# Data Analysis #1

*Dustin Leatherman*

*April 18, 2019*

```
# PUTTING THESE HERE INSTEAD OF A SEPARATE FILE FOR VISIBILTY

# pca_analysis
# Parameters:
# dataset - data.frame containing all items that will be run against pca
# type - enumeration for "original" or "standard". orginal applies pca to the orginal values while stan
# returns a tibble containing the original data, a PCA object, and an augmented set of the data with fi
pca_analysis <- function(dataset, type) {
  inputDataset <- dataset
  if(type == "original") {
    inputDataset %>%
      # calculate mean eigenvalue from covariance matrix to determine floor for principle component sel
      select_if(is.numeric) %>%
      mutate(avg_eigen = eigen(var(.))$values %>% mean) %>%
      # merge back the original data into data
      bind_cols(inputDataset %>% select_if(is.factor)) %>%
      nest() %>%
      mutate(
        pca = map(data, ~ prcomp(.x %>% select_if(is.numeric) %>%  select(-avg_eigen))),
        # add values from pca output onto original data
        pca_aug = map2(pca, data, ~augment(.x, data = .y))
      )
  } else if(type == "standard") {
    inputDataset %>%
      # calculate mean eigenvalue from correlation matrix to determine floor for principle component se
      select_if(is.numeric) %>%
      mutate(avg_eigen = eigen(cor(.))$values %>% mean) %>%
      # merge back the original data into data
      bind_cols(inputDataset %>% select_if(is.factor)) %>%
      nest() %>%
      mutate(
        pca = map(data, ~ prcomp(.x %>% select_if(is.numeric) %>% select(-avg_eigen), scale. = TRUE, cen
        # add values from pca output onto original data
        pca_aug = map2(pca, data, ~augment(.x, data = .y))
      )
  }
}

# pca_tidy
# Parameters:
# pca_analysis - a tibble returned from the pca_analysis function
# Returns: a tibble containing summarized pca information
pca_summary <- function(pca_analysis) {
  pca_analysis %>%
    unnest(pca_aug) %>%
    # calculate variance for all PC variables
    summarize_at(.vars = vars(contains("PC")), .funs = funs(var)) %>%
```

```
    # pivot columns to rows
    gather(
      key = pc,
      value = variance
    ) %>%
    mutate(
      # variance explained
      var_exp = variance / sum(variance),
      # cumulative sum of variance explained so far
      cum_var_exp = cumsum(var_exp),
      # name of principle component
      pc = str_replace(pc, ".fitted", "")
    )
}

# pca_kable
# Taking all the above parameters, produce a kable which highlights Principle Components where
# the variance exceeds the average eigenvalue
pca_kable <- function(pca_analysis, pca_summary, type) {
  caption_txt <- ifelse(type == "standard", "Standardized", "Original")
  pca_summary %>%
  kable(
    digits = 4,
    col.names = c("PC", "Variance", "Var. Explained", "Cumulative  Var. Explained"),
    caption = paste(caption_txt, "Principle Components. Bolded rows indicate where variance is greater
    ) %>%
    kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position") %>%
  # bold rows that are greater than the average eigenvalue.
  row_spec(which(pca_summary$variance >= (pca_analysis$pca_aug[[1]]$avg_eigen) %>% mean), bold = T)
}

# kable_pc_list
# Parameters:
# pca_analysis - output from pca_analysis function
# type - "original"/"standard" for type of analysis
# pc_list - vector of Principle Components to retrieve. i.e. c("PC1", "PC2")
# This function is a utility function to write out the PC coefficients in a formatted way
kable_pc_list <- function(pca_analysis, type, pc_list){
  caption_txt <- ifelse(type == "standard", "Standardized", "Original")

  pca_analysis$pca[[1]]$rotation[, pc_list] %>%
  kable(
    caption = paste("Coefficients for selected ", caption_txt, " Principle Components"),
    col.names = pc_list
  ) %>%
  kable_styling("striped", full_width = FALSE, latex_options = "hold_position")
}

# return correlation for a given principle component
pc_corr <- function(dataset, pc_num, type) {
  if(type == "standard") {
    R <- cor(dataset)
    pca <- prcomp(dataset, scale. = TRUE, center = TRUE)
```

```
    lambda <- eigen(R)$values
    diag <- diag(R)
  } else if (type == "original") {
    S <- var(dataset)
    pca <- prcomp(dataset)
    lambda <- eigen(S)$values
    diag <- diag(S)
  } else {
    stop(paste("Invalid type specified:", type))
  }
  pc <- pca$rotation[,pc_num]
  pc * sqrt(lambda[pc_num] / diag)
}

# run N knn simulations for a given k
knn_n <- function(training, testing, training.grouping, testing.grouping, k, simulations) {
  output <- lapply(1:simulations, function(i) {
    knnRes <- knn(training, testing, training.grouping, k)
    confusion <- as.matrix(table(Actual = testing.grouping, Predicted = knnRes))
    err.pcnt <- 1 - sum(diag(confusion))/length(testing.grouping)

    list(res = knnRes, confusion = confusion, error.pcnt = err.pcnt)
  })
  # error.pcnt is a nested within each index so this pulls it into a list for quick summary calculation
  output$error.pcnt =  Reduce(c, Reduce(c, map(output, function(z) z$error.pcnt)))
  output
}
```

# Introduction

What does a baseball player need to do over their career to end up in the Hall of Fame? In recent years, baseball writers who have voting privileges for the Hall of Fame have been using more advanced statistics to make a case for a certain player of interest; however, it was not always that way! In the past, anyone could submit a vote to the Hall of Fame for any reason so it is possible that many Hall of Famers' aren't statistically Hall of Fame material.

This analysis gathers summary statistics and uses them to investigate the distributions of each measure in order to test assumptions required for Discriminant Analysis and Classification. Dimension reduction is explored through Principle Component Analysis to gather a smaller number of predictors. These predictors are used in Discriminant Analysis to ascertain whether or not a Hall of Fame Winner can be be predicted from this dataset.

## Background

The dataset in question represents the population of the all non-pitchers who recieved a vote for the MLB Hall of Fame. A representative subset of this will be taken for training data. Since HoF is the classification of interest, a 30% stratified sample will be used to build the training set.

