Taylor Dredge
04-12-2024

Physical Characteristic Significance in the Identification of Poisonous Mushrooms

Abstract

This study uses various physical and measurable characteristics of a wild mushroom in machine learning classification algorithms to predict whether it is safe for consumption or not. In the study, various binary classification models are explored and compared. These include logistic regression, SVM, decision trees, random forest learning, AdaBoost, and gradient boosting models. Exploration is conducted into creating a model which is both simple for a user to input data and produces minimal false negative results. As a result, a random forest classification model was able to be constructed which accurately predicted the edibility of a given mushroom.

1. Introduction

In North America there are about 250 species of wild mushrooms which are poisonous. The severity of these toxins' effects on a person range from mild discomfort to in some cases organ failure and death. Unintentionally harvesting and consuming a poisonous mushroom account for roughly half of the severe mushroom poisoning cases in British Columbia (BCCDC, 2022). The symptoms caused from these poisonous mushrooms can be split into six groups: cytotoxic, neurotoxic, myotoxic, metabolic, gastrointestinal irritation, and miscellaneous adverse reactions (White et al., 2019). Cytotoxic and myotoxic poisonous mushrooms account for most of the lethal poisoning instances (Diaz, 2018). In 2021 the number of mushroom poisoning incidents in China from 25 provincial administrative divisions was 327. A total 923 patients were involved and resulted in 20 deaths leading to a 2.17% mortality rate (Li et al., 2022). Scientific knowledge of poisonous mushrooms has come as long way. We can now accurately identify known harmful mushrooms, how their different poisons effect humans, and how to effectively treat the poisonings (M-Q et al, 2022). Using a large dataset of various edible and poisonous mushrooms with descriptive characteristics, this paper explores the viability of different machine learning binary classification techniques to accurately predict if a mushroom is safe to consume. The paper then explores the viability of the models on inputs with more ambiguous mushroom characteristics removed. The false negative rate of a predictive model is important in health-related applications as in many cases, a mislabelled output can lead to dire health consequences for a person. In this case mislabelling a poisonous mushroom as edible can lead to serious health risks or even death if the

person consumes the mislabelled mushroom.

2. Data Description

The data used is from the UC Irvine machine learning repository and is named "Secondary Mushroom". The labelled data is useful for supervised learning techniques and consists of 20 variables/features a model can pull information form. These variables/features range from numeric values for measurable features, to categorical values such as colour and shape. The dataset is a collection of 61,068 hypothetical mushrooms with mushroom caps based on 173 species for a distribution of 353 mushrooms per species. Each mushroom was classified as "edible", "poisonous", "unknown", or "not recommended" which was reduced into a binary classification of edible and poisonous. Unknown and not recommended classifications were added to the poisonous class. Of the 20 features, 9 contain missing values (Wagner, D *et al.* (2021). Features with large amounts of missing data were removed as filling a large amount of missing datapoints would not improve a classification models accuracy and my induce bias. The remaining features with smaller amounts of missing values were filled using the popular and effective kNN imputer method (Brownlee, 2020). More detail on the specifics of each feature is described in Appendix A.

3. Background and Techniques

3.1 Classification Considerations

3.1.1 Imbalanced Datasets

For binary classification problems, an imbalanced dataset refers to a situation where one class's representation in the dataset is significantly more than the other. For example, the number of mushrooms labeled as *poisonous* compared to *edible* is outnumbered 5:1. Training models with this imbalance can lead to biases towards the majority class. These biases cause the model to struggle more when attempting to predict the minority class (Sivek, 2024). This is not an issue with this dataset as the distribution of poisonous and edible mushrooms is split 55-45.

3.1.2 Overfitting and Underfitting

When the model learns noise and outliers in the training data, it can negatively impact the models' performance on unseen data. The models' high sensitivity to fluctuations in the data creates a large variance error causing the model to be overfitted. On the contrary, if a model is too simplistic, it is unable to catch the overall trend of the data. It will have a large bias error and is considered underfitted (Sivek, 2024).

