edX Data Science: Capstone Course - Prediction of Mushroom Edibility Project

If a mushroom smells bad - don't eat it!

Elizabeth Plasse Dominguez

October 11, 2022

Overview

The goal of this Capstone Course - Prediction of Mushroom Edibility Project was to create two different machine learning models from the same mushroom data that predict whether a mushroom is edible or not and to evaluate the accuracy, sensitivity and specificity of prediction of the models.

The mushroom data for this project was the UC Irvine (UCI) Machine Learning Repository Mushroom Dataset[1] and comprised 23 categorical variables. "This data set includes descriptions of ... 23 species of gilled mushrooms in the Agaricus and Lepiota Family ..."[2]. Globally, there are over 14,000 species of mushrooms with about 13,000 gilled mushroom species so this mushroom data set was a small sample of the population.

Predicting the edibility of mushrooms is not easy. People who forage for mushrooms acquire a specific knowledge base in order to successfully gather edible mushrooms. They also generally use the rule "When in doubt throw it out!" with regard to edibility. Additionally, the UCI website page for the mushroom data cautions "The Guide [... Audubon Society Field Guide to ... Mushrooms] clearly states that there is no simple rule for determining the edibility of a mushroom ..."[2].

In light of these warnings and the small sample size, the prediction models resulting from this data should not be considered applicable to mushrooms in general and mushrooms should only be consumed when their edibility is known with certainty.

The data was downloaded, cleaned and manipulated into data frame training and test sets and one-hot coded matrix training and test sets.

Data investigation, analysis and visualization of the training set data, detailed in the Methods section, led to setting "edible" as the positive class and to the training of two prediction models - a regularized logistic regression model and a random forest classification model. The trained models were used to make predictions from the test set data. Due to the serious consequences of eating a poisonous mushroom the typical rule for classifying prediction probabilities as 1 for probabilities over 0.5 was made stricter by requiring prediction probabilities to be greater than 0.99 in order to be set equal to 1.

Both models predicted the edibility of mushrooms in the test data with high accuracy. The random forest model predicted with 100% accuracy. Sensitivity and specificity also both equaled 1. The regularized logistic model had an accuracy of 99.9%, a sensitivity of 1, a specificity of 0.997 and misclassified two edible observations as non-edible (false negatives). Perhaps most important neither model classified non-edible mushrooms as edible - no false positives. See the following summary.

Models	Accuracy	Sensitivity	Specificity	False Positives	False Negatives
Regularized Logistic	0.999	1.000	0.997	0	2
Random Forest	1.000	1.000	1.000	0	0

Methods

Data Exploration and Cleaning

The mushroom data download from the UCI website had no header. Column names and factor levels for all variables were provided on the UCI website page. See **Appendix** at end of this report for a list of variables and their levels (unused levels are not reported).

