Mushroom Classification

Poisonous or Edible?

Samuel L. Peoples

IST 707: Data Mining

11 December 2018

Introduction

- Goal: Predict whether mushroom is poisonous or edible from Kaggle UCI Machine Learning Mushroom dataset.
- Models: Decision Tree & Naïve Bayes using five-fold cross validation.
- Data Questions:
  - Which features are most indicated of poisonous and edible mushrooms?
  - Which features are the most ubiquitous across both poisonous and edible mushrooms?
  - Given a random mushroom, which how much certainty can its nature be predicted?

Data

- Initially twenty-two categorical features and one predictor
- One-hot encoded to boolean (0,1) columns, increasing dimensionality to ninety-six features and one predictor.
- Dropped low-value entries (represented by all-zeros) to avoid dummy-variable trap.

Importing Data

- Encoded data and libraries are imported.

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

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

Data Validation

- N/A values are checked-for, and the structure is inspected. No N/A values are found, noting the boolean features are *int*.

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

```
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]
```

Data Exploration:
Most Common Feature Among All

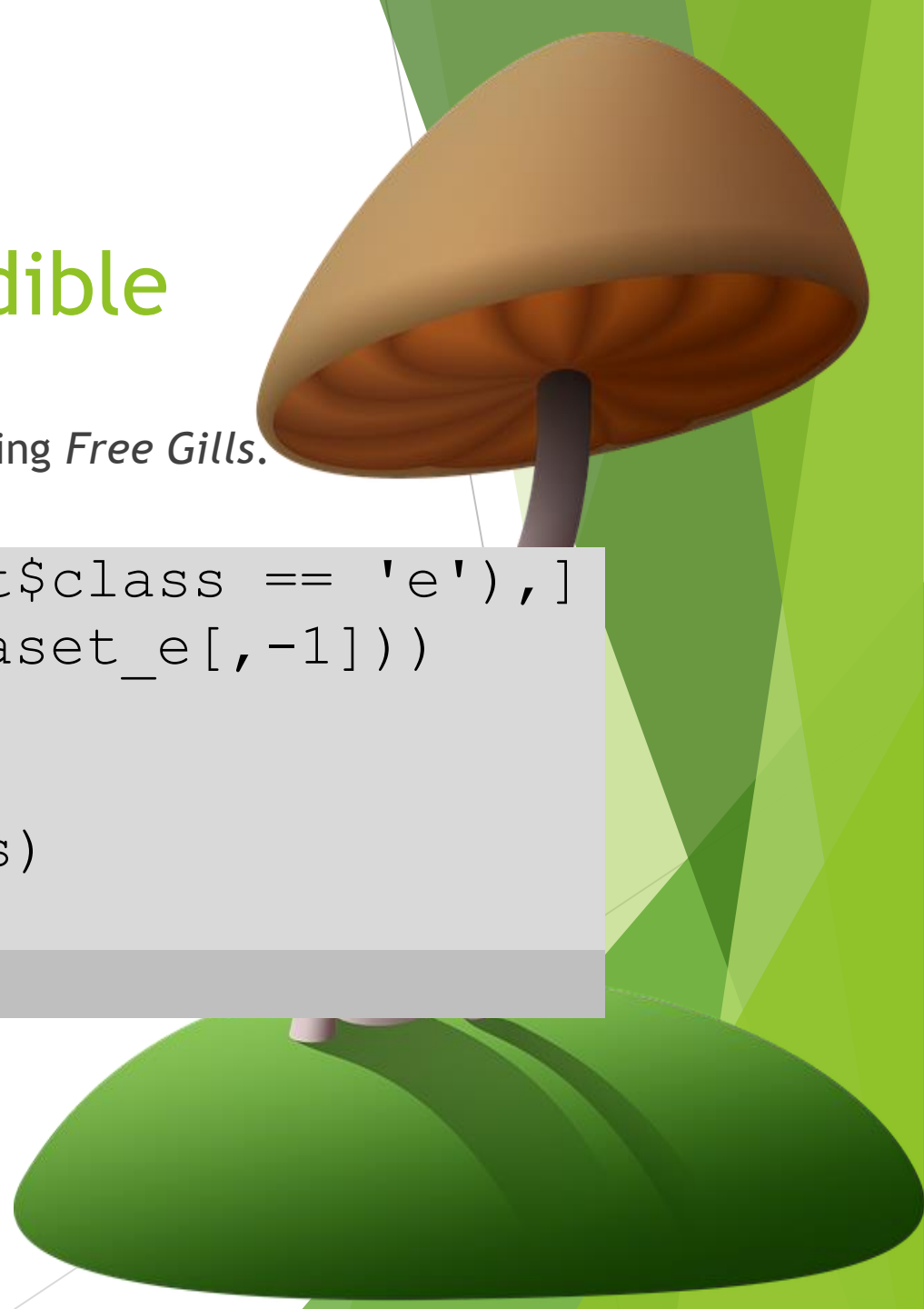
- Most common feature is found for the entire dataset, being a *White Veil*.

```
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"
```

