STOR 565 Fall 2019 Homework 4

Hunter Finger

Remark. This homework aims to help you further understand the model selection techniques in linear model. Credits for **Theoretical Part** and **Computational Part** are in total 100 pt. For **Computational Part** , please complete your answer in the **RMarkdown** file and summit your printed PDF homework created by it.

Computational Part

1.(*10 pt*) Consider the Nba data posted on the Sakai class site. Create a new data frame that contains the following columns:

- team
  - wins
  - points
  - points3
  - free_throws
  - off_rebounds
  - def_rebounds
  - assists
  - steals
  - personal_fouls

(a) Create box plots of the quantitative features (i.e. all but) teams to see if you should scale the data when performing PCA. Describe your findings in words.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(GGally)
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```
library(tidyverse)
```

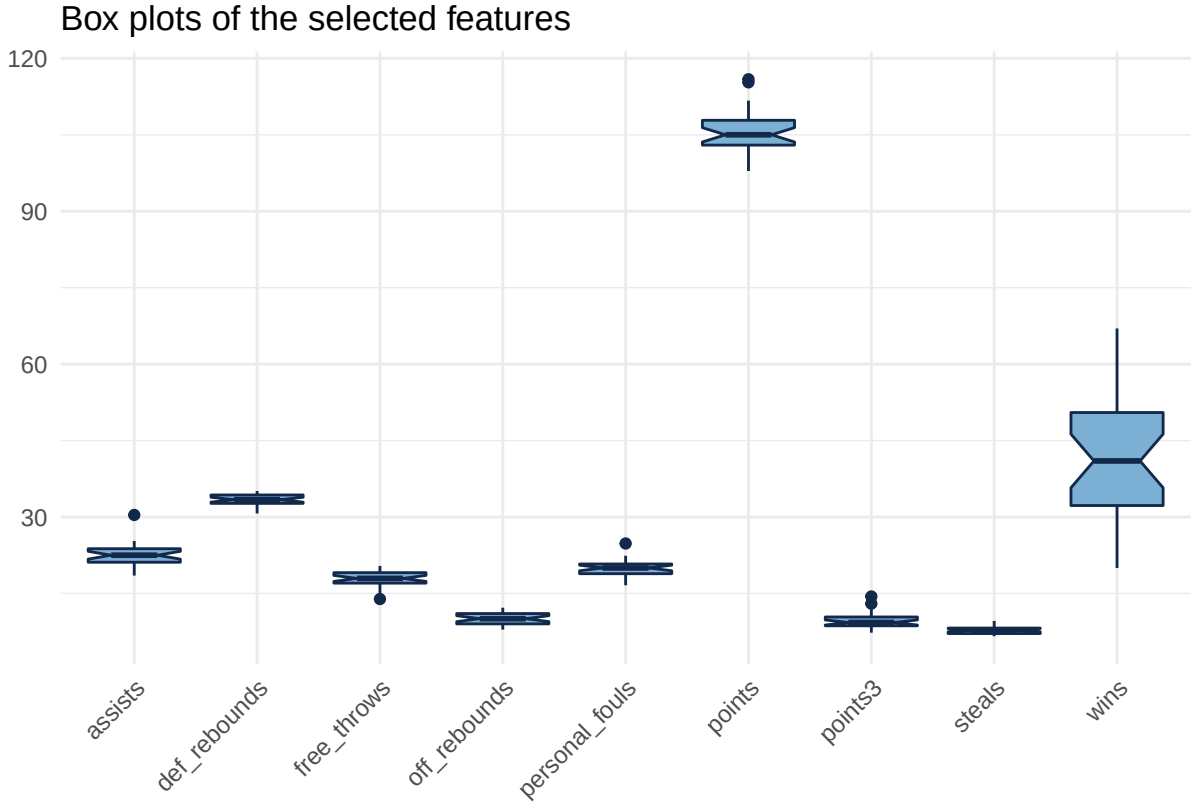
```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v tibble  2.0.0     v purrr   0.2.5
## v tidyr   0.8.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.3.0
```

```
## -- Conflicts ---------------------------------------------------------------------------------------
## x purrr::accumulate() masks foreach::accumulate()
## x tidyr::expand()     masks Matrix::expand()
## x dplyr::filter()     masks stats::filter()
## x dplyr::lag()        masks stats::lag()
## x purrr::when()       masks foreach::when()
```

