```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "-4U2x7XApAPv"
   },
   "outputs": [],
   "source": [
    "#import keras libraries\n",
    "from keras.models import Sequential\n",
    "from keras.layers import Dense\n",
    "from keras.layers import Convolution2D\n",
    "from keras.layers import MaxPooling2D\n",
    "from keras.layers import Flatten"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "GUqs8zuap0Ro"
   },
   "outputs": [],
  "source": [
    "#image preprocessing(or) image augmentation\n",
    "from keras.preprocessing.image import ImageDataGenerator"
  ]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "t44vJdxpqO67"
  },
  "outputs": [],
  "source": [
    "train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True,vertical_flip=True)\n",
    "#rescale => rescaling pixel value from 0 to 255 to 0 to 1\n",
    "#shear_range=> counter clock wise rotation(anti clock)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "bPtjB_31qZLl"
   },
   "outputs": [],
   "source": [
    "test_datagen = ImageDataGenerator(rescale=1./255)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "colab": {
     "base_uri": "https://localhost:8080/"
    },
    "id": "ltTuui5Kqdtp",
 "outputId": "2f168c3f-c51e-4c92-dc28-3d4ea011d4da"
},
"outputs": [
 {
  "output_type": "stream",
  "name": "stdout",
  "text": [
   "Found 4118 images belonging to 5 classes.\n"
  ]
 }
],
"source": [
    "x_train = train_datagen.flow_from_directory(\"/content/drive/MyDrive/ibm project/TRAIN_SET\",target_size=(64,64),batch_size=32,class_mode=\"binary\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "colab": {
     "base_uri": "https://localhost:8080/"
    },
    "id": "U9WzDTJHuiAh",
    "outputId": "87f6e98f-1cba-473a-b803-faa60d4eeb7d"
   },
   "outputs": [
    {
     "output_type": "stream",
     "name": "stdout",
     "text": [
      "Found 929 images belonging to 3 classes.\n"
     ]
    }
   ],
   "source": [
    "x_test = test_datagen.flow_from_directory(\"/content/drive/MyDrive/ibm project/TEST_SET\",target_size=(64,64),batch_size=32,class_mode=\"binary\")"
```

```
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
   "colab": {
    "base_uri": "https://localhost:8080/"
   },
   "id": "bApCdADGup8T",
   "outputId": "d57ab51e-f9c3-47b2-f19c-f25f10a7aec7"
  },
  "outputs": [
    {
     "output_type": "execute_result",
     "data": {
      "text/plain": [
       "{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}"
      ]
     },
     "metadata": {},
     "execution_count": 7
    }
   ],
   "source": [
  "x_train.class_indices"
 ]
},
{
 "cell_type": "code",
 "source": [
  "#checking the number of classes\n",
  "print(x_test.class_indices)"
 ],
 "metadata": {
  "colab": {
   "base_uri": "https://localhost:8080/"
  },
  "id": "9A3kmlgHz0Q7",
  "outputId": "d2e6daaa-dbe2-4552-ef65-d5e8bbe0d9ea"
},
"execution_count": null,
"outputs": [
  {
    "output_type": "stream",
    "name": "stdout",
    "text": [
      "{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2}\n"
    ]
  }
```

```
   ]
  },
  {
   "cell_type": "code",
   "source": [
    "from collections import Counter as c\n",
    "c(x_train .labels)"
   ],
   "metadata": {
    "colab": {
     "base_uri": "https://localhost:8080/"
    },
    "id": "yGeKS68E0bSP",
 "outputId": "cd5bac4d-ffb6-464b-d6f0-841ef62e776d"
},
"execution_count": null,
"outputs": [
 {
  "output_type": "execute_result",
  "data": {
   "text/plain": [
    "Counter({0: 995, 1: 1354, 2: 1019, 3: 275, 4: 475})"
   ]
  },
  "metadata": {},
  "execution_count": 11
```

```
    }
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "dx_5gTSAu0hY"
   },
   "outputs": [],
   "source": [
    "#Initializing the model\n",
    "model = Sequential()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "ufSbk5LVu9qU"
   },
   "outputs": [],
   "source": [
    "# add First convolution layer"
   ]
```

```
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "62dYvr9WvHlF"
  },
  "outputs": [],
  "source": [
    "model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation=\"relu\"))\n",
    "# 32 indicates => no of feature detectors\n",
    "#(3,3)=> kernel size (feature detector size)"
