Supplementary Material Online

Automatic classification of open-ended questions: check-all-that-apply questions

Schonlau, Matthias, University of Waterloo, Canada

Gweon, Hyukjun, Western University, Canada

Wenemark, Marika, Linköping University, Sweden and Centre for Organisational Support and Development, Region Östergötland, Sweden

Appendix A

```
# The following is R codes for implementing BR, ECC and RAKEL for multi-label data
# For each method, the function outputs the prediction result (as a data.frame) for the test data.
# The following codes use the 'e1071' package for the implementation of SVMs.

BR <- function(train,test,n_label)
{
  pred_br <- as.data.frame(matrix(0,nrow=nrow(test),ncol=n_label))
  for(i in 1:n_label)
  {
    tr <- cbind(y=as.factor(train[,i]),train[,-c(1:n_label)])
    ts <- cbind(y=as.factor(test[,i]),test[,-c(1:n_label)])
    if(length(table(train[,i]))>1)
    {
       m <- svm(y~.,data=tr,kernel="linear",scale=FALSE,probability=TRUE)
       pred_br[,i] <- as.numeric(predict(m,ts)) - 1
    }
  }
  return(pred_br)
}
```

```
ECC <- function(train,test,n_label,n_cc=10)
{
 M2 <- M4 <- list(1)
 n_tr <- nrow(train)
 lname <- as.vector(colnames(train)[1:n_label])
 zero_score_ecc1 <- zero_score_ecc2 <- matrix(0,nrow=n_cc,ncol=nrow(test))
 for(kk in 1:n_cc)
 {
  M2[[kk]] <- M4[[kk]] <- as.data.frame(matrix(0,nrow=nrow(test),ncol=n_label))
  a2 <- sample(n_tr, n_tr, replace=TRUE)
  tr_cc2 <- train[a2,]
  ind <- sample(n_label,n_label,replace=FALSE)
  CC_tr <- tr_cc2[,ind]
  CC_ts <- test[,ind]
  X_tr <- tr_cc2[,-(1:n_label)]
  X_ts <- test[,-(1:n_label)]
  tr_cc <- cbind(CC_tr,X_tr)
  ts_cc <- cbind(CC_ts,X_ts)
  xcol <- ncol(tr_cc) - n_label
  lname_cc <- lname[ind]
  for(i in 1:n_label)
  {
   if(i==1)
   {
    tr <- cbind(y=as.factor(tr_cc[,i]),tr_cc[,-c(1:n_label)])
    ts <- cbind(y=as.factor(ts_cc[,i]),ts_cc[,-c(1:n_label)])
    if(length(table(tr_cc[,i]))>1)
    {
       m <- svm(y~.,data=tr,kernel="linear",scale=FALSE,probability=TRUE)
       M2[[kk]][,i] <- as.numeric(predict(m,ts)) - 1
```

```
    }
   }
   if(i>1)
   {
    X_tr <- tr[,-1]
    X_ts <- ts[,-1]
    add <- tr_cc[,(i-1)]
    tr <- cbind(y=as.factor(tr_cc[,i]),X_tr,add)
    colnames(tr)[ncol(tr)] <- lname_cc[(i-1)]
    add <- round(M2[[kk]][,(i-1)])
    ts <- cbind(y=as.factor(ts_cc[,i]),X_ts,add)
    colnames(ts)[ncol(ts)] <- lname_cc[(i-1)]
    if(length(table(tr_cc[,i]))>1)
    {
       m <- svm(y~.,data=tr,kernel="linear",scale=FALSE,probability=TRUE)
       M2[[kk]][,i] <- as.numeric(predict(m,ts)) - 1
    }
   }
  }
  for(i in 1:n_label)
  {
   M4[[kk]][,ind[i]] <- M2[[kk]][,i]
  }
}
W <- as.data.frame(matrix(0,nrow=nrow(test),ncol=n_label))
for(i in 1:length(M4))
{
  W <- W + M4[[i]]
}
pred_M <- W/length(M4)
```

```
 train_label <- train[,1:n_label]
 tr_lc <- sum(apply(train_label,1,sum))/nrow(train_label)
 threshold <- seq(0,1,by=0.01)
 a <- length(threshold)
 dif <- numeric(a)
 D <- 1000
 for(j in 1:a)
 {
  pred_M3 <- 1*(pred_M>=threshold[j])
  ts_lc <- sum(apply(pred_M3,1,sum))/nrow(pred_M3)
  dif[j] <- abs(ts_lc - tr_lc)
 }
 min_j <- which.min(dif)
 pred_ecc <- 1*(pred_M>=threshold[min_j])
 return(pred_ecc)
}
```

```
RAKEL <- function(train,test,n_label,mm=2*length)
{
 c1 <- rep(c(0,1),4)
 c2 <- rep(c(0,0,1,1),2)
 c3 <- rep(c(0,0,1,1),each=2)
 chart <- matrix(c(c1,c2,c3),nrow=8)
 M_rakel <- combn(n_label,3)
 ind_rakel <- sample(ncol(M_rakel),mm,replace=FALSE)
 pred_rakel <- matrix(0,nrow=nrow(test),ncol=n_label)

 for(k in 1:mm)
 {
  y.tr <- as.matrix(train[,1:n_label])
  y.ts <- as.matrix(test[,1:n_label])
  sub.y <- y.tr[,M_rakel[,ind_rakel[k]]]
  sub.ts.y <- y.ts[,M_rakel[,ind_rakel[k]]]
  for(i in 1:3)
  {
   lp.y <- sub.y[,1]*100 + sub.y[,2]*10 + sub.y[,3]
   lp.ts.y <- sub.ts.y[,1]*100 + sub.ts.y[,2]*10 + sub.ts.y[,3]
  }
  mul.tr <- cbind(y=as.factor(lp.y),train[,-(1:n_label)])
  mul.ts <- cbind(y=as.factor(lp.ts.y),test[,-(1:n_label)])

  m_rakel <- svm(y~.,data=mul.tr,kernel="linear",scale=FALSE,probability=TRUE)
  pred <- predict(m_rakel,mul.ts)
  trans <- as.numeric(as.vector(pred))

  pred_mat <- matrix(0,nrow=nrow(test),ncol=3)
  c4 <- c(0,100,10,110,1,101,11,111)
```

```
  i <- 1
  for(i in 1:length(trans))
  {
    ind <- which(trans[i]==c4)
    pred_mat[i,] <- chart[ind,]
  }
  pred_rakel[,M_rakel[,ind_rakel[k]]] <- pred_rakel[,M_rakel[,ind_rakel[k]]] + pred_mat
}
temp3 <- numeric(n_label)
pred_rakel2 <- pred_rakel
for(i in 1:n_label)
{
  temp3[i] <- sum(M_rakel[,ind_rakel]==i)
  if(temp3[i]>0) pred_rakel2[,i] <- pred_rakel2[,i]/temp3[i]
  if(temp3[i]==0) pred_rakel2[,i] <- 0
}
pred_rakel2[pred_rakel2>=0.5] <- 1
pred_rakel2[pred_rakel2<0.5] <- 0
pred_rakel <- pred_rakel2
return(pred_rakel)
}
```