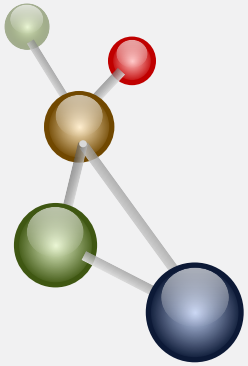
Zentralinstitut für
Seelische Gesundheit
Landesstiftung
des öffentlichen Rechts



Foundations of machine learning and high-dimensional data analysis

Emanuel Schwarz, PhD

Overview

Day 1

- Introduction
- Linear methods
  - Regression
  - Classification
- Model assessment and selection

Day 2

- Non-linear and probabilistic methods
- Model aggregation
- Variable selection

Day 3

- Predicting multiple classes or multiple outcomes
- Real-world challenges of machine learning

Goals

For a given method

- Be able to describe how it works
- Apply the method on data using R
- Export, describe and interpret (possibly visual) output from R
- Describe advantages and disadvantages of the method (bullet points)

Goals

Random Forest

How it works

Random forest is an ensemble learning method that …

How to use it in R

randomForest(Y~., data=X, ntree = 500, importance=TRUE)

Output from R

Screenshot 1

Screenshot 2

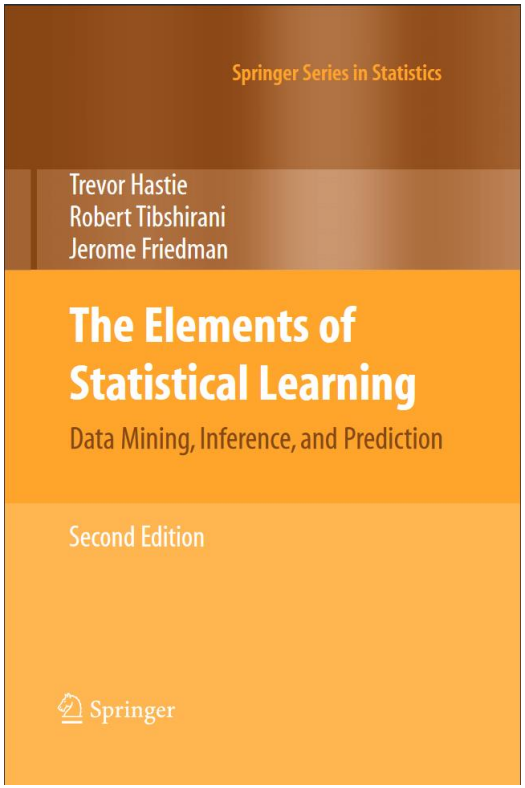
Description and interpretation

The variable importance ranks features by the GINI index …

In this data, the most important predictors are …

Advantages and disadvantages of random forest

- Random forests can incorporate categorical predictors
- Random forests often inferior to simpler, linear methods
- …


Springer Series in Statistics
Trevor Hastie
Robert Tibshirani
Jerome Friedman
The Elements of Statistical Learning
Data Mining, Inference, and Prediction
Second Edition
Springer

Introduction - short history of machine learning

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

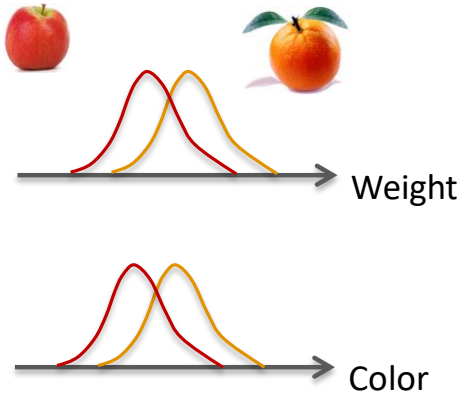
1936	**„Fisher's linear discriminant “ by Ronald Fisher**
1950	Alan Turing proposes „learning machine“ that could become artificially intelligent
1951	First neural nework by Marvin Minsky and Dean Edmonds
1952	First algorithms to play checkers developed by Arthur Samuel at IBM
1957	Invention of the perceptron by Frank Rosenblatt
1962	„proximity algorithm“ by Sebesteyen (nearest neighbor classification)
1963	Invention of a first version of Support Vector Machines
1967	Nearest Neighbor algorithm by Cover and Hart
1986	Invention of backpropagation (predecessor already described in 1970)
1992	ANN plays backgammon
1995	Invention of Random Forests
1995	Invention of soft margin SVM
1997	IMB‘s Deep Blue beats Kasparov at chess
2012	Recognition of cats on youtoube
2015,16,17	Google‘s AlphaGo beats human player at Go

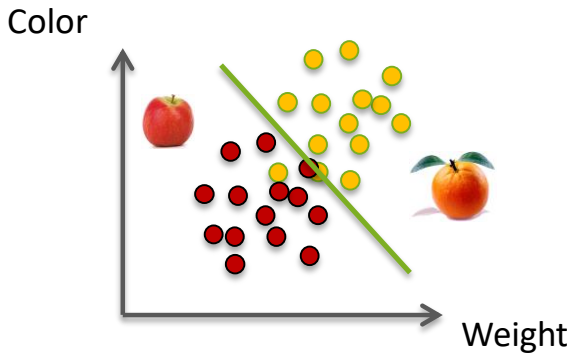
https://en.wikipedia.org/wiki/Timeline_of_machine_learning

Introduction - utility of machine learning

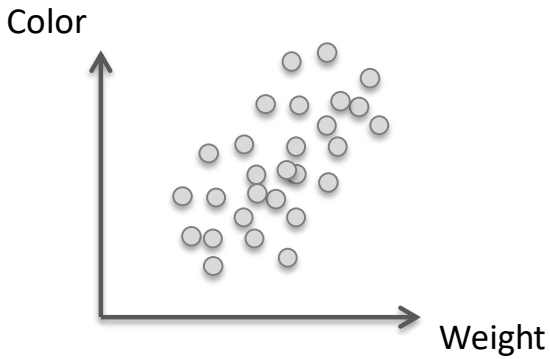
Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

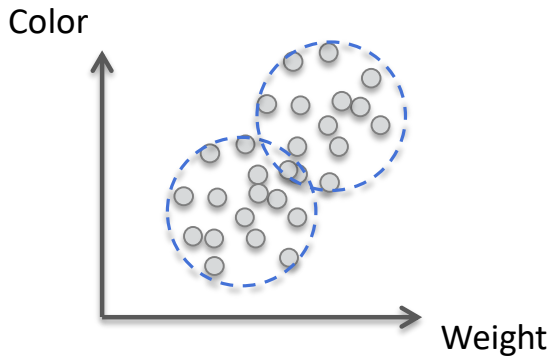
Supervised learning
(learning by example)





Unsupervised learning
(clustering)





Introduction - some example applications

Zentralinstitut für
Seelische Gesundheit
Landesstiftung
des öffentlichen Rechts

Medical diagnosis and prediction

- detection and classification of cancer using imaging data
- classification of malignancy from genetic and proteomics data

Search engines

Spam filtering

Computer games

Speech recognition

Face recognition

Fraud detection

Financial trading

Recognition of hand writing

Prediction of website user behavior

Introduction - terminology

Quantitative and qualitative (also known as categorical, discrete, factors) variables

Predictors, features and outcome

Dependant and independant variables

Grouping, class

Observation, instance

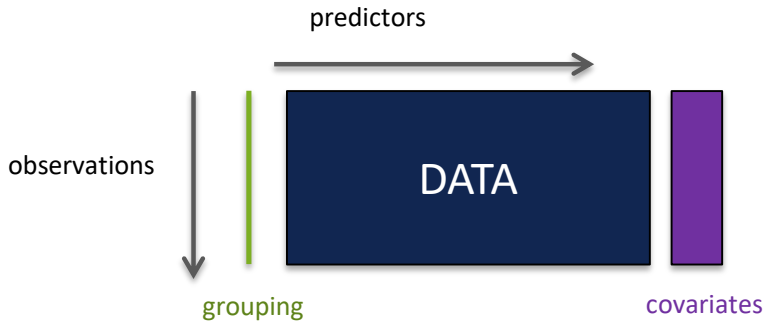
Covariates

Decision rule, classifier

Prediction and classification

Training data, test data

Supervised and unsupervised learning



Introduction - a first R session

A simple vector

```
> vec1 = c(1,2,3,4,5,6)
> vec1
[1] 1 2 3 4 5 6
```

A simple matrix

```
> mat1 = matrix( vec1, nrow = 3)
> mat1
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

A simple data frame

```
> dat1 = data.frame(mat1)
> dat1
  X1 X2
1  1  4
2  2  5
3  3  6
```

A simple factor

```
> fac1 = as.factor( c("a", "b", "c") )
> fac1
[1] a b c
Levels: a b c
```

Concatenate factor and data frame

```
> dat2 = cbind(fac1, dat1)
> dat2
  fac1 X1 X2
1    a  1  4
2    b  2  5
3    c  3  6
```

```
> is.numeric( dat2[,2] )
[1] TRUE
> is.factor( dat2[,1] )
[1] TRUE
```

Dataframe contains variables of different types

Introduction - a first R session

for loop

```
> for ( i in 1:10 ){

        print ( i )

  }
```

Storing the output

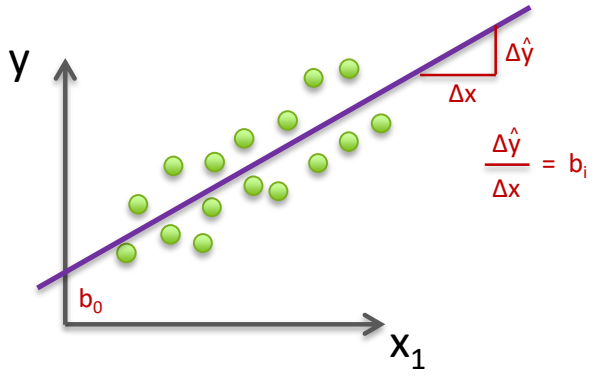
```
> result = numeric()
> for ( i in 1:10 ){

        result = c( result, i )

  }
> result
[1]  1  2  3  4  5  6  7  8  9 10
```

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

- Estimate values of a variable given values of multiple other variables

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n$$



Covariance: $cov_{XY} = E[(X - \mu_X)(Y - \mu_Y)]$

Correlation: $cor_{XY} = E[(X - \mu_X)(Y - \mu_Y)]/(\sigma_X \sigma_Y)$

Linear methods – ridge regression

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

- Reduction of model complexity through shrinkage of coefficients towards 0.

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{N} \underbrace{\|y - X\beta\|_2^2}_{\text{Loss}} + \underbrace{\lambda \|\beta\|_2^2}_{\text{Penalty}} \right\}$$

Loss
Least square objective

Penalty (λ * sum of squared coefficients)
Also known as L2 regularization

Advantages

- Useful for reduction of model complexity in datasets with p > n
- Works even if predictors are correlated

Disadvantages

- Coefficients never actually 0

Linear methods – LASSO

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

- Reduction of model complexity through shrinkage of coefficients towards 0.
- Set coefficients of useless predictors to 0.

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{N} \| y - X\beta \|_2^2 + \lambda \| \beta \|_1 \right\}$$

Loss
Least square objective

Penalty (λ * sum of absolute values of coefficients)
Also known as L1 regularization

Advantages

- Useful for reduction of model complexity in datasets with p > n
- Helps interpretation of complex datasets (higher sparsity)

Disadvantages

- Problematic with correlated variables

Regression / regularization - examples

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Load dataset in R

```
> library(mlbench)
> data(BreastCancer)
> dim(BreastCancer)
[1] 699  11
```

```
> head(BreastCancer, 2)
```

	Id	Cl.thickness	Cell.size	Cell.shape	Marg.adhesion	Epith.c.size	Bare.nuclei	Bl.cromatin	Normal.nucleoli	Mitoses	Class
1	1000025	5	1	1	1	2	1	3	1	1	benign
2	1002945	5	4	4	5	7	10	3	2	1	benign

1. Sample code number: id number
2. Clump Thickness: 1 - 10
3. Uniformity of Cell Size: 1 - 10
4. Uniformity of Cell Shape: 1 - 10
5. Marginal Adhesion: 1 - 10
6. Single Epithelial Cell Size: 1 - 10
7. Bare Nuclei: 1 - 10
8. Bland Chromatin: 1 - 10
9. Normal Nucleoli: 1 - 10
10. Mitoses: 1 - 10
11. Class

Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Regression / regularization - examples

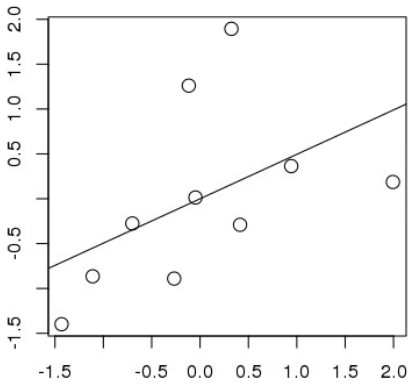

```
> var1 = as.numeric(BreastCancer[,2])
> var2 = as.numeric(BreastCancer[,3])
> lm( var1 ~ var2 )

Call:
lm(formula = var1 ~ var2)

Coefficients:
(Intercept)         var2
     2.5524       0.5951
```

Scale variables

```
> lm( scale( var1 ) ~ scale( var2 ) )

Call:
lm(formula = scale(var1) ~ scale(var2))

