Predicting Baseball Pitcher's Earned Run Average (ERA)

Aaron Wu, Ruchi Tiwari, Edwin Muñoz, Shakty Juarez

December 2024

Problem Statement

The primary objective of this project is to develop a predictive regression model that accurately forecasts a baseball pitcher's Earned Run Average (ERA) based on various performance metrics. With a dataset encompassing 106 potential predictors, the challenge lies in identifying the most significant regressors that contribute meaningfully to the prediction of ERA. This study aims to:

- Enhance the understanding of factors influencing pitcher performance.
- Provide a robust analytical tool for evaluating and forecasting ERA.
- Address multicollinearity and model overfitting using advanced regression techniques.

Introduction

Baseball analytics increasingly leverages statistical methodologies to evaluate player performance and inform strategic decisions. Among key performance indicators, ERA serves as a critical measure of a pitcher's effectiveness, representing the average number of earned runs allowed per nine innings pitched. Accurate prediction of ERA is invaluable for team management, player development, and fantasy sports enthusiasts.

This project aims to construct a regression model that predicts a pitcher's ERA using a comprehensive set of performance metrics. The dataset includes variables such as:

- Strike percentage (`k_percent`)
- Walk percentage (`bb_percent`)
- Whiff percentage (`whiff_percent`)
- Weighted on-base average (`woba`)
- Expected batting average (`xba`), among others.

The study's goals were:

1. Identify the most significant predictors of ERA.
2. Develop a model balancing accuracy and interpretability.
3. Address multicollinearity and overfitting with advanced techniques.
4. Validate the model using diagnostic tests for reliability.

Key Questions:

1. Which performance metrics are the most significant predictors of ERA?
2. How do advanced regression techniques (Ridge, LASSO, Elastic Net) improve model performance?
3. How can multicollinearity issues be mitigated?
4. How well does the final model generalize to new data?

Data Collection

The analysis utilizes two primary datasets:

- **Pitching Data 2023** (`pitching_data_2023.csv`): Comprehensive pitching statistics for the 2023 season.

- **Additional Stats** (`additional_stats.csv`): Supplementary statistics for deeper insights.

Datasets were merged using the unique identifier `last_name.first_name`. Preliminary data exploration ensured proper alignment and integrity. Summary statistics and visualizations were employed to understand variable distributions, highlight outliers, and identify potential multicollinearity issues.

Data Analysis and Results

Multiple linear regression models tested specific hypotheses:

Model 1: $\texttt{p\_era} \sim \texttt{k\_percent} + \texttt{bb\_percent}$

Hypothesis: Basic stats like strikeout and walk rates affect performance.

Model 2: $\texttt{p\_era} \sim \texttt{woba} + \texttt{whiff\_percent}$

Hypothesis: Preventing strong batting and inducing misses impacts effectiveness.

Model 3: $\texttt{p\_era} \sim \texttt{player\_age} + \texttt{xba} \times \texttt{woba}$

Hypothesis: Age and expected batting performance influence ERA.

Model 4: $\texttt{p\_era} \sim \texttt{swing\_percent} + \texttt{hard\_hit\_percent} + \texttt{whiff\_percent}$

Hypothesis: Swing behavior and contact quality explain ERA.

Model 5: $\texttt{p\_era} \sim \texttt{player\_age} \times \texttt{k\_percent} + \texttt{bb\_percent}$

Hypothesis: Control, strikeouts, and age impact ERA.

Model Evaluation:

- Residual vs. fitted plots indicated no clear patterns.
- AIC values identified **Model 2** (`woba + whiff_percent`) as the best fit with the lowest AIC (91.89).
- Coefficient plots revealed `woba` as a major predictor, with minimal impact from `player_age` and `swing_percent`.

Regularization Techniques

To address multicollinearity:

- **Ridge Regression:** Stabilized coefficients and optimized λ via cross-validation.
- **LASSO Regression:** Selected key predictors by shrinking weaker coefficients to zero.
- **Elastic Net:** Combined Ridge and LASSO penalties for balance and reduced overfitting.

