Machine Learning

TEAM BOLTZMANN MACHINE

The task

1. Create a convolutional neural network with Keras
2. Train that network on the ‘Volcanoes on Venus’ dataset and evaluate it with the provided test set
3. Implement early stopping in order to make the training process more automatic

These are the libraries which we need

- numpy
- pandas
- tensorflow
- tensorflow.keras
- matplotlib.pyplot

to mount the google drive

if it necessary

If you are planning to work online workspace, you need to mount the google drive in connection with your google account.

- from google.colab import drive
- drive.mount('/content/drive')

And you need to give your working path in your drive

- path='/content/drive/My Drive/yourworkingspace/'

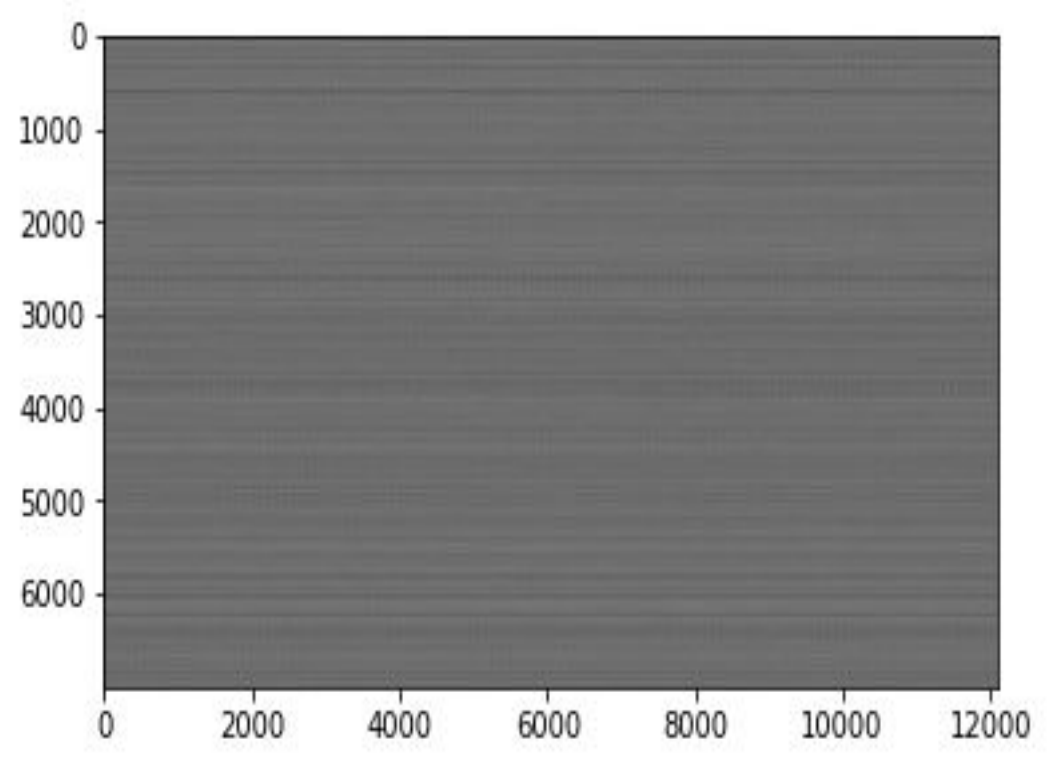
You can read your data in the desired format and assign any variable which you didn't before, and you need reshaping your array.
By reshaping we can add or remove dimensions or change number of elements in each dimension.

