Principal Component Regression

Sriya Cheedella, Mia Shu

12/2/2019

Introduction

- Motivation

  When we have more than two covariates, multicollinearity impacts our model construction, parameter estimation, and prediction. In order to reduce its impact on our model, we reduce multicollinearity among variables by fitting the Principal Components.

- Methodology

  Break the collinear parts into uncorrelated smaller parts

Definitions

- Multicollinearity

Multicollinearity exists among the predictor variables when these variables are correlated among themselves.

Example: weight and height; education level and salary

- Confounding

The result of multicollinearity is often termed confounding: the situation when the correlation between two variables is aberrant due to a third variable included in the analysis.

Regression Model

Simple linear regression

$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$

Y_i Response at ith trial

β_0, β_1 Regression coefficients

X_i Predictors at ith trial

$\varepsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$ Random error

Matrix Representation

Model $Y = X\beta + \varepsilon$

Residual $e_i = Y_i - \hat{Y}_i = Y - Xb$

b is the estimated vector of β

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \vdots \\ 1 & X_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Multiple Linear Regression

Model

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip} + \varepsilon_i$$

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & X_{11} & X_{12} & \cdots & X_{1p} \\ 1 & X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & X_{n1} & X_{n2} & \cdots & X_{np} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Diagnostic for Multicollinearity

- ggpairs(X)& cor(X)

  Look for high pairwise correlation

- vif(X)

  5-10 moderately high

  $<$ 10 extremely high

The Least Squares Estimator

$\mathbf{y}$	$\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$
$\mathbf{Ax}$	$\mathbf{A}^T$
$\mathbf{x}^T\mathbf{A}$	$\mathbf{A}$
$\mathbf{x}^T\mathbf{x}$	$2\mathbf{x}$
$\mathbf{x}^T\mathbf{Ax}$	$\mathbf{Ax} + \mathbf{A}^T\mathbf{x}$

$$
\begin{aligned}
RSS(b) &= \sum e_i^2 = e^\top e \\
&= (Y - Xb)^\top (Y - Xb) \\
&= Y^\top Y - Y^\top Xb - b^\top X^\top Y + b^\top X^\top Xb \\
\frac{dRSS}{db} &= -2X^\top Y + 2X^\top Xb = 0 \quad b = \left(X^\top X\right)^{-1} X^\top Y
\end{aligned}
$$

Variance Covariance Matrix

- Denoted $\sigma^2\{b\}$
- Estimate $\sigma^2 \rightarrow s^2 = MSE = \frac{\sum e_i^2}{n-2}$

$$\sigma^2\{b\} = \sigma^2 \left(X^\top X\right)^{-1}$$

- Multicollinearity

$\left(X^\top X\right)^{-1}$ close to singular

Sensitive to small perturbation $\Rightarrow$ Unreliable parameter estimates

Geometrical Representation

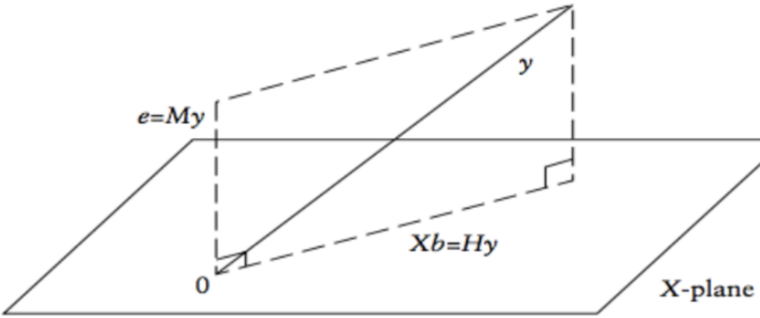


Figure 1: img geom

$$M = I - X\left(X^\top X\right)^{-1} X^\top$$

$$e = MY = Y + \hat{Y} = Y + Xb\hat{Y} = HY = Xb$$

$$H = X\left(X^\top X\right)^{-1} X^\top \Rightarrow \hat{Y} \perp e$$

Spectral Decomposition

$$A = \lambda_1 u_1 u_1^\top + \lambda_2 u_2 u_2^\top + \cdots + \lambda_n u_n u_n^\top$$

where A is a square symmetric matrix

$$A = PDP^\top$$

P Orthonormal eigenvectors

D Diagonal matrix of eigenvalues

Principal Component Analysis

- Reduce a large set of correlated predictor variables to a smaller uncorrelated set.
- The principal component for a set of vectors are a set of linear combinations of the vectors chosen so that such set captures the most information in a smaller subset of vectors.