```
nba = read.csv("nba-teams-2017.csv")
nba1 = nba %>% select(team, wins, points, points3, free_throws, off_rebounds, def_rebounds, assists, st

example_bball_long <- nba1 %>%
  gather(metric, value, -team)
ggplot(example_bball_long, aes(x = metric, y = value)) +
  geom_boxplot(fill = "#7BAFD4", colour = "#13294B", notch = TRUE) +
  labs(title = "Box plots of the selected features",
       x = "",
       y = "") +
  theme_minimal() + theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The data needs to be scaled because of the varying distributions for the variables. This can be seen on the boxplot where some plots can hardly be seen because of the scale of the data.

(b) Obtain PC loadings of the first four princple components (PCs). **Display only the first few elements of each!**

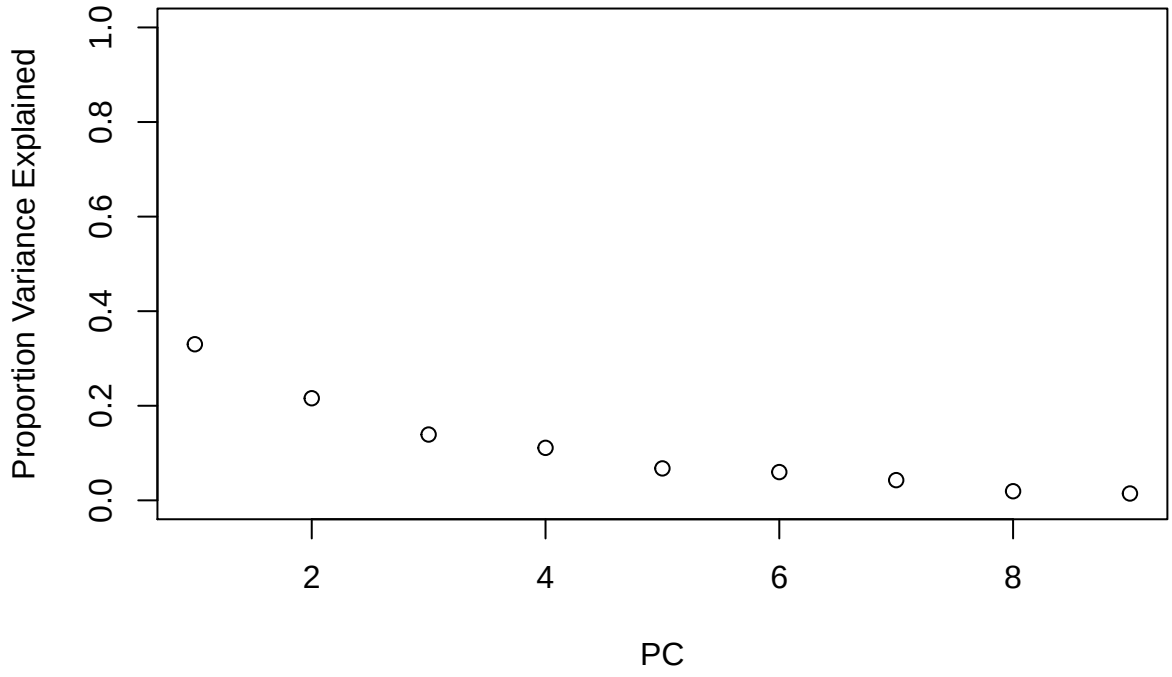
```
teams = nba1$team
standardized_nba = prcomp(nba1 %>%select(-team), scale = T)
standardized_nba$rotation[,1:4]
```

```
##                      PC1         PC2         PC3         PC4
```

```
## wins           -0.42366308  0.07555818 -0.11681772  0.21657745
## points         -0.50246947 -0.21708761  0.19677723 -0.07946391
## points3        -0.41664778  0.17268215 -0.08316881 -0.51204012
## free_throws    -0.24452950 -0.41519726  0.30946852 -0.33272209
## off_rebounds    0.08111297 -0.39160091  0.47926467  0.46463787
## def_rebounds   -0.26053718  0.26461371  0.57662166  0.15459073
## assists        -0.45236958  0.05322237 -0.26482231  0.27220000
## steals         -0.20525546 -0.41895791 -0.45168498  0.37698249
## personal_fouls  0.11583180 -0.58585494 -0.09277492 -0.34335891
```

(c) Plot a scree plot describing the amount explained by the various PCs.

