Mushroom Classification
Report

An effort to develop a reliable method for predicting fungui toxicity using a statistical analysis of their observed characeristics.

C. Kelly, T. Kulich, V.J. Vivas Chacon

Contents

Project Overview

The purpose of this project is to analyse a dataset following the CRISP-DM method. A dataset of recordings of mushrooms that are classified as poisonous and edible was extracted from Kaggle and used to implement a business solution for expected stakeholders. The dataset was specifically obtained from the following webpage: https://www.kaggle.com/uciml/mushroom-classification

The raw data was examined to reveal a list of categorical features, most of whom were nominal. Trends within the data were noted for their expected value to the model.

The preliminary analysis highlighted several concerns with the dataset, including the presence of extreme outliers, inter-feature correlation, high stability features, single value columns and missing data. Measures were taken in alleviating these issues and reducing the effect they would have on the end model.

The modelling process involved applying Decision Tree, Naïve Bayesian, and Rule Induction algorithms to the cleaned data. The end analysis of each of the algorithms was integrated into an ensemble majority classifier.

The results of the modelling process were evaluated using 10-fold cross-validation at each of the modelling steps. The evaluated model was confirmed to be of high accuracy in the precision-recall table, with a precision of 100% and a recall of 99-100%. A result such as this indicates exceptional quality of the data.

The model was deployed as a proof-of-concept application, a lightweight program written in python illustrating the functionality of a mushroom classifier and its business application. The program was written with the intention for it to be developed further with experienced app producers and delivered with appropriate monetisation. Recommendations for such a finished product are outside the scope of this project.

1: Business Understanding

1.1 Business problem framing

The dataset chosen has the potential to solve business problems for two main stakeholders.

Developers of a foraging app.

A profit-driven enterprise that specialises in existing foraging and outdoor tools has a business interest in a virtual tool that can be marketed and monetised towards an enthusiast demographic. If developed appropriately, a program utilising a classification model to determine if a mushroom is potentially poisonous can add additional functionality to existing products or as a standalone product.

Safety regulators for national parks and woodlands.

As part of a yearly audit, the safety management of a parks local fungi can be enhanced with a classification tool. This can be used to identify previously unseen specimens in the park, as well as invasive species because of climate change (Lockhart SR, 2017).

These problems seem to be well suitable for analytics solution with the data obtained as each kind of mushroom has several features which seem to be determining the poisonous/ non-poisonous character of each mushroom.

1.2: Business objectives

The main business objective for a private company is to increase profits. This would be achieved through downloads of the developed application, advertising in the app, and the acquisition of new capital from new clients through increased visibility of the company.

An objective list for a private company may take the following form.

Level	***Criteria***
Internal	Working and the reliable app will be developed meeting schedule and within budget
Medium-term	Profits from app downloads and advertising that will significantly exceed model and app development costs.
Long term	Acquire new customers through increased visibility of the company
For Customer	A reliable and stable tool which will help customer to easily distinguish edible and non-edible mushrooms

Additionally, the main business objective for a safety regulator would be the minimisation of risk to park guests and the balance of the park ecosystem.

An objective list for the park authorities may take the following form.

Level	*Criteria*
Internal	Working and reliable tool will be developed meeting schedule and within budget
Medium-term	Mushrooms currently recorded as being on the park premises would be classified with associated risks and decisions on cultivation or removal to be made.
Long term	Invasive species recently recorded on-site would be readily assessed using the obtained model, and decisions on the model applicability and future preventative measure would be made.
For Auditor	A reliable and stable tool that will help the auditor to easily distinguish edible and non-edible mushrooms

For both stakeholders, the model's use will depend on the accuracy of the classification results. An inaccurate model would pose a problem from a risk management standpoint if a poisonous mushroom were incorrectly identified as edible.

1.3: Assumptions and Caveats

The assessment and implementation of this model should be made with the following cautions of the data integrity. It is assumed the data used in the model is accurate and reliable. If systematic or experimental errors have accumulated in the dataset that was not otherwise recorded, this will influence the reliability of the model in a way that will not be easily detected through analysis.

As per the data description on Kaggle, the dataset includes descriptions of samples of 23 species of gilled mushrooms in the Agaricus and Lepiota Families. This means that the model is limited in the scope of the mushroom species that can be assessed.

The data was gathered in 1981 for the Audubon Society Field Guide to North American Mushrooms, as such it may be out of date for measurement techniques. The data was gathered in 1981 for the Audubon Society Field Guide to North American Mushrooms. As such, it may be out of date for measurement techniques.

According to the description on Kaggle, each specimen of the gathered dataset was identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. While this is important for minimising risk, it means that a mushroom predicted to be poisonous may be potentially edible. Practically speaking it is best to still regard it as poisonous.

1.4: Risk Management

With a subject as potentially dangerous as toxic mushrooms, an inaccurate model will inevitably lead to a mushroom falsely identified as edible.

From a business standpoint, if an inaccurate model is produced and consulted, the company must make the legal arrangements necessary to reduce liability in the event of accidental poisoning from a false diagnosis. A nature park must also prepare for the possibility of a guest poisoning themselves and should prepare emergency strategies.

From an analysis standpoint, inaccurate data must be identified and expunged from the model. The performance of the model should be scrutinised at the 95% confidence levelOverfitted model, failure to generalise.

As it is for other health related diagnoses (Hazra, 2017). The model must also be tested on new data with cross-validation to identify overfitting allowing for regularisation of the features.

1.5: Data mining goals

Considering the risks and business needs, the goal of the data mining and analysis process is to develop a deployable classification model with high accuracy above the 95% accuracy threshold. The primary focus is to minimise the false-positive criterium (poisonous mushrooms classified as edible by model), even at the cost of the false negative criterium being enlarged.

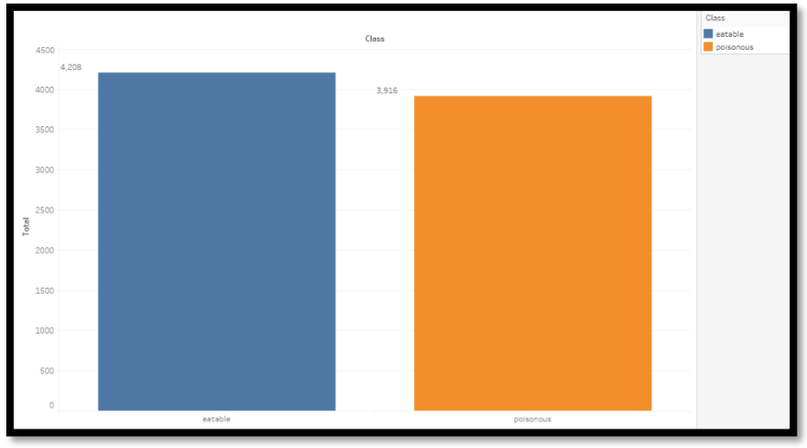
2: Data Understanding

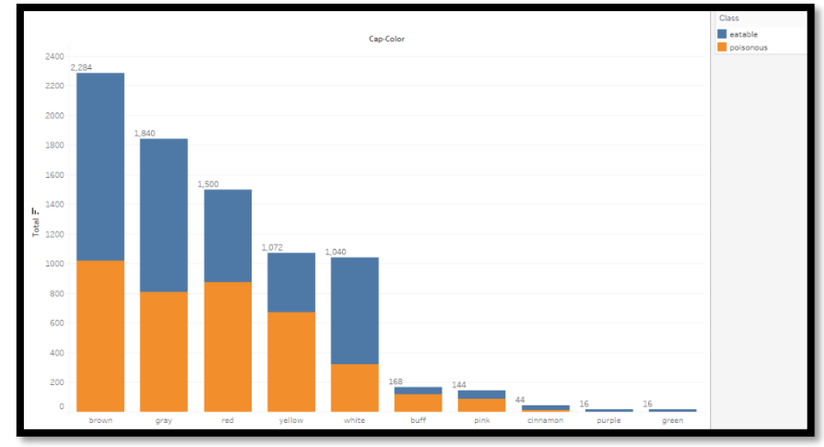
2.1: Analytics Base Table.

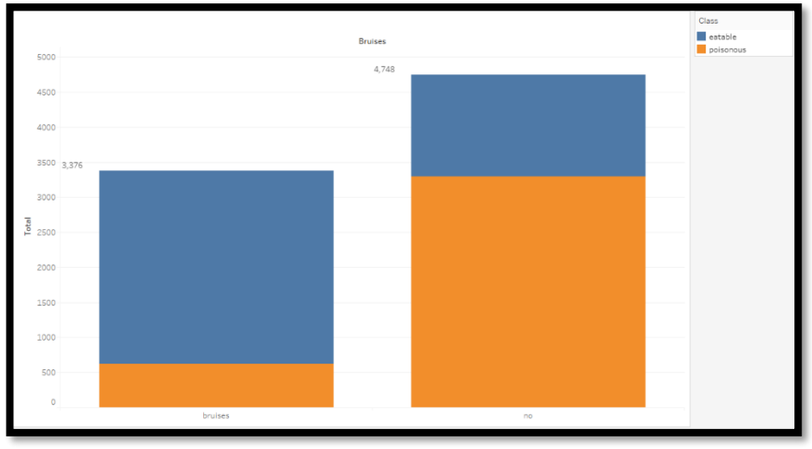
ID	Features	Description	Count	Missing	%Missing	Cardinality	1st Mode	1st Mode Freq	1st %Mode	2nd Mode	2nd Mode Freq	2nd %Mode
1	class	edible = e, poisonous = p	8124	0	0.00%	2	e	4208	51.80%	p	3916	48.20%
2	cap-shape	bell = b, conical = c, convex = x, flat = f, knobbed = k, sunken = s	8124	0	0.00%	6	x	3656	45.00%	f	3152	38.80%
3	cap-surface	fibrous = f, grooves = g, scaly = y, smooth = s	8124	0	0.00%	4	y	3244	39.93%	s	2556	31.46%
4	cap-color	brown = n, buff = b, cinnamon = c, gray = g, green = r, pink = p, purple = u, red = e, white = w, yellow = y	8124	0	0.00%	10	n	2284	28.11%	g	1840	22.65%
5	bruises	yes = t, no = f	8124	0	0.00%	2	f	4748	58.44%	t	3376	41.56%
6	odor	almond = a, anise = l, creosote = c, fishy = y, foul = f, musty = m, none = n, pungent = p, spicy = s	8124	0	0.00%	9	n	3528	43.43%	f	2160	26.59%
7	gill-attachment	attached = a, descending = d, free = f, notched = n	8124	0	0.00%	2	f	7914	97.42%	a	210	2.58%
8	gill-spacing	close = c, crowded = w, distant = d	8124	0	0.00%	2	c	6812	83.85%	w	1312	16.15%
9	gill-size	broad = b, narrow = n	8124	0	0.00%	2	b	5612	69.08%	n	2512	30.92%
10	gill-color	black = k, brown = n, buff = b, chocolate = h, gray = g, green = r, orange = o, pink = p, purple = u, red = e, white = w, yellow = y	8124	0	0.00%	12	b	1728	21.27%	p	1492	18.37%
11	stalk-shape	enlarging = e, tapering = t	8124	0	0.00%	2	t	4608	56.72%	e	3516	43.28%
12	stalk-root	bulbous = b, club = c, cup = u, equal = e, rhizomorphs = z, rooted = r, missing =?	5644	2480	30.53%	5	b	3776	66.90%	e	1120	19.84%
13	stalk-surface-above-ring	fibrous = f, scaly = y, silky = k, smooth = s	8124	0	0.00%	4	s	5176	63.71%	k	2372	29.20%
14	stalk-surface-below-ring	fibrous = f, scaly = y, silky = k, smooth = s	8124	0	0.00%	4	s	4936	60.76%	y	2304	28.36%

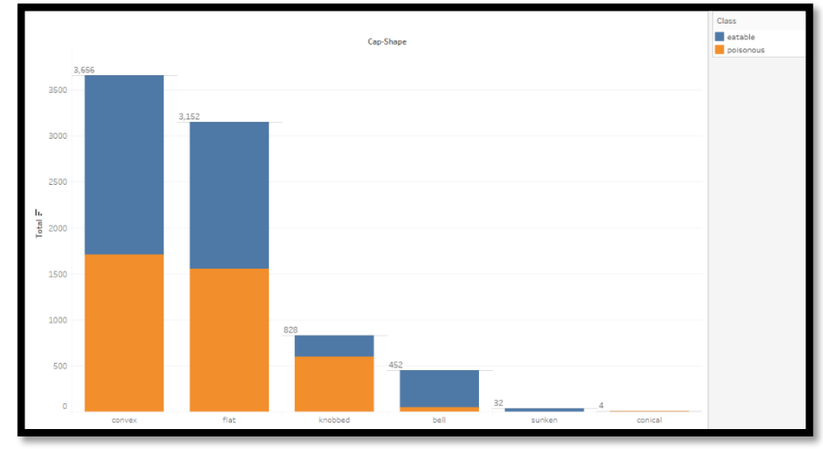
ID	Features	Description	Count	Missing	%Missing	Cardinality	1st Mode	1st Mode Freq	1st %Mode	2nd Mode	2nd Mode Freq	2nd %Mode
15	stalk-color-above-ring	brown = n, buff = b, cinnamon = c, gray = g, orange = o, pink = p, red = e, white = w, yellow = y	8124	0	0.00%	9	w	4464	54.95%	p	1872	23.04%
16	stalk-color-below-ring	brown = n, buff = b, cinnamon = c, gray = g, orange = o, pink = p, red = e, white = w, yellow = y	8124	0	0.00%	9	w	4384	53.96%	p	1872	23.04%
17	veil-type	Partial = p, Universal = u	8124	0	0.00%	1	p	8124	100.00%	NULL	0	0.00%
18	veil-color	brown = n, orange = o, white = w, yellow = y	8124	0	0.00%	4	w	7924	97.54%	n	96	1.18%
19	ring-number	none = n, one = o, two = t	8124	0	0.00%	3	o	7488	92.17%	t	600	7.39%
20	ring-type	cobwebby = c, evanescent = e, flaring = f, large = l, none = n, pendant = p, sheathing = s, zone = z	8124	0	0.00%	5	p	3968	48.84%	e	2776	34.17%
21	spore-print-color	black = k, brown = n, buff = b, chocolate = h, green = r, orange = o, purple = u, white = w, yellow = y	8124	0	0.00%	9	w	2388	29.39%	n	1968	24.22%
22	population	abundant = a, clustered = c, numerous = n, scattered = s, several = v, solitary = y	8124	0	0.00%	6	v	4040	49.73%	y	1712	21.07%
23	habitat	grasses = g, leaves = l, meadows = m, paths = p, urban = u, waste = w, woods = d	8124	0	0.00%	7	d	3148	38.75%	g	2148	26.44%

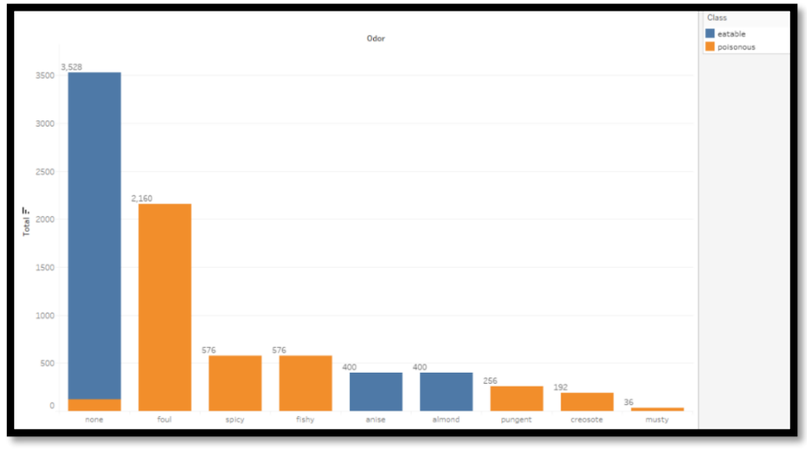
2.2: Bar chart of categorical features.

