Task4_ALI

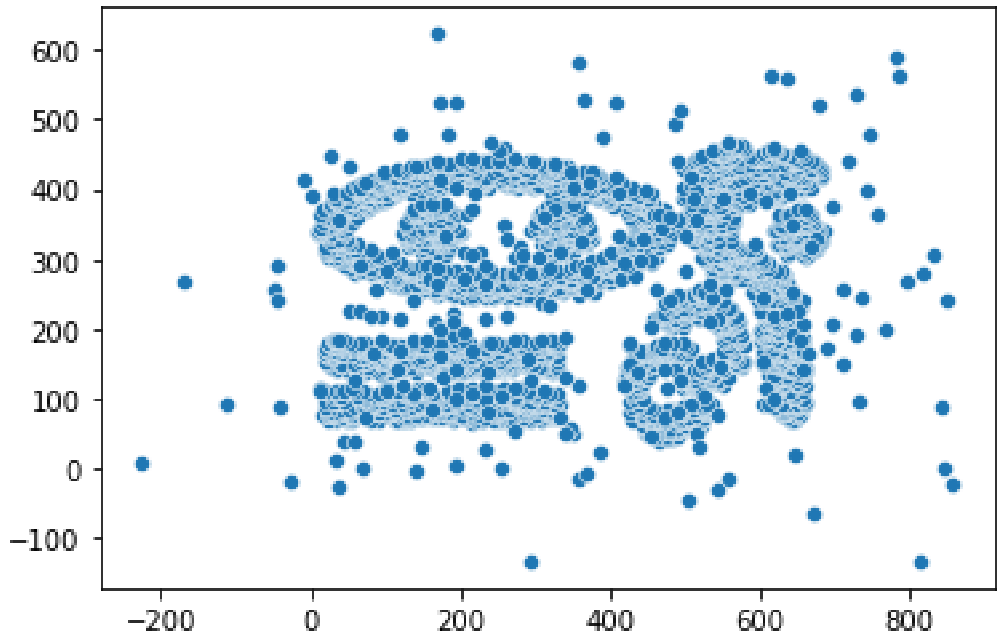
December 6, 2022

[2]:
```
import pandas as pd
import numpy as np
from sklearn import metrics
from sklearn.cluster import KMeans
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import matplotlib.pyplot as plt
import scipy.stats as stats
import sklearn
from sklearn.cluster import DBSCAN
from sklearn.neighbors import NearestNeighbors
from collections import Counter
from sklearn.preprocessing import StandardScaler
```

[3]:
```
df2 = pd.read_csv("Shuttle22.csv", header= None)
df = pd.read_csv("complex9_gn8.txt", header = None,sep=',')
```

[18]:
```
def purity_score(y_true, y_pred,outliers):
    # compute contingency matrix (also called confusion matrix)
    contingency_matrix = metrics.cluster.contingency_matrix(y_true, y_pred)
    # return purity
    return np.sum(np.amax(contingency_matrix, axis=0)) / np.
↪sum(contingency_matrix)
```

[87]:
```
X=df[[0,1]]
X=X.values
X
sns.scatterplot(X[:,0], X[:, 1])
data = df[[0,1]]
plt.show()
data,X
```



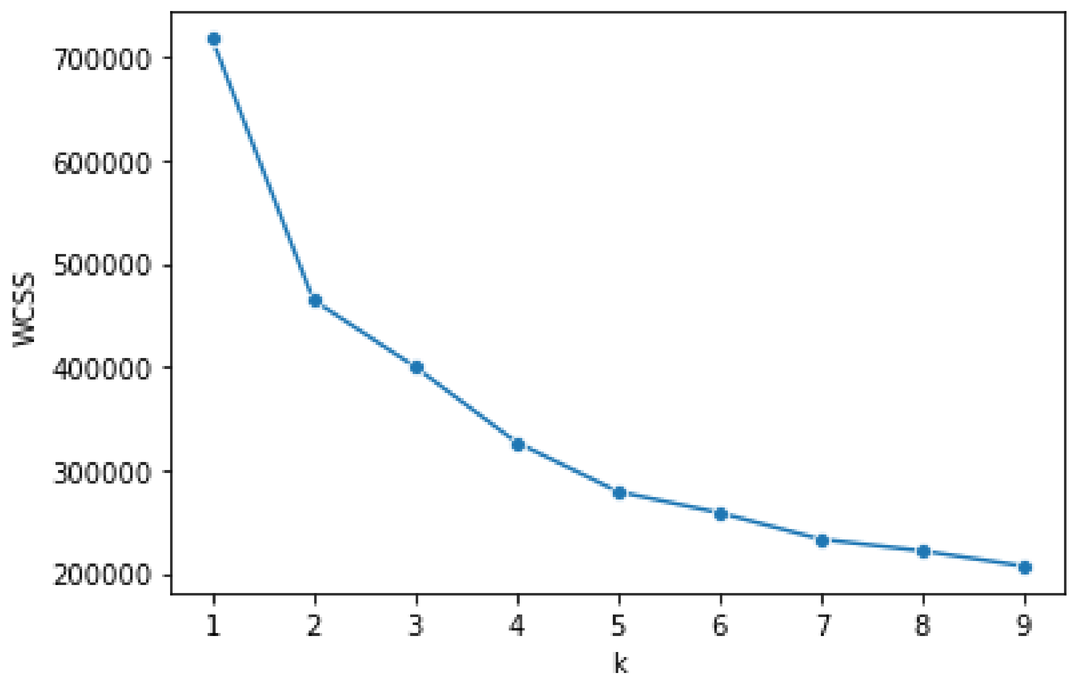

```
[87]: (            0         1
 0     660.976  304.2250
 1     636.213  306.1740
 2     662.753  307.5650
 3     657.487  307.7400
 4     635.273  308.1570
 …        …        …
 3268  728.899  535.6270
 3269  504.528  -46.2297
 3270  373.256  409.0260
 3271  850.838  242.7110
 3272  641.676  347.5440

 [3273 rows x 2 columns],
 array([[660.976, 304.225],
        [636.213, 306.174],
        [662.753, 307.565],
        …,
        [373.256, 409.026],
        [850.838, 242.711],
        [641.676, 347.544]]))
```

```
[102]: def calculate_cost(X, centroids, cluster):
           sum = 0
           for i, val in enumerate(X):
```

