PREDICTIVE SCORE, CROSS VALIDATION AND PENALIZED REGRESSION USING R

Damien Drubay damien.drubay@gustaveroussy.fr

SBE Gustave Roussy / Oncostat team - INSERM U1018

Course and codes available here: https://github.com/Oncostat

PLAN

1. Introduction
2. Model calibration
3. Cross-validation
4. High dimensional data analysis issues
5. Penalized regression
6. Alternatives methods
7. Conclusions

INTRODUCTION

PREDICTIVE SCORE

Resume the current characteristics of patient in one measure adapted to predict future of:

- biomarker value
- disease state
- patient survival probability
- ...

-> Simplify the decision for patient care

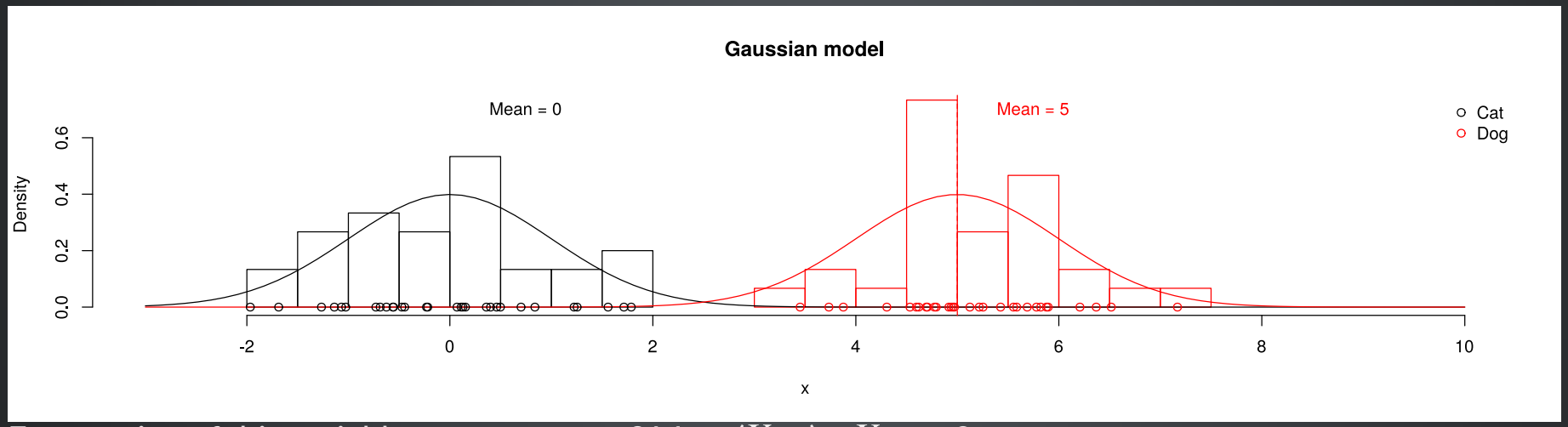
Use of statistical/probabilist model

INTRODUCTION

WHAT IS A STATISTICAL MODEL?

Model definition: *A simplified description, especially a mathematical one, of a system or process, to assist calculations and predictions.* (Oxford dictionary)

Example:



Expectation of this variable among new cat? Mean(X_{cat}) = $\bar{X}_{cat}$ = 0
Expectation of this variable among new male? Mean(X_{dog}) = $\bar{X}_{dog}$ = 5

Mean = expectation in specific group according to the gaussian model = prediction

INTRODUCTION

PREDICTIVE SCORE

Set up a predictive score

Classical procedure:

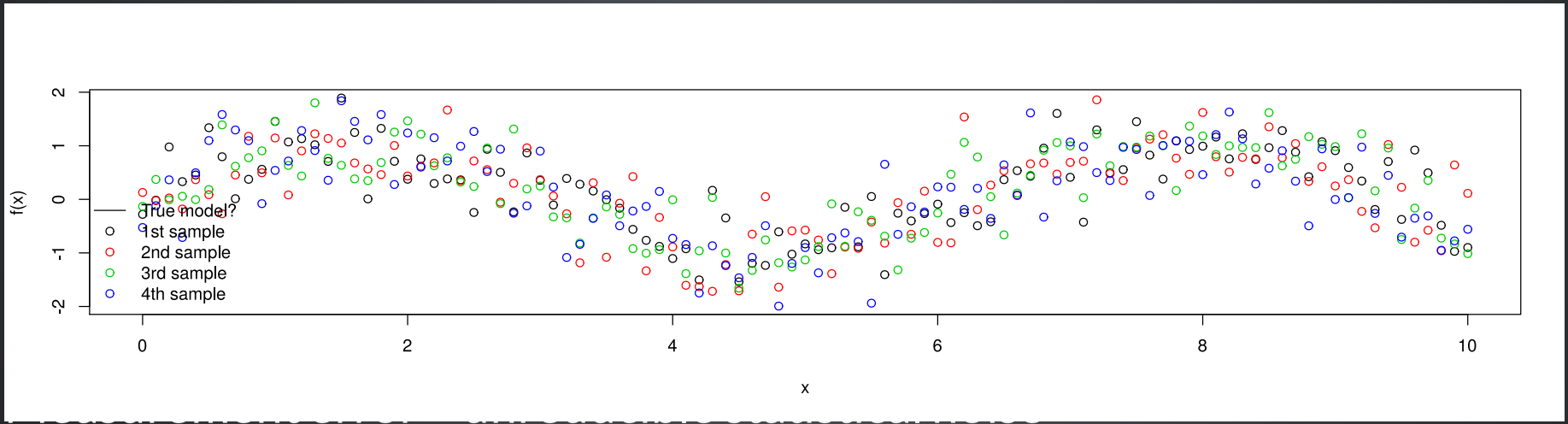
1. Set up statistical model with available data (= training)
2. Assign to new patient/observation the expectation of the model (= predictive score) according to his characteristics (= validation)

Main assumption: the same process will generate the future data

INTRODUCTION

WHAT IS A STATISTICAL MODEL?

Suppose that the data was generated according from a sine wave model:



But we don't know the true model!

Statistical modeling: try to find the underlying model

PLAN

1. Introduction
2. **Model calibration**
3. Cross-validation
4. High dimensional data analysis issues
5. Penalized regression
6. Alternatives methods
7. Conclusions

MODEL CALIBRATION

STATISTICAL MODELING STRATEGY FOR PREDICTION

Set up mathematical model:

$$Y = f(\mathbf{X}\beta)$$

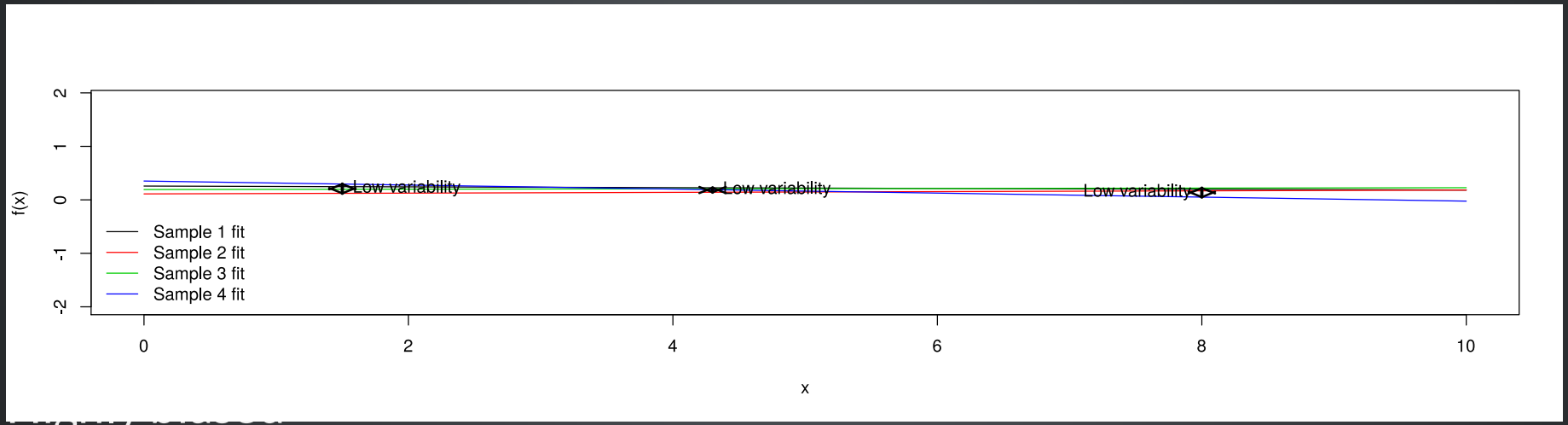
Y: Response
$\mathbf{X}$: individual characteristics
β: Effect of $\mathbf{X}$ on Y
f: Link function between $\mathbf{X}$ and Y

Example of linear model: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p + \epsilon$
With ϵ the error term

MODEL CALIBRATION

SIMPLE MODEL

Linear model prediction: $\bar{Y} = \beta_0 + \beta_1 X$



Highly biased

But it has low variance! (similar model whatever the dataset = robust model)

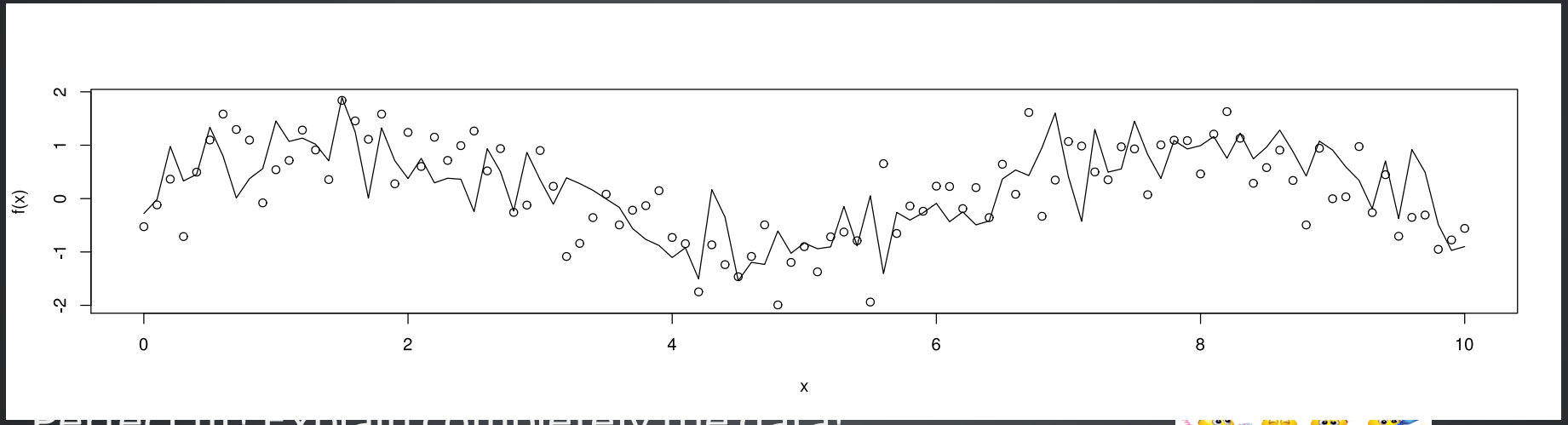
Stable for new dataset, but high prediction error

MODEL CALIBRATION

COMPLEX MODEL

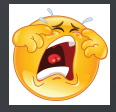
Complex model: linear extrapollation:

$$\bar{Y} = \beta_0 + \beta_1(X_2 - X_1) + \ldots + \beta_{n-1}(X_n - X_{n-1})$$



Perfect fit! Explain completely the data!

High variance! Model the measurement error (noise) which is not reproducible! High prediction error for new data!



MODEL CALIBRATION

HOW TO FIND THE OPTIMAL PREDICTIVE MODEL?

Prediction issues:

- Simple model has high bias but low variance
- Complex model has low bias but high variance

= Bias/variance trade-off

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y}_i)^2 = bias^2 + variance + \sigma^2$$

With n the sample size, y_i and $\bar{y}_i$ the observation and the prediction for the observation i, and σ^2 the irreducible random noise (e.g. measurement error)

Biased model is not a bad thing!
(if increase the bias highly decrease the variance)

MODEL CALIBRATION

HOW TO FIND THE OPTIMAL PREDICTIVE MODEL?

Interpretation issues:

Simple model is not enough informative

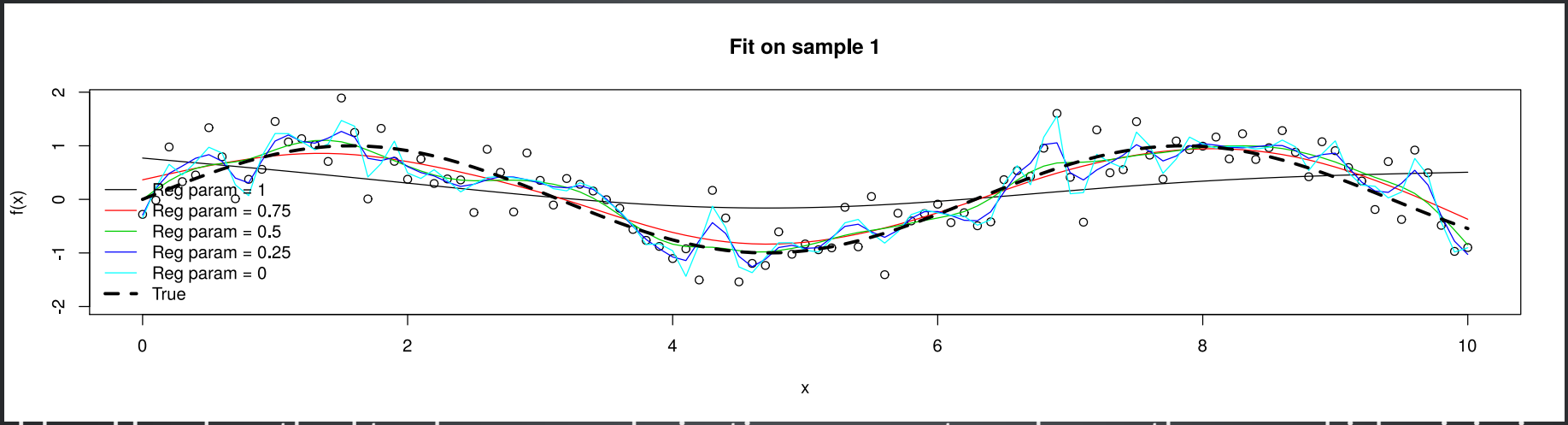
Perfect fit is not desirable: Parameters explain the underlying process, but also the random noise = coefficients interpretation issues!
Avoid over-paramterization

Objective: find a compromise

MODEL CALIBRATION

HOW TO FIND THE OPTIMAL PREDICTIVE MODEL?