Coefficients:
(Intercept)  scale(var2)
 -7.857e-17    6.449e-01

> cor(var1,var2)
[1] 0.6449125
```

Plotting

```
> plot(scale(var2), scale(var1), cex = 1.5)
> abline(  lm(scale(var1) ~ scale(var2)) )
```

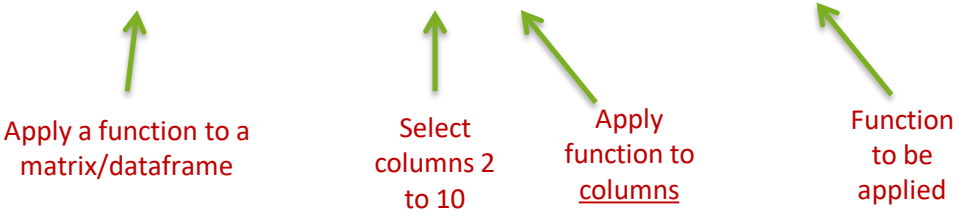


Regression / regularization - examples

Multiple regression

convert matrix to scaled numeric values

```
> BCnum  = apply(  BreastCancer[,2:10] ,  2 ,   function(x){  scale( as.numeric(x) )  })
```




```
> BCnum = data.frame(BCnum)
> model.mult = lm( Cl.thickness ~ . , data=BCnum )
> model.mult
Call:
lm(formula = Cl.thickness ~ ., data = BCnum)

Coefficients:
   (Intercept)       Cell.size      Cell.shape    Marg.adhesion    Epith.c.size     Bare.nuclei     Bl.cromatin  Normal.nucleoli         Mitoses
      0.006942        0.186738        0.254488        -0.079253         0.008610         0.243246         0.046748         0.061167         0.055528
```

Regression / regularization - examples

A short excursion into statitics

```
> model.mult
Call:
lm(formula = Cl.thickness ~ ., data = BCnum)

Coefficients:
   (Intercept)       Cell.size      Cell.shape    Marg.adhesion    Epith.c.size     Bare.nuclei    Bl.cromatin  Normal.nucleoli
      0.006942         0.186738         0.254488        -0.079253         0.008610         0.243246         0.046748         0.061167
       Mitoses
      0.055528
```

```
> summary(model.mult)

Call:
lm(formula = Cl.thickness ~ ., data = BCnum)
Residuals:
    Min      1Q  Median      3Q     Max
-2.15312 -0.67343  0.01769  0.55177  2.00991

Coefficients:
                 Estimate   Std. Error  t value Pr(>|t|)
(Intercept)      0.006942   0.027935    0.249    0.803808
Cell.size        0.186738   0.074294    2.513    0.012186 *
Cell.shape       0.254488   0.070459    3.612    0.000327 ***
Marg.adhesion   -0.079253   0.043650   -1.816    0.069867 .
Epith.c.size     0.008610   0.044514    0.193    0.846683
Bare.nuclei      0.243246   0.044062    5.521    4.83e-08 ***
Bl.cromatin      0.046748   0.047175    0.991    0.322063
Normal.nucleoli  0.061167   0.043536    1.405    0.160485
Mitoses          0.055528   0.032628    1.702    0.089238 .
---

Residual standard error: 0.73 on 674 degrees of freedom
  (16 observations deleted due to missingness)
Multiple R-squared:  0.4753,  Adjusted R-squared:  0.469
F-statistic: 76.31 on 8 and 674 DF,  p-value: < 2.2e-16
```

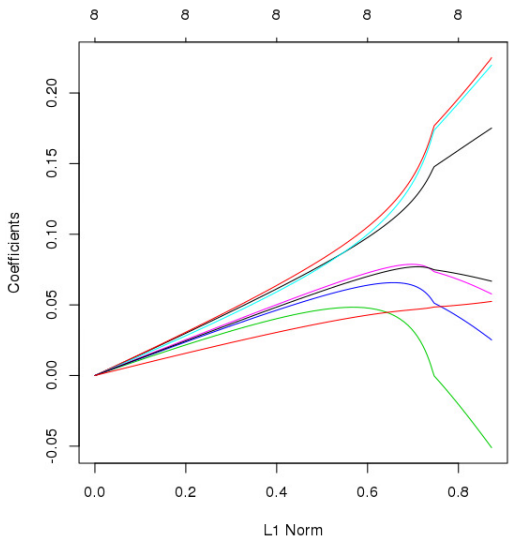
Likely important predictors

Regression / regularization - examples

Regularization – Ridge regression

```
> library( glmnet )
```

```
> BCnum_noNA = na.omit( BCnum )
> X = as.matrix( BCnum_noNA[,-1] )
> Y = as.matrix( BCnum_noNA[, 1] )
> ridge = glmnet(X, Y, family="gaussian", alpha = 0)
> plot(ridge)
```



Non-0 predictors

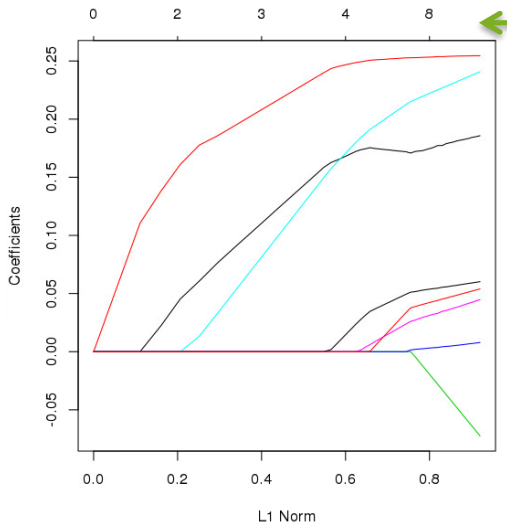
Regression / regularization - examples

Zentralinstitut für
Seelische Gesundheit
Landesstiftung
des öffentlichen Rechts

Regularization – Lasso

> lasso = glmnet(X, Y, family="gaussian", alpha = 1)

> plot(lasso)



Non-0 predictors

> lasso$beta

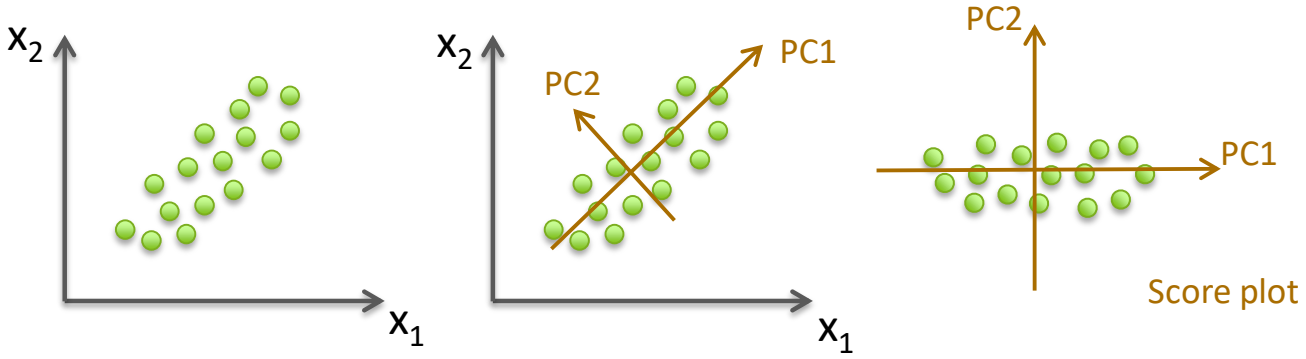
```
Cell.size       . .          .         0.02243836 0.04541849 0.06073417
Cell.shape      . 0.05783163 0.1105257 0.13820453 0.16112711 0.17767798
Marg.adhesion   . .          .         .          .          .
Epith.c.size    . .          .         .          .          .
Bare.nuclei     . .          .         .          .          0.01326851
Bl.cromatin     . .          .         .          .          .
Normal.nucleoli . .          .         .          .          .
Mitoses         . .          .         .          .          .
```

Linear methods – principal components analysis

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

- Reduction of data to a lower-dimensional space
- Each dimension is a linear combination of the original variables
- Dimensions are uncorrelated



Advantages

- Useful for visual inspection of high dimensional data (i.e. to search for outliers, confounders)
- No assumptions about underlying statitical model

Disadvantages

- Often not useful in noisy, high-dimensional datasets

PCA - example

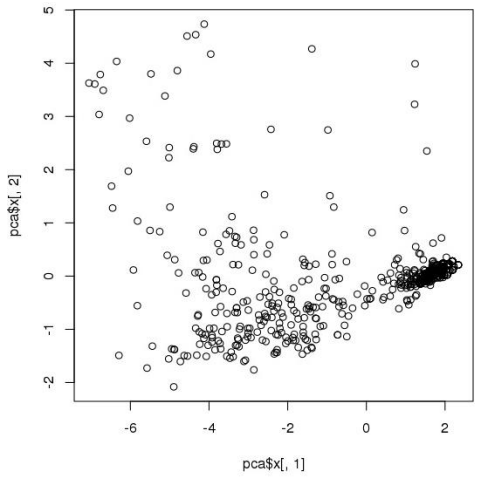
> pca = prcomp(BCnum_noNA)

```
> summary(pca)
Importance of components:
                          PC1    PC2     PC3     PC4    PC5     PC6     PC7
Standard deviation     2.4374 0.8887 0.73595 0.67968 0.6179 0.55074 0.54409
Proportion of Variance 0.6552 0.0871 0.05973 0.05095 0.0421 0.03345 0.03265
Cumulative Proportion  0.6552 0.7423 0.80199 0.85294 0.8950 0.92849 0.96114
                           PC8     PC9
Standard deviation     0.51292 0.29877
Proportion of Variance 0.02901 0.00984
Cumulative Proportion  0.99016 1.00000
```

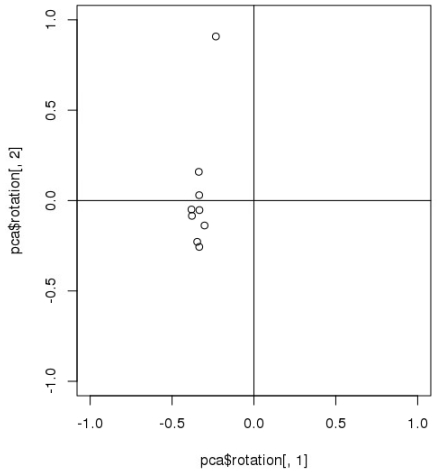
> plot(pca$x[,1] , pca$x[,2])

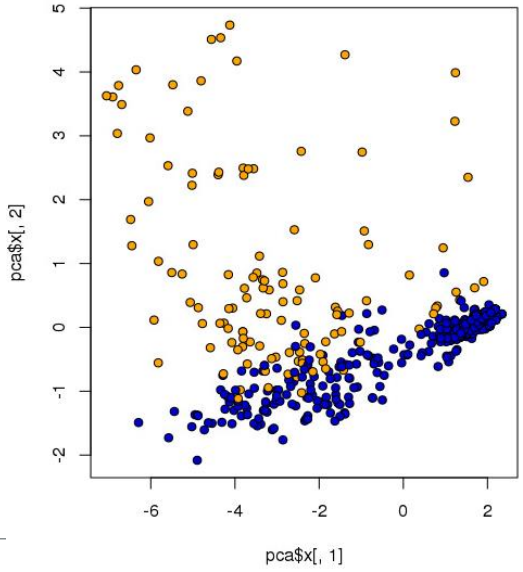
> plot(pca$rotation[,1], pca$rotation[,2], ylim=c(-1,1), xlim=c(-1,1))

Scores plot



Loadings plot





Linear methods – linear discriminant analysis

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

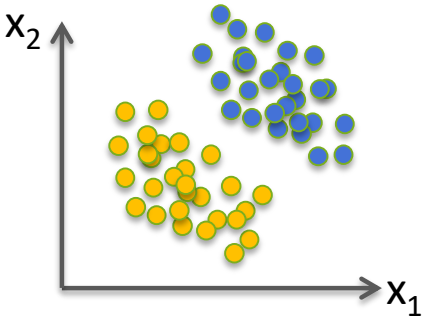
Concept

- Linear combination of features that disciminates two or more classes
- Generalization of Fishers's linear discriminant
- Assumes that conditional probability density functions $p(X|y=k)$ are normally distributed with mean and covariance parameters (μ_k, Σ_k)

$$p(X|y=k) = \frac{1}{(2\pi)^n|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(X-\mu_k)^t \Sigma_k^{-1}(X-\mu_k)\right)$$

Bayes Rule

$$p(y=k|X) = \frac{p(y=k)\, p(X|y=k)}{p(X)}$$



$p(y=k)$ class proportion from data

μ_k empirical class means from data

Σ_k covariance matrices from data

Linear methods – linear discriminant analysis

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

$$p(X|y=k) = \frac{1}{(2\pi)^n|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(X-\mu_k)^t \Sigma_k^{-1}(X-\mu_k)\right)$$

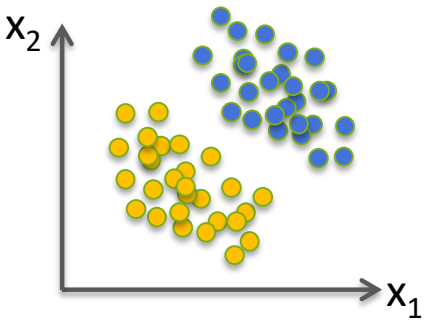
Bayes Rule

$$p(y=k|X) = \frac{p(y=k)\, p(X|y=k)}{p(X)}$$

$p(y=k)$ class proportion from data

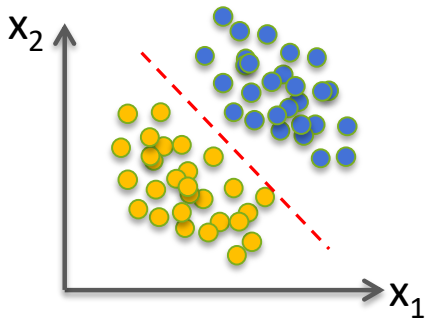
μ_k empirical class means from data

Σ_k covariance matrices from data



LDA: same covariance matrix in all classes

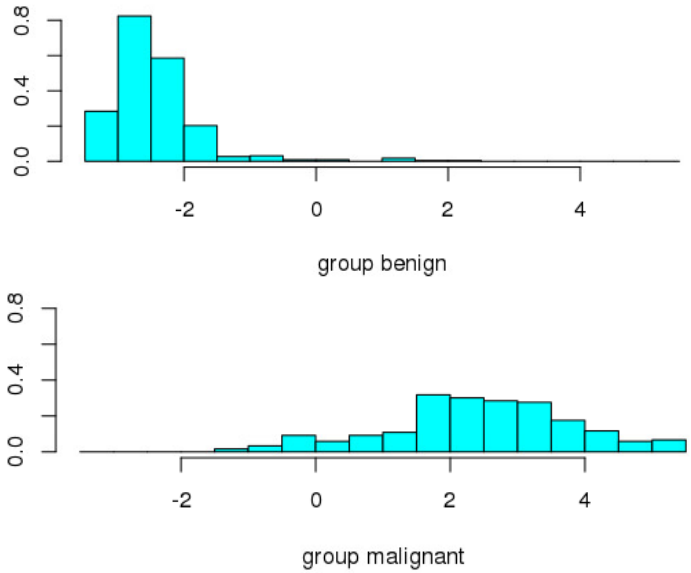
$$\log\frac{p(y=1|X)}{p(y=0|X)} = 0$$



LDA – example


