CSE455/CSE552 – Machine Learning (Spring 2015)

Homework #1 Report

Okan Akdoğan (121044017)

Part 1:

Code:

```
# Okan Akdogan 23/02/2016



#distance funcs for use in my knn implementation

euclid_dist <- function(p1,p2){

  if (length(p1)!=length(p2))
    return(-1)
  sum_sq <- 0
  for (i in 1:length(p1)) {
    sum_sq <- sum_sq + (p1[i]-p2[i])^2
  }
  return(as.numeric( sqrt(sum_sq)))
}

manhattan_dist <- function(p1,p2){
  if (length(p1)!=length(p2))
    return(-1)
  sum_sq <- 0
  for (i in 1:length(p1)) {
    sum_sq <- sum_sq + abs(p1[i]-p2[i])
  }
  return(as.numeric( sqrt(sum_sq)))

}
```

```
# knn implementation

myknn <- function(train,trainlabels,k,test,dist_func){

  #create dump vector to store result
  results <- c(1:as.numeric(nrow(test)))

  # for all test input
  for(t in 1:nrow(test)){

    # distance matrix to hold min k distances
    dist_mat <- matrix(data=Inf,k,2)

    #init matrix index column
    for(i in 1:k)
      dist_mat[i,2] <- 0

    # calculate in all train data
    for (v_ind in 1:nrow(train)) {
      # calc distance
      dist<- dist_func(train[v_ind,],test[t,])

      # check if you put in distance matrix
      # matrix holds min values distances and ther indicies
      for(i in 1:k){
        if (dist_mat[i,1] > dist ){
          dist_mat[i,1] <-dist
          dist_mat[i,2] <-v_ind
          break
        }
      }
    }

    #match min distance indicies with their labels
    res <- c(1:k) #empty vec

    for(i in 1:k){
      res[i]<- trainlabels[dist_mat[i,2]]
    }

    #table for see to frequency of labels
    res_t <- table(res)
    #select max frequency label
    results[t] <-as.numeric( names(which.max(res_t)))
  }
  #return all test results
  return(results)
}

#load leaf data
leaf_data <- read.table('leaf.dat',header = FALSE)

#Prepare Datas

normalize <- function(x){
  num <- x - min(x)
  denom <- max(x) - min(x)
  return (num/denom)
}



#LEAF PROCESS
#load leaf data
leaf_data <- read.table('leaf.dat',header = FALSE)

#shuffle
shuffle_leaf <- leaf_data[sample(nrow(leaf_data)),]

#normalize
```

```
norm_leaf <- as.data.frame(lapply(shuffle_leaf[2:16], normalize))
label_leaf <- shuffle_leaf[,1]
summary(norm_leaf)



#IRIS PROCESS


#Randomly shuffle the data
shuffle_iris<-iris[sample(nrow(iris)),]

# normalize iris data
norm_iris <- as.data.frame(lapply(shuffle_iris[1:4], normalize))
label_iris <- shuffle_iris[,5]
summary(norm_iris)



testWith_5_CrossValid <- function( data, labels){

  #Create 5 equally size folds
  folds <- cut(seq(1,nrow(data)),breaks=5,labels=FALSE)

  #Perform 5 fold cross validation with Euclidean and Manhattan distances

  # hold funcnumber for print
  fnum <- 0

  for(f in c(euclid_dist,manhattan_dist)){
    fnum<-fnum+1

    if(fnum==1){
      print("knn with Euclidean Distance")
    }else{
      print("knn with Manhattan Distance")
    }
    for(i in 1:5){
      #Segement your data by fold using the which() function
      testIndexes <- which(folds==i,arr.ind=TRUE)
      testData <- data[testIndexes, ]
      test_Labels <- labels[testIndexes]
      trainData <- data[-testIndexes, ]
      train_Labels <- labels[-testIndexes]

      system.time( myknn_res <- myknn(train=trainData,trainlabels = train_Labels,k=5,dist_func = f,test =testData))
      #print(myknn_res)
      # conf matrix
      conf <-table(test_Labels,myknn_res)

      cat("cross valid fold:",i,"\n")
      #print(conf)
      print(confMatrixMulti(conf))

    }
  }
}
confMatrixMulti <- function( confTable){

  confs <- matrix(data=NA, nrow = nrow(confTable),ncol = 4 )

  for (r in 1:nrow(confTable)) {

    TP <- 0
    FP <- 0

    for (c in 1:ncol(confTable)) {

      if( r==c )
       TP <- TP + confTable[r,c]
      else
       FP<- FP + confTable[r,c]
    }
    confs[r,1]=TP
    confs[r,2]=FP
  }

  for (r in 1:nrow(confs)) {

    TN <- 0
    FN <- 0
    for (or in 1:nrow(confs)) {
       if(r==or){
         #skip for this row
       }else{
         TN <- TN + confs[or,1]
         FN <- FN + confs[or,2]
       }
    }
    confs[r,3] <- TN
    confs[r,4] <- FN
  }

  colnames(confs) <- c('TP','FP','TN','FN')
  return(confs)
}

#test starts here

testWith_5_CrossValid(norm_iris,label_iris)
testWith_5_CrossValid(norm_leaf,label_leaf)

print("Code ends here")
```