Data from the UCI Mushrooms data set was downloaded into a data frame object named *all_mushroom_dat*, column names were added, and all variables were made into factors. Here is a summary of the 8124 rows and 23 columns of *all_mushroom_dat*.

```
'data.frame':   8124 obs. of  23 variables:
 $ is_edible               : Factor w/ 2 levels "e","p": 2 1 1 2 1 1 1 1 2 1 ...
 $ cap_shape               : Factor w/ 6 levels "b","c","f","k",..: 6 6 1 6 6 6 1 1 6 1 ...
 $ cap_surface             : Factor w/ 4 levels "f","g","s","y": 3 3 3 4 3 4 3 4 4 3 ...
 $ cap_color               : Factor w/ 10 levels "b","c","e","g",..: 5 10 9 9 4 10 9 9 9 10 ...
 $ does_it_bruise          : Factor w/ 2 levels "f","t": 2 2 2 2 1 2 2 2 2 2 ...
 $ odor                    : Factor w/ 9 levels "a","c","f","l",..: 7 1 4 7 6 1 1 4 7 1 ...
 $ gill_attachment         : Factor w/ 2 levels "a","f": 2 2 2 2 2 2 2 2 2 2 ...
 $ gill_spacing            : Factor w/ 2 levels "c","w": 1 1 1 1 2 1 1 1 1 1 ...
 $ gill_size               : Factor w/ 2 levels "b","n": 2 1 1 2 1 1 1 1 2 1 ...
 $ gill_color              : Factor w/ 12 levels "b","e","g","h",..: 5 5 6 6 5 6 3 6 8 3 ...
 $ stalk_shape             : Factor w/ 2 levels "e","t": 1 1 1 1 2 1 1 1 1 1 ...
 $ stalk_root              : Factor w/ 5 levels "?","b","c","e",..: 4 3 3 4 4 3 3 3 4 3 ...
 $ stalk_surface_above_ring: Factor w/ 4 levels "f","k","s","y": 3 3 3 3 3 3 3 3 3 3 ...
 $ stalk_surface_below_ring: Factor w/ 4 levels "f","k","s","y": 3 3 3 3 3 3 3 3 3 3 ...
 $ stalk_color_above_ring  : Factor w/ 9 levels "b","c","e","g",..: 8 8 8 8 8 8 8 8 8 8 ...
 $ stalk_color_below_ring  : Factor w/ 9 levels "b","c","e","g",..: 8 8 8 8 8 8 8 8 8 8 ...
 $ veil_type               : Factor w/ 1 level "p": 1 1 1 1 1 1 1 1 1 1 ...
 $ veil_color              : Factor w/ 4 levels "n","o","w","y": 3 3 3 3 3 3 3 3 3 3 ...
 $ ring_number             : Factor w/ 3 levels "n","o","t": 2 2 2 2 2 2 2 2 2 2 ...
 $ ring_type               : Factor w/ 5 levels "e","f","l","n",..: 5 5 5 5 1 5 5 5 5 5 ...
 $ spore_print_color       : Factor w/ 9 levels "b","h","k","n",..: 3 4 4 3 4 3 3 4 3 3 ...
 $ population              : Factor w/ 6 levels "a","c","n","s",..: 4 3 3 4 1 3 3 4 5 4 ...
 $ habitat                 : Factor w/ 7 levels "d","g","l","m",..: 6 2 4 6 2 2 4 4 2 4 ...
```

Initial exploration of *all_mushroom_dat* showed the data to be quite clean. It contained no NAs and no empty cells. Two of the independent variables, *veil_type* and *stalk_root*, however, stood out.

veil_type had one level with 8124 observations of "p". This variable had no variability so *veil_type* was eliminated from the data.

stalk_root had 5 levels with one marked "?" which contained 2480 missing observations. Elimination of this missing data was accomplished by removing the *stalk_root* column rather than the 2480 observation rows. This minimized the loss of data for analysis and the potential of skewing the sample by removing data from other independent variables.

The elimination of these two independent variables lead to creation of a data frame object *mushroom_dat.* This data frame contained 8124 rows and 21 columns. The dependent variable, *is_edible*, was a binary factor and was in the first column of this data frame. The 20 remaining independent variables were all factors and were in columns 2 - 21. The levels of the factors ranged from 2 to 12.

mushroom_dat was randomly partitioned into training (80%) and test (20%) sets.

Data Frame	Number of Rows	Number of Columns
mushroom_training_set	6498	21
mushroom_test_set	1626	21

The training and test sets contained:

- the dependent variable *is_edible,* and
- 20 independent variables: *cap_shape, cap_surface, cap_color, does_it_bruise, odor, gill_attachment, gill_spacing, gill_size, gill_color, stalk_shape, stalk_surface_above_ring, stalk_surface_below_ring, stalk_color_above_ring, stalk_color_below_ring, veil_color, ring_number, ring_type, spore_print_color, population, habitat.*

The following summary of the distribution of classes of the dependent variable in the training and test sets shows they are quite similar.

Dependent Variable (*is_edible*) Factor Levels	Percent of Dependent Variable in Training Set	Percent of Dependent Variable in Test Set
e (edible)	51.8%	51.8%
p (non-edible)	48.2%	48.2%

Analysis and Visualization

Plots showing the breakdown of edible and non-edible observations by factor level for each of the 20 independent variables in *mushroom_training_set* are presented below in Figure 1.

These plots provided insight into which of the independent variables may be important predictors. The Odor, Spore Print Color, Gill Color and Population variables stand out as potential good predictors. These four features have some levels with only edible or only non-edible observations and also have other levels with a majority of observations in one class. The Odor variable in particular seems to predict edibility well as observations of the dependent variable in all levels except one (the level "none") were either edible or non-edible.

Alternatively, the plots of the Cap Surface, Gill Attachment, Stalk Shape, Veil Color and Ring Number variables show that they may not be good predictors. Their breakdowns of observations by levels are close to half edible and half non-edible and are similar to just guessing or flipping a coin.

