Prediction-mushroom

s76134124 何佩勳

2025-06-17

目錄

一、資料簡述

1. 資料來源

Mushroom species drawn from source book:Patrick Hardin.Mushrooms & Toadstools.Zondervan, 1999

2. Coding book

以下表格為所有變數的定義整理 (coding book) :

Variable Name	Data Type	Definition	Note
family	nominal	Mushroom family name	Multinomial family category
name	nominal	Scientific name of mushroom	Multinomial species category

Variable Name	Data Type	Definition	Note
class	nominal	Edibility classification	p = poisonous, e = edible
cap-diameter	continuous	Cap diameter size in cm	Float value or range min-max or mean
cap-shape	nominal	Shape of the mushroom cap	b = bell, c = conical, x = convex, f = flat, s = sunken, p = spherical, o = others
cap-surface	nominal	Surface texture of the cap	i = fibrous, g = grooves, y = scaly, s = smooth, h = shiny, l = leathery, k = silky, t = sticky, w = wrinkled, e = fleshy
cap-color	nominal	Color of the mushroom cap	n = brown, b = buff, g = gray, r = green, p = pink, u = purple, e = red, w = white, y = yellow, l = blue, o = orange, k = black
does-bruise-bleed	nominal	Does it bruise or bleed	t = bruises or bleeding, f = no bruising or bleeding
gill-attachment	nominal	Attachment of gills to stem	a = adnate, x = adnexed, d = decurrent, e = free, s = sinuate, p = pores, f = none, ? = unknown
gill-spacing	nominal	Spacing between gills	c = close, d = distant, f = none
gill-color	nominal	Color of the gills	Same as cap-color plus f = none
stem-height	continuous	Stem height in cm	Float value or range min-max or mean
stem-width	continuous	Stem width in mm	Float value or range min-max or mean
stem-root	nominal	Type of stem root	b = bulbous, s = swollen, c = club, u = cup, e = equal, z = rhizomorphs, r = rooted
stem-surface	nominal	Surface texture of the stem	Same as cap-surface plus f = none
stem-color	nominal	Color of the stem	Same as cap-color plus f = none
veil-type	nominal	Type of veil covering	p = partial, u = universal
veil-color	nominal	Color of the veil	Same as cap-color plus f = none
has-ring	nominal	Presence of a ring	t = ring present, f = none
ring-type	nominal	Type of ring	c = cobwebby, e = evanescent, r = flaring, g = grooved, l = large, p = pendant, s = sheathing, z = zone, y = scaly, m = movable, f = none, ? = unknown
spore-print-color	nominal	Color of spore print	Same as cap-color

Variable Name	Data Type	Definition	Note
habitat	nominal	Where it is found	g = grasses, l = leaves, m = meadows, p = paths, h = heaths, u = urban, w = waste, d = woods
season	nominal	Season when it grows	s = spring, u = summer, a = autumn, w = winter

二、目標及分析流程

1. 目標

建立模型預測蘑菇是否有毒

2. 分析流程

1. 描述性統計
2. 資料前處理 + 缺失值處理
3. 以羅吉斯回歸預測
4. 以 SVM, 隨機森林,XG boost 預測
5. 效能評估:K-fold Cross Validation(5-fold)，指標使用：Accuracy、F1-score(預設 threshold:0.5)、AUC、混淆矩陣

三、描述性統計

```
library(reticulate)
library(Hmisc)
library(dplyr)
data <- read.csv("primary_data.csv", sep=";", stringsAsFactors=FALSE)
data <- data %>% mutate(across(everything(), trimws))
latex(describe(data), file="")
```

data
23 Variables 173 Observations

family

n	missing	distinct
173	0	23

```
lowest : Amanita Family     Bolbitius Family   Bolete Family      Bracket Fungi      Chanterelle Family
highest: Russula Family     Saddle-Cup Family  Stropharia Family  Tricholoma Family  Wax Gill Family
```

name

n	missing	distinct
173	0	173

```
lowest : Amethyst Deceiver       Aniseed Funnel Cap      Apricot Fungus          Bare-toothed Russula    Bay Bolete
highest: Yellow-gilled Russula   Yellow-staining Mushroom Yellow-stemmed Bell Cap Yellow Swamp Russula    Yellow Wax cap
```

class

n	missing	distinct
173	0	2

Value	e	p
Frequency	77	96
Proportion	0.445	0.555

---

cap.diameter

n	missing	distinct
173	0	51

```
lowest : [0.4, 1]   [0.5, 1.5] [0.5, 1]   [0.7, 1.3] [1, 1.5]
highest: [8, 14]    [8, 15]    [8, 20]    [8, 25]    [8, 30]
```

---

cap.shape

n	missing	distinct
173	0	27

```
lowest : [b, f, s] [b, f]    [b, x, f] [b, x]    [b]
highest: [x, f]    [x, o]    [x, p]    [x, s]    [x]
```

Cap.surface

n	missing	distinct
133	40	40

```
lowest : [d, e, y, i] [d, k, s]    [d, k]       [d, s]       [d]
highest: [t]          [w, t]       [w]          [y, s]       [y]
```

---

cap.color

n	missing	distinct
173	0	67

```
lowest : [b, p, e, y]    [b, u]          [b]             [e, n, p, w]    [e, n, y]
highest: [y, n]          [y, o, g, n, r] [y, o, r, n]    [y, o]          [y]
```

---

does.bruise.or.bleed

n	missing	distinct
173	0	2

Value	[f]	[t]
Frequency	143	30
Proportion	0.827	0.173

---

gill.attachment

n	missing	distinct
145	28	8

Value	[a, d]	[a]	[d]	[e]	[f]	[p]	[s]	[x]
Frequency	8	32	25	16	10	17	16	21
Proportion	0.055	0.221	0.172	0.110	0.069	0.117	0.110	0.145

---

gill.spacing

n	missing	distinct
102	71	3

Value	[c]	[d]	[f]
Frequency	70	22	10
Proportion	0.686	0.216	0.098

---

gill.color

n	missing	distinct
173	0	59

```
lowest : [b, p, w] [b, u]    [b]       [e]       [f]
highest: [y, o, e] [y, r, k] [y, r]    [y, w]    [y]
```

---

stem.height

n	missing	distinct
173	0	46

```
lowest : [0]      [1, 2]   [1, 3]   [10, 12] [10, 15], highest: [8, 12]  [8, 15]  [8, 20]  [8, 25]  [8, 30]
```

