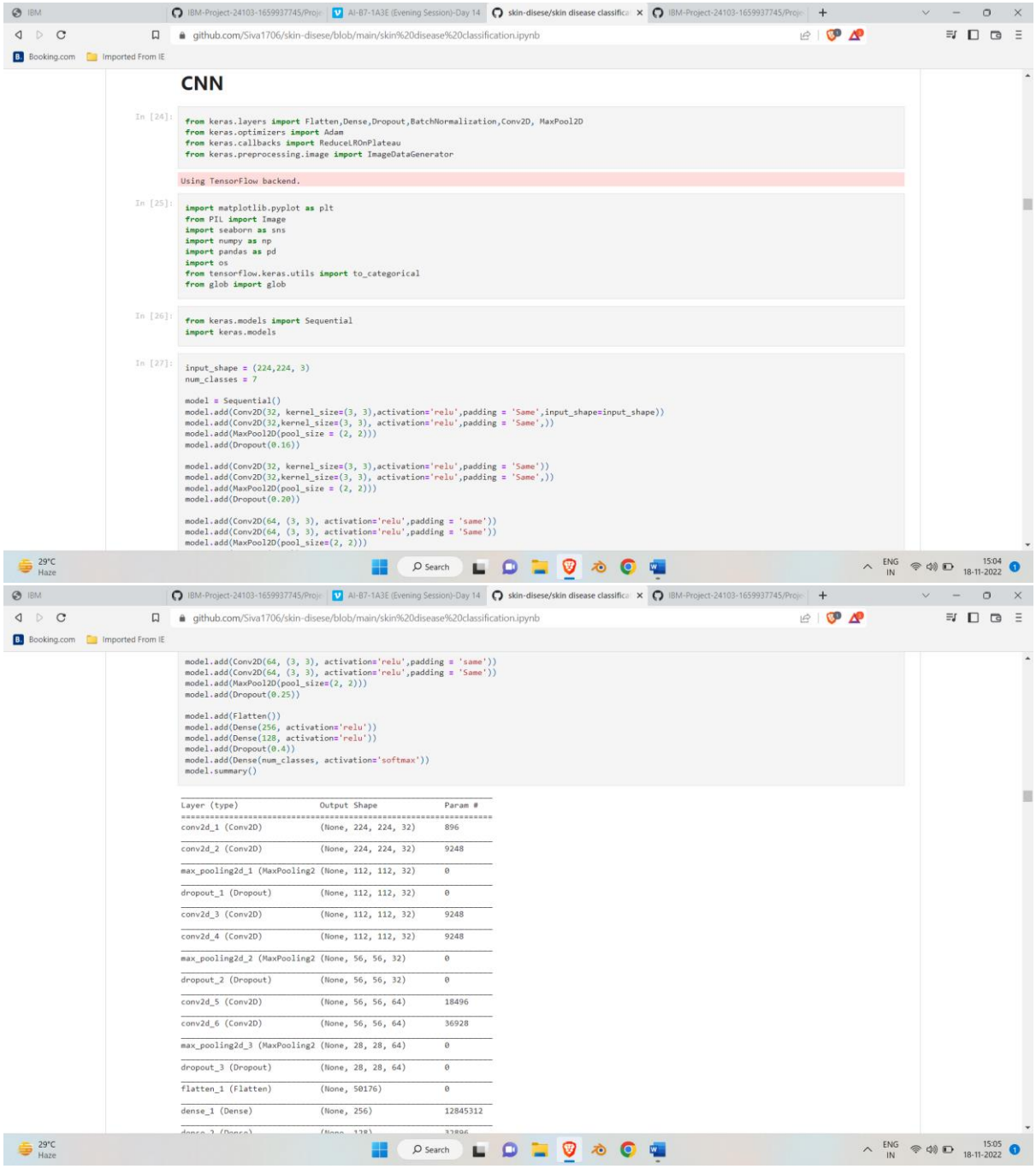
Image Processing:



CNN

In [24]:
```
from keras.layers import Flatten,Dense,Dropout,BatchNormalization,Conv2D, MaxPool2D
from keras.optimizers import Adam
from keras.callbacks import ReduceLROnPlateau
from keras.preprocessing.image import ImageDataGenerator
```

```
Using TensorFlow backend.
```

In [25]:
```
import matplotlib.pyplot as plt
from PIL import Image
import seaborn as sns
import numpy as np
import pandas as pd
import os
from tensorflow.keras.utils import to_categorical
from glob import glob
```

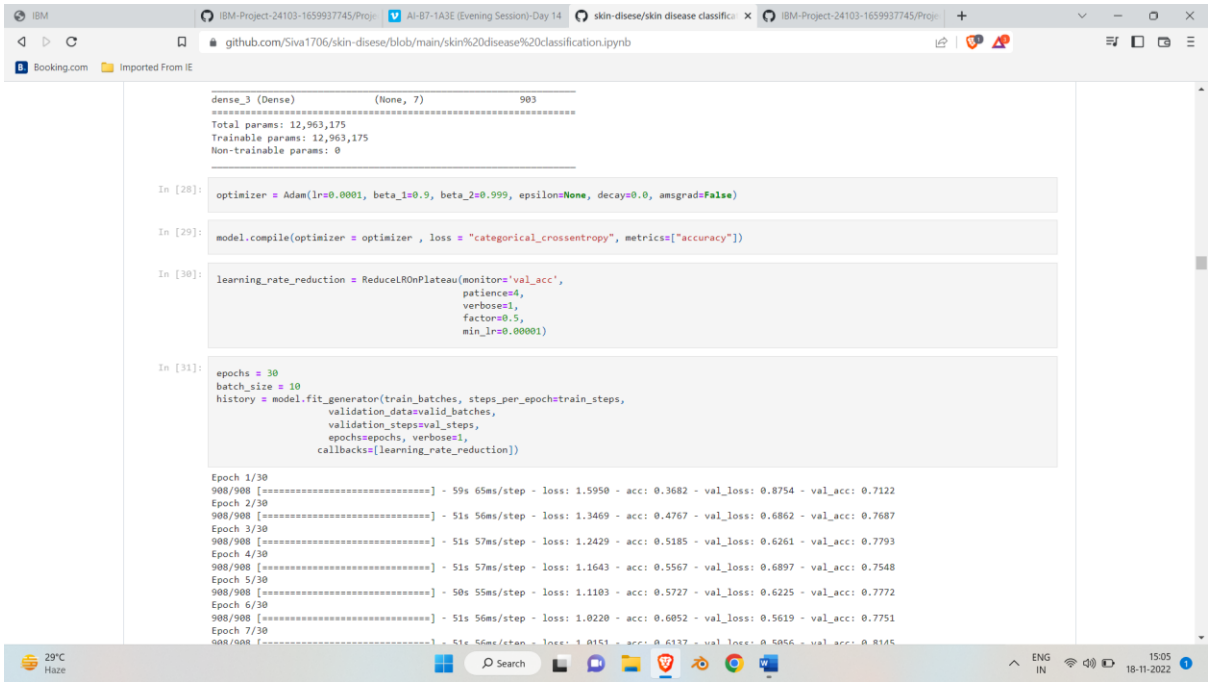
In [26]:
```
from keras.models import Sequential
import keras.models
```

In [27]:
```
input_shape = (224,224, 3)
num_classes = 7

model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',padding = 'Same',input_shape=input_shape))
model.add(Conv2D(32,kernel_size=(3, 3), activation='relu',padding = 'Same',))
model.add(MaxPool2D(pool_size = (2, 2)))
model.add(Dropout(0.16))

model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',padding = 'Same'))
model.add(Conv2D(32,kernel_size=(3, 3), activation='relu',padding = 'Same',))
model.add(MaxPool2D(pool_size = (2, 2)))
model.add(Dropout(0.20))

model.add(Conv2D(64, (3, 3), activation='relu',padding = 'same'))
model.add(Conv2D(64, (3, 3), activation='relu',padding = 'Same'))
model.add(MaxPool2D(pool_size=(2, 2)))
```



```
model.add(Conv2D(64, (3, 3), activation='relu',padding = 'same'))
model.add(Conv2D(64, (3, 3), activation='relu',padding = 'Same'))
model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(num_classes, activation='softmax'))
model.summary()
```

