Programming Assignment 1:
Data Preparation and Understanding

1. In this semester, we will be using the "Stanford Dogs" dataset (http://vision.stanford.edu/aditya86/ImageNetDogs/) for all our 4 programming assignments. There are a total of 120 classes (dog breeds). The number of images for each class ranges from 148 to 252.
Each student will
(a) be assigned 4 classes to work on the 4 assignments.
(b) download Images (and also Annotations - bounding boxes) datasets for the 4 classes to work on.
(c) create a Github account to share (as collaborator) their solution (Readme, Codes, Processed Dataset
for Code to run correctly) with the grader.

2. Use XML processing modules(https://docs.python.org/3/library/xml.html) to obtain bounding box information from Annotations datasets and scikit-Image (Reference: https://scikit-image. org/) to perform image processing and feature extraction.

```
import os

annotations_dir = '/content/sample_data/Annotation/'

if os.path.exists(annotations_dir):
   print("Annotation directory exists. Listing files...")
   for root, dirs, files in os.walk(annotations_dir):
      print(f"Found directory: {root}")
      for file in files:
         print(f"File: {file}")
else:
   print("Annotation directory not found!")
```

OUTPUT:

```
Streaming output truncated to the last 5000 lines.
File: n02109525_7444
File: n02109525_3360
File: n02109525_16487
File: n02109525_393
```

File: n02109525_11155
File: n02109525_10417
File: n02109525_997
File: n02109525_7497
File: n02109525_9345
File: n02109525_1880
File: n02109525_13580
File: n02109525_16981
File: n02109525_6931
File: n02109525_6019
File: n02109525_3531
File: n02109525_11452
File: n02109525_11732
File: n02109525_17140
File: n02109525_13154
File: n02109525_13420
File: n02109525_10332
File: n02109525_4516
File: n02109525_14297
File: n02109525_12736
File: n02109525_6593
File: n02109525_13906
File: n02109525_2840
File: n02109525_10297
File: n02109525_2385
File: n02109525_13702
File: n02109525_14198
File: n02109525_15506
File: n02109525_15376
File: n02109525_18136
File: n02109525_3667
File: n02109525_8822
File: n02109525_7982
File: n02109525_1864
File: n02109525_6663
File: n02109525_5385
File: n02109525_18371
File: n02109525_16215
File: n02109525_7085
File: n02109525_2104

File: n02109525_13812
File: n02109525_12610
File: n02109525_2369
File: n02109525_7579
File: n02109525_4308
File: n02109525_499
File: n02109525_1843
File: n02109525_9522
File: n02109525_539
File: n02109525_1575
File: n02109525_6693
File: n02109525_9130
File: n02109525_751
File: n02109525_1624
File: n02109525_12041
File: n02109525_4021
File: n02109525_10960
File: n02109525_2943
File: n02109525_1271
File: n02109525_8417
File: n02109525_18685
File: n02109525_444
Found directory: /content/sample_data/Annotation/ n02093991-Irish_terrier

(a) Cropping and Resize Images in Your 4-class Images Dataset: Use the bounding box information in the Annotations dataset relevant to your 4-class Images Dataset to crop the images in your dataset and then resize each image to a 128×128 pixel image.

```
import os
import numpy as np
from PIL import Image
import cv2
from sklearn.decomposition import PCA
from sklearn.preprocessing import normalize
from sklearn.metrics import pairwise
import matplotlib.pyplot as plt

images_folder_path = '/content/sample_data/Images'
cropped_images_path = '/content/sample_data/Cropped_Images'
```

```
os.makedirs(cropped_images_path, exist_ok=True)

def load_images(image_folder):
    images = []
    valid_extensions = ('.jpg', '.jpeg', '.png', '.bmp')
    for breed in os.listdir(image_folder):
        breed_path = os.path.join(image_folder, breed)
        if os.path.isdir(breed_path):
            for img_file in os.listdir(breed_path):
                img_path = os.path.join(breed_path, img_file)
                if img_file.lower().endswith(valid_extensions):
                    try:
                        img = Image.open(img_path).convert('RGB')
                        images.append(np.array(img))
                    except Exception as e:
                        print(f"Error loading image {img_path}: {e}")
    return images

dog_images = load_images(images_folder_path)

def crop_and_resize_images(images):
    resized_images = []
    for img in images:

        h, w, _ = img.shape
        center_h, center_w = h // 2, w // 2
        cropped_img = img[center_h-50:center_h+50, center_w-50:center_w+50]
                        resized_img = cv2.resize(cropped_img, (128, 128),
interpolation=cv2.INTER_AREA)
        resized_images.append(resized_img)
    return np.array(resized_images)

# Crop and resize images
cropped_resized_images = crop_and_resize_images(dog_images)

# Function to compute color histograms
def compute_histograms(images):
    histograms = []
    for img in images:
```

```
        hist_r = cv2.calcHist([img], [0], None, [256], [0, 256])
        hist_g = cv2.calcHist([img], [1], None, [256], [0, 256])
        hist_b = cv2.calcHist([img], [2], None, [256], [0, 256])
        hist = np.concatenate((hist_r, hist_g, hist_b), axis=0)
        histograms.append(hist.flatten())
    return np.array(histograms)

histograms = compute_histograms(cropped_resized_images)

def compute_similarity_measurements(histograms):
    distances = {}
    for i in range(len(histograms)):
        for j in range(i + 1, len(histograms)):

            euclidean_dist = np.linalg.norm(histograms[i] - histograms[j])
            distances[(i, j)] = {'Euclidean': euclidean_dist}

            manhattan_dist = np.sum(np.abs(histograms[i] - histograms[j]))
            distances[(i, j)]['Manhattan'] = manhattan_dist

                cosine_dist = pairwise.cosine_distances(histograms[i].reshape(1, -1),
histograms[j].reshape(1, -1))[0][0]
            distances[(i, j)]['Cosine'] = cosine_dist
    return distances
similarity_measurements = compute_similarity_measurements(histograms)

for key, value in similarity_measurements.items():
    print(f"Images {key}: {value}")
def perform_pca(histograms):
    histograms_normalized = normalize(histograms)
    pca = PCA(n_components=2)
    reduced_data = pca.fit_transform(histograms_normalized)
    plt.scatter(reduced_data[:, 0], reduced_data[:, 1])
    plt.title('PCA of Image Histograms')
    plt.xlabel('Principal Component 1')
    plt.ylabel('Principal Component 2')
    plt.show()
perform_pca(histograms)
```

OUTPUT:

Streaming output truncated to the last 5000 lines.

Images (646, 697): {'Euclidean': 2811.3718, 'Manhattan': 46942.0, 'Cosine': 0.5701544}
Images (646, 698): {'Euclidean': 3086.9768, 'Manhattan': 49076.0, 'Cosine': 0.6130254}
Images (646, 699): {'Euclidean': 2674.8308, 'Manhattan': 47516.0, 'Cosine': 0.52696234}
Images (646, 700): {'Euclidean': 2065.1426, 'Manhattan': 35790.0, 'Cosine': 0.32902163}
Images (646, 701): {'Euclidean': 1997.8348, 'Manhattan': 35080.0, 'Cosine': 0.29984534}
Images (646, 702): {'Euclidean': 2407.4373, 'Manhattan': 41164.0, 'Cosine': 0.38236046}
Images (646, 703): {'Euclidean': 2960.1902, 'Manhattan': 49540.0, 'Cosine': 0.505388}
Images (646, 704): {'Euclidean': 2352.2449, 'Manhattan': 37598.0, 'Cosine': 0.39181828}
Images (646, 705): {'Euclidean': 2388.227, 'Manhattan': 45458.0, 'Cosine': 0.41511792}
Images (646, 706): {'Euclidean': 2022.5815, 'Manhattan': 34176.0, 'Cosine': 0.30610037}
Images (646, 707): {'Euclidean': 2419.6743, 'Manhattan': 42416.0, 'Cosine': 0.46588677}
Images (646, 708): {'Euclidean': 2152.6133, 'Manhattan': 36682.0, 'Cosine': 0.3544513}
Images (646, 709): {'Euclidean': 3870.6355, 'Manhattan': 75798.0, 'Cosine': 0.79939294}
Images (646, 710): {'Euclidean': 2956.3518, 'Manhattan': 49362.0, 'Cosine': 0.58537686}
Images (646, 711): {'Euclidean': 3469.983, 'Manhattan': 50790.0, 'Cosine': 0.6803078}
Images (646, 712): {'Euclidean': 2545.4387, 'Manhattan': 46264.0, 'Cosine': 0.4435978}
Images (646, 713): {'Euclidean': 2563.3638, 'Manhattan': 44104.0, 'Cosine': 0.49194717}
Images (646, 714): {'Euclidean': 2881.096, 'Manhattan': 55832.0, 'Cosine': 0.61459064}

Images (646, 715): {'Euclidean': 2095.3745, 'Manhattan': 31298.0, 'Cosine': 0.32454503}
Images (646, 716): {'Euclidean': 2239.52, 'Manhattan': 36102.0, 'Cosine': 0.3925557}
Images (646, 717): {'Euclidean': 2470.4578, 'Manhattan': 43506.0, 'Cosine': 0.4748404}
Images (646, 718): {'Euclidean': 2580.6855, 'Manhattan': 47940.0, 'Cosine': 0.42773032}
Images (646, 719): {'Euclidean': 2647.9543, 'Manhattan': 44826.0, 'Cosine': 0.48982495}
Images (646, 720): {'Euclidean': 2109.786, 'Manhattan': 38520.0, 'Cosine': 0.33884966}
Images (646, 721): {'Euclidean': 2972.0496, 'Manhattan': 61996.0, 'Cosine': 0.63350385}
Images (646, 722): {'Euclidean': 3092.535, 'Manhattan': 50854.0, 'Cosine': 0.63020897}
Images (646, 723): {'Euclidean': 2659.2876, 'Manhattan': 48622.0, 'Cosine': 0.5401466}
Images (646, 724): {'Euclidean': 3867.2253, 'Manhattan': 60708.0, 'Cosine': 0.6660268}
Images (646, 725): {'Euclidean': 2540.68, 'Manhattan': 41070.0, 'Cosine': 0.48153234}
Images (646, 726): {'Euclidean': 2135.8638, 'Manhattan': 36054.0, 'Cosine': 0.35526013}
Images (646, 727): {'Euclidean': 2280.5227, 'Manhattan': 40130.0, 'Cosine': 0.413095}
Images (646, 728): {'Euclidean': 4034.8608, 'Manhattan': 66830.0, 'Cosine': 0.80802196}
Images (646, 729): {'Euclidean': 2148.0122, 'Manhattan': 34438.0, 'Cosine': 0.354649}
Images (646, 730): {'Euclidean': 2443.528, 'Manhattan': 38628.0, 'Cosine': 0.4676785}
Images (646, 731): {'Euclidean': 3391.1138, 'Manhattan': 52484.0, 'Cosine': 0.6717044}
Images (646, 732): {'Euclidean': 2066.5608, 'Manhattan': 36290.0, 'Cosine': 0.33138645}
Images (646, 733): {'Euclidean': 5656.0923, 'Manhattan': 79396.0, 'Cosine': 0.9335758}
Images (646, 734): {'Euclidean': 3428.6035, 'Manhattan': 55274.0, 'Cosine': 0.7191735}

