```
#==========================================================================
#        Using R: wrapper function for Classification Performance Evaluation
#==========================================================================
#Here, by help of existing course codes, we can develop a user-defined wrapper function
named "wrapper" to produce a series of binary classifier. The function asks for three
inputs of True Values, predicted value, and the probability of being 1. It outputs the
"Confusion Matrix", "Percentage of Concordance", "Percentage of Discordance", "Percentage
of Tied", "Pairs", "AUC", "D Statistic" and "KS Max". Also, it yields three graphs,
"ROC", "Distribution of Predicted Probabilities" and "Lift Chart".

wrapper=function(a, b, c){
  library(ROCR)
  input <- data.frame(True.values=a, Pred.values=b, Pred.probab=c)

  #Confusion Matrix
  confusion.Matrix <- confusionMatrix(input$Pred.values, input$True.values, positive="1")

  #ROC
  pred <- prediction(input$Pred.probab, input$True.values)
  ROC <- performance(pred, "tpr", "fpr")
  plot(ROC, colorize=TRUE, print.cutoffs.at=c(0.25,0.5,0.75));
  abline(0, 1, col="red")

  #Distribution of predicted probabilities values for the true positives and true
negatives
  y1 <- 0
  y2 <- 0
  for (runi in 1:length(pred@predictions)) {
    xx <- density(pred@predictions[[runi]][pred@labels[[runi]]==1])
    t1 <- max(xx$y)
    if(t1 > y1){y1 <- t1}
    yy <- density(pred@predictions[[runi]][pred@labels[[runi]]==0])
    t2 <- max(yy$y)
    if(t2 > y2){y2 <- t2}
  }
  y <- max(y1, y2)

  plot(0,0, type="n", xlim=c(0,1), ylim=c(0,y+1), xlab="Prediction", ylab="Density",
main="Distribution of predicted probabilities")
  for (runi in 1:length(pred@predictions)) {
    lines(density(pred@predictions[[runi]][pred@labels[[runi]]==1]), col="blue")
    lines(density(pred@predictions[[runi]][pred@labels[[runi]]==0]), col="green")
  }

  #Concordant Pairs and AUC
  Con_Dis_Data <- cbind(input$True.values, input$Pred.values)

  ones <- Con_Dis_Data[Con_Dis_Data[,1] == 1,]
  zeros <- Con_Dis_Data[Con_Dis_Data[,1] == 0,]

  conc <- matrix(0, dim(zeros)[1], dim(ones)[1])   #build a matrix of 0's
  disc <- matrix(0, dim(zeros)[1], dim(ones)[1])
  ties <- matrix(0, dim(zeros)[1], dim(ones)[1])
  for (j in 1:dim(zeros)[1]) {
    for (i in 1:dim(ones)[1]) {
      if (ones[i,2]>zeros[j,2]) {conc[j,i]=1}
      else if (ones[i,2]<zeros[j,2]) {disc[j,i]=1}
      else if (ones[i,2]==zeros[j,2]) {ties[j,i]=1}
    }
  }

  Pairs <- dim(zeros)[1]*dim(ones)[1]              #total number of pairs
  PercentConcordance <- (sum(conc)/Pairs)*100
  PercentDiscordance <- (sum(disc)/Pairs)*100
  PercentTied <- (sum(ties)/Pairs)*100
  AUC <- PercentConcordance +(0.5 * PercentTied)

  #D statistic (2009)
  probab.1 <- input[input$True.values==1,]
  probab.0 <- input[input$True.values==0,]
  D.statistic <- mean(probab.1$Pred.probab) - mean(probab.0$Pred.probab)

  #K-S chart (Kolmogorov-Smirnov chart)
  input$group <- cut(input$Pred.probab, seq(1,0,-.1), include.lowest=TRUE)
  xtab <- table(input$group, input$True.values)
  xtab

  #make empty dataframe
  KS <- data.frame(Group=numeric(10), CumPct0=numeric(10), CumPct1=numeric(10),
Dif=numeric(10))

  #fill data frame with information: Group ID, Cumulative % of 0's, of 1's and Difference
  for (i in 1:10) {
    KS$Group[i] <- i
    KS$CumPct0[i] <- sum(xtab[1:i,1]) / sum(xtab[,1])
    KS$CumPct1[i] <- sum(xtab[1:i,2]) / sum(xtab[,2])
    KS$Dif[i] <- abs(KS$CumPct0[i] - KS$CumPct1[i])
  }

  KS.max <- KS[KS$Dif==max(KS$Dif),]

  maxGroup <- KS[KS$Dif==max(KS$Dif),][1,1]

  #and the K-S chart
  ggplot(data=KS)+
    geom_line(aes(Group,CumPct0), color="blue")+
    geom_line(aes(Group,CumPct1), color="red")+
    geom_segment(x=maxGroup,xend=maxGroup, y=KS$CumPct0[maxGroup],
yend=KS$CumPct1[maxGroup])+
    labs(title="K-S Chart", x="Deciles", y="Cumulative Percent")

  #Lift Chart
  Lift.chart <- performance(pred, "lift", "rpp")
  plot(Lift.chart, colorize=TRUE, print.cutoffs.at=c(0.25,0.5,0.75))

  #Outputs
  return(list("Confusion Matrix"=confusion.Matrix,
              "Percent Concordance"=PercentConcordance,
              "Percent Discordance"=PercentDiscordance,
              "Percent Tied"=PercentTied,
              "Pairs"=Pairs,
              "AUC"=AUC,
              "D statistic"=D.statistic,
              "KS.max"=KS.max))
}
```