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 224, 224, 32)	896
conv2d_2 (Conv2D)	(None, 224, 224, 32)	9248
max_pooling2d_1 (MaxPooling2	(None, 112, 112, 32)	0
dropout_1 (Dropout)	(None, 112, 112, 32)	0
conv2d_3 (Conv2D)	(None, 112, 112, 32)	9248
conv2d_4 (Conv2D)	(None, 112, 112, 32)	9248
max_pooling2d_2 (MaxPooling2	(None, 56, 56, 32)	0
dropout_2 (Dropout)	(None, 56, 56, 32)	0
conv2d_5 (Conv2D)	(None, 56, 56, 64)	18496
conv2d_6 (Conv2D)	(None, 56, 56, 64)	36928
max_pooling2d_3 (MaxPooling2	(None, 28, 28, 64)	0
dropout_3 (Dropout)	(None, 28, 28, 64)	0
flatten_1 (Flatten)	(None, 50176)	0
dense_1 (Dense)	(None, 256)	12845312
dense_2 (Dense)	(None, 128)	32896


github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb
dense_3 (Dense) (None, 7) 903
Total params: 12,963,175
Trainable params: 12,963,175
Non-trainable params: 0
In [28]:
optimizer = Adam(lr=0.0001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False)
In [29]:
model.compile(optimizer = optimizer , loss = "categorical_crossentropy", metrics=["accuracy"])
In [30]:
learning_rate_reduction = ReduceLROnPlateau(monitor='val_acc',
patience=4,
verbose=1,
factor=0.5,
min_lr=0.00001)
In [31]:
epochs = 30
batch_size = 10
history = model.fit_generator(train_batches, steps_per_epoch=train_steps,
validation_data=valid_batches,
validation_steps=val_steps,
epochs=epochs, verbose=1,
callbacks=[learning_rate_reduction])
Epoch 1/30
908/908 [==========] - 59s 65ms/step - loss: 1.5950 - acc: 0.3682 - val_loss: 0.8754 - val_acc: 0.7122
Epoch 2/30
908/908 [==========] - 51s 56ms/step - loss: 1.3469 - acc: 0.4767 - val_loss: 0.6862 - val_acc: 0.7687
Epoch 3/30
908/908 [==========] - 51s 57ms/step - loss: 1.2429 - acc: 0.5185 - val_loss: 0.6261 - val_acc: 0.7793
Epoch 4/30
908/908 [==========] - 51s 57ms/step - loss: 1.1643 - acc: 0.5567 - val_loss: 0.6897 - val_acc: 0.7548
Epoch 5/30
908/908 [==========] - 50s 55ms/step - loss: 1.1103 - acc: 0.5727 - val_loss: 0.6225 - val_acc: 0.7772
Epoch 6/30
908/908 [==========] - 51s 56ms/step - loss: 1.0220 - acc: 0.6052 - val_loss: 0.5619 - val_acc: 0.7751
Epoch 7/30
29°C Haze
Search
ENG IN
15:05 18-11-2022


github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb
Epoch 12/30
908/908 [==========] - 51s 56ms/step - loss: 0.8422 - acc: 0.6783 - val_loss: 0.4474 - val_acc: 0.8348
Epoch 13/30
908/908 [==========] - 51s 56ms/step - loss: 0.7871 - acc: 0.7030 - val_loss: 0.4935 - val_acc: 0.8230
Epoch 14/30
908/908 [==========] - 51s 56ms/step - loss: 0.7359 - acc: 0.7214 - val_loss: 0.4791 - val_acc: 0.8113
Epoch 15/30
908/908 [==========] - 51s 56ms/step - loss: 0.7353 - acc: 0.7214 - val_loss: 0.4869 - val_acc: 0.8177
Epoch 16/30
908/908 [==========] - 52s 57ms/step - loss: 0.7133 - acc: 0.7320 - val_loss: 0.4269 - val_acc: 0.8305
Epoch 00016: ReduceLROnPlateau reducing learning rate to 4.999999873689376e-05.
Epoch 17/30
908/908 [==========] - 52s 57ms/step - loss: 0.6611 - acc: 0.7483 - val_loss: 0.4422 - val_acc: 0.8358
Epoch 18/30
908/908 [==========] - 51s 57ms/step - loss: 0.5641 - acc: 0.7840 - val_loss: 0.4380 - val_acc: 0.8497
Epoch 19/30
908/908 [==========] - 52s 57ms/step - loss: 0.5707 - acc: 0.7862 - val_loss: 0.4967 - val_acc: 0.8166
Epoch 20/30
908/908 [==========] - 51s 57ms/step - loss: 0.5524 - acc: 0.7934 - val_loss: 0.4569 - val_acc: 0.8390
Epoch 21/30
908/908 [==========] - 52s 57ms/step - loss: 0.5645 - acc: 0.7852 - val_loss: 0.4332 - val_acc: 0.8422
Epoch 22/30
908/908 [==========] - 51s 57ms/step - loss: 0.5230 - acc: 0.8036 - val_loss: 0.4826 - val_acc: 0.8348
Epoch 00022: ReduceLROnPlateau reducing learning rate to 2.499999936844688e-05.
Epoch 23/30
908/908 [==========] - 51s 57ms/step - loss: 0.4699 - acc: 0.8209 - val_loss: 0.4523 - val_acc: 0.8454
Epoch 24/30
908/908 [==========] - 51s 57ms/step - loss: 0.4454 - acc: 0.8312 - val_loss: 0.4741 - val_acc: 0.8401
Epoch 25/30
908/908 [==========] - 52s 57ms/step - loss: 0.4696 - acc: 0.8217 - val_loss: 0.4558 - val_acc: 0.8390
Epoch 26/30
908/908 [==========] - 51s 56ms/step - loss: 0.4278 - acc: 0.8371 - val_loss: 0.4708 - val_acc: 0.8443
Epoch 00026: ReduceLROnPlateau reducing learning rate to 1.249999968422344e-05.
Epoch 27/30
908/908 [==========] - 50s 56ms/step - loss: 0.4018 - acc: 0.8550 - val_loss: 0.4834 - val_acc: 0.8412
Epoch 28/30
908/908 [==========] - 51s 56ms/step - loss: 0.3987 - acc: 0.8500 - val_loss: 0.4679 - val_acc: 0.8561
Epoch 29/30
908/908 [==========] - 51s 57ms/step - loss: 0.3987 - acc: 0.8532 - val_loss: 0.4788 - val_acc: 0.8348
Epoch 30/30
908/908 [==========] - 53s 59ms/step - loss: 0.3897 - acc: 0.8571 - val_loss: 0.4808 - val_acc: 0.8507
29°C Haze
Search
ENG IN
15:05 18-11-2022

```
Epoch 28/30
908/908 [==============================] - 51s 56ms/step - loss: 0.3987 - acc: 0.8500 - val_loss: 0.4679 - val_acc: 0.8561
Epoch 29/30
908/908 [==============================] - 51s 57ms/step - loss: 0.3987 - acc: 0.8532 - val_loss: 0.4788 - val_acc: 0.8348
Epoch 30/30
908/908 [==============================] - 53s 59ms/step - loss: 0.3897 - acc: 0.8571 - val_loss: 0.4808 - val_acc: 0.8507
```

In [32]:

```
from sklearn.metrics import classification_report,confusion_matrix
```

In [33]:

```
from keras.utils.vis_utils import plot_model
plot_model(model, to_file='model_plot.png', show_shapes=True, show_layer_names=True)
```

In [34]:

```
val_loss, val_cat_acc = \
model.evaluate_generator(test_batches,
                         steps=len(df_val))

print('val_loss:', val_loss)
print('val_cat_acc:', val_cat_acc)

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

```
val_loss: 0.48075364217932615
val_cat_acc: 0.8507462686567164
```

In [35]:

```
model.metrics_names
```

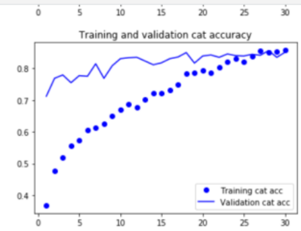
Out[35]:

```
['loss', 'acc']
```

In [36]:

```
import matplotlib.pyplot as plt

acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)

plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
```

github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb



In [37]:

```
test_labels = test_batches.classes
```

In [38]:

```
test_labels
```

Out[38]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
       3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6], dtype=int32)
```

In [39]:
```
test_batches.class_indices
```

Out[39]:
```
{'akiec': 0, 'bcc': 1, 'bkl': 2, 'df': 3, 'mel': 4, 'nv': 5, 'vasc': 6}
```

In [40]:
```
predictions = model.predict_generator(test_batches, steps=len(df_val), verbose=1)
```

```
938/938 [==============================] - 12s 12ms/step
```

In [41]:
```
predictions.shape
```

Out[41]:
```
(938, 7)
```

In [42]:
```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
```


IBM | IBM-Project-24103-1659937745/Proje | AI-B7-1A3E (Evening Session)-Day 14 | skin-disese/skin disease classifica | IBM-Project-24103-1659937745/Proje
github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb
Booking.com | Imported From IE

Out[41]: (938, 7)

In [42]:

```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')
    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.tight_layout()
```

In [43]:
```
test_labels.shape
```

Out[43]:
```
(938,)
```

In [44]:
```
cm = confusion_matrix(test_labels, predictions.argmax(axis=1))
```

29°C Haze | Search | ENG IN | 15:05 18-11-2022

Out[43]: (938,)

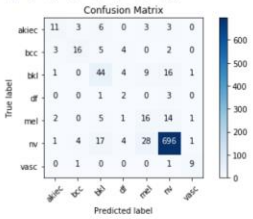
In [44]:
```
cm = confusion_matrix(test_labels, predictions.argmax(axis=1))
```

In [45]:
```
test_batches.class_indices
```

Out[45]: {'akiec': 0, 'bcc': 1, 'bkl': 2, 'df': 3, 'mel': 4, 'nv': 5, 'vasc': 6}

In [46]:
```
cm_plot_labels = ['akiec', 'bcc', 'bkl', 'df', 'mel','nv', 'vasc']

plot_confusion_matrix(cm, cm_plot_labels, title='Confusion Matrix')
```

```
Confusion matrix, without normalization
[[ 11   3   6   0   3   3   0]
 [  3  16   5   4   0   2   0]
 [  1   0  44   4   9  16   1]
 [  0   0   1   2   0   3   0]
 [  2   0   5   1  16  14   1]
 [  1   4  17   4  28 696   1]
 [  0   1   0   0   0   1   9]]
```



In [47]:
```
y_pred = np.argmax(predictions, axis=1)
```

Predicted label

In [47]:
```
y_pred = np.argmax(predictions, axis=1)
y_true = test_batches.classes
```

In [48]:
```
from sklearn.metrics import classification_report

report = classification_report(y_true, y_pred, target_names=cm_plot_labels)

print(report)
```

```
             precision    recall  f1-score   support

      akiec       0.61      0.42      0.50        26
        bcc       0.67      0.53      0.59        30
        bkl       0.56      0.59      0.58        75
         df       0.13      0.33      0.19         6
        mel       0.29      0.41      0.34        39
         nv       0.95      0.93      0.94       751
       vasc       0.75      0.82      0.78        11

avg / total       0.86      0.85      0.85       938
```

MobileNet

In [49]:
```
from numpy.random import seed
seed(101)
from tensorflow import set_random_seed
set_random_seed(101)

import pandas as pd
import numpy as np

import tensorflow
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.metrics import categorical_crossentropy
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
import os

from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
import itertools
import shutil
import matplotlib.pyplot as plt
%matplotlib inline

from tensorflow.keras.metrics import categorical_accuracy, top_k_categorical_accuracy
import matplotlib.pyplot as plt
```

In [50]:

```
datagen = ImageDataGenerator(
    preprocessing_function= \
    tensorflow.keras.applications.mobilenet.preprocess_input)

train_batches = datagen.flow_from_directory(train_path,
                                            target_size=(image_size,image_size),
                                            batch_size=train_batch_size)

valid_batches = datagen.flow_from_directory(valid_path,
                                            target_size=(image_size,image_size),
                                            batch_size=val_batch_size)

test_batches = datagen.flow_from_directory(valid_path,
                                            target_size=(image_size,image_size),
                                            batch_size=1,
                                            shuffle=False)
```

```
Found 38569 images belonging to 7 classes.
Found 938 images belonging to 7 classes.
Found 938 images belonging to 7 classes.
```

Modify MobileNet Model

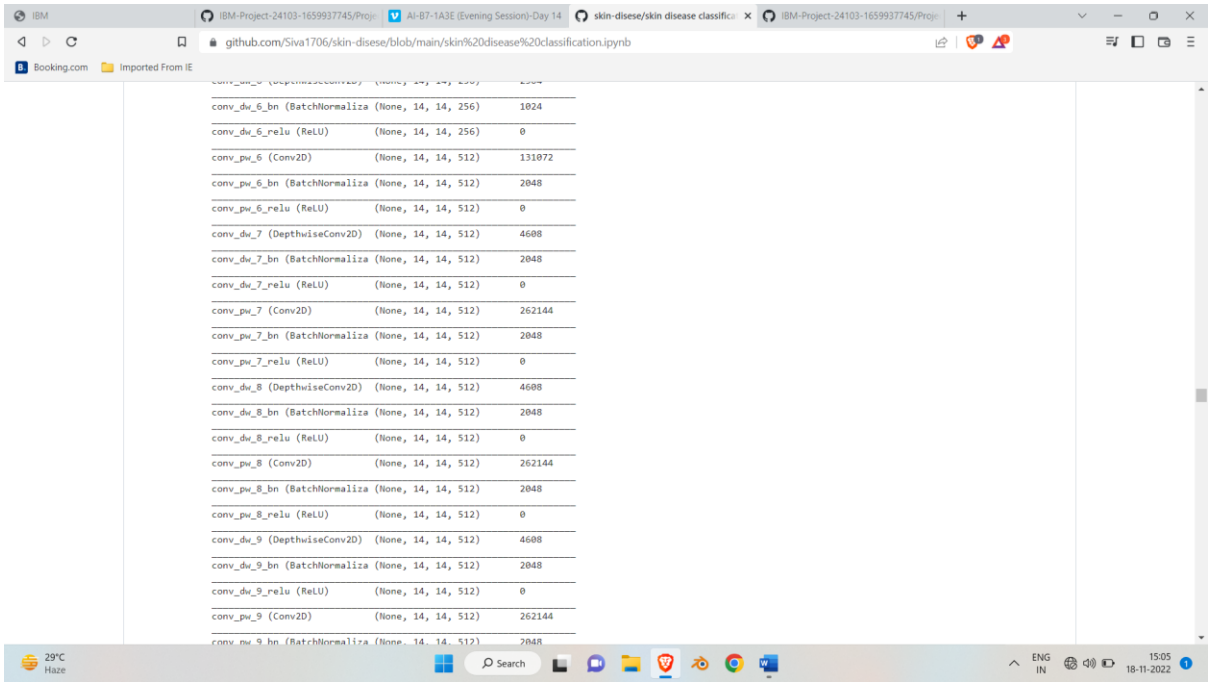
In [51]:

```
mobile = tensorflow.keras.applications.mobilenet.MobileNet()
```

```
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.6/mobilenet_1_0_224_tf.h5
17227776/17225924 [==============================] - 0s 0us/step
```

