Lab 4 report

Summary:

Random forest:

Tune ntree from 1 to 200, find best ntree=41;

```
pred   0   1
   0 220  42
   1  18  21

Accuracy:
 [1] 0.8006645
```

Adaboost. M1 with decision stump

Tune cp

cp=0.005

```
> table.ab

      0   1
  0 225  38
  1  13  25
> accuracy.ab=sum(diag(table.ab))/sum(table.ab)
> accuracy.ab
[1] 0.8305648
```

cp=0.01

```
> table.ab

      0   1
  0 225  37
  1  13  26
> accuracy.ab=sum(diag(table.ab))/sum(table.ab)
> accuracy.ab
[1] 0.833887
```

cp =0.05

```
> table.ab

      0   1
  0 227  40
```

```
  1  11  23

> accuracy.ab
[1] 0.8305648
```

Neura network(NN) model

Tune decay

Decay= 0.0001

```
> table.nn

predicted.nn   0   1
           0 226  46
           1  12  17
> accucary.nn=sum(diag(table.nn))/sum(table.nn)
> accucary.nn
[1] 0.807309
```

Decay= 0.001

```
> table.nn

predicted.nn   0   1
           0 227  40
           1  11  23
> accucary.nn=sum(diag(table.nn))/sum(table.nn)
> accucary.nn
[1] 0.8305648
```

decay=0.005

```
 > table.nn

 predicted.nn   0   1
            0 229  48
            1   9  15
 > accucary.nn=sum(diag(table.nn))/sum(table.nn)
 > accucary.nn
 [1] 0.8106312
```

Logistic regression
Variable selection:
Find out feature M is not useful in this model , thus, deliminate it and fit again.

```
> table.lr

predicted.lr   0   1
           0 226  42
           1  12  21
```

```
> accuracy.lr=sum(diag(table.lr))/sum(table.lr)
> accuracy.lr
[1] 0.820598
```

KNN

```
> table.knn

predicted.knn   0   1
            0 223  47
            1  15  16
> accuracy.knn=sum(diag(table.knn))/sum(table.knn)
> accuracy.knn
[1] 0.7940199
```

Naïve Bayes

```
> table.nb

predicted.nb   0   1
           0 226  44
           1  12  19
> accuracy.nb=sum(diag(table.nb))/sum(table.nb)
> accuracy.nb
[1] 0.8139535
```

Decision tree

```
> #Import Library
> library(rpart)
> #grow tree
> model.dt <- rpart(Class~., data = train2,method="class")
> predicted.dt=predict(model.dt,test2[,1:4], type = 'class')
> table.dt=table(predicted.dt, test2$Class)
> table.dt

predicted.dt   0   1
           0 223  38
           1  15  25
> accuracy.dt=sum(diag(table.dt))/sum(table.dt)
> accuracy.dt
[1] 0.8239203


>
```

Ensemble /unweighted /5

```
> table.ensemble.uw

predicted.ensemble.uw   0   1
                    0 229  43
                    1   9  20
```

```
> accucary.ensemble.uw=sum(diag(table.ensemble.uw))/sum(table.ensemble.uw)
> accucary.ensemble.uw
[1] 0.8272425


>
```

Ensemble /weighted /5

```
weight1=c(1,3,5,7,9)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 221  38
                   1  17  25
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8172757
```

```
weight2=c(5,4,5,4,4)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 224  36
                   1  14  27
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.833887
```

```
weight3=c(accuracy.lr/(1-accuracy.lr),accuracy.knn/(1-accuracy.knn),accuracy.dt/(1-accuracy.dt),accuracy.nb/(1-accuracy.nb),accucary.nn/(1-accucary.nn))
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 224  34
                   1  14  29
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8405316
```

Ensemble /unweighted /7

```
> table.ensemble.uw

predicted.ensemble.uw   0   1
                    0 229  45
                    1   9  18
> accucary.ensemble.uw=sum(diag(table.ensemble.uw))/sum(table.ensemble.uw)
> accucary.ensemble.uw
[1] 0.820598
```

