Homework 6: ISYE 6501 - Introduction to Analytics Modeling

Question 9.1

Prompt

Using the same crime data set `uscrime.txt` as in Question 8.2, 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)!)

Solution

First before utilizing the data set, the description should be recognized to identify exactly what is being worked with in the problem. The data set for `uscrime.txt` uses aggregate data on 47 states of the USA for 1960. The data set provided by [1] contains the following colunns:

- `M`: Percentage of males aged 14–24 in total state population
- `So`: Indicator variable for a southern state
- `Ed`: Mean years of schooling of the population aged 25 years or over
- `Po1`: Per capita expenditure on police protection in 1960
- `Po2`: Per capita expenditure on police protection in 1959
- `LF`: Labour force participation rate of civilian urban males in the age-group 14-24
- `M.F`: Number of males per 100 females
- `Pop`: State population in 1960 in hundred thousands
- `NW`: Percentage of nonwhites in the population
- `U1`: Unemployment rate of urban males 14–24
- `U2`: Unemployment rate of urban males 35–39
- `Wealth`: Wealth: median value of transferable assets or family income
- `Ineq`: Income inequality: percentage of families earning below half the median income
- `Prob`: Probability of imprisonment: ratio of number of commitments to number of offenses
- `Time`: Average time in months served by offenders in state prisons before their first release
- `Crime`: Crime rate: number of offenses per 100,000 population in 1960

As described in the prompt, a useful R function for Principal Component Analysis (PCA) to use is `prcomp()`. To compare the new model to the solution for Question 8.2 on the previous homework, the R packages `lm` and `glm` can be utilized. These packages are contained in the `stats` package, which are part of base R and loaded by default.

Loading US Crime Data

Initially to approach the problem, I set my current working directory to `HW6` and load the `uscrime.txt` data set into a table. Using `head()` I display the data to determine if the import was successful and observe the data set. Observing the data, there are 16 columns (variables) that contain 47 rows of data (for all 47 states in 1960) that are described in the list of variables stated above.

```
# Import libraries
# lm and glm are in the stats package loaded by default
library(ggplot2) # Plot functions

# Set the working directory
setwd("~/projects/ISYE6501/HW6")

# Load the crime data into a table
data <- read.table("data/uscrime.txt", stringsAsFactors = FALSE, header = TRUE)

# View the imported data to check import but only first few rows
cat(paste("\nUS Crime Data:\n"))
```

```
##
## US Crime Data:
```

```
head(data, 5)
```

```
##      M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq     Prob
## 1 15.1  1  9.1  5.8  5.6 0.510  95.0  33 30.1 0.108 4.1   3940 26.1 0.084602
## 2 14.3  0 11.3 10.3  9.5 0.583 101.2  13 10.2 0.096 3.6   5570 19.4 0.029599
## 3 14.2  1  8.9  4.5  4.4 0.533  96.9  18 21.9 0.094 3.3   3180 25.0 0.083401
## 4 13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7 0.015801
## 5 14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4 0.041399
##      Time Crime
## 1 26.2011   791
## 2 25.2999  1635
## 3 24.3006   578
## 4 29.9012  1969
## 5 21.2998  1234
```

Principle Component Analysis with Scaling

Principal component analysis (PCA) is an unsupervised approach that refers to the process by which principal components are computed, and the subsequent use of these components in understanding the data. PCA involves a set of features and no associated response. PCA is a popular approach for deriving a low-dimensional set of features from a large set of variables [2].

`prcomp` performs a principal components analysis, where `center` is a logical value indicated whether the variables should be shifted to zero centered [3]. Discussed further below, the variables *are shifted to zero centered* which is useful to determine the total variance in the data set.