Compromise model (ex: using penalized spline):

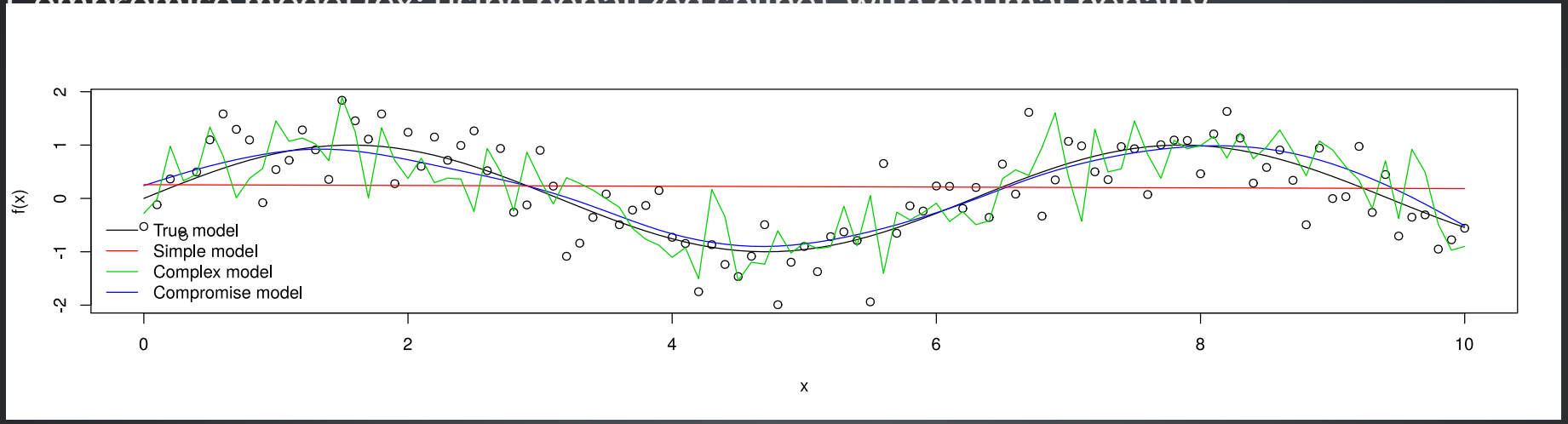


Idea: Usual method to choose regularization parameter: choose the one which minimize the prediction error
(more details in few slides)

MODEL CALIBRATION

HOW TO FIND THE OPTIMAL MODEL?

Compromise model (ex: using penalized spline), with optimal penalty



True model	Simple model	Complex model	Compromise model
20.953	62.363	0.000	19.120
24.945	70.506	47.010	27.123
21.544	70.053	35.142	21.725
26.694	78.368	49.846	29.932

MODEL CALIBRATION

HOW TO FIND THE OPTIMAL MODEL?

How can we measure prediction error on new dataset if we have only one dataset?

Idea: split your dataset in 2 subsamples:

- Fit the model on the first subsample (train your model)
- Predict the responses of the second subsample and measure the prediction error (test your model)

= Basis of the cross-validation!

PLAN

1. Introduction
2. Model calibration
3. Cross-validation
4. High dimensional data analysis issues
5. Penalized regression
6. Alternatives methods
7. Conclusions

CROSS-VALIDATION

Model comparison:

1. Split your data in k subsamples (usually 10)
2. Left one subsample out and fit (train) the models on the other subsamples
3. Predict the scores (one by model) on the left-out subsample (test the model on a data which has not been used to set-up the model = mimic the fact we have new data)
4. Evaluate the prediction accuracy of each model using utility measure (mean square error (MSE), area under the ROC curve (AUC_{ROC}),...)
5. Repeat 1. until all subsamples have been used as test sample
6. Provide the mean of the utility measure obtained at each iteration
7. Conclude: the best model is the one with the best utility measure (e.g. minimal MSE, maximal AUC,...)

CROSS-VALIDATION

TOY EXAMPLE

Create (simulate) a small dataset with n=30 and p=10 according to the following linear model:

$$Y \sim N(\mu, \sigma^2)$$

With

$$\mu = \beta_0 + \beta_1 X_1 + \ldots + \beta_{10} X_{10}$$

$$\sigma^2 = 0.5$$

Code:

```
#locate the console in the correct directory
setwd("home/sequencing/work/Penalized regression")
#random seed initialization (ensure reproducibility)
set.seed(123)
n<-30 #number of observations
#Effect of each covariate: 2 with large effect, 2 with moderate effect, 6 with no effect
betas<-c(10,2,2,0.5,0.5,0,0,0,0,0,0) #first value (10) is the intercept (beta0)
#Generate the 10 independent random gaussian variables
x<-sapply(1:10,function(x)rnorm(n))
#Generate response variable according to the beta*x + noise
y<-rnorm(n,cbind(rep(1,n),x)%*%betas,0.5)
#check, fitting linear model
toyData<-data.frame(y=y,x=x)
colnames(toyData)<-c("y",paste0("x",1:10))
```

RANDOM NUMBER GENERATION

```
#generate random numbers
rnorm(1)
rnorm(1)
```

Two different number are generated
But if we should have reproducible results? (other people would check your results)

```
#random seed initialization (ensure reproducibility)
set.seed(123)
rnorm(1)
rnorm(1)
set.seed(123)
rnorm(1)
rnorm(1)
```

Always two different numbers generated after the seed initialization
The same series of random numbers are generated after the same seed initialization
Another user using the same seed (and code) than you will find the same results, even for the generation of thousands random numbers

CROSS-VALIDATION

TRY WITH YOUR TOY EXAMPLE

Compare the following models using cross validation:

1. null model (only the intercept) -> the score will be the mean of y:

```
summary(lm(y~1,data=toyData)) #intercept=10.26229
mean(toyData$y) #10.26229
```

2. including X1 to X4 (all variables with effect on Y)
3. including X5 to X10 (all variables without effect on Y)
4. including all the variables

CROSS-VALIDATION

TRY WITH YOUR TOY EXAMPLE

```
#split your data in 10 subsamples
##Define n (the number of observation) subsample number indicators
subSamples<-rep(1:10,round(nrow(toyData))/10)
##shuffle these indicators
set.seed(123)
subSamples<-sample(subSamples,replace=FALSE)
#initialize
stockResults<-NULL
#for each subsample k
for(k in 1:10){
    #Fit on all data excepting subSample k
    fit0<-lm(y~1,data=toyData[subSamples!=k,]) #only the intercept
    fit1<-lm(y~x1+x2+x3+x4,data=toyData[subSamples!=k,])
    fit2<-lm(y~x5+x6+x7+x8+x9+x10,data=toyData[subSamples!=k,])
    fit3<-lm(y~.,data=toyData[subSamples!=k,]) #"." = all column excepting y
    #predict the response of the subSample k
    pred0<-predict(fit0,newdata=toyData[subSamples==k,])
    pred1<-predict(fit1,newdata=toyData[subSamples==k,])
    pred2<-predict(fit2,newdata=toyData[subSamples==k,])
    pred3<-predict(fit3,newdata=toyData[subSamples==k,])
    #Get the squared error
    sqError0<-sum((toyData$y[subSamples==k]-pred0)^2)
    sqError1<-sum((toyData$y[subSamples==k]-pred1)^2)
    sqError2<-sum((toyData$y[subSamples==k]-pred2)^2)
    sqError3<-sum((toyData$y[subSamples==k]-pred3)^2)
    #save the results
    stockResults<-rbind(stockResults,c(sqError0,sqError1,sqError2,sqError3))
}
#report the mean squared error for each model
apply(stockResults,2,mean)
#20.174441  1.200620 21.437024  1.559521
```

Best model = 2. (including all variables with effect on Y)
Model including all variables without effect on Y is wore than the NULL model -> illustration of overfitting (variance due to the measurement error is attributed to these variables -> simple model (only intercept) is better than complex model with no real effect)

CROSS VALIDATION

PREDICTION ACCURACY MEASURES

Choice of the parameterization: need to define a measure of the prediction accuracy

Several proposal which differ according to the type of the response:

1. Continuous: MSE
2. Binary: ROC curves
3. Survival binary criteria (e.g. 5-years survival): Time dependent ROC curves
4. Survival: calibration plot, concordance index
5. All: deviance = minus twice the log-likelihood on the left-out data ($-2loglik(\theta|x_{new})$

PLAN

1. Introduction
2. Model calibration
3. Cross-validation
4. High dimensional data analysis issues
5. Penalized regression
6. Alternatives methods
7. Conclusions

HIGH DIMENSIONAL DATA

Omic data:

- Low sample size (N < 1000)
- High number of variables (e.g. p > 10,000 genes)

p >>> N = high dimensional data!

Several problem with standard approach for estimation and selection = Curse of dimensionality! (Richard E. Bellman, 1961)

HIGH DIMENSIONAL DATA

CLASSICAL APPROACH PITFALLS

Selection procedure:

- High number of model to compare. Ex for 10 variables:
  `choose(10,1)+choose(10,2)+choose(10,3)+choose(10,4)+ choose(10,5)+choose(10,6)+choose(10,7)+choose(10,8)+ choose(10,9)+choose(10,10) #1023 unique combinations!`
- Stepwise selection: unstable result according the start model

```
set.seed(123)
n<-150 #number of observations
betas<-c(10,2,2,0.5,0.5,rep(0,46),rnorm(50,1,5)) #Effect of each covariate
stepX<<-sapply(1:100,function(x)rnorm(n)) #Generate the 100 independent random gaussian variables
stepY<-rnorm(n,cbind(rep(1,n),stepX)%*%betas,0.5)
stepData<-data.frame(y=stepY,x=stepX)
colnames(stepData)<-c("y",paste0("x",1:100))
library(MASS)
#fit linear model
fit0<-lm(y~1,data=stepData) #NULL model
fit1<-lm(y~.,data=stepData) #Model with all covariates
fit2<-lm(y~x1+x18+x35,data=stepData) #first starting model
#both (backward and forward) stepwise selection from the model fit2
a<-names(stepAIC(fit2,direction="both",trace=F,
                 scope = list(upper = fit1, lower = fit0))$coef) #variables in the final model
#both (backward and forward) stepwise selection from the model fit3
fit3<-lm(y~x15+x27+x85,data=stepData) #second starting model
b<-names(stepAIC(fit3,direction="both",trace=F,
                 scope = list(upper = fit1, lower = fit0))$coef) #variables in the final model
#difference between the variables selected according to the starting model
setdiff(a,b) #"x18" "x43" "x92" "x9"  "x33"
setdiff(b,a) #"x6"  "x5"  "x7"  "x46" "x16"
```

HIGH DIMENSIONAL DATA

CLASSICAL APPROACH PITFALLS

Issues for inference:

Large variance of maximum likelihood parameter estimates ($\hat{\beta}$) when p >>> n due to:

- Overfitting
- Multicollinearity

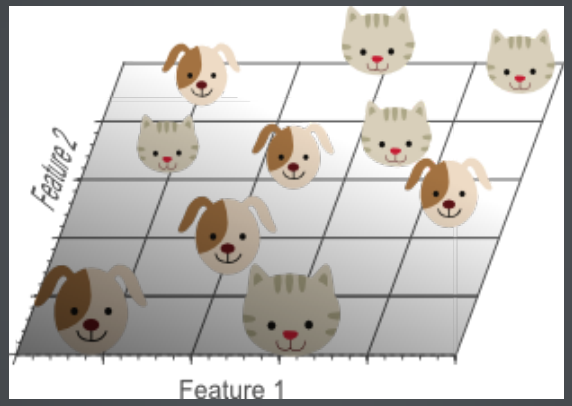
These issues also exist for smaller dimensional data, but they are less marked

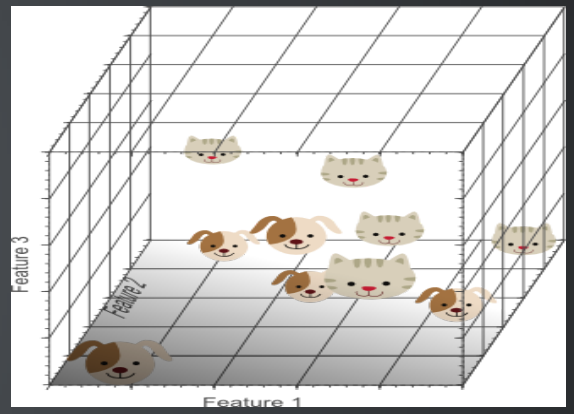
INTRODUCTION

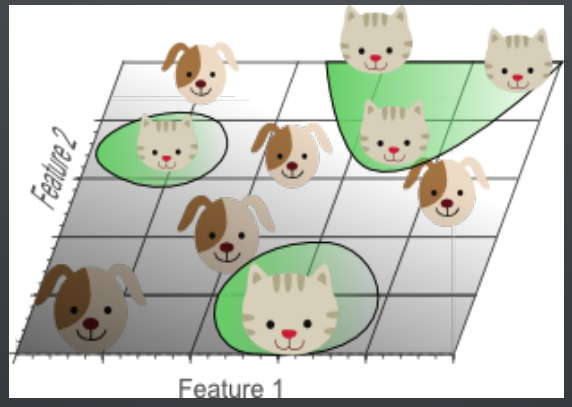
CURSE OF DIMENSIONALITY

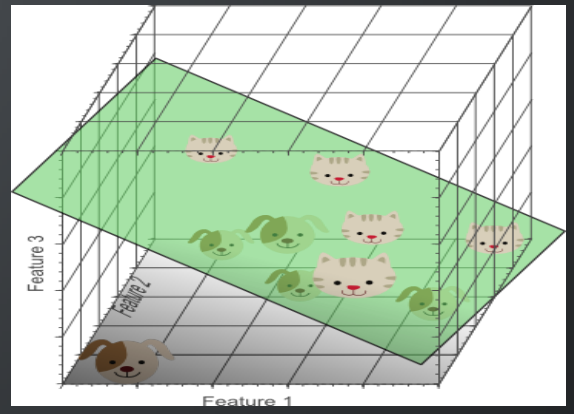
Overfitting (beyond random noise)
"Cute" example here

Use only the most discriminant variables -> selection









Considering lot of dimensions:
each observation is almost unique!
-> not suitable for prediction
(New cat can be in the right bottom corner...)

