Mushroom classification model

dgr

2022-06-30

I. Overview

For the final capstone project of the Harvard's Professional Certification in Data Science, I selected a publicly available dataset for a classification problem with the goal of building an algorithm that prescribes if a mushroom is edible or poisonous. I selected this dataset because 1) it was a different type of dataset and analysis (it is a classification problem) compared to the MovieLens analysis, and 2) it piqued my interest because of my ecology background.

The dataset was available for download from UC Irvine's Center for Machine Learning and Intelligent Systems at https://archive.ics.uci.edu/ml/datasets/Secondary+Mushroom+Dataset# and it included 61069 hypothetical mushrooms with caps based on 173 species (353 mushrooms per species). The initial set had 21 variables (i.e., columns), one being the outcome (edible vs. poisonous) and the other 20 being possible features. Several predictors had many missing values, thus I reduced their number to 11, which was still a large number as far as computational time. After initial data cleaning, exploration, and visualization, I split the dataset into training and test sets (50-50 split). I tested 6 prediction models, including logistic, k nearest neighbor, classification tree, random forest (2 models), and ensemble.

The best performing model was a random forest model computed using the caret::train function (method = "rf") with two tuning parameters. The accuracy was 0.9935. I then looked at variable importance and re-run the best model using only the five most important variable. The accuracy was lower at 0.9168, but this allowed for the selection of fewer variables to visually explore in more detail. I, thus, included a few graphs showing how these variables relate to each other.

I have successfully developed a model that predicted with great accuracy whether a mushroom is edible or poisonous. This project allowed me to develop a model on my own applying the knowledge I gain in the previous eight courses of this certification program, giving me confidence on the skills I gained, and inspiring me to learn more about the different approaches available for developing useful models.

II. Methods and Analysis

1. Data download

I downloaded the data from the UC Irvine website making sure the categorical variables are read as factors to facilitate model building.

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
if(!require(readr)) install.packages("readr", repos = "http://cran.us.r-project.org")
if(!require(mlbench)) install.packages("mlbench", repos = "http://cran.us.r-project.org")
if(!require(rpart.plot)) install.packages("rpart.plot", repos = "http://cran.us.r-project.org")
```

```
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(Rborist)) install.packages("Rborist", repos = "http://cran.us.r-project.org")
library(tidyverse)
library(caret)
library(data.table)
library(readr)
library(rpart.plot)
library(ggplot2)
library(mlbench)
library(Rborist)

temp <- tempfile()
url <- "https://archive.ics.uci.edu/ml/machine-learning-databases/00615/MushroomDataset.zip"
download.file(url, temp)
mush <- read.table(unz(temp, "MushroomDataset/secondary_data.csv"), sep = ";",
                   header = TRUE, stringsAsFactors = TRUE)
```

2. Data exploration and preparation

I looked at the structure and variable summary for the 21 variables in the set.

```
str(mush)
```

```
## 'data.frame':    61069 obs. of  21 variables:
##  $ class                : Factor w/ 2 levels "e","p": 2 2 2 2 2 2 2 2 2 2 ...
##  $ cap.diameter         : num  15.3 16.6 14.1 14.2 14.6 ...
##  $ cap.shape            : Factor w/ 7 levels "b","c","f","o",..: 7 7 7 3 7 7 3 7 3 3 ...
##  $ cap.surface          : Factor w/ 12 levels "","d","e","g",..: 4 4 4 5 5 4 5 5 4 4 ...
##  $ cap.color            : Factor w/ 12 levels "b","e","g","k",..: 7 7 7 2 7 7 7 2 7 2 ...
##  $ does.bruise.or.bleed: Factor w/ 2 levels "f","t": 1 1 1 1 1 1 1 1 1 1 ...
##  $ gill.attachment      : Factor w/ 8 levels "","a","d","e",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ gill.spacing         : Factor w/ 4 levels "","c","d","f": 1 1 1 1 1 1 1 1 1 1 ...
##  $ gill.color           : Factor w/ 12 levels "b","e","f","g",..: 11 11 11 11 11 11 11 11 11 11 ...
##  $ stem.height          : num  16.9 18 17.8 15.8 16.5 ...
##  $ stem.width           : num  17.1 18.2 17.7 16 17.2 ...
##  $ stem.root            : Factor w/ 6 levels "","b","c","f",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ stem.surface         : Factor w/ 9 levels "","f","g","h",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ stem.color           : Factor w/ 13 levels "b","e","f","g",..: 12 12 12 12 12 12 12 12 12 12 ...
##  $ veil.type            : Factor w/ 2 levels "","u": 2 2 2 2 2 2 2 2 2 2 ...
##  $ veil.color           : Factor w/ 7 levels "","e","k","n",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ has.ring             : Factor w/ 2 levels "f","t": 2 2 2 2 2 2 2 2 2 2 ...
##  $ ring.type            : Factor w/ 9 levels "","e","f","g",..: 4 4 4 7 7 7 4 7 7 7 ...
##  $ spore.print.color    : Factor w/ 8 levels "","g","k","n",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ habitat              : Factor w/ 8 levels "d","g","h","l",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ season               : Factor w/ 4 levels "a","s","u","w": 4 3 4 4 4 3 4 3 1 4 ...
```

```
head(mush)
```

```
##   class cap.diameter cap.shape cap.surface cap.color does.bruise.or.bleed
## 1     p        15.26         x           g         o                    f
## 2     p        16.60         x           g         o                    f
## 3     p        14.07         x           g         o                    f
## 4     p        14.17         f           h         e                    f
## 5     p        14.64         x           h         o                    f
## 6     p        15.34         x           g         o                    f
```

```
##   gill.attachment gill.spacing gill.color stem.height stem.width stem.root
## 1               e                      w       16.95      17.09         s
## 2               e                      w       17.99      18.19         s
## 3               e                      w       17.80      17.74         s
## 4               e                      w       15.77      15.98         s
## 5               e                      w       16.53      17.20         s
## 6               e                      w       17.84      18.79         s
##   stem.surface stem.color veil.type veil.color has.ring ring.type
## 1            y          w         u          w        t         g
## 2            y          w         u          w        t         g
## 3            y          w         u          w        t         g
## 4            y          w         u          w        t         p
## 5            y          w         u          w        t         p
## 6            y          w         u          w        t         p
##   spore.print.color habitat season
## 1                         d      w
## 2                         d      u
## 3                         d      w
## 4                         d      w
## 5                         d      w
## 6                         d      u
```

```
summary(mush, maxsum =20)
```

```
##  class      cap.diameter    cap.shape cap.surface cap.color
##  e:27181   Min.   : 0.380   b: 5694    :14120     b: 1230
##  p:33888   1st Qu.: 3.480   c: 1815   d: 4432     e: 4035
##            Median : 5.860   f:13404   e: 2584     g: 4420
##            Mean   : 6.734   o: 3460   g: 4724     k: 1279
##            3rd Qu.: 8.540   p: 2598   h: 4974     l:  828
##            Max.   :62.340   s: 7164   i: 2225     n:24218
##                             x:26934   k: 2303     o: 3656
##                                       l: 1412     p: 1703
##                                       s: 7608     r: 1782
##                                       t: 8196     u: 1709
##                                       w: 2150     w: 7666
##                                       y: 6341     y: 8543
##
##  does.bruise.or.bleed gill.attachment gill.spacing gill.color  stem.height
##  f:50479               : 9884          :25063      b:  954    Min.   : 0.000
##  t:10590              a:12698         c:24710      e: 1066    1st Qu.: 4.640
##                       d:10247         d: 7766      f: 3530    Median : 5.950
##                       e: 5648         f: 3530      g: 4118    Mean   : 6.582
##                       f: 3530                      k: 2375    3rd Qu.: 7.740
##                       p: 6001                      n: 9645    Max.   :33.920
##                       s: 5648                      o: 2909
##                       x: 7413                      p: 5983
##                                                    r: 1399
##                                                    u: 1023
##                                                    w:18521
##                                                    y: 9546
##
##    stem.width     stem.root stem.surface stem.color veil.type veil.color
##  Min.   :  0.00    :51538    :38124      b:  173     :57892    :53656
##  1st Qu.:  5.21   b: 3177   f: 1059      e: 2050    u: 3177   e:  181
##  Median : 10.19   c:  706   g: 1765      f: 1059              k:  353
##  Mean   : 12.15   f: 1059   h:  535      g: 2626              n:  525
##  3rd Qu.: 16.57   r: 1412   i: 4396      k:  837              u:  353
```

```
##  Max.   :103.91   s: 3177   k: 1581      l:  226             w: 5474
##                            s: 6025      n:18063             y:  527
##                            t: 2644      o: 2187
##                            y: 4940      p: 1025
##                                         r:  542
##                                         u: 1490
##                                         w:22926
##                                         y: 7865
##  has.ring  ring.type spore.print.color habitat   season
##  f:45890    : 2471    :54715           d:44209   a:30177
##  t:15179   e: 2435   g:  353           g: 7943   s: 2727
##            f:48361   k: 2118           h: 2001   u:22898
##            g: 1240   n: 1059           l: 3168   w: 5267
##            l: 1427   p: 1259           m: 2920
##            m:  353   r:  171           p:  360
##            p: 1265   u:  182           u:  115
##            r: 1399   w: 1212           w:  353
##            z: 2118
##
##
##
##
```

```
dim(mush)
```

```
## [1] 61069    21
```

Based on the summary, there are 21 variables (columns), and 61,069 observation (rows). Also, it is apparent from the summary that this dataset has some variables with many NAs, so the first step was to remove 9 of the variables, each with 2,471 or more missing values.

