R 語言與機器學習 (二)

丘祐瑋
David Chiu

評估分類模型

分類問題

- 根據已知標籤的訓練資料集(Training Set)，產生一個新模型，用以預測測試資料集(Testing Set)的標籤。

給予腫瘤的病理特徵, 預測是惡性腫瘤的可能性

資料集

- 給予腫瘤切片的特徵，預測該腫瘤是惡性腫瘤(malignant)還良性腫瘤(benign)？

1. Sample code number
2. Clump Thickness （腫塊厚度）
3. Uniformity of Cell Size （細胞大小）
4. Uniformity of Cell Shape （細胞形狀）
5. Marginal Adhesion （邊緣粘度）
6. Single Epithelial Cell Size（單獨上皮細胞大小
7. Bare Nuclei （裸細胞核）
8. Bland Chromatin（淡染色質）
9. Normal Nucleoli （正常細胞核）
10. Mitoses （分裂激素）
11. Class

讀取資料集

```
url <- 'http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data'
bc_data <- read.csv(url, header = FALSE)
colnames(bc_data) <- c("sample_code_number",
                       "clump_thickness",
                       "uniformity_of_cell_size",
                       "uniformity_of_cell_shape",
                       "marginal_adhesion",
                       "single_epithelial_cell_size",
                       "bare_nuclei",
                       "bland_chromatin",
                       "normal_nucleoli",
                       "mitosis",
                       "classes")

bc_data$classes <- ifelse(bc_data$classes == "2", "benign",
                          ifelse(bc_data$classes == "4", "malignant", NA))
```

資料清理與轉換

- 去除空值資料

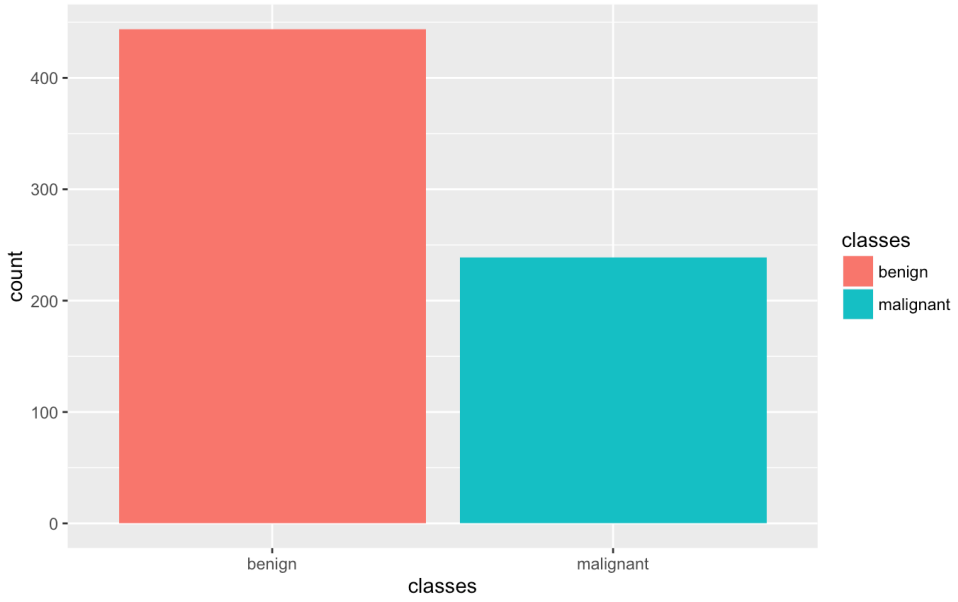
```
bc_data[bc_data == "?"] <- NA
sum(is.na(bc_data))
nrow(bc_data)

bc_data <- na.omit(bc_data)
sum(is.na(bc_data))
nrow(bc_data)
```

資料探索

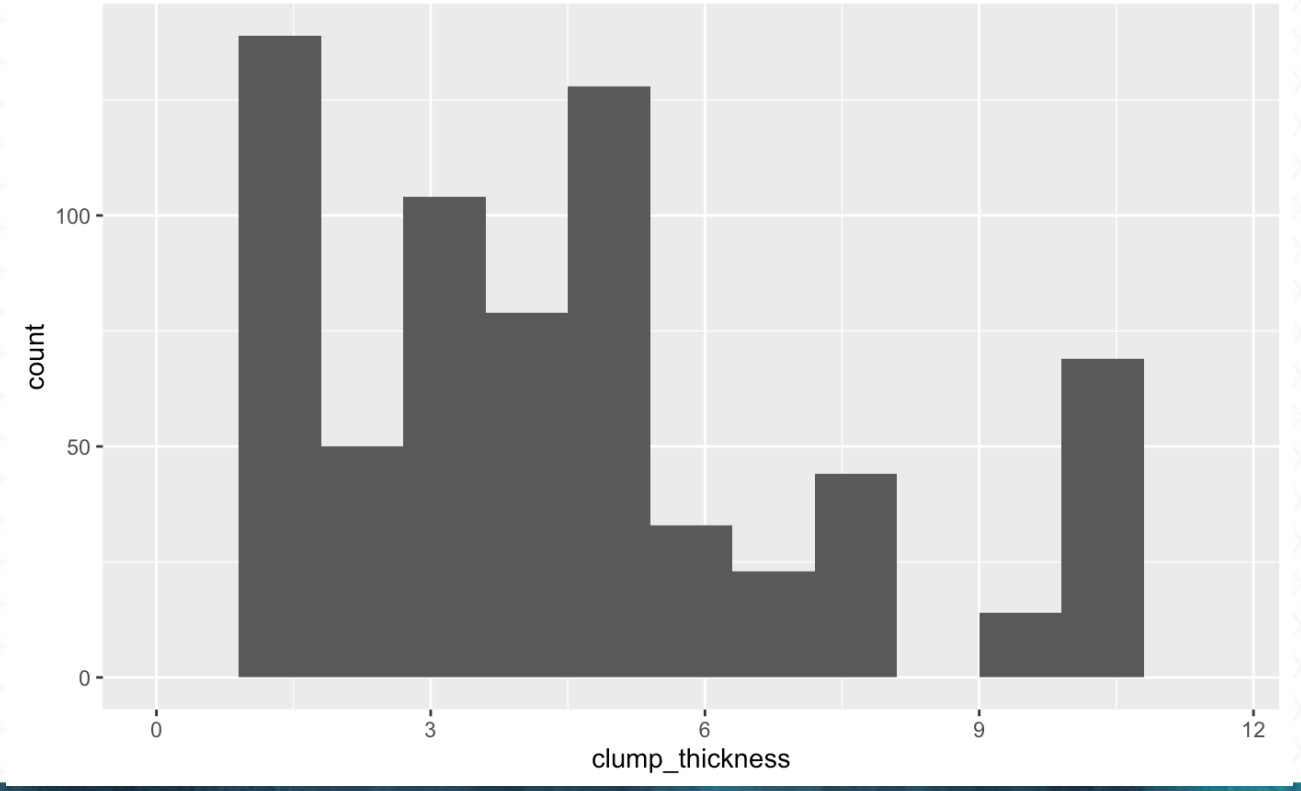
```
library(ggplot2)

ggplot(bc_data, aes(x = classes, fill = classes)) +
  geom_bar()
```



資料探索(二)

```
ggplot(bc_data, aes(x = clump_thickness)) +
  geom_histogram(bins = 10)
```

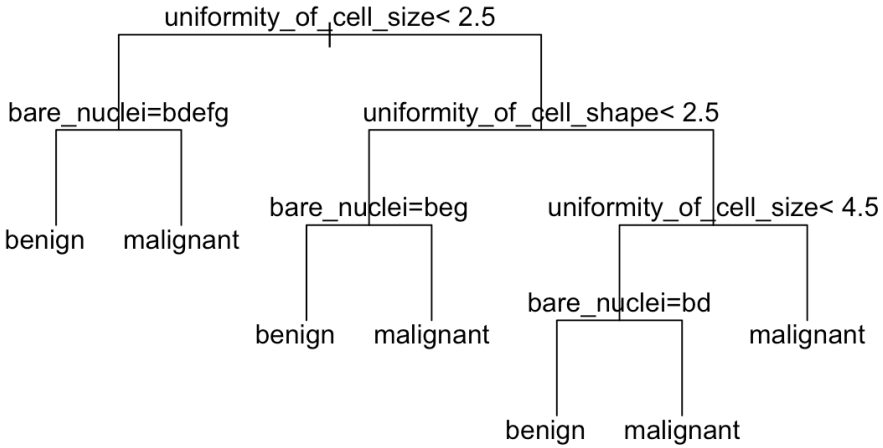


建立決策樹模型

```
set.seed(42)
fit <- rpart(classes ~ .,
             data = bc_data,
             method = "class")
```

繪製決策樹

```
plot(fit, margin = 0.1, compress = TRUE, uniform = TRUE)
text(fit)
```



計算準確率

$$準確度 = \frac{正確判斷的數量}{總判斷數量}$$

```
sum(bc_data$classes ==  predicted) / length(bc_data$classes)
```

準確率:　96.77892%

準確率

- 類別資料不平衡的情況下:
  - 假使客戶有1,000人，今天流失的客戶數量是50人，今天假使有一個預測模型的預測準確率有90%，試問這是個好的分類模型嗎?