In addition the plots of the Stalk Surface Above Ring and Stalk Surface Below Ring variables look very similar to each other as do the plots of the Stalk Color Above Ring and Stalk Color Below Ring variables. This suggests there is a high degree of collinearity between these pairs of variables.

Figure 1: Observations of Dependent Variable by Levels of Independent Variables

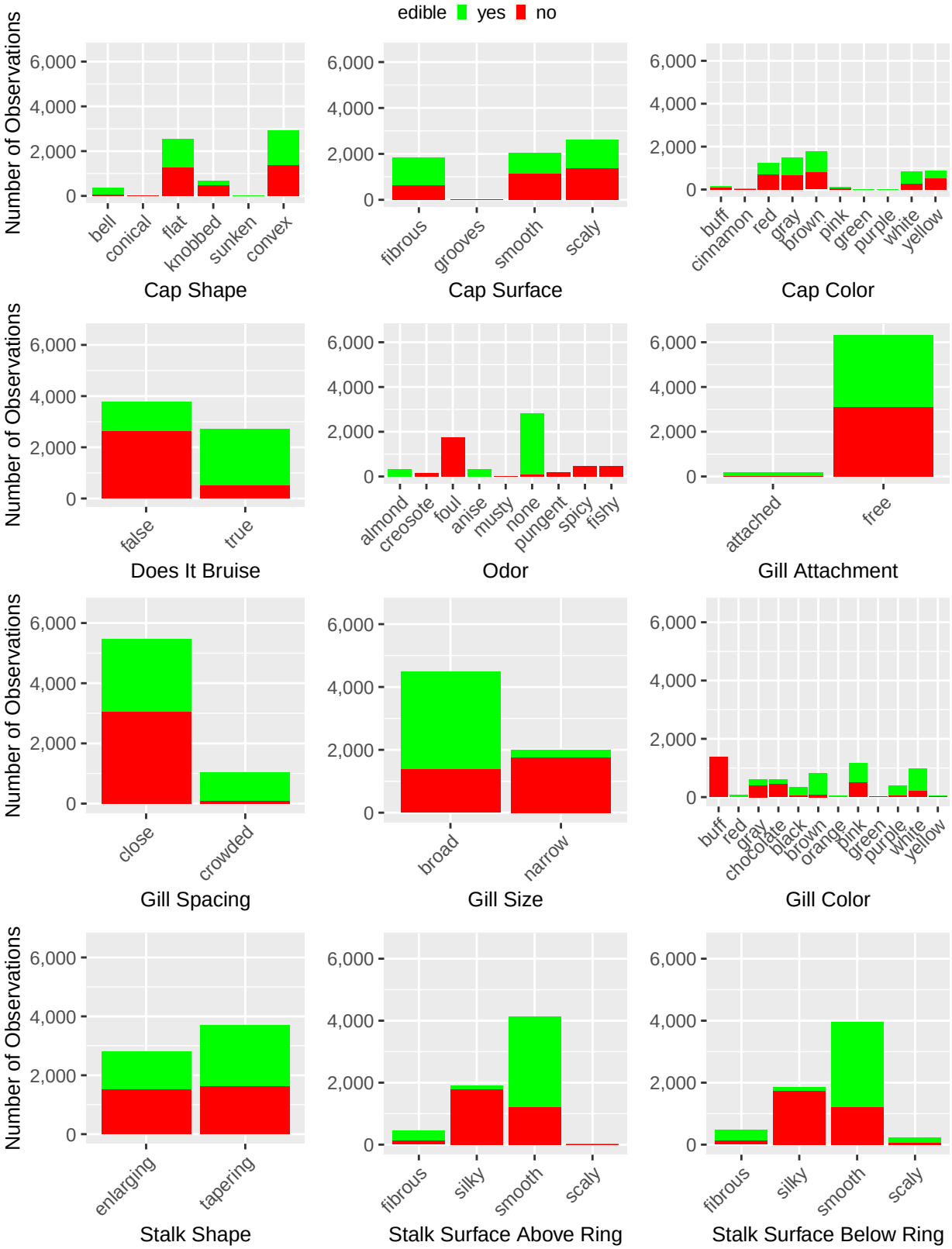
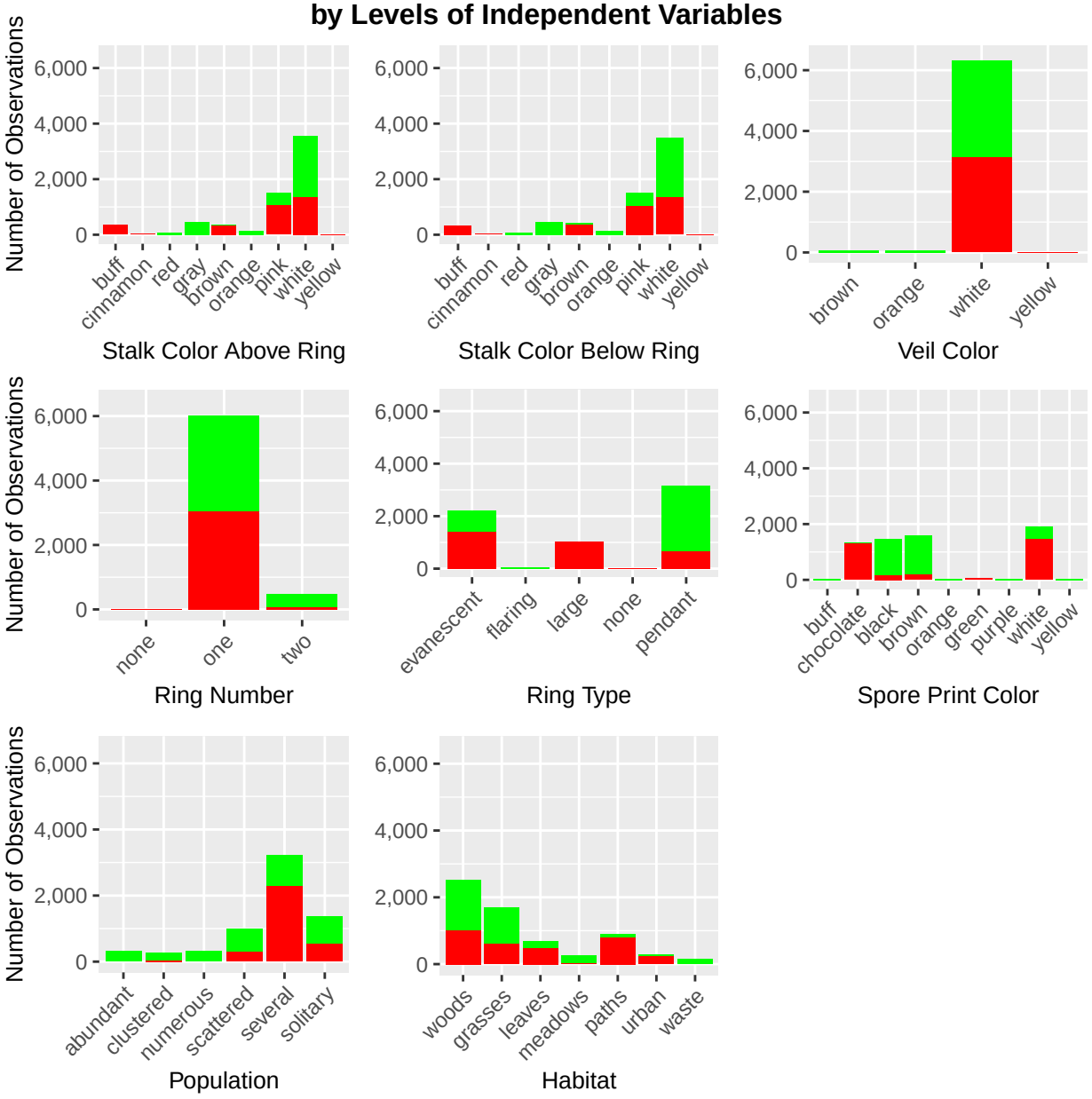


