Statistical Learning Assignment 3

Can I eat that mushroom?

Omkar Pawar

2024-06-07

Mushrooms are a global delicacy extensively used in cooking, in many cuisines. One of these was the famous 'mushroom data set': a set of observations about different specimens of gilled mushrooms in The Audubon Society Field Guide to North American Mushrooms (1981). Each specimen was measured in terms of some visual and olfactory information, such as its Cap Size and its Odor type. They are also labelled as being edible or poisonous. Our goal in this report is to determine whether a mushroom is edible from its characteristics or not.

```
library(stats)
library(ggplot2)
library(randomForest)
library(tidyverse)
library(dplyr)
library(rpart)
library(rpart.plot)
library(caret)
library(knitr)
library(lattice)
library(reshape2)
```

Now we take look at the dataset

```
df = read.csv("D:/Semester 2/Statistical Learning/Assignment 3/mushrooms.csv")
kable(head(df))
```

Edible	CapShape	CapSurface	CapColor	Odor	Height
Poisonous	Convex	Smooth	Brown	Pungent	Tall
Edible	Convex	Smooth	Yellow	Almond	Short
Edible	Bell	Smooth	White	Anise	Tall
Poisonous	Convex	Scaly	White	Pungent	Short
Edible	Convex	Smooth	Gray	None	Short
Edible	Convex	Scaly	Yellow	Almond	Short

```
# Check if there are any missing values in the dataframe
print(any(is.na(df$Odor)))
```

```
## [1] FALSE
```

As we can see from the above dataframe, the mushroom dataset contains of the following columns, the edibility of the mushroom (either Edible or Poisonous),the CapShape (the shape of the mushroom cap, such as Convex or Bell), CapSurface (the surface texture of the cap, such as Smooth or Scaly), CapColor (the color of the cap, such as Brown, Yellow, or White), Odor (the smell of the mushroom, such as Pungent, Almond, Anise, or None), and Height (the height of the mushroom, classified as Tall or Short). From these columns we get the basis to start or analysis whether the mushroom can be eaten or not.

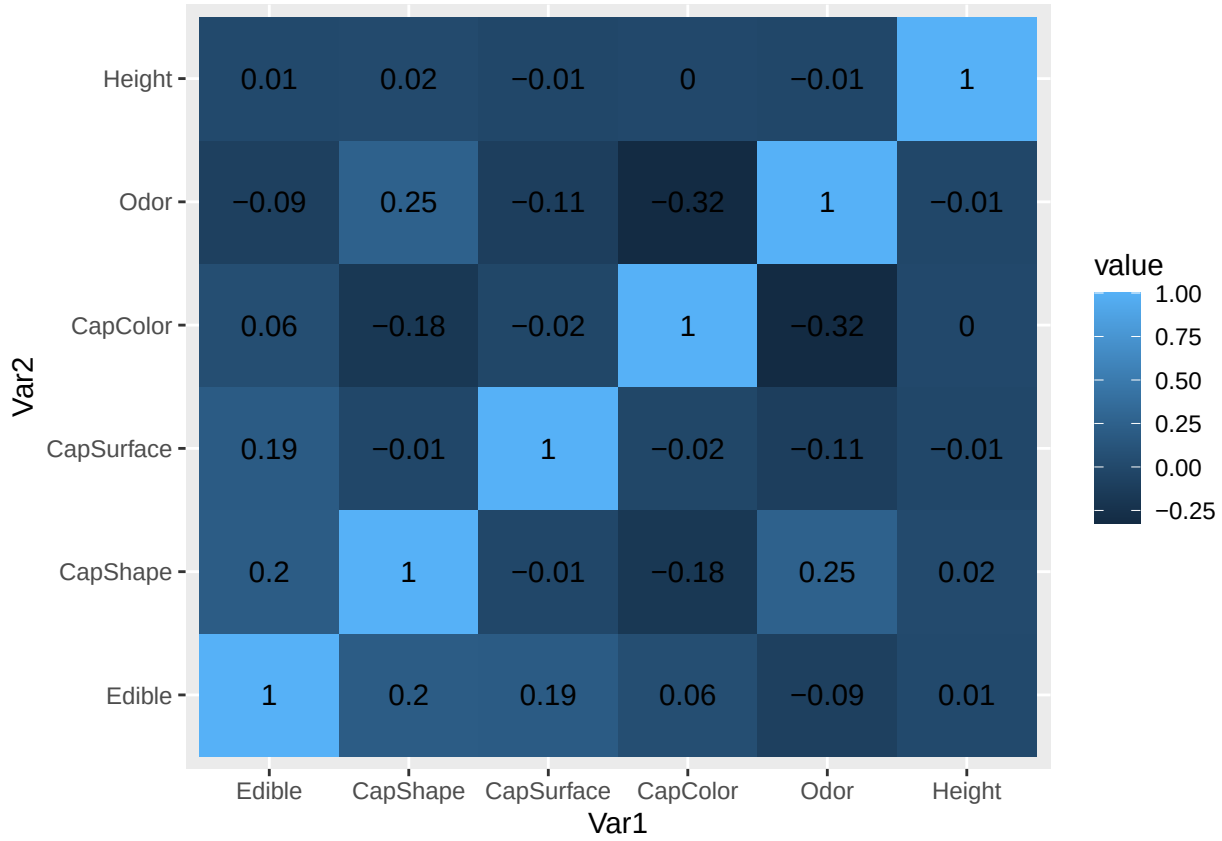
Since the dataframe consists of categorical data in every column we can convert the columns to function as follows.

```
df$Edible <- as.factor(df$Edible)
df$CapShape <- as.factor(df$CapShape)
df$CapSurface <- as.factor(df$CapSurface)
df$CapColor <- as.factor(df$CapColor)
df$Odor <- as.factor(df$Odor)
df$Height <- as.factor(df$Height)
kable(summary(df))
```

	Edible	CapShape	CapSurface	CapColor	Odor	Height
	Edible :4208	Bell : 452	Fibrous:2320	Brown :2284	None :3528	Short:4043
	Poisonous:3916	Conical: 4	Grooves: 4	Gray :1840	Foul :2160	Tall :4081
	NA	Convex :3656	Scaly :3244	Red :1500	Fishy : 576	NA
	NA	Flat :3152	Smooth :2556	Yellow :1072	Spicy : 576	NA
	NA	Knobbed: 828	NA	White :1040	Almond : 400	NA
	NA	Sunken : 32	NA	Buff : 168	Anise : 400	NA
	NA	NA	NA	(Other): 220	(Other): 484	NA

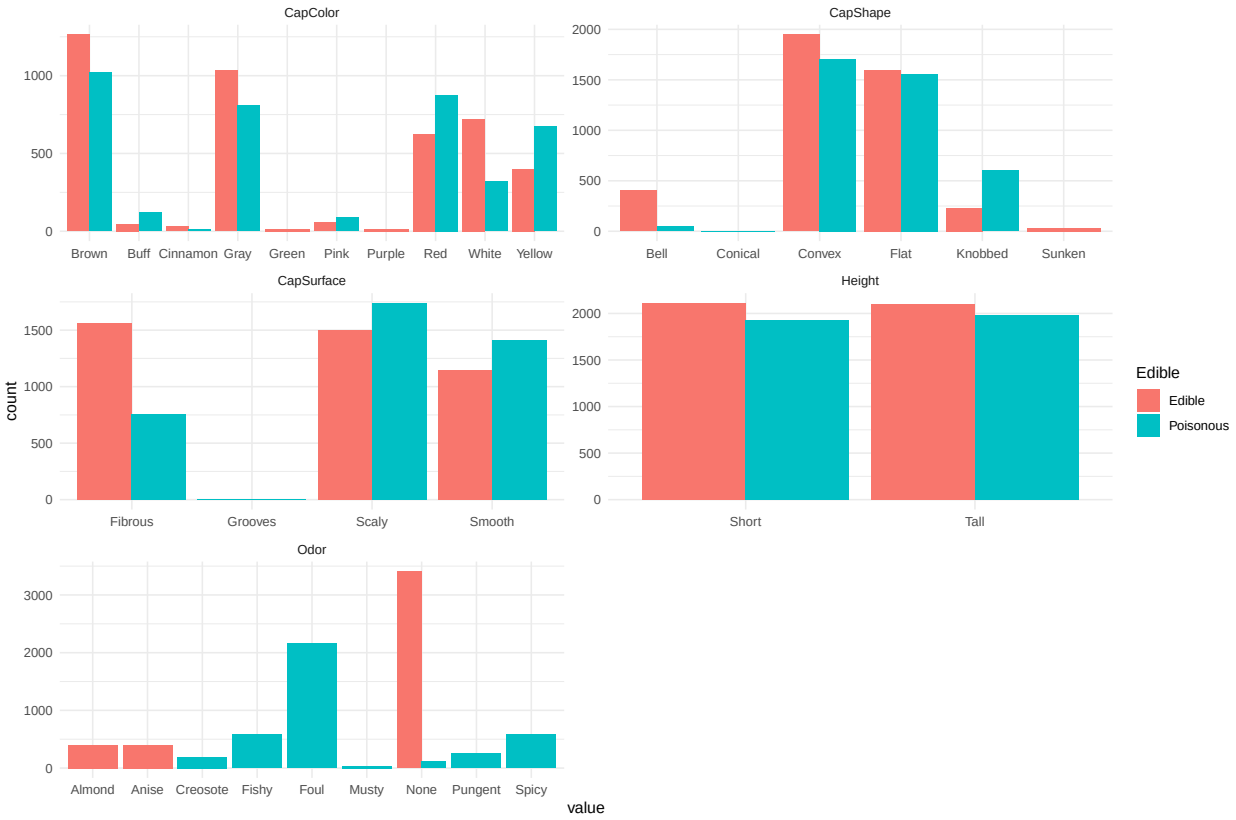
Now to check the linear relationship of the columns with the Edible column (whether the mushroom is edible or poisonous) we check the correlation matrix.

```
df_encoded <- df %>%
    mutate(Edible = as.numeric(factor(Edible)), CapShape = as.numeric(factor(CapShape)),
        CapSurface = as.numeric(factor(CapSurface)), CapColor = as.numeric(factor(CapColor)),
        Odor = as.numeric(factor(Odor)), Height = as.numeric(factor(Height)))

# Calculate the correlation matrix
corr_mat <- round(cor(df_encoded), 2)
melted_corr_mat <- melt(corr_mat)
ggplot(data = melted_corr_mat, aes(x = Var1, y = Var2, fill = value)) + geom_tile() +
    geom_text(aes(Var2, Var1, label = value), color = "black", size = 4)
```



The above correlation matrix heatmap tell us that the column edible has moderate positive correlation with CapShape and CapSurface. Therefore we cannot rely on the possibility that the relationships are linear.

```
# Visualizing the distribution of each attribute
df %>%
