Image Pre Processing

```
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.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='vRbVaygGdy71MU1EguFc_8FXxSewcIbSFXvAd9xXQBJq',
    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')

bucket = 'emergingmethodsforearlydetectiono-donotdelete-pr-
meoatyzjzinmpq'
object_key = 'Dataset.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 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)

pwd

'/home/wsuser/work'

pwd

'/home/wsuser/work'
```

```
import os
filenames=os.listdir('/home/wsuser/work/Dataset/train_set')
```

Import Keras libraries

```
import keras
```

Importing Image Data Generator from Keras

```
from matplotlib import pyplot as plt
from keras.preprocessing.image import ImageDataGenerator
```

Defining the parameter for image generator class

```
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,rotati
on_range=180,zoom_range=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,rotatio
n_range=180,zoom_range=0.2,horizontal_flip=True)

#Applying image data generator functionality to trainset
x_train=train_datagen.flow_from_directory('/home/wsuser/work/Dataset/
train_set/',
target_size=(150,150),batch_size=32,class_mode='binary')

Found 436 images belonging to 2 classes.

#Applying image data generator functionality to trainset
x_test=test_datagen.flow_from_directory('/home/wsuser/work/Dataset/
test_set',
target_size=(150,150),batch_size=32,class_mode='binary')

Found 121 images belonging to 2 classes.
```

MODEL BUILDING

```
#To define linear intialisation import Sequential
from keras.models import Sequential
#To add layers import Dense
from keras.layers import Dense
#To creat Convolution kernal import Convolution2D
from keras.layers import Convolution2D
#import Maxpooling layer
from keras.layers import MaxPooling2D
#import Flatten layer
from keras.layers import Flatten
import warnings
warnings.filterwarnings('ignore')
```

Initializing the model and Adding CNN and Dense layers

```
#initializing the model
model=Sequential()
#add convolution layer
model.add(Convolution2D(32,
(3,3),input_shape=(150,150,3),activation='relu'))
#add maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
#add convolution layer
model.add(Convolution2D(64,(3,3),activation='relu'))
#add maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
#add convolution layer
model.add(Convolution2D(128,(3,3),activation='relu'))
#add maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
#add convolution layer
model.add(Convolution2D(128,(3,3),activation='relu'))
#add maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
#add flatten layer
model.add(Flatten())

model.add(Dense(512,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
```

Configuring the learning process

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

Training the model

```
r=model.fit(x_train,epochs=10,validation_data=x_test)
```

```
Epoch 1/10
14/14 [==============================] - 85s 6s/step - loss: 0.5977 -
accuracy: 0.7936 - val_loss: 0.2293 - val_accuracy: 0.8926
Epoch 2/10
14/14 [==============================] - 82s 6s/step - loss: 0.3928 -
accuracy: 0.8601 - val_loss: 0.2015 - val_accuracy: 0.9752
Epoch 3/10
14/14 [==============================] - 85s 6s/step - loss: 0.2003 -
accuracy: 0.9266 - val_loss: 0.0283 - val_accuracy: 0.9917
Epoch 4/10
14/14 [==============================] - 84s 6s/step - loss: 0.1417 -
accuracy: 0.9518 - val_loss: 0.0387 - val_accuracy: 0.9917
Epoch 5/10
14/14 [==============================] - 84s 6s/step - loss: 0.1422 -
```

```
accuracy: 0.9427 - val_loss: 0.0265 - val_accuracy: 1.0000
Epoch 6/10
14/14 [==============================] - 83s 6s/step - loss: 0.1635 -
accuracy: 0.9381 - val_loss: 0.0399 - val_accuracy: 1.0000
Epoch 7/10
14/14 [==============================] - 84s 6s/step - loss: 0.1464 -
accuracy: 0.9404 - val_loss: 0.0292 - val_accuracy: 0.9917
Epoch 8/10
14/14 [==============================] - 83s 6s/step - loss: 0.1972 -
accuracy: 0.9197 - val_loss: 0.0668 - val_accuracy: 0.9752
Epoch 9/10
14/14 [==============================] - 84s 6s/step - loss: 0.1758 -
accuracy: 0.9243 - val_loss: 0.0404 - val_accuracy: 1.0000
Epoch 10/10
14/14 [==============================] - 80s 6s/step - loss: 0.1520 -
accuracy: 0.9427 - val_loss: 0.0528 - val_accuracy: 0.9917
```

Save the model

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

```
!tar -zcvf image-classification-model_new.tgz forestalert.h5
```

```
forestalert.h5
```

```
ls -1
```

```
Dataset/
forestalert.h5
image-classification-model_new.tgz
```

