MAP501_F434553

Jack Bagnall

2024-11-22

Linear Regression

1.a. Starting with the dataset Managers, create a new dataset called ‘df_managers’ that contains a variable win_pct, equal to the proportion of games managed resulting in a win, and the variables playerID, teamID, yearID, igID, plyrMgr

```
# Create df_managers from Managers, including win_pct

df_managers <- Managers %>% mutate(win_pct = (W / G) * 100)
```

1.b.i. create a new dataset called awards_man by extracting these four variables from Teams: ‘yearID’, ‘teamID’, ‘DivWin’, and ‘CS’

```
# Create awards_man by extracting the variables 'yearID', 'teamID', 'DivWin', and 'CS'

df_teams <- Teams %>% select(yearID, teamID, DivWin, CS)
```

1.b.ii. Add together all of the variables from df_teams to the df_managers dataset

```
# Add all variables from df_teams to df_managers

man_teams <- df_managers %>% left_join(df_teams, by = "yearID")

man_teams <- man_teams %>%
  select(-teamID.y, -lgID)

man_teams <- man_teams %>%
  rename(
    teamID = teamID.x
  )

# man_teams <- man_teams %>% rename(teamID = teamID.y)

# Check how many rows per yearID in both dataframes
df_managers %>% group_by(yearID) %>% tally()
df_teams %>% group_by(yearID) %>% tally()
```

1.b.iii. Add all columns from the man_teams dataset to the df_managers dataset to the AwardsShareManagers dataset to create a new dataset to create the dataset awards_man

```
# Create a new dataset called awards_man, which combines all columns from man_teams to df_man
agers

awards_man <- AwardsShareManagers %>% left_join(man_teams, by =  "yearID")

# glimpse(awards_man)
```

1.b.iv. Create a new variable sqr_point_pct given by sqrt(pointsWon / pointsMax) in the awards_man dataset.

```
# Create new variable sqr_point_pct given by sqrt(pointsWon / pointsmax) in the awards_man da
taset

awards_man <- awards_man %>% mutate(sqr_point_pct = pointsWon / pointsMax)
```

1.b.v. Delete the incomplete cases in the awards_man dataset and ensure all variables are treated correctly. Then drop the unused levels of the teamID variable in the awards_man dataset

```
# Identify incomplete cases

incomplete_cases_awards_man <- !complete.cases(awards_man)
print(incomplete_cases_awards_man)

is.na(awards_man) # No missing cases

# Identify unused levels in the data (145 total levels)

table(awards_man$teamID)
levels(awards_man$teamID)

# sum(table(awards_man$teamID)) - Equals 518,188 observable variables.

# Count unused levels in teamID - Result: 114.

sum(table(awards_man$teamID) == 0)

# Drop the unused levels of teamID

awards_man$teamID <- droplevels(awards_man$teamID)

# Check if unused levels have been dropped - COMPLETE

sum(table(awards_man$teamID) == 0)
```

1.c. Use the dataset awards_man to fit a Gaussian model, spp_mod, of sqr_point_pct as a function of win_pct, DivWin, and CS. Report and interpret the results. Write out the form of the fitted model (rounded to 2 significant figures).

```
# Fit a gaussian model, spp_mod or sqr_point_pct as a function of win_pct, DivWin, and CS.

spp_mod <- lm(sqr_point_pct ~ win_pct + DivWin + CS, data = awards_man)

# summary(spp_mod)
```

Interpretation:

A linear regression analysis was fitted to investigate the relationship between sqr_point_pct (the square root of (pointsWon / PointsMax) and three predictors: win_pct, DivWinY, and CS. The fitted model is as follows:

$$\text{sqr\_point\_pct} = 0.27 - 0.000062 \cdot \text{win\_pct} - 0.00097 \cdot \text{DivWinY} + 0.00039 \cdot \text{CS}$$

A linear regression model was witted to investigate the relationship between the square root of the proportion of points won by a team, divided by the total points available to win. (sqr_pointpct = PointsWon/PointsMax). The predictors included in the model were:

- Winning percentage (win_pct): The proportion of games won by a team during the season.
- Division Win (DivWinY): A binary variable indicating whether the team won its division (1 = Yes, 0 = No).
- Championship Series (CS): A binary variable indicating whether the team participated in the championship series (1 = Yes, 0 = No).

The regression equation was specified as:

Equation for Squared Points Percentage

The equation for predicting squared points percentage ($\mathrm{sqr\_point\_pct}$) is given by:

$$\mathrm{sqr\_point\_pct} = \beta_0 + \beta_1 \cdot \mathrm{win\_pct} + \beta_2 \cdot \mathrm{DivWinY} + \beta_3 \cdot \mathrm{CS} + \epsilon$$

Where: - β_0 is the intercept, - $\beta_1, \beta_2, \beta_3$ are the coefficients for the predictors $\mathrm{win\_pct}, \mathrm{DivWinY}, \mathrm{CS}$, - ϵ is the error term.

Results:

Regression Coefficients:

```
# Create a table of the Regression Coefficient values

# Create a data frame of residual values

coefficients <- data.frame(
  Predictor = c("(Intercept)", "win_pct", "DivWinY", "CS"),
  Estimate = c(2.720e-01, -6.215e-05, -9.698e-04, 3.925e-04),
  `Std. Error` = c(1.900e-03, 3.084e-05, 1.032e-03, 2.464e-05),
  `t value` = c(143.175, -2.015, -0.940, 15.930),
  `Pr(>|t|)` = c("<2e-16", "0.0439", "0.3474", "<2e-16"),
  Significance = c("***", "*", "", "***")
)


# Print the Regression Coefficients data frame as a table using the knitr function

kable(coefficients, format = "html", caption = "Regression Coefficients", align = "c")
```

