2025 CPBL 中華職棒勝負/比分預測模型

第三組

洪毅荃 郭懿葶 余振瑋 林宇軒 陳志恆

2025-06-19

目錄

1 研究背景與目的 2

2 資料介紹 2

3 變數定義 2

4 變數解釋 3
- 4.1 打者指標解釋 . . . 3
- 4.2 投手指標解釋 . . . 4

5 敘述統計 4

6 Logistic Regression 6
- 6.1 建模 . . . 6
- 6.2 變數選擇策略 . . . 6
- 6.3 Backward Selection . . . 7
- 6.4 預測結果 . . . 9

7 Bradley Terry Model 12
- 7.1 介紹 . . . 12
- 7.2 建模 . . . 12
- 7.3 實例 . . . 13

8 Elo Rating System 15
- 8.1 介紹 . . . 15
- 8.2 運作流程 . . . 15
- 8.3 實例 . . . 15

9 Poisson Regression 20
- 9.1 建模 . . . 20
- 9.2 變數選擇 . . . 20
- 9.3 模型檢驗 . . . 24
- 9.4 Backward Seletion . . . 31
- 9.5 實例 . . . 45

10 結論與未來展望 47

11 分工表 47

12 Reference 47

1 研究背景與目的

在臺灣，運動彩券投注已成為許多球迷觀賽之外的重要娛樂方式，尤其是棒球項目。一般民眾在投注棒球彩券時，往往依賴直覺判斷與網站所提供之少量資訊，如兩隊的對戰紀錄、近期戰績、以及當日雙方先發投手的防禦率（**ERA**）等。然而，這些資訊未必能全面反映比賽結果的潛在影響因素，可能導致投注判斷偏誤。

因此，本研究旨在以 **2025** 年中華職棒上半季的比賽資料為基礎，建立四種不同的統計模型，整合多項比賽相關變數（如打擊與投手表現、主客場因素、近期勝率等），以提升比賽結果的預測準確性，進而為投注者提供更具科學依據的參考依據，跳脫傳統直覺式判斷，以提升投注者的信心。

本研究所採用的四種預測模型包括：

(1) Logistic Regression（邏輯斯迴歸）：用於建構以主場球隊勝率為應變數的模型。

(2) Bradley-Terry 模型：用以建構球隊間相對勝率的配對比較模型。

(3) Elo Rating System：透過比賽結果動態調整球隊強度分數，稱之 **Elo** 分數，以兩隊之 **Elo** 分數預測勝率。

(4) Poisson Regression（卜瓦松迴歸）：以比數資料為基礎，預測雙方得分與勝負機率。

透過這些模型的建立與比較，本研究希望能找出最具預測效力的方法，為未來棒球彩券投注提供更具系統性與實證支持的判斷依據。

2 資料介紹

本研究資料涵蓋 **2025** 年 **4** 月 **2** 日至 **2025** 年 **6** 月 **4** 日期間的中華職棒例行賽，共計 **125** 場比賽。其中，本研究以 **2025** 年 **4** 月 **2** 日至 **2025** 年 **5** 月 **22** 日作為分析依據，而以 **2025** 年 **5** 月 **23** 日至 **2025** 年 **6** 月 **4** 日作為測試資料。

資料來源包括：

中華職棒大聯盟全球資訊網：提供官方比賽結果、對戰組合、先發投手、防禦率等基本資訊。

野球革命：提供進階球員與球隊統計數據，如 **OPS**（**On-base Plus Slugging**）、**ERA**（**Earned Run Average**）、**WHIP**（**Walks plus Hits per Inning Pitched**）等。

這些資料將作為模型訓練與預測之依據，涵蓋比賽結果、投打成績、對戰歷史、主客場表現等重要變數。

3 變數定義

本研究使用的變數依來源可區分為以下兩類：

(1) 隊伍逐日表現資料（共 **12** 個變數）

紀錄各球隊每日的比賽表現，包含：

- 一般變數（**3** 項）：比賽日期、球隊名稱、勝負結果。
- 打擊相關變數（**9** 項）：如打擊率（**AVG**）、上壘加長打率（**OPS**）、加權上壘率（**wOBA**）等指標。
- 投球相關變數（**2** 項）：包含非先發投手的獨立投手表現指數（**FIP**）與每局被上壘率（**WHIP**）。
- 近期表現變數（**1** 項）：以近 **10** 場比賽的勝率反映球隊近期狀態。

(2) 隊伍對戰資料（共 **9** 個變數）

紀錄每場比賽雙方的對戰情形，包含：

- 一般變數（**1** 項）：比賽日期。
- 主場與客場隊伍資料（各 **4** 項）：球隊名稱、勝負結果、先發投手 **ERA** 以及球隊得分。

這些變數將用於訓練與測試預測模型，並針對比賽結果進行比較與評估，作為後續投注建議的重要依據。

4 變數解釋

4.1 打者指標解釋

(1) **AVG (打擊率)**

- 計算公式：安打數/打數
- 安打數：打者擊球並安全上壘的次數。
- 打數：打者實際有機會完成打擊的次數（不包括保送、觸身球或犧牲打等情況）。

(2) **OPS (整體攻擊指數)**

- 計算公式：上壘率 + 長打率

(3) **OBP (上壘率)**

- 計算公式：(安打數 + 保送次數 + 被觸身球次數)/(打數 + 保送次數 + 高飛犧牲打 + 壘打數)
- 上壘率和打擊率的最大差別就在於保送，而保送多寡和球員的選球能力及攻擊策略有很大的關係。

(4) **SLG (長打率)**

- 計算公式：壘打數/打數
- 代表打擊者每一次擊球可以貢獻幾個壘包。

(5) **OPS+ (標準化攻擊指數)**

- 計算公式：100*[(上壘率/聯盟平均上壘率)+(長打率/聯盟平均長打率)-1]
- 讓球員個人數據與聯盟平均值進行比較。OPS+ = 100 為聯盟平均，OPS+ > 100 為優於聯盟平均，OPS+ < 100 為劣於聯盟平均，OPS+ 亦可能為負值。

(6) **ISO (純長打率)**

- 計算公式：SLG - AVG
- 由 ISO 可以看出一位打者的長打能力。

(7) **BABIP (場內安打率)**

- 計算公式：(安打-全壘打)/(打數-三振-全壘打-高飛犧牲打)
- 打者將球打進場內（不包括全壘打）形成安打的機率。

(8) **wOBA (加權上壘率)**

- 計算公式：(0.722 * 非故意四壞球保送 +0.751 * 觸身球保送 +0.9 * 一壘安打 +1.244 * 二壘安打 +1.554 * 三壘安打 +1.947 * 全壘打)/(打數 + 非故意四壞球保送 + 高飛犧牲打 + 觸身球保送)
- 因為為每種打擊結果分配了不同的權重，能更準確地反映了打者對球隊得分的實際貢獻。

(9) **BB% (保送率)**

- 計算公式：保送/打席

(10) **K% (三振率)**

- 計算公式：三振/打席

(11) **BB/K**

- 計算公式：保送/三振

\end{itemize}

4.2 投手指標解釋

(1) ERA (防禦率)

- **ERA = 自責分 x 9 / 所投局數**
- 投手每九局平均會讓對方打線得幾分（自責分）。
- **ERA** 越低，表示投手的表現越好，代表該名投手能有效地防止對手得分。

(2) ERA+ (標準化防禦率)

- **ERA+ = 100 x 聯盟平均 ERA / (投手 ERA x 球場因子)**
  * 球場因子（**Park Factor**）：調整球場特性對 **ERA** 的影響，考慮某些球場可能更有利於打擊或投球。
- 是一個標準化比較的防禦率指標，用來衡量投手表現在不同聯盟、不同球場環境下的相對表現，**ex：ERA+** 高於 **100** 表明投手在聯盟中具有更高的競爭力，能有效限制對手得分。

(3) FIP (獨立防禦率)

- **FIP = [(13 x HR + 3 x BB - 2 x K) / 投球局數] + C**
  ***HR**：被打全壘打數。
  ***BB**：投手所投的四壞保送數。
  ***K**：三振數。
  ***C**：一個調整常數，用來使聯盟平均 **FIP** 接近聯盟平均 **ERA**，通常由聯盟數據計算得出。
- 以投手核心能力為基礎的數據指標，旨在消除防守和運氣對防禦率（**ERA**）的影響，只計算全壘打、保送、三振這三項可以被投手獨立控制的變數，提供更純粹的投手表現評估。

(4) WHIP (每局被上壘率)

- **WHIP = 保送數（BB）+ 安打數（H）/ 投球局數（IP）**
- 投手每投一局所面對的上壘人數的指標。

5 敘述統計

- **4/2 - 5/22 的主場隊伍逐日資料 (能力的差值)**

```
latex(describe(total_diff), file= "", caption.placement='top')
```

total_diff
15 Variables 101 Observations

date

n	missing	distinct
101	0	42

lowest : 4月10日 4月11日 4月12日 4月13日 4月15日, highest: 5月3日 5月4日 5月6日 5月7日 5月9日

away_team

n	missing	distinct
101	0	6

Value	兄弟	台鋼	味全	統一	富邦	樂天
Frequency	15	15	19	18	15	19
Proportion	0.149	0.149	0.188	0.178	0.149	0.188

home_team

n	missing	distinct
101	0	6

Value	兄弟	台鋼	味全	統一	富邦	樂天
Frequency	18	20	16	16	17	14
Proportion	0.178	0.198	0.158	0.158	0.168	0.139

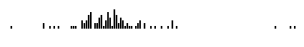
home_win

n	missing	distinct	Info	Sum	Mean
101	0	2	0.724	60	0.5941

away_win

n	missing	distinct	Info	Sum	Mean
101	0	2	0.724	41	0.4059

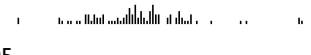
startERA_diff



n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10	.25	.50	.75	.90	.95
101	0	95	1	1.219	0.825	3.703	-3.90	-1.73	-0.90	0.76	2.04	4.70	6.74

lowest : -8.29 -5.4 -4.71 -4.4 -3.9 , highest: 6.75 7.07 16.36 17.64 18

BB_diff



n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10	.25	.50	.75	.90	.95
101	0	71	1	-0.09109	-0.2	3.641	-5.0	-3.7	-2.6	-0.3	1.3	3.2	4.9

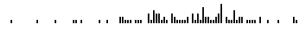
lowest : -8.3 -5.6 -5.5 -5.3 -5 , highest: 4.9 6.9 7.3 10.9 11

K_diff

n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10	.25	.50	.75	.90	.95
101	0	88	1	-0.00396	0.15	5.046	-7.5	-5.2	-2.2	0.5	2.7	4.1	5.9

lowest : -17.2 -9.1 -8.8 -8.5 -7.5 , highest: 7.6 9 11.3 13.5 15.6

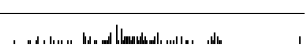
OBP_diff



n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10
101	0	78	1	-0.004109	-0.002	0.04675	-0.086	-0.054
.25	.50	.75	.90	.95				
-0.029	0.003	0.023	0.042	0.052				

lowest : -0.139 -0.12 -0.105 -0.092 -0.09 , highest: 0.052 0.058 0.067 0.078 0.08

