Support Vector Machine (SVM) Polynomial kernel

Feature space partition ensemble model for replication (FESPAE)

EGG data-based experiments

Article: Lizbeth Naranjo, Carlos J. Perez, Daniel F. Merino (2025). A data ensemble-based approach for detecting vocal disorders using replicated acoustic biomarkers from electroglottography. *Sensing and Bio-Sensing Research Journal*, vol, num, pages.

```
library(tidyverse)
library(e1071)
## change the address where the file will be saved
address = "~/Documents/GitHub/"
setwd("~/Documents/GitHub/")
```

EGG data-based experiments

```
## Comment or uncomment the options: EGG-a, EGG-i, EGG-u
```

```
## EGG-a
datos2 <- read.csv(paste0(address,"a_egg_saarbrucken.csv"),
                   sep = ";",header=TRUE, dec=",")

## name of the files to save results
archivo = "FESPAE_crossval_strata_allvar_SVM_poly3_Saarbruken_egg_a"
```

```
## EGG-i
## datos2 <- read.csv(paste0(address,"i_egg_saarbrucken.csv"),
##                    sep = ";",header=TRUE, dec=",")

## name of the files to save results
## archivo = "FESPAE_crossval_strata_allvar_SVM_poly3_Saarbruken_egg_i"
```

```
## EGG-u
## datos2 <- read.csv(paste0(address,"u_egg_saarbrucken.csv"),
##                    sep = ";",header=TRUE, dec=",")

## name of the files to save results
## archivo = "FESPAE_crossval_strata_allvar_SVM_poly3_Saarbruken_egg_u"
```

```
dim(datos2)
```

```
[1] 675  36
```

```
summary(datos2)
```

```
   ID_fact       status_fact      SEX             JITTER
 Min.   :  1.0   Min.   :0    Min.   :0.0000   Min.   :  0.11
 1st Qu.:169.5   1st Qu.:0    1st Qu.:0.0000   1st Qu.:  0.49
 Median :338.0   Median :1    Median :0.0000   Median :  5.52
 Mean   :338.0   Mean   :1    Mean   :0.4133   Mean   : 18.90
 3rd Qu.:506.5   3rd Qu.:2    3rd Qu.:1.0000   3rd Qu.: 25.89
 Max.   :675.0   Max.   :2    Max.   :1.0000   Max.   :281.41
   SHIMMER             CPP              D2              FZCF
 Min.   :0.01000   Min.   :12.26   Min.   : 2.480   Min.   :   2.00
 1st Qu.:0.03000   1st Qu.:21.08   1st Qu.: 4.130   1st Qu.:   5.00
 Median :0.04000   Median :24.60   Median : 5.100   Median :   9.00
 Mean   :0.05887   Mean   :24.26   Mean   : 5.666   Mean   :  45.86
 3rd Qu.:0.08000   3rd Qu.:27.36   3rd Qu.: 6.795   3rd Qu.:  20.00
 Max.   :0.37000   Max.   :35.94   Max.   :27.380   Max.   :5323.00
     GNE              HNR            HURST              LZ
 Min.   :0.3800   Min.   :-4.40   Min.   :0.0400   Min.   : 19.00
 1st Qu.:0.6300   1st Qu.:14.07   1st Qu.:0.4200   1st Qu.: 55.50
 Median :0.7700   Median :18.40   Median :0.6300   Median : 83.00
 Mean   :0.9037   Mean   :17.38   Mean   :0.7015   Mean   : 86.81
 3rd Qu.:1.0000   3rd Qu.:21.66   3rd Qu.:0.9100   3rd Qu.:107.00
 Max.   :4.6300   Max.   :32.35   Max.   :1.7800   Max.   :304.00
    MFCC0             MFCC1             MFCC2             MFCC3
 Min.   :-2.6200   Min.   :-23.850   Min.   :-47.190   Min.   :-49.37
 1st Qu.:-0.5700   1st Qu.:  3.300   1st Qu.:-22.285   1st Qu.:-24.59
 Median : 0.1700   Median :  9.150   Median : -9.750   Median :-15.21
 Mean   : 0.1388   Mean   :  8.767   Mean   : -9.716   Mean   :-14.79
 3rd Qu.: 0.8600   3rd Qu.: 14.980   3rd Qu.:  2.915   3rd Qu.: -4.03
 Max.   : 3.0700   Max.   : 30.920   Max.   : 23.390   Max.   : 19.56
    MFCC4             MFCC5             MFCC6             MFCC7
 Min.   :-61.070   Min.   :-67.210   Min.   :-62.600   Min.   :-39.460
 1st Qu.:-21.130   1st Qu.:-12.535   1st Qu.:-16.040   1st Qu.:-11.790
 Median : -8.770   Median : -2.020   Median : -3.600   Median : -2.960
 Mean   : -9.488   Mean   : -3.153   Mean   : -4.740   Mean   : -3.146
 3rd Qu.:  2.400   3rd Qu.:  7.065   3rd Qu.:  5.175   3rd Qu.:  4.690
 Max.   : 38.080   Max.   : 36.640   Max.   : 46.700   Max.   : 40.120
    MFCC8             MFCC9             MFCC10            MFCC11
