資料探勘期末報告

<世界名畫分析>

統計 4C　403658064　羅文傑

指導教授　　陳景祥

資料來源:

本次的資料來源，是上網搜尋 20 張世界名畫的圖片，並用 Google 表單製成線上問卷，隨機發給親朋好友，所蒐集得來的資料。

問卷網址:

https://docs.google.com/forms/d/1WKGHE2MxyjO5DfWqimsCV2MWYGS9jrnqT8QMKG2H3bk/viewform?c=0&w=1

運用 shiny 互動式網站:

https://wenjlo.shinyapps.io/shiny3/

資料回收後可得 28 筆資料，刪除無效問卷後(刪除有遺失值的資料)，實得 20 筆資料，也就是這 20 張圖片各有 20 個答案，

其中每題答案分別有:

喜歡(good)

普通(common)

不喜歡(bad)

分析步驟:

首先，從 goole 表單轉成 xlsx 檔，再將資料加以整理如下

並將 20 張圖片讀入 R

picture	good	common	bad
p1	5	14	1
p2	8	7	5
p3	9	8	3
p4	6	13	1
p5	15	4	1
p6	13	6	1
p7	10	9	1
p8	13	5	2
p9	12	6	2
p10	7	8	5
p11	13	4	3
p12	2	10	8
p13	14	5	1
p14	5	9	6
p15	4	9	7
p16	18	0	2
p17	13	5	2
p18	9	8	3
p19	9	7	4
p20	8	7	5

分析方法:

決策樹

類神經網路

集群分析

關聯規則分析

1.資料初步整理

將 20 個圖檔，透過 jpeg package 讀入 R，並計算出各別的 RGB 值以及灰階值。

這邊我將每筆資料原本的 good、common、bad 人數，透過加權公式 R code: eva=data$good*1+data$common*0+data$bad*-1 (我們將每筆資料的 喜歡人數*1 、普通人數*0、不喜歡人數*-1 然後加總)，而 eva 得到的分數，再透過分類成新的類別變數 evan。

R code: evan=recode(eva,"10:20='good';-20:0='bad';else='common'")

eva 分數	evan
10~20 分	Good
0~9	Common
0 以下	Bad

picture	good	common	bad	gray	red	green	blue	evan
p1	5	14	1	0.4188871028	0.4758767400	0.4039785621	0.3464953768	common
p2	8	7	5	0.5774311851	0.6049309964	0.5865923543	0.4587094785	common
p3	9	8	3	0.4379727848	0.6049309964	0.5865923543	0.4587094785	common
p4	6	13	1	0.5230512219	0.6049309964	0.5865923543	0.4587094785	common
p5	15	4	1	0.4487876181	0.6049309964	0.5865923543	0.4587094785	good
p6	13	6	1	0.5464648506	0.6049309964	0.5865923543	0.4587094785	good
p7	10	9	1	0.3453606417	0.6049309964	0.5865923543	0.4587094785	common
p8	13	5	2	0.4954652161	0.6049309964	0.5865923543	0.4587094785	good
p9	12	6	2	0.5188190364	0.6049309964	0.5865923543	0.4587094785	good
p10	7	8	5	0.5505802108	0.6049309964	0.5865923543	0.4587094785	common
p11	13	4	3	0.4891715641	0.6049309964	0.5865923543	0.4587094785	good
p12	2	10	8	0.3901785385	0.6049309964	0.5865923543	0.4587094785	bad
p13	14	5	1	0.4105986947	0.6049309964	0.5865923543	0.4587094785	good
p14	5	9	6	0.2690228477	0.6049309964	0.5865923543	0.4587094785	bad
p15	4	9	7	0.2946332586	0.6049309964	0.5865923543	0.4587094785	bad
p16	18	0	2	0.5414011833	0.6049309964	0.5865923543	0.4587094785	good
p17	13	5	2	0.4771793413	0.6049309964	0.5865923543	0.4587094785	good
p18	9	8	3	0.6446370450	0.6049309964	0.5865923543	0.4587094785	common
p19	9	7	4	0.5911413923	0.6049309964	0.5865923543	0.4587094785	common
p20	8	7	5	0.5200417967	0.6049309964	0.5865923543	0.4587094785	common

