Final Report: Mushroom Edibility Classifier

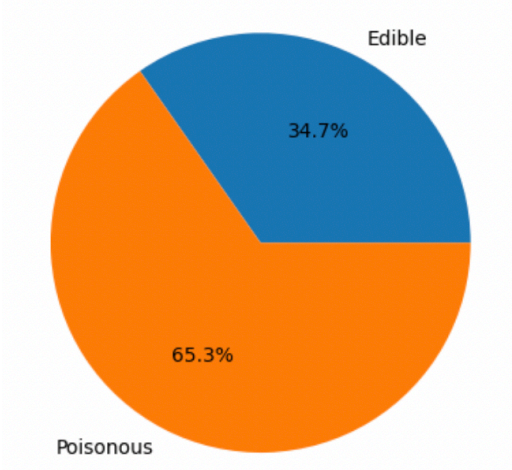
Problem Introduction

Many fear that wild mushrooms are generally poisonous and should be avoided. Image classification through machine learning can help determine a mushroom's toxicity from a photo. In this project, I experimented with pre-trained models from the Fastai library, ultimately selecting one that attained a 79% precision score and an 86% recall score for the poisonous mushroom class. While the model shows promise and is on the right track, it still requires further refinement before being ready for public use.

Data Wrangling and Exploratory Data Analysis

The original image files for this project were sourced from a kaggle.com user, Marcos Volpato, who had compiled them for a school research project. These images were categorized into four groups: edible sporocaps, edible mushroom sporocaps, poisonous sporocaps, and poisonous mushroom sporocaps. The initial data wrangling step involved renaming each image based on its directory name — either "poisonous" or "edible" — and subsequently consolidating them into these two primary classes.

To validate that each picture was neither corrupted or incomplete, I used Python Image Library's Image module and ran an iterated loop over each image with the verify method. All 3,400 images in the data passed the test. Moving forward with exploring the data, I compared the frequency of images in both classes:



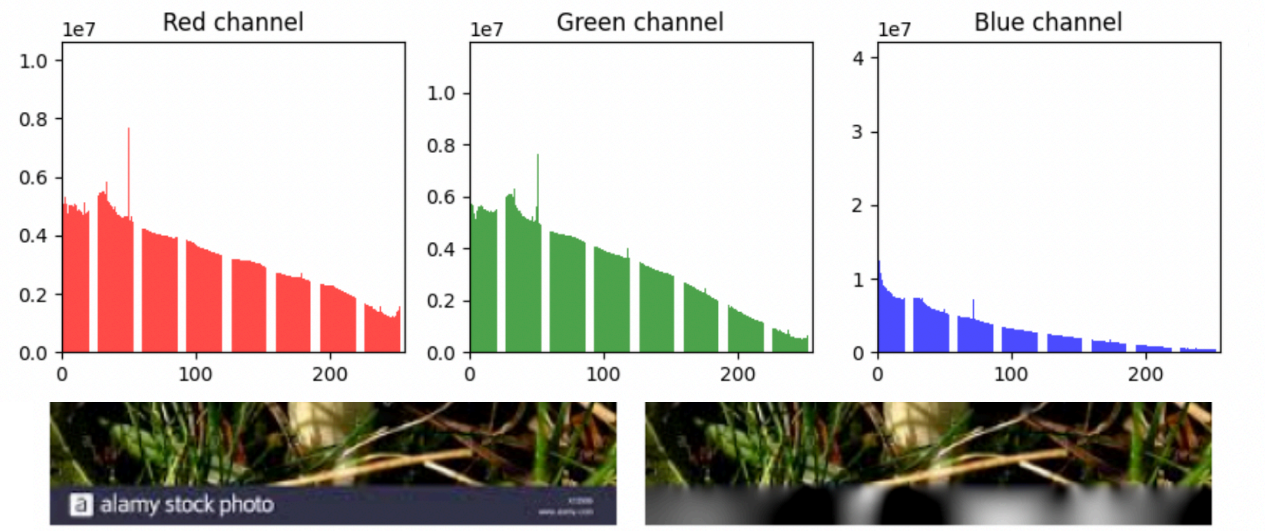
About one third of the images are of edible mushrooms while the remaining two thirds are poisonous mushrooms.

