Hm2

BenLarson

March 20, 2016

Homework 2

```
# script.dir <- dirname(sys.frame(1)$ofile)
require(knitr)
```

```
## Loading required package: knitr
```

```
opts_knit$set(root.dir= '.')
```

```
## [1] "/Users/Benjamin/Documents/PROBMOD/Hwk3"
```

```
## [1] 504
```

```
## [1] 32768
```

Bayesian networks were created by using the paths in the RiskFactorData. Each net contained the node and its parents. If a node had a parent it was added to the bayenet that is used throughout the homework. A node was created by using the createCPT.fromData function using the data and a list of it's parent nodes. This gives us probaility tables that we can now do analysis on. I found there were 504 probabilities needed for the table. This was found by summing the rows (probs column) of each node and adding them the nodes totals together. There were 32,768 probabilites in the joint distribution. The joint distribution was found by using the productFactor function to multiply every node and it's possible columns together.

probs	exercise	diabetes
0.1555074	2	1
0.0087785	2	2
0.8159050	2	3
0.0198090	2	4

probs	stroke	exercise	smoke
0.050901	1	2	1
0.949099	2	2	1

probs	attack	exercise	smoke
0.0755818	1	2	1
0.9244182	2	2	1

probs	angina	exercise	smoke

probs	angina	exercise	smoke
0.080087	1	2	1
0.919913	2	2	1

Above tables are for bad habits and the resulting probabilities for diabetes, stroke, heart attack, and agnina.

```r
kable(infer(diabetesnet,c('bmi','income'), c('smoke','exercise'), c(2,1)) )
```

probs	exercise	diabetes
0.1324724	1	1
0.0086683	1	2
0.8410585	1	3
0.0178008	1	4

```r
kable(infer(strokenet, c('bmi','income','cholesterol','bp'),c('smoke','exercise'), c(2,1)))
```

probs	stroke	exercise	smoke
0.0371955	1	1	2
0.9628045	2	1	2

```r
kable(infer(heartnet , c('bmi','income','cholesterol','bp'),c('smoke','exercise'), c(2,1)))
```

probs	attack	exercise	smoke
0.0533643	1	1	2
0.9466357	2	1	2

```r
kable(infer(anginanet, c('bmi','income','cholesterol','bp'),c('smoke','exercise'), c(2,1)))
```

probs	angina	exercise	smoke
0.0538606	1	1	2
0.9461394	2	1	2

Above tables are for good habits and the resulting probabilities for diabetes, stroke, heart attack, and agnina. Note please ignore the double table for angina… I'm unable to stop this.

```r
kable(infer(diabetesnet,c('smoke','income','exercise'), c('bmi','bp','cholesterol'), c(3,1,1)))
```

probs	bmi	diabetes
0.1198953	3	1

probs	bmi	diabetes
0.0074724	3	2
0.8549105	3	3
0.0177218	3	4

```r
kable(infer(strokenet, c('smoke','income','exercise'), c('bmi','bp','cholesterol'),  c(3,1,1)))
```

probs	bmi	bp	cholesterol	stroke
0.0837514	3	1	1	1
0.9162486	3	1	1	2

```r
kable(infer(heartnet,  c('smoke','income','exercise'), c('bmi','bp','cholesterol'),  c(3,1,1)))
```

probs	bmi	bp	cholesterol	attack
0.1398783	3	1	1	1
0.8601217	3	1	1	2

```r
kable(infer(anginanet, c('smoke','income','exercise'), c('bmi','bp','cholesterol'),  c(3,1,1)))
```

probs	cholesterol	bp	bmi	angina
0.1550232	1	1	3	1
0.8449768	1	1	3	2

Above tables are for poor health and the resulting probabilities for diabetes, stroke, heart attack, and agnina.

```r
kable(infer(diabetesnet,c('smoke','income','exercise'), c('bmi','bp','cholesterol'), c(2,3,2)))
```

probs	bmi	diabetes
0.0608828	2	1
0.0093110	2	2
0.9191494	2	3
0.0106567	2	4

```r
kable(infer(strokenet,  c('smoke','income','exercise'), c('bmi','bp','cholesterol'), c(2,3,2)))
```

