CMP6202

Artificial Intelligence and Machine Learning

YEVGENIYA KOVALCHUK

Ahmed Mohamed

18124225

[Ahmed.mohamed3@mail.bcu.ac.uk](mailto:Ahmed.mohamed3@mail.bcu.ac.uk)

18/12/2020

Word count: 3400

Table of Contents

[Abstract 3](#_Toc60047678)

[Introduction 3](#_Toc60047679)

[Dataset 3](#_Toc60047680)

[Problem 4](#_Toc60047681)

[Machine learning model 5](#_Toc60047682)

[Summary of the approach 5](#_Toc60047683)

[Data pre-processing, visualisation, feature selection 5](#_Toc60047684)

[Model training and evaluation 13](#_Toc60047685)

[Results and discussion 15](#_Toc60047686)

[Conclusion 18](#_Toc60047687)

[References 19](#_Toc60047688)

[Figure 1: The attributes and values of mushrooms in dataset 4](#_Toc60047715)

[Figure 2: Before The dataset is given headers 5](#_Toc60047716)

[Figure 4: After The dataset is given headers 5](#_Toc60047717)

[Figure 3: code snippet to show how to import and read the dataset 6](#_Toc60047718)

[Figure 5: code snippet to show how to change string values into integers in the dataset 6](#_Toc60047719)

[Figure 6: All values are changed from a string to an int 7](#_Toc60047720)

[Figure 7: Create A histogram for the first feature edible or poisonous mushroom at header 0 8](#_Toc60047721)

[Figure 8: Calculate how many times a value appears in the feature odor at header 5 8](#_Toc60047722)

[Figure 9: Calculate how many times a value appears in the feature gill-color at header 9 8](#_Toc60047723)

[Figure 10: A pairplot of all features in the dataset 9](#_Toc60047724)

[Figure 11: A pairplot of features between headers 0 - 7 10](#_Toc60047725)

[Figure 12: A pairplot of features between headers 8 - 14 11](#_Toc60047726)

[Figure 13: A pairplot of features between headers 15 - 22 12](#_Toc60047727)

[Figure 14: Create 3 separate pairplots from the features in the dataset and save then as a png 12](#_Toc60047728)

[Figure 15: split training and testing data 13](#_Toc60047729)

[Figure 16: using the Gaussian algorithm and creating a prediction dataset 13](#_Toc60047730)

[Figure 17:create and print out an accurancy model 14](#_Toc60047731)

[Figure 18: Create a confusion metrix array then use a heat map to visualise 14](#_Toc60047732)

[Figure 19: Histogram to show the Distribution of edible and poisonous mushrooms 15](#_Toc60047733)

[Figure 20: Histogram to show the Distribution of different mushroom odours 16](#_Toc60047734)

[Figure 21: Histogram to show the Distribution of different mushroom gill-colors 17](#_Toc60047735)

[Figure 22: The accuracy score of the prediction 17](#_Toc60047736)

[Figure 23: Visualisation of the confusion matrix array as a heat map 18](#_Toc60047737)

# Abstract

The purpose of this study is to determine if a machine learning model can be produced, that can correctly distinguish a poisonous mushroom from an edible mushroom. The model is required to have an accuracy score of over 90% to be deemed successful. The approach involves six stages.

First, the dataset needs to be imported and read. The dataset then needs to be visualised. Next, the dataset is spliced into training and testing datasets. A Gaussian Naive Bayes algorithm is used to build a model. Finally, an accuracy score needs to be produced. The accuracy score will then be visualised using a confusion matrix and a heat map. From the results, this machine learning model can be seen as successful as it has over a 90% accuracy score.

This model can accurately tell the difference between a poisonous mushroom from an edible mushroom 92.24% of the time.

# Introduction

The word mushroom is used to describe a variety of fungus, this includes fungi that have a stem and those that do not. The common name for the mushrooms that are discussed in this paper is from the Agaricus and Lepiota Family.

These mushrooms have a wide range of characteristics, this allows mycologist to categorise them. The majority of mushrooms are poisonous for humans to consume. Because there are over 64,000 types of mushrooms in the Ascomycota family it’s extremely difficult for the public to distinguish which ones are poisonous and which ones are edible.

In 2011 the health protection agency's national poison information service had received over 209 calls from NHS to staff trying to treat mushroom poisoning, 147 were from adult seeking medical attention eating mushrooms that they had picked on walks. The U.K.’s most commonly eaten poisonous mushroom is the yellow stained, this is because they can be easily confused with commonly edible varieties which look very similar.