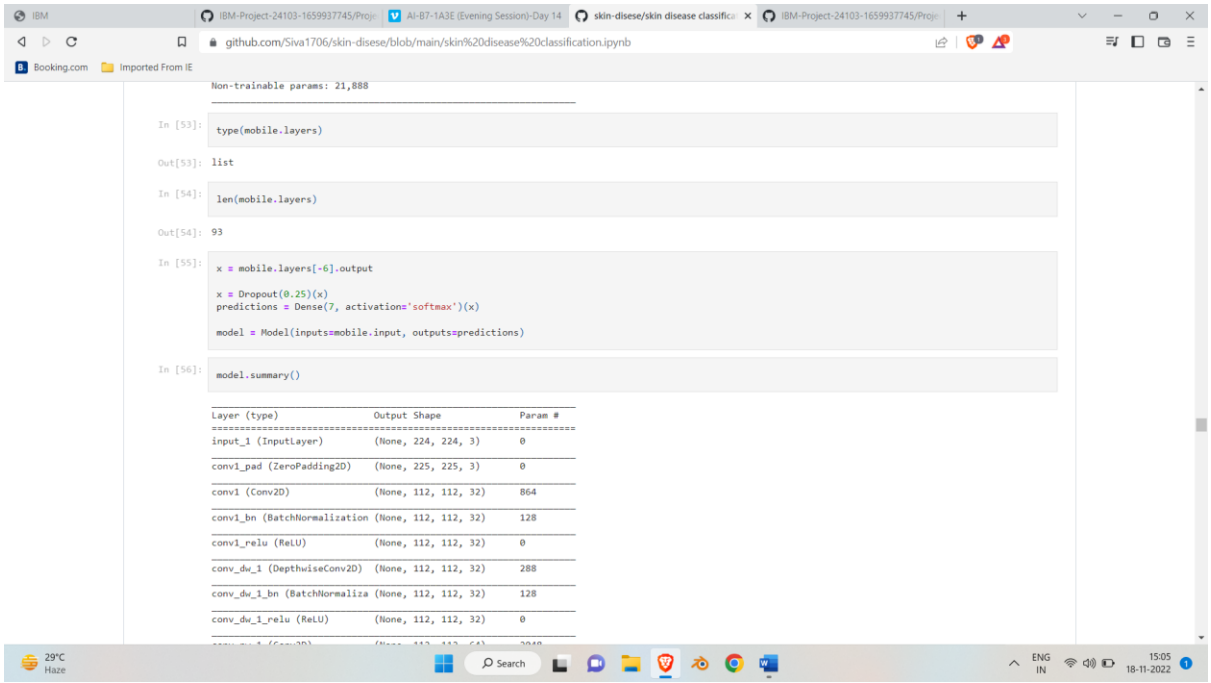
In [52]:

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
conv1_pad (ZeroPadding2D)	(None, 225, 225, 3)	0
conv1 (Conv2D)	(None, 112, 112, 32)	864
conv1_bn (BatchNormalization	(None, 112, 112, 32)	128
conv1_relu (ReLU)	(None, 112, 112, 32)	0
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288
conv_dw_1_bn (BatchNormaliza	(None, 112, 112, 32)	128
conv_dw_1_relu (ReLU)	(None, 112, 112, 32)	0
conv_pw_1 (Conv2D)	(None, 112, 112, 64)	2048
conv_pw_1_bn (BatchNormaliza	(None, 112, 112, 64)	256
conv_pw_1_relu (ReLU)	(None, 112, 112, 64)	0
conv_pad_2 (ZeroPadding2D)	(None, 113, 113, 64)	0
conv_dw_2 (DepthwiseConv2D)	(None, 56, 56, 64)	576
conv_dw_2_bn (BatchNormaliza	(None, 56, 56, 64)	256
conv_dw_2_relu (ReLU)	(None, 56, 56, 64)	0
conv_pw_2 (Conv2D)	(None, 56, 56, 128)	8192
conv_pw_2_bn (BatchNormaliza	(None, 56, 56, 128)	512
conv_pw_2_relu (ReLU)	(None, 56, 56, 128)	0
conv_dw_3 (DepthwiseConv2D)	(None, 56, 56, 128)	1152
conv_dw_3_bn (BatchNormaliza	(None, 56, 56, 128)	512
conv_dw_3_relu (ReLU)	(None, 56, 56, 128)	0



conv_dw_6_bn (BatchNormaliza	(None, 14, 14, 256)	1024
conv_dw_6_relu (ReLU)	(None, 14, 14, 256)	0
conv_pw_6 (Conv2D)	(None, 14, 14, 512)	131072
conv_pw_6_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_pw_6_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_7 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_7_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_dw_7_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_7 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_7_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_pw_7_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_8 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_8_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_dw_8_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_8 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_8_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_pw_8_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_9 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_9_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_dw_9_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_9 (Conv2D)	(None, 14, 14, 512)	262144

conv_pw_10_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_pw_10_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_11 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_11_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_dw_11_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_11 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_11_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_pw_11_relu (ReLU)	(None, 14, 14, 512)	0
conv_pad_12 (ZeroPadding2D)	(None, 15, 15, 512)	0
conv_dw_12 (DepthwiseConv2D)	(None, 7, 7, 512)	4608
conv_dw_12_bn (BatchNormaliz	(None, 7, 7, 512)	2048
conv_dw_12_relu (ReLU)	(None, 7, 7, 512)	0
conv_pw_12 (Conv2D)	(None, 7, 7, 1024)	524288
conv_pw_12_bn (BatchNormaliz	(None, 7, 7, 1024)	4096
conv_pw_12_relu (ReLU)	(None, 7, 7, 1024)	0
conv_dw_13 (DepthwiseConv2D)	(None, 7, 7, 1024)	9216
conv_dw_13_bn (BatchNormaliz	(None, 7, 7, 1024)	4096
conv_dw_13_relu (ReLU)	(None, 7, 7, 1024)	0
conv_pw_13 (Conv2D)	(None, 7, 7, 1024)	1048576
conv_pw_13_bn (BatchNormaliz	(None, 7, 7, 1024)	4096
conv_pw_13_relu (ReLU)	(None, 7, 7, 1024)	0
global_average_pooling2d (Gl	(None, 1024)	0


IBM
IBM-Project-24103-1659937745/Proje
AI-B7-1A3E (Evening Session)-Day 14
skin-disese/skin disease classifica
IBM-Project-24103-1659937745/Proje
github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb
Booking.com
Imported From IE
Non-trainable params: 21,888
In [53]:
type(mobile.layers)
Out[53]: list
In [54]:
len(mobile.layers)
Out[54]: 93
In [55]:
x = mobile.layers[-6].output
x = Dropout(0.25)(x)
predictions = Dense(7, activation='softmax')(x)
model = Model(inputs=mobile.input, outputs=predictions)
In [56]:
model.summary()
Layer (type) Output Shape Param #
input_1 (InputLayer) (None, 224, 224, 3) 0
conv1_pad (ZeroPadding2D) (None, 225, 225, 3) 0
conv1 (Conv2D) (None, 112, 112, 32) 864
conv1_bn (BatchNormalization (None, 112, 112, 32) 128
conv1_relu (ReLU) (None, 112, 112, 32) 0
conv_dw_1 (DepthwiseConv2D) (None, 112, 112, 32) 288
conv_dw_1_bn (BatchNormaliza (None, 112, 112, 32) 128
conv_dw_1_relu (ReLU) (None, 112, 112, 32) 0
29°C
Haze
Search
ENG
IN
15:05
18-11-2022


IBM
IBM-Project-24103-1659937745/Proje
AI-B7-1A3E (Evening Session)-Day 14
skin-disese/skin disease classifica
IBM-Project-24103-1659937745/Proje
github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb
Booking.com
Imported From IE
conv_dw_3_relu (ReLU) (None, 56, 56, 128) 0
conv_pw_3 (Conv2D) (None, 56, 56, 128) 16384
conv_pw_3_bn (BatchNormaliza (None, 56, 56, 128) 512
conv_pw_3_relu (ReLU) (None, 56, 56, 128) 0
conv_pad_4 (ZeroPadding2D) (None, 57, 57, 128) 0
conv_dw_4 (DepthwiseConv2D) (None, 28, 28, 128) 1152
conv_dw_4_bn (BatchNormaliza (None, 28, 28, 128) 512
conv_dw_4_relu (ReLU) (None, 28, 28, 128) 0
conv_pw_4 (Conv2D) (None, 28, 28, 256) 32768
conv_pw_4_bn (BatchNormaliza (None, 28, 28, 256) 1024
conv_pw_4_relu (ReLU) (None, 28, 28, 256) 0
conv_dw_5 (DepthwiseConv2D) (None, 28, 28, 256) 2304
conv_dw_5_bn (BatchNormaliza (None, 28, 28, 256) 1024
conv_dw_5_relu (ReLU) (None, 28, 28, 256) 0
conv_pw_5 (Conv2D) (None, 28, 28, 256) 65536
conv_pw_5_bn (BatchNormaliza (None, 28, 28, 256) 1024
conv_pw_5_relu (ReLU) (None, 28, 28, 256) 0
conv_pad_6 (ZeroPadding2D) (None, 29, 29, 256) 0
conv_dw_6 (DepthwiseConv2D) (None, 14, 14, 256) 2304
conv_dw_6_bn (BatchNormaliza (None, 14, 14, 256) 1024
conv_dw_6_relu (ReLU) (None, 14, 14, 256) 0
conv_pw_6 (Conv2D) (None, 14, 14, 512) 131072
29°C
Haze
Search
ENG
IN
15:05
18-11-2022

