Experiment-12

Classification of objects (Binary Classification)

Name : N U Praneeth Reddy **Reg.No:21BAI1500**

Aim: To perform binary classification on an image dataset.

Resources Used: Anaconda Python Environment
VSCode

Theory :

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

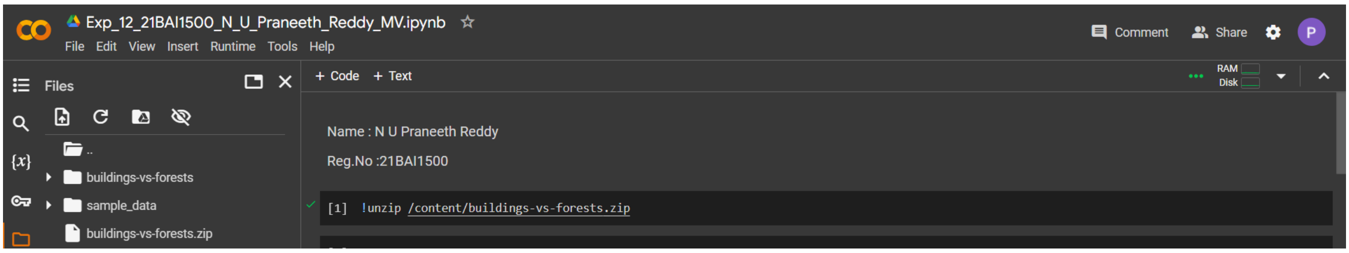
Tasks:

1. Download and prepare the dataset in such way that you have two splits- train set and test set (70/30)
2. Create the target class for your input namely building -- 1, forest –0
3. Extract features 3 to 5 features if you are using GLCM. 5-8 features if you are using LBP
4. Now train two ML models namely k-NN and SVM for binary classification. Choose appropriate parameters.
5. Compare ML performance for building vs forest classification based on Accuracy and F1-Score

Procedure:

Task 1: Download and prepare the dataset in such way that you have two splits- train set and test set (70/30)

