36-463/663: Multilevel & Hierarchical Models

Regression – Fit, Uncertainty & Improvement

Brian Junker
132E Baker Hall
brian@stat.cmu.edu

Reading and HW

- Reading:
  - G&H Ch's 3-4 this week
  - G&H Ch's 5-6 next week (prolly starting this Thurs)
    - I will not cover everything in the chapters
    - You will need to read & try some things on your own!
- HW03 posted
  - HW02 due today
  - HW01 returned today!
- Many more examples in R are online also!

Outline

- Writing the Linear Model – the Lazy, Long, and Matrix Ways
  - Aside: The Normal Distribution
- Fitting the Model
  - Things we can Estimate
  - Displaying Uncertainty
  - New data: Estimating the mean vs prediction
- Transformations
  - Extended Example with Logarithms
  - Some Do's and Don'ts for Discrete Variables
- G&H Advice on Model Building

Writing the Model – The Lazy Way

- Lazy way

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$$

- Quick to write, good for waving hands at
- Easy to replace abstract letters like β and X with possibly meaningful words like (coef1) and (mom.iq), e.g.:

```
(kid.score) = (intcpt) + (coef1)(mom.hs) +
              (coef2)(mom.iq) + (error)
```

- Emphasizes the functional relationship

Writing the model – The Long Way

$$\begin{aligned} y_1 &= \beta_1 X_{11} + \beta_2 X_{12} + \cdots + \beta_k X_{1k} + \epsilon_1 \\ y_2 &= \beta_1 X_{21} + \beta_2 X_{22} + \cdots + \beta_k X_{2k} + \epsilon_2 \\ &\vdots \\ y_n &= \beta_1 X_{n1} + \beta_2 X_{n2} + \cdots + \beta_k X_{nk} + \epsilon_n \end{aligned}$$

- n = number of "units" or "cases".
- k = number of "predictors" or "covariates".
- Note re-indexing of beta's (start at 1, not 0) and the "intercept" seems to be gone (it's really in X_{i1}).
- Can write more compactly as

$$y_i = \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_k X_{ik} + \epsilon_i, \; i = 1, \ldots n$$

Writing the Model – The Matrix Way

- Let $X_i = (X_{i1}, \ldots, X_{ik})$ and $\beta = (\beta_1, \ldots, \beta_k)^T$; then

$$\begin{aligned} y_i &= \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_k X_{ik} + \epsilon_i \\ &= X_i \beta + \epsilon_i \end{aligned}$$

- If we also stack $Y = (Y_1, \ldots, Y_n)^T$, $X = (X_1^T, \ldots, X_n^T)^T$, and $\epsilon = (\epsilon_1, \ldots \epsilon_n)^T$, we can write

$$Y = X\beta + \epsilon$$

- Has all the information of the Long Way
- Lends itself to compact formulas & computation

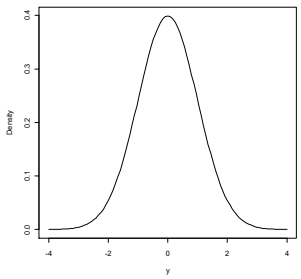
Writing the Model: Distributions

Three equivalent expressions:

- $y_i = X_i\beta + \epsilon_i,\ \epsilon_i \overset{iid}{\sim} N(0, \sigma^2),\ i = 1, \ldots, n$
- $y_i \sim N(X_i\beta, \sigma^2),\ i = 1, \ldots, n$
- $Y \sim N(X\beta, \sigma^2 I)$

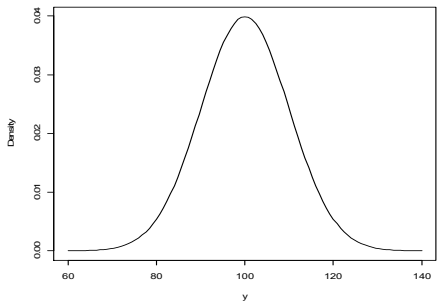
Aside: The Normal Distribution

- Y ~ N(0,1) iff $f(y) = \frac{1}{\sqrt{2\pi}} e^{-y^2/2}$



- Y ~ N(μ,σ^2) iff $\frac{y-\mu}{\sigma} \sim N(0,1)$

$$f(y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-\mu)^2}{2\sigma^2}}$$



Aside: The Normal Distribution

- $Y = (Y_1, \ldots, Y_n)^T \sim N(0, I) = N\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix}\right)$

$$f(y_1, \ldots, y_n) = \prod_{i=1}^{n} f(y_i) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} e^{-y_i^2/2}$$

