MUSHROOM EDIBILITY PREDICTION

November 11th 2022

Siham Elmali (694)
siham.elmali@gatech.edu

Mary Washington (303)
mwashington39@gatech.edu

Abstract

People are always looking for a means of incorporating nutrients into their diet. Mushrooms are a reliable source of vitamins, minerals, and other essential nutrients. They are abundant and grow in most biomes. The problem is that it is difficult to tell an edible mushroom apart from a poisonous one. In the Province of Parma, Italy, from January 1, 1996 - December 31, 2016, there were 443 mushroom poisonings. While most are not fatal, the risk is still there. Using a dataset with 173 unique species with 353 observations each (61,069 total observations) and 20 features (three metrical and 17 nominal), supervised learning was used to determine which of those features lent the most importance to identifying an edible mushroom from a poisonous one.

Four dataset permutations were used. Full dataset, full dataset without outliers, subset dataset using Shapley features, and subset dataset without outliers. Linear Discriminant Analysis (LDA), Logistic Regression, Support Vector Machine (SVM), Random Forest, Extreme Gradient Boosting (XGBoost), and neural network were the machine learning models used. These models were run on each of the 4 dataset permutations with hyperparameter tuning using a grid search and a repeated stratified k-fold cross validation (five-fold and 3 repeats). Recall (percentage of poisonous mushrooms correctly classified) was the performance metric prioritized because misclassifying poisonous mushrooms as edible carries a higher risk than misclassifying an edible as poisonous. LDA and Logistic Regression performed the worst with a Recall between 77.5 % - 79.8%. Random Forest, SVM, and Neural Net performed the best at 99.5% - 99.9% and had ~ 1.0 for the area under the ROC curve (AUC). They can correctly separate the two classes. The top overall feature importance, using Random Forest, were stem width, stem height, cap diameter, cap surface and cap shape.

Introduction

Mushrooms are an integral part of the ecosystem as they're responsible for breaking down matter and improving soil quality. These little fungi are a great source of vitamins, minerals, and other essential nutrients. Mushrooms are an important ingredient in one's kitchen—according to the U.S. Department of Agriculture, the average American eats approximately three pounds of mushrooms a year. Additionally, and in a recent report published by UCLA health, a review of 17 cancer studies between 1966 and 2022 showed that consuming 18 grams of mushrooms a day may lower cancer risks by up to 45%[1]. Mushrooms have been found to decrease the risk of cancer, lower sodium intake, promote lower cholesterol, protect brain health, provide a source of vitamin D, stimulate a healthier gut, and support a healthy immune system[1].

Most mushroom species are commonly found in the wilderness and have shared characteristics. Their similarities make it extremely hard to distinguish between the ones that are edible and those that are poisonous. The Province of Parma, Italy, did a study from January 1, 1996 – December 31, 2016. They found that there were 443 mushroom poisonings, with Autumn being the peak season[2]. The species could be identified in 397 of them[2]. This project was to see if there was a way to distinguish between an edible and poisonous mushroom.

There were quite a few data mining challenges for this project. The dataset that we chose was highly categorical (17 out 20 independent variables), lots of missing data (10 features have missing values and five had more than 65% missing values), with outliers. The feature space exploded from relatively small to highly dimensional after the encoding of categorical features.

The team faced computational challenges as well. Running a *GridSearchCV()* with a five-fold repeated three times on six different algorithms that require different computational complexity was extremely challenging. The team had to rely on GPU/TPU capabilities of Google Colab platform. Even with that, hyperparameter tuning and cross validation took a tremendous amount of time.

Problem Statement and Data Sources

The goal of this paper is to answer the following research questions:

What key attributes can help us predict the edibility of a mushroom?

How do we determine whether a mushroom is edible or poisonous?

To answer our research questions, the famous mushroom dataset from ***the UCI Machine Learning Repository*** was used. The dataset consists of **61,069** hypothetical mushrooms with caps based on **173** species (**353** mushrooms per species). Each mushroom is identified as edible, poisonous, or of unknown edibility and not recommended (the latter class was combined with the poisonous class). Of the 20 variables, 17 are nominal and three are metrical. A description of the features in the dataset can be found below.

Features	Description
class	p = poisonous, e = edible
cap diameter	float number in cm
cap shape	b = bell, c – conical, x = convex, f = flat, s = sunken, p = spherical, o = other
cap surface	i = fibrous, g = grooves, y = scaly, s = smooth, h = shiny, l = leathery, k = silky, t = sticky
cap color	n = brown, b = buff, g = gray, r = green, p = pink, u = purple, e = red, w = white, y = yellow, l = blue, o = orange, k = black
does bruise bleed	t = bruises or bleeding, f = no
gill attachment	a = adnate, x = adnexed, d = decurrent, e = free, s = sinuate, p = pores, f = none, ? = none
gill spacing	c = close, d = distant, f = none
gill color	n = brown, b = buff, g = gray, r = green, p = pink, u = purple, e = red, w = white, y = yellow, l = blue, o = orange, k = black, f = none
stem height	float number in cm
stem width	float number in mm
stem root	b = bulbous, s = swollen, c = club, u = cup, e = equal, z = rhizomorphs, r = rooted
stem surface	i = fibrous, g = grooves, y = scaly, s = smooth, h = shiny, l = leathery, k = silky, t = sticky, f = none
stem color	n = brown, b = buff, g = gray, r = green, p = pink, u = purple, e = red, w = white,

	y = yellow, l = blue, o = orange, k = black, f = none
veil type	p = partial, u = universals
veil color	n = brown, b = buff, g = gray, r = green, p = pink, u = purple, e = red, w = white, y = yellow, l = blue, o = orange, k = black, f = none
has ring	t = ring, f = none
ring type	c = cobwebby, e = evanescent, r = flaring, g = grooved, l = large, p = pendant, s = sheathing, z = zone, y = scaly, m = movable, f = none, ?= unknown
spore print color	n = brown, b = buff, g = gray, r = green, p = pink, u = purple, e = red, w = white, y = yellow, l = blue, o = orange, k = black
habitat	g = grasses, l = leaves, m = meadows, p = paths, h = heaths, u = urban, w = waste, d = woods
season	s = spring, u = summer, a = autumn, w = winter

Table 1: Description of the features

Proposed methodology

To complete this project, our team has chosen to take a structured approach by leveraging machine learning methodologies learned over the course of OMSA program. Our methodology includes the following steps: Exploratory Data Analysis, Data Preprocessing and Feature Engineering, Modeling, and Results & Analysis. The following chart shows the process the data had to go through before the modeling phase

