In [5]:

```
import csv
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from numpy.random import seed
from sklearn.model_selection import train_test_split
from keras.layers import Input, Dense
from keras.models import Model


#I've converted the .data file to .csv which is read and replaced ? to NaN in the files
COLUMNS_COUNT = 2

with open('water-treatment.data', 'r') as f:
    columns = [next(f).strip() for line in range(COLUMNS_COUNT)]
temp_df = pd.read_csv('water-treatment.data', skiprows=COLUMNS_COUNT, header=None, delimiter=';', skip_blank_lines=True)
even_df = temp_df.iloc[::2].reset_index(drop=True)
odd_df = temp_df.iloc[1::2].reset_index(drop=True)
df = pd.concat([even_df, odd_df], axis=1)
df.columns = columns
df.to_csv('out.csv', index=False)
text = open("out.csv", "r")
text = ''.join([i for i in text]) \
    .replace("?", "NaN")
x = open("out.csv","w")
x.writelines(text)
x.close()

reader=pd.read_csv('water-treatment.csv',header=None,delimiter=',');
df=pd.DataFrame(reader)
# print('Before Cleaning Up the DataSet\n')
# print(df)

#Calculating the Median of each Column and Replacing "NaN" with the Corresponding Median values
for i in range(1,39):
    mean = df.loc[:,i].mean()
    # print('The mean of column :'+str(i))
    # print(mean)
    df.loc[:,i].fillna(mean, inplace=True)

# print('After Cleaning Up the DataSet\n')
# print(df);

for i in range(1,39):
    for j in range(0,527):
        mean=df.loc[:,i].mean();
        stdevi=df.loc[:,i].std();
        df.loc[j,i]=(df.loc[j,i]-mean)/stdevi;
```

```
# print('After Normalization of the DataSet\n')
# print(df)

#Dropping the Date Column in the Dataset
# print('After Dropping the first date column of the DataSet\n')
df.drop(df.columns[0], axis=1, inplace=True)
# print('After Cleaning the Dataset')
# print(df)

# Implementing K-Means with Optimal 'K' Value
kmeans = KMeans(n_clusters=4, init='k-means++', max_iter=300, n_init=10, random_state=0)
pred_y = kmeans.fit_predict(df)


# Adjusting the Clustering output from 0-3 to 1-4
for i in range(len(pred_y)):
    if pred_y[i]==0:
        pred_y[i]=1
    elif pred_y[i]==1:
        pred_y[i]=2
    elif pred_y[i]==2:
        pred_y[i]=3
    else:
        pred_y[i]=4

# Adjusting the Output to the desired form so that the Clusters get renamed and appear in
 order
l1=[]
l2=[]
cnt=0
for k in pred_y:
    if not k in l1:
        l1.append(k)
        cnt=cnt+1
        l2.append(cnt)
for k in range(len(pred_y)):
    for k1 in range(len(l1)):
        if (pred_y[k]==l1[k1]):
            pred_y[k]=l2[k1]
            break

print('Clustering Output After Ordering In Specified Order')
print(pred_y)


ncol = 38
X_train, X_test, Y_train, Y_test = train_test_split(df,pred_y, train_size = 0.6, random_st
ate = seed(50))

input_dim = Input(shape = (ncol, ))
# DEFINE THE DIMENSION OF ENCODER ASSUMED 3
encoding_dim = 3
# DEFINE THE ENCODER LAYER
encoded = Dense(encoding_dim, activation = 'relu')(input_dim)
# DEFINE THE DECODER LAYER
decoded = Dense(ncol, activation = 'sigmoid')(encoded)
# COMBINE ENCODER AND DECODER INTO AN AUTOENCODER MODEL
```

```
autoencoder = Model(input = input_dim, output = decoded)
# CONFIGURE AND TRAIN THE AUTOENCODER
autoencoder.compile(optimizer = 'adadelta', loss = 'binary_crossentropy')
autoencoder.fit(X_train, X_train, nb_epoch = 50, batch_size = 100, shuffle = True, validat
ion_data = (X_test, X_test))
# THE ENCODER TO EXTRACT THE REDUCED DIMENSION FROM THE ABOVE AUTOENCODER
encoder = Model(input = input_dim, output = encoded)
encoded_input = Input(shape = (encoding_dim, ))
encoded_out = encoder.predict(df)
encoded_out[0:2]

print('Output after ENCODING\n')
print(encoded_out)
print('The Reduced Dimensions After AUTO ENCODING\n')
print(encoded_out.shape)
```

```
Clustering Output After Ordering In Specified Order
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
 1 2 2 1 2 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 2 1 1 2
 2 2 2 2 2 1 1 1 1 2 2 1 1 1 1 1 2 2 2 2 1 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 3 3 4 4 4 4]
```

```
/srv/conda/envs/notebook/lib/python3.7/site-packages/ipykernel_launcher.py:10
9: UserWarning: Update your `Model` call to the Keras 2 API: `Model(inputs=Te
nsor("in..., outputs=Tensor("de...)`
/srv/conda/envs/notebook/lib/python3.7/site-packages/ipykernel_launcher.py:11
2: UserWarning: The `nb_epoch` argument in `fit` has been renamed `epochs`.
```

```
Train on 316 samples, validate on 211 samples
Epoch 1/50
316/316 [==============================] - 0s 596us/step - loss: 0.6585 - val
_loss: 0.6488
Epoch 2/50
316/316 [==============================] - 0s 30us/step - loss: 0.6261 - val_
loss: 0.6219
Epoch 3/50
316/316 [==============================] - 0s 35us/step - loss: 0.5794 - val_
loss: 0.5853
Epoch 4/50
316/316 [==============================] - 0s 134us/step - loss: 0.5171 - val
_loss: 0.5290
Epoch 5/50
316/316 [==============================] - 0s 32us/step - loss: 0.4163 - val_
loss: 0.4538
Epoch 6/50
316/316 [==============================] - 0s 36us/step - loss: 0.2773 - val_
loss: 0.3576
Epoch 7/50
316/316 [==============================] - 0s 33us/step - loss: 0.1259 - val_
loss: 0.2038
Epoch 8/50
316/316 [==============================] - 0s 39us/step - loss: -0.0932 - val
_loss: 0.0149
Epoch 9/50
316/316 [==============================] - 0s 166us/step - loss: -0.3514 - va
l_loss: -0.2022
Epoch 10/50
316/316 [==============================] - 0s 33us/step - loss: -0.6714 - val
_loss: -0.4768
Epoch 11/50
316/316 [==============================] - 0s 33us/step - loss: -1.0888 - val
_loss: -0.8138
Epoch 12/50
316/316 [==============================] - 0s 31us/step - loss: -1.5669 - val
_loss: -1.1546
Epoch 13/50
316/316 [==============================] - 0s 32us/step - loss: -2.0437 - val
_loss: -1.4964
Epoch 14/50
316/316 [==============================] - 0s 154us/step - loss: -2.5264 - va
l_loss: -1.9939
Epoch 15/50
316/316 [==============================] - 0s 34us/step - loss: -3.1828 - val
_loss: -2.3336
Epoch 16/50
316/316 [==============================] - 0s 32us/step - loss: -3.6218 - val
_loss: -2.7474
Epoch 17/50
316/316 [==============================] - 0s 36us/step - loss: -4.1499 - val
_loss: -3.2218
Epoch 18/50
316/316 [==============================] - 0s 34us/step - loss: -4.7460 - val
_loss: -3.6761
Epoch 19/50
316/316 [==============================] - 0s 162us/step - loss: -5.3288 - va
```

```
l_loss: -4.1179
Epoch 20/50
316/316 [==============================] - 0s 39us/step - loss: -5.9043 - val
_loss: -4.5370
Epoch 21/50
316/316 [==============================] - 0s 35us/step - loss: -6.4888 - val
_loss: -4.9351
Epoch 22/50
316/316 [==============================] - 0s 33us/step - loss: -7.0357 - val
_loss: -5.3163
Epoch 23/50
316/316 [==============================] - 0s 172us/step - loss: -7.5076 - va
l_loss: -5.6449
Epoch 24/50
316/316 [==============================] - ETA: 0s - loss: -10.072 - 0s 36us/
step - loss: -7.8684 - val_loss: -5.9158
Epoch 25/50
316/316 [==============================] - 0s 32us/step - loss: -8.1819 - val
_loss: -6.1697
Epoch 26/50
316/316 [==============================] - 0s 32us/step - loss: -8.4713 - val
_loss: -6.3884
Epoch 27/50
316/316 [==============================] - 0s 41us/step - loss: -8.7160 - val
_loss: -6.5464
Epoch 28/50
