Samuel L. Peoples
IST 707 Data Mining
11 December 2018

Final Project Report
Mushroom Classification: Edible or Poisonous?

This project explored the data cleaning and classification necessary to predict whether a mushroom is poisonous or edible. The data was collected from Kaggle's Mushroom Classification dataset[1] Decision Tree and Naïve Bayes Models are compared in their run-time and accuracy to determine which of the two algorithms best predicts the nature of a given mushroom. Data questions this project will explore include:

- Which features are most indicative of poisonous and edible mushrooms?
- Which features are the most ubiquitous across both poisonous and edible mushrooms?
- Given a random mushroom, with how much certainty can its nature be predicted?

Beginning with a dataset with twenty-two categorical features and 8,124 entries, the categories were one-hot encoded, increasing the dimensionality of the dataset to ninety-six boolean features. These categories include *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,* and *habitat*. The data includes descriptions of hypothetical samples relating to twenty-three species of Agaricus and Lepiota mushrooms, each species is identified as either edible or poisonous.

Following the encoding process, removal of one feature is necessary to avoid the dummy-variable trap. These features are represented when all other categories are zero. Categories represented in this manner include *cap-shape-conical, cap-surface-grooves, cap-color-purple, bruises-false, odor-musty, gill-attachment-attached, gill-spacing-crowded, gill-size-narrow, gill-color-green, stalk-shape-enlarging, stalk-root-rooted, stalk-surface-above-ring-scaly, stalk-surface-below-ring-scaly, stalk-color-above-ring-yellow, stalk-color-below-ring-yellow, veil-color-yellow, ring-number-none, ring-type-none, spore-print-color-yellow, population-clustered,* and *habitat-waste*. There was also a categorical variable which was universal for all entries, *veil-type*, of which each entry was *partial*, thus this value was excluded from the encoded dataset, as it would not contribute to the predictions.

It was initially necessary to import the libraries and dataset. The encoded dataset is used to build the most accurate and descriptive model.

```
library(caTools)
library(rpart)
library(rpart.plot)
library(e1071)

fp = redacted
dataset = read.csv(paste(fp,'mushrooms_encoded.csv', sep=''))
```

The data is first validated and cleaned. It is verified that N/A values are not present in the dataset, and the structure of the data is inspected.

```
which(is.na(dataset))
integer(0)
```

[1] UCI Machine Learning repository. (2016, December 01). Mushroom Classification. Retrieved from https://www.kaggle.com/uciml/mushroom-classification

Samuel L. Peoples
IST 707 Data Mining
11 December 2018

```
str(dataset)
'data.frame':     8124 obs. of  97 variables:
 $ class                          : Factor w/ 2 levels "e","p": 2 2 2 2 1...
 $ cap.shape.bell                 : int  0 0 0 0 1 1 1 1 1 1 ...

 $ cap.shape.convex               : int  0 0 0 0 0 0 0 0 0 0 ...
 $ cap.shape.flat                 : int  0 0 0 0 0 0 0 0 0 0 ...
 $ cap.shape.knobbed              : int  0 0 0 0 0 0 0 0 0 0 ...
 $ cap.shape.sunken               : int  0 0 0 0 0 0 0 0 0 0 ...
 $ cap.surface.fibrous            : int  0 0 0 0 0 0 0 0 0 0 ...
[TRUNCATED]
```

