Homework 5

2025-10-02

Question 9.1

Using the same crime data set uscrime.txt as in Question 8.2 http://www.statsci.org/data/general/uscrime.txt (file uscrime.txt, description at http://www.statsci.org/data/general/uscrime.html), apply Principal Component Analysis and then create a regression model using the first few principal components. Specify your new model in terms of the original variables (not the principal components), and compare its quality to that of your solution to Question 8.2. You can use the R function `prcomp` for PCA. (Note that to first scale the data, you can include scale. = TRUE to scale as part of the PCA function. Don't forget that, to make a prediction for the new city, you'll need to unscale the coefficients (i.e., do the scaling calculation in reverse)!) Predict the observed crime_stats rate in a city with the following data:

M = 14.0	So = 0	Ed = 10.0
Po1 = 12.0	Po2 = 15.5	LF = 0.640
M.F = 94.0	Pop = 150	NW = 1.1
U1 = 0.120	U2 = 3.6	Wealth = 3200
Ineq = 20.1	Prob = 0.04	Time = 39.0

Show your model (factors used and their coefficients), the software output, and the quality of fit.

Note that because there are only 47 data points and 15 predictors, you'll probably notice some overfitting. We'll see ways of dealing with this sort of problem later in the course.

First, we load the necessary libraries:

```
library(printr)     # pretty print for Rmd
library(lubridate)  # dates
library(ggplot2)    # plots
library(dplyr)      # dataframes
library(tidyr)
library(tidyverse)
library(recipes)
library(caret)
library(MASS)      # step models

# set seed for reproducibility
set.seed(42)
```

Now let's load the data:

```
df <- read.table("./data 8.2/uscrime.txt", header =TRUE)
```

Note: As was stated in the question, our data set is extremely limited, therefore I will do 10-fold cross validation on the final model, but am not going to be creating a final test set.

Comparison to Homework 5

From Homework 5, my best model on all the training data had an adjusted R^2 of 0.8514 and in 10-fold cross validation had R^2 of 0.7953447, so those are the baselines I will be evaluating against. I am going to add the same features as I used previously.

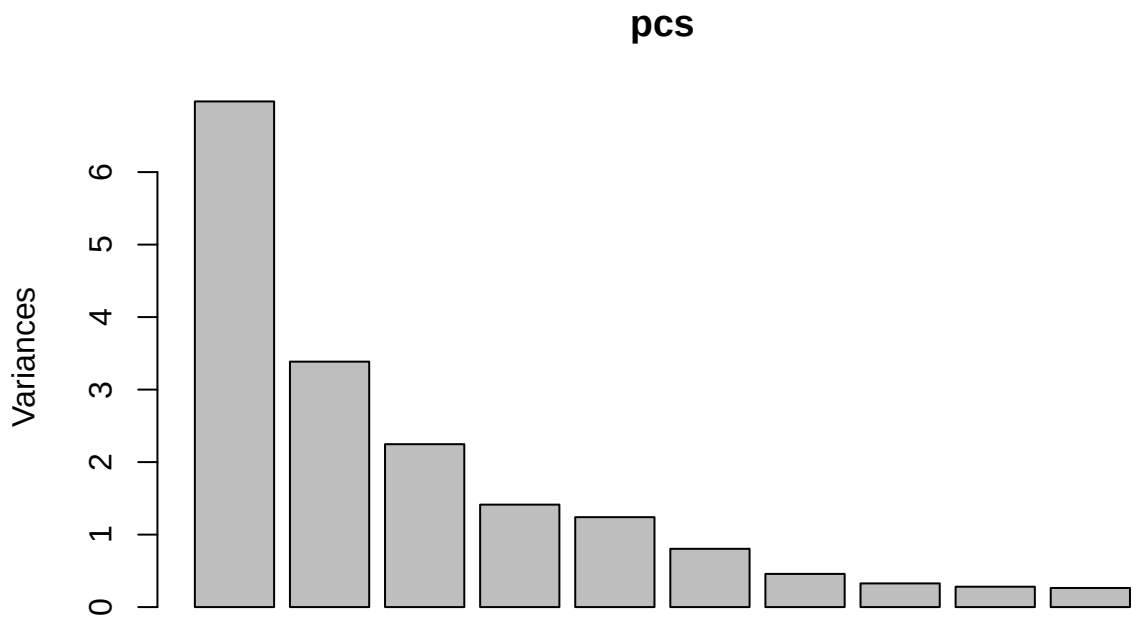
```
crime_stats <- df
crime_stats$LF2 = crime_stats$LF^2
crime_stats$NW2 = crime_stats$NW^2
crime_stats$InvProb = 1 / crime_stats$Prob
```

