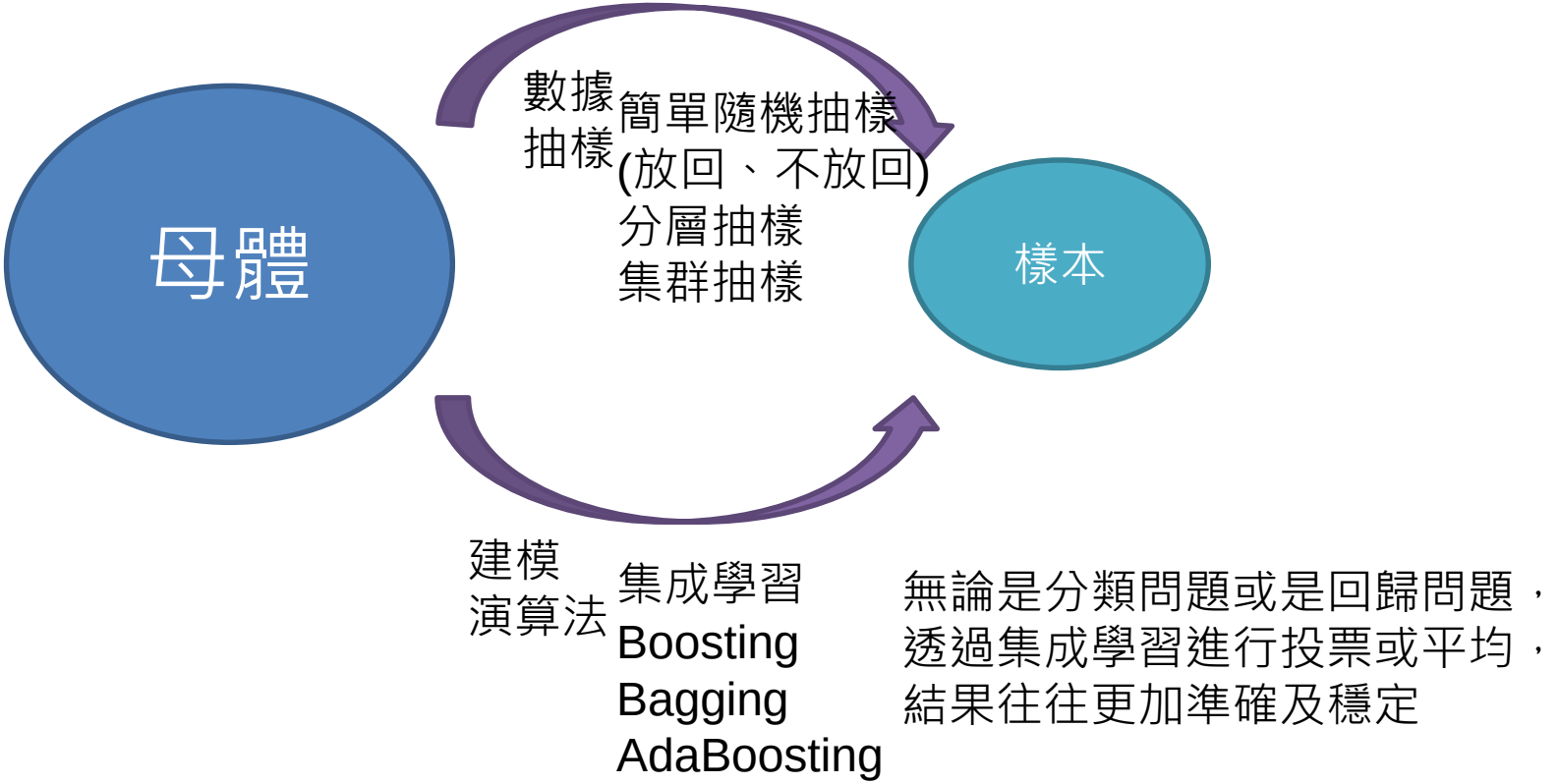
R-Ladies Taipei

R Basic
數據抽樣&集成學習

第五組
宜柔、昱璇、紹婷

概念

不可能普查，所以要抽部分樣本來取代母體



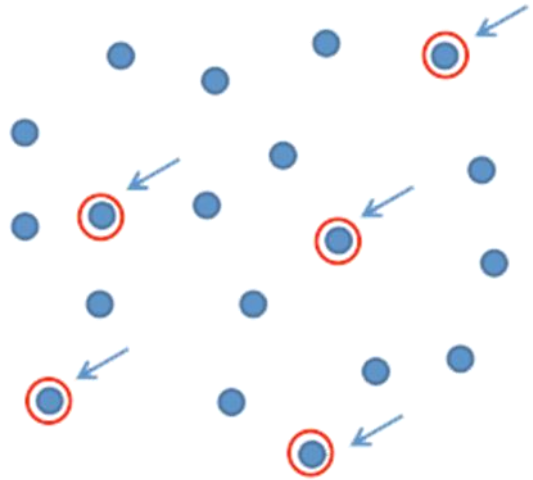
Agenda

1. 數據抽樣　　宜柔
2. 集成學習　　昱璇、紹婷

R-Ladies Taipei

簡單隨機抽樣（Simple Random Sampling)

