Data Science ‘Mushrooms’ Capstone Project

András Gelencsér

14 December 2019

Overview

This document describes the result of the capstone project for the HarvardX Data Science course based on the Mushrooms dataset avaiable in the UCI Machine Learning Repository (Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.)

The “Mushrooms” dataset was recorded down from The Audubon Society Field Guide to North American Mushrooms (1981). G. H. Lincoff (Pres.), New York: Alfred A. Knopf and is avaiable on the https://archive.ics.uci.edu/ml/datasets/mushroom url.

The goal of this project is to develop a model based on data science and machine learning techniques to predict if a mushroom is eatable based on the known attributes.

The dataset contains 8124 records and uses 22 attributes to describe the mushrooms. Each dataset contains additional attribute to flag if the mushroom is eatable or not.

We will use 80% of the avaiable data for train our model. The reamining 20% will be used as test dataset for evaluating the developed model.

For the developed model is very important to detect poisonus mushrooms to avoid possible health damages or death. Therefore we will evluate the model performance with the F1 score instead of the overal accuracy, beacuse it’s very important to reach high sensitivity with our model.

The following steps were done during the project for developing the algorithm:

1. Download the data and prepare for the analysis
2. Analyze the data structure
3. Define the classification model based on the result the data analysis
4. Implement and train the model based on the train set
5. Review the model based on the test set

The final model performed an F1 score of 1 by using a predicton based on the 9th most important features.

Methods & analysis

Preparation

For the analysis the data we will use the following R packages:

- tidyverse
- caret
- gridExtra
- ggplot2

the following R code loads and install the required packages if needed:

```r
if(!require(tidyverse)) install.packages("tidyverse",
                          repos = "http://cran.us.r-project.org", dependencies = TRUE)
if(!require(caret)) install.packages("caret",
                          repos = "http://cran.us.r-project.org", dependencies = TRUE)
if(!require(gridExtra)) install.packages("gridExtra",
                          repos = "http://cran.us.r-project.org", dependencies = TRUE)
if(!require(ggplot2)) install.packages("ggplot2",
                          repos = "http://cran.us.r-project.org", dependencies = TRUE)
if(!require(Rborist)) install.packages("Rborist",
                          repos = "http://cran.us.r-project.org", dependencies = TRUE)
```

At first, we need to prepare the data set for the analysis. The mushrooms dataseet contains 8,124 records about different mushrooms.

The following R code downloads the data from the https://archive.ics.uci.edu/ml/datasets/mushroom site. To avoid unnecessary data traffic, the data will be downloaded only if not done yet.

```
#create directory for the data file if necessary
if (!dir.exists("mushrooms")){
  dir.create("mushrooms")
}
baseurl = "https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/"
files_to_download = c("agaricus-lepiota.data", "agaricus-lepiota.names")

for (f in files_to_download){
  #download the files only if not done yet
  if (!file.exists(paste("./mushrooms/",f,sep='')))
  {
    download.file(paste(baseurl,f,sep=''), paste("./mushrooms/",f,sep=''))
  }
}

fl <- file("mushrooms/agaricus-lepiota.data")
mushroom_data <- str_split_fixed(readLines(fl), ",", 23)
close(fl)
```

The dataset is stored in the “agaricus-lepiota.data” file. This file contains 23 columns separated via comma. There is no header line in the file. The description file “agaricus-lepiota.names” gives us information about the classes and the 22 attributes in the data file.

So we can extract the data by parsing the data file. The following R code reads the data and stores in a data frame:

```
fl <- file("mushrooms/agaricus-lepiota.data")
mushroom_data <- str_split_fixed(readLines(fl), ",", 23)
close(fl)

#set the column names for the features
colnames(mushroom_data) <- 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")
#convert to data frame
mushroom_data <- as.data.frame(mushroom_data)

#remove temporary file variable
rm(fl)
```

For training the classification method we will use 80% of the avaiable data as training set. The remaining data will be used for evaluating the developed method. The following R code creates the train and test set:

```r
#initialize random sequenz
set.seed(1, sample.kind = "Rounding")
#create index for train and test set
#20% of the data will be used for the test set
test_idx = createDataPartition(y = mushroom_data$classes, times=1, p=0.2, list=FALSE)
train_data = mushroom_data[-test_idx,]
test_data = mushroom_data[test_idx,]
#remove temporary variables
rm(mushroom_data, test_idx)
```

Available Data

The dataset contains 8,124 records. For the analysis we have data about 6,500 mushrooms (80% of all data). All the features are categorical with different possible values.

