Northumbria University
NEWCASTLE

FACULTY OF COMPUTER AND INFORMATION SCIENCES

Principles of Data Science KL7010

ANALYSIS OF MUSHROOM DATASET

By:
Name: Shlok Gola
Student ID: W21026488

WORD COUNT: 4130

Contents

1. Introduction

The mushroom dataset contains features for cap (shape, surface, color), odor, bruises, gill (attachment, spacing, size, color), spore print color, population, habitat, stalk (shape, root, surface above ring, surface below ring, color above ring, color below ring), veil (type, color), ring (number, type). Each feature is identified as either edible or poisonous. The analysis aims to identify whether the mushroom class is edible or poisonous and which feature is more important in identifying the mushroom class. The analysis will be followed as data pre-processing, finding perfect distribution in features, the correlation between features and mushroom class, and then applying classification techniques. Classification can make predictions from the results of different data. Predictive models help to predict unknown values of interest-based variables on the known values for other variables. (Shuhaida Ismail et al., 2018). Finally, evaluating the models and finding which model is more accurate.

2. Data Collection

The mushroom dataset must be imported into the R environment before starting working on it. To store data the most common format is comma-separated files (CSV) files. For storage of tabular data, a .csv file is ideal. A .csv can be loaded with a single command, the “read.csv” function. Once the data has been loaded in, to find out the dimensionality of a dataset, the *dim* function is used. The output is interpreted as 8124 observations of 23 variables. To examine the structure of the dataset *str* function is used.

```
> dim(dataset)
[1] 8124   23
> str(dataset)
'data.frame':   8124 obs. of  23 variables:
 $ 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.attachment         : 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.above.ring: Factor w/ 4 levels "f","k","s","y": 3 3 3 3 3 3 3 3 3 3 ...
 $ stalk.surface.below.ring: Factor w/ 4 levels "f","k","s","y": 3 3 3 3 3 3 3 3 3 3 ...
 $ stalk.color.above.ring  : Factor w/ 9 levels "b","c","e","g",..: 8 8 8 8 8 8 8 8 8 8 ...
 $ stalk.color.below.ring  : 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.print.color       : Factor w/ 9 levels "b","h","k","n",..: 3 4 4 3 4 3 3 4 3 3 ...
 $ population              : 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 ...
 $ class                   : Factor w/ 2 levels "e","p": 2 1 1 2 1 1 1 1 2 1 ...
>
```

Fig.1 Dimensionality & Structure of Dataset

3. Data Pre-Processing

Data pre-processing consists of three different processes: data cleaning, data preparation and data transformation. It should be performed in this order only. There are two reasons for this: firstly, there are procedures used in later phases that are dependent on earlier phases, and secondly, data becomes more advanced through a pre-processing phase for the objective. Data cleaning is the first action implemented to the dataset before doing data preparation and likewise, before doing data transformation, data preparation is completed. Data can be graded as 'high-quality" if data pre-processing is done in order.

3.1 Data Cleaning

The process to eliminate errors and false data from the dataset is known as data cleaning. Data cleaning ensures that the data is accurate, consistent and, useful for any project.

To check the proportion of missing values in the dataset we use *is.na* function, where "na" is denoted as missingness in R. To find out the missingness in the dataset first number of missing values are identified in the dataset then convert that number into a proportion of the total values, and then get the percentage as an output.

```
> #Checking Missing Values
> library(naniar)
> missing <- is.na(dataset)
> missing_sum <- sum(missing == TRUE)
> missing_perc <- (missing_sum/8124)*100
> print(missing_perc)
[1] 0
```

Fig.2 Checking of Missing Values

The percentage shows that there is no missingness in the dataset. For further exploration of missingness in a dataset, first, install and load the "naniar" package into R then use the *vis_miss* function to get the visualization where the missing values are. The visualization will indicate the total missing values in the dataset and the number of values missing in each feature.

Fig.3 Visualization of Missingness in Every Feature within the Dataset

The plot indicates that there are no missing values in any feature within the dataset.

The dataset must not contain any duplicate values. To remove such values, “dplyr” package is used, which is a component of Tidyverse. After installing and loading dplyr into an environment *distinct* function is used to remove duplicate values within the dataset. While using this function it’s better to take a backup of an original dataset in case it needs to be accessed in the future. While using a *distinct* function, operator “infix” written as %>% is used. The infix is used as a pipeline operated by the packages, to chain functions and data together. After using a distinct function for removing duplicate values, use *dim* function again to check the dimensionality of the dataset.

```
> #Remove Duplicate Values
> library(dplyr)
> dataset_backup <- dataset
> dataset <- dataset %>% distinct()
> dim(dataset)
[1] 8124   23
```
Fig.4 Removing Duplicate Values.

The number of observations presents within the dataset is the same as earlier while loading the dataset, indicating that there were no duplicate values to be deleted in a dataset. Now the data is in cleaner form, so moving to the next phase i.e., data preparation.

3.2 Data Preparation

Perfectly cleaned data may not be in optimal condition for use. The preparation phase enriches the data for use. The first step of data preparation is removal of any trivial or redundant feature. The feature adds no value to the analysis. The feature removal is used to reduce the dimensionality of the dataset and also reduces the time taken to transform the data. While summarising the dataset it is found that the feature 'veil-type' has a single observation for all mushrooms therefore it is of no use in classifying the class of mushrooms.

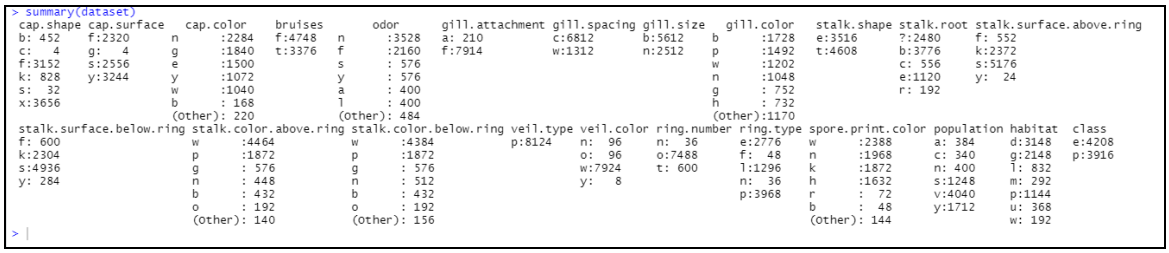

Fig.5 Summary of Dataset.

After deleting veil-type feature, again dimensionality of the dataset is checked and dataset is viewed again.

