PSTAT131_FinalProject

Justin Chan & Brandon Lee

12/2/2021

Partners: Justin Chan (6645626) and Brandon Lee (3263324)

PSTAT 131

Data

Census Data

```
state.name <- c(state.name, "District of Columbia")
state.abb <- c(state.abb, "DC")
## read in census data
census <- read_csv("~/Desktop/classes/fall 2021/pstat 131/acs2017_county_data.csv") %>% select(-CountyI
      filter(State != "PR")
```

Education Data

```
## read in education data
education <- read_csv("~/Desktop/classes/fall 2021/pstat 131/education.csv") %>%
  filter(!is.na(`2003 Rural-urban Continuum Code`)) %>%
  filter(State != "PR") %>% select(-`FIPS Code`,-`2003 Rural-urban Continuum Code`, -`2003 Urban Influe
  rename(County = `Area name`)
```

Preliminary Data Analysis

1)

```
dim(census)
```

```
## [1] 3142   31
```

```
sum(apply(is.na(census[]), 1, any))
```

```
## [1] 0
```

```
length(unique(census$State))
```

```
## [1] 51
```

2)

```
dim(education)
```

```
## [1] 3143   42
```

```r
sum(apply(is.na(education[]), 1, any))
```

```
## [1] 18
```

```r
# Number of distinct County in education
length(unique(education$County))
```

```
## [1] 1877
```

```r
# Number of distinct County in census
length(unique(census$County))
```

```
## [1] 1877
```

3)

```r
education <- na.omit(education)
```

4)

```r
education <- education %>% select(State, County, `Less than a high school diploma, 2015-19`, `High scho
```

5)

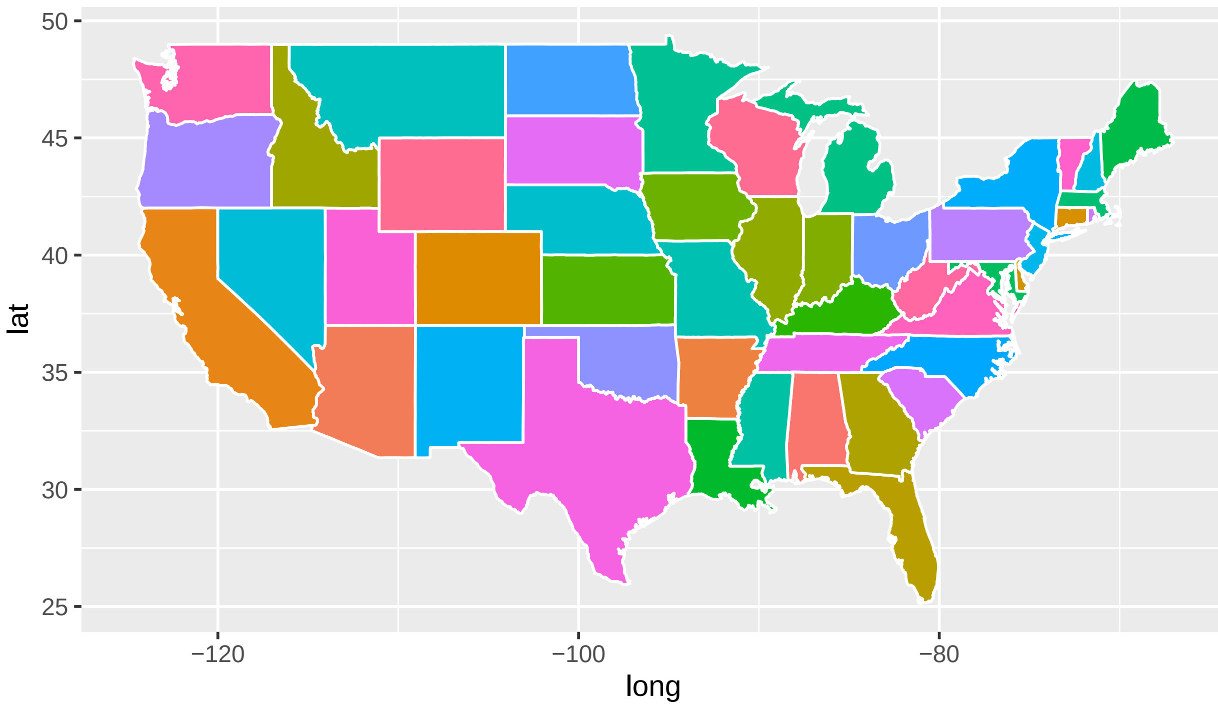
```r
education.state <- aggregate(education[,3:6], by = list(education$State), FUN = sum)
```

6)

```r
column_levels = colnames(education.state)[apply(education.state,1,which.max)]

state.level <- cbind(education.state, state_level = column_levels)
```

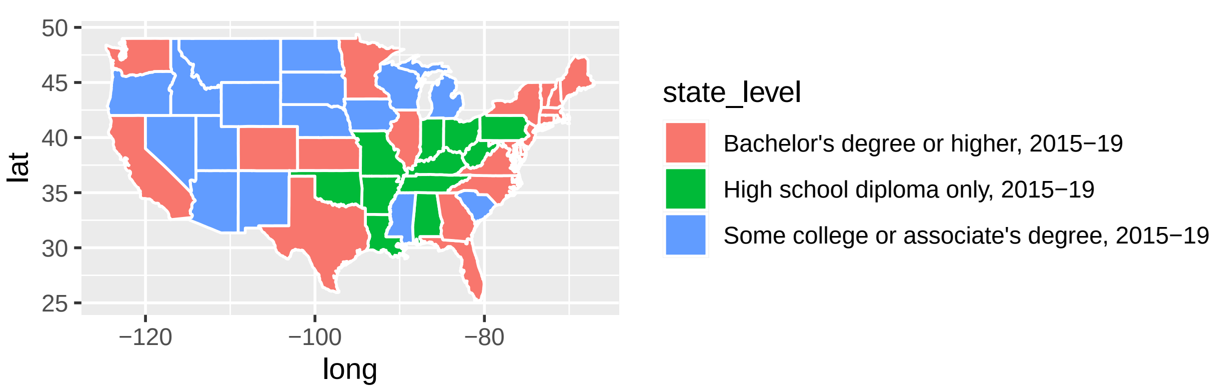
Visualization

```r
states <- map_data("state")
ggplot(data = states) +geom_polygon(aes(x = long, y = lat, fill = region, group = group),color = "white
guides(fill=FALSE) # color legend is unnecessary for this example and takes too long
```



