Harvard PH125.9x Capstone: Mushroom Classification Analysis

Mauro Berlanda

April 2020

Introduction

The last assignment of the Datascience Professional Certificate by HarvardX on edx is submitting its own report. The main goal of the project is to prove the ability to clearly communicate the process and the insights gained from an analysis.

We are going to use for this analysis the Mushroom records drawn from The Audubon Society Field Guide to North American Mushrooms (1981. This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family.

> The csv file containing the data was originally downloaded from Kaggle due to its ease of manipulation. The file has been commited in a github repository since Kaggle downloads require authentication. Being unable to retrieve the raw zip file due to a corrupted output (`unzip error -1`), my script is downloading the uncompressed csv file. It does not exceed 365Kb, so it can be requested without any performance or network traffic concern.

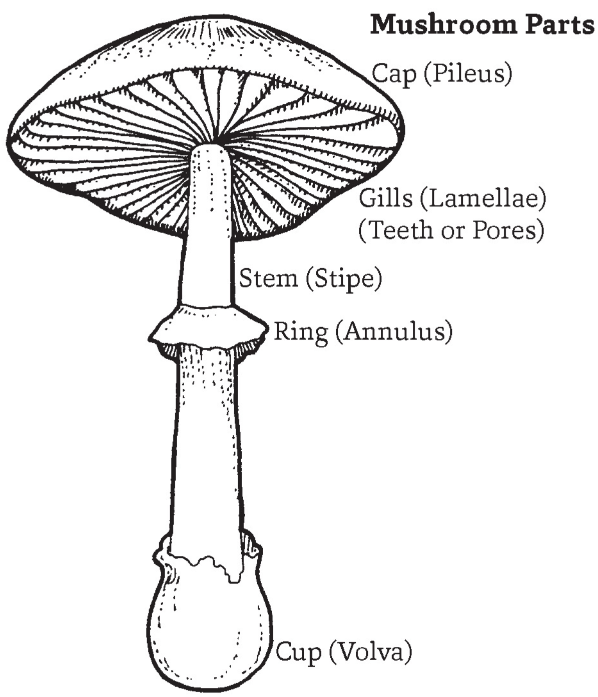
```r
file_url <- "https://raw.githubusercontent.com/mberlanda/ph125-9x-data-science-capstone/master/mushroom-
csv_filepath <- "./mushrooms.csv"

# Download the csv file if needed
if (!file.exists(csv_filepath)) {
  download.file(file_url, csv_filepath)
}

# Use read.csv to parse the file converting strings to factors
mushrooms <- read.csv(csv_filepath, header=TRUE, sep=",", stringsAsFactors=TRUE)
# Explore the columns and types of the dataset
str(mushrooms)
```

```
## 'data.frame':    8124 obs. of  23 variables:
##  $ class                   : Factor w/ 2 levels "e","p": 2 1 1 2 1 1 1 1 2 1 ...
##  $ cap.shape               : Factor w/ 6 levels "b","c","f","k",..: 6 6 1 6 6 6 1 1 6 1 ...
##  $ cap.surface             : Factor w/ 4 levels "f","g","s","y": 3 3 3 4 3 4 3 4 4 3 ...
##  $ cap.color               : Factor w/ 10 levels "b","c","e","g",..: 5 10 9 9 4 10 9 9 9 10 ...
##  $ bruises                 : Factor w/ 2 levels "f","t": 2 2 2 2 1 2 2 2 2 2 ...
##  $ odor                    : Factor w/ 9 levels "a","c","f","l",..: 7 1 4 7 6 1 1 4 7 1 ...
##  $ gill.attachment         : Factor w/ 2 levels "a","f": 2 2 2 2 2 2 2 2 2 2 ...
##  $ gill.spacing            : Factor w/ 2 levels "c","w": 1 1 1 1 2 1 1 1 1 1 ...
##  $ gill.size               : Factor w/ 2 levels "b","n": 2 1 1 2 1 1 1 1 2 1 ...
##  $ gill.color              : Factor w/ 12 levels "b","e","g","h",..: 5 5 6 6 5 6 3 6 8 3 ...
##  $ stalk.shape             : Factor w/ 2 levels "e","t": 1 1 1 1 2 1 1 1 1 1 ...
##  $ stalk.root              : Factor w/ 5 levels "?","b","c","e",..: 4 3 3 4 4 3 3 3 4 3 ...
##  $ stalk.surface.above.ring: Factor w/ 4 levels "f","k","s","y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ stalk.surface.below.ring: Factor w/ 4 levels "f","k","s","y": 3 3 3 3 3 3 3 3 3 3 ...
```

```
##  $ stalk.color.above.ring  : Factor w/ 9 levels "b","c","e","g",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ stalk.color.below.ring  : Factor w/ 9 levels "b","c","e","g",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ veil.type               : Factor w/ 1 level "p": 1 1 1 1 1 1 1 1 1 1 ...
##  $ veil.color              : Factor w/ 4 levels "n","o","w","y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ ring.number             : Factor w/ 3 levels "n","o","t": 2 2 2 2 2 2 2 2 2 2 ...
##  $ ring.type               : Factor w/ 5 levels "e","f","l","n",..: 5 5 5 5 1 5 5 5 5 5 ...
##  $ spore.print.color       : Factor w/ 9 levels "b","h","k","n",..: 3 4 4 3 4 3 3 4 3 3 ...
##  $ population              : Factor w/ 6 levels "a","c","n","s",..: 4 3 3 4 1 3 3 4 5 4 ...
##  $ habitat                 : Factor w/ 7 levels "d","g","l","m",..: 6 2 4 6 2 2 4 4 2 4 ...
```

```r
rm(file_url, csv_filepath)
```