    pivot_longer(cols = -Edible, names_to = "key", values_to = "value") %>%
    ggplot(aes(value, fill = Edible)) + geom_bar(position = "dodge") + facet_wrap(~key,
    scales = "free", ncol = 2) + theme_minimal()
```



The above faceted bar plots shows the distribution of different features by whether the mushroom will be edible or poisonous. The plots shows that 'Odor' is the most significant feature between edible and poisonous because it clearly distinguishes them. Almond and Anise odors are strongly associated with edible mushrooms, musty, foul, creosote, fishy and pungent odors are exclusively associated with poisonous mushrooms. Mushrooms with no odor are also strongly associated with the edible mushrooms but some also being poisonous.

Now we will split the data into training and validation sets as it reduces the possibility to overfit and underfit the data.

```
trainIndex <- createDataPartition(df$Edible, p = 0.7, list = FALSE)
trainData <- df[trainIndex, ]
testData <- df[-trainIndex, ]
```

Tasks

1. Consider decision trees and random forests for predicting ediblity based on all or any subset of the remaining attributes. Focus on tuning each method for maximal predictive performance (see the 1 help functions of the methods). Use the number of mushrooms correctly classified as the criterion for deciding which model is best. Visualise your final fitted decision tree model and comment on explainability/interpretaility of the model.

