Mushroom Project

Amber Palladino

August 5, 2019

Overview

This report describes my work on the Choose Your Own capstone project for the HarvardX online Data Science professional certificate program.

The project assignment is to choose a dataset for a machine learning project. Websites with publicly available datasets were provided as options for exploration. I selected a dataset about mushrooms, which is based on records drawn from The Audobon Society Field Guide. The dataset contains over 8,000 records, each of which corresponds to a hypothetical sample. Each sample is classified as edible or poisonous ("class"), and each record also includes descriptions of the sample's physical characteristics across 22 categories such as cap shape, odor, and gill color.

The goal of my project is to create an algorithm that categorizes mushrooms as edible or poisonous based on their physical characteristics. To accomplish this, I split the mushroom data into training and test sets. The training dataset was used for model exploration and evaluation, and to fit the final model. The test dataset was used solely to measure the accuracy of predictions generated using the final model.

Methods

The first step of the assignment requires code that installs and loads the R packages used within the project. The Mushroom dataset is then downloaded from the University of California - Irvine website as a text file. The text is parsed into a data frame for exploration and analysis.

```r
# Install required packages, if needed
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(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
if(!require(e1071)) install.packages("e1071", repos = "http://cran.us.r-project.org")

# Mushroom dataset: Description
# https://archive.ics.uci.edu/ml/datasets/Mushroom
# Mushroom dataset: Data Download
# https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.data

# Download mushroom data from the University of California - Irvine database
dl <- tempfile()
download.file("https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.data",
dl)

# Read downloaded .data file into an R data frame and add column names (based on dataset description)
mushrooms <- read.csv(dl, header=FALSE, sep=",", col.names = c("class", "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"))
```

Exploration includes displaying the first few rows of the dataset, confirming that the data were successfully stored as a data frame object, viewing the dimensions of the data frame, and examining a summary of the data. The most critical characteristic among the data points for each mushroom specimen is whether each sample is poisonous or edible, so distribution between those categories is examined specifically. Definition of the alphabetic variables used for each attribute can be viewed at https://archive.ics.uci.edu/ml/datasets/Mushroom.

```
# Data exploration and visualization
head(mushrooms)
```

```
##   class cap_shape cap_surface cap_color bruises odor gill_attachment
## 1     p         x           s         n       t    p               f
## 2     e         x           s         y       t    a               f
## 3     e         b           s         w       t    l               f
## 4     p         x           y         w       t    p               f
## 5     e         x           s         g       f    n               f
## 6     e         x           y         y       t    a               f
##   gill_spacing gill_size gill_color stalk_shape stalk_root
## 1            c         n          k           e          e
## 2            c         b          k           e          c
## 3            c         b          n           e          c
## 4            c         n          n           e          e
## 5            w         b          k           t          e
## 6            c         b          n           e          c
##   stalk_surface_above_ring stalk_surface_below_ring stalk_color_above_ring
## 1                        s                        s                      w
## 2                        s                        s                      w
## 3                        s                        s                      w
## 4                        s                        s                      w
## 5                        s                        s                      w
## 6                        s                        s                      w
##   stalk_color_below_ring veil_type veil_color ring_number ring_type
## 1                      w         p          w           o         p
## 2                      w         p          w           o         p
## 3                      w         p          w           o         p
## 4                      w         p          w           o         p
## 5                      w         p          w           o         e
## 6                      w         p          w           o         p
##   spore_print_color population habitat
## 1                 k          s       u
## 2                 n          n       g
## 3                 n          n       m
## 4                 k          s       u
## 5                 n          a       g
## 6                 k          n       g
```

```
class(mushrooms)
```

```
## [1] "data.frame"
```

