Exam 3

Code

4/25/2020

Hide

```
smoke <- read.csv("cancer.csv")

str(smoke)
```

```
'data.frame':   44 obs. of  9 variables:
 $ State        : Factor w/ 44 levels "AK","AL","AR",..: 2 4 3 5 6 8 7 9 11 12 ...
 $ Region       : Factor w/ 4 levels "East","North",..: 3 4 3 4 1 1 1 3 4 2 ...
 $ smoke_density: Factor w/ 5 levels "High","Low","Medium",..: 1 2 1 5 2 3 2 3 2 2 ...
 $ has_cancer   : num  18.2 25.8 18.2 28.6 31.1 ...
 $ More20       : num  2.9 3.52 2.99 4.46 5.11 4.78 5.6 4.46 3.08 4.75 ...
 $ Btw15to20    : num  17.1 19.8 16 22.1 22.8 ...
 $ Btw10to15    : num  1.59 2.75 2.02 2.66 3.35 3.36 3.13 2.41 2.46 2.95 ...
 $ Btw5to10     : num  6.15 6.61 6.94 7.06 7.2 6.45 7.08 6.07 6.62 7.27 ...
 $ Less5        : num  72.3 67.3 72.1 63.8 61.5 ...
```

Hide

```
#We need to shuffle the data:
set.seed(19792020)
group <- runif(nrow(smoke))

#putting random numbers in order and setting them to row locations in smoke:
smoke <- smoke[order(group),]

smoke
```

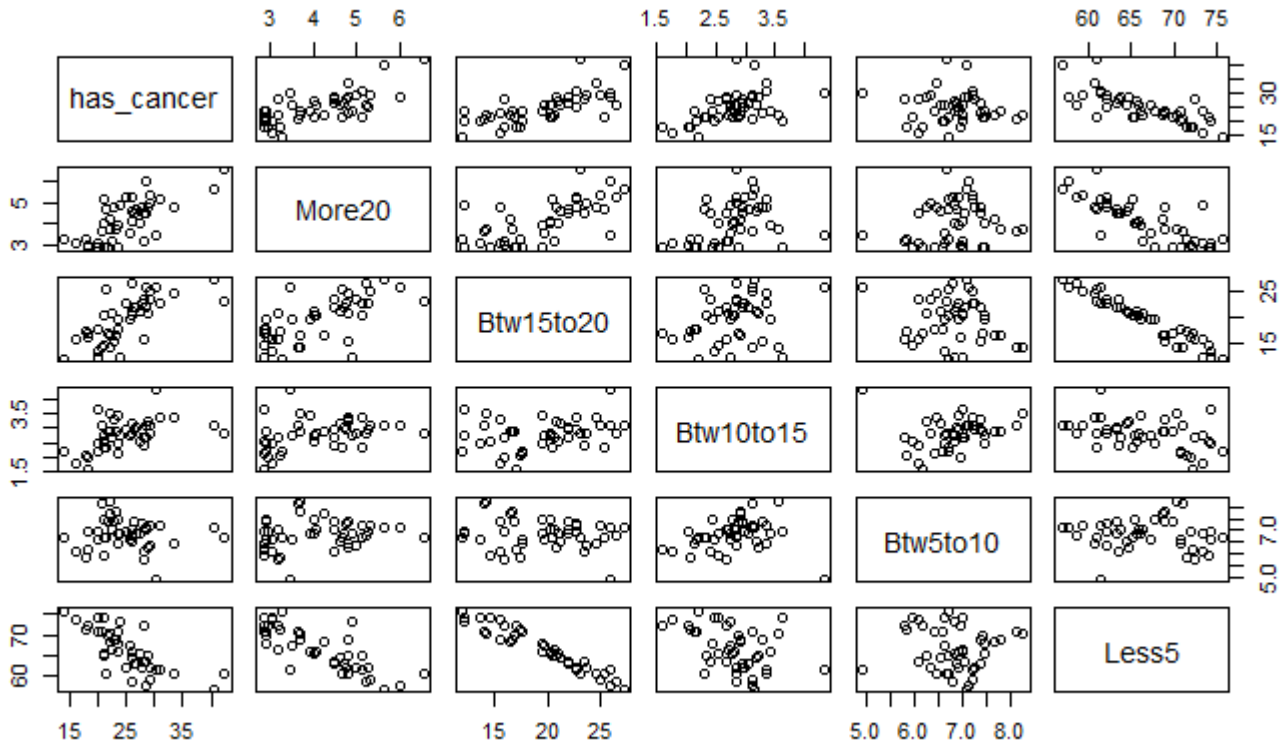
	State <fctr>	Regi… <fctr>	smoke_density <fctr>	has_cancer <dbl>	Mor… <dbl>	Btw15to20 <dbl>	Btw10to15 <dbl>	Btw5to10 <dbl>	Les… <dbl>
1	AL	South	High	18.20	2.90	17.05	1.59	6.15	72.31
41	WI	North	Medium	21.25	5.14	20.55	2.34	6.73	65.24
28	NY	East	Low	29.14	5.30	25.02	3.10	7.23	59.35
29	ND	North	Medium	19.96	2.89	12.12	3.62	6.99	74.38
9	ID	West	Low	20.10	3.08	13.58	2.46	6.62	74.26
8	FL	South	Medium	28.27	4.46	23.57	2.41	6.07	63.49
20	MN	East	Medium	22.06	3.72	14.20	3.54	8.28	70.26
17	MD	East	Low	25.91	5.21	26.48	2.85	6.81	58.65
24	NV	West	High	23.32	3.72	16.70	2.92	7.80	68.86

	State <fctr>	Regi… <fctr>	smoke_density <fctr>	has_cancer <dbl>	Mor… <dbl>	Btw15to20 <dbl>	Btw10to15 <dbl>	Btw5to10 <dbl>	Les… <dbl>
33	RI	East	Low	29.18	4.99	23.68	2.84	6.35	62.14

1-10 of 44 rows Previous **1** 2 3 4 5 Next

Hide

```
#70% of my data should be used for TRAINING!
smoke_train <- smoke[1:31,]

#30% of my data should be used for TESTING!
smoke_test <- smoke[32:44,]

#################################################################
#Regression Section:


#scatterplot matrix & correlation matrix:
pairs(smoke[, c(4,5,6,7,8,9)])
```



Hide

```
cor(smoke[, c(4,5,6,7,8,9)])
```

```
            has_cancer     More20  Btw15to20  Btw10to15    Btw5to10       Less5
has_cancer  1.00000000  0.7036219  0.6974025  0.4873896 -0.06848123 -0.75298786
More20      0.70362186  1.0000000  0.6585011  0.3588140  0.16215663 -0.79262369
Btw15to20   0.69740250  0.6585011  1.0000000  0.2827431 -0.15158448 -0.96427017
Btw10to15   0.48738962  0.3588140  0.2827431  1.0000000  0.18871294 -0.42806057
Btw5to10   -0.06848123  0.1621566 -0.1515845  0.1887129  1.00000000 -0.04940869
Less5      -0.75298786 -0.7926237 -0.9642702 -0.4280606 -0.04940869  1.00000000
```