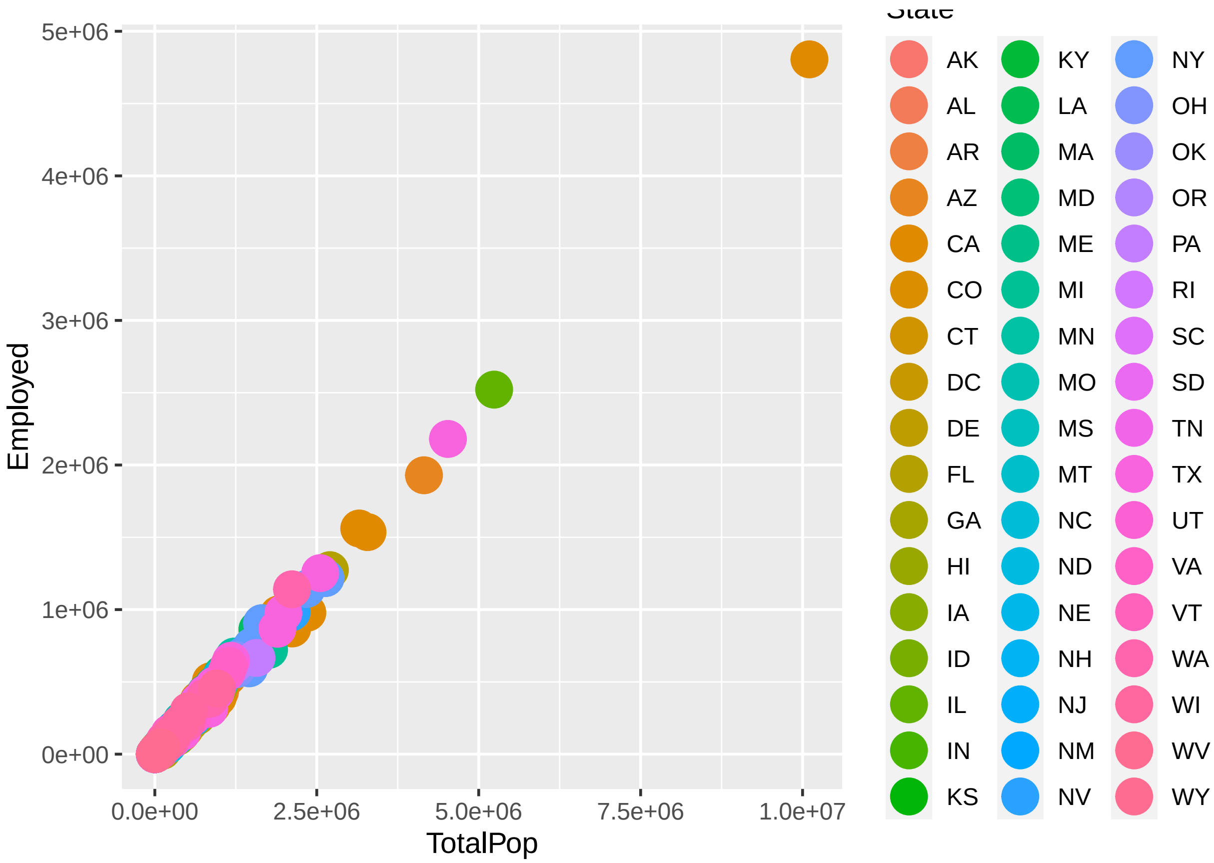
7)

```
state.level <- state.level %>% mutate(Group.1 = state.name[match(state.level$Group.1, state.abb)])
state.level$Group.1 <- tolower(state.level$Group.1)
state.merge <- states %>% left_join(state.level, by=  c("region" = "Group.1"))
ggplot(data = state.merge) +geom_polygon(aes(x = long, y = lat, fill = state_level, group = group),colo
```



8)

```
ggplot(census, aes(x=TotalPop, y=Employed, color= State)) + geom_point(size=6)
```



Answer: We plotted a scatter plot illustrating the number of employed individual compared to the total population of each of the states.

9)

```
#filters out missing values
census.clean <- na.omit(census)

census.clean <- census.clean %>% mutate(Men = Men / TotalPop, Employed = Employed / TotalPop, VotingAge
```

10)

```
head(census.clean, 5)
```

```
## # A tibble: 5 x 23
##   State County TotalPop   Men  Women White VotingAgeCitizen Poverty Professional
##   <chr> <chr>     <dbl> <dbl>  <dbl> <dbl>            <dbl>   <dbl>        <dbl>
## 1 AL    Autau~    55036 0.489  28137  75.4            0.745    13.7         35.3
## 2 AL    Baldw~   203360 0.489 103833  83.1            0.764    11.8         35.7
## 3 AL    Barbo~    26201 0.533  12225  45.7            0.774    27.2         25
## 4 AL    Bibb ~    22580 0.543  10329  74.6            0.782    15.2         24.4
## 5 AL    Bloun~    57667 0.494  29177  87.4            0.737    15.6         28.5
## # ... with 14 more variables: Service <dbl>, Office <dbl>, Production <dbl>,
## #   Drive <dbl>, Carpool <dbl>, Transit <dbl>, OtherTransp <dbl>,
## #   WorkAtHome <dbl>, MeanCommute <dbl>, Employed <dbl>, PrivateWork <dbl>,
## #   SelfEmployed <dbl>, FamilyWork <dbl>, Minority <dbl>
```

Dimensionality Reduction

11)

```
# run PCA for cleaned county level census data
pr.census = prcomp(census.clean[,-c(1:2)], scale = TRUE, center = TRUE)
# we chose to center and scale the features before running PCA in order to normalize the data so that e

# save PC1 and PC2 into two-column data frame
PC1 = pr.census$x[, 1]
PC2 = pr.census$x[, 2]
pc.county = data.frame(PC1, PC2)

# find three features with largest absolute values of PC1
PC1_loading = sort(abs(pr.census$rotation[, c(1)]), decreasing = TRUE)
head(PC1_loading, n = 3)
```

```
##   WorkAtHome SelfEmployed     Minority
##    0.3768940    0.3452038    0.3260893
```

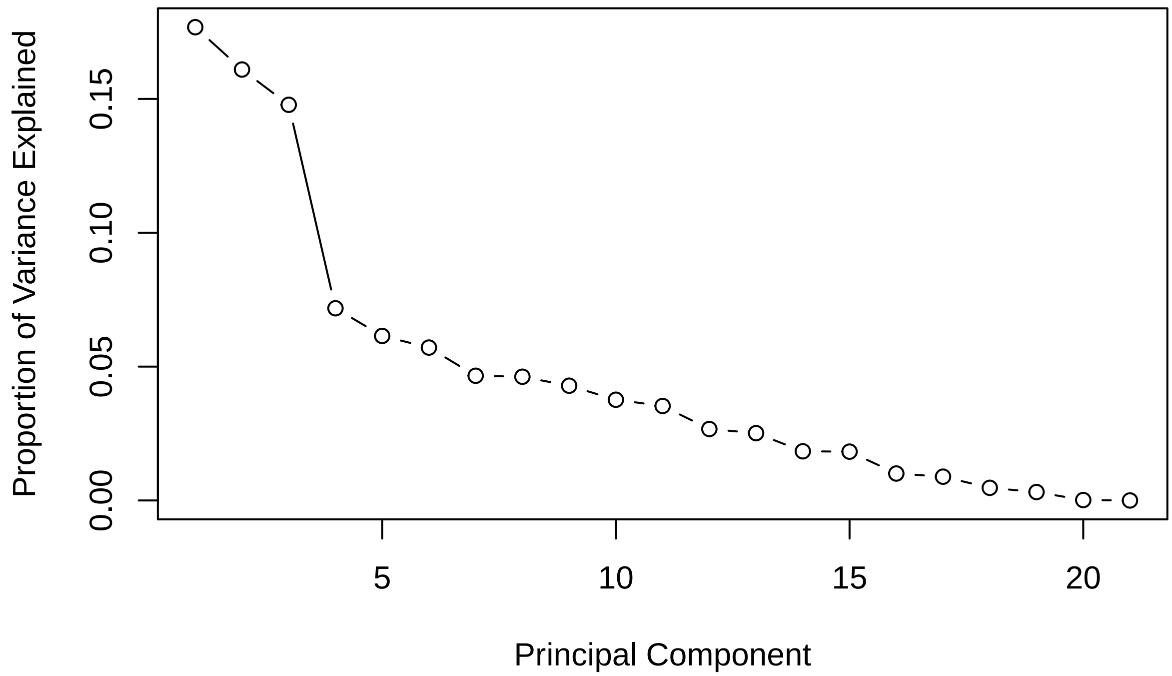
```
# three features with largest absolute values of first principal component are WorkAtHome, SelfEmployed

# find features that have opposite signs
diffsign = data.frame(pr.census$rotation[, 1], pr.census$rotation[, 2])
rows = c()
for (i in c(1:21)) {
  if (sign(diffsign[i, 1]) == sign(diffsign[i, 2])) {
    rows = c(rows, i)
  }
}
diffsign = diffsign [-c(rows),]
diffsign
```

```
##                    pr.census.rotation...1. pr.census.rotation...2.
## TotalPop                       0.084209128            -0.352440904
## Women                          0.084294222            -0.353273776
## White                         -0.322449822             0.321632308
## VotingAgeCitizen              -0.159465979             0.254913304
## Poverty                        0.322372526            -0.009731262
## Service                        0.173800329            -0.075687654
## Office                         0.125842149            -0.009498251
## Carpool                        0.115910771            -0.015936104
## Transit                        0.008546658            -0.323988499
## Minority                       0.326089271            -0.314564493
```

