Mushroom Classification

1 Introduction

In this project, we developed a binary classification model to predict whether a mushroom is edible from its physical characteristics. We used F-score and AUC to compare logistic regression, random forest, boosting, k-NN and neural network models. We also used logistic regression and random forest to identify the most useful predictors. The best models were k-NN and neural network. The most useful predictors were gill size, bruises, and gill spacing.

2 Dataset Description

The dataset describes 8,124 mushrooms, 52% of which were poisonous (UCI Machine Learning, 2016). All the 9 predictors were nominal, and are described in the table below.

Table 1: Descriptions of each variable

Variable	Type	Description
Edibility (Target variable)	**Binary**	**2 categories: edible, poisonous**
Bruises	Binary	2 categories: bruises, no bruises
Gill size	Binary	2 categories: broad, narrow
Stalk shape	Binary	2 categories: enlarging, tapering
Cap surface	Nominal	4 categories: fibrous, grooves, scaly, smooth
Gill attachment	Nominal	4 categories: attached, descending, free, notched
Veil color	Nominal	4 categories: brown, orange, white, yellow
Habitat	Nominal	7 categories: grasses, leaves, meadows, paths, urban, waste, woods
Gill spacing	Ordinal	3 categories: crowded, close, distant
Number of rings	Ordinal	3 categories: zero, one, two

Mushroom anatomy: The gills of a mushroom produce and release spores to reproduce. The cap protects the gills. The stalk raises the gills so that the spores can be dispersed by the wind.

Data Collection: The observations are hypothetical samples generated from 23 species of mushrooms. However, we do not know the species of each observation, or how the samples were generated.

3 Methods

Encoding categorical variables: We converted each nominal variable into an unordered factor, and each ordinal variable into an ordered factor.

Decreasing dataset size: Since our models were taking too long to train, we decreased the dataset size to 1,000 observations (500 edible mushrooms, 500 poisonous mushrooms).

Exploratory data analysis:

• **Predictor-target distributions:** For each predictor, we plotted a proportional stacked bar plot for edible and poisonous mushrooms. We then used these plots to identify the predictor categories most associated with poisonous mushrooms.

• **Clustering:** We clustered the unique observations in the dataset. Since our data was categorical, we couldn't use k-means to cluster the data. Instead, we used Partitioning Around Medoids (PAM). The best number of clusters had the largest average silhouette width.

• **Dimensionality Reduction:** We used dimensionality reduction to try to separate the edible and poisonous mushrooms in a lower-dimensional space. Since our data is categorical, we couldn't use Principal Component Analysis (PCA) for dimensionality reduction. Instead, we used Multiple Correspondence Analysis (MCA).

Train-test split: We split the dataset into an 80% training set and a 20% test set. This is necessary because we want to estimate each model's performance on data it was not trained on (the test set).

Hyperparameter tuning: All the models had hyperparameters, which are parameters that control the model's complexity. We used grid search and 5-fold cross-validation to approximate the best hyperparameters (using the "caret" package). For example, if we test 10 sets of hyperparameter values using 5-fold cross validation, then we obtain 5 validation set errors for each set of hyperparameter values. The best set of hyperparameter values has the lowest average validation set error.

• **Logistic regression:** Logistic regression assumes a linear relationship between the log odds of the positive class and the predictors. It can handle categorical predictors using one-hot encoding. We used elastic-net regularization to prevent overfitting. Elastic-net adds a lasso penalty term and a ridge penalty term to the objective function, which limit the magnitudes of the coefficients.

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_q x_q$$

$$\underset{\boldsymbol{\beta}}{\operatorname{argmin}}\left(RSS + \lambda \sum_{j=1}^{q}\left[\alpha\beta_j^2 + (1-\alpha)|\beta_j|\right]\right)$$

where p is the probability of being poisonous, RSS is the residual sum of squares, λ is the regularization parameter, and α is the balance between ridge and lasso.

