Assignment_9.R

Kun Wang

2020-11-10

GitHub Link:

.R File: https://github.com/kunwangRU/Survival-Analysis-of-Patients-with-Heart-Failure/blob/master/LDA%20(Assignment%209).R

```
#Loading Packages
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(psych)

## Warning: package 'psych' was built under R version 3.6.3

library(cluster)

## Warning: package 'cluster' was built under R version 3.6.3

library(fpc)

## Warning: package 'fpc' was built under R version 3.6.3

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.6.3

##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha

library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

library(ROCR)

## Warning: package 'ROCR' was built under R version 3.6.3

library(klaR)

#Loading dataset
rawdata <-read.csv("C:/Users/wangk/Desktop/Rutgers/Rutgers Courseware/Fall
2020/Multivariate Analysis/heart_failure_clinical_records_dataset.csv")
View(rawdata)

#Identifying different columns names
names(rawdata)

##  [1] "age"                       "anaemia"
##  [3] "creatinine_phosphokinase" "diabetes"
##  [5] "ejection_fraction"         "high_blood_pressure"
##  [7] "platelets"                 "serum_creatinine"
##  [9] "serum_sodium"              "sex"
## [11] "smoking"                   "time"
## [13] "DEATH_EVENT"

#Data Summary
str(rawdata)

## 'data.frame':    299 obs. of  13 variables:
##  $ age                      : num  75 55 65 50 65 90 75 60 65 80 ...
##  $ anaemia                  : int  0 0 0 1 1 1 1 1 0 1 ...
##  $ creatinine_phosphokinase: int  582 7861 146 111 160 47 246 315 157 123
...
##  $ diabetes                 : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ ejection_fraction        : int  20 38 20 20 20 40 15 60 65 35 ...
##  $ high_blood_pressure      : int  1 0 0 0 0 1 0 0 0 1 ...
##  $ platelets                : num  265000 263358 162000 210000 327000 ...
##  $ serum_creatinine         : num  1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4
...
##  $ serum_sodium             : int  130 136 129 137 116 132 137 131 138 133
...
##  $ sex                      : Factor w/ 2 levels "Female","male": 2 2 2 2 1
2 2 2 1 2 ...
##  $ smoking                  : int  0 0 1 0 0 1 0 1 0 1 ...
##  $ time                     : int  4 6 7 7 8 8 10 10 10 10 ...
##  $ DEATH_EVENT              : Factor w/ 2 levels "Death","No Death": 2 2 2
2 2 2 2 2 2 2 ...
```

```r
summary(rawdata)
```

```
##       age           anaemia       creatinine_phosphokinase
##  Min.   :40.00   Min.   :0.0000   Min.   :  23.0
##  1st Qu.:51.00   1st Qu.:0.0000   1st Qu.: 116.5
##  Median :60.00   Median :0.0000   Median : 250.0
##  Mean   :60.83   Mean   :0.4314   Mean   : 581.8
##  3rd Qu.:70.00   3rd Qu.:1.0000   3rd Qu.: 582.0
##  Max.   :95.00   Max.   :1.0000   Max.   :7861.0
##     diabetes      ejection_fraction high_blood_pressure   platelets
##  Min.   :0.0000   Min.   :14.00     Min.   :0.0000      Min.   : 25100
##  1st Qu.:0.0000   1st Qu.:30.00     1st Qu.:0.0000      1st Qu.:212500
##  Median :0.0000   Median :38.00     Median :0.0000      Median :262000
##  Mean   :0.4181   Mean   :38.08     Mean   :0.3512      Mean   :263358
##  3rd Qu.:1.0000   3rd Qu.:45.00     3rd Qu.:1.0000      3rd Qu.:303500
##  Max.   :1.0000   Max.   :80.00     Max.   :1.0000      Max.   :850000
##  serum_creatinine  serum_sodium       sex         smoking
##  Min.   :0.500    Min.   :113.0   Female:105   Min.   :0.0000
##  1st Qu.:0.900    1st Qu.:134.0   male  :194   1st Qu.:0.0000
##  Median :1.100    Median :137.0                Median :0.0000
##  Mean   :1.394    Mean   :136.6                Mean   :0.3211
##  3rd Qu.:1.400    3rd Qu.:140.0                3rd Qu.:1.0000
##  Max.   :9.400    Max.   :148.0                Max.   :1.0000
##       time         DEATH_EVENT
##  Min.   :  4.0   Death   :203
##  1st Qu.: 73.0   No Death: 96
##  Median :115.0
##  Mean   :130.3
##  3rd Qu.:203.0
##  Max.   :285.0
```

```r
head(rawdata)
```

```
##   age anaemia creatinine_phosphokinase diabetes ejection_fraction
## 1  75       0                     582        0                20
## 2  55       0                    7861        0                38
## 3  65       0                     146        0                20
## 4  50       1                     111        0                20
## 5  65       1                     160        1                20
## 6  90       1                      47        0                40
##   high_blood_pressure platelets serum_creatinine serum_sodium    sex
## 1                   1    265000              1.9          130   male
## 2                   0    263358              1.1          136   male
## 3                   0    162000              1.3          129   male
## 4                   0    210000              1.9          137   male
## 5                   0    327000              2.7          116 Female
## 6                   1    204000              2.1          132   male
##   smoking time DEATH_EVENT
## 1       0    4    No Death
## 2       0    6    No Death
```

```
## 3       1    7    No Death
## 4       0    7    No Death
## 5       0    8    No Death
## 6       1    8    No Death
```

```
dim(rawdata)
```

```
## [1] 299  13
```

```
#Data Cleaning

#Checking for missing values
is.null(rawdata)
```

```
## [1] FALSE
```

```
##The "FALSE" output shows there is no missing data in the dataset.


data <- as.matrix(rawdata[,c(1,3,5,7,8,9,12)])
dim(rawdata)
```

```
## [1] 299  13
```

```
dim(data)
```

```
## [1] 299   7
```

```
dataset <- cbind(data, as.numeric(rawdata$DEATH_EVENT)-1)
dim(dataset)
```

```
## [1] 299   8
```

```
colnames(dataset)[8] <- "death"

View(dataset)
str(dataset)
```

```
##  num [1:299, 1:8] 75 55 65 50 65 90 75 60 65 80 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:8] "age" "creatinine_phosphokinase" "ejection_fraction"
"platelets" ...
```

```
# Lets cut the data into two parts
smp_size_raw <- floor(0.75 * nrow(dataset))
train_ind_raw <- sample(nrow(dataset), size = smp_size_raw)
train_raw.df <- as.data.frame(dataset[train_ind_raw, ])
test_raw.df <- as.data.frame(dataset[-train_ind_raw, ])
# We have a training and a test set. Training is 75% and test is 25%
```

```r
dataset.lda <- lda(formula = train_raw.df$death ~ ., data = train_raw.df)
dataset.lda
```

```
## Call:
## lda(train_raw.df$death ~ ., data = train_raw.df)
##
## Prior probabilities of groups:
##         0         1
## 0.6696429 0.3303571
##
## Group means:
##        age creatinine_phosphokinase ejection_fraction platelets
## 0 58.51111                 524.4333          41.07333    274711
## 1 65.40091                 577.7027          32.78378    264309
##   serum_creatinine serum_sodium      time
## 0         1.175133     137.1267 154.18000
## 1         1.825811     135.2838  68.78378
##
## Coefficients of linear discriminants:
##                                    LD1
## age                         2.099868e-02
## creatinine_phosphokinase    7.198031e-05
## ejection_fraction          -4.598699e-02
## platelets                   2.220684e-07
## serum_creatinine            3.152938e-01
## serum_sodium               -2.596675e-02
## time                       -1.193570e-02
```

```r
summary(dataset.lda)
```