The code provided below utilizes the US crime data set. The 15 predictors are separated from the response variable `Crime` by defining the variable, `predictors`. The `prcomp` model is then utilized with the predictors data, shifted to zero centered, and with scaling enabled. A summary is returned to obtain an overview and results of the model.

```
# Define the PCA data as all 15 predictors
# Exclude the response variable 'Crime'
predictors <- data[, -ncol(data)]

# Apply PCA with scaling set to TRUE
pca_model_scaled <- prcomp(predictors, center = TRUE, scale. = TRUE)

# Summary of PCA results
cat(paste("\nPCA Model (Scaled) Summary:\n"))
```

```
##
## PCA Model (Scaled) Summary:
```

```
summary(pca_model_scaled)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.4534 1.6739 1.4160 1.07806 0.97893 0.74377 0.56729
## Proportion of Variance 0.4013 0.1868 0.1337 0.07748 0.06389 0.03688 0.02145
## Cumulative Proportion  0.4013 0.5880 0.7217 0.79920 0.86308 0.89996 0.92142
##                            PC8     PC9    PC10    PC11    PC12    PC13   PC14
## Standard deviation     0.55444 0.48493 0.44708 0.41915 0.35804 0.26333 0.2418
## Proportion of Variance 0.02049 0.01568 0.01333 0.01171 0.00855 0.00462 0.0039
## Cumulative Proportion  0.94191 0.95759 0.97091 0.98263 0.99117 0.99579 0.9997
##                           PC15
## Standard deviation     0.06793
## Proportion of Variance 0.00031
## Cumulative Proportion  1.00000
```

The summary returns two important variables. The first, `sdev` is standard deviations. The second return that is important is `rotation` which is the matrix of variable loadings (matrix w/ columns containing eigenvectors). Standard deviations denoted as `sdev` are the square roots of the eigenvalues of the covariance (correlation) matrix. To obtain the eigenvalues of the principle components, `sdev` is squared. The eigenvalues represent the amount of each variant of each principle component.

Determining Variance

To calculate the total variance, take the sum of the standard deviations squared. Assuming the variables are centered to mean zero, which was defined in the PCA model above by `center = TRUE`, the total variance in a data set is defined as [2]:

$$\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2 \tag{1}$$

In PCA, the total variance is preserved so the total sum is the the sum of variances of all variables. The proportion of total variance for each principal component needs to be determined, which is quantified by [2]:

$$\frac{\sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2} \tag{2}$$

The proportional variance printed above in the model summary can be quantified using Eq. (2). By taking the squared value of the standard deviations divided by the sum of the squared value of the standard devations, the proportional variance is defined as the variable `variance`. The proportional variance is stored in a dataframe that includes the number of principal components (`principal_components`) and the calculated proportional variance for each component (`prop_variance`). The variance data is printed to display that the calculated proportional variance values are identical to those displayed in `summary(pca_model_scaled)`.

```r
# Use sdev (std. deviations) of principal components
variance = pca_model_scaled$sdev^2 / sum(pca_model_scaled$sdev^2)

# Create a dataframe for plotting
variance_data <- data.frame(
  # Number of principal components
  principal_components = 1:length(variance),
  # Proportional Variance
  prop_variance = round(variance, 4)
)

head(variance_data, 5)
```

```
##   principal_components prop_variance
## 1                    1        0.4013
## 2                    2        0.1868
## 3                    3        0.1337
## 4                    4        0.0775
## 5                    5        0.0639
```

The calculation creates a vector of values that represent the fraction of total variance for each component between 0 and 1. The sum of all values is equivalent to 1.

