# FIT5201 -MACHINE LEARNING

## ASSESSMENT 2

Keerthana Muralitharan
Kmur0015@student.monash.edu
Student ID : 30159474

# Question 1 [EM for Document Clustering, 40 Marks]

1. Derive **Expectation** and **Maximization** steps of the hard-EM algorithm for Document Clustering, show your work in your submitted PDF report. In particular, include all model parameters that should be learnt and the exact expression (using the same math convention that we saw in the Module 4) that should be used to update these parameters during the learning process (ie., E step, M step and assignments).

**Solution:**

The EM algorithm is implemented based on two steps

**1. Expectation Step**

Gamma $\gamma(z_{n,k})$ – is the responsibility factor (estimated by **Mixing Component ($\rho_k$)** and the **Word proportion parameter ($\mu_{k,w}$)).**

These parameters are defined as

The mixing components: $\varphi_k = \frac{N_k}{N}$ where $N_k := \sum_{n=1}^{N} \gamma(z_{n,k})$

The word proportion parameters for each cluster: $\mu_{k,w} = \frac{\sum_{n=1}^{N} \gamma(z_{n,k}) c(w,d_n)}{\sum_{w' \in \mathcal{A}} \sum_{n=1}^{N} \gamma(z_{n,k}) c(w',d_n)}$

**2. Maximisation step**

In this step based on the Gamma calculated above, the new values of **Mixing Component ($\rho_k$)** and the **Word proportion parameter ($\mu_{k,w}$)** are calculated.

These values are again used in the E-step above to re-estimate the Gamma value and the process repeats over till the convergence is reached.

**Hard EM Algorithm:**

In Hard EM, each data point is assigned to one of clusters with probability of 1 and all other data points will be assigned to clusters with value as 0.

**The Algorithm is as follows:**

1. Choose an initial setting for the parameters of $\Theta^{old}$

   - $\Theta^{old}$ parameter is set initially ($N_k$, $\rho_k$, $\mu_{k,w}$)

   - From these initial values of $\Theta^{old}$, the value of **Gamma $\gamma(z_{n,k})$** is calculated

2. When the convergence is not met:

**E step**:

   1. Update $\gamma(z_{n,k})$ value based on the $\Theta^{old}$ **parameters.**

2. Apply the Hard EM algorithm approach that each data point in the cluster is assigned only to one class with a probability of 1 and others 0, by using the **ARGMAX** function that assigns 1 to the class with maximum probability.

$\gamma(z_{n,k})$ <- **argmax**$_z$ $p(z_{n,k} = 1 \mid d_n, \Theta^{old})$

**M Step:**

1. estimate **Mixing Component ($\rho_k$)** and the **Word proportion parameter ($\mu_{k,w}$)** from the calculated **Gamma $\gamma(z_{n,k})$**

**Mixing Component ($\rho_k$) -** To estimate this parameter, the $N_k$ (cluster size) is estimated first for each of the clusters. Where N is the total number of documents

**The mixing component is $\rho_k = N_k/N$**

**Word proportion parameter ($\mu_{k,w}$) -** proportion of a word in a cluster based on all the words in that cluster across all the documents.

$$\mu_{k,w} = \frac{\text{proportion of a word in all documents in that cluster}}{\text{proportion of all the words in that cluster for all the documents}}$$

2. After calculating these two values, they are used to estimate the $\gamma(z_{n,k})$ again repeat the process until the convergence is reached.

- Set $\Theta^{old}$ <- $\Theta^{new}$

**4. Perform a PCA on the clusterings that you get based on the hard-EM and soft-EM in the same way we did in Activity 4.2. Then, visualize the obtained clusters with different colors where x and y axes are the first two principal components (similar to Activity 4.2). Attach the plots to your PDF report and report how and why the hard and soft-EM are different, based on your plots in the report.**