 Min.   :-44.230   Min.   :-35.240   Min.   :-41.950   Min.   :-41.300
 1st Qu.:-14.685   1st Qu.:-12.990   1st Qu.:-12.415   1st Qu.:-11.705
 Median : -5.340   Median : -4.580   Median : -4.830   Median : -4.040
 Mean   : -5.353   Mean   : -3.993   Mean   : -4.637   Mean   : -4.130
 3rd Qu.:  2.660   3rd Qu.:  3.420   3rd Qu.:  2.760   3rd Qu.:  2.755
 Max.   : 49.090   Max.   : 49.370   Max.   : 41.220   Max.   : 36.180
    MFCC12          PERMUTATION          PPE             SHANNON
 Min.   :-36.840   Min.   :1.150   Min.   :0.1900   Min.   :11.79
 1st Qu.:-10.890   1st Qu.:1.640   1st Qu.:0.5300   1st Qu.:12.12
 Median : -3.390   Median :1.810   Median :0.5500   Median :12.16
 Mean   : -2.904   Mean   :1.827   Mean   :0.5312   Mean   :12.15
```

```
 3rd Qu.:  3.935   3rd Qu.:1.990   3rd Qu.:0.5700   3rd Qu.:12.19
 Max.   : 31.870   Max.   :2.550   Max.   :0.5700   Max.   :12.26
      ZCR          energyentropy   spectralcentroid spectralspread
 Min.   :0.01000   Min.   :2.540   Min.   :0.0600   Min.   :0.1100
 1st Qu.:0.04000   1st Qu.:3.230   1st Qu.:0.1200   1st Qu.:0.1600
 Median :0.07000   Median :3.300   Median :0.1500   Median :0.1800
 Mean   :0.07056   Mean   :3.248   Mean   :0.1545   Mean   :0.1805
 3rd Qu.:0.09000   3rd Qu.:3.310   3rd Qu.:0.1800   3rd Qu.:0.2000
 Max.   :0.23000   Max.   :3.320   Max.   :0.3200   Max.   :0.2800
 spectralentropy  spectralrolloff       RPDE             rep
 Min.   :0.0100   Min.   :0.0100   Min.   :0.1000   Min.   :1
 1st Qu.:0.1900   1st Qu.:0.0700   1st Qu.:0.2700   1st Qu.:1
 Median :0.6600   Median :0.1100   Median :0.3500   Median :2
 Mean   :0.6867   Mean   :0.1178   Mean   :0.3948   Mean   :2
 3rd Qu.:1.0700   3rd Qu.:0.1600   3rd Qu.:0.4950   3rd Qu.:3
 Max.   :1.9100   Max.   :0.4500   Max.   :0.9100   Max.   :3
```

```
head(datos2)
```

```
  ID_fact status_fact SEX JITTER SHIMMER   CPP   D2 FZCF  GNE   HNR HURST LZ
1       1           0   0   6.76    0.06 28.26 3.71   23 0.67 18.96  1.11 44
2       2           0   0   0.31    0.06 23.32 3.40   48 0.47 17.76  1.58 36
3       3           0   0   0.19    0.04 23.55 3.68   58 0.46 21.86  1.70 24
4       4           0   0   0.45    0.01 33.64 2.96   37 0.42 25.78  1.45 27
5       5           0   0   0.39    0.07 26.95 2.66   50 0.39 21.44  1.63 29
6       6           0   0   0.34    0.04 35.36 3.03   35 0.44 28.32  1.45 28
  MFCC0 MFCC1 MFCC2 MFCC3 MFCC4 MFCC5 MFCC6 MFCC7 MFCC8 MFCC9 MFCC10 MFCC11
1 -0.75 11.11 -0.88 -4.03  4.01 -0.10 -5.14 -4.07 -7.72 -9.94 -12.93 -13.10
2 -1.16 -0.60 12.22  4.06  5.84  3.35  6.71  1.68  2.86  3.01   1.88   1.35
3 -1.70  2.81 15.53  7.78  7.22  8.27  3.33  3.90  4.74  6.15   1.74   2.75
4 -1.00 11.34  5.69  1.20  1.84  8.61 -0.60  4.50  0.59  2.95  -0.09   1.14
5 -1.91 11.97  7.29  7.30  7.14  4.34  6.10  3.49  2.89  2.92   2.73   2.47
6 -1.43  6.93  3.04 11.37  1.89  1.65  3.27  5.75 -0.33  4.49  -1.93   2.00
  MFCC12 PERMUTATION  PPE SHANNON  ZCR energyentropy spectralcentroid
1 -10.64        1.94 0.40   12.18 0.02          3.30             0.12
2   2.93        2.53 0.57   12.20 0.01          3.21             0.13
3   2.26        2.41 0.53   12.16 0.01          3.26             0.12
4  -0.36        1.34 0.55   12.18 0.01          3.28             0.10
5   2.88        1.76 0.57   12.20 0.01          3.23             0.09
6  -1.05        1.38 0.55   12.05 0.01          3.29             0.13
  spectralspread spectralentropy spectralrolloff RPDE rep
1           0.19            0.16            0.04 0.50   1
2           0.23            0.07            0.02 0.57   2
3           0.21            0.06            0.02 0.43   3
4           0.18            0.07            0.03 0.41   1
5           0.16            0.03            0.01 0.47   2
6           0.20            0.09            0.03 0.28   3
```

