SummingUp ✓ 23.05.2018

~~Saturday, 21 April 2018 13:45~~

TMA4268 STATISTICAL LEARNING

fit → TRAINING SET $\{(x_1,y_1),\dots,(x_n,y_n)\}$ (independent pairs)

assess → TEST SET: same format

model selection → VALIDATION

VS. MODEL ASSESMENT

LINEAR REGRESSION (M3)

classical normal regression model
$$\begin{cases} \underset{n\times1}{Y} = \underset{n\times(p+1)}{X}\underset{(p+1)\times1}{\beta} + \underset{n\times1}{\varepsilon} \\ \varepsilon \sim N_n(0,\sigma^2 I)\end{cases}$$

X has rank $(p+1)$ = full rank ($n >> p+1$)

$\hat\beta = (X^TX)^{-1}X^TY$ ← minimize RSS

$\hat\beta \sim N_{(p+1)}(\beta, \sigma^2(X^TX)^{-1})$

$\hat\sigma^2 = \frac{1}{n-p-1}\cdot RSS$

$TSS = \sum_{i=1}^n (y_i-\bar y)^2$, $\bar y = \frac1n\sum_{i=1}^n y_i$ total sum of squares

$RSS = \sum_{i=1}^n (y_i-\hat y_i)^2$ residual sum of squares

$\hat y_i = x_i^T\hat\beta$

$R^2 = \frac{TSS-RSS}{TSS} = 1-\frac{RSS}{TSS}$ coeff. of determination, prop. of variance explained by regression

c_{jj} is jth diagonal element of $(X^TX)^{-1}$.

Inference for one β_j: based on $T_j = \frac{\hat\beta_j-\beta_j}{\hat\sigma\sqrt{c_{jj}}} \sim t_{n-p-1}$

$(1-\alpha)100\%$ Confidence interval for β_j: $[\hat\beta_j \pm t_{\frac\alpha2,n-p-1}\sqrt{c_{jj}}\,\hat\sigma]$

Prediction interval for Y_0 at x_0: $[x_0^T\hat\beta \pm t_{\frac\alpha2,n-p-1}\hat\sigma\sqrt{1+x_0^T(X^TX)^{-1}x_0}]$

Hypothesis test:
$H_0: \beta_j=0$ vs $H_1: \beta_j\neq0$
Reject H_0 when $T_0 = \frac{\hat\beta_j-0}{\hat\sigma\sqrt{c_{jj}}}$

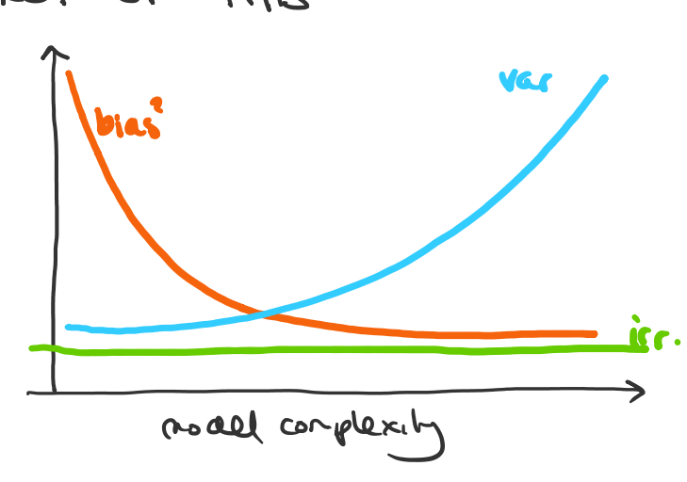
Is the regression significant?
$H_0: \beta_1=\beta_2=\dots=\beta_p=0$ vs H_1: at least one different
$F = \frac{(TSS-RSS)/p}{RSS/(n-p-1)} \sim F_{p,n-p-1}$ under H_0
P-value: $P(F > f_{obs})$, f_{obs} observed F. ($F_{p,n-p-1}$)

Residuals: $e_i = y_i-\hat y_i$ and scaled versions thereof → plot: $\hat y_i$ vs e_i → random, no trend
plot: qq plot e_i for normality → straight line

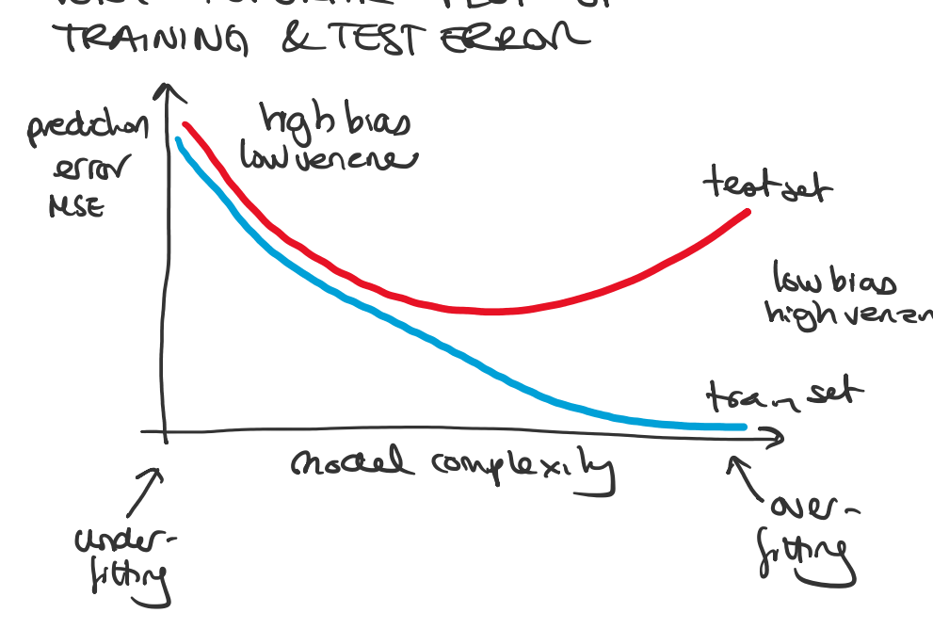
[M2+3] BIAS–VARIANCE trade-off in regression setting: $Y = f(x)+\varepsilon$ ($f(x)$ from)

Expected test mean squared error at x_0:
$$E[(Y-\hat f(x_0))^2] = \dots = \underbrace{Var(\varepsilon)}_{\text{irreducible error}} + \underbrace{Var[\hat f(x_0)]}_{\text{variance of prediction at }x_0} + \underbrace{Bias(\hat f(x_0))^2}_{\text{bias of prediction at }x_0}$$
$(f(x_0)-\hat f(x_0))^2$

PLOT OF THIS!



