期末報告

412210037 林威岑 & 412210010 施柏均

2024-12-14

# Topic:Diabetes Prediction

### Import csv data

# 設定檔案的路徑  
file\_path = "C:/Users/USER/Documents/diabetes.csv"  
  
# 載入 readr 套件  
library(readr)

## Warning: 套件 'readr' 是用 R 版本 4.4.2 來建造的

# 使用 read\_csv 匯入資料  
Diabetes\_data = read\_csv(file\_path)

## Rows: 768 Columns: 9  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (9): Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, D...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

# 檢查資料  
print("Original dimesions")

## [1] "Original dimesions"

dim(Diabetes\_data)

## [1] 768 9

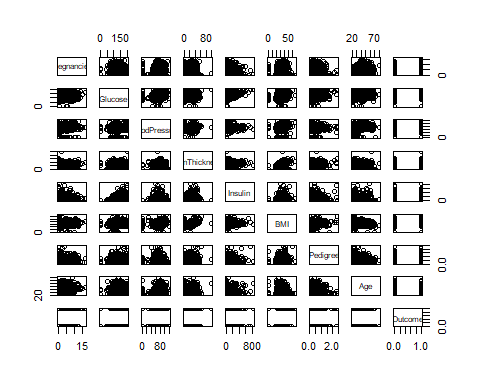
names(Diabetes\_data)

## [1] "Pregnancies" "Glucose"   
## [3] "BloodPressure" "SkinThickness"   
## [5] "Insulin" "BMI"   
## [7] "DiabetesPedigreeFunction" "Age"   
## [9] "Outcome"

summary(Diabetes\_data)

## Pregnancies Glucose BloodPressure SkinThickness   
## Min. : 0.000 Min. : 0.0 Min. : 0.00 Min. : 0.00   
## 1st Qu.: 1.000 1st Qu.: 99.0 1st Qu.: 62.00 1st Qu.: 0.00   
## Median : 3.000 Median :117.0 Median : 72.00 Median :23.00   
## Mean : 3.845 Mean :120.9 Mean : 69.11 Mean :20.54   
## 3rd Qu.: 6.000 3rd Qu.:140.2 3rd Qu.: 80.00 3rd Qu.:32.00   
## Max. :17.000 Max. :199.0 Max. :122.00 Max. :99.00   
## Insulin BMI DiabetesPedigreeFunction Age   
## Min. : 0.0 Min. : 0.00 Min. :0.0780 Min. :21.00   
## 1st Qu.: 0.0 1st Qu.:27.30 1st Qu.:0.2437 1st Qu.:24.00   
## Median : 30.5 Median :32.00 Median :0.3725 Median :29.00   
## Mean : 79.8 Mean :31.99 Mean :0.4719 Mean :33.24   
## 3rd Qu.:127.2 3rd Qu.:36.60 3rd Qu.:0.6262 3rd Qu.:41.00   
## Max. :846.0 Max. :67.10 Max. :2.4200 Max. :81.00   
## Outcome   
## Min. :0.000   
## 1st Qu.:0.000   
## Median :0.000   
## Mean :0.349   
## 3rd Qu.:1.000   
## Max. :1.000

# 觀察資料之間的關係  
pairs(Diabetes\_data)



cor(Diabetes\_data)

## Pregnancies Glucose BloodPressure SkinThickness  
## Pregnancies 1.00000000 0.12945867 0.14128198 -0.08167177  
## Glucose 0.12945867 1.00000000 0.15258959 0.05732789  
## BloodPressure 0.14128198 0.15258959 1.00000000 0.20737054  
## SkinThickness -0.08167177 0.05732789 0.20737054 1.00000000  
## Insulin -0.07353461 0.33135711 0.08893338 0.43678257  
## BMI 0.01768309 0.22107107 0.28180529 0.39257320  
## DiabetesPedigreeFunction -0.03352267 0.13733730 0.04126495 0.18392757  
## Age 0.54434123 0.26351432 0.23952795 -0.11397026  
## Outcome 0.22189815 0.46658140 0.06506836 0.07475223  
## Insulin BMI DiabetesPedigreeFunction  
## Pregnancies -0.07353461 0.01768309 -0.03352267  
## Glucose 0.33135711 0.22107107 0.13733730  
## BloodPressure 0.08893338 0.28180529 0.04126495  
## SkinThickness 0.43678257 0.39257320 0.18392757  
## Insulin 1.00000000 0.19785906 0.18507093  
## BMI 0.19785906 1.00000000 0.14064695  
## DiabetesPedigreeFunction 0.18507093 0.14064695 1.00000000  
## Age -0.04216295 0.03624187 0.03356131  
## Outcome 0.13054795 0.29269466 0.17384407  
## Age Outcome  
## Pregnancies 0.54434123 0.22189815  
## Glucose 0.26351432 0.46658140  
## BloodPressure 0.23952795 0.06506836  
## SkinThickness -0.11397026 0.07475223  
## Insulin -0.04216295 0.13054795  
## BMI 0.03624187 0.29269466  
## DiabetesPedigreeFunction 0.03356131 0.17384407  
## Age 1.00000000 0.23835598  
## Outcome 0.23835598 1.00000000

