Practice / kNN

Nithya Sarabudla

02-03-2024

Step 1 - Data collection

The dataset for this prostate cancer detection case study comprises 100 patient observations, designed for practicing the kNN algorithm in R. It includes 10 variables: 8 numeric, representing clinical measurements from Digital Rectal Exams (DRE), one categorical variable for classification purposes, and one ID variable for patient identification. This dataset simulates typical clinical data for modeling and prediction of oncogenic growth.

```
data <- read.csv("/Users/nithyasarabudla/Downloads/Prostate_Cancer.csv", stringsAsFactors = FALSE)
```

step 2 - preparing and exploring the data

Removes the first column from the dataset. This is likely the ID column, which is not useful for model training as it does not contain any predictive information.

```
data <- data[-1]
```

Creates a frequency table of the diagnosis_result variable. This step is important for understanding the distribution of diagnoses in the dataset, such as how many patients have been diagnosed as benign versus malignant.

```
table(data$diagnosis_result)
```

```
##
##  B  M
## 38 62
```

Converts the diagnosis_result variable into a factor with more readable labels (“Benign” and “Malignant”) and then calculates the percentage of each diagnosis.

```
data$diagnosis <- factor(data$diagnosis_result, levels = c("B", "M"), labels = c("Benign", "Malignant"))
round(prop.table(table(data$diagnosis)) * 100, digits = 1)
```

```
##
##    Benign Malignant
##        38        62
```

```r
# Normalizing numeric data to scale features between 0 and 1 for improving the kNN algorithm's performa
normalize <- function(x) {
return ((x - min(x)) / (max(x) - min(x))) }
# Applying the normalization function to numeric columns (assuming columns 2 to 9 are numeric)
data_n <- as.data.frame(lapply(data[2:9], normalize))
# Displaying a summary of the normalized data to check the scaling and understand the distribution
summary(data_n)
```

```
##      radius          texture         perimeter           area
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.1875   1st Qu.:0.1875   1st Qu.:0.2542   1st Qu.:0.1639
##  Median :0.5000   Median :0.4062   Median :0.3500   Median :0.2637
##  Mean   :0.4906   Mean   :0.4519   Mean   :0.3732   Mean   :0.2989
##  3rd Qu.:0.7500   3rd Qu.:0.7031   3rd Qu.:0.5188   3rd Qu.:0.4266
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##    smoothness      compactness        symmetry      fractal_dimension
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.3219   1st Qu.:0.1384   1st Qu.:0.2189   1st Qu.:0.1364
##  Median :0.4384   Median :0.2622   Median :0.3254   Median :0.2273
##  Mean   :0.4484   Mean   :0.2889   Mean   :0.3442   Mean   :0.2657
##  3rd Qu.:0.5753   3rd Qu.:0.3876   3rd Qu.:0.4379   3rd Qu.:0.3636
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

```r
# Splitting normalized data into training and test sets
# First 65 observations are assigned to the training set
data_train <- data_n[1:65,]
# Remaining observations (66 to 100) are assigned to the test set
data_test <- data_n[66:100,]

# Extracting diagnosis labels for the training set from the original dataset
data_train_labels <- data[1:65, 1]
# Extracting diagnosis labels for the test set from the original dataset
data_test_labels <- data[66:100, 1]
```

step 3 - Training a model on data

```r
# Installing and loading the 'class' package for kNN
library(class) # Load 'class' library to use the kNN function
```

```r
# The model predicts the test set labels with k = 10 nearest neighbors
data_test_pred <- knn(train = data_train, test = data_test, cl = data_train_labels, k=10)
# 'data_train' is the training data set, 'data_test' is the test data set to predict,
# 'cl' argument is the vector of class labels for the training set,
# and 'k' is the number of nearest neighbors to use for prediction.
```

step-4 - Evalute the model perfomance

```r
library(gmodels) # Load 'gmodels' library to access the CrossTable function
```