Re-Scale explanatory variables

```
## Scale the variables
datos2 <- as.data.frame(datos2)
datos2$STATUS_fact = as.factor(as.numeric(factor(datos2$status_fact)))

table(datos2$STATUS_fact)
```

```
  1   2   3
225 225 225
```

```
datos <- transform(datos2,
sJITTER= scale(JITTER), sSHIMMER= scale(SHIMMER), sCPP= scale(CPP),
sD2= scale(D2), sFZCF= scale(FZCF), sGNE= scale(GNE),
sHNR= scale(HNR), sHURST= scale(HURST), sLZ= scale(LZ),
sMFCC0= scale(MFCC0),
sMFCC1= scale(MFCC1), sMFCC2= scale(MFCC2), sMFCC3= scale(MFCC3),
sMFCC4= scale(MFCC4), sMFCC5= scale(MFCC5), sMFCC6= scale(MFCC6),
sMFCC7= scale(MFCC7), sMFCC8= scale(MFCC8), sMFCC9= scale(MFCC9),
sMFCC10= scale(MFCC10), sMFCC11= scale(MFCC11), sMFCC12= scale(MFCC12),
sPERMUTATION= scale(PERMUTATION), sPPE= scale(PPE), sSHANNON= scale(SHANNON),
sZCR= scale(ZCR),
senergyentropy= scale(energyentropy), sspectralcentroid= scale(spectralcentroid),
sspectralspread= scale(spectralspread), sspectralentropy= scale(spectralentropy),
sspectralrolloff= scale(spectralrolloff), sRPDE= scale(RPDE))

datos$ID_fact = rep(1:225,each=3)

dim(datos)
```

```
[1] 675  69
```

```
## data set
trainc <- datos %>% select(
sJITTER, sSHIMMER, sCPP, sD2, sFZCF,
sGNE, sHNR, sHURST, sLZ, sMFCC0,
sMFCC1, sMFCC2, sMFCC3, sMFCC4, sMFCC5,
sMFCC6, sMFCC7, sMFCC8, sMFCC9, sMFCC10,
sMFCC11, sMFCC12,
sPERMUTATION, sPPE, sSHANNON, sZCR,
senergyentropy, sspectralcentroid, sspectralspread,
sspectralentropy, sspectralrolloff, sRPDE,
STATUS_fact,SEX, rep,ID_fact)
```

