

# PROJECT DEVELOPMENT PHASE

**SPRINT-4** TRAINING MODEL ON IBM CLOUD

**DATE:** 11/11/2022

TEAM -ID :	PNT2022TMID04836
PROJECT NAME:	AI-powered Nutrition Analyzer for FitnessEnthusiasts

**API KEY:**

The screenshot shows the IBM Cloud IAM 'API keys' management interface. On the left is a dark sidebar with navigation links: IAM, Manage identities, Users, Trusted profiles, Service IDs, API keys (highlighted), Identity providers, Manage access, Access groups, Authorizations, Roles, Gain insight, Inactive identities, Inactive policies, and Settings. The main content area is titled 'API keys' and includes a description: 'Create, view, and work with API keys that you have access to manage. IBM Cloud API keys are associated with a user's identity and can be used to access cloud platform and classic infrastructure APIs, depending on the access that is assigned to the user. The following table displays a list of API keys created in this account. [Learn more.](#)' Below this is a 'View:' dropdown set to 'My IBM Cloud API keys'. A note states: 'API keys associated with a user's identity have the same access that the user is assigned across all accounts. To update the access for an API key, assign or remove access for the user.' A table lists API keys with columns: Status, Name, Description, and Date Created. One key is shown: 'Nutrition-Tracker-APIKEY' with description 'Nutrition-Analysis' and date '2022-11-17 17:28 GMT'. A 'Create' button is in the top right. At the bottom, a pagination bar shows 'Items per page: 25', '1-25 items', and 'Page 1'. A taskbar at the very bottom shows several open documents: 'TRAIN MO....docx', 'Sprint Delive....pdf', 'SPRINT 3.docx.pdf', 'sprint3.docx', 'SPRINT-2.docx.pdf', and 'SPRINT-1.docx.pdf', along with a 'Show all' button.

Status	Name	Description	Date Created
	Nutrition-Tracker-APIKEY	Nutrition-Analysis	2022-11-17 17:28 GMT

## LOADING DATA / LIBRARIES IN IBM CLOUD:

The screenshot shows the IBM Watson Studio interface. The top navigation bar includes the IBM logo, a search bar, and user account information (Mounika S's Account, Dallas). The main workspace displays a Jupyter notebook titled "AI-powered Nutrition Analyzer for Fitness Enthusiasts". The notebook contains the following code:

```
In [ ]: TEAMID : PNT2022TMD04836
PROJECT NAME : AI-powered Nutrition Analyzer for Fitness Enthusiasts

In [1]: !pip install keras
!pip install tensorflow
```

The output of the code execution is displayed below the input cells, showing the installation progress and version information for various dependencies. The output is truncated, but it includes the following information:

- Requirement already satisfied: keras in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (2.7.0)
- Requirement already satisfied: tensorflow in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (2.7.2)
- Requirement already satisfied: tensorflow-estimator<2.8,~=2.7.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (2.7.0)
- Requirement already satisfied: grpcio<2.0,>=1.24.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.42.0)
- Requirement already satisfied: tensorflow-io-gcs-filesystem==0.21.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (0.23.1)
- Requirement already satisfied: keras<2.8,>=2.7.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (2.7.0)
- Requirement already satisfied: termcolor==1.1.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.1.0)
- Requirement already satisfied: numpy>=1.14.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.20.3)
- Requirement already satisfied: typing-extensions>=3.6.6 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (4.1.1)
- Requirement already satisfied: keras-preprocessing>1.1.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.1.2)
- Requirement already satisfied: tensorboard==2.7 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (2.7.0)
- Requirement already satisfied: gast<0.5.0,>=0.2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (0.4.0)
- Requirement already satisfied: opt-einsum>=2.3.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (3.3.0)
- Requirement already satisfied: protobuf>=3.9.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (3.19.1)
- Requirement already satisfied: wheel<1.0,>=0.32.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (0.37.0)
- Requirement already satisfied: flatbuffers<3.0,>=1.12 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (2.0)
- Requirement already satisfied: absl-py>=0.4.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (0.12.0)
- Requirement already satisfied: h5py>=2.9.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (3.2.1)
- Requirement already satisfied: astunparse>=1.6.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.6.3)
- Requirement already satisfied: google-pasta>=0.1.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (0.2.0)
- Requirement already satisfied: wrapt>=1.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.12.1)
- Requirement already satisfied: six>=1.12.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.15.0)