- $Y \sim N(\mu, \Sigma)$ iff

$$\Sigma^{-1/2}(Y - \mu) \sim N(0, I)$$

…and some ugly formula for $f(y_1, \ldots, y_n)$…

Aside: The Normal Distribution

- When $Y \sim N(\mu, \Sigma)$, then

$$\mu = (\mu_1, \mu_2, \ldots, \mu_n)^T = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_n \end{bmatrix}$$

is the *mean vector*:

$$E[y_i] = \mu_i, \; i = 1, \ldots, n$$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_n^2 \end{bmatrix}$$

is the *variance-covariance matrix*:

$$\begin{aligned} \text{Var}(y_i) &= \sigma_i^2, \; i = 1, \ldots, n \\ \text{Cov}(y_i, y_j) &= \sigma_{ij}, \; i, j = 1, \ldots, n \end{aligned}$$

Writing the Model - Summary

- $Y \sim N(X\beta, \sigma^2 I)$ means:
  - $E[y_i] = X_i\beta$, i = 1, …,n
  - $Var(y_i) = \sigma^2$, i=1, …, n
  - $Cov(y_i,y_j) = 0, \forall\ i \neq j$
- We could also write
  - $Y = X\beta + \epsilon,\ \epsilon \sim N(0,\sigma^2 I)$
  - $y_i = X_i\beta + \epsilon_i,\ \epsilon_i \sim N(0,\sigma^2)$, iid
  - $y_i = \beta_1 X_{i1} + \ldots + \beta_k X_{iK} + \epsilon,\ \epsilon_i \sim N(0,\sigma^2)$, iid

Fitting the Model – Residual SD and Least Squares

- $Y_i = X_i\beta + \epsilon_i,\ \epsilon_i \sim N(0,\sigma^2)$
- So $Var(Y_i) = E[(Y_i - X_i\beta)^2] = E[\epsilon_i^2] = Var(\epsilon_i) = \sigma^2$
- Then we can estimate (MoM!):

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - X_i\beta)^2$$

- Fitting the model is basically just finding values β to minimize

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - X_i\beta)^2 = \frac{1}{n}(Y - X\beta)^T(Y - X\beta)$$

- It turns out that $\hat{\beta} = (X^T X)^{-1} X^T y$

Fitting the Model – Summary of Things we can Estimate…

$$y = X\beta + \epsilon,\ \epsilon \sim N(0, \sigma^2 I)$$

- $\hat{\beta} = (X^T X)^{-1} X^T y$
- $\hat{y} = X\hat{\beta} = X(X^T X)^{-1} X^T y = Hy$
- The "residual SD" is the square root of $\hat{\sigma}^2 = \frac{1}{n-k}\sum_{i=1}^{n}(y_i - X_i\hat{\beta})^2 = \frac{1}{n-k}(y - X\hat{\beta})^T(y - X\hat{\beta})$
- We can also calculate that

$$\begin{aligned} \mathrm{Var}(\hat{\beta}) &= (X^T X)^{-1}\sigma^2 \\ \mathrm{Var}(\hat{y}) &= X(X^T X)^{-1} X^T \sigma^2 \end{aligned}$$

Fitting the Model - Example

```
> library(arm)
> str(kidiq)
> y <- cbind(kidiq$kid.score)
> X <- cbind(1, with(kidiq,
   cbind(mom.hs,mom.iq)))
> dim(X)
[1] 434   3
> n <- dim(X)[1]
> k <- dim(X)[2]
> V <- solve(t(X) %*% X)
> beta.hat <- V %*% t(X) %*% y
> res.var <- t(y - X%*%beta.hat)
   %*% (y - X%*%beta.hat) / (n-k)
> res.sd <- sqrt(res.var)
>
> var.beta <- V * c(res.var)
> beta0.sd <- sqrt(var.beta[1,1])
> beta1.sd <- sqrt(var.beta[2,2])
> beta2.sd <- sqrt(var.beta[3,3])
```

```
> round(cbind(beta.hat,
   c(beta0.sd, beta1.sd,
   beta2.sd)), 2)
        [,1] [,2]
      25.73 5.88
mom.hs  5.95 2.21
mom.iq  0.56 0.06
> round(res.sd,2)
      [,1]
[1,] 18.14
> display(lm(kid.score ~ mom.hs
   + mom.iq, data=kidiq))

            coef.est  coef.se
(Intercept) 25.73     5.88
mom.hs       5.95     2.21
mom.iq       0.56     0.06
---
n = 434, k = 3
res sd = 18.14, R-Squared = 0.21
```