這邊的灰階值(gray)、RGB 值，是每張圖片各別的灰階與 RGB 取平均值。

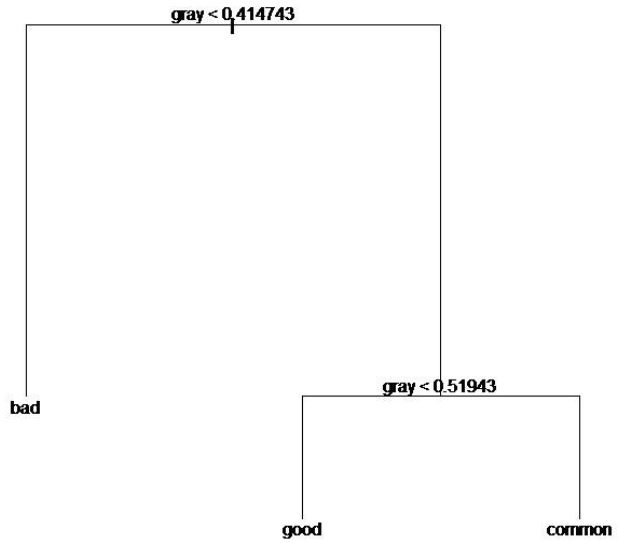
2.決策樹

tree=tree(evan~gray+red+green+blue,data=data2,method="class")

我們以 evan 分類變數(也就是我們加權後分類的變數)當反應變數，以 gray、 RGB 當 解釋變數 。

以資料的 30%當測試樣本(因為只有 20 筆資料，所以取比較大的比例)

結果如下:



訓練樣本 table &正確率、測試樣本 table&正確率

```
                train.predict
evan.traindata bad common good
        bad      2      0    0
        common   0      4    2
        good     0      2    4
```

```
[1] 0.7142857
> table.testdata
              test.predict
evan.testdata bad common good
       bad      1      0    0
       common   1      2    0
       good     1      0    1
> sum(diag(table.testdata))/sum(table.testdata)
[1] 0.6666667
```