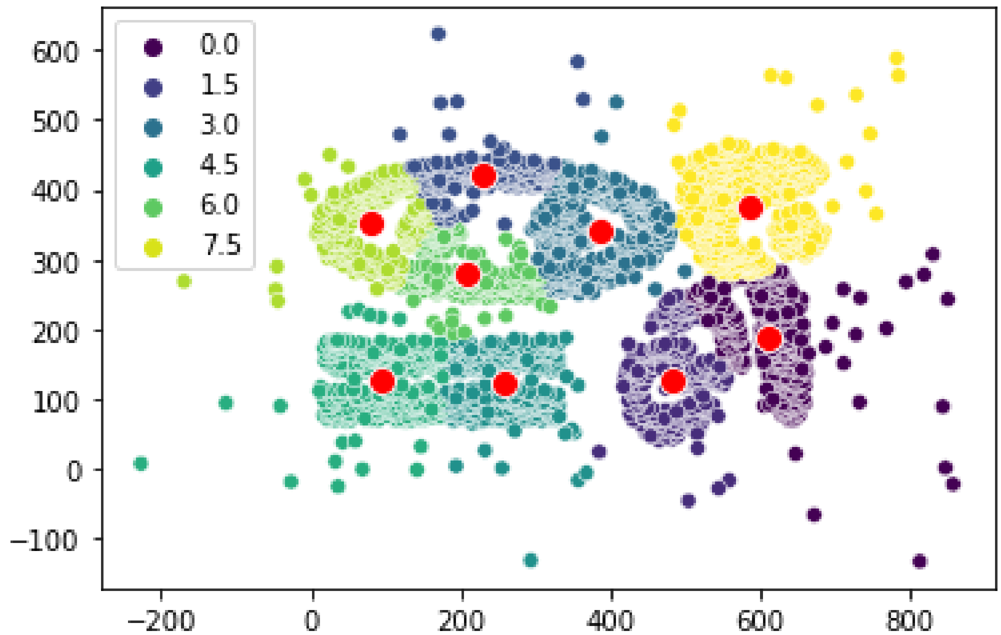
```
        sum += np.sqrt((centroids[int(cluster[i]), 0]-val[0])**2␣
↪+(centroids[int(cluster[i]), 1]-val[1])**2)
    return sum
def kmeans(X, k):
    diff = 1
    cluster = np.zeros(X.shape[0])
    centroids = data.sample(n=k).values
    while diff:
    # for each observation
        for i, row in enumerate(X):
            mn_dist = float('inf')
            # dist of the point from all centroids
            for idx, centroid in enumerate(centroids):
                d = np.sqrt((centroid[0]-row[0])**2 + (centroid[1]-row[1])**2)
                # store closest centroid
                if mn_dist > d:
                    mn_dist = d
                    cluster[i] = idx
        new_centroids = pd.DataFrame(X).groupby(by=cluster).mean().values
     # if centroids are same then leave
        if np.count_nonzero(centroids-new_centroids) == 0:
            diff = 0
        else:
            centroids = new_centroids
    return centroids, cluster
cost_list = []
for k in range(1, 10):
    centroids, cluster = kmeans(X, k)
    # WCSS (Within cluster sum of square)
    cost = calculate_cost(X, centroids, cluster)
    cost_list.append(cost)
```

[104]:

```
sns.lineplot(x=range(1,10), y=cost_list, marker='o')
plt.xlabel('k')
plt.ylabel('WCSS')
plt.show()
```