Diagnostics and Validation

1. **Residual Analysis:** Plots showed no significant patterns.
2. **QQ-Plot:** Confirmed normality of residuals.
3. **Influence Measures:** Identified influential outliers (e.g., Shohei Ohtani, Blake Snell).
4. **Autocorrelation:** Durbin-Watson test confirmed no autocorrelation.
5. **VIF Analysis:** Reduced multicollinearity post-regularization.

Summary and Discussion

This project successfully developed a robust regression model for predicting ERA, addressing multicollinearity and overfitting using advanced techniques. The final Elastic Net model achieved strong predictive capabilities, identifying `whiff_percent` and `woba:xba` interaction as key predictors.

Future Work

- Expand datasets to include multiple seasons.
- Incorporate additional metrics (e.g., pitch velocity, movement).

- Refine parameter tuning via granular grid search.
- Address outliers through specialized modeling approaches.

Appendix

```r
# This part of the code helps to determine which regressors out of the hundred in
# the dataset showed some level of promise in creating a regression model that could
# accurately predict pitcher ERA
library(tidyverse)
library(stats)

file_path <- "/Users/aaronwu/Desktop/stat410 copy/final/pitching_data_2023.csv"
file_path1 <- "/Users/aaronwu/Desktop/stat410 copy/final/additional_stats.csv"
data <- read.csv(file_path)
data1 <- read.csv(file_path1)
merged_data <- merge(data, data1, by = "last_name..first_name")

colnames(merged_data)
head(merged_data)

models <- list(
  lm_1 = lm(p_era ~ k_percent + bb_percent, data = merged_data),
  lm_2 = lm(p_era ~ woba + whiff_percent, data = merged_data),
  lm_3 = lm(p_era ~ player_age + xba * woba, data = merged_data),
  lm_4 = lm(p_era ~ swing_percent + hard_hit_percent + whiff_percent, data = merged_data),
  lm_5 = lm(p_era ~ player_age * k_percent + bb_percent, data =
```

```
    merged_data)
)

model_summaries <- lapply(models, summary)
model_aics <- sapply(models, AIC)
print(model_summaries)
print(model_aics)

par(mfrow = c(2, 2))
plot(models$lm_1)

coef_plots <- lapply(models, function(model) {
  coef_df <- broom::tidy(model)
  ggplot(coef_df, aes(x = term, y = estimate, fill = estimate > 0))
   +
    geom_col() +
    coord_flip() +
    labs(title = paste("Coefficients of", deparse(model$call[[2]])),
         x = "Terms",
         y = "Estimates")
})
print(coef_plots)

resid_fitted_plots <- lapply(models, function(model) {
  ggplot(model, aes(.fitted, .resid)) +
    geom_point() +
    geom_smooth(method = "lm", col = "red") +
    labs(title = paste("Residuals vs. Fitted for", deparse(model$
   call[[2]])),
         x = "Fitted values",
         y = "Residuals")
})
```

```
print(resid_fitted_plots)
```

Listing 1: Initial Data Analysis and Model Comparison

```
# This is the first model. We are trying to generate a linear
   regression model.
library(ggplot2)
library(broom)
library(dplyr)

model <- lm(p_era ~ k_percent + bb_percent + whiff_percent + woba +
   xba:woba,
            data = merged_data)
model_summary <- summary(model)
print(model_summary)

tidy_model <- tidy(model)
print(tidy_model)

ggplot(tidy_model, aes(x = term, y = estimate, fill = estimate > 0))
    +
  geom_col() +
  coord_flip() +
  labs(title = "Coefficients of the Linear Regression Model",
       x = "Terms", y = "Estimates") +
  scale_fill_manual(values = c("red", "blue"),
                    labels = c("Negative", "Positive"))
```