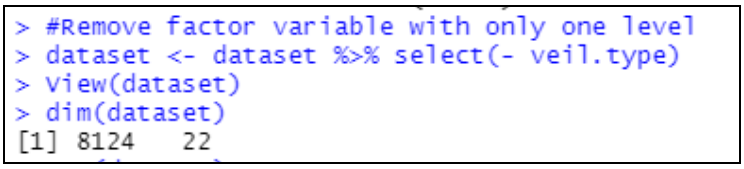

Fig.6 Removing Trivial Feature

3.3 Data Transformation

Data transformation is done to make dataset in better condition for statistical models. Renaming each variable makes the dataset more understandable and enhances the efficiency of data analysis. Using *glimpse* function check dataset whether names have been changed or not.

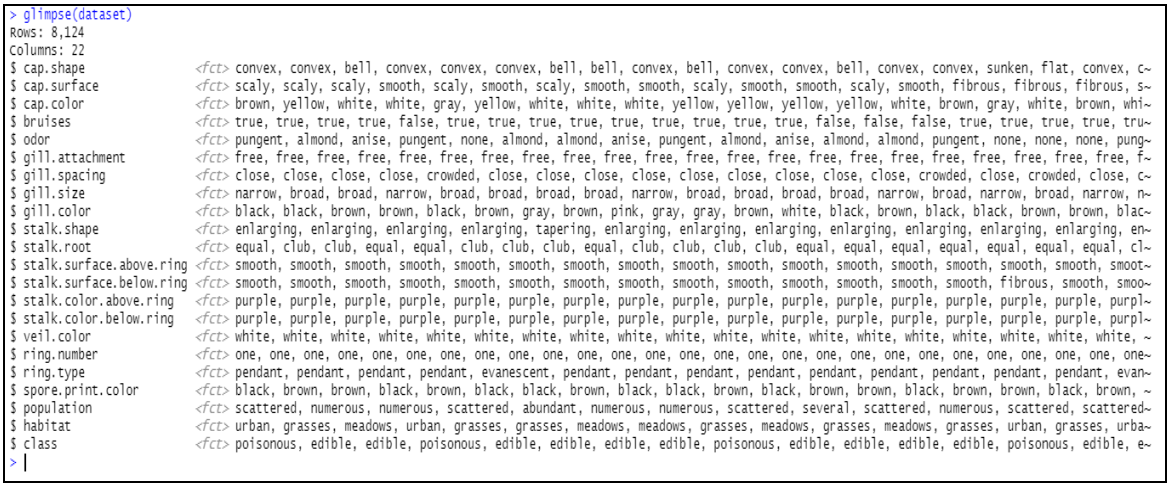

Fig.7 Dataset After Renaming the Variables.

4. Exploratory Data Analysis (EDA)

Exploratory data analysis is the fundamental analysis of the dataset to obtain important features in the dataset. EDA is not just a procedure that can be implemented, but an entire way to describe the dataset. The overall goal of EDA is to explain the hidden structures of a dataset. It predicts the development that helps in the selection of statistical methods. It helps in the selection of techniques and methods that are convenient. EDA can be performed in two ways non-graphical and graphical methods. For dataset mushroom, bar plots can be useful as compared to dot plots or scatter plots because for categorical data such visualization can be confusing. The first two attributes are plotted below – cap-shape, cap-surface.

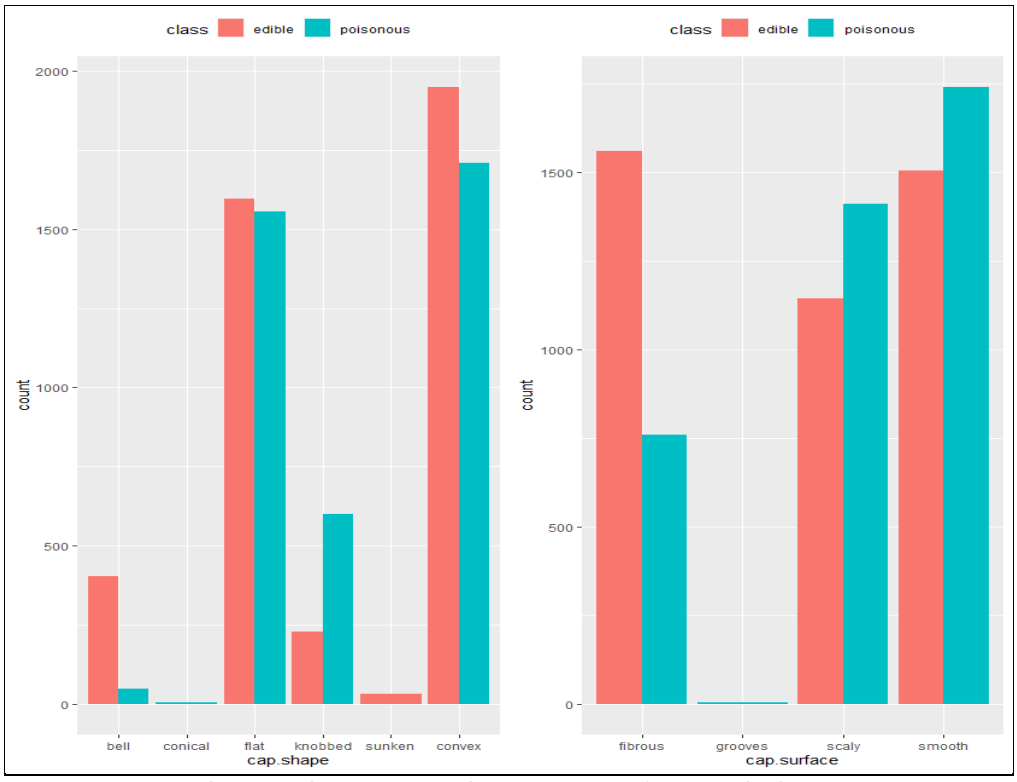

Fig.8 Mushroom Cap-Shape, Cap-Surface, and Class

In fig.8, mushrooms with flat cap-shape are largely edible(n=1596) and are equally poisonous(n=1556). The mushrooms are usually edible when cap-shape are bell(n=404) and convex(n=1948). All sunken cap-shaped mushrooms are edible and conical-shaped mushrooms are poisonous. The majority of mushrooms are poisonous when cap surface is scaly(n=1412) and smooth(n=1740). The majority of fibrous(n=1560) surface mushrooms are edible and all grooves surfaced mushrooms are poisonous.