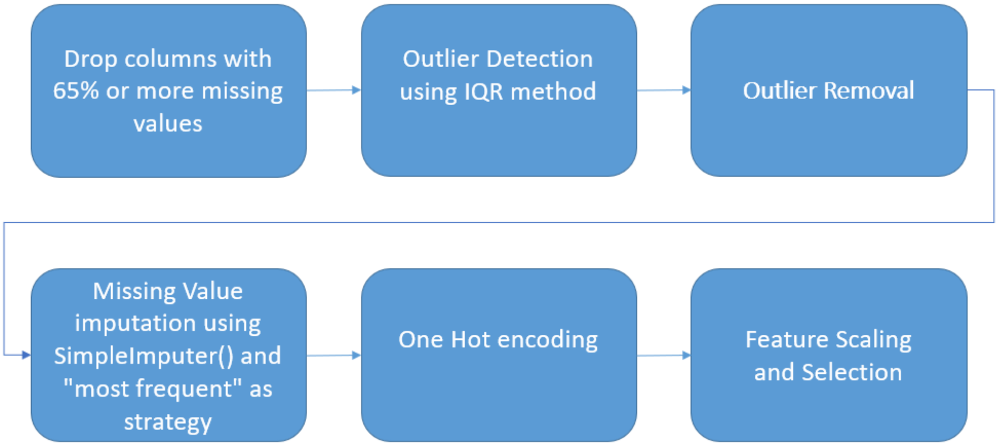

Drop columns with 65% or more missing values
Outlier Detection using IQR method
Outlier Removal
Missing Value imputation using SimpleImputer() and "most frequent" as strategy
One Hot encoding
Feature Scaling and Selection

Figure 1: Data preprocessing methodology

The data preprocessing phase took a tremendous amount of time as our dataset had several features (10 variables) with missing values and outliers (the three continuous variables). Several features were dropped (veil type, spore print color, veil color, stem root, gill spacing, and spore print color) due to a large percentage of missing values (see appendix figure 1). The remaining missing values were handled using imputation techniques.

After handling missing values, the three continuous variables in the dataset went through outlier detection using the Interquartile range (IQR) method. The IQR method is a statistical approach that calculates the difference between 25th percentile or Q1 and the 75th percentile or Q3 in the data. An observation is declared to be an outlier if it has a value 1.5x greater than IQR or 1.5x less than IQR (upper and lower bounds).

Next, the categorical features needed to have their values converted to indicator columns (e.g., season would be season_winter, season_summer, season_spring, season_autumn). This exploded out the feature space to 87 dimensions. The response variable (class) was also encoded using scikit-learn *LabelEncoder()* method. Additionally, the continuous variables were in different units (mm and cm) and had to be converted to millimeters. All training data was then scaled using scikit-learn *StandardScaler().*

Given the high dimensionality of our dataset, feature selection was required. Feature selection is a method to select the attributes in the data that are most relevant to the predictive modeling of the response variable (mushroom class). We initially selected a 10-fold cross validated Elastic Net model. However, none of the features had gone to zero. So, we had to explore other methods that might be better suited for highly categorical datasets. This is where we came across Shapley Values. Shapley (***SHapley Additive exPlanations***) values for feature importance or selection is a novel approach that calculates the average marginal contribution of each feature value across all possible combinations of features in a dataset[3] (see appendix figure 2). Random Forest was the model of choice when using SHAP python package for feature selection.

After feature selection, we split the data into training and testing sets using an 80/20 random split. For modeling, Linear Discriminant Analysis (LDA), Logistic Regression, Support Vector Machine (SVM), Extreme Gradient Boosting (XGBoost), Random Forest, and Neural Network were chosen. LDA and Logistic Regression were picked as baseline models because we wanted a model that had a linear decision boundary (splitting the dataset into their classes using a straight line). SVM, XGBoost, Random Forest, and Neural Networks are great for high dimension data and all, but SVM adds a penalty to incorrect classifications to give it more weight, so it has a better chance of getting classified correctly. All six models were cross validated using scikit-learn *GridSearchCV()* with a repeated stratified K-fold (five splits repeated three times).

The models were trained on all four different dataset variants (full dataset, full dataset without outliers, subset dataset (using features from Shapley values model), subset dataset without outliers). Several performance metrics were used to determine how well the cross validated models performed (see appendix). The two that were most important for our analysis were **Recall** and **Area Under the ROC Curve** (AUC). Recall (percentage of poisonous mushrooms correctly classified) is best when there are more risks of misclassifying something incorrectly for one class than another. Classifying a poisonous mushroom as edible has the risk of death, so that was of greater importance than simply misclassifying an edible mushroom as poisonous. AUC was used because it shows how well the models can separate the data into their respective classes.

Analysis and results

In this section, we begin analyzing each model's results per data variant. First, let's start by looking at the results using the full dataset (table below).

Models	Accuracy	Precision	Recall	F-1 Score	AUC
LDA	0.7731	0.7995	**0.7940**	0.7967	**0.8483**
Logistic Regression	0.7739	0.7987	**0.7972**	0.7980	**0.8540**
SVM	0.9939	0.9907	**0.9985**	0.9946	**0.9987**
Random Forest	0.9985	0.9981	**0.9993**	0.9987	**0.9999**
XGBoost	0.8905	0.9028	**0.9016**	0.9022	**0.9640**
Neural Net	0.9981	0.9972	**0.9994**	0.9983	**1.0000**

Table 2: Performance metrics per model using full dataset

As we analyze the results above, we see that Neural networks, Random Forest, and Support Vector Machines seem to consistently provide good results for all performance metrics used on testing data. As previously stated, we are prioritizing recall and the area under the roc curve as our main performance metrics of choice. In that regard, Neural Network is the best performing model. It not only does classify poisonous mushrooms correctly (Recall = 99.94%), but it also provides the ability to perfectly separate the two classes (auc = 1.0). By contrast, one can easily notice the poor performances of both Linear Discriminant Analysis and Logistic Regression models. The former's performance is to be expected because Discriminant Analysis assumes that the variables come from a multivariate normal distribution—hence all of them continuous. The later, however, makes no distributional assumptions of any kind—But it is prone to overfitting on high dimensional data.

