STATS 411 Project Code

Anum Damani

March 14, 2025

I used this NBA Players data from Kaggle: https://www.kaggle.com/datasets/justinas/nba-players-data

Exploratory Data Analysis (EDA):

```r
library(stats)
library(corrplot)
#https://www.kaggle.com/datasets/justinas/nba-players-data
data <- read.csv("/Users/anumdamani/stats411/project/all_seasons.csv")
colnames(data)
str(data)
unique(data$season)
numeric_data <- data[sapply(data, is.numeric)]
correlation_matrix <- cor(numeric_data)
correlation_matrix
corrplot(correlation_matrix, method = "circle")

summary(numeric_data)

# Subset the data to include only correlated variables
subset_data <- data.frame(data$player_height, data$player_weight, data$gp,
                          data$pts, data$reb, data$ast, data$oreb_pct,
                         data$dreb_pct, data$usg_pct, data$ast_pct)

cor_matrix <- cor(subset_data)
print(cor_matrix)

#png("/Users/anumdamani/stats411/project/images/correlation_plot.png", width = 800, height = 600)
corrplot(cor_matrix, method = "circle")
#dev.off()

corrplot(cor_matrix, method = "circle")
```

Principal Component Analysis (PCA):

```r
# Standardize the data
data_scaled <- scale(subset_data)

# Perform PCA
pca_result <- prcomp(data_scaled, center = TRUE, scale. = TRUE)
```

```r
# Summary of PCA results
summary(pca_result)

# Get the standard deviations of each principal component
sdev <- pca_result$sdev

# Calculate the proportion of variance explained by each principal component
explained_variance <- (sdev^2) / sum(sdev^2)

# Create a Scree plot
#png("/Users/anumdamani/stats411/project/images/scree_plot.png", width = 800, height = 600)
plot(explained_variance, type = "b", pch = 19, xlab = "Principal Components",
     ylab = "Proportion of Variance Explained",
     main = "Scree Plot", col = "blue")
#dev.off()

# Another way to create Scree Plot:
xbar <- colMeans(data_scaled)
S <- cov(data_scaled)
ld <- eigen(S)$values
ei <- eigen(S)$vectors
plot(1:length(ld), ld, type = "l", xlab = "Number of PCs", ylab="lambda_i")

# Plot first two principal components
plot(pca_result$x[, 1:2], main = "PCA - First Two Components",
     xlab = "Principal Component 1", ylab = "Principal Component 2")

# Scree plot
screeplot(pca_result, main = "Scree Plot", xlab = "Principal Components")

# Creating biplot
biplot(pca_result)

# Get the loadings
loadings <- pca_result$rotation
loadings_pc1 <- loadings[, 1]
loadings_pc2 <- loadings[, 2]
loadings_pc1
loadings_pc2

# Finding correlations between PCs and original variables:
cor_PC1 <- (ei[,1] * sqrt(ld[1])) / sqrt(diag(S))
cor_PC1
cor_PC2 <- (ei[,2] * sqrt(ld[2])) / sqrt(diag(S))
cor_PC2

# Covariances:
loadings_pc1 * pca_result$sdev[1]
loadings_pc2 * pca_result$sdev[2]
```