Figure 1 Continued: Observations of Dependent Variable by Levels of Independent Variables



Cramér's V[3] was chosen to investigate the level of association between the independent variables in *mushroom_training_set*. Cramér's V is a statistic that ranges from 0 to 1 and measures the association between two nominal[4] variables. Judging the level of association characterized by Cramér's V depends upon the number of degrees of freedom that exist in the comparison.

For this project the Cramér's V calculations between independent variables had 1 degree of freedom which was determined by taking the minimum of the number of rows minus one (6498 - 1 = 6497) and the number of columns minus one (2 -1 = 1). With one degree of freedom, pairs of independent variables with Cramér's Vs of 0.5 and above have medium to strong associations, while those with lower results have small associations.

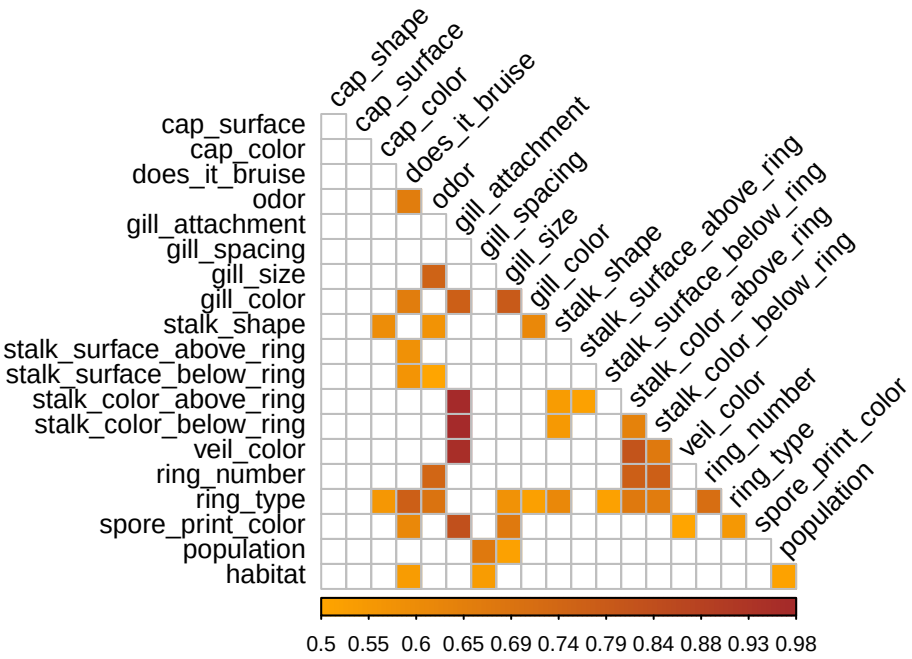
There is plenty of evidence of medium to strong association between the independent variables in the training set data as is shown in the Figure 2 below. In fact 90% of the independent variables had medium to strong associations. *cap_color*, *gill_spacing*, and *stalk_surface_above_ring* had medium to strong of association with two other independent variables while *ring_type* had it with eleven other independent variables. *gill_attachment*, *stalk_color_above_ring*, *stalk_color_below_ring*, and *veil_color* all have very strong associations with other independent variables.

Additionally, Figure 2 shows a strong relationship between *stalk_color_above_ring* and *stalk_color_below_ring* which showed very similar plots of breakdown of dependent variable observations by independent variable level in Figure 1. *stalk_surface_above_ring* and *stalk_surface_below_ring* also had very similar plots of breakdown of dependent variable observations by independent variable level in Figure 1 however they do not appear on the Cramér's V plot because their Cramér's V of 0.49 is just below the 0.5 cut off.

Two independent variables, *cap_shape* and *cap_surface*, had Cramér's Vs with the other independent variables that were all below 0.5 and thus small association with them.

Figure 2 shows the medium to strong associations in the independent variables with a color scale changing from orange to brown as the Cramér's V increases from 0.5 to just under 1. The diagonal of Cramér's V equal to 1 for a variable with itself has been omitted from the plot. Note also the two blank columns on the far left for the two independent variables with Cramér's Vs below the cutoff for the plot.

Figure 2: Association Between Independent Variables
Cramér's V of 0.5 or greater



Approach and Design for Models

Since *mushroom_training_set* has a binary dependent variable with only two classes and nominal categorical independent variables, logistic regression and random forest algorithms were chosen as the two models for training. Both of these machine learning methods perform well when predictioning with categorical data.

The data frames *mushroom_training_set* and *mushroom_test_set* which contain 21 categorical variables as factors were used for training and prediction ability testing the random forest model.

The matrices *x_train_glmnet*, *y_train_glmnet*, *x_test_glmnet*, and *y_test_glmnet* were used for training and prediction testing the regularized logisitc model. These matrices were created by one-hot encoding the *mushroom_training_set* and *mushroom_test_set* data frames and then manipulating the placement of the dependent variable. *x_train_glmnet* and *x_test_glmnet* contain the resulting 111 one-hot coded features created from the 20 independent variables in *mushroom_training_set* and *mushroom_test_set*. *y_train_glmnet* and *y_test_glmnet* contain the one-hot coded dependent variable created from the dependent variable in *mushroom_training_set* and *mushroom_test_set*.

The positive class for model training and prediction was chosen to be "edible".