Handling of under and overfitting is a trade-off between the two. To avoid either situation, the dataset underwent normalization in the preprocessing stage. From a visual inspection using scatter plots no anomalies were detected. An outlier detection method was trialed, however the outliers detected appeared to represent a cluster of a specific species of mushrooms and was thus not used. In the

training stage, the models were trained with a 5-fold cross validation.

3.2 Models

The classification models are supervised algorithms fitted to a labeled dataset. Each of the models listed have various strengths and weaknesses which are explored and tested.

3.2.1 Logistic Regression

Logistic regression is a simple machine learning technique which estimates the probability of an event happening. The dependent variable is bound between 0 and 1 where the outcome is the output of a logit transformation applied to the regression's probability output. It is a type of logit model used for classification problems and is commonly used in binary classification problems where the prediction only has two possible outcomes (Ibm, *What is logistic regression?* 2024).

3.2.2 SVM

Support vector machines (SVM) classifies data by determining the best hyperplane of which separates all the datapoints into their respective classes. A best fit hyperplane is where the margin between the two classes is maximized (Mathworks, 2024).

3.2.3 Boosting Algorithms

Two boosting algorithms used in this study are the adaptive boosting (adaboost) method and the gradient boosting method. Boosting algorithms improve a models' performance by using an ensemble method where multiple weak learners are combined into a single strong model. Adaboost and gradient boosting differ in how they go about addressing the weak models' weakness.

Adaboost starts with giving the same weight to each data point. After every decision tree, the weights are adjusted, giving more weight to incorrectly classified datapoints to try and correctly predict them in the next iteration.

Gradient boosting works much in the same way as adaboost. However gradient boosting does not give incorrectly classified datapoints more weight and instead optimises the loss function. It does this by generating base learners iteratively, so that the current base learner is more effective than the previous (AWS, 2024).

3.2.4 Tree Algorithms

Finally, this study also implemented two tree-based classification models, the basic decision tree, and the random forest classifier. Decision trees splits up the data into multiple decision nodes. Each node's question helps guide the data to a leaf node which represents the class prediction. The tree attempts to find the best split to subset the data. They are trained with a classification and regression tree algorithm (CART). The Gini impurity is a common metric to evaluate the quality of the data split.

The ensemble learning of multiple trees is known as the random forest algorithm. The random forest algorithm extends the bagging method by including feature

randomness to try and create an uncorrelated forest of decision trees. By only using a select feature set per tree, the random forest algorithm reduces the risk of over and underfitting the data (Ibm, *What is Random Forest?* 2024).

4. Analysis and Results

4.1 Preprocessing

Data preprocessing is a strongly recommended step before the data analysis. Its goal is to transform the data into a format which is both easier and is more effectively processed by the machine learning model created. Visual plots such as histograms, scatter plots, and heatmaps help aid each of these preprocessing techniques.

Firstly, the dataset description lists 9 features which contained missing values. Using a pie plot to plainly view the percentage of missing values on each of the features revealed five of the nine features had a missing value rate of greater than 60%. Filling features with a high missing value percentage is risky as it can introduce bias into the models. With an abundance of features, a limit of 25% missing values was placed on the dataset features, which resulted in six of the nine features with missing values to be dropped from the dataset. The remaining features with missing values were run through a kNN imputer algorithm from the python sklearn library to predict the missing values of the remaining features. Using the kNN imputer is a popular and effective technique in handling missing values from a dataset (Brownlee, 2020).

Normalizing the numerical values to a set range has been shown to increase a machine learning models' performance overall. Some models do not perform well during the learning process when numeric data have different distributions (Google, 2024). The stem-width feature was converted from mm to cm to match the unit of measure of the other numeric features. These numeric features were then transformed into a range of 0.0 to 1.0.

After scatterplots were created for the comparison and distribution of numerical features. It was determined there was no need to eliminate the outliers found using the IQR method. The plots revealed a large cluster of edible mushroom data in the upper end of some of the numeric spectrum. The IQR outlier detection method had labeled this cluster as outliers, however the concentration and size of the cluster is not indicative of an anomaly and is more akin to a specific specie of mushroom. Removal of these datapoints could introduce underfitting in the models created and were thus kept in the dataset.