透過決策樹，大致上有以下的結果

a. RGB 對決策樹分類沒有貢獻

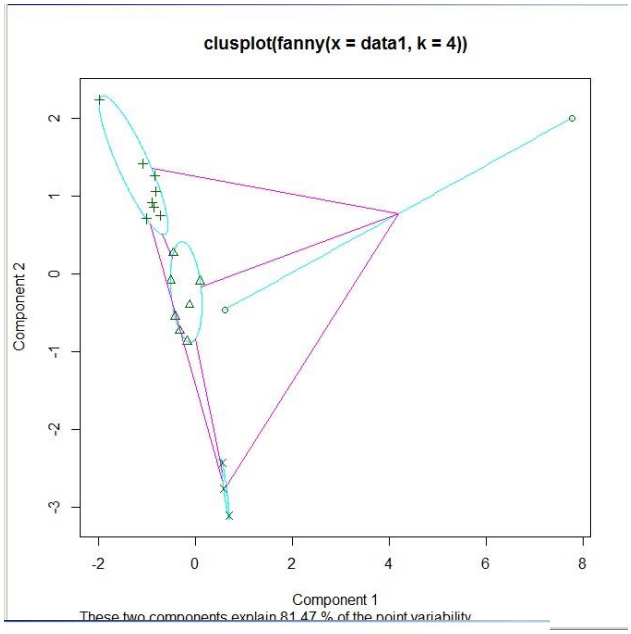
b. 平均灰階值<0.414743 -> bad

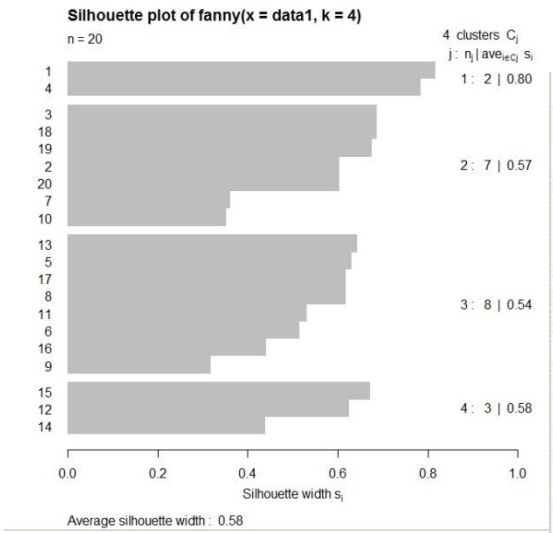
0.414743<平均灰階值<0.51943 -> good

平均灰階值>0.51943 -> common

3.集群分析

(我們運用 fpc package，得到的建議分群數為 4)

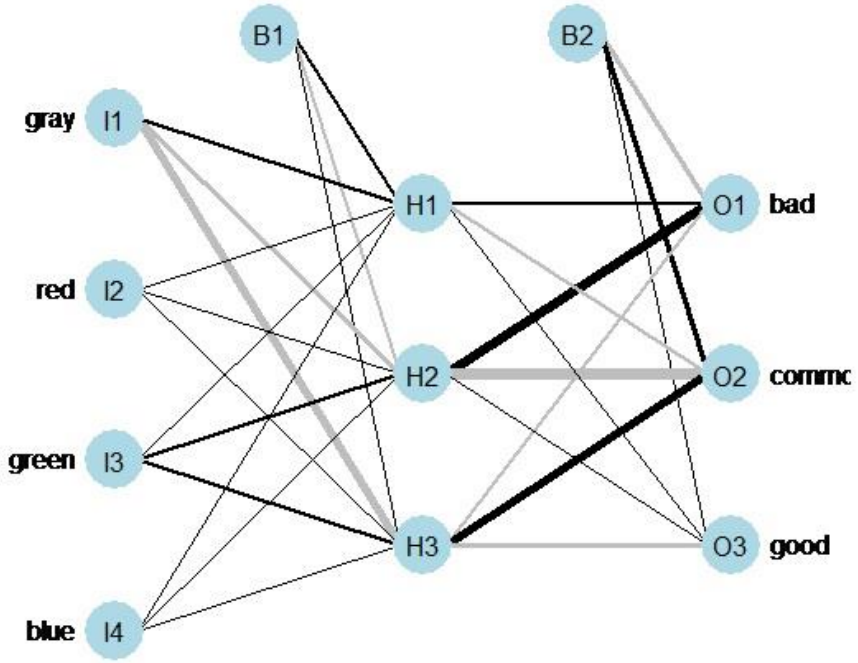




	cluster	neighbor	sil_width
1	1	4	0.8166612
4	1	2	0.7845621
3	2	3	0.6863635
18	2	3	0.6862598
19	2	3	0.6759807
2	2	4	0.6036032
20	2	4	0.6034220
7	2	3	0.3606923
10	2	4	0.3522061
13	3	2	0.6427289
5	3	2	0.6309031
17	3	2	0.6188602
8	3	2	0.6188555
11	3	2	0.5313731
6	3	2	0.5150654
16	3	2	0.4411138
9	3	2	0.3179708
15	4	2	0.6715245
12	4	2	0.6249767
14	4	2	0.4387434

查看各個圖片被分類到哪一群:

第一群:1,4
第二群:3,18,19,2,20,7,10
第三群:13,5,17,8,11,6,16,9
第四群:15,12,14
4.類神經網路(4-3-3)



