R Notebook

ECO518 PS4

0. Set up

```
# Load packages
if(!require(pacman)) install.packages("pacman")
```

```
## Loading required package: pacman
```

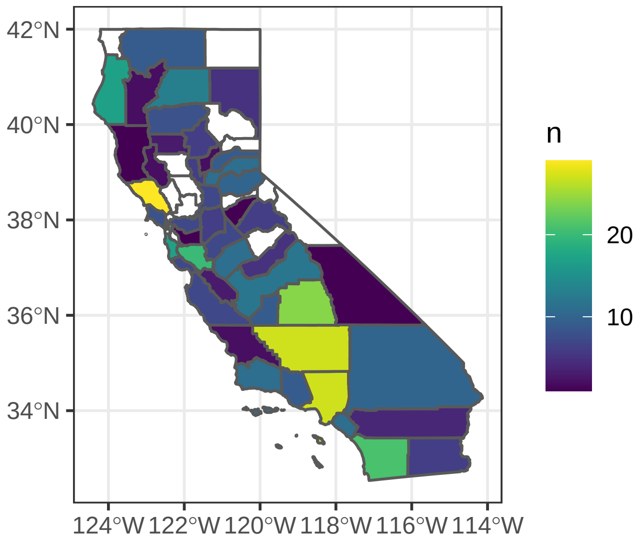
```
pacman::p_load(ggplot2, dplyr, sf, tigris,
               viridis, patchwork, sandwich, nlme, jtools, estimatr,
               stargazer, car)
theme_set(theme_bw())
# Set paths
dir <- paste0("/Users/tombearpark/Documents/princeton/1st_year/",
              "term2/ECO518_Metrics2/sims/exercises/4_grouped_data/")
out <- paste0(dir, "out/")
# Load in the data
load(paste0(dir, "caschool.RData"))
df <- tibble(caschool)
# load a shapefile for maps
cal <- counties(state = "California", cb = TRUE)
```

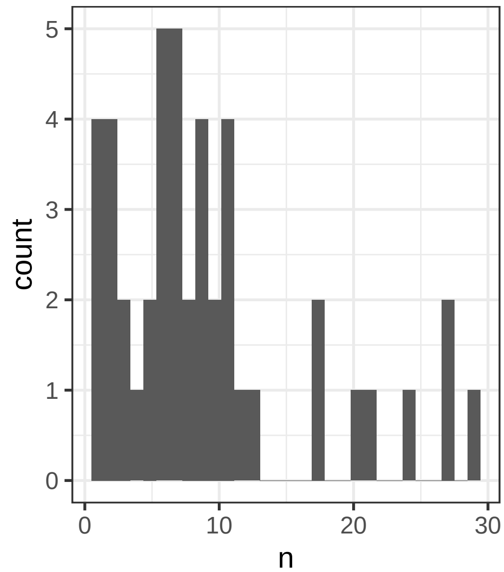
```
##   |                                                                              |
```

```
plot_df <-
  left_join(cal, df %>% group_by(county) %>% tally(),
      by = c("NAME" = "county")) %>%
  mutate(obs = ifelse(is.na(n), 0, n))

# Make sure the merge worked properly
stopifnot(
  dim(anti_join(df %>% group_by(county) %>% tally(),
              cal, by = c("county" = "NAME")))[1] == 0)

wrap_elements(
  (ggplot(plot_df) + geom_sf(aes(fill = n)) +
    scale_fill_viridis(na.value = "white")) +
  (ggplot(plot_df) + geom_histogram(aes(x = n))))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 13 rows containing non-finite values (stat_bin).
```





2. Exercise

Problem 1

Estimate a linear regression of the average test score (testscr) on student-teacher ratio, computers per student, and expenditures per student. Determine whether the three variables have explanatory power by an F-Test of the hypothesis that all three have zero coefficients and via the Bayesian information criterion (BIC). The latter can be computed from an F-statistic: The BIC rejects the restriction when the F-statistic exceeds the log of the sample size.

```
N <- length(df$avginc)
reg1 <- "testscr ~ str + comp_stu + expn_stu"
lm1 <- lm(data = df, formula(reg1))
```

The F stat is 14.96.

```
summary(lm1)
```

```
##
## Call:
## lm(formula = formula(reg1), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.660 -14.093  -0.733  13.079  45.975
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 663.317921  19.348195  34.283  < 2e-16 ***
## str          -1.303902   0.606981  -2.148   0.0323 *
## comp_stu     63.638660  14.477669   4.396  1.4e-05 ***
## expn_stu      0.001468   0.001799   0.816   0.4151
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.17 on 416 degrees of freedom
## Multiple R-squared:  0.09738,    Adjusted R-squared:  0.09087
## F-statistic: 14.96 on 3 and 416 DF,  p-value: 2.892e-09
```

```r
compare_models <- function(lm_restricted, lm_unrestricted, N){

    RSSR <- sum(lm_restricted$residuals^2)
    RSSU <- sum(lm_unrestricted$residuals^2)
    k    <- length(lm_unrestricted$coefficients) - length(lm_restricted$coefficients)

    Fstat <- ((RSSR - RSSU) / k) / (RSSU / (N-k-1))
    pVal  <- pf(Fstat, k, N-k-1, lower.tail = FALSE)

    BIC_R <- N * log(RSSR / N) + length(lm_restricted$coefficients)  * log(N)
    BIC_U <- N * log(RSSU / N) + length(lm_unrestricted$coefficients)  * log(N)

    lowerBIC <- ifelse(BIC_R < BIC_U, "restricted", "unrestricted")

    return(
      tibble(Fstat = Fstat, pVal = pVal,
             BIC_R = BIC_R, BIC_U = BIC_U, logN = log(N),
             lowest_BIC_model = lowerBIC))
}
lm0 <- lm(data = df, testscr ~ 1)
comparison_1 <- compare_models(lm0, lm1, N)
comparison_1
```

```
## # A tibble: 1 x 6
##   Fstat          pVal BIC_R BIC_U  logN lowest_BIC_model
##   <dbl>         <dbl> <dbl> <dbl> <dbl> <chr>
## 1  15.0 0.00000000289 2481. 2456.  6.04 unrestricted
```

