Assignment3

Group A 20

2022-11-25

Task 1

```
data3 <- read.csv("communities.csv")
set.seed(12345)

data3_full <- data3

data3_full[-ncol(data3_full)] <-
    data3_full %>%
      select( !ViolentCrimesPerPop) %>%
      apply( 2, scale)

data3_X <- data3_full[-ncol(data3_full)]
# no ViolentCrimesPerPop

S <- cov(data3_X)
eigenS <- eigen( S)

sumvar <- sum(eigenS$values)
for (i in 1:length(eigenS$values)) {
  if (sum(eigenS$values[1:i]) >= 0.95 * sumvar) {
    break;
  }
}
i
```

```
## [1] 35
```

```
# i:how many components are needed to obtain at least 95% of variance in the data

eigenS$values[c(1,2)] / sumvar
```

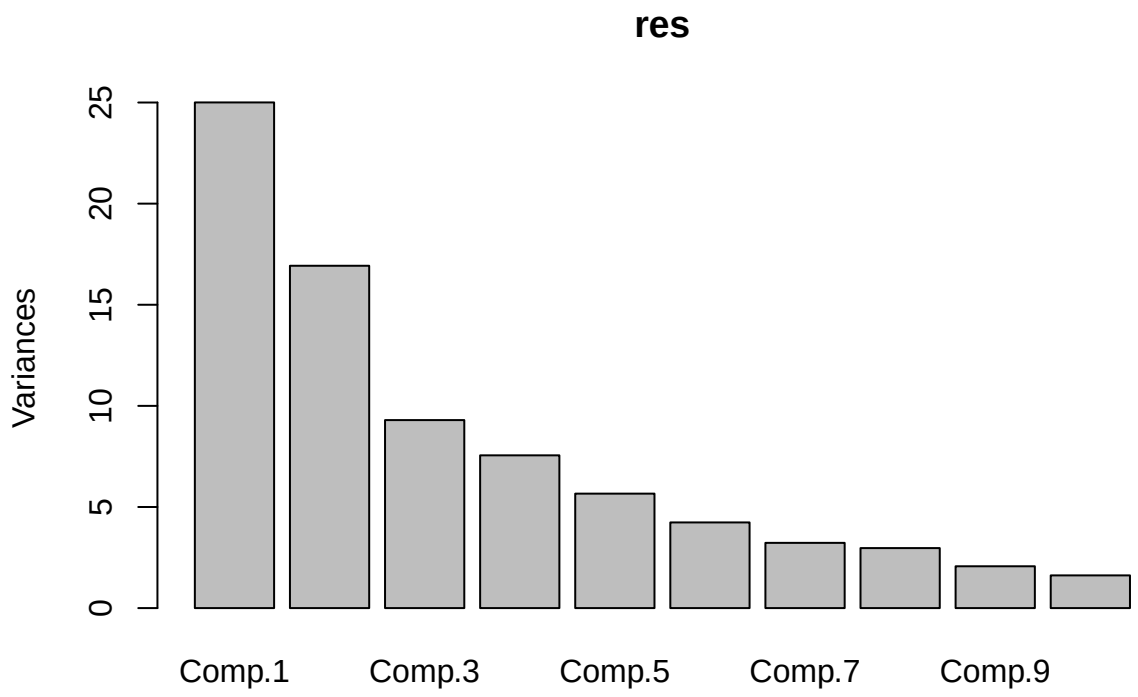
```
## [1] 0.2501699 0.1693597
```

```
# What is the proportion of variation explained by each of the first two principal components
```

- we can see that first 35 components obtain 95%+ of variance,
  and PC1 took 25% variation, PC2 took 17%

