統計學習理論簡介（PS5696）期末專題

B09302118　政治三　黃柏叡

壹、前言

棒球比賽中，在零好三壞且壘上無人的情況下，投手傾向丟出好球，避免四壞球保送，而筆者很好奇主審與投手間是否隱約有某種默契，使得主審對好球帶的判斷較為寬容，本文想透過統計學習模型，找出有甚麼特徵會使得主審裁判將好球帶以外的球判為好球？

在零好三壞且壘上無人時，筆者認為影響主審裁判公正判決的特徵，必須納入對戰投手、打者各自的進階數據，當然還有當下該球的進階數據，以及當下的賽況資料，一方面在投手配球策略上會針對打者、賽況進行評估決策，另一方面球本身的路徑、軌跡、球種、轉速，乃至於主審對於投手與打者的印象都有可能影響判決。

關於本文章節安排，首先將從資料取得來源與如何前處理開始說明，其次筆者將以 LASSO 正規化模型與主成分分析進行特徵選擇，並討論篩選出的特徵與依變數間的關聯，最後再將前段篩出的特徵作為自變數，套入 Logistic Regression, Probit Regression, Decision Tree, Random Forest, Support Vector Machine, Gradient Boost Machine 等等模型，對資料進行建模與預測，分析各個模型的預測準確率。

貳、資料來源與資料處理

在美國大聯盟的數據庫網站上，筆者利用設定搜尋功能，採用零好三壞、無人在壘、主審裁決為好球或壞球的逐球紀錄，並且將數據庫提供的資料下載為 csv 檔，在此檔案中，包含逐球的詳細記錄，諸如偵測球種、進壘點、球速、轉速、位移等等，另外也有當下局數、投手與打者等資料。

由於筆者猜測當下投手與打者的特徵可能也可能會造成影響，因此後續再以 Python Selenium 工具針對逐球紀錄中的投手與打者進行網路爬蟲，蒐集各自的 2022 賽季之進階數據為特徵，最後再將爬蟲得出的投手與打者進階數據，合併原先下載的逐球紀錄。

值得留意的是，筆者所定義的依變數為二元變數，1 表示為宣判好球但實際進壘點在好球帶之外的樣本，也有特別針對進壘點與偵測球種進行 one hot encoding，最終產生一份共 3415 筆樣本 87 個特徵的資料檔，表一為詳細的特徵欄位與說明，以上步驟皆以 Python 進行處理，接著將大致清理完的資料引入 R。

首先，先處理有關缺失值的部分，筆者發現 release_spin_rate 中有 9 筆缺失，release_extension 有 5 筆缺失，spin_axis 有 9 筆缺失，BB% 有 431 筆，保送率的部分稍微嚴重，筆者回溯資料比對，並不是爬蟲與合

併資料時所造成，保送率的缺失值疑慮，確實有待討論，不過為求分析便利，筆者姑且選擇將所有的缺失值，以該欄位完整案例之期望值填入。

其次，排除一些與分析結果無關或有其餘相似且可取代之欄位，如 X（以 Python 將資料匯出時自動產生的編碼）、description、stand、p_throws、inning_topbot、Name 以及 Pitcher。最後，確定丟入 LASSO 之前的資料有 3415 筆樣本，共計 79 個特徵欄位。