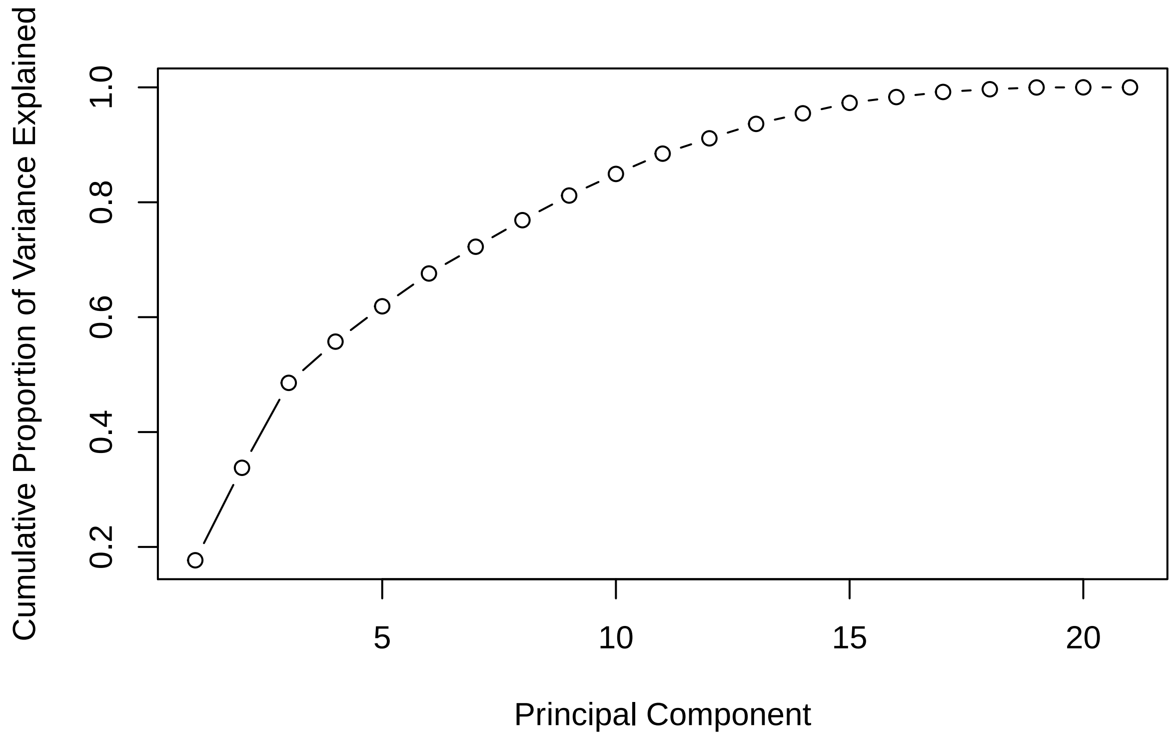
Answer: The features with opposite signs are TotalPop, Women, White, VotingAgeCitizen, Poverty, Service, Office, Carpool, Transit, and Minority. This means that there is a negative correlation between these features when taking PC1 and PC2.

12)

```
pr.var = pr.census$sdev^2
pve = pr.var/sum(pr.var)

# plot PVE
plot(pve, xlab = "Principal Component", ylab = "Proportion of Variance Explained", type = "b")
```




```
# plot Cumulative PVE
plot(cumsum(pve), xlab = "Principal Component", ylab = "Cumulative Proportion of Variance Explained", ty
```




```
# determine minimum number of PCs needed to capture 90% of variance for the analysis
bool_cumsum = ifelse(cumsum(pve) > 0.9, TRUE, FALSE)
min(which(cumsum(bool_cumsum) == TRUE))
```

```
## [1] 12
```

```
# We need 12 PCs in order to explain 90% of the total variation in the data
```

Clustering

13)

```
# compute a euclidean distance matrix between the subjects
census.clean.dist = dist(census.clean[, -c(1:2)])

# run hierarchical clustering using complete linkage
set.seed(123)
census.clean.hclust = hclust(census.clean.dist)

# cut the tree to partition the observations into 10 clusters
clus1 = cutree(census.clean.hclust, 10)
table(clus1)
```

```
## clus1
##    1    2    3    4    5    6    7    8    9   10
## 2789  245   73    2   12    1    2    5   12    1
```

```
# re-run hierarchical clustering w first 2 PC's from pc.county
pc.county.dist = dist(pc.county)
pc.county.hclust = hclust(pc.county.dist)
clus2 = cutree(pc.county.hclust, 10)
table(clus2)
```

```
## clus2
##    1    2    3    4    5    6    7    8    9   10
##  631 1590  123  658   24   76   13    1    5   21
```

```
which(census.clean$County == "Santa Barbara County")
```

```
## [1] 228
```

```
clus1[228]
```

```
## [1] 2
```

```
clus2[228]
```

```
## [1] 6
```

Modeling

```
all <- census.clean %>%left_join(education, by = c("State"="State", "County"="County")) %>%
  na.omit
```

14)

```
all <- all %>% mutate(Poverty = factor(ifelse(Poverty > 20, 1, 0))) %>% select(-State, -County)
```

```
# Partition the dataset into 80% training and 20% test data
set.seed(123)
n <- nrow(all)
idx.tr <- sample.int(n, 0.8*n)
all.tr <- all[idx.tr, ]
```

```r
all.te <- all[-idx.tr, ]

# define 10 cross-validation folds
set.seed(123)
nfold <- 10
folds <- sample(cut(1:nrow(all.tr), breaks=nfold, labels=FALSE))

# Error rate function
calc_error_rate = function(predicted.value, true.value){
  return(mean(true.value!=predicted.value))
}

# Creating records matrix
records = matrix(NA, nrow=3, ncol=2)
colnames(records) = c("train.error","test.error")
rownames(records) = c("tree","logistic","lasso")
```

Classification

15)

```r
# Decision Tree
all.tr.df <- data.frame(all.tr)
all.te.df <- data.frame(all.te)
colnames(all.tr.df)[22:25] <- c("LessThanHighschool", "Highschool", "College", "Bachelors")
colnames(all.te.df)[22:25] <- c("LessThanHighschool", "Highschool", "College", "Bachelors")
all.tree = tree(Poverty ~ ., data = all.tr.df)
summary(all.tree)
```

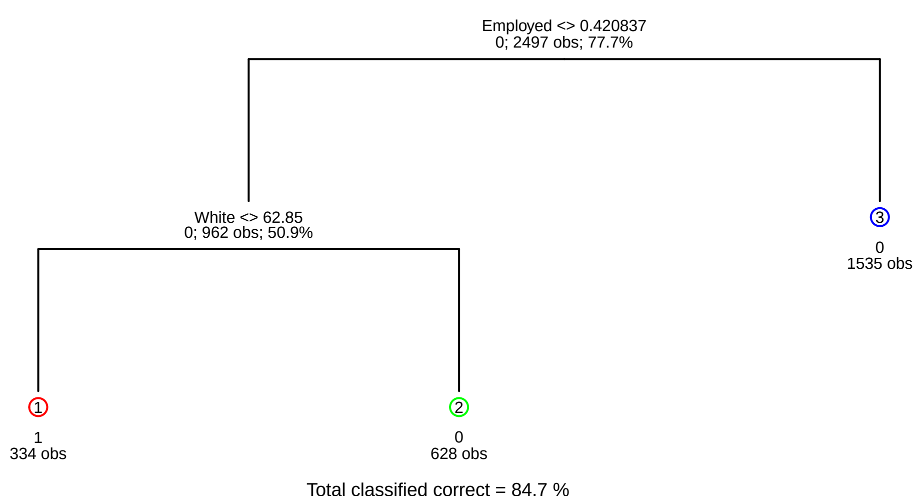
```
##
## Classification tree:
## tree(formula = Poverty ~ ., data = all.tr.df)
## Variables actually used in tree construction:
## [1] "Employed"     "White"        "Men"          "OtherTransp"  "SelfEmployed"
## Number of terminal nodes:  8
## Residual mean deviance:  0.65 = 1618 / 2489
## Misclassification error rate: 0.1534 = 383 / 2497
```

```r
# Drawing out tree
draw.tree(all.tree, nodeinfo = TRUE, cex = 0.5)
title("Tree")
```