The screenshot shows the IBM Watson Studio interface. The top navigation bar includes the IBM logo, a search bar, and user account information (Mounika S's Account, Dallas). The main workspace displays a Jupyter notebook titled "AI-powered Nutrition Analyzer F... / training model". The notebook contains the following code:

```
In [5]: import numpy as np
import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator

In [6]: train_datagen=ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)

In [7]: import os, types
import pandas as pd
from botocore.client import Config
import boto3

def _iter_(self): return 0

# @hidden cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = boto3.client(service_name='s3',
                           aws_access_key_id='Jt-H2xkr3uqgqKqSLrCgZWT2jyF4V5I8031qSMjg',
                           aws_secret_access_key='https://iam.cloud.ibm.com/oidc/token',
                           config=Config(signature_version='oauth'),
                           endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'ai-powered-nutrition-analyzer-for-fitness-enthusiasts-donotdelete-pr-xvq14lyx9l2gr1'
object_key = 'Dataset-Nutrition-Tracker.zip'

streaming_body_1 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

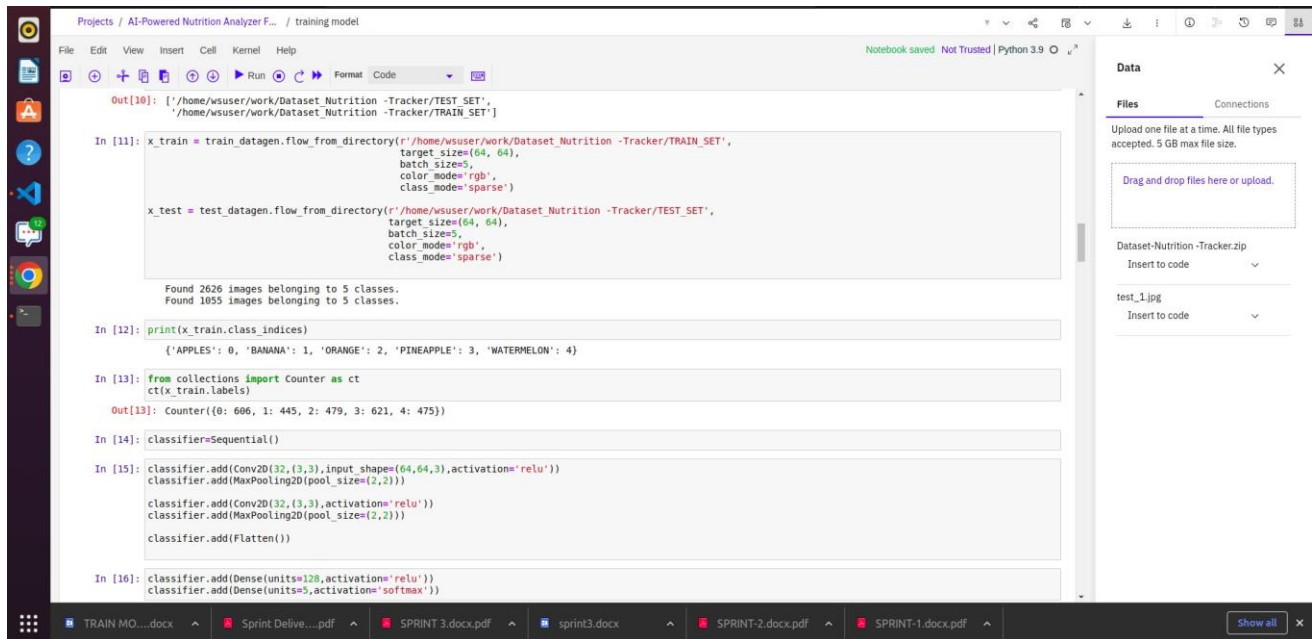
# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of boto3 and pandas to learn more about the possibilities to load the data.
# boto3 documentation: https://boto3.amazonaws.com/v1/documentation/api/latest/guide/quickstart.html#authentication
# pandas documentation: http://pandas.pydata.org/

In [8]: from io import BytesIO
import zipfile
```

