Mushroom Classification

Andreas Kaß, Cornelia Gruber, Felix Langer, Victoria Szabo

10.06.2020

source: https://www.shutterstock.com/de-de/search/mushroom?sort=popular&page=1

Collecting mushrooms requires general knowledge of poisonous or edible mushroom indicators and mushroom poisoning stages. Since mushrooms come in many different colors and shapes, their fit or unfit for human consumption is not necessarily obvious. Especially instead of asking your grandmother you want to rely on machine learning. In this case study we will find out which features and attributes correlating the most with the edibility of a mushroom.

Mushroom attributes

We have given to a friendly mycologist a mushroom encyclopedia and asked him to select the "attributes" for "poisonous". For the time being we can use various attributes of a mushroom, such as its cap color, its cap shape, its cap surface, its habitat or its stalk shape. The cap shape mushroom people are using are either "bell", "conical", "convex", "flat", "knobbed" or "sunken" (see figure below). The dataset contains 8124 observations with 23 nominal features.

source: https://www.mushroomobserver.org/account/api/observation/identification/

Dataset Overview

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

```
'data.frame':   8124 obs. of  23 :
 $ classes                 : chr  e e p e e e e e e e ...
 $ cap-shape               : chr  x b x x x x b b x x ...
 $ cap-surface             : chr  s s s y s s s s y y ...
 $ cap-color               : chr  n y w w g y w g n n ...
 $ bruises                 : chr  t t t t f f f f t t ...
 $ odor                    : chr  n a l y f m p f n a ...
 $ gill-attachment         : chr  f f f f f f f f f f ...
 $ gill-spacing            : chr  c c c c w w w w c c ...
 $ gill-size               : chr  b b n b b b b b b b ...
 $ gill-color              : chr  k k n k p p p p u u ...
 $ stalk-shape             : chr  e e e e t e e e e e ...
 $ stalk-root              : chr  b c c b b b b b b b ...
 $ stalk-surface-above-ring: chr  s s s s s s s s s s ...
 $ stalk-surface-below-ring: chr  s s s s s s s s s s ...
 $ stalk-color-above-ring  : chr  w w w w w w w w w w ...
 $ stalk-color-below-ring  : chr  w w w w w w w w w w ...
 $ veil-type               : chr  p p p p p p p p p p ...
 $ veil-color              : chr  w w w w w w w w w w ...
 $ ring-number             : chr  o o o o o o o o o o ...
 $ ring-type               : chr  p p p p e e e e p p ...
 $ spore-print-color       : chr  k k k k n n k n k n ...
 $ population              : chr  s n n s a v s v s a ...
 $ habitat                 : chr  u g m u g d d d u d ...
```

```
classes cap-shape cap-surface cap-color bruises  odor
   e    2012    3152      2556      2284    4748  3528
   p    1948    3944      2556      1840    3376  3796

gill-attachment gill-spacing gill-size gill-color stalk-shape stalk-root
      7914           6812       5612      1728       4608       2480
      7488           1312       2512      6484       3516       5644

stalk-surface-above-ring stalk-surface-below-ring stalk-color-above-ring
                    5176                     4936                     4464
                    5412                     5772                     5772

stalk-color-below-ring veil-type veil-color ring-number ring-type spore-print-color
                  4384       7924       7488        7488     3968              2388
                  5772       7924       1728        1728     4040              3968

population habitat
      4040    3148
      3968    4292

classes cap-shape cap-surface cap-color bruises  odor
   e    1556    3656      2320      2084    4420  3408
   p    1708    3488      2320      1756    3152  3488

gill-attachment gill-spacing gill-size gill-color stalk-shape stalk-root
           7914           6812       5612      1728       4608       2480
           7488           1312       2512      6484       3516       5644

stalk-surface-above-ring stalk-surface-below-ring stalk-color-above-ring
                    5176                     4936                     4464
                    5412                     5772                     5772

stalk-color-below-ring veil-type veil-color ring-number ring-type spore-print-color
                  4384       7924       7488        7488     3968              2388
                  5772       7924       1728        1728     4040              3968

population habitat
      4040    3148
      3968    4292

classes cap-shape cap-surface cap-color bruises  odor
   e    1420    2636      1876      1828    3328  2120
   p    1296    2456      1876      1284    2376  2456

gill-attachment gill-spacing gill-size gill-color stalk-shape stalk-root
           5934           4512       4172      1728       3516       1768
           5412            944       1872      4756       2772       3956

stalk-surface-above-ring stalk-surface-below-ring stalk-color-above-ring
                    3744                     3616                     3344
                    3916                     4228                     4228

stalk-color-below-ring veil-type veil-color ring-number ring-type spore-print-color
                  3216       5812       5412        5412     2952              1632
                  4228       5812       1728        1728     2952              2952

population habitat
      3040    2364
      2952    3332
```

In order to get a basic feeling for our mushrooms let's do some plotting.
Click-scores to be quite good indicator whether a mushroom is poisonous or edible.

Distribution of class labels - odor

Distribution of class labels - gill-color

Mushroom Learning

Evaluation Framework

mlr3 is a machine learning (ML) in R. The dataset is a nice mushroom learning Framework offering a common ground for all necessary steps as defining a task, training a learner, tuning a learner, and evaluating a learner.

