LINEAR ALGEBRA FOR AI AND ML
AI61003

Assignment-1

Utsav Mehta
20CS10069

Question - 4:

(a) The randomly generated matrix A of size 8x6 is:

```
A =
 [[1 1 0 1 0 1]
 [1 0 1 0 0 1]
 [1 0 0 0 1 0]
 [0 0 0 0 0 0]
 [0 0 1 1 0 1]
 [1 1 1 1 0 0]
 [0 1 1 1 1 0]
 [0 1 0 1 0 1]]
```

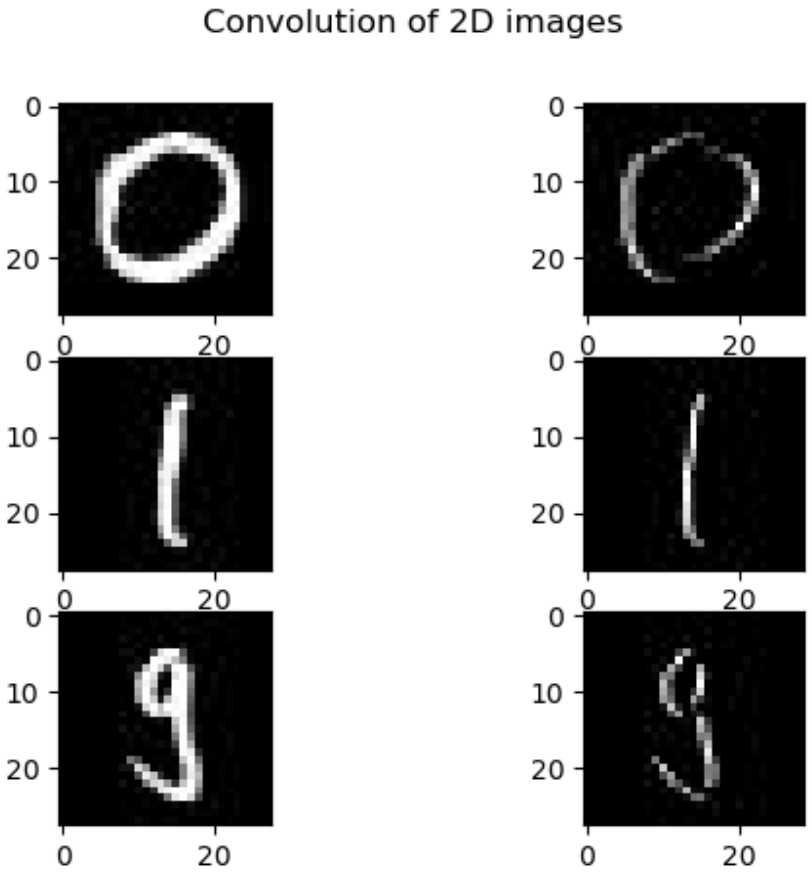
The kernel matrix B is:

```
B =
 [[-1  1]
 [ 1 -1]]
```

The convolution matrix C (= A*B) is:

```
C =
 [[-1  0  1 -1  1 -1  1]
 [ 0  1 -2  2 -1  0  0]
 [ 0  0  1 -1 -1  2 -1]
 [ 1 -1  0  0  1 -1  0]
 [ 0  0 -1  0  1 -1  1]
 [-1  0  1  0  0  1 -1]
 [ 1 -1  0  0 -1  1  0]
 [ 0  0  1 -1  1 -2  1]
 [ 0  1 -1  1 -1  1 -1]]
```

(b) The convolution plots of images of 0,1,9 taken from the MNIST dataset, with the kernel D (= [1 -1]) are as shown:



The operation A*D performs **edge detection** on the given image. It classifies those pixels as edges where a **lack of coherence among the intensity** of the neighboring pixels is observed, and hence the given convolution kernel D is used to detect edges in the original image A.