probs	bmi	bp	cholesterol	stroke
0.0137527	2	3	2	1
0.9862473	2	3	2	2

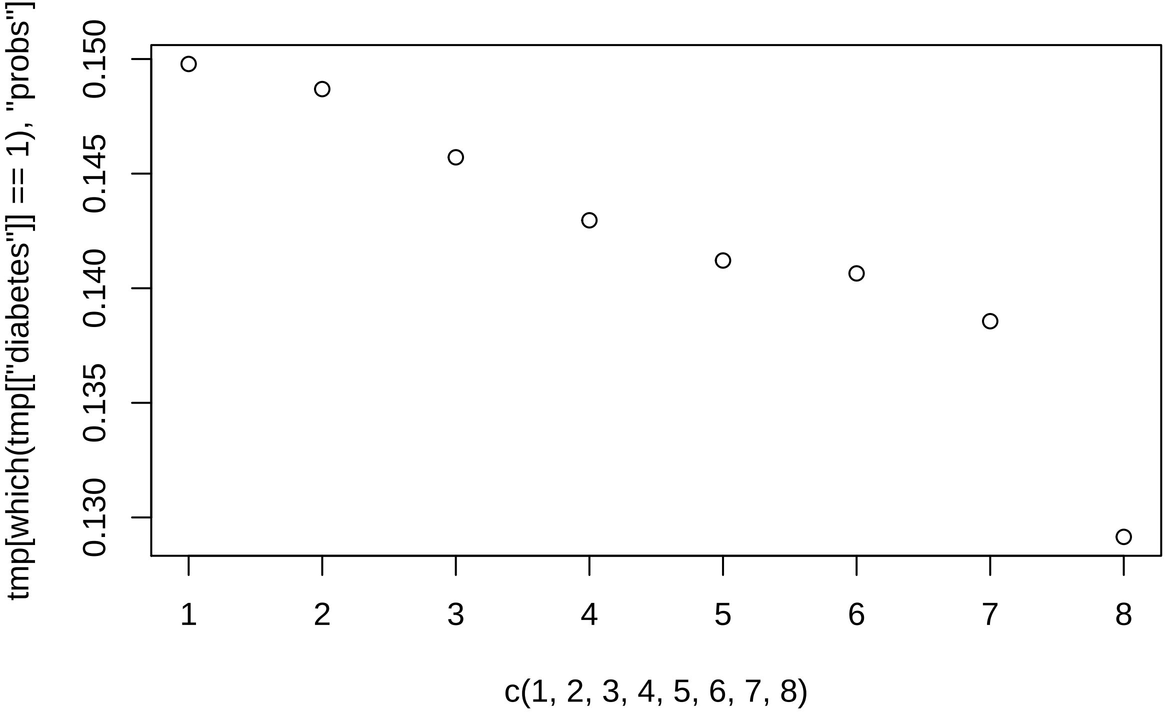
```
kable(infer(heartnet,  c('smoke','income','exercise'), c('bmi','bp','cholesterol'),  c(2,3,2)))
```