Task 2

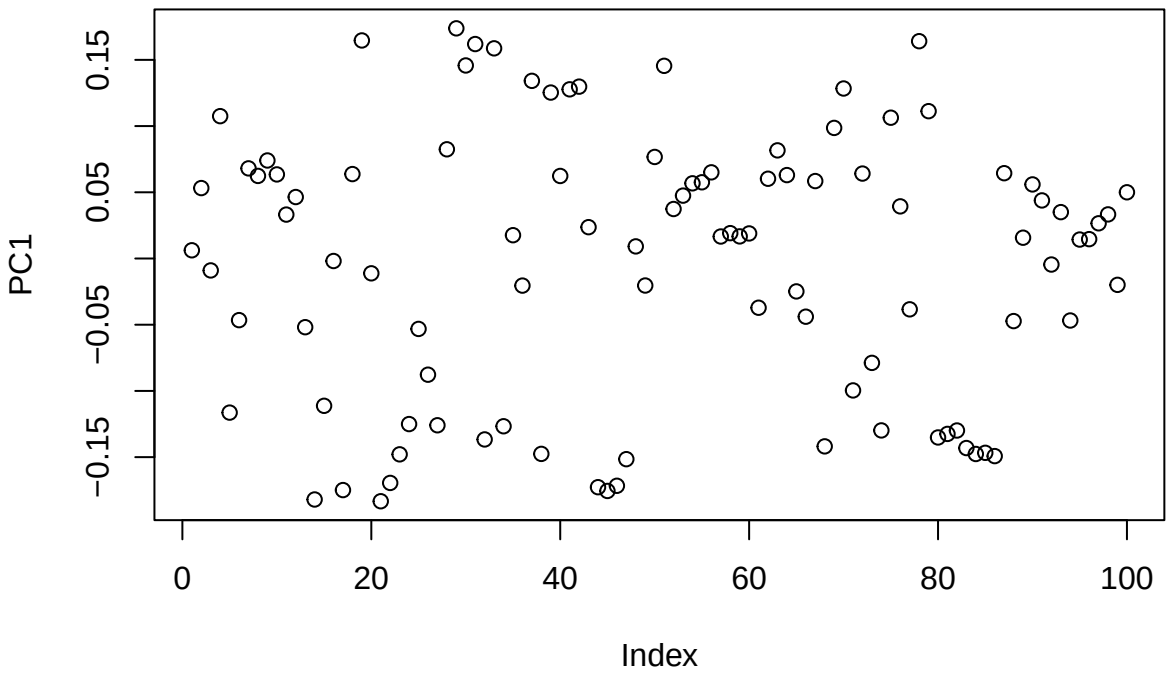
```
res <- princomp(data3_X)
lambda <- res$sdev^2
screeplot(res)
```




```
sort(abs(res$loadings[,1]), decreasing = TRUE)[1:5]
```

```
##      medFamInc      medIncome    PctKids2Par     pctWInvInc PctPopUnderPov
##      0.1833080      0.1819830      0.1755423      0.1748683      0.1737978
```

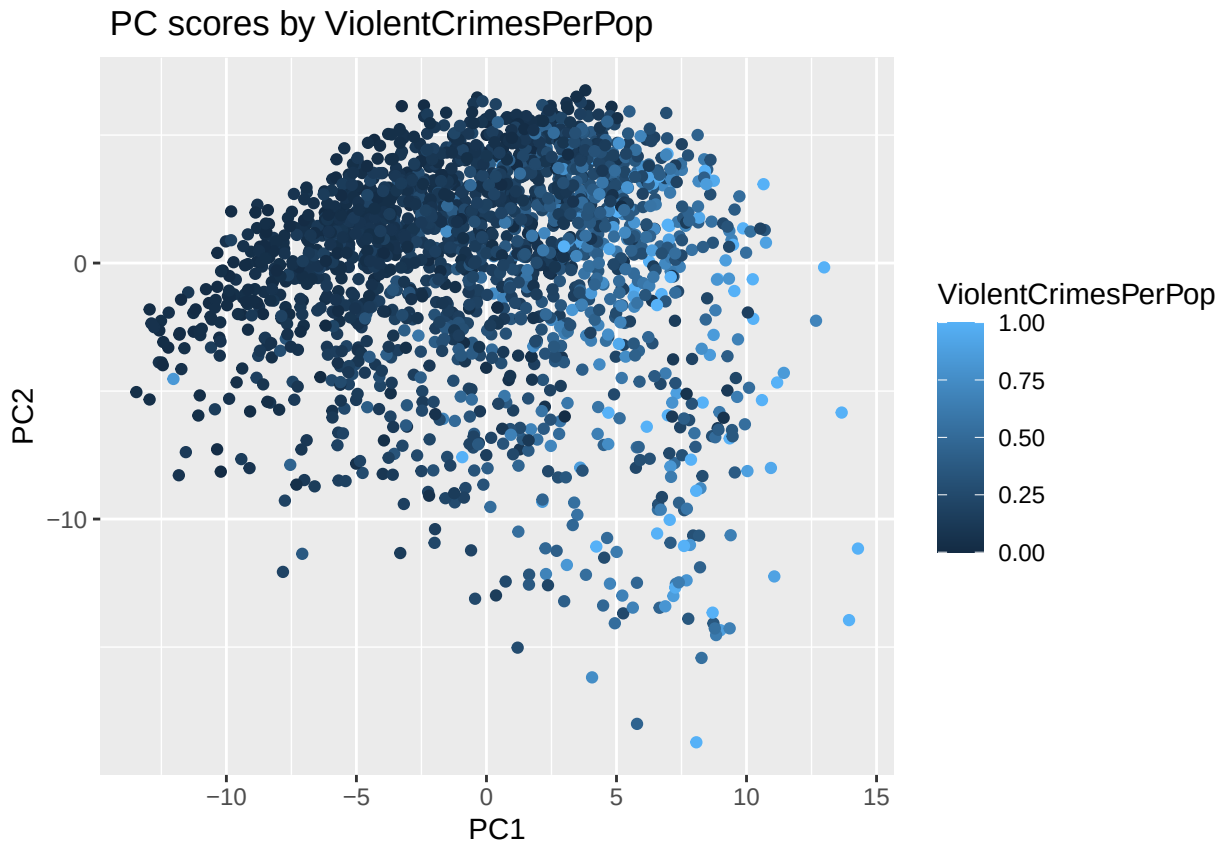
```
plot(eigenS[["vectors"]][,1],ylab = "PC1")
```