```
mush <- mush %>%
  select(class, cap.diameter, cap.shape, cap.color, does.bruise.or.bleed,
                       gill.color, stem.height, stem.width, stem.color, has.ring, habitat, season)
str(mush)
```

```
## 'data.frame':    61069 obs. of  12 variables:
##  $ class               : Factor w/ 2 levels "e","p": 2 2 2 2 2 2 2 2 2 2 ...
##  $ cap.diameter        : num  15.3 16.6 14.1 14.2 14.6 ...
##  $ cap.shape           : Factor w/ 7 levels "b","c","f","o",..: 7 7 7 3 7 7 3 7 3 3 ...
##  $ cap.color           : Factor w/ 12 levels "b","e","g","k",..: 7 7 7 2 7 7 7 2 7 2 ...
##  $ does.bruise.or.bleed: Factor w/ 2 levels "f","t": 1 1 1 1 1 1 1 1 1 1 ...
##  $ gill.color          : Factor w/ 12 levels "b","e","f","g",..: 11 11 11 11 11 11 11 11 11 11 ...
##  $ stem.height         : num  16.9 18 17.8 15.8 16.5 ...
##  $ stem.width          : num  17.1 18.2 17.7 16 17.2 ...
##  $ stem.color          : Factor w/ 13 levels "b","e","f","g",..: 12 12 12 12 12 12 12 12 12 12 ...
##  $ has.ring            : Factor w/ 2 levels "f","t": 2 2 2 2 2 2 2 2 2 2 ...
##  $ habitat             : Factor w/ 8 levels "d","g","h","l",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ season              : Factor w/ 4 levels "a","s","u","w": 4 3 4 4 4 3 4 3 1 4 ...
```

```
summary(mush, maxsum =20)
```

```
##  class      cap.diameter    cap.shape cap.color does.bruise.or.bleed gill.color
##  e:27181   Min.   : 0.380   b: 5694   b: 1230   f:50479              b:  954
##  p:33888   1st Qu.: 3.480   c: 1815   e: 4035   t:10590              e: 1066
##            Median : 5.860   f:13404   g: 4420                        f: 3530
```

```
##            Mean   : 6.734   o: 3460   k: 1279                      g: 4118
##            3rd Qu.: 8.540   p: 2598   l:  828                      k: 2375
##            Max.   :62.340   s: 7164   n:24218                      n: 9645
##                             x:26934   o: 3656                      o: 2909
##                                       p: 1703                      p: 5983
##                                       r: 1782                      r: 1399
##                                       u: 1709                      u: 1023
##                                       w: 7666                      w:18521
##                                       y: 8543                      y: 9546
##
##   stem.height       stem.width     stem.color has.ring  habitat   season
##  Min.   : 0.000   Min.   :  0.00   b:  173    f:45890   d:44209   a:30177
##  1st Qu.: 4.640   1st Qu.:  5.21   e: 2050    t:15179   g: 7943   s: 2727
##  Median : 5.950   Median : 10.19   f: 1059              h: 2001   u:22898
##  Mean   : 6.582   Mean   : 12.15   g: 2626              l: 3168   w: 5267
##  3rd Qu.: 7.740   3rd Qu.: 16.57   k:  837              m: 2920
##  Max.   :33.920   Max.   :103.91   l:  226              p:  360
##                                    n:18063              u:  115
##                                    o: 2187              w:  353
##                                    p: 1025
##                                    r:  542
##                                    u: 1490
##                                    w:22926
##                                    y: 7865
```

```
dim(mush)
```

```
## [1] 61069    12
```

The new dataset has now 12 variables, one of each ('class') is the variable that we want to predict. The other 11 variables are potential predictors. Next, I renamed the factor levels for the 9 categorical variables in the dataset to be able to make more informative graphs. Data set description is available at https://archive.ics.uci.edu/ml/datasets/Secondary+Mushroom+Dataset#

```
levels(mush$class) <- list("edible" = "e",  "poisonous" = "p")
levels(mush$cap.shape) <- list("bell"="b", "conical"="c", "convex"="x", "flat"="f",
                               "sunken"="s", "spherical"="p", "others"="o")
levels(mush$cap.color) <- list("brown"="n", "buff"="b", "gray"="g", "green"="r",
                               "pink"="p", "purple"="u", "red"="e", "white"="w",
                               "yellow"="y", "blue"="l", "orange"="o", "black"="k")
levels(mush$does.bruise.or.bleed) <- list("bruises-or-bleeding"="t", "no"="f")

levels(mush$gill.color) <- list("brown"="n", "buff"="b", "gray"="g", "green"="r", "pink"="p",
                                "purple"="u", "red"="e", "white"="w", "yellow"="y", "blue"="l",
                                "orange"="o", "black"="k", "none"="f")
levels(mush$stem.color) <- list("brown"="n", "buff"="b", "gray"="g", "green"="r", "pink"="p",
                                "purple"="u", "red"="e", "white"="w", "yellow"="y", "blue"="l",
                                "orange"="o", "black"="k", "none"="f")
levels(mush$has.ring) <- list("ring"="t", "none"="f")
levels(mush$habitat) <- list("grasses"="g", "leaves"="l", "meadows"="m", "paths"="p", "heaths"="h",
                             "urban"="u", "waste"="w", "woods"="d")
levels(mush$season) <- list("spring"="s", "summer"="u", "autumn"="a", "winter"="w")

