機器學習期末報告

Pima 印地安人糖尿病

HW16_M0928001_黃瑄惠

指導教授: 魏裕中老師

中華民國一百零九年六月十六日

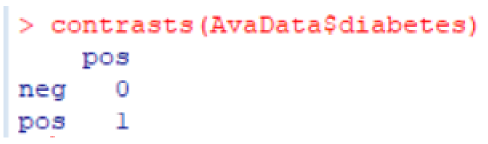
壹、前言

此數據最初來自美國糖尿病、消化與腎臟疾病研究所。數據集目的是基於數據中包含的某些診斷指標，預測患者是否患有糖尿病，數據內的所有患者皆為 Pima 印地安人血統 21 歲以上的女性。將是否患有糖尿病設為 response，此數據共含 768 個實例，1 個目標變數和 8 個反應變數。

貳、變數說明

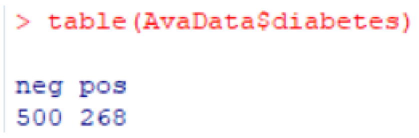
變數名稱	說明
pregnant	懷孕次數
glucose	葡萄糖，口服葡萄糖耐量測試 2 小時的血漿葡萄糖濃度
pressure	血壓，舒張壓（毫米汞柱）
triceps	皮膚厚度，三頭肌皮膚褶皺厚度（毫米）
insulin	胰島素，2 小時血清胰島素（mu U / ml）
mass	體重指數（體重（kg）/（身高（m））^2）
pedigree	糖尿病譜系函數
age	年齡（歲）
diabetes	768 個類別變量（0 或 1）中，268 個為 1，其他為 0

參、Diagnostics



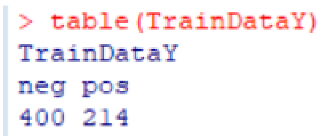
```
> contrasts(AvaData$diabetes)
    pos
neg   0
pos   1
```

目標變數 Y : neg、pos

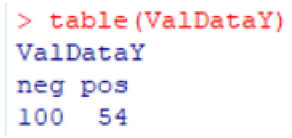


```
> table(AvaData$diabetes)

neg pos
500 268
```

neg : 500 筆　pos : 268 筆



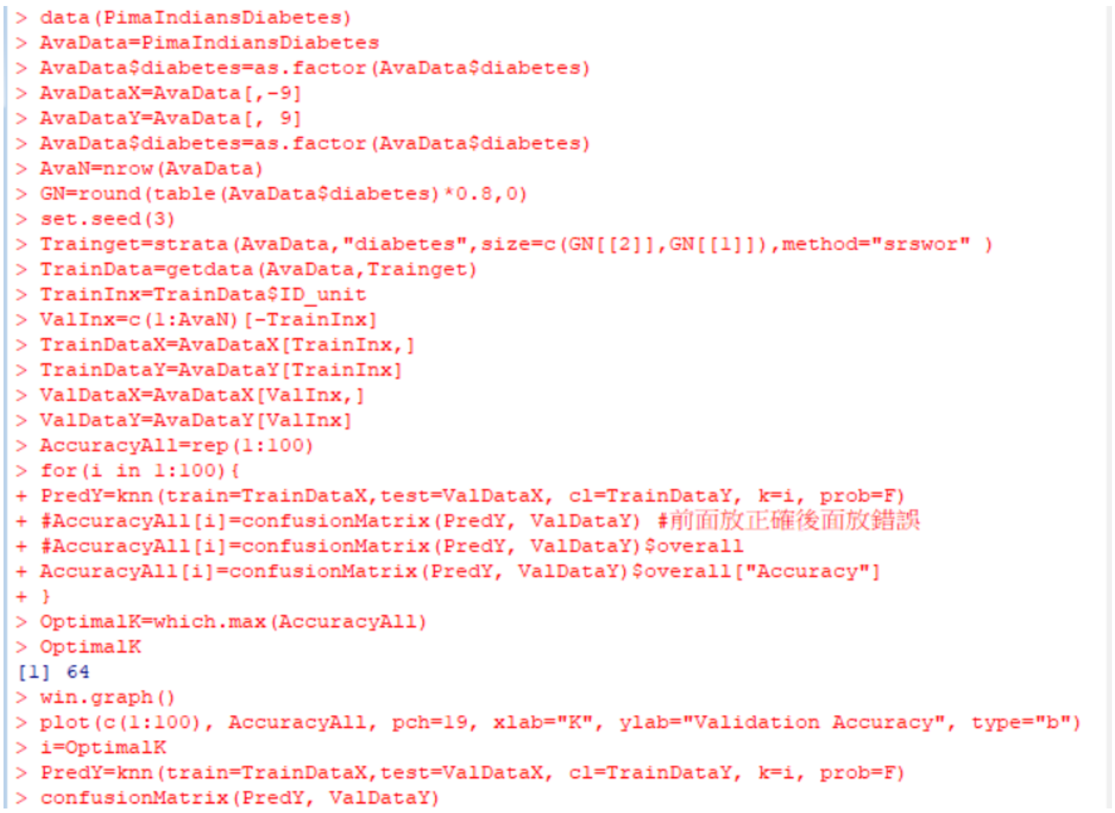
```
> table(TrainDataY)
TrainDataY
neg pos
400 214
```



```
> table(ValDataY)
ValDataY
neg pos
100  54
```

為求實驗準確性，資料集依比例將 80%設為 Training set、20%設為 Validation set 並套用到所有 modle 上

1. KNN



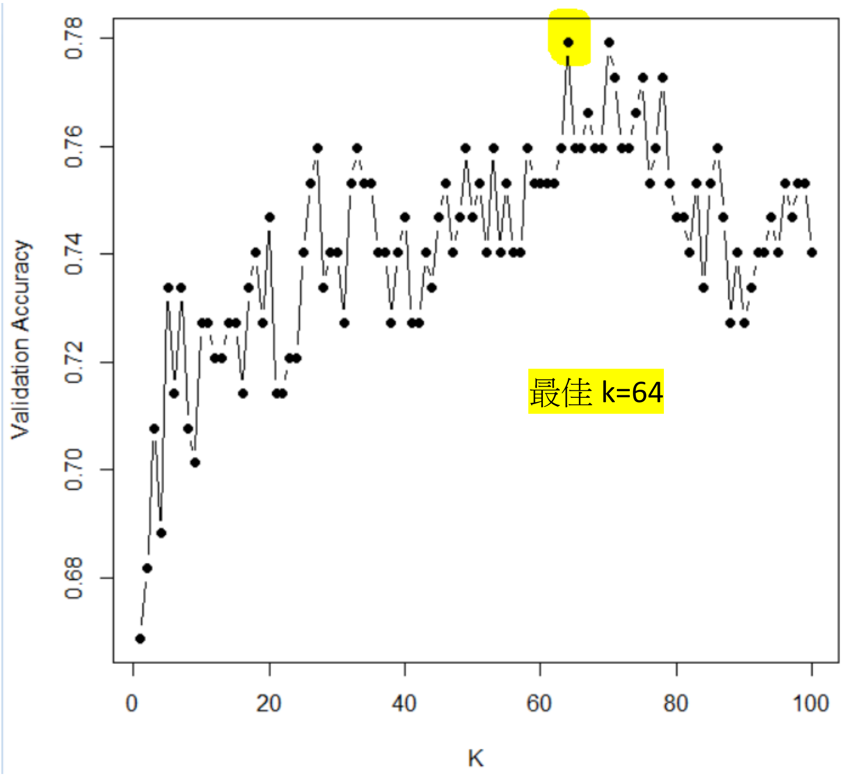
```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaDataX=AvaData[,-9]
> AvaDataY=AvaData[, 9]
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)
> set.seed(3)
> Trainget=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]),method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainDataX=AvaDataX[TrainInx,]
> TrainDataY=AvaDataY[TrainInx]
> ValDataX=AvaDataX[ValInx,]
> ValDataY=AvaDataY[ValInx]
> AccuracyAll=rep(1:100)
> for(i in 1:100){
+ PredY=knn(train=TrainDataX,test=ValDataX, cl=TrainDataY, k=i, prob=F)
+ #AccuracyAll[i]=confusionMatrix(PredY, ValDataY) #前面放正確後面放錯誤
+ #AccuracyAll[i]=confusionMatrix(PredY, ValDataY)$overall
+ AccuracyAll[i]=confusionMatrix(PredY, ValDataY)$overall["Accuracy"]
+ }
> OptimalK=which.max(AccuracyAll)
> OptimalK
[1] 64
> win.graph()
> plot(c(1:100), AccuracyAll, pch=19, xlab="K", ylab="Validation Accuracy", type="b")
> i=OptimalK
> PredY=knn(train=TrainDataX,test=ValDataX, cl=TrainDataY, k=i, prob=F)
> confusionMatrix(PredY, ValDataY)
```

