Model Building

```
pwd
```

```
'/home/wsuser/work'
```

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_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.
client_0e2c22ba144f4bc4b380cb4adffcdf31 =
ibm_boto3.client(service_name='s3',
    ibm_api_key_id='LS5quLWbdlmaYoNQwBpNPZtxDudV2qI9AjEw1Q_kh53k',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-
storage.appdomain.cloud')

streaming_body_1 =
client_0e2c22ba144f4bc4b380cb4adffcdf31.get_object(Bucket='gesturebase
dtoolforsterilebrowsin-donotdelete-pr-bypejx1cp3avvd',
Key='Dataset.zip')['Body']

# Your data file was loaded into a botocore.response.StreamingBody
object.
# Please read the documentation of ibm_boto3 and pandas to learn more
about the possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/

from io import BytesIO
import zipfile
unzip=zipfile.ZipFile(BytesIO(streaming_body_1.read()),'r')
file_paths=unzip.namelist()
for path in file_paths:
    unzip.extract(path)

# This library helps add support for large, multi-dimensional arrays
and matrices
import numpy as np
```

```
#open source used for both ML and DL for computation
import tensorflow as tf
#it is a plain stack of layers
from tensorflow.keras.models import Sequential
#Dense layer is the regular deeply connected neural network layer
from tensorflow.keras.layers import Dense,Flatten, Dropout
#Faltten-used fot flattening the input or change the dimension,
MaxPooling2D-for downsampling the image for Convolutional layer
from tensorflow.keras.layers import Convolution2D,MaxPooling2D
#Its used for different augmentation of the image
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

Importing libraries

Augmenting the data

```
#setting parameter for Image Data agumentation to the traing data
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
#Image Data agumentation to the testing data
test_datagen=ImageDataGenerator(rescale=1./255)
```

Loading our data and performing data agumentation

```
#performing data agumentation to train data
x_train =
train_datagen.flow_from_directory(r'/home/wsuser/work/Dataset/train',
                                            target_size=(64, 64),
                                            batch_size=3,
                                            color_mode='grayscale',
                                            class_mode='categorical')
#performing data agumentation to test data
x_test =
test_datagen.flow_from_directory(r'/home/wsuser/work/Dataset/test',
                                           target_size=(64, 64),
                                           batch_size=3,
                                           color_mode='grayscale',
                                           class_mode='categorical')
```

```
Found 594 images belonging to 6 classes.
Found 30 images belonging to 6 classes.
```

```
print(x_train.class_indices)#checking the number of classes
```

```
{'0': 0, '1': 1, '2': 2, '3': 3, '4': 4, '5': 5}
```

Model Creation

```
# Initializing the CNN
model = Sequential()
```

```
# First convolution layer and pooling
model.add(Convolution2D(32, (3, 3), input_shape=(64, 64, 1),
activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

# Second convolution layer and pooling
model.add(Convolution2D(32, (3, 3), activation='relu'))
# input_shape is going to be the pooled feature maps from the previous
convolution layer
model.add(MaxPooling2D(pool_size=(2,2)))

# Flattening the layers i.e. input layer
model.add(Flatten())

# Adding a fully connected layer, i.e. Hidden Layer
model.add(Dense(units=512 , activation='relu'))

# softmax for categorical analysis, Output Layer
model.add(Dense(units=6, activation='softmax'))

model.summary()#summary of our model
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        320

 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, 512)               3211776

 dense_1 (Dense)             (None, 6)                 3078

=================================================================
Total params: 3,224,422
Trainable params: 3,224,422
Non-trainable params: 0
_________________________________________________________________
```

Model Compilation

```
# Compiling the CNN
# categorical_crossentropy for more than 2
```

```
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
```