Finally, all categorical data was transformed using the one-hot encoding method which creates boolean features from their respective categorical feature values. The classification models used in this study do not work with string categorical features.

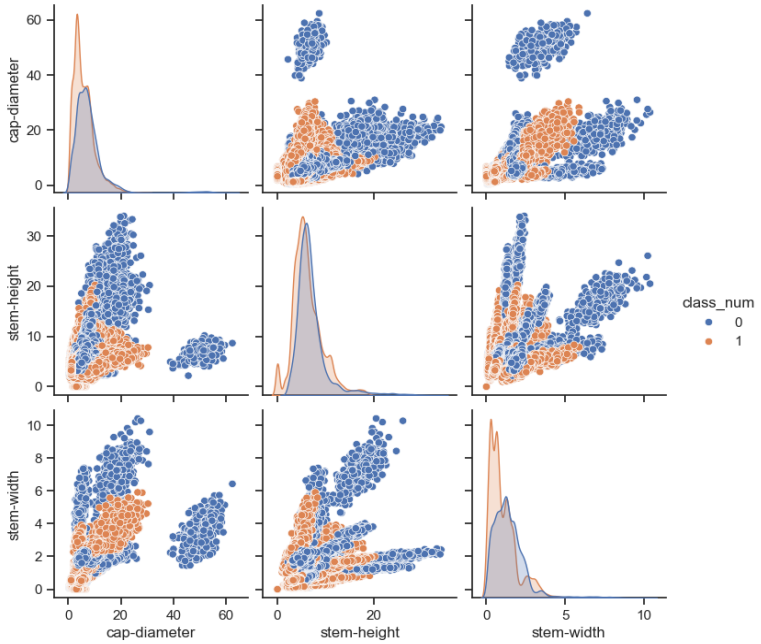

Figure 1. Seaborn pair plot of numerical data features. Class 0 and 1 represent edible and poisonous labels

Initial Analysis

A baseline of each model was trained on a 67-33% split of the dataset where two thirds of the data was used for training and the remaining third was used for testing. A 5-fold cross validation from the sklearn library in python used to train each model. This python class method returns a dictionary data structure with the models training time, scoring time, and scores.

The results in Table 1 show a strong performance from the tree algorithms with an average accuracy above 99%. The random forest classifier exabits a significantly longer training time when compared to the standard decision tree,

	Logistic Regression	**SVM**	**Adaboost**	**Gradient Boost**	**Decision Tree**	**Random Forest**
Avg Training Time (s)	0.29	17.17	2.26	6.77	0.29	2.82
Avg Score Time (s)	0.007	3.76	0.09	0.02	0.006	0.07
Accuracy (%)	77.17	98.93	78.1	91.14	99.59	99.90

Table 1. Initial Model 5-fold cross validation performance

however the accuracy of the random forest is 0.3% greater.

The next best performing model in terms of average accuracy, is the SVM at 98.9%. The SVM model however, take significantly longer than the other models to both train and score.

The boosting models follow the tree models performance with the gradient boosting model achieving an average accuracy score of 91% and adaboost at 78%. The logistic regression model was very fast to train and score, but the accuracy was the worst of the tested models.

Using the confusion matrix to plot the models correct and incorrect predictions, the random forest showed the least amount of false negative predictions. This is a good starting point for this study's goal in predicting if a mushroom is poisonous or not and minimizing the number of false negative predictions the model may make.

	Predicted False Label	Predicted True Label
False Label	9027	21
True Label	10	11095

Table 2. Random Forest Classifier Initial Confusion Matrix

4.2 Reducing Ambiguity

With 14 remaining features still in the dataset after preprocessing, there is risk in incorrectly labeling some of a mushrooms more ambiguous physical characteristics when entering feature data for a mushroom's edibility prediction. Features which rely on touch are subjective and can be interpreted differently between persons. An example is the mushroom cap surface texture. Some of the values for this feature include feeling leathery, smooth, silky, fleshy, etc. Accurately labeling a mushroom's texture is difficult, so in this section, these features are dropped from the dataset. Less ambiguous, but sill difficult so label features are ones of which are not common knowledge. The mushroom ring type, cap shape, gill attachment, and stem root type are also removed from the dataset.

