Lecture 2b

block 2

High-dimensional problems

Wide data

- **Wide data** $p \gg n$. Many variables, few data points.
  - **Genomics**
  - **Text**

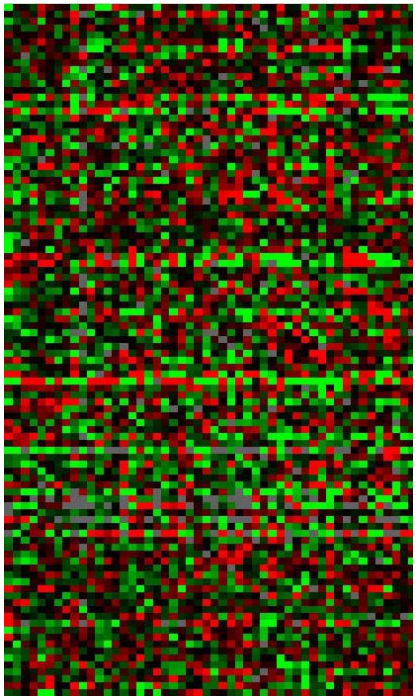
- **Tall data**: $\mathrm{p} \ll n$. Few variables, many data points.

Most of applications

- Economics, for ex. Currency exchange rates vs time
- Industry, Car performance characteristics vs probability of malfunctioning
- Surveys, customer satisfaction vs survey answers

- Tall and Wide. Supermarket scanners. Many purchases, many products.

Genomics-microarrays



Hastie et al:. *DNA microarray data: expression matrix of* 6830 *genes (rows) and* 64 *samples (columns), for the human tumor data. Only a random sample of* 100 *rows are shown. The display is a heat map, ranging from bright green (negative, under expressed) to bright red (positive, over expressed). Missing values are gray. The rows and columns are displayed in a randomly chosen order.*

Text – document classification

Document	has('ball')	has('EU')	has('political_arena')	wordlen	Lex. Div.	Topic
Article1	Yes	No	No	4.1	5.4	Sports
Article2	No	No	No	6.5	13.4	Sports
⋮	⋮	⋮		⋮	⋮	⋮
ArticleN	No	No	Yes	7.4	11.1	News

