Stat 228 Midterm 1

Marisa Papagelis

03/16/2022

```
score <- read.csv(file=paste(filepath, "Scorecard.csv", sep=""), header=TRUE)
score$CONTROL[score$CONTROL == 1] <- "Public"
score$CONTROL[score$CONTROL == 2] <- "Private nonprofit"
score$CONTROL[score$CONTROL == 3] <- "Private for-profit"
score$CONTROL <- as.factor(score$CONTROL)
score <- score[complete.cases(score),] #consider data without missing values
score <- score[,-c(1,2,4,5)] #remove two ID variables, city, and state post code (STABBR)
```

Data Exploration In order to run a Principal Component Analysis of the PCIP's variables, we first subset the data set to contain only the PCIP variables. Then, we run the analysis with no standardization and plot the first two principal components against each other on a scatter plot. The scatter plot between the first two principal components displays a sideways cone shaped pattern with a very dense middle/tip, hinting that the two principal components are not linearly associated. When looking at their sample correlation coefficient, the value is -0.248357386, suggesting that there is a very week negative linear association between the two. However, they still are orthogonal to each other as shown by the inner product of -3.529442e-13, which is essentially zero. The Wellesley College observation is indexed at row 1289 and has a PC1 value of -0.1260393 and a PC2 value of -0.07370612. The PC1 value is less in magnitude compared to most other colleges/universities and the PC2 value is similar in magnitude compared to most other colleges/universities. Generally speaking, we cannot look at principal components for practical meanings; however, since each of the principal components places a majority of its weight on a particular PCIP variable, we can look at the corresponding weights of the linear combinations for clues. From the coefficients of the PC's, we can see that for PC1, a large amount of the weight (-7.002470e-01) is on PCIP52 and for PC2, a large amount of the weight (8.762832e-01) is on PCIP51. We can therefore asses that PC1 represents a great majority of PCIP52 (Percentage of degrees awarded in Business, Management, Marketing, And Related Support Services) for each college/university and PC2 represents a great majority of PCIP51 (Percentage of degrees awarded in Health Professions And Related Programs) for each college/university. Looking at the Wellesley College observation, we can hypothesize that the college does not consist of a significant percentages of degrees awarded in Business, Management, Marketing, etc. (PC1's prominent predictor) or in Health Professions and Related Programs (PC2's prominent predictor) because the weights of these PC's are quite low. This makes sense based on Wellesley College's lack of Business and Health majors. Lastly, Approximately 58.46% of variations among the PCIP's variables are explained by the first two principal components.

```
PCIP <- score[c(4:41)] # subset PCIP variables for PCA
# PCA with no standardization
result = prcomp(PCIP,scale=FALSE, center=FALSE)
# result$rotation # used to look at PC weights
# Wellesley observation index
which(score$INSTNM == "Wellesley College") # 1289
```

```
## [1] 1289
```