• **Random Forest:** A decision tree splits the predictor space into regions using recursive binary splitting, and each region is assigned a predicted class. Since a single decision tree has high

variance, a random forest returns the average of a set of classification trees, where each tree is trained on a bootstrap sample of the training set. To decrease the correlation between the trees, each split only considers a subset of the predictors. The probability of being poisonous is the proportion of trees that predict poisonous.

$$\hat{f}(\boldsymbol{x}) = B^{-1}\sum_{b=1}^{B}\hat{f}^{*b}(\boldsymbol{x})$$

where B is the number of trees.

• **Boosting:** Boosting is an iterative algorithm that trains a decision tree on the residuals of the previous iteration's model, then adds a multiple of the decision tree to the model. We can tune the number of trees, the size of each tree (interaction depth), and the learning rate.

$$\hat{f}^{(i)}(\boldsymbol{x}) = \hat{f}^{(i-1)}(\boldsymbol{x}) + \lambda\hat{f}^{b}(\boldsymbol{x})$$

where λ is the learning rate.

• **k-NN:** k-Nearest Neighbors (k-NN) is a non-parametric method, meaning it makes no assumptions about the shape of the decision boundary. k-NN classifies each observation using the target variable categories of its k nearest training set observations in the predictor space. A larger k will return a smoother decision boundary. The probability of being poisonous is the proportion of neighbors that are poisonous.

• **Neural network:** Since this is a binary classification problem, the output layer contained 1 neuron with a sigmoid activation function. For simplicity, we only considered one hidden layer. More neurons in the hidden layer will increase the model variance, while a larger weight decay will decrease the model variance.

Probability threshold tuning: Since the models return the predicted probability of poisonous, we had to choose the best probability threshold. To do so, we calculated the test set F-score for a grid of probability thresholds, then returned the maximum F-score.

Model Evaluation:

• **F-score:** We evaluated the models by comparing their test set F-scores. The best model has the largest F-score (0 ≤ F-score ≤ 1). The F-score is the harmonic mean of a model's precision and recall. A model with high precision has few false positives (incorrectly predicted to be poisonous). Meanwhile, a model with high recall has few false negatives (incorrectly predicted to be edible).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad \text{F-score} = \frac{2}{\text{Precision}^{-1} + \text{Recall}^{-1}}$$

• **AUC:** To compare the models across all probability thresholds, we also compared their AUC (Area Under Curve) values. The best model has the largest AUC value (0.5 ≤ AUC ≤ 1) .

Model Interpretation: For some of the models, we were able to calculate predictor category

importances to determine which ones were the most useful for predicting edibility.

• **Logistic regression:** The exponentials of the fitted coefficients are the change in the odds of being poisonous compared to the baseline category.