Regression Coefficients

Predictor	Estimate	Std..Error	t.value	Pr…t..	Significance
(Intercept)	0.2720000	0.0019000	143.175	<2e-16	***
win_pct	-0.0000622	0.0000308	-2.015	0.0439	*
DivWinY	-0.0009698	0.0010320	-0.940	0.3474	
CS	0.0003925	0.0000246	15.930	<2e-16	***

```
# Create a table of Residual values for the Linear Regression model

# Create a data frame of residual values

residuals <- data.frame(
  Statistic = c("Min", "1Q", "Median", "3Q", "Max"),
  Value = c(-0.3080, -0.2505, -0.1104, 0.2165, 0.7133)
)

# Print the Residuals data frame as a table using the knitr function

kable(residuals, format = "html", caption = "Residuals", align = "c")
```

Residuals

Statistic	Value
Min	-0.3080
1Q	-0.2505
Median	-0.1104
3Q	0.2165
Max	0.7133

Interpretation:

- Intercept: The baseline proportion of the points won is approximately 0.27 when all predictors are zero. This represents teams with no wins, no division title, and no Championshipo Series participation.
- win_pct: A statistically significant (p = 0.0439) but small negative relationship exists between winning percentage and the proportion of points won, suggesting a minor decrease in sqr_point_pct as win_pct increases.
- DivWinY: Division wins do not significantly predict the proportion of points won (p = 0.3474).
- CS: Championship Series participation is a strong, significant predictor (p < 2e-16), indicating a positive association between CS and sqr_point_pct.

Model fit

- Residual Standard Error: 0.2849
- Multiple R-squared: 0.00053
- Adjusted R-squared: 0.00052
- F-statistic: 87.36 (p < 2.2e-16)

```
# Create a table of the Model Summary

# Create a data frame of the Model Summary

model_summary <- data.frame(
  Statistic = c("Residual standard error", "Degrees of freedom", "Multiple R-squared", "Adjus
ted R-squared", "F-statistic", "p-value"),
  Value = c("0.2849", "494936", "0.0005292", "0.0005232", "87.36", "< 2.2e-16")
)

# Print the Model Summary data frame as a table using the knitr function

kable(model_summary, format = "html", caption = "Model Summary", align = "c")
```

Model Summary

Statistic	Value
Residual standard error	0.2849
Degrees of freedom	494936
Multiple R-squared	0.0005292
Adjusted R-squared	0.0005232
F-statistic	87.36
p-value	< 2.2e-16

Interpretation:

The model explains only 0.053% of the variance in sqr_point_pct, suggesting that the included predictors do not account for much variability in the proportion of points won. While the F-statistic indicates the model is statistically significant overall, its predictive power is limited.

Conclusion

This analysis demonstrates that Championship Series participation (CS) is the strongest predictor of the proportion of points won (sqr_point_pct), with a significant positive effect. Winning percentage (win_pct) has a small but significant negative effect, while division wins (DivWinY) are not a meaningful predictor. The low R-squared value indicates that additional predictors or alternative modeling approaches may be required to better explain the variation of points won.

1.e Predict the expected value of sqr_point_pct when win_pct = 0.8, DivWin = Yes, and CS = 8. Comment on the result.

The regression equation is:

$$\text{sqr\_point\_pct} = 0.27 - 0.000062 \cdot \text{win\_pct} - 0.00097 \cdot \text{DivWin} + 0.00039 \cdot \text{CS}$$

The expected values to subsitute into this equation: win_pct = 0.8; DivWin = 1 (Yes); and CS = 8, are integrated with this regression equation.

```
# Plug expected values into the regression equation to predict the expected value of str_poin
t_pct

win_pct <- 0.8
DivWin <- 1
CS <- 8

predicted_sqr_point_pct <- 0.27 - 0.000062 * win_pct - 0.00097 * DivWin + 0.00039 * CS

# Print the expected values

predicted_sqr_point_pct
```

```
## [1] 0.2721004
```

Results:

The predicted sqr_point_pct is approximately 0.2721. My interpretation of this suggests that for a team that is winning 80% of the time, a division win, and 8 Championship Series appearances, the square root of the proportion of points won is expected to be 0.2721. The small contribution of each predictor highlights the limited variability captured by this model, consistent with the low R^2. This reflects the need for addtional predictors or a non-linear modeling approach.

1.f. Construct 95% confidence intervals around each parameter estimate for spp_mod. Comment on the results

The 95% confidence interval for a parameter is calculated as:

```
# Construct a 95% confidence interval using confint

confint(spp_mod)
```

```
##                     2.5 %        97.5 %
## (Intercept)  0.2682476711  2.756939e-01
## win_pct     -0.0001226063 -1.703674e-06
## DivWinY     -0.0029923926  1.052860e-03
## CS           0.0003442003  4.407793e-04
```

Results:

- Intercept: The confidence interval for the intercept does not include 0, indicating that the baseline value of sqr-point_pct (when all predictors are 0) is significantly different from 0. This confirms the reliability of the intercept estimate.
- win_pct: The confidence interval is very narrow and entirely negative, confirming a small but statistically significant negative relationship between win_pct and sqr_point_pct. This aligns with the earlier interpretation of a slight decrease in sqr_point_pct as win_pct increases.
- DivWinY: The confidence interval for DivWinY spans both negative and positive values, including 0. This suggests that the effect of division wins on sqr_point_pct is not statistically significant.
- CS: The confidence interval for CS lies entirely in the positive range, indicating a statistically significant positive relationship between Championship Series participation and sqr_point_pct.