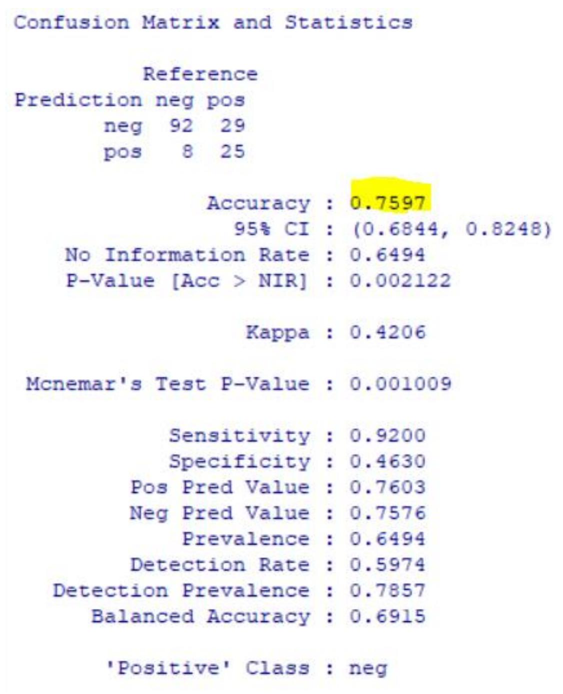
設定 tuning parameter 為 k=1,2,⋯,100

以 training set 建立 knn 分類模型

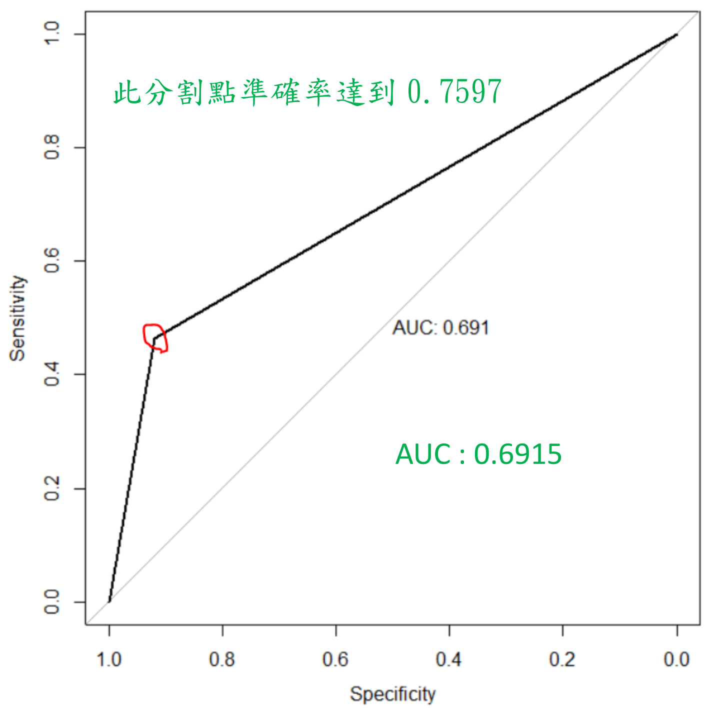
計算出所有準確率，挑出準確率最大時的 k

最後列出 confution matrix





```
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  92  29
       pos   8  25

               Accuracy : 0.7597
                 95% CI : (0.6844, 0.8248)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 0.002122

                  Kappa : 0.4206

 Mcnemar's Test P-Value : 0.001009

            Sensitivity : 0.9200
            Specificity : 0.4630
         Pos Pred Value : 0.7603
         Neg Pred Value : 0.7576
             Prevalence : 0.6494
         Detection Rate : 0.5974
   Detection Prevalence : 0.7857
      Balanced Accuracy : 0.6915

       'Positive' Class : neg
```



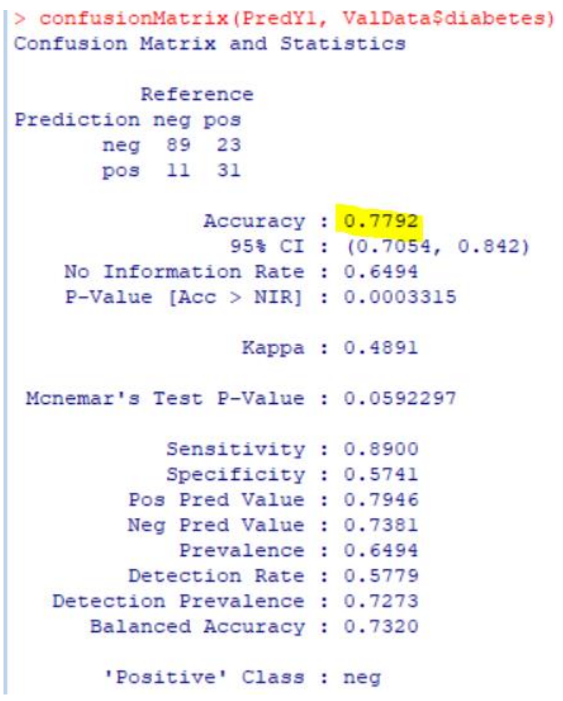
```
> win.graph()
> plot.roc(as.numeric(ValDataY),as.numeric(PredY), print.auc=TRUE)
```

	實際 neg (+	實際 pos (-	總計
預測 neg	92	29	121
預測 pos	8	25	33
總計	100	54	154
	Se(敏感度) =92/100 =0.92	Sp(特異度) =25/54 =0.463	準確率:117/154=0.7597

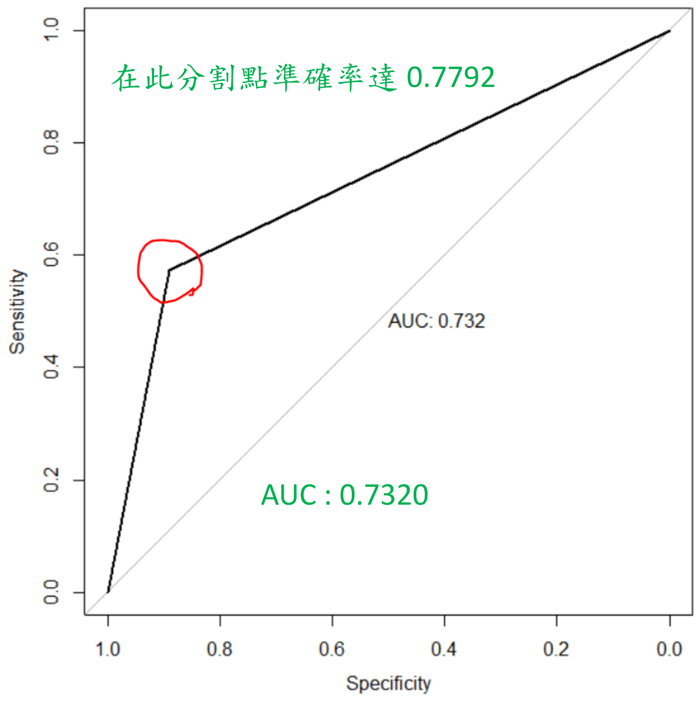
2. Logistic regression

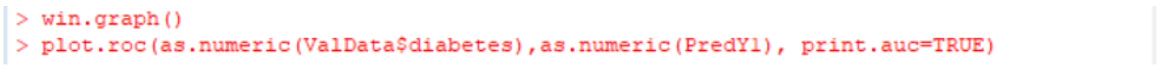
```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)
> set.seed(3)
> Traingeт=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]),method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainData=select(TrainData,-c("ID_unit","Prob","Stratum"))
> ValData=AvaData[ValInx,]
> ModelLog1=glm(formula=diabetes~.,family=binomial,data=TrainData)
> PreProb1=predict(ModelLog1, newdata=ValData[,-9],type="response")
> PredY1=as.factor(ifelse(PreProb1>0.5, "pos", "neg"))
> confusionMatrix(PredY1, ValData$diabetes)
```