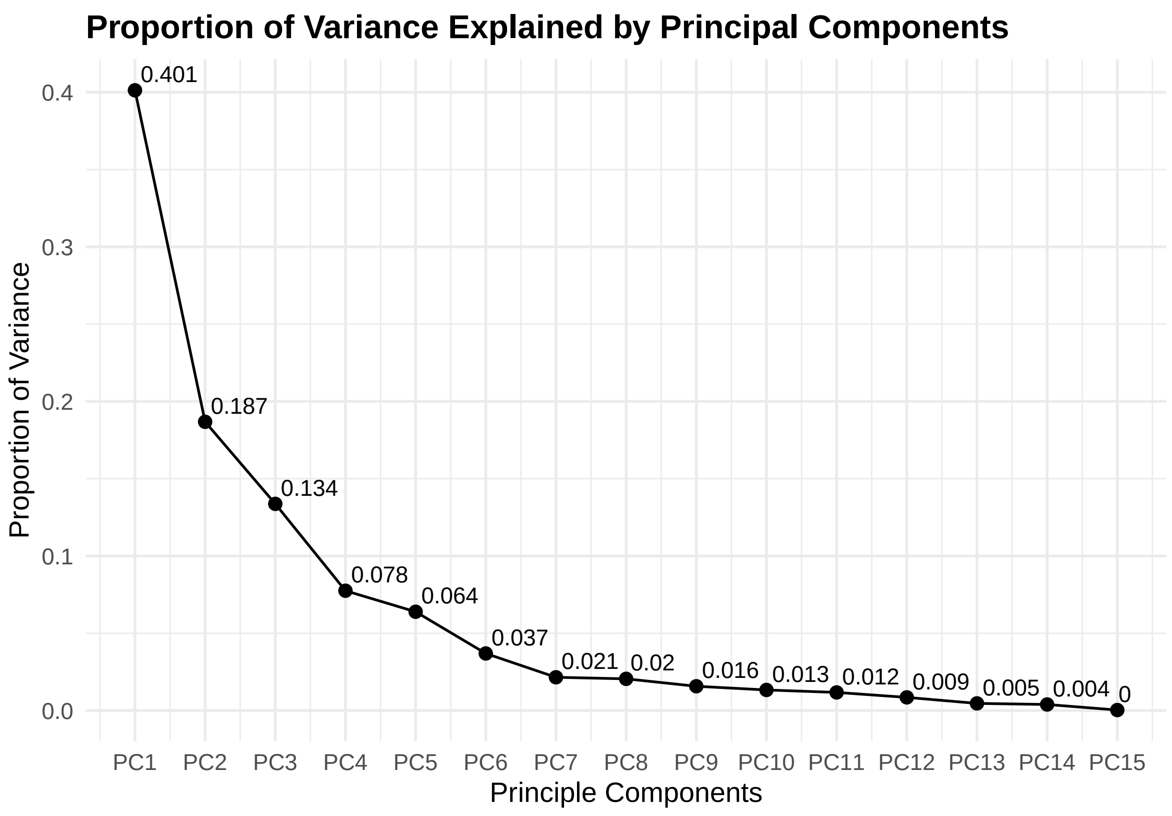
Observing the proportional variance calculated, it is analogous to the "Proportion of Variance" values provided in the PCA model summary.

Visualizing Proportion of Total Variance

Utilizing the proportional variance, a scree plot can be developed to visualize the proportion of total variance explained by each principal component. To create a scree plot, the dataframe created earlier is utilized that includes the necessary data for plotting with the R package `ggplot2`[4]. The scree plot is generated with the integer values for the component number along with the proportional variance. The scree plot can be generated by utilizing the following code:

```r
# Create the scree plot with ggplot2
scree <- ggplot(variance_data, aes(x = principal_components,
                                   y = prop_variance)) +
  # Size of the markers for data points
  geom_point(size = 2) +
  # Line plot
  geom_line() +
  # Label the resulting values adjacent to each data point
  geom_text(aes(label = round(prop_variance, 3)),
            hjust = -0.1, vjust = -0.5, size = 3) +
  # Add labels to the x-axis for all 15 components
  scale_x_continuous(breaks = 1:15, labels = paste0("PC", 1:15)) +
  labs(x = "Principle Components",
       y = "Proportion of Variance",
       title = "Proportion of Variance Explained by Principal Components") +
  theme_minimal() +
  theme(plot.title = element_text(face = "bold"))

scree
```



The scree plot is important as it assists in determining the number of principal components required [2]. To analyze the scree plot, the key is to identify the "elbow". The "elbow" is the point at which each principal component's value of the proportion of variance begins to level off or proportion of variance drops off. Observing the plot, the components on the left have the most variance and have more significance to explain the total variance.

To select an adequate number of principal components, several must be considered that cumulatively demonstrate 80-90% of the total variance. The sum of first 6 principal components (0.401, 0.187, 0.134, 0.077, 0.064, 0.037) is equivalent to 90% of the total variance. This leads to the conclusion that principle components 1-6 (PC1 - PC6) are the most significant while components 7-15 will not add much value. The sum of first 4 principal components (0.401, 0.187, 0.134, 0.077) is equivalent to 80% of the total variance, which is still acceptable. This leads to the conclusion that principle components 1-4 (PC1 - PC4) are the most significant. Choosing 4-6 principle components is adequate.

Relating Principal Components to Original Variables

Each principal component (PC) is a linear combination of the original variables (predictors). You can examine the loadings of each principal component to determine which original variables contribute most to the important PCs. Using `pca_result$rotation`, the loadings for each principal component are examined. Variables with high values in these loadings are significant.