Model fitting

```
# It will generate packets of train and test data for training
model.fit_generator(x_train,
                    steps_per_epoch = 594/3 ,
                    epochs = 25,
                    validation_data = x_test,
                    validation_steps = 30/3 )
```

```
/tmp/wsuser/ipykernel_156/804983804.py:2: UserWarning:
`Model.fit_generator` is deprecated and will be removed in a future
version. Please use `Model.fit`, which supports generators.
  model.fit_generator(x_train,

Epoch 1/25
198/198 [==============================] - 10s 49ms/step - loss:
1.2167 - accuracy: 0.5084 - val_loss: 0.6183 - val_accuracy: 0.8000
Epoch 2/25
198/198 [==============================] - 10s 49ms/step - loss:
0.5515 - accuracy: 0.7845 - val_loss: 0.5207 - val_accuracy: 0.8333
Epoch 3/25
198/198 [==============================] - 9s 46ms/step - loss: 0.3871
- accuracy: 0.8502 - val_loss: 0.3111 - val_accuracy: 0.8667
Epoch 4/25
198/198 [==============================] - 9s 45ms/step - loss: 0.2965
- accuracy: 0.8923 - val_loss: 0.2008 - val_accuracy: 0.9333
Epoch 5/25
198/198 [==============================] - 9s 46ms/step - loss: 0.2004
- accuracy: 0.9310 - val_loss: 0.1084 - val_accuracy: 0.9667
Epoch 6/25
198/198 [==============================] - 9s 46ms/step - loss: 0.1727
- accuracy: 0.9360 - val_loss: 0.1391 - val_accuracy: 0.9333
Epoch 7/25
198/198 [==============================] - 9s 46ms/step - loss: 0.1371
- accuracy: 0.9444 - val_loss: 0.2336 - val_accuracy: 0.9333
Epoch 8/25
198/198 [==============================] - 9s 48ms/step - loss: 0.0740
- accuracy: 0.9731 - val_loss: 0.2065 - val_accuracy: 0.9333
Epoch 9/25
198/198 [==============================] - 9s 47ms/step - loss: 0.0824
- accuracy: 0.9764 - val_loss: 0.1247 - val_accuracy: 0.9000
Epoch 10/25
198/198 [==============================] - 9s 46ms/step - loss: 0.0945
- accuracy: 0.9747 - val_loss: 0.2387 - val_accuracy: 0.9333
Epoch 11/25
198/198 [==============================] - 9s 44ms/step - loss: 0.0687
- accuracy: 0.9798 - val_loss: 0.2370 - val_accuracy: 0.9333
Epoch 12/25
```

```
198/198 [==============================] - 9s 46ms/step - loss: 0.0745
- accuracy: 0.9747 - val_loss: 0.1488 - val_accuracy: 0.9667
Epoch 13/25
198/198 [==============================] - 9s 44ms/step - loss: 0.0487
- accuracy: 0.9848 - val_loss: 0.2267 - val_accuracy: 0.9333
Epoch 14/25
198/198 [==============================] - 9s 48ms/step - loss: 0.0758
- accuracy: 0.9731 - val_loss: 0.6144 - val_accuracy: 0.8667
Epoch 15/25
198/198 [==============================] - 9s 48ms/step - loss: 0.0654
- accuracy: 0.9815 - val_loss: 0.1128 - val_accuracy: 0.9667
Epoch 16/25
198/198 [==============================] - 9s 47ms/step - loss: 0.0211
- accuracy: 0.9933 - val_loss: 0.2591 - val_accuracy: 0.9667
Epoch 17/25
198/198 [==============================] - 9s 45ms/step - loss: 0.0360
- accuracy: 0.9882 - val_loss: 0.1846 - val_accuracy: 0.9667
Epoch 18/25
198/198 [==============================] - 9s 46ms/step - loss: 0.0493
- accuracy: 0.9764 - val_loss: 0.2447 - val_accuracy: 0.9333
Epoch 19/25
198/198 [==============================] - 9s 46ms/step - loss: 0.0422
- accuracy: 0.9899 - val_loss: 0.1664 - val_accuracy: 0.9667
Epoch 20/25
198/198 [==============================] - 9s 47ms/step - loss: 0.0463
- accuracy: 0.9798 - val_loss: 0.0799 - val_accuracy: 0.9667
Epoch 21/25
198/198 [==============================] - 10s 49ms/step - loss:
0.0104 - accuracy: 0.9966 - val_loss: 0.1213 - val_accuracy: 0.9667
Epoch 22/25
198/198 [==============================] - 10s 49ms/step - loss:
0.0687 - accuracy: 0.9781 - val_loss: 0.1620 - val_accuracy: 0.9667
Epoch 23/25
198/198 [==============================] - 10s 48ms/step - loss:
0.0261 - accuracy: 0.9916 - val_loss: 0.2275 - val_accuracy: 0.9667
Epoch 24/25
198/198 [==============================] - 10s 48ms/step - loss:
0.0570 - accuracy: 0.9882 - val_loss: 0.1140 - val_accuracy: 0.9667
Epoch 25/25
198/198 [==============================] - 9s 47ms/step - loss: 0.0176
- accuracy: 0.9966 - val_loss: 0.3862 - val_accuracy: 0.9667

<keras.callbacks.History at 0x7f6610660190>
```