ISO_diff



n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10
101	0	69	1	-0.002297	-0.003	0.03819	-0.062	-0.049
.25	.50	.75	.90	.95				
-0.020	-0.004	0.014	0.048	0.052				

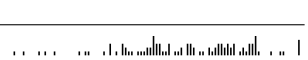
lowest : -0.077 -0.068 -0.067 -0.063 -0.062, highest: 0.05 0.052 0.053 0.054 0.104

RC_diff

n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10	.25	.50	.75	.90	.95
101	0	96	1	-1.787	-1.35	20.37	-28.1	-22.3	-13.9	-1.4	9.2	17.4	25.4

lowest : -51.4 -50.7 -48.5 -43.4 -39.6, highest: 29.3 36.5 38.6 39.9 40.7

wOBA_diff



n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10
101	0	78	1	-0.004406	-0.0025	0.04611	-0.073	-0.054
.25	.50	.75	.90	.95				
-0.027	-0.002	0.026	0.040	0.056				

lowest : -0.116 -0.109 -0.099 -0.095 -0.09 , highest: 0.043 0.05 0.056 0.059 0.068

FIP_diff

n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10	.25
101	0	98	1	0.004554	0.021	0.8288	-0.952	-0.742	-0.480
.50	.75	.90	.95						
-0.008	0.435	1.060	1.151						

lowest : -5.217 -1.081 -1.064 -1 -0.973, highest: 1.151 1.182 1.294 1.467 1.64

WHIP_diff

n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10	.25
101	0	97	1	-0.04427	0.016	0.4882	-1.033	-0.431	-0.205
.50	.75	.90	.95						
0.047	0.229	0.368	0.455						

lowest : -2.565 -1.647 -1.102 -1.04 -1.033, highest: 0.721 0.732 0.75 0.8 0.986

Win_diff

n	missing	distinct	Info	Mean	pMedian	Gmd	.05	.10	.25	.50	.75	.90	.95
101	0	12	0.976	-0.07921	0	2.66	-4	-3	-1	0	1	3	4

Value	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
Frequency	3	5	8	7	21	22	14	7	7	2	1	4
Proportion	0.030	0.050	0.079	0.069	0.208	0.218	0.139	0.069	0.069	0.020	0.010	0.040

For the frequency table, variable is rounded to the nearest 0

6 Logistic Regression

6.1 建模

由於若使用單一隊伍的勝率為應變數的模型，會沒有考慮到兩兩比較的情境，導致主客場雙方估計出來的勝率加總大於 1，因此建構以主場球隊勝率為應變數的模型，而自變數使用主客場隊伍的能力差。

6.2 變數選擇策略

由於棒球指標變數有嚴重的共線性問題，所以我們的策略是利用對於棒球的 Domain knowledge 以及對其餘變數做單變量分析 (篩選出 p-value < 0.25 的變數)，來選出我們最後的變數集。

(1) Domain knowledge for Baseball

由於我們蒐集到的打擊變數 (9 個)，其變數間多具有關聯，比如 AVG vs OBP 、SLG vs ISO 等，因此我們決定將重要的變數先篩選出來，並且排除掉與其有關聯的變數。

* 而一名打者的打擊能力可大致分為上壘能力及長打能力

* 而投手變數因只有先發 ERA、非先發 FIP 及 WHIP，因此不做篩選。

- 上壘能力 (AVG vs OBP):

  在傳統乃至現代棒球比賽中，球迷普遍習慣以打擊率（AVG）來評估選手的打擊表現。然而，與打擊率相比，上壘率（OBP）更能全面反映打者在比賽中對球隊的貢獻。打擊率僅計入安打的打席，忽略了保送（BB）等其他能讓打者上壘的方式，因而低估了選手的實際價值。

  此外，打擊率也較容易受到運氣因素的影響，例如場內安打率（BABIP）——這個指標常被用來衡量打者的「運氣成分」。當 BABIP 異常偏高時，可能會導致打擊率上升，而非真正反映打者的實力。

  綜合考量準確性與穩定性，我們選擇以上壘率（OBP）作為衡量打者上壘能力的主要變數。

- 長打能力 (SLG vs ISO):

  雖然長打率（SLG）會對二壘打、三壘打與全壘打給予較高權重，用以評估選手的長打能力，但其仍受到打擊率的影響。例如，一名選手若打擊率極高，即使多數安打是一壘安打，其 SLG 也可能偏高，導致長打能力被高估。

  相較之下，純長打率（ISO）則剔除了對一壘安打的考量，專注於額外壘打（extra-base hits），更能反映選手實際的長打貢獻。因此，在此分析中，我們選擇純長打率（ISO）作為衡量打者長打能力的主要變數。

  * 因此我們篩選出 OBP 及 ISO 作為進行分析的變數。

(2) 單變量分析

```
# outcome y
y <- "home_win"

# variables
x_vars <- c("wOBA_diff","RC_diff", "BB_diff", "K_diff",
            "startERA_diff","FIP_diff", "WHIP_diff", "Win_diff")

# save of the outcome of p-value
pvals <- numeric(length(x_vars))

# run logistic reg one by one
for (i in seq_along(x_vars)) {
  fml <- as.formula(paste0(y, " ~ ", x_vars[i]))
  model <- glm(fml, data = total_diff, family = binomial)
  pval <- summary(model)$coefficients[2, 4]
  pvals[i] <- pval
}
```

```r
# result
results <- data.frame(Variable = x_vars, P_value = pvals)
results$Keep <- results$P_value < 0.25
print(results)
```

```
       Variable    P_value  Keep
1     wOBA_diff 0.63452089 FALSE
2       RC_diff 0.31518435 FALSE
3       BB_diff 0.36982417 FALSE
4        K_diff 0.36788395 FALSE
5 startERA_diff 0.01093739  TRUE
6      FIP_diff 0.80421868 FALSE
7     WHIP_diff 0.05337577  TRUE
8      Win_diff 0.56916785 FALSE
```

由單變量分析的結果，我們選擇先發投手 **ERA** 差 **(startERA_diff)** 及非先發投手 **WHIP** 差 **(WHIP_diff)** 作為進行分析的變數。

因此，我們最後的變數集為 **OBP** 差、**ISO** 差、先發投手 **ERA** 差及非先發投手 **WHIP** 差。

6.3 Backward Selection

```r
glm12 <- glm(home_win ~ OBP_diff + ISO_diff +
               startERA_diff + WHIP_diff ,
             family = binomial, data = total_diff)

summary(glm12)
```

```
Call:
glm(formula = home_win ~ OBP_diff + ISO_diff + startERA_diff +
    WHIP_diff, family = binomial, data = total_diff)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.57607    0.23245   2.478   0.0132 *
OBP_diff       0.67209    5.79976   0.116   0.9077
ISO_diff      -0.62329    7.66570  -0.081   0.9352
startERA_diff -0.16243    0.07366  -2.205   0.0274 *
WHIP_diff     -0.73931    0.58834  -1.257   0.2089
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 136.42  on 100  degrees of freedom
Residual deviance: 125.08  on  96  degrees of freedom
AIC: 135.08

Number of Fisher Scoring iterations: 4
```

```r
car::vif(glm12)
```

```
     OBP_diff      ISO_diff startERA_diff     WHIP_diff
     1.183514      1.308103      1.057362      1.242294
```

```r
## backward

final_glm <- step(glm12, direction = "backward")
```

```
Start:  AIC=135.08
home_win ~ OBP_diff + ISO_diff + startERA_diff + WHIP_diff

                Df Deviance    AIC
- ISO_diff       1   125.08 133.09
- OBP_diff       1   125.09 133.09
- WHIP_diff      1   126.97 134.97
<none>               125.08 135.08
- startERA_diff  1   131.59 139.59

Step:  AIC=133.09
home_win ~ OBP_diff + startERA_diff + WHIP_diff

                Df Deviance    AIC
- OBP_diff       1   125.09 131.09
<none>               125.08 133.09
- WHIP_diff      1   127.12 133.12
- startERA_diff  1   131.80 137.80

Step:  AIC=131.09
home_win ~ startERA_diff + WHIP_diff

                Df Deviance    AIC
<none>               125.09 131.09
- WHIP_diff      1   127.22 131.22
- startERA_diff  1   131.82 135.82
```

```r
summary(final_glm)
```

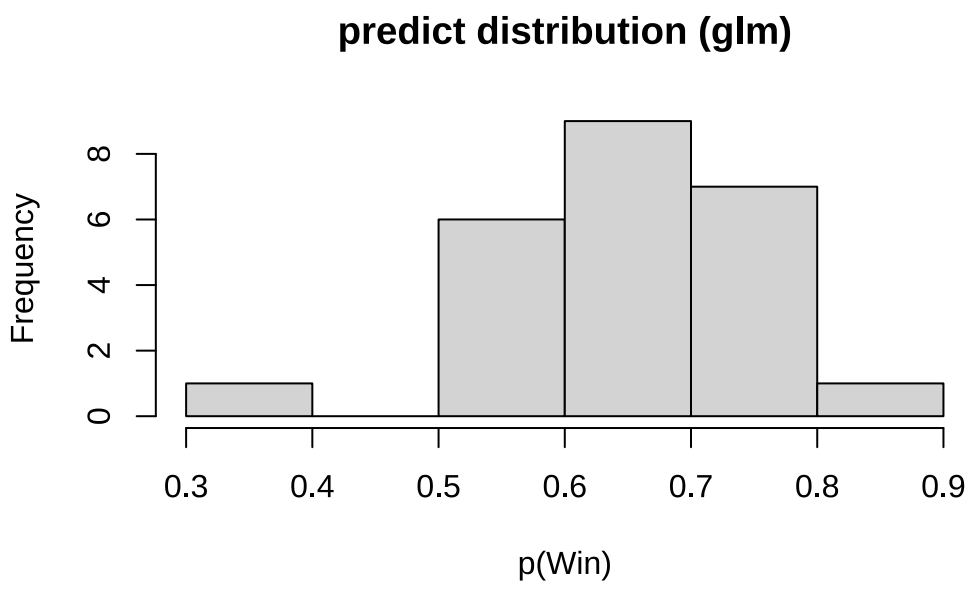
```
Call:
glm(formula = home_win ~ startERA_diff + WHIP_diff, family = binomial,
    data = total_diff)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.57564    0.22999   2.503   0.0123 *
startERA_diff -0.16369    0.07328  -2.234   0.0255 *
WHIP_diff     -0.73512    0.53984  -1.362   0.1733
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 136.42  on 100  degrees of freedom
Residual deviance: 125.09  on  98  degrees of freedom
AIC: 131.09

Number of Fisher Scoring iterations: 4
```

使用 **Backward selection** 進行變數篩選後，**Logistic Regression** 模型最終保留了「先發投手 **ERA** 差（**startERA_diff**）」及「非先發投手 **WHIP** 差（**WHIP_diff**）」兩項變數。這項結果顯示，在中華職棒（**CPBL**）的比賽中，打擊指標對於比

賽勝負的預測效果並不顯著，而投手表現相關的指標則對勝負結果具有顯著影響力。此一現象可能反映出：相較於打者的進攻能力，投手在比賽中的控分能力，尤其是先發與牛棚投手的表現落差，更直接左右比賽的勝負走向。