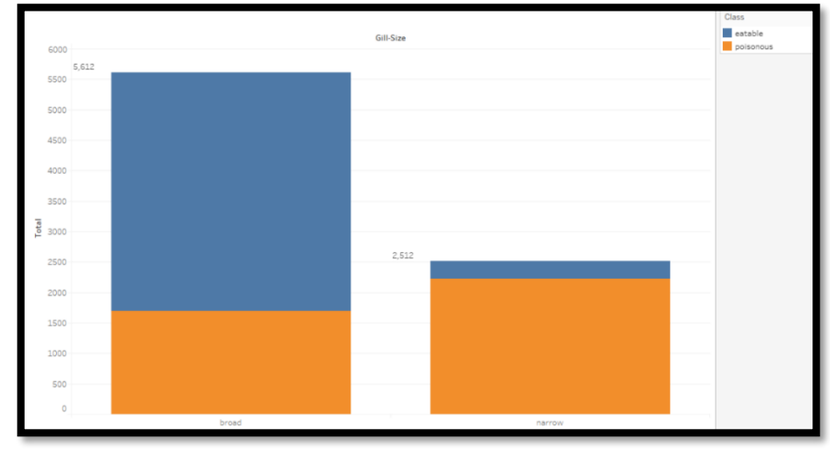


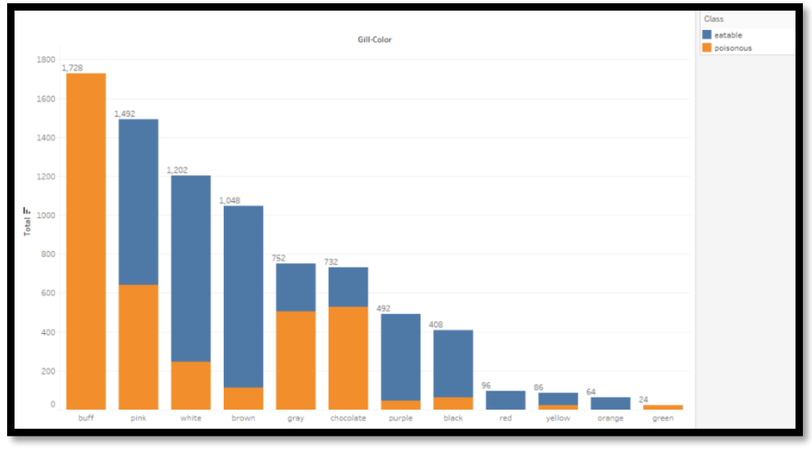


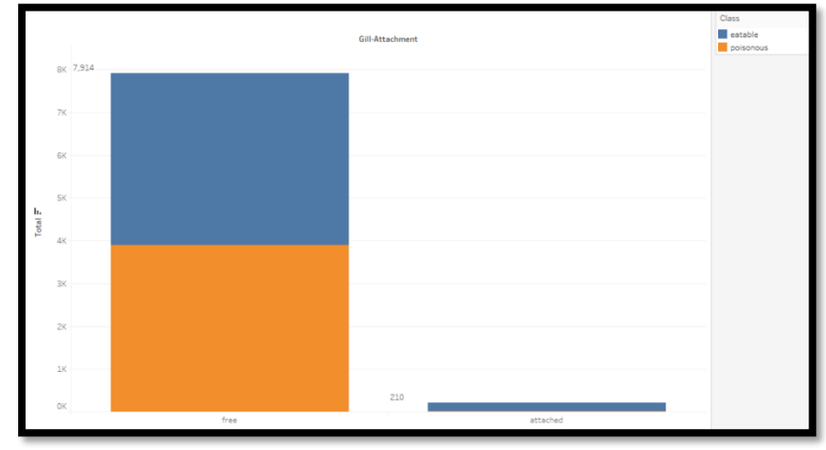


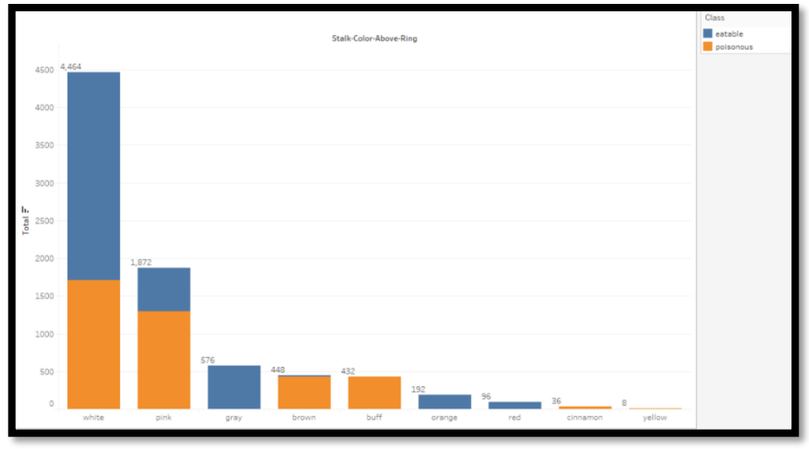


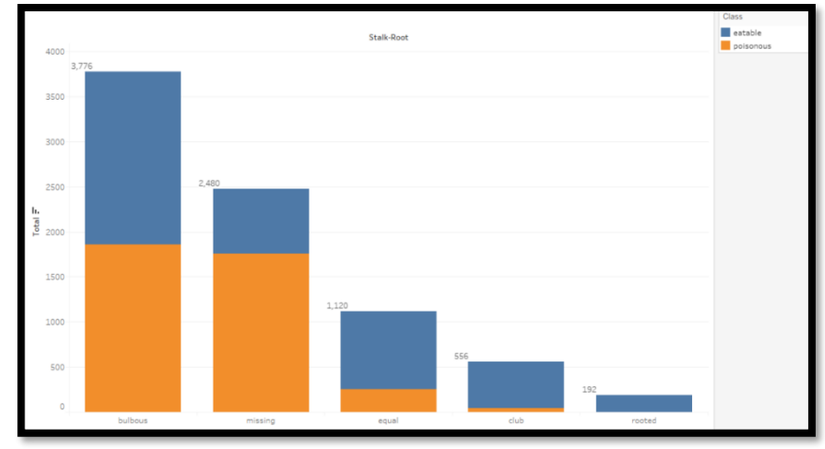

Odor
Class
eatable
poisonous
Total F..
3,528
2,160
576
576
400
400
256
192
36
none
foul
spicy
fishy
anise
almond
pungent
creosote
musty

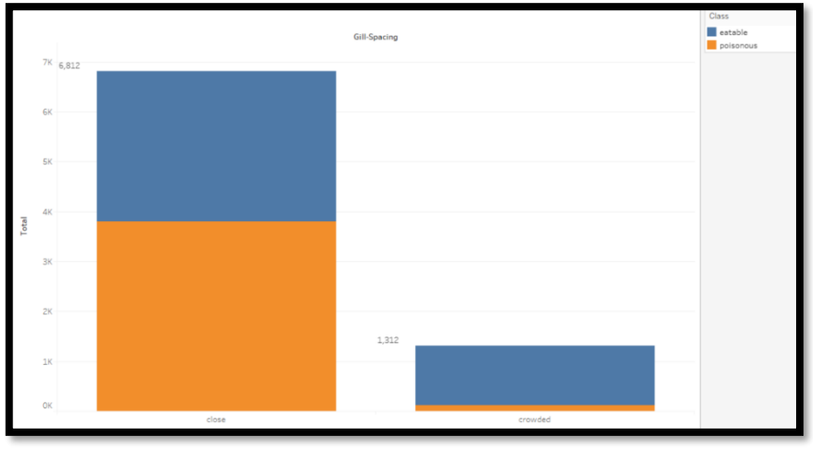

Gill-Size
Class
eatable
poisonous
Total
5,612
2,512
broad
narrow

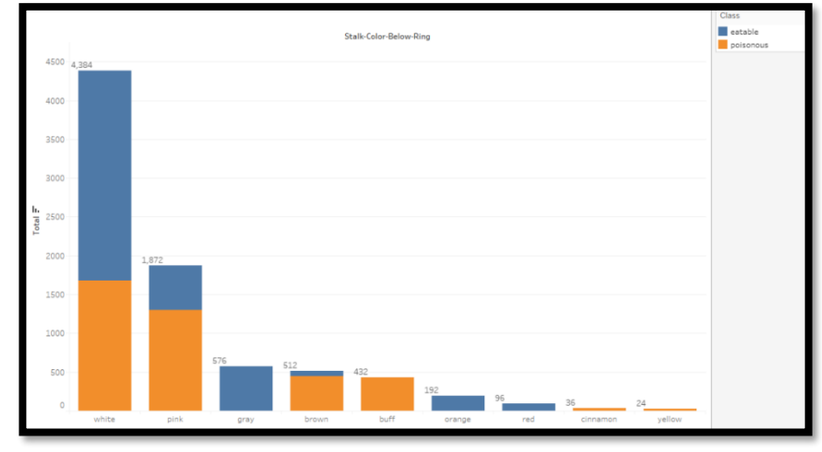

Gill-Color
Class
eatable
poisonous
Total F..
1,728
1,492
1,202
1,048
752
732
492
408
96
86
64
24
buff
pink
white
brown
gray
chocolate
purple
black
red
yellow
orange
green

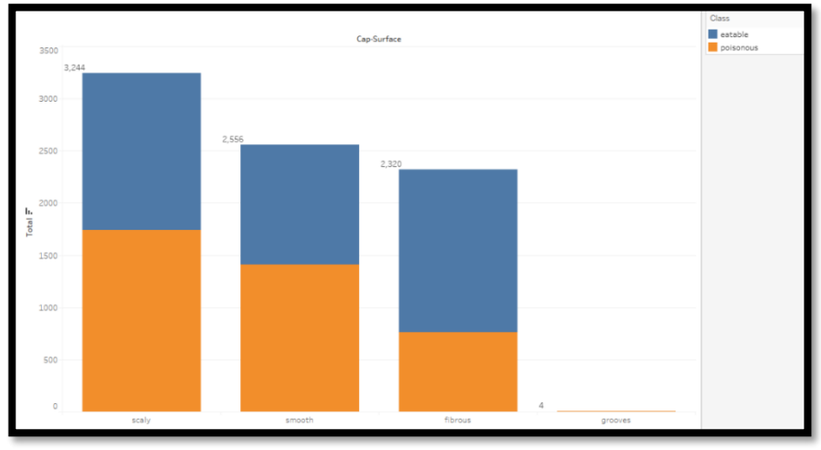

Gill-Attachment
Class
eatable
poisonous
Total F..
7,914
210
free
attached

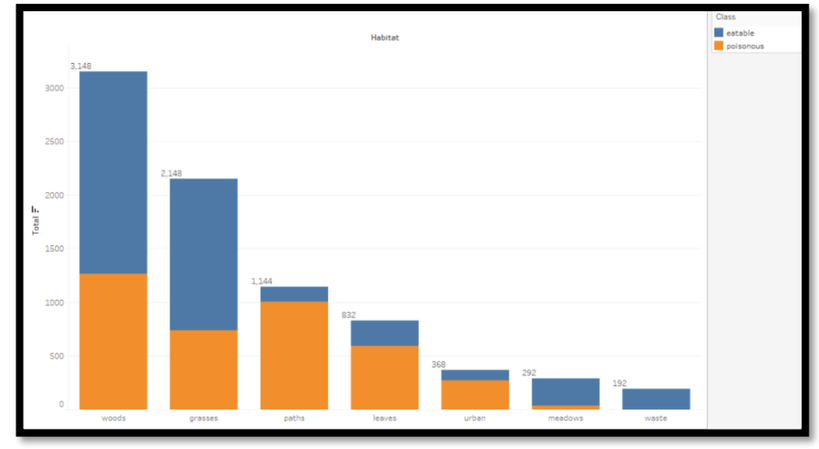

Stalk-Color-Above-Ring
Class
eatable
poisonous
Total F..
4,464
1,872
576
448
432
192
96
36
8
white
pink
gray
brown
buff
orange
red
cinnamon
yellow

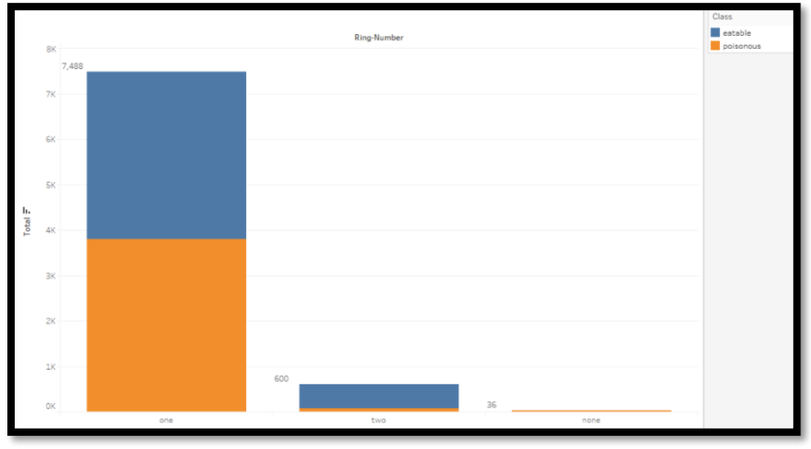

Stalk-Root
Class
eatable
poisonous
Total F..
3,776
2,480
1,120
556
192
bulbous
missing
equal
club
rooted