表一：特徵欄位與說明

特徵	說明	特徵	說明
result	是否誤判	Description	文字判決結果
release_speed	出手速度	Stand	打者站位
release_pos_x		p_throws	投手慣用手
release_pos_y	捕手視角之投手出手點	same_handed	打者與投手慣用手相同
release_pos_z		outs_when_up	當下出局數
release_extension	出手延伸	Inning	局數
release_spin_rate	轉速	inning_topbot	上下半場
spin_axis	2D 旋轉角度	at_bat_number	比賽當下累積壘打數
effective_speed	根據出手延伸調整之球速	home_score	主隊分數
delta_home_win_exp	判決前後主隊勝率差亦	away_score	客隊分數
delta_run_exp	判決前後得分期望值差異	score_differential	分差
Name	打者	Pitcher	投手
ERA	投手自責分率	x_ERA	投手期望自責分率
以下特徵打者與投手皆有，在名稱前加入 p_ 為投手數據，如 p_Age 為投手年齡			
Age	年齡	Sweet_Spot%	擊球（投球）甜蜜帶率
Pitches	投球數	XBA	期望打擊率
Batted_Balls	擊球數	XSLG	期望長打率
Barrels	出色擊球數	WOBA	加權上壘率
Barrel%	出色擊球率	XWOBA	加權上壘率_1
Barrel/PA	出色擊球數 / 打席數	XWOBACON	加權上壘率_2
Exit_Velocity	平均擊球（投球）初速	HardHit%	強勁擊球率
Max_EV	最快初速	K%	三振率
Launch_Angle	擊球（出手）角度	BB%	保送率
偵測球種	pitch_type_CH 變速球	pitch_type_CU 曲球	pitch_type_SI 伸卡球
	pitch_type_EP 慢速球	pitch_type_SV 滑曲球	pitch_type_SL 滑球
	pitch_type_FC 卡特球	pitch_type_FF 直球	pitch_type_ST sweeper
	pitch_type_FS 指叉球	pitch_type_KC 圈指曲	pitch_type_FA 無歸類
進壘點 zone		好球 1 - 9 壞球 11 - 14	

參、特徵篩選

資料集共有 79 個特徵欄位，為了避免訓練組的過度擬合（overfitting），使得更重要的測試組的準確性下降，且實際上建模也用不到全部的特徵欄位，因此筆者認為應該先對特徵欄位進行篩選。

本文採取的特徵篩選模型為最小絕對值收斂和選擇算子（LASSO），透過最小化均方誤加上 L1 懲罰項（懲罰值乘上迴歸係數之絕對值），這會造成許多特徵欄位的迴歸係數被設定為零，從而找到非零的係數，進而能達成降維與特徵篩選的效果。

具體而言，本文使用 $glmnet()$ 將依變數與自變數套入，設定 $alpha = 1$，採用 LASSO 迴歸，然後進行特徵篩選時所採用 $cv.glmnet()\$lambda.1se$ 最佳 $lambda$，其中最佳 $lambda$ 為 0.01897096，最重要的是，根據 LASSO 模型所篩選出的特徵有 outs_when_up、delta_run_exp、pitch_type_CU、zone_4.0、zone_5.0、zone_6.0、zone_8.0、zone_11.0、zone_12.0、zone_13.0 與 zone_14.0，表二為篩選出的特徵之迴歸係數。

表二：LASSO 篩選特徵與迴歸係數

欄位	係數
outs_when_up	-0.00414
delta_run_exp	-2.14279
pitch_type_CU	0.13891
zone_4	-0.00359
zone_5	-0.00967
zone_6	-0.00452
zone_8	-0.00933
zone_11	0.41860
zone_12	0.44159
zone_13	0.41990
zone_14	0.40312

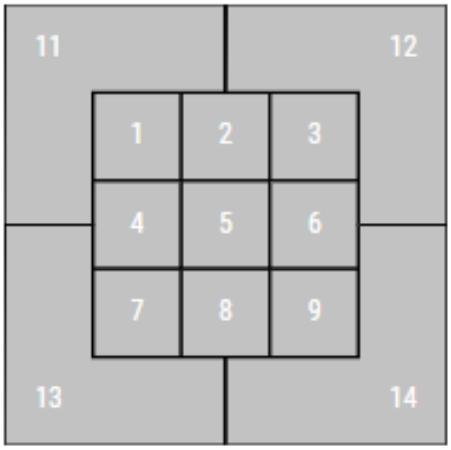
關於被篩選出特徵與依變數間的關聯，其實都滿直觀的。當下出局數越多時，有可能投手為了降低風險，因此傾向投出好球，即便進壘點可能不是好球帶中，或許差距也不遠，當然也有可能說明裁判想即早結束半局，因此對於好球帶的判定更為寬容，出現較不公正的判決。

Delta_run_exp 與好球帶的寬容判決必然是負相關，當好球帶判決較為公正時，則此時該名打者已經被四壞球保送上壘，得分機會增加，反之則降低。

進壘點的位置與依變數間也有相當合理的關聯，好球帶編碼請見圖

一，進壘點在好球帶中，不公正判決的機率降低，然而當進壘點偏離好球帶時，則不公正判決的機率提高，其中從這幾個編碼可以看出，好球帶4、5、6、8號位是比較精準而不易誤判的區塊，反觀好球帶外則是12號位的誤判機率最高，14號位的誤判機率最低，事實上是一個滿重要的發現。