VERY POPULAR PLOT OF TRAINING & TEST ERROR



MODEL SELECTION (M6)

$R^2_{adj} = 1-\frac{RSS/(n-p-1)}{TSS/(n-1)}$

$AIC = -2\,\text{loglikelihood} + 2\,(\text{number of param.})$ ($p+1$)

$BIC = -2\,\text{loglikelihood} + \log(n)\cdot$ — " —

penalizing of training error

} can be used in model selection using "only" the training data
- best subset selection: all 2^p models
- stepwise methods ← when p is large ($1+p(p+1)/2$ models)

When model selection is done on test set or with CV, then RSS, R^2, MSE (unpenalized) can be used.

RSS will always decrease (or stay unchanged) if new covariates are added to the regression (even if they are just noise – why?), and R^2 will always increase (or stay unchanged). Therefore R^2 can never be a criterion to choose between regr. models of different model complexity based on a training set alone

DIMENSIONALITY REDUCTION: Principal component regression PCA (M6+10)

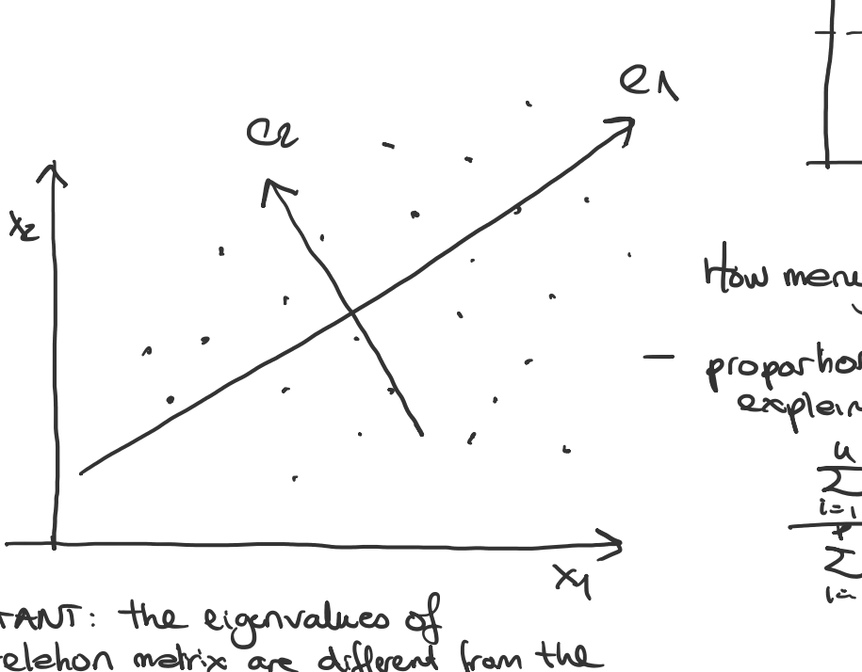
Have $\underset{n\times p}{X}$ with p large and/or many correlated covariates: want set of linear comb. of the X's that capture a large part of the variability – and where the sets are uncorrelated. → for interpretability, data reduction, visualization

Let Σ be covariance (or correlation) matrix of X, and (λ_i, e_i): $\Sigma = P\Lambda P^T$ eigenvalue/eigenvector pairs of Σ where $\lambda_1\geq\lambda_2\geq\dots\geq\lambda_p$. Then
$$[e_1 e_2 \cdots e_p]\begin{bmatrix}\lambda_1 & & 0\\ & \ddots & \\ 0 & & \lambda_p\end{bmatrix}\begin{bmatrix}e_1^T\\ e_2^T\\ \vdots \\ e_p^T\end{bmatrix}$$

construct $Z_i = e_i^T X$ (principal component scores; loadings; $p\times1$ random vector). Property: $Cov(Z) = \begin{bmatrix}\lambda_1 & 0 & 0\\ 0 & \ddots & \\ 0 & & \lambda_p\end{bmatrix}$ uncorrelated with $Var(Z_i)=\lambda_i$.

MLR can then be performed with Z as covariates instead of X. ⇒ PCR

but remember, the PC's are not selected based on Y only on X, so we assume variability in X is related to variability in Y



IMPORTANT: the eigenvalues of the correlation matrix are different from the eigenvalues of the covariance matrix – what you want depends on the situation – but mostly corr. is used.

How many PC's?
– proportion of total variance explained with k PC's
$\frac{\sum_{i=1}^k\lambda_i}{\sum_{i=1}^p\lambda_i}$ should be ~95%?

