Naive Bayes Mushroom

Eduardo Ortiz

2026-01-09

Introduction

Mushrooms exhibit a wide variety of physical characteristics, many of which can indicate whether they are safe to eat or potentially poisonous. In this analysis, we aim to classify mushrooms based on these characteristics using a probabilistic machine learning approach.

The dataset contains multiple categorical features, including cap shape, odor, gill size, and stalk properties. By applying a Naive Bayes classifier, we can calculate the likelihood that a given mushroom belongs to the edible or poisonous class, taking advantage of the conditional independence assumption to simplify computations.

This analysis will walk through: - Data preparation and exploration
- Training a Naive Bayes model
- Evaluating its predictions on unseen data
- Interpreting results with a focus on model performance and reliability

Data

The mushroom dataset contains observations of various mushroom species, each labeled as edible (e) or poisonous (p). The dataset includes categorical features such as cap shape, cap color, gill size, gill color, stalk shape, and odor. These features provide descriptive information that can help predict whether a mushroom is safe to eat. The dataset is clean, with no missing values, making it suitable for supervised classification using Naive Bayes.

The dataset is loaded from the data/mushrooms.csv file and examined using basic inspection functions such as str(). Initial exploration reveals that all features are categorical, and the target variable class is balanced between edible and poisonous mushrooms.

```
#read the dataset
mushroom <- read.csv("C:/Users/eddie/Downloads/data/mushrooms.csv")

# Check the structure of the dataset
str(mushroom)
```

```
## 'data.frame':    8124 obs. of  23 variables:
##  $ class                   : chr  "p" "e" "e" "p" ...
##  $ cap.shape               : chr  "x" "x" "b" "x" ...
##  $ cap.surface             : chr  "s" "s" "s" "y" ...
##  $ cap.color               : chr  "n" "y" "w" "w" ...
##  $ bruises                 : chr  "t" "t" "t" "t" ...
##  $ odor                    : chr  "p" "a" "l" "p" ...
```

```
##  $ gill.attachment         : chr  "f" "f" "f" "f" ...
##  $ gill.spacing            : chr  "c" "c" "c" "c" ...
##  $ gill.size               : chr  "n" "b" "b" "n" ...
##  $ gill.color              : chr  "k" "k" "n" "n" ...
##  $ stalk.shape             : chr  "e" "e" "e" "e" ...
##  $ stalk.root              : chr  "e" "c" "c" "e" ...
##  $ stalk.surface.above.ring: chr  "s" "s" "s" "s" ...
##  $ stalk.surface.below.ring: chr  "s" "s" "s" "s" ...
##  $ stalk.color.above.ring  : chr  "w" "w" "w" "w" ...
##  $ stalk.color.below.ring  : chr  "w" "w" "w" "w" ...
##  $ veil.type               : chr  "p" "p" "p" "p" ...
##  $ veil.color              : chr  "w" "w" "w" "w" ...
##  $ ring.number             : chr  "o" "o" "o" "o" ...
##  $ ring.type               : chr  "p" "p" "p" "p" ...
##  $ spore.print.color       : chr  "k" "n" "n" "k" ...
##  $ population              : chr  "s" "n" "n" "s" ...
##  $ habitat                 : chr  "u" "g" "m" "u" ...
```