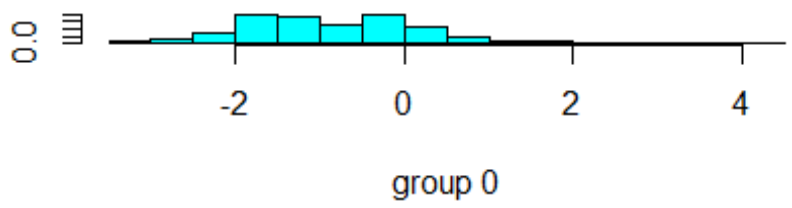
```
##         Length Class  Mode
## prior    2     -none- numeric
## counts   2     -none- numeric
## means   14     -none- numeric
## scaling  7     -none- numeric
## lev      2     -none- character
## svd      1     -none- numeric
## N        1     -none- numeric
## call     3     -none- call
## terms    3     terms  call
## xlevels  0     -none- list
```

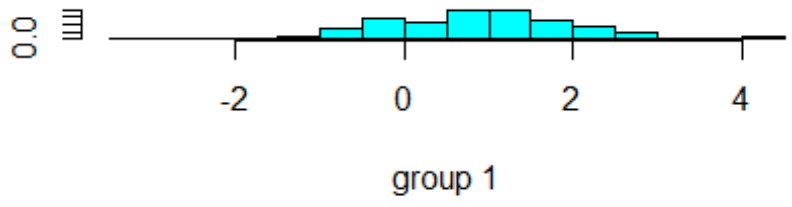
```r
print(dataset.lda)
```

```
## Call:
## lda(train_raw.df$death ~ ., data = train_raw.df)
##
## Prior probabilities of groups:
##         0         1
## 0.6696429 0.3303571
##
```

```
## Group means:
##        age creatinine_phosphokinase ejection_fraction platelets
## 0 58.51111                 524.4333          41.07333    274711
## 1 65.40091                 577.7027          32.78378    264309
##   serum_creatinine serum_sodium      time
## 0         1.175133     137.1267 154.18000
## 1         1.825811     135.2838  68.78378
##
## Coefficients of linear discriminants:
##                                    LD1
## age                         2.099868e-02
## creatinine_phosphokinase  7.198031e-05
## ejection_fraction        -4.598699e-02
## platelets                  2.220684e-07
## serum_creatinine           3.152938e-01
## serum_sodium              -2.596675e-02
## time                      -1.193570e-02
```

```
plot(dataset.lda)
```