PCA

Now that we have the same set of features, let's do PCA

```
pcs <- prcomp( subset(crime_stats, select=-c(Crime)) , scale = TRUE)
Xp <- as.data.frame(pcs$x)
Xp$Crime <- df$Crime

plot(pcs)
```



By the sixth principal component a large amount of our variance has been removed. Let's train a model on those and see what our prediction looks like.

```
lm_pc1_to_6 <- lm(Crime ~ PC1 + PC2 + PC3 + PC4 + PC5 + PC6, data=Xp)
summary(lm_pc1_to_6)
```

```
##
## Call:
## lm(formula = Crime ~ PC1 + PC2 + PC3 + PC4 + PC5 + PC6, data = Xp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -450.71 -152.69   15.45  187.05  471.98
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   905.09      38.37  23.589  < 2e-16 ***
## PC1            57.88      14.69   3.941 0.000317 ***
## PC2           -62.55      21.08  -2.967 0.005049 **
## PC3            47.61      25.87   1.840 0.073194 .
## PC4            71.44      32.63   2.190 0.034435 *
## PC5           152.70      34.82   4.385 8.19e-05 ***
## PC6          -120.05      43.26  -2.775 0.008351 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 263 on 40 degrees of freedom
## Multiple R-squared:  0.5978, Adjusted R-squared:  0.5374
## F-statistic: 9.908 on 6 and 40 DF,  p-value: 1.081e-06
```

```
lm_pc1_to_6_log <- lm(log(Crime) ~ PC1 + PC2 + PC3 + PC4 + PC5 + PC6, data=Xp)
print(
  paste(
    "ln(Crime) model adjusted R^2:", summary(lm_pc1_to_6_log)$adj.r.squared
  )
)
```

```
## [1] "ln(Crime) model adjusted R^2: 0.486372646189838"
```

Both the model against linear and log crime stats did significantly worse! Although we can see that the terms all show some significance showing that we are getting information out of the model.

Step AIC

Rather than relying on the transformed coefficients to select features, I am going to use them to concentrate information and use Stepwise model training to produce a parsimonious model.

Note: I also tested the linear model for this and the log version performed better, so I am not going to continue to run them both.

```
constant_model <- lm(log(Crime) ~ 1, data = Xp)
full_model <- lm(log(Crime) ~ ., data = Xp)

# pages of output
invisible(
  capture.output(
    step_model <- stepAIC(
      constant_model,
      scope = list(lower = constant_model, upper = full_model),
      direction = "both"
    )
  )
)

summary(step_model)
```

```
##
## Call:
## lm(formula = log(Crime) ~ PC5 + PC1 + PC7 + PC2 + PC16 + PC6 +
##     PC12 + PC4 + PC15 + PC3 + PC11 + PC13 + PC18 + PC8, data = Xp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.27435 -0.08691 -0.00765  0.08030  0.37447
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.724936   0.021243 316.577  < 2e-16 ***
## PC5          0.150447   0.019280   7.803 6.70e-09 ***
## PC1          0.055534   0.008131   6.830 1.00e-07 ***
## PC7         -0.194191   0.031737  -6.119 7.70e-07 ***
## PC2         -0.068439   0.011670  -5.864 1.61e-06 ***
## PC16        -0.856542   0.161595  -5.301 8.29e-06 ***
## PC6         -0.122778   0.023950  -5.126 1.38e-05 ***
## PC12        -0.240617   0.054361  -4.426 0.000104 ***
## PC4          0.079692   0.018063   4.412 0.000109 ***
## PC15         0.394579   0.095700   4.123 0.000248 ***
## PC3          0.055700   0.014325   3.888 0.000479 ***
## PC11         0.159681   0.048857   3.268 0.002587 **
## PC13        -0.199239   0.064044  -3.111 0.003905 **
## PC18        -2.682166   1.079731  -2.484 0.018412 *
## PC8         -0.065335   0.037576  -1.739 0.091695 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1456 on 32 degrees of freedom
## Multiple R-squared:  0.9127, Adjusted R-squared:  0.8745
## F-statistic: 23.89 on 14 and 32 DF,  p-value: 5.095e-13
```

Interestingly all the terms are significant (unlike my model from Homework 5), and the adjusted R^2 is a bit higher! Looking at the magnitudes of the coefficients PC1, PC3, and PC8 I would have expected them to be relatively unimportant; however, removing them caused a large degredation in quality of fit.