類神經網路預測正確率為 75%，還算可以。

各隱藏層的 weights 如下:

```
> result=nnet(evan~gray+red+green+blue,data=data2,size=3)
# weights:  27
initial  value 22.763883
iter  10 value 19.396115
iter  20 value 13.867662
iter  30 value 10.820284
iter  40 value 10.466846
iter  50 value 10.398693
iter  60 value 10.356530
iter  70 value 10.257691
iter  80 value 10.182978
iter  90 value 10.007176
iter 100 value 9.942053
final  value 9.942053
stopped after 100 iterations
> ypred=predict(result,data2,type="class")
> t1=table(data2$evan,ypred)
> sum(diag(t1))/sum(t1)
[1] 0.75
> summary(result)
a 4-3-3 network with 27 weights
options were - softmax modelling
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1
  9.61  12.86   2.80   0.74   1.46
 b->h2 i1->h2 i2->h2 i3->h2 i4->h2
 -9.42 -14.34   6.95  12.26   6.62
 b->h3 i1->h3 i2->h3 i3->h3 i4->h3
  1.09 -31.63   5.84   8.36   4.35
 b->o1 h1->o1 h2->o1 h3->o1
-17.08   7.78  38.60 -11.34
 b->o2 h1->o2 h2->o2 h3->o2
 15.00  -8.81 -40.55  29.66
 b->o3 h1->o3 h2->o3 h3->o3
  2.00   1.24   2.90 -17.67
```

5.關聯規則分析　　(supp=0.4 ,conf=0.6)

R code:

```
gray1=ordered(cut(data3$gray,breaks=4))
red1=ordered(cut(data3$red,breaks=4))
green1=ordered(cut(data3$green,breaks=4))
blue1=ordered(cut(data3$blue,breaks=4))
```

我們將灰階、RGB 四個數值，各切成四等份。

另外產生兩個 data frame，一個是 RGB 與 evan ，一個是 gray 與 evan

原因是因為灰階值本來就是透過 RGB 轉換公式得來，

若全部放在同一個 data frame，將會出現多筆 gray 與 RGB 相關的規則，

不方便我們尋找想要的資訊。

(1)RGB 與 evan 關聯規則

R code :　　data4=data.frame(red1,green1,blue1,data3$evan)