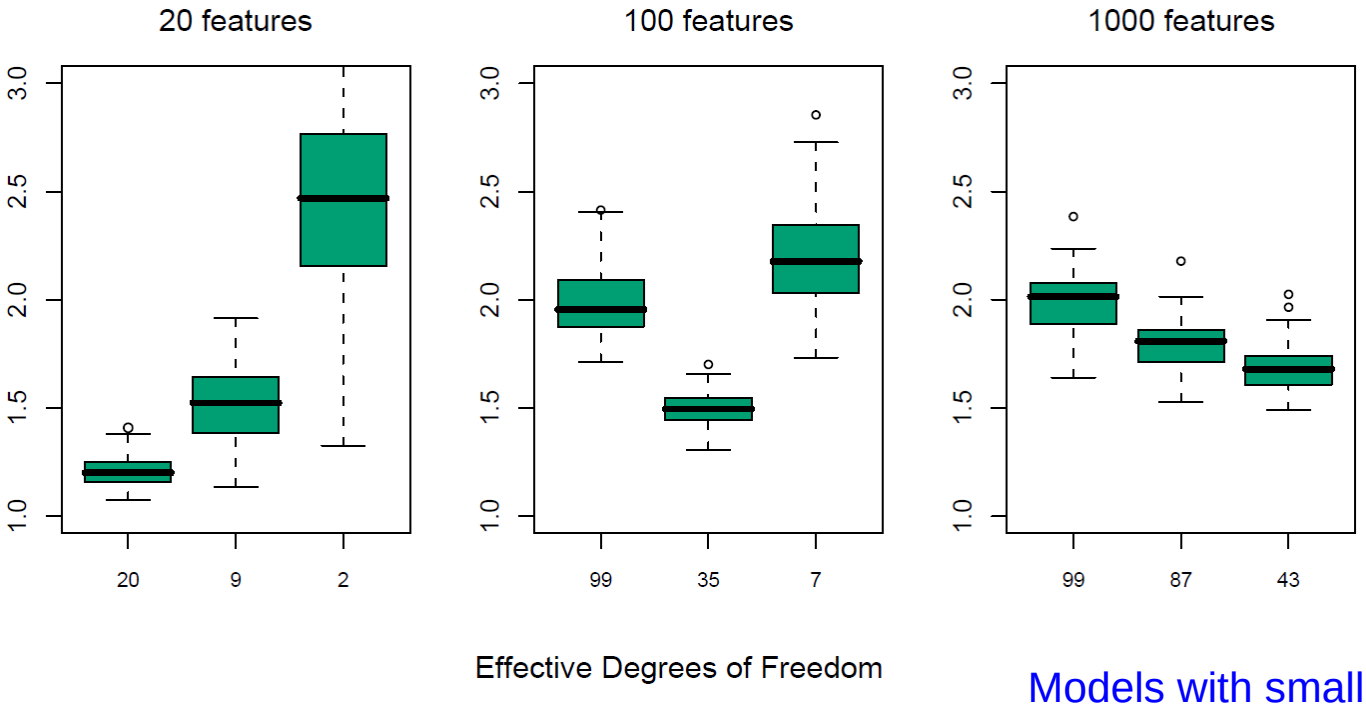
A problem with wide data

- Linear regression $\mu = w^T x, Y \sim N(\mu, \sigma^2)$
- ML solution $\widehat{w} = (X^T X)^{-1} X^T Y$
  - X is $n \times p$, has rank n
  - $X^T X$ is $p \times p$, has rank n
  - → $X^T X$ is not invertible!
- Solutions:
  - **Dimensionality reduction**: PCA, PCR
  - **Shrinkage**: Lasso, Ridge, Elastic network
  - **Forward variable selection**

- Algorithms need sometimes be modified for wide data.

Effective amount of features for wide data

- Linear response generated with different p, n=100
- Ridge is applied with different λ



Source: Hastie et al (2009)

Models with smaller effective number of features have better prediction

Classification: LDA

- Standard LDA

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$$

$$\hat{\boldsymbol{\mu}}_c = \frac{1}{N_c} \sum_{i:y_i=c} \mathbf{x}_i, \quad \hat{\boldsymbol{\Sigma}}_c = \frac{1}{N_c} \sum_{i:y_i=c} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)^T$$

$$\hat{\Sigma} = \frac{1}{N} \sum_{c=1}^{k} N_c \, \hat{\Sigma}_c$$

- → Σ^{-1} does not exist…

Classification: diagonal-covariance LDA

- Data is not enough to estimate dependences in covariance
- For wide data, we do **diagonal-covariance LDA** (naive Bayes):

$$\Sigma = diag\left(\sigma_1^2, .. \sigma_p^2\right)$$

- Discriminant function

$$\delta(x^{new}) = -\sum_{j=1}^{p} \frac{\left(x_j^{new} - \bar{x}_{kj}\right)^2}{s_j^2} + 2\log \pi_k$$

  - $s_j^2 = \frac{1}{n}\sum_i n_i var(x_j | Y = C_i)$
  - $\bar{x}_{kj} = mean(x_j | Y = C_k), \bar{x}_j = mean(x_j)$
- Classify to the highest discriminant function value
- **Drawback**: all features are in the model→ difficult to use in interpretations.

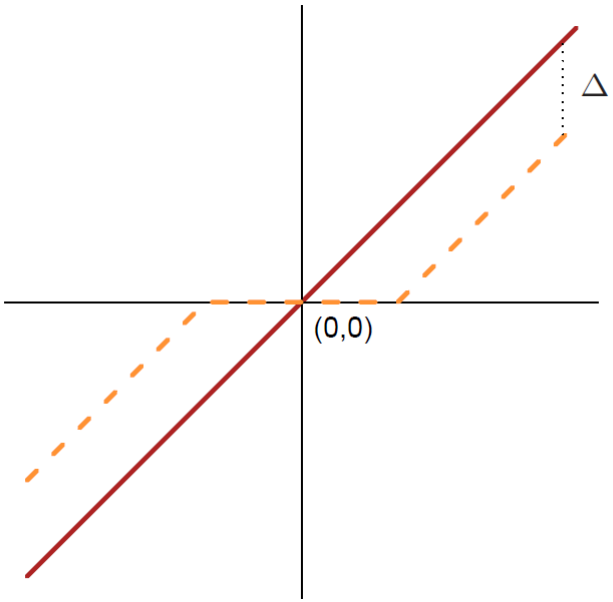
Classification: NSC

Nearest Shrunken Centroids

- Idea: Shrink classwise means towards overall mean

1. Compute $d_{kj} = \frac{\bar{x}_{kj} - \bar{x}_j}{m_k(s_j + s_0)}$
2. Shrink $d'_{kj} = sign(d_{kj})(|d_{kj}| - \Delta)_+$
3. Set $x'_{kj} = \bar{x}_j + m_k(s_j + s_0)d'_{kj}$

Only features with nonzero d'_{kj} contribute to classification!→insignificant features are shrunk!