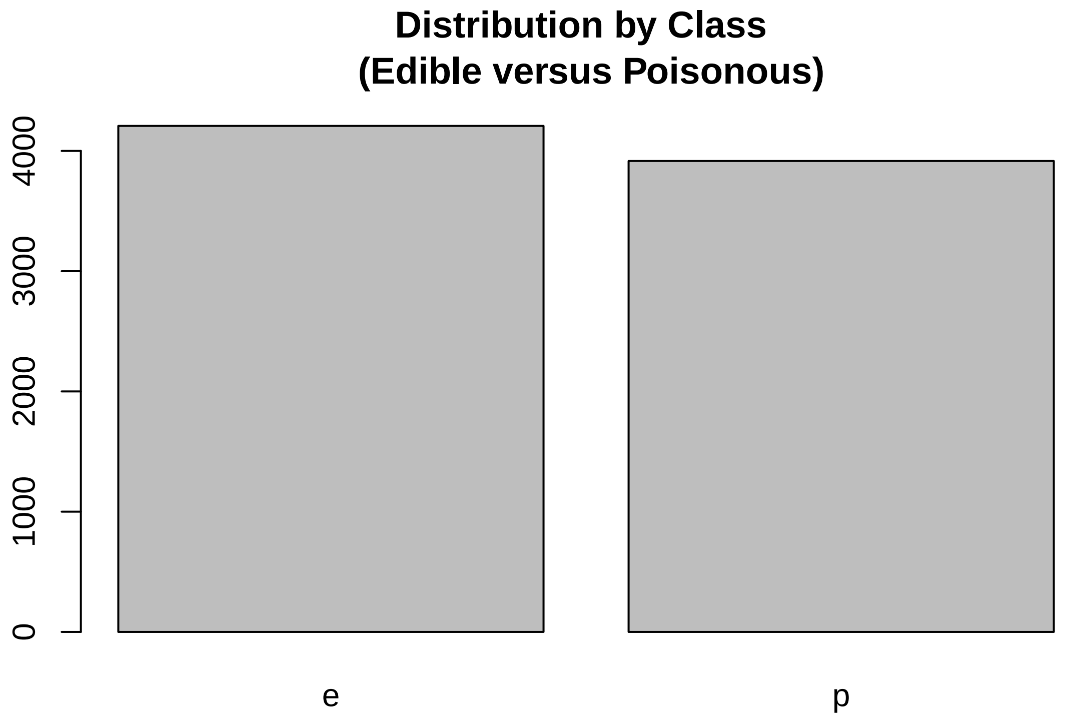
```
dim(mushrooms)
```

```
## [1] 8124   23
```

```r
summary(mushrooms)
```

```
##  class    cap_shape cap_surface   cap_color    bruises       odor     
##  e:4208   b: 452    f:2320      n      :2284   f:4748   n      :3528  
##  p:3916   c:   4    g:   4      g      :1840   t:3376   f      :2160  
##           f:3152    s:2556      e      :1500            s      : 576  
##           k: 828    y:3244      y      :1072            y      : 576  
##           s:  32                w      :1040            a      : 400  
##           x:3656                b      : 168            l      : 400  
##                                 (Other): 220            (Other): 484  
##  gill_attachment gill_spacing gill_size   gill_color   stalk_shape
##  a: 210          c:6812       b:5612    b      :1728   e:3516     
##  f:7914          w:1312       n:2512    p      :1492   t:4608     
##                                         w      :1202              
##                                         n      :1048              
##                                         g      : 752              
##                                         h      : 732              
##                                         (Other):1170              
##  stalk_root stalk_surface_above_ring stalk_surface_below_ring
##  ?:2480     f: 552                   f: 600                  
##  b:3776     k:2372                   k:2304                  
##  c: 556     s:5176                   s:4936                  
##  e:1120     y:  24                   y: 284                  
##  r: 192                                                      
##                                                              
##                                                              
##  stalk_color_above_ring stalk_color_below_ring veil_type veil_color
##  w      :4464           w      :4384           p:8124    n:  96    
##  p      :1872           p      :1872                     o:  96    
##  g      : 576           g      : 576                     w:7924    
##  n      : 448           n      : 512                     y:   8    
##  b      : 432           b      : 432                              
##  o      : 192           o      : 192                              
##  (Other): 140           (Other): 156                              
##  ring_number ring_type spore_print_color population habitat
##  n:  36      e:2776    w      :2388      a: 384     d:3148 
##  o:7488      f:  48    n      :1968      c: 340     g:2148 
##  t: 600      l:1296    k      :1872      n: 400     l: 832 
##              n:  36    h      :1632      s:1248     m: 292 
##              p:3968    r      :  72      v:4040     p:1144 
##                        b      :  48      y:1712     u: 368 
##                        (Other): 144                 w: 192 
```

```r
# The attribute being predicted by this algorithm is labeled "class", with values of "e" ("edible")
# or "p" ("poisonous")
plot(mushrooms$class, main = "Distribution by Class \n (Edible versus Poisonous)")
```



```r
sum(mushrooms$class == "e") / length(mushrooms$class)
```

```
## [1] 0.5179714
```

```r
# Approximately 52% of the mushrooms in the dataset are edible, and roughly 48% are poisonous
```

