**###Unit 3: 基礎分類 (監督式)**

**KNN (iris/gender)**

install.packages("class")

library(class)

data(iris)

tr.iris=rbind(iris[1:30,], iris[51:80,], iris[101:130,]) #training

ts.iris=rbind(iris[31:50,], iris[81:100,], iris[131:150,]) #testing

iris\_knn<- knn(tr.iris[,-5], ts.iris[,-5], cl=tr.iris[,5], k = 3) #只看測試集

table(ts.iris$Species, iris\_knn)

result\_knn<- table(ts.iris$Species, iris\_knn)

sum(diag(result\_knn))/sum(result\_knn) #predictive accuracy

sum( as.numeric(iris\_knn == ts.iris$Species) )/ nrow(ts.iris)

iris\_knn1<- knn(tr.iris[,-5], tr.iris[,-5], cl=tr.iris[,5], k = 3) #只看訓練集

table(tr.iris$Species, iris\_knn1)

gender\_size <-read.csv("gender\_size.csv")

data\_gender <-gender\_size

#data\_gender[,1:3] <-scale1(gender\_size)

#data\_gender[,4]<-as.factor(data\_gender[,4])

err\_knn=rep(0, 5) #針對性別資料集選取合適的k值

for (i in 1:5)

{

pre\_knn= knn(data\_gender[,-4], data\_gender[,-4], cl=data\_gender[,4], k=2\*i-1)

err\_knn[i]=sum(as.numeric (pre\_knn != data\_gender$Gender) / nrow(data\_gender))

}

err\_knn

plot(err\_knn, type="l", xlab="k", main="Selection of k=1,3,5,7,9")

gender\_knn=knn(data\_gender[,-4], data\_gender[,-4], cl=data\_gender[,4], k = 1)

table(data\_gender[,4], gender\_knn)

**KNN (cancer/glass)**

cancer\_data <-read.csv("breast\_cancer.csv", header=T, sep=",")

cancer<- cancer\_data[cancer\_data$Bare.Nuclei!= 9999, 2:11]

cancer.knn=knn(cancer[,-10], cancer[,-10], cl= cancer[,10], k = 3)

table(cancer$Class, cancer.knn)

cancer.tab= table(cancer$Class, cancer.knn)

sum(diag(cancer.tab))/sum(cancer.tab) #predictive accuracy

glass\_data<- read.table("glass.txt", header=F, sep=",") #玻璃資料集

glass<-glass\_data[,-1] #先移除資料序號

colnames(glass)=c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8", "x9", "type")

glass[,10]<- as.factor(glass$type)

glass.knn=knn(glass[,-10], glass[,-10], cl= glass[,10], k = 3)

glass.knn

glass.tab<-table(glass$type, glass.knn)

glass.tab

sum(diag(glass.tab))/sum(glass.tab) #predictive accuracy

**天真貝氏法**

**Naïve Bayes (iris)**

**data(iris)**

**attach(iris)**

**t**r.iris=rbind(iris[1:30,], iris[51:80,], iris[101:130,])

ts.iris=rbind(iris[31:50,], iris[81:100,], iris[131:150,])

install.packages("MASS")

install.packages("klaR")

library(MASS)

library(klaR)

fit\_bayes=NaiveBayes(Species ~ ., data= tr.iris)

names(fit\_bayes)

fit\_bayes$apriori

fit\_bayes$tables

pre\_bayes=predict(fit\_bayes, ts.iris)

pre\_bayes

table(ts.iris$Species, pre\_bayes$class)

**NaïveBayes (gender)**

fit\_bayes1=NaiveBayes(Gender~., data\_gender)

names(fit\_bayes1)

fit\_bayes1$apriori

fit\_bayes1$tables

pre\_bayes1=predict(fit\_bayes1, data\_gender[,-4])

table(data\_gender[,4], pre\_bayes1$class)

fit\_bayes2=NaiveBayes(data\_gender[,-4], data\_gender[,4])

pre\_bayes2=predict(fit\_bayes2, data\_gender[,-4])

table(data\_gender[,4], pre\_bayes2$class)