Images (646, 735): {'Euclidean': 2660.6082, 'Manhattan': 45038.0, 'Cosine': 0.4559487}
Images (646, 736): {'Euclidean': 2649.1794, 'Manhattan': 43756.0, 'Cosine': 0.46607792}
Images (646, 737): {'Euclidean': 2810.6252, 'Manhattan': 49572.0, 'Cosine': 0.5309368}
Images (646, 738): {'Euclidean': 2651.0854, 'Manhattan': 44442.0, 'Cosine': 0.5047672}
Images (646, 739): {'Euclidean': 2482.7126, 'Manhattan': 42278.0, 'Cosine': 0.4680118}
Images (646, 740): {'Euclidean': 2703.7053, 'Manhattan': 48914.0, 'Cosine': 0.503719}
Images (646, 741): {'Euclidean': 5791.429, 'Manhattan': 76426.0, 'Cosine': 0.9186419}
Images (646, 742): {'Euclidean': 2253.0474, 'Manhattan': 36748.0, 'Cosine': 0.3800438}
Images (646, 743): {'Euclidean': 2599.5122, 'Manhattan': 40294.0, 'Cosine': 0.48750114}
Images (646, 744): {'Euclidean': 2348.5962, 'Manhattan': 35462.0, 'Cosine': 0.40578473}
Images (646, 745): {'Euclidean': 2560.6719, 'Manhattan': 45880.0, 'Cosine': 0.5035371}
Images (646, 746): {'Euclidean': 2622.9358, 'Manhattan': 48354.0, 'Cosine': 0.5424799}
Images (647, 648): {'Euclidean': 2005.282, 'Manhattan': 38702.0, 'Cosine': 0.33022046}
Images (647, 649): {'Euclidean': 2600.0776, 'Manhattan': 55166.0, 'Cosine': 0.5686339}
Images (647, 650): {'Euclidean': 3533.3396, 'Manhattan': 55674.0, 'Cosine': 0.5975725}
Images (647, 651): {'Euclidean': 1760.663, 'Manhattan': 34844.0, 'Cosine': 0.24671894}
Images (647, 652): {'Euclidean': 3008.1802, 'Manhattan': 52220.0, 'Cosine': 0.49495667}
Images (647, 653): {'Euclidean': 1918.4952, 'Manhattan': 37302.0, 'Cosine': 0.2506531}
Images (647, 654): {'Euclidean': 2428.9204, 'Manhattan': 41546.0, 'Cosine': 0.2973776}
Images (647, 655): {'Euclidean': 1984.8667, 'Manhattan': 38240.0, 'Cosine': 0.32654643}

Images (647, 656): {'Euclidean': 3959.9895, 'Manhattan': 54068.0, 'Cosine': 0.44240123}
Images (647, 657): {'Euclidean': 1649.2501, 'Manhattan': 33736.0, 'Cosine': 0.21309185}
Images (647, 658): {'Euclidean': 2181.9666, 'Manhattan': 40188.0, 'Cosine': 0.31732398}
Images (647, 659): {'Euclidean': 1734.9017, 'Manhattan': 34326.0, 'Cosine': 0.237329}
Images (647, 660): {'Euclidean': 1875.2323, 'Manhattan': 35328.0, 'Cosine': 0.28543746}
Images (647, 661): {'Euclidean': 1642.0962, 'Manhattan': 27996.0, 'Cosine': 0.19069368}
Images (647, 662): {'Euclidean': 5617.3823, 'Manhattan': 65000.0, 'Cosine': 0.60325956}
Images (647, 663): {'Euclidean': 1783.968, 'Manhattan': 36050.0, 'Cosine': 0.25736243}
Images (647, 664): {'Euclidean': 1901.2927, 'Manhattan': 34850.0, 'Cosine': 0.28793502}
Images (647, 665): {'Euclidean': 2945.7937, 'Manhattan': 65704.0, 'Cosine': 0.6201422}
Images (647, 666): {'Euclidean': 2273.1304, 'Manhattan': 36930.0, 'Cosine': 0.26324767}
Images (647, 667): {'Euclidean': 2414.3926, 'Manhattan': 47706.0, 'Cosine': 0.44574028}
Images (647, 668): {'Euclidean': 2190.8345, 'Manhattan': 45426.0, 'Cosine': 0.3960874}
Images (647, 669): {'Euclidean': 2023.3591, 'Manhattan': 39688.0, 'Cosine': 0.33427233}
Images (647, 670): {'Euclidean': 2234.3933, 'Manhattan': 45524.0, 'Cosine': 0.38081467}
Images (647, 671): {'Euclidean': 1860.9857, 'Manhattan': 31998.0, 'Cosine': 0.2795974}
Images (647, 672): {'Euclidean': 2092.4043, 'Manhattan': 43756.0, 'Cosine': 0.37088162}
Images (647, 673): {'Euclidean': 1651.2892, 'Manhattan': 31030.0, 'Cosine': 0.18784148}
Images (647, 674): {'Euclidean': 2735.2104, 'Manhattan': 45558.0, 'Cosine': 0.4799983}
Images (647, 675): {'Euclidean': 1944.9375, 'Manhattan': 40336.0, 'Cosine': 0.30776155}

Images (647, 676): {'Euclidean': 2255.6453, 'Manhattan': 43314.0, 'Cosine': 0.41709566}
Images (647, 677): {'Euclidean': 2806.8018, 'Manhattan': 58218.0, 'Cosine': 0.5828359}
Images (647, 678): {'Euclidean': 2270.7637, 'Manhattan': 33040.0, 'Cosine': 0.3177355}
Images (647, 679): {'Euclidean': 2994.7727, 'Manhattan': 57796.0, 'Cosine': 0.6299492}
Images (647, 680): {'Euclidean': 2295.1218, 'Manhattan': 44662.0, 'Cosine': 0.43060553}
Images (647, 681): {'Euclidean': 2154.515, 'Manhattan': 34032.0, 'Cosine': 0.2841227}
Images (647, 682): {'Euclidean': 2717.1826, 'Manhattan': 55712.0, 'Cosine': 0.55326235}
Images (647, 683): {'Euclidean': 3716.7773, 'Manhattan': 45006.0, 'Cosine': 0.47956884}
Images (647, 684): {'Euclidean': 1758.8314, 'Manhattan': 35412.0, 'Cosine': 0.24882066}
Images (647, 685): {'Euclidean': 2736.4639, 'Manhattan': 52442.0, 'Cosine': 0.58919114}
Images (647, 686): {'Euclidean': 2303.403, 'Manhattan': 44902.0, 'Cosine': 0.43639123}
Images (647, 687): {'Euclidean': 2940.2344, 'Manhattan': 47392.0, 'Cosine': 0.3630991}
Images (647, 688): {'Euclidean': 2602.8074, 'Manhattan': 51182.0, 'Cosine': 0.50843835}
Images (647, 689): {'Euclidean': 1886.1681, 'Manhattan': 33374.0, 'Cosine': 0.2820109}
Images (647, 690): {'Euclidean': 2130.665, 'Manhattan': 42880.0, 'Cosine': 0.34728813}
Images (647, 691): {'Euclidean': 1668.8416, 'Manhattan': 31418.0, 'Cosine': 0.21705586}
Images (647, 692): {'Euclidean': 2844.6448, 'Manhattan': 49140.0, 'Cosine': 0.60140485}
Images (647, 693): {'Euclidean': 1726.4703, 'Manhattan': 34306.0, 'Cosine': 0.23622435}
Images (647, 694): {'Euclidean': 1745.3235, 'Manhattan': 34084.0, 'Cosine': 0.20041466}
Images (647, 695): {'Euclidean': 1864.9102, 'Manhattan': 29522.0, 'Cosine': 0.2254749}

Images (647, 696): {'Euclidean': 2038.3159, 'Manhattan': 38812.0, 'Cosine': 0.33457184}
Images (647, 697): {'Euclidean': 2420.0107, 'Manhattan': 42724.0, 'Cosine': 0.44348538}
Images (647, 698): {'Euclidean': 1127.6143, 'Manhattan': 20656.0, 'Cosine': 0.08411604}
Images (647, 699): {'Euclidean': 1878.7927, 'Manhattan': 35706.0, 'Cosine': 0.27141088}
Images (647, 700): {'Euclidean': 1917.5735, 'Manhattan': 37618.0, 'Cosine': 0.30384582}
Images (647, 701): {'Euclidean': 2465.5085, 'Manhattan': 50228.0, 'Cosine': 0.50295967}
Images (647, 702): {'Euclidean': 2721.2563, 'Manhattan': 47208.0, 'Cosine': 0.50978833}
Images (647, 703): {'Euclidean': 2715.057, 'Manhattan': 48198.0, 'Cosine': 0.43758708}
Images (647, 704): {'Euclidean': 2120.1245, 'Manhattan': 38656.0, 'Cosine': 0.33484817}
Images (647, 705): {'Euclidean': 2326.1968, 'Manhattan': 48334.0, 'Cosine': 0.4159711}
Images (647, 706): {'Euclidean': 2207.2634, 'Manhattan': 44802.0, 'Cosine': 0.39494812}
Images (647, 707): {'Euclidean': 1840.5483, 'Manhattan': 38104.0, 'Cosine': 0.275774}
Images (647, 708): {'Euclidean': 2464.437, 'Manhattan': 49144.0, 'Cosine': 0.5065484}
Images (647, 709): {'Euclidean': 2438.5586, 'Manhattan': 42180.0, 'Cosine': 0.31366122}
Images (647, 710): {'Euclidean': 2263.7769, 'Manhattan': 37800.0, 'Cosine': 0.35852832}
Images (647, 711): {'Euclidean': 2834.894, 'Manhattan': 44708.0, 'Cosine': 0.4651906}
Images (647, 712): {'Euclidean': 2525.0918, 'Manhattan': 49164.0, 'Cosine': 0.4572336}
Images (647, 713): {'Euclidean': 1805.4363, 'Manhattan': 37478.0, 'Cosine': 0.2538762}
Images (647, 714): {'Euclidean': 1621.5968, 'Manhattan': 31990.0, 'Cosine': 0.20105481}
Images (647, 715): {'Euclidean': 2060.3325, 'Manhattan': 38374.0, 'Cosine': 0.33446574}