```
> library( MASS )

> Y = BreastCancer$Class[as.numeric( rownames(BCnum_noNA) )]

> lda_model  =  lda(  BCnum_noNA, Y  )
> lda_model
```

```
Call:
lda(BCnum_noNA, Y)

Prior probabilities of groups:
   benign malignant
0.6500732 0.3499268

Group means:
          Cl.thickness  Cell.size Cell.shape Marg.adhesion Epith.c.size
benign      -0.5163031 -0.5991139 -0.6033235    -0.5113227   -0.5003454
malignant    0.9839489  1.1282891  1.1283071     0.9732182    0.9530493
          Bare.nuclei Bl.cromatin Normal.nucleoli    Mitoses
benign     -0.6031546  -0.5554687      -0.5258297 -0.3055827
malignant   1.1205047   1.0405038       0.9794192  0.5907003

Coefficients of linear discriminants:
                     LD1
Cl.thickness    0.514311678
Cell.size       0.383932596
Cell.shape      0.267705785
Marg.adhesion   0.135568441
Epith.c.size    0.128493457
Bare.nuclei     0.952535309
Bl.cromatin     0.269304190
Normal.nucleoli 0.325890665
Mitoses         0.009669641
```



LDA – example


```
> library( MASS )

> Y = BreastCancer$Class[as.numeric( rownames(BCnum_noNA) )]

> lda_model  =  lda(  BCnum_noNA, Y  )
> lda_model
```

```
Call:
lda(BCnum_noNA, Y)

Prior probabilities of groups:
   benign malignant
0.6500732 0.3499268

Group means:
          Cl.thickness  Cell.size Cell.shape Marg.adhesion Epith.c.size
benign      -0.5163031 -0.5991139 -0.6033235    -0.5113227   -0.5003454
malignant    0.9839489  1.1282891  1.1283071     0.9732182    0.9530493
          Bare.nuclei Bl.cromatin Normal.nucleoli    Mitoses
benign     -0.6031546  -0.5554687      -0.5258297 -0.3055827
malignant   1.1205047   1.0405038       0.9794192  0.5907003

Coefficients of linear discriminants:
                     LD1
Cl.thickness    0.514311678
Cell.size       0.383932596
Cell.shape      0.267705785
Marg.adhesion   0.135568441
Epith.c.size    0.128493457
Bare.nuclei     0.952535309
Bl.cromatin     0.269304190
Normal.nucleoli 0.325890665
Mitoses         0.009669641
```

Predict model

```
> pred = predict( lda_model, Bcnum_noNA)

> head( pred$posterior, 3)
```

```
        benign    malignant
1 0.999985969 1.403109e-05
2 0.001927822 9.980722e-01
3 0.999991529 8.470572e-06
```

Manually

```
means <- colSums(lda_model$prior * lda_model$means)
scaling <- lda_model$scaling
dm <- scale(lda_model$means, center = means, scale = FALSE) %*% scaling
x  <- scale(BCnum_noNA, center = means, scale = FALSE) %*% scaling
dist <- matrix(0.5 * rowSums(dm^2) - log(prior), nrow(BCnum_noNA),
        length(prior), byrow = TRUE) - x %*%  t(dm)

dist <- exp(-(dist - apply(dist, 1L, min, na.rm = TRUE)))

posterior <- dist/drop(dist %*% rep(1, 2))
```

```
        benign    malignant
1 0.999985969 1.403109e-05
2 0.001927822 9.980722e-01
3 0.999991529 8.470572e-06
```

LDA – advantages / disadvantages

Advantages

- Simple model that can generalize well when based on small number of predictors
- Fast to compute (no paramters to tune)

Disadvantages

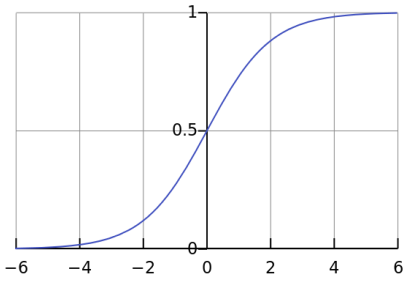
- May not work well when groups are not balanced
- Does not allow strong correlation between predictors
- Usually poor performance in large datasets without feature selection
- All predictors receive non-zero weight
- Decision boundary is linear (with class -1 decision discriminant functions)
- Assumes gaussian distribution of likelihood
- Assumes equal variance across groups -> differences in variance are not detected

Linear methods – logistic regression

Concept

- Regression with categorical dependent variable
- Binary logsitic regression estimates probability of a given outcome
- Closely linked to odds ratio estimation

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$



$$t = \beta_0 + \beta_1 x$$

$$F(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

$$g(F(x)) = ln\left(\frac{F(x)}{1 - F(x)}\right) = \beta_0 + \beta_1 x$$

Logistic regression - example

> BC_dataframe = data.frame(X = BCnum_noNA, Y = Y)

> logit_model = glm(Y ~ ., data = BC_dataframe, family = binomial(link = "logit"))

> logit_model

```
Call:  glm(formula = Y ~ ., family = binomial(link = "logit"), data = BC_dataframe)

Coefficients:
      (Intercept)     X.Cl.thickness         X.Cell.size        X.Cell.shape
          -1.13033            1.50646            -0.01916             0.95906
  X.Marg.adhesion     X.Epith.c.size       X.Bare.nuclei       X.Bl.cromatin
           0.94409            0.21398             1.39569             1.09041
X.Normal.nucleoli          X.Mitoses
           0.65052            0.91728

Degrees of Freedom: 682 Total (i.e. Null);  673 Residual
Null Deviance:      884.4
Residual Deviance: 102.9        AIC: 122.9
```

$$ln\left(\frac{\mathrm{F}(x)}{1-\mathrm{F}(x)}\right) = \beta_0 + \beta_1 x_1 + \ldots + \beta_n x_n$$

-> 1 SD increase in Cl.thickness increases log odds of malignancy by 1.50646

> summary(logit_model)

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.4841  -0.1153  -0.0619   0.0222   2.4698

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.13033    0.32303  -3.499 0.000467 ***
X.Cl.thickness    1.50646    0.39988   3.767 0.000165 ***
X.Cell.size      -0.01916    0.63799  -0.030 0.976039
X.Cell.shape      0.95906    0.68533   1.399 0.161688
X.Marg.adhesion   0.94409    0.35250   2.678 0.007400 **
X.Epith.c.size    0.21398    0.34674   0.617 0.537159
X.Bare.nuclei     1.39569    0.34195   4.082 4.47e-05 ***
X.Bl.cromatin     1.09041    0.41789   2.609 0.009073 **
X.Normal.nucleoli 0.65052    0.34467   1.887 0.059115 .
X.Mitoses         0.91728    0.56387   1.627 0.103788
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 884.35  on 682  degrees of freedom
Residual deviance: 102.89  on 673  degrees of freedom
AIC: 122.89
```

Logistic regression - example

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Risk ratio and odds ratio

	Placebo	Treatment
No recovery	N_P	N_T
Recovery	R_P	R_T

„Risk“ of recovery given treatment $R_T / (R_T + N_T)$

„Risk“ of recovery given placebo $R_P / (R_P + N_P)$

Risk ratio $$\frac{R_T / (R_T + N_T)}{R_P / (R_P + N_P)}$$

„Odds“ of recovery given treatment R_T / N_T

„Odds“ of recovery given placebo R_P / N_P

Odds ratio $$\frac{R_T / N_T}{R_P / N_P} = \frac{R_T \cdot N_P}{R_P \cdot N_T}$$

$$ln\left(\frac{F(x)}{1 - F(x)}\right) = \beta_0 + \beta_1 x_1$$

$$F(x) = \frac{\exp(\beta_0 + \beta_1 x_1)}{1 + \exp(\beta_0 + \beta_1 x_1)}$$

Logistic regression - example

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

$$F(x) = \frac{\exp(\beta_0 + \beta_1 x_1)}{1 + \exp(\beta_0 + \beta_1 x_1)}$$

	Placebo	Treatment
No recovery	$1 - F(0) = \frac{1}{1 + \exp(\beta_0)}$	$1 - F(1) = \frac{1}{1 + \exp(\beta_0 + \beta_1)}$
Recovery	$F(0) = \frac{\exp(\beta_0)}{1 + \exp(\beta_0)}$	$F(1) = \frac{\exp(\beta_0 + \beta_1)}{1 + \exp(\beta_0 + \beta_1)}$

$$OR = \frac{\frac{\exp(\beta_0 + \beta_1)}{1 + \exp(\beta_0 + \beta_1)} \cdot \frac{1}{1 + \exp(\beta_0)}}{\frac{\exp(\beta_0)}{1 + \exp(\beta_0)} \cdot \frac{1}{1 + \exp(\beta_0 + \beta_1)}} = \frac{\exp(\beta_0 + \beta_1)}{\exp(\beta_0)} = \frac{\exp(\beta_0) \cdot \exp(\beta_1)}{\exp(\beta_0)} = \exp(\beta_1)$$

Multiple logistic regression
conditional odds ratios

-> 1 SD increase in Cl.thickness increases log odds of malignancy by 1.50646

-> OR = exp(1.50646) = 4.51 -> Every 1 SD increase in Cl.thickness increases risk of malignancy by 351%

Logistic regression – advantages / disadvantages

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Advantages

- Simple structure, fast to compute (no paramters to tune)
- Gives probability estimates as predictions
- Odds ratios can be derived directly from beta values
- Less assumptions about data compared to LDA

Disadvantages

- Assumes linear relationship between predictors and logit function
- Strong correlation between predictors requires regularization
- Usually poor performance in large datasets without feature selection

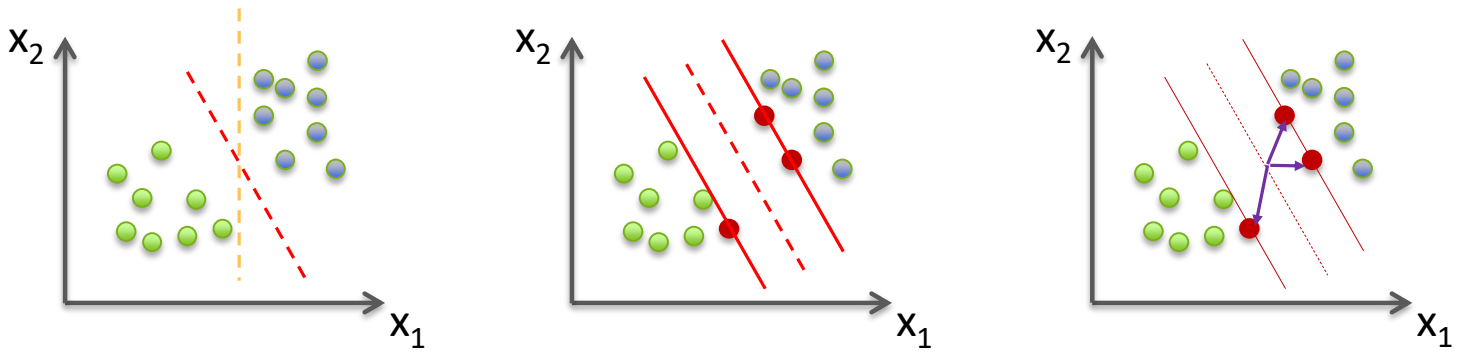
Logistic regression and LDA perform similarly when normality assumptions are met, otherwise logistic regression is preferable

Comparison of logistic regression and linear discriminant analysis: a simulation study
M Pohar, M Blas, S Turk - Metodoloski zvezki, 2004