Logistic Model

Logistic regression is a linear regression of the log odds of the dependent variable and is used to model a binary response variable. The dependent variable of *mushroom_training_set*, y_i for the i-th observation, is categorical and binary with its positive class equal to "e" for edible. The independent variables of *mushroom_training_set* in the vector X_i are all nominal and categorical.

A logistic model finds coefficients (the betas), that produce probabilities of $y_i = 1$ for given predictor variables x_i, which do the best job of accurately classifying the observed data points. The log odds transformation is used to assure that the output probabilities will be between 0 and 1. Additionally, logistic regression uses the log likelihood (L) to estimate the coefficients and probability distribution (via maximum likelihood estimation) that best explain the data.

The logistic model is defined by:

$$logit(p(X_i)) = log(\frac{p(X_i)}{1 - p(X_i)}) = \beta_0 + \beta_1 x_{i1} + ... + \beta_k x_{ik}$$

where:

$X_i = (x_{i1}, ..., x_{ik})^T$ are the k predictor variables for the i-th observation,

$p(X_i)$ is the probability that y_i equals 1, given X_i, and

$\beta_0, ..., \beta_k$ are the model coefficients.

The model minimizes the log likelihood function over the coefficients.

$$L = -log(\prod_{i:y_i=+} p(X_i) \prod_{j:y_j=-} (1 - p(X_j)))$$

where:

L is the log likelihood function

$\prod_{i:y_i=+}$ is the product over i for the positive group of y_i,

$\prod_{j:y_j=-}$is the product over j for the non-positive group of y_j.

Training of *mushroom_training_set* on this logistic regression model resulted in a model that did not converge and had singularities. These are signs of the multicollinearity and potential perfect separation in the independent variables as seen in the Analysis and Visualization section Figure 2.

Logistic regression with LASSO (least absolute shrinkage and selection operator) regularization can be helpful to produce a model that adjusts for multicollinearity and perfect separation. The regularization introduces bias while reducing variance. LASSO regularization adds a penalty term to the log likelihood function of the logistic regression model. This penalty is intended to reduce the coefficients of less significant independent variables to zero, reducing the effect of multicollinearity. It uses this log likelihood with LASSO penalty function.

$$L + \lambda(|\beta_0| + |\beta_1| + |\beta_2| + ... + |\beta_k|)$$

where:

L is the log likelihood function,

λ is a model parameter selected to minimize the out of sample error of the model, and

$\beta_0, ..., \beta_k$ are the model coefficients.

Training for a LASSO model required the training data to be in matrix format so *mushroom_training_set* and *mushroom_test_set* were one-hot encoded and these matrices were created:

Matrix	Number of Rows	Number of Columns
x_train_glmnet	6498	111
y_train_glmnet	6498	2
x_test_glmnet	1626	111
y_test_glmnet	1626	2

Additionally, cross validation was implemented to determine the best λ to use with the CV LASSO model when making predictions in order to minimized the out of sample error.

CV LASSO model training on *x_train_glmnet* and *y_train_glmnet* and cross validation was performed using the cv.glmnet function with:

- "family" set to "binomial", and
- "alpha" set to 1 (calls LASSO regularization).

The trained CV LASSO model converged with no singularities and found coefficients for 26 of the 111 one-hot coded independent variables. Here are the model selected variables with fitted log odds coefficients (including intercept):

	CV_LASSO_Coefficients
(Intercept)	-6.9
cap_shape_s	1.0
cap_surface_f	0.1
cap_surface_g	-3.8
cap_color_c	1.5
cap_color_n	0.2
cap_color_w	-0.8
odor_a	13.4
odor_f	-3.8
odor_l	13.4
odor_n	11.7
gill_size_b	4.3
gill_size_n	0.0
gill_color_w	-0.1
stalk_surface_above_ring_k	-2.9
stalk_surface_below_ring_y	-4.6
stalk_color_above_ring_o	1.1
stalk_color_below_ring_o	0.0
stalk_color_below_ring_y	-2.1
ring_number_t	1.6
ring_type_f	1.7
spore_print_color_r	-17.0
spore_print_color_u	0.1
spore_print_color_w	-0.8
population_c	-4.3
population_y	0.1
habitat_w	1.2

The signs of these coefficients match well with the plots of the independent variables in the Analysis and Visualization section Figure 1. For example, the odor variable coefficients for odor_a and odor_l are positive and their columns in the plot of the Odor variable show only edible observations. While the coefficient for odor_f is negative and its column in the plot of the Odor variable show only non-edible observations. It is interesting to note that large coefficients for odor_a, odor_f, odor_l, odor_n, spore_print_color_r, and population_c seem to agree with intuition from the Analysis and Visualization section that Odor, Spore Print Color and Population variables might be good predictors. However no work was done to determine the significance of these coefficients.