conv_pw_10_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_11 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_11_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_dw_11_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_11 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_11_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_pw_11_relu (ReLU)	(None, 14, 14, 512)	0
conv_pad_12 (ZeroPadding2D)	(None, 15, 15, 512)	0
conv_dw_12 (DepthwiseConv2D)	(None, 7, 7, 512)	4608
conv_dw_12_bn (BatchNormaliz	(None, 7, 7, 512)	2048
conv_dw_12_relu (ReLU)	(None, 7, 7, 512)	0
conv_pw_12 (Conv2D)	(None, 7, 7, 1024)	524288
conv_pw_12_bn (BatchNormaliz	(None, 7, 7, 1024)	4096
conv_pw_12_relu (ReLU)	(None, 7, 7, 1024)	0
conv_dw_13 (DepthwiseConv2D)	(None, 7, 7, 1024)	9216
conv_dw_13_bn (BatchNormaliz	(None, 7, 7, 1024)	4096
conv_dw_13_relu (ReLU)	(None, 7, 7, 1024)	0
conv_pw_13 (Conv2D)	(None, 7, 7, 1024)	1048576
conv_pw_13_bn (BatchNormaliz	(None, 7, 7, 1024)	4096
conv_pw_13_relu (ReLU)	(None, 7, 7, 1024)	0
global_average_pooling2d (Gl	(None, 1024)	0
dropout (Dropout)	(None, 1024)	0
dense (Dense)	(None, 7)	7175

IBM | IBM-Project-24103-1659937745/Proje | AI-B7-1A3E (Evening Session)-Day 14 | skin-disese/skin disease classifica | IBM-Project-24103-1659937745/Proje
github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb
Booking.com | Imported From IE

Train the Model

In [58]:
```
from tensorflow.keras.metrics import categorical_accuracy, top_k_categorical_accuracy

def top_3_accuracy(y_true, y_pred):
    return top_k_categorical_accuracy(y_true, y_pred, k=3)

def top_2_accuracy(y_true, y_pred):
    return top_k_categorical_accuracy(y_true, y_pred, k=2)
```

In [59]:
```
model.compile(Adam(lr=0.01), loss='categorical_crossentropy',
              metrics=[categorical_accuracy, top_2_accuracy, top_3_accuracy])
```

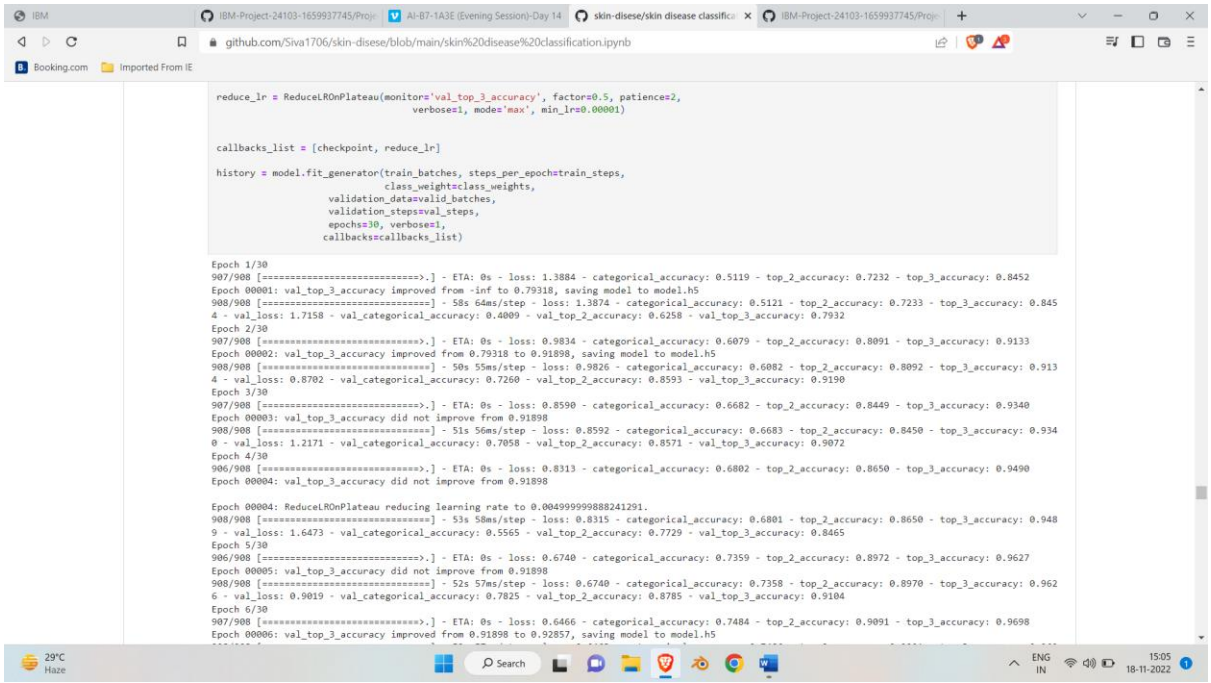
In [60]:
```
print(valid_batches.class_indices)
```

```
{'akiec': 0, 'bcc': 1, 'bkl': 2, 'df': 3, 'mel': 4, 'nv': 5, 'vasc': 6}
```

In [61]:
```
class_weights={
    0: 1.0, # akiec
    1: 1.0, # bcc
    2: 1.0, # bkl
    3: 1.0, # df
    4: 3.0, # mel # Try to make the model more sensitive to Melanoma.
    5: 1.0, # nv
    6: 1.0, # vasc
}
```

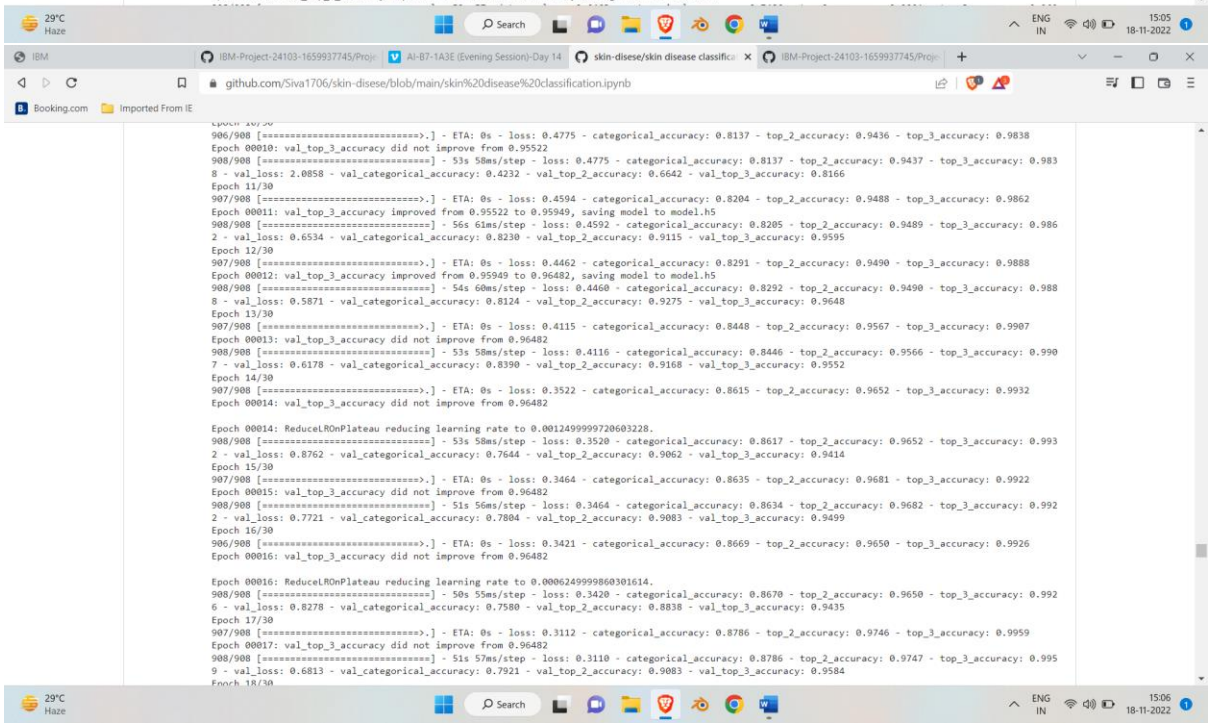
In [62]:
```
filepath = "model.h5"
checkpoint = ModelCheckpoint(filepath, monitor='val_top_3_accuracy', verbose=1,
                             save_best_only=True, mode='max')

reduce_lr = ReduceLROnPlateau(monitor='val_top_3_accuracy', factor=0.5, patience=2,
                              verbose=1, mode='max', min_lr=0.00001)


callbacks_list = [checkpoint, reduce_lr]
```