數據如何被分類? 如何被分錯?

- 需要有方法可分解並計算由分類器產生不同類型的正誤數量
  - 需要使用混淆矩陣(Confusion Matrix)

混淆矩陣

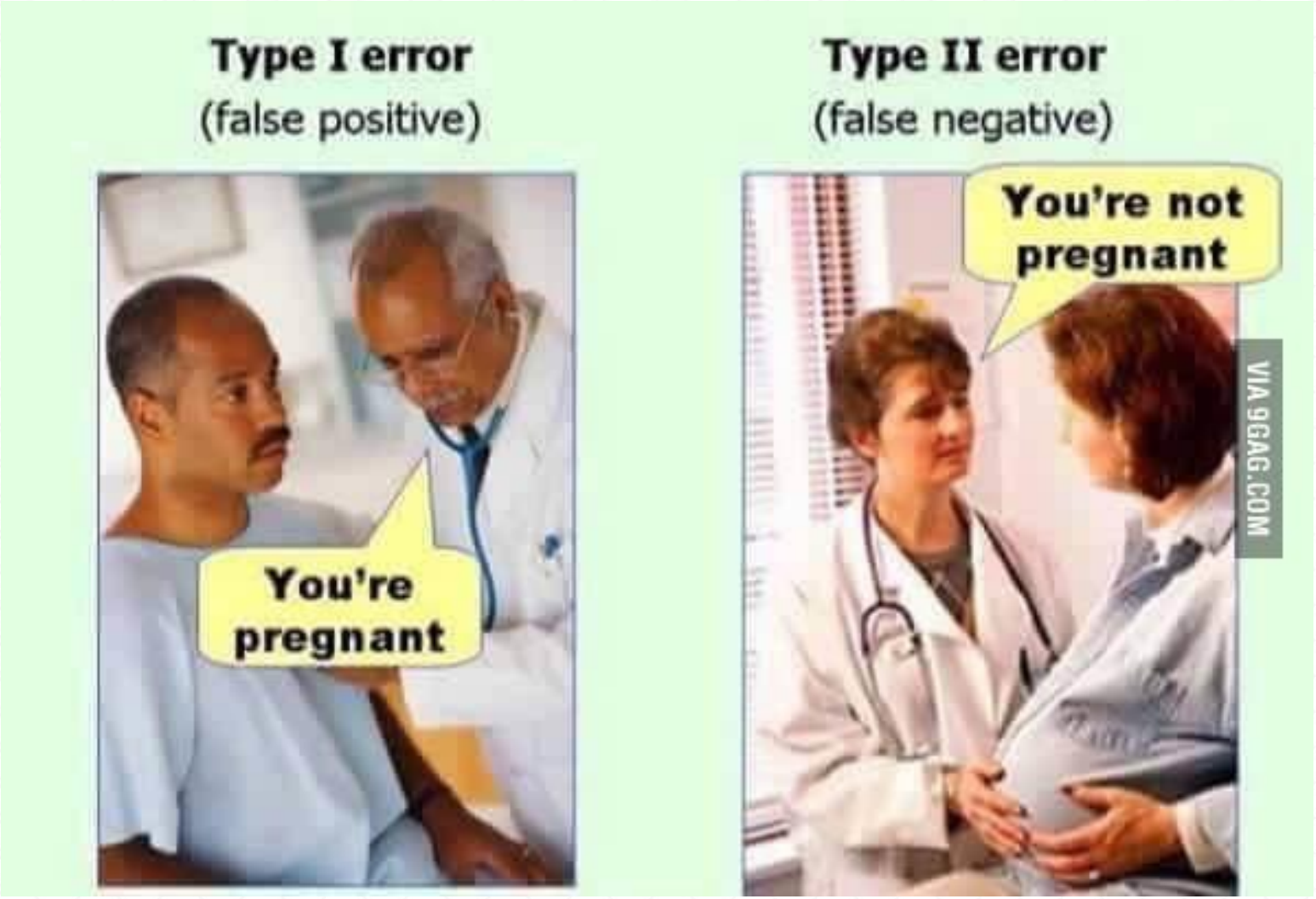
```
predicted <- predict(fit, bc_data, type = 'class')
table(bc_data$classes, predicted)
```

	predicted	
	benign	malignant
benign	431	13
malignant	9	230

混淆矩陣

- True positive ：代表檢測出有，且實際上有的狀況
- False positive：代表檢測出有，而實際上沒有的狀況
- True negative：代表檢測出無，且實際上無的狀況
- False negative：代表檢測出無，而實際上有的狀況

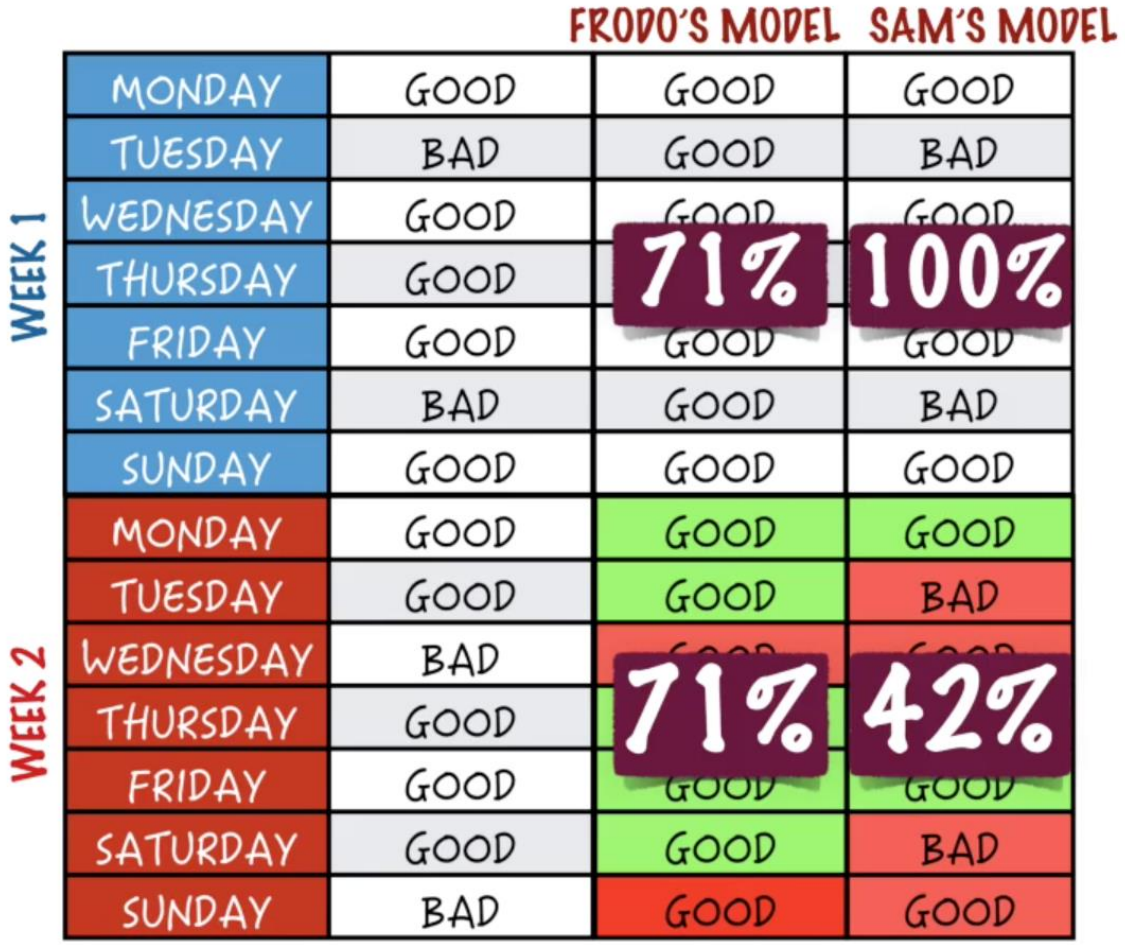
Type I & Type II Error



過度學習

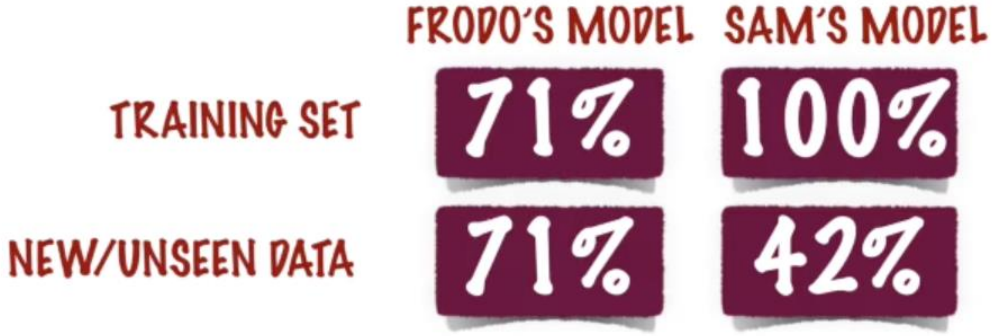
- 誤把雜訊當資訊
  - 過度假設
  - 過度解讀
  - 無法找出資料背後的事實
- 過度學習所得到的規則並非通則，只能應用於個案
  - 死記答案 v.s. 掌握原則

什麼是好模型?