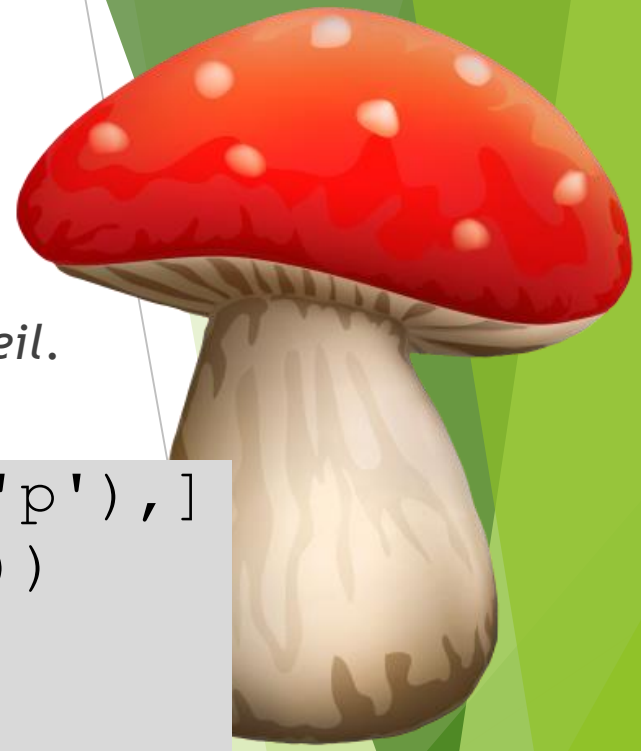
Data Exploration:
Most Common Feature Among Edible

- Most common feature is found for the edible mushrooms, being *Free Gills*.

```
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"
```



Data Exploration:
Most Common Feature Among Poisonous



- Most common feature is found for the poisonous mushrooms, being a *White Veil*.

```
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"
```

Data Transformation

- Features are encoded as factors and *class* levels are named. This will improve overall accuracy of models.

```
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
```

Decision Tree Classifier

- Tuned for accuracy, using five-fold cross validation, minsplit = 30, maxdept = 10, cp = .01. Process is timed for evaluation. Models saved to select highest accuracy.

```
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()
```

Naïve Bayes Classifier

- Tuned for accuracy, using five-fold cross validation, laplace = 0.01, threshold = 0.1. Process is timed for evaluation. Models saved to select highest accuracy.

```
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()
```