- BIC is smallest for the true model. BIC is smallest for the more complex model
  - We can also see that the F-stat is larger than the log of the sample size, so we reject the restriction
- F stat strongly rejects the Null that the coefficients aren't jointly significant

```r
Anova(lm1)
```

```
## Anova Table (Type II tests)
##
## Response: testscr
##           Sum Sq  Df F value    Pr(>F)
## str         1523   1  4.6147   0.03228 *
## comp_stu    6377   1 19.3217 1.404e-05 ***
```

```
## expn_stu     220   1  0.6655   0.41510
## Residuals 137298 416
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# run a clustered version
lm1_c <- lm_robust(formula(reg1), data = df, cluster = county)
summary(lm1_c)
```

```
##
## Call:
## lm_robust(formula = formula(reg1), data = df, clusters = county)
##
## Standard error type:  CR2
##
## Coefficients:
##                Estimate Std. Error t value  Pr(>|t|)   CI Lower  CI Upper    DF
## (Intercept) 663.317921  21.950633 30.2186 1.273e-21 618.174063 708.46178 25.72
## str          -1.303902   0.656269 -1.9868 5.796e-02  -2.655260   0.04746 25.09
## comp_stu     63.638660  18.873940  3.3718 2.384e-03  24.809323 102.46800 25.55
## expn_stu      0.001468   0.002717  0.5402 5.943e-01  -0.004154   0.00709 22.91
##
## Multiple R-squared:  0.09738 ,   Adjusted R-squared:  0.09087
## F-statistic: 6.537 on 3 and 44 DF,  p-value: 0.0009409
```

Problem 2

Do the same thing with a regression that adds the demographic variables: Average income, subsidized meals, calWorks per cent, and English learners percent. Again check whether the three "policy variables have explanatory power using an F test and BIC. Here you may need to extract the covariance matrix of coefficients from the lm() output to construct the F or chi-squared statistic.

```
reg2 <- paste0(reg1, " + avginc + meal_pct + calw_pct + el_pct")
lm2 <- lm(data = df, formula(reg2))
```

```
compare_models(lm_restricted = lm1, lm_unrestricted = lm2, N = N)
```

```
## # A tibble: 1 x 6
##   Fstat      pVal BIC_R BIC_U  logN lowest_BIC_model
##   <dbl>     <dbl> <dbl> <dbl> <dbl> <chr>
## 1  387. 1.36e-138 2456. 1827.  6.04 unrestricted
```

```
summary(lm2)
```

```
##
## Call:
## lm(formula = formula(reg2), data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -31.0536  -5.3377   0.1755   5.0343  26.4272
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.596e+02  9.023e+00  73.098  < 2e-16 ***
## str         -1.899e-01  2.835e-01  -0.670   0.5034
## comp_stu     1.189e+01  6.898e+00   1.724   0.0855 .
## expn_stu     1.526e-03  8.917e-04   1.712   0.0877 .
## avginc       6.217e-01  8.772e-02   7.087 5.98e-12 ***
## meal_pct    -3.756e-01  3.589e-02 -10.465  < 2e-16 ***
## calw_pct    -7.782e-02  5.722e-02  -1.360   0.1745
## el_pct      -1.981e-01  3.323e-02  -5.962 5.37e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.391 on 412 degrees of freedom
## Multiple R-squared:  0.8093, Adjusted R-squared:  0.806
## F-statistic: 249.7 on 7 and 412 DF,  p-value: < 2.2e-16
```

```
Anova(lm2)
```

```
## Anova Table (Type II tests)
##
## Response: testscr
##            Sum Sq  Df  F value    Pr(>F)
## str          31.6   1   0.4486   0.50337
## comp_stu    209.2   1   2.9710   0.08552 .
## expn_stu    206.3   1   2.9302   0.08769 .
## avginc     3536.7   1  50.2267 5.979e-12 ***
## meal_pct   7711.7   1 109.5178 < 2.2e-16 ***
## calw_pct    130.3   1   1.8498   0.17455
## el_pct     2502.8   1  35.5437 5.365e-09 ***
## Residuals 29011.1 412
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Once again, our tests prefer the more complex model

```
lm2_c <- lm_robust(formula(reg2), data = df, cluster = county)
summary(lm2_c)
```

```
##
## Call:
## lm_robust(formula = formula(reg2), data = df, clusters = county)
##
## Standard error type:  CR2
##
## Coefficients:
##               Estimate Std. Error t value  Pr(>|t|)   CI Lower   CI Upper    DF
## (Intercept) 659.587061  12.566652 52.4871 1.446e-27 633.734800 685.439322 25.57
## str          -0.189910   0.325244 -0.5839 5.645e-01  -0.859845   0.480025 24.94
## comp_stu     11.890284   7.657159  1.5528 1.328e-01  -3.864839  27.645406 25.48
## expn_stu      0.001526   0.001306  1.1685 2.550e-01  -0.001181   0.004234 22.20
```

```
## avginc        0.621673   0.100062  6.2129 3.462e-05   0.405035   0.838311 12.73
## meal_pct     -0.375618   0.044597 -8.4226 4.652e-08  -0.468557  -0.282679 20.30
## calw_pct     -0.077818   0.063414 -1.2271 2.459e-01  -0.217716   0.062080 10.79
## el_pct       -0.198137   0.038746 -5.1137 6.922e-05  -0.279448  -0.116826 18.29
##
## Multiple R-squared:  0.8093 ,    Adjusted R-squared:  0.806
## F-statistic: 407.6 on 7 and 44 DF,  p-value: < 2.2e-16
```

Problem 3

Repeat the previous estimations and tests in models that add county fixed effects. In R using lm(), this is accomplished by just adding "county" to the list of right-hand side variables. (county is a "factor" in the R dataframe, so R automatically converts it into the appropriate array of dummy variables when including it in a regression.)

```r
reg3 <- paste0(reg2, "+ county")
lm3 <- lm(data = df, formula(reg3))
lm3_c <- lm_robust(data = df, formula(reg3), cluster = county)
```

