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

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/conten
```

Importing Data

```
import pandas as pd

path = '/content/drive/My Drive/Major_project/IMPLEMENTATION1/'
path_s='/content/drive/My Drive/Major_project/IMPLEMENTATION1/sample/'


trainLabels = pd.read_csv(path + "trainLabels.csv")
trainLabels.head()
```

	image	level
0	10_left	0
1	10_right	0
2	13_left	0
3	13_right	0
4	15_left	1

```
!ls '/content/drive/My Drive/Major_project/IMPLEMENTATION1/sample'
```

```
10003_left.jpeg   10162_left.jpeg   1162_right.jpeg  1563_left.jpeg
10003_right.jpeg  10162_right.jpeg  1165_left.jpeg   1563_right.jpeg
10007_left.jpeg   10163_left.jpeg   1165_right.jpeg  1569_left.jpeg
10007_right.jpeg  10163_right.jpeg  1170_left.jpeg   1569_right.jpeg
10009_left.jpeg   10166_left.jpeg   1170_right.jpeg  1574_left.jpeg
```

```
10009_right.jpeg  10166_right.jpeg  1178_left.jpeg   1574_right.jpeg
1000_left.jpeg    10169_left.jpeg   117_left.jpeg    1579_left.jpeg
1000_right.jpeg   10169_right.jpeg  117_right.jpeg   1579_right.jpeg
10010_left.jpeg   10170_left.jpeg   1192_left.jpeg   1586_left.jpeg
10010_right.jpeg  10170_right.jpeg  1192_right.jpeg  1586_right.jpeg
10013_left.jpeg   10173_left.jpeg   1196_left.jpeg   1588_left.jpeg
10013_right.jpeg  10173_right.jpeg  1196_right.jpeg  1588_right.jpeg
10014_left.jpeg   10175_left.jpeg   1197_left.jpeg   1589_left.jpeg
10014_right.jpeg  10175_right.jpeg  1197_right.jpeg  1589_right.jpeg
10015_left.jpeg   10177_left.jpeg   1199_left.jpeg   1592_left.jpeg
10015_right.jpeg  10177_right.jpeg  1199_right.jpeg  1592_right.jpeg
10017_left.jpeg   1017_left.jpeg    1206_left.jpeg   1595_left.jpeg
10017_right.jpeg  1017_right.jpeg   1206_right.jpeg  1595_right.jpeg
10022_left.jpeg   10181_left.jpeg   1211_left.jpeg   159_left.jpeg
10022_right.jpeg  10181_right.jpeg  1211_right.jpeg  159_right.jpeg
10028_left.jpeg   10182_left.jpeg   1226_left.jpeg   15_left.jpeg
10028_right.jpeg  10182_right.jpeg  1226_right.jpeg  15_right.jpeg
10029_left.jpeg   10183_left.jpeg   1228_left.jpeg   1600_left.jpeg
10029_right.jpeg  10183_right.jpeg  1228_right.jpeg  1600_right.jpeg
1002_left.jpeg    10184_left.jpeg   122_left.jpeg    1609_left.jpeg
1002_right.jpeg   10184_right.jpeg  122_right.jpeg   1609_right.jpeg
10030_left.jpeg   10187_left.jpeg   1267_left.jpeg   1611_left.jpeg
10030_right.jpeg  10187_right.jpeg  1267_right.jpeg  1611_right.jpeg
10031_left.jpeg   10190_left.jpeg   1269_left.jpeg   161_left.jpeg
10031_right.jpeg  10190_right.jpeg  1269_right.jpeg  161_right.jpeg
10032_left.jpeg   10193_left.jpeg   1289_left.jpeg   1634_left.jpeg
10032_right.jpeg  10193_right.jpeg  1289_right.jpeg  1634_right.jpeg
10035_left.jpeg   10194_left.jpeg   1290_left.jpeg   1638_left.jpeg
