# Capstone Project

Machine Learning Engineer Nanodegree

Divya Amin
April 18th, 2019

## Definition

### Project Overview

Poisonous plants are those plants that produce toxins that deter herbivores before eating them. Plants cannot move so they have to protect themselves from other animals in other ways. The most common form of protection for plants is chemical. The poison produced by plants is usually meant to repel insects, but humans and other animals occasionally ingest or come into contact with the poisonous parts of the plants. As a result they suffer from a variety of symptoms such as mild discomfort, nausea, diarrhoea etc. But poisonous plants can cause fatal effects such as send a person into a coma, irregular heartbeats, convulsions etc. These plants can affect the lungs, digestive system, nervous system, heart as well as other parts in small amounts. In the cases of plants such as Daphne and Castor beans, a single seed(in the case of the Castor beans) or a few berries of Daphne can kill a person.
These poisonous plants can be mistaken for normal plants and so humans, especially children mistake them for a benign plant and may consume or touch the plants. Many humans except those who have jobs or hobbies that have them come across these plants on a regular basis such as forest rangers, hikers, people who work in nurseries or in maintenance of botanical parks, do not have knowledge about what these plants look like, the minor differences between the poisonous plants and their benign lookalikes. According to the National Capital Poison Center in the US, there were 660 poison exposures reported per 100,000 population. This machine learning algorithm can be used to identify a poisonous plant so that people who do not have prior knowledge of how these plants look can avoid or prevent other from eating or touching those plants.

### Problem Statement

**Description:** Building a poisonous plant classifier to identify poisonous plants in a picture.
**Challenge:** Many children and adults have little to no knowledge of poisonous plants and so can be harmed if they ingest or come into contact with the plants. Poisonous plants can look like normal benign plants and have a higher chance of being misidentified.
**Solution:** A Poisonous Plant Classifier that uses a Deep Convolutional Neural Network to identify the poisonous plant in the image uploaded. This problem is measurable in terms of accuracy and loss.

### Metrics

The evaluation metrics that will be considered for quantifying the model are accuracy and loss. Loss is the summation of errors made for each example for training or validation sets and is usually measured as negative log-likelihood for classification models like the one proposed here, which is given by:

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n} \log(\hat{y}^{(i)})$$

Negative log-likelihood is also known as Cross-entropy loss. It can also be written as:

$$CrossEntropyLoss = -(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i))$$

Where:

$y_i$ is actual label for $i^{th}$ training example.

$\hat{y}$ is predicted label for $i^{th}$ training example.

If $y_i = 1$, 2nd half of the cross entropy loss function disappears and if $y_i = 0$ the first half of the function disappears. So it is essentially the product of the log of the actual predicted probability for the ground truth class or actual class. This function is used to give a heavy penalty the predictions that are confident but wrong, which is necessary for identifying the poisonous plants because the model needs to be incredibly precise in identifying the poisonous plant as some poisonous plants look like benign plants or other poisonous plants and that usually confuses a human identifying the plants. So punishing the model using this loss function during training helps the model do precisely that.

Accuracy is ratio of the number of test samples predicted correctly to the total number of test samples. This again needs to be measured as we need to see how many plants has the model identified correctly. A low accuracy poisonous plant identifier means that the model is not identifying the poisonous plants correctly, which means it may be classifying one poisonous plant as a different poisonous plant very often. These metrics tell us whether the model is classifying plants correctly or not.

# Analysis

## Data Preprocessing

The poisonous-plants-images(link 3 in references) dataset contains hundreds of pictures of poisonous plants. These images are already segregated into training sets, testing sets and validation sets. This helps in training our poisonous plant classifier. This dataset contains 512 images of the following plants, which are also the classes or labels:

- Castor oil plant(castor_oil_plant)
- Wisteria(wisteria)
- Lily of the valley(lily_of_the_valley)
- Foxglove(foxglove)
- Dieffenbachia(dieffenbachia)
- Lilies(lilies)
- Rhubarb(rhubarb)
- Oleander(oleander)

**Fig 1**.Images from the poisonous-plants-images dataset

The images are all of varying resolutions. The dataset need not be cleaned as all the images are clear. None of the images are blurry or corrupt. All images are coloured. So data preprocessing was not needed

## Data Exploration

The following picture shows some of the images in the dataset and their actual classes or labels. Note that as mentioned in data preprocessing the images' dimensions were reduced to make them all square shaped.