str(mush)
```

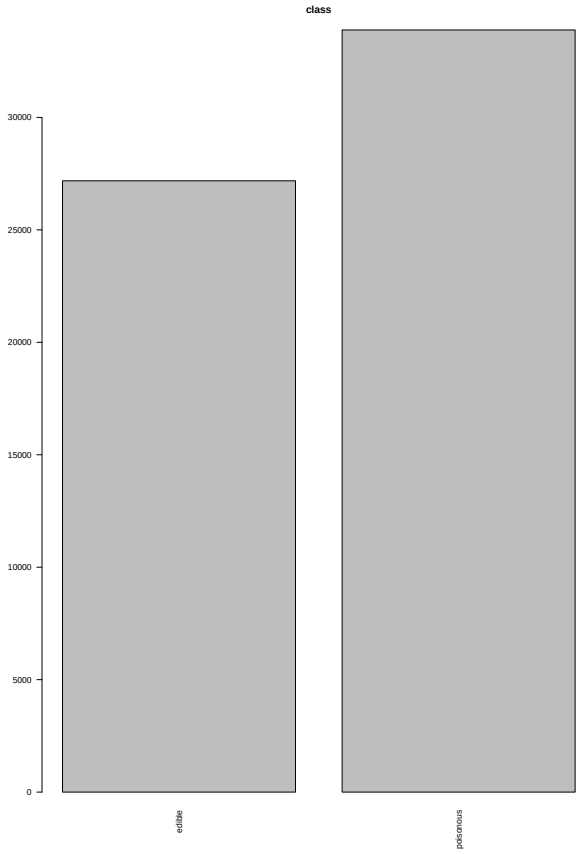
```
## 'data.frame':    61069 obs. of  12 variables:
##  $ class               : Factor w/ 2 levels "edible","poisonous": 2 2 2 2 2 2 2 2 2 2 ...
```

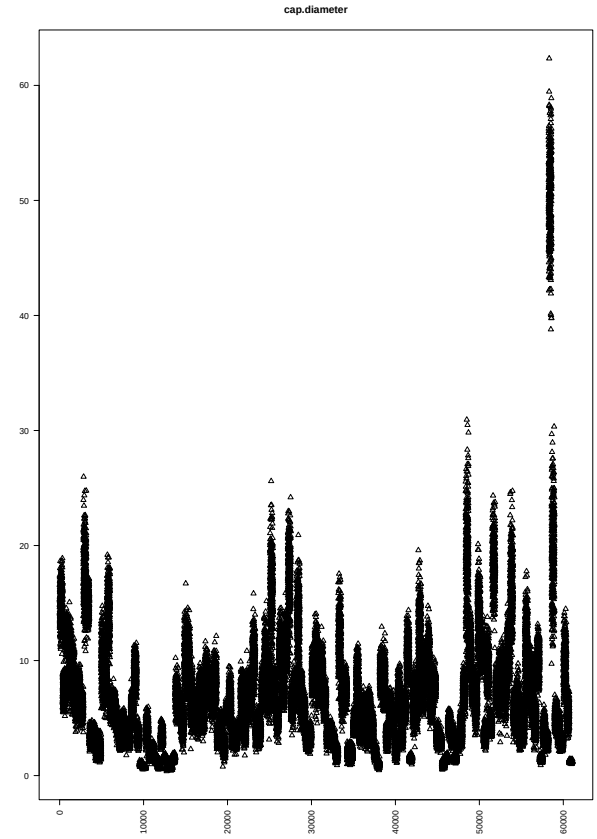
```
##  $ cap.diameter        : num  15.3 16.6 14.1 14.2 14.6 ...
##  $ cap.shape           : Factor w/ 7 levels "bell","conical",..: 3 3 3 4 3 3 4 3 4 4 ...
##  $ cap.color           : Factor w/ 12 levels "brown","buff",..: 11 11 11 7 11 11 11 7 11 7 ...
##  $ does.bruise.or.bleed: Factor w/ 2 levels "bruises-or-bleeding",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ gill.color          : Factor w/ 13 levels "brown","buff",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ stem.height         : num  16.9 18 17.8 15.8 16.5 ...
##  $ stem.width          : num  17.1 18.2 17.7 16 17.2 ...
##  $ stem.color          : Factor w/ 13 levels "brown","buff",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ has.ring            : Factor w/ 2 levels "ring","none": 1 1 1 1 1 1 1 1 1 1 ...
##  $ habitat             : Factor w/ 8 levels "grasses","leaves",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ season              : Factor w/ 4 levels "spring","summer",..: 4 2 4 4 4 2 4 2 3 4 ...
```

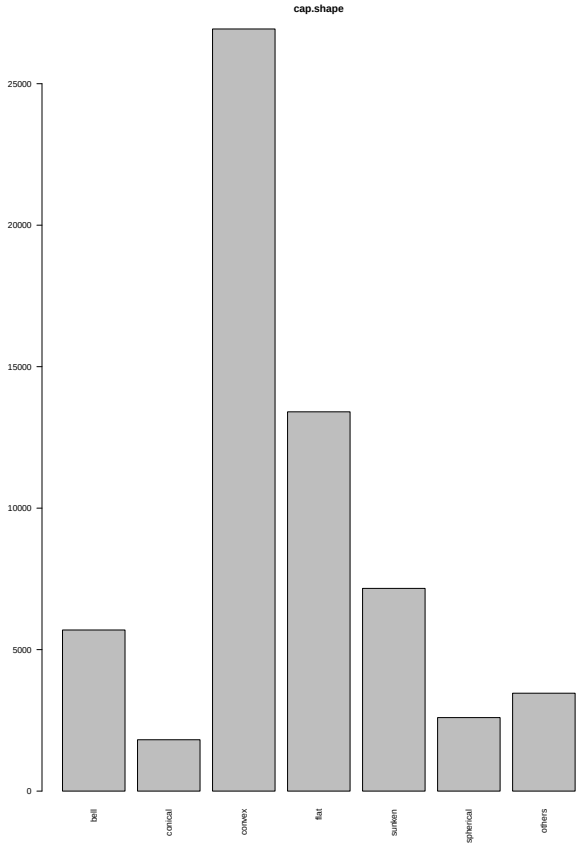
3. Data visualization

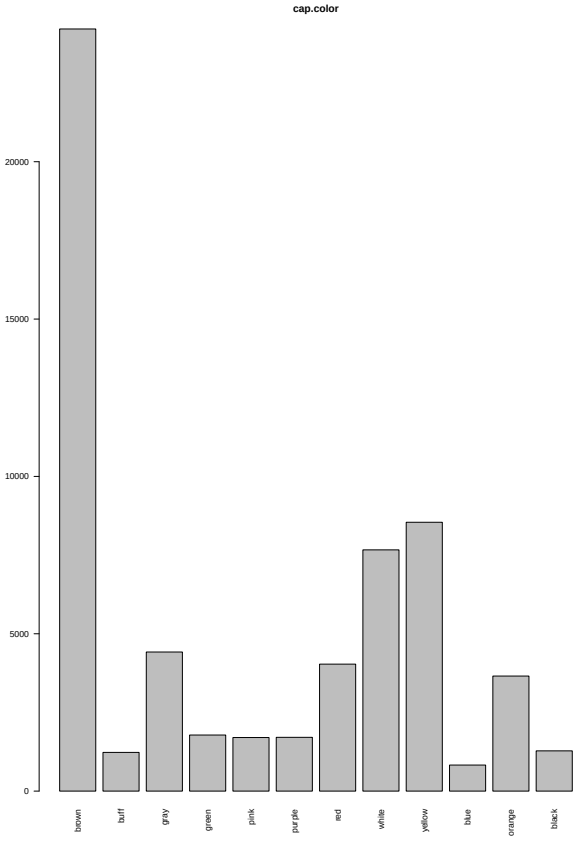
Before splitting the dataset into training and testing sets, I performed some visualizations to get a better idea of the structure of the datasets. These are univariate graphs showing the distribution of each of the 12 variables in the dataset, not the relationships among different variables which I explored after building the model.

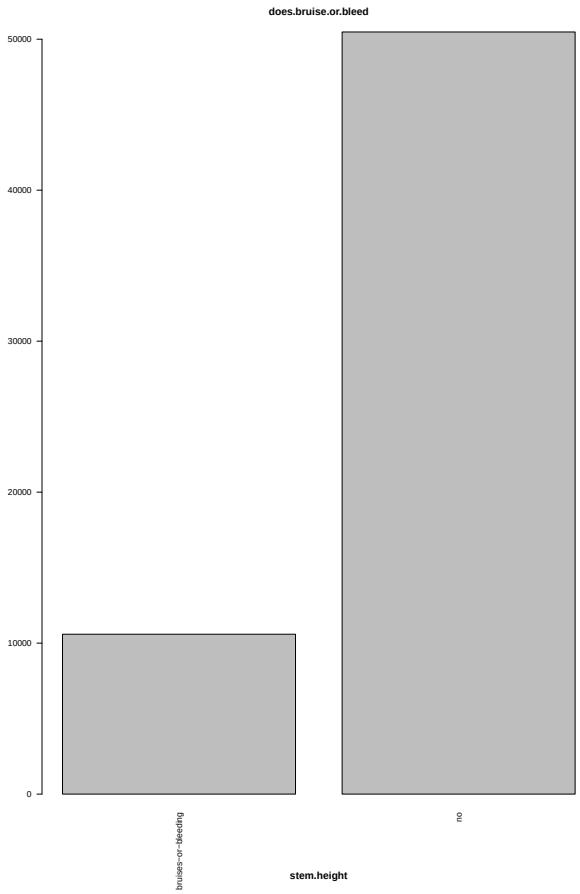
```
par(mfrow=c(2,2)) # Set up a 2 x 2 plotting space
distributions <-function(index)
{plot(mush[,index], main=names(mush[index]),pch=24, las=2,
      xlab="",ylab="")}
lapply(1:12,FUN=distributions)
```

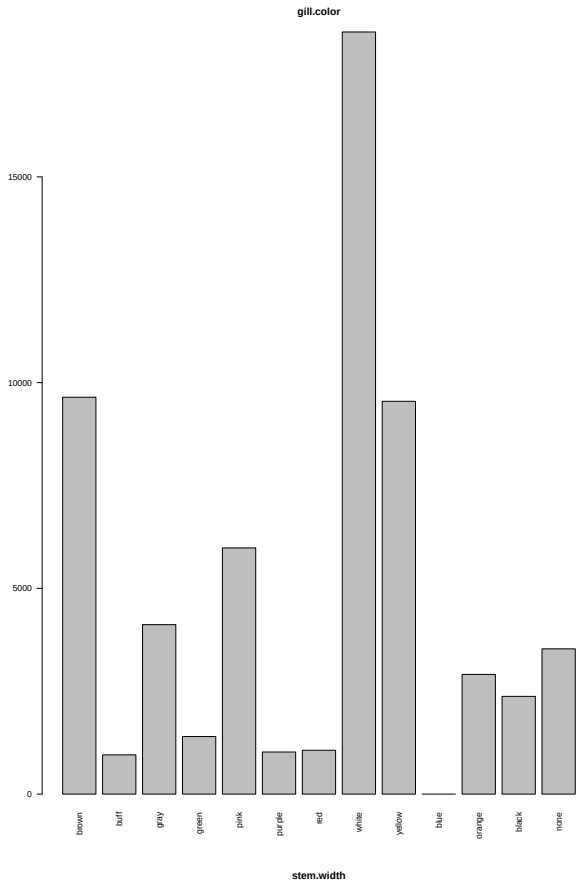

class
30000
25000
20000
15000
10000
5000
0
edible
poisonous

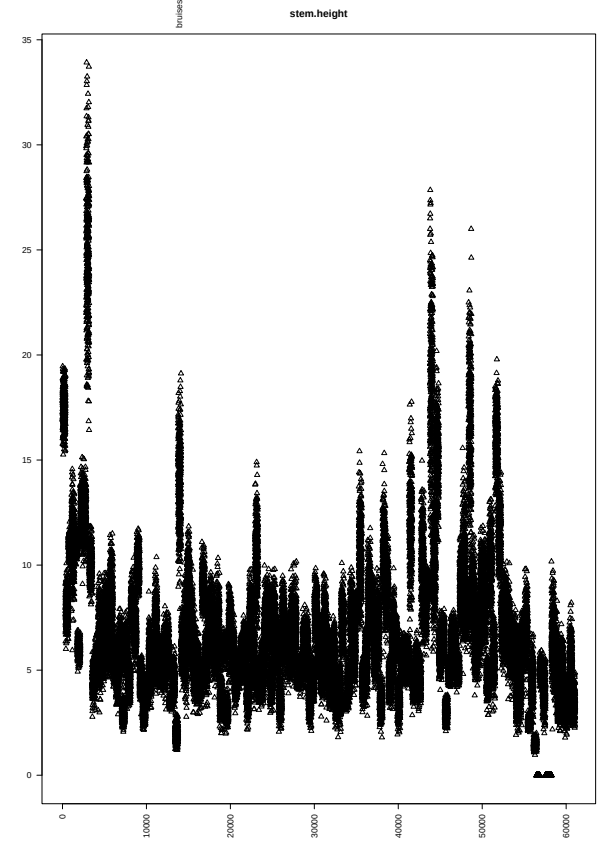

cap.diameter
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000
60000

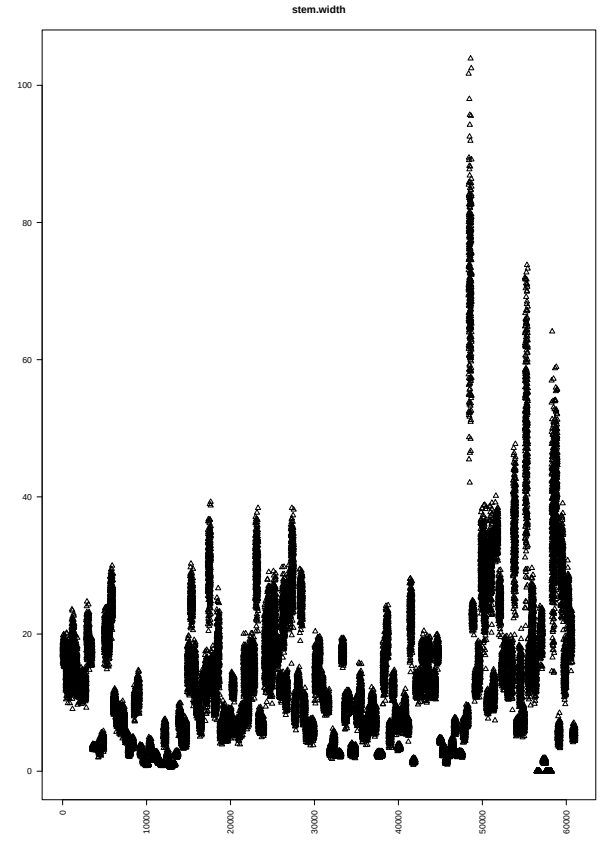

cap.shape
25000
20000
15000
10000
5000
0
bell
conical
convex
flat
sunken
spherical
others

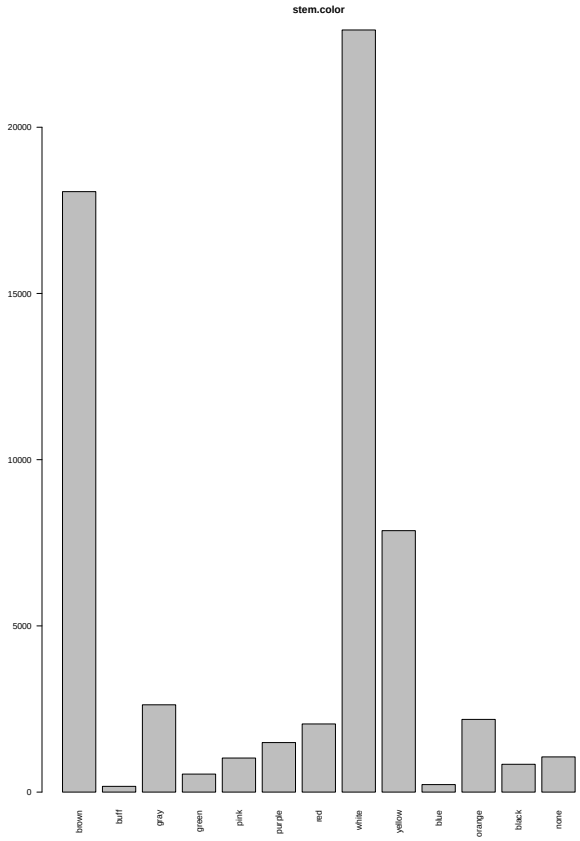

cap.color
20000
15000
10000
5000
0
brown
buff
gray
green
pink
purple
red
white
yellow
blue
orange
black

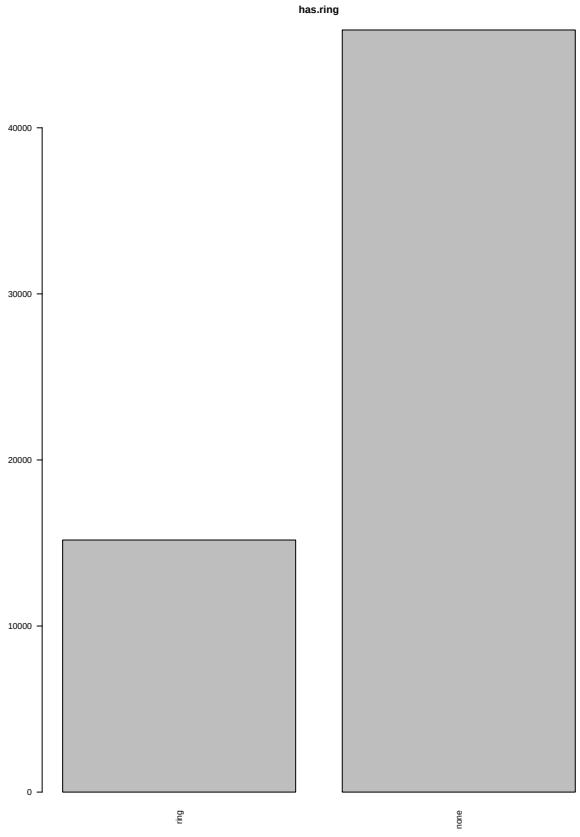

does.bruise.or.bleed
50000
40000
30000
20000
10000
0
bruises−or−bleeding
no

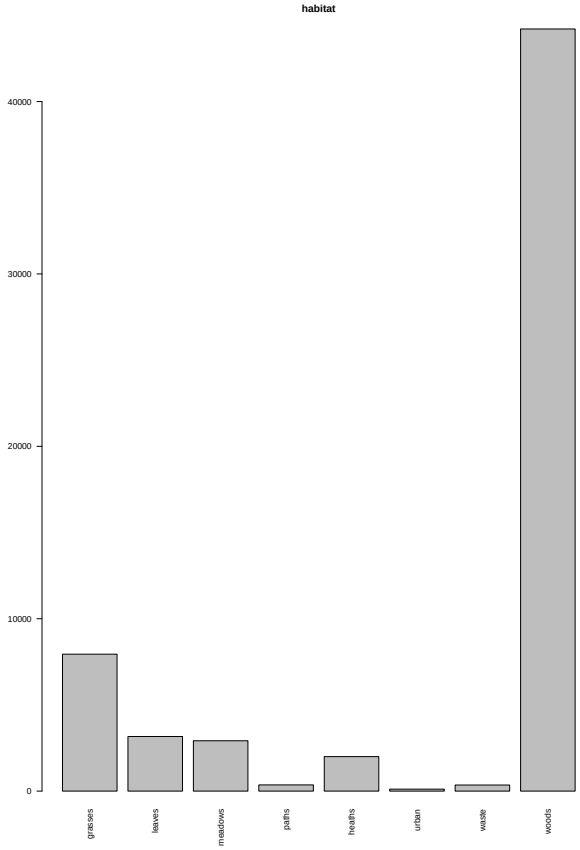

gill.color
15000
10000
5000
0
brown
buff
gray
green
pink
purple
red
white
yellow
blue
orange
black
none

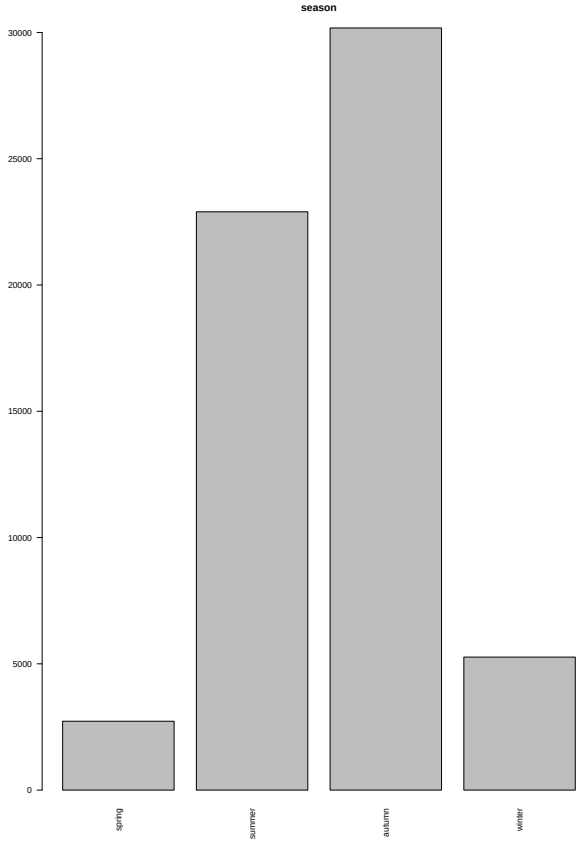

stem.height
35
30
25
20
15
10
5
0
0
10000
20000
30000
40000
50000
60000


stem.width
100
80
60
40
20
0
0
10000
20000
30000
40000
50000
60000


stem.color
20000
15000
10000
5000
0
brown
buff
gray
green
pink
purple
red
white
yellow
blue
orange
black
none


has.ring
40000
30000
20000
10000
0
ring
none


habitat
40000
30000
20000
10000
0
grasses
leaves
meadows
paths
heaths
urban
waste
woods


season
30000
25000
20000
15000
10000
5000
0
spring
summer
autumn
winter