Define Δ by CV

NSC: example

- LSVT Voice Rehabilitation Data Set
  - Target: Quality of voice rehabilitation
    - 1=acceptable, 2=not acceptable
  - Features: Properties of the signal (voice)
  - n=126, p=309

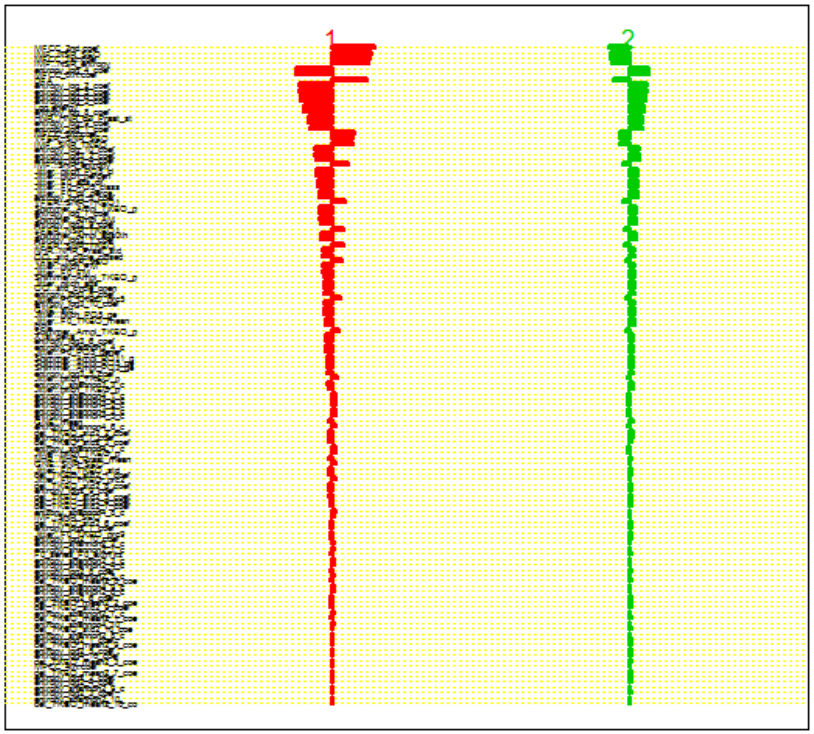
NSC: example

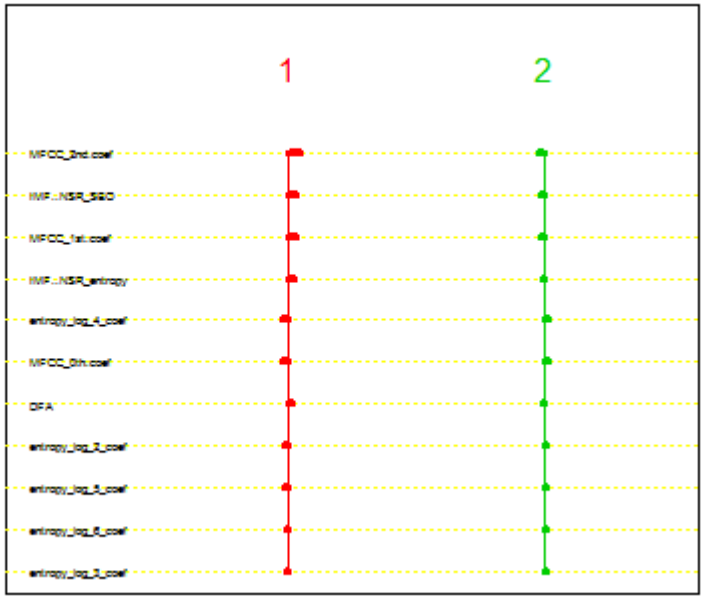
- Package **pamr**
  - pamr.train()
  - pamr.cv

```
data0=read.csv2("voice.csv")
data=data0
data=as.data.frame(scale(data))
data$Quality=as.factor(data0$Quality)
library(pamr)
rownames(data)=1:nrow(data)
x=t(data[,-311])
y=data[[311]]
mydata=list(x=x,y=as.factor(y),geneid=as.character(1:nrow(x)), genenames=rownames(x))
model=pamr.train(mydata,threshold=seq(0,4, 0.1))
pamr.plotcen(model, mydata, threshold=1)
pamr.plotcen(model, mydata, threshold=2.5)

a=pamr.listgenes(model,mydata,threshold=2.5)
cat( paste( colnames(data)[as.numeric(a[,1])], collapse='\n' ) )

cvmodel=pamr.cv(model,mydata)
print(cvmodel)
pamr.plotcv(cvmodel)
```