```
# This function compares the actual test labels with the predicted labels
# 'x' argument represents the actual labels, and 'y' argument represents the predicted labels from the
# 'prop.chisq=FALSE' disables the computation of the chi-squared proportion test statistic for the tabl
CrossTable(x= data_test_labels, y = data_test_pred, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  35
##
##
##                  | data_test_pred
## data_test_labels |         B |         M | Row Total |
## -----------------|-----------|-----------|-----------|
##                B |         6 |        13 |        19 |
##                  |     0.316 |     0.684 |     0.543 |
##                  |     1.000 |     0.448 |           |
##                  |     0.171 |     0.371 |           |
## -----------------|-----------|-----------|-----------|
##                M |         0 |        16 |        16 |
##                  |     0.000 |     1.000 |     0.457 |
##                  |     0.000 |     0.552 |           |
##                  |     0.000 |     0.457 |           |
## -----------------|-----------|-----------|-----------|
##     Column Total |         6 |        29 |        35 |
##                  |     0.171 |     0.829 |           |
## -----------------|-----------|-----------|-----------|
##
##
```

```
# This cross-tabulation provides a detailed breakdown of the model's predictions,
# including the number of true positives, false positives, true negatives, and false negatives.
# It helps in assessing the accuracy, sensitivity, specificity, and other performance metrics of the mo
```

The test data consisted of 35 observations. Out of which 5 cases have been accurately predicted (TN->True Negatives) as Benign (B) in nature which constitutes 14.3%. Also, 16 out of 35 observations were accurately predicted (TP-> True Positives) as Malignant (M) in nature which constitutes 45.7%. Thus a total of 16 out of 35 predictions where TP i.e, True Positive in nature.

There were no cases of False Negatives (FN) meaning no cases were recorded which actually are malignant in nature but got predicted as benign.

There were 14 cases of False Positives (FP) meaning 14 cases were actually benign in nature but got predicted as malignant.

The total accuracy of the model is 60 %((TN+TP)/35) which shows that there may be chances to improve the model performance

step- 5 - Improve the performance of the model

```
# Retraining the kNN model with k = 11 to explore the impact on model performance
data_test_pred_11 <- knn(train = data_train, test = data_test, cl = data_train_labels, k=11)
```

```
# Evaluating the model's performance with the adjusted k value
CrossTable(x= data_test_labels, y = data_test_pred_11, prop.chisq=FALSE)
```

Repeating the steps 3 and 4 by changing the k=11

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  35
##
##
##                  | data_test_pred_11
## data_test_labels |         B |         M | Row Total |
## -----------------|-----------|-----------|-----------|
##                B |         5 |        14 |        19 |
##                  |     0.263 |     0.737 |     0.543 |
##                  |     1.000 |     0.467 |           |
##                  |     0.143 |     0.400 |           |
## -----------------|-----------|-----------|-----------|
##                M |         0 |        16 |        16 |
##                  |     0.000 |     1.000 |     0.457 |
##                  |     0.000 |     0.533 |           |
##                  |     0.000 |     0.457 |           |
## -----------------|-----------|-----------|-----------|
##     Column Total |         5 |        30 |        35 |
##                  |     0.143 |     0.857 |           |
## -----------------|-----------|-----------|-----------|
##
##
```

For the dataset comprising 35 observations tested with the kNN model where k=11, we observed

True Negatives (TN): 5 instances were correctly identified as benign, which represents 14.3% of the total cases tested. True Positives (TP): The model successfully identified 16 malignant cases, which accounts for 45.7% of the overall tests conducted. False Negatives (FN): Importantly, there were no instances where the model failed to identify malignant cases as such. The absence of false negatives is critical, as it ensures that no malignant conditions are mistakenly overlooked, which could otherwise pose a significant risk to

patient health. False Positives (FP): However, there were 14 occurrences where benign cases were incorrectly classified as malignant. These false alarms, while not as critical as false negatives, suggest an area for improvement in reducing unnecessary anxiety or interventions for patients.

Given these results, the model achieves an overall accuracy rate of 60%. This calculation is based on the sum of true positives and true negatives divided by the total number of cases (i.e.(TN+TP)/35).