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

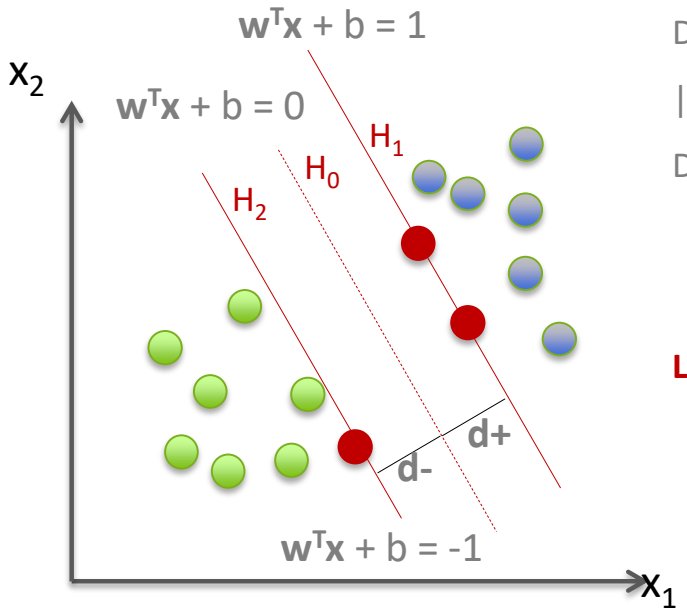
Concept

- Maximization of margin between classes



Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Any hyperplane has the form $\mathbf{w}^T\mathbf{x} + b = 0$



Distance between point (x_0,y_0) and line $(ax + bx + c = 0)$:

$|a\,x_0 + b\,y_0 + c\,| \,/\, \mathrm{sqrt}(a^2 + b^2)$

Distance between H_1 and H_0

$$|\mathbf{w}^T\mathbf{x} + b| \,/\, \|w\| = 1 \,/\, \|w\|$$

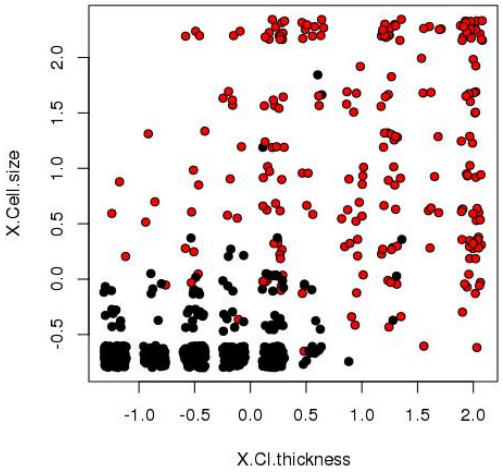
Learning:

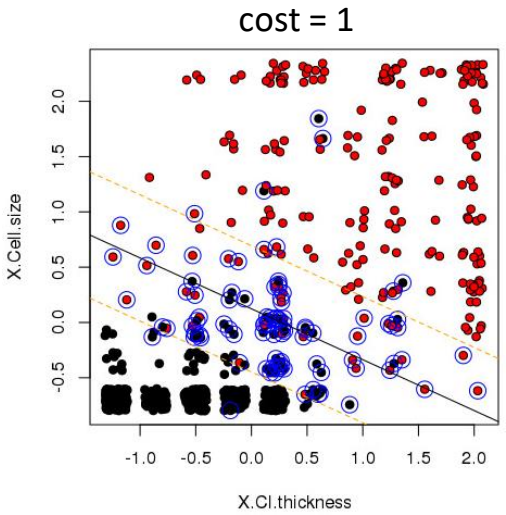
$$\min \|w\|^2 \; + C \sum_i^N \max(0, 1 - y_i f(x_i))$$

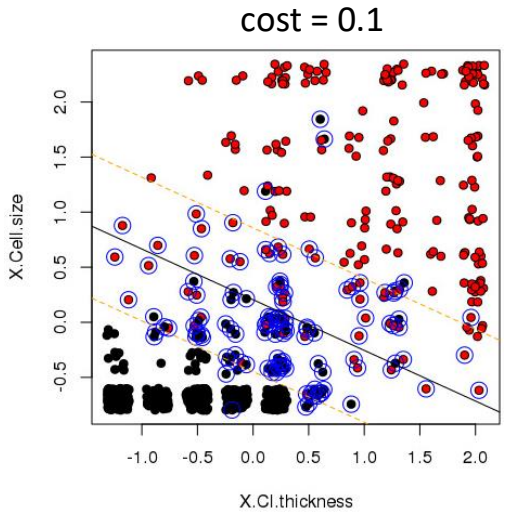
Linear SVM example

Zentralinstitut für
Seelische Gesundheit
Landesstiftung
des öffentlichen Rechts

```
> library( e1071 )

> svm_model  =  svm(  Y ~ ., data = BC_dataframe[,  c(1,2,10)  ],  kernel="linear"  , cost = 1 )

> svm_model
```







Linear SVM – advantages / disadvantages

Advantages

- Compared to non-linear SVM only one hyperparameter to tune
- Uses only points close to decision boundary, potential outliers less important
- Optimization leads to unique solution (unlike neural networks that can have local minima)

Disadvantages

- More complex structure, can take time compute (tuning)
- Non-parametric, more difficult to interpred

Model assessment and selection
Bias-variance trade-off

Zentralinstitut für
Seelische Gesundheit
Landesstiftung
des öffentlichen Rechts

Problem

We aim to simultaneously minimize two sources of error that limit generalizability

- Bias

  wrong assumption of the algorithm (i.e. consistent prediction of something wrong)
- Variance

  fluctuations in the prediction due to sampling variation (variable prediction of something right)

Expected generalization error: Function of Bias, variance, irreducible error

Bias typically results from underfitting (missing important dependencies in the data)

Variance typically results from overfitting (adapting the algorithm too much to training data)

The curse of dimensionality

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Problem

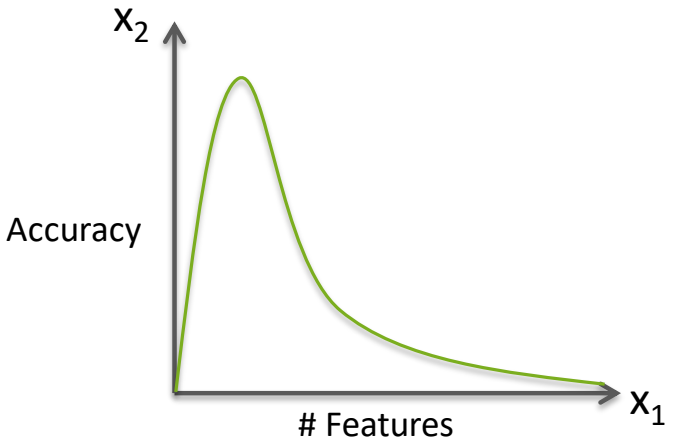
High data dimensionality → volume of data space increases → data becomes sparse

Required amount of data often grows exponentially with dimensionality

Hughes phenomenon:

extreme amounts of data needed to obtain multiple subjects with same value combinations

→ reduction of predictive power with fixed sample size



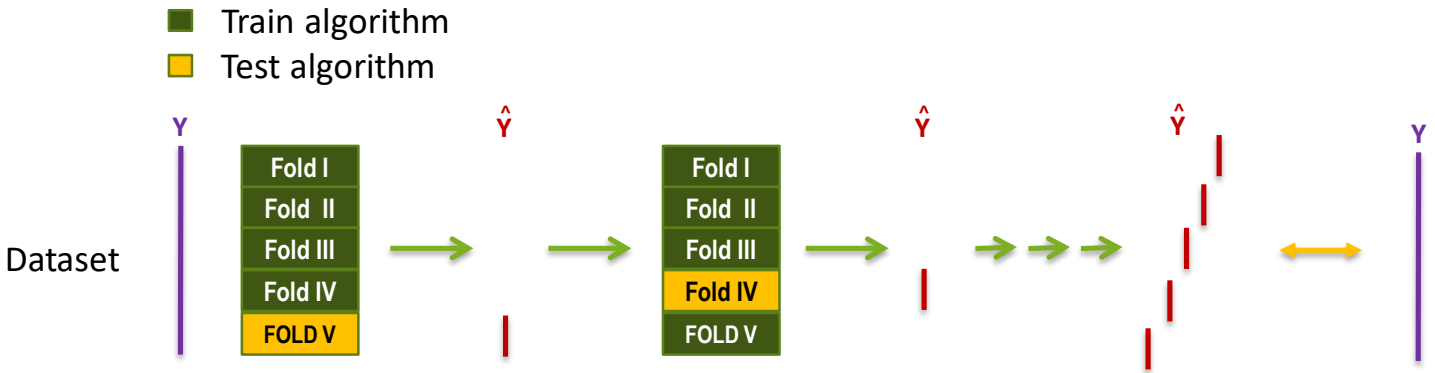
- Linear vs. non-linear classifiers
- Number of estimated parameters
- Dimensionality reduction ?
- Feature selection ?

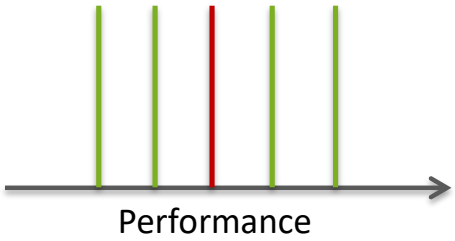
Cross-validation and bootstrapping

Objective

To estimate how well a given algorithm with perform in independent samples

Cross-validation





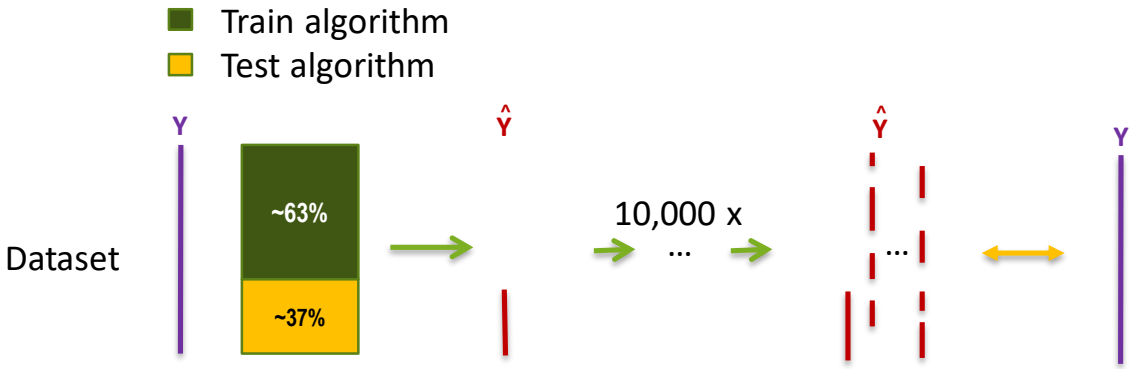
- Can have high variance
- Low bias

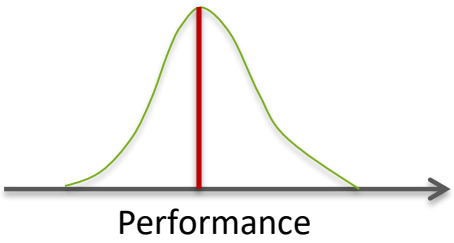
Cross-validation and bootstrapping

Objective

To estimate how well a given algorithm with perform in independent samples

Bootstrapping





- Low variance
- Results can be biased (often too pessimistic)
- Method adaptations (i.e. 632 bootstrap)

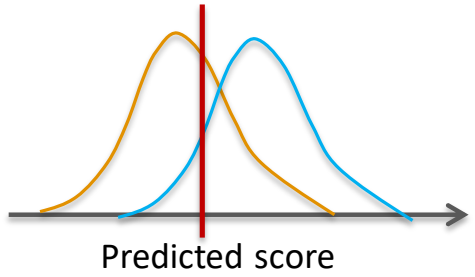
Performance estimation

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Classification

- **Error rate**

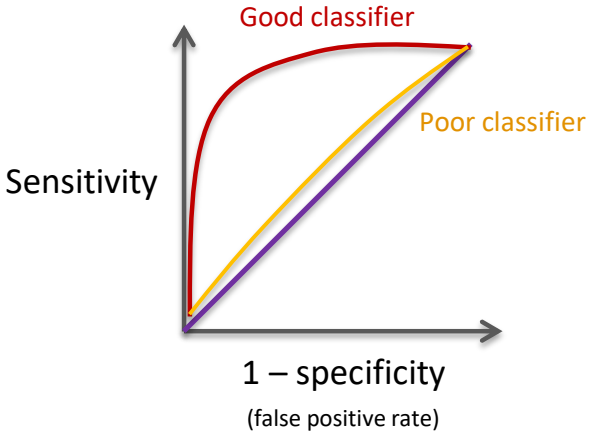
Predicted class \ Real class	0	1
0		
1		



- **Sensitivity and Specificty**

  Sensitivity: Percentage of cases identified correctly
  Specificity: Percentage of non-cases identified correctly