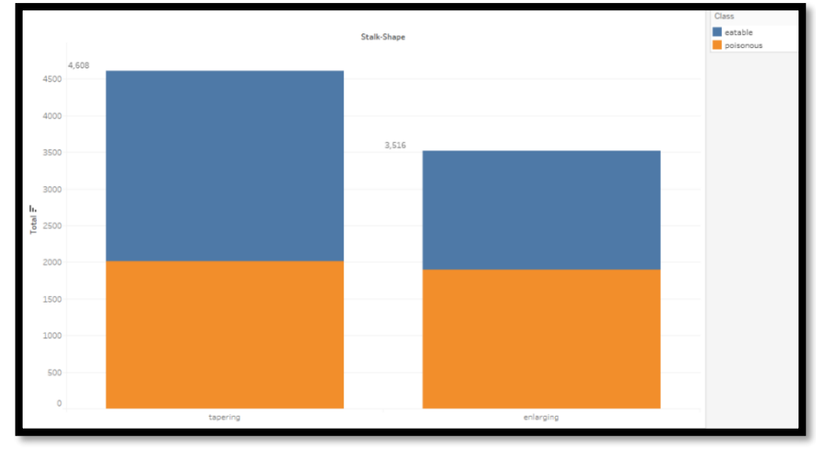

Gill-Spacing
Class
eatable
poisonous
Total
6,812
1,312
close
crowded

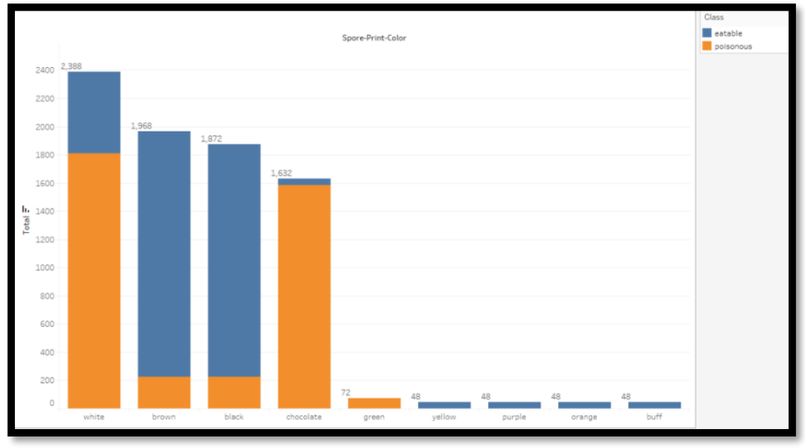

Stalk-Color-Below-Ring
Class
eatable
poisonous
Total F..
4,384
1,872
576
512
432
192
96
36
24
white
pink
gray
brown
buff
orange
red
cinnamon
yellow

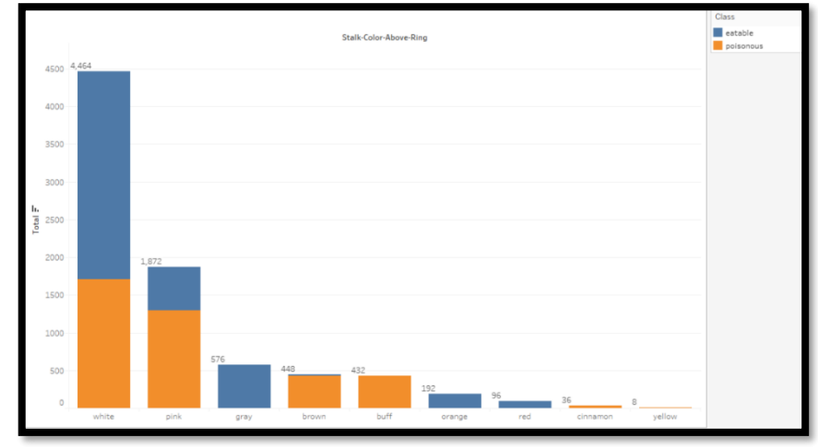

Cap-Surface
Class
eatable
poisonous
Total F..
3,244
2,556
2,320
4
scaly
smooth
fibrous
grooves

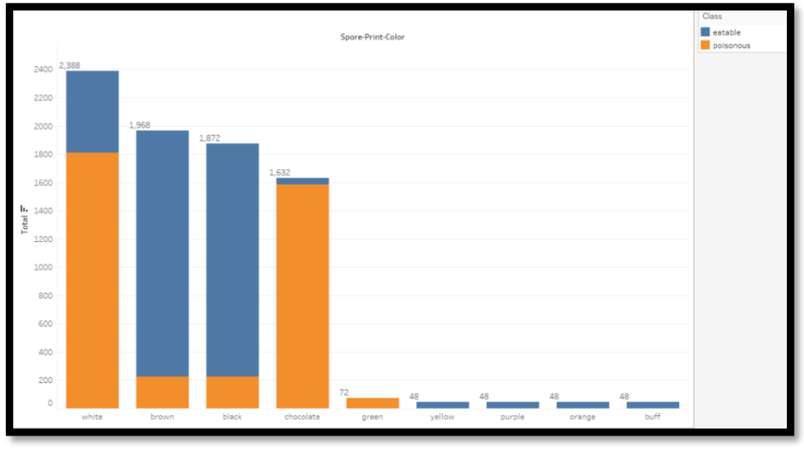

Habitat
Class
eatable
poisonous
Total F..
3,148
2,148
1,144
832
368
292
192
woods
grasses
paths
leaves
urban
meadows
waste

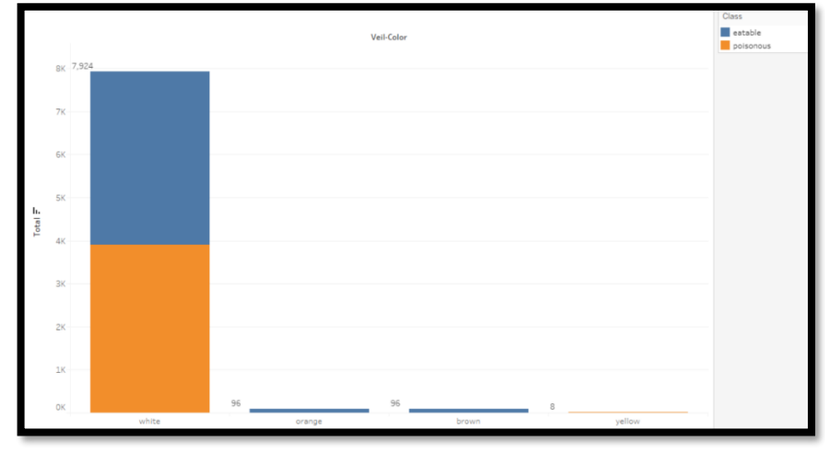

Ring-Number
Class
eatable
poisonous
Total F..
7,488
600
36
one
two
none

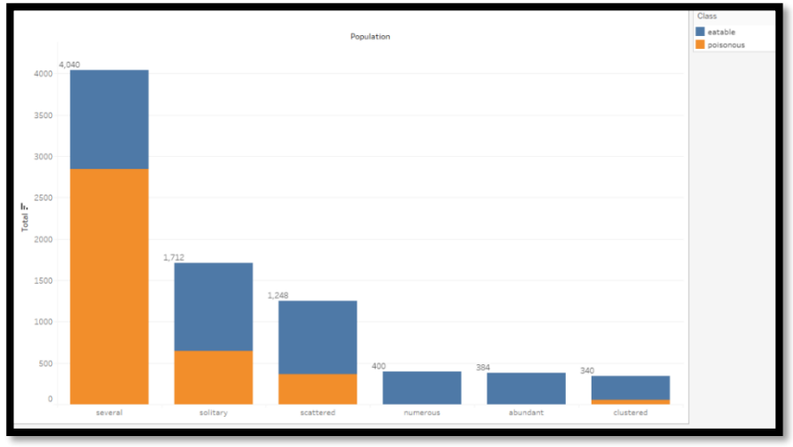

Stalk-Shape
Class
eatable
poisonous
Total F..
4,608
3,516
tapering
enlarging

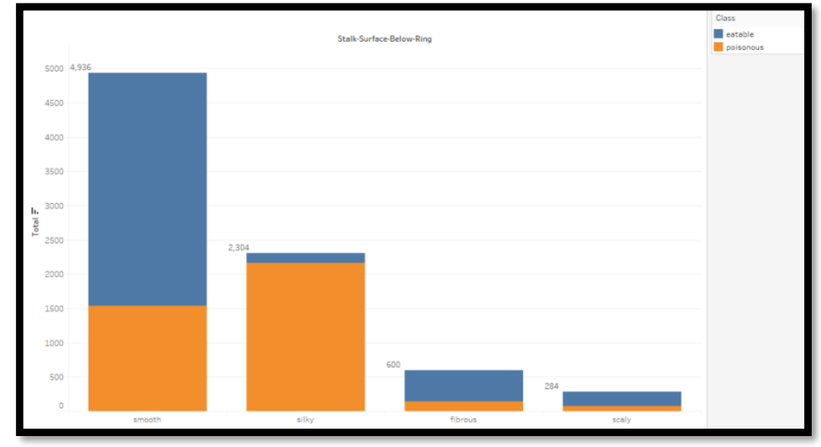

Spore-Print-Color
Class
eatable
poisonous
Total F..
2400
2200
2000
1800
1600
1400
1200
1000
800
600
400
200
0
2,388
1,968
1,872
1,632
72
48
48
48
48
white
brown
black
chocolate
green
yellow
purple
orange
buff

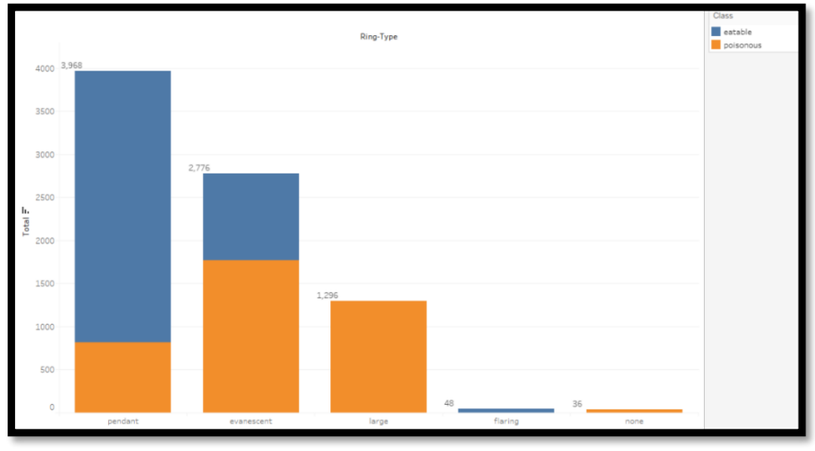

Stalk-Color-Above-Ring
Class
eatable
poisonous
Total F..
4500
4000
3500
3000
2500
2000
1500
1000
500
0
4,464
1,872
576
448
432
192
96
36
8
white
pink
gray
brown
buff
orange
red
cinnamon
yellow

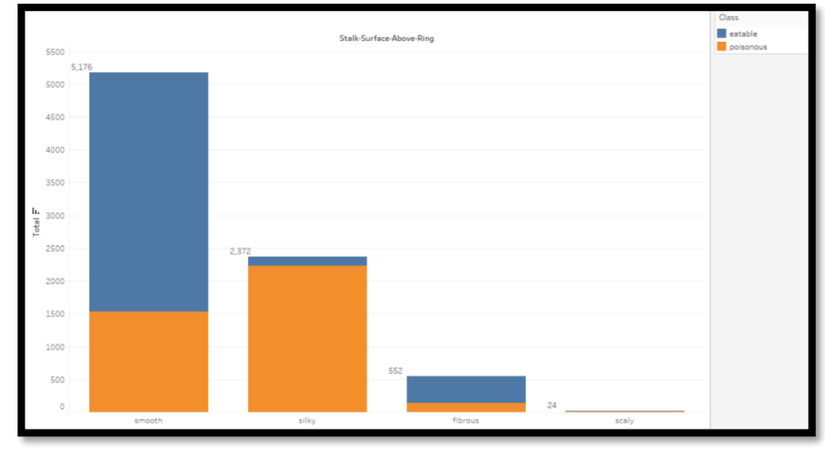

Spore-Print-Color
Class
eatable
poisonous
Total F..
2400
2200
2000
1800
1600
1400
1200
1000
800
600
400
200
0
2,388
1,968
1,872
1,632
72
48
48
48
48
white
brown
black
chocolate
green
yellow
purple
orange
buff

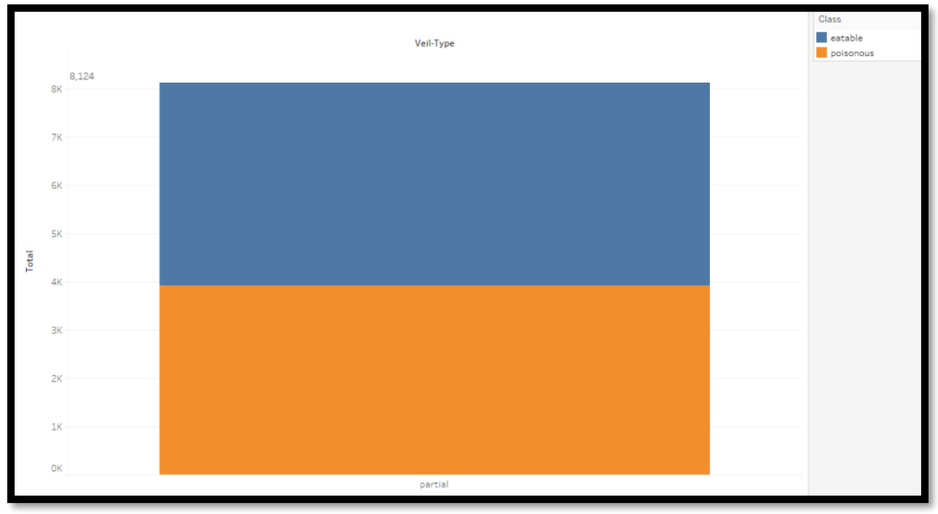

Veil-Color
Class
eatable
poisonous
Total F..
8K
7K
6K
5K
4K
3K
2K
1K
0K
7,924
96
96
8
white
orange
brown
yellow


Population
Class
eatable
poisonous
Total F.
4,040
1,712
1,248
400
384
340
several
solitary
scattered
numerous
abundant
clustered


Stalk-Surface-Below-Ring
Class
eatable
poisonous
Total F.
4,936
2,304
600
284
smooth
silky
fibrous
scaly


Ring-Type
Class
eatable
poisonous
Total F.
3,968
2,776
1,296
48
36
pendant
evanescent
large
flaring
none


Stalk-Surface-Above-Ring
Class
eatable
poisonous
Total F.
5,176
2,372
552
24
smooth
silky
fibrous
scaly


Veil-Type
Class
eatable
poisonous
8,124
8K
7K
6K
5K
4K
3K
2K
1K
0K
Total
partial

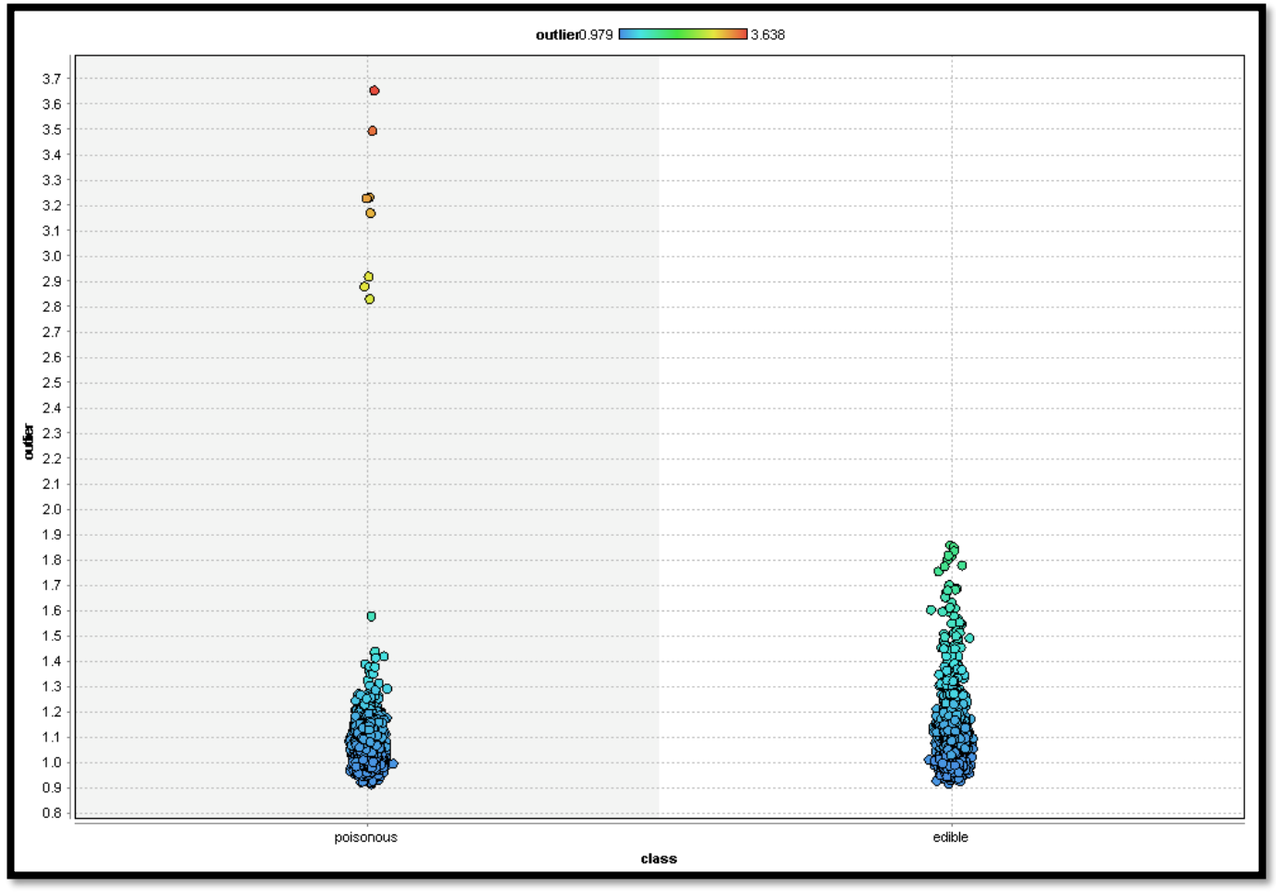
2.3: Analysis of the features.