	lhs		rhs	support	confidence	lift
4	{data3.evan=good}	=>	{red1=(0.573,0.605]}	0.40	1.0000000	1.0526316
5	{data3.evan=good}	=>	{green1=(0.541,0.587]}	0.40	1.0000000	1.0526316
6	{data3.evan=good}	=>	{blue1=(0.431,0.459]}	0.40	1.0000000	1.0526316
10	{red1=(0.573,0.605]}	=>	{green1=(0.541,0.587]}	0.95	1.0000000	1.0526316
11	{green1=(0.541,0.587]}	=>	{red1=(0.573,0.605]}	0.95	1.0000000	1.0526316
12	{red1=(0.573,0.605]}	=>	{blue1=(0.431,0.459]}	0.95	1.0000000	1.0526316
13	{blue1=(0.431,0.459]}	=>	{red1=(0.573,0.605]}	0.95	1.0000000	1.0526316
14	{green1=(0.541,0.587]}	=>	{blue1=(0.431,0.459]}	0.95	1.0000000	1.0526316
15	{blue1=(0.431,0.459]}	=>	{green1=(0.541,0.587]}	0.95	1.0000000	1.0526316
16	{red1=(0.573,0.605],data3.evan=good}	=>	{green1=(0.541,0.587]}	0.40	1.0000000	1.0526316
17	{green1=(0.541,0.587],data3.evan=good}	=>	{red1=(0.573,0.605]}	0.40	1.0000000	1.0526316
18	{red1=(0.573,0.605],data3.evan=good}	=>	{blue1=(0.431,0.459]}	0.40	1.0000000	1.0526316
19	{blue1=(0.431,0.459],data3.evan=good}	=>	{red1=(0.573,0.605]}	0.40	1.0000000	1.0526316
20	{green1=(0.541,0.587],data3.evan=good}	=>	{blue1=(0.431,0.459]}	0.40	1.0000000	1.0526316
21	{blue1=(0.431,0.459],data3.evan=good}	=>	{green1=(0.541,0.587]}	0.40	1.0000000	1.0526316
22	{red1=(0.573,0.605],data3.evan=common}	=>	{green1=(0.541,0.587]}	0.40	1.0000000	1.0526316
23	{green1=(0.541,0.587],data3.evan=common}	=>	{red1=(0.573,0.605]}	0.40	1.0000000	1.0526316
24	{red1=(0.573,0.605],data3.evan=common}	=>	{blue1=(0.431,0.459]}	0.40	1.0000000	1.0526316
25	{blue1=(0.431,0.459],data3.evan=common}	=>	{red1=(0.573,0.605]}	0.40	1.0000000	1.0526316
26	{green1=(0.541,0.587],data3.evan=common}	=>	{blue1=(0.431,0.459]}	0.40	1.0000000	1.0526316
27	{blue1=(0.431,0.459],data3.evan=common}	=>	{green1=(0.541,0.587]}	0.40	1.0000000	1.0526316
28	{red1=(0.573,0.605],green1=(0.541,0.587]}	=>	{blue1=(0.431,0.459]}	0.95	1.0000000	1.0526316
29	{red1=(0.573,0.605],blue1=(0.431,0.459]}	=>	{green1=(0.541,0.587]}	0.95	1.0000000	1.0526316
30	{green1=(0.541,0.587],blue1=(0.431,0.459]}	=>	{red1=(0.573,0.605]}	0.95	1.0000000	1.0526316
31	{red1=(0.573,0.605],green1=(0.541,0.587],data3.evan=good}	=>	{blue1=(0.431,0.459]}	0.40	1.0000000	1.0526316
32	{red1=(0.573,0.605],blue1=(0.431,0.459],data3.evan=good}	=>	{green1=(0.541,0.587]}	0.40	1.0000000	1.0526316
33	{green1=(0.541,0.587],blue1=(0.431,0.459],data3.evan=good}	=>	{red1=(0.573,0.605]}	0.40	1.0000000	1.0526316
34	{red1=(0.573,0.605],green1=(0.541,0.587],data3.evan=common}	=>	{blue1=(0.431,0.459]}	0.40	1.0000000	1.0526316
35	{red1=(0.573,0.605],blue1=(0.431,0.459],data3.evan=common}	=>	{green1=(0.541,0.587]}	0.40	1.0000000	1.0526316
36	{green1=(0.541,0.587],blue1=(0.431,0.459],data3.evan=common}	=>	{red1=(0.573,0.605]}	0.40	1.0000000	1.0526316
1	{}	=>	{red1=(0.573,0.605]}	0.95	0.9500000	1.0000000
2	{}	=>	{green1=(0.541,0.587]}	0.95	0.9500000	1.0000000
3	{}	=>	{blue1=(0.431,0.459]}	0.95	0.9500000	1.0000000
7	{data3.evan=common}	=>	{red1=(0.573,0.605]}	0.40	0.8888889	0.9356725
8	{data3.evan=common}	=>	{green1=(0.541,0.587]}	0.40	0.8888889	0.9356725
9	{data3.evan=common}	=>	{blue1=(0.431,0.459]}	0.40	0.8888889	0.9356725

5.關聯規則分析

(2) gray 與 evan 關聯規則分析

R code :data44=data.frame(gray1,data3$evan)

	lhs		rhs	support	confidence	lift
1	{data3.evan=good}	=>	{gray1=(0.457,0.551]}	0.3	0.7500000	1.666667
2	{gray1=(0.457,0.551]}	=>	{data3.evan=good}	0.3	0.6666667	1.666667

結論

決策樹 V.S 關聯規則

決策樹:

平均灰階值<0.414743 -> bad

0.414743< 平均灰階值<0.51943 -> good

平均灰階值>0.51943 -> common

關聯規則:

0.457<平均灰階值<0.551 -> good

0.573 < red < 0.605 -> good

0.541< green < 0.587 -> good

0.431 < blue <0.459 -> good.....etc

levels:

```
> levels(gray1)
[1] "(0.269,0.363]" "(0.363,0.457]" "(0.457,0.551]" "(0.551,0.645]"
> levels(red1)
[1] "(0.476,0.508]" "(0.573,0.605]"
> levels(green1)
[1] "(0.404,0.45]"  "(0.541,0.587]"
> levels(blue1)
[1] "(0.346,0.375]" "(0.431,0.459]"
```

我們可以發現，決策樹無法運用 RGB 的資訊做分類，而關聯規則分析雖然可以運用 RGB 做分類，但是灰階值部分只有一條規則(得到 good 的規則)，決策樹可以運用灰階值將圖片分類 good、common、bad。