---

stem.width

n	missing	distinct
173	0	48

lowest : [0.5, 1] [0] [1, 2] [1, 3] [1] , highest: [7, 15] [8, 12] [8, 15] [8, 18] [8, 20]

---

stem.root

n	missing	distinct
27	146	5

Value	[b]	[c]	[f]	[r]	[s]
Frequency	9	2	3	4	9
Proportion	0.333	0.074	0.111	0.148	0.333

---

stem.surface

n	missing	distinct
65	108	14

Value	[f]	[g]	[h]	[i, s]	[i, t]	[i, y]	[i]	[k, s]	[k]	[s, h]	[s]	[t]
Frequency	3	5	1	1	1	1	11	1	4	1	15	7
Proportion	0.046	0.077	0.015	0.015	0.015	0.015	0.169	0.015	0.062	0.015	0.231	0.108

Value	[y, s]	[y]
Frequency	1	13
Proportion	0.015	0.200

---

stem.color

n	missing	distinct
173	0	41

lowest : [b, u] [e, n] [e, u, y] [e, y] [e]
highest: [w] [y, e, n] [y, n] [y, o, k] [y]

---

veil.type

n	missing	distinct	value
9	164	1	[u]

Value	[u]
Frequency	9
Proportion	1

---

veil.color

n	missing	distinct
21	152	7

Value	[e, n]	[k]	[n]	[u]	[w]	[y, w]	[y]
Frequency	1	1	1	1	15	1	1
Proportion	0.048	0.048	0.048	0.048	0.714	0.048	0.048

---

has.ring

n	missing	distinct
173	0	2

Value	[f]	[t]
Frequency	130	43
Proportion	0.751	0.249

---

ring.type

n	missing	distinct
166	7	13

Value	[e, g]	[e]	[f]	[g, p]	[g]	[l, e]	[l, p]	[l, r]	[l]	[m]	[p]	[r]
Frequency	1	6	137	2	2	1	1	2	2	1	2	3
Proportion	0.006	0.036	0.825	0.012	0.012	0.006	0.006	0.012	0.012	0.006	0.012	0.018

Value	[z]
Frequency	6
Proportion	0.036

Spore.print.color

n	missing	distinct
18	155	8

Value	[g]	[k, r]	[k, u]	[k]	[n]	[p, w]	[p]	[w]
Frequency	1	1	1	5	3	1	3	3
Proportion	0.056	0.056	0.056	0.278	0.167	0.056	0.167	0.167

---

habitat

n	missing	distinct
173	0	21

```
lowest : [d, h]    [d]       [g, d, h] [g, d]     [g, h, d]
highest: [m, d]    [m, h]    [m]       [p, d]     [w]
```

---

season

n	missing	distinct
173	0	10

```
Value          [a, w]         [a]   [s, a, w] [s, u, a, w]    [s, u, a]       [s, u]
Frequency          15          16           1           13            5            3
Proportion      0.087       0.092       0.006        0.075        0.029        0.017

Value             [s]   [u, a, w]      [u, a]          [u]
Frequency           1          12         106            1
Proportion      0.006       0.069       0.613        0.006
```

四、資料前處理 + 缺失值處理

1. 缺失值處理

以下為有 missing data 的變項:

```r
colSums(data == "" | is.na(data))
```

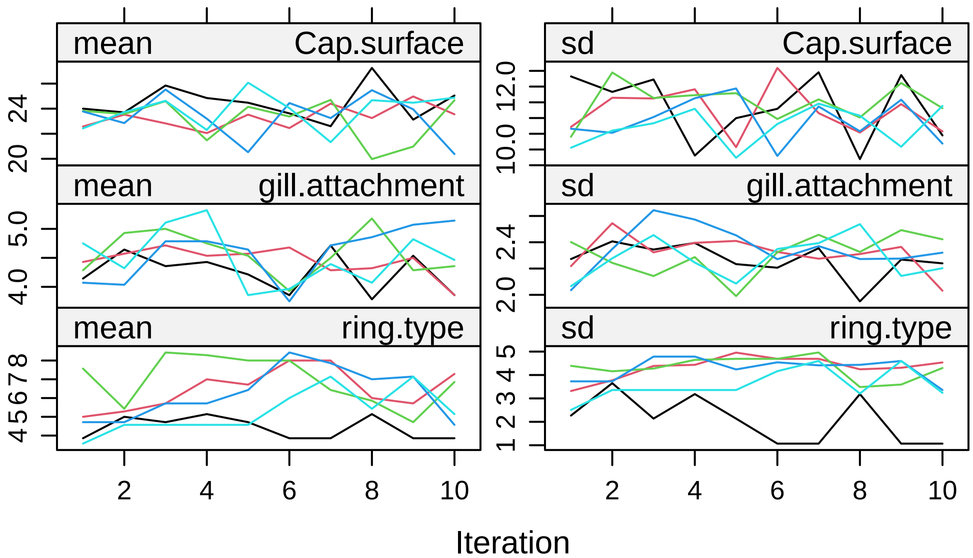
```
           family                name               class
                0                   0                   0
     cap.diameter           cap.shape         Cap.surface
                0                   0                  40
        cap.color does.bruise.or.bleed     gill.attachment
                0                   0                  28
     gill.spacing          gill.color         stem.height
               71                   0                   0
       stem.width           stem.root        stem.surface
                0                 146                 108
       stem.color           veil.type          veil.color
                0                 164                 152
         has.ring           ring.type   Spore.print.color
                0                   7                 155
          habitat              season
                0                   0
```