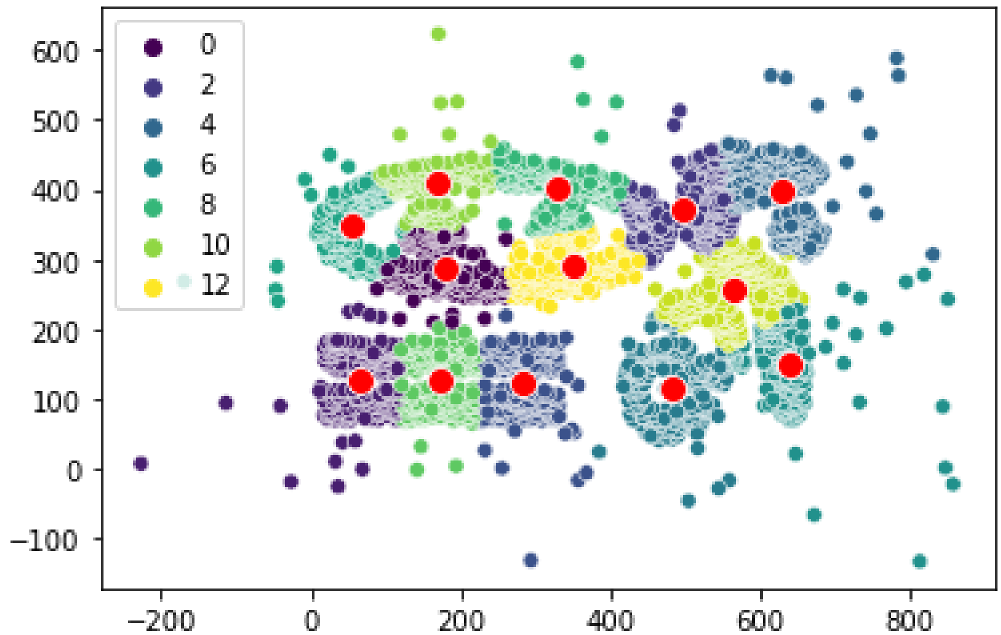
[143]:
```
palette=sns.color_palette("viridis", as_cmap=True)

k = 9
centroids, cluster = kmeans(X, k)

sns.scatterplot(X[:,0], X[:, 1], hue=cluster,palette=palette)
sns.scatterplot(centroids[:,0], centroids[:, 1], s=100, color='r')
plt.show()
```



[154]:
```
true = df[2]
purity_score(true,cluster)
```

[154]: 0.7100519401161014

[155]:
```
k = 13
centroids, cluster = kmeans(X, k)

sns.scatterplot(X[:,0], X[:, 1], hue=cluster,palette=palette)
sns.scatterplot(centroids[:,0], centroids[:, 1], s=100, color='r')
plt.show()
purity_score(true,cluster)
```



[155]: 0.6804155209288115

[4]:
```
z = df2[[0,1,2,3,4,5,6,7,8]]
ZSHUT=z.apply(stats.zscore)
ZSHUT[9] = df2[9]

zdata = ZSHUT[[0,1,2,3,4,5,6,7,8]]
zdata= zdata.values
zdata
#z
```

[4]:
```
array([[ 1.43954104e-01,  2.69627492e-01, -9.37819783e-01, …,
        -7.69740511e-01, -1.34679698e-01,  3.14970333e-01],
       [ 5.52518390e-01,  2.49473239e-04,  7.47063843e-01, …,
        -8.33105101e-02,  1.91967997e+00,  1.64237992e+00],
       [ 3.89092675e-01,  2.49473239e-04, -3.76191907e-01, …,
        -6.17200511e-01, -9.75099563e-01, -4.65858839e-01],
       …,
       [ 5.52518390e-01,  2.49473239e-04, -9.37819783e-01, …,
        -1.15109051e+00,  6.59050174e-01,  1.09579950e+00],
       [-9.18313039e-01,  2.49473239e-04,  1.98264517e+00, …,
         2.20478949e+00,  1.59285002e+00,  2.36887415e-01],
       [ 6.34231247e-01,  2.59045226e-02,  1.42101729e+00, …,
         3.74309491e-01, -2.28059683e-01, -3.87775921e-01]])
```

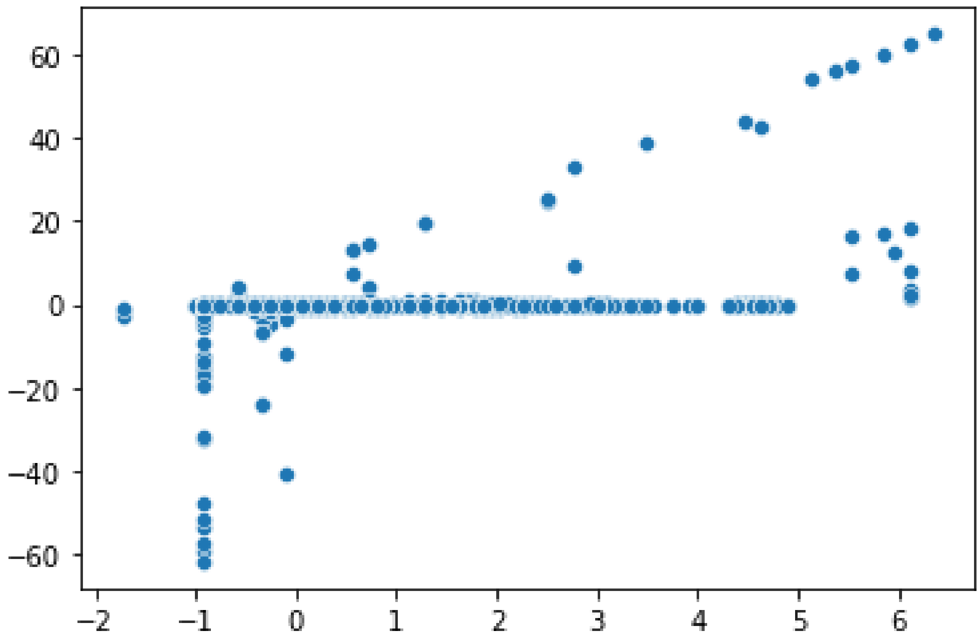
[5]:
```
data =ZSHUT[[0,1,2,3,4,5,6,7,8]]
data
```