1)Upload the zip file in Collab and unzip the zip file

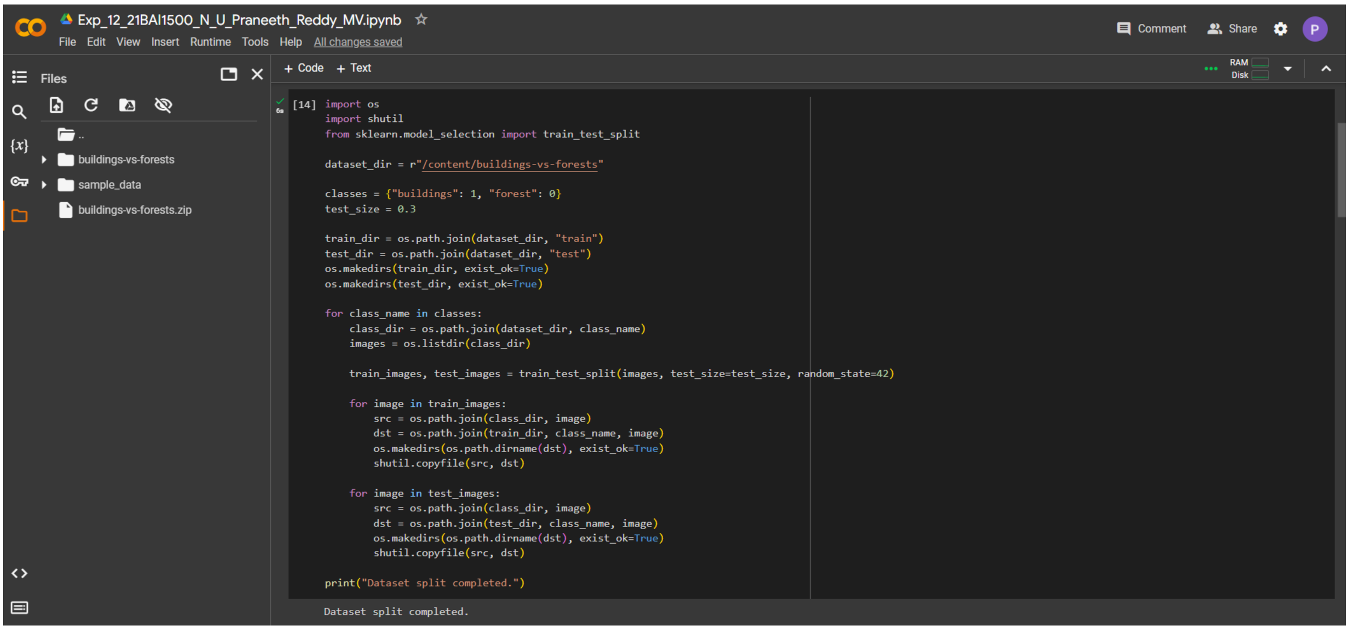


Name : N U Praneeth Reddy

Reg.No :21BAI1500

```
[1] !unzip /content/buildings-vs-forests.zip
```

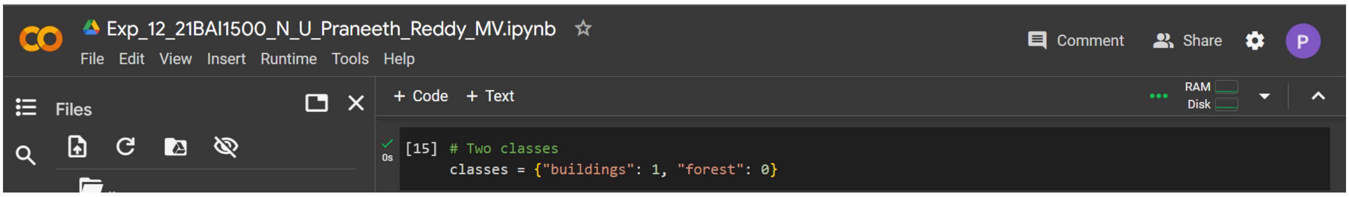
2) Set the images in two folders, with one having the images of the buildings and the other having the images of the forest and then split it into train and test with 70% - 30% split.


Exp_12_21BAI1500_N_U_Praneeth_Reddy_MV.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
+ Code + Text
buildings-vs-forests
sample_data
buildings-vs-forests.zip

```
[14] import os
     import shutil
     from sklearn.model_selection import train_test_split

     dataset_dir = r"/content/buildings-vs-forests"

     classes = {"buildings": 1, "forest": 0}
     test_size = 0.3

     train_dir = os.path.join(dataset_dir, "train")
     test_dir = os.path.join(dataset_dir, "test")
     os.makedirs(train_dir, exist_ok=True)
     os.makedirs(test_dir, exist_ok=True)

     for class_name in classes:
         class_dir = os.path.join(dataset_dir, class_name)
         images = os.listdir(class_dir)

         train_images, test_images = train_test_split(images, test_size=test_size, random_state=42)

         for image in train_images:
             src = os.path.join(class_dir, image)
             dst = os.path.join(train_dir, class_name, image)
             os.makedirs(os.path.dirname(dst), exist_ok=True)
             shutil.copyfile(src, dst)

         for image in test_images:
             src = os.path.join(class_dir, image)
             dst = os.path.join(test_dir, class_name, image)
             os.makedirs(os.path.dirname(dst), exist_ok=True)
             shutil.copyfile(src, dst)

     print("Dataset split completed.")
```

```
Dataset split completed.
```

