Edible or Poison? Mushroom Classification using key characteristics

Andrea Cruz

ABSTRACT

During these past few years, with the effects of the pandemic and need for distanced activities, there has been a shift in interest towards outdoor activities as well as a shift in perspective towards self-sustainability and self-provision. These activities include camping, hiking, and foraging. Ever growing in popularity especially, is foraging for edible mushrooms. However, high stakes accompany this type of foraging as a deadly poisonous mushroom can be one minor characteristic from being mistaken as an edible species. This project aims to develop a model that will help automate the classification of mushroom species and make this treasure hunting activity safer and more widely accessible. The mushroom dataset, obtained from the UCI Machine Learning Repository, was split into a training and test data set to implement classification models for identifying whether a particular mushroom is edible or poisonous. During a method comparison for efficiency and accuracy, a Decision Tree Model was determined to be the most viable model for the purposes of this project.

Key Words: Classification, Key Characteristics, Decision Tree

1 Introduction

Mushrooms are a type of fungi that many people are most familiar with seeing in conveniently packaged containers at their local supermarket or as a delicious topping on their favorite pizza. Over 10,000 known species have been identified by mycologists, but it is widely believed that this number could represent only a fraction of the species that exist. It has been a commonplace practice for individuals who live in rural areas to participate in foraging and agriculture to provide self-sustaining lifestyles. During recent years, there has been growing desire for all walks of people to join that type of lifestyle in urban and suburban settings. Unfortunately, as previous generations made transitions to urban/suburban lifestyles, that came with convenience stores and supermarkets with ready-packaged produce. Present day individuals do not necessarily have the handed-down knowledge or years of practice to be certain in the identification of edible vs inedible foraged goods. This is especially problematic in mushroom hunting because mistaking an inedible mushroom as edible, could not only be unpleasant but possibly deadly! According to the North American Mycology Association, as many of 11% of their toxicology reports resulted in death from 1975-2005[1]. Therefore, it is important to have a reliable classification system for individuals to utilize and keep safe.

While more information continues to circulate on the internet and in literature regarding mushroom expert knowledge to aid novice hunters, the certainty of using these methods cannot be considered reliable enough. Leaving this decision making with potentially deadly consequences to human judgement presents itself with opportunity to utilize data mining and modeling to increase reliability. This project aims to contribute additional data mining and further identify most important key characteristics that can help in-the-field hunters identify edible/poisonous mushrooms reliably.

2 Related Work

During the data understanding step of the data mining process several related projects were identified to evaluate different strategies and tools used for this problem.

Vanitha et al.[2] is a paper published in the Bioscience Biotechnology Research Communications journal. Throughout their experimental process, a software tool called the Waikato Environment for Knowledge Analysis (WEKA) was utilized for its collection of data mining algorithms and options for data preprocessing, clustering, classification, regression, and visualization. They compared two different methods (Wrapper and Filter) to select the most relevant attributes, assigned “odor” and “spore_print_color” to be Key attributes, and finally created a Decision tree. However, there was no method to evaluate this as a predictive model, which can be improved as part of this current project by splitting the dataset into a training and test subset.

Tank[3] authored an article on the online publishing company Medium which explored different classifier methods to approach this problem of classifying mushrooms. The methods implemented included decision tree, logistic regression, KNN, SVM, Naive Bayes, and Random Forest Classifiers methods. She implemented some useful python libraries (seaborn and Scikit-learn) to automate some otherwise tedious manual tasks which could be utilized as part of this current project. These tasks included implementation of above-mentioned methods, accuracy calculation, and incorrectly classified instances. A statement made in the article as it relates to identifying what attribute should be considered as most important was as follows: "Usually, the least correlating variable with variable of interest is the most important one for classification." Indicating, performing correlation analysis of the categorical variables can be an additional comparative method in the identification of a key attribute. For all methods implemented, Tank assigned "gill_color" as the most important attribute.

Lastly, a data mining project performed by Foltz & Dakshin[4] implemented many different models tried and evaluated all of them for their accuracy. As a consensus throughout all three related works, it would appear the method with the best accuracy is one that utilizes a decision tree once the identification of the most impactful attribute(s) is assigned as the key attribute.