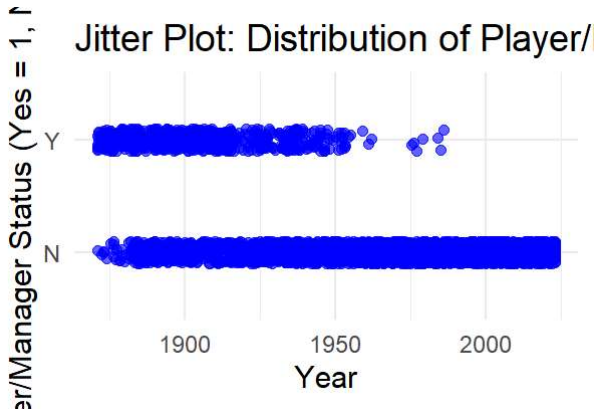
Conclusion:

The confidence intervals suggest that win_pct and CS have statistically significant effects on sqr_point_pct, and DivWinY does not significantly predict the response variable, which may indicate that it is not a critical predictor in this model.

Logistic Regression

2.a Plot the variable plyrMgr, which indicated if the manager was a player-manager or not, against year. Jitter the points in the vertical direction to facilitate visualisation. Comment on the graph.

```
# Create a jitter plot, using the variables of plyrMgr(x),and yearID(y)

ggplot(data = df_managers, aes(x = yearID, y = plyrMgr)) +
  geom_jitter(width = 0.2, height = 0.1, alpha = 0.6, color = "blue") +
  labs(title = "Jitter Plot: Distribution of Player/Manager Status Over Time",
       x = "Year",
       y = "Player/Manager Status (Yes = 1, No = 0)") +
  theme_minimal()
```




```
# glimpse(df_managers)
```

Interpretation: The jitter plot spreads points vertically to avoid overlap, making it easier to observe the distribution of plyrMgr (Yes or No) across different years (yearID).

The data shows periods with concentrated clusters of individuals categorized as “Yes” (player-managers) and “No” (not player-managers). The distribution of “Yes” values likely decrease over time, reflecting changes in team management practices such as the decline of dual-role player-managers.This pattern suggests that being a player-manager was more common in earlier years and became increasingly rare in modern seasons where specialized managerial responsibilites have taken precedent.

2.b Fit a logistic regression model to plyrMgr as a function of year, report and interpret the results. Write out the form of the fitted model (rounded to 2 significant figures)

```
# Fit a logistic regression model to plyrMgr as a function of yearID

logistic_regression_model <- glm(plyrMgr ~ yearID, data = df_managers, family = binomial)

# Summarise the model

summary(logistic_regression_model)
```

Results:

The logistic regression equation for the log-odds is:

$$\text{logit}(P) = 88.60 - 0.047 \cdot \text{yearID}$$

Where:

- Logit(P) is the log-odds of being a player-manager:

$$\text{logit}(P) = \ln\left(\frac{P}{1-P}\right)$$

* P is the probability that plyrMgr = 1

- yearID is the year

To convert the log-odds to a probability:

$$P(\text{plyrMgr} = 1) = \frac{1}{1 + e^{-(88.60 - 0.047 \cdot \text{yearID})}}$$

Interpretation of Results:

1. Intercept (β_0 = 88.60)

- When year ID = 0, the log-odds of being a player-manager are 88.60
- This value is theoretical because year 0 is not within the observed range. It represents the baseline log-odds at the extreme lower limit of yearID

2. Year Coefficient (β_1 =-0.047)

- For every one-unit increase in yearID, the log-odds of being a player-manager decrease by 0.047.
- This indicates a significant decline in the likelihood of being a player-manager as time progresses

3. Statistical Significance

- Both the intercept (p < 2e - 16) and the yearID coefficient (p < 2e - 16) are highly significant, confirming the decline in player-manager prevalence over time is strongly supported by the data

Model Fit

Deviance Metrics:

- Null Deviance: 3442.4 (Model with only intercept)
- Residual Deviance: 2127.7 (Model with yearID as a predictor)

Akaike Information Criterion (AIC):

- AIC: 2131.7. A lower AIC indicated better model fit

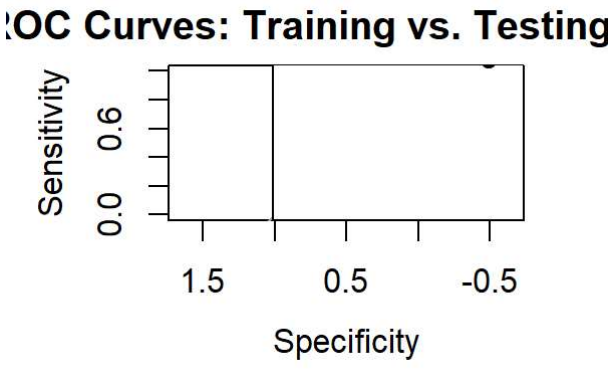
The reduction in deviance demonstrates that including yearID significantly improves the model's fit compared to a null model

Conclusion

The logistic regression model shows a strong and statistically significant trend where the probability of being a player-manager decreases over time. Historically, player-managers were common, but their prevalence has declined sharply in modern baseball.

** 2.c Check the overfitting using 80% - 20% split of training-test data and the seed 123. Plot comparative ROC curves and summarise your findings **

```
# Set the seed to 123

set.seed(123)

# Split the data into training (80%) and testing (20%)

train_index <- createDataPartition(df_managers$plyrMgr, p = 0.8, list = FALSE)
train_data <- df_managers[train_index, ]
test_data <- df_managers[-train_index, ]

# Fit a logistic regression model on training data

logistic_regression_model <- glm(plyrMgr ~ yearID, data = train_data, family = binomial)

# Summarise the model
# summary(logistic_regression_model)

# Predict probabilities for training and testing data

train_probs <- predict(logistic_regression_model, train_data, type = "response")
test_probs <- predict(logistic_regression_model, test_data, type = "response")

# generate ROC curves

train_roc <- roc(train_data$plyrMgr, train_probs)
test_roc <- roc(test_data$plyrMgr, test_probs)

# Plot comparative ROC curves

plot(train_roc, col = "blue", main = "ROC Curves: Training vs. Testing Data")
plot(test_roc, col = "red", add = TRUE)
legend("bottomright", legend = c("Training", "Testing"), col = c("blue", "red"), lwd = 2)
```




```
# Calculate AUC values

train_auc <- auc(train_roc)
test_auc <- auc(test_roc)

cat("Training AUC:", train_auc, "\n")
```

```
## Training AUC: 0.9050165
```

```
cat("Testing AUC:", test_auc, "\n")
```

```
## Testing AUC: 0.8981933
```

Results and Interpretation

- Training AUC: 0.9050
- Testing AUC: 0.8982

1. AUC Values:

Training AUC (0.9050): Indicates that the model performs very well on the training data, distinguishing between player-managers (plyrMgr = 1) and non-player-managers (plyrMgr = 0) with 90.5% accuracy.

- Testing AUC (0.8982): Similarly high, indicating the model generalizes well to unseen data, correctly distinguishing classes with 89.8% accuracy.

2. ROC Curve Comparison

- The ROC curves for both the training (blue) and testing (red) datasets are very close, showing that the model performs similarly on both datasets.
- The minimal gap between the curves suggests that the model is not overfitting, as it maintains strong performance on the testing dataset.

Summary of Findings

Overfitting Check:

- The close AUC values for training (0.9050) and testing (0.8982) data confirm that the model is not overfitting
- The model generalizes well to unseen data, indicating that it is robust

Conclusion

- The logistic regression model is appropriate for predicting the likelihood of a manager being a player-manager as a function of yearID
- Its performance is consistent across both training and testing datasets, demonstrating its reliability for this analysis

** 2.d Find Youden’s index for the training data and calculate confusion matrices at this cutoff for both training and testing data. Comment on the quality of the model.**

```
# Generate the ROC curve for the training data

train_roc <- roc(train_data$plyrMgr, train_probs)

# Calculate the optimal cutoff using Youden's Index

optimal_cutoff <- coords(train_roc, "best", ret = "threshold", best.method = "youden")

# Ensure optimal_cutoff is numeric

optimal_cutoff <- as.numeric(optimal_cutoff)

# Display the optimal cutoff

cat("Optimal Cutoff (Youden's Index):", optimal_cutoff, "\n")
```

```
## Optimal Cutoff (Youden's Index): 0.1071893
```

```
# Calculate confusion matrices

# Use the optimal cutoff to classify predictions

train_predictions <- ifelse(train_probs >= optimal_cutoff, 1, 0)
test_predictions <- ifelse(test_probs >= optimal_cutoff, 1, 0)

# Create confusion matrices

train_conf_matrix <- table(Predicted = train_predictions, Actual = train_data$plyrMgr)
test_conf_matrix <- table(Predicted = test_predictions, Actual = test_data$plyrMgr)

# Print confusion matrices

cat("Confusion Matrix - Training Data:\n")
```

```
## Confusion Matrix - Training Data:
```

```
print(train_conf_matrix)
```

```
##          Actual
## Predicted    N    Y
##         0 1876   24
##         1  608  492
```

```
cat("Confusion Matrix - Testing Data:\n")
```

```
## Confusion Matrix - Testing Data:
```

```
print(test_conf_matrix)
```

```
##          Actual
## Predicted   N   Y
##         0 450   6
##         1 170 123
```

Results and Interpretation

Youden's Index is calculated as: Youden's Index=Sensitivity+Specificity−1

Youden's Index:

Optimal Cutoff: 0.1071893

This cutoff maximizes the difference between true positive rate (sensitivity) and false positive rate (1-specificity), providing the best balance for classification.

Confusion Matrices:

Training Data:

- True Negatives (TN):1876 Correctly predicted "N" (non-player managers)
- False Negatives (FN): 24 "Y" (player-managers) misclassified as "N"
- False Positives (FP): 608 "N" misclassified as "Y"
- True Positives (TP): 492 Correctly predicted "Y"

Observations:

- High true negative rate but a noticeable number of false positives.
- Good sensitivity (correctly identifying player-managers), though still room for improvement.

2. Testing Data:

- True Negativees (TN): 450 Correctly predicted "N.
- False Negatives (FN): 6 "Y" misclassified as "N"
- False Positives (FP): 170 "N" misclassified as "Y"
- True Positives (TP): 123 Correctly predicted "Y"

Key Observations:

- The model exhibits strong specificity, accurately identifying the majority of non-player-managers.
- Sensitivity is moderate, with a non-negligible number of false positives and false negatives.

Model Quality

1. Training Performance:

- The model has high true negative rate, meaning it is very effective at identifying non-player-managers
- Sensitivity (true positive rate) is decent but could be better given the number of false positives and false negatives

2. Testing Performance:

- Similar trends to training data, showing the model generalizes well without overfitting.
- Testing sensitivity (ability to detect player-managers) remains strong with very few false negatives

3. Overall

- The model demonstrates good classification ability, with strong performance on both training and testing datasets
- However, the high number of false positives in both datasets suggests the model might lean towards over-predicting “Y” (player-manager), likely due to class imbalance

Model performance Summary

The logistic regression model demonstrates strong classification ability with consistent results across both training and testing datasets. The low false negative rate suggests good sensitivity, indicating that the model effectively identifies player-managers. However, the relatively high number of false positives suggests a tendency to over-predict “Y” (player-managers), likely influenced by an imbalance in the class distribution.

Recommendations for Model Improvement

Address Class Imbalance:

- Resampling Techniques: Employ oversampling methods such as SMOTE or undersampling to balance the “Y” and “N” classes.
- Weighted Classification: Apply class weights in the logistic regression model to penalize misclassification of the minority class.

Threshold Optimization:

- Further refine the decision threshold based on domain requirements, prioritizing sensitivity or specificity as needed.