# 資料前處理(刪除等於0的欄位，但不包含懷孕變數中的0以及判斷結果的0)  
Diabetes\_data = Diabetes\_data[rowSums(Diabetes\_data[, -c(1,9)] == 0)==0,]  
print("dimesions after delete rows which has 0\n")

## [1] "dimesions after delete rows which has 0\n"

dim(Diabetes\_data)

## [1] 392 9

# 處理完後，再引入資料   
attach(Diabetes\_data)  
knitr::opts\_chunk$set(echo = TRUE)

### Spilt half data into training set & testing set

set.seed(1)  
train = sample(1:nrow(Diabetes\_data), nrow(Diabetes\_data)/2)  
test = rep(1:nrow(Diabetes\_data))  
test = test[-train]

### Lec1.Linear Regression

lm.fit = lm(Outcome ~ Pregnancies + Glucose + BloodPressure + BMI + DiabetesPedigreeFunction + Age + SkinThickness + Insulin, data = Diabetes\_data)  
summary(lm.fit)

##   
## Call:  
## lm(formula = Outcome ~ Pregnancies + Glucose + BloodPressure +   
## BMI + DiabetesPedigreeFunction + Age + SkinThickness + Insulin,   
## data = Diabetes\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.07966 -0.25711 -0.06177 0.25851 1.03750   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.103e+00 1.436e-01 -7.681 1.34e-13 \*\*\*  
## Pregnancies 1.295e-02 8.364e-03 1.549 0.12230   
## Glucose 6.409e-03 8.159e-04 7.855 4.07e-14 \*\*\*  
## BloodPressure 5.465e-05 1.730e-03 0.032 0.97482   
## BMI 9.325e-03 3.901e-03 2.391 0.01730 \*   
## DiabetesPedigreeFunction 1.572e-01 5.804e-02 2.708 0.00707 \*\*   
## Age 5.878e-03 2.787e-03 2.109 0.03559 \*   
## SkinThickness 1.678e-03 2.522e-03 0.665 0.50631   
## Insulin -1.233e-04 2.045e-04 -0.603 0.54681   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3853 on 383 degrees of freedom  
## Multiple R-squared: 0.3458, Adjusted R-squared: 0.3321   
## F-statistic: 25.3 on 8 and 383 DF, p-value: < 2.2e-16

lm.fit = lm(Outcome ~ Pregnancies + Glucose + BMI +DiabetesPedigreeFunction, data = Diabetes\_data)  
summary(lm.fit)

##   
## Call:  
## lm(formula = Outcome ~ Pregnancies + Glucose + BMI + DiabetesPedigreeFunction,   
## data = Diabetes\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.17701 -0.25751 -0.07474 0.26297 1.03755   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.0109077 0.1120866 -9.019 < 2e-16 \*\*\*  
## Pregnancies 0.0256428 0.0062174 4.124 4.55e-05 \*\*\*  
## Glucose 0.0065590 0.0006655 9.856 < 2e-16 \*\*\*  
## BMI 0.0110866 0.0028731 3.859 0.000133 \*\*\*  
## DiabetesPedigreeFunction 0.1658826 0.0575870 2.881 0.004191 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.386 on 387 degrees of freedom  
## Multiple R-squared: 0.3364, Adjusted R-squared: 0.3295   
## F-statistic: 49.04 on 4 and 387 DF, p-value: < 2.2e-16

lm.fit = lm(Outcome ~ Pregnancies + Glucose + BloodPressure + BMI + DiabetesPedigreeFunction + Insulin, data = Diabetes\_data)  
summary(lm.fit)

##   
## Call:  
## lm(formula = Outcome ~ Pregnancies + Glucose + BloodPressure +   
## BMI + DiabetesPedigreeFunction + Insulin, data = Diabetes\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.13946 -0.26093 -0.07272 0.25821 1.05359   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.059e+00 1.405e-01 -7.539 3.42e-13 \*\*\*  
## Pregnancies 2.499e-02 6.369e-03 3.924 0.000103 \*\*\*  
## Glucose 6.723e-03 8.068e-04 8.333 1.41e-15 \*\*\*  
## BloodPressure 7.110e-04 1.710e-03 0.416 0.677814   
## BMI 1.087e-02 3.044e-03 3.571 0.000401 \*\*\*  
## DiabetesPedigreeFunction 1.693e-01 5.798e-02 2.920 0.003711 \*\*   
## Insulin -9.223e-05 2.048e-04 -0.450 0.652722   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3868 on 385 degrees of freedom  
## Multiple R-squared: 0.337, Adjusted R-squared: 0.3267   
## F-statistic: 32.62 on 6 and 385 DF, p-value: < 2.2e-16

lm.fit = lm(Outcome ~ Age + Glucose + BMI + DiabetesPedigreeFunction, data = Diabetes\_data)  
 summary(lm.fit)