Results:

```
--- Iris ---
cross valid fold: 1
     TP FP TN FN
```

```
[1,]  7  0 19  4
[2,] 12  2 14  2
[3,]  7  2 19  2
cross valid fold: 2
     TP FP TN FN
[1,] 16  0 13  1
[2,]  5  0 24  1
[3,]  8  1 21  0
cross valid fold: 3
     TP FP TN FN
[1,]  9  0 21  0
[2,]  9  0 21  0
[3,] 12  0 18  0
cross valid fold: 4
     TP FP TN FN
[1,] 12  0 18  0
[2,]  8  0 22  0
[3,] 10  0 20  0
cross valid fold: 5
     TP FP TN FN
[1,]  6  0 22  2
[2,] 12  2 16  0
[3,] 10  0 18  2
```

--- leaf----

```
"knn with Euclidean Distance"
cross valid fold: 1
      TP FP TN FN
 [1,]  5  1  7 55
 [2,]  1  1 11 55
 [3,]  1  2 11 54
 [4,]  0  1 12 55
 [5,]  2  0 10 56
 [6,]  0  1 12 55
 [7,]  3  0  9 56
 [8,]  0  1 12 55
 [9,]  0  1 12 55
[10,]  0  3 12 53
[11,]  0  3 12 53
[12,]  0  3 12 53
[13,]  0  3 12 53
[14,]  0  1 12 55
[15,]  0  1 12 55
[16,]  0  2 12 54
[17,]  0  2 12 54
[18,]  0  2 12 54
[19,]  0  5 12 51
[20,]  0  3 12 53
[21,]  0  2 12 54
[22,]  0  3 12 53
[23,]  0  1 12 55
[24,]  0  1 12 55
[25,]  0  3 12 53
[26,]  0  2 12 54
[27,]  0  3 12 53
[28,]  0  3 12 53
[29,]  0  2 12 54
cross valid fold: 2
      TP FP TN FN
 [1,]  1  1 23 43
```

```
  [2,]  0  3 24 41
  [3,]  0  1 24 43
  [4,]  0  3 24 41
  [5,]  0  2 24 42
  [6,]  0  5 24 39
  [7,]  2  0 22 44
  [8,]  2  2 22 42
  [9,]  2  0 22 44
 [10,]  6  1 18 43
 [11,]  2  1 22 43
 [12,]  1  0 23 44
 [13,]  0  2 24 42
 [14,]  1  2 23 42
 [15,]  1  3 23 41
 [16,]  3  0 21 44
 [17,]  1  0 23 44
 [18,]  1  0 23 44
 [19,]  1  2 23 42
 [20,]  0  2 24 42
 [21,]  0  2 24 42
 [22,]  0  1 24 43
 [23,]  0  3 24 41
 [24,]  0  2 24 42
 [25,]  0  3 24 41
 [26,]  0  1 24 43
 [27,]  0  2 24 42
cross valid fold: 3
       TP FP TN FN
  [1,]  1  0 23 44
  [2,]  1  0 23 44
  [3,]  1  0 23 44