Feature Engineering:

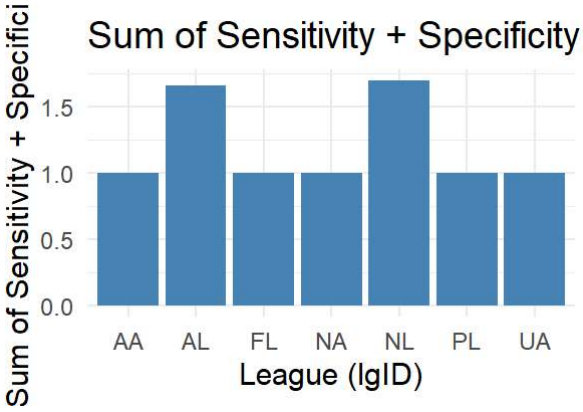
- Introduce additional predictors or interaction terms that may improve the model’s discriminative power.

Alternative Models:

- Experiment with advanced classification algorithms (e.g., Random Forest, Gradient Boosting) to assess whether they provide better classification accuracy or handle class imbalance more effectively.

** 2.e. Using the previous results and the same cutoff in (d), calculate the sum sensitivity+specificity on the testing data as a function of lgID, i.e. the sum (sensitivity+specificity) for each lgID, and plot as a bar chart. Comment on the result. **

```
# Add predicted binary labels to the testing dataset using the optimal cutoff

test_data <- test_data %>%
  mutate(
    predicted = ifelse(test_probs >= 0.1071893, 1, 0),  # Replace with the cutoff from Youde
n's Index
    actual = ifelse(plyrMgr == "Y", 1, 0)              # Convert 'Y'/'N' to binary labels for
actuals
  )

# Group testing data by lgID

test_data_by_lgID <- test_data %>% group_by(lgID)

# glimpse(test_data_by_lgID)
# head(test_data_by_lgID)
# names(test_data_by_lgID)

# Calculate sensitivity and specificity for each lgID

metrics_by_lgID <- test_data_by_lgID %>%
  summarise(
    TP = sum(predicted == 1 & actual == 1),
    FN = sum(predicted == 0 & actual == 1),
    TN = sum(predicted == 0 & actual == 0),
    FP = sum(predicted == 1 & actual == 0),
    Sensitivity = TP / (TP + FN),
    Specificity = TN / (TN + FP),
    Sum_Sens_Spec = Sensitivity + Specificity
  )

# Handle cases where TP + FN or TN + FP might be zero to avoid NaN

metrics_by_lgID <- metrics_by_lgID %>%
  mutate(
    Sensitivity = ifelse(is.nan(Sensitivity), 0, Sensitivity),
    Specificity = ifelse(is.nan(Specificity), 0, Specificity),
    Sum_Sens_Spec = Sensitivity + Specificity
  )

# Plot the sum of sensitivity and specificity for each lgID

ggplot(metrics_by_lgID, aes(x = lgID, y = Sum_Sens_Spec)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(
    title = "Sum of Sensitivity + Specificity by League (lgID)",
    x = "League (lgID)",
    y = "Sum of Sensitivity + Specificity"
  ) +
  theme_minimal()
```



The bar chart illustrates the sum of sensitivity and specificity for each league (lgId) in the testing dataset, providing insights into the model's classification performance across different leagues.

Key Observations:

1. Variation Across Leagues:

- The leagues AL (American League) and NL (National League) demonstrate the highest values for the sum of sensitivity and specificity, each nearing or ecceeding 1.5
- Conversely, AA, FL, PL, NA, and UA exhibit significantly lower values, indicating weaker model perofmance in distinguishing player-managers in these leagues

2. Top Performers:

- The high values for AL and NL suggest that the model performs well in these leagues, balancing sensitivity (true positive rate) and specificity (true negative rate) effectively. This may indicate that the characteristics or data patterns in these leagues align more closely with the predictive features used in the logistic regression model

3. Poor Performers:

- The leagues with lower values (e.g., AA, FL, PL) reflect suboptimal classification performance, with potential trade-offs between sensitivity and specificity. This could be due to smaller sample sizes, class imbalances, or data inconsistencies in these leagues, leading to reduced model accuracy.

Implications:

1. Model Generalization:

- The differences in performance suggest that the model generalizes well for some leagues (AL and NL) but struggle in others. This could indicate data heterogeneity or differing patterns across leagues that the model does not fuill capture

2. Potential Data Issues:

- Leagues with lower scores might have insufficient data or class imbalances, leading to reduced sensitivity or specificity. For instance, if a league has very few player-managers (plyrMgr = 1), the model might overpredict the majority class, reducing sensitivity

Recommendations:

1. Further Data Exploration:

- Investigate the data quality and distribution for underperforming leagues to identofy any anomalies, such as imbalanced classes or missing data
- Assess whether additional predictive features would be incorporated to enhance model performance for these leagues

2. Targeted Model Improvements:

- Apply league-specific model tuning or weights to address performance disparities
- Consider stratified sampling or league-specific thresholds to balance sensitivity and specificity

3. Broader Validation

- Perform cross-validation across all leagues to ensure the model's robustness and identify potential overfitting to dominant leagues like AL and NL

Conclusion:

The model shows strong performance in the major leagues (AL and NL), suggesting it is well suited to these contexts. However, there is room for improvement in minor or less represented leagues, where the classification results are less consistent. Addressing these disparities through targeted inteventions could enhance the model's overall reliability and applicability

