PCA and Factor Analysis

Jacob Kramp

8/8/2021

Introduction

We will be evaluating two datsets using a variety of parametric methods and evaluating which method will model our given data sets best. We will do so by using an objective measure of fit, Mean Squared Error, in a cross validation process. By splitting our data sets in training and testing subsets, we can give our models/methods enough information to make predictions on the test datasets in a supervised setting (We will know what the true observations are). Our first dataset includes information surrounding the exclusivity of a college and we have been asked to model exclusivity as a response variable. In the second data set, we will be using Principal component analysis and factor analysis to model a set of response variables providing information on image sequences of subjects while driving in real scenarios. Specifically, we'll model their head direction captured in the images.

Analysis of the College Dataset

The resulting MSE for each method attempted on the dataset is shown below. We observed that since there is not a significant amount of overfitting or collinearity, regularization methods didn't perform as well as using a Multiple Linear Regression model.

	MSE
Linear	363.2999
Ridge	368.5181
Lasso	365.4677
Elastic	367.0610
PCR	406.6160
PLSR	380.5376

To better show this, we can print the summary of our PCR result in R.

```
## Data:    X dimension: 582 15
##  Y dimension: 582 1
## Fit method: svdpc
## Number of components considered: 15
## TRAINING: % variance explained
```

```
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X  35.9922   55.554   64.093    70.34    76.38    80.92    85.06    88.48
## y   0.0611    6.017    6.304    15.13    15.34    16.77    18.96    20.71
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    91.35     93.72     95.66     97.38     98.59     99.43     100.0
## y    20.80     22.42     22.61     26.58     28.61     28.62      29.2
```

If PCR was expected to perform well, the amount of variance explained would taper off with each added component after a sufficient number of components were used. Here, we see that it takes almost all of the components to see a small exchange of information for shrinkage. This is not ideal, and this result is not surprising after seeing our MSE for PCR compared to the Linear Regression model. We essentially have to use almost all of our components.

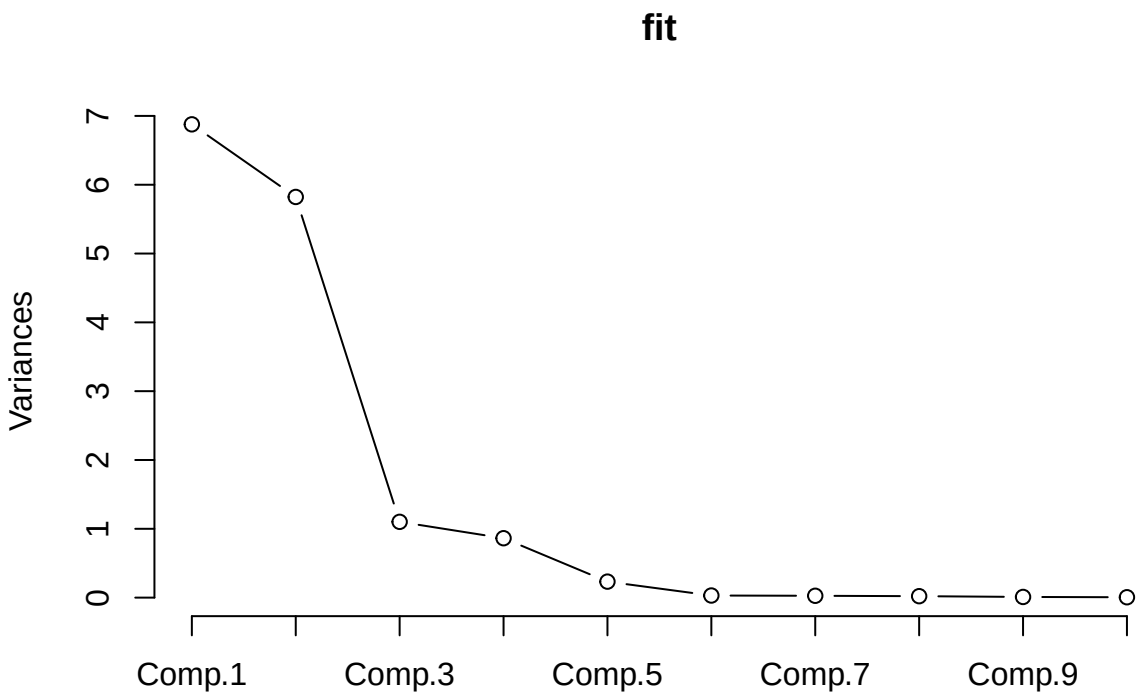
For comparison, we can also view the result of the PLSR model.

```
## Data:    X dimension: 582 15
##  Y dimension: 582 1
## Fit method: kernelpls
## Number of components considered: 15
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           18.35    16.97    16.30    16.13    15.99    15.98    15.98
## adjCV        18.35    16.97    16.28    16.12    15.96    15.95    15.95
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV       15.98    15.98    15.98     15.98     15.98     15.98     15.98
## adjCV    15.95    15.95    15.95     15.95     15.95     15.95     15.95
##        14 comps  15 comps
## CV        15.98     15.98
## adjCV     15.95     15.95
##
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X    17.65    31.94    60.43    64.62    70.04    75.47    78.07    80.71
## y    15.98    24.05    25.96    28.40    28.90    29.07    29.16    29.19
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    83.73     87.79     90.94     94.34     96.68     98.26     100.0
## y    29.20     29.20     29.20     29.20     29.20     29.20      29.2
```