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

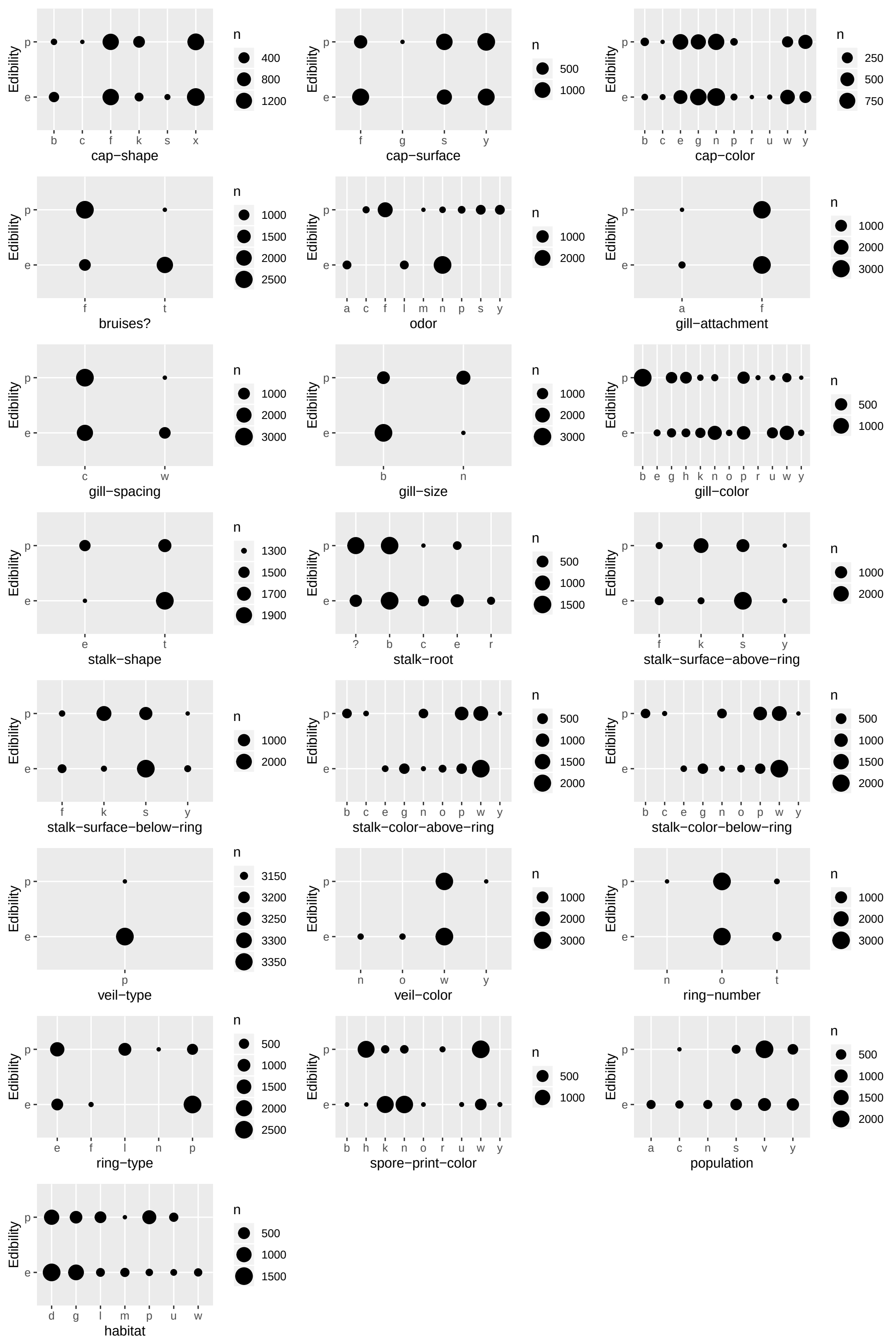
For testing the preformance the predition model we have about 1,600 mushrooms data (20% of all data) The dataset contains only the short 1 character code of the attribute values, but it's not disturbing for the model development.

Feature analysis

As first step we can investigate the distribution of the features corresponding to the edibility classification.

The following code plots this distribution for each feature.

```r
feature_ditribution_plot <- function(data){
  plots <- list()
  #Plots all the attributes for eatable <-> poisious
  for (i in 1:(ncol(data)-1))
  {
    summarized_data <- data %>% group_by(classes, .[,i+1]) %>% summarise(n = n())
    names(summarized_data)[2] <- "attr"
    plot <- summarized_data %>% ggplot(aes(attr , classes)) + geom_point(aes(size=n)) +
      xlab(names(data)[i+1]) + ylab("Edibility")
    plots[[i]] <- plot
  }
  rm(summarized_data, i, plot)
  grid.arrange(grobs=plots,ncol=3)
}

feature_ditribution_plot(train_data)
```


Edibility
p
e
n
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

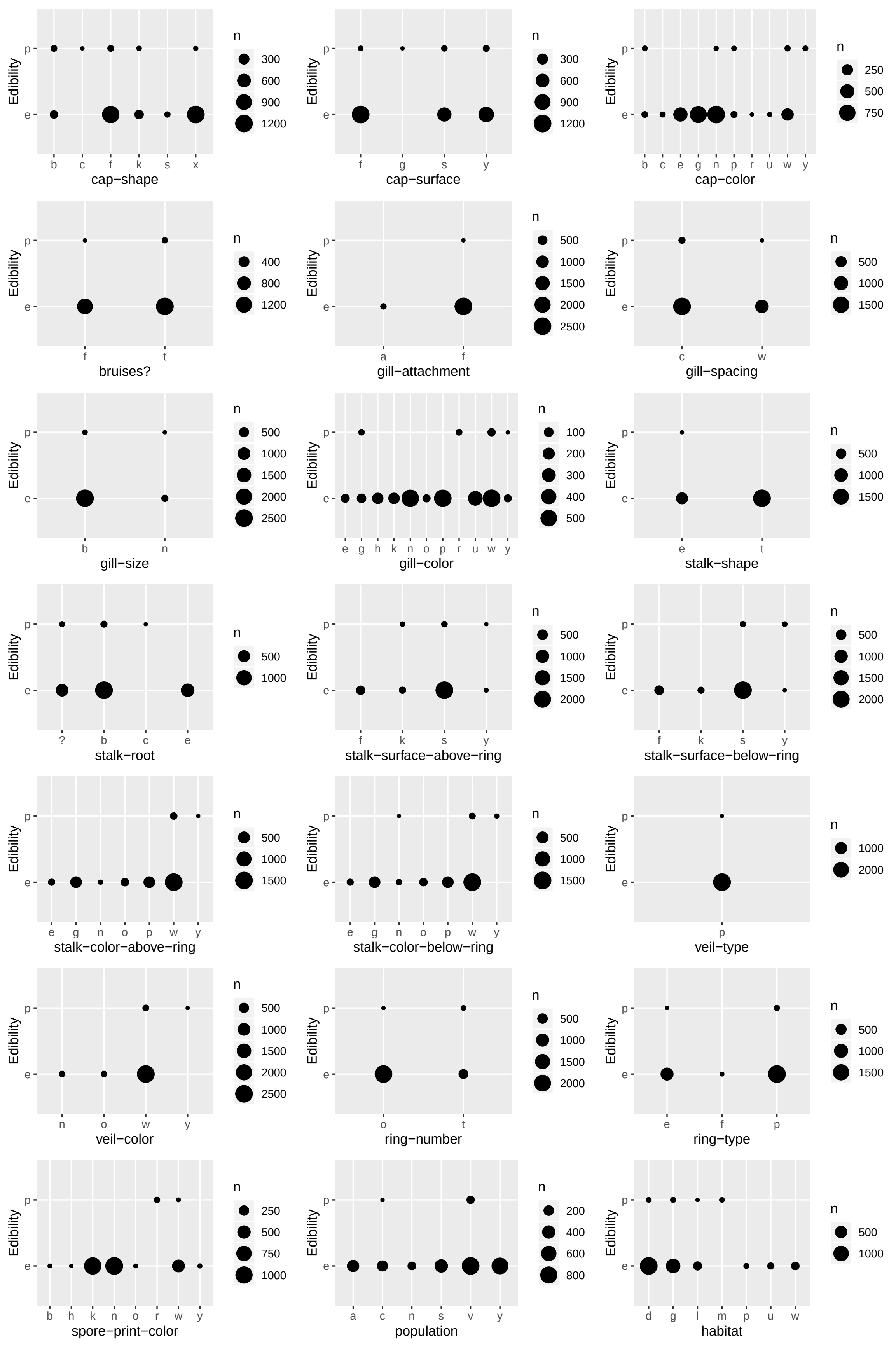
We can see, that the feature veil-type has only the value 'p' (partial), there is no data record with the veil-type value 'u' (universal). This feature can not provide decision information about the mushrooms, therefor we can remove this feature from the dataset.