##   
## Call:  
## lm(formula = Outcome ~ Age + Glucose + BMI + DiabetesPedigreeFunction,   
## data = Diabetes\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.11846 -0.26145 -0.07393 0.27717 1.04901   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.1183739 0.1164806 -9.601 < 2e-16 \*\*\*  
## Age 0.0088972 0.0020341 4.374 1.57e-05 \*\*\*  
## Glucose 0.0061314 0.0006882 8.909 < 2e-16 \*\*\*  
## BMI 0.0103822 0.0028591 3.631 0.00032 \*\*\*  
## DiabetesPedigreeFunction 0.1529780 0.0574792 2.661 0.00810 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.385 on 387 degrees of freedom  
## Multiple R-squared: 0.3398, Adjusted R-squared: 0.333   
## F-statistic: 49.8 on 4 and 387 DF, p-value: < 2.2e-16

knitr::opts\_chunk$set(echo = TRUE)

### Lec2.use logistic regression

# fitting glm model with all parameters  
glm.fit = glm(Outcome~Pregnancies + Glucose + BloodPressure + SkinThickness + Insulin + BMI + DiabetesPedigreeFunction + Age, data=Diabetes\_data, subset=train, family=binomial)  
summary(glm.fit)

##   
## Call:  
## glm(formula = Outcome ~ Pregnancies + Glucose + BloodPressure +   
## SkinThickness + Insulin + BMI + DiabetesPedigreeFunction +   
## Age, family = binomial, data = Diabetes\_data, subset = train)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.104e+01 1.844e+00 -5.985 2.17e-09 \*\*\*  
## Pregnancies -6.846e-02 7.533e-02 -0.909 0.3635   
## Glucose 3.261e-02 7.669e-03 4.253 2.11e-05 \*\*\*  
## BloodPressure 1.397e-02 1.868e-02 0.748 0.4546   
## SkinThickness 1.429e-02 2.526e-02 0.565 0.5717   
## Insulin -2.441e-04 2.089e-03 -0.117 0.9070   
## BMI 8.917e-02 4.124e-02 2.162 0.0306 \*   
## DiabetesPedigreeFunction 7.012e-01 6.649e-01 1.055 0.2916   
## Age 4.730e-02 2.594e-02 1.823 0.0683 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 243.07 on 195 degrees of freedom  
## Residual deviance: 171.41 on 187 degrees of freedom  
## AIC: 189.41  
##   
## Number of Fisher Scoring iterations: 5

# fitting again with selected important parameters  
glm.fit.1 = glm(Outcome~ Glucose + BMI + Age, data=Diabetes\_data, subset=train, family=binomial)  
summary(glm.fit.1)

##   
## Call:  
## glm(formula = Outcome ~ Glucose + BMI + Age, family = binomial,   
## data = Diabetes\_data, subset = train)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -10.082459 1.529839 -6.591 4.38e-11 \*\*\*  
## Glucose 0.033218 0.006333 5.246 1.56e-07 \*\*\*  
## BMI 0.111617 0.031846 3.505 0.000457 \*\*\*  
## Age 0.038158 0.018840 2.025 0.042830 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 243.07 on 195 degrees of freedom  
## Residual deviance: 174.06 on 192 degrees of freedom  
## AIC: 182.06  
##   
## Number of Fisher Scoring iterations: 5

# show result model coefficient  
summary(glm.fit.1)$coef

## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -10.08245946 1.529838981 -6.590536 4.382404e-11  
## Glucose 0.03321775 0.006332572 5.245539 1.558262e-07  
## BMI 0.11161699 0.031845680 3.504934 4.567211e-04  
## Age 0.03815754 0.018839859 2.025362 4.283016e-02

# predict testing data  
glm.probs = predict(glm.fit.1,Diabetes\_data[test,],type="response")  
glm.pred = rep(0,nrow(Diabetes\_data)/2)  
glm.pred[glm.probs>0.5] = 1  
  
# show result of prediction & accuracy  
table(glm.pred, Outcome[test])

##   
## glm.pred 0 1  
## 0 117 31  
## 1 10 38

mean(glm.pred==Outcome[test])