The same analysis was conducted to measure the impact removing these features have on each models' performance. From the results in Table 3, all models except for the decision tree and random forest models, saw a significant drop in accuracy. The precise drop ranges, however it is generally about 7% lower compared to the full dataset. The tree models saw a slight drop in accuracy performance. In the next section, this study uses a technique to try and bring the impacted models accuracy scores into a competitive range again.

Model	**Accuracy (%)**
Logistic Regression	70.36
SVM	91.02
Adaboost	72.84
Gradient Boosting	83.43
Decision Tree	98.40
Random Forest	99.40

Table 3. Accuracy after ambiguous features removed

4.3 Hypertuning

With excellent performing decision tree-based models this study could end there and select the random forest model as a suitable model capable of being deployed in the public, however hypertuning offers the ability to better tune the models created to increase their performance in this study's use case. Hypertuning is the method of iteratively changing model specific parameters for the training process and comparing their scores against the previous scores.

From the sklearn library, the RandomizedSearchCV class was used to randomly select parameters from a given parameter set and run a 5-fold cross validation on the model. This is repeated for a specified number of iterations (100 in this study). The resulting scores and best parameters of the optimal model are returned for each model.

The results of the hypertuning on each of the models in Table 4 had varying success in improving a models accuracy score. Most of the models' accuracy scores were improved slightly or remained the same with the elimination of the six ambiguous features earlier. Due to the computation time of the SVM model, hypertuning was not performed on it. A noticeable improvement was seen from the gradient boosting models accuracy performance. The model saw a 16% increase in accuracy to match the random forest classifiers performance.

Model	**Accuracy (%)**
Logistic Regression	70.62
SVM	N/A
Adaboost	73.55
Gradient Boosting	99.40
Decision Tree	98.40
Random Forest	99.40

Table 4. Model accuracy after hypertuning

	Predicted False Label	Predicted True Label
False Label	8885	64
True Label	60	11144

Table 5. Random Forest confusion matrix after hypertuning

5. Visualization

Various plots were used in this study for exploratory analysis of the preprocessing stage and model performance analysis. In the preprocessing stage, this study benefited from the seaborn library's pairplot function. The function plots pairwise scatter and histogram plots in a grid to compare the distribution of numerical data. The distributions found from these plots allowed the study to determine that the outliers listed from the IQR outlier detection method were not anomalies and should not be removed from the dataset. A heatmap visual was also beneficial in the preprocessing step as this study used a modified version to plot only the highly correlated features. These features were easily discoverable for removal. And finally, bar and pie plots were used to view the distribution of missing values in features and explore the

distribution of categorical feature data between the class labels.

When evaluating each models' performance, bar plots and the confusion matrix plots were valuable tools in comparing the models' metrics. The confusion matrix shows a helpful overview on number of correct and incorrect predictions the model made on the test data. Viewing the distribution of true and negative predictions each model made is central in this classification problem as this study aims to minimize the false negative predictions a model may make in the field. In the code set, an area under the receiver operating characteristics curve (AUC-ROC) was created. On the three best performing models the curves appear as lines instead. Bar charts in the model creation stage was used to plot each models feature importance metrics. For all the models except the logistic regression model, the numeric features stem-width, stem-height, and cap-diameter were determined to be the most important features in the models' prediction decision.

Figure 2. Gradient Boosting models' learning curve

Finally, a learning curve and validation curve were used on the gradient boosting model. The learning curve displays the effect the data size has on the models' accuracy. In this study, the curve was linear, where more data samples increased the models' performance on unseen data. The validation curve shows a parameters effect on the models' performance. The visual is reflective of the hypertuning process in fine tuning the machine learning model to the problem from a given dataset.