The files related to this question are -

- q4_20CS10069.py
- 0.jpg
- 1.jpg
- 9.jpg
- q4_q11_report_20CS10069.pdf

Code snippet for question - 4:

```python
from random import choice
import numpy as np
from matplotlib import pyplot
import cv2


def convolution_2D(A, B):
    m, n = A.shape
    p, q = B.shape

    C = np.array([[0] * (n + q - 1) for _ in range(m + p - 1)])
    for r in range(1, m+p):
        for s in range(1, n+q):
            for i in range(1, m+1):
                k = r + 1 - i
                if k > 0 and k <= p:
                    for j in range(1, n+1):
                        l = s + 1 - j
                        if l > 0 and l <= q:
                            C[r-1][s-1] += A[i-1][j-1] * B[k-1][l-1]
    return C


def main():

    ########------------------ PART A ------------------########

    A = np.array([[choice([0, 1]) for _ in range(6)] for _ in range(8)])
    B = np.array([[-1, 1], [1, -1]])

    print("A = \n", A)
    print("B = \n", B)
    C = convolution_2D(A, B)
    print("C = \n", C)
```

```python
    ########------------------ PART B ------------------########
    B = [[1, -1]]
    D = np.array(B)

    imgs = ["0.jpg", "1.jpg", "9.jpg"]
    pos = 1

    for img in imgs:
        A = cv2.imread(img, 0)
        C = convolution_2D(A, D)

        for i in range(C.shape[0]):
            for j in range(C.shape[1]):
                if C[i][j] < 0:
                    C[i][j] = 0
                elif C[i][j] > 255:
                    C[i][j] = 255

        pyplot.subplot(3, 2, pos)
        pyplot.imshow(A, cmap='gray')
        pyplot.subplot(3, 2, pos+1)
        pyplot.imshow(C, cmap='gray')
        pos+=2

    pyplot.suptitle("Convolution of 2D images")
    pyplot.show()

if __name__ == "__main__":
    main()
```

Question - 11:

Since we choose 100 images for each of the 10 digits, hence number of training examples (N) = **1000**
Since each training example consists of 28x28 images, hence number of features (n) = 28*28 = **784**

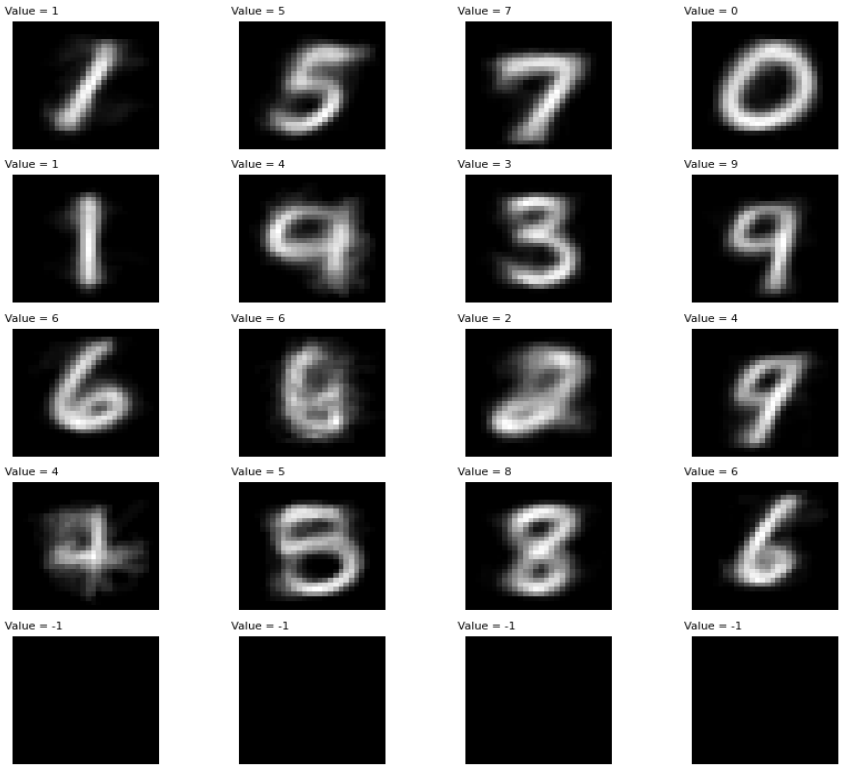
```
The number of data points (N) = 1000
The size of each data point (n) = 784
```

The **convergence criterion** chosen is -
If the absolute difference between the latest and the previous “cluster_cost” or J is **less than 10^-6**, declare convergence.