Cross-validation

Now before transforming our coefficients back to true space. let's do cross validation on the PC model.

```
train_control <- trainControl(method = "cv", number = 10)
cross_val_stats <- train(
  formula(step_model),
  data = Xp,
  method = "lm",
  trControl = train_control
)

cross_val_stats
```

```
## Linear Regression
##
## 47 samples
## 14 predictors
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 42, 42, 42, 43, 42, 44, ...
## Resampling results:
##
##   RMSE       Rsquared  MAE
##   0.1945226  0.818317  0.1652626
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

I did not know that R^2 could be higher while MAE and RMSE were also higher. I am not finding a clear explanation of this and will get clarification. However, some of the resources I am seeing state that the RMSE and MAE are more reliable metrics suggesting that the model is worse.

Datapoint prediction

Let's see what the predicted crime_stats for the specified parameters is

```
crime_test <- data.frame(
  M      = 14.0, So     = 0, Ed     = 10.0, Po1    = 12.0,
  Po2    = 15.5, LF     = 0.640, M.F    = 94.0, Pop    = 150,
  NW     = 1.1, U1     = 0.120, U2     = 3.6, Wealth = 3200,
  Ineq   = 20.1, Prob   = 0.04, Time   = 39.0
) %>% mutate(
  InvProb = 1 / Prob,
  LF2 = LF^2,
  NW2 = NW^2
)

crime_test_scaled <- as.data.frame(predict(pcs, crime_test))

# reverse the log scaling because for some reason that's not done automatically
exp(predict(step_model, crime_test_scaled))
```

```
##        1
## 614.3736
```

This is significantly higher than the result from Homework 5 (490.4742). I am very curious which model was better and wish we had a large enough dataset to split out a test set.

Conversion to original space

Finally, let's transform our coefficients back from PC space to our original space. The full conversion (including scaling and centering) is[1]. However, we need to limit to just the coefficients in our model.

```
pc_space_intercept <- step_model$coefficients[1]
coefficients <- step_model$coefficients[-1] # exclude intercept
components <- names(coefficients)

scaled_coefficients <- ((1/pcs$scale) * t(pcs$rotation[,components] %*% coefficients))
```

We also need to account for centering, I am personally thinking of this as converting from the point-slope form of a line to the point-intercept form. That means that to get our intercept in the original space we add the intercept in PC space to the scaled coefficients times the negative centers

```
point_contribution_to_intercept <- -sum(scaled_coefficients * pcs$center)
original_space_intercept <- pc_space_intercept + point_contribution_to_intercept
```

Therefore our equation is

```
eq <- paste0(
  "log(Crime) = ",
  signif(original_space_intercept, digits=4),
  " + ",
  paste0(
    signif(scaled_coefficients, digits=4),
    " * ",
    colnames(scaled_coefficients), collapse = " + "
  )
)

eq <- gsub("\\+ -", "- ", eq)

cat(strwrap(eq, width = 80), sep = "\n")
```

log(Crime) = -11.82 + 0.1015 * M + 0.02764 * So + 0.2135 * Ed + 0.1217 * Po1 - 0.03115 * Po2 + 48.95 * LF - 0.03443 * M.F - 0.002515 * Pop + 0.05369 * NW + 4.714 * U1 + 0.06541 * U2 + 9.418e-05 * Wealth + 0.08 * Ineq - 1.082 * Prob - 0.005965 * Time - 40.69 * LF2 - 0.001285 * NW2 + 0.006471 * InvProb

[1] https://stackoverflow.com/a/29784476/1543042

Comparison and conclusions

Below I show both the coefficients from the PCA model as well as the coefficients from an equivilent procedure from Homework 5 (although I did not display unscalled coefficients and so needed to generate a like comparison). As would be expected given the model has predictive value, the coefficients of the common terms are within a factor of 2; although I am actually surprised they diverged as much as they do. I would suspect that given how poorly the raw PCA model performed and that StepAIC pulled in down to PC18, that this problem is not actually well suited for PCA and the StepAIC was attempting to approximate the non-PCA version.

Variable	PCA	HW5	PCA/HW5
So	2.764E-02	-1.501E-01	-0.18
NW	5.369E-02	6.802E-02	0.79
Wealth	9.418E-05	1.169E-04	0.81
Ed	2.135E-01	2.594E-01	0.82
NW2	-1.285E-03	-1.551E-03	0.83
U2	6.541E-02	7.853E-02	0.83
Ineq	8.000E-02	8.282E-02	0.97
InvProb	6.471E-03	6.489E-03	1.00
M	1.015E-01	7.679E-02	1.32
Pop	-2.515E-03	-1.833E-03	1.37
Po1	1.217E-01	7.009E-02	1.74
Po2	-3.115E-02	-	-
LF	4.895E+01	-	-
M.F	-3.443E-02	-	-
U1	4.714E+00	-	-
Prob	-1.082E+00	-	-
Time	-5.965E-03	-	-
LF2	-4.069E+01	-	-