From the comprehensive evaluation of all three approaches, this project seeks to build on the previous conclusions by utilizing comparative methods to evaluate the identification of the key attributes based on the given data set and evaluation metrics to ensure the model implemented performs at an acceptable accuracy.

3 Proposed Work

The dataset used for this project was gathered from Kaggle as published by the UCI machine learning repository[5]. The tools used in the analysis include a machine learning software library called Scikit-learn.

Following the data mining pipeline, the main tasks of this project are broken down as follows:

1. Data understanding

The domain knowledge and problem to be solved was identified as part of the Abstract and Introduction of this report.

2. Data preprocessing

During the data preprocessing step, the dataset was loaded in and cleaned up to address any potential issues with the dataset. Luckily, this dataset was relatively tidy and did not have any N/A values needing to be addressed. Additionally, attributes of interest were manipulated through transformations, integration, or reduction of the dataset. The target attribute, defined in the dataset as "class", indicated if the mushroom was considered edible ("e") or poisonous ("p"). The distribution of the dataset indicated a relatively balanced breakdown (Figure 1) in relation to the target attribute.

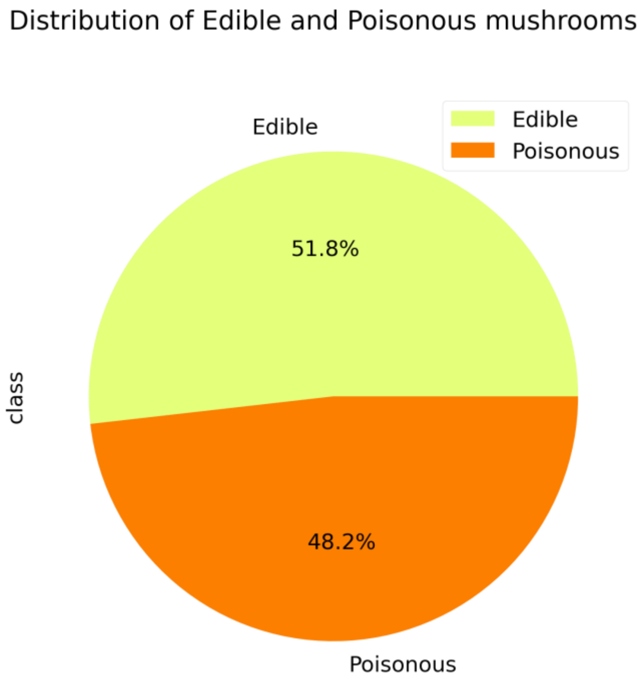
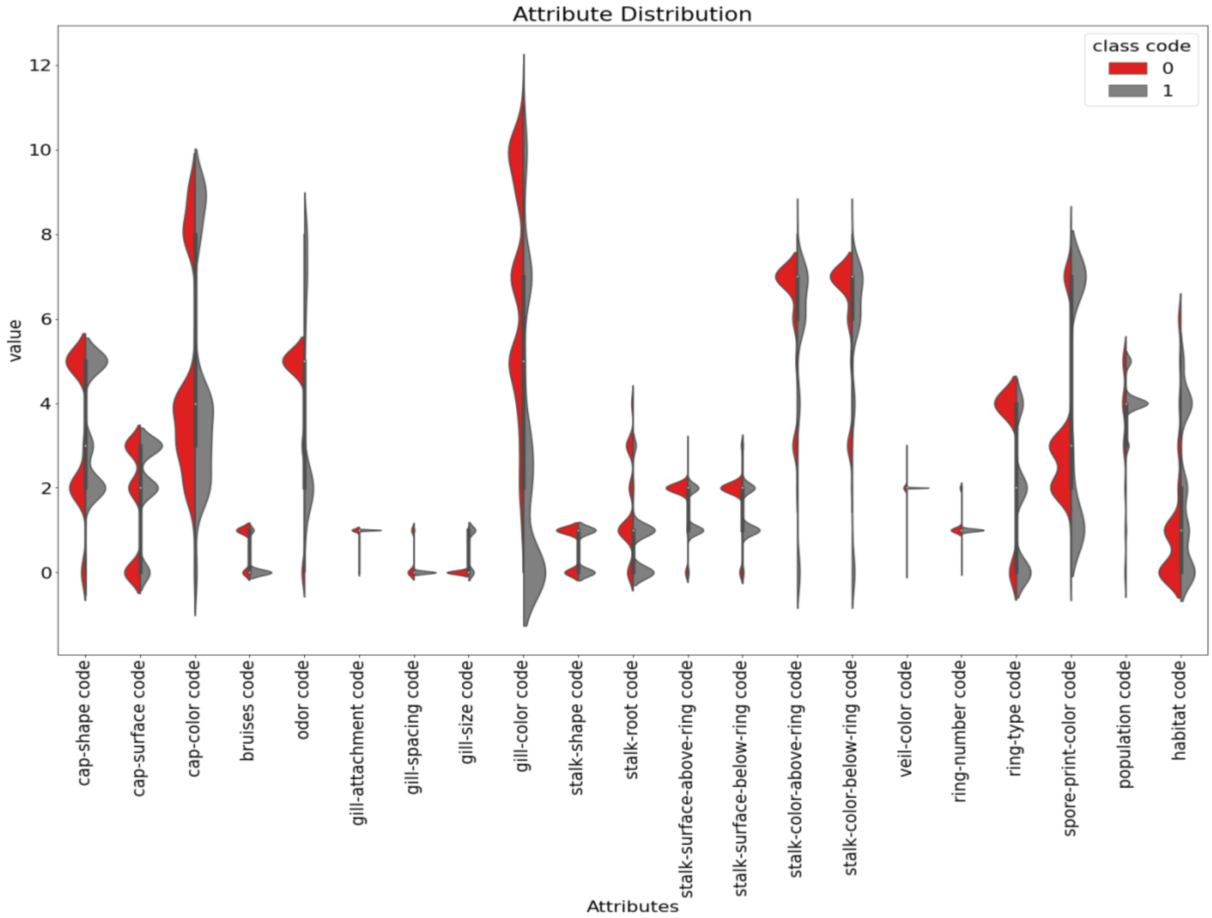