Crossvalidation

Subspaces

```
## Function to compute the Mode
Mode <- function(x, na.rm = FALSE) {
  if(na.rm){
    x = x[!is.na(x)]
    }
  ux <- unique(x)
  return(ux[which.max(tabulate(match(x, ux)))])
}

## Partition of subspaces
## The feature space is randomly partitioned into K subspaces with roughly equal sizes
## k = number of predictors
## K = subspaces

K0 = 4  ## sub-spaces
k = 32  ## explanatory variables
k2 = round(k/K0)
space = 1:k
subspaces = rep(list(rep(NA,k2)),K0)  ## Subspaces
set.seed(12345)
for(j in 1:(K0-1)){
    space1 = sample(space, size=k2, replace=FALSE)
    space = setdiff(space,space1)
    subspaces[[j]] = space1[order(space1)]
}
space1 = space
subspaces[[K0]] = space1[order(space1)]
## 32 features = 1x32, 2x16, 4x8,
subspaces
```

```
## [[1]]
## [1] 11 14 16 19 24 26 28 29
##
## [[2]]
## [1]  2  6  7 10 12 21 30 32
##
## [[3]]
## [1]  1  4  5  9 13 15 27 31
##
## [[4]]
## [1]  3  8 17 18 20 22 23 25
```

Training and testing data subsets

```
## Select data: 75% training & 25% testing stratified per category
SIM = 100   ## repeat N times the cross-validation process
N = 225   ## sample size
Nfit = 168   ## sample size for training subset
Ntest = 57   ## sample size for testing subset
Ncat = 75   ## sample size per category
Ncatfit = 56   ## training per category
Ncattest = 19   ## testing per category
FIT <- matrix(0,SIM,Nfit)   ## training subsets
TEST <- matrix(0,SIM,Ntest)   ## testing subsets

categoria = trainc %>% filter(rep==1) %>% select(STATUS_fact)
categoria = as.numeric(categoria$STATUS_fact)
id = 1:N
set.seed(12345)
for(si in 1:SIM){
  for(j in 1:3){
    idcat = id[categoria==j]   ## stratified per category j
    ran0 = sample(idcat, size=Ncatfit, replace=FALSE)

    FIT[si,(j-1)*Ncatfit+1:Ncatfit] <- sort(ran0)
    TEST[si,(j-1)*Ncattest+1:Ncattest] <- setdiff(idcat,ran0)
} }
```

Classification metrics for models predicting nominal outcomes

```
## Functions to compute classification metrics
## Ytrue = true response variable
## Ypred = predicted outcome
## cat = category
## TP = true positive
## TN = true negative
## FP = false positive
## FN = false negative

## Function to compute the precision per class=cat
fn_precision_class <- function(Ytrue,Ypred,cat){
  TP = sum(Ypred[Ytrue==cat]==cat)
  FP = sum(Ypred[Ytrue!=cat]==cat)
  precision = TP/(TP+FP)
  return(precision)
}

## Function to compute the recall per class=cat
fn_recall_class <- function(Ytrue,Ypred,cat){ ## cat==category
  TP = sum(Ypred[Ytrue==cat]==cat)
  FN = sum(Ypred[Ytrue==cat]!=cat)
  recall = TP/(TP+FN)
  return(recall)
}

## Function to compute the F1-score per class=cat
fn_f1score_class <- function(Ytrue,Ypred,cat){ ## cat==category
  TP = sum(Ypred[Ytrue==cat]==cat)
  FP = sum(Ypred[Ytrue!=cat]==cat)
  FN = sum(Ypred[Ytrue==cat]!=cat)
  precision = TP/(TP+FP)
  recall = TP/(TP+FN)
  f1score = 2*(precision*recall)/(precision+recall)
  return(f1score)
}

## To save classification metrics
## Fitxxx: metric for training subset. Testxxx: metric for testing subset
FitAccuracy = TestAccuracy <- array(NA,dim=c(SIM,1))  ## Accuracy Rate
FitPrecisionClass = TestPrecisionClass <- array(NA,dim=c(SIM,1,3))  ## Precision per class
FitRecallClass = TestRecallClass <- array(NA,dim=c(SIM,1,3))  ## Recall per class
FitF1ScoreClass = TestF1ScoreClass <- array(NA,dim=c(SIM,1,3))  ## F1-score per class
FitPrecisionMacroAve = TestPrecisionMacroAve <- array(NA,dim=c(SIM,1))  ## Precision Macro Average
FitRecallMacroAve = TestRecallMacroAve <- array(NA,dim=c(SIM,1))  ## Recall Macro Average
FitF1ScoreMacroAve = TestF1ScoreMacroAve <- array(NA,dim=c(SIM,1))  ## F1-score Macro Average
```

