Report on Mushroom Dataset using Random Forest, Naïve Bayes, and Decision Tree Algorithm

By: Dimple Chowdhari

Table of content:

Introduction

A random forest is an ensemble approach that consists of numerous individual decision trees. The ensemble approach uses the divide and conquers method. The principle is that a "weak learners" group can come together to form a “strong learner”. (Benyamin, 2020) The random forest has the name forest because it is a collection of many decision trees instead of depending on a single decision tree. A collection of trees is called a forest.

It is a classification algorithm that consists of numerous decision trees. It uses bagging and feature randomness when building each tree, it uses bagging and randomness and creates a forest of trees that are not correlated whose prediction accurate than that of an individual tree. (Benyamin, 2020)

Naïve Bayes classifier is a probabilistic machine learning model used for the classification task. Machine learning studies the development of computer processes to transform data into intelligent action. (Lantz, 2013).

Naïve Bayes is called “naïve” because it makes the impression that the occurrence of a certain attribute is independent of the occurrence of other attributes. In simpler terms, it assumes that in a class, the presence of a particular feature is not linked to the other features present. (Ray, 2017)

Decision Trees are a non-parametric supervised learning method used for classification and regression. They are a tree graph model with the terminal nodes representing classification outcomes and decisions. The decisions are based on the combination of experiences from solving similar cases, scientific research results, and personal experience. (Science, 2020)

Decision trees can be used in health care industries, business development, project management, the criminal justice system, and sports. They help organize and manage raw data with minimum preprocessing. ("Uses of Decision Trees in Business Data Mining | Research Optimus", 2015)

The decision tree breaks down a data set into smaller and smaller subsets. The leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called the root node. (Sehra, 2018)

The objective of the mushroom dataset was to identify edible and poisonous mushrooms using its features using Random forest, Naïve Bayes, and Decision Trees models.

Analysis

Below is the mushroom data set with 8124 observations and 23 variables which consist of names such as type, odor, gill attach, stalk root, veil color, etc. The source of data is at the UCI Machine Learning Repository website. The objective of the data is to identify the poisonous and edible mushrooms using the variables. (Lantz, 2013).

DATA EXPLORATION

```
dim(mushroom)

## [1] 8124   23
```

```
> head(mushroom)
  type cap_shape cap_surface cap_color bruises odor gill_attach gill_spacing gill_size gill_color stalk_shape stalk_root
1    p         x           s         n       t    p           f            c         n          k           e          e
2    e         x           s         y       t    a           f            c         b          k           e          c
3    e         b           s         w       t    l           f            c         b          n           e          c
4    p         x           y         w       t    p           f            c         n          n           e          e
5    e         x           s         g       f    n           f            w         b          k           t          e
6    e         x           y         y       t    a           f            c         b          n           e          c
  stalk_surface_abo stalk_surface_bel stalk_color_abo stalk_color_bel veil_color ring_number ring_type spore_color pop habitat
1                 s                 s               w               w          w           o         p           k   s       u
2                 s                 s               w               w          w           o         p           n   n       g
3                 s                 s               w               w          w           o         p           n   n       m
4                 s                 s               w               w          w           o         p           k   s       u
5                 s                 s               w               w          w           o         e           n   a       g
6                 s                 s               w               w          w           o         p           k   n       g
```

The variables names are:

```
names(mushroom)

##  [1] "type"              "cap_shape"         "cap_surface"
##  [4] "cap_color"         "bruises"           "odor"
##  [7] "gill_attach"       "gill_spacing"      "gill_size"
## [10] "gill_color"        "stalk_shape"       "stalk_root"
## [13] "stalk_surface_abo" "stalk_surface_bel" "stalk_color_abo"
```

The structure of the data is made up of factor variables. The mushroom species were classified as poisonous or edible which was used to establish the edibility of the mushroom. The data included information about 8,124 mushroom samples from 23 species of gilled mushrooms.

Random forest, Naïve Bayes and Decision Trees were used to model the data.

The five-number summary which consists of range, median, and quartiles is carried out. This method provides a way to determine the shape of the dataset. Using the smallest and largest values of each variable and comparing the distance from the median recognize the shape of the dataset. (Levine & Stephan, 2013)

```
summary(mushroom)

##  type     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_attach gill_spacing gill_size   gill_color   stalk_shape stalk_root
##  a: 210      c:6812       b:5612    b      :1728   e:3516      ?:2480
##  f:7914      w:1312       n:2512    p      :1492   t:4608      b:3776
##                                     w      :1202               c: 556
##                                     n      :1048               e:1120
##                                     g      : 752               r: 192
##                                     h      : 732
##                                     (Other):1170
```

Veil_type does not vary across samples; it does not provide any useful information for prediction. It eliminates the feature (partial) from the mushroom's data frame.

```
#samples, it does not provide
mushroom$veil_type <- NULL
```

The data has 4,208 edible mushrooms and 3916 poisonous mushrooms.

```
table(mushroom$type)

##
##    e    p
## 4208 3916
```

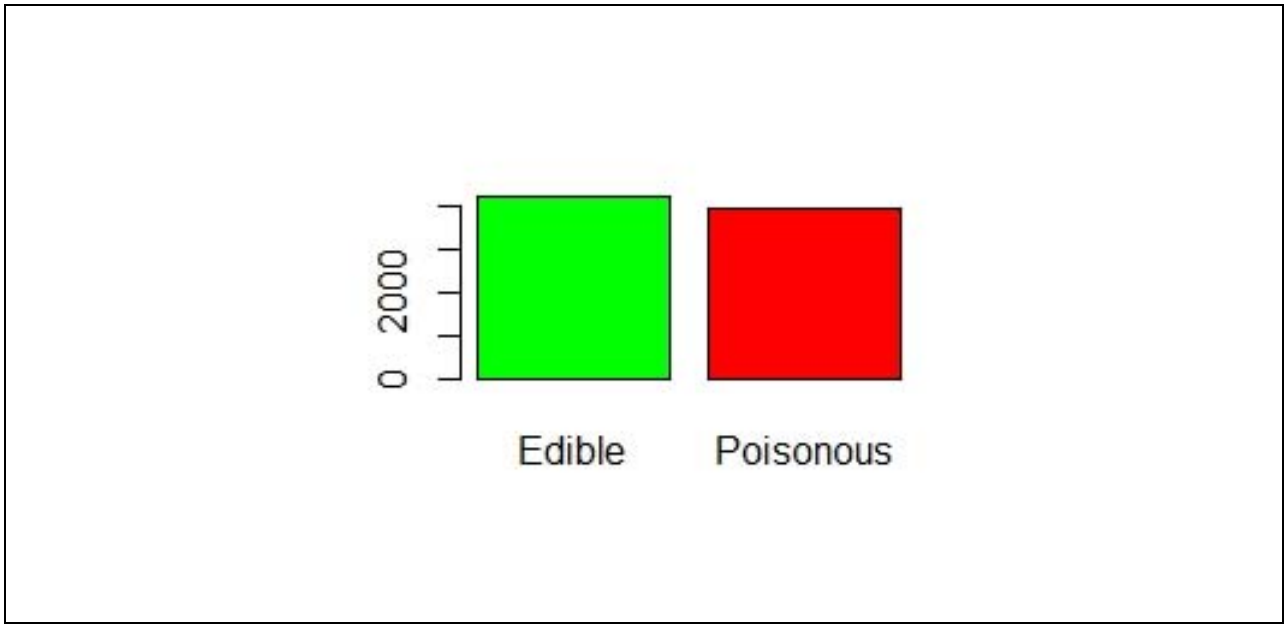
52 percent of the mushroom samples are edible, while 48 percent are poisonous.

```
round(prop.table(table(mushroom$type))*100)

##
##  e  p
## 52 48
```