The data is explored and in this process, the most common feature for *Both*, *Edible*, and *Poisonous* is found. Where the most common feature across the dataset was a *White Veil*, for *Edible* it was *Free Gills*, and for *Poisonous*, it was a *White Veil*. This suggests that the most common feature may not be very significant in this analysis.

```
# Most common among all
sums <- as.data.frame(colSums(dataset[,-1]))
names(sums) <- "sum"
sums$feature <- rownames(sums)
sums$class <- length(dataset$class)
sums$feature[which.max(sums$sum)]
 "veil.color.white"

# Most common among edible
dataset_e <- dataset[which(dataset$class == 'e'),]
sums <- as.data.frame(colSums(dataset_e[,-1]))
names(sums) <- "sum"
sums$feature <- rownames(sums)
sums$class <- length(dataset$class)
sums$feature[which.max(sums$sum)]
 "gill.attachment.free"

# Most common among poisonous
dataset_p <- dataset[which(dataset$class == 'p'),]
sums <- as.data.frame(colSums(dataset_p[,-1]))
names(sums) <- "sum"
sums$feature <- rownames(sums)
sums$class <- length(dataset$class)
sums$feature[which.max(sums$sum)]
 "veil.color.white"
```

Each of the ninety-seven variables are transformed into factors, and the levels for *class* are labeled as *Poisonous* and *Edible*. Then the structure is inspected once again. This is done because the accuracy of the models is greater with factors.

```
dataset[] <- lapply(dataset, factor)
levels(dataset$class) <- c("Edible", "Poisonous")
str(dataset)
'data.frame':     8124 obs. of  97 variables:
 $ class                          : Factor w/ 2 levels "Edible","Poisonous":
 $ cap.shape.bell                 : Factor w/ 2 levels "0","1": 1 1 1 1 2 2
 $ cap.shape.convex               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1
```

```
 $ cap.shape.flat                : Factor w/ 2 levels "0","1": 1 1 1 1 1 1
 $ cap.shape.knobbed             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1
 $ cap.shape.sunken              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1
 $ cap.surface.fibrous           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1
```

The decision tree analysis was conducted first. The process was timed, and manual five-fold cross validation is completed using a test-set derived from thirty percent of the given data. The *sample.split* function from the *CaTools* library samples equally based on the provided feature to reduce overfitting and poor sampling. The *Decision Tree* classifier is trained using the *rpart* package, with a *minsplit* of thirty, *maxdepth* of ten, and *cp* of 0.01. each classifier is trained and saved to select the most accurate from the analysis. The same process is repeated for the Naïve Bayes classifier, where a *laplace* of 0.01 and *threshold* of 0.1 are used. These parameters where tuned to provide the greatest accuracy.

```
#Decision Trees
dt_result <- c()
dt_start <- proc.time()
for(i in 1:5){
  split <- sample.split(dataset$class, SplitRatio = 0.70)
  test_set <- subset(dataset, split == FALSE)
  training_set <- subset(dataset, split == TRUE)
  # Fitting Decision Tree Classification to the Training Set
  dt_classifier <- rpart(formula = class ~., data = training_set,
                         minsplit = 30, maxdepth = 10, cp = .01)
  if(i == 1){ dt_classifier_1 <- dt_classifier }
  if(i == 2){ dt_classifier_2 <- dt_classifier }
  if(i == 3){ dt_classifier_3 <- dt_classifier }
  if(i == 4){ dt_classifier_4 <- dt_classifier }
  if(i == 5){ dt_classifier_5 <- dt_classifier }
  dt_pred <- predict(dt_classifier, newdata = test_set[-1], type = 'class')
  dt_cm <- table(test_set[,1], dt_pred)
  dt_acc <- (dt_cm[1]+dt_cm[4]) / (dt_cm[1]+dt_cm[2]+dt_cm[3]+dt_cm[4])
  dt_result <- c(dt_result,dt_acc)
}
dt_end = proc.time()

#Naive Bayes
nb_result <- c()
nb_start <- proc.time()
for(i in 1:5){
  split <- sample.split(dataset$class, SplitRatio = 0.70)
  test_set <- subset(dataset, split == FALSE)
  training_set <- subset(dataset, split == TRUE)
  nb_classifier <- naiveBayes(formula = class~.,data = training_set,
                              laplace = .01, threshold = .1)
  if(i == 1){ nb_classifier_1 <- nb_classifier }
  if(i == 2){ nb_classifier_2 <- nb_classifier }
  if(i == 3){ nb_classifier_3 <- nb_classifier }
  if(i == 4){ nb_classifier_3 <- nb_classifier }
  if(i == 5){ nb_classifier_3 <- nb_classifier }
  # Predicting the Test Set results
  nb_pred <- predict(nb_classifier, newdata = test_set[-1])
  # Creating the confusion_matrix
  nb_cm <- table(test_set[,1], nb_pred)
```

```
  nb_acc <- (nb_cm[1]+nb_cm[4]) / (nb_cm[1]+nb_cm[2]+nb_cm[3]+nb_cm[4])
  nb_result <- c(nb_result,nb_acc)
}
nb_end <- proc.time()
```