Ensemble /weighted /7

weight4=c(1,3,5,7,9,11,13)

```
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 216  35
                   1  22  28
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8106312
```

weight5=c(5,4,5,4,4,5,4)

```
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 225  33
                   1  13  30
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8471761
```

weight6=c(accuracy.lr/(1-accuracy.lr),accuracy.knn/(1-accuracy.knn),accuracy.dt/(1-accuracy.dt),accuracy.nb/(1-accuracy.nb),
accucary.nn/(1-accucary.nn),accucary.rf/(1-accucary.rf),accucary.ab/(1-accucary.ab))

```
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 221  30
                   1  17  33
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8438538
```

Discussion:

	accuacy
random forest	0.8006645
AdaBoost.M1(decision stump)	0.833887
Neural Network	0.8305648
KNN	0.7940199
NB	0.8139535
Logistic regression	0.820598
decision tree	0.8239203
ensemble/unweighted /5	0.8272425
ensemble/weighted /5	0.8172757
ensemble/weighted /5	0.833887
ensemble/weighted /5	0.8405316
ensemble/unweighted /7	0.820598
ensemble/weighted /7	0.8106312

ensemble/weighted /7	0.8471761
ensemble/weighted /7	0.8438538

From above,
1. we can get better accuracy by tuneing parameters in the base classifier like rf, nn, adaboost…
2. adaBoost has higher accuracy than other single classiers like rf, nn, knn…
3. ensemble classifiers generally have Higher accuracy than other single classifiers.
4. weighted ensemble, it seems to have better accuracy than unweighted ones in general, and If we cho ose appropriate weight vector, the accuracy will be much higher than other classifiers.
5. it seems the optimal weight vector in weight ensemble is accuracy.x/(1-accuracy.x), where accuracy.x is the accuracy of the single classfier x.

Output

1.random forest

```
test=read.csv("C:\\Users\\shuzhang\\Downloads\\lab4-test.csv")

train1=train
train1[,1:4]=train1[,1:4]/max(train1[,1:4])
test1=test
test1[,1:4]=test1[,1:4]/max(test1[,1:4])

train2=train1
train2$Class=as.factor(train2$Class)
test2=test1
test2$Class=as.factor(test2$Class)

####################Randome forest##############################################
library(randomForest)
train$Class=as.factor((train$Class))
test$Class=as.factor((test$Class))
###tune ntree
ntree=function(n){
  best.ntree=1
  accuracyTest=vector("numeric")
  for(i in 20:n){
    set.seed(100)
    model <- randomForest(Class ~ ., data = train2,ntree=i)
    pred <- predict(model, newdata = test2[,1:4])
    table=table(pred, test2$Class)
    accuracy=sum(diag(table))/sum(table)
    #print(i)
    accuracyTest=c(accuracyTest,accuracy)
}
  best.ntree=which.max(accuracyTest)
  model <- randomForest(Class ~ ., data = train2,ntree=best.ntree)
```

```
 pred <- predict(model, newdata = test2[,1:4],type="class")
 table=table(pred, test2$Class)
 accuracy=sum(diag(table))/sum(table)
 list(best.ntree,pred,table,accuracy)
}
a=ntree(200)
predicted.rf=ntree(200)[[2]]
table.rf=ntree(200)[[3]]
accuracy.rf=ntree(200)[[4]]

> a=ntree(200)
> a
[[1]]
[1] 41

[[2]]
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
20  21  22  23  24  25
  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
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44
45  46  47  48  49  50
  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
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69
70  71  72  73  74  75
  0   0   0   1   0   0   1   0   1   1   1   0   1   1   1   1   0   1   1
0   1   0   1   1   1
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94
95  96  97  98  99 100
  0   0   1   1   1   1   0   1   1   0   1   0   0   1   1   0   0   0   0
0   0   0   0   0   0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 1
20 121 122 123 124 125
  0   0   1   1   1   1   1   0   1   0   1   0   0   0   0   0   0   0   0
1   0   1   0   0   1
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 1
45 146 147 148 149 150
  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
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 1
70 171 172 173 174 175
  0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 1
95 196 197 198 199 200
  0   0   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0
0   0   0   0   0   0
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 2
20 221 222 223 224 225
  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
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 2
45 246 247 248 249 250
```

```
  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
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 2
70 271 272 273 274 275
  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
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 2
95 296 297 298 299 300
  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
301
  0
Levels: 0 1

[[3]]

pred   0   1
   0 220  42
   1  18  21

[[4]]
[1] 0.8006645

>
```