On the other side, there are some features with values that can decide the classification. E.g. if we see a mushroom with the value 'f' (foul) for the feature odor, we can classify the mushroom as poisonous. The feature values 'c', 'f', 'l', 'm', 'p', 's', 'y' for the attribute odor can give us a classification. With this information we can classify 3,689 mushrooms. We can not classify only the mushrooms with the odor value 'n' (none). There are 2,809 mushrooms remaining in the dataset without classification.

We can try to analyse all the mushrooms, where the odor value is 'n'. For this analysis we can remove the attribute odor from the dataset, because we examine a specific odor value.

The following code removes the odor attribute and plots the feature distirbution for the remaining data:

```r
odor_n <- train_data %>% filter(`odor` == 'n') %>% select(-`odor`)
feature_ditribution_plot(odor_n)
```


Edibility
p
e
n
cap−shape
b c f k s x
300 600 900 1200
cap−surface
f g s y
300 600 900 1200
cap−color
b c e g n p r u w y
250 500 750
bruises?
f t
400 800 1200
gill−attachment
a f
500 1000 1500 2000 2500
gill−spacing
c w
500 1000 1500
gill−size
b n
500 1000 1500 2000 2500
gill−color
e g h k n o p r u w y
100 200 300 400 500
stalk−shape
e t
500 1000 1500
stalk−root
? b c e
500 1000
stalk−surface−above−ring
f k s y
500 1000 1500 2000
stalk−surface−below−ring
f k s y
500 1000 1500 2000
stalk−color−above−ring
e g n o p w y
500 1000 1500
stalk−color−below−ring
e g n o p w y
500 1000 1500
veil−type
p
1000 2000
veil−color
n o w y
500 1000 1500 2000 2500
ring−number
o t
500 1000 1500 2000
ring−type
e f p
500 1000 1500
spore−print−color
b h k n o r w y
250 500 750 1000
population
a c n s v y
200 400 600 800
habitat
d g l m p u w
500 1000

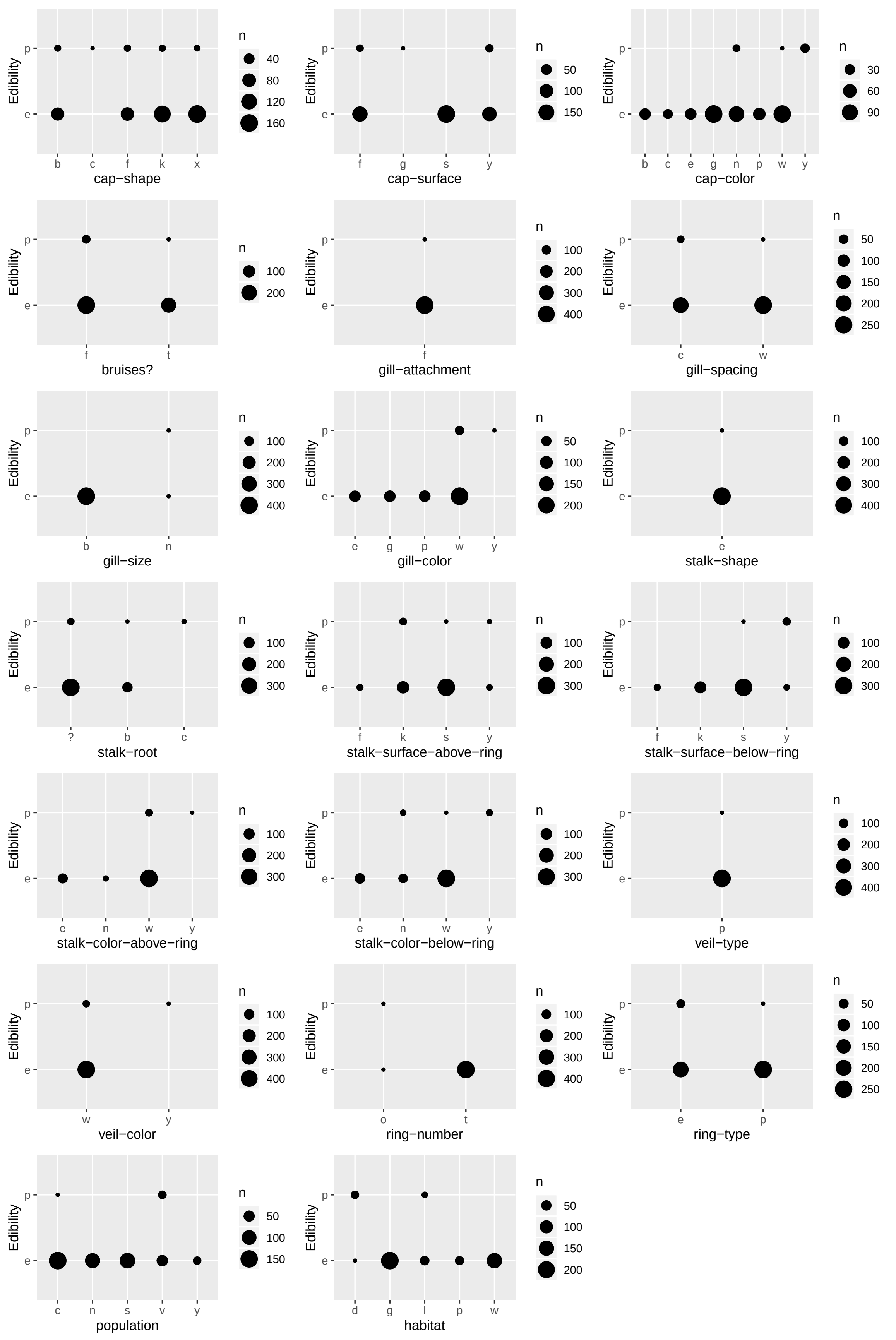
In the reduced data set we see again some features that we can use for classification the mushrooms.