Procedure

- Standardize $\frac{X-\mu}{\sigma}$
- Find $X^\top X = PDP^\top = Z^\top Z$

Singular Value Decomposition of X $\Rightarrow$ Truncated SVD

Maximize Rayleigh Coefficients

$w_1 = \text{argmax}\left\{\frac{w^\top x^\top x w}{w^\top w}\right\}$

$x_k = x - \sum_{s=1}^{k-1} x w_s w_s^\top$
$w_k = \text{argmax}\left\{\frac{w^\top x_k^\top x_k w}{w^\top w}\right\}$

Procedure Continued

- Step Two

Fit Y on Z (OLS)

- Step Three

Choose components

- Step Four

Transform back to x scale

RidgeReg Data

```r
X1 <- 1:18; X2 <- c(2,4,6,7,7,7,8,10,12,13,13,13,14,16,18,19,19,19); X3 <- c(1,2,4,3,2,1,1
testdf <- data.frame(cbind(X1,X2,X3,Y))
testmod <- lm(Y~., data = testdf)
# vif(testmod)
testpcr <- pcr(Y~., data = testdf, scale=TRUE, validation = "CV")
cor(testdf)
```

```
##             X1        X2          X3         Y
## X1  1.00000000 0.9878415 -0.01505107 0.9855438
## X2  0.98784149 1.0000000  0.13381266 0.9955738
## X3 -0.01505107 0.1338127  1.00000000 0.1165387
## Y   0.98554384 0.9955738  0.11653870 1.0000000
```

```r
summary(testpcr)
```

```
## Data:    X dimension: 18 3
##  Y dimension: 18 1
## Fit method: svdpc
## Number of components considered: 3
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps
## CV           11.19    1.992    1.282    1.308
## adjCV        11.19    1.721    1.269    1.289
##
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps
## X    66.50    99.97   100.00
## Y    99.05    99.05    99.15
```

Iris Data

```r
irismod <- lm(Sepal.Length~., data = iris)
# vif(irismod)
cor(iris[1:4])
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
## Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
## Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
## Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
```

```r
irispcr <- pcr(Sepal.Length~., data = iris, scale = TRUE, validation = "CV")
summary(irispcr)
```

```
## Data:    X dimension: 150 5
##  Y dimension: 150 1
## Fit method: svdpc
## Number of components considered: 5
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps
## CV          0.8308   0.5122   0.5086   0.3955   0.3348   0.3191
## adjCV       0.8308   0.5118   0.5080   0.3948   0.3341   0.3181
##
## TRAINING: % variance explained
##               1 comps  2 comps  3 comps  4 comps  5 comps
## X               56.20    88.62    99.07    99.73   100.00
## Sepal.Length    62.71    63.58    78.44    84.95    86.73
```

Formula One Racing Data: Description

- Provides data from Formula One World Championships from 1950-2017 about constructors, lap times, race drivers, etc.
- Given 13 .csv files to parse from.
- We wanted to see which variables best captured the time spent on a circuit.
- The columns used were circuit times, number of laps, number of pit stops, pit stop times, constructor points and driver points.

Verify Multicollinearity

```r
cor(hungres[,-c(1,2)])
```

```
##                milliseconds        laps    pitStops MillisecondsPS   conPoints
## milliseconds     1.00000000  0.92071476 -0.08693557    -0.03347315  0.03369049
## laps             0.92071476  1.00000000 -0.06366250    -0.06574144  0.33044566
## pitStops        -0.08693557 -0.06366250  1.00000000     0.97835968 -0.05373143
## MillisecondsPS  -0.03347315 -0.06574144  0.97835968     1.00000000 -0.15541410
## conPoints        0.03369049  0.33044566 -0.05373143    -0.15541410  1.00000000
## drivPoints       0.11927110  0.37747886 -0.10851265    -0.20113019  0.88398060
##                drivPoints
## milliseconds    0.1192711
## laps            0.3774789
## pitStops       -0.1085127
## MillisecondsPS -0.2011302
## conPoints       0.8839806
## drivPoints      1.0000000
```

```r
hungmod <- lm(milliseconds ~ laps + pitStops + MillisecondsPS + conPoints + drivPoints,
              data = hungres)
# vif(hungmod)
```