Frodo 說：
這餐廳每天的服務品質都很優秀

Sam 說:
這餐廳除了星期二與星期六外, 出餐品質都很優秀



好模型的標準是, 不管在訓練還是測試資料集表現要一致

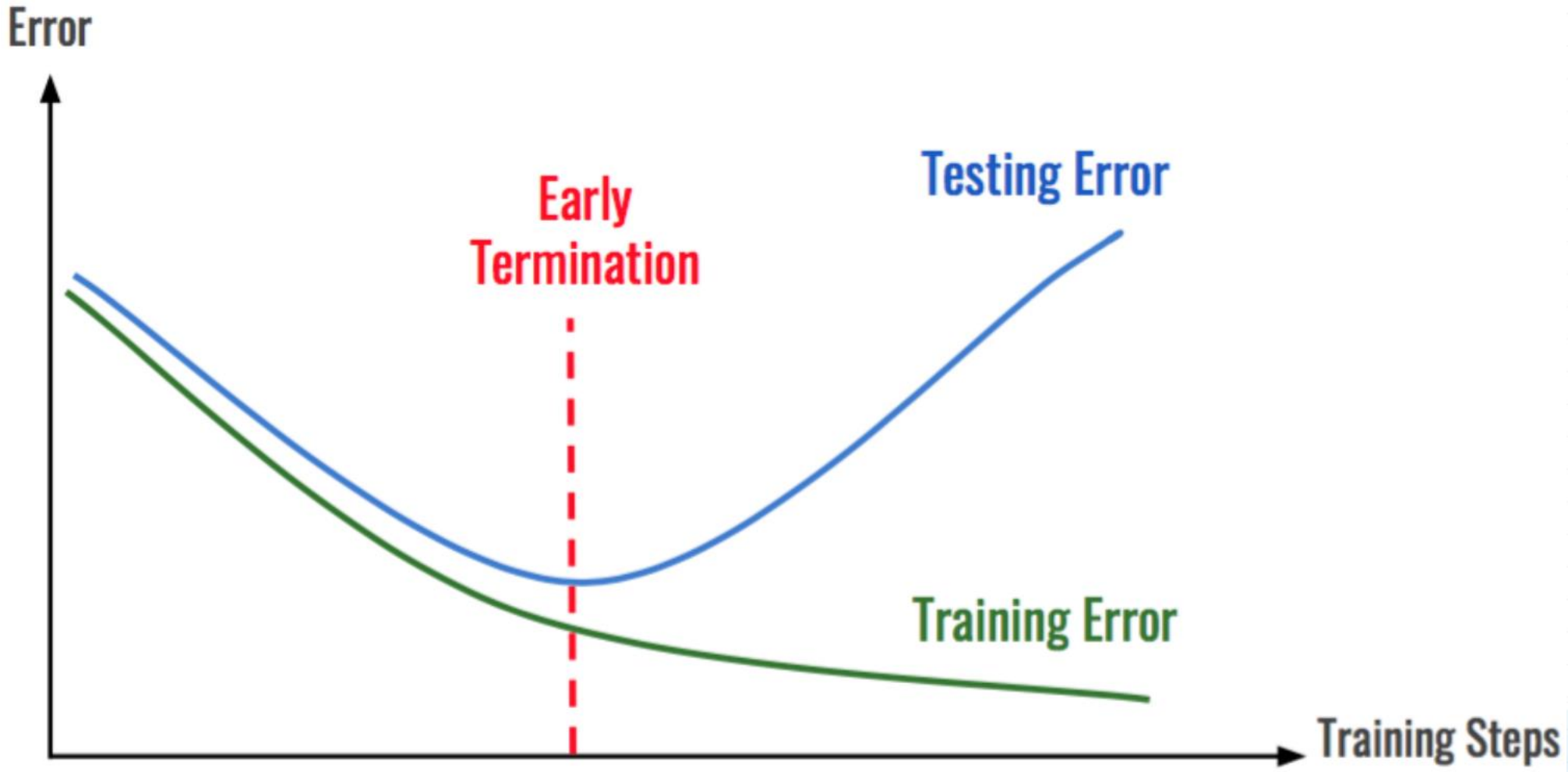
什麼是好模型?

	Low *Training* Error	High *Training* Error
Low *Testing* Error	The model is learning!	Probably some error in your code. Or you've created a *psychic* AI.
High *Testing* Error	OVERFITTING	The model is not learning.

恰到好的學習

- 如何避免過度學習(Overfitting)