**\*\*\*\*\* NaïveBayes (titanic)**

titanic\_data <-read.csv("titanic.csv", header=T, sep=",")

titanic<-na.exclude(titanic\_data) #去除缺失值

attach(titanic)

rw<-c(which(age==9999), which(fare==9999)) #尋找age及fare缺失的紀錄

titan<-titanic[-rw, 2:5]

titan$survival<-titanic[-rw, 10] #設定survival類別

titan[which(titan[,1]==0), 1]<-"male" #設定男性的名稱

titan[which(titan[,1]==1), 1]<-"female" #設定女性的名稱

titan[,1]=as.factor(titan$gender)

titan[which(titan[,5]==1), 5]<-"yes"

titan[which(titan[,5]==0), 5]<-"no"

titan[,5]<-as.factor(titan[,"survival"])

attach(titan)

np<-sample(1:nrow(titan), round(0.4\*nrow(titan)) )

tr.tan<-titan[-np,] #訓練集

ts.tan<-titan[np,] #測試集

fit\_bayes3=NaiveBayes(tr.tan[,-5], tr.tan[,5])

names(fit\_bayes3)

fit\_bayes3$apriori

fit\_bayes3$tables

pre\_bayes3=predict(fit\_bayes3, ts.tan[,-5])

table(ts.tan[,5], pre\_bayes3$class)

result\_bayes<- table(ts.tan[,5], pre\_bayes3$class)

sum(diag(result\_bayes))/sum(result\_bayes) #predictive accuracy

sum( as.numeric(pre\_bayes3$class == ts.tan[,5]) )/ nrow(ts.tan)

**\*\*\*二元羅吉斯 (binary LR)**

gender\_size <-read.csv("[gender\_size.csv](file:///\\gender_size.csv)")

attach(gender\_size)

install.packages("ellipse")

library(ellipse)

var=c(1:3)

corr=cor(gender\_size[var], use="pairwise")

corr

plotcorr(corr, diag=T, type="lower")

attach(gender\_size) #未標準化的性別資料集

LR0<-glm(formula=Gender~Height+Weight+Waist, family=binomial(link="logit"), data=gender\_size)

summary(LR0) #自變數不夠獨立致使迴歸係數不顯著

x1=list(Height= 175, Weight=68, Waist=32)

x2=list(Height= 165, Weight=50, Waist=27)

predict.glm(LR0, type="response", newdata= x1)

predict.glm(LR0, type="response", newdata= x2)

pre.LR0=predict.glm(LR0, type="response", newdata=gender\_size)

result0<-ifelse (round(pre.LR0)>=0.5, "male", "female")

tab0=table(gender\_size$Gender, result0)

tab0

cat("predictive accuracy", 100\*sum(diag(tab0))/sum(tab0), "% \n")

#經標準化的性別資料集

maxd<-c(max(gender\_size[,1]), max(gender\_size[,2]), max(gender\_size[,3]))

mind<-c(min(gender\_size[,1]), min(gender\_size[,2]), min(gender\_size[,3]))

rang<-maxd-mind

data\_gender<-gender\_size #initialization

for (j in 1:(ncol(data\_gender)-1))

{

data\_gender[, j]=(gender\_size[, j]-mind[j]) / rang[j]

}

attach(data\_gender)

LR1<-glm(formula=Gender~Height+Weight+Waist, family=binomial(link="logit"), data=data\_gender)

summary(LR1)

predict.glm(LR1, type="response", newdata= data.frame (Height= (175-mind[1])/rang[1], Weight=(68-mind[2])/rang[2], Waist=(32-mind[3])/rang[3] ))

predict.glm(LR1, type="response", newdata= data.frame (Height= (165-mind[1])/rang[1], Weight=(50-mind[2])/rang[2], Waist=(27-mind[3])/rang[3] ))

pre.LR1=predict.glm(LR1, type="response", newdata=data\_gender)

result1<-ifelse(pre.LR1>0.5, "male", "female")

table(data\_gender[,4], result1)

LR2<-glm(Gender~ Height+Waist, data=gender\_size, family=binomial(link="logit"))

summary(LR2) #去掉體重

pre.LR2=predict.glm(LR2, type="response", newdata=gender\_size)

result2<-ifelse(pre.LR2>0.5, "male", "female")

table(gender\_size[, 4], result2)