```
## [[1]]
##      [,1]
## [1,]  0.7
## [2,]  1.9
##
## [[2]]
## NULL
##
## [[3]]
##      [,1]
## [1,]  0.7
## [2,]  1.9
## [3,]  3.1
## [4,]  4.3
## [5,]  5.5
## [6,]  6.7
## [7,]  7.9
##
## [[4]]
##       [,1]
##  [1,]  0.7
##  [2,]  1.9
##  [3,]  3.1
##  [4,]  4.3
##  [5,]  5.5
##  [6,]  6.7
##  [7,]  7.9
##  [8,]  9.1
##  [9,] 10.3
## [10,] 11.5
## [11,] 12.7
## [12,] 13.9
##
## [[5]]
##      [,1]
## [1,]  0.7
## [2,]  1.9
##
## [[6]]
##       [,1]
##  [1,]  0.7
##  [2,]  1.9
##  [3,]  3.1
##  [4,]  4.3
##  [5,]  5.5
##  [6,]  6.7
##  [7,]  7.9
##  [8,]  9.1
##  [9,] 10.3
## [10,] 11.5
## [11,] 12.7
## [12,] 13.9
## [13,] 15.1
##
## [[7]]
## NULL
##
## [[8]]
## NULL
```

```
##
## [[9]]
##       [,1]
##  [1,]  0.7
##  [2,]  1.9
##  [3,]  3.1
##  [4,]  4.3
##  [5,]  5.5
##  [6,]  6.7
##  [7,]  7.9
##  [8,]  9.1
##  [9,] 10.3
## [10,] 11.5
## [11,] 12.7
## [12,] 13.9
## [13,] 15.1
##
## [[10]]
##      [,1]
## [1,]  0.7
## [2,]  1.9
##
## [[11]]
##      [,1]
## [1,]  0.7
## [2,]  1.9
## [3,]  3.1
## [4,]  4.3
## [5,]  5.5
## [6,]  6.7
## [7,]  7.9
## [8,]  9.1
##
## [[12]]
##      [,1]
## [1,]  0.7
## [2,]  1.9
## [3,]  3.1
## [4,]  4.3
```

We can see that there is a balanced number of 'edible' and 'poisonous' outcomes. Cap diameter (continuous variable measured in cm) shows a peak in very large caps (6 cm). There were 7 cap shapes with convex shape being most frequent, followed by flat. Out of the 12 cap color possible, brown was by far the most common color, follow by yellow and white.

We can also see that there are roughly 5 times more mushrooms that do not bruise or bleed than those that do. The most frequent gill color (gill is the structure underneath the mushroom cap) was white, followed by brown and yellow. Stem height (continuous variable measured in cm) shows a peak at very small sizes (less than 0.5 cm) and then between 4 and 5.5 cm. Stem width (continuous variable measured in mm) shows that larger values between 5 and 6 mm were most common.

Of the 13 possible stem colors, the most common colors were white and brown. There were roughly three more times mushrooms in the dataset that did not have a ring versus those that had a ring (this is a structure found on the stem of mature mushrooms). Not surprisingly, the most common habitat was woods, and most common seasons were summer and autumn.

4. Testing models

In an initial attempt to build models with 11 predictors, I observed very long computational times. Thus, I looked at methods of trimming down the number of predictors (i.e., feature selection). I found that the Recursive Feature Elimination (RFE) method, which uses a simple backwards selection algorithm, may work for this type of dataset. However, it did not work well for my computer due to a large computational time (>60 min). As an alternative, I used variable importance after selecting the best model.

(This is the code I tried to use for RFE:

#>control <- rfeControl(functions=rfFuncs, method="cv", number=10)

#>results <- rfe(mush[,2:12], mush[,1], sizes=c(5:7), rfeControl=control)

#>print(results)

#>predictors(results)

#>plot(results, type=c("g", "o")))

Next, I created the training and test sets. I initially chose an 80 - 20 partition between training and test sets, but, again, computational times for each model I tested were long, so I decided to use a 50-50 partition and this worked better, without significantly reducing the accuracy of the models. I checked the results of the partition using str() and dim().