ID	Features	Analysis
1	class	The class label is binominal: *Edible* and *Poisonous*. The feature is slightly imbalanced, with poisonous being the minority class. The difference between one class and the other one is 3.58%.
2	cap-shape	The feature is comprised of 6 different classes and is quite unbalanced. For instance, "Convex", the majority class, represents 45% of the dataset vs Conical represents just 0.0004%. However, the first two features (Convex and Flat) reach over 80% of all data points. The *Poisonous* and *Edible* classes seem to be evenly distributed among the shapes too.
3	cap-surface	The datapoints of this feature are more evenly distributed between three major classes: Scaly, Smooth, and Fibrous. The exception is "Grooves" which represents about 0.0004% of the dataset. Scaly is about 40%, while Smooth and Fibrous are about 30% each. There seems to be an even distribution of *Edible* and *Poisonous* class, especially in Scaly and Smooth. However, there appears to be a slight tendency towards the *Edible* class in the Fibrous category.
4	cap-color	Cap colour is comprised of 10 different classes. There are some indicators that say that if the cap colour is purple or green, therefore, the mushroom must be edible. The other categories are less evident as the *Edible* and *Poisonous* classes seem to be fairly distributed among the different colours. Nevertheless, it could be said that when the cap colour is white, then the mushroom is more likely to be edible. On the contrary, when the colour is red or yellow, the tendency is to be *Poisonous*.
5	bruises	Binominal feature with "Bruises" class and "No" class. As per data, it can be affirmed that the majority of the mushrooms (58%) do not have bruises. Nevertheless, there is a clear signal that suggests that if a mushroom does not have a bruise, it is much more likely to be *Poisonous*. On the other hand, if a mushroom does have bruises, it is probably *Edible*.
6	odor	Odour is quite an interesting feature in this dataset. It is composed of 8 different smells or no smell at all. According to the data, 45% of the mushrooms do not have a smell, but more than 90% of the mushroom is edible. It is important to highlight that there is distinguished segmentation of the mushrooms along with all the other categories. For instance, if a mushroom smells Foul, Spice, Fishy, Pungent, Creosote or Musty, then the probability of being *Poisonous* is 100%. These last 6 categories represent 46% of the dataset. On the other hand, if a data point is classified as Anise or Almond, then it is 100% probable to be *Edible*. This feature has a strong correlation with the targeted label as laid out in sections 2.5 and 3.5.
7	gill-attachment	Binominal attribute composed of Free and Attached categories. This attribute is highly imbalanced. Over 97% of the dataset is represented in the "Free" class, out of which the *Edible* and *Poisonous* class are about 50/50. Edible mushrooms also represent over 80% of the dataset within the "Attached" class.

ID	Features	Analysis
8	gill-spacing	Binominal attribute composed of Close and Crowded. Over 83% of the dataset is represented in the "Close" class, out of which the *Edible* and *Poisonous* class are about 50/50. Edible mushrooms represent also over 85% of the dataset in the "Crowded" class.
9	gill-size	Binominal feature composed of Broad and Narrow. This attribute is imbalanced as over 69% of the dataset is represented in the "Broad". However, within the Broad class, *Edible* is 70% of the dataset. Additionally, it is also observed that a mushroom is about 90% more likely to be *Poisonous* when it belongs to the "Narrow" category.
10	gill-color	This feature has 10 different categories. The dataset is quite unbalanced as the difference between the 1st mode and the least repeated value is more than 98%. Yet, there are quite very important insights out of this feature. For instance, Orange and Red would classify a mushroom as 100% *Edible*, but they together represent just less than 2% of the data. Nevertheless, if the colour is Buff or Green, it is 100% possible that they are *Poisonous*, representing 21.5% of the dataset. Additionally, there is a tendency towards the *Edible* class when the gill colour is White or Brown, but *Poisonous* when the colour is Grey or Chocolate.
11	stalk-shape	Binomial feature with "Tapering" and "Enlarging" classes. The feature is unbalanced as Tapering is the majority class with 57% of the dataset. The segmentation by labels seems to be distributed, especially in Enlarging, although there is a slight trend towards the *Edible* class in the Tapering category.
12	stalk-root	This polynomial attribute is of interest to the modelling process. About 31% of the values are missing in this column, with a proportion of about 2.44 *Poisonous* : 1 *Edible*. The class that repeats the most is Bulbous (46%) out of which the *Poisonous/Edible* relationship is almost 50/50. Within the Equal and Club categories, the trend is towards the Edible class, while Rooted has 100% of being *Edible*.
13	stalk-surface-above-ring	This attribute has 4 classes, the most dominant one is "Smooth" representing almost 64% of records. The ability of this feature to predict edibility of mushrooms is limited as all 4 classes have records that can have both labels: edible and poisonous. However, mushrooms with "silky" stalk surface tend to be poisonous and on the other hand records with class "smooth" are mostly *Edible.* edible.
14	stalk-surface-below-ring	Stalk surface below the ring is a feature that is highly correlated with Stalk surface above the ring attribute (shown in section 2.5). It also has the same 4 classes: "smooth" (61% of records), "silky" (28%), "scaly" (4%) and "fibrous" (7%). It does not provide a clear cut between edible and poisonous mushrooms; however, similar as for previous attribute, records in class "smooth" are more likely to be edible (69% of cases) and "silky" ones are more likely to be "poisonous" (94% of cases).
15	stalk-color-below-ring	Stalk colour below ring is comprised of 9 different classes, with colour "white" covering 54% of records and "pink" 23%. For 6 classes datapoints clearly determine edibility as "grey", "orange" and "red" stalk colour below ring mean edible mushroom and "buff", "cinnamon" and "yellow" poisonous.

ID	Features	Analysis
16	stalk-color-above-ring	This feature is highly correlated with stalk colour below ring attribute, it also has 9 different classes, and the distribution of classes is similar, with “white” colour representing almost 55% of records. Same as for stalk root below ring feature, classes “grey”, “orange” and “red” are 100% edible and “buff”, “cinnamon” and “yellow” are poisonous.
17	veil-type	Veil type has cardinality 1, all records have label “partial”. Thus,Therefore, it does not provide any information that can be helpful in classification exercise.
18	veil-color	Veil colour has cardinality 4, however class “white” is very dominant, with more than 97% of records. Observations in this class might be both *Edible* and *Poisonous* with distribution between two labels being roughly equal. On the other hand, the 3 small classes provide clear information about edibility, brown and orange representing edible mushrooms and yellow representing poisonous ones.
19	ring-number	Ring number is highly imbalanced feature as over 92% of observations are in class “one”. Ability of this feature to determine edibility of mushrooms is limited as records with ring number “one” have approximately same distribution of edible and Poisonous labels. Beneficial for classification model can be mainly class “none” (<0.5%) for which label is always *Poisonous*. Only Less than half percent of records are in class “none”.
20	ring-type	Datapoints of this feature have in our dataset classes “pendant” (49% of records), evanescent (34%), “large” (16%), “flaring” (<1%) and “none” (<1%). Except class large, for which class is always poisonous, ring type does not provide much more useful information to distinguish edibility.
21	spore-print-color	Spore print colour feature has cardinality of 9 and in models applied on dataset proved to be one of the most important features for classification of edibility of mushrooms. Classes, “buff”, “orange”, “purple” and “yellow” signify edible mushrooms, “green” ones are always poisonous. Records with classes “black”, “brown”, “chocolate” and “white” can have both labels edible and poisonous.
22	population	Population is represented by 6 different classes in dataset with class “several” being highly overrepresent (almost half of the records). Unlike “abundant”, “clustered” and “numerous” as each of them is present only for 4-5% of datapoints. This feature however does not show high significance for distinguishing between *Edible* and *Poisonous* mushrooms.
23	habitat	Habitat feature is comprised of 7 classes. The distribution of classes is somehow imbalanced as big proportion of data is represented by woods (39%) and grasses (26%) and on the other end meadows, urban and waste represent each only between 2-5% of data. However, this feature did not prove to be important in establishing edibility of mushroom.

2.4: Identifying outliers.



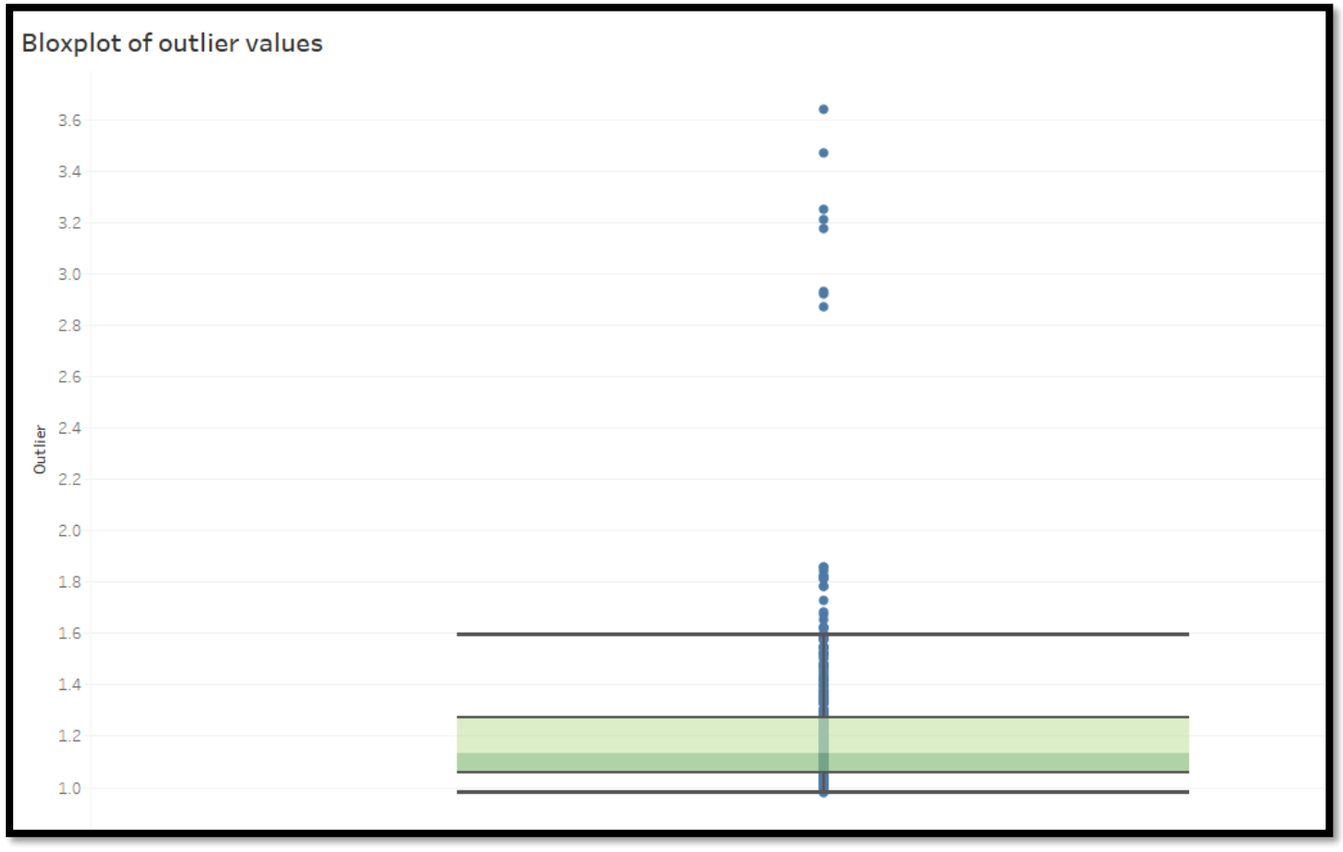
Method: Local Outlier Factor (LOF).

Description: According to RapidMiner (2021), the LOF operator uses k-nearest neighbour approach to identify the density of an object. In general terms, the higher the score value, the more likely the datapoint is an outlier.

Hyperparameters used:

1. Minimal points lower bound: 10
2. Minimal point higher bound: 20
3. Distance: Euclidean distance.

Analysis: After applying the Local Outlier Factor (LOF), it was possible to identify 8 outliers in the dataset. These outliers correspond to the Poisonous class in the dataset. As seen from the graph, the majority of the datapoints have an outlier score below 1.9, while these 8 datapoints have a score above 2.8, with the most eccentric point having a 3.64 score value. These points have formed a sub-cluster that is clearly defined.



Method: Boxplot in Tableau

Description: Boxplot graph created in Tableau with outlier values extracted from the Local Outlier Factor (LOF) operators in RapidMiner.

Hyperparameters used: 1.5 * IQR.