**\*\*\*\*\*Probit 分類 (類似累積機率密度)**

PRO <- glm(Gender~Height+Weight+Waist, data = gender\_size, family = quasibinomial(link ="probit"))

summary(PRO)

pre.PRO= predict.glm(PRO, type="response", newdata= gender\_size)

result.PRO<-ifelse (pre.PRO>=0.5, "male", "female")

table(gender\_size[, 4], result.PRO)

**\*\*\*\*\*二元羅吉斯using Titanic**

attach(titan)

np<-sample(1: nrow(titan), round(0.4\*nrow(titan)) )

tr.tan<-titan[-np,]

ts.tan<-titan[np,]

LR<-glm(formula=survival~fare+age+class+gender, data=tr.tan, family=binomial(link="logit"), na.action=na.exclude)

summary(LR)

pre.LR= predict.glm(LR, type="response", newdata=ts.tan)

result<-ifelse (round(pre.LR)>=0.5, "yes", "no")

tab.LR<-table(ts.tan$survival, result)

tab.LR

cat("predictive accuracy", 100\*sum(diag(tab.LR))/sum(tab.LR), "% \n")

predict.glm(LR, type="response", newdata=data.frame(fare=30, age=1, class="2", gender="male"))

predict.glm(LR, type="response", newdata=data.frame(fare=30, age=40, class="2", gender="male"))

predict.glm(LR, type="response", newdata=data.frame(fare=50, age=30, class="3", gender="female"))

predict.glm(LR, type="response", newdata=data.frame(fare=50, age=30, class="1", gender="female"))

attach(titan) #指定資料集

classtab<-table(class, survival)

prop.table(classtab, 1)

gendertab<-table(gender, survival)

prop.table(gendertab, 1)

**\*\*\*\*\* 多元LR (multi-class)**

install.packages("nnet")

library(nnet)

data(iris)

tr.iris=rbind(iris[1:30,], iris[51:80,], iris[101:130,])

ts.iris=rbind(iris[31:50,], iris[81:100,], iris[131:150,])

LR.iris<-multinom(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width, data=**t**r.iris, Hess=TRUE)

s1=summary(LR.iris)

coef(s1)

exp(coef(s1))

confint(LR.iris)

x.145<- as.numeric(c(1, iris[145, -5])) #測試樣本

beta.versicolor=as.numeric(coef(s1)[1,])

ver=exp(x.145 %\*% beta.versicolor) #versicolor vs. setosa

beta.virginca= as.numeric(coef(s1)[2,])

vir=exp(x.145 %\*% beta.virginca) #virginca vs. setosa

pb.setosa=1/(1+ver+vir) #計算相對機率

pb.versicolor=ver/(1+ver+vir)

pb.virginca=vir/(1+ver+vir)

tr.pred=predict(LR.iris, **t**r.iris[, -5])

table(**t**r.iris$Species, tr.pred)

sum( as.numeric(tr.pred == **t**r.iris[, 5]) )/ nrow(**t**r.iris)

ts.pred=predict(LR.iris, ts.iris[,-5])

table(ts.iris$Species, ts.pred)

sum( as.numeric(ts.pred == ts.iris[, 5]) )/ nrow(ts.iris)

**\*\*\*\*\* CART (Gini index)/CHAID decision tree (Chi-square test)**

attach(iris)

tr.iris=rbind(iris[1:30,], iris[51:80,], iris[101:130,])

ts.iris=rbind(iris[31:50,], iris[81:100,], iris[131:150,])

install.packages("tree") #CART

library(tree)

iris.tree<-tree(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width, data=tr.iris)

plot(iris.tree)

text(iris.tree)

pr.iris=predict(iris.tree, tr.iris, type="class")

table(tr.iris$Species, pr.iris)

ps.iris=predict(iris.tree, ts.iris, type="class")

table(ts.iris$Species, ps.iris)

install.packages("party") #CHAID

library(party)

iris\_tree <- ctree(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=tr.iris)

print(iris\_tree)  
plot(iris\_tree)

table(predict(iris\_tree), tr.iris$Species) #訓練集

table(predict(iris\_tree, newdata = ts.iris), ts.iris$Species) #測試集

data\_gender<- read.csv("gender\_size.csv")

attach(data\_gender)

gender.tree=tree(Gender~Waist+Height+Weight, data=data\_gender)

plot(gender.tree)

text(gender.tree)

pred.gender <- predict(gender.tree, data\_gender[,-4], type="class")

table(true= data\_gender$Gender, pred= pred.gender)

sum( as.numeric(pred.gender == data\_gender[,4]) )/ nrow(data\_gender)