Loss function:
regression { Quadratic: $\sum_{i=1}^n (y_i-f(x_i))^2$
classification { 0/1: $y_i = f(x_i)$ → 0 loss; $y_i\neq f(x_i)$ → 1 loss

minimize 0/1 loss

Bayes classifier: classify to the class with the highest probability $P(Y=k\mid X=x)$ – and in simulation experiments this is known.
Bayes class boundary: boundary of Bayes classifier.
Bayes error = error rate of Bayes classifier

MODEL REGULARIZATION (M6)

Motivation: LS regression gives unbiased $\hat\beta$, but variance might be large, e.g. when covariates are correlated → $(X^TX)^{-1}\sigma^2$ (or p is large compared to n). If we instead look at biased (shrunken) $\hat\beta$ we might get smaller $Var(\hat\beta)+(\hat\beta-\beta)^2$? (brutto variance)

Ridge (L2) regression:
minimize$_\beta$ $RSS + \lambda\sum_{j=1}^p\beta_j^2$
not scale invariant – useful to scale covariates before using ridge; β_0 not penalized

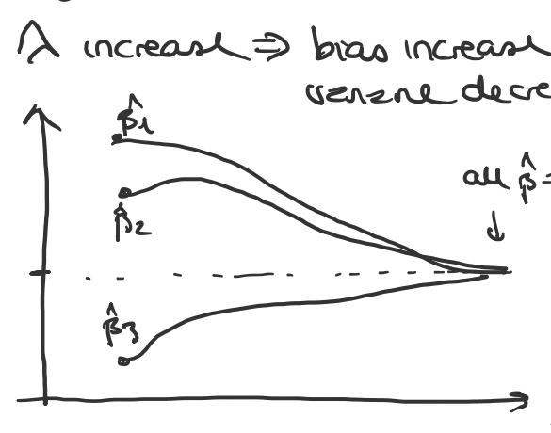
This also reduces the effective number of parameters

λ: tuning parameter ← decided by CV ← value $\hat\lambda$ giving min MSE

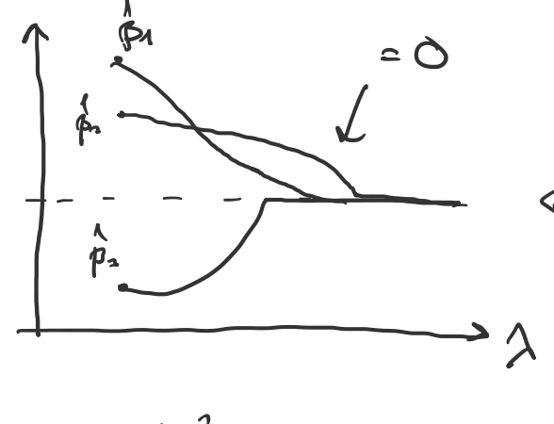
λ increase ⇒ bias increase, variance decrease

Interpretation of plots:

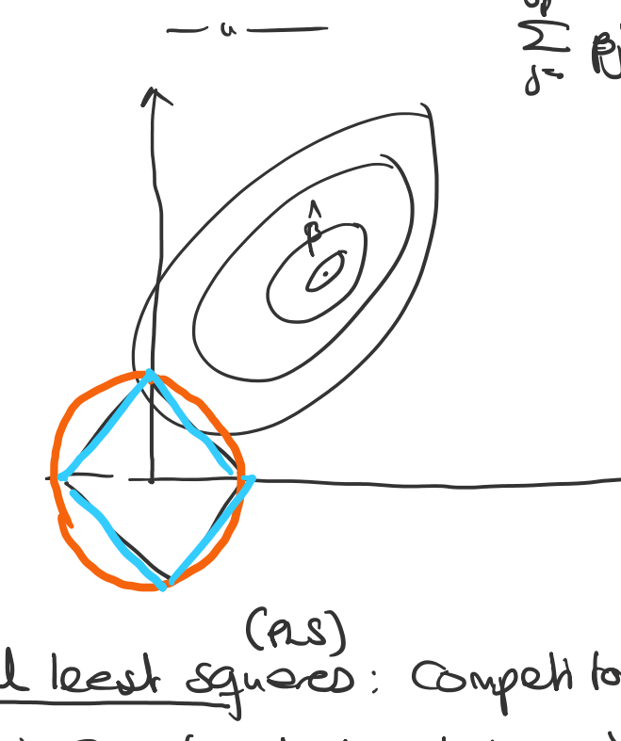
ridge:



Lasso (L1) regression: ← automatically performs model selection
minimize$_\beta$ $RSS + \lambda\sum_{j=1}^p|\beta_j|$



Dual problem
min RSS subj. to $\sum_{j=1}^p|\beta_j|\leq s$ lasso
— " — $\sum_{j=1}^p\beta_j^2\leq s$ ridge



Partial least squares (PLS): Competitor to PCR, but now Z is found also taking Y into account. More details in MA8701.

MOVING BEYOND LINEARITY (M7)

$Y = f(x)+\varepsilon$ and MLR $f(x)=X\beta$ → $\hat\beta = (X^TX)^{-1}X^TY$

X can be replaced by a set of basis functions $X^* = \begin{bmatrix}1 & b_1(x_1) & b_2(x_1) & \cdots & b_k(x_1)\\ 1 & \vdots & \vdots & & \vdots\\ 1 & b_1(x_n) & \cdots & \cdots & b_k(x_n)\end{bmatrix}$ ← but we only look at one x at a time first. $x_1 = x$ for obs 1, $x_n = x$ for obs n

So we still have linearity in β's – but now $f(x) = X^*\beta$ So $\hat\beta = (X^{*T}X^*)^{-1}X^{*T}Y$

1) POLYNOMIAL regression: $b_k(x) = x^k$

2) STEP FUNCTIONS: $b_k(x) = I(c_k\leq x < c_{k+1})$
similar to dummy variable coding, but now x might be continuous