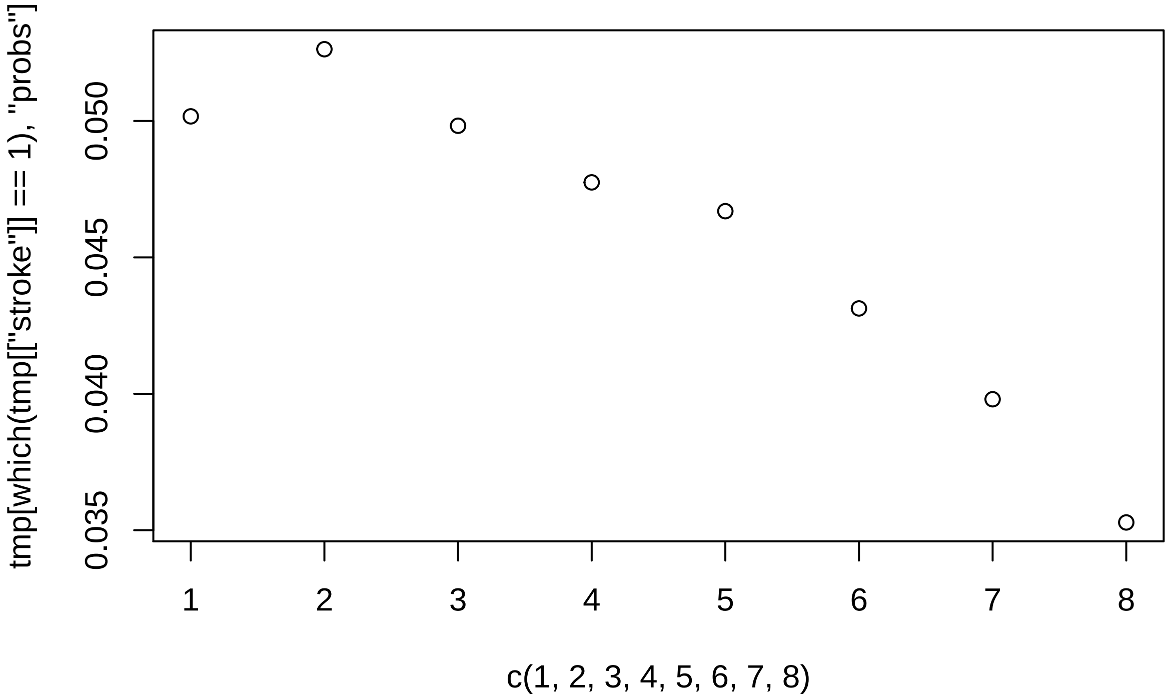
probs	bmi	bp	cholesterol	attack
0.0153899	2	3	2	1
0.9846101	2	3	2	2

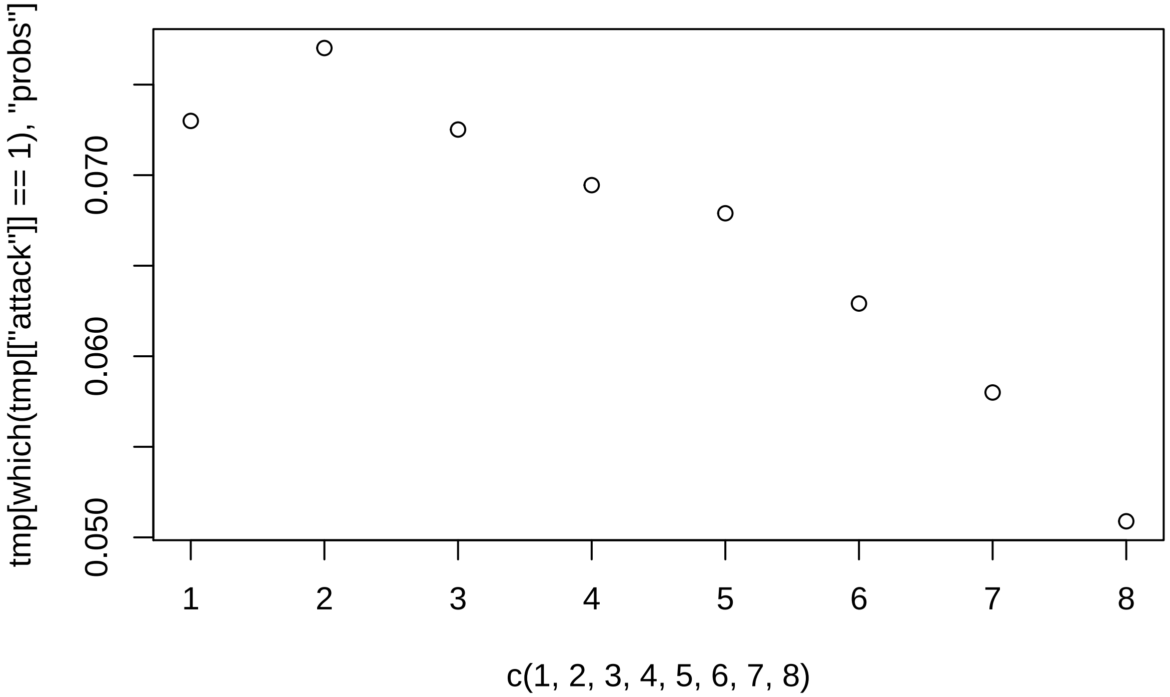
```
kable(infer(anginanet, c('smoke','income','exercise'), c('bmi','bp','cholesterol'),  c(2,3,2)))
```