- 特性：排除各種人為因素的抽樣，使每個樣本被抽到的比例皆相同
- 常見方法：抽籤，亂數表
- 優點：操作簡單
- 缺點：可能產生較大誤差
- 適用：母體規模較小且分佈較均勻



抽樣基本語法

Sample (x, size, replace=FALSE, prob= NULL)

```
sample(1:3, size = 3, replace = FALSE)
```

```
## [1] 1 3 2
```

```
sample(c(2,4,6), size = 4, replace = TRUE)
```

```
## [1] 4 4 4 2
```

Refernce: https://rpubs.com/mingyeh/218609

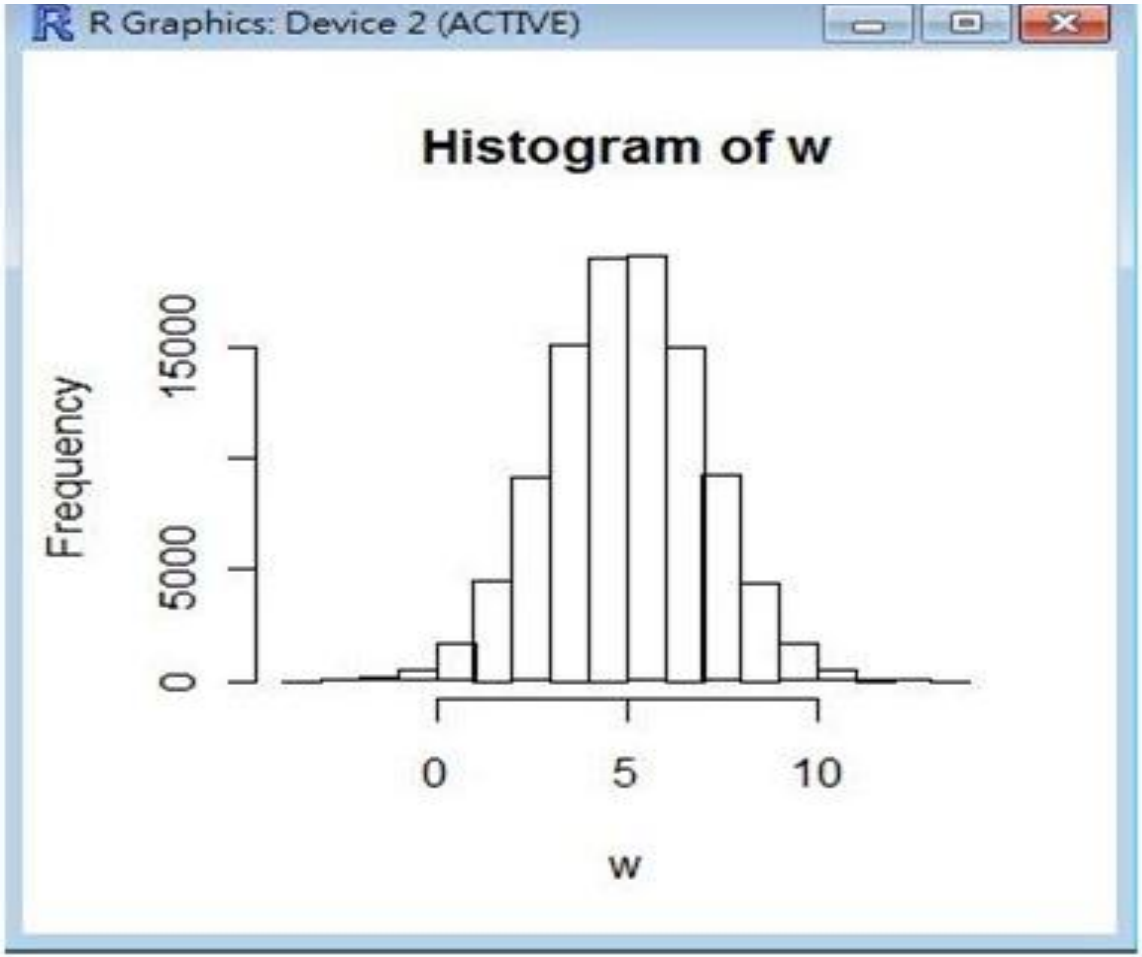
在機率模型前加 "r"，可產生隨機樣本

R-Ladies Taipei

```
> rbinom(20, 5, 0.5)
 [1] 4 3 3 4 2 4 3 1 2 3 4 3 2 2 2 4 2 3 1 1
> rpois(20, 3.5)
 [1] 2 1 4 2 1 6 3 6 1 3 3 6 6 0 4 2 6 4 6 2
> runif(20, min = 3, max = 8)
 [1] 3.933526 3.201883 7.592147 5.207603 4.897806 3.848298 4.521461 4.437873
 [9] 3.655640 5.633540 6.557995 5.430671 6.502675 5.637283 7.713699 5.841052
[17] 6.859493 5.987991 3.752924 7.480678
> rnorm(20, mean = 5.0, sd = 2.0)
 [1] 6.150209 4.743013 3.328734 5.096294 4.922795 6.272768 4.862825 8.036376