3) REGRESSION SPLINES: combine polynomials & steps at knots
order-M spline: $b_j(x) = x^j$ $j=1,\dots,M-1$
$b_{M-1+k}(x) = (x-c_k)_+^{M-1}$, $k=1,\dots,K$
where knots $c_1,\dots,c_K$ are chosen (not estimated)
The resulting curve is smooth and deriv. up to order M−2 continuous (also in knots). Cubic spline popular.
NATURAL CUBIC spline = cubic spline that is linear at ends
Now bases as $b_1(x) = x$, $b_{k+2}(x) = d_k(x)-d_K(x)$ (small k, small k, capital K)
$d_k(x) = \frac{(x-c_k)_+^3-(x-c_{K+1})_+^3}{c_{K+1}-c_k}$ (small k, capital)

4) SMOOTHING SPLINE ← penalization problem (similar to ridge & lasso). g(x) smoothing spline
$$\min \sum_{i=1}^n(y_i-g(x_i))^2 + \lambda\int g''(t)^2dt$$
tuning; penalize variation in g; how much g turns
$\lambda=0$ ginterpolates
It turns out (MA8701) that a smoothing spline is a natural cubic spline with knots at the unique values of x. → lovely maths with smoother matrix! (beyond our scope in this course)

λ can be chosen by CV, or some version of AIC. Then effective number of parameters is calculated from the trace of the smoother matrix. (beyond our scope)

5) Local regression: smoothed version of KNN regression
minimize$_\beta$ $\sum_{i=1}^n K_{i0}(y_i-\beta_0-\beta_1x_i-\beta_2x_i^2)^2$
close observation higher weight than far apart; different versions

6) Additive model: put 1-5 all together
$X = [1\ X_1\ X_2\ X_3]$
choose for x_1 maybe poly?; choice for x_2, maybe natural cubic spline; choice for x_3, maybe smoothing spline? new x

EXTENSION: this additive model can be used on $p_i = \frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)}$ to generalize to logistic regression and make that class boundaries non-linear in X-space.

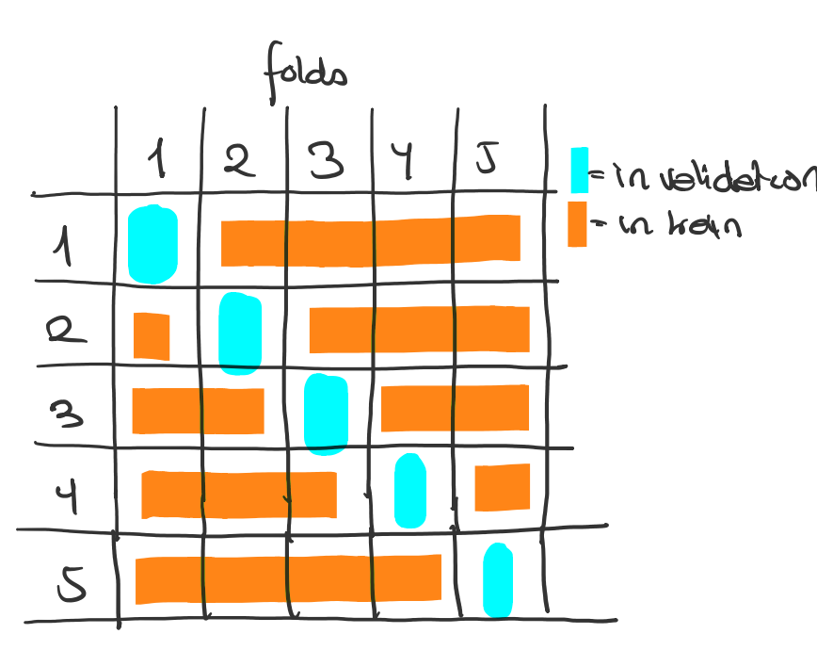
RESAMPLING METHODS (M5)

DATA rich situation (sometimes, e.g. when we generate data ourselves)

TRAIN	VALIDATE	TEST
↑ fit model	↑ model selection (complexity)	↑ model assesment

DATA not so rich = common situation:
→ Validation set approach ($V\uparrow B\uparrow$)
→ Leave one out cross validation LOOCV ($V\uparrow$, $B\downarrow$ due to corr.)
→ k-fold cross validation (CV)
• biased estimate (B) (as compared to using the whole ds. to train) • high/low variance (V) • randomness • comp. issues: expensive
pros and cons of the three methods ⇓ compromise 5 or 10-fold CV

Assess with $MSE_i = (y_i-\hat y_i)^2$ for each obs ← regression
or 0/1: $I(y_i\neq\hat y_i)$ ← classification
Sum over folds and average.



$CV = \frac1n\sum_{j=1}^k MSE_j\cdot n_j$ (regr)
$MSE_j = \frac1{n_j}\sum_{i\in j\text{th fold}}(y_i-\hat y_i)^2$ ↑ prediction when trained without jth fold

How to PERFORM CV: ALL PARTS of model fitting must be inside CV loop unless → selection bias. Eg: cancer predicted perfectly from 5 genes example

More RESAMPLING (M5)

BOOTSTRAPPING: data set of n observations. Estimate $\hat f$ by $\frac1n$.
Draw random sample from $\hat f$ → draw with replacement from data → one bootstrap sample
The probability that x_i is in bootstrap sample is $1-(1-\frac1n)^n \xrightarrow[n\to\infty]{} 0.632$
We may estimate $SD(\hat\beta_j)$ from regression – using bootstrapping. CI's in TMA4300 Comp. stat.
$\widehat{SD}(\hat\beta_j) = \sqrt{\frac1{B-1}\sum_{b=1}^B(\hat\beta_b-\frac1B\sum\hat\beta_b)^2}$ B = total no. of bootstrap samples

CLASSIFICATION (M4)

(Y_i, x_i) with $Y_i\in\{1,\dots,K\}$ or $Y_i=\{0,1\}$ or $\{-1,1\}$ (as in M9)