Figure 1 Distribution of Mushrooms by "class"

The distribution of the rest of the attributes were analyzed to determine if any follow a similar balanced dual breakdown that could be correlated with the "class" attribute distribution. This was primarily done with distribution plots to identify any correlation between categorical variables. None such pattern could be observed and indicated more complex analysis would be required. However, during this analysis the attribute of "veil-type" was revealed to only contain one unique and would not serve much use in classification of edible vs poisonous mushrooms. Therefore, it was dropped from the dataset.

Figure 2 Attribute distribution after data was encoded



With the Scikit-learn software, categorical data must be encoded numerically in order to implement machine learning models. All column types were changed from “object” to “category” so they could be encoded with the Scikit-learn function LabelEncoder. After this transformation was performed, additional data visualization in the form of a violin plot was constructed to help identify distribution patterns or correlation in relation to target attribute, “class” (Figure 2). The “class” code was as follows:

- 0 = Edible
- 1 = Poisonous

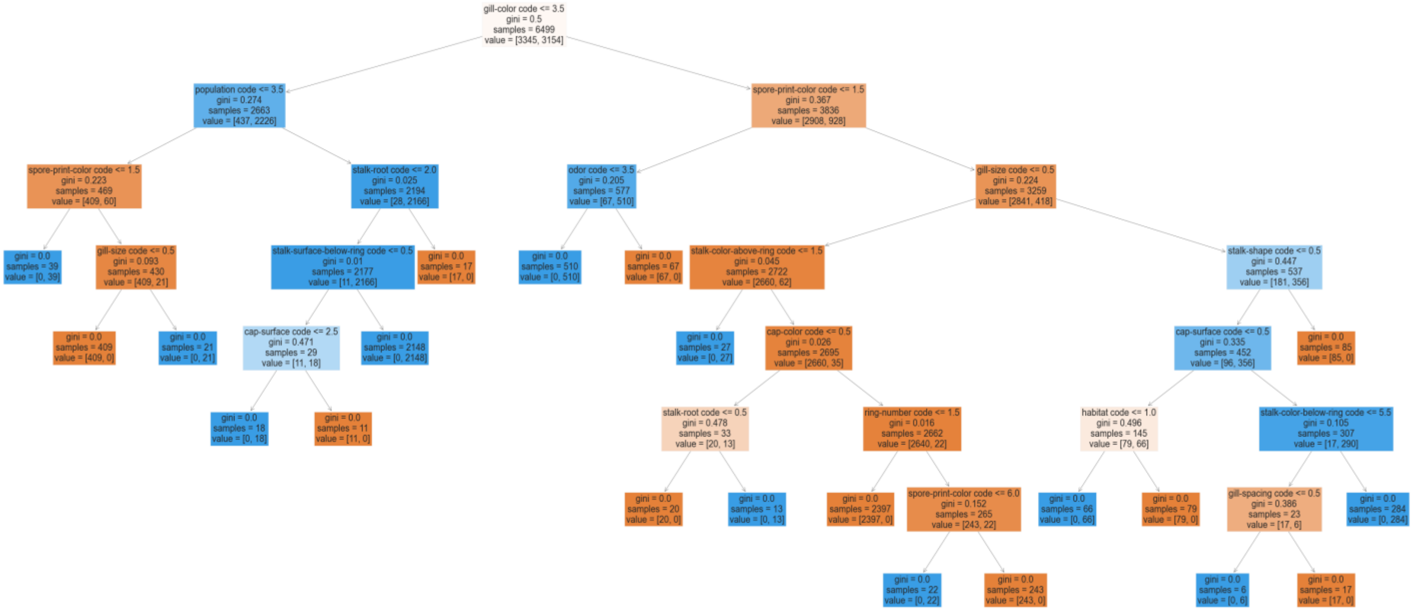
There was some distinct separation between “class” in attribute "gill-color" above or below a value of 3. We can also see some distinct clustering for each class in the attribute "odor" around 1-3 vs 4-6. The distributions for these two attributes were further isolated and broken down by individual attribute values. The class separation about the value of 3 for “gill-color” seemed more distinct that the value ranges for “odor. This could be indicators that “gill-color” and “odor” may be stronger contributors to mushroom classification.

Lastly, the encoded data set was split into a training and testing subset in an 80:20 ratio respectively, for use in model implementation and prediction.

3. Data Modeling

From the initial Data understanding step, there was evidence from previous related works and domain knowledge that identified that this problem can be solved using “Classification” or supervised learning modeling. Especially relevant is the use of Decision Tree Induction as the target class is predefined as “edible” or “poisonous” and all other classes can be incorporated using attribute split. The proposed models to be implemented for this classification were Decision Tree, Random Forrest, and Naïve Bayes Classification.

