Preparing a Dataset for Data Modelling

Candace Grant

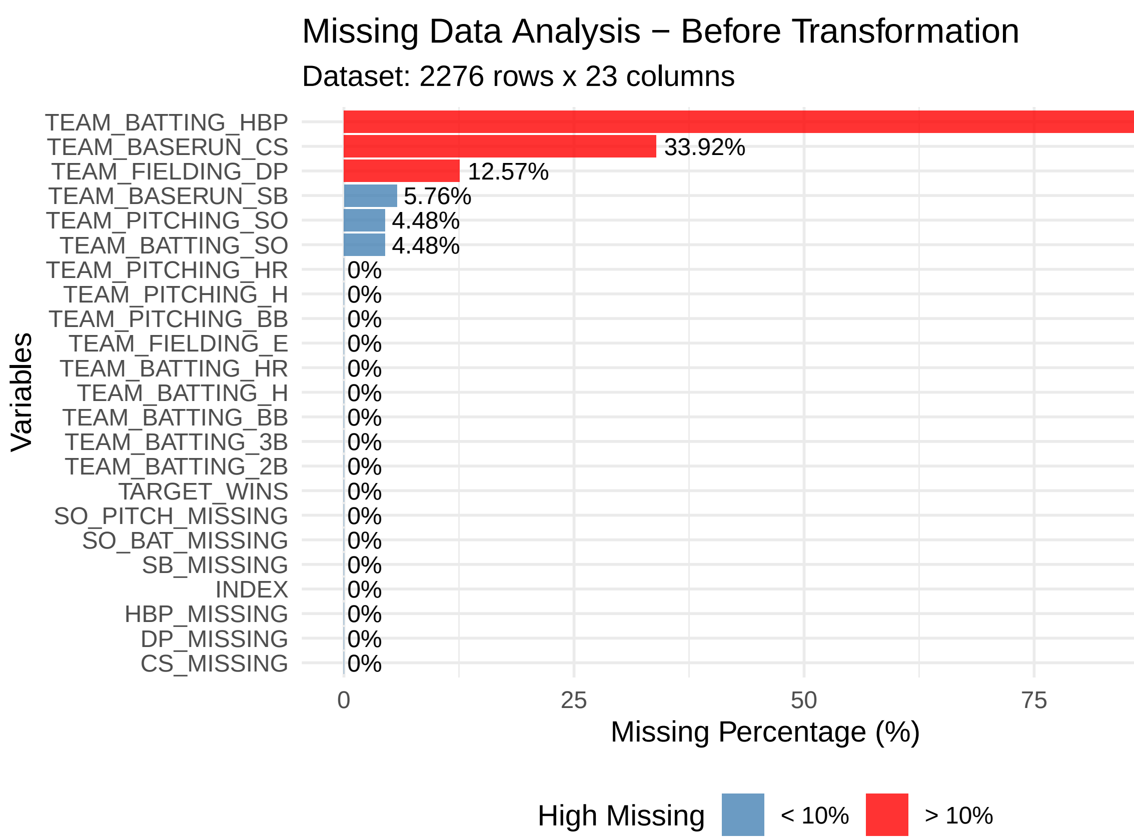
2025-09-23

```r
# Read the csv file
moneyball_training_data <- read.csv("moneyball_training_data.csv")
```