$p_k(x_0) = P(Y=k\mid X=x_0)$ posterior class probability

$\pi_k = P(Y=k)$ prior class probability

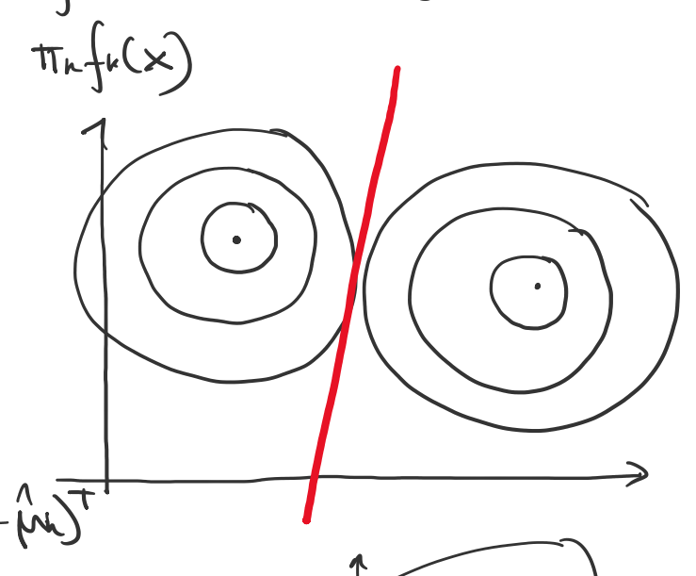
$f_k(x) = $ "$P(X=x\mid Y=k)$" is pdf of x in class k, class conditional distribution

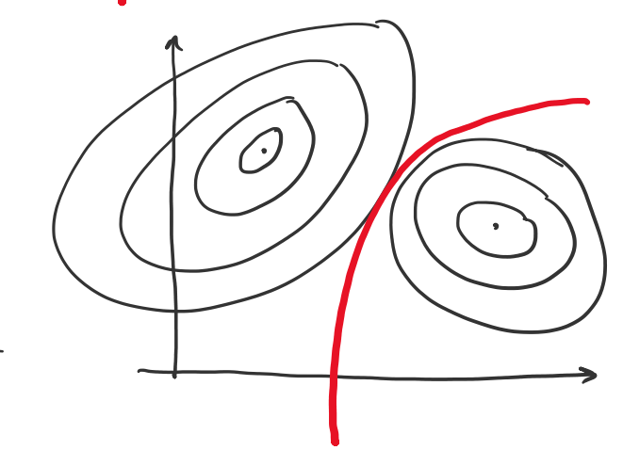
diagnostic paradigm = directly estimate $P(Y=k\mid X=x)$

Sampling paradigm = estimate π_k and $f_k(x)$ and classify to largest $\pi_kf_k(x)$

● Sampling paradigm: LDA & QDA
LDA: $f_k(x)$ is $N(\mu_k,\Sigma)$. $\hat\mu_k = \frac1{n_k}\sum_{i:\{y_i=k\}}x_i$
QDA: $f_k(x)$ is $N(\mu_k,\Sigma_k)$
$\hat\Sigma_k = \frac1{n_k-1}\sum_{i:\{y_i=k\}}(x_i-\hat\mu_k)(x_i-\hat\mu_k)^T$
$\hat\pi_k = \frac{n_k}{n}$
$\hat\Sigma = \sum_{k=1}^K\frac{n_k-1}{n-K}\cdot\hat\Sigma_k$

$\pi_k\cdot f_k(x) = \pi_j f_j(x)$ → linear class boundaries for LDA, quadratic QDA





● Diagnostic paradigm: KNN & logistic regression (trees & SVM too)

KNN: $\hat P(Y=j\mid X=x_0) = \frac1K\sum_{i\in N_0}I(y_i=j)$, N_0 = neighborhood of x_0

Logistic regression $(k=2)$: $Y_i = \begin{cases}0 & \text{with prob. }(1-p_i)\\ 1 & p_i\end{cases}$
where $p_i = \frac{\exp\{x_i^T\beta\}}{1+\exp\{x_i^T\beta\}}$ ← logistic function aka sigmoid
$x_i = \begin{bmatrix}1\\x_{i1}\\ \vdots\\ x_{ip}\end{bmatrix}$ $X = \begin{bmatrix}x_1^T\\ \vdots\\ x_n^T\end{bmatrix}$

Prediction at x_0: $\hat p(x_0) = \frac{\exp(x_0^T\hat\beta)}{1+\exp(x_0^T\hat\beta)}$

Parameter estimates found by maximum likelihood (TMA4315 GLM) not on closed form.

odds $= \frac{p_i}{1-p_i}$ so $p_i=\frac12$ gives odds 1, and is relevant because:
$\frac{P(Y_i=1\mid x_{i1}+1)}{P(Y_i=0\mid x_{i1}+1)} = \frac{P(Y_i=1\mid x_{i1})}{P(Y_i=0\mid x_{i1})}\cdot\exp(\beta_1)$

If x_1 increase to x_1+1, but all other x's are kept fixed then the odds is multiplied by $\exp(\beta_1)$.

Class boundary is linear in x-space, because $\log\left(\frac{p(x)}{1-p(x)}\right) = x^T\beta$ and set a cut-off on $p(x)$ (e.g. 0.5) and solve: $0 = x^T\beta = \beta_0+\beta_1x_1+\dots+\beta_px_p$ is the eq. for a hyperplane.