Hide

```
#Using the aggregate function, find the mean percent of smokers with lung cancer based on the Re
gion and smoke_density variables:

aggregate(has_cancer ~ Region + smoke_density, data = smoke, FUN= mean)
```

Region <fctr>	smoke_density <fctr>	has_cancer <dbl>
North	High	26.70667
South	High	19.20800
West	High	26.83000
East	Low	28.36667
North	Low	30.41667
South	Low	22.57000
West	Low	22.71000
East	Medium	27.09000
North	Medium	23.26600
South	Medium	23.75667

1-10 of 14 rows Previous **1** 2 Next

Hide

```
#Using the training data, create a linear regression model with the Less5 column as x (independe
nt variable), and the has_cancer column as y (dependent variable). Call the model m1 and use the
summary command to determine the value of R2. The model could possibly be used to predict the pe
rcentage of smokers that might have lung cancer in each state.

m1<-lm(has_cancer ~ Less5, data = smoke_train)

summary(m1)
```

```
Call:
lm(formula = has_cancer ~ Less5, data = smoke_train)

Residuals:
    Min      1Q  Median      3Q     Max
-5.2264 -2.2023 -0.4965  1.7174 13.1241

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  79.4180     8.8867   8.937 7.93e-10 ***
Less5        -0.8232     0.1327  -6.203 9.15e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.706 on 29 degrees of freedom
Multiple R-squared:  0.5702,    Adjusted R-squared:  0.5554
F-statistic: 38.47 on 1 and 29 DF,  p-value: 9.152e-07
```

Hide

```
#Using the training data, create a multiple linear regression model with the has_cancer column a
s y (dependen tvariable) and all other numerical columns (except last column) as x’s (independen
t variables). Call the model m2 and use the summary command to determine the value of R2. Is thi
s model a better predictor than the model created in problem 3?

m2 <- lm(has_cancer ~ More20 + Btw15to20 + Btw10to15 + Btw5to10, data = smoke)

summary(m2)
```

```
Call:
lm(formula = has_cancer ~ More20 + Btw15to20 + Btw10to15 + Btw5to10,
    data = smoke)

Residuals:
    Min      1Q  Median      3Q     Max
-5.7970 -2.4566  0.0087  1.3145  9.9185

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.5897     6.7021   0.983  0.33156
More20        2.3780     0.7756   3.066  0.00393 **
Btw15to20     0.4330     0.1755   2.466  0.01815 *
Btw10to15     2.9272     1.0918   2.681  0.01070 *
Btw5to10     -1.1954     0.8940  -1.337  0.18889
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.407 on 39 degrees of freedom
Multiple R-squared:  0.661, Adjusted R-squared:  0.6262
F-statistic: 19.01 on 4 and 39 DF,  p-value: 9.582e-09
```

NOTE: The R-Squared value for the multiple regression model is larger, therefore it is the better model!

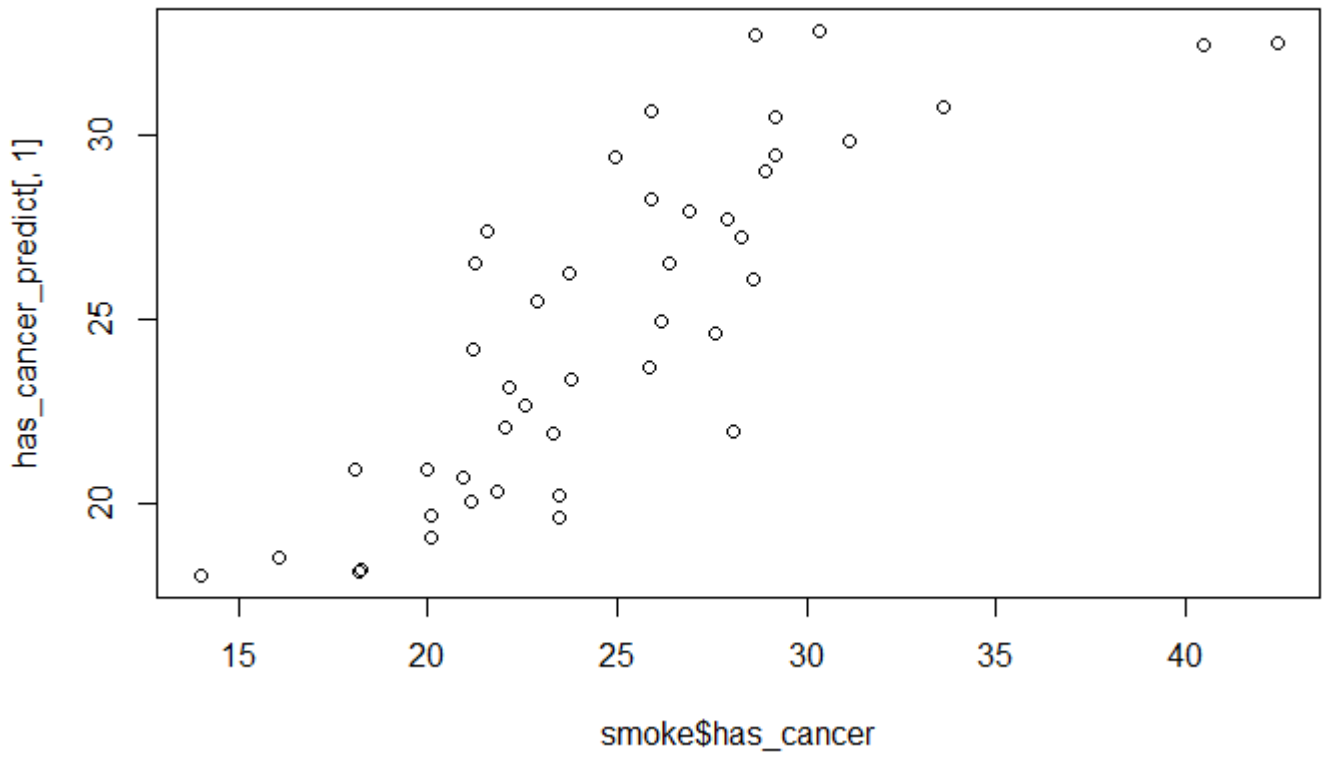
Hide

```
#Using the model created in problem 4, predict the percentage of smokers with lung cancers for t
he test dataframe. With this prediction, create a scatterplot of the predicted values vs the act
ual values of smokers with lung cancer.

has_cancer_predict <- predict(m2,data.frame("More20" = smoke$More20, "Btw15to20" = smoke$Btw15to
20,"Btw10to15" = smoke$Btw10to15,"Btw5to10" = smoke$Btw5to10),interval ="prediction")

has_cancer_predict
```

```
        fit      lwr      upr
1  18.16989 10.73968 25.60011
2  26.51395 19.28248 33.74543
3  30.45666 23.30276 37.61055
4  20.94968 13.33301 28.56634
5  19.08039 11.92418 26.23659
6  27.19832 20.01987 34.37678
7  22.04765 14.45354 29.64177
8  30.64504 23.45325 37.83683
9  21.88899 14.70082 29.07716
10 29.43025 22.32784 36.53266
11 23.12710 15.98268 30.27151
12 26.21695 19.09524 33.33866
13 20.71517 13.30632 28.12402
14 18.07894 10.76799 25.38988
15 18.23495 11.03121 25.43868
16 29.39154 22.30861 36.47447
17 24.59145 17.57597 31.60693
18 29.82434 22.71738 36.93129
19 18.53317 11.18209 25.88425
20 21.96845 14.67201 29.26490
21 23.68049 16.65125 30.70973
22 32.48145 24.81819 40.14471
23 24.19862 17.09983 31.29741
24 28.22404 21.17341 35.27467
25 19.70418 12.54845 26.85991
26 26.50940 19.43698 33.58181
27 20.91637 13.63516 28.19758
28 32.66638 25.39526 39.93750
29 29.01219 21.83373 36.19066
30 26.09721 19.05922 33.13520
31 27.70072 20.63427 34.76717
32 20.06620 12.78820 27.34421
33 32.81002 24.04993 41.57012
34 24.96183 17.97619 31.94747
35 23.34596 15.50508 31.18683
36 22.68452 15.47233 29.89672
37 19.63039 12.43987 26.82090
38 32.41133 25.14978 39.67287
39 25.45898 18.11842 32.79954
40 20.23166 12.82524 27.63808
41 27.37702 20.09374 34.66029
42 30.71013 23.55336 37.86689
43 27.91744 20.91098 34.92389
44 20.36061 13.11965 27.60157
```