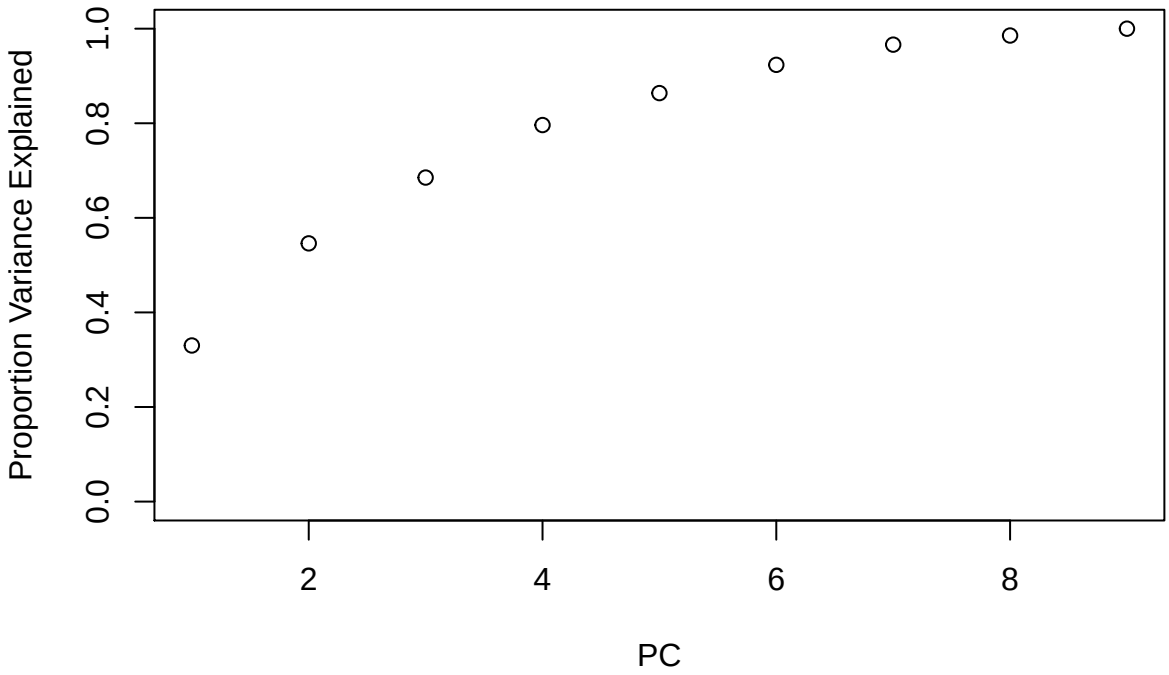
```
nba_variance = standardized_nba$sdev^2
prop_var_expl = nba_variance/sum(nba_variance)
plot(prop_var_expl, xlab = "PC", ylab = "Proportion Variance Explained", ylim = c(0,1) )
```



(d) Make another plot showing the cumulative percent of the variance explained. Precisely: for each $1 \leqslant k \leqslant 10$ you are plotting