29°C Haze | Search | ENG IN | 15:05 18-11-2022




```
reduce_lr = ReduceLROnPlateau(monitor='val_top_3_accuracy', factor=0.5, patiences=2,
                              verbose=1, mode='max', min_lr=0.00001)


callbacks_list = [checkpoint, reduce_lr]

history = model.fit_generator(train_batches, steps_per_epoch=train_steps,
                              class_weight=class_weights,
                    validation_data=valid_batches,
                    validation_steps=val_steps,
                    epochs=30, verbose=1,
                   callbacks=callbacks_list)
```

```
Epoch 1/30
907/908 [============================>.] - ETA: 0s - loss: 1.3884 - categorical_accuracy: 0.5119 - top_2_accuracy: 0.7232 - top_3_accuracy: 0.8452
Epoch 00001: val_top_3_accuracy improved from -inf to 0.79318, saving model to model.h5
908/908 [==============================] - 58s 64ms/step - loss: 1.3874 - categorical_accuracy: 0.5121 - top_2_accuracy: 0.7233 - top_3_accuracy: 0.845
4 - val_loss: 1.7158 - val_categorical_accuracy: 0.4009 - val_top_2_accuracy: 0.6258 - val_top_3_accuracy: 0.7932
Epoch 2/30
907/908 [============================>.] - ETA: 0s - loss: 0.9834 - categorical_accuracy: 0.6079 - top_2_accuracy: 0.8091 - top_3_accuracy: 0.9133
Epoch 00002: val_top_3_accuracy improved from 0.79318 to 0.91898, saving model to model.h5
908/908 [==============================] - 50s 55ms/step - loss: 0.9826 - categorical_accuracy: 0.6082 - top_2_accuracy: 0.8092 - top_3_accuracy: 0.913
4 - val_loss: 0.8702 - val_categorical_accuracy: 0.7260 - val_top_2_accuracy: 0.8593 - val_top_3_accuracy: 0.9190
Epoch 3/30
907/908 [============================>.] - ETA: 0s - loss: 0.8590 - categorical_accuracy: 0.6682 - top_2_accuracy: 0.8449 - top_3_accuracy: 0.9340
Epoch 00003: val_top_3_accuracy did not improve from 0.91898
908/908 [==============================] - 51s 56ms/step - loss: 0.8592 - categorical_accuracy: 0.6683 - top_2_accuracy: 0.8450 - top_3_accuracy: 0.934
0 - val_loss: 1.2171 - val_categorical_accuracy: 0.7058 - val_top_2_accuracy: 0.8571 - val_top_3_accuracy: 0.9072
Epoch 4/30
906/908 [============================>.] - ETA: 0s - loss: 0.8313 - categorical_accuracy: 0.6802 - top_2_accuracy: 0.8650 - top_3_accuracy: 0.9490
Epoch 00004: val_top_3_accuracy did not improve from 0.91898

Epoch 00004: ReduceLROnPlateau reducing learning rate to 0.0049999999888241291.
908/908 [==============================] - 53s 58ms/step - loss: 0.8315 - categorical_accuracy: 0.6801 - top_2_accuracy: 0.8650 - top_3_accuracy: 0.948
9 - val_loss: 1.6473 - val_categorical_accuracy: 0.5565 - val_top_2_accuracy: 0.7729 - val_top_3_accuracy: 0.8465
Epoch 5/30
906/908 [============================>.] - ETA: 0s - loss: 0.6740 - categorical_accuracy: 0.7359 - top_2_accuracy: 0.8972 - top_3_accuracy: 0.9627
Epoch 00005: val_top_3_accuracy did not improve from 0.91898
908/908 [==============================] - 52s 57ms/step - loss: 0.6740 - categorical_accuracy: 0.7358 - top_2_accuracy: 0.8970 - top_3_accuracy: 0.962
6 - val_loss: 0.9019 - val_categorical_accuracy: 0.7825 - val_top_2_accuracy: 0.8785 - val_top_3_accuracy: 0.9104
Epoch 6/30
907/908 [============================>.] - ETA: 0s - loss: 0.6466 - categorical_accuracy: 0.7484 - top_2_accuracy: 0.9091 - top_3_accuracy: 0.9698
Epoch 00006: val_top_3_accuracy improved from 0.91898 to 0.92857, saving model to model.h5
```




```
906/908 [============================>.] - ETA: 0s - loss: 0.4775 - categorical_accuracy: 0.8137 - top_2_accuracy: 0.9436 - top_3_accuracy: 0.9838
Epoch 00010: val_top_3_accuracy did not improve from 0.95522
908/908 [==============================] - 53s 58ms/step - loss: 0.4775 - categorical_accuracy: 0.8137 - top_2_accuracy: 0.9437 - top_3_accuracy: 0.983
8 - val_loss: 2.0858 - val_categorical_accuracy: 0.4232 - val_top_2_accuracy: 0.6642 - val_top_3_accuracy: 0.8166
Epoch 11/30
907/908 [============================>.] - ETA: 0s - loss: 0.4594 - categorical_accuracy: 0.8204 - top_2_accuracy: 0.9488 - top_3_accuracy: 0.9862
Epoch 00011: val_top_3_accuracy improved from 0.95522 to 0.95949, saving model to model.h5
908/908 [==============================] - 56s 61ms/step - loss: 0.4592 - categorical_accuracy: 0.8205 - top_2_accuracy: 0.9489 - top_3_accuracy: 0.986
2 - val_loss: 0.6534 - val_categorical_accuracy: 0.8230 - val_top_2_accuracy: 0.9115 - val_top_3_accuracy: 0.9595
Epoch 12/30
907/908 [============================>.] - ETA: 0s - loss: 0.4462 - categorical_accuracy: 0.8291 - top_2_accuracy: 0.9490 - top_3_accuracy: 0.9888
Epoch 00012: val_top_3_accuracy improved from 0.95949 to 0.96482, saving model to model.h5
908/908 [==============================] - 54s 60ms/step - loss: 0.4460 - categorical_accuracy: 0.8292 - top_2_accuracy: 0.9490 - top_3_accuracy: 0.988
8 - val_loss: 0.5871 - val_categorical_accuracy: 0.8124 - val_top_2_accuracy: 0.9275 - val_top_3_accuracy: 0.9648
Epoch 13/30
907/908 [============================>.] - ETA: 0s - loss: 0.4115 - categorical_accuracy: 0.8448 - top_2_accuracy: 0.9567 - top_3_accuracy: 0.9907
Epoch 00013: val_top_3_accuracy did not improve from 0.96482
908/908 [==============================] - 53s 58ms/step - loss: 0.4116 - categorical_accuracy: 0.8446 - top_2_accuracy: 0.9566 - top_3_accuracy: 0.990
7 - val_loss: 0.6178 - val_categorical_accuracy: 0.8390 - val_top_2_accuracy: 0.9168 - val_top_3_accuracy: 0.9552
Epoch 14/30
907/908 [============================>.] - ETA: 0s - loss: 0.3522 - categorical_accuracy: 0.8615 - top_2_accuracy: 0.9652 - top_3_accuracy: 0.9932
Epoch 00014: val_top_3_accuracy did not improve from 0.96482

Epoch 00014: ReduceLROnPlateau reducing learning rate to 0.0012499999720603228.