## Variables of Interest

### Categorical

| Field Name | Description | Type |
|---|---|---|
| HoF | This is a "Yes"/"No" indicator on whether or not the player is in the Hall of Fame. | categorical |

### Continuous

| Field Name | Description | Type |
|---|---|---|
| Yrs | How many seasons did the player play in MLB | continuous |
| WAR | Baseball References's measure of Wins Above Replacement. This is a single number that describes the number of wins the player added to their teams over the course of their career. | continuous |
| WAR7 | The sum of the seven best seasons of WAR in the player's career. It may not be seven seasons in a row. | continuous |
| JAWS | Developed by Baseball Prospectus. It contains a combination of career and 7-year peak WAR totals allowing for comparison to average Hall of Fame players by position. | continuous |
| Jpos | The average JAWS score for all Hall of Fame players at this position plus overall Hall of Fame averages for positions with fewer inducted players. | continuous |
| JAWSRatio | JAWS divided by Jpos and multiplied by 100. | continuous |
| G | Games played during a player's career. | continuous |
| AB | at bats during a player's career. | continuous |
| R | Runs scored during a player's career. | continuous |
| H | Hits during a player's career. | continuous |
| HR | Home runs during a player's career. | continuous |
| RBI | Runs batted during a player's career. | continuous |
| SB | Stolen bases during a player's career. | continuous |
| BB | Walks during a player's career. | continuous |
| BA | Batting average. | continuous |
| OBP | On Base Percentage. This is the sum of the number of hits, walks, and times hit by a pitch divided by the sum of the number of at bats, walks, times hit by a pitch, and sacrifice flies. | continuous |
| SLG | Slugging Percentage. This is the number of bases divided by the number of at bats. Every single is one base, double is two bases, triple is three bases, and home run is four bases in the numerator of this calculation. | continuous |
| OPS | On Base Percentage plus slugging percentage | continuous |
| OPS+ | OPS adjusted to the player's ball park. 100 is an average hitter. | continuous |

## Summary Statistics

General summary statistics for each of the variables are shown below.

```
# Overall statistics
baseball %>%
  select(-Name, -HoF) %>%
  summarize_all(
    funs(
```

```
      Min=min,
      Q25 = quantile(., 0.25),
      Median=median,
      Q75 = quantile(., 0.75),
      Max=max,
      Mean=mean,
      Stdev = sd,
      N = n()
      )
    ) %>%
  gather(stat, val) %>%
  separate(stat, into = c("Variable", "stat"), sep = "_") %>%
  spread(stat, val) %>%
  select(Variable, Min, Q25, Median, Q75, Max, Mean, Stdev, N) %>%
  kable(
    digits = 4,
    caption = "Continuous Variable Summary for Hall of Fame Nominees and Winners"
  ) %>%
       kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position")
```

Table 3: Continuous Variable Summary for Hall of Fame Nominees and Winners

| Variable | Min | Q25 | Median | Q75 | Max | Mean | Stdev | N |
|---|---|---|---|---|---|---|---|---|
| AB | 186.0000 | 5013.5000 | 6390.0000 | 7792.0000 | 14053.0000 | 6403.4498 | 2075.3312 | 607 |
| BA | 0.2080 | 0.2670 | 0.2820 | 0.2975 | 0.3660 | 0.2834 | 0.0239 | 607 |
| BB | 7.0000 | 430.0000 | 623.0000 | 870.0000 | 2558.0000 | 682.7298 | 358.0039 | 607 |
| G | 67.0000 | 1441.0000 | 1806.0000 | 2145.0000 | 3562.0000 | 1792.6227 | 529.9208 | 607 |
| H | 54.0000 | 1387.0000 | 1790.0000 | 2248.0000 | 4256.0000 | 1830.4596 | 653.2518 | 607 |
| HR | 1.0000 | 47.0000 | 118.0000 | 253.0000 | 762.0000 | 166.8567 | 147.3545 | 607 |
| JAWS | -2.0000 | 21.7500 | 32.7000 | 44.3000 | 123.4000 | 34.4308 | 19.3539 | 607 |
| JAWSratio | -4.5455 | 39.6800 | 59.0494 | 81.4813 | 220.1869 | 63.0552 | 34.7478 | 607 |
| Jpos | 33.0000 | 53.5000 | 55.0000 | 57.8000 | 57.9000 | 54.3687 | 4.3026 | 607 |
| OBP | 0.2540 | 0.3320 | 0.3540 | 0.3725 | 0.4820 | 0.3534 | 0.0323 | 607 |
| OPS | 0.5290 | 0.7210 | 0.7840 | 0.8350 | 1.1640 | 0.7800 | 0.0902 | 607 |
| OPSadj | 22.0000 | 98.0000 | 113.0000 | 126.0000 | 206.0000 | 112.7512 | 22.2150 | 607 |
| R | 25.0000 | 686.0000 | 930.0000 | 1193.5000 | 2295.0000 | 958.0610 | 399.3304 | 607 |
| RBI | 24.0000 | 580.0000 | 860.0000 | 1173.5000 | 2297.0000 | 896.3855 | 407.1120 | 607 |
| SB | 2.0000 | 41.5000 | 90.0000 | 201.0000 | 1406.0000 | 151.2768 | 164.6460 | 607 |
| SLG | 0.2560 | 0.3805 | 0.4270 | 0.4695 | 0.6900 | 0.4267 | 0.0657 | 607 |
| WAR | -5.3000 | 22.8500 | 37.0000 | 52.6000 | 162.8000 | 40.0114 | 25.8353 | 607 |
| WAR7 | 0.0000 | 20.4000 | 28.4000 | 36.8000 | 84.7000 | 28.8397 | 13.2823 | 607 |
| Yrs | 4.0000 | 13.0000 | 16.0000 | 18.0000 | 27.0000 | 15.7414 | 3.6289 | 607 |

```
sum.grouped <-
  baseball %>%
  select(-Name) %>%
  group_by(HoF) %>%
  summarize_all(
    funs(
      Min=min,
      Q25 = quantile(., 0.25),
      Median=median,
```

```
    Q75 = quantile(., 0.75),
    Max=max,
    Mean=mean,
    Stdev = sd,
    N = n()
    )
  )

sum.grouped %>%
  filter(HoF == "Yes") %>%
  select(-HoF) %>%
  gather(stat, val) %>%
  separate(stat, into = c("Variable", "stat"), sep = "_") %>%
  spread(stat, val) %>%
  select(Variable, Min, Q25, Median, Q75, Max, Mean, Stdev, N) %>%
  kable(
    caption = "Hall of Fame Winner Variable Summary",
    digits = 4
  ) %>%
      kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position")
```