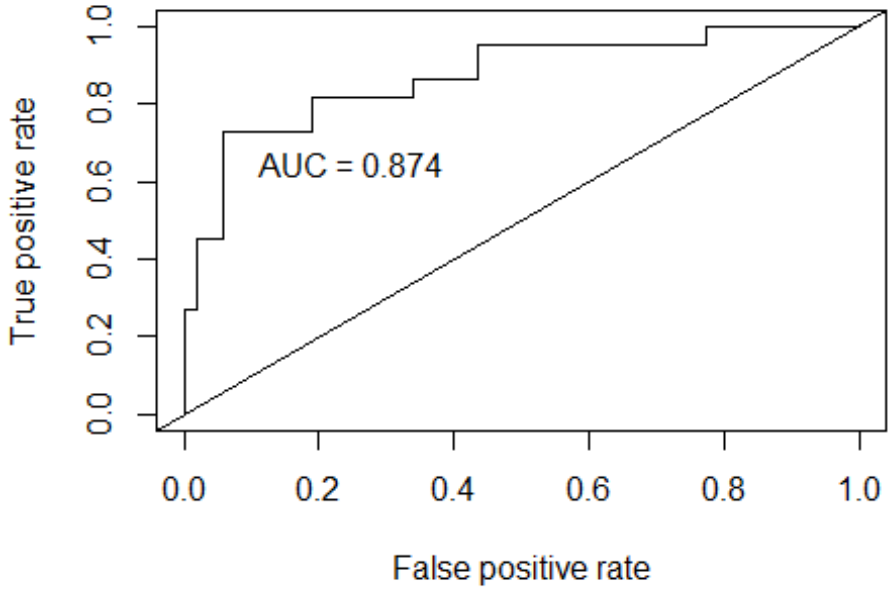

```
dataset.lda.predict <- predict(dataset.lda, newdata = test_raw.df)
dataset.lda.predict$class
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 1 0 0 0 1 1 1 0 0 1 0 1 0 0 0 0 0
## [36] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [71] 0 0 0 0 0
## Levels: 0 1
```

```r
View(dataset.lda.predict)
dataset.lda.predict$x
```

```
##            LD1
## 1   1.77283231
## 2   1.32158942
## 3   1.18085883
## 4   1.79210926
## 5   1.97850321
## 6   1.18386128
## 7   1.19499836
## 8   1.50028178
## 9   1.17551727
## 10  1.57263752
## 11  1.31404519
## 12  1.84632784
## 13  0.62829563
## 14  1.33248189
## 15 -0.08220599
## 16  0.50167386
## 17  1.67836521
## 18  0.10520202
## 19  1.12624607
## 20 -0.05581531
## 21  0.43100048
## 22 -0.94366567
## 23  1.20367843
## 24  0.99582475
## 25  0.87891896
## 26 -0.42548970
## 27 -0.86757743
## 28  1.56579281
## 29 -0.20059261
## 30  1.10428398
## 31 -0.03733506
## 32  0.12002184
## 33  0.41892436
## 34  0.63568386
## 35  0.32532663
## 36  0.08670294
## 37 -0.11959639
## 38  0.38097103
## 39  1.00267756
## 40 -0.31447150
## 41  0.05353679
## 42 -0.21252017
## 43 -1.51949666
## 44 -0.26084694
## 45 -1.16251832
## 46  0.49900431
```

```
## 47 -0.70681931
## 48 -1.37787276
## 49 -1.38300714
## 50 -0.08273190
## 51 -0.53465769
## 52 -1.69099749
## 53 -1.00363963
## 54 -0.70984865
## 55 -1.55468559
## 56 -0.57211393
## 57 -1.09491722
## 58 -1.05918353
## 59 -1.21633471
## 60 -1.14937040
## 61 -1.68649907
## 62 -1.44410078
## 63 -0.40172670
## 64 -1.54662511
## 65 -1.78830983
## 66 -2.25248497
## 67 -1.45747720
## 68 -1.31497705
## 69 -1.71142594
## 70 -1.52057671
## 71 -1.26038873
## 72 -1.54707427
## 73 -1.86985773
## 74 -2.13635845
## 75 -2.34897960

# Get the posteriors as a dataframe.
dataset.lda.predict.posteriors <-
as.data.frame(dataset.lda.predict$posterior)

#create ROC/AUC curve
pred <- prediction(dataset.lda.predict.posteriors[,2], test_raw.df$death)
roc.perf = performance(pred, measure = "tpr", x.measure = "fpr")
auc.train <- performance(pred, measure = "auc")
auc.train <- auc.train@y.values
plot(roc.perf)
abline(a=0, b= 1)
text(x = .25, y = .65 ,paste("AUC = ", round(auc.train[[1]],3), sep = ""))
```




```
#Since the AUC value 0.82 is close to 1.0, we can say, it is a good model

summary(dataset.lda)

##         Length Class  Mode
## prior    2     -none- numeric
## counts   2     -none- numeric
## means   14     -none- numeric
## scaling  7     -none- numeric
## lev      2     -none- character
## svd      1     -none- numeric
## N        1     -none- numeric
## call     3     -none- call
## terms    3     terms  call
## xlevels  0     -none- list

(prop = dataset.lda$svd^2/sum(dataset.lda$svd^2))

## [1] 1

#we can use the singular values to compute the amount of the between-group
variance that is explained by each linear discriminant.
#We see that the first linear discriminant (dataset.lda) explains almost all
of the between-group variance in the dataset

data2<- as.data.frame(dataset)
View(data2)
```

```
dataset.lda2 <- lda(formula = death ~ ., data = data2, CV = TRUE)
dataset.lda2
```

```
## $class
##   [1] 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1
0
##  [36] 1 1 0 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 0 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1
1
##  [71] 0 0 1 0 1 1 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 1 1
0
## [106] 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 1 0
0
## [141] 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0
## [176] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0
## [211] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0
## [246] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0
## [281] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## Levels: 0 1
##
## $posterior
##                0           1
## 1   0.0179741183 0.9820258817
## 2   0.0818920058 0.9181079942
## 3   0.0438234013 0.9561765987
## 4   0.0900721530 0.9099278470
## 5   0.0078978522 0.9921021478
## 6   0.0495485966 0.9504514034
## 7   0.0359811561 0.9640188439
## 8   0.6479695126 0.3520304874
## 9   0.6995478474 0.3004521526
## 10  0.0001731388 0.9998268612
## 11  0.0237996127 0.9762003873
## 12  0.1737865410 0.8262134590
## 13  0.2761567756 0.7238432244
## 14  0.4112232438 0.5887767562
## 15  0.2944061933 0.7055938067
## 16  0.2560491258 0.7439508742
## 17  0.1771974576 0.8228025424
## 18  0.0844280524 0.9155719476
## 19  0.1370342377 0.8629657623
## 20  0.4048626531 0.5951373469
## 21  0.1081968278 0.8918031722
## 22  0.1419476488 0.8580523512
## 23  0.2975498733 0.7024501267
## 24  0.7957060295 0.2042939705
## 25  0.0716034594 0.9283965406
```

```
## 26  0.1837559784 0.8162440216
## 27  0.1540106078 0.8459893922
## 28  0.3591955618 0.6408044382
## 29  0.0235606150 0.9764393850
## 30  0.0848719425 0.9151280575
## 31  0.0645559940 0.9354440060
## 32  0.0773591483 0.9226408517
## 33  0.3355171612 0.6644828388
## 34  0.3342967284 0.6657032716
## 35  0.6284449498 0.3715550502
## 36  0.0517658579 0.9482341421
## 37  0.3012797349 0.6987202651
## 38  0.5100966908 0.4899033092
## 39  0.0669054479 0.9330945521
## 40  0.2105032665 0.7894967335
## 41  0.0514107856 0.9485892144
## 42  0.3407811544 0.6592188456
## 43  0.4216099380 0.5783900620
## 44  0.4516074089 0.5483925911
## 45  0.8279629449 0.1720370551
## 46  0.3510097399 0.6489902601
## 47  0.1994406652 0.8005593348
## 48  0.5517036015 0.4482963985
## 49  0.0067972052 0.9932027948
## 50  0.4311130020 0.5688869980
## 51  0.1928818194 0.8071181806
## 52  0.2464312761 0.7535687239
## 53  0.0929493327 0.9070506673
## 54  0.5781551848 0.4218448152
## 55  0.2381177386 0.7618822614
## 56  0.0512207566 0.9487792434
## 57  0.1612050422 0.8387949578
## 58  0.6105366907 0.3894633093
## 59  0.2853939248 0.7146060752
## 60  0.1329823221 0.8670176779
## 61  0.2196064148 0.7803935852
## 62  0.4009659385 0.5990340615
## 63  0.5714771335 0.4285228665
## 64  0.7838333229 0.2161666771
## 65  0.9781299059 0.0218700941
## 66  0.0553625134 0.9446374866
## 67  0.3148673593 0.6851326407
## 68  0.2991800021 0.7008199979
## 69  0.3167386599 0.6832613401
## 70  0.2204277321 0.7795722679
## 71  0.8436355331 0.1563644669
## 72  0.5855824149 0.4144175851
## 73  0.0911759526 0.9088240474
## 74  0.7092078756 0.2907921244
## 75  0.1733953052 0.8266046948
```

```
## 76  0.4014011531 0.5985988469
## 77  0.8869095071 0.1130904929
## 78  0.8141266733 0.1858733267
## 79  0.4349613816 0.5650386184
## 80  0.8151002204 0.1848997796
## 81  0.4460590743 0.5539409257
## 82  0.7180389749 0.2819610251
## 83  0.1665911237 0.8334088763
## 84  0.4654653211 0.5345346789
## 85  0.5019391842 0.4980608158
## 86  0.9120458636 0.0879541364
## 87  0.6351321949 0.3648678051
## 88  0.9156792617 0.0843207383
## 89  0.8557839420 0.1442160580
## 90  0.5233252900 0.4766747100
## 91  0.6977813065 0.3022186935
## 92  0.6784926566 0.3215073434
## 93  0.9447273066 0.0552726934
## 94  0.3073240315 0.6926759685
## 95  0.7910902946 0.2089097054
## 96  0.9340809068 0.0659190932
## 97  0.2992716559 0.7007283441
## 98  0.8622578668 0.1377421332
## 99  0.4207813828 0.5792186172
## 100 0.7185316381 0.2814683619
## 101 0.4415035014 0.5584964986
## 102 0.6120615389 0.3879384611
## 103 0.2699941283 0.7300058717
## 104 0.3853794560 0.6146205440
## 105 0.7695612352 0.2304387648
## 106 0.4073064625 0.5926935375
## 107 0.7710234455 0.2289765545
## 108 0.7114043816 0.2885956184
## 109 0.5511150431 0.4488849569
## 110 0.7656266992 0.2343733008
## 111 0.7698229954 0.2301770046
## 112 0.6508287223 0.3491712777
## 113 0.4513454377 0.5486545623
## 114 0.8855008713 0.1144991287
## 115 0.5052502157 0.4947497843
## 116 0.7597527787 0.2402472213
## 117 0.9380953874 0.0619046126
## 118 0.5065318068 0.4934681932
## 119 0.9303197912 0.0696802088
## 120 0.2900197077 0.7099802923
## 121 0.8762469195 0.1237530805
## 122 0.6562349108 0.3437650892
## 123 0.7913926174 0.2086073826
## 124 0.6769816452 0.3230183548
## 125 0.2943767018 0.7056232982
```

```
## 126 0.9024941057 0.0975058943
## 127 0.2197899184 0.7802100816
## 128 0.9524498008 0.0475501992
## 129 0.6584631712 0.3415368288
## 130 0.5007868714 0.4992131286
## 131 0.9503428972 0.0496571028
## 132 0.0594016209 0.9405983791
## 133 0.8468115465 0.1531884535
## 134 0.9528408304 0.0471591696
## 135 0.1850029100 0.8149970900
## 136 0.6260130965 0.3739869035
## 137 0.9360457533 0.0639542467
## 138 0.2019130396 0.7980869604
## 139 0.7005363271 0.2994636729
## 140 0.7116072544 0.2883927456
## 141 0.5202886300 0.4797113700
## 142 0.8679471830 0.1320528170
## 143 0.6880012832 0.3119987168
## 144 0.8481525558 0.1518474442
## 145 0.3552152762 0.6447847238
## 146 0.8324205891 0.1675794109
## 147 0.7709292332 0.2290707668
## 148 0.9180917356 0.0819082644
## 149 0.3374567274 0.6625432726
## 150 0.6506772918 0.3493227082
## 151 0.5780003874 0.4219996126
## 152 0.9602812631 0.0397187369
## 153 0.9054540578 0.0945459422
## 154 0.7571257986 0.2428742014
## 155 0.7499123386 0.2500876614
## 156 0.5519648779 0.4480351221
## 157 0.7911794274 0.2088205726
## 158 0.7071806788 0.2928193212
## 159 0.6721506882 0.3278493118
## 160 0.8985860876 0.1014139124
## 161 0.7544045486 0.2455954514
## 162 0.8970084114 0.1029915886
## 163 0.8313440672 0.1686559328
## 164 0.8613763087 0.1386236913
## 165 0.8012029594 0.1987970406
## 166 0.6090743099 0.3909256901
## 167 0.9744315100 0.0255684900
## 168 0.3749285278 0.6250714722
## 169 0.8616927583 0.1383072417
## 170 0.7522325223 0.2477674777
## 171 0.8456721196 0.1543278804
## 172 0.8555584637 0.1444415363
## 173 0.9744217305 0.0255782695
## 174 0.7978700455 0.2021299545
## 175 0.8444181228 0.1555818772
```