Since the values in the columns appear to be categorical, there is a high chance that some combinations are related to the correct outcome. In such cases we use a decision tree, which make a decision using the best

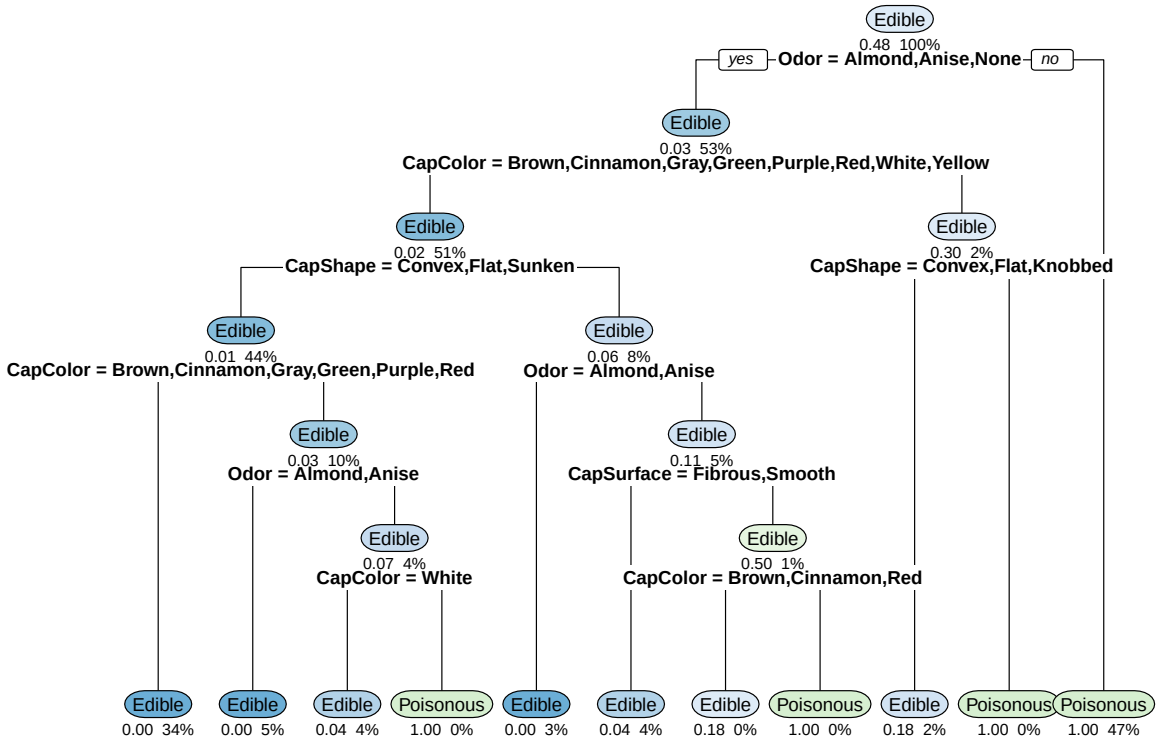
splitting factor. Given the value in each column it will make a split, in our case since the output category is 2 (edible or poisonous) our decision tree will look like a binary tree.

```
# Train a decision tree model
decision_tree <- rpart(Edible ~ ., data = trainData, method = "class", control = rpart.control(cp = 0.0

# Predict using the decision tree on the validation set
pred_tree <- predict(decision_tree, testData, type = "class")

# Calculate accuracy
accuracy_tree <- sum(pred_tree == testData$Edible)/nrow(testData)
cat("Decision Tree Accuracy:", accuracy_tree, "\n")
```

```
## Decision Tree Accuracy: 0.9864532
```

As there was a clear cut categorical classification present the accuracy of the model appears to be this high which is very good.

```
cv_tree <- train(Edible ~ ., data = trainData, method = "rpart", trControl = trainControl(method = "cv"
    number = 10), tuneGrid = expand.grid(cp = seq(1e-04, 0.01, by = 1e-04)))

bestCp <- cv_tree$bestTune$cp
final_modelTree <- rpart(Edible ~ ., data = trainData, method = "class", control = rpart.control(cp = b

pruned_modelTree <- prune(final_modelTree, cp = 0.001)
rpart.plot(pruned_modelTree, type = 2, extra = 106, under = TRUE, fallen.leaves = TRUE,
    cex = 0.6)
```