 [9] 4.198432 5.467984 2.046450 6.452511 2.088256 5.349187 3.074408 3.628072
[17] 3.421388 7.242598 3.125895 9.865341
> rexp(20, rate=2.0)
 [1] 0.17667426 0.49729383 0.12786107 0.13983412 0.44683515 1.30482842
 [7] 0.28512544 1.61472266 0.23220649 0.39089780 0.05947224 1.42892610
[13] 0.02555552 0.69409186 0.68228242 0.22542362 0.33590791 0.14684937
[19] 0.34995146 0.80595369
```

Refernce: https://rpubs.com/mingyeh/218609

```
> w = rnorm(100000, mean=5.0, sd=2.0)
> hist(w)
```



Refernce: https://rpubs.com/mingyeh/218609

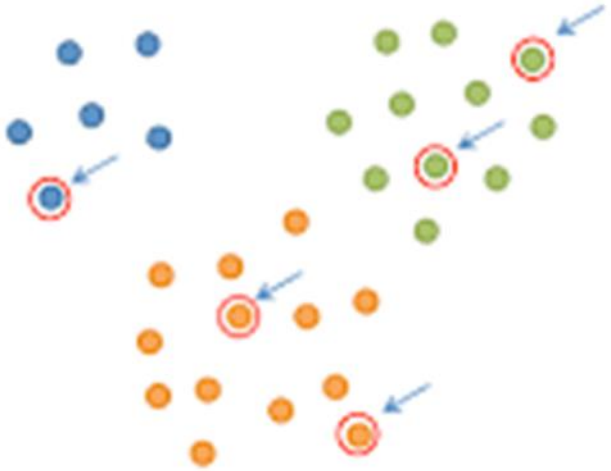
R-Ladies Taipei

分層抽樣（ Stratified Sampling)

- 特性：將母體分為幾個互斥的群體，再對各層進行簡單隨機抽樣
  （性別、年齡、 公司部門、各學系）

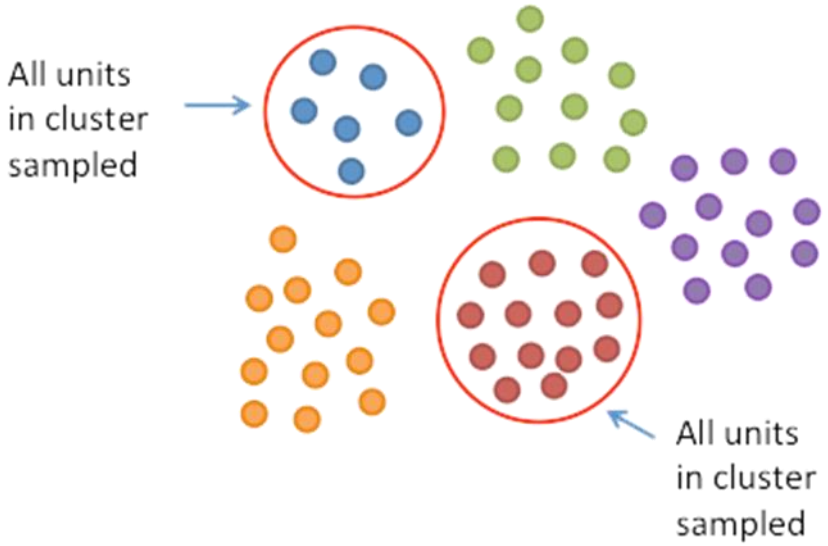
- 優點：減少誤差，提高樣本估計值的可靠性
  觀察層裡的特性
  利於層和層之間的比較

- 適用：母體規模較大、內部結構複雜



集群抽樣(Cluster Sampling)

- 方法：將數個小的樣本組合成一個較大的群集，再從其中抽出一個群集調查
- 缺點：若樣本分佈不均可能產生較大統計誤差
- 適用：各單位間的差異較大，而各群之間差異較小時



抽樣不只用在蒐集數據及建模，
演算法本身也可能包含多次抽樣，如集成學習

先來看一個案例...