To improve the domain knowledge, you can find below an image illustrating the different parts of a mushroom:



All the attributes are factors and they represent the following abbreviations:

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

The classes used for the outcome are `edible` or `poisonous`. In the Kaggle version of the data set there is no `unknown` classification value and missing values have been removed:

```
unique(mushrooms$class)
```

```
## [1] p e
## Levels: e p
```

```
sum(is.na(mushrooms))
```

```
## [1] 0
```

Analysis

The main challenges raised by this data set are:

a. all the features are categorical => the core of the solution will be in the data wrangling part
b. the amount of observation is small (overall around 8k, but we will split it into a training and test sets) => we will need to implement some techniques to avoid overfitting

Data Wrangling

If we try to approach this classification problem as the `tissue_gene_expression` data set distributed by the `dslabs` package, we will quickly realise that the most of utilities used in the PH125.x courses won't be available.

```
# format as list of a features matrix and outcome vector
formatted_mushrooms <- list(
  x = mushrooms %>% select(-class) %>% as.matrix,
  y = mushrooms$class
)

# Partition train and test set
set.seed(22, sample.kind="Rounding")
test_index <- createDataPartition(y=formatted_mushrooms$y, times=1, p=.30, list=FALSE)

train_set <- list(
  x = formatted_mushrooms$x[-test_index,],
  y = formatted_mushrooms$y[-test_index]
)

hclust(train_set$x)
# Error in if (is.na(n) || n > 65536L) stop("size cannot be NA nor exceed 65536") :
# missing value where TRUE/FALSE needed

pca <- prcomp(train_set$x)
# Error in colMeans(x, na.rm = TRUE) : 'x' must be numeric
```

The problem is the most of libraries cannot work with categorical features only. A solution to this problem can be *encoding the categorical features*. Even if the most of resources I googled were explaining how to perform this technique with python (All about Categorical Variable Encoding, How to Encode Categorical Data), there is an excellent R blog post providing a wide overview on the topic.

The most important concept to recall before continuing are:

- Categorical variables can be considered:
  - *Nominal* (e.g. pen/pencil/eraser or cow/dog/cat)
  - *Ordinal* (e.g. excellent/good/bad or fantastic/ok/don't like).
- The three main routes to encode factors string data type are:
  - *Classic Encoders*: e.g. ordinal, OneHot, Binary, Frequency, Hashing …
  - *Contrast Encoders*: encode data by looking at different levels of features
  - *Bayesian Encoders*: use the target as the foundation of the encoding

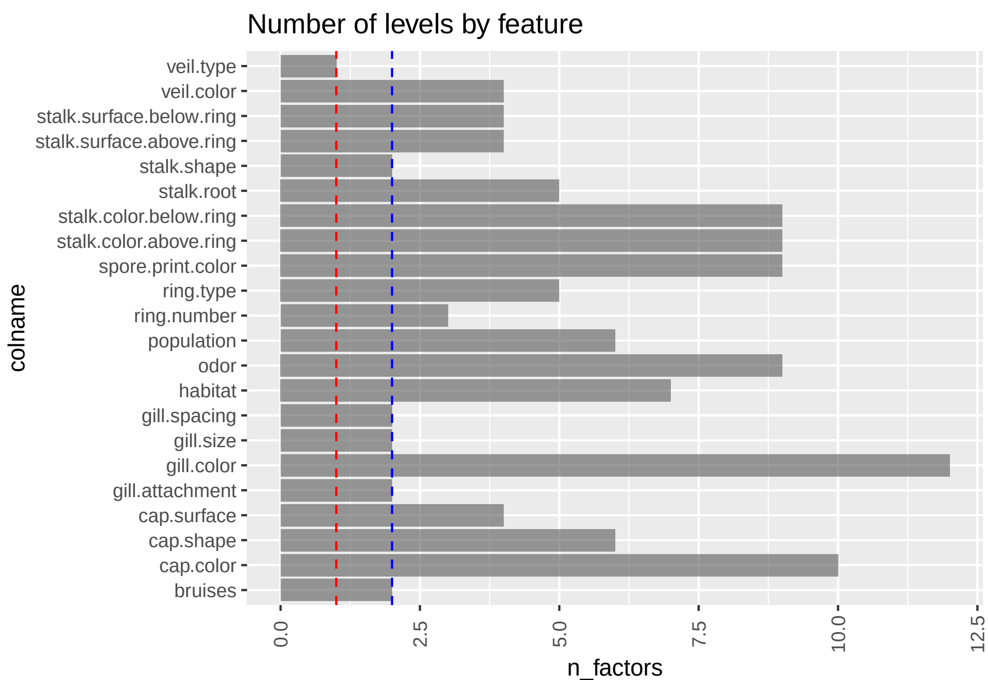


Table 1: Attributes with less than 4 factors

colname	n_factors
ring.number	3
bruises	2
gill.attachment	2
gill.spacing	2
gill.size	2
stalk.shape	2
veil.type	1

We can make these insights actionable with the attributes description provided in the Introduction:

- `veil.type` attribute can be ignored since it has only one level
- `bruises` attribute can be considered as a logical/binary vector
- `ring.number` attribute is an ordinal variable

- `population` attribute may be considered as an ordinal variable as well, but it is ambigous (at least for my domain understanding level)
- the other 18 attributes can be considered as nominal variables