- Only 5 features have a variance which is greater than 5 and 2 features over 10 as we can seen in the plot, so we can consider that there isn't much features have a notable contribution. And from the trace plot of PC1, we can see that it's rather evenly distributed and few features contributes more then an absolute value of 0.15.
  5 features contribute mostly:
  medFamInc; medIncome; PctKids2Par; pctWInvInc; PctPopUnderPov
  We can see that all these 5 features highly related to economic level, which may positively of negatively contrubute to the crime level.

```
df_plotPCScores <- data.frame(PC1 = res$scores[,1], PC2 = res$scores[,2], ViolentCrimesPerPop = data3$V

ggplot(df_plotPCScores, aes(x = PC1, y = PC2),)+
  geom_point(aes(colour = ViolentCrimesPerPop))+
  labs(title = " PC scores by ViolentCrimesPerPop")
```



- pop with a higher PC1 score tend to have a higher violent crimes rate. While PC2 doesn't have the same visibly enough contribute to the crimes rate in comparison with PC1

Task 3

```
n <- nrow(data3)
r_train <- sample(1:n , floor(0.5 * n) )
data3_train <- data3[r_train,]
data3_test <- data3[-r_train,]


data3_train_scale <- as.data.frame(scale(data3_train, scale = FALSE))
data3_test_scale <- as.data.frame(scale(data3_test, scale = FALSE))

lm_q3 <- glm(ViolentCrimesPerPop ~.-1 , data = data3_train_scale, family = gaussian)

err_train <- mean(lm_q3$residuals^2)

fit_test <- predict.glm(lm_q3, newdata = data3_test_scale)
err_test <- mean((fit_test - data3_test_scale$ViolentCrimesPerPop)^2)

err_train # train errors
```

```
## [1] 0.01406865
```

```
err_test # test errors
```

```
## [1] 0.02170806
```

- This model has a 0.01906891 test error,and hard to be considered having a good quality