Images (647, 716): {'Euclidean': 1611.5695, 'Manhattan': 29984.0, 'Cosine': 0.20396858}
Images (647, 717): {'Euclidean': 1871.267, 'Manhattan': 37468.0, 'Cosine': 0.28193128}
Images (647, 718): {'Euclidean': 3334.5781, 'Manhattan': 67760.0, 'Cosine': 0.74404216}
Images (647, 719): {'Euclidean': 2388.786, 'Manhattan': 42646.0, 'Cosine': 0.41811514}
Images (647, 720): {'Euclidean': 2412.7217, 'Manhattan': 50280.0, 'Cosine': 0.48363978}
Images (647, 721): {'Euclidean': 2842.7612, 'Manhattan': 61502.0, 'Cosine': 0.6084411}
Images (647, 722): {'Euclidean': 2982.3699, 'Manhattan': 51696.0, 'Cosine': 0.6114951}
Images (647, 723): {'Euclidean': 1673.6033, 'Manhattan': 35306.0, 'Cosine': 0.21908414}
Images (647, 724): {'Euclidean': 3969.1309, 'Manhattan': 66658.0, 'Cosine': 0.722657}
Images (647, 725): {'Euclidean': 2096.7336, 'Manhattan': 38078.0, 'Cosine': 0.34444928}
Images (647, 726): {'Euclidean': 1928.9945, 'Manhattan': 37848.0, 'Cosine': 0.30770338}
Images (647, 727): {'Euclidean': 1659.6035, 'Manhattan': 34050.0, 'Cosine': 0.2177468}
Images (647, 728): {'Euclidean': 4172.9316, 'Manhattan': 75254.0, 'Cosine': 0.89416945}
Images (647, 729): {'Euclidean': 1710.6285, 'Manhattan': 30552.0, 'Cosine': 0.23322892}
Images (647, 730): {'Euclidean': 2205.9292, 'Manhattan': 40722.0, 'Cosine': 0.40299547}
Images (647, 731): {'Euclidean': 2045.6525, 'Manhattan': 32238.0, 'Cosine': 0.24711746}
Images (647, 732): {'Euclidean': 1927.8662, 'Manhattan': 38018.0, 'Cosine': 0.3098005}
Images (647, 733): {'Euclidean': 5127.613, 'Manhattan': 63496.0, 'Cosine': 0.74456877}
Images (647, 734): {'Euclidean': 2324.9343, 'Manhattan': 33084.0, 'Cosine': 0.34103608}
Images (647, 735): {'Euclidean': 2435.9, 'Manhattan': 44056.0, 'Cosine': 0.39768958}

Images (647, 736): {'Euclidean': 2203.185, 'Manhattan': 40282.0, 'Cosine': 0.33649987}
Images (647, 737): {'Euclidean': 2155.4016, 'Manhattan': 37700.0, 'Cosine': 0.32629704}
Images (647, 738): {'Euclidean': 1803.7456, 'Manhattan': 32874.0, 'Cosine': 0.24443221}
Images (647, 739): {'Euclidean': 1030.8569, 'Manhattan': 21330.0, 'Cosine': 0.075351596}
Images (647, 740): {'Euclidean': 2178.716, 'Manhattan': 38520.0, 'Cosine': 0.34256685}
Images (647, 741): {'Euclidean': 5253.444, 'Manhattan': 61634.0, 'Cosine': 0.7273202}
Images (647, 742): {'Euclidean': 2107.0535, 'Manhattan': 38230.0, 'Cosine': 0.35259795}
Images (647, 743): {'Euclidean': 1555.4446, 'Manhattan': 29474.0, 'Cosine': 0.18178308}
Images (647, 744): {'Euclidean': 1802.4767, 'Manhattan': 32144.0, 'Cosine': 0.2505052}
Images (647, 745): {'Euclidean': 1906.2167, 'Manhattan': 37592.0, 'Cosine': 0.289783}
Images (647, 746): {'Euclidean': 1553.4967, 'Manhattan': 28680.0, 'Cosine': 0.18802762}
Images (648, 649): {'Euclidean': 1925.4033, 'Manhattan': 40116.0, 'Cosine': 0.4069891}
Images (648, 650): {'Euclidean': 3614.5122, 'Manhattan': 63172.0, 'Cosine': 0.70038277}
Images (648, 651): {'Euclidean': 1474.213, 'Manhattan': 24776.0, 'Cosine': 0.23277551}
Images (648, 652): {'Euclidean': 2845.4775, 'Manhattan': 55336.0, 'Cosine': 0.48326898}
Images (648, 653): {'Euclidean': 2369.158, 'Manhattan': 41820.0, 'Cosine': 0.446436}
Images (648, 654): {'Euclidean': 2868.449, 'Manhattan': 54910.0, 'Cosine': 0.45984387}
Images (648, 655): {'Euclidean': 1326.7555, 'Manhattan': 25052.0, 'Cosine': 0.20128828}
Images (648, 656): {'Euclidean': 4486.6865, 'Manhattan': 69752.0, 'Cosine': 0.6478324}
Images (648, 657): {'Euclidean': 1305.1039, 'Manhattan': 24938.0, 'Cosine': 0.20261759}

Images (648, 658): {'Euclidean': 2492.7734, 'Manhattan': 47542.0, 'Cosine': 0.48239887}
Images (648, 659): {'Euclidean': 1687.11, 'Manhattan': 35520.0, 'Cosine': 0.29697257}
Images (648, 660): {'Euclidean': 1440.6492, 'Manhattan': 27744.0, 'Cosine': 0.23007768}
Images (648, 661): {'Euclidean': 2116.2751, 'Manhattan': 38230.0, 'Cosine': 0.3718801}
Images (648, 662): {'Euclidean': 5698.3247, 'Manhattan': 69764.0, 'Cosine': 0.63714445}
Images (648, 663): {'Euclidean': 1312.8953, 'Manhattan': 29906.0, 'Cosine': 0.19730753}
Images (648, 664): {'Euclidean': 1579.743, 'Manhattan': 31550.0, 'Cosine': 0.2601695}
Images (648, 665): {'Euclidean': 2454.6882, 'Manhattan': 55784.0, 'Cosine': 0.51709026}
Images (648, 666): {'Euclidean': 2792.0537, 'Manhattan': 53992.0, 'Cosine': 0.4423579}
Images (648, 667): {'Euclidean': 2187.73, 'Manhattan': 44098.0, 'Cosine': 0.453525}
Images (648, 668): {'Euclidean': 1683.5278, 'Manhattan': 31904.0, 'Cosine': 0.30975902}
Images (648, 669): {'Euclidean': 1630.9335, 'Manhattan': 29480.0, 'Cosine': 0.29010248}
Images (648, 670): {'Euclidean': 1809.6403, 'Manhattan': 34474.0, 'Cosine': 0.30674422}
Images (648, 671): {'Euclidean': 1433.3192, 'Manhattan': 27722.0, 'Cosine': 0.22458076}
Images (648, 672): {'Euclidean': 1532.2631, 'Manhattan': 30440.0, 'Cosine': 0.27609706}
Images (648, 673): {'Euclidean': 2129.1768, 'Manhattan': 42030.0, 'Cosine': 0.36093372}
Images (648, 674): {'Euclidean': 2195.6838, 'Manhattan': 40876.0, 'Cosine': 0.34131157}
Images (648, 675): {'Euclidean': 1873.8218, 'Manhattan': 37076.0, 'Cosine': 0.3849331}
Images (648, 676): {'Euclidean': 1819.8539, 'Manhattan': 32450.0, 'Cosine': 0.35529482}
Images (648, 677): {'Euclidean': 1810.67, 'Manhattan': 38834.0, 'Cosine': 0.28902256}

Images (648, 678): {'Euclidean': 2488.1973, 'Manhattan': 46182.0, 'Cosine': 0.42950976}
Images (648, 679): {'Euclidean': 2205.869, 'Manhattan': 45880.0, 'Cosine': 0.40172923}
Images (648, 680): {'Euclidean': 1861.543, 'Manhattan': 39544.0, 'Cosine': 0.36881888}
Images (648, 681): {'Euclidean': 2482.3137, 'Manhattan': 47282.0, 'Cosine': 0.42464578}
Images (648, 682): {'Euclidean': 2041.1428, 'Manhattan': 40230.0, 'Cosine': 0.37962848}
Images (648, 683): {'Euclidean': 4162.154, 'Manhattan': 50960.0, 'Cosine': 0.67891073}
Images (648, 684): {'Euclidean': 1404.5476, 'Manhattan': 28426.0, 'Cosine': 0.22291547}
Images (648, 685): {'Euclidean': 2074.0034, 'Manhattan': 41130.0, 'Cosine': 0.42412823}
Images (648, 686): {'Euclidean': 1800.1466, 'Manhattan': 36656.0, 'Cosine': 0.348571}
Images (648, 687): {'Euclidean': 3518.909, 'Manhattan': 60724.0, 'Cosine': 0.59283006}
Images (648, 688): {'Euclidean': 2132.9219, 'Manhattan': 41688.0, 'Cosine': 0.41700667}
Images (648, 689): {'Euclidean': 1709.9906, 'Manhattan': 35702.0, 'Cosine': 0.30192626}
Images (648, 690): {'Euclidean': 1612.0477, 'Manhattan': 29842.0, 'Cosine': 0.24344742}
Images (648, 691): {'Euclidean': 1544.9912, 'Manhattan': 31006.0, 'Cosine': 0.24224937}
Images (648, 692): {'Euclidean': 2265.1985, 'Manhattan': 38312.0, 'Cosine': 0.46447736}
Images (648, 693): {'Euclidean': 1735.7943, 'Manhattan': 35018.0, 'Cosine': 0.32128924}
Images (648, 694): {'Euclidean': 2365.3684, 'Manhattan': 47598.0, 'Cosine': 0.42431676}
Images (648, 695): {'Euclidean': 2554.4993, 'Manhattan': 43920.0, 'Cosine': 0.49161243}
Images (648, 696): {'Euclidean': 1733.2241, 'Manhattan': 35334.0, 'Cosine': 0.31664658}
Images (648, 697): {'Euclidean': 1471.3151, 'Manhattan': 28180.0, 'Cosine': 0.19575381}

Images (648, 698): {'Euclidean': 2086.1606, 'Manhattan': 41136.0, 'Cosine': 0.3279736}
Images (648, 699): {'Euclidean': 1653.3427, 'Manhattan': 31454.0, 'Cosine': 0.26178837}
Images (648, 700): {'Euclidean': 1366.0066, 'Manhattan': 26498.0, 'Cosine': 0.21675712}
Images (648, 701): {'Euclidean': 1771.9092, 'Manhattan': 38646.0, 'Cosine': 0.3379004}
Images (648, 702): {'Euclidean': 2532.7917, 'Manhattan': 51158.0, 'Cosine': 0.52250516}
Images (648, 703): {'Euclidean': 2686.296, 'Manhattan': 54818.0, 'Cosine': 0.47968698}
Images (648, 704): {'Euclidean': 1960.9615, 'Manhattan': 40558.0, 'Cosine': 0.34707093}
Images (648, 705): {'Euclidean': 1876.4978, 'Manhattan': 39302.0, 'Cosine': 0.33478427}
Images (648, 706): {'Euclidean': 1796.1442, 'Manhattan': 38758.0, 'Cosine': 0.33991313}
Images (648, 707): {'Euclidean': 1411.582, 'Manhattan': 25132.0, 'Cosine': 0.22673738}
Images (648, 708): {'Euclidean': 1493.2843, 'Manhattan': 29848.0, 'Cosine': 0.24320287}
Images (648, 709): {'Euclidean': 2991.676, 'Manhattan': 62630.0, 'Cosine': 0.5366725}
Images (648, 710): {'Euclidean': 2080.8186, 'Manhattan': 38866.0, 'Cosine': 0.3521254}
Images (648, 711): {'Euclidean': 2749.549, 'Manhattan': 39630.0, 'Cosine': 0.4866066}
Images (648, 712): {'Euclidean': 1953.2578, 'Manhattan': 34624.0, 'Cosine': 0.32099217}
Images (648, 713): {'Euclidean': 1684.3729, 'Manhattan': 34012.0, 'Cosine': 0.28224993}
Images (648, 714): {'Euclidean': 1868.8846, 'Manhattan': 39700.0, 'Cosine': 0.33792198}
Images (648, 715): {'Euclidean': 1437.271, 'Manhattan': 28284.0, 'Cosine': 0.20603645}
Images (648, 716): {'Euclidean': 1106.434, 'Manhattan': 21142.0, 'Cosine': 0.13897228}
Images (648, 717): {'Euclidean': 1405.5312, 'Manhattan': 25908.0, 'Cosine': 0.21347547}