[5]:
```
              0         1         2        3         4         5         6  \
0      0.143954  0.269627 -0.937820 -0.00711 -0.302395 -0.007391 -0.769741
1      0.552518  0.000249  0.747064 -0.00711 -1.595103  0.112097 -0.083311
2      0.389093  0.000249 -0.376192 -0.00711  0.805641 -0.030369 -0.617201
3     -0.918313  0.000249 -1.050145 -0.00711 -0.302395  0.075331  0.221769
4     -0.918313  0.000249 -0.713169 -0.00711 -0.025386 -0.126878  0.450579
…           …         …         …        …         …         …         …
57995  2.595340  0.000249 -0.151541 -0.00711 -3.257157 -0.140665 -2.523951
57996  0.552518  0.000249 -0.488517 -0.00711 -2.518466  0.107501 -0.846011
57997  0.552518  0.000249 -0.937820 -0.00711 -1.041086 -0.108496 -1.151091
57998 -0.918313  0.000249  1.982645 -0.00711 -0.764077 -0.080922  2.204789
57999  0.634231  0.025905  1.421017 -0.00711  0.805641 -0.002795  0.374309

             7         8
0     -0.13468  0.314970
1      1.91968  1.642380
2     -0.97510 -0.465859
3     -0.13468 -0.231610
4     -0.22806 -0.465859
…          …         …
57995  3.22700  3.984867
57996  2.38658  2.423209
57997  0.65905  1.095800
57998  1.59285  0.236887
57999 -0.22806 -0.387776

[58000 rows x 9 columns]
```

[353]:
```
k = 3

centroids, cluster = kmeans(zdata, k)


true2 = ZSHUT[9]
purity_score(true2,cluster)
#cluster
```