Decision Tree Results

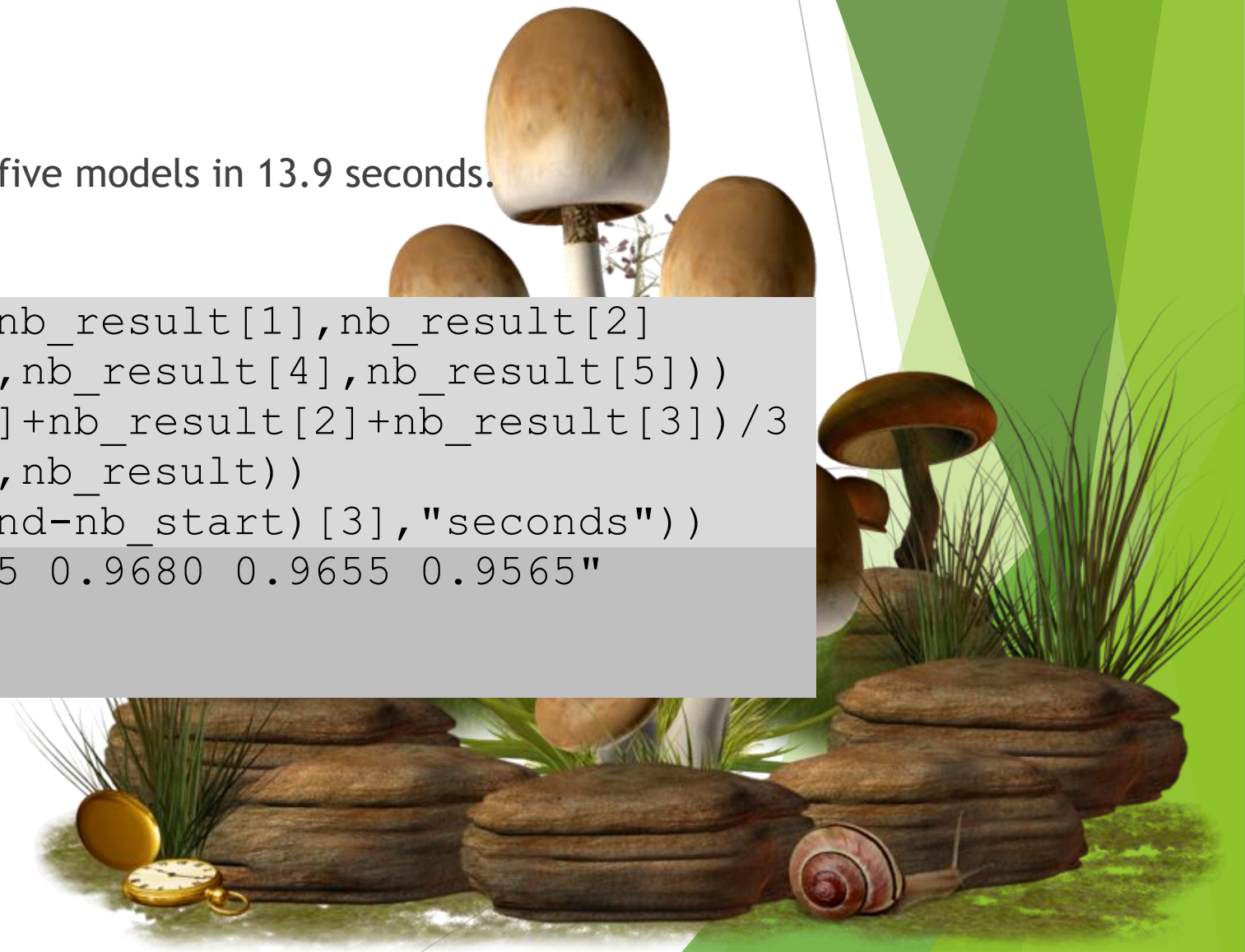
- Highest accuracy was 99.55%, training five models in 1.7 seconds.
- Average: 99.14% Accuracy

```
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"
```

Naïve Bayes Results

- Highest accuracy was 96.8%, training five models in 13.9 seconds.
- Average: 96.51% Accuracy