```r
compare_models(lm2, lm3, N)
```

```
## # A tibble: 1 x 6
##   Fstat      pVal BIC_R BIC_U  logN lowest_BIC_model
##   <dbl>     <dbl> <dbl> <dbl> <dbl> <chr>
## 1  2.27 0.0000199 1827. 1994.  6.04 restricted
```

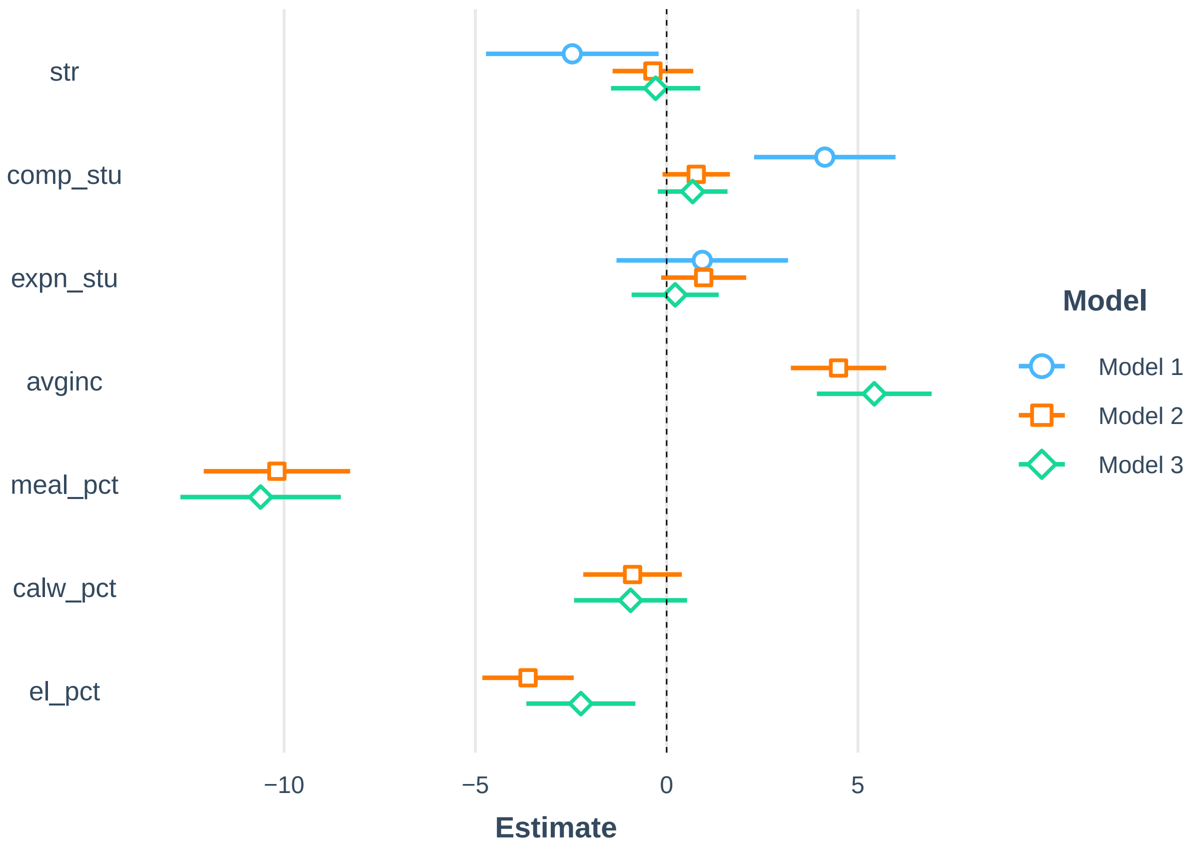
- This time, our model doesn't want us to choose the extra complexity in the BIC.
- However, our F-stat approach prefers the more complex model

```r
Anova(lm3)
```

```
## Anova Table (Type II tests)
##
## Response: testscr
##            Sum Sq  Df F value    Pr(>F)
## str          14.7   1  0.2363  0.627173
## comp_stu    133.7   1  2.1483  0.143580
## expn_stu      9.2   1  0.1477  0.700940
## avginc     3149.0   1 50.6039 5.937e-12 ***
## meal_pct   6168.1   1 99.1184 < 2.2e-16 ***
## calw_pct     98.1   1  1.5761  0.210119
## el_pct      596.1   1  9.5798  0.002118 **
## county     6110.7  44  2.2317 3.170e-05 ***
## Residuals 22900.4 368
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
vars <- c("str", "comp_stu", "expn_stu", "avginc", "meal_pct", "calw_pct", "el_pct" )
plot_summs(lm1, lm2, lm3, coefs = vars, scale = TRUE)
```

```
## Registered S3 methods overwritten by 'broom':
##   method            from
##   tidy.glht         jtools
##   tidy.summary.glht jtools

## Loading required namespace: broom.mixed

## Registered S3 method overwritten by 'broom.mixed':
##   method      from
##   tidy.gamlss broom
```



Problem 4

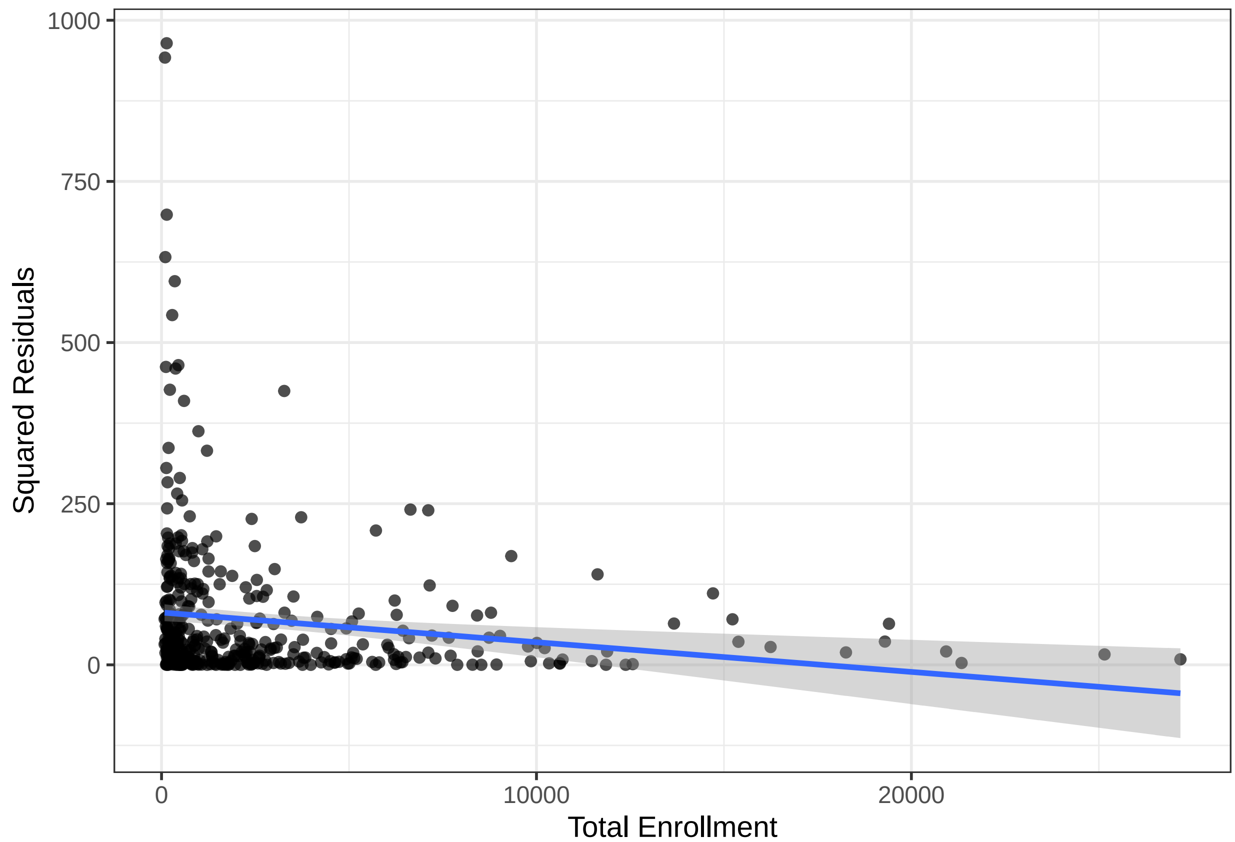
The districts vary greatly in size. Average scores might have more sampling variation in small districts. Plot the squared residuals from the estimated model in 2 against the total enrollment variable. Estimate a linear regression of these squared residuals on 1/enrl_tot. Use the inverse of these predicted values as the weights argument in lm() (or otherwise estimate the corresponding weighted regression estimates) in the question 2 regression.

First, lets plot the squared residuals against the total enrollment variable.