This algorithm aims to determine when given enough data on the physical attributes of a mushroom, if they can be classified as poisonous or edible correctly. This would be done by creating a database that contains known poisonous and edible mushrooms with 22 different physical attributes. The algorithm would then predict the likelihood of a mushroom being poisonous or edible from the given attributes.

# Dataset

This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family (pp. 500-525). Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. The latter is combined with definitely poisonous. The data set consists of 8123 data points with 22 different attributes, as shown in figure 1.

|  |  |
| --- | --- |
| Mushroom attributes | values |
| Poisonous or edible | Poisonous=p, edible=e |
| cap-shape | bell=b,conical=c,convex=x,flat=f, knobbed=k,sunken=s |
| cap-shape | fibrous=f,grooves=g,scaly=y,smooth=s |
| cap-color | brown=n,buff=b,cinnamon=c,gray=g,green=r, pink=p,purple=u,red=e,white=w,yellow=y |
| bruises? | bruises=t,no=f |
| odor | almond=a,anise=l,creosote=c,fishy=y,foul=f, musty=m,none=n,pungent=p,spicy=s |
| gill-attachment | attached=a,descending=d,free=f,notched=n |
| gill-attachment | close=c,crowded=w,distant=d |
| gill-size | broad=b,narrow=n |
| 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 |
| stalk-shape | enlarging=e,tapering=t |
| stalk-root | bulbous=b,club=c,cup=u,equal=e, rhizomorphs=z,rooted=r,missing=? |
| stalk-surface-above-ring | fibrous=f,scaly=y,silky=k,smooth=s |
| stalk-color-above-ring | brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y |
| stalk-color-below-ring | brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y |
| veil-type | partial=p,universal=u |
| veil-color | brown=n,orange=o,white=w,yellow=y |
| ring-number | none=n,one=o,two=t |
| ring-type | cobwebby=c,evanescent=e,flaring=f,large=l, none=n,pendant=p,sheathing=s,zone=z |
| spore-print | black=k,brown=n,buff=b,chocolate=h,green=r, orange=o,purple=u,white=w,yellow=y |
| population | abundant=a,clustered=c,numerous=n, scattered=s,several=v,solitary=y |
| habitat | grasses=g,leaves=l,meadows=m,paths=p, urban=u,waste=w,woods=d |

Figure : The attributes and values of mushrooms in dataset

The data points in the dataset are categorical, allowing the distribution of each data points to be plotted on a histogram easily.

The type of prediction used for this dataset is classification as the values are categorical meaning there is no order to them. The data points can be separated into distinct features which later can be split into training and testing data sets to be used by the chosen model to create an accurate prediction. The model will be able to predict how likely all features match the target feature based on what it learns from the training dataset.

# Problem

There are over 600 known poisonous mushrooms in the wild. There are 6000 ingestions of poisonous mushrooms annually in the United States. The majority of people who forage for mushrooms do not ingest poisonous ones however, many do. Depending on the species of mushrooms and amount of toxin ingested there are a number of symptoms that can appear 10 - 12 hours after it is consumed. The symptoms range from acute gastroenteritis to symptoms of nausea vomiting all the way to organ failure such as a liver failure. ;ref;

This can be a major problem for someone who is not experienced in distinguishing different species apart, my machine learning algorithm is designed to predict if a mushroom is poisonous or edible when given a list of physical attributes. The accuracy of the algorithm needs to be extremely high, as it could save someone from being taken to the hospital or in extreme cases prevent fatalities.

All attributes required can be collected from a visual/odour based inspection of the mushroom, this lowers the expertise level needed to use the algorithm.

# Machine learning model

## Summary of the approach

The first step is to import and read the Dataset file, this is done with the Pandas library. Pandas is a library written for python to manipulate and analyse data (DataFrames, 2020). The next steps include understanding and visualising the data this can be done by plotting the distribution of each future. mainly histograms were used for this dataset. The data is then required to be split into a set of target data and training data. The next is to build a model using the Guasian Naive Bayes algorithm. Finally, the model is given an accuracy score out of 100%. and the accuracy is visualised using a confusion matrix plus a heat map from the seaborn library.

## Data pre-processing, visualisation, feature selection

The Panda library needs to be imported, this is done by doing ‘Import pandas as pd’, the ‘pd’ variable makes it easier to reference further down the code. Pandas contains a method called ‘.read\_csv’ which can read the CSV file format and store it inside a pandas data frame (as shown in figure 4). That data frame is then returned to the variable ‘data’.