Many images in the dataset were great quality that captured the top of the cap, the gils underneath, and the stem of the mushroom in question. For example:

This image is exceptional given that the mushroom to the right is still attached to the ground, and it validates the background habitat is exactly the area which this species grows in. However, from previously spot-checking other images in this dataset I observed a trend of watermarks and URL banners across the bottom edge. To investigate a solution for these, I experimented with another image:

In this example, diagonal watermarks labeled "alamy" are present, along with a bar at the bottom edge displaying the text "alma stock photo". After generating a mask to address these image anomalies, I employed Sk-Image's `inpaint_biharmonic` function to reconstruct those areas:
While this `inpaint_biharmonic` effort did help to remove some of the unrelated markups, I determined that the original image may be sufficient for my project. The mask corrections may take more important textures away from the image than the watermarks. Additional augmentation can be done after the model metrics had been revealed.

Exploring pixel intensities across RGB channels in a collection offers insights into the dataset's characteristics and quality. The distribution of pixel intensities within the RGB channels can reveal information about the lighting conditions, color balance, and potential biases within the dataset. Here are the results from all images in the dataset:



What's apparent is that the blue hues are a lot less prevalent compared to red and green. Also, there are some spikes in intensity within both the red and green channels, but it is missing from the blue channel.

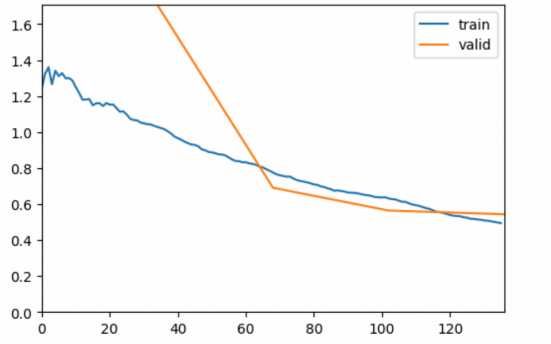
Modeling

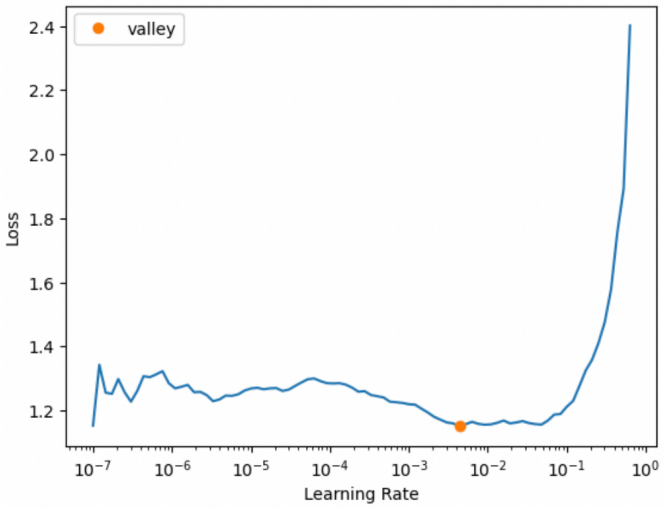
Before modeling, I converted all image files to the .jpg format and divided the collection into training, validation, and testing sets. Subsequently, using Fastai's Datablock API, I constructed two Dataloaders objects: one housing the training and validation data, and the other for the test data.

I trained and validated three models using Fastai's vision_learner model:

Model 1. This model employed Resnet34, resized images to 128 pixels, and ran for 4 epochs.

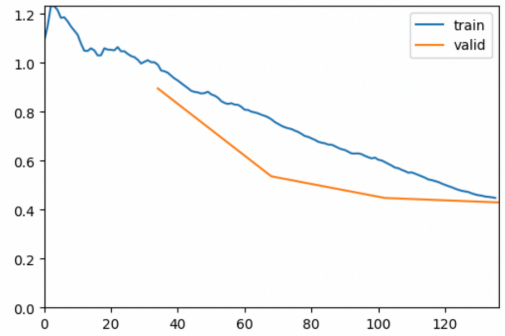
epoch	train_loss	valid_loss	precision_score	recall_score	time
0	1.034712	1.706136	0.694165	[0.15083799 0.94520548]	00:17
1	0.784716	0.690517	0.863946	[0.77653631 0.69589041]	00:18
2	0.637967	0.564080	0.833803	[0.67039106 0.8109589]	00:17
3	0.494150	0.543230	0.825000	[0.64804469 0.81369863]	00:17

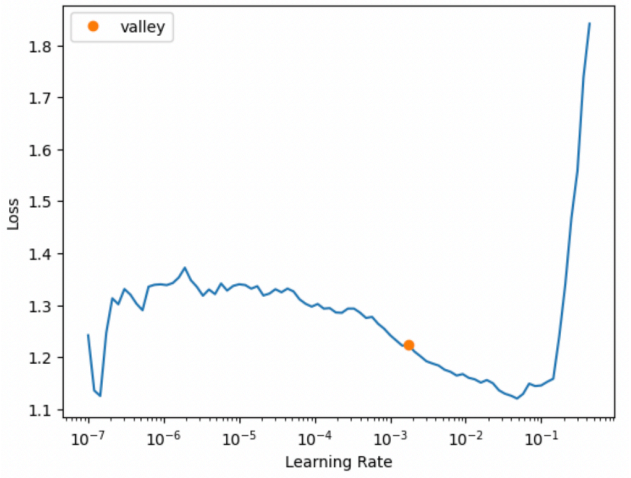




Model 2: This model employed Resnet34, resized the images to 224 pixels, and ran for 4 epochs.