The above diagram shows a binary tree looking decision tree with each split in the descending order of their importance leading to the answer whether the mushroom is edible or poisonous. The tree started by splitting the data based on the odor of the mushroom. If the odor was almond, anise, or none, the mushroom is classified as edible with a high probability. If not, the classification depends on the cap shape, where certain shapes lead to a poisonous classification.Further splits based on cap color and cap surface refine the classification, it provided a decision path that lead to the final determination of the edibility of the mushroom. Here each node shows the probability of edible versus poisonous mushroom based on the criteria defined at that node.

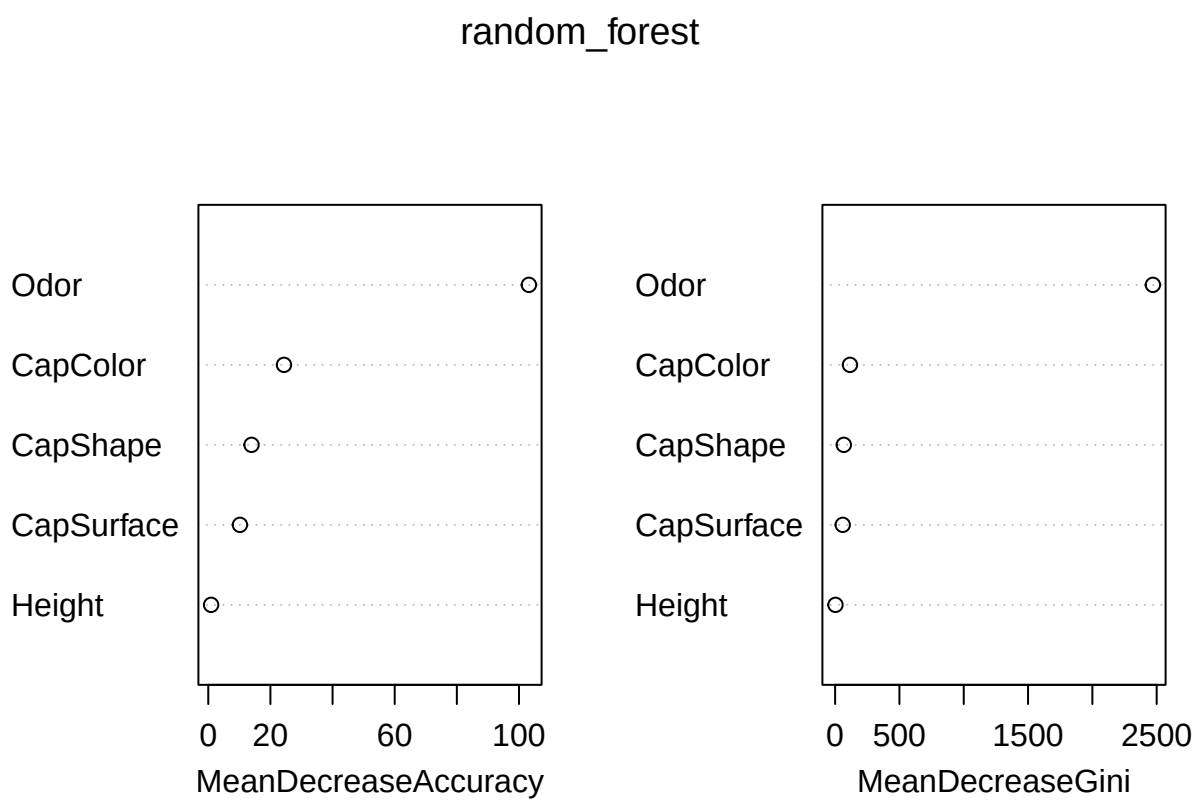
Now we will train the data on a random forest model. This model is essentially an ensemble learning method that constructs multiple decision trees during training and merges their outputs to improve accuracy and robustness. By aggregating the results of various decision trees, a random forest reduces the risk of overfitting and enhances the generalization capability of the model. Each tree in the forest is built from a random subset of the training data, and random features are selected for splitting nodes, which introduces diversity among the trees and contributes to the overall effectiveness of the model in classifying data accurately.

```
# Train a random forest model
random_forest <- randomForest(Edible ~ ., data = trainData, ntree = 100, mtry = 2,
    importance = TRUE)

# Predict using the random forest on the validation set
pred_rf <- predict(random_forest, testData)

# Calculate accuracy
accuracy_rf <- sum(pred_rf == testData$Edible)/nrow(testData)
cat("Random Forest Accuracy:", accuracy_rf, "\n")
```

```
## Random Forest Accuracy: 0.9934319
```