First, we can use 4 principle components for simplicity. By looking at `rotation`, identify the original variables with the largest loadings (absolute value) in PC1 to PC4. From this we can determine 6 factors that are of potential use.

```
pca_model_scaled$rotation[, 1:4]
```

```
##                PC1         PC2           PC3         PC4
## M      -0.30371194  0.06280357  0.1724199946 -0.02035537
## So     -0.33088129 -0.15837219  0.0155433104  0.29247181
## Ed      0.33962148  0.21461152  0.0677396249  0.07974375
## Po1     0.30863412 -0.26981761  0.0506458161  0.33325059
## Po2     0.31099285 -0.26396300  0.0530651173  0.35192809
## LF      0.17617757  0.31943042  0.2715301768 -0.14326529
## M.F     0.11638221  0.39434428 -0.2031621598  0.01048029
## Pop     0.11307836 -0.46723456  0.0770210971 -0.03210513
## NW     -0.29358647 -0.22801119  0.0788156621  0.23925971
## U1      0.04050137  0.00807439 -0.6590290980 -0.18279096
## U2      0.01812228 -0.27971336 -0.5785006293 -0.06889312
## Wealth  0.37970331 -0.07718862  0.0100647664  0.11781752
## Ineq   -0.36579778 -0.02752240 -0.0002944563 -0.08066612
## Prob   -0.25888661  0.15831708 -0.1176726436  0.49303389
## Time   -0.02062867 -0.38014836  0.2235664632 -0.54059002
```

The 6 factors of use are `Wealth`, `Ineq`, `So`, `Po1`, `Po2`, `M.F`, `Pop` and `LF`, as they have the highest absolute values or loading.

Regression Model using 4 Principle Components (6-factor)

A regression model is developed with `lm` to predict the crime rate. Using 4 principal components, 6 factors are identifed for use. The 6 factors of use are `Wealth`, `Ineq`, `So`, `Po1`, `Po2`, `M.F`, `Pop` and `LF`. A new dataframe is created with the 6 factors and original data, stored in `selected_vars`. **Note that this is unscaled data to make a prediction.** A summary of the regression model is stated using the `lm` model and original data stored in `selected_vars`. Finally, a crime rate is predicted using data for a new city from Question 8.2, with the final result printed below.

```
selected_vars <- data[, c("Wealth", "Ineq", "So", "Po1", "Po2", "M.F", "Pop",
                          "LF", "Crime")]

# Build the linear regression model
regression_model <- lm(Crime ~ Wealth + Ineq + So + Po1 + Po2 + M.F + Pop + LF,
                       data = selected_vars)

# Summary of Regression Model
cat(paste("\nRegression Model Summary:\n"))
```

```
##
## Regression Model Summary:
```

```
lm_summary <- summary(regression_model)
print(lm_summary)
```

```
##
## Call:
## lm(formula = Crime ~ Wealth + Ineq + So + Po1 + Po2 + M.F + Pop +
##     LF, data = selected_vars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -618.14 -152.88   18.15  119.32  412.05
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4914.9666  1552.6699  -3.165  0.00305 **
## Wealth          0.1509     0.1078   1.401  0.16947
## Ineq           69.1230    24.8433   2.782  0.00836 **
## So             34.8461   130.7486   0.267  0.79129
## Po1           142.4053   113.0817   1.259  0.21560
## Po2           -30.4493   120.7998  -0.252  0.80235
## M.F            23.2343    16.5454   1.404  0.16836
## Pop            -0.6720     1.3909  -0.483  0.63177
## LF            799.6707  1177.9274   0.679  0.50133
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 243.9 on 38 degrees of freedom
## Multiple R-squared:  0.6716, Adjusted R-squared:  0.6024
## F-statistic: 9.713 on 8 and 38 DF,  p-value: 3.26e-07
```

```
# Create a data frame with the given city data
new_city <- 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
)

# Make predictions using the linear regression model
predicted_crime <- predict(regression_model, newdata = new_city)

# Print the predictions
cat(paste("\nPredicted Crime (4 PC 6-factors):\n"))
```

```
##
## Predicted Crime (4 PC 6-factors):
```

```
predicted_crime
```

```
##        1
## 789.2719
```

The predicted crime rate result is 789 offenses per 100,000 population in 1960 considering the provided city data.