2. Adaboost.m1 with decision stumps

install.packages("adabag")

library("adabag")

set.seed(125)

```
> set.seed(125)
> model.ab <- boosting(Class ~ ., data = train2,control = rpart.control(maxde
pth = 1,cp=0.005))
>
> barplot(model.ab$imp[order(model.ab$imp, decreasing = TRUE)],
+         ylim = c(0, 100), main = "Variables Relative Importance",
+          col = "lightblue")
> predicted.ab=predict(model.ab,test2[,1:4])
> predicted.ab$class
  [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"
 [25] "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"
 [49] "0" "0" "0" "0" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1"
"0" "1" "0" "0" "1" "0"
 [73] "0" "1" "1" "0" "1" "0" "1" "1" "1" "1" "1" "0" "1" "0" "1" "1" "0" "1"
"1" "1" "0" "1" "0" "1"
 [97] "0" "1" "0" "0" "0" "0" "0" "0" "1" "0" "0" "1" "0" "0" "0" "0" "0" "0"
"0" "1" "0" "0" "0" "0"
[121] "0" "0" "1" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
```

```
[145] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[169] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[193] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[217] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[241] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[265] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[289] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
> table.ab=table(predicted.ab$class, test2[,5])
> accuracy.ab=sum(diag(table.ab))/sum(table.ab)
> accuracy.ab
[1] 0.8272425
> model.ab <- boosting(Class ~ ., data = train2,control = rpart.control(maxde
pth = 1,cp=0.01))
> barplot(model.ab$imp[order(model.ab$imp, decreasing = TRUE)],
+         ylim = c(0, 100), main = "Variables Relative Importance",
+         col = "lightblue")
> predicted.ab=predict(model.ab,test2[,1:4])
> predicted.ab$class
  [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"
 [25] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
 [49] "0" "0" "0" "0" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1"
"1" "1" "0" "0" "1" "0"
 [73] "0" "1" "1" "0" "1" "0" "1" "1" "1" "1" "1" "0" "1" "0" "1" "0" "0" "1"
"0" "1" "0" "0" "0" "1"
 [97] "0" "1" "0" "0" "0" "1" "0" "0" "0" "0" "0" "1" "0" "0" "0" "0" "0" "0"
"0" "1" "0" "0" "0" "0"
[121] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[145] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[169] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[193] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[217] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[241] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[265] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
"0" "0" "0" "0" "0" "0"
[289] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
> table.ab=table(predicted.ab$class, test2[,5])
> accuracy.ab=sum(diag(table.ab))/sum(table.ab)
> accuracy.ab
[1] 0.820598
> set.seed(125)
 > table.ab

      0   1
```

```
  0 227  40
  1  11  23
> set.seed(125)
> model.ab <- boosting(Class ~ ., data = train2,control = rpart.control(maxdepth = 1,cp=0
>
> barplot(model.ab$imp[order(model.ab$imp, decreasing = TRUE)],
+         ylim = c(0, 100), main = "Variables Relative Importance",
+         col = "lightblue")
> predicted.ab=predict(model.ab,test2[,1:4])
> predicted.ab$class
  [1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
 [25] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
 [49] "0" "0" "0" "0" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "0"
 [73] "0" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "0" "1" "0" "1" "1" "0" "1" "1" "1" "0"
 [97] "0" "1" "0" "0" "0" "1" "0" "0" "1" "0" "0" "1" "0" "0" "0" "0" "0" "0" "0" "1" "0"
[121] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[145] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[169] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[193] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[217] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "0" "0" "0" "0" "0" "0"