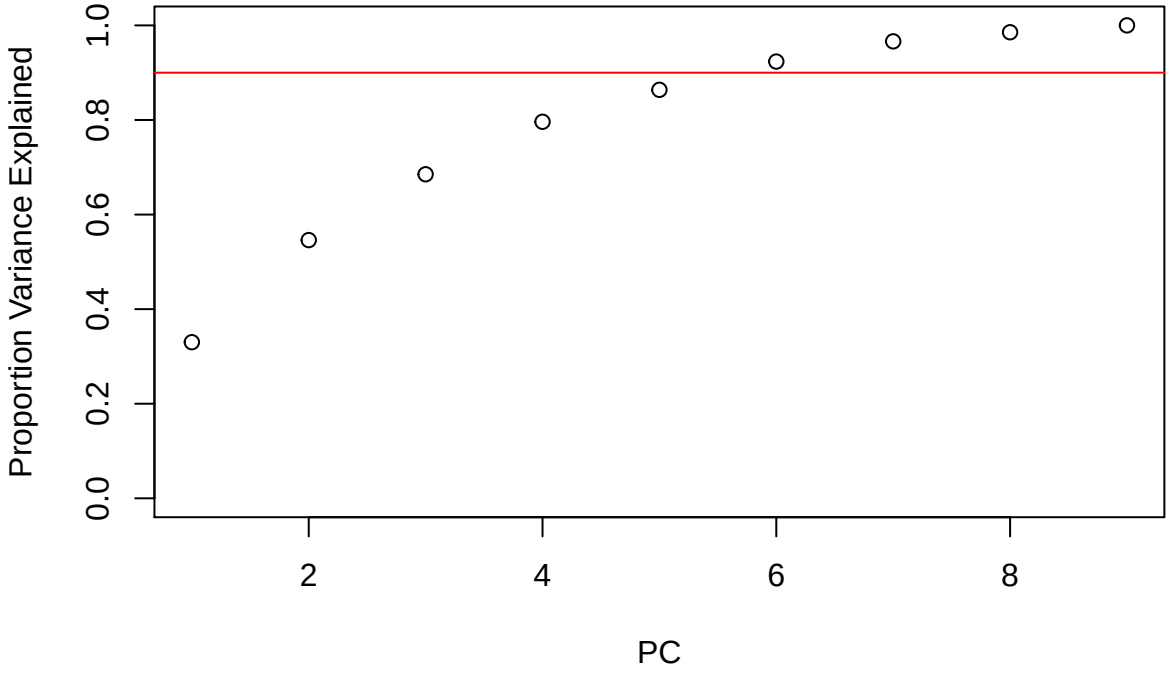
$$\frac{\sum_{j=1}^{k} d_j^2}{\sum_{j=1}^{10} d_j^2}$$

```
plot(cumsum(prop_var_expl), xlab = "PC", ylab = "Proportion Variance Explained", ylim = c(0,1) )
```



(e) If you were to retain all PCs which explain at least 90% of the variance, how many PCs would you retain?

```
plot(cumsum(prop_var_expl), xlab = "PC", ylab = "Proportion Variance Explained", ylim = c(0,1) )
abline(h = .9, col = "red")
```




```
As can be seen in the plot above, 6 PC scores need to be retained in order to have >90% of the variance
```

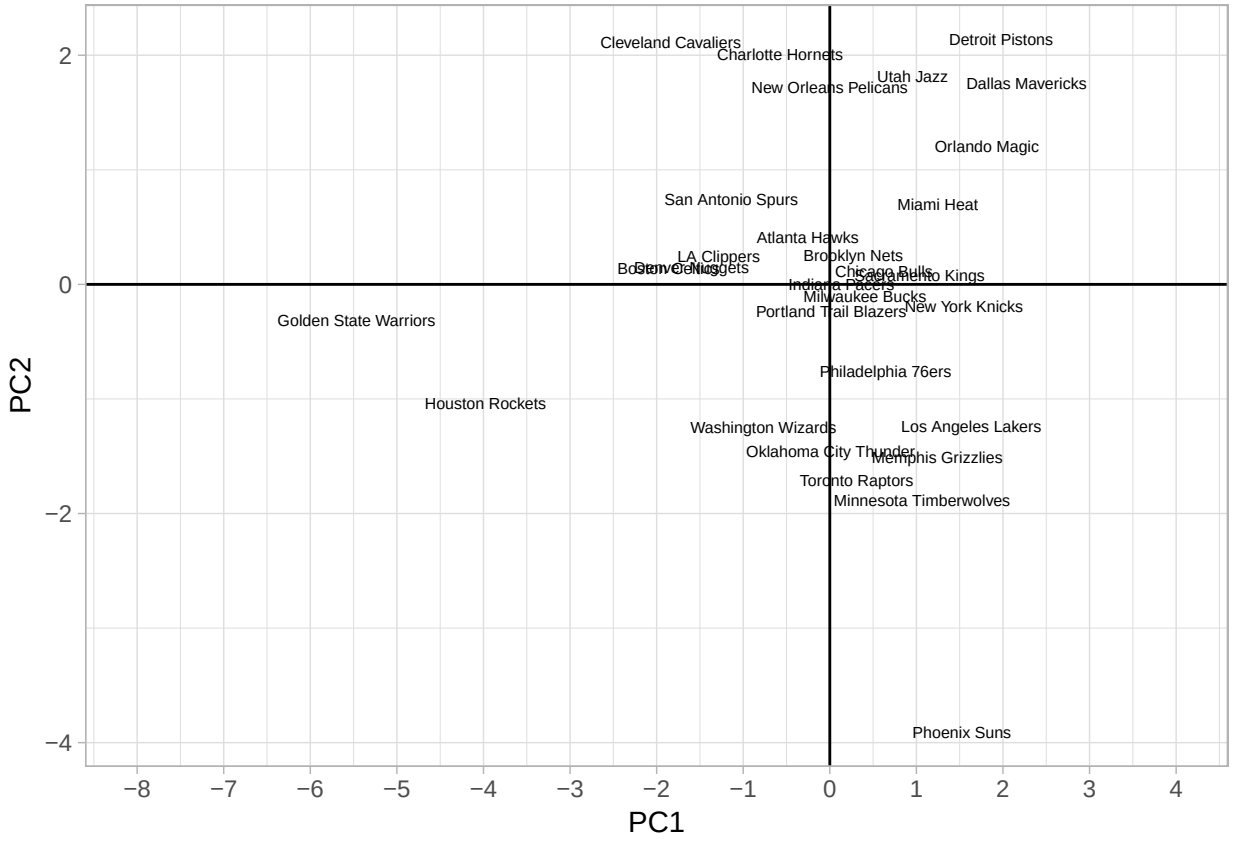
(f) Plot PC1 vs PC2 with the team names and describe your findings.

```
scores = standardized_nba$x
plot_data = scores %>%
  data.frame() %>%
  mutate(team = nba1$team) %>%
```

```
  select(team, everything())
```

```
## Warning: The `printer` argument is deprecated as of rlang 0.3.0.
## This warning is displayed once per session.
```

```
ggplot(plot_data, aes(x = PC1, y = PC2)) +
 geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  geom_text(aes(label = team), size = 2) +
  scale_x_continuous(breaks = -10:10) +
  coord_cartesian(xlim = c(-8, 4)) +
  theme_light()
```