```
set.seed(1, sample.kind="Rounding")
index <- createDataPartition(y = mush$class, times = 1, p = 0.5, list = FALSE)
mush_train <- mush[-index,]
mush_test <- mush[index,]

str(mush_train)
```

```
## 'data.frame':    30534 obs. of  12 variables:
##  $ class               : Factor w/ 2 levels "edible","poisonous": 2 2 2 2 2 2 2 2 2 2 ...
##  $ cap.diameter        : num  16.6 14.1 14.2 14.6 14.8 ...
##  $ cap.shape           : Factor w/ 7 levels "bell","conical",..: 3 3 4 3 4 3 3 3 3 3 ...
##  $ cap.color           : Factor w/ 12 levels "brown","buff",..: 11 11 7 11 11 11 11 7 7 7 ...
##  $ does.bruise.or.bleed: Factor w/ 2 levels "bruises-or-bleeding",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ gill.color          : Factor w/ 13 levels "brown","buff",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ stem.height         : num  18 17.8 15.8 16.5 17.7 ...
##  $ stem.width          : num  18.2 17.7 16 17.2 16.9 ...
##  $ stem.color          : Factor w/ 13 levels "brown","buff",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ has.ring            : Factor w/ 2 levels "ring","none": 1 1 1 1 1 1 1 1 1 1 ...
##  $ habitat             : Factor w/ 8 levels "grasses","leaves",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ season              : Factor w/ 4 levels "spring","summer",..: 2 4 4 4 4 2 2 3 2 4 ...
```

```
dim(mush_train)
```

```
## [1] 30534    12
```

```
str(mush_test)
```

```
## 'data.frame':    30535 obs. of  12 variables:
##  $ class               : Factor w/ 2 levels "edible","poisonous": 2 2 2 2 2 2 2 2 2 2 ...
##  $ cap.diameter        : num  15.3 15.3 14.9 12.8 13.6 ...
##  $ cap.shape           : Factor w/ 7 levels "bell","conical",..: 3 3 3 4 4 4 3 3 4 3 ...
##  $ cap.color           : Factor w/ 12 levels "brown","buff",..: 11 11 7 11 7 7 11 7 7 11 ...
##  $ does.bruise.or.bleed: Factor w/ 2 levels "bruises-or-bleeding",..: 2 2 2 2 2 2 2 2 2 2 ...
```

```
##  $ gill.color          : Factor w/ 13 levels "brown","buff",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ stem.height         : num  16.9 17.8 17 17.3 16 ...
##  $ stem.width          : num  17.1 18.8 17.4 18.7 16.9 ...
##  $ stem.color          : Factor w/ 13 levels "brown","buff",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ has.ring            : Factor w/ 2 levels "ring","none": 1 1 1 1 1 1 1 1 1 1 ...
##  $ habitat             : Factor w/ 8 levels "grasses","leaves",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ season              : Factor w/ 4 levels "spring","summer",..: 4 2 2 3 4 3 2 3 3 3 ...
```

```
dim(mush_test)
```

```
## [1] 30535    12
```

In the next step, I tested several models, including logistic, k nearest neighbor, classification tree, random forest, and ensemble. Methods such as LDA, QDA, and Loess of which we learned in the course, were not appropriate because the mushroom dataset includes non-numeric (i.e., nominal) predictors.

Model 1 - Logistic regression

I started by testing the simplest model, logistic regression with glm function in caret::train.

```
train_glm <- train(mush_train[,2:12], mush_train[,1], method = "glm")
glm_preds <- predict(train_glm, mush_test[,2:12])
mean(glm_preds == mush_test[,1])  #accuracy
```

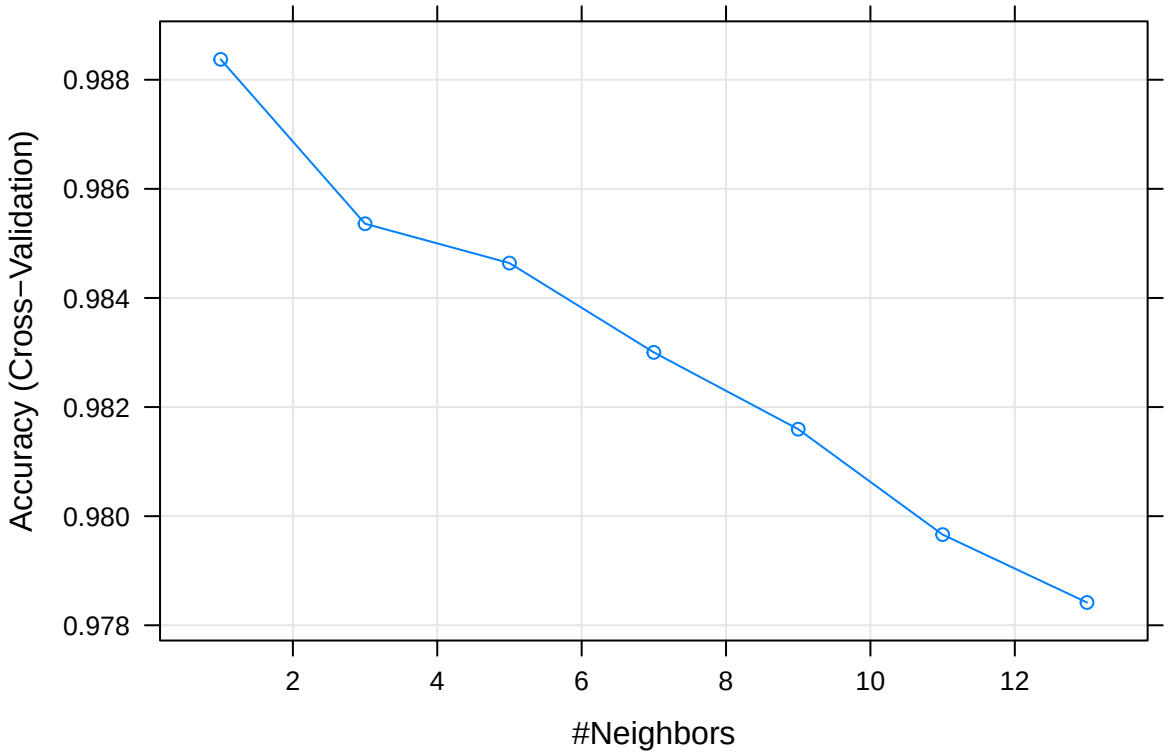
```
## [1] 0.731685
```

The accuracy was 0.731685. This was not great, but it was a starting point.

Model 2 - K-nearest neighbor (knn)

The second model was based on knn. I used cross-validation (method = "cv") with number = 10 (this tells the algorithm to break the set into 10-folds for cross-validation). k is the number of neighbors (this parameter can be tuned). tuneGrid tells which values the main parameter will take. I plotted the model accuracy as a function of k and then calculated the predictions and the final accuracy of the best knn model.

```
control <- trainControl(method = "cv", number = 10, p =.9)
train_knn <- train(class ~ ., method = "knn", data = mush_train,
                   tuneGrid = data.frame(k = seq(1, 13, by = 2)),
                   trControl = control)
plot(train_knn)
```




```
train_knn$bestTune
```

```
##   k
## 1 1
```

```
knn_preds <- predict(train_knn, mush_test)
confusionMatrix(knn_preds, mush_test$class)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9901097
```

The accuracy of the knn model was 0.9901097, quite an improvement over logistic regression.

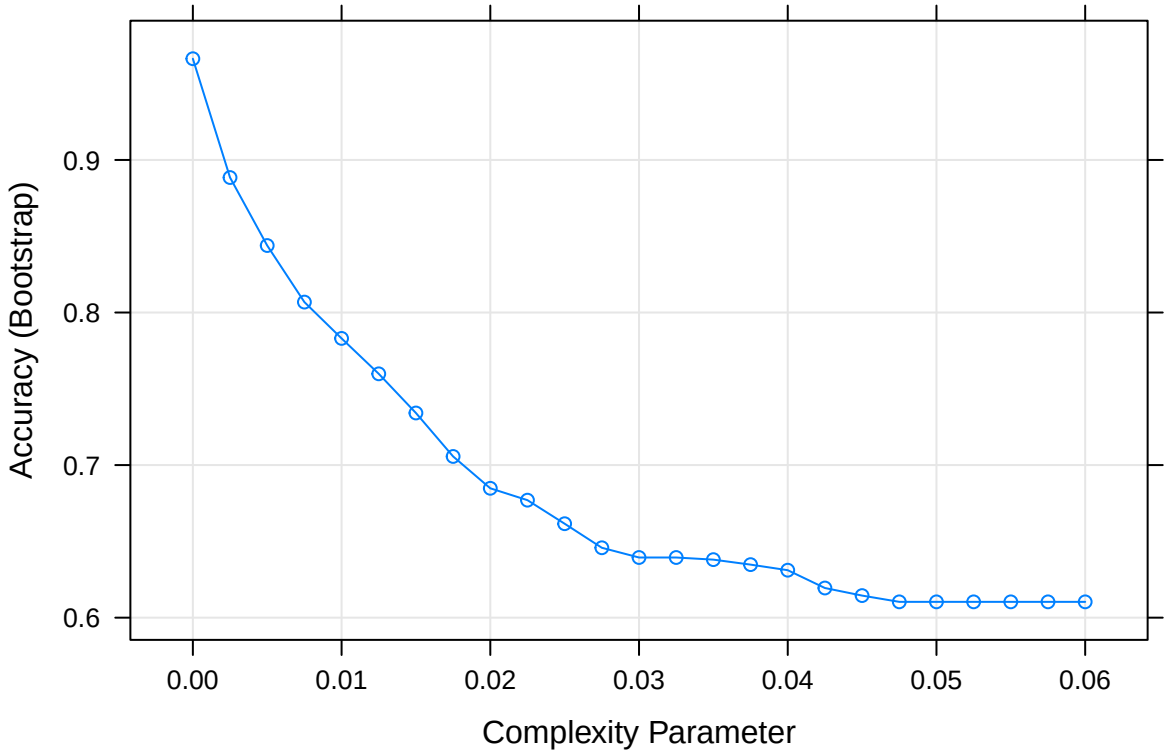
Model 3 - Classification tree - rpart

The third model was based on rpart. The tuning parameter here was the Complexity Parameter (cp) which tells the model that any split that does not decrease the overall lack of fit by a factor of cp is not attempted. I plotted the model accuracy as a function of cp.