E.g. if we see a mushroom with the value ‘k’ (black) for the feature spore-print-color, we can classify the mushroom as edible. The feature values ‘b’, ‘h’, ‘k’, ‘n’, ‘o’, ‘r’, ‘y’ for the feature spore-print-color can give us a classification for the remaining data set. We can not make a classification only it the spore-print-color value is ‘w’ (white).

We can reuse the same logik like for the feature odor and filter the remaing data with the spore-print-color feature value ‘w’.

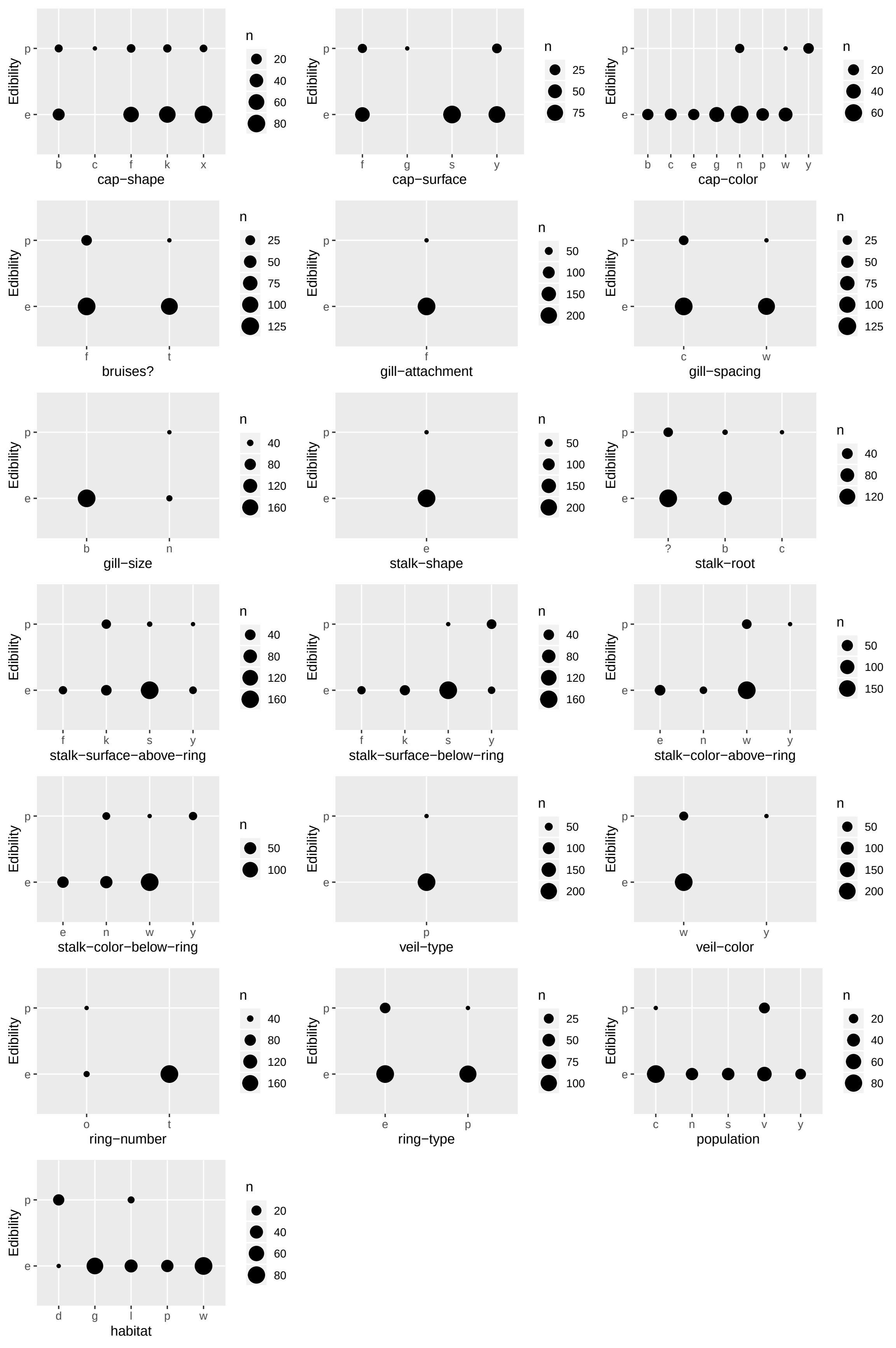
The following code removes the odor attribute and plots the feature distirbution for the remaining data:

```
spore_print_color_w <- odor_n %>% filter(`spore-print-color` == 'w') %>%
                       select(-`spore-print-color`)
feature_ditribution_plot(spore_print_color_w)
```