圖一：好球帶編碼



筆者認為最有趣的發現莫過於，大聯盟裁判對於曲球的誤判率些微較高，換言之，當投手丟曲球時，即便在好球帶以外，但裁判仍有些微較高的機率將其判定為好球，根據筆者主觀的猜測，因為曲球的位移量較大，且進壘路徑與投手出手點存在頗大的視覺落差，導致主審在判定上難以固定，或許可說是當前人工判定好球帶的一項特點。

透過LASSO迴歸篩選出的11項特徵欄位，筆者又建構了一份新的資料集，以下的建模與分析，筆者除了以篩選過的新資料集進行預測之外，也會同時與沒有篩選特徵的模型比較預測力。

肆、建模

一、採用模型

由於資料集的依變數為二元變數，因此主要是採取分類模型，包含Logit、Probit、Support Vector Machine、Gradient Boost Machine、Decision Tree與Random Forest。另外，未經篩選之資料集作為對照組，則是選用Support Vector Machine。

除了單次測試組的預測正確率之外，也會以AUC比較各個模型之間的性能，當然也會比較篩選過後與未經篩選的模型預測性能。

二、資料切割與交叉驗證

在本文中為求便利性，訓練組與測試組的拆分比例固定為0.75，交叉驗證皆採取10折，順帶一提，若有需要設定隨機數$set.seed()$一律採取123。

三、引入模型

1. Logit 與 Probit

這兩個模型在應用上，不需要設定許多參數，公式為 $result \sim .$，直接將依變數對所有特徵欄位迴歸，$family = binomial$，唯一差別只在於使用 $glm()$ 函數時，Probit 需要特別註明 $link = "Probit"$ ，而 Logit 本身就是預設，因此不必特別調整。

2. Support Vector Machine

在套用模型之前，有特別先將依變數的型態調整為 factor，接著在參數設定上，$scale = True$，$kernel = "radial"$，設定多項式次數至多為三次，$cost\ function = 1$，輸出不需為機率預測值。

3. Gradient Boost Machine

在 Gradient Boost Machine 中，筆者有針對超參數進行自動調整，比如 $shrinkage$、$interaction.depth$、$n.minobsinnode$、$bag.fraction$，判斷預測結果是否進步的指標是平方均方誤（root mean square error），同時也會將 $optimal\_tree$ 記錄下來，調整後找到最佳參數組合為如表三。

表三：Gradient Boost Machine 超參數最適組合

超參數	最適組合
shrinkage	0.1
interaction.depth	5
n.minobsinnode	5
bag.fraction	0.5
optimal_trees	373

4. Decision Tree

套入 Decision Tree 模型前，先將依變數型別轉換為 factor，同樣有調整超參數，$minsplit$ 以及 $maxdepth$，比較基準是 complexity 以及 mean square error，調整後的最適組合請見表四。

表四：Decision Tree 超參數最適組合

超參數	最適組合
min_split	5
max_depth	8
complexity_para	0.01

5. Random Forest

基本上，由於 Random Forest 可以視為多重的 Decision Tree 組合而成，筆者的操作過程也同樣先轉換依變數型別，接著調整超參數，*mtry* 表示使用的特徵數量，*node_size* 表示節點的數量，*p* 表示每組 Decision Tree 的樣本比例，以 out-of-bag RMSE (OOB_rmse)的平方均方誤來評估模型預測性能的指標，調整後的最適組合請見表五。

表五：Random Forest 超參數最適組合

超參數	最適組合
mtry	4
node_size	3
p	0.5

伍、分析

一、正確率比較

資料切割比例為 0.75，因此訓練組共有 2562 筆樣本，測試組共有 853 筆樣本，將訓練資料引入各個模型設定來訓練模型，再將測試組資料引入訓練好的模型，因為依變數是二元變數，所以可以直接比較正確率，關於各個模型的正確率請見表六。