XYZ公司明年股價是否會漲6%?

根據專家經驗:

1、XYZ公司職員：在過去，他有70%的時候判斷是正確的。
2、XYZ公司的財務顧問：在過去，他有75%的時候判斷是正確的。
3、股市操盤手：在過去，他有70%的時候判斷是正確的。
4、競爭對手的職員：在過去，他有60%的時候判斷是正確的。
5、同一領域的市場研究團隊：在過去，他們有75%的時候判斷是正確的。
6、社交媒體專家：在過去，他有65%的時候判斷是正確的。

組合正確率：
（假設所有預測都是相互獨立的）
1 – 30%*25%*30%*40%*25%*35%
= 1 – 0.07875 = 99.92125%

如果都是XYZ公司職員呢?還會得到99%以上的準確率嗎？

XYZ公司明年股價是否會漲6%?

根據專家經驗:

1、XYZ公司職員：在過去，他有70%的時候判斷是正確的。
2、XYZ公司職員：在過去，他有70%的時候判斷是正確的。
3、 XYZ公司職員：在過去，他有70%的時候判斷是正確的。
4、 XYZ公司職員：在過去，他有70%的時候判斷是正確的。
5、 XYZ公司職員：在過去，他有70%的時候判斷是正確的。
6、 XYZ公司職員：在過去，他有70%的時候判斷是正確的。

很顯然不會，因為這次的預測都是在相似的資訊集上做出的。他們都會受到相似資訊集的影響，並且他們建議中唯一的不同是每個人對公司有不同的看法。

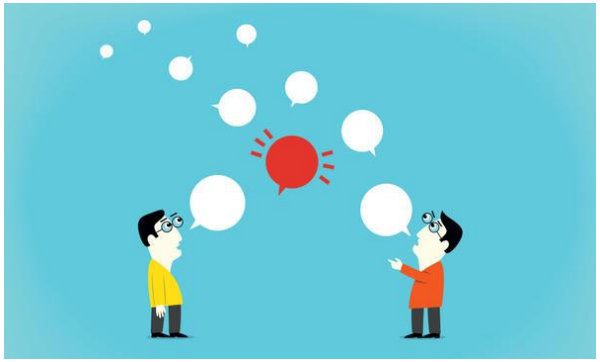
R-Ladies Taipei

導致模型不同的4個主要因素

R-Ladies Taipei

1、不同資料來源

2、不同假設



3、不同建模技術

4、初始化參數不同

Reference: http://dataunion.org/20742.html

為什麼我們要使用集成學習?

- 三個臭皮匠勝過一個諸葛亮

- 通過聯合幾個模型來說明提高機器學習結果，與單一模型相比，這種方法可以很好地提升模型的預測性能

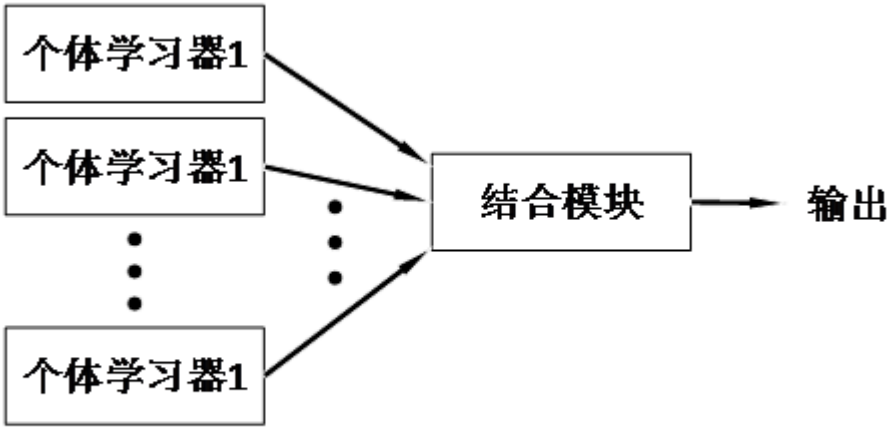


图 1集成学习示意图

參考:周志华的《机器学习》

集成學習中「整合」的主要分類

R-Ladies Taipei

blending: aggregate after getting g_t;
learning: aggregate as well as getting g_t