Tree


```r
# Cross-validation
cv.tree <- cv.tree(all.tree, FUN =prune.misclass, rand = folds)
best.cv = min(cv.tree$size[cv.tree$dev == min(cv.tree$dev)])
prune.tree = prune.tree(all.tree, best = best.cv)

# Drawing prune tree
draw.tree(prune.tree, nodeinfo = TRUE, cex = 0.5)
title("Prune Tree")
```

Prune Tree




```r
set.seed(123)
test.pred.prune.tree = predict(prune.tree, all.te.df, type = "class")
train.pred.prune.tree = predict(prune.tree, all.tr.df, type = "class")

records[1,1] = calc_error_rate(train.pred.prune.tree, all.tr.df$Poverty)
records[1,2] = calc_error_rate(test.pred.prune.tree, all.te.df$Poverty)
```

```
records
```

```
##          train.error test.error
## tree       0.1533841     0.1648
## logistic          NA         NA
## lasso             NA         NA
```

16)

```r
# run a logistic regression to predict poverty in each county
all.trx = all.tr %>% select(-Poverty)
all.try = all.tr$Poverty
all.tex = all.te %>% select(-Poverty)
all.tey = all.te$Poverty
glm.fit = glm(Poverty ~ ., data = all.tr, family = binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Poverty ~ ., family = binomial, data = all.tr)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.1674  -0.4192  -0.1589  -0.0014   3.3874
##
## Coefficients: (1 not defined because of singularities)
##                                                 Estimate Std. Error z value
## (Intercept)                                    2.337e+01  5.592e+00   4.180
## TotalPop                                       4.491e-05  4.488e-05   1.001
## Men                                           -3.018e+01  3.300e+00  -9.147
## Women                                          2.486e-04  8.918e-05   2.788
## White                                          2.056e-02  3.920e-02   0.525
## VotingAgeCitizen                               3.902e+00  1.977e+00   1.974
## Professional                                   5.296e-02  2.514e-02   2.106
## Service                                        9.637e-02  2.868e-02   3.360
## Office                                         1.291e-02  3.048e-02   0.423
## Production                                     7.888e-02  2.304e-02   3.424
## Drive                                         -4.712e-02  2.886e-02  -1.633
## Carpool                                        5.694e-04  3.628e-02   0.016
## Transit                                        1.281e-01  6.457e-02   1.983
## OtherTransp                                   -1.122e-01  6.634e-02  -1.691
## WorkAtHome                                    -1.201e-01  4.792e-02  -2.506
## MeanCommute                                   -2.867e-02  1.641e-02  -1.747
## Employed                                      -2.950e+01  1.981e+00 -14.889
## PrivateWork                                   -2.717e-02  1.693e-02  -1.604
## SelfEmployed                                  -4.931e-02  3.204e-02  -1.539
## FamilyWork                                    -1.418e-01  1.800e-01  -0.788
## Minority                                       5.480e-02  3.907e-02   1.403
## `Less than a high school diploma, 2015-19`    -1.873e-04  3.778e-05  -4.958
## `High school diploma only, 2015-19`           -2.423e-04  3.496e-05  -6.931
## `Some college or associate's degree, 2015-19` -3.672e-04  4.778e-05  -7.686
## `Bachelor's degree or higher, 2015-19`        -2.404e-04  3.485e-05  -6.897
## Total_Population                                      NA         NA      NA
##                                               Pr(>|z|)
```

```
## (Intercept)                                      2.92e-05 ***
## TotalPop                                         0.317045
## Men                                               < 2e-16 ***
## Women                                            0.005311 **
## White                                            0.599885
## VotingAgeCitizen                                 0.048364 *
## Professional                                     0.035163 *
## Service                                          0.000780 ***
## Office                                           0.671945
## Production                                       0.000618 ***
## Drive                                            0.102520
## Carpool                                          0.987481
## Transit                                          0.047324 *
## OtherTransp                                      0.090929 .
## WorkAtHome                                       0.012218 *
## MeanCommute                                      0.080623 .
## Employed                                          < 2e-16 ***
## PrivateWork                                      0.108641
## SelfEmployed                                     0.123774
## FamilyWork                                       0.430727
## Minority                                         0.160701
## `Less than a high school diploma, 2015-19`      7.13e-07 ***
## `High school diploma only, 2015-19`             4.19e-12 ***
## `Some college or associate's degree, 2015-19`   1.52e-14 ***
## `Bachelor's degree or higher, 2015-19`          5.33e-12 ***
## Total_Population                                       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2650.6  on 2496  degrees of freedom
## Residual deviance: 1357.0  on 2472  degrees of freedom
## AIC: 1407
##
## Number of Fisher Scoring iterations: 9
```

```
fit.train = predict(glm.fit, all.trx, type = "response")
train.pred.glm = rep("0", length(all.try))
train.pred.glm[fit.train >= 0.5] = "1"

fit.test = predict(glm.fit, all.tex, type = "response")
test.pred.glm = rep("0", length(all.tey))
test.pred.glm[fit.test > 0.5] = "1"

# save training and test errors to records variable
records[2,1] = calc_error_rate(train.pred.glm, all.try)
records[2,2] = calc_error_rate(test.pred.glm, all.tey)
records
```

```
##          train.error test.error
## tree       0.1533841     0.1648
## logistic   0.1249499     0.1232
## lasso             NA         NA
```

Answer: The variables production; private work; less than a high school diploma, 1990; high school diploma only, 1990; and bachelor's degree or higher, 2015-19 are all statistically significant at level 0.01.

Answer: The variable production has a coefficient 5.797e-02, which means for every one unit change in production, the log odds of being in poverty increases by 5.797e-02, holding other variables fixed. Similarly the variable private work has a coefficient -4.570e-02, which means for every one unit change in private work, the log odds of being in poverty decreases by 4.570e-02, holding other variables fixed.