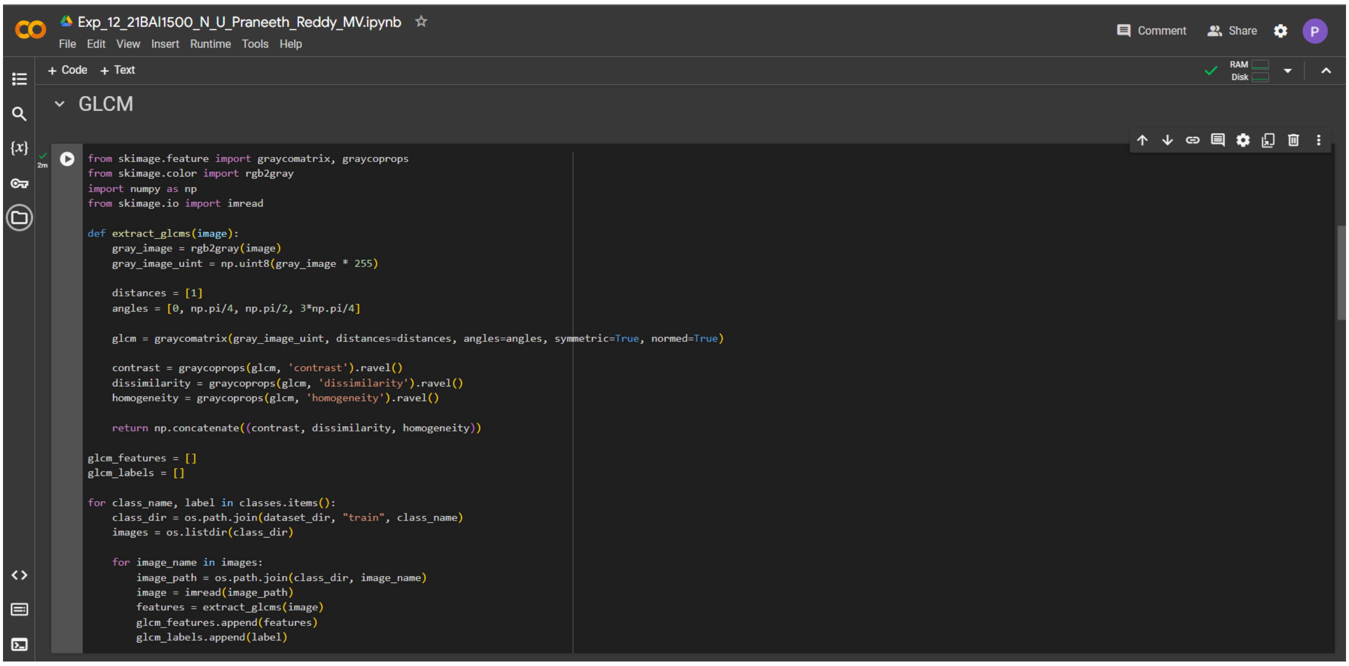
Task -2: Create the target class for your input namely building -- 1, forest –0


Exp_12_21BAI1500_N_U_Praneeth_Reddy_MV.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
Files
+ Code + Text

```
[15] # Two classes
     classes = {"buildings": 1, "forest": 0}
```

Task -3: Extract features 3 to 5 features if you are using GLCM. 58 features if you are using LBP

For GLCM :


Exp_12_21BAI1500_N_U_Praneeth_Reddy_MV.ipynb
File Edit View Insert Runtime Tools Help
Comment Share
+ Code + Text
RAM Disk
GLCM
2m