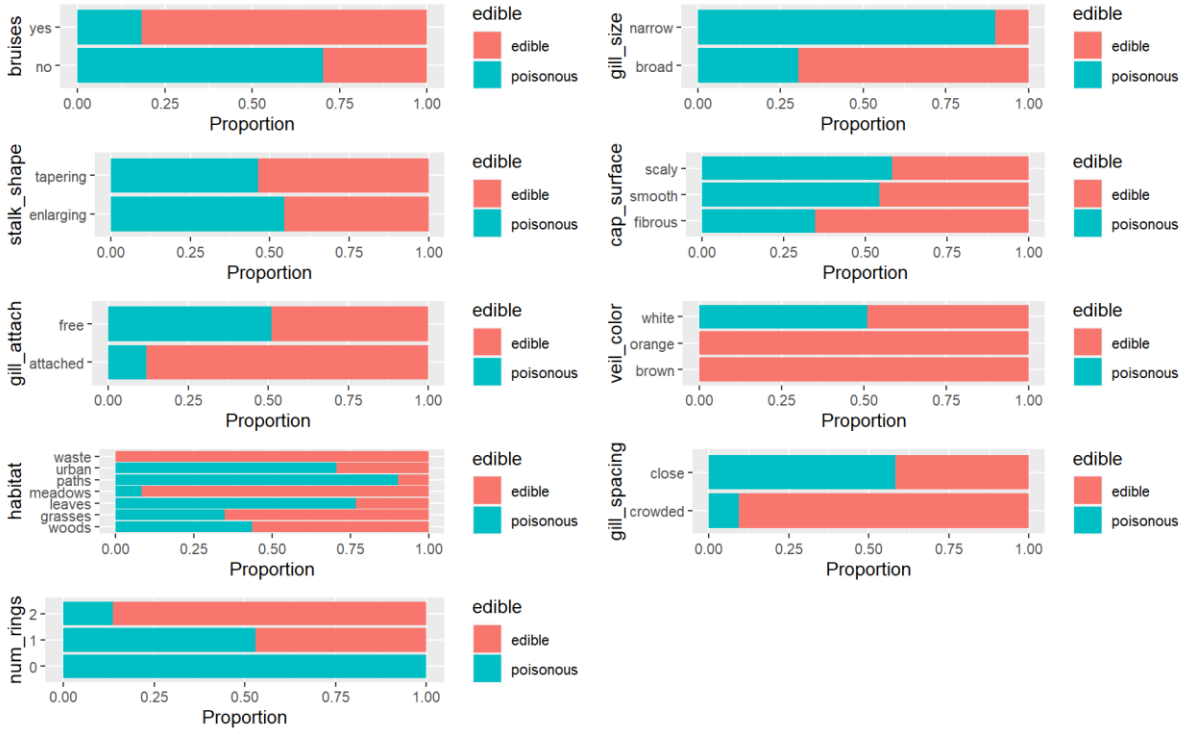
• **Random Forest:** The Gini index measures the purity of a node in a decision tree. A smaller Gini value indicates higher purity. Each split in a decision tree decreases the Gini index of the children nodes relative to the parent node. The larger the average decrease in Gini index for all splits using predictor A, the more useful predictor A is.

4 Results

4.1 Exploratory data analysis

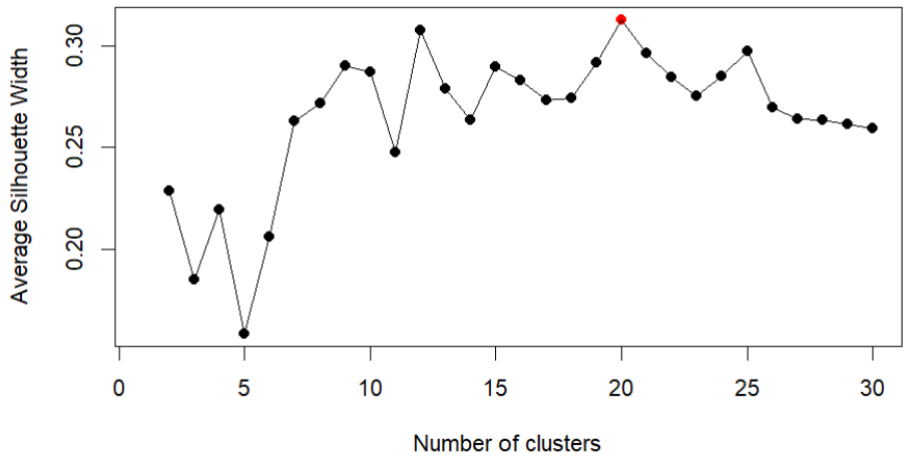
• **Proportional bar plots:** From the figure below, we observed that the three predictor categories with the largest proportion of edible mushrooms were: habitat(waste), veil_color(orange), veil_color(brown). Similarly, the three predictor categories with the largest proportion of poisonous mushrooms were: num_rings(0), gill_size(narrow), habitat(paths).

Figure 1: Proportion of poisonous mushrooms within each category of the predictors



• **Clustering:** The dataset of 1,000 observations only contained 53 unique observations. We used PAM to cluster these 53 observations. In the figure below, we see that the best number of clusters was 20 (red dot).

