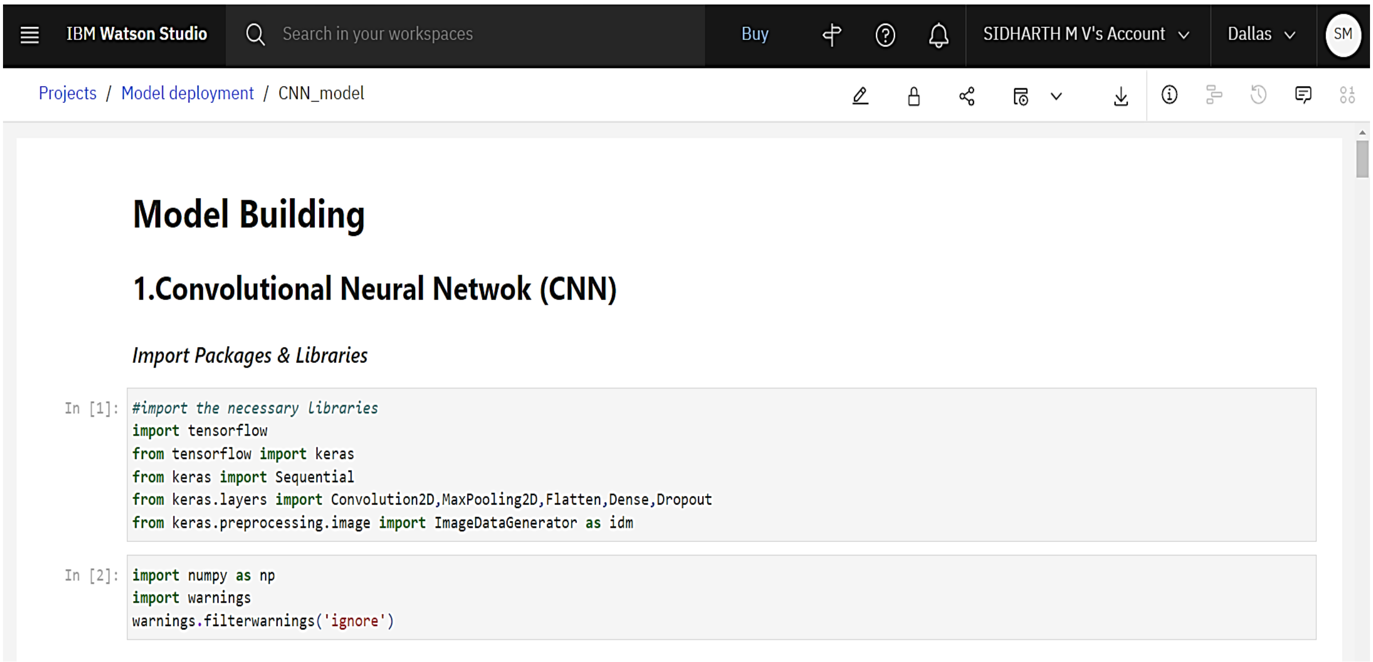
Train the model on IBM Cloud

Team ID	PNT2022TMID27942
Project Name	Project - Natural Disasters Intensity Analysis and Classification using Artificial Intelligence


IBM Watson Studio
Search in your workspaces
Buy
SIDHARTH M V's Account
Dallas
SM
Projects / Model deployment / CNN_model