The model also produced a very low mean cross-validated error (binomial deviance) of 0.002 which is a sign that the model fits the data well.

CV LASSO model prediction was performed using the predict function with:

- *cv_lasso_model*,
- *x_test_glmnet* (the test data features),
- "s" set to "lambda.min",
- "type" set to "response".

Lambda.min determined by the cross validation was used to produced probabilities for all observations. The strict rule of using 0.99 as the hurdle for positive prediction was implemented and probabilities were then converted to predictions of 1 if they were higher than 0.99 or 0 if they were not. These predictions were then compared to *y_test_glmnet* (test data results) and confusion matrix statistics were calculated.

Random Forest Model

Random forest is a nonlinear ensemble machine learning method for classification/prediction of binary categorical variables which:

- takes bootstrapped (random with replacement) samples from the training data and builds a tree for each sample,
- considers a random subset of the independent variables at each node of each decision tree,
- constructs many decision trees from the training data,
- considers the class of votes, positive or negative, from all trees for each observation, and
- selects the output class for each observation based upon which class has the majority of the considered votes from all trees.

For this project, given the high cost of making a wrong prediction and eating a poisonous mushroom, the selection of output class for each observation is based upon which class has greater than 0.99 of the considered votes from all trees (note "cutoff" argument setting below). The random forest model was first trained on the data frame *mushroom_training_set* using the randomForest function with default settings of:

- "ntree" set to 500,
- "mtry" set to 4, and
- "cutoff" set to (0.99, 0.01).

This model had an out-of-bag error of 2.69% and misclassified 175 edible observations as non-edible. A plot of the model error as a function of number of trees was consulted and a minimum of error was evident at about 300 trees. Tuning of mtry was performed manually using the tuneRF function. An optimal mtry of 8 resulted.

Using this tuned mtry a random forest model was trained on the data frame *mushroom_training_set* using the randomForest function with:

- *is_edible* from *mushroom_training_set* as the dependent binary variable and positive class set to "edible",
- the 20 other independent variables from *mushroom_training_set* as features for training,
- "ntree" set to 500,
- "mtry" tuned and set to optimal 8, and
- "cutoff" set to (0.99, 0.01).

The trained model had a low 0.12% out-of-bag estimate of error with only 8 false negatives predicted (edible mushrooms predicted to be non-edible).

One way to look at variable importance in the trained model involves the mean decrease accuracy. Mean decrease accuracy is an indicator of the reduction in model accuracy if that variable is removed from the model. Larger values of mean decrease accuracy indicate more important variables. Here are the model variables in decreasing order of mean decrease accuracy:

	MeanDecreaseAccuracy
odor	0.301
spore_print_color	0.093
gill_color	0.034
stalk_surface_above_ring	0.027
gill_size	0.019
stalk_color_below_ring	0.016
ring_type	0.015
habitat	0.014
population	0.013
ring_number	0.012
gill_spacing	0.010
stalk_surface_below_ring	0.008
stalk_shape	0.006
cap_color	0.005
does_it_bruise	0.005
stalk_color_above_ring	0.004
cap_surface	0.002
cap_shape	0.001
gill_attachment	0.000
veil_color	0.000

odor, *spore_print_color, gill_color* and *stalk_surface_above_ring* had the highest mean decrease accuracies. Once again odor seems to be an important predictor for mushroom edibility.

Random forest model prediction was performed using the predict function with:

- *random_forest_model*, and
- *mushroom_test_set* (test data independent variables).

This produced predictions which were then compared to the actual dependent variable observations in the test data (*mushroom_test_set$is_edible*) and confusion matrix statistics were calculated.

Results

When modeling the prediction of mushroom edibility the serious consequences of eating a poisonous mushroom argue for a very strict rule of what gets classified as edible. The rule of accepting probabilities above 0.5 as positive is not appropriate and therefore the very constrained rule of accepting probabilities over 0.99 was used. In the logistic model this was accomplished when model generated probabilities were converted to predictions. In the random forest model this was accomplished through the use of the randomForest function argument "cutoff".