Listing 2: Initial Linear Regression Model

```
# Look at the VIF to determine if methods like Ridge would be useful
library(car)

model <- lm(p_era ~ k_percent + bb_percent + whiff_percent + woba +
```

```
   xba * woba,
5            data = merged_data)
6 vif_values <- vif(model)
7 print(vif_values)
8 # In this case, high VIF for woba, xba, and woba:xba, so we try
```

Listing 3: VIF Analysis

```
1 # This is a ridge prediction. Is there any other way we can optimize
     this?
2 library(glmnet)
3
4 predictors <- model.matrix(p_era ~ k_percent + bb_percent + whiff_
     percent +
5                           woba + xba * woba, data = merged_data)[,-1]
6 response <- merged_data$p_era
7 lambda_values <- 10^seq(10, -2, length = 100)
8
9 ridge_model <- glmnet(predictors, response, alpha = 0,
10                       lambda = lambda_values, standardize = TRUE)
11 cv_ridge <- cv.glmnet(predictors, response, alpha = 0,
12                        type.measure = "mse", nfolds = 10)
13 plot(cv_ridge)
14
15 best_lambda <- cv_ridge$lambda.min
16 best_ridge_model <- glmnet(predictors, response, alpha = 0,
17                            lambda = best_lambda, standardize = TRUE)
18 coef(best_ridge_model)
19 # From the results, it seems fine, but we need to check for
20 # residuals/other techniques to validate model
```

Listing 4: Ridge Regression Analysis

```
1 predicted_values <- predict(best_ridge_model, s = "lambda.min", newx
```

```
    = predictors)
residuals <- response - predicted_values

plot(residuals, type = 'p', main = "Residual Plot",
     xlab = "Predicted Values", ylab = "Residuals")
abline(h = 0, col = "red")
#Residual plot looks good, let's look at other plots
```

Listing 5: Ridge Regression Residual Analysis

```
# Trying other techniques like looking at QQ-plot, leverage points,
# autocorrelation checks, Durbin-Watson and Ljung-Box tests
library(car)
library(lmtest)
library(stats)

qqnorm(residuals)
qqline(residuals, col = "red")

influencePlot(model, id.method="identify", main="Influence Plot",
              sub="Circle size is proportional to Cook's Distance")

acf(residuals(model), main="ACF of Residuals")

dw_result <- dwtest(model)
print(dw_result)

lb_test <- Box.test(residuals(model), type = "Ljung-Box")
print(lb_test)

# Overall, all of the techniques showed that the model we created
   was good.
# Out of curiosity, I want to examine what other models look like.
```

```
# First, looking at making some of the coefficients polynomial,
# and second looking at a LASSO regression
```

Listing 6: Model Diagnostics

```
# Make some coefficients polynomial
library(stats)

poly_model <- lm(p_era ~ poly(k_percent, 2) + poly(bb_percent, 2) +
                 poly(whiff_percent, 2) + poly(woba, 2) + poly(xba, 2) +
                 poly(woba, 1)*poly(xba, 1), data = merged_data)
summary(poly_model)
plot(poly_model)

# Polynomial slightly fits the data better, but at the cost of
# overfitting/homoscedacity. Overall tradeoff is that it can capture more
# complex relationships, but is less robust for predicting on newer data
```

Listing 7: Polynomial Regression Analysis

```
# LASSO regression
library(glmnet)

predictors <- model.matrix(p_era ~ k_percent + bb_percent + whiff_percent +
                           woba + xba * woba, data = merged_data)[,-1]
response <- merged_data$p_era

set.seed(123)
lasso_model <- glmnet(predictors, response, alpha = 1,
```

```
                        lambda = 10^seq(4, -2, length = 100))
cv_lasso <- cv.glmnet(predictors, response, alpha = 1)
plot(cv_lasso)

best_lambda <- cv_lasso$lambda.min
best_lasso_model <- glmnet(predictors, response, alpha = 1, lambda =
    best_lambda)
#print(coef(best_lasso_model))

# LASSO shows that there is actually two variables, whiff_percent
   and woba: xba,
# that are causing potentially a more complex model than necessary.
# Because of that, let's try combining the LASSO and Ridge into an
# Elastic Net Regression
```

