R Notebook

Code ▾

This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

Hide

```
library(dplyr)
library(tidyverse)
library(bnlearn)
library(caret)
library(e1071)
```

Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.

Hide

```
#1 . Read the data from 2020_bn_nb_data.txt file :-
grades<-read.table('D:/6th sem/Artificial Intelligence/2020_bn_nb_data.txt', header = TRUE )
head(grades)
```

	EC100 <chr>	EC160 <chr>	IT101 <chr>	IT161 <chr>	MA101 <chr>	PH100 <chr>	PH160 <chr>	HS101 <chr>	QP <chr>
1	BC	CC	BB	BC	CC	BC	AA	BB	y
2	CC	BC	BB	BB	CC	BC	AB	BB	y
3	AB	BB	AB	AB	BB	CC	BC	AB	y
4	BC	CC	BB	BB	BB	BB	BC	BB	y
5	BC	AB	CD	BC	BC	BC	BC	CD	y
6	DD	CC	DD	CD	CD	CC	BC	BC	n

6 rows

Hide

```
print("dimensions of data given :")
```

```
[1] "dimensions of data given :"
```

Hide

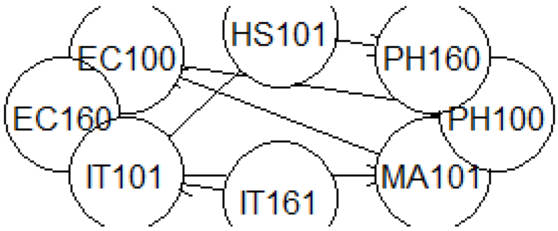
```
print(dim(grades))
```

```
[1] 232   9
```

Hide

```
#2.Consider grades earned in each of the courses as random variables and learn the dependenci
es between courses.

grades<-lapply(grades , as.factor)
grades<-data.frame(grades)
grades.net<-hc(grades[,-9],score="k2")
plot(grades.net)
```



Hide

```
print(grades.net)
```

```


  Bayesian network learned via Score-based methods

  model:
   [IT161][IT101|IT161][MA101|IT101][HS101|IT101][EC100|MA101][PH160|HS101][EC160|EC100][PH10
0|EC100]
  nodes:                                 8
  arcs:                                  7
    undirected arcs:                     0
    directed arcs:                       7
  average markov blanket size:           1.75
  average neighbourhood size:            1.75
  average branching factor:              0.88

  learning algorithm:                    Hill-Climbing
  score:                                 Cooper & Herskovits' K2
  tests used in the learning procedure:  105
  optimized:                             TRUE
```

Hide

```
#3. Using the data, learn the CPTs for each course node.

dag = model2network("[IT161][IT101|IT161][MA101|IT101][HS101|IT101][EC100|MA101][PH160|HS101]
[EC160|EC100][PH100|EC100]")
grades.fit = bn.fit(dag, grades[,-9])
print('Conditional Probability tables for each Nodes : ')
```

```
[1] "Conditional Probability tables for each Nodes : "
```