Figure 3 Decision Tree (Gini Impurity)



The Decision tree model was implemented with the Decision Tree Classifier model from the Scikit-learn library.[5] (Figure 3) The criterion for determining hierarchy of attribute assignments was evaluated with Gini impurity and, alternatively, Information Gain.

An alternate method of implementing a decision tree utilizes combining many decision trees models while introducing some randomness. This model was implemented with the Random Forest Classifier model, also from the Scikit-learn library[6]. After the implementation and analysis of the Decision tree model, the default criterion of Gini Impurity was maintained for the Random Forest Classifier.

Lastly, a Naïve Bayes model was implemented using a Gaussian Naïve bayes algorithm, again from the Scikit-learn library[7].

4 Evaluation

The experimental set up was as follows:

Data was preprocessed in preparation for implementation of various classification models. The Decision Tree and Random Forrest Classifiers utilized Gini Impurity or Information Gain criterion to determine which attribute is considered most important for classifying the mushrooms. The Gaussian Naïve Bayes classifier utilizes probabilities of the target attribute values based on the dataset used for training. After dataset has been preprocessed, it was split into training and testing subsets for data modeling evaluation, in an 80:20 proportion. For the modeling evaluation, the attribute aiming to be predicted was “class”, which indicates if mushroom is edible or poisonous. Decision Tree induction, Random Forrest, and Gaussian Naïve Bayes classification models were applied on the training dataset to develop a predictive model that was then used on the test dataset. The predicative model’s determination of “class” was evaluated against the test dataset’s true assignment of “class” for accuracy. Effectiveness was assessed using the Scikit-learn “classification report” function and accuracy calculations with an acceptance criterion of $\geq$ 95%. Efficiency was assessed comparatively by the time it took to train the models. For models that did not meet accuracy acceptance criteria, false negative rates (i.e. a mushroom is identified as edible when it is poisonous) were additionally reported.