NSC: example

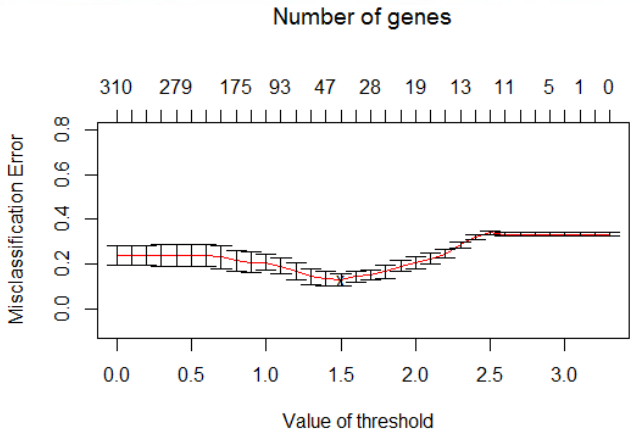
- Centroid plot, $\Delta = 1$ and $\Delta = 2.5$





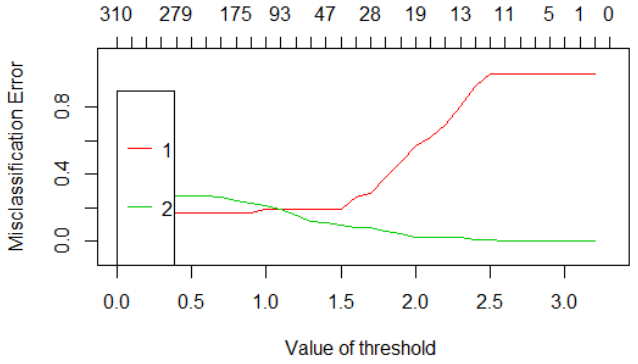
NSC: example

```
> pamr.listgenes(model,mydata,threshold=2.5)
      id  1-score 2-score
 [1,] 86  0.0897  -0.0449        MFCC_2nd.coef
 [2,] 80  0.0702  -0.0351        IMF..NSR_SEO
 [3,] 85  0.0652  -0.0326        MFCC_1st.coef
 [4,] 82  0.0517  -0.0259        IMF..NSR_entropy
 [5,] 153 -0.0507 0.0253         entropy_log_4_coef
 [6,] 84  -0.05   0.025          MFCC_0th.coef
 [7,] 60  0.0359  -0.0179        DFA
 [8,] 151 -0.0316 0.0158         entropy_log_2_coef
 [9,] 154 -0.0299 0.0149         entropy_log_5_coef
[10,] 155 -0.0193 0.0096         entropy_log_6_coef
[11,] 152 -0.018  0.009          entropy_log_3_coef
```



- Confusion matrix optimal Δ

	Pred 1	Pred 2
True 1	33	9
True 2	5	79



RDA

Regularized discriminant analysis

- Another way of solving singularity of Σ
  - γ is some constant

$$\hat{\Sigma}(\gamma) = \gamma\hat{\Sigma} + (1-\gamma)diag(\hat{\Sigma})$$

- $\gamma = 0 \rightarrow$ diagonal-covariance LDA
- γ is chosen by CV
- R: rda() in **klaR**

Regularized logistic regression

- Usual logistic regression

$$p(Y = C_i|x) = \frac{e^{w_{i0} + \boldsymbol{w_i^T x}}}{\sum_{j=1}^{K} e^{w_{j0} + \boldsymbol{w_j^T x}}} = softmax(w_{i0} + \boldsymbol{w_i^T x})$$

- Lp -Regularization:

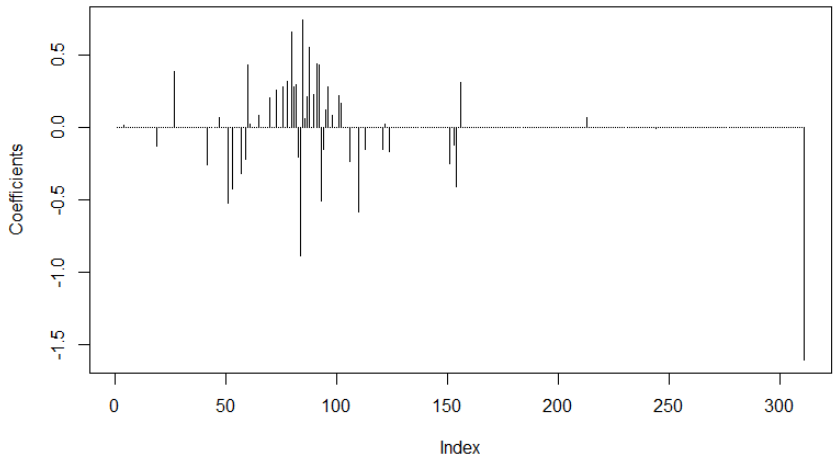
$$\max_{w} \sum_{i=1}^{n} \log p(Y_i|x_i) - \frac{\lambda}{2} \sum_{k=1}^{K} \|\boldsymbol{w}_i\|^p$$

- Parameter redunancy is solved
- **L1 regularization:** some w are shrunk to 0
- Numerical optimization is used to solve
- R: LiblineaR() in package **LiblineaR**

L1 logistic regression

- Voice rehabilitation

```
W=model2$W
plot(t(W), type="h", ylab="Coefficients")
```



	Pred 1	Pred 2
True 1	41	1
True 2	0	84

Overfitted?