Table 4: Hall of Fame Winner Variable Summary

| Variable | Min | Q25 | Median | Q75 | Max | Mean | Stdev | N |
|---|---|---|---|---|---|---|---|---|
| AB | 4205.0000 | 6624.0000 | 8134.0000 | 9288.0000 | 12364.0000 | 8007.9045 | 1792.0740 | 157 |
| BA | 0.2530 | 0.2840 | 0.3030 | 0.3180 | 0.3660 | 0.3023 | 0.0241 | 157 |
| BB | 308.0000 | 650.0000 | 849.0000 | 1129.0000 | 2190.0000 | 919.7261 | 389.8591 | 157 |
| G | 1211.0000 | 1795.0000 | 2164.0000 | 2499.0000 | 3308.0000 | 2165.2420 | 480.9455 | 157 |
| H | 1161.0000 | 2048.0000 | 2386.0000 | 2839.0000 | 4189.0000 | 2419.3822 | 565.7568 | 157 |
| HR | 11.0000 | 75.0000 | 170.0000 | 361.0000 | 755.0000 | 226.8153 | 180.9208 | 157 |
| JAWS | 17.6000 | 41.9000 | 52.6000 | 61.2000 | 123.4000 | 54.5025 | 18.8944 | 157 |
| JAWSratio | 30.4498 | 76.1364 | 96.4912 | 113.2727 | 213.4948 | 99.3852 | 33.0504 | 157 |
| Jpos | 44.0000 | 54.7000 | 55.7000 | 57.8000 | 57.9000 | 54.8363 | 3.8404 | 157 |
| OBP | 0.2990 | 0.3560 | 0.3760 | 0.3950 | 0.4820 | 0.3766 | 0.0304 | 157 |
| OPS | 0.6530 | 0.7970 | 0.8370 | 0.8880 | 1.1640 | 0.8415 | 0.0862 | 157 |
| OPSadj | 82.0000 | 115.0000 | 128.0000 | 141.0000 | 206.0000 | 128.8535 | 20.8793 | 157 |
| R | 579.0000 | 1094.0000 | 1291.0000 | 1583.0000 | 2295.0000 | 1334.6369 | 349.8845 | 157 |
| RBI | 530.0000 | 952.0000 | 1209.0000 | 1529.0000 | 2297.0000 | 1236.2866 | 385.0121 | 157 |
| SB | 8.0000 | 67.0000 | 153.0000 | 330.0000 | 1406.0000 | 225.1019 | 222.7155 | 157 |
| SLG | 0.3160 | 0.4270 | 0.4620 | 0.5050 | 0.6900 | 0.4651 | 0.0650 | 157 |
| WAR | 16.2000 | 49.4000 | 63.0000 | 75.2000 | 162.1000 | 66.8739 | 26.7982 | 157 |
| WAR7 | 18.9000 | 34.6000 | 41.3000 | 47.7000 | 84.7000 | 42.1268 | 11.6400 | 157 |
| Yrs | 10.0000 | 16.0000 | 18.0000 | 20.0000 | 27.0000 | 18.0255 | 3.4566 | 157 |

```
sum.grouped %>%
  filter(HoF == "No") %>%
  select(-HoF) %>%
  gather(stat, val) %>%
  separate(stat, into = c("Variable", "stat"), sep = "_") %>%
  spread(stat, val) %>%
  select(Variable, Min, Q25, Median, Q75, Max, Mean, Stdev, N) %>%
  kable(
```

```
    caption = "Hall of Fame Nominee Variable Summary",
    digits = 4
  ) %>%
    kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position")
```

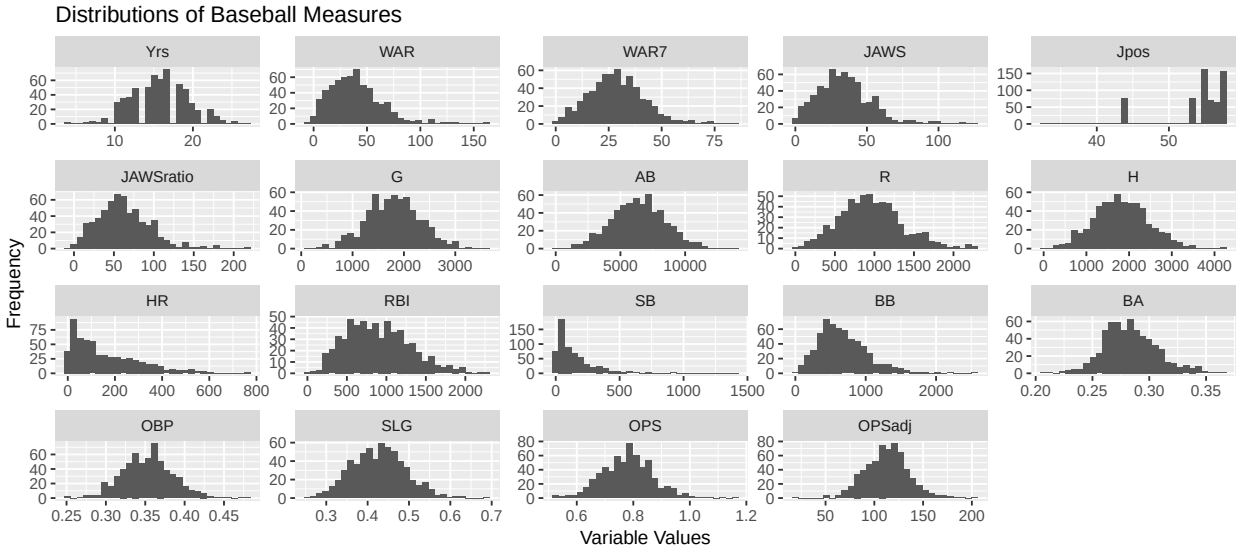
Table 5: Hall of Fame Nominee Variable Summary