Hide

```
print(grades.fit)
```

```
  Bayesian network parameters

  Parameters of node EC100 (multinomial distribution)

Conditional probability table:
 
     MA101
EC100         AA         AB         BB         BC         CC         CD         DD          F
   AA 0.75000000 0.07692308 0.03846154 0.01851852 0.00000000 0.00000000 0.00000000 0.00000000
   AB 0.00000000 0.46153846 0.25000000 0.05555556 0.00000000 0.00000000 0.00000000 0.00000000
   BB 0.25000000 0.23076923 0.32692308 0.22222222 0.04081633 0.00000000 0.00000000 0.00000000
   BC 0.00000000 0.15384615 0.28846154 0.27777778 0.32653061 0.00000000 0.00000000 0.00000000
   CC 0.00000000 0.07692308 0.09615385 0.24074074 0.32653061 0.04166667 0.00000000 0.00000000
   CD 0.00000000 0.00000000 0.00000000 0.12962963 0.26530612 0.33333333 0.04761905 0.00000000
   DD 0.00000000 0.00000000 0.00000000 0.03703704 0.04081633 0.50000000 0.19047619 0.00000000
   F  0.00000000 0.00000000 0.00000000 0.01851852 0.00000000 0.12500000 0.76190476 1.00000000

  Parameters of node EC160 (multinomial distribution)

Conditional probability table:
 
     EC100
EC160         AA         AB         BB         BC         CC         CD         DD          F
   AA 0.42857143 0.22727273 0.05714286 0.04166667 0.00000000 0.00000000 0.00000000 0.00000000
   AB 0.42857143 0.22727273 0.08571429 0.04166667 0.08333333 0.00000000 0.00000000 0.00000000
   BB 0.14285714 0.31818182 0.20000000 0.22916667 0.08333333 0.03448276 0.05000000 0.00000000
   BC 0.00000000 0.22727273 0.42857143 0.43750000 0.36111111 0.17241379 0.00000000 0.00000000
   CC 0.00000000 0.00000000 0.22857143 0.25000000 0.30555556 0.34482759 0.25000000 0.02857143
   CD 0.00000000 0.00000000 0.00000000 0.00000000 0.11111111 0.27586207 0.55000000 0.40000000
   DD 0.00000000 0.00000000 0.00000000 0.00000000 0.05555556 0.17241379 0.15000000 0.34285714
   F  0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.22857143

  Parameters of node HS101 (multinomial distribution)

Conditional probability table:
 
     IT101
HS101         AA         AB         BB         BC         CC         CD         DD          F
   AA 0.58333333 0.56000000 0.32352941 0.10204082 0.07142857 0.05714286 0.00000000 0.00000000
   AB 0.33333333 0.24000000 0.11764706 0.22448980 0.14285714 0.08571429 0.00000000 0.00000000
   BB 0.00000000 0.12000000 0.26470588 0.26530612 0.26190476 0.11428571 0.00000000 0.00000000
   BC 0.08333333 0.08000000 0.08823529 0.24489796 0.23809524 0.20000000 0.04347826 0.00000000
   CC 0.00000000 0.00000000 0.11764706 0.12244898 0.14285714 0.11428571 0.26086957 0.00000000
   CD 0.00000000 0.00000000 0.05882353 0.02040816 0.14285714 0.20000000 0.13043478 0.08333333
   DD 0.00000000 0.00000000 0.02941176 0.02040816 0.00000000 0.22857143 0.52173913 0.58333333
   F  0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.04347826 0.33333333

  Parameters of node IT101 (multinomial distribution)

Conditional probability table:
 
     IT161
IT101         AA         AB         BB         BC         CC         CD         DD          F
   AA 0.35000000 0.08000000 0.05714286 0.02040816 0.00000000 0.00000000 0.00000000 0.00000000
   AB 0.30000000 0.40000000 0.17142857 0.02040816 0.02380952 0.02857143 0.00000000 0.00000000
   BB 0.25000000 0.40000000 0.31428571 0.14285714 0.00000000 0.02857143 0.00000000 0.00000000
```

```
   BC 0.10000000 0.04000000 0.28571429 0.36734694 0.28571429 0.14285714 0.04347826 0.00000000
   CC 0.00000000 0.08000000 0.14285714 0.32653061 0.33333333 0.11428571 0.04347826 0.00000000
   CD 0.00000000 0.00000000 0.02857143 0.12244898 0.26190476 0.31428571 0.21739130 0.33333333
   DD 0.00000000 0.00000000 0.00000000 0.00000000 0.04761905 0.34285714 0.39130435 0.00000000
   F  0.00000000 0.00000000 0.00000000 0.00000000 0.04761905 0.02857143 0.30434783 0.66666667

  Parameters of node IT161 (multinomial distribution)

Conditional probability table:
         AA         AB         BB         BC         CC         CD         DD          F
0.08620690 0.10775862 0.15086207 0.21120690 0.18103448 0.15086207 0.09913793 0.01293103

  Parameters of node MA101 (multinomial distribution)

Conditional probability table:

     IT101
MA101         AA         AB         BB         BC         CC         CD         DD          F
   AA 0.16666667 0.04000000 0.00000000 0.00000000 0.02380952 0.00000000 0.00000000 0.00000000
   AB 0.25000000 0.20000000 0.02941176 0.08163265 0.00000000 0.00000000 0.00000000 0.00000000
   BB 0.33333333 0.56000000 0.38235294 0.22448980 0.19047619 0.05714286 0.00000000 0.00000000
   BC 0.16666667 0.16000000 0.29411765 0.36734694 0.23809524 0.22857143 0.08695652 0.00000000
   CC 0.08333333 0.00000000 0.20588235 0.28571429 0.35714286 0.31428571 0.04347826 0.00000000
   CD 0.00000000 0.04000000 0.08823529 0.02040816 0.16666667 0.11428571 0.30434783 0.08333333
   DD 0.00000000 0.00000000 0.00000000 0.02040816 0.02380952 0.22857143 0.39130435 0.16666667
   F  0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.05714286 0.17391304 0.75000000

  Parameters of node PH100 (multinomial distribution)

Conditional probability table:

     EC100
PH100         AA         AB         BB         BC         CC         CD         DD          F
   AA 0.71428571 0.40909091 0.22857143 0.08333333 0.00000000 0.00000000 0.00000000 0.00000000
   AB 0.14285714 0.31818182 0.20000000 0.18750000 0.05555556 0.00000000 0.00000000 0.00000000
   BB 0.00000000 0.18181818 0.31428571 0.29166667 0.13888889 0.03448276 0.05000000 0.00000000
   BC 0.14285714 0.04545455 0.14285714 0.22916667 0.33333333 0.13793103 0.00000000 0.00000000
   CC 0.00000000 0.04545455 0.11428571 0.18750000 0.25000000 0.41379310 0.20000000 0.02857143
   CD 0.00000000 0.00000000 0.00000000 0.02083333 0.19444444 0.31034483 0.45000000 0.11428571
   DD 0.00000000 0.00000000 0.00000000 0.00000000 0.02777778 0.10344828 0.20000000 0.45714286
   F  0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.10000000 0.40000000

  Parameters of node PH160 (multinomial distribution)

Conditional probability table:

     HS101
PH160         AA         AB         BB         BC         CC         CD         DD          F
   AA 0.23809524 0.17647059 0.05000000 0.11111111 0.07692308 0.10000000 0.03448276 0.00000000
   AB 0.23809524 0.11764706 0.15000000 0.13888889 0.07692308 0.10000000 0.10344828 0.00000000
   BB 0.16666667 0.26470588 0.17500000 0.16666667 0.00000000 0.00000000 0.00000000 0.20000000
   BC 0.21428571 0.32352941 0.45000000 0.22222222 0.50000000 0.30000000 0.10344828 0.00000000
   CC 0.09523810 0.08823529 0.12500000 0.30555556 0.15384615 0.45000000 0.24137931 0.00000000
   CD 0.04761905 0.02941176 0.02500000 0.05555556 0.11538462 0.05000000 0.37931034 0.00000000
   DD 0.00000000 0.00000000 0.02500000 0.00000000 0.07692308 0.00000000 0.13793103 0.40000000
   F  0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.40000000
```