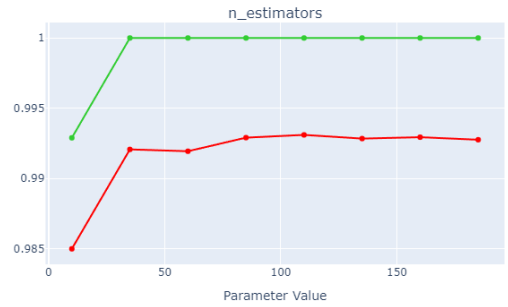

Figure 3. Gradient Boosting models' validation curve for the n-estimators parameter

6. Discussion

From the results of the model analysis, it was discovered that the random forest classification model and gradient boosting model had outperformed the others in all performance metrics. The logistic regression model was the least accurate followed by the adaboost model. The SVM model was found to be the slowest to train and slowest to test. Given the poor performance and long training

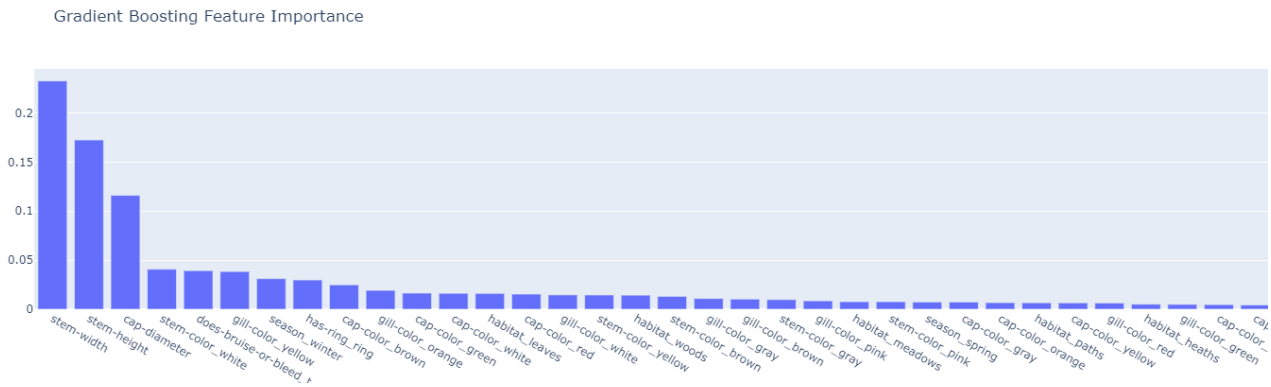

Figure 4. Gradient Boosting Hypertuned Feature Importance Chart

time, the SVM model was chosen to not undergo hypertuning given the available

high performing boosting and tree models.

The mushrooms cap diameter, stem width and height are shown to be significant explanatory factors in the high performing models prediction process. This was expected from the clearly shown cluster of edible mushroom-classed datapoints in the upper quadrant of the scatter plots involving these features. A mushroom with a large cap diameter to stem size was determined to be a strong indication that the mushroom is safe for consumption. From these feature distributions, poisonous mushrooms tend to occupy the small to medium size ranges.

	Logistic Regression	**SVM**	**Adaboost**	**Gradient Boosting**	**Decision Tree**	**Random Forest**
Accuracy (%)	70.1	91.0	73.6	99.4	98.4	99.4
Precision (%)	72.2	92.3	75.5	99.4	98.8	99.4
Recall (%)	75.3	91.3	77.6	99.4	98.2	99.5
False Positive Rate (%)	36.4	9.6	31.5	0.7	1.5	0.7
False Negative Rate (%)	24.7	8.7	22.4	0.5	1.8	0.5
F1 Score (%)	73.7	91.8	76.5	99.4	98.5	99.4

Table 6. Final model performance metrics

The categorical features distribution of edible and not edible mushrooms has less clear visible trends, however there are a few notable features which aided the decision process of the models. Certain mushroom cap colours lean heavily to the poisonous category. These include green, red, orange, and pink. The colour for other parts of the mushroom similarly showed specific colours were more indicative of a mushrooms' poisonous penitential.

Abstract and subjective features introduce potential user error when entering the information into a model. Error on the inputs will induce error on the output. To prevent this, features such as gill-attachment, and cap-surface (texture) were removed from the dataset. The removal of these features negatively impacted the performance of all model's accuracy with some more affected than others. The tree models saw minimal impact.