| Variable | Min | Q25 | Median | Q75 | Max | Mean | Stdev | N |
|---|---|---|---|---|---|---|---|---|
| AB | 186.0000 | 4596.2500 | 5928.0000 | 7152.2500 | 14053.0000 | 5843.6733 | 1866.2843 | 450 |
| BA | 0.2080 | 0.2642 | 0.2770 | 0.2900 | 0.3560 | 0.2768 | 0.0200 | 450 |
| BB | 7.0000 | 383.7500 | 559.0000 | 792.0000 | 2558.0000 | 600.0444 | 306.0772 | 450 |
| G | 67.0000 | 1341.0000 | 1676.0000 | 1994.0000 | 3562.0000 | 1662.6200 | 482.8608 | 450 |
| H | 54.0000 | 1252.2500 | 1624.5000 | 2000.5000 | 4256.0000 | 1624.9911 | 548.7798 | 450 |
| HR | 1.0000 | 39.0000 | 105.0000 | 232.7500 | 762.0000 | 145.9378 | 127.4271 | 450 |
| JAWS | -2.0000 | 17.2250 | 26.9500 | 36.2750 | 117.8000 | 27.4280 | 13.8381 | 450 |
| JAWSratio | -4.5455 | 31.2608 | 50.3999 | 65.3642 | 220.1869 | 50.3800 | 25.0509 | 450 |
| Jpos | 33.0000 | 53.5000 | 55.0000 | 57.0000 | 57.9000 | 54.2056 | 4.4450 | 450 |
| OBP | 0.2540 | 0.3260 | 0.3450 | 0.3640 | 0.4440 | 0.3453 | 0.0288 | 450 |
| OPS | 0.5290 | 0.7042 | 0.7600 | 0.8108 | 1.0510 | 0.7586 | 0.0813 | 450 |
| OPSadj | 22.0000 | 94.0000 | 109.0000 | 120.0000 | 182.0000 | 107.1333 | 19.8068 | 450 |
| R | 25.0000 | 618.2500 | 834.5000 | 1035.2500 | 2227.0000 | 826.6778 | 325.2769 | 450 |
| RBI | 24.0000 | 514.0000 | 728.5000 | 1027.7500 | 1996.0000 | 777.7978 | 343.0733 | 450 |
| SB | 2.0000 | 38.0000 | 80.5000 | 171.7500 | 752.0000 | 125.5200 | 129.5485 | 450 |
| SLG | 0.2560 | 0.3690 | 0.4110 | 0.4548 | 0.6070 | 0.4133 | 0.0605 | 450 |
| WAR | -5.3000 | 17.6000 | 29.1000 | 41.1750 | 162.8000 | 30.6393 | 17.6349 | 450 |
| WAR7 | 0.0000 | 16.8000 | 24.4500 | 31.0750 | 72.7000 | 24.2040 | 10.3808 | 450 |
| Yrs | 4.0000 | 13.0000 | 15.0000 | 17.0000 | 24.0000 | 14.9444 | 3.3409 | 450 |

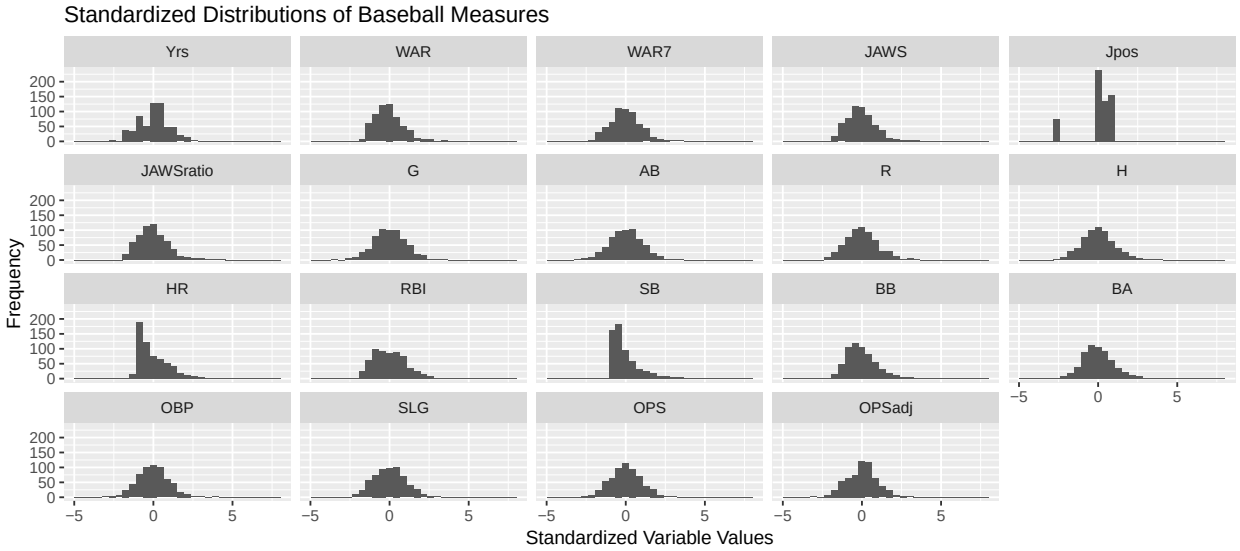
### Distributions

Are the variables normally distributed?

```
# Histogram matrix with free scales
melt(baseball) %>%
   ggplot(aes(value)) +
    geom_histogram() +
    facet_wrap(~variable, scales = "free") +
    xlab("Variable Values") +
    ylab("Frequency") +
    ggtitle("Distributions of Baseball Measures")
```




```
# Standardized Histogram matrix
baseball %>%
  select(-Name, -HoF) %>%
  scale %>%
  melt %>%
  ggplot(aes(value)) +
    geom_histogram() +
    facet_wrap(~Var2) +
    xlab("Standardized Variable Values") +
    ylab("Frequency") +
    ggtitle("Standardized Distributions of Baseball Measures")
```




While a handful appear to be normally distributed, a Shapiro-Wilk Test indicates that there is convincing evidence that all variables except for **G**, **AB**, and **H** are non-normal. Additionally though not picture, excluding the columns with negative values (JAWS, JAWSratio, and WAR) and applying a log transformation does not yield differing results.

```r
# Run shapiro-wilk tests for all columns and display to table, highlighting variables with low p-values
meltedRows <-
  baseball %>%
  select(-Name, -HoF) %>%
  apply(., 2, shapiro.test) %>%
  map(function(res) res$p.value) %>%
  melt %>%
  select(Variable = L1, Value = value)

meltedRows %>%
  kable(
    caption = "Shapiro-Wilk Tests for Baseball Measures",
    digits = 4
  ) %>%
  kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position") %>%
  row_spec(which(meltedRows$Value <= 0.05), bold = T)
```