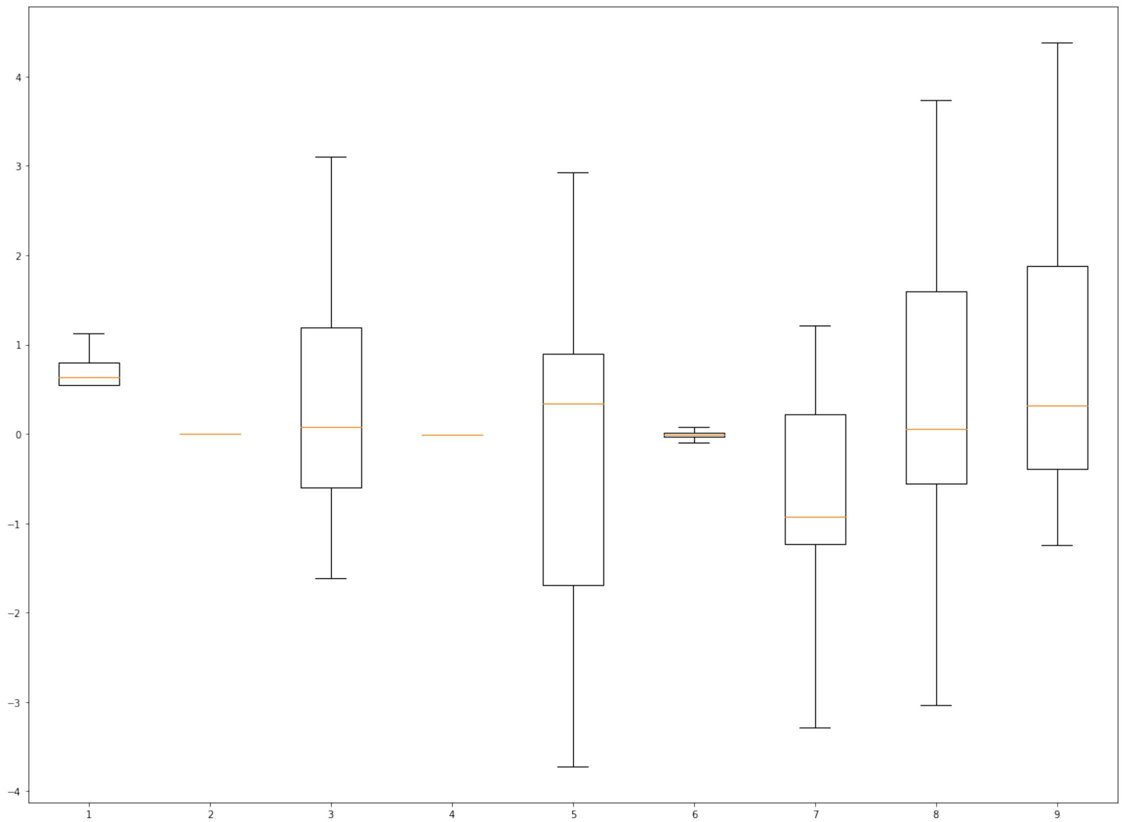
[353]: 0.868948275862069

[340]:
```
sns.scatterplot(zdata[:,0], zdata[:, 1])
plt.show()
```

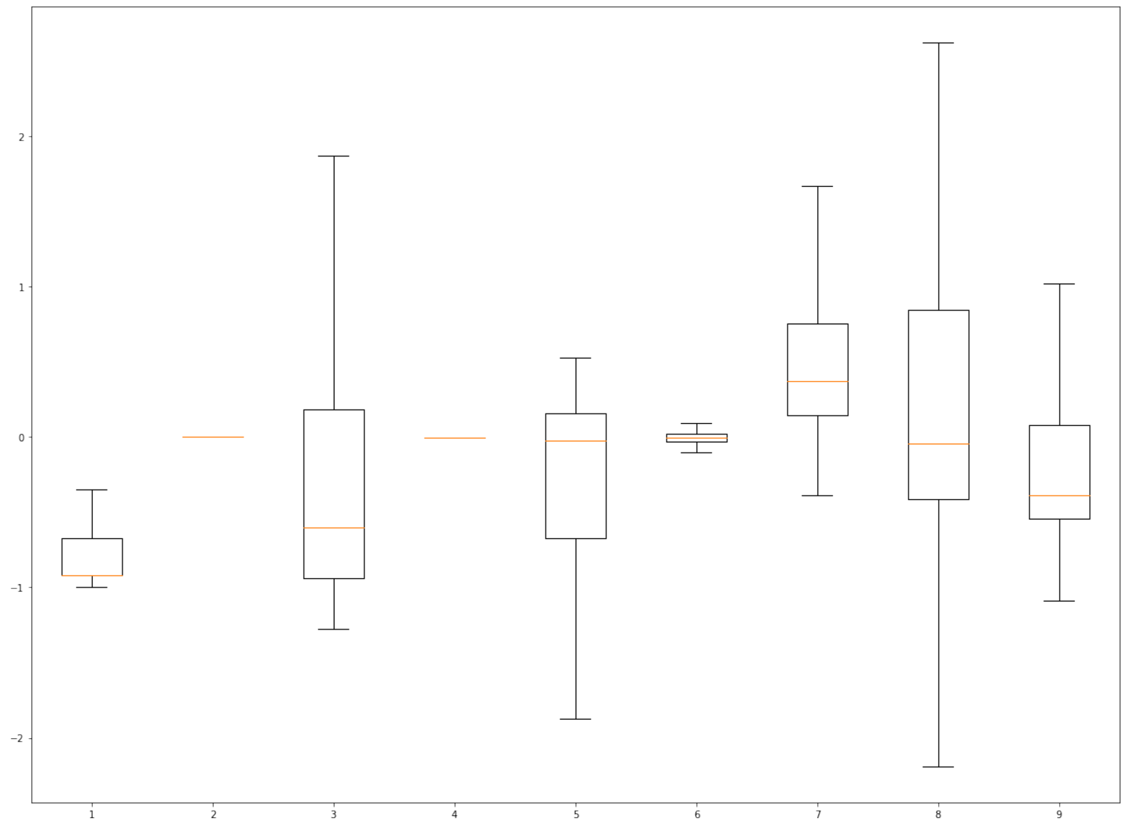


[362]:
```
c=data
c[9] = cluster.tolist()
c.loc[c[9]==1]
box0 =c.loc[c[9]==0]
box1 =c.loc[c[9]==1]
box2 =c.loc[c[9]==2]
box0=box0[[0,1,2,3,4,5,6,7,8]]
box1=box1[[0,1,2,3,4,5,6,7,8]]
box2=box2[[0,1,2,3,4,5,6,7,8]]
```

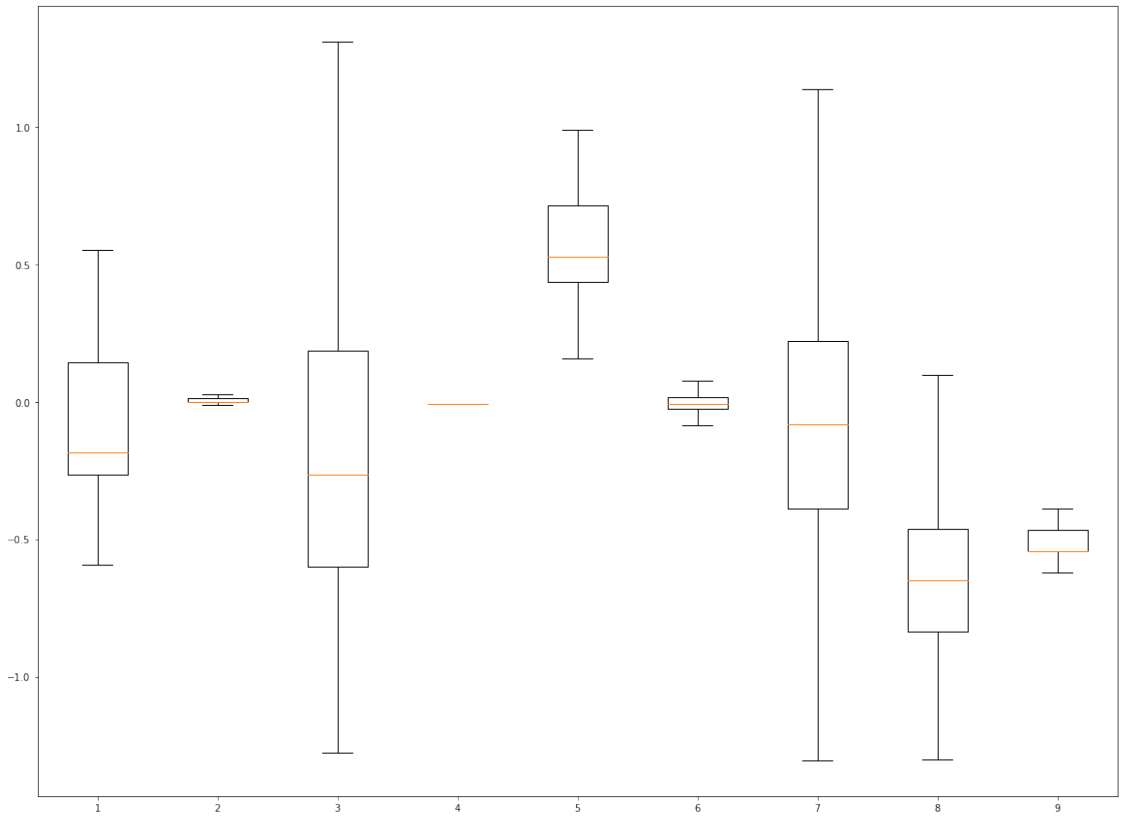
[369]:
```
plt.figure(figsize=(20,15))
plt.boxplot(box0.values,showfliers=False,
            flierprops={'marker': 'o', 'markersize': 1, 'markerfacecolor':␣
 ↪'fuchsia'})
plt.show()
```