EXPLORATORY DATA ANALYSIS

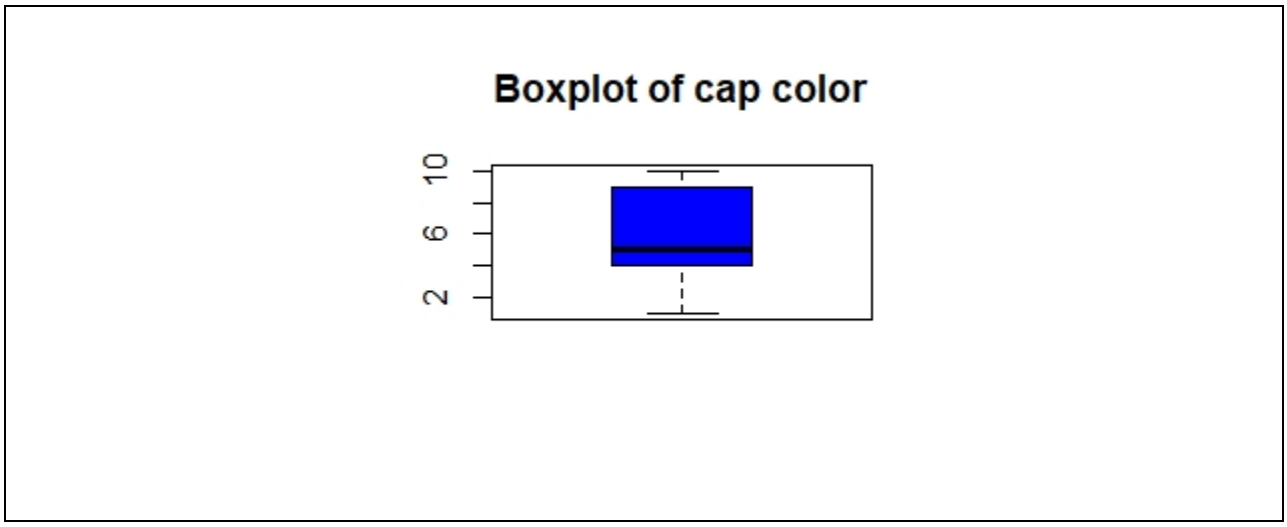
The barplot suggests that the amount of edible mushrooms is higher compared to poison mushrooms. The class levels are approximately 50% each.

BARPLOT FOR EDIBLE AND POISONOUS MUSHROOMS




```
> ##EDA
> nb_value_counts = table(as.numeric(df$type))
> nb_value_counts

   1    2
4208 3916
> barplot(nb_value_counts, names.arg=c('Edible', 'Poisonous'), col=c('Green', 'Red'))
> boxplot(as.numeric(df$cap_color), col="blue", main="Boxplot of cap color")
> summary(as.numeric(df$cap_color))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   4.000   5.000   5.505   9.000  10.000
```

The boxplot represents a graphical representation based on the five-number summary. This shows that a plot against cap color is a right-skewed distribution. Since the distance between the smallest value and median is less than the distance from the median to the largest value. The median is at 5 units.



MODEL TRAINING

To prepare the data for modeling, we split the data into test and train datasets. The training will use 75 percent of the data and 25 percent for testing, which will provide us with 100 records to simulate new features. The training dataset builds the model and a test dataset to evaluates the performance of the model on new data.

```
> ##train and test model
> df <- mushroom
> smp_size <- floor(0.75 * nrow(df))
> ## set the seed to make your partition reproducible
> set.seed(1)
> train_ind <- sample(seq_len(nrow(df)), size = smp_size)
> train <- df[train_ind, ]
> head(train)
     type cap_shape cap_surface cap_color bruises odor gill_attach gill_spacing
1017    e         f           y         n       t    l           f            c
8004    e         f           s         n       f    n           a            c
4775    p         x           f         y       f    f           f            c
2177    e         f           f         n       t    n           f            c
5026    p         f           y         g       f    f           f            c
1533    p         f           s         w       t    p           f            c
     gill_size gill_color stalk_shape stalk_root stalk_surface_abo stalk_surface_bel
1017         b          n           e          r                 s                 y
8004         b          n           e          ?                 s                 s
4775         b          h           e          b                 k                 k
2177         b          w           t          b                 s                 s
5026         b          g           e          b                 k                 k
1533         n          n           e          e                 s                 s
```

```
> head(test)
   type cap_shape cap_surface cap_color bruises odor gill_attach gill_spacing
5     e         x           s         g       f    n           f            w
8     e         b           y         w       t    l           f            c
10    e         b           s         y       t    a           f            c
11    e         x           y         y       t    l           f            c
12    e         x           y         y       t    a           f            c
13    e         b           s         y       t    a           f            c
   gill_size gill_color stalk_shape stalk_root stalk_surface_abo stalk_surface_bel
5          b          k           t          e                  s                  s
8          b          n           e          c                  s                  s
10         b          g           e          c                  s                  s
11         b          g           e          c                  s                  s
12         b          n           e          c                  s                  s
13         b          w           e          c                  s                  s
```

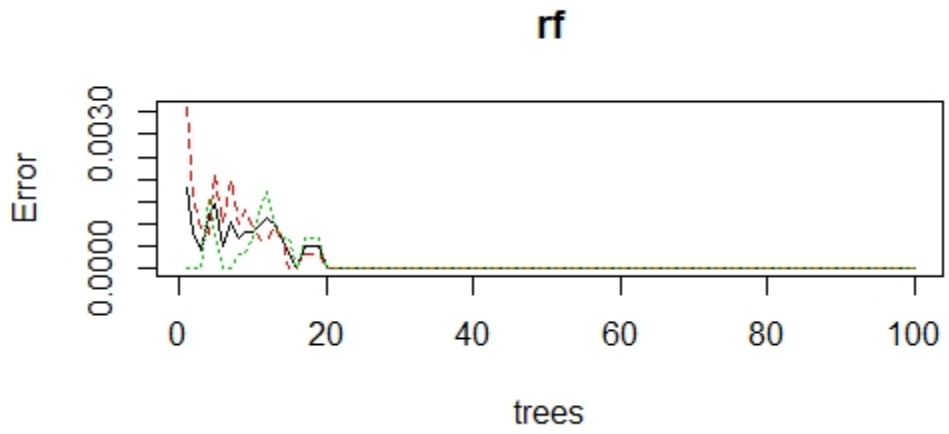
MODEL EVALUATION

RANDOM FOREST

Random forest runtimes are quite fast, and they can deal with a large number of missing data. It has an unexcelled accuracy among current algorithms. Efficiently runs well on large data sets and gives estimates which variables are important in the classification. (Benyamin, 2020)

The disadvantage creates a lot of trees and combines their outputs and it takes a longer training period to train as compared to decision trees as it generates a lot of trees (instead of one tree in case of the decision tree) and makes a decision on the majority of votes. (Kumar, 2020)

To carry out the random forest model, the package random forest was installed. Fitting the random forest model to the training data was done. The plot below indicates that after about 20 trees, there are no changes in errors. The error fluctuates at the beginning of the trees.



The number of variables tried at each split to be 4 and an OOB estimate of error rate 0.06%. The training model fits the training data almost perfectly. There was only one mushroom which was classified incorrectly. The model would have predicted 1 to be poisonous and it would have turned out to be edible.