  ]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "0RoS09jlvROB"
  },
  "outputs": [],
  "source": [
    "# add Maxpooling layer"
  ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "7tIjlFq_vaMc"
 },
 "outputs": [],
 "source": [
  "model.add(MaxPooling2D(pool_size=(2,2)))"
 ]
},
{
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "lnioOB-s9CaM"
   },
   "outputs": [],
   "source": [
    "#Second convolution layer and pooling\n",
    "model.add(Convolution2D(32,(3,3),activation='relu'))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "bAcEug9x-Rqm"
   },
   "outputs": [],
   "source": [
    "model.add(MaxPooling2D(pool_size=(2,2)))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
  "metadata": {
   "id": "hFOgQQQb_Inn"
  },
  "outputs": [],
  "source": [
   "#Flattening the layers\n",
   "model.add(Flatten())"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "v1LSVWYs_g2v"
   },
   "outputs": [],
   "source": [
    "model.add(Dense(units=128,activation='relu'))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "DKg4TBZZ_zT6"
   },
   "outputs": [],
   "source": [
    "model.add(Dense(units=5,activation='softmax'))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "eCB4ZIxOvh4G"
   },
   "outputs": [],
  "source": [
    "# add flatten layer => input to your ANN"
  ]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "agjb4SXivnq_"
  },
  "outputs": [],
  "source": [
    "model.add(Flatten())"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
   "colab": {
    "base_uri": "https://localhost:8080/"
   },
   "id": "fGDMWXyMwSWs",
   "outputId": "e6a3a789-c1aa-406c-886a-6a40f77b71b7"
  },
"outputs": [
 {
  "output_type": "stream",
  "name": "stdout",
  "text": [
   "Model: \"sequential\"\n",
   "_________________________________________________________________\n",
   " Layer (type)                Output Shape              Param #   \n",
   "=================================================================\n",
   " conv2d (Conv2D)             (None, 62, 62, 32)        896       \n",
   "                                                                 \n",
   " max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0         \n",
   " )                                                               \n",
"                                                                 \n",
" conv2d_1 (Conv2D)           (None, 29, 29, 32)        9248      \n",
"                                                                 \n",
" max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)      0         \n",
" 2D)                                                             \n",
"                                                                 \n",
" flatten (Flatten)           (None, 6272)              0         \n",
"                                                                 \n",
" dense (Dense)               (None, 128)               802944    \n",
"                                                                 \n",
" dense_1 (Dense)             (None, 5)                 645       \n",
"                                                                 \n",
      " flatten_1 (Flatten)        (None, 5)                 0         \n",
      "                                                                 \n",
      "=================================================================\n",
      "Total params: 813,733\n",
      "Trainable params: 813,733\n",
      "Non-trainable params: 0\n",
      "__________________________________________________________________\n"
     ]
    }
   ],
   "source": [
    "model.summary()"
   ]
```

```
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "EQirf5FewdjE"
 },
 "outputs": [],
 "source": [
  "# adding dense layer"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "2tPWSWhNwgGB"
 },
 "outputs": [],
 "source": [
  "#hidden layer"
 ]
},
{
 "cell_type": "code",
  "execution_count": null,
  "metadata": {
   "id": "gE4dkAxfwlQU"
  },
  "outputs": [],
  "source": [
   "model.add(Dense(units=300,kernel_initializer=\"random_uniform\",activation=\"relu\"))"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
 "metadata": {
  "id": "Qa_XY5iiwwnX"
 },
 "outputs": [],
 "source": [