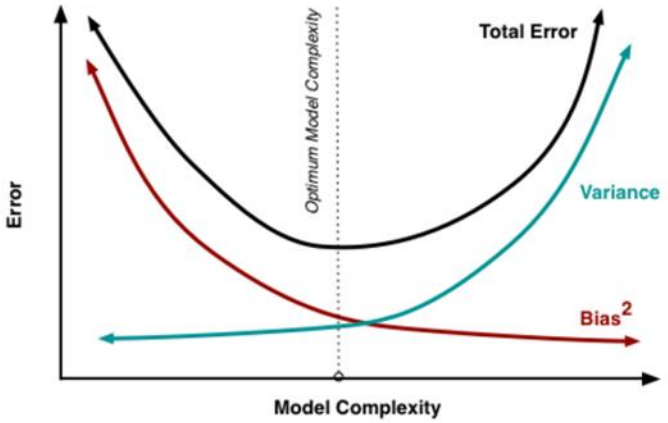
aggregation type	blending	learning
uniform	voting/averaging	Bagging
non-uniform	linear	AdaBoost
conditional	stacking	Decision Tree

集成學習中「學習」的主要分類與目的

R-Ladies Taipei

blending: aggregate after getting g_t;
learning: aggregate as well as getting g_t

aggregation type	blending	learning
uniform	voting/averaging	Bagging
non-uniform	linear	AdaBoost
conditional	stacking	Decision Tree



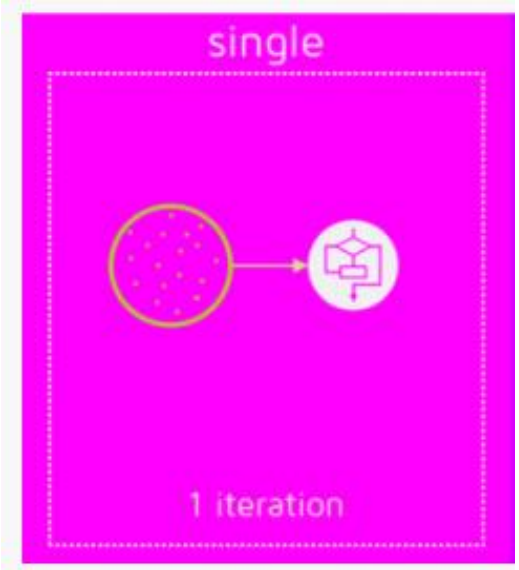
$$Err(x) = \left(E[\hat{f}(x)] - f(x)\right)^2 + E\left[\hat{f}(x) - E[\hat{f}(x)]\right]^2 + \sigma_e^2$$

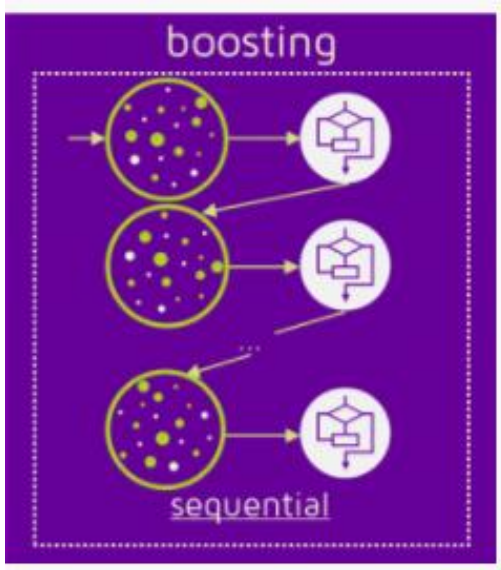
$$Err(x) = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