The original dataset does not have any headers (as shown in figure 2), this makes it more difficult to traverse columns and rows as there is no header to refer to. However the ‘header=none’ parameter in 'pd.read\_csv' creates a numerical header for each column starting at 0, Later on (as shown in figure 3), these headers can be used to refer to the columns.

|  |
| --- |
|  |

Figure : Before The dataset is given headers

|  |
| --- |
|  |

Figure : After The dataset is given headers

|  |
| --- |
| 1 # Author: Ahmed Mohamed, ID: 18124225  2 # a machine learning model to predict if a mushroom is poisonous or edible.  4 # to manipulate data as a dataframe  **5 import** **pandas** **as** **pd**  6 # import numpy as np  **7 import** **numpy** **as** **np**  25 # import dataset.  26 data = pd.read\_csv('mushroms.csv', header=None) |

Figure : code snippet to show how to import and read the dataset

At this point the data frame contains string values, these values now need to be converted into integers. The function label encoder from the sklearn library can be used to solve this problem. This function can be used to transform non-numerical labels into numerical labels (LabelEncoder, 2020).

The following loop in figure 5 traverses the pandas data frame, each column is a separate feature. Each feature has multiple values which are then given a numerical value from 0 onwards (as seen in figure 6). The exception prints out an error message that including both the feature and the error code. The result of the Encoder is returned to the variable ‘data’.

|  |
| --- |
| 9 # to pre-process data  **10 from** **sklearn** **import** preprocessing  15 # to change string values into ints  **16 from** **sklearn.preprocessing** **import** LabelEncoder  30 # use LabelEncoder()from scikit-learnlibrary to convert string feature 31 values to ints.  **32 def** **Encoder**(data):  33 le = LabelEncoder()  34 # Loop though all columns in the dataset.  35 **for** feature **in** data:  36 **try**:  37 # Fit label encoder and return encoded labels.  38 data[feature] = le.fit\_transform(data[feature])  39 **except**:  40 **print**('Error encoding ' + feature)  41 **return** data  43 data = Encoder(data) |

Figure : code snippet to show how to change string values into integers in the dataset

|  |
| --- |
|  |

Figure : All values are changed from a string to an int

To allow visualisation of the dataset the following libraries need to be imported. The matplotlib library and the seaborn library.

Histograms are used to show the distribution of data by grouping values into ranges this is illustrated with bars of different heights. The height of each bar shows how many values fall into that range (Research Ltd, 2018).

The first histogram is used to display the distribution of edible and poisonous mushrooms in the dataset. This histogram contains only two bars, each bar is given a label from the Array stored in the ‘names’ variable. ‘.figure’ from matplotlib creates a new figure, then the histogram is given its dimensions at line 41. Line 42 places the histogram into the given dimensions. The feature at header 0 is used to plot the histogram, the bins are then given a range to allow the bins to be the same distance apart, the width is minimised 0.8 this helps distinguish the two bars apart from each other (This can be seen in figure 7).

Finally, the bars are aligned to the left this means that they will line up with the labels on the X-axes; and the colour is changed to red. Line 43 changes the frequency of the ticks on the Y and X axes this is done with a range value to allow the bars and the ticks to match, then the ‘labels’ parameter is linked with the array ‘names’. Lastly, the title can be set, the X axes and Y axes can also be labelled. At line 49 the plotted histogram is saved as a ‘.png’ file; line 50 is then used to display the histogram.

|  |
| --- |
| 11 # to visualise data and results  **12 import** **matplotlib.pyplot** **as** **plt**  45 # Histogram graph for the Distribution of edible and poisonous mushrooms.  46 names = ['edible', 'poisonous']  47 # initiate figure.  48 fig = plt.figure()  49 # set hist dimensions.  50 ax = fig.add\_subplot(**1**, **1**, **1**)  51 # Set amount of bins and their properties.  52 ax.hist(data[**0**], bins= range(**3**), rwidth=**0.8**, align='left', color='red')  53 # set frequency of ticks on the x axis.  54 plt.xticks(ticks=np.arange(**2**), labels=names)  55 # set titles and labels.  56 plt.title('A histogram to show the distribution of '  57 '**\n**'  58 'edible and poisonous mushrooms in dataset')  59 plt.xlabel('class of muchrooms')  60 plt.ylabel('Amount of mushrooms')  61 # Save hist as a png.  62 plt.savefig("e-p-distribution.png")  63 # Display hist.  64 plt.show() |

Figure : Create A histogram for the first feature edible or poisonous mushroom at header 0

The process to plot a histogram for the features at header 5 and 9 is roughly the same, as the first histogram. The only difference is the method ‘.value\_counts()’ is used (as can be seen in figure 8,9), this creates a list that shows the frequency a value appears (value\_counts, 2020), for the features at header 9 and 19, the list will be in descending order so that the first element is the most frequently occurring value. This list is then used to plot the histogram.