Hide

```
#4.What grade will a student get in PH100 if he earns DD in EC100, CC in IT101 and CD in MA10
1.

g1<-cpquery(grades.fit, event =c(PH100=="AA"), evidence = (EC100=="DD" & IT101=="CC" & MA101=
="CD") , n =1000)
g2<-cpquery(grades.fit, event =c(PH100=="AB"), evidence = (EC100=="DD" & IT101=="CC" & MA101=
="CD") , n =1000)
g3<-cpquery(grades.fit, event =c(PH100=="BB"), evidence = (EC100=="DD" & IT101=="CC" & MA101=
="CD") , n =1000)
g4<-cpquery(grades.fit, event =c(PH100=="BC"), evidence = (EC100=="DD" & IT101=="CC" & MA101=
="CD") , n =1000)
g5<-cpquery(grades.fit, event =c(PH100=="CC"), evidence = (EC100=="DD" & IT101=="CC" & MA101=
="CD") , n =1000)
g6<-cpquery(grades.fit, event =c(PH100=="CD"), evidence = (EC100=="DD" & IT101=="CC" & MA101=
="CD") , n =1000)
g7<-cpquery(grades.fit, event =c(PH100=="DD"), evidence = (EC100=="DD" & IT101=="CC" & MA101=
="CD") , n =1000)
g8<-cpquery(grades.fit, event =c(PH100=="F"), evidence = (EC100=="DD" & IT101=="CC" & MA101==
"CD") , n =1000)
Ph100_grade = c(g1,g2,g3,g4,g5,g6,g7,g8)
print("Probability for all grades PH100 : ")
```

```
[1] "Probability for all grades PH100 : "
```