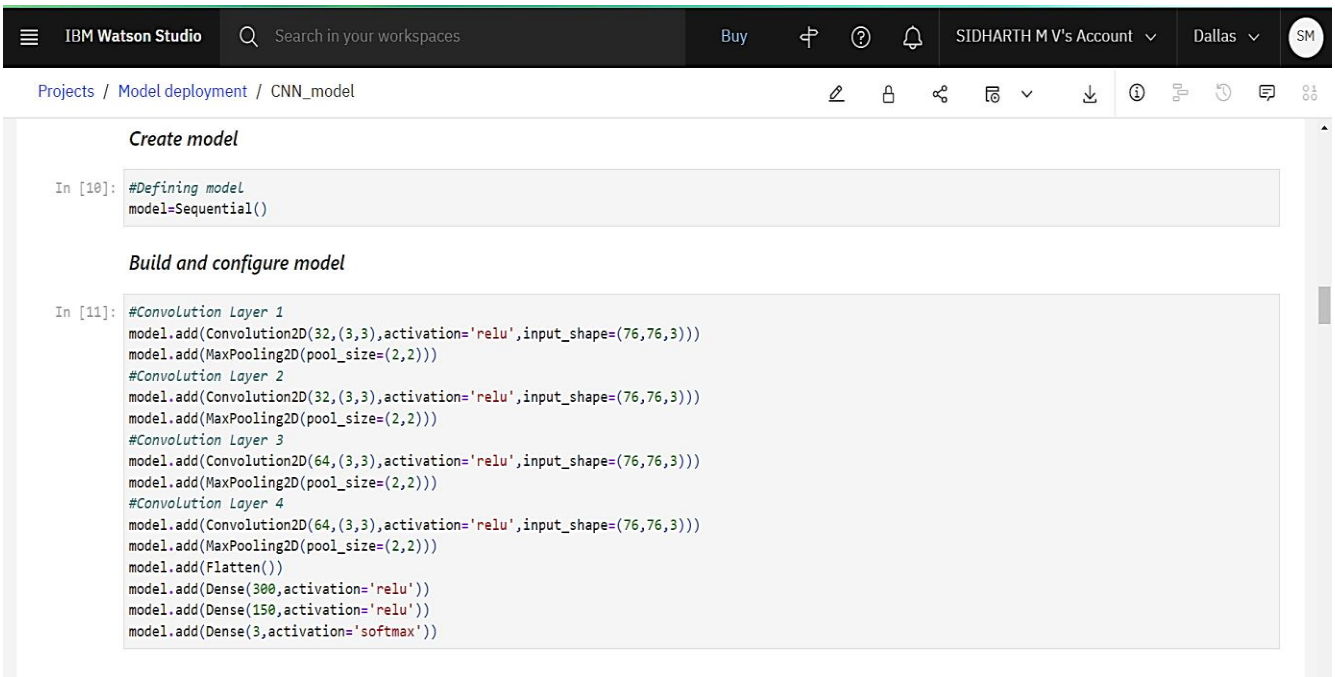
Model Building

1.Convolutional Neural Netwok (CNN)

Import Packages & Libraries

In [1]:
```
#import the necessary libraries
import tensorflow
from tensorflow import keras
from keras import Sequential
from keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense,Dropout
from keras.preprocessing.image import ImageDataGenerator as idm
```