10035_right.jpeg  10194_right.jpeg  1290_right.jpeg  1638_right.jpeg
10042_left.jpeg   10199_left.jpeg   1296_left.jpeg   163_left.jpeg
10042_right.jpeg  10199_right.jpeg  1296_right.jpeg  163_right.jpeg
10043_left.jpeg   10206_left.jpeg   1297_left.jpeg   1640_left.jpeg
10043_right.jpeg  10206_right.jpeg  1297_right.jpeg  1640_right.jpeg
10046_left.jpeg   10207_left.jpeg   1307_left.jpeg   1643_left.jpeg
10046_right.jpeg  10207_right.jpeg  1307_right.jpeg  1643_right.jpeg
10047_left.jpeg   1020_left.jpeg    1309_left.jpeg   1649_left.jpeg
10047_right.jpeg  1020_right.jpeg   1309_right.jpeg  1649_right.jpeg
10048_left.jpeg   10210_left.jpeg   1318_left.jpeg   1655_left.jpeg
10048_right.jpeg  10210_right.jpeg  1318_right.jpeg  1655_right.jpeg
10050_left.jpeg   10212_left.jpeg   1330_left.jpeg   1659_left.jpeg
10050_right.jpeg  10212_right.jpeg  1330_right.jpeg  1659_right.jpeg
10053_left.jpeg   10218_left.jpeg   1334_left.jpeg   1663_left.jpeg
10053_right.jpeg  10218_right.jpeg  1334_right.jpeg  1663_right.jpeg
10058_left.jpeg   1021_left.jpeg    1346_left.jpeg   1666_left.jpeg
10058_right.jpeg  1021_right.jpeg   1346_right.jpeg  1666_right.jpeg
10059_left.jpeg   10220_left.jpeg   1347_left.jpeg   1675_left.jpeg
10059_right.jpeg  10220_right.jpeg  1347_right.jpeg  1675_right.jpeg
10061_left.jpeg   10221_left.jpeg   1349_left.jpeg   167_left.jpeg
10061_right.jpeg  10221_right.jpeg  1349_right.jpeg  167_right.jpeg
10065_left.jpeg   10223_left.jpeg   1351_left.jpeg   1680_left.jpeg
10065_right.jpeg  10223_right.jpeg  1351_right.jpeg  1680_right.jpeg
10069_left.jpeg   10224_left.jpeg   1357_left.jpeg   169_left.jpeg
10069_right.jpeg  10224_right.jpeg  1357_right.jpeg  169_right.jpeg
10073_left.jpeg   10226_left.jpeg   1362_left.jpeg   1706_left.jpeg
10073_right.jpeg  10226_right.jpeg  1362_right.jpeg  1706_right.jpeg
10078_left.jpeg   10230_left.jpeg   1363_left.jpeg   1710_right.jpeg
10078_right.jpeg  10230_right.jpeg  1363_right.jpeg  1713_left.jpeg
10081_left.jpeg   10232_left.jpeg   1369_left.jpeg   1713_right.jpeg
10081_right.jpeg  10232_right.jpeg  1369_right.jpeg  1718_left.jpeg
10085_left.jpeg   10233_left.jpeg   1370_left.jpeg   1718_right.jpeg
10085_right.jpeg  10233_right.jpeg  1370_right.jpeg  1722_left.jpeg
1008_left.jpeg    10234_left.jpeg   1374_left.jpeg   1732_left.jpeg
1008_right.jpeg   10234_right.jpeg  1374_right.jpeg  1732_right.jpeg
10092_left.jpeg   1024_left.jpeg    1377_left.jpeg   1737_left.jpeg
10092_right.jpeg  1024_right.jpeg   1377_right.jpeg  1737_right.jpeg
10094_left.jpeg   1027_left.jpeg    1378_left.jpeg   1743_left.jpeg
10094_right.jpeg  1027_right.jpeg   1378_right.jpeg  1743_right.jpeg
10095_left.jpeg   1029_left.jpeg    1396_left.jpeg   1749_left.jpeg