```
train_rpart <- train(class ~ .,
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0.0, 0.06, len = 25)),
                     data = mush_train)
train_rpart$bestTune
```

```
##   cp
## 1  0
```

```
plot(train_rpart)
```




```
rpart_preds <- predict(train_rpart, mush_test)
confusionMatrix(rpart_preds, mush_test$class)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9697396
```

rpart model accuracy was 0.9697396.

Model 4 - Random forest using "Rborist" package

The fourth model used the newer package "Rborist" developed for random forests. Parameters that can be tuned included: nTree - the number of trees to grow, nSamp - number of rows to sample, per tree, minNode - parameter that represents the minimum number of distinct row references to split a node, and predFixed - the number of trial predictors for a split (same as mtry in the next model).

```
control <-trainControl(method = "cv", number = 10, p =0.8)
grid <- expand.grid(minNode = c(1,2,3,4,5), predFixed = c(1,3,5,7,9,15,25))
train_rf_1 <- train(class ~ .,
                    method = "Rborist", nTree = 100, trControl = control,
```

```
                tuneGrid = grid,
                data = mush_train,
                nSamp = 2000)
train_rf_1$bestTune
```

```
##    predFixed minNode
## 17         5       3
```

```
fit_rf_1 <- Rborist(mush_train[,2:12], mush_train[,1],
                 nTree = 1000,
                 minNode =train_rf_1$bestTune$minNode,
                 predFixed = train_rf_1$bestTune$predFixed)
rf_1_preds <- predict(train_rf_1,  mush_test)
confusionMatrix(rf_1_preds, mush_test$class)$overall["Accuracy"]
```

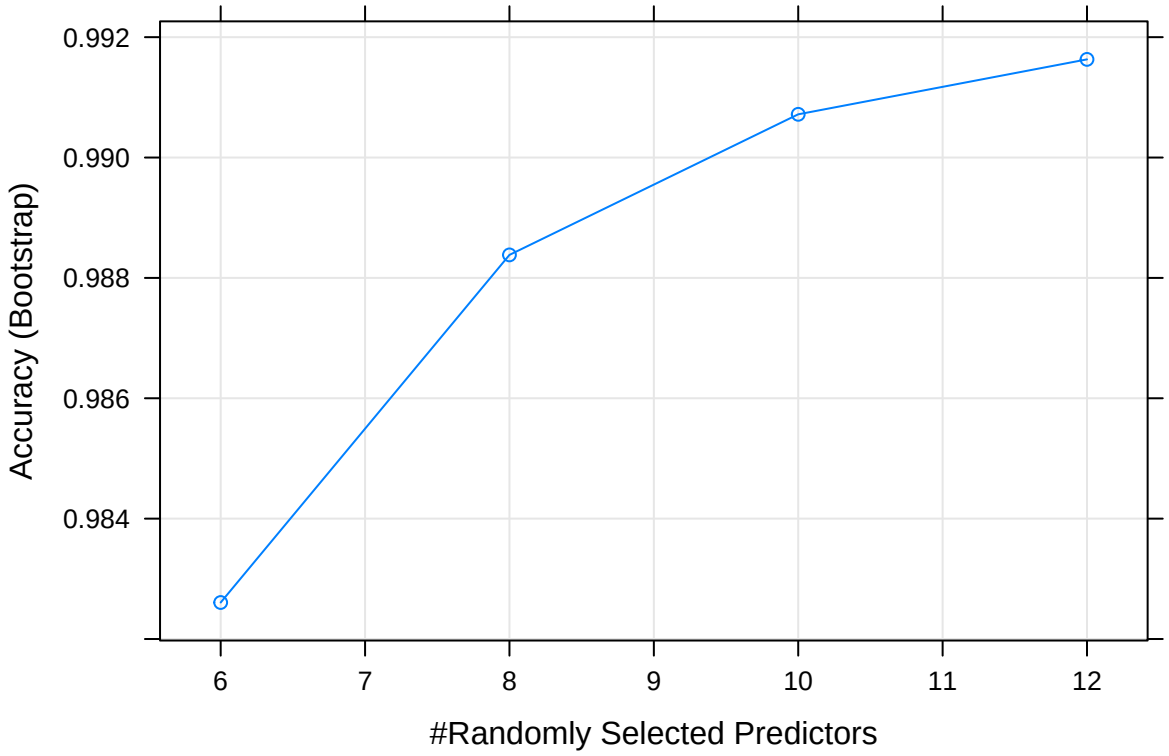
```
##  Accuracy
## 0.9835271
```

Random forest (Rborist) accuracy was 0.9835271, higher than the one from the rpart model but lower than the knn model accuracy.

Model 5 - Random forest using "caret::train" package

For this fourth model, mtry is a tuning parameter (same as predFixed in the previous model) and represents the number of variables randomly sampled as candidates at each split. ntree is the number of trees to grow in the forest and it cannot be tuned, it is just set fixed. However, the larger the ntree, the more accurate the model. It also means that the larger ntree, the longer it takes to run the model.

```
train_rf_2 <- train(class ~ .,
                    method = "rf", ntree = 100,
                    tuneGrid = data.frame(mtry =  seq(6, 12, 2)),
                    data = mush_train)
plot(train_rf_2)
```




```
rf_2_preds <- predict(train_rf_2, mush_test)
confusionMatrix(rf_2_preds, mush_test$class)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9934829
```

Random forest (train -rf) accuracy was 0.993829.

Model 6 - Ensemble

This sixth model is an ensemble, which means that it takes several models and combines them to produce an outcome with the hope that it reduces the error and thus increases accuracy.

```
ensemble <- cbind(glm = glm_preds == "edible",
                  knn = knn_preds == "edible",
                  rpart = rpart_preds == "edible",
                  rf_1 = rf_1_preds == "edible",
                  rf_2 = rf_2_preds == "edible")

ensemble_preds <- ifelse(rowMeans(ensemble) > 0.5, "edible", "poisonous")
mean(ensemble_preds == mush_test[,1]) #accuracy
```

```
## [1] 0.9912232
```

Ensemble accuracy was 0.9912232. This is good, but not the best.

III. Modeling Results

I summarized the results from running these six models in a table, for easier examination.

```
models <- c("Logistic regression", "K nearest neighbors", "Classification tree",
"Random forest-Rborist", "Random forest-rf", "Ensemble")
accuracy <- c(mean(glm_preds == mush_test[,1]),
              mean(knn_preds == mush_test[,1]),
              mean(rpart_preds == mush_test[,1]),
              mean(rf_1_preds == mush_test[,1]),
              mean(rf_2_preds == mush_test[,1]),
              mean(ensemble_preds == mush_test[,1]))
data.frame(Model = models, Accuracy = accuracy)
```

```
##                   Model  Accuracy
## 1   Logistic regression 0.7316850
## 2   K nearest neighbors 0.9901097
## 3   Classification tree 0.9697396
## 4 Random forest-Rborist 0.9835271
## 5      Random forest-rf 0.9934829
## 6              Ensemble 0.9912232
```

We can see that there are three models with accuracy above 0.99. The best model is based on the Random Forest (rf_2 method) and has a prediction accuracy of 0.9935. I then looked at additional details for this model, such as sensitivity and specificity:

```
confusionMatrix(rf_2_preds, mush_test$class)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  edible poisonous
##   edible     13479        87
##   poisonous    112     16857
##
##                Accuracy : 0.9935
##                  95% CI : (0.9925, 0.9944)
##     No Information Rate : 0.5549
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9868
##
##  Mcnemar's Test P-Value : 0.08888
##
##             Sensitivity : 0.9918
##             Specificity : 0.9949
##          Pos Pred Value : 0.9936
##          Neg Pred Value : 0.9934
##              Prevalence : 0.4451
##          Detection Rate : 0.4414
##    Detection Prevalence : 0.4443
##       Balanced Accuracy : 0.9933
##
##        'Positive' Class : edible
##
```

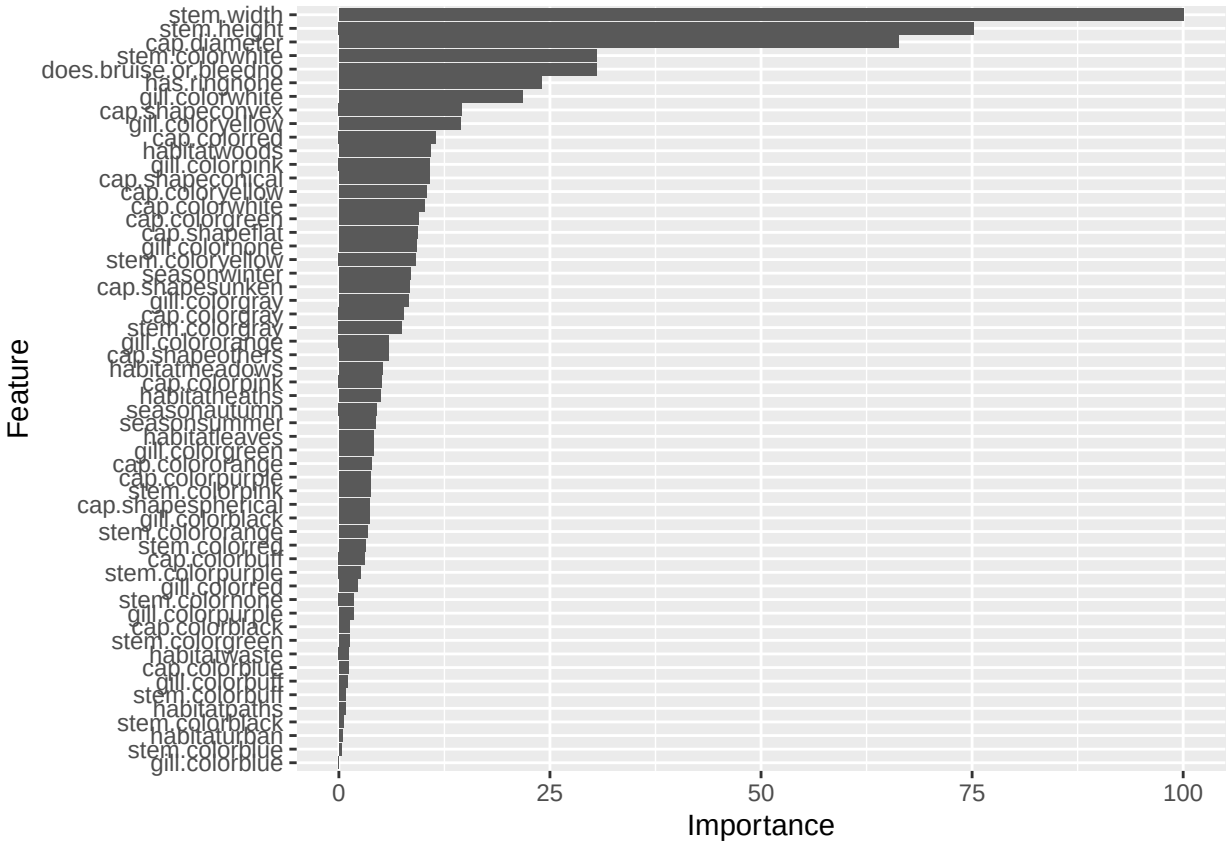
With both Sensitivity and Specificity above 0.99, I got another indication that this model performs quite well.

With this method, however, a decision tree cannot be build to visualize the classification rules because the results come from a forest made up of many trees. One way to get an insight on the contribution of each predictor to the model is by looking at Variable Importance scores, which quantify the impact of the predictor rather than a specific effect, such as how a variable responds to the variation in another variable.