Analysis: The values of the boxplot can be found below:

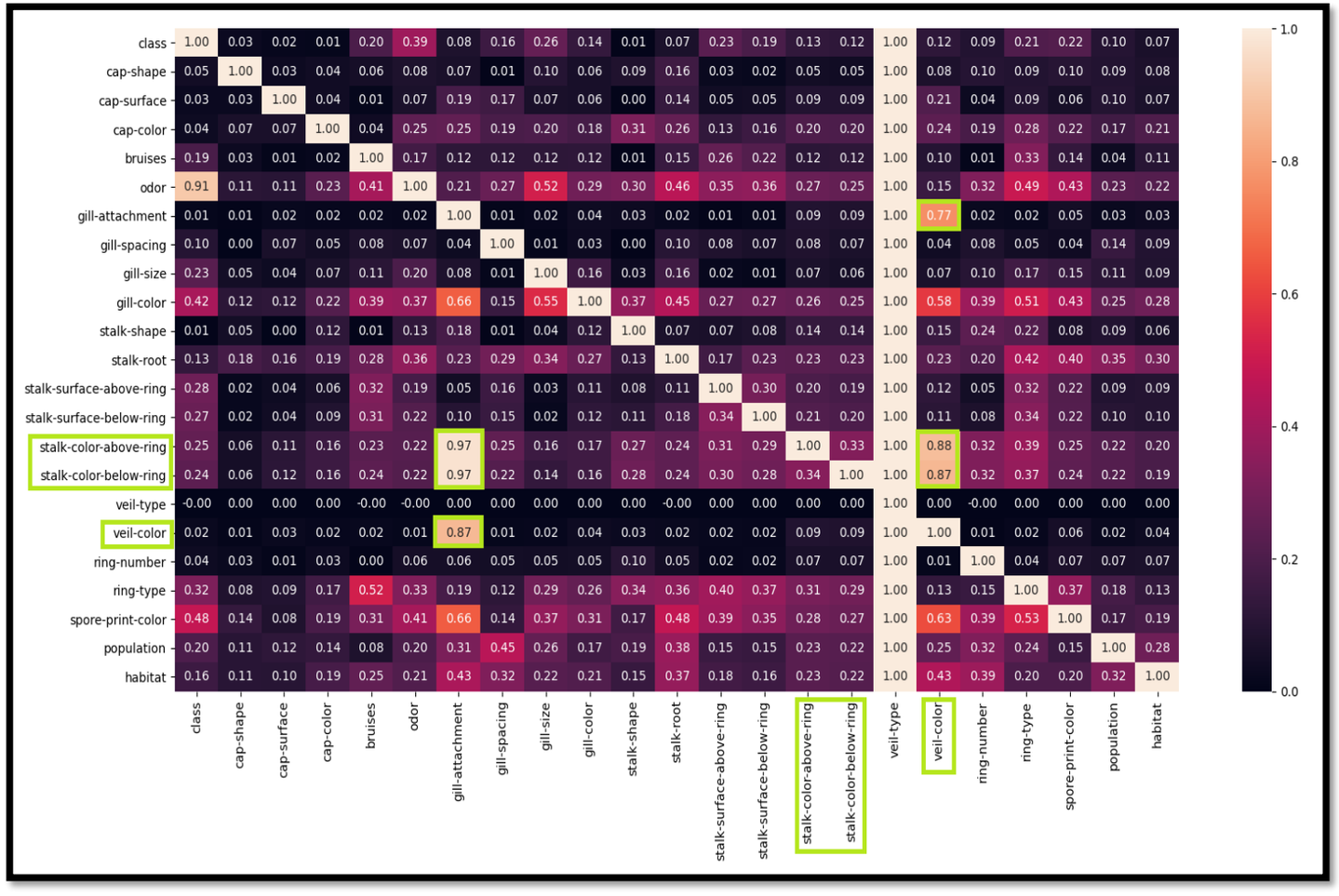
1. Upper whisker: 1.593
2. Higher hinge: 1.273
3. Median: 1.130
4. Lower hinge: 1.057
5. Lower whisker: 0.979

Therefore, we can calculate the number of outliers by type:

a) Mild outliers outside 1.5* IQR: 25 datapoints
b) Extreme outliers outside 3 * IQR: 8 datapoints.

It is concluded that the number of outliers would vary depending on the factor selected for the IQR. However, the 8 datapoints previously mentioned are considered extreme outliers.

2.5: Identifying Correlation Between Features.



Method: Theil's Uncertainty Coefficient

Description: According to (Press, et al., 2007) Theil's Uncertainty Coefficient is derived from informational entropy and describes the fraction that a given Y can be predicted by X. The coefficient is calculated via the following relation:

$$U(X|Y) = \frac{H(X) - H(X|Y)}{H(X)}$$

where the entropies H is given by

$$H(X) = -\sum_{X} P_X(x) log\ P_X(x)$$

$$H(X|Y) = -\sum_{X} P_{X,Y}(x,y) log\ P_{X,Y}(x|y)$$

Where $P_X(x)$ is the probability of x occurring in the dataset.

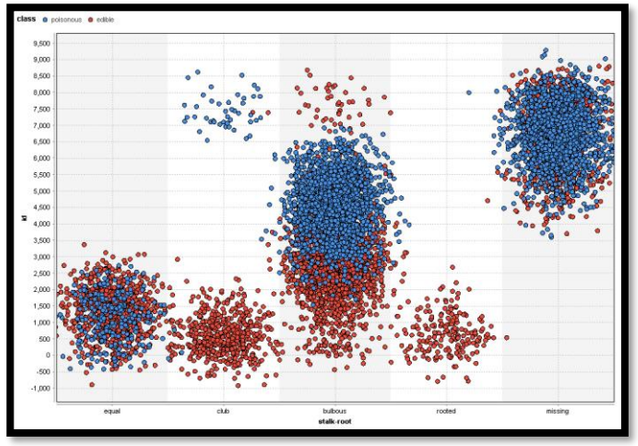
The coefficient ranges between 0 and 1, where 0 has high entropy and no correlation and 1 has low entropy and high correlation.

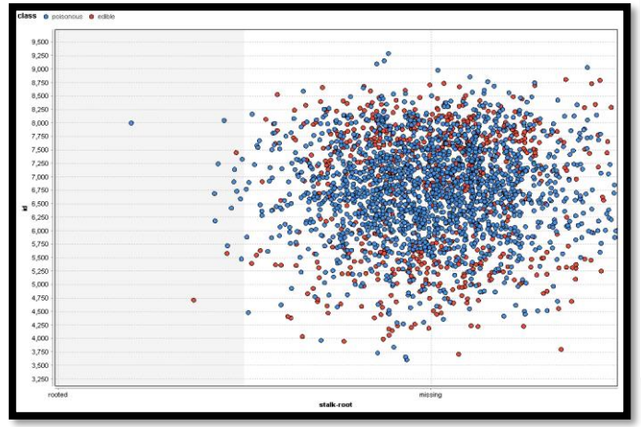
Analysis:

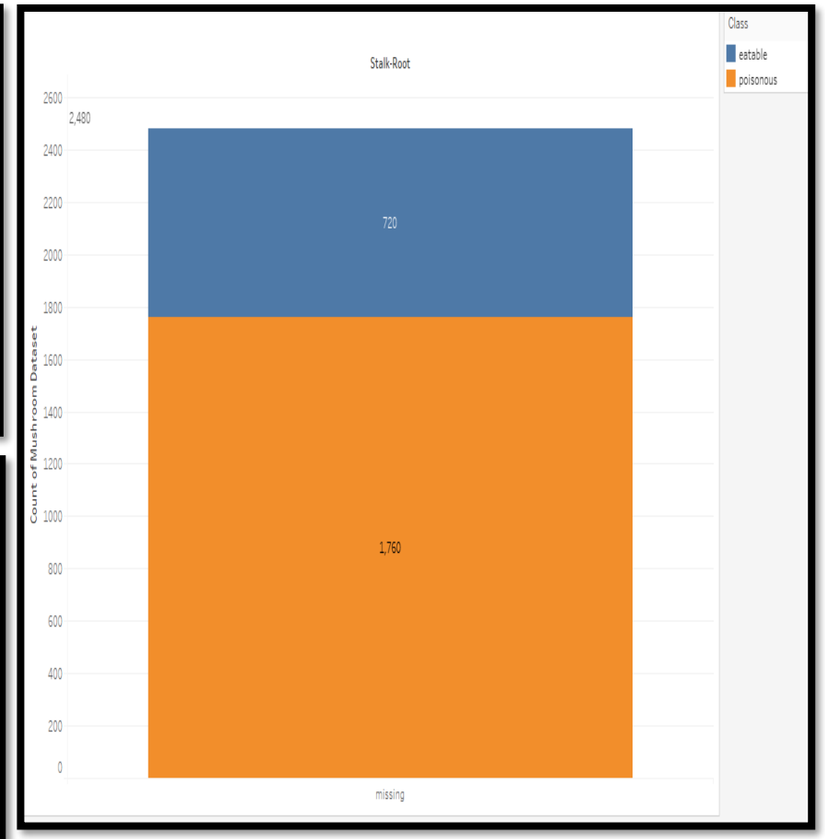
Using the uncertainty coefficient, a matrix plot of the features X against features Y was produced (left). Significant Intra-feature association was found between stalk-color-above-ring, stalk-color-below-ring, veil-color with gill-attachment. The threshold chosen for significance was U >= 0.7, similar to other correlation coefficients.

Furthermore, this analysis revealed that mushroom odor is feature most associated with the class. Class. However, was not significantly associated with odor, implying a one-way relation. Later in modelling, odor was found to be the most important feature.

2.6: Identifying missing values.







Method: Counting.

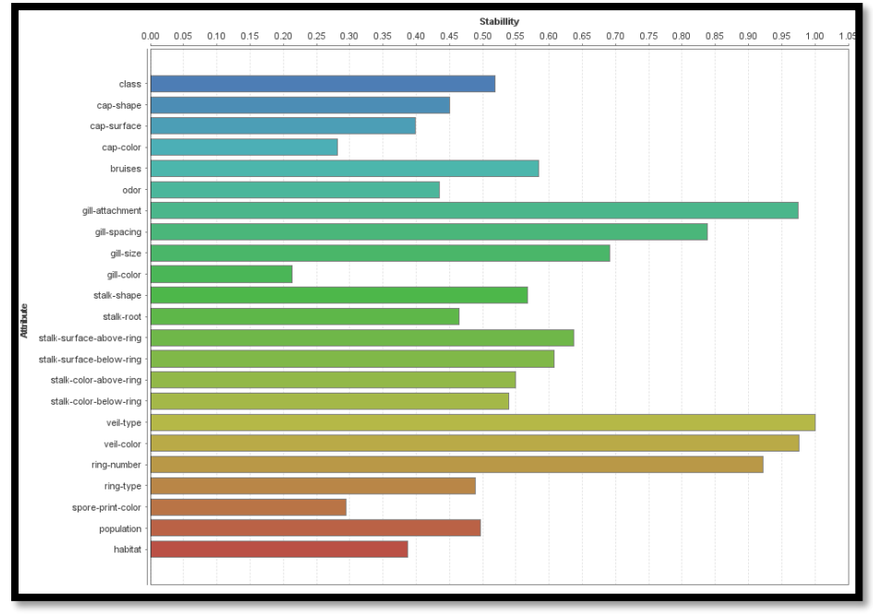
Description: Counting missing values in the dataset per feature and filtering by predicted label.

Hyperparameters: n/a

Analysis: After analysing all the features, it was possible to identify that *Stalk-root* is the only one that contains missing values, in total 2,480 nulls out of which 720 belongs to the *Edible* class (29%) and 1760 to *Poisonous* class (71%). This deviation has a proportion of around 2.44:1.

2.7: Identifying stability score.

Feature	*Stability score*
veil-type	1
veil-colour	0.975382
gill-attachment	0.974151
ring-number	0.921713
gill-spacing	0.838503
gill-size	0.690793
stalk-surface-above-ring	0.637125
stalk-surface-below-ring	0.607582
bruises	0.584441
stalk-shape	0.567208
stalk-colour-above-ring	0.549483
stalk-colour-below-ring	0.539636
class	0.517971
population	0.497292
ring-type	0.488429
stalk-root	0.464796
cap-shape	0.450025
odor	0.434269
cap-surface	0.399311
habitat	0.387494
spore-print-color	0.293944
cap-color	0.281142
gill-color	0.212703



Method: Quality measure Operator in RapidMiner

Description: The Quality measure operator was applied on the dataset and the stability score was extracted from the output. According to co-founder of RapidMiner, Dr. Mierswa (2019), the value is calculated based on the following formula:

$$stability = \frac{\sum m}{\sum r}$$

Where:

a) m: Mode in the attribute.

b) r: total number of rows in the attribute without missing values.

The score varies from 0 to 1. Desired threshold < 0.9 (or 90%).

Hyperparameters: n/a

Analysis: As per the output obtained, there great majority of the dataset has a good stability score value, with only 4 features above 0.9:

a) Gill-attachment
b) Veil type
c) Veil colour
d) Ring number

2.8: Identifying data quality issues.

Table 3. Data quality issues identified in the dataset.

Feature name	***Data quality issue***	***Potential handling strategy***
Class	Imbalanced classes	Upsampling the minority class (poisonous).
stalk-root	30% of missing values	Drop the feature from the dataset
veil-type	Single value feature, highly stable	Drop the feature from the dataset
Class	Outliers	Remove the eight (8) outliers from the dataset
Stalk colour above the ring	Highly related data	Drop the feature from the dataset
Stalk colour below the ring	Highly related data	Drop the feature from the dataset
Veil colour	Highly related data, highly stable	Drop the feature from the dataset
Gill-attachment	Highly stable	Drop the feature from the dataset
Ring number	Highly stable	Drop the feature from the dataset
All features	Classes' names	Rename all the classes with a meaningful value

Note: No duplicates were found in the dataset. Thus, this quality issue was not considered during the analysis or post data processing steps.

3: Data Preparation

3.1: Upsampling minority class.

As it was shown in the data understanding section, the labels (Poisonous vs Edible) are slightly unbalanced. Since the main goal of the project is to be able to predict the classes based on a series of attributes, this situation represents a problem. According to Kotu & Bala (2019), when a classification model is trained in an unbalanced dataset, the resulting class label recall will also be skewed. In other words, the classification model created will be biased towards the majority class. This issue means that the model will be better at predicting one class over others. Consequently, the dataset must be balanced in order to apply a classification model.

The technique chosen to complete such tasks was Synthetic Minority Oversampling Technique (SMOTE) as it is described by Chawla, et al. The technique chosen to complete such tasks was Synthetic Minority Oversampling Technique (SMOTE), as it is described by Chawla, et al. (2002) as a great method to generate artificial but similar data. The technique accomplishes this process by applying k-nearest-neighbour algorithms in the dataset which selects samples of the minority class, finds the closes data points, and calculates the distances from them. The technique accomplishes this process by applying k-nearest-neighbour algorithms in the dataset, which selects samples of the minority class, finds the closes data points, and calculates the distances from them. The authors also mentioned that combining undersampling of the majority class and then applying SMOTE to the minority class performs better than applying plain undersampling. Nevertheless, as the difference between one class and the other one was just 3.58%, it was decided to upsample the minority class directly with the hyperparameter of K = 5. As a result, the minority class was balanced to match the majority class: 4208 data points each.

3.2: Removing missing values.

Missing values is defined as data value that is not stored in the dataset (Hyun, 2013). This is a recurrent problem in all areas of data analytics and can have a significant impact on the deployment of models and analysis of results. For instance, it reduces statistical power, can cause bias, and can affect the representativeness of some classes. Therefore, managing them promptly and adequately is a vital part of data preparation. One thing to consider when dealing with null values is the relationship with the predicted label. The best cases scenario is to have “Missing values completely at random” (or MCAR) as this has little effect in making a predictive model with a predisposition towards one of the labels. However,

this is not always the case, known as “Missing values not at random” (or MNAR), and some extra measure might be required (Hyun, 2013).