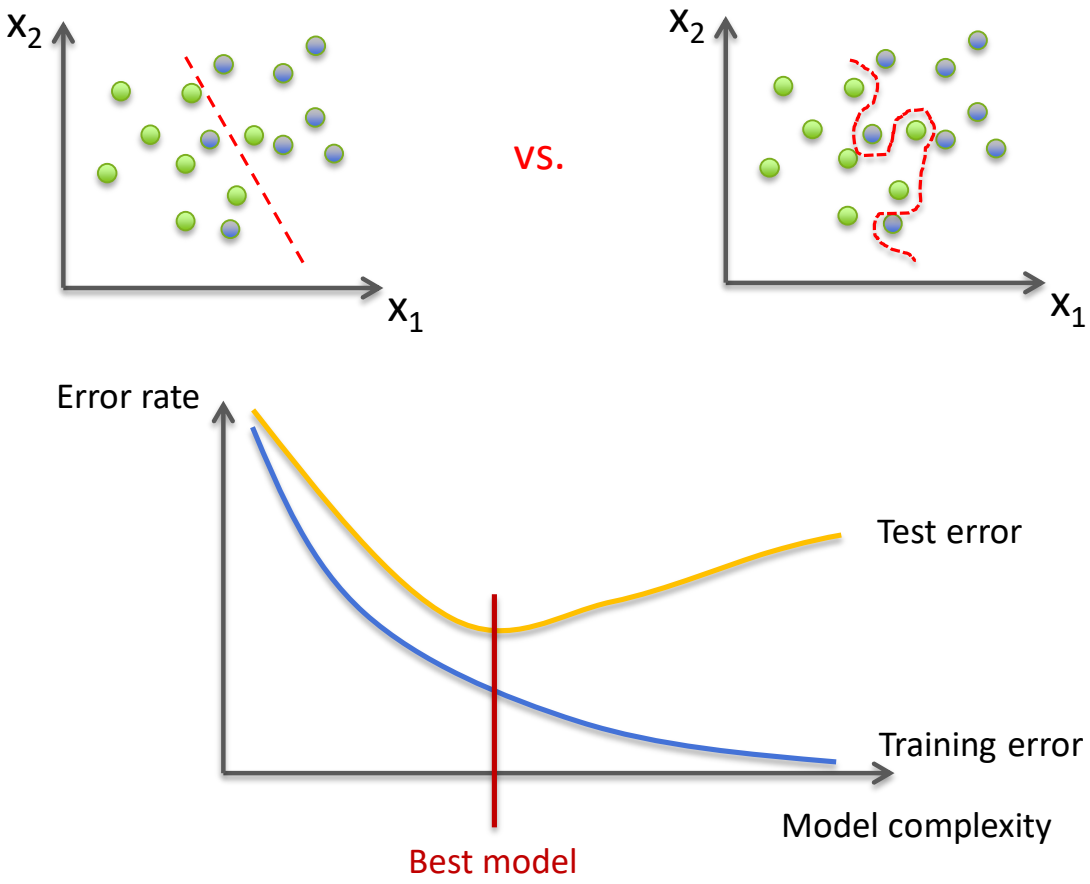
- **AUC (Area under the Receiver-Operating-Characteristic Curve)**



Regression

- **Mean squared error**

Overfitting



Examples

Crossvalidation

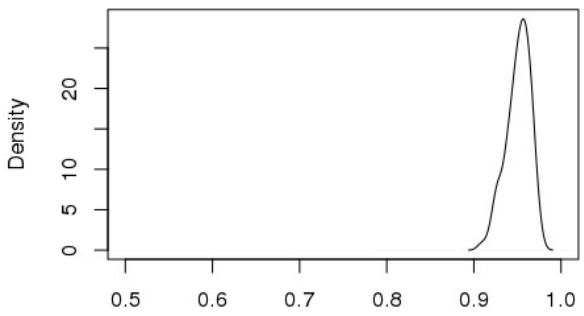
```
> require(caret)
> Cvfolds  = createFolds(   c(1: nrow(BC_dataframe)  )  , k = 10 )

> result = numeric()
> for( i in 1:length(Cvfolds)) {
        trainindex  =  unlist (Cvfolds[-i] )
        testindex   = unlist (Cvfolds[i] )

        lda_model_CV   =  lda(  Y ~ ., data = BC_dataframe[trainindex , ])
        prediction  =  predict (lda_model_CV, BC_dataframe[testindex , ])$class

        sensitivity  = sum(BC_dataframe$Y[testindex] == "malignant" & prediction == "malignant" )/ sum(BC_dataframe$Y[testindex] == "malignant")
        specificity  = sum(BC_dataframe$Y[testindex] == "benign" & prediction == "benign" )/ sum(BC_dataframe$Y[testindex] == "benign")

        result = c(result , (sensitivity  + specificity) / 2  )

}

> mean ( result )
[1] 0.9509605
```

Examples

Bootstrapping

```
> result = numeric()
> for( i in 1: 100 ) {
        trainindex  =  sample (c (1: nrow(BC_dataframe)) , replace=TRUE )
        testindex   = setdiff(c (1: nrow(BC_dataframe)) , trainindex )

        lda_model_CV   =  lda(  Y ~ ., data = BC_dataframe[trainindex , ])
        prediction  =  predict (lda_model_CV, BC_dataframe[testindex , ])$class

        sensitivity  = sum(BC_dataframe$Y[testindex] == "malignant" & prediction == "malignant" )/ sum(BC_dataframe$Y[testindex] == "malignant")
        specificity  = sum(BC_dataframe$Y[testindex] == "benign" & prediction == "benign" )/ sum(BC_dataframe$Y[testindex] == "benign")

        result = c(result , (sensitivity  + specificity) / 2  )

}

> mean ( result )
[1] 0.9507989
> quantile(result , c(0.025, 0.975)
      2.5%       97.5%
0.9219437  0.9719703
```

```
> plot( density( result ) )
```



Overview

Day 1

- Introduction
- Linear methods
  - Regression
  - Classification
- Model assessment and selection

Day 2

- Non-linear and probabilistic methods
- Model aggregation
- Variable selection

Day 3

- Predicting multiple classes or multiple outcomes
- Semi-supervised machine learning

END OF DAY 1

Overview

Day 1

- Introduction
- Linear methods
  - Regression
  - Classification
- Model assessment and selection

Day 2

- Non-linear and probabilistic methods
- Model aggregation
- Variable selection

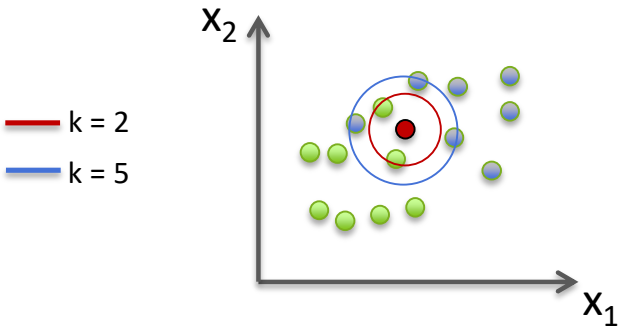
Day 3

- Predicting multiple classes or multiple outcomes
- Semi-supervised machine learning
- Real-world challenges of machine learning

K-nearest neighbors

Concept

- Classification: Majority class of the k nearest neighbors in the feature space
- Regression: Average value of the k nearest neighbors



Advantages

- Simple, non-parametric
- Non-linear decision boundary

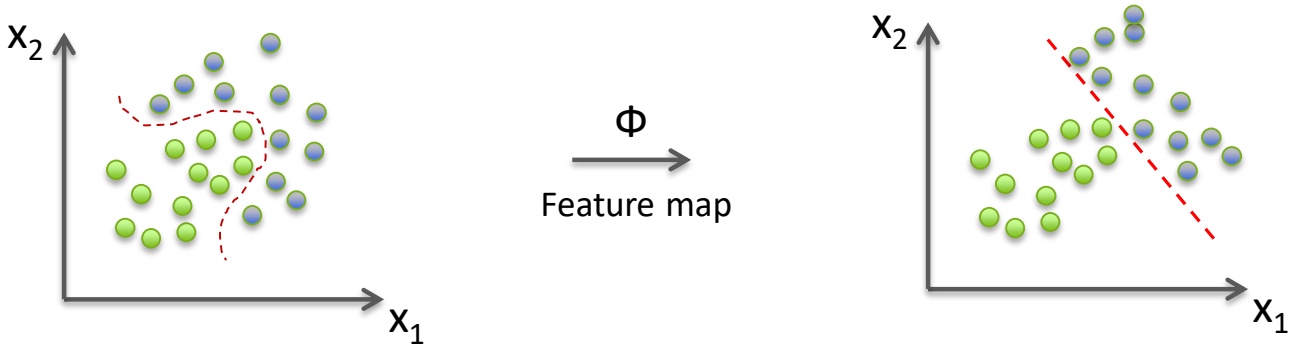
Disadvantages

- Special consideration for skewed class distributions
- Often worse performance than other algorithms

Non-linear support vector machines

Concept

- Maximization of margin between classes
- Uses linear SVM together with a **kernel trick**



- Transformed feature space and hyperplane decision boundary

$$\mathbf{w}^T\mathbf{x} + b \longrightarrow \mathbf{w}^T \Phi(\mathbf{x}) + b$$

- **Learning:**

$$\min\|w\|^2 + C\sum_{i}^{N} \max(0, 1 - y_i f(x_i))$$

Non-linear support vector machines

- **Learning of „primal classifier“** $f(x) = \mathrm{w}^{\mathsf{T}}\,\Phi(\mathrm{x}) + \mathrm{b}$: $\min\|w\|^2 + C\sum_i^N \max(0, 1 - y_i f(x_i))$
- **Learning of „dual classifier“** $f(x) = \sum_{n=1}^{S} \alpha_n y_n k(x_n, x) + b$:

$$\max_{\alpha_n>0} \sum_n \alpha_n - \frac{1}{2}\sum_{n,k} \alpha_n \alpha_k y_n y_k k(x_n, x_k) \quad \text{subject to} \quad 0 \le \alpha_n \le C, \quad \sum_n \alpha_n y_n = 0$$

- Frequently used kernel: Radial Basis Function (RBF)

$$f(x) = \sum_n^S \alpha_n y_n k(x_n, x) + b \xrightarrow{\text{Gaussian Kernel}} f(x) = \sum_n^S \alpha_n y_n exp(-\|x - x_n\|^2/2\sigma) + b$$

The smaller sigma, the more prediction depends on nearest neighbours

$$f(x) = \sum_n^S \alpha_n y_n exp(-\gamma\|x - x_n\|^2) + b$$

- Modulation of γ and *C*

Non-linear support vector machines

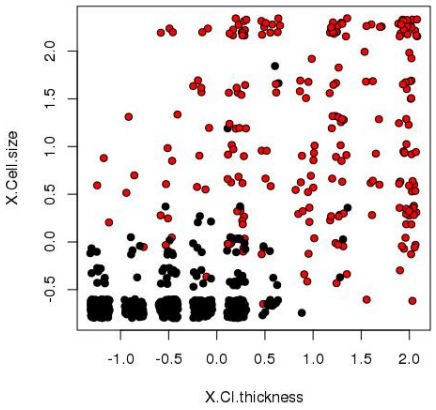
LOAD DATASET

```
data(BreastCancer)

BCnum  = apply(  BreastCancer[,2:10] ,  2 ,   function(x){  scale( as.numeric(x) )  })

BCnum = data.frame(BCnum)

BCnum_noNA = na.omit( BCnum )

Y = BreastCancer$Class[as.numeric( rownames(BCnum_noNA) )]

BC_dataframe = data.frame(X = BCnum_noNA, Y = Y)
```

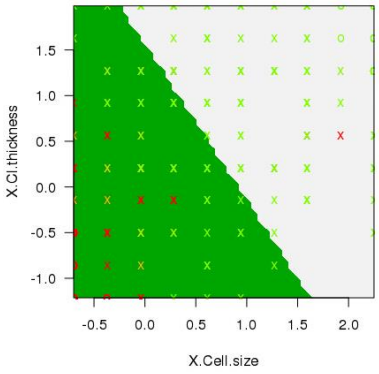
Non-linear support vector machines

Zentralinstitut für
Seelische Gesundheit
Landesstiftung
des öffentlichen Rechts

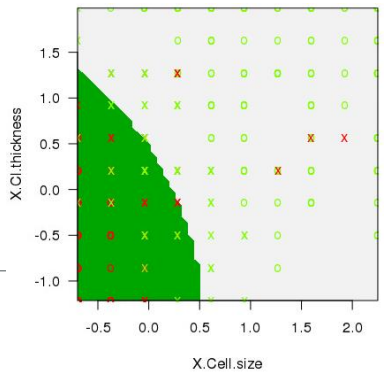
```
> library( e1071 )

> svm_model_rbf  =  svm(  Y ~ ., data = BC_dataframe[,  c(1,2,10)  ],  kernel= "radial"  , cost = 1, gamma = 0.001)

> svm_model

> plot(svm_model_rbf, BC_dataframe[,  c(1,2,10)  ],symbolPalette = rainbow(4),color.palette = terrain.colors)
```



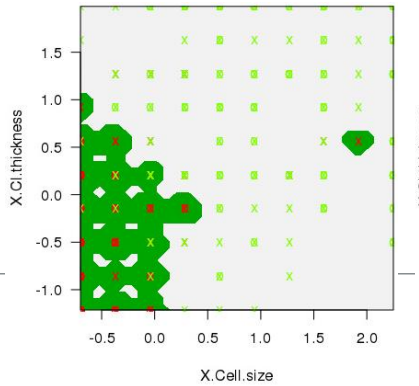
cost = 1, gamma = 0.001



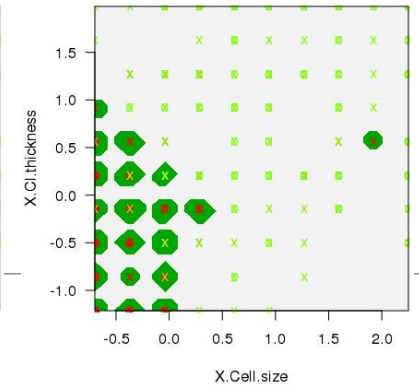
cost = 1, gamma = 0.1



cost = 1, gamma = 50



cost = 1, gamma = 100

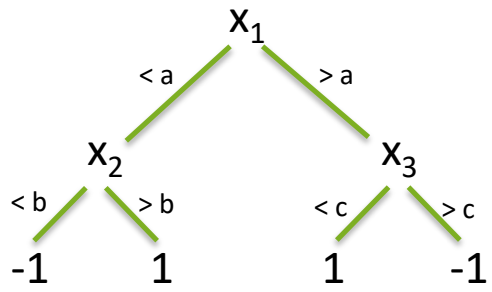


Classification trees

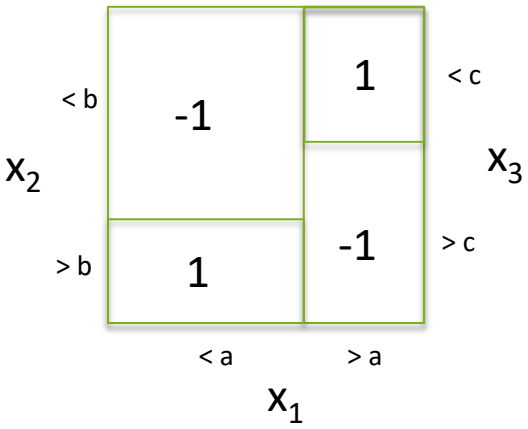
Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