```
df$u2 <- lm2$residuals
df$sqr_u2 <- lm2$residuals ^ 2
ggplot(df, aes(x = enrl_tot, y = sqr_u2)) +
  geom_point(alpha= .7) +
```

```
  xlab("Total Enrollment") + ylab("Squared Residuals") +
  geom_smooth(method = lm)
```

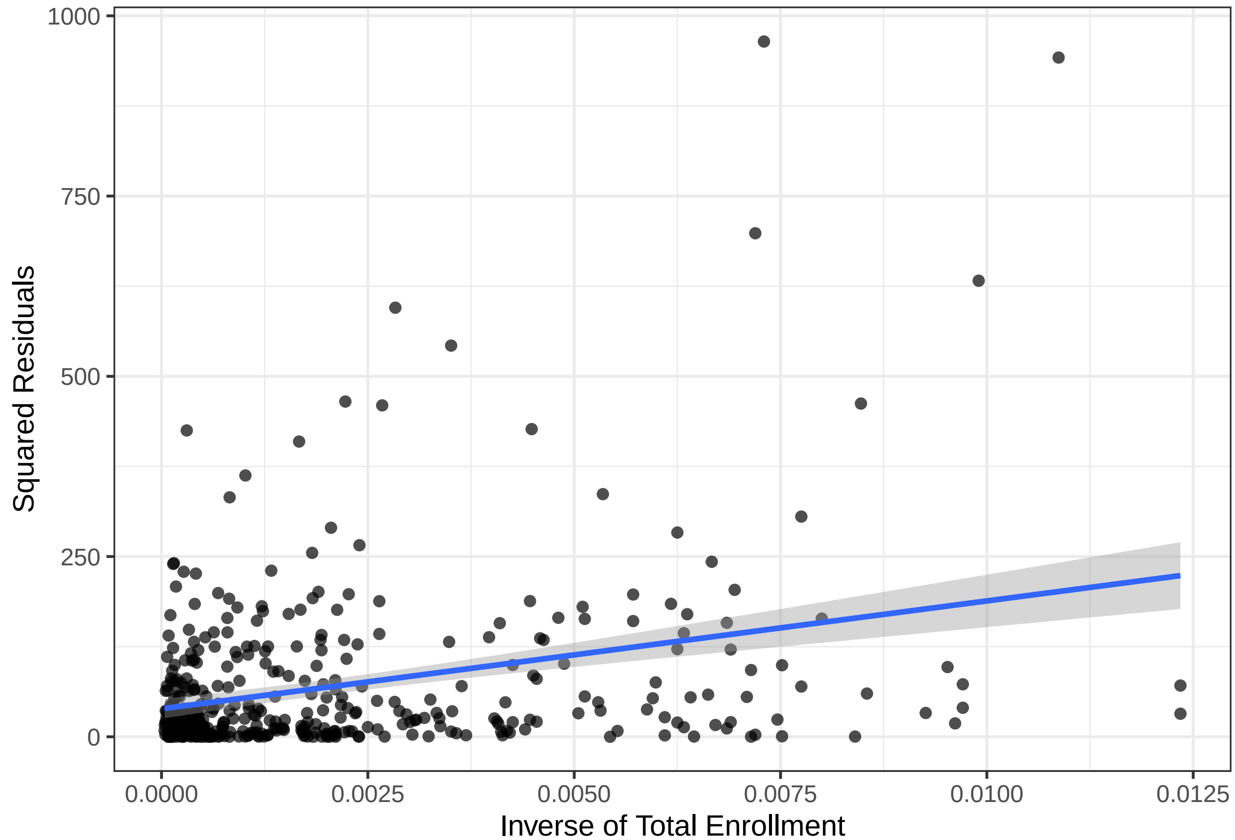
```
## `geom_smooth()` using formula 'y ~ x'
```



Lets also plot the squared residuals against the inverse of total enrollment, since that relationship is what we are going to use for our weighting scheme.

```
ggplot(df, aes(x = 1/enrl_tot, y = sqr_u2)) +
  geom_point(alpha= .7) +
  xlab("Inverse of Total Enrollment") + ylab("Squared Residuals") +
  geom_smooth(method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



We can estimate a linear regression, to calculate the weights…

```
df$inv_enrl_tot <- 1 / df$enrl_tot
lm4_w <- lm(data = df, sqr_u2 ~ inv_enrl_tot)
```