- 缺失值超過 4 成的變項推測為不重要的變項，故直接刪除不予進行分析，總共刪除了 stem.root、stem.surface、veil.type、veil.color、Spore.print.color。
- 剩下缺失變項使用 MICE 進行差補，以多重差補 5 次後檢視圖形沒有明顯發散，最終使用第一組資料進行差補。

```r
library(mice)
# 刪除法
data_clean <- data %>% select(where(~ mean(. != "" & !is.na(.)) > 0.6))
colSums(data_clean == "" | is.na(data_clean))

# 差補法
vars_for_impute <- c("Cap.surface", "gill.attachment", "ring.type")
data_mice <- data_clean

data_mice <- data_mice %>%
```

```r
  mutate(across(all_of(vars_for_impute), ~ factor(ifelse(. == "", NA, .))))

## 設定差補方法：categorical 變數使用 "polyreg"（多項 logistic）
method_vec <- make.method(data_mice)
method_vec[vars_for_impute] <- "polyreg"
method_vec[setdiff(names(method_vec), vars_for_impute)] <- ""  # 其他變數不差補

## 執行多重差補（設定 m = 5 重複 5 次）
imp <- mice(data_mice, method = method_vec, m = 5, maxit = 10, seed = 123)
```

```r
## 繪圖看差補結果
plot(imp)
```




```r
## 使用第一組差補結果匯出
data_imputed <- complete(imp, 1)

## 確認全部已差補完成
colSums(is.na(data_imputed))
```

```
             family                name               class 
                  0                   0                   0 
       cap.diameter           cap.shape         Cap.surface 
                  0                   0                   0 
          cap.color does.bruise.or.bleed     gill.attachment 
                  0                   0                   0 
         gill.color         stem.height          stem.width 
                  0                   0                   0 
         stem.color            has.ring           ring.type 
                  0                   0                   0 
            habitat              season 
                  0                   0 
```

2. 資料前處理: 更改 coding 方式

(1) 類別變項整理:One hot encoding

```
# 定義 one-hot encoding 函數（處理多重類別值）
process_column_to_matrix <- function(data, column_name) {
  if (!(column_name %in% names(data))) {
    stop(paste(" 欄位", column_name, " 不存在於資料中！"))
  }

  # 處理格式：移除中括號、空白、NA
  col_data <- gsub("\\[|\\]", "", data[[column_name]])  # 去除 []
  col_data <- gsub(" ", "", col_data)                    # 去除空白
  col_data[is.na(col_data)] <- ""                        # NA → 空字串

  # 找出所有類別
  levels <- unique(unlist(strsplit(col_data, ",")))
  levels <- levels[levels != ""]  # 移除空白層級

  # 建立 one-hot 矩陣
  ta <- matrix(0, nrow = nrow(data), ncol = length(levels))
  colnames(ta) <- paste(column_name, levels, sep = "_")

  # 為每個 level 建立 dummy 欄
  for (i in seq_along(levels)) {
    ta[grepl(paste0("\\b", levels[i], "\\b"), col_data), i] <- 1
  }

  return(as.data.frame(ta))
}
# 確保欄位名稱為小寫
names(data_imputed) <- tolower(names(data_imputed))

# 要處理的類別欄位
columns_to_process <- c("cap.shape", "cap.surface", "cap.color",
  "does.bruise.or.bleed", "gill.attachment", "gill.color",
  "stem.color", "has.ring", "ring.type", "habitat", "season"
)

# 套用函數並建立 one-hot 結果
processed_tables <- lapply(columns_to_process, function(col) {
  process_column_to_matrix(data_imputed, col)
})

# 合併所有欄位為最終 one-hot 資料
final_table <- do.call(cbind, processed_tables)
```

(2) 連續變項整理: 有兩個數值的取平均數並取代:

```
process_numbers <- function(x) {
  x <- gsub("\\[|\\]", "", x)
  numbers <- as.numeric(unlist(strsplit(x, ",")))
  if (length(numbers) == 2) {
    return(mean(numbers))
  } else {
    return(numbers)
```

```
  }
}

data_imputed <- data_imputed %>%
  mutate(across(c(4, 11, 12), ~sapply(., process_numbers)))
```

(3) 將類別變項和連續變項合併成最終資料

```
numeric_data <- data_imputed[, c(4, 11, 12)]

# 確認這三欄都是 numeric（若不是就轉換）
numeric_data <- numeric_data %>%
  mutate(across(everything(), as.numeric))

# 合併成最終建模資料集
final_data <- cbind(data_imputed[,1:3],numeric_data,final_table)
```

(4) 以是否有毒製作 Table 1:

```
library(table1)

data_filter <- final_data[,-c(1,2)]
data_filter <- data_filter %>%
  mutate(across(c(1, 5:90), as.factor))
data_filter <- data_filter %>%
  mutate(across(2:4, as.numeric))
table1(~.|class,data = data_filter)
```

	e	p	Overall
	(N=77)	(N=96)	(N=173)
cap.diameter			
Mean (SD)	7.81 (6.26)	5.88 (3.85)	6.74 (5.14)
Median [Min, Max]	6.50 [1.00, 50.0]	5.00 [0.700, 19.0]	6.00 [0.700, 50.0]
stem.height			
Mean (SD)	7.05 (3.48)	6.22 (3.05)	6.59 (3.26)
Median [Min, Max]	6.00 [2.50, 25.0]	5.50 [0, 17.5]	6.00 [0, 25.0]
stem.width			
Mean (SD)	14.4 (10.8)	10.4 (8.66)	12.2 (9.86)
Median [Min, Max]	12.5 [1.00, 70.0]	7.50 [0, 40.0]	10.0 [0, 70.0]
cap.shape_x			
0	23 (29.9%)	40 (41.7%)	63 (36.4%)
1	54 (70.1%)	56 (58.3%)	110 (63.6%)
cap.shape_f			
0	41 (53.2%)	58 (60.4%)	99 (57.2%)
1	36 (46.8%)	38 (39.6%)	74 (42.8%)
cap.shape_p			
0	67 (87.0%)	91 (94.8%)	158 (91.3%)
1	10 (13.0%)	5 (5.2%)	15 (8.7%)
cap.shape_b			
0	72 (93.5%)	78 (81.3%)	150 (86.7%)
1	5 (6.5%)	18 (18.8%)	23 (13.3%)
cap.shape_c			
0	73 (94.8%)	92 (95.8%)	165 (95.4%)