## [1] 0.7908163

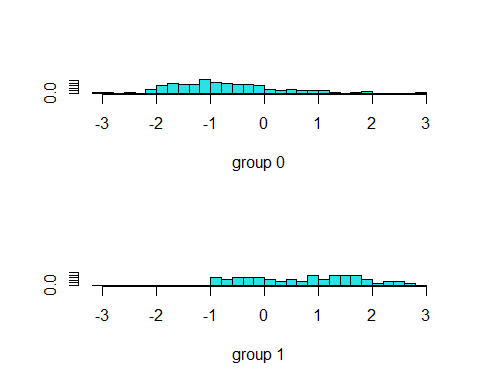
knitr::opts\_chunk$set(echo = TRUE)

### Lec3.LDA

library(MASS)  
lda1.fit = lda(Outcome ~ Pregnancies + Glucose + BloodPressure + BMI + DiabetesPedigreeFunction, data = Diabetes\_data, subset = train)  
lda1.fit

## Call:  
## lda(Outcome ~ Pregnancies + Glucose + BloodPressure + BMI + DiabetesPedigreeFunction,   
## data = Diabetes\_data, subset = train)  
##   
## Prior probabilities of groups:  
## 0 1   
## 0.6887755 0.3112245   
##   
## Group means:  
## Pregnancies Glucose BloodPressure BMI DiabetesPedigreeFunction  
## 0 3.066667 112.4296 69.34815 31.49556 0.4506741  
## 1 4.163934 147.6066 75.54098 36.00820 0.5709180  
##   
## Coefficients of linear discriminants:  
## LD1  
## Pregnancies 0.01984221  
## Glucose 0.02788976  
## BloodPressure 0.01419150  
## BMI 0.06666257  
## DiabetesPedigreeFunction 0.50711705

plot(lda1.fit)



lda1.pred = predict(lda1.fit,Diabetes\_data[test,])  
lda1.class = lda1.pred$class  
  
# show result of prediction & accuracy   
table(lda1.class, Outcome[test])

##   
## lda1.class 0 1  
## 0 117 32  
## 1 10 37

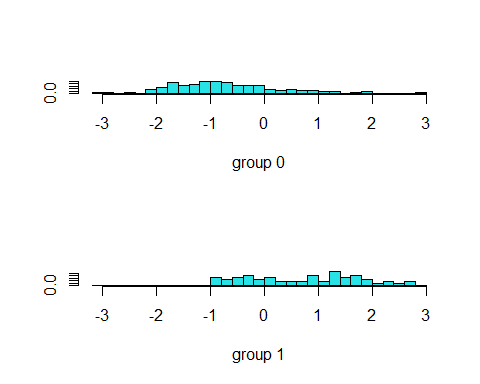
mean(lda1.class == Outcome[test])

## [1] 0.7857143

lda2.fit = lda(Outcome ~ Pregnancies + Glucose + BloodPressure + BMI + DiabetesPedigreeFunction + Insulin, data = Diabetes\_data, subset = train)  
lda2.fit

## Call:  
## lda(Outcome ~ Pregnancies + Glucose + BloodPressure + BMI + DiabetesPedigreeFunction +   
## Insulin, data = Diabetes\_data, subset = train)  
##   
## Prior probabilities of groups:  
## 0 1   
## 0.6887755 0.3112245   
##   
## Group means:  
## Pregnancies Glucose BloodPressure BMI DiabetesPedigreeFunction Insulin  
## 0 3.066667 112.4296 69.34815 31.49556 0.4506741 128.5111  
## 1 4.163934 147.6066 75.54098 36.00820 0.5709180 213.0164  
##   
## Coefficients of linear discriminants:  
## LD1  
## Pregnancies 0.0192794919  
## Glucose 0.0283883702  
## BloodPressure 0.0140799340  
## BMI 0.0672568521  
## DiabetesPedigreeFunction 0.5198510234  
## Insulin -0.0002382696

plot(lda2.fit)



lda2.pred = predict(lda2.fit,Diabetes\_data[test,])  
lda2.class = lda2.pred$class  
  
# show result of prediction & accuracy  
table(lda2.class, Outcome[test])

##   
## lda2.class 0 1  
## 0 117 33  
## 1 10 36

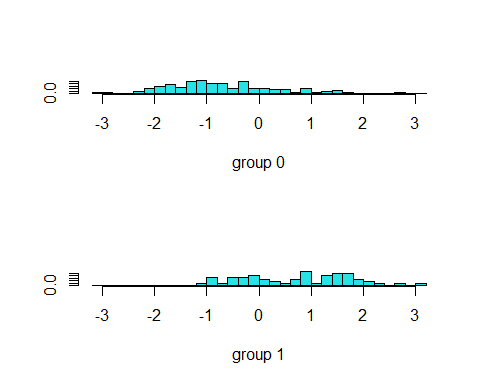
mean(lda2.class == Outcome[test])