6.4 預測結果

```
# 套用模型預測機率
test_diff$predicted_prob <- predict(final_glm, newdata = test_diff, type = "response")

test_diff$Vic <- factor(test_diff$home_win, levels = c(0,1), labels = c("Lose", "Win"))
prob_test <- predict(final_glm, newdata = test_diff, type = "response")
y_test <- test_diff$home_win


roc_test <- roc(response = y_test, predictor = prob_test)

# 找最佳 cutoff (Youden Index)
opt_cut <- coords(roc_test, "best", ret = "threshold", best.method = "youden")
# opt_cut = 0.6790077

pred_o <- ifelse(prob_test > 0.6790077, 1, 0)


hist(prob_test, main = "predict distribution (glm)", xlab = "p(Win)")
```




```
test_diff$pred_factor <- factor(pred_o, levels = c(0, 1), labels = c("Lose", "Win"))
test_diff$actual_factor <- factor(y_test, levels = c(0,1), labels = c("Lose", "Win"))
confusionMatrix(test_diff$pred_factor, test_diff$actual_factor, positive = "Win")
```

```
Confusion Matrix and Statistics

          Reference
Prediction Lose Win
      Lose    7   8
      Win     1   8
```

```
               Accuracy : 0.625
                 95% CI : (0.4059, 0.812)
    No Information Rate : 0.6667
    P-Value [Acc > NIR] : 0.7462

                  Kappa : 0.3077

 Mcnemar's Test P-Value : 0.0455

            Sensitivity : 0.5000
            Specificity : 0.8750
         Pos Pred Value : 0.8889
         Neg Pred Value : 0.4667
             Prevalence : 0.6667
         Detection Rate : 0.3333
   Detection Prevalence : 0.3750
      Balanced Accuracy : 0.6875

       'Positive' Class : Win
```

由於我們的模型是預測主場勝負，而在棒球比賽中存在主場優勢，主場勝率大約為 60%，因此我們不使用 0.5 作為判斷預測勝負的閾值，而是使用 Youden Index 作為閾值，最後我們預測 2025 年 5 月 23 日至 2025 年 6 月 4 日比賽的準確率為 62.5%。

以下為 2025 年 5 月 23 日至 2025 年 6 月 4 日共 24 場比賽，模型預測與實際比賽的結果。

```r
head_num <- 24


result_df <- data.frame(
  Date      = test_diff$date[1:24],
  Actual    = test_diff$actual_factor[1:24],
  Predicted = test_diff$pred_factor[1:24]
)

# 顯示結果
print(result_df)
```

```
         Date Actual Predicted
1  2025-05-23   Lose      Lose
2  2025-05-23    Win      Lose
3  2025-05-23    Win       Win
4  2025-05-24   Lose       Win
5  2025-05-24    Win       Win
6  2025-05-24    Win       Win
7  2025-05-25   Lose      Lose
8  2025-05-25   Lose      Lose
9  2025-05-25    Win      Lose
10 2025-05-27   Lose      Lose
11 2025-05-27    Win      Lose
12 2025-05-28    Win      Lose
13 2025-05-28    Win      Lose
14 2025-05-30    Win      Lose
15 2025-05-30   Lose      Lose
16 2025-05-30    Win      Lose
17 2025-05-31   Lose      Lose
```

18	2025-05-31	Win	Win
19	2025-05-31	Lose	Lose
20	2025-06-01	Win	Lose
21	2025-06-01	Win	Win
22	2025-06-01	Win	Win
23	2025-06-03	Win	Win
24	2025-06-04	Win	Win

7 Bradley Terry Model

7.1 介紹

Bradley–Terry Model 是一種用於成對比較的統計模型，能夠根據歷史對戰結果估計各隊的能力值，進而推算任意兩隊之間的勝率。在本研究中，我們套用 **Bradley–Terry** 模型，除了基礎的隊伍能力差外，還納入其他可能影響勝負的變數，包括主客場因素與先發投手的 **ERA**。這些變數有助於捕捉場地與當場對戰條件的即時影響，使預測更貼近實際情境。

而在模型中會有三個限制：

(1) 兩兩之間需要有對戰紀錄

(2) 不能有平手的情況

(3) 結構性分離，結構性分離為一支隊伍不能只有贏或輸的比賽，這樣的模型估出來的係數會發散掉。

7.2 建模

```
# 確保 Team1 / Team2 為同一 levels 的 factor
all_teams <- factor(union(X42_46$home_team, X42_46$away_team))

# 改變 baseline
# all_teams <- factor(union(X42_46$home_team, X42_46$away_team)
#                     ,levels = c(" 統一", " 味全", " 樂天", " 兄弟", " 富邦", " 台鋼"))

X42_46$home_team <- factor(X42_46$home_team, levels = levels(all_teams))
X42_46$away_team <- factor(X42_46$away_team, levels = levels(all_teams))

# 將主客場（factor）轉為 0/1 數值
X42_46$home_team1 <- as.integer(as.character(X42_46$主客場))
X42_46$home_team2 <- 1 - X42_46$home_team1

# 建立 outcome（Team1 是否勝利）
X42_46$outcome <- cbind(X42_46$home_win, X42_46$away_win)

# === 建立 player1 / player2 的隊伍變數 ===
player1 <- data.frame(
  team = X42_46$home_team,
  ERA = X42_46$home_startERA,
  home = X42_46$home_team1
)

player2 <- data.frame(
  team = X42_46$away_team,
  ERA = X42_46$away_startERA,
  home = X42_46$home_team2
)


############## model #############

bt_model <- BTm(
  outcome = X42_46$outcome,
  player1 = player1,
  player2 = player2,
  formula = ~ ERA + home + team,
  id = "team"
```

```
)

summary(bt_model)
```

```
Call:
BTm(outcome = X42_46$outcome, player1 = player1, player2 = player2,
    formula = ~ERA + home + team, id = "team")

Coefficients:
         Estimate Std. Error z value Pr(>|z|)
ERA      -0.16901    0.07163  -2.360  0.01830 *
home      0.65330    0.21321   3.064  0.00218 **
team台鋼 -0.56022    0.44071  -1.271  0.20367
team味全 -0.33665    0.44049  -0.764  0.44470
team統一 -0.18549    0.43889  -0.423  0.67257
team富邦 -0.87213    0.46830  -1.862  0.06255 .
team樂天 -0.11687    0.42591  -0.274  0.78378
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 173.29  on 125  degrees of freedom
Residual deviance: 153.00  on 118  degrees of freedom
AIC: 167

Number of Fisher Scoring iterations: 4
```

7.3 實例

預測 6/3 (客: 富邦 vs 主: 味全) 的比賽

- 富邦: 先發投手的 ERA(3.44)、客場
- 味全: 先發投手的 ERA(2.15)、主場

```
BTabilities(bt_model)
```

```
        ability      s.e.
兄弟  0.0000000 0.0000000
台鋼 -0.5602227 0.4407133
味全 -0.3366547 0.4404904
統一 -0.1854857 0.4388856
富邦 -0.8721326 0.4682979
樂天 -0.1168688 0.4259097
```

以兄弟隊為基準隊伍，其它隊伍的能力值則會反映出與兄弟隊相對的差距。可以看出樂天最接近兄弟，而富邦的能力值最低，顯著低後於其他隊伍。

```
new_player1 <- data.frame(
  team = factor(" 味全", levels = levels(X42_46$home_team)),
  ERA = 2.15,
  home = 1
)

new_player2 <- data.frame(
  team = factor(" 富邦", levels = levels(X42_46$home_team)),
```

```
  ERA = 3.44,
  home = 0
)

predict(bt_model,newdata = list(player1 = new_player1, player2 = new_player2),
        type = "response")
```

```
        1
0.8032606
```

下一個對戰組合為富邦 vs 味全，而將兩隊主客場資訊及先發投手 ERA 放進模型預測，得知味全下一場對上富邦的勝率為 80%。

8 Elo Rating System

8.1 介紹

Elo 等級分制度源自於美國西洋棋協會，於 **1960** 年第一次出現這種計分方法，由匈牙利裔美國物理學家 **Arpad Elo** 所創建，並於 **1970** 年國際棋聯（**FIDE**）正式開始使用作為全球棋手排名依據，而時至今日被廣泛用於西洋棋、維奇、足球、電子競技的排名等。

Elo 模型是基於統計學的一個評估棋手水準的方法，原始的 **Elo** 模型假設選手實力服從常態分布，並依據對戰雙方的預測勝率與實際比賽結果動態調整等級分。然而，現實情況中選手或隊伍的表現波動不一定符合常態分布，因此，部分現代多以採用對數分布來進行預測，使模型能更準確且更及時的反映競賽表現。

在本研究中，我們將 **Elo** 模型應用於棒球比賽中，透過每場比賽結果更新各隊的實力分數，進而預測未來對戰中的勝率與比賽結果。

8.2 運作流程

(1) 初始: 每隊分配初始 **Elo** 分數，通常設為 **1500** 分，作為起始的實力基準。

(2) 預測: 計算雙方預測勝率，以 **A** 隊對 **B** 隊為例，**A** 隊的預測勝率計算公式為：

$$P_A = \frac{1}{1 + 10^{\left(\frac{R_B - R_A}{400}\right)}}$$

其中，400 為常數，可以根據想要的動態更新速度自行調整（數值越小，分數變動越快）。

(3) 結果: 比賽結束後，根據實際勝負結果調整 **Elo** 分數，以 **A** 為例，其更新後的分數計算公式：

$$R_{A_{\text{NEW}}} = R_A + K(S_A - P_A)$$

- K：調整係數（決定分數調整幅度，常見值為 **20** 或 **32**）
- S_A：比賽實際結果（勝 = **1**，和 = **0.5**，負 = **0**）
- P_A：預測勝率（根據 **Elo** 模型計算）

(4) 重複: 每場比賽後，根據更新後的 **Elo** 分數重新計算下一場比賽的預測勝率。依此循環，模型會隨賽季進行逐漸修正球隊實力評估。

8.3 實例

***** 以 **2025** 年中華職棒上半季為例

首先，本研究之初始分數以傳統基準值 **1500** 分為中心進行調整，參考 **2025** 年中華職棒官辦熱身賽的排名，具體設定如下：

球隊	初始 Elo 分數
富邦悍將	1560
味全龍	1530
樂天桃猿	1500
台鋼雄鷹	1500
統一獅	1470
中信兄弟	1440

表 1: 各球隊初始 Elo 分數設定

為提升模型適配度，本研究進一步納入兩項調整因素：

- 主場優勢：主場球隊加上 **50** 分。
- 先發投手 **ERA** 差異：以 **ERA** 差乘以調整係數 **20**。

以 **2025** 年 **3** 月 **30** 日，統一獅（**U**）對戰台鋼雄鷹（**T**）為例，統一獅（**U**）的預測勝率 **(**P_U**)** 計算方式如下：