SVM

- Support Vector Machine do not suffer from $p \gg n$ problem
  - Largest margin can be found even if the data is perfectly separable

Computational shortcuts p>>>n

- SVD decomposition $X = UDV^T = RV^T$
- If model is linear in parameters and has quadratic penalties:
  - Transform data observations from X into R
  - Minimize loss (minus log likelihood) with R instead of X and get $\boldsymbol{\theta}$
  - Original parameters $\boldsymbol{w} = V\boldsymbol{\theta}$
- Can be applied to many methods

- Example: ridge regression

Elastic net

- L1 regularization

$$\min_{w} -\log p(D|\boldsymbol{w}) + \lambda\|\boldsymbol{w}\|_1$$

$$\|w\|_1 = \sum_i |w_i|$$

- For p>n, LASSO can extract at most n nonzero components
  - Severe regularization if $p \gg n$
- L1 regularization→selects some feature among the correlated ones
- L2 regularization→ w's of the correlated variables are shrunk towards each other are nonzero

Elastic net

Combine L1 and L2 to diminish effect of L1 regularization.

- Elastic net regularization:

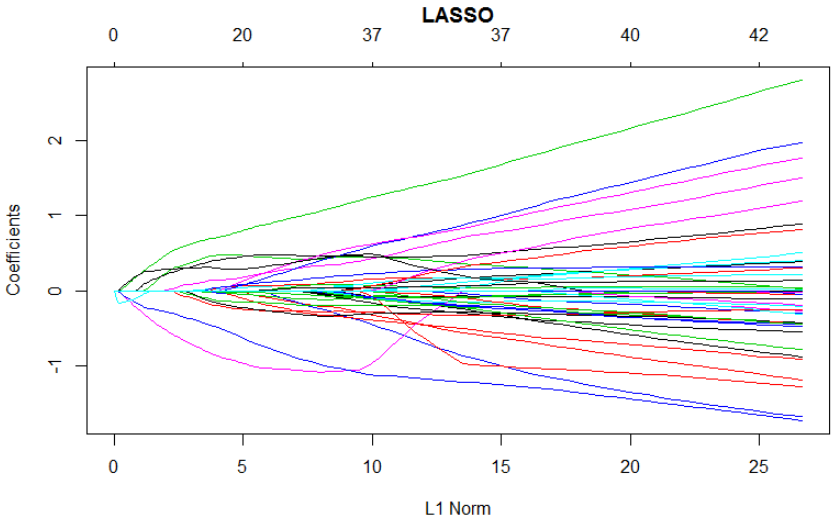
$$\min_{w} -\log p(D|\boldsymbol{w}) + \lambda(\alpha\|\boldsymbol{w}\|_1 + (1-\alpha)\,\|w\|_2)$$

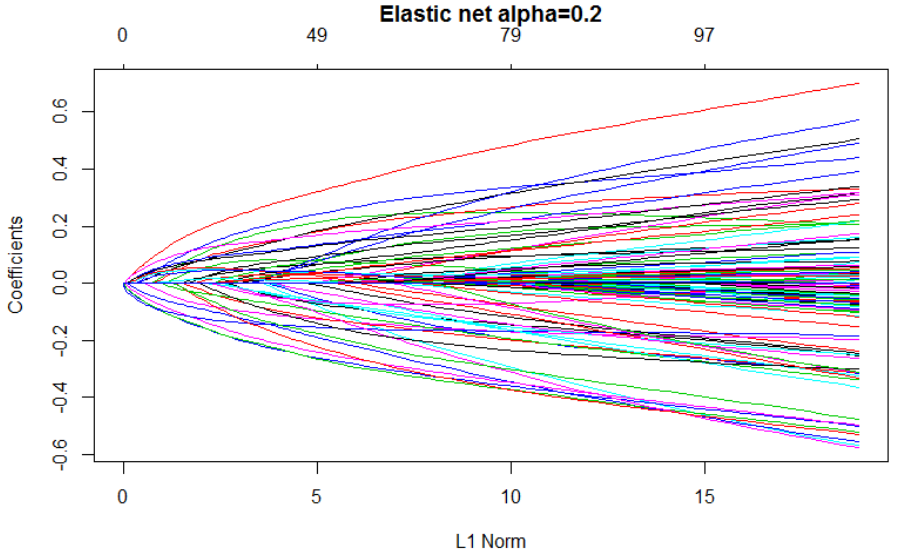
- α is set ad hoc or chosen by CV
- Elastic net may select more than n features