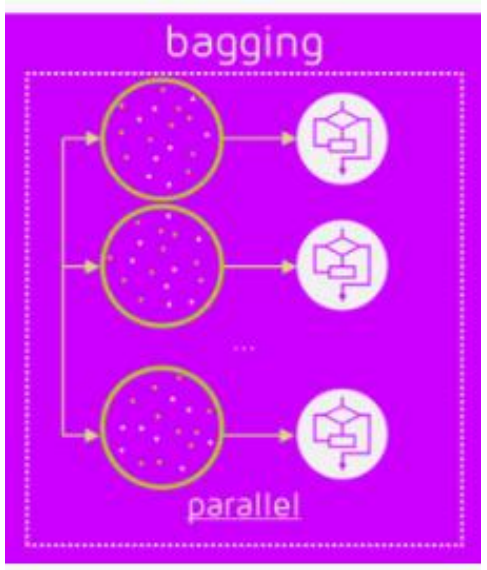
- 增加Accuracy: 關注降低bias,
  例:Boosting

- 增加Diversity: 關注降低variance,
  例: Bagging

集成學習中「學習」的主要分類與目的



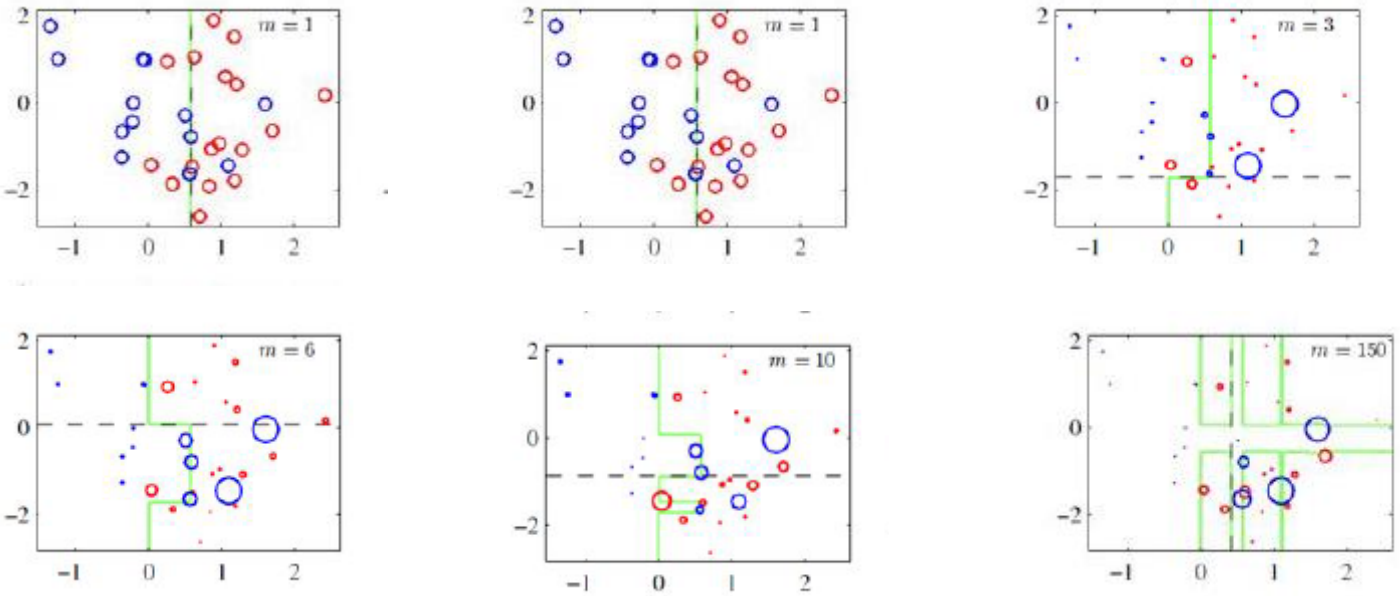




Reference: https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/

Boosting的原理

關注在被分錯的點

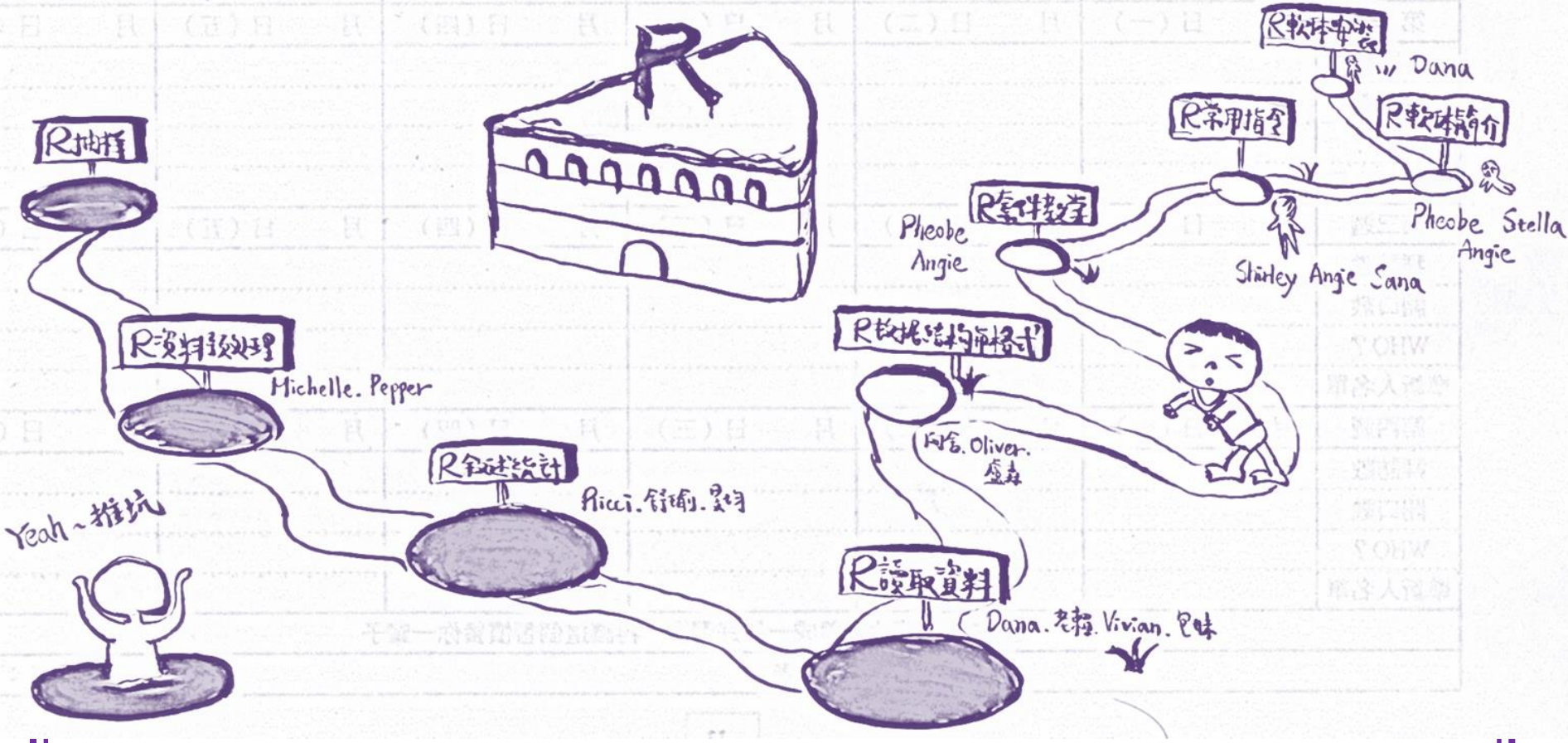


- 弱學習器 -> 強學習器
- 給定T輪內, 提升性能
- 將T個學習器綜合起來

參考:周志华的《机器学习》

2017 R-basic Schedule

完整內容: https://rladiestaipei.github.io/R-Ladies-Taipei/



案例: 預測客戶是否訂閱銀行定期存款

R-Ladies Taipei

Y: 是否訂閱定期存款

X: age, job, marital(婚姻), education, default(是否無信用違約), balance(年均餘額), housing(是否無房貸), loan(是否有個人貸款), contact, day, month, duration(最後一次聯繫的持續時間(秒)), compaign(該次項目中，聯繫總次數), pdays(最後一次聯繫距今日數), previous(該次項目前聯繫總次數), poutcome(之前營銷項目的結果)

使用套件:

Adabag (使用bagging及boosting), rpart(決策樹，可同時處理數值, 類別, 次序等各類型變量)

Ref: 數據挖掘: R語言實戰 Ch. 10

匯入資料並切分train和test資料集

```r