- 統一獅（U）原始 Elo 分數：R_U = 1452.005
- 台鋼雄鷹（T）原始 Elo 分數：R_T = 1500
- 主場加成：home=50
- 先發投手 ERA（台鋼雄鷹（T）：2.77，統一獅（U）：3.67）：

$$\text{adj}_{\text{factor}} = 20(2.77 - 3.67) = -18$$

- 代入公式，並得到該場比賽統一獅（U）的勝率：

$$P_U = \frac{1}{1+10^{\left(\frac{1500-(1452.005+50+(-18))}{400}\right)}} = \frac{1}{1+10^{\left(\frac{1500-1484.005}{400}\right)}} = \frac{1}{1+10^{0.0399875}} \quad 0.4770$$

本場比賽由統一獅（U）獲勝，因此我們根據實際結果調整 Elo 分數。以統一獅（U）為例，其調整公式的計算方式如下：

- $K = 20$（調整係數）
- $S_U = 1$（統一獅獲勝，勝場指標為 1）
- $P_U = 0.4770$（根據 Elo 模型預測統一獅的勝率）

根據 Elo 更新公式：

$$R_{U_{NEW}} = R_U + K(S_U - P_U)$$

代入數值：

$$R_{U_{NEW}} = 1452.005 + 20(1 - 0.4770) = 1462.465 \quad 1462.5$$

因此，1462.5 將作為下一場比賽統一獅（U）的初始 Elo 分數。
每場比賽結束後，雙方的 Elo 分數都會即時更新，用於預測下一場比賽結果。
隨著球季進行，Elo 分數會動態反映球隊實力變化，形成一套隨時間推移的實力指標，反映球隊狀態與趨勢。

```
# 計算 Elo 分數：每個排名差距 30 分，從第一名 1560 開始
# 找出唯一排名，排序後對應到 Elo
unique_ranks <- sort(unique(elo_init$排名))
rank_to_elo <- setNames(1560 - (0:(length(unique_ranks) - 1)) * 30, unique_ranks)

# 套用 Elo 分數
elo_init <- elo_init %>%
  mutate(elo_init = rank_to_elo[as.character(排名)])
print(elo_init)
```

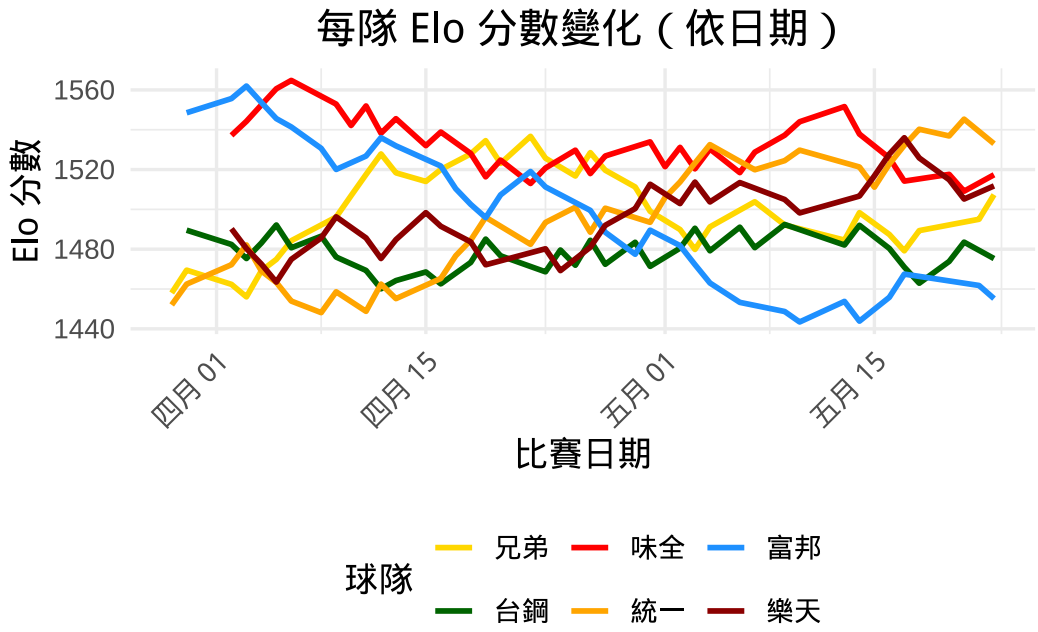
```
# A tibble: 6 x 7
   排名 球隊             win  lose deuce  勝率 elo_init
  <dbl> <chr>          <dbl> <dbl> <dbl> <dbl>    <dbl>
1     1 富邦悍將           6     4     0 0.6       1560
2     2 味全龍             5     4     1 0.556     1530
3     3 樂天桃猿           5     5     0 0.5       1500
4     3 台鋼雄鷹           5     5     0 0.5       1500
5     5 中信兄弟           4     5     1 0.444     1470
6     6 統一7-ELEVEN獅     4     6     0 0.4       1440
```

```
elo_ratings <- c(
  " 富邦" = 1560,
  " 味全" = 1530,
  " 樂天" = 1500,
```

```
  " 台鋼" = 1500,
  " 兄弟" = 1470,
  " 統一" = 1440
)

# 主場球隊 +50 分
home <- 50

# 調整係數（K-factor）
K <- 20

# 建立紀錄表
results_2 <- data.frame()

match_df <- match_df %>%
  mutate(date = as.Date(date))  # 去除時間與時區資訊，只保留日期

update_elo <- function(R_A, R_B, S_A, K = 20, hfa = 0, adj = 0) {
  # 加入主場優勢與投手調整
  R_A_adj <- R_A + hfa + adj
  R_B_adj <- R_B

  # 勝率預測
  P_A <- 1 / (1 + 10^((R_B_adj - R_A_adj) / 400))
  P_B <- 1 - P_A

  # Elo 更新
  new_R_A <- R_A + K * (S_A - P_A)
  new_R_B <- R_B + K * ((1 - S_A) - P_B)

  return(list(R_A = new_R_A, R_B = new_R_B, P_A = P_A, P_B = P_B))
}


for (i in 1:nrow(match_df)) {
  row <- match_df[i, ]

  # 基本欄位
  team_A <- row$home_team
  team_B <- row$away_team
  score_A <- row$home_scores
  score_B <- row$away_scores
  game_date <- row$date

  # 勝者判斷
  winner <- ifelse(score_A > score_B, team_A, team_B)
  S_A <- ifelse(winner == team_A, 1, 0)

  # 讀取當前 Elo 分數
  R_A <- elo_ratings[team_A]
  R_B <- elo_ratings[team_B]

  # 主場優勢
  hfa <- home

  # 先發 ERA 差異：用該場比賽資訊
```

```r
  home_era <- row$home_startERA
  away_era <- row$away_startERA

  # 投手調整因子：投手越好，分數越高
  adj_factor <- 20 * (away_era - home_era)  # 如果主隊投手比較好，分數變高

  # 更新 Elo 分數
  result_2 <- update_elo(R_A, R_B, S_A, K = K, hfa = hfa, adj = adj_factor)
  new_R_A <- result_2$R_A
  new_R_B <- result_2$R_B
  P_A <- result_2$P_A
  P_B <- result_2$P_B

  # 儲存比賽紀錄
  results_2 <- rbind(results_2, data.frame(
    date = as.Date(game_date),  # 確保是標準 Date 類型
    team_A = team_A,
    team_B = team_B,
    score_A = score_A,
    score_B = score_B,
    winner = winner,
    home_era = home_era,
    away_era = away_era,
    Elo_A_before = R_A,
    Elo_B_before = R_B,
    P_team_A_win = round(P_A, 4),
    P_team_B_win = round(P_B, 4),
    Elo_A_after = round(new_R_A, 4),
    Elo_B_after = round(new_R_B, 4)
  ))

  # 更新球隊 Elo
  elo_ratings[team_A] <- new_R_A
  elo_ratings[team_B] <- new_R_B
}

elo_long <- results_2 %>%
  dplyr::select(date, team = team_A, elo = Elo_A_after) %>%
  dplyr::bind_rows(
    results_2 %>%
      dplyr::select(date, team = team_B, elo = Elo_B_after)
  ) %>%
  dplyr::arrange(team, date)


# 設定球隊顏色
team_colors <- c(
  " 富邦" = "#1E90FF",    # dodgerblue
  " 兄弟" = "#FFD700",    # yellow
  " 樂天" = "#8B0000",    # dark red
  " 味全" = "#FF0000",    # red
  " 統一" = "#FFA500",    # orange
  " 台鋼" = "#006400"     # dark green
)

# 畫圖
```

```r
ggplot(elo_long, aes(x = date, y = elo, color = team)) +
  geom_line(linewidth = 1) +
  scale_color_manual(values = team_colors) +
  labs(
    title = " 每隊 Elo 分數變化（依日期）",
    x = " 比賽日期",
    y = "Elo 分數",
    color = " 球隊"
  ) +
  theme_minimal(base_family = "Noto Sans TC", base_size = 13) +
  theme(
    plot.title = element_text(face = "bold", hjust = 0.5),
    legend.position = "bottom",
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```



以上圖表呈現了 2025 年 4 月 2 日至 5 月 22 日各隊隨比賽進行所累積的 Elo 分數變化情形。橫軸為比賽日期，縱軸為 Elo 分數，圖中不同顏色代表各隊。

不過，需要注意的是，Elo 分數的高低所反映的是球隊「近期相對實力表現」，不一定完全對應戰績或排名，舉例來說，若一支球隊擊敗 Elo 分數高的對手，將會獲得較多分數；反之，輸給 Elo 分數較低的隊伍，會失去較多分數，因此，Elo 分數並非戰績排名指標，而是動態實力評估，須搭配賽程與對手強度一同解讀。

9 Poisson Regression

9.1 建模

對六隊分別建立 **Poisson Regression Model**，因為是希望預測某特定隊伍的比分，而得分所受因素為自己球隊的打擊能力以及敵方球隊的防守能力，所以在投手指標的部分，使用的是該場敵隊的投手指標變數。

建模流程:

(1) 將各隊最後選擇的變數集對各隊建立 **Poisson Regression**

(2) 檢驗 **Overdispersion**，有 **Overdispersion**，使用負二項回歸

(3) 進行 **backward selection**

9.2 變數選擇

與 **Logistic Regression** 的變數選擇策略一樣，使用對於棒球的 **Domain knowledge** 以及對其餘變數做單變量分析 **(**篩選出 **p-value < 0.25** 的變數**)**，來選出我們最後的變數集。

(1) 兄弟

```
# 應變數
y <- "score"
# 自變數列表
x_vars <- c("wOBA", "`BB%`", "`K%`", "RC", " 近 10 場勝場",
            "`先發 ERA(對手)`", "`非先發 FIP(對手)`", "`非先發 WHIP(對手)`")

# 儲存 p 值結果
pvals <- numeric(length(x_vars))

# 一個一個跑 logistic 回歸
for (i in seq_along(x_vars)) {
  fml <- as.formula(paste0(y, " ~ ", x_vars[i]))
  model <- glm(fml, data = poi_bro, family = poisson)
  pval <- summary(model)$coefficients[2, 4]  # 第二行就是該變數的 p 值
  pvals[i] <- pval
}

# 結果整理
results <- data.frame(Variable = x_vars, P_value = pvals)
results$Keep <- results$P_value < 0.25
print(results)
```