Hide

```
print( "AA          AB          BB          BC          CC          CD          DD          F")
```

```
[1] "AA          AB          BB          BC          CC          CD          DD          F"
```

Hide

```
print(Ph100_grade)
```

```
[1] 0.0000000 0.0000000 0.0000000 0.0000000 0.1764706 0.3125000 0.1500000 0.0000000
```

Hide

```
#p1 <-cpquery(grades.fit ,(PH100=="AA") , (EC100=="DD" & IT101="CC" & MA101="CD") , N=1000)
```

Hide

```
#5. Convert each grade to corresponding number so that I can fit a model into it
convert_grades <- function(x) {
  A <- factor(x, levels=c("AA", "AB",
                     "BB", "BC",
                     "CC", "CD",
                     "DD",  "F",
                     "y" , "n"))
  values <- c(10, 9,
              8, 7,
              6, 5,
              4, 3,
              TRUE , FALSE)
  values[A]
}

num_grades <- grades
num_grades[] <- lapply(num_grades, convert_grades)
print(num_grades)
```

EC100 <dbl>	EC160 <dbl>	IT101 <dbl>	IT161 <dbl>	MA101 <dbl>	PH100 <dbl>	PH160 <dbl>	HS101 <dbl>	QP <dbl>
7	6	8	7	6	7	10	8	1
6	7	8	8	6	7	9	8	1
9	8	9	9	8	6	7	9	1
7	6	8	8	8	8	7	8	1
7	9	5	7	7	7	7	5	1
4	6	4	5	5	6	7	7	0
7	8	5	6	6	7	8	9	1
8	6	6	6	8	8	8	7	1
10	9	10	10	10	10	8	9	1
8	8	7	6	9	10	9	10	1

1-10 of 232 rows Previous 1 2 3 4 5 6 ... 24 Next

Hide

```
NA
NA
```

Hide

```
print(dim(grades))
```

```
[1] 232   9
```

Hide

```
x_train = num_grades[1:165,-9]
x_test = num_grades[166:232,-9]
y_train = num_grades[1:165,9]
y_test = num_grades[166:232,9]
print(y_train)
```

```
  [1] 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 0 0 1
1 0 1 1 1 1 1 0
 [53] 1 1 1 0 0 1 1 0 0 0 1 0 1 1 1 1 0 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 0 1 0 1 1 1 0 1 0 1
1 1 0 0 1 0 1 1
[105] 1 1 1 1 1 1 0 0 0 1 0 1 1 0 1 1 1 1 0 0 1 0 1 1 0 0 1 1 1 1 0 1 1 1 0 0 1 1 1 1 1 1 0 0
1 0 0 0 0 1 0 0
[157] 0 1 1 1 1 0 1 1 1
```