Fitting the Model – Example Cont'd

- The square roots of the diagonal elements of

```
> var.beta
                            mom.hs       mom.iq
       34.51806922 -0.05610582 -0.337161456
mom.hs -0.05610582  4.89211314 -0.037876974
mom.iq -0.33716146 -0.03787697  0.003669219
```

  give us the SE's of the $\hat{\beta}$'s

- The off-diagonal elements give us

$$\mathrm{Cov}(\hat{\beta}_{int}, \hat{\beta}_{hs}) = -0.06, \ \mathrm{Cov}(\hat{\beta}_{int}, \hat{\beta}_{iq}) = -0.34, \ \text{etc.}$$

- The sim() function from library(arm) uses formulas like these to simulate new data (and regression estimates) from a model

Fitting the Model – Example Cont'd

- What about $R^2 = 0.21$ at the end of the display() output?
  - Raw variance of Y = $\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{y})^2 = 416.6$
  - Modeled variance = $\frac{1}{n-1}\sum_{i=1}^{n}(\hat{y}_i - \overline{y})^2 = 89.2$
  - $R^2 = \frac{\text{modeled variance}}{\text{raw variance}} = \frac{89.2}{416.6} = 0.21$
  - It can be shown that (modeled variance) + (residual variance) = (raw variance), so you also sometimes see

$$R^2 = \frac{\text{(raw variance)} - \text{(residual variance)}}{\text{(raw variance)}} = 1 - \frac{\hat{\sigma}^2}{\mathrm{Var}(Y)} = 1 - \frac{328.9}{416.6} = 0.21$$

  - The name R^2 comes from

$$r = \mathrm{Cor}(y, \hat{y}) = 0.46; \ r^2 = (0.46)^2 = 0.21$$

Fitting the Model – Displaying the Uncertainty

- Statistical estimates are influenced by
  - The underlying relationship we are hoping to find
  - Random variation or 'noise' in the data
- How much does the 'noise' affect our uncertainty about the underlying relationships?
- Can use sim() function in library(arm) to display the uncertainty

Fitting the Model – Displaying the Uncertainty

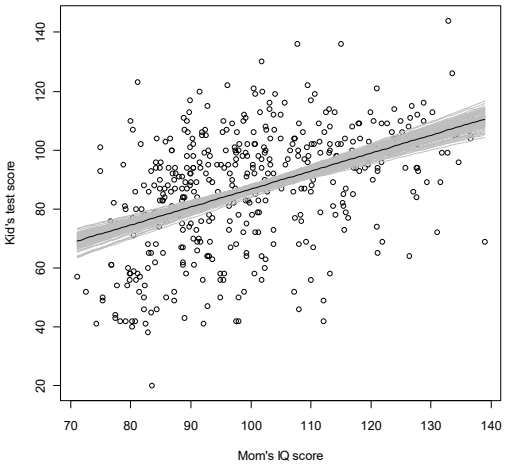
```
attach(kidiq)

fit.lm.2 <- lm(kid.score ~ mom.iq)
fit.sim.2 <- sim(fit.lm.2,n.sims=100)
plot(kid.score ~ mom.iq, xlab="Mom's
    IQ score", ylab="Kid's test score")

plotit <- function(beta)
    {curve(cbind(1,x)%*%beta, add=T,
    col="Grey")}
apply(fit.sim.2@coef,1,plotit)

curve(cbind(1,x)%*%coef(fit.lm.2),
    add=T,col="Black")

detach()
```



Fitting the Model – Predicting New y's

- If we know $\hat{\beta}$ and we have a new set of X values, X_{new}, then it is easy to get $y_{new} = X_{new}\hat{\beta}$

```
> x.new <- data.frame(mom.hs = 1, mom.iq=100)
> predict(fit.lm.5, x.new, interval="confidence", level=0.95)
       fit      lwr      upr
1 88.07226 86.12407 90.02045
> predict(fit.lm.5, x.new, interval="prediction", level=0.95)
       fit      lwr      upr
1 88.07226 52.37369 123.7708
```