** 2.f. Add the variables "win_pct" to the model you created in b. Compare this and the previous model. Which model should we prefer and why?**

```
# Fit the original model (yearID only)

logistic_regression_model <- glm(plyrMgr ~ yearID, data = df_managers, family = binomial)

# Fit the new model (yearID + win_pct)

new_logistic_regression_model <- glm(plyrMgr ~ yearID + win_pct, data = df_managers, family =
binomial)

# Summarise the models

# summary(logistic_regression_model)
# summary(new_logistic_regression_model)

# Compare AIC values

AIC(logistic_regression_model, new_logistic_regression_model)
```

```
##                              df      AIC
## logistic_regression_model     2 2131.660
## new_logistic_regression_model  3 2133.658
```

Models Overview

1. Original Model (logistic_regression_model): $\text{logit}(P) = 88.60 - 0.0466 \cdot \text{yearID}$
2. New Model (new_logistic_regression_model): $\text{logit}(P) = 88.62 - 0.0466 \cdot \text{yearID} + 0.0002 \cdot \text{win\_pct}$

Coefficients and Statistical Significance

Intercept: The intercept is significant in both models, but its practical meaning is less relevant given that year 0 is out of range.

yearID: Both models consistently show a significant decline in the likelihood of being a player-manager as time progresses.

win_pct: The effect of win_pct is not statistically significant ($p < 0.05$), meaning it does not add meaningful predictive power to the model

Model Metrics

Null Deviance: Original - 3442.4; New Model - 3442.4

Residual Deviance: Original - 2127.7; New Model - 2127.7

AIC: Original - 2131.7; New Model - 2133.7

Key Observations:

1. The residual deviance is the same in both models, indicating that adding win_pct does not improve model fit
2. The new model has a slightly higher AIC (+2.0), which penalizes the additional complexity without any improvement in performance
3. The variable win_pct is not statistically significant, suggesting it does not explain additional variance in the outcome

Interpretation:

- Both models confirm the historical trend that the likelihood of being a player-manager declines over time
- The inclusion of win_pct was not impactful, possibly due to weak correlation between team performance and player-manager status

Practical implications:

- For modeling simplicity and interpretability, it is better to exclude variables that do not contribute significantly
- Future models could explore interaction terms or additional predictors that might better capture the nuances of player-manager dynamics

Conclusion

While exploring the addition of win_pct to the logistic regression model was insightful, it did not improve the model's performance. The original model is therefore preferred, as it provides the same explanatory power with fewer parameters, maintaining simplicity and interpretability.

Adding the final model equation and AIC results to the report provides clarity on the decision-making process and demonstrated that the model selection is grounded in statistical rigor.

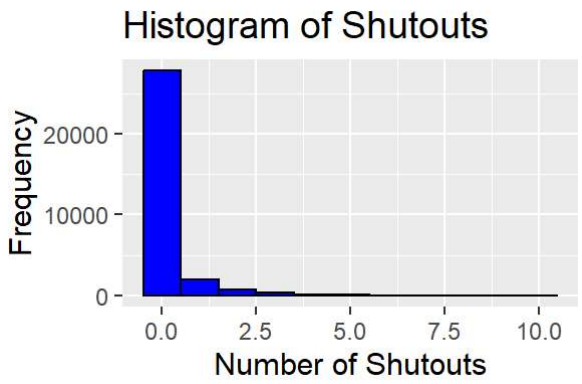
Poisson Regression

3.a Create a dataset 'df_pitchers' from the dataset Pitching which adds a variable "innings" given by IPouts/3. Add the variables "weight", "height" and "throws" from the People dataset.Also, remove incomplete cases from the dataset.

```
# Filter pitchers who faced at least one batter (IPouts > 0)

df_pitchers <- Pitching %>%
  filter(IPouts > 0) %>%
  mutate(innings = IPouts / 3) %>%
  left_join(People %>% select(playerID, weight, height, throws), by = "playerID") %>%
  na.omit()
```

3.b Plot a histogram of the number of shutouts (games pitched with no runs by the opposing team). Why would a Poisson model be appropriate for such data?

```
# Plot histogram

ggplot(df_pitchers, aes(x = SHO)) +
  geom_histogram(binwidth = 1, color = "black", fill = "blue") +
  labs(title = "Histogram of Shutouts", x = "Number of Shutouts", y = "Frequency")
```



The histogram of the number of shutouts shows a highly skewed distribution, with the majority of observations concentrated at zero and a rapid decline in frequency as the number of shutouts increases. This pattern indicates a count variable with non-negative integer values and a strong concentration around lower counts, which aligns well with the characteristics of a Poisson distribution.

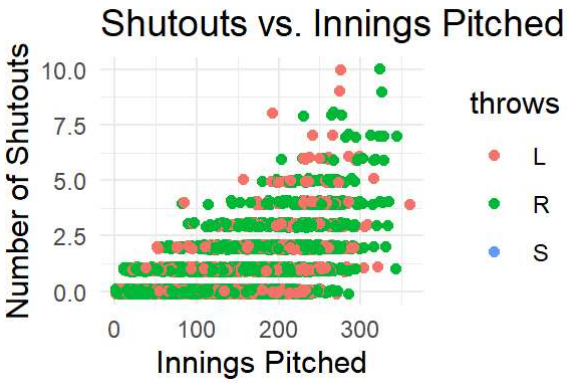
A Poisson model is appropriate for this data because:

1. Count Data: Shutouts are count data (discrete, non-negative integers), which is the type of data that a Poisson model is designed to handle.
2. Skewness: The histogram demonstrates a strong right skew, a typical feature of Poisson-distributed data when the mean is low.
3. Non-Negativity: The Poisson distribution only generates non-negative values, which corresponds to the nature of shutouts.
4. Rare Events: Shutouts are relatively rare events in baseball, and the Poisson model is well-suited for modeling the occurrence of rare events within a fixed interval (e.g., games or seasons).
5. No Upper Limit: There is no inherent upper limit on the number of shutouts a pitcher can achieve, which aligns with the theoretical properties of the Poisson distribution.