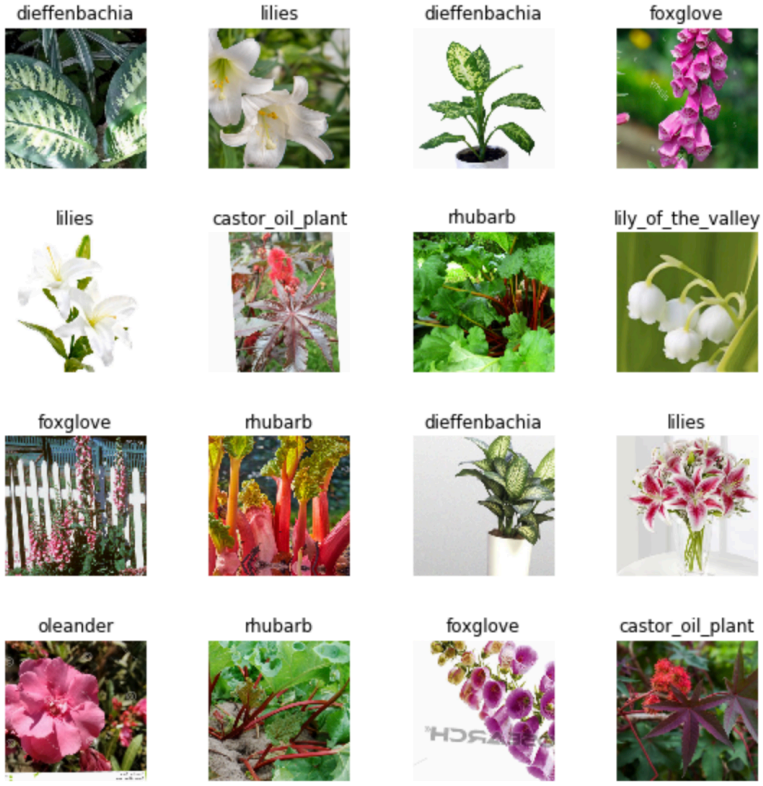



**Fig 2**. Some preprocessed images from poisonous-plants-dataset

## Exploratory Visualisation

In the 16 images that were randomly displayed from the datasets in Fig 2. ,we can see that is at least 1 picture of each class of poisonous plant. Since I was using the FastAI library to train my model and the dataset just had the class and image itself , I haven't really thought of a characteristic that I could visualise.

## Algorithms and Techniques

The classifier used is a Convolutional Neural Network(CNN), which is an algorithm used primarily for image processing tasks, including classification. These neural networks are suited for large amounts of data which is a requirement the dataset fulfils. The algorithm outputs an assigned probability of each class.
The following parameters can be tuned to optimise the classifier:

- Classification thresholds
- Training parameters
  - Number of epochs
  - Batch size(how many images to look at once during a single training step)
  - Learning rate(how fast to learn)
  - Weight decay(which prevents the model form being dominated by a few neurons)
  - Solver type(what algorithm to use for learning)
  - Momentum (takes the previous learning step into account when calculating the next one).
  - Neural network architecture contains the following parameters:
    - Number of layers
    - The types of layers(convolutional, fully connected and pooling layers)

- Layer parameters(link 9 in reference)
- Preprocessing parameter(none in this case)

Here we used both training and validation sets to train the model. Both the sets were loaded in the RAM.

## Benchmark

The benchmark model(given in link 4 of References) uses a Resnet18 model. As the model is a classification model, the metric used in the model is accuracy. This model, like the one proposed here is also used to identify which poisonous plants are recognised in the images.

The model is measured using the metrics accuracy and loss. This Resnet18 model has an accuracy of 93.39622497558594% and a loss of 0.2408438771963119 5. In one picture the model predicted that the picture was of an oleander when it was actually a foxglove. So this solution will be compared with the proposed solution on the basis of loss and accuracy. This result was obtained by actually training the Resnet18 model on the data.
So my goal here was to increase the accuracy of the model and decrease its loss.

