Practice / kNN

Prachi Sardana

Loaded the prostate cancer data and structured the data which contain 100 observations of 10 variables. The data consist of 8 numerical variables and 1 categorical variable ID

cancer\_data <- read.csv("C:/Users/sanya/Downloads/Prostate\_Cancer (1).csv")  
str(cancer\_data)

## 'data.frame': 100 obs. of 10 variables:  
## $ id : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ diagnosis\_result : chr "M" "B" "M" "M" ...  
## $ radius : int 23 9 21 14 9 25 16 15 19 25 ...  
## $ texture : int 12 13 27 16 19 25 26 18 24 11 ...  
## $ perimeter : int 151 133 130 78 135 83 120 90 88 84 ...  
## $ area : int 954 1326 1203 386 1297 477 1040 578 520 476 ...  
## $ smoothness : num 0.143 0.143 0.125 0.07 0.141 0.128 0.095 0.119 0.127 0.119 ...  
## $ compactness : num 0.278 0.079 0.16 0.284 0.133 0.17 0.109 0.165 0.193 0.24 ...  
## $ symmetry : num 0.242 0.181 0.207 0.26 0.181 0.209 0.179 0.22 0.235 0.203 ...  
## $ fractal\_dimension: num 0.079 0.057 0.06 0.097 0.059 0.076 0.057 0.075 0.074 0.082 ...

Removing the first variable id from the data.Through table() function determining the diagnosis result of patients where 38 patients having benign tumor while 62 patients have malignant tumor.Refactored the diagnosis into benign and malignant term of prostate cancer.

prostatecancer\_data <- cancer\_data[-1]  
  
### Through table function determining the diagnosis result of patients where 38 patients having benign tumor while 62 patients have malignant tumor  
table(prostatecancer\_data$diagnosis\_result)

##   
## B M   
## 38 62

### To refactor them into benign and malignant term  
  
prostatecancer\_data$diagnosis <- factor(prostatecancer\_data$diagnosis\_result, levels = c("B","M"), labels = c("Benign","Malignant"))  
  
### To interpret it in percentages  
round(prop.table(table(prostatecancer\_data$diagnosis))\*100, digits = 1)

##   
## Benign Malignant   
## 38 62

Normalizing the numeric data using min-max normalization function for (Feature scaling) except the diagnosis result which is non numeric. Used summary() function to see if various parameters like mean , median , mode etc are normalized.

normalize <- function(x){  
 return ((x - min(x))/max(x)- min(x))}  
  
### Normalizing the prostate cancer data. Since the first variable is diagnosis result which is not numeric, we'll normalize the data from the 2nd row till 9th since there are 10 rows in total  
prostatecancer\_data\_n <- as.data.frame(lapply(prostatecancer\_data[2:9],normalize))  
  
# using summary function to check if the features are normalized  
summary(prostatecancer\_data\_n$radius)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -9.000 -8.880 -8.680 -8.686 -8.520 -8.360

summary(prostatecancer\_data\_n$perimeter)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -52.00 -51.82 -51.76 -51.74 -51.64 -51.30

summary(prostatecancer\_data\_n$area)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -202.0 -201.9 -201.8 -201.7 -201.6 -201.1

After feature normalization Splitting the prostrate cancer data set into training and test data where 1 to 65 was taken as a training data and rest as testing data. Labelled, the target to predict the diagnosis result

training\_prc <- prostatecancer\_data\_n[1:65,]  
test\_prc <- prostatecancer\_data\_n[66:100,]  
  
### Labeling the target variable diagnosis result in 1st coloumn  
  
training\_prc\_labels <- prostatecancer\_data[1:65,1]  
testing\_prc\_labels <- prostatecancer\_data[66:100,1]

Training model knn using the knn formula(train,test,cl = label, k) where cl = class label and K is calculated by sqrt of **number of observation**. Here observations are 100 hence **k = 10**. Imported the library class and predicted the model prostrate cancer predict by knn formula

library(class)

## Warning: package 'class' was built under R version 4.2.3

prostate\_cancer\_pred <- knn(train = training\_prc,test = test\_prc,cl = training\_prc\_labels,k = 10)