##**(C4.5之J48需整合Java介面??)**

install.packages("rJava")

library(rJava)

install.packages("RWeka")

library(RWeka)

attach(iris)

iris.c45<-J48(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,tr.iris)

summary(iris.c45)

plot(iris.c45)

tr.tree=predict(iris.c45, tr.iris, type="class")

table(tr.iris$Species, tr.tree)

sum( as.numeric(tr.tree == tr.iris[,5]) )/ nrow(tr.iris)

ts.tree=predict(iris.c45, ts.iris, type="class")

table(ts.iris$Species, ts.tree)

sum( as.numeric(ts.tree== ts.iris[,5]) )/ nrow(ts.iris)

**\*\*\*\*CART (redwine) #classification and regression tree (Gini Index)**

install.packages("rpart.plot")

library(rpart.plot)

install.packages("rpart")

library(rpart)

redwine=read.csv("[winequality-red.csv](file:///\\winequality-red.csv)") #紅酒資料集

hist(redwine$quality)

char=0

for (i in 1:nrow(redwine))

{

if (redwine [i,12]>6) char[i]="excellent"

else if(redwine [i,12]==6) char[i]="good"

else if(redwine [i,12]==5) char[i]="medium"

else char[i]="bad"

}

redwine [, 12]=factor(char)

summary(redwine$quality)

colnames(redwine)=c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8", "x9", "x10", "x11", "quality")

red.id=sample(nrow(redwine), 1000)

tr.red= redwine[red.id, ]

ts.red= redwine[-red.id, ]

attach(redwine)

red.DT= quality~x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11

rp.red=rpart(red.DT, tr.red, method="class", minsplit=10, minbucket=10)

#minsplit=10 (最少包含樣本數) #maxdepth=3 (最大深度)

print(rp.red)

summary(rp.red)

rpart.plot(rp.red, fallen.leaves=TRUE)

catr.red=predict(rp.red, tr.red, type="class") #training set

table(tr.red$quality, catr.red)

sum( as.numeric(catr.red == tr.red[,12]) )/ nrow(tr.red)

cats.red=predict(rp.red, ts.red, type="class") #testing set

table(ts.red$quality, cats.red)

sum( as.numeric(cats.red == ts.red[,12]) )/ nrow(ts.red)

**\*\*\*CART (whitewine)**

wine=read.csv("[winequality-white.csv](file:///\\winequality-white.csv)")

hist(wine$quality)

char=0

for (i in 1:nrow(wine))

{

if (wine[i,12]>6) char[i]="good"

else if(wine[i,12]==6) char[i]="medium"

else char[i]="bad"

}

wine[, 12]=factor(char)

summary(wine$quality)

colnames(wine)=c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8", "x9", "x10", "x11", "quality")

white.id=sample(nrow(wine), 3000)

tr.white=wine[white.id, ]

ts.white=wine[-white.id, ]

white.DT=quality~x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11

rp.white= rpart(white.DT, tr.white, method="class", minsplit=30, minbucket=30)

print(rp.white)

summary(rp.white)

rpart.plot(rp.white, type=4, fallen.leaves=TRUE)

catr.white=predict(rp.white, tr.white, type="class") #training set

table(tr.white$quality, catr.white)

sum( as.numeric(catr.white == tr.white[,12]) )/ nrow(tr.white)

cats.white=predict(rp.white, ts.white, type="class") #testing set

table(ts.white$quality, cats.white)

sum( as.numeric(cats.white == ts.white[,12]) )/ nrow(ts.white)

**\*\*\*\*\*C5.0 (titanic)**

install.packages("C50")

library(C50)

attach(titan)

np<-sample(1: nrow(titan), round(0.4\*nrow(titan)) )

tr.tan<-titan[-np,]

ts.tan<-titan[np,]

DT=C5.0(tr.tan[,-5], tr.tan[,5], rules=FALSE) #minCases=10?