2. (*15 pt*). **Important:** Please see the "Principal_components.rmd" file under the R demonstration folder in Lecture 5 for Sakai to see the code for manipulating figures. Using code like that demonstrated in class, download the .png file containing an image of a house posted to Redfin in the Data folder on Sakai. Note, you may have to download this first and then open it from your own computer. Set X to be the pixel intensity associated with the **red color** in the image using code like that performed in class. See the *Value* section of `?readPNG` to remind yourself of the organization of the raster array output of that function.

Answer the following questions:

(a) What are the dimensions of X? Plot a histogram of the pixel intensities within the image.

```
pic = "Redfin_house.png"
picRGB = readPNG(pic)
picPlot = as.raster(picRGB)
grid.raster(picPlot)
```

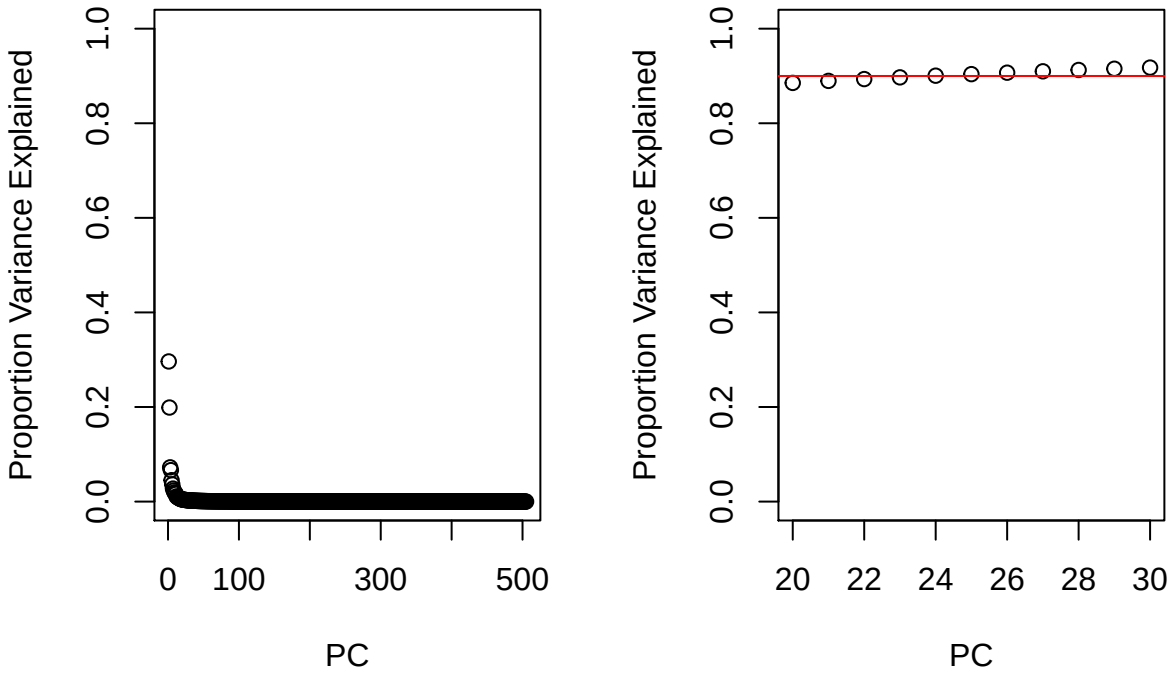

```
X = picRGB[,,1]
pic.pr = prcomp(X, scale =T)
dim(pic.pr$rotation)
```

```
## [1] 798 505
```