Hide

```
#scatterplot w/ predicted values on the y-axis & actual values on the x-axis:
plot(has_cancer_predict[,1] ~ smoke$has_cancer)
```



Hide

```
NA
NA
```

Hide

```
#############################################################
#kNN section:

#Use the k nearest neighbor function in R and columns 4 to 8 of the training dataframe, to creat
e a model that will predict the Region. Use k=3. Call the model m3. Compare the results of your
 model on the training and test dataframe to see how well it predicts. You should have two table
s of predictions. One for the training dataframe and the other for the test dataframe.

#calling in class package that contains kNN:
require(class)

#only using columns 4 thru 8:
smoke_train <- smoke[1:31, 4:8]
smoke_test <- smoke[32:44, 4:8]

#isolating "Type" column to make target variables:
smoke_train_target <- smoke[1:31,2]
smoke_test_target <- smoke[32:44,2]

#using kNN:
#predictions:
m3 <- knn(train = smoke_train, test = smoke_test, cl = smoke_train_target, k = 3)

#using table() to make a Confusion Matrix to see how well my model predicted the Types:
#the diagonals are the correctly predicted classifications:
table(smoke_test_target, m3)
```

```
                m3
smoke_test_target East North South West
            East     3     2     0    0
            North    1     0     0    1
            South    1     1     1    2
            West     1     0     0    0
```

Hide

```
#Use the k nearest neighbor function in R and columns 4 to 8 of the training dataframe, to creat
e a model that will predict the smoke_density. Use k=3. Call the model m4. Compare the results o
f your model on the training and test dataframe to see how well it predicts. You should have two
tables of predictions. One for the training dataframe and the other for the test dataframe.

#only using columns 4 thru 8:
smoke_train <- smoke[1:31, 4:8]
smoke_test <- smoke[32:44, 4:8]

#isolating "Type" column to make target variables:
smoke_train_target <- smoke[1:31,3]
smoke_test_target <- smoke[32:44,3]

#using kNN:
#predictions:
m4 <- knn(train = smoke_train, test = smoke_test, cl = smoke_train_target, k = 3)

#using table() to make a Confusion Matrix to see how well my model predicted the Types:
#the diagonals are the correctly predicted classifications:
table(smoke_test_target, m4)
```

```
                m4
smoke_test_target High Low Medium Very High Very Low
        High         0   3      0         0        0
        Low          0   3      0         0        0
        Medium       1   3      0         0        0
        Very High    0   1      2         0        0
        Very Low     0   0      0         0        0
```

Hide

```
#########################################################
#Decision Trees (C5.0) Section:

#Use the C5.0 function in R and columns 4 to 8 of the training dataframe, to create a model that
will predict the Region. Call the model m5. Compare the results of your model on the training an
d test dataframe to see how well it predicts. You should have two tables of predictions. One for
the training dataframe and the other for the test dataframe. Finally, PLOT the Decision Tree.

smoke1 <- smoke[,c(2,4,5,6,7,8)]


#85% of my data should be used for TRAINING!
smoke1_train <- smoke1[1:31,]

#15% of my data should be used for TESTING!
smoke1_test <- smoke1[32:44,]

m5 <- C5.0(smoke1_train[,-1], smoke1_train[,1])

summary(m5)
```

```
Call:
C5.0.default(x = smoke1_train[, -1], y = smoke1_train[, 1])


C5.0 [Release 2.07 GPL Edition]     Sat Apr 25 18:41:42 2020
-------------------------------

Class specified by attribute `outcome'

Read 31 cases (6 attributes) from undefined.data

Decision tree:

Btw10to15 <= 2.41: South (8/3)
Btw10to15 > 2.41:
:...More20 <= 4.47: North (13/7)
    More20 > 4.47:
    :...Btw10to15 > 2.97: East (5)
        Btw10to15 <= 2.97:
        :...Btw15to20 <= 23.03: North (3)
            Btw15to20 > 23.03: East (2)


Evaluation on training data (31 cases):

        Decision Tree
      ----------------
      Size      Errors

         5   10(32.3%)   <<


       (a)   (b)   (c)   (d)    <-classified as
      ----  ----  ----  ----
         7     1     1          (a): class East
               9     1          (b): class North
                     5          (c): class South
               6     1          (d): class West


    Attribute usage:

    100.00% Btw10to15
     74.19% More20
     16.13% Btw15to20


Time: 0.0 secs
```

Hide

```
p1 <- predict(m5,smoke1_test[,])
p1
```

```
 [1] North North North North North South East  East  North South East  East  North
Levels: East North South West
```