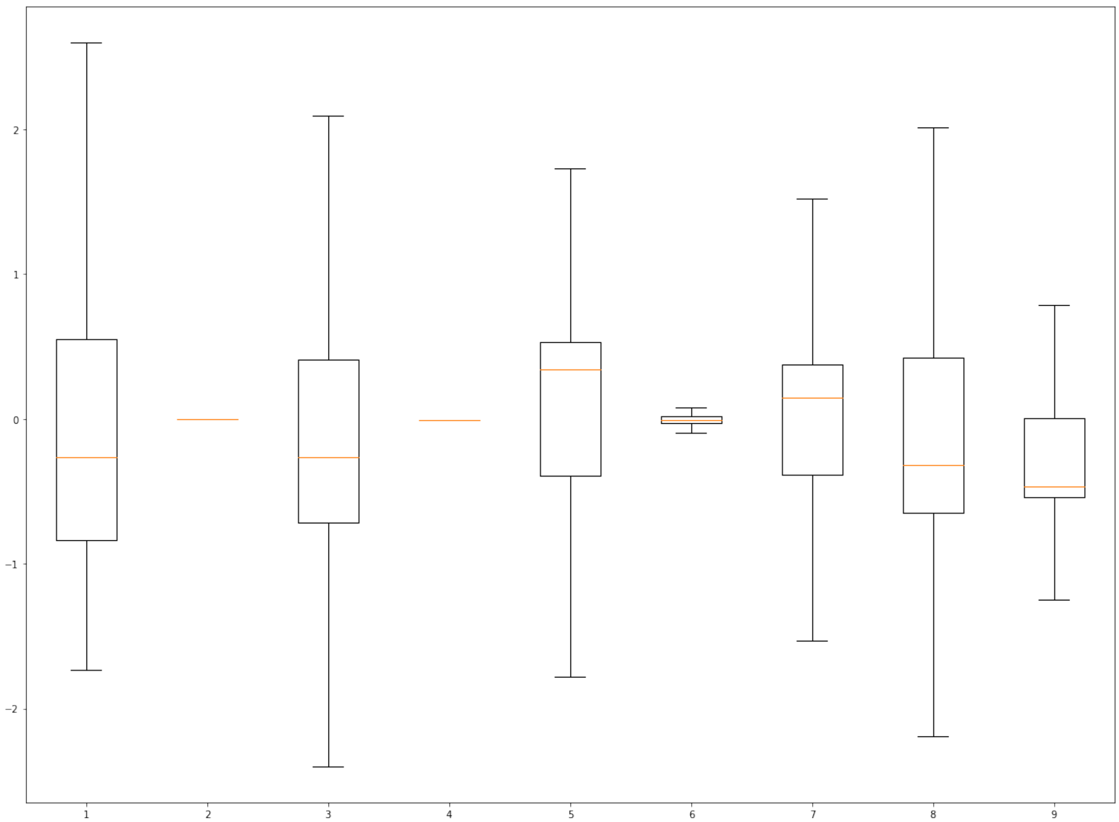
[365]:
```
plt.figure(figsize=(20,15))
plt.boxplot(box1.values,showfliers=False,
            flierprops={'marker': 'o', 'markersize': 1, 'markerfacecolor':␣
 ↪'fuchsia'})
plt.show()
```



[366]:
```
plt.figure(figsize=(20,15))
plt.boxplot(box2.values,showfliers=False,
            flierprops={'marker': 'o', 'markersize': 1, 'markerfacecolor':␣
 ↪'fuchsia'})
plt.show()
```



[373]:
```
plt.figure(figsize=(20,15))
plt.boxplot(data.values,showfliers=False,
            flierprops={'marker': 'o', 'markersize': 1, 'markerfacecolor':␣
 ↪'fuchsia'})
plt.show()
```