10095_right.jpeg  1029_right.jpeg   1396_right.jpeg  1749_right.jpeg
10099_left.jpeg   102_left.jpeg     13_left.jpeg     1751_left.jpeg
10099_right.jpeg  102_right.jpeg    13_right.jpeg    1751_right.jpeg
100_left.jpeg     1030_left.jpeg    1407_right.jpeg  1754_left.jpeg
100_right.jpeg    1030_right.jpeg   140_left.jpeg    1754_right.jpeg
```

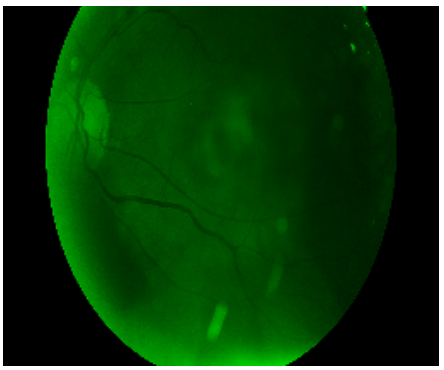
```
10100_left.jpeg   1032_left.jpeg    140_right.jpeg   1756_left.jpeg
10100_right.jpeg  1032_right.jpeg   1412_right.jpeg  1756_right.jpeg
10104_left.jpeg   1034_left.jpeg    1419_left.jpeg   1775_left.jpeg
10104_right.jpeg  1034_right.jpeg   1419_right.jpeg  1775_right.jpeg
10109_left.jpeg   1035_left.jpeg    1425_left.jpeg   1776_left.jpeg
10109_right.jpeg  1035_right.jpeg   1425_right.jpeg  1776_right.jpeg
10112_left.jpeg   1036_left.jpeg    1429_left.jpeg   177_left.jpeg
10112_right.jpeg  1036_right.jpeg   1429_right.jpeg  1785_left.jpeg
10115_left.jpeg   1037_left.jpeg    1430_left.jpeg   1785_right.jpeg
10115_right.jpeg  1037_right.jpeg   1430_right.jpeg  178_left.jpeg
10116_left.jpeg   1040_left.jpeg    1431_left.jpeg   178_right.jpeg
10116_right.jpeg  1040_right.jpeg   1431_right.jpeg  1801_left.jpeg
1011_left.jpeg    1041_left.jpeg    1454_left.jpeg   1801_right.jpeg
1011_right.jpeg   1041_right.jpeg   1454_right.jpeg  1802_left.jpeg
10120_left.jpeg   1042_left.jpeg    1475_left.jpeg   1802_right.jpeg
10120_right.jpeg  1042_right.jpeg   1475_right.jpeg  1810_left.jpeg
10124_left.jpeg   1044_left.jpeg    1480_left.jpeg   1810_right.jpeg
10124_right.jpeg  1044_right.jpeg   1480_right.jpeg  1817_left.jpeg
10125_left.jpeg   104_left.jpeg     1481_left.jpeg   1817_right.jpeg
10125_right.jpeg  104_right.jpeg    1481_right.jpeg  1831_left.jpeg
10126_left.jpeg   1050_left.jpeg    1488_left.jpeg   1831_right.jpeg
10126_right.jpeg  1050_right.jpeg   1488_right.jpeg  1832_left.jpeg
10129_left.jpeg   1051_left.jpeg    1492_left.jpeg   1832_right.jpeg
10129_right.jpeg  1051_right.jpeg   1492_right.jpeg  184_left.jpeg
1012_left.jpeg    1055_left.jpeg    1495_left.jpeg   184_right.jpeg
1012_right.jpeg   1055_right.jpeg   1495_right.jpeg  1853_left.jpeg
10130_left.jpeg   1058_left.jpeg    1496_left.jpeg   1853_right.jpeg
10130_right.jpeg  1058_right.jpeg   1496_right.jpeg  1856_left.jpeg
10131_left.jpeg   1059_left.jpeg    1499_left.jpeg   1856_right.jpeg
10131_right.jpeg  1059_right.jpeg   1499_right.jpeg  1862_left.jpeg
10134_left.jpeg   1061_left.jpeg    149_left.jpeg    1862_right.jpeg
10134_right.jpeg  1061_right.jpeg   149_right.jpeg   1869_left.jpeg
10135_left.jpeg   1088_left.jpeg    1502_left.jpeg   1869_right.jpeg
10135_right.jpeg  1088_right.jpeg   1502_right.jpeg  1878_left.jpeg
10137_left.jpeg   1092_left.jpeg    1505_left.jpeg   1878_right.jpeg
```