Now, let's look at the modeling results using full dataset without outliers. It's worth noting that the outliers only applied to the three continuous features we have (cap diameter, stem height, and stem width). The table below reports the performance metrics of each model

Models	Accuracy	Precision	Recall	F-1 Score	AUC
LDA	0.7805	0.8151	**0.7928**	0.8038	**0.8578**
Logistic Regression	0.7797	0.8106	**0.7982**	0.8043	**0.8639**
SVM	0.9935	0.9924	**0.9960**	0.9942	**0.9977**
Random Forest	0.9986	0.9984	**0.9991**	0.9987	**1.0000**
XGBoost	0.9142	0.9309	**0.9168**	0.9238	**0.9756**
Neural Net	0.9985	0.9981	**0.9992**	0.9986	**1.0000**

Table 3: Performance metrics per model using full dataset without outliers

As the outliers were simply dropped, the dataset size was reduced from 61069 observations to 55728 observations. We clearly see that removing outliers didn't affect our best or worst performing models by much.

Next, let's look at the results using the subset features that were selected by Shapley values model.

Models	Accuracy	Precision	Recall	F-1 Score	AUC
LDA	0.7084	0.7227	**0.7776**	0.7491	**0.7626**
Logistic Regression	0.7057	0.7205	**0.7753**	0.7469	**0.7641**
SVM	0.9530	0.9599	**0.9560**	0.9580	**0.9925**
Random Forest	0.9938	0.9942	**0.9947**	0.9944	**0.9996**
XGBoost	0.8660	0.8761	**0.8860**	0.8810	**0.9378**
Neural Net	0.9931	0.9928	**0.9950**	0.9940	**0.9997**

Table 4: Performance metrics per model using subset dataset

From the table above, we see that Support vector machine model's recall, accuracy, and precision dropped by whopping **5%.** The AUC, that is the measure of the entire two-dimensional are underneath The **Receiver operating characteristic curve (True positive Rate (Recall) vs. False Positive Rate)** remains ~1.0.

Finally, let's look at the models' performance using subset dataset without outliers

Models	Accuracy	Precision	Recall	F-1 Score	AUC
LDA	0.6988	0.7158	**0.7781**	0.7457	**0.7707**
Logistic Regression	0.7003	0.7155	**0.7834**	0.7479	**0.7732**
SVM	0.9586	0.9712	**0.9553**	0.9631	**0.9940**
Random Forest	0.9932	0.9929	**0.9951**	0.9940	**0.9996**
XGBoost	0.8673	0.8794	**0.8879**	0.8836	**0.9409**
Neural Net	0.9925	0.9906	**0.9962**	0.9934	**0.9997**

Table 5: Performance metrics per model using subset dataset without outliers

Here, we see a slight performance improvement in Support Vector Machine metrics. Everything else remains comparable to the subset data with outliers.

Based on the above results, the models that performed the best testing data were Neural Networks and Random Forest, both trained on the full dataset without outliers. Both models unfortunately are less interpretable than say Logistic Regression or LDA. However, Random Forest and/or Neural Networks would be the model(s) of choice in production.

For reproducibility purposes, all this analysis was conducted using Python and Google Colab, a cloud based Jupyter Notebook environment.

Conclusions

When evaluating the results of all models and data variants, it was clear that Neural Networks and Random Forest outperformed the other models with the highest Recall and highest auc. Support Vector Machine was always the third best with a Recall greater than 95%. XGBoost wasn't a bad choice either with an average Recall of 89.8%. LDA and Logistic Regression, our baseline models, were consistently the worst. LDA performed better on the dataset with all the features and Logistic Regression performed better on the subset dataset.

According to Random Forest feature importance, the most distinguishing characteristics of determining whether you should eat a mushroom or not is to look at its stem width, stem height, cap diameter, if the cap surface is either sticky, smooth or silky, and if the cap is a convex or bell shaped. All feature importance plots per data variant are shown in the appendix.

Recall was the preferred performance metric as no one wants to possibly eat a poisonous mushroom, thinking it was an edible one. Neural Network, Random Forest, and SVM are capable of correctly separating the two classes of mushrooms (edible/poisonous). This data is not linearly separable, which we see with LDA and Logistic Regression. It also suffers from the curse of dimensionality because we had to expand the columns. This means that the more dimensions (features), the more data points are needed, which then increases the time and space complexity of the algorithms.

While our best performant models are great at separating the two classes of mushrooms, it is important to note that with high dimensional data that is mostly categorical, there will always be a risk of overfitting. Another aspect to keep in mind is the testing data used was also scaled. It would be interesting to see how the models would fair on non-scaled testing data. Another angle to explore is the use of a 60/20/20 or 70/20/10 splits, where 60% or 70% is designed for **training**, 20% for **validation,** and the remaining 20% or 10% is dedicated to **testing**.

Lessons learned

This project was challenging and quite fun at the same time. First and foremost, the most challenging part was choosing a dataset that was extremely categorical in nature. All baseline models chosen during the project proposal phase have performed poorly due to the nature of this data. The feature space exploded due to the one hot encoding of categorical variables. We suddenly had to deal with a highly dimensional dataset (87 features) that is mainly comprised of binary features. We knew that feature selection was required, but all methods learned in the course didn't perform well. This led to investigating other methods that might be better suited to handle this type of data. Reading through the data mining and machine learning literature allowed to come across Shapley values, a state of the art, model agnostic method for feature importance and selection in some case. Additionally, our team was very ambitious with the number of algorithms chosen. We didn't process the fact that each model is going to need hyperparameter tuning and be cross validated. We relied on *GridSearchCV()* from scikit-learn and that alone took a tremendous amount of time to run for each model and for every data variant. The good news is that *GridSearchCV()* picks the best model with the best parameters automatically.

One thing for sure that my partner and I can agree on is how much we learned about mushrooms, their nutritional, and health benefits. It is amazing how these little fungi can be a great source of vitamins and minerals, while also being an integral part of the ecosystem by breaking down matter and improving soil quality.

In terms of course structure, we believe the course to be well made with few areas of improvement. It would be nice if each module had further readings (i.e., academic papers on the topic covered in the module). In terms of peer reviews, it is highly recommended to require an x-word amount as minimum feedback to make sure students not only read the reports but also leave constructive feedback for their peers.

Other than that, the course was well run and had some of the most engaging course staff in the program.