17)

```
set.seed(123)
lambda.equ = seq(1,20) * 1e-5
dat = model.matrix(Poverty ~ ., data = all)
x.train = dat[idx.tr,]
x.test = dat[-idx.tr,]

lasso.cv = cv.glmnet(x.train, all.tr$Poverty, alpha = 1, nfolds = 10, lambda = lambda.equ,family = "bin
best.lam = lasso.cv$lambda.min
best.lam
```

```
## [1] 1e-05
```

```
log.lasso = glmnet(dat, all$Poverty, alpha=1, nfolds = 10, lambda = best.lam, family = "binomial")
coef(log.lasso)
```

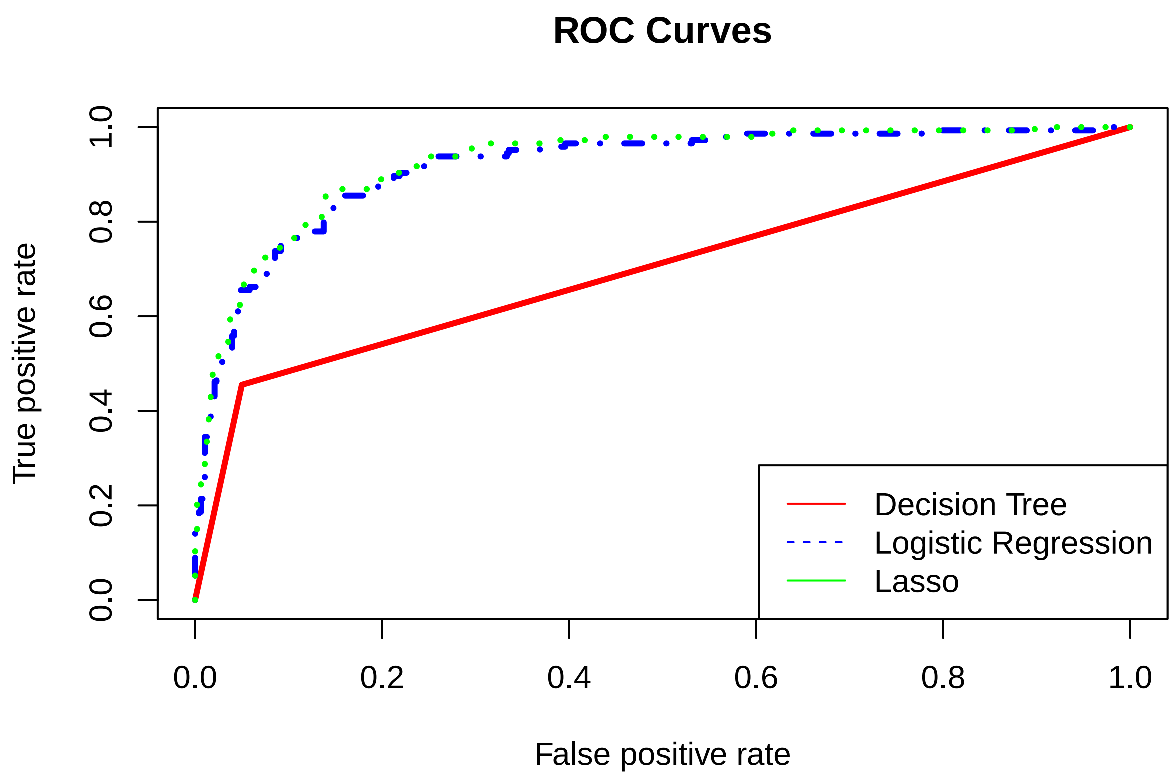
```
## 27 x 1 sparse Matrix of class "dgCMatrix"
##                                                        s0
## (Intercept)                                  2.201858e+01
## (Intercept)                                  .
## TotalPop                                     6.843049e-07
## Men                                         -2.870240e+01
## Women                                        2.837849e-04
## White                                        4.268383e-02
## VotingAgeCitizen                             3.777651e+00
## Professional                                 3.581687e-02
## Service                                      7.956846e-02
## Office                                       1.003646e-03
## Production                                   7.304135e-02
## Drive                                       -5.318192e-02
## Carpool                                     -1.027876e-02
## Transit                                      1.208183e-01
## OtherTransp                                 -6.818947e-02
## WorkAtHome                                  -1.374278e-01
## MeanCommute                                 -3.343090e-02
## Employed                                    -3.041732e+01
## PrivateWork                                 -2.157207e-02
## SelfEmployed                                -2.905276e-02
## FamilyWork                                  -3.263020e-03
## Minority                                     7.724469e-02
## `Less than a high school diploma, 2015-19`    -1.762874e-04
## `High school diploma only, 2015-19`           -1.940443e-04
## `Some college or associate's degree, 2015-19` -2.902850e-04
## `Bachelor's degree or higher, 2015-19`        -2.137114e-04
## Total_Population                             .
```

```r
lasso.train = predict(log.lasso, s = best.lam, newx = x.train)
train.pred.lasso = ifelse(lasso.train > 0.5, "1", "0")
lasso.test = predict(log.lasso,newx = x.test,s= best.lam)
test.pred.lasso = ifelse(lasso.test > 0.5, "1", "0")

lasso.test.error = calc_error_rate(test.pred.lasso, all.te$Poverty)
lasso.train.error = calc_error_rate(train.pred.lasso, all.tr$Poverty)


records[3,2]= lasso.test.error
records[3,1] = lasso.train.error
records
```

```
##          train.error test.error
## tree       0.1533841     0.1648
## logistic   0.1249499     0.1232
## lasso      0.1297557     0.1328
```

18)

```r
colnames(all.tex)[21:24] <- c("LessThanHighschool", "Highschool", "College", "Bachelors")

# compute ROC curves for decision tree, logistic regression, and lasso logistic regression
pruned.pred.tree = predict(prune.tree, all.tex, type = "class")
pred.tree = prediction(as.numeric(pruned.pred.tree), as.numeric(all.tey))
pred.log = prediction(as.numeric(fit.test), as.numeric(all.tey))
pred.lasso = prediction(lasso.test, as.numeric(all.tey))
tree.perf = performance(pred.tree, measure = "tpr", x.measure = "fpr")
log.perf = performance(pred.log, measure = "tpr", x.measure = "fpr")
lasso.perf = performance(pred.lasso, measure = "tpr", x.measure = "fpr")
plot(tree.perf, col = "red", lwd = 3, main = "ROC Curves")
plot(log.perf, col = "blue", lty = 4, lwd = 3, main = "ROC Curves", add = TRUE)
plot(lasso.perf, col = "green", lty = 3, lwd = 3, main = "ROC Curves", add = TRUE)
legend("bottomright", legend = c("Decision Tree", "Logistic Regression", "Lasso"), col = c("red", "blue
```

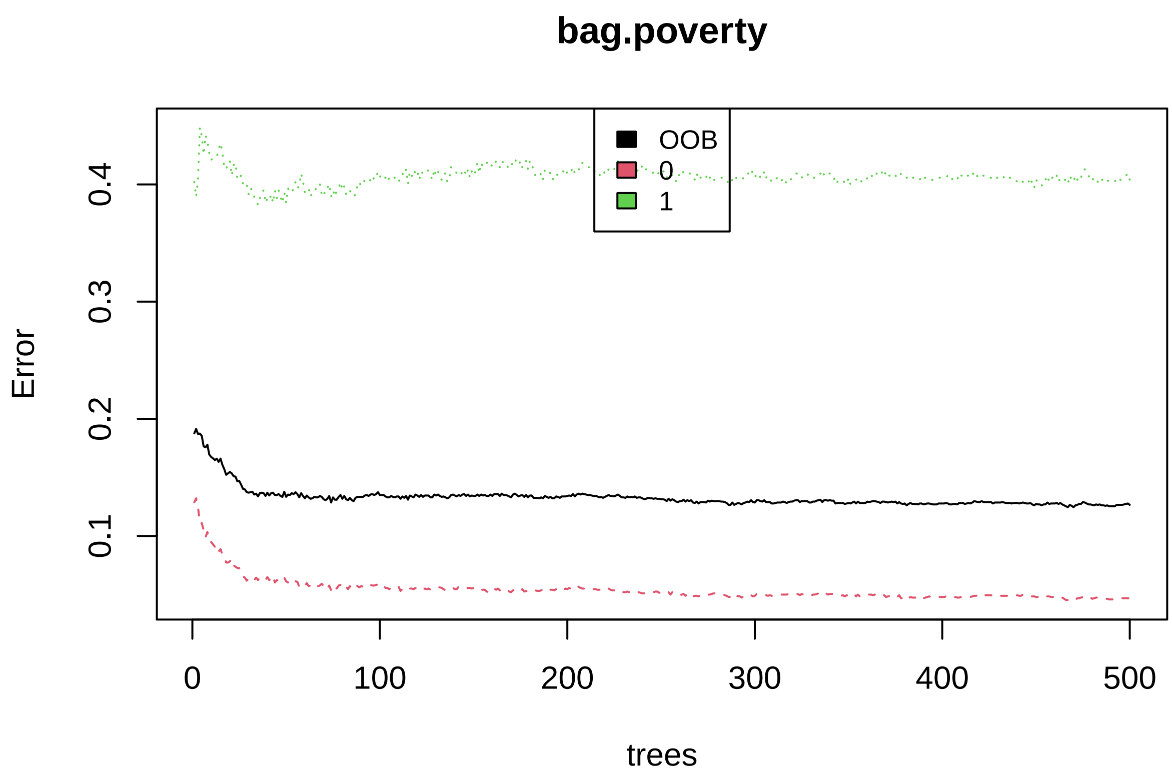


19)

```r
# Random Forest
set.seed(123)
bag.poverty = randomForest(Poverty ~ ., data = all.tr.df, importance = TRUE)
bag.poverty
```

```
##
## Call:
##  randomForest(formula = Poverty ~ ., data = all.tr.df, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 12.66%
## Confusion matrix:
##      0   1 class.error
## 0 1849  91  0.04690722
## 1  225 332  0.40394973
```