Listing 8: LASSO Regression Analysis

```
# Elastic Net Regression
library(glmnet)

predictors <- model.matrix(p_era ~ k_percent + bb_percent + whiff_
   percent +
                           woba + xba * woba, data = merged_data)
   [,-1]
response <- merged_data$p_era

set.seed(123)
cv_model <- cv.glmnet(predictors, response, alpha = 0.5, family = "
   gaussian",
                      standardize = TRUE, type.measure = "mse",
   nfolds = 10)
plot(cv_model)

```

```r
best_lambda <- cv_model$lambda.min
final_model <- glmnet(predictors, response, alpha = 0.5,
                      lambda = best_lambda, standardize = TRUE)
print(coef(final_model))

# This model looks super good. However, we need to validate it one more time.
```

Listing 9: Elastic Net Regression Analysis

```r
library(glmnet)
library(car)
library(caret)
library(Metrics)

predictions <- predict(final_model, s = best_lambda, newx = predictors)
r_squared <- cor(predictions, response)^2
residuals <- response - predictions

plot(predictions, residuals, main = "Residuals vs Fitted",
     xlab = "Fitted Values", ylab = "Residuals")
abline(h = 0, col = "red")

qqnorm(residuals)
qqline(residuals, col = "red")

standard_model <- lm(p_era ~ .,
                     data = as.data.frame(cbind(p_era = response, predictors)))
vif_values <- vif(standard_model)
print(vif_values)

```

```r
plot(predictions, sqrt(abs(residuals)), main = "Scale-Location Plot",
     xlab = "Fitted Values", ylab = "Sqrt(|Residuals|)")
abline(h = 0, col = "red")

leverage_values <- hatvalues(standard_model)
plot(leverage_values, main = "Leverage Plot")
abline(h = 2 * mean(leverage_values), col = "red", lty = 2)

cooksd <- cooks.distance(standard_model)
plot(cooksd, main = "Cook's Distance", type = "h")
threshold <- 4 / (length(response) - length(coef(standard_model)))
abline(h = threshold, col = "red", lty = 2)

cat("R-squared:", r_squared, "\n")

# Overall, this model is probably the best we will achieve. Only thing is
# potentially looking at Cook's Distance because there are 4 outliers.
```

Listing 10: Final Model Validation

```r
# I just want to check outliers
library(glmnet)
library(dplyr)

predictors <- model.matrix(p_era ~ k_percent + bb_percent + whiff_percent +
                           woba + xba * woba, data = merged_data)[,-1]
response <- merged_data$p_era

```

```r
set.seed(123)
cv_model <- cv.glmnet(predictors, response, alpha = 0.5, family = "gaussian")
best_lambda <- cv_model$lambda.min
final_model <- glmnet(predictors, response, alpha = 0.5, lambda = best_lambda)

cooks_values <- cooks.distance(lm(p_era ~ .,
                                  data = as.data.frame(cbind(p_era = response,
                                                             predictors))))
leverage_values <- hatvalues(lm(p_era ~ .,
                                data = as.data.frame(cbind(p_era = response,
                                                           predictors))))

cooks_threshold <- 4 / (nrow(predictors) - ncol(predictors) - 1)
high_cooks <- which(cooks_values > cooks_threshold)

leverage_threshold <- 2 * ncol(predictors) / nrow(predictors)
high_leverage <- which(leverage_values > leverage_threshold)

influential_points <- sort(unique(c(high_cooks, high_leverage)))
influential_data <- merged_data[influential_points, ]
print(influential_data)

plot(response, pch = 20,
     col = ifelse(seq_along(response) %in% influential_points, "red", "black"))
legend("topright", legend = c("Influential", "Not influential"),
```

```
34        col = c("red", "black"), pch = 20)
35
36 # Most of these players have a few things in common:
37 # 1. High k_percent and bb_percent like Trevor Williams, Michael
      Kopech, Luke Weaver
38 # 2. Extreme innings pitched like Shohei Ohtani, Blake Snell
39 # 3. Combination of these 2: Spencer Strider
```