[241] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[265] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[289] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
> table.ab=table(predicted.ab$class, test2[,5])
> table.ab

      0   1
  0 225  37
  1  13  26
> accuracy.ab=sum(diag(table.ab))/sum(table.ab)
> accuracy.ab
[1] 0.833887
> model.ab <- boosting(Class ~ ., data = train2,control = rpart.control(maxdepth = 1,cp=0
>
> barplot(model.ab$imp[order(model.ab$imp, decreasing = TRUE)],
+         ylim = c(0, 100), main = "Variables Relative Importance",
+          col = "lightblue")
> predicted.ab=predict(model.ab,test2[,1:4])
> predicted.ab$class
  [1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
 [25] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
 [49] "0" "0" "0" "0" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "0"
 [73] "0" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "0" "1" "0" "1" "1" "0" "1" "1" "1" "0"
 [97] "0" "1" "0" "0" "0" "1" "0" "0" "1" "0" "0" "1" "0" "0" "0" "0" "0" "0" "0" "1" "0"
[121] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[145] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[169] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[193] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[217] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "0" "0" "0" "0" "0" "0"
[241] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[265] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
[289] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
> table.ab=table(predicted.ab$class, test2[,5])
> table.ab

      0   1
  0 225  38
  1  13  25
> accuracy.ab=sum(diag(table.ab))/sum(table.ab)
> accuracy.ab
[1] 0.8305648
```

```
>
> library(nnet)
> # fit model model.ab$class
>
>
> model.nn <- nnet(Class~., data=train2, size=4, decay=0.0001, maxit=6000)
# weights:  25
initial  value 334.386645
iter  10 value 254.441502
iter  20 value 248.415513
iter  30 value 246.407025
iter  40 value 245.157596
iter  50 value 243.797540
iter  60 value 233.346446
iter  70 value 227.029066
iter  80 value 226.446905
iter  90 value 225.022529
iter 100 value 224.625579
iter 110 value 223.641646
iter 120 value 222.714594
iter 130 value 222.260238
iter 140 value 221.371477
iter 150 value 220.545625
iter 160 value 220.337560
iter 170 value 220.223165
iter 180 value 220.192531
iter 190 value 220.180332
iter 200 value 220.163626
iter 210 value 220.151278
iter 220 value 220.125414
iter 230 value 220.119871
final  value 220.117587
converged
> # summarize the fit
> summary(model.nn)
a 4-4-1 network with 25 weights
options were - entropy fitting  decay=1e-04
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1
 -0.48  35.38   0.00   0.44  29.17
 b->h2 i1->h2 i2->h2 i3->h2 i4->h2
  2.37   1.74   0.11  21.56  -3.59
 b->h3 i1->h3 i2->h3 i3->h3 i4->h3
  1.66  59.61  -0.01  -3.05  21.09
 b->h4 i1->h4 i2->h4 i3->h4 i4->h4
 -2.41  36.48   0.03   5.04  -1.42
  b->o  h1->o  h2->o  h3->o  h4->o
 32.90 -39.22  33.05 -57.20 -29.33
> # make predictions
> predicted.nn <- predict(model.nn , test2[,1:4], type="class")
> # summarize accuracy
> table.nn=table(predicted.nn, test2$Class)
> table.nn

predicted.nn   0   1
           0 227  40
           1  11  23
> accucary.nn=sum(diag(table.nn))/sum(table.nn)
> accucary.nn
[1] 0.8305648
```

```
> model.nn <- nnet(Class~., data=train2, size=4, decay=0.001, maxit=6000)
# weights:  25
initial  value 367.688533
iter  10 value 254.829814
iter  20 value 251.968174
iter  30 value 247.285720
iter  40 value 246.865729
iter  50 value 246.779823
iter  60 value 246.678188
iter  70 value 244.278247
iter  80 value 239.063747
iter  90 value 232.832537
iter 100 value 228.142143
iter 110 value 227.733843
iter 120 value 227.577433
iter 130 value 227.540598
iter 140 value 227.510212
iter 150 value 227.500673
iter 160 value 227.497205
iter 170 value 227.492246
final  value 227.491536
converged
> # summarize the fit
> summary(model.nn)
a 4-4-1 network with 25 weights
options were - entropy fitting  decay=0.001
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1
