Sardar Patel Institute of Technology

(Electronics & Telecommunication Department)

Name: Bhavesh Jadhav
Third Year EXTC

Major Project Data Science

1.} MAJOR PROJECT 1

Choose any dataset of your choice and apply a suitable CLASSIFIER/REGRESSOR .

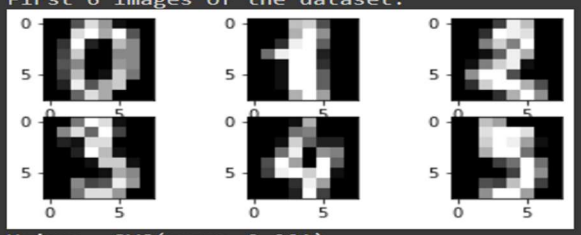
→In this project I have tried to apply all basic concepts that were taught during my lectures.

Python Code:

```
from __future__ import division
from sklearn.datasets import load_digits
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split as tts
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
# Loading digits dataset
digits = load_digits()
# Create feature matrix
x = digits.data
# Create target vector
y = digits.target

# First 6 images stored in the images attribute of the dataset
print("First 6 images of the dataset: ")

for x in range (6):

    plt.subplot(330 + 1 + x)
    plt.imshow(digits.images[x], cmap=plt.get_cmap('gray'))

plt.show()
# Flattening the image to apply classifier
n_samples = len(digits.images)
data = digits.images.reshape((n_samples, -1))

# Splitting the data into training and testing
```

```
x_train, x_test, y_train, y_test = train_test_split(data, digits.targ
et, test_size=0.5, shuffle=False)

# Creating a classifier. SVM is set as default but you can test out o
ther two as well by commenting out SVM and un-
commennting the one you wish to try
clf = svm.SVC (gamma=0.001)

# Decision Tree Classifier
#clf = tree.DecisionTreeClassifier()

# Random Forest Classifier
#clf = RandomForestClassifier()

# Printing the details of the Classifier used
print ("Using: ", clf)

# Training
clf.fit(x_train, y_train)
# Predicting
predictions = clf.predict(x_test)
#print ("\nPredictions:", predictions)

score = 0
for i in range(len(predictions)):

    if predictions[i] == y_test[i]:

        score += 1

print ("Accuracy:", (score / len(predictions)) * 100, "%")
# print accuracy_score(test_labels, predictions)
```

```
First 6 images of the dataset:
```



```
Using:  SVC(gamma=0.001)
Accuracy: 96.88542825361512 %
```

2.} MAJOR PROJECT 2

Create any of the Image Processing Projects using Numpy and/or OpenCV.

→Here I have tried to apply all basic concepts regarding image processing.

Python Code:

```
# importing all the required libraries
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image, ImageOps
```

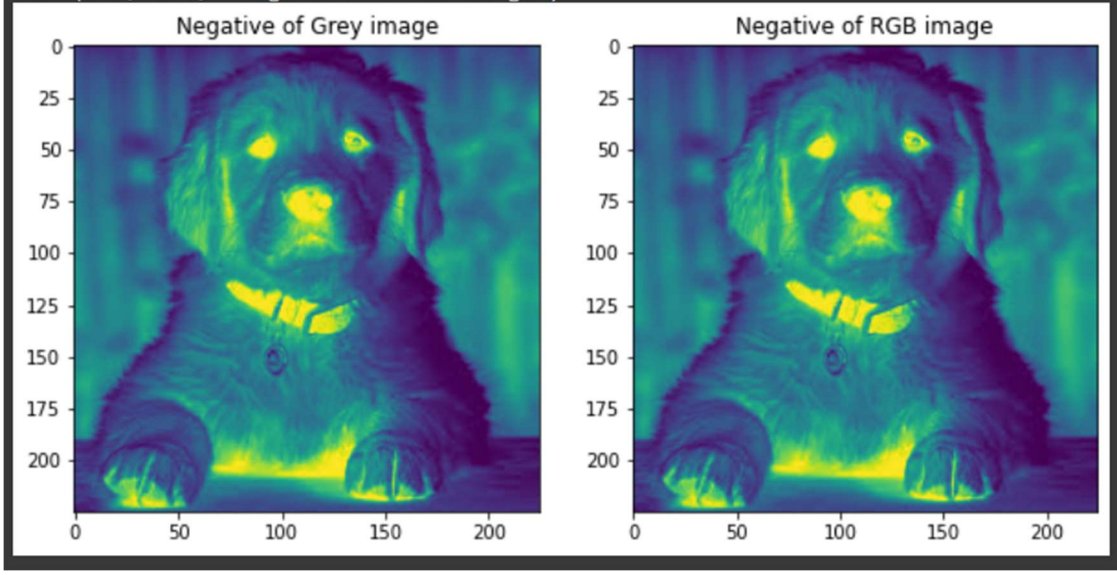
```
import cv2
img=cv2.imread('7.jpg',0)
```

```
print('# of dims: ',img.ndim)      # dimension of an image
print('Img shape: ',img.shape)     # shape of an image
print('Dtype: ',img.dtype)
print(img[20, 20])                 # pixel value at [R, G, B]
```

```
# of dims:  2
Img shape:  (225, 225)
Dtype:  uint8
196
```