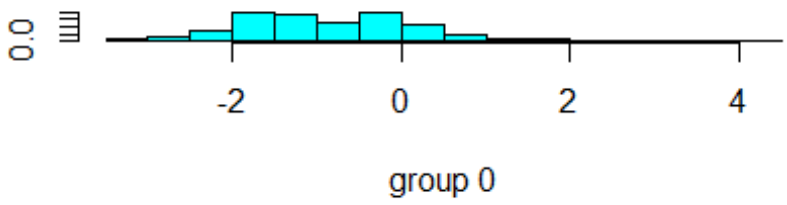
```
## 176 0.9775656146 0.0224343854
## 177 0.7896187178 0.2103812822
## 178 0.9730222579 0.0269777421
## 179 0.9809357386 0.0190642614
## 180 0.9344377311 0.0655622689
## 181 0.9075614793 0.0924385207
## 182 0.7699686000 0.2300314000
## 183 0.7022155808 0.2977844192
## 184 0.6577454496 0.3422545504
## 185 0.8350403132 0.1649596868
## 186 0.8486675651 0.1513324349
## 187 0.9836070238 0.0163929762
## 188 0.7555848625 0.2444151375
## 189 0.9336095181 0.0663904819
## 190 0.9859017641 0.0140982359
## 191 0.6324127477 0.3675872523
## 192 0.9753070015 0.0246929985
## 193 0.9527811974 0.0472188026
## 194 0.8511080740 0.1488919260
## 195 0.8263895236 0.1736104764
## 196 0.9350839623 0.0649160377
## 197 0.9639865589 0.0360134411
## 198 0.9079292916 0.0920707084
## 199 0.8454005263 0.1545994737
## 200 0.5016117565 0.4983882435
## 201 0.9526571582 0.0473428418
## 202 0.9940657110 0.0059342890
## 203 0.9856193717 0.0143806283
## 204 0.5448776639 0.4551223361
## 205 0.9255989422 0.0744010578
## 206 0.9787164294 0.0212835706
## 207 0.9866608404 0.0133391596
## 208 0.8571001357 0.1428998643
## 209 0.9414641629 0.0585358371
## 210 0.9341447866 0.0658552134
## 211 0.7072253502 0.2927746498
## 212 0.9956254562 0.0043745438
## 213 0.9515033744 0.0484966256
## 214 0.9073656272 0.0926343728
## 215 0.9305733762 0.0694266238
## 216 0.8577356263 0.1422643737
## 217 0.9746651353 0.0253348647
## 218 0.8393664987 0.1606335013
## 219 0.8910507507 0.1089492493
## 220 0.9687978296 0.0312021704
## 221 0.6320548773 0.3679451227
## 222 0.9864512329 0.0135487671
## 223 0.9795485129 0.0204514871
## 224 0.9303742255 0.0696257745
## 225 0.9192786249 0.0807213751
```

