

## Train Images :

The screenshot displays a Jupyter Notebook interface with two visible code cells. The top cell, titled "Create a train set that excludes images that are in the val set", contains Python code for identifying and separating training and validation data. The bottom cell, titled "Transfer the Images into the Folders", contains code for moving image files into specific directories based on their labels.

**Create a train set that excludes images that are in the val set**

```
In [11]: def identify_val_rows(x):
val_list = list(df_val['image_id'])

if str(x) in val_list:
    return 'val'
else:
    return 'train'

df_data['train_or_val'] = df_data['image_id'].apply(identify_val_rows)
df_data['train_or_val'] = df_data['train_or_val'].apply(identify_val_rows)

df_train = df_data[df_data['train_or_val'] != 'train']

print(len(df_train))
print(len(df_val))

9077
938

In [12]: df_train['dx'].value_counts()

Out[12]:
nv      5954
mel     1074
bkl     1024
bcc       484
akiec    301
vasc     131
df        109
Name: dx, dtype: int64

In [13]: df_val['dx'].value_counts()

Out[13]:
nv       751
bkl       75
mel       39
bcc        30
```

**Transfer the Images into the Folders**

```
In [14]: df_data.set_index('image_id', inplace=True)

In [15]: folder_1 = os.listdir('../input/ham10000_images_part_1')
folder_2 = os.listdir('../input/ham10000_images_part_2')

train_list = list(df_train['image_id'])
val_list = list(df_val['image_id'])

for image in train_list:
    fname = image + '.jpg'
    label = df_data.loc[image, 'dx']

    if fname in folder_1:
        src = os.path.join('../input/ham10000_images_part_1', fname)
        dst = os.path.join(train_dir, label, fname)
        shutil.copyfile(src, dst)

    if fname in folder_2:
        src = os.path.join('../input/ham10000_images_part_2', fname)
        dst = os.path.join(train_dir, label, fname)
        shutil.copyfile(src, dst)

for image in val_list:
    fname = image + '.jpg'
    label = df_data.loc[image, 'dx']

    if fname in folder_1:
        src = os.path.join('../input/ham10000_images_part_1', fname)
        dst = os.path.join(val_dir, label, fname)
        shutil.copyfile(src, dst)

    if fname in folder_2:
        src = os.path.join('../input/ham10000_images_part_2', fname)
        dst = os.path.join(val_dir, label, fname)
```

Copy the train images into aug\_dir

```
In [18]: class_list = ['mel', 'bkl', 'bcc', 'akiec', 'vasc', 'df']

for item in class_list:
    aug_dir = 'aug_dir'
    os.mkdir(aug_dir)
    img_dir = os.path.join(aug_dir, 'img_dir')
    os.mkdir(img_dir)

    img_class = item

    img_list = os.listdir('base_dir/train_dir/' + img_class)

    for fname in img_list:
        src = os.path.join('base_dir/train_dir/' + img_class, fname)
        dst = os.path.join(img_dir, fname)
        shutil.copyfile(src, dst)

    path = aug_dir
    save_path = 'base_dir/train_dir/' + img_class

    datagen = ImageDataGenerator(
        rotation_range=180,
        width_shift_range=0.1,
        height_shift_range=0.1,
        zoom_range=0.1,
        horizontal_flip=True,
        vertical_flip=True,
        #brightness_range=(0.9, 1.1),
        fill_mode='nearest')

    batch_size = 50

    aug_datagen = datagen.flow_from_directory(path,
                                              save_to_dir=save_path,
                                              save_format='jpg',
                                              target_size=(224, 224),
                                              batch_size=batch_size)
```

```
num_files = len(os.listdir(img_dir))
num_batches = int(np.ceil((num_aug_images_wanted-num_files)/batch_size))

for i in range(0,num_batches):
    imgs, labels = next(aug_datagen)
    shutil.rmtree('aug_dir')
```

Found 1074 images belonging to 1 classes.  
Found 1024 images belonging to 1 classes.  
Found 484 images belonging to 1 classes.  
Found 301 images belonging to 1 classes.  
Found 131 images belonging to 1 classes.  
Found 109 images belonging to 1 classes.

```
In [19]: print(len(os.listdir('base_dir/train_dir/nv')))
print(len(os.listdir('base_dir/train_dir/mel')))
print(len(os.listdir('base_dir/train_dir/bkl')))
print(len(os.listdir('base_dir/train_dir/bcc')))
print(len(os.listdir('base_dir/train_dir/akiec')))
print(len(os.listdir('base_dir/train_dir/vasc')))
print(len(os.listdir('base_dir/train_dir/df')))

5954
5920
5920
5958
5217
5290
4410
```

```
In [20]: print(len(os.listdir('base_dir/val_dir/nv')))
print(len(os.listdir('base_dir/val_dir/mel')))
print(len(os.listdir('base_dir/val_dir/bkl')))
print(len(os.listdir('base_dir/val_dir/bcc')))
print(len(os.listdir('base_dir/val_dir/akiec')))
print(len(os.listdir('base_dir/val_dir/vasc')))
print(len(os.listdir('base_dir/val_dir/df')))

751
20
```

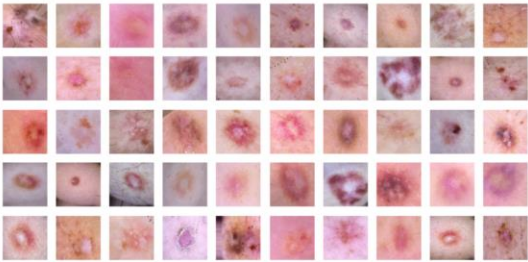
IBMProject-24103-1659937745/ProjAI-B7-1A3E (Evening Session)-Day 14skin-disease/skin disease classificIBMProject-24103-1659937745/Proj

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


Booking.comImported From IE

### Visualize 50 augmented images

```
In [21]: def plots(imgs, figsize=(12,6), rows=5, interp=False, titles=None): # 12,6
if type(imgs[0]) is np.ndarray:
    imgs = np.array(imgs).astype(np.uint8)
    if (imgs.shape[-1] != 3):
        imgs = imgs.transpose((0,2,3,1))
f = plt.figure(figsize=figsize)
cols = len(imgs)//rows if len(imgs) % 2 == 0 else len(imgs)//rows + 1
for i in range(len(imgs)):
    sp = f.add_subplot(rows, cols, i+1)
    sp.axis('Off')
    if titles is not None:
        sp.set_title(titles[i], fontsize=16)
    plt.imshow(imgs[i], interpolation=None if interp else 'none')
plots(imgs, titles=None)
```



29°C  
Haze



ENG  
IN

15:02  
18-11-2022