LAU7-By-Shantal-Cruz

```r
#Getting started with Naive Bayes
#Install the package
#install.packages("e1071")
#Loading the library
library(e1071)
#The documentation also contains an example implementation of Titanic dataset
#Next load the Titanic dataset
data("Titanic")
#Save into a data frame and view it
Titanic_df=as.data.frame(Titanic)
#Creating data from table
repeating_sequence=rep.int(seq_len(nrow(Titanic_df)), Titanic_df$Freq)
#This will repeat each combination equal to the frequency of each combination

#Create the dataset by row repetition created
Titanic_dataset=Titanic_df[repeating_sequence,]
#We no longer need the frequency, drop the feature
Titanic_dataset$Freq=NULL

#Fitting the Naive Bayes model
Naive_Bayes_Model=naiveBayes(Survived ~., data=Titanic_dataset)
#What does the model say? Print the model summary
Naive_Bayes_Model
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##       No      Yes
## 0.676965 0.323035
##
## Conditional probabilities:
##      Class
## Y           1st        2nd        3rd       Crew
##   No  0.08187919 0.11208054 0.35436242 0.45167785
##   Yes 0.28551336 0.16596343 0.25035162 0.29817159
##
##      Sex
## Y          Male     Female
##   No  0.91543624 0.08456376
##   Yes 0.51617440 0.48382560
##
##      Age
```

```
## Y          Child      Adult
##   No  0.03489933 0.96510067
##   Yes 0.08016878 0.91983122
```

```r
#Prediction on the dataset
NB_Predictions=predict(Naive_Bayes_Model,Titanic_dataset)
#Confusion matrix to check accuracy
table(NB_Predictions,Titanic_dataset$Survived)
```

```
##
## NB_Predictions   No  Yes
##            No  1364  362
##            Yes  126  349
```

```r
#Getting started with Naive Bayes in mlr
#Install the package
#install.packages("mlr")
#Loading the library
library(mlr)
```

```
## Loading required package: ParamHelpers
```

```
## Warning message: 'mlr' is in 'maintenance-only' mode since July 2019.
## Future development will only happen in 'mlr3'
## (<https://mlr3.mlr-org.com>). Due to the focus on 'mlr3' there might be
## uncaught bugs meanwhile in {mlr} - please consider switching.
```

```
##
## Attaching package: 'mlr'
```

```
## The following object is masked from 'package:e1071':
##
##     impute
```

```r
#Create a classification task for learning on Titanic Dataset and specify the target feature
task = makeClassifTask(data = Titanic_dataset, target = "Survived")

#Initialize the Naive Bayes classifier
selected_model = makeLearner("classif.naiveBayes")

#Train the model
NB_mlr = train(selected_model, task)

#Read the model learned
NB_mlr$learner.model
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##       No      Yes
## 0.676965 0.323035
##
## Conditional probabilities:
```

```
##      Class
## Y           1st        2nd        3rd       Crew
##   No  0.08187919 0.11208054 0.35436242 0.45167785
##   Yes 0.28551336 0.16596343 0.25035162 0.29817159
##
##      Sex
## Y          Male     Female
##   No  0.91543624 0.08456376
##   Yes 0.51617440 0.48382560
##
##      Age
## Y         Child      Adult
##   No  0.03489933 0.96510067
##   Yes 0.08016878 0.91983122
```

```r
#Predict on the dataset without passing the target feature
predictions_mlr = as.data.frame(predict(NB_mlr, newdata = Titanic_dataset[,1:3]))

##Confusion matrix to check accuracy
table(predictions_mlr[,1],Titanic_dataset$Survived)
```

```
##
##         No  Yes
##   No  1364  362
##   Yes  126  349
```

```r
# K-Nearest-Neighbors

# import libraries
library(FNN)
library(MASS)
data(Boston)

# setup samples
set.seed(1)
boston_idx = sample(1:nrow(Boston), size = 250)
trn_boston = Boston[boston_idx, ]
tst_boston  = Boston[-boston_idx, ]

X_trn_boston = trn_boston["lstat"]
X_tst_boston = tst_boston["lstat"]
y_trn_boston = trn_boston["medv"]
y_tst_boston = tst_boston["medv"]

X_trn_boston_min = min(X_trn_boston)
X_trn_boston_max = max(X_trn_boston)
lstat_grid = data.frame(lstat = seq(X_trn_boston_min, X_trn_boston_max,
                                    by = 0.01))

# setup models
pred_001 = knn.reg(train = X_trn_boston, test = lstat_grid, y = y_trn_boston, k = 1)
pred_005 = knn.reg(train = X_trn_boston, test = lstat_grid, y = y_trn_boston, k = 5)
pred_010 = knn.reg(train = X_trn_boston, test = lstat_grid, y = y_trn_boston, k = 10)
pred_050 = knn.reg(train = X_trn_boston, test = lstat_grid, y = y_trn_boston, k = 50)
pred_100 = knn.reg(train = X_trn_boston, test = lstat_grid, y = y_trn_boston, k = 100)
```

```
pred_250 = knn.reg(train = X_trn_boston, test = lstat_grid, y = y_trn_boston, k = 250)

par(mfrow = c(3, 2))

plot(medv ~ lstat, data = trn_boston, cex = .8, col = "dodgerblue", main = "k = 1")
lines(lstat_grid$lstat, pred_001$pred, col = "darkorange", lwd = 0.25)

plot(medv ~ lstat, data = trn_boston, cex = .8, col = "dodgerblue", main = "k = 5")
lines(lstat_grid$lstat, pred_005$pred, col = "darkorange", lwd = 0.75)

plot(medv ~ lstat, data = trn_boston, cex = .8, col = "dodgerblue", main = "k = 10")
lines(lstat_grid$lstat, pred_010$pred, col = "darkorange", lwd = 1)

plot(medv ~ lstat, data = trn_boston, cex = .8, col = "dodgerblue", main = "k = 25")
lines(lstat_grid$lstat, pred_050$pred, col = "darkorange", lwd = 1.5)

plot(medv ~ lstat, data = trn_boston, cex = .8, col = "dodgerblue", main = "k = 50")
lines(lstat_grid$lstat, pred_100$pred, col = "darkorange", lwd = 2)

plot(medv ~ lstat, data = trn_boston, cex = .8, col = "dodgerblue", main = "k = 250")
lines(lstat_grid$lstat, pred_250$pred, col = "darkorange", lwd = 2)
```