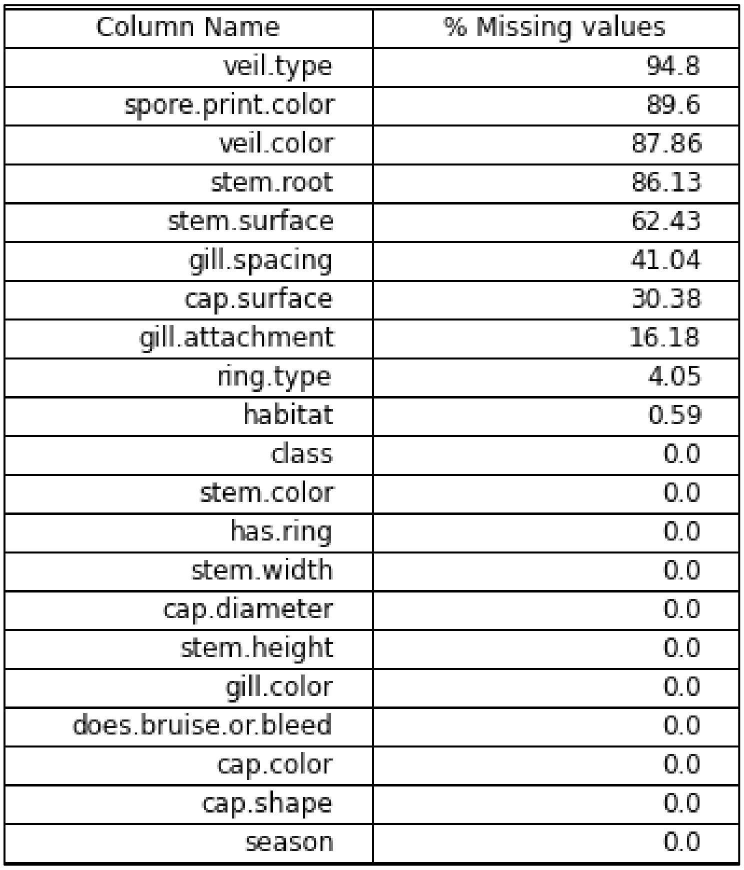
References

1. "*7 Health Benefits of Mushrooms*", UCLA Health, 24 Jan. 2022.

2. Cervellin G, Comelli I, Rastelli G, et al. *Epidemiology and clinics of mushroom poisoning in Northern Italy: A 21-year retrospective analysis*. Human & Experimental Toxicology. 2018.

3. Rathi, Prakhar. "*A Novel Approach to Feature Importance — Shapley Additive Explanations*" , Towardsdatascience, 2 July 2020.

Appendix



Column Name	% Missing values
veil.type	94.8
spore.print.color	89.6
veil.color	87.86
stem.root	86.13
stem.surface	62.43
gill.spacing	41.04
cap.surface	30.38
gill.attachment	16.18
ring.type	4.05
habitat	0.59
class	0.0
stem.color	0.0
has.ring	0.0
stem.width	0.0
cap.diameter	0.0
stem.height	0.0
gill.color	0.0
does.bruise.or.bleed	0.0
cap.color	0.0
cap.shape	0.0
season	0.0