	e	p	Overall
1	4 (5.2%)	4 (4.2%)	8 (4.6%)
cap.shape_s			
0	60 (77.9%)	77 (80.2%)	137 (79.2%)
1	17 (22.1%)	19 (19.8%)	36 (20.8%)
cap.shape_o			
0	73 (94.8%)	88 (91.7%)	161 (93.1%)
1	4 (5.2%)	8 (8.3%)	12 (6.9%)
cap.surface_g			
0	67 (87.0%)	84 (87.5%)	151 (87.3%)
1	10 (13.0%)	12 (12.5%)	22 (12.7%)
cap.surface_h			
0	60 (77.9%)	78 (81.3%)	138 (79.8%)
1	17 (22.1%)	18 (18.8%)	35 (20.2%)
cap.surface_y			
0	63 (81.8%)	84 (87.5%)	147 (85.0%)
1	14 (18.2%)	12 (12.5%)	26 (15.0%)
cap.surface_t			
0	53 (68.8%)	69 (71.9%)	122 (70.5%)
1	24 (31.2%)	27 (28.1%)	51 (29.5%)
cap.surface_e			
0	72 (93.5%)	88 (91.7%)	160 (92.5%)
1	5 (6.5%)	8 (8.3%)	13 (7.5%)
cap.surface_d			
0	67 (87.0%)	84 (87.5%)	151 (87.3%)
1	10 (13.0%)	12 (12.5%)	22 (12.7%)
cap.surface_k			
0	75 (97.4%)	84 (87.5%)	159 (91.9%)
1	2 (2.6%)	12 (12.5%)	14 (8.1%)
cap.surface_s			
0	55 (71.4%)	74 (77.1%)	129 (74.6%)
1	22 (28.6%)	22 (22.9%)	44 (25.4%)
cap.surface_l			
0	74 (96.1%)	93 (96.9%)	167 (96.5%)
1	3 (3.9%)	3 (3.1%)	6 (3.5%)
cap.surface_w			
0	71 (92.2%)	88 (91.7%)	159 (91.9%)
1	6 (7.8%)	8 (8.3%)	14 (8.1%)
cap.surface_i			
0	75 (97.4%)	89 (92.7%)	164 (94.8%)
1	2 (2.6%)	7 (7.3%)	9 (5.2%)
cap.color_e			
0	70 (90.9%)	78 (81.3%)	148 (85.5%)
1	7 (9.1%)	18 (18.8%)	25 (14.5%)
cap.color_o			
0	70 (90.9%)	81 (84.4%)	151 (87.3%)
1	7 (9.1%)	15 (15.6%)	22 (12.7%)
cap.color_n			
0	24 (31.2%)	39 (40.6%)	63 (36.4%)
1	53 (68.8%)	57 (59.4%)	110 (63.6%)
cap.color_g			
0	63 (81.8%)	82 (85.4%)	145 (83.8%)
1	14 (18.2%)	14 (14.6%)	28 (16.2%)

	e	p	Overall
cap.color__r			
0	75 (97.4%)	85 (88.5%)	160 (92.5%)
1	2 (2.6%)	11 (11.5%)	13 (7.5%)
cap.color__w			
0	60 (77.9%)	78 (81.3%)	138 (79.8%)
1	17 (22.1%)	18 (18.8%)	35 (20.2%)
cap.color__y			
0	61 (79.2%)	68 (70.8%)	129 (74.6%)
1	16 (20.8%)	28 (29.2%)	44 (25.4%)
cap.color__p			
0	73 (94.8%)	89 (92.7%)	162 (93.6%)
1	4 (5.2%)	7 (7.3%)	11 (6.4%)
cap.color__b			
0	72 (93.5%)	94 (97.9%)	166 (96.0%)
1	5 (6.5%)	2 (2.1%)	7 (4.0%)
cap.color__u			
0	72 (93.5%)	91 (94.8%)	163 (94.2%)
1	5 (6.5%)	5 (5.2%)	10 (5.8%)
cap.color__l			
0	73 (94.8%)	94 (97.9%)	167 (96.5%)
1	4 (5.2%)	2 (2.1%)	6 (3.5%)
cap.color__k			
0	74 (96.1%)	90 (93.8%)	164 (94.8%)
1	3 (3.9%)	6 (6.3%)	9 (5.2%)
does.bruise.or.bleed__f			
0	14 (18.2%)	16 (16.7%)	30 (17.3%)
1	63 (81.8%)	80 (83.3%)	143 (82.7%)
does.bruise.or.bleed__t			
0	63 (81.8%)	80 (83.3%)	143 (82.7%)
1	14 (18.2%)	16 (16.7%)	30 (17.3%)
gill.attachment__e			
0	65 (84.4%)	89 (92.7%)	154 (89.0%)
1	12 (15.6%)	7 (7.3%)	19 (11.0%)
gill.attachment__p			
0	64 (83.1%)	90 (93.8%)	154 (89.0%)
1	13 (16.9%)	6 (6.3%)	19 (11.0%)
gill.attachment__a			
0	58 (75.3%)	64 (66.7%)	122 (70.5%)
1	19 (24.7%)	32 (33.3%)	51 (29.5%)
gill.attachment__d			
0	60 (77.9%)	73 (76.0%)	133 (76.9%)
1	17 (22.1%)	23 (24.0%)	40 (23.1%)
gill.attachment__s			
0	70 (90.9%)	83 (86.5%)	153 (88.4%)
1	7 (9.1%)	13 (13.5%)	20 (11.6%)
gill.attachment__x			
0	67 (87.0%)	83 (86.5%)	150 (86.7%)
1	10 (13.0%)	13 (13.5%)	23 (13.3%)
gill.attachment__f			
0	73 (94.8%)	89 (92.7%)	162 (93.6%)
1	4 (5.2%)	7 (7.3%)	11 (6.4%)
gill.color__w			