```
            Variable     P_value  Keep
1               wOBA 0.002893746  TRUE
2              `BB%` 0.380945370 FALSE
3              `K%` 0.492026356 FALSE
4                RC 0.322328587 FALSE
5         近10場勝場 0.479833685 FALSE
6    `先發ERA(對手)` 0.001610826  TRUE
7  `非先發FIP(對手)` 0.040532286  TRUE
8 `非先發WHIP(對手)` 0.519434043 FALSE
```

最後兄弟的變數集為 **OBP** 差、**ISO** 差、**wOBA** 差、先發 **ERA** 差 **(**對手**)** 及非先發 **FIP** 差 **(**對手**)**。

(2) 味全

```
# 應變數
y <- "score"
# 自變數列表
```

```r
x_vars <- c("wOBA", "`BB%`", "`K%`" , "RC", " 近 10 場勝場",
            "`先發 ERA(對手)`", "`非先發 FIP(對手)`", "`非先發 WHIP(對手)`")

# 儲存 p 值結果
pvals <- numeric(length(x_vars))

# 一個一個跑 logistic 回歸
for (i in seq_along(x_vars)) {
  fml <- as.formula(paste0(y, " ~ ", x_vars[i]))
  model <- glm(fml, data = poi_dragon, family = poisson)
  pval <- summary(model)$coefficients[2, 4]  # 第二行就是該變數的 p 值
  pvals[i] <- pval
}

# 結果整理
results <- data.frame(Variable = x_vars, P_value = pvals)
results$Keep <- results$P_value < 0.25
print(results)
```

```
            Variable    P_value  Keep
1               wOBA 0.55116953 FALSE
2              `BB%` 0.48286565 FALSE
3               `K%` 0.01901801  TRUE
4                 RC 0.14700625  TRUE
5         近10場勝場 0.30506139 FALSE
6    `先發ERA(對手)` 0.99455918 FALSE
7  `非先發FIP(對手)` 0.19355832  TRUE
8 `非先發WHIP(對手)` 0.50375372 FALSE
```

最後味全的變數集為 **OBP 差、ISO 差、K% 差、RC 差及非先發 FIP 差 (對手)**。

(3) 富邦

```r
# 應變數
y <- "score"
# 自變數列表
x_vars <- c("wOBA", "`BB%`", "`K%`", "RC",  " 近 10 場勝場",
            "`先發 ERA(對手)`", "`非先發 FIP(對手)`", "`非先發 WHIP(對手)`")

# 儲存 p 值結果
pvals <- numeric(length(x_vars))

# 一個一個跑 logistic 回歸
for (i in seq_along(x_vars)) {
  fml <- as.formula(paste0(y, " ~ ", x_vars[i]))
  model <- glm(fml, data = poi_fubon, family = poisson)
  pval <- summary(model)$coefficients[2, 4]  # 第二行就是該變數的 p 值
  pvals[i] <- pval
}

# 結果整理
results <- data.frame(Variable = x_vars, P_value = pvals)
results$Keep <- results$P_value < 0.25
print(results)
```

```
            Variable    P_value  Keep
1               wOBA 0.10983522  TRUE
2              `BB%` 0.02356139  TRUE
```

```
3               ‘K%‘ 0.95860388 FALSE
4                 RC 0.96154373 FALSE
5         近10場勝場 0.15668784  TRUE
6    ‘先發ERA(對手)‘ 0.84507367 FALSE
7  ‘非先發FIP(對手)‘ 0.40400136 FALSE
8 ‘非先發WHIP(對手)‘ 0.01314951  TRUE
```

最後富邦的變數集為 OBP 差、ISO 差、wOBA 差、BB% 差 (對手)、近 10 場勝場差、及非先發 WHIP 差 (對手)。

(4) 台鋼

```r
# 應變數
y <- "score"
# 自變數列表
x_vars <- c("wOBA", "`BB%`", "`K%`", "RC",  " 近 10 場勝場",
            "`先發 ERA(對手)`", "`非先發 FIP(對手)`", "`非先發 WHIP(對手)`")

# 儲存 p 值結果
pvals <- numeric(length(x_vars))

# 一個一個跑 logistic 回歸
for (i in seq_along(x_vars)) {
  fml <- as.formula(paste0(y, " ~ ", x_vars[i]))
  model <- glm(fml, data = poi_hawk, family = poisson)
  pval <- summary(model)$coefficients[2, 4]  # 第二行就是該變數的 p 值
  pvals[i] <- pval
}

# 結果整理
results <- data.frame(Variable = x_vars, P_value = pvals)
results$Keep <- results$P_value < 0.25
print(results)
```

```
            Variable    P_value  Keep
1               wOBA 0.20519638  TRUE
2              ‘BB%‘ 0.17135223  TRUE
3               ‘K%‘ 0.15912269  TRUE
4                 RC 0.02616117  TRUE
5         近10場勝場 0.39545247 FALSE
6    ‘先發ERA(對手)‘ 0.75328703 FALSE
7  ‘非先發FIP(對手)‘ 0.89533442 FALSE
8 ‘非先發WHIP(對手)‘ 0.88231664 FALSE
```

最後台鋼的變數集為 OBP 差、ISO 差、wOBA 差、BB% 差、K% 差及 RC 差。

(5) 統一

```r
# 應變數
y <- "score"
# 自變數列表
x_vars <- c("wOBA", "`BB%`", "`K%`", "RC",  " 近 10 場勝場",
            "`先發 ERA(對手)`", "`非先發 FIP(對手)`", "`非先發 WHIP(對手)`")

# 儲存 p 值結果
pvals <- numeric(length(x_vars))

# 一個一個跑 logistic 回歸
for (i in seq_along(x_vars)) {
  fml <- as.formula(paste0(y, " ~ ", x_vars[i]))
  model <- glm(fml, data = poi_lion, family = poisson)
```

```
  pval <- summary(model)$coefficients[2, 4]  # 第二行就是該變數的 p 值
  pvals[i] <- pval
}

# 結果整理
results <- data.frame(Variable = x_vars, P_value = pvals)
results$Keep <- results$P_value < 0.25
print(results)
```

```
            Variable     P_value  Keep
1               wOBA 0.077493073  TRUE
2              `BB%` 0.238875082  TRUE
3              `K%` 0.048097435  TRUE
4                RC 0.279380358 FALSE
5         近10場勝場 0.428880000 FALSE
6    `先發ERA(對手)` 0.049332609  TRUE
7  `非先發FIP(對手)` 0.800202544 FALSE
8 `非先發WHIP(對手)` 0.002315261  TRUE
```

最後統一的變數集為 OBP 差、ISO 差、wOBA 差、BB% 差、K% 差、先發 ERA 差 (對手) 及非先發 WHIP 差 (對手)。

(6) 樂天

```
# 應變數
y <- "score"
# 自變數列表
x_vars <- c("wOBA", "`BB%`", "`K%`", "RC",  " 近 10 場勝場",
            "`先發 ERA(對手)`", "`非先發 FIP(對手)`", "`非先發 WHIP(對手)`")

# 儲存 p 值結果
pvals <- numeric(length(x_vars))

# 一個一個跑 logistic 回歸
for (i in seq_along(x_vars)) {
  fml <- as.formula(paste0(y, " ~ ", x_vars[i]))
  model <- glm(fml, data = poi_monkey, family = poisson)
  pval <- summary(model)$coefficients[2, 4]  # 第二行就是該變數的 p 值
  pvals[i] <- pval
}

# 結果整理
results <- data.frame(Variable = x_vars, P_value = pvals)
results$Keep <- results$P_value < 0.25
print(results)
```

```
            Variable    P_value  Keep
1               wOBA 0.30177151 FALSE
2              `BB%` 0.11932841  TRUE
3              `K%` 0.09503832  TRUE
4                RC 0.17425457  TRUE
5         近10場勝場 0.59193672 FALSE
6    `先發ERA(對手)` 0.16159154  TRUE
7  `非先發FIP(對手)` 0.28550802 FALSE
8 `非先發WHIP(對手)` 0.01641866  TRUE
```

最後樂天的變數集為 OBP 差、ISO 差、BB% 差、K% 差、RC 差、先發 ERA 差 (對手) 及非先發 WHIP 差 (對手)。

9.3 模型檢驗

```
##### 統一 #####
poi2_lion <- glm(score ~ OBP + ISO + wOBA + `K%` + `BB%` +
                   `先發 ERA(對手)` + `非先發 WHIP(對手)`, family = poisson, data = poi_lion)
summary(poi2_lion)
```

```
Call:
glm(formula = score ~ OBP + ISO + wOBA + `K%` + `BB%` + `先發ERA(對手)` +
    `非先發WHIP(對手)`, family = poisson, data = poi_lion)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)         -2.91155    4.75271  -0.613  0.54014
OBP                  5.42661  100.52271   0.054  0.95695
ISO                 12.33181   31.95667   0.386  0.69958
wOBA               -24.13554  104.85183  -0.230  0.81795
`K%`                 0.42018    0.14859   2.828  0.00469 **
`BB%`               -0.16036    0.31652  -0.507  0.61242
`先發ERA(對手)`      0.04800    0.03799   1.263  0.20642
`非先發WHIP(對手)`   2.39023    0.54850   4.358 1.31e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 78.562  on 33  degrees of freedom
Residual deviance: 46.508  on 26  degrees of freedom
AIC: 165.97

Number of Fisher Scoring iterations: 5
```

```
car::vif(poi2_lion)
```

```
              OBP               ISO              wOBA              `K%`
       273.470389         13.197341        389.603911          2.186339
           `BB%`    `先發ERA(對手)` `非先發WHIP(對手)`
        11.264287          1.180353          1.611945
```

```
# 檢驗零膨脹
zip_mod <- zeroinfl(score ~ OBP + ISO + wOBA + `K%` + `BB%` +
                      `先發 ERA(對手)` + `非先發 WHIP(對手)` |
                      OBP + ISO + wOBA + `K%` + `BB%` +
                      `先發 ERA(對手)` + `非先發 WHIP(對手)`, data = poi_lion)
pois_mod <- glm(score ~ OBP + ISO + wOBA + `K%` + `BB%` +
                  `先發 ERA(對手)` + `非先發 WHIP(對手)` , family = poisson, data = poi_lion)
vuong(zip_mod, pois_mod)
```

```
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
             Vuong z-statistic             H_A  p-value
Raw                  1.5366115 model1 > model2 0.062194
AIC-corrected       -0.6395507 model2 > model1 0.261232
BIC-corrected       -2.3003548 model2 > model1 0.010714
```

```r
# 檢驗 overdispersion
dispersiontest(poi2_lion)
```

```
    Overdispersion test

data:  poi2_lion
z = 1.1209, p-value = 0.1312
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
   1.23935
```

```r
##### 兄弟 #####
poi2_bro <- glm(score ~ OBP + ISO + wOBA +
                  `先發 ERA(對手)` + `非先發 FIP(對手)`, family = poisson, data = poi_bro)
summary(poi2_bro)
```