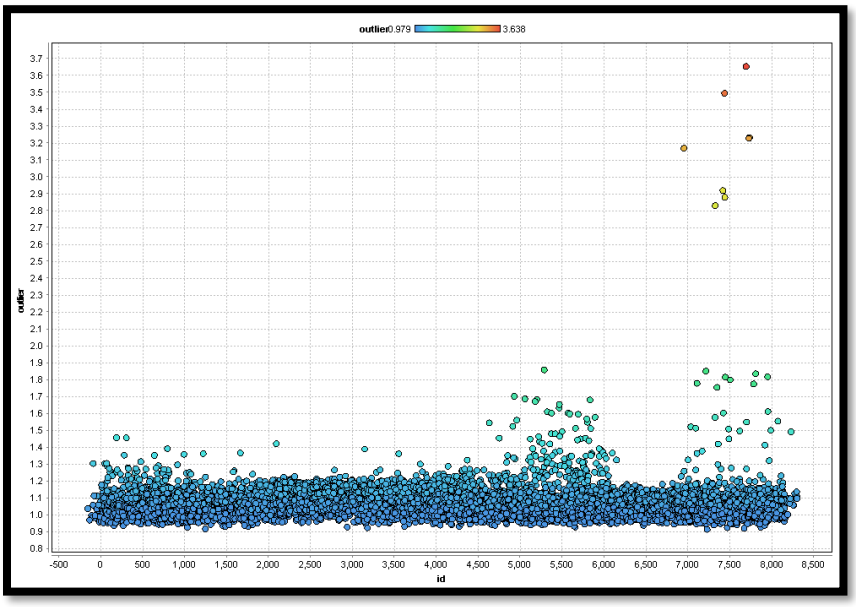
Additionally, Hyun (2013) affirms that one common approach to sort the missing values is by applying Listwise deletion of those rows. However, if the dataset does not satisfy the MCAR assumption, then it can produce a biased result. Moreover, Sharma (2018) adds that when a column contains more than 30% of null values, then it should be dropped altogether.

Therefore, considering that the proportion of the missing values per class is 2.44 *Poisonous* vs 1 *Edible* and the percentage is of null values in the feature is 30.53%, it was decided to remove it from the analysis as deleting the rows could lead to an unbalanced dataset. This could influence the statistical power of the model, but it is the best approach to minimise bias.

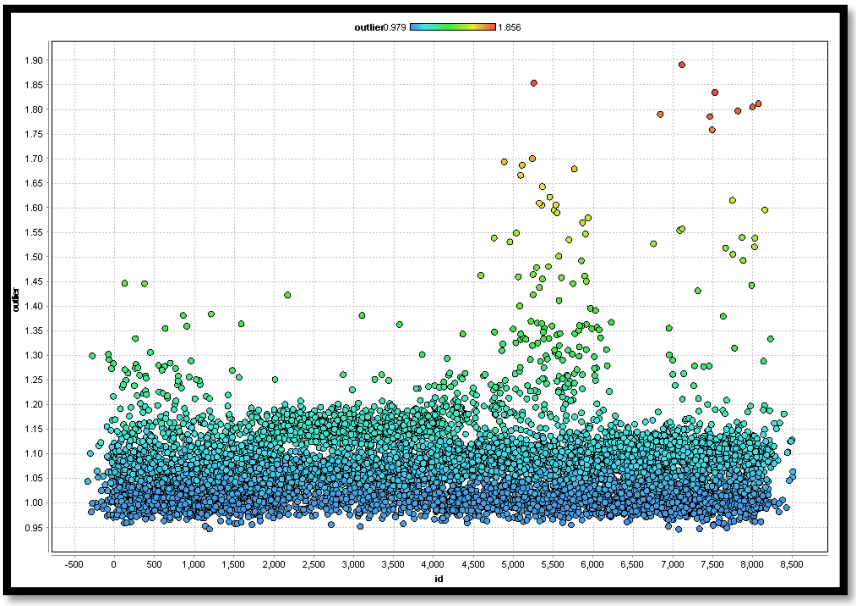
3.3: Removing outliers.

According to Kotu & Bala (2019), the finding of outliers is part of the anomaly detection process that must be carried out on the dataset. The authors affirm that these objects stand out among other data points and do not follow the normal pattern of the dataset. It is also known that outliers tend to form a different cluster of values because they are too far away from the main clusters. The cause of having outliers in the dataset can be difficult, if not impossible, to know with exactitude, as these ones can be due to the wrong collection of data or data with high variability. Sometimes, outliers are the values of interests because we are concerned in what is outside a normal distribution, and other times, these values must be filtered out to avoid having a skewed dataset, which can affect the performance of the model. Nevertheless, the authors warn that without knowing the real reason for these outliers to occur in the dataset, it is risky to remove the data points as one could be deleting actual variance in the population (Kotu & Bala, 2019).

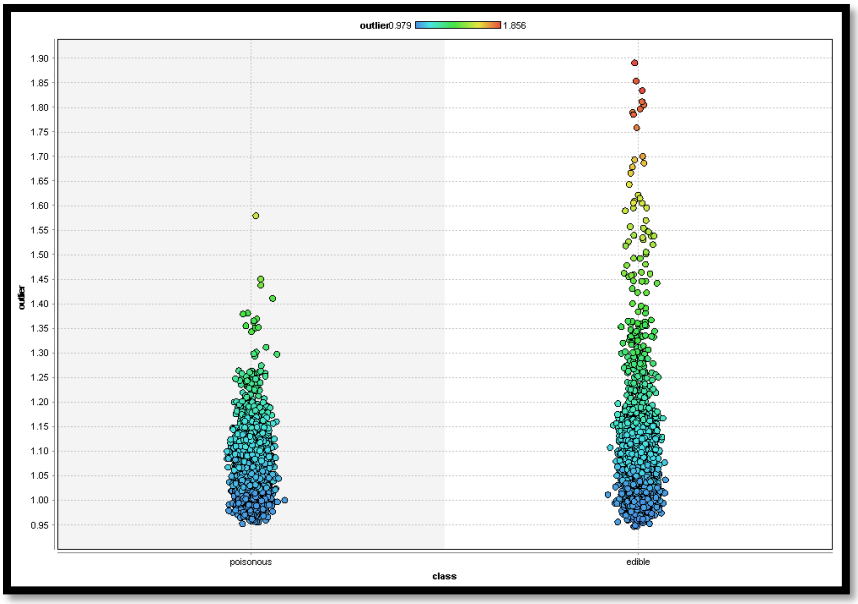
Considering the above statements, it was agreed that only the extreme values would be removed from the dataset in order to maintain variance. As it was revealed before, it was possible to detect a total of 8 extreme values with an outlier score above 1.921. As it was explained, these 8 outliers have formed a sub-cluster that does not follow the majority of the classes. Keeping in mind that we are trying to build a model that will classify whether a mushroom is poisonous or edible, we are not interested in predicting this cluster of outliers. As a result, they have been filtered out to ensure that the most accurate outcome is obtained.



Plot of outlier scores before removing extreme values.



Plot of outlier score after removing extreme values.



Plot of outlier score by class after removing extreme outliers.

3.4: Removing features with single values.

As it was described in the ABT table and association matrix, there is one feature with a unique value "*veil-type"* which needs to be removed from the dataset. As the column is a zero variance predictor with a unique value, it is highly correlated to all the features and does not apport any additional information to the classification model.

3.5: Removing correlated features.

As described in part 4 of Data exploration. The columns with the highest intra-feature correlation were removed from the dataset. Theil's uncertainty coefficient was chosen as the measure of correlation for the following reasons.

1. It allows for the correlation of categorical features to be calculated using mutual information and conditional entropy.
2. Unlike Chi Squared Independence tests, we can get a measure of how strong the correlation is on an absolute scale.
3. Cramér's V coefficient is another measure of nominal association that was considered; however, Theil's U coefficient has the advantage of being asymmetric in measuring correlation, whereas Cramér's V is symmetric (like Pearson's R). A symmetric association loses more information overall than an asymmetric association.

The coefficient ranges between 0 and 1, with the value 0 indicating that x and y have no association and the value 1 indicating that knowledge of x completely predicts y. This also follows true for y predicting x and crucially, the correlation coefficient for both cases do not equal and are thus asymmetric (Press, et al., 2007). Similarly to the Pearson's R coefficient, the conventional threshold for significance is 0.7, where values above are said to be highly correlated.

3.6: Removing highly stable features.

According to Mierswa (2019), highly stable features are defined as columns with practically constant values. A feature is considered highly stable when the first mode of the feature represents more than 90% of the values, and therefore, it does not apport enough variability in the dataset. Therefore, they can be treated as columns with single values. Doctor Mierswa also affirms that these columns should not be included in the classification models.

Subsequently, based on the considerations above, the features *ring number* and *gill attachment* were dropped from the dataset as their stability score is above 90%. Another feature very close to the threshold was *gill spacing* with 84%, but it was decided to keep it as per recommendation.

3.7: Renaming features' values.

Since the main goal of the project is to build a model that can predict whether a mushroom is poisonous or edible, it is important that model generates interpretable results so that they can be applied in the real world. As such, and for convenience, a script in Excel was run on the original dataset, which replaced all the coding variables for a more meaningful value. For instance, the feature "Class" had two classes: P and E, which were replaced by Poisonous and Edible, respectively. This process was carried out on all the features.

4: Modelling

Based on the problem and type of dataset associated with it, it was agreed to use three different models, which are part of an ensemble model to predict the classification: Decision Trees, Rule Induction and Naïve Bayesian. A brief description of the model and parameters used can be found below:

4.1: Decision Tree model

Model Description	According to Kotu & Deshpande (2019), decision trees algorithms are among the most common and widely used algorithms in data science. One of the significant points is their interpretability and the trim work the data preparation step requires before implementing them. The authors explain that this model can be executed when the targeted label is categorical in nature and which can have more than two classes. However, whenever the targeted label is numerical, the decision tree will work as a regression model, called Regression Trees. Decision trees create a flowchart of the relationships of the features. It tests each class at the nodes and computes mathematical operations to calculate the homogeneity of the feature — the more homogenous, the better for the split. In other words, the uncertainty is reduced. A common approach to this is by calculating Information gain, which is calculated based on the classes of a feature and the relationship with the targeted label. In general, the higher the information gain value, the better, and the feature with the highest value will be used to start the split the tree, the process is repeated until the information gain is zero or a threshold is set to stop the growing, also called as “pruning”, alternatively, gain ration, a variation of information gain, can be used to prevent bias in the calculations, especially when the cardinality of a feature is very high. (Kotu & Bala, 2019).
Model Parameters	Criterion: Gain Ratio Max depth: 10 Confidence: 0.1 Minimal gain: 0.01 Minimal leaf size: 2 Minimal size for split: 4

Input	Decision trees are quite flexible at data types for the input. They can work with both numerical and categorical values. They are good at dealing with outliers as well as missing values. The targeted feature in the training set must be balanced, otherwise the tree will learn to predict better one class over the other ones. Upsampling the minority class is often a good solution (Kotu & Bala, 2019).
Output	The result would be a model that split the data at nodes, each node has a likelihood, and it is based on the information gain previously calculated for each feature. Once the model is presented unseen data, it will run test the dataset at each node and make a prediction which then will be compared to the actual label an calculate accuracy scores (Moner, 2019).
Pros	Decision trees have some significant advantages over other classification models. Among the pros, one could list: a) They are pretty easy to understand and visualise, this means that even non-experts in data science could understand the rationale and meaning of the tree; b) Decision trees intuitively perform feature selection as only the most important features will be taken into account for its constructions; c) flexibility in their data input, as it accepts both numerical and categorical data types, it is also good at handling missing values and outliers; and d) Little effort in data preparation (Gupta, 2017).
Cons	Decision trees present, however, a few disadvantages too. For instance, they tend to overfit the data and therefore, they can be poor predictors if this is not controlled. If a tree is allowed to grow fully, it can get very complex, and it loses its interpretability. They can also be very volatile, meaning that small changes in the training set can lead to entirely different decision trees with different results (Moner, 2019).

4.2: Naïve Bayesian Model

Model Description	This model is also knowing as the conditional probability model as it is based on the Bayes theorem for classification purposes. The main characteristic of this model is that it assumes independence of all the features, making the process much simpler as it changes the calculations from dependent

	conditional probability to an independent conditional probability. The general equation for the calculation follows: $$P\ (y|x) = \frac{P(x|y) * P(y)}{P(x)}$$
Model Parameters	Laplace correction will assign a very small probability to missing values in the test set, if any (Kotu & Bala, 2019).
Input	The naïve Bayesian model requires categorical data for the features. When some numerical features are present, these ones will need to be transformed into categorical one. A very straightforward method is setting up some threshold for a range of values, if true, the new label is assigned. This process, however, can lead to loss of information. Alternatively, it is possible to use a normal distribution function for the calculation of the probability densities (Kotu & Bala, 2019).
Output	The output of the model is a learned table of probabilities that can be easily retrieved to calculate the likelihood of a set of values being classified.
Pros	It is quite robust against outliers and missing values, and it is good at solving multi-class predictions. When the assumption of independence holds true, it can perform excellently. It is easy to set, and once the rules are learned, unseen data can be classified easily. (Brownlee, 2019). It works well with high dimensionality problems, like text mining (Yıldırım, 2020).
Cons	One disadvantage of this model is its assumption of independence between attributes. One way to correct this problem is by creating a correlation matrix and identify the high-correlated features and drop them from the dataset. But this will also lead to lose of valuable information and reduces complexity of the model which can lead to underfitting. The performance of the model also decreases when the dependence of the features increases. It can lead to lose of information when numerical date must be transformed into categorical values (Brownlee, 2019).

4.3: Rule induction model

Model Description	Rule induction is an algorithm that leverage the relationships from the attributes and returns a set of rules. When regularities can be found in the data, these ones can be expressed as a rule in the form of expressions: If (attribute 1, value) – Then (class, value) (Grzymala-Busse, 2005). There are different methods to leverage the rules out of the relationships of the dataset. One of them is called "Rule Induction" which extracts the rules out the dataset directly, and Tree to Rules, which builds a decision tree to extract the rules; this method is considered indirect (Kotu & Bala, 2019).
Model Parameters	For the purpose of this project, "Rule Induction" method was selected. Criterion: Information gain Sample ratio: 0.9 pureness: 0.9 Minimal prune benefit: 0.25
Input	All attributes are independent variables, while the label must be a dependent variable. Values can be either categorical or numerical (Grzymala-Busse, 2005).
Output	Set of rules to classify a set of values into a categorical label (Kotu & Bala, 2019).
Pros	One of the advantages of the model is its interpretability. The set of rules are easily explainable to non-technical users. Therefore, it is not only a predictive model but also a descriptive one (Kotu & Bala, 2019).
Cons	It is a greedy algorithm that tends to overlearn the training set; hence, the results obtained from the model could not be globally applicable to all scenarios (Kotu & Bala, 2019).