1. For random initialization of cluster representatives –
    a. Training the dataset for k=20 and plotting the cluster representatives after convergence, also maintaining the count of iterations -

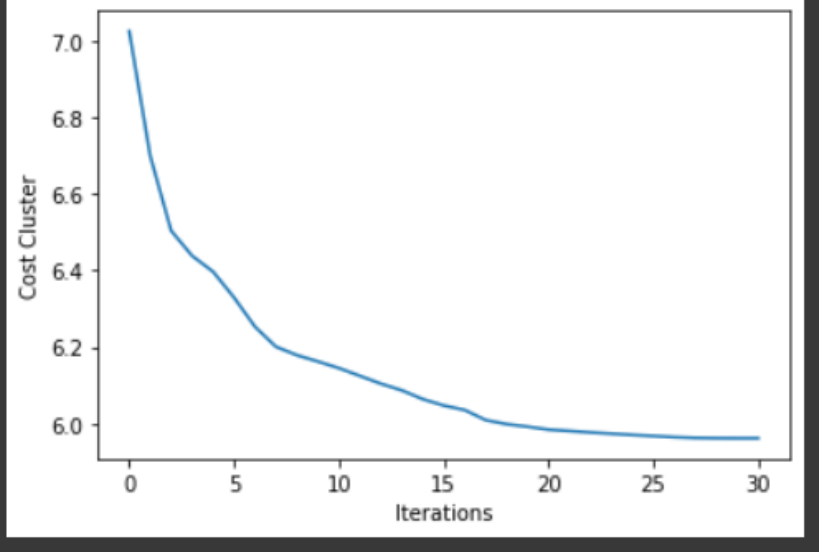
```
Converged after 31 iterations.
```

Cluster - Representatives



b. Choosing 50 random mnist images as test samples and calculating the accuracy of the dataset -

```
Accuracy of the model is: 0.66
```

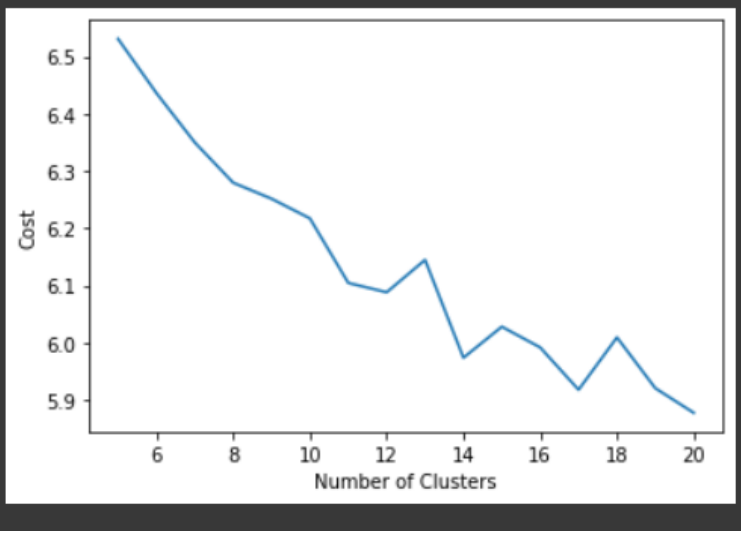


Hence accuracy of the model is 0.66 (or 66%).

c. Obtaining the optimal value of k from 5 to 20, by plotting the values of J_clust (or “cost_cluster”) for different values of k-

```
Number of Clusters (k) = 5, Cost (J) = 6.531
Number of Clusters (k) = 6, Cost (J) = 6.4362
Number of Clusters (k) = 7, Cost (J) = 6.3498
Number of Clusters (k) = 8, Cost (J) = 6.2794
Number of Clusters (k) = 9, Cost (J) = 6.2513
Number of Clusters (k) = 10, Cost (J) = 6.217
Number of Clusters (k) = 11, Cost (J) = 6.1041
Number of Clusters (k) = 12, Cost (J) = 6.0878
Number of Clusters (k) = 13, Cost (J) = 6.1445
Number of Clusters (k) = 14, Cost (J) = 5.9733
Number of Clusters (k) = 15, Cost (J) = 6.0276
Number of Clusters (k) = 16, Cost (J) = 5.9913
Number of Clusters (k) = 17, Cost (J) = 5.9174
Number of Clusters (k) = 18, Cost (J) = 6.0089
Number of Clusters (k) = 19, Cost (J) = 5.92
Number of Clusters (k) = 20, Cost (J) = 5.877
```

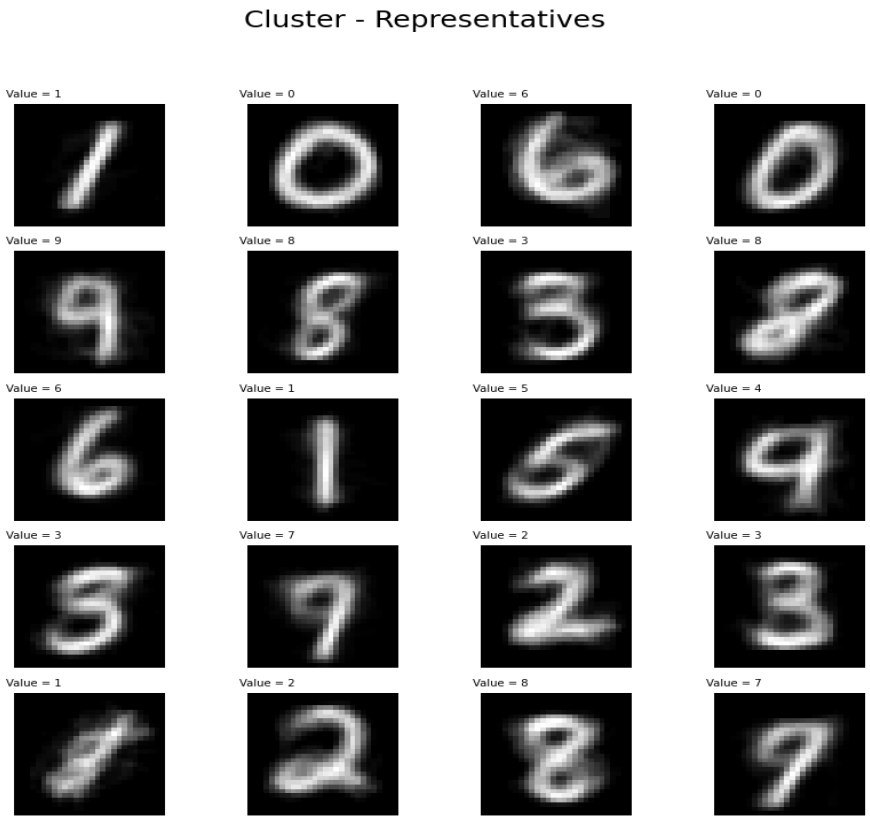
```
Min Cost Clustering (J_min) = 5.877 for k = 20
```



The minimum value of J occurs for Number of clusters (k) = 20.

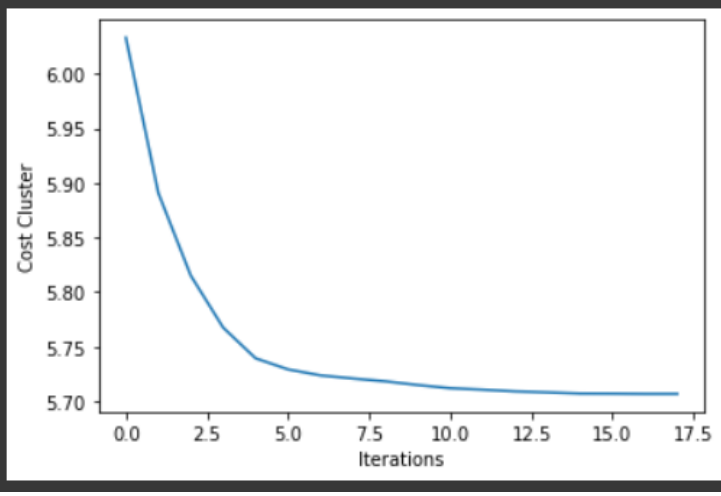
2. For initialization of cluster representatives from the dataset –
    a. Training the dataset for k=20 and plotting the cluster representatives after convergence, also maintaining the count of iterations -

```
Converged after 18 iterations.
```