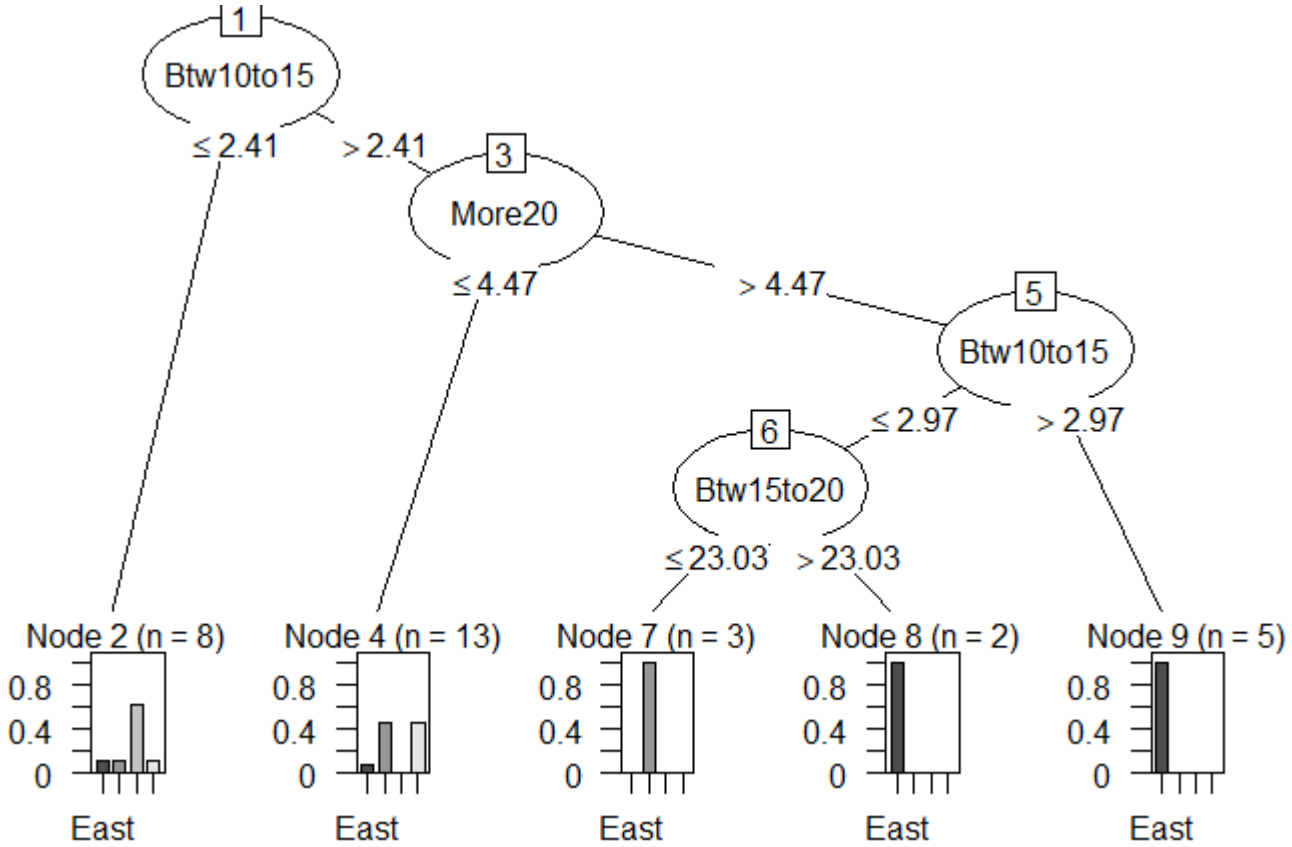
Hide

```
# table(actual values, predicted values):
table(smoke1_test[,1], Predicted = p1)
```

```
      Predicted
       East North South West
  East     4     1     0    0
  North    0     1     1    0
  South    0     4     1    0
  West     0     1     0    0
```

Hide

```
#plotting:
plot(m5)
```



Hide

```
NA
NA
NA
NA
```

Hide

```
#Use the C5.0 function in R and columns 4 to 8 of the training dataframe, to create a model that
will predict the smoke_density. Call the model m6. Compare the results of your model on the trai
ning and test dataframe to see how well it predicts. You should have two tables of predictions.
 One for the training dataframe and the other for the test dataframe. Finally, PLOT the Decision
Tree.


smoke2 <- smoke[,c(3,4,5,6,7,8)]


#85% of my data should be used for TRAINING!
smoke2_train <- smoke2[1:31,]

#15% of my data should be used for TESTING!
smoke2_test <- smoke2[32:44,]

m6 <- C5.0(smoke2_train[,-1], smoke2_train[,1])

summary(m6)
```

```
Call:
C5.0.default(x = smoke2_train[, -1], y = smoke2_train[, 1])


C5.0 [Release 2.07 GPL Edition]     Sat Apr 25 18:41:43 2020
-------------------------------

Class specified by attribute `outcome'

Read 31 cases (6 attributes) from undefined.data

Decision tree:

More20 <= 3.06: High (5/1)
More20 > 3.06:
:...Btw10to15 <= 2.66:
    :...has_cancer <= 20.1: Low (3/1)
    :   has_cancer > 20.1:
    :   :...Btw5to10 <= 6.73: Medium (3)
    :       Btw5to10 > 6.73: High (2/1)
    Btw10to15 > 2.66:
    :...Btw15to20 > 22.72: Low (7)
        Btw15to20 <= 22.72:
        :...More20 > 4.63: Medium (2)
            More20 <= 4.63:
            :...Btw15to20 <= 16.59: Medium (3/1)
                Btw15to20 > 16.59:
                :...Btw5to10 <= 7.12: Low (3)
                    Btw5to10 > 7.12: High (3/1)


Evaluation on training data (31 cases):

        Decision Tree
      ----------------
      Size      Errors

         9    5(16.1%)   <<


       (a)   (b)   (c)   (d)   (e)    <-classified as
      ----  ----  ----  ----  ----
         7                            (a): class High
         1    12     1                (b): class Low
         1           7                (c): class Medium
                                      (d): class Very High
         1     1                      (e): class Very Low


    Attribute usage:

    100.00% More20
```

```
     83.87% Btw10to15
     58.06% Btw15to20
     35.48% Btw5to10
     25.81% has_cancer


Time: 0.0 secs
```

Hide

```
p2 <- predict(m5,smoke2_test[,])
p2
```

```
 [1] North North North North North South East  East  North South East  East  North
Levels: East North South West
```

