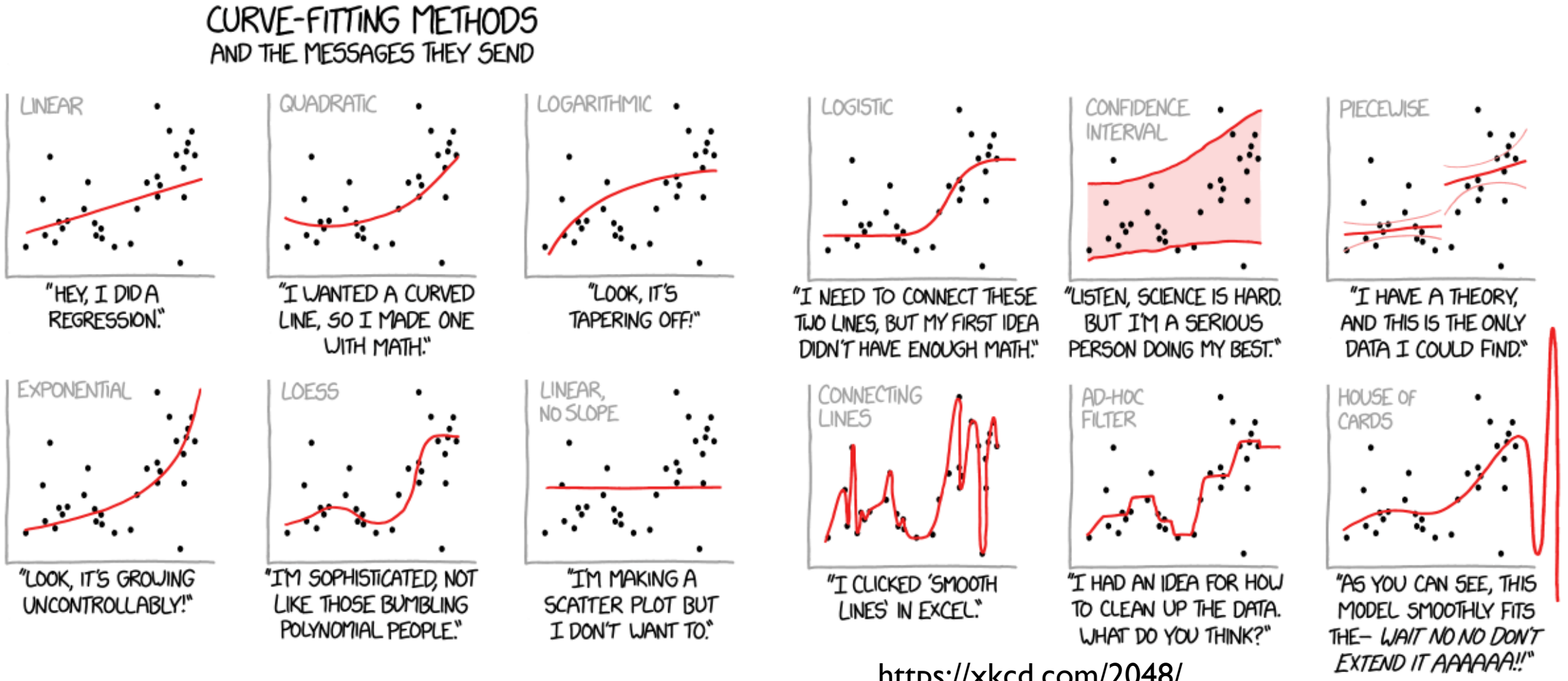
ELEMENTS OF DATA SCIENCE AND STATISTICAL LEARNING

SPRING 2020

Week 6

CURVE FITTING



https://xkcd.com/2048/

OUTLINE

- Preface: beyond linearity (polynomial regression/refresher)
- Extensions of linear regression and related models
  - Step functions
  - Regression splines, natural splines
  - Smoothing splines
  - Local regression
  - Generalized Additive Models
- Examples, with resampling to assess test error

BENDING YOUR LINEAR MODEL: WHEN, WHY AND HOW

- When there is enough statistical evidence that the bias of linear model is too high
  - Residuals vs fitted plot
- And, there is enough data to keep variance of the resulting more flexible model in check
  - Probably if there is enough data to confidently detect that bias is too high, there is enough data to fit "better" model
- And, we do not have enough insight into the problem to use specific non-linear model based on science/engineering
  - E.g. Michaelis-Menten for enzyme kinetics, exponential or sigmoid curve
  - MASS, nls, …
- This week we cover extensions of linear models increasing their flexibility while still relying on least squares fit or modifications thereof
  - It always has a solution, even if subpar, unlike non-linear regression where conversion to optimum is not guaranteed
- These models still rely on simplifying assumption that observed data is correctly represented by independent random variability around average model fit
  - Settings when randomness is not independent (e.g. repeatedly measured test scores for individual students across multiple schools across multiple districts) represent a field of their own: LME, GEE, etc.

POLYNOMIAL REGRESSION

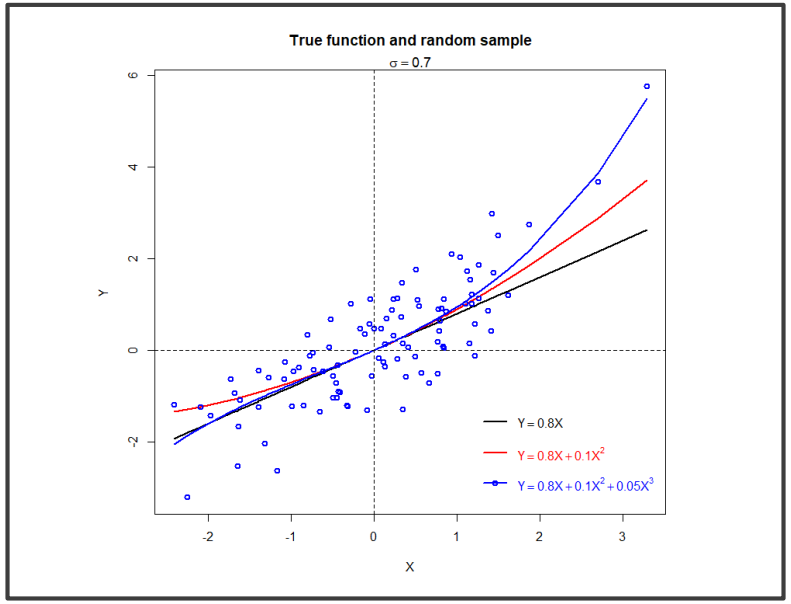
- We have already discussed this straightforward extension of a linear model:
  - In addition to X we can introduce additional variables $X^2, X^3, \ldots$ and build a model:

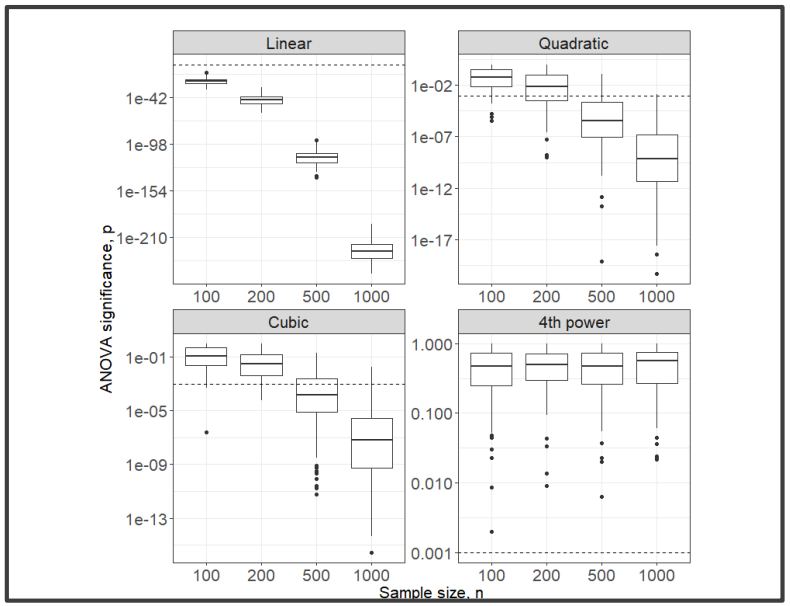
    $Y = X + X^2 + \cdots + \varepsilon,$ i.e. $y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \varepsilon_i$

  - This can be generalized to the case of multiple regression with predictors $X_1, \ldots, X_p$: consider additional variables $X_1^2, X_1^3, \ldots, X_2^2, X_2^3, \ldots, \ldots, X_p^2, X_p^3$
- How to fit in R:
  - Use the same function `lm()`. It's a conventional linear model after all, we just introduced additional variables.
  - Since character "^" has a special meaning in R formulas, we need to "protect" it in order to give it the original meaning "to the power of": `lm( Y ~ X + I(X^2) + I(X^3) )`
  - … or explicitly build additional vectors/columns and refer to them:

```
my.data <- data.frame(Y=…,X=…)
my.data$X2 <- my.data$X^2
lm( Y ~ X + X2, data=my.data)
```

  - …or use function **`poly()`**: `lm( Y ~ poly(X,3) )` or `lm( Y ~ poly(X,3,raw=T)`; the former builds *orthogonal polynomials*, while the latter is equivalent to Y~X+I(X^2)+I(X^3) in this example





HOW MUCH DATA?

- We have seen already an illustration of higher-order dependency washed out by noise (see slides for Week 4)
- Let's add a little more depth and see how the fit to the data generated from the *same* underlying process changes with the amount of data available:
- Larger sample sizes are needed to reliably estimate higher power polynomial regression terms
  - For any given problem the exact relationship between significance and sample size will be different, but this is a general effect!
- What is shown in the plots:
  - Top: the data; true underlying dependence is 3rd order (blue) line; simulated observations (blue dots) are true dependence + normal noise; note that black/red lines are not regression fits but rather the true dependence with higher order term(s) dropped
  - Bottom: simulate a dataset (of the type shown on top), fit *four* models ($Y \sim X, Y \sim X+X^2, Y \sim X+X^2+X^3, Y \sim X+X^2+X^3+X^4$), calculate ANOVA p-values of each model; repeat multiple times to obtain a distribution of p-values; repeat for different sample sizes.

CODE FOR PREVIOUS PLOTS

```
a <- 0.8; b <- 0.1; c <- 0.05; errEps <- 0.7
x <- sort(rnorm(100))
y1 <- a*x; y2 <- a*x + b*x^2; y3 <- a*x+b*x^2+c*x^3
ySmpl <- y3+rnorm(length(x),sd=errEps)
plot(range(x),range(c(y1,y2,y3,ySmpl)),type="n",xlab="X",ylab="Y",
     main="True function and random sample")
points(x,y1,col=1,type="l",lwd=2)
points(x,y2,col=2,type="l",lwd=2)
points(x,y3,col=4,type="l",lwd=2)
points(x,ySmpl,col=4,lwd=2)
abline(h=0,lty=2)
abline(v=0,lty=2)
legend("bottomright",c(parse(text=paste0("Y==paste(",a,",X)")),
                parse(text=paste0("Y==paste(",a,",X)+paste(",b,",X^2)")),
                parse(text=paste0("Y==paste(",a,",X)+paste(",b,",X^2)+paste(",c,",X^3)"))),
                lty=1,lwd=2,col=c(1,2,4),text.col=c(1,2,4),bty="n",pch=c(NA,NA,1))
mtext(bquote(sigma==.(errEps)),cex=1.1)
```

CODE FOR PREVIOUS PLOTS

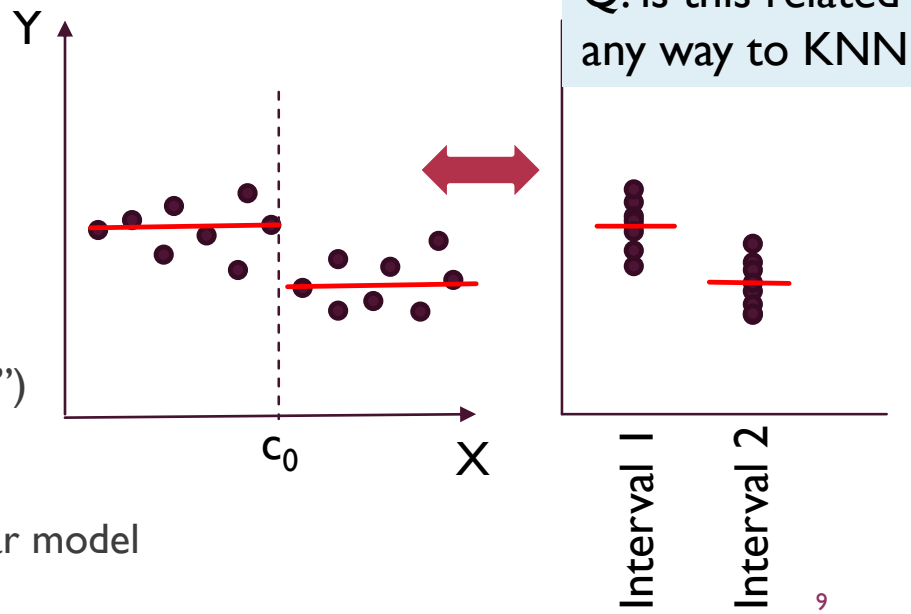
```
dfTmp <- NULL
for ( smplSz in c(100,200,500,1000) ) {
  for ( iTry in 1:100 ) {
    x <- rnorm(smplSz)
    y <- a*x+b*x^2+c*x^3+rnorm(smplSz,sd=errEps)
    lm0 <- lm(y~1); lm1 <- lm(y~x); lm2 <- lm(y~x+I(x^2))
    lm3 <- lm(y~x+I(x^2)+I(x^3)); lm4 <- lm(y~x+I(x^2)+I(x^3)+I(x^4))
    aRes <- anova(lm0,lm1,lm2,lm3,lm4)
    dfTmp <- rbind(dfTmp,data.frame(n=smplSz,
                  term=c("Linear","Quadratic","Cubic","4th power"),pAnova=aRes[-1,"Pr(>F)"]))
  }
}
dfTmp[,"term"] <- factor(dfTmp[,"term"],c("Linear","Quadratic","Cubic","4th power"))
dfTmp[dfTmp[,"pAnova"]==0,"pAnova"] <- min(dfTmp[dfTmp[,"pAnova"]>0,"pAnova"])
ggplot(dfTmp,aes(x=factor(n),y=pAnova))+geom_boxplot()+theme_bw()+scale_y_log10()+
      geom_hline(yintercept=0.001,linetype=2)+facet_wrap(~term,scales="free",ncol=2)+
      xlab("Sample size, n")+ylab("ANOVA significance, p")+
      theme(axis.title = element_text(size=16),
            axis.text = element_text(size=16),
            strip.text=element_text(size=16))
```

STEP FUNCTION REGRESSION

- If we are allowed to introduce new variables (X^2, X^3, …) as powers of X, why not functions $f_i(X)$ in general?
- Select K points $c_1 \ldots, c_K$ that split the range of X into K+1 intervals and introduce "functions" (new variables) $C_0(X)=I(X<c_1)$; $C_1(X)=I(c_1<X<c_2)$, … where $I(\ )$ is an *indicator function* (1 when condition is true, 0 otherwise)
- Write linear model as (note that C_0 is linearly dependent on the remaining variables: $C_0=1-C_1-\ldots-C_K$)

$$y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \cdots + \beta_K C_K(x_i) + \varepsilon_i$$

  - Exactly one of C_i is non-zero for any given value x.
  - For *all* values $x < c_1$ (i.e. when $C_0 = 1$), we always predict β_0
  - For all values $c_i<x<c_{i+1}$ (i.e. $C_i=1$) we predict $\beta_0 + \beta_i$
  - Thus we simply predict mean X in each interval!
- This is but a fancy formulation for the following procedure:
  - Split domain of X into intervals, turning X into a categorical variable Z ("bin")
  - Z now has values (levels) "interval 1", "interval 2", …
  - Represent Z with dummy variables using standard one-hot encoding, fit linear model

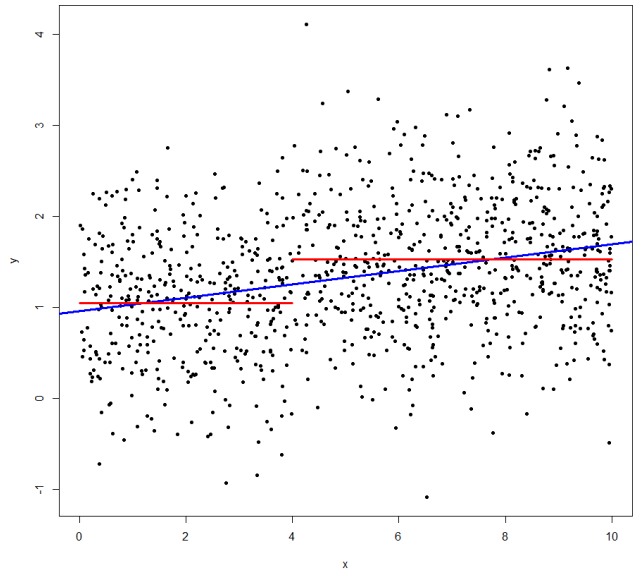


Q: is this related in any way to KNN?

