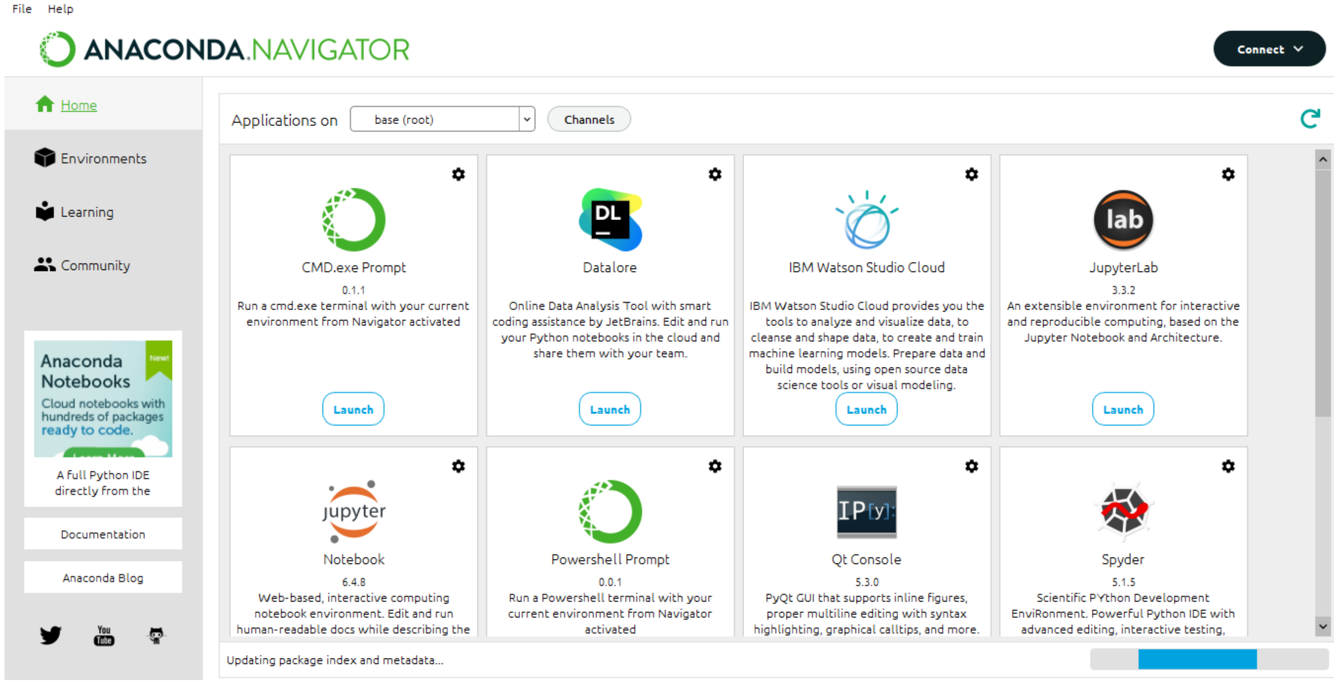
PROJECT DEVELOPMENT PHASE

DELIVERY OF SPRINT-2

DATE:	05 NOV 2022
TEAM ID:	PNT2022TMID39878
PROJECT:	A GESTURE BASED TOOL FOR STERILE BROWSING OF RADIOLOGY IMAGES
MAXIMUM MARKS:	

PREREQUISITES:

For this project we must download and install anaconda navigator, python, Jupyter notebook and pip libraries.



Cloud notebooks with hundreds of packages ready to code.