|  |
| --- |
| 67 # Return a Series containing counts of unique values.  68 odor\_data = data[**5**].value\_counts() |

Figure : Calculate how many times a value appears in the feature odor at header 5

|  |
| --- |
| 88 # Return a Series containing counts of unique values.  89 gill\_colors\_data = data[**9**].value\_counts() |

Figure : Calculate how many times a value appears in the feature gill-color at header 9

Finally, a pairplot is used from the Seaborn library, all features in the dataset can be plotted onto one graph however because the dataset is very large it makes it difficult for the graph to be read (as seen in figure 10). To solve this problem three separate pairplots are created so the graphs are legible (as seen in figures 11,12,13). To select the required features the parameter ‘vars’ is used, this allows the header value for each feature to be chosen (as seen in figure 14). Once each pairplot is created it is then be saved as a ‘png’ file (as seen in figure 14).

|  |
| --- |
|  |

Figure : A pairplot of all features in the dataset

|  |
| --- |
|  |

Figure : A pairplot of features between headers 0 - 7

|  |
| --- |
|  |

Figure : A pairplot of features between headers 8 - 14

|  |
| --- |
|  |

Figure : A pairplot of features between headers 15 - 22

|  |
| --- |
| 21 # Seaborn visualization library  **22 import** **seaborn** **as** **sns**  106 # Create the pairplots for all data in the dataset and select the required features.  107 data\_plot1 = sns.pairplot(data, vars=[**1**,**2**,**3**,**4**,**5**,**6**,**7**])  108 data\_plot2 = sns.pairplot(data, vars=[**8**,**9**,**10**,**11**,**12**,**13**,**14**])  109 data\_plot3 = sns.pairplot(data, vars=[**15**,**16**,**17**,**18**,**19**,**20**,**21**,**22**])  110 # Save all pairplots as a png file format.  111 data\_plot1.savefig("seaborn-pairplot-1.png") #Save the graph as a png  112 data\_plot2.savefig("seaborn-pairplot-2.png") #Save the graph as a png  113 data\_plot3.savefig("seaborn-pairplot-3.png") #Save the graph as a png  114 # Display all pariplots.  115 plt.show() |

Figure : Create 3 separate pairplots from the features in the dataset and save then as a png

## Model training and evaluation

The model requires a target dataset and a feature dataset which is constructed from the original dataset. The target dataset is the first feature in the ‘CSV’ file, which shows if a mushroom is edible or poisonous, all the other features are used for feature testing and training data.

At this stage, the data is split into feature training/test and target training/test data sets, 67% - 33% split was used in this model a random state of 10 was also used a (as seen in figure 15). This ensures that the target and training datasets remain the same every time the code is executed if the random state is not specified then every time the code is executed a new random value is generated; the target and training datasets would contain different values each time.

|  |
| --- |
| 118 # data slicing  119 # Select all rows for all column expect column at header 0 as that will be use for the target.  120 features = data.iloc[:, **1**:]  121 # Select all rows for the column at header 0, which shows if the mushroom is poisonous or edible.  122 target = data.iloc[:, **0**]  123 # split the data into training and testing sets.  124 features\_train, features\_test, target\_train, target\_test = train\_test\_split(features, 125 # Split training and testing size 67% to 33%  126 target, test\_size=**0.33**, random\_state=**10**) |

Figure : split training and testing data

The Gaussian Naive Bayes algorithm was used to build the model. Naive Bayes is a machine learning method that is used to predict the likelihood that an event will occur given a dataset. There are three types of Naive Bayes models, Multinomial, Bernoulli, and Gaussian (Ray, 2017). The Gaussian model is used here as it’s the best out of the three for making predictions from normally distributed features, which the dataset is.

The model is then trained using the target training dataset and features training dataset, which creates a predicted dataset (figure 16).

|  |
| --- |
| 17 # to modelgaussian naive bayes classifier  **18 from** **sklearn.naive\_bayes** **import** GaussianNB  128 # building model using Gaussian Naive Bayes algorithm.  129 model = GaussianNB()  130 # Fit Gaussian Naive Bayes according to features\_train and target\_train.  131 model.fit(features\_train, target\_train)  132 # Perform classification.  133 target\_pred = model.predict(features\_test) |

A prediction is then made using the feature test dataset. Finally, an accuracy score is printed out, this uses the target test data sets and the predicted dataset (figure 17).

