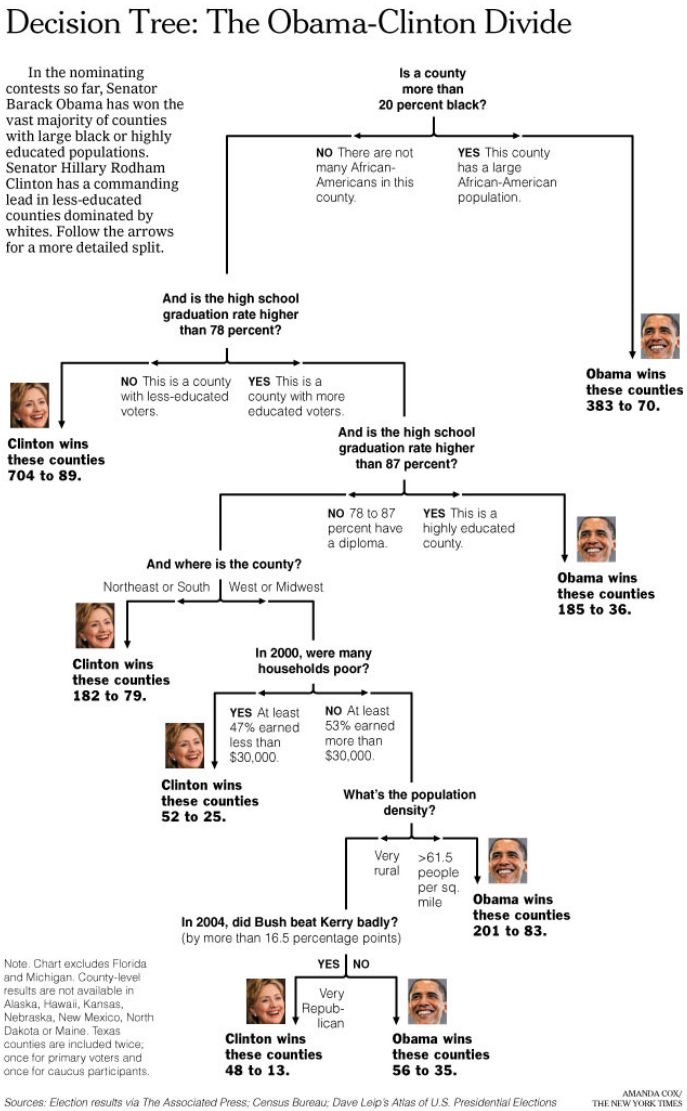
Chapter 6

Generalizations of regression

Regression trees

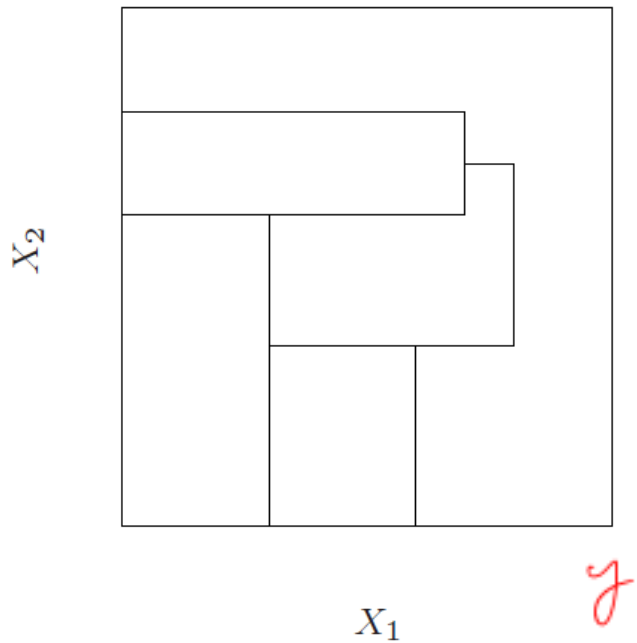


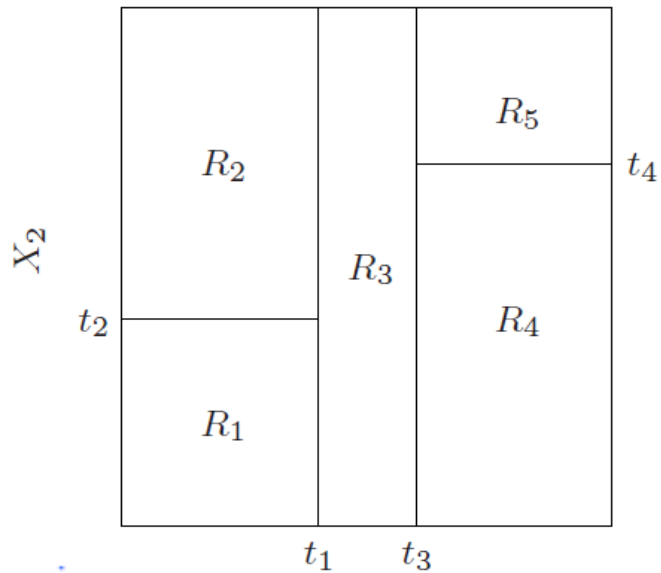
Note: a simple linear regression is too restrictive for large data sets.

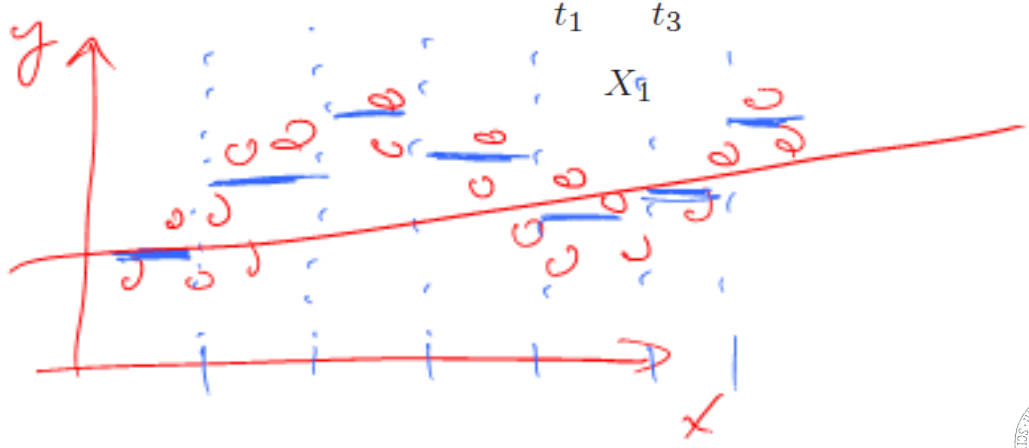
Regression trees offer a flexible technique with results, which are easy to interpret

General strategy:

- The values of the explanatory variables are split into P disjunct regions (rectangles) $R_1, \ldots, R_P$: binary splitting



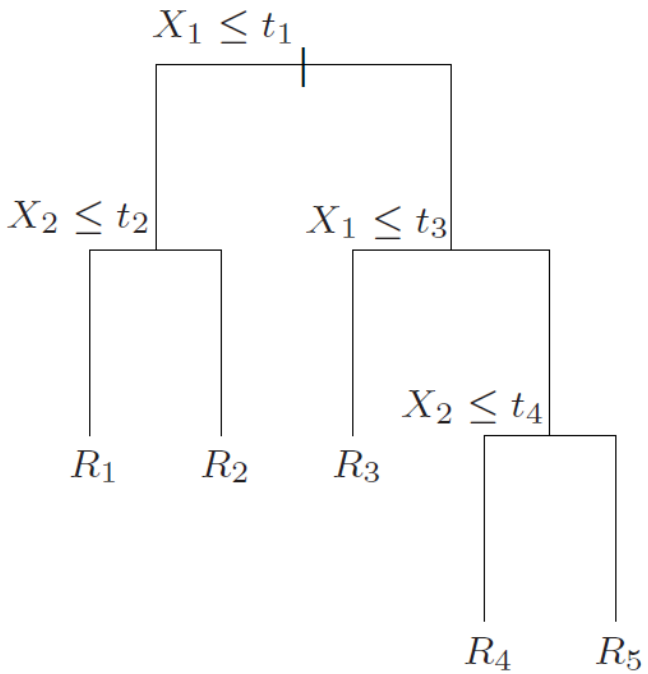


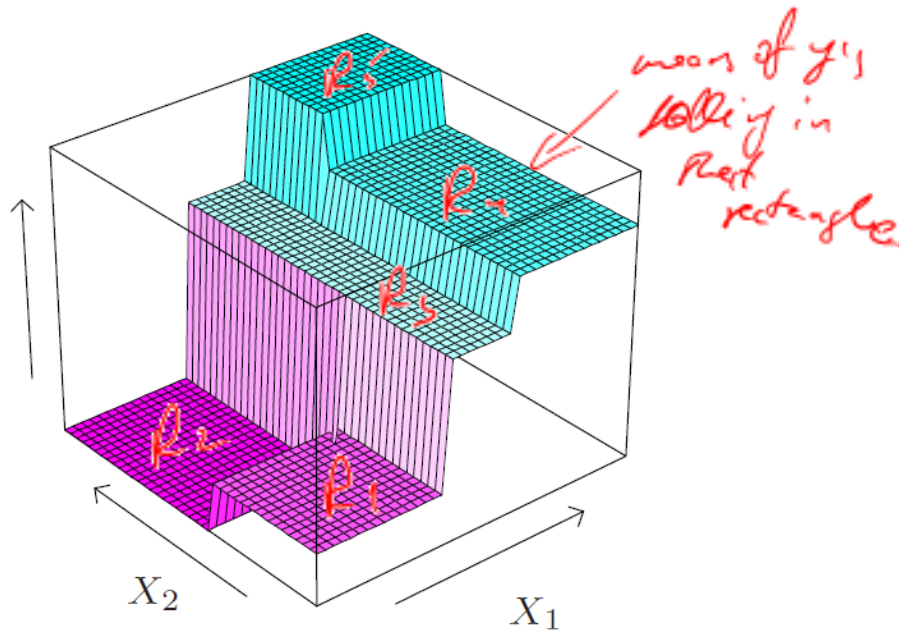


Source: Hastie et al. (2001)

- In each rectangle we fit a simple model

  e.g. a constant, i.e. the forecast in rectangle R_p is the mean of all Y-values falling into this rectangle.





Source: Hastie et al. (2001)

Question: how to determine the regions?

OLS method:

$$\sum_{i=1}^{n}(y_i - \boldsymbol{x}_i'\boldsymbol{b})^2 \longrightarrow min, \quad bzgl. \quad \boldsymbol{b}.$$

For regression trees:

$$\sum_{p=1}^{P}\sum_{i\in R_p}(y_i - \hat{y}_{R_p})^2 \longrightarrow min, \quad w.r.t. \quad R_1, \ldots, R_p,$$

where $\hat{y}_{R_p}$ is the mean of observations in the p-th rectangle.