The right sidebar shows the "Data" panel with a "Files" tab. It lists the following files:

- Dataset-Nutrition-Tracker.zip
- test\_1.jpg

## CREATING /TRAINING THE MODEL IN IBM CLOUD:



The screenshot shows a Jupyter Notebook interface with the following code and output:

```
Out[10]: ['/home/wuser/work/Dataset_Nutrition -Tracker/TEST_SET',
          '/home/wuser/work/Dataset_Nutrition -Tracker/TRAIN_SET']

In [11]: x_train = train_datagen.flow_from_directory(r'/home/wuser/work/Dataset_Nutrition -Tracker/TRAIN_SET',
                                                    target_size=(64, 64),
                                                    batch_size=5,
                                                    color_mode='rgb',
                                                    class_mode='sparse')

x_test = test_datagen.flow_from_directory(r'/home/wuser/work/Dataset_Nutrition -Tracker/TEST_SET',
                                          target_size=(64, 64),
                                          batch_size=5,
                                          color_mode='rgb',
                                          class_mode='sparse')

Found 2626 images belonging to 5 classes.
Found 1055 images belonging to 5 classes.

In [12]: print(x_train.class_indices)
{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}

In [13]: from collections import Counter as ct
ct(x_train.labels)

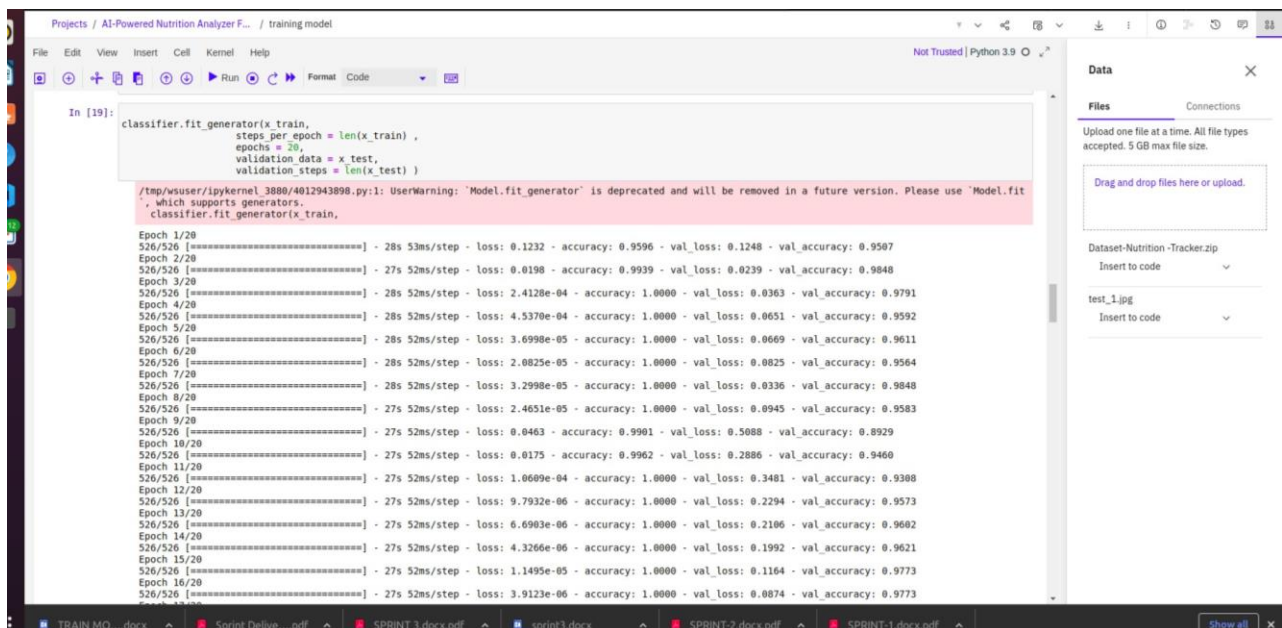
Out[13]: Counter({0: 606, 1: 445, 2: 479, 3: 621, 4: 475})

In [14]: classifier=Sequential()

In [15]: classifier.add(Conv2D(32,(3,3),input_shape=(64,64,3),activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2,2)))
classifier.add(Conv2D(32,(3,3),activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2,2)))
classifier.add(Flatten())

In [16]: classifier.add(Dense(units=120,activation='relu'))
classifier.add(Dense(units=5,activation='softmax'))
```