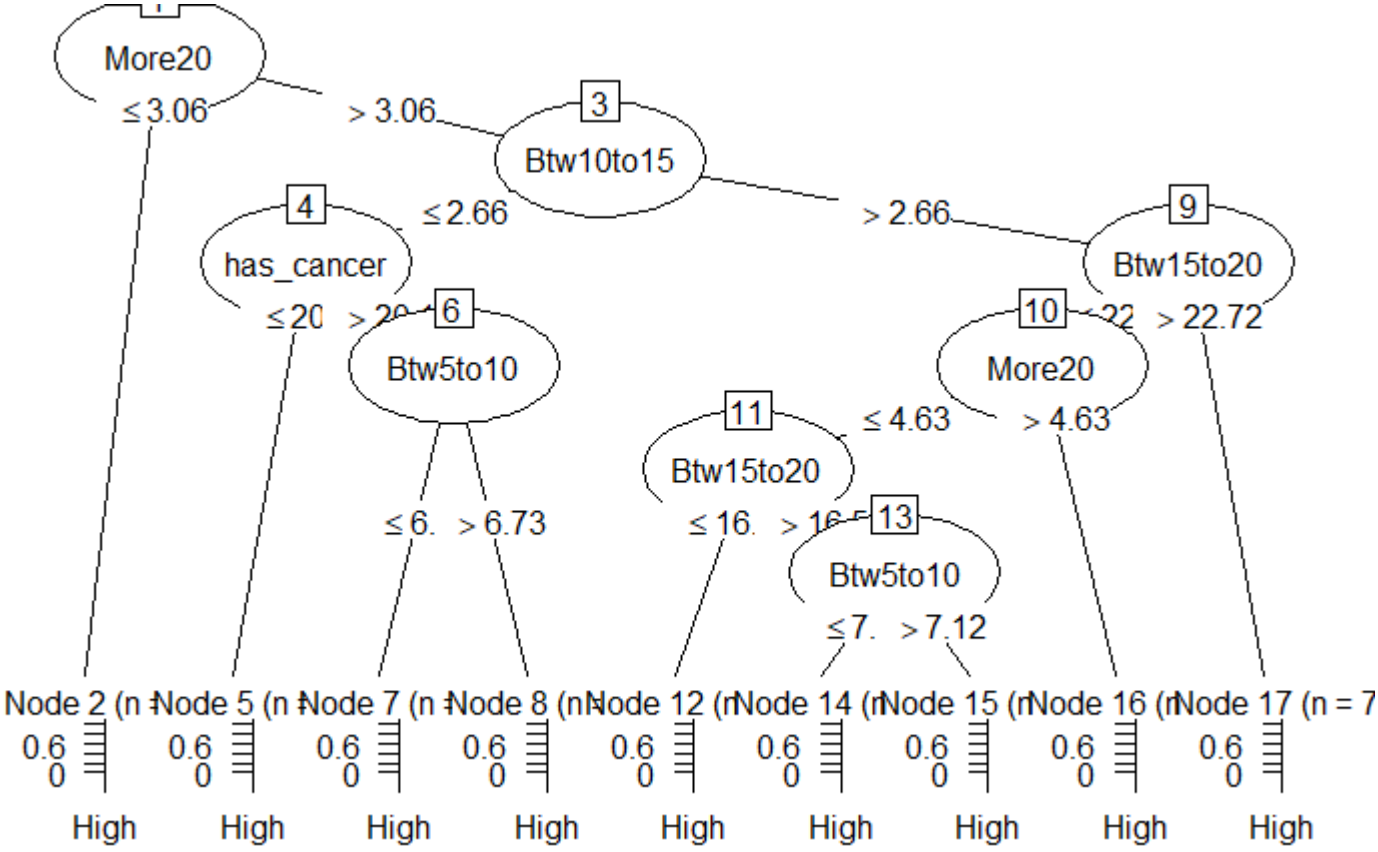
Hide

```
# table(actual values, predicted values):
table(smoke2_test[,1], Predicted = p2)
```

```
           Predicted
            East North South West
  High         0     3     0    0
  Low          2     1     0    0
  Medium       1     3     0    0
  Very High    1     0     2    0
  Very Low     0     0     0    0
```

Hide

```
#plotting:
plot(m6)
```



Hide

NA
NA
NA
NA

Hide

```
#########################################################
#Decision Trees (rpart) Section:

#Use the rpart function in R and columns 4 to 8 of the training dataframe, to create a model tha
t will predict the Region. Call the model m7. Compare the results of your model on the training
 and test dataframe to see how well it predicts. You should have two tables of predictions. One
 for the training dataframe and the other for the test dataframe. Finally, PLOT the Decision Tre
e using rpart.plot

smoke3 <- smoke[,c(2,4,5,6,7,8)]


#85% of my data should be used for TRAINING!
smoke3_train <- smoke3[1:31,]

#15% of my data should be used for TESTING!
smoke3_test <- smoke3[32:44,]


m7 <- rpart(Region ~ . , data = smoke3_train[,], method = "class")



p3 <- predict(m7, smoke3_test[,], type = "class")
p3
```

```
   27    44    11    32    37    14     7    42    31    15     6    18    13
North North North  East North South  East  East North  East  East  East North
Levels: East North South West
```

Hide

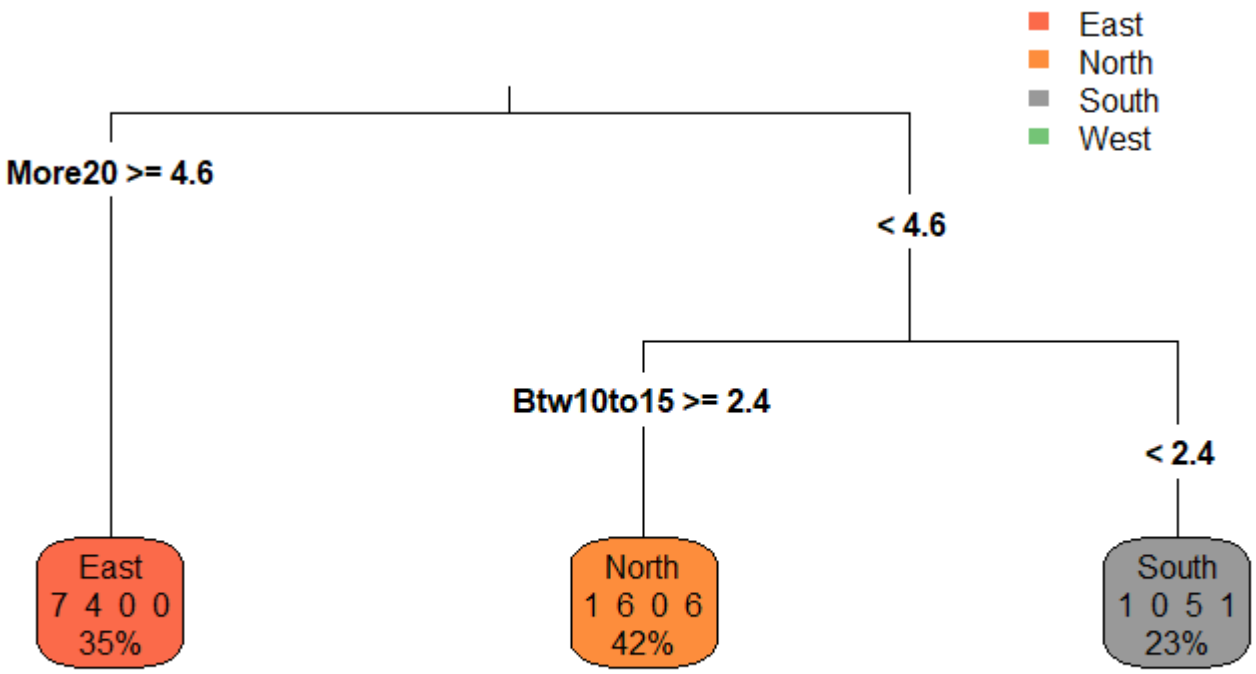
```
# table(actual values, predicted values):
table(smoke3_test[,1], Predicted = p3)
```

```
       Predicted
        East North South West
  East     5     0     0    0
  North    0     1     1    0
  South    1     4     0    0
  West     0     1     0    0
```