Training set 帶入建構 Logistic regression，求出預測值並計算準確率



```
> confusionMatrix(PredY1, ValData$diabetes)
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  89  23
       pos  11  31

               Accuracy : 0.7792
                 95% CI : (0.7054, 0.842)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 0.0003315

                  Kappa : 0.4891

 Mcnemar's Test P-Value : 0.0592297

            Sensitivity : 0.8900
            Specificity : 0.5741
         Pos Pred Value : 0.7946
         Neg Pred Value : 0.7381
             Prevalence : 0.6494
         Detection Rate : 0.5779
   Detection Prevalence : 0.7273
      Balanced Accuracy : 0.7320

       'Positive' Class : neg
```

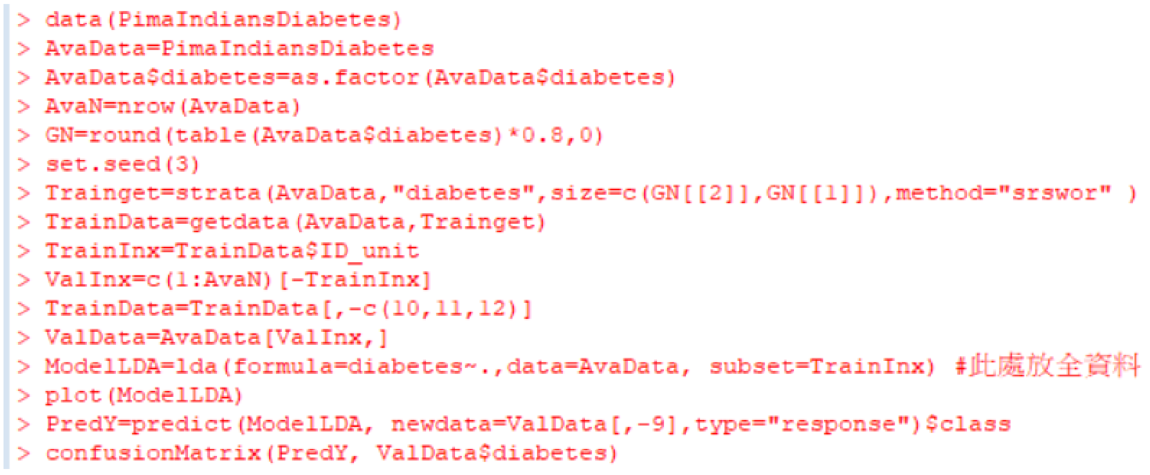




```
> win.graph()
> plot.roc(as.numeric(ValData$diabetes),as.numeric(PredY1), print.auc=TRUE)
```

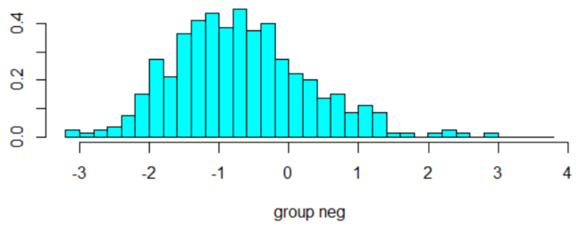
	實際 neg (+	實際 pos (-	總計
預測 neg	89	23	112
預測 pos	11	31	42
總計	100	54	154
	Se(敏感度) =89/100 =0.89	Sp(特異度) =31/54 =0.6494	準確率:120/154=0.7792

3. LDA

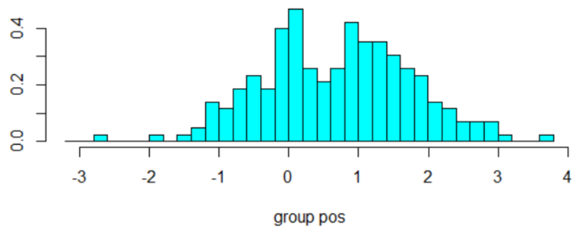


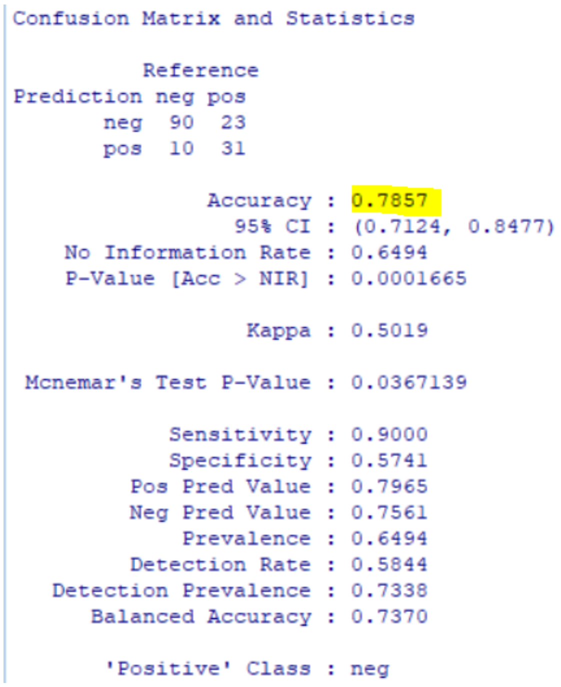
```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)
> set.seed(3)
> Trainget=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]),method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainData=TrainData[,-c(10,11,12)]
> ValData=AvaData[ValInx,]
> ModelLDA=lda(formula=diabetes~.,data=AvaData, subset=TrainInx) #此處放全資料
> plot(ModelLDA)
> PredY=predict(ModelLDA, newdata=ValData[,-9],type="response")$class
> confusionMatrix(PredY, ValData$diabetes)
```

以 training set 建立 LDA 模型並以 validation set 的代入建構的 LDA model，得到 Y 的預測情況並計算準確率

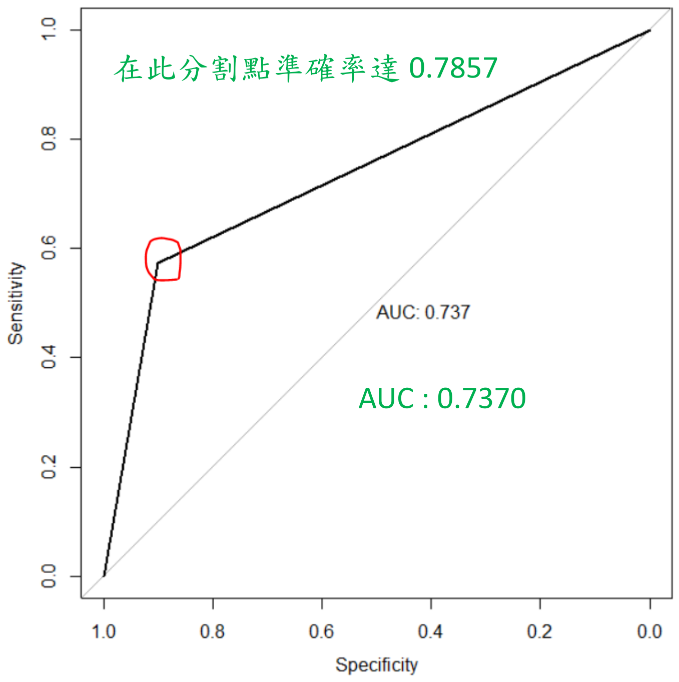


Y 變數 neg、pos 接近呈常態
兩者分散程度差異不大
適合用 Logistic regression





```
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  90  23
       pos  10  31

               Accuracy : 0.7857
                 95% CI : (0.7124, 0.8477)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 0.0001665

                  Kappa : 0.5019

 Mcnemar's Test P-Value : 0.0367139

            Sensitivity : 0.9000
            Specificity : 0.5741
         Pos Pred Value : 0.7965
         Neg Pred Value : 0.7561
             Prevalence : 0.6494
         Detection Rate : 0.5844
   Detection Prevalence : 0.7338
      Balanced Accuracy : 0.7370

       'Positive' Class : neg
```



```
> win.graph()
> plot.roc(as.numeric(ValData$diabetes),as.numeric(PredY),print.auc=TRUE)
```