Repeating the steps 3 and 4 by changing the k=13

```
# Retraining the kNN model with k = 13 to explore the impact on model performance
data_test_pred_13 <- knn(train = data_train, test = data_test, cl = data_train_labels, k=13)
```

```
# Evaluating the model's performance with the adjusted k value
CrossTable(x= data_test_labels, y = data_test_pred_13, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  35
##
##
##                  | data_test_pred_13
## data_test_labels |         B |         M | Row Total |
## -----------------|-----------|-----------|-----------|
##                B |         5 |        14 |        19 |
##                  |     0.263 |     0.737 |     0.543 |
##                  |     1.000 |     0.467 |           |
##                  |     0.143 |     0.400 |           |
## -----------------|-----------|-----------|-----------|
##                M |         0 |        16 |        16 |
##                  |     0.000 |     1.000 |     0.457 |
##                  |     0.000 |     0.533 |           |
##                  |     0.000 |     0.457 |           |
## -----------------|-----------|-----------|-----------|
##     Column Total |         5 |        30 |        35 |
##                  |     0.143 |     0.857 |           |
## -----------------|-----------|-----------|-----------|
##
##
```

The model successfully identified 5 cases as benign (True Negatives), making up 14.3% of the dataset, and accurately detected 16 cases as malignant (True Positives), which constitutes 45.7%. There were 14 instances where benign cases were incorrectly labeled as malignant (False Positives), but no malignant cases were misclassified as benign (False Negatives). This means that the model showed a precision in identifying malignant cases without any missed detections. However, the challenge lies in reducing the number of benign cases mistakenly identified as malignant. The overall accuracy of the model stands at 60%

When evaluating the model's performance with different k values of 10, 11, and 13, each scenario yielded the same accuracy rate of 60%.

Question 3

```r
# Load the caret Package
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```r
# Train the kNN Model Using caret
set.seed(123) # For reproducibility
train_control <- trainControl(method="cv", number=10) # 10-fold cross-validation
kNN_model_caret <- train(x=data_train, y=data_train_labels, method="knn", trControl=train_control, tunel
```

```r
# Make Predictions and Evaluate Accuracy
predictions_caret <- predict(kNN_model_caret, newdata=data_test)
```

```r
# Ensure both are factors
predictions_caret <- as.factor(predictions_caret)
data_test_labels <- as.factor(data_test_labels)

# Ensure both have the same levels, assuming the levels are 'level1', 'level2', etc.
common_levels <- union(levels(predictions_caret), levels(data_test_labels))
predictions_caret <- factor(predictions_caret, levels = common_levels)
data_test_labels <- factor(data_test_labels, levels = common_levels)

# Now you can call confusionMatrix
confusionMatrix(predictions_caret, data_test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B  9  1
##          M 10 15
##
##                Accuracy : 0.6857
##                  95% CI : (0.5071, 0.8315)
##     No Information Rate : 0.5429
##     P-Value [Acc > NIR] : 0.06195
##
##                   Kappa : 0.3937
##
##  Mcnemar's Test P-Value : 0.01586
##
##             Sensitivity : 0.4737
##             Specificity : 0.9375
```

```
##          Pos Pred Value : 0.9000
##          Neg Pred Value : 0.6000
##              Prevalence : 0.5429
##          Detection Rate : 0.2571
##    Detection Prevalence : 0.2857
##       Balanced Accuracy : 0.7056
##
##        'Positive' Class : B
##
```

The kNN model implemented using the caret package demonstrated a notable improvement in accuracy, reaching approximately 68.57%, compared to the consistent 60% accuracy observed with earlier implementations across different k values. This enhancement underscores the benefit of utilizing caret for its sophisticated model tuning and evaluation capabilities.

The model's sensitivity, or its ability to correctly identify benign cases, was reported at 47.37%, indicating a moderate effectiveness in detecting the 'Positive' class within this context. However, the specificity was impressively high at 93.75%, highlighting the model's proficiency in accurately identifying malignant cases, which is crucial for minimizing false positive diagnoses in a clinical setting.

Moreover, the positive predictive value of 90% for benign predictions illustrates the model's reliability in its positive classifications, although the negative predictive value of 60% suggests there is room for improvement in accurately identifying true negatives, or malignant cases. The balanced accuracy of 70.56% offers a more comprehensive view of the model's performance, considering both sensitivity and specificity, thus providing a balanced measure of its diagnostic ability.