A full Python IDE directly from the

Run a cmd.exe terminal with your current environment from Navigator activated

Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the

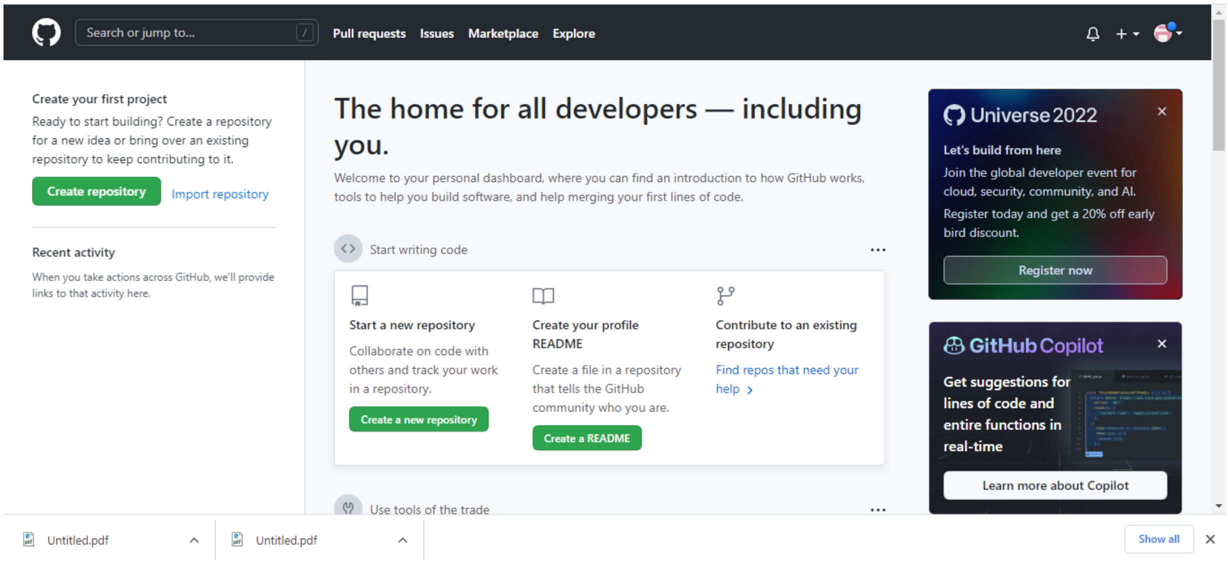
Run a Powershell terminal with your current environment from Navigator activated

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing,

Updating package index and metadata...

CREATING A GIT HUB ACCOUNT:



Create your first project

Ready to start building? Create a repository for a new idea or bring over an existing repository to keep contributing to it.

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

The home for all developers — including you.

Welcome to your personal dashboard, where you can find an introduction to how GitHub works, tools to help you build software, and help merging your first lines of code.

Collaborate on code with others and track your work in a repository.

Create a file in a repository that tells the GitHub community who you are.

Let's build from here

Join the global developer event for cloud, security, community, and AI.

Register today and get a 20% off early bird discount.

Get suggestions for lines of code and entire functions in real-time

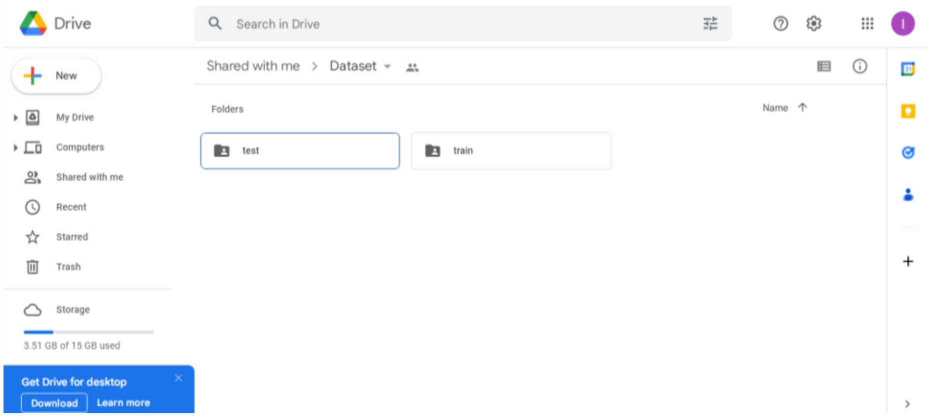
PRIOR KNOWLEDGE:

Understand and learn about the deep learning concepts such as;