Next we run a regression weighted by the inverse of the residuals

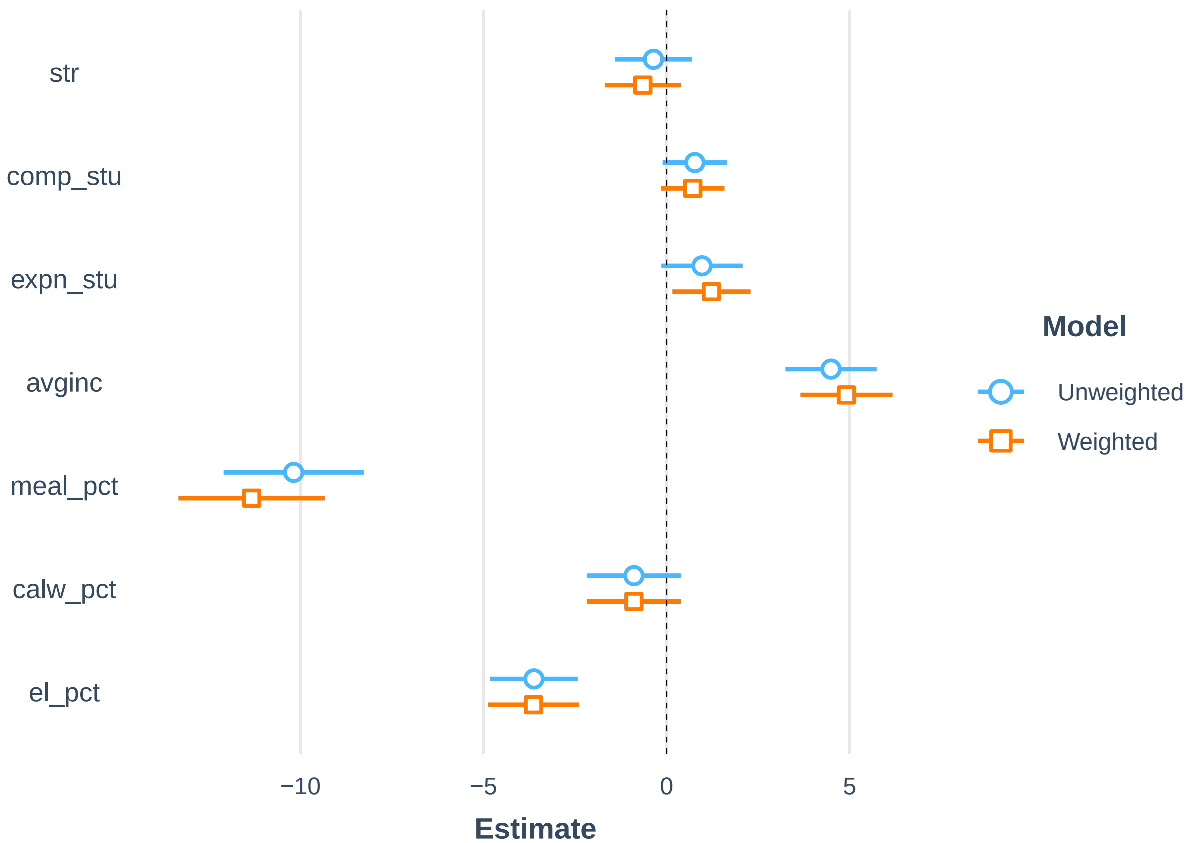
```
df$weights_lm4 <- 1 / lm4_w$fitted.values
lm4 <- lm(data = df, formula(reg2), weights = df$weights_lm4)
summary(lm4)
```

```
##
## Call:
## lm(formula = formula(reg2), data = df, weights = df$weights_lm4)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7312 -0.6772  0.0079  0.6082  3.2024
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.602e+02  9.052e+00  72.934  < 2e-16 ***
## str         -3.455e-01  2.811e-01  -1.229   0.2198
## comp_stu     1.136e+01  6.986e+00   1.626   0.1048
## expn_stu     2.007e-03  8.895e-04   2.256   0.0246 *
## avginc       6.221e-01  8.106e-02   7.675 1.21e-13 ***
```

```
## meal_pct    -3.857e-01  3.463e-02 -11.138  < 2e-16 ***
## calw_pct    -6.986e-02  5.104e-02  -1.369   0.1718
## el_pct      -1.773e-01  3.078e-02  -5.761 1.64e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.009 on 412 degrees of freedom
## Multiple R-squared:  0.8435, Adjusted R-squared:  0.8409
## F-statistic: 317.3 on 7 and 412 DF,  p-value: < 2.2e-16
```

Compare model 2 to this weighted version…

```
plot_summs(lm2, lm4, scale = TRUE, model.names = c("Unweighted", "Weighted"))
```

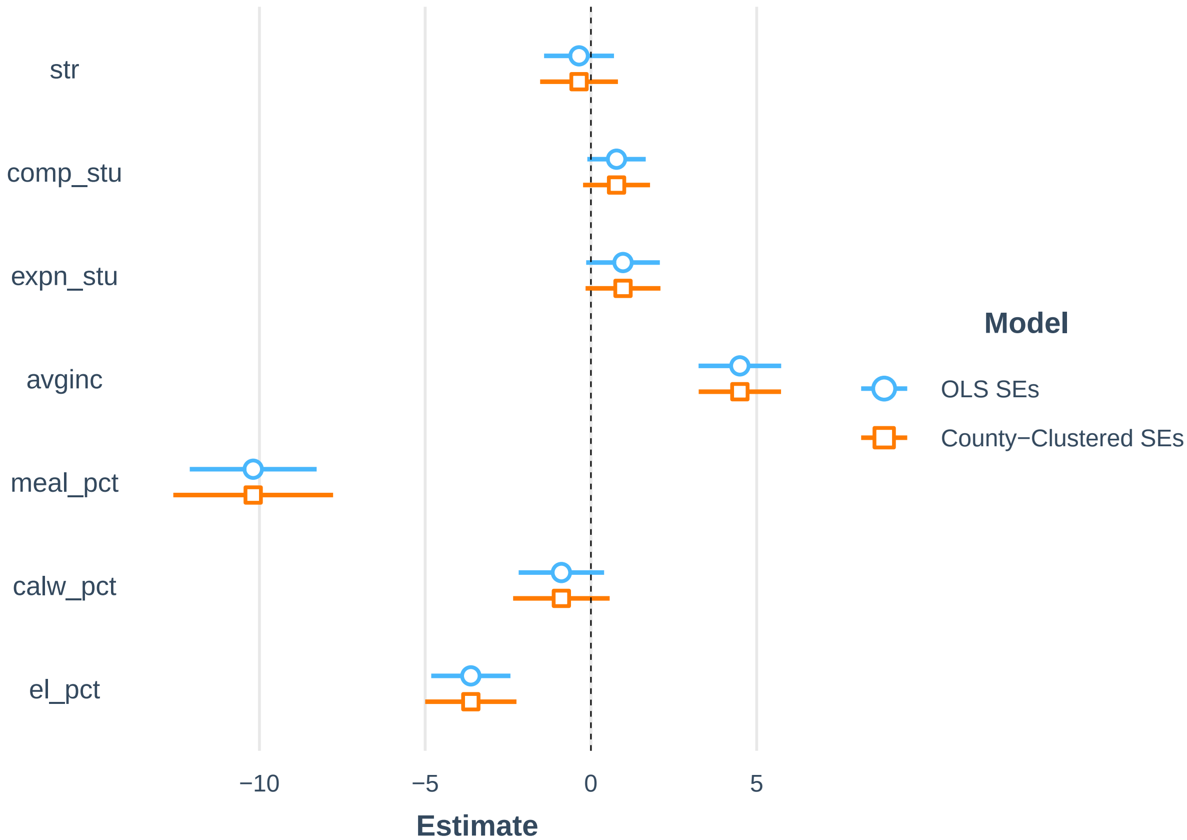


Problem 5

For at least two of the above regression models, calculate standard errors clustered by county. This is done very easily with the vcovCL() function from the sandwich package — so easily that if you're doing it this way you might want to see how much difference it makes in all of the above regressions.

```
plot_summs(lm2, lm2, scale = TRUE,
           robust = list(FALSE, c(cluster = df$county)),
           model.names = c("OLS SEs", "County-Clustered SEs"))
```

```
## Warning in if (type == TRUE) {: the condition has length > 1 and only the first
## element will be used
```



This is done in code throughout. General comment - clustering doesn't make much difference in this setup.