- Classification process based on rules with tree structure



- Binary recursive partitioning



Metrics

- Gini impurity

$$I_G(f) = \sum_{i=1}^{J} f_i(1 - f_i)$$

- Entropy

$$H(T) = -\sum_{i=1}^{J} \mathrm{p_i} \log_2 \mathrm{p_i}$$

Classification trees

Methods aspects

- Determines best possible split at a particular stage -> local optimum, not globally best tree
- Prone to overfitting
  - adjustment of tree complexity (depth)
  - pruning

Pruning

Remove parts of the tree that have small impact on accuracy

Top-down (remove branches at their root) or bottom up (start with leafs)

A simple example

Reduced error pruning:

start with leaf and replace by majority class; keep change if performance not affected

Classification trees

Zentralinstitut für
Seelische Gesundheit
Landesstiftung
des öffentlichen Rechts

Three versions commonly used in R

tree *library(tree)*

- Simple recursive partitioning using node impurity

Classification and regression trees (CART) *library(rpart)*

- Simple recursive partitioning using node impurity
- Faster compared to tree

Random forests *library(randomForest)*

- Ensemble method

Classification trees

Advantages

- Can incorporate non-numeric predictors easily
- Naturally encode interactions in the model
- Random forest implementations have built-in variable importance assessment

Disadvantages

- May have complex structure, large trees difficult to interpret
- Generalization performance often worse than for simpler, linear algorithms

Concept

- Communicating nodes (artificual neurons) organized in layers
- The simple case: the perceptron

Linear decision boundary

$$f(x) = \begin{cases} 1 & if\ w \cdot x + b > 0 \\ 0 & otherwise \end{cases}$$

Learning

Initialize weights to 0

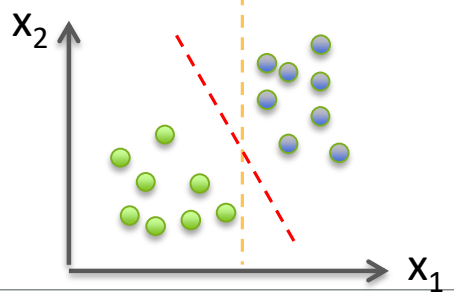
Determine predicted class

Update weights to minimize prediction error

$$w_i(t+1) = w_i(t) + \left(d_j - y_j(t)\right) x_{j,i}$$

for all features i to n

Linear SVM -> Perceptron of optimal stability



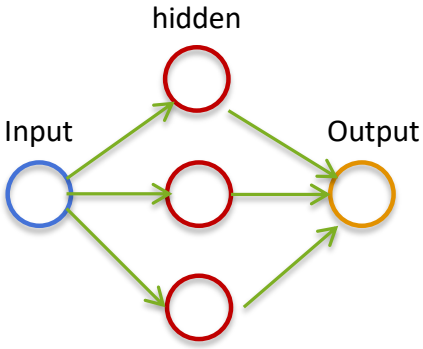
Neural networks

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

- Communicating nodes (artificial neurons) organized in multiple layers
- More complicated: hidden layers

Multilayer perceptron „Deep neural network“



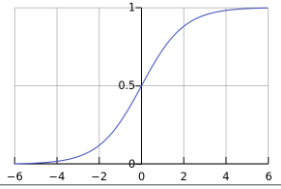
Parameters

- Conenctions between nodes
- Weight of connections
- Function that converts input to output

- Graphical representation of a set of functions

$$f(x) = K\left(\sum_i w_i g_i(x)\right)$$

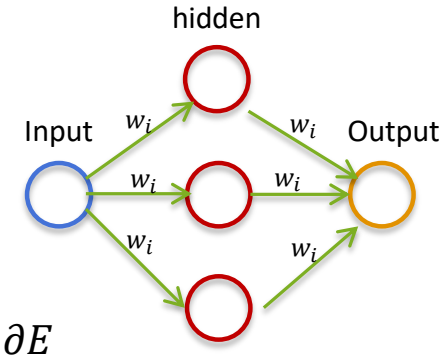
- All but input node are artificial neurons
- Activation functions are non-linear (otherwise multilayer percepton reduces to simple perceptron)
  - Sigmoids $y(v_i) = \tanh(v_i)$ -> hyperbolic tangent
    $y(v_i) = (1 + e^{-v_i})^{-1}$ -> logistic function



Learning through backpropagation

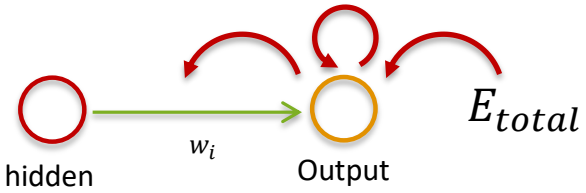
Again, weights change iteratively based on prediction error

$$E_{total} = \frac{1}{2}\sum(target - output)^2$$



Gradient $\nabla E = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots, \frac{\partial E}{\partial w_l}\right)$ optimize $\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$

$$\frac{\partial E}{\partial w_i} = \frac{\partial inp}{\partial w_i} * \frac{\partial out}{\partial inp} * \frac{\partial E}{\partial out}$$



Neural networks

Advantages

- Can have non-linear decision boundary
- Can easily incorporate prediction of multiple outcomes
- Deep networks work well for some applications (i.e. face recognition)

Disadvantages

- Black-box model, difficult to interpret
- Can be prone to overfitting
- Optimal training not straightforward
- Optimization can identify local minimum
- Requires relatively large sample sizes to learn parameters accurately

Concept

- Simple probabilistic classifier based on Bayes' theorem
- Makes strong independence assumptions about predictors

$$p(C_k|x_1, x_2, \dots, x_n)$$

$$p(C_k|x) = \frac{p(C_k)\, p(x|C_k)}{p(x)}$$

$$p(x_1, \dots, x_n, C_k) = p(x_1|x_2, \dots, x_n, C_k)\, p(x_2|x_3, \dots, x_n, C_k) \cdots p(x_n|C_k)\, p(C_k)$$

- Naive conditional independence assumption

$$p(x_1|x_2, \dots, x_n, C_k) = p(x_1|C_k)$$

$$p(C_k|x) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

$$\hat{y} = \underset{k \in \{1, \dots, K\}}{argmax}\ p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

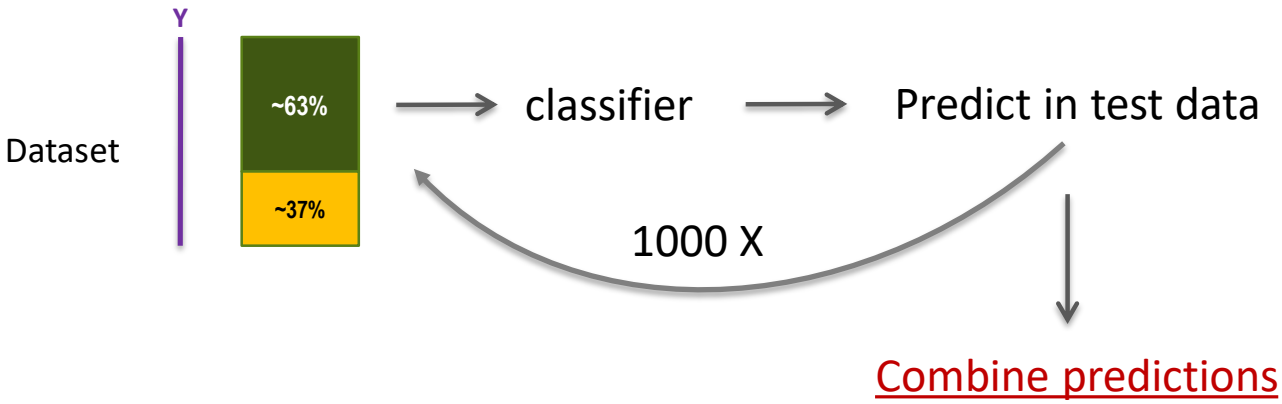
maximum a posteriori decision rule

Model aggregation - Bagging

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

- Bootstrap aggregation
- Build multiple models on bootstrap samples of original data and
  - Average output (regression)
  - Majority vote (classification)



Model aggregation - Boosting

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

- Sequential learning of individually weak classifiers
- Data used at each step depends on performance of previous classifier
- **Final output linear combination of weak classifiers**
- **Weak classifiers derived in *m* iterations**

$$C_{(m-1)}(x_i) = \alpha_1 k_1(x_i) + \cdots + \alpha_{m-1} k_{m-1}(x_i) \quad \text{expand to} \quad C_{(m)}(x_i) = C_{(m-1)}(x_i) + \alpha_m k_m(x_i)$$

- **Choose classifier that minimizes total weighted error**

$$\sum_{y_i \neq k_m(x_i)} w_i^{(m)} \quad \text{where} \quad w_i^{(m)} = e^{-y_i C_{m-1}(x_i)}$$

- **Calculate** $\epsilon_m = \sum_{y_i \neq k_m(x_i)} w_i^{(m)} \Big/ \sum_{i=1}^{N} w_i^{(m)}$

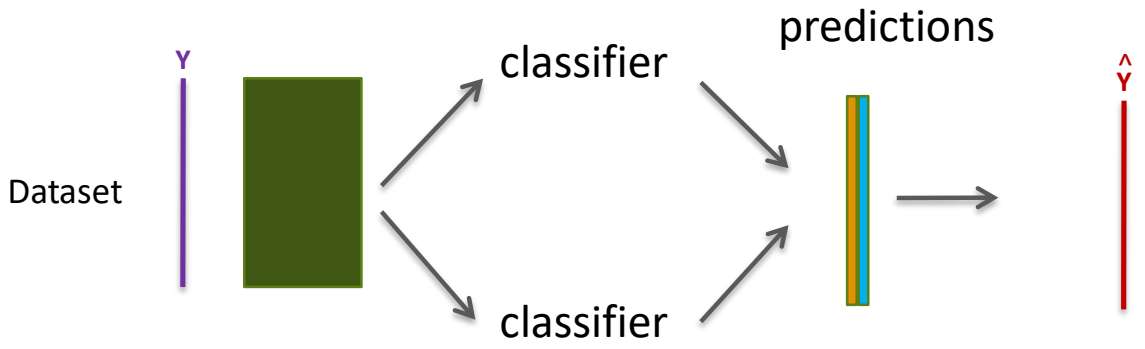
- **Determine weight** $\alpha_m = \frac{1}{2} \ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$

Adaboost (adaptive boosting)

Model aggregation - Averaging

Concept

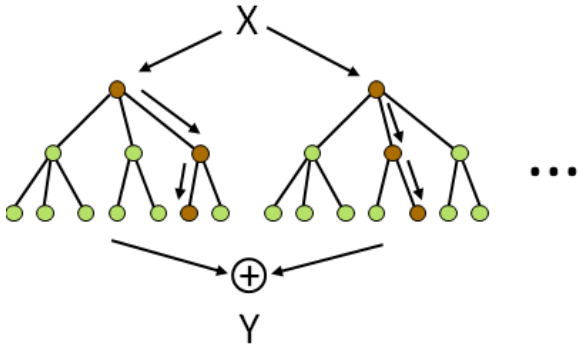
- Train multiple algorithms on the same dataset
- Committee / Ensemble methods
- Derive class from averaging predictions / majority vote



Model aggregation – ensemble methods

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

- **Random Forest**
  - Learn many trees
  - Each based on bootstrap sample of subjects
  - At each split consider random sample of variables
  - Variable importance by Gini or permutation



Random Forest example

```
> library(randomForest)

> RF_model = randomForest ( Y ~ ., data = BC_dataframe , ntree = 1000, importance = TRUE)
```

```
> RF_model

Call:
 randomForest(formula = Y ~ ., data = BC_dataframe, ntree = 1000,      importance = TRUE)
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 3

        OOB estimate of  error rate: 2.49%
Confusion matrix:
          benign malignant class.error
benign       433        11  0.02477477
malignant      6       233  0.02510460
```

Out-of-bag error rate

Model aggregation – ensemble methods

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

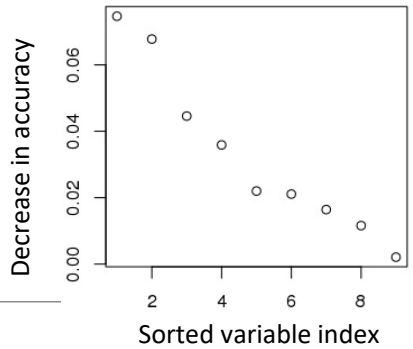
Random Forest example - identify predictor importance

```
> RF_model$importance[order(RF_model$importance[,3],decreasing=TRUE),]
```

	benign	malignant	MeanDecreaseAccuracy	MeanDecreaseGini
X.Cell.size	0.071024554	0.0821424723	0.074633061	85.990168
X.Bare.nuclei	0.071375721	0.0613776745	0.067757264	51.483354
X.Cell.shape	0.017647380	0.0949512254	0.044564282	70.250943
X.Cl.thickness	0.036897932	0.0340185250	0.035877434	15.220007
X.Bl.cromatin	0.010310288	0.0436611125	0.021993896	30.489107
X.Normal.nucleoli	0.025230367	0.0132581847	0.021085275	21.902367
X.Marg.adhesion	0.010979611	0.0267407180	0.016404022	7.980254
X.Epith.c.size	0.011756176	0.0112901582	0.011579938	24.847444
X.Mitoses	0.002902759	0.0005631446	0.002085165	2.090589