Reading the file

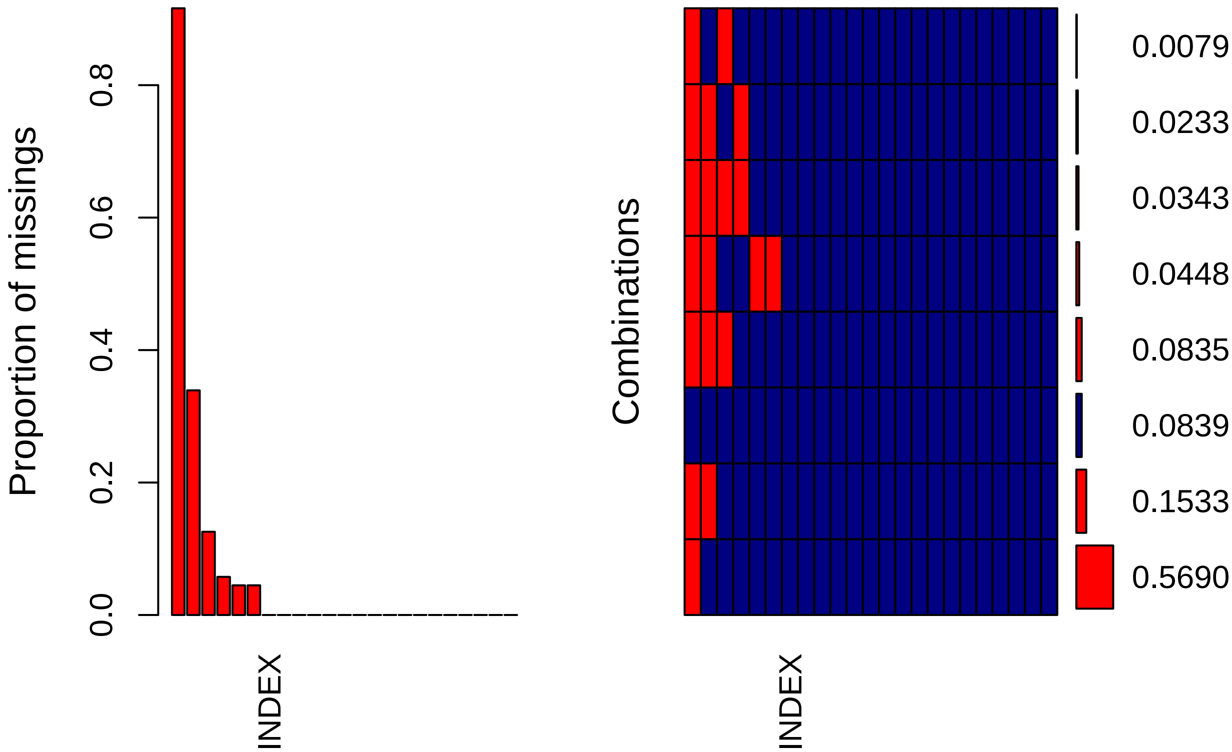
```r
# Missing values analysis function
missing_analysis <- function(df) {
  missing_summary <- data.frame(
    Column = names(df),
    Missing_Count = sapply(df, function(x) sum(is.na(x))),
    Total_Rows = nrow(df),
    Missing_Percent = round(sapply(df, function(x) sum(is.na(x))/length(x) * 100), 2)
  )
  missing_summary[order(-missing_summary$Missing_Percent), ]
}

# Missing data visualization function
plot_missing_data <- function(df, title_suffix = "") {
  missing_df <- missing_analysis(df)
  ggplot(missing_df, aes(x = reorder(Column, Missing_Percent), y = Missing_Percent)) +
    geom_col(aes(fill = Missing_Percent > 10), alpha = 0.8) +
    geom_text(aes(label = paste0(Missing_Percent, "%")),
              hjust = -0.1, size = 3) +
    scale_fill_manual(values = c("steelblue", "red"),
                      name = "High Missing", labels = c("< 10%", "> 10%")) +
    coord_flip() +
    labs(title = paste("Missing Data Analysis", title_suffix),
         subtitle = paste("Dataset:", nrow(df), "rows x", ncol(df), "columns"),
         x = "Variables", y = "Missing Percentage (%)") +
    theme_minimal() +
    theme(legend.position = "bottom")
}

# Create missing flags based on original data patterns
if("TEAM_BASERUN_CS" %in% names(moneyball_training_data)) {
  moneyball_training_data$CS_MISSING <- ifelse(is.na(moneyball_training_data$TEAM_BASERUN_CS), 1, 0)
```

```r
}
if("TEAM_FIELDING_DP" %in% names(moneyball_training_data)) {
  moneyball_training_data$DP_MISSING <- ifelse(is.na(moneyball_training_data$TEAM_FIELDING_DP), 1, 0)
}
if("TEAM_BATTING_SO" %in% names(moneyball_training_data)) {
  moneyball_training_data$SO_BAT_MISSING <- ifelse(is.na(moneyball_training_data$TEAM_BATTING_SO), 1, 0
}
if("TEAM_PITCHING_SO" %in% names(moneyball_training_data)) {
  moneyball_training_data$SO_PITCH_MISSING <- ifelse(is.na(moneyball_training_data$TEAM_PITCHING_SO), 1
}
if("TEAM_BASERUN_SB" %in% names(moneyball_training_data)) {
  moneyball_training_data$SB_MISSING <- ifelse(is.na(moneyball_training_data$TEAM_BASERUN_SB), 1, 0)
}
if("TEAM_BATTING_HBP" %in% names(moneyball_training_data)) {
  moneyball_training_data$HBP_MISSING <- ifelse(is.na(moneyball_training_data$TEAM_BATTING_HBP), 1, 0)
}

# Missing value visualization
p1_before <- plot_missing_data(moneyball_training_data, "- Before Transformation")
print(p1_before)
```