In [2]:
```
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

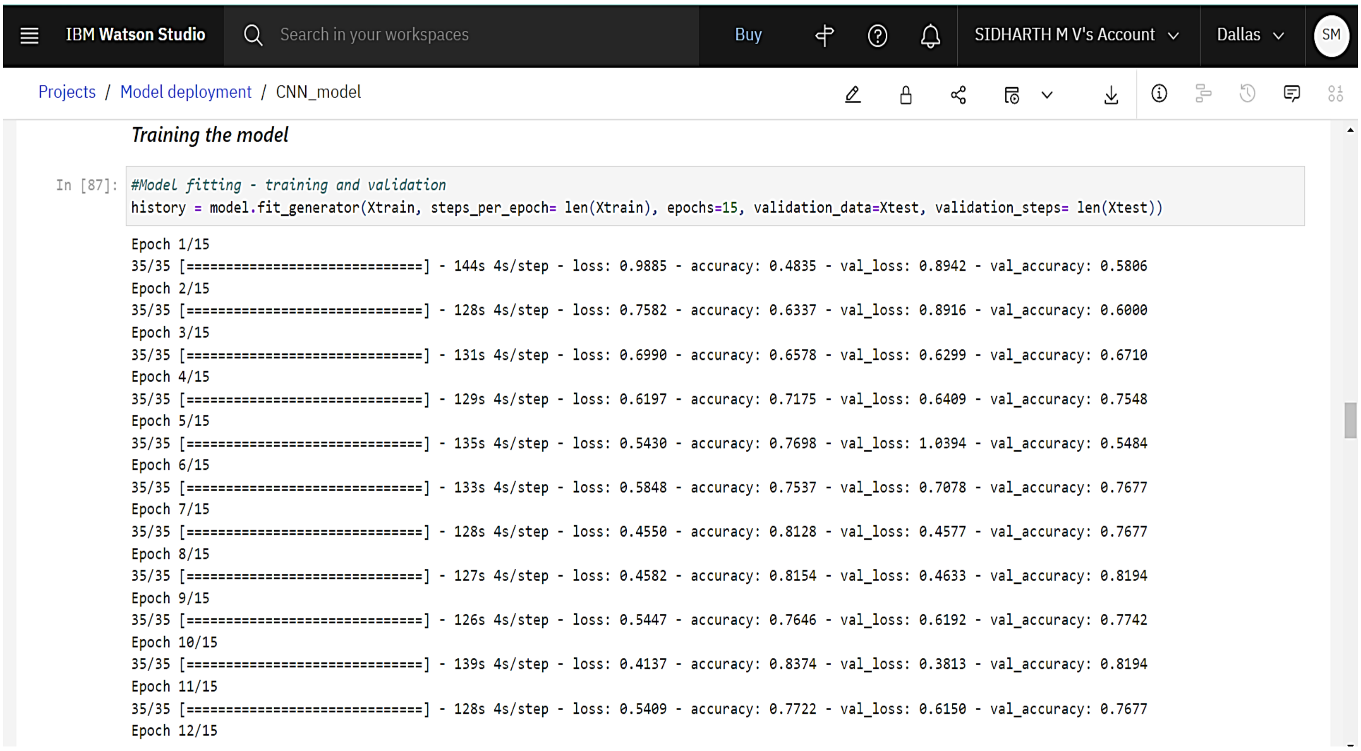

IBM Watson Studio
Search in your workspaces
Buy
SIDHARTH M V's Account
Dallas
SM
Projects / Model deployment / CNN_model

Create model

In [10]:
```
#Defining model
model=Sequential()
```

Build and configure model

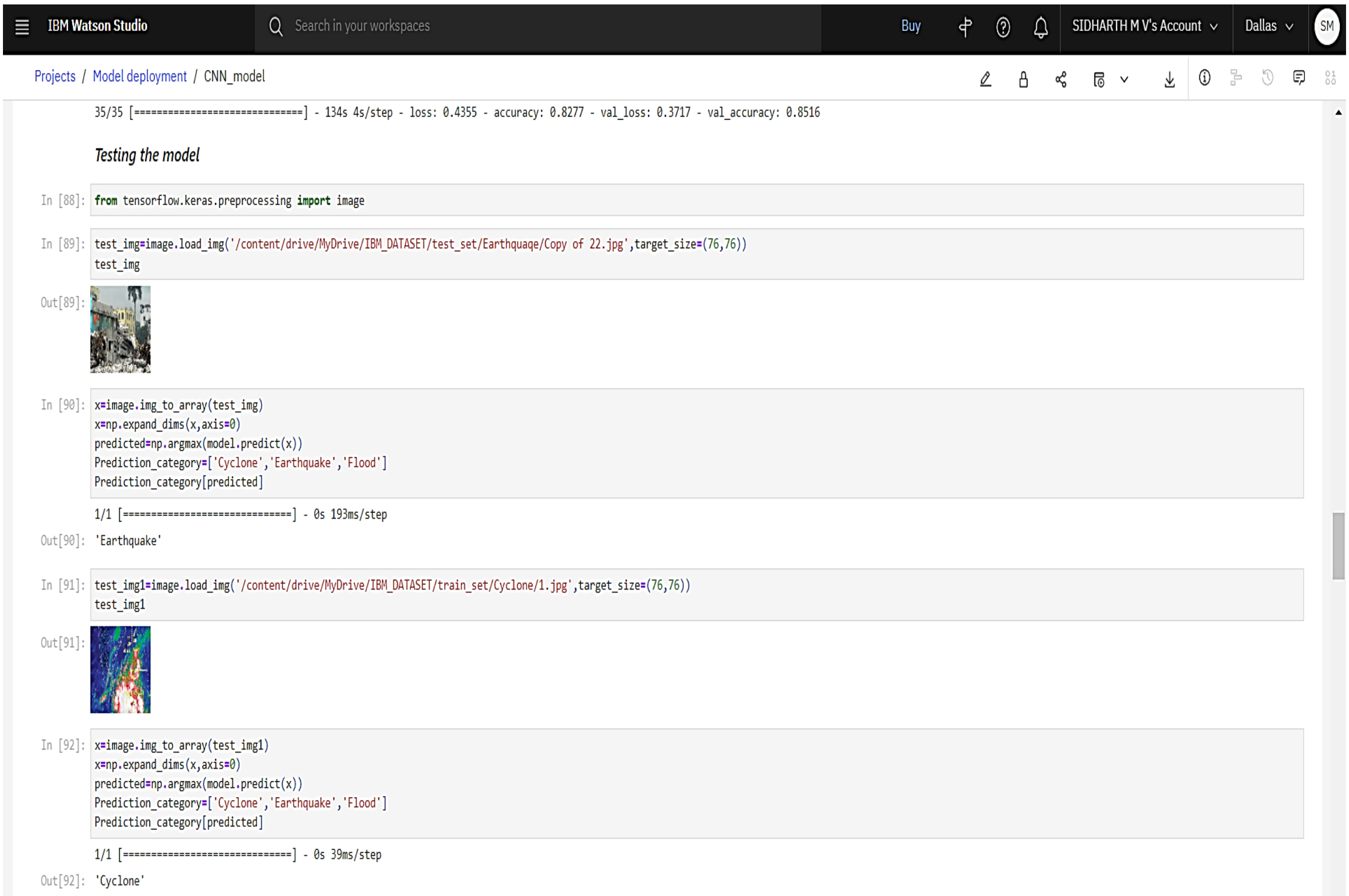
In [11]:
```
#Convolution Layer 1
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(76,76,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
#Convolution Layer 2
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(76,76,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
#Convolution Layer 3
model.add(Convolution2D(64,(3,3),activation='relu',input_shape=(76,76,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
#Convolution Layer 4
model.add(Convolution2D(64,(3,3),activation='relu',input_shape=(76,76,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))
model.add(Dense(3,activation='softmax'))
```



IBM Watson Studio
Search in your workspaces
Buy
SIDHARTH M V's Account
Dallas
SM
Projects / Model deployment / CNN_model

Training the model