Table 6: Shapiro-Wilk Tests for Baseball Measures

| Variable | Value |
|---|---|
| **Yrs** | **0.0012** |
| **WAR** | **0.0000** |
| **WAR7** | **0.0000** |
| **JAWS** | **0.0000** |
| **Jpos** | **0.0000** |
| **JAWSratio** | **0.0000** |
| G | 0.9678 |
| AB | 0.9921 |
| **R** | **0.0001** |
| H | 0.0538 |
| **HR** | **0.0000** |
| **RBI** | **0.0000** |
| **SB** | **0.0000** |
| **BB** | **0.0000** |
| **BA** | **0.0002** |
| **OBP** | **0.0319** |
| **SLG** | **0.0317** |
| **OPS** | **0.0035** |
| **OPSadj** | **0.0023** |

### Equal Variances

Are there equal variance between Hall of Fame Winners and Non-Hall of Fame Nominees?

```r
meltedRows <-
  baseball %>%
  select_if(is.numeric) %>%
  apply(., 2, function(c) leveneTest(y = c, group = baseball$HoF, data = .)) %>%
  map(function(res) res$`Pr(>F)`[1]) %>%
  melt %>%
  select(Variable = L1, Value = value)
```

```
meltedRows %>%
  kable(
    caption = "Levene's Test for Equal Variance between Hall of Fame Winners and Hall of Fame Nominees"
    digits = 4
  ) %>%
  kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position") %>%
  row_spec(which(meltedRows$Value <= 0.05), bold = T)
```

Table 7: Levene's Test for Equal Variance between Hall of Fame Winners and Hall of Fame Nominees

| Variable | Value |
|---|---|
| Yrs | 0.5643 |
| **WAR** | **0.0000** |
| WAR7 | 0.3203 |
| **JAWS** | **0.0012** |
| Jpos | 0.2381 |
| **JAWSratio** | **0.0014** |
| G | 0.6672 |
| AB | 0.6749 |
| R | 0.2090 |
| H | 0.5741 |
| **HR** | **0.0000** |
| RBI | 0.0712 |
| **SB** | **0.0000** |
| **BB** | **0.0011** |
| **BA** | **0.0004** |
| OBP | 0.6644 |
| SLG | 0.6795 |
| OPS | 0.9320 |
| OPSadj | 0.7958 |

There is convincing evidence that WAR, JAWS, Jpos, JAWSratiom HR, SB, BB, and BA have non-equal variances for Hall of Fame Winners and Hall of Fame Nominees.

## Analysis

### Data Reduction Through Principle Component Analysis

With 19 continuous dimensions, can Principle Component Analysis be used to reduce dimensions while retaining a large amount of explained variance? Given that the scales for the variables are not homogenous, principle components against standardized variables should be run. The floor of the mean eigenvalue from the correlation matrix is used to determine which Principle Components should be selected use in further analysis.

```
prcomp.analysis <-
  baseball %>%
  pca_analysis(., "standard")

# get the summary statistics in tibble form
prcomp.summary <- pca_summary(prcomp.analysis)
```

```
# print out a kable showing which principle components should be selected
pca_kable(prcomp.analysis, prcomp.summary, "standard")
```

Table 8: Standardized Principle Components. Bolded rows indicate where variance is greater than the average eigenvalue. These are the chosen PCs

| PC | Variance | Var. Explained | Cumulative Var. Explained |
|---|---|---|---|
| **PC1** | **11.6034** | **0.6107** | **0.6107** |
| **PC2** | **2.6168** | **0.1377** | **0.7484** |
| **PC3** | **1.5182** | **0.0799** | **0.8283** |
| **PC4** | **1.0291** | **0.0542** | **0.8825** |
| PC5 | 0.7479 | 0.0394 | 0.9219 |
| PC6 | 0.4733 | 0.0249 | 0.9468 |
| PC7 | 0.4502 | 0.0237 | 0.9705 |
| PC8 | 0.2359 | 0.0124 | 0.9829 |
| PC9 | 0.0918 | 0.0048 | 0.9877 |
| PC10 | 0.0768 | 0.0040 | 0.9918 |
| PC11 | 0.0509 | 0.0027 | 0.9944 |
| PC12 | 0.0457 | 0.0024 | 0.9968 |
| PC13 | 0.0276 | 0.0015 | 0.9983 |
| PC14 | 0.0179 | 0.0009 | 0.9992 |
| PC15 | 0.0080 | 0.0004 | 0.9997 |
| PC16 | 0.0043 | 0.0002 | 0.9999 |
| PC17 | 0.0021 | 0.0001 | 1.0000 |
| PC18 | 0.0002 | 0.0000 | 1.0000 |
| PC19 | 0.0000 | 0.0000 | 1.0000 |

There are four principle components that can be used to explain 88.25% of the variability within the data. This significantly reduces potential variables in our models from 19 to 4. Listed below are the coefficients for the first four principle components.

```
# print coefficients for 4 principle components
kable_pc_list(prcomp.analysis, "standard", c("PC1", "PC2", "PC3", "PC4"))
```

Table 9: Coefficients for selected Standardized Principle Components

| | PC1 | PC2 | PC3 | PC4 |
|---|---|---|---|---|
| Yrs | 0.1917199 | -0.3261328 | -0.1762935 | 0.1124091 |
| WAR | 0.2744984 | -0.0375130 | 0.0615155 | 0.2050946 |
| WAR7 | 0.2707184 | 0.0252625 | 0.0695967 | 0.1598593 |
| JAWS | 0.2760978 | -0.0164217 | 0.0648268 | 0.1917946 |
| Jpos | 0.0864120 | 0.0102121 | 0.3104342 | -0.7941355 |
| JAWSratio | 0.2713002 | -0.0155594 | 0.0243229 | 0.2931694 |
| G | 0.2387916 | -0.3157802 | -0.1351448 | -0.0984047 |
| AB | 0.2408756 | -0.3101257 | -0.0450257 | -0.1572629 |
| R | 0.2706447 | -0.1528720 | 0.1107416 | -0.0972100 |
| H | 0.2560575 | -0.2277965 | 0.0543836 | -0.1219962 |
| HR | 0.2040842 | 0.1153563 | -0.4816115 | -0.1909742 |
| RBI | 0.2636774 | -0.0059832 | -0.2409501 | -0.1380877 |
| SB | 0.0953520 | -0.2797300 | 0.5196258 | 0.0625241 |
| BB | 0.2458355 | -0.0528521 | -0.0834126 | 0.0591980 |
| BA | 0.1669953 | 0.2521045 | 0.3964864 | 0.0062593 |
| OBP | 0.2060246 | 0.3032109 | 0.2549802 | 0.1425718 |
| SLG | 0.2141470 | 0.3579357 | -0.1788806 | -0.1452180 |
| OPS | 0.2294886 | 0.3687190 | -0.0384713 | -0.0557608 |
| OPSadj | 0.2321369 | 0.3253757 | 0.0307670 | 0.0098161 |