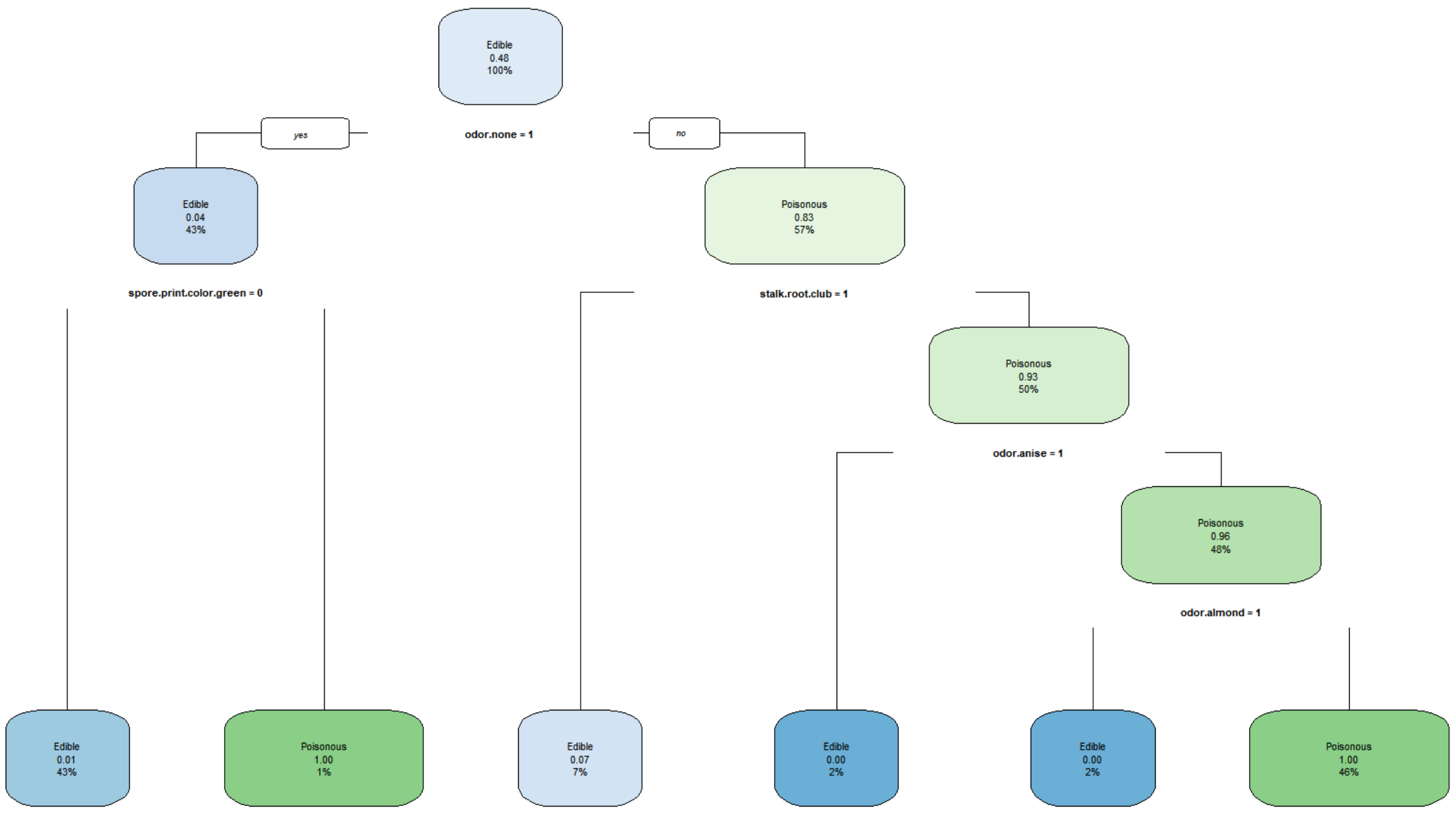
```
print(paste("NB Result:",nb_result[1],nb_result[2]
            ,nb_result[3],nb_result[4],nb_result[5]))
nb_result <- (nb_result[1]+nb_result[2]+nb_result[3])/3
print(paste("NB Average:",nb_result))
print(paste("Time:",(nb_end-nb_start)[3],"seconds"))
 "NB Result: 0.9639 0.9635 0.9680 0.9655 0.9565"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



Selected Model

- The second decision tree model is selected because the average accuracy was greatest for the decision tree models, and the second had the greatest accuracy overall.

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


Edible
0.48
100%
odor.none = 1
yes
Edible
0.04
43%
spore.print.color.green = 0
no
Poisonous
0.83
57%
stalk.root.club = 1
Edible
0.01
43%
Poisonous
1.00
1%
Edible
0.07
7%
Poisonous
0.93
50%
odor.anise = 1
Edible
0.00
2%
Poisonous
0.96
48%
odor.almond = 1
Edible
0.00
2%
Poisonous
1.00
46%

Conclusion

- The *Decision Tree* classifier performed with greater accuracy - nearly eight times faster.
- Which features are most indicative of poisonous and edible mushrooms?
  - 43% of all data suggest that *No Odor* and *Spore Print Color Not Green* indicates *Edible*
  - 46% of all data suggest that having an *Odor* that is neither *Anise*, or *Almond*, and not having a *Clubbed Stalk Root* indicates *Poisonous*.
- Which features are the most ubiquitous across both poisonous and edible mushrooms?
  - *Edible* Mushrooms: *Free Gills*
  - *Poisonous & All* Mushrooms: *White Veil*
- Given a random mushroom, with how much certainty can its nature be predicted?
  - 99.55% accuracy using the second *Decision Tree* model.