Cross-validation

```
##-----------------------------------------------------
for(sim in 1:SIM){ ### BEGIN sim
##-----------------------------------------------------

my_fit = FIT[sim,]   ## training subset
my_test = TEST[sim,]   ## testing subset

## Training data subset
train1 <- trainc %>% filter(ID_fact%in%my_fit, rep==1)  ## repetition=1
train2 <- trainc %>% filter(ID_fact%in%my_fit, rep==2)  ## repetition=2
train3 <- trainc %>% filter(ID_fact%in%my_fit, rep==3)  ## repetition=3

Yc = train1$STATUS_fact   ## categorical response variable for training
n = length(Yc)
G = 3 # classes

## Testing data subset
test1 <- trainc %>% filter(ID_fact%in%my_test, rep==1)  ## repetition=1
test2 <- trainc %>% filter(ID_fact%in%my_test, rep==2)  ## repetition=2
test3 <- trainc %>% filter(ID_fact%in%my_test, rep==3)  ## repetition=3

Yc.new = test1$STATUS_fact   ## categorical response variable for testing
n.new = length(Yc.new)

## Delete variables which are not used
train1 <- train1 %>% select(-c(rep,ID_fact))
train2 <- train2 %>% select(-c(rep,ID_fact))
train3 <- train3 %>% select(-c(rep,ID_fact))
test1 <- test1 %>% select(-c(rep,ID_fact))
test2 <- test2 %>% select(-c(rep,ID_fact))
test3 <- test3 %>% select(-c(rep,ID_fact))

##-----------------------------------------------------
## Algorithm FESPAE
## Feature space partition ensemble model for replication
##-----------------------------------------------------

## Algo1: The feature space is randomly partitioned into M subspaces, {S1,S2,...,SM}

K0 = 4  ## sub-spaces
k = 32  ## explanatory variables
k2 = round(k/K0)
space = 1:k
subspaces = rep(list(rep(NA,k2)),K0)  ## Subspaces
set.seed(12345)
for(j in 1:(K0-1)){
    space1 = sample(space, size=k2, replace=FALSE)
    space = setdiff(space,space1)
    subspaces[[j]] = space1[order(space1)]
}
space1 = space
```

```r
subspaces[[K0]] = space1[order(space1)]
# 32 features = 1x32, 2x16, 4x8,

##-------------------------------------------------
## Algo2: for feature subspace m = 1 to M do

pred.vgam = array(NA,dim=c(n,K0,3))  ## 3 repetitions
pred.new.vgam = array(NA,dim=c(n.new,K0,3))  ## 3 repetitions
##-------------------------------------------------
## Algo3: for replication j = 1 to J do

## REPLICATION j=1:
for(parti1 in 1:K0){  ## partition of the subspaces
train1_par = train1[,c(subspaces[[parti1]],k+1)]
test1_par = test1[,c(subspaces[[parti1]],k+1)]

## Algo4: Fit a classifier $T(xj,z)$, $xj\in Sm$, to the training data
mod1 <- tune( "svm",  STATUS_fact ~ . ,
              data = train1_par,
              kernel = "polynomial", degree=3,
              ranges = list(cost=c(0.01,0.1,0.5,1,5,10,20,50)) )
## summary(mod1)
mejor_mod1 <- mod1$best.model

## Algo5: Compute the C response probabilities {\pi^(m,j)_{ic}}, for i=1,...,n.
## Predictions
predict1 <- predict(mejor_mod1, newdata = train1_par)
predict1.new <- predict(mejor_mod1, newdata = test1_par)

pred.vgam[,parti1,1] = predict1
pred.new.vgam[,parti1,1] = predict1.new
}

## REPLICATION j=2:
for(parti2 in 1:K0){  ## partition of the subspaces
train2_par = train2[,c(subspaces[[parti2]],k+1)]
test2_par = test2[,c(subspaces[[parti2]],k+1)]

## Algo4: Fit a classifier $T(xj,z)$, $xj\in Sm$, to the training data
mod2 <- tune( "svm",  STATUS_fact ~ . ,
