HW2_solution

```
library(ISLR)
library(nnet)
library(MASS)
library(e1071)
library(caret)
library(glmnet)
library(class)


data(OJ)
y <- OJ[, 1]
x <- scale(OJ[, -c(1,11:14,17)])
thre = seq(0,1,0.001)

scorefun <- function(decision,y){

  FP <- sum(decision[y=="CH"]=="MM")
  TN <- sum(decision[y=="CH"]=="CH")
  FN <- sum(decision[y=="MM"]=="CH")
  TP <- sum(decision[y=="MM"]=="MM")
  ACC <- (TP+TN)/(TP+FP+TN+FN)
  F1 <- (2*TP)/(2*TP+FP+FN)
  demons <- (TP+FP)*(TP+FN)*(TN+FP)*(TN+FN)
  MCC <- ifelse(demons == 0, 0,(TP*TN-FP*FN)/sqrt(demons))
  return(cbind(ACC=ACC, F1=F1, MCC=MCC))
}
```

```r
fun <- function(type=c('lr','lda','qda','nb','knn'),thre,x,y,z,v,w,s,pred,pred_test,...){

  if (type == 'knn'){
    score <- matrix(NA, length(thre), 3)
    score_test <- matrix(NA,1,3)
    colnames(score) <- colnames(score_test) <- c('ACC','F1','MCC')

    for (i in 1:length(thre)) {
      knn.pred <-knn(x, y, z, k = thre[i])
      score[i,] <- scorefun(knn.pred,v)
    }

    score_wh <- apply(score, 2, function(x) list(which(x==max(x))))
    score_thre<- cbind(ACC=min(thre[unlist(score_wh$ACC)]),
                       F1=min(thre[unlist(score_wh$F1)]),
                       MCC=min(thre[unlist(score_wh$MCC)]))
    for (j in 1:3){
      knn.pred_test <-knn(x, s, z, k=score_thre[,j])
      score_test[,j]<- scorefun(knn.pred_test,w)[j]
    }
  }else{

    score <- matrix(NA,length(thre),3)
    score_test <- matrix(NA,1,3)
    colnames(score) <- colnames(score_test) <- c('ACC','F1','MCC')

    for (i in 1:length(thre)) {
      decision = rep("CH", length(v))
      decision[pred > thre[i]] = "MM"
      score[i,] <- scorefun(decision,v)
    }

    score_wh <- apply(score,2,function(x) list(which(x==max(x))))
    score_thre<- cbind(ACC=mean(thre[unlist(score_wh$ACC)]),
                       F1=mean(thre[unlist(score_wh$F1)]),
                       MCC=mean(thre[unlist(score_wh$MCC)]))
    for (j in 1:3){
      decision_test <- rep("CH", length(y_test))
      decision_test[pred_test > score_thre[j]] = "MM"
      score_test[,j] <- scorefun(decision_test,w)[j]
    }
  }

  out <- list(ACC=score[,1], F1=score[,2], MCC=score[,3],
              thresholds=score_thre, score=score_test)
  return(out)
}
```

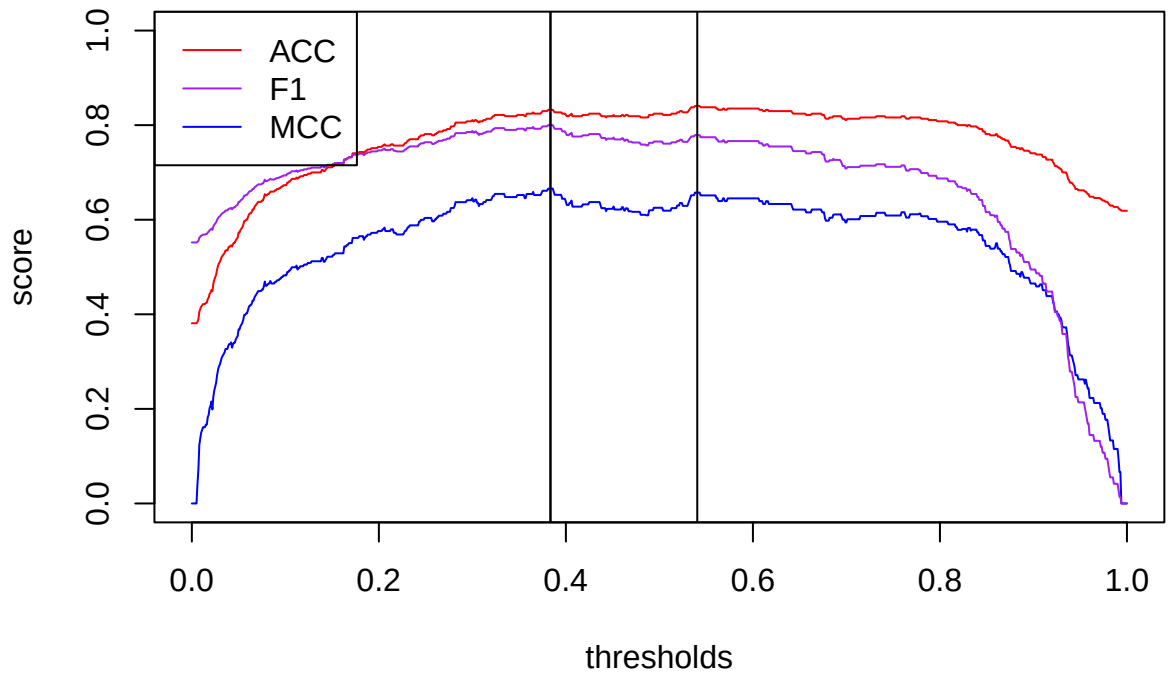
```r
set.seed(1111)
M <- sample(rep(c(-1, 0, 1), c(600, 370, 100)))
x_tran <- x[M==-1,];y_tran <- y[M==-1]
x_vald <- x[M==0,];y_vald <- y[M==0]
x_test <- x[M==1,];y_test <- y[M==1]

tran_data <- data.frame(y = y_tran, x_tran)
vald_data <-  data.frame(y = y_vald, x_vald)
test_data <- data.frame(y = y_test, x_test)

thre <- seq(0,1,0.001)
```