The right sidebar shows the 'Data' panel with a 'Files' tab. It contains a list of files: 'Dataset-Nutrition -Tracker.zip' and 'test\_1.jpg'. Below the list, there is a prompt: 'Upload one file at a time. All file types accepted. 5 GB max file size.' and a button: 'Drag and drop files here or upload.'



The screenshot shows the same Jupyter Notebook interface, but now with the training progress displayed. The code in cell [19] is:

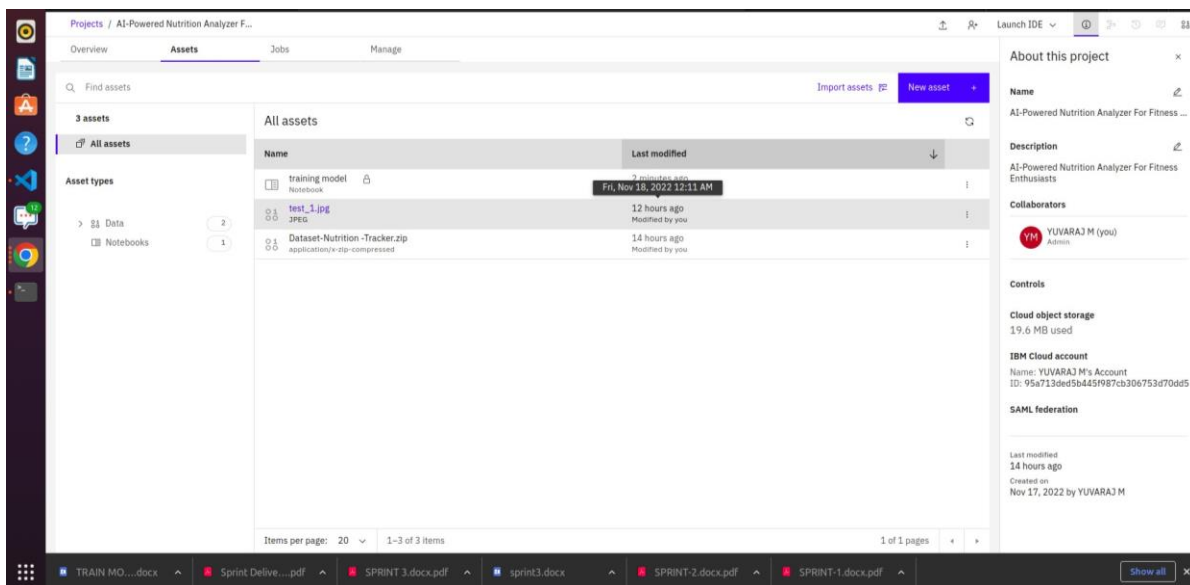
```
In [19]: classifier.fit_generator(x_train,
                                steps_per_epoch = len(x_train) ,
                                epochs = 20,
                                validation_data = x_test,
                                validation_steps = len(x_test) )
```

The output shows a warning message: '/tmp/wuser/ipykernel\_3880/4012943898.py:1: UserWarning: 'Model.fit\_generator' is deprecated and will be removed in a future version. Please use 'Model.fit', which supports generators.'