集群分析

從集群分析的結果來看，大部份的資料被分類在 2、3 群。

第 1 群的資料只有 2 個圖片，而且距離還蠻遠的，在這一群的分群效果看來不是太理想。

綜合以上分析，本次收集到的資料，我發現圖片假如平均灰階值介於 0.45~0.5 左右，容易被大家所喜愛，若以這 20 張圖片來看 0.45~0.5 的灰階值介於中間，換句話來說大家不喜歡太亮或太暗的圖案，尤其是太暗的圖案，容易分類到不喜歡；而太亮的圖案容易被分類到普通。

Rcode:

```
setwd("D:/")
data=read.csv("data.csv")
head(data)

#讀 20 張圖檔
library(jpeg)
pic1=readJPEG("images/1.jpg")
pic2=readJPEG("images/2.jpg")
pic3=readJPEG("images/3.jpg")
pic4=readJPEG("images/4.jpg")
pic5=readJPEG("images/5.jpg")
pic6=readJPEG("images/6.jpg")
pic7=readJPEG("images/7.jpg")
pic8=readJPEG("images/8.jpg")
pic9=readJPEG("images/9.jpg")
pic10=readJPEG("images/10.jpg")
pic11=readJPEG("images/11.jpg")
pic12=readJPEG("images/12.jpg")
pic13=readJPEG("images/13.jpg")
pic14=readJPEG("images/14.jpg")
pic15=readJPEG("images/15.jpg")
pic16=readJPEG("images/16.jpg")
pic17=readJPEG("images/17.jpg")
pic18=readJPEG("images/18.jpg")
pic19=readJPEG("images/19.jpg")
pic20=readJPEG("images/20.jpg")

#轉換成灰階
pic1a=0.2989*pic1[,,1]+0.5866*pic1[,,2]+0.1145*pic1[,,3]
pic2a=0.2989*pic2[,,1]+0.5866*pic2[,,2]+0.1145*pic2[,,3]
pic3a=0.2989*pic3[,,1]+0.5866*pic3[,,2]+0.1145*pic3[,,3]
pic4a=0.2989*pic4[,,1]+0.5866*pic4[,,2]+0.1145*pic4[,,3]
pic5a=0.2989*pic5[,,1]+0.5866*pic5[,,2]+0.1145*pic5[,,3]
pic6a=0.2989*pic6[,,1]+0.5866*pic6[,,2]+0.1145*pic6[,,3]
pic7a=0.2989*pic7[,,1]+0.5866*pic7[,,2]+0.1145*pic7[,,3]
pic8a=0.2989*pic8[,,1]+0.5866*pic8[,,2]+0.1145*pic8[,,3]
pic9a=0.2989*pic9[,,1]+0.5866*pic9[,,2]+0.1145*pic9[,,3]
```

```
pic10a=0.2989*pic10[,,1]+0.5866*pic10[,,2]+0.1145*pic10[,,3]
pic11a=0.2989*pic11[,,1]+0.5866*pic11[,,2]+0.1145*pic11[,,3]
pic12a=0.2989*pic12[,,1]+0.5866*pic12[,,2]+0.1145*pic12[,,3]
pic13a=0.2989*pic13[,,1]+0.5866*pic13[,,2]+0.1145*pic13[,,3]
pic14a=0.2989*pic14[,,1]+0.5866*pic14[,,2]+0.1145*pic14[,,3]
pic15a=0.2989*pic15[,,1]+0.5866*pic15[,,2]+0.1145*pic15[,,3]
pic16a=0.2989*pic16[,,1]+0.5866*pic16[,,2]+0.1145*pic16[,,3]
pic17a=0.2989*pic17[,,1]+0.5866*pic17[,,2]+0.1145*pic17[,,3]
pic18a=0.2989*pic18[,,1]+0.5866*pic18[,,2]+0.1145*pic18[,,3]
pic19a=0.2989*pic19[,,1]+0.5866*pic19[,,2]+0.1145*pic19[,,3]
pic20a=0.2989*pic20[,,1]+0.5866*pic20[,,2]+0.1145*pic20[,,3]

gray=c(mean(pic1a),mean(pic2a),mean(pic3a),mean(pic4a),mean(pic5a),mean(pic6a
),mean(pic7a),mean(pic8a),mean(pic9a),mean(pic10a)
          ,mean(pic11a),mean(pic12a),mean(pic13a),mean(pic14a),mean(pic15a),mea
n(pic16a),mean(pic17a),mean(pic18a),mean(pic19a),mean(pic20a))
red=c(mean(pic1[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]
),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),

mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean
(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]),mean(pic2[,,1]))


green=c(mean(pic1[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,
2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),

mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean
(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]),mean(pic2[,,2]))

blue=c(mean(pic1[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3
]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),

mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean
(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]),mean(pic2[,,3]))

eva=data$good*1+data$common*0+data$bad*-1

library(car)
```

```
evan=recode(eva,"10:20='good';-20:0='bad';else='common'")
data2=cbind(data,gray,red,green,blue,evan)

library(tree)
set.seed(6)
n=0.3*nrow(data2)
test.index=sample(1:nrow(data2),n)
data2.train=data2[-test.index,]
data2.test=data2[test.index,]

#決策樹
tree=tree(evan~gray+red+green+blue,data=data2,method="class")
plot(tree)
text(tree)
#訓練樣本正確率
evan.traindata=data2$evan[-test.index]
train.predict=factor(predict(tree,data2.train,type="class"),levels=levels(evan.traindat
a))
table.traindata=table(evan.traindata,train.predict)
sum(diag(table.traindata))/sum(table.traindata)

#測試樣本正確率
evan.testdata=data2$evan[test.index]
test.predict=factor(predict(tree,data2.test,type="class"),levels=levels(evan.testdata))
table.testdata=table(evan.testdata,test.predict)
sum(diag(table.testdata))/sum(table.testdata)

#類神經
library(nnet)
set.seed(6)
result=nnet(evan~gray+red+green+blue,data=data2,size=3)
#顯示預測正確率
ypred=predict(result,data2,type="class")
t1=table(data2$evan,ypred)
sum(diag(t1))/sum(t1)
summary(result)
#畫圖
library(NeuralNetTools)
```

```
plotnet(result)


data1=data2[,2:8]
library(fpc)
pamk(data1,2:10)

library(cluster)
result1=fanny(data1,4)
result1
summary(result)

plot(result1)
#查看各資料被分類到哪一群
m=result1$silinfo$widths
m[1:20,]
#關聯規則分析
library(arules)
data3=data2[,5:9]
gray1=ordered(cut(data3$gray,breaks=4))
red1=ordered(cut(data3$red,breaks=4))
green1=ordered(cut(data3$green,breaks=4))
blue1=ordered(cut(data3$blue,breaks=4))
#RGB  關聯規則
data4=data.frame(red1,green1,blue1,data3$evan)
data5=as(data4,"transactions")


inspect(data5[1:10,])

rules=apriori(data5,parameter=list(supp=0.2,conf=0.6,target="rules"))
inspect(head(sort(rules,by="lift"),n=100))
#gray    關聯規則
data44=data.frame(gray1,data3$evan)
data55=as(data44,"transactions")


inspect(data55[1:10,])
```

```
rules=apriori(data55,parameter=list(supp=0.2,conf=0.6,target="rules"))
inspect(head(sort(rules,by="lift"),n=100))
```