## [1] 0.7806122

lda3.fit = lda(Outcome ~ Age + Glucose + BMI + DiabetesPedigreeFunction, data = Diabetes\_data, subset = train)  
lda3.fit

## Call:  
## lda(Outcome ~ Age + Glucose + BMI + DiabetesPedigreeFunction,   
## data = Diabetes\_data, subset = train)  
##   
## Prior probabilities of groups:  
## 0 1   
## 0.6887755 0.3112245   
##   
## Group means:  
## Age Glucose BMI DiabetesPedigreeFunction  
## 0 28.97037 112.4296 31.49556 0.4506741  
## 1 35.13115 147.6066 36.00820 0.5709180  
##   
## Coefficients of linear discriminants:  
## LD1  
## Age 0.02786757  
## Glucose 0.02601238  
## BMI 0.07485925  
## DiabetesPedigreeFunction 0.46170142

plot(lda3.fit)



lda3.pred = predict(lda3.fit,Diabetes\_data[test,])  
lda3.class = lda3.pred$class  
  
# show result of prediction & accuracy  
table(lda3.class, Outcome[test])

##   
## lda3.class 0 1  
## 0 115 31  
## 1 12 38

mean(lda3.class == Outcome[test])

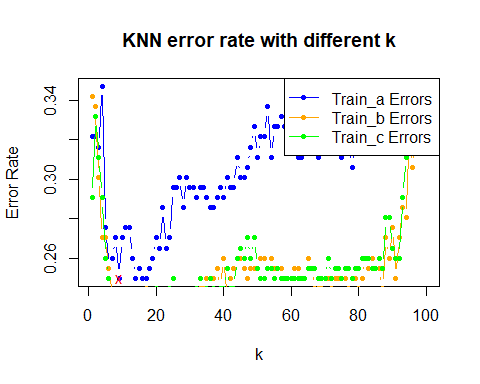
## [1] 0.7806122

knitr::opts\_chunk$set(echo = TRUE)

### Lec4.use k-nearest neighbor(KNN)

#### 跑迴圈k = 1~100，找出最佳的k

# import function  
library(class)  
  
# 設定訓練資料和測試資料 (a:全部的參數 / b:重要的參數 / c:再刪除BMI)  
train\_a = Diabetes\_data[train,-9]   
test\_a = Diabetes\_data[test,-9]  
  
train\_b = cbind(Glucose, BMI, DiabetesPedigreeFunction, Age)[train,]   
test\_b = cbind(Glucose, BMI, DiabetesPedigreeFunction, Age)[test,]  
  
train\_c = cbind(Glucose, DiabetesPedigreeFunction, Age)[train,]   
test\_c = cbind(Glucose, DiabetesPedigreeFunction, Age)[test,]  
   
# 初始化   
train\_a.err = rep(0,100)   
train\_b.err = rep(0,100)   
train\_c.err = rep(0,100)   
little\_train\_a = 1   
little\_train\_b = 1   
little\_train\_c = 1   
best\_train\_a\_k = 0   
best\_train\_b\_k = 0   
best\_train\_c\_k = 0   
  
# 計算預測及誤差，還有找出最佳k   
for(x in 1:100){   
 set.seed(1)  
 train\_a.pred = knn(train\_a , test\_a , Outcome[train] , k=x)   
 train\_b.pred = knn(train\_b , test\_b , Outcome[train] , k=x)   
 train\_c.pred = knn(train\_c , test\_c , Outcome[train] , k=x)   
 train\_a.err[x] = mean(train\_a.pred != Outcome[test])   
 train\_b.err[x] = mean(train\_b.pred != Outcome[test])   
 train\_c.err[x] = mean(train\_c.pred != Outcome[test])   
 if(little\_train\_a>train\_a.err[x]){   
 little\_train\_a = train\_a.err[x]   
 best\_train\_a\_k = x   
 }   
 if(little\_train\_b>train\_b.err[x]){   
 little\_train\_b = train\_b.err[x]   
 best\_train\_b\_k = x   
 }  
 if(little\_train\_c>train\_c.err[x]){   
 little\_train\_c = train\_c.err[x]   
 best\_train\_c\_k = x   
 }  
}   
# 畫圖   
plot(1:100 , train\_a.err , type="b" , col="blue" ,main="KNN error rate with different k" , xlab="k" , ylab="Error Rate" , pch=20)   
points(1:100 , train\_b.err , type="b" , col="orange" , pch=20)   
points(1:100 , train\_c.err , type="b" , col="green" , pch=20)   
points(best\_train\_a\_k, little\_train\_a , type="b" , col="red" , pch="x")   
points(best\_train\_b\_k , little\_train\_b , type="b" , col="red" , pch="x")   
points(best\_train\_c\_k , little\_train\_c , type="b" , col="red" , pch="x")   
legend("topright" , legend = c("Train\_a Errors","Train\_b Errors","Train\_c Errors") , col=c("blue","orange","green") , pch=20 , lty=1)