Figure : using the Gaussian algorithm and creating a prediction dataset

|  |
| --- |
| 19 # to calculate the accuracy score of the model  **20 from** **sklearn.metrics** **import** accuracy\_score  135 # print accuracy score  **136 print**(accuracy\_score(target\_test, target\_pred, normalize=True) \* **100**) |

Figure :create and print out an accurancy model

A confusion matrix is required, this can only be used for a classification prediction. The confusion matrix will calculate how many times the machine learning algorithm has been correct or incorrect for each mushroom in the dataset. The confusion metric requires the target test dataset and the target prediction dataset. To visualise the array created by the confusion matrix a heat map is used (figure 18).

|  |
| --- |
| 23 # to visualise accuracy score  **24 from** **sklearn.metrics** **import** confusion\_matrix  138 # create a matrix with the prediction data from the model  139 cf\_matrix = confusion\_matrix(target\_test, target\_pred)  140 # create a heat map with the seaborn library for the matrix  141 sns.heatmap(cf\_matrix / np.sum(cf\_matrix), annot=True,  142 fmt='.2%', cmap='Blues')  143 # save heat map as a png file format.  144 plt.savefig("e-p-predictions.png")  145 # Display the heat map.  146 plt.show() |

Figure : Create a confusion metrix array then use a heat map to visualise

## Results and discussion

|  |
| --- |
|  |

Figure : Histogram to show the Distribution of edible and poisonous mushrooms

Figure 19, which is created using the first feature in the dataset, shows that the distribution of edible and poisonous mushrooms is about the same. This should allow for a more accurate prediction from the model.

|  |
| --- |
|  |

Figure : Histogram to show the Distribution of different mushroom odours

|  |
| --- |
|  |

Figure : Histogram to show the Distribution of different mushroom gill-colors

However, this kind of distribution is not the same for all features in the dataset, some features have a more bias distribution (as seen in figures 20,21). This can be due to a limited amount of data points in the dataset.

However, the distribution is fairly even for most features in the dataset which can correspond to a more accurate prediction from the model.

|  |
| --- |
|  |

Figure : The accuracy score of the prediction

The accuracy score that has been produced is 92% (as seen in figure 22), in my opinion, this is an extremely high accuracy rate. This machine learning algorithm can correctly predict if a mushroom is poisonous or edible when given accurate attributes.

|  |
| --- |
|  |

Figure : Visualisation of the confusion matrix array as a heat map

When looking at figure 23 created using the confusion matrix array, the error rate for both poisonous and edible mushroom is roughly the same. On this heat map the label '1' represents a poisonous mushroom, the label '0' represents an edible mushroom. The model incorrectly predicted a mushroom is edible when it’s not 3.66% of the time, the model also incorrectly labelled a mushroom as poisonous when in fact it was edible, 4.10% of the time. Overall, the error is 7.76%.

# Conclusion

In conclusion, looking at the prediction score of 92% for this model illustrates that a mushroom can be labelled as edible or poisonous just from physical observations, I believe that this accuracy score can be increased closer to 100% if a different algorithm is used to build the model. For example, a different naive bias algorithm as it could allow for a better fit. Also, if a larger dataset is used the training dataset size would be increased resulting in a more accurate model.

One of the downfalls of this dataset is that there are too many null data points. This limited the amount of data that could be used for both training and testing. One way to solve this problem is to change those null values to an integer using the label encode from the sklearn library. I was not able to carry out this operation, for this model those null values were ignored. Another thing that could make this model more accurate is increasing the number of features in the dataset, this would allow the model to have more training data to compare to the target feature.

# References

DataFrames, P., 2020. *Intro To Data Structures — Pandas 1.2.0 Documentation*. [online] Pandas.pydata.org. Available at: <https://pandas.pydata.org/pandas-docs/stable/user\_guide/dsintro.html#dataframe> [Accessed 27 December 2020].

LabelEncoder, S., 2020. *Sklearn.Preprocessing.Labelencoder — Scikit-Learn 0.24.0 Documentation*. [online] Scikit-learn.org. Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html> [Accessed 27 December 2020].

Ray, S., 2017. *Learn Naive Bayes Algorithm | Naive Bayes Classifier Examples*. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/> [Accessed 27 December 2020].

Research Ltd, L., 2018. *Histograms - Understanding The Properties Of Histograms, What They Show, And When And How To Use Them | Laerd Statistics*. [online] Statistics.laerd.com. Available at: <https://statistics.laerd.com/statistical-guides/understanding-histograms.php> [Accessed 27 December 2020].

value\_counts, P., 2020. *Pandas.Series.Value\_Counts — Pandas 1.2.0 Documentation*. [online] Pandas.pydata.org. Available at: <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.value\_counts.html> [Accessed 27 December 2020].