[64]:
```
def dbscan(dataset, epsilon, minsamples):
    dbscan_data = dataset[[0,1,2,3,4,5,6,7,8,9]]
    model1 = DBSCAN(eps=epsilon, min_samples=minsamples, metric='euclidean').\
    fit(dbscan_data)

    outliers_df = dataset[model1.labels_ == -1]
    num_clusters = len(set(model1.labels_))

    clusters_df = [dataset[model1.labels_ == n] for n in range(num_clusters)]

    colors = model1.labels_
    color_clusters = colors[colors != -1]
    color_outliers = 'white'
    clusters1 = Counter(model1.labels_)
    return clusters_df

def dbscanplot(dataset, epsilon, minsamples):
    dbscan_data = dataset[[0,1,2,3,4,5,6,7,8,9]]

    #return dbscan_data
    model1 = DBSCAN(eps=epsilon, min_samples=minsamples, metric='euclidean').\
```

```
    fit(dbscan_data)
    #return model1
    outliers_df = dataset[model1.labels_ == -1]
    clusters_df = dataset[model1.labels_ != -1]
    return model1.labels_
    '''colors = model1.labels_
    color_clusters = colors[colors != -1]
    color_outliers = 'white'
    clusters1 = Counter(model1.labels_)
    print(clusters_df)
    print(clusters1)
    #print(dataset[model1.labels_ == -1].head())
    print('number of clusters: {}'.format(len(clusters1)-1))
    #dbscan_plot(clusters_df,color_clusters)'''
def dbscan_plot(cluster,cluster_colors):
    fig = plt.figure()
    ax = fig.add_axes([.2,.2,2,2])
    ax.scatter(ZSHUT[[0,1,2,3,4,5,6,7,8]],
    c = cluster_colors, edgecolors = 'black', s = 70)
    ax.set_xlabel('Latitude', fontsize=10)
    ax.set_ylabel('Longitude', fontsize=10)
    plt.title('title',fontsize=12)
    plt.grid(which='major',color='#cccccc', alpha=0.45)
    plt.show()
```

[8]:
```
dbscandata= ZSHUT[[0,1,2,3,4,5,6,7,8]]
dbscanplot(dbscandata,.26,100)
```

[8]:
```
array([-1,  0,  1, …,  3,  5,  1], dtype=int64)
```

[9]:
```
clusterlist =dbscan(dbscandata,.26,100)
```

[11]:
```
q = dbscanplot(dbscandata,.26,100)
```

[36]:
```
n=np.delete(q, np.where(q == -1))
ZSHUT[9].max
```

[36]:
```
<bound method NDFrame._add_numeric_operations.<locals>.max of 0        2
1        4
2        1
3        1
4        1
        ..
57995    5
57996    4
57997    4
57998    1
```

```
57999    4
Name: 9, Length: 58000, dtype: int64>
```

[40]:

```
clusterlist[1]
```

[40]:

```
              0         1         2         3         4         5         6  \
2      0.389093  0.000249 -0.376192 -0.007110  0.805641 -0.030369 -0.617201
3     -0.918313  0.000249 -1.050145 -0.007110 -0.302395  0.075331  0.221769
4     -0.918313  0.000249 -0.713169 -0.007110 -0.025386 -0.126878  0.450579
7      0.552518 -0.012578  1.084041 -0.089254  0.897977 -0.025773  0.221769
10    -0.918313  0.000249 -0.825494 -0.061873 -1.041086 -0.007391  0.374309
…          …         …         …         …         …         …         …
57990 -0.836600  0.025905 -0.713169 -0.007110  0.159287  0.075331  0.374309
57992 -0.754887 -0.025406 -0.600843 -0.116636  0.159287 -0.007391  0.298039
57993 -0.428036  0.000249 -0.488517  0.020271  0.343959 -0.048752 -0.007041
57994  0.062241  0.000249  0.185436 -0.007110  0.528632 -0.062539  0.069229
57999  0.634231  0.025905  1.421017 -0.007110  0.805641 -0.002795  0.374309

             7         8
2     -0.97510 -0.465859
3     -0.13468 -0.231610
4     -0.22806 -0.465859
7     -0.46151 -0.465859
10     0.65905  0.393053
…         …         …
57990 -0.46151 -0.543942
57992 -0.46151 -0.543942
57993 -0.55489 -0.465859
57994 -0.46151 -0.465859
57999 -0.22806 -0.387776

[42135 rows x 9 columns]
```

[51]:

```
ze = ZSHUT
ze[10]=q
ze2 = ze[ze[10] != -1]

truer = ze2[9]
mine = ze2[10]
purity_score(truer,mine)
```

[51]: 0.9325662054264966

[4]:

```
def dbscanplot1(dataset, epsilon, minsamples):
    dbscan_data = dataset[[0,1,2,3,4,5,6,7,8]]

    #return dbscan_data
```

```python
    model1 = DBSCAN(eps=epsilon, min_samples=minsamples, metric='euclidean').\
    fit(dbscan_data)
    #return model1
    outliers_df = dataset[model1.labels_ == -1]
    clusters_df = dataset[model1.labels_ != -1]
    colors = model1.labels_
    color_clusters = colors[colors != -1]
    color_outliers = 'white'
    clusters1 = Counter(model1.labels_)
    #return model1.labels_
    #print(clusters_df)
    print(clusters1)
    #print(dataset[model1.labels_ == -1].head())
    print('number of clusters: {}'.format(len(clusters1)-1))
    #dbscan_plot(clusters_df,color_clusters)



def dbscanplot2(dataset, epsilon, minsamples):
    dbscan_data = dataset[[0,1,2,3,4,5,6,7,8]]

    #return dbscan_data
    model1 = DBSCAN(eps=epsilon, min_samples=minsamples, metric='euclidean').\
    fit(dbscan_data)
    #return model1
    outliers_df = dataset[model1.labels_ == -1]
    clusters_df = dataset[model1.labels_ != -1]
    colors = model1.labels_
    color_clusters = colors[colors != -1]
    color_outliers = 'white'
    clusters1 = Counter(model1.labels_)
    return model1.labels_
```