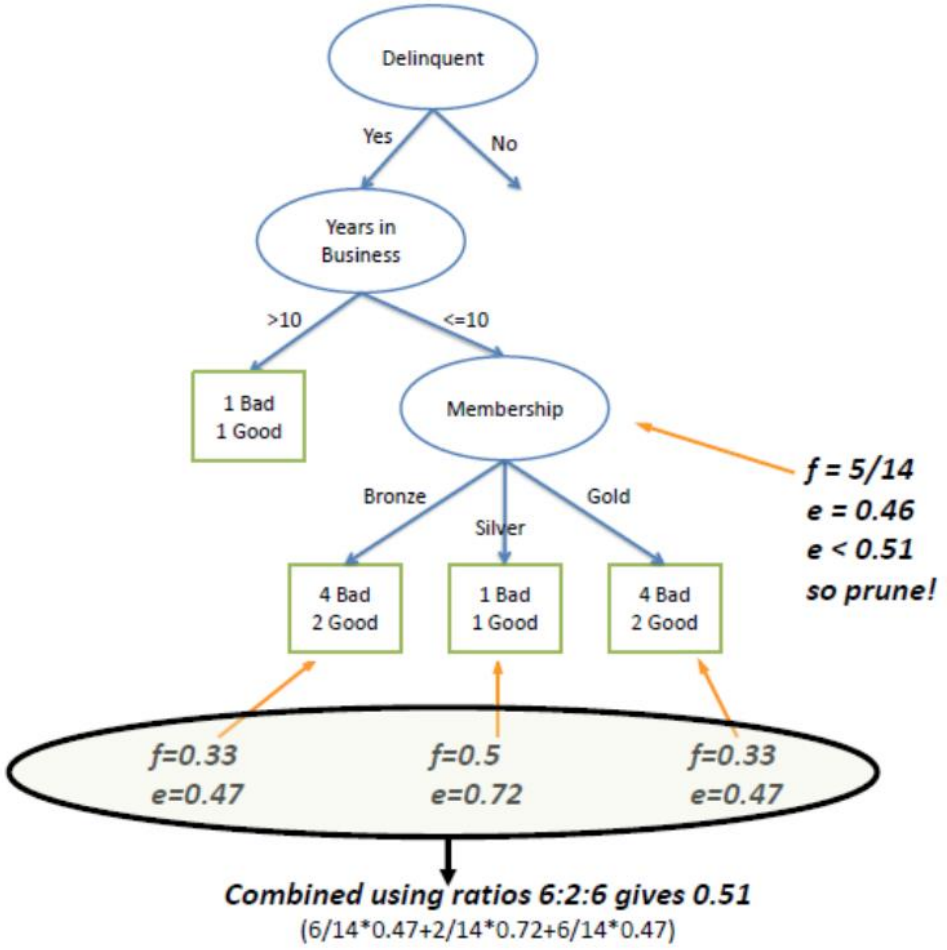
在過度適配前停止


Error
Early
Termination
Testing Error
Training Error
Training Steps

避免過度學習

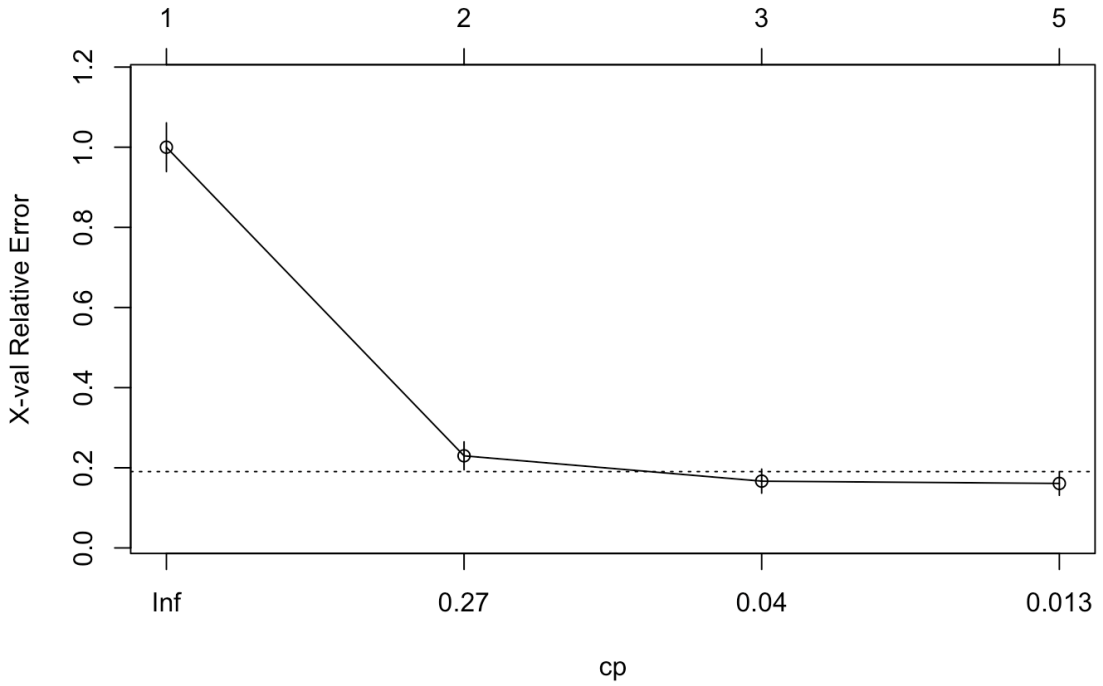
- 預先剪枝(Pre-pruning)：設定條件，當條件到達時，樹就停止生長

- 後剪枝(Post-pruning) ：等樹發展完全以後，再行剪枝



檢視剪枝條件

```
summary(fit)
printcp(fit)
plotcp(fit)
```



進行後剪枝(Post-pruning)

```
# 找出Cross Validation Error 最小的切割
min_split <- which.min(fit$cptable[,"xerror"])

# 設定停止條件
stop_criteria <- fit$cptable[min_split ,"CP"]

# 產生剪枝後的決策樹
prune.fit <- prune(fit, cp= stop_criteria)
```

Ctree 與條件推斷決策樹

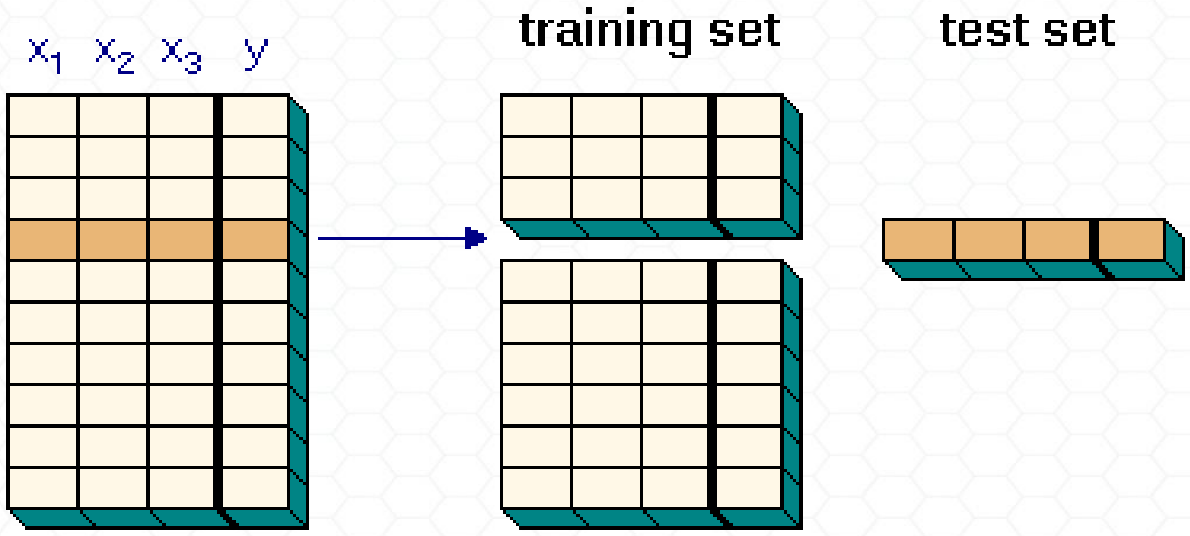
- Party
  - 根據統計檢驗來確定參數和分割點的選擇
    - 先假設所有參數與因變數均獨立
    - 對它們進行卡方獨立檢驗
    - 檢驗P值小於閥值的引數加入模型
    - 相關性最強的引數作為第一次分割的引數
  - 參數選擇好後，用置換檢驗來選擇分割點
  - 用party建立的決策樹不需要剪枝(Prune)
    - 因為閥值就決定了模型的複雜程度。

預剪枝模型

```
# 使用ctree 建立模型 (預剪枝模型)
library(party)
fit <- ctree(classes ~ ., data = train_data)
plot(fit)
```

測試模型

- 使用外部資料或是一部分的內部資料來測試資料



訓練模型與測試模型都為同一份
有球員兼裁判的嫌疑

建立訓練與測試資料集

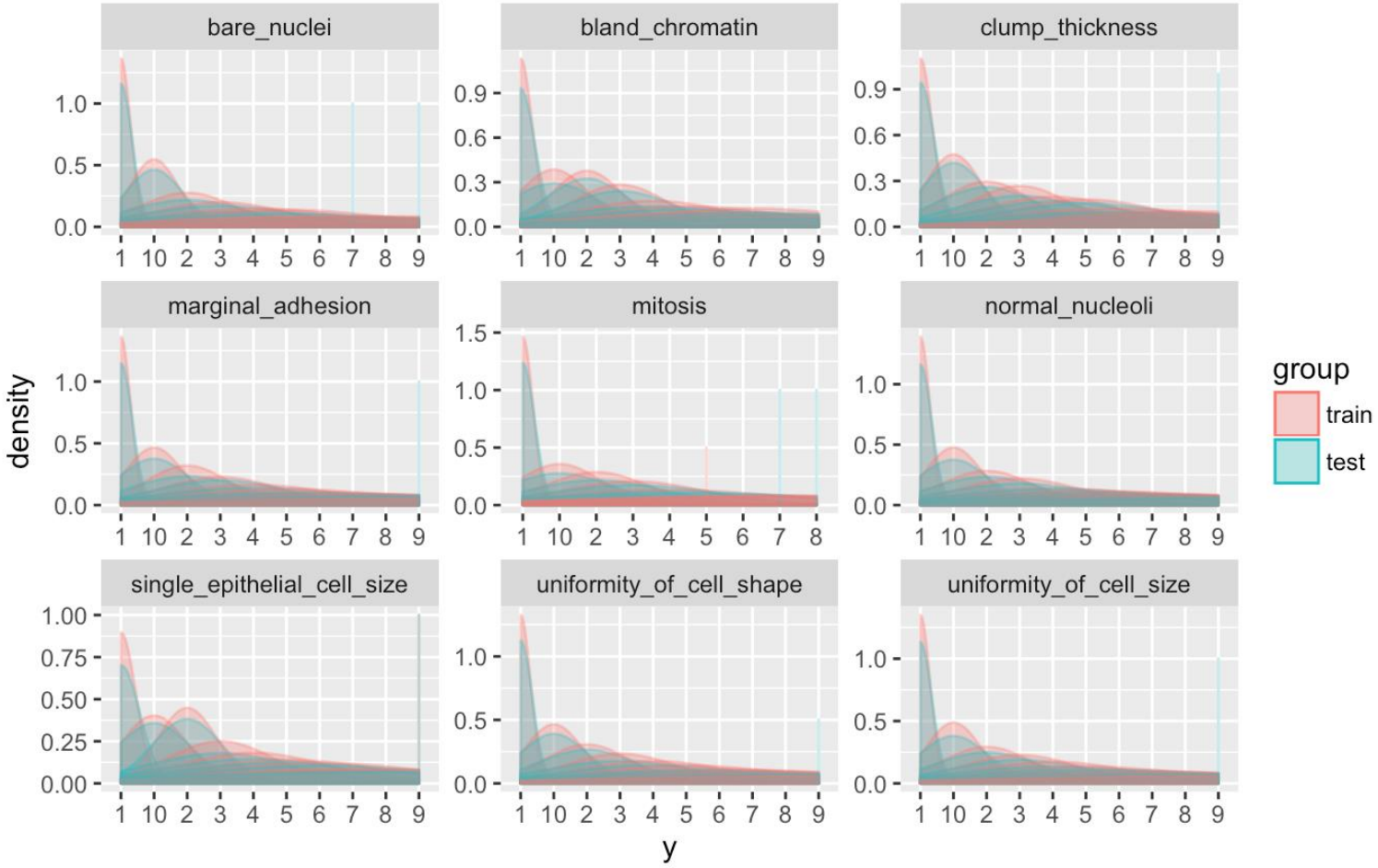
```
set.seed(42)
idx <- sample.int(2, nrow(bc_data), replace = TRUE,
prob = c(0.7,0.3))
train_data <-bc_data[idx==1,]
test_data <- bc_data[idx==2,]
```