Problem 6

Estimate a random effects model, with county effects. In R, use the lme() function from the nlme package to estimate the 7-variable regression, with random effects by county. You do this by giving lme the argument random = ~1 | county. Also use the argument method="ML", so that the estimation is by maximum likelihood.

```
RE <- lme(data = df, formula(reg2), random = ~1 | county, method = "ML")
summary(RE)$tTable
```

```
##                    Value    Std.Error  DF    t-value       p-value
## (Intercept) 661.036834029 9.0964677878 368 72.6696174 2.445055e-220
## str          -0.196646628 0.2912970412 368 -0.6750725  5.000536e-01
## comp_stu     12.606870943 6.8619471986 368  1.8372148  6.698437e-02
## expn_stu      0.001068223 0.0008875184 368  1.2036069  2.295152e-01
## avginc        0.664923420 0.0925252296 368  7.1864012  3.727999e-12
## meal_pct     -0.367437398 0.0370049535 368 -9.9294111  9.623245e-21
## calw_pct     -0.084161241 0.0589872135 368 -1.4267709  1.544938e-01
## el_pct       -0.186981628 0.0347232272 368 -5.3849150  1.294861e-07
```

```
summary(RE)
```

```
## Linear mixed-effects model fit by maximum likelihood
```

```
##   Data: df
##        AIC      BIC    logLik
##   2985.113 3025.516 -1482.557
##
## Random effects:
##  Formula: ~1 | county
##         (Intercept) Residual
## StdDev:    2.466434 8.004958
##
## Fixed effects:  formula(reg2)
##                Value Std.Error  DF  t-value p-value
## (Intercept) 661.0368  9.096468 368 72.66962  0.0000
## str          -0.1966  0.291297 368 -0.67507  0.5001
## comp_stu     12.6069  6.861947 368  1.83721  0.0670
## expn_stu      0.0011  0.000888 368  1.20361  0.2295
## avginc        0.6649  0.092525 368  7.18640  0.0000
## meal_pct     -0.3674  0.037005 368 -9.92941  0.0000
## calw_pct     -0.0842  0.058987 368 -1.42677  0.1545
## el_pct       -0.1870  0.034723 368 -5.38492  0.0000
##  Correlation:
##          (Intr) str    cmp_st expn_s avginc ml_pct clw_pc
## str      -0.910
## comp_stu -0.127  0.142
## expn_stu -0.750  0.513 -0.137
## avginc   -0.097  0.032 -0.024 -0.294
## meal_pct -0.097 -0.010 -0.059 -0.110  0.501
## calw_pct  0.073 -0.006  0.118 -0.129 -0.034 -0.625
## el_pct    0.062 -0.032  0.157  0.005 -0.165 -0.649  0.293
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.86054173 -0.60780373 -0.01266389  0.59529612  2.87465042
##
## Number of Observations: 420
## Number of Groups: 45
```

Problem 7

Compare the random effects 7-variable model to the fixed effects model. In R, you can do this by re-estimating the fixed-effect model with the gls()function from the nlme package, again being sure to use method="ML" argument. The summary() function applied to either random effects or fixed effects models computed this way deliver both log likelihood and BIC values, so the models can be compared both by afrequentist chi-squared test based on the log likelihood and via the BIC.

```
FE <- gls(data = df, formula(paste0(reg2, "+ county")), method = "ML")
summary(FE)$tTable[1:8, ]
```

```
##                     Value    Std.Error    t-value       p-value
## (Intercept)  6.714925e+02 1.288580e+01 52.1110365 5.977392e-172
## str         -1.524921e-01 3.136940e-01 -0.4861174  6.271733e-01
## comp_stu     1.045821e+01 7.135233e+00  1.4657140  1.435801e-01
## expn_stu     3.512003e-04 9.137477e-04  0.3843515  7.009399e-01
## avginc       7.510105e-01 1.055733e-01  7.1136391  5.937188e-12
```

```
## meal_pct    -3.913137e-01 3.930501e-02 -9.9558217  7.808023e-21
## calw_pct    -8.233494e-02 6.558305e-02 -1.2554301  2.101192e-01
## el_pct      -1.226652e-01 3.963174e-02 -3.0951246  2.117931e-03
```

Problem 8

Finally, for the random effects model, use a regression of its squared residuals on 1/(total enrollment) to generate weights for a weighted random effects estimation; see if this improves likelihood and/or changes important estimates. [Note: I think that in the nlme estimation functions the "weights" arguments are variance scales — the inverse of the weights used in lm(). So you would use a weights= ~w argument to lme() if you used weights=1/w in lm()].

```
df$re_sqr_resid <- RE$residuals[,"county"]^2
lm8_w           <- lm(data = df,  re_sqr_resid ~ inv_enrl_tot)
df$re_weights   <- lm8_w$fitted.values

RE_w <- lme(data = df, formula(reg2), random = ~1 | county,
            method = "ML", weights = ~re_weights)
summary(RE_w)$tTable
```