Question 1

```r
LR <- glm(y ~ ., data=tran_data, family="binomial")
pred_LR <- predict(LR, vald_data,type='response')
pred_LR_test <- predict(LR, test_data,type='response')
Q1 <- fun(type='lr',thre=thre,v=y_vald,pred=pred_LR,w=y_test,pred_test=pred_LR_test)
Q1s <- Q1$thresholds
Q1s
```

```
##         ACC     F1    MCC
## [1,] 0.5405 0.3835 0.3835
```

```
plot(thre,Q1$MCC,type='l',xlab='thresholds',ylab='score',col='blue',ylim=c(0,1))
lines(thre,Q1$ACC,col='red')
lines(thre,Q1$F1,col='purple')
abline(v=Q1$thresholds[1])
abline(v=Q1$thresholds[2])
abline(v=Q1$thresholds[3])
legend('topleft',c('ACC','F1','MCC'), col=c('red','purple','blue'),lty=1)
```



Question 2

```
Q2 <- Q1$score
Q2
```

```
##        ACC        F1       MCC
## [1,] 0.87 0.7901235 0.6688529
```

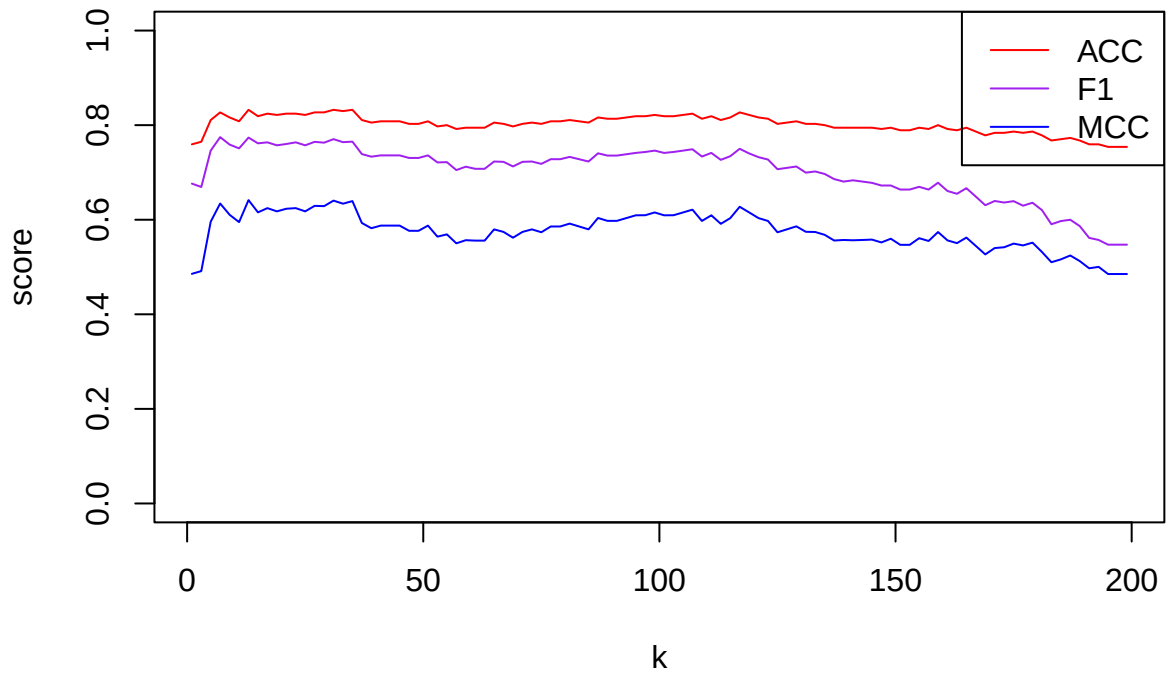
Question 3

```
#LDA
LDA <- lda(y ~ ., data=tran_data)
pred_LDA <- predict(LDA, vald_data)$posterior[,2]
pred_LDA_test <- predict(LDA, test_data)$posterior[,2]
LDA1 <- fun(type='lda',thre=thre,pred=pred_LDA,pred_test=pred_LDA_test,
            v=y_vald,w=y_test)$score

#QDA
QDA <- qda(y ~ ., data=tran_data)
pred_QDA <- predict(QDA, vald_data)$posterior[,2]
pred_QDA_test <- predict(QDA, test_data)$posterior[,2]
QDA1 <- fun(type='qda',thre=thre,pred=pred_QDA,pred_test=pred_QDA_test,
            v=y_vald,w=y_test)$score

#NB
NB <- naiveBayes(y ~., data=tran_data)
pred_NB <- predict(NB,vald_data,type='raw')[,2]
pred_NB_test <- predict(NB, test_data,type='raw')[,2]
NB1 <- fun(type='nb',thre=thre,pred=pred_NB,pred_test=pred_NB_test,
           v=y_vald,w=y_test)$score

Q3 <- list(LDA=LDA1, QDA=QDA1, NB=NB1)
Q3
```

```
## $LDA
##       ACC        F1       MCC
## [1,] 0.85 0.7804878 0.6378677
##
## $QDA
##       ACC        F1       MCC
## [1,] 0.74 0.7042254 0.5416138
##
## $NB
##       ACC        F1       MCC
## [1,] 0.79 0.7142857 0.5355293
```