比較訓練資料與測試資料的差別

```
library(dplyr)
library(tidyr)

rbind(data.frame(group = "train", train_data),
      data.frame(group = "test", test_data)) %>%
  gather(x, y, clump_thickness:mitosis) %>%
  ggplot(aes(x = y, color = group, fill = group)) +
    geom_density(alpha = 0.3) +
    facet_wrap( ~ x, scales = "free", ncol = 3)
```

比較訓練資料與測試資料的差別



使用訓練資料建立分類樹

```
set.seed(42)
fit <- rpart(classes ~ .,
             data = train_data,
             method = "class")

plot(fit, margin = 0.1, compress = TRUE, uniform = TRUE)
text(fit)
```

使用測試資料驗證模型

```
predicted <- predict(fit, test_data, type = 'class')
table(test_data$classes, predicted)
```

	predicted	
	benign	malignant
benign	127	6
malignant	2	63

```
sum(test_data$classes ==  predicted) / length(test_data$classes)
```

準確率:　95.9596%

進行交叉驗證

■ Holdout 驗證

隨機從最初的樣本中選出部分，形成交叉驗證數據，而剩餘的就當做訓練數據。一般來說，少於原本樣本三分之一的數據被選做驗證數據

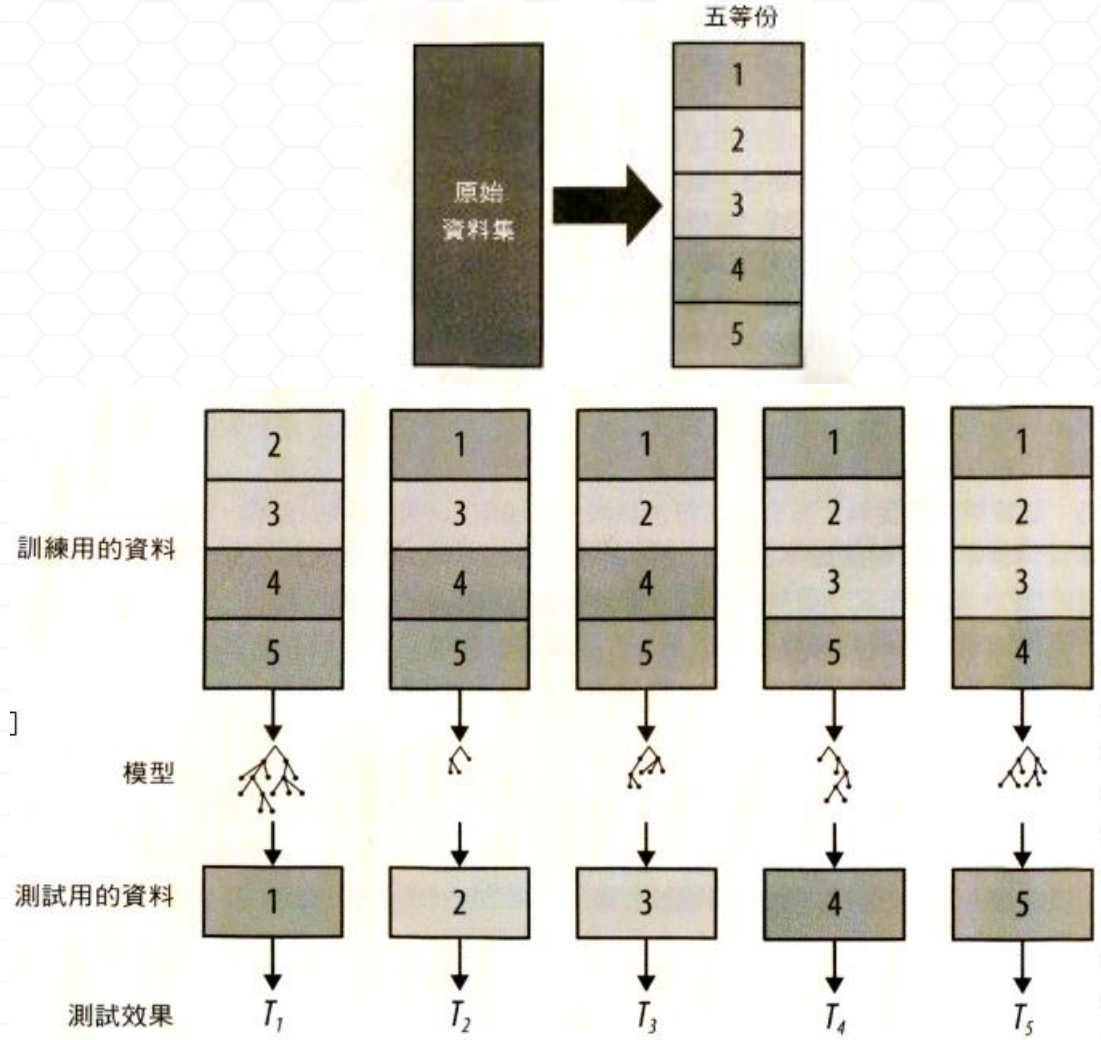
■ *K*-fold cross-validation

K次交叉驗證，初始採樣分割成K個子樣本，一個單獨的子樣本被保留作為驗證模型的數據，其他K-1個樣本用來訓練。交叉驗證重複K次

■ 留一驗證

正如名稱所建議， 留一驗證（**LOOCV**）意指只使用原本樣本中的一項來當做驗證資料， 而剩餘的則留下來當做訓練資料

交叉驗證分析 (Cross Validation)



實作 10-Fold Cross Validation

```
set.seed(123)
idx <- sample.int(10, nrow(bc_data), replace=TRUE)
models <- c()
accuracies <- c()
for(i in 1:10){
    training_set <- bc_data[idx != i, ]
    test_set <- bc_data[idx == i, ]
    fit <- rpart(classes ~., data = training_set)
    models <- c(models, fit)
    predicted <- predict(fit, test_set, type= 'class')
    acc <- sum(predicted == test_set$classes) / length(test_set$classes)
    accuracies <- c(accuracies, acc)
}
accuracies
```

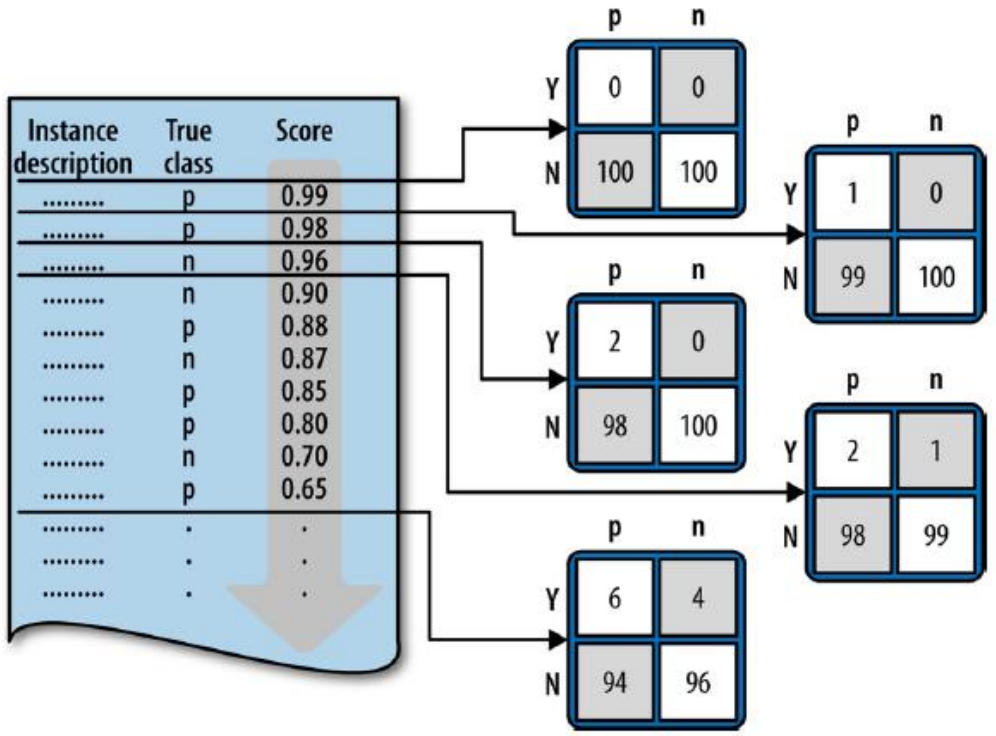
使用caret 進行交叉驗證

```
library(caret)
control <- trainControl(method="repeatedcv", number=10, repeats=3)
model <- train(classes~., data=bc_data, method="rpart", trControl=control)
model
```

