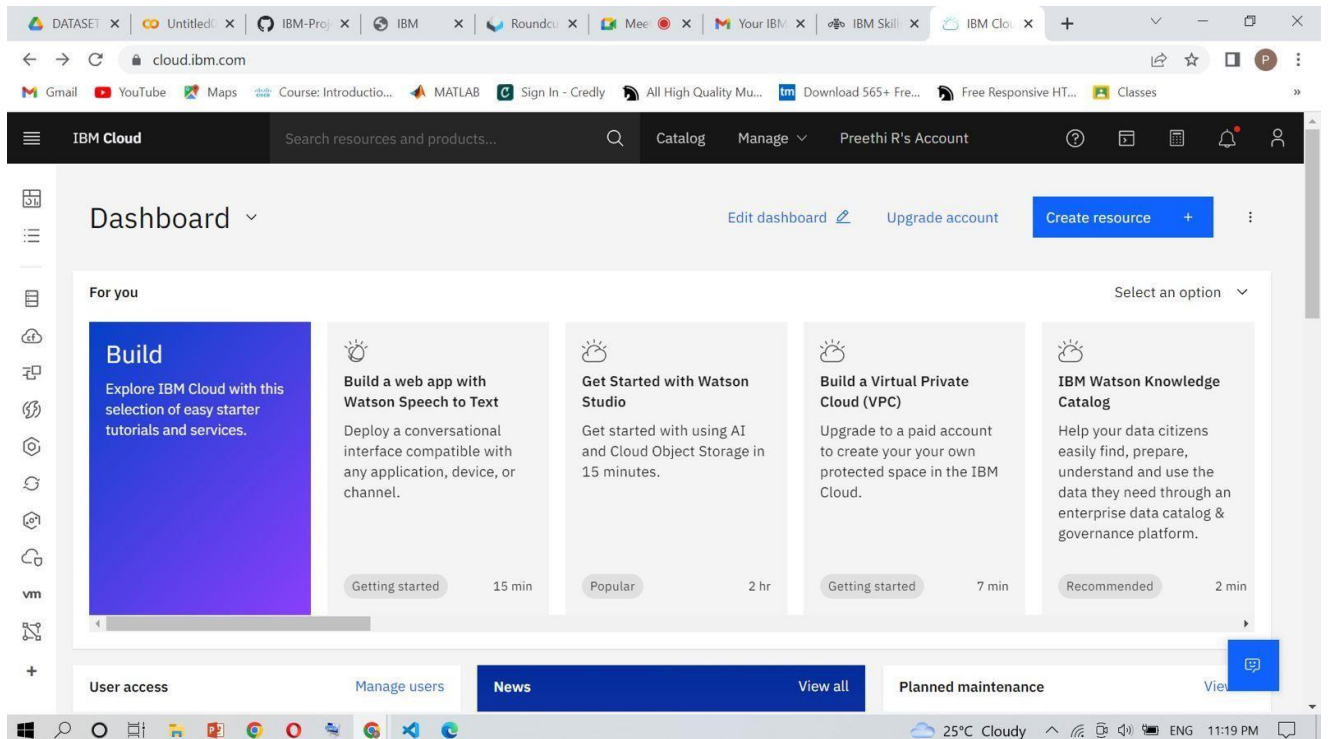


Sprint 4

REGISTER FOR CLOUD ACCOUNT

DATE	22 November 2022
TEAM ID	PNT2022TMID14614
PROJECT NAME	AI-powered Nutrition Analyzer for Fitness Enthusiasts



The screenshot displays the IBM Cloud Dashboard interface. At the top, a navigation bar includes the IBM Cloud logo, a search bar, and links for Catalog, Manage, and the user's account (Preethi R's Account). The main content area is titled "Dashboard" and features a "For you" section with five recommended actions: "Build" (a large blue box), "Build a web app with Watson Speech to Text" (15 min), "Get Started with Watson Studio" (2 hr), "Build a Virtual Private Cloud (VPC)" (7 min), and "IBM Watson Knowledge Catalog" (2 min). Each action includes a brief description and a "Getting started" button. The bottom of the dashboard shows sections for "User access", "News", and "Planned maintenance". The browser's address bar shows "cloud.ibm.com", and the Windows taskbar at the bottom indicates the system time as 11:19 PM on 25°C Cloudy.

cloud.ibm.com/user

IBM Cloud

Search resources and products...

Profile

Active sessions

Login settings

Notification preferences

Contact information

Edit

User ID

pree19ec106@rmkcet.ac.in

Password

Email

pree19ec106@rmkcet.ac.in

Role

Click **Edit** to enter your role.