Table 1: Missing Values

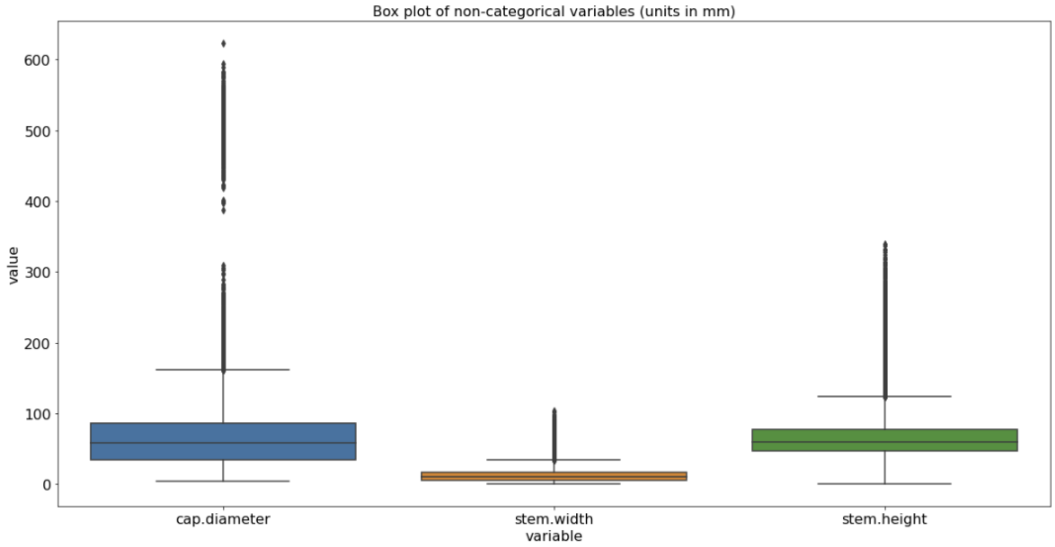


Figure 1: Outliers detection using a Box plot

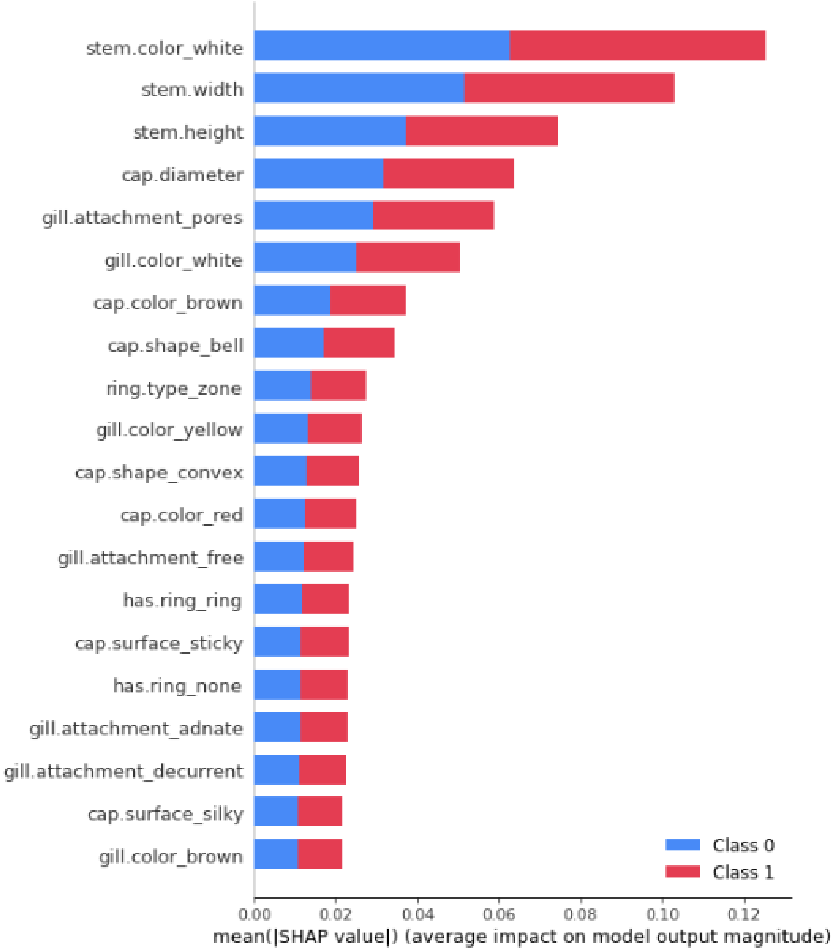


Figure 2: Feature Importance using Shapely Values

Full dataset results

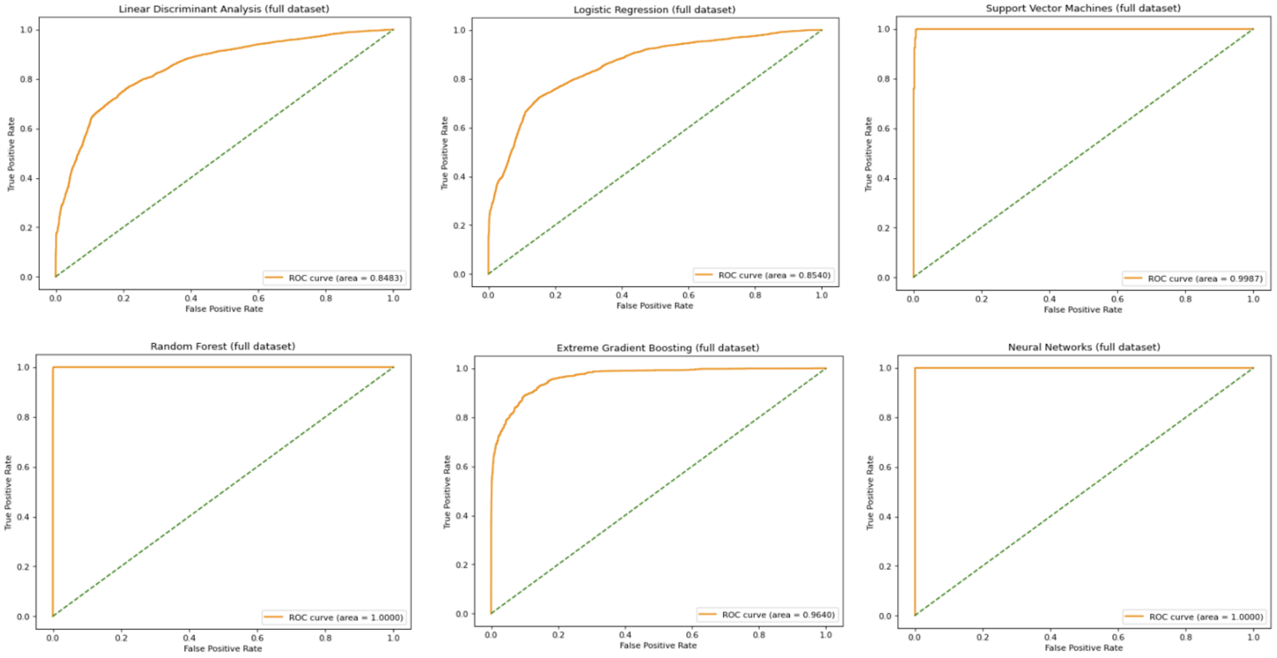


Figure 3: Area Under ROC curve per model

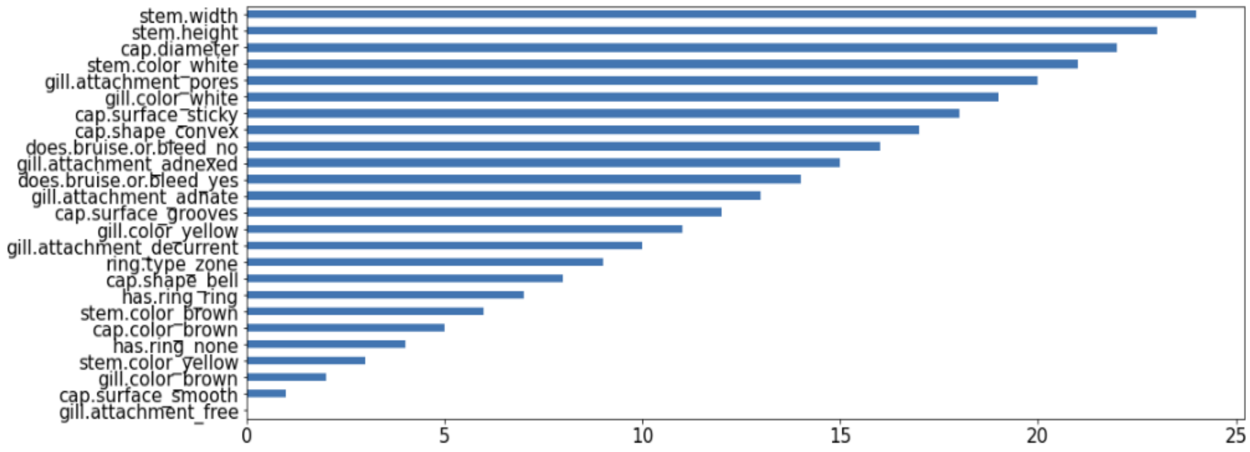