[178]:
```python
yo =dbscanplot1(ZSHUT,.66,100)
```

```
Counter({0: 54416, 1: 2334, 2: 887, -1: 363})
number of clusters: 3
```

[167]:
```python
ssa =dbscanplot1(ZSHUT,.46,10)
```

```
Counter({0: 54398, 1: 2324, 2: 885, -1: 332, 4: 19, 3: 17, 5: 15, 6: 10})
number of clusters: 7
```

[164]:
```python
#(Counter(yo1[0])+Counter(yo[1])+Counter(yo[2])+Counter(yo[3])+Counter(yo[-1]))
Counter(yo1)
```

[164]:
```
Counter({0: 54470, 1: 2353, -1: 252, 2: 887, 3: 38})
```

[180]:
```
252/5800
```

[180]: 0.043448275862068966

[174]:
```
yo1 =dbscanplot2(ZSHUT,.66,10)
```

[175]:
```
clusters11 = Counter(yo1)
clusters11
```

[175]: Counter({0: 54470, 1: 2353, -1: 252, 2: 887, 3: 38})

[176]:
```
ze1 = ZSHUT
ze1[10]=yo1
ze22 = ze1[ze1[10] != -1]

truer = ze22[9]
mine = ze22[10]
purity_score(truer,mine)
#np.sum(1/55348)
```

[176]: 0.8432153494493316

[120]:
```
contingency_matrix = metrics.cluster.contingency_matrix(truer, mine)
    #return np.sum(np.amax(contingency_matrix, axis=0)) / np.
 ↪sum(contingency_matrix)
np.sum(np.amax(contingency_matrix, axis=0)/np.sum(contingency_matrix))
```

[120]: 0.9325662054264966

[ ]:
```
###start of search procedure###
yo =dbscanplot1(ZSHUT,.66,10) #gets the number of clusters and outlier␣
 ↪percentage
yo1 =dbscanplot2(ZSHUT,.66,10) #gets the cluster_df which holds labels for the␣
 ↪clusters

ze1 = ZSHUT #initilize the ZSHUT dataset
ze1[10]=yo1 #add the clusters_df to our zshut dataset
ze22 = ze1[ze1[10] != -1] #wherever the cluster is labeled -1 aka an outlier we␣
 ↪will remove from the dataset

truer = ze22[9] #set original cluster labels as y_true
mine = ze22[10] #set the new labels we obtained as y_pred
purity_score(truer,mine) #run our purity score function to get the purity score␣
 ↪of our sclustering
```

[6]:
```
shuttle = df2[[0,1,2,3,4,5,6,7,8]]
shuttle
```

[6]:

```
         0   1    2  3   4   5   6    7    8
0       50  21   77  0  28   0  27   48   22
1       55   0   92  0   0  26  36   92   56
2       53   0   82  0  52  -5  29   30    2
3       37   0   76  0  28  18  40   48    8
4       37   0   79  0  34 -26  43   46    2
…      ..  ..   … ..  ..  ..  ..   …    …
57995   80   0   84  0 -36 -29   4  120  116
57996   55   0   81  0 -20  25  26  102   76
57997   55   0   77  0  12 -22  22   65   42
57998   37   0  103  0  18 -16  66   85   20
57999   56   2   98  0  52   1  42   46    4

[58000 rows x 9 columns]
```

[ ]:

```
###start of search procedure###
yo =dbscanplot1(shuttle,.66,10) #gets the number of clusters and outlier␣
 ↪percentage
yo1 =dbscanplot2(shuttle,.66,10) #gets the cluster_df which holds labels for␣
 ↪the clusters

ze1 = shuttle #initilize the ZSHUT dataset
ze1[10]=yo1 #add the clusters_df to our zshut dataset
ze22 = ze1[ze1[10] != -1] #wherever the cluster is labeled -1 aka an outlier we␣
 ↪will remove from the dataset

truer = ze22[9] #set original cluster labels as y_true
mine = ze22[10] #set the new labels we obtained as y_pred
purity_score(truer,mine) #run our purity score function to get the purity score␣
 ↪of our sclustering
```

[10]:

```
yo =dbscanplot1(shuttle,10,10) #gets the number of clusters and outlier␣
 ↪percentage
```

```
Counter({0: 54172, 1: 2295, 2: 882, -1: 615, 3: 24, 4: 12})
number of clusters: 5
```

[11]:

```
yo1 =dbscanplot2(shuttle,10,10) #gets the cluster_df which holds labels for the␣
 ↪clusters
```

[19]:

```
ze1 = df2 #initilize the ZSHUT dataset
ze1[10]=yo1 #add the clusters_df to our zshut dataset
ze22 = ze1[ze1[10] != -1] #wherever the cluster is labeled -1 aka an outlier we␣
 ↪will remove from the dataset
truer = ze22[9] #set original cluster labels as y_true
mine = ze22[10] #set the new labels we obtained as y_pred
```

```
purity_score(truer,mine) #run our purity score function to get the purity score␣
 ↪of our sclustering
```

[19]: 0.84522087653568