Group 5 Appendix

**Group 5: Daniel Brauckman, Jordan Gussett, Bryan Loeffler and Elijah Stahr**

2023-11-28

# Section 1: Data Cleaning and Visualization

# Load data sets  
batting <- read.csv("https://raw.githubusercontent.com/estahr/mva-project/main/Batting.csv")  
master <- read.csv("https://github.com/estahr/mva-project/raw/main/Master.csv")  
salaries <- read.csv("https://raw.githubusercontent.com/estahr/mva-project/main/Salaries.csv")  
  
# Keep only 2015 data, and remove columns that we are not analyzing  
  
batting2015 = batting[which(batting$yearID == 2015),]  
batting2015 = batting2015[,-c(2:5)]  
master = master[,c(1,16)]  
salaries2015 = salaries[which(salaries$yearID == 2015),]  
salaries2015 = salaries2015[,-c(1:3)]  
  
# We prepare for merge by checking if playerID is unique for all datasets  
  
cat("Non unique playerID in salaries dataset:" )  
print(length(salaries2015$playerID)-length(unique(salaries2015$playerID)))  
print("Non unique playerID in master dataset:" )  
print(nrow(master)-nrow(unique(master)))  
print("Non unique playerID in batting dataset:" )  
print(length(batting2015$playerID) - length(unique(batting2015$playerID)))  
  
# We can see that we need to adjust the batting dataset to have unique playerID  
# The reason the playerID is not unique is because mid-season trades duplicate the playerID  
# We don't care what team the player played for, just their stats, so we will combine rows and add stats  
  
batting2015 = aggregate(batting2015[,-1], batting2015["playerID"], sum)  
  
print("Non unique playerID in batting dataset (after aggregation):" )  
print(length(batting2015$playerID) - length(unique(batting2015$playerID)))  
  
# Now all playerID values are unique and we can merge  
  
data = merge(master, batting2015)  
data = merge(data, salaries2015)  
  
print("Merged Size" )  
print(dim(data))  
  
print("Size after removing missing data (there is none):")  
data = na.omit(data)  
print(dim(data))  
  
print("Size after removing players with 3 or less at bats:")  
data = data[-which(data$AB <= 3),]  
  
#qualified hitters  
#q\_data = data[-which((data$AB / data$G) <= 3),]  
  
print(dim(data))  
  
library(data.table)  
  
# Replace old column names with the actual words, might take this out if it makes outputs too messy  
setnames(data, old = c(3:19),  
 new = c("game", "at-bat", "run", "hit", "double", "triple", "home\_run", "run\_batted\_in", "stolen\_base", "caught\_stealing", "walk", "strike\_out", "intentional\_walk", "hit\_by\_pitch", "sacrifice\_hit", "sacrifice\_fly", "ground\_into\_double\_play"))  
  
# Make the row names into the unique playerID rather than arbitrary index  
rownames(data) = data[,c(1)]

#more visualizations  
  
# Remove given names for now  
numericalData = data[,-c(1,2)]  
  
#Plot assumed valuable statistics in comparison to salary

### Section 1.1

plot(numericalData$game, numericalData$salary, xlab = "Games", ylab = "Salary (in millions)", main = "Total Games Played to Salary", yaxt = "n")  
axis(side = 2, at = pretty(numericalData$salary), labels = paste0(pretty(numericalData$salary) / 1e6, "M"))  
regression\_model <- lm(salary ~ game, data = numericalData)  
abline(regression\_model, col = "red")

### Section 1.2

plot(numericalData$hit, numericalData$salary, xlab = "Hits", ylab = "Salary (in millions)", main = "Total Hits to Salary", yaxt = "n")  
axis(side = 2, at = pretty(numericalData$salary), labels = paste0(pretty(numericalData$salary) / 1e6, "M"))  
regression\_model <- lm(salary ~ hit, data = numericalData)  
abline(regression\_model, col = "red")