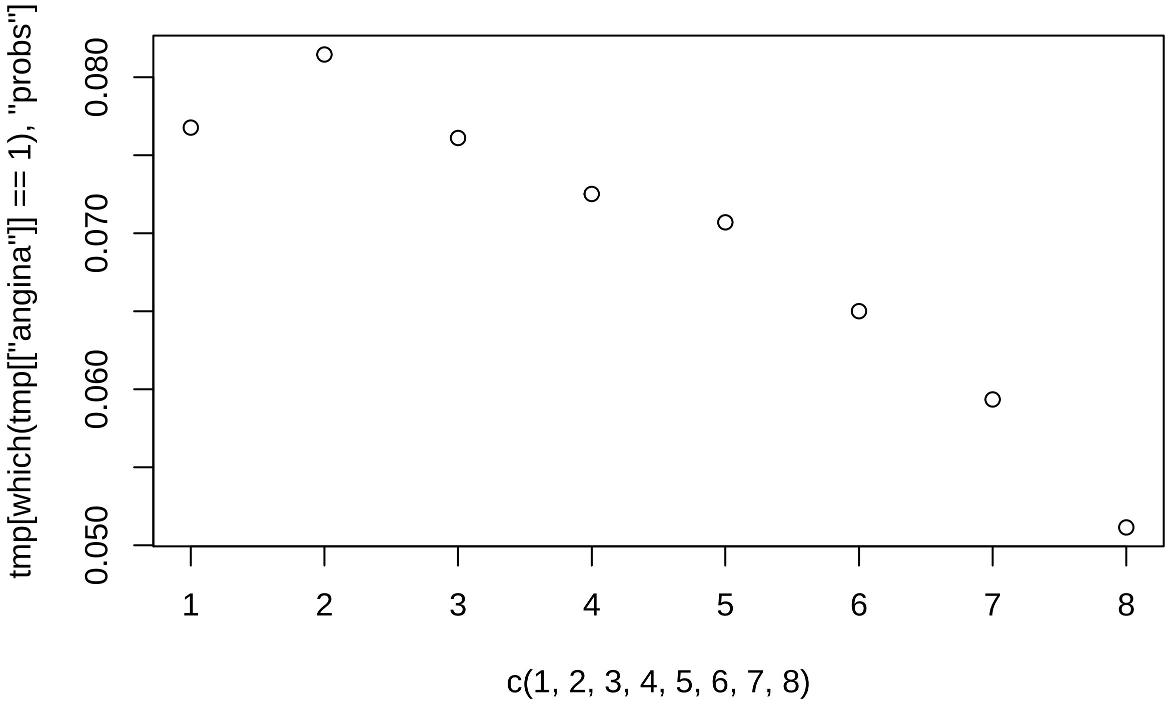
probs	cholesterol	bp	bmi	angina
0.0128474	2	3	2	1
0.9871526	2	3	2	2

Above tables are for good health and the resulting probabilities for diabetes, stroke, heart attack, and agnina.




tmp[which(tmp[["stroke"]] == 1), "probs"]
0.035
0.040
0.045
0.050
1
2
3
4
5
6
7
8
c(1, 2, 3, 4, 5, 6, 7, 8)


tmp[which(tmp[["attack"]] == 1), "probs"]
0.050
0.060
0.070
1
2
3
4
5
6
7
8
c(1, 2, 3, 4, 5, 6, 7, 8)



The above plots are evaluate the probability of someones income on the probability they have one of the four health outcomes. This is found by evaluating the joint probability of the respective nets, and then marginalizing out the factors that aren't income and then plotting. It is overwhelmingly obvious that the higher your income the more likely you are to avoid the health problems we are considering. Suprisingly the plot isn't linear. The lowest income, 1, has a slightly less probability that than 2. The assumption that the missing links between exercise and smoking and the health outcomes is that smoking and exercise conditionally change other nodes. Considering the path from the health outcomes to smoking and exercise we see that they are not conditionally independent. If the parent to a health outcome is observed who's parent is smoking or exercise then the health outcomes are conditionally independent of smoking or exercise. So informally it is saying that smoking and exercise aren't direct factors in health problems.