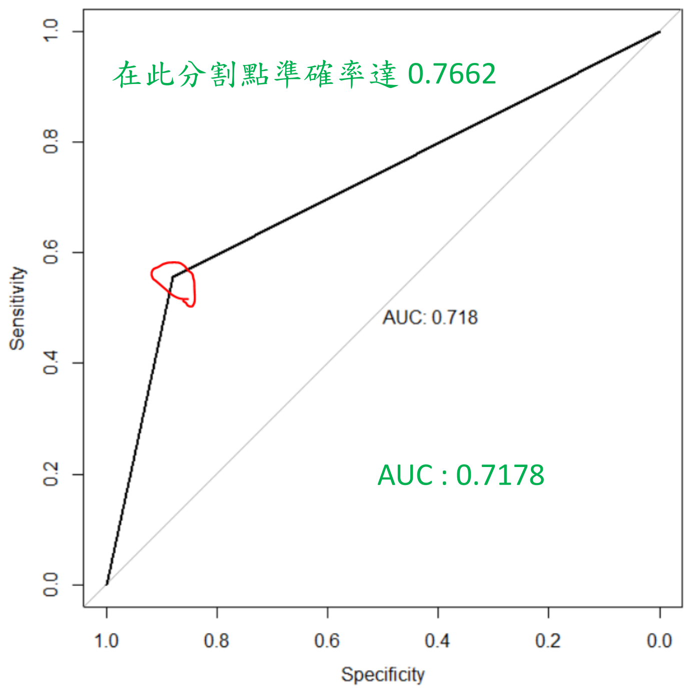
	實際 neg (+	實際 pos (-	總計
預測 neg	90	23	113
預測 pos	10	31	41
總計	100	54	154
	Se(敏感度) =90/100 =0.9	Sp(特異度) =31/54 =0.5741	正確率:121/154=0.7857

4. QDA

```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)
> set.seed(3)
> Trainget=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]), method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainData=TrainData[,-c(10,11,12)]
> ValData=AvaData[ValInx,]
> ModelQDA=qda(formula=diabetes~.,data=AvaData, subset=TrainInx)
> PredY=predict(ModelQDA, newdata=ValData[,-9],type="response")$class
> confusionMatrix(PredY, ValData$diabetes)
```

以 training set 建構 QDA model，以 validation set 代入建構的 QDA model 得到 Y 的預測情況並計算出準確率

```
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  88  24
       pos  12  30

               Accuracy : 0.7662
                 95% CI : (0.6914, 0.8306)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 0.001182

                  Kappa : 0.459

 Mcnemar's Test P-Value : 0.066753

            Sensitivity : 0.8800
            Specificity : 0.5556
         Pos Pred Value : 0.7857
         Neg Pred Value : 0.7143
             Prevalence : 0.6494
         Detection Rate : 0.5714
   Detection Prevalence : 0.7273
      Balanced Accuracy : 0.7178

       'Positive' Class : neg
```



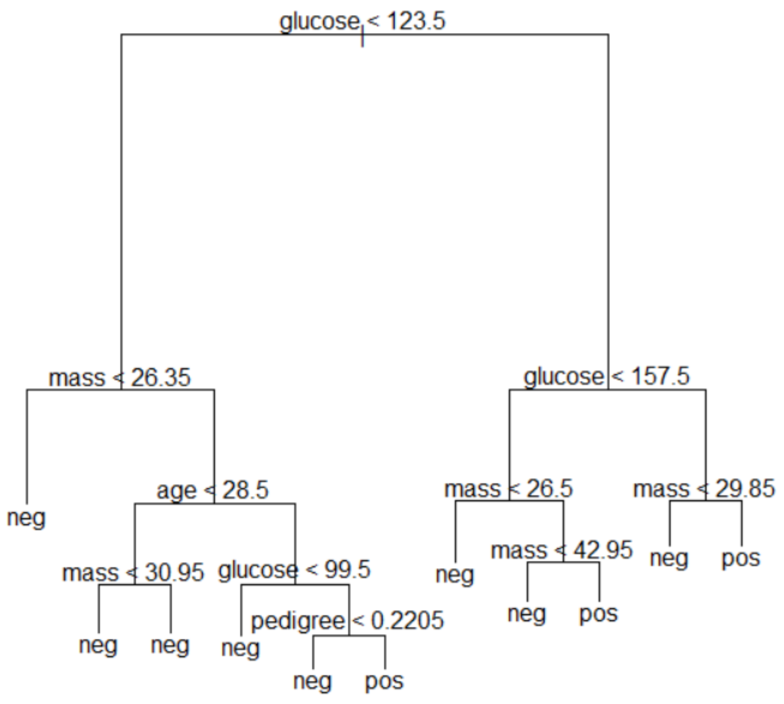
```
> win.graph()
> plot.roc(as.numeric(ValData$diabetes),as.numeric(PredY), print.auc=TRUE)
```

	實際 neg (+	實際 pos (-	總計
預測 neg	88	24	112
預測 pos	12	30	42
總計	100	54	154
	Se(敏感度) =88/100 =0.88	Sp(特異度) =30/54 =0.5556	正確率:118/154=0.7662

5. Classification Tree

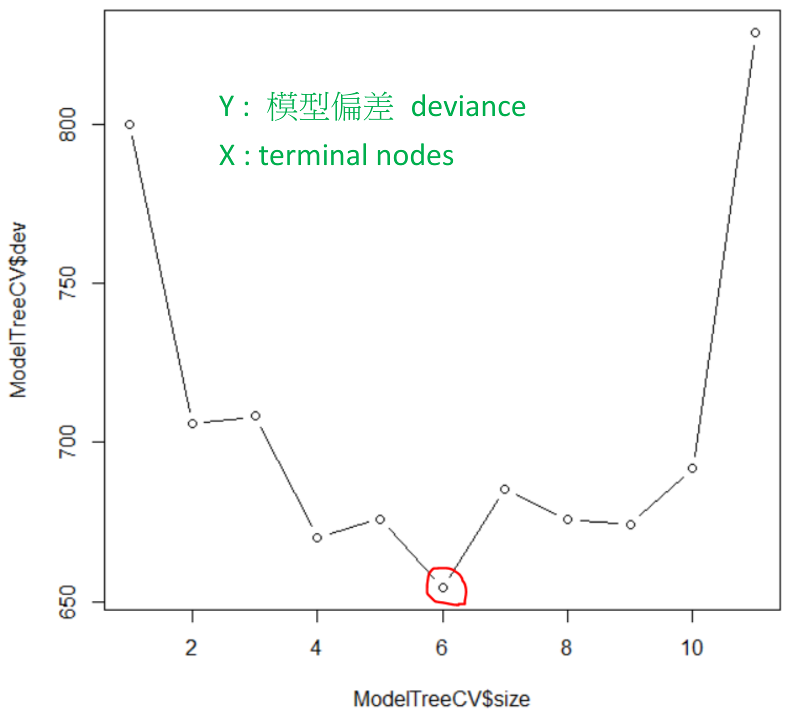
```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)  #先做一個table分成R、SO,再各取80%，才
> set.seed(3)
> Trainget=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]),method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainData=TrainData[,-c(10,11,12)]
> ValData=AvaData[ValInx,]
>
> ModelTree=tree(diabetes~., data=AvaData, subset=TrainInx)  #data放全資料
> win.graph()
> plot(ModelTree)
> text(ModelTree)   #文字附上
> ModelTreeCV=cv.tree(ModelTree)   #用cross validation方法看不同尺寸的樹
> win.graph()
> plot(ModelTreeCV$size, ModelTreeCV$dev, type="b")
>
> ModelPruneTree=prune.tree(ModelTree,best=6)
> win.graph()
> plot(ModelPruneTree)
> text(ModelPruneTree)
>
> PredProbY=predict(ModelPruneTree,ValData[,-9])
> PredY=as.factor(ifelse(as.data.frame(PredProbY)$pos>0.5,"pos","neg"))
> confusionMatrix(PredY, ValData$diabetes)
```

由 training set 建構一棵 classification tree



11 個 terminal nodes

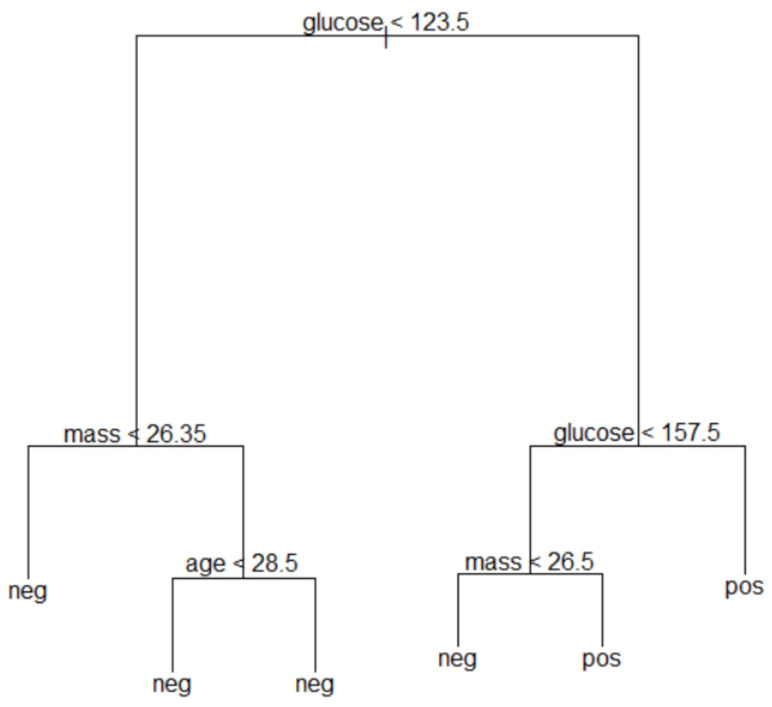
接著用 10—fold cross validation 方法來看不同尺寸的樹，並判斷是否修剪樹