Missing value analysis

```r
# VIM aggregation plot
VIM::aggr(moneyball_training_data, col = c('navyblue','red'),
          numbers = TRUE, sortVars = TRUE)
```




```
##
##  Variables sorted by number of missings:
##          Variable      Count
##  TEAM_BATTING_HBP 0.91608084
##   TEAM_BASERUN_CS 0.33919156
##  TEAM_FIELDING_DP 0.12565905
##   TEAM_BASERUN_SB 0.05755712
##   TEAM_BATTING_SO 0.04481547
##  TEAM_PITCHING_SO 0.04481547
##             INDEX 0.00000000
##       TARGET_WINS 0.00000000
##    TEAM_BATTING_H 0.00000000
##   TEAM_BATTING_2B 0.00000000
##   TEAM_BATTING_3B 0.00000000
##   TEAM_BATTING_HR 0.00000000
##   TEAM_BATTING_BB 0.00000000
##   TEAM_PITCHING_H 0.00000000
##  TEAM_PITCHING_HR 0.00000000
##  TEAM_PITCHING_BB 0.00000000
##   TEAM_FIELDING_E 0.00000000
##        CS_MISSING 0.00000000
##        DP_MISSING 0.00000000
##    SO_BAT_MISSING 0.00000000
##  SO_PITCH_MISSING 0.00000000
##        SB_MISSING 0.00000000
##       HBP_MISSING 0.00000000
```

```r
# Function to identify outliers using IQR method
identify_outliers <- function(x) {
  Q1 <- quantile(x, 0.25, na.rm = TRUE)
  Q3 <- quantile(x, 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1
  lower <- Q1 - 1.5 * IQR
  upper <- Q3 + 1.5 * IQR
  return(x < lower | x > upper)
}

# Apply to numerical columns (excluding INDEX and missing flags)
numeric_cols <- sapply(moneyball_training_data, is.numeric)
numeric_cols <- names(numeric_cols[numeric_cols == TRUE])
numeric_cols <- numeric_cols[!grepl("INDEX|MISSING", numeric_cols)]

# Create outlier flags
for(col in numeric_cols) {
  flag_name <- paste0(col, "_OUTLIER")
  moneyball_training_data[[flag_name]] <- identify_outliers(moneyball_training_data[[col]])
}
```

Outlier Detection

```r
# Safely drop HBP columns if they exist
if("TEAM_BATTING_HBP" %in% names(moneyball_training_data)) {
  moneyball_training_data <- moneyball_training_data %>% select(-TEAM_BATTING_HBP)
  cat("Dropped TEAM_BATTING_HBP column\n")
}
```

Missing Value Imputation

```
## Dropped TEAM_BATTING_HBP column
```

```r
if("HBP_MISSING" %in% names(moneyball_training_data)) {
  moneyball_training_data <- moneyball_training_data %>% select(-HBP_MISSING)
  cat("Dropped HBP_MISSING column\n")
}
```