# 輸出最佳的train k和test k，以及其所對應的正確率   
c(best\_train\_a\_k,best\_train\_b\_k,best\_train\_c\_k)

## [1] 9 10 11

c(round(1-little\_train\_a,digits=5),round(1-little\_train\_b,digits=5),round(1-little\_train\_c,digits=5))

## [1] 0.75000 0.78571 0.80102

knitr::opts\_chunk$set(echo = TRUE)

### Lec5.PCA

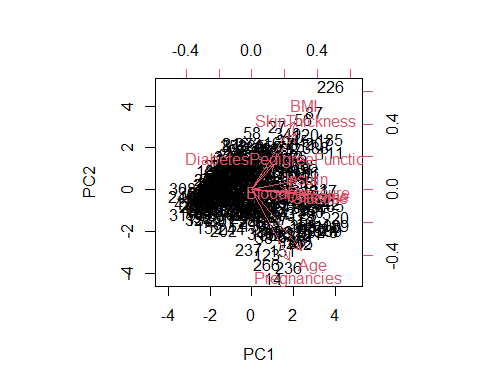
pr.out=prcomp(Diabetes\_data,scale=TRUE)  
names(pr.out)

## [1] "sdev" "rotation" "center" "scale" "x"

pr.out$rotation

## PC1 PC2 PC3 PC4  
## Pregnancies 0.2861619 -0.54048074 0.2600234 -0.18854762  
## Glucose 0.4174054 -0.04390767 -0.4194181 0.25906169  
## BloodPressure 0.2822264 -0.01607912 0.4345492 0.29956591  
## SkinThickness 0.3333287 0.42666496 0.3323389 -0.08920747  
## Insulin 0.3375709 0.07236187 -0.4916387 0.31711763  
## BMI 0.3234157 0.51692606 0.2817824 0.03002165  
## DiabetesPedigreeFunction 0.1672021 0.18723663 -0.2575468 -0.81903172  
## Age 0.3773315 -0.46387809 0.1739967 -0.13893589  
## Outcome 0.4052615 -0.03101504 -0.1986934 -0.08976693  
## PC5 PC6 PC7 PC8  
## Pregnancies 0.24955642 0.11832785 -0.09735567 -0.21736288  
## Glucose -0.06139077 -0.06728382 0.69533182 -0.29050231  
## BloodPressure -0.77534260 0.07799010 -0.02984687 0.14738893  
## SkinThickness 0.38349232 0.06377869 0.32248505 0.56208037  
## Insulin 0.09000243 0.50275840 -0.47226034 0.21963675  
## BMI 0.14166125 0.01500220 -0.25337161 -0.66619419  
## DiabetesPedigreeFunction -0.37873608 0.22789381 0.03650428 -0.01968351  
## Age 0.11553573 0.16504285 0.02644896 0.06421974  
## Outcome -0.02779195 -0.79947196 -0.33598956 0.18476289  
## PC9  
## Pregnancies 0.62435959  
## Glucose 0.06816477  
## BloodPressure 0.10800026  
## SkinThickness 0.13182123  
## Insulin 0.08000558  
## BMI -0.13993679  
## DiabetesPedigreeFunction 0.05240235  
## Age -0.73989772  
## Outcome 0.01760479

biplot(pr.out, scale=0)



pr.out$scale

## Pregnancies Glucose BloodPressure   
## 3.2114245 30.8607806 12.4960916   
## SkinThickness Insulin BMI   
## 10.5164239 118.8416898 7.0276592   
## DiabetesPedigreeFunction Age Outcome   
## 0.3454880 10.2007765 0.4714014