Note: direct optimization is hardly possible $\rightsquigarrow$ recursive binary splitting

Step 1

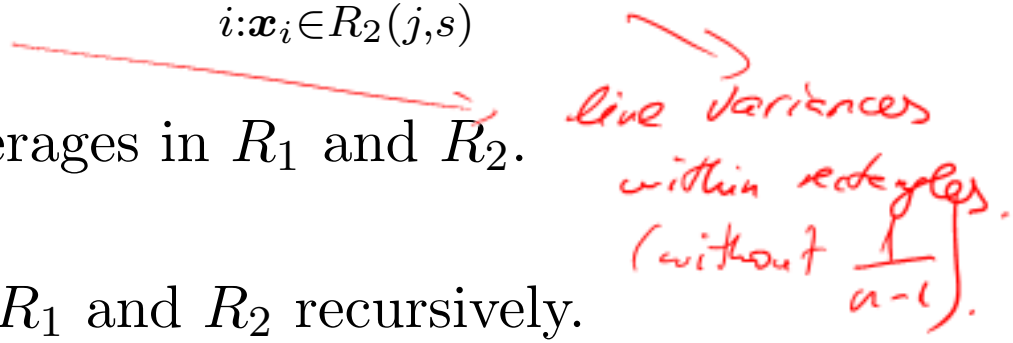
- Find the variable X_j and the splitting point s, which separates the space into two regions:

$$R_1(j,s) = \{\boldsymbol{X} | X_j \leq s\} \quad \text{and} \quad R_2(j,s) = \{\boldsymbol{X} | X_j > s\}.$$

- j and s are determined using the following objective function

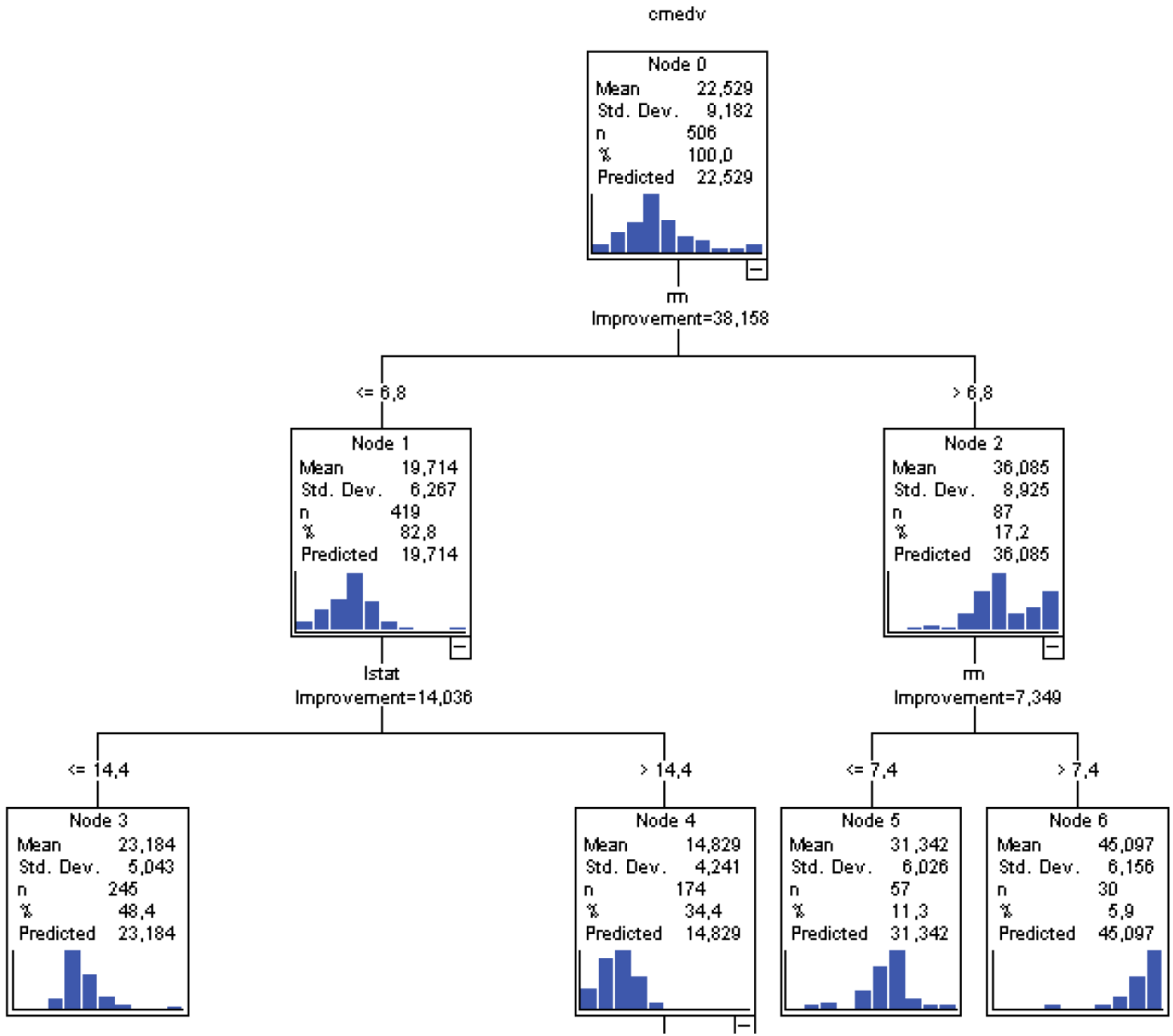
$$\sum_{i:\boldsymbol{x}_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\boldsymbol{x}_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2,$$

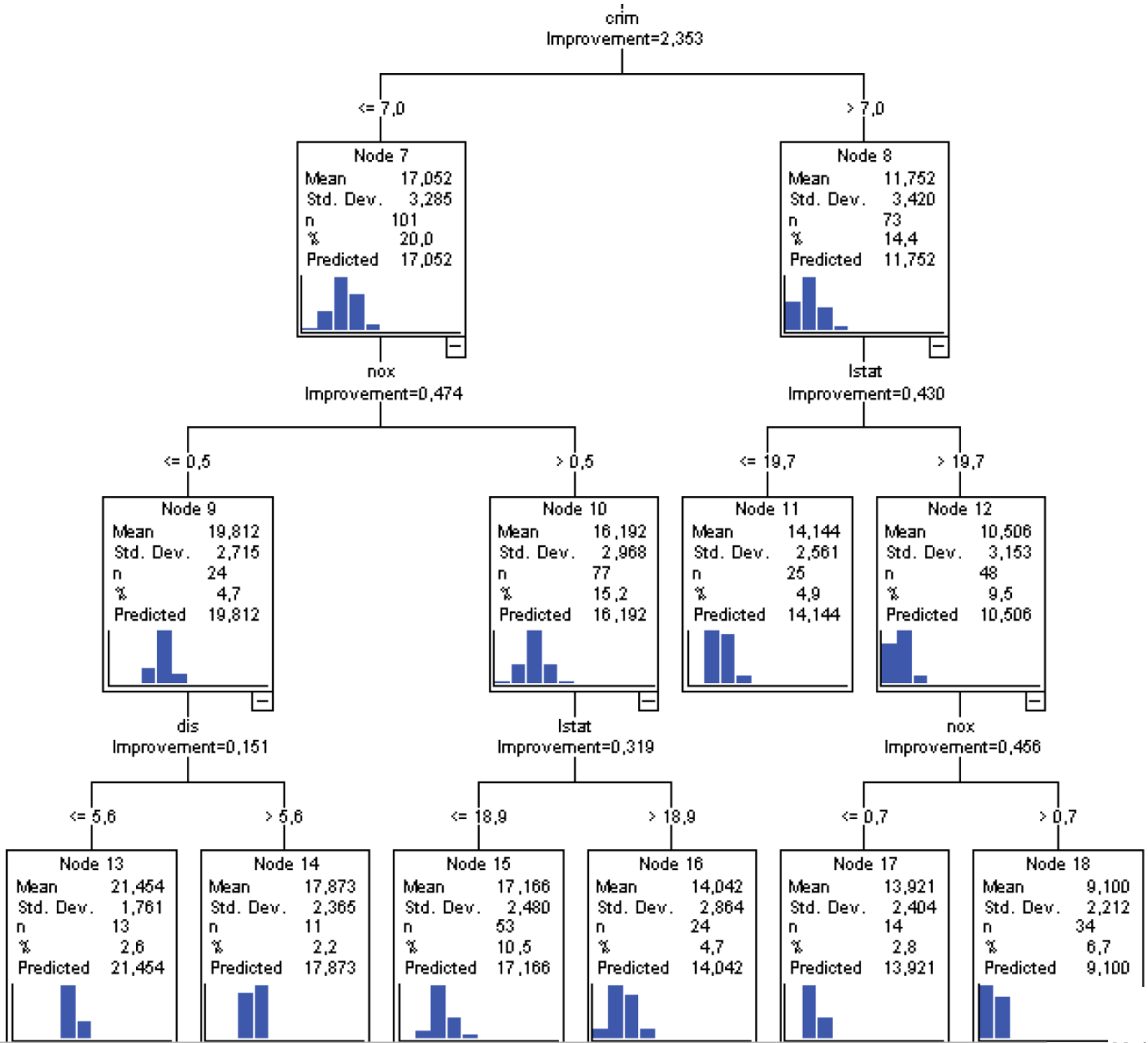
where $\hat{y}_{R_1}$ and $\hat{y}_{R_2}$ are averages in R_1 and R_2.



Step 2

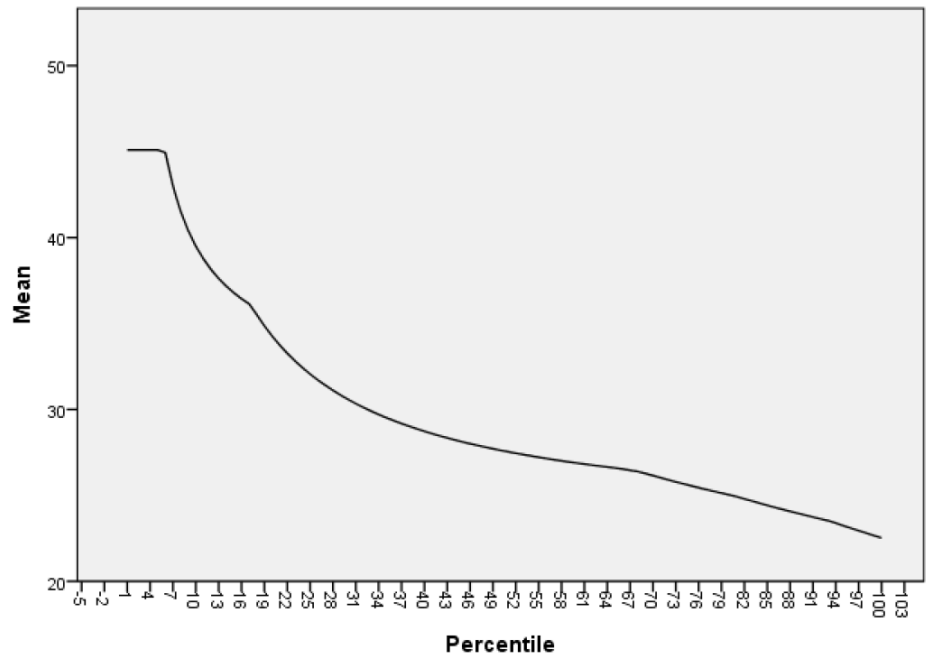
Repeat Step 1 to split regions R_1 and R_2 recursively.