### Correlation

Determining the correlation between a Principle Component and the original variable helps show which variables are most correlated with a given Principle Component.

```
contvars <- baseball %>% select(-Name, -HoF)

# wanted to do this in a more functional way but spent enough time on it
correlations <- data.frame(
  PC1 = pc_corr(contvars, 1, "standard"),
  PC2 = pc_corr(contvars, 2, "standard"),
  PC3 = pc_corr(contvars, 3, "standard"),
  PC4 = pc_corr(contvars, 4, "standard")
)

correlations %>%
  kable(
    digits = 4
  ) %>%
   kable_styling(full_width = FALSE, bootstrap_options = "striped", latex_options = "hold_position")
```

### Interpretation

| | PC1 | PC2 | PC3 | PC4 |
|---|---|---|---|---|
| Yrs | 0.6531 | -0.5276 | -0.2172 | 0.1140 |
| WAR | 0.9350 | -0.0607 | 0.0758 | 0.2081 |
| WAR7 | 0.9222 | 0.0409 | 0.0858 | 0.1622 |
| JAWS | 0.9405 | -0.0266 | 0.0799 | 0.1946 |
| Jpos | 0.2944 | 0.0165 | 0.3825 | -0.8056 |
| JAWSratio | 0.9242 | -0.0252 | 0.0300 | 0.2974 |
| G | 0.8134 | -0.5108 | -0.1665 | -0.0998 |
| AB | 0.8205 | -0.5017 | -0.0555 | -0.1595 |
| R | 0.9219 | -0.2473 | 0.1365 | -0.0986 |
| H | 0.8722 | -0.3685 | 0.0670 | -0.1238 |
| HR | 0.6952 | 0.1866 | -0.5934 | -0.1937 |
| RBI | 0.8982 | -0.0097 | -0.2969 | -0.1401 |
| SB | 0.3248 | -0.4525 | 0.6403 | 0.0634 |
| BB | 0.8374 | -0.0855 | -0.1028 | 0.0601 |
| BA | 0.5688 | 0.4078 | 0.4885 | 0.0063 |
| OBP | 0.7018 | 0.4905 | 0.3142 | 0.1446 |
| SLG | 0.7295 | 0.5790 | -0.2204 | -0.1473 |
| OPS | 0.7817 | 0.5965 | -0.0474 | -0.0566 |
| OPSadj | 0.7907 | 0.5263 | 0.0379 | 0.0100 |

**PC1**

The coefficients for PC1 are all positive and roughly a similar magnitude (0.9 - 0.28) indicating that this represents a weighted average of the measures. The magnitudes are also positively correlated with the correlation values which further supports this.

**PC2**

The coefficients for PC2 are a mix of both positive and negative values. The positive values are dominated by BA, OPB, SLG, OPS, OPSadj and the negative values are dominated by Yrs, G, AB, H, and SB. The correlation value of the largest magnitude is associated with OPS at 0.5965 which is considered a weak correlation. At best, this principle component can be interpreted as a difference in weighted averages between BA, OPB, SLG, OPS, OPSadj and Yrs, G, AB, H, SB.

**PC3**

Akin to PC2, there are a mix of both positive and negative values. The positive values are dominated by SB, BA, Jpos, and OBP while the negative values are dominated by HR and RBI. The largest correlation value is associated with SB as 0.6403 which is considered a weak correlation. At best. this principle component can be interpreted as a difference in weighted averages between SB, BA, Jpos, OBP and HR, RBI.

**PC4**

There is a mix of positive and negative components but the Jpos coefficient dominates values on both side. Jpos has the largest correlation value with -0.8056. This principle component can interpreted as an effect of Jpos.

**Discriminant Analysis**

```
baseball.aug <- prcomp.analysis$pca_aug[[1]] %>%
   dplyr::select(Name, HoF, PC1.fit = .fittedPC1, PC2.fit = .fittedPC2, PC3.fit = .fittedPC3, PC4.fit =
# create a 30% sample for training data. The 30% is arbitrary
baseball.training <-
  stratified(baseball.aug, c("HoF"), .3)

# Names are unique so exclude the players from our training set to make up our testing set
baseball.testing <- baseball.aug %>% filter(!(Name %in% baseball.training$Name))
```

### Assumptions

Discriminant Analysis can be formed given these assumptions are met:

1. Multivariate Normality
2. Multicollinearity
3. Equal Covariance Matrices
4. Independence

### Multivariate Normality

Shapiro-Wilk Tests for each variable for both the Hall of Fame Winners and Nominee groups show that there is a mix of normal variables and non-normal variables. Since multivariate normality is defined as a linear combination of normal variables, the non-normal variables cause both groups to be considered **not** multivariate normal. This is confirmed by the results for the Mardia Test for multivariate normality.

```
mardia.res.yes <-
  baseball.aug %>%
  filter(HoF == "Yes") %>%
  dplyr::select(-Name, -HoF) %>%
  mvn(mvnTest = "mardia")

mardia.res.yes$univariateNormality %>%
  kable(
    caption = "Univariate Tests for Normality for Hall of Fame Winners",
    digits = 4
  ) %>%
  kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position") %>%
  row_spec(which(stringr::str_trim(mardia.res.yes$univariateNormality$Normality) == "YES"), bold = T)
```

Table 10: Univariate Tests for Normality for Hall of Fame Winners

| Test | Variable | Statistic | p value | Normality |
|---|---|---|---|---|
| Shapiro-Wilk | PC1.fit | 0.9667 | 0.0008 | NO |
| **Shapiro-Wilk** | **PC2.fit** | **0.9940** | **0.7616** | **YES** |
| **Shapiro-Wilk** | **PC3.fit** | **0.9885** | **0.2241** | **YES** |
| Shapiro-Wilk | PC4.fit | 0.9551 | 0.0001 | NO |

```
mardia.res.yes$multivariateNormality %>%
  kable(
    caption = "Multivariate Tests for Normality for Hall of Fame Winners",
    digits = 4
```

```
  ) %>%
  kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position") %>%
  row_spec(which(stringr::str_trim(mardia.res.yes$multivariateNormality$Result) == "YES"), bold = T)
```