Task is mlr3 contains the data as well as meta information such as the name of the target variable or the a "classification" "regression" task.

```
# Initialize classification task
task_mushroom = mlr_tasks$get("classif.mushroom_data")
                 id backend nrow ncol pkgs
1: classif.iris    iris   150    5 mlr3
2: mlr_tasks_iris  iris   150    5 mlr3
Warnings:
1: 'mlr_tasks_iris' is deprecated in mlr3 v0.5.0.
```

Additionally to building a machine learning model the classes in mlr3 provide a interface to obtain an unbiased generalization error for our estimates. Therefore we decided for a nested resampling strategy with a 5-fold CV in inner loop and a 10-fold CV in the outer loop. The number of folds was chosen with respect to keeping the fold sized balanced and still considering enough data.

```
# Define 5-fold cross validation for inner loop
# 5-fold gives balanced CV for tuning
inner_resampling = rsmp("cv", folds = 5)
# Define 10-fold cross validation for outer loop
outer_resampling = rsmp("cv", folds = 10)
```

While tuning and all the subtly hooked up the right things when measures is useful for assessing the performance for the more relevant parts of machine learning, it is also time consuming.
Since the range of values of the hyperparameters is quite limited we chose grid search for optimizing.

```
# Minimal learner
lrn_ranger = lrn("classif.ranger")
# Define parameter space
search_space = ps(
  num.trees = p_int(lower = 1, upper = 2000),
  mtry = p_int(lower = 1, upper = 5),
  sample.fraction = p_dbl(lower = 0.1, upper = 0.9),
  replace = p_lgl(),
  respect.unordered.factors = p_fct(c("order", "partition"))
)
# Define search strategy
tuner = tnr("grid_search")
# Define measure for parameter tuning
measure_auc_cv = msr("classif.auc", predict_type = "prob")
# Define measure for benchmark
measure_auc = msrs(c("classif.auc", "classif.fbeta"),
                   fbeta.beta = 2,
                   predict_type = "prob")
# Define terminator for parameter tuning
terminator_evals = trm("evals", n_evals = 50)
# Define learner for parameter tuning
learner_tuned = auto_tuner(
  tuner = tuner,
  learner = lrn_ranger,
  resampling = inner_resampling,
  measure = measure_auc_cv,
  terminator = terminator_evals,
  search_space = search_space
)
# Nested resampling for unbiased evaluation
rr = resample(task = task_mushroom,
              learner = learner_tuned,
              resampling = outer_resampling,
              store_models = TRUE)
rr$aggregate(measure_auc)
rr$score()[, .(iteration, task_id, learner_id, resampling_id, classif.auc)]
# Extract inner tuned parameters
rr$extract_inner_tuned_values()[, .(iteration, mtry, sample.fraction, replace, respect.unordered.factors, num.trees)]
# Ensure all models are stored
rr$ignoreed(TRUE)
# Create a new learner with the best parameters
lrn_ranger_best = rr$learners[[1]]$model$learner
# Train the learner on the full data set
lrn_ranger_best$train(task_mushroom)
# Look at the model
lrn_ranger_best$model
# Predicting on the full data set
pred = lrn_ranger_best$predict(task_mushroom)
# Look at the confusion matrix
pred$confusion
# Look at the measures
pred$score(measures = msrs(c("classif.acc", "classif.ppv", "classif.tpr", "classif.fnr", "classif.fpr", "classif.tnr")))
# Look at the ROC curve
autoplot(pred, type = "roc")
# Look at the threshold vs cost curve
autoplot(pred, type = "threshold")
```

Since it all was that easy to unlock a sandbox-scenario-during-transporting-when-printing-the-benchmark-result-problem.

```
# Benchmark
design = benchmark_grid(
  tasks = task_mushroom,
  learners = lrn("classif.ranger",
                 num.trees = c(500, 1000, 2000),
                 mtry = c(1, 2, 3, 4, 5),
                 sample.fraction = c(0.1, 0.2, 0.3, 0.4, 0.5),
                 replace = c(TRUE, FALSE),
                 respect.unordered.factors = c("order", "partition"),
                 predict_type = "prob"),
  resamplings = rsmp("cv", folds = 10)
)
# Load the benchmark grid
bgrid = benchmark(design)
# Save the benchmark grid
saveRDS(bgrid, file = "./data/bgrid.rds")
# Load the benchmark grid
bgrid = readRDS(file = "./data/bgrid.rds")
# Look at the benchmark results
bgrid$aggregate(measures = msrs(c("classif.acc", "classif.ppv", "classif.tpr", "classif.fnr", "classif.fpr", "classif.tnr")))[, .(learner_id, classif.acc, classif.ppv, classif.tpr, classif.fnr, classif.fpr, classif.tnr)]
# Look at the ROC curve
autoplot(bgrid, type = "roc")
# Look at the threshold vs cost curve
autoplot(bgrid, type = "threshold")
```

Find of all one that has to errors or warnings occurred during resampling when printing the benchmark result problem).

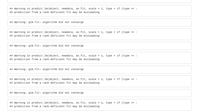
```
# Benchmarking
# Load the benchmark grid
bgrid = readRDS(file = "./data/bgrid.rds")
# Look at the benchmark results
bgrid$aggregate(measures = msrs(c("classif.acc", "classif.ppv", "classif.tpr", "classif.fnr", "classif.fpr", "classif.tnr")))[, .(learner_id, classif.acc, classif.ppv, classif.tpr, classif.fnr, classif.fpr, classif.tnr)]
# Look at the ROC curve
autoplot(bgrid, type = "roc")
# Look at the threshold vs cost curve
autoplot(bgrid, type = "threshold")
```

An expected or disappointed model shows up a classification error of close to 0.5.

```
autoplot(bgrid)
```

```
autoplot(bgrid, type = "roc")
```

ROC




```
Cmp_learner_performance = bm$aggregate(measures)
```

```
[1]  1  2  1  1  1
```