  2.24  -1.43   0.00   1.34  -1.29
 b->h2 i1->h2 i2->h2 i3->h2 i4->h2
  0.17  39.40  -0.02  -1.48  31.11
 b->h3 i1->h3 i2->h3 i3->h3 i4->h3
 -0.47   0.39  -0.08 -17.64   1.50
 b->h4 i1->h4 i2->h4 i3->h4 i4->h4
  0.70 -17.02   0.00  -3.24 -12.73
  b->o  h1->o  h2->o  h3->o  h4->o
  7.04   6.79 -49.91  -9.17  23.08
> # make predictions
> predicted.nn <- predict(model.nn , test2[,1:4], type="class")
> # summarize accuracy
> table.nn=table(predicted.nn, test2$Class)
> table.nn

predicted.nn   0   1
           0 229  48
           1   9  15
> accucary.nn=sum(diag(table.nn))/sum(table.nn)
> accucary.nn
[1] 0.8106312


 > model.nn <- nnet(Class~., data=train2, size=4, decay=0.005, maxit=6000)
 # weights:  25
 initial  value 373.110600
 iter  10 value 249.473847
 iter  20 value 248.145629
```

```
iter  30 value 246.822786
iter  40 value 245.952340
iter  50 value 244.208126
iter  60 value 243.747339
iter  70 value 243.493844
iter  80 value 243.424669
iter  90 value 243.394275
iter 100 value 243.354468
iter 110 value 243.298658
iter 120 value 243.192467
iter 130 value 243.101598
iter 140 value 243.040595
iter 150 value 242.970083
iter 160 value 242.947702
iter 170 value 242.935332
iter 180 value 242.927470
iter 190 value 242.925630
final  value 242.925310
converged
> # summarize the fit
> summary(model.nn)
a 4-4-1 network with 25 weights
options were - entropy fitting  decay=0.005
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1
  0.21   6.58  -0.01  -2.04  14.60
 b->h2 i1->h2 i2->h2 i3->h2 i4->h2
  0.81  -7.69  -0.02  -4.41 -17.46
 b->h3 i1->h3 i2->h3 i3->h3 i4->h3
 -0.28   1.77  -0.05 -11.83   3.96
 b->h4 i1->h4 i2->h4 i3->h4 i4->h4
  0.21   6.98  -0.01  -2.04  15.49
  b->o  h1->o  h2->o  h3->o  h4->o
  6.10 -15.83  19.78  -8.95 -16.81
> # make predictions
> predicted.nn <- predict(model.nn , test2[,1:4], type="class")
> # summarize accuracy
> table.nn=table(predicted.nn, test2$Class)
> table.nn

predicted.nn   0   1
           0 237  60
           1   1   3
> accucary.nn=sum(diag(table.nn))/sum(table.nn)
> accucary.nn
[1] 0.7973422


>



> logistic <- glm(Class ~ ., data = train2,family=binomial("logit"))
> anova(logistic)
Analysis of Deviance Table

Model: binomial, link: logit

Response: Class

Terms added sequentially (first to last)
```

```
     Df Deviance Resid. Df Resid. Dev
NULL                   446     509.74
R     1  29.6241       445     480.12
F     1   8.9152       444     471.20
M     0   0.0000       444     471.20
T     1  11.7245       443     459.48
> summary(logistic)

Call:
glm(formula = Class ~ ., family = binomial("logit"), data = train2)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6009  -0.8415  -0.5578   0.8134   2.5225

Coefficients: (1 not defined because of singularities)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    -0.4513     0.2205  -2.047  0.04070 *
R           -1182.7272   289.8257  -4.081 4.49e-05 ***
F            1751.2735   426.0991   4.110 3.96e-05 ***
M                   NA         NA      NA       NA
T            -309.5670    96.1268  -3.220  0.00128 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 509.74  on 446  degrees of freedom
Residual deviance: 459.48  on 443  degrees of freedom
AIC: 467.48

Number of Fisher Scoring iterations: 4

>
> head(train2)
        R       F    M       T Class
1 0.00016 0.00400 1.00 0.00784     1
2 0.00000 0.00104 0.26 0.00224     1
3 0.00008 0.00128 0.32 0.00280     1
4 0.00016 0.00160 0.40 0.00360     1
5 0.00008 0.00192 0.48 0.00616     0
6 0.00032 0.00032 0.08 0.00032     0
> model.lr <- glm(Class ~ ., data = train2[,-3],family=binomial("logit"))
> summary(model.LR)
Error in summary(model.LR) : object 'model.LR' not found
> anova(model.LR)
Error in anova(model.LR) : object 'model.LR' not found
> model.lr <- glm(Class ~ ., data = train2[,-3],family=binomial("logit"))
> predicted.lr= predict(model.lr,test2[,c(1,2,4)],type="response")
> predicted.lr <- ifelse(predicted.lr > 0.4,1,0)
> table.lr=table(predicted.lr, test2$Class)
> table.lr

predicted.lr   0   1
