```
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


from keras.preprocessing.image import ImageDataGenerator


train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizonta


test_datagen = ImageDataGenerator(rescale=1./255)



x_train = train_datagen.flow_from_directory("/content/drive/MyDrive/AI_IBM/Dataset/TRAIN_S

     Found 4119 images belonging to 5 classes. x_test =


test_datagen.flow_from_directory("/content/drive/MyDrive/AI_IBM/Dataset/TEST_SET"


     Found 929 images belonging to 5 classes.


x_train.class_indices

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


print(x_test.class_indices)

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


from collections import Counter as c c(x_train
.labels)

     Counter({0: 995, 1: 1355, 2: 1019, 3: 275, 4: 475})


model = Sequential()


model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))


model.add(MaxPooling2D(pool_size=(2,2)))


model.add(Convolution2D(32,(3,3),activation='relu'))


model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten()) model.add(Dense(units=128,activation='relu'))
model.add(Dense(units=5,activation='softmax'))


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 (MaxPooling  (None, 14, 14, 32)       0
     2D)
     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
     _________________________________________________________________


model.add(Dense(units=300,kernel_initializer="random_uniform",activation="relu"))
model.add(Dense(units=200,kernel_initializer="random_uniform",activation="relu"))


model.add(Dense(units=5,kernel_initializer="random_uniform",activation="softmax"))
len(x_train)

     129


model.add(Dense(units=128,activation="relu",kernel_initializer="random_uniform"))
model.add(Dense(units=1,activation="sigmoid",kernel_initializer="random_uniform"))


model.compile(loss="binary_crossentropy",optimizer="adam",metrics=['accuracy'])


model.fit_generator(x_train,steps_per_epoch=len(x_train), validation_data=x_test, validati
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: `Model.f
       """Entry point for launching an IPython kernel. Epoch
     1/20
     129/129 [==============================] - 42s 323ms/step - loss: -579.1954 - accurac
     Epoch 2/20
     129/129 [==============================] - 35s 272ms/step - loss: -630.7393 - accurac
     Epoch 3/20
     129/129 [==============================] - 35s 273ms/step - loss: -683.9399 - accurac
     Epoch 4/20
     129/129 [==============================] - 35s 274ms/step - loss: -738.6011 - accurac
     Epoch 5/20
     129/129 [==============================] - 36s 275ms/step - loss: -795.0793 - accurac
     Epoch 6/20
```

```
129/129 [==============================] - 37s 286ms/step - loss: -853.5035 -
accuracy Epoch 7/20
129/129 [==============================] - 36s 276ms/step - loss: -913.4440 -
accuracy
Epoch 8/20
129/129 [==============================] - 36s 275ms/step - loss: -974.8712 -
accuracy Epoch 9/20
129/129 [==============================] - 35s 274ms/step - loss: -1037.6532 -
accuracy Epoch 10/20
129/129 [==============================] - 36s 275ms/step - loss: -1101.9432 -
accuracy Epoch 11/20
129/129 [==============================] - 35s 273ms/step - loss: -1167.7832 -
accuracy
Epoch 12/20
129/129 [==============================] - 35s 273ms/step - loss: -1235.0177 -
accuracy Epoch 13/20
129/129 [==============================] - 35s 274ms/step - loss: -1303.9956 -
accuracy Epoch 14/20
129/129 [==============================] - 35s 274ms/step - loss: -1374.5148 -
accuracy
Epoch 15/20
129/129 [==============================] - 36s 276ms/step - loss: -1446.9734 -
accuracy
Epoch 16/20
129/129 [==============================] - 35s 274ms/step - loss: -1520.6868 -
accuracy Epoch 17/20
129/129 [==============================] - 35s 273ms/step - loss: -1596.1498 -
accuracy Epoch 18/20
129/129 [==============================] - 35s 271ms/step - loss: -1673.0337 -
accuracy
Epoch 19/20
129/129 [==============================] - 35s 273ms/step - loss: -1751.5466 -
accuracy Epoch 20/20
129/129 [==============================] - 35s 270ms/step - loss: -1831.8647 - accura
<keras.callbacks.History at 0x7f60240c4c10>
```

```
model.save("nutrition.h5")
```

```
from tensorflow.keras.models import load_model
from keras.preprocessing  import image model
=load_model("nutrition.h5")
```

```
import numpy as np
```

```
from tensorflow.keras.utils import load_img from tensorflow.keras.utils import
img_to_array img = load_img(r'/content/drive/MyDrive/AI_IBM/Nutrition Analysis Using Image
Classificati x = img_to_array(img)

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 424ms/step array([0])
```

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

```
     'APPLES'
```