The Decision Tree algorithm utilizes supervised learning to produce a hierarchal tree starting at a root node that feeds into internal nodes based on particular attributes until the subset becomes homogenous. Typically, a greedy divide and conquer algorithm implemented, and in the case of the Decision Tree Classifier of the Scikit-learn, an optimized classification and regression tree (CART) algorithm is used. The criterion is defaulted to using Gini impurity to identify the most important attribute to split nodes. The impurity value is a probability that a given data is misclassified (a lower value is more desirable).

Alternatively, the criterion of "entropy" uses Information Gain to identify the most important attribute. Both criteria were implemented, with consistent results of "gill-color" assigned as the root node (and most important attribute) and an accuracy rating of 100% for predicting the class in the test data subset. Training of the Decision Tree model, with either criterion, took between 1.1- 2.1 seconds.

The Random Forrest Classifier utilizes the same principles as the Decision Tree Classifier, however instead of a singular decision tree it combines many decision trees models to account for inherent randomness that is often observed in real-world data. The most important attribute was assigned to root node of each decision tree, therefore a breakdown of root node attribute distribution was explored. This yielded consistent results with previous observations, "gill-color" was the top count, followed closely by "odor." Application of this model to the testing dataset yielded an accuracy rating of 100%. Training of the Random Forrest model took 4.9 seconds.

The Naïve Bayes Classifier utilizes Bayes Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \qquad (1)$$

which draws on probability that a mushroom is edible or poisonous given different present attributes. A Gaussian Naive Bayes algorithm was implemented due to the presence of continuous attribute values as a result of previous encoding. The naïve bayes classifier was the only model with ≤ 100% accuracy (90.65%), therefore a confusion matrix was generated to further explore performance of implemented model (Figure 4). The confusion matrix indicates counts and proportion of correctly identified "True" instances and misidentified "False" instances. Additionally, due to this algorithm's structure, there is no attribute that is considered most important for identification because all attributes are assumed to have an equal effect on the outcome. Training of the Naïve Bayes model took 0.7 seconds.