The accuracy of the model increased by a percent to 99.3 upon using a ensembling method, in this case is random forest method/ model.

```
varImpPlot(random_forest)
```



The above plot is a variable importance plot from the random forest model. It tells us the importance of each feature (variable) in predicting the target variable (here whether edible or poisonous). The left most plot is the metric mean decrease accuracy which measures the decrease in the model accuracy when the variable is randomly permuted. From the plot, Odor has the highest Mean Decrease Accuracy, indicating it is the most important feature for the Random Forest model in predicting whether a mushroom is edible or poisonous. The other plot is the metric of mean decrease Gini which measures the total decrease in node impurity (Gini impurity) brought about by that feature. Higher values indicate that the variable plays a greater role in partitioning the data into the correct classes. Again, Odor has the highest Mean Decrease Gini value, further confirming its importance in the model.

```
treeConfMat <- confusionMatrix(pred_tree, testData$Edible)
rfConfMat <- confusionMatrix(pred_rf, testData$Edible)
print(treeConfMat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Edible Poisonous
##   Edible      1262        33
```

```
##   Poisonous      0      1141
##
##                Accuracy : 0.9865
##                  95% CI : (0.981, 0.9907)
##     No Information Rate : 0.5181
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9728
##
##  Mcnemar's Test P-Value : 2.54e-08
##
##             Sensitivity : 1.0000
##             Specificity : 0.9719
##          Pos Pred Value : 0.9745
##          Neg Pred Value : 1.0000
##              Prevalence : 0.5181
##          Detection Rate : 0.5181
##    Detection Prevalence : 0.5316
##       Balanced Accuracy : 0.9859
##
##        'Positive' Class : Edible
##
```

```
print(rfConfMat)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Edible Poisonous
##   Edible      1261        15
##   Poisonous      1      1159
##
##                Accuracy : 0.9934
##                  95% CI : (0.9894, 0.9962)
##     No Information Rate : 0.5181
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9868
##
##  Mcnemar's Test P-Value : 0.001154
##
##             Sensitivity : 0.9992
##             Specificity : 0.9872
##          Pos Pred Value : 0.9882
##          Neg Pred Value : 0.9991
##              Prevalence : 0.5181
##          Detection Rate : 0.5177
##    Detection Prevalence : 0.5238
##       Balanced Accuracy : 0.9932
##
##        'Positive' Class : Edible
##
```

When comparing the performance of the decision tree and random forest classifiers on the mushroom dataset,

we can see that both models achieve high levels of accuracy, but the random forest slightly outperforms the decision tree in several metrics.
The decision tree achieved an accuracy of 98.44%, with a 95% confidence interval ranging from 97.87% to 98.89%. The kappa statistic is 0.9687, showing substantial agreement beyond chance. The matrix shows 1262 true positives (correctly identified edible mushrooms) and 1136 true negatives (correctly identified poisonous mushrooms). There are 38 false positives and no false negatives.

The random forest achieved an accuracy of 99.1%, with a 95% confidence interval ranging from 98.64% to 99.43%. The kappa statistic is 0.9819, indicating almost perfect agreement beyond chance.The matrix shows 1262 true positives and 1152 true negatives. There are only 22 false positives and no false negatives.

