**MACHINE LEARNING PROJECT**

**BINARY CLASSIFICATION OF MUSHROOMS**

**USING THE KNN**

**Table of Contents**

[**PROBLEM STATEMENT AND OBJECTIVES** 3](#_Toc166680273)

[**INTRODUCTION & DESCRIPTION OF DATA** 4](#_Toc166680274)

[**LITERATURE REVIEW & RELATED WORK** 5](#_Toc166680275)

[**MODEL APPROACH AND PROJECT TRAJECTORY** 6](#_Toc166680276)

[**RESULTS, CONCLUSION & FUTURE WORK** 8](#_Toc166680277)

# **PROBLEM STATEMENT AND OBJECTIVES**

The primary problem addressed by this project is the accurate classification of mushrooms as edible or non-edible based on their various attributes. is crucial for preventing mushroom poisoning and ensuring safe consumption. Misidentification of mushrooms is serious on the grounds that consuming poisonous mushrooms can prompt extreme medical problems, including gastrointestinal misery, organ disappointment, and even demise. Many poisonous mushrooms closely resemble edible varieties, making visual identification challenging and increasing the risk of accidental poisoning.

This project deploys KNN – K-Nearest Neighbor classification model (supervised learning) to categorize the mushrooms as either edible or non-edible.

The major objectives of the project include:

* Preprocessing the dataset to handle categorical variables effectively.
* Tuning the KNN algorithm's hyperparameters to optimize performance.
* Evaluating the model's accuracy, precision, recall, and F1-score to ensure reliable predictions.

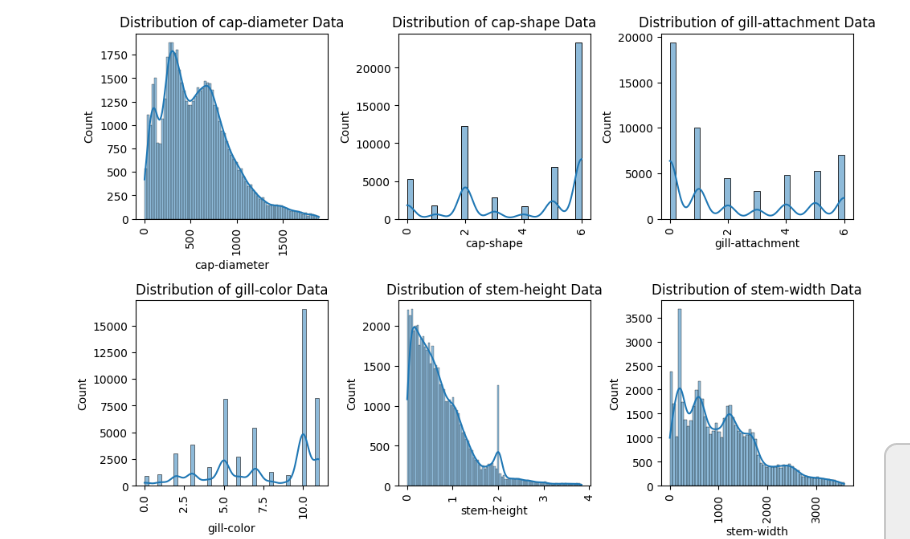
# **INTRODUCTION & DESCRIPTION OF DATA**

The dataset has been downloaded from Kaggle. It includes eight features of mushrooms:

* Cap diameter
* Cap shape
* Gill attachment
* Gill color
* Stem height
* Stem width
* Stem color
* Season

The output column describes the target class (edible mushrooms – 0, poisonous mushrooms – 1). This dataset was cleaned using various techniques such as modal imputation, one-hot encoding, z-score normalization, and feature selection. The cleaned dataset is available on Kaggle and has been downloaded from there.

The preliminary EDA was helpful in gaining insights into the data. It described the distribution of each feature in the dataset as shown in the following diagrams:



# **LITERATURE REVIEW & RELATED WORK**

The source of the dataset is Kaggle ([*https://www.kaggle.com/datasets/prishasawhney/mushroom-dataset*](https://www.kaggle.com/datasets/prishasawhney/mushroom-dataset)).

To choose the most suited classification algorithm for this project, the *freeCodeCamp.org* YouTube video on machine learning was viewed (*link:* [*https://youtu.be/i\_LwzRVP7bg?feature=shared*](https://youtu.be/i_LwzRVP7bg?feature=shared)*).* The EDA was observed from the following project: *https://www.kaggle.com/code/ssyyhh/eda-of-class-classification-randomforest-99*.

The KNN implementation videos were viewed, and their links are as follows:

* [*https://youtu.be/iJfcRV4PPnY?feature=shared*](https://youtu.be/iJfcRV4PPnY?feature=shared)
* [*https://youtu.be/2iSGWHIpB5k?feature=shared*](https://youtu.be/2iSGWHIpB5k?feature=shared)
* [*https://youtu.be/OO7Y5wQWnQs?feature=shared*](https://youtu.be/OO7Y5wQWnQs?feature=shared)
* [*https://youtu.be/rTEtEy5o3X0?feature=shared*](https://youtu.be/rTEtEy5o3X0?feature=shared)
* *https://youtu.be/CRW-OM4AlNs?feature=shared*

The previous projects on this dataset include the use of various machine learning algorithms like support vector machine (SVM), ensemble learning and random forest classifier and, artificial neural network (ANN). They are all listed on the Kaggle.