- R: glmnet() in **glmnet** package
  - Specify ”family” for classification or regression

Elastic net

- Voice rehabilitation





Comparative analysis

- Gene expression data

Methods	CV errors (SE) Out of 144	Test errors Out of 54	Number of Genes Used
1. Nearest shrunken centroids	35 (5.0)	17	6,520
2. L_2-penalized discriminant analysis	25 (4.1)	12	16,063
3. Support vector classifier	26 (4.2)	14	16,063
4. Lasso regression (one vs all)	30.7 (1.8)	12.5	1,429
5. k-nearest neighbors	41 (4.6)	26	16,063
6. L_2-penalized multinomial	26 (4.2)	15	16,063
7. L_1-penalized multinomial	17 (2.8)	13	269
8. Elastic-net penalized multinomial	22 (3.7)	11.8	384

When features are not available

- Sometimes it is difficult to define or use the feature set
  - Molecule
  - Text document
    - possible, but can be very high dimensional
- ..but a proximity measure $K(x, x')$ is easier to define
  - Ex: How much one document is different from another one

Source:http://images.wisegeek.com/illustration-of-a-molecule.jpg

→ We can compute similarity matrix
$\boldsymbol{K = XX^T}$

When features are not available

- Many methods can use $\boldsymbol{K}$ instead of $\boldsymbol{X}$
  - Note: p is not involved in calculations!!
- SVM: kernel trick→ $\boldsymbol{K}$ can be used directly
- K-Nearest neighbors
  - Transform similarity into distance $d_{ij}^2 = K(x_i, x_i) + K(x_i, x_j) - 2K(x_i, x_j)$
  - Use distances to find neighbors
- Can also be done for
  - Logistic and multinomial regression with L2 penalty
  - LDA
  - PCA: kernel PCA

Kernel PCA

- Usual PCA
  - Center $\boldsymbol{X}$
  - Find $\boldsymbol{S}\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i, \boldsymbol{S} = \frac{1}{n}\boldsymbol{X}^T\boldsymbol{X}, \boldsymbol{S} = [p \times p]$
    - $\boldsymbol{u}_i$ has dimension p
  - Project data on PCs: $Z = X\ U$

- Problems: X is unknown, and it can be p can be very large

Kernel PCA

- Kernel PCA: Equivalent formulation

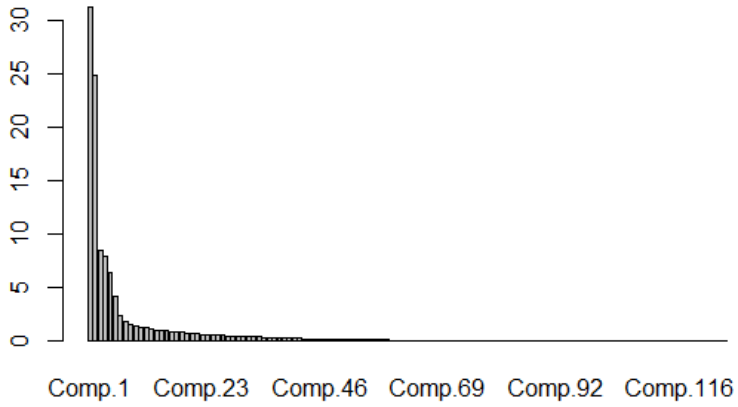
1. Solve $\boldsymbol{K'a_i} = \lambda_i' \boldsymbol{a_i}, i = 1,..M$
   - $\boldsymbol{K} = \left\| K(\boldsymbol{x}_i, \boldsymbol{x}_j), i, j = 1,..n \right\|$
   - Centering $\boldsymbol{K'} = \boldsymbol{K} - \boldsymbol{1_n K} - \boldsymbol{K 1_n} + \boldsymbol{1_n K 1_n}$
   - $\lambda_i = \lambda_i'/n$
2. Scores for PC_i: $z_i(\boldsymbol{x}) = \sum_{i=1}^{n} a_{in} K(\boldsymbol{x}, \boldsymbol{x}_n)$