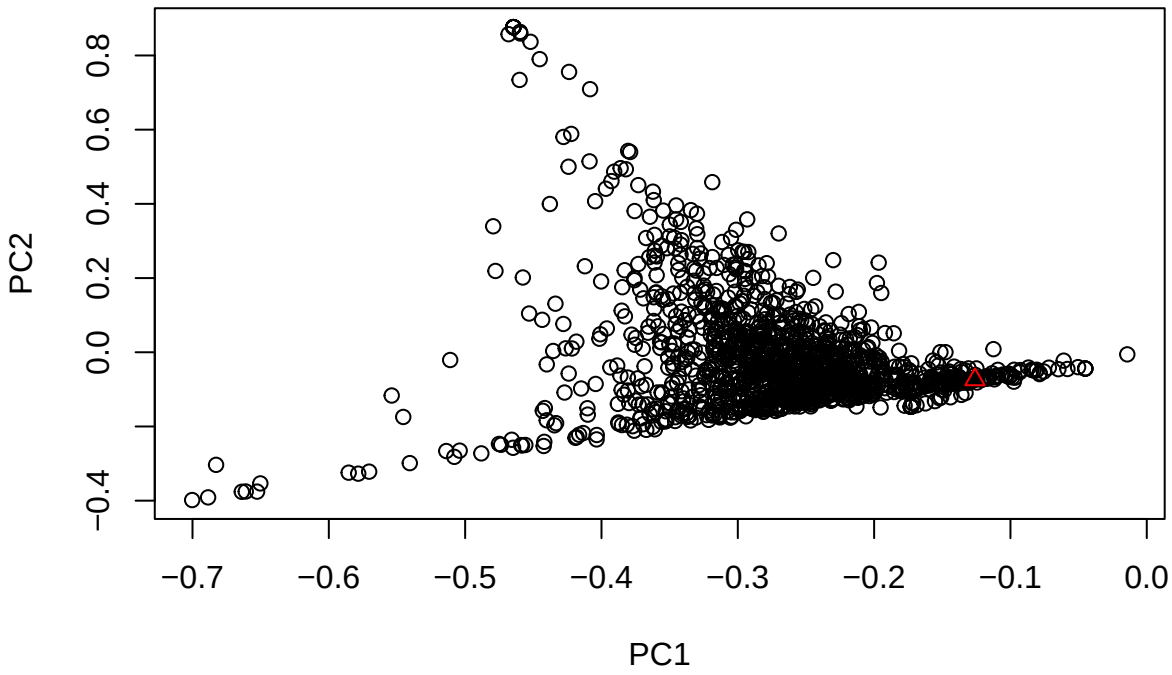
```
# find PC1 and PC2 values to plot Wellesley point
result$x[1289,1] # -0.1260393
```

```
## [1] -0.1260393
```

```
result$x[1289,2] # -0.07370612
```

```
## [1] -0.07370612
```

```
# Visualize first two principal components
plot(result$x[,1],result$x[,2],xlab="PC1",ylab="PC2", col='black')
points(-0.1260393,-0.07370612, col = "red", pch=24)
```




```
# Compute correlation between PC1 and PC2 (commented out b/c long)
# cor(result$x) # -0.248357386
# Compute inner product
t(result$x[,1])%*%result$x[,2] # -3.529442e-13
```

```
##                [,1]
## [1,] -3.529442e-13
```

```
# Percent of variations
V = (result$sdev)^2
V[1]/sum(V)
```

```
## [1] 0.436922
```

```
sum(V[1:2])/sum(V)
```

```
## [1] 0.5846455
```

Stepwise Regression For the model fitting process, we first use the correlation matrix and the vif step function to address the existence of multicollinearity among the potential predictor variables. Before that, we remove INSTNM (categorical var), md_earn_wne_p10 (response var), and remove CONTROL (categorical var) from our original data set. According to the vif step function, with a threshold of 5, PCIP51, the Percentage of degrees awarded in Health Professions And Related Programs, needs to be removed from the data set. Besides PCIP51, no other variables in the remaining dataset have a vif greater than 5. We note note that the vif of SAT_AVG remains larger than the rest (4.947149). Additionally, the largest pairwise correlation is between RET_FT4 and SAT_AVG with a correlateion coefficient of 0.7573872, so the current

set of predictors does have some multicollinear structure. After removing PCIP51 from the original data set, we move forward with model selection using step-wise regression. The optimal first-order AIC model that predicts median earnings after 10 years of graduation consists of 24 predictors, while the optimal first-order BIC model consists of 12 predictors (both sets of predictors summarized in the code below). In terms of model fit, the optimal AIC model has an adjusted R-squared of 0.8045 and the optimal BIC model has an adjusted R-squared of 0.7981. By 5-fold cross validation, the optimal AIC model has a CV score of 21958516, while the optimal BIC model has a CV score of 22067494. Because the AIC optimal model has a smaller CV score, indicating a stronger predictive power on independent observations, and a larger adjusted R-squared, indicating a better model fit, we would prefer optimal AIC model.