Question 4

```
k <- seq(1,200,2)
knn <- fun(type='knn',thre=k,x=x_tran,y=x_vald,z=y_tran,v=y_vald,w=y_test,s=x_test)

plot(k,knn$MCC,type='l',xlab='k',ylab='score',col='blue',ylim=c(0,1))
lines(k,knn$ACC,col='red')
lines(k,knn$F1,col='purple')
legend('topright',c('ACC','F1','MCC'), col=c('red','purple','blue'),lty=1)
```




```
Q4 <-knn$score
Q4
```

```
##        ACC        F1       MCC
## [1,] 0.81 0.7887324 0.5852924
```

Question 5

```
set.seed(1234)
M <- rep(c(-1, 0, 1), c(600, 370, 100))
M_ <- apply(matrix(M, length(M), 100), 2, sample)

thre <- seq(0,1,0.001)
k <- seq(1,200,2)
Q5 <- array(NA,c(100,3,5))
colnames(Q5) <- c('ACC','F1','MCC')

for(j in 1:100){
  M <- M_[,j]
  x_tran <- x[M==-1,];y_tran <- y[M==-1]
  x_vald <- x[M==0,];y_vald <- y[M==0]
  x_test <- x[M==1,];y_test <- y[M==1]

  tran_data <- data.frame(y = y_tran, x_tran)
  vald_data <- data.frame(y = y_vald, x_vald)
  test_data <- data.frame(y = y_test, x_test)

  #LR
  LR <- glm(y ~ ., data=tran_data, family="binomial")
  pred_LR <- predict(LR, vald_data,type='response')
  pred_LR_test <- predict(LR, test_data,type='response')
  Q5[j,,1] <- fun(type='lr',thre=thre,v=y_vald,pred=pred_LR,
                  w=y_test,pred_test=pred_LR_test)$score

  #LDA
  LDA <- lda(y ~ ., data=tran_data)
  pred_LDA <- predict(LDA, vald_data)$posterior[,2]
  pred_LDA_test <- predict(LDA, test_data)$posterior[,2]
  Q5[j,,2]<- fun(type='lda',thre=thre,pred=pred_LDA,pred_test=pred_LDA_test,
                 v=y_vald,w=y_test)$score

  #QDA
  QDA <- qda(y ~ ., data=tran_data)
  pred_QDA <- predict(QDA, vald_data)$posterior[,2]
  pred_QDA_test <- predict(QDA, test_data)$posterior[,2]
  Q5[j,,3]<- fun(type='qda',thre=thre,pred=pred_QDA,pred_test=pred_QDA_test,
                 v=y_vald,w=y_test)$score

  #NB
  NB <- naiveBayes(y ~., data=tran_data)
  pred_NB <- predict(NB,vald_data,type='raw')[,2]
  pred_NB_test<- predict(NB, test_data,type='raw')[,2]
  Q5[j,,4]<- fun(type='nb',thre=thre,pred=pred_NB,pred_test=pred_NB_test,
                 v=y_vald,w=y_test)$score

  #KNN
  Q5[j,,5]<- fun(type='knn',thre=k,x=x_tran,y=x_vald,z=y_tran,v=y_vald,
                 w=y_test,s=x_test)$score
}
```

```r
Q5_table <- apply(Q5,c(2,3),mean)
colnames(Q5_table) <- c('LR','LDA','QDA','NB','KNN')
Q5_table
```

```
##            LR       LDA       QDA        NB       KNN
## ACC 0.8222000 0.8176000 0.7885000 0.7792000 0.7902000
## F1  0.7719514 0.7721626 0.7383222 0.7109886 0.7206220
## MCC 0.6263166 0.6204818 0.5544242 0.5285466 0.5531418
```