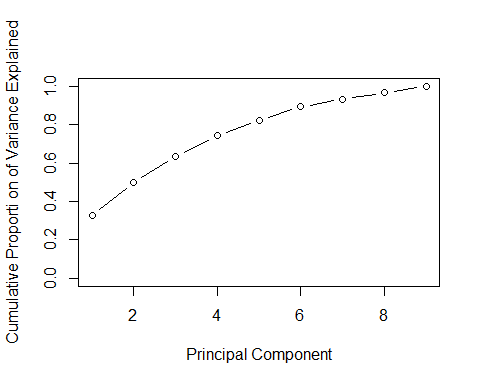
pr.out$sdev

## [1] 1.7136649 1.2481719 1.1091720 0.9796174 0.8486866 0.7996861 0.5963075  
## [8] 0.5510200 0.5445494

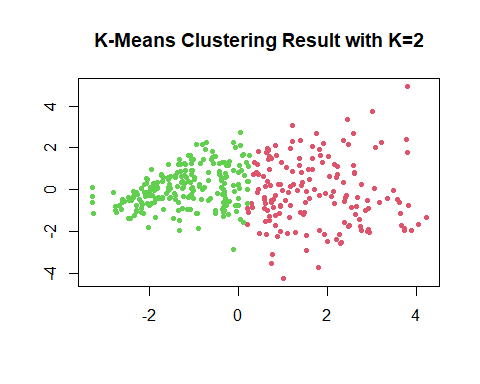
pr.var = pr.out$sdev^2  
pve = pr.var/sum(pr.var)  
pve

## [1] 0.32629417 0.17310368 0.13669583 0.10662782 0.08002989 0.07105532 0.03950918  
## [8] 0.03373589 0.03294822

plot(cumsum(pve), xlab="Principal Component", ylab="Cumulative Proporti on of Variance Explained", ylim=c(0,1),type='b')



# 將PC1,PC2 作為要被分群的資料  
train\_pca = pr.out$x[,1:3]  
km.out = kmeans(train\_pca,2,nstart=30)  
plot(train\_pca , col=(km.out$cluster+1) , main="K-Means Clustering Result with K=2" , xlab="" , ylab="" , pch=20 , cex=1)



result = rep(1,length(km.out$cluster))  
result[km.out$cluster==2] = 0  
  
# show the result and accuracy  
table(result, t(Diabetes\_data["Outcome"]))

##   
## result 0 1  
## 0 214 15  
## 1 48 115

mean(result == t(Diabetes\_data["Outcome"]))

## [1] 0.8392857

knitr::opts\_chunk$set(echo = TRUE)

### Lec11.Dicision tree (Classification tree)

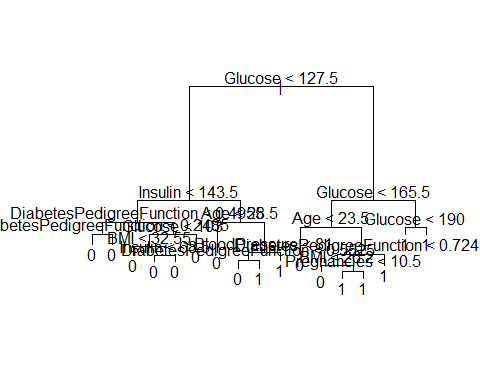
library(tree)

## Warning: 套件 'tree' 是用 R 版本 4.4.2 來建造的

Diabetes\_data$Outcome = as.factor(Diabetes\_data$Outcome)  
tree.d = tree(Outcome~.-Outcome, Diabetes\_data)  
summary(tree.d)

##   
## Classification tree:  
## tree(formula = Outcome ~ . - Outcome, data = Diabetes\_data)  
## Variables actually used in tree construction:  
## [1] "Glucose" "Insulin"   
## [3] "DiabetesPedigreeFunction" "BMI"   
## [5] "Age" "BloodPressure"   
## [7] "Pregnancies"   
## Number of terminal nodes: 17   
## Residual mean deviance: 0.6294 = 236 / 375   
## Misclassification error rate: 0.148 = 58 / 392

plot(tree.d)  
text(tree.d,pretty=0)



knitr::opts\_chunk$set(echo = TRUE)

### Lec11.Random Forest

# import functions  
library(randomForest)

## Warning: 套件 'randomForest' 是用 R 版本 4.4.2 來建造的

## randomForest 4.7-1.2

## Type rfNews() to see new features/changes/bug fixes.

Diabetes\_data$Outcome = as.factor(Diabetes\_data$Outcome)  
  
best\_x = 0  
best\_forest = 0  
best\_forest\_pred = 0  
best\_forest\_acc = 0  
  
# build a forest with training set  
for(x in 1:8){  
 Diabete\_forest = randomForest(Outcome~., data=Diabetes\_data, subset = train, mtry=x, importance=TRUE)  
 # predict testing data  
 Diabete\_forest\_pred = predict(Diabete\_forest, newdata=Diabetes\_data[test, ])  
 accur\_forest = mean(Outcome[test]==Diabete\_forest\_pred)  
 if(accur\_forest > best\_forest\_acc){  
 best\_x = x  
 best\_forest = Diabete\_forest  
 best\_forest\_pred = Diabete\_forest\_pred  
 best\_forest\_acc = accur\_forest  
 }  
}  
best\_forest