STEP FUNCTION REGRESSION IN R

- As indicated in previous slide, all one needs to do is to convert X into a categorical variable and run a standard linear model (which will automatically build one-hot encoded dummy variables for a categorical predictor)

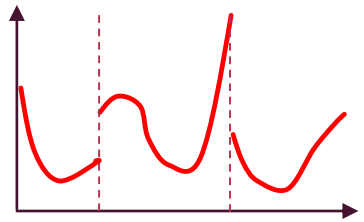
```
x=runif(1000,0,10) # generate example data
y=ifelse(x<4,1,1.5)+rnorm(1000,sd=0.7)
M1=lm(y~x)  # conventional LM
sqrt(mean((mean(y)-y)^2)) # fitting with mean(Y) alone
[1] 0.7468261
sqrt(mean((predict(M1)-y)^2)) # fitting with M1
[1] 0.7151541
Xcat <- cut(x,breaks=c(0,4,10)) # turn X to categorical!
Xcat[1:5]
[1] (0,4]  (0,4]  (0,4]  (4,10] (4,10]
Levels: (0,4] (4,10]
M2=lm(y~Xcat) # run step function regression against Xcat
coef(M2)
(Intercept)  Xcat(4,10]
  1.0414789   0.4813408
sqrt(mean((predict(M2)-y)^2)) # fitting with M2
[1] 0.70934
# plot results:
plot(x,y,cex=0.7,pch=19)
abline(M1,col="blue",lwd=3)
lines(c(0,4),rep(coef(M2)[1],2),col="red",lwd=3)
lines(c(4,10),rep(sum(coef(M2)),2),col="red",lwd=3)
```



Of course a much more interesting (and difficult) question is *where* to draw cut(s) if we do not know in advance (which is much more common). This is a *segmentation problem*, which requires different methods (e.g. regression trees: try `rpart(y~x)`)

REGRESSION SPLINES

- We can take it one (or few) step(s) further: now that we allow ourselves to *segment* the range of X, why do we have to limit the model to piece-wise constant? Can we fit linear (or polynomial) regression model in each interval?
  - Yes we can (at least we can try)! However, the "model" shown below does not look like a particularly good one…



  - We can get a more reasonable curve if require it to be *smooth* (i.e. continuous, and the derivatives to be continuous too)
  - Standard prescription: use polynomial of degree *d* in each segment and require the function itself and its derivatives up to d-1 to be continuous at each interval boundary c_i (also known as a *knot*). The fit built in this way is known as *spline*.
    - Polynomial of degree d has (d+1) coefficients; if we have K knots (i.e. K+1 intervals), then the total number of parameters is (K+1)(d+1)
    - At each knot we have d conditions that constrain the parameters, for the total of Kd constraints
    - The total number of free parameters of the model (model's degrees of freedom) is Kd+d+K+1-Kd =K+d+1.
    - For instance, cubic spline (i.e. using 3rd degree polynomials in each interval + smoothness condition) has K+4 parameters (degrees of freedom)

LINEAR MODEL FRAMEWORK AND SPLINES

- It turns out that the standard linear model framework can be used to fit splines efficiently as well!
- We do not need to fit individual polynomial model in each interval (how would we even satisfy the smoothness constraints?!)
- Instead, we can choose *spline basis functions* appropriately, e.g. for the cubic spline:

  $$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \cdots + \beta_{K+3} b_{K+3}(x_i) + \varepsilon_i$$ (note that we have to have K+4 degrees of freedom)

  where one possible choice for functions b( ) is:

  $b_1(x)=x$, $b_2(x)=x^2$, $b_3(x)=x^3$, and $b_{3+i}(x)$ = { $(x-c_i)^3$ if $x > c_i$, 0 otherwise }

  Note that the smoothness condition is automatically enforced at each knot c_i, so we can fit in the whole range of X!

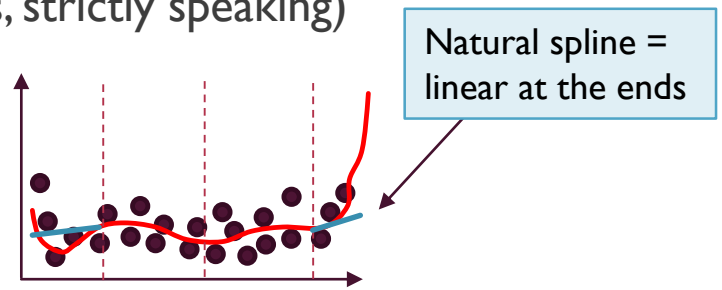
  Prescription: compute values of new predictors $b_1, \ldots, b_{K+d}$ for each datapoint x_i, (it's just some special variable transformation!), then run linear regression against new variables!!

- Of course, there is a function in R that does just that, automatically (examples will follow):

```
# will fit cubic spline with two knots at 3 and 5 in this example; need package splines
# function bs() will automatically compute transformation into basis splines; see help for more details
lm(Y~bs(X, knots=c(3,5) ), data=my.data)
```

NATURAL SPLINES

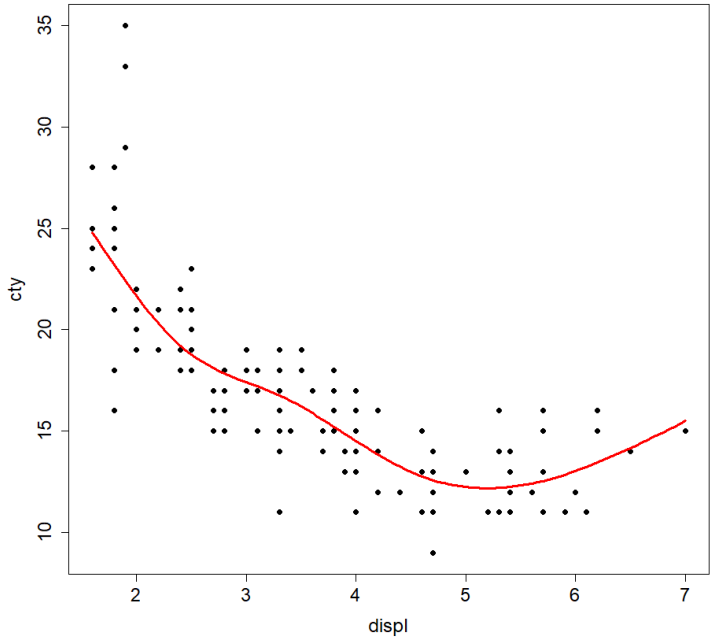
- The splines are well constrained and smooth *inside* the domain of X (at the knots, strictly speaking)
  - Nothing prevents a spline from exhibiting weird(er) behavior at the edges
- Solution: introduce additional constraints, e.g.:
  - Fit in the leftmost and the rightmost intervals must be *linear* (called **natural spline**)



- In R, fit linear model over a set of basis function, the transformation is automatically performed by function `ns()`:

```
library(ggplot2) # needed to get 'mpg' dataset
M<-lm(cty~ns(displ,df=4),data=mpg)
attr(ns(mpg$displ,df=4),"knots") # knots automatically chosen for df=4
25% 50% 75%
2.4 3.3 4.6
M
Call:
lm(formula = cty ~ ns(displ, df = 4), data = mpg)

Coefficients:
       (Intercept)  ns(displ, df = 4)1 ...
            24.794              -7.710 ...
plot(cty~displ,data=mpg,pch=19,cex.lab=1.5,cex.axis=1.5)
d<-seq(min(mpg$displ),max(mpg$displ),by=0.05)
lines(d,predict(M,newdata = data.frame(displ=d)),col="red",lwd=3)
```



SMOOTHING SPLINES

- Short of hand-picking knots or attempting some segmentation, what other options do we have?
- We want to fit some curve *g(x)*, such that the residual sum of squares $\sum_{i=1}^{n}(y_i - g(x_i))^2$ is minimized, of course, but we do not want *g(x)* to be too wiggly (otherwise, if *g(x)* is too flexible, we can even interpolate all data points!)
- Add a penalty! Objective function to minimize: $\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int [g''(x)]^2 dx$
- Interestingly enough, the requirement for the specific objective function shown above to reach its minimum constraints *g(x)* pretty strongly: it can be shown that *g(x)* must be a *natural spline* with knots at all unique values of $x_1, \ldots, x_n$, but with shrunken coefficients (not unlike coefficient shrinkage in ridge regression!)
- Aren't all unique values of $x_1, \ldots, x_n$ just too many knots??
  - Not really – the parameter λ controls and restricts the flexibility of the model! This is again similar to what we have seen with ridge regression: we are allowed to include many variables (=many degrees of freedom), but the penalty controls the variance of the model!
  - The smoothing parameter λ must be tuned via cross-validation
  - A very efficient computational strategy exists where LOO cross-validation error can be computed at the cost of a single fit
- In R: `smooth.spline()` in package stats.

LOCAL REGRESSION

- Stepwise regression, regression splines, natural splines: split range of X into intervals, fit a model in each interval
- KNN: for any point x we want to make a prediction for, take K nearest neighbors x_i in the training data, compute average y_i for those points.
- **Local regression**: essentially, does to KNN what splines do to stepwise regression (i.e. fit a more complex model instead of just taking the average). For any point *x* we want to make a prediction for:
  - Take K training data points that are the closest to *x*
  - Assign weights w_i to each data point, in such a way that those K closest points get non-zero weights and the closer the point to *x*, the larger w_i.; all other points get weight 0
  - Fit *weighted* least squares model which minimizes, in case of linear model, $\sum_{i=1}^{n} w_i \, (y_i - \beta_0 - \beta_1 x_i)^2$ (since all weights w_i outside of the closest K points are 0, we just fit weighted regression model on those K points). Higher order (weighted) polynomial model can be used too!
  - Predict for *x* as $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$, where $\hat{\beta}$ are the fitted coefficients
- Considerations when performing local regression: how many neighbors, K; how to define weights w_i (cross-validation?)
- Generalizes well to p multiple variables (use p-dimensional neighborhoods, or build a model that's global wrt some variables and local wrt the others)

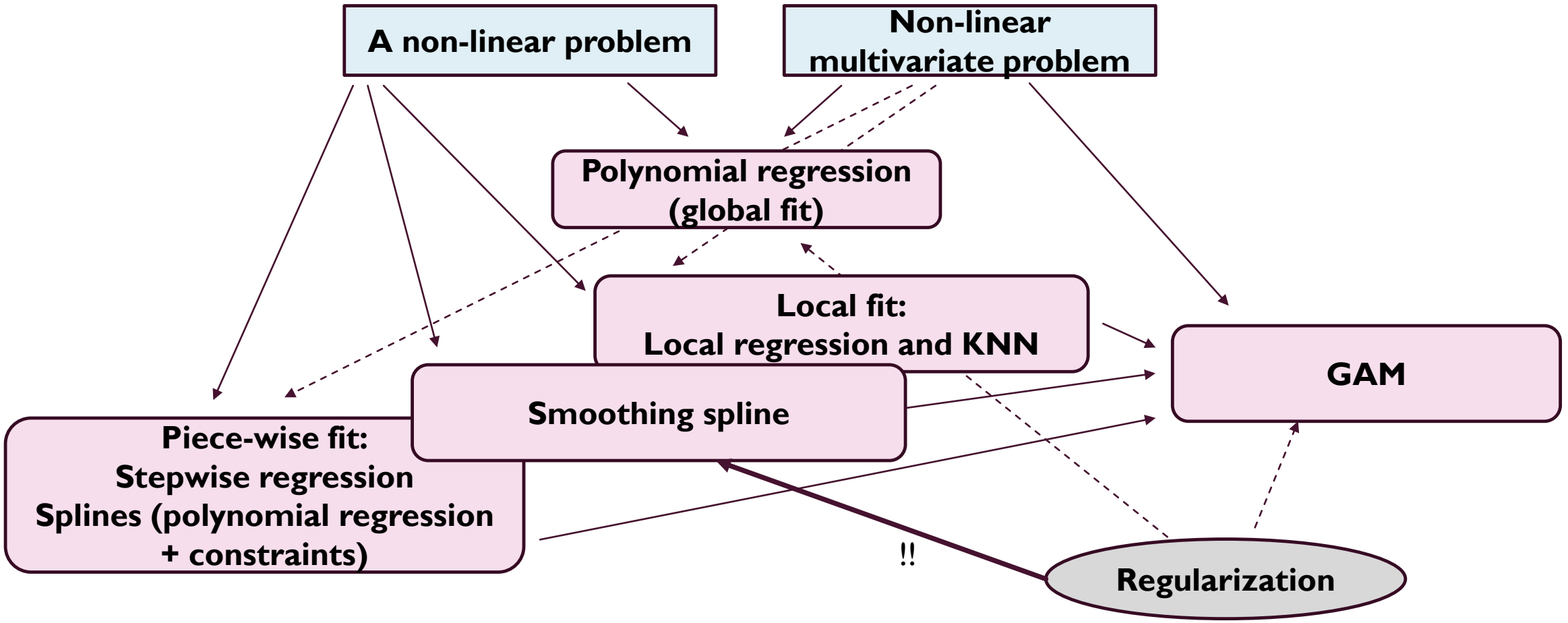
GENERALIZED ADDITIVE MODELS

- Multi-variable models; as the name suggests just a generalization of what we have already seen!
- For instance, we considered multi-variable polynomial regression models: $Y \sim X_1 + X_1^2 + X_2 + X_2^2 + \cdots$
  - These can be fitted in a straightforward way using standard linear model machinery (`lm()`) – we have done that
- In the general case a GAM with predictors $X_1, X_2, \ldots$ is represented as $y_i = \beta_0 + f_1(x_{1i}) + f_2(x_{2i}) + \cdots + \epsilon_i$
  - The only requirement is additivity with respect to the effects of different predictor variables
  - But those effects can be, generally speaking, any functions
  - For instance, we can fit using regression or smoothing splines for each (or some!) functions f
  - In R, this can be done using function `gam()` from package `gam`