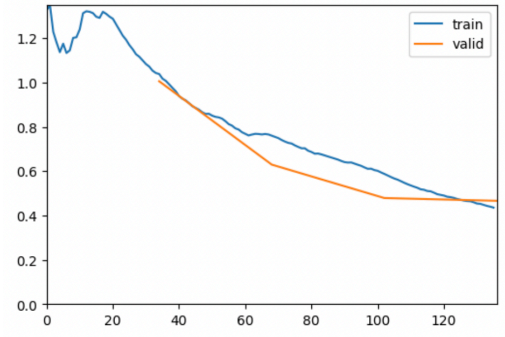
epoch	train_loss	valid_loss	precision_score	recall_score	time
0	1.002490	0.895864	0.746377	[0.41340782 0.84657534]	00:35
1	0.777751	0.536440	0.795970	[0.54748603 0.86575342]	00:37
2	0.601333	0.447965	0.834211	[0.64804469 0.86849315]	03:40
3	0.448319	0.429897	0.871148	[0.74301676 0.85205479]	00:35

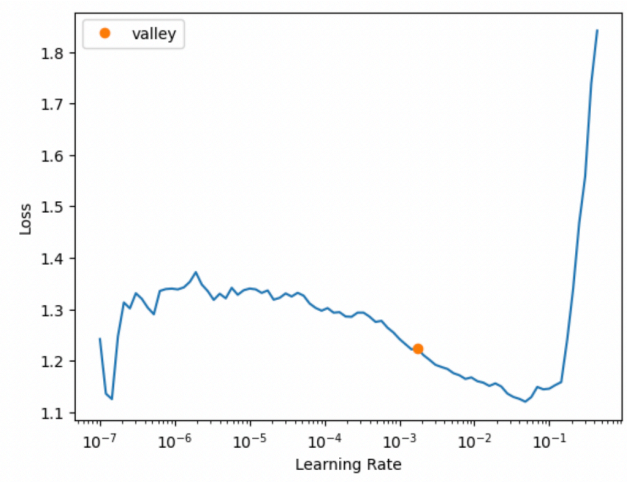




Model 3. This model employed Resnet50, resized images to 128 pixels, and ran for 4 epochs.

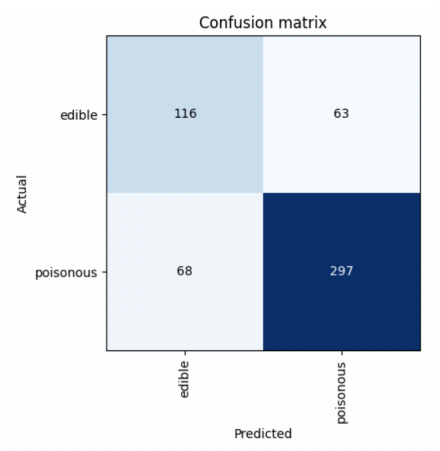
epoch	train_loss	valid_loss	precision_score	recall_score	time
0	1.042761	1.005499	0.713684	[0.24022346 0.92876712]	00:27
1	0.765568	0.629498	0.746606	[0.37430168 0.90410959]	00:27
2	0.594200	0.478642	0.833780	[0.65363128 0.85205479]	00:26
3	0.435454	0.466091	0.845304	[0.68715084 0.83835616]	00:27