HIGH DIMENSIONAL DATA

CURSE OF DIMENSIONALITY

Multicollinearity issues:

Study of the risk of lung cancer
All smokers drink coffee in my dataset

Univariate analysis -> smoking and coffee increase the risk of cancer

But we know that smoking increases the risk of cancer -> coffee = confounding factor

But without a priori knowledge on human physiology, toxicity,... Which one can increase the risk of lung cancer?
Careful conclusion: One of them of both together (So, I don't know...)

Exactly the situation of a (frequentist or non-informative prior bayesian) model
In presence of correlated explanatory variables, the uncertainty on (variance of) parameter values is high in the multivariate analysis (careful conclusion)

HIGH DIMENSIONAL DATA

CURSE OF DIMENSIONALITY

Multicollinearity issues:

```
library(mvtnorm)
#simulate uncorrelated variables
set.seed(123)
collinX<-rmvnorm(30,c(2,15),matrix(c(2,0,0,5),2,2))
cor(collinX)
x11()
plot(collinX)
collinY<-rnorm(30,cbind(rep(1,30),collinX)%*%c(10,1.5,3),0.5)
collin<-data.frame(y=collinY,x=collinX)
summary(fit1<-lm(y~.,collin))

#simulate correlated variables
set.seed(123)
collinX<-rmvnorm(30,c(2,15),matrix(c(2,3,3,5),2,2))
cor(collinX)
x11()
plot(collinX)
collinY<-rnorm(30,cbind(rep(1,30),collinX)%*%c(10,1.5,3),0.5)
collin<-data.frame(y=collinY,x=collinX)
summary(fit2<-lm(y~.,collin))
#little bias but > x2 inflated standard error
summary(fit2)$coef[,2]/summary(fit1)$coef[,2]
#(Intercept)         x.1         x.2
#   2.257904    2.834002    3.161421
```

HIGH DIMENSIONAL DATA

CURSE OF DIMENSIONALITY

Multicollinearity issues can occur even with independent variables (due to the low sample size vs number of variables):

```
library(mvtnorm)
#simulate uncorrelated variables
set.seed(123)
#10 variables
collinX<-rmvnorm(100,rnorm(10),diag(10))
cor(collinX)
sum(cor(collinX)[lower.tri(cor(collinX))]>0.3) #0 correlation > 0.3
#500 variables
collinX<-rmvnorm(100,rnorm(100),diag(100))
sum(cor(collinX)[lower.tri(cor(collinX))]>0.3) #9
#1000 variables
collinX<-rmvnorm(100,rnorm(500),diag(500))
sum(cor(collinX)[lower.tri(cor(collinX))]>0.3) #172
#10000 variables
collinX<-rmvnorm(100,rnorm(1000),diag(1000))
sum(cor(collinX)[lower.tri(cor(collinX))]>0.3) #641
```

INTRODUCTION

CURSE OF DIMENSIONALITY

Multicollinearity + overfitting = high parameters estimates

Idea: Penalize the high β values during optimization

PLAN

1. Introduction
2. Model calibration
3. Cross-validation
4. High dimensional data analysis issues
5. Penalized regression
6. Alternatives methods
7. Conclusions

PENALIZED REGRESSION

Many penalized models. We'll see the 3 more classical in this course:

- Ridge
- Lasso
- Elastic net

Examples using R package glmnet:

- Predict biomarker change (gaussian variable)
- Predict survival

PENALIZED REGRESSION

PENALTY

Classical optimization for linear model: minimize the residual sum of squares (RSS)

$$min\{\frac{1}{N}\sum_{i=1}^{N}(y_i - \beta_0 - x_i^T\beta)^2\} = min\{RSS(\beta)\}$$

General purpose: Classical optimization for statistical model: maximize the log-likelihood (loglik)

$$max\{loglik(\beta)\} = min\{-loglik(\beta)\}$$

PENALIZED REGRESSION

PENALTY

Penalized likelihood optimization, minimize:

$$-loglik(\beta) + \lambda\ penalty(\beta) \quad (1)$$

ex: $l1$—norm penalty = $\sum_{j=1}^{p} |\beta_j|$

Increase of β increases the value of (1) that we want minimize
-> Favorize low β value

λ = tuning parameter, increase or decrease the strength of the penalty

Note 1: Different λ values lead to the same likelihood: lambda$ cannot be estimated with the other parameters using classical maximum likelihood method

Choose according another criteria: the prediction accuracy (using CV)

PENALIZED REGRESSION

PENALTY

Note 2: Using the $l1$—norm penalization, penalized likelihood equivalent to:

$$min\{-loglik(\beta)\} \text{ subject to } \sum_{j=1}^{p} |\beta_j| < t, \text{ with } < t < \infty$$

Maximal value of the sum of the parameters = t

Suggests a trade-off between the β: an increase of the number of parameters shrink all the βs (biased parameter estimates)

PENALIZED REGRESSION

RIDGE REGRESSION

Type 2 penalty:

$$-loglik(\beta) + \lambda \sum_{j=1}^{p} \beta_j^2$$

Shrink the parameters to low (non zero) values:

1. (+) Limit the β inflation due to overfitting and multicollinearity consequences
2. (-) No selection:
    - Complex interpretation
    - Trade-off between all parameter values -> high number of parameters = high constraint (low coefficient values, even for important variables)

PENALIZED REGRESSION

LASSO REGRESSION

Type 1 penalty:

$$-loglik(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$$

Can shrink the parameters values to 0 -> Allows selection:

- (+) Shrink the parameter of non-informative variables to 0 -> simpler interpretation than Ridge
- (-) For highly correlated $\mathbf{X}$, strong selection may shrink to 0 their parameters excepting that of the most correlated X to the response (i.e. may exclude informative variables, ex of lung cancer: smoking is selected, mutations related to smoking are not selected)

PENALIZED REGRESSION

ELASTIC NET

Elastic net regularization = mixture of LASSO and ridge penalty:

$$-loglik(\beta) + \lambda \left[\alpha \sum_{j=1}^{p} |\beta_j| + \frac{1-\alpha}{2} \sum_{j=1}^{p} \beta_j^2 \right]$$

(+) Weaker selection than LASSO (informative correlated variables moire often selected)

(-) Additional parameter to estimate (α)

PENALIZED REGRESSION

ELASTIC NET

Penalized regression: Suitable for prediction and selection

BUT...

Not suitable for "inference"

Biased parameter estimates:

- Value cannot be interpreted (biased Hazard Ratio of 1.1?)
- No p-value! (can perturb some practitionners)

PENALIZED REGRESSION

TOY EXAMPLE

Fit the 3 types of penalized regression:

```
#load package
library(glmnet)
#format the data for the function
y<-toyData$y
x<-as.matrix(subset(toyData,select=-c(y)))
#fit Ridge regression (alpha=0)
fitRidge<-glmnet(x=x,y=y,family="gaussian",alpha=0)

#fit elastic net regression (0<alpha<1, here use 0.5)
fitElasticNet<-glmnet(x=x,y=y,family="gaussian",alpha=0.5)

#fit LASSO regression (alpha=1)
fitLASSO<-glmnet(x=x,y=y,family="gaussian",alpha=1)
```

PENALIZED REGRESSION

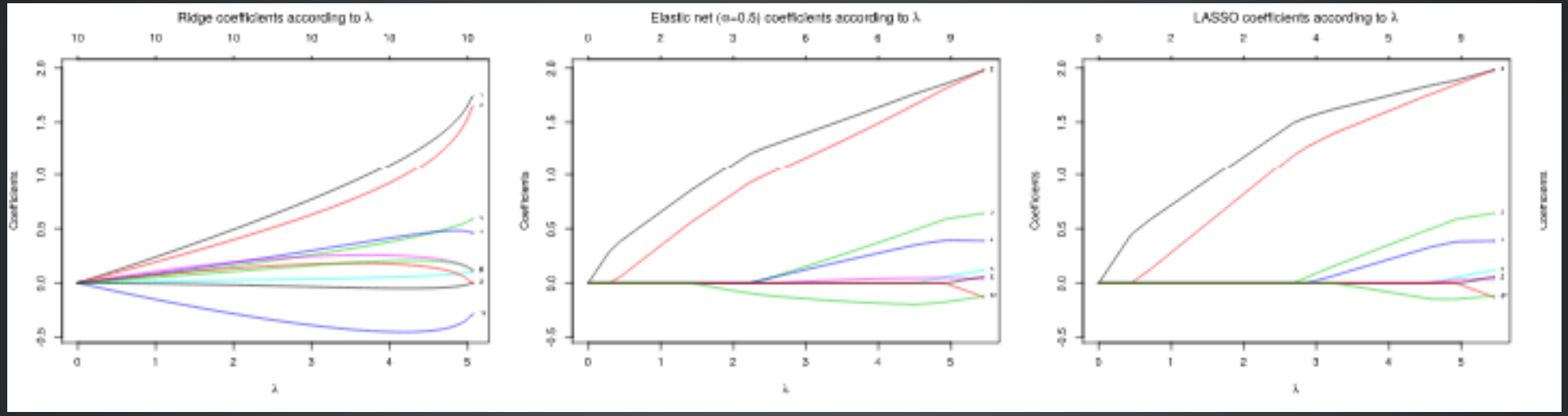
TOY EXAMPLE

Plot the values of the coefficients according to the λ value:

```
#get the min and the max coefficients values of each regression to standardize y-axis
#(in order to simplify comparison)
ylim<-range(c(as.vector(fitRidge$beta),
              as.vector(fitElasticNet$beta),
              as.vector(fitLASSO$beta)
            ))

#open new plot window
x11(width=14)
par(mfrow=c(1,3)) #align on 1 line the 3 plots

#plot coefficients of Ridge regression according to lambda value
plot(fitRidge,ylim=ylim,xlab=expression(lambda),label=TRUE)
title(expression(paste("Ridge coefficients according to ",lambda,sep="")),line=3)

#plot coefficients of elastic net regression according to lambda value
plot(fitElasticNet,ylim=ylim,xlab=expression(lambda),label=TRUE)
title(expression(paste("Elastic net (",alpha,
                "=0.5) coefficients according to ",lambda,sep="")),line=3)

#plot coefficients of LASSO regression according to lambda value
plot(fitLASSO,ylim=ylim,xlab=expression(lambda),label=TRUE)
title(expression(paste("LASSO coefficients according to ",lambda,sep="")),line=3)
```

PENALIZED REGRESSION

TOY EXAMPLE



How to choose λ value?

Cross validation: Prediction criteria = prediction accuracy

Easy to use with glmnet, just add "cv." to the function (cv.glmnet(...))
Default option of this function: Cross validation procedure will evaluate prediction accuracy of models with 100 different λ values

PENALIZED REGRESSION

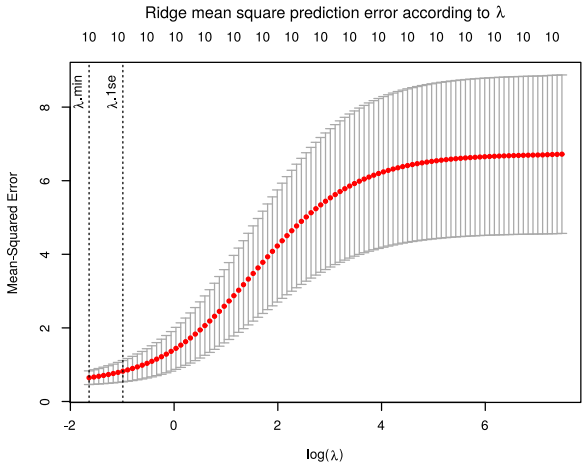
TOY EXAMPLE

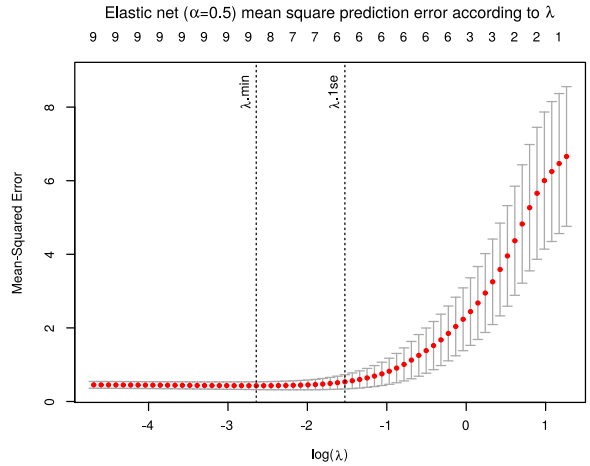
Plot the prediction error according to the λ value:

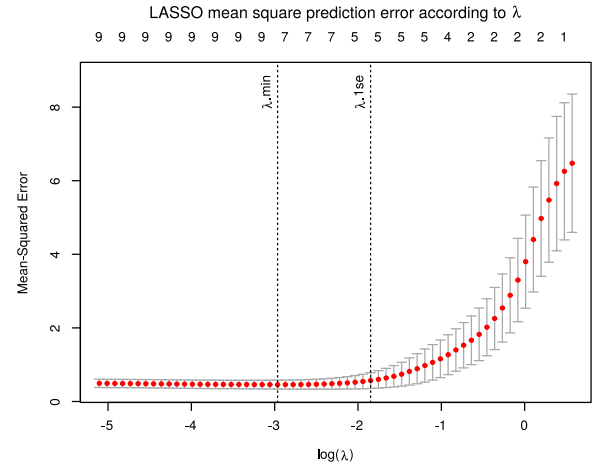
```
#use CV to measure MSE for different lambda values
#don't forget to initialize the seed!
set.seed(123)
fitRidge<-cv.glmnet(x=x,y=y,family="gaussian",alpha=0)
fitElasticNet<-cv.glmnet(x=x,y=y,family="gaussian",alpha=0.5)
fitLASSO<-cv.glmnet(x=x,y=y,family="gaussian",alpha=1)
#get the range of the confidence intervals of the cvl in order to standardize the y-axis
ylim<-c(min(c(fitRidge$cvlo,fitElasticNet$cvlo,fitLASSO$cvlo)),
        max(c(fitRidge$cvup,fitElasticNet$cvup,fitLASSO$cvup)))
x11(width=14)
par(mfrow=c(1,3))
#Plots
plot(fitRidge,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("Ridge mean square prediction error according to ",
                lambda,sep="")),line=3)
plot(fitElasticNet,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("Elastic net (",alpha,
                "=0.5) mean square prediction error according to ",lambda,sep="")),line=3)
plot(fitLASSO,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("LASSO mean square prediction error according to ",
                lambda,sep="")),line=3)
```

PENALIZED REGRESSION

TOY EXAMPLE







λ_{min} = minimal evaluated λ value?

