Milestone Report 2

Mushroom Classification

Sergio Robledo

December 3, 2019

Mushroom Foraging (Machine Learning)

We have completed and explained the data preprocessing and exploration steps done to the mushroom dataset in the first Milestone Report; in this report we will explore the steps needed to create a machine learning model capable of correctly classifying the edibility of mushrooms 100% of the time. For this part, we will rely heavily on the insights gained during the exploration phase of the project; more specifically, we will build a machine learning model that only uses the most important mushroom characteristics arrived upon after exploring the mushroom dataset.

Final Dataset

The dataset we will pass to the machine learning model will only use 6 of the 22 mushroom characteristics found in the mushroom dataset; these include the mushrooms' odor, spore print color, stalk surface below their rings, cap surface, cap color and gill size. Since machine learning models do not accept name categories as input, we will have to one-hot encode the columns; one-hot encoding involves creating a new column for each category of a characteristic and setting the column values to 1 if the characteristic is present and 0 otherwise. Below we can see the top five observations before and after one-hot encoding.

	odor
0	pungent
1	almond
2	anise
3	pungent
4	none

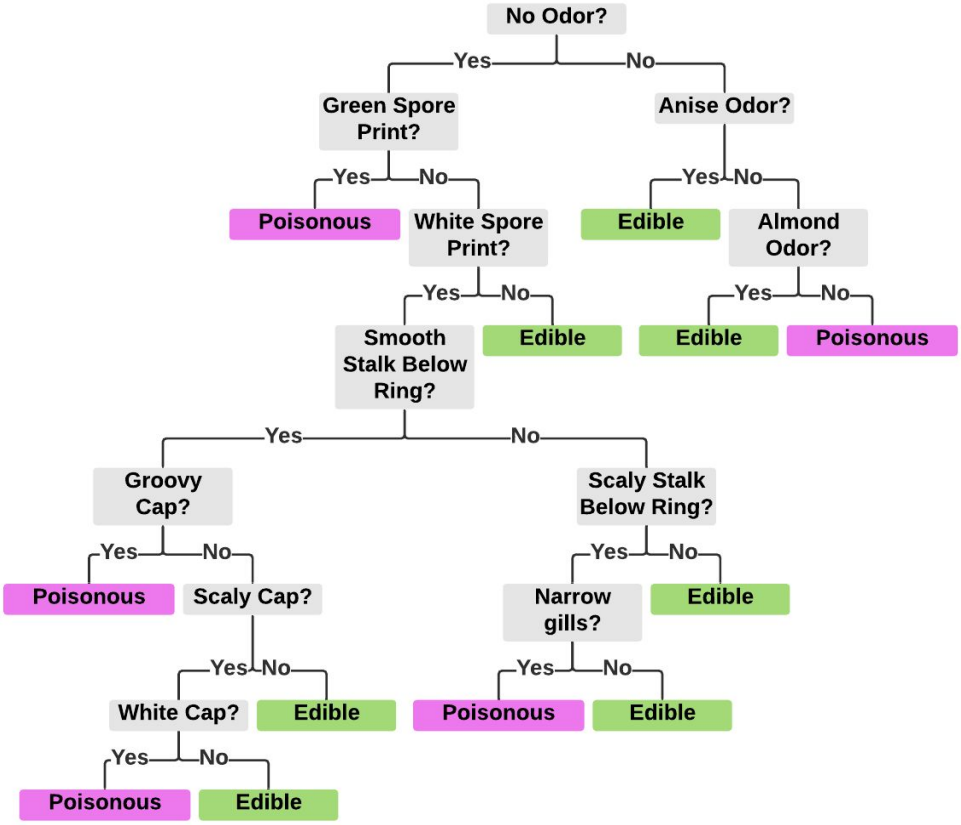
	odor_anise	odor_creosote	odor_fishy	odor_foul	odor_musty	odor_none	odor_pungent	odor_spicy
0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0
4	0	0	0	0	0	1	0	0