# Methodology

## Data Preprocessing

Here the images just had the class given to them i.e what poisonous plant they are and the images were already segregated into training, testing and validation sets. There were no images that were corrupt or blurry or had any other problem. So no data preprocessing was done.

## Implementation

The workflow was as follows:

1. Load the training and validation images. The images in the dataset given do not require preprocessing.

2. Explore the data by checking the classes and using show_batch() to display some of the poisonous plants.

3. Training the model on the training data and logging the training/validation loss and accuracy and plotting the confusion matrix for result interpretation. Here I used accuracy as a metric to train the CNN.

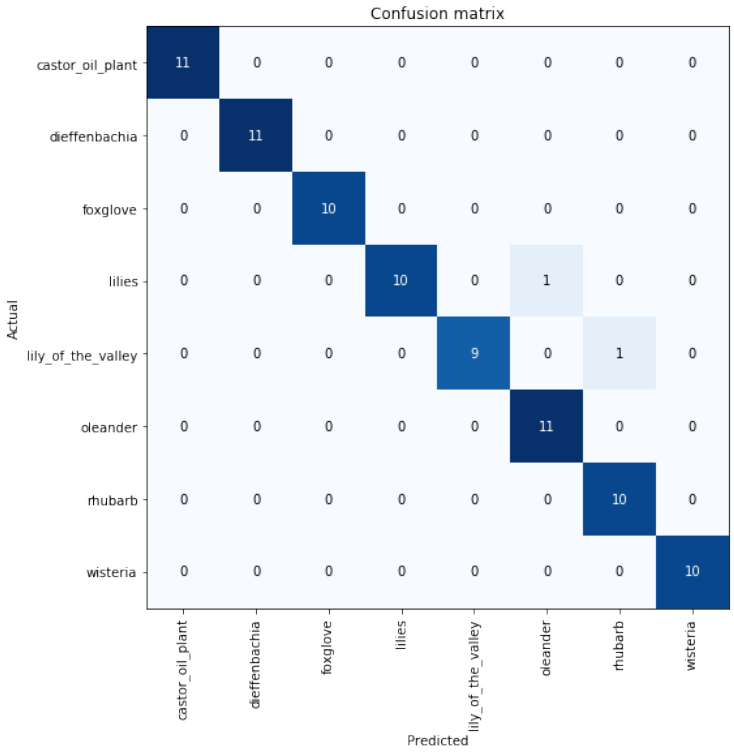
4. Fine tune the network and train again if accuracy not high enough.

5. Make the network predict on the test data and logging the accuracy and loss on that data.

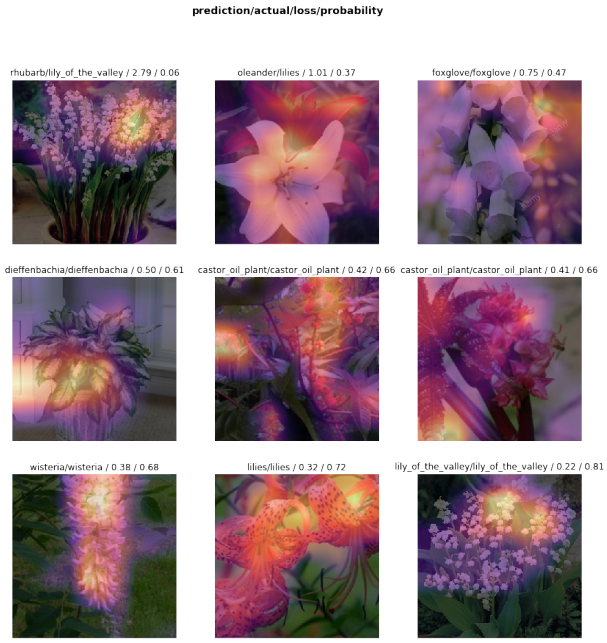
6. Save and freeze the neural network.