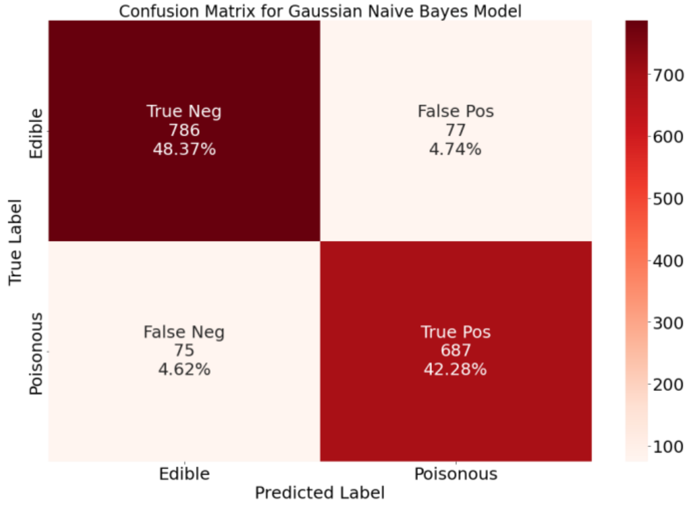


Figure 4 Naive Bayes Confusion Matrix

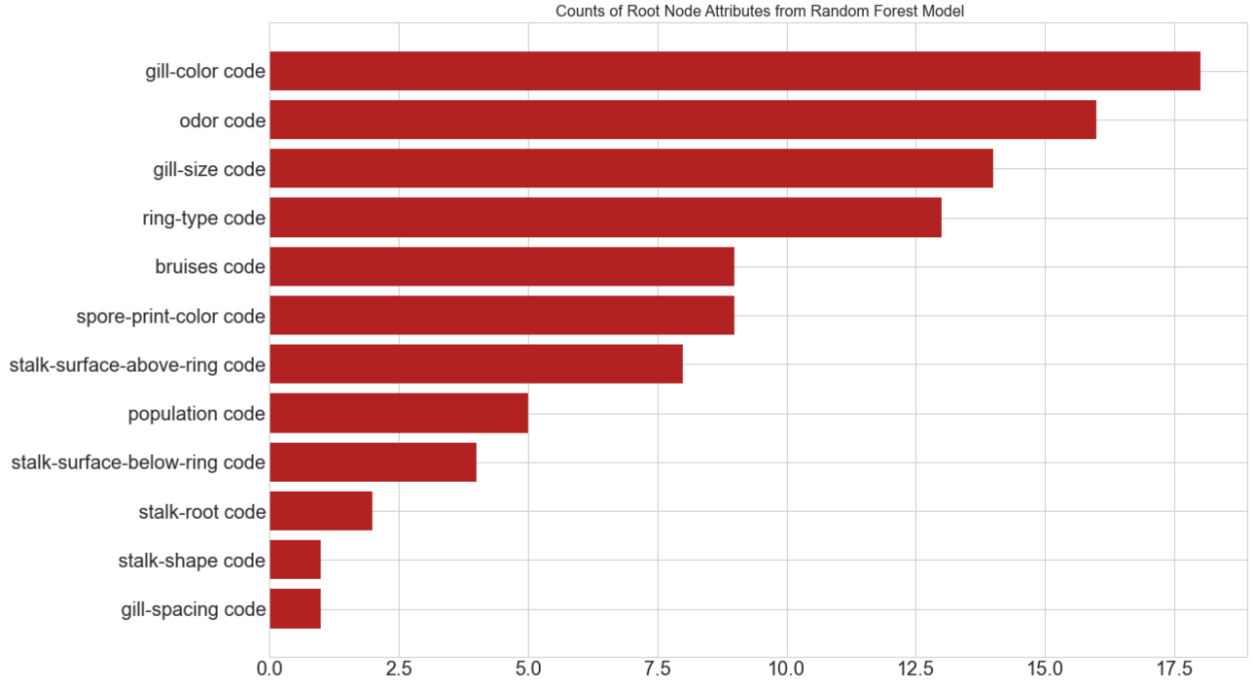
5 Discussion

Further comparison of all three models implemented allowed for exploration into efficiency and practicality of the best fit model for this task.

The Naïve bayes classifier was a good starting point to test out classification models. It was the fastest model to implement but also the least accurate. Some assumptions that were made to implement the algorithm may have contribute to the lower accuracy level. The Naïve bayes algorithm makes calculations assuming that predictors are independent of each other, which is not always (and most likely not in this case) true in real-world applications. Additionally, the specific type (Gaussian) of Naïve Bayes used is for data with continuous values. Although encoding of the attribute values made data continuous, the actual attribute values represent discrete values, additionally contributing to the error. In this situation, the worst-case scenario would involve making a false negative prediction (i.e. the mushroom is predicted to be edible when it is actually poisonous). In this model false negative claims were made 4.62% of the time. Considering the severity that making this mistake could lead to and that other models provide higher accuracy results; this model should not be selected as the best fit model.

The Decision Tree Classifier model appeared beneficial because it provides an easy to interpret visualization and can account for possible high correlation attribute relationships. Some of the general drawbacks to using this model were adjusted for to improve robustness. This includes its propensity to overfitting, which was taken care

Figure 5 Root Node Attribute Counts for Random Forrest



of by splitting up the dataset, and SciKit-learn's limitations of being unable to support categorical variables, which were subsequently encoded.[9] The main area for improvement lies in its costly nature due to the utilization of a greedy algorithm. Despite its greedy nature, the combination of high accuracy and time efficiency puts the Decision Tree Classifier as the best fit model for this task.

