Homework 06 Solutions

1 Problem 1

```
library(tidyverse)
library(kableExtra)
library(GGally)
library(grid)
library(gridExtra)
library(ggplotify)
library(reshape2)

cdidata <- read.table("cdi.dat",header=T)
```

Note: There is more detail in the solutions for problem #1 than I would expect to see in a technical appendix. I have included extra detail so that you can see some of my thinking, and some of my approaches to problem-solving in applied regression. When you begin to convert your revised solutions to problem #1 into a technical appendix for the project 1 IDMRAD paper, you should include technical material (and explanations) that help support the claims that you will be making in the main paper. You know you have done the technical appendix right, if you can tell the reader after each claim you make, which page(s) of the technical appendix has more evidence, calculations, etc., to back up that claim. If you do not refer to part of your technical appendix in the main body of your paper, you may not need to include that part in your technical appendix at all.

1.1 Part a

1.1.1 Make a table or tables showing appropriate summary statistics for each variable in the data set. Note that summary statistics for continuous variables will be different from the summary statistics for categorical variables.

First I'll just look at the "head" of all the variables (see Tables 1 and 2); this is a bit like looking at `str()`, which is also worth looking at (I split the "head" into two parts, because there are too many variables to fit horizontally on the page).

```
head(cdidata[,1:10]) %>% kbl(booktabs=T,caption=" ") %>% kable_classic()
```

Table 1:

id	county	state	land.area	pop	pop.18_34	pop.65_plus	doctors	hosp.beds	crimes
1	Los_Angeles	CA	4060	8863164	32.1	9.7	23677	27700	688936
2	Cook	IL	946	5105067	29.2	12.4	15153	21550	436936
3	Harris	TX	1729	2818199	31.3	7.1	7553	12449	253526
4	San_Diego	CA	4205	2498016	33.5	10.9	5905	6179	173821
5	Orange	CA	790	2410556	32.6	9.2	6062	6369	144524
6	Kings	NY	71	2300664	28.3	12.4	4861	8942	680966

Table 2:

id	pct.hs.grad	pct.bach.deg	pct.below.pov	pct.unemp	per.cap.income	tot.income	region
1	70.0	22.3	11.6	8.0	20786	184230	W
2	73.4	22.8	11.1	7.2	21729	110928	NC
3	74.9	25.4	12.5	5.7	19517	55003	S
4	81.9	25.3	8.1	6.1	19588	48931	W
5	81.2	27.8	5.2	4.8	24400	58818	W
6	63.7	16.6	19.5	9.5	16803	38658	NE

Table 3:

	unique values
id	440
county	373
state	48
land.area	384
pop	440
pop.18_34	149
pop.65_plus	137
doctors	360
hosp.beds	391
crimes	437
pct.hs.grad	223
pct.bach.deg	220
pct.below.pov	155
pct.unemp	97
per.cap.income	436
tot.income	428
region	4

```
head(cdidata[,c(1,11:17)]) %>% kbl(booktabs=T,caption=" ") %>% kable_classic()
```

We can also check to see how many unique values each variable has (Table 3; this is especially relevant for the categorical variables)

```
apply(cdidata,2,function(x) {length(unique(x))}) %>%
  kbl(booktabs=T,col.names="unique values",caption=" ") %>%
  kable_classic(full_width=F)
```

It looks like `id` is just the same as the row number for each row of the data frame – not useful for data analysis!

`state` has 48 values – that is a lot, and so I may set this variable aside until later in my analyses

`county` is a categorical variable with nearly as many unique values (373) as rows in the cdidata data frame (440). A little more exploration (see R code just below) shows that if I combine `county` with `state`, I get 440 unique values: some counties in different states have the same name. The upshot is that we only have one observation per *unique* county (what we might have thought were multiple observations per county are really single observations from counties with the same name in different states), and so `county` is not a useful

variable to include in models (though it might be useful to identify outliers and other interesting data points in our analyses)

```
county.state <- with(cdidata,paste(county,state))
tmp <- as.data.frame(matrix(sort(county.state),ncol=4))
names(tmp) <- paste("Counties",c("1-110","111-220","221-330","331-440"))
tmp[1:30,] %>% kbl(booktabs=T,longtable=T,caption=" ") %>% kable_classic(full_width=F)
```

Table 4:

Counties 1-110	Counties 111-220	Counties 221-330	Counties 331-440
Ada ID	Ector TX	Lycoming PA	Rockingham NH
Adams CO	El_Dorado CA	Macomb MI	Rockland NY
Aiken SC	El_Paso CO	Macon IL	Rowan NC
Alachua FL	El_Paso TX	Madison AL	Rutherford TN
Alamance NC	Elkhart IN	Madison IL	Sacramento CA
Alameda CA	Erie NY	Madison IN	Saginaw MI
Albany NY	Erie PA	Mahoning OH	Salt_Lake UT
Alexandria_City VA	Escambia FL	Manatee FL	San_Bernardino CA
Allegheny PA	Essex MA	Marathon WI	San_Diego CA
Allen IN	Essex NJ	Maricopa AZ	San_Francisco CA
Allen OH	Fairfax_County VA	Marin CA	San_Joaquin CA
Anderson SC	Fairfield CT	Marion FL	San_Luis_Obispo CA
Androscoggin ME	Fairfield OH	Marion IN	San_Mateo CA
Anne_Arundel MD	Fayette KY	Marion OR	Sangamon IL
Arapahoe CO	Fayette PA	Martin FL	Santa_Barbara CA
Arlington_County VA	Florence SC	Maui HI	Santa_Clara CA
Atlantic NJ	Forsyth NC	McHenry IL	Santa_Cruz CA
Baltimore MD	Fort_Bend TX	McLean IL	Sarasota FL
Baltimore_City MD	Franklin OH	McLennan TX	Saratoga NY
Barnstable MA	Franklin PA	Mecklenburg NC	Sarpy NE
Bay FL	Frederick MD	Medina OH	Schenectady NY
Bay MI	Fresno CA	Merced CA	Schuylkill PA
Beaver PA	Fulton GA	Mercer NJ	Sedgwick KS
Bell TX	Galveston TX	Mercer PA	Seminole FL
Benton WA	Gaston NC	Merrimack NH	Shasta CA
Bergen NJ	Genesee MI	Middlesex CT	Shawnee KS
Berks PA	Gloucester NJ	Middlesex MA	Sheboygan WI
Berkshire MA	Greene MO	Middlesex NJ	Shelby TN
Bernalillo NM	Greene OH	Midland TX	Smith TX
Berrien MI	Greenville SC	Milwaukee WI	Snohomish WA

(I have only listed the first 30 rows, to give a sense of the list of county+state combinations; you can delete the `[1:30,]` in the R code to see the whole table. . .)

Scanning through this table of county & state names, we see that Allen is the first county name that appears in multiple states. And a little further down, both "Baltimore MD" and "Baltimore_City MD" are listed; this makes me wonder whether these two data points are really independent (Baltimore the city is inside Baltimore the county!), but I will ignore possible dependence like this in my analyses below.

Next let's make a table with the usual one-dimensional summary statistics (Table 5)

Table 5:

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	SD
land.area	15.0	451.25	656.50	1041.41	946.75	20062.0	1549.92
pop	100043.0	139027.25	217280.50	393010.92	436064.50	8863164.0	601987.02
pop.18_34	16.4	26.20	28.10	28.57	30.02	49.7	4.19
pop.65_plus	3.0	9.88	11.75	12.17	13.62	33.8	3.99
doctors	39.0	182.75	401.00	988.00	1036.00	23677.0	1789.75
hosp.beds	92.0	390.75	755.00	1458.63	1575.75	27700.0	2289.13
crimes	563.0	6219.50	11820.50	27111.62	26279.50	688936.0	58237.51
pct.hs.grad	46.6	73.88	77.70	77.56	82.40	92.9	7.02
pct.bach.deg	8.1	15.28	19.70	21.08	25.33	52.3	7.65
pct.below.pov	1.4	5.30	7.90	8.72	10.90	36.3	4.66
pct.unemp	2.2	5.10	6.20	6.60	7.50	21.3	2.34
per.cap.income	8899.0	16118.25	17759.00	18561.48	20270.00	37541.0	4059.19
tot.income	1141.0	2311.00	3857.00	7869.27	8654.25	184230.0	12884.32

Table 6:

	NC	NE	S	W
Freq	108	103	152	77

```
cdinumeric <- cdidata[,-c(1,2,3,17)] ## get rid of id, county, state and (for now) region
apply(cdinumeric,2,function(x) c(summary(x),SD=sd(x))) %>% as.data.frame %>% t() %>%
  round(digits=2) %>% kbl(booktabs=T,caption=" ") %>% kable_classic()
```

and a separate table for the one categorical variable that I'll be using a lot (Table 6)

```
tmp <- rbind(with(cdidata,table(region)))
row.names(tmp) <- "Freq"
tmp %>% kbl(booktabs=T,caption=" ") %>% kable_classic(full_width=F)
```

I don't see anything really exciting in the continuous variables (Table 5). There are several variables with Mean substantially larger than Median (`land.area`, `pop`, `doctors`, `hosp.beds`, `crimes`, `per.cap.income`, and `total.income`), indicating possible right-skewing. There are no variables with Mean substantially smaller than Median.

For the `region` variable (Table 6), it might be of some interest that the most counties are in the South (region 'S') and the least are in the west (region 'W'). The low number of counties in the West could be indicative of a lack of sampling in the West, or it could be that counties are just larger (in land area) in the West, so there are fewer counties to sample from. Similarly, the high number of counties in the South could be indicative of over-sampling, or perhap the South simply has a lot of counties that cover only small land areas.

1.1.2 Indicate where (in which variables) there is missing data (NA's), if any, how much there is (in each variable) and why it might be there.

We can check for NA's directly:

```
apply(cdidata,2,function(x) any(is.na(x)) )
```

```
##             id         county          state      land.area            pop
##          FALSE          FALSE          FALSE          FALSE          FALSE
##      pop.18_34    pop.65_plus        doctors      hosp.beds         crimes
##          FALSE          FALSE          FALSE          FALSE          FALSE
##    pct.hs.grad   pct.bach.deg  pct.below.pov      pct.unemp per.cap.income
##          FALSE          FALSE          FALSE          FALSE          FALSE
##     tot.income         region
##          FALSE          FALSE
```

There do not appear to be any missing values in the data! (In general we might also check for old-fashioned missing value codes like "9", "99", "98", etc., but there's no evidence of that in Table 5 (look at the Min and Max values - no "9's", "99's", etc.))

1.1.3 Make some appropriate descriptive EDA plots to illustrate any important features of the variables or possible important relationships among them.

We want to capture (i) univariatoe distributions [primarily historgrams, boxplots, and the like] and (ii) two-variable relationships [primarily scatter plots].

We could try `ggpairs` from `library(GGally)` to capture both (i) and (ii), but there are too many variables to make this a legible plot. Instead, we'll do (i) and (ii) separately.

For much of the rest of this problem, I want `region` to be in the data frame as well as the numerical variables, so I will create that now:

```
cdigood <- data.frame(cdinumeric,region=cdidata$region)
```

First, univariate distributions:

```
## You can get almost the same thing as below with
##
## ggplot(gather(cdinumeric), aes(value)) +
##    geom_histogram(bins=30) +
##    facet_wrap(~key, scales = 'free_x')

hist.builder <- function(df) {  ## creates a list of graphs
    result <- NULL
    for (var in names(df)) {
      d <- data.frame(dd=df[,var])
      if(mode(df[,var])=="numeric") {
        p <- ggplot(d,aes(x=dd)) + geom_histogram() +
              ggtitle(var) + xlab("")
      } else {
          p <- ggplot(d,aes(x=dd)) + geom_bar() +
                ggtitle(var) + xlab("")
      }
    result <- c(result,list(p))
    }
    return(result)
}
```

```
grid.arrange(grobs=hist.builder(cdigood))  ## from library(grid) & library(gridExtra)
```