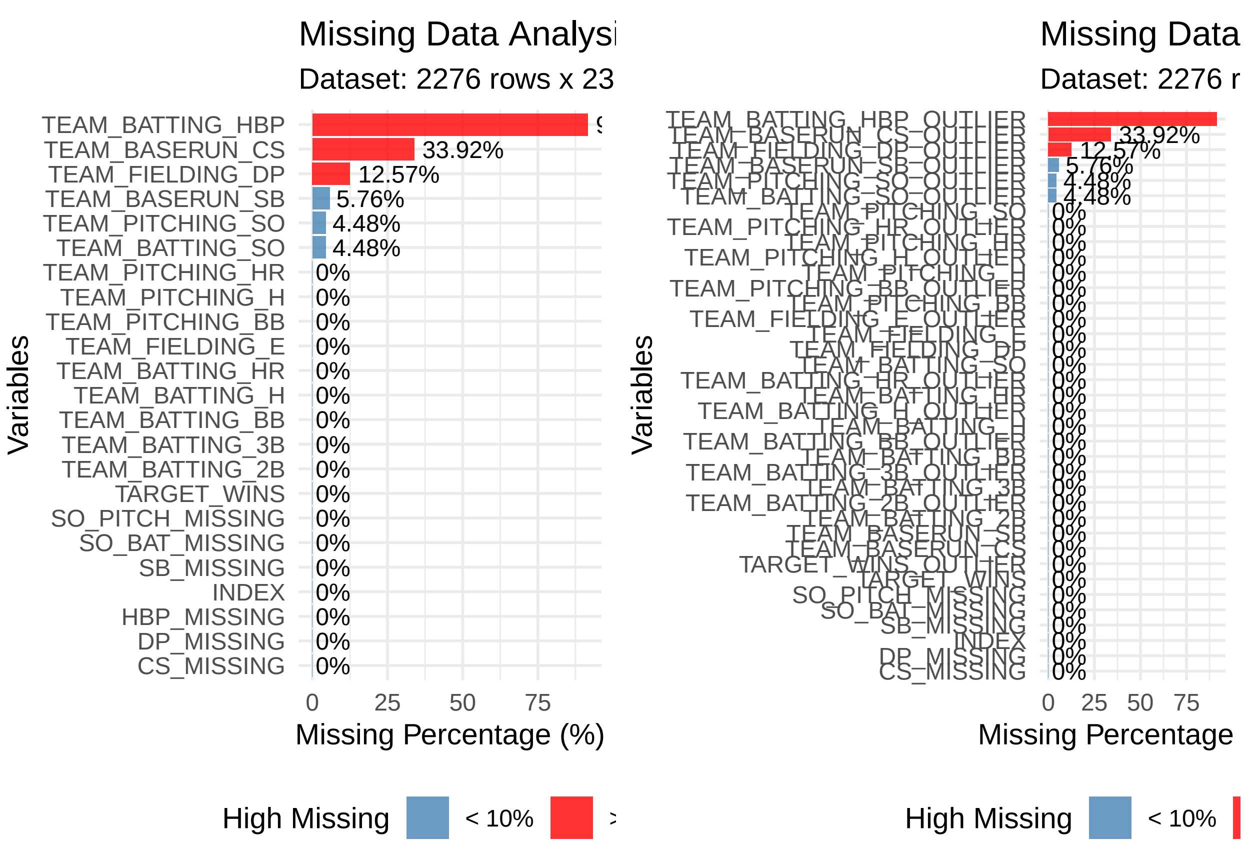
```
## Dropped HBP_MISSING column
```

```r
# Impute remaining missing values with median (more robust than mean)
numeric_impute_cols <- c("TEAM_BATTING_SO", "TEAM_BASERUN_SB", "TEAM_BASERUN_CS",
                         "TEAM_PITCHING_SO", "TEAM_FIELDING_DP")

# Filter to only columns that actually exist
numeric_impute_cols <- numeric_impute_cols[numeric_impute_cols %in% names(moneyball_training_data)]

for(col in numeric_impute_cols) {
```

```r
  missing_count_before <- sum(is.na(moneyball_training_data[[col]]))

  if(missing_count_before > 0) {
    median_val <- median(moneyball_training_data[[col]], na.rm = TRUE)
    moneyball_training_data[[col]][is.na(moneyball_training_data[[col]])] <- median_val
    missing_count_after <- sum(is.na(moneyball_training_data[[col]]))

    cat("  Imputed", missing_count_before, "missing values in", col,
        "with median:", median_val,
        "(", missing_count_after, "remaining )\n")
  } else {
    cat("• No missing values found in", col, "\n")
  }
}
```

```
##   Imputed 102 missing values in TEAM_BATTING_SO with median: 750 ( 0 remaining )
##   Imputed 131 missing values in TEAM_BASERUN_SB with median: 101 ( 0 remaining )
##   Imputed 772 missing values in TEAM_BASERUN_CS with median: 49 ( 0 remaining )
##   Imputed 102 missing values in TEAM_PITCHING_SO with median: 813.5 ( 0 remaining )
##   Imputed 286 missing values in TEAM_FIELDING_DP with median: 149 ( 0 remaining )
```

```r
# After imputation visualization
p1_after <- plot_missing_data(moneyball_training_data, "- After Imputation")
missing_comparison <- grid.arrange(p1_before, p1_after, ncol = 2)
```

