**KNN IMPLEMENTATION ON 20 IMAGES**

**Data Specifications:**

**Total Images: 20 (10 Cats and 10 Dogs)**

**Split Ratio: 80/20**

from skimage.feature import hog

import numpy as np

import cv2

import matplotlib.pyplot as plt

from matplotlib import image as mpimg

import os

import glob

from sklearn.neighbors import kNeighborsClassifier

from sklearn import metrics

**Image Featuring Technique**

img = np.array(mpimg.imread('Cats and Dogs/training\_set/dogs/Dog 03.jpg'))

img.setflags(write=1)

print('image:', img.shape)

plt.imshow(img)

plt.show()

**Resizing:**

resized\_img = cv2.resize(img,(64, 128))

plt.imshow(resized\_img)

plt.show()

print(resized\_img.shape)

**Hog Features:**

fd, hog\_image = hog(resized\_img, visualize=True, multichannel=True)

print(fd.shape)

print (fd)

print(hog\_image.shape)

plt.axis("off")

plt.imshow(hog\_image, cmap="gray")

plt.show()

**Training Dataset:**

# DOG TRAIN DATA

data\_dogs = []

for entry in glob.glob("Cats and Dogs/training\_set/dogs/\*.jpg"):

  img = np.array(mpimg.imread(entry))

  resized\_img = cv2.resized(img,(64,32))

  fd = hog(resized\_img)

  data\_dogs.append(fd)

# CATS TRAIN DATA

data\_cats = []

for entry in glob.glob("Cats and Dogs/training\_set/cats/\*.jpg"):

  img = np.array(mpimg.imread(entry))

  resized\_img = cv2.resized(img,(64,32))

  fd = hog(resized\_img)

  data\_dogs.append(fd)

# COMBINE TRAIN DATA

train\_data = data\_cats + data\_dogs

print(len(train\_data))

Train Data Labeling:

train\_lable\_list = []

for i in range(len(train\_data)):

  if i < 8:

    train\_lable\_list.append('cat')

  else:

    train\_lable\_list.append('dog')

  print(train\_lable\_list)

  print(len(train\_lable\_list))

**Test Dataset:**

# LABELED DOGS TEST DATA

test\_dogs = []

for entry in glob.glob("Cats and Dogs/test\_set/dogs/\*.jpg"):

  img = np.array(mpimg.imread(entry))

  resized\_img = cv2.resize(img,(64, 32))

  fd = hog(resized\_img)

  test\_dict = {'data':fd, 'lable':'dog'}

  test\_dogs.append(test\_dict)

# LABELED CATS TEST DATA

test\_cats = []

for entry in glob.glob("Cats and Dogs/test\_set/CATS/\*.jpg"):

  img = np.array(mpimg.imread(entry))

  resized\_img = cv2.resize(img,(64, 32))

  fd = hog(resized\_img)

  test\_dict = {'data':fd, 'lable':'cat'}

  test\_dogs.append(test\_dict)

# LABELED COMBINED TEST DATA

test\_data = test\_dogs + test\_cats

print(len(test\_data))

Separating Data:

test\_features = []

test\_lables = []

for i in test\_data:

  test\_lables.append(i['lable'])

  test\_features.appennd(i['data'])

print(len(test\_features))

print(test\_lables)

**KNN Implementation:**

Euclidean Training:

x\_axis\_k\_points = []

# LIST OF METRICS

f1\_euclidean = []

accuracies\_euclidean = []

conf\_matrix\_euclidean = []

for k in range(5):

  #kNN CLASSIFIER Train data

  knn\_euclidean = kNeighborsClassifier(n\_neighbors=k+1)

  knn\_euclidean.fit(train\_data,train\_lable\_list)

  # kNN Classifier Prediction

  pred\_lables\_euclidean = knn\_euclidean.predict(test\_features)

  # Accuracy of prediction

  acc\_euclidean = knn\_euclidean.score(test\_features,test\_lables)

  accuracies\_euclidean.append(acc\_euclidean)

  # Confusion Matrix of Prediction

  conf\_matrix\_euclidean.append(metrics.confusion\_matrix(test\_lables, pred\_lables\_euclidean))

  # F1 Score of Prediction

  f1\_euclidean.append(metrics.f1\_score(test\_lables, pred\_lables\_euclidean, pos\_lable='dog'))

  x\_axis\_k\_points.append(k+1)