Evaluating the model performance using crosstable function and loading gmodel package in which this function is located

Out of 35 observations, total 9 cases have been accurately predicted as Benign i.e 25.7% were true negative and 16 out of 35 (45.7%) were accurately predicted as malignant which were true positive. Thus a total 16 out of 35 predictions were true positive.There were no cases of false negative which were malignant in nature.There were 19 cases which were actually benign in nature which got predicted as malignant tumor among the patients.Accuracy of model **(TN+TP/35)** = 71.4%

library(gmodels)

## Warning: package 'gmodels' was built under R version 4.2.3

CrossTable(x = testing\_prc\_labels, y = prostate\_cancer\_pred, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 35   
##   
##   
## | prostate\_cancer\_pred   
## testing\_prc\_labels | B | M | Row Total |   
## -------------------|-----------|-----------|-----------|  
## B | 8 | 11 | 19 |   
## | 0.421 | 0.579 | 0.543 |   
## | 1.000 | 0.407 | |   
## | 0.229 | 0.314 | |   
## -------------------|-----------|-----------|-----------|  
## M | 0 | 16 | 16 |   
## | 0.000 | 1.000 | 0.457 |   
## | 0.000 | 0.593 | |   
## | 0.000 | 0.457 | |   
## -------------------|-----------|-----------|-----------|  
## Column Total | 8 | 27 | 35 |   
## | 0.229 | 0.771 | |   
## -------------------|-----------|-----------|-----------|  
##   
##

Knn classification using **caret package**

library(caret)

## Warning: package 'caret' was built under R version 4.2.3

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.2.3

## Loading required package: lattice

str(prostatecancer\_data)

## 'data.frame': 100 obs. of 10 variables:  
## $ diagnosis\_result : chr "M" "B" "M" "M" ...  
## $ radius : int 23 9 21 14 9 25 16 15 19 25 ...  
## $ texture : int 12 13 27 16 19 25 26 18 24 11 ...  
## $ perimeter : int 151 133 130 78 135 83 120 90 88 84 ...  
## $ area : int 954 1326 1203 386 1297 477 1040 578 520 476 ...  
## $ smoothness : num 0.143 0.143 0.125 0.07 0.141 0.128 0.095 0.119 0.127 0.119 ...  
## $ compactness : num 0.278 0.079 0.16 0.284 0.133 0.17 0.109 0.165 0.193 0.24 ...  
## $ symmetry : num 0.242 0.181 0.207 0.26 0.181 0.209 0.179 0.22 0.235 0.203 ...  
## $ fractal\_dimension: num 0.079 0.057 0.06 0.097 0.059 0.076 0.057 0.075 0.074 0.082 ...  
## $ diagnosis : Factor w/ 2 levels "Benign","Malignant": 2 1 2 2 2 1 2 2 2 2 ...

Seeded the data to 124 to make the data replicable used createDataPartition() from caret package to split data into training and test data set. y parameter specifies the target variable , p is the proportion of data to assign it to training\_set and list = false to make sure that the result is a vector of indices. Checked the dimensions using dim() function for training and test data set.

## Set.seed method used so that data can be replicable  
## createDataPartition function split the data into training and test set.  
set.seed(124)  
intrain <- createDataPartition(y = prostatecancer\_data$diagnosis\_result, p = 0.7, list = FALSE)  
  
# training set  
train\_prc <- prostatecancer\_data[intrain,]  
#testing set  
testset\_prc <- prostatecancer\_data[-intrain,]  
  
# Checking the dimensions of training and test data set  
dim(train\_prc)

## [1] 71 10

dim(testset\_prc)

## [1] 29 10

Converted “diagnosis\_result” column in the training dataset (train\_prc) into a factor variable which helps in classification tasks. Used TrainControl() function to control the train process by using the method “repeatedcv”(repeated cross validation) with 10-fold cross-validation, repeated three times which helps in assessing the model performance.Used k-Nearest Neighbors (k-NN) classification model for training the function. It uses the “diagnosis\_result” as the target variable and includes all the other columns in the dataset. The “knn” method is used for k-NN classification. The trControl parameter specifies the control settings for training (cross-validation), and tuneLength indicates how many different values of “k” to consider.knn\_fit prints the summary of the k-NN model, which includes information about the training process, the best model parameters, and performance statistics.based on knn fit k = 5 was used for finding the accuracy of model.