  [4,]  0  3 24 41
  [5,]  0  3 24 41
  [6,]  0  3 24 41
  [7,]  0  3 24 41
  [8,]  1  2 23 42
  [9,]  0  3 24 41
 [10,]  0  5 24 39
 [11,]  1  3 23 41
 [12,]  0  3 24 41
 [13,]  0  1 24 43
 [14,]  1  0 23 44
 [15,]  2  1 22 43
 [16,]  2  2 22 42
 [17,]  0  1 24 43
 [18,]  1  3 23 41
 [19,]  4  0 20 44
 [20,]  1  3 23 41
 [21,]  3  2 21 42
 [22,]  1  0 23 44
 [23,]  0  3 24 41
 [24,]  1  0 23 44
 [25,]  1  0 23 44
 [26,]  2  0 22 44
cross valid fold: 4
       TP FP TN FN
  [1,]  0  2 24 42
  [2,]  2  2 22 42
  [3,]  2  1 22 43
  [4,]  1  2 23 42
  [5,]  4  0 20 44
  [6,]  2  0 22 44
  [7,]  1  0 23 44
  [8,]  2  0 22 44
  [9,]  1  4 23 40
 [10,]  2  0 22 44
 [11,]  1  0 23 44
 [12,]  1  1 23 43
 [13,]  2  0 22 44
 [14,]  1  0 23 44
 [15,]  2  1 22 43
 [16,]  0  2 24 42
 [17,]  0  3 24 41
 [18,]  0  4 24 40
 [19,]  0  1 24 43
 [20,]  0  2 24 42
 [21,]  0  4 24 40
 [22,]  0  1 24 43
 [23,]  0  2 24 42
 [24,]  0  1 24 43
 [25,]  0  2 24 42
 [26,]  0  2 24 42
 [27,]  0  1 24 43
 [28,]  0  3 24 41
 [29,]  0  1 24 43
 [30,]  0  2 24 42
 cross valid fold: 5
       TP FP TN FN
  [1,]  1  0 30 37
  [2,]  2  1 29 36
  [3,]  0  3 31 34
  [4,]  3  0 28 37
  [5,]  1  0 30 37
  [6,]  1  0 30 37
  [7,]  0  1 31 36
  [8,]  1  4 30 33
  [9,]  2  0 29 37
 [10,]  1  3 30 34
 [11,]  1  1 30 36
 [12,]  1  1 30 36
 [13,]  1  3 30 34
 [14,]  1  1 30 36
 [15,]  0  3 31 34
 [16,]  1  0 30 37
 [17,]  0  1 31 36
 [18,]  1  1 30 36
 [19,]  1  2 30 35
 [20,]  2  0 29 37
 [21,]  1  2 30 35
 [22,]  2  1 29 36
 [23,]  0  3 31 34
 [24,]  1  1 30 36
 [25,]  2  1 29 36
 [26,]  2  4 29 33
 [27,]  2  0 29 37
```

Comments:

Knn iris data ise ile güzel sonuçlar Verdi. Ancak leaf datasında iyi sonuçlar alamadım. Confusion matrisleri her bir sınıf için çıkardım.

Part 2:

Code:

```
Part1 ile aynı kod.
```

Results:

```
--- iris ---

knn with Manhattan Distance"
cross valid fold: 1
     TP FP TN FN
[1,]  7  0 20  3
[2,] 13  1 14  2
[3,]  7  2 20  1
cross valid fold: 2
     TP FP TN FN
[1,] 16  0 13  1
[2,]  5  0 24  1
[3,]  8  1 21  0
cross valid fold: 3
     TP FP TN FN
[1,]  9  0 20  1
[2,]  9  0 20  1
[3,] 11  1 18  0
cross valid fold: 4
     TP FP TN FN
[1,] 12  0 18  0
[2,]  8  0 22  0
[3,] 10  0 20  0
cross valid fold: 5
     TP FP TN FN
[1,]  6  0 21  3
[2,] 12  2 15  1
[3,]  9  1 18  2



---leaf ---

knn with Manhattan Distance"
cross valid fold: 1
      TP FP TN FN
 [1,]  3  3  5 57
 [2,]  2  0  6 60
 [3,]  1  2  7 58
 [4,]  0  1  8 59
 [5,]  2  0  6 60
 [6,]  0  1  8 59
 [7,]  0  3  8 57
 [8,]  0  1  8 59
 [9,]  0  1  8 59
[10,]  0  3  8 57
[11,]  0  3  8 57
[12,]  0  3  8 57
[13,]  0  3  8 57
[14,]  0  1  8 59
[15,]  0  1  8 59
[16,]  0  2  8 58
[17,]  0  2  8 58
[18,]  0  2  8 58
```

```
 [19,]  0  5  8 55
 [20,]  0  3  8 57
 [21,]  0  2  8 58
 [22,]  0  3  8 57
 [23,]  0  1  8 59
 [24,]  0  1  8 59
 [25,]  0  3  8 57
 [26,]  0  2  8 58
 [27,]  0  3  8 57
 [28,]  0  3  8 57
 [29,]  0  2  8 58
cross valid fold: 2
      TP FP TN FN
 [1,]  0  2  8 58
 [2,]  0  3  8 57
 [3,]  0  1  8 59
 [4,]  0  3  8 57
 [5,]  0  2  8 58
 [6,]  0  5  8 55
 [7,]  0  2  8 58
 [8,]  0  4  8 56
 [9,]  0  2  8 58
[10,]  0  7  8 53
[11,]  0  3  8 57
[12,]  0  1  8 59
[13,]  0  2  8 58
[14,]  2  1  6 59
[15,]  2  2  6 58
[16,]  2  1  6 59
[17,]  1  0  7 60
[18,]  1  0  7 60
[19,]  0  3  8 57
[20,]  0  2  8 58
[21,]  0  2  8 58
[22,]  0  1  8 59
[23,]  0  3  8 57
[24,]  0  2  8 58
[25,]  0  3  8 57
[26,]  0  1  8 59
[27,]  0  2  8 58
cross valid fold: 3
      TP FP TN FN
 [1,]  1  0  3 64
 [2,]  1  0  3 64
 [3,]  1  0  3 64
 [4,]  0  3  4 61
 [5,]  0  3  4 61
 [6,]  0  3  4 61
 [7,]  0  3  4 61
 [8,]  1  2  3 62
 [9,]  0  3  4 61
[10,]  0  5  4 59
[11,]  0  4  4 60
[12,]  0  3  4 61
[13,]  0  1  4 63
[14,]  0  1  4 63
[15,]  0  3  4 61
[16,]  0  4  4 60
[17,]  0  1  4 63
[18,]  0  4  4 60
[19,]  0  4  4 60
[20,]  0  4  4 60
```

```
[21,]  0  5  4 59
[22,]  0  1  4 63
[23,]  0  3  4 61
[24,]  0  1  4 63
[25,]  0  1  4 63
[26,]  0  2  4 62
cross valid fold: 4
      TP FP TN FN
 [1,]  1  1 30 36
 [2,]  2  2 29 35
 [3,]  2  1 29 36
 [4,]  1  2 30 35
 [5,]  3  1 28 36
 [6,]  1  1 30 36
 [7,]  1  0 30 37
 [8,]  2  0 29 37
 [9,]  2  3 29 34
[10,]  2  0 29 37
[11,]  1  0 30 37
[12,]  1  1 30 36
[13,]  2  0 29 37
[14,]  0  1 31 36
[15,]  3  0 28 37
[16,]  2  0 29 37
[17,]  3  0 28 37
[18,]  2  2 29 35
[19,]  0  1 31 36
[20,]  0  2 31 35
[21,]  0  4 31 33
[22,]  0  1 31 36
[23,]  0  2 31 35
[24,]  0  1 31 36
[25,]  0  2 31 35
[26,]  0  2 31 35
[27,]  0  1 31 36
[28,]  0  3 31 34
[29,]  0  1 31 36
[30,]  0  2 31 35
cross valid fold: 5
      TP FP TN FN
 [1,]  1  0  2 65
 [2,]  2  1  1 64
 [3,]  0  3  3 62
 [4,]  0  3  3 62
 [5,]  0  1  3 64
 [6,]  0  1  3 64
 [7,]  0  1  3 64
 [8,]  0  5  3 60
 [9,]  0  2  3 63
[10,]  0  4  3 61
[11,]  0  2  3 63
[12,]  0  2  3 63
[13,]  0  4  3 61
[14,]  0  2  3 63
[15,]  0  3  3 62
[16,]  0  1  3 64
[17,]  0  1  3 64
[18,]  0  2  3 63
[19,]  0  3  3 62
[20,]  0  2  3 63
[21,]  0  3  3 62
[22,]  0  3  3 62
```

```
[23,]  0  3  3 62
[24,]  0  2  3 63
[25,]  0  3  3 62
[26,]  0  6  3 59
[27,]  0  2  3 63
```