## Refinement

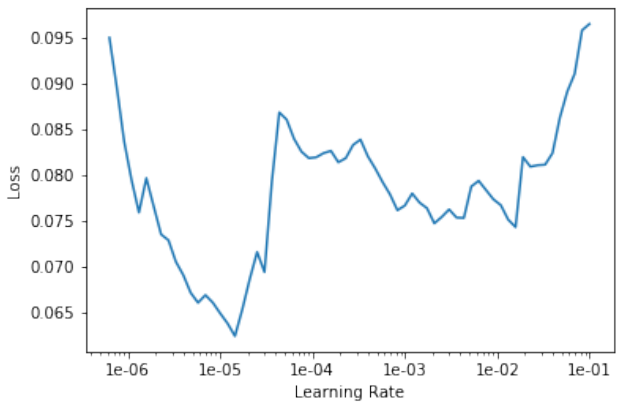
As mentioned in the benchmark section, the Resnet18 model had an accuracy(in%) of 93.39622497558594% and a loss of 0.2408438771963119 5. First I trained Resnet50 model(using fit_one_cycle(), which used One cycle scheduling)for 4 epochs without any fine tuning. The accuracy was 0.976190, training and validation loss was 0.078985 and 0.084846 respectively. I interpreted the classifier using ClassificationInterpretation.from_learner() and then plotted the confusion matrix.




Then I used plot_top_losses() to find some of the wrongly classified plants. They were as follows:




To find all the incorrectly classified images, I used most_confused(min_val=0). Then I saved the model. Then I plotted a graph of learning rate and loss, which was as follows:



After plotting the graph, I trained the Resnet50 model(using fit_one_cycle(), which used One cycle scheduling) with a parameter max_lr=slice(4e-6,1e-5) and for 4 epochs.

# Results

## Model Evaluation and Validation

During development, a validation set was used to evaluate the model.

The final model used was the Resnet50(Link 9 in References) Deep Convolutional Neural Network architecture. This is because Resnet50 uses residual learning, where we attempt to learn the subtraction of feature from the input of a Neural Network layer instead of trying to learn some features at the end of a layer. Resnet50 does this by using shortcut connections i.e directly connecting the output of nth layer to the (n+x)th layer. In the case of most neural networks as we go deeper(add more layers) the training of the network takes longer and the accuracy starts degrading as well. Resnet50 is much easier to train and solves the problem of degrading accuracy as well.

As mentioned earlier I finally trained a fine tuned Resnet50 model using one cycle scheduling(link 11 in References) with a parameter max_lr=slice(4e-6,1e-5) and for 4 epochs.

This was done because I unfroze my Resnet50 model I ran earlier. max_lr being equal to slice(X,Y) means we use X as learning rate for the pre-trained model and Y as the learning rate for the 2nd layer group of fully connected layers. One cycle scheduler uses an array in which each element represents the learning rate for each group of layers and when we have to update the epoch or step the learning rate is used for that group.

The result after training the final Resnet50 model was as follows:

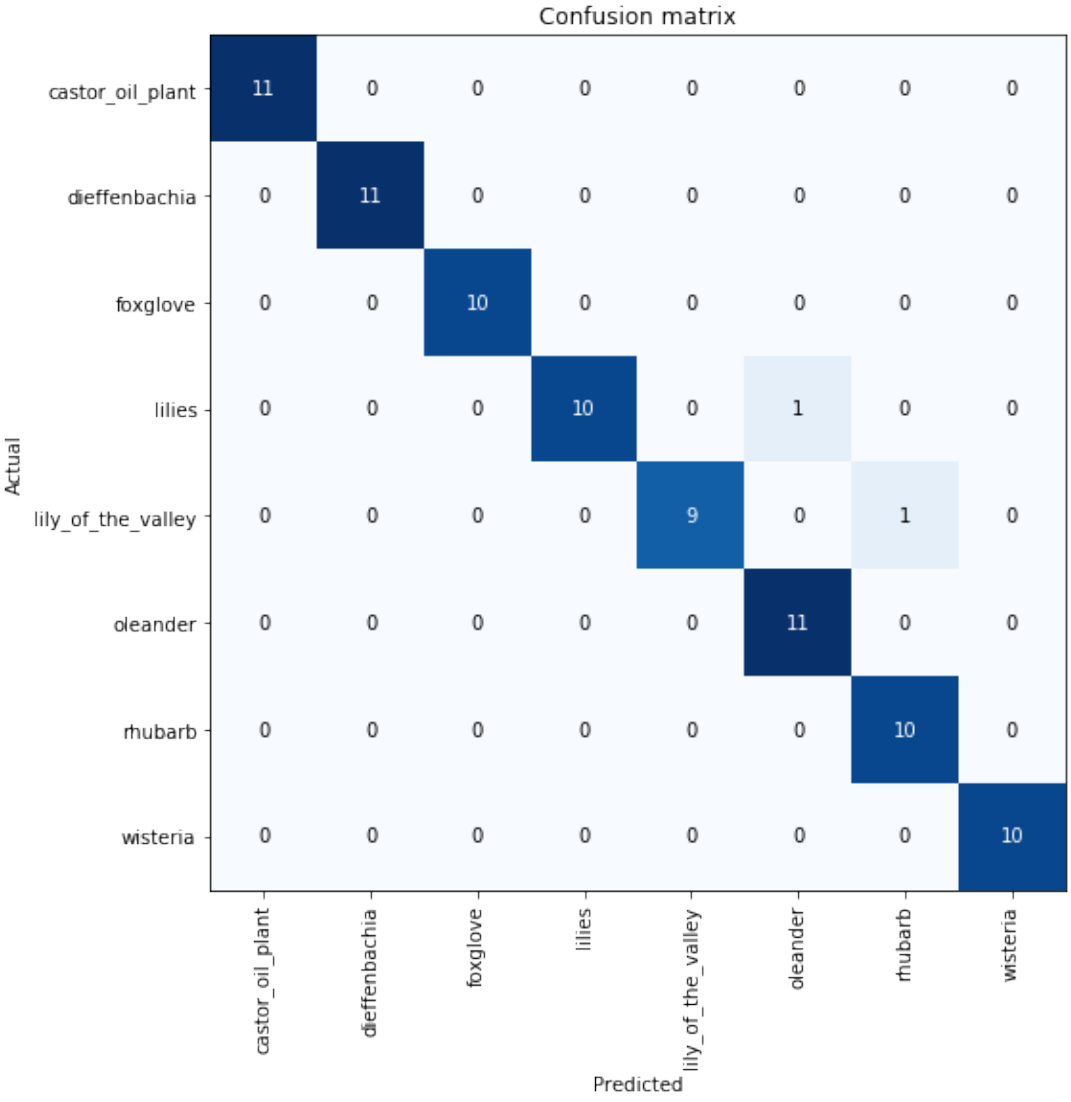
```
In [75]: learn.unfreeze()
         learn.fit_one_cycle(4,max_lr=slice(4e-6,1e-5))
```