Hide

```
#plotting:
rpart.plot(m7, type=3, extra=101, fallen.leaves= TRUE)
```

```
Bad 'data' field in model 'call' (expected a data.frame or a matrix).
To silence this warning:
    Call rpart.plot with roundint=FALSE,
    or rebuild the rpart model with model=TRUE.
```



Hide

```
#Use the rpart function in R and columns 4 to 8 of the training dataframe, to create a model tha
t will predict the smoke_density. Call the model m8. Compare the results of your model on the tr
aining and test dataframe to see how well it predicts. You should have two tables of prediction
s. One for the training dataframe and the other for the test dataframe. Finally, PLOT the Decisi
on Tree using rpart.plot.


smoke4 <- smoke[,c(3,4,5,6,7,8)]


#85% of my data should be used for TRAINING!
smoke4_train <- smoke4[1:31,]

#15% of my data should be used for TESTING!
smoke4_test <- smoke4[32:44,]


m8 <- rpart(smoke_density ~ . , data = smoke4_train[,], method = "class")



p4 <- predict(m8, smoke4_test[,], type = "class")
p4
```

```
    27     44     11     32     37     14      7     42     31     15      6     18     13
  High    Low   High   High   High   High    Low Medium   High    Low    Low Medium   High
Levels: High Low Medium Very High Very Low
```

Hide

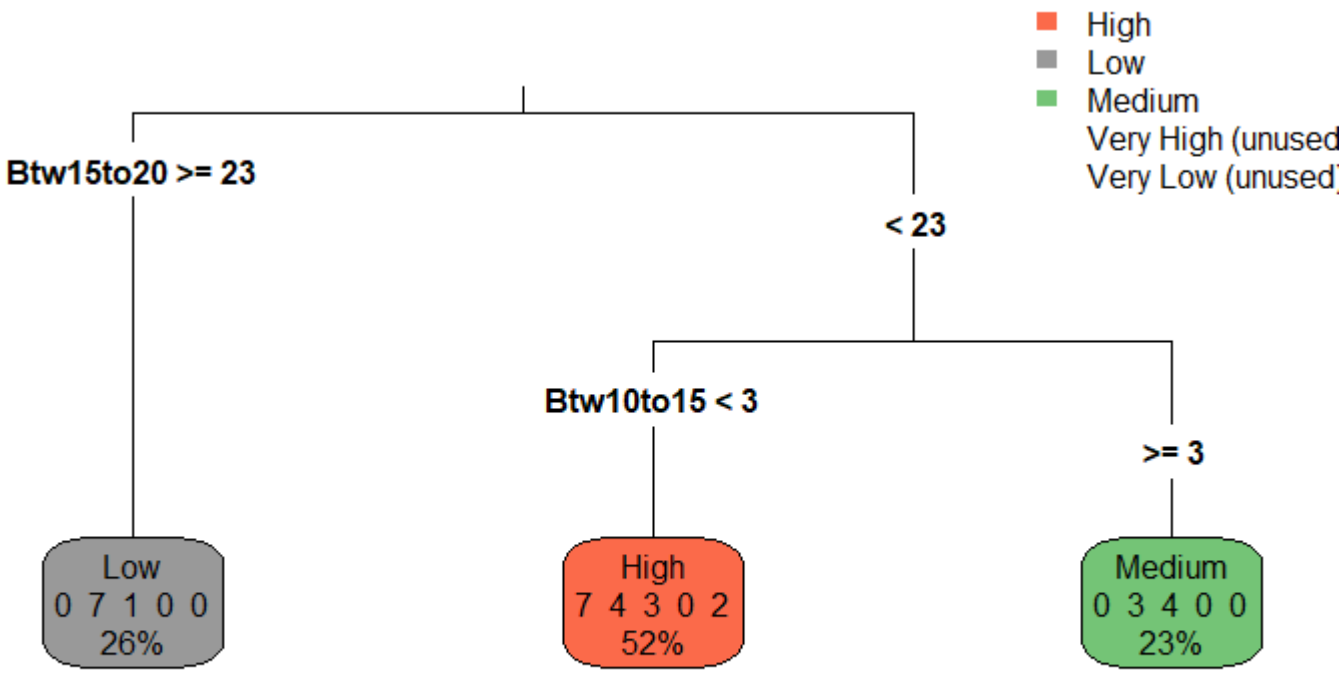
```
# table(actual values, predicted values):
table(smoke4_test[,1], Predicted = p4)
```

```
          Predicted
           High Low Medium Very High Very Low
  High        2   1      0         0        0
  Low         1   1      1         0        0
  Medium      3   1      0         0        0
  Very High   1   1      1         0        0
  Very Low    0   0      0         0        0
```

Hide

```
#plotting:
rpart.plot(m8, type=3, extra=101, fallen.leaves= TRUE)
```

```
Bad 'data' field in model 'call' (expected a data.frame or a matrix).
To silence this warning:
    Call rpart.plot with roundint=FALSE,
    or rebuild the rpart model with model=TRUE.
```



Hide

```
#Association Rules Section:

Baky <- read.transactions("Bakery1.txt" , sep = ",")

summary(Baky)
```

```
transactions as itemMatrix in sparse format with
 9531 rows (elements/itemsets/transactions) and
 1932 columns (items) and a density of 0.0005175983

most frequent items:
  Bread  Coffee     Tea    Cake  Pastry (Other)
   1491    1471     325     277     250    5717

element (itemset/transaction) length distribution:
sizes
   1
9531

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      1       1       1       1       1       1

includes extended item information - examples:
```

	labels <chr>
1	Adjustment
2	Afternoon with the baker
3	Afternoon with the baker Spanish Brunch

3 rows

Hide

```
inspect(Baky[1:5]) #inspecting the first 5 transactions in Grocy.
```

	items <fctr>
[1]	{Bread}
[2]	{Scandinavian}
[3]	{Jam Cookies}
[4]	{Muffin}
[5]	{Pastry Bread}