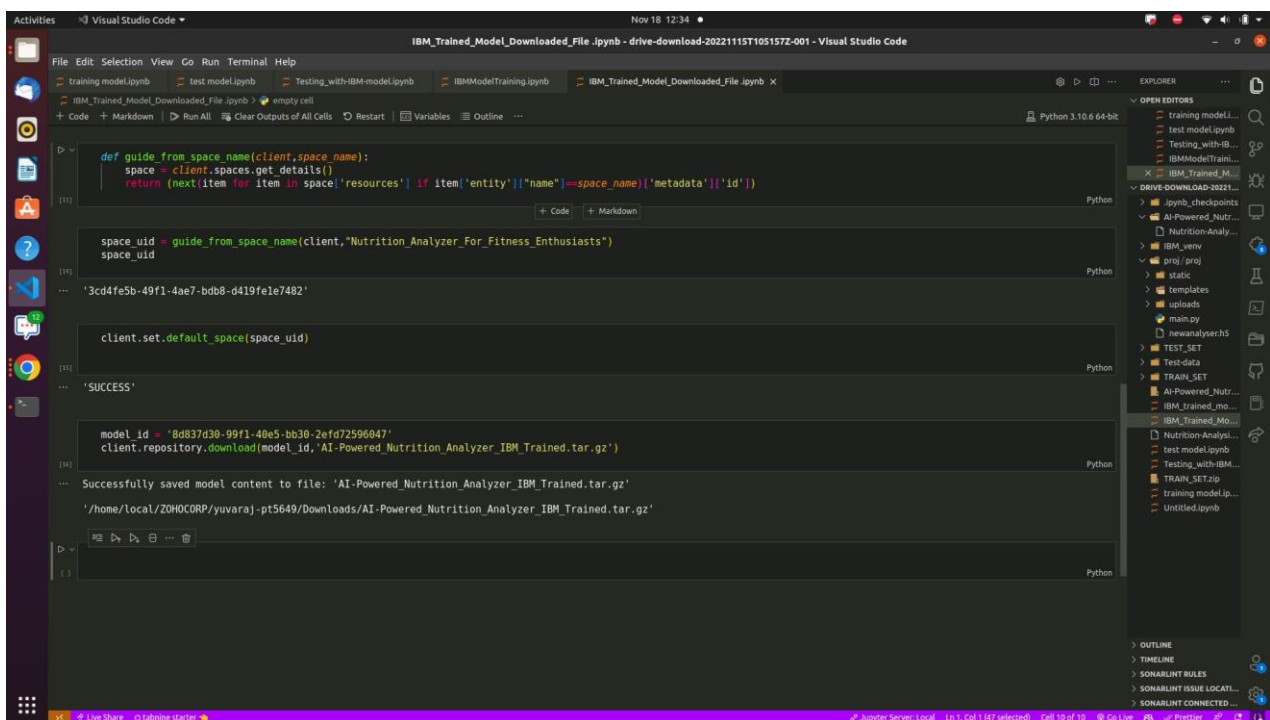
The training progress is shown as a series of lines for each epoch, including the number of steps, loss, accuracy, and validation loss/accuracy. The progress bar for each epoch is shown as a series of equals signs.

```
Epoch 1/20
526/526 [=====] - 28s 53ms/step - loss: 0.1232 - accuracy: 0.9596 - val_loss: 0.1248 - val_accuracy: 0.9507
Epoch 2/20
526/526 [=====] - 27s 52ms/step - loss: 0.0198 - accuracy: 0.9939 - val_loss: 0.0239 - val_accuracy: 0.9848
Epoch 3/20
526/526 [=====] - 28s 52ms/step - loss: 2.4128e-04 - accuracy: 1.0000 - val_loss: 0.0363 - val_accuracy: 0.9791
Epoch 4/20
526/526 [=====] - 28s 52ms/step - loss: 4.5370e-04 - accuracy: 1.0000 - val_loss: 0.0651 - val_accuracy: 0.9592
Epoch 5/20
526/526 [=====] - 28s 52ms/step - loss: 3.6998e-05 - accuracy: 1.0000 - val_loss: 0.0669 - val_accuracy: 0.9611
Epoch 6/20
526/526 [=====] - 28s 52ms/step - loss: 2.0825e-05 - accuracy: 1.0000 - val_loss: 0.0825 - val_accuracy: 0.9564
Epoch 7/20
526/526 [=====] - 28s 52ms/step - loss: 3.2998e-05 - accuracy: 1.0000 - val_loss: 0.0336 - val_accuracy: 0.9848
Epoch 8/20
526/526 [=====] - 27s 52ms/step - loss: 2.4651e-05 - accuracy: 1.0000 - val_loss: 0.0945 - val_accuracy: 0.9583
Epoch 9/20
526/526 [=====] - 27s 52ms/step - loss: 0.0463 - accuracy: 0.9901 - val_loss: 0.5088 - val_accuracy: 0.8929
Epoch 10/20
526/526 [=====] - 27s 52ms/step - loss: 0.0175 - accuracy: 0.9962 - val_loss: 0.2886 - val_accuracy: 0.9460
Epoch 11/20
526/526 [=====] - 27s 52ms/step - loss: 1.0609e-04 - accuracy: 1.0000 - val_loss: 0.3481 - val_accuracy: 0.9308
Epoch 12/20
526/526 [=====] - 27s 52ms/step - loss: 9.7932e-06 - accuracy: 1.0000 - val_loss: 0.2294 - val_accuracy: 0.9573
Epoch 13/20
526/526 [=====] - 27s 52ms/step - loss: 6.6903e-06 - accuracy: 1.0000 - val_loss: 0.2106 - val_accuracy: 0.9602
Epoch 14/20
526/526 [=====] - 27s 52ms/step - loss: 4.3266e-06 - accuracy: 1.0000 - val_loss: 0.1992 - val_accuracy: 0.9621
Epoch 15/20
526/526 [=====] - 27s 52ms/step - loss: 1.1495e-05 - accuracy: 1.0000 - val_loss: 0.1164 - val_accuracy: 0.9773
Epoch 16/20
526/526 [=====] - 27s 52ms/step - loss: 3.9123e-06 - accuracy: 1.0000 - val_loss: 0.0874 - val_accuracy: 0.9773
```