表六：模型正確率

模型	正確率
Logit	1
Probit	1
Support Vector Machine	1
Gradient Boost Machine	1
Decision Tree	1
Random Forest	1
Compared_SVM	0.936694

預測結果相當令人訝異，有經過特徵篩選的資料集，不管在任何一個模型中，正確率都能達到 100%，換言之，當球路的進壘點位於好球帶以外時，被判為好球的樣本全部都預測正確，而當球路的進壘點於好球帶外，被判為壞球的樣本，以及於好球帶內，被判為好球的樣本，都全部預測正確。反觀，未經特徵篩選的資料集，若採用 SVM 進行分類，正確率大約只有 94%，與經過特徵篩選再訓練的模型的效能

上存在落差。

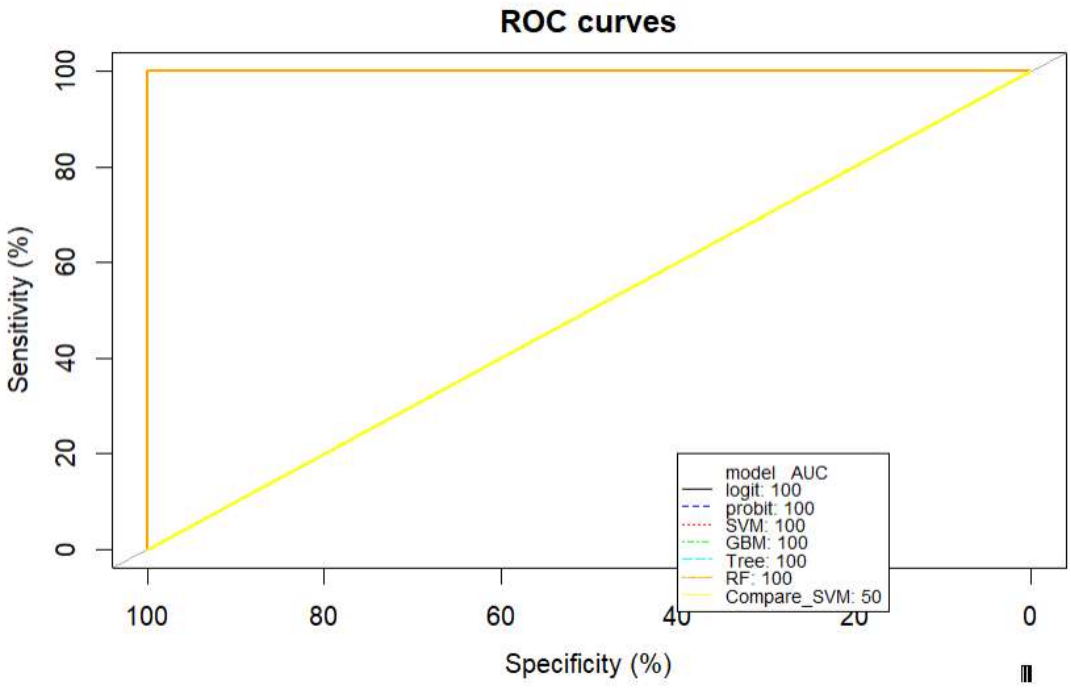
若實際觀察混淆矩陣和預測結果，就可以發現，在測試資料集中，實際資料一共有 799 個 0 與 54 個 1，經過特徵篩選的模型可以將 54 個 1 完全預測正確。

然而，未經特徵篩選的 SVM，將測資全部預測為 0，因此其中 799 個 0 預測正確，但是 54 個 0 預測錯誤，因此正確率才會只有 94%。樂觀地說，未經特徵篩選的模型，能夠明確地辨識公正的好壞球裁決，但是沒辦法正確地預測哪些壞球會被寬容而判成好球。

二、AUC 比較

ROC 曲線與 AUC 可以用來比較分類器模型的優劣，圖二為 ROC 曲線圖，同時也有顯示 AUC 數據，由於經過特徵篩選的各個模型 AUC 皆達到 100，因此 ROC 曲線疊合，不過，未經特徵篩選的 Support Vector Machine 在 AUC 只有 50，某程度上是一個不及格的模型，由此，亦可知特徵篩選確實有效提升模型的性能。

