Group 7

2022/4/15

```
library(MASS)
library(class)
library(caret)
library(e1071)
library(mboost)
library(plyr)
library(import)
library(ipred)
library(LiblineaR)
library(naivebayes)
```

```
set.seed(1082)
data=read.csv("train_haralick.csv")[2:15]
data$Label <- factor(data$Label)
head(data)
```

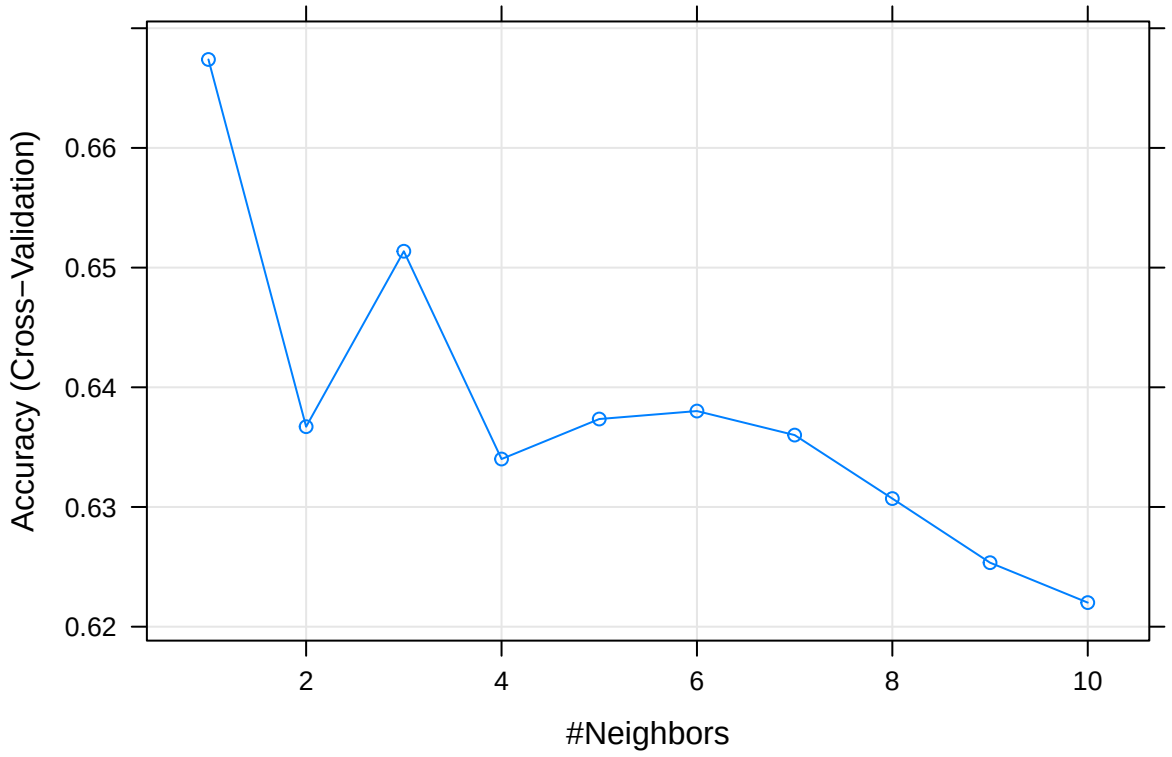
```
##             X0         X1         X2       X3         X4       X5        X6
## 1 0.0002262115 714.260755 0.08693126 391.1341 0.04802351 346.7689  850.2755
## 2 0.0004772510  98.809934 0.90161066 502.1069 0.13408052 358.7879 1909.6177
## 3 0.0038342554   3.342049 0.99550599 371.7698 0.48388371 357.8895 1483.7370
## 4 0.0024503480   4.640281 0.99659493 681.3819 0.44946174 413.9483 2720.8872
## 5 0.0026477972   2.706267 0.99675230 416.6336 0.51848060 406.9480 1663.8282
## 6 0.0017734583   3.634416 0.99716403 640.8222 0.48123683 372.7826 2559.6544
##         X7        X8           X9      X10          X11       X12 Label
## 1 6.885095 12.571946 6.860737e-05 5.793479 -0.006468766 0.2660565     0
## 2 7.403880 11.605845 2.217826e-04 4.339433 -0.196541451 0.9521447     1
## 3 6.894114  8.758403 1.033588e-03 2.116119 -0.515693055 0.9987487     1
## 4 7.580373  9.613825 8.864515e-04 2.330350 -0.544206041 0.9995776     5
## 5 7.343552  9.040460 1.118748e-03 1.988782 -0.576473047 0.9996316     5
## 6 7.695696  9.602540 1.010622e-03 2.164329 -0.566461963 0.9997128     5
```