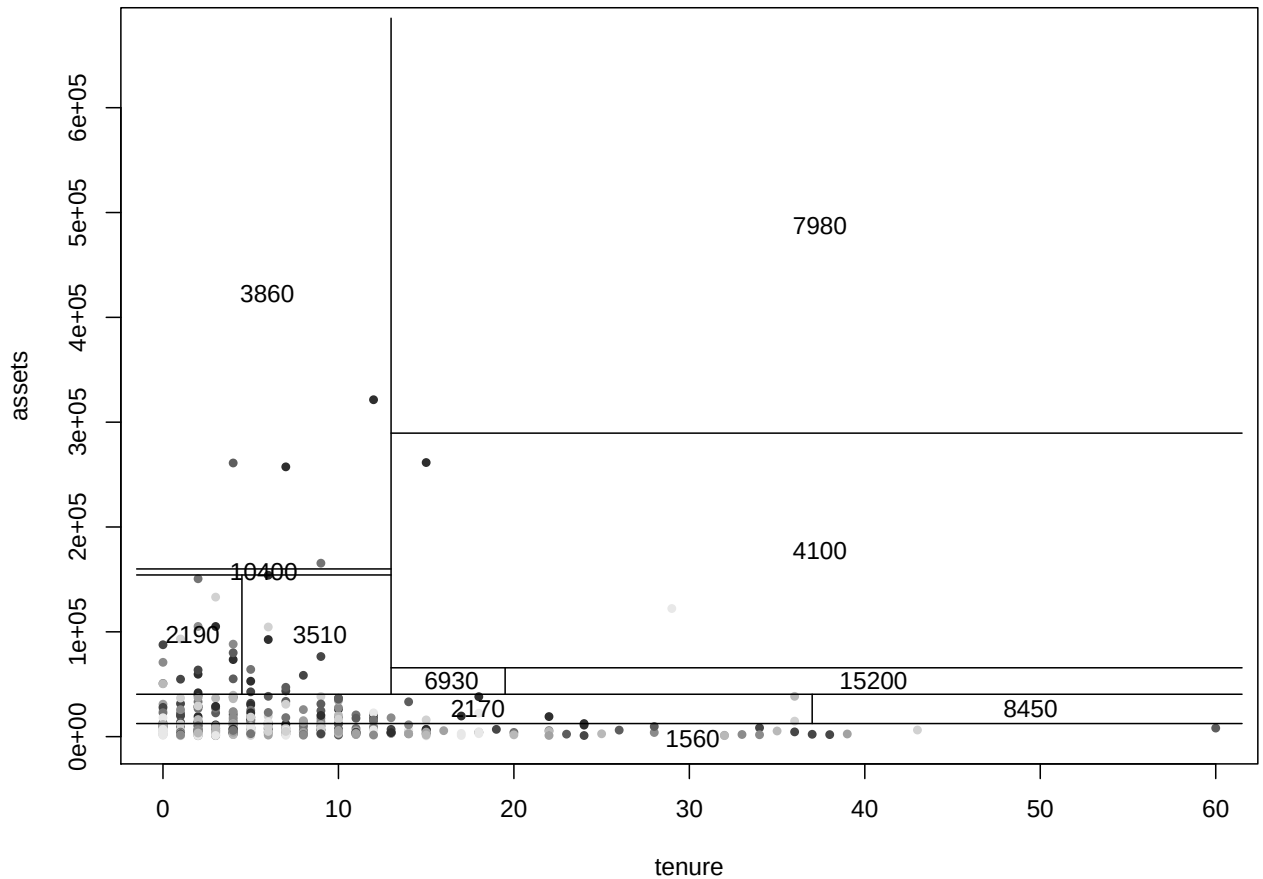


Gain Summary for Nodes

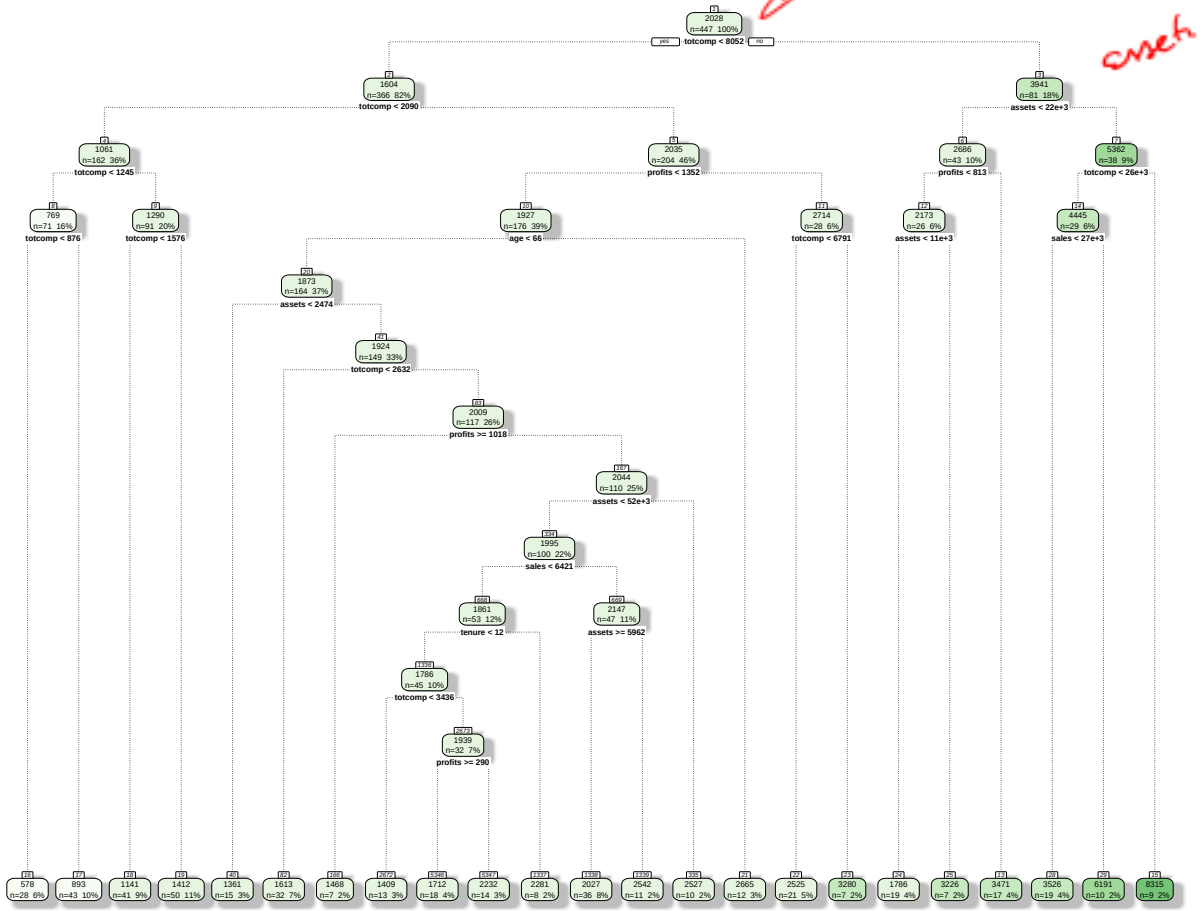
	Node-by-Node			Cumulative		
Node	N	Percent	Mean	N	Percent	Mean
6	30	5,9%	45,10	30	5,9%	45,10
5	57	11,3%	31,34	87	17,2%	36,09
3	245	48,4%	23,18	332	65,6%	26,56
13	13	2,6%	21,45	345	68,2%	26,37
14	11	2,2%	17,87	356	70,4%	26,11
15	53	10,5%	17,17	409	80,8%	24,95
11	25	4,9%	14,14	434	85,8%	24,33
16	24	4,7%	14,04	458	90,5%	23,79
17	14	2,8%	13,92	472	93,3%	23,50
18	34	6,7%	9,10	506	100,0%	22,53



```
> library("tree")
> tree.ceo = tree(salary ~ tenure + assets, data=ceo)
> plot(ceo$tenure,ceo$assets, type="p", pch=20, xlab="tenure", ylab="assets")
> partition.tree(tree.ceo, ordvars=c("tenure","assets"), add=TRUE)
```



```
> library("rpart")
> rpart.ceo = rpart(salary ~ .,data=ceo,control=rpart.control(cp = 0.001))
> fancyRpartPlot(rpart.ceo)
```



Rattle 2017−Nov−16 09:30:27 okhrinya

Note: Using CART we can grow the tree to saturation.

- Fix the maximal number of splittings and a lower bound for the number of observations per region.
- Fix the minimal change in the objective function.
- tree prunning: after the optimal tree is found, it is shortened

$$R_\alpha(T) = \frac{1}{\sum_i (y_i - \bar{y})^2} \sum_{m=1}^{|T|} \sum_{i:\boldsymbol{x}_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

where $|T|$ is the number of terminal nodes in a tree and α is the complexity parameter.

Key properties of CARTs

- For given α it is possible to determine the tree $T(\alpha)$ with the smallest $R_\alpha(T)$ uniquely
- If $\alpha > \beta$ then $T(\alpha) = T(\beta)$ or $T(\alpha)$ is a strict subtree of $T(\beta)$.

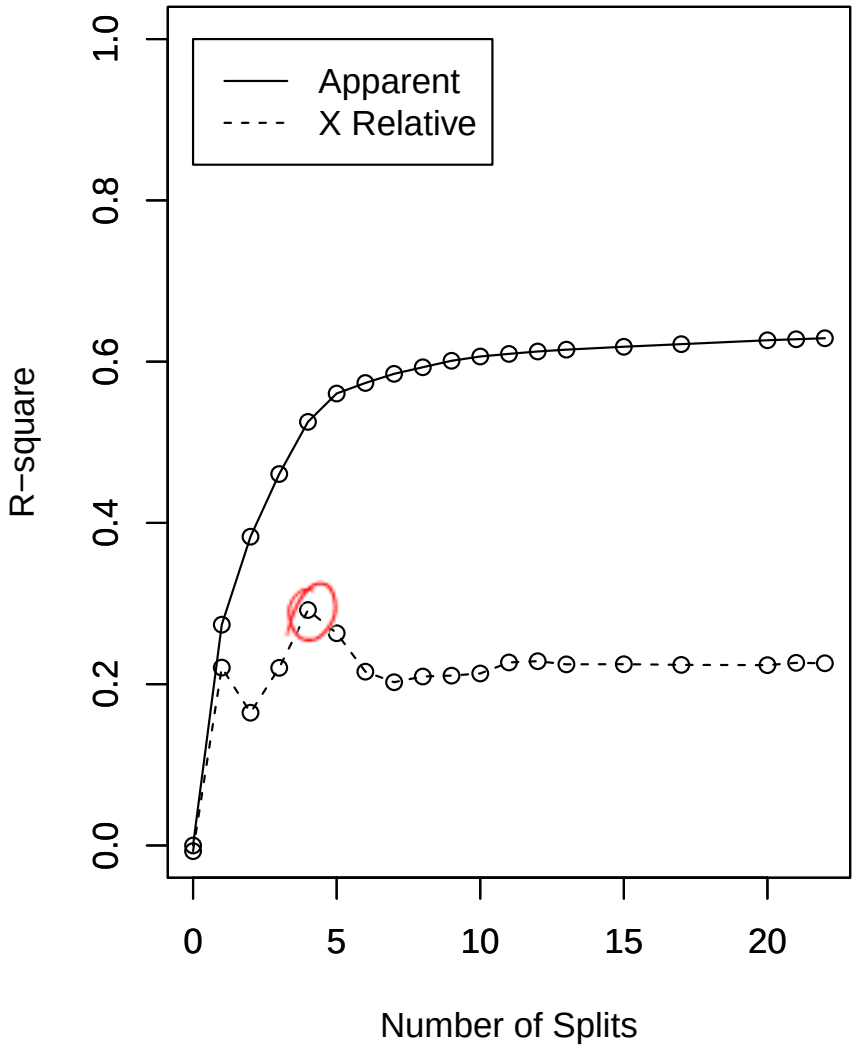
```
> printcp(rpart.ceo)
> printcp(rpart.ceo)

Regression tree:
rpart(formula = salary ~ ., data = ceo, control = rpart.control(cp = 0.001, xval = 10))
Variables actually used in tree construction:
[1] age     assets  profits sales   tenure  totcomp
Root node error: 1323386794/447 = 2960597

n= 447

          CP nsplit rel error  xerror     xstd
1  0.2738266      0   1.00000 1.00703 0.19979
2  0.1091070      1   0.72617 0.77938 0.13638
3  0.0777412      2   0.61707 0.83535 0.15451
4  0.0646524      3   0.53933 0.77968 0.15159
5  0.0351651      4   0.47467 0.70797 0.15182
6  0.0130789      5   0.43951 0.73676 0.17353
7  0.0113130      6   0.42643 0.78445 0.17517
8  0.0081763      7   0.41512 0.79755 0.17644
9  0.0080167      8   0.40694 0.79045 0.17654
10 0.0052976      9   0.39892 0.78938 0.17620
11 0.0032733     10   0.39363 0.78670 0.17590
12 0.0029769     11   0.39035 0.77301 0.17377
13 0.0022593     12   0.38737 0.77146 0.17367
14 0.0017931     13   0.38512 0.77539 0.17310
15 0.0016171     15   0.38153 0.77520 0.17313
16 0.0016099     17   0.37829 0.77603 0.17313
17 0.0012678     20   0.37346 0.77635 0.17313
18 0.0012449     21   0.37220 0.77348 0.17303
19 0.0010000     22   0.37095 0.77403 0.17301
```