Images (648, 718): {'Euclidean': 2650.17, 'Manhattan': 55380.0, 'Cosine': 0.5515017}
Images (648, 719): {'Euclidean': 1881.9862, 'Manhattan': 31974.0, 'Cosine': 0.30846924}
Images (648, 720): {'Euclidean': 1454.1603, 'Manhattan': 27036.0, 'Cosine': 0.22981316}
Images (648, 721): {'Euclidean': 2325.0962, 'Manhattan': 52164.0, 'Cosine': 0.49985862}
Images (648, 722): {'Euclidean': 2601.7183, 'Manhattan': 43804.0, 'Cosine': 0.5519189}
Images (648, 723): {'Euclidean': 1709.741, 'Manhattan': 31816.0, 'Cosine': 0.3008989}
Images (648, 724): {'Euclidean': 3304.421, 'Manhattan': 52000.0, 'Cosine': 0.51138467}
Images (648, 725): {'Euclidean': 1680.5922, 'Manhattan': 33850.0, 'Cosine': 0.2792259}
Images (648, 726): {'Euclidean': 1355.3353, 'Manhattan': 25524.0, 'Cosine': 0.21277744}
Images (648, 727): {'Euclidean': 1224.9114, 'Manhattan': 26438.0, 'Cosine': 0.17446709}
Images (648, 728): {'Euclidean': 3261.008, 'Manhattan': 57302.0, 'Cosine': 0.5852137}
Images (648, 729): {'Euclidean': 1317.3359, 'Manhattan': 25258.0, 'Cosine': 0.19219261}
Images (648, 730): {'Euclidean': 1296.7614, 'Manhattan': 22692.0, 'Cosine': 0.18478727}
Images (648, 731): {'Euclidean': 2339.524, 'Manhattan': 42210.0, 'Cosine': 0.353608}
Images (648, 732): {'Euclidean': 1170.9902, 'Manhattan': 21172.0, 'Cosine': 0.16306895}
Images (648, 733): {'Euclidean': 4956.2827, 'Manhattan': 65264.0, 'Cosine': 0.7188952}
Images (648, 734): {'Euclidean': 2501.2654, 'Manhattan': 39812.0, 'Cosine': 0.44886786}
Images (648, 735): {'Euclidean': 2321.5205, 'Manhattan': 46414.0, 'Cosine': 0.41623068}
Images (648, 736): {'Euclidean': 2116.2249, 'Manhattan': 43864.0, 'Cosine': 0.35959518}
Images (648, 737): {'Euclidean': 2075.8896, 'Manhattan': 43254.0, 'Cosine': 0.35268927}

Images (648, 738): {'Euclidean': 1526.2084, 'Manhattan': 30900.0, 'Cosine': 0.21012318}
Images (648, 739): {'Euclidean': 1494.7201, 'Manhattan': 29548.0, 'Cosine': 0.22924566}
Images (648, 740): {'Euclidean': 2110.0493, 'Manhattan': 40962.0, 'Cosine': 0.38247263}
Images (648, 741): {'Euclidean': 5112.394, 'Manhattan': 60628.0, 'Cosine': 0.70758134}
Images (648, 742): {'Euclidean': 1651.2734, 'Manhattan': 32360.0, 'Cosine': 0.27711427}
Images (648, 743): {'Euclidean': 1851.023, 'Manhattan': 35982.0, 'Cosine': 0.31772816}
Images (648, 744): {'Euclidean': 1731.7373, 'Manhattan': 35554.0, 'Cosine': 0.2911452}
Images (648, 745): {'Euclidean': 1560.4878, 'Manhattan': 30142.0, 'Cosine': 0.25448358}
Images (648, 746): {'Euclidean': 1353.8826, 'Manhattan': 27276.0, 'Cosine': 0.2003935}
Images (649, 650): {'Euclidean': 2956.9827, 'Manhattan': 55290.0, 'Cosine': 0.41986668}
Images (649, 651): {'Euclidean': 2060.676, 'Manhattan': 42462.0, 'Cosine': 0.45070744}
Images (649, 652): {'Euclidean': 3207.738, 'Manhattan': 64500.0, 'Cosine': 0.63731956}
Images (649, 653): {'Euclidean': 2717.966, 'Manhattan': 49918.0, 'Cosine': 0.5942613}
Images (649, 654): {'Euclidean': 3552.752, 'Manhattan': 70184.0, 'Cosine': 0.7608482}
Images (649, 655): {'Euclidean': 1873.597, 'Manhattan': 40800.0, 'Cosine': 0.39715314}
Images (649, 656): {'Euclidean': 4858.274, 'Manhattan': 78126.0, 'Cosine': 0.81642896}
Images (649, 657): {'Euclidean': 1587.4004, 'Manhattan': 33558.0, 'Cosine': 0.29728162}
Images (649, 658): {'Euclidean': 2615.969, 'Manhattan': 54582.0, 'Cosine': 0.53202915}
Images (649, 659): {'Euclidean': 1822.9224, 'Manhattan': 40828.0, 'Cosine': 0.34375548}
Images (649, 660): {'Euclidean': 1707.2926, 'Manhattan': 37138.0, 'Cosine': 0.3195789}

Images (649, 661): {'Euclidean': 2458.3308, 'Manhattan': 47478.0, 'Cosine': 0.50724566}
Images (649, 662): {'Euclidean': 6242.694, 'Manhattan': 80678.0, 'Cosine': 0.88504344}
Images (649, 663): {'Euclidean': 1400.2264, 'Manhattan': 29976.0, 'Cosine': 0.22164059}
Images (649, 664): {'Euclidean': 1937.6223, 'Manhattan': 42928.0, 'Cosine': 0.38849688}
Images (649, 665): {'Euclidean': 1555.8271, 'Manhattan': 28366.0, 'Cosine': 0.19392693}
Images (649, 666): {'Euclidean': 3535.3284, 'Manhattan': 71412.0, 'Cosine': 0.7686866}
Images (649, 667): {'Euclidean': 1842.9661, 'Manhattan': 34062.0, 'Cosine': 0.3169763}
Images (649, 668): {'Euclidean': 1599.263, 'Manhattan': 26290.0, 'Cosine': 0.27658904}
Images (649, 669): {'Euclidean': 1901.0686, 'Manhattan': 42110.0, 'Cosine': 0.39006513}
Images (649, 670): {'Euclidean': 2255.317, 'Manhattan': 47634.0, 'Cosine': 0.47790754}
Images (649, 671): {'Euclidean': 1828.9539, 'Manhattan': 41356.0, 'Cosine': 0.36187625}
Images (649, 672): {'Euclidean': 1361.3523, 'Manhattan': 29124.0, 'Cosine': 0.21433353}
Images (649, 673): {'Euclidean': 2847.6785, 'Manhattan': 59718.0, 'Cosine': 0.66535497}
Images (649, 674): {'Euclidean': 2805.331, 'Manhattan': 54874.0, 'Cosine': 0.58506656}
Images (649, 675): {'Euclidean': 1620.9775, 'Manhattan': 33532.0, 'Cosine': 0.28499067}
Images (649, 676): {'Euclidean': 1907.071, 'Manhattan': 36666.0, 'Cosine': 0.38627732}
Images (649, 677): {'Euclidean': 1785.7565, 'Manhattan': 38344.0, 'Cosine': 0.27936143}
Images (649, 678): {'Euclidean': 3079.2544, 'Manhattan': 62578.0, 'Cosine': 0.6847265}
Images (649, 679): {'Euclidean': 2409.7534, 'Manhattan': 45906.0, 'Cosine': 0.4815563}
Images (649, 680): {'Euclidean': 1578.7096, 'Manhattan': 34312.0, 'Cosine': 0.26252925}

Images (649, 681): {'Euclidean': 2992.234, 'Manhattan': 64696.0, 'Cosine': 0.6403272}
Images (649, 682): {'Euclidean': 1516.974, 'Manhattan': 28206.0, 'Cosine': 0.2031002}
Images (649, 683): {'Euclidean': 4361.542, 'Manhattan': 56706.0, 'Cosine': 0.76859105}
Images (649, 684): {'Euclidean': 1794.8047, 'Manhattan': 40226.0, 'Cosine': 0.35989702}
Images (649, 685): {'Euclidean': 1180.2762, 'Manhattan': 23912.0, 'Cosine': 0.13317549}
Images (649, 686): {'Euclidean': 1180.7566, 'Manhattan': 23756.0, 'Cosine': 0.1483491}
Images (649, 687): {'Euclidean': 3692.285, 'Manhattan': 61456.0, 'Cosine': 0.6678078}
Images (649, 688): {'Euclidean': 1528.8263, 'Manhattan': 27112.0, 'Cosine': 0.20724726}
Images (649, 689): {'Euclidean': 1606.6101, 'Manhattan': 32938.0, 'Cosine': 0.26401508}
Images (649, 690): {'Euclidean': 2141.7156, 'Manhattan': 42364.0, 'Cosine': 0.43311727}
Images (649, 691): {'Euclidean': 2175.384, 'Manhattan': 49370.0, 'Cosine': 0.47871983}
Images (649, 692): {'Euclidean': 2128.5525, 'Manhattan': 35532.0, 'Cosine': 0.40562463}
Images (649, 693): {'Euclidean': 1899.2993, 'Manhattan': 42420.0, 'Cosine': 0.38098347}
Images (649, 694): {'Euclidean': 2589.7405, 'Manhattan': 52612.0, 'Cosine': 0.5139582}
Images (649, 695): {'Euclidean': 2871.4746, 'Manhattan': 57408.0, 'Cosine': 0.6290007}
Images (649, 696): {'Euclidean': 1805.1521, 'Manhattan': 37582.0, 'Cosine': 0.34029806}
Images (649, 697): {'Euclidean': 2236.102, 'Manhattan': 36382.0, 'Cosine': 0.4631946}
Images (649, 698): {'Euclidean': 2846.009, 'Manhattan': 58878.0, 'Cosine': 0.6391078}
Images (649, 699): {'Euclidean': 2223.1562, 'Manhattan': 48798.0, 'Cosine': 0.47516692}
Images (649, 700): {'Euclidean': 1262.7621, 'Manhattan': 26862.0, 'Cosine': 0.18239379}