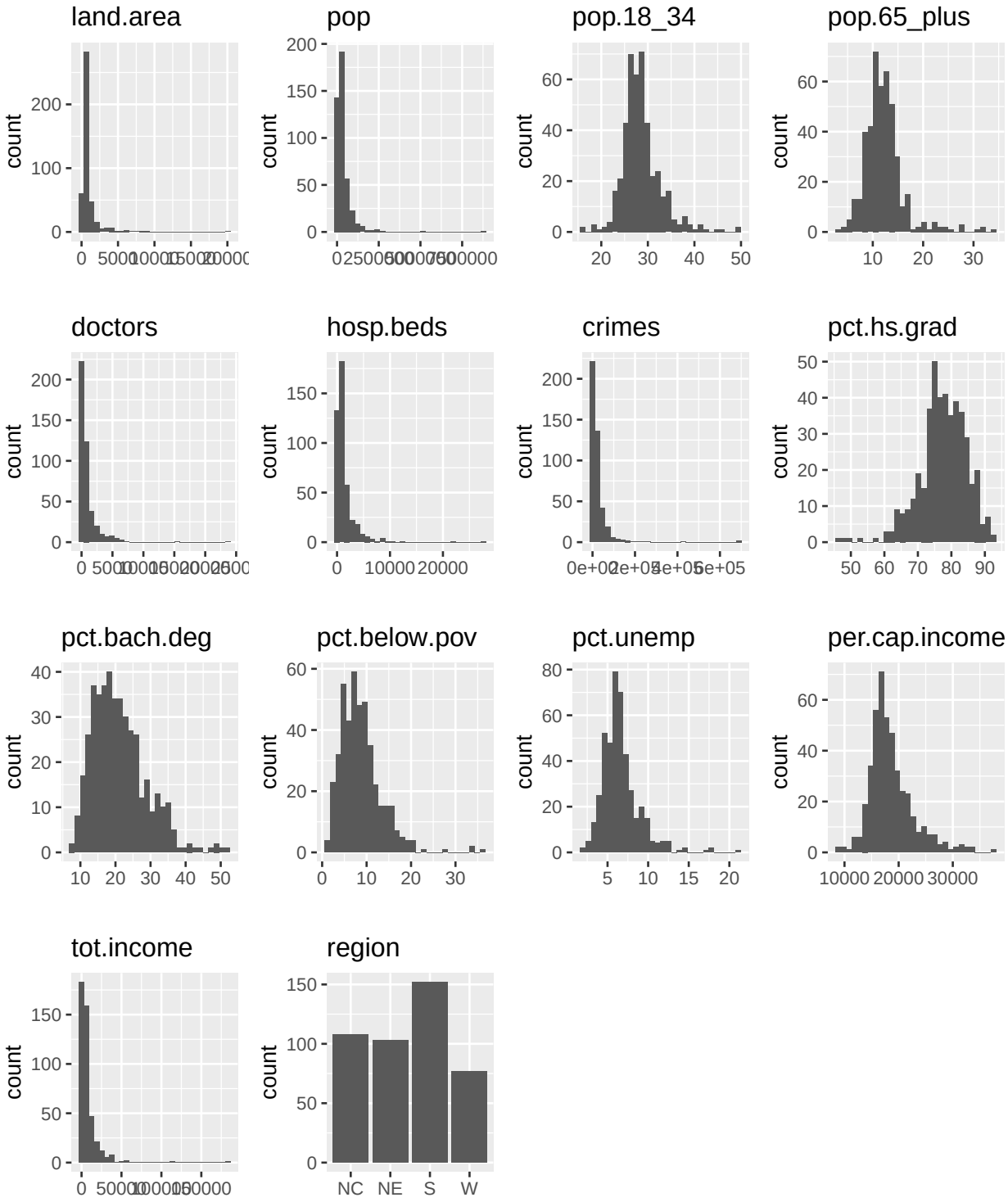


Figure 1: Distributions of Variables

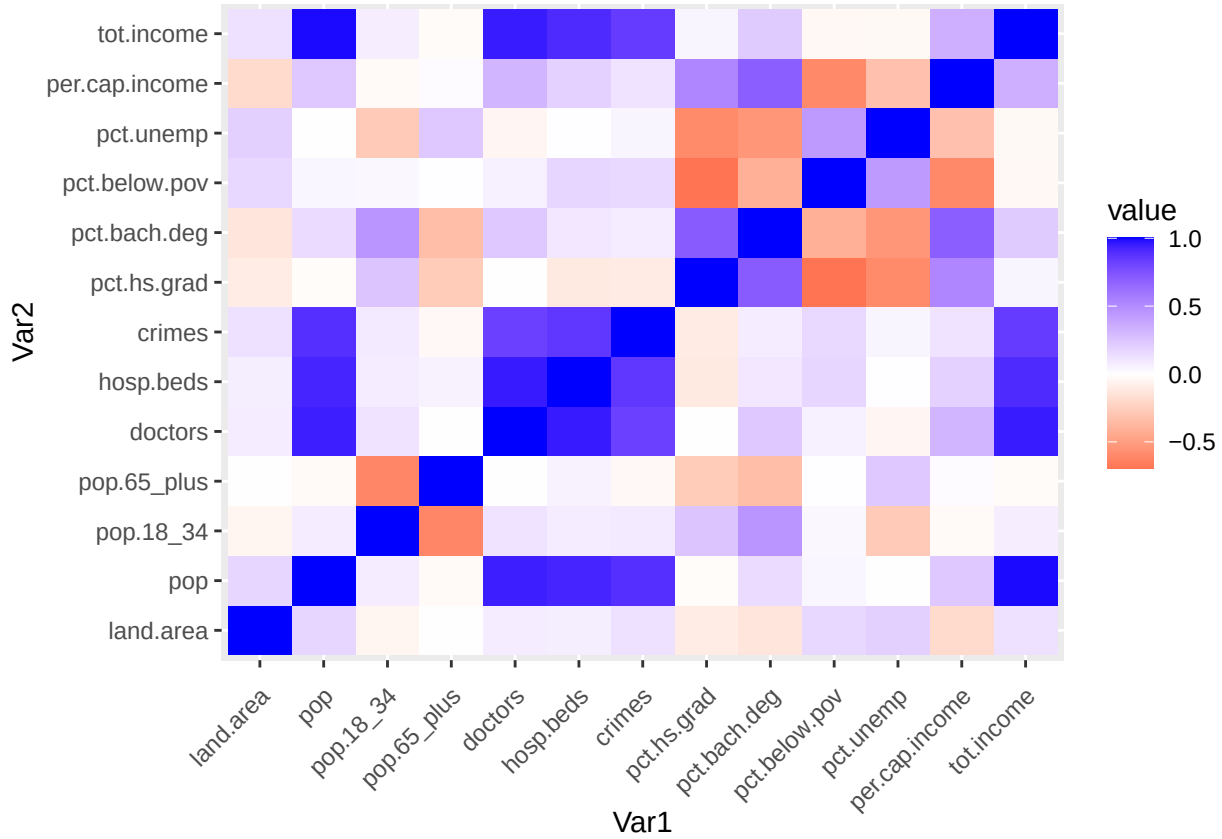
It looks from the histograms like the variables that will really need attention (because they are severely right-skewed) are `land.area`, `pop`, `doctors`, `hosp.beds`, `crimes`, and `tot.income`, and maybe `per.cap.income`. These are the same variables that we identified from Table 5 above.

And now for bivariate relationships... We really only care about two things:

- checking for high correlations among the predictors
- checking to see if the predictors are linearly related to `per.cap.income`

When we have a large number of variables, it can be a more efficient use of the page to make a heatmap of the correlation matrix, rather than looking at the correlation matrix itself:

```
corgraph <- function(df) {
  cormat <- cor(df)
  melted_cormat <- melt(cormat)   ## need library(reshape2) for this...
  ggplot(data = melted_cormat, aes(x=Var1, y=Var2, fill=value)) +
    geom_tile() +
    theme(axis.text.x = element_text(angle = 45,vjust=0.9,hjust=1)) +
    scale_fill_gradient2(low="red",mid="white",high="blue")
}

corgraph(cdinumeric)
```



We can make the following conclusions from the correlation matrix:

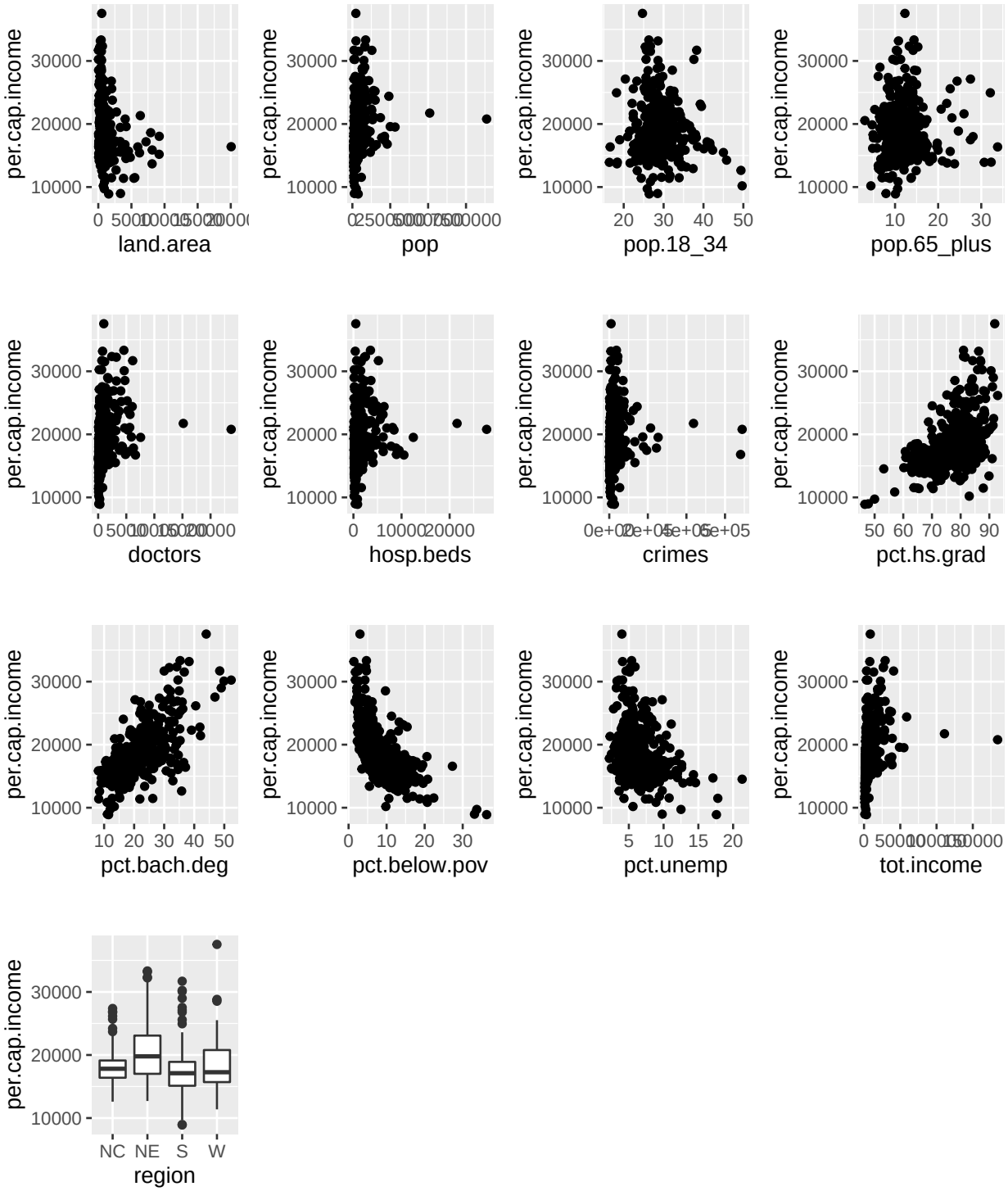
- `tot.income` and `pop` are highly correlated (no surprise there)
- both are reasonably highly correlated with `crimes`, `hosp.beds` and `doctors`
- the three variables `crimes`, `hosp.beds` and `doctors` seem strongly correlated with one another

- `per.cap.income` isn't really highly correlated with anything, but the best possibilities seem to be `pct.hs.grad`, `pct.bach.deg` (postively correlated with `per.cap.income`) and `pct.below.pov`, `pct.unemp` (negatively correlated with `per.cap.income`); all four of these variables are moderately highly correlated with one another

These observations suggest that we may run into multi-collinearity problems when we start fitting models, but there is also some hope that we can make a good model for `per.cap.income`.

Now we turn to scatter plots, but we are just going to concentrate on relationships with `per.cap.income`:

```
## I'm not sure how I'd do this directly with ggplot...

scatter.builder <- function(df,yvar="per.cap.income") {
    result <- NULL
    y.index <- grep(yvar,names(df))
    for (xvar in names(df)[-y.index]) {
      d <- data.frame(xx=df[,xvar],yy=df[,y.index])
      if(mode(df[,xvar])=="numeric") {
        p <- ggplot(d,aes(x=xx,y=yy)) + geom_point() +
               ggtitle("") + xlab(xvar) + ylab(yvar)
      } else {
          p <- ggplot(d,aes(x=xx,y=yy)) + geom_boxplot(notch=F) +
                 ggtitle("") + xlab(xvar) + ylab(yvar)
      }
    result <- c(result,list(p))
    }
    return(result)
}

grid.arrange(grobs=scatter.builder(cdigood))
```



The best possibilities for predicting `per.cap.income` are the same variables we identified from the correlation matrix: `pct.hs.grad`, `pct.bach.deg`,`pct.below.pov`, and `pct.unemp`. The last plot shows how `per.cap.income` varies across the four regions of the country. There is a lot of overlap in the boxplots, but the Northeast and the West seem to be doing a little better than the North Central and South regions.

1.1.4 You were not asked to do any transofrmations at this point, but it is a good idea to stop and think about transformations for a minute anyway...

I am not going to work too hard on transformations for now. I am going to take the logarithms of the variables I identified earlier, to address heavy skewing and potential leverage and influence issues. But I will leave the other variables alone, even though there may be some skew in them. I can explain the logarithms to the social scientist in terms of percent-change concepts, as we have done in class. And the more untransformed variables there are, the easier it will be for the social scientist to think about the model(s) I present.

(Another plausible approach would be to observe that essentially all of the continuous variables are skewed to some extent, and so one could just log all of them for starters.)

```
cdilogs <- cdigood

skewed.vars <- c("land.area", "pop", "doctors", "hosp.beds", "crimes", "tot.income",
                 "per.cap.income")

for (tmp in skewed.vars) {
  loc <- grep(paste("^",tmp,"$",sep=""),names(cdilogs))
    ## note special characters "^" and "$" to anchor
    ## the search pattern at the begining and end of the
    ## string
  cdilogs[,loc] <- log(cdilogs[,loc])
  names(cdilogs)[loc] <- paste("log.",names(cdilogs)[loc],sep="")
}

## Now I get to re-use those crazy functions I wrote...
```

```
grid.arrange(grobs=hist.builder(cdilogs))
```