908/908 [==============================] - 53s 58ms/step - loss: 0.3520 - categorical_accuracy: 0.8617 - top_2_accuracy: 0.9652 - top_3_accuracy: 0.993
2 - val_loss: 0.8762 - val_categorical_accuracy: 0.7644 - val_top_2_accuracy: 0.9062 - val_top_3_accuracy: 0.9414
Epoch 15/30
907/908 [============================>.] - ETA: 0s - loss: 0.3464 - categorical_accuracy: 0.8635 - top_2_accuracy: 0.9681 - top_3_accuracy: 0.9922
Epoch 00015: val_top_3_accuracy did not improve from 0.96482
908/908 [==============================] - 51s 56ms/step - loss: 0.3464 - categorical_accuracy: 0.8634 - top_2_accuracy: 0.9682 - top_3_accuracy: 0.992
2 - val_loss: 0.7721 - val_categorical_accuracy: 0.7804 - val_top_2_accuracy: 0.9083 - val_top_3_accuracy: 0.9499
Epoch 16/30
906/908 [============================>.] - ETA: 0s - loss: 0.3421 - categorical_accuracy: 0.8669 - top_2_accuracy: 0.9650 - top_3_accuracy: 0.9926
Epoch 00016: val_top_3_accuracy did not improve from 0.96482

Epoch 00016: ReduceLROnPlateau reducing learning rate to 0.0006249999860301614.
908/908 [==============================] - 50s 55ms/step - loss: 0.3420 - categorical_accuracy: 0.8670 - top_2_accuracy: 0.9650 - top_3_accuracy: 0.992
6 - val_loss: 0.8278 - val_categorical_accuracy: 0.7580 - val_top_2_accuracy: 0.8838 - val_top_3_accuracy: 0.9435
Epoch 17/30
907/908 [============================>.] - ETA: 0s - loss: 0.3112 - categorical_accuracy: 0.8786 - top_2_accuracy: 0.9746 - top_3_accuracy: 0.9959
Epoch 00017: val_top_3_accuracy did not improve from 0.96482
908/908 [==============================] - 51s 57ms/step - loss: 0.3110 - categorical_accuracy: 0.8786 - top_2_accuracy: 0.9747 - top_3_accuracy: 0.995
9 - val_loss: 0.6813 - val_categorical_accuracy: 0.7921 - val_top_2_accuracy: 0.9083 - val_top_3_accuracy: 0.9584
Epoch 18/30
```

```
Epoch 00018: val_top_3_accuracy did not improve from 0.96482

Epoch 00018: ReduceLROnPlateau reducing learning rate to 0.0003124999930150807.
908/908 [==============================] - 50s 55ms/step - loss: 0.2585 - categorical_accuracy: 0.8996 - top_2_accuracy: 0.9804 - top_3_accuracy: 0.996
8 - val_loss: 0.8121 - val_categorical_accuracy: 0.7708 - val_top_2_accuracy: 0.9019 - val_top_3_accuracy: 0.9563
Epoch 19/30
907/908 [==============================>.] - ETA: 0s - loss: 0.2562 - categorical_accuracy: 0.8997 - top_2_accuracy: 0.9767 - top_3_accuracy: 0.9968
Epoch 00019: val_top_3_accuracy did not improve from 0.96482
908/908 [==============================] - 52s 57ms/step - loss: 0.2563 - categorical_accuracy: 0.8996 - top_2_accuracy: 0.9768 - top_3_accuracy: 0.996
8 - val_loss: 0.8831 - val_categorical_accuracy: 0.7580 - val_top_2_accuracy: 0.8987 - val_top_3_accuracy: 0.9531
Epoch 20/30
907/908 [==============================>.] - ETA: 0s - loss: 0.2612 - categorical_accuracy: 0.8994 - top_2_accuracy: 0.9776 - top_3_accuracy: 0.9966
Epoch 00020: val_top_3_accuracy did not improve from 0.96482

Epoch 00020: ReduceLROnPlateau reducing learning rate to 0.0001562499965075403.
908/908 [==============================] - 52s 57ms/step - loss: 0.2613 - categorical_accuracy: 0.8993 - top_2_accuracy: 0.9776 - top_3_accuracy: 0.996
6 - val_loss: 0.8695 - val_categorical_accuracy: 0.7655 - val_top_2_accuracy: 0.8966 - val_top_3_accuracy: 0.9467
Epoch 21/30
907/908 [==============================>.] - ETA: 0s - loss: 0.2478 - categorical_accuracy: 0.9030 - top_2_accuracy: 0.9808 - top_3_accuracy: 0.9978
Epoch 00021: val_top_3_accuracy did not improve from 0.96482
908/908 [==============================] - 53s 58ms/step - loss: 0.2478 - categorical_accuracy: 0.9029 - top_2_accuracy: 0.9808 - top_3_accuracy: 0.997
8 - val_loss: 0.8246 - val_categorical_accuracy: 0.7644 - val_top_2_accuracy: 0.9019 - val_top_3_accuracy: 0.9510
Epoch 22/30
906/908 [==============================>.] - ETA: 0s - loss: 0.2378 - categorical_accuracy: 0.9075 - top_2_accuracy: 0.9829 - top_3_accuracy: 0.9976
Epoch 00022: val_top_3_accuracy did not improve from 0.96482

Epoch 00022: ReduceLROnPlateau reducing learning rate to 7.812499825377017e-05.
908/908 [==============================] - 52s 57ms/step - loss: 0.2376 - categorical_accuracy: 0.9076 - top_2_accuracy: 0.9829 - top_3_accuracy: 0.997
6 - val_loss: 0.8392 - val_categorical_accuracy: 0.7591 - val_top_2_accuracy: 0.9041 - val_top_3_accuracy: 0.9552
Epoch 23/30
906/908 [==============================>.] - ETA: 0s - loss: 0.2312 - categorical_accuracy: 0.9093 - top_2_accuracy: 0.9826 - top_3_accuracy: 0.9969
Epoch 00023: val_top_3_accuracy did not improve from 0.96482
908/908 [==============================] - 51s 56ms/step - loss: 0.2310 - categorical_accuracy: 0.9094 - top_2_accuracy: 0.9825 - top_3_accuracy: 0.996
9 - val_loss: 0.9009 - val_categorical_accuracy: 0.7580 - val_top_2_accuracy: 0.8945 - val_top_3_accuracy: 0.9435
Epoch 24/30
906/908 [==============================>.] - ETA: 0s - loss: 0.2318 - categorical_accuracy: 0.9091 - top_2_accuracy: 0.9825 - top_3_accuracy: 0.9971
Epoch 00024: val_top_3_accuracy did not improve from 0.96482

Epoch 00024: ReduceLROnPlateau reducing learning rate to 3.9062499126885086e-05.
908/908 [==============================] - 51s 56ms/step - loss: 0.2326 - categorical_accuracy: 0.9088 - top_2_accuracy: 0.9824 - top_3_accuracy: 0.997
0 - val_loss: 0.8637 - val_categorical_accuracy: 0.7612 - val_top_2_accuracy: 0.8955 - val_top_3_accuracy: 0.9542
Epoch 25/30
907/908 [==============================>.] - ETA: 0s - loss: 0.2384 - categorical_accuracy: 0.9107 - top_2_accuracy: 0.9830 - top_3_accuracy: 0.9967
Epoch 00025: val_top_3_accuracy did not improve from 0.96482
```