Listing 11: Outlier Analysis

Self Reflections

Aaron Wu - Throughout this project, I dedicated approximately 15 hours to data collection, analysis, and report preparation. This endeavor significantly enhanced my understanding of regression modeling, particularly in handling multicollinearity and implementing regularization techniques. One major challenge was managing the extensive number of predictors, which necessitated a methodical approach to variable selection and model optimization. I learned the importance of thorough data exploration and the role of diagnostic tests in validating model assumptions. Encountering influential outliers underscored the need for vigilance in model interpretation and the consideration of exceptional cases. If I were to undertake this project again, I would allocate more time to exploring interactive visualization tools to gain deeper insights into data patterns. Additionally, incorporating domain expertise earlier in the process could guide more informed variable selection and modeling decisions. My advice to future students is to embrace a structured approach to data analysis, remain open to iterative model refinement, and prioritize understanding the underlying assumptions of each statistical method employed. Effective time management and continuous learning are key to navigating complex analytical projects successfully.

Ruchi Tiwari - Over the course of 2–3 weeks, I dedicated approximately 15 hours to the project, with much of the initial time spent identifying a suitable dataset and establishing a clear direction. Through this process, I learned that teamwork and consistent communication are critical to ensuring everyone remains aligned, especially during complex phases

of the project. A strong grasp of domain knowledge also proved essential for interpreting model results, managing high-leverage points or outliers, and making informed decisions about which variables to retain, transform, or interact within the model. I also realized the importance of evaluating multicollinearity among predictors and ensuring assumptions like linearity and homoscedasticity were met to maintain the reliability of the model. While we conducted thorough validation and testing, future iterations could benefit from additional diagnostic checks, such as cross-validation and sensitivity analyses, to strengthen confidence in the model's robustness. My key advice for future students is to start early, as the final weeks can become particularly demanding, and to review slide decks ahead of time to deepen your understanding of validation and modeling techniques before they're covered in class. Taking these steps, along with dedicating time to thoroughly explore your data and its statistical properties, can make the project process more manageable and ultimately more rewarding.

Edwin Muñoz - I spent about 15 hours working throughout the course of the project. Most of that time was spent validating the models to see which fit the problem best according to different metrics. This project taught me how to choose a new model based on the previous model's performance, which metrics to look at, what trends to look for, and how to do this all in a team. Although we were pretty happy with our result, we had a bit of a struggle dealing with the outliers', more specifically how they made it difficult to measure performance and to chose the next model. Were we to do this again, it would be good to include further diagnostic checks to reassure how well our model did. My biggest advice to future students would be to start early and to take a good look at your data before making any call, as it is easy to get behind or start on the wrong path.

Shakty Juarez - I dedicated approximately 15 hours to this project, focusing on designing and implementing the regularization techniques, including Ridge, LASSO, and Elastic Net regression. This involved parameter tuning, selecting optimal lambda values through cross-validation, and ensuring that significant predictors were retained while mitigating multicollinearity.Through this group project, I gained a deeper understanding of how regularization techniques stabilize models and improve predictive accuracy. Working with baseball

data also deepened my appreciation for how statistical methods can be applied to real-world sports analytics specifically in baseball player performance. If I were to revisit this project, I would incorporate additional data sets to potentially explore more interaction terms or nonlinear transformations to capture more complex relationships in predicting player woba. My advice to future students is to dedicate time to researching your options for advanced model validation techniques thoroughly, and find ones that fit your specific dataset the best.