Finally, the results are prepared, displaying the run-time and accuracy of each model that was developed. The *Decision Tree* classifier ran over eight times as fast as the *Naïve Bayes*, taking 1.7 seconds to complete the training of five models, averaging .34 seconds per model, where the *Naïve Bayes* classifier performed the same number of training cycles in 13.9 seconds, averaging 2.78 seconds per model. However, the *Decision Tree* performed with 99.14% accuracy, on average, where the *Naïve Bayes* performed with only 96.51% accuracy; this is 2.63% more accurate. Thus, the second *Decision Tree* model is selected for continued analysis.

```
print(paste("DT Result:", dt_result[1],dt_result[2]
             ,dt_result[3],dt_result[4],dt_result[5]))
dt_result <- (dt_result[1]+dt_result[2]+dt_result[3])/3
print(paste("DT Average:",dt_result))
print(paste("Time:",(dt_end-dt_start)[3],"seconds"))
 "DT Result: 0.9881 0.9955 0.9906 0.9885 0.9922"
 "DT Average: 0.9914"
 "Time: 1.7 seconds"

print(paste("NB Result:",nb_result[1],nb_result[2]
            ,nb_result[3],nb_result[4],nb_result[5]))
nb_result <- (nb_result[1]+nb_result[2]+nb_result[3])/3
print(paste("NB Average:",nb_result))
print(paste("Time:",(nb_end-nb_start)[3],"seconds"))
 "NB Result: 0.9639 0.9635 0.9680 0.9655 0.9565"
 "NB Average: 0.9651"
 "Time: 13.9 seconds"
```

The selected model is visualized using *rpart.plot* (figure 1). This revealed that the most distinguishing features for *Edible* mushrooms include *No Odor*, and *Spore Print Color Not Green*. If a mushroom does have an *Odor*, it is most likely *Poisonous*. Less significant features include a *Clubbed Stalk Root*, and *Anise* or *Almond Odor*, where having these features typically indicate an *Edible* mushroom.

```
rpart.plot(dt_classifier_2)
```

In conclusion, the most accurate classifier was also the fastest classifier, with an average of .34 seconds for training, with an accuracy of 99.55%. This was unexpected, because *Naïve Bayes* has typically performed with greater accuracy than *Decision Trees* in the past. The most common features for all mushrooms and *Poisonous* mushrooms is a *White Veil*, while the most common feature for *Edible* mushrooms is a *Free Gill Attachment*. Features which indicate E*dible* mushrooms include *No Odor*, and *Spore Print Color Not Green*, which explains 43% of the data. Features which indicate a *Poisonous* mushroom include having an *Odor*, and not having a *Stalked Root Club* or *Anise* or *Almond Odor*, explaining 46% of the data. These two indicators together explain 89% of the data with 99.55% accuracy.