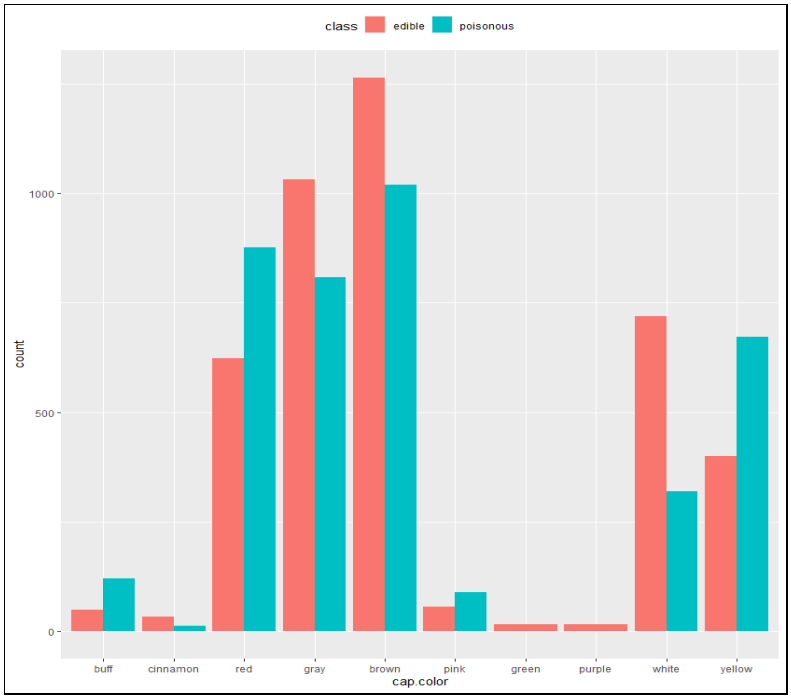

Fig.9 Mushroom Cap-Color and Class

In fig.9, the majority of mushrooms are edible when cap-color of mushrooms are grey(n=1032), brown(n=1264), cinnamon(n=32) and white(n=720). All the mushrooms are edible when mushroom cap-color is green and purple. The majority of mushrooms are poisonous when cap-color of mushrooms are buff(n=120), red(n=876), pink(n=88) and yellow(n=672).

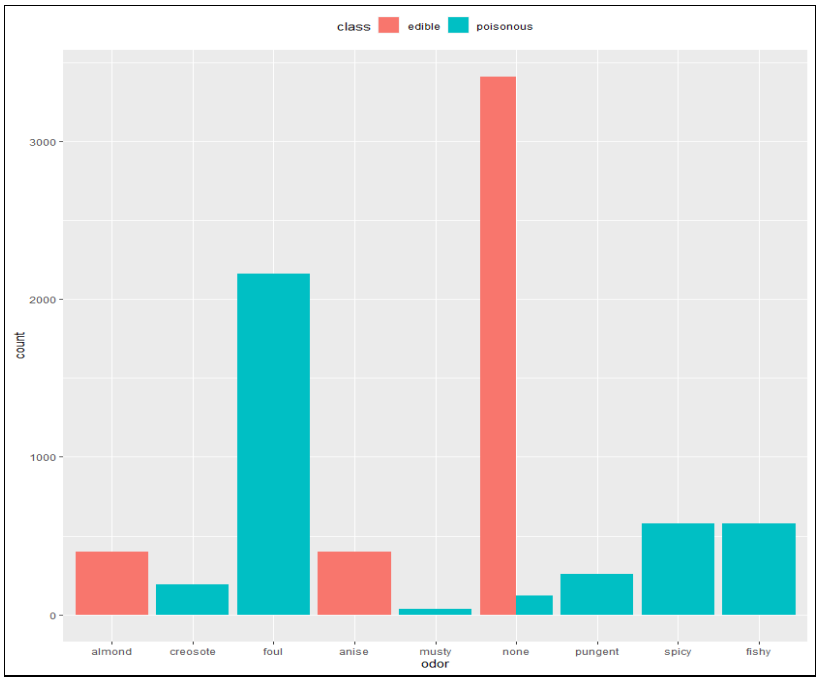

Fig.10 Mushroom Odor and Class

It can be easily noticed from fig.10, that all the mushrooms are edible when the odor is almond and anise. The majority of mushrooms are edible as well when there is

none(n=3408) odor of mushrooms. All the mushrooms are poisonous when the odor of mushrooms is creosote, foul, musty, pungent, spicy and fishy.

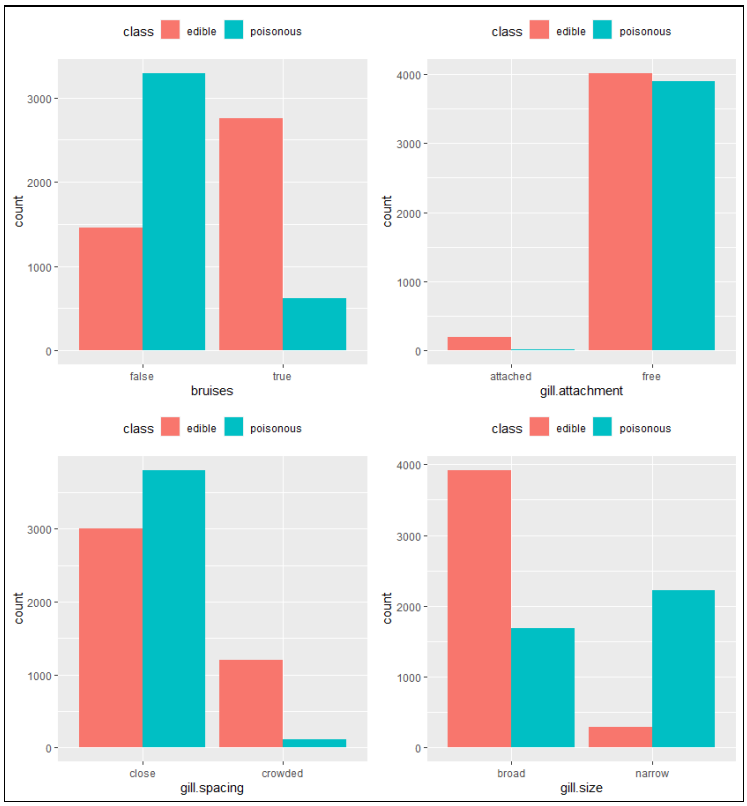

Fig.11 Mushroom Bruises, Gill-Attachment, Gill-Spacing, Gill-Size and Class

As shown in fig.11, the majority of mushrooms are edible if it has bruises and if it doesn't have bruises the majority of mushrooms are poisonous. If the mushroom gills are attached the majority of mushrooms are edible(n=192) and if the mushroom gills are free the majority of mushrooms are edible(n=4016) as well but equally poisonous(n=3898) also. When the gill-spacing of mushrooms is crowded the majority of mushrooms are edible and when the gill-spacing of mushrooms is close the majority of mushrooms are poisonous. Similarly, when the gill-size of mushrooms is broad the majority of mushrooms are edible and when the gill-size of mushrooms is narrow the majority of mushrooms are poisonous.