           0 226  42
           1  12  21
> accuracy.lr=sum(diag(table.lr))/sum(table.lr)
```

```
> accuracy.lr
[1] 0.820598
```

```
> library(class)
> #Fitting model
> model.knn <-knn(train2[,-5],test2[,-5],train2$Class,k=5)
> #install.packages("gmodels")
> #library(gmodels)
> #CrossTable(x=test2$Class,y=fit,prop.chisq = FALSE)
> predicted.knn=model.knn
> table.knn=table(predicted.knn, test2$Class)
> table.knn

predicted.knn   0   1
            0 223  47
            1  15  16
> accuracy.knn=sum(diag(table.knn))/sum(table.knn)
> accuracy.knn
[1] 0.7940199


>
```

```
> library(e1071)
> #Fitting model
> model.nb <-naiveBayes(Class ~ ., data = train2)
> summary(model.nb)
        Length Class  Mode
apriori 2      table  numeric
tables  4      -none- list
levels  2      -none- character
call    4      -none- call
> #Predict Output
> predicted.nb= predict(model.nb,test2[,1:4])
>
> table.nb=table(predicted.nb, test2$Class)
> table.nb

predicted.nb   0   1
           0 226  44
           1  12  19
> accuracy.nb=sum(diag(table.nb))/sum(table.nb)
> accuracy.nb
[1] 0.8139535


>
```

```
> df=cbind(predicted.lr,as.numeric(levels(predicted.knn))[predicted.knn],as.numeric(level
)[predicted.dt],as.numeric(levels(predicted.nb))[predicted.nb],as.numeric(predicted.nn))
> colnames(df)=c("predicted.lr","predicted.knn","predicted.dt","predicted.nb","predicted.
> predicted.ensemble.uw=apply(df,1,function(x) names(which.max(table(x))))
> predicted.ensemble.uw
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
```

```
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
"0" "0" "0" "1" "1" "1" "1" "0" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97
"1" "1" "1" "1" "1" "0" "0" "1" "1" "0" "1" "0" "0" "1" "0" "0" "0" "0" "0" "0" "1" "0" "
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 1
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 1
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 1
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 2
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 2
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 2
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 2
"0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "
301
"0"
> table.ensemble.uw=table(predicted.ensemble.uw,test2$Class)
> table.ensemble.uw

predicted.ensemble.uw   0   1
                    0 229  43
                    1   9  20
> accucary.ensemble.uw=sum(diag(table.ensemble.uw))/sum(table.ensemble.uw)
> accucary.ensemble.uw
[1] 0.8272425


>

> weight1=c(1,3,5,7,9)
> df1=df*weight1
> predicted.ensemble.w=apply(df1,1,sum)
> table(predicted.ensemble.w)
predicted.ensemble.w
  0   1   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
238   4   2   4   4   1   3   3   4   3   2   4   2   1   4   1   2   3   1   4   1   3
> predicted.ensemble.w=ifelse(predicted.ensemble.w>8,1,0)
> predicted.ensemble.w
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  0   0   0   1   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   1   0   1
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97
  1   1   1   1   1   0   1   1   1   0   1   1   1   1   0   0   0   0   1   0   1   0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
  0   1   1   1   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 1
  0   0   0   0   0   0   0   0   0   0   0   1   0   1   0   0   0   0   0   0   0   0
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 2
```

```
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
301
  0
> table.ensemble.w=table(predicted.ensemble.w,test2$Class)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 221  38
                   1  17  25
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8172757


>


> weight1=c(1,3,5,7,9)
> df1=df*weight1
> predicted.ensemble.w=apply(df1,1,sum)
> table(predicted.ensemble.w)
predicted.ensemble.w
  0   1   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
238   4   2   4   4   1   3   3   4   3   2   4   2   1   4   1   2   3   1   4   1   3