Default range of evaluated λ is not adapted for ridge regression!

ADD arrow to lambda min of ridge! Tested lambda grid has to be extend to check if lower lambda value could be better

PENALIZED REGRESSION

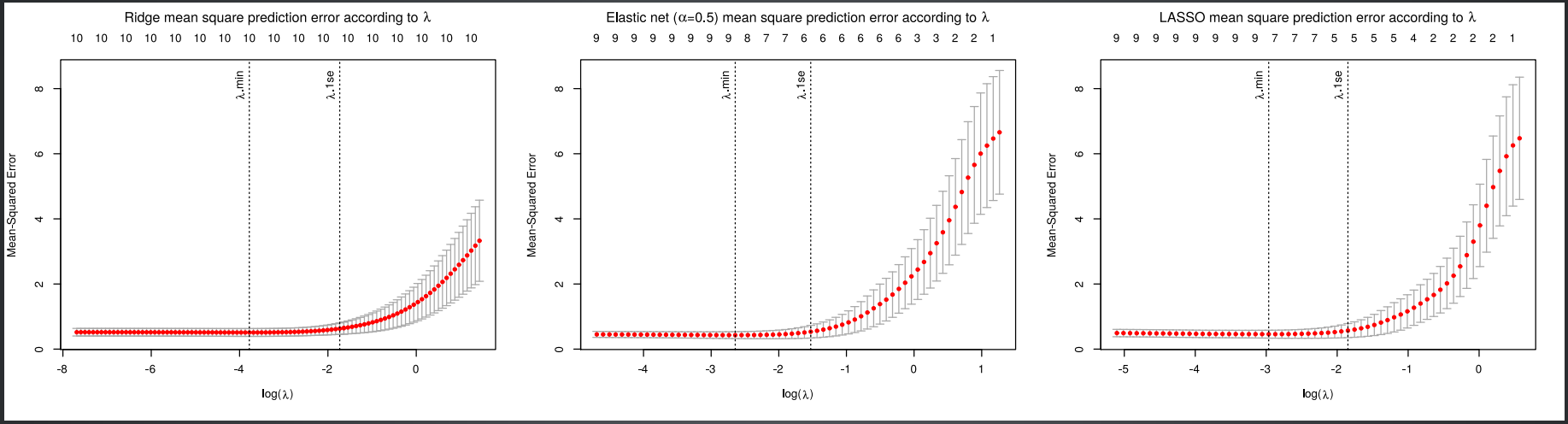
TOY EXAMPLE

Extend the grid of evaluated λs to the left

```
#get the previous grid of log lambda
ridgeGrid<-log(fitRidge$lambda)

#keep only the one-third lower lambda values (with the lower MSE values.
#The values at the right of the plot are not interesting because of their too high MSE)
third<-round(length(ridgeGrid)/3)

#lambda values are inversed, keep only the last one third of the vector
ridgeGrid<-ridgeGrid[c((2*third):length(ridgeGrid))]

#add the same number of new values with the same step
lastGridVal<-ridgeGrid[length(ridgeGrid)]
ridgeGrid<-exp(c(ridgeGrid,seq(lastGridVal,lastGridVal+2*third*diff(ridgeGrid)[1],
        by=diff(ridgeGrid)[1])))

#new fit of the model using the new lambda grid
set.seed(123)
fitRidge<-cv.glmnet(x=x,y=y,family="gaussian",alpha=0,lambda=ridgeGrid)
#Use the code for plot of the previous slide
```

PENALIZED REGRESSION

TOY EXAMPLE



λ_{min} or λ_{1se} (= higher λ value in 1 standard error interval of λ_{min})?
No consensus... (Ternes 2016)

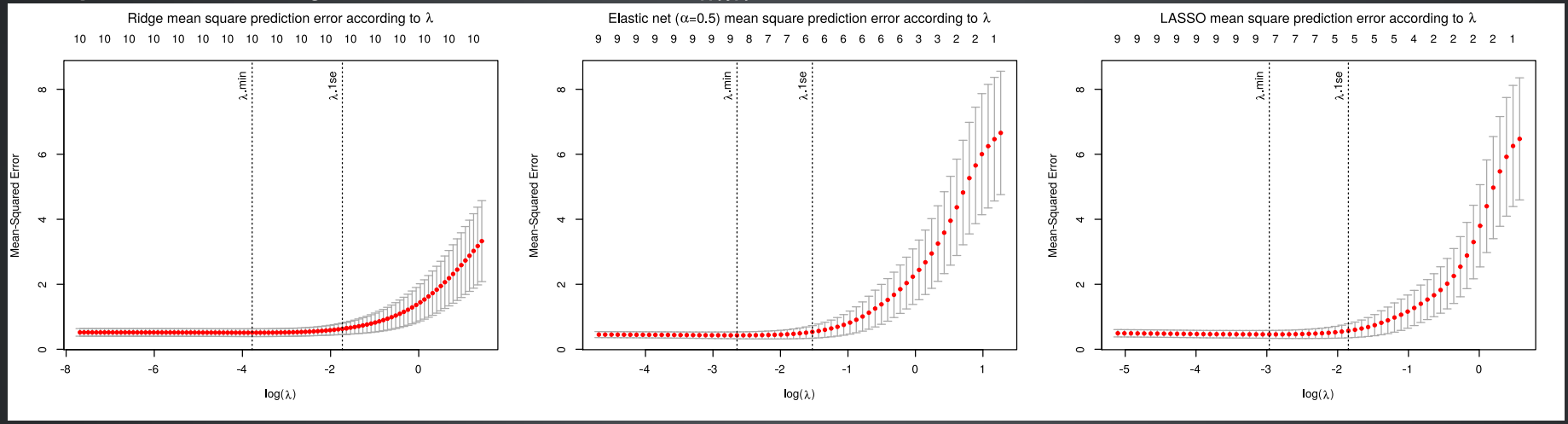
For the selection (elastic net and LASSO)

- λ_{min} provides to model with more selected variables : potential false positives (7-8 variables/4 with real effect)
- λ_{1se} provides to simpler model : potential false negatives (illustrated later)

PENALIZED REGRESSION

TOY EXAMPLE

For prediction objective, lets focus on λ_{min}



Which penalty? If selection is not an objective, compare the prediction error:

```
#Get minimum MSE (related to lambda min)
min(fitRidge$cvm)
min(fitElasticNet$cvm)
min(fitLASSO$cvm)
```

	Ridge	Elastic Net	LASSO
MSE of λ_{min}	0.51	0.43	0.46

Elastic Net! But CV is random procedure! Stability of this result?

PENALIZED REGRESSION

TOY EXAMPLE

Use several CV procedure and check MSE and variable selection

```
#get the function specific to this course
source("stability.R")
#Get the lambda.min of 100 different CV procedures (forgot the seed here)
#(seed is initialized to 123 for the first CV inside the function)
#can be change by the option seed=...)
ridgeCVs<-getLambda(x,y,lambdaType="lambda.min")
elasticNetCVs<-getLambda(x,y,lambdaType="lambda.min",alpha=0.5)
lassoCVs<-getLambda(x,y,lambdaType="lambda.min",alpha=1)
#"mean" MSE
mean(ridgeCVs$lambdaTypeMSE) #0.6433856
mean(elasticNetCVs$lambdaTypeMSE) #0.4400005
mean(lassoCVs$lambdaTypeMSE) #0.429817
```

LASSO! But EN results are very close:

```
x11()
plot(density(lassoCVs$lambdaTypeMSE),xlim=c(0.1,0.9),xlab="MSE",
    main="Distribution of MSE from 100 different CV")
lines(density(elasticNetCVs$lambdaTypeMSE),col=2)
lines(density(ridgeCVs$lambdaTypeMSE),col=3)
legend("topright",c("LASSO error","Elastic Net error","Ridge"),col=1:3,lty=1,bty="n")
```

Stability of variable selection?

PENALIZED REGRESSION

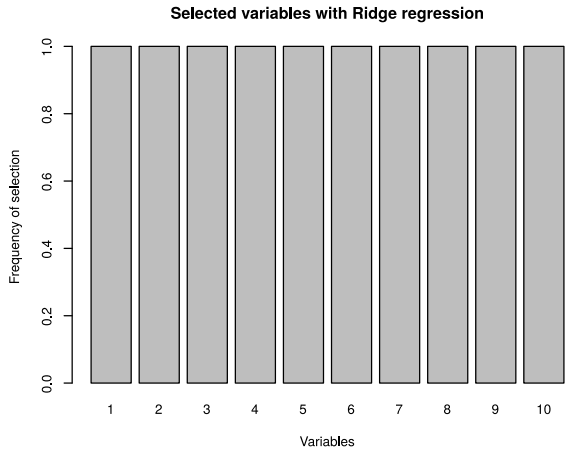
TOY EXAMPLE

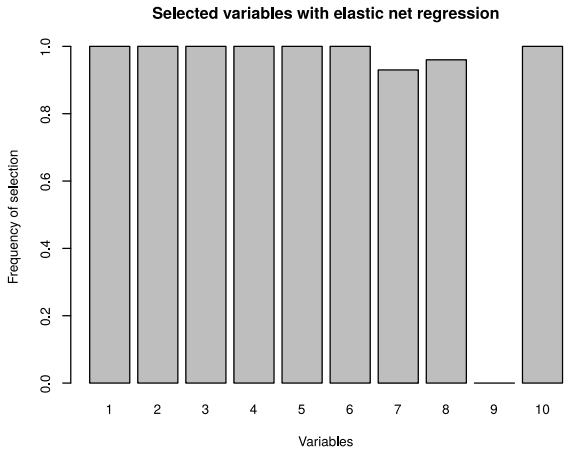
Selection frequency:

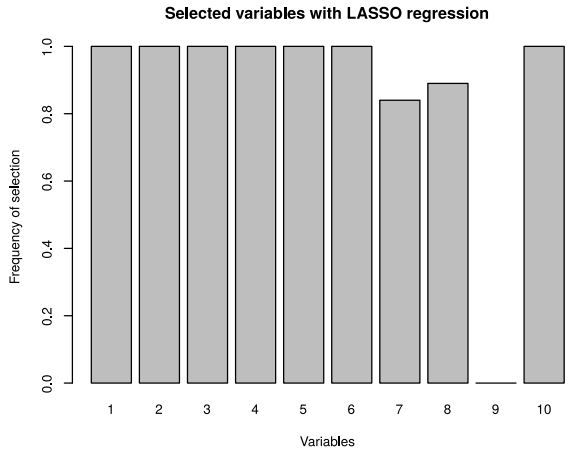
```
#plot the selection frequency of variables during the 100 different CV procedures
x11(width=14)
par(mfrow=c(1,3))
barplot(table(unlist(ridgeCVs$selectedVar))/100,xlab="Variables",
        ylab="Frequency of selection",main="Selected variables with Ridge regression")
barplot(completeTable(table(unlist(elasticNetCVs$selectedVar)),1:10)/100,xlab="Variables",
        ylab="Frequency of selection",main="Selected variables with elastic net regression")
barplot(completeTable(table(unlist(lassoCVs$selectedVar)),1:10)/100,xlab="Variables",
        ylab="Frequency of selection",main="Selected variables with LASSO regression")
```

PENALIZED REGRESSION

TOY EXAMPLE







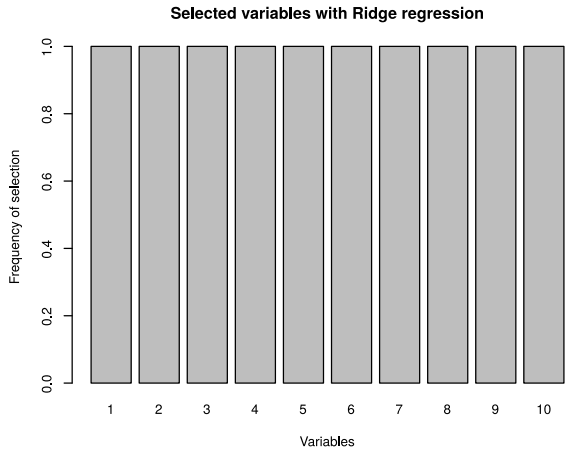
Ridge does not make selection (variables selected all the times)