```
In [87]: #Model fitting - training and validation
         history = model.fit_generator(Xtrain, steps_per_epoch= len(Xtrain), epochs=15, validation_data=Xtest, validation_steps= len(Xtest))

Epoch 1/15
35/35 [==============================] - 144s 4s/step - loss: 0.9885 - accuracy: 0.4835 - val_loss: 0.8942 - val_accuracy: 0.5806
Epoch 2/15
35/35 [==============================] - 128s 4s/step - loss: 0.7582 - accuracy: 0.6337 - val_loss: 0.8916 - val_accuracy: 0.6000
Epoch 3/15
35/35 [==============================] - 131s 4s/step - loss: 0.6990 - accuracy: 0.6578 - val_loss: 0.6299 - val_accuracy: 0.6710
Epoch 4/15
35/35 [==============================] - 129s 4s/step - loss: 0.6197 - accuracy: 0.7175 - val_loss: 0.6409 - val_accuracy: 0.7548
Epoch 5/15
35/35 [==============================] - 135s 4s/step - loss: 0.5430 - accuracy: 0.7698 - val_loss: 1.0394 - val_accuracy: 0.5484
Epoch 6/15
35/35 [==============================] - 133s 4s/step - loss: 0.5848 - accuracy: 0.7537 - val_loss: 0.7078 - val_accuracy: 0.7677
Epoch 7/15
35/35 [==============================] - 128s 4s/step - loss: 0.4550 - accuracy: 0.8128 - val_loss: 0.4577 - val_accuracy: 0.7677
Epoch 8/15
35/35 [==============================] - 127s 4s/step - loss: 0.4582 - accuracy: 0.8154 - val_loss: 0.4633 - val_accuracy: 0.8194
Epoch 9/15
35/35 [==============================] - 126s 4s/step - loss: 0.5447 - accuracy: 0.7646 - val_loss: 0.6192 - val_accuracy: 0.7742
Epoch 10/15
35/35 [==============================] - 139s 4s/step - loss: 0.4137 - accuracy: 0.8374 - val_loss: 0.3813 - val_accuracy: 0.8194
Epoch 11/15
35/35 [==============================] - 128s 4s/step - loss: 0.5409 - accuracy: 0.7722 - val_loss: 0.6150 - val_accuracy: 0.7677
Epoch 12/15
```

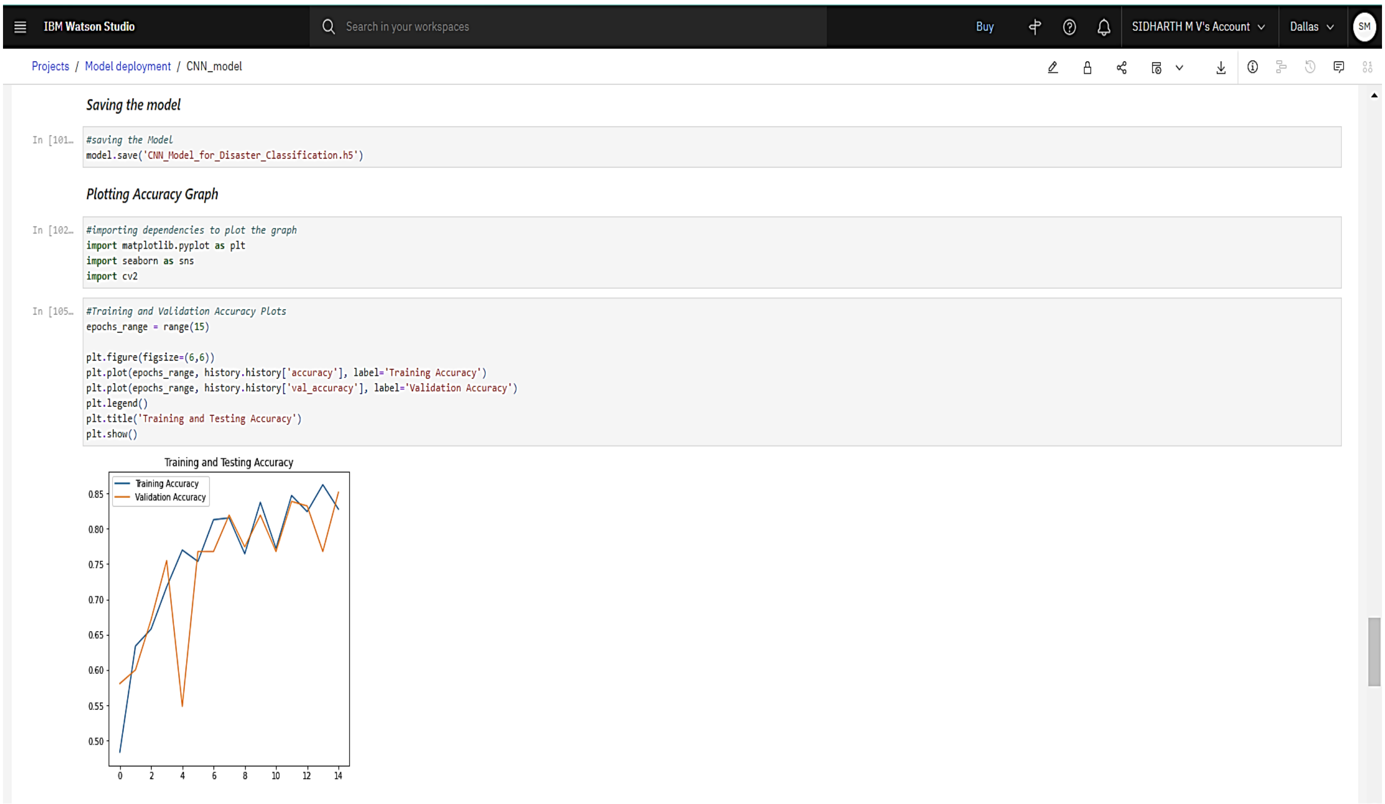


IBM Watson Studio
Search in your workspaces
Buy
SIDHARTH M V's Account
Dallas
SM
Projects / Model deployment / CNN_model