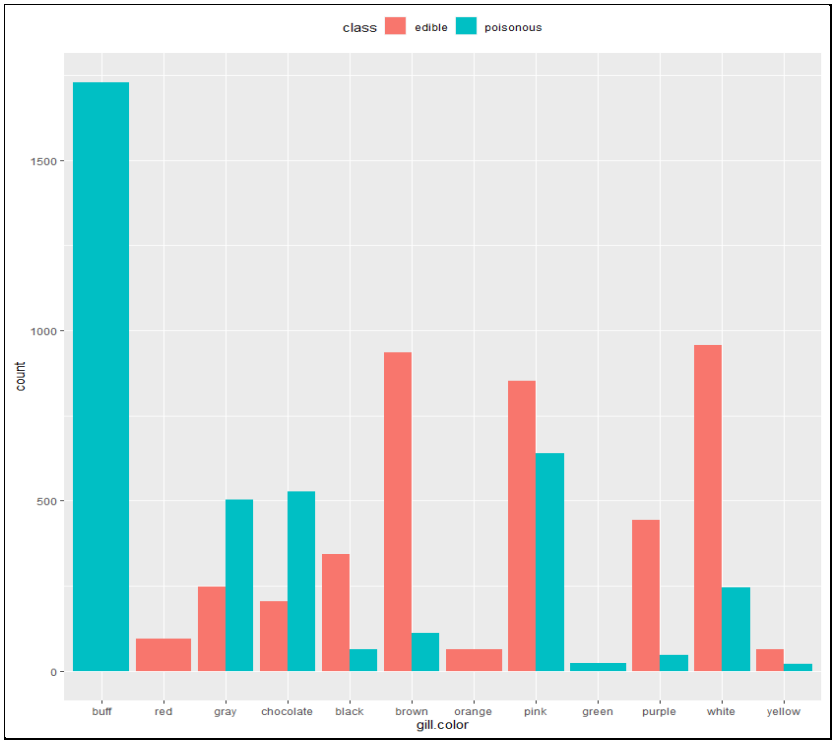

Fig.12 Mushroom Gill-Color and Class

In fig.12, when the gill-color of mushroom is red and orange all the mushrooms are edible and when the gill-color of mushroom is buff and green all the mushrooms are poisonous. The majority of mushrooms are edible when the gill-color are black(n=344), brown(n=936), pink(n=852), purple(n=444), white(n=956) and yellow(n=64). The majority of mushrooms are poisonous when the gill-color are grey(n=504) and chocolate(n=528).

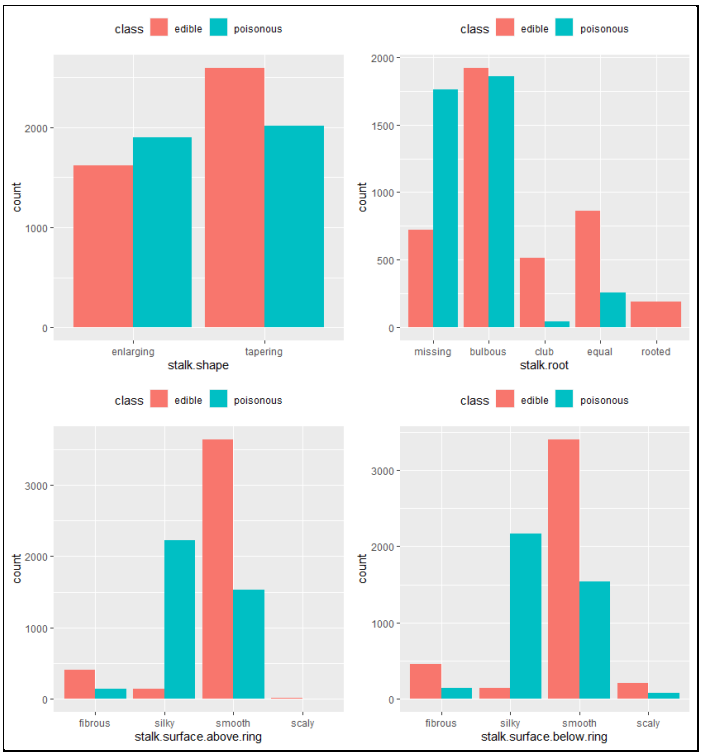

Fig.13 Mushroom Stalk-Shape, Stalk-Root, Stalk-Surface-Above-Ring, Stalk-Surface-Below-Ring and Class

It can be seen in fig.13, when the stalk-shape of mushroom is tapering the majority of mushrooms are edible(n=2592) and when the stalk-shape of mushroom is enlarging the majority of mushrooms are poisonous(n=1900). Similarly, the majority of mushrooms are edible when the stalk-root of mushrooms are bulbous(n=1920), club(n=512) and equal(n=864) and the majority of mushrooms are poisonous when the stalk-root of mushrooms are missing(n=1760). All the mushrooms are edible if the stalk-root of mushrooms are rooted. It is clear from the plot that majority of mushrooms are edible when stalk-surface are fibrous, smooth and scaly either above or below the ring, the majority of mushrooms are poisonous stalk-surface is silky either above or below the ring.

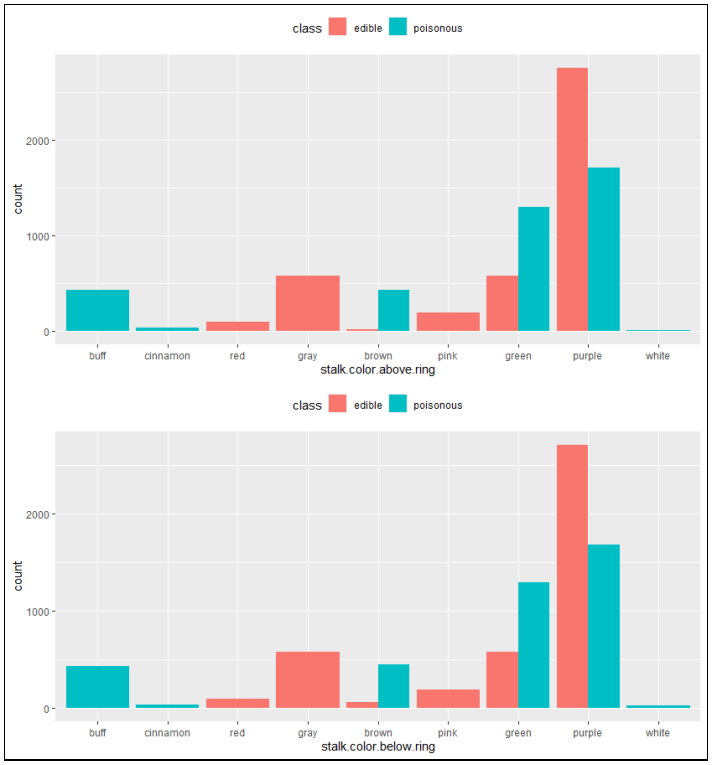

Fig.14. Mushroom Stalk-Color-Above-Ring, Stalk-Color-Below-Ring and Class

In fig.14, it is clear that all the mushrooms are edible when stalk-color of mushrooms are red, grey, and pink either above or below the ring similarly, the mushrooms are poisonous when stalk-color of mushroom are buff, cinnamon and white either above or below the ring. When the stalk-color of mushroom is purple the majority of mushrooms are edible either above or below the ring and when the stalk color of mushrooms are green and brown the majority of mushrooms are poisonous either above or below the ring.