Comments:

Distance fonksiyonu model sonuçlarımı confusion matrislerinde görüldüğü gibi değiştirmiştir. Kbaca incelersek Euclidean farkındaki gibi iyi sonuçlar çıkardı.

Part 3:

Code:

```
# Okan Akdogan 25/02/2016



# tool functions


confMatrixMulti <- function( confTable){


  confs <- matrix(data=NA, nrow = nrow(confTable),ncol = 4 )


  for (r in 1:nrow(confTable)) {


    TP <- 0
    FP <- 0


    for (c in 1:ncol(confTable)) {


      if( r==c )
        TP <- TP + confTable[r,c]
      else
        FP<- FP + confTable[r,c]


    }
    confs[r,1]=TP
    confs[r,2]=FP
  }

  for (r in 1:nrow(confs)) {


    TN <- 0
    FN <- 0
    for (or in 1:nrow(confs)) {
      if(r==or){
        #skip for this row
      }else{
        TN <- TN + confs[or,1]
        FN <- FN + confs[or,2]
      }
    }
    confs[r,3] <- TN
    confs[r,4] <- FN
  }

  colnames(confs) <- c('TP','FP','TN','FN')
  return(confs)
}

#load leaf data
leaf_data <- read.table('leaf.dat',header = FALSE)

#Prepare Datas

normalize <- function(x){
  num <- x - min(x)
  denom <- max(x) - min(x)
  return (num/denom)
}



#LEAF PROCESS
```

```
#shuffle
shuffle_leaf <- leaf_data[sample(nrow(leaf_data)),]

#normalize
norm_leaf <- as.data.frame(lapply(shuffle_leaf[2:16], normalize))
label_leaf <- shuffle_leaf[,1]
summary(norm_leaf)
#IRIS PROCESS



#Randomly shuffle the data
shuffle_iris<-iris[sample(nrow(iris)),]

# normalize iris data
norm_iris <- as.data.frame(lapply(shuffle_iris[1:4], normalize))
label_iris <- shuffle_iris[,5]
summary(norm_iris)



makeSVMTest <- function(data,labels){

  #Create 10 equally size folds
  folds <- cut(seq(1,nrow(data)),breaks=5,labels=FALSE)

  #Perform 5 fold cross validation with Euclidean and Manhattan distances

  #needs lib install with
  #> install.packages("e1071")

  library(e1071)



  for(i in 1:5){
    #Segement your data by fold using the which() function
    testIndexes <- which(folds==i,arr.ind=TRUE)
    testData <- data[testIndexes, ]
    test_Labels <- labels[testIndexes]
    trainData <- data[-testIndexes, ]
    train_Labels <- labels[-testIndexes]

    svm.model <- svm(trainData,train_Labels)


    poly_svm.model <- svm(trainData,train_Labels,kernel = 'polynomial',degree = 2)
    #print(svm.model)
    pred <- predict(svm.model,testData)

    conf <- table(test_Labels,pred)

    pred_poly <- predict(poly_svm.model,testData)



    conf_poly <- table(test_Labels,pred_poly)



    #print(conf)
    #print(conf_poly)

    print(confMatrixMulti(conf))
    print(confMatrixMulti(conf_poly))
  }

}

makeSVMTest(norm_iris,label_iris)
makeSVMTest(norm_leaf,label_leaf)
```