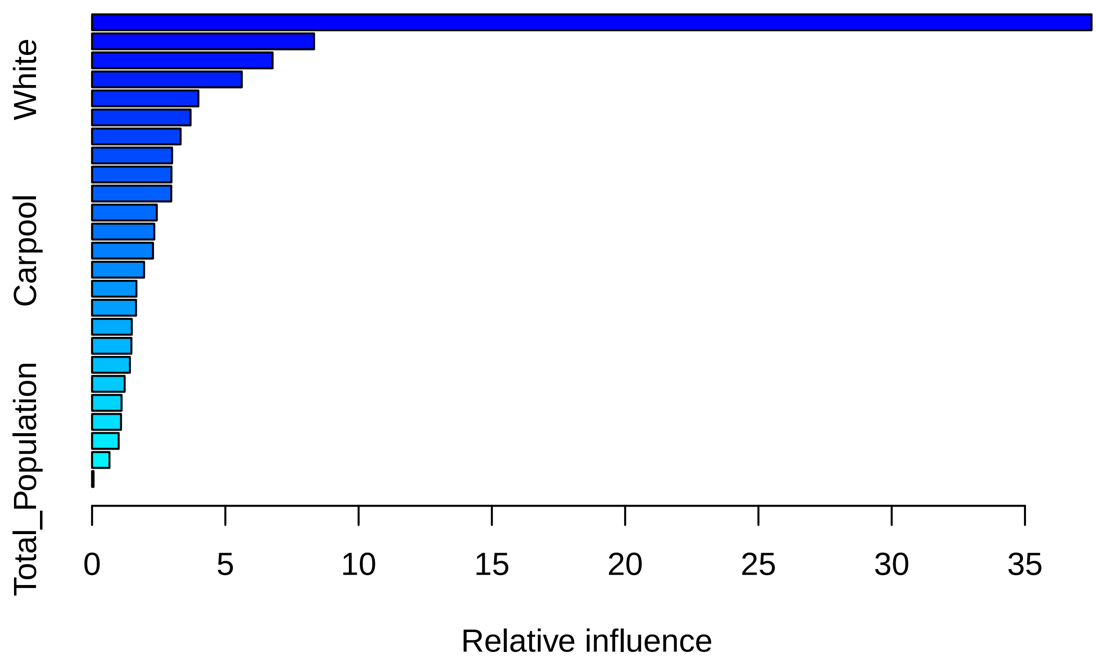
```r
plot(bag.poverty)
legend("top", colnames(bag.poverty$err.rate), col=1:4, cex = 0.8, fill=1:4)
```




```
yhat.bag = predict(bag.poverty, newdata = all.te.df, type = "response")
test.bag.error = mean(yhat.bag != all.te.df$Poverty)
test.bag.error
```

```
## [1] 0.12
```

```
# Boosting
boost.poverty = gbm(ifelse(Poverty == "1", 1, 0) ~ ., data = all.tr.df, distribution = "bernoulli", n.t
summary(boost.poverty)
```



```
##                                     var     rel.inf
## Employed                       Employed 37.49851983
## Men                                 Men  8.32810839
## Minority                       Minority  6.77330986
## White                             White  5.61715484
## VotingAgeCitizen       VotingAgeCitizen  3.98925697
## WorkAtHome                   WorkAtHome  3.69583912
## Professional               Professional  3.32221691
## SelfEmployed               SelfEmployed  3.00233932
## Service                         Service  2.97950047
## MeanCommute                 MeanCommute  2.97101389
## Production                   Production  2.42775447
## OtherTransp                 OtherTransp  2.33442236
## Carpool                         Carpool  2.28615679
## PrivateWork                 PrivateWork  1.95068742
## LessThanHighschool LessThanHighschool  1.66387000
## Women                             Women  1.64981442
## College                         College  1.49012793
## Drive                             Drive  1.47579030
## Office                           Office  1.42045085
## Bachelors                     Bachelors  1.22290263
## Highschool                   Highschool  1.10714162
## Transit                         Transit  1.08589546
## TotalPop                       TotalPop  0.99875176
## FamilyWork                   FamilyWork  0.65275709
## Total_Population       Total_Population  0.05621729
```

```r
yhat.boost = predict(boost.poverty, newdata = all.te.df, n.trees = 500, type = "response")
yhat.boost = ifelse(yhat.boost > 0.5, 1, 0)
test.boost.error = mean(yhat.boost != ifelse(all.te.df$Poverty == "1", 1, 0))
test.boost.error
```

```
## [1] 0.1296
```

Answer: The test errors for random forest and boosting are 0.12 and 0.136, respectively. These values are both lower than the test error for tree, and are around the same as logistic and lasso.

20)

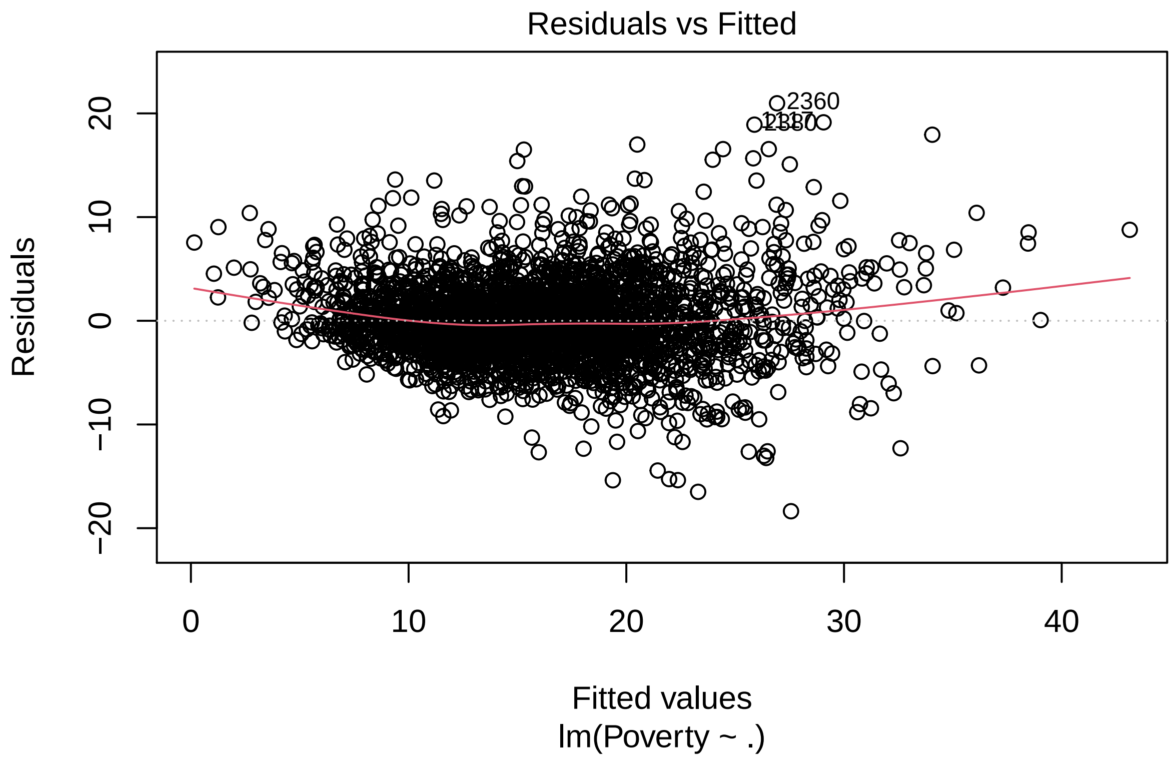
```r
all_regression <- census.clean %>%left_join(education, by = c("State"="State", "County"="County")) %>%
  na.omit
all_regression <- all_regression %>% select(-State, -County)
model1 <- lm(Poverty ~ ., data = all_regression)
summary(model1)
```