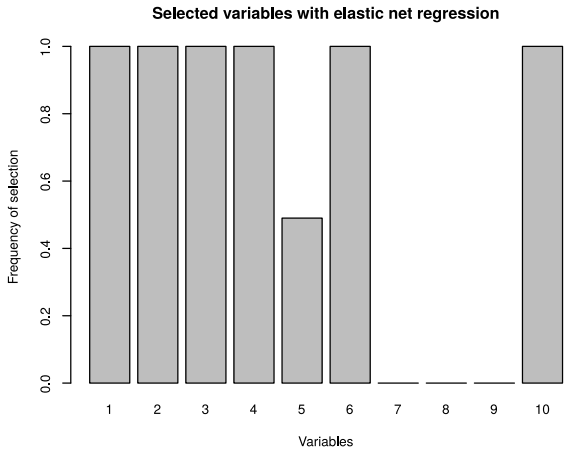
All variables with true effect (X1 to X4) are always selected by LASSO and EN

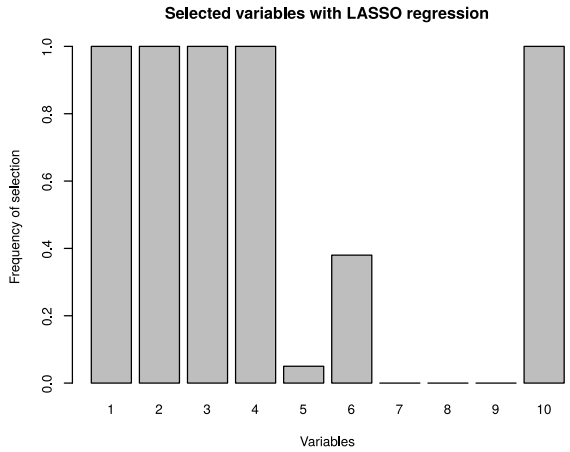
EN less strict than LASSO = higher number of false positives with EN (in this case)

PENALIZED REGRESSION

TOY EXAMPLE







Less selected variables

-> closer to true model (X1 to X4)

BUT higher prediction error

PENALIZED REGRESSION

TOY EXAMPLE

Compare with standard selection approach:

```
#backward selection
library(MASS) #For stepAIC function
fit<-lm(y~x1+x2+x3+x4+x5+x6+x7+x8+x9+x10,data=toyData)
final<-stepAIC(fit,direction="both",trace=F)$coef
names(final)
#final model: y~x1+x2+x3+x4+x5+x8

#compare predictive accuracy with LASSO using CV
subSamples<-rep(1:10,round(nrow(toyData))/10)
set.seed(123)
subSamples<-sample(subSamples,replace=FALSE)
stockResults<-NULL
for(k in 1:10){
    fit0<-lm(y~x1+x2+x3+x4+x5+x8,data=toyData[subSamples!=k,])
    pred0<-predict(fit0,newdata=toyData[subSamples==k,])
    sqError0<-sum((toyData$y[subSamples==k]-pred0)^2)
    stockResults<-c(stockResults,sqError0)
}
mean(stockResults) #1.142053
set.seed(123)
fit1<-cv.glmnet(x=x,y=y,family="gaussian",alpha=1,foldid=subSamples)
fit1$cvm[fit1$lambda==fit1$lambda.1se]
#0.577808 = LASSO with lambda.1se better than model selected by AIC
min(fit1$cvm)
#0.4741752 = LASSO with lambda.min is the best
```

PENALIZED REGRESSION

TOY EXAMPLE

Compare with standard selection approach:

```
#compare predictive accuracy with LASSO on new data
#fit the "best" model according to the CV on the entire toyData
fit0<-lm(y~x1+x2+x3+x4+x5+x8,data=toyData)

#Create a new data
set.seed(1) #another seed than toyData (123)
x2<-sapply(1:10,function(x)rnorm(n))
y2<-rnorm(n,cbind(rep(1,n),x2)%*%betas,0.5)
toyData2<-data.frame(y=y2,x=x2)
colnames(toyData2)<-c("y",paste0("x",1:10))
#Predict on the new data according to the model fitted on toyData
pred0<-predict(fit0,newdata=toyData2)
x2<-as.matrix(subset(toyData2,select=-c(y)))
#Predict using the model with lambda.min value of the fit on toyData
pred1<-predict(fit1,newx=x2,s="lambda.min")
pred2<-predict(fit1,newx=x2,s="lambda.1se")
sum((toyData2$y-pred0)^2) #9.201504
sum((toyData2$y-pred1)^2) #8.342336  = LASSO provided better predictions
sum((toyData2$y-pred2)^2)
#Larger error than CV, but cannot be compared:
#CV is mean of the sum of squared error of 30/10=3 samples
#Here, it is the sum of squared error of 30 samples
```

PENALIZED REGRESSION

TOY EXAMPLE

Conclusion (for this example):

1. LASSO and elastic net had more predictive accuracy due to their lower number of included variables
2. Including all variables, ridge regression had to reduce the values of the coefficients relatively to compensate the coefficients of the false positives

All variables with true effect (X1 to X4) are always selected by LASSO and EN

EN less strict than LASSO = higher number of false positives with EN (in this case)

PENALIZED REGRESSION

APPLICATION 1

Breast cancer datasets:

- Gene expression (add source):
  - Context:
  - Primary endpoint: Predict the absolute change at 3 months in the log-scale expression of aurora kinase A (AURKA) gene involved in regulation, proliferation and cell survival.
  - n=56, p=8143

PENALIZED REGRESSION

APPLICATION 1

Format the data:

```
#load package
library(glmnet)
#import data
load("f_overall.Rdata")
data<-f.overall #assign data to on object with simpler name
dim(data) #be careful for vizualization! 8145 columns!
#get response variable
y<-data[,grep("y_A",colnames(data))]
#check the distribution of response variable
plot(density(y)) #Gaussian approximation acceptable
#get explicative variables
x<-as.matrix(data[,-grep("y_",colnames(data))])
```

PENALIZED REGRESSION

APPLICATION 1

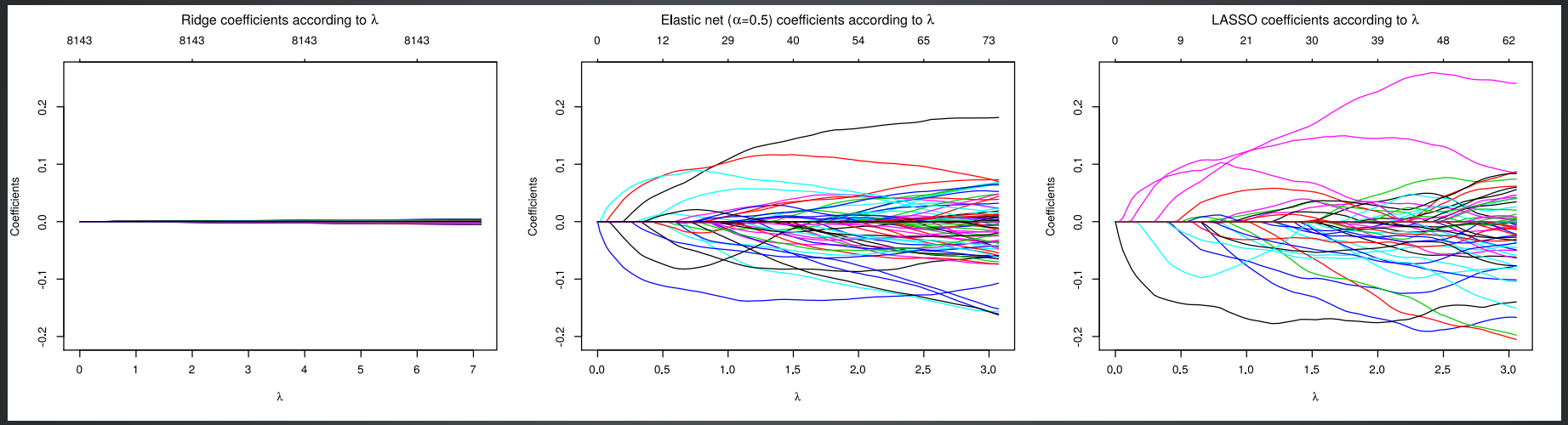
Plot the values of the coefficients according to the λ value:

```
#Fit models
fitRidge<-glmnet(x=x,y=y,family="gaussian",alpha=0)
fitElasticNet<-glmnet(x=x,y=y,family="gaussian",alpha=0.5)
fitLASSO<-glmnet(x=x,y=y,family="gaussian",alpha=1)

#get the range of the coefficient to standardize plot y-axes
ylim<-range(c(as.vector(fitRidge$beta),as.vector(fitElasticNet$beta),
              as.vector(fitLASSO$beta)))

#plots
x11(width=14)
par(mfrow=c(1,3))
plot(fitRidge,ylim=ylim,xlab=expression(lambda))
title(expression(paste("Ridge coefficients according to ",lambda,sep="")),line=3)
plot(fitElasticNet,ylim=ylim,xlab=expression(lambda))
title(expression(paste("Elastic net (",alpha,"=0.5) coefficients according to ",
                       lambda,sep="")),line=3)
plot(fitLASSO,ylim=ylim,xlab=expression(lambda))
title(expression(paste("LASSO coefficients according to ",lambda,sep="")),line=3)
```

PENALIZED REGRESSION

APPLICATION 1

Results:



Lots of parameters! (p=8143)

Constrain of the sum of parameter lead the Ridge penalty to compensate the effect of the p variables -> all parameters values are shrinked to low non-zero values
As several LASSO and EN coefficients are equal to 0, the constrain on the other parameters is lower (thus they can have higher values)

PENALIZED REGRESSION

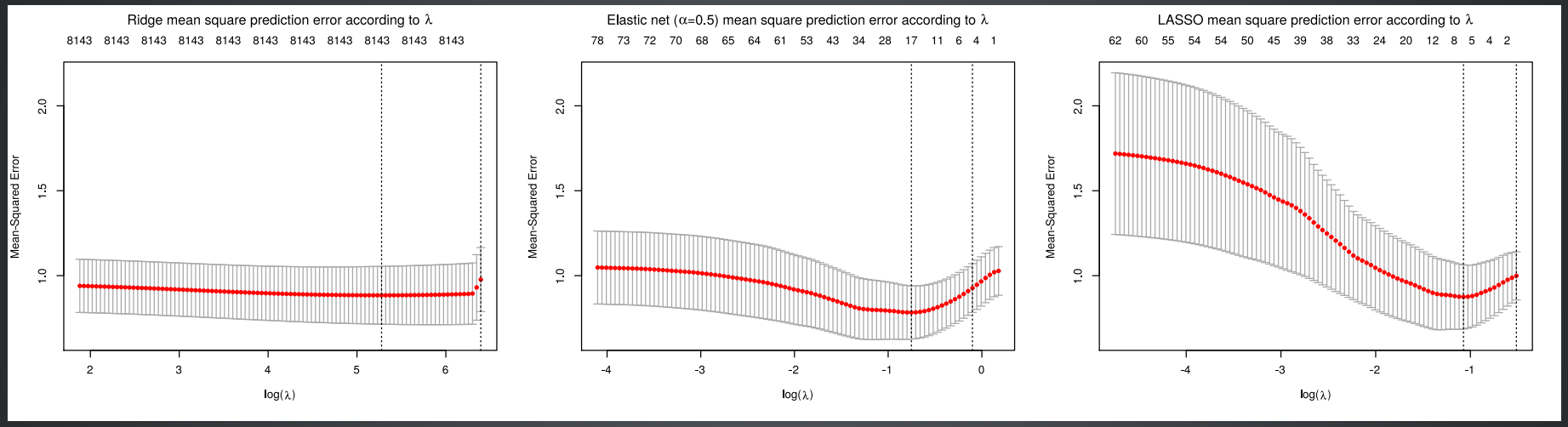
APPLICATION 1

Prediction accuracy?

```
#use CV to measure MSE for different lambda values
#don't forget to initialize the seed!
set.seed(123)
fitRidge<-cv.glmnet(x=x,y=y,family="gaussian",alpha=0)
fitElasticNet<-cv.glmnet(x=x,y=y,family="gaussian",alpha=0.5)
fitLASSO<-cv.glmnet(x=x,y=y,family="gaussian",alpha=1)
#get the range of the confidence intervals of the cvl in order to standardize the y-axis
ylim<-c(min(c(fitRidge$cvlo,fitElasticNet$cvlo,fitLASSO$cvlo)),
        max(c(fitRidge$cvup,fitElasticNet$cvup,fitLASSO$cvup)))
x11(width=14)
par(mfrow=c(1,3))
#Plots
plot(fitRidge,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("Ridge mean square prediction error according to ",
                lambda,sep="")),line=3)
plot(fitElasticNet,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("Elastic net (",alpha,
                "=0.5) mean square prediction error according to ",lambda,sep="")),line=3)
plot(fitLASSO,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("LASSO mean square prediction error according to ",
                lambda,sep="")),line=3)
```

PENALIZED REGRESSION

APPLICATION 1



λ_{1se} is the last value to the right for Ridge and LASSO

The range of evaluated λ is to limited

Extend the range of λ to the right (if you plan to use λ_{1se})