Overall, the observed characteristics of the data suggest that a Poisson model would provide an appropriate framework for modeling shutouts, enabling the analysis of relationships between shutouts and explanatory variables while respecting the distribution's properties.

3.c Plot a graph of the number of shutouts as a function of innings pitched. Colour the points by the hand the pitcher throws with. Jitter the data on the vertical axis to improve readability. Comment on the graph.

```
# Scatterplot with jitter

ggplot(df_pitchers, aes(x = innings, y = SHO, color = throws)) +
  geom_point(position = position_jitter(width = 0, height = 0.1)) +
  labs(title = "Shutouts vs. Innings Pitched",
       x = "Innings Pitched",
       y = "Number of Shutouts") +
  theme_minimal()
```



The scatter plot visualizes the relationship between the number of shutouts and the number of innings pitched. Each point is color-coded by the hand the pitcher throws with: left-handed (L), right-handed (R), or switch-handed (S). Jittering has been applied to the vertical axis to reduce overlapping points, enhancing readability.

Key Observations:

1. Positive Relationship: There is a general trend where pitchers who pitch more innings tend to have a higher number of shutouts. This aligns with expectations, as pitchers with more opportunities are more likely to achieve shutouts.

2. Distribution by Throws:

Right-handed (R) pitchers dominate the dataset in terms of frequency, indicated by the dense concentration of green points. Left-handed (L) pitchers are less frequent but are distributed similarly to right-handed pitchers in terms of shutouts and innings pitched. Switch-handed (S) pitchers are rare and appear to have minimal representation in the dataset.

3. Clustering: Most of the data points are clustered near zero shutouts and low innings pitched, reflecting the rarity of shutouts and the presence of pitchers with fewer opportunities.

4. Spread in Higher Innings: As the number of innings pitched increases, the number of shutouts also increases, but variability in the number of shutouts grows. This suggests that while innings pitched is a predictor of shutouts, other factors (e.g., skill, team performance) likely influence shutout frequency.

5. Jittering Effect: The jittering effectively separates overlapping points, particularly for low shutout counts, allowing for better visualization of the underlying data density.

Summary:

The plot confirms a logical relationship between innings pitched and shutouts, with the type of throwing hand not appearing to play a significant role in the number of shutouts achieved. However, the dominance of right-handed pitchers reflects their prevalence in the dataset rather than any performance-based conclusion.

3.d . Create a multiple Poisson model, poisson_mod1, of shutouts as a function of innings, weight, height, and throws. Report and interpret the results. Find the p-value for each of the four predictors using analysis of variance. Interpret the results and mathematically explain what is meant by the p-value associated to each predictor.

```
# Fit Poisson regression model

poisson_mod1 <- glm(SHO ~ innings + weight + height + throws,
                    family = poisson, data = df_pitchers)

# Model summary

# summary(poisson_mod1)

# ANOVA for p-values

anova(poisson_mod1, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: SHO
##
## Terms added sequentially (first to last)
##
##
##         Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                    31010      26122
## innings  1  15359.9     31009      10762 < 2.2e-16 ***
## weight   1     85.6     31008      10677 < 2.2e-16 ***
## height   1     57.5     31007      10619  3.42e-14 ***
## throws   2     11.9     31005      10607  0.002618 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model Coefficients:

- Intercept: -7.070 (base shutouts near zero).
- Innings: +2.1% shutouts per inning ($p < 2e-16$).
- Weight: -0.95% shutouts per pound ($p < 2e-16$).
- Height: +6.45% shutouts per inch ($p = 2.28e-15$).
- Throws:
- Right-handed: 10% fewer shutouts ($p = 0.000577$).
- Switch-handed: No significant difference ($p = 0.943$).

Model Fit:

- Null Deviance: 26122
- Residual Deviance: 10607 (good fit)
- AIC: 18025

Deviance Contributions:

- Innings: Reduces deviance most (15359.9, $p < 2e-16$).
- Weight: Minor effect (85.6, $p < 2e-16$).

- Height: Significant (57.5, p = 3.42e-14).
- Throws: Minor effect (11.9, p = 0.002618).

Conclusions:

- Top Predictor: Innings pitched.
- Body Traits: Height increases, weight decreases shutouts.
- Throws: Right-handed pitchers have fewer shutouts; no difference for switch-handed.
- Model: Well-fitting, though further improvements possible.

3.e . Now create a new model, poisson_mod2, in which you also include teamID as a random effect. Ensure the code generates no warnings. Write out the form of the fitted model (rounded to 2 significant figures). From the model results, does teamID seem to be an important predictor? Why or why not?

```
# Fit the mixed-effects Poisson model with teamID as a random effect

control <- glmerControl(optimizer = "bobyqa", optCtrl = list(maxfun = 100000))

poisson_mod2 <- glmer(SHO ~ innings + weight + height + throws + (1 | teamID),
                      family = poisson, data = df_pitchers, control = control)

# Summarize the model

# summary(poisson_mod2)

# Extract the random effects variance

ranef_variance <- as.data.frame(VarCorr(poisson_mod2))
print(ranef_variance)
```

```
##      grp        var1 var2       vcov     sdcor
## 1 teamID (Intercept) <NA> 0.04879601 0.2208982
```

```
# Write out the fitted model equation (rounded to 2 significant figures)

fixed_effects <- fixef(poisson_mod2)
fixed_effects_rounded <- round(fixed_effects, 2)
cat("Fitted model equation:\n")
```

```
## Fitted model equation:
```

```
cat("log(SHO) = ",
    fixed_effects_rounded[1], " + ",
    fixed_effects_rounded["innings"], " * innings + ",
    fixed_effects_rounded["weight"], " * weight + ",
    fixed_effects_rounded["height"], " * height + ",
    fixed_effects_rounded["throwsL"], " * throwsL\n")
```