probs	smoke	exercise	diabetes
0.2191751	1	2	1
0.0066632	1	2	2
0.7511762	1	2	3
0.0229855	1	2	4

probs	smoke	exercise	stroke
0.0801954	1	2	1
0.9198046	1	2	2

probs	smoke	exercise	attack
0.1205514	1	2	1
0.8794486	1	2	2

probs	smoke	exercise	angina
0.1170839	1	2	1
0.8829161	1	2	2

Above tables with bad habits. Directly linking smoking and exercise to the health factors.

probs	smoke	exercise	diabetes
0.1004388	2	1	1
0.0093148	2	1	2
0.8750927	2	1	3
0.0151536	2	1	4

probs	smoke	exercise	stroke
0.0255604	2	1	1
0.9744396	2	1	2

probs	smoke	exercise	attack
0.0318827	2	1	1
0.9681173	2	1	2

probs	smoke	exercise	angina
0.0357139	2	1	1
0.9642861	2	1	2

Above tables with good habits. Directly linking smoking and exercise to the health factors.

probs	bmi	diabetes
0.1196741	3	1
0.0075058	3	2
0.8550904	3	3
0.0177296	3	4

probs	bmi	bp	cholesterol	stroke
0.0854038	3	1	1	1
0.9145962	3	1	1	2

probs	bmi	bp	cholesterol	attack
0.1415756	3	1	1	1
0.8584244	3	1	1	2

probs	cholesterol	bp	bmi	angina
0.1567437	1	1	3	1
0.8432563	1	1	3	2

Above tables with bad health. Directly linking smoking and exercise to the health factors.

probs	bmi	diabetes
0.0613541	2	1
0.0093339	2	2
0.9186096	2	3
0.0107024	2	4

probs	bmi	bp	cholesterol	stroke
0.0132257	2	3	2	1
0.9867743	2	3	2	2

probs	bmi	bp	cholesterol	attack
0.0148278	2	3	2	1
0.9851722	2	3	2	2

probs	cholesterol	bp	bmi	angina
0.0125751	2	3	2	1
0.9874249	2	3	2	2

Above tables with good health. Directly linking smoking and exercise to the health factors.

Looking at the tables and comparing the probability of not having health issues increases with good habits and good health compared to bad habits and poor health. Overall we see that adding the link of exercise and smoking increases the probability of not having health problems with good habits and good health. This intuitively makes sense. This suggests that the links do have a direct correlation with habits and health with health outcomes. The assumptions in the first graph were still valid, but not as accurate. They were created in a way that smoking and exercise only had an effect on bmi, bp, and cholesterol. There are possibly numerous other factors that smoking and exercise could effect that effects the health outcomes.

The lack of edges between health outcomes is indicating that the outcomes are independent of each other. While intuitively we know that diabetes can effect your probability of having many other health issues.

