Q38

Rahul Atre

2023-11-22

Q38 [Classification]

Do exercises 3 and 4, from DUDADS, chapter 21, using the wine.csv dataset.

Exercise 3: Construct and evaluate naïve Bayes classifiers for the Wine and for the 2011 Gapminder dataset.

Let us first evaluate a naive bayes classifier for the wine dataset:

```
set.seed(144) #Setting seed for reproducibility

wine_df = read.csv("wine.csv")
wine_df$Class = as.factor(wine_df$Class) #Convert to a factor

#Let us assume that the response variable we would like to predict is Class

n = nrow(wine_df) #number of rows

#Let us do a 70/30 split on the training/test data

training_set = wine_df[sample(n, 0.7 * n), ]
test_set = wine_df[setdiff(1:n, rownames(training_set)), ]

wine_bayes_model = naive_bayes(Class ~ ., data = training_set)
wine_prediction = predict(wine_bayes_model, test_set)
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```
table(wine_prediction, test_set$Class) #Confusion matrix
```

```
##
## wine_prediction  1  2  3
##               1 18  0  0
##               2  1 23  0
##               3  0  1 11
```

```
wine_accuracy = sum(wine_prediction == test_set$Class) / nrow(test_set)
wine_accuracy
```

```
## [1] 0.962963
```

Therefore, for the Naive Bayes Classifier of the wine dataset, we obtain an overall accuracy rate of 96.29%.

(Optional): Let us now apply a similar procedure on the Gapminder dataset:

```
set.seed(144) #Setting seed for reproducibility

gapminder_df = read.csv("gapminder.csv")
gapminder_df = na.omit(gapminder_df)
gapminder_df$region = as.factor(gapminder_df$region) #Convert to a factor

#Let us assume that the response variable we would like to predict is region using the other variables

n = nrow(gapminder_df) #number of rows

#Let us do a 70/30 split on the training/test data

training_set_gap = gapminder_df[sample(n, 0.7 * n), ]
test_set_gap = gapminder_df[setdiff(1:n, rownames(training_set_gap)), ]

gapminder_bayes_model = naive_bayes(region ~ ., data = training_set_gap)
```

```
## Warning: naive_bayes(): Feature country - zero probabilities are present.
## Consider Laplace smoothing.
```

```
## Warning: naive_bayes(): Feature continent - zero probabilities are present.
## Consider Laplace smoothing.
```

```
gapminder_prediction = predict(gapminder_bayes_model, test_set_gap)
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```
gapminder_accuracy = sum(gapminder_prediction == test_set_gap$region) / nrow(test_set_gap)
gapminder_accuracy
```

```
## [1] 0.3560666
```

For the gapminder classifier, we wanted to predict the region of a country based on all other parameters (life expectancy, infant mortality, etc). From our results, we can see that the overall accuracy is 35.61%, which is much lower than the Wine dataset. Thus, we can conclude that the naive bayes classifier works better on the Wine dataset than Gapminder.

Exercise 4: Construct and evaluate CART models for the Wine and for the Wisconsin Breast Cancer datasets.

Let us first perform the CART model for the Wine dataset:

```
set.seed(144) #Setting seed for reproducibility

wine_df = read.csv("wine.csv")
wine_df$Class = as.factor(wine_df$Class) #Convert to a factor
```

```r
#Let us assume that the response variable we would like to predict is Class

n = nrow(wine_df) #number of rows

#Let us do a 70/30 split on the training/test data

training_set = wine_df[sample(n, 0.7 * n), ]
test_set = wine_df[setdiff(1:n, rownames(training_set)), ]

wine_CART_model = rpart(Class ~ ., data = training_set)
wine_prediction = predict(wine_CART_model, test_set, type = "class")

table(wine_prediction, test_set$Class) #Confusion matrix
```

```
##
## wine_prediction  1  2  3
##               1 14  0  1
##               2  5 24  0
##               3  0  0 10
```

```r
wine_accuracy = sum(wine_prediction == test_set$Class) / nrow(test_set)
wine_accuracy
```

```
## [1] 0.8888889
```

Above, we can see that for the CART model of the Wine dataset, the accuracy rate is 88.88%. This is slightly lower than the accuracy of the Naive Bayes model used in Exercise 8.

(Optional): For the Wisconsin Breast Cancer Dataset, I could not find it on BrightSpace or Slack, so I downloaded it from Kaggle instead. It appears to be the same data. We will now peform the CART method on this dataset.

```r
set.seed(144) #Setting seed for reproducibility

cancer_df = read.csv("wisc_bc_data.csv")
cancer_df = na.omit(cancer_df)
cancer_df$diagnosis = as.factor(cancer_df$diagnosis) #Convert to a factor

#Let us assume that the response variable we would like to predict is the final diagnosis

n = nrow(cancer_df) #number of rows

#Let us do a 70/30 split on the training/test data

training_set_cancer = cancer_df[sample(n, 0.7 * n), ]
test_set_cancer = cancer_df[setdiff(1:n, rownames(training_set_cancer)), ]

cancer_CART_model = rpart(diagnosis ~ ., training_set_cancer)
cancer_prediction = predict(cancer_CART_model, test_set_cancer, type = "class")

table(cancer_prediction, test_set_cancer$diagnosis) #Confusion matrix
```

```
##
## cancer_prediction  B  M
##                 B 93  6
##                 M  8 64
```

```
cancer_accuracy = sum(cancer_prediction == test_set_cancer$diagnosis)/nrow(test_set_cancer)
cancer_accuracy
```

```
## [1] 0.9181287
```

Above, we can see that for the CART model of the breast cancer dataset, the diagnosis prediction has an accuracy of 91.81%, which is much better than the Wine dataset's accuracy score.