	e	p	Overall
0	39 (50.6%)	61 (63.5%)	100 (57.8%)
1	38 (49.4%)	35 (36.5%)	73 (42.2%)
gill.color_n			
0	62 (80.5%)	64 (66.7%)	126 (72.8%)
1	15 (19.5%)	32 (33.3%)	47 (27.2%)
gill.color_p			
0	65 (84.4%)	80 (83.3%)	145 (83.8%)
1	12 (15.6%)	16 (16.7%)	28 (16.2%)
gill.color_u			
0	74 (96.1%)	92 (95.8%)	166 (96.0%)
1	3 (3.9%)	4 (4.2%)	7 (4.0%)
gill.color_b			
0	74 (96.1%)	94 (97.9%)	168 (97.1%)
1	3 (3.9%)	2 (2.1%)	5 (2.9%)
gill.color_g			
0	67 (87.0%)	83 (86.5%)	150 (86.7%)
1	10 (13.0%)	13 (13.5%)	23 (13.3%)
gill.color_y			
0	60 (77.9%)	69 (71.9%)	129 (74.6%)
1	17 (22.1%)	27 (28.1%)	44 (25.4%)
gill.color_r			
0	75 (97.4%)	90 (93.8%)	165 (95.4%)
1	2 (2.6%)	6 (6.3%)	8 (4.6%)
gill.color_e			
0	75 (97.4%)	92 (95.8%)	167 (96.5%)
1	2 (2.6%)	4 (4.2%)	6 (3.5%)
gill.color_o			
0	72 (93.5%)	88 (91.7%)	160 (92.5%)
1	5 (6.5%)	8 (8.3%)	13 (7.5%)
gill.color_k			
0	71 (92.2%)	87 (90.6%)	158 (91.3%)
1	6 (7.8%)	9 (9.4%)	15 (8.7%)
gill.color_f			
0	73 (94.8%)	90 (93.8%)	163 (94.2%)
1	4 (5.2%)	6 (6.3%)	10 (5.8%)
stem.color_w			
0	34 (44.2%)	65 (67.7%)	99 (57.2%)
1	43 (55.8%)	31 (32.3%)	74 (42.8%)
stem.color_y			
0	68 (88.3%)	73 (76.0%)	141 (81.5%)
1	9 (11.7%)	23 (24.0%)	32 (18.5%)
stem.color_n			
0	50 (64.9%)	53 (55.2%)	103 (59.5%)
1	27 (35.1%)	43 (44.8%)	70 (40.5%)
stem.color_b			
0	76 (98.7%)	96 (100%)	172 (99.4%)
1	1 (1.3%)	0 (0%)	1 (0.6%)
stem.color_u			
0	75 (97.4%)	91 (94.8%)	166 (96.0%)
1	2 (2.6%)	5 (5.2%)	7 (4.0%)
stem.color_l			
0	76 (98.7%)	95 (99.0%)	171 (98.8%)

	e	p	Overall
1	1 (1.3%)	1 (1.0%)	2 (1.2%)
stem.color__r			
0	76 (98.7%)	93 (96.9%)	169 (97.7%)
1	1 (1.3%)	3 (3.1%)	4 (2.3%)
stem.color__p			
0	76 (98.7%)	93 (96.9%)	169 (97.7%)
1	1 (1.3%)	3 (3.1%)	4 (2.3%)
stem.color__e			
0	74 (96.1%)	88 (91.7%)	162 (93.6%)
1	3 (3.9%)	8 (8.3%)	11 (6.4%)
stem.color__k			
0	76 (98.7%)	93 (96.9%)	169 (97.7%)
1	1 (1.3%)	3 (3.1%)	4 (2.3%)
stem.color__g			
0	70 (90.9%)	89 (92.7%)	159 (91.9%)
1	7 (9.1%)	7 (7.3%)	14 (8.1%)
stem.color__o			
0	72 (93.5%)	89 (92.7%)	161 (93.1%)
1	5 (6.5%)	7 (7.3%)	12 (6.9%)
stem.color__f			
0	77 (100%)	93 (96.9%)	170 (98.3%)
1	0 (0%)	3 (3.1%)	3 (1.7%)
has.ring__t			
0	60 (77.9%)	70 (72.9%)	130 (75.1%)
1	17 (22.1%)	26 (27.1%)	43 (24.9%)
has.ring__f			
0	17 (22.1%)	26 (27.1%)	43 (24.9%)
1	60 (77.9%)	70 (72.9%)	130 (75.1%)
ring.type__g			
0	73 (94.8%)	92 (95.8%)	165 (95.4%)
1	4 (5.2%)	4 (4.2%)	8 (4.6%)
ring.type__p			
0	75 (97.4%)	93 (96.9%)	168 (97.1%)
1	2 (2.6%)	3 (3.1%)	5 (2.9%)
ring.type__e			
0	74 (96.1%)	91 (94.8%)	165 (95.4%)
1	3 (3.9%)	5 (5.2%)	8 (4.6%)
ring.type__l			
0	73 (94.8%)	94 (97.9%)	167 (96.5%)
1	4 (5.2%)	2 (2.1%)	6 (3.5%)
ring.type__f			
0	14 (18.2%)	18 (18.8%)	32 (18.5%)
1	63 (81.8%)	78 (81.3%)	141 (81.5%)
ring.type__m			
0	76 (98.7%)	96 (100%)	172 (99.4%)
1	1 (1.3%)	0 (0%)	1 (0.6%)
ring.type__r			
0	74 (96.1%)	94 (97.9%)	168 (97.1%)
1	3 (3.9%)	2 (2.1%)	5 (2.9%)
ring.type__z			
0	77 (100%)	90 (93.8%)	167 (96.5%)
1	0 (0%)	6 (6.3%)	6 (3.5%)

	e	p	Overall
habitat_d			
0	8 (10.4%)	14 (14.6%)	22 (12.7%)
1	69 (89.6%)	82 (85.4%)	151 (87.3%)
habitat_m			
0	69 (89.6%)	87 (90.6%)	156 (90.2%)
1	8 (10.4%)	9 (9.4%)	17 (9.8%)
habitat_g			
0	62 (80.5%)	73 (76.0%)	135 (78.0%)
1	15 (19.5%)	23 (24.0%)	38 (22.0%)
habitat_h			
0	72 (93.5%)	88 (91.7%)	160 (92.5%)
1	5 (6.5%)	8 (8.3%)	13 (7.5%)
habitat_l			
0	66 (85.7%)	89 (92.7%)	155 (89.6%)
1	11 (14.3%)	7 (7.3%)	18 (10.4%)
habitat_p			
0	77 (100%)	94 (97.9%)	171 (98.8%)
1	0 (0%)	2 (2.1%)	2 (1.2%)
habitat_w			
0	76 (98.7%)	96 (100%)	172 (99.4%)
1	1 (1.3%)	0 (0%)	1 (0.6%)
habitat_u			
0	76 (98.7%)	96 (100%)	172 (99.4%)
1	1 (1.3%)	0 (0%)	1 (0.6%)
season_u			
0	16 (20.8%)	17 (17.7%)	33 (19.1%)
1	61 (79.2%)	79 (82.3%)	140 (80.9%)
season_a			
0	3 (3.9%)	2 (2.1%)	5 (2.9%)
1	74 (96.1%)	94 (97.9%)	168 (97.1%)
season_w			
0	52 (67.5%)	80 (83.3%)	132 (76.3%)
1	25 (32.5%)	16 (16.7%)	41 (23.7%)
season_s			
0	65 (84.4%)	85 (88.5%)	150 (86.7%)
1	12 (15.6%)	11 (11.5%)	23 (13.3%)