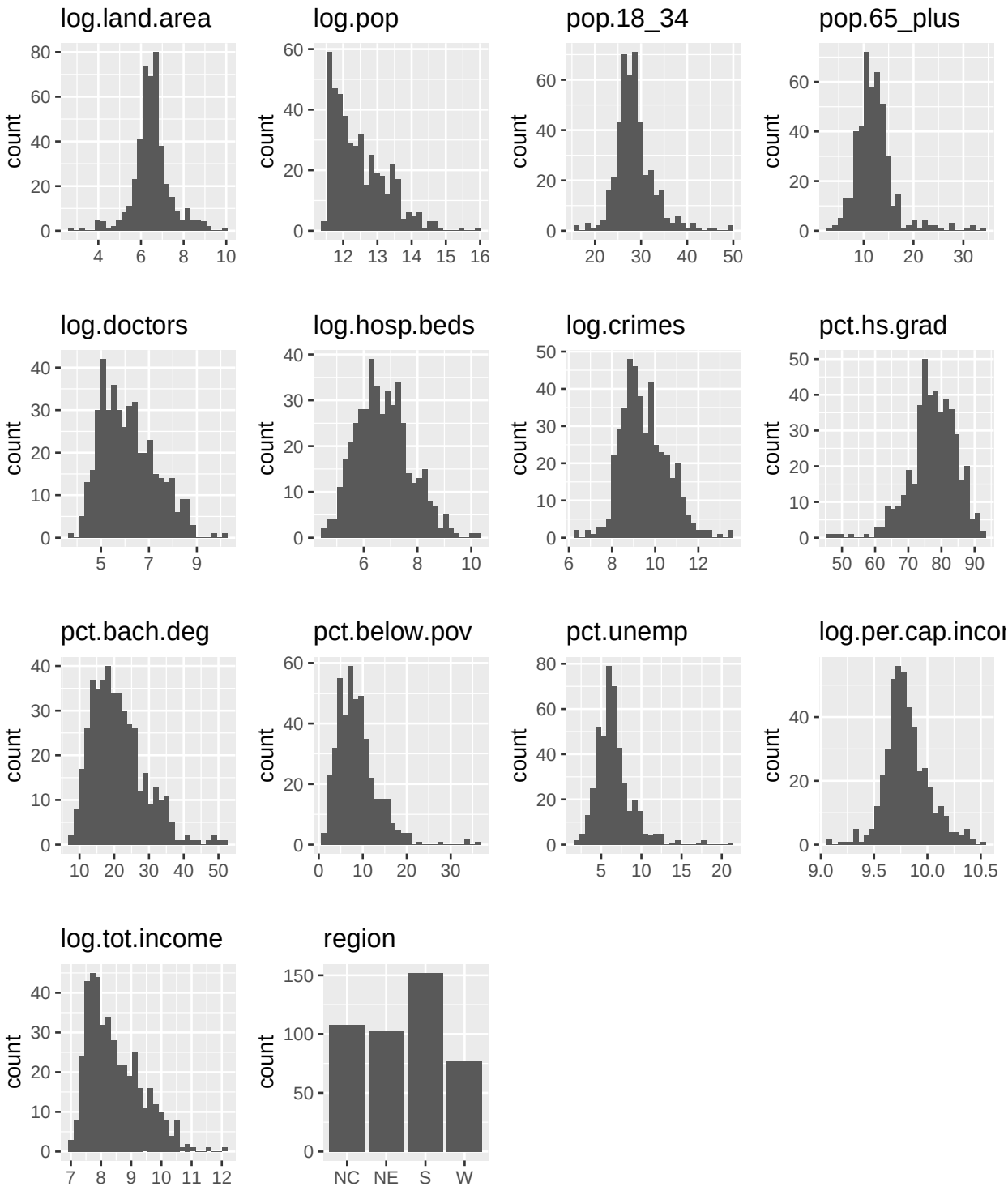


Figure 2: Histograms after log transformations

```
corgraph(cdilogs[,-grep("region",names(cdilogs))])
```

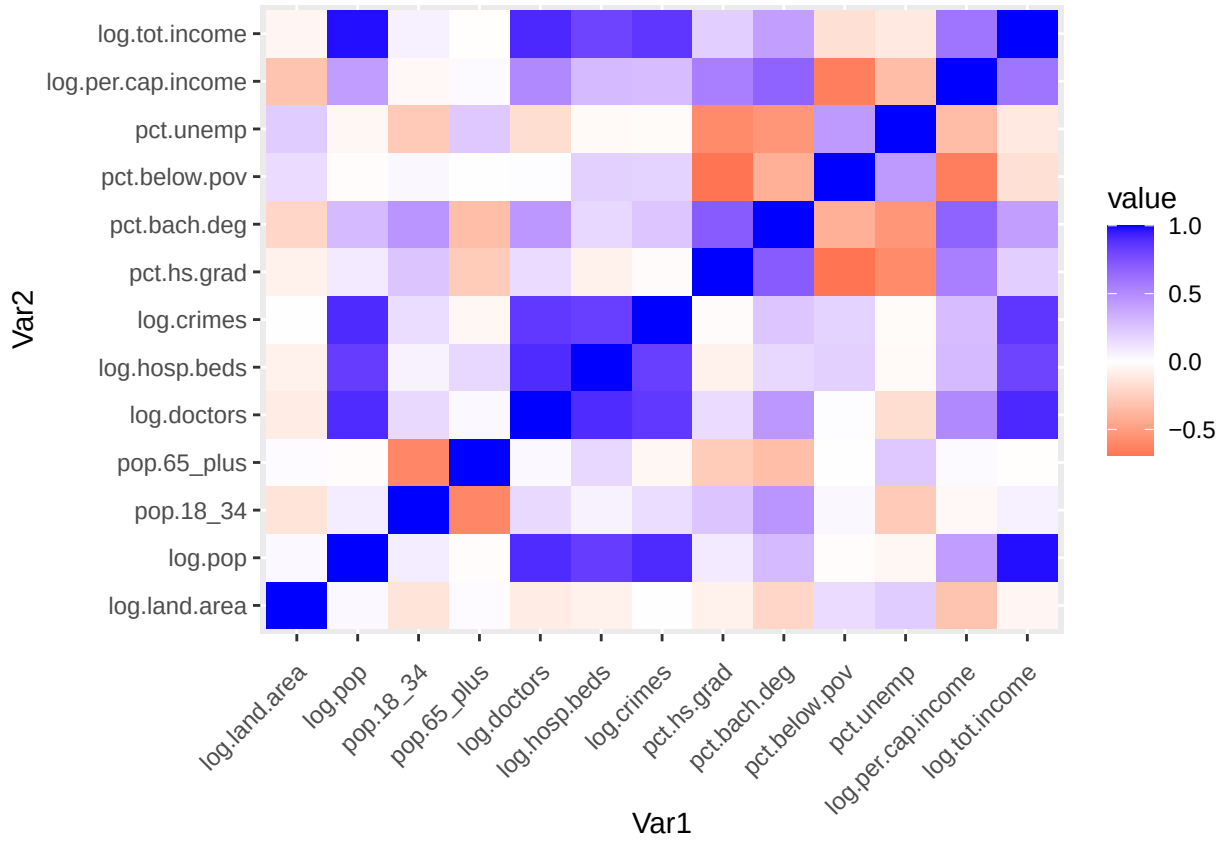


Figure 3: Correlations after log transformations

```
grid.arrange(grobs=scatter.builder(cdilogs,"log.per.cap.income"))
```

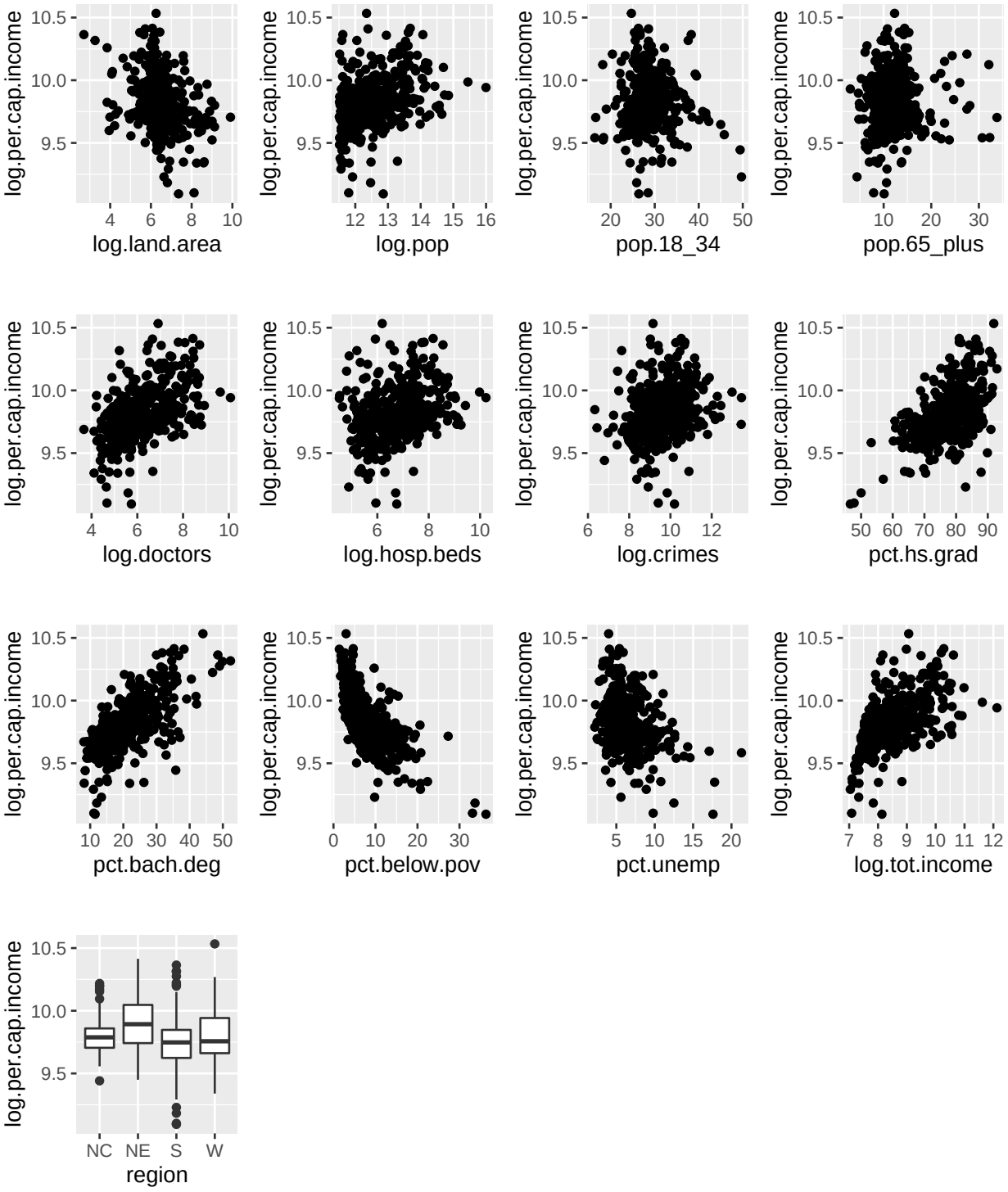


Figure 4: Scatterplots after log transformations

Except for `log.pop` (which I probably won't use much anyway, since `per.cap.income` = `tot.income` / `pop` is a deterministic function of `pop`), the skewing seems to have largely been brought under control. The

correlations are similar to, but a little stronger than, the correlations for the untransformed variables, and we can see from the scatter plots that the linear relationships we thought were there seem stronger, and there may be some other relationships to pay attention to.

1.2 Part B

Build a regression model that predicts per-capita income from crime rate and region of the country. Should there be any interactions in the model? What does your model say about the relationship between per- capita income and crime rate? Do your answers change, depending on whether you use number of crimes, or "per-capita crime" = (number of crimes)/(population) as a crime rate measure? If so, which one best answers the question? Why? Show the fitted model results and explain your answer to these questions in terms of those results.

For each version of the income and crime variables, there are essentially three models to think about. Since logarithms cleaned up a lot of the skewing in the data, I will only use log-transformed variables.

For log.per.cap.income and log.crimes, the three models to consider are:

```
ancova.01 <- lm(log.per.cap.income ~ log.crimes,data=cdilogs)

ancova.02 <- lm(log.per.cap.income ~ log.crimes + region,data=cdilogs)

ancova.03 <- lm(log.per.cap.income ~ log.crimes * region,data=cdilogs)
```

Residual plots appear in Figure 5 below. None of them are awful, so we could trust F-tests to compare these models:

```
anova(ancova.01,ancova.02, ancova.03)
```

```
## Analysis of Variance Table
##
## Model 1: log.per.cap.income ~ log.crimes
## Model 2: log.per.cap.income ~ log.crimes + region
## Model 3: log.per.cap.income ~ log.crimes * region
##   Res.Df    RSS Df Sum of Sq       F    Pr(>F)
## 1    438 17.271
## 2    435 14.949  3   2.32194 22.4823 1.523e-13 ***
## 3    432 14.872  3   0.07678  0.7434    0.5266
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It looks like ancova.02 (the additive, or "parallel lines" ANCOVA model with no interactions) is doing the best.

In order to compare this with a model involving "per-capita crime" = (number of crimes)/(population), we have to construct the new variable:

```
attach(cdigood)
per.cap.crime <- crimes/pop
log.per.cap.crimes <-log(per.cap.crime)
detach()
```

Note that I could equally well have calculated

```
attach(cdigood)
log.per.cap.crimes <-log(crimes) - log(pop)
detach()
```

Now let's look at the same three models with this new variable:

```
ancova.04 <- lm(log.per.cap.income ~ log.per.cap.crimes,data=cdilogs)

ancova.05 <- lm(log.per.cap.income ~ log.per.cap.crimes + region,data=cdilogs)

ancova.06 <- lm(log.per.cap.income ~ log.per.cap.crimes * region,data=cdilogs)
```

Again, the residual plots in Figure 5 look OK, so we try F-tests to compare these nested models

```
anova(ancova.04,ancova.05, ancova.06)
```

```
## Analysis of Variance Table
##
## Model 1: log.per.cap.income ~ log.per.cap.crimes
## Model 2: log.per.cap.income ~ log.per.cap.crimes + region
## Model 3: log.per.cap.income ~ log.per.cap.crimes * region
##   Res.Df    RSS Df Sum of Sq       F    Pr(>F)
## 1    438 18.697
## 2    435 16.952  3   1.74465 14.8407 3.263e-09 ***
## 3    432 16.928  3   0.02408  0.2048     0.893
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Now ancova.05 (once again, the additive or "parallel lines" model) is doing the best.

If we want to compare these two winners (ancova.02 vs. ancova.05), we need to use AIC or BIC, because the two winners are not nested models (you can't get one from the other by imposing one or more linear constraints).

```
AIC(ancova.02,ancova.05)
```

```
##           df       AIC
## ancova.02  6 -227.4746
## ancova.05  6 -172.1347
```

```
BIC(ancova.02,ancova.05)
```

```
##           df       BIC
## ancova.02  6 -202.9539
## ancova.05  6 -147.6140
```

And so it seems ancova.02 is the best model, by either AIC or BIC (remember smaller is better!).

Recalling the model formula for ancova.02 and a brief summary of the model

```
formula(ancova.02)
```

```
## log.per.cap.income ~ log.crimes + region
```

```
round(coef(summary(ancova.02)),2)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)     9.19       0.08  115.13     0.00
## log.crimes      0.07       0.01    7.92     0.00
## regionNE        0.10       0.03    4.09     0.00
## regionS        -0.09       0.02   -3.68     0.00
## regionW        -0.06       0.03   -1.96     0.05
```

we can interpret the model as follows (using the handout "log xform and percent interpretation.pdf" from week03):

- All across the US, for every 1% increase in crimes, we expect a 0.07% increase in per-capita income, om average (this increase is statistically significant, but is it practically significant?).
- Different regions of the country have different baseline per-capita incomes however: In the NC region, the baseline salary is `exp(9.19) =` \$9,798.65. In the NE it is `exp(9.19+0.010) =` \$10,829.18, and so forth, so in the S it is \$8,955.29, and in the W it is \$9,228.02. All of these region baselines are, according to the model, significantly different from the NC baseline.
- So, according to the model, the *level* of salary varies with region in the US, but the *way it is related to crime* does not.