圖二：模型 ROC 曲線圖



三、迴歸係數與重要特徵值

最後，訓練模型中的迴歸係數，或者如何進行分類，如何應用特徵欄位，值得討論，除了 Support Vector Machine 之外，其餘的模型皆有納入本段內容。

一、Logit

首先，Logit 模型的迴歸係數請見表七，此處筆者並沒有要做假

設檢定，因此較單純討論迴歸係數的大小與方向。就係數與依變數的變動方向而言，與 LASSO 篩選特徵時的觀察基本上是一致的，出局數與依變數同向，判決前後得分期望值差異與依變數反向，曲球與依變數同向，好球帶以內進壘點與依變數反向，好球帶以外進壘點與依變數同向。論係數大小，又以判決前後得分期望值差異最為重要，其次則是好球帶以外的進壘點。

表七：Logit 迴歸係數

特徵欄位	迴歸係數
outs_when_up	15.1152013
delta_run_exp	-699.2750438
pitch_type_CU	4.9158852
zone_4	-0.1700082
zone_5	-0.197352
zone_6	-0.1785726
zone_8	-0.383808
zone_11	52.1836586
zone_12	52.3811138
zone_13	52.2513377
zone_14	52.2682433

二、Probit

事實上，Probit 模型的迴歸係數大小與方向，與 Logit 迴歸係數的推論是一致的，差別只是兩者的轉換 CDF 不同，至於預測結果都有達到 100 正確率。

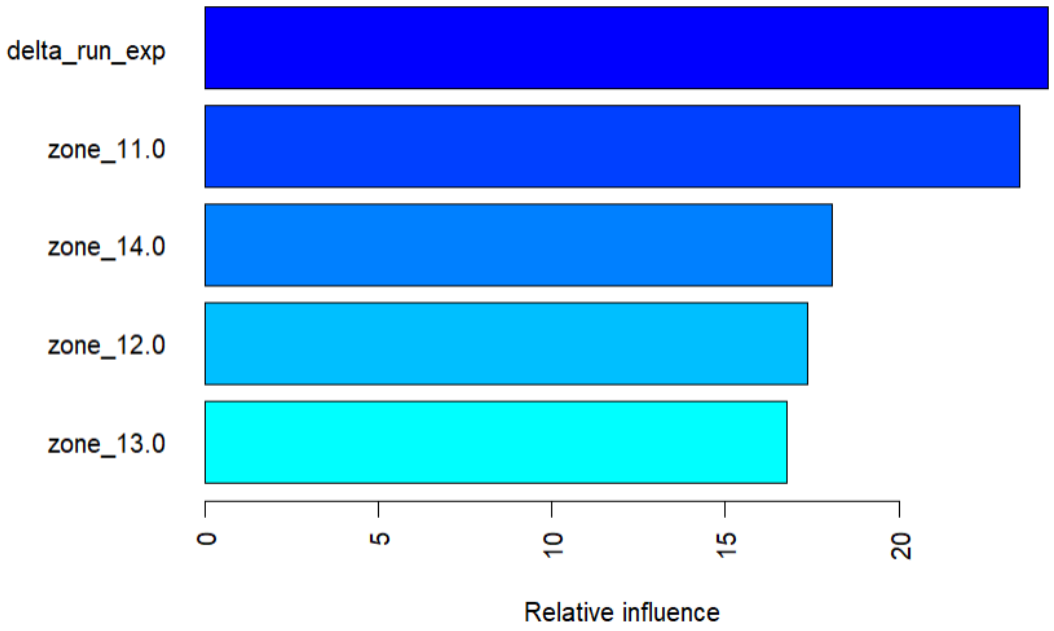
表八：Probit 迴歸係數

特徵欄位	迴歸係數
outs_when_up	4.52154487
delta_run_exp	-195.8011792
pitch_type_CU	0.82798299
zone_4	-0.02635914
zone_5	-0.02928888
zone_6	-0.02632436
zone_8	-0.05967798
zone_11	13.80566474
zone_12	13.83842804
zone_13	13.81631418
zone_14	13.81970489