Decrease in accuracy when variable is permuted (mean over all classes)

Decrease in Gini index (mean over all classes)

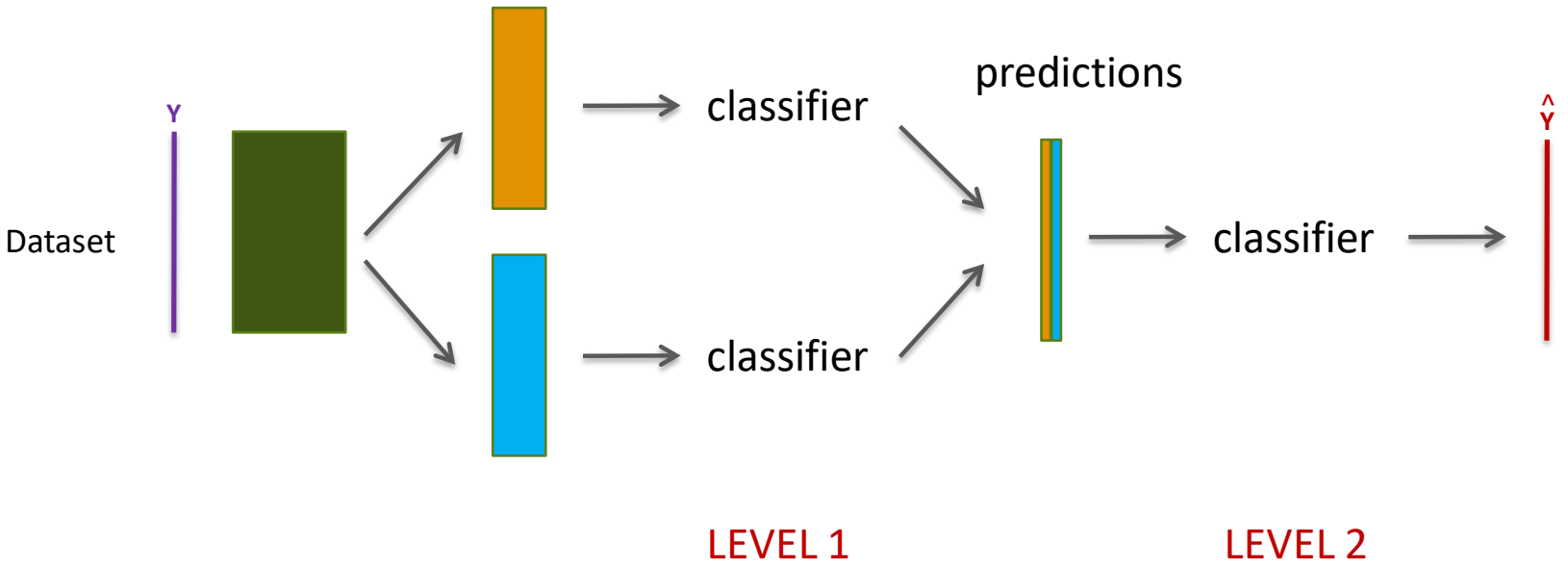


Model aggregation - Stacking

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Concept

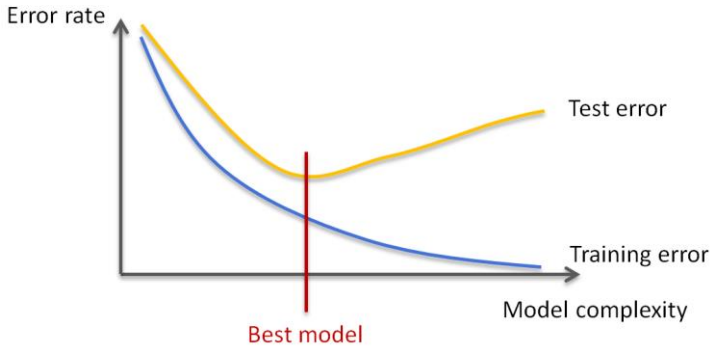
- Split variables into multiple sets
- Learn separate classifier on each set (level 1) and predict
- Use predictions to learn another classifier (level 2) and make final prediction



Variable selection

Concept

- Selection of variable subset with better generalization
- Also known as **feature selection**
- Improved interpretability of models



- Procedure to (1) propose variable subset (2) compare performance
- Exhaustive search computationally infeasible for even moderate numbers of features

- Main approaches
  - Wrappers
  - Filters
  - Embedded methods

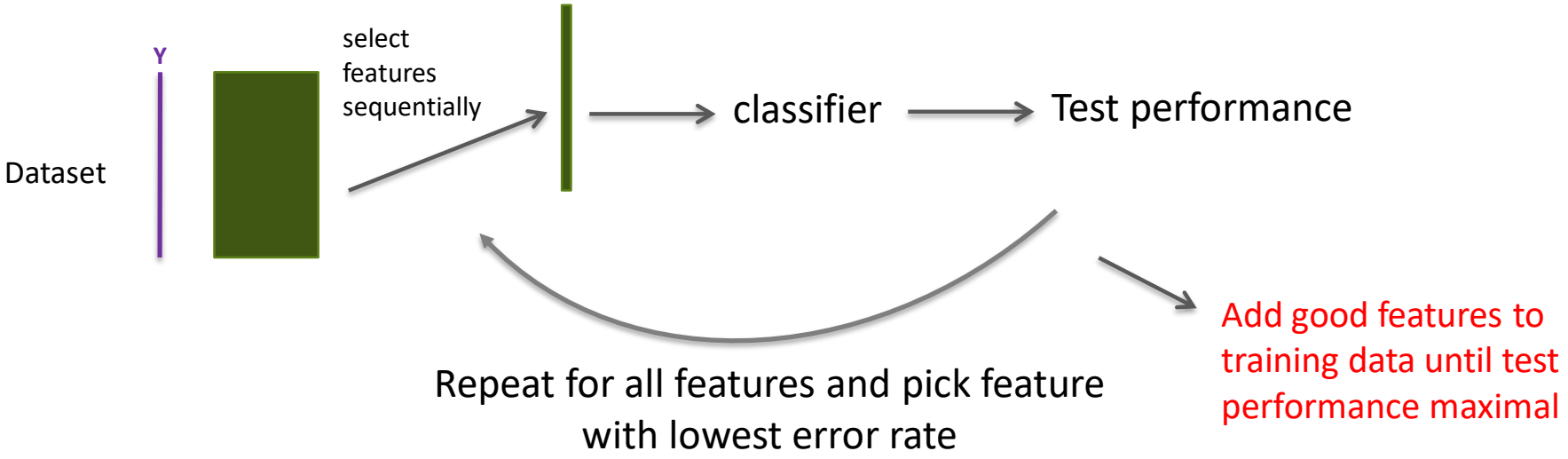
Variable selection - wrappers

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Wrappers

- Intuitive procedure:
  - Build separate model on each feature subset and predict in test data
  - Computationally most expensive

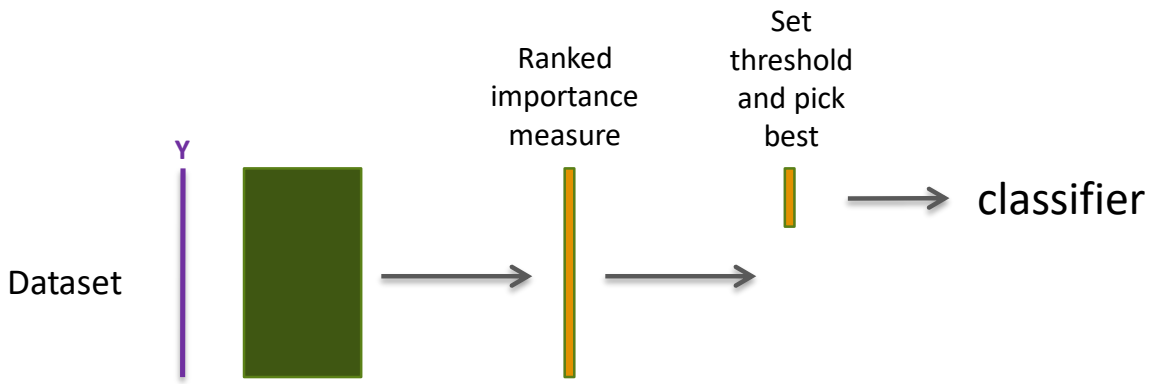
- I.e. forward selection



Variable selection - filters

Filters

- Use proxy for error rate to evaluate feature subset
  - Scoring of features based on mutual information, correlation etc.
  - Many methods produce ranked lists of features
  - Best cut-off to be chosen by cross-validation



Variable selection – embedded methods

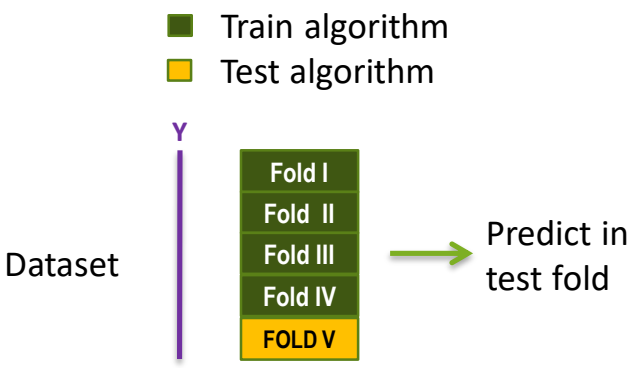
Embedded methods

- Selection of features during model building process
  - i.e. LASSO
- Recursive feature elimination -> remove features with low weight

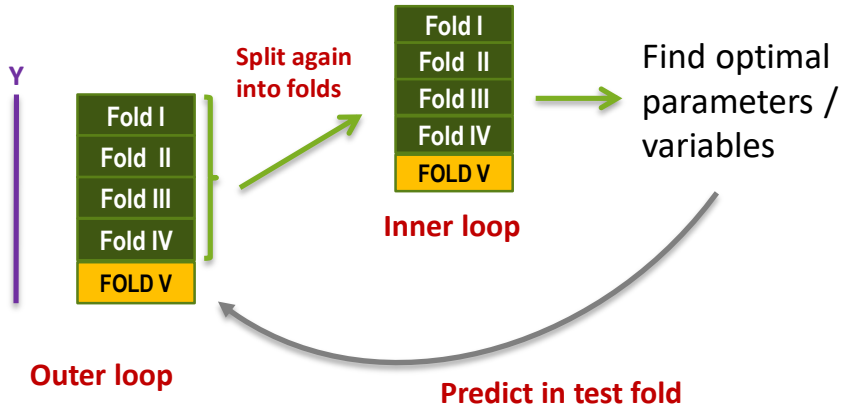
Variable selection – nested cross-validation

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Cross-validation



Nested cross-validation



Variable selection – nested cross-validation

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Example – pick optimal k for knn and test performance

```
> library(caret)
> library(class)
> Cvfolds  = createFolds(   c(1: nrow(BC_dataframe) )  , k = 10 )

> result = numeric()
> for( i in 1:length(Cvfolds)) {
        INNERfolds  = createFolds(   c(1: length(unlist (Cvfolds[-i] ))  )  , k = 5)
        kresult = numeric()
        for(ktry in 1 : 10){
                inner_result = numeric()
                for( ii in 1:length(INNERfolds) ){
                        trainindexINNER  =  unlist (Cvfolds[ -i ] )[unlist (INNERfolds [-ii] )]
                        testindexINNER   =   unlist (Cvfolds[ -i ] ) [unlist (INNERfolds [ii] )]

                        knn_model = knn(BC_dataframe[trainindexINNER , -10], BC_dataframe[testindexINNER , -10],
                                                cl = BC_dataframe[trainindexINNER,10], k = ktry )

                        error = sum( knn_model != BC_dataframe[testindexINNER,10] )  /  length(knn_model)
                        inner_result  = c(inner_result , error)
                }
                kresult = rbind(kresult , c(ktry, mean(inner_result ) ))
        }
        bestk = kresult[ which(kresult[,2] == min(kresult[,2]))[1], 1]
        trainindexOUTER  =  unlist (Cvfolds[ -i ] )
        testindexOUTER    = unlist (Cvfolds[ i ] )
        knn_model = knn(BC_dataframe[trainindexOUTER , -10], BC_dataframe[testindexOUTER , -10],  cl = BC_dataframe[trainindexOUTER ,10], k = bestk )
        error = sum( knn_model != BC_dataframe[testindexOUTER  ,10] )  /  length(knn_model)
        result = c(result, error)
}
> mean ( result )
[1] 0.03220375
```

Create outer folds

Outer loop

Create inner folds

run through k from 1 to 10

Inner loop

Save error rates of inner folds

Mean error rate across inner folds for every ktry

Find k with mimium error rate

Predict model with optimal k in outer test fold

Variable selection – nested cross-validation

Overview

Day 1

- Introduction
- Linear methods
  - Regression
  - Classification
- Model assessment and selection

Day 2

- Non-linear and probabilistic methods
- Model aggregation
- Variable selection

Day 3

- Predicting multiple classes or multiple outcomes
- Semi-supervised machine learning

END OF DAY 2

Overview

Day 1

- Introduction
- Linear methods
  - Regression
  - Classification
- Model assessment and selection