The optimal models *without the coefficients* are as follows. Reference the summary tables in the code for the coefficients. Optimal AIC Model: md_earn_wne_p10-hat = CONTROL + SAT_AVG + PCIP01 + PCIP10 + PCIP11 + PCIP13 + PCIP14 + PCIP22 + PCIP23 + PCIP27 + PCIP30 + PCIP31 + PCIP38 + PCIP39 + PCIP44 + PCIP45 + PCIP47 + PCIP49 + PCIP50 + UGDS + PCTPELL + RET_FT4 + PCTFLOAN + gt_25k_p6; Optimal BIC Model: md_earn_wne_p10-hat = SAT_AVG + PCIP11 + PCIP13 + PCIP14 + PCIP45 + PCIP47 + PCIP49 + UGDS + PCTPELL + RET_FT4 + PCTFLOAN + gt_25k_p6

```
# use vif and correlation matrix to diagnose existence of multi-collinearity
# remove INSTNM, md_earn_wne_p10 (response var), and remove CONTROL (categorical var)
vif <- score[-c(1,2,47)]
# cor(vif)
# VIF step function with threshold 5
library(usdm)
```

```
## Loading required package: sp
```

```
## Loading required package: raster
```

```
vifstep(vif,th=5) # remove PCIP51
```

```
## 1 variables from the 45 input variables have collinearity problem:
##
## PCIP51
##
## After excluding the collinear variables, the linear correlation coefficients ranges between:
## min correlation ( PCIP43 ~ PCIP29 ):  7.805391e-05
## max correlation ( RET_FT4 ~ SAT_AVG ):  0.7573872
##
## ---------- VIFs of the remained variables --------
##             Variables      VIF
## 1             SAT_AVG 4.947149
## 2              PCIP01 1.169393
## 3              PCIP03 1.346751
## 4              PCIP04 1.143904
## 5              PCIP05 1.528775
## 6              PCIP09 1.237840
## 7              PCIP10 1.110959
## 8              PCIP11 1.152247
## 9              PCIP12 1.187346
## 10             PCIP13 1.490704
## 11             PCIP14 1.795479
## 12             PCIP15 1.301491
## 13             PCIP16 1.570811
## 14             PCIP19 1.128990
## 15             PCIP22 1.048825
## 16             PCIP23 1.897314
```

```
## 17             PCIP24 1.572899
## 18             PCIP25 1.018175
## 19             PCIP26 1.816074
## 20             PCIP27 1.760918
## 21             PCIP29 1.024632
## 22             PCIP30 1.134748
## 23             PCIP31 1.210055
## 24             PCIP38 1.102861
## 25             PCIP39 1.737036
## 26             PCIP40 1.858974
## 27             PCIP41 1.013585
## 28             PCIP42 1.352214
## 29             PCIP43 1.296154
## 30             PCIP44 1.145020
## 31             PCIP45 2.558258
## 32             PCIP46 1.375119
## 33             PCIP47 1.829000
## 34             PCIP48 1.349562
## 35             PCIP49 1.472614
## 36             PCIP50 2.114628
## 37             PCIP52 1.545396
## 38             PCIP54 1.849018
## 39               UGDS 1.559980
## 40            PCTPELL 3.968361
## 41            RET_FT4 2.907715
## 42           PCTFLOAN 3.283749
## 43 GRAD_DEBT_MDN_SUPP 2.276307
## 44          gt_25k_p6 3.370275
```

```
# remove INSTNM and PCIP51 from original data set for model selection
score.new <- score[,-c(1,39)]
attach(score.new)
model.full <- lm(md_earn_wne_p10~.,data=score.new) # fit model with all predictors
# Step-wise Regression
# AIC model
AIC.model <- step(model.full,direction="both", trace=0,  steps=1000, k=2)
summary(AIC.model)
```

```
##
## Call:
## lm(formula = md_earn_wne_p10 ~ CONTROL + SAT_AVG + PCIP01 + PCIP10 +
##     PCIP11 + PCIP13 + PCIP14 + PCIP22 + PCIP23 + PCIP27 + PCIP30 +
##     PCIP31 + PCIP38 + PCIP39 + PCIP44 + PCIP45 + PCIP47 + PCIP49 +
##     PCIP50 + UGDS + PCTPELL + RET_FT4 + PCTFLOAN + gt_25k_p6,
##     data = score.new)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -14602  -2553   -234   1959  55253
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -1.402e+04  3.096e+03  -4.529 6.48e-06 ***
## CONTROLPrivate nonprofit    6.868e+02  1.719e+03   0.400 0.689484
## CONTROLPublic              -4.381e+02  1.726e+03  -0.254 0.799719
```

```