There is an argument that per-capita crime is more comparable to, or at least on the same scale as, per-capita income, so we will briefly look at the second-best model, anova.05, to see how it compares to ancova.02:

```
formula(ancova.05)
```

```
## log.per.cap.income ~ log.per.cap.crimes + region
```

```
round(coef(summary(ancova.05)),2)
```

```
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)            9.94       0.07  143.30     0.00
## log.per.cap.crimes     0.04       0.02    1.98     0.05
## regionNE               0.11       0.03    4.15     0.00
## regionS               -0.07       0.03   -2.84     0.00
## regionW               -0.02       0.03   -0.81     0.42
```

The interpretation of this model is similar to that of ancova.02:

- All across the US, for every 1% increase in per-capita crime, there is an associated 0.04% increase in per-capita income (so, slightly smaller effect, but still statistically significant; should that matter?).
- The regional baseline salaries are: NC: \$20,743.74, NE: \$23,155.79, S: is \$19,341.34, and W: \$20,332.99. All but the W region have baselines that are, according to the model, significantly different from the NC baseline.
- Again, the level of salary varies with region, but not the way it varies with crime, according to the model.

One would have to choose which of these two models one would want to feature (and interpret!) in the main body of the IDMRAD paper...

	df	AIC	BIC
ancova.01	3	-169.9466	-157.6863
ancova.02	6	-227.4746	-202.9539
ancova.03	9	-223.7402	-186.9593
ancova.04	3	-135.0340	-122.7737
ancova.05	6	-172.1347	-147.6140
ancova.06	9	-166.7601	-129.9792

ASIDE:

We could also just have compared all 6 models with AIC or BIC. In this case, we get the same result: the additive model ancova.02 has both the lowest AIC and the lowest BIC values.

```
data.frame(AIC=AIC(ancova.01,ancova.02,ancova.03,ancova.04,ancova.05,ancova.06),
BIC=BIC(ancova.01,ancova.02,ancova.03,ancova.04,ancova.05,ancova.06))[,-3] %>%
  kbl(booktabs=T,col.names=c("df","AIC","BIC")) %>% kable_classic(full_width=F)
```

RESIDUAL PLOTS:

And finally, here are the residual disgnostic plots that justified our earlier use of Anova F-tests to compare nested models.

```
oldmar <- par()$mar

par(mfrow=c(6,4))

par(mar=c(2,2,2,2))

invisible(lapply(list(ancova.01,ancova.02,ancova.03,ancova.04,ancova.05,ancova.06),
                 function(x) plot(x,cex.main=0.5)))
```

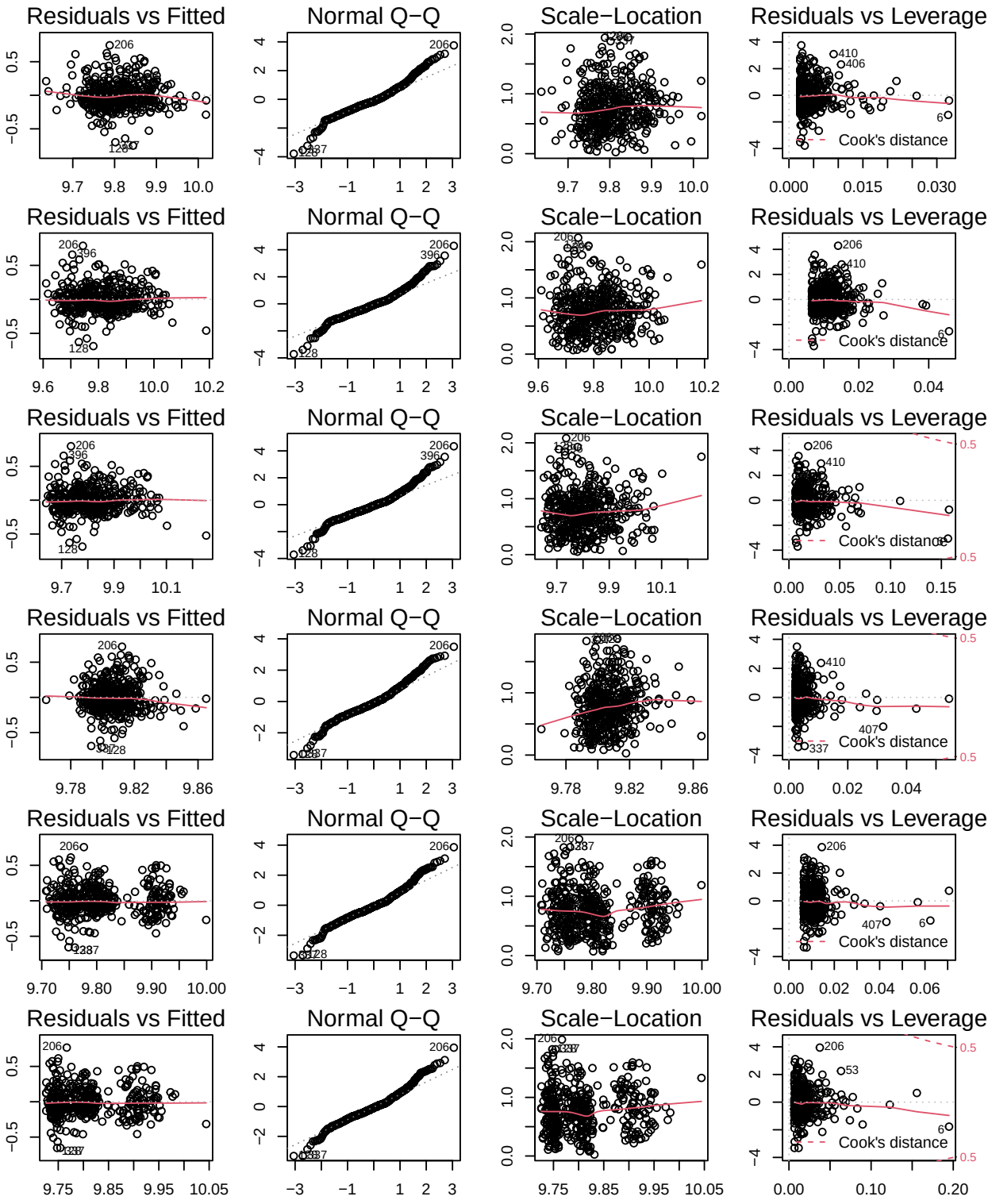


Figure 5: Residual diagnostics for all 6 ANCOVA models, in order: ancova.01, ancova.02, ancova.03, ancova.04, ancova.05, ancova.06

```
par(mar=oldmar)
```

1.3 Part C

Use methods we have discussed in class and/or methods from Sheather Chapters 5, 6 & 7 (including, as needed: transformations, interactions, variable selection, residual analysis, fit indices, etc.) to find the multiple regression model predicting per-capita income from the other variables, that makes the "best" tradeoff between the following criteria:

- **Reflects the social science and the meaning of the variables**
- **Satisfies modeling assumptions**
- **Clearly indicated by the data**
- **Can be explained to someone who is more interested in social, economic and health factors than in mathematics and statistics.**

No matter what you do, you are likely to be unhappy with some or all of these criteria; the better you make one criterion, the worse another is likely to get. So you will have to find a compromise or tradeoff between these criteria. Explain how you decided to make the tradeoff(s) you made.

I am reasonably happy with the set of variables in the `cdilogs` data frame, so for now I will do model selection using just these variables. As shown in the first part of the answer to part (a), `id` and `county` are not useful predictors. `state` might be, but I am going to see how far I can get without `state`, and then perhaps incorprate it at the end.

Before beginning, I need to take log.pop and log.tot.income out of consideration, since per.cap.income is a deterministic function of them (so if they are included, no other predictors can possibly matter, and so I won't learn anything about what is associated with per.cap.income).

```
omit <- c(grep("log.pop",names(cdilogs)),grep("log.tot.income",names(cdilogs)))
cdilogred <- cdilogs[,-omit]
```

Because `region` is categorical, and almost none of our variable selection functions are smart about dealing with the group of indicator variables associated with a categorical variable, I am going to first work *without* the `region` variable, and then I will try to include the `region` variable in some intelligent way.

```
cdilogred.cont <- cdilogred[,-grep("region",names(cdilogred))]
## a data set to work with that doesn't have 'region' in it -- only the continuous variables...
names(cdilogred.cont)
```

```
##  [1] "log.land.area"      "pop.18_34"          "pop.65_plus"
##  [4] "log.doctors"        "log.hosp.beds"      "log.crimes"
##  [7] "pct.hs.grad"        "pct.bach.deg"       "pct.below.pov"
## [10] "pct.unemp"          "log.per.cap.income"
```

```
## there are 10 predictor variables and the response variable log.per.cap.income...
```

ALL-SUBSETS:

I'll start with all-subsets regression.

```
library(leaps)  ## for the regsubsets() function
library(car)    ## for the subsets() plotting function
```

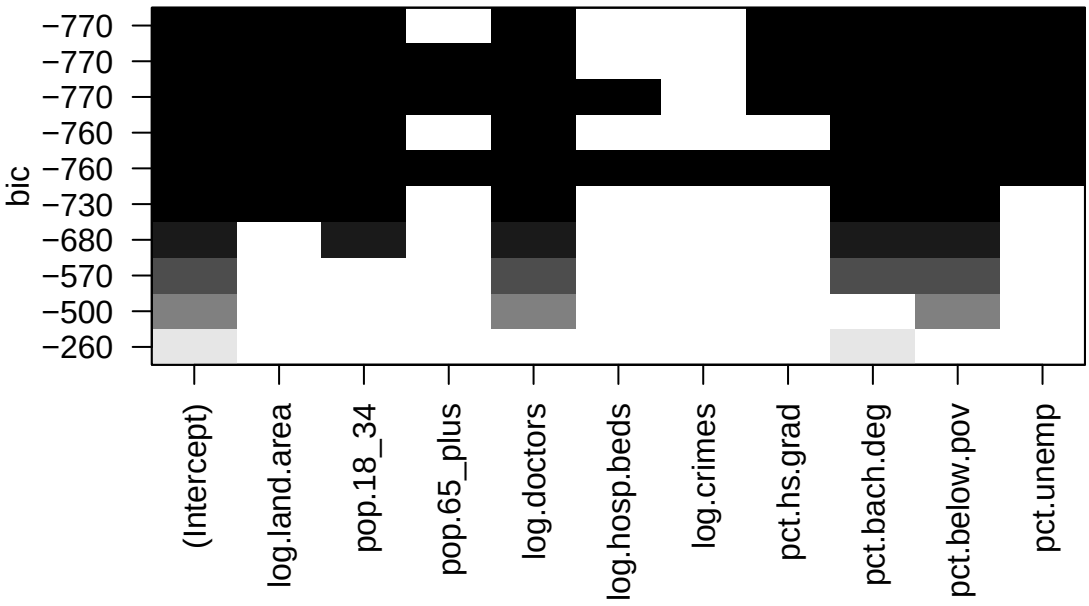
A little preliminary exploration convinces me of two things:

- There are too many variables for the subsets() function to be an effective way to summarize the results of regsubsets()

- The documentation and a bit of googling shows that I have some other tools to work with:
  - plot() makes a *different* graphical summary of regsubsets results that *is useful*
  - summary() of regsubsets results can be stored in a variable, and there are several named components which can be used to locate the "best" (i.e. lowest-BIC) model
  - coef() works on regsubsets results, if you can identify which model in the sequence of models from regsubsets you want

So... onward!

```
all.subsets.01 <- regsubsets(log.per.cap.income ~ ., data=cdilogred.cont,nvmax=10)
plot(all.subsets.01)
```



In the plot, the dark squares indicate which variables are in the model that has the BIC values on the left. The darker the squares, the better the model (remember, small is good for BIC). Rather than read them off the plot, though, we can use the components of summary() to get them directly.

We begin by locating the model with the lowest BIC...

```
all.subsets.01.summary <- summary(all.subsets.01)
names(all.subsets.01.summary)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```
all.subsets.01.summary$bic
```

```
##  [1] -257.5260 -502.4302 -572.5538 -682.8532 -732.1894 -761.5908 -772.0715
##  [8] -770.5990 -766.2235 -760.4131
```

```
min(all.subsets.01.summary$bic)
```

```
## [1] -772.0715
```

```
print(best.model <- which.min(all.subsets.01.summary$bic))
```

```
## [1] 7
```

```
coef(all.subsets.01,best.model)
```

```
##   (Intercept) log.land.area     pop.18_34   log.doctors   pct.hs.grad
##  10.222495041  -0.035674062  -0.013900201   0.060676872  -0.004406396
##  pct.bach.deg pct.below.pov     pct.unemp
##   0.015385301  -0.024278371   0.010603691
```

Note that the variables listed here are the same as the ones at the top of the graph we made.