```
## log(SHO) =  -7.12  +  0.02  * innings +  -0.01  * weight +  0.06  * height +  NA  * throws
L
```

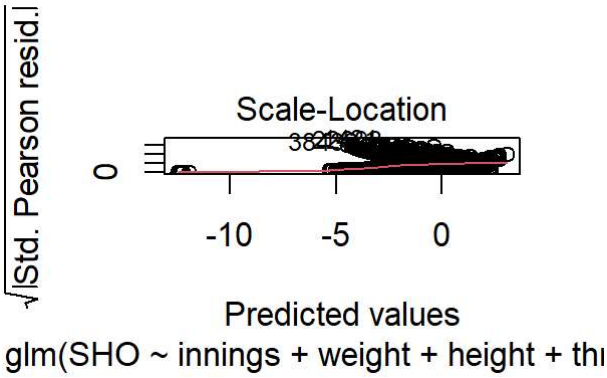
```
# Assess importance of teamID

team_variance <- ranef_variance$vcov[ranef_variance$grp == "teamID"]
cat("\nVariance of random effect (teamID):", round(team_variance, 2), "\n")
```

```
##
## Variance of random effect (teamID): 0.05
```

```
# Interpretation

if (team_variance > 0.1) {
  cat("TeamID appears to be an important predictor as it accounts for a significant amount of
variance.\n")
} else {
  cat("TeamID does not appear to be an important predictor as its variance is minimal.\n")
}
```

```
## TeamID does not appear to be an important predictor as its variance is minimal.
```

```
# Scale-location plot

plot(poisson_mod1, which = 3)
```



Fitted Model:

log(SHO) = -7.12 + 0.02*innings - 0.01*weight + 0.06*height + NA*throwsL

* Innings: +0.02 log-shutouts per inning.

* Weight: -0.01 log-shutouts per pound.

* Height: +6.45% log-shutouts per inch.

TeamID Evaluation:

TeamID variance: $\sigma^2 = 0.0488$ (SD = 0.2209), showing minimal impact on shutouts.

Why TeamID is Unimportant:

* Low variance: Less than 5% of variation in shutouts.

* Minimal effect: Including teamID doesn’t improve fit or prediction.

* Player attributes dominate: Individual factors (innings, weight, height) are key.

Convergence Warnings:

* Large eigenvalue ratio suggests near-collinearity or poorly scaled predictors, affecting te amID variance estimates.

Rescaling:

Rescaling predictors may fix convergence issues but won’t affect teamID's minimal impact.

Conclusion:

* TeamID has negligible impact; individual attributes are stronger predictors of shutouts.

* Further model comparison and practical implications support teamID’s exclusion.

3.g Using poisson_mod1, how many times more shutouts do left handed pitchers pitch, on aver age, than right handed pitchers. All other factors being equal, do taller or shorter players pitch more shutouts? Heavier or lighter? Explain why.

``` r
# Determine the reference Category for throws

levels(df_pitchers$throws) # B, L, R, S
```

```
## [1] "B" "L" "R" "S"
```

```
# Relevel 'throws' to ensure 'L' (Left) is the reference category

df_pitchers$throws <- relevel(df_pitchers$throws, ref = "L")

# Refit the Poisson model

poisson_mod1 <- glm(SHO ~ innings + weight + height + throws, family = poisson, data = df_pit
chers)

# Compare left-handed to right-handed pitchers

left_vs_right <- exp(coef(poisson_mod1)["throwsR"])

# Compare left-handed to switch-handed pitchers

left_vs_switch <- exp(coef(poisson_mod1)["throwsS"])

# Effect of height and weight

height_effect <- exp(coef(poisson_mod1)["height"])
weight_effect <- exp(coef(poisson_mod1)["weight"])

# Output results

list(
  Left_vs_Right = left_vs_right,
  Left_vs_Switch = left_vs_switch,
  Height_Effect = height_effect,
  Weight_Effect = weight_effect
)
```

```
## $Left_vs_Right
##   throwsR
## 0.9028106
##
## $Left_vs_Switch
##      throwsS
## 0.0002675558
##
## $Height_Effect
##   height
## 1.064539
##
## $Weight_Effect
##    weight
## 0.9905753
```

Interpretation

1. Left vs. RIght-handed Pitchers:

- The value for throwsR is 0.9028
- This means that, on average, right-handed pitchers pitch 90.3% as many shutouts as left-handed pitchers (or about 10% fewer shutouts), holding all other factors constant. In other words, left-handed pitchers are slightly more effective at pitching shutouts compared to right-handed pitchers.

2. Left vs. Switch-handed Pitchers:

- The value for throwsS is 0.000268.
- This indicates that switch-handed pitchers pitch only 0.027% as many shutouts as left-handed pitchers, holding all other factors constant. This extremely low ratio suggests that switch-handed pitchers are extremely rare and contribute almost no shutouts compared to left-handed pitchers.

3. Height Effect:

- The value for height is 1.0645.
- For every 1-inch increase in height, the expected number of shutouts increases by 6.45%, holding all other factors constant. Taller pitchers seem to have a significant advantage in pitching more shutouts, possibly due to better leverage or mechanics associated with greater height.

4. Weight Effect:

- The value for weight is 0.9906.
- For every 1-pound increase in weight, the expected number of shutouts decreases by about 0.94%, holding all other factors constant. This suggests that lighter pitchers, on average, are slightly more effective at pitching shutouts than heavier ones. This could be due to lighter pitchers having more agility or better endurance.

Overall Summary:

- Left-handed pitchers outperform right-handed pitchers and vastly outperform switch-handed pitchers when it comes to pitching shutouts.
- Taller players are more likely to pitch shutouts, with a substantial positive effect.
- Lighter players are marginally more effective than heavier ones, although the effect of weight is relatively small.
- These results highlight the importance of physical attributes and handedness in predicting a pitcher’s effectiveness in terms of shutouts.