5 rows

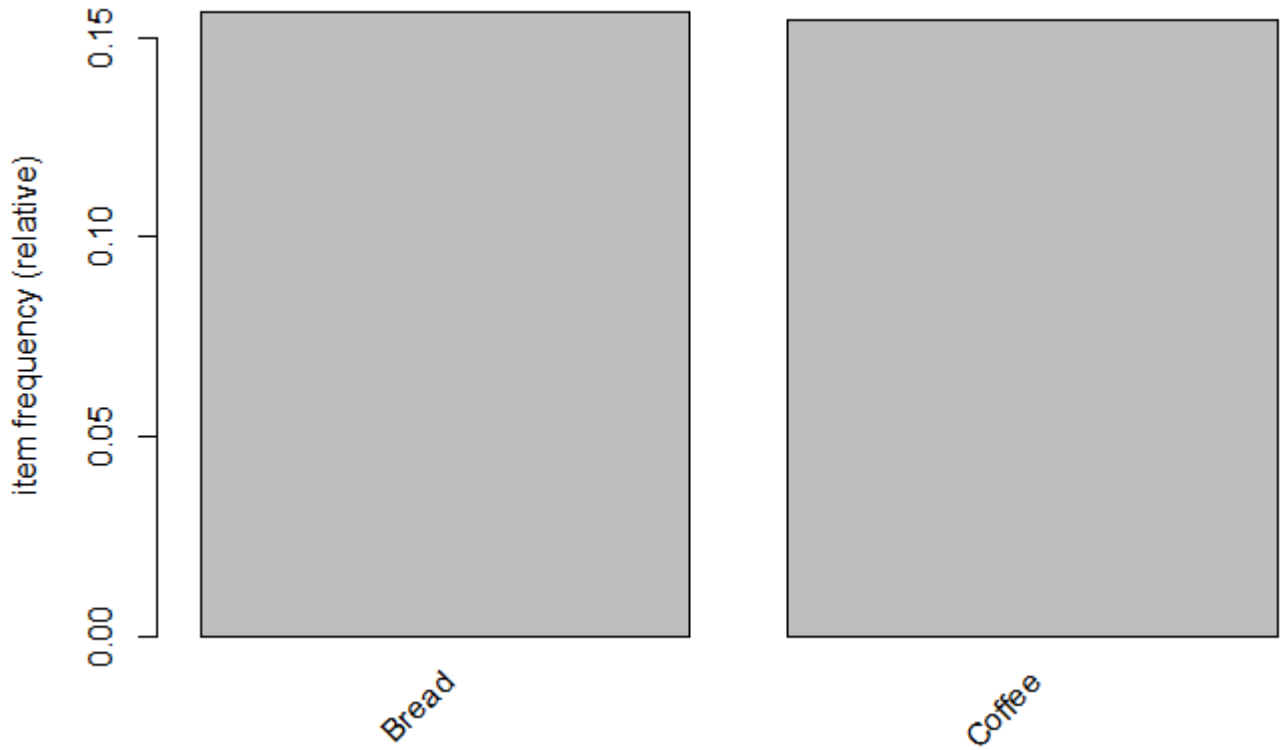
Hide

```
itemFrequency(Baky[,1:6]) #looking at first 6 items' (columns) frequencies.
```

```
                          Adjustment                  Afternoon with the baker
                        0.0001049208                        0.0032525443
Afternoon with the baker Spanish Brunch                           Alfajores
                        0.0001049208                        0.0078690589
                  Alfajores Alfajores          Alfajores Alfajores Bread
                        0.0002098416                        0.0001049208
```

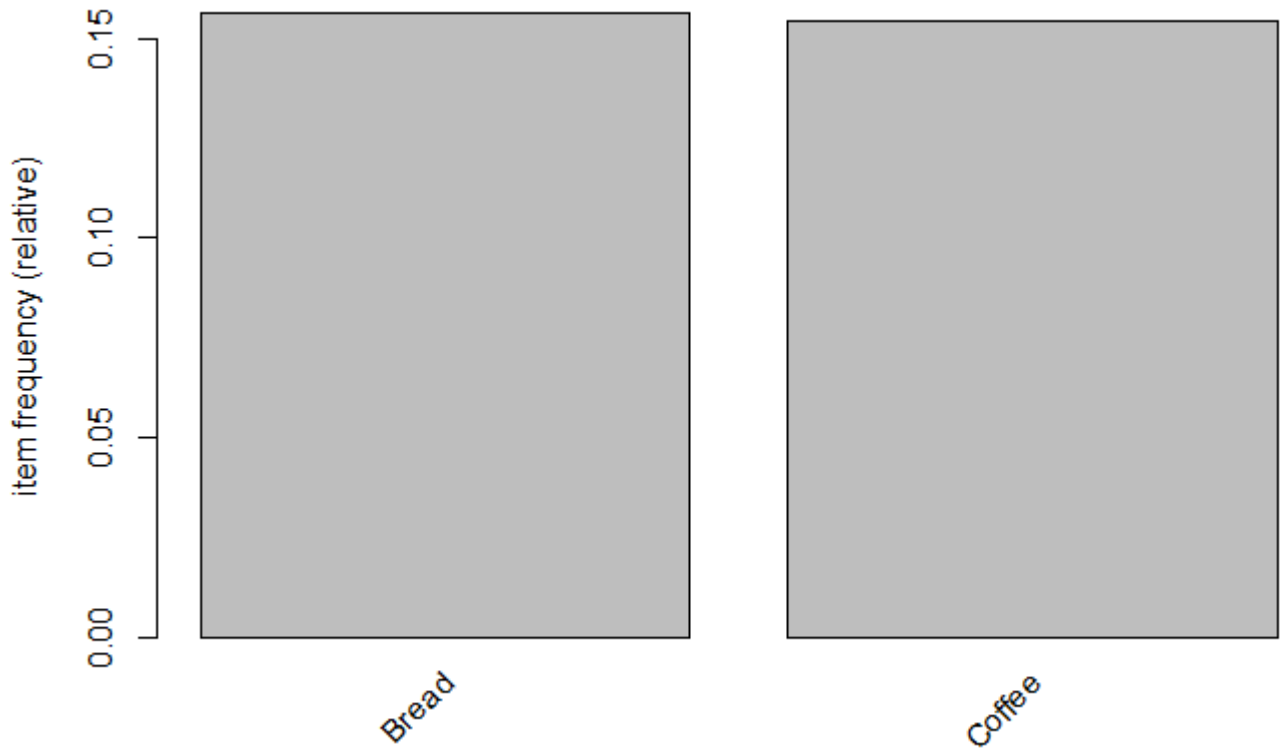
Hide

```
itemFrequencyPlot(Baky, support = .1) #graphs the items that show up in 20+% of the transaction
s.
```