- The ***confidence interval*** represents uncertainty locating the regression line in the grey band
- The ***prediction interval*** represents the additional uncertainty locating the data point in the data cloud

Transforming Variables

- We transform variables for many reasons
  - To make the data more interpretable

$$x - mean(x);\ \frac{x - mean(x)}{SD(x)};\ a + bx;\ \log(x)$$

  - To symmetrize variables and to pull in outliers that might otherwise be too influential on the fit

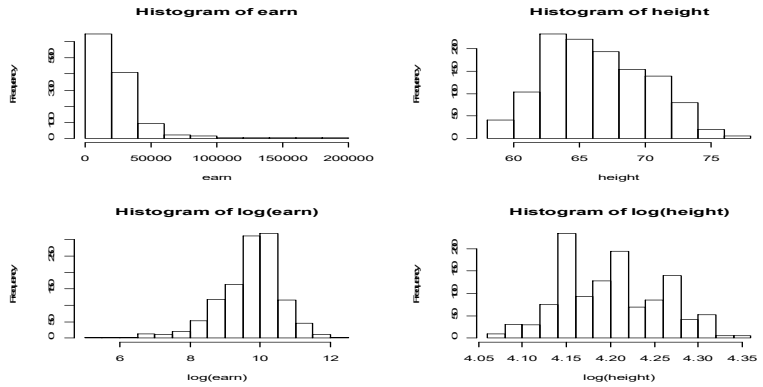
$$log(x);\ \sqrt{x};\ x^p$$

  - To explore for nonlinear relationships

    (many functions, but also converting a variable into a 'stairstep function' etc.)
  - To convert index and factor variables into indicators

Transforming Variables – log(y), log(x)

- Positive counts or measures (heights, incomes, etc.) tend to be positively skewed.
- The outliers tend to be influential, and the skewing shows up in residual plots, in regression.
- Logarithms often symmetrize the data and pull in outliers



Transformations - logarithms

- We tend to prefer natural log ($\log_e$, ln), rather than $\log_{10}$ or some other logarithm. *We always write just* log(x) *or* log(y). *Never* ln, $\log_e$…
- Often:
  - log(y) will help with skewing and other problems in the residuals
  - log(x) will help with overly influential outliers
- Since log(1+u) ≈ u for small u, a change in log(y) is like a percent change in y. Thus, in

$$\log(y) = \beta_0 + \beta_1 x + \epsilon$$

$\beta_1 \approx$ percent change in y for one unit change in x.

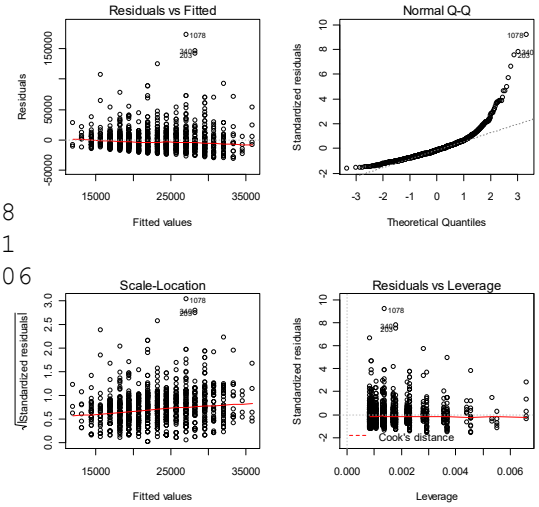
Transformations – Earnings Example

- We want (!?) to predict earnings from height and gender.

```
model.0 <- lm(earn ~ height)
display(model.0)

             coef.est  coef.se
(Intercept) -61316.28   9525.18
height       1262.33    142.11
resid sd = 18865.08, R-Sq = 0.06

par(mfrow=c(2,2))
plot(model.0)
par(mfrow=c(1,1))
```