Before we interpret the coefficients, let's refit the model so we can get standard errors, etc. To make life easy, we'll set up a temporary data frame with just the variables from the lowest-BIC model.

```
all.subsets.01.summary$which[best.model,]
```

```
##   (Intercept) log.land.area     pop.18_34   pop.65_plus   log.doctors
##          TRUE          TRUE          TRUE         FALSE          TRUE
## log.hosp.beds    log.crimes   pct.hs.grad  pct.bach.deg pct.below.pov
##         FALSE         FALSE          TRUE          TRUE          TRUE
##     pct.unemp
##          TRUE
```

```
tmp <- cdilogred.cont[,all.subsets.01.summary$which[best.model,][-1]]
all.subsets.01.final.model <- lm(log.per.cap.income ~ .,data=tmp)
summary(all.subsets.01.final.model)$coef
```

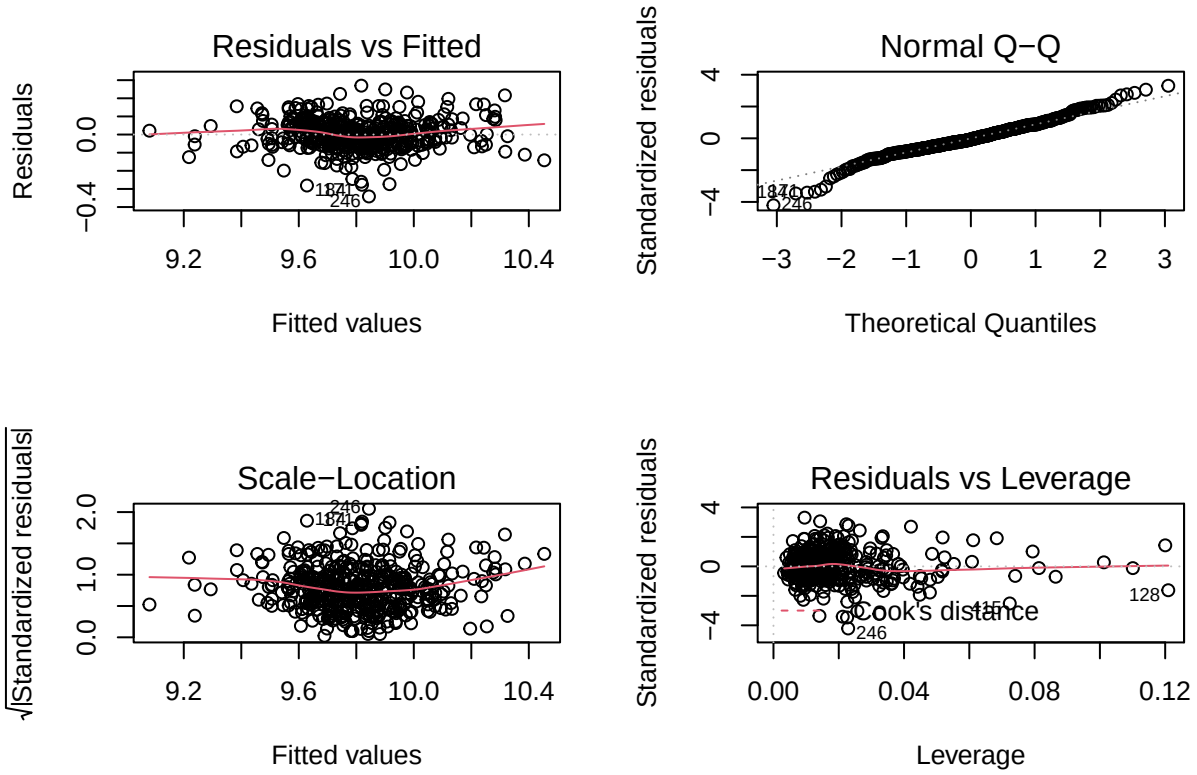
```
##                   Estimate   Std. Error    t value      Pr(>|t|)
## (Intercept)   10.222495041 0.0931210074 109.776465 1.127483e-317
## log.land.area -0.035674062 0.0047767371  -7.468291  4.533156e-13
## pop.18_34     -0.013900201 0.0011113007 -12.508046  7.514862e-31
## log.doctors    0.060676872 0.0040183327  15.100012  1.133432e-41
## pct.hs.grad   -0.004406396 0.0010822796  -4.071403  5.558448e-05
## pct.bach.deg   0.015385301 0.0009245509  16.640838  2.100590e-48
## pct.below.pov -0.024278371 0.0012583372 -19.294011  2.812246e-60
## pct.unemp      0.010603691 0.0021771148   4.870525  1.564524e-06
```

All the predictors have coefficients significantly different from zero. However, most of the coefficients are small, and some seem to have the wrong sign (e.g. `pct.hs.grad` and `pct.unemp`). Before diving into what might be going on, let's at least check VIFs and residual diagnostics...

```
vif(all.subsets.01.final.model)
```

```
## log.land.area     pop.18_34   log.doctors   pct.hs.grad  pct.bach.deg
##      1.131867      1.416145      1.379671      3.763103      3.269565
## pct.below.pov     pct.unemp
##      2.241555      1.691280
```

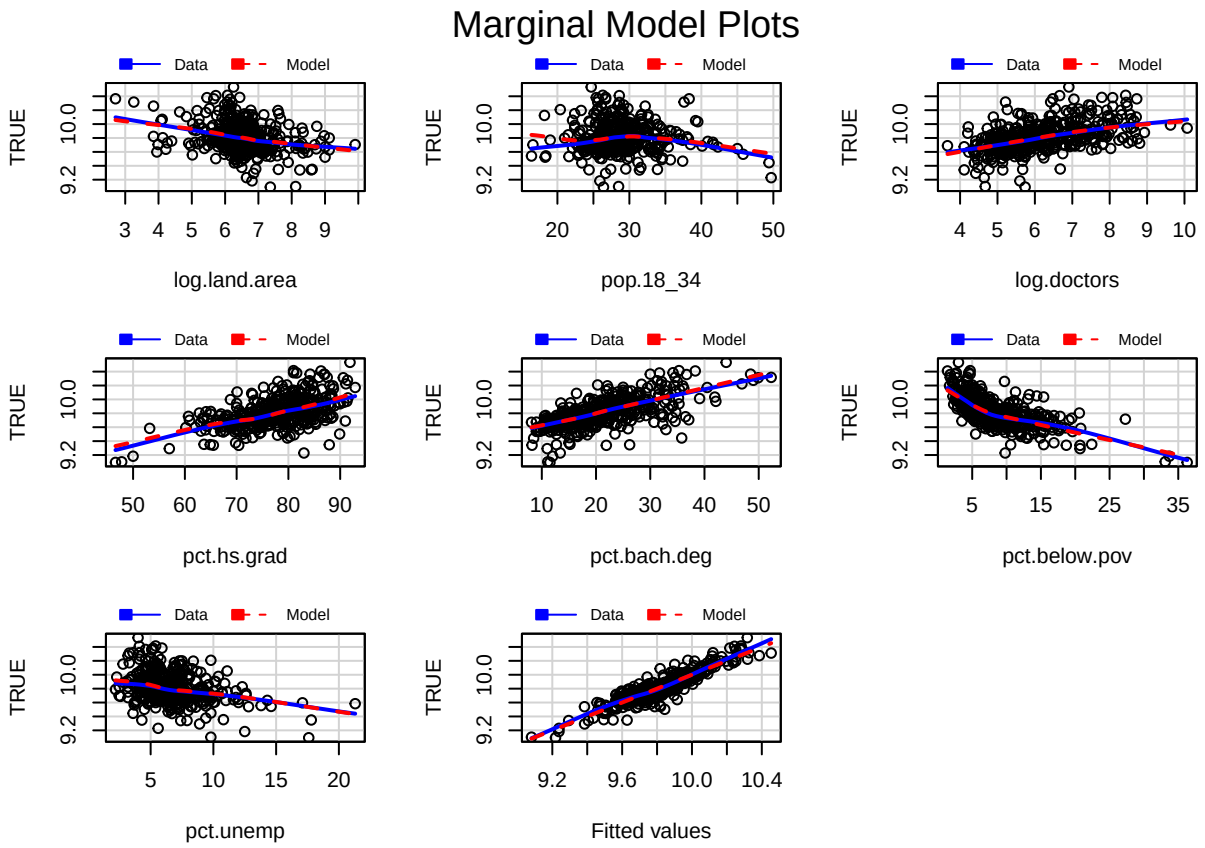
```
par(mfrow=c(2,2))
plot(all.subsets.01.final.model)
```



None of the VIFs seem excessively large, and the diagnostic plots don't show much except that the QQ plot suggests both the left and the right tails are a bit longer than expected for the normal distribution.

We can also look at marginal model plots, to see if we are missing a transformation, interaction, etc.

```
mmps(all.subsets.01.final.model)
```



The marginal model plots look very good – the blue data-based curves line up well with the red model-based curves. We don't seem to be missing any important transformations, interactions, etc.

One last thing to try is to see if interaction with `region` helps in any way.

```
tmp <- cbind(tmp,region=cdilogred$region)
all.subsets.01.final.with.region <- lm(log.per.cap.income ~ .*region,data=tmp)
summary(all.subsets.01.final.with.region)
```

```
##
## Call:
## lm(formula = log.per.cap.income ~ . * region, data = tmp)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.250782 -0.042332 -0.002298  0.040559  0.313570
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            10.1244260  0.2826240  35.823  < 2e-16 ***
## log.land.area          -0.0364187  0.0151355  -2.406 0.016564 *
## pop.18_34              -0.0147940  0.0026043  -5.681 2.55e-08 ***
## log.doctors             0.0544169  0.0093221   5.837 1.08e-08 ***
## pct.hs.grad            -0.0024773  0.0034110  -0.726 0.468088
## pct.bach.deg            0.0140833  0.0029254   4.814 2.09e-06 ***
## pct.below.pov          -0.0237085  0.0036234  -6.543 1.81e-10 ***
## pct.unemp               0.0180393  0.0048923   3.687 0.000257 ***
## regionNE                0.3243992  0.3577081   0.907 0.365004
```

```
## regionS                -0.0345856  0.3131668  -0.110 0.912116
## regionW                 1.5043946  0.4226868   3.559 0.000416 ***
## log.land.area:regionNE -0.0037179  0.0201435  -0.185 0.853656
## log.land.area:regionS  -0.0047582  0.0174155  -0.273 0.784825
## log.land.area:regionW   0.0151234  0.0181871   0.832 0.406154
## pop.18_34:regionNE     -0.0024780  0.0036873  -0.672 0.501939
## pop.18_34:regionS      -0.0008777  0.0030680  -0.286 0.774970
## pop.18_34:regionW       0.0014122  0.0040925   0.345 0.730220
## log.doctors:regionNE   -0.0046251  0.0132571  -0.349 0.727359
## log.doctors:regionS     0.0043337  0.0114401   0.379 0.705019
## log.doctors:regionW    -0.0034863  0.0131576  -0.265 0.791173
## pct.hs.grad:regionNE   -0.0037529  0.0044150  -0.850 0.395813
## pct.hs.grad:regionS     0.0021198  0.0037853   0.560 0.575790
## pct.hs.grad:regionW    -0.0190188  0.0045881  -4.145 4.13e-05 ***
## pct.bach.deg:regionNE   0.0069429  0.0040312   1.722 0.085776 .
## pct.bach.deg:regionS   -0.0015774  0.0032000  -0.493 0.622328
## pct.bach.deg:regionW    0.0071026  0.0036374   1.953 0.051541 .
## pct.below.pov:regionNE -0.0014134  0.0050896  -0.278 0.781381
## pct.below.pov:regionS   0.0072764  0.0040739   1.786 0.074827 .
## pct.below.pov:regionW  -0.0161639  0.0054271  -2.978 0.003071 **
## pct.unemp:regionNE     -0.0083596  0.0073758  -1.133 0.257720
## pct.unemp:regionS      -0.0249396  0.0065867  -3.786 0.000176 ***
## pct.unemp:regionW      -0.0201466  0.0067713  -2.975 0.003101 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0759 on 408 degrees of freedom
## Multiple R-squared:  0.8747, Adjusted R-squared:  0.8652
## F-statistic: 91.91 on 31 and 408 DF,  p-value: < 2.2e-16
```

A reasonable rule of thumb is: if any indicator for a categorical variable seems important (e.g. a statistically significant coefficient), then keep the whole categorical variable. If none of them seem important, then drop the variable. The same thing works for interactions with categorical variables.

(A more formally correct thing to do would be to check to see whether to include each group of variables involving ***region*** *with partial F tests (the anova() function), AIC, and/or BIC, in a kind of stepwise regression procedure.)*

Using the rule of thumb, we would

- Keep: `region`, `region:pct.hs.grad`, `region:pct.below.pov`, `region:pct.unemp`, and maybe `region:pct.bach.deg`
- Drop: `region:log.land.area`, `region:pop.18_34`, `region:log.doctors`

Thus we arrive at

```
all.subsets.01.final.with.some.region <- update(all.subsets.01.final.with.region,
                                                . ~ . - region:log.land.area -
                                                  region:pop.18_34 - region:log.doctors)
summary(all.subsets.01.final.with.some.region)
```

```
##
## Call:
## lm(formula = log.per.cap.income ~ log.land.area + pop.18_34 +
##     log.doctors + pct.hs.grad + pct.bach.deg + pct.below.pov +
##     pct.unemp + region + pct.hs.grad:region + pct.bach.deg:region +
##     pct.below.pov:region + pct.unemp:region, data = tmp)
```

```
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.268015 -0.043459 -0.002511  0.039967  0.313939
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             10.125260   0.251582  40.246  < 2e-16 ***
## log.land.area           -0.034569   0.005376  -6.430 3.50e-10 ***
## pop.18_34               -0.015404   0.001087 -14.170  < 2e-16 ***
## log.doctors              0.055342   0.004034  13.720  < 2e-16 ***
## pct.hs.grad             -0.002503   0.003151  -0.794 0.427456
## pct.bach.deg             0.014208   0.002108   6.741 5.24e-11 ***
## pct.below.pov           -0.023634   0.003351  -7.054 7.30e-12 ***
## pct.unemp                0.017787   0.004783   3.719 0.000228 ***
## regionNE                 0.219429   0.302526   0.725 0.468661
## regionS                 -0.062648   0.276125  -0.227 0.820627
## regionW                  1.629351   0.357633   4.556 6.86e-06 ***
## pct.hs.grad:regionNE    -0.003640   0.003876  -0.939 0.348271
## pct.hs.grad:regionS      0.002014   0.003539   0.569 0.569690
## pct.hs.grad:regionW     -0.018916   0.004204  -4.499 8.85e-06 ***
## pct.bach.deg:regionNE    0.005905   0.002618   2.256 0.024611 *
## pct.bach.deg:regionS    -0.001298   0.002321  -0.559 0.576352
## pct.bach.deg:regionW     0.006326   0.002620   2.415 0.016183 *
## pct.below.pov:regionNE -0.002435   0.004647  -0.524 0.600488
## pct.below.pov:regionS    0.007137   0.003686   1.937 0.053482 .
## pct.below.pov:regionW   -0.015224   0.005169  -2.945 0.003407 **
## pct.unemp:regionNE      -0.007967   0.007255  -1.098 0.272761
## pct.unemp:regionS       -0.024668   0.006377  -3.868 0.000127 ***
## pct.unemp:regionW       -0.019757   0.006603  -2.992 0.002935 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07545 on 417 degrees of freedom
## Multiple R-squared:  0.8735, Adjusted R-squared:  0.8668
## F-statistic: 130.9 on 22 and 417 DF,  p-value: < 2.2e-16
```

This seems to be working... All of the main effects and interactions that involve `region` have at least one significant term, and the R^2 and residual SE hardly moved at all.