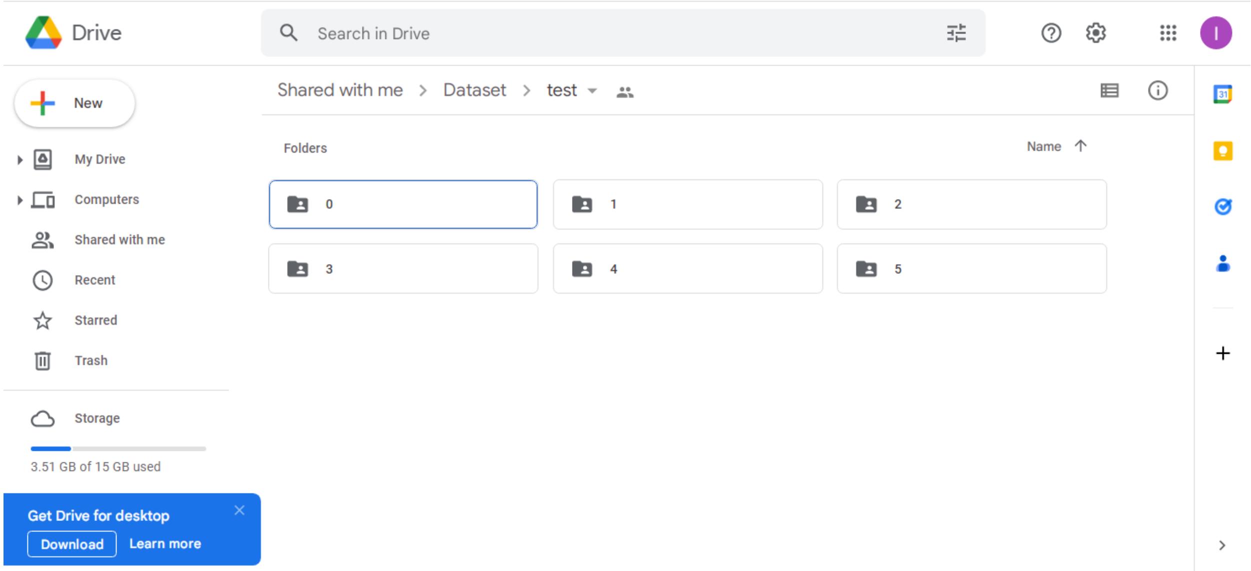
1. CNN
2. OpenCV
3. Flask

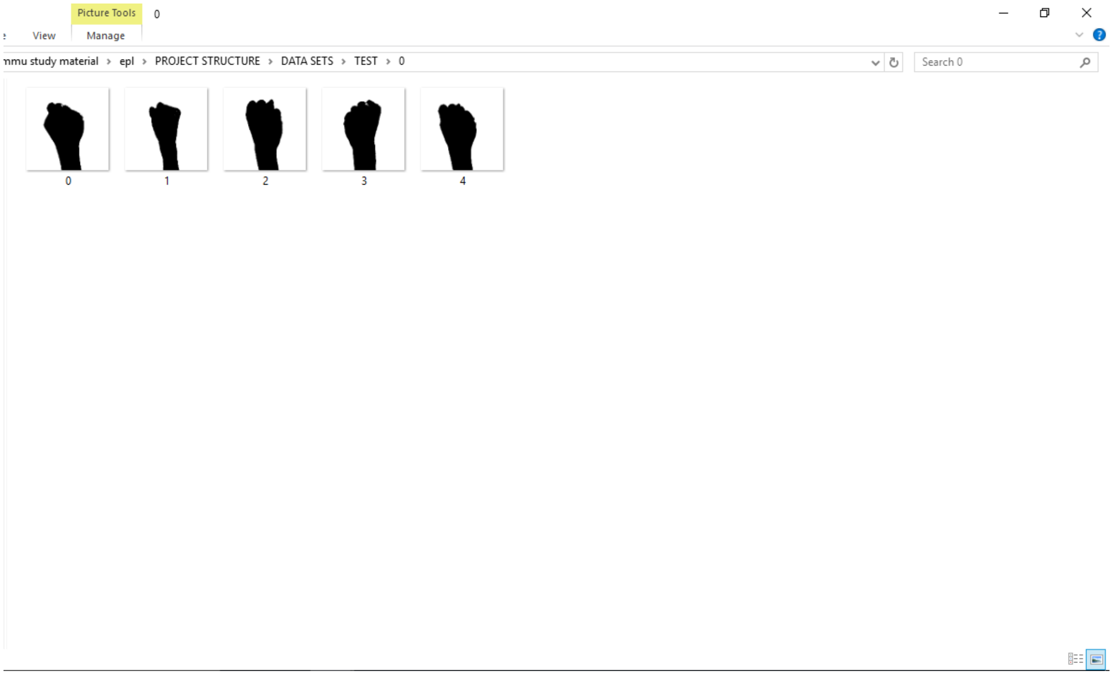
DATA COLLECTION:

Collect the data sets required. Create two different folders for test data and train data.

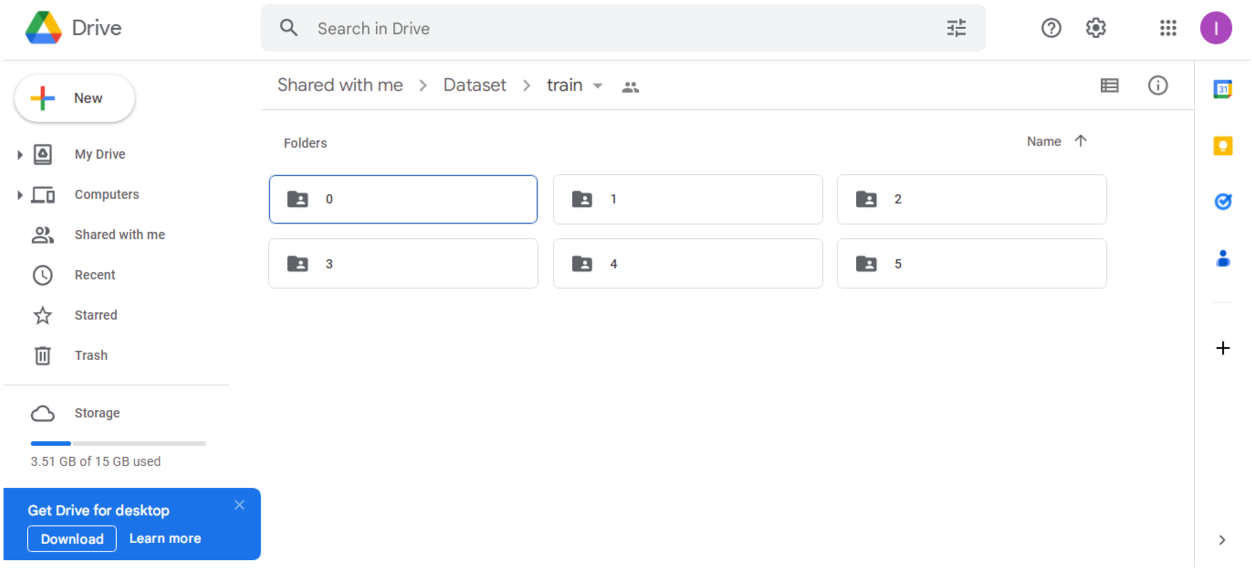


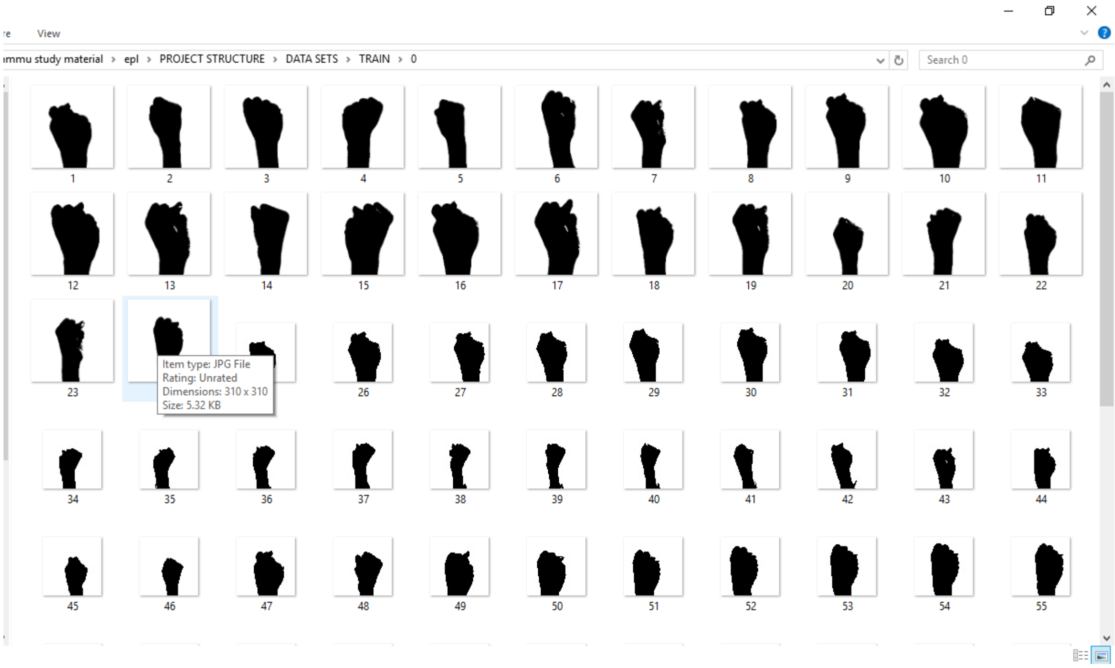
TEST DATA:


Drive
Search in Drive
New
My Drive
Computers
Shared with me
Recent
Starred
Trash
Storage
3.51 GB of 15 GB used
Get Drive for desktop
Download