PROS AND CONS OF GAMS

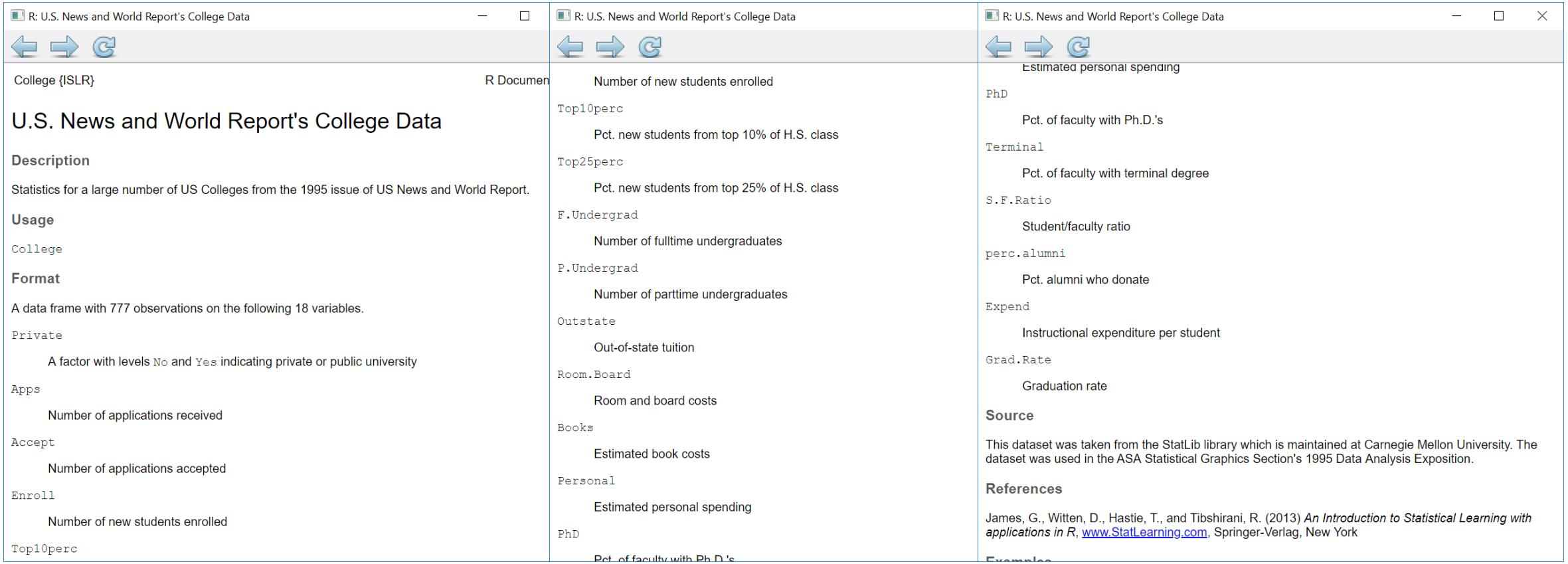
- We can model non-linear dependencies (beyond simple polynomial regression that is) on multiple predictor variables
- Instead of trying multiple explicit non-linear transformations on X_i, we can fit with less parametrized dependencies such as splines or local weighted regression
- Allow link functions (more on that when we discuss classification problem)
- Interpretable, as one can always examine each individual (fitted) function $f_i(x_i)$ in order to understand the effect of each predictor variable
- Can control the complexity of the model by explicitly specifying the number of (effective) degrees of freedom to use (e.g. with smoothing splines)
- Not well suited for assessing complex interactions between predictor variables
  - Can be *partially* cured by explicitly creating additional variables, e.g. $X_2X_5\ldots$
  - … or by using low-dimensional functions $f_{jk}(X_j, X_k)$ – for instance, we could use 2D local regression for the latter or a 2D spline…

SUMMARY


A non-linear problem
Non-linear multivariate problem
Polynomial regression (global fit)
Local fit: Local regression and KNN
GAM
Smoothing spline
Piece-wise fit: Stepwise regression Splines (polynomial regression + constraints)
!!
Regularization

DATASET FOR THIS WEEK

- Algae is too small and too noisy to see anything non-linear
- Let's turn our attention to College data from ISLR



R: U.S. News and World Report's College Data

College {ISLR} R Documen

U.S. News and World Report's College Data

Description

Statistics for a large number of US Colleges from the 1995 issue of US News and World Report.

Usage

```
College
```

Format

A data frame with 777 observations on the following 18 variables.

`Private`
: A factor with levels `No` and `Yes` indicating private or public university

`Apps`
: Number of applications received

`Accept`
: Number of applications accepted

`Enroll`
: Number of new students enrolled

`Top10perc`
: Pct. new students from top 10% of H.S. class

`Top25perc`
: Pct. new students from top 25% of H.S. class

`F.Undergrad`
: Number of fulltime undergraduates

`P.Undergrad`
: Number of parttime undergraduates

`Outstate`
: Out-of-state tuition

`Room.Board`
: Room and board costs

`Books`
: Estimated book costs

`Personal`
: Estimated personal spending

`PhD`
: Pct. of faculty with Ph.D.'s

`Terminal`
: Pct. of faculty with terminal degree

`S.F.Ratio`
: Student/faculty ratio

`perc.alumni`
: Pct. alumni who donate

`Expend`
: Instructional expenditure per student

`Grad.Rate`
: Graduation rate

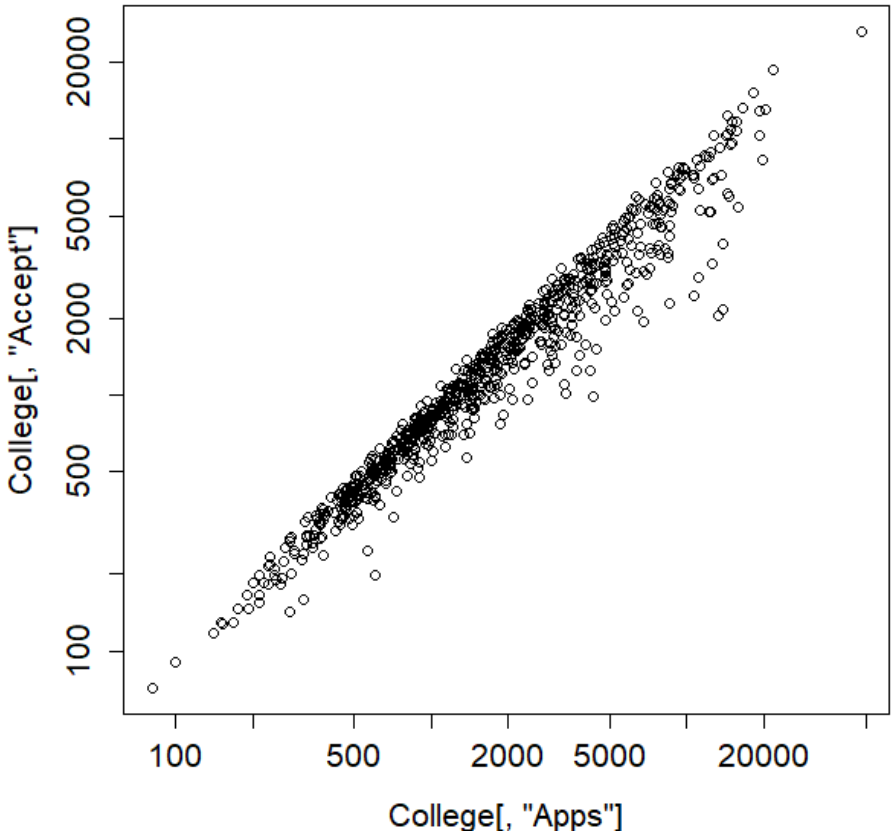
Source

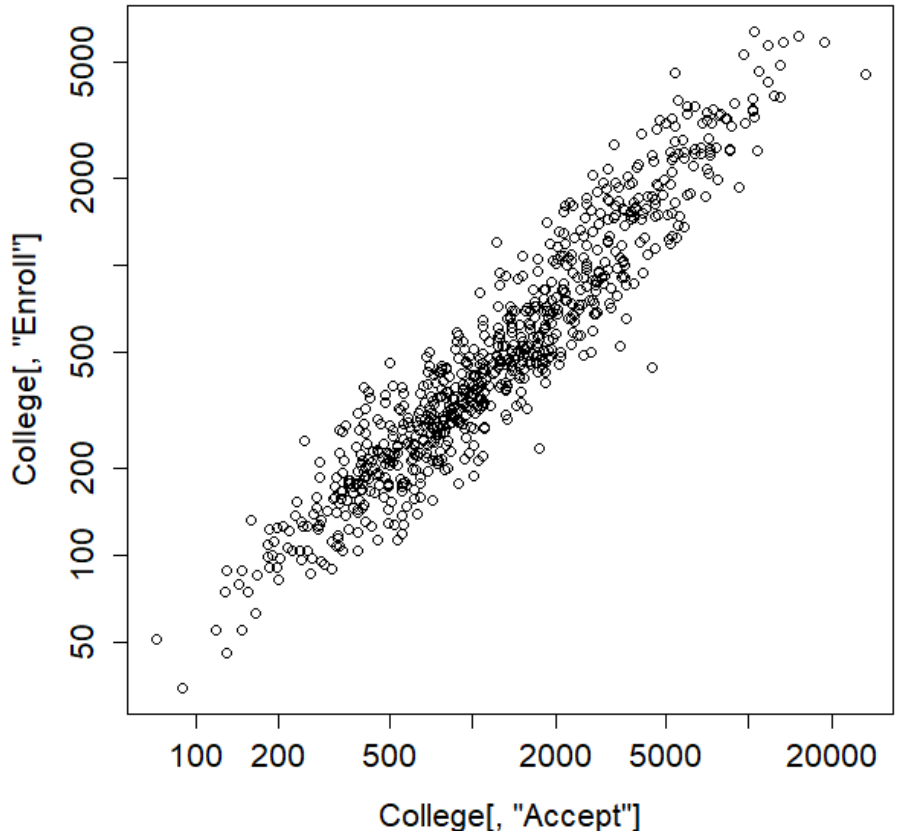
This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University. The dataset was used in the ASA Statistical Graphics Section's 1995 Data Analysis Exposition.

References

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013) *An Introduction to Statistical Learning with applications in R*, www.StatLearning.com, Springer-Verlag, New York

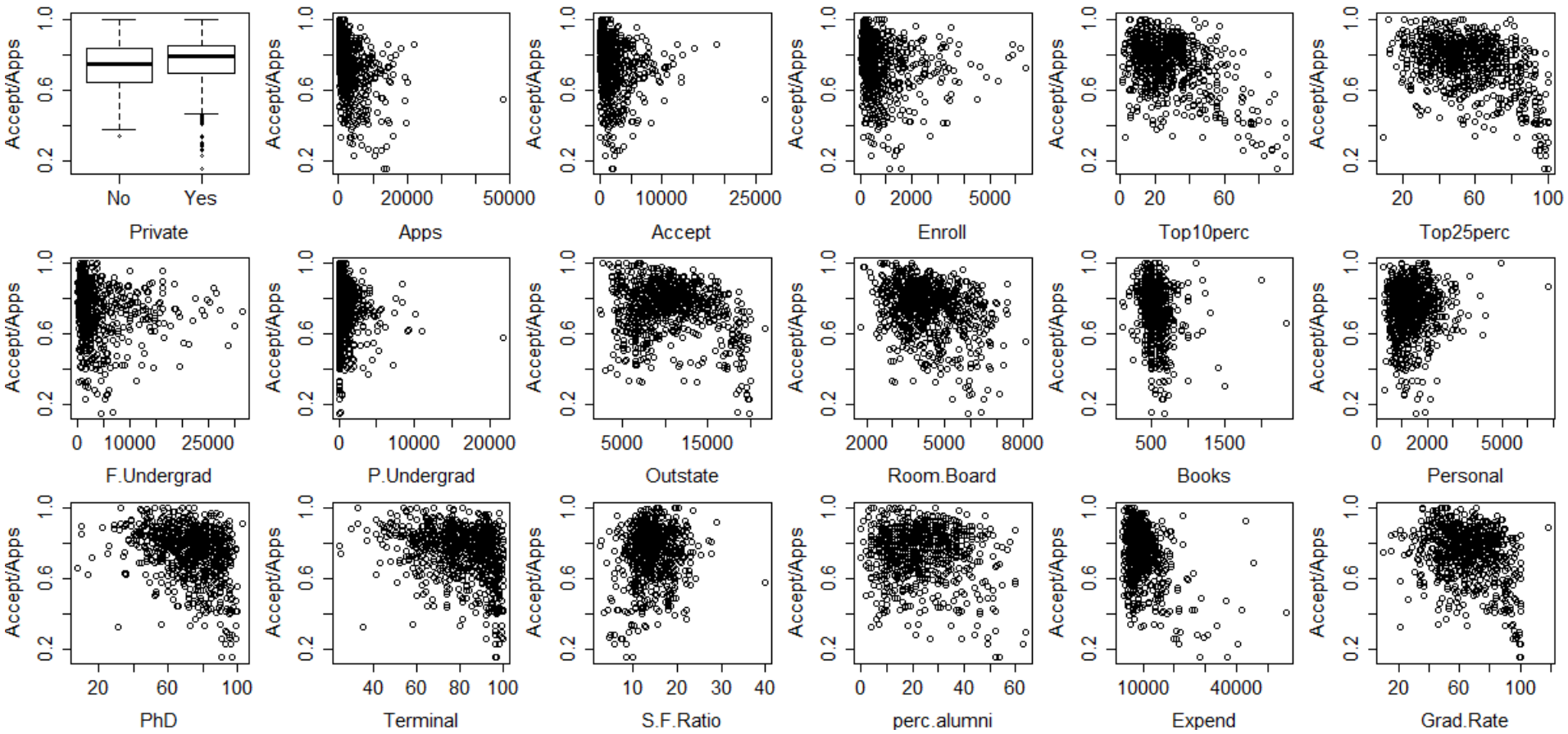
TRIVIA





- Sure, numbers of applications accepted and received are highly correlated, as are accepted and enrolled, but those are probably rather trivial and not very interesting relationships to "discover"

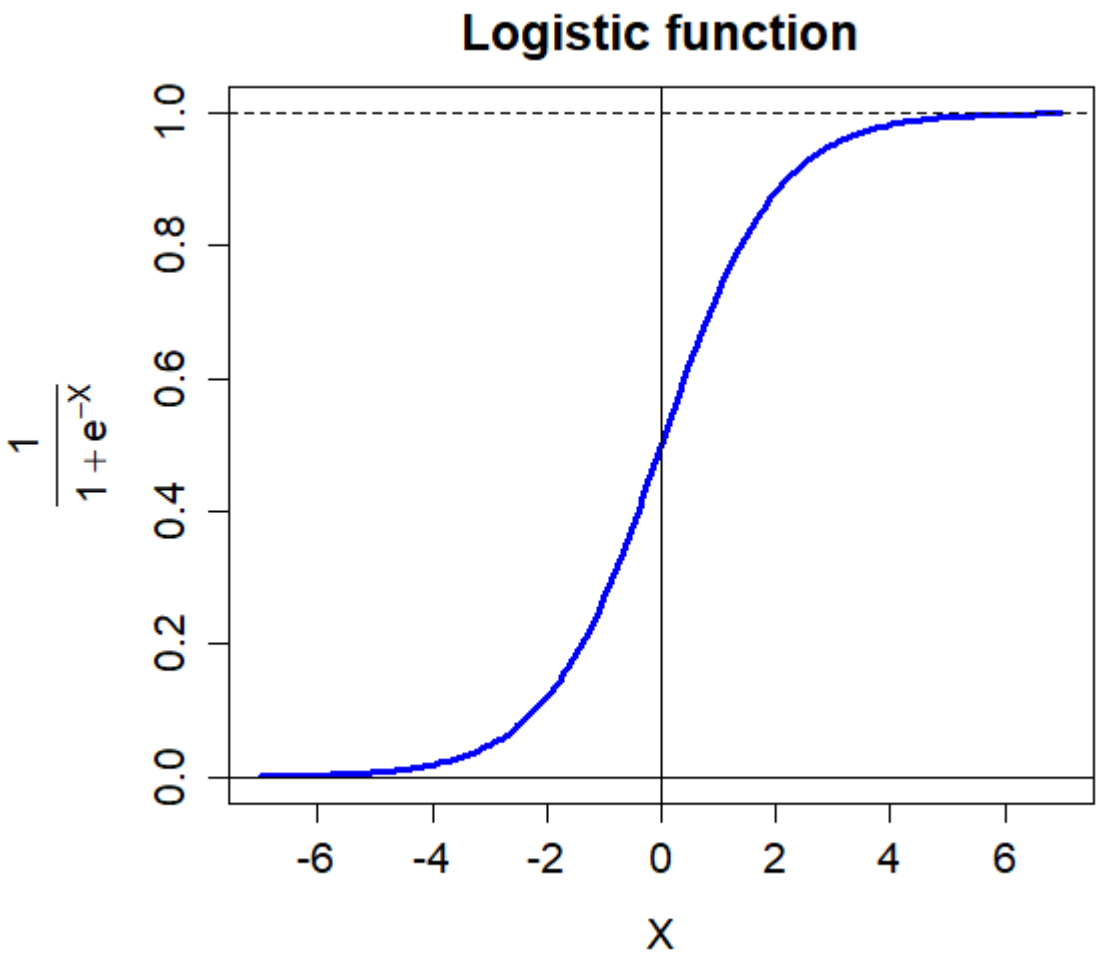
HOW ABOUT THE RATE OF ACCEPTANCE?