交叉驗證分析的功能

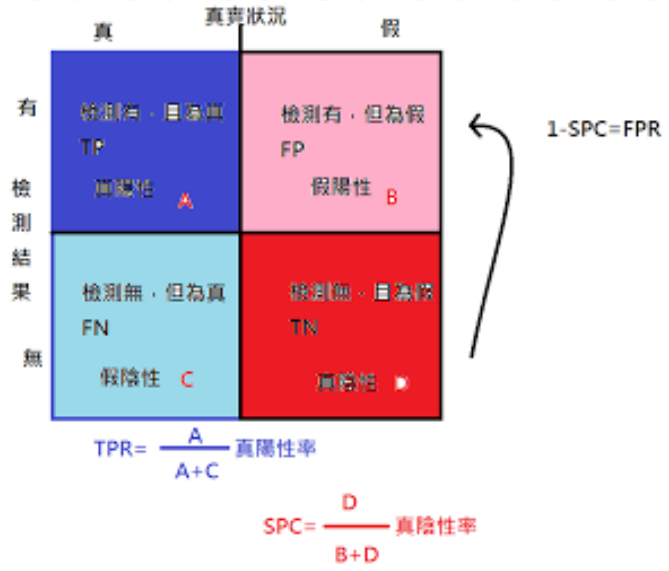
- 選擇演算法
  - 比較 SVM 與決策樹
- 調整演算法使用的參數
  - e.g. 選擇最適合的決策樹深度
- 辨認哪一些參數是有用的
  - 並不是使用所有參數都合適

考慮不同成本下的混淆矩陣

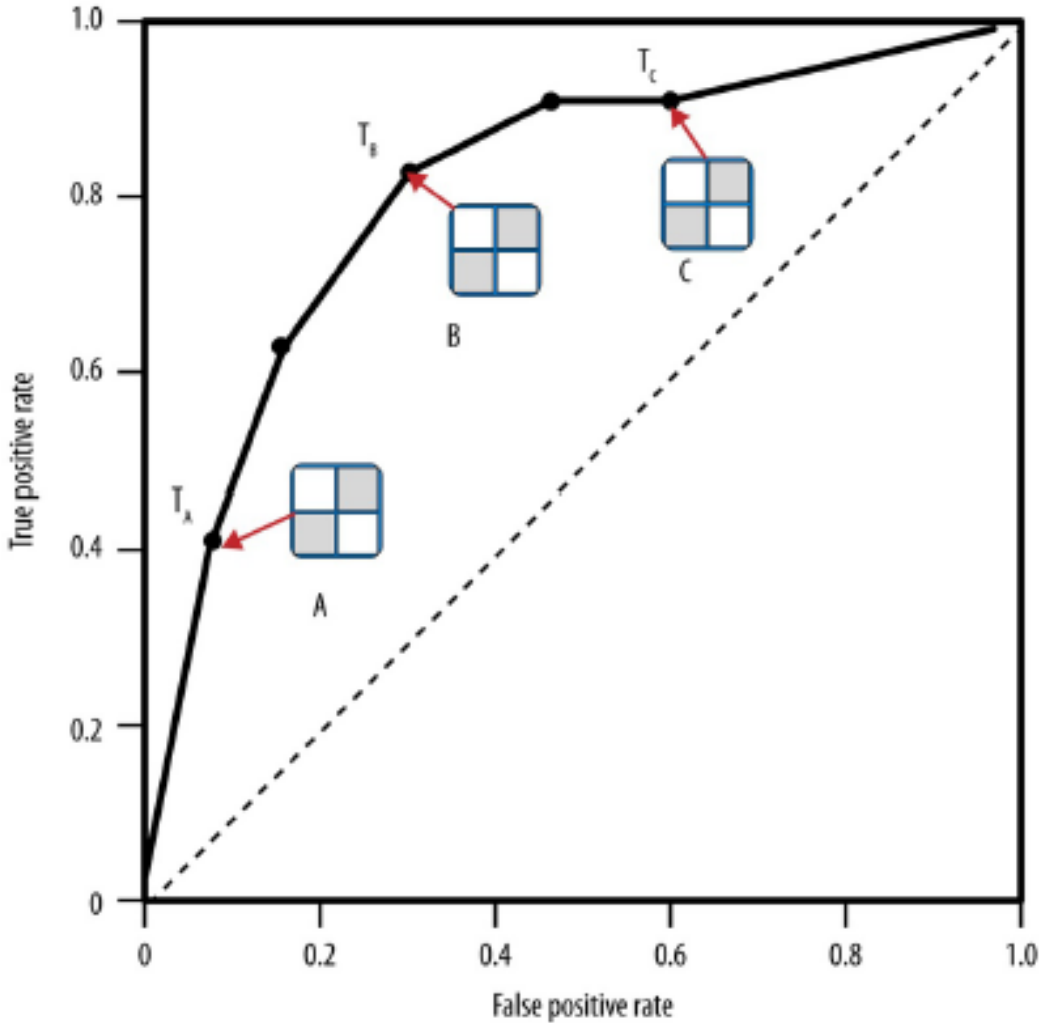


評估結果

- True positive rate：代表所有陽性樣本中，得以正確檢測出陽性結果的機率，以TP/(TP+FN)計算，又稱為靈敏度(sensitivity)。
- True negative rate，代表所有陰性樣本中，得以正確檢測出陰性結果的機率，以TN/(FP+TN)計算，又稱為特異性(specificity)。
- False positive rate：代表所有陰性樣本中，檢測出假陽性的機率，以FP/(TN+FP)計算，常以(1-SPC)的方式呈現。



如何繪製ROC 曲線

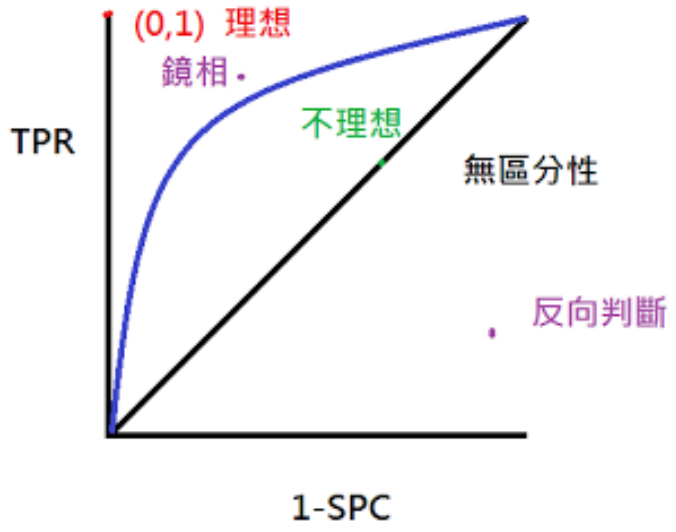

1.0
0.8
0.6
0.4
0.2
0
True positive rate
0
0.2
0.4
0.6
0.8
1.0
False positive rate
T_A
T_B
T_C
A
B
C

ROC 曲線

接收者操作特徵(receiver operating characteristic, ROC curve)