Results:

```
--iris--
5-cross fold valid.

TP FP TN FN
[1,] 11  0 18  1
[2,] 10  1 19  0
```

```
[3,]  8  0 21  1
      TP FP TN FN
[1,] 10  1 19  0
[2,] 11  0 18  1
[3,]  8  0 21  1
      TP FP TN FN
[1,]  9  0 18  3
[2,]  9  0 18  3
[3,]  9  3 18  0
      TP FP TN FN
[1,]  8  1 15  6
[2,]  9  0 14  7
[3,]  6  6 17  1
      TP FP TN FN
[1,] 11  0 17  2
[2,]  8  2 20  0
[3,]  9  0 19  2
```

--- leaf ---

```
TP  FP  TN  FN
  [1,]  0  1  2 65
  [2,]  0  2  2 64
  [3,]  1  1  1 65
  [4,]  0  2  2 64
  [5,]  0  1  2 65
  [6,]  0  1  2 65
  [7,]  1  5  1 61
  [8,]  0  2  2 64
  [9,]  0  5  2 61
 [10,]  0  1  2 65
 [11,]  0  2  2 64
 [12,]  0  6  2 60
 [13,]  0  2  2 64
 [14,]  0  3  2 63
 [15,]  0  3  2 63
 [16,]  0  3  2 63
 [17,]  0  1  2 65
 [18,]  0  2  2 64
 [19,]  0  5  2 61
 [20,]  0  2  2 64
 [21,]  0  2  2 64
 [22,]  0  2  2 64
 [23,]  0  3  2 63
 [24,]  0  4  2 62
 [25,]  0  2  2 64
 [26,]  0  3  2 63
       TP FP TN FN
  [1,]  0  1  2 65
  [2,]  0  2  2 64
  [3,]  0  2  2 64
  [4,]  0  2  2 64
  [5,]  0  1  2 65
  [6,]  0  1  2 65
  [7,]  0  6  2 60
  [8,]  0  2  2 64
  [9,]  1  4  1 62
 [10,]  0  1  2 65
 [11,]  0  2  2 64
 [12,]  0  6  2 60
 [13,]  0  2  2 64
 [14,]  0  3  2 63
```

```
[15,]  0  3  2 63
[16,]  0  3  2 63
[17,]  0  1  2 65
[18,]  0  2  2 64
[19,]  1  4  1 62
[20,]  0  2  2 64
[21,]  0  2  2 64
[22,]  0  2  2 64
[23,]  0  3  2 63
[24,]  0  4  2 62
[25,]  0  2  2 64
[26,]  0  3  2 63
      TP FP TN FN
 [1,]  0  3  2 63
 [2,]  0  2  2 64
 [3,]  0  1  2 65
 [4,]  1  4  1 62
 [5,]  0  6  2 60
 [6,]  0  6  2 60
 [7,]  0  2  2 64
 [8,]  0  2  2 64
 [9,]  1  4  1 62
[10,]  0  3  2 63
[11,]  0  2  2 64
[12,]  0  2  2 64
[13,]  0  1  2 65
[14,]  0  1  2 65
[15,]  0  3  2 63
[16,]  0  2  2 64
[17,]  0  3  2 63
[18,]  0  2  2 64
[19,]  0  1  2 65
[20,]  0  2  2 64
[21,]  0  5  2 61
[22,]  0  2  2 64
[23,]  0  1  2 65
[24,]  0  2  2 64
[25,]  0  2  2 64
[26,]  0  1  2 65
[27,]  0  1  2 65
      TP FP TN FN
 [1,]  0  3  1 64
 [2,]  0  2  1 65
 [3,]  0  1  1 66
 [4,]  0  5  1 62
 [5,]  0  6  1 61
 [6,]  0  6  1 61
 [7,]  0  2  1 65
 [8,]  0  2  1 65
 [9,]  1  4  0 63
[10,]  0  3  1 64
[11,]  0  2  1 65
[12,]  0  2  1 65
[13,]  0  1  1 66
[14,]  0  1  1 66
[15,]  0  3  1 64
[16,]  0  2  1 65
[17,]  0  3  1 64
[18,]  0  2  1 65
[19,]  0  1  1 66
[20,]  0  2  1 65
[21,]  0  5  1 62
```

```
[22,]  0  2  1 65
[23,]  0  1  1 66
[24,]  0  2  1 65
[25,]  0  2  1 65
[26,]  0  1  1 66
[27,]  0  1  1 66
      TP FP TN FN
 [1,]  0  4  0 64
 [2,]  0  3  0 65
 [3,]  0  3  0 65
 [4,]  0  2  0 66
 [5,]  0  3  0 65
 [6,]  0  1  0 67
 [7,]  0  3  0 65
 [8,]  0  2  0 66
 [9,]  0  3  0 65
[10,]  0  2  0 66
[11,]  0  3  0 65
[12,]  0  1  0 67
[13,]  0  1  0 67
[14,]  0  4  0 64
[15,]  0  1  0 67
[16,]  0  5  0 63
[17,]  0  2  0 66
[18,]  0  5  0 63
[19,]  0  2  0 66
[20,]  0  2  0 66
[21,]  0  1  0 67
[22,]  0  1  0 67
[23,]  0  1  0 67
[24,]  0  2  0 66
[25,]  0  4  0 64
[26,]  0  1  0 67
[27,]  0  4  0 64
[28,]  0  2  0 66
```