KNN

```
trControl=trainControl(method  = "cv",number  = 5)
knn.fit <- train(Label ~ .,
             method = "knn",
             tuneGrid= expand.grid(k = 1:10),
             trControl= trControl,
             metric = "Accuracy",
             data  = data)
knn.fit
```

```
## k-Nearest Neighbors
##
## 1500 samples
##   13 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1199, 1202, 1199, 1201, 1199
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.6673898  0.5805034
##    2  0.6367092  0.5412416
##    3  0.6513672  0.5580551
##    4  0.6340069  0.5343046
##    5  0.6373491  0.5375623
##    6  0.6380157  0.5378683
##    7  0.6360069  0.5330851
##    8  0.6307023  0.5269964
##    9  0.6253467  0.5187588
##   10  0.6220110  0.5143842
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 1.
```

```
plot(knn.fit)
```



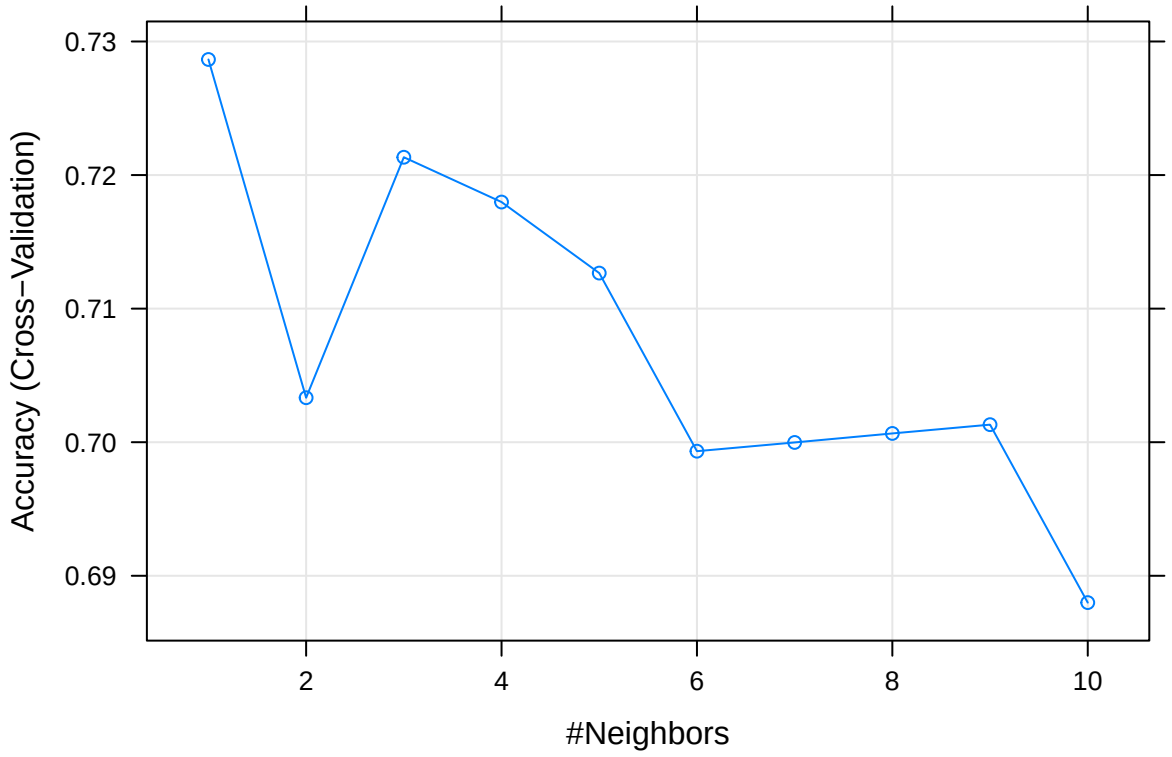

```
confusionMatrix(knn.fit,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##            Reference
## Prediction   0   1   2   3   4   5
##          0 312  35  25  38  13   0
##          1  28 131   0  19  47  51
##          2  17   0  30   3   0   0
##          3  34  19   3 149   9   2
##          4   4  56   0  14  71  10
##          5   0  62   0   4   6 308
##
##  Accuracy (average) : 0.6673
```

```
trControl=trainControl(method  = "cv",number  = 5)
knn.fit.2 <- train(Label ~ X0+X1+X2+X3+X4+X5+X7+X8+X9+X10+X11+X12,
             method = "knn",
             tuneGrid= expand.grid(k = 1:10),
             trControl= trControl,
             metric = "Accuracy",
             data  = data)
knn.fit.2
```

```
## k-Nearest Neighbors
##
## 1500 samples
##   12 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1199, 1200, 1201, 1200, 1200
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.7286544  0.6583471
##    2  0.7033298  0.6264325
##    3  0.7213344  0.6484070
##    4  0.7179810  0.6437876
##    5  0.7126609  0.6355645
##    6  0.6993276  0.6194185
##    7  0.6999831  0.6190879
##    8  0.7006564  0.6191464
##    9  0.7013142  0.6187743
##   10  0.6879919  0.6015207
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 1.
```

```
plot(knn.fit.2)
```




```
confusionMatrix(knn.fit.2,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##            Reference
## Prediction   0   1   2   3   4   5
##          0 314  28  24  31   8   0
##          1  16 165   0  21  42  33
##          2  26   0  30   1   0   0
##          3  32  19   4 162   5   1
##          4   7  44   0  12  88   3
##          5   0  47   0   0   3 334
##
##  Accuracy (average) : 0.7287
```