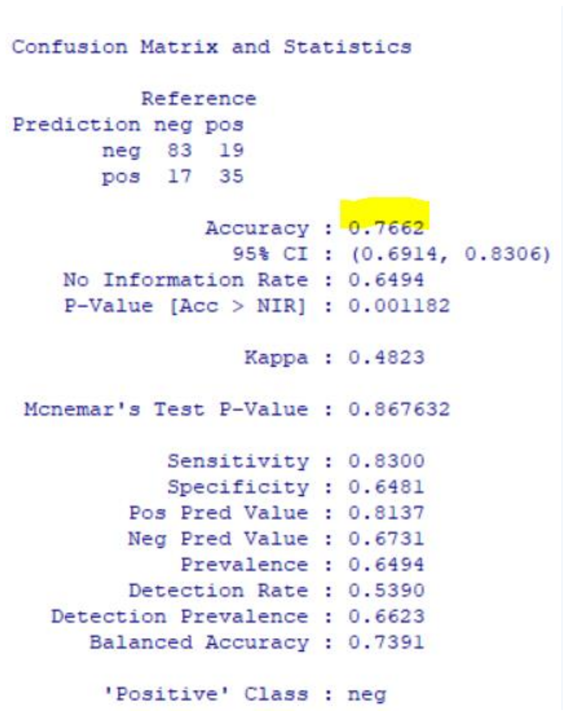
Deviance :模型偏差
模型偏差大小可以反映一個模型你和數據的程度，偏差高代表該模型對數據的擬合越差。

因而選擇 deviance(dev)最低的，或者 deviance 下降程度趨於平緩時的 terminal nodes 數(size)來修剪樹。

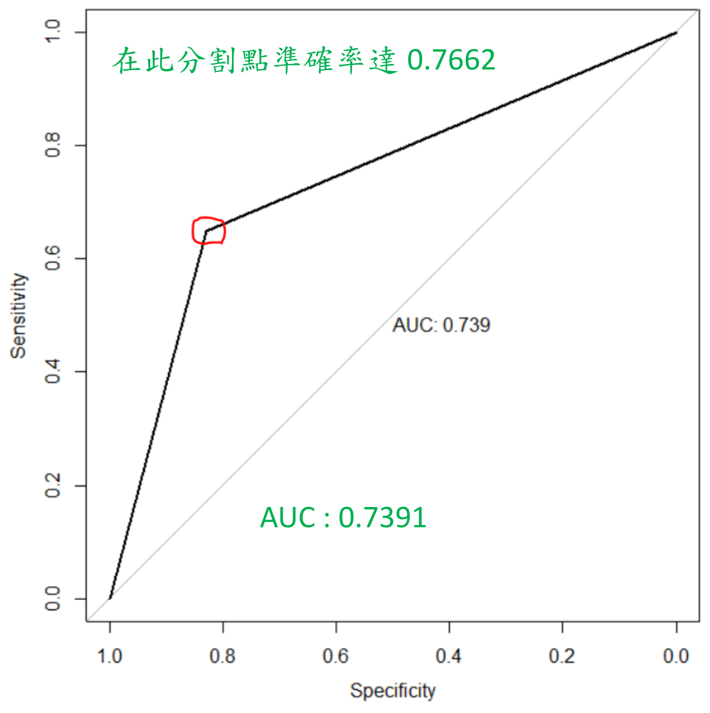
將 validation data 的 X 帶入修剪過後的樹，得到 Y 預測值並計算 accuracy



由挑選出的最佳 tree size:6 來建構樹



```
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  83  19
       pos  17  35

               Accuracy : 0.7662
                 95% CI : (0.6914, 0.8306)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 0.001182

                  Kappa : 0.4823

 Mcnemar's Test P-Value : 0.867632

            Sensitivity : 0.8300
            Specificity : 0.6481
         Pos Pred Value : 0.8137
         Neg Pred Value : 0.6731
             Prevalence : 0.6494
         Detection Rate : 0.5390
   Detection Prevalence : 0.6623
      Balanced Accuracy : 0.7391

       'Positive' Class : neg
```



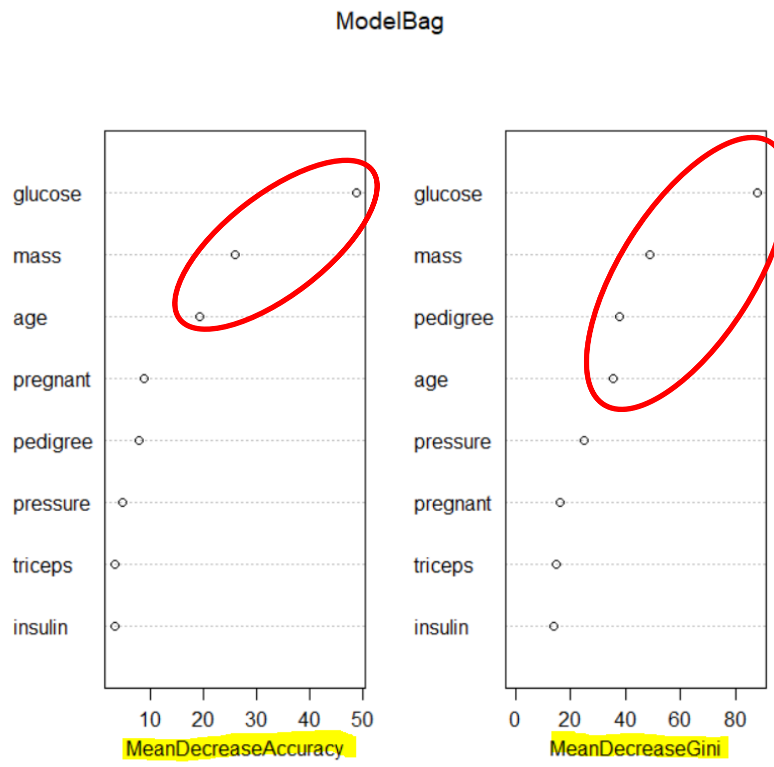
```
> win.graph()
> plot.roc(as.numeric(ValData$diabetes),as.numeric(PredY), print.auc=TRUE)
```

	實際 neg (+	實際 pos (-	總計
預測 neg	83	19	102
預測 pos	17	35	52
總計	100	54	154
	Se(敏感度) =83/100 =0.83	Sp(特異度) =35/54 =0.6481	正確率:118/154=0.7662

6. Bagging

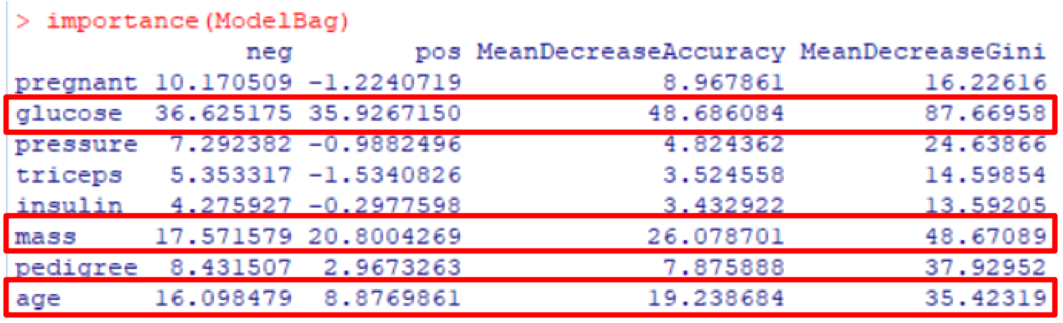
```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)
> set.seed(3)
> Trainget=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]),method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainData=TrainData[,-c(10,11,12)]
> ValData=AvaData[ValInx,]
> ModelBag=randomForest(diabetes~., data=AvaData,subset=TrainInx, mtry=8, importance=T)
> PredY=predict(ModelBag,newdata=ValData[,-9],type="response")
> confusionMatrix(PredY, ValData$diabetes)
```