```
Call:
glm(formula = score ~ OBP + ISO + wOBA + `先發ERA(對手)` +
    `非先發FIP(對手)`, family = poisson, data = poi_bro)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)         8.40704    2.71020   3.102 0.001922 **
OBP                40.11815   39.10665   1.026 0.304955
ISO                32.82290   36.16108   0.908 0.364044
wOBA              -74.28920   49.85917  -1.490 0.136229
`先發ERA(對手)`     0.19878    0.05805   3.424 0.000617 ***
`非先發FIP(對手)`  -0.31802    0.21316  -1.492 0.135711
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 80.616  on 32  degrees of freedom
Residual deviance: 55.798  on 27  degrees of freedom
AIC: 163.07

Number of Fisher Scoring iterations: 5
```

```r
car::vif(poi2_bro)
```

```
             OBP              ISO             wOBA   `先發ERA(對手)`
       22.514332         6.401516        35.421918         1.208896
`非先發FIP(對手)`
        1.361488
```

```r
# 檢驗零膨脹
zip_mod <- zeroinfl(score ~ OBP + ISO + wOBA +
                      `先發 ERA(對手)` + `非先發 FIP(對手)` |
                      OBP + ISO + wOBA +
                      `先發 ERA(對手)` + `非先發 FIP(對手)` , data = poi_bro)
pois_mod <- glm(score ~ OBP + ISO + wOBA +
                  `先發 ERA(對手)` + `非先發 FIP(對手)` , family = poisson, data = poi_bro)
vuong(zip_mod, pois_mod)
```

```
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A  p-value
Raw                   1.4830083 model1 > model2 0.069036
AIC-corrected        -0.6039046 model2 > model1 0.272954
BIC-corrected        -2.1654451 model2 > model1 0.015177
```

```r
# 檢驗 overdispersion
dispersiontest(poi2_bro)
```

```
    Overdispersion test

data:  poi2_bro
z = 2.0742, p-value = 0.01903
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.609073
```

```r
##### 味全 #####
poi2_dragon <- glm(score ~ OBP + ISO + `K%` + RC +
                     `非先發 FIP(對手)`, family = poisson, data = poi_dragon)
summary(poi2_dragon)
```

```
Call:
glm(formula = score ~ OBP + ISO + `K%` + RC + `非先發FIP(對手)`,
    family = poisson, data = poi_dragon)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)        0.734052   2.939817   0.250 0.802825
OBP               -4.524285  11.824687  -0.383 0.702006
ISO                2.879988  14.738783   0.195 0.845078
`K%`               0.370593   0.071272   5.200    2e-07 ***
RC                -0.014795   0.005693  -2.599 0.009356 **
`非先發FIP(對手)` -0.952209   0.249513  -3.816 0.000135 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 106.100  on 34  degrees of freedom
Residual deviance:  70.085  on 29  degrees of freedom
AIC: 176.43

Number of Fisher Scoring iterations: 5
```

```r
car::vif(poi2_dragon)
```

```
             OBP              ISO             `K%`               RC
        6.229305         7.811869         1.650705         5.293109
`非先發FIP(對手)`
        1.319041
```

```r
# 檢驗零膨脹
zip_mod <- zeroinfl(score ~ OBP + ISO + `K%` + RC +
                      `非先發 FIP(對手)`
                    | OBP + ISO + `K%` + RC +
                      `非先發 FIP(對手)` , data = poi_dragon)
pois_mod <- glm(score ~ OBP + ISO + `K%` + RC +
                  `非先發 FIP(對手)` , family = poisson, data = poi_dragon)
vuong(zip_mod, pois_mod)
```

```
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A   p-value
Raw                   0.9255129 model1 > model2 0.1773496
AIC-corrected        -0.9251764 model2 > model1 0.1774371
BIC-corrected        -2.3644093 model2 > model1 0.0090294
```

```r
# 檢驗 overdispersion
dispersiontest(poi2_dragon)
```

```
    Overdispersion test

data:  poi2_dragon
z = 1.8727, p-value = 0.03055
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  2.104706
```

```r
##### 台鋼 #####
poi2_hawk <- glm(score ~ OBP + ISO + wOBA + RC +`BB%` +`K%`, family = poisson, data = poi_hawk)
summary(poi2_hawk)
```

```
Call:
glm(formula = score ~ OBP + ISO + wOBA + RC + `BB%` + `K%`, family = poisson,
    data = poi_hawk)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  23.111654   7.537045   3.066  0.00217 **
OBP          15.652332 150.958217   0.104  0.91742
ISO          28.663938  50.188209   0.571  0.56791
wOBA        -77.836094 162.185395  -0.480  0.63128
RC            0.021350   0.008136   2.624  0.00868 **
`BB%`        -0.236450   0.426361  -0.555  0.57918
`K%`         -0.427833   0.158949  -2.692  0.00711 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 69.696  on 34  degrees of freedom
Residual deviance: 53.969  on 28  degrees of freedom
AIC: 161.97
```

```
Number of Fisher Scoring iterations: 5
```

```
car::vif(poi2_hawk)
```

```
        OBP         ISO        wOBA          RC       ‘BB%‘        ‘K%‘
2127.194508   47.771147 2222.737664    9.869817    4.724111   15.936381
```

```
# 檢驗零膨脹
zip_mod <- zeroinfl(score ~ OBP + ISO + wOBA + RC +`BB%` +`K%`
                    | OBP + ISO + wOBA + RC +`BB%` +`K%` , data = poi_hawk)
pois_mod <- glm(score ~ OBP + ISO + wOBA + RC +`BB%` +`K%` , family = poisson, data = poi_hawk)
vuong(zip_mod, pois_mod)
```

```
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A  p-value
Raw                   2.1959478 model1 > model2 0.014048
AIC-corrected         1.0922338 model1 > model2 0.137365
BIC-corrected         0.2339041 model1 > model2 0.407530
```

```
# 檢驗 overdispersion
dispersiontest(poi2_hawk)
```

```
    Overdispersion test

data:  poi2_hawk
z = 0.92964, p-value = 0.1763
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.256098
```

```
##### 樂天 #####
poi2_monkey <- glm(score ~ OBP + ISO + `K%` + `BB%` + RC +
                     `先發 ERA(對手)` + `非先發 WHIP(對手)`, family = poisson, data = poi_monkey)
summary(poi2_monkey)
```

```
Call:
glm(formula = score ~ OBP + ISO + ‘K%‘ + ‘BB%‘ + RC + ‘先發ERA(對手)‘ +
    ‘非先發WHIP(對手)‘, family = poisson, data = poi_monkey)

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          5.301230   4.578472   1.158   0.2469
OBP                -10.219180   8.708308  -1.173   0.2406
ISO                 -6.883914  12.269820  -0.561   0.5748
‘K%‘                -0.078300   0.170576  -0.459   0.6462
‘BB%‘               -0.068514   0.164144  -0.417   0.6764
RC                   0.003262   0.004736   0.689   0.4910
‘先發ERA(對手)‘      0.065891   0.048720   1.352   0.1762
‘非先發WHIP(對手)‘   0.925291   0.435922   2.123   0.0338 *
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for poisson family taken to be 1)
```

```
    Null deviance: 61.541  on 32  degrees of freedom
Residual deviance: 48.626  on 25  degrees of freedom
AIC: 161.91

Number of Fisher Scoring iterations: 5
```

```
car::vif(poi2_monkey)
```

```
             OBP              ISO             `K%`             `BB%`
        1.572400         3.530503         9.962321         8.998734
              RC `先發ERA(對手)` `非先發WHIP(對手)`
        3.368230         1.172028         1.906193
```

```
# 檢驗零膨脹
zip_mod <- zeroinfl(score ~ OBP + ISO + `K%` + `BB%` + RC +
                      `先發 ERA(對手)` + `非先發 WHIP(對手)`
                    | OBP + ISO + `K%` + `BB%` + RC +
                      `先發 ERA(對手)` + `非先發 WHIP(對手)` , data = poi_monkey)
pois_mod <- glm(score ~ OBP + ISO + `K%` + `BB%` + RC +
                  `先發 ERA(對手)` + `非先發 WHIP(對手)` , family = poisson, data = poi_monkey)
vuong(zip_mod, pois_mod)
```

```
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A   p-value
Raw                    1.063643 model1 > model2 0.1437453
AIC-corrected         -1.237988 model2 > model1 0.1078602
BIC-corrected         -2.960193 model2 > model1 0.0015372
```

```
# 檢驗 overdispersion
dispersiontest(poi2_monkey)
```

```
    Overdispersion test

data:  poi2_monkey
z = 0.9255, p-value = 0.1774
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.283633
```

```
##### 富邦 #####
poi2_fubon <- glm(score ~ OBP + ISO + wOBA + `BB%`+
                    近 10 場勝場 + `非先發 WHIP(對手)`, family = poisson, data = poi_fubon)
summary(poi2_fubon)
```

```
Call:
glm(formula = score ~ OBP + ISO + wOBA + `BB%` + 近10場勝場 +
    `非先發WHIP(對手)`, family = poisson, data = poi_fubon)

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)          2.63557    2.94395   0.895    0.371
OBP                -69.59964  146.83933  -0.474    0.636
```

```
ISO                -57.92048   55.28880  -1.048    0.295
wOBA                92.03392  165.07603   0.558    0.577
`BB%`              -0.33913    0.41288  -0.821    0.411
近10場勝場          -0.06985    0.15115  -0.462    0.644
`非先發WHIP(對手)`   0.66201    0.43828   1.510    0.131

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 54.304  on 31  degrees of freedom
Residual deviance: 41.593  on 25  degrees of freedom
AIC: 140.3

Number of Fisher Scoring iterations: 5
```

```
car::vif(poi2_fubon)
```

```
              OBP                ISO               wOBA              `BB%`
      1997.517182         124.906333        3208.576383           7.266891
       近10場勝場 `非先發WHIP(對手)`
         1.737186           1.223795
```

```
# 檢驗零膨脹
zip_mod <- zeroinfl(score ~ OBP + ISO + wOBA + `BB%`+
                      近 10 場勝場 + `非先發 WHIP(對手)`
                    | OBP + ISO + wOBA + `BB%`+
                      近 10 場勝場 + `非先發 WHIP(對手)` , data = poi_fubon)
pois_mod <- glm(score ~ OBP + ISO + wOBA + `BB%`+
                  近 10 場勝場 + `非先發 WHIP(對手)` , family = poisson, data = poi_fubon)
vuong(zip_mod, pois_mod)
```

```
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A  p-value
Raw                    1.943173 model1 > model2 0.025998
AIC-corrected         -0.180541 model2 > model1 0.428364
BIC-corrected         -1.736943 model2 > model1 0.041199
```

```
# 檢驗 overdispersion
dispersiontest(poi2_fubon)
```

```
    Overdispersion test

data:  poi2_fubon
z = 0.68433, p-value = 0.2469
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.175487
```

經過對各隊進行 Zero-Inflation 及 Over-dispersion 的檢驗後，結果為:

(1) 六隊均無 Zero-Inflation 的問題

(2) 僅兄弟跟味全有 Over-dispersion，因此這兩隊使用負二項迴歸模型

9.4 Backward Seletion

- 對兄弟跟味全使用負二項迴歸模型

```
nb_bro <- glm.nb(score ~ OBP + ISO + wOBA + `先發 ERA(對手)` + `非先發 FIP(對手)`, data = poi_bro)
summary(nb_bro)
```