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

```
Collecting watson-machine-learning-client
  Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl
(538 kB)
ent 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: tabulate in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(0.8.9)
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: certifi in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (2022.9.24)
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: 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: urllib3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (1.26.7)
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: 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: 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: 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: 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: 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: 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
```

Replacing the credentials that were got from Watson Machine Learning service

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

"apikey" :"4edltW4RsVTMfg_Ni5NrIwkltXTmh3Y8xg3ms3Ysrt1d"
                }
client = APIClient(wml_credentials)

client = APIClient(wml_credentials)

def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    #print(space)
    return(next(item for item in space['resources'] if item['entity']
['name'] == space_name)['metadata']['id'])

space_uid = guid_from_space_name(client, 'Forest fire detection')
print("Space UID = " + space_uid)

Space UID = 4b776d80-ba2d-460b-ac44-dbbc2508e7d5

client.set.default_space(space_uid)

'SUCCESS'

client.software_specifications.list()

-----------------------------  ------------------------------------
----
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
autoai-kb_rt22.2-py3.10         125b6d9a-5b1f-5e8d-972a-b251688ccf40
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
kernel-spark3.3-r3.6            1c9e5454-f216-59dd-a20e-474a5cdf5988
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
kernel-spark3.3-py3.9           2b7961e2-e3b1-5a8c-a491-482c8368839a
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
autoai-ts_rt22.2-py3.10         396b2e83-0953-5b86-9a55-7ce1628a406f
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
```

```
pytorch-onnx_rt22.2-py3.10     40e73f55-783a-5535-b3fa-0c8b94291431
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
runtime-22.2-py3.10-xc         5e8cddff-db4a-5a6a-b8aa-2d4af9864dab
base
autoai-kb_3.1-py3.7            632d4b22-10aa-5180-88f0-f52dfb6444d7
base
-----------------------------  ------------------------------------
----
Note: Only first 50 records were displayed. To display more use
'limit' parameter.
```

```
software_spec_uid =
client.software_specifications.get_uid_by_name("tensorflow_rt22.1-
py3.9")
software_spec_uid
```

```
'acd9c798-6974-5d2f-a657-ce06e986df4d'
```

```
model_details = client.repository.store_model(model='image-
classification-model_new.tgz',meta_props={
client.repository.ModelMetaNames.NAME:"CNN",
client.repository.ModelMetaNames.TYPE:"tensorflow_rt22.1",
client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid}
                                             )
model_id = client.repository.get_model_uid(model_details)
```

```
This method is deprecated, please use get_model_id()
```

```
model_id
```

```
'a4275dc7-5c37-4454-afe9-42efaa67bb58'
```

```
client.repository.download("a4275dc7-5c37-4454-afe9-42efaa67bb58",
'forestfire1.tar.gz')
```

```
Successfully saved model content to file: 'forestfire1.tar.gz'
```

```
'/home/wsuser/work/forestfire1.tar.gz'
```

Predictions(with fire)

```
#import load model from keras.model
from keras.models import load_model
#import image from keras
from tensorflow.keras.preprocessing import image
import numpy as np
#import cv2
import cv2
#load the saved model
model=load_model("forestalert.h5")
img=image.load_img('/home/wsuser/work/Dataset/test_set/with
fire/FORESTFIRE (1).jpg')
x=image.img_to_array(img)
res=cv2.resize(x,dsize=(150,150),interpolation=cv2.INTER_CUBIC)
#expand the image shape
x=np.expand_dims(res,axis=0)
```

```
pred=model.predict(x)
pred = int(pred[0][0])
pred
int(pred)
```

```
1
```

```
if pred==1:
    print('Forest Fire')
elif pred==0:
    print('No Fire')
```

```
Forest Fire
```

Predictions(without fire)

```
#import load model from keras.model
from keras.models import load_model
#import image from keras
from tensorflow.keras.preprocessing import image
```

```
import numpy as np
#import cv2
import cv2
#load the saved model
model=load_model("forestalert.h5")
img=image.load_img('/home/wsuser/work/Dataset/test_set/forest/55967210
1517195076987621071193712n.jpg')
x=image.img_to_array(img)
res=cv2.resize(x,dsize=(150,150),interpolation=cv2.INTER_CUBIC)
#expand the image shape
x=np.expand_dims(res,axis=0)
```

```
pred=model.predict(x)
pred = int(pred[0][0])
pred
int(pred)
```

```
0
```

```
if pred==1:
    print('Forest Fire')
elif pred==0:
    print('No Fire')
```

```
No Fire
```