IBM
IBM-Project-24103-1659937745/Proje
AI-B7-1A3E (Evening Session)-Day 14
skin-disese/skin disease classifica
IBM-Project-24103-1659937745/Proje
github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb
Booking.com
Imported From IE

Evaluate the model using the val set

In [63]:
```
model.metrics_names
```

Out[63]:
```
['loss', 'categorical_accuracy', 'top_2_accuracy', 'top_3_accuracy']
```

In [64]:
```
val_loss, val_cat_acc, val_top_2_acc, val_top_3_acc = \
model.evaluate_generator(test_batches,
                         steps=len(df_val))

print('val_loss:', val_loss)
print('val_cat_acc:', val_cat_acc)
print('val_top_2_acc:', val_top_2_acc)
print('val_top_3_acc:', val_top_3_acc)
```

```
val_loss: 0.8684576310404912
val_cat_acc: 0.7601279317697228
val_top_2_acc: 0.8955223880597015
val_top_3_acc: 0.9509594882729211
```

In [65]:
```
model.load_weights('model.h5')

val_loss, val_cat_acc, val_top_2_acc, val_top_3_acc = \
model.evaluate_generator(test_batches,
                         steps=len(df_val))

print('val_loss:', val_loss)
print('val_cat_acc:', val_cat_acc)
print('val_top_2_acc:', val_top_2_acc)
print('val_top_3_acc:', val_top_3_acc)
```

```
val_loss: 0.587132240817163
val_cat_acc: 0.8123667377398721
val_top_2_acc: 0.9275053304904051
val_top_3_acc: 0.964818763326226
```

Plot the Training Curves

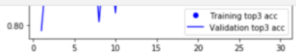
In [66]:
```
import matplotlib.pyplot as plt
```

29°C Haze
Search
ENG IN
15:06 18-11-2022

Plot the Training Curves

In [66]:

```
import matplotlib.pyplot as plt

acc = history.history['categorical_accuracy']
val_acc = history.history['val_categorical_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
train_top2_acc = history.history['top_2_accuracy']
val_top2_acc = history.history['val_top_2_accuracy']
train_top3_acc = history.history['top_3_accuracy']
val_top3_acc = history.history['val_top_3_accuracy']
epochs = range(1, len(acc) + 1)

plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.figure()

plt.plot(epochs, acc, 'bo', label='Training cat acc')
plt.plot(epochs, val_acc, 'b', label='Validation cat acc')
plt.title('Training and validation cat accuracy')
plt.legend()
plt.figure()


plt.plot(epochs, train_top2_acc, 'bo', label='Training top2 acc')
plt.plot(epochs, val_top2_acc, 'b', label='Validation top2 acc')
plt.title('Training and validation top2 accuracy')
plt.legend()
plt.figure()
plt.plot(epochs, train_top3_acc, 'bo', label='Training top3 acc')
plt.plot(epochs, val_top3_acc, 'b', label='Validation top3 acc')
plt.title('Training and validation top3 accuracy')
plt.legend()


plt.show()
```



Create a Confusion Matrix

In [67]:

```
test_labels = test_batches.classes
```

In [68]:

```
test_labels
```

Out[68]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
       3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6], dtype=int32)
```

In [69]:
```
test_batches.class_indices
```

Out[69]:
```
{'akiec': 0, 'bcc': 1, 'bkl': 2, 'df': 3, 'mel': 4, 'nv': 5, 'vasc': 6}
```

In [70]:
```
predictions = model.predict_generator(test_batches, steps=len(df_val), verbose=1)
```

```
938/938 [==============================] - 13s 14ms/step
```

In [71]:
```
predictions.shape
```

Out[71]:
```
(938, 7)
```

In [72]:
```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
```



```
    plt.xlabel('Predicted label')
    plt.tight_layout()
```

In [73]:
```
test_labels.shape
```

Out[73]:
```
(938,)
```

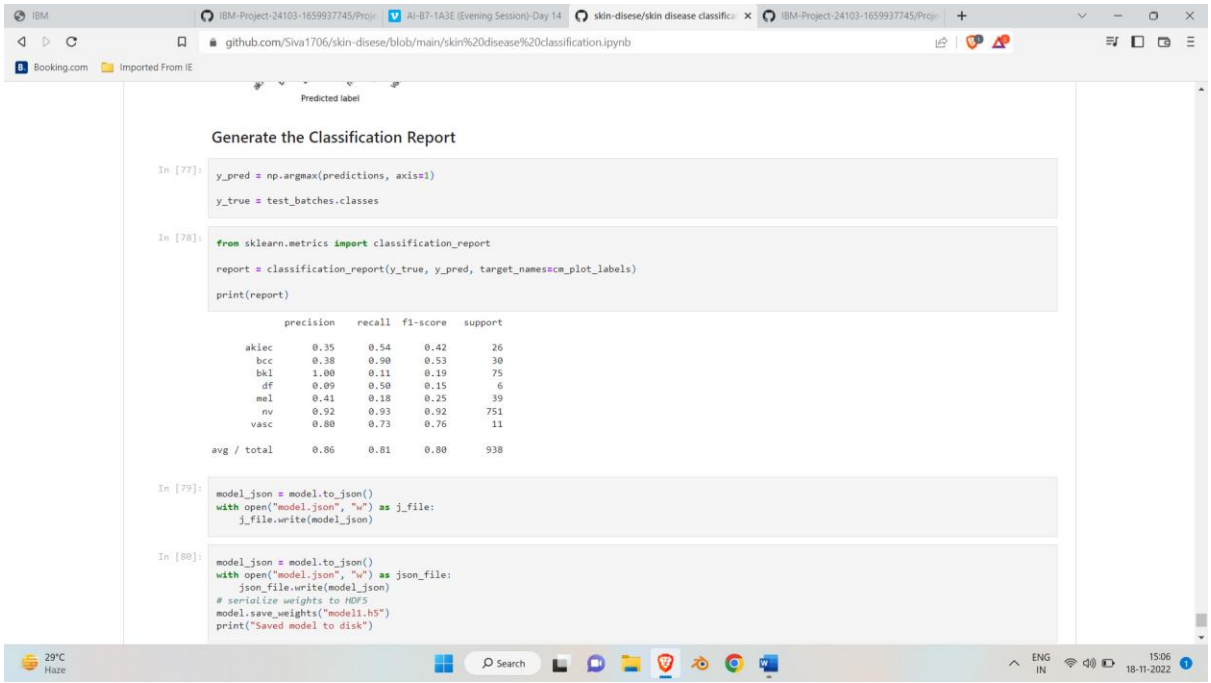
In [74]:
```
cm = confusion_matrix(test_labels, predictions.argmax(axis=1))
```

In [75]:
```
test_batches.class_indices
```

Out[75]:
```
{'akiec': 0, 'bcc': 1, 'bkl': 2, 'df': 3, 'mel': 4, 'nv': 5, 'vasc': 6}
```

In [76]:
```
cm_plot_labels = ['akiec', 'bcc', 'bkl', 'df', 'mel','nv', 'vasc']

plot_confusion_matrix(cm, cm_plot_labels, title='Confusion Matrix')
```

```
Confusion matrix, without normalization
[[ 14   7   0   3   0   2   0]
 [  1  27   0   1   0   1   0]
 [ 18  10   8   2   5  32   0]
 [  0   0   0   3   0   3   0]
 [  3   6   0   1   7  21   1]
 [  4  21   0  25   5 695   1]
 [  0   0   0   0   0   3   8]]
```


IBM
IBM-Project-24103-1659937745/Proje
AI-B7-1A3E (Evening Session)-Day 14
skin-disese/skin disease classifica
IBM-Project-24103-1659937745/Proje
github.com/Siva1706/skin-disese/blob/main/skin%20disease%20classification.ipynb
Booking.com
Imported From IE
Predicted label
Generate the Classification Report
In [77]:
y_pred = np.argmax(predictions, axis=1)
y_true = test_batches.classes
In [78]:
from sklearn.metrics import classification_report
report = classification_report(y_true, y_pred, target_names=cm_plot_labels)
print(report)
precision recall f1-score support
akiec 0.35 0.54 0.42 26
bcc 0.38 0.90 0.53 30
bkl 1.00 0.11 0.19 75
df 0.09 0.50 0.15 6
mel 0.41 0.18 0.25 39
nv 0.92 0.93 0.92 751
vasc 0.80 0.73 0.76 11
avg / total 0.86 0.81 0.80 938
In [79]:
model_json = model.to_json()
with open("model.json", "w") as j_file:
j_file.write(model_json)
In [80]:
model_json = model.to_json()
with open("model.json", "w") as json_file:
json_file.write(model_json)
# serialize weights to HDF5
model.save_weights("model1.h5")
print("Saved model to disk")
29°C
Haze
Search
ENG
IN
15:06
18-11-2022