Canonical Correlation Analysis (CCA):

```r
library(stats)
library(CCA)
library(CCP)
library(MASS)
data <- read.csv("/Users/anumdamani/stats411/project/all_seasons.csv")

# Subset the data to include only variables of interest

# Physical attributes:
# y1 = player height
# y2 = player weight

# Performance attributes:
# z1 = points
# z2 = rebounds
# z3 = oreb_pct
# z4 = dreb_pct
# z5 = games played (gp)
# z6 = assists (ast)
# z7 = usage percentage (usg_pct)
# z8 = assist percentage (ast_pct)

# So, p = 2 and q = 8
subset_data <- data.frame(
  data$player_height, data$player_weight, data$gp,
  data$pts, data$reb, data$ast, data$oreb_pct,
  data$dreb_pct, data$usg_pct, data$ast_pct
)

subset_data <- scale(subset_data) # Standardize the data
cor_matrix <- cor(subset_data)
print(cor_matrix)
head(subset_data)

(R1 <- cor(subset_data))
R11 <- R1[1:2, 1:2]
R12 <- R1[1:2, 3:10]
R21 <- R1[3:10, 1:2]
R22 <- R1[3:10, 3:10]

eR11 <- eigen(R11)
R11.5 <- eR11$vectors %*% diag(eR11$values^(-1/2)) %*% t(eR11$vectors)
eR22 <- eigen(R22)
R22.5 <- eR22$vectors %*% diag(eR22$values^(-1/2)) %*% t(eR22$vectors)
E1 <- R11.5 %*% R12 %*% solve(R22) %*% R21 %*% R11.5
E2 <- R22.5 %*% R21 %*% solve(R11) %*% R12 %*% R22.5
E1
E2

eigen(E1)$values
eigen(E2)$values
```

```
# min(p, q) = min(2, 8) = 2 sample canonical correlations
(cc1 <- sqrt(eigen(E1)$values[1])) # First sample canonical correlation
(cc2 <- sqrt(eigen(E2)$values[2])) # Second sample canonical correlation

# First pair of canonical variables
(a1 <- R11.5 %*% (eigen(E1)$vectors[,1]))
(b1 <- R22.5 %*% (eigen(E2)$vectors[,1]))
# Second pair of canonical variables
(a2 <- R11.5 %*% (eigen(E1)$vectors[,2]))
(b2 <- R22.5 %*% (eigen(E2)$vectors[,2]))

# Correlations between canonical variates and original variables
(r_ux1 <- t(cbind(a1, a2)) %*% R11)
(r_ux2 <- t(cbind(a1, a2)) %*% R12)
(r_vx2 <- t(cbind(b1, b2)) %*% R22)
(r_vx1 <- t(cbind(b1, b2)) %*% R21)

# Evaluating Matrices of Errors of Approximation
az1 <- solve(-t(cbind(a1, a2)))
bz1 <- ginv(t(cbind(b1, b2)))
# First canonical variates:
# 1. R11 vs. SCov of Z1
az1[,2] %*% t(az1[,2])
# 2. R22 vs. SCov of Z2
bz1[,2] %*% t(bz1[,2])
# 3. R12 vs. SCov of (Z1 and Z2)
cc2 * az1[,2] %*% t(bz1[,2])

# Proportions of explained sample variance
# Proportion of total standardized sample variance in Z1 explained by U1
sum(diag((az1[,1] %*% t(az1[,1])))) / 2
# Proportion of total standardized sample variance in Z2 explained by V1
sum(diag((bz1[,1] %*% t(bz1[,1])))) / 8
# Proportion of total standardized sample variance in Z1 explained by U2
sum(diag((az1[,2] %*% t(az1[,2])))) / 2
# Proportion of total standardized sample variance in Z2 explained by V2
sum(diag((bz1[,2] %*% t(bz1[,2])))) / 8

# Canonical correlation analysis
cancor_result <- cancor(subset_data[, 1:2], subset_data[, 3:10])
print(cancor_result$cor) # Canonical correlations
print(cancor_result$xcoef) # Canonical coefficients for physical attributes
print(cancor_result$ycoef) # Canonical coefficients for performance attributes


# First hypothesis test:
# Likelihood ratio test
rho1 <- cc1
rho1
rho2 <- cc2
rho2
n <- nrow(data) # 12,844 observations
p <- 2
```

```r
q <- 8
wilks <- (1-(rho1^2))*(1-(rho2^2))
log_wilks <- log(wilks)
test_statistic <- -(n-1-0.5*(p+q+1))*log_wilks
test_statistic
critical_value <- qchisq(1 - 0.05, df=p*q)
critical_value

# Second hypothesis test
wilks2 <- (1-(rho2^2))
log_wilks <- log(wilks2)
test_statistic <- -(n - 1 - 0.5*(p+q+1)) * log_wilks
test_statistic
critical_value <- qchisq(1-0.05, df=(2-1)*(8-1))
critical_value
```