由 training set 建構 bagging model



MeanDecreaseAccuracy:
準確率越高越好，平均下降的準確率越大代表變數 X 越重要。

MeanDecreaseGini:
Gini 越大純度越高，平均下降的 Gini 值越大代表變數 X 越重要。

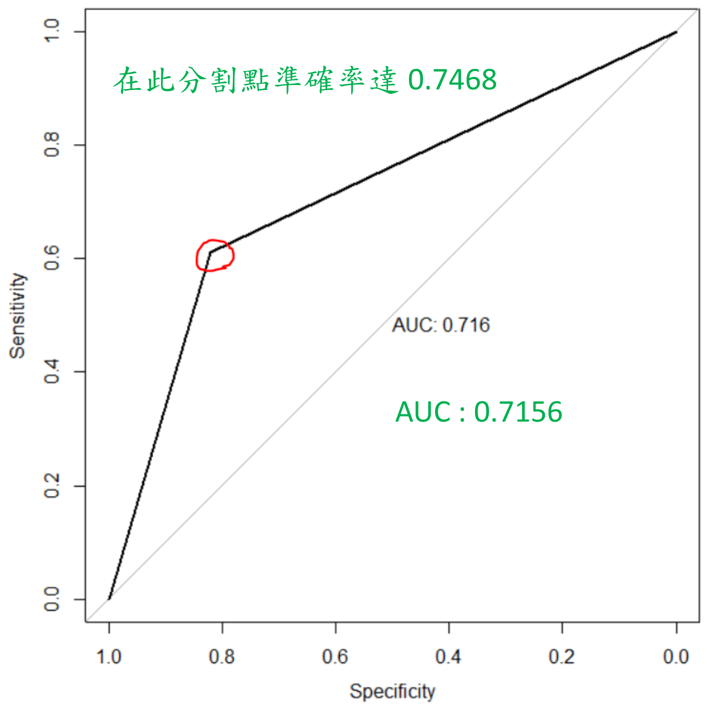


```
> importance(ModelBag)
               neg        pos MeanDecreaseAccuracy MeanDecreaseGini
pregnant 10.170509 -1.2240719             8.967861         16.22616
glucose  36.625175 35.9267150            48.686084         87.66958
pressure  7.292382 -0.9882496             4.824362         24.63866
triceps   5.353317 -1.5340826             3.524558         14.59854
insulin   4.275927 -0.2977598             3.432922         13.59205
mass     17.571579 20.8004269            26.078701         48.67089
pedigree  8.431507  2.9673263             7.875888         37.92952
age      16.098479  8.8769861            19.238684         35.42319
```

由上圖也可看出 glucose、mass、age 三變數較為重要
左邊兩行數據顯示，少了此變數後影響 Y(neg、pos)準確率，使之下降程度

接著由 validation set 的 X 帶入 bagging model 預測 Y，並計算準確率

```
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  82  21
       pos  18  33

               Accuracy : 0.7468
                 95% CI : (0.6705, 0.8133)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 0.006192

                  Kappa : 0.4367

 Mcnemar's Test P-Value : 0.748774

            Sensitivity : 0.8200
            Specificity : 0.6111
         Pos Pred Value : 0.7961
         Neg Pred Value : 0.6471
             Prevalence : 0.6494
         Detection Rate : 0.5325
   Detection Prevalence : 0.6688
      Balanced Accuracy : 0.7156

       'Positive' Class : neg
```



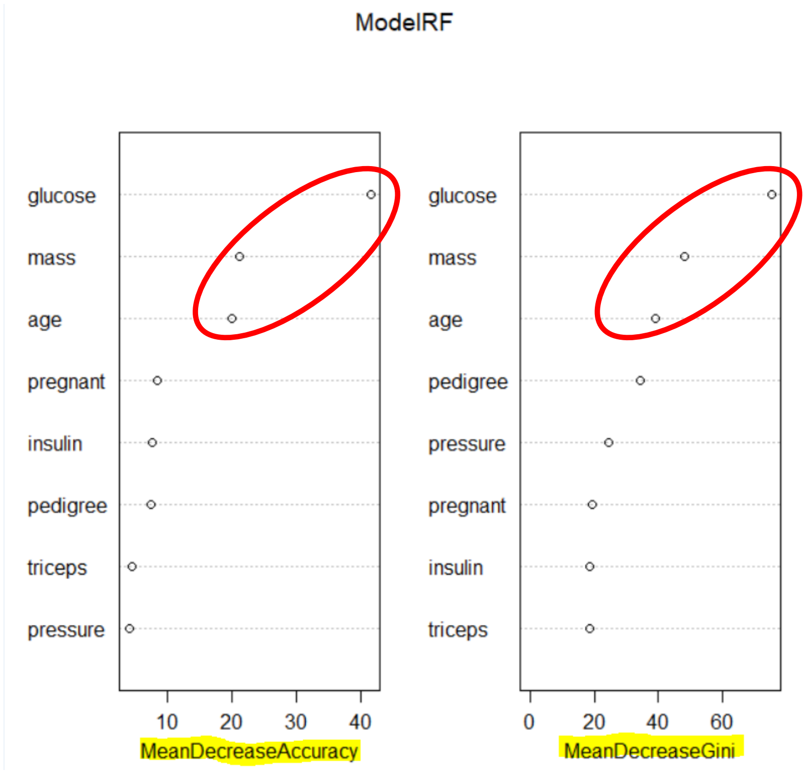
```
> win.graph()
> plot.roc(as.numeric(ValData$diabetes),as.numeric(PredY), print.auc=TRUE)
```

	實際 neg (+	實際 pos (-	總計
預測 neg	82	21	103
預測 pos	18	33	51
總計	100	54	154
	Se(敏感度) =82/100 =0.82	Sp(特異度) =33/54 0.6111	正確率:115/154=0.7468

7. Random Forest

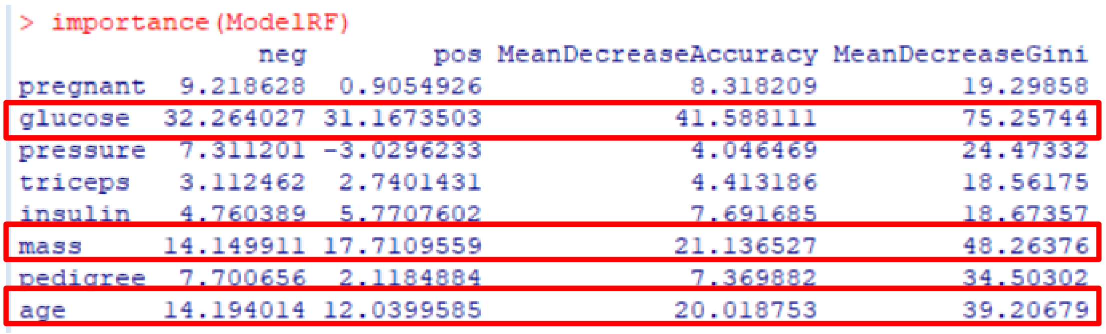
```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)
> set.seed(3)
> Trainget=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]),method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainData=TrainData[-c(10,11,12)]
> ValData=AvaData[ValInx,]
> ModelRF=randomForest(diabetes~., data=AvaData, subset=TrainInx, mtry=3, importance=T)
> PredY=predict(ModelRF,newdata=ValData[,-9],type="response")
> confusionMatrix(PredY, ValData$diabetes)
```

由 training set 建構 random forest model



MeanDecreaseAccuracy:
準確率越高越好，平均下降的準確率越大代表變數 X 越重要。

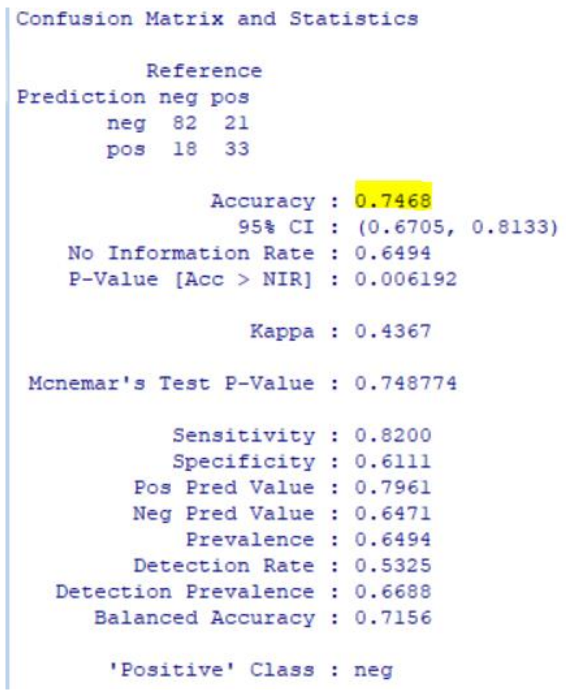
MeanDecreaseGini:
Gini 越大純度越高，平均下降的 Gini 值越大代表變數 X 越重要。