Notice that here we do see an "elbowing" effect where there is a small amount of added variance explained after 4-6 components are added. This tells us that PLSR may have performed better and made more sense to use over PCR, but it still didn't perform better than our Multiple Linear Regression Model.We can use just 6 of our components here.

Analysis of Facial Recognition Data

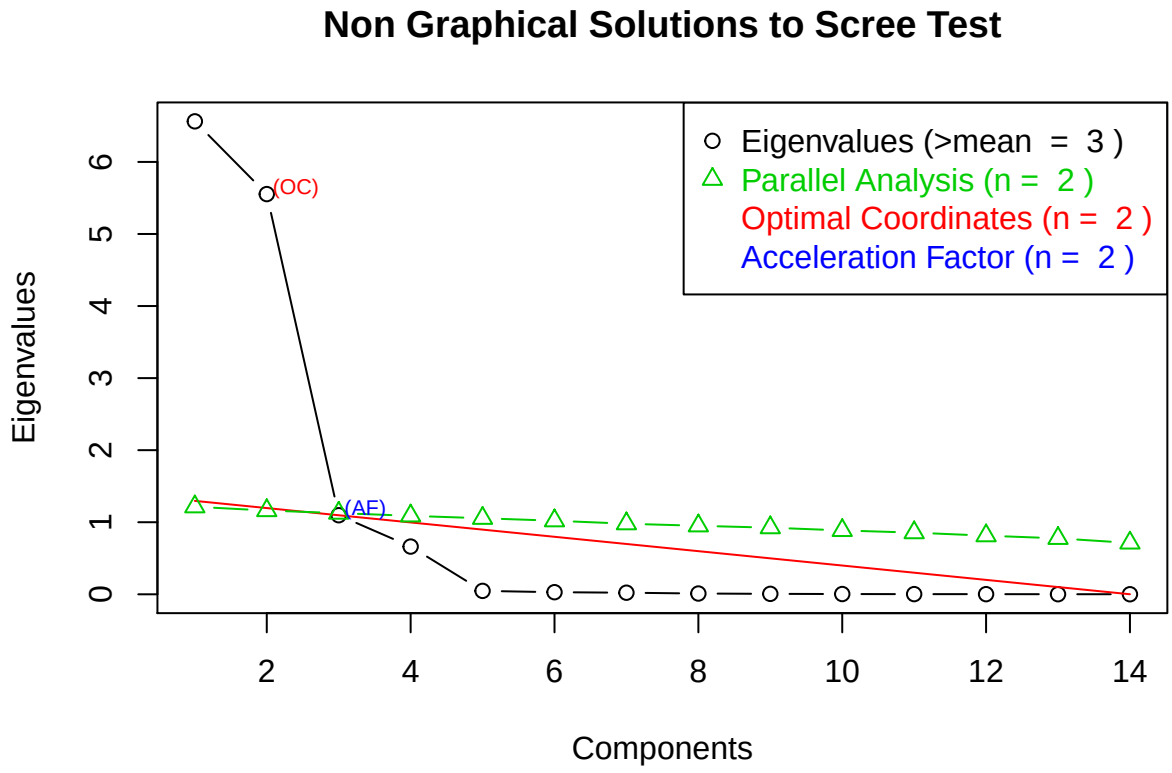
We begin to analyze the data using Principal Component Analysis.




```
##            PC1         PC2          PC3          PC4
## xF   0.2942435 -0.26070636  0.078744995 -0.188899657
## yF  -0.2794330 -0.29479659  0.046130612  0.032777586
## wF  -0.1206052  0.08913905 -0.766422460  0.568148900
## hF   0.1585780 -0.10946051 -0.628050184 -0.709415013
## xRE  0.2738470 -0.29522750  0.022563377  0.144893240
## yRE -0.2831645 -0.29171738  0.001006468 -0.008940596
## xLE  0.2881683 -0.28016465 -0.034533147  0.078343843
## yLE -0.2705769 -0.30511584  0.016749591  0.011283653
## xN   0.2832853 -0.27525736  0.020665047  0.246382359
## yN  -0.2768979 -0.29728337 -0.025268870 -0.046098948
## xRM  0.2788090 -0.28061881 -0.002109732  0.192831770
## yRM -0.2722651 -0.30239743 -0.037238484 -0.043792537
## xLM  0.3280473 -0.21838995 -0.069718427  0.082286981
## yLM -0.2604178 -0.31515751 -0.022807691 -0.033737931
```