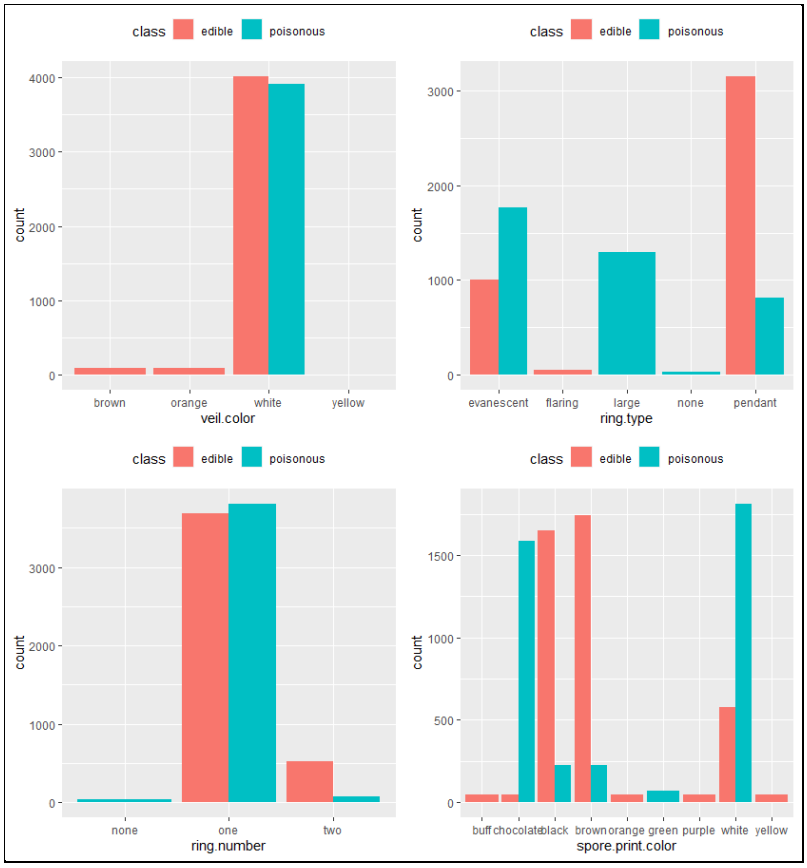

Fig.15 Mushroom Veil-Color, Ring-Type, Ring-Number, Spore-Print-Color and Color

In the fig.15, it is clearly visible that mushrooms are completely edible if the veil-color of mushrooms are brown and orange and completely poisonous if the veil-color of mushrooms is yellow. If the veil-color of mushrooms is white mushrooms are usually edible(n=4016) but equally poisonous(n=3908) as well. When the ring-type of mushrooms are large and none the mushrooms are completely poisonous and are completely edible if the ring-type of mushroom is flaring. The majority of mushrooms are edible if the ring-type is pendant and majority of mushrooms are poisonous if the ring-type is evanescent. If there is no ring in the mushroom, they are completely poisonous whereas majority of mushrooms are edible if they have two rings. If mushrooms have one ring then a majority of mushrooms are poisonous(n=3808) but are equally edible(n=3680) also. If the spore-print-color of mushrooms are black(n=1648) and brown(n=1744) the majority of mushrooms are edible and if the spore-print-color of mushrooms are chocolate(n=1548) and white(n=1812) the majority of mushrooms are poisonous. If spore-print-color of mushrooms are buff, orange, purple and yellow then mushrooms are completely edible and if the color is green the mushrooms are completely poisonous.

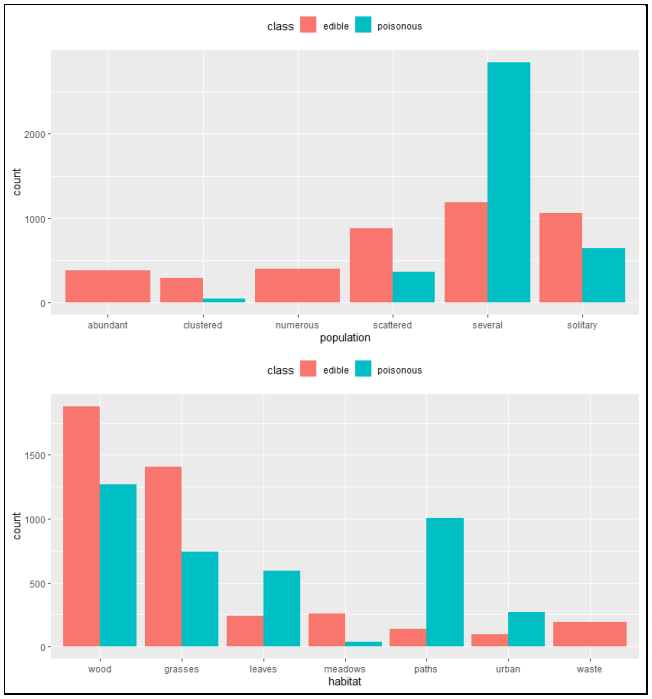

Fig.16 Mushroom Population, Habitat and Class

In fig.16, the majority of population are edible if the population is either clustered, scattered or solitary and a majority of mushroom are poisonous if a population is several. The mushrooms are completely edible if the population is abundant and numerous. Similarly, the majority of mushrooms are edible that habitat in woods, grasses and meadows areas and mushrooms are poisonous that habitat in leaves, paths and urban areas. The mushroom is completely edible in waste areas.

There is no perfect distribution in any of the features through which class of mushroom can be predicted. But some features like odor, ring-type, gill-color, stalk-color both above and below ring, gill-size have good separation to predict the class of mushrooms. Levels to variables entirely represent either edible or poisonous mushrooms. Odor has the clearest distribution that describes the class of mushrooms which is mainly poisonous.

The important part of EDA is recognizing of correlation between independent and dependent features in the dataset. This method can be useful in producing the correlation matrix. But the problem is that the dataset is categorical which will restrain the production of a correlation matrix. Moreover, each feature has several levels that are unordered. Such unordered categorical features are called nominal, meaning they don't have any natural order and therefore cannot be converted to natural numbers. 'In the measurement ranking, interval features are highest, ordinal (ordered) features are next, and nominal is lowest. Statistical methods for features of one type can also be used with features at higher levels but not at lower levels' (Agresti, 2013).