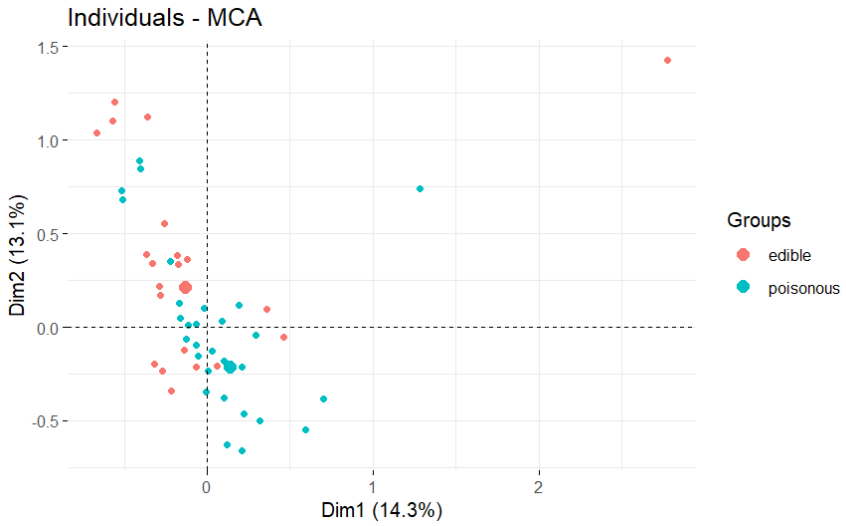
Figure 2: Partitioning around medoids clustering



6 of the 20 clusters only contained poisonous mushrooms (calculations in Appendix, Section 2: Exploratory Data Analysis). Another 6 clusters only contained edible mushrooms. The remaining 8 clusters contained a mix of edible and poisonous mushrooms. We assume that mushrooms within the same species have the same edibility. Hence, some of our clusters may not correspond to species.

• **Dimensionality Reduction:** In the figure below, we see that MCA was able to separate some of the edible (upper left) and poisonous (lower right) mushrooms. The first two components only accounted for 27.4% of the variance in the data.

Figure 3: Multiple correspondence analysis dimensionality reduction

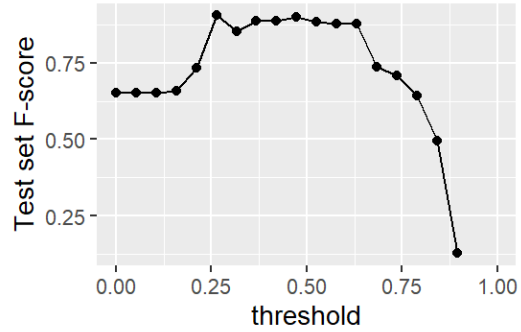


4.2 Main data analysis

Probability threshold tuning: In the figure below, we observe that for logistic regression, the 0.26 probability threshold yielded the largest test set F-scores. The lower the threshold, the

more mushrooms that are classified as being poisonous.

Figure 4: F-scores for logistic regression

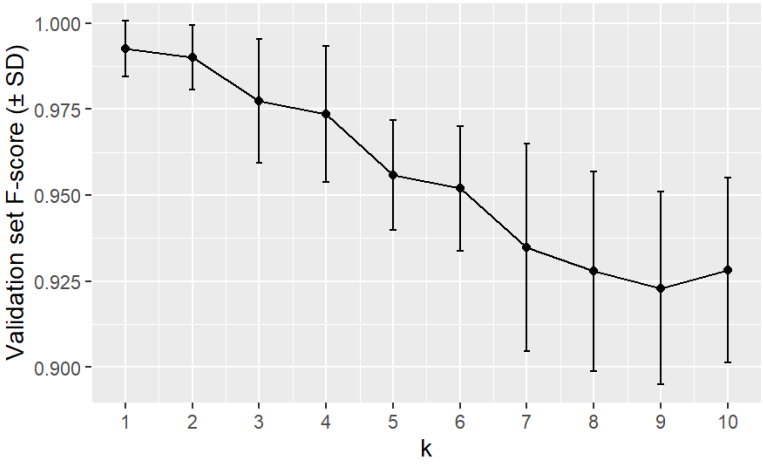


To save space, we excluded the other threshold plots.