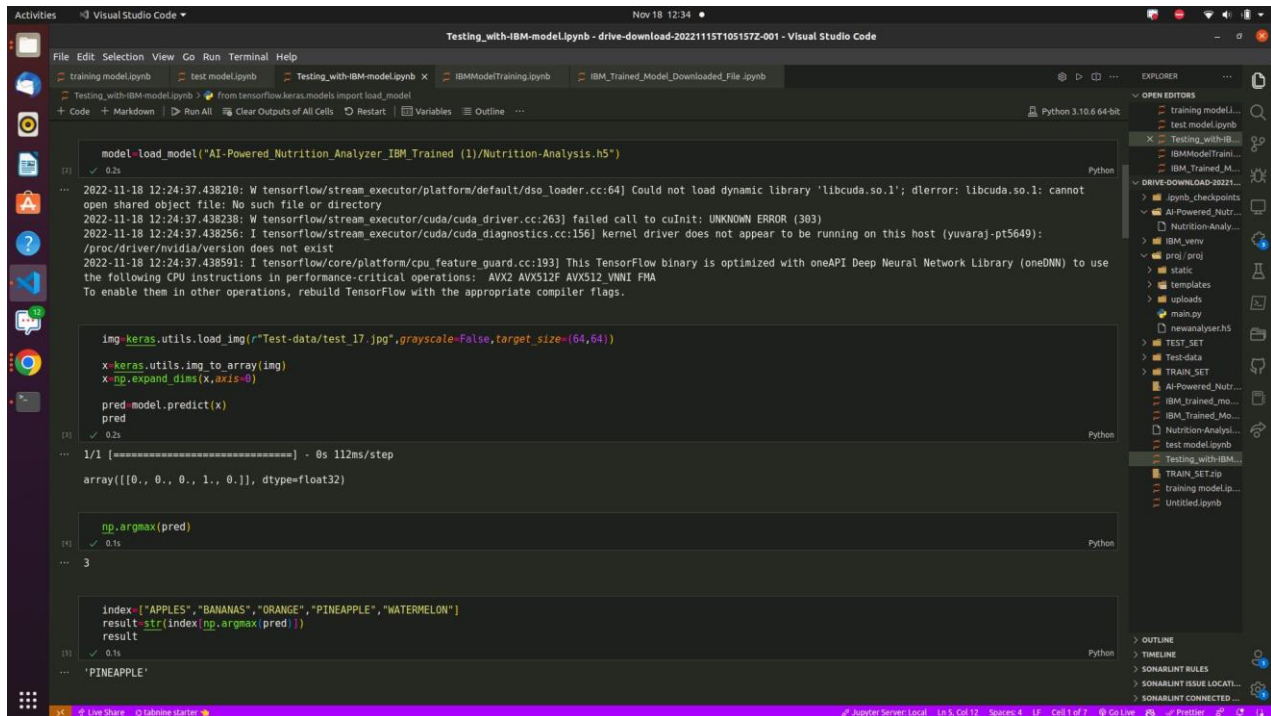
## ASSETS IN IBM CLOUD:



## DOWNLOADING THE TRAINED MODEL IN IBM CLOUD:



## TESTING THE MODEL THAT IS TRAINED IN IBM CLOUD:



```
model=load_model("AI-Powered_Nutrition_Analyzer_IBM_Trained (1)/Nutrition-Analysis.h5")

img=keras.utils.load_img("Test-data/test_17.jpg",grayscale=False,target_size=(64,64))
x=keras.utils.img_to_array(img)
x=np.expand_dims(x,axis=0)

pred=model.predict(x)
pred

1/1 [=====] - 0s 112ms/step

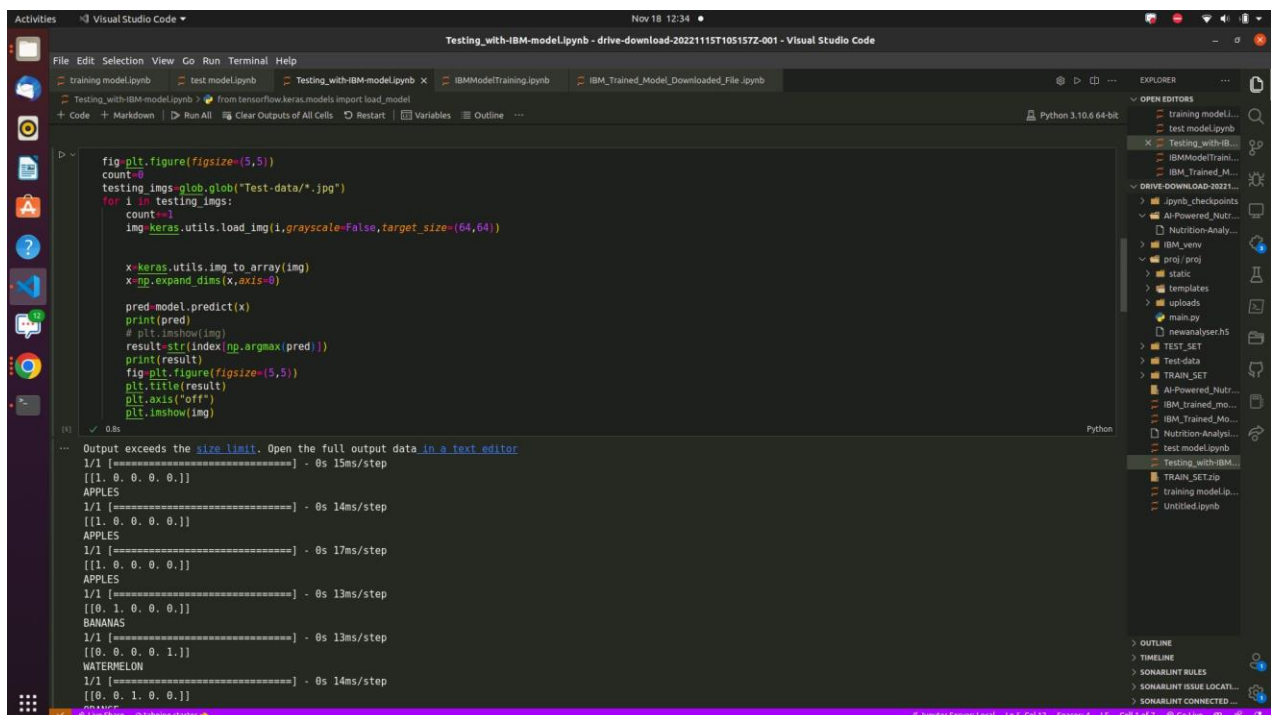
array([[0., 0., 0., 1., 0.]], dtype=float32)

np.argmax(pred)

3

index=["APPLES","BANANAS","ORANGE","PINEAPPLE","WATERMELON"]
result=str(index[np.argmax(pred)])
result

'PINEAPPLE'
```



```
fig=plt.figure(figsize=(5,5))
count=0
testing_imgs=glob("Test-data/*.jpg")
for i in testing_imgs:
    count+=1
    img=keras.utils.load_img(i,grayscale=False,target_size=(64,64))

    x=keras.utils.img_to_array(img)
    x=np.expand_dims(x,axis=0)

    pred=model.predict(x)
    print(pred)
    # plt.imshow(img)
    result=str(index[np.argmax(pred)])
    print(result)
    fig=plt.figure(figsize=(5,5))
    plt.title(result)
    plt.axis("off")
    plt.imshow(img)

Output exceeds the size limit. Open the full output data in a text editor
1/1 [=====] - 0s 15ms/step
[[1. 0. 0. 0. 0.]]
APPLES
1/1 [=====] - 0s 14ms/step
[[1. 0. 0. 0. 0.]]
APPLES
1/1 [=====] - 0s 17ms/step
[[1. 0. 0. 0. 0.]]
APPLES
1/1 [=====] - 0s 13ms/step
[[0. 1. 0. 0. 0.]]
BANANAS
1/1 [=====] - 0s 13ms/step
[[0. 0. 0. 0. 1.]]
WATERMELON
1/1 [=====] - 0s 14ms/step
[[0. 0. 1. 0. 0.]]
ORANGE
```