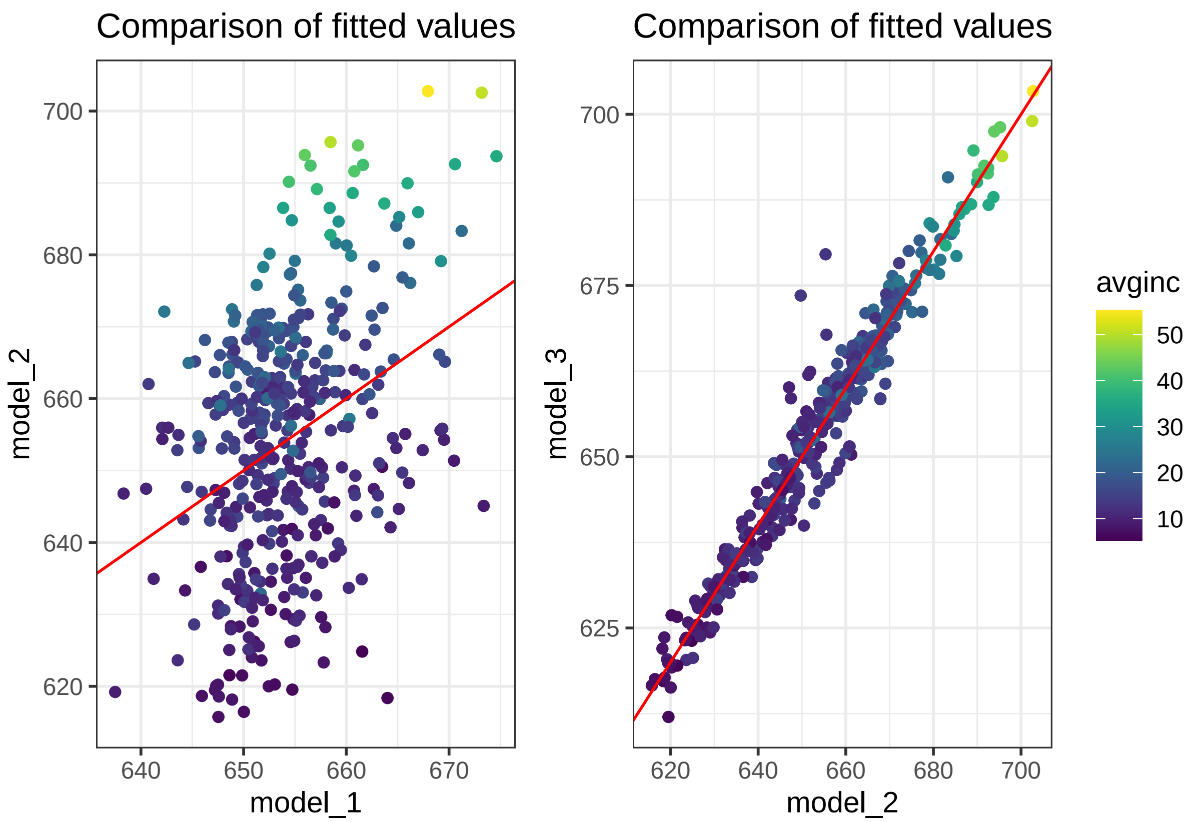
```
##                    Value    Std.Error  DF    t-value       p-value
## (Intercept) 660.917809859 9.1503020743 368  72.229070 1.979801e-219
## str          -0.353137974 0.2901083886 368  -1.217262  2.242845e-01
## comp_stu     12.638742122 6.9264918214 368   1.824696  6.885753e-02
## expn_stu      0.001556244 0.0008818318 368   1.764785  7.842936e-02
## avginc        0.686849009 0.0843735423 368   8.140573  6.152343e-15
## meal_pct     -0.370073550 0.0352340979 368 -10.503279  9.580776e-23
## calw_pct     -0.075814218 0.0525721136 368  -1.442099  1.501248e-01
## el_pct       -0.172368003 0.0319298944 368  -5.398327  1.208316e-07
```

Problem 9

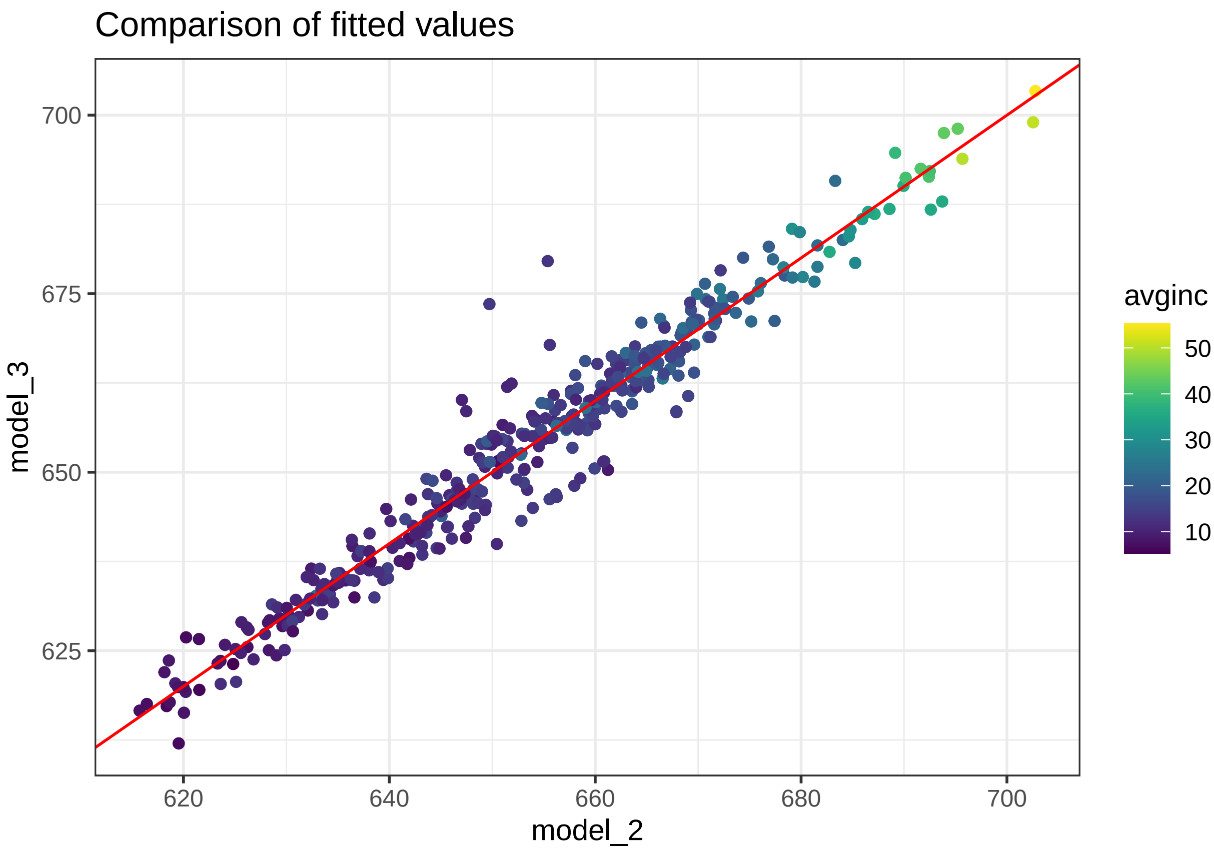
Be ready to discuss: Does the evidence favor an important effect from the "controllable" variables? The sizes and signs of the estimated effects, not just the significance levels of tests, should inform your views on this.

```
comp_df <- data.frame(
  df,
  model_1 = lm1$fitted.values,
  model_2 = lm2$fitted.values,
  model_3 = lm3$fitted.values)

(ggplot(data = comp_df) +
  geom_point(aes(x = model_1, y = model_2, color = avginc)) +
  scale_color_viridis() +
  geom_abline(slope = 1, color = "red") +
  ggtitle("Comparison of fitted values") +
    theme(legend.position = "none")) + (
ggplot(data = comp_df) +
  geom_point(aes(x = model_2, y = model_3, color = avginc)) +
  scale_color_viridis() +
```

```
  geom_abline(slope = 1, color = "red") +
  ggtitle("Comparison of fitted values"))
```




```
ggplot(data = comp_df) +
  geom_point(aes(x = model_2, y = model_3, color = avginc)) +
  scale_color_viridis() +
  geom_abline(slope = 1, color = "red") +
  ggtitle("Comparison of fitted values")
```