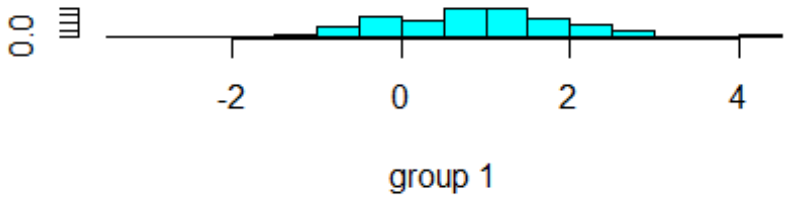
```
## 226 0.9573609923 0.0426390077
## 227 0.8707680972 0.1292319028
## 228 0.9410644577 0.0589355423
## 229 0.2378807811 0.7621192189
## 230 0.8219283103 0.1780716897
## 231 0.8626109698 0.1373890302
## 232 0.9205132600 0.0794867400
## 233 0.9834681811 0.0165318189
## 234 0.9644300105 0.0355699895
## 235 0.9826669891 0.0173330109
## 236 0.9716732475 0.0283267525
## 237 0.9861211540 0.0138788460
## 238 0.8681099855 0.1318900145
## 239 0.9583214393 0.0416785607
## 240 0.9825174111 0.0174825889
## 241 0.9430699104 0.0569300896
## 242 0.9066555794 0.0933444206
## 243 0.9813522995 0.0186477005
## 244 0.9568365679 0.0431634321
## 245 0.9478161233 0.0521838767
## 246 0.9603449460 0.0396550540
## 247 0.9208437091 0.0791562909
## 248 0.7838893776 0.2161106224
## 249 0.9870006142 0.0129993858
## 250 0.9657050314 0.0342949686
## 251 0.9616396017 0.0383603983
## 252 0.9800085608 0.0199914392
## 253 0.9850410320 0.0149589680
## 254 0.9185694745 0.0814305255
## 255 0.9959470671 0.0040529329
## 256 0.9735660505 0.0264339495
## 257 0.9539034748 0.0460965252
## 258 0.9827115384 0.0172884616
## 259 0.9714533601 0.0285466399
## 260 0.9942952131 0.0057047869
## 261 0.9868455406 0.0131544594
## 262 0.9778359422 0.0221640578
## 263 0.8951768541 0.1048231459
## 264 0.9957505117 0.0042494883
## 265 0.9802195378 0.0197804622
## 266 0.9890130746 0.0109869254
## 267 0.8918317008 0.1081682992
## 268 0.9816230977 0.0183769023
## 269 0.9928248084 0.0071751916
## 270 0.9870081265 0.0129918735
## 271 0.9632881503 0.0367118497
## 272 0.9880764803 0.0119235197
## 273 0.9687555046 0.0312444954
## 274 0.9957991830 0.0042008170
## 275 0.9704102363 0.0295897637
```

```
## 276 0.9924543896 0.0075456104
## 277 0.9745783511 0.0254216489
## 278 0.9751537120 0.0248462880
## 279 0.9802495084 0.0197504916
## 280 0.9853134108 0.0146865892
## 281 0.9626822211 0.0373177789
## 282 0.9288911886 0.0711088114
## 283 0.8736427610 0.1263572390
## 284 0.9745992507 0.0254007493
## 285 0.9947810570 0.0052189430
## 286 0.9876192662 0.0123807338
## 287 0.9765987237 0.0234012763
## 288 0.9973408493 0.0026591507
## 289 0.9821014319 0.0178985681
## 290 0.9731778282 0.0268221718
## 291 0.9986496566 0.0013503434
## 292 0.9815515499 0.0184484501
## 293 0.9932623187 0.0067376813
## 294 0.9868891226 0.0131108774
## 295 0.9926371630 0.0073628370
## 296 0.9889101553 0.0110898447
## 297 0.9993780933 0.0006219067
## 298 0.9923538772 0.0076461228
## 299 0.9960970251 0.0039029749
##
## $terms
## death ~ age + creatinine_phosphokinase + ejection_fraction +
##     platelets + serum_creatinine + serum_sodium + time
## attr(,"variables")
## list(death, age, creatinine_phosphokinase, ejection_fraction,
##     platelets, serum_creatinine, serum_sodium, time)
## attr(,"factors")
##                          age creatinine_phosphokinase ejection_fraction
## death                      0                        0                 0
## age                        1                        0                 0
## creatinine_phosphokinase   0                        1                 0
## ejection_fraction          0                        0                 1
## platelets                  0                        0                 0
## serum_creatinine           0                        0                 0
## serum_sodium               0                        0                 0
## time                       0                        0                 0
##                          platelets serum_creatinine serum_sodium time
## death                            0                0            0    0
## age                              0                0            0    0
## creatinine_phosphokinase         0                0            0    0
## ejection_fraction                0                0            0    0
## platelets                        1                0            0    0
## serum_creatinine                 0                1            0    0
## serum_sodium                     0                0            1    0
## time                             0                0            0    1
```