1.以假陽性率(False Positive Rate, FPR)為X軸，代表在所有陰性相本中，被判斷為陽性(假陽性)的機率，又寫為(1-特異性)。

2.以真陽性率(True Positive Rate, TPR)為Y軸，代表在所有陽性樣本中，被判斷為陽性(真陽性)的機率，又稱為敏感性



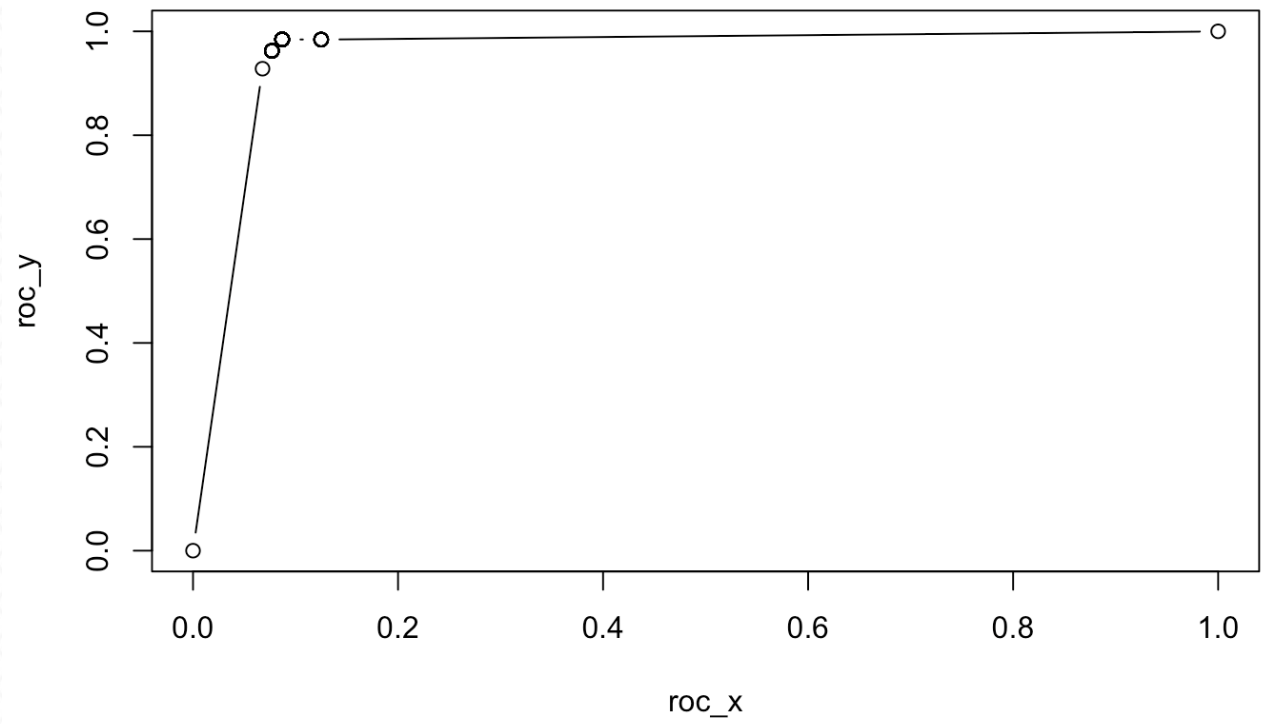
手動實作ROC 曲線

```
library(caret)
prediction <- predict(fit, test_data, type = "prob")
roc_x <- c(0)
roc_y <- c(0)
for(i in seq(0,1,0.01)){
  res <- as.factor(ifelse(prediction[,1] >= i, 'benign', 'malignant'))
  tb  <- table(test_data$classes, res)
  if (ncol(tb) == 2){
    cm <- confusionMatrix(tb)
    x <- 1 - cm$byClass[2]
    y <- cm$byClass[1]
    roc_x <- c(roc_x, x)
    roc_y <- c(roc_y, y)
  }
}
roc_x <- c(roc_x, 1)
roc_y <- c(roc_y, 1)
```

調整不同成本

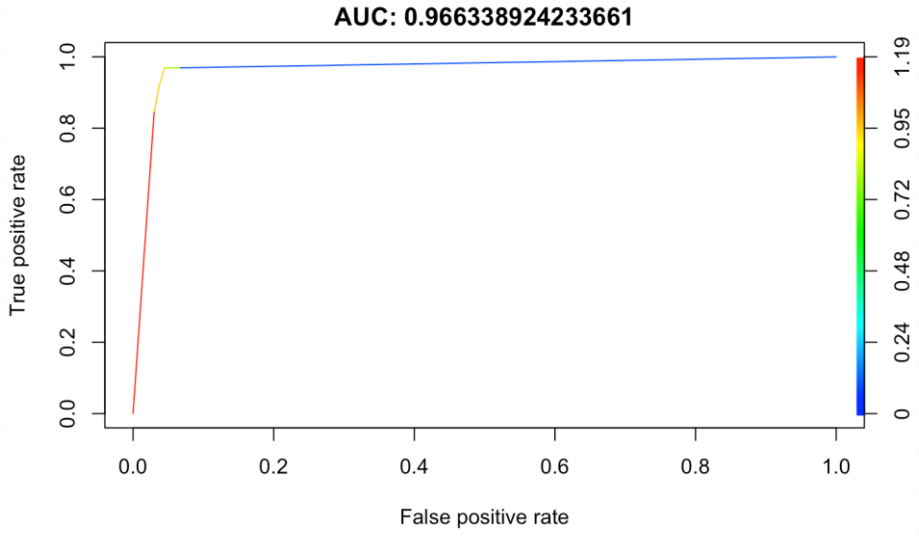
繪製 ROC Curve

```
plot(roc_x, roc_y, type='b')
```



使用 ROCR 套件

```
library(ROCR)
predictions <- predict(fit, test_data, type="prob")
pred.to.roc <- predictions[, 2]
pred.rocr <- prediction(pred.to.roc, as.factor(test_data$classes))
perf.rocr <- performance(pred.rocr, measure = "auc", x.measure = "cutoff")
perf.tpr.rocr <- performance(pred.rocr, "tpr","fpr")
plot(perf.tpr.rocr, colorize=T,main=paste("AUC:",(perf.rocr@y.values)))
```



AUC

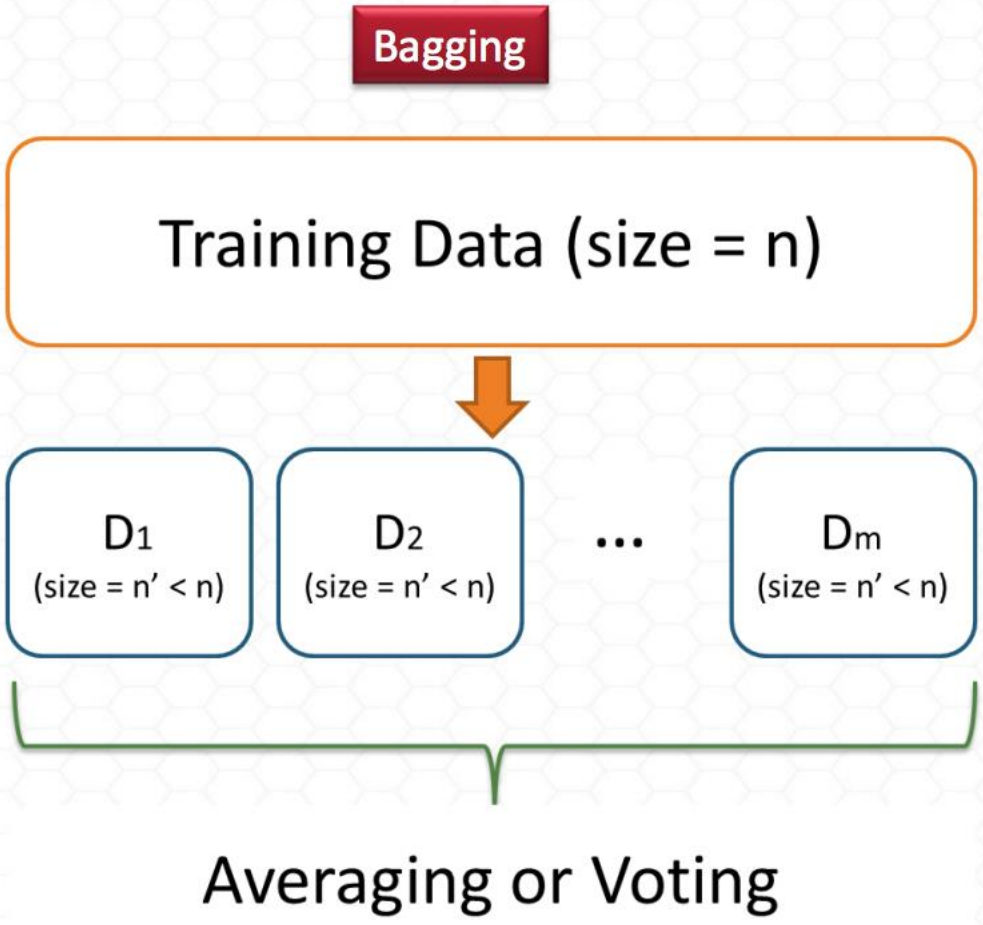
- 曲線下面積(Area Under Curve, AUC)為此篩檢方式性能優劣之指標，AUC越接近1，代表此篩檢方式效能越佳。指標可參考以下條件。