316/316 [==============================] - 0s 159us/step - loss: -8.8860 - va
l_loss: -6.6516
Epoch 29/50
316/316 [==============================] - 0s 39us/step - loss: -8.9897 - val
_loss: -6.7597
Epoch 30/50
316/316 [==============================] - 0s 37us/step - loss: -9.0834 - val
_loss: -6.8637
Epoch 31/50
316/316 [==============================] - 0s 32us/step - loss: -9.1681 - val
_loss: -6.9308
Epoch 32/50
316/316 [==============================] - 0s 40us/step - loss: -9.2286 - val
_loss: -6.9894
Epoch 33/50
316/316 [==============================] - 0s 35us/step - loss: -9.2741 - val
_loss: -7.0414
Epoch 34/50
316/316 [==============================] - 0s 32us/step - loss: -9.3158 - val
_loss: -7.1048
Epoch 35/50
316/316 [==============================] - 0s 39us/step - loss: -9.3638 - val
_loss: -7.1534
Epoch 36/50
316/316 [==============================] - 0s 40us/step - loss: -9.4036 - val
_loss: -7.1982
Epoch 37/50
316/316 [==============================] - 0s 172us/step - loss: -9.4403 - va
l_loss: -7.2287
Epoch 38/50
316/316 [==============================] - 0s 40us/step - loss: -9.4629 - val
```

```
_loss: -7.2551
Epoch 39/50
316/316 [==============================] - ETA: 0s - loss: -8.64 - 0s 44us/st
ep - loss: -9.4876 - val_loss: -7.2882
Epoch 40/50
316/316 [==============================] - 0s 39us/step - loss: -9.5157 - val
_loss: -7.3124
Epoch 41/50
316/316 [==============================] - 0s 152us/step - loss: -9.5380 - va
l_loss: -7.3317
Epoch 42/50
316/316 [==============================] - 0s 36us/step - loss: -9.5557 - val
_loss: -7.3516
Epoch 43/50
316/316 [==============================] - 0s 37us/step - loss: -9.5771 - val
_loss: -7.3734
Epoch 44/50
316/316 [==============================] - 0s 32us/step - loss: -9.5969 - val
_loss: -7.3939
Epoch 45/50
316/316 [==============================] - 0s 35us/step - loss: -9.6136 - val
_loss: -7.4123
Epoch 46/50
316/316 [==============================] - 0s 36us/step - loss: -9.6321 - val
_loss: -7.4327
Epoch 47/50
316/316 [==============================] - 0s 38us/step - loss: -9.6522 - val
_loss: -7.4488
Epoch 48/50
316/316 [==============================] - 0s 34us/step - loss: -9.6685 - val
_loss: -7.4621
Epoch 49/50
316/316 [==============================] - 0s 36us/step - loss: -9.6844 - val
_loss: -7.4840
Epoch 50/50
316/316 [==============================] - 0s 40us/step - loss: -9.7023 - val
_loss: -7.4983
Output after ENCODING

[[  5.3291535   1.5111139   0.       ]
 [ 12.833196    3.6987188   0.       ]
 [  2.474523    0.9178072   0.       ]
 ...
 [228.05164    33.823265    0.       ]
 [290.23547    46.11997     0.       ]
 [366.95267    63.122272    0.       ]]
The Reduced Dimensions After AUTO ENCODING

(527, 3)


/srv/conda/envs/notebook/lib/python3.7/site-packages/ipykernel_launcher.py:11
4: UserWarning: Update your `Model` call to the Keras 2 API: `Model(inputs=Te
nsor("in..., outputs=Tensor("de...)`
```