PENALIZED REGRESSION

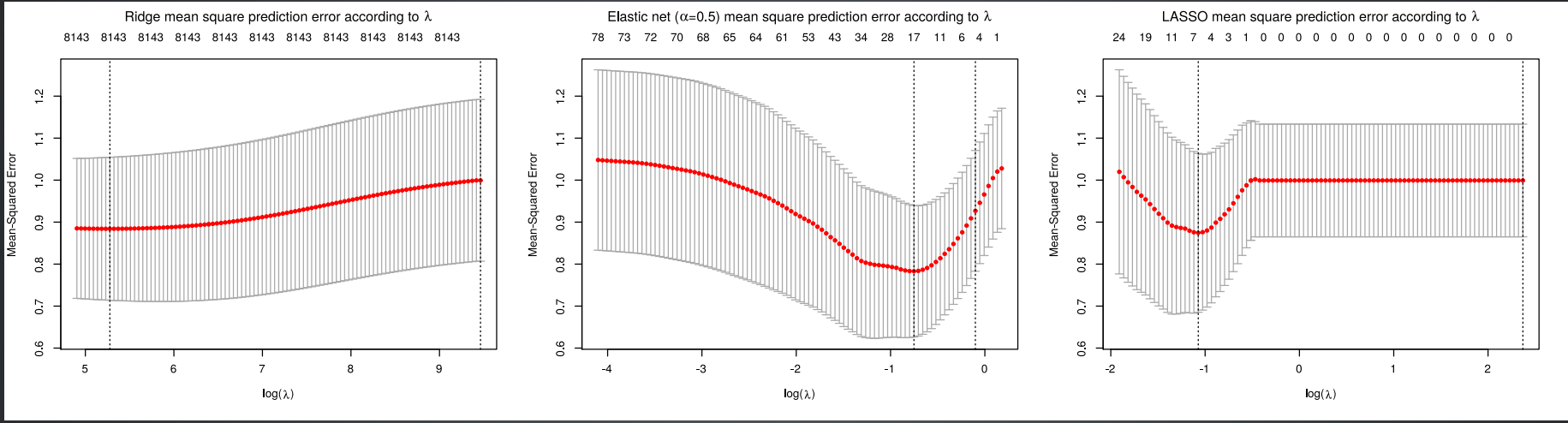
APPLICATION 1

Extend the grid of evaluated λs to the left

```
#get the previous grid of log lambda
ridgeGrid<-log(fitRidge$lambda)
lassoGrid<-log(fitLASSO$lambda)
#keep only the one-third lower lambda values (with the lower MSE values.
#The values at the right of the plot are not interesting because of their too high MSE)
ridgeThird<-round(length(ridgeGrid)/3)
lassoThird<-round(length(lassoGrid)/3)
#lambda values are inversed, keep only the first one third of the vector
ridgeGrid<-ridgeGrid[1:ridgeThird]
lassoGrid<-lassoGrid[1:lassoThird]
#add the same number of new values with the same step
firstGridVal<-ridgeGrid[1]
ridgeGrid<-exp(c(seq(firstGridVal-2*ridgeThird*diff(ridgeGrid)[1],firstGridVal,
        by=diff(ridgeGrid)[1]),ridgeGrid))
firstGridVal<-lassoGrid[1]
lassoGrid<-exp(c(seq(firstGridVal-2*lassoThird*diff(lassoGrid)[1],firstGridVal,
        by=diff(lassoGrid)[1]),lassoGrid))

#new fit of the models using the new lambda grids
set.seed(123)
fitRidge<-cv.glmnet(x=x,y=y,family="gaussian",alpha=0,lambda=ridgeGrid)
#refit EN to ensure that the seed is the same for fitLASSO than in the previous slide
fitElasticNet<-cv.glmnet(x=x,y=y,family="gaussian",alpha=0.5)
fitLASSO<-cv.glmnet(x=x,y=y,family="gaussian",alpha=1,lambda=lassoGrid)
#Use the code of the previous slide (from ylim<-...) for plot
```

PENALIZED REGRESSION

APPLICATION 1



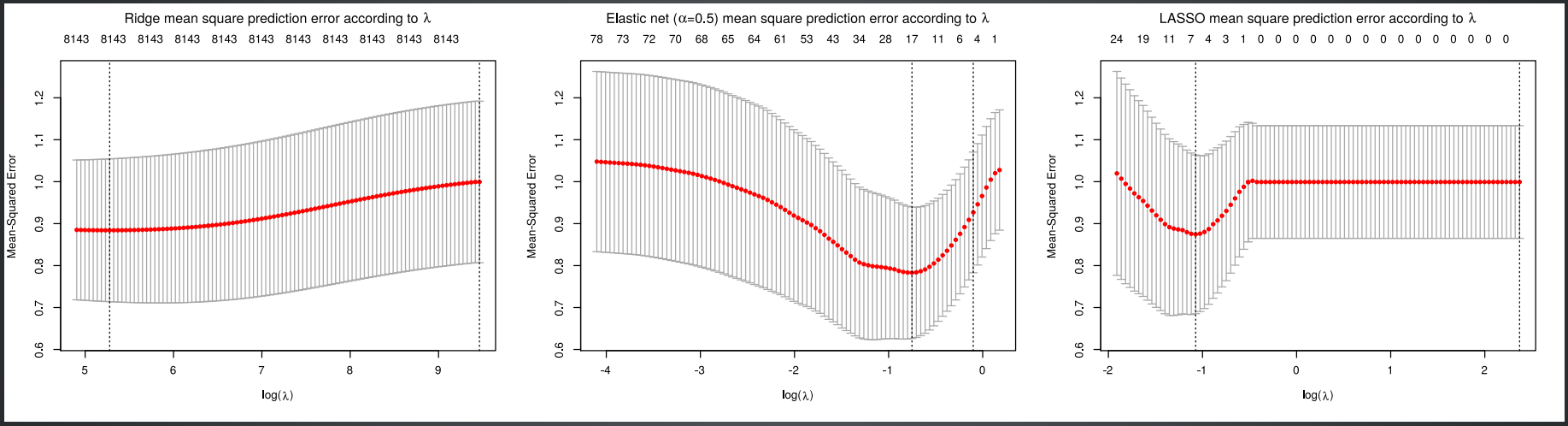
Due to the large standard error and the slow increase of MSE, the lambda grid should be extended to very large λ values to obtain λ_{1se} for Ridge

λ_{1se} for LASSO is all λ values for the NULL model: no unique solution, but 0 selected variables...

With prediction objective, we will use the λ_{min} in the next step

PENALIZED REGRESSION

APPLICATION 1




```
#Get minimum MSE (related to lambda min)
min(fitRidge$cvm)
min(fitElasticNet$cvm)
min(fitLASSO$cvm)
```

	Ridge	Elastic Net	LASSO
MSE of λ_{min}	0.88	0.78	0.87

Elastic Net! But don't forgot that CV is random procedure! Stability of this result?

PENALIZED REGRESSION

APPLICATION 1

Use several CV procedure and check MSE and variable selection

```
#Get the lambda.min of 100 different CV procedures (forgot the seed here)
ridgeCVs<-getLambda(x,y,lambdaType="lambda.min")
elasticNetCVs<-getLambda(x,y,lambdaType="lambda.min",alpha=0.5)
lassoCVs<-getLambda(x,y,lambdaType="lambda.min",alpha=1)
#"mean" MSE
mean(ridgeCVs$lambdaType) #243.7545!!!!
mean(elasticNetCVs$lambdaType) #0.5870683
mean(lassoCVs$lambdaType) #0.3107707
```

LASSO!

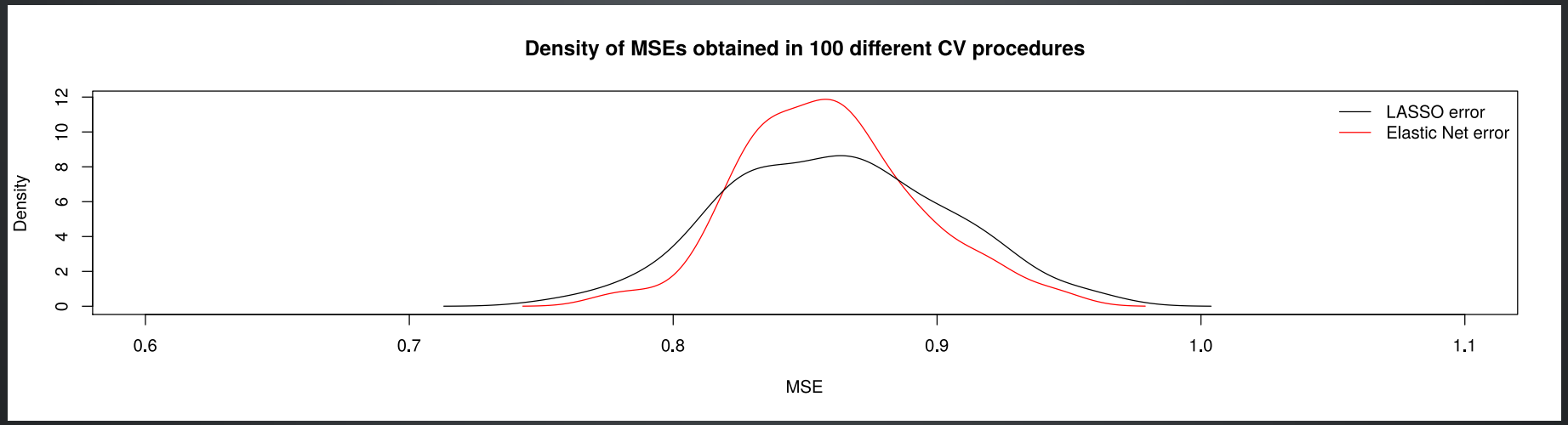
Ridge has very high MSE, likely due to a too high number of non-zero parameters (overfitting?)

PENALIZED REGRESSION

APPLICATION 1

LASSO and EN give similar results

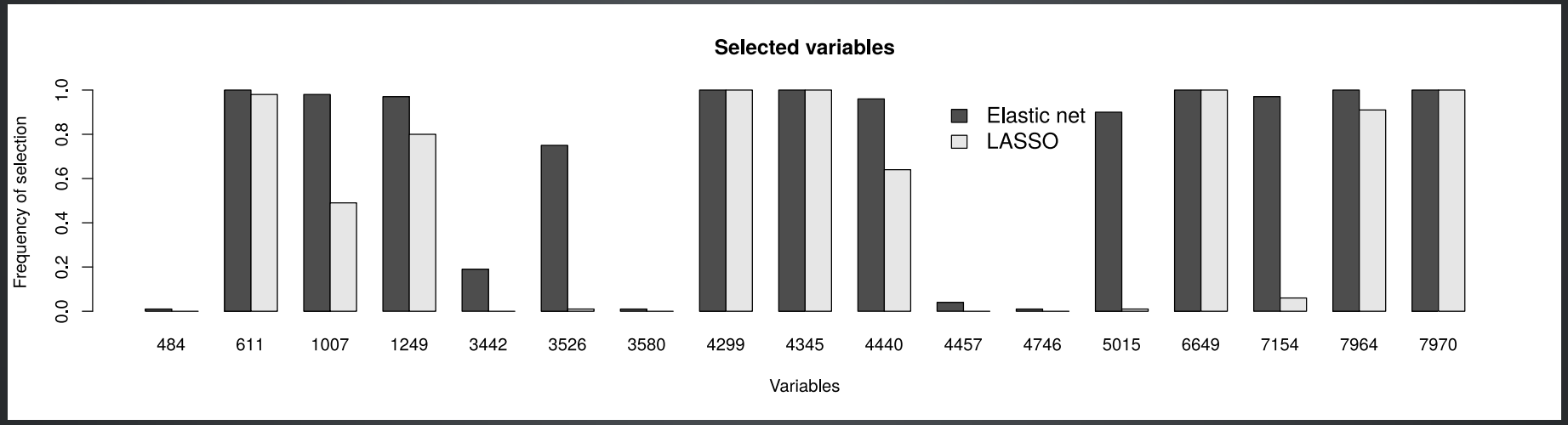
```
x11()
plot(density(lassoCVs$lambdaTypeMSE),xlim=c(0.6,1.1),
            main="Density of MSEs obtained in 100 different CV procedures",xlab="MSE")
lines(density(elasticNetCVs$lambdaTypeMSE),col=2)
legend("topright",c("LASSO error","Elastic Net error"),col=1:2,lty=1,bty="n")
```



PENALIZED REGRESSION

APPLICATION 1

Stable variable selection?



EM selected only 17 variables, including the 12 selected by LASSO in at least 1 CV

But, only 6 variables selected by LASSO in more than 90% of CV

PENALIZED REGRESSION

APPLICATION 1

For similar predictive accuracy:

- 6 genes selected by LASSO
- 17 selected by EN

Presentation of the results?

Proposition (illustrative example):

Gene name	Gene description	Selected by LASSO?	Selected by elastic net?
Gene1	zinc finger protein 106 homolog (mouse)	yes	yes
Gene2	mitochondrial ribosomal protein L4	No	yes
...	...	...	...
Gene17	ribonucleotide reductase M2	yes	yes

PENALIZED REGRESSION

APPLICATION 1

BUT...