# data #
setwd("D:/Dana/Others/RBasic/bank-additional/bank-additional")
data=read.csv("bank-additional.csv",header=TRUE,sep=";")
head(data)
dim(data)
summary(data)
sum(data$y=="yes"); sum(data$y=="no")
sub=sample(1:nrow(data),round(nrow(data)/4))
length(sub)
data_train=data[-sub,]
data_test=data[sub,]
dim(data_train);dim(data_test)
```

結果

```
 cons.conf.idx     euribor3m      nr.employed     y
 Min.   :-50.8   Min.   :0.635   Min.   :4964   no :3668
 1st Qu.:-42.7   1st Qu.:1.334   1st Qu.:5099   yes: 451
 Median :-41.8   Median :4.857   Median :5191
 Mean   :-40.5   Mean   :3.621   Mean   :5166
 3rd Qu.:-36.4   3rd Qu.:4.961   3rd Qu.:5228
 Max.   :-26.9   Max.   :5.045   Max.   :5228

> sum(data$y=="yes"); sum(data$y=="no")
[1] 451
[1] 3668
> sub=sample(1:nrow(data),round(nrow(data)/4))
> length(sub)
[1] 1030
> data_train=data[-sub,]
> data_test=data[sub,]
> dim(data_train);dim(data_test)
[1] 3089   21
[1] 1030   21
```