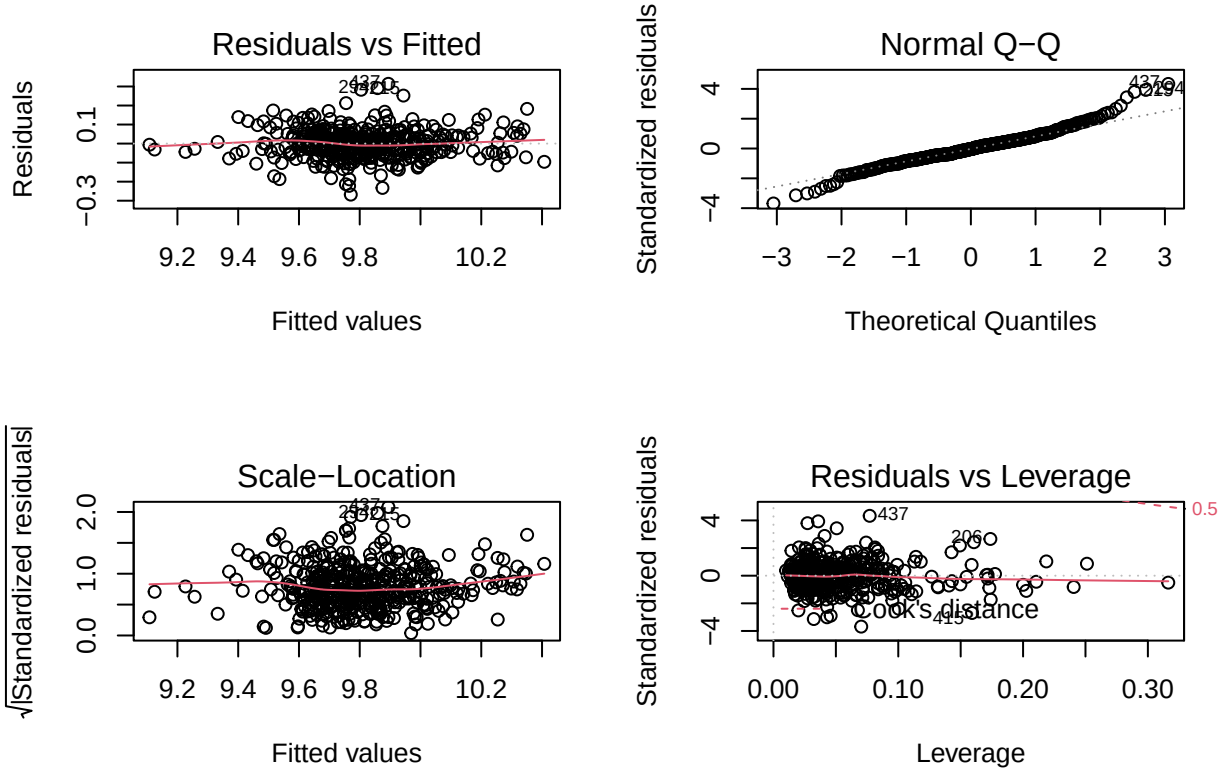
Let's quickly check VIFs and diagnostics, and then compare to the best BIC model that we obtained from stepwise without the `region` term.

```
vif(all.subsets.01.final.with.some.region)
```

```
##                              GVIF Df GVIF^(1/(2*Df))
## log.land.area        1.693937e+00  1        1.301513
## pop.18_34            1.600959e+00  1        1.265290
## log.doctors          1.642537e+00  1        1.281615
## pct.hs.grad          3.768383e+01  1        6.138716
## pct.bach.deg         2.007247e+01  1        4.480231
## pct.below.pov        1.877727e+01  1        4.333275
## pct.unemp            9.644220e+00  1        3.105515
## region               4.181419e+08  3       27.345589
## pct.hs.grad:region   3.373864e+08  3       26.384848
## pct.bach.deg:region  2.600598e+04  3        5.443090
```

```
## pct.below.pov:region 6.114866e+03  3        4.276264
## pct.unemp:region     1.364209e+04  3        4.888176
```

```
par(mfrow=c(2,2))
plot(all.subsets.01.final.with.some.region)
```



As usual, adding interactions usually creates collinearities (the interactions are necessarily somewhat collinear with their main effects). We worry less about this, because the collinearities do not appear to be causing the t-statistics and p-values in the model summary to behave strangely.

The diagnostic plots look about as good or as bad as the last set we looked at. It seems to me that in this analysis, the residual diagnostic plots are always going the toughest to feel totally satisfied about.

Now let's compare our model with some region terms to the model we got from all-subsets, but with region not included:

```
anova(all.subsets.01.final.model,all.subsets.01.final.with.some.region)
```

```
## Analysis of Variance Table
##
## Model 1: log.per.cap.income ~ log.land.area + pop.18_34 + log.doctors +
##     pct.hs.grad + pct.bach.deg + pct.below.pov + pct.unemp
## Model 2: log.per.cap.income ~ log.land.area + pop.18_34 + log.doctors +
##     pct.hs.grad + pct.bach.deg + pct.below.pov + pct.unemp +
##     region + pct.hs.grad:region + pct.bach.deg:region + pct.below.pov:region +
##     pct.unemp:region
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    432 2.9051
## 2    417 2.3736 15   0.53148 6.2249 6.673e-12 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
AIC(all.subsets.01.final.model,all.subsets.01.final.with.some.region)
```

```
##                                       df       AIC
## all.subsets.01.final.model             9  -942.274
## all.subsets.01.final.with.some.region 24 -1001.178
```

```
BIC(all.subsets.01.final.model,all.subsets.01.final.with.some.region)
```

```
##                                       df       BIC
## all.subsets.01.final.model             9 -905.4931
## all.subsets.01.final.with.some.region 24 -903.0957
```

The anova (F test) and AIC really like the model with the region terms in it! On the other hand, BIC prefers the simpler model. This often happens: BIC chooses a simpler model than other methods, because it is going for a parsimonius explanatory model, rather than a highly predictive model.

Finally, let's examine the table of estimated coefficients again for the model with some region terms in it, and offer some interpretations...

```
formula(all.subsets.01.final.with.some.region)
```

```
## log.per.cap.income ~ log.land.area + pop.18_34 + log.doctors +
##     pct.hs.grad + pct.bach.deg + pct.below.pov + pct.unemp +
##     region + pct.hs.grad:region + pct.bach.deg:region + pct.below.pov:region +
##     pct.unemp:region
```

```
round(summary(all.subsets.01.final.with.some.region)$coef,2)
```

```
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 10.13       0.25   40.25     0.00
## log.land.area               -0.03       0.01   -6.43     0.00
## pop.18_34                   -0.02       0.00  -14.17     0.00
## log.doctors                  0.06       0.00   13.72     0.00
## pct.hs.grad                  0.00       0.00   -0.79     0.43
## pct.bach.deg                 0.01       0.00    6.74     0.00
## pct.below.pov               -0.02       0.00   -7.05     0.00
## pct.unemp                    0.02       0.00    3.72     0.00
## regionNE                     0.22       0.30    0.73     0.47
## regionS                     -0.06       0.28   -0.23     0.82
## regionW                      1.63       0.36    4.56     0.00
## pct.hs.grad:regionNE         0.00       0.00   -0.94     0.35
## pct.hs.grad:regionS          0.00       0.00    0.57     0.57
## pct.hs.grad:regionW         -0.02       0.00   -4.50     0.00
## pct.bach.deg:regionNE        0.01       0.00    2.26     0.02
## pct.bach.deg:regionS         0.00       0.00   -0.56     0.58
## pct.bach.deg:regionW         0.01       0.00    2.41     0.02
## pct.below.pov:regionNE       0.00       0.00   -0.52     0.60
## pct.below.pov:regionS        0.01       0.00    1.94     0.05
## pct.below.pov:regionW       -0.02       0.01   -2.95     0.00
## pct.unemp:regionNE          -0.01       0.01   -1.10     0.27
## pct.unemp:regionS           -0.02       0.01   -3.87     0.00
## pct.unemp:regionW           -0.02       0.01   -2.99     0.00
```

Here are some things we can say (this is only a few examples, there are more things to say!):

- For every 1% increase in a county's land area, there is a 0.03% decrease in expected per-capita income.

(We might conjecture that this could be due to an urban-rural contrast: rural counties tend to be bigger than urban ones).

- For every 1% increase in the number of doctors in a county, the expected per-capita income increases by about 0.06%. That makes sense; doctors are well-paid and could be big contributors to the per-capita average income.
- For every 1 percentage point increase in the percent of the population aged 18–34, there iz an expected 2% drop in per-capita income. (We might conjecture that this is because 18–34 year olds are not at peak earning capacity yet and so perhaps their lower incomes drags down the per-capita average).
- percent of the population that are high school graduates doesn't have much effect, except in the South, where a one percentage point increase in hs graduates induces an expected 2% decrease in per-capita income. I don't hae a good explanation for this: It might depend on whether college graduates are counted as a subset of hs graduates rather than counting them separately, or it might have something to do with some unique feature of economics in the southern region of the US.
- In the main effect for region, and in several of the interactions for region, the West shows up as deviating significantly from the North Central part of the US
- etc.

STEPWISE REGRESSION:

Now let's try to see how far we get with stepwise regression. I am going to take the same strategy: first try stepwise without `region`, and then see if adding `region` helps.

```
library(MASS)
```

```
stepwise.base <- lm(log.per.cap.income ~ .,data=cdilogred.cont)
## try to duplicate all-subsets with BIC
step.result.01.bic <- stepAIC(stepwise.base,
                      scope=list(lower = ~ 1, upper = ~ .),
                      k=log(dim(cdilogred.cont)[1]),
                      trace=F)

anova(all.subsets.01.final.model,step.result.01.bic)
```

```
## Analysis of Variance Table
##
## Model 1: log.per.cap.income ~ log.land.area + pop.18_34 + log.doctors +
##     pct.hs.grad + pct.bach.deg + pct.below.pov + pct.unemp
## Model 2: log.per.cap.income ~ log.land.area + pop.18_34 + log.doctors +
##     pct.hs.grad + pct.bach.deg + pct.below.pov + pct.unemp
##   Res.Df    RSS Df Sum of Sq F Pr(>F)
## 1    432 2.9051
## 2    432 2.9051  0         0
```

We can see that stepwise regression using the BIC criterion actually found exactly the all-subsets model, which is great.

```
## now try with AIC
step.result.01.aic <- stepAIC(stepwise.base,
                      scope=list(lower = ~ 1, upper = ~ .),
                      k=2,
                      trace=F)

anova(all.subsets.01.final.model,step.result.01.aic)
```

```
## Analysis of Variance Table
```

```
##
## Model 1: log.per.cap.income ~ log.land.area + pop.18_34 + log.doctors +
##     pct.hs.grad + pct.bach.deg + pct.below.pov + pct.unemp
## Model 2: log.per.cap.income ~ log.land.area + pop.18_34 + pop.65_plus +
##     log.doctors + pct.hs.grad + pct.bach.deg + pct.below.pov +
##     pct.unemp
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1    432 2.9051
## 2    431 2.8748  1  0.030306 4.5437 0.03361 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Stepwise regression using the AIC criterion added one more predictor, namely `pop.65_plus`. Looking at the coefficient estimates below, we can say that, although the coefficient on `pop.65_plus` is significantly different from zero, its effect on expected per-capita income appears to be quite small. Without knowing more about the economics, it's hard to make a decision about whether to keep `pop.65_plus` or not.

```
round(summary(step.result.01.aic)$coef,2)
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)      10.32        0.1  100.56     0.00
## log.land.area    -0.04        0.0   -7.65     0.00
## pop.18_34        -0.02        0.0  -11.82     0.00
## pop.65_plus       0.00        0.0   -2.13     0.03
## log.doctors       0.06        0.0   15.26     0.00
## pct.hs.grad       0.00        0.0   -4.30     0.00
## pct.bach.deg      0.02        0.0   16.46     0.00
## pct.below.pov    -0.02        0.0  -19.49     0.00
## pct.unemp         0.01        0.0    4.96     0.00
```

Well, now let's go for broke and see what happens if we just explore 2-way interactions briefly.

```
step.result.02.bic <- stepAIC(stepwise.base,scope=list(lower = ~ 1, upper = ~ .^2),
                      k=log(dim(cdilogred.cont)[1]),                          ## BIC penalty.
                      trace=F)

step.result.02.aic <- stepAIC(stepwise.base,scope=list(lower = ~ 1, upper = ~ .^2),
                      k=2,                                                    ## AIC penalty.
                      trace=F)

comparison <- cbind(
AIC(all.subsets.01.final.model,step.result.01.aic,step.result.01.bic,
    step.result.02.aic,step.result.02.bic),
BIC(all.subsets.01.final.model,step.result.01.aic,step.result.01.bic,
    step.result.02.aic,step.result.02.bic))
comparison <- comparison[,-3]
names(comparison) <- c("df","AIC","BIC")
comparison %>% kbl(booktabs=T) %>% kable_classic()
```

As you can see in the table, both AIC and BIC like models with some interactions. This is a quandary: now the model is getting complicated to explain to someone.

	df	AIC	BIC
all.subsets.01.final.model	9	-942.2740	-905.4931
step.result.01.aic	10	-944.8883	-904.0206
step.result.01.bic	9	-942.2740	-905.4931
step.result.02.aic	27	-1064.7253	-954.3824
step.result.02.bic	12	-1020.6026	-971.5613

Let's look at the two models with interactons and see what we can say:

```
round(summary(step.result.02.bic)$coef,2)
```

```
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  10.83       0.13   83.19     0.00
## log.land.area                -0.03       0.00   -7.53     0.00
## pop.18_34                    -0.02       0.00  -13.09     0.00
## pop.65_plus                  -0.02       0.00   -8.81     0.00
## log.doctors                   0.05       0.00   13.29     0.00
## pct.hs.grad                  -0.01       0.00   -5.32     0.00
## pct.bach.deg                  0.01       0.00    3.01     0.00
## pct.below.pov                -0.02       0.00  -20.42     0.00
## pct.unemp                    -0.02       0.01   -1.83     0.07
## pop.65_plus:pct.bach.deg      0.00       0.00    8.33     0.00
## pct.hs.grad:pct.unemp         0.00       0.00    2.73     0.01
```