```
Call:
glm.nb(formula = score ~ OBP + ISO + wOBA + ‘先發ERA(對手)‘ +
    ‘非先發FIP(對手)‘, data = poi_bro, init.theta = 6.660330084,
    link = log)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)         9.25637    3.67687   2.517  0.01182 *
OBP                31.86106   51.95221   0.613  0.53969
ISO                30.04242   47.03557   0.639  0.52301
wOBA              -67.24904   67.40207  -0.998  0.31841
‘先發ERA(對手)‘     0.21325    0.07601   2.806  0.00502 **
‘非先發FIP(對手)‘  -0.33552    0.26387  -1.272  0.20354
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for Negative Binomial(6.6603) family taken to be 1)

    Null deviance: 51.956  on 32  degrees of freedom
Residual deviance: 36.362  on 27  degrees of freedom
AIC: 160.34

Number of Fisher Scoring iterations: 1

              Theta:  6.66
          Std. Err.:  4.44

 2 x log-likelihood:  -146.343
```

```
nb_dragon <- glm.nb(score ~ OBP + ISO + RC + `K%` + `非先發 FIP(對手)`, data = poi_dragon)
summary(nb_dragon)
```

```
Call:
glm.nb(formula = score ~ OBP + ISO + RC + ‘K%‘ + ‘非先發FIP(對手)‘,
    data = poi_dragon, init.theta = 3.415518606, link = log)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)         1.610971   4.110753   0.392 0.695138
OBP                -8.664887  17.231994  -0.503 0.615079
ISO                 6.487991  22.262758   0.291 0.770724
RC                 -0.015814   0.008395  -1.884 0.059600 .
‘K%‘                0.381027   0.108901   3.499 0.000467 ***
‘非先發FIP(對手)‘ -0.922363   0.358940  -2.570 0.010179 *
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for Negative Binomial(3.4155) family taken to be 1)
```

```
    Null deviance: 53.513  on 34  degrees of freedom
Residual deviance: 37.052  on 29  degrees of freedom
AIC: 168.04

Number of Fisher Scoring iterations: 1

              Theta:  3.42
          Std. Err.:  1.73

 2 x log-likelihood:  -154.045
```

Backward Seletion

- 從以下結果可以發現每隊得分的變數都不一樣，可能跟每隊有自己的戰術執行、選手特質有關。

(1) 兄弟

```
##### 兄弟 #####
poi2_bro1 <- step(nb_bro, direction = "backward")
```

```
Start:  AIC=158.34
score ~ OBP + ISO + wOBA + `先發ERA(對手)` + `非先發FIP(對手)`

                     Df Deviance    AIC
- OBP                 1   36.739 156.72
- ISO                 1   36.791 156.77
- wOBA                1   37.364 157.34
- `非先發FIP(對手)`   1   37.934 157.91
<none>                    36.362 158.34
- `先發ERA(對手)`     1   43.900 163.88

Step:  AIC=156.71
score ~ ISO + wOBA + `先發ERA(對手)` + `非先發FIP(對手)`

                     Df Deviance    AIC
- ISO                 1   36.132 154.78
- `非先發FIP(對手)`   1   37.738 156.39
<none>                    36.063 156.71
- wOBA                1   38.680 157.33
- `先發ERA(對手)`     1   44.398 163.05

Step:  AIC=154.78
score ~ wOBA + `先發ERA(對手)` + `非先發FIP(對手)`

                     Df Deviance    AIC
- `非先發FIP(對手)`   1   37.774 154.39
<none>                    36.166 154.78
- wOBA                1   39.368 155.98
- `先發ERA(對手)`     1   45.275 161.89

Step:  AIC=154.37
score ~ wOBA + `先發ERA(對手)`

                   Df Deviance    AIC
<none>                  36.668 154.37
- wOBA              1   43.931 159.63
- `先發ERA(對手)`   1   44.273 159.97
```

```
summary(poi2_bro1)
```

```
Call:
glm.nb(formula = score ~ wOBA + `先發ERA(對手)`, data = poi_bro,
    init.theta = 5.761999247, link = log)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)       10.84776    3.57766   3.032  0.00243 **
wOBA             -33.56575   12.05250  -2.785  0.00535 **
`先發ERA(對手)`    0.20639    0.07297   2.829  0.00468 **
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(5.762) family taken to be 1)

    Null deviance: 49.452  on 32  degrees of freedom
Residual deviance: 36.668  on 30  degrees of freedom
AIC: 156.37

Number of Fisher Scoring iterations: 1

              Theta:  5.76
          Std. Err.:  3.55

 2 x log-likelihood:  -148.368
```

(2) 味全

```
##### 味全 #####
poi2_dragon1 <- step(nb_dragon, direction = "backward")
```

```
Start:  AIC=166.04
score ~ OBP + ISO + RC + `K%` + `非先發FIP(對手)`

                    Df Deviance    AIC
- ISO                1   37.130 164.12
- OBP                1   37.270 164.26
<none>                   37.052 166.04
- RC                 1   40.458 167.45
- `非先發FIP(對手)`  1   44.159 171.15
- `K%`               1   49.239 176.23

Step:  AIC=164.12
score ~ OBP + RC + `K%` + `非先發FIP(對手)`

                    Df Deviance    AIC
- OBP                1   37.439 162.40
<none>                   37.160 164.12
- `非先發FIP(對手)`  1   44.201 169.16
- RC                 1   46.233 171.20
- `K%`               1   50.416 175.38

Step:  AIC=162.4
score ~ RC + `K%` + `非先發FIP(對手)`

                    Df Deviance    AIC
<none>                   37.246 162.40
- `非先發FIP(對手)`  1   44.954 168.11
- RC                 1   46.445 169.60
- `K%`               1   51.222 174.38
```

```
summary(poi2_dragon1)
```

```
Call:
glm.nb(formula = score ~ RC + `K%` + `非先發FIP(對手)`,
    data = poi_dragon, init.theta = 3.380434311, link = log)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)       -0.343939   1.513738  -0.227 0.820259
RC                -0.012800   0.004538  -2.821 0.004790 **
`K%`               0.347755   0.099468   3.496 0.000472 ***
`非先發FIP(對手)` -0.941803   0.355121  -2.652 0.008000 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(3.3804) family taken to be 1)

    Null deviance: 53.263  on 34  degrees of freedom
Residual deviance: 37.246  on 31  degrees of freedom
AIC: 164.4

Number of Fisher Scoring iterations: 1
```

```
            Theta:  3.38
        Std. Err.:  1.72

2 x log-likelihood:  -154.401
```

(3) 台鋼

```
##### 台鋼 #####
poi2_hawk1 <- step(poi2_hawk, direction = "backward")
```

```
Start:  AIC=161.97
score ~ OBP + ISO + wOBA + RC + `BB%` + `K%`

        Df Deviance    AIC
- OBP    1   53.980 159.98
- wOBA   1   54.200 160.20
- `BB%`  1   54.274 160.28
- ISO    1   54.294 160.30
<none>       53.969 161.97
- `K%`   1   61.302 167.30
- RC     1   61.367 167.37

Step:  AIC=159.98
score ~ ISO + wOBA + RC + `BB%` + `K%`

        Df Deviance    AIC
- `BB%`  1   54.528 158.53
<none>       53.980 159.98
- ISO    1   57.090 161.09
- `K%`   1   63.827 167.83
- RC     1   64.325 168.33
- wOBA   1   64.690 168.69

Step:  AIC=158.53
score ~ ISO + wOBA + RC + `K%`

       Df Deviance    AIC
<none>      54.528 158.53
- ISO   1   58.058 160.06
- `K%`  1   63.966 165.97
- wOBA  1   64.696 166.70
- RC    1   64.913 166.91
```

```
summary(poi2_hawk1)
```

```
Call:
glm(formula = score ~ ISO + wOBA + RC + `K%`, family = poisson,
    data = poi_hawk)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  20.865573   6.123879   3.407 0.000656 ***
ISO          25.361963  13.305823   1.906 0.056640 .
wOBA        -58.098998  17.576212  -3.306 0.000948 ***
RC            0.019103   0.005934   3.219 0.001285 **
`K%`         -0.404267   0.131751  -3.068 0.002152 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 69.696  on 34  degrees of freedom
```

```
Residual deviance: 54.528  on 30  degrees of freedom
AIC: 158.53

Number of Fisher Scoring iterations: 5
```

```r
# 做 overdispersion 檢定
dispersiontest(poi2_hawk1)
```

```
    Overdispersion test

data:  poi2_hawk1
z = 0.86705, p-value = 0.193
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.235141
```

(4) 樂天

```
##### 樂天 #####
poi2_monkey1 <- step(poi2_monkey, direction = "backward")
```

```
Start:  AIC=161.91
score ~ OBP + ISO + `K%` + `BB%` + RC + `先發ERA(對手)` +
    `非先發WHIP(對手)`

                      Df Deviance    AIC
- `BB%`                1   48.805 160.09
- `K%`                 1   48.843 160.13
- ISO                  1   48.942 160.23
- RC                   1   49.101 160.39
- OBP                  1   49.973 161.26
- `先發ERA(對手)`      1   50.410 161.69
<none>                     48.626 161.91
- `非先發WHIP(對手)`   1   53.023 164.31

Step:  AIC=160.09
score ~ OBP + ISO + `K%` + RC + `先發ERA(對手)` + `非先發WHIP(對手)`

                      Df Deviance    AIC
- `K%`                 1   48.845 158.13
- ISO                  1   48.952 158.24
- RC                   1   49.164 158.45
- OBP                  1   50.401 159.69
- `先發ERA(對手)`      1   50.494 159.78
<none>                     48.805 160.09
- `非先發WHIP(對手)`   1   53.023 162.31

Step:  AIC=158.13
score ~ OBP + ISO + RC + `先發ERA(對手)` + `非先發WHIP(對手)`

                      Df Deviance    AIC
- ISO                  1   48.962 156.25
- RC                   1   49.227 156.51
- OBP                  1   50.401 157.69
- `先發ERA(對手)`      1   50.534 157.82
<none>                     48.845 158.13
- `非先發WHIP(對手)`   1   53.403 160.69

Step:  AIC=156.25
score ~ OBP + RC + `先發ERA(對手)` + `非先發WHIP(對手)`

                      Df Deviance    AIC
- RC                   1   49.230 154.51
<none>                     48.962 156.25
- OBP                  1   51.126 156.41
- `先發ERA(對手)`      1   51.255 156.54
- `非先發WHIP(對手)`   1   55.091 160.38

Step:  AIC=154.51
score ~ OBP + `先發ERA(對手)` + `非先發WHIP(對手)`

                      Df Deviance    AIC
<none>                     49.230 154.51
```

```
- ‘先發ERA(對手)‘     1   51.408 154.69
- OBP                 1   53.342 156.63
- ‘非先發WHIP(對手)‘  1   55.393 158.68
```

```
summary(poi2_monkey1)
```

```

Call:
glm(formula = score ~ OBP + ‘先發ERA(對手)‘ + ‘非先發WHIP(對手)‘,
    family = poisson, data = poi_monkey)