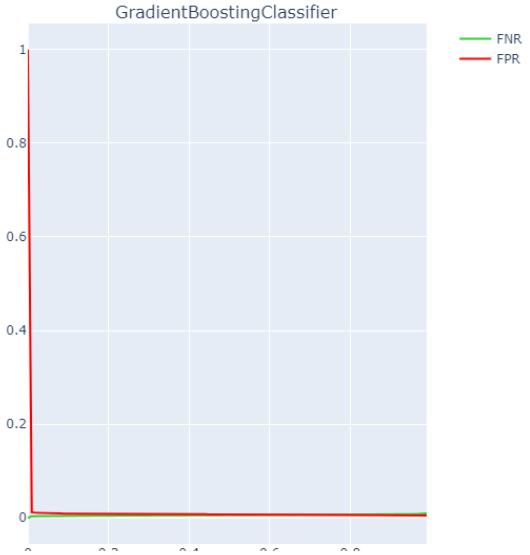

Figure 5. False negative and false positive curve

To increase the viability of the underperforming models, hypertuning was conducted on each model minus the SVM model for reasons mentioned earlier. The process is time consuming, however the results clearly show a large performance increase in the gradient boosting model. The increase in the decision tree and random forest algorithms were small minimal, which is expected as their accuracy were already greater than 98%.

The performance of each model was noted using common performance metrics which include accuracy, precision, recall, f1-score, and the false positive and negative rates. The false negative rate is particularly of interest as minimizing this metric will minimize the possibility the generated model will incorrectly label a poisonous mushroom as edible. This mislabelling has the possibility of hospitalization or even death if the mislabel mushroom is consumed. The decision threshold for each model was plotted against the false positive and false negative rates. The curves for the gradient boosting model are shown in Figure 5. These plots show the false negative rates can be reduced at the expense of the false positive rate. The gradient boosting models' curve shows the false negative rate can be improved with little impact on the false positive rate.

7. Conclusion

This study showed, given a set of physical and non-ambiguous mushroom characteristics, a model can be created which can accurately and reliably predict

the edibility of a wild mushroom picked when forging. Multiple supervised binary classification algorithms were tested resulting in some models producing near perfect prediction results on the test data after hyperparameter tuning. The gradient boosting model and random forest classifier model achieved an accuracy score of above 99%. However, the gradient boosting model can achieve a better false negative rate with little impact to the false positive rate by adjusting the prediction threshold. The performance was made possible by the large dataset as seen in the gradient boosting models learning curve.

In the future, a full grid search of a models' parameters during the hyperparameter tuning process could lead to improved results for other models, and other classification models could be explored which may perform better on training time. The model may also be expanded to more mushroom species found worldwide. And finally, it would be beneficial to conduct a similar study on a dataset of non-hypothetical mushroom data.

8. References

Wagner, D *et al.* (2021). Secondary Mushroom [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5FP5Q.

BCCDC. (2022). *Fact Sheet: Wild Mushrooms May Be Poisonous*. Retrieved 2022, from http://www.bccdc.ca/resource-gallery/Documents/Educational%20Materials/EH/FPS/Fruit%20and%20Veg/Wild%20Mushrooms%20May%20Be%20Poisonous%20Fact%20Sheet.pdf.

Google. (n.d.). *Numerical Data: Normalization | machine learning | google for developers*. Google. https://developers.google.com/machine-learning/crash-course/numerical-data/normalization

Brownlee, J. (2020) *KNN imputation for missing values in machine learning, MachineLearningMastery.com*. Available at: https://machinelearningmastery.com/knn-imputation-for-missing-values-in-machine-learning/ (Accessed: 04 December 2024).

Dadpour, B. *et al.* (2017) *Mushroom poisoning in the northeast of Iran; a retrospective 6-year epidemiologic study, Emergency (Tehran, Iran)*. Available at: https://pmc.ncbi.nlm.nih.gov/articles/PMC5325892/ (Accessed: 04 December 2024).