(b) Plot the scree plots for this data, which illustrate the percentage variation explained against the number of principal components and the cumulative percentage variation explained against the number of principal components. How many PCs are needed to explain 90% of the total variation of X?

```
pic_variance = pic.pr$sdev^2
prop_var_expl = pic_variance/sum(pic_variance)
par(mfrow = c(1,2))
plot(prop_var_expl, xlab = "PC", ylab = "Proportion Variance Explained", ylim = c(0,1) )

plot(cumsum(prop_var_expl), xlab = "PC", ylab = "Proportion Variance Explained", ylim = c(0,1), xlim = c
abline(h = .9, col = "red")
```



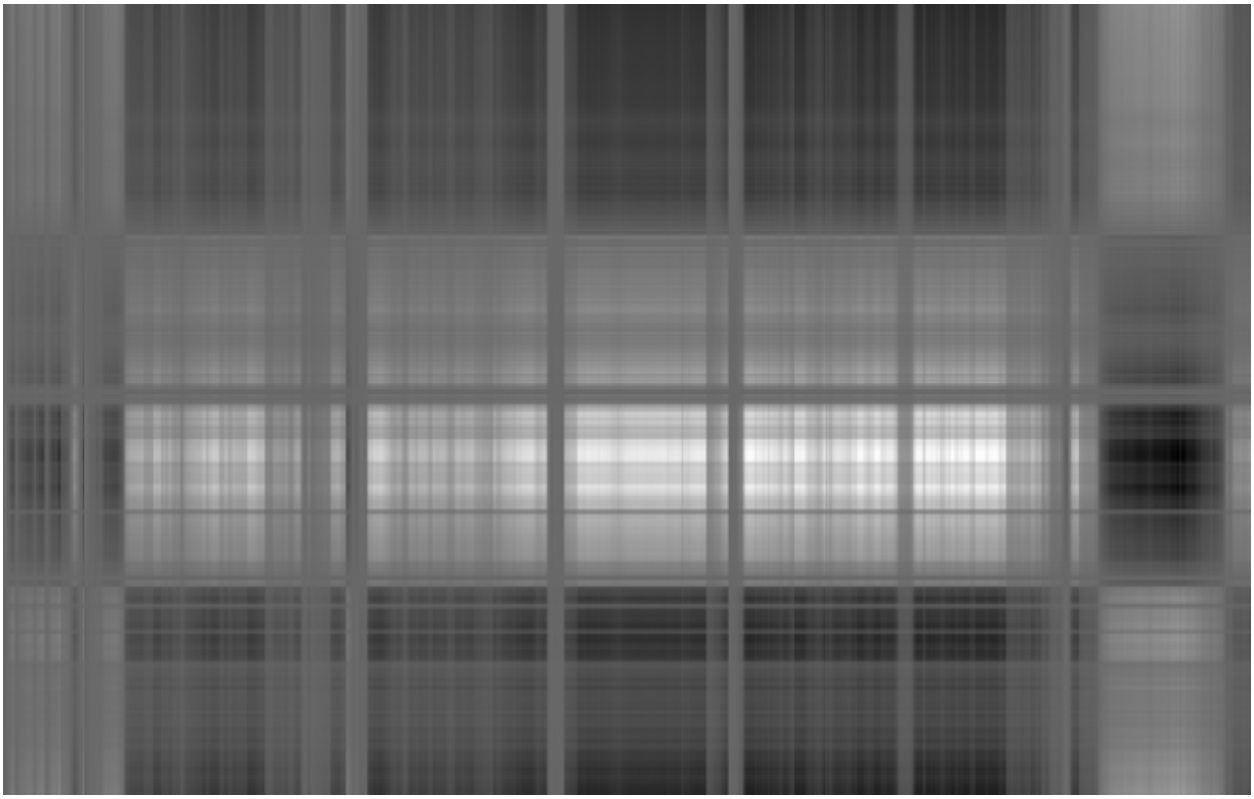

```
As can be seen from the plot, 24 PC scores are needed to account for >= 90% of the data.
```