Data Partition

The dataset is split into a training set (70%) and a test set (30%) using the createDataPartition() function from the caret package. This ensures that the model is trained on a majority of the data while reserving a portion for unbiased evaluation. Feature columns are separated from the target labels to prepare for model training.

```
# Data Partition
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.1
```

```
## Loading required package: lattice
```

```
set.seed(123)
partition <- createDataPartition(mushroom$class, p = 0.7, list = FALSE)
train.df <- mushroom[partition, ]
test.df <- mushroom[-partition, ]

# Check the structure of the training and testing sets
str(train.df)
```

```
## 'data.frame':    5688 obs. of  23 variables:
##  $ class                   : chr  "p" "e" "e" "p" ...
##  $ cap.shape               : chr  "x" "x" "b" "x" ...
##  $ cap.surface             : chr  "s" "s" "s" "y" ...
##  $ cap.color               : chr  "n" "y" "w" "w" ...
##  $ bruises                 : chr  "t" "t" "t" "t" ...
```

```
##  $ odor                    : chr  "p" "a" "l" "p" ...
##  $ gill.attachment         : chr  "f" "f" "f" "f" ...
##  $ gill.spacing            : chr  "c" "c" "c" "c" ...
##  $ gill.size               : chr  "n" "b" "b" "n" ...
##  $ gill.color              : chr  "k" "k" "n" "n" ...
##  $ stalk.shape             : chr  "e" "e" "e" "e" ...
##  $ stalk.root              : chr  "e" "c" "c" "e" ...
##  $ stalk.surface.above.ring: chr  "s" "s" "s" "s" ...
##  $ stalk.surface.below.ring: chr  "s" "s" "s" "s" ...
##  $ stalk.color.above.ring  : chr  "w" "w" "w" "w" ...
##  $ stalk.color.below.ring  : chr  "w" "w" "w" "w" ...
##  $ veil.type               : chr  "p" "p" "p" "p" ...
##  $ veil.color              : chr  "w" "w" "w" "w" ...
##  $ ring.number             : chr  "o" "o" "o" "o" ...
##  $ ring.type               : chr  "p" "p" "p" "p" ...
##  $ spore.print.color       : chr  "k" "n" "n" "k" ...
##  $ population              : chr  "s" "n" "n" "s" ...
##  $ habitat                 : chr  "u" "g" "m" "u" ...
```

```
str(test.df)
```

```
## 'data.frame':    2436 obs. of  23 variables:
##  $ class                   : chr  "e" "e" "p" "p" ...
##  $ cap.shape               : chr  "x" "b" "x" "x" ...
##  $ cap.surface             : chr  "s" "y" "y" "y" ...
##  $ cap.color               : chr  "g" "w" "w" "w" ...
##  $ bruises                 : chr  "f" "t" "t" "t" ...
##  $ odor                    : chr  "n" "l" "p" "p" ...
##  $ gill.attachment         : chr  "f" "f" "f" "f" ...
##  $ gill.spacing            : chr  "w" "c" "c" "c" ...
##  $ gill.size               : chr  "b" "b" "n" "n" ...
##  $ gill.color              : chr  "k" "n" "p" "k" ...
##  $ stalk.shape             : chr  "t" "e" "e" "e" ...
##  $ stalk.root              : chr  "e" "c" "e" "e" ...
##  $ stalk.surface.above.ring: chr  "s" "s" "s" "s" ...
##  $ stalk.surface.below.ring: chr  "s" "s" "s" "s" ...
##  $ stalk.color.above.ring  : chr  "w" "w" "w" "w" ...
##  $ stalk.color.below.ring  : chr  "w" "w" "w" "w" ...
##  $ veil.type               : chr  "p" "p" "p" "p" ...
##  $ veil.color              : chr  "w" "w" "w" "w" ...
##  $ ring.number             : chr  "o" "o" "o" "o" ...
##  $ ring.type               : chr  "e" "p" "p" "p" ...
##  $ spore.print.color       : chr  "n" "n" "k" "n" ...
##  $ population              : chr  "a" "s" "v" "v" ...
##  $ habitat                 : chr  "g" "m" "g" "u" ...
```

```
# Extract labels
trainLabels <- train.df$class
testLabels <- test.df$class
```