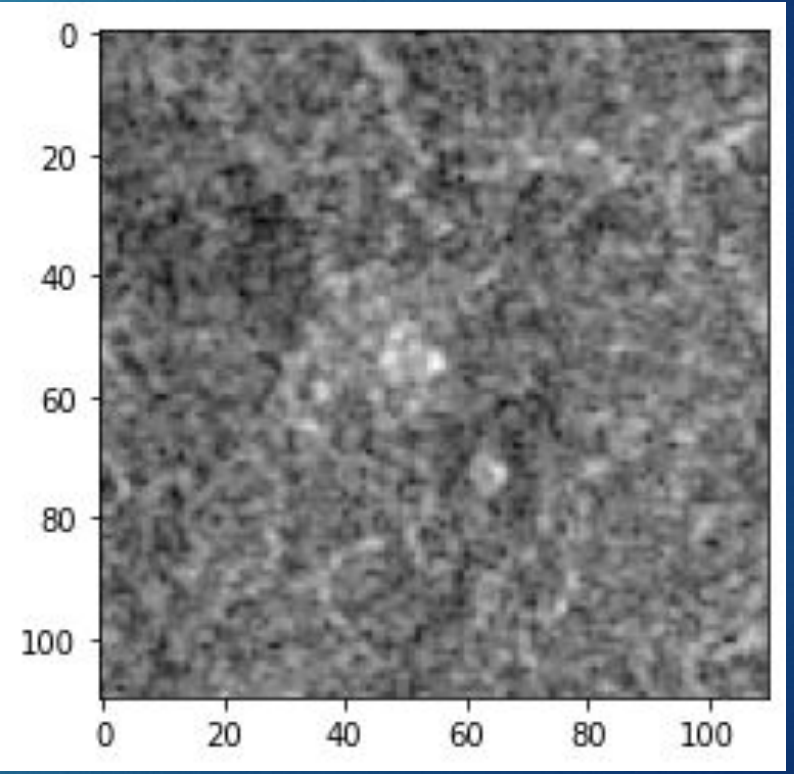
- train = pd.read_csv(path+'train_images.csv',header=None)
- train_lab = pd.read_csv(path+'train_labels.csv')
- test = pd. read_csv(path+'test_images.csv',header=None)
- test_lab = pd.read_csv(path+'test_label.csv')

- x_train = np.reshape(train.values, (7000, 110, 110))
- x_test = np.reshape(test.values, (2734, 110, 110))
- y_train = train_lab['Volcano?'].values
- y_test = test_lab['Volcano?'].values

raw image and reshaped image

plt.imshow(train, cmap='gray')
plt.imshow(x_train[0], cmap='gray')





The task part 2

1. Scale the values
2. Expand dimensions to tell Keras how many channels you have (in this case 1, because it is a grayscale image)

scaling and expanding all images convert class vectors to binary class matrices

- x_train = x_train.astype('float32') / 255.
- x_test = x_test.astype('float32') / 255.

- x_train = np.expand_dims(x_train, -1)
- x_test = np.expand_dims(x_test, -1)

- y_train = keras.utils.to_categorical(y_train, num_classes)
- y_test = keras.utils.to_categorical(y_test, num_classes)

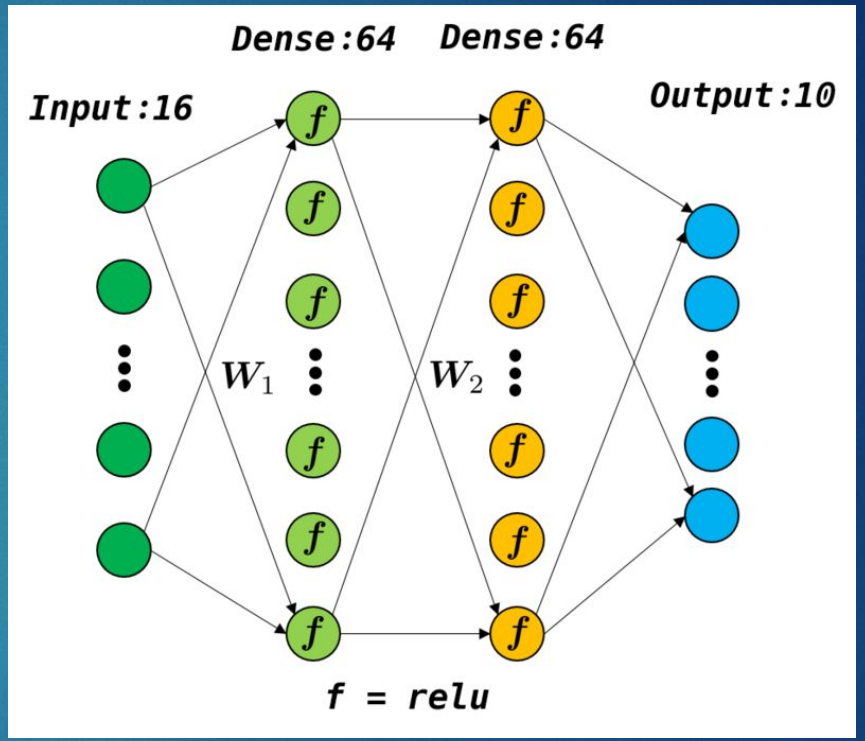
- x_train.shape = (7000, 110, 110, 1)
- x_test.shape = (2734, 110, 110, 1)

- y_train.shape = (7000,2)
- y_test.shape = (2734, 2)

- y_train[0], y_test[0] = [0. 1.], [1. 0.]