(c) For $d = 1, 5, 10, 15, 20, 30, 50, 100, 200$ project the image onto the first d principal components and plot the resulting compressed image for each d. For each of the nine plots, include the cumulative percentage variation explained by the projection in the title of your plots.

```
loading = pic.pr$rotation
pc.image = list()
num_pc = c(1, 5, 10, 15, 20, 30, 50, 100, 200)

image = scale(X)
for(i in 1:length(num_pc)){
  u.proj = loading
  u.proj[, (num_pc[i] + 1) : 505] <- 0
  projection <- (image%*%u.proj)%*%t(u.proj)
  scaled <- (projection - min(as.numeric(projection)))
  scaled <- scaled / max(as.numeric(scaled))
  pc.image[[i]] <- as.raster(scaled)
}

grid.raster(pc.image[[1]])
```



```
grid.raster(pc.image[[2]])
```

```
grid.raster(pc.image[[3]])
```

```
grid.raster(pc.image[[4]])
```

```
grid.raster(pc.image[[5]])
```

```
grid.raster(pc.image[[6]])
```

```
grid.raster(pc.image[[7]])
```

```
grid.raster(pc.image[[8]])
```

```
grid.raster(pc.image[[9]])
```

3. (Prediction, Textbook 6.9, *15 pt*) In this exercise, we will predict the number of applications received using the other variables in the `College` data set from `ISLR` package.

(a) Randomly split the data set into two sets, a training and a test set, as evenly as possible. There is an odd number of observations, so make the test set larger.

```
data(College)
set.seed(83)
train = sample(1:dim(College)[1], dim(College)[1]/2)
test = -train
College.train = College[train,]
College.test = College[test,]
```

(b) Fit a linear model using least squares on the training set, and report the test error obtained.

```
linmod = lm(Apps ~ ., data = College.train)
linmod.pred = predict(linmod, College.test)
lm.mspe = mean((linmod.pred-College.test$Apps)^2)
lm.mspe
```

```
## [1] 1275125
```

(c) Fit a ridge regression model on the training set, with λ chosen by 5-fold cross-validation. Report the test error obtained.

```
train_matrix = model.matrix(Apps~., data = College.train)
test_matrix = model.matrix(Apps ~., data = College.test)
grid = 10 ^seq(10,-2, length = 100)
ridge.mod = glmnet(train_matrix, College.train$Apps, alpha = 0, lambda = grid)
```

```
cv.ridge = cv.glmnet(train_matrix, College.train$Apps, alpha = 0,
                     lambda = grid, nfolds = 5)
bestlam = cv.ridge$lambda.min
pred.ridge <- predict(ridge.mod, s = bestlam, newx = test_matrix)
ridge.mspe = mean((pred.ridge - College.test$Apps)^2)
ridge.mspe
```

```
## [1] 1274764
```