> predicted.ensemble.w=ifelse(predicted.ensemble.w>8,1,0)
> predicted.ensemble.w
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  0   0   0   1   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   0   1   0
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97
  1   1   1   1   1   0   1   1   1   0   1   1   1   1   0   0   0   0   1   0   1   0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
  0   1   1   1   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 1
  0   0   0   0   0   0   0   0   0   0   0   1   0   1   0   0   0   0   0   0   0   0
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
301
  0
> table.ensemble.w=table(predicted.ensemble.w,test2$Class)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 221  38
                   1  17  25
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
```

```
> accucary.ensemble.w
[1] 0.8172757
> weight2=c(5,4,5,4,4)
> df1=df*weight2
> predicted.ensemble.w=apply(df1,1,sum)
> table(predicted.ensemble.w)
predicted.ensemble.w
  0   4   5   8   9  10  12  13  14  17  18  22
238   9   8   5  11   1   1   8   3   7   6   4
> predicted.ensemble.w=ifelse(predicted.ensemble.w>8,1,0)
> predicted.ensemble.w
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  0   0   0   1   1   1   1   1   1   1   1   0   1   1   1   1   1   1   1   1   1   1
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97
  1   1   1   1   1   1   0   1   1   1   1   1   0   1   1   0   0   0   1   0   1   0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
  0   0   0   1   1   1   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
301
  0
> table.ensemble.w=table(predicted.ensemble.w,test2$Class)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 224  36
                   1  14  27
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.833887
> weight3=c(accuracy.lr/(1-accuracy.lr),accuracy.knn/(1-accuracy.knn),accuracy.dt/(1-accu
cy.nb/(1-accuracy.nb),accucary.nn/(1-accucary.nn))
> df1=df*weight
> predicted.ensemble.w=apply(df1,1,sum)
> table(predicted.ensemble.w)
predicted.ensemble.w
               0 3.77777777777778 4.18965517241379            4.375 4.57407407407408 4.67
             238                2                4                3                4
7.96743295019157 8.15277777777778 8.35185185185185 8.45702306079664 8.56465517241379 8.76
               3                1                4                1                1
8.86890045543266 8.94907407407408 9.05424528301887 9.25331935709294 12.3424329501916 12.5
               1                2                1                1                1
12.6466782332104 12.7268518518519 12.8320230607966 13.1387292464879 13.2439004554327 13.4
               1                2                2                1                1
13.6283193570929 16.9165070242656 17.0216782332104 17.2207523072845 17.8179745295067 21.5
               1                4                3                3                3
> predicted.ensemble.w=ifelse(predicted.ensemble.w>8,1,0)
> predicted.ensemble.w
```

```
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  0   0   0   1   1   1   1   1   1   1   1   0   1   1   1   1   1   1   1   1   1   1
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97
  1   1   1   1   1   1   0   1   1   1   1   1   0   1   1   0   0   0   1   0   1   0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
  0   0   1   1   1   1   0   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
301
  0
> table.ensemble.w=table(predicted.ensemble.w,test2$Class)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 224  34
                   1  14  29
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8405316



weight6=c(accuracy.lr/(1-accuracy.lr),accuracy.knn/(1-accuracy.knn),accuracy.dt/(1-accura
.nb/(1-accuracy.nb),
+           accucary.nn/(1-accucary.nn),accuracy.rf/(1-accuracy.rf),accuracy.ab/(1-accura
> df1.7=df.7*weight6
> predicted.ensemble.w=apply(df1.7,1,sum)
> table(predicted.ensemble.w)
predicted.ensemble.w