Images (649, 701): {'Euclidean': 1189.5848, 'Manhattan': 25000.0, 'Cosine': 0.15066767}
Images (649, 702): {'Euclidean': 1791.8125, 'Manhattan': 36522.0, 'Cosine': 0.24540293}
Images (649, 703): {'Euclidean': 2995.2368, 'Manhattan': 64624.0, 'Cosine': 0.610334}
Images (649, 704): {'Euclidean': 1951.8094, 'Manhattan': 40904.0, 'Cosine': 0.3416012}
Images (649, 705): {'Euclidean': 1963.2972, 'Manhattan': 42082.0, 'Cosine': 0.36455894}
Images (649, 706): {'Euclidean': 1296.3857, 'Manhattan': 25488.0, 'Cosine': 0.17505056}
Images (649, 707): {'Euclidean': 1690.0698, 'Manhattan': 32364.0, 'Cosine': 0.3213011}
Images (649, 708): {'Euclidean': 1297.3164, 'Manhattan': 25028.0, 'Cosine': 0.18165785}
Images (649, 709): {'Euclidean': 3488.8574, 'Manhattan': 76638.0, 'Cosine': 0.76155245}
Images (649, 710): {'Euclidean': 2693.4143, 'Manhattan': 56092.0, 'Cosine': 0.6043656}
Images (649, 711): {'Euclidean': 3075.79, 'Manhattan': 48198.0, 'Cosine': 0.6254461}
Images (649, 712): {'Euclidean': 2056.7678, 'Manhattan': 42250.0, 'Cosine': 0.35645175}
Images (649, 713): {'Euclidean': 1713.7812, 'Manhattan': 30558.0, 'Cosine': 0.29009616}
Images (649, 714): {'Euclidean': 2197.9668, 'Manhattan': 48604.0, 'Cosine': 0.4660927}
Images (649, 715): {'Euclidean': 1941.9619, 'Manhattan': 41942.0, 'Cosine': 0.3761046}
Images (649, 716): {'Euclidean': 1679.1951, 'Manhattan': 37056.0, 'Cosine': 0.31668317}
Images (649, 717): {'Euclidean': 2049.7341, 'Manhattan': 45416.0, 'Cosine': 0.4496554}
Images (649, 718): {'Euclidean': 1789.1881, 'Manhattan': 37320.0, 'Cosine': 0.23029196}
Images (649, 719): {'Euclidean': 2206.3862, 'Manhattan': 43330.0, 'Cosine': 0.4276898}
Images (649, 720): {'Euclidean': 1326.0958, 'Manhattan': 28746.0, 'Cosine': 0.18916988}

Images (649, 721): {'Euclidean': 1801.1274, 'Manhattan': 36060.0, 'Cosine': 0.29328513}
Images (649, 722): {'Euclidean': 2077.0315, 'Manhattan': 31342.0, 'Cosine': 0.3388431}
Images (649, 723): {'Euclidean': 2199.9424, 'Manhattan': 46962.0, 'Cosine': 0.49492246}
Images (649, 724): {'Euclidean': 3487.1606, 'Manhattan': 55146.0, 'Cosine': 0.58718836}
Images (649, 725): {'Euclidean': 2126.7444, 'Manhattan': 46082.0, 'Cosine': 0.44616616}
Images (649, 726): {'Euclidean': 1252.4232, 'Manhattan': 27408.0, 'Cosine': 0.17895925}
Images (649, 727): {'Euclidean': 1486.6385, 'Manhattan': 31546.0, 'Cosine': 0.25423568}
Images (649, 728): {'Euclidean': 3368.346, 'Manhattan': 48248.0, 'Cosine': 0.6303845}
Images (649, 729): {'Euclidean': 1758.5597, 'Manhattan': 40374.0, 'Cosine': 0.33875096}
Images (649, 730): {'Euclidean': 1958.7052, 'Manhattan': 37748.0, 'Cosine': 0.41712654}
Images (649, 731): {'Euclidean': 3107.3599, 'Manhattan': 56952.0, 'Cosine': 0.6715207}
Images (649, 732): {'Euclidean': 1430.5614, 'Manhattan': 30830.0, 'Cosine': 0.24189168}
Images (649, 733): {'Euclidean': 5310.649, 'Manhattan': 76882.0, 'Cosine': 0.88317275}
Images (649, 734): {'Euclidean': 3008.4563, 'Manhattan': 56038.0, 'Cosine': 0.66875714}
Images (649, 735): {'Euclidean': 2575.6711, 'Manhattan': 55534.0, 'Cosine': 0.51806474}
Images (649, 736): {'Euclidean': 2590.2205, 'Manhattan': 57238.0, 'Cosine': 0.5499157}
Images (649, 737): {'Euclidean': 2662.1145, 'Manhattan': 61538.0, 'Cosine': 0.5930487}
Images (649, 738): {'Euclidean': 2426.9727, 'Manhattan': 54580.0, 'Cosine': 0.54482234}
Images (649, 739): {'Euclidean': 2063.2737, 'Manhattan': 45822.0, 'Cosine': 0.434838}
Images (649, 740): {'Euclidean': 2463.538, 'Manhattan': 54366.0, 'Cosine': 0.5251088}

Images (649, 741): {'Euclidean': 5492.4595, 'Manhattan': 72482.0, 'Cosine': 0.88318425}
Images (649, 742): {'Euclidean': 1973.2233, 'Manhattan': 44050.0, 'Cosine': 0.39343917}
Images (649, 743): {'Euclidean': 2412.3083, 'Manhattan': 50392.0, 'Cosine': 0.54263866}
Images (649, 744): {'Euclidean': 2188.0576, 'Manhattan': 48562.0, 'Cosine': 0.4645927}
Images (649, 745): {'Euclidean': 1935.7118, 'Manhattan': 43244.0, 'Cosine': 0.38864422}
Images (649, 746): {'Euclidean': 2099.4219, 'Manhattan': 46302.0, 'Cosine': 0.47688437}
Images (650, 651): {'Euclidean': 3581.398, 'Manhattan': 61302.0, 'Cosine': 0.6757497}
Images (650, 652): {'Euclidean': 4541.8037, 'Manhattan': 76378.0, 'Cosine': 0.8596072}
Images (650, 653): {'Euclidean': 3604.081, 'Manhattan': 53046.0, 'Cosine': 0.6114893}
Images (650, 654): {'Euclidean': 4241.8477, 'Manhattan': 62962.0, 'Cosine': 0.72857404}
Images (650, 655): {'Euclidean': 3680.6091, 'Manhattan': 67816.0, 'Cosine': 0.74175227}
Images (650, 656): {'Euclidean': 5048.5537, 'Manhattan': 68872.0, 'Cosine': 0.69838107}
Images (650, 657): {'Euclidean': 3194.7341, 'Manhattan': 54860.0, 'Cosine': 0.52177787}
Images (650, 658): {'Euclidean': 3707.1296, 'Manhattan': 56856.0, 'Cosine': 0.6412393}
Images (650, 659): {'Euclidean': 3284.4744, 'Manhattan': 58204.0, 'Cosine': 0.54412156}
Images (650, 660): {'Euclidean': 3503.1335, 'Manhattan': 61954.0, 'Cosine': 0.6486214}
Images (650, 661): {'Euclidean': 3616.873, 'Manhattan': 60830.0, 'Cosine': 0.6291589}
Images (650, 662): {'Euclidean': 6555.8706, 'Manhattan': 77212.0, 'Cosine': 0.82921046}
Images (650, 663): {'Euclidean': 3226.8198, 'Manhattan': 57614.0, 'Cosine': 0.53177214}
Images (650, 664): {'Euclidean': 3689.9167, 'Manhattan': 65874.0, 'Cosine': 0.71753496}

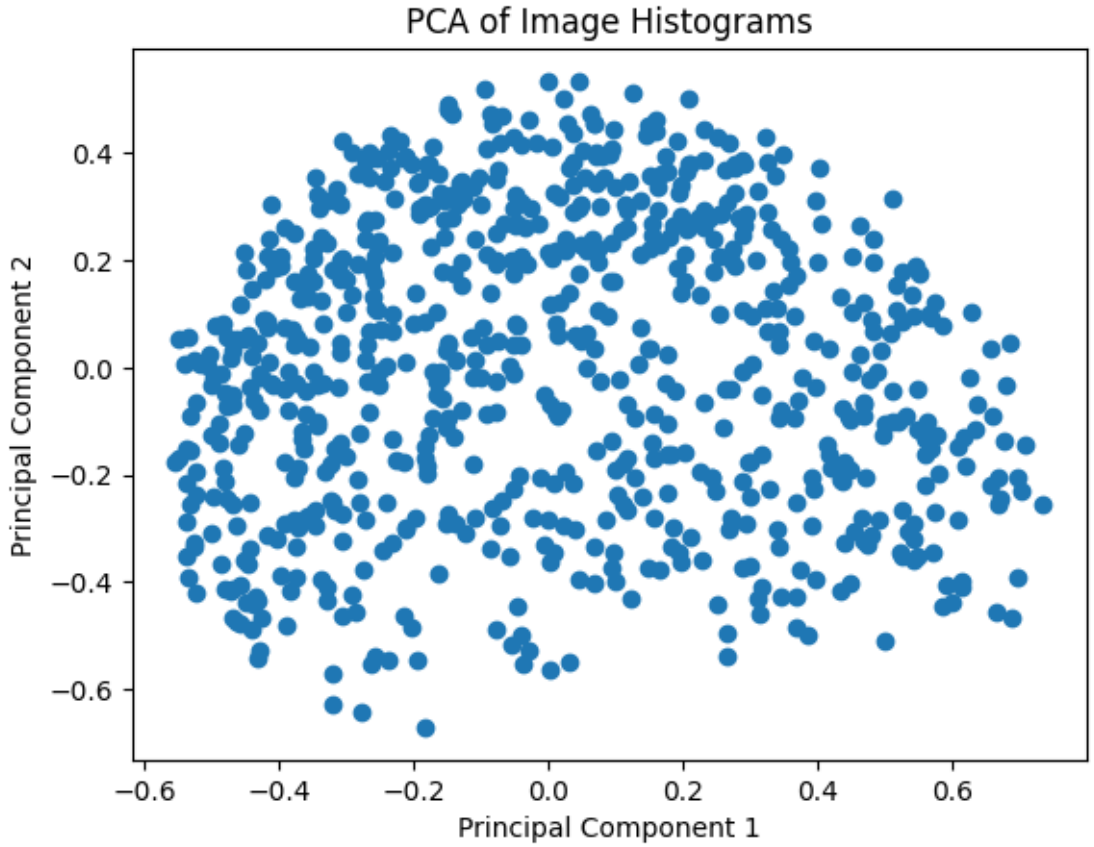
Images (650, 665): {'Euclidean': 3088.3652, 'Manhattan': 55822.0, 'Cosine': 0.44725722}
Images (650, 666): {'Euclidean': 4177.1133, 'Manhattan': 61060.0, 'Cosine': 0.7142697}
Images (650, 667): {'Euclidean': 3085.5803, 'Manhattan': 51616.0, 'Cosine': 0.45776707}
Images (650, 668): {'Euclidean': 3338.5994, 'Manhattan': 58766.0, 'Cosine': 0.57338834}
Images (650, 669): {'Euclidean': 3522.976, 'Manhattan': 61940.0, 'Cosine': 0.65377384}
Images (650, 670): {'Euclidean': 3694.9111, 'Manhattan': 63246.0, 'Cosine': 0.69098747}
Images (650, 671): {'Euclidean': 3655.129, 'Manhattan': 63438.0, 'Cosine': 0.7150046}
Images (650, 672): {'Euclidean': 3075.258, 'Manhattan': 53106.0, 'Cosine': 0.46844685}
Images (650, 673): {'Euclidean': 3769.4607, 'Manhattan': 57734.0, 'Cosine': 0.6770822}
Images (650, 674): {'Euclidean': 4387.714, 'Manhattan': 77532.0, 'Cosine': 0.88792825}
Images (650, 675): {'Euclidean': 2922.0547, 'Manhattan': 48692.0, 'Cosine': 0.4066869}
Images (650, 676): {'Euclidean': 3202.9644, 'Manhattan': 52390.0, 'Cosine': 0.5146738}
Images (650, 677): {'Euclidean': 3971.4265, 'Manhattan': 75358.0, 'Cosine': 0.7920069}
Images (650, 678): {'Euclidean': 4121.423, 'Manhattan': 64762.0, 'Cosine': 0.76434565}
Images (650, 679): {'Euclidean': 3877.7195, 'Manhattan': 71142.0, 'Cosine': 0.72737}
Images (650, 680): {'Euclidean': 3186.7417, 'Manhattan': 57952.0, 'Cosine': 0.50733227}
Images (650, 681): {'Euclidean': 4124.5503, 'Manhattan': 66426.0, 'Cosine': 0.76373315}
Images (650, 682): {'Euclidean': 3315.3887, 'Manhattan': 58488.0, 'Cosine': 0.5354399}
Images (650, 683): {'Euclidean': 5134.442, 'Manhattan': 71388.0, 'Cosine': 0.82393515}
Images (650, 684): {'Euclidean': 3297.9114, 'Manhattan': 56394.0, 'Cosine': 0.56092966}