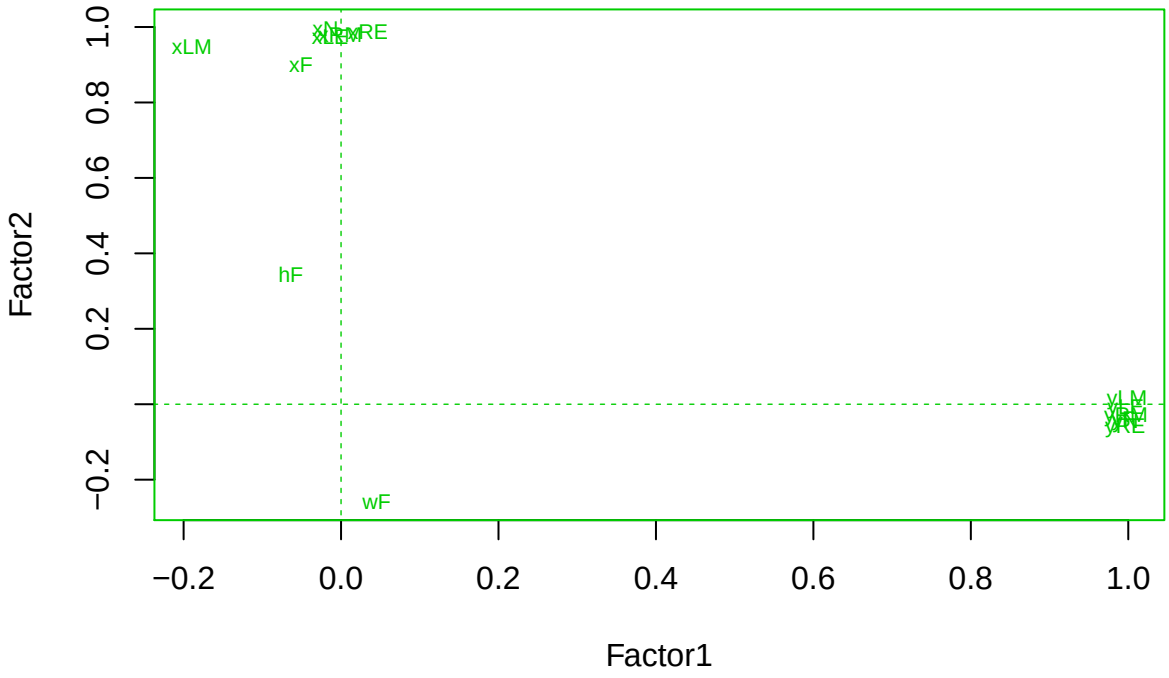
From the above scree plot, we can observe that the elbowing effect begins to happen at 4 components. Beyond 4-5 components, the amount of variance explained by adding additonal components severely decreases. Observe that in our 4 chosen components, the first 2 components loads quite a bit of information of the position of your face and the other two components are loading information of the actual dimensions of your face. This si showing that we can reduce our number of predictors to four principal components containing information on the dimensions and positions of your face.