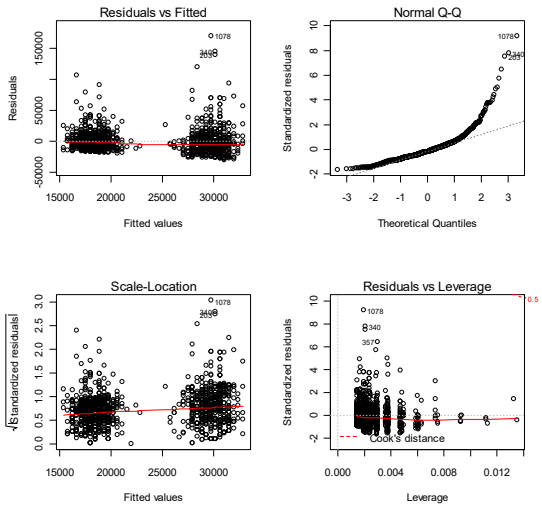


Transformations – Earnings Cont'd

```
model.1 <- lm(earn ~ height + male)
display(model.1)

             coef.est  coef.se
(Intercept) -10334.71  12722.07
height         442.93    196.62
male          9087.87   1529.93
resid sd = 18599.06, R-Sq = 0.09

par(mfrow=c(2,2))
plot(model.1)
par(mfrow=c(1,1))
```

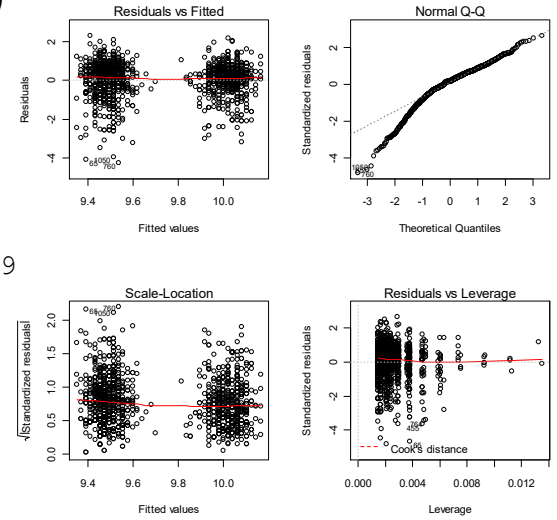


Transformations – Earnings Cont'd

```
model.2 <- lm(log(earn) ~ height + male)
display(model.2)
```

```
            coef.est coef.se
(Intercept) 8.15     0.60
height      0.02     0.01
male        0.42     0.07
resid sd = 0.88, R-Sq = 0.09
```

```
par(mfrow=c(2,2))
plot(model.2)
par(mfrow=c(1,1))
```

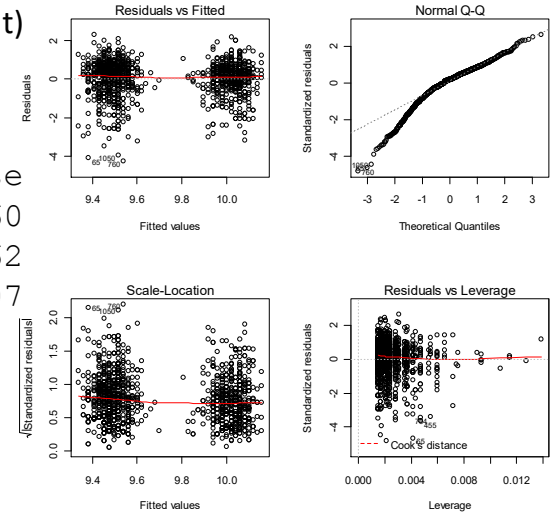


Transformations – Earnings Cont'd

```
model.3 <- lm(log(earn) ~ log(height)
    + male)
display(model.3)
```

```
              coef.est coef.se
(Intercept)   3.62     2.60
log(height)   1.41     0.62
male          0.42     0.07
resid sd = 0.88, R-Sq =
   0.09
```

```
par(mfrow=c(2,2))
plot(model.3)
par(mfrow=c(1,1))
```



The analysis isn't done yet, but you can see some of the effects of logs

Discrete Predictors: Indexes and Indicators

- Indicator variables are 0/1 variables that indicate the presence or absence of some factor or feature (male/female, hs/no-hs, etc.).
- Index variables are basically the same as "factors" in R. They just indicate values of other variables, often without any order or numerical value.