Bagged CART

```
treebag.fit <- train(Label ~ .,
             method = "treebag",
             trControl= trControl,
             metric = "Accuracy",
             data = data)
treebag.fit
```

```
## Bagged CART
##
## 1500 samples
##   13 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1201, 1201, 1200, 1197, 1201
## Resampling results:
##
##   Accuracy   Kappa
##   0.8960038  0.8683655
```

```
confusionMatrix(treebag.fit,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction   0   1   2   3   4   5
##          0 379  16  17  19   6   0
##          1   5 261   0  17  13   4
##          2   2   0  41   1   0   0
##          3   9  10   0 177   8   0
##          4   0   8   0  13 119   0
##          5   0   8   0   0   0 367
##
##  Accuracy (average) : 0.896
```

```
treebag.fit.2 <- train(Label ~X0+X1+X2+X3+X4+X5+X7+X8+X9+X10+X11+X12,
             method = "treebag",
             trControl= trControl,
             metric = "Accuracy",
             data = data)
treebag.fit.2
```

```
## Bagged CART
##
## 1500 samples
##   12 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1200, 1201, 1201, 1200, 1198
## Resampling results:
##
##   Accuracy   Kappa
##   0.9007053  0.8746138
```

```
confusionMatrix(treebag.fit.2,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction   0   1   2   3   4   5
##          0 375  15  17  18   0   0
##          1   6 257   0  21   8   3
##          2   6   0  41   0   0   0
##          3   8  14   0 178   6   0
##          4   0   9   0  10 132   0
##          5   0   8   0   0   0 368
##
##  Accuracy (average) : 0.9007
```