Images (650, 685): {'Euclidean': 3560.0356, 'Manhattan': 63614.0, 'Cosine': 0.64577264}
Images (650, 686): {'Euclidean': 3374.5366, 'Manhattan': 60464.0, 'Cosine': 0.5860319}
Images (650, 687): {'Euclidean': 4336.4263, 'Manhattan': 67438.0, 'Cosine': 0.6803801}
Images (650, 688): {'Euclidean': 2948.678, 'Manhattan': 52848.0, 'Cosine': 0.4089765}
Images (650, 689): {'Euclidean': 3532.465, 'Manhattan': 62812.0, 'Cosine': 0.64546144}
Images (650, 690): {'Euclidean': 3758.3425, 'Manhattan': 65730.0, 'Cosine': 0.7197198}
Images (650, 691): {'Euclidean': 3573.346, 'Manhattan': 60984.0, 'Cosine': 0.65964}
Images (650, 692): {'Euclidean': 3593.5056, 'Manhattan': 60090.0, 'Cosine': 0.63826036}
Images (650, 693): {'Euclidean': 3115.8462, 'Manhattan': 51354.0, 'Cosine': 0.47964227}
Images (650, 694): {'Euclidean': 3676.0332, 'Manhattan': 59612.0, 'Cosine': 0.62669075}
Images (650, 695): {'Euclidean': 3680.1228, 'Manhattan': 53780.0, 'Cosine': 0.6231716}
Images (650, 696): {'Euclidean': 3428.7627, 'Manhattan': 59868.0, 'Cosine': 0.6053996}
Images (650, 697): {'Euclidean': 3670.108, 'Manhattan': 62070.0, 'Cosine': 0.6769089}
Images (650, 698): {'Euclidean': 3754.2305, 'Manhattan': 60238.0, 'Cosine': 0.6597406}
Images (650, 699): {'Euclidean': 3739.9265, 'Manhattan': 63066.0, 'Cosine': 0.7169063}
Images (650, 700): {'Euclidean': 3146.2034, 'Manhattan': 56172.0, 'Cosine': 0.4983334}
Images (650, 701): {'Euclidean': 3230.6353, 'Manhattan': 62610.0, 'Cosine': 0.5262445}
Images (650, 702): {'Euclidean': 3696.1433, 'Manhattan': 60986.0, 'Cosine': 0.6488001}
Images (650, 703): {'Euclidean': 4382.6084, 'Manhattan': 76934.0, 'Cosine': 0.8438587}
Images (650, 704): {'Euclidean': 3782.5713, 'Manhattan': 66052.0, 'Cosine': 0.716524}

Images (650, 705): {'Euclidean': 3768.7976, 'Manhattan': 72218.0, 'Cosine': 0.72562385}
Images (650, 706): {'Euclidean': 3367.5144, 'Manhattan': 61952.0, 'Cosine': 0.57954055}
Images (650, 707): {'Euclidean': 3258.461, 'Manhattan': 56434.0, 'Cosine': 0.54472864}
Images (650, 708): {'Euclidean': 3530.004, 'Manhattan': 66350.0, 'Cosine': 0.65696174}
Images (650, 709): {'Euclidean': 4168.066, 'Manhattan': 64616.0, 'Cosine': 0.71870667}
Images (650, 710): {'Euclidean': 4019.4622, 'Manhattan': 67694.0, 'Cosine': 0.7829925}
Images (650, 711): {'Euclidean': 4059.2744, 'Manhattan': 64726.0, 'Cosine': 0.7105556}
Images (650, 712): {'Euclidean': 3944.0393, 'Manhattan': 66296.0, 'Cosine': 0.7643397}
Images (650, 713): {'Euclidean': 3245.4731, 'Manhattan': 54006.0, 'Cosine': 0.5226608}
Images (650, 714): {'Euclidean': 3200.2559, 'Manhattan': 48818.0, 'Cosine': 0.5018724}
Images (650, 715): {'Euclidean': 3892.5134, 'Manhattan': 71332.0, 'Cosine': 0.7980249}
Images (650, 716): {'Euclidean': 3483.3596, 'Manhattan': 61420.0, 'Cosine': 0.6453843}
Images (650, 717): {'Euclidean': 3600.9792, 'Manhattan': 61954.0, 'Cosine': 0.68687606}
Images (650, 718): {'Euclidean': 3417.513, 'Manhattan': 61230.0, 'Cosine': 0.54037046}
Images (650, 719): {'Euclidean': 3829.1934, 'Manhattan': 65182.0, 'Cosine': 0.72745216}
Images (650, 720): {'Euclidean': 3507.5344, 'Manhattan': 66538.0, 'Cosine': 0.64608926}
Images (650, 721): {'Euclidean': 3065.5322, 'Manhattan': 52846.0, 'Cosine': 0.4481858}
Images (650, 722): {'Euclidean': 3298.7017, 'Manhattan': 52972.0, 'Cosine': 0.50804067}
Images (650, 723): {'Euclidean': 3326.421, 'Manhattan': 53560.0, 'Cosine': 0.5590743}
Images (650, 724): {'Euclidean': 5009.569, 'Manhattan': 80726.0, 'Cosine': 0.91521245}

Images (650, 725): {'Euclidean': 3828.2957, 'Manhattan': 65990.0, 'Cosine': 0.76435477}
Images (650, 726): {'Euclidean': 2993.9663, 'Manhattan': 53740.0, 'Cosine': 0.43449128}
Images (650, 727): {'Euclidean': 3165.2131, 'Manhattan': 56260.0, 'Cosine': 0.50674886}
Images (650, 728): {'Euclidean': 4465.737, 'Manhattan': 70272.0, 'Cosine': 0.7824625}
Images (650, 729): {'Euclidean': 3514.3733, 'Manhattan': 60898.0, 'Cosine': 0.65350753}
Images (650, 730): {'Euclidean': 3615.8174, 'Manhattan': 62620.0, 'Cosine': 0.69817054}
Images (650, 731): {'Euclidean': 4135.2905, 'Manhattan': 63322.0, 'Cosine': 0.7550596}
Images (650, 732): {'Euclidean': 3420.1523, 'Manhattan': 61302.0, 'Cosine': 0.62903297}
Images (650, 733): {'Euclidean': 6018.27, 'Manhattan': 75260.0, 'Cosine': 0.9159551}
Images (650, 734): {'Euclidean': 4104.8877, 'Manhattan': 61472.0, 'Cosine': 0.76790017}
Images (650, 735): {'Euclidean': 4138.7456, 'Manhattan': 72846.0, 'Cosine': 0.8098629}
Images (650, 736): {'Euclidean': 4068.461, 'Manhattan': 72006.0, 'Cosine': 0.79853845}
Images (650, 737): {'Euclidean': 3970.4182, 'Manhattan': 67526.0, 'Cosine': 0.76482093}
Images (650, 738): {'Euclidean': 3881.4924, 'Manhattan': 66748.0, 'Cosine': 0.7645081}
Images (650, 739): {'Euclidean': 3424.2744, 'Manhattan': 56500.0, 'Cosine': 0.59899414}
Images (650, 740): {'Euclidean': 3834.425, 'Manhattan': 64620.0, 'Cosine': 0.7225076}
Images (650, 741): {'Euclidean': 6089.3667, 'Manhattan': 71912.0, 'Cosine': 0.8879161}
Images (650, 742): {'Euclidean': 3709.7502, 'Manhattan': 66244.0, 'Cosine': 0.7192794}
Images (650, 743): {'Euclidean': 3446.023, 'Manhattan': 57534.0, 'Cosine': 0.5885538}
Images (650, 744): {'Euclidean': 3679.7158, 'Manhattan': 63452.0, 'Cosine': 0.6939398}

Images (650, 745): {'Euclidean': 3161.9707, 'Manhattan': 53920.0, 'Cosine': 0.49592495}
Images (650, 746): {'Euclidean': 3447.0916, 'Manhattan': 57950.0, 'Cosine': 0.62055147}
Images (651, 652): {'Euclidean': 2979.6057, 'Manhattan': 57940.0, 'Cosine': 0.53284705}
Images (651, 653): {'Euclidean': 1871.4711, 'Manhattan': 34312.0, 'Cosine': 0.26415545}
}
Images (684,