              data = train2_par,
              kernel = "polynomial", degree=3,
              ranges = list(cost=c(0.01,0.1,0.5,1,5,10,20,50)) )
## summary(mod2)
mejor_mod2 <- mod2$best.model

## Algo5: Compute the C response probabilities {\pi^(m,j)_{ic}}, for i=1,...,n.
## Predictions
predict2 <- predict(mejor_mod2, newdata = train2_par)
predict2.new <- predict(mejor_mod2, newdata = test2_par)

pred.vgam[,parti2,2] = predict2
pred.new.vgam[,parti2,2] = predict2.new
```

```
}

## REPLICATION j=3:
for(parti3 in 1:K0){  ## partition of the subspaces
train3_par = train3[,c(subspaces[[parti3]],k+1)]
test3_par = test3[,c(subspaces[[parti3]],k+1)]

## Algo4: Fit a classifier $T(xj,z)$, $xj\in Sm$, to the training data
mod3 <- tune( "svm",  STATUS_fact ~ . ,
              data = train3_par,
              kernel = "polynomial", degree=3,
              ranges = list(cost=c(0.01,0.1,0.5,1,5,10,20,50)) )
## summary(mod3)
mejor_mod3 <- mod3$best.model

## Algo5: Compute the C response probabilities {\pi^(m,j)_{ic}}, for i=1,...,n.
## Predictions
predict3 <- predict(mejor_mod3, newdata = train3_par)
predict3.new <- predict(mejor_mod3, newdata = test3_par)

pred.vgam[,parti3,3] = predict3
pred.new.vgam[,parti3,3] = predict3.new
}

##-------------------------------------------------
## Algo6: End for replication j = 1 to J
## Algo7: End for feature subspace m = 1 to M
##-------------------------------------------------
## Algo8: Output: compute the response probabilities $\pi_{ic} = mean({\pi^(m,j)_{ic}})
##-------------------------------------------------
## Algo8: Output: compute the response category T*(x,z) = arg max {\pi_{ic}}

pred.vgam_max = array(NA,dim=n)
for(i in 1:n){
  pred.vgam_max[i] = Mode(pred.vgam[i,,])
}
### Predict new subjects
pred.new.vgam_max = array(NA,dim=n.new)
for(i in 1:n.new){
  pred.new.vgam_max[i] = Mode(pred.new.vgam[i,,])
}

##-------------------------------------------------
## End FESPAE
##-------------------------------------------------
## Classification Metrics for models predicting nominal outcomes

## Accuracy Rate
FitAccuracy[sim,] = c(sum(Yc==pred.vgam_max)/n)

TestAccuracy[sim,] = c(sum(Yc.new==pred.new.vgam_max)/n.new)

## Precision
```

```
for(cate in 1:3){
  FitPrecisionClass[sim,1, cate] = fn_precision_class(Yc, pred.vgam_max, cate)
  TestPrecisionClass[sim,1, cate] = fn_precision_class(Yc.new, pred.new.vgam_max, cate)
}
FitPrecisionMacroAve[sim, 1] = mean(FitPrecisionClass[sim, 1,])
TestPrecisionMacroAve[sim,1] = mean(TestPrecisionClass[sim,1,])

## Recall
for(cate in 1:3){
  FitRecallClass[sim,1, cate] = fn_recall_class(Yc, pred.vgam_max, cate)
  TestRecallClass[sim,1, cate] = fn_recall_class(Yc.new, pred.new.vgam_max, cate)
}
FitRecallMacroAve[sim, 1] = mean(FitRecallClass[sim, 1,])
TestRecallMacroAve[sim,1] = mean(TestRecallClass[sim,1,])

## F1-Score
for(cate in 1:3){
  FitF1ScoreClass[sim,1, cate]= fn_f1score_class(Yc, pred.vgam_max, cate)
  TestF1ScoreClass[sim,1, cate] = fn_f1score_class(Yc.new, pred.new.vgam_max, cate)
}
FitF1ScoreMacroAve[sim, 1] = mean(FitF1ScoreClass[sim, 1,])
TestF1ScoreMacroAve[sim,1] = mean(TestF1ScoreClass[sim,1,])

##------------------------------------------------
} ## END sim
##------------------------------------------------
```