### Section 1.3

plot(numericalData$home\_run, numericalData$salary, xlab = "Home Runs", ylab = "Salary (in millions)", main = "Total Home Runs to Salary", yaxt = "n")  
axis(side = 2, at = pretty(numericalData$salary), labels = paste0(pretty(numericalData$salary) / 1e6, "M"))  
regression\_model <- lm(salary ~ home\_run, data = numericalData)  
abline(regression\_model, col = "red")

### Section 1.4

plot(numericalData$run\_batted\_in, numericalData$salary, xlab = "RBI", ylab = "Salary (in millions)", main = "Total RBI to Salary", yaxt = "n")  
axis(side = 2, at = pretty(numericalData$salary), labels = paste0(pretty(numericalData$salary) / 1e6, "M"))  
regression\_model <- lm(salary ~ run\_batted\_in, data = numericalData)  
abline(regression\_model, col = "red")

### Section 1.5

library(corrplot)

## corrplot 0.92 loaded

corrplot(cor(numericalData))

# Section 2: Dimensionality Reduction

# Remove given names for now  
numericalData = data[,-c(1,2)]  
  
#PCA  
  
mydata.pca = princomp(numericalData, cor=TRUE)  
  
  
#this looks awful with observations but at least gives a visualization of what something like salary is closely correlated to  
#looks better without the observations added in

### Section 2: Table 1

# We note that the first four loadings account for 81% of the variance, so we focus on those  
options(digits = 2)  
summary(mydata.pca)

## Importance of components:  
## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8  
## Standard deviation 3.30 1.41 0.987 0.869 0.820 0.783 0.672 0.632  
## Proportion of Variance 0.61 0.11 0.054 0.042 0.037 0.034 0.025 0.022  
## Cumulative Proportion 0.61 0.72 0.771 0.813 0.850 0.884 0.909 0.931  
## Comp.9 Comp.10 Comp.11 Comp.12 Comp.13 Comp.14 Comp.15  
## Standard deviation 0.622 0.448 0.43 0.3933 0.379 0.2664 0.2118  
## Proportion of Variance 0.021 0.011 0.01 0.0086 0.008 0.0039 0.0025  
## Cumulative Proportion 0.953 0.964 0.97 0.9829 0.991 0.9948 0.9973  
## Comp.16 Comp.17 Comp.18  
## Standard deviation 0.1632 0.12633 0.07513  
## Proportion of Variance 0.0015 0.00089 0.00031  
## Cumulative Proportion 0.9988 0.99969 1.00000

round(mydata.pca$loadings[,1:4], 2)

## Comp.1 Comp.2 Comp.3 Comp.4  
## game 0.29 0.05 0.02 0.13  
## at-bat 0.30 0.04 -0.02 0.09  
## run 0.30 0.04 0.00 -0.01  
## hit 0.29 0.04 -0.01 0.03  
## double 0.29 -0.02 0.03 0.05  
## triple 0.17 0.39 0.14 -0.10  
## home\_run 0.25 -0.23 0.06 0.03  
## run\_batted\_in 0.29 -0.14 0.02 0.07  
## stolen\_base 0.16 0.48 -0.12 -0.37  
## caught\_stealing 0.18 0.45 -0.03 -0.35  
## walk 0.27 -0.11 -0.02 -0.06  
## strike\_out 0.27 -0.01 0.06 0.13  
## intentional\_walk 0.19 -0.25 -0.04 -0.22  
## hit\_by\_pitch 0.20 0.00 0.14 0.19  
## sacrifice\_hit -0.04 0.38 -0.56 0.65  
## sacrifice\_fly 0.24 -0.09 -0.09 0.09  
## ground\_into\_double\_play 0.24 -0.14 -0.02 0.17  
## salary 0.07 -0.29 -0.78 -0.37