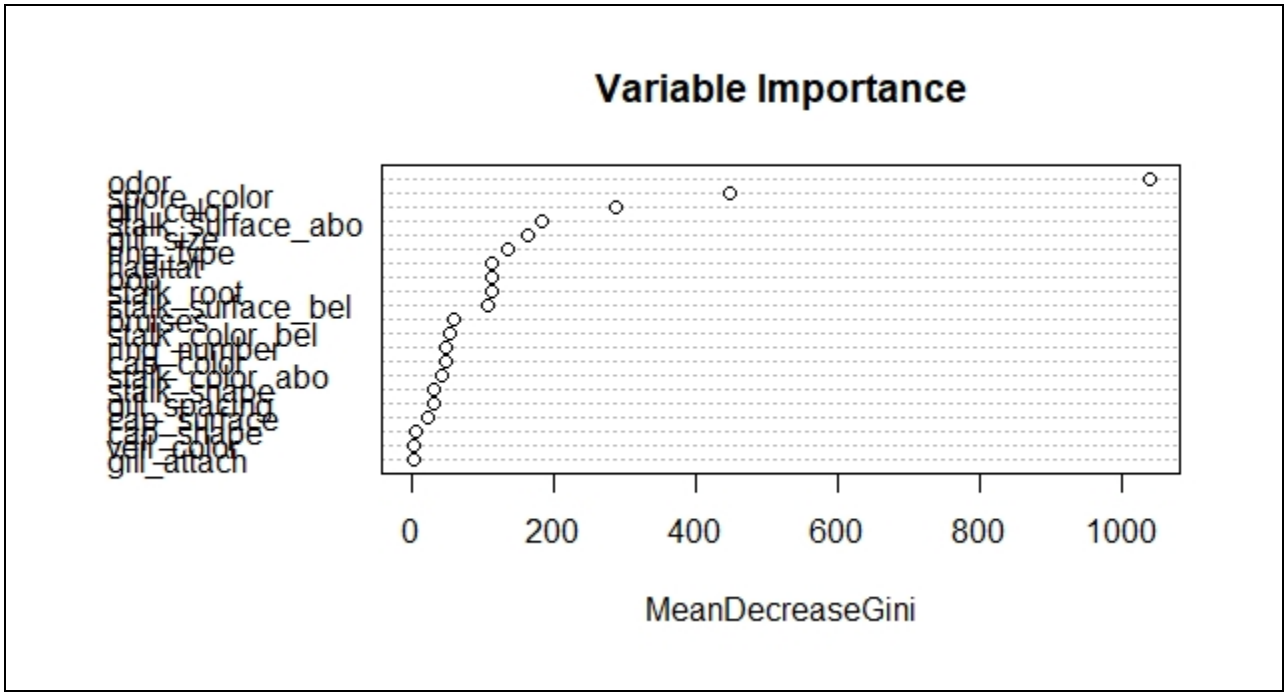
As expected, the output notes that the random forest included 100 trees and tried 4 variables at each split. According to the display confusion matrix, the error rate is 0% percent which is not worse than the other ensemble methods. This reflects the out-of-bag error rate (labeled OOB estimate of error rate), which is an unbiased estimate of the test set error. This means that it should be a fairly reasonable estimate of future performance. (Lantz, 2013)

```
> rf = randomForest(type ~ .,
+                   ntree = 100,
+                   data=train)
> plot(rf)
> print(rf)

Call:
 randomForest(formula = type ~ ., data = train, ntree = 100)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 4

        OOB estimate of  error rate: 0%
Confusion matrix:
     e    p class.error
e 3148    0           0
p    0 2945           0
```

The training dataset was used to model the data to predict whether the mushroom was edible or poisonous. It predicted the response accurately with zero false positives and zero false negatives. It made a prediction accuracy of 100% with a confidence interval of 95%. Kappa = 1 means that the prediction was correct 100% of the time.

```
> # Create Confusion Matrix
> coMa <- confusionMatrix(test$predicted.response,as.factor(test$predicted.response))
> coMa
Confusion Matrix and Statistics

          Reference
Prediction    e    p
         e 1060    0
         p    0  971

               Accuracy : 1
                 95% CI : (0.9982, 1)
    No Information Rate : 0.5219
    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 : 0.5219
         Detection Rate : 0.5219
   Detection Prevalence : 0.5219
      Balanced Accuracy : 1.0000

       'Positive' Class : e

> acc_rt <- coMa$overall["Accuracy"]
> acc_rt
Accuracy
       1
```

Below is a list of the variables that predict whether the mushroom is poisonous or edible. The odor is the first significant variable that can detect edibility or poisonous followed by spore color, gill color, gill size, etc.



The odor is the most important variable according to mean decreasing Gini i.e. information gain. Veil Type created no information gain as there was only one VeilType, hence it does not impact the classification results.

```
> print(var.imp[order(var.imp$MeanDecreaseGini,decreasing = T),])
                  MeanDecreaseGini         Variables
odor                1041.9748530              odor
spore_color          449.4488773       spore_color
gill_color           286.6114929        gill_color
stalk_surface_abo    181.4967408 stalk_surface_abo
gill_size            162.9707182         gill_size
ring_type            135.1907583         ring_type
habitat              113.2027263           habitat
pop                  112.2509866               pop
stalk_root           111.1131890        stalk_root
stalk_surface_bel    105.3537327 stalk_surface_bel
bruises               57.7422613           bruises
stalk_color_bel       54.2937172   stalk_color_bel
ring_number           47.5561875       ring_number
cap_color             46.4503747         cap_color
stalk_color_abo       40.7554903   stalk_color_abo
stalk_shape           30.1603702       stalk_shape
gill_spacing          29.6451075      gill_spacing
cap_surface           21.7426841       cap_surface
cap_shape              6.4251349         cap_shape
veil_color             1.3821077        veil_color
gill_attach            0.9005613       gill_attach
```

NAÏVE BAYES

It requires a small amount of training data to estimate the test data hence training period time is less. It is easier to implement. When the assumption of independent predictors holds true, a Naive Bayes classifier performs better as compared to other models. (Kumar, 2020)

Naive Bayes assumes all the attributes are mutually independent and it is impossible to get completely independent predictors. (Kumar, 2020)

The next step is to train the model on data using the library(e1071) it contains a naive Bayes classifier object that can be used to make predictions: mushroom_classifier <- naive Bayes(type ~ ., train, test,laplace = 20)

```
> ##MODEL EVALUATION - NAIVE BAYES
> mushroom_classifer<-naiveBayes(type ~ ., train, test,laplace = 20)
> mushroom_prediction<-predict(mushroom_classifer,test)
> mushroom_table<-CrossTable(mushroom_prediction,test$predicted.response,prop.chisq = FALSE, prop.t = FALSE,
dnn = c('predicted', 'actual'))


   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|-------------------------|


Total Observations in Table:  2031


             | actual
   predicted |         e |         p | Row Total |
-------------|-----------|-----------|-----------|
           e |      1044 |       129 |      1173 |
             |     0.890 |     0.110 |     0.578 |
             |     0.985 |     0.133 |           |
-------------|-----------|-----------|-----------|
           p |        16 |       842 |       858 |
             |     0.019 |     0.981 |     0.422 |
             |     0.015 |     0.867 |           |
-------------|-----------|-----------|-----------|
Column Total |      1060 |       971 |      2031 |
             |     0.522 |     0.478 |           |
-------------|-----------|-----------|-----------|
```

The table suggests that 129 of 1173 edible mushrooms were incorrectly classified as poisonous, 16 of 858 poison mushrooms were incorrectly classified as edible. Improving model performance, Laplace estimator when allows words that appeared in zero edibility or zero poisonings to have an indisputable say in the classification process. If the categorical variable has a category in the test data set, which was not observed in the training data set, then the model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as Zero Frequency. To solve this, we can use the smoothing technique called Laplace estimation. (Kumar, 2020)

```
> coMa <- confusionMatrix(mushroom_prediction,test$predicted.response)
> acc_nb <- coMa$overall["Accuracy"]
> acc_nb
 Accuracy
0.9286066
> coMa
Confusion Matrix and Statistics

          Reference
Prediction    e    p
         e 1044  129
         p   16  842

               Accuracy : 0.9286
                 95% CI : (0.9165, 0.9394)
    No Information Rate : 0.5219
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8562

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9849
            Specificity : 0.8671
         Pos Pred Value : 0.8900
         Neg Pred Value : 0.9814
             Prevalence : 0.5219
         Detection Rate : 0.5140
   Detection Prevalence : 0.5775
      Balanced Accuracy : 0.9260

       'Positive' Class : e
```

The mushroom_classifier variable builds the classifier while the mushroom_prediction predicts the edibility type. The training dataset was used to model the data to predict whether the mushroom was edible or poisonous. It predicted the response as 129 false positives and 16 false negatives. It made a prediction accuracy of 92.8% with a confidence interval of 95%. Kappa = 0.85 means that the prediction was correct 85.6% of the time. The model's performance is approximately 92.8% effective.