4.4: Ensemble Vote Model

Model Description	The three models above have been combined into an ensemble model. These models employ multiple base models to predict a result. The reason to do that is to reduce the generalised error of each individual model build (Kotu & Bala, 2019). However, in terms of this project, it was decided to use an ensemble model to show how other algorithms would perform in the dataset.
Model Parameters	Decision Tree Base Model Naïve Bayesian Base Model Rule Induction Base Model
Input	1. Categorical variables 2. Balanced data: Minority class up-sampled through SMOTE process. 3. Feature selection: Highly correlated data was removed. 4. Outliers removed: Extreme values were removed.
Output	Generalised model with vote process (majority vote).

5: Evaluation

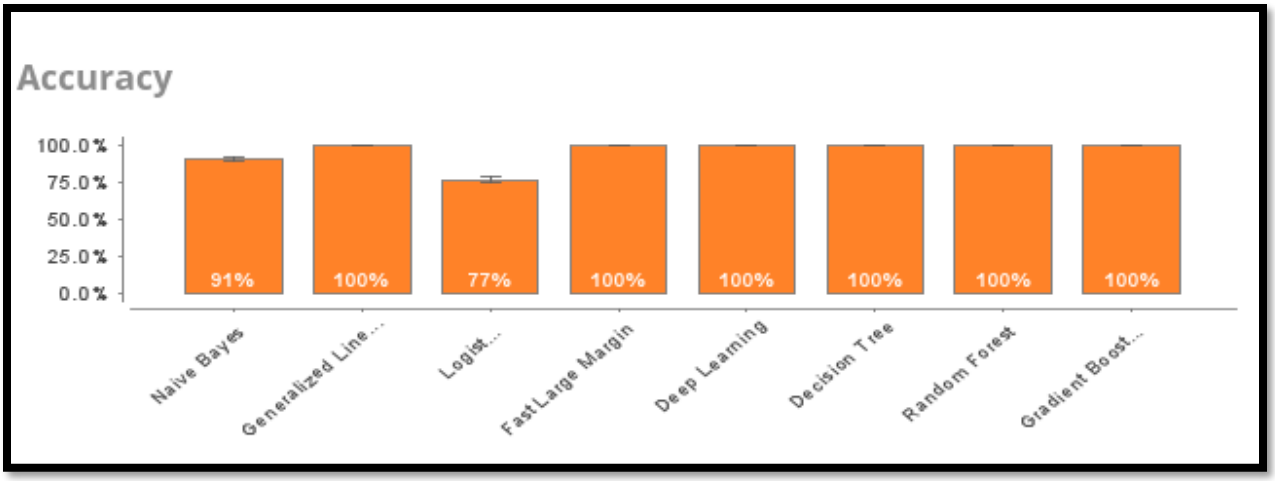
5.1: Pre-evaluation (Auto-model).

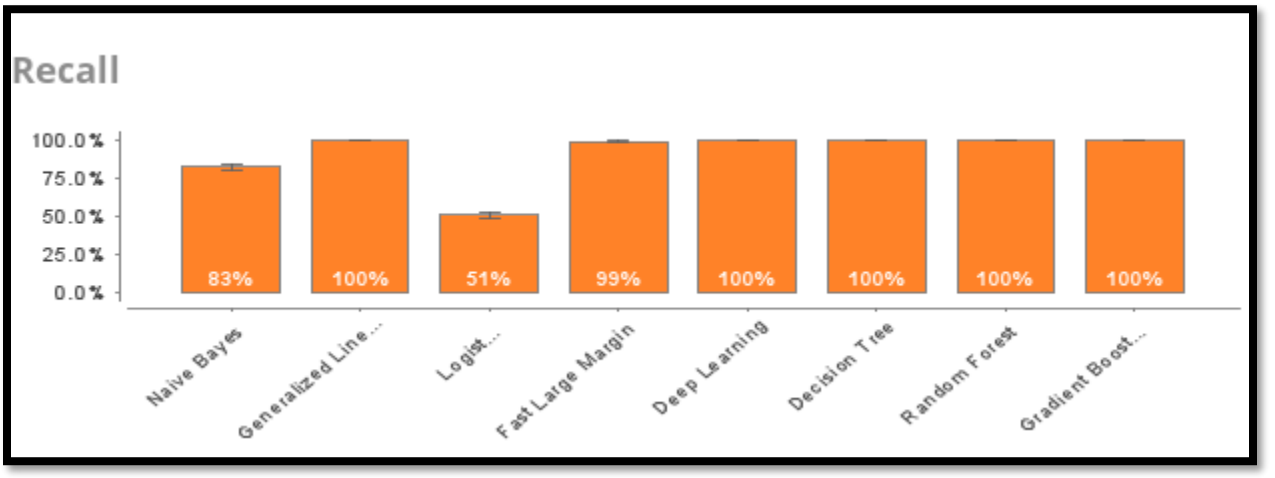
Before selecting the algorithms to be used in the modelling, the raw data was run in the Auto-model process in RapidMiner for a pre-evaluation. What was found was that the majority of the models were throwing very high accuracy scores.

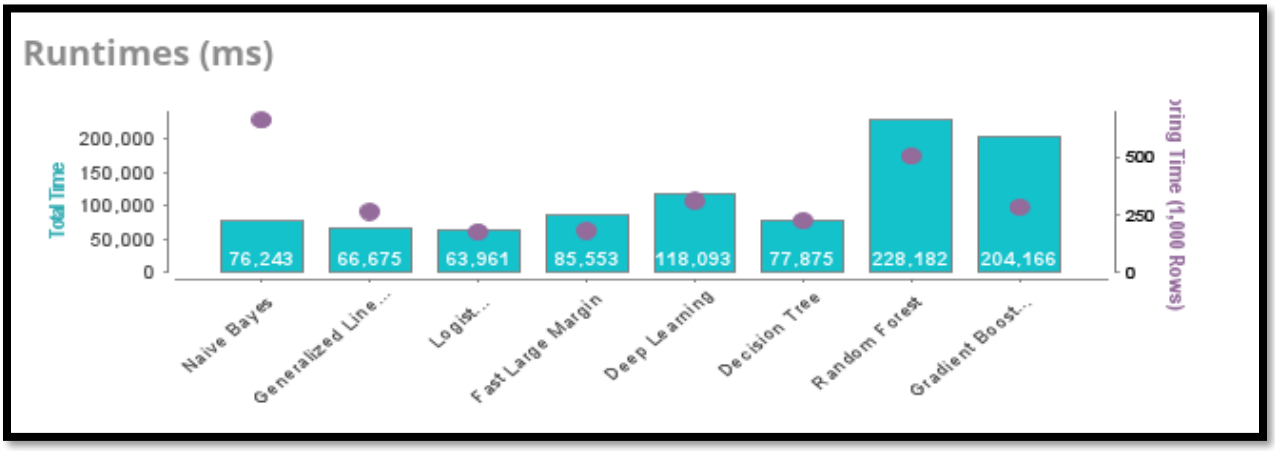
The parameters were set in such a way that the poisonous label was considered the positive class in the models. Regarding the accuracy value, almost all the models predicted with 100% accuracy, such as the decision tree model that had been capable of predicting with excellent accuracy level. The exceptions were Naïve Bayesian and Logistic Regression algorithms, with 91.29% and 76.65%, respectively. Thus, both models had a considerably high classification error, especially Logistic Regression with 23.35%.

Additionally, it is essential to highlight that the recall value varies significantly between the model, the Decision Tree, for instance, has 100% recall score. On the other hand, although the Naïve Bayesian model has an accuracy of 91.29%, the recall was just 83%, making a poor predictor of the positive class (poisonous). Moreover, the Logistic Regression did even worse, as it was able to correctly predict just 51% of the positive class. Also, the running time was taken into consideration, as some models with great accuracy were taking much more time to train to produce the same results, such as Random Forest and Gradient Boosting tree.

Based on the results obtained from the Auto-model evaluation, it was decided to select the Decision Tree model as it met perfect accuracy, recall and running time. In addition, for practical reasons, it was also agreed to choose the Naïve Bayesian model to test whether the algorithm was able to improve after data preparation. Finally, Rule induction, which was not part of Auto-model pre-evaluation, was also selected to assess if it is able to correctly classify the positive class based on the relationship of the features.







Comparison of the Auto Model results per model (Accuracy, Recall and Runtimes)

Table 4. Auto-model results for pre-evaluation of raw data.

Model	*Accuracy*	*Classification Error*	*Precision*	*Recall*	*Sensitivity*	*Specificity*	*Standard Deviation*	*Gains*	*Total Time*	*Training Time*	*Scoring Time*
Naive Bayes	91.29%	8.71%	99.13%	82.74%	82.74%	99.34%	1.11%	1844.0	76243.0	29.7	661.7
Generalised Linear Model	100.00%	0.00%	100.00%	100.00%	100.00%	100.00%	0.00%	2284.0	66675.0	69.9	263.5
Logistic Regression	76.65%	23.35%	100.00%	51.04%	51.04%	100.00%	2.00%	1128.0	63961.0	59.8	177.0
Fast Large Margin	99.74%	0.26%	100.00%	99.46%	99.46%	100.00%	0.18%	2180.0	85553.0	16.2	181.0
Deep Learning	100.00%	0.00%	100.00%	100.00%	100.00%	100.00%	0.00%	2284.0	118093.0	191.8	311.5
Decision Tree	99.96%	0.04%	100.00%	99.92%	99.92%	100.00%	0.10%	2246.0	77875.0	18.6	224.4
Random Forest	100.00%	0.00%	100.00%	100.00%	100.00%	100.00%	0.00%	2248.0	228182.0	28.1	504.2
Gradient Boosted Trees	100.00%	0.00%	100.00%	100.00%	100.00%	100.00%	0.00%	2284.0	204166.0	207.7	283.8

5.2: Model evaluation of model selected before cleaning the dataset.

Table 5. Pre evaluation scores for the models selected.

Results	*Decision Tree**	*Naïve Bayesian**	*Rule Induction***
Accuracy	100%	91.3%	97.78%
Classification Error	0.00%	8.70%	2.22%
Recall	99.90%	82.70%	96.68%
Precision	100%	99.15%	98.70%
F measure	100%	90.20%	97.68%
AUC	1.000	0.999	0.986
Sensitivity	99.90%	82.70%	96.68%
Specificity	100%	99.30%	98.81%
Training time	16 ms	79 ms	n/a

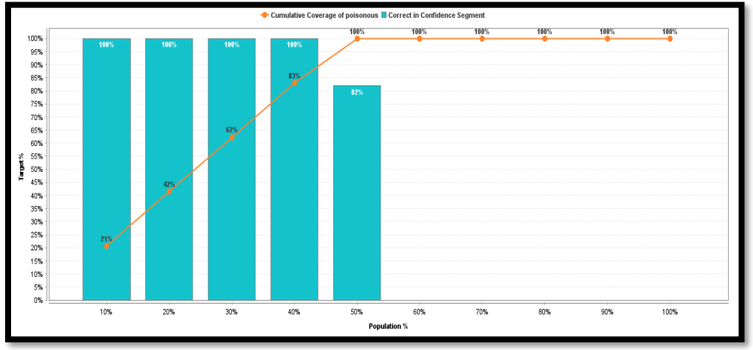
*Scores extracted from Auto-model evaluation from RapidMiner.
**Scores extracted from a process created for Rule Induction with raw data.

Three algorithms (Decision Tree, Naïve Bayesian, and Rule Induction) were firstly applied to the dataset before cleaning and other adjustment using the Auto-Model feature of RapidMiner. The decision tree provided the best results, with an accuracy of 100%, and all other performance indicators show perfect or nearly perfect result (Sensitivity, Recall 99.9%). The second-best performing model was Rule induction with an accuracy of 97.78%. Other performance indicators were in a range between 96.68 % and 98.81%. The naïve Bayesian model achieved the lowest performance, with an accuracy of 91.3% and other indicators ranging between 82.7% and 99.3%. This model also reached the lowest Sensitivity, which is for our data set a significant indicator as it shows that the share of correctly predicted poisonous mushrooms. In other words, for Naïve Bayesian model, there is a relatively high number (194) of mushrooms that were predicted to be edible but were poisonous. However, even though the models were predicting great results, it was noticed that the recall score for the Naïve Bayesian and Rule Induction models were way lower than the Decision Tree. Due to the nature of the project (eating poisonous mushrooms could be fatal), it was necessary to improve the model to increase their accuracy and recall value.

Decision Tree

Confusion Matrix

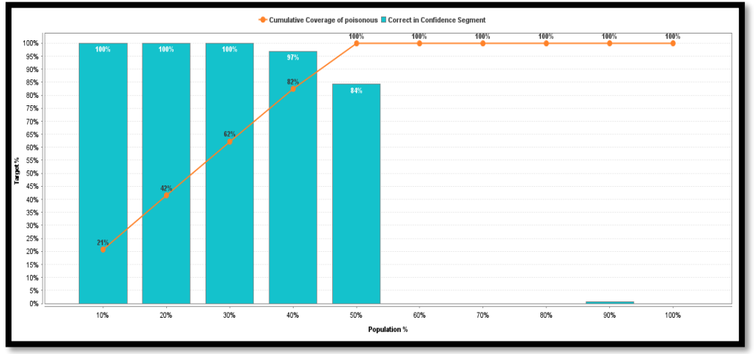
	true poisonous	true edible	class precision
pred. poisonous	1123	0	100.00%
pred. edible	1	1196	99.92%
class recall	99.91%	100.00%	



Naïve Bayesian

Confusion Matrix

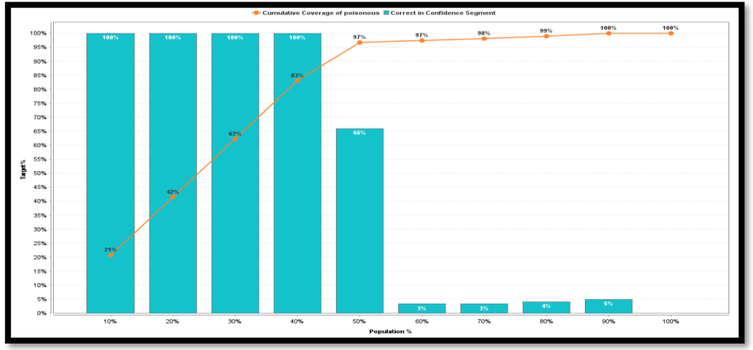
	true edible	true poisonous	class precision
pred. edible	1188	194	85.96%
pred. poisonous	8	930	99.15%
class recall	99.33%	82.74%	



Rule Induction

accuracy: 97.78%

	true poisonous	true edible	class precision
pred. poisonous	1136	15	98.70%
pred. edible	39	1247	96.97%
class recall	96.68%	98.81%	



5.3: Model evaluation of model selected after cleaning the dataset.

Table 6. Evaluation scores after cleaning dataset for the models selected.

Results	*Decision Tree*	*Naïve Bayesian*	*Rule Induction*	*Ensemble vote model*
Accuracy	100%	99.88 %	98.64%	99.92%
Classification Error	0.00%	0.12%	1.36%	0.08%
Recall	100%	99.83%	97.18%	99.83%
Precision	100%	99.91%	100%	100%
F measure	100%	99.87%	98.57%	99.91%
AUC	1.00	1.00	0.997	1.00
Sensitivity	100%	99.83%	97.18%	99.83%
Specificity	100%	99.92%	100%	100%
Runtime	23 s	22 s	22 s	23 s

After cleaning and applying data cleaning and adjustments described in the section above, all three techniques were applied on-again dataset. All models showed an increase in their performance, with accuracy higher than 98.5%. The best performing model is still the decision tree. It delivered 100% accuracy, and all other performance parameters (Classification Error, Recall, Precision, F measure and AUC, Sensitivity and Specificity) also gave perfect result.