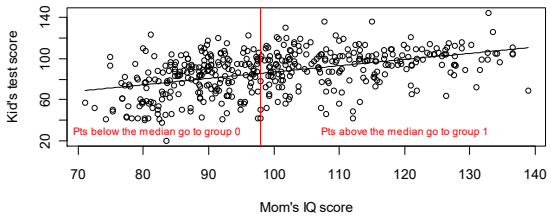
Discrete Predictors: Example 1

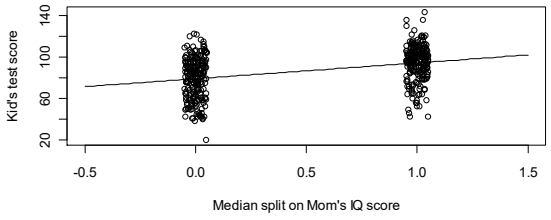
- Can lose predictive power and interpretation when we convert a continuous variable to an indicator or index "to simplify things".

```
attach(kidiq)
str(kidiq)

iq.fit.1 <- lm(kid.score ~ mom.iq)
display(iq.fit.1)
            coef.est coef.se
(Intercept) 25.80     5.92
mom.iq       0.61     0.06
resid sd = 18.27, R-Sq = 0.20

lohi.iq <- ifelse(mom.iq<median(mom.iq),0,1)
iq.fit.2 <- lm(kid.score ~ lohi.iq)
display(iq.fit.2)
            coef.est coef.se
(Intercept) 79.09     1.29
lohi.iq     15.34     1.82
resid sd = 18.93, R-Sq = 0.14
```





Discrete Predictors: Example 2

- Indexes should not be treated as continuous. R can help, with “as.factor()”.

```
# mom.work takes the values
#
# 1 = did not work first 3 years of child's life
# 2 = worked in 2nd or 3rd year of child's life
# 3 = worked part time in first year of child's life
# 4 = worked full time in first year of child's life

iq.fit.3 <- lm(kid.score ~ mom.iq + mom.hs + mom.work)
display(iq.fit.3)
            coef.est coef.se
(Intercept) 25.38     6.07
mom.iq       0.56     0.06
mom.hs       5.83     2.28
mom.work     0.18     0.76
res sd = 18.16, R-Sq = 0.21
```

```
# the following analysis makes more sense…

iq.fit.4 <- lm(kid.score ~ mom.iq + mom.hs +
as.factor(mom.work))
display(iq.fit.4)

                        coef.est coef.se
(Intercept)               24.90     6.11
mom.iq                     0.55     0.06
mom.hs                     5.71     2.28
as.factor(mom.work)2       2.88     2.81
as.factor(mom.work)3       5.62     3.24
as.factor(mom.work)4       1.49     2.51
res sd = 18.13, R-Sq = 0.22
```

Note that one level of the mom.work factor is gone – where did it go?

Discrete Predictors, Ex. 2 Cont’d

- We can build our own indicators for levels of mom.work as well.

```
work.none <- ifelse(mom.work==1,1,0)
work.23  <- ifelse(mom.work==2,1,0)
work.1p  <- ifelse(mom.work==3,1,0)
work.1f  <- ifelse(mom.work==4,1,0)

iq.fit.5 <- lm(kid.score ~ mom.iq + mom.hs + work.none +
        work.23 + work.1p + work.1f)
display(iq.fit.5)
            coef.est coef.se
(Intercept) 26.39     6.10
mom.iq       0.55     0.06
mom.hs       5.71     2.28
work.none   -1.49     2.51
work.23      1.39     2.26
work.1p      4.13     2.76

res sd = 18.13, R-Sq d = 0.22
```

```
iq.fit.6 <- lm(kid.score ~ mom.iq + mom.hs +
        work.23 + work.1p + work.1f)
display(iq.fit.6)
 coef.est coef.se
(Intercept) 24.90     6.11
mom.iq       0.55     0.06
mom.hs       5.71     2.28
work.23      2.88     2.81
work.1p      5.62     3.24
work.1f      1.49     2.51
res sd = 18.13, R-Sq = 0.22
```

Discrete Predictors, Example 3

- To explore non-linear relationships, it can make sense to break a continuous variable up into many discrete parts.

```
# range(mom.iq)

bin <- NULL
for (i in (7:13)*10) {
  bin <- cbind(bin,ifelse(
      (mom.iq>i)&(mom.iq<(i+10)),1,0))
}

iq.fit.10 <- lm(kid.score ~ bin)
display(iq.fit.10)
            coef.est coef.se
(Intercept) 102.87     4.73
bin1        -37.80     5.86
bin2        -24.30     5.06
bin3        -18.23     5.05
bin4         -9.32     5.14
bin5         -8.95     5.30
bin6         -3.89     5.57
res sd = 18.32, R-Sq = 0.21
```

```
# remove the intercept to get
# estimates in all seven bins.

iq.fit.11 <- lm(kid.score ~ bin - 1)
display(iq.fit.11)

     coef.est coef.se
bin1   65.07     3.46
bin2   78.56     1.81
bin3   84.64     1.76
bin4   93.55     2.02
bin5   93.92     2.38
bin6   98.97     2.93
bin7  102.87     4.73
res sd = 18.32, R-Sq = 0.96
```