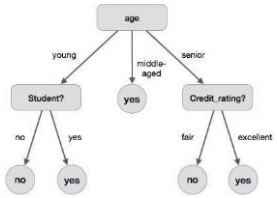
使用bagging演算法

```
# Bagging #

install.packages("adabag")
install.packages("rpart")
library(adabag)
library(rpart)

bag=bagging(y~.,data_train,mfinal=5)
names(bag)
bag$formula
bag$trees[2]
bag$votes[105:115,]
bag$prob[105:115,]
bag$class[105:115]
bag$samples[105:115,]
bag$importance
barplot(bag$importance)
```

5棵決策樹

結果

```
> bag$class[105:115]
 [1] "no" "no" "no" "no" "no" "no" "no" "no" "no" "no" "no"
> bag$samples[105:115,]
      [,1] [,2] [,3] [,4] [,5]
 [1,] 2185  860 2151 2346 1605
 [2,] 2313 1520 1682  670  178
 [3,] 2803 2834 1024 1754  395
 [4,] 1897  480 1411 1195 2623
 [5,]  760 2589 1254 2082 2608
 [6,] 1367 1243 1927 1499  581
 [7,] 2232  939 1582  191 1725
 [8,] 1892 1177 1403 1037 2446
 [9,]  541  911 1984 1749 1820
[10,] 1940 1200 2575 1696 2174
[11,] 1104 2251 1519 1615 1733
> bag$importance
          age      campaign cons.conf.idx cons.price.idx
    1.4129035     0.9551904     0.0000000     0.6736715
      contact   day_of_week       default      duration
    1.1976443     1.4922972     0.0000000    38.9805736
    education  emp.var.rate     euribor3m       housing
    4.6773608     0.3503128     1.4191590     0.9406554
          job          loan       marital         month
    6.1435929     0.0000000     0.4976807     8.1059550
```



X 5

來看bagging後的模型預測的結果如何

```
bag1=bagging(y~.,data_train,mfinal=5,control=rpart.control(maxdepth=3))
bag1$trees[2]

pre_bag=predict(bag,data_test)
names(pre_bag)
pre_bag$votes[1:10,]
pre_bag$prob[1:10,]
pre_bag$class[1:10]
pre_bag$confusion
pre_bag$error

sub_minor=which(data_test$y=="yes")
sub_major=which(data_test$y=="no")
length(sub_minor); length(sub_major)
```

決策樹演算法的控制參數

結果

```
> pre_bag$class[1:10]
 [1] "no"  "no"  "no"  "no"  "no"  "no"  "no"  "yes" "no"  "no"
> pre_bag$confusion
               Observed Class
Predicted Class  no yes
            no  890  54
            yes  31  55
> pre_bag$error
[1] 0.08252427
> sub_minor=which(data_test$y=="yes")
> sub_major=which(data_test$y=="no")
> length(sub_minor); length(sub_major)
[1] 109
[1] 921
> err_bag=sum(pre_bag$class!=data_test$y)/nrow(data_test)
> err_minor_bag=sum(pre_bag$class[sub_minor]!=data_test$y[sub_minor])/
length(sub_minor)
> err_major_bag=sum(pre_bag$class[sub_major]!=data_test$y[sub_major])/
length(sub_major)
> err_bag; err_minor_bag; err_major_bag
[1] 0.08252427
[1] 0.4954128
[1] 0.03365907
```

同樣步驟，改以boosting模型

```
err_bag=sum(pre_bag$class!=data_test$y)/nrow(data_test)
err_minor_bag=sum(pre_bag$class[sub_minor]!=data_test$y[sub_minor])/length(sub_minor)
err_major_bag=sum(pre_bag$class[sub_major]!=data_test$y[sub_major])/length(sub_major)
err_bag; err_minor_bag; err_major_bag

# Boosting #

boo=boosting(y~.,data_train,mfinal=5)
pre_boo=predict(boo,data_test)

err_boo=sum(pre_boo$class!=data_test$y)/nrow(data_test)
err_minor_boo=sum(pre_boo$class[sub_minor]!=data_test$y[sub_minor])/length(sub_minor)
err_major_boo=sum(pre_boo$class[sub_major]!=data_test$y[sub_major])/length(sub_major)
err_boo; err_minor_boo; err_major_boo
```

結果

```
> boo=boosting(y~.,data_train,mfinal=5)
> pre_boo=predict(boo,data_test)
> err_boo=sum(pre_boo$class!=data_test$y)/nrow(data_test)
> err_minor_boo=sum(pre_boo$class[sub_minor]!=data_test$y[sub_minor])/
length(sub_minor)
> err_major_boo=sum(pre_boo$class[sub_major]!=data_test$y[sub_major])/
length(sub_major)
> err_boo; err_minor_boo; err_major_boo
[1] 0.1
[1] 0.4678899
[1] 0.05646037
```

minor錯誤率下降