### Section 2.1

biplot(mydata.pca, col = c("black","red"), cex = 0.6, xlabs = rep("", nrow(numericalData)))

baseball.d = as.dist(1-cor(numericalData))  
  
hept.mds = cmdscale(baseball.d)

### Section 2.2

#Full plot  
  
plot(hept.mds, type = "n",  
 main = "MDS of Batting Data & Salary",  
 xlab = "First Component", ylab = "Second Component")  
  
text(hept.mds, labels = colnames(numericalData), cex = 1.2)

### Section 2.3

#Zoom in on left cluster  
  
plot(hept.mds, type = "n",  
 main = "MDS of Batting Data & Salary (zoomed in on left cluster)",  
 xlab = "First Component", ylab = "Second Component",  
 ylim = c(-0.15,0.25), xlim = c(-0.35, 0))  
  
text(hept.mds, labels = colnames(numericalData), cex = 0.8)

# Section 3: Cluster Analysis

baseball.hcs2 = hclust(baseball.d, "complete")  
# We chose complete linkage clustering

### Section 3.1

plot(rev(baseball.hcs2$height), main = "Complete Linkage HC Scree Plot")

### Section 3.2

plot(baseball.hcs2, main = "Complete Linkage HC Dendogram")

#scale data for kmeans  
nd.s <- scale(numericalData)  
  
set.seed(10)  
#perform kmeans clustering  
km <- kmeans(nd.s, centers = 4)  
  
  
#Observe variable z-score averages for each cluster  
km\_mean1 <- colMeans(subset(nd.s, km$cluster == 1))  
km\_mean2 <- colMeans(subset(nd.s, km$cluster == 2))  
km\_mean3 <- colMeans(subset(nd.s, km$cluster == 3))  
km\_mean4 <- colMeans(subset(nd.s, km$cluster == 4))

### Section 3: Table 1

#table showing km clusters  
table(km$cluster)

##   
## 1 2 3 4   
## 146 228 67 90

# Section 4: Confirmatory Factor Analysis

# Use if needed  
#install.packages(c("EFAtools", "psych", "sem"))  
  
library(sem)  
library(psych)  
library(EFAtools)

##   
## Attaching package: 'EFAtools'

## The following object is masked from 'package:psych':  
##   
## KMO

# We remove columns here that were too highly correlated. These were tested by running EFA repeatedly until goodness of fit statistics reached an acceptable level.  
efaData = numericalData[,-c(2:4,8,12,15)]  
  
# Run tests to see if data is suitable for EFA  
BARTLETT(efaData)

## ℹ 'x' was not a correlation matrix. Correlations are found from entered raw data.

KMO(efaData)

## ℹ 'x' was not a correlation matrix. Correlations are found from entered raw data.

nd.fa = fa(efaData, fm = "ml", rotate = "varimax", nfactors = 3)

### Section 4: Table 1

print(nd.fa, cut = 0.3)