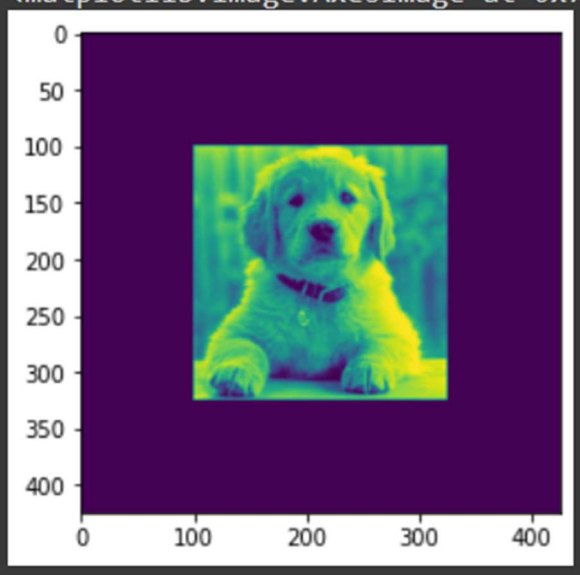
```
fig = plt.figure(figsize=(10, 10))
img_grey = 255*3 - img         # 255 * 3 because we added along channe
l axis previously
fig.add_subplot(1, 2, 1)
plt.imshow(img_grey)
plt.title('Negative of Grey image')
img = 255 - img
fig.add_subplot(1, 2, 2)
plt.imshow(img)
plt.title('Negative of RGB image')
```

```
Text(0.5, 1.0, 'Negative of RGB image')
```



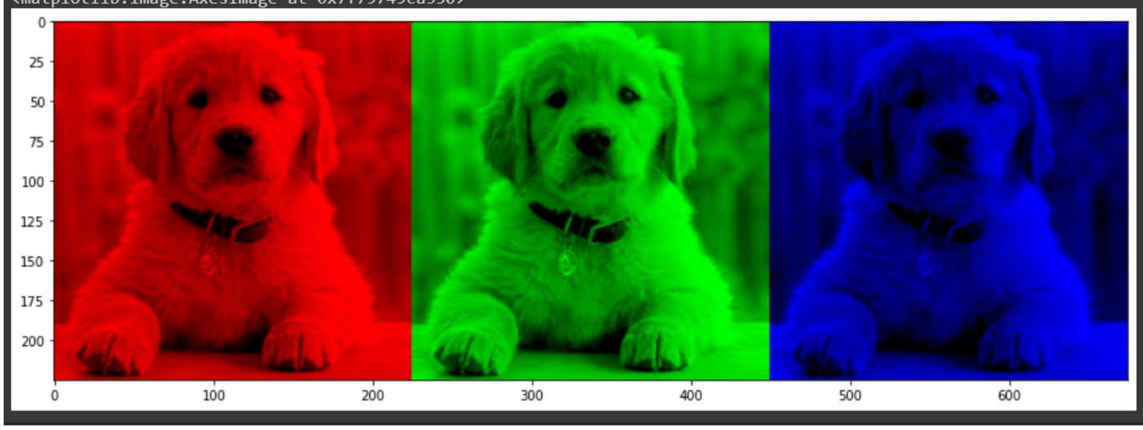

```
img = np.array(Image.open('7.jpg'))
img_grey = img.sum(2) / (255*3)
img0 = img_grey.copy()
img0 = np.pad(img0, ((100,100),(100,100)), mode='constant')
plt.imshow(img0)
```

```
<matplotlib.image.AxesImage at 0x7f79749d6f50>
```