Edibility
p
e
n
cap−shape
b c f k x
40 80 120 160
cap−surface
f g s y
50 100 150
cap−color
b c e g n p w y
30 60 90
bruises?
f t
100 200
gill−attachment
f
100 200 300 400
gill−spacing
c w
50 100 150 200 250
gill−size
b n
100 200 300 400
gill−color
e g p w y
50 100 150 200
stalk−shape
e
100 200 300 400
stalk−root
? b c
100 200 300
stalk−surface−above−ring
f k s y
100 200 300
stalk−surface−below−ring
f k s y
100 200 300
stalk−color−above−ring
e n w y
100 200 300
stalk−color−below−ring
e n w y
100 200 300
veil−type
p
100 200 300 400
veil−color
w y
100 200 300 400
ring−number
o t
100 200 300 400
ring−type
e p
50 100 150 200 250
population
c n s v y
50 100 150
habitat
d g l p w
50 100 150 200

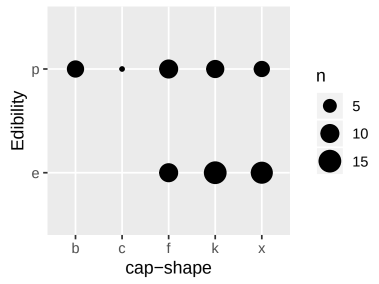
We can repeate this analysis and look for features that we can use for classifying the remaining data set. As next step we can get the gill-color feature and examine only the mushrooms with the grill color ‘w’ (white).

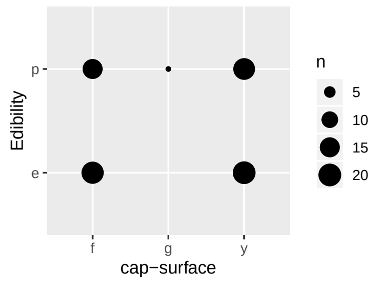
```r
gill_color_w <- spore_print_color_w %>% filter(`gill-color` == 'w') %>%
                select(-`gill-color`)
feature_ditribution_plot(gill_color_w)
```

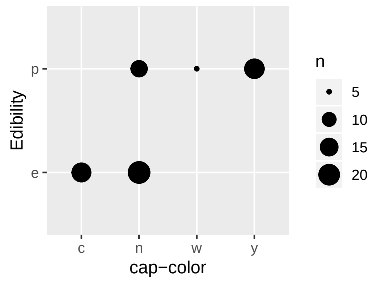

Edibility
p
e
n
cap−shape
b c f k x
20 40 60 80
cap−surface
f g s y
25 50 75
cap−color
b c e g n p w y
20 40 60
bruises?
f t
25 50 75 100 125
gill−attachment
f
50 100 150 200
gill−spacing
c w
25 50 75 100 125
gill−size
b n
40 80 120 160
stalk−shape
e
50 100 150 200
stalk−root
? b c
40 80 120
stalk−surface−above−ring
f k s y
40 80 120 160
stalk−surface−below−ring
f k s y
40 80 120 160
stalk−color−above−ring
e n w y
50 100 150
stalk−color−below−ring
e n w y
50 100
veil−type
p
50 100 150 200
veil−color
w y
50 100 150 200
ring−number
o t
40 80 120 160
ring−type
e p
25 50 75 100
population
c n s v y
20 40 60 80
habitat
d g l p w
20 40 60 80

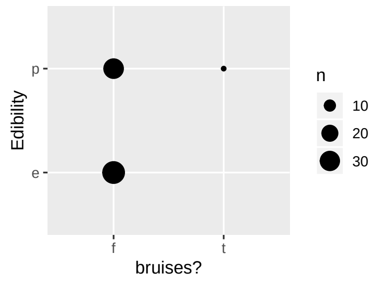
As next step we can get the gill-size feature and examine only the mushrooms with the grill size ‘n’ (narrow).

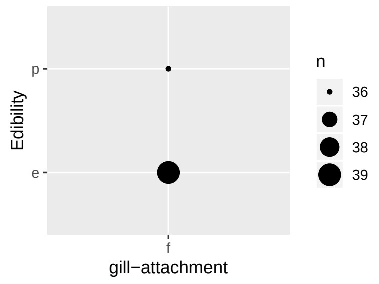
```r
gill_size_n <- gill_color_w %>% filter(`gill-size` == 'n') %>% select(-`gill-size`)
feature_ditribution_plot(gill_size_n)
```