Regression Model using 6 Principle Components (11-factor)

Next, 6 principle components are used for a crime rate prediction. The sum of first 6 principal components (0.401, 0.187, 0.134, 0.077, 0.064, 0.037) is equivalent to 90% of the total variance. By looking at `rotation`, identify the original variables with the largest loadings (absolute value) in PC1 to PC6. From this we can determine 11 factors that are of potential use.

```
pca_model_scaled$rotation[, 1:6]
```

```
##                PC1         PC2           PC3         PC4         PC5
## M      -0.30371194  0.06280357  0.1724199946 -0.02035537 -0.35832737
## So     -0.33088129 -0.15837219  0.0155433104  0.29247181 -0.12061130
## Ed      0.33962148  0.21461152  0.0677396249  0.07974375 -0.02442839
## Po1     0.30863412 -0.26981761  0.0506458161  0.33325059 -0.23527680
## Po2     0.31099285 -0.26396300  0.0530651173  0.35192809 -0.20473383
## LF      0.17617757  0.31943042  0.2715301768 -0.14326529 -0.39407588
## M.F     0.11638221  0.39434428 -0.2031621598  0.01048029 -0.57877443
## Pop     0.11307836 -0.46723456  0.0770210971 -0.03210513 -0.08317034
## NW     -0.29358647 -0.22801119  0.0788156621  0.23925971 -0.36079387
## U1      0.04050137  0.00807439 -0.6590290980 -0.18279096 -0.13136873
## U2      0.01812228 -0.27971336 -0.5785006293 -0.06889312 -0.13499487
## Wealth  0.37970331 -0.07718862  0.0100647664  0.11781752  0.01167683
## Ineq   -0.36579778 -0.02752240 -0.0002944563 -0.08066612 -0.21672823
## Prob   -0.25888661  0.15831708 -0.1176726436  0.49303389  0.16562829
## Time   -0.02062867 -0.38014836  0.2235664632 -0.54059002 -0.14764767
##                  PC6
## M      -0.449132706
## So     -0.100500743
## Ed     -0.008571367
## Po1    -0.095776709
## Po2    -0.119524780
## LF      0.504234275
## M.F    -0.074501901
## Pop     0.547098563
## NW      0.051219538
## U1      0.017385981
## U2      0.048155286
## Wealth -0.154683104
## Ineq    0.272027031
## Prob    0.283535996
## Time   -0.148203050
```

The 11 factors of use are `Wealth`, `Ineq`, `So`, `Po1`, `Po2`, `M.F`, `Pop`, `LF`, `U1`, `U2` and `Prob`. A new dataframe is created with the 11 factors and original data, stored in `selected_vars2`. **Note that this is unscaled data to make a prediction.**

```r
# Create a new dataset with selected variables
selected_vars2 <- data[, c("Wealth", "Ineq", "So", "Po1", "Po2", "M.F", "Pop",
                          "LF", "U1", "U2", "Prob", "Crime")]

# Build the linear regression model
regression_model2 <- lm(Crime ~ Wealth + Ineq + So + Po1 + Po2 + M.F + Pop +
                         LF + U1 + U2 + Prob, data = selected_vars2)

# Summary of Regression Model
cat(paste("\nRegression Model Summary (6 PC 11-factors):\n"))
```

```
##
## Regression Model Summary (6 PC 11-factors):
```

```r
lm_summary2 <- summary(regression_model2)
print(lm_summary2)
```

```
##
## Call:
## lm(formula = Crime ~ Wealth + Ineq + So + Po1 + Po2 + M.F + Pop +
##     LF + U1 + U2 + Prob, data = selected_vars2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -562.3 -133.3    2.4  138.0  390.0
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.812e+03  1.620e+03  -2.971  0.00533 **
## Wealth       8.593e-02  1.085e-01   0.792  0.43382
## Ineq         5.610e+01  2.485e+01   2.257  0.03034 *
## So           1.030e+02  1.467e+02   0.702  0.48717
## Po1          1.364e+02  1.114e+02   1.225  0.22893
## Po2         -3.283e+01  1.186e+02  -0.277  0.78357
## M.F          3.317e+01  2.108e+01   1.573  0.12461
## Pop         -1.049e+00  1.408e+00  -0.745  0.46118
## LF           5.850e+02  1.483e+03   0.394  0.69570
## U1          -3.344e+03  4.525e+03  -0.739  0.46479
## U2           6.597e+01  8.671e+01   0.761  0.45189
## Prob        -4.450e+03  2.048e+03  -2.173  0.03661 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 235.5 on 35 degrees of freedom
## Multiple R-squared:  0.7179, Adjusted R-squared:  0.6293
## F-statistic: 8.098 on 11 and 35 DF,  p-value: 8.559e-07
```

```r
# Make predictions using the linear regression model
predicted_crime2 <- predict(regression_model2, newdata = new_city)

# Print Predictions
cat(paste("\nPredicted Crime (6 PC 11-factors):\n"))
```

```
##
## Predicted Crime (6 PC 11-factors):
```

```r
print(predicted_crime2)
```

```
##        1
## 712.0623
```

The predicted crime rate result is 712 offenses per 100,000 population in 1960 considering the provided city data.