Principal Component Analysis

```
hungmat <- as.matrix(hungres[,-c(1,2)])
pca <- prcomp(hungmat, scale = TRUE, center = TRUE)
summary(pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     1.5802 1.3618 1.2192 0.33964 0.19139 0.09921
## Proportion of Variance 0.4162 0.3091 0.2477 0.01923 0.00611 0.00164
## Cumulative Proportion  0.4162 0.7253 0.9730 0.99225 0.99836 1.00000
```

Principal Component Regression

```r
hungpcr <- pcr(milliseconds ~ laps + pitStops + MillisecondsPS +
                 conPoints + drivPoints, data = hungres, scale = TRUE,
                 validation = "CV")
summary(hungpcr)
```

```
## Data:    X dimension: 24 5
##  Y dimension: 24 1
## Fit method: svdpc
## Number of components considered: 5
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps
## CV          205719   207896   209380    78638    76811    75405
## adjCV       205719   207447   208295    76846    75056    73201
##
## TRAINING: % variance explained
##               1 comps  2 comps  3 comps  4 comps  5 comps
## X              46.579    81.88    97.42    99.69   100.00
## milliseconds    7.567    10.79    92.87    93.22    94.96
```

```r
coef(hungpcr, intercept = TRUE)
```

```
## , , 5 comps
##
##                 milliseconds
## (Intercept)      1283658.573
## laps              199457.435
## pitStops         -157998.285
## MillisecondsPS    154526.064
## conPoints         -48469.888
## drivPoints          5510.244
```

```r
hungpcrlog <- pcr(log(milliseconds) ~ laps + pitStops +
                    log(MillisecondsPS) + conPoints + drivPoints,
                    data = hungres, scale = TRUE, validation = "CV")
summary(hungpcrlog)
```

```
## Data:    X dimension: 24 5
##  Y dimension: 24 1
## Fit method: svdpc
## Number of components considered: 5
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps
## CV         0.03605  0.03608  0.03642  0.01537  0.01492  0.01311
## adjCV      0.03605  0.03598  0.03624  0.01508  0.01464  0.01283
##
## TRAINING: % variance explained
##                    1 comps  2 comps  3 comps  4 comps  5 comps
## X                   46.592    81.87    97.40    99.66   100.00
## log(milliseconds)    8.334    10.83    92.45    92.80    95.05
```

```r
coef(hungpcrlog, intercept = TRUE)
```

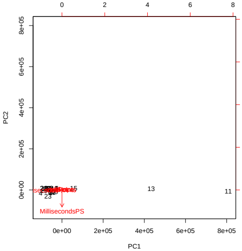
```
## , , 5 comps
##
##                      log(milliseconds)
## (Intercept)               13.386699700
## laps                       0.035582342
## pitStops                  -0.029313474
## log(MillisecondsPS)        0.028648215
## conPoints                 -0.009769960
## drivPoints                 0.001987028
```

Importance of Standardization

```r
noscale <- prcomp(hungres[,-c(1,2)], scale = FALSE)
summary(noscale)
```

```
## Importance of components:
##                              PC1       PC2   PC3  PC4    PC5     PC6
## Standard deviation     2.014e+05 1.081e+04 97.98 20.3 0.7074 0.08798
## Proportion of Variance 9.971e-01 2.870e-03  0.00  0.0 0.0000 0.00000
## Cumulative Proportion  9.971e-01 1.000e+00  1.00  1.0 1.0000 1.00000
```

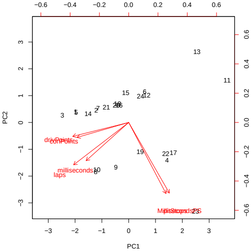
```r
biplot(noscale, scale = 0)
```




```r
scale <- prcomp(hungmat, scale = TRUE, center = TRUE)
summary(scale)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     1.5802 1.3618 1.2192 0.33964 0.19139 0.09921
## Proportion of Variance 0.4162 0.3091 0.2477 0.01923 0.00611 0.00164
## Cumulative Proportion  0.4162 0.7253 0.9730 0.99225 0.99836 1.00000
```