Table 11: Multivariate Tests for Normality for Hall of Fame Winners

| Test | Statistic | p value | Result |
|---|---|---|---|
| Mardia Skewness | 127.700202019266 | 1.05119460276813e-17 | NO |
| Mardia Kurtosis | 2.31507371588937 | 0.0206088981874308 | NO |
| MVN | NA | NA | NO |

```
mardia.res.no <-
  baseball.aug %>%
  filter(HoF == "No") %>%
  dplyr::select(-Name, -HoF) %>%
  mvn(mvnTest = "mardia")

mardia.res.no$univariateNormality %>%
  kable(
    caption = "Univariate Tests for Normality for Hall of Fame Nominees",
    digits = 4
  ) %>%
  kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position") %>%
  row_spec(which(stringr::str_trim(mardia.res.no$univariateNormality$Normality) == "NO"), bold = T)
```

Table 12: Univariate Tests for Normality for Hall of Fame Nominees

| Test | Variable | Statistic | p value | Normality |
|---|---|---|---|---|
| **Shapiro-Wilk** | **PC1.fit** | **0.9867** | **4e-04** | **NO** |
| Shapiro-Wilk | PC2.fit | 0.9975 | 0.7507 | YES |
| Shapiro-Wilk | PC3.fit | 0.9966 | 0.4612 | YES |
| **Shapiro-Wilk** | **PC4.fit** | **0.8968** | **<0.001** | **NO** |

```
mardia.res.no$multivariateNormality %>%
  kable(
    caption = "Multivariate Tests for Normality for Hall of Fame Nominees",
    digits = 4
  ) %>%
  kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position") %>%
  row_spec(which(stringr::str_trim(mardia.res.no$multivariateNormality$Result) == "YES"), bold = T)
```

Table 13: Multivariate Tests for Normality for Hall of Fame Nominees

| Test | Statistic | p value | Result |
|---|---|---|---|
| Mardia Skewness | 290.709761151269 | 6.35086597325384e-50 | NO |
| Mardia Kurtosis | 6.02451067437034 | 1.69622005330439e-09 | NO |
| MVN | NA | NA | NO |

**Equal Covariance Matrices**

Since the multivariate normality is violated, this cannot be tested with Box's M Test. Levene's Test can be used to test equal variances for the univariate case. There is convincing evidence that PC2 and PC3 have unequal variances. Given that at least one of univariate tests have shown unequal variance, it can be assumed that the covariance matrices are also non-homogenous.

Since this assumption is not met, LDA **cannot** be used. Instead Quadratic Discriminant Analysis is appropriate.

```
pc1 <- leveneTest(baseball.aug$PC1.fit, baseball.aug$HoF, data = baseball.aug)
pc2 <- leveneTest(baseball.aug$PC2.fit, baseball.aug$HoF, data = baseball.aug)
pc3 <- leveneTest(baseball.aug$PC3.fit, baseball.aug$HoF, data = baseball.aug)
pc4 <- leveneTest(baseball.aug$PC4.fit, baseball.aug$HoF, data = baseball.aug)

data.frame(pc1 = pc1$`Pr(>F)`[1], pc2 = pc2$`Pr(>F)`[1], pc3 = pc3$`Pr(>F)`[1], pc4 = pc4$`Pr(>F)`[1]) %
  kable(
    caption = "Univariate Levene's Tests on each Principle Component",
    digits = 4
  ) %>%
    kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position")
```

Table 14: Univariate Levene's Tests on each Principle Component

| pc1 | pc2 | pc3 | pc4 |
|---|---|---|---|
| 0.5262 | 0.0015 | 0 | 0.6944 |

### Multicollinearity and Independence

Principle Components are uncorrelated and independent from each other by definition so their use satisifies both assumptions.

### A Note About SMOTE

Synthetic Minority Over-sample Technique (SMOTE) is a well known algorthim for unbalanced classification problems. SMOTE under samples the majority group while oversampling the minority group in order to create a balanced dataset. Behind the scenes, synthetic records for the minority and majority groups are created based on the KNN algorithm. This is useful for classification problems where a given dataset is unbalanced. Because SMOTE creates synthetic data based on KNN, doing so violates the independence assumption which bars its use in QDA but its effect is explored for KNN.

### Results

Despite the principle components not meeting the multivariate normality assumption, QDA still provides predictive value because we are interested in a binary classifier. Using a 30% stratified sample as our training set, QDA gives us an error rate of 14.11% when run against the test set which is the remainder of the data.

```
baseball.aug.qda <- qda(HoF ~  PC1.fit + PC2.fit + PC3.fit + PC4.fit, data = baseball.training)

confusion <- as.matrix(table(baseball.testing$HoF, predict(baseball.aug.qda, baseball.testing)$class))
err.pcnt <- 1 - sum(diag(confusion)) / length(baseball.testing$HoF)

kable(
  confusion,
```

```
  caption = "Confusion Matrix for Actual vs Predicted for QDA"
  ) %>%
  kable_styling(full_width = F, bootstrap_options = "striped", latex_options = "hold_position") %>%
  add_header_above(c(" ", "Predicted" = 2))
```

Table 15: Confusion Matrix for Actual vs Predicted for QDA

| | Predicted | |
|---|---|---|
| | No | Yes |
| No | 292 | 23 |
| Yes | 41 | 69 |

Can the error rate be improved with Cross Validation? An error rate of 14% is yielded which is a slight improvement opposed to using a training set.

```
baseball.aug.qda.cv <- qda(HoF ~  PC1.fit + PC2.fit + PC3.fit + PC4.fit, data = baseball.aug, CV = TRUE)

confusion <- as.matrix(table(baseball.aug$HoF, baseball.aug.qda.cv$class))
err.pcnt <- 1 - sum(diag(confusion)) / length(baseball.aug$HoF)

kable(
  confusion,
  caption = "Confusion Matrix for Actual vs Predicted for QDA with Cross Validation"
  ) %>%
  kable_styling(full_width = F, bootstrap_options = "striped", latex_options = "hold_position") %>%
  add_header_above(c(" ", "Predicted" = 2))
```

Table 16: Confusion Matrix for Actual vs Predicted for QDA with Cross Validation

| | Predicted | |
|---|---|---|
| | No | Yes |
| No | 428 | 22 |
| Yes | 63 | 94 |

### KNN

For comparison, knn is also run using a training dataset with a 30% stratified sample on HoF. Multiple simulations were run for k = 10 through 14 and the average error rate was taken. K was determined by the following formula:

$$K = \sqrt{min(n_i)} = \sqrt{157} \approx 13$$

Error percentages vary between executions of KNN but the percentages are generally between 13-20 percent though it can exceed this. Not pictured is the number of executions of 1000 iterations done on this dataset. The lower end of this range matches what QDA provides consistently.