```
35/35 [==============================] - 134s 4s/step - loss: 0.4355 - accuracy: 0.8277 - val_loss: 0.3717 - val_accuracy: 0.8516
```

Testing the model

```
In [88]: from tensorflow.keras.preprocessing import image

In [89]: test_img=image.load_img('/content/drive/MyDrive/IBM_DATASET/test_set/Earthquaqe/Copy of 22.jpg',target_size=(76,76))
         test_img

Out[89]:

In [90]: x=image.img_to_array(test_img)
         x=np.expand_dims(x,axis=0)
         predicted=np.argmax(model.predict(x))
         Prediction_category=['Cyclone','Earthquake','Flood']
         Prediction_category[predicted]

1/1 [==============================] - 0s 193ms/step

Out[90]: 'Earthquake'

In [91]: test_img1=image.load_img('/content/drive/MyDrive/IBM_DATASET/train_set/Cyclone/1.jpg',target_size=(76,76))
         test_img1

Out[91]:

In [92]: x=image.img_to_array(test_img1)
         x=np.expand_dims(x,axis=0)
         predicted=np.argmax(model.predict(x))
         Prediction_category=['Cyclone','Earthquake','Flood']
         Prediction_category[predicted]

1/1 [==============================] - 0s 39ms/step

Out[92]: 'Cyclone'
```



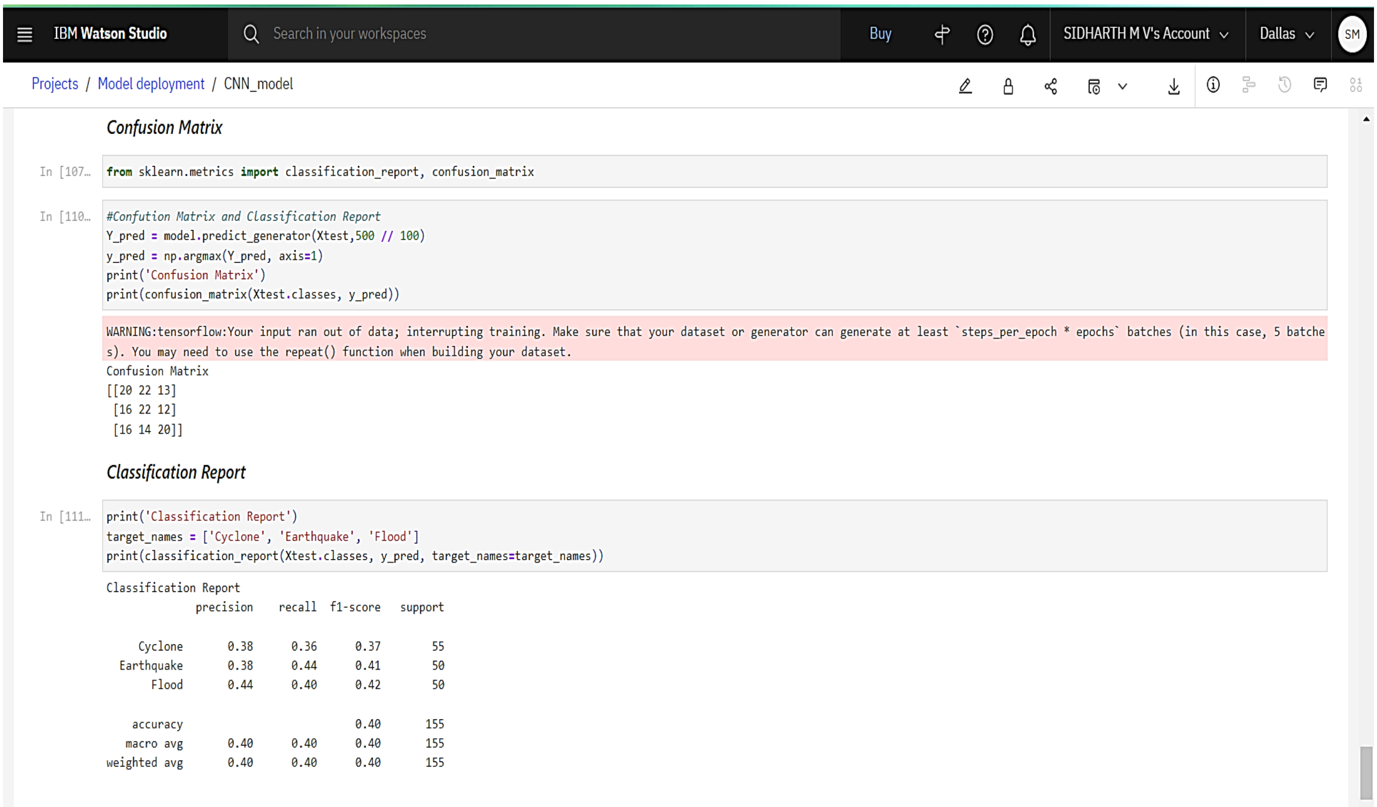
Saving the model

```
In [101…
#saving the Model
model.save('CNN_Model_for_Disaster_Classification.h5')
```

Plotting Accuracy Graph

```
In [102…
#importing dependencies to plot the graph
import matplotlib.pyplot as plt
import seaborn as sns
import cv2
```

```
In [105…
#Training and Validation Accuracy Plots
epochs_range = range(15)

plt.figure(figsize=(6,6))
