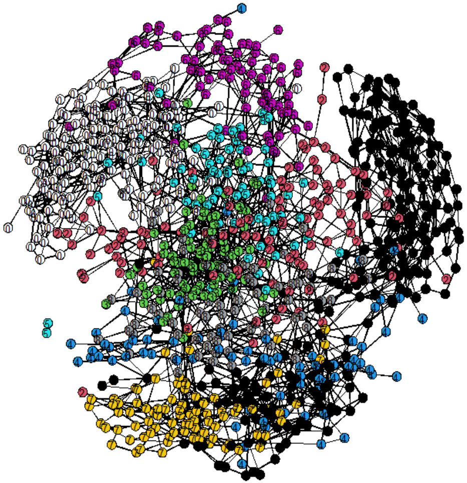
- Does it work?
- Like sparse regression, there are some caveats. Too many highly correlated Xs, we cannot identify the network model.
- Is the data sparse?
- Fixes: randomized lasso. Run glasso many times with random penalties: check how often a graph-link is selected.
- High-dimensional data? First filter. If a set of variables has no correlation with any member of another set exceeding λ, you can run glasso separately on the sets (implemented in huge package).

Examples: Graphical Lasso



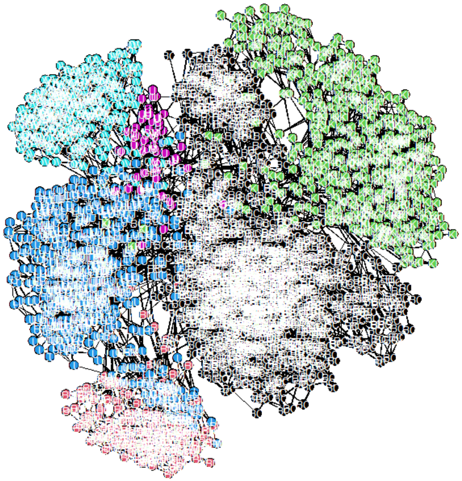
Networks at different levels of sparsity regulation.

Examples: Graphical Lasso



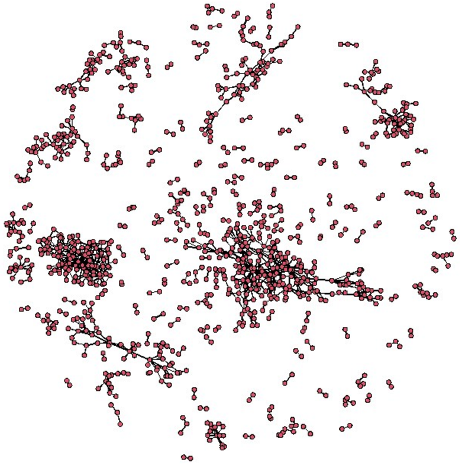
Networks on the digits data

Examples: Graphical Lasso



Networks from the cancer data.

Examples: Graphical Lasso



Networks from the cancer data - but now on the genes instead of the patients.

Take-home message

- Clustering - a more difficult task than classification because there is no "ground truth"
- Remember - clustering algorithms may make implicit assumptions about the data and what constitutes a good cluster - i.e. kmeans looking for spherical clusters
- Think carefully about scaling/standardizing the data and what impact this may have
- Selecting the number of clusters is another challenge
- Resampling can be used for consensus clustering - and also to select the number of clusters based on stability
- Lots of algorithms out there!!! Good idea to compare a few.