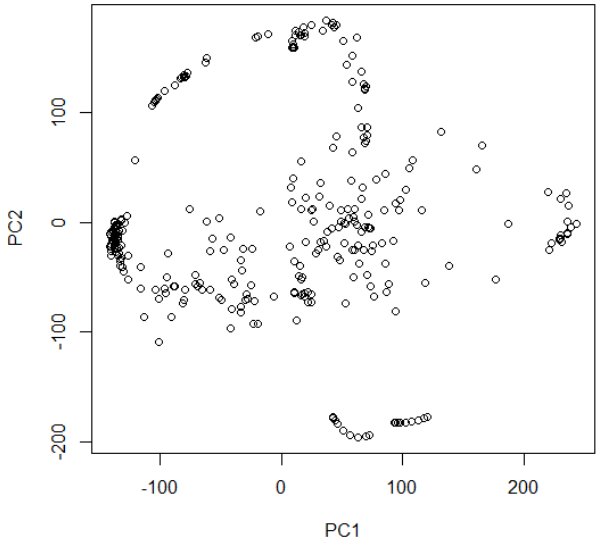
- There are at most *n* eigenvectors even if *p>>n*

Kernel PCA in R

- Use kpca() in **kernlab**

```
library(kernlab)
K <- as.kernelMatrix(crossprod(t(x)))
res=kpca(K)
barplot(res@eig)
plot(res@rotated[,1], res@rotated[,2], xlab="PC1",
ylab="PC2")
```





Feature assessment

- Which features are important?
  - Ex: Which protein values differ between normal and cancer samples

- P-values in our predictive models can not be computed (too few observations)

- → Traditional hypothesis testing is used

Feature assessment

- Individual gene: t-test

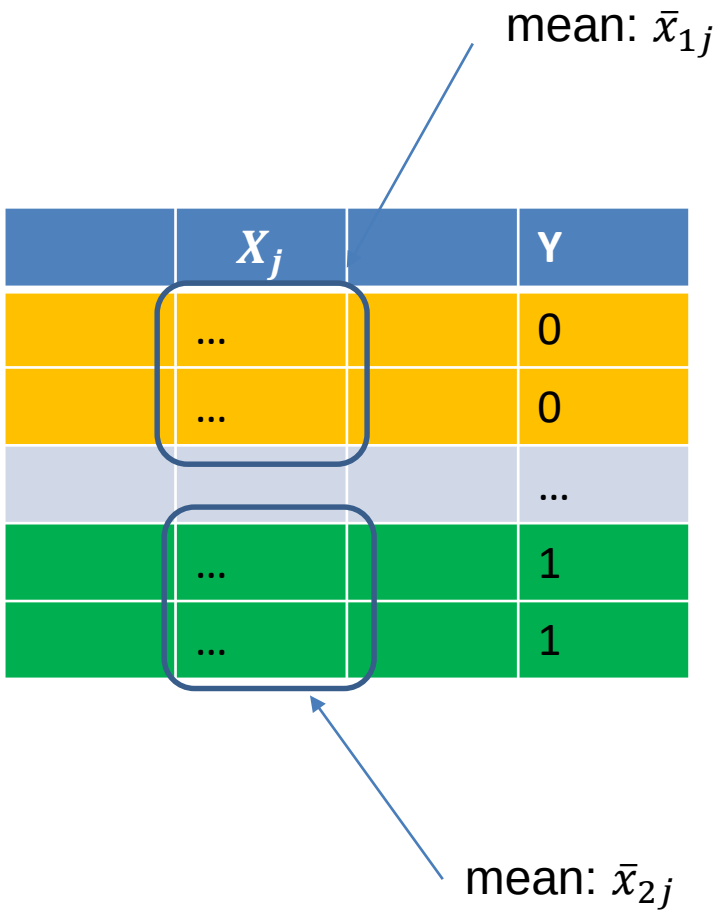
H_{0j}: $treatment\ has\ no\ effect\ on\ gene\ j$

H_{1j}: $treatment\ has\ an\ effect\ on\ gene\ j$

$$t = \frac{\bar{x}_{2j} - \bar{x}_{1j}}{se_j}$$

- Alternatively, nonparametric tests (permutation tests) can be used to compare two populations
- Testing hypothesis for all genes?→multiple hypothesis testing
- Control family-wise error rate
  - Bonferroni correction: $\alpha' = \alpha/M$
  - Ex: α=0.05, M=12000→$\alpha' \approx 10^{-6}$

In practice, no genes with such small p-values



Feature assessment

- Hypothesis testing Voice Rehabilitation
  - Feature ”MFCC_2nd.coef”

```
res=t.test(MFCC_2nd.coef~Quality,data=data,
alternative="two.sided")
res$p.value
```

```
> res$p.value
[1] 1.21246e-11
```

```
res=oneway_test(MFCC_2nd.coef~as.factor(Qu
ality), data=data,paired=FALSE)
pvalue(res)
```

```
> pvalue(res)
[1] 3.166942e-09
```

Feature assessment

- Alternative: **false discovery rate** (FDR)
  - Can not be exactly computed in practice