##   
## Call:  
## randomForest(formula = Outcome ~ ., data = Diabetes\_data, mtry = x, importance = TRUE, subset = train)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 5  
##   
## OOB estimate of error rate: 22.45%  
## Confusion matrix:  
## 0 1 class.error  
## 0 120 15 0.1111111  
## 1 29 32 0.4754098

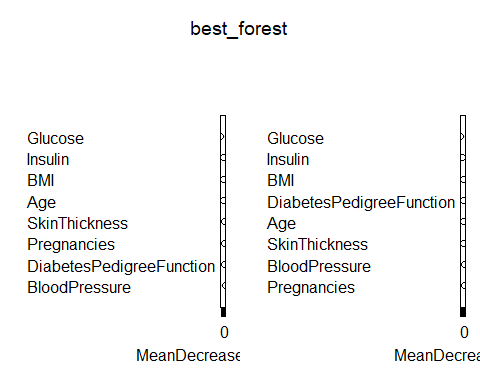
best\_x

## [1] 5

importance(best\_forest)

## 0 1 MeanDecreaseAccuracy  
## Pregnancies 5.209779 -3.9645770 2.5510697  
## Glucose 13.930817 18.0659369 22.4067909  
## BloodPressure -1.548222 2.6431990 0.2360875  
## SkinThickness 3.736586 2.9358896 4.6014575  
## Insulin 4.532332 11.7465439 10.9214549  
## BMI 6.720604 4.3479663 7.7778515  
## DiabetesPedigreeFunction 2.292963 0.5420898 1.9294765  
## Age 5.805903 2.2142898 6.2447697  
## MeanDecreaseGini  
## Pregnancies 4.543178  
## Glucose 24.617193  
## BloodPressure 5.939702  
## SkinThickness 6.961102  
## Insulin 13.453945  
## BMI 10.707598  
## DiabetesPedigreeFunction 9.548912  
## Age 7.700717

varImpPlot(best\_forest)



# show the predict result  
confusion\_matrix = table(best\_forest\_pred, Outcome[test])  
confusion\_matrix

##   
## best\_forest\_pred 0 1  
## 0 116 33  
## 1 11 36

best\_forest\_acc

## [1] 0.7755102

knitr::opts\_chunk$set(echo = TRUE)

### Lec14.SVM

# import function  
library(e1071)

## Warning: 套件 'e1071' 是用 R 版本 4.4.2 來建造的

Diabetes\_data$Outcome = as.factor(Diabetes\_data$Outcome)  
  
tune.out = tune(svm, Outcome~.,data=Diabetes\_data[train,], kernel="radial", ranges=list(cost=c(0.1,1,10,100,1000), gamma=c(0.5,1,2,3,4)))  
summary(tune.out)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 1 0.5  
##   
## - best performance: 0.2713158   
##   
## - Detailed performance results:  
## cost gamma error dispersion  
## 1 1e-01 0.5 0.3126316 0.1158944  
## 2 1e+00 0.5 0.2713158 0.1154664  
## 3 1e+01 0.5 0.3078947 0.1433453  
## 4 1e+02 0.5 0.3078947 0.1433453  
## 5 1e+03 0.5 0.3078947 0.1433453  
## 6 1e-01 1.0 0.3126316 0.1158944  
## 7 1e+00 1.0 0.3071053 0.1000389  
## 8 1e+01 1.0 0.3073684 0.1372452  
## 9 1e+02 1.0 0.3073684 0.1372452  
## 10 1e+03 1.0 0.3073684 0.1372452  
## 11 1e-01 2.0 0.3126316 0.1158944  
## 12 1e+00 2.0 0.3126316 0.1158944  
## 13 1e+01 2.0 0.3073684 0.1115367  
## 14 1e+02 2.0 0.3073684 0.1115367  
## 15 1e+03 2.0 0.3073684 0.1115367  
## 16 1e-01 3.0 0.3126316 0.1158944  
## 17 1e+00 3.0 0.3126316 0.1158944  
## 18 1e+01 3.0 0.3126316 0.1158944  
## 19 1e+02 3.0 0.3126316 0.1158944  
## 20 1e+03 3.0 0.3126316 0.1158944  
## 21 1e-01 4.0 0.3126316 0.1158944  
## 22 1e+00 4.0 0.3126316 0.1158944  
## 23 1e+01 4.0 0.3126316 0.1158944  
## 24 1e+02 4.0 0.3126316 0.1158944  
## 25 1e+03 4.0 0.3126316 0.1158944

true=Outcome[test]  
pred=predict(tune.out$best.model, newx=Diabetes\_data[test,])  
  
table(true, pred)

## pred  
## true 0 1  
## 0 96 31  
## 1 48 21

mean(pred == true)

## [1] 0.5969388

knitr::opts\_chunk$set(echo = TRUE)