Logistic Model

CV LASSO model test set prediction results:

Accuracy	0.999
Sensitivity	1
Specificity	0.997
False Positives	2
False Negatives	0

The CV LASSO model's accuracy was near perfect when predicting from the test set data and no false positive classifications were predicted. Only two observations were incorrectly classified - 2 edible mushrooms were classified as inedible (false negatives). The model's sensitivity, the ability to predict positive outcomes (edible), was 1 and it's specificity, the ability to predict negative outcomes (non-edible), was 0.997.

Random Forest

Random Forest model test set prediction results:

Accuracy	1
Sensitivity	1
Specificity	1
False Positives	0
False Negatives	0

The Random Forest model's prediction from the test set data was excellent with accuracy, sensitivity and specificity all 1. The model predicted all observations correctly.

Conclusion

The goal of this project was accomplished. Two prediction models of the edibility of mushrooms were successfully constructed using the UCI Mushroom data set. One model utilized the random forest algorithm, the other, regularized logistic regression. Both models were very accurate predictors of the test set data. The random forest model predicted perfectly on the test set - no misclassifications. The regularized logistic model correctly predicted 99.9% of the test data and only misclassified 2 edible mushrooms as non-edible. There were no classifications of non-edible mushrooms as edible in either model which would of course be a serious problem.

One of the limitations of this work is that little was done to determine the role of individual variables as predictors. The random forest model produced the mean decrease accuracies of the variables and these were presented. Coefficients from the regularized logistic model were presented but without understanding the significance of each. Better understanding of the role of each variable in prediction is an area for potential future work which may lead to models with fewer features.

References

1. https://archive-beta.ics.uci.edu/ml/datasets/mushroom
2. Quote from website page in reference 1.
3. Cramér, Harald. 1946. *Mathematical Methods of Statistics.* Princeton: Princeton University Press, Page 282 (Chapter 21. The two dimensional case).
4. A nominal variable is a type of categorical variable where the information in the variable's levels have no numeric value.

Appendix

Table of Mushroom Variables with Levels

Dependent Variable (1)

is_ edible
- e – edible
- p – non-edible

Independent Variables (22)

cap_ shape
- b – bell
- c – conical
- f – flat
- k – knobbed
- s – sunken
- x – convex

cap_ surface
- f – fibrous
- g – grooves
- s – smooth
- y - scaly

cap_ color
- b – buff
- c – cinnamon
- e – red
- g – gray
- n – brown
- p – pink
- r – green
- u – purple
- w – white
- y - yellow

does_ it_ bruise
- f – false
- t - true

odor
- a – almond
- c – creosote
- f – foul
- l – anise
- m – musty
- n – none
- p – pungent
- s – spicy
- y - fishy

gill_ attachment
- a – attached
- f - free

Independent Variables Continued

gill_ spacing
- c – close
- w - crowded

gill_ size
- b – broad
- n - narrow

gill_ color
- b – buff
- e – red
- g – gray
- h – chocolate
- k - black
- n – brown
- o - orange
- p – pink
- r – green
- u – purple
- w – white
- y - yellow

stalk_ shape
- e – enlarging
- t - tapering

stalk_ root
- ? – missing
- b – bulbous
- c – club
- e – equal
- r - rooted

stalk_ surface_ above_ ring
- f – fibrous
- k – silky
- s – smooth
- y - scaly

stalk_ surface_ above_ ring
- f – fibrous
- k – silky
- s – smooth
- y - scaly

stalk_ color_ above_ ring
- b – buff
- c – cinnamon
- e – red
- g – gray
- n – brown
- o - orange
- p – pink
- w – white
- y - yellow

stalk_ color_ below_ ring
- b – buff
- c – cinnamon
- e – red
- g – gray
- n – brown
- o - orange
- p – pink
- w – white
- y - yellow

veil_ color
- n - brown
- o - orange
- w – white
- y – yellow

veil_ type
- p - partial

ring_ number
- n – none
- o – one
- t – two

Independent Variables Continued

ring_ tpye

- e – evanescent
- f – flaring
- l – large
- n – none
- p - pendant

spore_ print_ color

- b – buff
- h – chocolate
- k - black
- n – brown
- o - orange
- r – green
- u – purple
- w – white
- y - yellow

population

- a – abundant
- c – clustered
- n – numerous
- s – scattered
- v – several
- y - solitary

habitat

- d – woods
- g – grasses
- l – leaves
- m – meadows
- p – paths
- u – urban
- w - waste