Validated only by CV procedure -> require new dataset to test this model

If you have new dataset use its covariates matrix `newDataX` to define the predictive score:

```
predScore<-predict(fitLASSO,newx=newDataX,s="lambda.min")
```

PENALIZED REGRESSION

APPLICATION 2

Breast cancer datasets:

- Ternes *et al*. 2017 (biospear R package):
  - Primary endpoint: Distant-relapse free survival (DFS) time (in years)
  - Treatment arm (Anthracycline-based adjuvant chemotherapy with (treat = +0.5) or without (treat = -0.5) taxane)
  - n=614, p=1689 (gene expression values)

PENALIZED REGRESSION

APPLICATION 2

Format the data:

```
#load package
library(glmnet)
#import data
load("Breast.rda")
dim(Breast) #be careful for vizualization! 8145 columns!
#get response variable
y<-Surv(Breast$time,Breast$status)
plot(y)
#get trt and gene expression values
x<-as.matrix(Breast[,-c(1:2)])
```

PENALIZED REGRESSION

APPLICATION 2

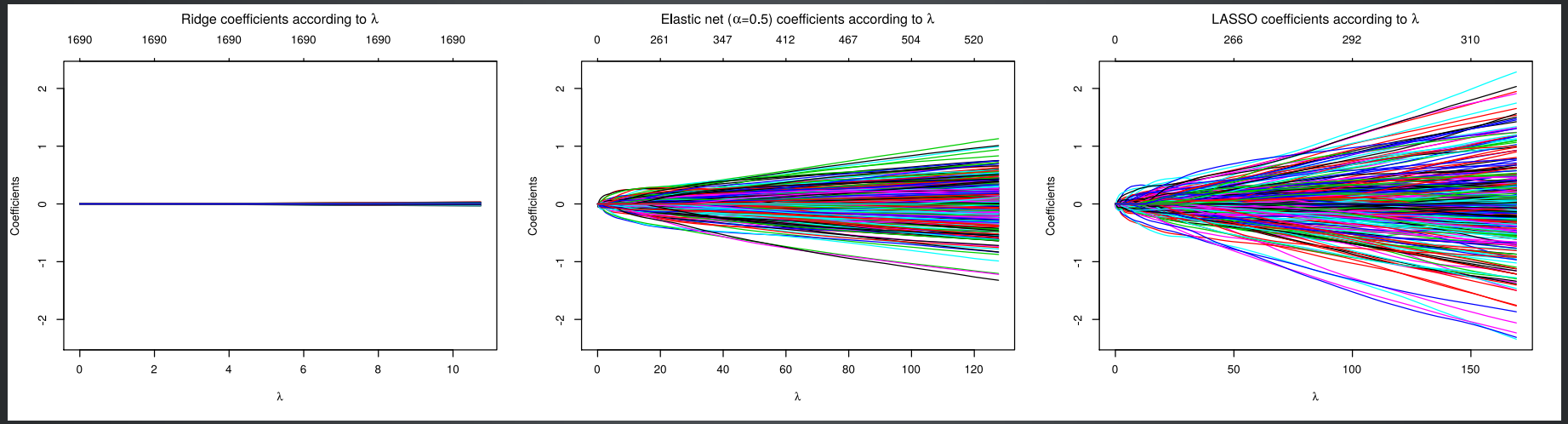
Plot the values of the coefficients according to the λ value:

```
#Fit models
fitRidge<-glmnet(x=x,y=y,family="cox",alpha=0)
fitElasticNet<-glmnet(x=x,y=y,family="cox",alpha=0.5)
fitLASSO<-glmnet(x=x,y=y,family="cox",alpha=1)

#get the range of the coefficient to standardize plot y-axes
ylim<-range(c(as.vector(fitRidge$beta),as.vector(fitElasticNet$beta),
              as.vector(fitLASSO$beta)))

#plots
x11(width=14)
par(mfrow=c(1,3))
plot(fitRidge,ylim=ylim,xlab=expression(lambda))
title(expression(paste("Ridge coefficients according to ",lambda,sep="")),line=3)
plot(fitElasticNet,ylim=ylim,xlab=expression(lambda))
title(expression(paste("Elastic net (",alpha,"=0.5) coefficients according to ",
                       lambda,sep="")),line=3)
plot(fitLASSO,ylim=ylim,xlab=expression(lambda))
title(expression(paste("LASSO coefficients according to ",lambda,sep="")),line=3)
```

PENALIZED REGRESSION

APPLICATION 2

Results:



Constrain of the sum of parameter lead the Ridge penalty to compensate the effect of the p variables -> all parameters values are shrinked to 0
As several LASSO and EN coefficients are equal to 0, the constrain on the other parameters is lower (thus they can have higher values)

PENALIZED REGRESSION

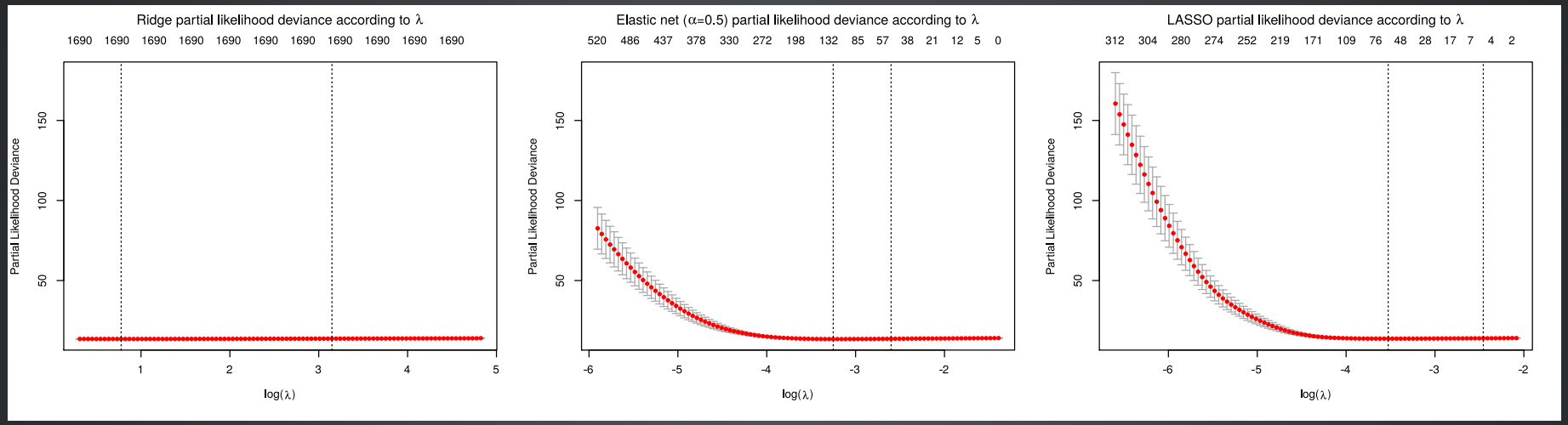
APPLICATION 2

Prediction accuracy?

```
#use CV to measure MSE for different lambda values
#don't forget to initialize the seed!
set.seed(123)
fitRidge<-cv.glmnet(x=x,y=y,family="cox",alpha=0)
fitElasticNet<-cv.glmnet(x=x,y=y,family="cox",alpha=0.5)
fitLASSO<-cv.glmnet(x=x,y=y,family="cox",alpha=1)
#get the range of the confidence intervals of the cvl in order to standardize the y-axis
ylim<-c(min(c(fitRidge$cvlo,fitElasticNet$cvlo,fitLASSO$cvlo)),
        max(c(fitRidge$cvup,fitElasticNet$cvup,fitLASSO$cvup)))
x11(width=14)
par(mfrow=c(1,3))
#Plots
plot(fitRidge,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("Ridge mean square prediction error according to ",
                lambda,sep="")),line=3)
plot(fitElasticNet,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("Elastic net (",alpha,
                "=0.5) mean square prediction error according to ",lambda,sep="")),line=3)
plot(fitLASSO,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("LASSO mean square prediction error according to ",
                lambda,sep="")),line=3)
```

PENALIZED REGRESSION

APPLICATION 2

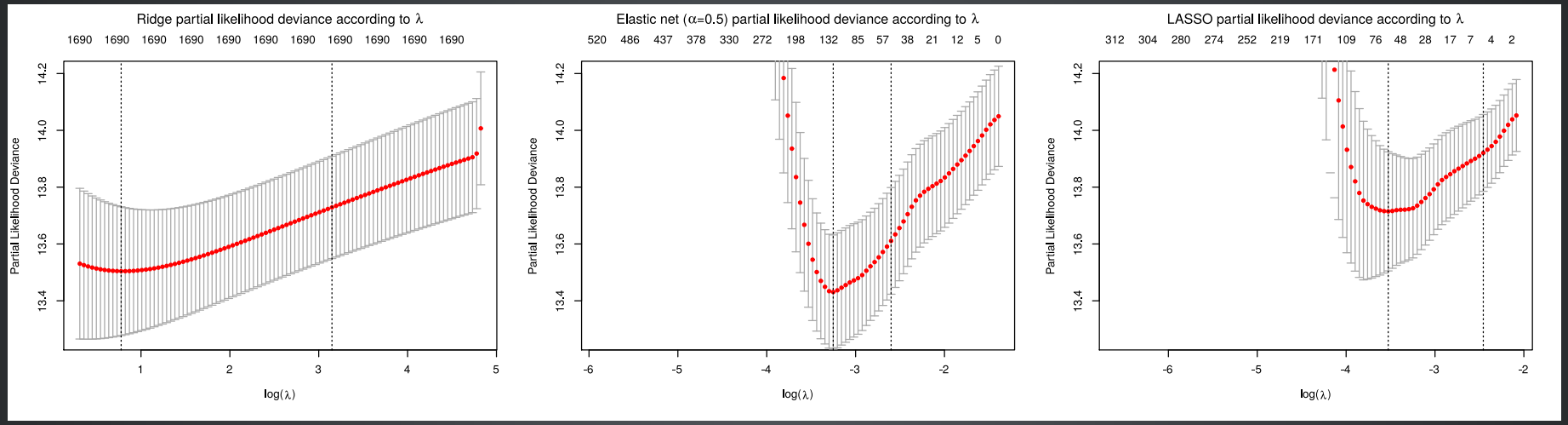


MSE extremely stable according to the λ of Ridge regression -> Zoom!

```
ylim<-c(min(fitRidge$cvlo),max(fitRidge$cvup)) #just focus on the Ridge plot coordinate
x11(width=14)
par(mfrow=c(1,3))
plot(fitRidge,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("Ridge mean square prediction error according to ",
                lambda,sep="")),line=3)
plot(fitElasticNet,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("Elastic net (",alpha,
                "=0.5) mean square prediction error according to ",lambda,sep="")),line=3)
plot(fitLASSO,ylim=ylim,xlab=expression(paste("log(",lambda,")",sep="")))
title(expression(paste("LASSO mean square prediction error according to ",
                lambda,sep="")),line=3)
```

PENALIZED REGRESSION

APPLICATION 2



High number of selected variables using λ_{min}

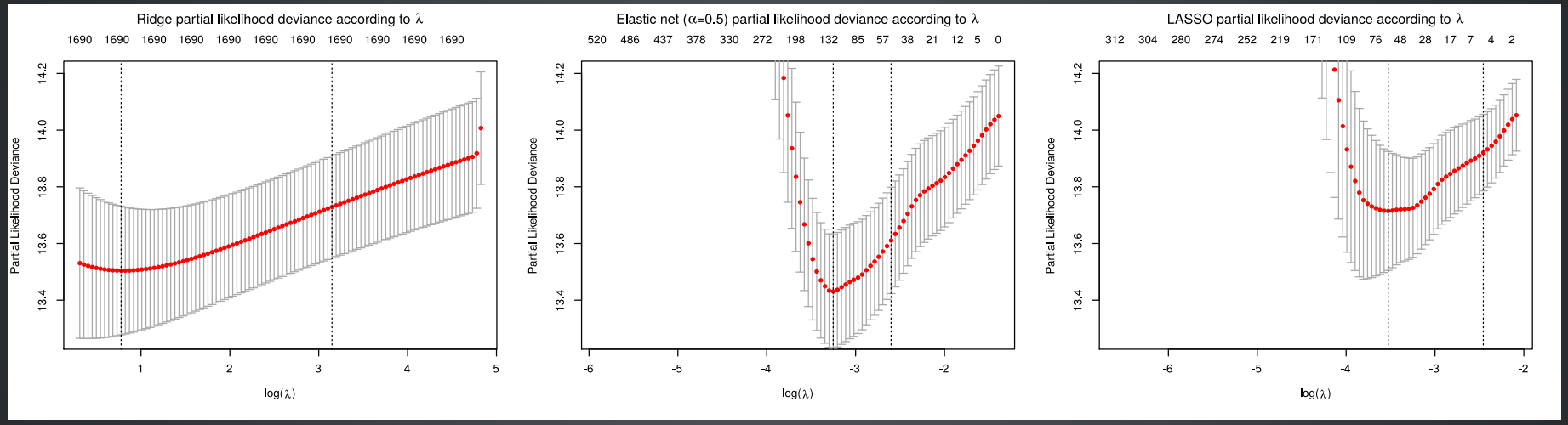
```
fitElasticNet$nzero[fitElasticNet$lambda==fitElasticNet$lambda.min] #122
fitLASSO$nzero[fitLASSO$lambda==fitLASSO$lambda.min] #61
```

Stronger selection using λ_{1se} (but higher error...)

```
fitElasticNet$nzero[fitElasticNet$lambda==fitElasticNet$lambda.1se] #48
fitLASSO$nzero[fitLASSO$lambda==fitLASSO$lambda.1se] #6
```

PENALIZED REGRESSION

APPLICATION 2




```
#Get minimum MSE (related to lambda min)
min(fitRidge$cvm)
min(fitElasticNet$cvm)
min(fitLASSO$cvm)
```

	Ridge	Elastic Net	LASSO
MSE of λ_{min}	13.50	13.43	13.71

Elastic Net! But all are very closed Stability of this result?

PENALIZED REGRESSION

APPLICATION 2

Use several CV procedure and check MSE and variable selection

```
#Get the lambda.min of 100 different CV procedures (forgot the seed here)
ridgeCVs<-getLambda(x,y,lambdaType="lambda.min")
elasticNetCVs<-getLambda(x,y,lambdaType="lambda.min",alpha=0.5)
lassoCVs<-getLambda(x,y,lambdaType="lambda.min",alpha=1)
#"mean" Deviance
mean(ridgeCVs$lambdaTypeMSE) #13.55854
mean(elasticNetCVs$lambdaTypeMSE) #13.53134
mean(lassoCVs$lambdaTypeMSE) #13.56802
```

Very closed results and stable!

