

Project Development Phase

Sprint – 3

Date	11 November 2022
Team ID	PNT2022TMID21516
Project Name	AI-powered Nutrition Analyzer for Fitness Enthusiasts

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

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[ ] #import keras libraries
    from keras.models import Sequential
    from keras.layers import Dense
    from keras.layers import Convolution2D
    from keras.layers import MaxPooling2D
    from keras.layers import Flatten
```

```
#image preprocessing(or) image augmentation
from keras.preprocessing.image import ImageDataGenerator

[ ] train_datagen = ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True, vertical_flip=True)
    #rescale => rescaling pixel value from 0 to 255 to 0 to 1
    #shear_range=> counter clock wise rotation(anti clock)
```

```
[ ] test_datagen = ImageDataGenerator(rescale=1./255)

[ ] x_train = train_datagen.flow_from_directory("/content/drive/MyDrive/Sri's IBM project/TRAIN_SET", target_size=(64,64), batch_size=32, class_mode="t
Found 2610 images belonging to 5 classes.
+ Code + Text
[ ] x_test = test_datagen.flow_from_directory("/content/drive/MyDrive/Sri's IBM project/TEST_SET", target_size=(64,64), batch_size=32, class_mode="bin
Found 1055 images belonging to 5 classes.
```

```
[ ] x_test = test_datagen.flow_from_directory("/content/drive/MyDrive/Sri's IBM project/TEST_SET", target_size=(64,64), batch_size=32, class_mode="bin
Found 1055 images belonging to 5 classes.

[ ] x_train.class_indices

{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}
```

```
[ ] from collections import Counter as c
    c(x_train.labels)

Counter({0: 606, 1: 445, 2: 479, 3: 621, 4: 459})
```

```
[ ] #Initializing the model
model = Sequential()

▶ # add First convolution layer
model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))
# 32 indicates => no of feature detectors
#(3,3)=> kernel size (feature detector size)

[ ] # add Maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))

[ ] #Second convolution layer and pooling
model.add(Convolution2D(32,(3,3),activation='relu'))

[ ] model.add(MaxPooling2D(pool_size=(2,2)))
#Flattening the layers
model.add(Flatten())
model.add(Dense(units=128,activation='relu'))
model.add(Dense(units=5,activation='softmax'))

[ ] # add flatten layer => input to your ANN
model.add(Flatten())
model.summary()
```

Model: "sequential"

```
▶ # add flatten layer => input to your ANN
model.add(Flatten())
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
conv2d_1 (Conv2D)	(None, 29, 29, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 32)	0
flatten (Flatten)	(None, 6272)	0
dense (Dense)	(None, 128)	802944
dense_1 (Dense)	(None, 5)	645
flatten_1 (Flatten)	(None, 5)	0

=====
 Total params: 813,733
 Trainable params: 813,733
 Non-trainable params: 0
 =====

```
[ ] # adding dense layer
model.add(Dense(units=300,kernel_initializer="random_uniform",activation="relu"))
model.add(Dense(units=200,kernel_initializer="random_uniform",activation="relu"))
```

```
[ ] #output layer
model.add(Dense(units=4,kernel_initializer="random_uniform",activation="softmax"))
len(x_train)
```

82

```
▶ #Ann starts so need to add dense layers
model.add(Dense(units=128,activation="relu",kernel_initializer="random_uniform"))
model.add(Dense(units=1,activation="sigmoid",kernel_initializer="random_uniform"))
```

```
▶ #Compile the model
model.compile(loss="binary_crossentropy",optimizer="adam",metrics=['accuracy'])
```

```
▶ #Train the model
model.fit_generator(x_train,steps_per_epoch=len(x_train), validation_data=x_test, validation_steps=len(x_test), epochs= 20)
```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future

```
Epoch 1/20
82/82 [=====] - 709s 9s/step - loss: 0.1100 - accuracy: 0.1736 - val_loss: -1.0631 - val_accuracy: 0.2720
Epoch 2/20
82/82 [=====] - 24s 291ms/step - loss: -2.8702 - accuracy: 0.1705 - val_loss: -5.4632 - val_accuracy: 0.2720
Epoch 3/20
82/82 [=====] - 22s 263ms/step - loss: -8.3526 - accuracy: 0.1705 - val_loss: -12.8106 - val_accuracy: 0.2720
Epoch 4/20
82/82 [=====] - 24s 288ms/step - loss: -16.9286 - accuracy: 0.1705 - val_loss: -23.7377 - val_accuracy: 0.2720
Epoch 5/20
82/82 [=====] - 22s 263ms/step - loss: -28.8029 - accuracy: 0.1705 - val_loss: -37.9295 - val_accuracy: 0.2720
Epoch 6/20
82/82 [=====] - 23s 283ms/step - loss: -44.0002 - accuracy: 0.1705 - val_loss: -55.4858 - val_accuracy: 0.2720
Epoch 7/20
82/82 [=====] - 22s 264ms/step - loss: -62.2679 - accuracy: 0.1705 - val_loss: -76.4685 - val_accuracy: 0.2720
Epoch 8/20
82/82 [=====] - 24s 286ms/step - loss: -83.5602 - accuracy: 0.1705 - val_loss: -100.6702 - val_accuracy: 0.2720
Epoch 9/20
82/82 [=====] - 22s 263ms/step - loss: -107.7657 - accuracy: 0.1705 - val_loss: -127.5378 - val_accuracy: 0.2720
Epoch 10/20
82/82 [=====] - 21s 256ms/step - loss: -134.6819 - accuracy: 0.1705 - val_loss: -157.5612 - val_accuracy: 0.2720
Epoch 11/20
82/82 [=====] - 21s 259ms/step - loss: -164.3762 - accuracy: 0.1705 - val_loss: -189.9892 - val_accuracy: 0.2720
Epoch 12/20
82/82 [=====] - 24s 292ms/step - loss: -196.3868 - accuracy: 0.1705 - val_loss: -225.3566 - val_accuracy: 0.2720
Epoch 13/20
82/82 [=====] - 22s 265ms/step - loss: -231.1082 - accuracy: 0.1705 - val_loss: -263.1507 - val_accuracy: 0.2720
```

```
[ ] model.save("/content/drive/MyDrive/Sri's IBM project/nutrition.h5")
```

Shared with me > Sri's IBM project ▾ 👤

Name ↑

👤 .ipynb_checkpoints

👤 TEST_SET

👤 TRAIN_SET

📄 nutrition.h5 👤

```
[ ] #Prediction the result
    from tensorflow.keras.models import load_model
    from keras.preprocessing import image
    # model =load_model("/content/drive/MyDrive/Sri's IBM project/nutrition.h5")
```

```
▶ import numpy as np
   from tensorflow.keras.utils import load_img
   from tensorflow.keras.utils import img_to_array
   #loading of the image
   img = load_img(r'/content/drive/MyDrive/Apple.jpg', grayscale=False,target_size=(64,64))
   #image to array
   x = img_to_array(img)
   #changing the shape
   x= np.expand_dims(x,axis = 0)
   predict_x=model.predict(x)
   classes_x=np.argmax(predict_x,axis = -1)
   classes_x
```

1/1 [=====] - 0s 19ms/step
array([0])

```
[ ] index=['APPLES', 'BANANA', 'ORANGE', 'PINEAPPLE', 'WATERMELON']
    result=str(index[classes_x[0]])
    result
```

'APPLES'



sri's_image_classify.ip
ynb

The output file:



nutrition.h5