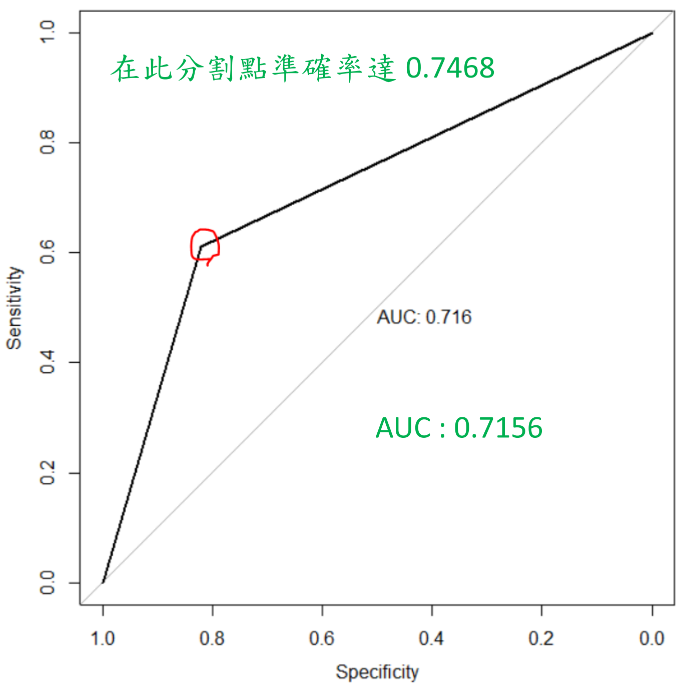
```
> importance(ModelRF)
               neg        pos MeanDecreaseAccuracy MeanDecreaseGini
pregnant  9.218628  0.9054926             8.318209         19.29858
glucose  32.264027 31.1673503            41.588111         75.25744
pressure  7.311201 -3.0296233             4.046469         24.47332
triceps   3.112462  2.7401431             4.413186         18.56175
insulin   4.760389  5.7707602             7.691685         18.67357
mass     14.149911 17.7109559            21.136527         48.26376
pedigree  7.700656  2.1184884             7.369882         34.50302
age      14.194014 12.0399585            20.018753         39.20679
```

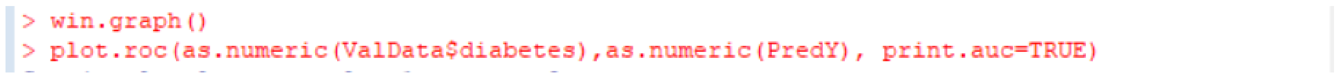
由上圖也可看出 glucose、mass、age 三變數較為重要
左邊兩行數據顯示，少了此變數後影響 Y(neg、pos)準確率，使之下降程度

接著由 validation set 的 X 帶入 bagging model 預測 Y，並計算準確率



```
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  82  21
       pos  18  33

               Accuracy : 0.7468
                 95% CI : (0.6705, 0.8133)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 0.006192

                  Kappa : 0.4367

 Mcnemar's Test P-Value : 0.748774

            Sensitivity : 0.8200
            Specificity : 0.6111
         Pos Pred Value : 0.7961
         Neg Pred Value : 0.6471
             Prevalence : 0.6494
         Detection Rate : 0.5325
   Detection Prevalence : 0.6688
      Balanced Accuracy : 0.7156

       'Positive' Class : neg
```



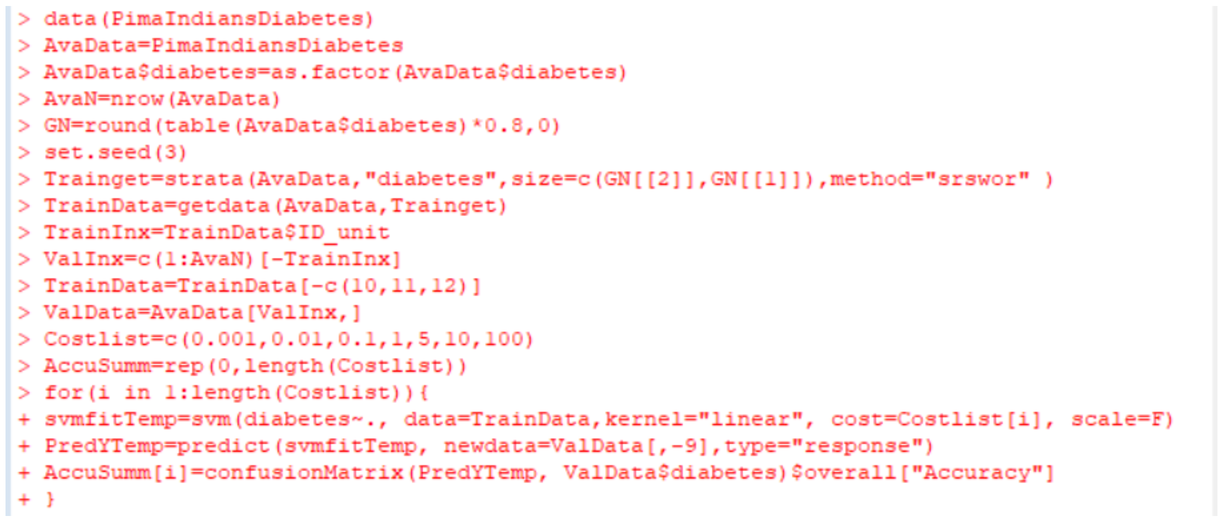


```
> win.graph()
> plot.roc(as.numeric(ValData$diabetes),as.numeric(PredY), print.auc=TRUE)
```

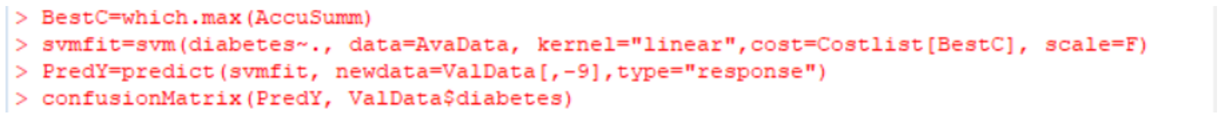
	實際 neg (+	實際 pos (-	總計
預測 neg	82	21	103
預測 pos	18	33	51
總計	100	54	154
	Se(敏感度) =82/100 =0.82	Sp(特異度) =33/54 =0.6111	正確率:115/154=0.7468

發現 : Bagging 和 Random forest 的預測結果差不多。

8. Support Vector

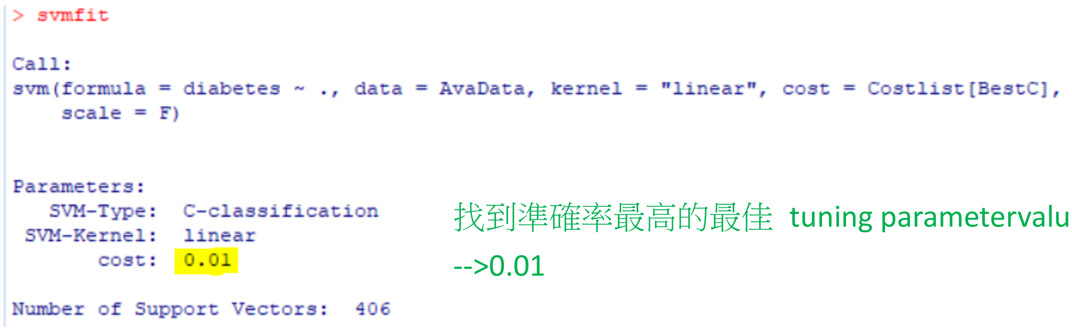


```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)
> set.seed(3)
> Trainget=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]),method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainData=TrainData[-c(10,11,12)]
> ValData=AvaData[ValInx,]
> Costlist=c(0.001,0.01,0.1,1,5,10,100)
> AccuSumm=rep(0,length(Costlist))
> for(i in 1:length(Costlist)){
+ svmfitTemp=svm(diabetes~., data=TrainData,kernel="linear", cost=Costlist[i], scale=F)
+ PredYTemp=predict(svmfitTemp, newdata=ValData[,-9],type="response")
+ AccuSumm[i]=confusionMatrix(PredYTemp, ValData$diabetes)$overall["Accuracy"]
+ }
```