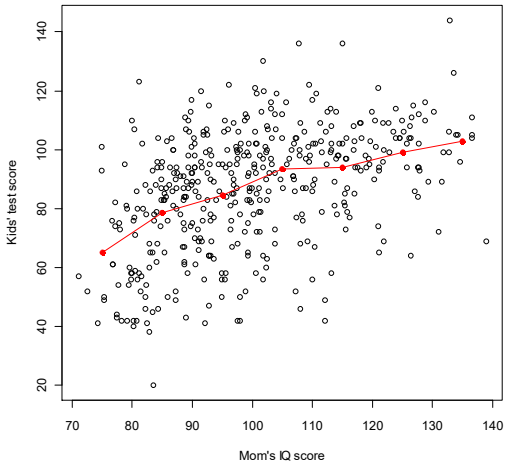
R^2 not meaning-ful when intercept missing!

Discrete Predictors, Ex 3, Continued

```
plot(kid.score ~ mom.iq,xlab="Mom's IQ
     score", ylab="Kids' test score")

points((7:13)*10 + 5,
     coef(iq.fit.11),pch=19,col="Red")
lines((7:13)*10 + 5,
     coef(iq.fit.11),col="Red")
```

- To see the nonlinear trend, connect the dots…
- Each dot is really just a mean of the y-values in that bin…



Discrete Predictors, Ex 3, Continued

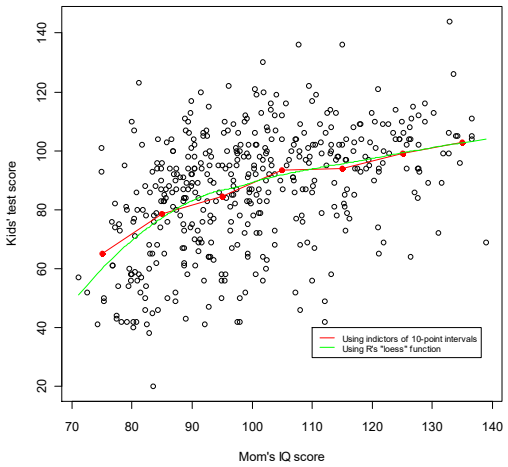
```
plot(kid.score ~ mom.iq,xlab="Mom's IQ score",
     ylab="Kids' test score")

points((7:13)*10 + 5,
     coef(iq.fit.11),pch=19,col="Red")
lines((7:13)*10 + 5, coef(iq.fit.11),col="Red")

# there are other, more sophisticated ways to do
     this

ord <- order(mom.iq)
x <- mom.iq[ord]
y <- loess(kid.score~mom.iq)$fitted[ord]
lines(x,y, col="Green")

legend(110,40,lty=1,col=c("Red","Green"),
     legend=
      c("Using indictors of 10-point intervals",
       "Using R's \"loess\" function"),cex=0.65)
```



G&H Advice on Model-Building

- Include all input variables (X) that "substantive theory" says might predict the outcome (Y).
- Sometimes predictors can be combined (summed, averaged, differenced, etc.) before including them in the model.
- What to do about "significant" and "nonsignificant" coefficients (e.g. at 0.05 level):
  - If a predictor is ***significant*** and has the ***right sign***, keep it in the model!
  - If a predictor is ***nonsignificant*** but has the ***right sign***, it generally doesn't help or hurt much to leave it in the model
  - If a predictor is ***nonsignificant*** and has the ***wrong sign***, consider removing it from the model
  - If a predictor is ***significant*** and has the ***wrong sign***, think hard:
    - Does it make sense after all?
    - Could it could be caused by "lurking variables" that you can measure and include in the model?

Summary

- Writing the Linear Model – the Lazy, Long, and Matrix Ways
- Fitting the Model
- Transformations
- G&H Advice on Model Building
- Reading:
  - G&H CH's 3-4 this week
  - G&H Ch's 5-6 starting Thursday
  - Many more examples at the class website – try them!
- Hw03 due next Tuesday.