Lab 6-KNN Classification

Jay Singhvi

03/08/2024

A KNN analysis on Breast Cancer Data. (4 Questions, 28 points)

In this lab, you will fit a KNN model to a data set and find the optimal number for K using cross validation. Then you will evaluate the KNN model's classification performance on the test set.

You will work with the Wisconsin Diagnostic Breast Cancer (WDBC) data set. This dataset contains various measurements of cell nuclei gathered from biopsies of breast tissue. Each row (case) of measurements is associated with a classification of "B" for benign, and "M" for malignant (cancer).

QUESTION 1 (3 parts, 6pts) MAP TO GRADESCOPE #### Acquire and Process the Data

1a(2pts):
Read in the data from the file "wdbc_data.csv" and assign it to the variable "df.wdbc". Remove the id column. The predicted variable is "diagnosis". Make that column a factor.

```
df.wdbc = read.csv("wdbc_data.csv")
df.wdbc = subset(df.wdbc, select = -c(id))
df.wdbc$diagnosis = factor(df.wdbc$diagnosis)
```

Check for Imbalance Because we are performing classification, we need to check for possible imbalance in the predicted variable, diagnosis. The minority class has the smaller number of occurrences in the data. The percent of the minority class is calculated: (minority class count / data size). The degree of imbalance can be described with these terms: Degree of Imbalance/Percent of Minority Class: None >40% Mild 20-40% Moderate 1-20% Extreme <1%

1b(2pts):
Print a table that shows the frequencies of the levels of the predicted variable, "diagnosis" in the data set. Use the kable function to render the table. Set the column names to "diagnosis" and "count". Add a caption: 'Diagnosis Levels in Dataset'.

```
diag.tbl<-table(df.wdbc$diagnosis)
kable(diag.tbl, col.names = c("Diagnosis", "Count"), caption = "Diagnosis Levels in Dataset")
```

Table 1: Diagnosis Levels in Dataset

Diagnosis	Count
B	357
M	212

1c(2pts):
What is the minority class? What percent (to the nearest whole number) of the data is the minority class? What is the Degree of Imbalance term that is associated with this percent?

minority class:M , minority class% = 37.25% , Degree of Imbalance term: Mild

QUESTION 2 (3 parts, 6pts) MAP TO GRADESCOPE

Train and Test Sets. 2a(2pts): In the chunk below, use the createDataPartition function in the caret library to create train and test sets. Use 75% of the data for training.

```
RNGversion("4.3.2")
set.seed(123456)
train_index <- createDataPartition(df.wdbc$diagnosis, p = 0.75, list = FALSE)
train_data <- df.wdbc[train_index, ]
test_data <- df.wdbc[-train_index, ]
```

Fit Model to Training Data with Cross Validation. 2b(2pts): In the chunk below, write code that trains a KNN model on the train set. Use functions from the caret package. Use repeated cross validation with a repeat of 3, and set the "classProbs" parameter to TRUE. The model formula: The diagnosis column is the predicted variable. Use all of the remaining columns as predictors in the model.

The training and fitting parameters are specified by calling "trainControl" with some parameters. In this case, we are specifying repeated cross validation with 3 repeats. This means 10-fold cross validation will be performed three times.

Use the "preProcess" parameter to center and scale the predictors in preparation for using Euclidean distance. The metric to use is "Accuracy", and the "tuneLength" parameter should be set to 20. Print the model, the object returned from the train function, by typing it in the console.

```
RNGversion("4.3.2")
set.seed(123456)

ctrl <- trainControl(method="repeatedcv", number=10, repeats = 3, classProbs = TRUE)
knn.mod <- train(diagnosis ~ ., data = train_data, method = "knn", trControl = ctrl, preProcess = c("ce

knn.mod
```

```
## k-Nearest Neighbors
##
## 427 samples
##  30 predictor
##   2 classes: 'B', 'M'
##
## Pre-processing: centered (30), scaled (30)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 384, 384, 384, 384, 385, 384, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.9625877  0.9188346
##    7  0.9641381  0.9221784
##    9  0.9625692  0.9185388
##   11  0.9618125  0.9165266
##   13  0.9594869  0.9111693
##   15  0.9594869  0.9109994
##   17  0.9587117  0.9090420
##   19  0.9563861  0.9038200
##   21  0.9555740  0.9020925
##   23  0.9540052  0.8986075
##   25  0.9493355  0.8878675
##   27  0.9485419  0.8859703
##   29  0.9485419  0.8860249
##   31  0.9477667  0.8843874
##   33  0.9461979  0.8808760
##   35  0.9469915  0.8821674
##   37  0.9470100  0.8819499
##   39  0.9462163  0.8802872
##   41  0.9469915  0.8822055
##   43  0.9462163  0.8803905
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 7.
```

2c(2pts): Inspect the output that was printed in the console. What is the best value for K? What was the model's best accuracy?

best K: 7 , best accuracy: 0.9641381

QUESTION 3 (1 part, 2pts) MAP TO GRADESCOPE

Evaluating the model. Now evaluate the model's predictive ability with a confusion matrix.

Confusion Matrix. 3a(2pts): Write code in the chunk below that uses the model to make classifications on the data in the test set. Call the predict function to get the model's predictions on the test set. Call the confusionMatrix function to print a confusion matrix to see how the model's predictions compare the the actual values in the test set. Add a parameter to the call to the confusionMatrix to set the positive outcome level to malignant (positive='M').

```r
knn.predict <- predict(knn.mod, newdata = test_data)
confusionMatrix(knn.predict, test_data$diagnosis, positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 88  3
##          M  1 50
##
##                Accuracy : 0.9718
##                  95% CI : (0.9294, 0.9923)
##     No Information Rate : 0.6268
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9393
##
##  Mcnemar's Test P-Value : 0.6171
##
##             Sensitivity : 0.9434
##             Specificity : 0.9888
##          Pos Pred Value : 0.9804
##          Neg Pred Value : 0.9670
##              Prevalence : 0.3732
##          Detection Rate : 0.3521
##    Detection Prevalence : 0.3592
##       Balanced Accuracy : 0.9661
##
##        'Positive' Class : M
##
```

QUESTION 4 (7 parts, 14pts) MAP TO GRADESCOPE

Summary Questions Please answer the following questions based on the results of the model's predictions on the test set.

4a(2pts):
What is the model's misclassification rate? How would you describe the model's performance on the test set? How does it compare with a classifier that predicts the outcome levels randomly?

misclassification rate: 2.82 %,

4b(2pts): How many false positives and false negatives were recorded? What was the misclassification rate (express as a percent to the nearest tenths place)?

false positives:3 , false negatives:1

4c(2pts): What does it mean in terms of diagnosis for a patient if the model's prediction was a false positive?

that the patient had no cancer but was treated for cancer

4d(2pts): What is the false positive rate?

false positive rate = 2.1%

4e(2pts): What does it mean for a patient if the model's prediction was a false negative?

that the person had cancer but the prediction was not that he didn't have cancer

4f(2pts): What is the false negative rate?

false negative rate = 0.7%

4g(2pts): Which do you think is more important, a false positive or a false negative? Why?

a false negative as the person did have a cancer but was predicted that he did not have a cancer which is more harmful compared to false positive hence false negative is more important to consider.