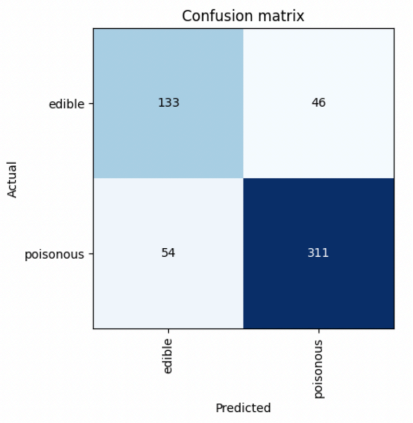


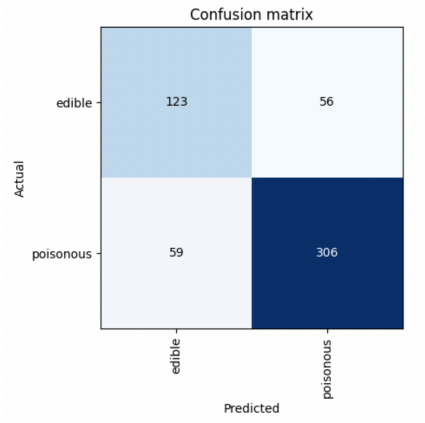


Among all the evaluated models, Model 2 consistently outperformed the others, leading in validation loss, precision, and recall for both classes.

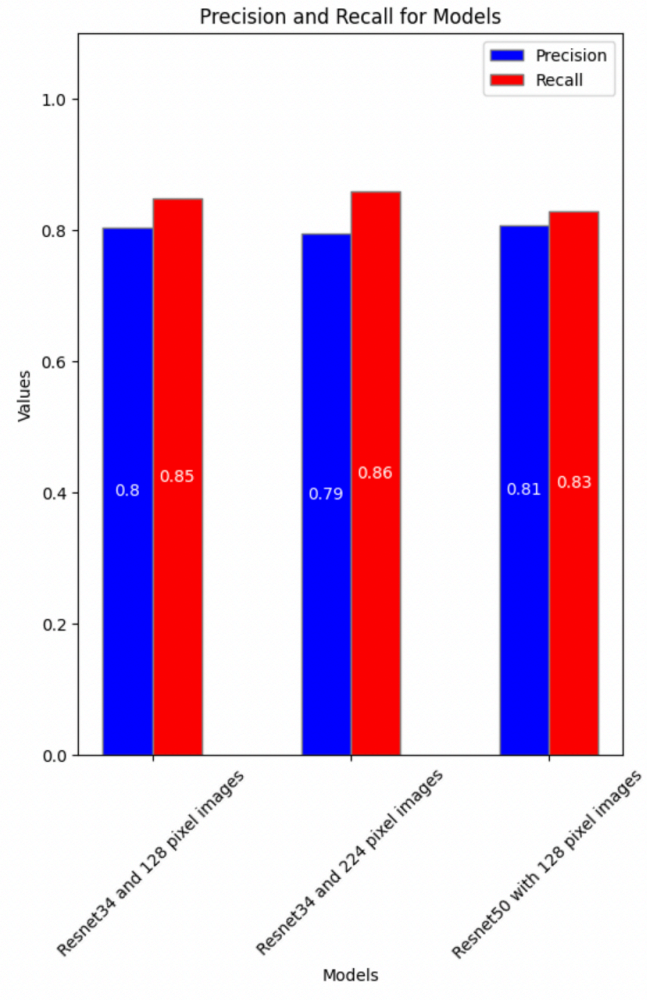
For a more detailed performance evaluation, I generated the confusion matrices for each model.







Lastly, I followed up with a precision and recall analysis on the test set's poisonous class of each model to gauge its potential real-world efficacy:



Conclusions

If precision in predicting poisonous mushrooms is our primary objective, the model utilizing Resnet50 with 128-pixel images emerges as the leader, demonstrating an accuracy of 81%. However, if our aim leans more towards maximizing the coverage of correct predictions for poisonous mushrooms (i.e., recall), then the model incorporating Resnet34 and 224-pixel images is superior with an accuracy of 86%. When evaluating the three confusion matrices, the second model consistently outperformed the others across all metrics. Therefore, based on the collective insights, the second model is my top recommendation for this project.

Further Research

To enhance my model's performance, prioritizing the addition of high-quality images to the dataset is crucial. These images should be accurately labeled as either poisonous or edible. By "high quality", I'm not only referring to resolution but also certain specific attributes of the images. It's essential for the image to frame the mushroom in a manner that captures the top of the cap, the underside, and the stem. Moreover, the background, which gives insights into the mushroom's habitat, is also vital.

I'm also considering leveraging more advanced hardware to process the existing data. This would involve utilizing a GPU system capable of handling Resnet50 and 224-pixel image sizes since my current machine was unable to manage this workload.

Further improvements could be made by addressing watermark issues present in the current dataset. Techniques such as thresholding and edge detection with biharmonic_inpainting can be explored. However, care must be taken to validate that these modifications don't inadvertently strip away crucial image features under the guise of removing markups. Additionally, I plan to adjust the blue pixel distribution to ensure it aligns more closely with the distribution curves of the red and green channels.