三、Gradient Boost Machine

在 Gradient Boost Machine 中，有內建的功能可以查看特徵欄位的相對重要性，由於只有以下五個特徵欄位有被採用，因此相對重要性也就只有展示這五項的重要性，係數本身不是很重要，畢竟不能用來解釋、推論，筆者認為只需要知道相對大小即可，以圖三為例，在該模型中，delta_run_exp 是最重要的特徵，好球帶以外的進壘點次之，至於出局數和曲球則被省略。

圖三：Gradient Boost Machine Relative Influence



四、Decision Tree

筆者提供的是 Decision Tree 的決策路徑，請見圖四，可知此處採用的特徵欄位與 Gradient Boost Machine 相同，先從判決前後得分期望值差異開始判斷，若小於 0.0065，則必然為 0，接著再從進壘點 11、12、14、13 依序區別。

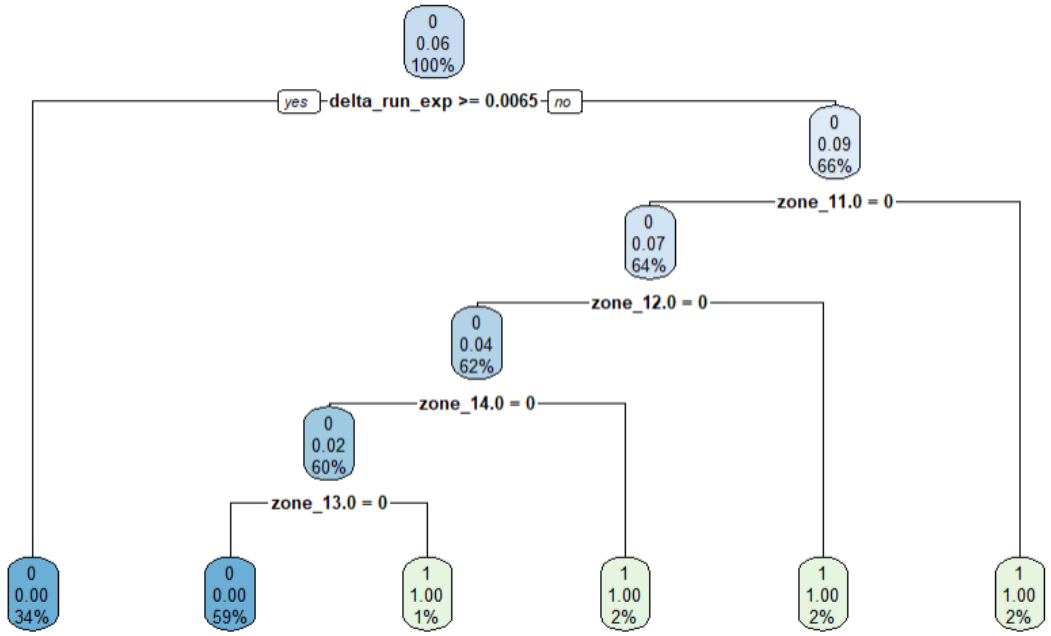
五、Random Forest

Random Forest 的重要特徵值，前五項與決策樹的決策路徑順序相同，後續還有再加入些微權重的曲球、好球帶內進壘點以及出局數，不過，其實除了前五項特徵值以外的特徵值，影響並不大，請見圖五。