Name

Preethi R

Industry

Click **Edit** to enter your industry.

Language

English

Upload a photo

25°C Cloudy

ENG 11:19 PM

https://cloud.ibm.com

IBM Cloud

Search resources and products...

Dashboard

Edit dashboard

Upgrade account

Create resource

For you

Build

Explore IBM Cloud with this selection of easy starter tutorials and services.

Build a web app with Watson Speech to Text

Deploy a conversational interface compatible with any application, device, or channel.

Get started with Watson Studio

Get started with using AI and Cloud Object Storage in 15 minutes.

IBM Watson Knowledge Catalog

Help your data citizens easily find, prepare, understand and use the data they need through an enterprise data catalog & governance platform.

Build a Virtual Private Cloud (VPC)

Upgrade to a paid account to create your own protected space in the IBM Cloud.

App Connect

Instantly connect applications, data, heritage systems and modern technologies, even without a single line of code, with IBM App Connect.

Build a virtual machine

Lift and shift your VMware workloads to the IBM Cloud.

Get started with Discovery

Get up to speed with Discovery with so many tutorials, deep-dive and complete example working code.

News

Introducing Badges to IBM Cloud Certification

Announcing IBM SaaSOne Network Performance Management Version 6.4

IBM Named a Leader in Gartner Magic Quadrant for Full Life Cycle API Management

IBM to Introduce a New Incident Management SaaS Offering

Recent support cases

Planned maintenance

IBM Cloud status

No issues

Usage

User access

Manage users

10:50 PM 22-11-2022

IBM Cloud

Profile

Active sessions

Login settings

Notification preferences

Search resources and products...

Catalog

Manage

PREETHI S's Account

Profile

Contact information

Upload a photo

User ID

pree19ec107@rmkce.ac.in

Password

Email

pree19ec107@rmkce.ac.in

Role

Click Edit to enter your role.

Name

PREETHI S

Industry

Click Edit to enter your industry.

Language

English

Edit

Type here to search

69%

Raining now

10:52 PM

22-11-2022

TRAIN MODEL ON IBM

The screenshot displays the IBM Watson Studio web interface. The browser's address bar shows the URL: `dataplatform.cloud.ibm.com/analytics/notebooks/v2/9ae3b406-22fc-40b2-a6fd-9070a0ed4932?projectId=d6dcbc93-9f6d-4ea7-9096-05fa308bab26&con...`. The interface includes a top navigation bar with the IBM Watson Studio logo, a search bar, and user account information (Preethi R's Account, Dallas). Below this is a breadcrumb trail: `Projects / Model Building / Model_Building`. The notebook editor shows a Jupyter-style interface with a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for adding cells, saving, and running. The notebook content is divided into two sections: **Load The Dataset** and **Image Preprocessing**. The **Load The Dataset** section contains three code cells:
1. `In [1]: from google.colab import drive
drive.mount('/content/drive')` with the output `Mounted at /content/drive`.
2. `In [2]: Data_trainpath='/content/drive/MyDrive/Dataset/TRAIN_SET'`
3. `In [3]: Data_testpath='/content/drive/MyDrive/Dataset/TEST_SET'`
The **Image Preprocessing** section contains one code cell:
`In [4]: from keras.preprocessing.image import ImageDataGenerator`
The bottom of the image shows a Windows taskbar with various application icons and a system tray indicating `27°C Cloudy` and the time `11:55 PM`.

IBM Watson Studio

Search in your workspaces

Buy

Preethi R's Account

Dallas

PR

Projects / Model Building / Model_Building

File Edit View Insert Cell Kernel Help

Not Trusted | Python 3.9

Format Markdown

Load The Dataset

```
In [1]: from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive

In [2]: Data_trainpath='/content/drive/MyDrive/Dataset/TRAIN_SET'

In [3]: Data_testpath='/content/drive/MyDrive/Dataset/TEST_SET'
```

Image Preprocessing

Import The ImageDataGenerator Library

```
In [4]: from keras.preprocessing.image import ImageDataGenerator
```