```
from PIL import Image
from keras.preprocessing import image
import os
import numpy as np

# resize the image to (256, 256)
img_rows, img_cols = 256, 256

listing = os.listdir(path_s)
#listing.remove("trainLabels.csv")
#listing.remove("sampleSubmission.csv")
#listing.remove("sample_data")

immatrix = []
imlabel = []

for file in listing:
    base = os.path.basename(path_s+file)
    fileName = os.path.splitext(base)[0]
    imlabel.append(trainLabels.loc[trainLabels.image==fileName, 'level'].values[0])
    im = Image.open(path_s+file)
    img = np.array(im.resize((img_rows,img_cols)))

    # convert to green channel only #(Because human eye is more sensitive to green than other colors)
    img[:,:,[0,2]] = 0
    immatrix.append(img)
```

```
Using TensorFlow backend.
```

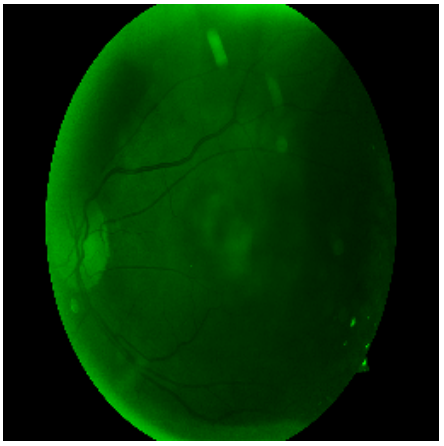
```
print("level:",imlabel[1])
im
```

```
level: 0
```



```
import random

# define transformation methods
def horizontal_flip(image_array):
    return image_array[:, ::-1]

def vertical_flip(image_array):
    return image_array[::-1,:]

def random_transform(image_array):
    if random.random() < 0.5:
        return vertical_flip(image_array)
    else:
        return horizontal_flip(image_array)
```

```
im = Image.fromarray(vertical_flip(immatrix[1]),'RGB')
im
```



```
length = len(immatrix)
for i in range(length):
    if random.random() < 0.1:
        immatrix.append(random_transform(immatrix[i]))
        imlabel.append(imlabel[i])

print("Size of image array before augmentation: ", length)
print("Size fo image array after augmentation: ", len(immatrix))
```

```
Size of image array before augmentation:  549
Size fo image array after augmentation:  603
```

```
from sklearn.utils import shuffle


data,label = shuffle(immatrix, imlabel, random_state=42)
train_data = [data,label]



"""import matplotlib.pyplot as plt
```

```
plt.hist(label)
plt.show()"""
```

```
'import matplotlib.pyplot as plt\n\nplt.hist(label)\nplt.show()'
```

Input Data Parameters

```
    from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(train_data[0], train_data[1], test_size = 0.1, random

print(np.array(x_train).shape)
print(np.array(y_train).shape)
```

```
(542, 256, 256, 3)
(542,)
```

```
from keras.utils import np_utils

y_train = np_utils.to_categorical(np.array(y_train), 5)
y_test = np_utils.to_categorical(np.array(y_test), 5)

x_train = np.array(x_train).astype("float32")/255.
x_test = np.array(x_test).astype("float32")/255.

print(np.array(y_train).shape)
```

```
(542, 5)
```

MODEL

```
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPooling2D, BatchNormalization

model = Sequential()

model.add(Conv2D(32, (3, 3), padding='same', activation='relu', input_shape=x_train[0].shape))# 32=filter
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D())# Takes max value in each stride(Dimension of image is reduced to 128*128)
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), padding='same', activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D())
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), padding='same', activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D())
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(activation='softmax', units=5))

model.compile(loss='categorical_crossentropy', optimizer = keras.optimizers.SGD(lr=0.0001, momentum=0.9),
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/op_def_
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py
```

```
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 256, 256, 32)      896
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 254, 254, 32)      9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 127, 127, 32)      0
_________________________________________________________________
dropout_1 (Dropout)          (None, 127, 127, 32)      0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 127, 127, 64)      18496
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 125, 125, 64)      36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 62, 62, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 62, 62, 64)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 62, 62, 64)        36928
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 60, 60, 64)        36928
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 30, 30, 64)        0
_________________________________________________________________
dropout_3 (Dropout)          (None, 30, 30, 64)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 57600)             0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               29491712
_________________________________________________________________
dropout_4 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 5)                 2565
=================================================================
Total params: 29,633,701
Trainable params: 29,633,701
Non-trainable params: 0
_________________________________________________________________
```

Training DATA

```
model.fit(x_train, y_train, batch_size = 64, epochs=10, shuffle=True, verbose=2)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/ops/math_ops.py:3
Instructions for updating:
Use tf.cast instead.
Epoch 1/10
 - 13s - loss: 1.5997 - acc: 0.3266
Epoch 2/10
```