Learn more
Shared with me > Dataset > test
Folders
Name
0
1
2
3
4
5


Picture Tools
0
View
Manage
nmu study material > epl > PROJECT STRUCTURE > DATA SETS > TEST > 0
Search 0
0
1
2
3
4

TRAIN DATA:



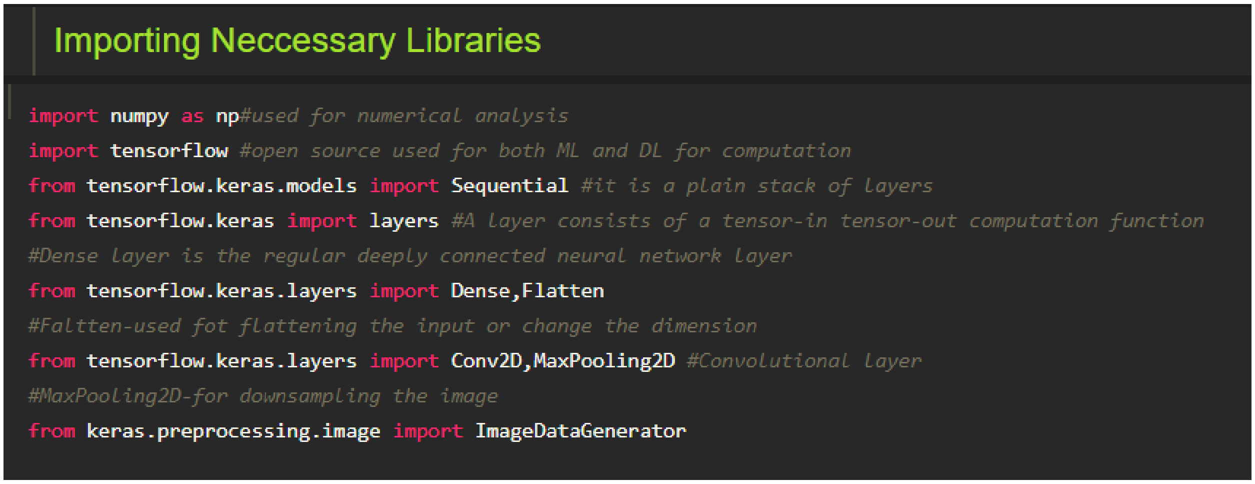


Item type: JPG File
Rating: Unrated
Dimensions: 310 x 310
Size: 5.32 KB

MODEL BUILDING:

Model building involves a chain of tasks to be completed like

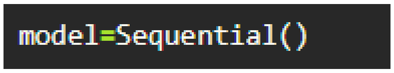
1. Importing model building libraries



Importing Neccessary Libraries

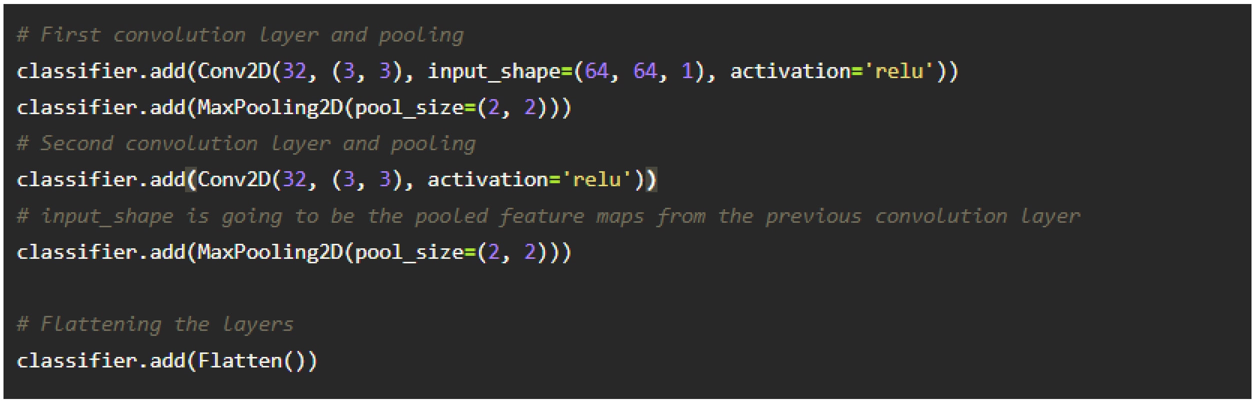
```
import numpy as np#used for numerical analysis
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A layer consists of a tensor-in tensor-out computation function
#Dense layer is the regular deeply connected neural network layer
from tensorflow.keras.layers import Dense,Flatten
#Faltten-used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D,MaxPooling2D #Convolutional layer
#MaxPooling2D-for downsampling the image
from keras.preprocessing.image import ImageDataGenerator
```