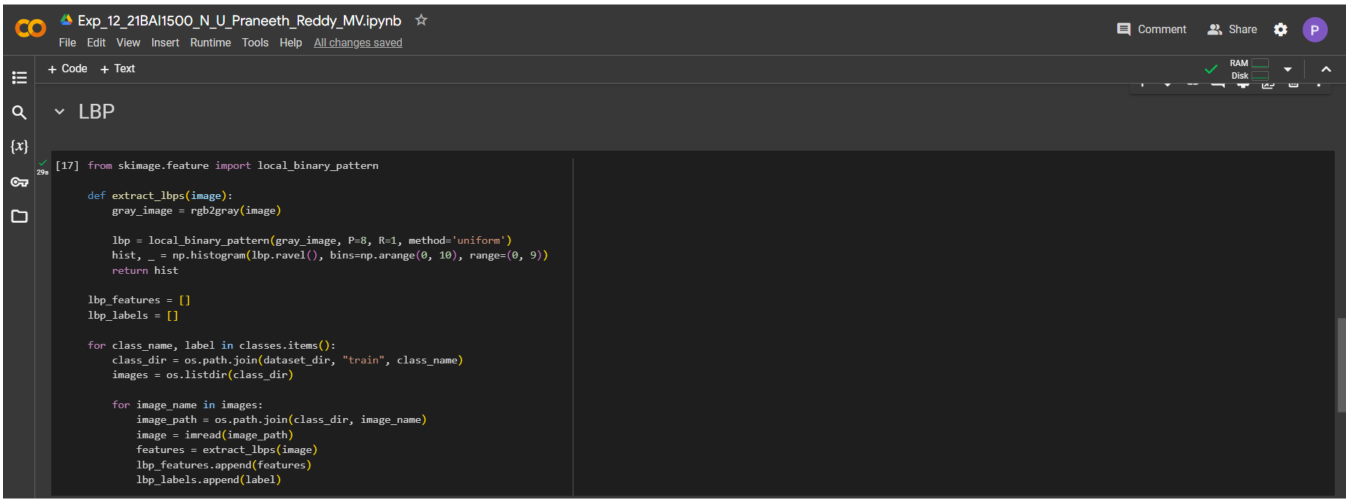
```
from skimage.feature import graycomatrix, graycoprops
from skimage.color import rgb2gray
import numpy as np
from skimage.io import imread

def extract_glcms(image):
    gray_image = rgb2gray(image)
    gray_image_uint = np.uint8(gray_image * 255)

    distances = [1]
    angles = [0, np.pi/4, np.pi/2, 3*np.pi/4]

    glcm = graycomatrix(gray_image_uint, distances=distances, angles=angles, symmetric=True, normed=True)

    contrast = graycoprops(glcm, 'contrast').ravel()
    dissimilarity = graycoprops(glcm, 'dissimilarity').ravel()
    homogeneity = graycoprops(glcm, 'homogeneity').ravel()

    return np.concatenate((contrast, dissimilarity, homogeneity))

glcm_features = []
glcm_labels = []

for class_name, label in classes.items():
    class_dir = os.path.join(dataset_dir, "train", class_name)
    images = os.listdir(class_dir)

    for image_name in images:
        image_path = os.path.join(class_dir, image_name)
        image = imread(image_path)
        features = extract_glcms(image)
        glcm_features.append(features)
        glcm_labels.append(label)
```

We are extracting 4 features in the GLCM which are at angles 0,pi/4, pi/2, 3pi/4.

The features extracted are stored in a list names glcm features and its corresponding label is appended to the glcm labels list.

We are going to use these features and labels for running machine learning models.

For LBP :


Exp_12_21BAI1500_N_U_Praneeth_Reddy_MV.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
LBP
29s

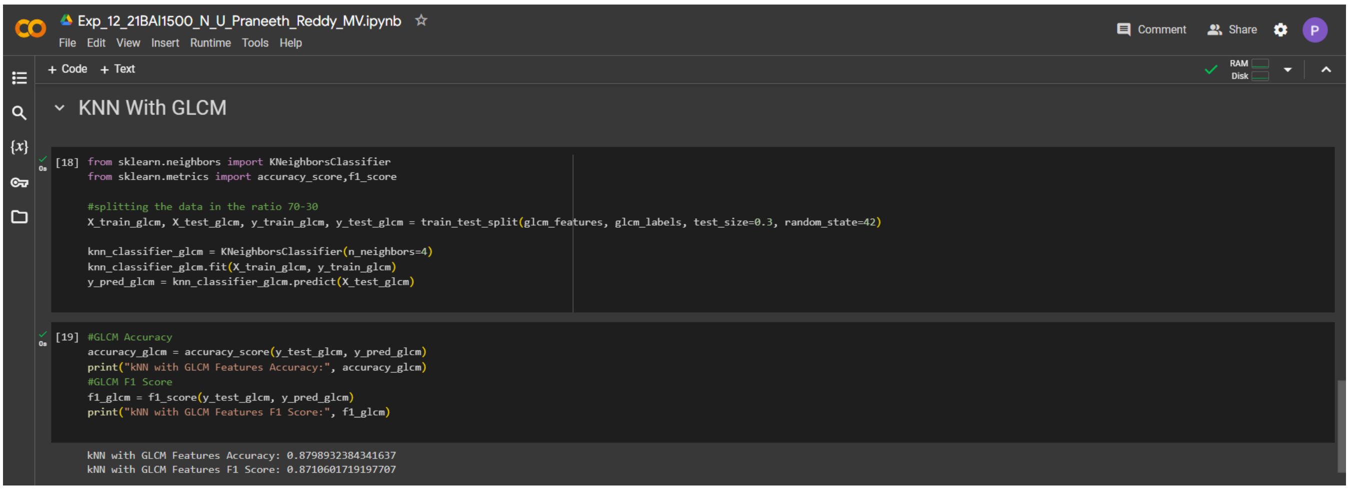
```
[17] from skimage.feature import local_binary_pattern

     def extract_lbps(image):
         gray_image = rgb2gray(image)

         lbp = local_binary_pattern(gray_image, P=8, R=1, method='uniform')
         hist, _ = np.histogram(lbp.ravel(), bins=np.arange(0, 10), range=(0, 9))
         return hist

     lbp_features = []
     lbp_labels = []

     for class_name, label in classes.items():
         class_dir = os.path.join(dataset_dir, "train", class_name)
         images = os.listdir(class_dir)

         for image_name in images:
             image_path = os.path.join(class_dir, image_name)
             image = imread(image_path)
             features = extract_lbps(image)
             lbp_features.append(features)
             lbp_labels.append(label)
```