I divided the data into training (‘mtrain’) and test (‘mtest’) segments (90% and 10% of the data, respectively). I further divided the training dataset into a training subset (‘mtrain_subset’) and a test subset (‘mtest_subset’) to be used for model exploration and evaluation (80% and 20%, respectively).

```r
# Set seed to ensure that partition results will be the same every time the code is executed
# If using R 3.5 or earlier, use `set.seed(1)` instead
set.seed(1, sample.kind="Rounding")

# Partition data into training and test sets
# Test set will be 10% of mushroom data
test_index <- createDataPartition(y = mushrooms$class, times = 1, p = 0.1, list = FALSE)
mtrain <- mushrooms[-test_index,]
mtest <- mushrooms[test_index,]

# Create train and test subsets within the training data for algorithm evaluation
# Evaluation test subset will be 20% of training data
test_index_subset <- createDataPartition(y = mtrain$class, times = 1, p = 0.2, list = FALSE)
mtrain_subset <- mtrain[-test_index,]
mtest_subset <- mtrain[test_index,]
```

```r
# Remove temporary objects
rm(dl, test_index, test_index_subset)
```

I first explored making predictions using a classification tree method. Specifically, I used the RPART function (Recursive Partitioning and Regression Trees) available in the "rpart" R library. This function examines the data and determines the best ways to branch the algorithm for accurate prediction.