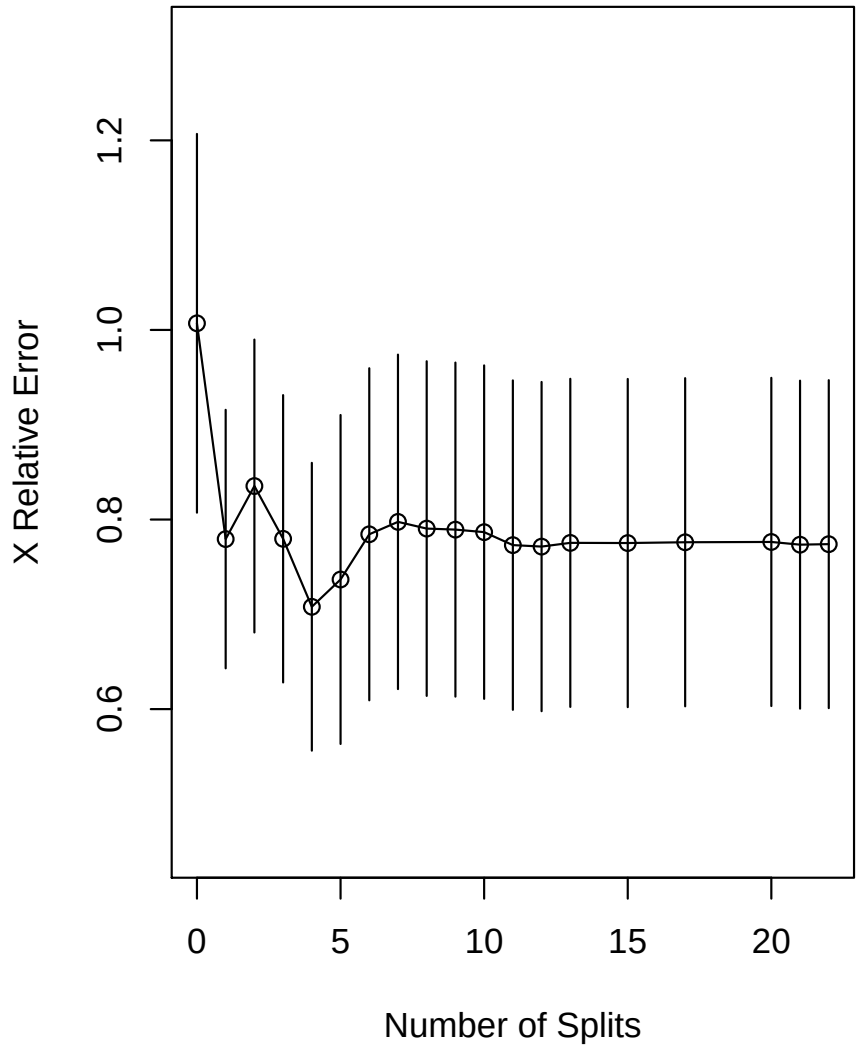
Q: How to choose the overall optimal α or subtree? $\rightsquigarrow$ cross-validation

- The sequence of trees T_0 (no splits) to T_m (m splits) uniquely determines the sequence of possible α's

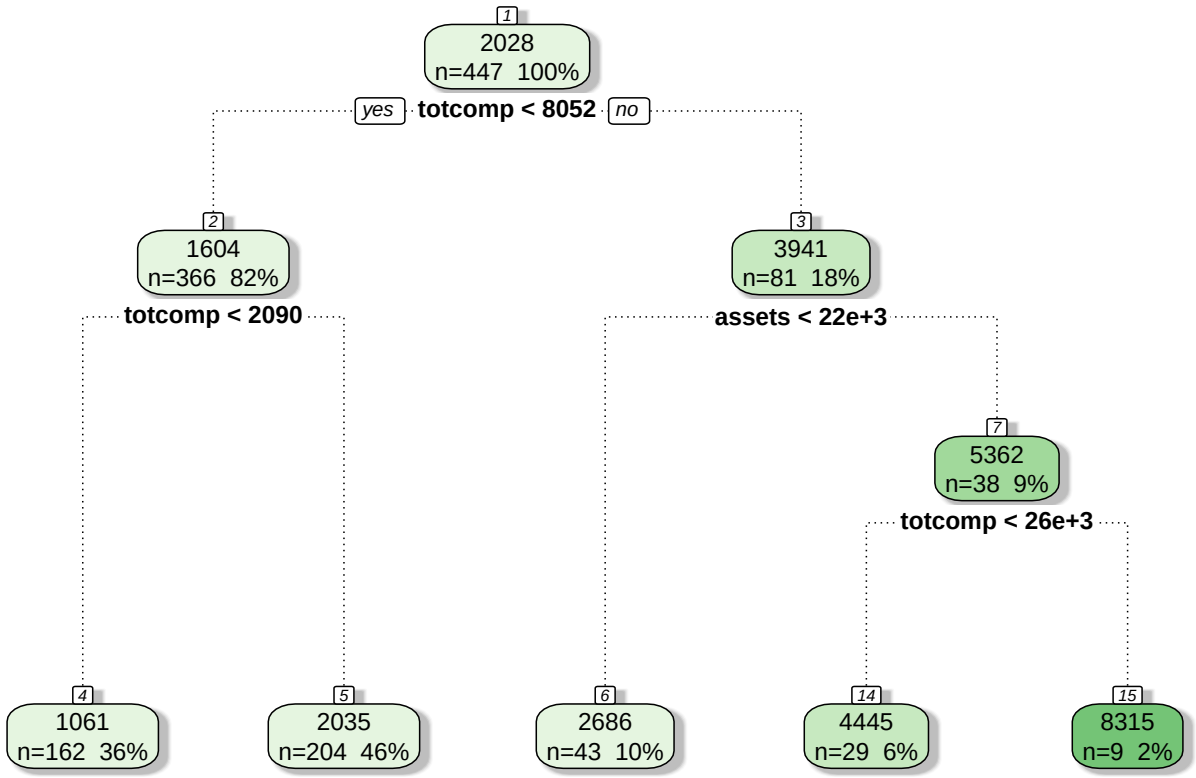
$$\infty, \alpha_1, \ldots, \alpha_{m-1}, \alpha_{min}$$

- Any α between $(\alpha_i, \alpha_{i+1}]$ leads to the same optimal subtree
- Define $\beta_i = \sqrt{\alpha_i \alpha_{i+1}}$ as an "average" CP for every interval
- Split the data into B subsets $G_1, \ldots, G_B$ (10 by default)
  - For every subset excluding the G_i's determine $T_{\beta_1}, \ldots T_{\beta_m}$
  - Compute the relative MSE as the forecast loss for elements in G_i
- Compute the average loss over all G_i's and choose β (and thus the optimal subtree) which corresponds to the smallest one.





```
> cp.min = which.min(rpart.ceo$cptable[,4]);
> rpart.ceo.prune=prune(rpart.ceo, cp=rpart.ceo$cptable[cp.min,1])
> rpart.ceo.prune$variable.importance/sum(rpart.ceo.prune$variable.importance)
   totcomp     assets      sales    profits     tenure        age
0.52984007 0.18398873 0.10229360 0.09962347 0.05557637 0.02867777
```

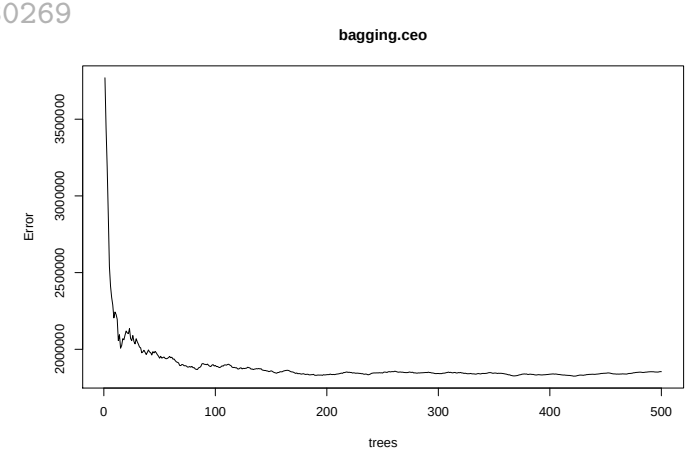


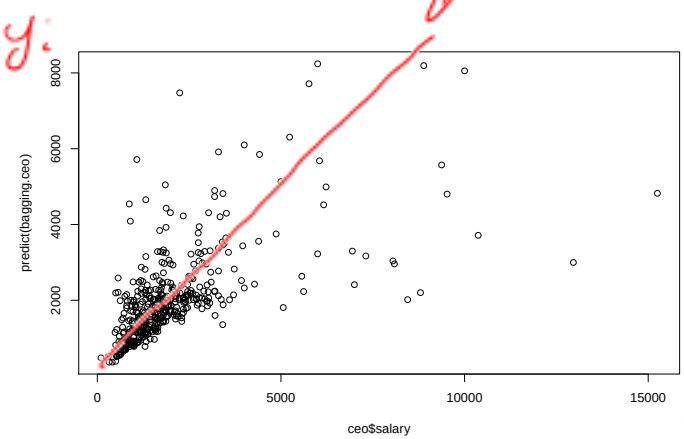
Generalizations

- Bagging: if you use for CART just a subsample, then you obtain a completely different tree.
  - Fit a CART to B random subsamples (*bootstrap*).
  - The error is measured on the remaining observations out-of-bag.
  - The final forecast is:

$$\hat{f}_{avr}(\boldsymbol{x}_0) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(\boldsymbol{x}_0).$$

```
> bagging.ceo= randomForest(salary ~ ., data=ceo, mtry=6)
> predict(bagging.ceo);
> cor(ceo$salary,predict(bagging.ceo))
[1] 0.6180269
```





problem: trees will be similar if you have a dominant variable ...

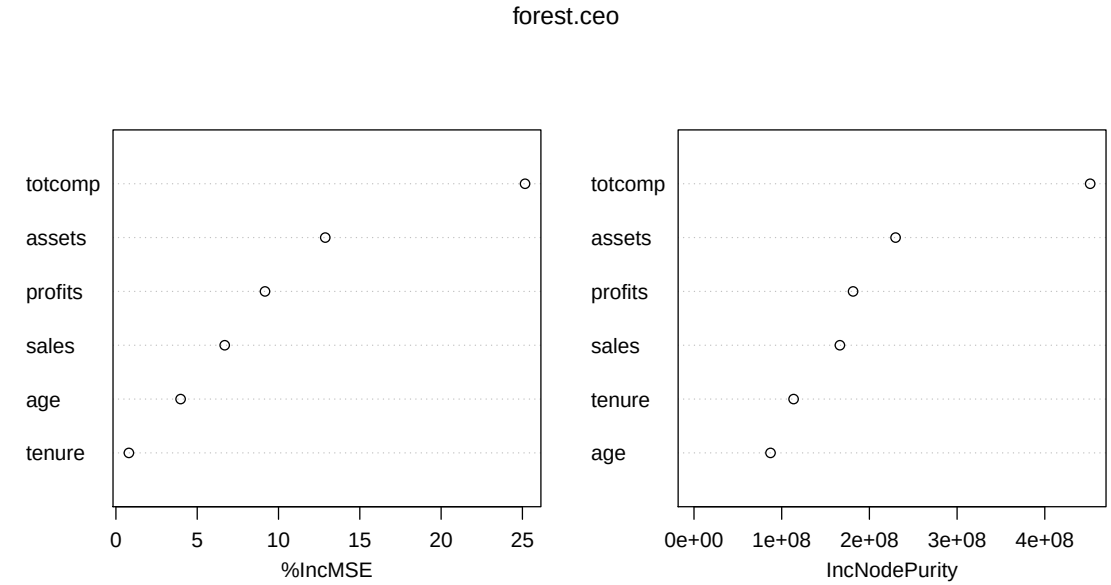
Random Forests: is a generalization of *Bagging*

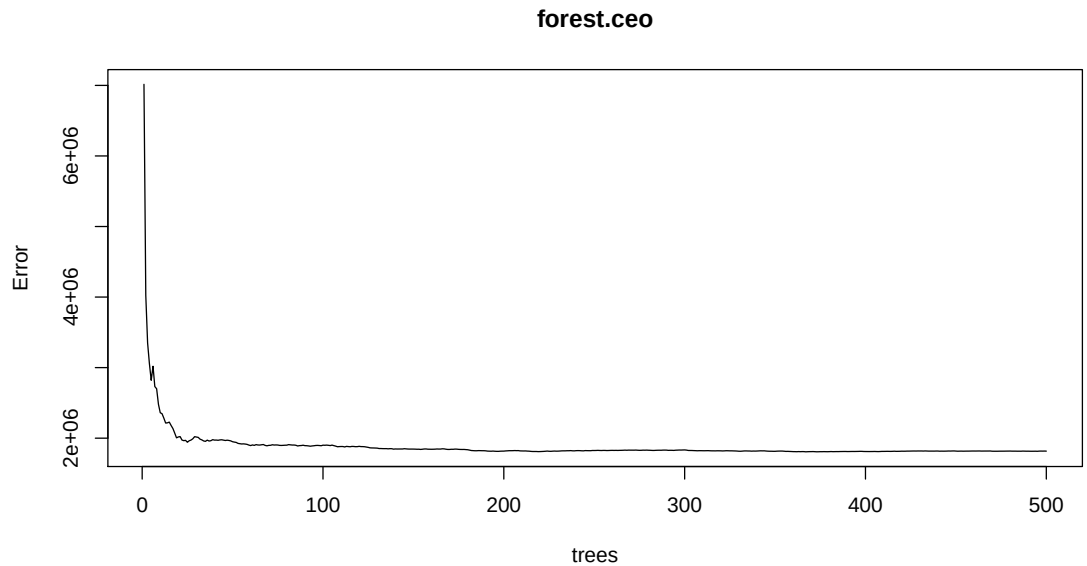
- For each splitting you consider not all explanatory variables bus just a subset of size $M \approx \sqrt{J}$
  ↳ draw not only a subsample of observations but also a subset of X-variables.
- ... this makes the trees more heterogenous and "uncorrelated" in terms of forecasts
- Each tree is grown on a bootstrap sample (as for bagging)
- The importance of a variable is measured by increase in (a) MSE ; (b) in node impurity over the out-of-bag sample if the variable is permuted

$$\Delta MSE_{j,b} = \frac{1}{|\overline{\mathcal{B}}_b|} \sum_{k \in \overline{\mathcal{B}}_b} \hat{u}^2(x_{1k}, \ldots, x_{Jk}) - \frac{1}{|\overline{\mathcal{B}}_b|} \sum_{k \in \overline{\mathcal{B}}_b} \tilde{\hat{u}}_k^2(x_{1k}, ..., x_{j-1,k}, \tilde{x}_{jk}, x_{j+1,k}, \ldots, x_{Jk}),$$

where $\tilde{\boldsymbol{x}}_j$ are the randomly permuted (reordered) observations on the jth variable and $\overline{\mathcal{B}}_b$ is the bth out-of-bag subsample.

```
> forest.ceo= randomForest(salary ~ ., data=ceo, importance=T)
> varImpPlot(forest.ceo)
```