**Pardon the formatting beyond this point. Unresolved technical issues arose when compiling to PDF**

```
training.pc <-
  baseball.training %>%
  dplyr::select(-Name, -HoF)
```

```
testing.pc <-
  baseball.testing %>%
  dplyr::select(-Name, -HoF)

knn14 <- knn_n(training.pc, testing.pc, baseball.training$HoF, baseball.testing$HoF, 14, 1000)
knn13 <- knn_n(training.pc, testing.pc, baseball.training$HoF, baseball.testing$HoF, 13, 1000)
knn12 <- knn_n(training.pc, testing.pc, baseball.training$HoF, baseball.testing$HoF, 12, 1000)
knn11 <- knn_n(training.pc, testing.pc, baseball.training$HoF, baseball.testing$HoF, 11, 1000)
knn10 <- knn_n(training.pc, testing.pc, baseball.training$HoF, baseball.testing$HoF, 10, 1000)

# summary statistics on error percentages
knn14.err <- knn14$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 14, Mean = value)
knn13.err <- knn13$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 13, Mean = value)
knn12.err <- knn12$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 12, Mean = value)
knn11.err <- knn11$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 11, Mean = value)
knn10.err <- knn10$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 10, Mean = value)

knn.err <- bind_rows(knn14.err, knn13.err, knn12.err, knn11.err, knn10.err)

as.data.frame(knn.err) %>%
  dplyr::select(K, Mean) %>%
  kable(
    caption = "Average Error % for 1000 KNN simulations",
    digits = 4
  ) %>%
 kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position")
```

\begin{table}[!h]

\caption{Average Error % for 1000 KNN simulations}

| K | Mean |
|---|---|
| 14 | 0.1565 |
| 13 | 0.1576 |
| 12 | 0.1528 |
| 11 | 0.1435 |
| 10 | 0.1434 |

\end{table}

Can usage of SMOTE on the fitted PCA values improve our error percentage? K is chosen based on the method described above though $n_i = 1413$ after SMOTE has been applied. It can have a significant impact on the results depending on the outcome. This is not useful if a consistent answer is needed.

```
# SMOTE proved to not be helpful in improving error percentages. It was performed after PCA since it vi

baseball.smote <- SMOTE(HoF ~ ., as.data.frame(baseball.aug), perc.over = 800, perc.under = 113, k = 13)

# create a 30% sample for training data. The 30% is arbitrary
baseball.smote.training <- stratified(baseball.smote, c("HoF"), .3)

# Names are unique so exclude the players from our training set to make up our testing set
baseball.smote.testing <- baseball.smote %>% filter(!(Name %in% baseball.smote.training$Name))

training.pc.smote <-
```

```
  baseball.smote.training %>%
  dplyr::select(-Name, -HoF)

testing.pc.smote <-
  baseball.smote.testing %>%
  dplyr::select(-Name, -HoF)

knn34.smote <- knn_n(training.pc.smote, testing.pc.smote, baseball.smote.training$HoF, baseball.smote.te
knn35.smote <- knn_n(training.pc.smote, testing.pc.smote, baseball.smote.training$HoF, baseball.smote.te
knn36.smote <- knn_n(training.pc.smote, testing.pc.smote, baseball.smote.training$HoF, baseball.smote.te
knn37.smote <- knn_n(training.pc.smote, testing.pc.smote, baseball.smote.training$HoF, baseball.smote.te
knn38.smote <- knn_n(training.pc.smote, testing.pc.smote, baseball.smote.training$HoF, baseball.smote.te
# summary statistics on error percentages
knn34.err.smote <- knn34.smote$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 34, Mean = value)
knn35.err.smote <- knn35.smote$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 35, Mean = value)
knn36.err.smote <- knn36.smote$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 36, Mean = value)
knn37.err.smote <- knn37.smote$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 37, Mean = value)
knn38.err.smote <- knn38.smote$error.pcnt %>% mean %>% as.tibble %>% mutate(K = 38, Mean = value)

knn.err.smote <- bind_rows(knn34.err.smote, knn35.err.smote, knn36.err.smote, knn37.err.smote, knn38.err
knn.err.smote %>%
  dplyr::select(K, Mean) %>%
   kable(
     caption = "Average Error % for 1000 KNN simulations - SMOTE'd Data",
     digits = 4
   ) %>%
    kable_styling(full_width = T, bootstrap_options = "striped", latex_options = "hold_position")
```

\begin{table}[!h]

\caption{Average Error % for 1000 KNN simulations - SMOTE'd Data}

| K | Mean |
|---|---|
| 34 | 0.1432 |
| 35 | 0.1431 |
| 36 | 0.1336 |
| 37 | 0.1304 |
| 38 | 0.1387 |

\end{table}

## Conclusion

Quadratic Discriminant Analysis with Cross Validation gives the best error rate with the most consistent result. The analysis was done using Principle Components in order to reduce the dimensions of the data and mitigate the effects of multicollinearity. Using standardized principle components, a Hall of Fame winner can be predicted with 85.89% accuracy. This isn't the best accuracy percentage but it is passible. Interpretation of the impact of certain predictors is difficult given that principle components were standardized and fed into the model. In terms of satisfying the main goal which was to produce the best predictive accuracy rate, that has been met.

**Final Thoughts**

Much of the data is on varying scales which means it has to be standardized for any analysis. While this is fine for interpretation for the signifance of variables, it makes it difficult to create a classic interpretation. i.e. an X increase in Y yields an increase D in Z. I pursued this analysis to provide a smaller set of predictors to improve Classification percentages. Were I to analyze this set further, I would start looking into correlations between variables to see what fields could be used directly through classification rather than abstracted through principle components.