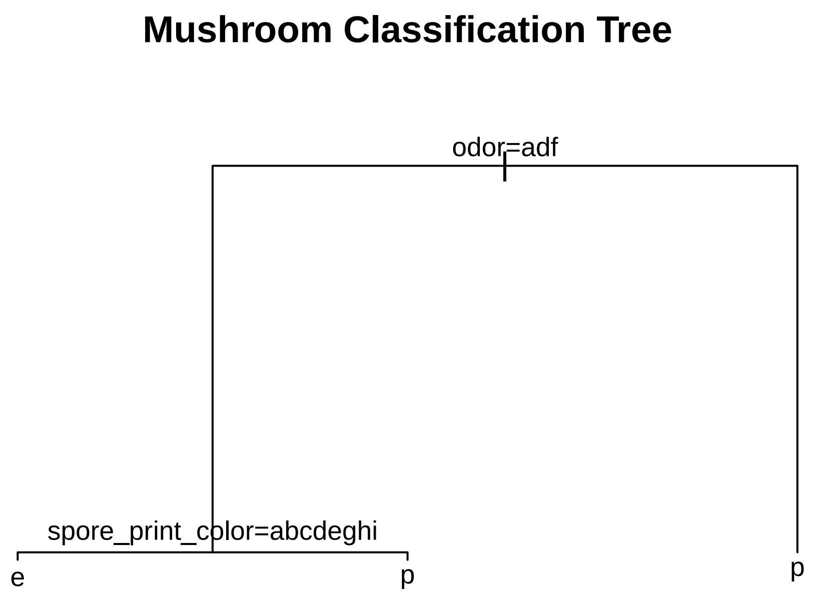
```r
# Use rpart to generate a classification tree
rp_fit <- rpart(class ~ .,
                data = mtrain_subset)

# Visualize classification tree structure and summary of nodes generated
printcp(rp_fit)
```

```
##
## Classification tree:
## rpart(formula = class ~ ., data = mtrain_subset)
##
## Variables actually used in tree construction:
## [1] odor              spore_print_color
##
## Root node error: 3175/6593 = 0.48157
##
## n= 6593
##
##         CP nsplit rel error   xerror      xstd
## 1 0.969764      0  1.000000 1.000000 0.0127783
## 2 0.018898      1  0.030236 0.030236 0.0030634
## 3 0.010000      2  0.011339 0.011339 0.0018846
```

```r
summary(rp_fit)
```

```
## Call:
## rpart(formula = class ~ ., data = mtrain_subset)
##   n= 6593
##
##           CP nsplit  rel error     xerror        xstd
## 1 0.96976378      0 1.00000000 1.00000000 0.012778287
## 2 0.01889764      1 0.03023622 0.03023622 0.003063422
## 3 0.01000000      2 0.01133858 0.01133858 0.001884597
##
## Variable importance
##                     odor        spore_print_color               gill_color
##                       25                       19                       15
## stalk_surface_below_ring stalk_surface_above_ring                ring_type
##                       14                       13                       13
##
## Node number 1: 6593 observations,    complexity param=0.9697638
##   predicted class=e  expected loss=0.4815714  P(node) =1
##     class counts:  3418  3175
##    probabilities: 0.518 0.482
##   left son=2 (3514 obs) right son=3 (3079 obs)
##   Primary splits:
##       odor                     splits as  LRRLRLRRR,    improve=3105.267, (0 missing)
##       spore_print_color        splits as  LRLLLRLRL,    improve=1772.838, (0 missing)
##       gill_color               splits as  RLRRLLLLRLLL, improve=1250.248, (0 missing)
##       stalk_surface_above_ring splits as  LRLL,         improve=1137.090, (0 missing)
##       stalk_surface_below_ring splits as  LRLL,         improve=1107.941, (0 missing)
##   Surrogate splits:
##       spore_print_color        splits as  LRLLLLLRL,    agree=0.861, adj=0.703, (0 split)
##       gill_color               splits as  RLRRLLLLLLLL, agree=0.811, adj=0.595, (0 split)
##       stalk_surface_below_ring splits as  LRLL,         agree=0.785, adj=0.539, (0 split)
##       stalk_surface_above_ring splits as  LRLL,         agree=0.782, adj=0.533, (0 split)
##       ring_type                splits as  RLRRL,        agree=0.782, adj=0.532, (0 split)
##
## Node number 2: 3514 observations,    complexity param=0.01889764
##   predicted class=e  expected loss=0.02731929  P(node) =0.5329895
##     class counts:  3418    96
##    probabilities: 0.973 0.027
##   left son=4 (3454 obs) right son=5 (60 obs)
##   Primary splits:
##       spore_print_color      splits as  LLLLLRLLL,    improve=115.50510, (0 missing)
##       gill_color             splits as  -LLLLLLLRLLL, improve= 39.97542, (0 missing)
##       stalk_color_below_ring splits as  --LLLLLLR,    improve= 34.23524, (0 missing)
##       cap_color              splits as  RLLLLRLLLL,   improve= 20.62596, (0 missing)
##       ring_number            splits as  -LR,          improve= 10.18787, (0 missing)
##   Surrogate splits:
##       gill_color splits as  -LLLLLLLRLLL, agree=0.989, adj=0.35, (0 split)
##
## Node number 3: 3079 observations
##   predicted class=p  expected loss=0  P(node) =0.4670105
##     class counts:     0  3079
##    probabilities: 0.000 1.000
##
## Node number 4: 3454 observations
##   predicted class=e  expected loss=0.0104227  P(node) =0.523889
##     class counts:  3418    36
##    probabilities: 0.990 0.010
##
## Node number 5: 60 observations
##   predicted class=p  expected loss=0  P(node) =0.009100561
##     class counts:     0    60
##    probabilities: 0.000 1.000
```

```r
plot(rp_fit, margin = 0.1, main = "Mushroom Classification Tree")
text(rp_fit, cex = 0.8)
```



The Confusion Parameter (CP) is a key performance indicator of the classification tree method. The objective is to find a model that minimizes the value of CP. Refining the classification tree to use the smallest CP value is called "pruning" the tree.

```r
# Determine lowest CP to create pruned tree
min_cp <- rp_fit$cptable[which.min(rp_fit$cptable[,"xerror"]), "CP"]
pruned <- prune(rp_fit, cp = min_cp)
```