plt.plot(epochs_range, history.history['accuracy'], label='Training Accuracy')
plt.plot(epochs_range, history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title('Training and Testing Accuracy')
plt.show()
```


IBM Watson Studio
Search in your workspaces
Buy
SIDHARTH M V's Account
Dallas
SM
Projects / Model deployment / CNN_model

Confusion Matrix

```
In [107…
from sklearn.metrics import classification_report, confusion_matrix
```

```
In [110…
#Confution Matrix and Classification Report
Y_pred = model.predict_generator(Xtest,500 // 100)
y_pred = np.argmax(Y_pred, axis=1)
print('Confusion Matrix')
print(confusion_matrix(Xtest.classes, y_pred))
```

```
WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your dataset or generator can generate at least `steps_per_epoch * epochs` batches (in this case, 5 batche
s). You may need to use the repeat() function when building your dataset.
Confusion Matrix
[[20 22 13]
 [16 22 12]
 [16 14 20]]
```

Classification Report

```
In [111…
print('Classification Report')
target_names = ['Cyclone', 'Earthquake', 'Flood']
print(classification_report(Xtest.classes, y_pred, target_names=target_names))
```

```
Classification Report
              precision    recall  f1-score   support

     Cyclone       0.38      0.36      0.37        55
  Earthquake       0.38      0.44      0.41        50
       Flood       0.44      0.40      0.42        50

    accuracy                           0.40       155
   macro avg       0.40      0.40      0.40       155
weighted avg       0.40      0.40      0.40       155
```