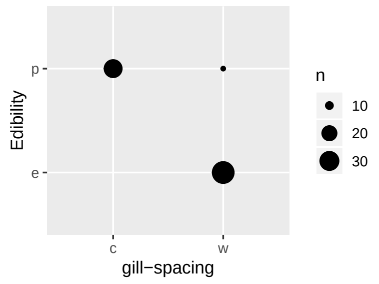

Edibility
p
e
b c f k x
cap−shape
n
5
10
15

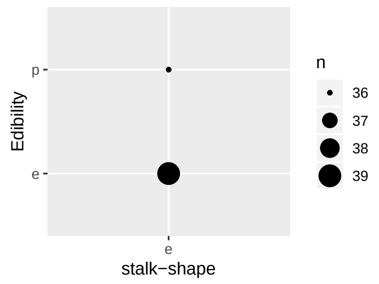

Edibility
p
e
f g y
cap−surface
n
5
10
15
20

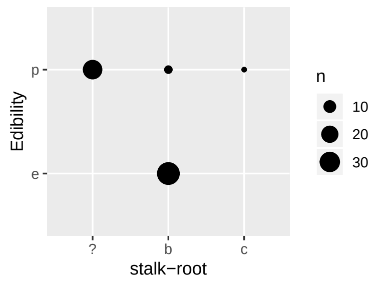

Edibility
p
e
c n w y
cap−color
n
5
10
15
20

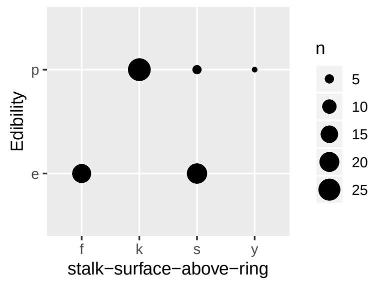

Edibility
p
e
f t
bruises?
n
10
20
30

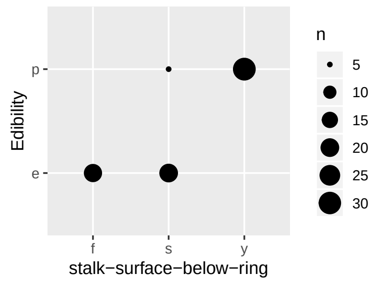

Edibility
p
e
f
gill−attachment
n
36
37
38
39

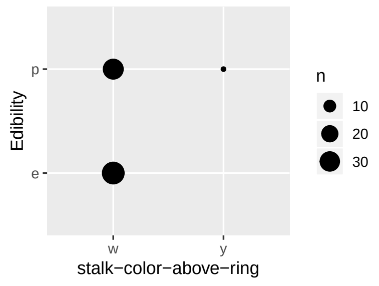

Edibility
p
e
c w
gill−spacing
n
10
20
30

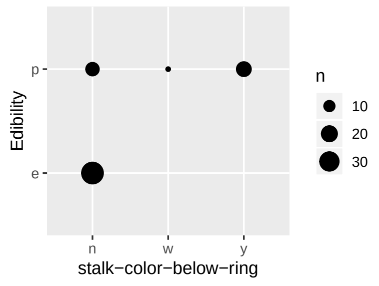

Edibility
p
e
e
stalk−shape
n
36
37
38
39

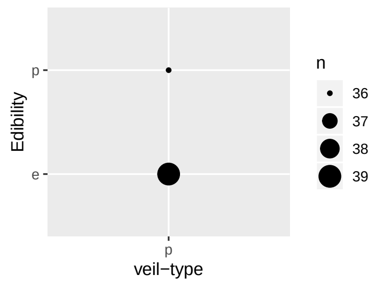

Edibility
p
e
? b c
stalk−root
n
10
20
30

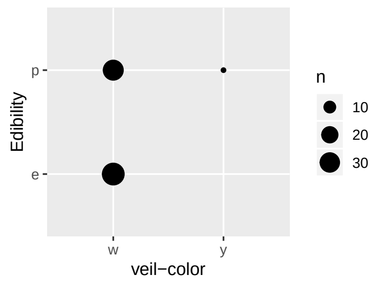

Edibility
p
e
f k s y
stalk−surface−above−ring
n
5
10
15
20
25

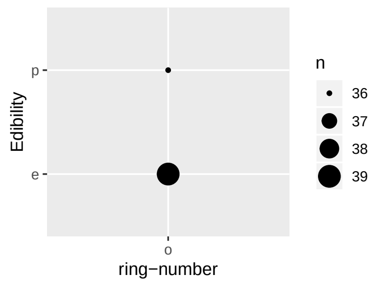

Edibility
p
e
f s y
stalk−surface−below−ring
n
5
10
15
20
25
30

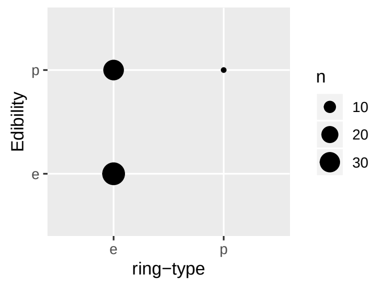

Edibility
p
e
w y
stalk−color−above−ring
n
10
20
30

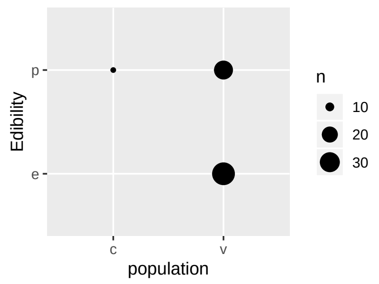

Edibility
p
e
n w y
stalk−color−below−ring
n
10
20
30

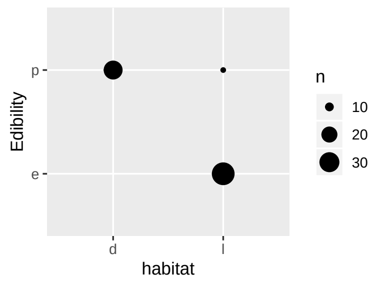

Edibility
p
e
p
veil−type
n
36
37
38
39


Edibility
p
e
w y
veil−color
n
10
20
30


Edibility
p
e
o
ring−number
n
36
37
38
39

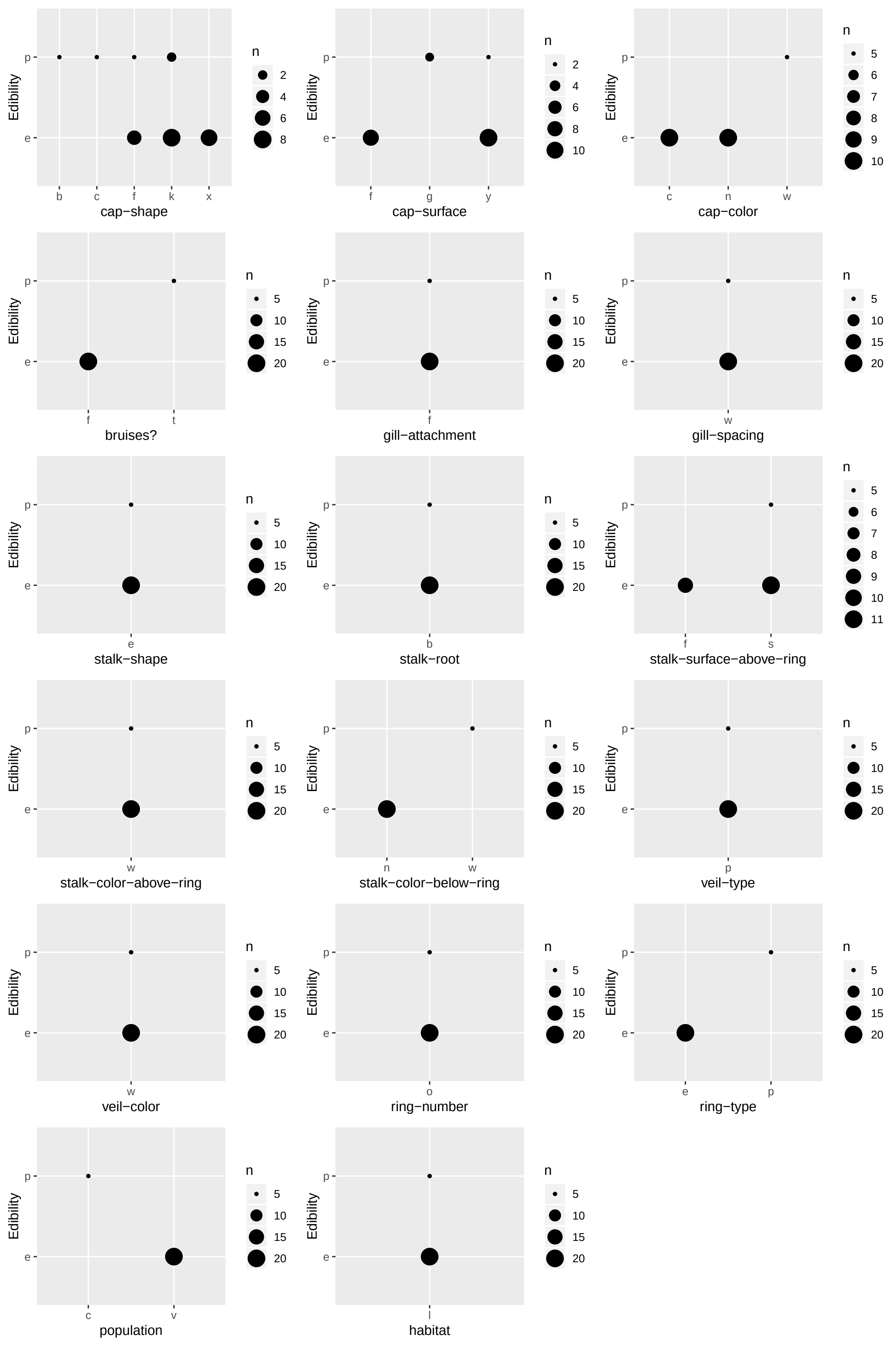

Edibility
p
e
e p
ring−type
n
10
20
30


Edibility
p
e
c v
population
n
10
20
30


Edibility
p
e
d l
habitat
n
10
20
30

As next step we can get the stalk-surface-below-ring feature and examine only the mushrooms with the stalk-surface-belowe-ring ‘s’ (smooth).

```
stalk_surface_below <- gill_size_n %>% filter(`stalk-surface-below-ring` == 's') %>%
                      select(-`stalk-surface-below-ring`)
feature_ditribution_plot(stalk_surface_below)
```


Edibility
p
e
n
b c f k x
cap−shape
2 4 6 8
f g y
cap−surface
2 4 6 8 10
c n w
cap−color