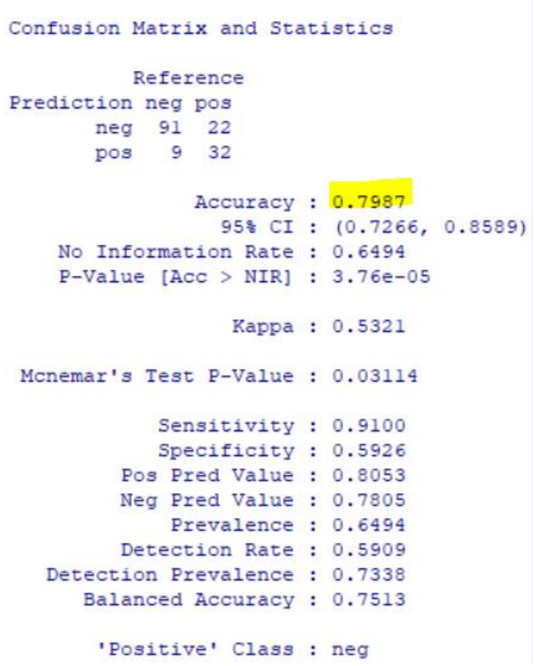
```
> BestC=which.max(AccuSumm)
> svmfit=svm(diabetes~., data=AvaData, kernel="linear",cost=Costlist[BestC], scale=F)
> PredY=predict(svmfit, newdata=ValData[,-9],type="response")
> confusionMatrix(PredY, ValData$diabetes)
```

設定各種 cost(tuning parameter)的值(0.001,0.01,0.1,1,5,10,100)
對每個 tuning parameter 值，以 training set 建構 model 並以 validation set 的 X 代入建構的 model，得到 Y 的預測情況並計算準確率

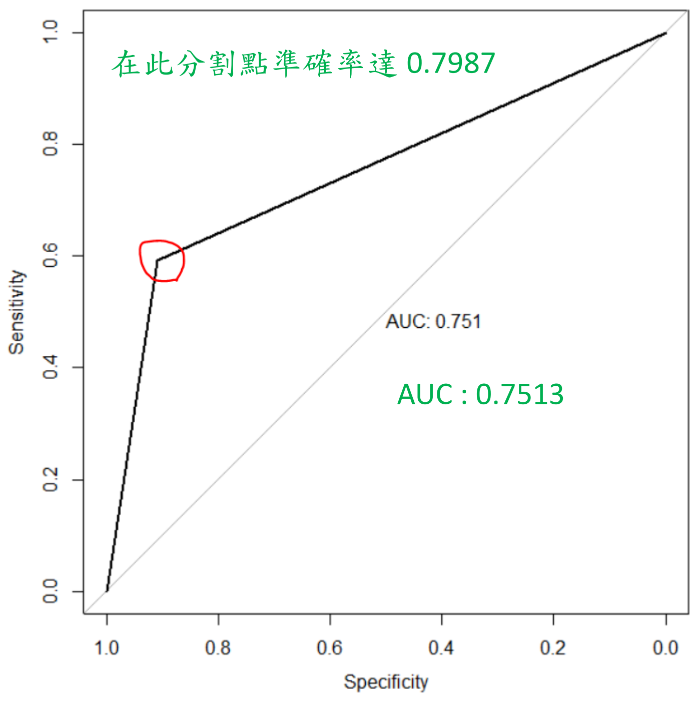


```
> svmfit

Call:
svm(formula = diabetes ~ ., data = AvaData, kernel = "linear", cost = Costlist[BestC],
    scale = F)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  0.01

Number of Support Vectors:  406
```

找到準確率最高的最佳 tuning parametervalue
-->0.01



```
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  91  22
       pos   9  32

               Accuracy : 0.7987
                 95% CI : (0.7266, 0.8589)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 3.76e-05

                  Kappa : 0.5321

 Mcnemar's Test P-Value : 0.03114

            Sensitivity : 0.9100
            Specificity : 0.5926
         Pos Pred Value : 0.8053
         Neg Pred Value : 0.7805
             Prevalence : 0.6494
         Detection Rate : 0.5909
   Detection Prevalence : 0.7338
      Balanced Accuracy : 0.7513

       'Positive' Class : neg
```



```
> win.graph()
> plot.roc(as.numeric(ValData$diabetes),as.numeric(PredY),print.auc=TRUE)
```

	實際 neg (+	實際 pos (-	總計
預測 neg	91	22	113
預測 pos	9	32	41
總計	100	54	154
	Se(敏感度) =91/100 =0.91	Sp(特異度) =32/54 =0.5926	正確率:123/154=0.7987

9. Neural network

```
> data(PimaIndiansDiabetes)
> AvaData=PimaIndiansDiabetes
> AvaData$diabetes=as.factor(AvaData$diabetes)
> AvaN=nrow(AvaData)
> GN=round(table(AvaData$diabetes)*0.8,0)
> set.seed(3)
> Trainget=strata(AvaData,"diabetes",size=c(GN[[2]],GN[[1]]),method="srswor" )
> TrainData=getdata(AvaData,Trainget)
> TrainInx=TrainData$ID_unit
> ValInx=c(1:AvaN)[-TrainInx]
> TrainData=TrainData[-c(10,11,12)]
> ValData=AvaData[ValInx,]
> NNfit=nnet(diabetes~., data=TrainData, size=25, range=0.7)
# weights:  251
initial  value 493.071588
iter  10 value 368.892983
iter  20 value 354.408205
iter  30 value 343.291718
iter  40 value 333.208536
iter  50 value 320.507387
iter  60 value 314.469485
iter  70 value 303.304727
iter  80 value 297.108047
iter  90 value 288.648807
iter 100 value 281.086314
final  value 281.086314
stopped after 100 iterations
> PredY=predict(NNfit,newdata=ValData[,-9],type="class")
> confusionMatrix(as.factor(PredY), ValData$diabetes)
```

由 training set 建構 neural network model

設定 number of unit in the hidden layer (size) =25

設定 initial weight 的 range 為 [-0.7,0.7]

```
Confusion Matrix and Statistics

          Reference
Prediction neg pos
       neg  80  28
       pos  20  26

               Accuracy : 0.6883
                 95% CI : (0.6088, 0.7604)
    No Information Rate : 0.6494
    P-Value [Acc > NIR] : 0.1768

                  Kappa : 0.2914

 Mcnemar's Test P-Value : 0.3123

            Sensitivity : 0.8000
            Specificity : 0.4815
         Pos Pred Value : 0.7407
         Neg Pred Value : 0.5652
             Prevalence : 0.6494
         Detection Rate : 0.5195
   Detection Prevalence : 0.7013
      Balanced Accuracy : 0.6407

       'Positive' Class : neg
```

AUC : 0.6883

在此分割點準確率達 0.6407

用此方法的準確率卻不高的原因可能是因為本身資料集的 Y 就不均勻

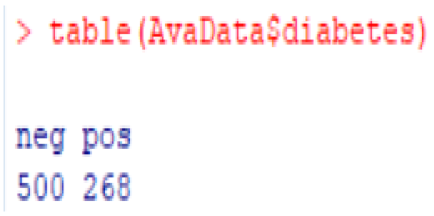
```
> table(AvaData$diabetes)

neg pos
500 268
```

	實際 neg (+	實際 pos (-	總計
預測 neg	80	28	108
預測 pos	20	26	46
總計	100	54	154
	Se(敏感度) =80/100 =0.8	Sp(特異度) =26/54 =0.4815	正確率:106/154=0.6883

肆、總結

	Se(敏感度)	Sp(特異度)	準確率
KNN	0.92	0.4630	0.7597
LOGISTIC	0.89	0.6494	0.7792
LDA	0.90	0.5741	0.7857
QDA	0.88	0.5556	0.7662
CLASSSIFICATION TREE	0.83	0.6481	0.7662
BAGGING	0.82	0.6111	0.7468
RANDOM FOREST	0.82	0.6111	0.7468
SUPPORT VECTOR	0.91	0.5926	0.7987
NEURAL NETWORK	0.80	0.4815	0.6883

用 neural network 方法的準確率卻不高的原因可能是因為本身資料集的 Y 就不均勻



```
> table(AvaData$diabetes)

neg pos
500 268
```

伍、資料來源

https://www.kaggle.com/uciml/pima-indians-diabetes-database