# Mushroom

## Bayesian Network

JUNE 13, 2017

KE DING

## 1. Introduction

In this case study, I'll be training a Bayesian Network to analyze a mushroom dataset. Mushroom records is drawn from The Audubon Society Field Guide to North American Mushrooms (1981). G. H. Lincoff (Pres.), New York: Alfred A. Knopf.

This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family (pp. 500-525). Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one. The Guide clearly states that there is no simple rule for determining the edibility of a mushroom; no rule like ``leaflets three, let it be'' for Poisonous Oak and Ivy.

Bayesian network can capture the joint probabilities by using conditional probabilities formula, which is saying, it works great for probabilistic queries when some variables are dependent to one another or independent in condition. Bayesian network is common applied to discrete variables.

Section 1 is a brief introduction of the case study. Section 2 is the description of the dataset and the feature explanation in this dataset. Section 3 is the data cleaning process. In section 4, some exploratory analysis is conducted to get a rough sense of the dataset. Experimental process and results, conclusions and insights generated from the experimental results are presented in section 5. And section 6 provides the limitations and conclusion of this case study. Appendix is the coding part.

## 2. Data Description

### 2.1 Dataset

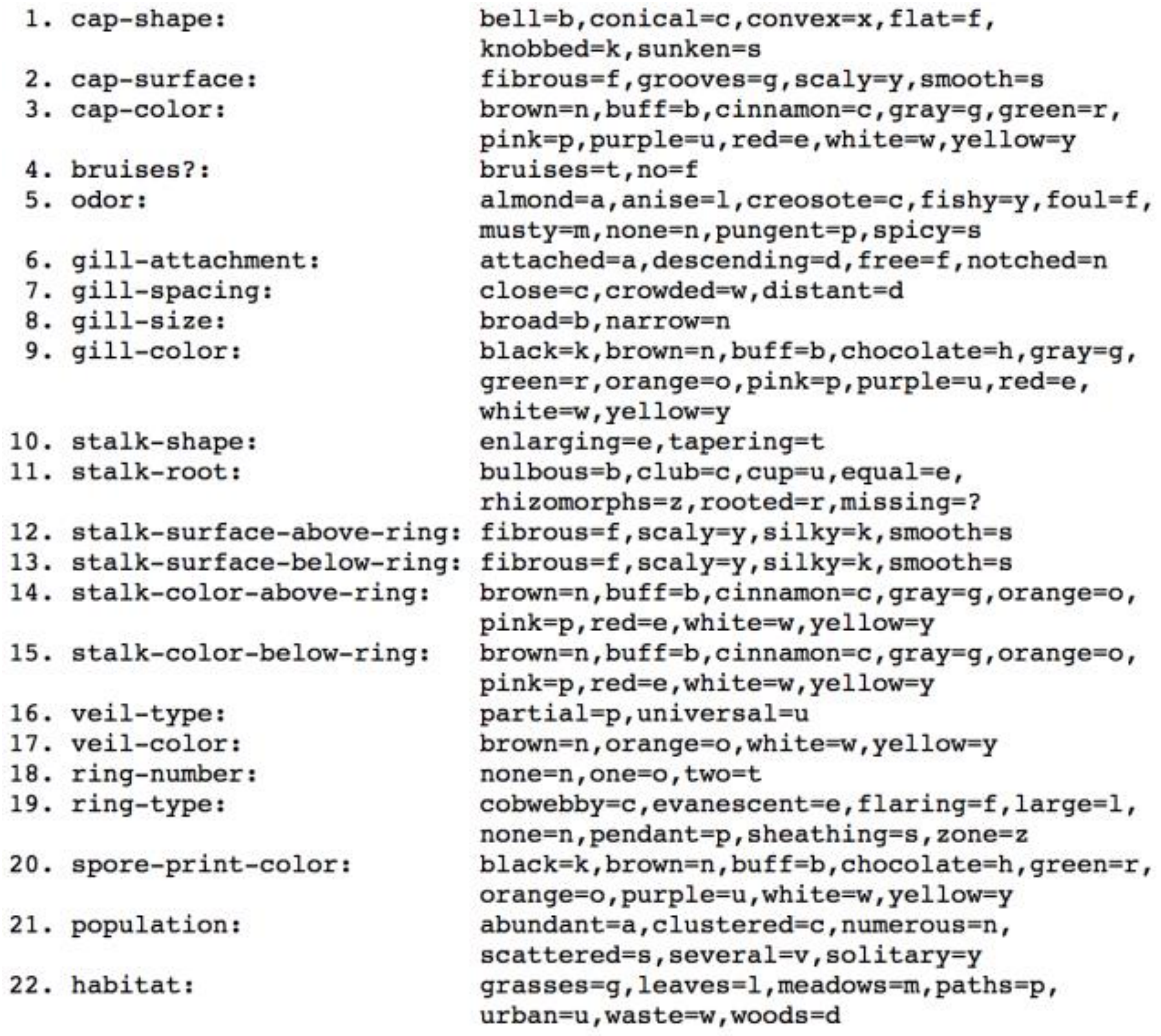
The dataset I'm using is a mushroom classification dataset which is drawn from The Audubon Society Field Guide to North American Mushrooms (1981). G. H. Lincoff (Pres.), New York: Alfred A. Knopf. The link of this dataset is *https://archive.ics.uci.edu/ml/datasets/mushroom*.

The dataset has 8124 observations, and 22 attributes.

Detailed information of the data attributes is presented later.

## 2.2 Attributes

There are 22 attributes and 1 class variable.



| | |
|---|---|
| 1. cap-shape: | bell=b,conical=c,convex=x,flat=f, knobbed=k,sunken=s |
| 2. cap-surface: | fibrous=f,grooves=g,scaly=y,smooth=s |
| 3. cap-color: | brown=n,buff=b,cinnamon=c,gray=g,green=r, pink=p,purple=u,red=e,white=w,yellow=y |
| 4. bruises?: | bruises=t,no=f |
| 5. odor: | almond=a,anise=l,creosote=c,fishy=y,foul=f, musty=m,none=n,pungent=p,spicy=s |
| 6. gill-attachment: | attached=a,descending=d,free=f,notched=n |
| 7. gill-spacing: | close=c,crowded=w,distant=d |
| 8. gill-size: | broad=b,narrow=n |
| 9. gill-color: | black=k,brown=n,buff=b,chocolate=h,gray=g, green=r,orange=o,pink=p,purple=u,red=e, white=w,yellow=y |
| 10. stalk-shape: | enlarging=e,tapering=t |
| 11. stalk-root: | bulbous=b,club=c,cup=u,equal=e, rhizomorphs=z,rooted=r,missing=? |
| 12. stalk-surface-above-ring: | fibrous=f,scaly=y,silky=k,smooth=s |
| 13. stalk-surface-below-ring: | fibrous=f,scaly=y,silky=k,smooth=s |
| 14. stalk-color-above-ring: | brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y |
| 15. stalk-color-below-ring: | brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y |
| 16. veil-type: | partial=p,universal=u |
| 17. veil-color: | brown=n,orange=o,white=w,yellow=y |
| 18. ring-number: | none=n,one=o,two=t |
| 19. ring-type: | cobwebby=c,evanescent=e,flaring=f,large=l, none=n,pendant=p,sheathing=s,zone=z |
| 20. spore-print-color: | black=k,brown=n,buff=b,chocolate=h,green=r, orange=o,purple=u,white=w,yellow=y |
| 21. population: | abundant=a,clustered=c,numerous=n, scattered=s,several=v,solitary=y |
| 22. habitat: | grasses=g,leaves=l,meadows=m,paths=p, urban=u,waste=w,woods=d |

### 2.3 Classification Labels

The dependent variable in this study is the class. It has 2 levels: edible=e, poisonous=p.