Now we can take a look at the results of a Factor Analysis.




```
##  [1] "converged"    "loadings"     "uniquenesses" "correlation"  "criteria"
##  [6] "factors"      "dof"          "method"       "rotmat"       "STATISTIC"
## [11] "PVAL"         "n.obs"        "call"
```

```
##
## Loadings:
##     Factor1 Factor2 Factor3 Factor4
## xF           0.900   0.260  -0.338
## yF   0.994
## wF          -0.257   0.106   0.842
## hF           0.345   0.842   0.117
## xRE          0.989
## yRE  0.996
## xLE          0.975   0.157
## yLE  0.997
## xN           0.996
## yN   0.996
## xRM          0.980
## yRM  0.997
## xLM -0.190   0.949   0.188
## yLM  0.998
##
##                Factor1 Factor2 Factor3 Factor4
## SS loadings      6.004   5.787   0.868   0.865
## Proportion Var   0.429   0.413   0.062   0.062
## Cumulative Var   0.429   0.842   0.904   0.966
```



Both the graph and the loading variables shown in the printed table are showing us the same thing. The first factor is loading information for our X position data, factor 2 is loading data for our y position, factor 3 is obviously loading the most information in the hf variable, and factor 4 contains information on the wf information. The factor analysis has literally told us that our set of predictor variables can be reduced to those four factors. Although that information is very useful, we cannot make predictions using our factor analysis. We cannot create our source data that we would need to predict and Factor analysis is purely exploratory.