- Less obvious correlations, for some attributes perhaps nonlinear, distributions of the observations call for transformations for most of them

AGAIN ABOUT DATA TRANSFORMATIONS

- Model predictions within the outcome domain
  - E.g. fraction within [0,1], weight≥ 0, etc.
- Logistic function $y = \frac{1}{1+e^{-x}}$ maps $[-\infty, \infty]$ to [0,1]
- $Y = -\ln\left[\frac{\text{Apps}}{\text{Accept}} - 1\right]$ will be the transformed outcome
- Same transformation of "percentage" predictors…
  - After division by 100
  - And offsetting points at 0, 1 or beyond
  - Alternatively, cleaning them up
- …will decrease "bunching" near 100%
- The rest of the attributes will be log-transformed
  - Counts, dollars and ratio(s) except for categorical Private
  - Assuming multiplicative error

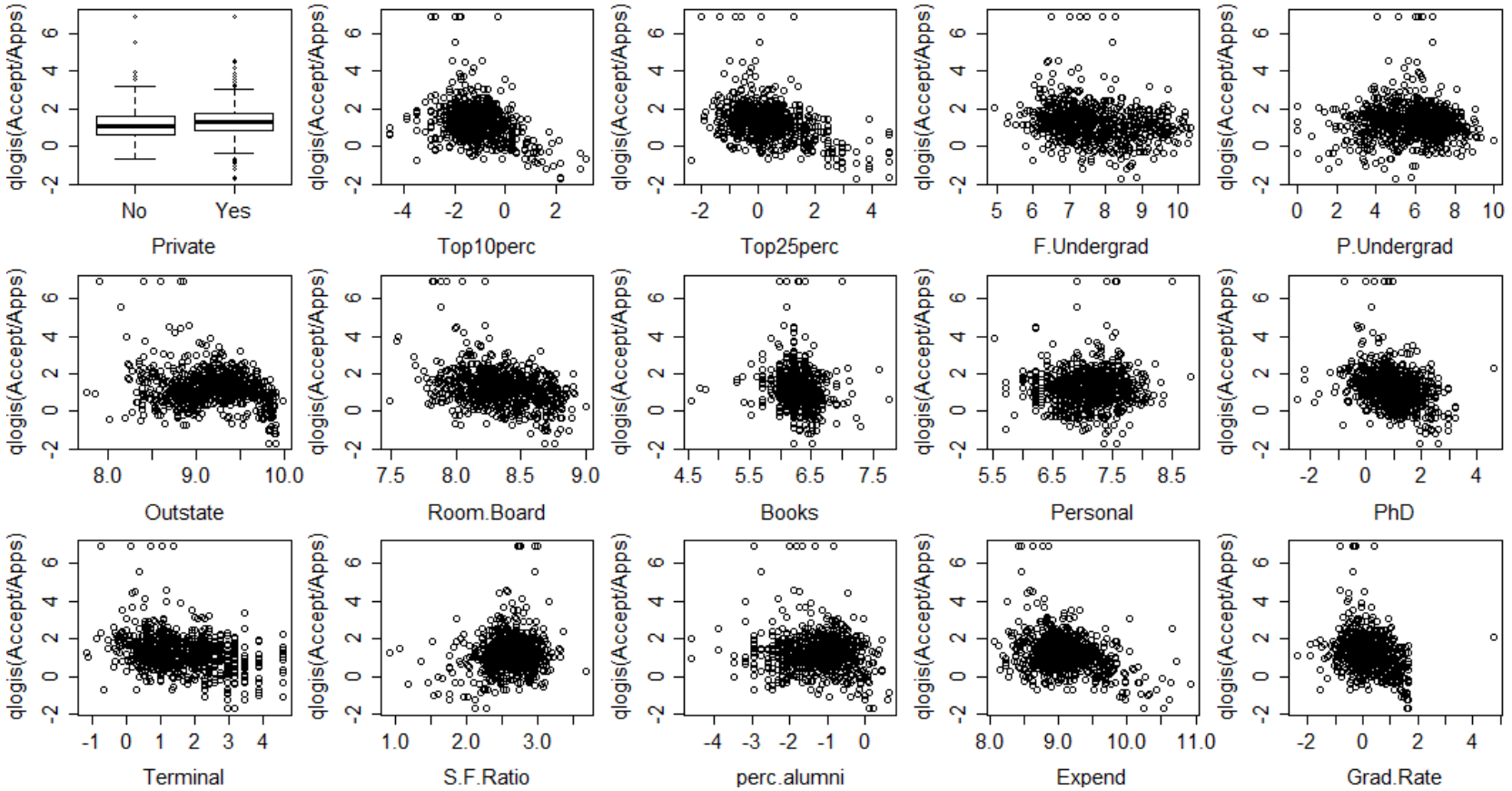


TRANSFORMATION NUANCES

```
> sort(College[,"Accept"]/College[,"Apps"],decreasing=TRUE)[1:10]
 [1] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
 [7] 0.9971165 0.9906542 0.9900990 0.9892473
> College[,"Y"] <- qlogis(0.999*College[,"Accept"]/College[,"Apps"])
>
> College[,"Top10perc"] <- qlogis(College[,"Top10perc"]/100)
> range(College[,"Top25perc"])
[1]   9 100
> College[,"Top25perc"] <- qlogis(College[,"Top25perc"]/101)
> range(College[,"PhD"])
[1]   8 103
> College[,"PhD"] <- qlogis(College[,"PhD"]/max(College[,"PhD"]+1))
> range(College[,"Terminal"])
[1]  24 100
> College[,"Terminal"] <- qlogis(College[,"Terminal"]/101)
> range(College[,"perc.alumni"])
[1]  0 64
> College[,"perc.alumni"] <- qlogis((1+College[,"perc.alumni"])/100)
> range(College[,"Grad.Rate"])
[1]  10 118
> College[,"Grad.Rate"] <- qlogis(College[,"Grad.Rate"]/max(College[,"Grad.Rate"]+1))
```

And log-transform: F.Undergrad, P.Undergrad, Outstate, Room.Board, Books, Personal, S.F.Ratio and Expend

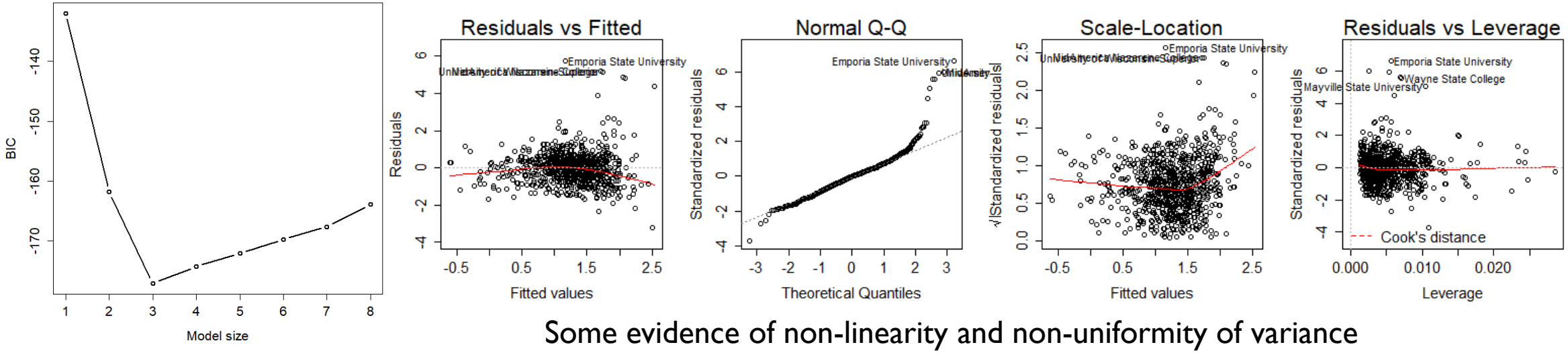
AFTER TRANSFORMATIONS



- Still a lot of variability, but now more evenly / symmetrically distributed observations and outcome that can assume any value on real line $\mathbb{R}^1$
- Better way to deal with those outliers (e.g. 118% graduation rate) is to track them down and clean them up

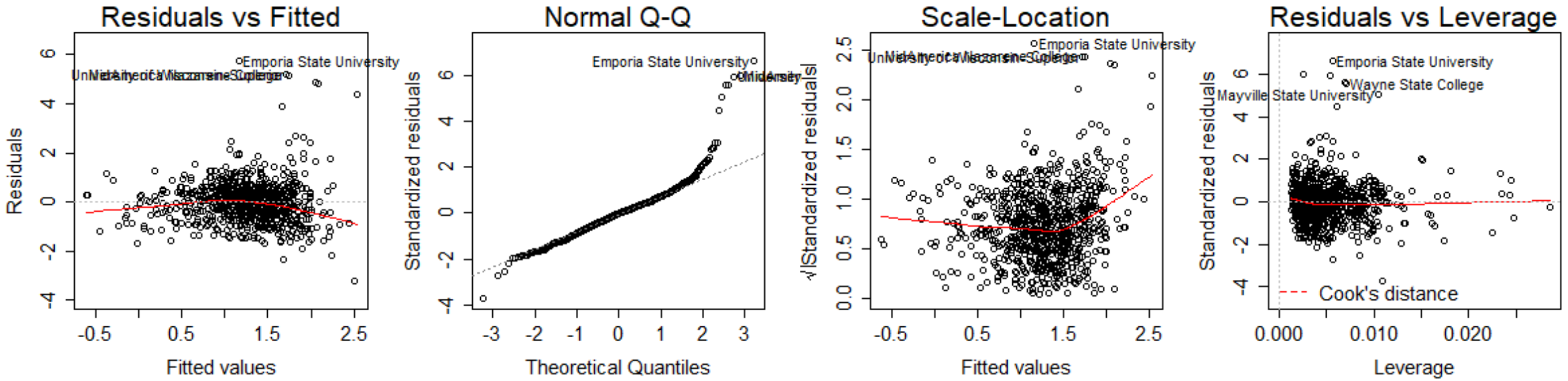
WHAT ABOUT LINEAR TERMS ONLY?

```
> rsSumm <- summary(regsubsets(Y~.,College[,!colnames(College)%in%c("Accept","Apps","Enroll")]))
> plot(rsSumm$bic,type="b",xlab="Model size",ylab="BIC")
> rsSumm$which[3,]
(Intercept)  PrivateYes    Top10perc    Top25perc F.Undergrad P.Undergrad     Outstate  Room.Board
       TRUE        FALSE        FALSE         TRUE         TRUE        FALSE        FALSE         TRUE
      Books     Personal          PhD     Terminal   S.F.Ratio perc.alumni       Expend   Grad.Rate
      FALSE        FALSE        FALSE        FALSE        FALSE        FALSE        FALSE        FALSE
> old.par <- par(mfrow=c(2,2))
> plot(lm(Y~Top25perc+F.Undergrad+Room.Board,College))
> par(old.par)
```



Some evidence of non-linearity and non-uniformity of variance

OBSERVATIONS WITH THE LARGEST RESIDUALS



- Some of the points resulting in the largest residuals are those where Accept == Apps:
  - Emporia State University, Mayville State University, MidAmerica Nazarene College, Southwest Baptist University, University of Wisconsin-Superior, Wayne State College
  - Could be data entry issue, could be truth…
- Excluding does not quite fix the diagnostic plots either, for now we proceed with the entire dataset

FIRST IMPRESSIONS

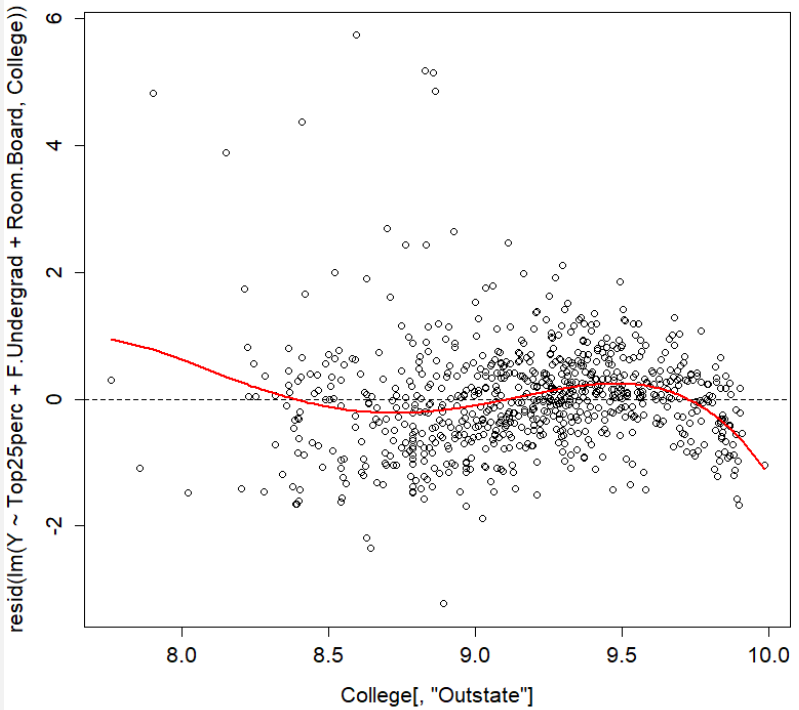
```
> summary(lm(Y~Top25perc+F.Undergrad+Room.Board,College))
Call:
lm(formula = Y ~ Top25perc + F.Undergrad + Room.Board, data = College)
Residuals:
    Min      1Q  Median      3Q     Max
-3.2304 -0.5101 -0.0034  0.3954  5.7418
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.93086    1.15940   8.566  < 2e-16 ***
Top25perc   -0.27434    0.03266  -8.399  < 2e-16 ***
F.Undergrad -0.14891    0.03167  -4.701 3.06e-06 ***
Room.Board  -0.89178    0.13272  -6.719 3.55e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.8711 on 773 degrees of freedom
Multiple R-squared:  0.2306,    Adjusted R-squared:  0.2276
F-statistic: 77.22 on 3 and 773 DF,  p-value: < 2.2e-16
```

- Universities with higher fraction of better high school graduates, larger number of full-time undergraduates and higher cost of room and board are more selective in their admission process.
- What about non-linear effects?