Figure 4: Feature Importance

Full dataset without outliers’ results

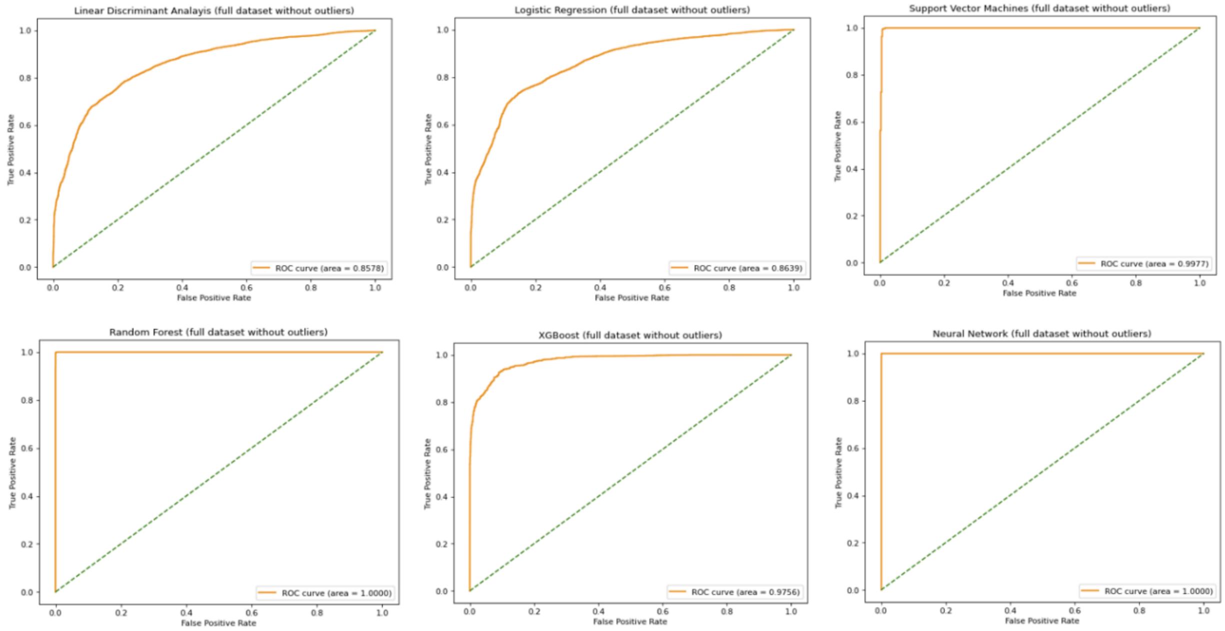


Figure 5: Area Under ROC curve per model

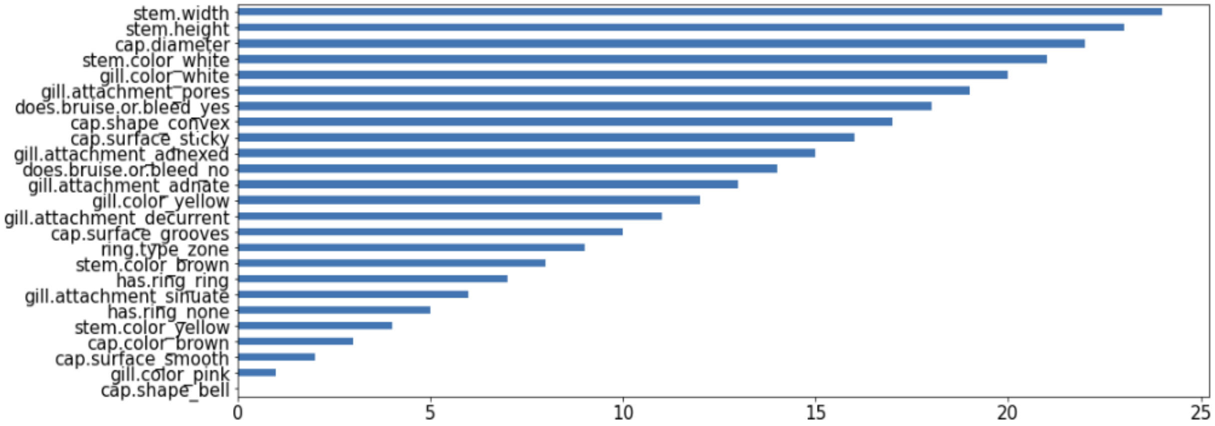


Figure 6: Feature Importance

Subset dataset’ results

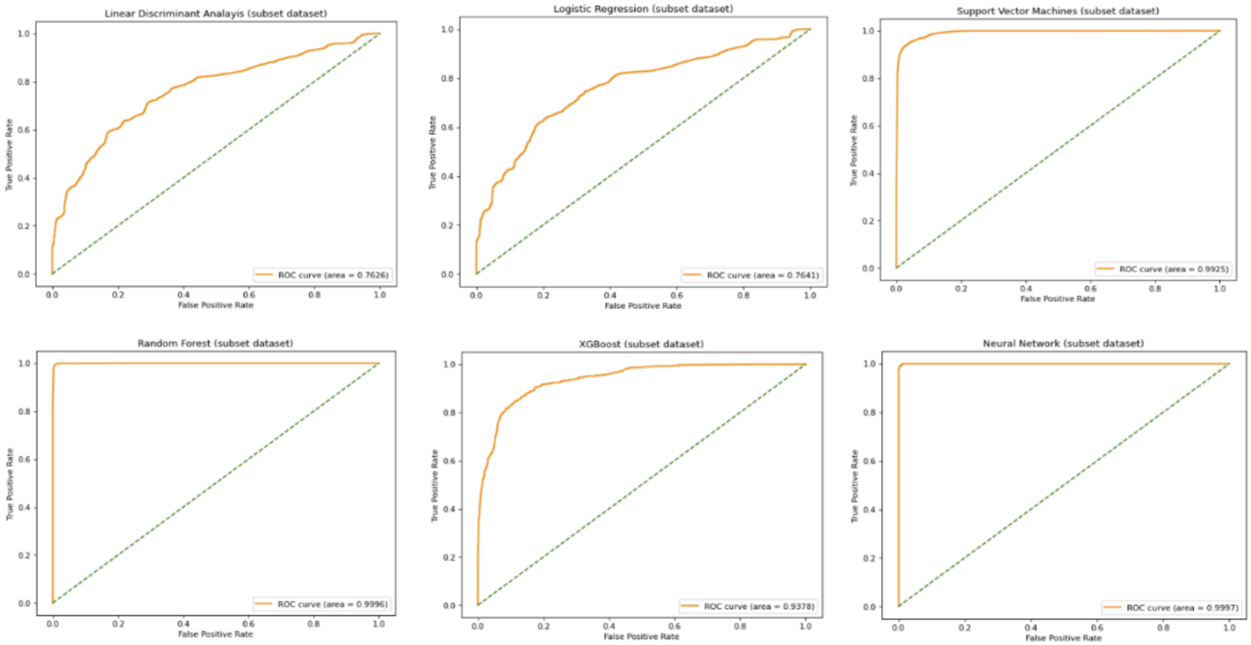


Figure 7: Area Under ROC curve per model