## 3. Data Cleaning

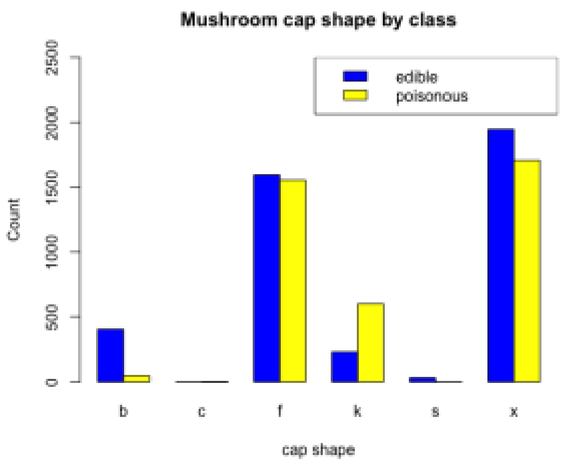
After importing the dataset, I found that all the variables do not have labels, so firstly, adding column names as showed in the dataset file. Then, I checked if there's any missing value and found out it's a complete dataset with no missing value. Although the variable stalk_root has a level called "?", which means missing and there're 2480 observations of that, so I just count this as one level. All the rest variables look tidy and clean.

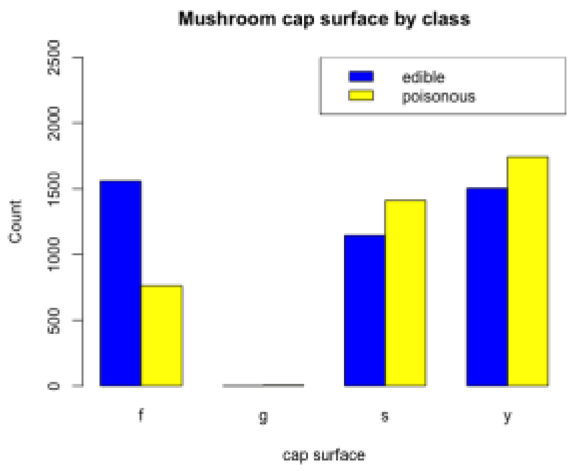
Then I checked the structure of the dataset and found that all the class variable is factor. Therefore, there's no need to convert any of them for building Bayesian network. Moreover, attribute 17, veil_type, has two level p and u but all the observations in this dataset are of level p, which means it only has one level here. Bayesian network only works with variables of at least two levels so I removed this variable.

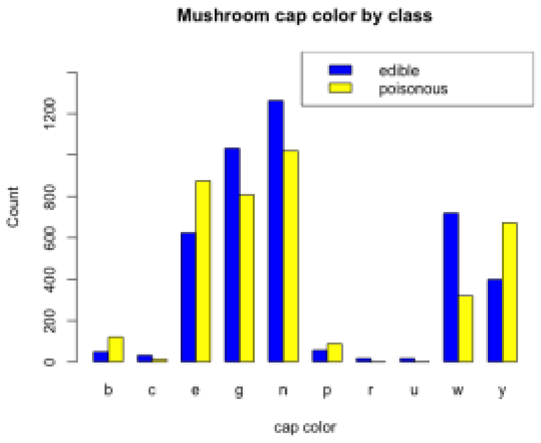
## 4. Data Analysis

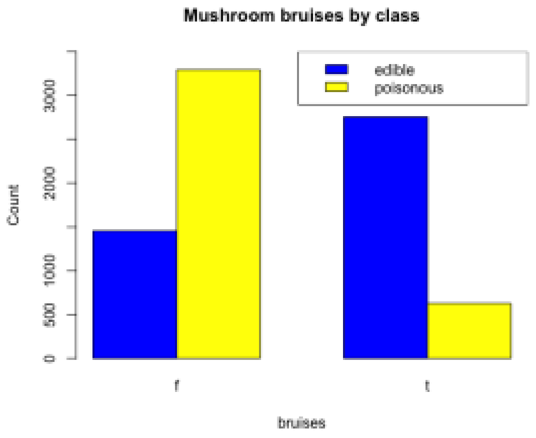
Now take a look at the structure of the dataset after cleaning. There're 8124 observation of 22 variables, all the variables are factor with more than 1 level. 4208 of all are edible mushroom, which counts as 51.8% of the data; the rest 3916 are poisonous mushroom, counting as 48.2%.

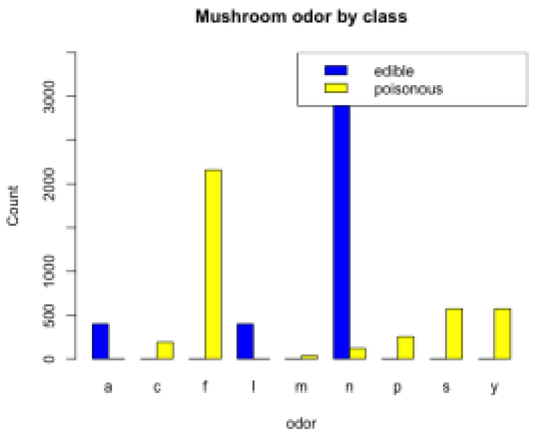
Since all the variables are discrete factors, I didn't do any correlation analysis on them, instead, I plotted some charts of each attribute grouped by class.

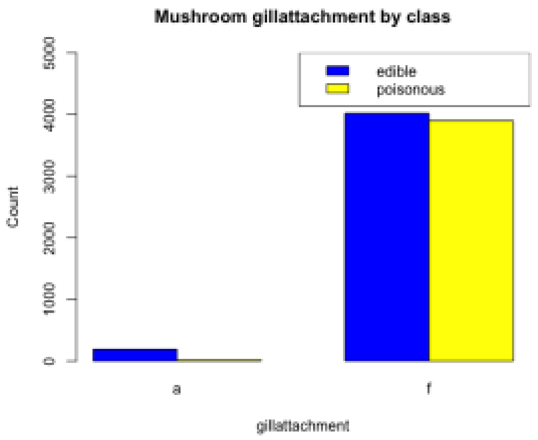

Mushroom cap shape by class
edible
poisonous
Count
0
500
1000
1500
2000
2500
b
c
f
k
s
x
cap shape

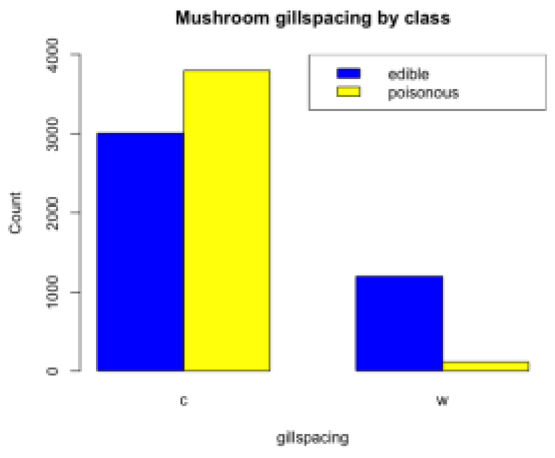

Mushroom cap surface by class
edible
poisonous
Count
0
500
1000
1500
2000
2500
f
g
s
y
cap surface

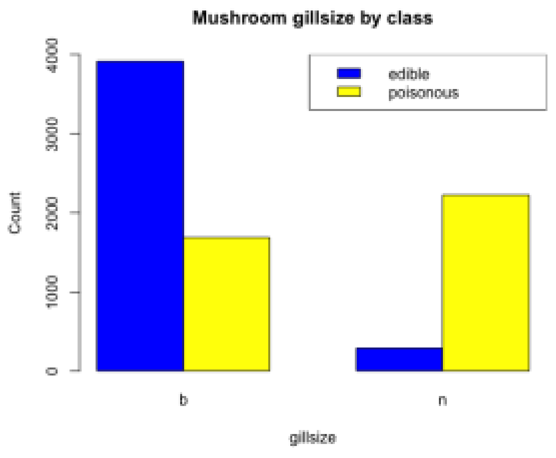

Mushroom cap color by class
edible
poisonous
Count
0
200
400
600
800
1200
b
c
e
g
n
p
r
u
w
y
cap color