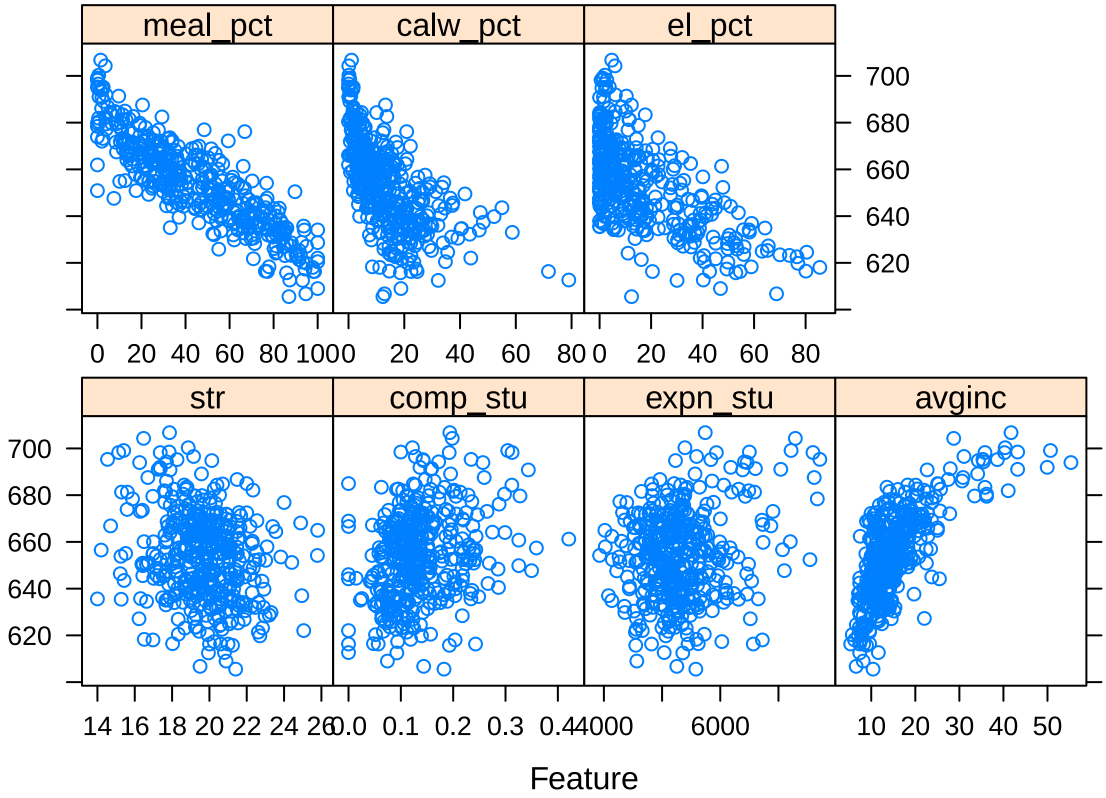

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked _by_ '.GlobalEnv':
##
##     compare_models
```

```
featurePlot(x = df[vars], y = df$testscr)
```

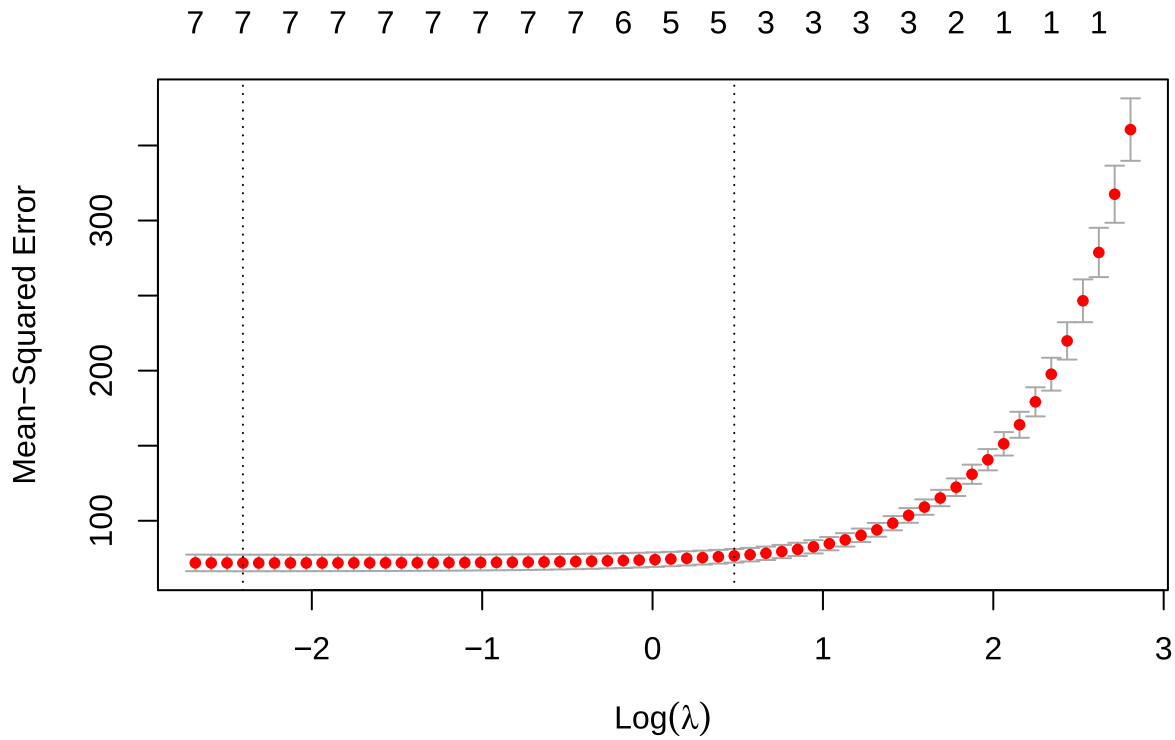



```
# try a lasso regression
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1
```

```
y <- df$testscr %>% as.matrix()
X <- model.matrix(formula(paste0(reg2, "+ log(avginc)")), df)

# Helped function for processing results
get_results = function(fit){
    tmp_coeffs <- coef(fit)
    myResults <- data.frame(
        feature = tmp_coeffs@Dimnames[[1]][ which(tmp_coeffs != 0 ) ], #intercept included
        coef    = tmp_coeffs              [ which(tmp_coeffs != 0 ) ]  #intercept included
    )
    return(myResults)
}

fit <-  cv.glmnet(X, y)
plot(fit)
```




```r
get_results(fit)
```

```
##       feature        coef
## 1 (Intercept) 665.4959536
## 2      avginc   0.5428310
## 3    meal_pct  -0.3909220
## 4      el_pct  -0.1380925
```