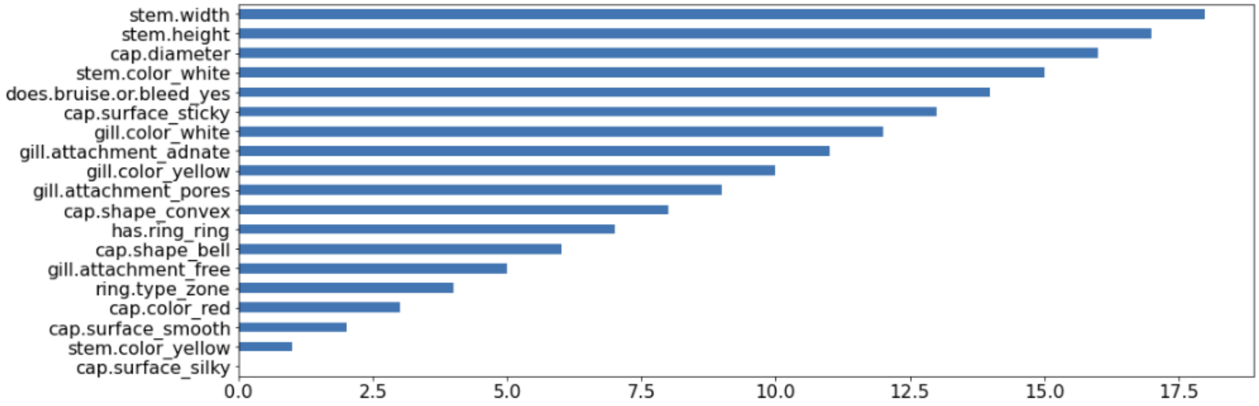


Figure 8: Feature Importance

Subset dataset without outliers' results

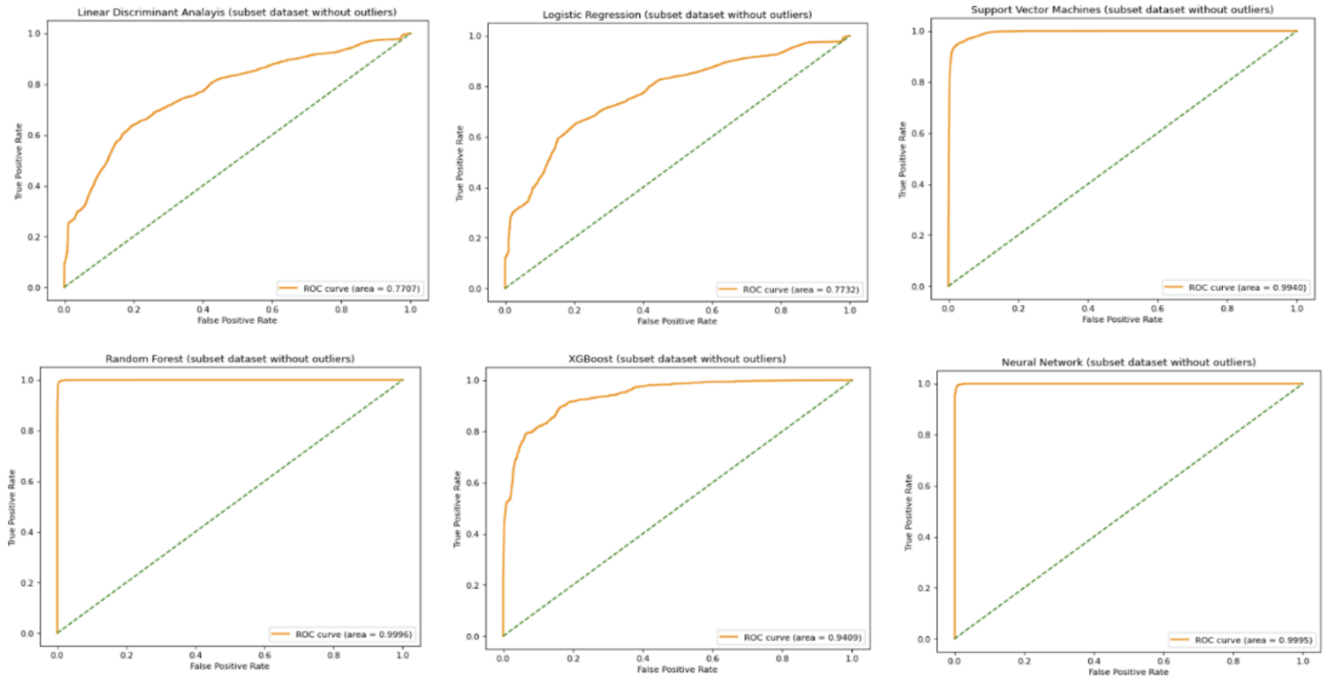


Figure 9: Area Under ROC curve per model

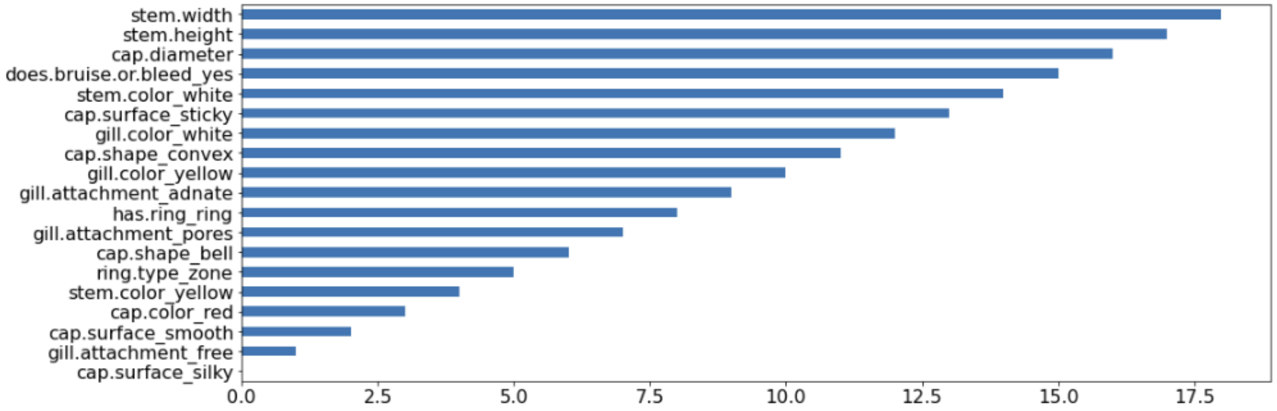


Figure 10: Feature Importance

Cross-validation results for select models (XGBoost and RF)