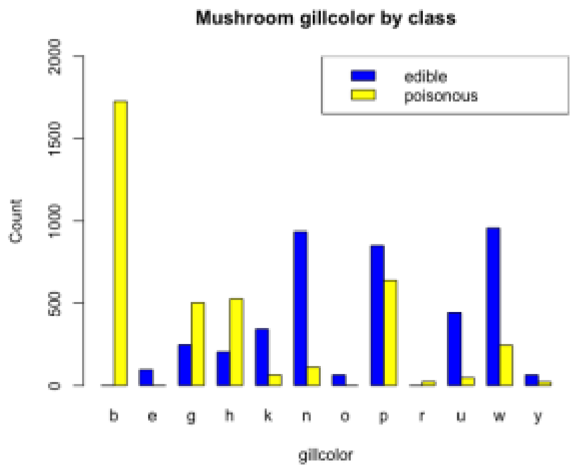

Mushroom bruises by class
edible
poisonous
Count
0
500
1000
2000
3000
f
t
bruises

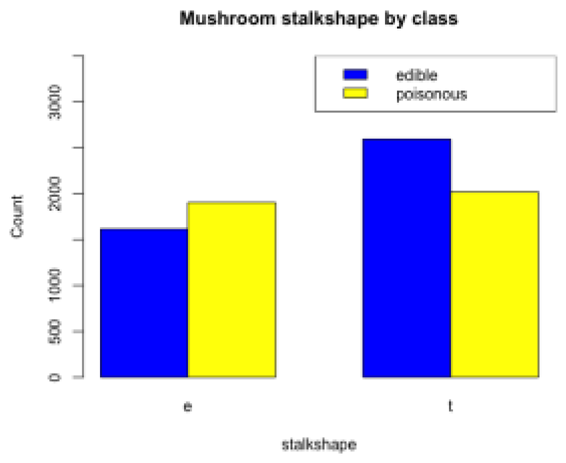

Mushroom odor by class
edible
poisonous
Count
0
500
1000
2000
3000
a
c
f
l
m
n
p
s
y
odor

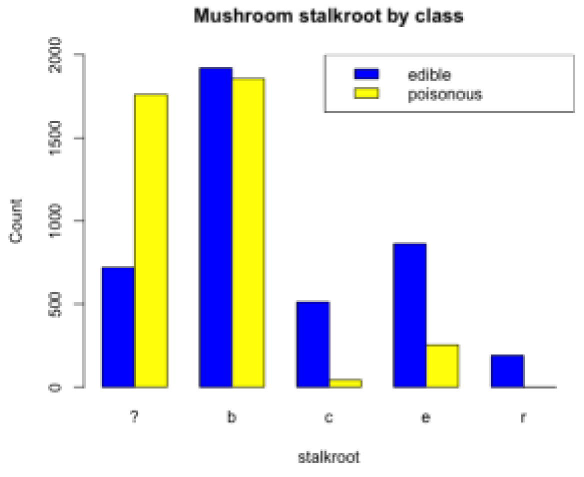

Mushroom gillattachment by class
edible
poisonous
Count
0
1000
2000
3000
4000
5000
a
f
gillattachment

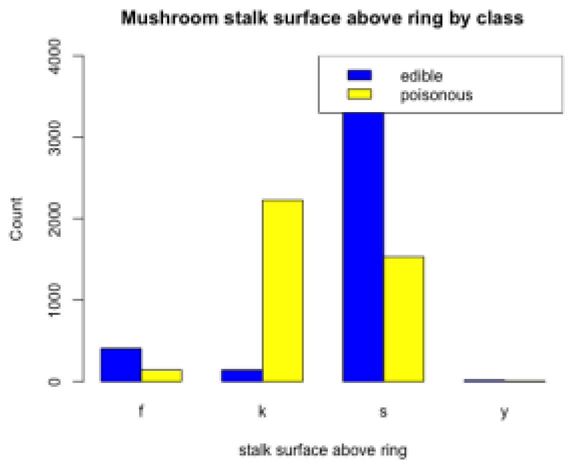

Mushroom gillspacing by class
edible
poisonous
Count
0
1000
2000
3000
4000
c
w
gillspacing

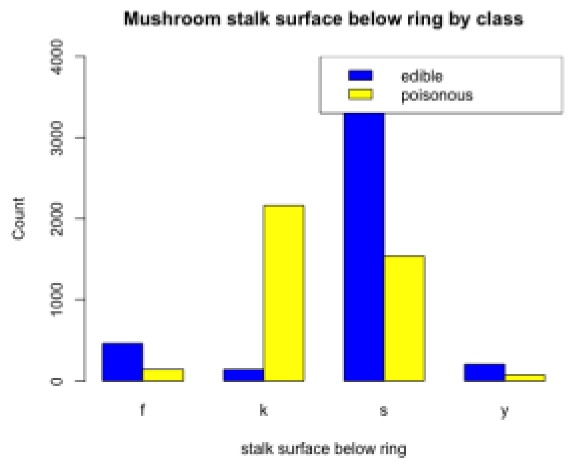

Mushroom gillsize by class
edible
poisonous
Count
0
1000
2000
3000
4000
b
n
gillsize

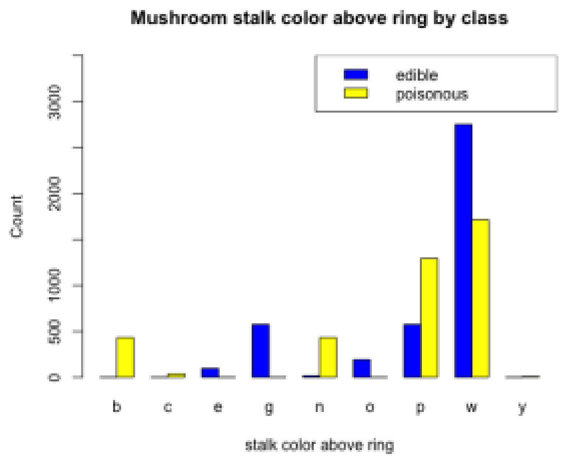

Mushroom gillcolor by class
edible
poisonous
Count
0
500
1000
1500
2000
b
e
g
h
k
n
o
p
r
u
w
y
gillcolor

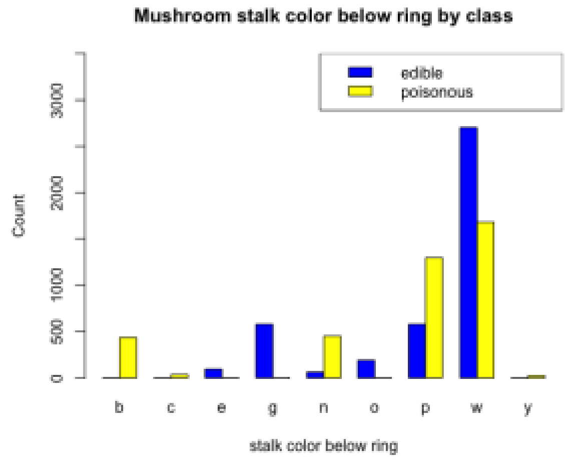

Mushroom stalkshape by class
edible
poisonous
Count
0
500
1000
2000
3000
e
t
stalkshape

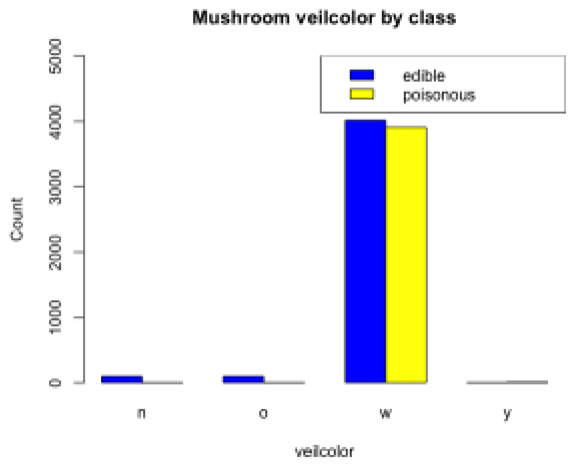

Mushroom stalkroot by class
edible
poisonous
Count
0
500
1000
1500
2000
?
b
c
e
r
stalkroot