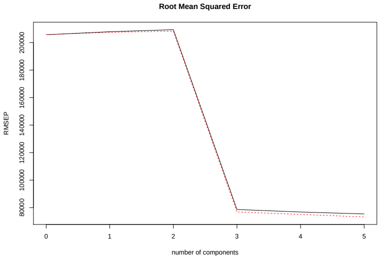
```r
biplot(scale, scale = 0)
```

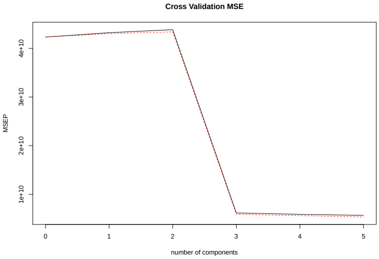


Validation Plots

```r
validationplot(hungpcr, main = "Root Mean Squa
```

```r
validationplot(hungpcr, val.type="MSEP", main
```

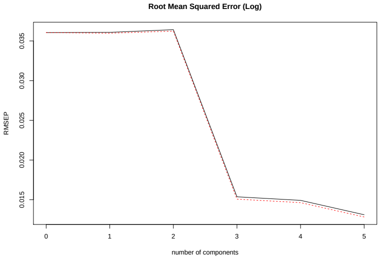


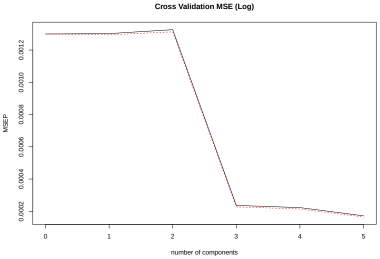


Validation Plots (Log)

```
validationplot(hungpcrlog, main = "Root Mean S
```

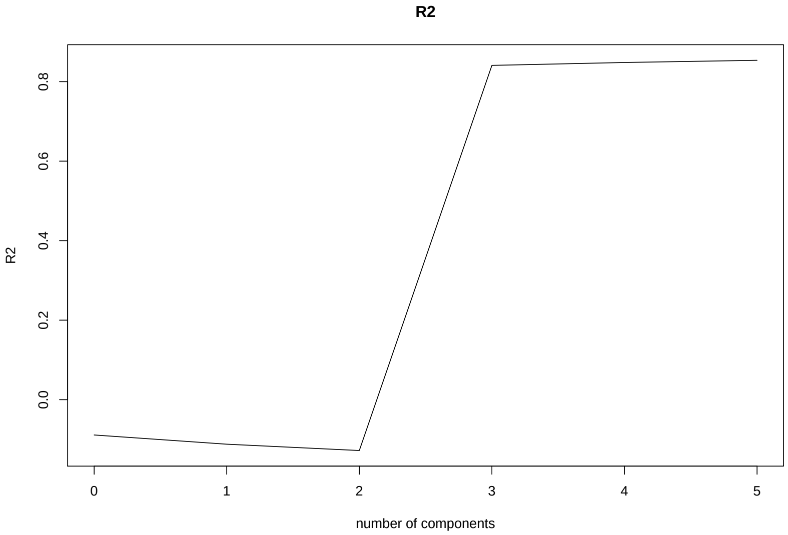
```
validationplot(hungpcrlog, val.type="MSEP", ma
```





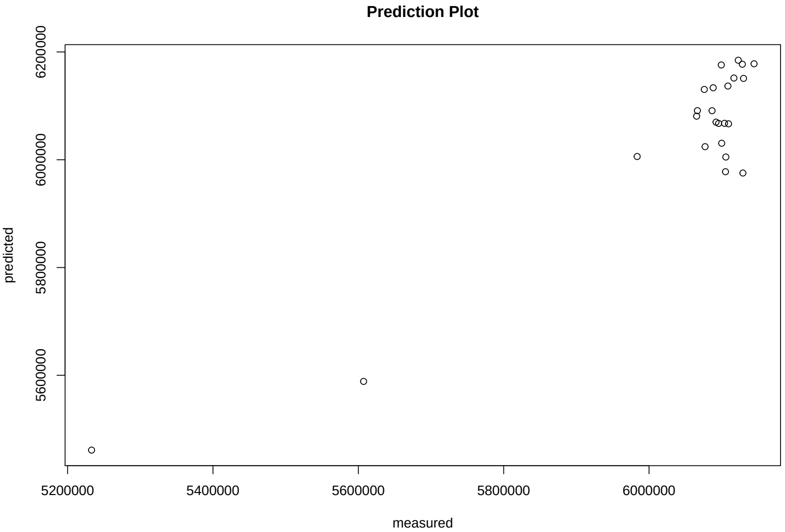
R^2 Plot

```
validationplot(hungpcr, val.type = "R2", main = "R2")
```