## SAT_AVG                   7.075e+00  2.042e+00   3.465 0.000547 ***
## PCIP01                   -1.084e+04  5.189e+03  -2.090 0.036852 *
## PCIP10                    2.280e+04  1.261e+04   1.808 0.070819 .
## PCIP11                   -1.304e+04  3.581e+03  -3.640 0.000283 ***
## PCIP13                   -1.093e+04  2.003e+03  -5.456 5.83e-08 ***
## PCIP14                    2.111e+04  1.693e+03  12.471  < 2e-16 ***
## PCIP22                   -3.107e+04  1.519e+04  -2.045 0.041087 *
## PCIP23                   -1.341e+04  5.752e+03  -2.332 0.019862 *
## PCIP27                    2.554e+04  1.164e+04   2.194 0.028415 *
## PCIP30                   -7.052e+03  3.416e+03  -2.064 0.039186 *
## PCIP31                   -1.238e+04  3.755e+03  -3.297 0.001005 **
## PCIP38                   -7.910e+03  5.251e+03  -1.506 0.132252
## PCIP39                   -5.141e+03  1.361e+03  -3.776 0.000166 ***
## PCIP44                   -9.588e+03  4.178e+03  -2.295 0.021893 *
## PCIP45                    1.369e+04  2.515e+03   5.445 6.19e-08 ***
## PCIP47                   -3.511e+04  1.247e+04  -2.815 0.004953 **
## PCIP49                    2.602e+04  5.281e+03   4.927 9.45e-07 ***
## PCIP50                   -2.385e+03  1.283e+03  -1.859 0.063288 .
## UGDS                     -7.446e-02  2.358e-02  -3.158 0.001625 **
## PCTPELL                   9.871e+03  1.618e+03   6.103 1.38e-09 ***
## RET_FT4                   1.438e+04  1.824e+03   7.887 6.56e-15 ***
## PCTFLOAN                 -5.656e+03  1.079e+03  -5.241 1.86e-07 ***
## gt_25k_p6                 6.009e+04  1.914e+03  31.398  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4472 on 1292 degrees of freedom
## Multiple R-squared:  0.8082, Adjusted R-squared:  0.8045
## F-statistic: 217.7 on 25 and 1292 DF,  p-value: < 2.2e-16
```

```
# BIC model
BIC.model <- step(model.full,direction="both", trace=0,  steps=1000, k=log(length(score.new$CONTROL)))
summary(BIC.model)
```

```
##
## Call:
## lm(formula = md_earn_wne_p10 ~ SAT_AVG + PCIP11 + PCIP13 + PCIP14 +
##     PCIP45 + PCIP47 + PCIP49 + UGDS + PCTPELL + RET_FT4 + PCTFLOAN +
##     gt_25k_p6, data = score.new)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -16251  -2479   -264   2073  55925
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.892e+04  2.457e+03  -7.700 2.68e-14 ***
## SAT_AVG      8.122e+00  1.883e+00   4.313 1.73e-05 ***
## PCIP11      -1.143e+04  3.411e+03  -3.352 0.000825 ***
## PCIP13      -1.055e+04  1.938e+03  -5.442 6.30e-08 ***
## PCIP14       2.265e+04  1.628e+03  13.912  < 2e-16 ***
## PCIP45       1.538e+04  2.224e+03   6.916 7.27e-12 ***
## PCIP47      -3.354e+04  1.252e+04  -2.679 0.007488 **
## PCIP49       2.596e+04  5.268e+03   4.929 9.34e-07 ***
## UGDS        -1.250e-01  1.974e-02  -6.330 3.36e-10 ***
```

```
## PCTPELL      1.060e+04  1.572e+03   6.745 2.29e-11 ***
## RET_FT4      1.464e+04  1.788e+03   8.187 6.31e-16 ***
## PCTFLOAN    -5.130e+03  1.031e+03  -4.974 7.43e-07 ***
## gt_25k_p6    6.343e+04  1.638e+03  38.732  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4544 on 1305 degrees of freedom
## Multiple R-squared:  0.7999, Adjusted R-squared:  0.7981
## F-statistic: 434.7 on 12 and 1305 DF,  p-value: < 2.2e-16
```

```
# 5-fold Cross Validation
set.seed(1)
n <- 1318 #sample size
K <- 5 #5-fold CV as an example
n.fold <- floor(n/K) #size of each fold, rounded down to the nearest integer (so the last fold might be
n.shuffle <- sample(1:n, n, replace=FALSE) #shuffle the n indexes
index.fold <- list()
for(i in 1:K) {
if(i<K) {
index.fold[[i]] <- n.shuffle[((i-1)*n.fold+1):(i*n.fold)]
}else {
index.fold[[i]] <- n.shuffle[((K-1)*n.fold+1):n]}}
# AIC model
CV.score <- 0
for(i in 1:K) {
#fit the full model based on the data excluding the ith fold
fit <- lm(md_earn_wne_p10 ~ as.factor(CONTROL) + SAT_AVG + PCIP01 + PCIP10 + PCIP11 + PCIP13 + PCIP14
  + PCIP22 + PCIP23 + PCIP27 + PCIP30 + PCIP31 + PCIP38 + PCIP39 + PCIP44 + PCIP45 + PCIP47
  + PCIP49 + PCIP50 + UGDS + PCTPELL + RET_FT4 + PCTFLOAN + gt_25k_p6, data=score.new[-index.fold[[i]],]
pred <- predict(fit,score.new[index.fold[[i]],]) #make prediction on each observation in the ith fold
CV.score <- CV.score+(1/n)*sum((md_earn_wne_p10[index.fold[[i]]]-pred)^2)} #compute average squared err
CV.score # 21958516
```

```
## [1] 21958516
```

```
# BIC model
CV.score <- 0
for(i in 1:K) {
#fit the full model based on the data excluding the ith fold
fit <- lm(md_earn_wne_p10 ~ SAT_AVG + PCIP11 + PCIP13 + PCIP14 + PCIP45 + PCIP47 + PCIP49 + UGDS
  + PCTPELL + RET_FT4 + PCTFLOAN + gt_25k_p6, data=score.new[-index.fold[[i]],])
pred <- predict(fit,score.new[index.fold[[i]],]) #make prediction on each observation in the ith fold
CV.score <- CV.score+(1/n)*sum((md_earn_wne_p10[index.fold[[i]]]-pred)^2)} #compute average squared err
CV.score # 22067494
```

```
## [1] 22067494
```