Visual Studio Code interface showing a Jupyter Notebook titled "Testing\_with-IBM-model.ipynb". The notebook displays three images of apples, each labeled "APPLES". The first image is a single red apple. The second image is a pile of red apples. The third image is a single red apple. The notebook is running on a Python 3.10.6 64-bit environment. The Explorer sidebar shows the file structure, including folders like "DRIVE-DOWNLOAD-20221115T105157Z-001" and "TRAIN\_SET.zip". The Output sidebar shows the execution results of the notebook cells.

Nov 18 12:34

Testing\_with-IBM-model.ipynb - drive-download-20221115T105157Z-001 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

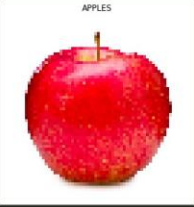
training model.ipynb test model.ipynb Testing\_with-IBM-model.ipynb x IBMModelTraining.ipynb IBM\_Trained\_Model\_Downloaded\_File.ipynb

Testing\_with-IBM-model.ipynb from tensorflow.keras.models import load\_model


+ Code + Markdown + Run All Clear Outputs of All Cells Restart Variables Outline

<Figure size 360x360 with 0 Axes>


APPLES



APPLES



APPLES



OUTLINE

TIMELINE

SONARLINT RULES

SONARLINT ISSUE LOCATIONS

SONARLINT CONNECTED...

Python 3.10.6 64-bit

DRIVE-DOWNLOAD-20221115T105157Z-001

IBMModelTraining.ipynb

IBM\_Trained\_Model\_Downloaded\_File.ipynb

training model.ipynb

test model.ipynb

TRAIN\_SET.zip

training model.ipynb

Untitled.ipynb

Live Share

Tabnine starter

Python Server: Local

Ln 5, Col 12

Spaces: 4

LF

Cell 1 of 7

Go Live