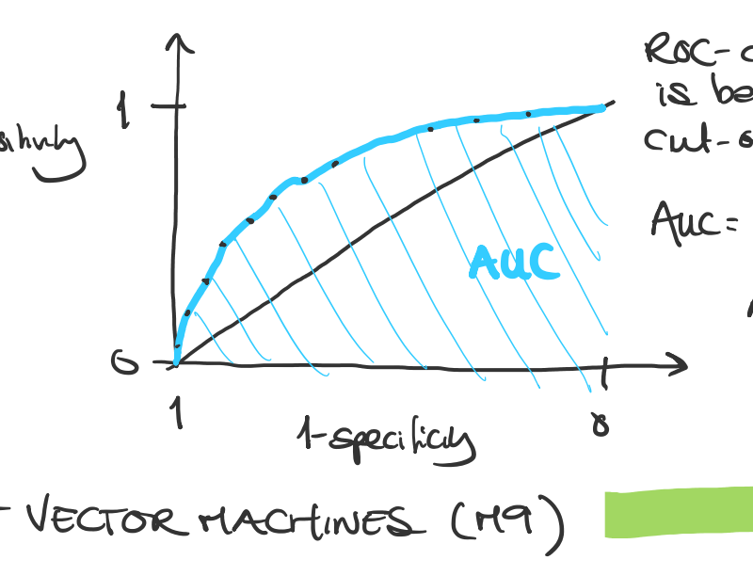
Assessment of classification → given a cut off on p (maybe 0.5) (n observations; on test data)

Confusion matrix:

		predicted class 0	predicted class 1	
true class	0	TN	FP	N
	1	FN	TP	P

may report misclassification rate $1-\frac{(TN+TP)}{n}$
or accuracy = rate of correct classifications: $\frac{TN+TP}{n}$

Sensitivity = TP/(FN+TP)
Specificity = TN/(TN+FP)



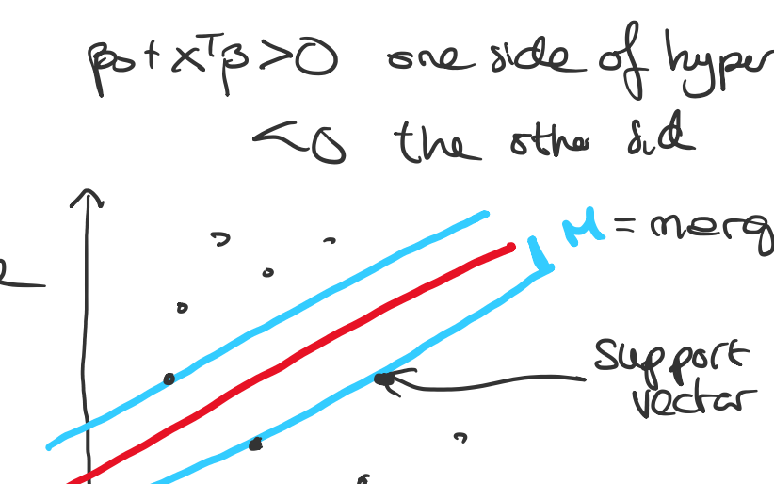
ROC-curve, each • is based on one cut-off on p.
AUC = area under ROC-curve
AUC = ½ = random guessing

SUPPORT VECTOR MACHINES (M9)

A method both for regression and classification, but we only consider classification, and two classes (coded as $\{-1,1\}$) } standardized x's are preferred!

Aim: find (high dimensional) hyperplane that separate (perfectly) the two classes
$\beta_0+x^T\beta=0$ with normalized βs $\sum_{j=1}^p\beta_j^2=1$
$\beta_0+x^T\beta>0$ one side of hyperplane
<0 the other side } $y_i\cdot(\beta_0+x_i^T\beta)>0$ if correct classified x_i

separable problem



Maximal margin classifier:
$\max_{\beta_0,\beta_1,\dots,M} M$ subject to $\sum_1^p\beta_j^2=1$
$y_i(\beta_0+x_i^T\beta)\geq M$ $\forall i=1,\dots,n$

CONNECTION SVM & logistic regression (ridge version)
SVM: $\min_\beta\left\{\underbrace{\sum_{i=1}^n\max(0,1-y_if(x_i))}_{\text{hinge loss}}+\lambda\underbrace{\sum_{j=1}^p\beta_j^2}_{\text{ridge penalty}}\right\}$
logistic regr. has $\log(1+\exp(-y_if(x_i)))$ which looks similar
$f(x_i)$ might contain nonlinear terms

result: the hyperplane only depends on the observations that lie on the margin or on the wrong side of the margin = "support vectors"

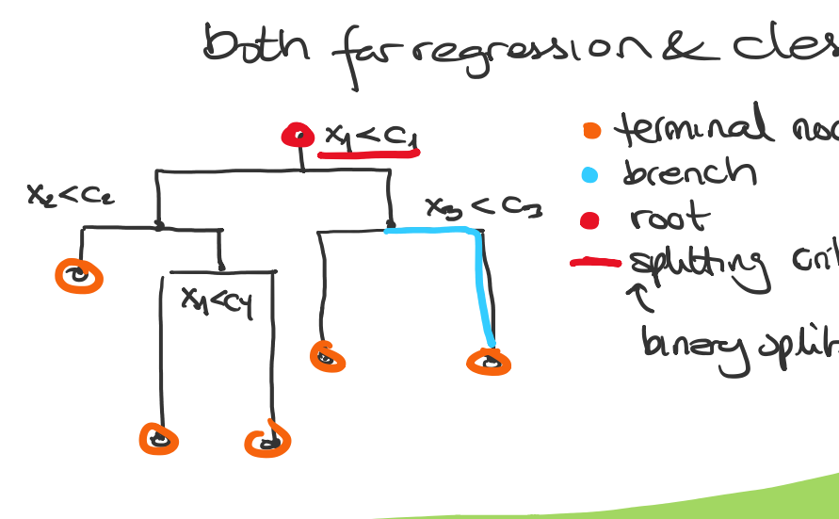
Support vector classifier: non-separable case – $\varepsilon_1,\dots,\varepsilon_n$: slack variables
$\max_{\beta_0,\beta_1,\dots,\beta_p,\varepsilon_1,\dots,\varepsilon_n,M} M$ subject to $\sum_1^p\beta_j^2=1$, $y_i(\beta_0+x_i^T\beta)\geq M(1-\varepsilon_i)\ \forall i$, $\varepsilon_i\geq0$, $\sum_1^n\varepsilon_i\leq C$
{ C is tuning parameter chosen by CV
→ Classification rule: $f(x^*) = \beta_0+x^{*T}\beta$ and if $\begin{cases}f(x^*)<0 & \text{set } y^*=-1\\ >0 & \text{set } y^*=1\end{cases}$

to understand skills in optimization is needed (Lagrange primal & dual problem)