(b) Feature Extraction: Edge histogram AND Similarity Measurements
i. Choose 1 image from each class. ii. Convert the color images to grayscale images.

```
import os
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
from sklearn.decomposition import PCA
```

```
from sklearn.metrics import pairwise
import cv2
import zipfile

images_zip_path = '/content/Breeds.zip'
images_folder_path = '/content/Breeds'  # Extracted directory
cropped_images_path = '/content/sample_data/Cropped_Images'

# Function to unzip files
def unzip_files(zip_file_path, extract_path):
    with zipfile.ZipFile(zip_file_path, 'r') as zip_ref:
        zip_ref.extractall(extract_path)

# Unzip the images
unzip_files(images_zip_path, images_folder_path)

# Function to load images (No changes here)
def load_images(folder):
    images = []
    for dirpath, _, files in os.walk(folder):
        for filename in files:
            if filename.endswith(('.png', '.jpg', '.jpeg')):
                img_path = os.path.join(dirpath, filename)
                img = Image.open(img_path).convert('RGB')
                images.append(np.array(img))
    return images

# Function to crop and resize images (No changes here)
def crop_and_resize_images(images, size=(128, 128)):
    cropped_resized_images = []
    for img in images:

        width, height = img.shape[1], img.shape[0]
        left = (width - size[0]) / 2
        top = (height - size[1]) / 2
        right = (width + size[0]) / 2
        bottom = (height + size[1]) / 2

        cropped_img = img[int(top):int(bottom), int(left):int(right)]
        resized_img = cv2.resize(cropped_img, size)
```

```
        cropped_resized_images.append(resized_img)
    return cropped_resized_images

def compute_histograms(images):
    histograms = []
    for img in images:
        if len(img.shape) == 2:  # Check if image is grayscale
            hist = cv2.calcHist([img], [0], None, [256], [0, 256])
        else:  # Calculate histogram for each channel if RGB
             hist = cv2.calcHist([img], [0, 1, 2], None, [256, 256, 256], [0, 256, 0, 256,
0, 256])
        histograms.append(hist.flatten())
    return histograms

def compute_similarity_measurements(histograms):
    for i in range(len(histograms)):
        for j in range(i + 1, len(histograms)):
            euclidean_distance = np.linalg.norm(histograms[i] - histograms[j])
            manhattan_distance = np.sum(np.abs(histograms[i] - histograms[j]))
                    cosine_distance = 1 - pairwise.cosine_similarity([histograms[i]],
[histograms[j]])[0][0]
                    print(f'Images ({i}, {j}): {{\'Euclidean\': {euclidean_distance},
\'Manhattan\': {manhattan_distance}, \'Cosine\': {cosine_distance}}}')

def perform_pca(histograms):
    pca = PCA(n_components=2)
    reduced_data = pca.fit_transform(histograms)

    plt.figure(figsize=(20,15))
    plt.scatter(reduced_data[:, 0], reduced_data[:, 1], alpha=0.5)
    plt.title('PCA of Image Histograms')
    plt.xlabel('Principal Component 1')
    plt.ylabel('Principal Component 2')
    plt.grid()
    plt.show()

def plot_images(images, titles=None, cols=4):  # Reduce number of columns
    n_images = len(images)
    rows = (n_images + cols - 1) // cols
    plt.figure(figsize=(250, 250))  # Increase figure size
```

```
    for i in range(n_images):
        plt.subplot(rows, cols, i + 1)
        plt.imshow(images[i].astype(np.uint8))
        plt.axis('off')
        if titles is not None:
            plt.title(titles[i], fontsize=2)  # Reduce font size
    # plt.tight_layout()
    plt.show()

if __name__ == "__main__":

    dog_images = load_images(images_folder_path)

            plot_images(dog_images, titles=[f'Image {i+1}' for i in
range(len(dog_images))])

    cropped_resized_images = crop_and_resize_images(dog_images)

    histograms = compute_histograms(cropped_resized_images)

    if histograms: # Check if histograms is not empty before proceeding
        compute_similarity_measurements(histograms)

        perform_pca(histograms)

     plot_images(cropped_resized_images, titles=[f'Cropped Image {i+1}' for i in
range(len(cropped_resized_images))])
```

OUTPUT:

3. Next, we perform some text processing steps on a tweet (i.e., text) dataset. The dataset file is in json format and each dataset consists of

- **Training Set: 3,000 records**
- **Test Set: 1,500 records**
- **Validation Set: 400 records**

```
import json
file_path = '/content/train.json'
data = []
with open(file_path, 'r') as file:
   for line in file:
      data.append(json.loads(line))
print(json.dumps(data[0], indent=4))
```

OUTPUT:

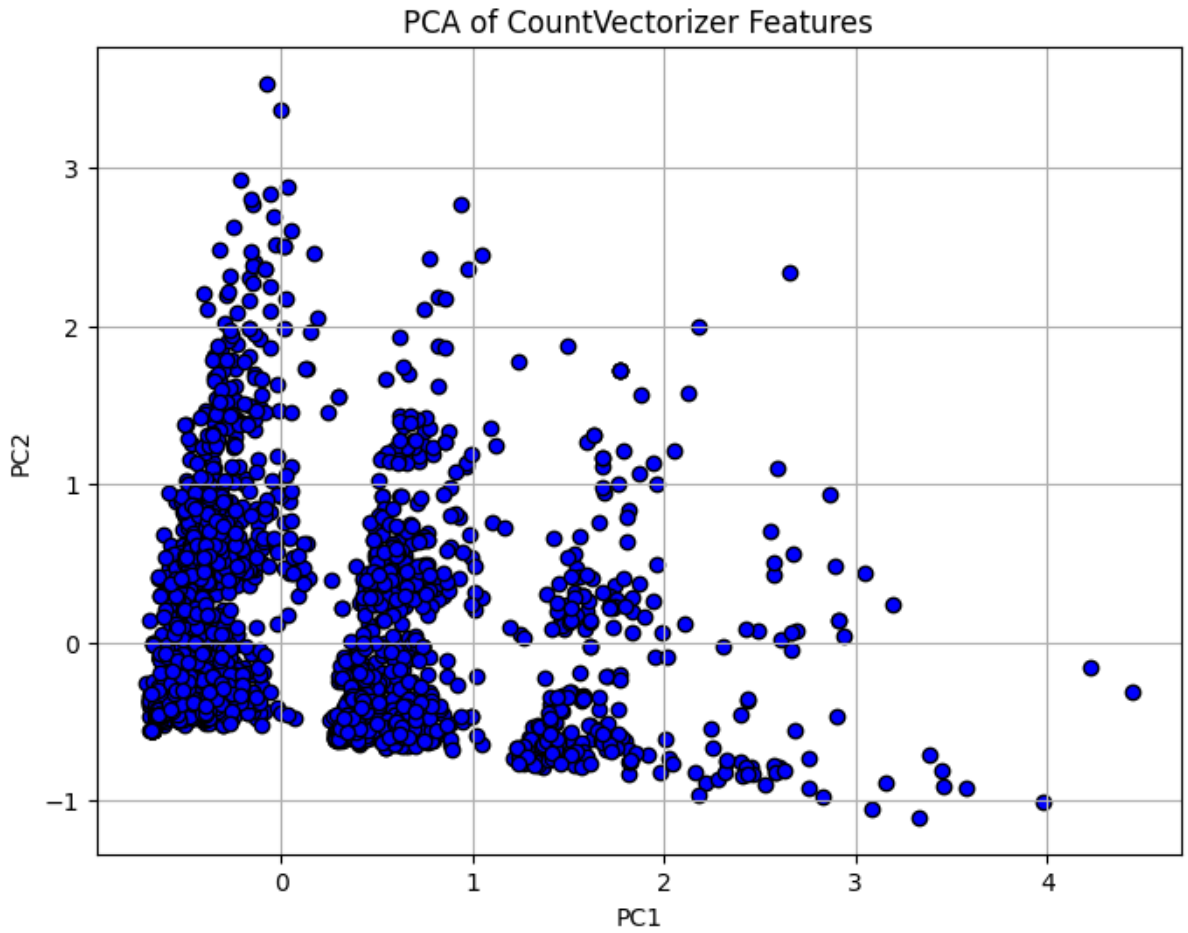

```
{
   "ID": "2017-En-20191",
   "Tweet": "Dates in the glove box' is pure panic excuse #GBBO",
   "anger": false,
   "anticipation": true,
   "disgust": true,
   "fear": true,
   "joy": false,
   "love": false,
   "optimism": false,
   "pessimism": false,
   "sadness": true,
   "surprise": false,
   "trust": false
}
```