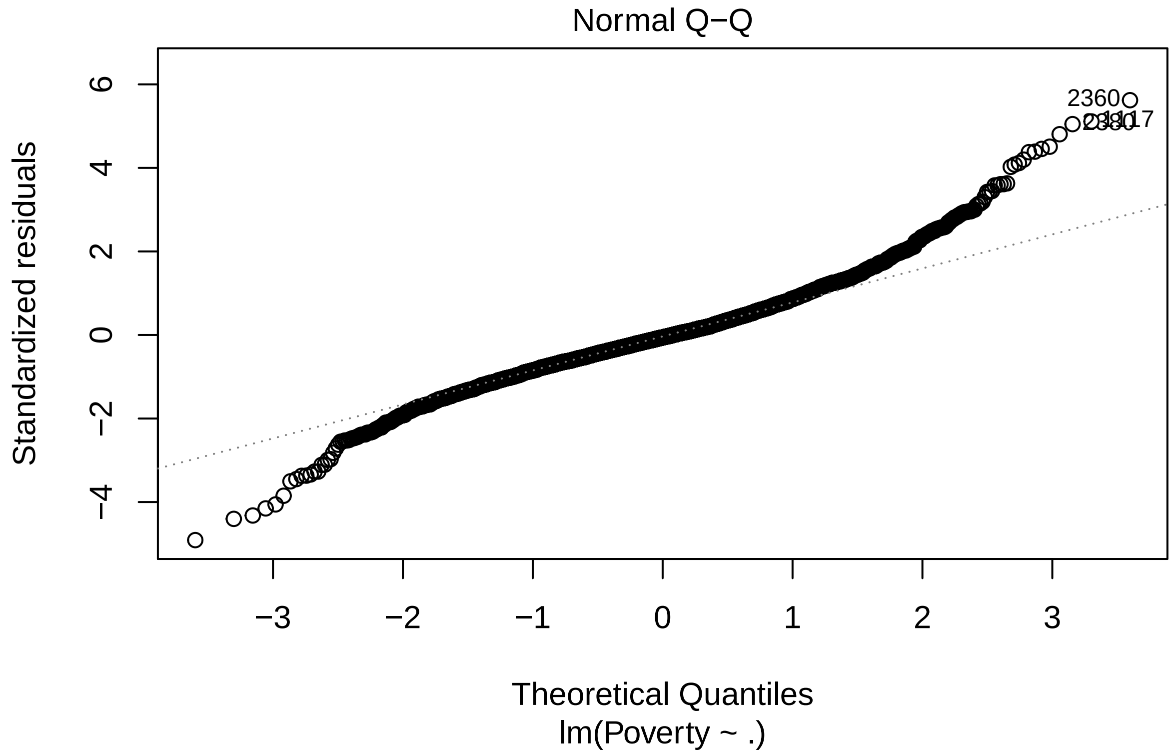
```
##
## Call:
## lm(formula = Poverty ~ ., data = all_regression)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.3653  -2.1945  -0.2262   1.9468  20.9757
##
## Coefficients: (1 not defined because of singularities)
##                                                  Estimate Std. Error t value
```

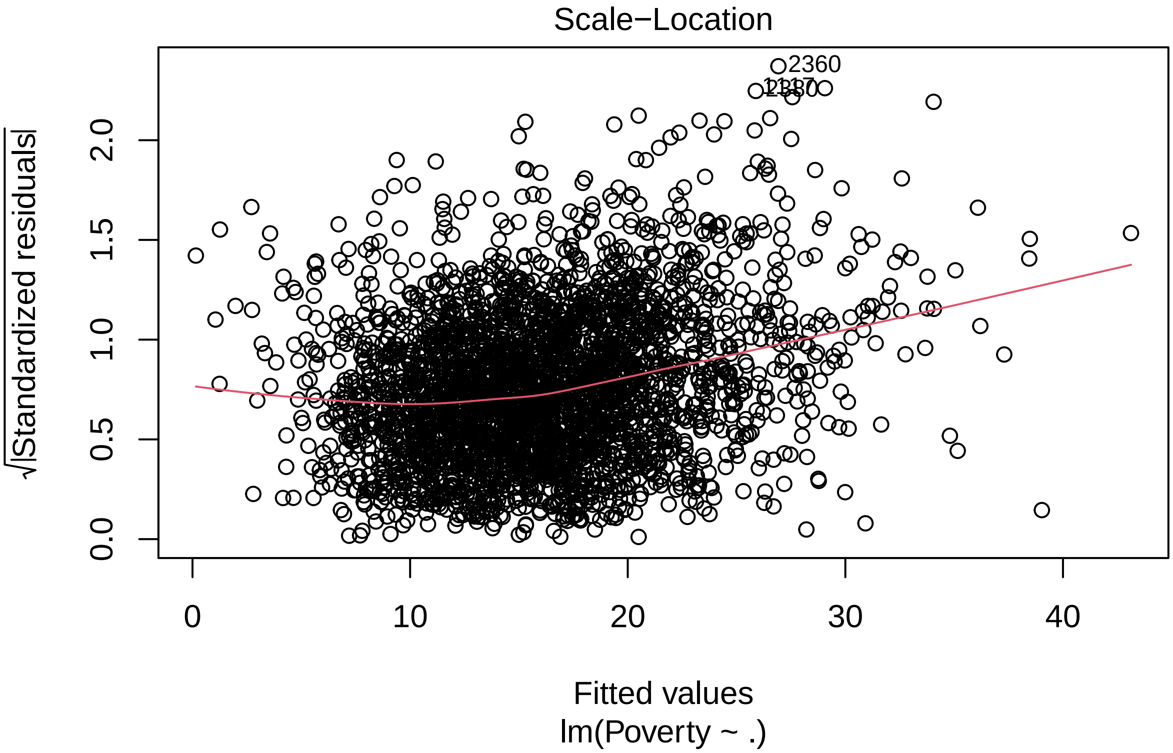
```
## (Intercept)                                 6.988e+01  5.894e+00  11.855
## TotalPop                                    2.378e-05  1.932e-05   1.231
## Men                                        -6.653e+01  3.197e+00 -20.806
## Women                                       2.782e-05  3.655e-05   0.761
## White                                       1.119e-01  4.324e-02   2.587
## VotingAgeCitizen                            7.227e+00  1.877e+00   3.850
## Professional                                2.252e-02  2.366e-02   0.952
## Service                                     2.035e-01  2.942e-02   6.915
## Office                                      8.897e-03  3.151e-02   0.282
## Production                                  2.034e-01  2.471e-02   8.233
## Drive                                      -1.073e-01  2.801e-02  -3.832
## Carpool                                    -6.627e-02  3.833e-02  -1.729
## Transit                                     1.356e-02  4.571e-02   0.297
## OtherTransp                                -1.207e-01  6.610e-02  -1.826
## WorkAtHome                                 -1.175e-01  4.605e-02  -2.552
## MeanCommute                                -1.362e-01  1.520e-02  -8.958
## Employed                                   -6.232e+01  1.585e+00 -39.309
## PrivateWork                                -7.134e-02  1.594e-02  -4.475
## SelfEmployed                               -1.092e-01  2.994e-02  -3.648
## FamilyWork                                  4.566e-01  1.687e-01   2.707
## Minority                                    1.976e-01  4.314e-02   4.581
## `Less than a high school diploma, 2015-19`    -5.362e-05  1.300e-05  -4.124
## `High school diploma only, 2015-19`           -6.416e-05  1.240e-05  -5.176
## `Some college or associate's degree, 2015-19` -6.045e-05  1.274e-05  -4.744
## `Bachelor's degree or higher, 2015-19`        -4.743e-05  8.647e-06  -5.485
## Total_Population                                   NA         NA      NA
##                                            Pr(>|t|)
## (Intercept)                                 < 2e-16 ***
## TotalPop                                   0.218354
## Men                                         < 2e-16 ***
## Women                                      0.446640
## White                                      0.009731 **
## VotingAgeCitizen                           0.000121 ***
## Professional                               0.341179
## Service                                    5.64e-12 ***
## Office                                     0.777689
## Production                                 2.65e-16 ***
## Drive                                      0.000130 ***
## Carpool                                    0.083907 .
## Transit                                    0.766791
## OtherTransp                                0.068006 .
## WorkAtHome                                 0.010761 *
## MeanCommute                                 < 2e-16 ***
## Employed                                    < 2e-16 ***
## PrivateWork                                7.92e-06 ***
## SelfEmployed                               0.000268 ***
## FamilyWork                                 0.006836 **
## Minority                                   4.81e-06 ***
## `Less than a high school diploma, 2015-19`    3.81e-05 ***
## `High school diploma only, 2015-19`           2.41e-07 ***
## `Some college or associate's degree, 2015-19` 2.19e-06 ***
## `Bachelor's degree or higher, 2015-19`        4.46e-08 ***
## Total_Population                                 NA
## ---
```