  "model.add(Dense(units=200,kernel_initializer=\"random_uniform\",activation=\"relu\"))"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "LK3wwTiKw5D0"
```

```
   },
   "outputs": [],
   "source": [
    "#output layer"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "colab": {
     "base_uri": "https://localhost:8080/"
  },
  "id": "0tEhMxf-w9mU",
  "outputId": "75ff58d8-a81d-4a9e-d08b-669a7ad64c10"
},
"outputs": [
  {
    "output_type": "execute_result",
    "data": {
      "text/plain": [
        "129"
      ]
    },
    "metadata": {},
   "execution_count": 30
  }
 ],
 "source": [
  "model.add(Dense(units=4,kernel_initializer=\"random_uniform\",activation=\"softmax\"))\n",
  "len(x_train)"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
    "id": "yV6nAWK2xC2e"
   },
   "outputs": [],
   "source": [
    "#Ann starts so need to add dense layers"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "ej3QucuhxImk"
   },
  "outputs": [],
  "source": [
   "model.add(Dense(units=128,activation=\"relu\",kernel_initializer=\"random_uniform\"))"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
   "id": "f_cjd0eTxXa1"
  },
  "outputs": [],
   "source": [
    "model.add(Dense(units=1,activation=\"sigmoid\",kernel_initializer=\"random_uniform\"))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "q846LaeFx3BK"
   },
   "outputs": [],
   "source": [
    "#Compile the model\n",
    "model.compile(loss=\"binary_crossentropy\",optimizer=\"adam\",metrics=['accuracy'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "4fAss-XEyHCe"
   },
   "outputs": [],
   "source": [
    "#Train the model"
   ]
```

```
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "colab": {
   "base_uri": "https://localhost:8080/"
  },
  "id": "hgVQdW_cyb9l",
  "outputId": "01e2b5a1-f81a-4547-bf21-21e5814100dc"
 },
 "outputs": [
{
 "metadata": {
  "tags": null
 },
 "name": "stderr",
 "output_type": "stream",
 "text": [
  "/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.\n",
  "  \"\"\"Entry point for launching an IPython kernel.\n"
 ]
},
{
     "output_type": "stream",
     "name": "stdout",
     "text": [
      "Epoch 1/20\n",
      "129/129 [==============================] - 2459s 19s/step - loss: -0.0526 - accuracy: 0.3273 - val_loss: 0.1126 - val_accuracy: 0.4467\n",
      "Epoch 2/20\n",
      "129/129 [==============================] - 36s 277ms/step - loss: -3.0746 - accuracy: 0.3288 - val_loss: 0.2155 - val_accuracy: 0.4467\n",
      "Epoch 3/20\n",
      "129/129 [==============================] - 35s 268ms/step - loss: -8.7866 - accuracy: 0.3288 - val_loss: 0.5095 - val_accuracy: 0.4467\n",
      "Epoch 4/20\n",
      "129/129 [==============================] - 36s 281ms/step - loss: -17.7107 - accuracy: 0.3288 - val_loss: 0.9337 - val_accuracy: 0.4467\n",
    "Epoch 5/20\n",
    "129/129 [==============================] - 36s 282ms/step - loss: -29.8704 - accuracy: 0.3288 - val_loss: 1.4811 - val_accuracy: 0.4467\n",
    "Epoch 6/20\n",
    "129/129 [==============================] - 36s 277ms/step - loss: -45.0273 - accuracy: 0.3288 - val_loss: 2.1422 - val_accuracy: 0.4467\n",
    "Epoch 7/20\n",
    "129/129 [==============================] - 35s 269ms/step - loss: -62.9152 - accuracy: 0.3288 - val_loss: 2.9106 - val_accuracy: 0.4467\n",
    "Epoch 8/20\n",
    "129/129 [==============================] - 40s 309ms/step - loss: -83.5868 - accuracy: 0.3288 - val_loss: 3.7855 - val_accuracy: 0.4467\n",
    "Epoch 9/20\n",
    "129/129 [==============================] - 36s 281ms/step - loss: -106.7443 - accuracy: 0.3288 - val_loss: 4.7640 - val_accuracy: 0.4467\n",
    "Epoch 10/20\n",
    "129/129 [==============================] - 36s 278ms/step - loss: -132.3641 - accuracy: 0.3288 - val_loss: 5.8398 - val_accuracy: 0.4467\n",
    "Epoch 11/20\n",
    "129/129 [==============================] - 35s 271ms/step - loss: -160.3758 - accuracy: 0.3288 - val_loss: 7.0081 - val_accuracy: 0.4467\n",
    "Epoch 12/20\n",
    "129/129 [==============================] - 35s 269ms/step - loss: -190.6966 - accuracy: 0.3288 - val_loss: 8.2454 - val_accuracy: 0.4467\n",
    "Epoch 13/20\n",