Results

Accuracy Rate

```
columna = c("ensemble")
renglon = c("fit_mean","fit_sd","test_mean","test_sd")

summary(FitAccuracy)
```

```
##        V1
##  Min.   :0.7679
##  1st Qu.:0.8616
##  Median :0.8988
##  Mean   :0.8961
##  3rd Qu.:0.9301
##  Max.   :0.9881
```

```
apply(FitAccuracy,2,"sd")
```

```
## [1] 0.04749427
```

```
summary(TestAccuracy)
```

```
##        V1
##  Min.   :0.4737
##  1st Qu.:0.5614
##  Median :0.5965
##  Mean   :0.6012
##  3rd Qu.:0.6360
##  Max.   :0.7368
```

```
apply(TestAccuracy,2,"sd")
```

```
## [1] 0.05701816
```

```
RESaccuracy <- rbind(apply(FitAccuracy,2,"mean"), apply(FitAccuracy,2,"sd"),
                     apply(TestAccuracy,2,"mean"),apply(TestAccuracy,2,"sd"))
colnames(RESaccuracy) = columna
rownames(RESaccuracy) = renglon
write.csv(RESaccuracy, file=paste0(archivo,"_accuracy",".csv"))
```

Precision Macro Average

```
summary(FitPrecisionMacroAve)
```

```
##        V1
##  Min.   :0.8469
##  1st Qu.:0.8966
##  Median :0.9187
##  Mean   :0.9187
##  3rd Qu.:0.9386
##  Max.   :0.9885
```

```
apply(FitPrecisionMacroAve,2,"sd")
```

```
## [1] 0.02991854
```

```
summary(TestPrecisionMacroAve)
```

```
##        V1
##  Min.   :0.5333
##  1st Qu.:0.6348
##  Median :0.6849
##  Mean   :0.6851
##  3rd Qu.:0.7342
##  Max.   :0.8211
```

```
apply(TestPrecisionMacroAve,2,"sd")
```

```
## [1] 0.06253809
```

```
RESprecision <- rbind(apply(FitPrecisionMacroAve,2,"mean"), apply(FitPrecisionMacroAve,2,"sd"),
                      apply(TestPrecisionMacroAve,2,"mean"),apply(TestPrecisionMacroAve,2,"sd"))
colnames(RESprecision) = columna
rownames(RESprecision) = renglon
write.csv(RESprecision, file=paste0(archivo,"_precision",".csv"))
```

Recall Macro Average

```
summary(FitRecallMacroAve)
```

```
##        V1
##  Min.   :0.7679
##  1st Qu.:0.8616
##  Median :0.8988
##  Mean   :0.8961
##  3rd Qu.:0.9301
##  Max.   :0.9881
```

```
apply(FitRecallMacroAve,2,"sd")
```

```
## [1] 0.04749427
```

```
summary(TestRecallMacroAve)
```

```
##        V1
##  Min.   :0.4737
##  1st Qu.:0.5614
##  Median :0.5965
##  Mean   :0.6012
##  3rd Qu.:0.6360
##  Max.   :0.7368
```

```
apply(TestRecallMacroAve,2,"sd")
```

```
## [1] 0.05701816
```

```
RESrecall <- rbind(apply(FitRecallMacroAve,2,"mean"), apply(FitRecallMacroAve,2,"sd"),
                   apply(TestRecallMacroAve,2,"mean"),apply(TestRecallMacroAve,2,"sd"))
colnames(RESrecall) = columna
rownames(RESrecall) = renglon
write.csv(RESrecall, file=paste0(archivo,"_recall",".csv"))
```

F1-Score Macro Average

```
summary(FitF1ScoreMacroAve)
```

```
##        V1
##  Min.   :0.7739
##  1st Qu.:0.8648
##  Median :0.9010
##  Mean   :0.8983
##  3rd Qu.:0.9312
##  Max.   :0.9881
```

```
apply(FitF1ScoreMacroAve,2,"sd")
```

```
## [1] 0.04599244
```

```
summary(TestF1ScoreMacroAve)
```

```
##        V1
##  Min.   :0.4542
##  1st Qu.:0.5492
##  Median :0.5914
##  Mean   :0.5936
##  3rd Qu.:0.6334
##  Max.   :0.7445
```

```
apply(TestF1ScoreMacroAve,2,"sd")
```

```
## [1] 0.06290705
```

```
RESf1score <- rbind(apply(FitF1ScoreMacroAve,2,"mean"), apply(FitF1ScoreMacroAve,2,"sd"),
                    apply(TestF1ScoreMacroAve,2,"mean"),apply(TestF1ScoreMacroAve,2,"sd"))
colnames(RESf1score) = columna
rownames(RESf1score) = renglon
write.csv(RESf1score, file=paste0(archivo,"_f1score",".csv"))
```