The Pearson's Chi-Square Test is the most common test used for calculating the correlation between categorical features. There are three forms of tests in the Pearson Chi-Square Test family – the goodness of fit, independence, and homogeneity test. Each of these tests is distinct with a different conclusion and explanation. The goodness of fit test is used when the sample is compared to a population with known parameters on a variable of interest. The test of independence is used to determine

whether two categorical features in a dataset are independent of each other or not. The test of homogeneity determines whether two or more independent datasets are different in their distribution on a single feature of interest. (Todd Michael Franke et al., 2012)

The Pearson's Chi-Square Test of Independence is the assumption test used for statistical analysis of categorical data. The correlations of categorical features can be understood by significance test and effect size (strength of dependency). The chi-square test of independence determines whether two categorical features are statistically dependent on each other or not. The chi-square test assumes a null hypothesis and an alternate hypothesis.
H_0: The two features are independent.
H_1: The two features are dependent.
In general, if the p-value that comes in the result is less than the predetermined value i.e., 0.05, then the null hypothesis is rejected. (Dutt, 2017)

Apply chi-square test of independence to check if the features are dependent or not. The correlation of every feature will be checked with the class of mushrooms. After this to check the effect size (strength of association) Goodman's Kruskal Tau test is applied. The calculation of correlation explains whether the strength of a relationship is weak or strong.

Features	X-Squared	df	p-value
Cap-shape	489.92	5	< 2.2e-16
Cap-surface	315.04	3	< 2.2e-16
Cap-color	387.6	9	< 2.2e-16
Bruises	2014.4	1	< 2.2e-16
Odor	7659.7	8	< 2.2e-16
Gill-attachment	133.99	1	< 2.2e-16
Gill-spacing	984.14	1	< 2.2e-16
Gill-size	2366.8	1	< 2.2e-16
Gill-color	3765.7	11	< 2.2e-16
Stalk-shape	84.14	1	< 2.2e-16
Stalk-root	1344.4	4	< 2.2e-16
Stalk-surface-above-ring	2808.3	3	< 2.2e-16
Stalk-surface-below-ring	2684.5	3	< 2.2e-16
Stalk-color-above-ring	2237.9	8	< 2.2e-16
Stalk-color-above-ring	2152.4	8	< 2.2e-16
Veil-color	191.22	3	< 2.2e-16
Ring-number	374.74	2	< 2.2e-16
Ring-type	2956.6	4	< 2.2e-16
Spore-print-color	4602	8	< 2.2e-16
Population	1929.7	5	< 2.2e-16
Habitat	1573.8	6	< 2.2e-16

Table-1 Chi-Squared Test Values

Since the p-value < 2.2e-16 is less than the predetermined value 0.05, thus rejecting null hypothesis and accepting alternate hypothesis, concluding that every feature is dependent on to class feature of mushroom.

On the basis of Chi-Square Test, the correlation of class feature with each feature in mushroom dataset, in descending order, as follows: odor (7659.4), spore-print-color (4602), gill-color (3765.7), ring-type (2959.6), stalk-surface-above-ring (2808.3), stalk-surface-below-ring (2684.5), gill-size (2366.8), stalk-color-above-ring (2237.9), stalk-color-below-ring (2152.4), bruises (2014.4), population (1929.7), habitat (1573.8), stalk-root (1344.4), gill-spacing (984.14), cap-shape (489.92), cap-color (387.6), ring-number (374.74), cap-surface (315.04), veil-color (191.22), gill-attachment (133.99), stalk-shape (84.14).

The ten features whose squared value is highest among all other features are selected for Goodman & Kruskal's Tau test. The strength of relationship of those features will be checked with the class feature of mushroom.

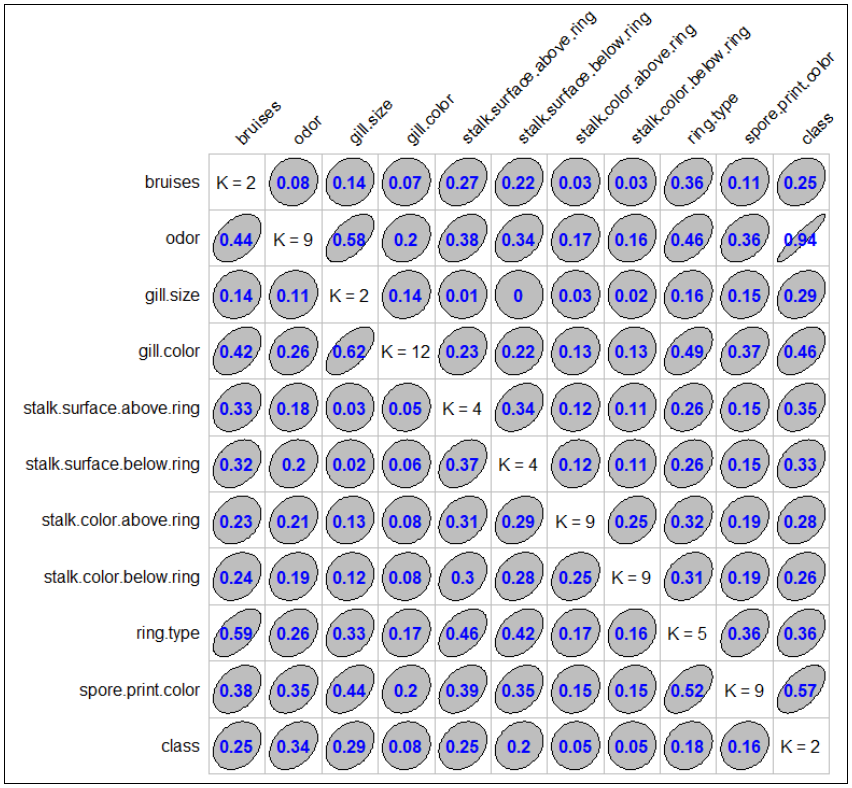

Fig.17 Strength of Association Plot

In fig.17, the diagonal element k is referred to as different levels of each feature. The off-diagonal numerical values represent the strength of relationship in between the features T(x,y) where x represents the features in the row and y represents the features in the column. From the plot, the most strong association, T(x,y) = 0.94 is between odor and class. The feature odor can perfectly predict the class of mushrooms.

5. Classification Models

Classification is the technique of determining or predicting of class that an object belongs to. The training dataset is used by classification model to map objects based on features to classes. To predict the class mushroom the classification models are used. The classification models used for prediction of mushroom class are decision trees, random forest.