HERE IS ONE FOR OUT-OF-STATE TUITION

```
> summary(lm(Y~Top25perc+F.Undergrad+Room.Board+poly(Outstate,4),College))
Call:
lm(formula = Y ~ Top25perc + F.Undergrad + Room.Board + poly(Outstate,
    4), data = College)
…
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         11.32583    1.34640   8.412  < 2e-16 ***
Top25perc           -0.24087    0.03618  -6.658 5.27e-11 ***
F.Undergrad         -0.10899    0.03332  -3.271  0.00112 **
Room.Board          -1.09663    0.16259  -6.745 3.01e-11 ***
poly(Outstate, 4)1   1.50869    1.27062   1.187  0.23545
poly(Outstate, 4)2 -1.15074    0.89589  -1.284  0.19937
poly(Outstate, 4)3 -5.46075    0.87054  -6.273 5.91e-10 ***
poly(Outstate, 4)4 -2.32028    0.85196  -2.723  0.00661 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8471 on 769 degrees of freedom
Multiple R-squared:  0.2762,    Adjusted R-squared:  0.2696
F-statistic: 41.92 on 7 and 769 DF,  p-value: < 2.2e-16
```



HOW DID WE FIND IT?

- By adding 4-th degree polynomial terms – poly(…,4) – for each predictor one at a time to the model proposed by regsubsets…
- …and using anova() to compare fits from those two models
- The one with the most significant improvement of the fit – the one adding orthogonal polynomials up to the 4-th degree for Outstate – was chosen
- Of course, there are many alternative strategies conceivable and caveats to consider:
- Do we know anything about the problem to guide us in our variable selection?
- Do we use higher degree (e.g. 4) and use up more degrees of freedom, or lower power (e.g. 2) at the risk of missing non-linearities of higher order than quadratic?
- Do we start with exploring non-linearities as higher powers of linear terms or interactions between them
  - $Y=X_1+X_2+X_1*X_2$ is also non-linear in the space of (X_1,X_2)
- Does inclusion of non-linear terms render previously included ones less impactful?
- Would the most significant non-linear term change for the dataset without "outliers"?

ANOVA ON NESTED MODELS

- Unlike adjusted R^2, AIC/BIC, etc. ANOVA on two (or more) **nested** models yields p-value (significance)
  - For the improvement of model fit with the increase in model complexity (adding predictors or higher powers thereof)
- Two models are nested if the simpler one (aka "reduced") is a special case of the more complex one (aka "full") from setting some of the β's for the "full" model to zero:
  - Nested: y~a and y~a+b (same if $\beta_b=0$)
  - **Not** nested: y~a+b and y~a+c (which one is more complex??)
  - Nested: y~a and $y\sim a+a^2+a^3$ (same if $\beta_2=\beta_3=0$)
  - **Not** nested: y~a+b+a:b and $y\sim a+b+a^2+b^2$ (cannot set β's in only one model to zero to obtain the other one)
- Why do they **have** to be nested?? Because the theory about the distributional properties of the fit improvement is available only under "nested" assumption (as well as normality/homogeneity/independence of the residuals)
  - I.e. for not nested model there is no theory to estimate statistical significance (p-value) of the fit improvement
- ANOVA test on two (or more models) does **NOT** tell us which one is "true" model
  - It only compares the fit improvement to what is expected under the null of no effect for that term for nested model
  - "It is easy to argue that statistics is about…nothing!" [G.E.Dallal, LHSP]

ANOVA ON NESTED MODELS: EXAMPLES

```
> anova(lm(Y~Top25perc+F.Undergrad+Room.Board,College),
        lm(Y~Top25perc+F.Undergrad+Room.Board+poly(PhD,4),College))
Model 1: Y ~ Top25perc + F.Undergrad + Room.Board
Model 2: Y ~ Top25perc + F.Undergrad + Room.Board + poly(PhD, 4)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1    773 586.62
2    769 586.01  4   0.61194 0.2008  0.938
> anova(lm(Y~Top25perc+F.Undergrad+Room.Board,College),
        lm(Y~Top25perc+F.Undergrad+Room.Board+poly(Outstate,4),College))
Model 1: Y ~ Top25perc + F.Undergrad + Room.Board
Model 2: Y ~ Top25perc + F.Undergrad + Room.Board + poly(Outstate, 4)
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    773 586.62
2    769 551.84  4    34.781 12.117 1.479e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
```

Which term improves the model fit more "significantly"?

Output lightly edited for compactness

- Easier to readily assess the impact of adding several terms (e.g. polynomial or categorical variables)

ANOVA ON NESTED MODELS: EXAMPLES

```
> summary(lm(Y~Top25perc+F.Undergrad+Room.Board+poly(Outstate,4),College))
…
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         11.32583    1.34640   8.412  < 2e-16 ***
Top25perc           -0.24087    0.03618  -6.658 5.27e-11 ***
F.Undergrad         -0.10899    0.03332  -3.271  0.00112 **
Room.Board          -1.09663    0.16259  -6.745 3.01e-11 ***
poly(Outstate, 4)1   1.50869    1.27062   1.187  0.23545
poly(Outstate, 4)2  -1.15074    0.89589  -1.284  0.19937
poly(Outstate, 4)3  -5.46075    0.87054  -6.273 5.91e-10 ***
poly(Outstate, 4)4  -2.32028    0.85196  -2.723  0.00661 **
…
Residual standard error: 0.8471 on 769 degrees of freedom
Multiple R-squared:  0.2762,    Adjusted R-squared:  0.2696
F-statistic: 41.92 on 7 and 769 DF,  p-value: < 2.2e-16
>
```

- Evaluation of the same model with summary requires reasoning about the effects of multiple polynomial terms

ANOVA ON NESTED MODELS: EXAMPLES

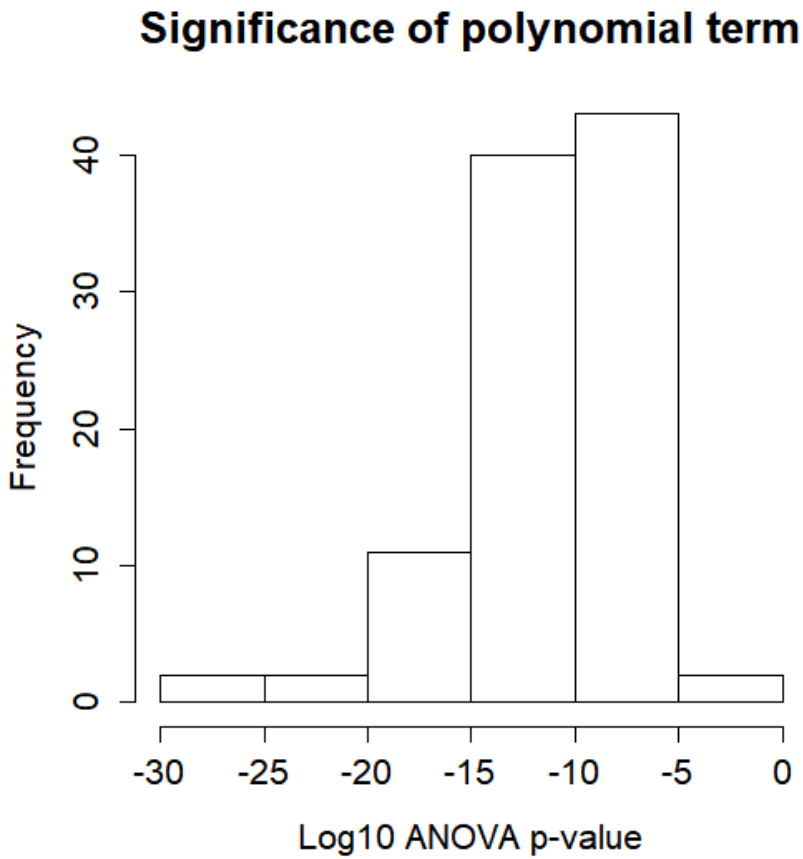
```
> anova(lm(Y~Top25perc+F.Undergrad+Room.Board,College),
        lm(Y~Top25perc+F.Undergrad+Room.Board+poly(PhD,4),College),test="Chisq")
Model 1: Y ~ Top25perc + F.Undergrad + Room.Board
Model 2: Y ~ Top25perc + F.Undergrad + Room.Board + poly(PhD, 4)
  Res.Df    RSS Df Sum of Sq Pr(>Chi)
1    773 586.62
2    769 586.01  4   0.61194    0.938
> anova(lm(Y~Top25perc+F.Undergrad+Room.Board,College),
        lm(Y~Top25perc+F.Undergrad+Room.Board+poly(Outstate,4),College),test="Chisq")
Model 1: Y ~ Top25perc + F.Undergrad + Room.Board
Model 2: Y ~ Top25perc + F.Undergrad + Room.Board + poly(Outstate, 4)
  Res.Df    RSS Df Sum of Sq  Pr(>Chi)
1    773 586.62
2    769 551.84  4    34.781 7.536e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
```

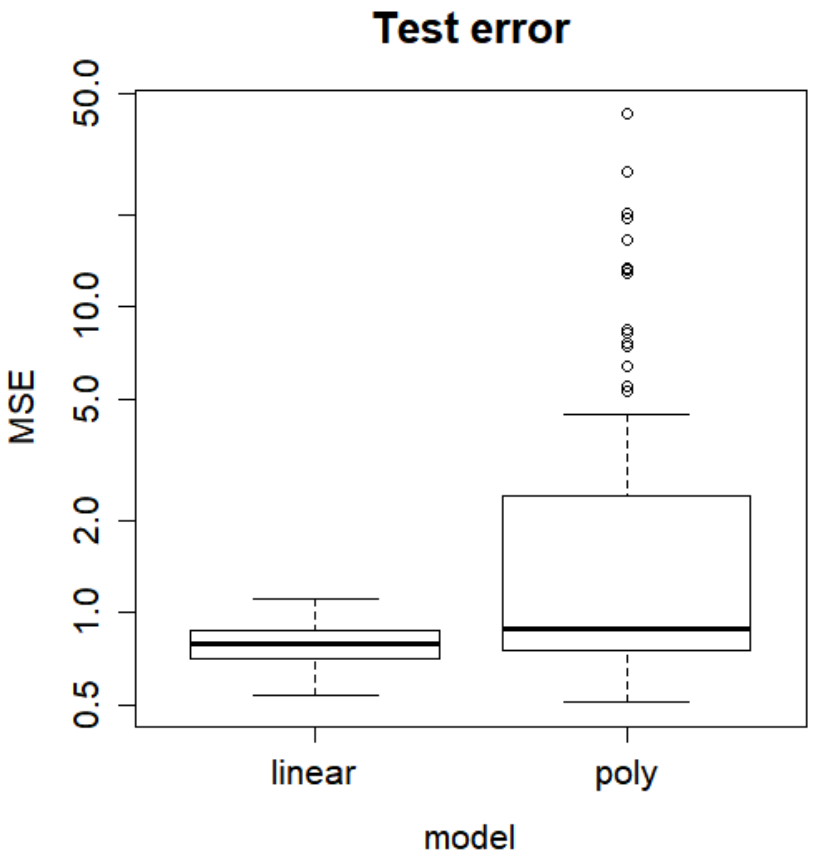
Different test, slightly different p-values, same conclusions re:significance

Output lightly edited for compactness

- You may also encounter ANOVA comparing models by their likelihood ratio that relies on χ^2 test to estimate the significance (in some settings it is a default / preferred test)

TEST ERROR WITH AND WITHOUT POLYNOMIAL TERM

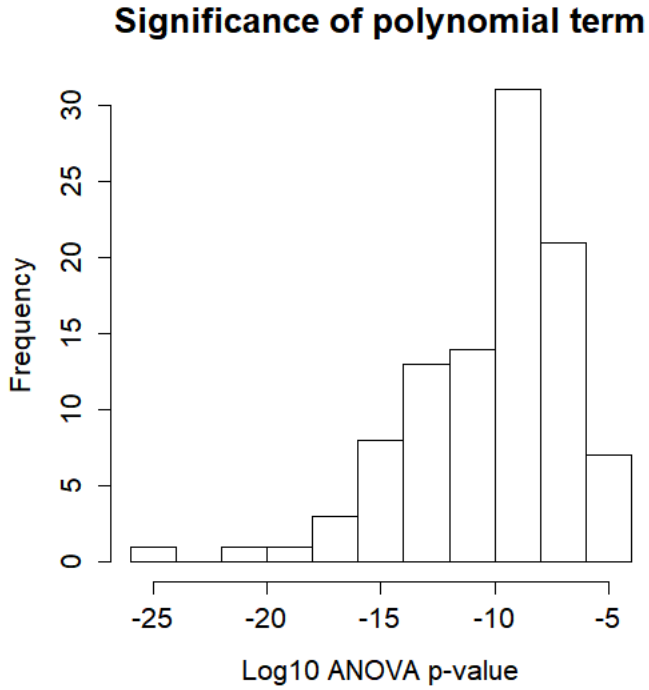


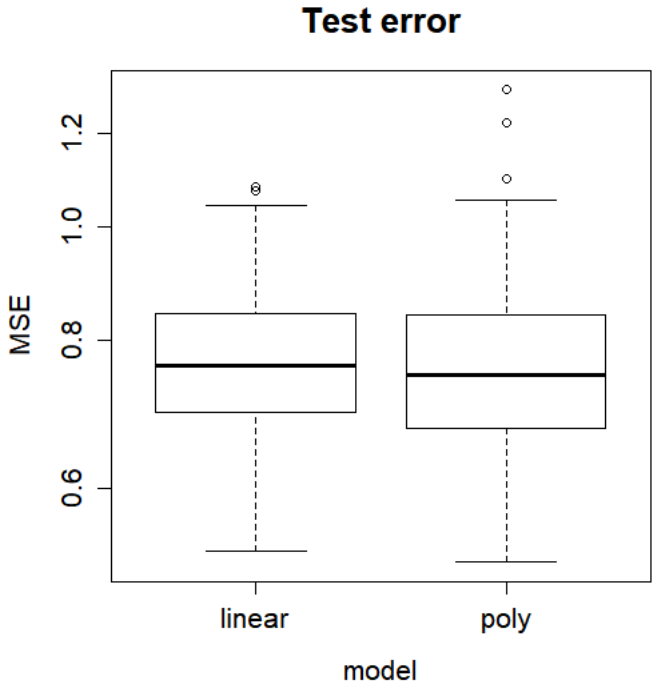


- Polynomial term is highly significant by ANOVA, but its inclusion in the model results in higher and highly variable test error – notice log scale for test error boxplot! Why would be that??