有毒組與無毒組間比例為 45%:55%**，無太嚴重資料不平衡，但有發現有一些變項在有毒組或無毒組幾乎都是** 0 **或** 1**，表示不具有區分有毒無毒的能力，故將兩組皆是** 0(**或** 1) **占比高達** 85% **的這些變項刪除不予列入模型。**

```
# 將 class 改無毒為 0 有毒為 1
data_filter$class <- factor(data_filter$class, levels = c("e", "p"))
data_filter$class<- ifelse(data_filter$class == "e", 0, 1)

# 製作函數
# 建立一個空的變數儲存要刪除的欄位
vars_to_drop <- c()

# 針對每一個變數（從第 5 到 90 欄為例）
for (var in names(data_filter)[5:90]) {
  vec <- data_filter[[var]]
```

```r
  # 確保是 0/1 的 factor 或 numeric 欄位
  if ((is.factor(vec) && all(levels(vec) %in% c("0", "1"))) ||
      (is.numeric(vec) && all(unique(na.omit(vec)) %in% c(0, 1)))) {

    # 拆成兩組
    vec_0 <- vec[data_filter$class == 0]
    vec_1 <- vec[data_filter$class == 1]

    # 計算每組中 1 和 0 的比例
    p1_0 <- mean(as.numeric(as.character(vec_0)) == 1, na.rm = TRUE)
    p1_1 <- mean(as.numeric(as.character(vec_1)) == 1, na.rm = TRUE)
    p0_0 <- mean(as.numeric(as.character(vec_0)) == 0, na.rm = TRUE)
    p0_1 <- mean(as.numeric(as.character(vec_1)) == 0, na.rm = TRUE)

    # 若兩組都 1 的比例  0.85 或兩組都 0 的比例  0.85 → 刪除
    if ((p1_0 >= 0.85 && p1_1 >= 0.85) || (p0_0 >= 0.85 && p0_1 >= 0.85)) {
      vars_to_drop <- c(vars_to_drop, var)
    }
  }
}

# 輸出要刪除的變數
print(vars_to_drop)
```

```
 [1] "cap.shape_p"       "cap.shape_c"       "cap.shape_o"
 [4] "cap.surface_g"     "cap.surface_e"     "cap.surface_d"
 [7] "cap.surface_k"     "cap.surface_l"     "cap.surface_w"
[10] "cap.surface_i"     "cap.color_r"       "cap.color_p"
[13] "cap.color_b"       "cap.color_u"       "cap.color_l"
[16] "cap.color_k"       "gill.attachment_s" "gill.attachment_x"
[19] "gill.attachment_f" "gill.color_u"      "gill.color_b"
[22] "gill.color_g"      "gill.color_r"      "gill.color_e"
[25] "gill.color_o"      "gill.color_k"      "gill.color_f"
[28] "stem.color_b"      "stem.color_u"      "stem.color_l"
[31] "stem.color_r"      "stem.color_p"      "stem.color_e"
[34] "stem.color_k"      "stem.color_g"      "stem.color_o"
[37] "stem.color_f"      "ring.type_g"       "ring.type_p"
[40] "ring.type_e"       "ring.type_l"       "ring.type_m"
[43] "ring.type_r"       "ring.type_z"       "habitat_d"
[46] "habitat_m"         "habitat_h"         "habitat_l"
[49] "habitat_p"         "habitat_w"         "habitat_u"
[52] "season_a"
```

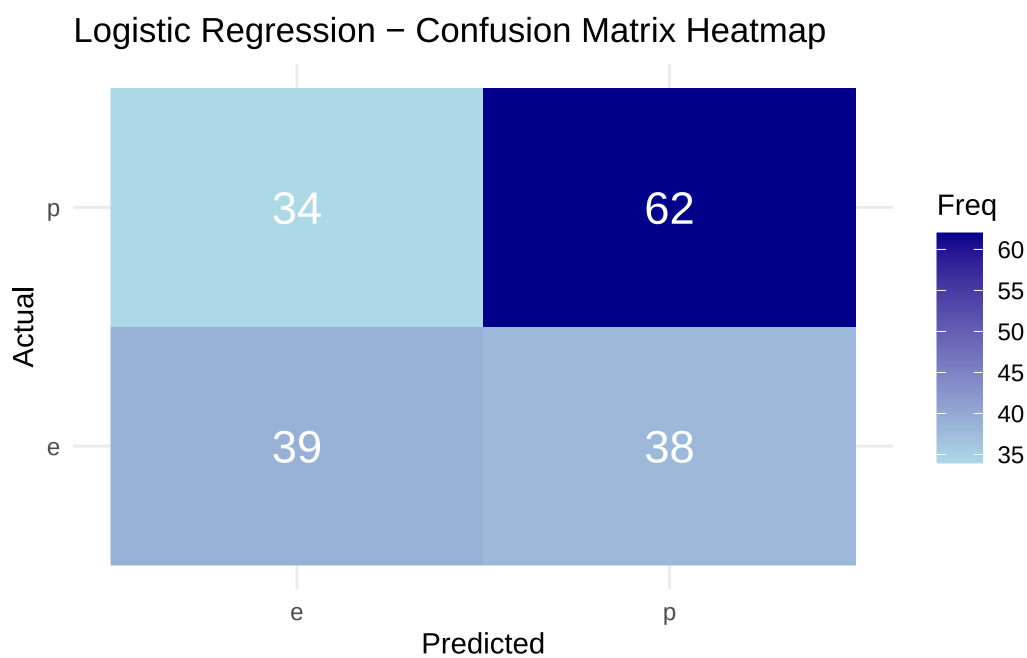
```r
# 建立新的資料框（刪除變數後）
del_data_filter <- data_filter[, !names(data_filter) %in% vars_to_drop]
```