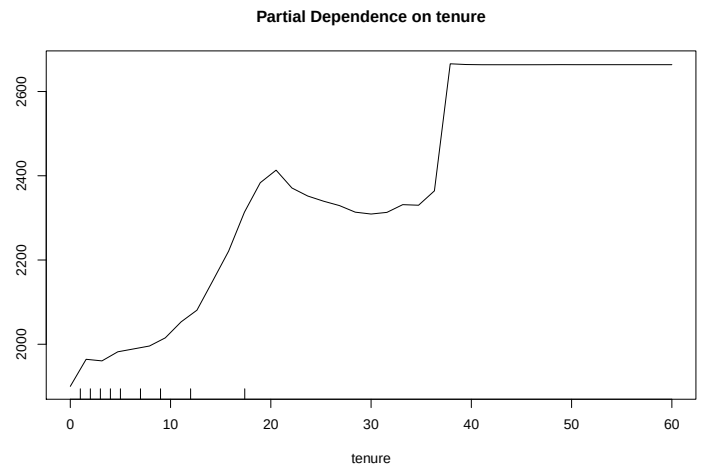


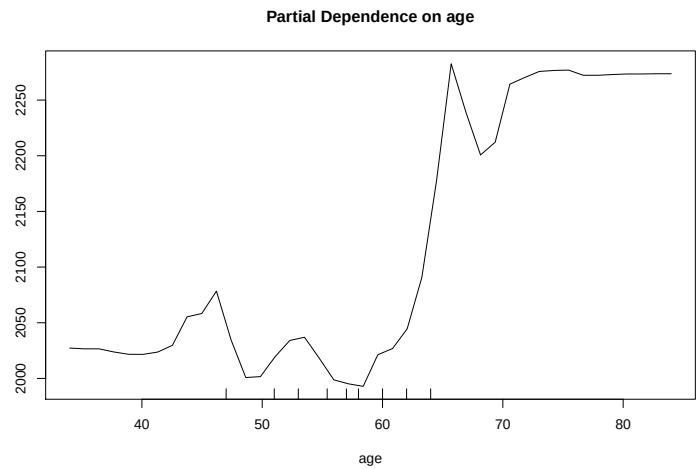


Partial dependence plots: visualize the marginal impact of a variable/feature

$$\tilde{f}_j(x) = \frac{1}{K}\sum_{k=1}^{K}\hat{f}(x_{1k}, ..., x_{j-1,k}, x, x_{j+1,k}, \ldots, x_{Jk})$$

```
> partialPlot(forest.ceo, pred.data=ceo, x.var=tenure)
```





Ctree; C5.0

CHAID

- An alternative approach is CHAID (Chi-square Automatic Interaction Detectors): allows not only for binary splitting and is similar to ANOVA.
- Analysis is a generalization of two-sample test for the mean.
- Idea: let G be the number of splittings for variable X. We test if there is a significant difference between the means of Y in different regions.

CART – no tests ⇒ just optimization

CHAID – test. ⇒ we test the difference in the means for two or more rectangles

split only if $H_0: \mu_1 = \mu_2$ is rejected.

triple-splitting if $H_0: \mu_1 = \mu_2 = \mu_3$ is rejected.

Total sum of squares

$$TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2 = \sum_{g=1}^{G}\sum_{i:\boldsymbol{x}_i \in R_g}(y_i - \bar{y})^2$$

Within sum of squares

$$WSS = \sum_{g=1}^{G}\sum_{i:\boldsymbol{x}_i \in R_g}(y_i - \bar{y}_{R_g})^2$$

Between sum of squares

$$BSS = TSS - WSS = \sum_{g=1}^{G}|R_g|(\bar{y}_{R_g} - \bar{y})^2.$$

$H_0 : \mu_1 = \cdots = \mu_G$ vs $H_1 : \mu_i \neq \mu_j$ for at least one pair i, j

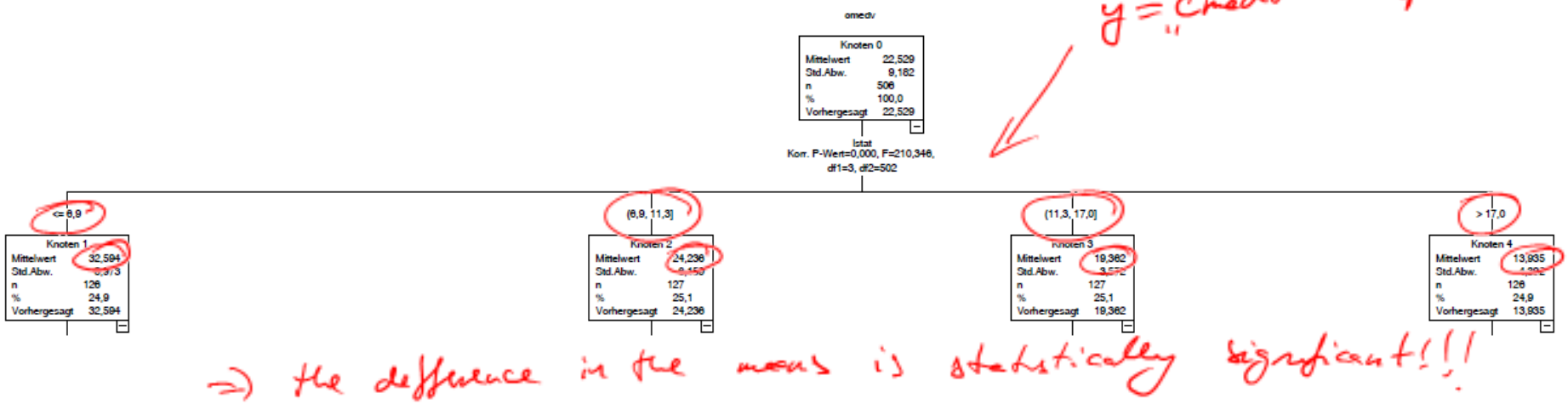
Test statistic: $F = \dfrac{BSS/(G-1)}{WSS/(n-G)} \sim F_{G-1,n-G}$

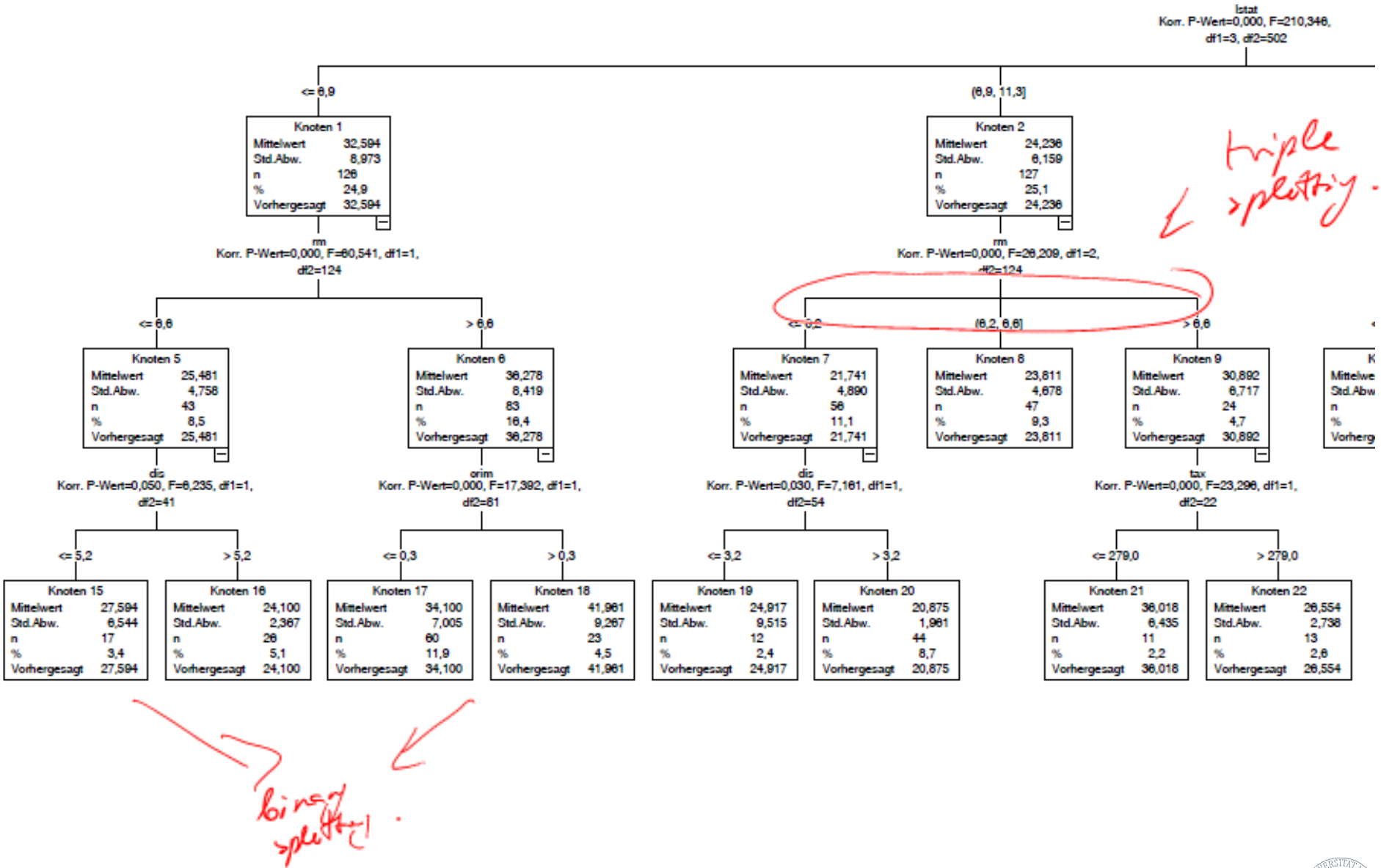
splitting in G rectangles simultaneously

F-distribution

Idea:

- For each predictor we determine the optimal splitting, i.e. the regions with the smallest p-value of the test.
- The p-values should be corrected due to multiple testing (Bonferroni correction).
- The predictor with the smallest corrected p-value is used for splitting.