probs	stroke
0.0423948	1
0.9576052	2

probs	diabetes	stroke
0.0455432	1	1
0.9544568	1	2

probs	stroke
0.0423948	1
0.9576052	2

probs	diabetes	stroke
0.0418514	3	1
0.9581486	3	2

We see that having diabetes doesn't effect our first and third table. This makes sense because there is no path between them. We see that having a stroke is not depedent on diabetes. The interesting tables are the second and fourth. In these tables we see that having diabetes does, in fact, increase our risk for a stroke. This is a very small difference, but it did affect our results.

```
#####################################
## Simple chain example: x -> y -> z
#####################################
x = createCPT(list("x"), probs = c(0.3, 0.7), levelsList = list(c("T", "F")))
y.x = createCPT(list("y", "x"), probs = c(0.8, 0.4, 0.2, 0.6),
                levelsList = list(c("T", "F"), c("T", "F")))
z.y = createCPT(list("z", "y"), probs = c(0.5, 0.6, 0.5, 0.4),
                levelsList = list(c("T", "F"), c("T", "F")))

(xyzNet = list("x" = x, "y" = y.x, "z" = z.y))
```

```
## $x
##   probs x
## 1   0.3 T
## 2   0.7 F
##
## $y
##   probs y x
## 1   0.8 T T
## 2   0.4 T F
## 3   0.2 F T
## 4   0.6 F F
##
## $z
##   probs z y
## 1   0.5 T T
## 2   0.6 T F
## 3   0.5 F T
## 4   0.4 F F
```

```
# Some simple operations you might try to check your code
productFactor(x, y.x)
```

```
##   probs x y
## 1  0.28 F T
## 2  0.42 F F
## 3  0.24 T T
## 4  0.06 T F
```

```
productFactor(productFactor(x, y.x), z.y)
```

```
##   probs y x z
## 1 0.252 F F T
## 2 0.168 F F F
## 3 0.036 F T T
## 4 0.024 F T F
## 5 0.140 T F T
## 6 0.140 T F F
## 7 0.120 T T T
## 8 0.120 T T F
```

```
marginalizeFactor(productFactor(x, y.x), "x")
```

```
##   probs y
## 1  0.48 F
## 2  0.52 T
```

```
marginalizeFactor(productFactor(y.x, z.y), "z")
```

```
##   probs y
## 1     F F
## 2     F T
## 3     T F
## 4     T T
```

```
# Notice in the observe function, you just need to delete rows that are
# inconsistent with the given observations. Factors do not need to be combined
# or normalized in this step.
observe(xyzNet, "x", "T")
```

```
## $x
##   probs x
## 1   0.3 T
##
## $y
##   probs y x
## 1   0.8 T T
## 3   0.2 F T
##
## $z
##   probs z y
## 1   0.5 T T
## 2   0.6 T F
## 3   0.5 F T
## 4   0.4 F F
```

```
observe(xyzNet, c("x", "y"), c("T", "T"))
```

```
## $x
##   probs x
## 1   0.3 T
##
## $y
##   probs y x
## 1   0.8 T T
##
## $z
##   probs z y
## 1   0.5 T T
## 3   0.5 F T
```

```
# Marginalize must first combine all factors involving the variable to
# marginalize. Again, this operation may lead to factors that aren't
# probabilities.
marginalize(xyzNet, "x")
```

```
## $x
##   probs
## 1     1
##
## $y
##   probs y
## 1   0.8 F
## 2   1.2 T
##
## $z
##   probs z y
## 1   0.4 F F
## 2   0.6 T F
## 3   0.5 F T
## 4   0.5 T T
```

```
marginalize(xyzNet, "y")
```

```
## $x
##   probs
## 1     F
## 2     T
##
## $y
##   probs
## 1     F
## 2     T
##
## $z
##   probs z
## 1   0.9 F
## 2   1.1 T
```

```
marginalize(xyzNet, "z")
```

```
## $x
##   probs
## 1     F
## 2     T
## 
## $y
##   probs y
## 1     F F
## 2     F T
## 3     T F
## 4     T T
## 
## $z
##   probs y
## 1     1 F
## 2     1 T
```

```
marginalize(xyzNet, c("x", "z"))
```

```
## $x
##   probs
## 1     1
## 
## $y
##   probs y
## 1   0.8 F
## 2   1.2 T
## 
## $z
##   probs y
## 1     1 F
## 2     1 T
```

```
##############################
## Bishop book (Ch 8) example
##############################
b = createCPT(list("battery"), probs = c(0.9, 0.1), levelsList = list(c(1, 0)))
f = createCPT(list("fuel"), probs = c(0.9, 0.1), levelsList = list(c(1, 0)))
g.bf = createCPT(list("gauge", "battery", "fuel"),
                 probs = c(0.8, 0.2, 0.2, 0.1, 0.2, 0.8, 0.8, 0.9),
                 levelsList = list(c(1, 0), c(1, 0), c(1, 0)))

carNet = list("battery" = b, "fuel" = f, "gauge" = g.bf)

## Some examples:
## Notice that different order of operations give the same answer
## (rows/columns may be permuted)
productFactor(productFactor(b, f), g.bf)