Support vector machine: $f(x) = \beta_0+\sum_{i\in S}\alpha_iK(x,x_i)$
$x^T\beta$ replaced by more general nonlinear functions that are written as sums of kernels (generalization of inner products)
$\max_{\beta_0,\alpha_1,\dots,\alpha_n,\varepsilon_1,\dots,\varepsilon_n,M} M$, $y_i(f(x_i))\geq M(1-\varepsilon_i)$, $\varepsilon_i\geq0$, $\sum\varepsilon_i\leq C$, $\sum_1^p\beta_j^2=1$
where $f(x) = \beta_0+\sum_{i\in S}\alpha_iK(x,x_i)$
tuning: (C,γ) by CV
popular kernel: radial kernel: $K(x_i,x_i') = \exp\left(-\gamma\sum_{j=1}^p(x_{ij}-x_{i'j})^2\right)$

TREE-BASED METHODS (M8)

both for regression & classification – and we have nonlinearities & interactions between covariates!



1) Want to minimize an error criterion on the whole tree construction – but since computationally infeasible a greedy approach "recursive binary splitting" is used.
$R_1(j,s) = \{x\mid x_j<s\}$, $R_2(j,s) = \{x\mid x_j\geq s\}$: find predictor (j) and splitting point (s) that minimize "RSS"
Regression: $\underbrace{\sum_{i:x_i\in R_1}(y_i-\hat y_{R_1})^2}_{RSS(R_1)}+\underbrace{\sum_{i:x_i\in R_2}(y_i-\hat y_{R_2})^2}_{RSS(R_2)}$
$\hat y_{R_1}$ = mean of training resp. in R_1.
Classification: similar, but using impurity measure
$\hat p_{jk} = \frac1{N_j}\underbrace{\sum_{i:x_i\in R_j}I(y_i=k)}_{n_{jk}}$
• Gini index: In region R_j (leaf node) with N_j obs.
$G = \sum_{k=1}^K\hat p_{jk}(1-\hat p_{jk})$
• Cross entropy: $D = -\sum_{k=1}^K\hat p_{jk}\cdot\log\hat p_{jk} = -\sum_{k=1}^K\left(\frac{n_{jk}}{N_j}\cdot\log\left(\frac{n_{jk}}{N_j}\right)\right)$
• Deviance: $-2\sum_{k=1}^K n_{jk}\cdot\log(\hat p_{jk}) = -2\sum_{k=1}^K n_{jk}\cdot\log\left(\frac{n_{jk}}{N_j}\right)$ } gives the same tree

2) Continue splitting until we reach decided stopping criterion. Eg: reduction in RSS not big, less than 5 obs in a terminal node.

3) This full tree is used for prediction by dropping a new obs down the tree and predict as
a) regression: $\hat y = \frac1{N_j}\sum_{i:x_i\in R_j}y_i$ (R_j region (leaf)) ← y_i's in the training set
b) classification $\hat y$ = majority vote among all y_i's in R_j
or $\hat p_{kj} = \frac1{N_j}\sum_{i:x_i\in R_j}I(y_i=k)$ (n_{jk}) and classify to the class with the highest probability.

4) From a full tree to a pruned tree
Reduce the size of the tree by cutting branches.
Cost complexity pruning: for a complexity param α
minimize$_T$ $C_\alpha(T) = Q(T)+\alpha\cdot|T|$ (tree; cost function from the fitting (Gini, deviance, …); number of terminal nodes) α found by CV.

5) Evaluate model fit on a test set! Same criteria as regression (RSS) or classification (ROC, AUC, misclassification rate).

Trees meet bootstrapping ⇒ FORESTS ← why? More powerful prediction models – but loose in interpretation

A) BAGGING
Bootstrap aggregation (if indep. obs $\bar X_1,\dots,\bar X_n$)
Motivation: $Var(\bar X) = \frac{\sigma^2}{n}$ → we construct many trees and average them to define $\hat y$. The B trees we grow (full) from B bootstrap samples. $\hat f_{bag}(x) = \frac1B\sum_{b=1}^B\hat f^{*b}(x)$ (one tree!)

B) RANDOM FOREST – but not maximal effect of bagging because trees are not independent ⇒ we make them more uncorrelated by not allowing all p covariates to be chosen in each binary split.
Rule of thumb: $m=\sqrt p$ classification and $m=\frac p3$ for regression.

OOB = out-of-bag: for each bootstrap sample there are on average $\frac23$ of the observations in the sample (bag). The last $\frac13$ are used for prediction. We will then have $\approx\frac B3$ predictions for observation i – which we average.

Variable importance plots: – give the total amount of RSS (or Gini) decrease over splits of a predictor – averaged over all B trees

C) BOOSTING: many variants, but we only considered boosting for trees
Fit one tree – make residuals – fit a tree to the residuals → update
$\hat f(0)=0$; $\hat f^b(x)$ (with d splits)
$\hat f(x)\leftarrow\hat f(x)+\lambda\cdot\hat f^b(x)$
$r\leftarrow r-\lambda\hat f^b(x)$
$\hat f(x) = \sum_{b=1}^B\lambda\hat f^b(x)$
Tuning parameters: B = number of trees; λ = shrinkage parameter; d = number of tree splits

UNSUPERVISED LEARNING (M10)

Look for underlying structure or groupings in data – no Y only $\underset{n\times p}{X}$.