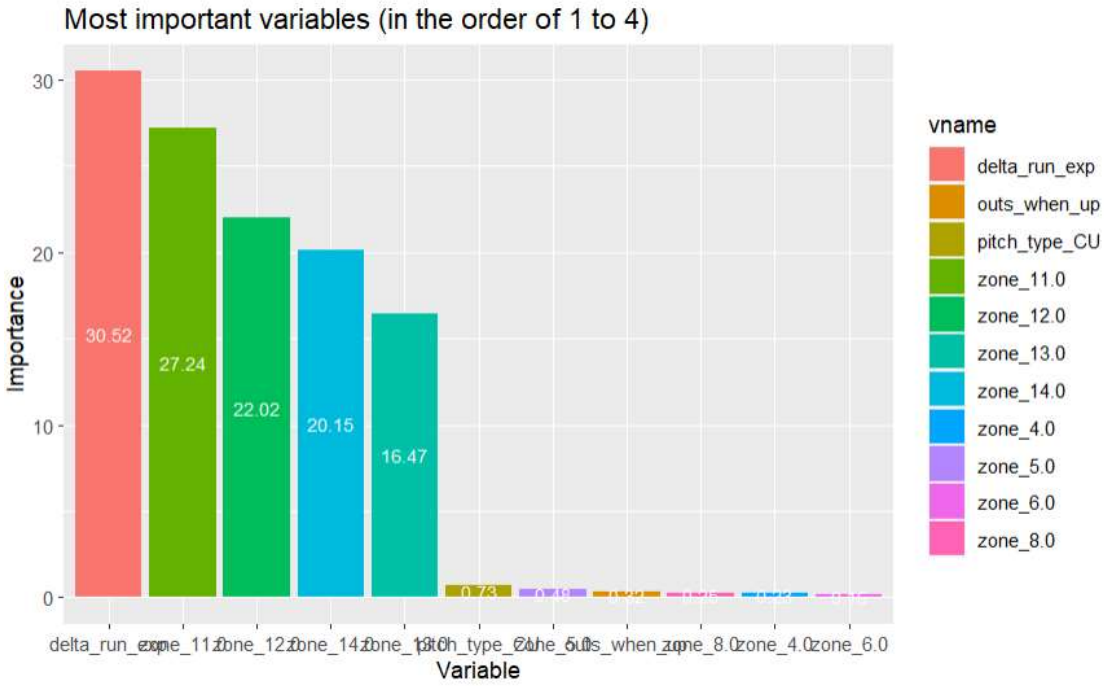
從以上幾個模型迴歸係數、決策路徑或重要特徵值可知，綜觀各個模型最重要的判斷依據，肯定是判決前後得分期望值差異，就筆者的觀察而言，這確實是一項很重要的特徵，因為零好三壞的情境下，若守備方能取得一記好球，必然能有效降低攻擊方得分的機率，而當進壘點位於好球帶以外，卻被寬容的好球帶判決成為好球，形同沒收了攻擊方一次四壞球上

壘的機會，在現代棒球觀念中，上壘不僅是評估一位打擊者能力的指標，同時是開啟球隊得分的第一扇門。此外，進壘點在好球帶以外的區域，與依變數之間的關係是絕對正相關，考慮判決前後得分期望值變動後，接著考慮好球帶以外進壘點也是一個非常直觀的路徑。

圖四：Decision Tree 決策路徑



圖五：Random Forest 重要特徵值



陸、結論

表六紀錄各個模型的預測準確率，圖三有各個模型的 ROC 曲線與 AUC 值，可知經過特徵篩選訓練出的模型，效能基本上都非常好，預測正確率基本上都有達到 100%，至於未經特徵篩選直接引入的 Support Vector Machine 模型在預測準確率上有達到 94%，但 ROC 與 AUC 的數據異常糟糕，原因是此模型將所有測資都預測為 0，但測資實際上仍有 54 個真值為 1，所以真值為 1 的樣本則完全沒有正確預測。

關於重要特徵欄位，在經過特徵篩選的各個訓練模型中，判決前後得分期望值差異是最重要的，其次則是好球帶以外的進壘點。

總的來說，筆者認為本次專題可說相當有收穫，不僅發現經過特徵篩選，能有效提升模型預測性能，甚至使得預測正確率達到 100%，AUC 數據亦令我非常驚訝，另一方面，在重要特徵欄位的部分，與直觀推論、實際觀察都十分吻合，因而產生一個具體且有趣的故事。

柒、自我檢討

對於是否採用判決前後得分期望值差異為特徵欄位，筆者是相當有疑慮的，一來判決前後得分期望值差異的計算過程不得而知，二來判決前後得分期望值差異實際上與裁判判決的決策路徑不相關，是故判決前後得分期望值差異作為特徵欄位，不見得能解釋為「預測」裁判在零好三壞且無人在壘時的好球帶判決。

事實上，筆者有另外執行不採用判決前後得分期望值差異的預測模型，結果發現各個模型的預測力大幅下降，換言之，若單以進壘點以及其他特徵來預測時，模型預測力會減損，筆者認為未來還有待開發其餘特徵，尋找判決前後得分期望值的替代方案，除此之外，僅考慮本文的預測結果而言，本次專題仍是相當有收穫。