27°C Cloudy 11:55 PM

This screenshot shows a Jupyter notebook in IBM Watson Studio. The browser tabs at the top include IBM, Meet, New Tab, IBM-Project-2425, Service Details, and Model_Building. The notebook's URL is `dataplatfrom.cloud.ibm.com/analytics/notebooks/v2/9ae3b406-22fc-40b2-a6fd-9070a0ed4932?projectid=d6dcbc93-9f6d-4ea7-9096-05fa308bab26&con...`. The interface includes a top bar with 'IBM Watson Studio', a search bar, and user account information. The notebook's breadcrumb is 'Projects / Model Building / Model_Building'. The toolbar shows 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', and 'Help' menus, along with a 'Run' button and a 'Format' dropdown set to 'Markdown'. The code in the notebook is as follows:

```
In [6]: test_datagen=ImageDataGenerator(rescale=1./255)

Apply Image DataGenerator Functionality To Trainset And Testset

In [9]: x_train = train_datagen.flow_from_directory(Data_trainpath,target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse')
Found 3492 images belonging to 5 classes.

In [10]: x_test = train_datagen.flow_from_directory(Data_testpath,target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse')
Found 976 images belonging to 5 classes.

In [11]: print(x_train.class_indices)
{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}

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

In [13]: from collections import Counter as c
```

The bottom status bar shows '27°C Cloudy', system icons, and the time '11:55 PM'.

This screenshot shows a Jupyter notebook in IBM Watson Studio, continuing the previous notebook. The browser tabs and interface elements are identical. The notebook's breadcrumb is 'Projects / Model Building / Model_Building'. The code in the notebook is as follows:

```
In [4]: from keras.preprocessing.image import ImageDataGenerator

Configure ImageDataGenerator Class

In [5]: train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)

In [6]: test_datagen=ImageDataGenerator(rescale=1./255)

Apply Image DataGenerator Functionality To Trainset And Testset

In [9]: x_train = train_datagen.flow_from_directory(Data_trainpath,target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse')
Found 3492 images belonging to 5 classes.
```

The bottom status bar shows '27°C Cloudy', system icons, and the time '11:55 PM'.

This screenshot shows a Jupyter notebook in IBM Watson Studio. The browser tabs at the top include IBM, Meet, New Tab, IBM-Project-2425, Service Details, and Model_Building. The notebook's address bar shows a URL from dataplatform.cloud.ibm.com. The interface includes a top navigation bar with 'IBM Watson Studio', a search bar, and user account information for 'Preethi R's Account' in 'Dallas'. The notebook title is 'Model_Building'. The toolbar shows 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', and 'Help' menus, along with a 'Not Trusted' warning and 'Python 3.9' version. The code area contains the following:

```
In [6]: test_datagen=ImageDataGenerator(rescale=1./255)
```

Apply Image DataGenerator Functionality To Trainset And Testset

```
In [9]: x_train = train_datagen.flow_from_directory(Data_trainpath,target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse')
Found 3492 images belonging to 5 classes.
```

```
In [10]: x_test = train_datagen.flow_from_directory(Data_testpath,target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse')
Found 976 images belonging to 5 classes.
```

```
In [11]: print(x_train.class_indices)
{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}
```

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

```
In [13]: from collections import Counter as c
```

The bottom status bar shows '27°C Cloudy', system icons, and the time '11:55 PM'.

This screenshot shows the same Jupyter notebook in IBM Watson Studio, continuing the model building process. The browser tabs and interface elements are identical to the previous screenshot. The code area contains the following:

```
In [15]: model=Sequential()
```

Adding CNN Layers

First Convolution Layer and pooling

```
In [16]: model.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

Second Convolution Layer and pooling

```
In [17]: model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

Flatten layer

```
In [18]: model.add(Flatten())
```

The bottom status bar shows '27°C Cloudy', system icons, and the time '11:55 PM'.

IBM Watson Studio interface showing a Jupyter Notebook titled "Model_Building". The notebook is running Python 3.9. The code in the notebook is:

```
In [19]: model.add(Dense(units=128, activation='relu'))
model.add(Dense(units=5, activation='softmax'))

In [20]: model.summary()
```

The output of the summary is:

```
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
```

IBM Watson Studio interface showing the same Jupyter Notebook. The code in the notebook is:

```
Trainable params: 813,733
Non-trainable params: 0

Configure The Learning Process

In [21]: model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

Train The Model

In [22]: model.fit_generator(
    generator=x_train, steps_per_epoch = len(x_train),
    epochs=15, validation_data=x_test, validation_steps = len(x_test))
```

The output of the fit_generator is:

```
Epoch 1/15
699/699 [=====] - 933s 1s/step - loss: 0.7478 - accuracy: 0.7228 - val_loss: 0.6479 - val_accuracy: 0.7643
Epoch 2/15
699/699 [=====] - 43s 61ms/step - loss: 0.5850 - accuracy: 0.7941 - val_loss: 0.6027 - val_accuracy: 0.7961
```