# **MODEL APPROACH AND PROJECT TRAJECTORY**

The K-nearest neighbor algorithm was used because of its ability to deal with categorical data. It is a straightforward and easy to understand and implement algorithm that solves such classification problems in an accurate manner. Moreover, the availability of python libraries available for KNN made it easy to implement.

K-nearest neighbor makes use of two hyperparameters; K and the distance between data points (to quantify the similarity between two points). It is also known as an “instance-based” classifier as it makes use of the feature similarity. KNN stores all the available cases and classifies the new cases based on how the majority of their neighbors have been classified. It gives discrete output (example categories).

The *k* refers to the number of nearest neighbors to include in the majority voting process. There are different *distance metrics* such as Euclidean, Manhattan and Minkowski.

The mushroom dataset was pre-processed and cleaned. The categorical values like the season, had been label encoded already. This was then downloaded from Kaggle, and the number of instances was reduced to 1500 as KNN works better on smaller datasets and an equal number of instances of both classes was used to ensure that the model is well-trained for both.

The different libraries were installed and imported. The dataset was downloaded, its number of instances were reduced and then it was used to make the data frame (df – variable name). The feature columns (such as cap diameter, season) were used to create the 2-D matrix X and, the target class column was made a vector, Y. The data in these X and Y variables was then split into two parts; the training data and the testing data. There was an eighty-twenty split (i.e. eight percent of the data was used for training and 20% for testing purposes).

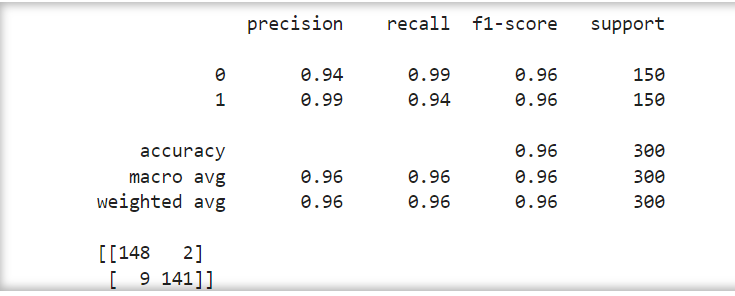
The *KNeighborsClassifier* class was used that makes use of the Euclidean distance to quantify the similarity between two points. The value of *K* in the code is 5 (so the five nearest neighbors will participate in the majority voting). Since this is a binary classification problem, the value of *k* should be an odd number.

Feature scaling (i.e. data standardization or normalization) was conducted using the *StandardScaler* class of the *sklearn.preprocessing* library that uses the standardization method for scaling. Scaling ensures that all features have a similar scale, preventing certain features from dominating the learning process due to their larger magnitude.

The model was then trained and tested. Finally, the evaluation of the model was performed through the confusion matrix and the classification report. The confusion matrix shows the true and predicted classes. The classification report includes precision (the ratio of correctly predicted positive observations to the total predicted positive observations), recall (the ratio of correctly predicted positive observations to all the positive observations), F1-score (the harmonic mean of precision and recall), and support for each class (i.e. the total number of instances of the true response).

# **RESULTS, CONCLUSION & FUTURE WORK**

The output of the KNN implementation is depicted in the following screenshot:



The results in the classification report are as follows:

* Precision for edible mushrooms (class 0): 94%
* Precision for non-edible mushrooms (class 1): 99%
* Recall for edible mushrooms: 99%
* Recall for non-edible mushrooms: 94%
* F1-score for edible mushrooms: 96%
* F1-score for non-edible mushrooms: 96%
* Support for edible mushrooms: 150
* Support for non-edible mushrooms: 150

The results of the confusion matrix show that out of the 150 edible mushrooms, the model classified 148 correctly and misclassified 2 of them as non-edible. Out of the non-edible mushrooms, it classified correctly 141 and misclassified 9 as edible.

**Interpretation**

The precision shows that the overall accuracy of the model is 96%. This indicates that the KNN model is an ideal model for the binary classification of mushrooms, and it is reliable. Moreover, the model has achieved high precision, recall and F1-score for both types of mushrooms. It has a well-balanced performance. However, the model cannot be used for other datasets, as a generic model, without further validation.

**Future Work**

To obtain better results, the following steps should be taken:

* Testing on a larger dataset that has a greater number of instances
* Different parameter tuning techniques should be put to use so that the optimal value of *K* can be found
* Creation of new features from the existing features to help the model learn the patterns better
* Use of other classification models (like random forest) to compare the results and choose the best-suited model for this problem