Comments:

Iris verisi ile iyi sonuçlar alabilirken leaf datası ile kötü sonuç aldım. Sorunun ne olduğunu henz çözemedim.

Part 4:

Code:

Part3 ile ayn kod

Results:

```
--- iris ---

TP  FP  TN  FN
[1,] 11  0 17  2
[2,] 10  0 18  2
[3,]  7  2 21  0
     TP FP TN FN
```

```
[1,] 11  1 18  0
[2,]  8  0 21  1
[3,] 10  0 19  1
     TP FP TN FN
[1,]  9  3 15  3
[2,]  7  1 17  5
[3,]  8  2 16  4
     TP FP TN FN
[1,]  7  0 22  1
[2,] 12  0 17  1
[3,] 10  1 19  0
     TP FP TN FN
[1,]  6  1 19  4
[2,] 12  0 13  5
[3,]  7  4 18  1
```

```
----- Leaf ---

TP FP TN FN
 [1,]  0  4  1 63
 [2,]  1  2  0 65
 [3,]  0  3  1 64
 [4,]  0  2  1 65
 [5,]  0  3  1 64
 [6,]  0  1  1 66
 [7,]  0  3  1 64
 [8,]  0  2  1 65
 [9,]  0  3  1 64
[10,]  0  2  1 65
[11,]  0  3  1 64
[12,]  0  1  1 66
[13,]  0  1  1 66
[14,]  0  4  1 63
[15,]  0  1  1 66
[16,]  0  5  1 62
[17,]  0  2  1 65
[18,]  0  5  1 62
[19,]  0  2  1 65
[20,]  0  2  1 65
[21,]  0  1  1 66
[22,]  0  1  1 66
[23,]  0  1  1 66
[24,]  0  2  1 65
[25,]  0  4  1 63
[26,]  0  1  1 66
[27,]  0  4  1 63
[28,]  0  2  1 65
      TP FP TN FN
 [1,]  0  4  2 62
 [2,]  0  1  2 65
 [3,]  0  2  2 64
 [4,]  1  2  1 64
 [5,]  0  3  2 63
 [6,]  0  1  2 65
 [7,]  0  1  2 65
 [8,]  0  3  2 63
 [9,]  0  2  2 64
[10,]  0  1  2 65
[11,]  0  4  2 62
[12,]  0  5  2 61
[13,]  0  1  2 65
[14,]  0  2  2 64
```

```
[15,]  0  3  2 63
[16,]  0  1  2 65
[17,]  0  2  2 64
[18,]  0  1  2 65
[19,]  0  1  2 65
[20,]  0  4  2 62
[21,]  0  3  2 63
[22,]  1  3  1 63
[23,]  0  2  2 64
[24,]  0  4  2 62
[25,]  0  4  2 62
[26,]  0  2  2 64
[27,]  0  4  2 62
      TP FP TN FN
 [1,]  0  4  2 62
 [2,]  0  1  2 65
 [3,]  0  2  2 64
 [4,]  0  3  2 63
 [5,]  1  2  1 64
 [6,]  0  1  2 65
 [7,]  0  1  2 65