## Factor Analysis using method = ml  
## Call: fa(r = efaData, nfactors = 3, rotate = "varimax", fm = "ml")  
## Standardized loadings (pattern matrix) based upon correlation matrix  
## ML1 ML2 ML3 h2 u2 com  
## game 0.76 0.42 0.41 0.93 0.074 2.2  
## double 0.73 0.38 0.48 0.90 0.101 2.3  
## triple 0.42 0.58 0.51 0.486 1.8  
## home\_run 0.52 0.69 0.77 0.234 2.0  
## stolen\_base 0.86 0.77 0.231 1.1  
## caught\_stealing 0.89 0.84 0.158 1.1  
## walk 0.49 0.31 0.72 0.86 0.141 2.2  
## intentional\_walk 0.69 0.56 0.444 1.3  
## hit\_by\_pitch 0.42 0.35 0.38 0.618 2.8  
## sacrifice\_fly 0.60 0.43 0.60 0.399 2.1  
## ground\_into\_double\_play 0.74 0.36 0.70 0.300 1.5  
## salary 0.32 0.12 0.880 1.3  
##   
## ML1 ML2 ML3  
## SS loadings 3.02 2.50 2.42  
## Proportion Var 0.25 0.21 0.20  
## Cumulative Var 0.25 0.46 0.66  
## Proportion Explained 0.38 0.32 0.30  
## Cumulative Proportion 0.38 0.70 1.00  
##   
## Mean item complexity = 1.8  
## Test of the hypothesis that 3 factors are sufficient.  
##   
## df null model = 66 with the objective function = 9.5 with Chi Square = 4992  
## df of the model are 33 and the objective function was 0.22   
##   
## The root mean square of the residuals (RMSR) is 0.02   
## The df corrected root mean square of the residuals is 0.03   
##   
## The harmonic n.obs is 531 with the empirical chi square 38 with prob < 0.25   
## The total n.obs was 531 with Likelihood Chi Square = 116 with prob < 4.1e-11   
##   
## Tucker Lewis Index of factoring reliability = 0.97  
## RMSEA index = 0.069 and the 90 % confidence intervals are 0.055 0.083  
## BIC = -91  
## Fit based upon off diagonal values = 1  
## Measures of factor score adequacy   
## ML1 ML2 ML3  
## Correlation of (regression) scores with factors 0.90 0.94 0.87  
## Multiple R square of scores with factors 0.80 0.88 0.76  
## Minimum correlation of possible factor scores 0.61 0.76 0.52

# We can see that RMSR as well as various other goodness of fit indices indicate that this model is probably a good fit for our data.

### Section 4.1

# This diagram shows our model  
  
fa.diagram(nd.fa)

# Some of these explanatory factors make sense with our knowledge of baseball i.e. factor 3 indicates the influence of salary <-> home\_run <-> intentional\_walk  
# We now continue to CFA.  
  
baseball\_model <- specifyModel(text = "  
Slow -> double , lambda1, NA  
Slow -> ground\_into\_double\_play , lambda2, NA  
Slow -> sacrifice\_fly , lambda3, NA  
Slow -> hit\_by\_pitch , lambda4, NA  
Fast -> caught\_stealing, lambda5, NA  
Fast -> stolen\_base , lambda6, NA  
Fast -> triple , lambda7, NA  
Big -> intentional\_walk , lambda8, NA  
Big -> walk , lambda9, NA  
Big -> salary , lambda10, NA  
double <-> double , psi1 , NA  
ground\_into\_double\_play <-> ground\_into\_double\_play , psi2 , NA  
sacrifice\_fly <-> sacrifice\_fly , psi3 , NA  
hit\_by\_pitch <-> hit\_by\_pitch , psi4 , NA  
caught\_stealing <-> caught\_stealing, psi5 , NA  
stolen\_base <-> stolen\_base , psi6 , NA  
triple <-> triple , psi7 , NA  
intentional\_walk <-> intentional\_walk , psi8 , NA  
walk <-> walk , psi9 , NA  
salary <-> salary , psi10 , NA  
Slow <-> Slow, NA , 1  
Fast <-> Fast, NA , 1  
Big <-> Big, NA , 1  
Slow <-> Fast, phi12 , NA  
Fast <-> Big, phi13 , NA  
Big <-> Slow, phi14 , NA")

## NOTE: it is generally simpler to use specifyEquations() or cfa()  
## see ?specifyEquations

options(fit.indices = c("GFI", "AGFI", "SRMR"))  
  
baseball.sem = sem(baseball\_model, cor(efaData), nrow(efaData))  
  
baseball.sem.smy = summary(baseball.sem)  
  
round(baseball.sem.smy$GFI, 2)  
round(baseball.sem.smy$AGFI, 2)  
round(baseball.sem.smy$SRMR, 2)  
  
baseball.sem.smy$coeff

### Section 4.2

# CFA model plot with disturbance values   
library(semPlot)  
semPaths(baseball.sem, rotation = 2, 'est', edge.label.cex = 1.2)