DECISION TREES

The advantage of the decision tree is the intuitive nature and ease of interpreting what will happen in different decision scenarios. It requires little data preparation from the user and there is no need to normalize data ("Uses of Decision Trees in Business Data Mining | Research Optimus", 2015)

It involves a higher time to train the model and a slight data change can cause a large change in the structure of the decision tree causing instability. Decision Tree algorithm is inadequate for applying regression and predicting continuous values. ("Uses of Decision Trees in Business Data Mining | Research Optimus", 2015)

To carry out the decision tree model, the package rpart was installed. To prepare the data for modeling, we split the data into test and train datasets. The training will use 75 percent of the data and 25 percent for testing. The number of samples is 6093. The model evaluation indicates that four mushrooms of the test data were misclassified as poisonous in the edible section. The model indicates it had a 99.8% accuracy at classifying edible and poison mushrooms. It made a prediction accuracy of 99.8% with a confidence interval of 95%. Kappa = 0.996 means that the prediction was correct 99.6% of the time. The model's performance is approximately 99.8% effective.

```
> ##MODEL EVALUATION - DECISION TREE
> ##Train the model
> mod1 <- rpart(type~.,data=train,control=rpart.control(cp=0.005,xval=10))
> print(mod1)
n= 6093

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 6093 2945 e (0.516658461 0.483341539)
  2) odor=a,l,n 3241   93 e (0.971305153 0.028694847)
    4) spore_color=b,h,k,n,o,u,w,y 3188   40 e (0.987452949 0.012547051)
      8) stalk_color_bel=e,g,n,o,p,w 3168   20 e (0.993686869 0.006313131) *
      9) stalk_color_bel=y 20    0 p (0.000000000 1.000000000) *
    5) spore_color=r 53    0 p (0.000000000 1.000000000) *
  3) odor=c,f,m,p,s,y 2852    0 p (0.000000000 1.000000000) *
> mod1.p <- as.party(mod1)
> plot(mod1.p)
> ##model evaluation
> pred <- predict(mod1,test,type = "class")
> table(pred,test$type)

pred    e    p
   e 1060    4
   p    0  967
```

```
pred    e    p
   e 1060    4
   p    0  967
> coMa <- confusionMatrix(pred, reference = test$type)
> coMa
Confusion Matrix and Statistics

          Reference
Prediction    e    p
         e 1060    4
         p    0  967

               Accuracy : 0.998
                 95% CI : (0.995, 0.9995)
    No Information Rate : 0.5219
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9961

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 1.0000
            Specificity : 0.9959
         Pos Pred Value : 0.9962
         Neg Pred Value : 1.0000
             Prevalence : 0.5219
         Detection Rate : 0.5219
   Detection Prevalence : 0.5239
      Balanced Accuracy : 0.9979

       'Positive' Class : e
```

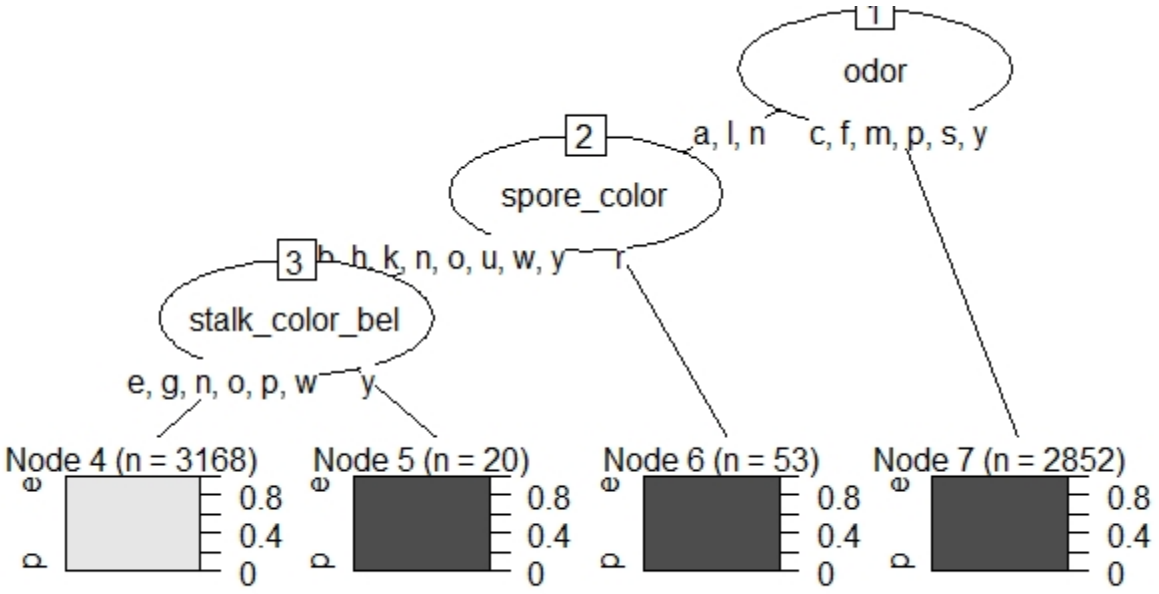
The model evaluation indicates that four mushrooms of the test data were misclassified as poisonous in the edible section. The model indicates it had a 99.8% accuracy at classifying edible and poison mushrooms. It made a prediction accuracy of 99.8% with a confidence interval of 95%. Kappa = 0.996 means that the prediction was correct 99.6% of the time. The model's performance is approximately 99.8% effective.

```
> acc_dt <- coMa$overall["Accuracy"]
> acc_dt
 Accuracy
0.9980305
> confusionMatrix(pred, test$type)
Confusion Matrix and Statistics

          Reference
Prediction    e    p
         e 1060    4
         p    0  967

               Accuracy : 0.998
                 95% CI : (0.995, 0.9995)
    No Information Rate : 0.5219
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9961

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 1.0000
            Specificity : 0.9959
         Pos Pred Value : 0.9962
         Neg Pred Value : 1.0000
             Prevalence : 0.5219
         Detection Rate : 0.5219
   Detection Prevalence : 0.5239
      Balanced Accuracy : 0.9979

       'Positive' Class : e
```

The Cross-Validation is set to 10, also the complexity parameter has been set to 0.005 to better fit the model. The plot below suggests the following highly edible and poisonous mushrooms:

Poisonous: odor - c,f,m,p,s,y; spore color – r; stalk.color - y

Edible: odor - a,l,n ; spore.color - b,k,n,o,u,w; stalk.color - e,g,n,o,p,w



MODEL ACCURACY

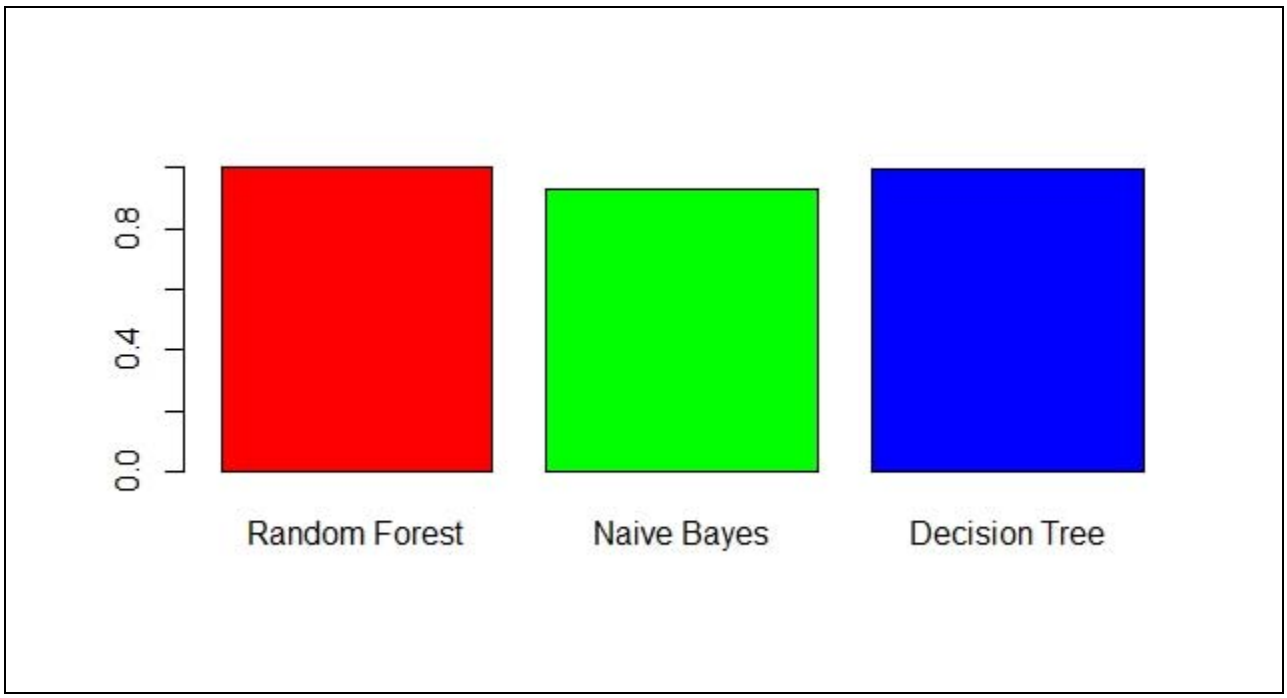
The model accuracy indicates that the performance of Random Forest was 100% effective, Naïve Bayes was 92.8% effective and Decision Tree was 99.8% effective. Naïve Bayes had the lowest percent accuracy while Random Forest was the highest but all the models were greater than 90% to predict whether the mushroom was edible or poisonous with regards to its features.

```
> ##MODEL ACCURACY
> # make a dataframe with each of the accuracy
> df_acc <- data.frame(model=c("Random Forest", "Naive Bayes", "Decision Tree"), Accuracy=c(acc_rt, acc_nb,
 acc_dt))
> df_acc
           model  Accuracy
1 Random Forest 1.0000000
2   Naive Bayes 0.9286066
3 Decision Tree 0.9980305
```

Filter

	model	Accuracy
1	Random Forest	1.0000000
2	Naive Bayes	0.9286066
3	Decision Tree	0.9980305

BARPLOT FOR MODEL ACCURACY



```r
##package installation

library(caret)

## Warning: package 'caret' was built under R version 3.6.3

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.3

library(randomForest)

## Warning: package 'randomForest' was built under R version 3.6.3

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

library(rpart)

## Warning: package 'rpart' was built under R version 3.6.3

library(partykit)

## Warning: package 'partykit' was built under R version 3.6.3

## Loading required package: grid

## Loading required package: libcoin

## Warning: package 'libcoin' was built under R version 3.6.2

## Loading required package: mvtnorm

library(e1071)

## Warning: package 'e1071' was built under R version 3.6.3

library(gmodels)

## Warning: package 'gmodels' was built under R version 3.6.3

##loading data
mushroom <- read.csv('agaricus-lepiota.csv',header=F,sep=',')

##explore data
```

```r
names(mushroom) <- c('type','cap_shape','cap_surface','cap_color','bruises','odor','gill_attach','gill_spacing','gill_size','gill_color','stalk_shape','stalk_root','stalk_surface_abo','stalk_surface_bel','stalk_color_abo','stalk_color_bel','veil_type','veil_color','ring_number','ring_type','spore_color','pop','habitat')
#8124 observations, 23 variables
dim(mushroom)

## [1] 8124   23

str(mushroom)

## 'data.frame':    8124 obs. of  23 variables:
##  $ type             : Factor w/ 2 levels "e","p": 2 1 1 2 1 1 1 1 2 1 ...
##  $ cap_shape        : Factor w/ 6 levels "b","c","f","k",..: 6 6 1 6 6 6 1 1 6 1 ...
##  $ cap_surface      : Factor w/ 4 levels "f","g","s","y": 3 3 3 4 3 4 3 4 4 3 ...
##  $ cap_color        : Factor w/ 10 levels "b","c","e","g",..: 5 10 9 9 4 10 9 9 9 10 ...
##  $ bruises          : Factor w/ 2 levels "f","t": 2 2 2 2 1 2 2 2 2 2 ...
##  $ odor             : Factor w/ 9 levels "a","c","f","l",..: 7 1 4 7 6 1 1 4 7 1 ...
##  $ gill_attach      : Factor w/ 2 levels "a","f": 2 2 2 2 2 2 2 2 2 2 ...
##  $ gill_spacing     : Factor w/ 2 levels "c","w": 1 1 1 1 2 1 1 1 1 1 ...
##  $ gill_size        : Factor w/ 2 levels "b","n": 2 1 1 2 1 1 1 1 2 1 ...
##  $ gill_color       : Factor w/ 12 levels "b","e","g","h",..: 5 5 6 6 5 6 3 6 8 3 ...
##  $ stalk_shape      : Factor w/ 2 levels "e","t": 1 1 1 1 2 1 1 1 1 1 ...
##  $ stalk_root       : Factor w/ 5 levels "?","b","c","e",..: 4 3 3 4 4 3 3 3 4 3 ...
##  $ stalk_surface_abo: Factor w/ 4 levels "f","k","s","y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ stalk_surface_bel: Factor w/ 4 levels "f","k","s","y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ stalk_color_abo  : Factor w/ 9 levels "b","c","e","g",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ stalk_color_bel  : Factor w/ 9 levels "b","c","e","g",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ veil_type        : Factor w/ 1 level "p": 1 1 1 1 1 1 1 1 1 1 ...
##  $ veil_color       : Factor w/ 4 levels "n","o","w","y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ ring_number      : Factor w/ 3 levels "n","o","t": 2 2 2 2 2 2 2 2 2 2 ...
##  $ ring_type        : Factor w/ 5 levels "e","f","l","n",..: 5 5 5 5 1 5 5 5 5 5 ...
##  $ spore_color      : Factor w/ 9 levels "b","h","k","n",..: 3 4 4 3 4 3 3 4 3 3 ...
##  $ pop              : Factor w/ 6 levels "a","c","n","s",..: 4 3 3 4 1 3 3 4 5 4 ...
```

```
##  $ habitat          : Factor w/ 7 levels "d","g","l","m",..: 6 2 4 6 2 2 4
4 2 4 ...
```

```
summary(mushroom)
```

```
##  type     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_attach gill_spacing gill_size   gill_color   stalk_shape stalk_root
##  a: 210      c:6812       b:5612    b      :1728   e:3516      ?:2480
##  f:7914      w:1312       n:2512    p      :1492   t:4608      b:3776
##                                     w      :1202               c: 556
##                                     n      :1048               e:1120
##                                     g      : 752               r: 192
##                                     h      : 732
##                                     (Other):1170
##  stalk_surface_abo stalk_surface_bel stalk_color_abo stalk_color_bel veil_
type
##  f: 552            f: 600            w      :4464    w      :4384    p:812
4
##  k:2372            k:2304            p      :1872    p      :1872
##  s:5176            s:4936            g      : 576    g      : 576
##  y:  24            y: 284            n      : 448    n      : 512
##                                      b      : 432    b      : 432
##                                      o      : 192    o      : 192
##                                      (Other): 140    (Other): 156
##  veil_color ring_number ring_type  spore_color   pop      habitat
##  n:  96     n:  36      e:2776    w      :2388   a: 384   d:3148
##  o:  96     o:7488      f:  48    n      :1968   c: 340   g:2148
##  w:7924     t: 600      l:1296    k      :1872   n: 400   l: 832
##  y:   8                 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
```

```
#It is likely that this variable was
#somehow coded incorrectly. In any case, since veil_type does not vary across
#samples, it does not provide any useful information for prediction.
mushroom$veil_type <- NULL
#eliminates the feature (partial) from the mushrooms data frame
#class  edible = e , poisonous = p
table(mushroom$type)
```

```
##
##    e    p
## 4208 3916
```

```r
round(prop.table(table(mushroom$type))*100)
```

```
##
##  e  p
## 52 48
```