五、以羅吉斯回歸預測

```r
library(caret)
library(ggplot2)
library(dplyr)
library(yardstick)
```

```r
# 通用模型評估函數
evaluate_model <- function(model_object, model_name = "Model") {
  pred <- model_object$pred %>%
    mutate(obs = factor(obs, levels = c("e", "p")),
           pred = factor(pred, levels = c("e", "p")))

  # 混淆矩陣熱力圖
  conf_tbl <- as.data.frame(table(Predicted = pred$pred, Actual = pred$obs))
  print(
    ggplot(conf_tbl, aes(x = Predicted, y = Actual, fill = Freq)) +
      geom_tile() +
      geom_text(aes(label = Freq), color = "white", size = 6) +
      scale_fill_gradient(low = "lightblue", high = "darkblue") +
      labs(title = paste(model_name, "- Confusion Matrix Heatmap")) +
      theme_minimal()
  )

  # 評估指標
  pos_class <- levels(pred$obs)[2]  # 預設第二個為正類
  auc <- roc_auc_vec(truth = pred$obs, estimate = pred[[pos_class]], event_level = "second")
  acc <- accuracy_vec(truth = pred$obs, estimate = pred$pred)
  f1  <- f_meas_vec(truth = pred$obs, estimate = pred$pred, event_level = "second")

  tibble(
    Model = model_name,
    Metric = c("Accuracy", "F1 Score", "AUC"),
    Value = c(acc, f1, auc)
  )
}
# 設定 class 為 e/p 兩類
del_data_filter$class <- ifelse(del_data_filter$class == "0", "e", "p")
del_data_filter$class <- factor(del_data_filter$class, levels = c("e", "p"))

# 指定 factor 與 numeric 欄位
del_data_filter <- del_data_filter %>% mutate(across(c(1, 5:38), as.factor))
del_data_filter <- del_data_filter %>% mutate(across(2:4, as.numeric))

# 交叉驗證設定
ctrl <- trainControl(
  method = "cv",
  number = 5,
  classProbs = TRUE,
  summaryFunction = defaultSummary,
  savePredictions = "final"
)
```

```r
set.seed(123)
logit_cv <- train(
  class ~ ., data =del_data_filter,
  method = "glm", family = "binomial",
  trControl = ctrl, metric = "Accuracy"
)
logit_result <- evaluate_model(logit_cv, "Logistic Regression")
```

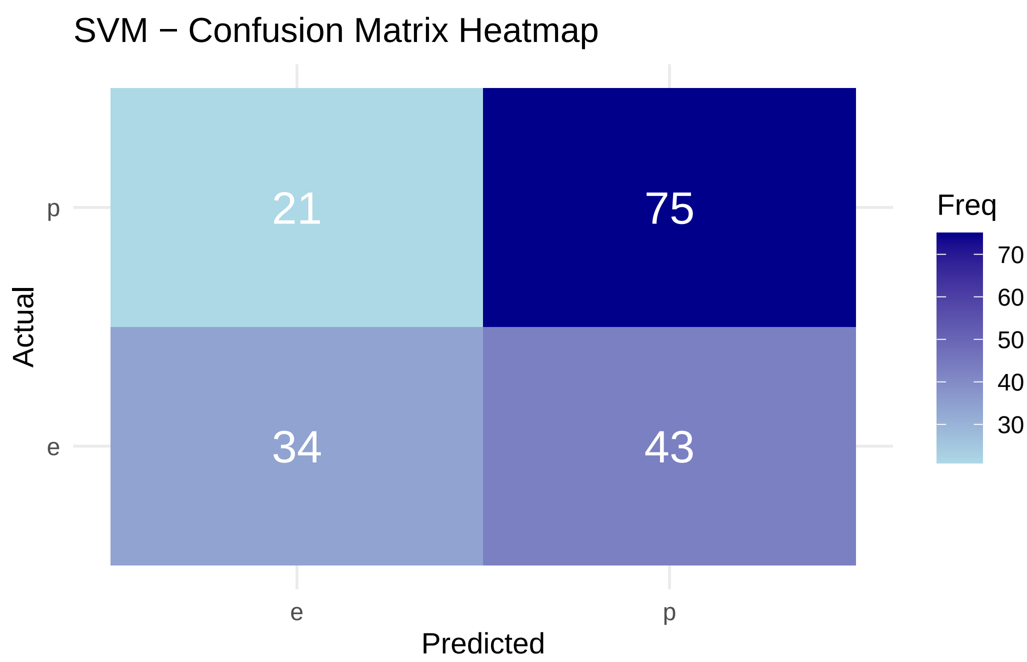


六、以 SVM, 決策樹, 隨機森林,XG boost 預測

1. SVM

參數設定:
- 使用 RBF kernel, 並對輸入資料標準化 (中心化 + 標準差為 1) , 先嘗試不同參數組合, 最終選擇:
- 超參數 C:0.01
- 超參數 sigma:0.001

```
library(kernlab)
set.seed(123)

svm_grid <- expand.grid(
  C = c(0.01, 0.1, 1, 10, 100),
  sigma = c(0.001, 0.01, 0.1, 1)
)

svm_cv <- train(
  class ~ ., data = del_data_filter,
  method = "svmRadial",
  trControl = ctrl,
  preProcess = c("center", "scale"),
  tuneGrid = svm_grid,
  metric = "Accuracy"
)

# 查看所有組合與準確率
#svm_cv$results

svm_result <- evaluate_model(svm_cv, "SVM")
```