```
cat("\nR2 = ",summary(step.result.02.bic)$r.squared)
```

```
##
## R2 =  0.8721672
```

```
cat("\nR2adj = ",summary(step.result.02.bic)$adj.r.squared)
```

```
##
## R2adj =  0.8691874
```

```
round(summary(step.result.02.aic)$coef,2)
```

```
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     10.24       0.79   12.98     0.00
## log.land.area                    0.21       0.08    2.50     0.01
## pop.18_34                       -0.03       0.01   -4.73     0.00
## pop.65_plus                     -0.02       0.01   -1.92     0.06
## log.doctors                     -0.17       0.07   -2.53     0.01
## log.hosp.beds                   -0.11       0.10   -1.05     0.29
## log.crimes                       0.03       0.02    1.36     0.17
## pct.hs.grad                      0.00       0.01   -0.06     0.95
## pct.bach.deg                     0.06       0.01    4.60     0.00
## pct.below.pov                   -0.03       0.02   -1.52     0.13
## pct.unemp                       -0.01       0.01   -1.03     0.30
## pop.65_plus:pct.bach.deg         0.00       0.00    6.89     0.00
## log.land.area:pct.below.pov      0.00       0.00   -2.59     0.01
## log.land.area:pct.hs.grad        0.00       0.00   -2.39     0.02
## pct.hs.grad:pct.unemp            0.00       0.00    2.17     0.03
## pct.bach.deg:pct.below.pov       0.00       0.00   -5.81     0.00
## pop.18_34:log.doctors            0.01       0.00    2.74     0.01
## pct.hs.grad:pct.bach.deg         0.00       0.00   -4.10     0.00
```

```
## pct.bach.deg:pct.unemp          0.00       0.00   -1.80     0.07
## log.hosp.beds:pct.below.pov     0.01       0.00    4.52     0.00
## pop.18_34:pct.below.pov         0.00       0.00    3.29     0.00
## log.crimes:pct.below.pov        0.00       0.00   -2.82     0.01
## log.hosp.beds:pct.hs.grad       0.00       0.00    2.69     0.01
## pop.18_34:log.hosp.beds         0.00       0.00   -1.68     0.09
## log.land.area:pop.65_plus       0.00       0.00   -1.89     0.06
## pop.65_plus:log.crimes          0.00       0.00    1.60     0.11
```

```
cat("\nR2 = ",summary(step.result.02.aic)$r.squared)
```

```
##
## R2 =  0.8919859
```

```
cat("\nR2adj = ",summary(step.result.02.aic)$adj.r.squared)
```

```
##
## R2adj =  0.8854633
```

In the BIC-based model, there are two interactions that appear to be statistically significant, but their practical effect is almost zero on per-capita income.

The story is similar in the AIC model, which has many more interaction terms, but only two with an effect as large as 0.01.

Although both interactions models produced big jumps in AIC and BIC (much bigger than 10!), the improvement in R^2 and R^2_{adj} is pretty small, for all the terms that have been added to the models.

For these reasons I might be willing to discuss these interactions with the social scientist, but I am disinclined to include them in a final model.

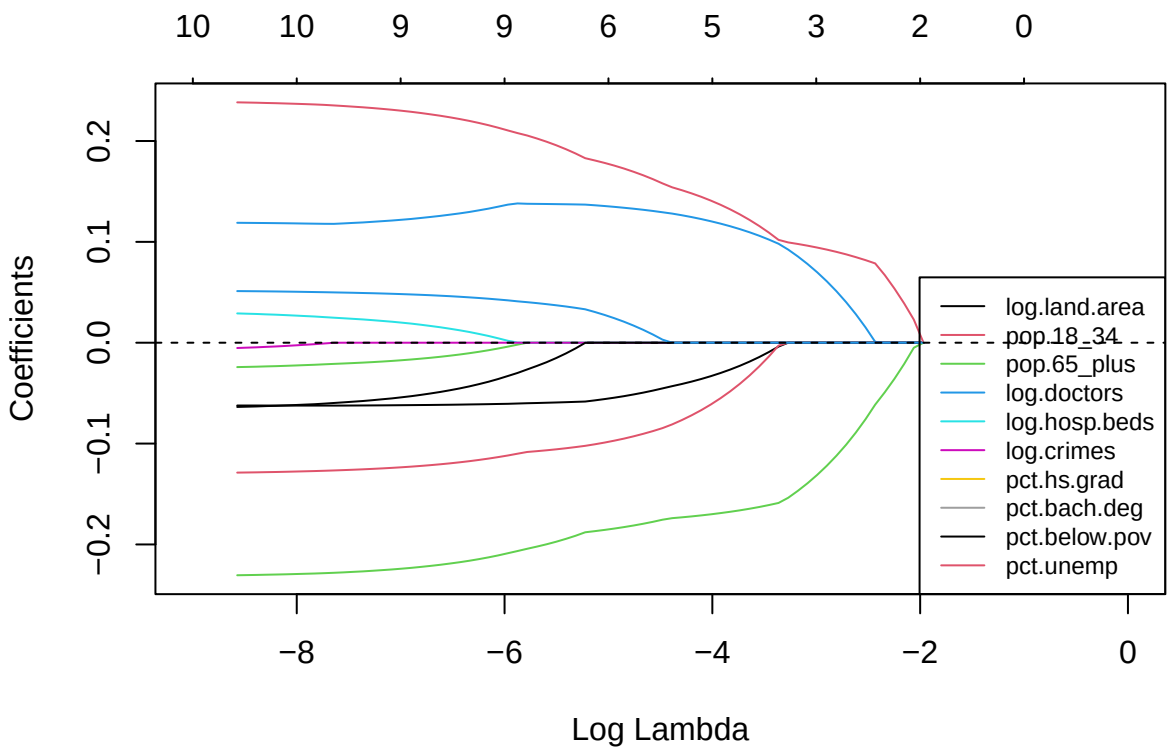
If I stick with the model found by all-subsets and stepAIC with a BIC penalty, then my conclusions about adding interactions with `region` will also be the same, and I will once again be led to `all.subsets.01.final.with.some.region`, which has some interesting and mostly-interpretable structure.

LASSO:

The last variable selection method I'll consider is the lasso.

```
library(glmnet)
library(arm)
```

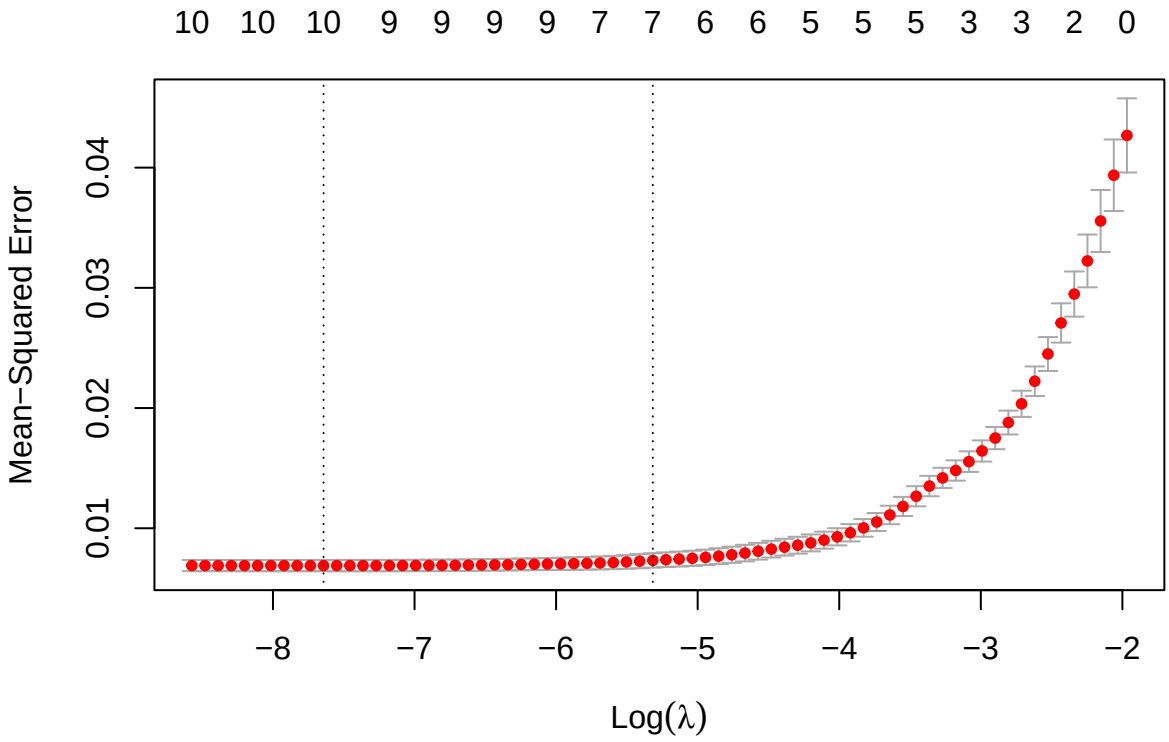
```
loc <- grep("log.per.cap.income",names(cdilogred.cont))

y <- cdilogred.cont[,loc]

X <- apply(as.matrix(cdilogred.cont[,-loc]),2,function(x) rescale(x,"full"))

Xnames <- dimnames(X)[[2]]

lasso.result <- glmnet(X,y)

plot(lasso.result,xvar="lambda",xlim=c(-9,0))

abline(h=0,lty=2)

legend('bottomright',lty=1,col=1:length(Xnames),legend=Xnames,cex=0.75)
```



There is not such an obvious place to cut the shrinkage plot, to help determine what variables should be kept in the model.

Let's see if choosing lambda by cross-valiation works:

```
cv.lasso.result <- cv.glmnet(X,y)

plot(cv.lasso.result)
```




```
c(lambda.1se=cv.lasso.result$lambda.1se,lambda.min=cv.lasso.result$lambda.min)
```

```
##   lambda.1se   lambda.min
## 0.0049081710 0.0004795332
```

```
tmp <- cbind(coef(cv.lasso.result,s=cv.lasso.result$lambda.min),
             coef(cv.lasso.result,s=cv.lasso.result$lambda.1se)
            )
dimnames(tmp)[[2]] <- c("lambda(minMSE)","lambda(minMSE+1se)")

tmp
```

```
## 11 x 2 sparse Matrix of class "dgCMatrix"
##               lambda(minMSE) lambda(minMSE+1se)
## (Intercept)     9.806955e+00        9.806954586
## log.land.area  -6.233501e-02       -0.058650987
## pop.18_34      -1.266292e-01       -0.103473858
## pop.65_plus    -2.099172e-02        .
## log.doctors     1.178374e-01        0.137118673
## log.hosp.beds   2.479267e-02        .
## log.crimes     -4.850651e-05        .
## pct.hs.grad    -5.975737e-02       -0.005665686
## pct.bach.deg    2.352630e-01        0.187649870
## pct.below.pov  -2.281218e-01       -0.191409203
## pct.unemp       4.988665e-02        0.034590648
```

It's interesting to note that the model which minimizes 10-fold cross-validation error contains all 10 predictors (or sometimes 9—log.crimes can get left out—depending on what random folds we get), while the model that

is 1 SE above that is again the same model that we first saw with all-subsets regression.

Although the magnitudes of the coefficients are a little different from what we saw before (because they arise from mimimizing RSS penalized by the L1 penalty, rather than minimizing straight RSS), the signs are the same.

Once again, if we were to settle on this model, and try to bring in `region`, chances are we would end up with the same model as `all.subsets.01.final.with.some.region`.

```
round(summary(all.subsets.01.final.with.some.region)$coef,2)
```

```
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 10.13       0.25   40.25     0.00
## log.land.area               -0.03       0.01   -6.43     0.00
## pop.18_34                   -0.02       0.00  -14.17     0.00
## log.doctors                  0.06       0.00   13.72     0.00
## pct.hs.grad                  0.00       0.00   -0.79     0.43
## pct.bach.deg                 0.01       0.00    6.74     0.00
## pct.below.pov               -0.02       0.00   -7.05     0.00
## pct.unemp                    0.02       0.00    3.72     0.00
## regionNE                     0.22       0.30    0.73     0.47
## regionS                     -0.06       0.28   -0.23     0.82
## regionW                      1.63       0.36    4.56     0.00
## pct.hs.grad:regionNE         0.00       0.00   -0.94     0.35
## pct.hs.grad:regionS          0.00       0.00    0.57     0.57
## pct.hs.grad:regionW         -0.02       0.00   -4.50     0.00
## pct.bach.deg:regionNE        0.01       0.00    2.26     0.02
## pct.bach.deg:regionS         0.00       0.00   -0.56     0.58
## pct.bach.deg:regionW         0.01       0.00    2.41     0.02
## pct.below.pov:regionNE       0.00       0.00   -0.52     0.60
## pct.below.pov:regionS        0.01       0.00    1.94     0.05
## pct.below.pov:regionW       -0.02       0.01   -2.95     0.00
## pct.unemp:regionNE          -0.01       0.01   -1.10     0.27
## pct.unemp:regionS           -0.02       0.01   -3.87     0.00
## pct.unemp:regionW           -0.02       0.01   -2.99     0.00
```

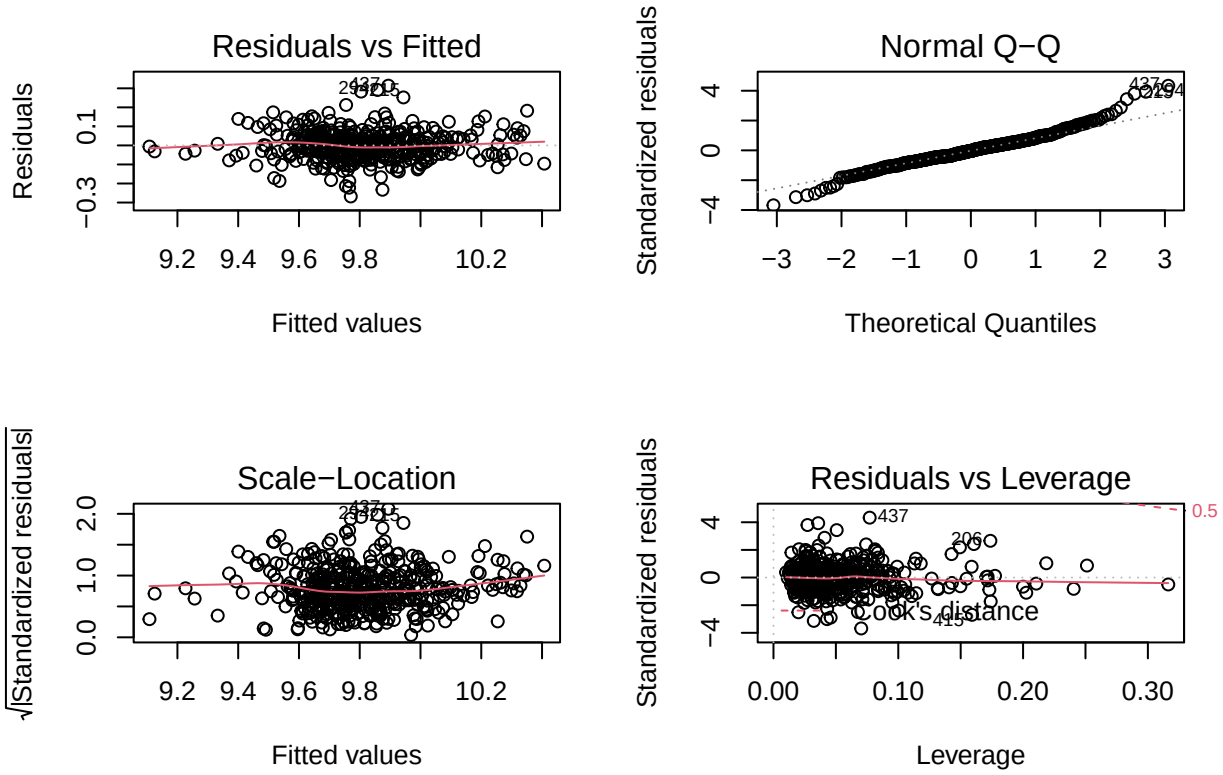
```
cat("\nR2 = ",summary(all.subsets.01.final.with.some.region)$r.squared)
```