shiny server code

```
data<-read.csv("data/Res.csv",header = T)

t1=table(data$picture1)
t2=table(data$picture2)
t3=table(data$picture3)
t4=table(data$picture4)
t5=table(data$picture5)
t6=table(data$picture6)
t7=table(data$picture7)
t8=table(data$picture8)
t9=table(data$picture9)
t10=table(data$picture10)
t11=table(data$picture11)
t12=table(data$picture12)
t13=table(data$picture13)
t14=table(data$picture14)
t15=table(data$picture15)
t16=table(data$picture16)
t17=table(data$picture17)
t18=table(data$picture18)
t19=table(data$picture19)
t20=table(data$picture20)

library(shiny)
library(datasets)


shinyServer(function(input, output,session) {



    output$image2 <- renderImage({
        if (is.null(input$picture))
```

```
    return(NULL)

if (input$picture == "picture_1") {
    return(list(
        src = "images/1.jpg",
        contentType = "image/jpeg"

    ))
}
if (input$picture == "picture_2") {
    return(list(
        src = "images/2.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_3") {
    return(list(
        src = "images/3.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_4") {
    return(list(
        src = "images/4.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_5") {
    return(list(
        src = "images/5.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_6") {
```

```
    return(list(
        src = "images/6.jpg",
        filetype = "image/jpeg"

    ))
}

if (input$picture == "picture_7") {
    return(list(
        src = "images/7.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_8") {
    return(list(
        src = "images/8.jpg",
        filetype = "image/jpeg"

    ))
}

if (input$picture == "picture_9") {
    return(list(
        src = "images/9.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_10") {
    return(list(
        src = "images/10.jpg",
        filetype = "image/jpeg"

    ))
}

if (input$picture == "picture_11") {
```

```
    return(list(
        src = "images/11.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_12") {
    return(list(
        src = "images/12.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_13") {
    return(list(
        src = "images/13.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_14") {
    return(list(
        src = "images/14.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_15") {
    return(list(
        src = "images/15.jpg",
        filetype = "image/jpeg"

    ))
}
if (input$picture == "picture_16") {
    return(list(
        src = "images/16.jpg",
        filetype = "image/jpeg"
```

```
        ))
    }
    if (input$picture == "picture_17") {
        return(list(
            src = "images/17.jpg",
            filetype = "image/jpeg"

        ))
    }
    if (input$picture == "picture_18") {
        return(list(
            src = "images/18.jpg",
            filetype = "image/jpeg"

        ))
    }
    if (input$picture == "picture_19") {
        return(list(
            src = "images/19.jpg",
            filetype = "image/jpeg"

        ))
    }
    if (input$picture == "picture_20") {
        return(list(
            src = "images/20.jpg",
            filetype = "image/jpeg"

        ))
    }



}, deleteFile = FALSE)
# Return the requested dataset
datasetInput <- reactive({
    switch(input$dataset,
```

```
                    "picture1" = t1,"picture2"=t2,"picture3"=
t3,"picture4"=t4,"picture5"=t5,

"picture6"=t6,"picture7"=t7,"picture8"=t8,"picture9"=t9,"picture10"=t10,

"picture11"=t11,"picture12"=t12,"picture13"=t13,"picture14"=t14,"picture15"=t15,

"picture16"=t16,"picture17"=t17,"picture18"=t18,"picture19"=t19,"picture20"=t20)

  })


  output$summary <- renderPrint({
    dataset <- datasetInput()
    summary(dataset)
  })

  # Show the first "n" observations
  output$view <- renderTable({
    datasetInput()
  })

  #pie chart
  output$plot<-renderPlot({
    dataset<-datasetInput()
    pie(dataset)

  })

  })
```

shiny ui code:

```
library(shiny)
shinyUI(fluidPage(
  titlePanel("DataMining for final report"),



  sidebarLayout(
```

```
        sidebarPanel(
            selectInput("picture", "Choose a picture    for view :",
                                choices =
c("picture_1","picture1","picture_2","picture_3","picture_4","picture_5",

"picture_6","picture_7","picture_8","picture_9","picture_10",

"picture_11","picture_12","picture_13","picture_14","picture_15",

"picture_16","picture_17","picture_18","picture_19","picture_20")),
  selectInput("dataset", "Choose a picture    for Observations table & Pie chart:",
                              choices =
c("picture1","picture2","picture3","picture4","picture5",

"picture6","picture7","picture8","picture9","picture10",

"picture11","picture12","picture13","picture14","picture15",

"picture16","picture17","picture18","picture19","picture20"))
          ),



  mainPanel(
            h1("View picture"),
            imageOutput("image2"),

        br(),
        br(),
        br(),
        br(),
        br(),
        br(),
        br(),
        br(),
        br(),
        h1("Observations Table"),
        tableOutput("view"),
```

```
            h1("Pie chart for picture"),
            plotOutput("plot")



        )
    )
))
```