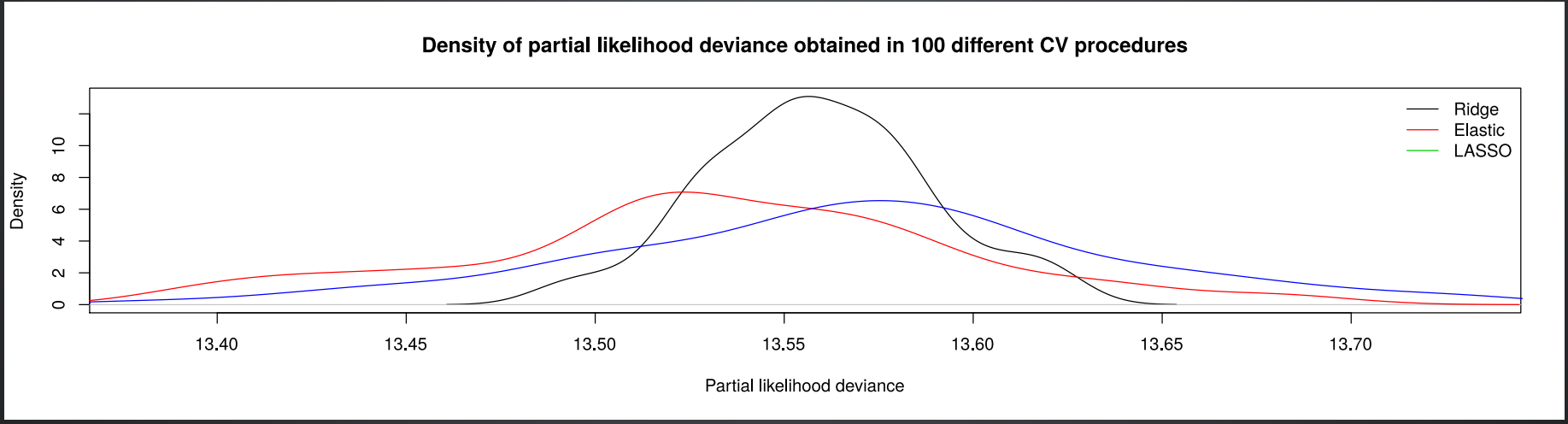
```
range(ridgeCVs$lambdaTypeMSE) #min=13.49009; max=13.62443
range(elasticNetCVs$lambdaTypeMSE) #min=13.39271; max=13.68951
range(lassoCVs$lambdaTypeMSE) #min=13.38026; max=13.73086
```

PENALIZED REGRESSION

APPLICATION 2

Ridge results more stable:

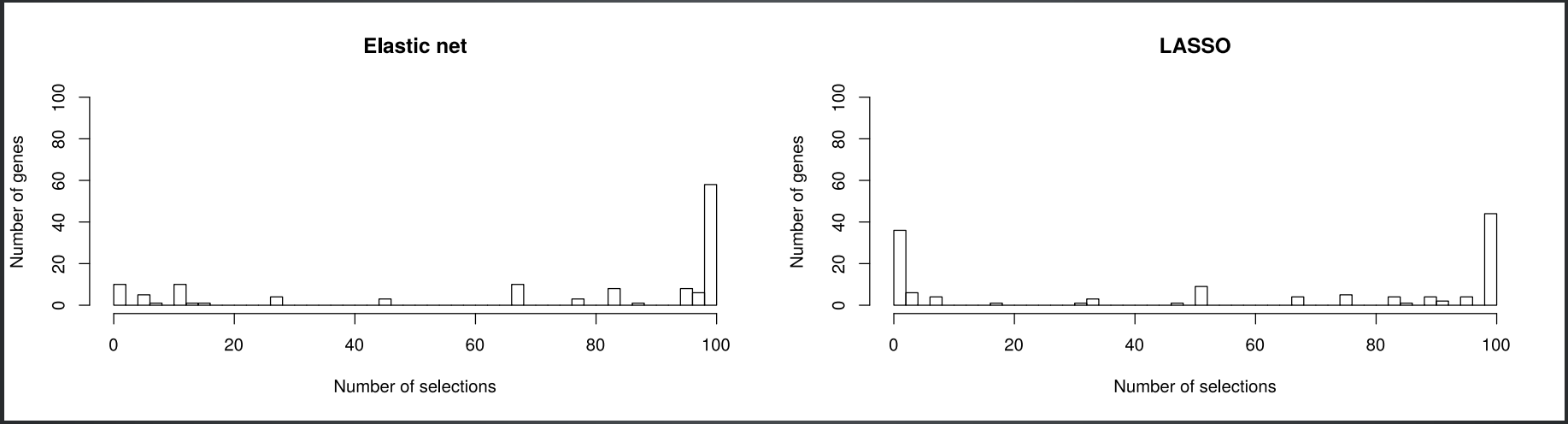
```
x11()
plot(density(lassoCVs$lambdaTypeMSE),xlim=c(0.6,1.1),
            main="Density of partial likelihood deviances obtained in 100 different CV procedures",
            xlab="Partial likelihood deviance")
lines(density(elasticNetCVs$lambdaTypeMSE),col=2)
legend("topright",c("LASSO error","Elastic Net error"),col=1:2,lty=1,bty="n")
```



PENALIZED REGRESSION

APPLICATION 2

Stable variable selection?



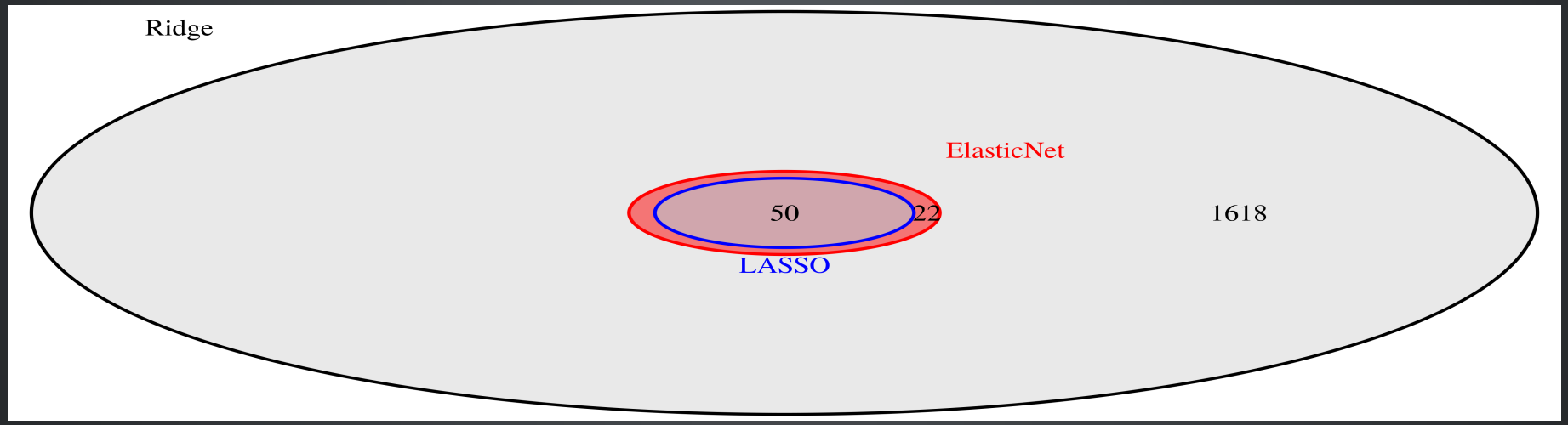
72 genes selected in more than 90% of CV by EN

50 genes selected in more than 90% of CV by LASSO

PENALIZED REGRESSION

APPLICATION 2

Stable variable selection across method?



(Venn diagram for genes selected in more than 90%)

PENALIZED REGRESSION

APPLICATION 2

DON'T FORGET THAT

These results are validated only by CV procedure -> require new dataset to test this model

PLAN

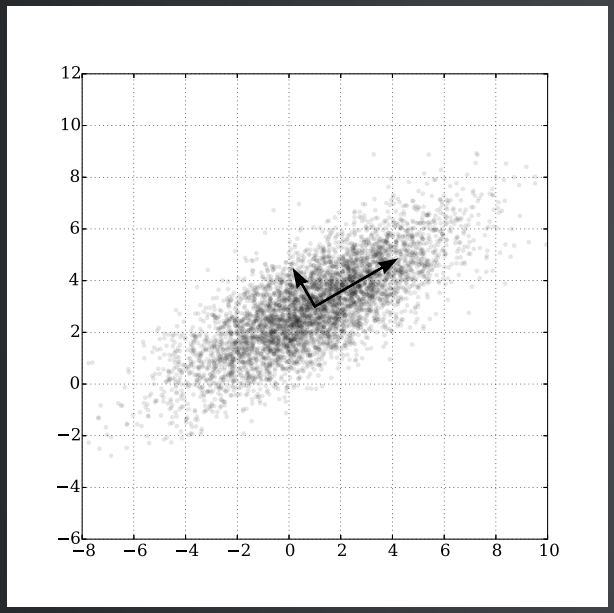
1. Introduction
2. Model calibration
3. Cross-validation
4. High dimensional data analysis issues
5. Penalized regression
6. Alternative methods
7. Conclusions

ALTERNATIVE METHODS

1. Dimension reduction
2. Causal approach
3. Machine learning algorithms

ALTERNATIVE METHODS

DIMENSION REDUCTION



- Principal components analysis
- Correspondence analysis
- Independent components analysis
- PLS regression
- Factor analysis
- ...

Reduce p to smaller k components
Limits = no selection, all biomarkers included in all component:
Clinical/biological interpretation difficult

ALTERNATIVE METHODS

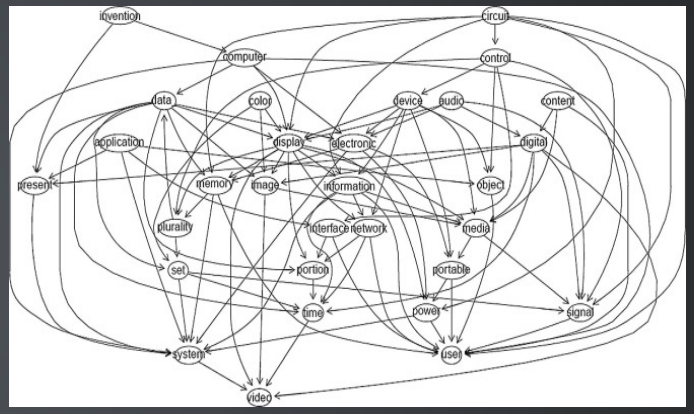
1. Dimension reduction
2. Causal approach
3. Machine learning algorithms

ALTERNATIVE METHODS

CAUSAL APPROACH

Modeling the link between variables:

- Can be based on prior assumptions on variables relations, but can become quickly complex!
- Without prior assumptions on variables relations: Very intensive/long computation (e.g. PC algorithm)



Very useful explicative model!

But no improvement regarding to the penalized regression for the prediction (using the dataset of the application 1, Ternes 2014)

ALTERNATIVE METHODS

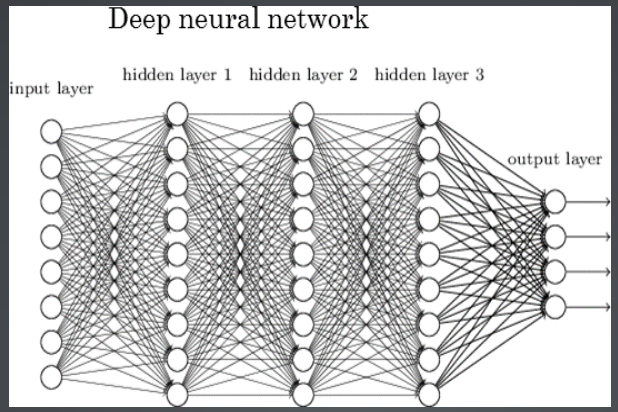
1. Dimension reduction
2. Causal approach
3. Machine learning algorithms

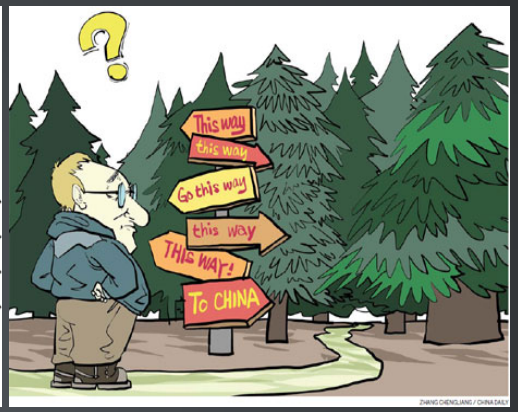
ALTERNATIVE METHODS

MACHINE LEARNING ALGORITHMS

Natural vocabulary:

- **(deep) neural network**
- **random forest**
- **SVM**
- **...**





Complex models: take into account non-linear relationships, interactions,(...)
Algorithms with excellent predictive accuracy, if representative/unbiased training set!
+ require a large database for the training set! (thousands/millions observations)
-> Suitable for big data (imaging or social network) analysis but not for small sample size high dimensional data (omics, for the moment...)

BUT...

- Interpretation of the contribution of the explanatory variables by human being impossible
- Overfitting risk!

PLAN

1. Introduction
2. Model calibration
3. Cross-validation
4. High dimensional data analysis issues
5. Penalized regression
6. Alternative methods
7. Conclusions

CONCLUSIONS

Predictive score very useful, to plan treatment for a patient for example

New methodological problems arise with the omics

High dimension for small sample size prevent tu use standard approach

Complex methods have good prediction accuracy, but medical research required also comprehensive method to validate biological mechanism assumption

Penalized regression = compromise between prediction accuracy, interpretability and mathematical complexity

CONCLUSIONS

(+)

- Good predictions
- Simple interpretation (LASSO & EN: variable selection)
- Parametric approach: more stable than non-Parametric

(-)

- Choice of the penalty
- Risk of false negative high for correlated variables (especially using LASSO)
- Parametric approach: modelling assumptions may reduce the predictive accuracy if missspecified (interactions? Non-linear effects?...)

CONCLUSIONS

Selection using penalized regression = very active research field: (De Bin 2014, De Bin 2016))

- Adaptive LASSO
- Group LASSO
- Variable interactions
- ...

REFERENCES

Reference of data applications:

1. Ternès N, Arnedos M, Koscielny S, Michiels S, Lanoy E. Statistical methods applied to omics data: predicting response to neoadjuvant therapy in breast cancer. Curr Opin Oncol. 2014 Nov;26(6):576-83
2. Ternès N, Rotolo F, Michiels S. biospear: an R package for biomarker selection in penalized Cox regression. Bioinformatics 2017

To go further:

- De Bin R, Sauerbrei W, Boulesteix AL. Investigating the prediction ability of survival models based on both clinical and omics data: two case studies. Stat Med. 2014 30;33(30):5310-29
- De Bin, R. Boosting in Cox regression: a comparison between the likelihood-based and the model-based approaches with focus on the R-packages CoxBoost and mboost. Computational Statistics 2016 31, 513–531