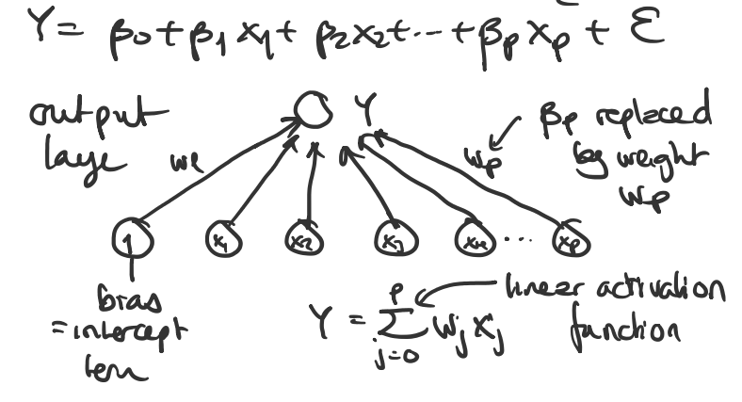
Principal component analysis (see M6)
PC loadings: interpret effect of each covariate on each component – average effect, diff between A and B etc
PC score: plot observations in PC1 & PC2, … can be used for quality control
PC scores → MLR = PCR (M6)

K-means clustering (non-overlapping clusters)
* Number of clusters given (or selected separately)
* Define cluster centroids → classify observations to clusters (closest)
* Recalculate centroids
Euclidean distance popular (but also (weighed) Mahlanobis)
Generalizations to medioids etc.

Hierarchical clustering
– works in a sequential way by connecting observations that are similar
* Similarity measures (dissimilarity): Euclidean, Correlation
* Linkage: how to calculate dissimilarity between groups of observations?
– single: minimum
– average: average
– complete: maximum
* Presented in a dendrogram – choose where to cut the dendrogram to get a number of clusters.

NEURAL NETWORKS (M11)

a) Possible to represent MLR & logistic regression as graph (one input and one output layer)
$Y = \beta_0+\beta_1x_1+\beta_2x_2+\dots+\beta_px_p+\varepsilon$



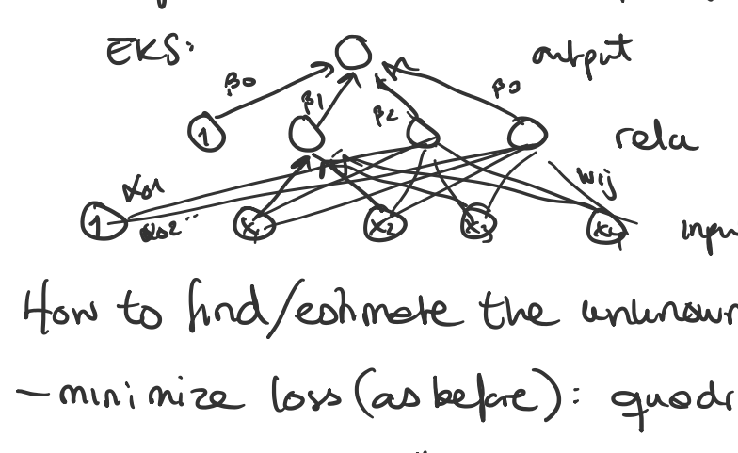
$Y = \sum_{j=0}^p w_jx_j$ ← linear activation function

$\log\left(\frac{p}{1-p}\right) = \beta_0+\beta_1x_1+\dots+\beta_px_p \iff p = \frac{\exp(\beta_0+\dots)}{1+\exp(\beta_0+\dots)} = \frac{\exp(a)}{1+\exp(a)}$
same graph, but now output layer has logistic activation function (sigmoid) used for two-class classification problems
for K classes: K outputs (dummy) and softmax activation $\frac{\exp(a_k)}{\sum_{k=1}^K\exp(a_k)}$ ensure sum to 1

technical issue: data represented as tensors (generalization of vector, matrix)
Does calc. in the network easy to perform, also for images & text.

b) Instead of M7 where we did nonlinear function of each covariate, we instead look at nonlinear function of sums of covariates pr. layer – and add many layers. Popular non-linear activation function is relu(a) = max(0,a). For each layer we specify number of nodes and choice of activation function ← feedforward network.

EKS:



$Y = \beta_0+\sum_{j=1}^3\max\left(\alpha_{0j}+\sum_{i=1}^4w_{ij}x_i, 0\right)\cdot\beta_j$ (these are often also called w's (but I wanted to avoid many indecies))
number of unknown parameters? 4 in hidden to output, 15 in input to hidden, 19; p=4 inputs + bias

keras in R: use to setup and call backend (Tensorflow, …) ← model level library
use "pipe" type notation to set up model architecture and send to estimation and then to evaluation

c) How to find/estimate the unknown weight (= parameters)?
– minimize loss (as before): quadratic loss for regression = "mse"
K=2 classes: "binary crossentropy" = as in tree and logistic regression: $-y_i\log(p_i)-(1-y_i)\log(1-p_i)$
K classes: "categorical crossentropy": $-\sum_{j=1}^K y_{ij}\log(p_{ij})$
– optimization: rmsprop, adam, … (adjust gradient)
gradient based: ladder method, with stoch. version with "batches" of observations (can use chain rule for differentiation)
popular: mini-batch stochastic gradient descent
epoch: one run through all observations

d) Evaluation of performance: "metrics"
mae = mean abs. error, accuracy = avg correct classif.

e) avoiding overfitting:
– smaller network or more data
– regularization (L1, L2, combine)
– drop-out
– early stopping
⊗ "unsolved": how do we assess the uncertainty in the network fitted?

f) many cool extensions
we have briefly looked at recurrent nets & convolution nets

g) data types popular: images, text that M1-10 not have solved.