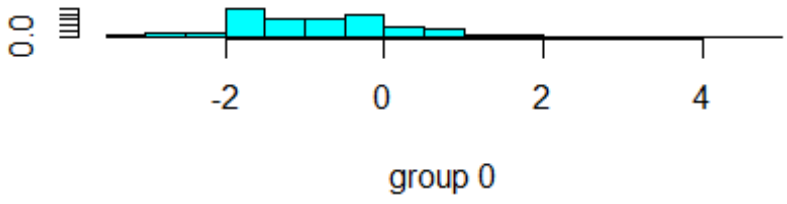
```
## attr(,"term.labels")
## [1] "age"                       "creatinine_phosphokinase"
## [3] "ejection_fraction"         "platelets"
## [5] "serum_creatinine"          "serum_sodium"
## [7] "time"
## attr(,"order")
## [1] 1 1 1 1 1 1 1
## attr(,"intercept")
## [1] 1
## attr(,"response")
## [1] 1
## attr(,".Environment")
## <environment: R_GlobalEnv>
## attr(,"predvars")
## list(death, age, creatinine_phosphokinase, ejection_fraction,
##     platelets, serum_creatinine, serum_sodium, time)
## attr(,"dataClasses")
##                   death                     age creatinine_phosphokinase
##               "numeric"               "numeric"                 "numeric"
##        ejection_fraction               platelets         serum_creatinine
##               "numeric"               "numeric"                 "numeric"
##             serum_sodium                    time
##               "numeric"               "numeric"
##
## $call
## lda(formula = death ~ ., data = data2, CV = TRUE)
##
## $xlevels
## named list()

head(dataset.lda2$class)

## [1] 1 1 1 1 1 1
## Levels: 0 1

#the Maximum a Posteriori Probability (MAP) classification (a factor)
#posterior: posterior probabilities for the classes.
head(dataset.lda2$posterior, 3)

##           0         1
## 1 0.01797412 0.9820259
## 2 0.08189201 0.9181080
## 3 0.04382340 0.9561766

#Dividing the data into two parts 50% each
train <- sample(1:299, 150)

# training model
dataset.lda3 <- lda(death ~ .,
          data2,
          prior = c(1,1)/2,
```