```
imp <- varImp(train_rf_2)
print(imp)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 56)
##
##                          Overall
## stem.width               100.000
## stem.height               75.224
## cap.diameter              66.271
## stem.colorwhite           30.517
## does.bruise.or.bleedno    30.475
## has.ringnone              23.986
## gill.colorwhite           21.736
## cap.shapeconvex           14.596
## gill.coloryellow          14.430
## cap.colorred              11.515
## habitatwoods              10.823
## gill.colorpink            10.796
## cap.shapeconical          10.710
## cap.coloryellow           10.419
## cap.colorwhite            10.122
## cap.colorgreen             9.398
## cap.shapeflat              9.313
## gill.colornone             9.194
## stem.coloryellow           9.146
## seasonwinter               8.458
```

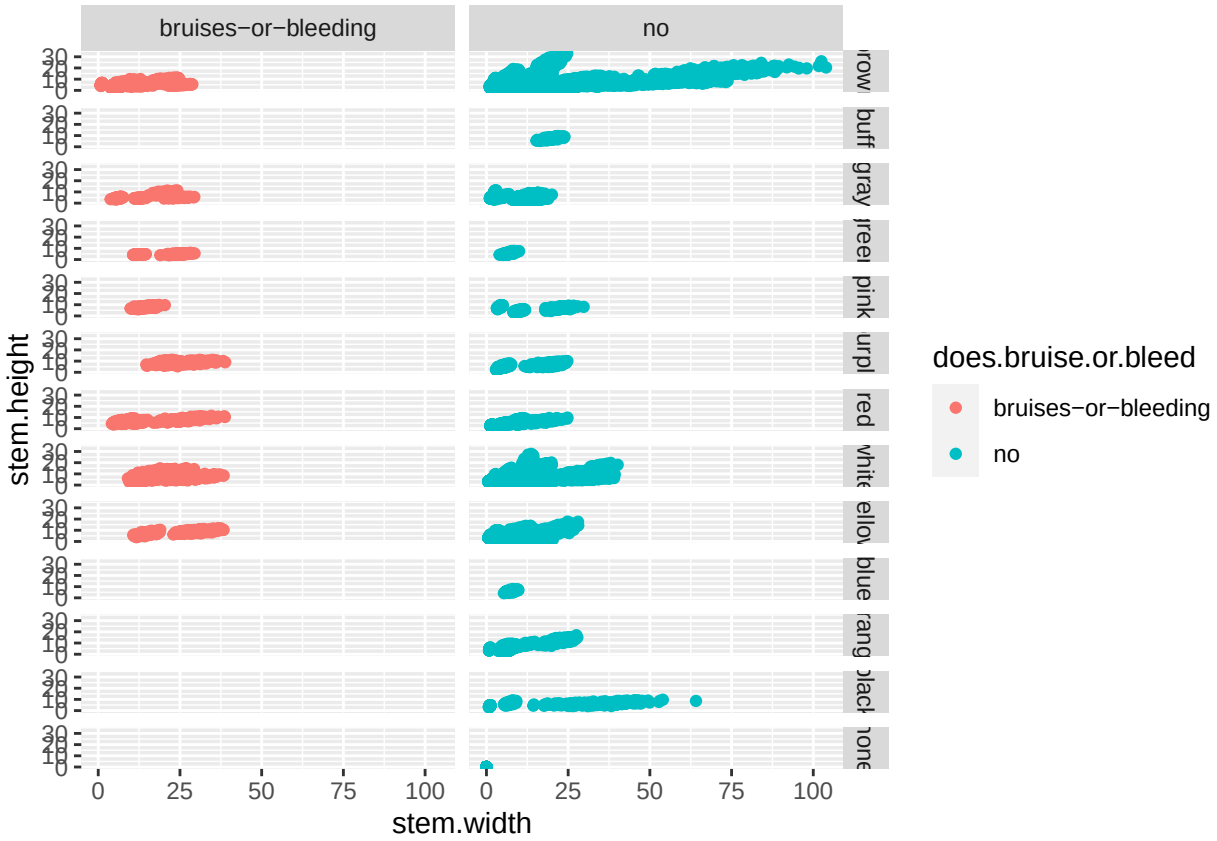
```
ggplot(imp)
```



Most imp variables were: stem.width, stem.height, cap.diameter, (all above 66%), and then does.bruise.or.bleed and stem color. Next, I rerun the best rf_2 model with these five most important variables and the accuracy was 0.9160635, still fair (although, when it comes to eating mushrooms I would rather be 99% accurate in predicting that a mushroom is not poisonous rather than 92%). For exploration and visualization purposes, I considered this an acceptable accuracy. Thus, for the last step I created a few graphs using these five most important predictors and the outcome variable "class" and using the entire 'mush' dataset.

First graph looked at stem height, stem width, bruising & bleeding, and stem color.

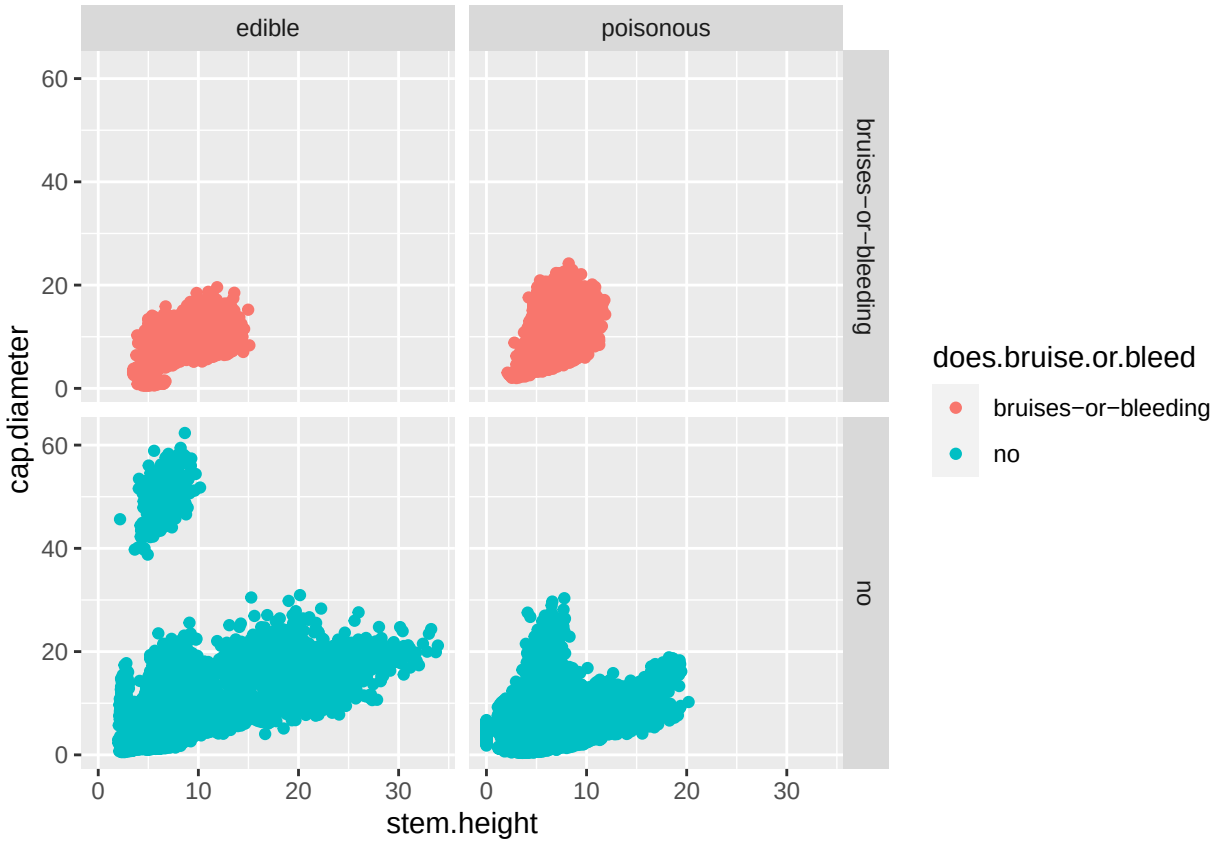
```
mush %>% group_by(stem.color) %>%
  ggplot(aes(stem.width, stem.height, col = does.bruise.or.bleed )) +
         geom_point() +facet_grid(stem.color ~ does.bruise.or.bleed)
```



This first set of scatterplots suggests there may be a difference in stem width between mushrooms that bruise or bleed and those that don't, with the latter being shorter and also not getting as tall with a stem width increase. There is however a lot more variation in the stem height as stem width increases in the mushrooms that do not bruise or bleed, especially those with brown, white, or yellow stems. It seems that mushrooms with brown stems can get wide and tall and are less likely to bruise.

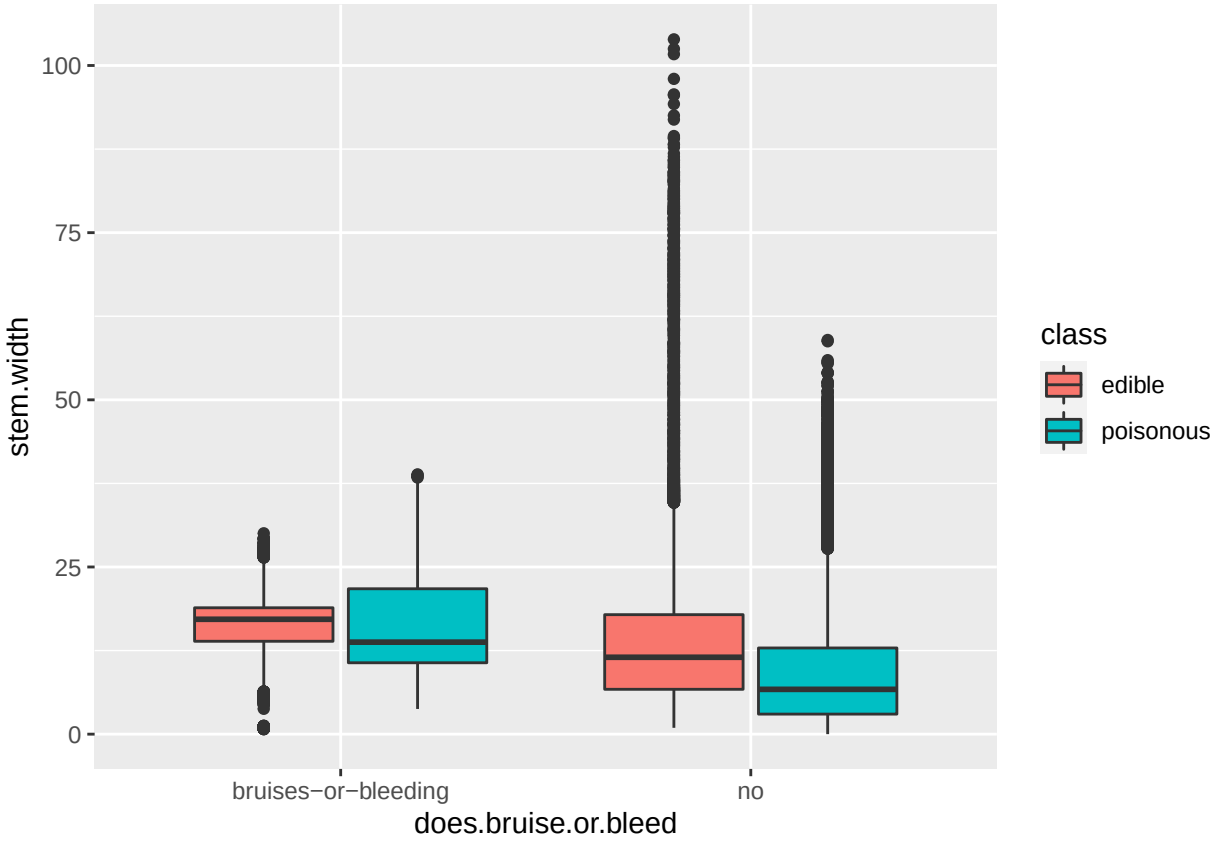
The second graph looked at stem height, cap diameter, bruising & bleeding, and edibility.