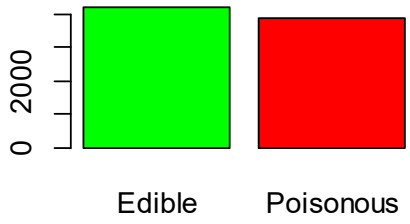
```r
#About 52 percent of the mushroom samples (N = 4,208) are edible, while 48 pe
rcent
#(N = 3,916) are poisonous. As the class levels are split into about 50/50, w
e do not
#need to worry about imbalanced data.


##EDA
nb_value_counts = table(as.numeric(df$type))
```

```
## Error in df$type: object of type 'closure' is not subsettable
```

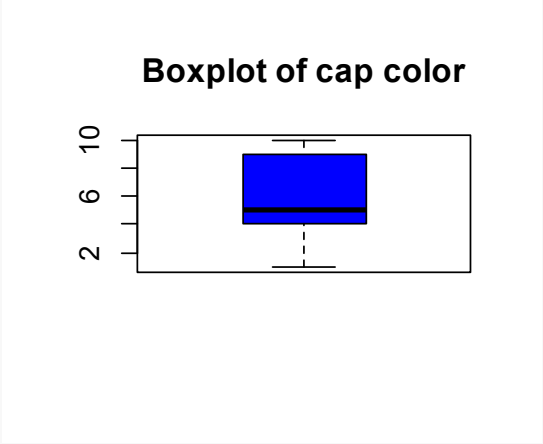
```r
nb_value_counts
```

```
   1    2
4208 3916
```

```r
barplot(nb_value_counts, names.arg=c('Edible', 'Poisonous'), col=c('Green', '
Red'))
```




```r
boxplot(as.numeric(df$cap_color), col="blue", main="Boxplot of cap color")
```




```
summary(as.numeric(df$cap_color))
```

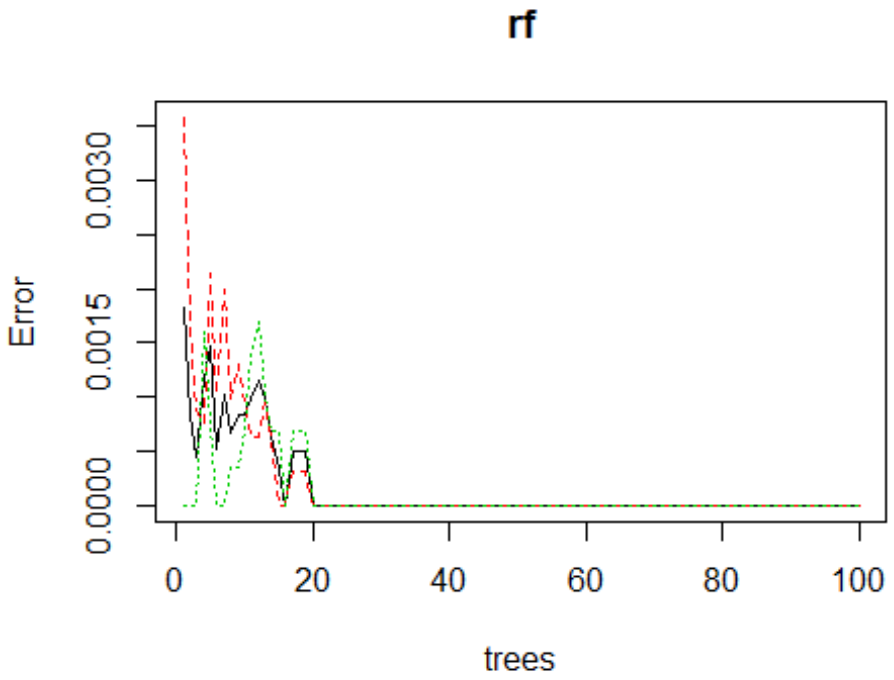
```
> summary(as.numeric(df$cap_color))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   4.000   5.000   5.505   9.000  10.000
```

```
##train and test model
df <- mushroom

smp_size <- floor(0.75 * nrow(df))

## set the seed to make your partition reproducible
set.seed(1)
train_ind <- sample(seq_len(nrow(df)), size = smp_size)

train <- df[train_ind, ]
head(train)

##      type cap_shape cap_surface cap_color bruises odor gill_attach gill_sp
acing
## 1017    e         f           y         n       t    l           f
c
## 8004    e         f           s         n       f    n           a
c
## 4775    p         x           f         y       f    f           f
c
## 2177    e         f           f         n       t    n           f
c
## 5026    p         f           y         g       f    f           f
c
## 1533    p         f           s         w       t    p           f
c
##      gill_size gill_color stalk_shape stalk_root stalk_surface_abo
```

```
## 1017         b           n           e           r                  s
## 8004         b           n           e           ?                  s
## 4775         b           h           e           b                  k
## 2177         b           w           t           b                  s
## 5026         b           g           e           b                  k
## 1533         n           n           e           e                  s
##      stalk_surface_bel stalk_color_abo stalk_color_bel veil_color ring_num
ber
## 1017                 y               w               w          w
o
## 8004                 s               o               o          n
o
## 4775                 k               p               n          w
o
## 2177                 s               g               w          w
o
## 5026                 k               n               p          w
o
## 1533                 s               w               w          w
o
##      ring_type spore_color pop habitat
## 1017         p           n   s       g
## 8004         p           o   v       l
## 4775         l           h   v       d
## 2177         p           k   y       d
## 5026         l           h   y       d
## 1533         p           n   s       u

test <- df[-train_ind, ]
head(test)

##    type cap_shape cap_surface cap_color bruises odor gill_attach gill_spac
ing
## 5     e         x           s         g       f    n           f
w
## 8     e         b           y         w       t    l           f
c
## 10    e         b           s         y       t    a           f
c
## 11    e         x           y         y       t    l           f
c
## 12    e         x           y         y       t    a           f
c
## 13    e         b           s         y       t    a           f
c
##    gill_size gill_color stalk_shape stalk_root stalk_surface_abo
## 5          b          k           t          e                  s
## 8          b          n           e          c                  s
## 10         b          g           e          c                  s
## 11         b          g           e          c                  s
```

```
## 12          b           n           e           c                   s
## 13          b           w           e           c                   s
##    stalk_surface_bel stalk_color_abo stalk_color_bel veil_color ring_numbe
r
## 5                   s               w               w           w
o
## 8                   s               w               w           w
o
## 10                  s               w               w           w
o
## 11                  s               w               w           w
o
## 12                  s               w               w           w
o
## 13                  s               w               w           w
o
##    ring_type spore_color pop habitat
## 5          e           n   a       g
## 8          p           n   s       m
## 10         p           k   s       m
## 11         p           n   n       g
## 12         p           k   s       m
## 13         p           n   s       g
```