Hyperparameter tuning: For models with 1 hyperparameter, we plotted a validation curve. For models with 2 hyperparameters, we plotted a validation heatmap. For models with more than 2 hyperparameters, it is difficult to visualize the grid search.

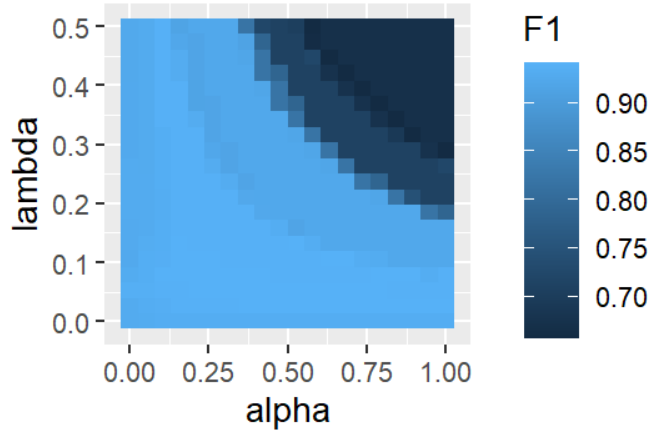
In the figure below, we observe that k-NN with k = 1 yielded the largest average validation set F-score. We believe this is because the dataset only contains 53 unique observations. Hence, for many observations, the nearest neighbor is an identical observation that has the same values for all predictors.

Figure 5: Validation curve for a model with 1 hyperparameter (k-NN)



In the figure below, we observe that logistic regression with $\alpha = 0.16$ and $\lambda = 0.13$ yielded the largest average validation set F-score. Models with large α and large λ have a low F-score because they underfit the data. This is because a larger α removes more predictors (lasso regularization) and a larger λ penalizes the coefficient magnitudes more.

Figure 6: Validation heatmap for a model with 2 hyperparameters (logistic regression)

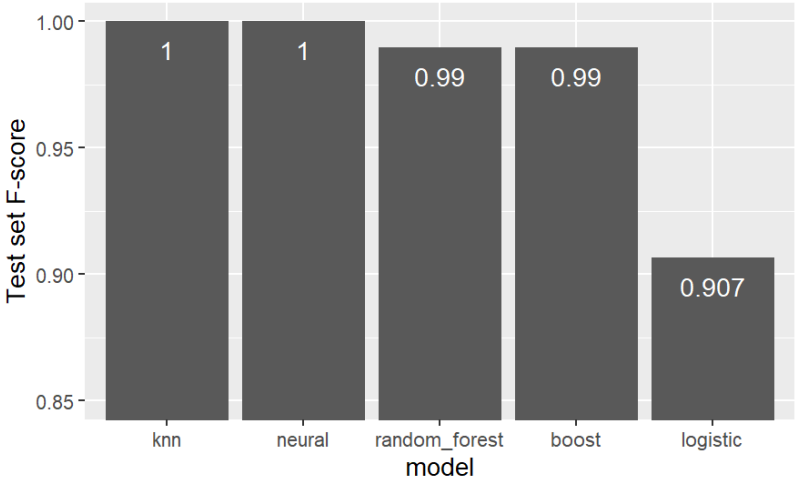


For the other validation plots, refer to the Appendix, Section 3: Statistical Models.

Model Evaluation:

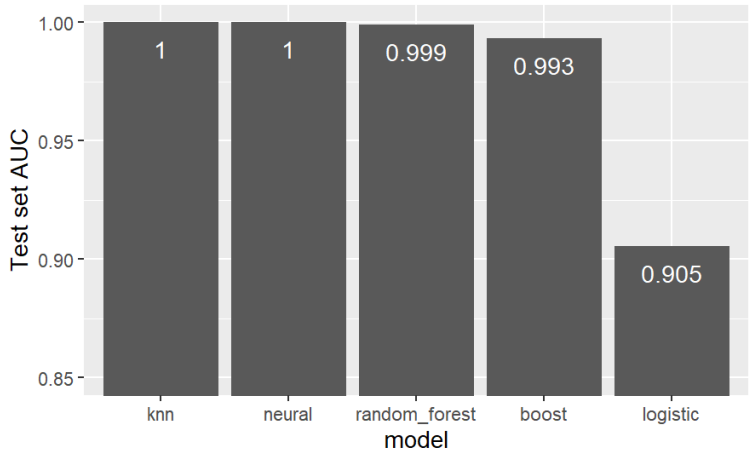
• **F-score:** In the figure below, we observe that k-NN and neural network obtained perfect test set F-scores. This means that these models correctly classified each observation in the test set. Meanwhile, logistic regression obtained the smallest F-score. This suggests that the data is not linearly separable.

Figure 7: Test set F-scores of each model



• **AUC:** In the figure below, we observe that k-NN and neural network also achieved perfect test set AUCs. This means that across all probability thresholds, these models had zero false negatives (incorrectly predicted to be edible).

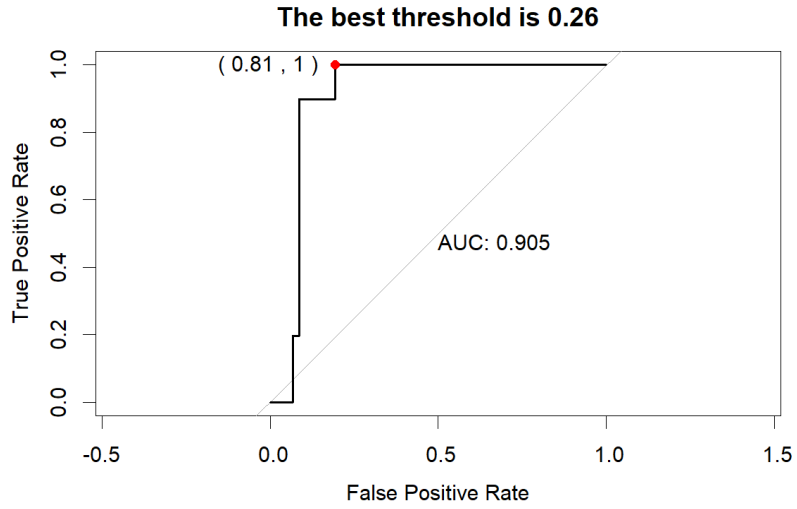
Figure 8: AUC for each model



$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad \text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

In the figure below, we see how the true positive rate and false positive rate change with the probability threshold. The red dot corresponds to the best probability threshold of 0.26, which we identified in Figure 4 (F-scores for different probability thresholds). This threshold had a false positive rate of 0.81 and a true positive rate of 1.0.

Figure 9: ROC Curve for logistic regression

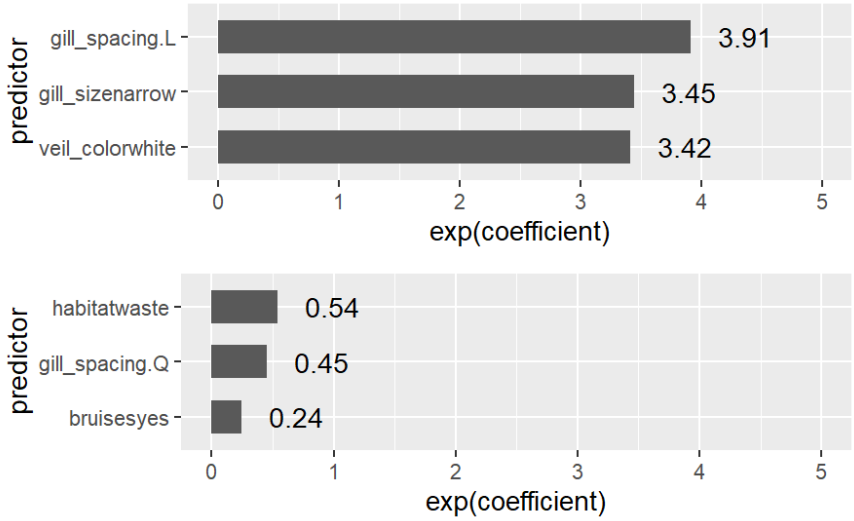


Model Interpretation:

• **Logistic regression**: In the figure below, we observe that compared to the baseline category of gill size(broad), gill size(narrow) increased the odds of being poisonous by 291%. This agrees with our observation in Figure 1 (proportional bar plots) that 90% of the mushrooms with gill size(narrow) were poisonous, but only 31% of the mushrooms with gill size(broad) were

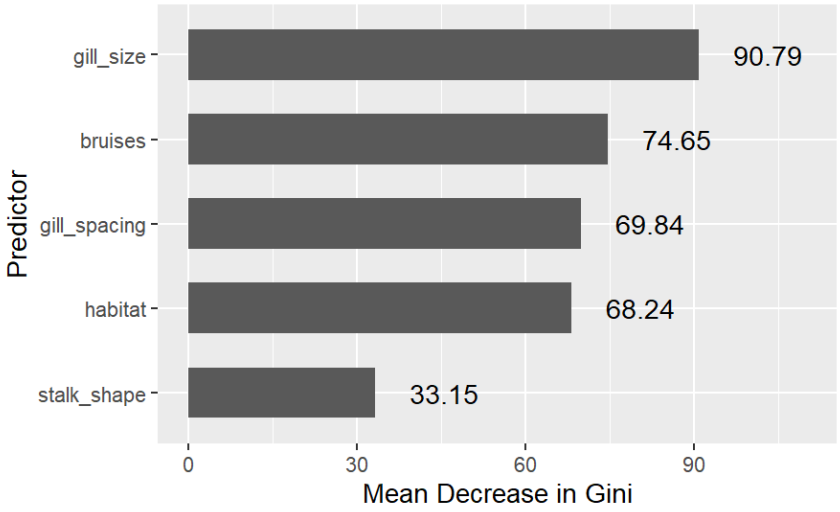
poisonous. Similarly, compared to the baseline category of bruises(no), bruises(yes) decreased the odds of being poisonous by 76%.

Figure 10: The predictor categories with the three largest (top) and three smallest (bottom) logistic regression exp(coefficient) values



• **Random forest**: In the figure below, we observe that the most influential predictors were gill_size, bruise, and gill_spacing.

Figure 11: The five predictors with the largest random forest importances



The partial dependence probability (PDP) for predictor A, category 1 is calculated by setting the value of predictor A to category 1 for all observations. The PDP is the average predicted probability of poisonous across all observations. In the table below, we see that bruises(no) and gill_size(narrow) have a larger PDP. This agrees with the proportional bar plots in Figure 1.

Table 2: Random forest partial dependence probabilities of poisonous

	Bruises(no)	Bruises(yes)
Gill_size(broad)	0.61	0.21
Gill_size(narrow)	0.72	0.62

5 Conclusion

The best classification models were k-NN and neural network, which both achieved a perfect F-score and a perfect AUC. Furthermore, logistic regression and random forest both indicated that the most important predictors were gill_size, bruises, and gill_spacing. The categories gill_size(narrow), bruises(no), and gill_spacing(close) all increased the probability of being poisonous.

Limitations and suggestions:

• **Predictor Selection:** It may be possible to achieve a perfect F-score using fewer predictors. Since we calculated the predictor importances, we could use recursive feature elimination to iteratively remove the least important predictor and re-train the best performing classification model.

• **Data Representativeness:** Given that the dataset was limited to samples from 23 species, our results may not generalize to other mushroom types. In future studies, we could include more diverse species to increase the model's robustness across different mushroom species.

6 References

UCI Machine Learning. (2016). Mushroom Classification. Kaggle.
https://www.kaggle.com/datasets/uciml/mushroom-classification/data

7 Appendix

R Code Sections
1 Data Preparation
2 Exploratory Data Analysis
3 Statistical Models
4 Model Performance
5 Model Interpretation