```
mush %>% group_by(class) %>%
 ggplot(aes(stem.height, cap.diameter, col = does.bruise.or.bleed )) +
 geom_point() +facet_grid( does.bruise.or.bleed ~ class)
```



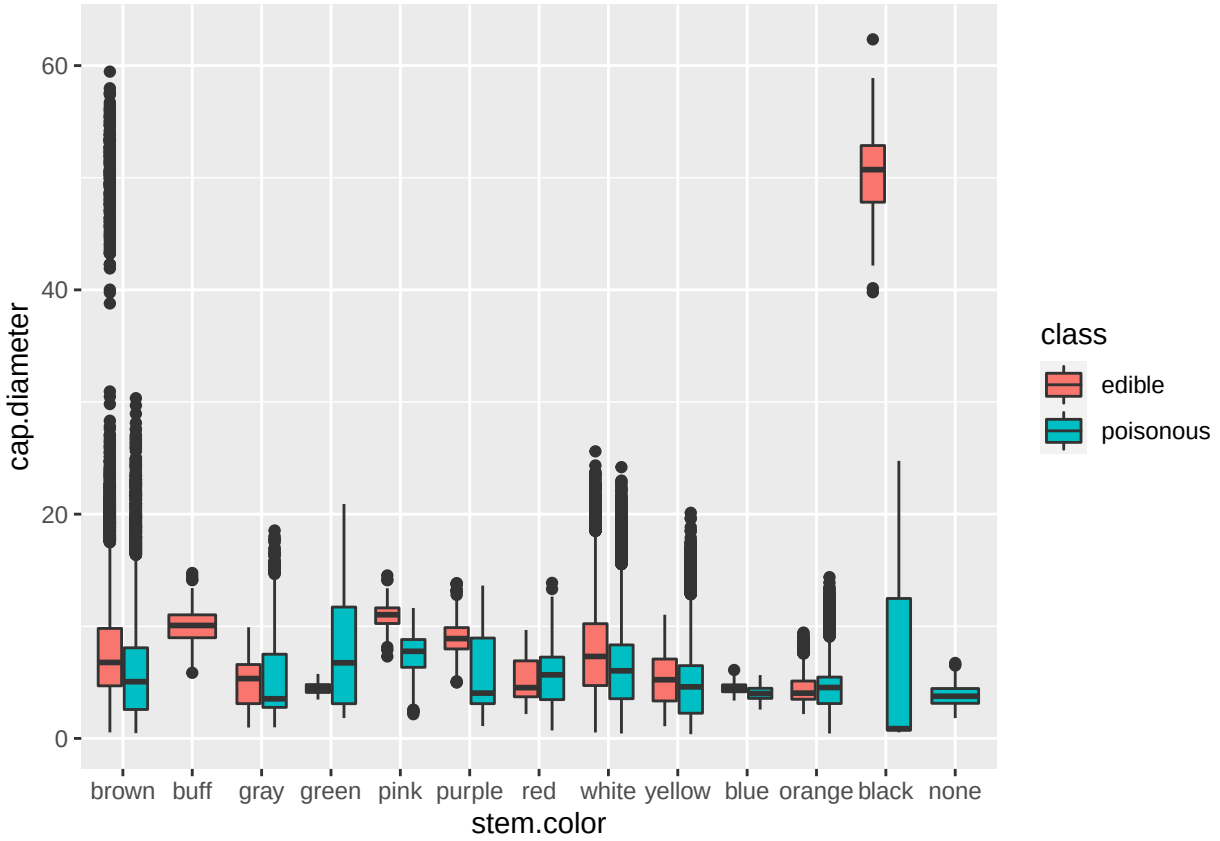
This second graph, a series of scatterplots, shows an interesting pattern for edible mushrooms that do not bruise or bleed as they seem to include some of the largest mushrooms in terms of height but also two separate groups, one of smaller cap diameter (up to about 30 mm) and one of larger cap sizes (between 40 and 60 mm). We can also see that mushrooms that do bruise or bleed tend to be smaller in height and cap diameter, regardless of edibility.

```
mush %>% ggplot(aes(does.bruise.or.bleed, stem.width, fill = class)) +
  geom_boxplot()
```



The third graph, a boxplot, compares edible and poisonous mushrooms by stem width and whether they bruise/bleed or not. Mushrooms that do not bruise/bleed can have large stems, although the median is actually lower for both edible and especially for poisonous mushrooms compared to those that bruise/bleed.

```
mush %>% ggplot(aes(stem.color, cap.diameter, fill = class)) +
  geom_boxplot()
```



This last graph, also a boxplot, compares edible and poisonous mushrooms by stem color and cap diameter. We can see again that brown is a frequent stem color and that edible mushrooms can have larger cap diameter. Note also that poisonous mushrooms don't have buff stem color, and if you find a mushroom with black stem, it is likely edible only if it has a large cap.

IV. Conclusions

I selected a classification problem with the goal of building a model predicting whether a mushroom is edible or poisonous. I built the model based on three numerical characteristics and eight nominal (categorical) variables. My best model was based on a random forest algorithm and achieved a 99.4% accuracy. This is quite an important achievement with practical implications in the context of determining if consuming an individual object in the set (in this case, a mushroom) will result in poisoning, or even death.

I also realize that there are limitations to the work I presented here. First, finding a way to include or explore all 20 initial predictor variables (assuming access to a more powerful computer or faster algorithms) would be beneficial. With a more powerful computer I would also like to perform the RFE method of feature selection and redo the work with an 80-20 split between training and test sets. For a future iteration, I would like to learn more about normalization as well as about converting categories into numeric variables, maybe trying a method called one-hot encoding. Also I would look into dummy coding.

I much enjoyed the challenge and the process of building the model and writing about it.

V. References

Dataset download - https://archive.ics.uci.edu/ml/datasets/Secondary+Mushroom+Dataset#

Feature selection - https://machinelearningmastery.com/feature-selection-with-the-caret-r-package/ http://topepo.github.io/caret/recursive-feature-elimination.html#rfe

Random Forest - https://machinelearningmastery.com/tune-machine-learning-algorithms-in-r/#:~:text=mtry%3A%20Number%20of%20variables%20randomly,Number%20of%20trees%20to%20grow.