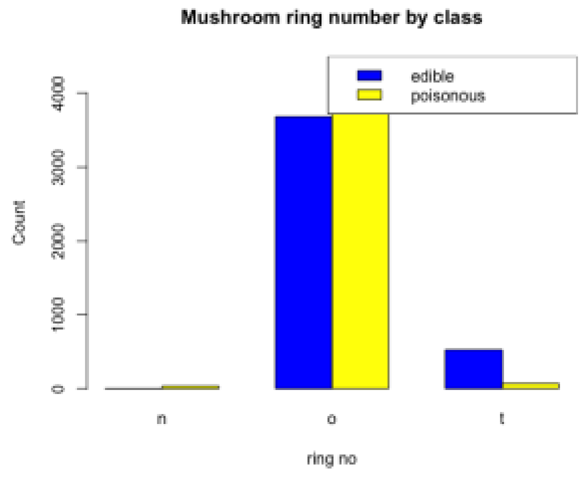

Mushroom stalk surface above ring by class
edible
poisonous
Count
0
1000
2000
3000
4000
f
k
s
y
stalk surface above ring

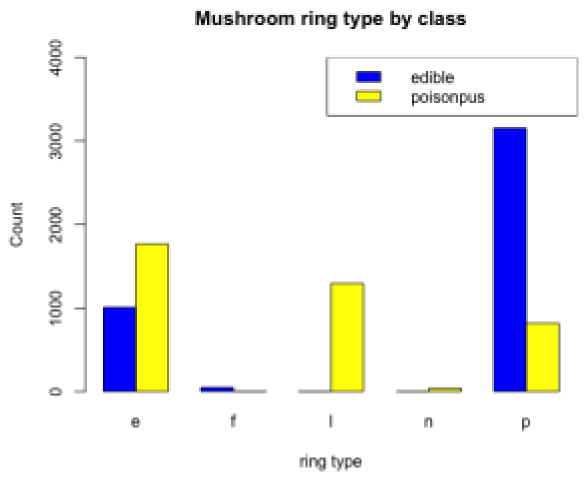

Mushroom stalk surface below ring by class
edible
poisonous
Count
0
1000
2000
3000
4000
f
k
s
y
stalk surface below ring

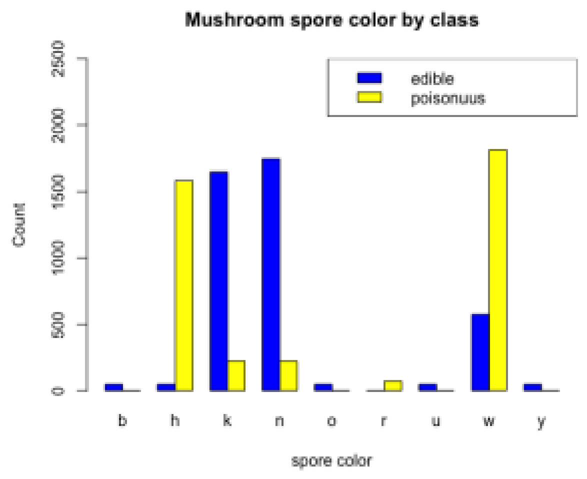

Mushroom stalk color above ring by class
edible
poisonous
Count
0
500
1000
2000
3000
b
c
e
g
n
o
p
w
y
stalk color above ring

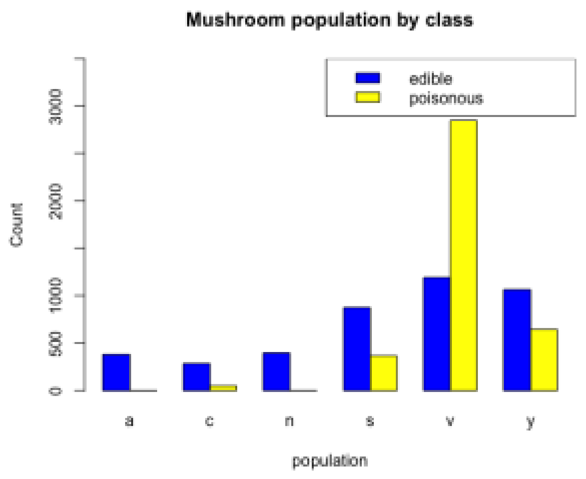

Mushroom stalk color below ring by class
edible
poisonous
Count
0
500
1000
2000
3000
b
c
e
g
n
o
p
w
y
stalk color below ring

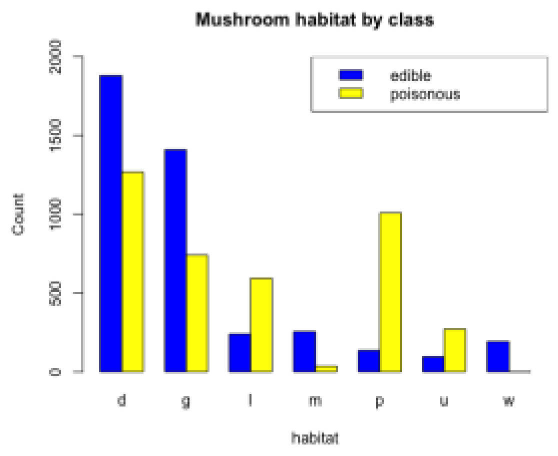

Mushroom veilcolor by class
edible
poisonous
Count
0
1000
2000
3000
4000
5000
n
o
w
y
veilcolor


Mushroom ring number by class
edible
poisonous
Count
0
1000
2000
3000
4000
n
o
t
ring no


Mushroom ring type by class
edible
poisonpus
Count
0
1000
2000
3000
4000
e
f
l
n
p
ring type







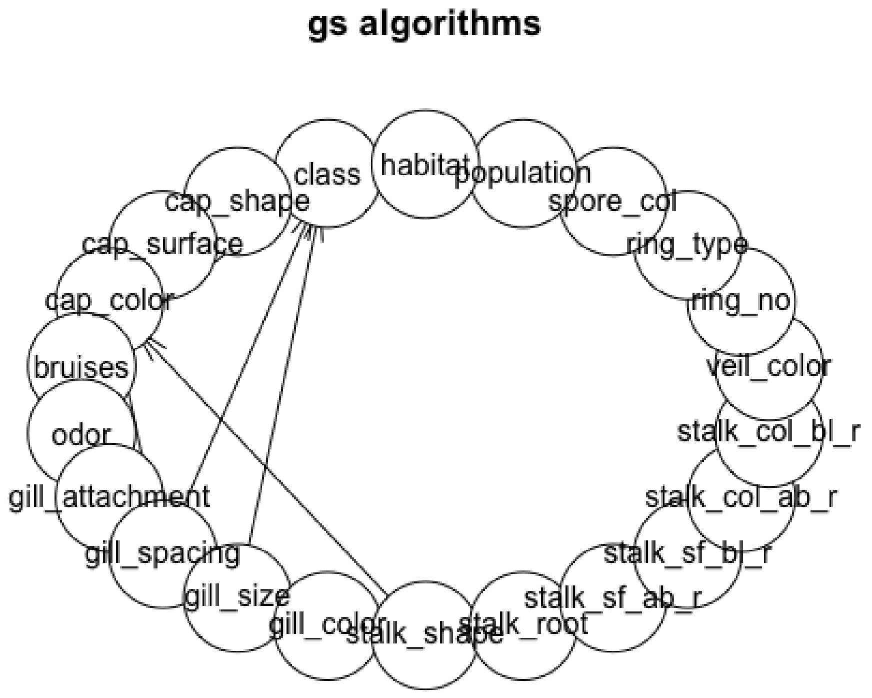
The above bar charts present the frequency of different level of each variable grouped by class. Roughly say, some variables such as cap shape, cap surface, gill attachment, stalk shape, veil color, ring number, and so on do not have a distinct separation in terms of edible or poisonous, why some other variables such as odor and spore color would be a better to differentiate edible from poisonous mushrooms.