The left image shows the odor column of the original dataset, and the right image shows the one-hot encoded odor columns of the processed dataset; as it can be seen by comparing both datasets, the one-hot encoded dataset creates a column for every type of odor and marks a 1 if that type of odor was observed for a given mushroom, and since

there can only be one odor recorded per mushroom, only one column will contain a value of 1 and the rest will contain 0s for that row. Once we have one-hot encoded the original dataset containing the 6 important mushroom characteristics, we just need to split the one-hot encoded dataset into training and testing datasets; the training set will contain 75% of the mushrooms observed, and the test dataset will contain the remaining 25% observations which will not be used at all during the machine learning model's training phase. This ensures that the final performance achieved by the learning model is valid, because it confirms that the model is not cheating its way to a stellar test score by having seen the answers to the test dataset before the testing phase. The dataset is now ready, and the machine learning model can be created.

Machine Learning Model

When hypothesizing what a good starting classifier would be, the Random Forest Classifier came to mind, because it is made up of decision trees; decision trees begin by looking at one of the mushroom's characteristics and depending on what value is in that column, it moves on to what it considers the mushroom's next most important feature that will help with the classification of the mushroom. For example, from the exploration of the mushroom dataset we know that mushrooms with no odor are mostly edible with the exception of a few mushrooms that are poisonous, so just looking at the odor of an odorless mushroom is not enough to separate the edible and poisonous mushrooms; therefore, the task of the random forest classifier would be to find what features will help it move forward in the edibility classification of such mushrooms. Random forests are made up of multiple decision trees that each look at various combinations of mushroom characteristics to arrive at a classification, and by choosing the most popular classification among the decision trees, the random forest classifier is able to be a fairly accurate classifier; below is an example of what an optimal decision tree would look like if it used the most important characteristics found during the dataset's exploration.



If a random forest classifier contained this single decision tree, it would be able to correctly classify the edibility of mushrooms 100% of the time given the current dataset. Due to the Random Forest classifier's ability to handle one-hot encoded data very well, which is what the mushroom dataset exclusively contains, it was selected for the first draft of what was to become the final machine learning model. Not surprisingly, after training the scikit-learn's default Random Forest classifier, it was able to correctly classify the edibility of the testing dataset's mushrooms; in other words, the out of the box classifier was able to classify the edibility of 2,031 never-before seen mushrooms without a single misclassification. When trying to arrive at an optimal machine learning model, the best performing model with the lowest level of complexity should be chosen, and since scikit-learn's base Random Forest classifier was able to achieve the accuracy desired, 100% accuracy, there was no need to further fine-tune the model. If there had been a need to fine-tune the random forest classifier, the two most important parameters of the random forest are the number of estimators, a.k.a. number of decision trees, and the maximum number of features a single decision tree could choose from as its next

splitting node; as the number of estimators increase, the model's performance increases at the expense of efficiency and simplicity, and as the number of max features the model can consider for its next internal node decreases, its ability to generalize to new observations improves, because it is forced to learn from multiple features instead of the same ones over and over again. Scikit-learn's default Random Forest classifier sets the number of estimators to 10 the maximum number of features considered to the square root of the number of features in the dataset; even though only 6 mushroom features are really being used, because of one-hot encoding which uses a column for every category within a feature, the dataset passed to the model contains 32 features, and therefore, the maximum number of features the default random forest classifier can consider for each split is 5.

Conclusion

The most important lesson to be learned from the completion of this project is how far thorough exploration of the dataset can take you, because by taking the time to understand what mushroom characteristics separated edible and poisonous mushrooms the best, we were able to use a very simple machine learning model that required very little preparation; after exploring the mushroom dataset, we were able to reduce the number of features from 22 to 6, and the only step required by the random forest classifier was the one-hot encoding of the categorical dataset. After training the final random forest classifier on the training dataset, it achieved a 100% accuracy score on the remaining 2,031 mushrooms found in the testing dataset; no fine-tuning was needed since the default random forest configuration was able to achieve the desired accuracy score set at the beginning of the project. In the future, as more mushrooms are observed, an occasion may arise where the current features and model are no longer able to fully capture the growing complexity of the mushroom dataset; in the case such occasion occurs, a greater number of mushroom features or a more complex machine learning model will have to be used, or both.