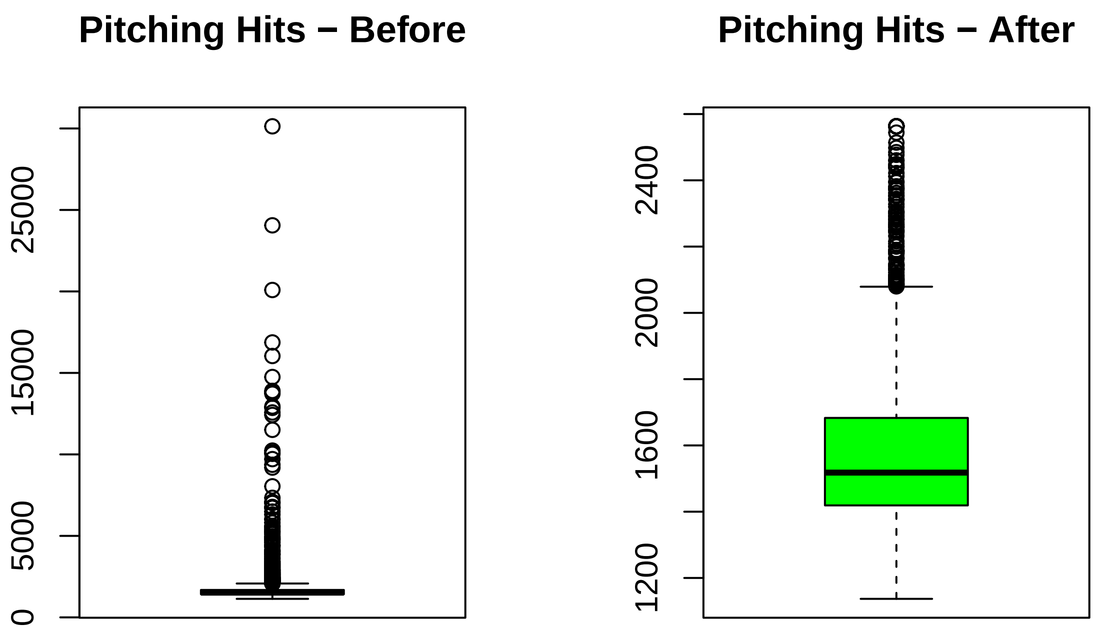


```r
# Cap extreme outliers at 95th percentile
if("TEAM_PITCHING_H" %in% names(moneyball_training_data)) {
  moneyball_training_data$TEAM_PITCHING_H_CAPPED <- pmin(moneyball_training_data$TEAM_PITCHING_H,
                                                        quantile(moneyball_training_data$TEAM_PITCHING_H, 0.95))
}

if("TEAM_FIELDING_E" %in% names(moneyball_training_data)) {
  moneyball_training_data$TEAM_FIELDING_E_CAPPED <- pmin(moneyball_training_data$TEAM_FIELDING_E,
                                                        quantile(moneyball_training_data$TEAM_FIELDING_E, 0.95))
}

cat("Outliers capped at 95th percentile for available variables\n")
```