The Random Forest Classifier model was implemented to explore if overfitting could further be avoided through the addition of randomness and averaged combination of multiple different decision trees. By exploring the distribution of the root node attribute assignment, the highest count was assigned to "gill-color" (Figure 5). This provided additional confidence in the construction of the singular Decision Tree Classifier model previously. It's high accuracy results in combination with confidence gained from real-world simulated randomness put the Random Forest Classifier as a contender for the best fit model. However, with the addition of that confidence, some efficiency was sacrificed since training this model took twice as long than the Decision Tree Classifier. In terms of practical application, a faster model (with the same accuracy) is likely going to be more appealing for the end user.

Some potential challenges that arose during this experiment include an inability to identify a clear singular key characteristic in the Naïve Bayes Classifier to compare with the assignment of the other two models. Fortunately, since its accuracy performance was below the acceptance criteria, the focus was shifted towards the alternate approaches which could provide that information.

The bulk of the project proposal documentation was completed within 1 week. This includes the written report and slide deck capturing method and plan of the experiment. The next step of performing data mining and implementing all three models took roughly 1-2 weeks. This was done using python and a Jupyter notebook. Lastly, drawing conclusions and formalization of the project report was completed in 1 week. This entails finalizing the written report and performing presentation of findings.

6 Conclusion

Classification of edible and poisonous mushroom continues to be relevant as more species are discovered and more people become interested in foraging. Past researchers have taken different approaches to this problem, and this project has aimed to incorporate past knowledge with additional methods to build off previous work. Of the three models implemented, Decision Tree Induction was determined to be the best fit model for this task with an accuracy level of 100%, key attribute identification of "gill-color", and relatively fast and efficient implementation. Some room for improvement would be to continue work on implementing a more efficient algorithm that is not as greedy, and therefore takes less time to be trained. Additionally, confidence in this predictive model can continue to build as more data is gathered about additional species to retrain and retest. This will provide additional data points that will continuously increase robustness of the model.

REFERENCES

[1] Michael W. Beug, PhD, 2020, Summary of the Poisoning Reports in the NAMA Mushroom Poisoning Case Registry: 2018 through 2020, Retrieved July 28, 2022 from https://namyco.org/docs/2018-2020_Summary_of_Mushroom_Poisoning_Reports_Corrected_final.pdf

[2] V. Vanitha, M.N. Ahil, and N. Rajathi, 2020. Classification of Mushrooms to Detect their Edibility Based on Key Attributes *Biosc.Biotech.Res.Comm.* 13, 11 (December 2020) 37-41. DOI: http://dx.doi.org/10.21786/bbrc/13.11/9

[3] Kanchi Tank, 2020. Mushroom Classification Using Different Classifiers: Introduction to classification using Decision Tree, Logistic Regression, KNN, SVM, Naive Bayes, Random Forest Classifiers with Python (September 2020) Retrived July, 28, 2022 from https://medium.com/analytics-vidhya/mushroom-classification-using-different-classifiers-aa338c1cd0ff

[4] Lauren Foltz and Baskar Dakshin, 2018. Mushroom Classification. IST 707 Data Analytics Final Project. Syracuse University, Syracuse, NY. Retrieved July 28, 2022 from http://laurenfoltz.com/content/1-projects/1-data-mining-project/data-mining-final-report.pdf

[5] SciKit Learn. 1.10. Decision trees. Retrieved from https://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart

[6] SciKit Learn. Sklearn.ensemble.RandomForestClassifier. Retrieved from https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier

[7] SciKit Learn. Sklearn.naive_bayes.GaussanNB. Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html#sklearn.naive_bayes.GaussianNB

[8] Gandhi, Rohith, 2018. Naïve Bayes Classifier. (May 2018). . Retreived August 10,2022 from https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c#:~:text=A%20Naive%20Bayes%20classifier%20is,based%20on%20the%20Bayes%20theorem.

[9] IBM. Analytics: What is a Decision Tree? Retrieved from https://www.ibm.com/topics/decision-trees#:~:text=A%20decision%20tree%20is%20a,internal%20nodes%20and%20leaf%20nodes.