Saving model

```
# Save the model
model.save('gesture.h5')
```

```
!tar -zcvf Gesture-based-Radiology-Images.tgz gesture.h5

gesture.h5
```

```
ls
```

```
Dataset/  Gesture-based-Radiology-Images.tgz  gesture.h5
```

```
model_json = model.to_json()
with open("model-bw.json", "w") as json_file:
    json_file.write(model_json)
```

IBM Deployment

```
!pip install watson-machine-learning-client
```

```
Collecting watson-machine-learning-client
  Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl
(538 kB)
ent already satisfied: certifi in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (2021.10.8)
Requirement already satisfied: requests in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(2.26.0)
Requirement already satisfied: lomond in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (0.3.3)
Requirement already satisfied: pandas in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (1.3.4)
Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(2.11.0)
Requirement already satisfied: urllib3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (1.26.7)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(0.8.9)
Requirement already satisfied: boto3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (1.18.21)
Requirement already satisfied: tqdm in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (4.62.3)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3-
>watson-machine-learning-client) (0.10.0)
Requirement already satisfied: botocore<1.22.0,>=1.21.21 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3-
>watson-machine-learning-client) (1.21.41)
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3-
```

```
>watson-machine-learning-client) (0.5.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client)
(2.8.2)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>=2.1-
>botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client)
(1.15.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-
sdk->watson-machine-learning-client) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-
sdk->watson-machine-learning-client) (2.11.0)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from requests->watson-machine-
learning-client) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests-
>watson-machine-learning-client) (2.0.4)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-
client) (2021.3)
Requirement already satisfied: numpy>=1.17.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas-
>watson-machine-learning-client) (1.20.3)
Installing collected packages: watson-machine-learning-client
Successfully installed watson-machine-learning-client-1.0.391
```

```
from ibm_watson_machine_learning import APIClient
wml_credentials={
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"nYpdCnUh2oai8TfRIynB7V0uV4Hupcmt5FxAWxPTg1zb"
}

client=APIClient(wml_credentials)

client=APIClient(wml_credentials)

client
```

```
<ibm_watson_machine_learning.client.APIClient at 0x7f6610660af0>
```

```
def guid_space_name(client,Gesture):
    space=client.spaces.get_details()
    return(next(item for item in space['resources'] if item['entity']
['name']==Gesture)['metadata']['id'])

space_uid=guid_space_name(client,'Gesture')
print("Space UID "+space_uid)
```

```
Space UID 549dc1c8-1f0b-43f7-a6d9-2956539b64a6

client.set.default_space(space_uid)

'SUCCESS'

client.software_specifications.list(100)

------------------------------  ------------------------------------
----
NAME                            ASSET_ID
TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9
base
kernel-spark3.2-scala2.12       020d69ce-7ac1-5e68-ac1a-31189867356a
base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288
base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687
base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee
base
pytorch-onnx_rt22.1-py3.9       0b848dd4-e681-5599-be41-b5f6fccc6471
base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda
base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306
base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22
base
pytorch_1.1-py3.6               10ac12d6-6b30-4ccd-8392-3e922c096a92
base
tensorflow_1.15-py3.6-ddl       111e41b3-de2d-5422-a4d6-bf776828c4b7
base
runtime-22.1-py3.9              12b83a17-24d8-5082-900f-0ab31fbfd3cb
base
scikit-learn_0.22-py3.6         154010fa-5b3b-4ac1-82af-4d5ee5abbc85
base
default_r3.6                    1b70aec3-ab34-4b87-8aa0-a4a3c8296a36
base
pytorch-onnx_1.3-py3.6          1bc6029a-cc97-56da-b8e0-39c3880dbbe7
base
pytorch-onnx_rt22.1-py3.9-edt   1d362186-7ad5-5b59-8b6c-9d0880bde37f
base
tensorflow_2.1-py3.6            1eb25b84-d6ed-5dde-b6a5-3fbdf1665666
base
spark-mllib_3.2                 20047f72-0a98-58c7-9ff5-a77b012eb8f5
base
tensorflow_2.4-py3.8-horovod    217c16f6-178f-56bf-824a-b19f20564c49
base
runtime-22.1-py3.9-cuda         26215f05-08c3-5a41-a1b0-da66306ce658
```