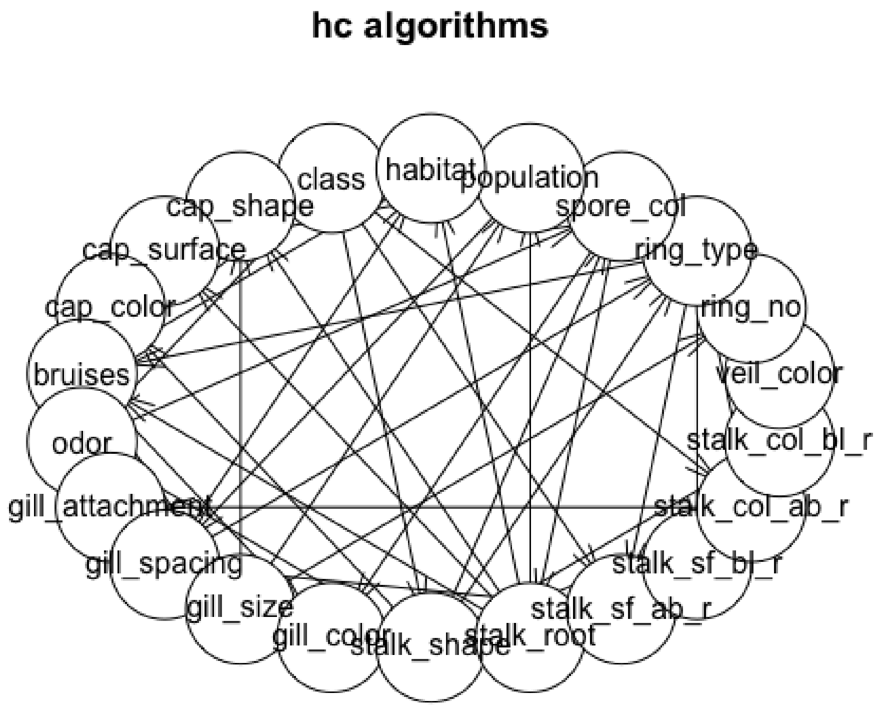
## 5. Experimental Results and Analysis

In this case study, I built several Bayesian network classifiers and random forest as comparison.

To evaluate the classifiers, I used 10-fold cross validations through caret.package in R to create confusion matrix, calculate accuracy, error rate, and so on.

First, I created two different Bayesian networks using Grow-Shrink and Hill-Climbing algorithm.



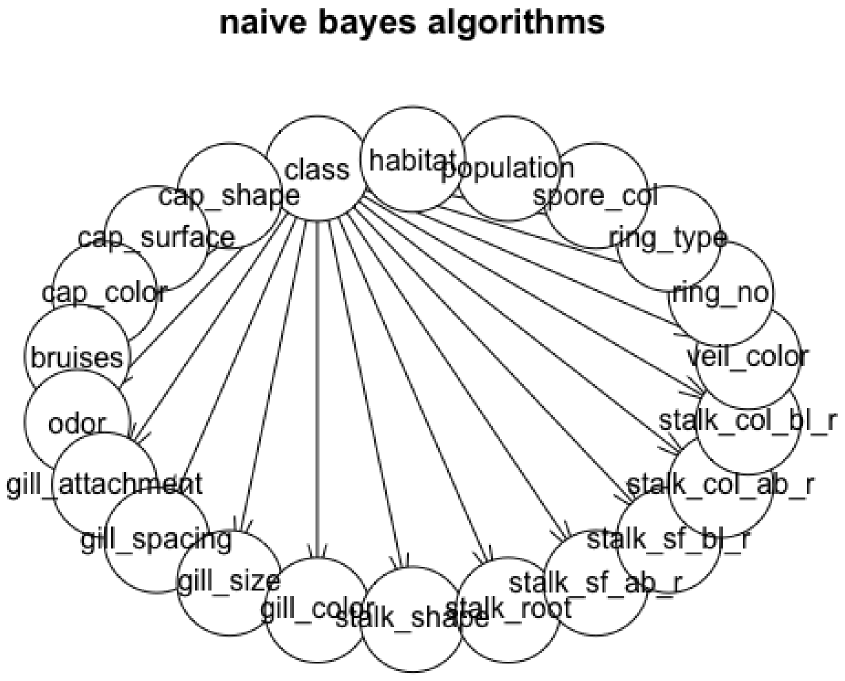


Grow-Shrink model only connects several of the variables while Hill-Climbing uses all the variables. The comparing result shows that TP is 2, FP is 48, and FN is 8.

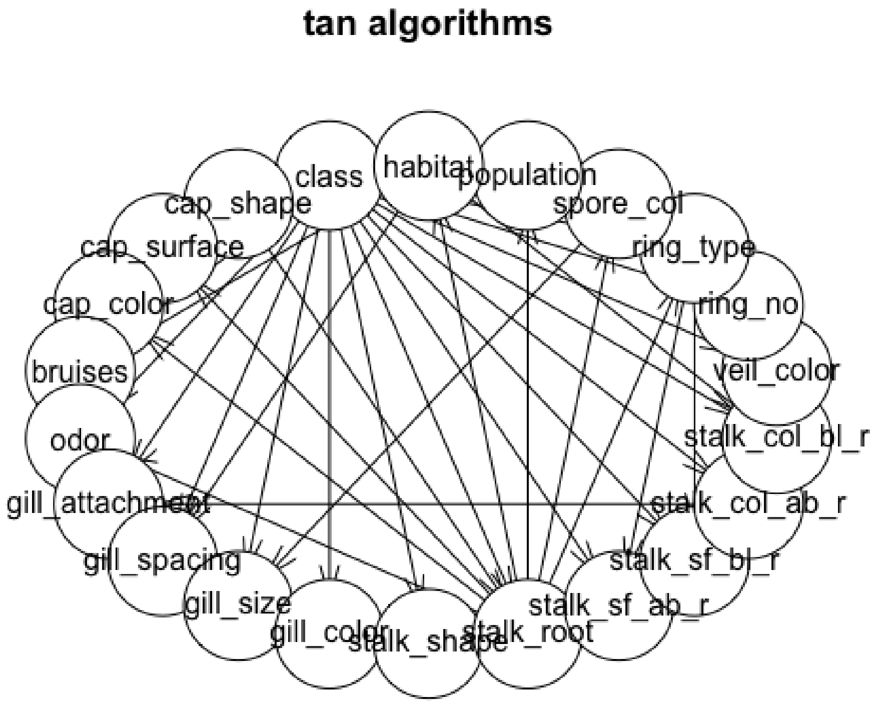
Then, I try to use 10-fold cross validation to fit parameters of the two Bayesian networks. However, due to the way Bayesian networks are defined, it is possible to estimate their parameters only if the network structure is completely directed. The GS algorithm is a partially directed network, so it cannot be handled by parameter fitting. I must automatically set arcs to make it work. But due to lack of domain knowledge, I decided not to change the network by myself.

After applying 10-fold cross validation to my hc network using bn.cv function, the expected loss was extremely big, 0.4820. Then I used bn.fit to fit the gs network and hc network respectively. The confusion matrix result also shows that hc network is not working at all, it classifies all the test data as edible, which leads to an accuracy of 0.5143. Hence, hill-climbing algorithm will not be an option for mushroom dataset.

After trying out constraint-based algorithm and score-based algorithm, I also applied naïve Bayes algorithm using 10-fold cross validation on the data, getting a cv accuracy of 92.04%.



Then, I tried to use Tree Augmented Naive Bayes classifier. However, as far as I can find, to use TAN with cross validation, a package called bnclassify must be installed, but unfortunately it’s already removed from the CRAN repository. But I still applied TAN classifier and calculate the confusion matrix using test dataset.