Baumtabelle

Knoten	Mittelwert	Standardabweichung	N	Prozent	Vorhergesager Mittelwert	Übergeordneter Knoten	Primäre unabhängige Variable					
							Variable	Sig.[a]	F	df1	df2	Werte aufteilen
0	22,53	9,182	506	100,0%	22,53							
1	32,59	8,973	126	24,9%	32,59	0	lstat	,000	210,346	3	502	<= 6,9
2	24,24	6,159	127	25,1%	24,24	0	lstat	,000	210,346	3	502	(6,9, 11,3]
3	19,36	3,572	127	25,1%	19,36	0	lstat	,000	210,346	3	502	(11,3, 17,0]
4	13,93	4,392	126	24,9%	13,93	0	lstat	,000	210,346	3	502	> 17,0
5	25,48	4,758	43	8,5%	25,48	1	rm	,000	60,541	1	124	<= 6,6
6	36,28	8,419	83	16,4%	36,28	1	rm	,000	60,541	1	124	> 6,6
7	21,74	4,890	56	11,1%	21,74	2	rm	,000	26,209	2	124	<= 6,2
8	23,81	4,678	47	9,3%	23,81	2	rm	,000	26,209	2	124	(6,2, 6,6]
9	30,89	6,717	24	4,7%	30,89	2	rm	,000	26,209	2	124	> 6,6
10	22,59	3,393	17	3,4%	22,59	3	tax	,000	18,192	1	125	<= 279,0
11	18,86	3,345	110	21,7%	18,86	3	tax	,000	18,192	1	125	> 279,0
12	19,31	2,881	14	2,8%	19,31	4	nox	,000	36,153	2	123	<= ,5
13	15,72	3,623	44	8,7%	15,72	4	nox	,000	36,153	2	123	(,5, ,6]
14	11,67	3,554	68	13,4%	11,67	4	nox	,000	36,153	2	123	> ,6
15	27,59	6,544	17	3,4%	27,59	5	dis	,050	6,235	1	41	<= 5,2
16	24,10	2,367	26	5,1%	24,10	5	dis	,050	6,235	1	41	> 5,2
17	34,10	7,005	60	11,9%	34,10	6	crim	,000	17,392	1	81	<= ,3
18	41,96	9,267	23	4,5%	41,96	6	crim	,000	17,392	1	81	> ,3
19	24,92	9,515	12	2,4%	24,92	7	dis	,030	7,161	1	54	<= 3,2
20	20,88	1,961	44	8,7%	20,88	7	dis	,030	7,161	1	54	> 3,2
21	36,02	6,435	11	2,2%	36,02	9	tax	,000	23,296	1	22	<= 279,0
22	26,55	2,738	13	2,6%	26,55	9	tax	,000	23,296	1	22	> 279,0
23	19,51	2,847	75	14,8%	19,51	11	nox	,007	9,596	1	108	<= ,6
24	17,47	3,911	35	6,9%	17,47	11	nox	,007	9,596	1	108	> ,6
25	17,24	3,698	24	4,7%	17,24	13	rad	,022	11,566	1	42	2,0; 5,0; 6,0; 24
26	13,90	2,596	20	4,0%	13,90	13	rad	,022	11,566	1	42	4,0
27	10,85	3,087	56	11,1%	10,85	14	dis	,000	22,769	1	66	<= 2,1
28	15,53	3,094	12	2,4%	15,53	14	dis	,000	22,769	1	66	> 2,1

Nonlinear regression

The general form of a nonlinear regression is:

$$y_k = h(\boldsymbol{x}_k, \boldsymbol{\beta}) + u_k,$$

where $h(\cdot, \cdot)$ is some unknown function of the regressors and parameters.

- $y = e^{\beta_0} e^{\beta_1 x_1} e^{\beta_2 x_2} e^{u}$ - can be linearized
- $y = \beta_0 + \beta_1 e^{\beta_2 x_1} + u$ - cannot be linearized
- $y = \beta_0 + \beta_1 x_1^{\gamma} + u$ - cannot be linearized

A popular special case of the non-linear regression is the single-index model

$$y_k = h(\boldsymbol{x}_k' \boldsymbol{\beta}) + u_k,$$

thus h is a function of a linear combination of the regressors.

Assumptions

- as before +
- $E(\boldsymbol{u}|\boldsymbol{X}) = \mathbf{0}$ is replaced with $E(u_i|h(\boldsymbol{x}_i, \boldsymbol{\beta})) = 0$: if u is uncorrelated with $\boldsymbol{x}$ it still may be correlated with some function of $\boldsymbol{x}$. In general $E(\boldsymbol{u}|\boldsymbol{X}) = \mathbf{0}$ is not needed.
- Identifiability of the model parameters: the model is identifiable if there is no a non-zero parameter $\boldsymbol{\beta}_0$, such that $h(\boldsymbol{x}_i, \boldsymbol{\beta}_0) = h(\boldsymbol{x}_i, \boldsymbol{\beta})$ for all $\boldsymbol{x}_i$.

  Note: in the linear regression it is sufficient to assume $rank(\boldsymbol{X}'\boldsymbol{X}) = J + 1$. Here it is not enough.

$$y = \frac{2\cdot\beta_0 + 2\beta_1 x_1}{2\cdot\beta_2 + 2\beta_3 x_2} + u.$$

the parameters cannot be identified uniquely!

$\Rightarrow$ impose some restriction: $\beta_0 = 1$ $\quad y = \frac{1 + \beta_1 x_1}{\beta_2 + \beta_3 x_2} + u.$

$$y = \beta_0 + \beta_1 \cdot e^{\beta_2 + \beta_3 x} + u = \beta_0 + \underbrace{\beta_1 \cdot e^{\beta_2}}_{\beta_4} \cdot e^{\beta_3 x} + u.$$

Estimation: the LS estimation can be used, but the asymptotic theory follows in a straightforward way from the quasi (!) maximum-likelihood estimation.

Assuming Gaussian residuals it holds:

density function of y's

Gaussian density

our nonlinear component

$$\begin{aligned}
\mathcal{L}(y_k|\boldsymbol{x}_k, \boldsymbol{\beta}, u_k) &= \frac{1}{K}\sum_{k=1}^{K} \ln f(y_k|\boldsymbol{x}_k, \boldsymbol{\beta}, u_k) \\
&= \frac{1}{K}\sum_{k=1}^{K} \ln\left\{\frac{1}{\sqrt{2\sigma^2}} exp\left(-\frac{1}{2\sigma^2}(y_k - h(\boldsymbol{x}_k, \boldsymbol{\beta}))^2\right)\right\} \\
&= \frac{1}{K}\sum_{k=1}^{K} \left[-\ln\sqrt{2\sigma^2} - \frac{1}{2\sigma^2}(y_k - h(\boldsymbol{x}_k, \boldsymbol{\beta}))^2\right] \longrightarrow max
\end{aligned}$$

w.r.t. β

Thus the first order conditions for $\boldsymbol{\beta}$ are

system of nonlinear equations

$$\frac{\partial \mathcal{L}(y_k|\boldsymbol{x}_k, \boldsymbol{\beta}, u_k)}{\partial \boldsymbol{\beta}} = \sum_{k=1}^{K}(y_k - h(\boldsymbol{x}_k, \boldsymbol{\beta}))\frac{\partial h(\boldsymbol{x}_k, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \mathbf{0}.$$

$\Rightarrow$ no explicit solution.

$\rightsquigarrow$ mostly a highly nonlinear system of equations solved numerically.

$\Rightarrow$ in contrast to $\hat{\beta} = (X'X)^{-1}X'y$ in the linear model.

Consequences: since the resulting $\hat{\boldsymbol{\beta}}$ is a non-linear function of the residuals u_k

- ... the unbiasedness can not be proven in simple fashion;
- ... the variance of $\hat{\boldsymbol{\beta}}$ is not easy to derive;
- ... the exact distribution of $\hat{\boldsymbol{\beta}}$ is not Gaussian;
- ... all the inferences, like tests, are valid only asymptotically.

→ t-test; F-test

but the ML estimators are consistent and efficient (they possess the smallest variance among all consistent and asymptotically normal estimators)

Where all this comes from?

- Taylor expansion of $f(x)$ in neighborhood of x_0

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}f''(x_0)(x - x_0)^2 + ...$$

- Exact Taylor expansion of $f(x)$ in neighborhood of x_0 (mean-value theorem)

$$f(x) = f(x_0) + f'(x_+)(x - x_0),$$

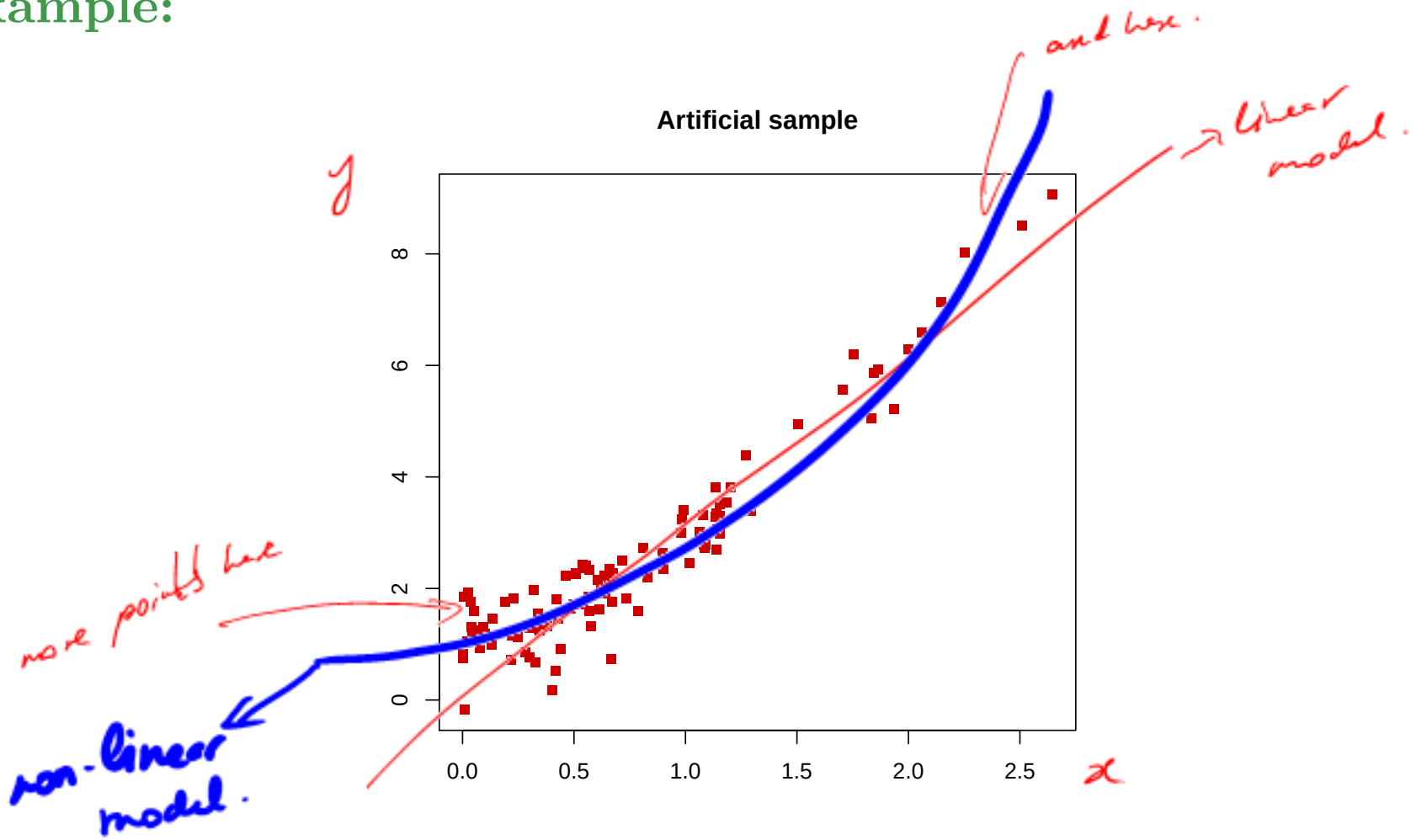
where x_+ lies between x and x_0.