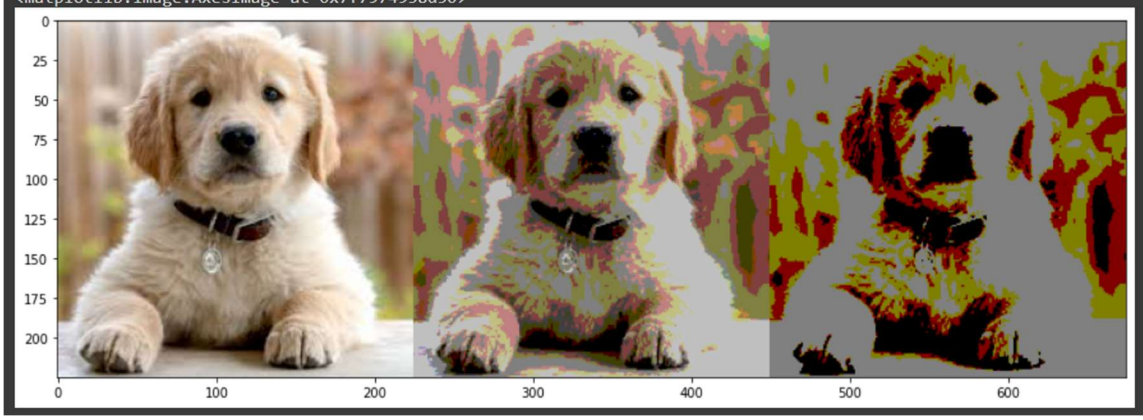
```
img = np.array(Image.open('7.jpg'))
img_R, img_G, img_B = img.copy(), img.copy(), img.copy()
img_R[:, :, (1, 2)] = 0
img_G[:, :, (0, 2)] = 0
img_B[:, :, (0, 1)] = 0
img_rgb = np.concatenate((img_R,img_G,img_B), axis=1)
plt.figure(figsize=(15, 15))
plt.imshow(img_rgb)
```

```
<matplotlib.image.AxesImage at 0x7f79749ca350>
```

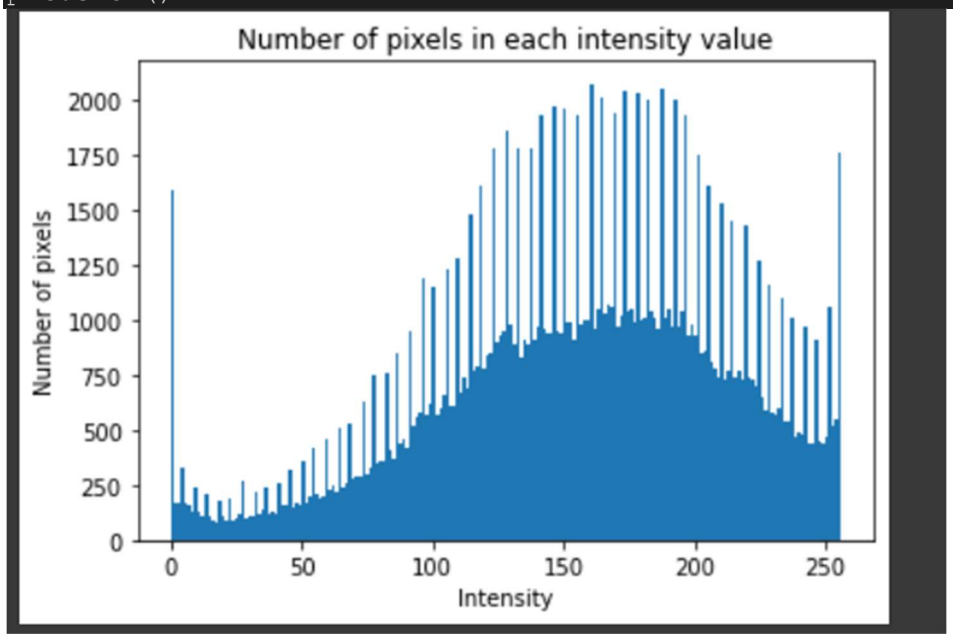



```
img = np.array(Image.open('7.jpg'))
# Making Pixel values discrete by first division by // which gives in
t and then multiply by the same factor
img_0 = (img // 64) * 64
img_1 = (img // 128) * 128
img_all = np.concatenate((img, img_0, img_1), axis=1)
plt.figure(figsize=(15, 15))
plt.imshow(img_all)
```

```
<matplotlib.image.AxesImage at 0x7f7974938d50>
```



```
img = np.array(Image.open('7.jpg'))
img_flat = img.flatten()
plt.hist(img_flat, bins=200, range=[0, 256])
plt.title("Number of pixels in each intensity value")
plt.xlabel("Intensity")
plt.ylabel("Number of pixels")
plt.show()
```



Github Link:
https://github.com/lunaticfringe18/RINEX6

-x-x-x-x-x-