Day 2

- Non-linear and probabilistic methods
- Model aggregation
- Variable selection

Day 3

- Predicting multiple classes or multiple outcomes
- Semi-supervised machine learning

Multi-class machine learning

Strategies

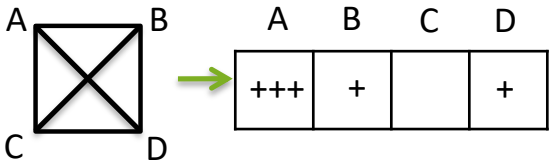
Multiple binary classification

One vs rest

- One classifier per class
- classify each subject using all classifiers -> determine classifier score
- Decide for class with highest score among all classifiers

One vs one

- Train classifiers for all pairwise combinations: (k(k-1)) / 2
- Vote among classifiers
- Decide for majority class



Multi-class machine learning

Strategies

Multiple binary classification

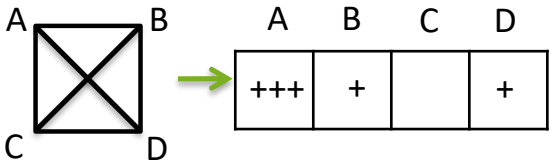
One vs rest

- One classifier per class
- classify each subject using all classifiers -> determine classifier score
- Decide for class with highest score among all classifiers

PROBLEMS
Confidence bounds on scores
Unbalance distributions

One vs one

- Train classifiers for all pairwise combinations: (k(k-1)) / 2
- Vote among classifiers
- Decide for majority class



PROBLEMS
Classes can receive same number of votes

Multi-class machine learning

LOAD DATASET

```
data(BreastCancer)

BCnum  = apply(  BreastCancer[,2:10] ,  2 ,   function(x){  scale( as.numeric(x) )  })

BCnum = data.frame(BCnum)

BCnum_noNA = na.omit( BCnum )

Y = BreastCancer$Class[as.numeric( rownames(BCnum_noNA) )]

BC_dataframe = data.frame(X = BCnum_noNA, Y = Y)
```

Multi-class machine learning

Example based on random forests

Add artificial third group to data

```
> BC_add = sapply( c(1:100) , function(x) { colMeans( BC_dataframe[sample(c(1:nrow(BC_dataframe)),2),-10]) })

> BC_add = data.frame( t ( BC_add ) )
> colnames(BC_add) = colnames(BC_dataframe)[1:9]
> BC_add$Y = rep ("intermediate", nrow(BC_add))
> BC_mult = rbind(BC_dataframe, BC_add )

> library(randomForest)

> RF_model = randomForest ( Y ~ ., data = BC_mult , ntree = 1000, importance = TRUE)
```

```
Call:
 randomForest(formula = Y ~ ., data = BC_mult, ntree = 1000, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 3

        OOB estimate of  error rate: 8.17%
Confusion matrix:
             benign malignant intermediate class.error
benign          434        10            0  0.02252252
malignant         7       231            1  0.03347280
intermediate     30        16           54  0.46000000
```

Canonical correlation analysis

Concept

- Identify dependency between two sets of variables
- **Find solution where linear combinations in each set have strongest association across sets**

Find vectors **a** and **b** where **a'X** and **b'Y** maximize correlation **ρ(a'X, b'Y)**

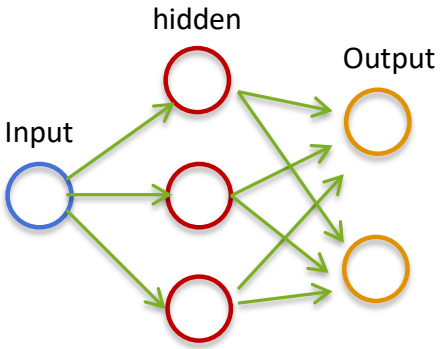
First pair of canonical covariates

- **Find further pairs that are uncorrelated with other pairs**

- Implemented in R *library(cancor)*

The multilayer perceptron

Multilayer perceptron
„Deep neural network“



Fully connected NN with 1 hidden layer

```
> library(nnet)

> NN_model   =   nnet( Y ~.  , data = BC_mult,  size = 6,   rang = 0.1, decay = 1e-2, maxit = 2000)
```

Multiclass performance measures

Average per-class accuracy

Multi-class AUC

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management, 45*, p. 427-437.

Exercise

Use cross-validation to determine optimal number of neurons in hidden layer of NN

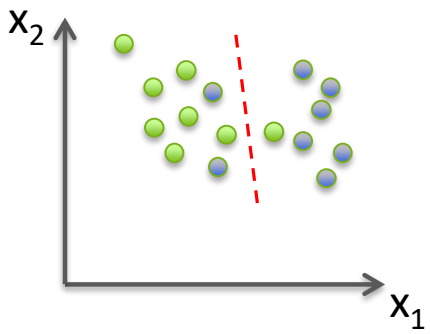
```
> library(nnet)
> require(caret)
> Cvfolds  = createFolds(   c(1: nrow(BC_mult)  )  , k = 5)

> result = numeric()
> for( i in 1:length(Cvfolds)) {
        trainindex  =  unlist (Cvfolds[-i] )
        testindex   = unlist (Cvfolds[i] )

        nnumber = numeric()
        for(ntest in 4:6){
                NN_model    =   nnet( Y ~.  , data = BC_mult[trainindex , ],  size = ntest ,   rang = 0.1, decay = 1e-2, maxit = 2000)
                prediction    =  predict (NN_model , BC_mult[testindex , ], type = "class")
                cont.mat = table( BC_mult$Y[testindex]  ,  prediction )

                cont.tab = table( factor( prediction, levels =c("intermediate", "benign"  ,    "malignant") ),
                        factor( BC_mult$Y[testindex], levels =c("intermediate", "benign"  ,    "malignant") ) )

                accuracy = diag (cont.tab/ colSums(cont.tab))
                nnumber  = c(nnumber , mean(accuracy))
        }

        result = rbind(result , nnumber)
}

cvResult = colMeans( result)
c(4:6) [ which(cvResult  == max(cvResult ) ) ]
```

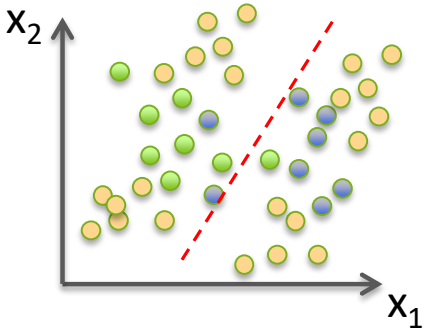
Semi-supervised machine learning

Concept

- Use partially unlabeled data for training of supervised classifier

- Transductive learning → estimate lables of unlabeled data
- Inductive learning → find optimal mapping from X to Y





Assumptions (at least one of the following)

- Smoothness
  - Points that are close likely have the same label
- Clustering
  - Data organized in clusters with observations having same label
- Manifold
  - Data can be represented in lower dimensional space

Interesting example: transductive SVM

Real-world challenges

Typical issues of high-dimensional data analyis

- Unequal group sizes
- Missing values
- Confounding
- Feature correlation
- Very high dimensionality of data
- Generalizability with p >> n
- Study design and theoretical bounds of perfomance

- Performance measures beyond accuracy ?

Unequal group sizes

Challenge:

- One group is strongly over-represented
- Leads to preferred identification of the larger group
    - Results in sensitivity or specificity issues

Potential remedies:

- Matching
- Undersampling
- Oversampling

Unequal group sizes

Undersampling using LDA

PROBLEMS
CV folds can be unbalanced
-> stratified CV

```
➢ require(caret)
Cvfolds_1  = lapply(createFolds(   which(BC_dataframe$Y == "malignant") , k = 10 ),
                    function(x){which(BC_dataframe$Y == "malignant")[x]   })
Cvfolds_2  = lapply(createFolds(   which(BC_dataframe$Y == "benign" ) , k = 10 ),
                    function(x){which(BC_dataframe$Y == " benign")[x]   })
> Cvfolds     =  mapply(c, Cvfolds_1 , Cvfolds_2)

> result = numeric()
> for( i in 1:length(Cvfolds)) {
          trainindex  =  unlist (Cvfolds[-i] )
          testindex   = unlist (Cvfolds[i] )

          acc_prediction = numeric()
          for(ii in 1:10) {
                    minclasssize = min( table(BC_dataframe$Y [trainindex] ) )
                    s1 = sample( which(BC_dataframe$Y [trainindex] == "malignant"),   size = minclasssize, replace=TRUE)
                    s2 = sample( which(BC_dataframe$Y [trainindex] == "benign"),   size = minclasssize, replace=TRUE)
                    undersampling = c(s1, s2)

                    lda_model_CV   =  lda(  Y ~ ., data = BC_dataframe[trainindex , ][undersampling, ])
                    prediction  =  predict (lda_model_CV, BC_dataframe[testindex , ])$class

                    sensitivity  = sum(BC_dataframe$Y[testindex] == "malignant" & prediction == "malignant" )/ sum(BC_dataframe$Y[testindex] == "malignant")
                    specificity  = sum(BC_dataframe$Y[testindex] == "benign" & prediction == "benign" )/ sum(BC_dataframe$Y[testindex] == "benign")

                     acc_prediction = c(acc_prediction , (sensitivity  + specificity) / 2  )
          }

           result  = c(  result ,  mean (acc_prediction)  )
  }
  > mean ( result )
   [1] 0.9493505
```

Missing values

Challenge:

- Predictors contain missing values
- Most methods implemented in R do not accept data with missing values

Potential remedies:

- Remove subjects with missing values
- Remove predictors with missing values
- Replace missing values

Potential problems

- Values not missing at random
  - Subject removal or value replacement can lead to bias

Confounders

Challenge:

- A confounder is associated with the outcome as well as the predictor
- Example: over-representation of males among patients and gender-associated predictors

Potential remedies:

- Remove confounded predictors
- Remove association between confounder and outcome through matching
- Remove association between confounder and predictor (i.e. through residualization)

Potential problems

- Matching often not possible
- Machine learning sensitive for combined effect of multiple, weakly confounded predictors
- Association between confounder and predictor may not be linear -> residual bias
- Artefacts after removal of confounder effects
- Confounder itself may have important role for illness

Feature correlation

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Challenge:

- Multiple features show strong correlation
- Example: Single Nucleotide Polymorphisms in linkage dysequillibrium, co-expressed genes, …
- Problem for variable selection
- Can be problematic for estimation of accurate feature coefficients

Potential remedies:

- Reduction of dimensionality through aggregation of correlated predictors
- Use methods that are less sensitive to feature correlation

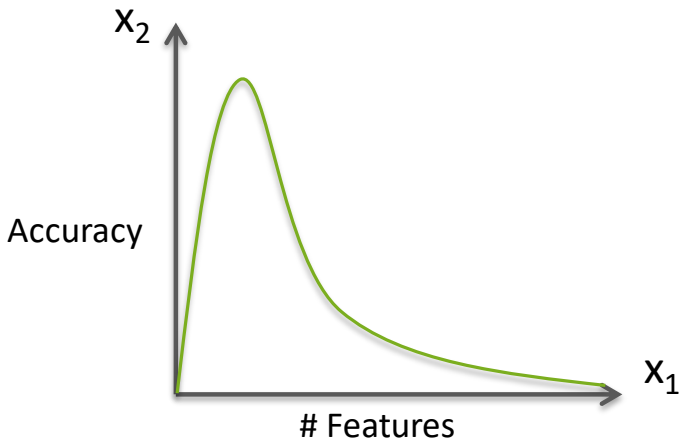
Potential problems

- Aggregation of correlated predictors often with arbitrary elements (method, cutoffs etc.)
- Potential loss of data

Data dimensionality / generalizability with n << p

Zentralinstitut für Seelische Gesundheit
Landesstiftung des öffentlichen Rechts

Challenge:



PROBLEMS
How to select features with >1,000,000 variables ?

Potential remedies:

- Use of additional information for feature stratification
  - biological pathway data
  - Additional data modalities
- Hierarchical machine learning methods
- Look for biological reproducibility !

Study design / Theoretical bounds on generalizability

ELSEVIER

Discrete Applied Mathematics

Volume 42, Issue 1, 26 February 1993, Pages 65-73

Bounding sample size with the Vapnik-Chervonenkis dimension

John Shawe-Taylor

- Theoretical bounds often too conservative
  - unrealistically large sample numbers
  - Bound on generalizability too low

- Sample size estimation from pilot data
  - Strongly depends on data modality, likely effect sizes that are difficult to predict

Performance measures beyond accuracy

- Selection stability under sampling variability
- Longitudinal measurement stability
- Cost-effectiveness (constraint-programming, learning with privileged information)
- Cross-platform transferability

Goals

Random Forest

How it works

Random forest is an ensemble learning method that …

How to use it in R

randomForest(Y~., data=X, ntree = 500, importance=TRUE)

Output from R

Screenshot 1

Screenshot 2

Description and interpretation

The variable importance ranks features by the GINI index …

In this data, the most important predictors are …

Advantages and disadvantages of random forest

- Random forests can incorporate categorical predictors
- Random forests often inferior to simpler, linear methods
- …

Expression dataset – 500 subjects, 17000 genes, two classes

1. Data completeness (meta)
2. Analyze meta data
   Group imbalances -> pontential confounding
3. Matching
4. Quality control
   1. Missing values
   2. Filtering
   3. Normalization
5. Data structure, PCA
   1. Outliers
   2. Residual confounding
6. Confounder adjustment
7. Univariate analysis
   1. Adjust for multiple hypothesis testing
8. Machine learning
   1. Variable selection (nested cross-validation)
   2. Testing in independent datasets
   3. Test disease specificity
9. Biolog. interpretation