summary(DT)

plot(DT, main="titanic")

tr.DT<-predict(DT, tr.tan, type="class")

table(tr.tan$survival, tr.DT)

sum( as.numeric(tr.DT == tr.tan[,5]) )/ nrow(tr.tan)

ts.DT<-predict(DT, ts.tan, type="class")

table(ts.tan$survival, ts.DT)

sum( as.numeric(ts.DT == ts.tan[,5]) )/ nrow(ts.tan)

**\*\*\*\*\*C5.0 (TeleComm\_churn)**

churn\_data<-read.table("churn.txt", header=T, sep=",")

churn\_data<-churn\_data[,-1] #去掉顧客ID, x1: time spent on long-distance call per month, x2: international calls, x3: local calls, x4: number of dropped calls, x5: payment method for the bill, x6: tariff for local calls, x7: tariff for long-distance calls, x8: age, x9: gender, x10: martial status, x11: # of children, x12: income, x13: car owner, x14: churn labels// Current (not churned), Vol: valuable churner, InVol: leavers (don’t care)

colnames(churn\_data)=c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8", "x9", "x10", "x11", "x12", "x13", "Y")

churn.id<-sample(1: nrow(churn\_data), round(0.4\*nrow(churn\_data)) )

churn.tr<- churn\_data[-churn.id,]

churn.ts<- churn\_data[churn.id,]

churn.C50<- C5.0(churn.tr[,-14], churn.tr[,14], CF=0.25, rules=FALSE, minCases=20, earlyStopping = TRUE)

summary(churn.C50) #檢查變數重要性

plot(churn.C50, main="churn")

tr.C50<- predict(churn.C50, churn.tr, type="class")

table(churn.tr$Y, tr.C50)

sum( as.numeric(tr.C50 == churn.tr[, 14]) )/ nrow(churn.tr)

ts.C50<- predict(churn.C50, churn.ts, type="class")

table(churn.ts$Y, ts.C50)

sum( as.numeric(ts.C50 == churn.ts[, 14]) )/ nrow(churn.ts)

churn.new<-churn\_data[,c(2,3,8,9,12,14)] #remove redundant variables

churn.tr1<- churn.new[-churn.id,]

churn.ts1<- churn.new[churn.id,]

churn.C51<- C5.0(churn.tr1[,-6], churn.tr1[,6], CF=0.5, rules=TRUE, minCases=50, earlyStopping = TRUE)

summary(churn.C51)

tr.C51<- predict(churn.C51, churn.tr1, type="class")

table(churn.tr1$Y, tr.C51)

sum( as.numeric(tr.C51 == churn.tr1[,"Y"]) )/ nrow(churn.tr1)

ts.C51<- predict(churn.C51, churn.ts1, type="class")

table(churn.ts1$Y, ts.C51)

sum( as.numeric(ts.C51 == churn.ts1[,"Y"]) )/ nrow(churn.ts1)

\*\*\*\*\*\*\*CART/Bank資料集 (不分訓練集/測試集)

bank=read.csv("bank.csv", header=TRUE, sep=";")

summary(bank)

colnames(bank)=c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8", "x9", "x10", "x11", "x12", "x13", "x14", "x15", "x16", "y")

bank.DT=y~x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16

rp.bank= rpart(bank.DT, bank, method="class", minsplit=10, minbucket=10)

print(rp.bank)

summary(rp.bank)

rpart.plot(rp.bank, type=4, fallen.leaves=TRUE)

cart.bank=predict(rp.bank, bank, type="class")

table(bank$y, cart.bank)

sum( as.numeric(cart.bank == bank[,"y"]) )/ nrow(bank)

bank1<-bank[,c(1,2,3,6,10,11,12,14,16,17)] #remove redundant variables

bank.DT1=y~x1+x2+x3+x6+x10+x11+x12+x14+x16

rp.bank1= rpart(bank.DT1, bank1, method="class", minsplit=10, minbucket=10)

print(rp.bank1)

summary(rp.bank1)

rpart.plot(rp.bank1, type=4, fallen.leaves=FALSE)

cart.bank1=predict(rp.bank1, bank1, type="class")

table(bank1$y, cart.bank1)

sum( as.numeric(cart.bank1 == bank1[,"y"]) )/ nrow(bank1)