Creating the model

```
model = keras.Sequential(
    [
        keras.Input(shape=(110 , 110, 1)),
        layers.Conv2D(32, kernel_size=(3, 3), padding='same',activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), padding='same',activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dense(num_classes, activation="sigmoid"),
    ]
)
```



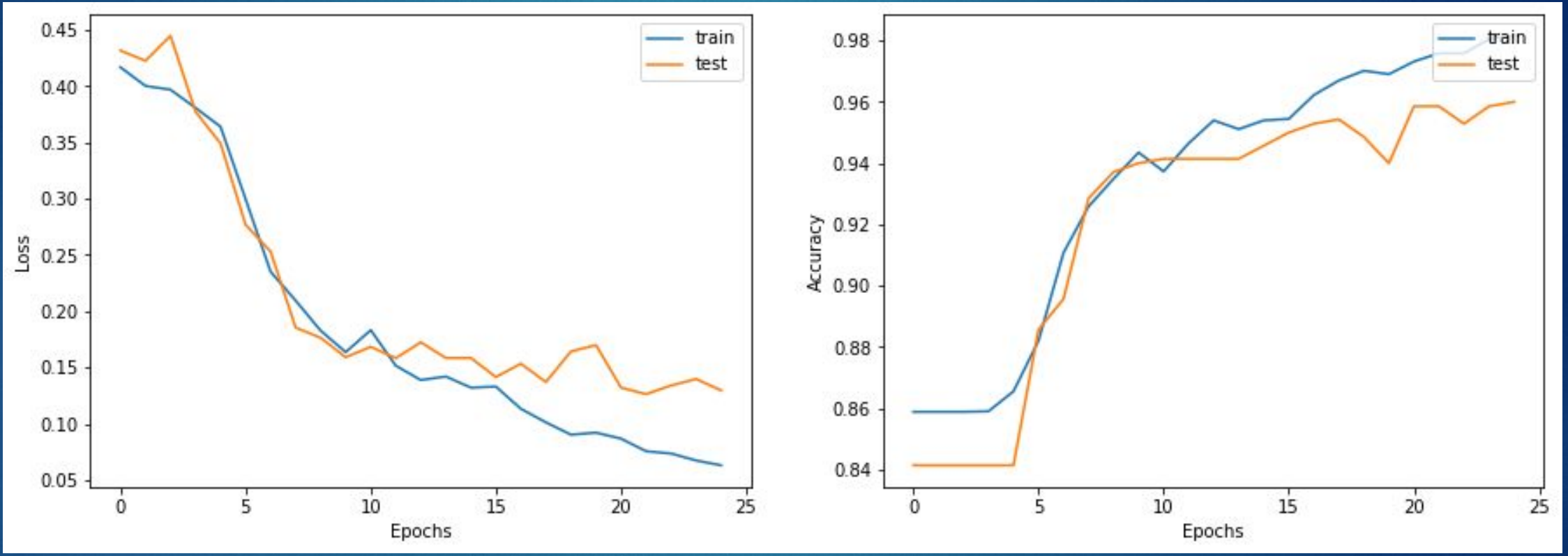
Model summary

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 110, 110, 32)	320
max_pooling2d (MaxPooling2D)	(None, 55, 55, 32)	0
conv2d_1 (Conv2D)	(None, 55, 55, 64)	18496
max_pooling2d_1 (MaxPooling2	(None, 27, 27, 64)	0
flatten (Flatten)	(None, 46656)	0
dense (Dense)	(None, 2)	93314

Total params: 112,130
Trainable params: 112,130
Non-trainable params: 0

Evaluated model history

Test loss: 0.15228846669197083
Test accuracy: 95.39%



THANK YOU