Model Training

A Naive Bayes classifier is trained on the training set using the naiveBayes() function from the e1071 package with Laplace smoothing. This probabilistic model leverages the categorical features to estimate the likelihood of a mushroom being edible or poisonous. Naive Bayes is chosen because it handles categorical data efficiently and provides interpretable class probabilities.

```
# Create the model
library(e1071)
mushroom_bayes <- naiveBayes(train.df[, -1], trainLabels, laplace = 1)

# Display the model
print(mushroom_bayes)
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = train.df[, -1], y = trainLabels, laplace = 1)
##
## A-priori probabilities:
## trainLabels
##         e         p
## 0.5179325 0.4820675
##
## Conditional probabilities:
##            cap.shape
## trainLabels            b            c            f            k            s
##           e 0.0926680244 0.0003394433 0.3754243041 0.0505770536 0.0088255261
##           p 0.0120350109 0.0014587892 0.3916849015 0.1597374179 0.0003646973
##            cap.shape
## trainLabels            x
##           e 0.4742023082
##           p 0.4369073669
##
##            cap.surface
## trainLabels            f            g            s            y
##           e 0.3713509844 0.0003394433 0.2644263408 0.3652410048
##           p 0.1940189643 0.0014587892 0.3603209336 0.4456601021
##
##            cap.color
## trainLabels            b            c            e            g            n
##           e 0.0128988459 0.0081466395 0.1496945010 0.2403258656 0.3004073320
##           p 0.0295404814 0.0032822757 0.2250182349 0.2111597374 0.2647702407
##            cap.color
## trainLabels            p            r            u            w            y
##           e 0.0132382892 0.0044127631 0.0050916497 0.1704005431 0.0987780041
##           p 0.0233406273 0.0003646973 0.0003646973 0.0795040117 0.1663019694
##
##            bruises
## trainLabels         f         t
##           e 0.3448744 0.6558045
##           p 0.8512035 0.1495259
```

```
##
##            odor
## trainLabels            a            c            f            l            m
##           e 0.0974202308 0.0003394433 0.0003394433 0.0940257977 0.0003394433
##           p 0.0003646973 0.0506929249 0.5485047411 0.0003646973 0.0087527352
##            odor
## trainLabels            n            p            s            y
##           e 0.8095723014 0.0003394433 0.0003394433 0.0003394433
##           p 0.0309992706 0.0587162655 0.1553610503 0.1495258935
##
##            gill.attachment
## trainLabels           a           f
##           e 0.046843177 0.953835709
##           p 0.005835157 0.994894238
##
##            gill.spacing
## trainLabels          c          w
##           e 0.71894094 0.28173795
##           p 0.97191831 0.02881109
##
##            gill.size
## trainLabels          b          n
##           e 0.93007468 0.07060421
##           p 0.42742524 0.57330416
##
##            gill.color
## trainLabels            b            e            g            h            k
##           e 0.0003394433 0.0230821453 0.0549898167 0.0529531568 0.0824847251
##           p 0.4514952589 0.0003646973 0.1283734500 0.1363967907 0.0153172867
##            gill.color
## trainLabels            n            o            p            r            u
##           e 0.2240325866 0.0169721656 0.2016293279 0.0003394433 0.1038696538
##           p 0.0299051787 0.0003646973 0.1564551422 0.0065645514 0.0127644055
##            gill.color
## trainLabels            w            y
##           e 0.2270875764 0.0162932790
##           p 0.0609044493 0.0054704595
##
##            stalk.shape
## trainLabels         e         t
##           e 0.3822132 0.6184657
##           p 0.4777535 0.5229759
##
##            stalk.root
## trainLabels            ?            b            c            e            r
##           e 0.1683638832 0.4555329260 0.1225390360 0.2087576375 0.0465037339
##           p 0.4606126915 0.4711889132 0.0109409190 0.0587162655 0.0003646973
##
##            stalk.surface.above.ring
## trainLabels           f           k           s           y
##           e 0.100135777 0.029871012 0.867617108 0.003733876
##           p 0.036469730 0.576951131 0.385485047 0.002552881
##
##            stalk.surface.below.ring
```

```
## trainLabels          f          k          s          y
##           e 0.11303462 0.03054990 0.80787508 0.04989817
##           p 0.03610503 0.55397520 0.39132020 0.02005835
##
##            stalk.color.above.ring
## trainLabels            b            c            e            g            n
##           e 0.0003394433 0.0003394433 0.0230821453 0.1347589952 0.0037338764
##           p 0.1108679796 0.0087527352 0.0003646973 0.0003646973 0.1123267688
##            stalk.color.above.ring
## trainLabels            o            p            w            y
##           e 0.0468431772 0.1395112016 0.6541072641 0.0003394433
##           p 0.0003646973 0.3329686360 0.4347191831 0.0025528811
##
##            stalk.color.below.ring
## trainLabels            b            c            e            g            n
##           e 0.0003394433 0.0003394433 0.0241004752 0.1388323150 0.0159538357
##           p 0.1126914661 0.0087527352 0.0003646973 0.0003646973 0.1105032823
##            stalk.color.below.ring
## trainLabels            o            p            w            y
##           e 0.0468431772 0.1310251188 0.6452817379 0.0003394433
##           p 0.0003646973 0.3391684902 0.4245076586 0.0065645514
##
##            veil.type
## trainLabels        p
##           e 1.000339
##           p 1.000365
##
##            veil.color
## trainLabels            n            o            w            y
##           e 0.0230821453 0.0241004752 0.9538357094 0.0003394433
##           p 0.0003646973 0.0003646973 0.9981765135 0.0025528811
##
##            ring.number
## trainLabels            n            o            t
##           e 0.0003394433 0.8811948405 0.1194840462
##           p 0.0087527352 0.9744711889 0.0178701678
##
##            ring.type
## trainLabels            e            f            l            n            p
##           e 0.2460964019 0.0135777325 0.0003394433 0.0003394433 0.7413441955
##           p 0.4628008753 0.0003646973 0.3300510576 0.0087527352 0.1998541211
##
##            spore.print.color
## trainLabels            b            h            k            n            o
##           e 0.0112016293 0.0135777325 0.3991853360 0.4090291921 0.0122199593
##           p 0.0003646973 0.4015317287 0.0557986871 0.0536105033 0.0003646973
##            spore.print.color
## trainLabels            r            u            w            y
##           e 0.0003394433 0.0118805160 0.1317040054 0.0139171758
##           p 0.0178701678 0.0003646973 0.4730123997 0.0003646973
##
##            population
## trainLabels            a            c            n            s            v
##           e 0.0936863544 0.0695858792 0.0896130346 0.2101154107 0.2820773931
```

```
##           p 0.0003646973 0.0127644055 0.0003646973 0.0853391685 0.7374179431
##            population
## trainLabels            y
##           e 0.2569585879
##           p 0.1659372721
##
##            habitat
## trainLabels            d            g            l            m            p
##           e 0.4470468432 0.3302783435 0.0590631365 0.0604209097 0.0349626612
##           p 0.3278628738 0.1783369803 0.1557257476 0.0094821298 0.2636761488
##            habitat
## trainLabels            u            w
##           e 0.0227427020 0.0478615071
##           p 0.0671043034 0.0003646973
```