Total time: 01:34

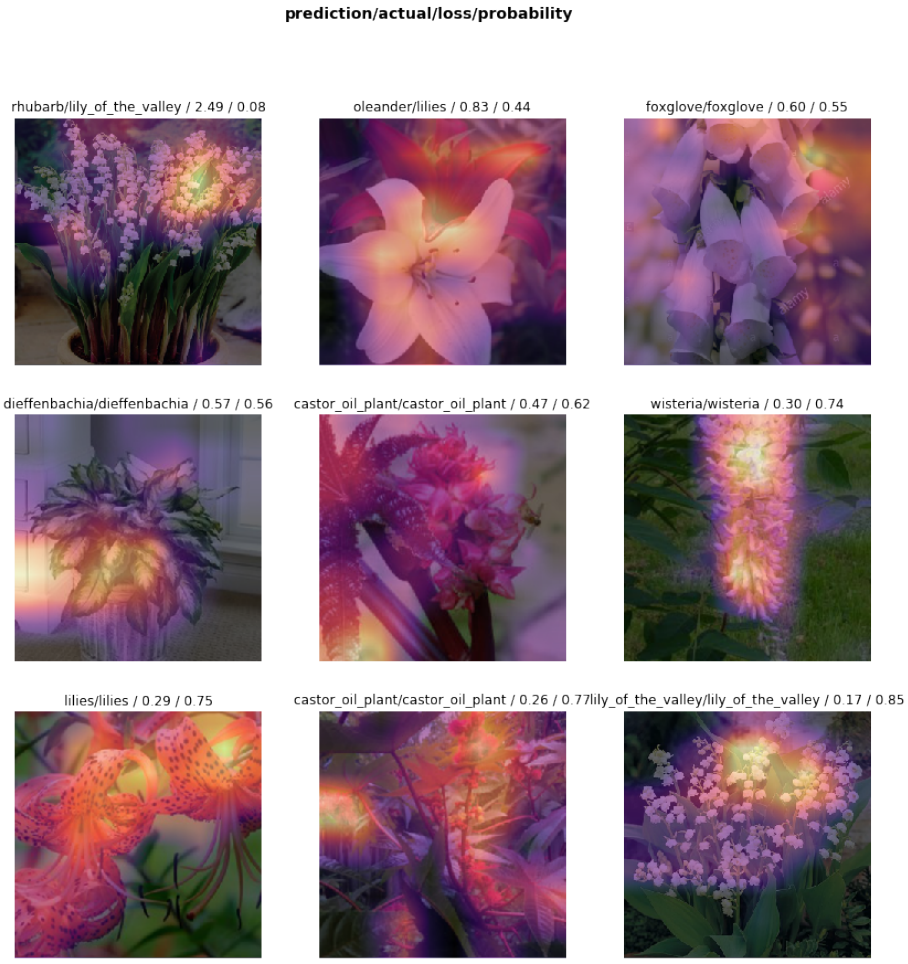
| epoch | train_loss | valid_loss | accuracy | time |
|---|---|---|---|---|
| 0 | 0.089648 | 0.097992 | 0.964286 | 00:24 |
| 1 | 0.094519 | 0.090515 | 0.976190 | 00:22 |
| 2 | 0.081331 | 0.088479 | 0.976190 | 00:23 |
| 3 | 0.078985 | 0.084846 | 0.976190 | 00:23 |

## Justification

As mentioned earlier the final training loss and validation loss are 0.078985 and 0.084846 respectively and the accuracy is 0.976190.On plotting the confusion matrix, we get:



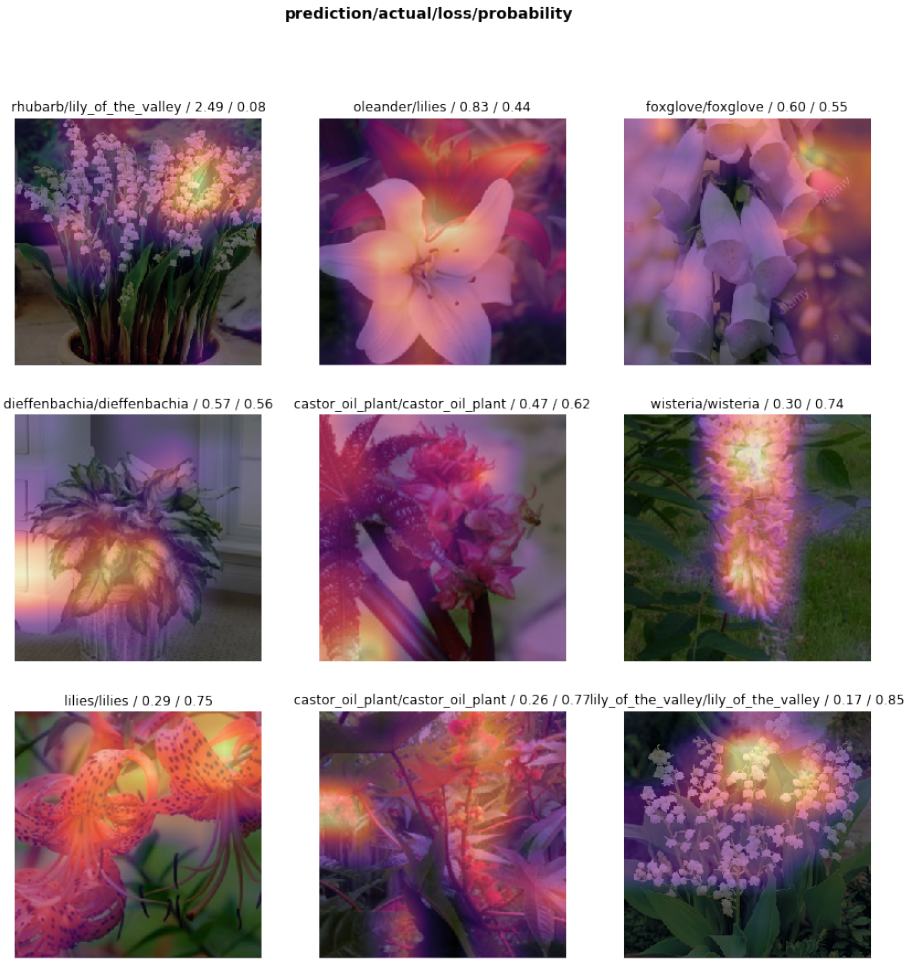

On displaying the top losses:



Here we can see from the above image that the final(fine tuned with learning rate) Resnet50 model is evidently more accurate than the non fine-tuned Resnet50 model. The losses to are much lower than in the non fine-tuned model. The probabilities too have increased in the new model. So we can say the final Resnet50 model has made a significant improvement than the earlier model and has solved the problem. Also on predicting the test data, the fine-tuned Resnet50 model had a loss of 0.13860592246055603 and an accuracy percentage of: 95.28302001953125 %

# Conclusion

# Free-Form Visualization




For this part I reinserted the top losses result from the earlier section. We are considering the predicted label, actual label of the picture, the loss and the probability. Here the loss shows how far from the actual label is the predicted label the model gave for these plants in a measurable way. The probability shows how likely the picture has the predicted label. This is a good way to take a peek at the performance of the model on the pictures

## Reflection

In this project, I've essentially done the following things:

- I found a problem to solve and the dataset to be used while solving the problem.
- I downloaded the data and viewed the some random samples.
- A benchmark was found for the classifier.
- The classifier was trained using the data in the dataset. Then the results were interpreted.
- As the accuracy was not satisfactory, the classifier was fine-tuned in terms of the learning rate and then trained again.
- The results and performance of the final poisonous plant classifier model was interpreted again.
- Then the final model was used to predict on test data.

Out of these, I found the part of fine-tuning my classifier after the initial training to be difficult as I had to familiarise myself with the Fastai library, its methods and how do we use the parameters in those methods, especially the max_lr parameter that was instrumental in improving the model's performance and accuracy by changing the learning rate for a group of CNN layers after updating the step.

## Improvement

Currently, the next steps for this problem would be to deploy it in an web or mobile application that can be used by children or adults who may not be well informed about how the poisonous plants look like. I also intend to expand the dataset to include non-poisonous plants in order to make it a model that can actually classify plants that are poisonous or non-poisonous and identify what plant it is based on the picture of the plant. This would be very useful for people to avoid coming in contact with poisonous plants and will help solve the problem of poisonous plants looking like harmless plants better as well as educate people, especially children on the plants present in different places such as their own backyard or in a forest trail.

## References


1. https://www.poison.org/poison-statistics-national-data-from-2016
2. https://aggie-horticulture.tamu.edu/earthkind/landscape/poisonous-plants-resources/common-poisonous-plants-and-plant-parts/
3. https://www.kaggle.com/nitron/poisonous-plants-images
4. https://www.kaggle.com/nitron/poisonous-plant-classifier-renset18
5. https://stackoverflow.com/questions/34518656/how-to-interpret-loss-and-accuracy-for-a-machine-learning-model
6. https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23
7. https://github.com/udacity/machine-learning/blob/master/projects/capstone/report-example-1.pdf
8. https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/
9. https://medium.com/@14prakash/understanding-and-implementing-architectures-of-resnet-and-resnext-for-state-of-the-art-image-cf51669e1624
10. https://forums.fast.ai/t/shedding-some-light-about-lr-management-in-fastai/43708/2
11. https://sgugger.github.io/the-1cycle-policy.html