FOLLOW-UP NOTES

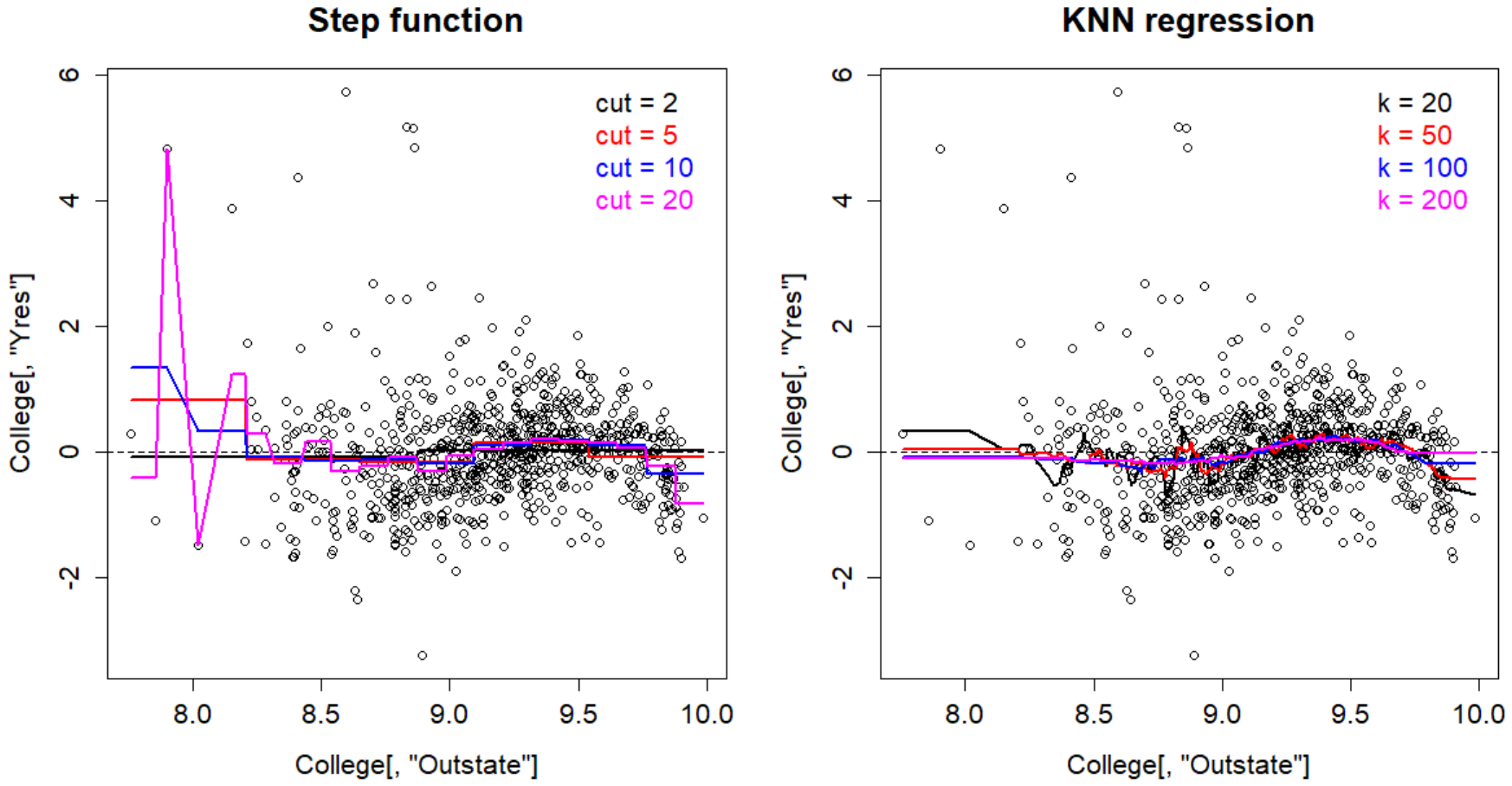
- Check observed data vs. training and test predictions
- Notice that one observation consistently yields strikingly high residual in test data
- Find out that it is the one with Grad.Rate=118% and obtain more expected results upon excluding it:





Still, the improvement in test error is unremarkable despite high significance of the polynomial term by anova on training data

STEP FUNCTIONS FOR COLLEGE DATASET: EXAMPLE



- Let us use Outstate to model residuals from "the best" linear fit

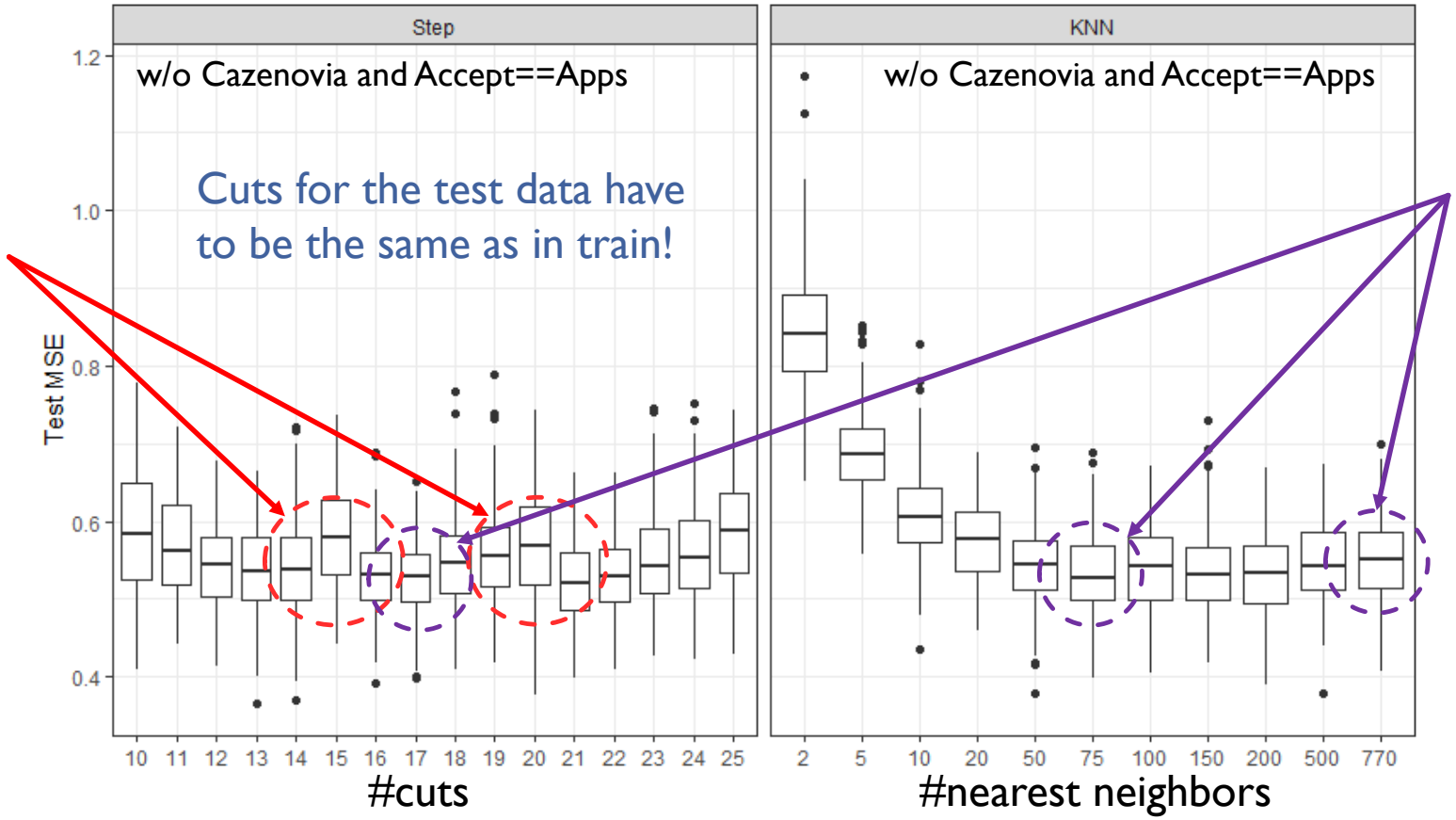
CODE FOR THE PREVIOUS PLOTS

```
College[,"Yres"] <- resid(lm(Y~Top25perc+F.Undergrad+Room.Board,College))
# Step:
plot(College[,"Outstate"],College[,"Yres"],main="Step function");abline(h=0,lty=2)
for ( nCut in c(2,5,10,20) ) {
  lmTmp <- lm(as.formula(paste0("Yres~cut(Outstate,",nCut,")")),College)
  points(College[order(College[,"Outstate"]),"Outstate"],
         predict(lmTmp)[order(College[,"Outstate"])],type="l",lwd=2,
         col=c("2"=1,"5"=2,"10"=4,"20"=6)[as.character(nCut)])
}
legend("topright",paste("cut =",c(2,5,10,20)),text.col=c(1,2,4,6),bty="n")
# KNN:
plot(College[,"Outstate"],College[,"Yres"],main="KNN regression");abline(h=0,lty=2)
for ( kTmp in c(20,50,100,200) ) {
  knnRegTmp <- knn.reg(matrix(College[,"Outstate"]),
                       matrix(College[,"Outstate"]),College[,"Yres"],k=kTmp)
  points(College[order(College[,"Outstate"]),"Outstate"],
         knnRegTmp$pred[order(College[,"Outstate"])],lwd=2,
         col=c("20"=1,"50"=2,"100"=4,"200"=6)[as.character(kTmp)],type="l")
}
legend("topright",paste("k =",c(20,50,100,200)),text.col=c(1,2,4,6),bty="n")
```

HOW TO COMPARE AND CHOOSE?

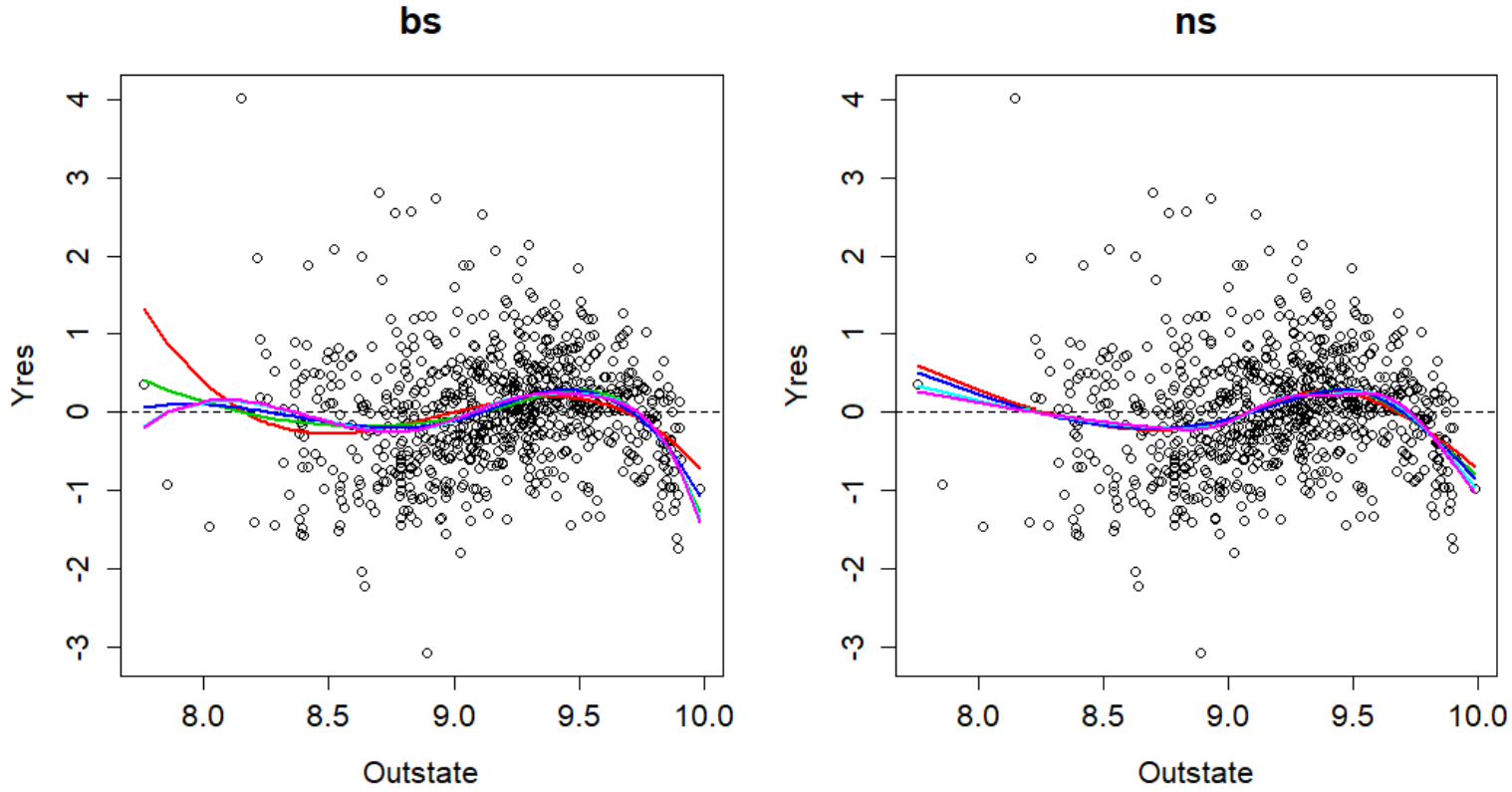
- ***Can we use ANOVA? Are these models nested?***
- ***What about comparing test error over the range of model complexity?***

Small changes in the number of cuts can noticeably affect test error



The lowest test error from step function is comparable to that of KNN and predicting to constant train average