We are extracting eight features using LBP from the given image dataset.

The extracted features are stored in a list named lbp features with its corresponding label appended to the lbp labels list.

Task -4,5: Now train two ML models namely k-NN and SVM for binary classification. Choose appropriate parameters and print the accuracy score and the f1 score of the models.

KNN with GLCM :


Exp_12_21BAI1500_N_U_Praneeth_Reddy_MV.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
+ Code + Text
RAM
Disk

KNN With GLCM

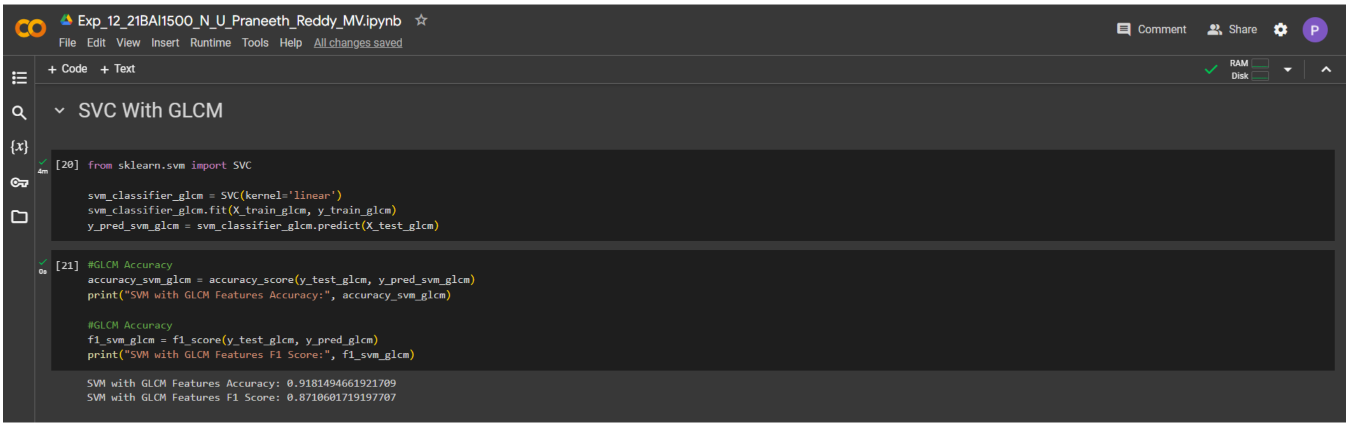
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score,f1_score

#splitting the data in the ratio 70-30
X_train_glcm, X_test_glcm, y_train_glcm, y_test_glcm = train_test_split(glcm_features, glcm_labels, test_size=0.3, random_state=42)

knn_classifier_glcm = KNeighborsClassifier(n_neighbors=4)
knn_classifier_glcm.fit(X_train_glcm, y_train_glcm)
y_pred_glcm = knn_classifier_glcm.predict(X_test_glcm)
```

