Connecting the Google Drive to Google Colab

The data we are using is already uploaded in the google drive. We need to use the data into notebook to our drive so that we can access the data.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
    Drive already mounted at /content/drive; to attempt to forcibly remount,
```

Import **os** so that we can use commands just like we use in a terminal in colab.

IMAGE PREPROCESSING USING SKIMAGE AND OPENC

In this first part of the project we will be using SKimage, OpenCV and Matplotlib to get some
modeling.

DataSet that We will be using:

Facial emotion recognition is the process of detecting human emotions from facial express
between human beings, even they don't say a word. The human brain recognizes emotions a
developed that can recognize emotions as well.

The dataset we will be using is the one from kaggle, by Jonathan Oheix and here's the link fo
total and contains images for different seven expressions which are labeled as:

1. Angry
2. Disgust
3. Fear
4. Happy
5. Neutral
6. Sad
7. Surprise

And we will be classifying the images of human faces among these 7 categories mentioned

```
import os
os.chdir('/content/drive/My Drive/F
```

listdir command is used to get list of all the files and folders present in the current working
path specify.

```
os.listdir()
```

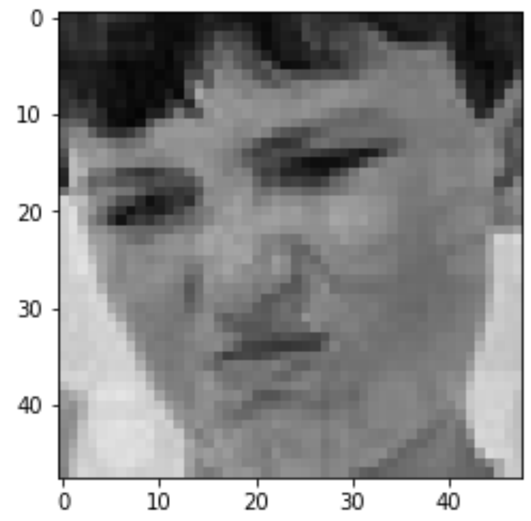
```
['angry',
 'surprise',
 'happy',
 'fear',
 'neutral',
 'sad',
 'disgust',
 'haarcascade_frontalface_alt.xml']
```

```
#IMPORTING REQUIRED LIBERARIES

import cv2
import numpy as np
import matplotlib.pyplot as plt
import skimage
import skimage.transfor
```

```
FUNCTION TO SHOW IMAGE

 image_show(img_path):
mg = cv2.imread(img_path)
lt.imshow(img, cmap='gray')
lt.show()
rint(f'The shape of the      is {i
image_show('/content/dr[illegible]rive
```



```
The shape of the image is (48, 48, 3)
```

PREPROCESS

Let's define a function to preprocess the image before passing it to the model. The processi

1. READING THE IMAGE: We have used imread function of openCV for this.
2. Converting the RGb image to the grayscale image.
3. Resizing the image to a new size.

The function at last return the processed image.

```
## PREPROCESSING IMAGES

def preprocess(img_path):
  img = cv2.imread(img_path)
  grey = cv2.cvtColor(img, cv2.COL
  grey = skimage.transform.resize(
  return grey
```

LOAD THE IMAGES

We are loading 500 images of eaxh of the expressions and labeling them as numbers 0 to 6

```
x = []   # image array
y = []   # target labels

for exp in os.listdir():
  count = 0  ## STORES THE COUNT OI
  print(f'Loading images for: {exp
  if(exp=='angry'):
    label=0
  elif(exp=='surprise'):
    label=1
  elif(exp=='happy'):
    label=2
  elif(exp=='fear'):
    label=3
  elif(exp=='neutral'):
    label=4
  elif(exp=='sad'):
    label=5
  else:
    label=6

  try:
    for img in os.listdir('./'+exp
      im = preprocess('./'+exp+'/'
      im = np.asarray(im)
      x.append(im)
      y.append(label)
      count+=1
      if(count==500):
        break  #MAKE SURE THAT WE I
  except:
    continue

x = np.asarray(x)
y = np.asarray(y)