               0 4.01666666666667 4.28070175438596            4.375 4.57407407407408 4.67
             236                 4                 2                 2                 1
            5.02 7.70967741935484 8.03333333333333 8.56140350877193              8.75 9.14
               1                 1                 1                 2                 2
           10.04 11.5645161290323             12.05 12.8421052631579            13.125 13.7
               3                 1                 2                 2                 2
14.0377358490566             15.06 15.4193548387097 16.0666666666667 17.1228070175439
               2                 2                 2                 3                 3
18.2962962962963 18.7169811320755 19.2741935483871             20.08 20.0833333333333
               3                 2                 1                 3                 1
23.1290322580645 23.3962264150943              25.1 25.6842105263158 28.0754716981132
               1                 1                 1                 2                 1
> predicted.ensemble.w=ifelse(predicted.ensemble.w>8,1,0)
> predicted.ensemble.w
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
```

```
  0   0   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   1   0   1   0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
  0   1   1   1   1   1   0   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
301
  0
> table.ensemble.w=table(predicted.ensemble.w,test2$Class)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 221  30
                   1  17  33
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8438538
> weight5=c(5,4,5,4,4,5,4)
> df1.7=df.7*weight5
> predicted.ensemble.w=apply(df1.7,1,sum)
> table(predicted.ensemble.w)
predicted.ensemble.w
  0   4   5   8  10  12  15  16  20  24  25  30
236   5   9   8   3   7   6   9  12   3   2   1
> predicted.ensemble.w=ifelse(predicted.ensemble.w>8,1,0)
> predicted.ensemble.w
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  0   0   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97
  1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   0   1   0   1   0   1   0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
  0   0   0   1   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
301
```

```
  0
> table.ensemble.w=table(predicted.ensemble.w,test2$Class)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 225  33
                   1  13  30
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8471761
> weight4=c(1,3,5,7,9,11,13)
> df1.7=df.7*weight4
> predicted.ensemble.w=apply(df1.7,1,sum)
> table(predicted.ensemble.w)
predicted.ensemble.w
  0   1   2   3   4   5   6   7   9  11  12  13  14  15  18  20  21  22  25  26  27  28
236   1   2   2   3   4   1   2   3   4   2   1   2   3   3   2   2   1   1   3   2   1
 36  39  44  52  54  55  65
  3   2   3   3   2   1   1
> predicted.ensemble.w=ifelse(predicted.ensemble.w>8,1,0)
> predicted.ensemble.w
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  0   0   0   1   1   1   0   1   1   1   1   1   1   0   1   1   1   1   1   1   0   1
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97
  1   1   0   1   1   1   1   1   1   0   1   1   1   1   1   1   0   0   1   0   1   0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 1
  0   1   1   1   1   0   0   0   0   1   1   0   0   0   0   1   0   1   0   0   0   0
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 1
  0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 1
  0   0   0   0   0   0   0   0   0   0   0   1   1   1   0   0   0   0   0   0   0   0
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 2
  0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 2
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
301
  0
> table.ensemble.w=table(predicted.ensemble.w,test2$Class)
> table.ensemble.w

predicted.ensemble.w   0   1
                   0 216  35
                   1  22  28
> accucary.ensemble.w=sum(diag(table.ensemble.w))/sum(table.ensemble.w)
> accucary.ensemble.w
[1] 0.8106312



>
```