Samuel L. Peoples
IST 707 Data Mining
11 December 2018

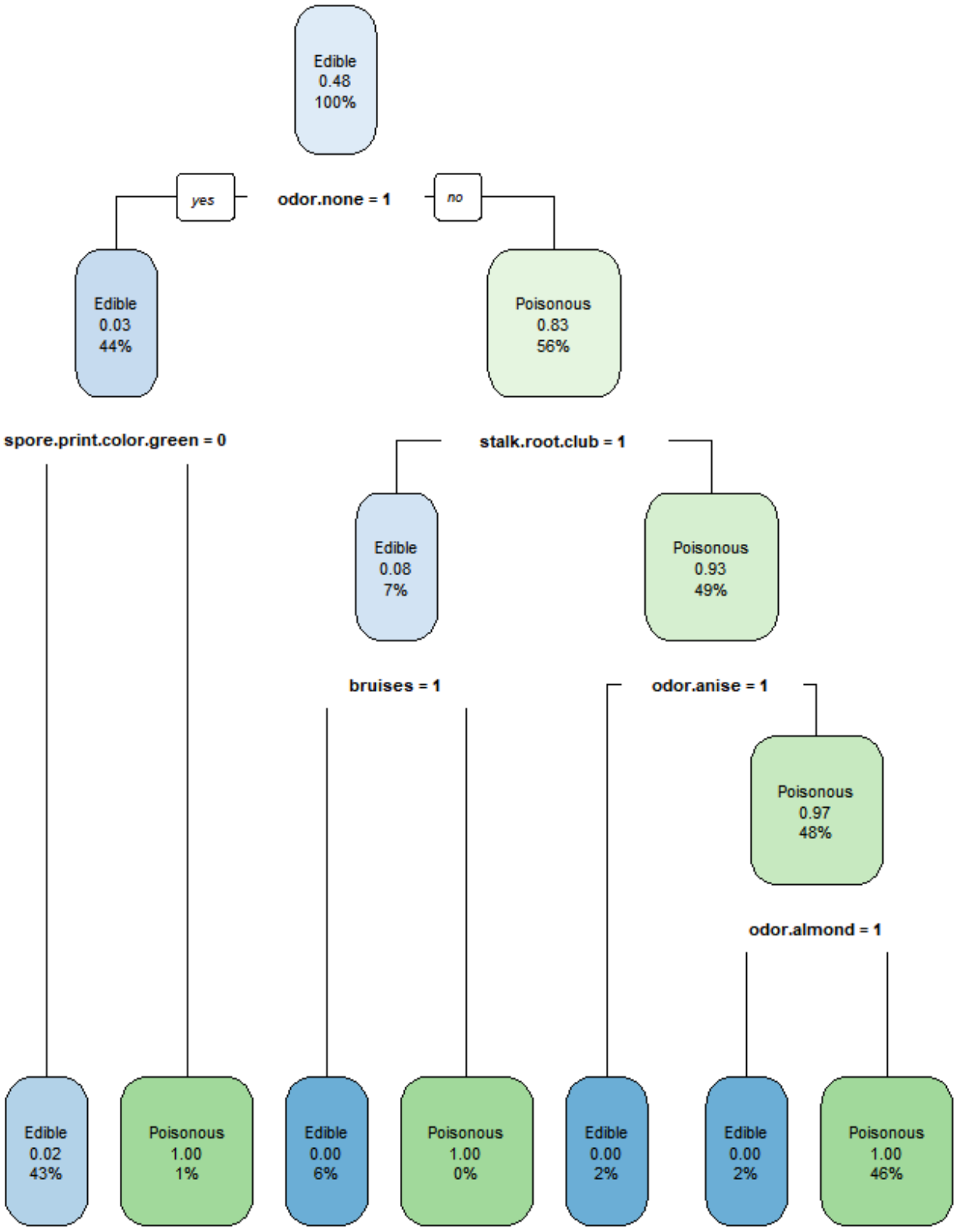

Figure 1: Plot of `dt_classifier_2`

Samuel L. Peoples
IST 707 Data Mining
11 December 2018

References

1. UCI Machine Learning repository. (2016, December 01). Mushroom Classification. Retrieved from https://www.kaggle.com/uciml/mushroom-classification