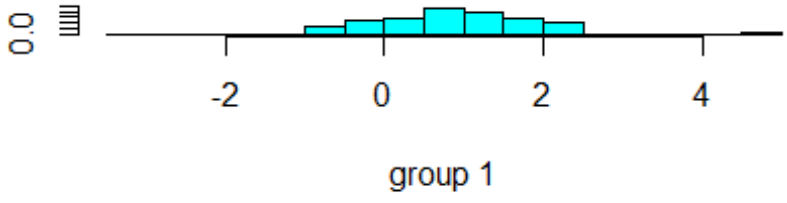
```
          subset = train)

# predictions
plda = predict(object = dataset.lda3,
               newdata = data2[-train, ])
head(plda$class)

## [1] 1 1 1 0 1 1
## Levels: 0 1

# posterior prob
head(plda$posterior, 6)

##             0         1
## 3  0.03260689 0.9673931
## 4  0.03598577 0.9640142
## 7  0.01966265 0.9803374
## 8  0.61393675 0.3860633
## 18 0.04882990 0.9511701
## 20 0.35848051 0.6415195

head(plda$x, 3)

##        LD1
## 3 1.876900
## 4 1.820374
## 7 2.164294

plot(dataset.lda)
```






```
plot(dataset.lda3)
```

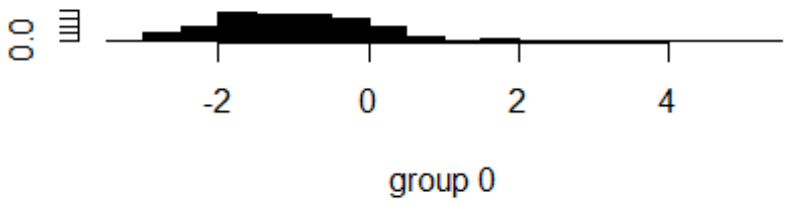


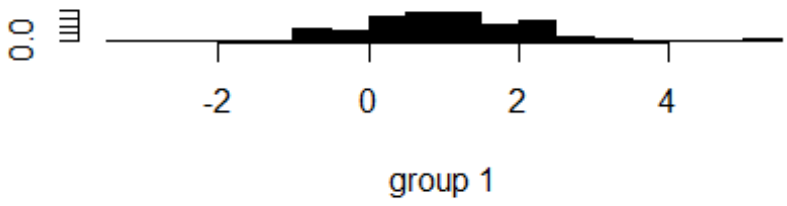


```r
dataset.lda <- lda(death ~ .,
         data2,
         prior = c(1,1)/2)
prop.lda = dataset.lda$svd^2/sum(dataset.lda$svd^2)
plda <- predict(object = dataset.lda,
                newdata = data2)
data3 = data.frame(death = data2[,"death"],lda = plda$x)


# Dividing the dataset using second approach

set.seed(101) # Nothing is random!!
sample_n(data2,10)
```

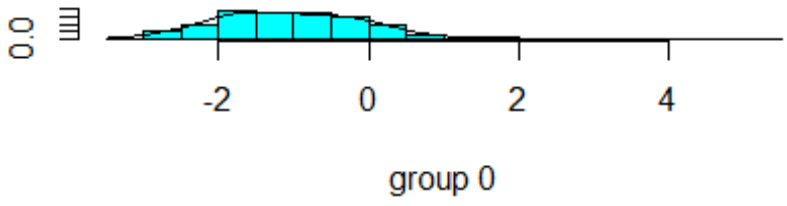
```
##    age creatinine_phosphokinase ejection_fraction platelets
## 1   58                      144                38    327000
## 2   60                     2281                40    283000
## 3   61                       80                38    282000
## 4   58                      200                60    300000
## 5   53                     1808                60    249000
## 6   45                      582                55    543000
## 7   65                      157                65    263358
## 8   73                     1185                40    220000
## 9   85                      910                50    235000
## 10  50                      196                45    395000
##    serum_creatinine serum_sodium time death
## 1               0.7          142   83     0
## 2               1.0          141  187     0
## 3               1.4          137  213     0
## 4               0.8          137  104     0
## 5               0.7          138  106     0
## 6               1.0          132  250     0
## 7               1.5          138   10     1
## 8               0.9          141  213     0
## 9               1.3          134  121     0
## 10              1.6          136  285     0
```