base
do_py3.8 295addb5-9ef9-547e-9bf4-92ae3563e720
base
autoai-ts_3.8-py3.8 2aa0c932-798f-5ae9-abd6-15e0c2402fb5
base
tensorflow_1.15-py3.6 2b73a275-7cbf-420b-a912-eae7f436e0bc
base
pytorch_1.2-py3.6 2c8ef57d-2687-4b7d-acce-01f94976dac1
base
spark-mllib_2.3 2e51f700-bca0-4b0d-88dc-5c6791338875
base
pytorch-onnx_1.1-py3.6-edt 32983cea-3f32-4400-8965-dde874a8d67e
base
spark-mllib_3.0-py37 36507ebe-8770-55ba-ab2a-eafe787600e9
base
spark-mllib_2.4 390d21f8-e58b-4fac-9c55-d7ceda621326
base
xgboost_0.82-py3.6 39e31acd-5f30-41dc-ae44-60233c80306e
base
pytorch-onnx_1.2-py3.6-edt 40589d0e-7019-4e28-8daa-fb03b6f4fe12
base
default_r36py38 41c247d3-45f8-5a71-b065-8580229facf0
base
autoai-ts_rt22.1-py3.9 4269d26e-07ba-5d40-8f66-2d495b0c71f7
base
autoai-obm_3.0 42b92e18-d9ab-567f-988a-4240ba1ed5f7
base
pmml-3.0_4.3 493bcb95-16f1-5bc5-bee8-81b8af80e9c7
base
spark-mllib_2.4-r_3.6 49403dff-92e9-4c87-a3d7-a42d0021c095
base
xgboost_0.90-py3.6 4ff8d6c2-1343-4c18-85e1-689c965304d3
base
pytorch-onnx_1.1-py3.6 50f95b2a-bc16-43bb-bc94-b0bed208c60b
base
autoai-ts_3.9-py3.8 52c57136-80fa-572e-8728-a5e7cbb42cde
base
spark-mllib_2.4-scala_2.11 55a70f99-7320-4be5-9fb9-9edb5a443af5
base
spark-mllib_3.0 5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9
base
autoai-obm_2.0 5c2e37fa-80b8-5e77-840f-d912469614ee
base
spss-modeler_18.1 5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b
base
cuda-py3.8 5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e
base
autoai-kb_3.1-py3.7 632d4b22-10aa-5180-88f0-f52dfb6444d7
base
pytorch-onnx_1.7-py3.8 634d3cdc-b562-5bf9-a2d4-ea90a478456b

```
base
spark-mllib_2.3-r_3.6            6586b9e3-ccd6-4f92-900f-0f8cb2bd6f0c
base
tensorflow_2.4-py3.7             65e171d7-72d1-55d9-8ebb-f813d620c9bb
base
spss-modeler_18.2                687eddc9-028a-4117-b9dd-e57b36f1efa5
base
pytorch-onnx_1.2-py3.6           692a6a4d-2c4d-45ff-a1ed-b167ee55469a
base
spark-mllib_2.3-scala_2.11       7963efe5-bbec-417e-92cf-0574e21b4e8d
base
spark-mllib_2.4-py37             7abc992b-b685-532b-a122-a396a3cdbaab
base
caffe_1.0-py3.6                  7bb3dbe2-da6e-4145-918d-b6d84aa93b6b
base
pytorch-onnx_1.7-py3.7           812c6631-42b7-5613-982b-02098e6c909c
base
cuda-py3.6                       82c79ece-4d12-40e6-8787-a7b9e0f62770
base
tensorflow_1.15-py3.6-horovod    8964680e-d5e4-5bb8-919b-8342c6c0dfd8
base
hybrid_0.1                       8c1a58c6-62b5-4dc4-987a-df751c2756b6
base
pytorch-onnx_1.3-py3.7           8d5d8a87-a912-54cf-81ec-3914adaa988d
base
caffe-ibm_1.0-py3.6              8d863266-7927-4d1e-97d7-56a7f4c0a19b
base
spss-modeler_17.1                902d0051-84bd-4af6-ab6b-8f6aa6fdeabb
base
do_12.10                         9100fd72-8159-4eb9-8a0b-a87e12eefa36
base
do_py3.7                         9447fa8b-2051-4d24-9eef-5acb0e3c59f8
base
spark-mllib_3.0-r_3.6            94bb6052-c837-589d-83f1-f4142f219e32
base
cuda-py3.7-opence                94e9652b-7f2d-59d5-ba5a-23a414ea488f
base
nlp-py3.8                        96e60351-99d4-5a1c-9cc0-473ac1b5a864
base
cuda-py3.7                       9a44990c-1aa1-4c7d-baf8-c4099011741c
base
hybrid_0.2                       9b3f9040-9cee-4ead-8d7a-780600f542f7