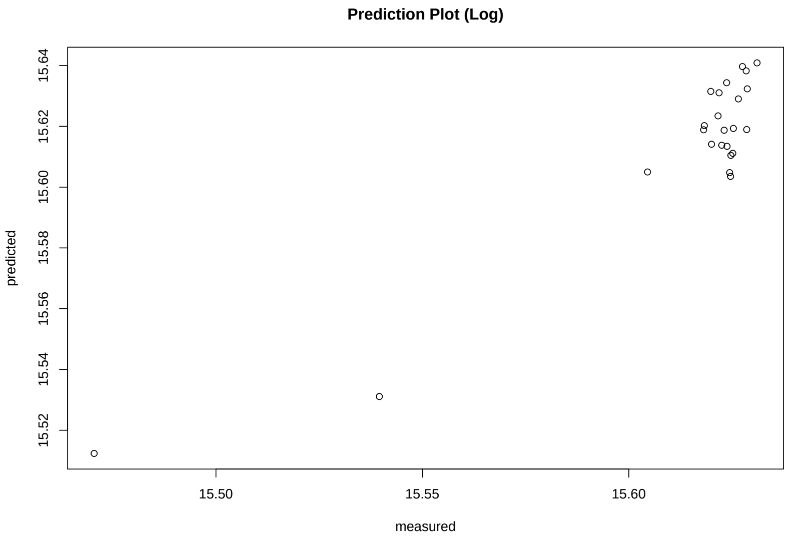
Prediction Plot

```r
predplot(hungpcr, main = "Prediction Plot")
```



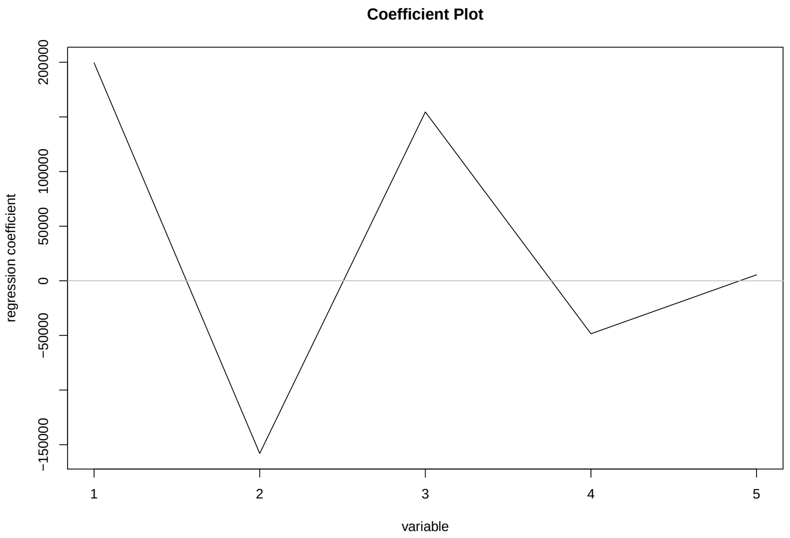
Prediction Plot (Log)

```
predplot(hungpcrlog, main = "Prediction Plot (Log)")
```



Coefficient Plot

```
coefplot(hungpcr, main = "Coefficient Plot")
```



References

1. Michael H. Kutner, Christopher J. Nachtsheim, and John Neter. Applied Linear Regression Models. New York, NY: McGraw-Hill/Irwin, 2004.
2. Johnston, R., Jones, K. & Manley, D. Qual Quant (2018) 52: 1957. https://doi.org/10.1007/s11135-017-0584-6
3. https://www.whitman.edu/Documents/Academics/Mathematics/2017/Perez.pdf
4. https://datascienceplus.com/multicollinearity-in-r/
5. https://web.njit.edu/~wguo/Math644_2012/Math644_Chapter%201_part2.pdf
6. https://ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Principal_Components_Regression.pdf
7. https://en.wikipedia.org/wiki/Principal_component_analysis

References Continued

8. Dr. Christian Lucero, CMDA 4654 Lecture 08, Correlation and Least Squares, Lecture 14 MLR
9. Volodymyr Kuleshov, Fast algorithms for sparse principal component analysis based on Rayleigh quotient iteration http://proceedings.mlr.press/v28/kuleshov13.pdf
10. https://ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Principal_Components_Regression.pdf
11. https://www.r-bloggers.com/performing-principal-components-regression-pcr-in-r/