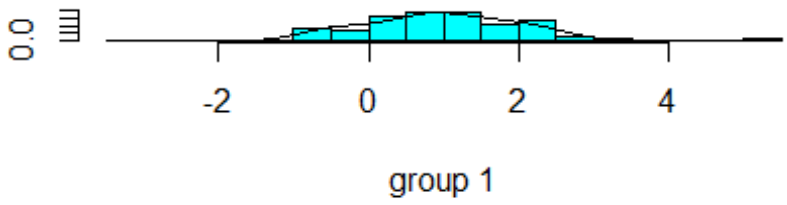
```r
# Divinding dataset into 75/25 using Dplyr preserves class
training_sample <- sample(c(TRUE, FALSE), nrow(data2), replace = T, prob =
c(0.75,0.25))
train <- data2[training_sample, ]
test <- data2[!training_sample, ]

#lets run LDA like before
lda.data2 <- lda(death ~ ., train)

# Plotting the data to better understand the model
plot(lda.data2, col = as.integer(train$death))
```



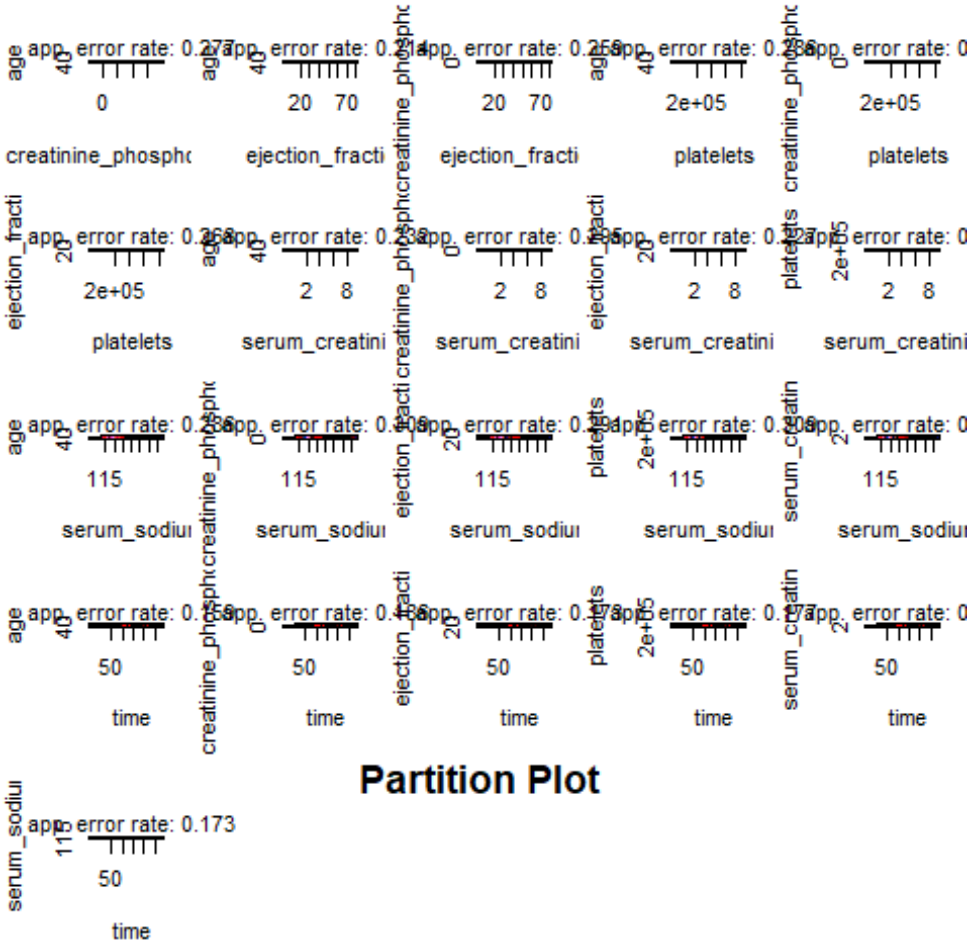



```r
plot(lda.data2, dimen = 1, type = "b")
```





```r
View(data2)
# Partition plots
partimat(death ~ age + creatinine_phosphokinase + ejection_fraction +
platelets + serum_creatinine + serum_sodium + time, data=train, method="lda")
```



```r
# Checking the accuracy for training set to understand how good the model is
lda.train <- predict(lda.data2)
train$lda <- lda.train$class
table(train$lda,train$death)
```

```
##
##       0   1
##   0 141  19
##   1   9  51
```

```r
#Checking the accuracy for testing set to understand how good the model is
lda.test <- predict(lda.data2,test)
test$lda <- lda.test$class
table(test$lda,test$death)
```

```
##
##      0  1
##   0 39  6
##   1 14 20
```