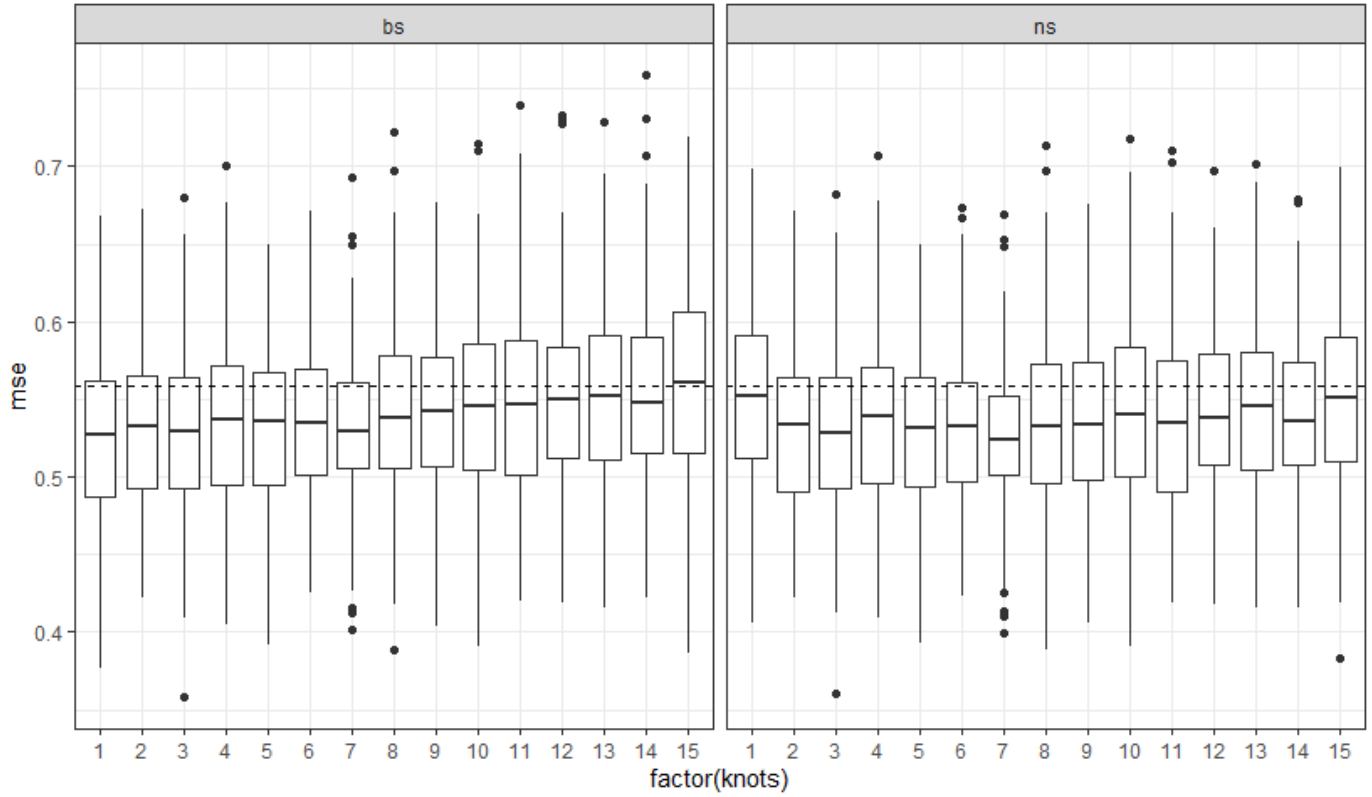
B-SPLINE AND NATURAL SPLINE FOR COLLEGE DATA: EXAMPLES



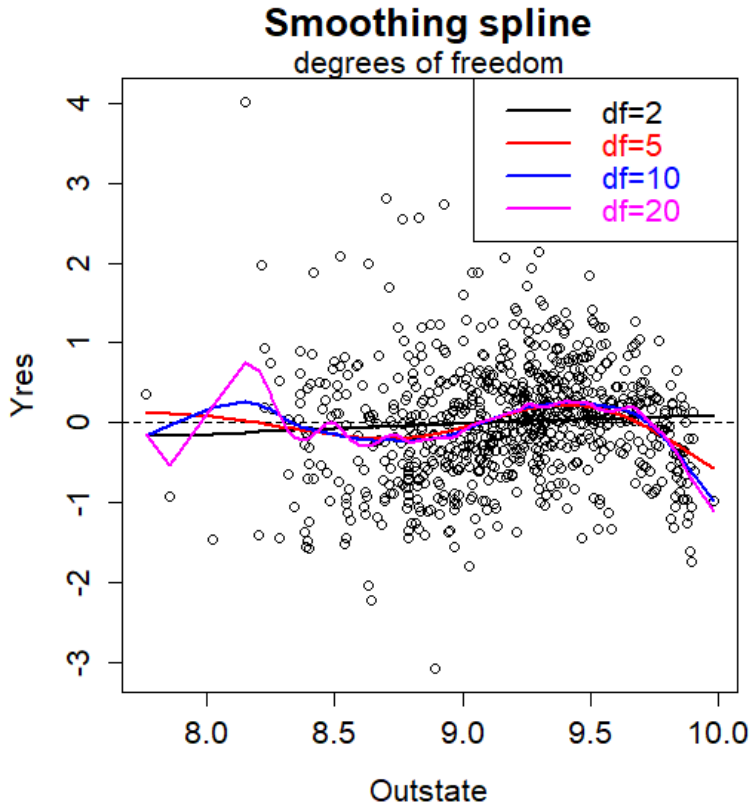
CODE FOR THE PREVIOUS PLOTS

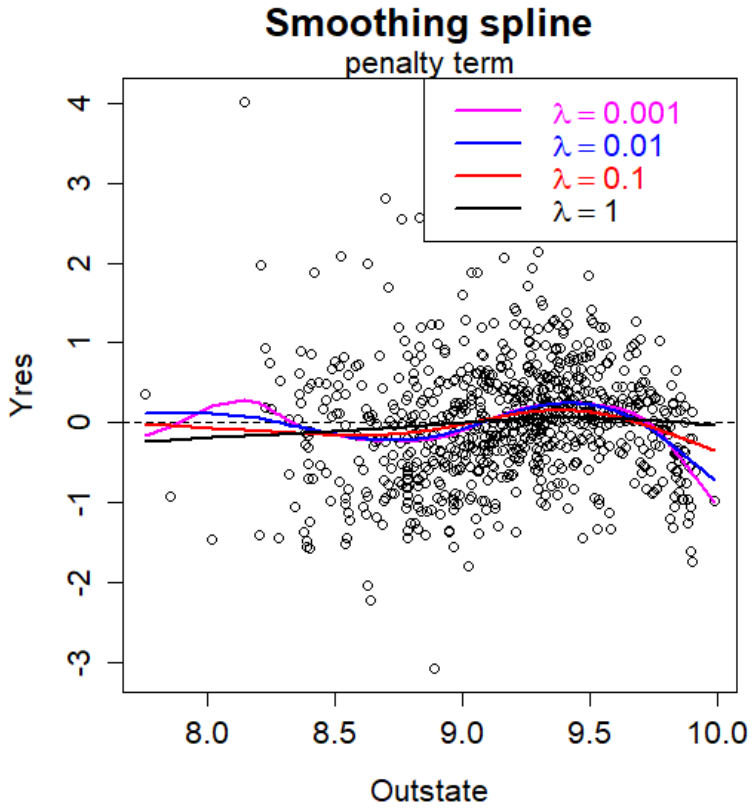
```
tmpDat <- CollegeWoAcceptAllCazenovia
tmpDat[,"Yres"] <- resid(lm(Y~Top25perc+F.Undergrad+Room.Board,tmpDat))
old.par <- par(mfrow=c(1,2),ps=16)
for ( tTmp in c("b","n") ) {
  plot(tmpDat[,"Outstate"],tmpDat[,"Yres"],xlab="Outstate",ylab="Yres",main=paste0(tTmp,"s"))
  abline(h=0,lty=2)
  for ( tmpKnots in 3:7 ) {
    if( tTmp == "b" ) {
      lmTmp <- lm(Yres~bs(Outstate,df=tmpKnots),data=tmpDat)
    }
    if( tTmp == "n" ) {
      lmTmp <- lm(Yres~ns(Outstate,df=tmpKnots),data=tmpDat)
    }
    points(sort(tmpDat[,"Outstate"]),
           predict(lmTmp)[order(tmpDat[,"Outstate"])],type="l",col=tmpKnots-1,lwd=2)
  }
}
par(old.par)
```

REGRESSION SPLINES FOR COLLEGE DATASET: BOOTSTRAP



SMOOTHING SPLINE FOR COLLEGE DATASET: EXAMPLE

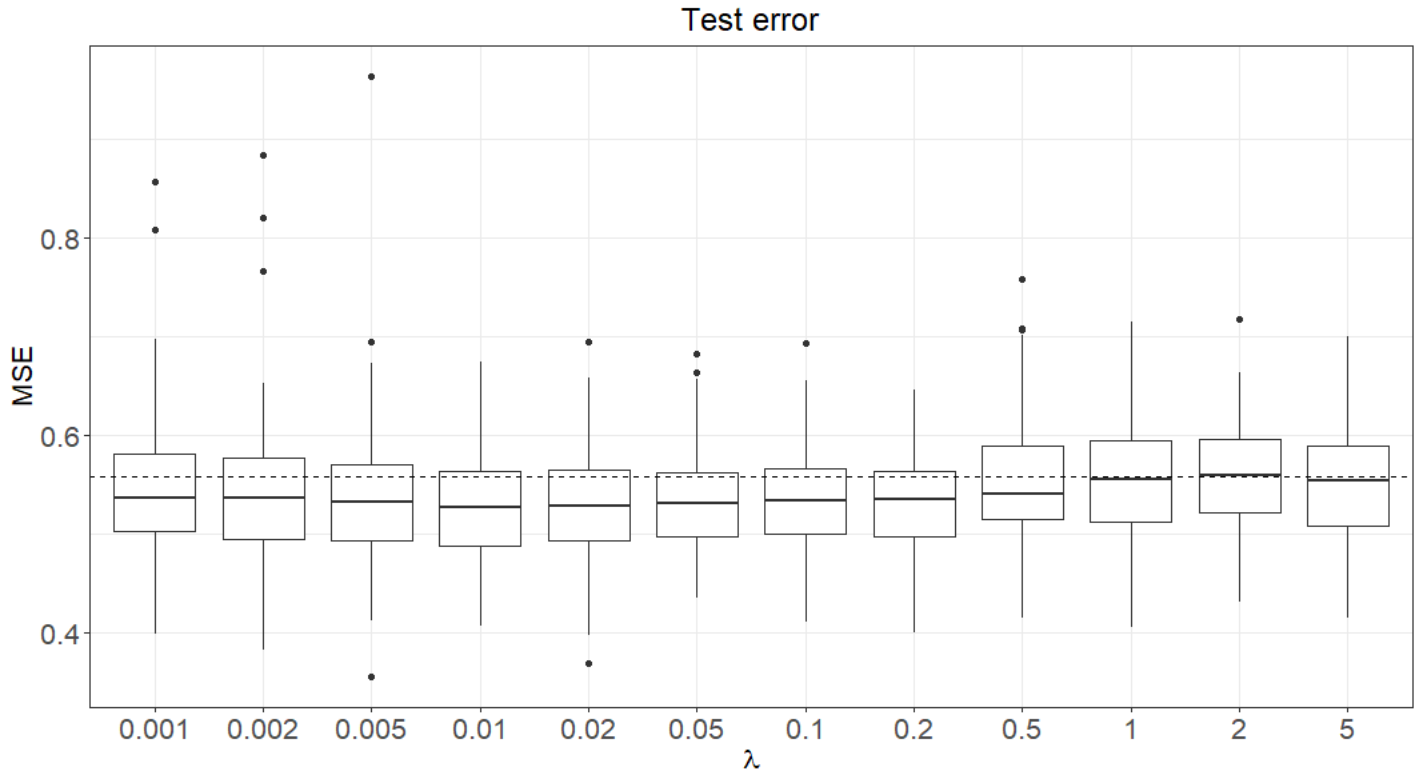




CODE FOR THE PLOTS ABOVE

```
tmpDat <- CollegeWoAcceptAllCazenovia
tmpDat[,"Yres"] <- resid(lm(Y~Top25perc+F.Undergrad+Room.Board,tmpDat))
old.par <- par(mfrow=c(1,2),ps=16)
plot(tmpDat[,"Outstate"],tmpDat[,"Yres"],xlab="Outstate",ylab="Yres",main="Smoothing spline")
mtext("degrees of freedom"); abline(h=0,lty=2)
clrsTmp <- c("black","red","blue","magenta"); names(clrsTmp) <- c("2","5","10","20")
for ( dfTmp in c(2,5,10,20) ) {
  ssDefaultTmp <- smooth.spline(tmpDat[,"Outstate"],tmpDat[,"Yres"],df=dfTmp)
  lines(ssDefaultTmp,col=clrsTmp[as.character(dfTmp)],lwd=2)
}
legend("topright",paste0("df=",c(2,5,10,20)),lwd=2,col=clrsTmp,text.col=clrsTmp,lty=1)
plot(tmpDat[,"Outstate"],tmpDat[,"Yres"],xlab="Outstate",ylab="Yres",main="Smoothing spline")
mtext("penalty term"); abline(h=0,lty=2)
names(clrsTmp) <- c("1","0.1","0.01","0.001")
for ( lambdaTmp in c(0.001,0.01,0.1,1) ) {
  ssDefaultTmp <- smooth.spline(tmpDat[,"Outstate"],tmpDat[,"Yres"],lambda=lambdaTmp)
  lines(ssDefaultTmp,col=clrsTmp[as.character(lambdaTmp)],lwd=2)
}
legend("topright",parse(text=paste0("lambda==",c(0.001,0.01,0.1,1))),lwd=2,col=rev(clrsTmp),text.
col=rev(clrsTmp),lty=1)
par(old.par)
```

TEST ERROR FOR SMOOTHING SPLINE: COLLEGE DATASET

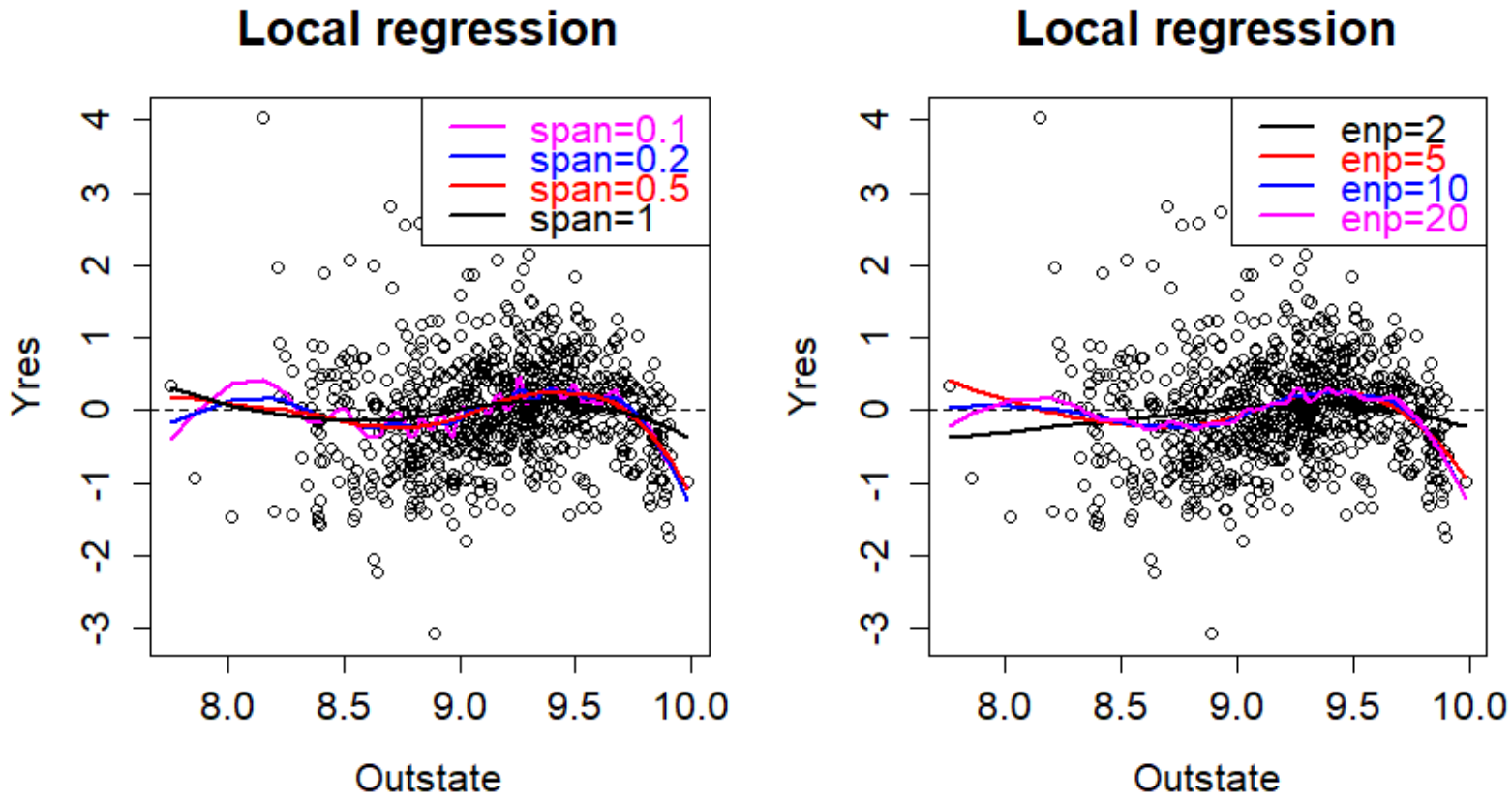


- For large lambda the test error is close to the variance of the outcome

CODE FOR THE ABOVE PLOT

```
nTries <- 200
dfTmp <- NULL
for ( lambdaTmp in c(0.001,0.002,0.005,0.01,0.02,0.05,0.1,0.2,0.5,1.0,2.0,5.0) ) {
  for ( iTry in 1:nTries ) {
    trainIdx <- sample(nrow(tmpDat),nrow(tmpDat),replace = TRUE)
    trainDat <- tmpDat[trainIdx,]
    testDat <- tmpDat[-trainIdx,]
    ssTrainTmp <- smooth.spline(trainDat[,"Outstate"],trainDat[,"Yres"],lambda=lambdaTmp)
    ssTestPred <- predict(ssTrainTmp,testDat[,"Outstate"])
    testMSE <- mean((testDat[,"Yres"]-ssTestPred[["y"]])^2)
    dfTmp <- rbind(dfTmp,data.frame(lambda=lambdaTmp,mse=testMSE))
  }
}
ggplot(dfTmp,aes(x=factor(lambda),y=mse))+geom_boxplot()+theme_bw()+
xlab(expression(lambda))+ylab("MSE")+ggtitle("Test error")+
theme(plot.title = element_text(hjust=0.5,size=18),
axis.text = element_text(size=16),axis.title = element_text(size=16))+
geom_hline(yintercept = var(tmpDat[,"Yres"]),lty=2)
```

