In [6]:
```
import os

directory = os.getcwd()
images = r'cropped_images'

import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
# as in assignment 2 added normalization

img=[]
hist=[]
breed=[]

class_labels = {'n02085936-Maltese_dog':0,'n02096294-Australian_terrier':1,'n021065

for folder in os.listdir(images):
    path = os.path.join(images, folder)
    if os.path.isdir(path):
        Images = os.listdir(path)
        crop_images = [image for image in Images if image.lower().endswith(('.jpg')
        for image in crop_images:
            src_path = os.path.join(path, image)
            img.append(src_path)
            breed.append(class_labels[folder])

import numpy as np
from skimage import filters
from skimage import data, exposure, img_as_float,color
def angle(dx, dy):
 """Calculate the angles between horizontal and vertical operators."""
 return np.mod(np.arctan2(dy, dx), np.pi)

from skimage import filters
from skimage import io
for image in img:
    dog_img   = io.imread(image)
    dog_img=color.rgb2gray(dog_img)
    angle_sobel = angle(filters.sobel_h(dog_img),filters.sobel_v(dog_img))
    Hist,_=exposure.histogram(angle_sobel, nbins=36)
    hist.append(Hist/np.sum(Hist))


hist=np.array(hist)
breed=np.array(breed)


#PCA
from sklearn.decomposition import PCA
```

```
pca = PCA(n_components=2)
edgehist=pca.fit_transform(hist)
```

In [7]:
```
from sklearn.cluster import KMeans,BisectingKMeans,SpectralClustering,Agglomerative
from sklearn.metrics import silhouette_score,fowlkes_mallows_score

model_dict={"Random"  :KMeans(init="random",n_clusters=4),
        "KMeans++":KMeans(init="k-means++",n_clusters=4),
        "BisectingKmeans":BisectingKMeans(init="random",n_clusters=4),
        "SpectralClustering":SpectralClustering(n_clusters=4),
         "Agglomerate":AgglomerativeClustering(n_clusters=4) }
algorith=[]
silhouette=[]
fowlkesmallows=[]
for method,model in model_dict.items():
    if method == "Agglomerate":
        for link in ['single','complete', 'average','ward']:
            model=AgglomerativeClustering(n_clusters=4,linkage=link)
            predict=model.fit_predict(edgehist)
            algorith.append(link)
            silhouette.append(silhouette_score(edgehist,predict))
            fowlkesmallows.append(fowlkes_mallows_score(breed,predict))
    else:
        model=model.fit(edgehist)
        algorith.append(method)
        silhouette.append(silhouette_score(edgehist,model.labels_))
        fowlkesmallows.append(fowlkes_mallows_score(breed,model.labels_))
```

In [14]:
```
#DBSCAN
from sklearn.cluster import DBSCAN
model = DBSCAN(eps=0.01, min_samples=3).fit(edgehist)
pred = model.labels_
n_clusters_ = len(set(pred)) - (1 if -1 in pred else 0)
print("Estimated number of clusters: %d" % n_clusters_)
algorith.append('DBSCAN')
silhouette.append(silhouette_score(edgehist,pred))
fowlkesmallows.append(fowlkes_mallows_score(breed,pred))
```

```
Estimated number of clusters: 4
```

DBSCAN parameters used eps as 0.01 and min_samples as 3 in order to get 4 clusters

In [16]:
```
import pandas as pd
algorithm_scores=pd.DataFrame({'method': algorith, 'fowlkes-mallows': fowlkesmallow
```

best to worst based on fowlkes score

In [17]:
```
algorithm_scores.sort_values('fowlkes-mallows',ascending=False)
```

Out[17]:

	method	fowlkes-mallows	silhouette
4	single	0.506164	0.698595
6	average	0.505364	0.504076
8	DBSCAN	0.497707	0.262266
5	complete	0.356079	0.437289
3	SpectralClustering	0.346667	0.001596
0	Random	0.310303	0.433507
1	KMeans++	0.310303	0.433507
2	BisectingKmeans	0.305513	0.396617
7	ward	0.299091	0.414192

best to worst silhouette score

In [18]:
```
algorithm_scores.sort_values('silhouette',ascending=False)
```

Out[18]:

	method	fowlkes-mallows	silhouette
4	single	0.506164	0.698595
6	average	0.505364	0.504076
5	complete	0.356079	0.437289
0	Random	0.310303	0.433507
1	KMeans++	0.310303	0.433507
7	ward	0.299091	0.414192
2	BisectingKmeans	0.305513	0.396617
8	DBSCAN	0.497707	0.262266
3	SpectralClustering	0.346667	0.001596

In [ ]:
```
### https://scikit-learn.org/stable/modules/clustering.html
```