    b. Choosing 50 random mnist images as test samples and calculating the accuracy of the dataset -

```
Accuracy of the model is: 0.66
```

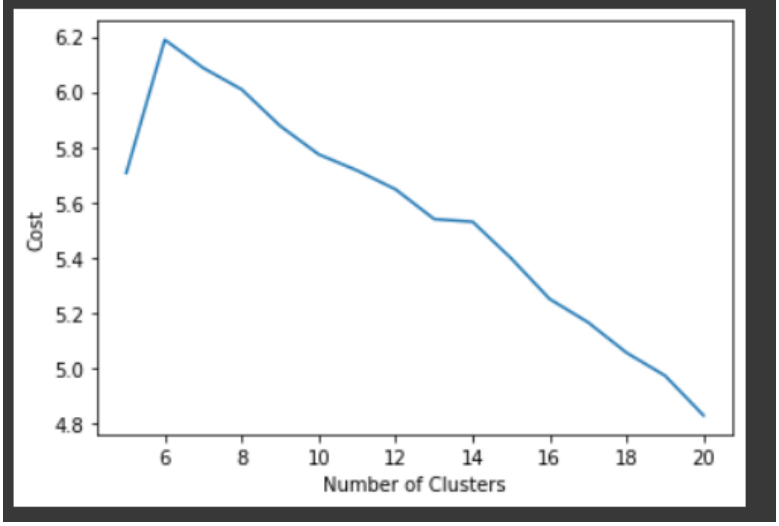


Hence the accuracy of the model is 0.66 (= 66%).

c. Obtaining the optimal value of k from 5 to 20, by plotting the values of J_clust (or “cost_cluster”) for different values of k-

```
Number of Clusters (k) = 5, Cost (J) = 5.7068
Number of Clusters (k) = 6, Cost (J) = 6.1882
Number of Clusters (k) = 7, Cost (J) = 6.0863
Number of Clusters (k) = 8, Cost (J) = 6.0085
Number of Clusters (k) = 9, Cost (J) = 5.8766
Number of Clusters (k) = 10, Cost (J) = 5.7739
Number of Clusters (k) = 11, Cost (J) = 5.7156
Number of Clusters (k) = 12, Cost (J) = 5.6472
Number of Clusters (k) = 13, Cost (J) = 5.54
Number of Clusters (k) = 14, Cost (J) = 5.5302
Number of Clusters (k) = 15, Cost (J) = 5.3982
Number of Clusters (k) = 16, Cost (J) = 5.2512
Number of Clusters (k) = 17, Cost (J) = 5.1666
Number of Clusters (k) = 18, Cost (J) = 5.0568
Number of Clusters (k) = 19, Cost (J) = 4.9739
Number of Clusters (k) = 20, Cost (J) = 4.8301
```

```
Min Cost Clustering (J_min) = 4.8301 for k = 20
```



The minimum value of J occurs for Number of Clusters (k) = 20.

CONCLUSION –

- ❖ It should be noted that although the accuracy of the model is same for both case(1) and case(2), the algorithm converges much faster in case(2), i.e., it requires less iterations to satisfy the convergence criterion.
- ❖ Also, the J_clust value is lower when the initial cluster representatives are chosen from the dataset rather than random assignment.
- ❖ Thus, it is clear that there is an improvement in the performance of the model, in case(2), over convergence and J value.

The files related to this question are -

- q11_20CS10069.ipynb
- q4_q11_report_20CS10069.pdf
- Note - all the graphs/images/plots shown for this question can be found in the .ipynb file submitted.