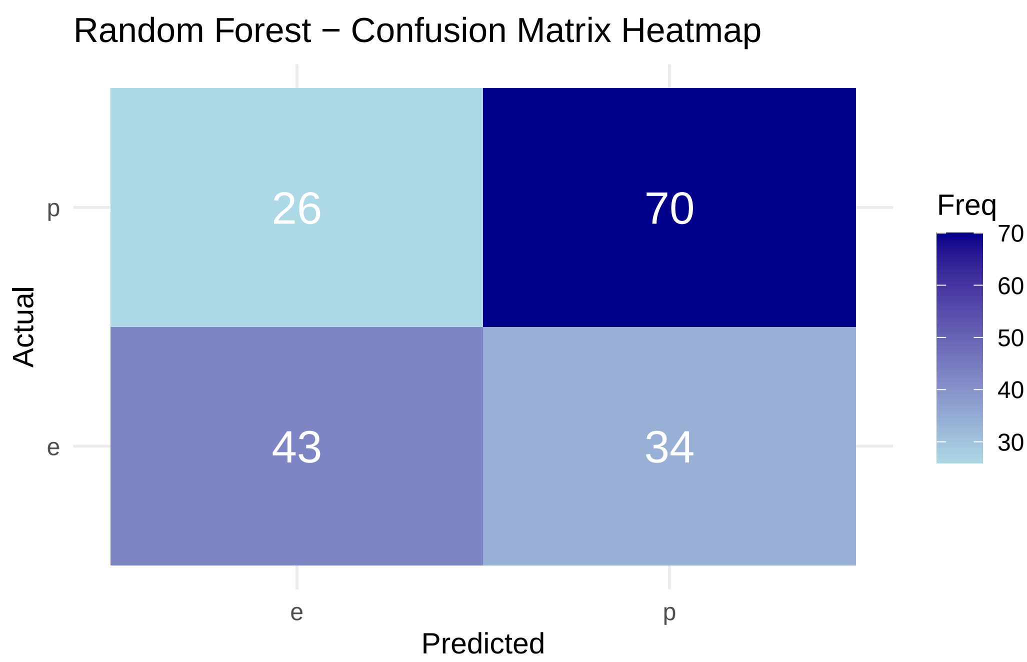


2. 隨機森林

參數設定:
- 嘗試不同參數組合，並決定樹的數量 1000，最終設定: - mtry:2

```
library(randomForest)

set.seed(123)
rf_grid <- expand.grid(
  mtry = c(2, 4, 6, 8, 10, 15)
)

rf_cv <- train(
  class ~ ., data = del_data_filter,
  method = "rf",
  trControl = ctrl,
  tuneGrid = rf_grid,
  ntree = 1000,
  metric = "Accuracy"
)
```

```
# 查看所有組合與準確率
# rf_cv$results

rf_result <- evaluate_model(rf_cv, "Random Forest")
```

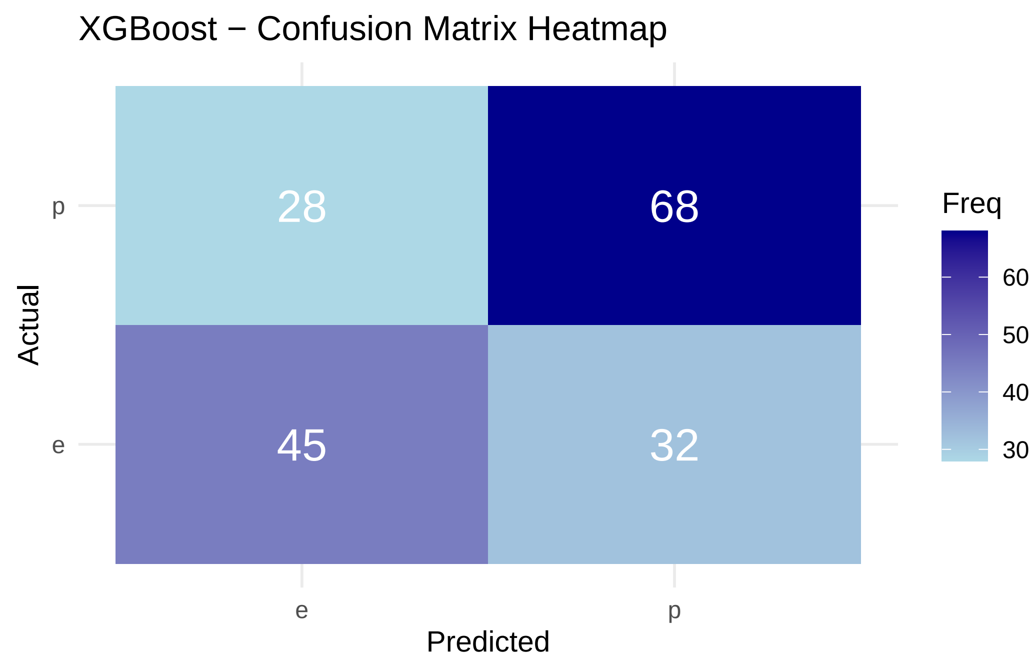


3. XGBoost

參數設定:
- nrounds = 樹的數量:300
- max_depth = 樹的深度:5
- eta = 學習率:0.1
- gamma = 控制是否允許節點分裂:1
- colsample_bytree = 抽樣變數比例:0.8
- min_child_weight = 限制節點最少樣本數:1
- subsample = 抽樣資料比例:0.5

```
library(xgboost)

set.seed(123)
xgb_grid <- expand.grid(
  nrounds = c(300, 500),
  max_depth = c(3, 5, 10),
  eta = c(0.01, 0.1),
  gamma = c(0, 1),
  colsample_bytree = c(0.5, 0.8),
  min_child_weight = c(1, 5),
  subsample = c(0.5)
)

xgb_cv <- train(
  class ~ ., data = del_data_filter,
  method = "xgbTree",
  trControl = ctrl,
  tuneGrid = xgb_grid,
  metric = "Accuracy"
)
```

```
# 最佳參數
#xgb_cv$bestTune
xgb_result <- evaluate_model(xgb_cv, "XGBoost")
```



七、比較所有模型的 Accuracy、AUC、F1 score

```
library(tidyr)
all_results <- bind_rows(
  logit_result,
  svm_result,
  rf_result,
  xgb_result
)

all_results <- all_results %>%
  pivot_wider(
    names_from = Metric,
    values_from = Value
  )
print(all_results)
```

```
# A tibble: 4 x 4
  Model               Accuracy `F1 Score`   AUC
  <chr>                  <dbl>      <dbl> <dbl>
1 Logistic Regression    0.584      0.633 0.602
2 SVM                    0.630      0.701 0.644
3 Random Forest          0.653      0.7   0.681
4 XGBoost                0.653      0.694 0.695
```

結論：綜合判斷,XGBoost 預測能力較好。

八、討論

1. 機器學習方法預測結果較好, 原因?

- 因為幾乎變項都是類別型變項, 使用 one-hot encoding, 因此變項太多了
- 變數之間非線性關係較明顯

2. 還可以再改進的方向?

- 因為主觀將兩組幾乎都是 0 或 1 的變項篩掉, 沒有考慮變數之間相關性問題, 可能還可以做個卡方檢定或 heat map 視覺化去看哪幾個變項可能有相關
- 嘗試採用其他變數篩選的方法: 如 LASSO
- 嘗試 Ensemble 合併機器學習模型

3. 其他?

- Discriminant analysis 假設基於 multivariate normal, 因此傳統統計方法中的監督式學習較不適合此資料。