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)          4.58034    2.29886   1.992   0.0463 *
OBP                -13.60717    6.97443  -1.951   0.0511 .
‘先發ERA(對手)‘      0.06687    0.04463   1.498   0.1340
‘非先發WHIP(對手)‘   0.76453    0.31895   2.397   0.0165 *
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 61.541  on 32  degrees of freedom
Residual deviance: 49.230  on 29  degrees of freedom
AIC: 154.51

Number of Fisher Scoring iterations: 5
```

```
# 做 overdispersion 檢定
dispersiontest(poi2_monkey1)
```

```

    Overdispersion test

data:  poi2_monkey1
z = 1.003, p-value = 0.1579
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.320832
```

(5) 富邦

```
##### 富邦 #####
poi2_fubon1 <- step(poi2_fubon, direction = "backward")
```

```
Start:  AIC=140.3
score ~ OBP + ISO + wOBA + `BB%` + 近10場勝場 + `非先發WHIP(對手)`

                     Df Deviance    AIC
- 近10場勝場          1   41.807 138.51
- OBP                 1   41.819 138.53
- wOBA                1   41.907 138.62
- `BB%`               1   42.267 138.97
- ISO                 1   42.717 139.43
<none>                    41.593 140.30
- `非先發WHIP(對手)`  1   44.060 140.77

Step:  AIC=138.52
score ~ OBP + ISO + wOBA + `BB%` + `非先發WHIP(對手)`

                     Df Deviance    AIC
- `BB%`               1   42.288 137.00
- OBP                 1   42.374 137.08
- wOBA                1   42.554 137.26
- ISO                 1   43.795 138.50
<none>                    41.807 138.51
- `非先發WHIP(對手)`  1   44.355 139.06

Step:  AIC=137
score ~ OBP + ISO + wOBA + `非先發WHIP(對手)`

                     Df Deviance    AIC
<none>                    42.288 137.00
- `非先發WHIP(對手)`  1   44.909 137.62
- OBP                 1   46.171 138.88
- wOBA                1   46.747 139.46
- ISO                 1   47.412 140.12
```

```
summary(poi2_fubon1)
```

```

Call:
glm(formula = score ~ OBP + ISO + wOBA + `非先發WHIP(對手)`,
    family = poisson, data = poi_fubon)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)          1.2355     2.4025   0.514   0.6071
OBP               -170.9508    85.0527  -2.010   0.0444 *
ISO                -90.9359    39.8014  -2.285   0.0223 *
wOBA               206.2086    95.7813   2.153   0.0313 *
`非先發WHIP(對手)`   0.6840     0.4384   1.560   0.1187
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 54.304  on 31  degrees of freedom
```

```
Residual deviance: 42.288  on 27  degrees of freedom
AIC: 137

Number of Fisher Scoring iterations: 5
```

```r
# 做 overdispersion 檢定
dispersiontest(poi2_fubon1)
```

```
    Overdispersion test

data:  poi2_fubon1
z = 0.78405, p-value = 0.2165
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.218726
```

(6) 統一

```
##### 統一 #####
poi2_lion1 <- step(poi2_lion, direction = "backward")
```

```
Start:  AIC=165.97
score ~ OBP + ISO + wOBA + `K%` + `BB%` + `先發ERA(對手)` +
    `非先發WHIP(對手)`

                     Df Deviance    AIC
- OBP                 1   46.510 163.97
- wOBA                1   46.561 164.03
- ISO                 1   46.657 164.12
- `BB%`               1   46.764 164.23
- `先發ERA(對手)`     1   48.076 165.54
<none>                    46.508 165.97
- `K%`                1   54.461 171.93
- `非先發WHIP(對手)`  1   65.888 183.35

Step:  AIC=163.97
score ~ ISO + wOBA + `K%` + `BB%` + `先發ERA(對手)` + `非先發WHIP(對手)`

                     Df Deviance    AIC
- `BB%`               1   47.194 162.66
- ISO                 1   47.286 162.75
- `先發ERA(對手)`     1   48.117 163.58
<none>                    46.510 163.97
- wOBA                1   49.739 165.20
- `K%`                1   56.230 171.69
- `非先發WHIP(對手)`  1   65.889 181.35

Step:  AIC=162.66
score ~ ISO + wOBA + `K%` + `先發ERA(對手)` + `非先發WHIP(對手)`

                     Df Deviance    AIC
- ISO                 1   47.587 161.05
- `先發ERA(對手)`     1   48.474 161.94
<none>                    47.194 162.66
- wOBA                1   50.187 163.65
- `K%`                1   56.245 169.71
- `非先發WHIP(對手)`  1   67.615 181.08

Step:  AIC=161.05
score ~ wOBA + `K%` + `先發ERA(對手)` + `非先發WHIP(對手)`

                     Df Deviance    AIC
- `先發ERA(對手)`     1   48.995 160.46
<none>                    47.587 161.05
- wOBA                1   50.426 161.89
- `K%`                1   59.342 170.81
- `非先發WHIP(對手)`  1   68.119 179.58

Step:  AIC=160.46
score ~ wOBA + `K%` + `非先發WHIP(對手)`

                     Df Deviance    AIC
<none>                    48.995 160.46
```

```
- wOBA                1   52.842 162.31
- `K%`               1   60.243 169.71
- `非先發WHIP(對手)`  1   72.872 182.34
```

```
summary(poi2_lion1)
```

```

Call:
glm(formula = score ~ wOBA + `K%` + `非先發WHIP(對手)`,
    family = poisson, data = poi_lion)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)         -4.4232     3.0446  -1.453  0.14628
wOBA               -10.9051     5.6226  -1.940  0.05244 .
`K%`                 0.3952     0.1204   3.283  0.00103 **
`非先發WHIP(對手)`   2.4821     0.5108   4.860 1.18e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 78.562  on 33  degrees of freedom
Residual deviance: 48.995  on 30  degrees of freedom
AIC: 160.46

Number of Fisher Scoring iterations: 5
```

```
# 做 overdispersion 檢定
dispersiontest(poi2_lion1)
```

```

    Overdispersion test

data:  poi2_lion1
z = 1.2995, p-value = 0.09688
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.348383
```

9.5 實例

預測 6/1 (客: 樂天 vs 主: 統一) 的比賽

- 樂天: 需要自己的 OBP(0.328)，對手的 ERA(3.02)、WHIP(1.347)
- 統一: 需要自己的 wOBA(0.322)、K%(15)，對手的 WHIP(1.146)

```
##### 統一 #####
names(poi_lion)[12] <- "K"
names(poi_lion)[17] <- "WHIP"
poi2_lion <- glm(score ~ wOBA + K + WHIP, family = poisson, data = poi_lion)
summary(poi2_lion)
```

```
Call:
glm(formula = score ~ wOBA + K + WHIP, family = poisson, data = poi_lion)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -4.4232     3.0446  -1.453  0.14628
wOBA        -10.9051     5.6226  -1.940  0.05244 .
K             0.3952     0.1204   3.283  0.00103 **
WHIP          2.4821     0.5108   4.860 1.18e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 78.562  on 33  degrees of freedom
Residual deviance: 48.995  on 30  degrees of freedom
AIC: 160.46

Number of Fisher Scoring iterations: 5
```

```
##### 樂天 #####
names(poi_monkey)[17] <- "WHIP"
names(poi_monkey)[15] <- "ERA"
poi2_monkey <- glm(score ~ OBP + ERA + WHIP, family = poisson, data = poi_monkey)
summary(poi2_monkey)
```

```
Call:
glm(formula = score ~ OBP + ERA + WHIP, family = poisson, data = poi_monkey)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   4.58034    2.29886   1.992   0.0463 *
OBP         -13.60717    6.97443  -1.951   0.0511 .
ERA           0.06687    0.04463   1.498   0.1340
WHIP          0.76453    0.31895   2.397   0.0165 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 61.541  on 32  degrees of freedom
Residual deviance: 49.230  on 29  degrees of freedom
AIC: 154.51
```

```
Number of Fisher Scoring iterations: 5
```

```r
# 預測 6/1（客：樂天 vs 主：統一）

new_lion <- data.frame(
   wOBA = 0.322,
   K = 15,
   WHIP = 1.146
)

new_monkey <- data.frame(
   OBP = 0.328,
   ERA =  3.02,
   WHIP = 1.347
)

pred_prob_lion <- predict(poi2_lion, newdata = new_lion, type = "response")
print(pred_prob_lion)
```

```
       1
2.312823
```

```r
pred_prob_monkey <- predict(poi2_monkey, newdata = new_monkey, type = "response")
print(pred_prob_monkey)
```

```
       1
3.853357
```

最後可以得到模型預測:

- 樂天: 約得 3.8 分
- 統一: 約得 2.3 分

10 結論與未來展望

在 **Logistic Regression** 中，雖然打擊能力對於比賽勝負的預測效果並不顯著，但這並不代表打擊表現在比賽中不重要，而是說明在現有模型與變數設定下，投手指標在統計上更能解釋勝負的變異，未來可考慮更細緻的打擊數據 (如擊球初速、擊球距離等) 納入分析。

在 **Bardley Terry Model** 及 **Elo-Rating System** 中，不僅考量了棒球比賽中是兩兩對戰的性質，還考慮到任意兩隊的利歷史對戰戰績，進而估算出各個球隊的能力值並作為變數去預測勝率。在 **Bardley Terry Model**，我們只放了主客場及先發投手 **ERA**，雖然相較於一般的 **BTM** 已經多了很多資訊，但對於棒球比賽仍遠遠不夠，而 **Elo-Rating System** 則是反映動態實力評估，但同樣的，我們所考慮的資訊對於預測勝率仍有所不足。

在 **Poisson Regression** 中，雖說顯著的係數變多，使得我們在預測時能給予更多的資訊，但估計出的變數係數有些是不太合理的，因而導致我們在預測的時候結果並不是很好，但可能是由於我們拆分成 **6** 隊分別建模，而一隊的樣本數只有約 **30** 場，若未來使用整季的資料去分析，在樣本數足夠的情況下，或許這個方法能做得更好。

對於上述方法，以現有資料來預測勝負仍有改善的空間，在棒球比賽中，雖說各打擊數據對於比賽結果並未顯著，但評估一名打者 (一個隊伍) 的打擊能力也並非只看單一打擊指標，而是綜合起來比較，也許將代表不同能力的打擊指標提取出來，或是將所有打擊指標降維，可能就能讓模型考慮到打擊的表現。因此若是只需預測，除了增加有用的變數外，也可搭配主成分分析 (**PCA**) 等方法去改善我們的模型。

11 分工表

項目	負責人員
收集資料	陳志恆、林宇軒、余振瑋
討論	洪毅荃、郭懿葶、余振瑋、陳志恆、林宇軒
Logistic Regression	洪毅荃
Bradley Terry Model	余振瑋
Elo Rating System	郭懿葶
Poisson Regression	洪毅荃
PPT	洪毅荃、郭懿葶、余振瑋
口頭報告	洪毅荃、郭懿葶、余振瑋
書面報告	洪毅荃、郭懿葶、余振瑋
coding 顧問	熱狗 (第六組)

12 Reference

(1) 野球革命

(2) 中華職棒大聯盟全球資訊網

(3) 台灣運彩

(4) Agresti, A. (2018). An Introduction to Categorical Data Analysis (3rd ed.). Hoboken, NJ: Wiley.