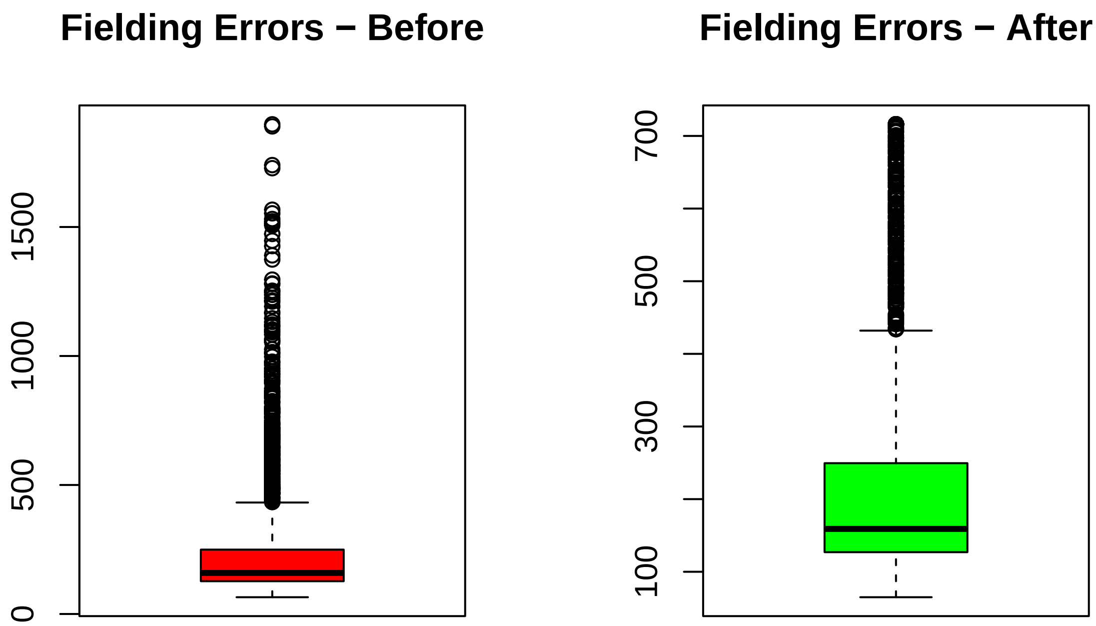
Managing Outliers

```
## Outliers capped at 95th percentile for available variables
```

```r
# Before/after visualization function
show_outlier_treatment <- function(data, original_var, capped_var, title) {
  if(original_var %in% names(data) && capped_var %in% names(data)) {
    par(mfrow = c(1, 2))
    boxplot(data[[original_var]], main = paste(title, "- Before"), col = "red")
    boxplot(data[[capped_var]], main = paste(title, "- After"), col = "green")
    par(mfrow = c(1, 1))
  }
}

# Show treatment results for available variables
show_outlier_treatment(moneyball_training_data, "TEAM_PITCHING_H", "TEAM_PITCHING_H_CAPPED", "Pitching I
```



```r
show_outlier_treatment(moneyball_training_data, "TEAM_FIELDING_E", "TEAM_FIELDING_E_CAPPED", "Fielding 
```




```r
show_outlier_treatment(moneyball_training_data, "TEAM_PITCHING_SO", "TEAM_PITCHING_SO_CAPPED", "Pitchin
```

```r
# Create meaningful baseball metrics
moneyball_training_data <- moneyball_training_data %>%
  mutate(
    # Offensive Metrics
    SINGLES = TEAM_BATTING_H - TEAM_BATTING_2B - TEAM_BATTING_3B - TEAM_BATTING_HR,
    TOTAL_BASES = SINGLES + (2 * TEAM_BATTING_2B) + (3 * TEAM_BATTING_3B) + (4 * TEAM_BATTING_HR),

    # Base running efficiency
    SB_SUCCESS_RATE = ifelse(TEAM_BASERUN_SB + TEAM_BASERUN_CS > 0,
                             TEAM_BASERUN_SB / (TEAM_BASERUN_SB + TEAM_BASERUN_CS), 0),

    # Pitching effectiveness (lower is better)
    WHIP_PROXY = (TEAM_PITCHING_H + TEAM_PITCHING_BB) / 162, # Assuming 162 games

    # Defensive efficiency
    ERROR_RATE = TEAM_FIELDING_E / (TEAM_PITCHING_H + TEAM_PITCHING_BB), # Rough proxy

    # Power metrics
    POWER_RATIO = (TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_HR) / TEAM_BATTING_H,
    HR_RATIO = TEAM_BATTING_HR / TEAM_BATTING_H,

    # Discipline metrics
    BB_SO_RATIO = TEAM_BATTING_BB / TEAM_BATTING_SO,
    PITCHING_K_BB_RATIO = TEAM_PITCHING_SO / TEAM_PITCHING_BB
  )
```

Feature Engineering

```r
# Skewness checks
skewed_vars <- moneyball_training_data %>%
  select_if(is.numeric) %>%
  select(-INDEX) %>%
  summarise_all(~abs(psych::skew(., na.rm = TRUE))) %>%
  gather(key = "Variable", value = "Skewness") %>%
  filter(Skewness > 1) %>%
  arrange(desc(Skewness))

print("Highly skewed variables (|skewness| > 1):")
```

