Apply Naive Bayes from klaR Package

Dr. Nishat Mohammad

02/15/2024

1. Load Requirements.

1.1. Loading `klaR` Package.

```
library(klaR)
```

1.2. Loading `iris` dataset.

```
data(iris)
```

2. Explore `iris` Dataset.

```
# Get number of rows in iris dataset
nrow(iris)
```

```
## [1] 150
```

```
# Look at the summary of statistics of the dataset
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

```r
# Look at the first 6 rows of iris data set
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

The number of rows in iris are "150". This was gotten by passing iris to `nrow()` function.
The summary of the descriptive statistics for the variables in the iris data by mean, median, min, max are gotten by passing iris to `summary()` function and can be seen above.
The first 6 rows of the iris dataset can be seen by passing iris to `head()` function and can be viewed above. the default number of rows is 6 for this function and can be adjusted by passing the required number of rows to the n option like this `head(iris, n=10)` if 10 rows are needed to be viewed.
The iris dataset describes the data collected for 3 species (Setosa, Versicolor and Virginica) based on the Sepal and Petal lengths and widths.

3. Split Data into Train and Test Sets.

3.1. Define the indices for test set rows.

```r
# The tests set
testidx <- which(1:length(iris[, 1]) %% 5 == 0)
head(testidx)
```

```
## [1]  5 10 15 20 25 30
```

This line of code extracts the index for every fifth (5th) row of iris data set by giving the length of the first column by 5 without a remainder (remainder is 0).

3.2. Define train and test.

```r
# separate into training and testing datasets
iristrain <- iris[-testidx,]
iristest <- iris[testidx,]
```

Take all the data in `testidx` as test form the iris data set and the rest of the indices will be for the train set. This defines the test and train sets as: 30 rows for test and *150-30*= 120 rows for the train which is a ratio of 80:20 for the train and test sets respectively.

4. Naive Bayes Model.

```r
# apply Naive Bayes
nbmodel <- NaiveBayes(Species~., data=iristrain)
```

The name of the Naive bayes model is `nbmodel`.
Modelling is done using the `NaiveBayes()` function form the `e1071` package.
The model predicts the species form the `Species` variable of iris data by adding tilda `~` after the `Species` variable name. The dot `.` indicates that all other variables are to be used as for the prediction of the species.
The `iristrain` is the train set created earlier and will be used to train the model by passing it to the `data=` option.

5. Model Accuracy.

```r
# check the accuracy
prediction <- predict(nbmodel, iristest[,-5])
table(prediction$class, iristest[,5])
```

```
##
##              setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         10         2
##   virginica       0          0         8
```

First the prediction is done using the model `nbmodel` and all the data in the test set `iristest` except the fifth column with species names by using the `predict()` function.
Then a table is used to check the predictions made by looking at the classes gotten form the Naive Bayes through the `class` form the `predict()` function object. also the fifth column of the test set is passed to get the **true cases** for those species in the firth column of the test set.
From the table generated, the following inferences can be made:

True Positives: setosa 10, versicolor 10 and virginica 8 making a total of 28.
True Negatives: The true cases alone for this data set as explained previously.
False positives: setosa 0, versicolor 2 (the model predicted these as virginica), virginica 0. The probability that this model predicts true cases correctly is *28/30 = 0.933*.
The accuracy is *(True Positives + True Negatives) / total_obs = (28+x)/30*, where x is the sum of True negatives.
If the true negatives were to be gotten from: total_obs - TP it would imply that the true negatives and false positives are the same which sounds erronious so I will stick with accuracy of the model to be *(28+x)/30*, where x is the sum of True negatives. but in the vice versa the accuracy will be equal to 30/30 which is almost impossible in my opinion.