```
#GLCM Accuracy
accuracy_glcm = accuracy_score(y_test_glcm, y_pred_glcm)
print("kNN with GLCM Features Accuracy:", accuracy_glcm)
#GLCM F1 Score
f1_glcm = f1_score(y_test_glcm, y_pred_glcm)
print("kNN with GLCM Features F1 Score:", f1_glcm)
```

```
kNN with GLCM Features Accuracy: 0.8798932384341637
kNN with GLCM Features F1 Score: 0.8710601719197707
```

We have built a KNN model with 4 neighbors and GLCM features and got an accuracy score of 0.88 and F1 score of 0.87.

SVC with GLCM :


Exp_12_21BAI1500_N_U_Praneeth_Reddy_MV.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk

SVC With GLCM

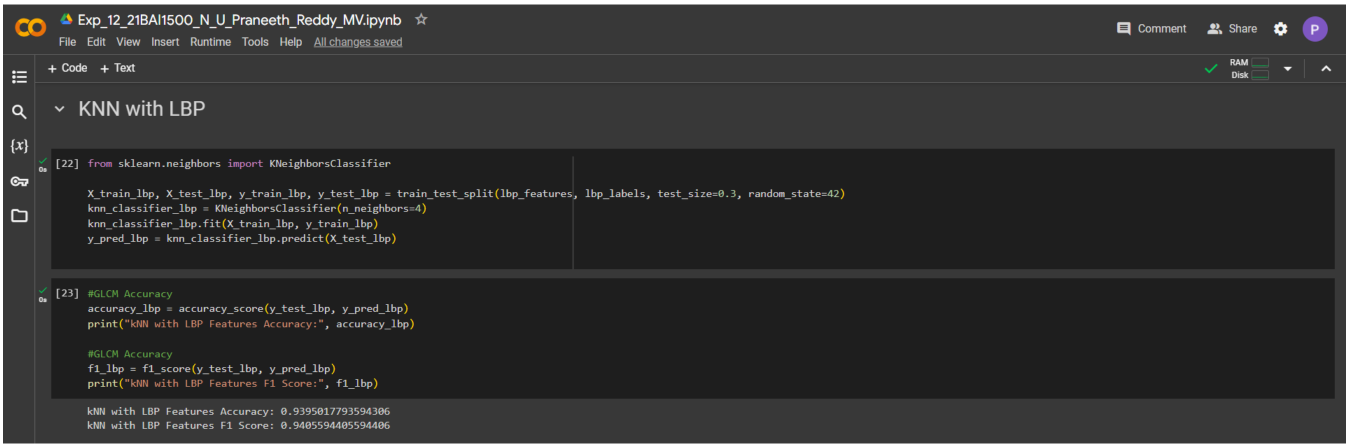
```
from sklearn.svm import SVC

svm_classifier_glcm = SVC(kernel='linear')
svm_classifier_glcm.fit(X_train_glcm, y_train_glcm)
y_pred_svm_glcm = svm_classifier_glcm.predict(X_test_glcm)
```

```
#GLCM Accuracy
accuracy_svm_glcm = accuracy_score(y_test_glcm, y_pred_svm_glcm)
print("SVM with GLCM Features Accuracy:", accuracy_svm_glcm)

#GLCM Accuracy
f1_svm_glcm = f1_score(y_test_glcm, y_pred_glcm)
print("SVM with GLCM Features F1 Score:", f1_svm_glcm)
```