- $$\left.\frac{\partial \mathcal{L}(y_k|\boldsymbol{x}_k, \boldsymbol{\beta}_k, u_k)}{\partial \boldsymbol{\beta}}\right|_{\hat{\boldsymbol{\beta}}} = \left.\frac{\partial \mathcal{L}(y_k|\boldsymbol{x}_k, \boldsymbol{\beta}_k, u_k)}{\partial \boldsymbol{\beta}}\right|_{\boldsymbol{\beta}} + \left.\frac{\partial^2 \mathcal{L}(y_k|\boldsymbol{x}_k, \boldsymbol{\beta}_k, u_k)}{\partial \boldsymbol{\beta}\partial \boldsymbol{\beta}'}\right|_{\boldsymbol{\beta}_+} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})$$

- $$\sqrt{K}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) = -\left(\left.\frac{\partial^2 \mathcal{L}(y_k|\boldsymbol{x}_k, \boldsymbol{\beta}_k, u_k)}{\partial \boldsymbol{\beta}\partial \boldsymbol{\beta}'}\right|_{\boldsymbol{\beta}_+}\right)^{-1} \sqrt{K}\left.\frac{\partial \mathcal{L}(y_k|\boldsymbol{x}_k, \boldsymbol{\beta}_k, u_k)}{\partial \boldsymbol{\beta}}\right|_{\boldsymbol{\beta}}$$

- $$\sqrt{K}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \overset{approx}{\sim} N(\mathbf{0}, \boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{A}^{-1})$$

Example:



- Model 1 : $y = \beta_0 + \beta_1 x + u$
- Model 2 : $y = \beta_0 + \beta_1 x^{\beta_2} + u$

cannot be linearized

power regression.

linear models

non-linear least-squares.

```
> z1 = lm(y~x):
> z2 = nls(y~ b0+b1*x^b2, start=list(b0=0, b1=1, b2=2))
> summary(z1)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.44547    0.09694   4.595 1.29e-05 ***
x            2.78805    0.09787  28.488  < 2e-16 ***
---
Residual standard error: 0.604 on 98 degrees of freedom
Multiple R-squared:  0.8923,    Adjusted R-squared:  0.8912
F-statistic: 811.5 on 1 and 98 DF,  p-value: < 2.2e-16
```

starting values for the optimization

from linear model

$\Rightarrow$ from here linear model is a good model, with significant β_1 and high R^2!

$\Rightarrow$ look at residuals!!!

```
> summary(z2)
Formula: y ~ b0 + b1 * x^b2

Parameters:
   Estimate Std. Error t value Pr(>|t|)
b0  1.08702    0.09345   11.63   <2e-16 ***
b1  1.77926    0.13041   13.64   <2e-16 ***
b2  1.56810    0.08382   18.71   <2e-16 ***
---

Residual standard error: 0.4678 on 97 degrees of freedom

Number of iterations to convergence: 5
Achieved convergence tolerance: 2.905e-06
```

Note: that $\beta_1 = 2.78805$ from the linear model has no relationship to β_1 and β_2 in the non-linear model.

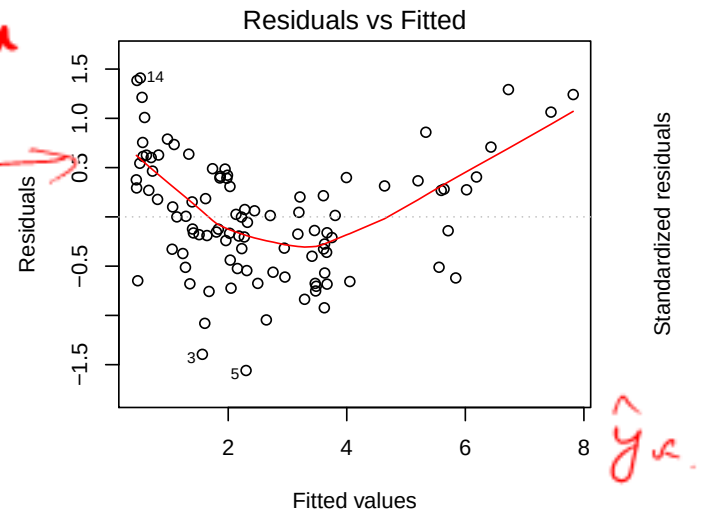
close to the true values!

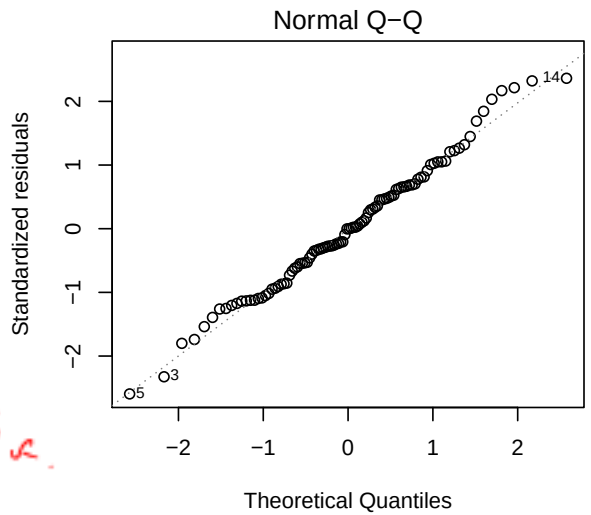
β_0 β_1 β_2

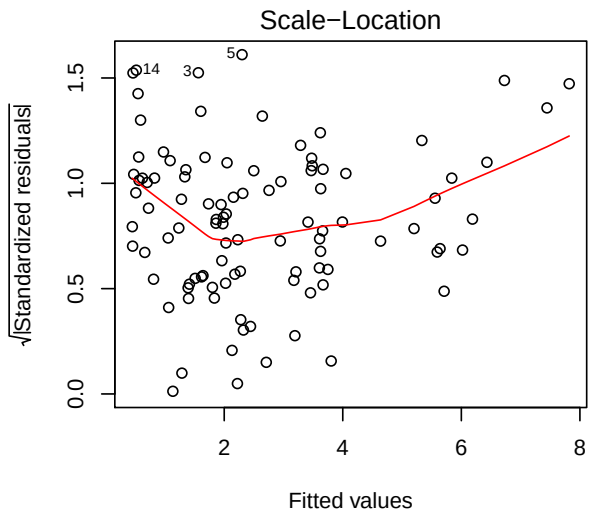
True model: $y = 1 + 2x^{1.5} + u,\ u \sim N(0, 0.5^2)$.

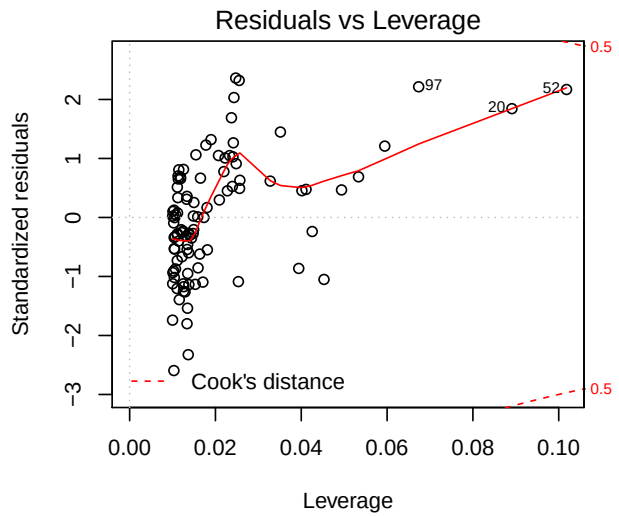
I used to simulate the data

lm(y ~ x)









LR: big data: first estimate the model $\hat{\beta}_0, \dots \hat{\beta}_J \Rightarrow$ only after this the model selection

K - small; J - big $\Rightarrow$ low precision $\Rightarrow$ many β_j insignificant $\Rightarrow$ unreliable selection.

Lasso regression

Here: model selection and estimation simultaneously

Problems:

- accuracy: In K is much larger than J, then the variances are small and the inferences are precise. Low number of observations per parameter implies general high variability.
- interpretability: In large data sets there always irrelevant variables which make the economic interpretability difficult.
- sparsity: only a subset of the explanatory variables is relevant economically and statistically.

Solution: stepwise variable selection procedures based on statistical properties of the estimators or lasso regression

Idea: minimize the sum of squared residuals with constrains on the parameters. ⇒ model selection: which β are set to zero and which not!

The objective function of the OLS procedure is replaced with

$$\sum_{k=1}^{K}(y_k - \beta_0 - \sum_{j=1}^{J}\beta_j x_{kj})^2 \quad + \quad \lambda \sum_{j=1}^{J}|\beta_j| \longrightarrow min, \text{ w.r.t } \beta_j\text{'s}$$

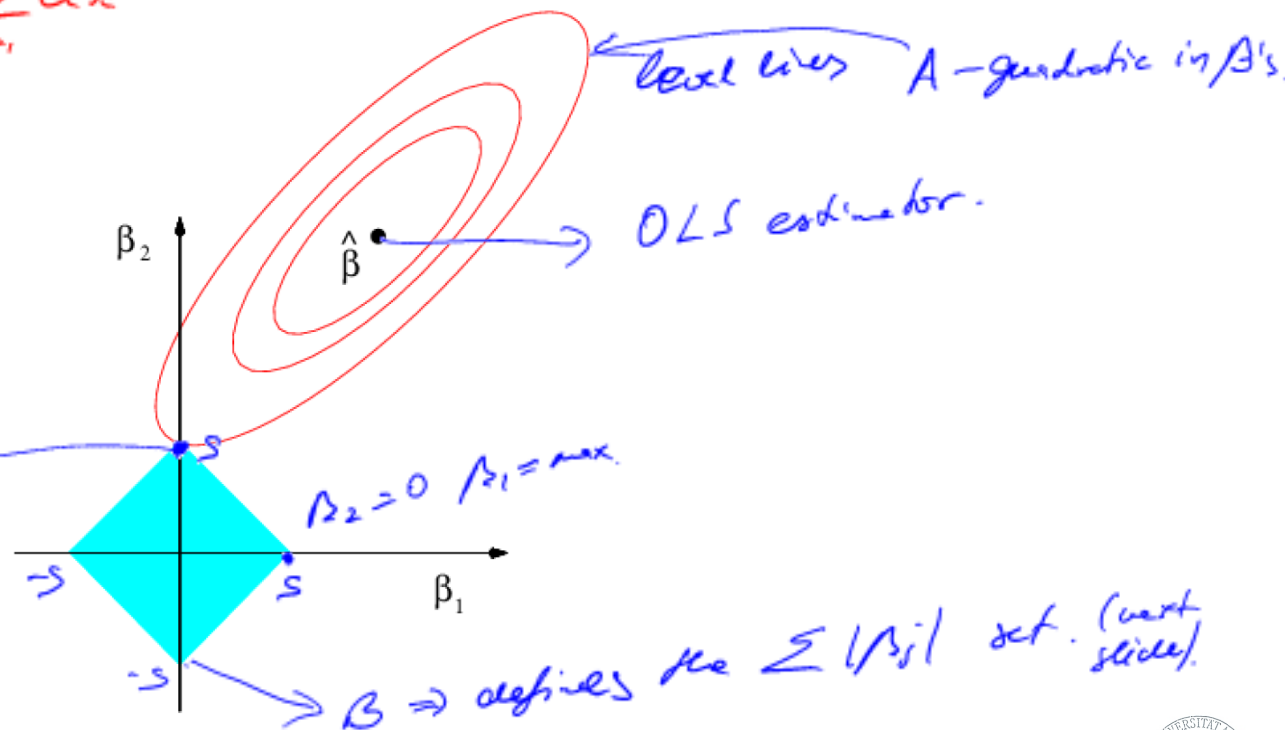
A — control parameter — β — to set rid of >0 <0

as in the OLS: $\sum_{k=1}^{K} u_k^2$ → forces some of β to become zero.

λ – large ⇒ strict penalization ⇒ more β's are zero

λ – small ⇒ soft penalization

$\lambda = 0$ ⇒ OLS



$f(x) \to \min,\ g(x) \le a \Rightarrow f(x) + \lambda(g(x)-a) \to \min,\ \lambda$ – Lagrange.

Note:

- The problem is equivalent to the following problem, i.e. for each λ there exists s such that both problems lead to the same lasso-coefficients.

OLS $\Rightarrow$

$$\sum_{k=1}^{K}(y_k - \beta_0 - \sum_{j=1}^{J}\beta_j x_{kj})^2 \longrightarrow min, \text{ w.r.t. } \beta_j\text{'s}$$

$$\text{s.t.} \quad \sum_{j=1}^{J}|\beta_j| \le s.$$

constrained OLS

$s \longleftrightarrow \lambda$

- Minimizing the objective is not trivial and there many specific numerical methods developed for this purpose.
- Selecting a good value for λ is crucial. The optimal value is chosen by cross-validation.

Special case

Assume an individual constant for each observation:

$$\sum_{k=1}^{K}(y_k - \beta_k)^2$$

replace our x'p by a individual constant, for every yk.

with the OLS solution $\hat{\beta}_k = y_k$.