Index (Code)

```
#Calculate exclusivity

college <- mutate(college,exclusivity = (100*(Apps-Accept)/(Apps) + 100*(Enroll/Accept)))
exclusivity <- college$exclusivity
#Normalize predictors
znorm <- function(x){
  m <- mean(x)
  s <- sd(x)
  (x-m)/s
}

college <- lapply(college %>% dplyr::select(Private,Top10perc,Top25perc,
                          F.Undergrad,P.Undergrad,Outstate,
```

```
                          Room.Board,Books,Personal,
                          PhD,Terminal,S.F.Ratio,perc.alumni,
                          Expend,Grad.Rate),function(col){
                            c <- as.numeric(col)
                            znorm(c)
}) %>% bind_cols() %>% data.frame
college$exclusivity <- exclusivity
#Split into training and test data

train <- sample(1:n,size = floor(.75*n),replace=F)
c.train <- college[train,]
c.test <- college[-train,]


#Attach training data
attach(c.train)
#Linear Model

linear <- lm(exclusivity~Private + Top10perc + Top25perc + F.Undergrad +
               P.Undergrad + Outstate + Room.Board + Books + Personal +
               PhD + Terminal + S.F.Ratio + perc.alumni + Expend + Grad.Rate,data=c.train)
summary(linear)

linear.AIC <- step(linear)
summary(linear.AIC)

#Evaluate the performance of the linear model

p <- predict(linear.AIC,c.test)
linear.mse <- mean((c.test$exclusivity - p)^2)


#Ridge regression
library(MASS)
library(caret)
library(glmnet)

# Predictor variables
x <- model.matrix(exclusivity~Private + Top10perc + Top25perc + F.Undergrad +
                    P.Undergrad + Outstate + Room.Board + Books + Personal +
                    PhD + Terminal + S.F.Ratio + perc.alumni + Expend + Grad.Rate, c.train)[,-1]

# Outcome variable
y <- c.train$exclusivity

#Find CV best lambda for Ridge Regression
#alpha is the blending parameter for lambda.  if alpha = 0, all of lambda is L2 norm (ridge)
#If alpha = 1 then all of lamba is L1 norm (Lasso).  If alpha is anything between then
#you get a portion of lambda for L1 and a portion for L2.

cv <- cv.glmnet(x, y, alpha = 0)
cv$lambda.min
```

```
model <- glmnet(x, y, alpha = 0, lambda = cv$lambda.min)
coef(model)
summary(model)

#Make predictions
x.test <-  model.matrix(exclusivity~Private + Top10perc + Top25perc + F.Undergrad +
                          P.Undergrad + Outstate + Room.Board + Books + Personal +
                          PhD + Terminal + S.F.Ratio + perc.alumni + Expend + Grad.Rate, c.test)[,-1]
p <- predict(model,x.test)
Ridge.mse <- mean((c.test$exclusivity - p)^2)


#Lasso Model
cv <- cv.glmnet(x, y, alpha = 1)
cv$lambda.min

model <- glmnet(x, y, alpha = 1, lambda = cv$lambda.min)
coef(model)
summary(model)

#Make predictions
x.test <-  model.matrix(exclusivity~Private + Top10perc + Top25perc + F.Undergrad +
                          P.Undergrad + Outstate + Room.Board + Books + Personal +
                          PhD + Terminal + S.F.Ratio + perc.alumni + Expend + Grad.Rate, c.test)[,-1]
p <- predict(model,x.test)
Lasso.mse <- mean((c.test$exclusivity - p)^2)


#Elastic Net

elastic <- train(
  exclusivity~Private + Top10perc + Top25perc + F.Undergrad +
    P.Undergrad + Outstate + Room.Board + Books + Personal +
    PhD + Terminal + S.F.Ratio + perc.alumni + Expend + Grad.Rate,
  data = c.train, method = "glmnet",
  trControl = trainControl("cv", number = 10),
  tuneLength = 10
)
# Model coefficients
coef(elastic$finalModel, elastic$bestTune$lambda)
# Make predictions
p <- elastic %>% predict(c.test)
Elastic.mse <- mean((c.test$exclusivity - p)^2)


# PCR
require(pls)

pcr_model = pcr(y~x,ncomp=12)
pcr_summary <-summary(pcr_model)

p <- pcr_model %>% predict(x.test)
pcr.mse <- mean((c.test$exclusivity - p)^2)
```

```r
#PLSR
partial_model = plsr(y~x,validation="CV",ncomp=6)
summary(partial_model)

p <- partial_model %>% predict(x.test)
plsr.mse <- mean((c.test$exclusivity - p)^2)


#Problem 2

drive <- read.csv("C:\\Users\\Jacob\\Documents\\Math 537\\drivPoints.txt")
no <- lapply(drive[,6:ncol(drive)],function(col){
  znorm(col)
}) %>% bind_cols() %>% data.frame
pred <- drive[,5]
x <- data.frame(pred,no)

# Pricipal Components Analysis
# entering raw data and extracting PCs
# from the correlation matrix
fit <- princomp(x, cor=TRUE)
#summary(fit) # print variance accounted for
#loadings(fit) # pc loadings
plot(fit,type="lines") # scree plot
#fit$scores # the principal components
#biplot(fit)
pca <- prcomp(x[,-1])
pca$rotation[,1:4]


# Factor Analysis

#In this dataset the 4,5 columns are response information

#We're first going to do some exploratory factor analysis, then we'll play around with some confirmator

X = drive[,-c(1:5)]

# Determine Number of Factors to Extract
library(nFactors)
ev <- eigen(cor(X)) # get eigenvalues
ap <- parallel(subject=nrow(X),var=ncol(X),
               rep=100,cent=.05)
nS <- nScree(x=ev$values, aparallel=ap$eigen$qevpea)
plotnScree(nS)

fact.model = factanal(X,factors=4,rotation="varimax")
fact.model
names(fact.model)
fact.model$loadings
loadings = fact.model$loadings[,1:3]
plot(loadings,type="n")
text(loadings,labels=names(X),cex=.7)
```

```
lines(c(-2,2),c(0,0),lty=2,lwd=.7)
lines(c(0,0),c(-2,2),lty=2,lwd=.7)
```