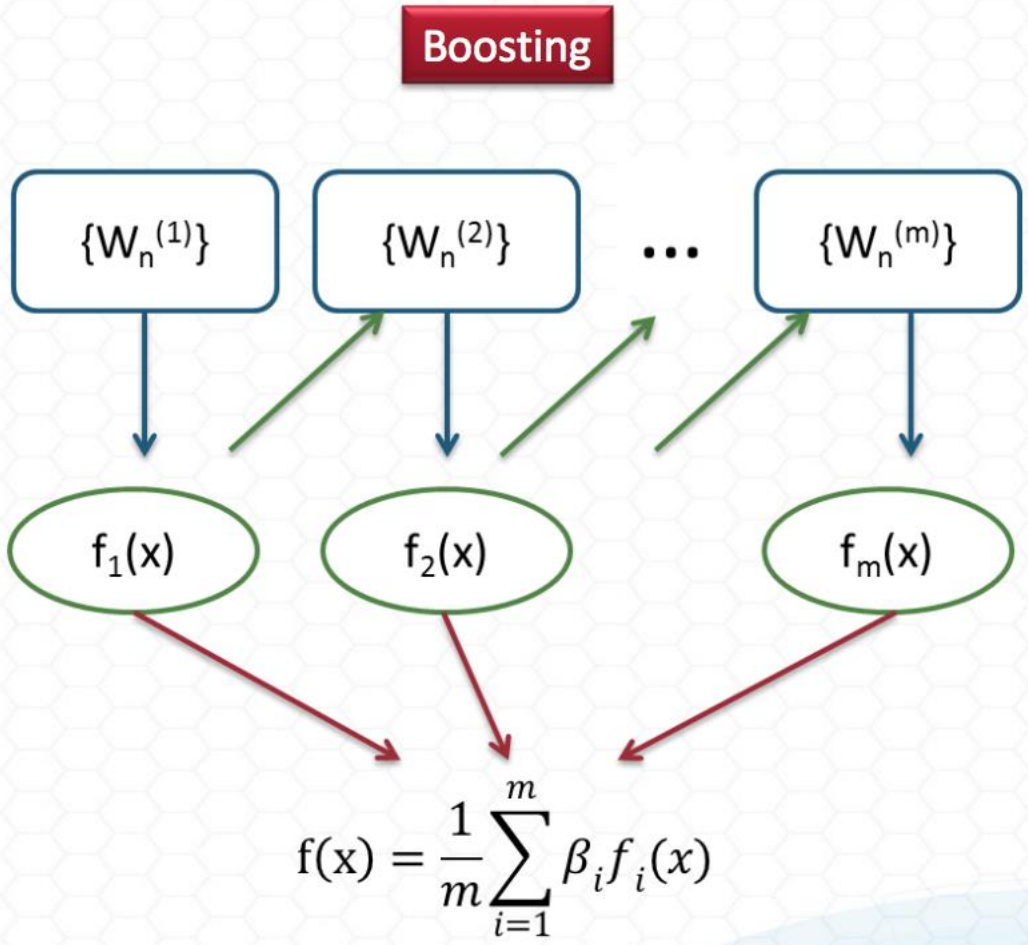
AUC數值	解釋
1	完美分類器，無論cut-off point如何設定都可正確預測。通常不存在
0.5<AUC<1	優於隨機，妥善設定可有預測價值
0.5	同隨機，預測訊息沒有價值

Ensemble 集成方法

- 奧坎剃刀 (Occam's Razor)
  - An Explanation of the data should be made as simple as possible, but no simpler
- 將多個簡單的模型組合起來, 效果會比單一模型要好

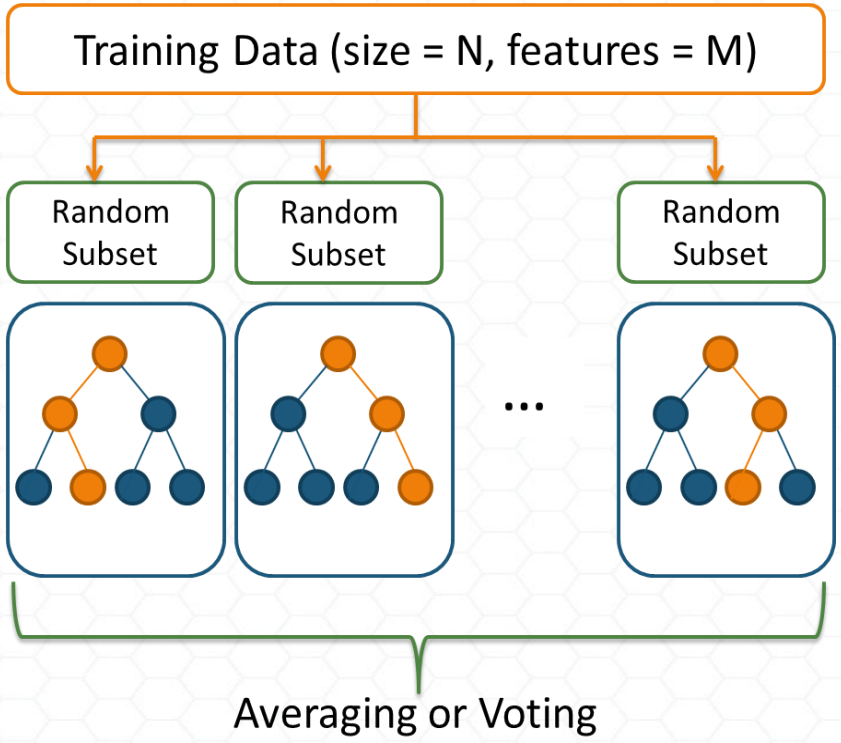
集成機器學習 (Ensemble Learning)





隨機森林 (Random Forest)

- N 多少樹, M 多少個特徵



建立隨機森林

```
library(randomForest)

train_data$classes <- as.factor(train_data$classes)
forest <- randomForest(classes ~., data = train_data, ntree=200, importance=T, proximity=T)

forest.predicted <- predict(forest, test_data, type = "class")
table(test_data$classes,forest.predicted)
```

替換演算法

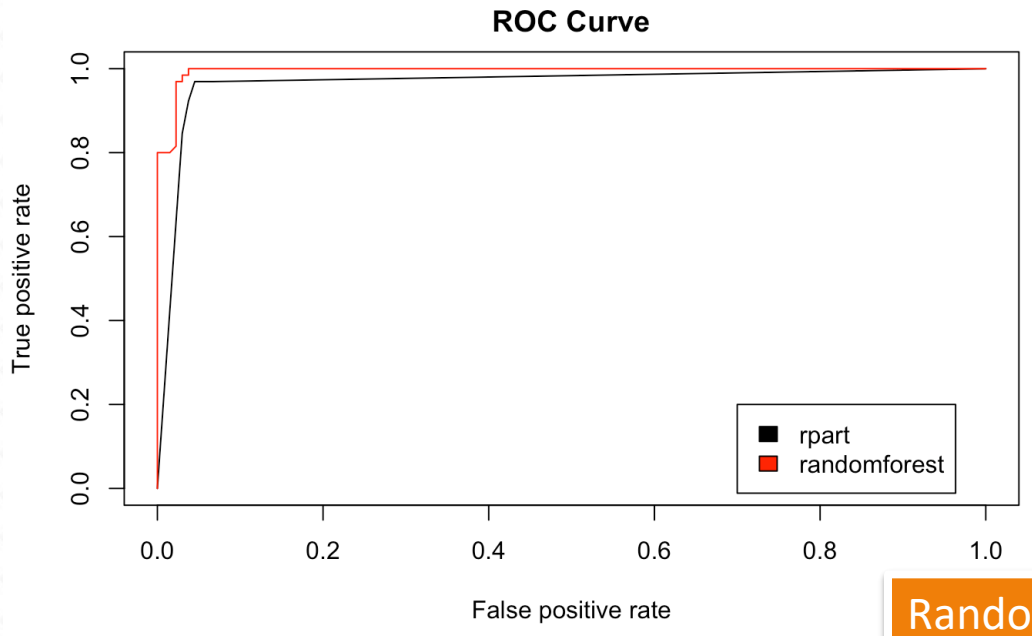
各自產生不同成本下的預測結果

```
# 決策樹
predictions1 <- predict(fit, test_data, type="prob")
pred.to.roc1 <- predictions1[, 2]
pred.rocr1 <- prediction(pred.to.roc1, as.factor(test_data$classes))
perf.rocr1 <- performance(pred.rocr1, measure = "auc", x.measure = "cutoff")
perf.tpr.rocr1 <- performance(pred.rocr1, "tpr","fpr")

# 隨機森林
predictions2 <- predict(forest, test_data, type="prob")
pred.to.roc2 <- predictions2[, 2]
pred.rocr2 <- prediction(pred.to.roc2, as.factor(test_data$classes))
perf.rocr2 <- performance(pred.rocr2, measure = "auc", x.measure = "cutoff")
perf.tpr.rocr2 <- performance(pred.rocr2, "tpr","fpr")
```

比較 ROC

```
plot(perf.tpr.rocr1,main='ROC Curve', col=1)
legend(0.7, 0.2, c('rpart', 'randomforest'), 1:2)
plot(perf.tpr.rocr2, col=2, add=TRUE)
```



Random Forest 預測能力明顯較好

THANK YOU