    "129/129 [==============================] - 36s 279ms/step - loss: -223.1146 - accuracy: 0.3288 - val_loss: 9.6145 - val_accuracy: 0.4467\n",
    "Epoch 14/20\n",
    "129/129 [==============================] - 36s 280ms/step - loss: -257.9082 - accuracy: 0.3288 - val_loss: 11.0088 - val_accuracy: 0.4467\n",
    "Epoch 15/20\n",
    "129/129 [==============================] - 37s 290ms/step - loss: -294.5687 - accuracy: 0.3288 - val_loss: 12.5175 - val_accuracy: 0.4467\n",
    "Epoch 16/20\n",
    "129/129 [==============================] - 34s 266ms/step - loss: -333.2441 - accuracy: 0.3288 - val_loss: 14.1130 - val_accuracy: 0.4467\n",
    "Epoch 17/20\n",
    "129/129 [==============================] - 36s 279ms/step - loss: -374.0325 - accuracy: 0.3288 - val_loss: 15.7641 - val_accuracy: 0.4467\n",
    "Epoch 18/20\n",
    "129/129 [==============================] - 36s 278ms/step - loss: -416.7053 - accuracy: 0.3288 - val_loss: 17.5287 - val_accuracy: 0.4467\n",
    "Epoch 19/20\n",
    "129/129 [==============================] - 35s 267ms/step - loss: -461.2285 - accuracy: 0.3288 - val_loss: 19.3238 - val_accuracy: 0.4467\n",
      "Epoch 20/20\n",
      "129/129 [==============================] - 34s 265ms/step - loss: -507.5266 - accuracy: 0.3288 - val_loss: 21.2192 - val_accuracy: 0.4467\n"
     ]
    },
    {
     "output_type": "execute_result",
     "data": {
      "text/plain": [
       "<keras.callbacks.History at 0x7f5c66ea6f50>"
      ]
     },
     "metadata": {},
     "execution_count": 36
    }
   ],
   "source": [
    "model.fit_generator(x_train,steps_per_epoch=len(x_train), validation_data=x_test, validation_steps=len(x_test), epochs=20)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "5nrwRs8k5rSf"
   },
   "outputs": [],
   "source": [
    "model.save(\"nutrition.h5\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "JR93P4teGyAb"
   },
  "outputs": [],
  "source": [
    "#Prediction the result"
  ]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "qCIJVUjdGzw9"
  },
  "outputs": [],
  "source": [
    "from tensorflow.keras.models import load_model\n",
    "from keras.preprocessing  import image\n",
    "model =load_model(\"nutrition.h5\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "2f9AzoEwKLqB"
   },
   "outputs": [],
      "source": [
        "import numpy as np\n"
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "from tensorflow.keras.utils import load_img\n",
        "from tensorflow.keras.utils import img_to_array\n",
        "#loading of the image\n",
        "img = load_img(r'/content/drive/MyDrive/ibm project/Sample_Images-20221102T071233Z-001/Sample_Images/Test_Image3.jpg', grayscale=False,target_size=(64,64))\n",
        "#image to array \n",
 "x = img_to_array(img)\n",
 "#changing the shape\n",
 "x= np.expand_dims(x,axis = 0)\n",
 "predict_x=model.predict(x)\n",
 "classes_x=np.argmax(predict_x,axis = -1)\n",
 "classes_x"
],
"metadata": {
 "colab": {
  "base_uri": "https://localhost:8080/"
 },
 "id": "CPvf0dfowTAL",
 "outputId": "1855f68a-13eb-4a61-9baa-93b3e31eb9f9"
```

```
},
"execution_count": null,
"outputs": [
 {
  "output_type": "stream",
  "name": "stdout",
  "text": [
   "1/1 [==============================] - 0s 166ms/step\n"
  ]
 },
 {
  "output_type": "execute_result",
   "data": {
    "text/plain": [
     "array([0])"
    ]
   },
   "metadata": {},
   "execution_count": 48
  }
 ]
},
{
 "cell_type": "code",
 "source": [
  "index=['APPLES', 'BANANA', 'ORANGE', 'PINEAPPLE', 'WATERMELON']\n",
  "result=str(index[classes_x[0]])\n",
  "result"
],
"metadata": {
  "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 36
  },
  "id": "3LzViysVEDln",
  "outputId": "0c9c54b0-fe74-479e-9a7c-51083f302ff4"
},
"execution_count": null,
"outputs": [
  {
    "output_type": "execute_result",
    "data": {
      "text/plain": [
        "'APPLES'"
      ],
      "application/vnd.google.colaboratory.intrinsic+json": {
        "type": "string"
      }
    },
    "metadata": {},
    "execution_count": 49
   }
  ]
 }
],
"metadata": {
 "colab": {
  "provenance": []
 },
 "kernelspec": {
  "display_name": "Python 3",
  "name": "python3"
  },
  "language_info": {
   "name": "python"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 0
}
```