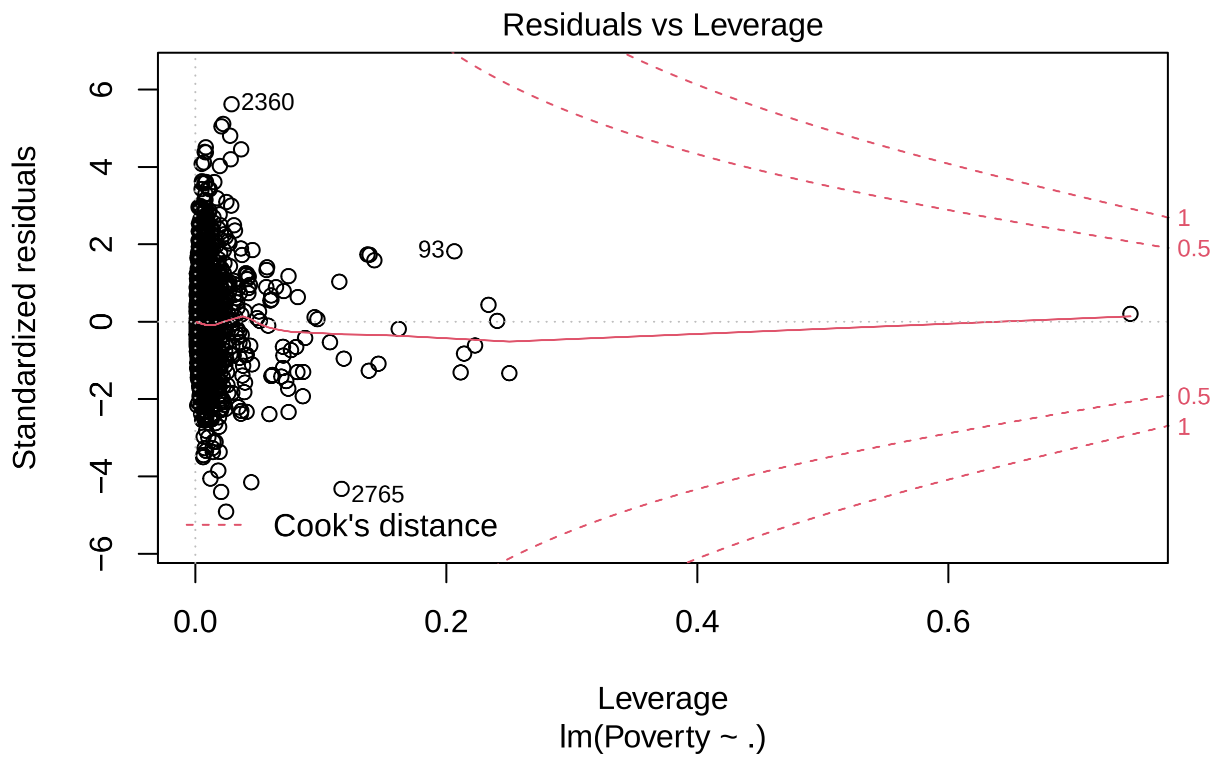
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.787 on 3097 degrees of freedom
## Multiple R-squared:  0.6686, Adjusted R-squared:  0.666
## F-statistic: 260.3 on 24 and 3097 DF,  p-value: < 2.2e-16
```

```
plot(model1)
```




Normal Q−Q
Standardized residuals
−4
−2
0
2
4
6
−3
−2
−1
0
1
2
3
2360
Theoretical Quantiles
lm(Poverty ~ .)


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2360
0
10
20
30
40
Fitted values
lm(Poverty ~ .)




```
test_MSE = mean((all_regression$Poverty - predict.lm(model1, all_regression))^2)
test_MSE
```

```
## [1] 14.22317
```

Answer: For question 20, we decided to use a linear regression model to predict poverty for each county given all the other predictor variables. We calculated an adjusted R-squared value of 0.666 and a test MSE of 14.22317. We also got a p-value of 2.2e-16 and a large F-statistic of 260.3. While this model is one option, we prefer the other classification models used as they are better suited to analyze such a large amount of variables.

21)

```
fullrecords = matrix(NA, nrow=5, ncol=2)
colnames(fullrecords) = c("train.error","test.error")
rownames(fullrecords) = c("tree","logistic","lasso", "Random Forest", "Boosting")

fullrecords[1,1] = calc_error_rate(train.pred.prune.tree, all.tr.df$Poverty)
fullrecords[1,2] = calc_error_rate(test.pred.prune.tree, all.te.df$Poverty)
fullrecords[2,1] = calc_error_rate(train.pred.glm, all.try)
fullrecords[2,2] = calc_error_rate(test.pred.glm, all.tey)
fullrecords[3,2]= lasso.test.error
fullrecords[3,1] = lasso.train.error
fullrecords[4,2] = test.bag.error
fullrecords[5,2] = test.boost.error
fullrecords
```

```
##                train.error test.error
## tree             0.1533841     0.1648
## logistic         0.1249499     0.1232
## lasso            0.1297557     0.1328
```

```
## Random Forest          NA     0.1200
## Boosting               NA     0.1296
```

```
sprintf("Test MSE for Linear Regression: %s", test_MSE)
```

```
## [1] "Test MSE for Linear Regression: 14.2231746255133"
```

Answer: When looking at the training and test errors for each different classification model, they were all very similar besides decision trees. While random forest had the lowest test error, we believe logistic regression is a more appropriate method of classification as it is able to accommodate all of the variables. In addition, logistic regression does not require as much computation, making it a more efficient model. When trying linear regression, we found it to not be a very good model as it isn't able to represent all the different variables well. PCA would have been another option to compare with the classification models as it uses a few components to represent a large majority of the data rather than having many variables. These methods could also be used to look at different counties or even countries around the world.