Task 4

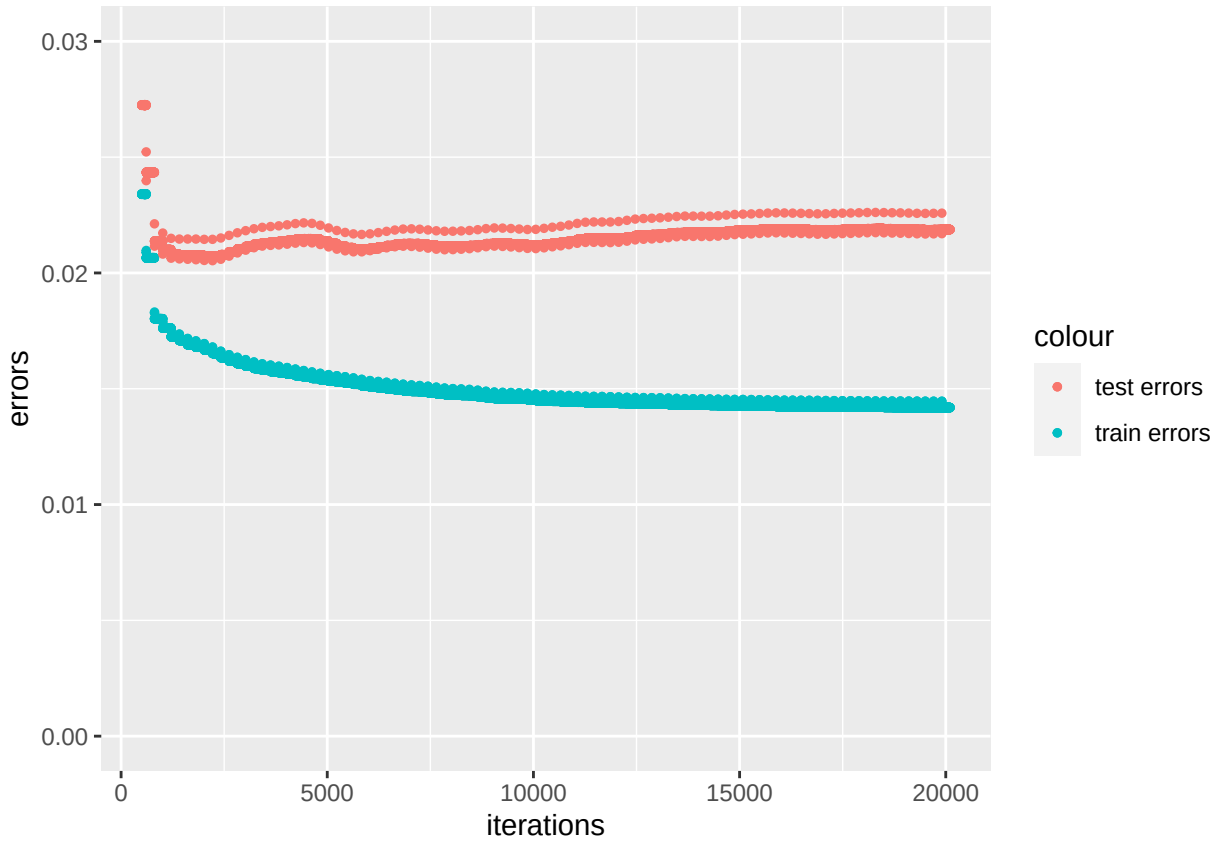
```
err_train_liklihood_optimed <- c()
err_test_liklihood_optimed <- c()

fun_cost <- function(theta, dataIn){
  y_hat <- theta %*% t(as.matrix(dataIn[,1:100]))
  y <- dataIn[,101]
  cost <- sum((y_hat - y)^2) / n


  pred_train_liklihood <- theta %*% t(data3_train_scale[,1:100])
  pred_test_liklihood <- theta %*% t(data3_test_scale[,1:100])
  i <<- i+1
  err_train_liklihood_optimed[i] <<- mean((pred_train_liklihood - data3_train_scale$ViolentCrimesPerPop
  err_test_liklihood_optimed[i] <<- mean((pred_test_liklihood - data3_test_scale$ViolentCrimesPerPop)^2
  return(cost)
}

theta0 <- rep(0,100)
i <- 0
theta_hat_optim <- optim(theta0, fun_cost, dataIn = data3_train_scale, method = "BFGS")

ggplot(data = data.frame(err_train_liklihood_optimed, err_test_liklihood_optimed) )+
  geom_point(aes(x = 1:i, y = err_train_liklihood_optimed, color = 'train errors'),
             size = 1) +
  geom_point(aes(x = 1:i, y = err_test_liklihood_optimed, color = 'test errors'),
             size = 1) +
  xlim(500, i) +
  ylim(0, 0.03) +
  labs(
    main = "errors by iterations",
    x = "iterations",
    y = "errors",
  )
```




```
err_train_liklihood_optimed[7500]
```

```
## [1] 0.01483463
```

```
err_test_liklihood_optimed[7500]
```

```
## [1] 0.02119672
```

- from the plot we can choose iteration 7500 as an optimal choice, and the corresponding error is 0.01721628 for training set and 0.01871308 for testing set, it's slightly better than results in step 3 (0.01906891 test error) but not significant.