Diaz, J.H. (2018) 'Amatoxin-containing mushroom poisonings: Species, toxidromes, treatments, and outcomes', *Wilderness & Environmental Medicine*, 29(1), pp. 111–118. doi:10.1016/j.wem.2017.10.002.

He, M.-Q. *et al.* (2022) 'Potential benefits and harms: a review of poisonous mushrooms in the world', *Fungal Biology Reviews*, 42, pp. 56–68. doi:https://doi.org/10.1016/j.fbr.2022.06.002.

Ibm (2024a) *What is logistic regression?*, *IBM*. Available at: https://www.ibm.com/topics/logistic-regression (Accessed: 04 December 2024).

Ibm (2024b) *What is Random Forest?*, *IBM*. Available at: https://www.ibm.com/topics/random-forest (Accessed: 04 December 2024).

Sivek, S.C. (2024) *Mastering binary classification: A powerful predictive analytics tool*, *Pecan AI*. Available at: https://www.pecan.ai/blog/mastering-binary-classification-model-predictive-analytics/ (Accessed: 04 December 2024).

Train SVM classifiers using Gaussian Kernel (no date) *MathWorks*. Available at: https://www.mathworks.com/help/stats/support-vector-machines-for-binary-classification.html (Accessed: 04 December 2024).

What is boosting? - boosting in machine learning explained - AWS (no date) *What is Boosting?* Available at: https://aws.amazon.com/what-is/boosting/ (Accessed: 04 December 2024).

White, J. *et al.* (2019) 'Mushroom poisoning: A proposed new clinical classification', *Toxicon*, 157, pp. 53–65. doi:10.1016/j.toxicon.2018.11.007.

One hot encoding in machine learning (2024) *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/ml-one-hot-encoding/ (Accessed: 04 December 2024).

A.

Data Description

#	Feature Name	Feature Details (n: nominal, m: metrical)
1	cap-diameter (m)	float number in cm
2	cap-shape (n)	bell=b, conical=c, convex=x, flat=f, sunken=s, spherical=p, others=o
3	cap-surface (n)	fibrous=i, grooves=g, scaly=y, smooth=s, shiny=h, leathery=l, silky=k, sticky=t, wrinkled=w, fleshy=e4. cap-color (n)
5	does-bruise-bleed (n)	bruises-or-bleeding=t, no=f
6	gill-attachment (n)	adnate=a, adnexed=x, decurrent=d, free=e, sinuate=s, pores=p, none=f, unknown=?
7	gill-spacing (n)	close=c, distant=d, none=f
8	gill-color (n)	see cap-color + none=f
9	stem-height (m)	float number in cm
10	stem-width (m)	float number in mm
11	stem-root (n)	bulbous=b, swollen=s, club=c, cup=u, equal=e, rhizomorphs=z, rooted=r
12	stem-surface (n)	see cap-surface + none=f
13	stem-color (n)	see cap-color + none=f
14	veil-type (n)	partial=p, universal=u
15	veil-color (n)	see cap-color + none=f
16	has-ring (n)	ring=t, none=f
17	ring-type (n)	cobwebby=c, evanescent=e, flaring=r, grooved=g, large=l, pendant=p, sheathing=s, zone=z, scaly=y, movable=m, none=f, unknown=?
18	spore-print-color (n)	see cap color
19	habitat (n)	grasses=g, leaves=l, meadows=m, paths=p, heaths=h, urban=u, waste=w, woods=d
20	season (n)	spring=s, summer=u, autumn=a, winter=w
21	Class label (n)	edible=e, poisonous=p

B.

Hypertuned Models

Best Parameters

Model	Best Parameters
Logistic Regression	solver='liblinear', penalty='l1', C=61.0
SVM	N/A
Adaboost	n_estimators=220, learning_rate=2.0
Gradient Boosting	n_estimators=90, max_depth=9, loss='exponential', learning_rate=1.7000000000000002
Decision Tree	splitter='best', min_samples_split=4, min_samples_leaf=2, max_depth=25, criterion='entropy'
Random Forest	n_estimators=140, max_depth=64, criterion='gini', bootstrap=False