print(x.shape)
print(y.shape)
```

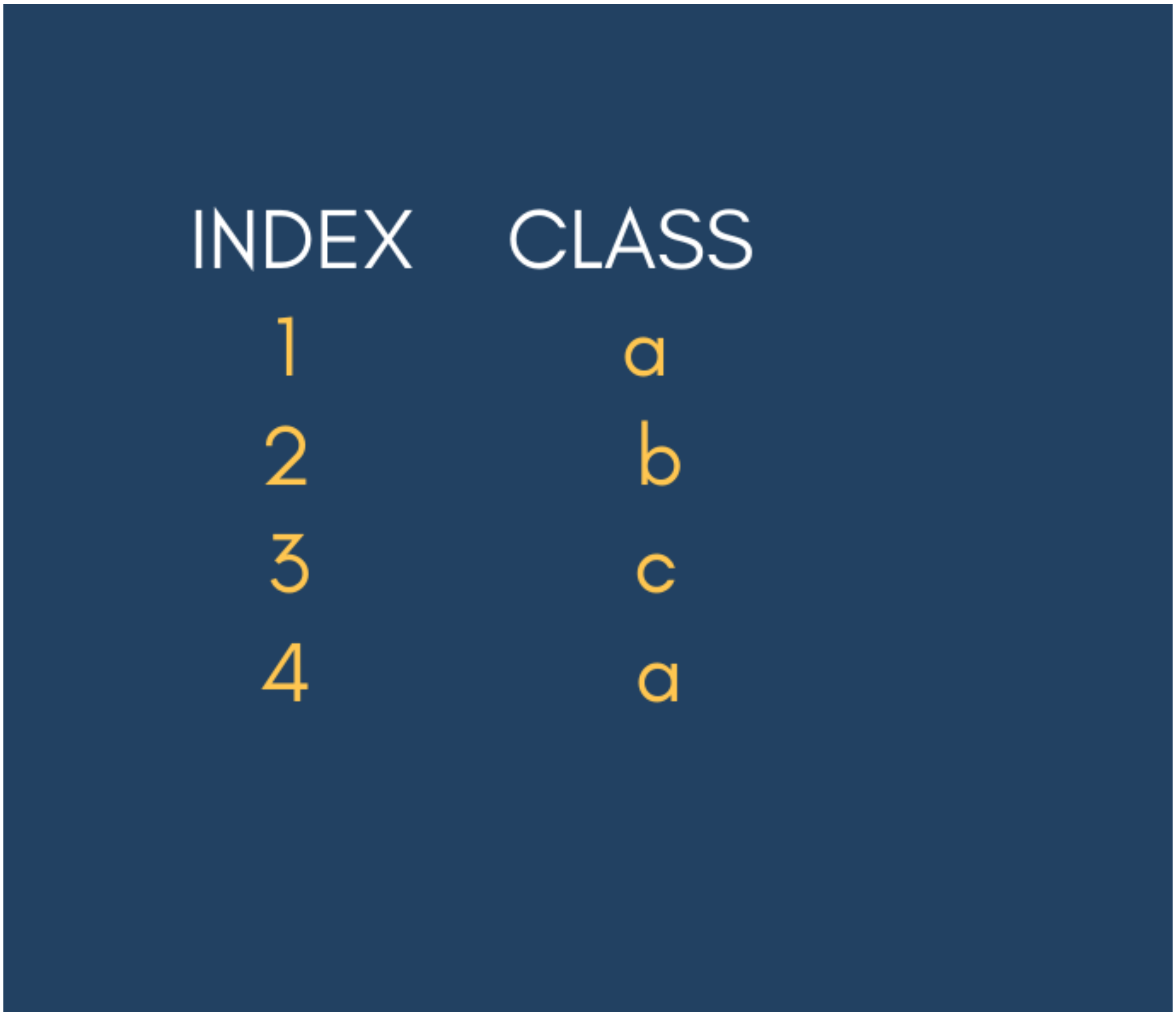
```
Loading images for: angry
Loading images for: surprise
Loading images for: happy
Loading images for: fear
```

Dummy Variables

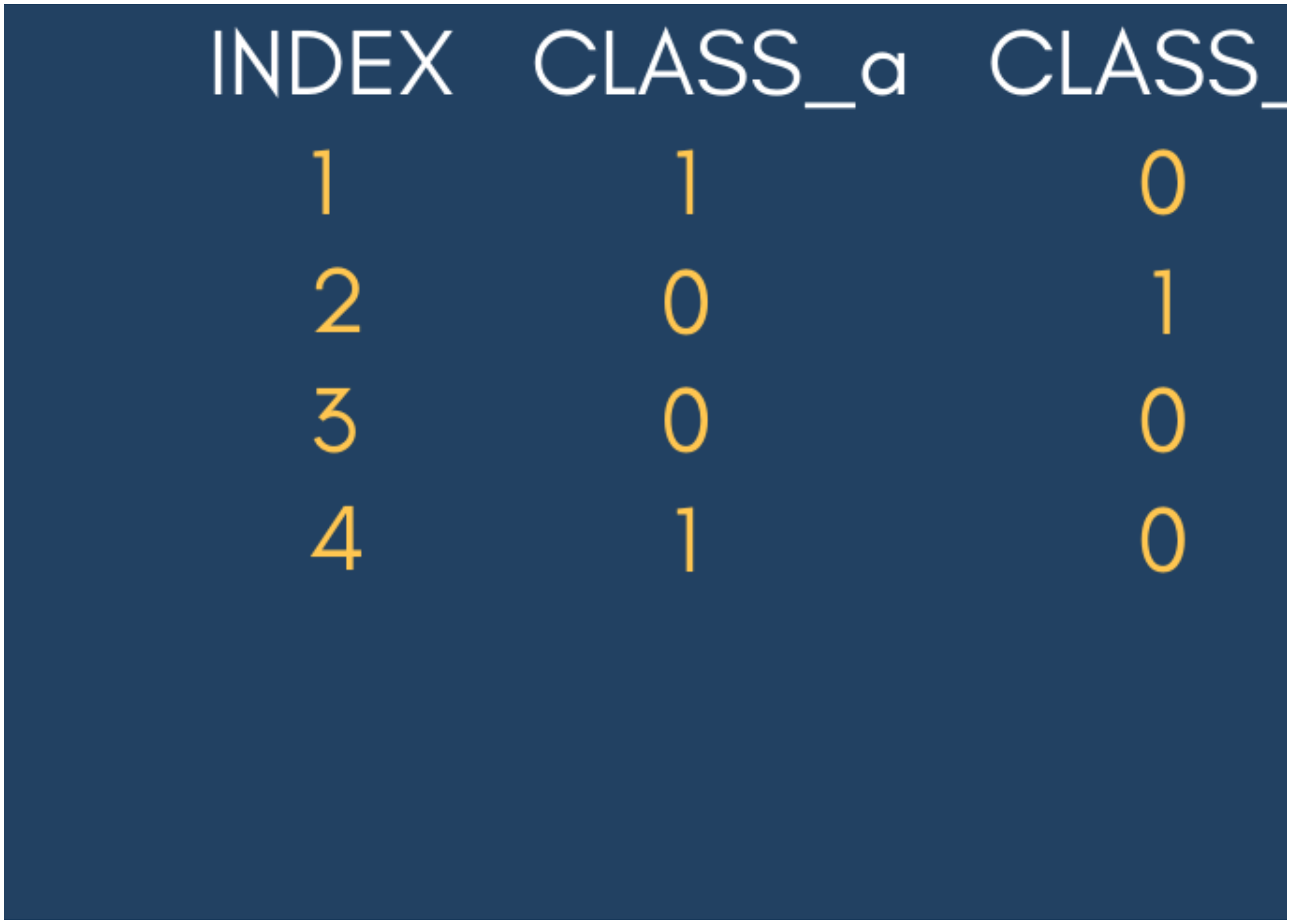
> Columns like season, weathersit, mnth, hr, weekday contains finite discrete values whic
> need a way to represent these values such that the values remain independent of each

The best way is to make a seperate column for each of the input value and represent values
input is that column or not.

Let's see by an example what it means:



INDEX	CLASS
1	a
2	b
3	c
4	a



INDEX	CLASS_a	CLASS_
1	1	0
2	0	1
3	0	0
4	1	0

```
from keras.utils import np_utils
from sklearn.utils import shuffle

y = np_utils.to_categorical(y,num_
x,y= shuffle(x,y,random_state=13)

print(y)
```

```
     [[0. 0. 0. ... 0. 0. 1.]
      [0. 0. 0. ... 1. 0. 0.]
      [0. 0. 1. ... 0. 0. 0.]
      ...
      [1. 0. 0. ... 0. 0. 0.]
      [0. 0. 0. ... 1. 0. 0.]
      [1. 0. 0. ... 0. 0. 0.]]
```

```
print(y.shape)
```

```
    (3436, 7)
```

Time to build the network

TensorFlow is a free and open-source software library for dataflow and differentiable progra
math library, and is also used for machine learning applications such as neural networks.

```
import tensorflow as tf
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense, Con
```

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(2
model.add(MaxPool2D(pool_size=(2, 2
model.add(Conv2D(64, kernel_size=(2
model.add(MaxPool2D(pool_size=(2, 2
model.add(Conv2D(128, kernel_size=(
model.add(Flatten())
model.add(Dense(128))
model.add(Dropout(0.2))
model.add(Dense(64))
model.add(Dense(7, activation='soft
```