Linear Discriminant Analysis

```
findLinearCombos(data[1:13])
```

```
## $linearCombos
## $linearCombos[[1]]
## [1] 7 2 4
##
##
## $remove
## [1] 7
```

```
lda.fit <- train(Label ~ .,
             method  = "lda",
             trControl= trControl,
             metric = "Accuracy",
             data = data)
lda.fit
```

```
## Linear Discriminant Analysis
##
## 1500 samples
##   13 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1200, 1201, 1201, 1199, 1199
## Resampling results:
##
##   Accuracy   Kappa
##   0.7932715  0.7405382
```

```
confusionMatrix(lda.fit,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction   0   1   2   3   4   5
##          0 283  11  14  64   7   0
##          1  55 256   0  16   1   0
##          2  35   0  40   4   0   0
##          3  22   5   4 109   7   0
##          4   0  13   0  34 131   0
##          5   0  18   0   0   0 371
##
##  Accuracy (average) : 0.7933
```

```r
lda.fit.2 <- train(Label ~ X0+X1+X2+X3+X4+X5+X7+X8+X9+X10+X11+X12,
             method = "lda",
             trControl= trControl,
             metric = "Accuracy",
             data= data)
lda.fit.2
```

```
## Linear Discriminant Analysis
##
## 1500 samples
##   12 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1201, 1200, 1200, 1200, 1199
## Resampling results:
##
##   Accuracy   Kappa
##   0.7833124  0.7281857
```

```r
confusionMatrix(lda.fit.2,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction   0   1   2   3   4   5
##          0 279  11  15  63   8   0
##          1  55 255   0  18   3   0
##          2  39   0  36   5   0   0
##          3  22   5   7 106   7   0
##          4   0  14   0  35 128   0
##          5   0  18   0   0   0 371
##
##  Accuracy (average) : 0.7833
```

Quadratic Discriminant Analysis

```
qda.fit <- train(Label~X0+X1+X2+X3+X4+X5+X7+X8+X9+X10+X11+X12,method= "qda",
trControl = trControl,
metric= "Accuracy",
data= data)
qda.fit
```

```
## Quadratic Discriminant Analysis
##
## 1500 samples
##   12 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1201, 1199, 1201, 1198, 1201
## Resampling results:
##
##   Accuracy   Kappa
##   0.8499639  0.8130465
```

```
confusionMatrix(qda.fit,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction   0   1   2   3   4   5
##          0 304   7   5  34   1   0
##          1  20 258   0  17   4   1
##          2  57   0  52  10   0   0
##          3  14  10   1 159   9   0
##          4   0  16   0   7 132   0
##          5   0  12   0   0   0 370
##
##  Accuracy (average) : 0.85
```

Naive Bayes

```
nb.fit <- train(Label~.,method= "naive_bayes",
trControl = trControl,
metric= "Accuracy",
data= data)
nb.fit
```

```
## Naive Bayes
##
## 1500 samples
##   13 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1200, 1200, 1200, 1200, 1200
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.5860000  0.5056501
##    TRUE      0.6586667  0.5841683
##
## Tuning parameter 'laplace' was held constant at a value of 0
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were laplace = 0, usekernel = TRUE
##  and adjust = 1.
```

```
confusionMatrix(nb.fit,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction   0   1   2   3   4   5
##          0 168   1  13  34   1   0
##          1  67 174   0  13  13   4
##          2  98   0  43  32   0   0
##          3  38  17   2 125  21   0
##          4  24  93   0  23 111   0
##          5   0  18   0   0   0 367
##
##  Accuracy (average) : 0.6587
```

```
nb.fit.2 <- train(Label~X0+X1+X2+X3+X4+X5+X7+X8+X9+X10+X11+X12,method= "naive_bayes",
trControl = trControl,
metric= "Accuracy",
data= data)
nb.fit.2
```

```
## Naive Bayes
##
## 1500 samples
##   12 predictor
##    6 classes: '0', '1', '2', '3', '4', '5'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1200, 1200, 1201, 1199, 1200
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.5866605  0.5069964
##    TRUE      0.6493429  0.5728260
##
## Tuning parameter 'laplace' was held constant at a value of 0
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were laplace = 0, usekernel = TRUE
##  and adjust = 1.
```

```
confusionMatrix(nb.fit.2,norm="none")
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction   0   1   2   3   4   5
##          0 150   1  12  34   0   0
##          1  77 170   0  14  12   6
##          2  90   0  43  25   0   0
##          3  53  18   3 134  22   0
##          4  25  93   0  20 112   0
##          5   0  21   0   0   0 365
##
##  Accuracy (average) : 0.6493
```

Result

Model	Predictor	Parameter	Accuracy
KNN	All Predictors	K=1	0.6674
KNN	Without X_6	K=1	0.7287
Bagged CART	All Predictors	Default	0.896
Bagged CART	Without X_6	Default	0.9007
LDA	ALL Predictors	Default	0.7933
LDA	Without X_6	Default	0.7833
QDA	Without X_6	Default	0.85
Naive Bayes	ALL Predictors	Default	0.6587
Naive Bayes	Without X_6	Default	0.6493