Before building the models dataset is divided into two subsets: training and testing dataset. The training set trains the model, as in, the characteristics of data are learned within the dataset. The testing set tests the model so that efficacy and effectiveness of the model can be evaluated. The dataset is divided into an 80/20 ratio.

```
> set.seed(2000)
> dataset[,"train"] <- ifelse(runif(nrow(dataset)) < 0.8,1,0)
> train_dataset <- dataset[dataset$train == "1", ]
> test_dataset <- dataset[dataset$train == "0", ]
> view(train_dataset)
> view(test_dataset)
> train_dataset <- train_dataset[-23]
> test_dataset <- test_dataset[-23]
> view(train_dataset)
> view(test_dataset)
```

Fig.18 Splitting Dataset into Training and Testing Set

For splitting dataset, a pseudo-random value is allocated to all observations within dataset either 0 or 1, with the probability of 0.8 for 1 and 0.2 for 0. The *set.seed* function is used to generate the pseudo-random number and make sure that when the same seed value is used the experiment is reproducible. Then the value allocation is performed, creating a new feature “train” depicting which observation will be in training or testing data. Now dividing the dataset into training and testing sets, where the value of train is 1, the observation will be moved to a training set and where the value of train is 0, the observation will be moved to testing set. Once the dataset is divided the train feature is removed, as the effectiveness of the classification will be restrained by the randomness inherent in that feature.

5.1 Decision Tree

A decision tree is a robust classifier but comparatively straightforward to set up and use. It can be used on a variety of numerical and categorical dataset. It is a tree concept: every feature in the dataset is represented by a leaf node, with the branch representing all the values within each feature. To make a decision tree, there are three important information is required, the formula to target the feature, the data used to build the tree and the class method that is used to convert as a binary classifier within a package. Visualize the tree to find the results after a tree has been built.

```
> library(rpart)
> library(rpart.plot)
> model_tree <- rpart(class ~ ., data = train_dataset, method = "class")
> rpart.plot(model_tree, extra = 106)
> trees_pred <- predict(model_tree, newdata = test_dataset, type = "class")
```

Fig.19 Building Decision Tree Model

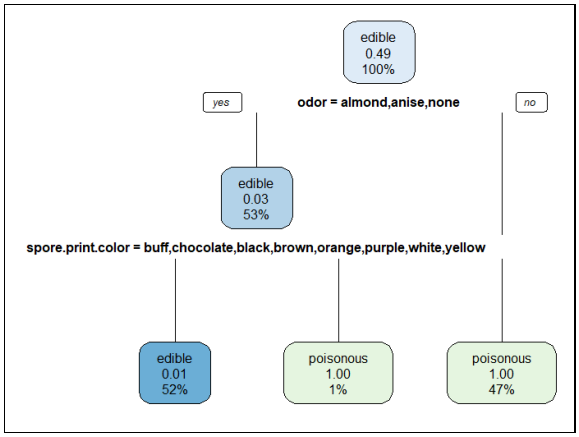

Fig.20 Decision Tree

The advantage of this model is that information gathered from the trees makes the decision more effective. Testing data must be prepared once the model has been trained. The model is applied to the test dataset to generate the predicted values. The *predict* function is used for this, it has three elements: a model to be used for prediction, testing dataset and type of prediction to be made.

Now that the classes have been predicted, confusion matrix can be created that allows to access the predictive performance of the model. The values that are predicted by the model are compared to the actual values of training data, are shown in confusion matrix. Four possible combinations are possible:

- True Positive(TP): If both values are 1(yes).
- True Negative(TN): If both values are 0(no).
- False Positive(FP): If predicted value is 1, but actual value is 0.
- False Negative(FN): If predicted value is 0, but actual value is 1.

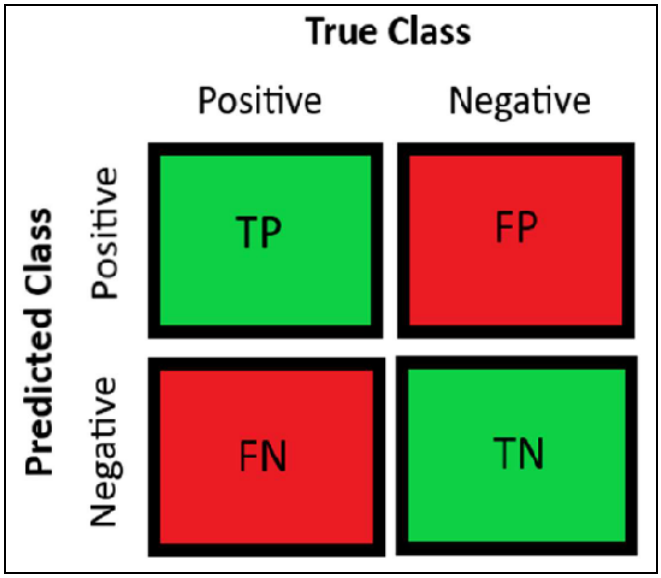

Fig.21 Confusion Matrix (Hashtings, 2021a)

Using *table* function confusion matrix can be created on the console, that shows the number of observations in each of the groups above.

```
> table(predicted = trees_pred, actual = test_dataset$class)
           actual
predicted   edible poisonous
  edible       877        11
  poisonous      0       732
```

Fig.22 Extracted Confusion Matrix

The performance of the classifier can be identified through each column in the confusion matrix, but accuracy will be use for overall performance as a metric. The model accuracy is the total number of TP and TN over the total number of observations in the test dataset. To calculate the accuracy of the model *mean* function is used.

```
> mean(trees_pred == test_dataset$class)
[1] 0.9932099
```

Fig.23 Calculating Accuracy of Model

The value 0.9932 is converted to percentage and can be read as 99.3%, meaning that model can predict 99.3% correct class.

5.2 Random Forest

While decision trees in an effective model, they can be sometimes overfitted and may not be the best predictive performance. The method that can overcome all these problems is the use of many trees at the same time, the method is known as random forest. A random forest is a collection of a decision tree, each with a subset of total features used. To get a single output for a specific data object the average of all outputs from all the trees in a forest is taken.

For random forest, *cforest* function is used, which trains the forest model. The function needs three things: response feature, dataset used to train the model, family of random forest to be used using *control* term. Two additional parameters are set in this element of function: *mtry*, sets the number of features to be used in a tree, and *ntree*, sets the number of trees that should be created in the forest. Again after training the model, *predict* function is used to apply the model on test dataset to evaluate the predictive capacity. In this, type should be "response", as probabilistic class prediction will be produced. Once again the confusion matrix will be created when the class has been predicted. Now again based on this matrix, accuracy will be calculated.

```
> library(party)
> forest_model <- cforest(class ~., data = train_dataset, control = cforest_unbiased(mtry = 15, ntree = 50))
> fm_pred <- predict(forest_model, newdata = test_dataset, type = "response")
> table(predicted = fm_pred, actual = test_dataset$class)
           actual
predicted   edible poisonous
  edible       877         4
  poisonous      0       739
> mean(fm_pred == test_dataset$class)
[1] 0.9975309
```

Fig.24 Random Forest

The value can be read as 99.7%. Random Forests are also useful because they help to identify the most important features in the dataset that contributes the most to the prediction made by the model.