With lasso we obtain:

lasso part

$$\sum_{k=1}^{K}(y_k - \beta_k)^2 + \lambda \sum_{k=1}^{K} |\beta_k| \longrightarrow min$$

$2(y_k - \beta_k) + \lambda = 0$

$\beta_k = y_k + \lambda/2$.

with the solution

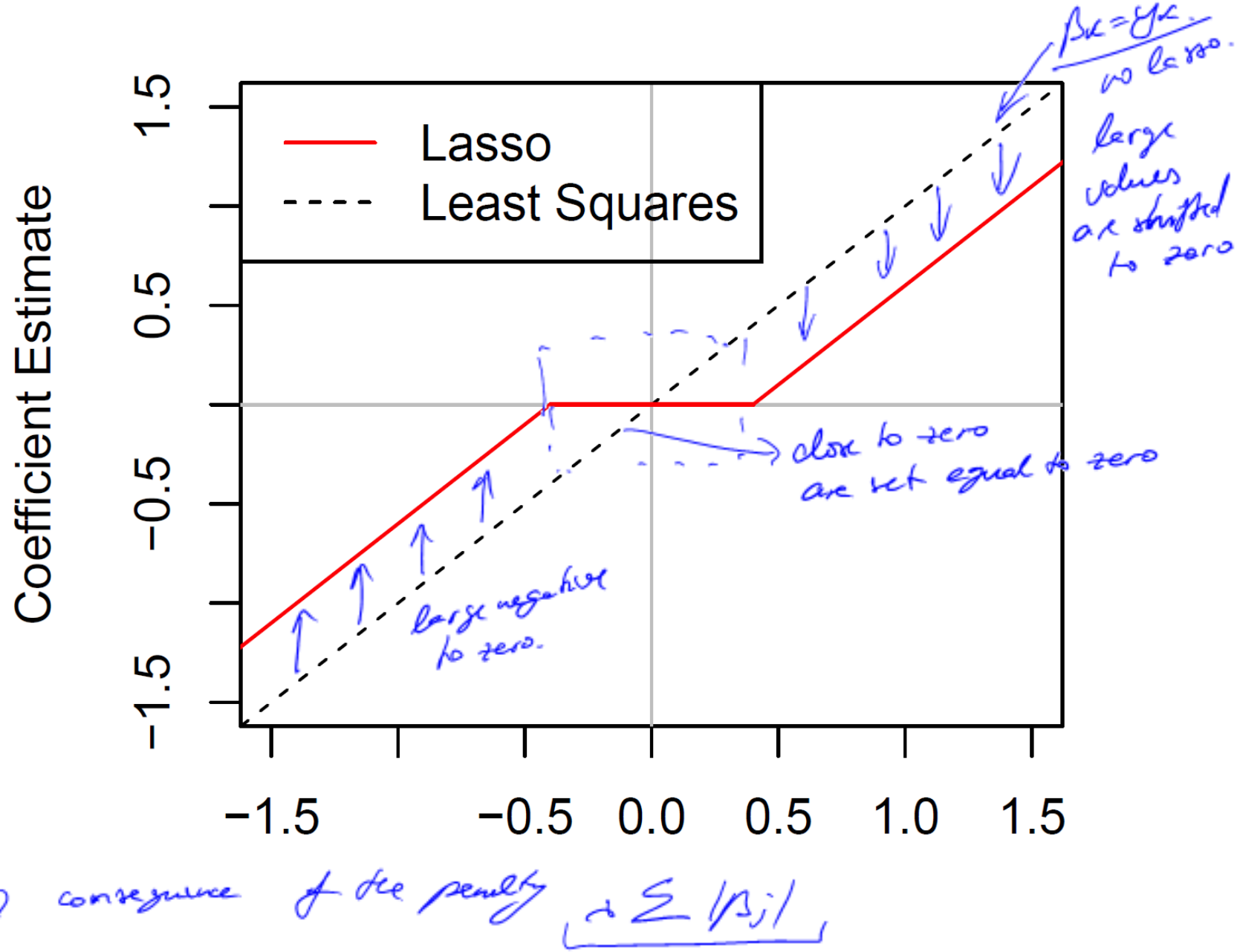
lasso estimator of the constant.

$$\hat{\beta}_k^{(lasso)} = \begin{cases} y_k - \lambda/2, & \text{if } y_k \geq \lambda/2 \\ y_k + \lambda/2, & \text{if } y_k \leq -\lambda/2 \\ 0, & \text{if } |y_k| \leq \lambda/2 \end{cases}$$

$\Rightarrow$ y_k is large positive

$\Rightarrow$ y_k is large negative.

$\Rightarrow$ y_k is close to zero.

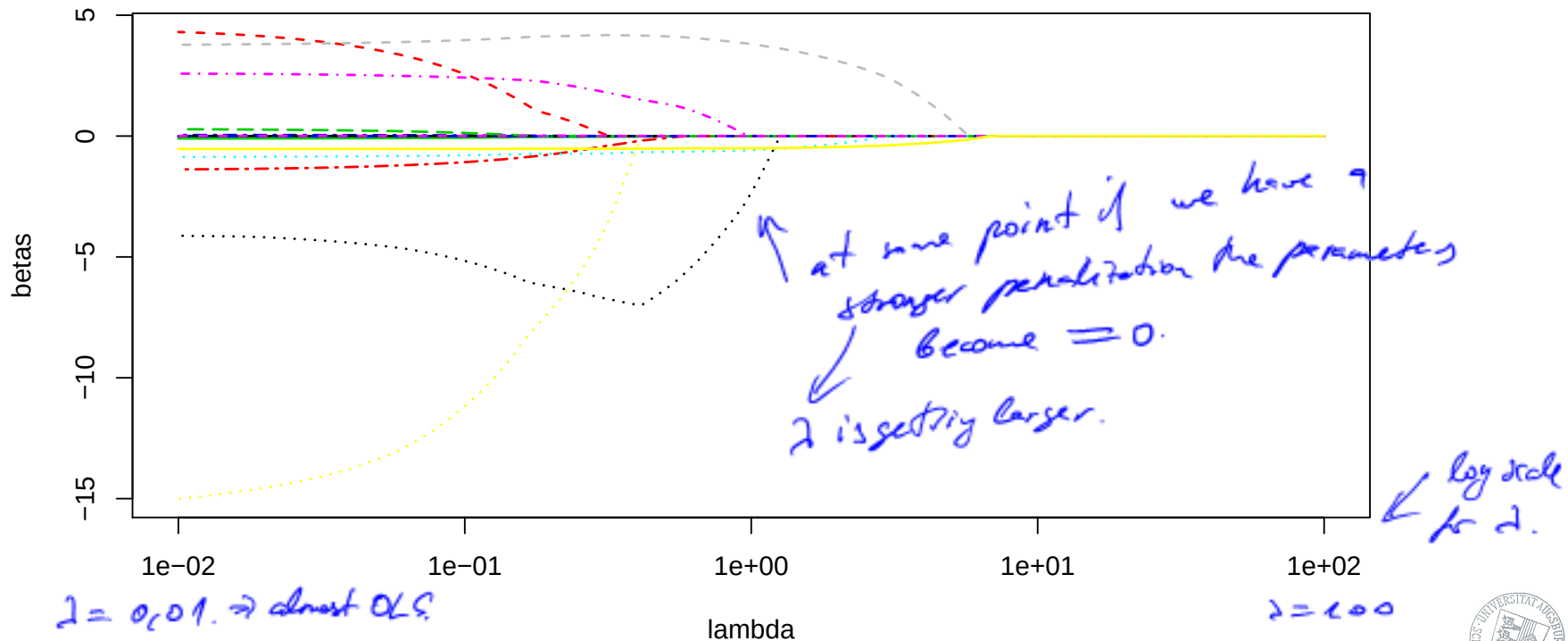


Example:

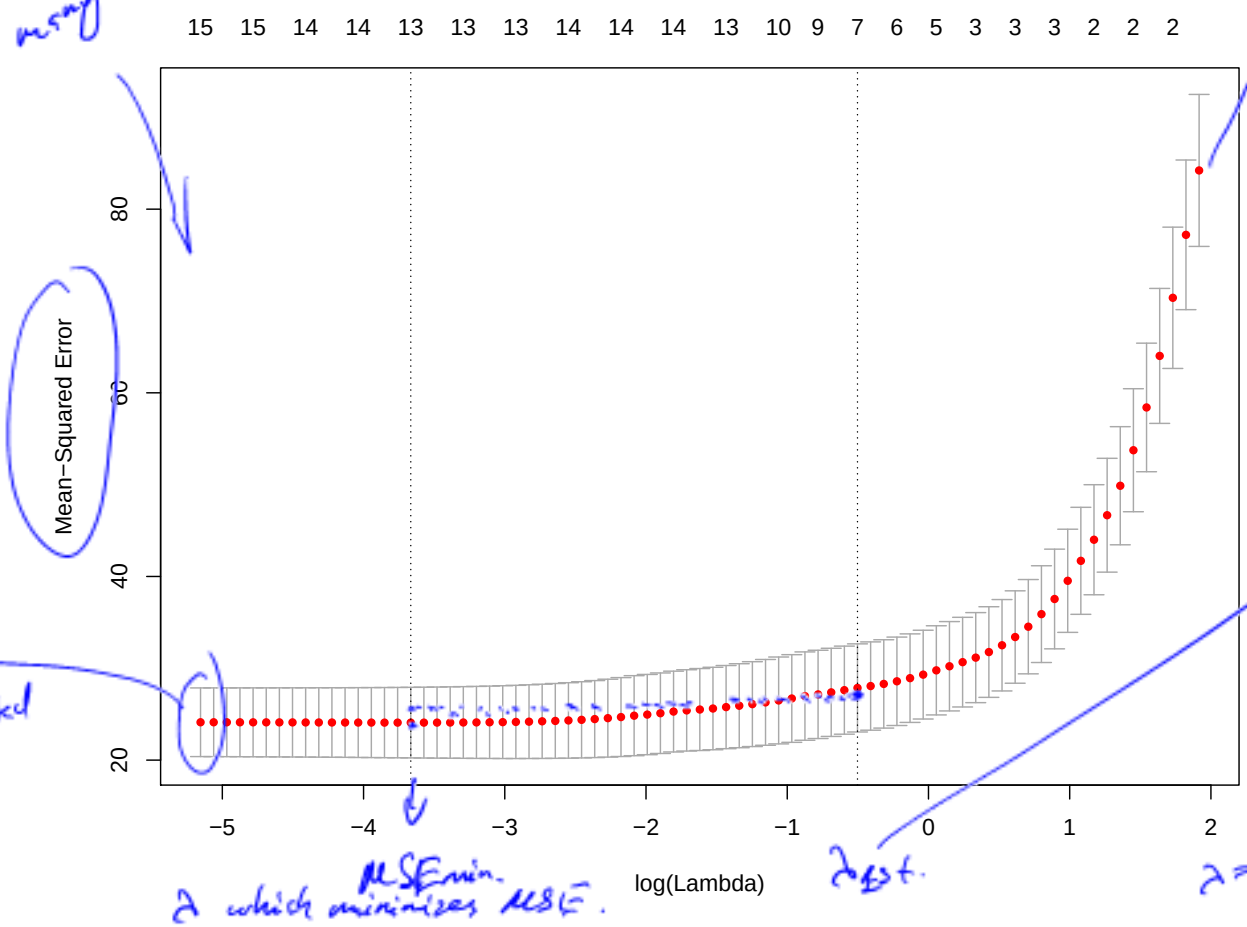
Trevor Hastie (Statistical learning book!

lasso.

```
> grid = 10^seq(2,-2, length=100)
> lasso = glmnet(X, y.boston, alpha=1, lambda=grid);
> plot(lasso$beta)
```



```
> cv.lasso = cv.glmnet(X, y.boston, alpha=1);
> plot(cv.lasso)
> cv.lasso$lambda.min
[1] 0.0255856
```



```
> lasso.coef = predict(lasso, type="coefficients", s=cv.lasso$lambda.min);
> lasso.coef
16 x 1 sparse Matrix of class "dgCMatrix"
                         1
(Intercept) -4.392079e+02
lon         -4.250433e+00
lat          4.017435e+00
crim        -9.553123e-02
zn           4.149031e-02
indus        .
chas1        2.557345e+00
nox         -1.432740e+01
rm           3.816713e+00
age          .
dis         -1.329954e+00
rad          2.572257e-01
tax         -1.071061e-02
ptratio     -8.520927e-01
b            8.974756e-03
lstat       -5.326668e-01
```

λ_2 which minimizes MSE

$\Rightarrow$ set = 0

others are smaller, than OLS (in most of the cases)

$\Rightarrow$ set = 0