After trying out different Bayesian networks, I also built a decision tree to add some different perspective.

| | #node | # arcs | # undirected arcs | accuracy | 95% CI |
|---|---|---|---|---|---|
| gs | 22 | 10 | 0 | 94.09% | (0.9284, 0.9518) |
| hc | 22 | 50 | 0 | 51.43% | (0.4898, 0.5388) |
| NB | 22 | 21 | 0 | 92.99% | (0.8808, 0.9676) |
| TAN | 22 | 41 | 0 | 80.93% | (0.7894, 0.828) |
| Decision tree | # | # | # | 95.95% | (0.933, 0.9557) |

This table above shows some features of the classifiers I built. But the accuracy of gs and TAN is not calculated by 10-fold cross validation. The table below is the confusion matrices of the classifiers being predicted on test dataset.

| GS | | e | p |
|---|---|---|---|
| Predi -cted | e | 828 | 81 |
| | p | 16 | 716 |

| HC | | e | p |
|---|---|---|---|
| Predi- cted | e | 844 | 797 |
| | p | 0 | 0 |

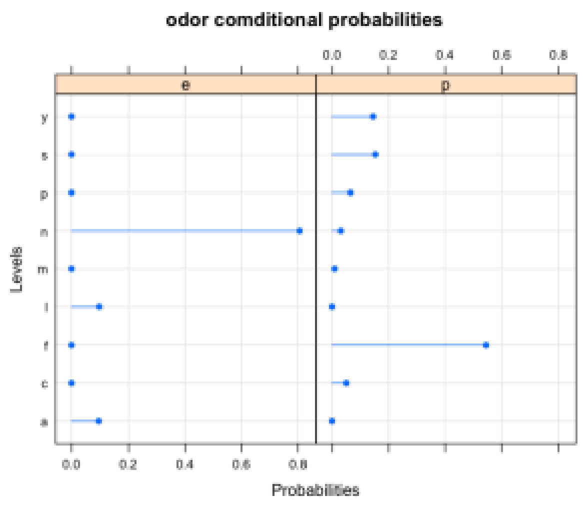
| NB | | e | p |
|---|---|---|---|
| Predi -cted | e | 826 | 97 |
| | p | 18 | 700 |

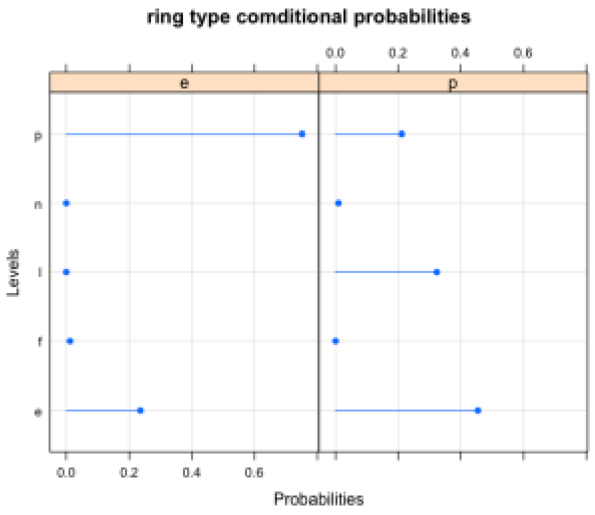
| TAN | | e | p |
|---|---|---|---|
| Predi- cted | e | 691 | 160 |
| | p | 153 | 637 |

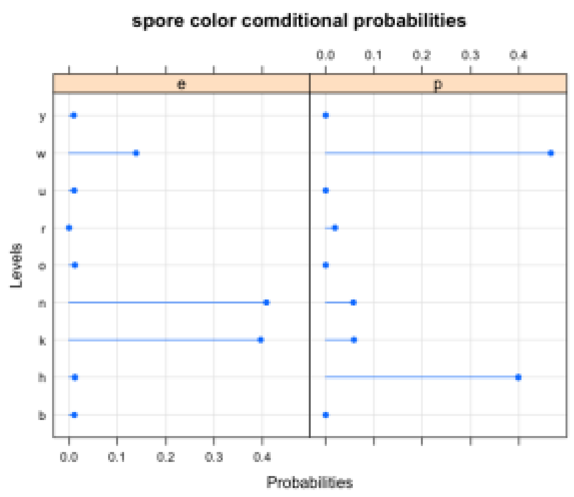
| DT | | e | p |
|---|---|---|---|
| Predi -cted | e | 784 | 30 |
| | p | 60 | 767 |

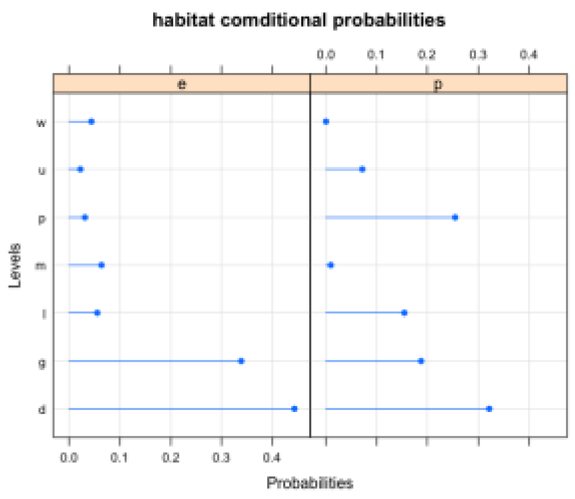
Generally, although GS algorithm seems to have a better predicting accuracy but since it's not validated by 10-fold cross validation, it's not as trustworthy as others. And the graphical model of GS algorithm is partially directed with a great number of attributes not being connected to any other attributes. After comparing all the methods, I decided to go with Naïve Bayes classifier. Also decision tree has a higher accuracy, but Bayesian network can measure the joint probabilities using conditional probabilities and display the change in probabilities when attribute value changes.

Here is the plot conditional probabilities of some attributes.

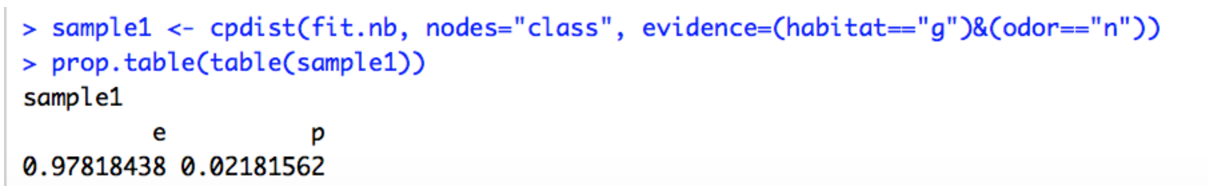








After choosing the classifier, we can now test a probability of certain observation. For example, when habitat is “g” and odor is “n”, NB will be able to provide a probability of that observation belong to each class as shown below.



```
> sample1 <- cpdist(fit.nb, nodes="class", evidence=(habitat=="g")&(odor=="n"))
> prop.table(table(sample1))
sample1
         e          p 
0.97818438 0.02181562
```

After conducting all those analysis, complex Bayesian network such as gs and hc didn’t have any strikingly advantage over naïve Bayes. From my point of view, it is because the specialty of this mushroom dataset. The dataset has 22 variables and they don’t seem to have a lot of effect on one another, therefore, a simple naïve Bayes classifier may even work better here.

Now let’s apply the NM model on the test dataset and calculate the confusion matrix.

| NB | | e | p |
|---|---|---|---|
| Predicted | e | 840 | 1 |
| | p | 4 | 796 |

The performance looks decent. It has the accuracy of 99.7% and 95% confidence interval of (0.9929, 0.999).

## 6. Conclusion

There’re some limitations in this study. First, the dataset might not be the perfect for Bayesian network. This mushroom dataset has simplex relationships between its attributes and I wasn’t able to look into the joint probabilities with the final model I chose. When using the Grow-Shrink algorithm, arcs need to be manually set and added but due to the lack of domain knowledge, I was not able to solve that issue, which may lead to the mediocre performance of GS model. Moreover, I fail on conducting 10-fold cross validation on the GS and TAN classifiers, also due to partially directed arcs and the removal of a R package.

The analysis shows that the advantage of Bayesian Network is that is can capture the joint probabilities by using conditional probabilities formula, and also can calculate conditional probabilities under certain condition, which is saying, it works great for probabilistic queries. As a result, it will be perfect for incomplete data and the calculation of conditional probabilities when changing one or several values of the attributes.