5 6 7 8 9 10
f t
bruises?
5 10 15 20
f
gill−attachment
w
gill−spacing
e
stalk−shape
b
stalk−root
f s
stalk−surface−above−ring
5 6 7 8 9 10 11
w
stalk−color−above−ring
n w
stalk−color−below−ring
p
veil−type
w
veil−color
o
ring−number
e p
ring−type
c v
population
l
habitat

The remaining dataset has only 25 mushrooms. As we can see on the plot, now we can use some feature for classifying this remaining mushrooms. E.g. the feature ring-type give us a good classification: if we have a ring type of 'e' (evanescent) than the mushroom is edible. On the other case, if we have a ring-type of 'p' (pedant), the mushroom is poisonous.

With the steps about we defined a decision tree for classifying all the mushrooms in the train set. We can use this information to develop a classification model for the mushrooms.

The analyse above give us an other important information: some fatures are more relevant for the classification some features gives no additional information for the method. That means, the different features have different information weight for the classification.

We have to try to calculate a correlation between the features and the edibility. For the unordered categorical features we can use the uncertinity coefficient as correlation score. (See: https://en.wikipedia.org/wiki/Uncertainty_coefficient)

The uncertinity score builds on the conditinal entropy form the information theory.

The entropy for a single feature is defined as follows:

$$H(X) = -\sum_{x} P_X(x) log P_X(x)$$

The conditional entropy of X given Y is defined as follows:

$$H(X|Y) = -\sum_{x,y} P_{X,Y}(x,y) log P_{X|Y}(x|y)$$

The uncertinity coefficient of the two features can be calculated as follows:

$$U(X|Y) = \frac{H(X) - H(X|Y)}{H(X)}$$

We can implement an R function to calculate the entropy, conditional entropy and the uncertinity coefficient.