Hide

```
print(length(y_test))
```

```
[1] 67
```

Hide

```
#print(dim(x_test))
print(str(x_train))
```

```
'data.frame':   165 obs. of  8 variables:
 $ EC100: num  7 6 9 7 7 4 7 8 10 8 ...
 $ EC160: num  6 7 8 6 9 6 8 6 9 8 ...
 $ IT101: num  8 8 9 8 5 4 5 6 10 7 ...
 $ IT161: num  7 8 9 8 7 5 6 6 10 6 ...
 $ MA101: num  6 6 8 8 7 5 6 8 10 9 ...
 $ PH100: num  7 7 6 8 7 6 7 8 10 10 ...
 $ PH160: num  10 9 7 7 7 7 8 8 8 9 ...
 $ HS101: num  8 8 9 8 5 7 9 7 9 10 ...
NULL
```

Hide

```
#6 . split data into training and test data sets
num_grades$QP <- factor(num_grades$QP, levels = c(0,1), labels = c("False", "True"))
indxTrain <- createDataPartition(y = num_grades$QP,p = 0.70,list = FALSE)
training <- num_grades[indxTrain,]
testing <- num_grades[-indxTrain,] #Check dimensions of the split > prop.table(table(data$Out
come)) * 100
print(dim(testing))
```

```
[1] 69  9
```

Hide

```
print(head(training))
```

	EC100 <dbl>	EC160 <dbl>	IT101 <dbl>	IT161 <dbl>	MA101 <dbl>	PH100 <dbl>	PH160 <dbl>	HS101 <dbl>	QP <fctr>
1	7	6	8	7	6	7	10	8	True
2	6	7	8	8	6	7	9	8	True
3	9	8	9	9	8	6	7	9	True
4	7	6	8	8	8	8	7	8	True
5	7	9	5	7	7	7	7	5	True
6	4	6	4	5	5	6	7	7	False

6 rows

Hide

```
X_Train = training[,-9]
Y_Train = training$QP

V= sample_n(num_grades, 165)
XX_Train = V[,-9]
YY_Train = V$QP
```

Hide

```
#7. Training the Model for given data
model = train(XX_Train , YY_Train ,'nb',trControl=trainControl(method='cv',number=10))
```

```
Numerical 0 probability for all classes with observation 15Numerical 0 probability for all cl
asses with observation 15Numerical 0 probability for all classes with observation 1Numerical
0 probability for all classes with observation 1Numerical 0 probability for all classes with
observation 4Numerical 0 probability for all classes with observation 2Numerical 0 probabilit
y for all classes with observation 8Numerical 0 probability for all classes with observation
9Numerical 0 probability for all classes with observation 17Numerical 0 probability for all c
lasses with observation 17Numerical 0 probability for all classes with observation 10Numerica
l 0 probability for all classes with observation 10Numerical 0 probability for all classes wi
th observation 1Numerical 0 probability for all classes with observation 1Numerical 0 probabi
lity for all classes with observation 5Numerical 0 probability for all classes with observati
on 5Numerical 0 probability for all classes with observation 8Numerical 0 probability for all
classes with observation 14
```

Hide

```
print(model)
```

```
Naive Bayes

165 samples
  8 predictor
  2 classes: 'False', 'True'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 149, 149, 149, 147, 148, 148, ...
Resampling results across tuning parameters:

  usekernel  Accuracy   Kappa
  FALSE      0.9816176  0.9558449
   TRUE      0.9878676  0.9712296

Tuning parameter 'fL' was held constant at a value of 0
Tuning parameter 'adjust' was held constant at
 a value of 1
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were fL = 0, usekernel = TRUE and adjust = 1.
```

Hide

```
# 8 .Get the confusion matrix to see accuracy value and other parameter values confusionMatri
x(Predict, num_grades$QP )
Predict <- predict(model,newdata = testing[,-9] )
```

```
Numerical 0 probability for all classes with observation 2Numerical 0 probability for all cla
sses with observation 47
```