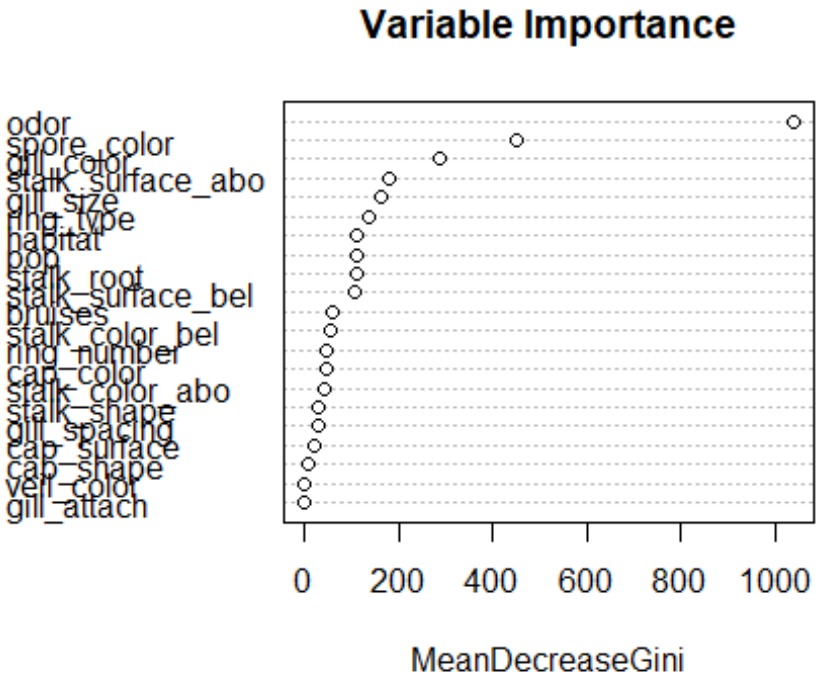
```r
##MODEL EVALUATION - RANDOM FOREST

rf = randomForest(type ~ .,
                  ntree = 100,
                  data=train)
plot(rf)
```




```
print(rf)

##
## Call:
##  randomForest(formula = type ~ ., data = train, ntree = 100)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 0%
## Confusion matrix:
##      e    p class.error
## e 3148    0           0
## p    0 2945           0

# using TEST dataset
# Predicting response variable
test$predicted.response = predict(rf , test)
test$predicted.response <- as.factor(test$predicted.response)
test$predicted.response

##    [1] e e e e e e p e e e e e e e e e e e e e e e e e e e e e e e e e e e e e
## e e e
##   [38] e e e e e e e e e e e e e e e e p e e e e p e e e e e e e e p e e e e e
## e e e
##   [75] e e e e e e e e e e e e e e p e e e e e p e e e e e e e e e e e e e e e
## e e e
```

```
##  [112] e e e e e e e e e p e e p e e e e e e e e e e e e e e e e e e p e e e e
e e e
##  [149] e e e e e e e e e e e e e e e e e p e e e e e e e e e e e e e e e e e e
p e p
##  [186] e e e e e e e e e e e e e e e e e e e e e e e e e e p e e e e e p e p e
e e e
##  [223] p e p p e e e p e e e e e e p e e e e e e e e e e e e e p e e e e e e e
e e e
##  [260] e e e e e e e e e p e e p e e e e e e e e e e e e e e e e e e e e e e e e
e e p
## [1148] p e p p p p p p e p p p p p p p p p p p p p p p p p e p p p p p p p p p
p p p
## [1925] e e p p p p p e p p e e p p e e p e p e e p p p p p e e e e e e p e p
p p e
## [1962] p p p p p p e p e e p e e p e p e p p e e e p e p p e p p p p e p e e
e p e
## [1999] e e e p p e p e p p e p p e p e p e e p e e e p e p p e e e e e p
## Levels: e p

# Create Confusion Matrix
coMa <- confusionMatrix(test$predicted.response,as.factor(test$predicted.resp
onse))
coMa

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    e    p
##          e 1060    0
##          p    0  971
##
##                Accuracy : 1
##                  95% CI : (0.9982, 1)
##     No Information Rate : 0.5219
##     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 : 0.5219
##          Detection Rate : 0.5219
##    Detection Prevalence : 0.5219
##       Balanced Accuracy : 1.0000
##
```

```
##        'Positive' Class : e
##

acc_rt <- coMa$overall["Accuracy"]
acc_rt

## Accuracy
##        1

##important variable in predicting edibility
varImpPlot(rf,
           sort = T,
           main = "Variable Importance")
```




```
##The importance of each attribute according to Mean Decrease Gini is listed
below.
var.imp = data.frame(importance(rf, type=2))

# make row names as columns
var.imp$Variables = row.names(var.imp)
print(var.imp[order(var.imp$MeanDecreaseGini,decreasing = T),])

##                   MeanDecreaseGini         Variables
## odor                 1041.9748530              odor
## spore_color           449.4488773       spore_color
## gill_color            286.6114929        gill_color
## stalk_surface_abo     181.4967408 stalk_surface_abo
## gill_size             162.9707182         gill_size
```

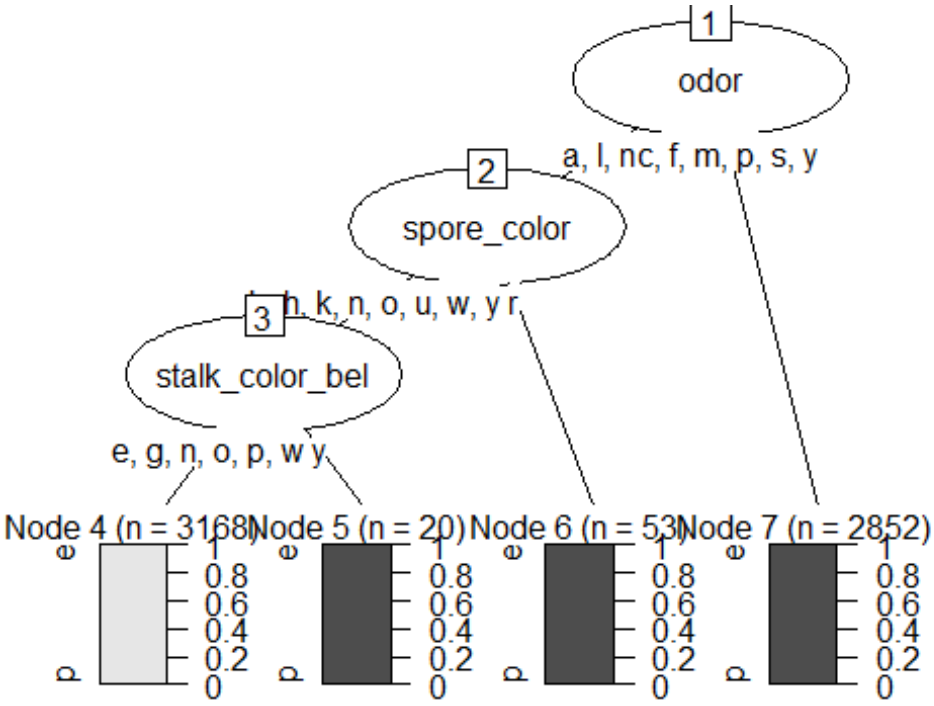
```
## ring_type              135.1907583         ring_type
## habitat                113.2027263           habitat
## pop                    112.2509866               pop
## stalk_root             111.1131890        stalk_root
## stalk_surface_bel      105.3537327 stalk_surface_bel
## bruises                 57.7422613           bruises
## stalk_color_bel         54.2937172   stalk_color_bel
## ring_number             47.5561875       ring_number
## cap_color               46.4503747         cap_color
## stalk_color_abo         40.7554903   stalk_color_abo
## stalk_shape             30.1603702       stalk_shape
## gill_spacing            29.6451075      gill_spacing
## cap_surface             21.7426841       cap_surface
## cap_shape                6.4251349         cap_shape
## veil_color               1.3821077        veil_color
## gill_attach              0.9005613       gill_attach
```

```
##MODEL EVALUATION - NAIVE BAYES
mushroom_classifer<-naiveBayes(type ~ ., train, test,laplace = 20)
mushroom_prediction<-predict(mushroom_classifer,test)
mushroom_table<-CrossTable(mushroom_prediction,test$predicted.response,prop.c
hisq = FALSE, prop.t = FALSE,dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |-------------------------|
##
##
## Total Observations in Table:  2031
##
##
##              | actual
##    predicted |         e |         p | Row Total |
## -------------|-----------|-----------|-----------|
##            e |      1044 |       129 |      1173 |
##              |     0.890 |     0.110 |     0.578 |
##              |     0.985 |     0.133 |           |
## -------------|-----------|-----------|-----------|
##            p |        16 |       842 |       858 |
##              |     0.019 |     0.981 |     0.422 |
##              |     0.015 |     0.867 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1060 |       971 |      2031 |
##              |     0.522 |     0.478 |           |
## -------------|-----------|-----------|-----------|
```

```
##
##

coMa <- confusionMatrix(mushroom_prediction,test$predicted.response)
coMa

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    e    p
##          e 1044  129
##          p   16  842
##
##                Accuracy : 0.9286
##                  95% CI : (0.9165, 0.9394)
##     No Information Rate : 0.5219
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8562
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9849
##             Specificity : 0.8671
##          Pos Pred Value : 0.8900
##          Neg Pred Value : 0.9814
##              Prevalence : 0.5219
##          Detection Rate : 0.5140
##    Detection Prevalence : 0.5775
##       Balanced Accuracy : 0.9260
##
##        'Positive' Class : e
##

acc_nb <- coMa$overall["Accuracy"]
acc_nb

##  Accuracy
## 0.9286066

##MODEL EVALUATION - DECISION TREE
##Train the model
mod1 <- rpart(type~.,data=train,control=rpart.control(cp=0.005,xval=10))
print(mod1)

## n= 6093
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 6093 2945 e (0.516658461 0.483341539)
```

```
##   2) odor=a,l,n 3241   93 e (0.971305153 0.028694847)
##     4) spore_color=b,h,k,n,o,u,w,y 3188   40 e (0.987452949 0.012547051)
##       8) stalk_color_bel=e,g,n,o,p,w 3168   20 e (0.993686869 0.006313131)
*
##       9) stalk_color_bel=y 20    0 p (0.000000000 1.000000000) *
##     5) spore_color=r 53    0 p (0.000000000 1.000000000) *
##   3) odor=c,f,m,p,s,y 2852    0 p (0.000000000 1.000000000) *

mod1.p <- as.party(mod1)
plot(mod1.p)
```