Comparison to Question 8.2: R-Squared

The table below compares the R^2 and adjusted R^2 values between Question 8.2 with using PCA including 4 principle components and 6 principle components. Utilizing training data is still a concern for some models because of overfitting. Not only is this from using training data, but also because we only have 47 data points.

Using 4 principle components and 6 factors was an improvement compared to 6-factor cross-validated results. 6 principle components and 11 factors was an even better result and much improved from solutions in Question 8.2.

Model	R^2	Adjusted R^2
15-factor (Training Set)	0.803	0.708
15-factor (Cross-Validated)	0.413	0.130
6-factor (Training Set)	0.766	0.731
6-factor (Cross-Validated)	0.638	0.584
4 PC 6-factor	0.672	0.602
6 PC 11-factor	0.718	0.629

The reasoning for the original 15-factor and 6-factor models performing poorly in Question 8.2 is shown by using cross-validation with a lower R^2 value that suggest overfitting. As for PCA results in the bottom two rows, using PCA transforms original data into uncorrelated components to eliminate multicollinearity issues.

With 4 principle components there is enough variance in the data set that can be explained that allows the model to use less predictors with similar performance. Although the training set R^2 (0.672) is lower than the 15-factor model, the adjusted R^2 is higher (0.602) than the 15-factor cross-validated model (0.130). This shows that the model is more robust and performs better on new data because PCA reduces noise and complexity.

With 6 principal components, the model incorporates a larger amount of variance from the original data, leading to a higher R^2 (0.718) and adjusted R^2 (0.629). This model finds a balance between complexity and variance, with 11 significant factors. The performance on cross-validation is better than both the 15-factor and 6-factor models, and this suggests that PCA has effectively reduced irrelevant noise and multicollinearity while keeping important patterns from the data.

Comparison to Question 8.2: Crime Rate

Comparing to Question 8.2 in the previous homework, crime rate was predicted on training sets to be 155 (15-factor) and 1304 (6-factor). Using the `lm` model with 15 predictors on the training set, the predicted crime rate result was 155 offenses per 100,000 population in 1960 considering the provided city data. This was an issue because by comparing to the provided data set, the minimum value for `Crime` was 342, more than twice as high. The `lm` model with 6 predictors was used as an attempt to resolve the issue of using 15 predictors where many were not significant. The predicted crime rate result was 1304 offenses per 100,000 population. This was more in range with the data as it is above the minimum value of `Crime` (342) and below the maximum (1993).

Compare the results from Question 8.2 to the predicted crime rate with the use of PCA and *unscaled* predictions in the table below. In addition for reference, the table also includes the mean and median of crime rate in the original data.

Model	Predicted Crime Rate
15-factor (Training Set)	155
6-factor (Training Set)	1304
4 PC 6-factor	789
6 PC 11-factor	712
Mean	905
Median	831

PCA provides advantages when predicting the crime rate because it reduces overfitting and multicollinearity. The 15-factor (Training Set) model crime rate is severely underfitted because of high correlation between predictors and noise in the data set. The 6-factor (Training Set) model predicts a reasonable crime rate value, but is far above the mean and median. PCA, on the other hand, transforms the original variables into uncorrelated components that represent the most important patterns in the data. In the 4 PC 6-factor model, the predicted crime rate of 789 is very close to the median crime rate (831). Similarly, the 6 PC 11-factor model predicts a crime rate of 712, also much closer to the actual median and mean values. These results suggest that the model effectively captures the key patterns associated with crime prediction. In addition, even with a large number of factors (11), using PCA performs well to predict the crime rate unlike the 15-factor training set. The reasoning for this is the PCA process removes multicollinearity, reduces noise, and allows for more accurate and stable predictions, leading to better performance.

References

[1] O. -Australasian Data and S. Library, “Effect of punishment regimes on crime rates.” http://www.statsci.org/data/general/uscrime.html, 2024.

[2] J. Gareth, W. Daniela, H. Trevor, and T. Robert, *An introduction to statistical learning: With applications in r*. Spinger, 2013.

[3] “Prcomp: Principal components analysis - r documentation — rdocumentation.org.” https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/prcomp.

[4] H. Wickham, *ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York, 2016.