Log Transformation of Skewed Variables

```
## [1] "Highly skewed variables (|skewness| > 1):"
```

```r
print(skewed_vars)
```

```
##                  Variable  Skewness
## 1        TEAM_PITCHING_SO 22.690450
## 2              WHIP_PROXY 10.330139
## 3         TEAM_PITCHING_H 10.329511
## 4        TEAM_PITCHING_BB  6.743899
## 5          SO_BAT_MISSING  4.397174
## 6        SO_PITCH_MISSING  4.397174
## 7              SB_MISSING  3.796854
## 8         TEAM_FIELDING_E  2.990466
## 9         TEAM_BASERUN_CS  2.602172
## 10             DP_MISSING  2.257219
## 11        TEAM_BASERUN_SB  2.065828
## 12     PITCHING_K_BB_RATIO  2.047632
## 13                SINGLES  2.046819
## 14 TEAM_FIELDING_E_CAPPED  1.784478
## 15             ERROR_RATE  1.781655
## 16 TEAM_PITCHING_H_CAPPED  1.760199
## 17         TEAM_BATTING_H  1.571333
## 18        TEAM_BATTING_3B  1.109465
## 19        TEAM_BATTING_BB  1.025760
```

```r
# Apply log transformation to highly skewed variables
for(var in skewed_vars$Variable) {
  if(min(moneyball_training_data[[var]], na.rm = TRUE) > 0) {
    new_var_name <- paste0("LOG_", var)
    moneyball_training_data[[new_var_name]] <- log(moneyball_training_data[[var]])
  }
}
```

```r
# Create performance tiers based on key metrics
moneyball_training_data <- moneyball_training_data %>%
```

```r
  mutate(
    # Offensive performance tiers
    OFFENSIVE_TIER = case_when(
      TEAM_BATTING_H >= quantile(TEAM_BATTING_H, 0.75, na.rm = TRUE) ~ "High",
      TEAM_BATTING_H >= quantile(TEAM_BATTING_H, 0.25, na.rm = TRUE) ~ "Medium",
      TRUE ~ "Low"
    ),

    # Pitching performance tiers (lower hits allowed = better)
    PITCHING_TIER = case_when(
      TEAM_PITCHING_H <= quantile(TEAM_PITCHING_H, 0.25, na.rm = TRUE) ~ "Elite",
      TEAM_PITCHING_H <= quantile(TEAM_PITCHING_H, 0.75, na.rm = TRUE) ~ "Average",
      TRUE ~ "Poor"
    ),

    # Error buckets
    ERROR_BUCKET = case_when(
      TEAM_FIELDING_E <= quantile(TEAM_FIELDING_E, 0.33, na.rm = TRUE) ~ "Low_Errors",
      TEAM_FIELDING_E <= quantile(TEAM_FIELDING_E, 0.67, na.rm = TRUE) ~ "Medium_Errors",
      TRUE ~ "High_Errors"
    )
  )

# Convert categorical variables to factors
categorical_vars <- c("OFFENSIVE_TIER", "PITCHING_TIER", "ERROR_BUCKET")
moneyball_training_data[categorical_vars] <- lapply(moneyball_training_data[categorical_vars], as.facto
```

Categorical Variables and Bucketing

```r
# Summary of final dataset
cat("\nFinal Dataset Summary:\n")
```

Data Quality Check

```
##
## Final Dataset Summary:
```

```r
cat("Dimensions:", dim(moneyball_training_data), "\n")
```

```
## Dimensions: 2276 59
```

```r
cat("Missing values remaining:", sum(is.na(moneyball_training_data)), "\n")
```

```
## Missing values remaining: 3480
```

