Mushroom Data Set

Data Set Information:

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.

Attribute Information:

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

R Code:

- setwd('~/Desktop') #set working directory to desktop
- mushroom<-read.table("mushroom.data")
- attach(mushroom)
  # give attributes to all the columns using colnames method
- colnames(mushroom) = c("classes", "cap-shape", "cap-surface", "cap-color", "bruises?", "odor", "gill-attachment", "gill-spacing", "gill-size", "gill-color", "stalk-shape", "stalk-root", "stalk-surface-above-ring", "stalk-surface-below-ring", "stalk-color-above-ring", "stalk-color-below-ring", "veil-type", "veil-color", "ring-number", "ring-type", "spore-print-color", "population", "habitat")
- sapply(mushroom, class) # to find the attribute types from the table
- library(arules) #import arules package
- rules<-apriori(mushroom) #apriori method without conditions
- sink('out.txt') #output result in a text file name out
- inspect(rules)
  The output is as follows:

	lhs		rhs	support	confidence	lift
[1]	{}	=>	{gill-spacing=c,}	0.8385032	0.8385032	1.0000000
[2]	{}	=>	{ring-number=o,}	0.9217134	0.9217134	1.0000000
[3]	{}	=>	{gill-attachment=f,}	0.9741507	0.9741507	1.0000000
[4]	{}	=>	{veil-color=w,}	0.9753816	0.9753816	1.0000000
[5]	{}	=>	{veil-type=p,}	1.0000000	1.0000000	1.0000000
[6]	{cap-shape=k,}	=>	{veil-type=p,}	0.1019202	1.0000000	1.0000000
[7]	{habitat=l}	=>	{bruises?=f,}	0.1014279	0.9903846	1.6945840
[8]	{habitat=l}	=>	{ring-number=o,}	0.1024126	1.0000000	1.0849359
[9]	{habitat=l}	=>	{veil-type=p,}	0.1024126	1.0000000	1.0000000
[10]	{cap-color=w,}	=>	{stalk-color-below-ring=w,}	0.1280158	1.0000000	1.8531022
[11]	{cap-color=w,}	=>	{stalk-color-above-ring=w,}	0.1280158	1.0000000	1.8198925
[12]	{cap-color=w,}	=>	{ring-number=o,}	0.1073363	0.8384615	0.9096770
[13]	{cap-color=w,}	=>	{gill-attachment=f,}	0.1280158	1.0000000	1.0265353
[14]	{cap-color=w,}	=>	{veil-color=w,}	0.1280158	1.0000000	1.0252398
[15]	{cap-color=w,}	=>	{veil-type=p,}	0.1280158	1.0000000	1.0000000
[16]	{gill-color=n,}	=>	{ring-type=p,}	0.1053668	0.8167939	1.6722867
[17]	{gill-color=n,}	=>	{classes=e,}	0.1152142	0.8931298	1.7242838
[18]	{gill-color=n,}	=>	{stalk-surface-below-ring=s,}	0.1093058	0.8473282	1.3945897
[19]	{gill-color=n,}	=>	{stalk-surface-above-ring=s,}	0.1171837	0.9083969	1.4257760
[20]	{gill-color=n,}	=>	{gill-size=b,}	0.1083210	0.8396947	1.2155523

The above output goes on up to 3,315,185 rules.
We don't want all the rules we are interested in rules that are either edible or poisonous and whose minimum support is 30 % and the confidence is 90%. Hence we modify the apriori function by passing parameters to it to filter the results.

rules with rhs containing "edible" and "poisonous" only

- rules <- apriori(mushroom,
  parameter = list(minlen=2, supp=0.3, conf=0.8),
  appearance = list(rhs=c("classes=e,", "classes=p,"),
  default="lhs"),
  control = list(verbose=F)) #apriori method with conditions

This method returns 576 rules having classes as edible and poisonous only with the specified support and confidence

- rules.sorted <- sort(rules, by="lift")
- sink("out1.txt")
- inspect(rules.sorted)

By the above steps the rules are sorted by lift and results are stored in the out1 text file. Some of the output is as follows:

	lhs	rhs	support	confidence	lift
[1]	{bruises?=f,, gill-attachment=f,, gill-spacing=c,, population=v,}	=> {classes=p,}	0.3023141	0.9715190	2.015480
[2]	{bruises?=f,, gill-spacing=c,, veil-color=w,, population=v,}	=> {classes=p,}	0.3023141	0.9715190	2.015480
[3]	{bruises?=f,, gill-attachment=f,, gill-spacing=c,, ring-number=o,, population=v,}	=> {classes=p,}	0.3023141	0.9715190	2.015480
[4]	{bruises?=f,, gill-spacing=c,, veil-color=w,, ring-number=o,, population=v,}	=> {classes=p,}	0.3023141	0.9715190	2.015480
[5]	{bruises?=f,, gill-attachment=f,, gill-spacing=c,, veil-color=w,, population=v,}	=> {classes=p,}	0.3023141	0.9715190	2.015480

find redundant rules

- subset.matrix <- is.subset(rules.sorted, rules.sorted)
- subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA
- redundant <- colSums(subset.matrix, na.rm=T) >= 1
- sink("out2.txt")
- which(redundant)

The above function returns values that are redundant in the out2 text file as shoen below:

[3] {classes=p, bruises?=f, gill-attachment=f, gill-spacing=c, ring-number=o, population=v,}

[4] {classes=p, bruises?=f, gill-spacing=c, veil-color=w, ring-number=o, population=v,}

The above rules states that the 4th rule is included in the 3rd rule and hence is redundant and we need to eliminate such redundancies by inspecting all the rules we got from the apriori method with parameters.

```
# remove redundant rules
```

- rules.pruned <- rules.sorted[!redundant]
- sink("out3.txt")
- inspect(rules.pruned)

By using the above method, we eliminate all the redundant rules which is narrowed down to 53 rules and the result is stored in the out3 text file.

```
#Visualize
```

- library(arulesViz)
- plot(rules)

The above method is used to visualize all the rules after pruning which is shown in the scatterplot below: