Capstone Project Weekly Progress Report

Semester	Fall 2022
Course Code	AML 2404
Section	Section 2
Project Title	Skin Diseases Classification using Deep Learning
Group Name	G
Student names/Student IDs	Tomson George (C0857730) Praveen Mahaulpatha (C0860583) Thulana Abeywardana (C0861333) Jaskaran Singh Moti (C0860026)
Reporting Week	Week 8 (**30** October 2022 to **05** November 2022)
Faculty Supervisor	William Pourmajidi

1. Tasks Outlined in Previous Weekly Progress Report

Task 01: Modify Model 01 for better performance
Responsible: (Praveen)
Model 01 was kept on hold to proceed with other aspects of the project for two weeks. The reason was Model 02 performed better and utilizing the time for MLOps seemed a better option at the time. But since Model 02 is experiencing difficulties, we have decided to take up Model 01 and work on improving the accuracy.

Task 02: Learn and Implement the read/write functionality with the help of Boto3
Responsible: (Tomson)
Boto3 is the Amazon Web Services (AWS) Software Development Kit (SDK) for Python, which allows Python developers to write software that makes use of services like Amazon S3 and Amazon EC2. After installation, a successful connection to the S3 was created and tested. Also a file was uploaded to S3 via python code.

Task 03: Improving Model performance of Model 02
Responsible: (Thulana)
In addition to the image augmentation methods applied in the week 6, images were zoomed as data augmentation. Then optimizer changed from “Adam” to “RMSProp”; and finally learning rate was reduced. However, any of step did not lead to achive disired model performnces.

Task 04: Learn how to use plotly to read images and give representations related to it.
Responsible: (Jaskaran)
To get the Plotly library to produce at its highest level, we had to experiment and test it more. This job demonstrates how to better visualise our model and the dataset using Plotly (Images). We may show and study image data without requiring Matplotlib by using plotly. Using plotly, we can create heatmaps, a single-channel image, automatically resize images, combine images with charts, and even work with 3D images.

2. Progress Made in Reporting Week

Task 1 : Modify Model 01 for better performance

In this task, model 1 was modified and created Model 03 for ease of reference.

Step 1:
We have changed the structure of the neural network as the first step and the model performance is as Figure 2.1.

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(64,64,3)
Horizontal flip				True
Zoom Range				0.2
Convo2D	16	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Output Layer				4

Table: 2.1

Parameter	Value
Learning Rate	0.001
Epochs	40
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 2.2

Figure 2.1

Step 2:

We have identified the diseases we consider for the 3 classes look very similar to each other. This makes the model training hard and generalization more complex hence, as a work around to test if the model performs lower due to this issue in the data we have selected, we changed the 3 classes with diseases that significantly looked different from each other. The model output was mentioned in Figure 2.2.

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(64,64,3)
Horizontal flip				True
Zoom Range				0.2
Convo2D	16	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Output Layer				4

Table: 2.3

Parameter	Value
Learning Rate	0.001
Epochs	40
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 2.4

Figure 2.2

Step 3:

It was found that the dataset for training was imbalanced. The classes that were selected , number of images and their corresponding ratio of images are mentioned in Table 2.5.

Class	Training	Testing	Ratio
Hair Loss	240	61	1
Melanoma	464	117	2
Nail Fungus	1,041	262	4

Table: 2.5

We have down-sampled to check whether the effect was with respect to this and the below results were observed in Figure 2.3.

Figure 2.3

The conclusion for the increasing validation loss and stagnated validation accuracy could be due to ;

- The down-sampled number of images is not sufficient
- Not sufficient data Augmentation
- Not sufficient Layers to learn better parameters
- The effect of not having a dropout layer

These will be experimented in the Week 09.

Task 2: Learn and Implement the read/write functionality with the help of Boto3
Responsible: (Tomson)

Boto3 is a library for Python in order to interact with the AWS cloud via a terminal. To learn botot3, first, an environment was created using the following code.
$ python -m venv .venv
$ . .venv/bin/activate
Then credentials are set up in a hidden folder named 'aws' as follows, and a default region was set up.

1. In "`~/.aws/credentials`",

```
[default]
aws_access_key_id = YOUR_KEY
aws_secret_access_key = YOUR_SECRET
```

2. In "`~/.aws/config`",

```
[default]
region=us-east-1
```

The file was then uploaded by replacing the path and executing the following code.

```
import boto3
s3_connection = boto3.connect_s3()
bucket = s3_connection.get_bucket('your bucket name')
key = boto3.s3.key.Key(bucket, 'some_file.zip')
with open('some_file.zip') as f:
    key.send_file(f)
```

Figure 2.4

Task 3 : Thulana: Improving Model performance – Model 02

Step 1:

The model was tested for a new dataset with following parameters:

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				**horizontal_and_vertical**
Random rotation				**0.2**
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				64
Dropout				0.5
Output Layer				4

Table: 3.1

Parameter	Value
Learning Rate	0.01
Epochs	50
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 3.2

Following results were observed:

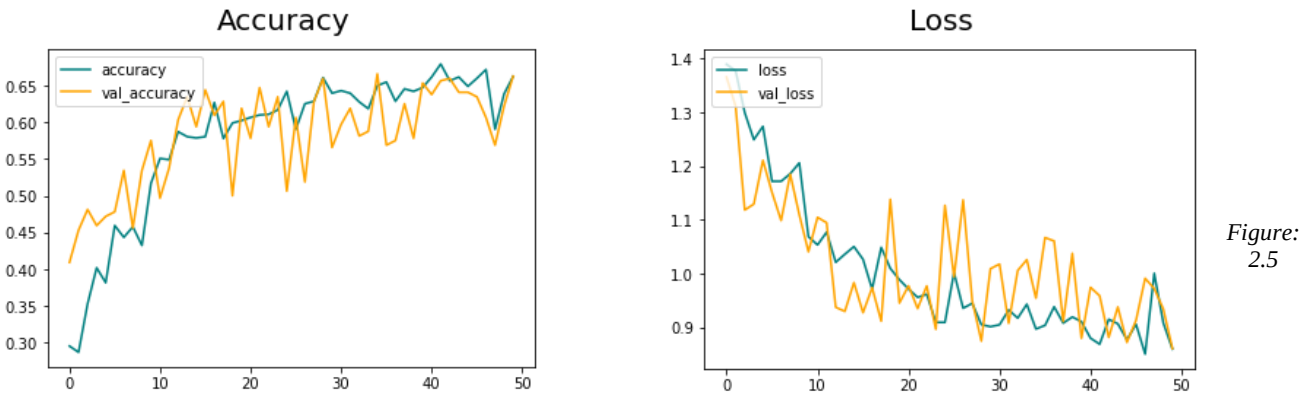

Figure: 2.5

Note: Even though the model shows a comparatively good performance; it has not achieved a desired level. The Accuracy is not goes above 0.55.

Step 2:
In this step, images were resized to 224 x 224 and run the model with same parameters (for 20 epochs) as table 3.1 and 3.2:.Following results were observed:

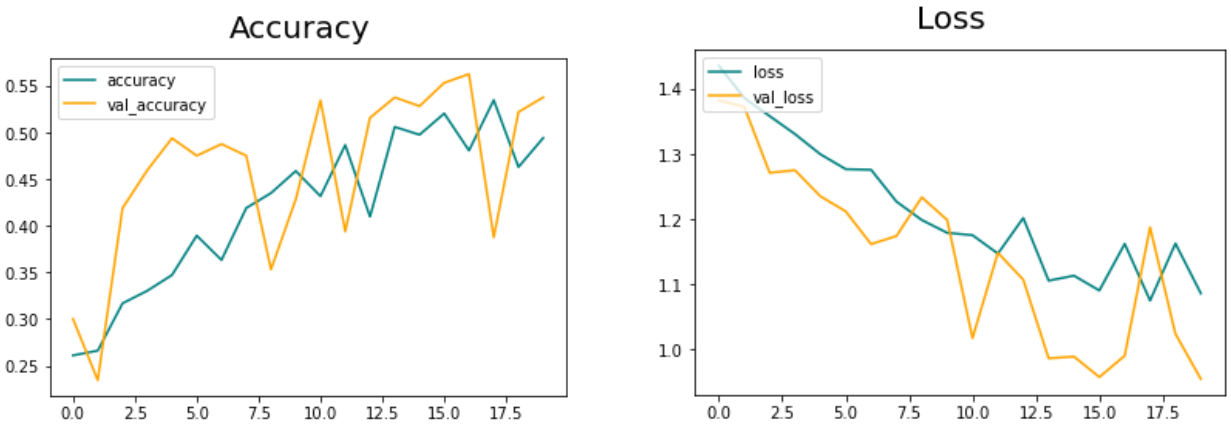

Figure: 2.6

Note: Desired model performances did not achieve.

Step 3:
In this step the model was simplified as follows:

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Convo2D	32	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5

Dense				64
Dropout				0.5
Output Layer				4

Table: 2.3

Parameter	Value
Learning Rate	0.01
Epochs	20
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 2.4

Following results were observed:

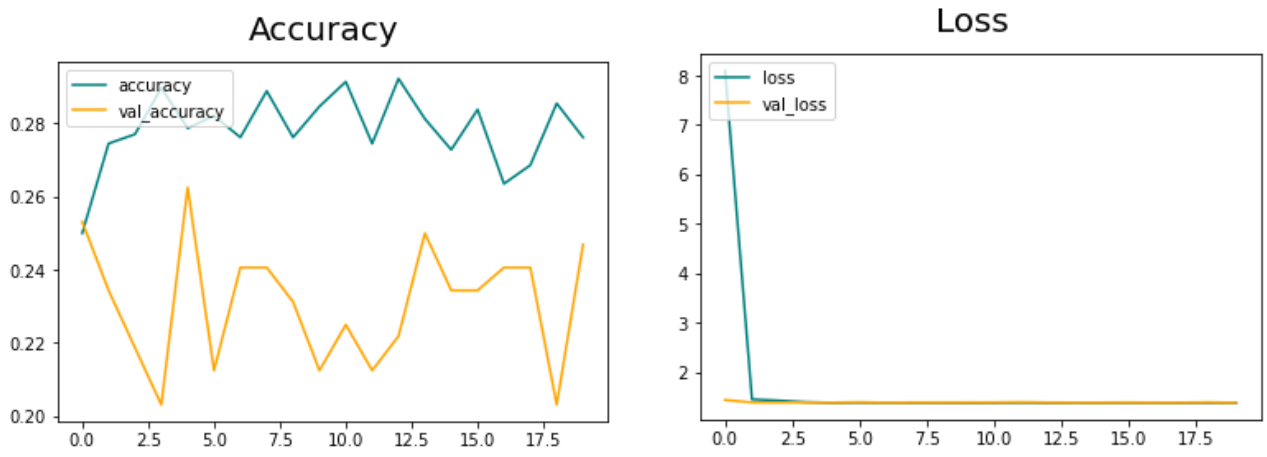

Figure: 2.7

Note: Model performances were decreased comparatively to the above step 2 and step 3

Step 4:

We inspected the dataset manually, and found that same types of images are included in different classes which is difficult to classify to the model. Therefore, a dataset with variety of classes were selected. The model was trained as per the following parameters:

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				**horizontal_and_vertical**
Random rotation				**0.2**
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				

Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				64
Dropout				0.5
Output Layer				4

Table: 2.5

Parameter	Value
Learning Rate	0.01
Epochs	20
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 2.6

Further number of images selected for each class in this stage is as follows:

Class	Number of images
Class 1	239
Class 2	453
Class 3	1017
Class 4	617

Table: 2.7

Following results were observed:

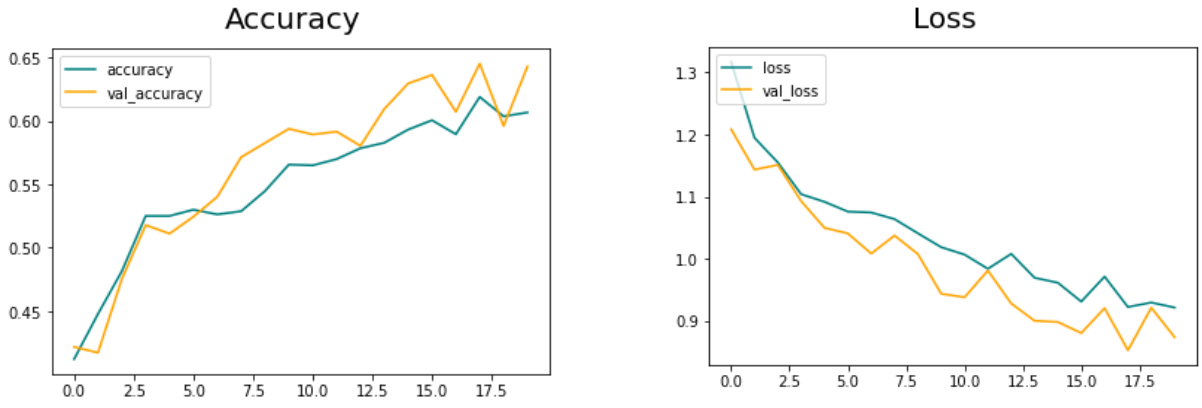

Figure 2.8

Note: Model achieved comparatively better performances. However not up to the desired level

Step 5

The model was trained for 50 epochs and other parameters were remained as same in above step 4 (table 3.5, 3.6 and 3.7). Then following results were observed:

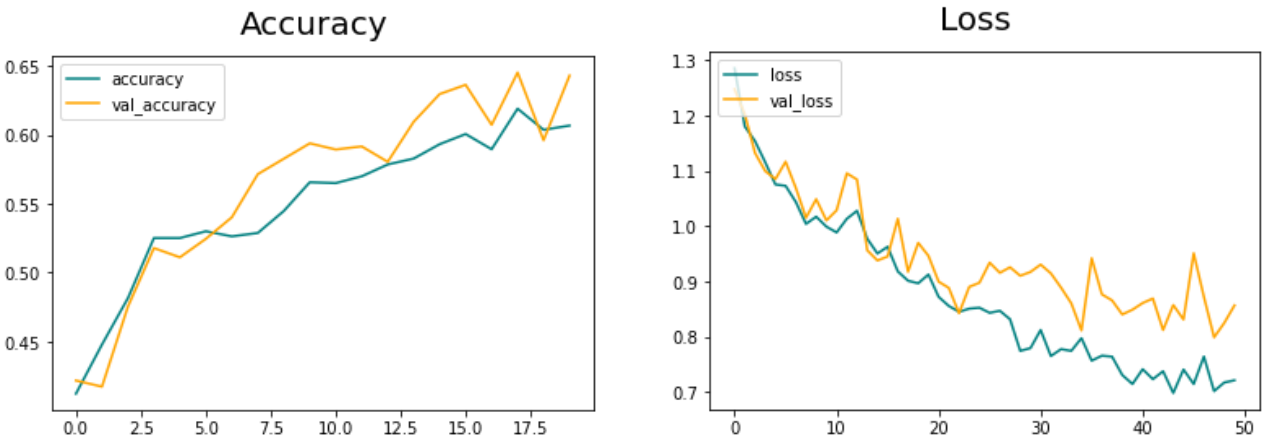

Figure 2.9

Task 04: Learn how to use plotly to read images and give representations related to it.
Responsible: (Jaskaran)

`px.imshow` displays multichannel (RGB) or single-channel ("grayscale") image data

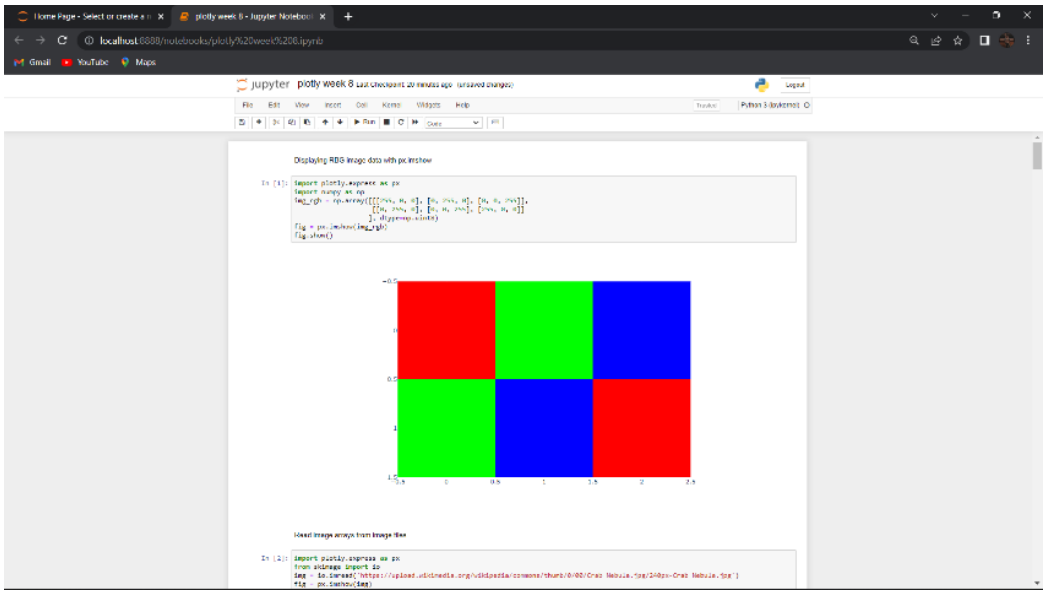

Figure 2.10

In order to create a numerical array to be passed to `px.imshow`, you can use a third-party You can use a third-party library like PIL, scikit-image, or opencv to generate a numerical array that

can be provided to px.imshow. The steps below demonstrate how to use skimage.io.imread to open an image from a file.

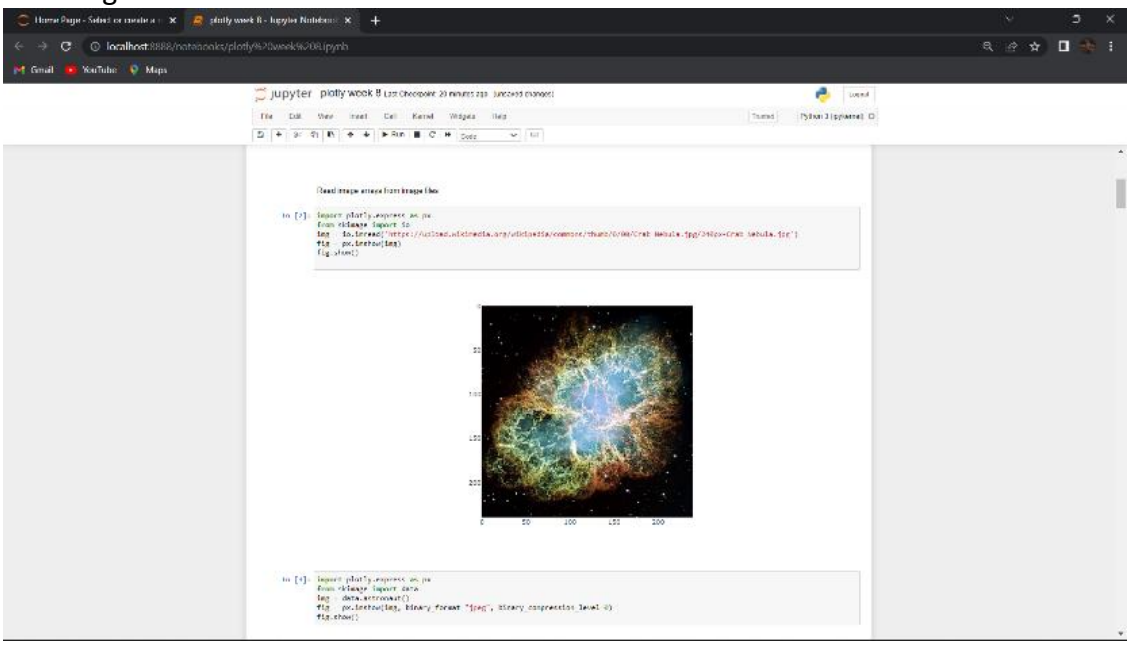

Figure 2.11

For a 2D image, `px.imshow` uses a colorscale to map scalar data to colors. The default colorscale is the one of the active template

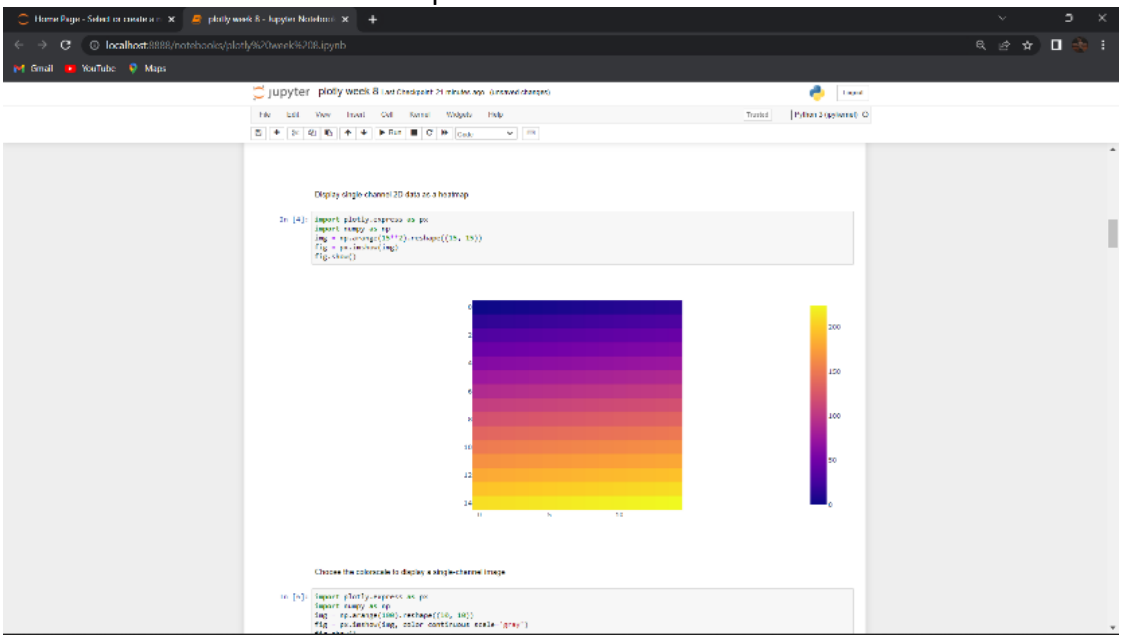

Figure 2.12

customize the continuous color scale just like with any other Plotly Express function. However, `color_continuous_scale` is ignored when using `binary_string=True`, since the image is always represented as grayscale

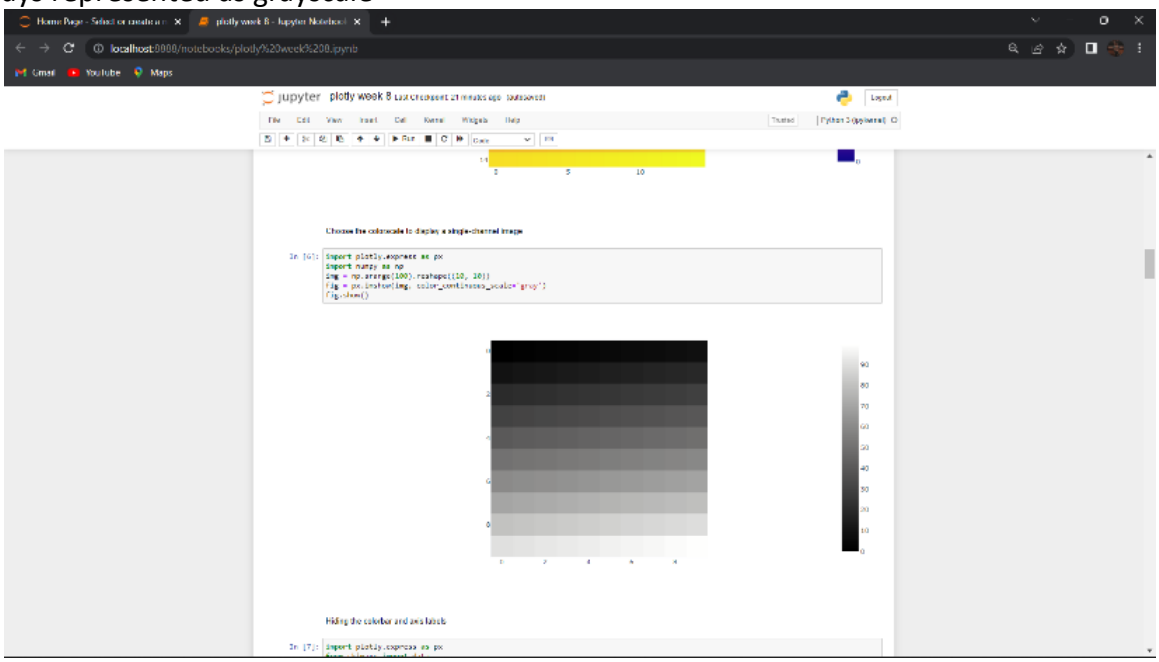

Figure 2.13

Hiding the colorbar and axis labels

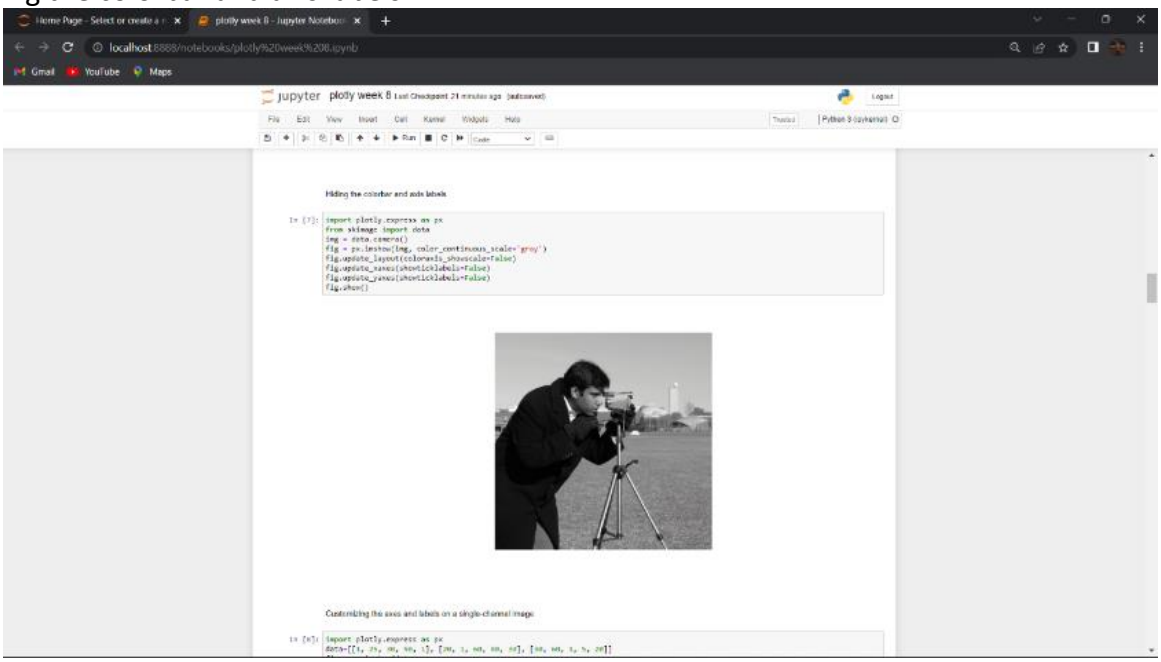

Figure 2.14

the x, y and labels arguments to customize the display of a heatmap, and use .update_xaxes() to move the x axis tick labels to the top

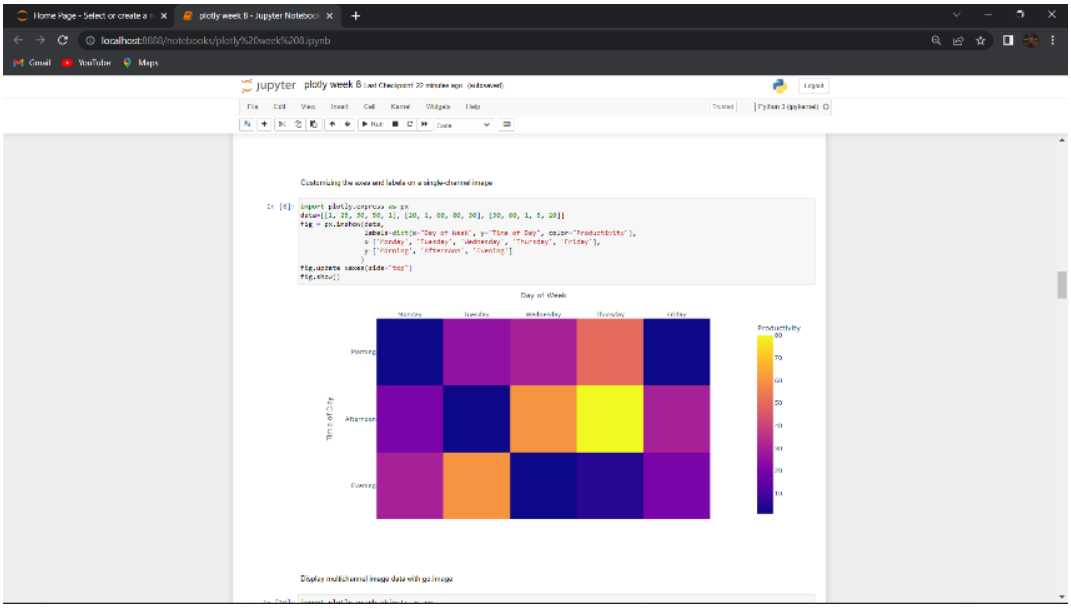

Figure 2.15

The z parameter of go.Image passes image data in the form of an array or a list of numerical values, but it is also possible to use the source parameter, which takes a b64 binary string. Thanks to png or jpg compression, using source is a way to reduce the quantity of data passed to the browser, and also to reduce the serialization time of the figure, resulting in increased performance.

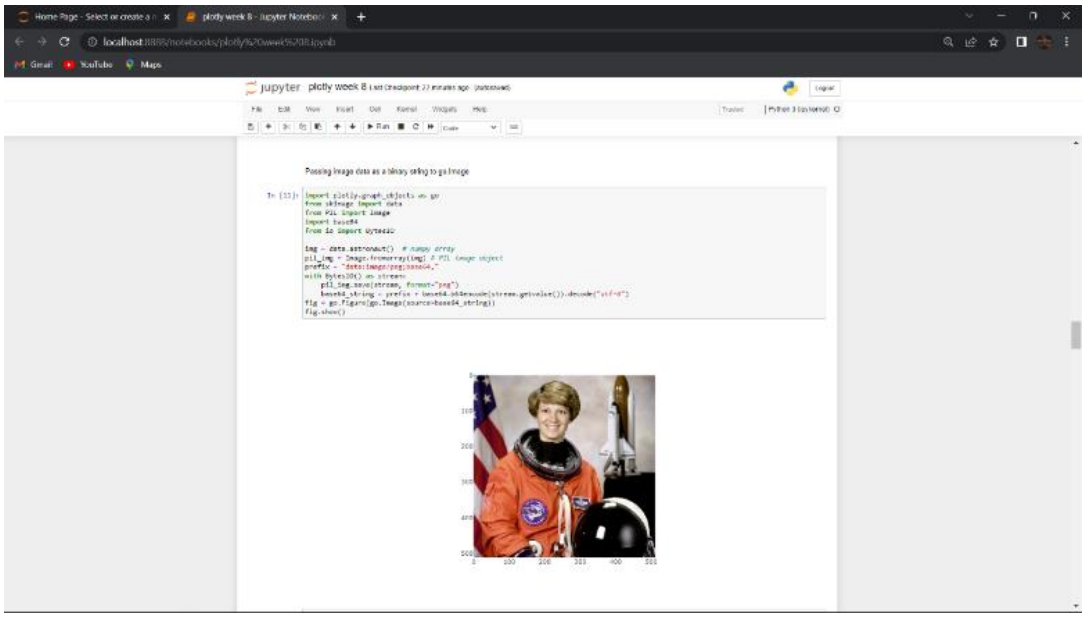

Figure 2.16

Ticks and margins around image data

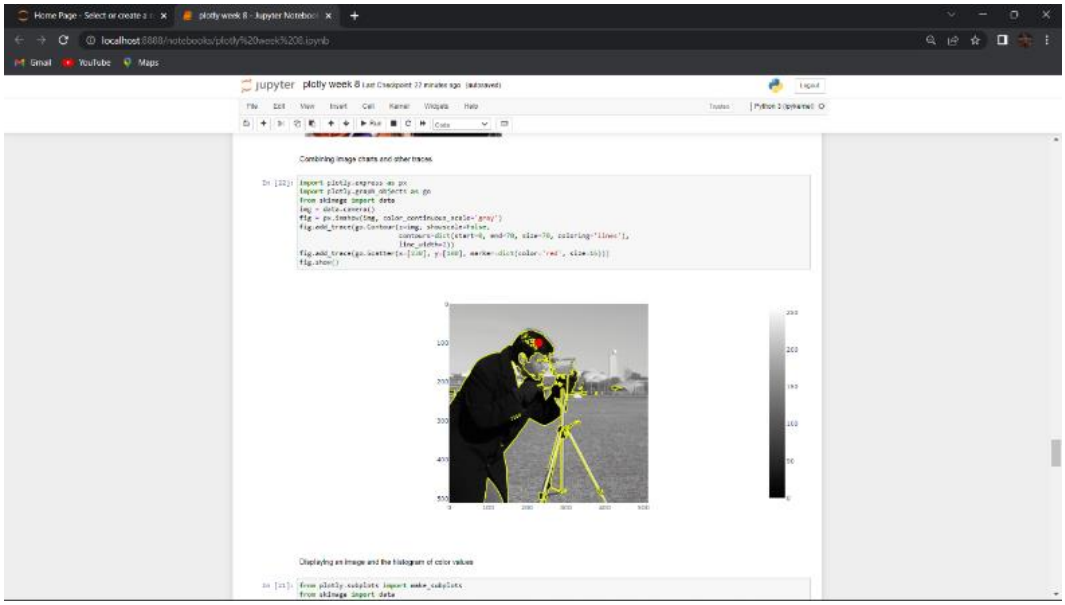

Figure 2.17

Combining image charts and other traces

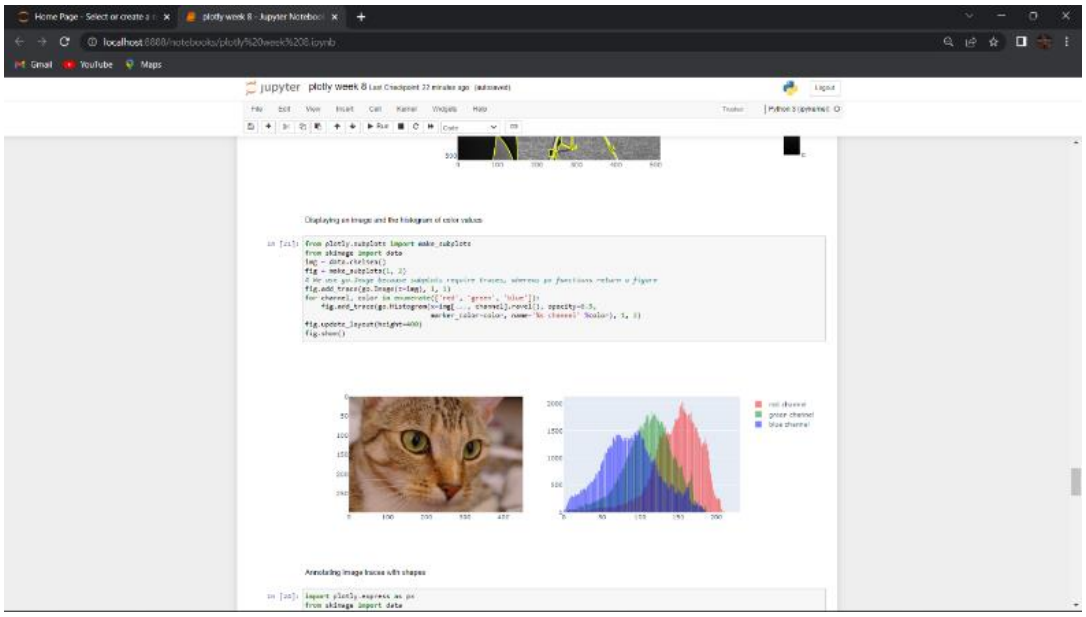

Figure 2.18

For three-dimensional image datasets, obtained for example by MRI or CT in medical imaging, one can explore the dataset by representing its different planes as facets.
The `facet_col` argument specifies along which axis the image is sliced through to make the

facets. Facets can also be used to represent several images of equal shape, like in the example below where different values of the blurring parameter of a Gaussian filter are compared.

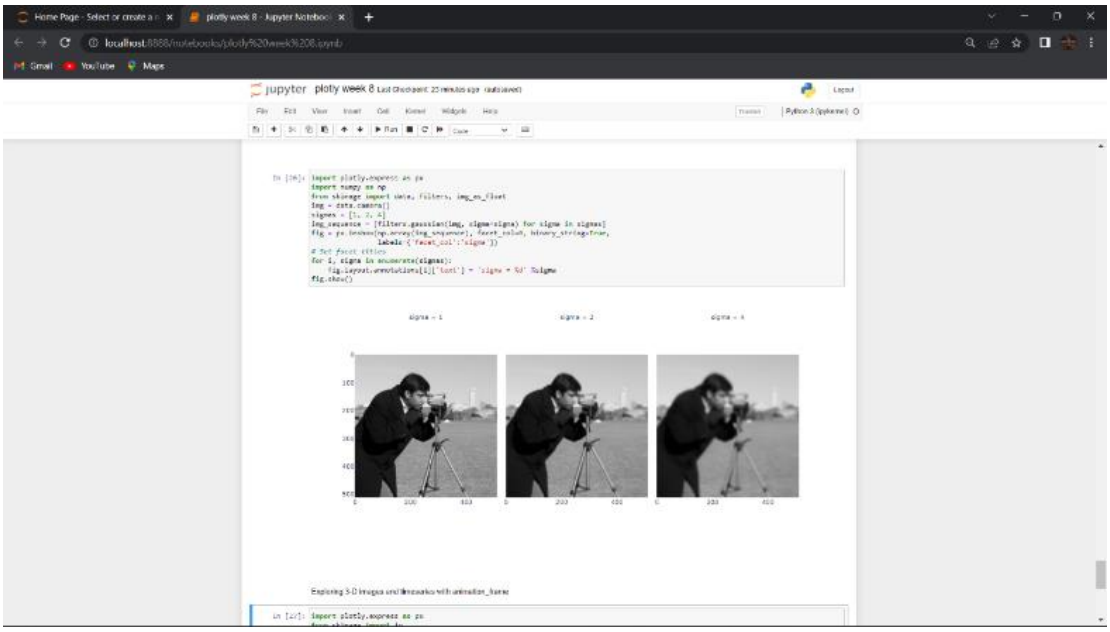

Figure 2.19

3. Difficulties Encountered in Reporting Week

a) Finding suitable classes with sufficient images.
b) An error called 'NoCredentialsError()' was thrown by the python interpreter which took some time to figure out the reason and solve it.
c) Similar types of images found in different classes.

4. Tasks to Be Completed in Next Week

Tasks	Responsible
Implement a function to transfer the uploaded image data to the ML model.	Tomson
Work on Model 03 for better performance	Praveen
Cleansing the dataset, address the imbalance of the dataset and evaluate the model with F1 score.	Thulana
Work on representations for one the model using plotly.	Jaskaran