I used the pruned tree to generate classification predictions for the training subset, and then evaluated the success of the predictions using a confusion matrix table. The table identifies how many of the samples in the training subset were edible and how many were poisonous, and compares those counts with the model's predictions.

```r
# Generate classification predictions on training data and view as confusion matrix table
table(mtrain_subset$class, predict(pruned, type = "class"), dnn = c("Actual", "Predicted"))
```

```
##       Predicted
## Actual    e    p
##      e 3418    0
##      p   36 3139
```

The results of the classification tree predictions were fairly good - of the 6,593 total observations in the training subset, only 36 were misclassified, which is an error rate of approximately 0.5%. However, the incorrect classifications were 36 poisonous samples that were classified as edible, which is problematic. To try to reduce the number of false-positive results, I next applied the random forest model, which generates 500 classification trees using samples of the training subset (which are randomly selected with replacement, using the bootstrap method). The accuracy of the 500 classification trees is evaluated, and the best model is selected.

```r
# Use RandomForest to generate a series of classification trees
rf_fit <- randomForest(class ~ .,
                       data = mtrain_subset)

# View RandomForest results
print(rf_fit)
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = mtrain_subset)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 0%
## Confusion matrix:
##      e    p class.error
## e 3418    0           0
## p    0 3175           0
```

```r
# View importance of each predictor
importance(rf_fit)
```

```
##                          MeanDecreaseGini
## cap_shape                        9.324338
## cap_surface                     22.459517
## cap_color                       47.409016
## bruises                         80.492166
## odor                          1143.913921
## gill_attachment                  1.114460
## gill_spacing                    63.485975
## gill_size                      186.856178
## gill_color                     245.391935
## stalk_shape                     39.109247
## stalk_root                      88.184927
## stalk_surface_above_ring       198.382270
## stalk_surface_below_ring       137.491346
## stalk_color_above_ring          55.551769
## stalk_color_below_ring          46.216190
## veil_type                        0.000000
## veil_color                       2.322452
## ring_number                     41.889585
## ring_type                      157.698093
## spore_print_color              479.652194
## population                     142.920226
## habitat                         91.326632
```

I then used the best model to create predictions for the full training subset.

```r
# Create predictions for the training subset and determine accuracy of model
pred <- predict(rf_fit, newdata = mtrain_subset)
confusionMatrix(pred, mtrain_subset$class)$overall["Accuracy"]
```

```
## Accuracy
##        1
```

The algorithm was able to correctly classify all of the specimens in the training subset. I next evaluated how well the model classified the specimens in the test subset.

```
# Create predictions for the test subset and determine accuracy of model to
# ensure the model has not been overfit
confusionMatrix(predict(rf_fit, mtest_subset), mtest_subset$class)$overall["Accuracy"]
```

```
## Accuracy
##        1
```

Predictions generated for the test subset of the training data were also correct. This successful classification method can now be trained on the complete set of training data, and then predictions can be created for the test data.

```
# Execute final prediction, using full training dataset to train model
final_rf_fit <- randomForest(class ~ .,
                             data = mtrain)

# Generate predictions for test data using final model and view results and accuracy
confusionMatrix(predict(final_rf_fit, mtest), mtest$class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   e   p
##          e 421   0
##          p   0 392
##
##                Accuracy : 1
##                  95% CI : (0.9955, 1)
##     No Information Rate : 0.5178
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.5178
##          Detection Rate : 0.5178
##    Detection Prevalence : 0.5178
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : e
##
```

Results

The final model is able to predict whether each specimen is edible or poisonous with 100% accuracy.

Conclusion

The ability to predict whether a mushroom specimen is poisonous based on its physical characteristics can be incredibly useful, especially in a wilderness survival setting.

A potential improvement to this algorithm would be to eliminate characteristics from the algorithm if they contribute little to no predictive power. For example, the initial data exploration showed that all specimens had a "veil_type" value of "p" ("partial"), so the random forest importance results showed a Mean Decrease Gini value of 0 for that predictor. Similarly, the importance results showed only 1.11460 Mean Decrease Gini value for the "gill_attachment" variable, indicating that it contributed very little predictive power. This particular dataset was not tremendously large, but if this algorithm was to be used on a larger dataset, its speed and efficiency could be increased by removing characteristics that are not very useful in making predictions about whether a specimen is edible or poisonous.