LOESS MODEL FOR COLLEGE DATASET: EXAMPLE

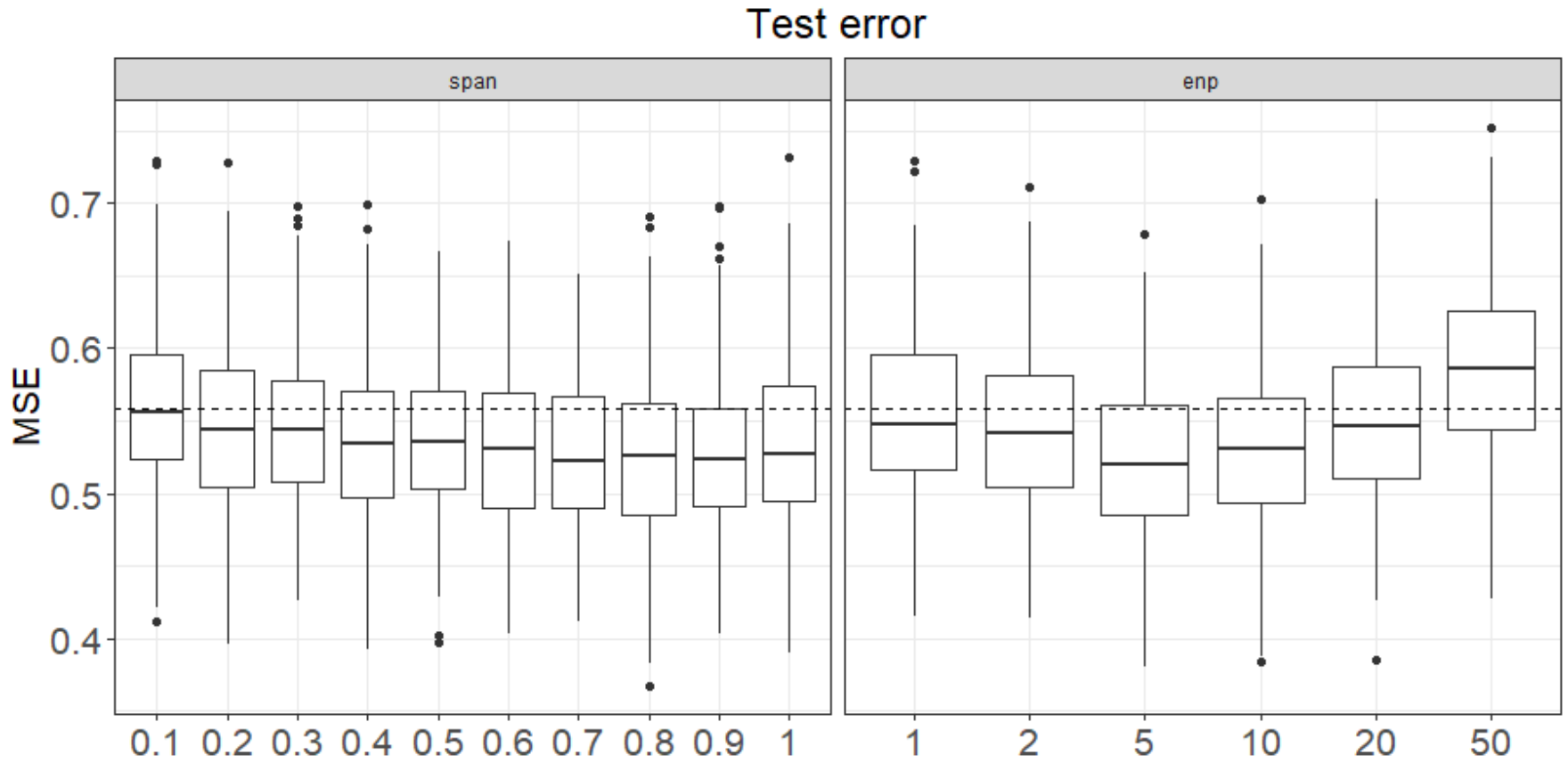


- Can be parameterized by the fraction of the range of the predictor values (span) or by the "equivalent number of parameters" (enp~1/span)

CODE FOR THE PREVIOUS PLOTS

```
tmpDat <- College[College[,"Accept"]!=College[,"Apps"]&rownames(College)!="Cazenovia College",]
tmpDat[,"Yres"] <- resid(lm(Y~Top25perc+F.Undergrad+Room.Board,tmpDat))
tmpDat <- tmpDat[order(tmpDat[,"Outstate"]),]
plot(tmpDat[,"Outstate"],tmpDat[,"Yres"],xlab="Outstate",ylab="Yres",main="Local regression")
abline(h=0,lty=2)
clrsTmp <- c("1"="black","0.5"="red","0.2"="blue","0.1"="magenta")
for ( sTmp in c(0.1,0.2,0.5,1.0) ) {
  loessTmp <- loess(Yres~Outstate,tmpDat,span=sTmp)
  lines(tmpDat[,"Outstate"],predict(loessTmp),col=clrsTmp[as.character(sTmp)],lwd=2)
}
legend("topright",paste0("span=",names(clrsTmp)),lwd=2,col=clrsTmp,text.col=clrsTmp,lty=1)
plot(tmpDat[,"Outstate"],tmpDat[,"Yres"],xlab="Outstate",ylab="Yres",main="Local regression")
abline(h=0,lty=2)
clrsTmp <- c("2"="black","5"="red","10"="blue","20"="magenta")
for ( eTmp in c(2,5,10,20) ) {
  loessTmp <- loess(Yres~Outstate,tmpDat,enp.target=eTmp)
  lines(tmpDat[,"Outstate"],predict(loessTmp),col=clrsTmp[as.character(eTmp)],lwd=2)
}
legend("topright",paste0("enp=",names(clrsTmp)),lwd=2,col=clrsTmp,text.col=clrsTmp,lty=1)
```

LOCAL REGRESSION TEST ERROR FOR COLLEGE DATASET



- By default, does not extrapolate; can be enabled (see examples in ?loess), but likely results in highly(!) variable predictions; here test observations outside of training span were omitted

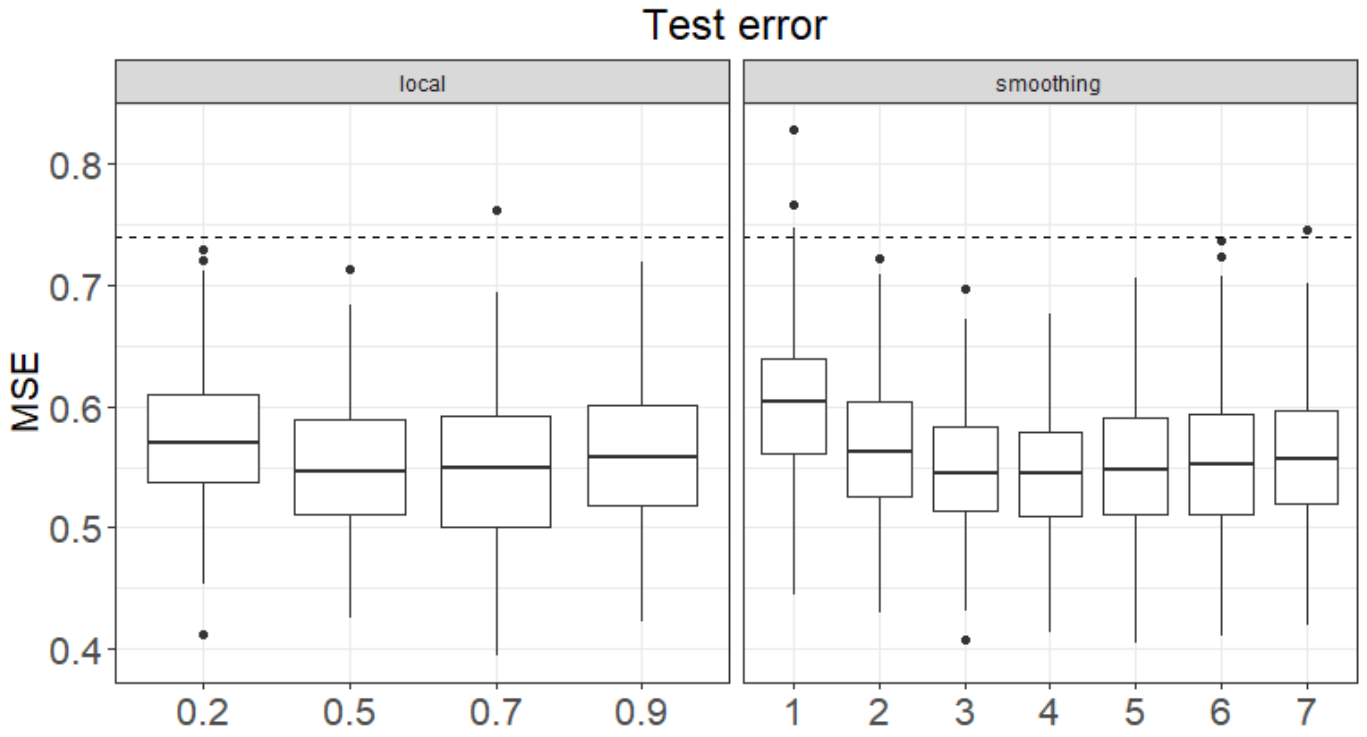
CODE FOR THE PREVIOUS PLOTS

```
nTries <- 200; dfTmp <- NULL; lTmp <- list(span=c(1:10)/10,enp=c(1,2,5,10,20,50))
for ( pTmp in c("span","enp") ) {
  for ( sTmp in lTmp[[pTmp]] ) {
    for ( iTry in 1:nTries ) {
      trainIdx <- sample(nrow(tmpDat),nrow(tmpDat),replace = TRUE)
      trainDat <- tmpDat[trainIdx,]; testDat <- tmpDat[-trainIdx,]
      if ( pTmp == "span" ) {
        loessTmp <- loess(Yres~Outstate,trainDat,span=sTmp)
      } else {
        loessTmp <- loess(Yres~Outstate,trainDat,enp.target=sTmp)
      }
      loessTestPred <- predict(loessTmp,testDat)
      testMSE <- mean((testDat[,"Yres"]-loessTestPred)^2,na.rm=TRUE)
      dfTmp <- rbind(dfTmp,data.frame(p=pTmp,val=sTmp,mse=testMSE))
    }
  }
}
ggplot(dfTmp,aes(x=factor(val),y=mse))+geom_boxplot()+theme_bw()+xlab("Span")+ylab("MSE")+
       ggtitle("Test error")+theme(plot.title = element_text(hjust=0.5,size=18),axis.text =
element_text(size=16),axis.title = element_text(size=16))+
       geom_hline(yintercept = var(tmpDat[,"Yres"]),lty=2)+facet_wrap(~p,scales="free_x")+xlab("")
```

EXAMPLE OF GAM FOR COLLEGE DATA

```
> tmpDat <- College[College[,"Accept"]!=College[,"Apps"]&rownames(College)!="Cazenovia College",]
> gam1s <- gam(Y~s(Top25perc,4),data=tmpDat)
> gam2s <- gam(Y~s(Top25perc,4)+s(Outstate,4),data=tmpDat)
> gam3s <- gam(Y~s(Top25perc,4)+s(Outstate,4)+s(Top10perc,4),data=tmpDat)
> anova(gam1s,gam2s,gam3s)
Model 1: Y ~ s(Top25perc, 4)
Model 2: Y ~ s(Top25perc, 4) + s(Outstate, 4)
Model 3: Y ~ s(Top25perc, 4) + s(Outstate, 4) + s(Top10perc, 4)
  Resid. Df Resid. Dev     Df Deviance  Pr(>Chi)
1       765     440.34
2       761     413.93 4.0000   26.405 3.732e-10 ***
3       757     400.32 3.9997   13.616 3.555e-05 ***
> gam1lo <- gam(Y~lo(Top25perc,span=0.7),data=tmpDat)
> gam2lo <- gam(Y~lo(Top25perc,span=0.7)+lo(Outstate,span=0.7),data=tmpDat)
> gam3lo <- gam(Y~lo(Top25perc,span=0.7)+lo(Outstate,span=0.7)+lo(Top10perc,span=0.7),data=tmpDat)
> anova(gam1lo,gam2lo,gam3lo)
Model 1: Y ~ lo(Top25perc, span = 0.7)
Model 2: Y ~ lo(Top25perc, span = 0.7) + lo(Outstate, span = 0.7)
Model 3: Y ~ lo(Top25perc, span = 0.7) + lo(Outstate, span = 0.7) + lo(Top10perc,
    span = 0.7)
  Resid. Df Resid. Dev     Df Deviance  Pr(>Chi)
1    766.07     443.60
2    763.29     420.62 2.7799   22.977 1.767e-09 ***
3    760.12     406.42 3.1646   14.205 8.973e-06 ***
>
```

GAM TEST ERROR FOR COLLEGE DATA



- Span of 0.5-0.7 for local regression (on Top25perc, Outstate and Top10perc) or degree of 3-5 result in lower test error
- Subject to choosing those variables on the entire dataset, and using the same parameter (span or degree) for all

CODE FOR THE ABOVE PLOTS

```
nTries <- 200; dfTmp <- NULL
lTmp <- list(local=c(0.2,0.5,0.7,0.9),smoothing=1:7)
for ( pTmp in names(lTmp) ) {
  for ( sTmp in lTmp[[pTmp]] ) {
    for ( iTry in 1:nTries ) {
      trainIdx <- sample(nrow(tmpDat),nrow(tmpDat),replace = TRUE)
      trainDat <- tmpDat[trainIdx,]; testDat <- tmpDat[-trainIdx,]
      if ( pTmp == "local" ) {
        gamTmp <-
gam(Y~lo(Top25perc,span=sTmp)+lo(Outstate,span=sTmp)+lo(Top10perc,span=sTmp),data=trainDat)
      }
      if ( pTmp == "smoothing" ) {
        gamTmp <- gam(Y~s(Top25perc,sTmp)+s(Outstate,sTmp)+s(Top10perc,sTmp),data=trainDat)
      }
      gamTestPred <- predict(gamTmp,testDat)
      testMSE <- mean((testDat[,"Y"]-gamTestPred)^2,na.rm=TRUE)
      dfTmp <- rbind(dfTmp,data.frame(p=pTmp,val=sTmp,mse=testMSE))
    }
  }
}
ggplot(dfTmp,aes(x=factor(val),y=mse))+geom_boxplot()+theme_bw()+xlab("")+ylab("MSE")+ggtitle("Test
error")+theme(plot.title = element_text(hjust=0.5,size=18),axis.text =
element_text(size=16),axis.title = element_text(size=16))+geom_hline(yintercept =
var(tmpDat[,"Y"]),lty=2)+facet_wrap(~p,scales="free_x")+xlab("")
```