1. XGBoost

Performance metrics per tree

Trees	training rmse	training logloss	training auc	training classification error	validation rmse	validation logloss	validation auc	validation classification error
0	0.497	0.6872	0.5	0.4453	0.4969	0.6869	0.5	0.4442
5	0.3183	0.3804	0.9998	0.0057	0.319	0.3813	0.9997	0.0071
10	0.2085	0.2283	0.9999	0.0031	0.21	0.2298	0.9999	0.0042
15	0.1419	0.1445	1	0.0027	0.1436	0.1458	1	0.0029
20	0.1005	0.0947	1	0.0021	0.1026	0.0959	1	0.0024
25	0.0732	0.0623	1	0.0017	0.0753	0.0634	1	0.0019
30	0.0561	0.042	1	0.0014	0.0584	0.043	1	0.0016
35	0.0446	0.0286	1	0.0012	0.0473	0.0296	1	0.0012
40	0.037	0.0199	1	0.0009	0.0405	0.021	1	0.0012
45	0.0324	0.0143	1	0.0008	0.0355	0.0152	1	0.001
50	0.0289	0.0104	1	0.0007	0.0321	0.0112	1	0.0009
55	0.0261	0.0077	1	0.0006	0.03	0.0085	1	0.0008
60	0.0244	0.006	1	0.0006	0.0285	0.0068	1	0.0007
65	0.0222	0.0045	1	0.0005	0.0268	0.0053	1	0.0007
70	0.0207	0.0035	1	0.0004	0.0261	0.0044	1	0.0006
75	0.0196	0.0028	1	0.0004	0.0254	0.0037	1	0.0005
80	0.0184	0.0023	1	0.0003	0.0247	0.0032	1	0.0005
85	0.0171	0.0018	1	0.0003	0.024	0.0028	1	0.0006
90	0.0164	0.0016	1	0.0002	0.0231	0.0024	1	0.0005
95	0.0157	0.0014	1	0.0002	0.0232	0.0023	1	0.0005
100	0.015	0.0012	1	0.0002	0.0229	0.0021	1	0.0006
105	0.015	0.0012	1	0.0002	0.0229	0.0021	1	0.0006
110	0.0149	0.0012	1	0.0002	0.0229	0.0021	1	0.0006
115	0.0141	0.001	1	0.0002	0.0228	0.002	1	0.0005
120	0.0135	0.0009	1	0.0002	0.0228	0.002	1	0.0006
125	0.0124	0.0008	1	0.0001	0.0225	0.0018	1	0.0005
130	0.0118	0.0007	1	0.0001	0.0227	0.0019	1	0.0006
135	0.0116	0.0007	1	0.0001	0.0226	0.0018	1	0.0006
140	0.0116	0.0007	1	0.0001	0.0226	0.0018	1	0.0006
145	0.0116	0.0007	1	0.0001	0.0226	0.0018	1	0.0006
150	0.0113	0.0007	1	0.0001	0.0226	0.0018	1	0.0006
155	0.0109	0.0006	1	0	0.0231	0.0018	1	0.0006
156	0.0108	0.0006	1	0	0.0232	0.0018	1	

Cross validation performance metrics per fold

	mean	sd	cv_1	cv_2	cv_3	cv_4	cv_5
accuracy	0.9993	0.0002	0.9993	0.9995	0.9995	0.999	0.9995
auc	1	0	1	1	1	1	1
err	0.0007	0.0002	0.0007	0.0005	0.0005	0.001	0.0005
f1 score	0.9994	0.0002	0.9994	0.9995	0.9995	0.9991	0.9995
logloss	0.0025	0.0006	0.0026	0.0022	0.002	0.0035	0.0022
max_per_class_error	0.0012	0.0008	0.0016	0.0006	0.0007	0.0023	0.0007
mean_per_class_accuracy	0.9993	0.0003	0.9992	0.9995	0.9995	0.9989	0.9995
mean_per_class_error	0.0007	0.0003	0.0008	0.0005	0.0005	0.0011	0.0005
mse	0.0007	0.0002	0.0007	0.0005	0.0005	0.0009	0.0006
precision	0.9991	0.0006	0.9987	0.9996	0.9994	0.9982	0.9994
r2	0.9974	0.0007	0.997	0.9978	0.998	0.9962	0.9977
recall	0.9997	0.0002	1	0.9994	0.9996	1	0.9996
rmse	0.0253	0.0034	0.027	0.0231	0.0224	0.0305	0.0237
specificity	0.9989	0.0008	0.9984	0.9995	0.9993	0.9977	0.9993

2. Random Forest

Performance metrics per tree

Trees	training rmse	training logloss	training auc	training classification error	validation rmse	validation logloss	validation auc	validation classification error
0	.	.	.	.	.	.	.	.
5	0.0886	0.0774	0.9979	0.0081	0.0633	0.0214	1	0.002
10	0.0694	0.0313	0.9995	0.0035	0.0511	0.018	1	0.0013
15	0.0621	0.0247	0.9998	0.0025	0.0483	0.0176	1	0.0012
20	0.0559	0.0193	0.9999	0.0017	0.0455	0.0173	1	0.0009
25	0.0526	0.0174	1	0.0015	0.0442	0.0164	1	0.001
30	0.0508	0.0173	1	0.0013	0.0436	0.0165	1	0.0007
35	0.0501	0.0178	1	0.0013	0.0442	0.0172	1	0.0008
38	0.049	0.0175	1	0.0012	0.0436	0.017	1	0.0008

Cross-validation performance metrics per fold

	mean	sd	cv_1	cv_2	cv_3	cv_4	cv_5
accuracy	0.9991	0.0002	0.9989	0.9991	0.9994	0.9989	0.9992
auc	1	0	1	1	1	1	1
err	0.0009	0.0002	0.0011	0.0009	0.0006	0.0011	0.0008
f1 score	0.9992	0.0002	0.999	0.9992	0.9994	0.999	0.9993
logloss	0.017	0.0009	0.0164	0.0185	0.0161	0.0171	0.0168
max_per_class_error	0.0013	0.0003	0.0014	0.0014	0.0011	0.0016	0.0009
mean_per_class_accuracy	0.9991	0.0002	0.9989	0.999	0.9993	0.9988	0.9992
mean_per_class_error	0.0009	0.0002	0.0011	0.001	0.0007	0.0012	0.0008
mse	0.002	0.0001	0.002	0.0022	0.0019	0.002	0.0019
precision	0.999	0.0003	0.9989	0.9989	0.9991	0.9987	0.9994
r2	0.9919	0.0005	0.992	0.991	0.9923	0.9919	0.9923
recall	0.9993	0.0003	0.9991	0.9994	0.9998	0.9993	0.9991
rmse	0.0448	0.0015	0.0444	0.0472	0.0437	0.0449	0.0437
specificity	0.9988	0.0003	0.9986	0.9986	0.9989	0.9984	0.9993