The second-best performing model on the dataset was Naïve Bayesian with an accuracy of 99,88% and all other performance parameters above 99.80%, which is an increase of more than 8%. Moreover, the recall score had a significant improvement from 82.7% to 99.83% (more than 17%).

The lowest performance was recorded using Rule Induction with performance indicators in a range between 97% and 99%. Rule induction model also had the lowest Sensitivity, meaning the highest number of poisonous mushrooms that were incorrectly classified as edible (33). The ensemble model also delivers very high performance, with an accuracy of 99.92% and all other parameters between 99.80% and 100%.

There was no significant difference in the runtime for applied models; for the machine, it was run on times in range between 22 – 23 seconds were recorded for four processes, slowest being decision tree and Ensemble vote model. The most time-consuming part of the processes was data preparation, mainly the

identification of outliers. Considering performance indicators and runtime, the most suitable model for the problem seems to be a decision tree.

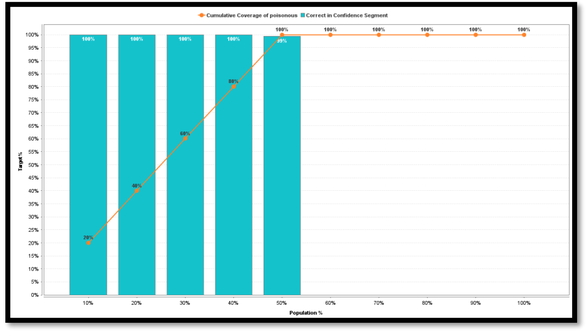
Additionally, although the Decision Tree model was giving perfect scores, it was decided that, for practical reason, an ensemble model would be created to evaluate the performance overall. The majority vote method was chosen, and as can be appreciated in the table above, it has also produced excellent scores. It should be noted that the decision tree model is still performing better than the Vote model. Nevertheless, this last one has overperformed the Naïve Bayesian and Rule Induction models: The ensemble model has an accuracy of 99.92% vs 99.88% & 98.64%, respectively, while the recall value is higher by 2.66% against the Rule Induction model. This process has confirmed that ensemble models tend to perform better than base models.

Finally, one more thing to highlight is the lift charts of the models. As it can be observed, the decision tree model has achieved better results as it was expected, as it would only take 50% of the population to correctly classified 100% the positive class (poisonous mushrooms). In comparison, the Rule Induction model would require 90% of the data to produce the same result. The ensemble model would be able to correctly predict the classification in 50% of the population, surpassing the capabilities of the Naïve Bayesian and Rule Induction model. Finally, in general terms, this means that the models would be good at classifying and ordering the dataset than randomly selecting a portion of the data to pick the same percentage of the poisonous class (or % Target), which could translate to a reduction of costs for model implementation.

Decision Tree

accuracy: 100.00%

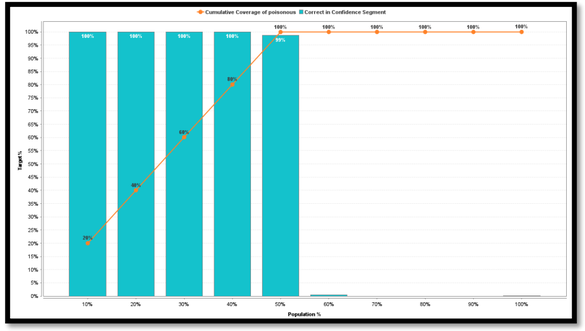
	true poisonous	true edible	class precision
pred. poisonous	1172	0	100.00%
pred. edible	0	1262	100.00%
class recall	100.00%	100.00%	



Naïve Bayesian

accuracy: 99.88%

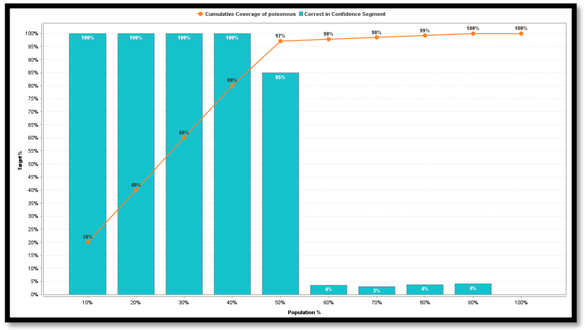
	true edible	true poisonous	class precision
pred. edible	1261	2	99.84%
pred. poisonous	1	1170	99.91%
class recall	99.92%	99.83%	



Rule Induction

accuracy: 98.64%

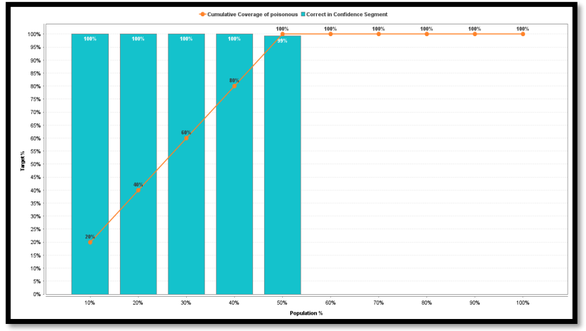
	true edible	true poisonous	class precision
pred. edible	1262	33	97.45%
pred. poisonous	0	1139	100.00%
class recall	100.00%	97.18%	



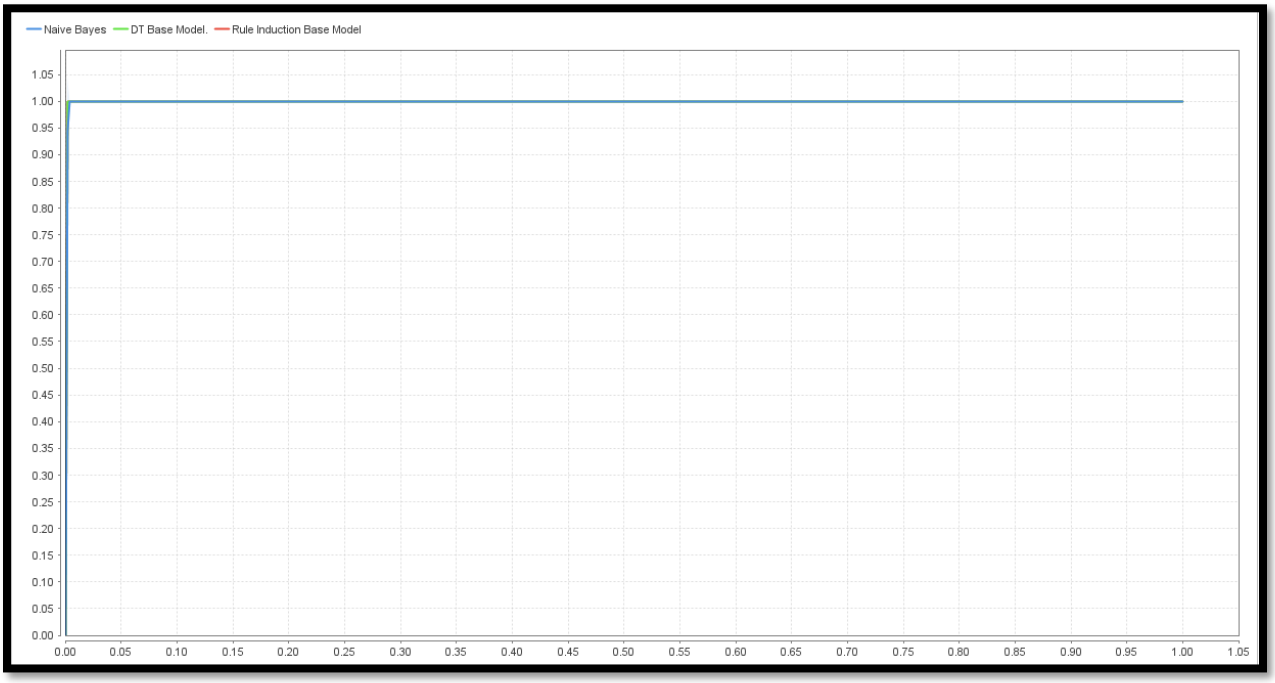
Ensemble Model

accuracy: 99.92%

	true poisonous	true edible	class precision
pred. poisonous	1170	0	100.00%
pred. edible	2	1262	99.84%
class recall	99.83%	100.00%	



5.4: ROC comparison



As it is known, ROC plots true positive rate vs false positive rate, and it is a great indicator to compare graphically multiple models and their performance. These types of curves can tell how well the models are able to correctly identify the positive class in a dataset. Generally, if a ROC is close to 1, then the more accurately predicts the class of interest. At the same time, a more linear model would represent random chance models and, therefore, not a very good one (Kotu & Bala, 2019).

Based on the above statements, it is clearly appreciated that all models are fantastic at predicting the positive class, as the graph shows that they all are nearly at a value of 1. Furthermore, Mierswa (2019) explains that the models that are too close to the value of 1 should be considered suspicious, and their results should be taken cautiously. However, as it was presented during the data understanding chapter, the mushroom dataset was created synthetically. Thus, we are conscious that it would be hard to find such results in reality, but for practical purposes, it does fulfil the objective of the analysis.

6: Deployment

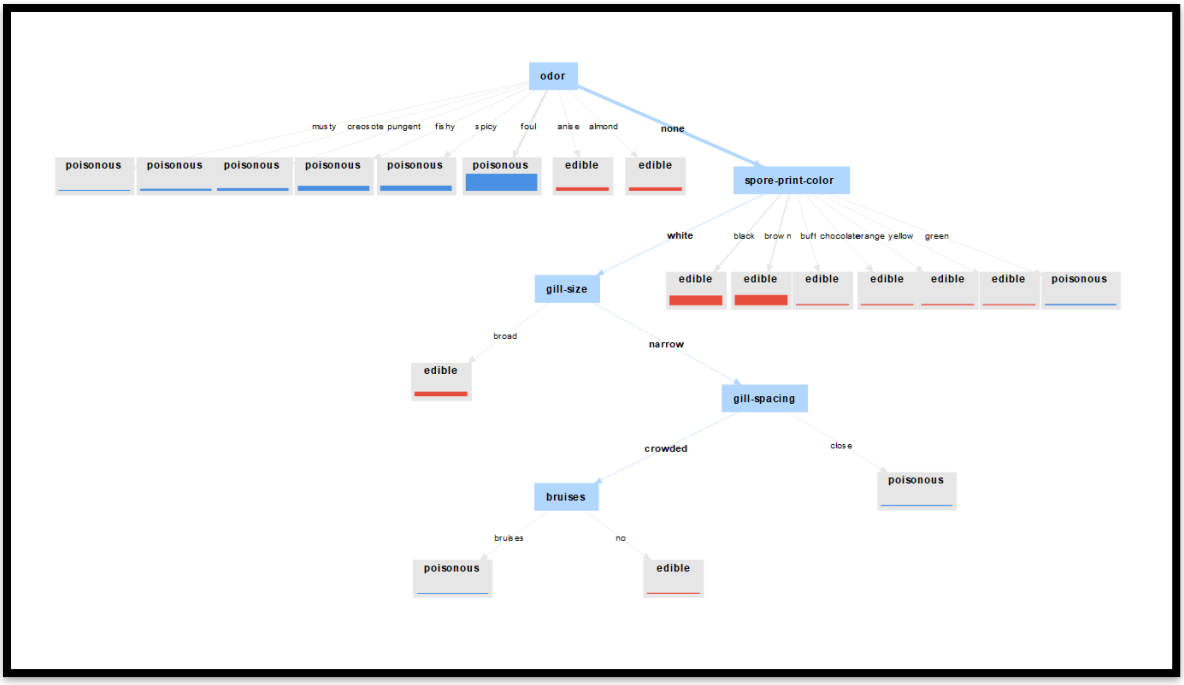
6.1: Shruum Application

The work performed on the Mushroom dataset was deployed as a lightweight application, where end user can, by following a step-by-step procedure, enter the features of an observed mushroom and find out if the mushroom is edible or poisonous.

The Decision Tree was model with the highest accuracy and recall and was used as the basis of the program. This had the added benefit of providing the input structure of the program as the program can follow a simple dendritic tree to reach a final classification of the mushroom. The characteristics are requested from the user in order of highest importance to the class: 1. odor, 2. spore print colour, 3. gill size, 4. gill spacing, 5. Bruises.

The application does not use the full ensemble model as the run time for a single query would be too long for an end user. The run time for the full process in Rapid-Miner for example was 23 seconds on average.

Snapshot of the Shruum program, written in Python.



Decision tree visualisation used to build the Python application.

6.2: Future Work

While the proof of concept using the Decision Tree is acceptable as a prototype for future commission, the ensemble model obtained through the data mining process can also be deployed for batch queries, using the developed Rapid Miner process. The increased data input would justify the additional query length.

7: Bibliography

Brownlee, J., 2019. *How to Develop a Naive Bayes Classifier from Scratch in Python.* [Online] Available at: https://machinelearningmastery.com/classification-as-conditional-probability-and-the-naive-bayes-algorithm/
[Accessed 29 03 2021].

Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P., 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal Of Artificial Intelligence Research,* Volume 16, pp. 321-357.

Grzymala-Busse, J., 2005. Rule Induction.. In: R. L. Maimon O., ed. *Data Mining and Knowledge Discovery Handbook.* 1st ed. Boston: Springer, pp. 277-294.

Gupta, P., 2017. *Decision Trees in Machine Learning.* [Online] Available at: https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052 [Accessed 29 03 2021].

Hazra, A., 2017. Using the confidence interval confidently. *Journal of Thoracic Disease,* Vol 9(No 10), pp. 4125-4130.

Hyun, K., 2013. The prevention and handling of the missing data. *Korean Journal of Anesthesiology,* 64(5), p. 402–406.

Kotu, V. & Bala, D., 2019. *Data Science: Concepts and Practice.* 2nd ed. Cambridge: Morgan Kaufmann.

Lockhart SR, E. K. V. S., 2017. Simultaneous emergence of multidrug-resistant Candida auris on 3 continents confirmed by whole-genome sequencing and epidemiological analyses.. *Clinical Infectious Diseases ,* Volume 64(Issue 2), p. Pages 134–140.

Mierswa, I., 2019. *RapidMiner.* [Online] Available at: https://community.rapidminer.com/discussion/54916/auto-model-and-variables-quality [Accessed 05 04 2021].

Moner, C., 2019. *A Guide for Making Black Box Models Explainable.* [Online] Available at: https://christophm.github.io/interpretable-ml-book/ [Accessed 29 03 2021].

Press, W. H., Teukolsky, S. A., Vetterling, W. T. & Flannery, B. P., 2007. Numerical Recipes, The Art of Scientific Computing. In: 3rd ed. New York: Cambridge University Press, p. 761.

Sharma, M., 2018. *09. How to treat missing values.* [Online] Available at: https://datasciencebeginners.com/2018/11/06/09-how-to-treat-missing-values/ [Accessed 03 04 2021].

Yıldırım, S., 2020. *Towards Data Science.* [Online] Available at: https://towardsdatascience.com/naive-bayes-classifier-explained-50f9723571ed [Accessed 29 03 2021].