```

```
##   probs battery fuel gauge
## 1 0.001       0    0     1
## 2 0.009       0    0     0
## 3 0.018       0    1     1
## 4 0.072       0    1     0
## 5 0.018       1    0     1
## 6 0.072       1    0     0
## 7 0.648       1    1     1
## 8 0.162       1    1     0
```

```r
productFactor(productFactor(g.bf, f), b)
```

```
##   probs battery fuel gauge
## 1 0.001       0    0     1
## 2 0.009       0    0     0
## 3 0.018       0    1     1
## 4 0.072       0    1     0
## 5 0.018       1    0     1
## 6 0.072       1    0     0
## 7 0.648       1    1     1
## 8 0.162       1    1     0
```

```r
marginalizeFactor(productFactor(g.bf, b), "gauge")
```

```
##   probs battery fuel
## 1   0.1       0    0
## 2   0.9       1    0
## 3   0.1       0    1
## 4   0.9       1    1
```

```r
productFactor(marginalizeFactor(g.bf, "gauge"), b)
```

```
##   probs battery fuel
## 1   0.1       0    0
## 2   0.1       0    1
## 3   0.9       1    0
## 4   0.9       1    1
```

```r
productFactor(marginalizeFactor(productFactor(g.bf, b), "battery"), f)
```

```
##   probs fuel gauge
## 1 0.081    0     0
## 2 0.019    0     1
## 3 0.234    1     0
## 4 0.666    1     1
```

```r
marginalizeFactor(productFactor(productFactor(g.bf, f), b), "battery")
```

```
##   probs fuel gauge
## 1 0.081    0     0
## 2 0.234    1     0
## 3 0.019    0     1
## 4 0.666    1     1
```

```r
marginalizeFactor(productFactor(marginalizeFactor(productFactor(g.bf, b), "battery"), f), "gauge")
```

```
##   probs fuel
## 1   0.1    0
## 2   0.9    1
```

```r
marginalizeFactor(productFactor(marginalizeFactor(productFactor(g.bf, b), "battery"), f), "fuel")
```

```
##   probs gauge
## 1 0.315     0
## 2 0.685     1
```

```r
# ## Examples computed in book (see pg. 377)
infer(carNet, c("battery", "fuel"), NULL, NULL)     ## (8.30)
```

```
##   probs gauge
## 1 0.315     0
## 2 0.685     1
```

```r
infer(carNet, c("battery"), "fuel", 0)              ## (8.31)
```

```
##   probs fuel gauge
## 1  0.81    0     0
## 2  0.19    0     1
```

```r
infer(carNet, c("battery"), "gauge", 0)             ## (8.32)
```

```
##       probs fuel gauge
## 1 0.2571429    0     0
## 2 0.7428571    1     0
```

```r
infer(carNet, NULL, c("gauge", "battery"), c(0, 0)) ## (8.33)
```

```
##       probs battery fuel gauge
## 1 0.1111111       0    0     0
## 2 0.8888889       0    1     0
```

```r
###########################################################################
## Kevin Murphy's Example: http://www.cs.ubc.ca/~murphyk/Bayes/bnintro.html
########################################################r###################
c = createCPT(list("cloudy"), probs = c(0.5, 0.5),
              levelsList = list(c("F", "T")))
r.c = createCPT(list("rain", "cloudy"), probs = c(0.8, 0.2, 0.2, 0.8),
                levelsList = list(c("F", "T"), c("F", "T")))
s.c = createCPT(c("sprinkler", "cloudy"), probs = c(0.5, 0.9, 0.5, 0.1),
                levelsList = list(c("F", "T"), c("F", "T")))
w.sr = createCPT(list("wet", "sprinkler", "rain"),
                 probs = c(1, 0.1, 0.1, 0.01, 0, 0.9, 0.9, 0.99),
                 levelsList = list(c("F", "T"), c("F", "T"), c("F", "T")))
```

```r
grassNet = list("cloudy" = c, "rain" = r.c, "sprinkler" = s.c, "wet" = w.sr)

## Test your infer() method by replicating the computations on the website!!


###########################################################################
## Class example
###########################################################################
traffic = createCPT(list("traffic"), probs = c(0.5, 0.5), levelsList = list(c("F", "T")))
alarm = createCPT(list("alarm"), probs = c(0.5, 0.5), levelsList = list(c("F", "T")))
late = createCPT(list("late", "alarm", "traffic"),
                 probs = c(0.8, 0.2, 0.8, 0.6, 0.2, 0.8, 0.2, 0.4),
                 levelsList = list(c("F", "T"), c("F", "T"), c("F", "T")))

lateNet = list(traffic, alarm, late)
```