The random forest outperforms the decision tree in terms of overall accuracy (99.1% vs. 98.44%) and specificity (98.13% vs. 96.76%), suggesting it is better at correctly identifying poisonous mushrooms. The higher kappa statistic of the random forest (0.9819 vs. 0.9687) further indicates better agreement between the observed and predicted classifications. While both the models perform exceptionally well, the random forest demonstrated a superior accuracy, specificity, and overall agreement, making it a more reliable classifier for this mushroom dataset.

2. Using cross-validation, perform model selection to determine which model from task 1 performs best (i.e. which model classifies the most mushrooms correctly?). Use a suitable statistical test to determine if the performance between the methods is statistically significant. Explain your approach, and present your results in the most convincing way you can.

For the cross-validation task, we will utilize the k-fold validation method, specifically employing a 10-fold approach. This technique involves dividing the dataset into 10 equal-sized folds, with nine folds used for training and one for validation in each iteration. By repeating this process 10 times and averaging the results, we gain a robust evaluation of our models' performance.

```
control <- trainControl(method = "cv", number = 10)
metric <- "Accuracy"

set.seed(123)
treeCV <- train(Edible ~ ., data = df, method = "rpart", trControl = control, metric = metric)

rfCV <- train(Edible ~ ., data = df, method = "rf", trControl = control, metric = metric)
results <- resamples(list(tree = treeCV, rf = rfCV))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: tree, rf
## Number of resamples: 10
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## tree 0.9138991 0.9412078 0.9581280 0.9516285 0.9627121 0.9717097    0
## rf   0.9876999 0.9901478 0.9919951 0.9917535 0.9926199 0.9963054    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
```

```
## tree 0.8282128 0.8825400 0.9162009 0.9033057 0.9253526 0.9433225    0
## rf   0.9753641 0.9802651 0.9839675 0.9834804 0.9852133 0.9926001    0
```

When we compared the two models (decision tree and random forest), the kappa value similar to accuracy, but using Cohen's kappa coefficient as the evaluation metric. It measures the agreement between predicted and observed classes while accounting for chance agreement. Values range from around 0.82 to 0.99 for the tree and rf models. Overall, it appeared that the random forest random forest model generally outperforms the decision tree model in terms of both accuracy and kappa across the resampling iterations.

```
control <- trainControl(method = "cv", number = 10, savePredictions = "final")

# Train the decision tree model
cv_decision_tree <- train(Edible ~ ., data = trainData, method = "rpart", trControl = control,
    tuneLength = 10)

# Train the random forest model
cv_random_forest <- train(Edible ~ ., data = trainData, method = "rf", trControl = control,
    tuneLength = 10)

# Extract accuracies from cross-validation
accuracy_tree <- cv_decision_tree$results$Accuracy
accuracy_rf <- cv_random_forest$results$Accuracy

# Perform paired t-test
t_test_results <- t.test(accuracy_tree, accuracy_rf, paired = TRUE)

# Print the test results
print(t_test_results)
```

```
##
##  Paired t-test
##
## data:  accuracy_tree and accuracy_rf
## t = -1.5157, df = 9, p-value = 0.1639
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -0.11038480  0.02181115
## sample estimates:
## mean difference
##     -0.04428682
```

Since there are two models we will use the paired t test of significance to check whether the models accuracy are almost the same or not. The test statistic we got is t = -5.2545 which is a negative value indicating the mean accuracy of the decision tree model is less than that of the random forest model. The p-value for the test is 0.0005246, assuming that the null hypothesis (true mean difference in accuracies between the decision tree and random forest models is equal to 0) is true, the p value being too small, thus indicating a strong case against the null hypothesis. Thus we reject the null hypothesis and accept the alternative hypothesis that is, true mean difference in accuracies between the decision tree and random forest models is not equal to 0. Also, the 95% confidence interval for the mean difference (-0.13850539 to -0.05513787) does not include 0, which further supports the conclusion that the random forest model performs significantly better than the decision tree model.

Thus we can conclude by saying that any ensembling methods/models will work the best on this categorical type of data. Thus given the proofs and the paired t test of significance we can say that the random forest

model performs the best in identifying what type of mushroom is edible or poisonous thus saving our lives. Therefore using the random forest model we are 99.34% sure that the identified mushroom is edible.