# summary(prostatecancer\_data)  
  
train\_prc$diagnosis\_result = factor(train\_prc[["diagnosis\_result"]])  
  
# Train control using repeatedcv method   
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)  
  
# setting the seed to 124  
set.seed(124)  
  
# Knn fit   
  
# Used k-Nearest Neighbors (k-NN) classification model for training the function.  
knn\_fit <- train(diagnosis\_result ~., data = train\_prc, method = "knn",  
 trControl=trctrl,  
 preProcess = c("center", "scale"),  
 tuneLength = 10)  
  
# knn\_fit prints the summary of the k-NN model, which includes information about the training process, the best model parameters, and performance statistics.  
  
knn\_fit

## k-Nearest Neighbors   
##   
## 71 samples  
## 9 predictor  
## 2 classes: 'B', 'M'   
##   
## Pre-processing: centered (9), scaled (9)   
## Resampling: Cross-Validated (10 fold, repeated 3 times)   
## Summary of sample sizes: 64, 63, 64, 64, 65, 64, ...   
## Resampling results across tuning parameters:  
##   
## k Accuracy Kappa   
## 5 0.9773810 0.9469370  
## 7 0.9732143 0.9374132  
## 9 0.9690476 0.9290799  
## 11 0.9593254 0.9064608  
## 13 0.9498016 0.8852487  
## 15 0.9498016 0.8861710  
## 17 0.9402778 0.8649589  
## 19 0.9319444 0.8460700  
## 21 0.9271825 0.8359251  
## 23 0.9265873 0.8343023  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was k = 5.

Predicting the test data by using knn-fit. It computes the predicted diagnosis results based on the model’s input features and store these predictions in the test\_pred variable. The test set levels are factored such that it’s the same factor levels for training set as well which is important to make comparisons and evaluate model performance. It also computes accuracy (e.g., true positives, true negatives, false positives, and false negatives).

# Predicting the test data  
test\_pred <- predict(knn\_fit, newdata = testset\_prc)  
print(test\_pred)

## [1] M M M M M M M M M M M M M M M B B B B B B M B B M B B M M  
## Levels: B M

# Factoring the test data set levels diagnosis\_result  
testset\_prc$diagnosis\_result <- factor(testset\_prc$diagnosis\_result, levels = levels(train\_prc$diagnosis\_result))  
  
# creating a confusion matrix to evaluate the performance of KNN model  
confusionMatrix(test\_pred, reference = testset\_prc$diagnosis\_result)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction B M  
## B 10 0  
## M 1 18  
##   
## Accuracy : 0.9655   
## 95% CI : (0.8224, 0.9991)  
## No Information Rate : 0.6207   
## P-Value [Acc > NIR] : 1.844e-05   
##   
## Kappa : 0.9254   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9091   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9474   
## Prevalence : 0.3793   
## Detection Rate : 0.3448   
## Detection Prevalence : 0.3448   
## Balanced Accuracy : 0.9545   
##   
## 'Positive' Class : B   
##

Based on the confusion matrix the accuracy came out to be 96.5 %. The matrix gives the reference and the prediction labels. The true positives were found to be 10 i.e the instances where the class was truly benign. The true negatives were 18 which were classified as malignance. The false positive was 1 which was incorrectly classified as Benign. False negative were 0 that were incorrectly classified as malignant. Based on 95% confidence interval the value of accuracy ranges from 0.822 to 0.991.kappa value is 0.92 indicating substancial agreement with the prediction and true labels.

*Comparing the model accuracy*

The kNN algorithm calculated in respect to caret package were 96.5 % which is higher while accuracy through class package came out to be 71.4 %. Since it uses different methods for finding the accuracy like class package using cross validation method and caret method is based on classification regression. Both used different K values for checking the accuracies of the model.