	Called nonsignif	Called signif	Total
H0 true	U	V	M0
H0 false	T	S	M1
Total	M-R	R	M

$$FDR = E\left(\frac{V}{R}\right)$$

Feature assessment

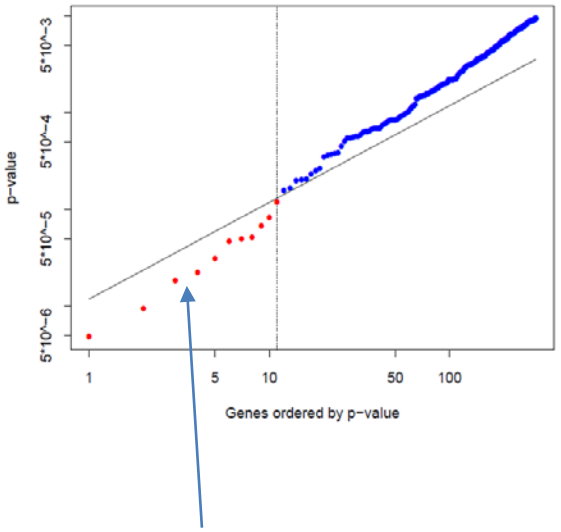
- Benjamini-Hochberg method (BH method)
  - Shown that $FDR(BH) < \alpha$ for independent hypotheses
  - →we can control FDR!

Algorithm 18.2 *Benjamini–Hochberg (BH) Method.*

1. Fix the false discovery rate α and let $p_{(1)} \leq p_{(2)} \leq \cdots \leq p_{(M)}$ denote the ordered p-values
2. Define

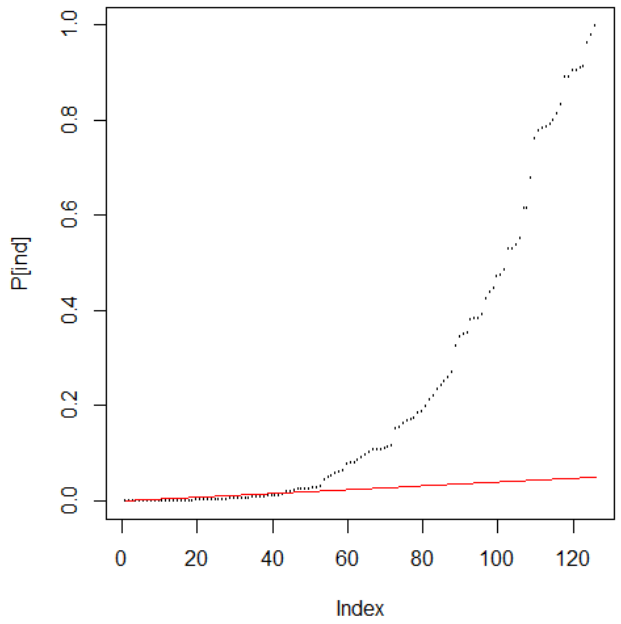
$$L = \max\left\{j : p_{(j)} < \alpha \cdot \frac{j}{M}\right\}.$$

3. Reject all hypotheses H_{0j} for which $p_j \leq p_{(L)}$, the BH rejection threshold.



Feature assessment

- Voice rehabilitation




```
> cat( paste( Feats, collapse='\n' ) )
MFCC_2nd.coef
IMF..NSR_SEO
MFCC_1st.coef
IMF..NSR_entropy
MFCC_0th.coef
DFA
Log.energy
HNR..HNR_dB_Praat_std
MFCC_3rd.coef
VFER..SNR_SEO
IMF..SNR_TKEO
IMF..SNR_entropy
Jitter..pitch_PQ5_classical_Schoentgen
Jitter..pitch_percent
Jitter..F0_abs_dif
Jitter..F0_PQ5_classical_Schoentgen
Jitter..F0_dif_percent
VFER..SNR_TKEO1
Shimmer..Ampl_TKEO_prc25
Shimmer..Ampl_AM
VFER..NSR_TKEO1
Shimmer..Ampl_abs0th_perturb
VFER..SNR_TKEO
NHR..NHR_Praat_std
OQ..std_cycle_closed
VFER..NSR_SEO
Jitter..F0_FM
Jitter..pitch_FM
Shimmer..Ampl_TKEO_prc75
Jitter..pitch_abs
OQ..std_cycle_open
Jitter..F0_TKEO_prc5
GNE..std
Jitter..pitch_PQ5_generalised_Schoentgen
Jitter..F0_TKEO_mean
PPE
Shimmer..Ampl_TKEO_prc95
X1st.delta
Jitter..F0_PQ5_generalised_Schoentgen
Shimmer..Ampl_PQ3_generalised_Schoentgen
Shimmer..Ampl_PQ5_generalised_Schoentgen
Shimmer..Ampl_PQ11_generalised_Schoentgen
Jitter..pitch_TKEO_prc25
```