## 7. Appendix

```
library(bnlearn)
# source("http://bioconductor.org/biocLite.R")
# biocLite(c("graph", "RBGL", "Rgraphviz"))
library(Rgraphviz)
library(caret)
library(rpart)
library(klaR)
#library(bnclassify)


setwd("~/Desktop/CSC 529/case study 3")
mushroom <- read.table("agaricus-lepiota.data", sep = ",", header = F)
head(mushroom)
str(mushroom)

#data cleaning
#detect missing value
apply(is.na(mushroom),2,sum)

#add varible names
colnames(mushroom)[1] <- "class"
colnames(mushroom)[2] <- "cap_shape"
colnames(mushroom)[3] <- "cap_surface"
colnames(mushroom)[4] <- "cap_color"
colnames(mushroom)[5] <- "bruises"
colnames(mushroom)[6] <- "odor"
colnames(mushroom)[7] <- "gill_attachment"
colnames(mushroom)[8] <- "gill_spacing"
colnames(mushroom)[9] <- "gill_size"
colnames(mushroom)[10] <- "gill_color"
colnames(mushroom)[11] <- "stalk_shape"
colnames(mushroom)[12] <- "stalk_root"
colnames(mushroom)[13] <- "stalk_sf_ab_r"
colnames(mushroom)[14] <- "stalk_sf_bl_r"
colnames(mushroom)[15] <- "stalk_col_ab_r"
colnames(mushroom)[16] <- "stalk_col_bl_r"
colnames(mushroom)[17] <- "veil_type"
colnames(mushroom)[18] <- "veil_color"
colnames(mushroom)[19] <- "ring_no"
colnames(mushroom)[20] <- "ring_type"
colnames(mushroom)[21] <- "spore_col"
colnames(mushroom)[22] <- "population"
colnames(mushroom)[23] <- "habitat"


#since variable 17, veil-color only has 1 level, and bayesian network need at
least 2 levels to work with, I removed variable 17 here.
mushroom$veil_type <- NULL

#exploratory analysis
str(mushroom)
summary(mushroom)

#plot frequency bar charts
par(mfcol=c(1, 1))
#cap shape grouped by class
capshape_class <- mushroom[ , c(1, 2)]
head(capshape_class)
tab_capshape_class <- table(capshape_class)
tab_capshape_class
barplot(tab_capshape_class, main="Mushroom cap shape by class",
xlab="cap shape", ylab="Count",
        names.arg=c("b","c","f","k","s","x"), font.main=2, col=c("blue",
"yellow"), beside=T, ylim = c(0,2500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#cap surface grouped by class
capsurface_class <- mushroom[ , c(1, 3)]
head(capsurface_class)
tab_capsurface_class <- table(capsurface_class)
tab_capsurface_class
barplot(tab_capsurface_class, main="Mushroom cap surface by class",
xlab="cap surface", ylab="Count",
    names.arg=c("f","g","s","y"),font.main=2, col=c("blue", "yellow"),
beside=T, ylim = c(0,2500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))
```

```
#cap color grouped by class
capcolor_class <- mushroom[ , c(1, 4)]
head(capcolor_class)
tab_capcolor_class <- table(capcolor_class)
tab_capcolor_class
barplot(tab_capcolor_class, main="Mushroom cap color by class",
xlab="cap color", ylab="Count",
        names.arg=c("b","c","e","g","n","p","r","u","w","y"),font.main=2,
col=c("blue", "yellow"), beside=T, ylim = c(0,1500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#bruises grouped by class
bruises_class <- mushroom[ , c(1, 5)]
head(bruises_class)
tab_bruises_class <- table(bruises_class)
tab_bruises_class
barplot(tab_bruises_class, main="Mushroom bruises by class",
xlab="bruises", ylab="Count",
        names.arg=c("f","t"),font.main=2, col=c("blue", "yellow"), beside=T,
ylim = c(0,3500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#odor grouped by class
odor_class <- mushroom[ , c(1, 6)]
head(odor_class)
tab_odor_class <- table(odor_class)
tab_odor_class
barplot(tab_odor_class, main="Mushroom odor by class", xlab="odor",
ylab="Count",
        names.arg=c("a","c","f","l","m","n","p","s","y"), font.main=2, col=c("blue",
"yellow"), beside=T, ylim = c(0,3500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#gillattachment grouped by class
gillattachment_class <- mushroom[ , c(1, 7)]
head(gillattachment_class)
tab_gillattachment_class <- table(gillattachment_class)
tab_gillattachment_class
barplot(tab_gillattachment_class, main="Mushroom gillattachment by class",
xlab="gillattachment", ylab="Count",
        names.arg=c("a","f"),  font.main=2,  col=c("blue",  "yellow"),  beside=T,
ylim = c(0,5000))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#gillspacing grouped by class
gillspacing_class <- mushroom[ , c(1, 8)]
head(gillspacing_class)
tab_gillspacing_class <- table(gillspacing_class)
tab_gillspacing_class
barplot(tab_gillspacing_class,   main="Mushroom   gillspacing   by   class",
xlab="gillspacing", ylab="Count",
        names.arg=c("c","w"),  font.main=2,  col=c("blue",  "yellow"),  beside=T,
ylim = c(0,4000))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#gillsize grouped by class
gillsize_class <- mushroom[ , c(1, 9)]
head(gillsize_class)
tab_gillsize_class <- table(gillsize_class)
tab_gillsize_class
barplot(tab_gillsize_class, main="Mushroom gillsize by class", xlab="gillsize",
ylab="Count",
        names.arg=c("b","n"),  font.main=2,  col=c("blue",  "yellow"),  beside=T,
ylim = c(0,4000))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#gillcolor grouped by class
gillcolor_class <- mushroom[ , c(1, 10)]
head(gillcolor_class)
tab_gillcolor_class <- table(gillcolor_class)
tab_gillcolor_class
barplot(tab_gillcolor_class,    main="Mushroom    gillcolor    by    class",
xlab="gillcolor", ylab="Count",
        names.arg=c("b","e","g","h","k","n","o","p","r","u","w","y"),   font.main=2,
col=c("blue", "yellow"),
        beside=T, ylim = c(0,2000))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#stalkshape grouped by class
stalkshape_class <- mushroom[ , c(1, 11)]
```

```
head(stalkshape_class)
tab_stalkshape_class <- table(stalkshape_class)
tab_stalkshape_class
barplot(tab_stalkshape_class, main="Mushroom stalkshape by class",
xlab="stalkshape", ylab="Count",
        names.arg=c("e","t"), font.main=2, col=c("blue", "yellow"),
        beside=T, ylim = c(0,3500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#stalkroot grouped by class
stalkroot_class <- mushroom[ , c(1, 12)]
head(stalkroot_class)
tab_stalkroot_class <- table(stalkroot_class)
tab_stalkroot_class
barplot(tab_stalkroot_class, main="Mushroom stalkroot by class",
xlab="stalkroot", ylab="Count",
        names.arg=c("?","b","c","e","r"), font.main=2, col=c("blue", "yellow"),
        beside=T, ylim = c(0,2000))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#stalksfabovering grouped by class
stalksfabovering_class <- mushroom[ , c(1, 13)]
head(stalksfabovering_class)
tab_stalksfabovering_class <- table(stalksfabovering_class)
tab_stalksfabovering_class
barplot(tab_stalksfabovering_class, main="Mushroom stalk surface above
ring by class", xlab="stalk surface above ring", ylab="Count",
        names.arg=c("f","k","s","y"), font.main=2, col=c("blue", "yellow"),
        beside=T, ylim = c(0,4000))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#stalksfbelowring grouped by class
stalksfbelowring_class <- mushroom[ , c(1, 14)]
head(stalksfbelowring_class)
tab_stalksfbelowring_class <- table(stalksfbelowring_class)
tab_stalksfbelowring_class
barplot(tab_stalksfbelowring_class, main="Mushroom stalk surface below
ring by class", xlab="stalk surface below ring", ylab="Count",
        names.arg=c("f","k","s","y"), font.main=2, col=c("blue", "yellow"),
        beside=T, ylim = c(0,4000))
```

```