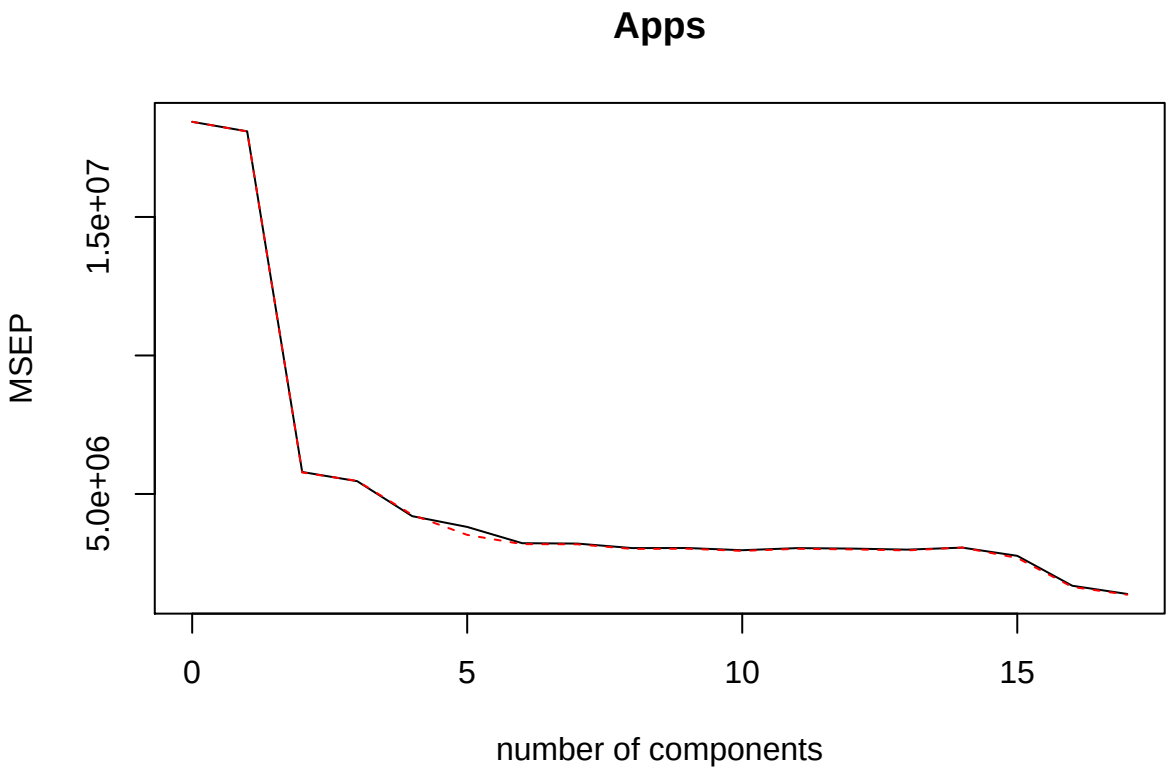
(d) Fit a LASSO model on the training set, with λ chosen by 5-fold cross-validation. Report the test error obtained, along with the number of non-zero coefficient estimates.

```
lasso.mod = glmnet(train_matrix, College.train$Apps, alpha = 1, lambda = grid)
cv.lasso = cv.glmnet(train_matrix, College.train$Apps, alpha = 1,
                     lambda = grid, nfolds = 5)
bestlam = cv.lasso$lambda.min
pred.ridge <- predict(lasso.mod, s = bestlam, newx = test_matrix)
lasso.mspe = mean((pred.ridge - College.test$Apps)^2)
lasso.mspe
```

```
## [1] 1273750
```

(e) Fit a PCR model on the training set, with M chosen by 5-fold cross-validation. Report the test error obtained, along with the value of M selected by cross-validation.

```
pcr.mod <- pcr(Apps ~ ., data = College.train,
               scale = TRUE, validation = "CV")
validationplot(pcr.mod, val.type = "MSEP")
```




```
pred.pcr <- predict(pcr.mod, College.test, ncomp = 9, folds = 5)
pcr.mspe = mean((pred.pcr - College.test$Apps)^2)
pcr.mspe
```

```
## [1] 1955606
```

(f) Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these four approaches?

The MSPE for every model is very close to each other. This shows that all of the models are fairly close to one another in term of accuracy. The lasso model is the most accurate model for prediction followed by the ridge, linear and PCR models in that order. As shown below, all of the R^2 values are around 88% indicating pretty good accuracy.

```
test.avg = mean(College.test$Apps)
1 - lm.mspe/mean((test.avg - College.test$Apps)^2)
```

```
## [1] 0.8894204
```

```
1 - ridge.mspe/mean((test.avg - College.test$Apps)^2)
```

```
## [1] 0.8894518
```

```
1 - lasso.mspe/mean((test.avg - College.test$Apps)^2)
```

```
## [1] 0.8895397
```

```
1 - pcr.mspe/mean((test.avg - College.test$Apps)^2)
```

```
## [1] 0.8304088
```