```
SVM with GLCM Features Accuracy: 0.9181494661921709
SVM with GLCM Features F1 Score: 0.8710601719197707
```

We have built an SVM model with GLCM features and got an accuracy score of 0.91and F1 score of 0.87.

KNN with LBP :


Exp_12_21BAI1500_N_U_Praneeth_Reddy_MV.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
KNN with LBP

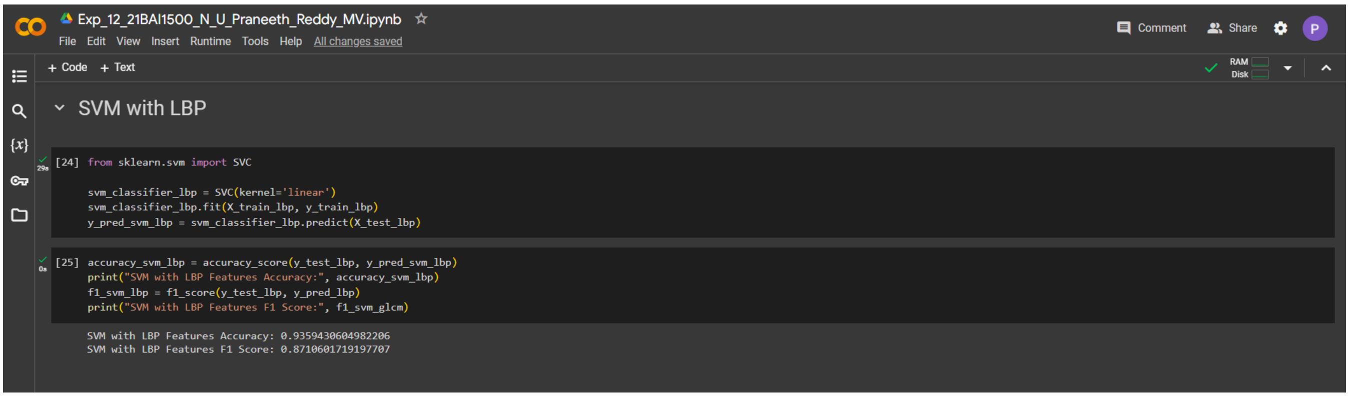
```
[22] from sklearn.neighbors import KNeighborsClassifier

     X_train_lbp, X_test_lbp, y_train_lbp, y_test_lbp = train_test_split(lbp_features, lbp_labels, test_size=0.3, random_state=42)
     knn_classifier_lbp = KNeighborsClassifier(n_neighbors=4)
     knn_classifier_lbp.fit(X_train_lbp, y_train_lbp)
     y_pred_lbp = knn_classifier_lbp.predict(X_test_lbp)
```

```
[23] #GLCM Accuracy
     accuracy_lbp = accuracy_score(y_test_lbp, y_pred_lbp)
     print("kNN with LBP Features Accuracy:", accuracy_lbp)

     #GLCM Accuracy
     f1_lbp = f1_score(y_test_lbp, y_pred_lbp)
     print("kNN with LBP Features F1 Score:", f1_lbp)
```

```
kNN with LBP Features Accuracy: 0.9395017793594306
kNN with LBP Features F1 Score: 0.9405594405594406
```

We have built a KNN model with 4 neighbors and LBP features and got an accuracy score of 0.939 and F1 score of 0.94.

SVC with LBP :


Exp_12_21BAI1500_N_U_Praneeth_Reddy_MV.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
SVM with LBP

```
[24] from sklearn.svm import SVC

     svm_classifier_lbp = SVC(kernel='linear')
     svm_classifier_lbp.fit(X_train_lbp, y_train_lbp)
     y_pred_svm_lbp = svm_classifier_lbp.predict(X_test_lbp)
```

```
[25] accuracy_svm_lbp = accuracy_score(y_test_lbp, y_pred_svm_lbp)
     print("SVM with LBP Features Accuracy:", accuracy_svm_lbp)
     f1_svm_lbp = f1_score(y_test_lbp, y_pred_lbp)
     print("SVM with LBP Features F1 Score:", f1_svm_glcm)
```

```
SVM with LBP Features Accuracy: 0.9359430604982206
SVM with LBP Features F1 Score: 0.8710601719197707
```

We have built an SVM model with LBP features and got an accuracy score of 0.935 and F1 score of 0.871.

Results: The given tasks have been done with the help machine learning models and GLCM and LBP feature extractors.

Conclusion: Python program have been created to classify two types of images using the features generated from GLCM and LBP with KNN model and SVM model.

Google Collab Link :

https://colab.research.google.com/drive/1bd4POT5XWBB2UTJIRhKXZWN-rlU12Fau?usp=sharing