 [8,]  0  3  2 63
 [9,]  0  2  2 64
[10,]  0  1  2 65
[11,]  0  4  2 62
[12,]  0  5  2 61
[13,]  0  1  2 65
[14,]  0  2  2 64
[15,]  0  3  2 63
[16,]  0  1  2 65
[17,]  0  2  2 64
[18,]  0  1  2 65
[19,]  0  1  2 65
[20,]  0  4  2 62
[21,]  0  3  2 63
[22,]  0  4  2 62
[23,]  0  2  2 64
[24,]  1  3  1 63
[25,]  0  4  2 62
[26,]  0  2  2 64
[27,]  0  4  2 62
      TP FP TN FN
 [1,]  0  4  2 62
 [2,]  0  1  2 65
 [3,]  0  2  2 64
 [4,]  0  3  2 63
 [5,]  0  1  2 65
 [6,]  0  2  2 64
 [7,]  0  1  2 65
 [8,]  0  1  2 65
 [9,]  0  1  2 65
[10,]  0  4  2 62
[11,]  0  3  2 63
[12,]  0  1  2 65
[13,]  1  2  1 64
[14,]  0  2  2 64
[15,]  0  1  2 65
[16,]  0  4  2 62
[17,]  0  5  2 61
[18,]  0  2  2 64
[19,]  0  1  2 65
[20,]  0  2  2 64
```

```
[21,]   0  2  2 64
[22,]   1  1  1 65
[23,]   0  1  2 65
[24,]   0  3  2 63
[25,]   0  4  2 62
[26,]   0  3  2 63
[27,]   0  1  2 65
[28,]   0  3  2 63
[29,]   0  2  2 64
[30,]   0  3  2 63
       TP FP TN FN
 [1,]   0  4  1 63
 [2,]   0  1  1 66
 [3,]   0  2  1 65
 [4,]   0  3  1 64
 [5,]   0  1  1 66
 [6,]   0  2  1 65
 [7,]   0  1  1 66
 [8,]   0  1  1 66
 [9,]   0  1  1 66
[10,]   0  4  1 63
[11,]   0  3  1 64
[12,]   0  1  1 66
[13,]   0  3  1 64
[14,]   0  2  1 65
[15,]   0  1  1 66
[16,]   0  4  1 63
[17,]   0  5  1 62
[18,]   0  2  1 65
[19,]   0  1  1 66
[20,]   0  2  1 65
[21,]   0  2  1 65
[22,]   0  2  1 65
[23,]   0  1  1 66
[24,]   0  3  1 64
[25,]   0  4  1 63
[26,]   0  3  1 64
[27,]   1  0  0 67
[28,]   0  3  1 64
[29,]   0  2  1 65
[30,]   0  3  1 64
```

Comments:

Polynomial svm ile de diğer sonuçlar gibi iris verisi ile iyi leaf verisi ile kötü sonuçlar aldım. Leaf datası ile çalışma şeklim yüzünden doğru uygulama yapamıyor olabilirim.