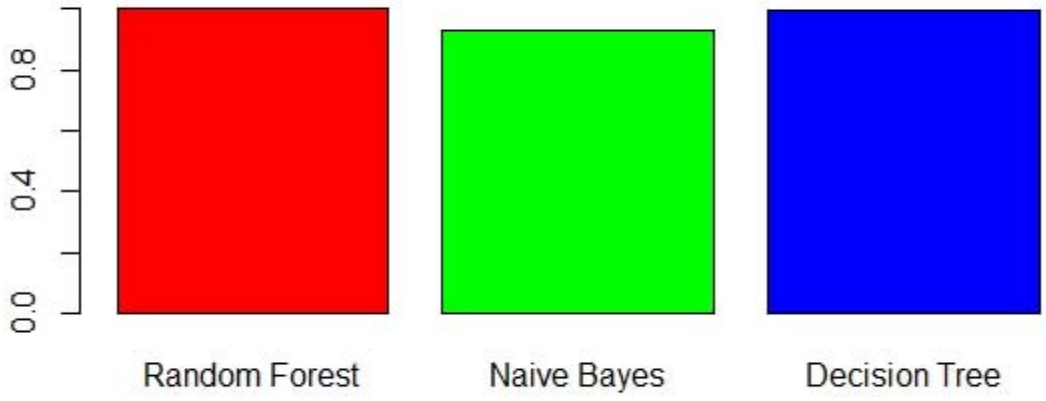

```
##model evaluation
pred <- predict(mod1,test,type = "class")
table(pred,test$type)

##
## pred    e    p
##    e 1060    4
##    p    0  967

coMa <- confusionMatrix(pred, reference = test$type)
acc_dt <- coMa$overall["Accuracy"]
acc_dt

##  Accuracy
## 0.9980305

confusionMatrix(pred, test$type)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    e    p
##          e 1060    4
##          p    0  967
##
##                Accuracy : 0.998
##                  95% CI : (0.995, 0.9995)
##     No Information Rate : 0.5219
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9961
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 1.0000
##             Specificity : 0.9959
##          Pos Pred Value : 0.9962
##          Neg Pred Value : 1.0000
##              Prevalence : 0.5219
##          Detection Rate : 0.5219
##    Detection Prevalence : 0.5239
##       Balanced Accuracy : 0.9979
##
##        'Positive' Class : e
##
```

```
##MODEL ACCURACY
# make a dataframe with each of the accuracy
df_acc <- data.frame(model=c("Random Forest", "Naive Bayes", "Decision Tree")
, Accuracy=c(acc_rt, acc_nb, acc_dt))
df_acc
```

```
##           model  Accuracy
## 1   Random Forest 1.0000000
## 2   Naive Bayes 0.9286066
## 3 Decision Tree 0.9980305
```

```
barplot(df_acc$Accuracy, names.arg=df_acc$model, col= c("red", "green", "blue
"))
```


0.0
0.4
0.8
Random Forest
Naive Bayes
Decision Tree

Conclusion

In conclusion, Random Forest is the best model for classification prediction to predict the mushroom edibility since the model was 100% accurate. The Random forest is a classification algorithm that consists of numerous decision trees. It uses bagging and feature randomness when building each tree, it uses bagging and randomness and creates a forest of trees that are not correlated whose prediction accurate than that of an individual tree. It does not overfit and can run as many trees as one wants.

Naïve Bayes is called "naïve" because it makes the impression that the occurrence of a certain attribute is independent of the occurrence of other attributes. In simpler terms, it assumes that in a class, the presence of a particular feature is not linked to the other features present. The predictors or features are independent of each other.

Decision Tree algorithm is supervised learning algorithms and solve regression and classification problems. They are a tree graph model with the terminal nodes representing classification outcomes and decisions. The decision tree breaks down a data set into smaller and smaller subsets. The leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called the root node.

The mushroom dataset, the goal was to build a classifier that predicts species that is edible or poisonous. Variables such as odor, spore color, and gill color were the common variables that were used to predict the species edibility and determine the poisonous mushrooms. It made a prediction accuracy of 100% with a confidence interval of 95%, Kappa = 1 means that the prediction was correct 100% of the time for Random Forest. It made a prediction accuracy of 92.8% with a confidence interval of 95%, Kappa = 0.85 means that the prediction was correct 85% of the time for Naïve Bayes. It made a prediction accuracy of 99.8% with a confidence interval of 95%, Kappa = 0.996 means that the prediction was correct 99.6% of the time for Decision Trees.

There was a high number of mushrooms classified incorrectly using a Naive Bayes model. The decision tree had four mushrooms classified erroneously while random tree had zero misclassified.

References

Lantz, B. (2013). Machine Learning with R. Retrieved 10 May 2020, from https://www.amazon.com/Machine-Learning-R-Brett-Lantz/dp/1782162143

Levine, D., & Stephan, D. (2013). Statistics for Managers (6th ed., pp. 105). New Jersey: Pearson.

Ray, S. (2017). Learn the Naive Bayes Algorithm | Naive Bayes Classifier Examples. Retrieved 12 May 2020, from https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

Strickland, J. (2014). Predictive Analytics Using R (1st ed., pp. 119-121). Colorado Springs.

Science, S. (2020). Decision Trees: Athlete Care in Big Data — Sparta Science - Movement Vital Sign. Retrieved 14 May 2020, from https://spartascience.com/resources/decision-trees-athlete-care-in-big-data

Uses of Decision Trees in Business Data Mining | Research Optimus. (2015). Retrieved 14 May 2020, from https://www.researchoptimus.com/blog/uses-of-decision-trees-in-business-data-mining/

Sehra, C. (2018). Decision Trees Explained Easily. Retrieved 14 May 2020, from https://medium.com/@chiragsehra42/decision-trees-explained-easily-28f23241248

Kumar, N. (2020). Advantages and Disadvantages of the Random Forest Algorithm in Machine Learning. Retrieved 14 May 2020, from http://theprofessionalspoint.blogspot.com/2019/02/advantages-and-disadvantages-of-random.html

Waldron, M. (2015). Naive Bayes for Dummies; A Simple Explanation. Retrieved 15 May 2020, from https://www.datasciencecentral.com/profiles/blogs/naive-bayes-for-dummies-a-simple-explanation

Barnwal, M. (2019). Random forests® explained intuitively - KDnuggets. Retrieved 14 May 2020, from https://www.kdnuggets.com/2019/01/random-forests-explained-intuitively.html