Lasso Regression When considering the Lasso regression for building a first-order model, we first needed to define an X matrix of all predictor variables, including dummy variables created for the categorical predictor CONTROL. These dummy variables were created as CONTROLprivprof, CONTROLnonprof, and CONTROLpublic, each one taking a value of 1 if the original CONTROL variable is equal to "Private for-profit," "Private nonprofit," or "Public" respectively, and a value of 0 otherwise. We run the lasso regression and find that 12 of the coefficients were set to zero exactly (PCIP03, PCIP05, PCIP12, PCIP16, PCIP19, PCIP24, PCIP29, PCIP41, PCIP42, PCIP48, CONTROLprivprof and CONTROLpublic). We find the optimal lambda value that minimizes test MSE to be 78.31361 and the Lasso model to include 35

predictor variables: SAT_AVG, PCIP01, PCIP04, PCIP09, PCIP10, PCIP11, PCIP13, PCIP14, PCIP15, PCIP22, PCIP23, PCIP25, PCIP26, PCIP27, PCIP30, PCIP31, PCIP38, PCIP39, PCIP40, PCIP43, PCIP44, PCIP45, PCIP46, PCIP47, PCIP49, PCIP50, PCIP52, PCIP54, UGDS, PCTPELL, RET_FT4, PCTFLOAN, GRAD_DEBT_MDN_SUPP, gt_25k_p6 and CONTROLnonprof. The Lasso model is different, with many more predictors, from the AIC and BIC models found in the step-wise regression. In practice, there is a trade-off between bias and variance when choosing whether to use step-wise regression or Lasso. This trade-off must be evaluated by the researcher, and the decision is based on personal preference or the needs of the specific model in question. Although the Lasso solution tends to produce parameter estimations that are less variable than the LS solution, in practice, I might prefer the a step-wise regression model because as an LS solution it is unbiased and will not penalize individual parameters to zero exactly. Generally, Lasso tends to produce parameter estimations that are less variable than LS step-wise solutions, creating a smaller SD; however, step-wise techniques tend to choose smaller, more precise (in terms of predictors) models which may be easier for me to work with.

```
library(glmnet)
```

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

```
## Loaded glmnet 4.1-3
```

```
# create CONTROL dummy vars
score.new$CONTROLprivprof <- 0
score.new$CONTROLprivprof[score.new$CONTROL == "Private for-profit"] <- 1
score.new$CONTROLnonprof <- 0
score.new$CONTROLnonprof[score.new$CONTROL == "Private nonprofit"] <- 1
score.new$CONTROLpublic <- 0
score.new$CONTROLpublic[score.new$CONTROL == "Public"] <- 1
# remove CONTROL, md_earn_wne_p10 from score.new to get x-vars
xvars <- score.new[-c(1,45)]
x <- as.matrix(xvars)
y <- score.new$md_earn_wne_p10
# Lasso solution
result.lasso <- cv.glmnet(x=x,y=y,family="gaussian",alpha=1)
result.lasso # 35 non-zero predictors
```

```
##
## Call:  cv.glmnet(x = x, y = y, family = "gaussian", alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Index  Measure      SE Nonzero
## min   78.3    51 22146612 3061984      35
## 1se  965.5    24 24994701 3383766       6
```

```
# report lasso coefs at lambda level that yields min CV MSE
coefs.lasso <- as.vector(coef(result.lasso,s=result.lasso$lambda.min))
# coefficients set to 0 exactly
sum(coefs.lasso==0) # 12
```

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

```
# find coefficients of optimal model
model.lasso <- glmnet(x, y, alpha = 1, lambda = result.lasso$lambda.min)
# coef(model.lasso)
```