Evaluation

The Naive Bayes model was trained on 70% of the mushroom dataset and tested on the remaining 30%. Predictions were made on the test set, and both the predicted and true class labels were converted to factors to ensure proper comparison. The levels of the predicted labels were aligned with the true labels to avoid any mismatches. The resulting confusion matrix provides a detailed assessment of the model's performance. Given that the mushroom dataset consists of categorical features representing physical characteristics of mushrooms (such as cap shape, gill color, and odor), the Naive Bayes classifier is well-suited because it assumes feature independence and handles categorical data efficiently. The confusion matrix summarizes the number of correctly and incorrectly classified instances, allowing us to evaluate metrics such as accuracy, sensitivity, and specificity. Overall, this approach provides a clear understanding of the model's predictive power and highlights areas where misclassification may occur, ensuring that edible and poisonous mushrooms can be reliably distinguished based on the features in the dataset.

```r
# Run prediction on the test set
m_pred <- predict(mushroom_bayes, test.df[, -1])
```

```r
# Convert predicted and true labels to factors
m_pred <- factor(m_pred)
testLabels <- factor(testLabels)
```

```r
# Ensure levels match
levels(m_pred) <- levels(testLabels)
```

```r
# Calculate confusion matrix
confusionMatrix(m_pred, testLabels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    e    p
##          e 1256  143
##          p    6 1031
##
##                Accuracy : 0.9388
##                  95% CI : (0.9286, 0.948)
```

```
##     No Information Rate : 0.5181
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.877
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9952
##             Specificity : 0.8782
##          Pos Pred Value : 0.8978
##          Neg Pred Value : 0.9942
##              Prevalence : 0.5181
##          Detection Rate : 0.5156
##    Detection Prevalence : 0.5743
##       Balanced Accuracy : 0.9367
##
##        'Positive' Class : e
##
```

Results

The Naive Bayes model achieved 93.88% accuracy on the test set, demonstrating strong overall performance. It correctly identified nearly all edible mushrooms (sensitivity 99.52%) and most poisonous mushrooms (specificity 87.82%). The high Kappa value (0.877) confirms strong agreement between predicted and true classes, showing that the categorical features in the dataset provide clear signals for distinguishing edible and poisonous mushrooms.