```
Epoch 2/10
 - 3s - loss: 1.5807 - acc: 0.5627
Epoch 3/10
 - 3s - loss: 1.5558 - acc: 0.6734
Epoch 4/10
 - 3s - loss: 1.5161 - acc: 0.7066
```

Prediction

```
predictions = model.predict(x_test)
predictions
```

```
array([[0.5780761 , 0.09461828, 0.15250176, 0.08684509, 0.08795873],
       [0.629143  , 0.08148205, 0.14001   , 0.07347154, 0.07589347],
       [0.5495231 , 0.10200721, 0.15903297, 0.09438422, 0.09505248],
       [0.59158486, 0.09079698, 0.15071052, 0.08269171, 0.08421589],
       [0.5583748 , 0.09957556, 0.15758628, 0.09146781, 0.09299555],
```

```
[0.5739008 , 0.09535644, 0.15368885, 0.08790624, 0.08914769],
[0.5540983 , 0.10068357, 0.15825234, 0.0929853 , 0.09398045],
[0.56842035, 0.09685867, 0.15424848, 0.08938818, 0.09108431],
[0.63747054, 0.07936095, 0.13894136, 0.07111991, 0.0731073 ],
[0.55499196, 0.10064775, 0.15781179, 0.0928616 , 0.09368689],
[0.6187889 , 0.08380418, 0.14248215, 0.07641768, 0.07850713],
[0.5734024 , 0.09540585, 0.1547461 , 0.08769451, 0.08875114],
```

Evaluating

```
score = model.evaluate(x_test, y_test, verbose=0)
print(score)
```

```
[1.0605540549168821, 0.6721311446096077]
```

```
[0.57346255, 0.09567214, 0.15429991, 0.08780926, 0.08875608],
[0.54547805, 0.10306647, 0.15986879, 0.09521086, 0.09637579],
[0.5703307 , 0.0965352 , 0.15499993, 0.08823699, 0.08989722],
[0.53750473, 0.10535413, 0.16145022, 0.09744503, 0.09824588],
[0.58499366, 0.09311945, 0.15111734, 0.08494262, 0.0858269 ],
[0.55215514, 0.10195999, 0.15838398, 0.0928552 , 0.0946457 ],
[0.59032387, 0.09142389, 0.15026626, 0.08347742, 0.08450855],
[0.5414829 , 0.10431808, 0.1607477 , 0.09640978, 0.09704151],
[0.5850711 , 0.09231877, 0.15195675, 0.08456452, 0.08608889],
[0.55954   , 0.09989772, 0.1563495 , 0.0912514 , 0.09296136],
[0.57830584, 0.09449593, 0.1530922 , 0.08584143, 0.08826451],
[0.52740777, 0.10789976, 0.16272292, 0.10057264, 0.10139691],
[0.57274294, 0.09660713, 0.15261352, 0.08875547, 0.08928099],
[0.55115414, 0.10138675, 0.15900691, 0.09376634, 0.09468593],
[0.51517737, 0.11127628, 0.16602981, 0.10336532, 0.10415129],
[0.63652766, 0.07950936, 0.13930337, 0.0713691 , 0.07329047],
[0.53549206, 0.10572834, 0.162062  , 0.09789051, 0.09882706],
[0.57073045, 0.09661161, 0.15411708, 0.08875508, 0.08978578],
[0.5591984 , 0.09944284, 0.15680209, 0.09181578, 0.09274086],
[0.58070874, 0.09370738, 0.15260474, 0.08635961, 0.0866195 ],
[0.5370737 , 0.1054162 , 0.16147982, 0.09781758, 0.09821282],
[0.5833239 , 0.09293648, 0.15139017, 0.08529577, 0.08705372],
[0.615677  , 0.08403347, 0.1446909 , 0.07678344, 0.07881525],
[0.61998755, 0.08288613, 0.14265195, 0.0756042 , 0.07887018],
[0.6606859 , 0.07359168, 0.13168226, 0.0655616 , 0.0684786 ],
[0.5916723 , 0.09062289, 0.15028825, 0.08299875, 0.08441782],
[0.6023731 , 0.08804872, 0.14682542, 0.08040732, 0.08234541],
[0.61404794, 0.08505619, 0.14474675, 0.07745178, 0.07869732],
[0.5631176 , 0.0977737 , 0.15569997, 0.09075908, 0.0926496 ],
[0.5683451 , 0.09702868, 0.15472263, 0.08947407, 0.09042953],
```