2. Initializing the model



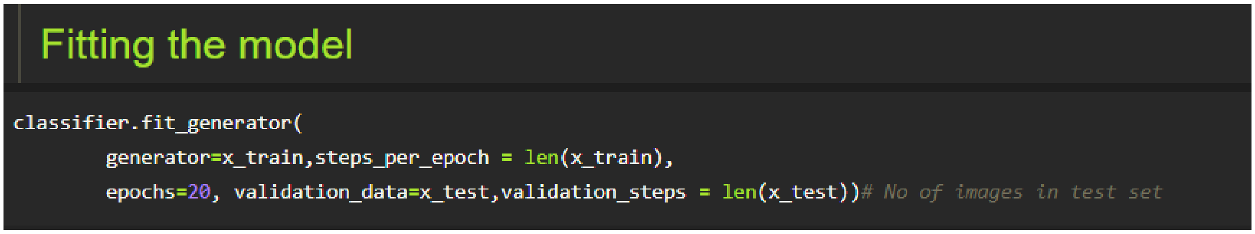
```
model=Sequential()
```

3. Adding CNN layers



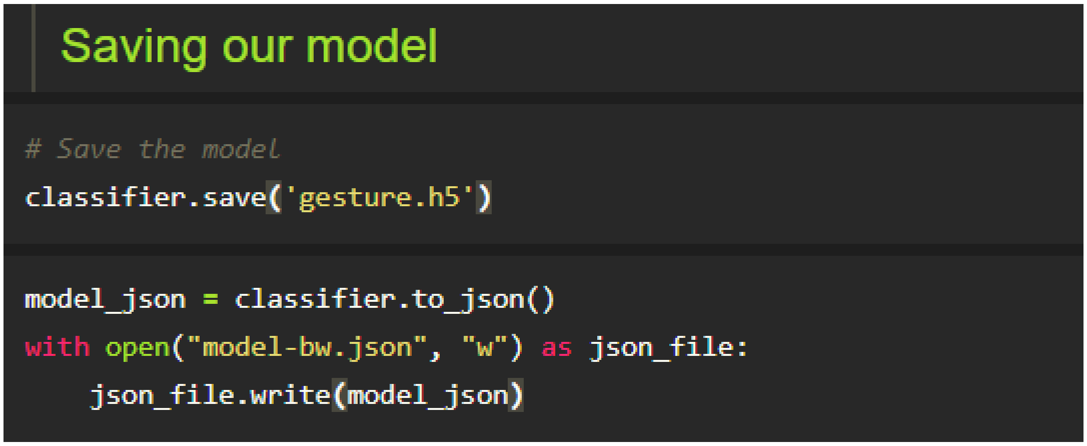
```
# First convolution layer and pooling
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 1), activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
# Second convolution layer and pooling
classifier.add(Conv2D(32, (3, 3), activation='relu'))
# input_shape is going to be the pooled feature maps from the previous convolution layer
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Flattening the layers
classifier.add(Flatten())
```

4. Train, save and test the model



Fitting the model

```
classifier.fit_generator(
        generator=x_train,steps_per_epoch = len(x_train),
        epochs=20, validation_data=x_test,validation_steps = len(x_test))# No of images in test set
```

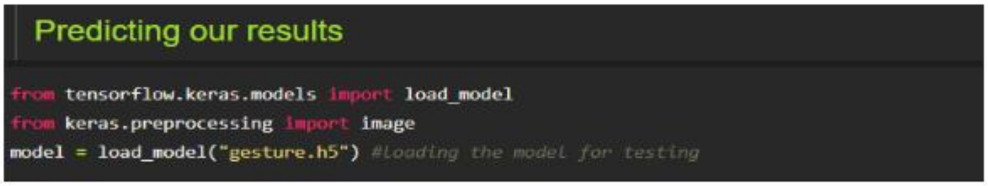


Saving our model

```
# Save the model
classifier.save('gesture.h5')
```

```
model_json = classifier.to_json()
with open("model-bw.json", "w") as json_file:
    json_file.write(model_json)
```

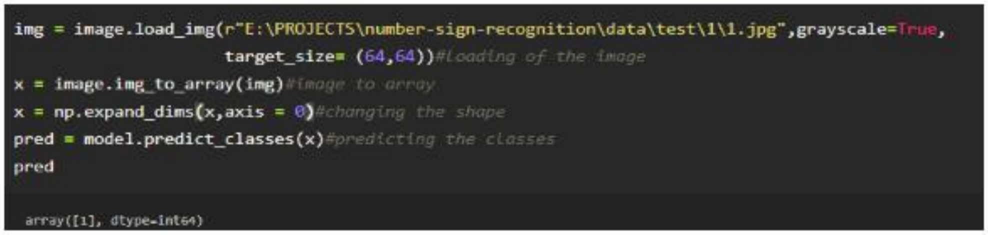
Load the saved model using load_model



Predicting our results

```
from tensorflow.keras.models import load_model
from keras.preprocessing import image
model = load_model("gesture.h5") #Loading the model for testing
```

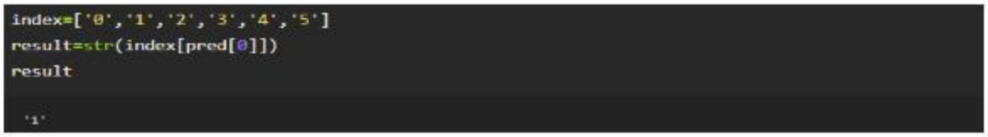
Taking an image as input and checking the results



```
img = image.load_img(r"E:\PROJECTS\number-sign-recognition\data\test\1\1.jpg",grayscale=True,
                     target_size= (64,64))#Loading of the image
x = image.img_to_array(img)#image to array
x = np.expand_dims(x,axis = 0)#changing the shape
pred = model.predict_classes(x)#predicting the classes
pred
```

```
array([1], dtype=int64)
```

By using the model we are predicting the output for the given input image



```
index=['0','1','2','3','4','5']
result=str(index[pred[0]])
result
```

```
'1'
```

The predicted class index name will be printed here.