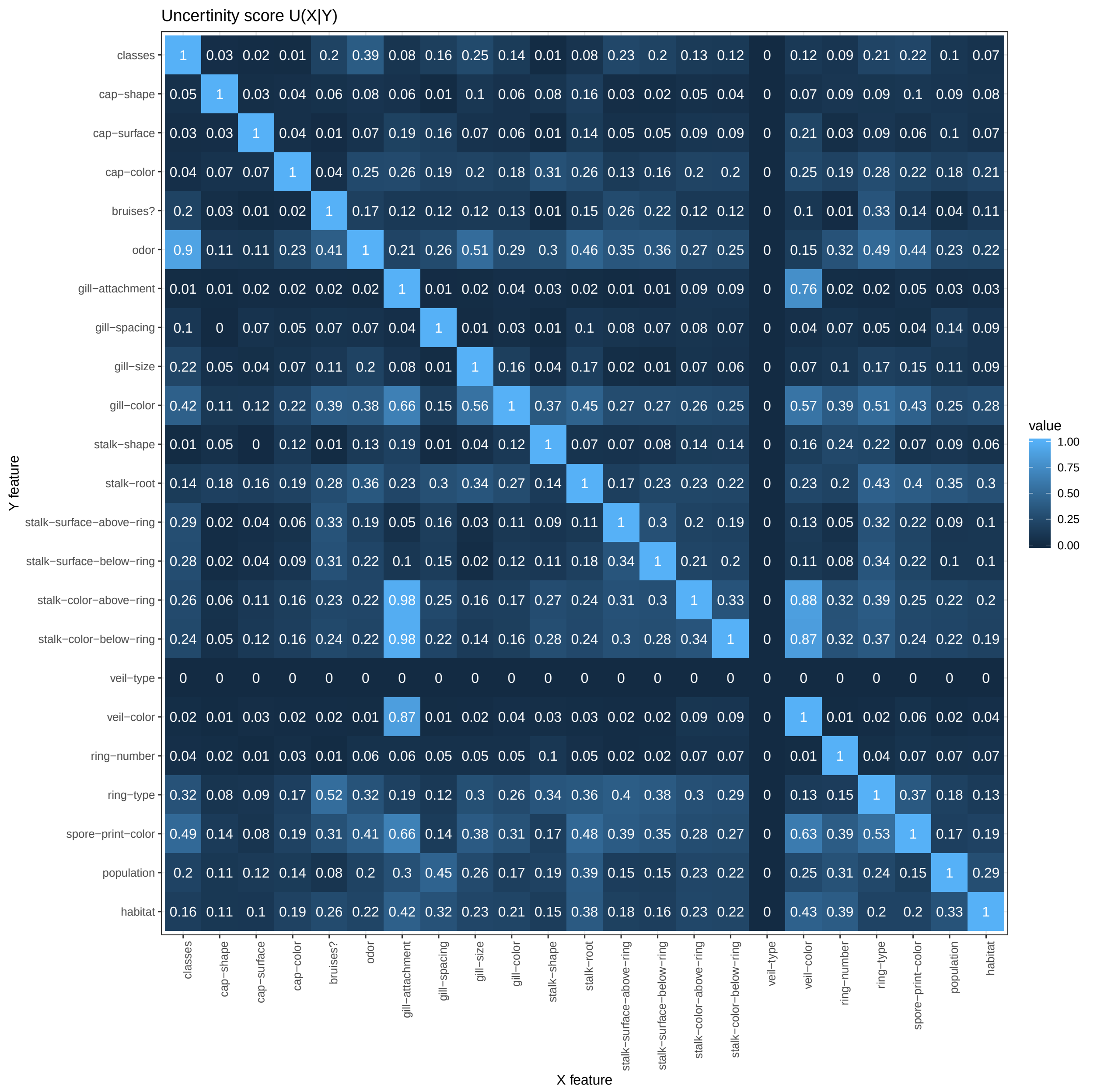
The following R code implements the calculation functions:

```
entropy <- function(dataset, colX){

  ret <- 0
  summarized_data <- dataset %>% group_by(.[,colX]) %>% summarise(n = n())
  rowCount <- nrow(dataset)
  level_items <- levels(dataset[,colX])
  for (item in level_items){
    prob <- summarized_data %>% filter(.[,1] == item) %>% pull(n) / rowCount
    ret <- ret + prob * log(prob,base = 2)
  }
  return (-ret)
}

cond_entropy <- function(dataset, colX, colY){
  ret <- 0
  rowCount <- nrow(dataset)
  summarized_y <- dataset %>% group_by(.[,colY]) %>% summarise(n = n())
  names(summarized_y)[1] <- "Y"
  level_itemsy <- levels(summarized_y$Y)

  for (itemy in level_itemsy){
    proby <- summarized_y %>% filter(Y == itemy) %>% pull(n) / rowCount
```

```r
    summarized_x <- dataset %>% filter(.[,colY] == itemy) %>% group_by(.[,colX]) %>%
                    summarise(n = n())
    names(summarized_x)[1] <- "X"
    level_itemsx <- levels(summarized_x$X)
    for (itemx in level_itemsx){
      filtered <- summarized_x %>% filter(X == itemx)
      if (nrow(filtered) > 0 && proby != 0){
        probxy <-  filtered$n / rowCount
        probx_at_y <- probxy / proby
        ret <- ret + probxy * log(probx_at_y,base = 2)
      }
    }
  }
  return (-ret)
}

uncertinity <- function(dataset, colX, colY){
  entr <- entropy(dataset,colX)
  cond_entr <- cond_entropy(dataset, colX, colY)
  if (entr == 0){
    return(0)
  }
  return ( (entr - cond_entr) / entr)
}
```

We can plot the uncertinity coefficient beween all the features as a matrix. The following R code creates the plot:

```r
uncertinity_plot <- function(dataset){
  #calculate uncertinity for all feature pairs
  uncertinity_df <- data_frame(X=character(), idx=numeric(), Y=character(),
                               idy=numeric(), value=numeric())
  labels <- names(dataset)
  for (i in 1:ncol(dataset))
  {
    for(j in 1:ncol(dataset))
    {
      uncertinity_df <- bind_rows(uncertinity_df, data_frame(Y = labels[j], idy=j ,
                                 X=labels[i], idx=i, value=uncertinity(train_data,i,j)))
    }
  }

  #plot the uncertinity with colors
  uncertinity_df %>% ggplot(aes(x=reorder(X,idx), y=reorder(Y,-idy), fill=value)) +
    geom_tile() + geom_text(aes(label=round(value,2)), color='white') +
    theme_bw() + theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
    xlab('X feature') + ylab('Y feature') + ggtitle('Uncertinity score U(X|Y)')
}

uncertinity_plot(train_data)
```



This plot shows the uncertinity coefficient between all feature pairs. The light blue values show a high uncertinity coefficient (high correlation) the datk blue values shows a low uncertinity coefficient (low correlation) The value 1 in the matrix shows a perfect correlation, therefore the matrix diagonal elements are 1. (The diagonal elements show the correlation of the features with himself, therefore is a perfect correlation)

We see, that the feature veil-type has 0 correlation with all other features. In the first data analysis we saw, that the feature veil-type has only one value, that causes the 0 correlation.

For the classification model we examine the first column of the matrix. This column shows the correlation betweem edibility and other feature. As we see, the feature odor has the highest correlation with the edibility. The other feature have only a correlation belowe 0.5.

We can use this information for developing a classification model that use only the features with high correlation with the edibility.

The classification model

Decision tree model

Feature based model

Implementation & result

Conclusion