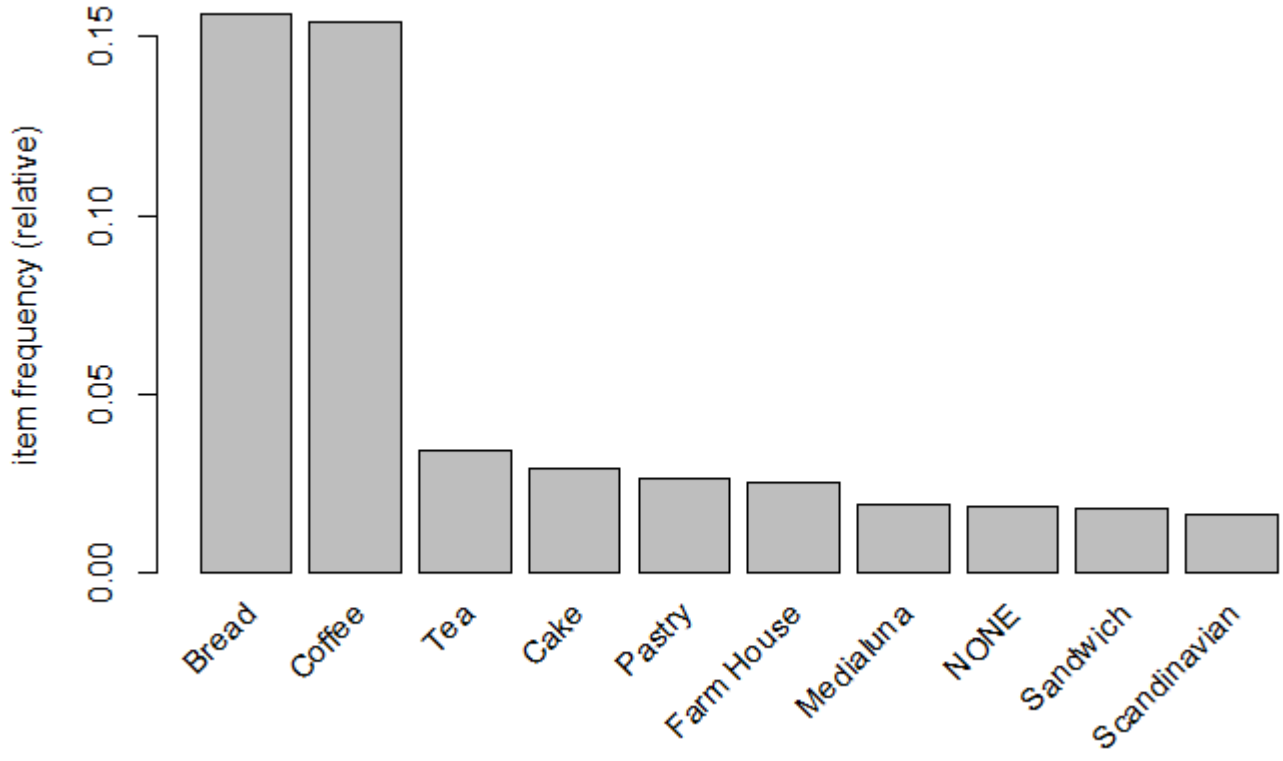
Hide

```
itemFrequencyPlot(Baky, support = .05) #graphs the items that show up in 37+% of the transaction
s.
```



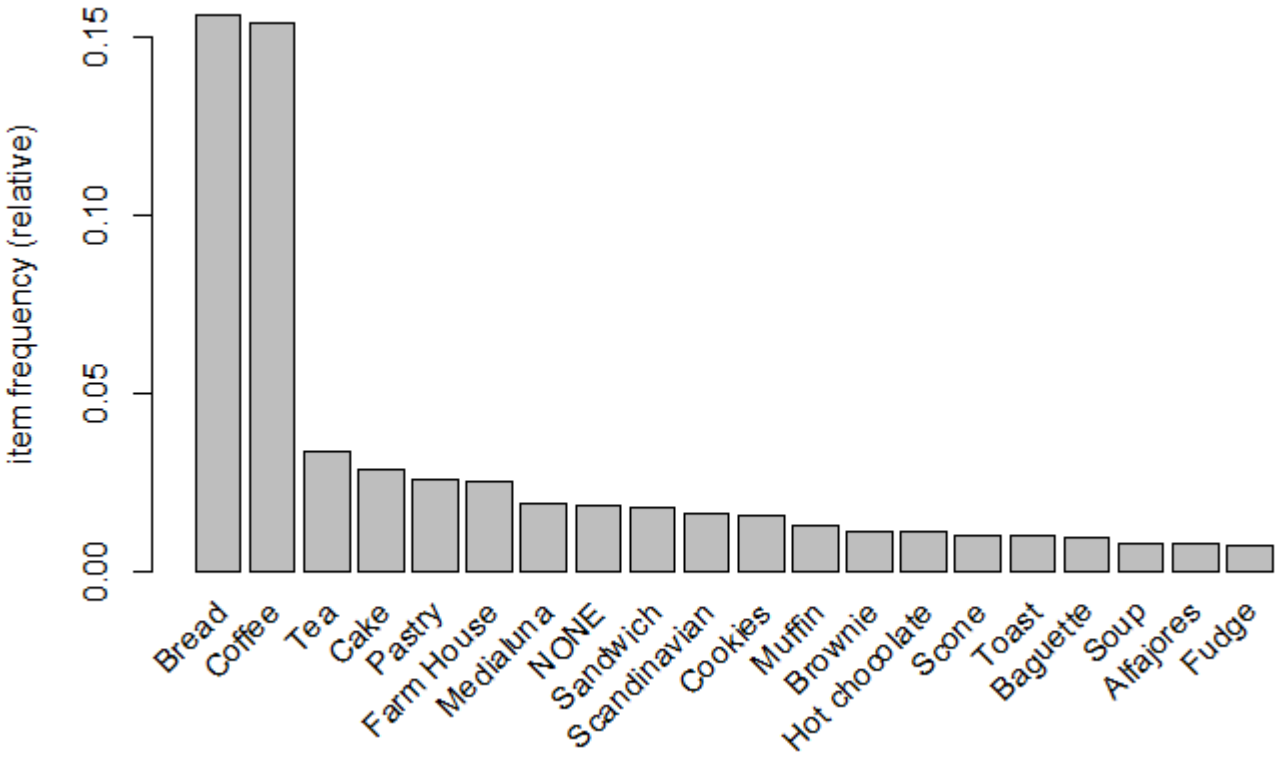
Hide

```
itemFrequencyPlot(Baky, topN = 10) #orders the data from greatest to least "support".
```



Hide

```
itemFrequencyPlot(Baky, topN = 20) #orders the data from greatest to least "support".
```



Hide

```
apm1 <- apriori(Baky, parameter = list(support=0.003,confidence = 0.04, minlen = 2))
```

```
Apriori

Parameter specification:
```

confidence <dbl>	minval <dbl>	s… <dbl>	ar… <fctr>	aval <lgl>	originalSupport <lgl>	maxtime <dbl>	support <dbl>	minlen <int>
0.04	0.1	1	none	FALSE	TRUE	5	0.003	2

1 row | 1-10 of 12 columns

```
Algorithmic control:
```

filter <dbl>	tree <lgl>	heap <lgl>	memopt <lgl>	load <lgl>	sort <int>	verbose <lgl>
0.1	TRUE	TRUE	FALSE	TRUE	2	TRUE

1 row

```
Absolute minimum support count: 28

set item appearances ...[0 item(s)] done [0.01s].
set transactions ...[1932 item(s), 9531 transaction(s)] done [0.02s].
sorting and recoding items ... [33 item(s)] done [0.01s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 done [0.00s].
writing ... [0 rule(s)] done [0.01s].
creating S4 object  ... done [0.01s].
```

Hide

```
summary(apm1)
```

```
set of 0 rules
```

Hide

```
inspect(sort(apm1 , by ="lift"))
```