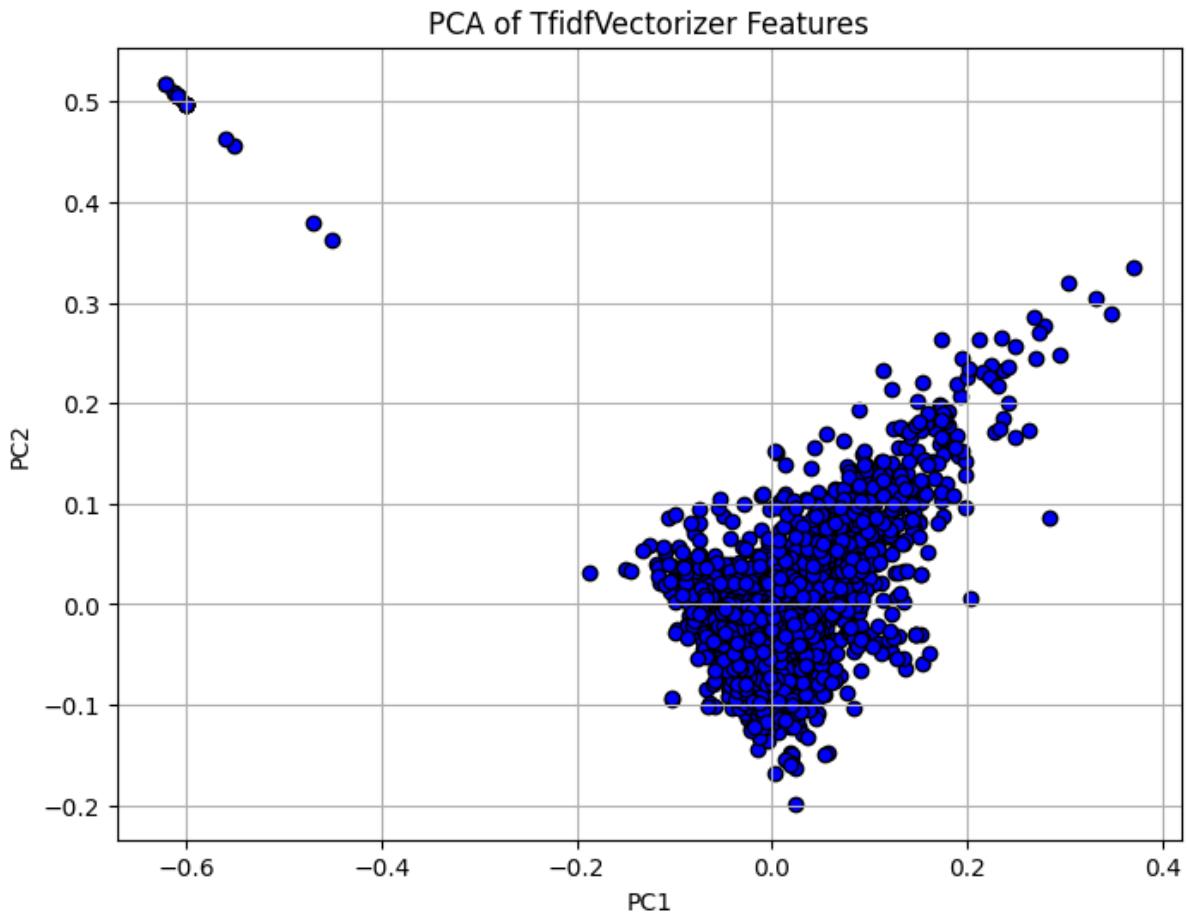
4. You will use the simple countvectorizer and tfidfvectorizer in https://scikit-learn.org/stable/api/sklearn.feature_extraction.html#module-sklearn.feature_extraction.text to extract
(1) token (feature) counts, and
(2) TF-IDF feature (counts), respectively

```
import json
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

file_path = 'train.json'
data = []
with open(file_path, 'r') as file:
    for line in file:
        data.append(json.loads(line))

texts = [entry['Tweet'] for entry in data]

count_vectorizer = CountVectorizer()
count_vectors = count_vectorizer.fit_transform(texts)

tfidf_vectorizer = TfidfVectorizer()
tfidf_vectors = tfidf_vectorizer.fit_transform(texts)

pca = PCA(n_components=2)
count_pca = pca.fit_transform(count_vectors.toarray())
tfidf_pca = pca.fit_transform(tfidf_vectors.toarray())

def plot_pca(pca_result, title):
    plt.figure(figsize=(8, 6))
    plt.scatter(pca_result[:, 0], pca_result[:, 1], c='blue', marker='o',
edgecolor='k')
    plt.title(title)
    plt.xlabel('PC1')
    plt.ylabel('PC2')
    plt.grid(True)
    plt.show()

plot_pca(count_pca, 'PCA of CountVectorizer Features')
```

```
plot_pca(tfidf_pca, 'PCA of TfidfVectorizer Features')

count_dimensionality = count_vectors.shape
tfidf_dimensionality = tfidf_vectors.shape

print(f"Dimensionality of CountVectorizer representation: {count_dimensionality}")
print(f"Dimensionality of TfidfVectorizer representation: {tfidf_dimensionality}")
```

OUTPUT:





5. Using the two sets of processed text data in Item 4,

• Pick four classes which you think will be separable. State the four classes.

• Perform dimensionality reduction similar to 2(d) with reduced to

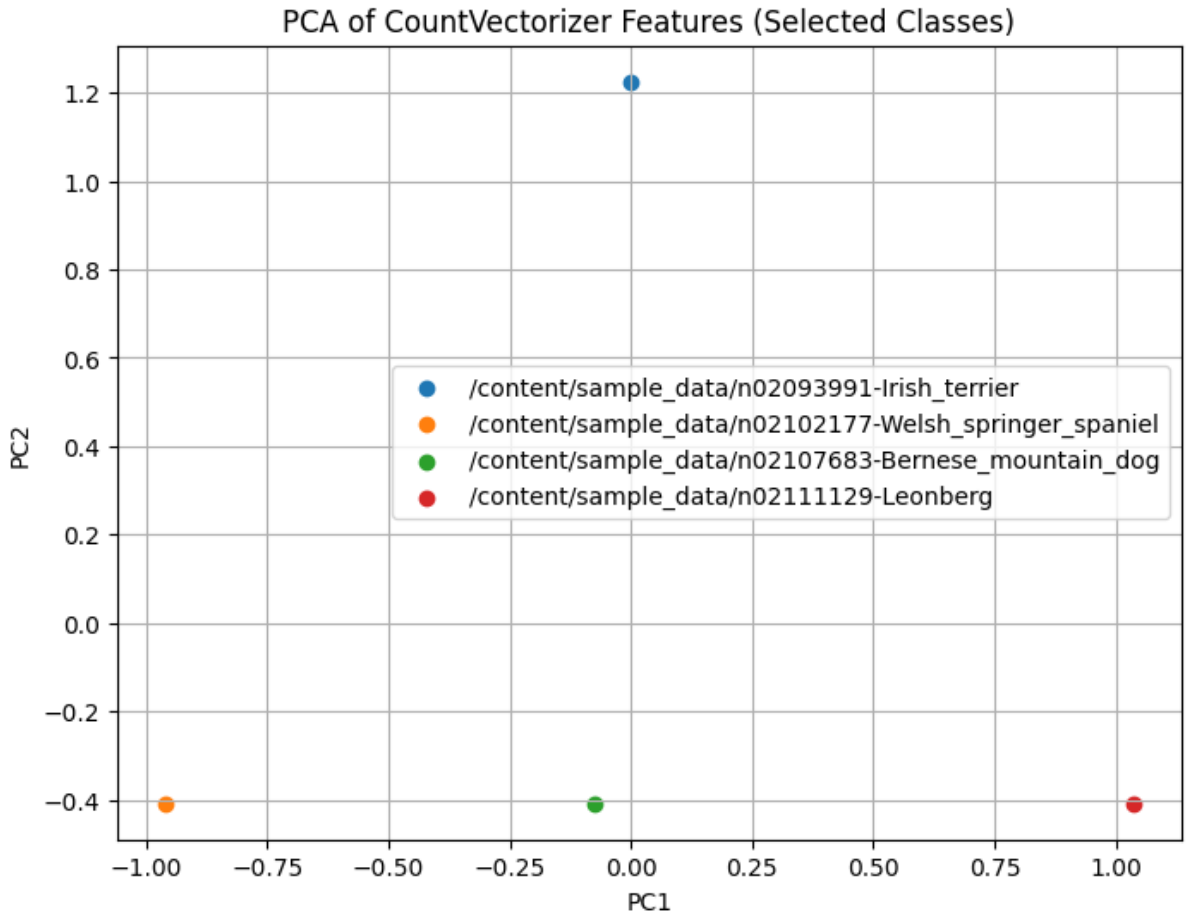
• Plot the 2D points using four different colors for data from the four classes for both token count features and tf-idf features in two separate plots.

• How many classes are visually separable (i.e., non-overlapping) for both plots?

```
# Import necessary libraries
import os
import tarfile
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt
from skimage import filters, exposure
from skimage.feature import hog
from sklearn.decomposition import PCA
from sklearn.metrics import pairwise
from sklearn.feature_extraction.text import CountVectorizer
```

```
data = [
        {'Tweet': '/content/sample_data/n02093991-Irish_terrier', 'Class': '/content/sample_data/n02093991-Irish_terrier'}, # Added 'Class' key to the first dictionary
  {'Tweet': '/content/sample_data/n02102177-Welsh_springer_spaniel', 'Class': '/content/sample_data/n02102177-Welsh_springer_spaniel'},
  {'Tweet': '/content/sample_data/n02107683-Bernese_mountain_dog', 'Class': '/content/sample_data/n02107683-Bernese_mountain_dog'},
        {'Tweet': '/content/sample_data/n02111129-Leonberg', 'Class': '/content/sample_data/n02111129-Leonberg'},
]

selected_classes = ['/content/sample_data/n02093991-Irish_terrier', '/content/sample_data/n02102177-Welsh_springer_spaniel',
'/content/sample_data/n02107683-Bernese_mountain_dog',
'/content/sample_data/n02111129-Leonberg']  # Replace with your actual class labels

filtered_data = [entry for entry in data if entry['Class'] in selected_classes]

filtered_texts = [entry['Tweet'] for entry in filtered_data]
filtered_classes = [entry['Class'] for entry in filtered_data]

count_vectorizer = CountVectorizer()
count_vectors = count_vectorizer.fit_transform(filtered_texts)

count_vectors_filtered = count_vectorizer.transform(filtered_texts)
pca_count_filtered = PCA(n_components=2).fit_transform(count_vectors_filtered.toarray())

plt.figure(figsize=(8, 6))
for class_label in selected_classes:
  indices = [i for i, cls in enumerate(filtered_classes) if cls == class_label]
    plt.scatter(pca_count_filtered[indices, 0], pca_count_filtered[indices, 1], label=class_label)

plt.title('PCA of CountVectorizer Features (Selected Classes)')
plt.xlabel('PC1')
plt.ylabel('PC2')
```

```
plt.legend()
plt.grid(True)
plt.show()

# 8. Plot PCA results for TfidfVectorizer features (if applicable)
# ...
```




```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

data = [
   {'Tweet': '/content/sample_data/n02093991-Irish_terrier', 'Class':
'/content/sample_data/n02093991-Irish_terrier'}, # Added 'Class' key to the
first dictionary
   {'Tweet': '/content/sample_data/n02102177-Welsh_springer_spaniel',
'Class': '/content/sample_data/n02102177-Welsh_springer_spaniel'},
   {'Tweet': '/content/sample_data/n02107683-Bernese_mountain_dog',
'Class': '/content/sample_data/n02107683-Bernese_mountain_dog'},
```

```
   {'Tweet': '/content/sample_data/n02111129-Leonberg', 'Class':
'/content/sample_data/n02111129-Leonberg'},
]

selected_classes = ['/content/sample_data/n02093991-Irish_terrier',
'/content/sample_data/n02102177-Welsh_springer_spaniel',
'/content/sample_data/n02107683-Bernese_mountain_dog',
'/content/sample_data/n02111129-Leonberg']  # Replace with your actual class
labels

filtered_data = [entry for entry in data if entry['Class'] in selected_classes]

filtered_texts = [entry['Tweet'] for entry in filtered_data]
filtered_classes = [entry['Class'] for entry in filtered_data]

tfidf_vectorizer = TfidfVectorizer()
tfidf_vectors = tfidf_vectorizer.fit_transform(filtered_texts)

tfidf_vectors_filtered = tfidf_vectorizer.transform(filtered_texts)
pca_tfidf_filtered =
PCA(n_components=2).fit_transform(tfidf_vectors_filtered.toarray())

plt.figure(figsize=(8, 6))
for class_label in selected_classes:
  indices = [i for i, cls in enumerate(filtered_classes) if cls == class_label]
  plt.scatter(pca_tfidf_filtered[indices, 0], pca_tfidf_filtered[indices, 1],
label=class_label)

plt.title('PCA of TfidfVectorizer Features (Selected Classes)')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.legend()
plt.grid(True)
plt.show()
```


PCA of TfidfVectorizer Features (Selected Classes)
/content/sample_data/n02093991-Irish_terrier
/content/sample_data/n02102177-Welsh_springer_spaniel
/content/sample_data/n02107683-Bernese_mountain_dog
/content/sample_data/n02111129-Leonberg
PC1
PC2
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.2
0.0
0.2
0.4
0.6
0.8