IBM Watson Studio interface showing the Model_Buidling project. The output displays training progress for 15 epochs, including loss, accuracy, and validation metrics.

```
Epoch 2/15
699/699 [=====] - 43s 61ms/step - loss: 0.5850 - accuracy: 0.7941 - val_loss: 0.6027 - val_accuracy: 0.7961
Epoch 3/15
699/699 [=====] - 41s 58ms/step - loss: 0.5283 - accuracy: 0.8064 - val_loss: 0.6776 - val_accuracy: 0.7664
Epoch 4/15
699/699 [=====] - 39s 56ms/step - loss: 0.4930 - accuracy: 0.8230 - val_loss: 0.5407 - val_accuracy: 0.8043
Epoch 5/15
699/699 [=====] - 41s 58ms/step - loss: 0.4620 - accuracy: 0.8259 - val_loss: 0.5942 - val_accuracy: 0.7736
Epoch 6/15
699/699 [=====] - 41s 58ms/step - loss: 0.4349 - accuracy: 0.8408 - val_loss: 0.6177 - val_accuracy: 0.7715
Epoch 7/15
699/699 [=====] - 39s 55ms/step - loss: 0.4055 - accuracy: 0.8462 - val_loss: 0.5708 - val_accuracy: 0.7971
Epoch 8/15
699/699 [=====] - 43s 61ms/step - loss: 0.3919 - accuracy: 0.8571 - val_loss: 0.4557 - val_accuracy: 0.8504
Epoch 9/15
699/699 [=====] - 44s 63ms/step - loss: 0.3472 - accuracy: 0.8697 - val_loss: 0.5714 - val_accuracy: 0.8309
Epoch 10/15
699/699 [=====] - 44s 63ms/step - loss: 0.3513 - accuracy: 0.8766 - val_loss: 0.5235 - val_accuracy: 0.8391
Epoch 11/15
699/699 [=====] - 43s 62ms/step - loss: 0.2960 - accuracy: 0.8935 - val_loss: 0.4929 - val_accuracy: 0.8494
Epoch 12/15
699/699 [=====] - 46s 65ms/step - loss: 0.2846 - accuracy: 0.8943 - val_loss: 0.5114 - val_accuracy: 0.8258
Epoch 13/15
699/699 [=====] - 39s 56ms/step - loss: 0.2540 - accuracy: 0.9006 - val_loss: 0.5052 - val_accuracy: 0.8494
```

IBM Watson Studio interface showing the Model_Buidling project. The code cells demonstrate saving the model and testing it on a new image.

```
In [23]: model.save('nutrition.h5')
```

Save The Model

```
In [24]: from tensorflow.keras.models import load_model
from keras.preprocessing import image
final_model = load_model("nutrition.h5")
```

Test The Model

```
In [25]: from tensorflow.keras.utils import img_to_array
```

```
In [26]: img = tensorflow.keras.utils.load_img("/content/drive/MyDrive/Nutrition Image Analysis using CNN and Rapid API/Nutrition Analysis Using Image C
x = img_to_array(img)
x = np.expand_dims(x,axis = 0)
pred = np.argmax(final_model.predict(x),axis=1)
pred
```

1/1 [=====] - 0s 101ms/step

Out[26]: array([21])

IBM Watson Studio interface showing a Jupyter Notebook session. The notebook is titled "Model_Building" and is running Python 3.9. The code in the notebook is as follows:


```
In [26]: img = tensorflow.keras.utils.load_img("/content/drive/MyDrive/Nutrition Image Analysis using CNN and Rapid API/Nutrition Analysis Using Image C
x = img_to_array(img)
x = np.expand_dims(x,axis = 0)
pred =np.argmax(final_model.predict(x),axis=1)
pred

1/1 [=====] - 0s 101ms/step

Out[26]: array([2])

In [27]: index=['APPLES', 'BANANA', 'ORANGE', 'PINEAPPLE', 'WATERMELON']
result=index[pred[0]]
result

Out[27]: 'ORANGE'

In [28]: img
Out[28]: 
```

The notebook output shows the prediction of the fruit class as 'ORANGE' based on the image input.

IBM Watson Studio interface showing a Jupyter Notebook session. The notebook is titled "Model_Building" and is running Python 3.9. The code in the notebook is as follows:


```
In [29]: img = tensorflow.keras.utils.load_img("/content/drive/MyDrive/Nutrition Image Analysis using CNN and Rapid API/Nutrition Analysis Using Image C
x = img_to_array(img)
x = np.expand_dims(x,axis = 0)
pred =np.argmax(final_model.predict(x),axis=1)
pred

1/1 [=====] - 0s 18ms/step

Out[29]: array([0])

In [30]: result=index[pred[0]]
result

Out[30]: 'APPLES'

In [31]: img
Out[31]: 
```

The notebook output shows the prediction of the fruit class as 'APPLES' based on the image input.