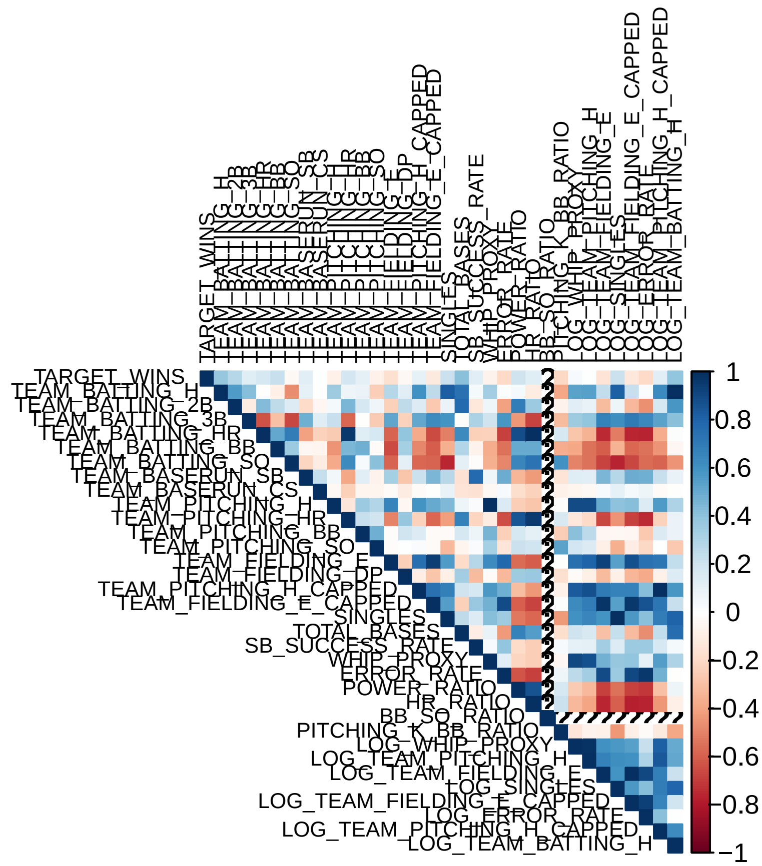
```r
# Display structure of key engineered features
engineered_features <- c("SINGLES", "TOTAL_BASES", "SB_SUCCESS_RATE", "WHIP_PROXY",
                         "POWER_RATIO", "BB_SO_RATIO", "OFFENSIVE_TIER", "PITCHING_TIER")

cat("\nEngineered Features Summary:\n")
```

```
##
## Engineered Features Summary:
```

```r
print(summary(moneyball_training_data[engineered_features]))
```

```
##     SINGLES        TOTAL_BASES   SB_SUCCESS_RATE    WHIP_PROXY
##  Min.   : 709.0   Min.   :1026   Min.   :0.0000   Min.   :  9.469
##  1st Qu.: 990.8   1st Qu.:1947   1st Qu.:0.5913   1st Qu.: 11.969
##  Median :1050.0   Median :2126   Median :0.6730   Median : 12.802
##  Mean   :1073.2   Mean   :2120   Mean   :0.6635   Mean   : 14.396
##  3rd Qu.:1129.0   3rd Qu.:2285   3rd Qu.:0.7373   3rd Qu.: 13.995
##  Max.   :2112.0   Max.   :3290   Max.   :0.9343   Max.   :194.000
##
##   POWER_RATIO       BB_SO_RATIO     OFFENSIVE_TIER PITCHING_TIER
##  Min.   :0.1134   Min.   :0.1180   High  : 569    Average:1129
##  1st Qu.:0.2366   1st Qu.:0.5450   Low   : 567    Elite  : 578
##  Median :0.2699   Median :0.6564   Medium:1140    Poor   : 569
##  Mean   :0.2694   Mean   :   Inf
##  3rd Qu.:0.3029   3rd Qu.:0.9069
##  Max.   :0.3937   Max.   :   Inf
##                   NA's   :1
```

```r
# Correlation matrix for key variables
numeric_for_corr <- moneyball_training_data %>%
  select_if(is.numeric) %>%
  select(-INDEX, -contains("MISSING"), -contains("OUTLIER"))

# Simple correlation plot without clustering
correlation_matrix <- cor(numeric_for_corr, use = "complete.obs")
corrplot(correlation_matrix, method = "color", type = "upper",
         order = "original",  # No clustering
         tl.cex = 0.7, tl.col = "black")
```



Correlation Analysis

```r
cat("\nData preparation completed successfully!\n")
```

```
##
## Data preparation completed successfully!
```

```r
cat("Ready for modeling with", ncol(moneyball_training_data), "variables and", nrow(moneyball_training_
```

```
## Ready for modeling with 59 variables and 2276 observations.
```