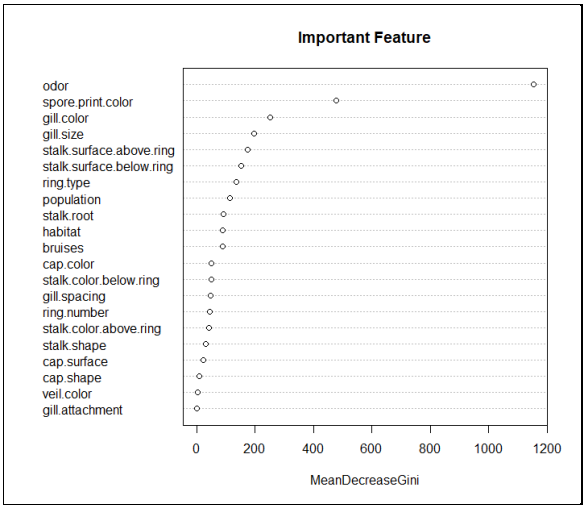

Fig.25 Important Feature Plot

From the plot the features that came out as the most important feature in the prediction are odor and spore-print-color, these features are same that are identified by decision tree. This concludes that both models are strong and results that are produced are reliable.

6. Model Evaluation

For model evaluation first the overall accuracy of models will be calculated using the formula given below (Hashtings, 2021b):

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

Fig.26 Accuracy Formula

For Decision Tree we have values:
TP- 877
TN- 732
FP- 11
FN- 0

```
> TP <- 877
> TN <- 732
> FP <- 11
> FN <- 0
> Acc <- (TP+TN) / (TP+TN+FP+FN)
> Acc
[1] 0.9932099
>
```

Fig.27 Accuracy of Decision Tree

For Random Forest we have values:
TP- 877
TN- 739
FP- 4
FN- 0

```
> TP <- 877
> TN <- 739
> FP <- 4
> FN <- 0
> Acc <- (TP+TN) / (TP+TN+FP+FN)
> Acc
[1] 0.9975309
>
```

Fig.28 Accuracy of Random Forest

According to accuracy paradox, the accuracy value of decision tree is 99.3% and random forest is 99.7% which is a positive result. How well the models performed is still not clear. To find this clearly, two additional metrics are calculated: F1 score and Matthews Correlation Coefficient (MCC).

6.1 F1 Score

For F1 score to be calculated, precision and recall must be determined by the formulas given below:

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

Fig.29 Precision and Recall Formulas

For Decision Tree, the precision value is 0.987 and recall value is 1. Such values for the model, indicate that model is capable of identifying whether the class of mushroom is edible or poisonous.

```
> precision <- TP / (TP+FP)
> recall <- TP / (TP+FN)
> precision
[1] 0.9876126
> recall
[1] 1
>
```

Fig.30 Precision & Recall Value for Decision Tree

For Random Forest, the precision value is 0.995 And recall value is 1. The model is extremely capable of identifying the class of mushroom.

```
> precision <- TP / (TP+FP)
> recall <- TP / (TP+FN)
> precision
[1] 0.9954597
> recall
[1] 1
```

Fig.31 Precision & Recall Value for Random Forest

Now with both precision and recall value F1 score is calculated with formula given below:

$$F1 = 2 \cdot \frac{P \cdot R}{P + R}$$

Fig.32 F1 Score Formula

For Decision tree, F1 score is 0.993 which is extremely strong. One of the reasons for such value is the large number of true positive values identified by the model.

```
> #F1 Score Decision Tree
> F1 <- 2 * ((precision * recall) / (precision + recall))
> F1
[1] 0.9937677
>
```

Fig.33 F1 Score for Decision Tree

For Random Forest, F1 Score is 0.997, a large number of true positive values can be one of the reasons for these extremely strong value.

```
> #F1 Score Random Forest
> F1 <- 2 * ((precision * recall) / (precision + recall))
> F1
[1] 0.9977247
>
```

Fig.34 F1 Score for Random Forest

6.2 Matthews Correlation Coefficient (MCC)

Now, MCC value for the models will be calculated. The values that are found in confusion matrix will be used again to calculate this using the formula given below:

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP) \cdot (TP + FN) \cdot (TN + FP) \cdot (TN + FN)}}$$

Fig.35 MCC Formula

For Decision Tree, the MCC value is 0.986, when combined with accuracy (0.993) and F1 score (0.987) suggests that the model is satisfactory in predicting the class of mushrooms.

```
> # MCC for Decision Tree
> mcc <- ((TP * TN) - (FP * FN)) / (((TP+FP) * (TP+FN) * (TN+FP) * (TN+FN)) ^ 0.5)
> mcc
[1] 0.9864031
>
```

Fig.36 MCC Value for Decision Tree

For Random Forest, the MCC value is 0.995, when combined with accuracy (0.997) and F1 score (0.995) suggests that model is extremely satisfactory.

```
> # MCC for Random Forest
> mcc <- ((TP * TN) - (FP * FN)) / (((TP+FP) * (TP+FN) * (TN+FP) * (TN+FN)) ^ 0.5)
> mcc
[1] 0.995038
>
```

Fig.37 MCC Value for Random Forest

7. Conclusion

The analysis of dataset evaluates that the feature “odor” is best for the prediction of mushroom classes (either edible or poisonous) in all the ways that are analyzed. The bar plot of odor has the 100% of cases for each level representing whether mushroom is edible or poisonous. It also indicates that odor has the indication of most poisonous mushroom. In chi-square test, odor has highest value for correlation with mushroom class and again in strength of relationship test it has highest association. The classification models, decision tree whose accuracy is 99.3% and random forest whose accuracy is 99.7% depict that odor is the most important feature for predicting whether the mushrooms are edible or poisonous. After model evaluation, it is clear that both models are robust to predict the mushroom class. The study suggests that the random forest model is the most accurate in predicting the mushroom classes.

References

AGRESTI, A. 2013. *Categorical Data Analysis*, Wiley.

DUTT, A. 2017. To eat or not to eat! That's the question? Measuring the Association Between Categorical Variables. *Stories Data Speak* [Online]. Available from: https://duttashi.github.io/blog/to-eat-or-not-to-eat/ [Accessed June 03.

HASHTINGS, D. 2021a. Classification & Clustering Workshop. *Northumbria University*.

HASHTINGS, D. 2021b. Model Evaluation. *Northumbria University*.

SHUHAIDA ISMAIL, AMY ROSSHAIDA ZAINAL & MUSTAPHA, A. 2018. Behavioural Features for Mushroom Classification.

TODD MICHAEL FRANKE, TIMOTHY HO & CHRISTIE, C. A. 2012. The Chi-Square Test: Often Used and More Often Misinterpreted. *American Journal of Evaluation***,** 11.