Hide

```
#print(Predict)
cm <- table(testing$QP, Predict)
print("\nConfusion matrix = ")
```

```
[1] "\nConfusion matrix = "
```

Hide

```
print(cm)
```

```
       Predict
        False True
  False    21    0
  True      3   45
```

Hide

```
n = sum(cm) # number of instances
diag = diag(cm)
acc = sum(diag)/n
sprintf("Accurcy = %f",acc*100)
```

```
[1] "Accurcy = 95.652174"
```

Hide

```
#9, Picking 20 random insances and predict
Random_20 = sample_n(num_grades, 30)
Predict_20 <- predict(model,newdata = Random_20[,-9] )
```

```
Numerical 0 probability for all classes with observation 5
```

Hide

```
cm <- table(Random_20$QP, Predict_20)
print(cm)
```

```
       Predict_20
        False True
  False    11    0
  True      1   18
```

Hide

```
accuracy <- mean(Random_20$QP == Predict_20)
error <- mean(Random_20$QP != Predict_20)
sprintf("Error = %f", error)
```

```
[1] "Error = 0.033333"
```

Hide

```
sprintf("accuracy = %f",accuracy)
```

```
[1] "accuracy = 0.966667"
```

Hide

```
#10 repeat previos part on dependent data
#print(head(num_grades))
num_grades$IT101 = (num_grades$EC100 + num_grades$EC160)/2
num_grades$MA101 = (num_grades$IT161 + num_grades$PH100)/2
num_grades$PH160 = num_grades$PH100
#print(head(num_grades))
indxTrain <- createDataPartition(y = num_grades$QP,p = 0.70,list = FALSE)
training <- num_grades[indxTrain,]
testing <- num_grades[-indxTrain,] #Check dimensions of the split > prop.table(table(data$Out
come)) * 100
#print(dim(testing))
#print(head(training))
X_Train = training[,-9]
Y_Train = training$QP
model = train(XX_Train , YY_Train ,'nb',trControl=trainControl(method='cv',number=10))
```

```
Numerical 0 probability for all classes with observation 6Numerical 0 probability for all cla
sses with observation 3Numerical 0 probability for all classes with observation 4Numerical 0
probability for all classes with observation 4Numerical 0 probability for all classes with ob
servation 10Numerical 0 probability for all classes with observation 5Numerical 0 probability
for all classes with observation 17Numerical 0 probability for all classes with observation 5
Numerical 0 probability for all classes with observation 15Numerical 0 probability for all cl
asses with observation 5Numerical 0 probability for all classes with observation 7Numerical 0
probability for all classes with observation 15Numerical 0 probability for all classes with o
bservation 13Numerical 0 probability for all classes with observation 2Numerical 0 probabilit
y for all classes with observation 2Numerical 0 probability for all classes with observation
5Numerical 0 probability for all classes with observation 5Numerical 0 probability for all cl
asses with observation 5Numerical 0 probability for all classes with observation 9
```

Hide

```
print(model)
```

```
Naive Bayes

165 samples
  8 predictor
  2 classes: 'False', 'True'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 147, 149, 147, 149, 149, 149, ...
Resampling results across tuning parameters:

  usekernel  Accuracy   Kappa
  FALSE      0.9812500  0.9519623
   TRUE      0.9635621  0.9168068

Tuning parameter 'fL' was held constant at a value of 0
Tuning parameter 'adjust' was held constant at
 a value of 1
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were fL = 0, usekernel = FALSE and adjust = 1.
```

Hide

```
Predict <- predict(model,newdata = testing[,-9] )
```

```
Numerical 0 probability for all classes with observation 46Numerical 0 probability for all cl
asses with observation 61
```

Hide

```
cm <- table(testing$QP, Predict)
print(cm)
```

```
       Predict
        False True
  False    20    1
  True      2   46
```

Hide

```
n = sum(cm) # number of instances
diag = diag(cm)
acc = sum(diag)/n
sprintf("Accuracy = %f",acc*100)
```

```
[1] "Accuracy = 95.652174"
```