base
spark-mllib_3.0-py38             9f7a8fc1-4d3c-5e65-ab90-41fa8de2d418
base
autoai-kb_3.3-py3.7              a545cca3-02df-5c61-9e88-998b09dc79af
base
spark-mllib_3.0-py39             a6082a27-5acc-5163-b02c-6b96916eb5e0
base
runtime-22.1-py3.9-do            a7e7dbf1-1d03-5544-994d-e5ec845ce99a
```

```
base
default_py3.8                      ab9e1b80-f2ce-592c-a7d2-4f2344f77194
base
tensorflow_rt22.1-py3.9            acd9c798-6974-5d2f-a657-ce06e986df4d
base
kernel-spark3.2-py3.9              ad7033ee-794e-58cf-812e-a95f4b64b207
base
autoai-obm_2.0 with Spark 3.0      af10f35f-69fa-5d66-9bf5-acb58434263a
base
default_py3.7_opence               c2057dd4-f42c-5f77-a02f-72bdbd3282c9
base
tensorflow_2.1-py3.7               c4032338-2a40-500a-beef-b01ab2667e27
base
do_py3.7_opence                    cc8f8976-b74a-551a-bb66-6377f8d865b4
base
autoai-kb_3.0-py3.6                d139f196-e04b-5d8b-9140-9a10ca1fa91a
base
spark-mllib_3.0-py36               d82546d5-dd78-5fbb-9131-2ec309bc56ed
base
autoai-kb_3.4-py3.8                da9b39c3-758c-5a4f-9cfd-457dd4d8c395
base
kernel-spark3.2-r3.6               db2fe4d6-d641-5d05-9972-73c654c60e0a
base
autoai-kb_rt22.1-py3.9             db6afe93-665f-5910-b117-d879897404d9
base
tensorflow_rt22.1-py3.9-horovod    dda170cc-ca67-5da7-9b7a-cf84c6987fae
base
autoai-ts_1.0-py3.7                deef04f0-0c42-5147-9711-89f9904299db
base
tensorflow_2.1-py3.7-horovod       e384fce5-fdd1-53f8-bc71-11326c9c635f
base
default_py3.7                      e4429883-c883-42b6-87a8-f419d64088cd
base
do_22.1                            e51999ba-6452-5f1f-8287-17228b88b652
base
autoai-obm_3.2                     eae86aab-da30-5229-a6a6-1d0d4e368983
base
do_20.1                            f686cdd9-7904-5f9d-a732-01b0d6b10dc5
base
scikit-learn_0.19-py3.6            f963fa9d-4bb7-5652-9c5d-8d9289ef6ad9
base
tensorflow_2.4-py3.8               fe185c44-9a99-5425-986b-59bd1d2eda46
base
--------------------------------  ------------------------------------
----

import tensorflow as tf
tf.__version__

'2.7.1'
```

```
import keras as ks
ks.__version__
```

```
'2.7.0'
```

```
software_space_uid=client.software_specifications.get_uid_by_name('ten
sorflow_rt22.1-py3.9-horovod')
```

```
software_space_uid
```

```
'dda170cc-ca67-5da7-9b7a-cf84c6987fae'
```

```
model_details=client.repository.store_model(model='Gesture-based-
Radiology-Images.tgz',meta_props={
    client.repository.ModelMetaNames.NAME:"CNN Model Building",
    client.repository.ModelMetaNames.TYPE:'keras_2.2.4',

client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid
})
```

```
Note: Warnings!! :  Model type keras_2.2.4 is deprecated. We recommend
you use a supported model type. See Supported Frameworks
https://dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/pm_se
rvice_supported_frameworks.html
```

```
model_id=client.repository.get_model_id(model_details)
```

```
model_id
```

```
'311d2317-4ede-4507-8cbf-451c147e9846'
```

```
client.repository.download(model_id,'animal.tar.gb')
```

```
Successfully saved model content to file: 'animal.tar.gb'
```

```
'/home/wsuser/work/animal.tar.gb'
```