```
##
## R2 =  0.8734897
```

```
cat("\nR2adj = ",summary(all.subsets.01.final.with.some.region)$adj.r.squared)
```

```
##
## R2adj =  0.8668153
```

```
par(mfrow=c(2,2))
plot(all.subsets.01.final.with.some.region)
```



Here are a few of the models plus's and minus's, and some tradeoffs I considered:

- **Plus's:**
  - The model is fairly parsimonious, and most of the estimated coefficients have the expected sign.
  - The model is confirmed by stepwise and lasso procedures.
  - Those procedures also found more complex models with somewhat better fit, but improvements in fit seemed small compared to the added complexity of the model.
  - All of the variables are either in their original scale, or the have been replaced with their logarithm. This facilitates explaining the models to anyone who is interested in and knowledgeable about the social science & economics but less knowledgeable about technical matters.
- **Minus's:**
  - The coefficient on `pct.unemp` seems to go the wrong way, and the coefficient on 'pct.hs.grad' is quite small, statistically and practically (it remains in the model because there is a noticable interaction that it participates in).
  - The residual diagnostic plots are just OK.
  - The fact that stepwise regression found some well-fitting models with interactions between continuous variables suggests exploring those more complex models in the future.
  - I did not have time to explore the state variable at all. Some of the relationship between these demographic variables and per capita income might be explainable in terms of varying economic policy from one state to the next. (If one includes `state` in the model, one could take out `region` because the two are perfectly collinear (states are entirely nested within regions).

Finally, it would be very useful to have additional data to compare some of the models we found. We are using reasonable methods for variable selection, but since it is all within-sample (our entire data set is our training sample), there is ample room for overfitting noise in the data. Some of our inferences about which variables to leave in or take out may be based on overly optimistic standard error estimates, for example. If

num variables	minority	beautyf2upper	beautyfupperdiv	btystdave	btystdf2u	btystdfu	female	nonenglish	onecredit	percentevaluating	blkandwhite	BIC
1									*			-14.0820700135937
2				*					*			-30.0187608382119
3			*				*		*			-42.0422726206511
4			*				*		*	*		-52.6352707419912
5	*		*				*		*	*		-61.8604866501093
6			*				*	*	*	*	*	-68.7273809819797
7	*		*				*	*	*	*	*	-68.168278525645
8		*	*		*		*	*	*	*	*	-67.592998887792
9	*	*	*		*		*	*	*	*	*	-67.4303455606005
10	*	*	*		*	*	*	*	*	*	*	-64.9400133420853

we were able to cross-validate on some new or hold-out data, we might be able to better distinguish what is the best model, at least in terms of prediction error.

2 Problem 2

This will be much quicker and will mostly consist of code and comments written by a TA from an earlier version of this course.

```
beauty <- read_csv("../ProfEvaltnsBeautyPublic.csv") %>%
  dplyr::select(-profevaluation, -profnumber, -multipleclass,
                -starts_with("class"))
```

2.1 Part a

```
library(leaps)
all_subsets <- regsubsets(courseevaluation ~.,
                          data = beauty,
                          nvmax = 10, intercept = T,
                          method = "exhaustive")

summ <- summary(all_subsets)

best_model_index <- which.min(summ$bic)

dropped <- summ$outmat %>% apply(2, function(col) {mean(col == " ") == 1})
all_subset_info <- cbind(summ$outmat[,!dropped], BIC = summ$bic)
rownames(all_subset_info) <- 1:nrow(all_subset_info)
all_subset_info %>% data.frame %>%
  rownames_to_column(var = "num variables") %>%
  knitr::kable() %>%
  kableExtra::kable_styling(latex_options = "scale_down") %>%
  row_spec(best_model_index, background = "gray")
```

```
var_names <- colnames(all_subset_info)[all_subset_info[best_model_index,] == "*"]

all_subsets_model <- lm(as.formula(paste("courseevaluation ~",
                                         paste0(var_names, collapse = " + "))),
                        data = beauty)

summary(all_subsets_model)
```

```
##
## Call:
## lm(formula = as.formula(paste("courseevaluation ~", paste0(var_names,
##     collapse = " + "))), data = beauty)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.97607 -0.30533  0.04791  0.37959  1.04118
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        3.355273   0.113922  29.452  < 2e-16 ***
## beautyfupperdiv    0.058282   0.012559   4.641 4.54e-06 ***
## female            -0.254260   0.048290  -5.265 2.16e-07 ***
## nonenglish        -0.388186   0.097125  -3.997 7.49e-05 ***
## onecredit          0.500846   0.099146   5.052 6.35e-07 ***
## percentevaluating  0.005502   0.001415   3.889 0.000115 ***
## blkandwhite        0.251170   0.063629   3.947 9.15e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.495 on 456 degrees of freedom
## Multiple R-squared:  0.2143, Adjusted R-squared:  0.204
## F-statistic: 20.73 on 6 and 456 DF,  p-value: < 2.2e-16
```

2.2 Part b

We'll show forward regression.

```
forward_step <- step(lm(courseevaluation ~1, data = beauty),
  scope  = as.formula(paste("courseevaluation ~",
                      paste0(names(beauty)[names(beauty) != "courseevaluation"],
                             collapse = " + "))),
                      direction = "forward", trace = 0)

summary(forward_step)
```

```
##
## Call:
## lm(formula = courseevaluation ~ onecredit + btystdave + percentevaluating +
##     female + minority + blkandwhite + nonenglish + beautyfupperdiv +
##     btystdfu + btystdf2u + tenuretrack + formal + age, data = beauty)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.86588 -0.29589  0.04495  0.33963  1.06133
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -0.836838   1.639295  -0.510 0.609962
## onecredit          0.503749   0.107838   4.671 3.96e-06 ***
## btystdave         -0.167839   0.075372  -2.227 0.026456 *
## percentevaluating  0.004850   0.001451   3.342 0.000902 ***
## female            -0.302259   0.052308  -5.778 1.41e-08 ***
## minority          -0.155301   0.074788  -2.077 0.038412 *
## blkandwhite        0.285158   0.069381   4.110 4.70e-05 ***
## nonenglish        -0.381223   0.104535  -3.647 0.000297 ***
## beautyfupperdiv    0.939194   0.323894   2.900 0.003918 **
## btystdfu          -1.770340   0.656800  -2.695 0.007294 **
## btystdf2u          0.092159   0.042790   2.154 0.031790 *
```

```
## tenuretrack       -0.132773   0.062020  -2.141 0.032828 *
## formal             0.143150   0.069643   2.055 0.040410 *
## age               -0.004442   0.002837  -1.566 0.118145
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4867 on 449 degrees of freedom
## Multiple R-squared:  0.2523, Adjusted R-squared:  0.2306
## F-statistic: 11.65 on 13 and 449 DF,  p-value: < 2.2e-16
```

2.3 Part c

```
library(glmnet)
cv_lasso_selection <- cv.glmnet(x = as.matrix(beauty[,names(beauty) != "courseevaluation"]),
          y = beauty$courseevaluation, alpha = 1)
```

```
lasso <- glmnet(x = as.matrix(beauty[,names(beauty) != "courseevaluation"]),
                y = beauty$courseevaluation,
                alpha = 1, lambda = cv_lasso_selection$lambda.1se)


coef_lasso <- as.matrix(coef(lasso))
coef_lasso_df <- data.frame(columns = names(coef_lasso[coef_lasso[,1] != 0,]),
         beta = coef_lasso[coef_lasso[,1] != 0,])
rownames(coef_lasso_df) <- NULL

coef_lasso_df
```

```
##              columns          beta
## 1        (Intercept)  3.533028e+00
## 2           minority -4.888278e-02
## 3      beautyf2upper  5.133109e-03
## 4    beautyfupperdiv  2.973839e-02
## 5          btystdf2u  3.833213e-06
## 6             female -1.166587e-01
## 7           fulldept  7.914660e-02
## 8         nonenglish -1.675603e-01
## 9          onecredit  3.553839e-01
## 10 percentevaluating  3.304740e-03
## 11       tenuretrack -2.928520e-03
## 12       blkandwhite  1.190623e-01
```

2.4 Part d

```
#basically coding up a document term matrix function for some reason

dtm_coef <- function(names_list){
  all_words <- unique(unlist(names_list))
  matrix_info <- matrix(nrow = length(names_list), ncol = length(all_words))

  r_idx <- 1
  for (col_names in names_list){
    matrix_info[r_idx,] <- all_words %in% col_names
    r_idx <- r_idx + 1
```

```
  }

  colnames(matrix_info) <- all_words
  rownames(matrix_info) <- names(names_list)

  return(matrix_info)
}

variables_in_model <- list(
  "best subset" = names(coef(all_subsets_model)),
  "lasso" = coef_lasso_df$columns,
  "forward stepwise" = names(forward_step$coefficients),
  "homework 05, 2c" = c(names(beauty[,names(beauty) != "courseevaluation"]),
                        "profevaluation", "(Intercept)")
)
```

```
t(dtm_coef(variables_in_model)) %>%
  data.frame(check.names = F) %>%
  rownames_to_column(var = "features") %>%
  mutate(`best subset` = ifelse(`best subset`,
                                cell_spec(`best subset`, "latex", color = "black"),
                                cell_spec(`best subset`, "latex", color = "red")),
         `lasso` = ifelse(`lasso`,
                                cell_spec(`lasso`, "latex", color = "black"),
                                cell_spec(`lasso`, "latex", color = "red")),
         `forward stepwise` = ifelse(`forward stepwise`,
                                cell_spec(`forward stepwise`, "latex", color = "black"),
                                cell_spec(`forward stepwise`, "latex", color = "red")),
         `homework 05, 2c` = ifelse(`homework 05, 2c`,
                                cell_spec(`homework 05, 2c`, "latex", color = "black"),
                                cell_spec(`homework 05, 2c`, "latex", color = "red"))) %>%
  knitr::kable(format = "latex", caption = "Features include in each model.",
               escape = F) %>%
  kableExtra::kable_styling(latex_options = c("scale_down"),)
```

The number of features (excluding the intercept) in our models increase from 6 in the best subset model, lasso with 10, forward stepwise regression selecting 14 and our full model with 30 features in homework 05. If we take out the lasso model these models are nested inside each other. **It's hard to evaluate prediction models as we left nothing to evaluate them on.** We observe that the lasso had actually higher training mse than the best subset model - but that's not to crazy because the penalty means that we're get more training error allowed.

Following the advice of Gelman and Hill from the end of lecture 11, it might also be useful to compare tables of estimated coefficients and their SE's, to see whether the direction and size of influence of each predictor on teacher ratings is (a) plausible and (b) consistent (or inconsistent) across models. As usual, however, since we are selecting variables and fitting the model on the same data, we can be misled by overly optimistic standard errors, overfitting to noise in the data, etc. So this approach ultimately requires some care.

Table 7: Features include in each model.

features	best subset	lasso	forward stepwise	homework 05, 2c
(Intercept)	TRUE	TRUE	TRUE	TRUE
beautyfupperdiv	TRUE	TRUE	TRUE	TRUE
female	TRUE	TRUE	TRUE	TRUE
nonenglish	TRUE	TRUE	TRUE	TRUE
onecredit	TRUE	TRUE	TRUE	TRUE
percentevaluating	TRUE	TRUE	TRUE	TRUE
blkandwhite	TRUE	TRUE	TRUE	TRUE
minority	FALSE	TRUE	TRUE	TRUE
beautyf2upper	FALSE	TRUE	FALSE	TRUE
btystdf2u	FALSE	TRUE	TRUE	TRUE
fulldept	FALSE	TRUE	FALSE	TRUE
tenuretrack	FALSE	TRUE	TRUE	TRUE
btystdave	FALSE	FALSE	TRUE	TRUE
btystdfu	FALSE	FALSE	TRUE	TRUE
formal	FALSE	FALSE	TRUE	TRUE
age	FALSE	FALSE	TRUE	TRUE
tenured	FALSE	FALSE	FALSE	TRUE
beautyflowerdiv	FALSE	FALSE	FALSE	TRUE
beautym2upper	FALSE	FALSE	FALSE	TRUE
beautymlowerdiv	FALSE	FALSE	FALSE	TRUE
beautymupperdiv	FALSE	FALSE	FALSE	TRUE
btystdfl	FALSE	FALSE	FALSE	TRUE
btystdm2u	FALSE	FALSE	FALSE	TRUE
btystdml	FALSE	FALSE	FALSE	TRUE
btystdmu	FALSE	FALSE	FALSE	TRUE
didevaluation	FALSE	FALSE	FALSE	TRUE
lower	FALSE	FALSE	FALSE	TRUE
students	FALSE	FALSE	FALSE	TRUE
btystdvariance	FALSE	FALSE	FALSE	TRUE
btystdavepos	FALSE	FALSE	FALSE	TRUE
btystdaveneg	FALSE	FALSE	FALSE	TRUE
profevaluation	FALSE	FALSE	FALSE	TRUE