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#stalkclabovering grouped by class
stalkclabovering_class <- mushroom[ , c(1, 15)]
head(stalkclabovering_class)
tab_stalkclabovering_class <- table(stalkclabovering_class)
tab_stalkclabovering_class
barplot(tab_stalkclabovering_class, main="Mushroom stalk color above ring
by class", xlab="stalk color above ring", ylab="Count",
        names.arg=c("b","c","e","g","n","o","p","w","y"),        font.main=2,
col=c("blue", "yellow"),
        beside=T, ylim = c(0,3500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#stalkclbelowring grouped by class
stalkclbelowring_class <- mushroom[ , c(1, 16)]
head(stalkclbelowring_class)
tab_stalkclbelowring_class <- table(stalkclbelowring_class)
tab_stalkclbelowring_class
barplot(tab_stalkclbelowring_class, main="Mushroom stalk color below ring
by class", xlab="stalk color below ring", ylab="Count",
        names.arg=c("b","c","e","g","n","o","p","w","y"),        font.main=2,
col=c("blue", "yellow"),
        beside=T, ylim = c(0,3500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#veilcolor grouped by class
veilcolor_class <- mushroom[ , c(1, 17)]
head(veilcolor_class)
tab_veilcolor_class <- table(veilcolor_class)
tab_veilcolor_class
barplot(tab_veilcolor_class,   main="Mushroom   veilcolor   by   class",
xlab="veilcolor", ylab="Count",
        names.arg=c("n","o","w","y"), font.main=2, col=c("blue", "yellow"),
        beside=T, ylim = c(0,5000))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#ringno grouped by class
ringno_class <- mushroom[ , c(1, 18)]
head(ringno_class)
```

```
tab_ringno_class <- table(ringno_class)
tab_ringno_class
barplot(tab_ringno_class, main="Mushroom ring number by class",
xlab="ring no", ylab="Count",
        names.arg=c("n","o","t"), font.main=2, col=c("blue", "yellow"),
        beside=T, ylim = c(0,4500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#ringtype grouped by class
ringtype_class <- mushroom[ , c(1, 19)]
head(ringtype_class)
tab_ringtype_class <- table(ringtype_class)
tab_ringtype_class
barplot(tab_ringtype_class, main="Mushroom ring type by class", xlab="ring
type", ylab="Count",
        names.arg=c("e","f","l","n","p"), font.main=2, col=c("blue", "yellow"),
        beside=T, ylim = c(0,4000))
legend(x="topright", legend=c("edible", "poisonuus"), fill=c("blue", "yellow"))

#sporecolor grouped by class
sporecolor_class <- mushroom[ , c(1, 20)]
head(sporecolor_class)
tab_sporecolor_class <- table(sporecolor_class)
tab_sporecolor_class
barplot(tab_sporecolor_class, main="Mushroom spore color by class",
xlab="spore color", ylab="Count",
        names.arg=c("b","h","k","n","o","r","u","w","y"),             font.main=2,
col=c("blue", "yellow"),
        beside=T, ylim = c(0,2500))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#population grouped by class
population_class <- mushroom[ , c(1, 21)]
head(population_class)
tab_population_class <- table(population_class)
tab_population_class
barplot(tab_population_class, main="Mushroom population by class",
xlab="population", ylab="Count",
        names.arg=c("a","c","n","s","v","y"),
        font.main=2, col=c("blue", "yellow"), beside=T, ylim = c(0,3500))
```

```
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))

#habitat grouped by class
habitat_class <- mushroom[ , c(1, 22)]
head(habitat_class)
tab_habitat_class <- table(habitat_class)
tab_habitat_class
barplot(tab_habitat_class, main="Mushroom habitat by class", xlab="habitat",
ylab="Count",
        names.arg=c("d","g","l","m","p","u","w"),     font.main=2,     col=c("blue",
"yellow"), beside=T, ylim = c(0,2000))
legend(x="topright", legend=c("edible", "poisonous"), fill=c("blue", "yellow"))



#create training and test dataset
ind <- sample(2, nrow(mushroom), replace=TRUE, prob=c(0.8, 0.2))
train <- mushroom[ind==1,]
test <- mushroom[ind==2,]

#create gs bayesian network
mushroom.gs <- gs(train)
mushroom.gs
plot(mushroom.gs, main = "gs algorithms")
modelstring(mushroom.gs)

#fit parameters of gs bayesian network
fit.gs <- bn.fit(mushroom.gs, data=train, method="mle")
summary(fit.gs)
fit.gs
mushroom.gs
pred.fit.gs <- predict(fit.gs, data=test, node="class")
pred.fit.gs
t.fit.gs <- table(pred.fit.gs,test$class)
confusionMatrix(t.fit.gs)
#   mushroom.gs.bic   <-   bn.cv(train,   bn="gs",   loss="pred",   loss.args   =
list(target="class"))

# mushroom.iamb <- iamb(mushroom)
# mushroom.fast <- fast.iamb(mushroom)
# mushroom.inter<- inter.iamb(mushroom)
# compare(mushroom.gs, mushroom.iamb)
# compare(mushroom.gs, mushroom.fast)
# compare(mushroom.gs, mushroom.inter)

#create hc bayesian network
mushroom.hc <- hc(train, score="bic")
mushroom.hc
plot(mushroom.hc, main = "hc algorithms")
modelstring(mushroom.hc)

compare(target=mushroom.gs, current=mushroom.hc, arcs=F)

#build hc bayesian network using 10-fold cross validation
mushroom.hc.bic <- bn.cv(mushroom, bn="hc", loss="pred", loss.args =
list(target="class"))
#mushroom.gs <- bn.cv(mushroom, bn="gs", loss="pred", loss.args =
list(target="class"))
mushroom.hc.bic
# mushroom.hc.bde <- bn.cv(mushroom, bn="hc", algorithm.args =
list(score="bde", iss=1))
# mushroom.hc.bde

#fit parameters of hc bayesian network
fit.hc <- bn.fit(mushroom.hc, data=train, method="mle")
pred.fit.hc <- predict(fit.hc, data=test, node="class")
pred.fit.hc
t.fit.hc <- table(pred.fit.hc,test$class)
confusionMatrix(t.fit.hc)

#naive bayes
train_control <- trainControl(method="cv", number=10)
nb.cv <- train(class~., data=train, trControl=train_control, method="nb")
nb.cv
pred.nb.cv <- predict(nb.cv, data=test)
t.nb.cv <- table(pred.nb.cv,test$class)
confusionMatrix(t.nb.cv)

#naive bayes model
nb <- naive.bayes(train, "class")
```

```
nb
plot(nb, main = "naive bayes algorithms")

#fit parameters of nb bayesian network
fit.nb <- bn.fit(nb, data=train, method="mle")
pred.fit.nb <- predict(fit.nb, data=test)
t.fit.nb <- table(pred.fit.nb,test$class)
confusionMatrix(t.fit.nb)

sample1 <- cpdist(fit.nb, nodes="class",
evidence=(habitat=="g")&(odor=="n"))
prop.table(table(sample1))

nb
fit.nb
bn.fit.dotplot(fit.nb$odor, main="odor comditional probabilities")
bn.fit.dotplot(fit.nb$ring_type, main="ring type comditional probabilities")
bn.fit.dotplot(fit.nb$spore_col, main="spore color comditional probabilities")
bn.fit.dotplot(fit.nb$habitat, main="habitat comditional probabilities")


compare(mushroom.gs, nb)

#Tree Augmented Naive Bayes Classifier
#tan <- train(class~., data=train, trControl=train_control, method="tan",
tuneGrid=grid)
tan <- tree.bayes(mushroom, "class")
tan
pred.tan <- predict(tan, test)
t.tan <- table(pred.tan,test$class)
confusionMatrix(t.tan)
plot(tan, main = "tan algorithms")
modelstring(tan)

#fit parameters of TAN bayesian network
fit.tan <- bn.fit(tan, data=train, method="mle")
summary(fit.tan)
fit.tan
pred.fit.tan <- predict(fit.tan, data=test)
t.fit.tan <- table(pred.fit.tan,test$class)
```

```
confusionMatrix(t.fit.tan)

compare(nb, tan)

#create a decision tree
tree <- train(class~., data=train, trControl=train_control, method="rpart")
tree
pred.tree <- predict(tree, test)
t.tree <- table(pred.tree,test$class)
confusionMatrix(t.tree)
```