**Solution:**

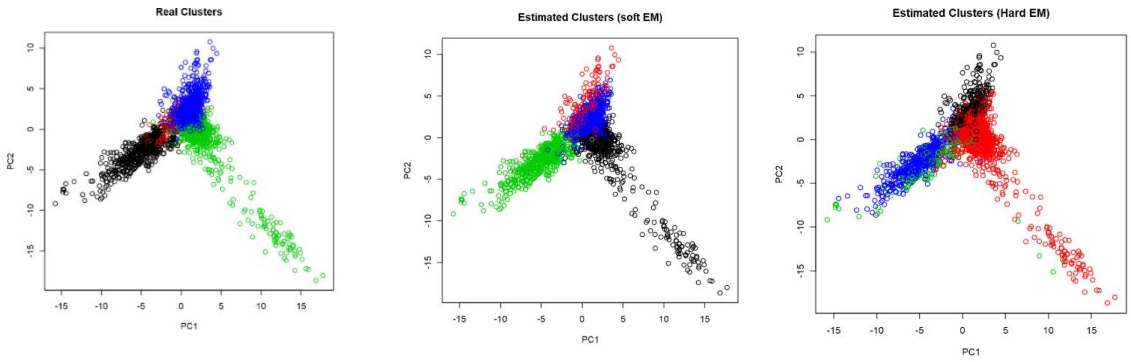



*Fig 1. Document Clustering [Original Clusters, Soft EM, Hard EM]*

As we could see the distribution is same across the soft EM document and hard EM document clustering. Also, we could see that the train objects for both the methods seems to be decreasing to the epoch count value. These 2 methods return the negative log likelihood (i.e) log P(counts|model).In soft EM we have all the 4 clusters are distinguishable as it uses probability for every datapoint . While the

Hard- EM classifies into 2 possibilities with the cluster probability with value as 1 and 0, so there are many chances of datapoints in the clusters to get overlapped after PCA.

## Question 2 [Neural Network's Decision Boundary, 30 Marks]

I. Load **Task2B_train.csv** and **Task2B_test.csv** sets, plot the training data with classes are marked with different colors, and attach the plot to your PDF report.

**Solution:**

The datasets are read and the train dataset alone is plotted with its classes marked in different colors and symbols are represented for class 0 as "-" and class 1 as "+".The train dataset has null values also, which are not plotted in the graph.

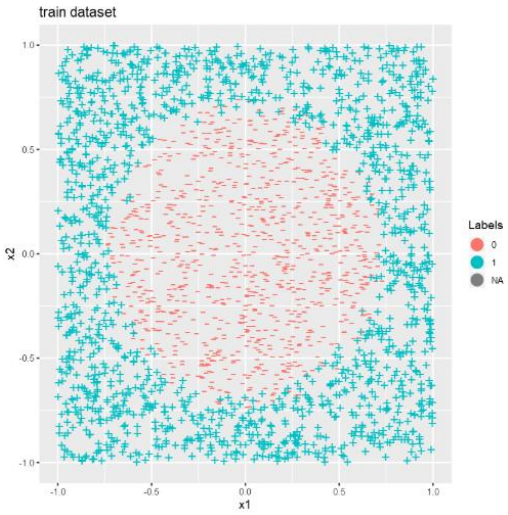



*Fig 2. Train dataset*

**Question II.**

Train two perceptron models on the loaded training data by setting the learning rates η to .01 and .09 respectively, using a code from Activity **3.1**. Calculate the test errors of two models and find the best η and its corresponding model, then plot the test data while the points are colored with their estimated class labels using the best model that you have selected; attach the plot to your PDF report.

**Inference :**

```
[1] "Mean Error rate - 0.01"
0.519960784313725
[1] "Mean Error rate - 0.09"
0.502235294117647
```

*Fig 3a. Mean Error rate for different learning rate*

Based on the errors for both the learning rates, It is found that the best model is built with higher learning rate – (0.09).Perceptron is a linear decision boundary, so it does not match with the data. This is the plot below

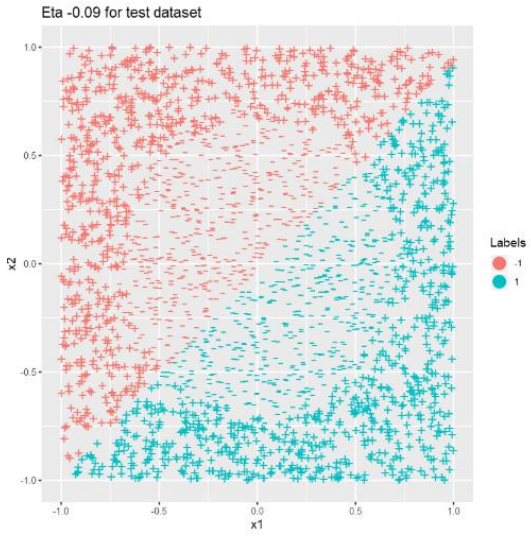
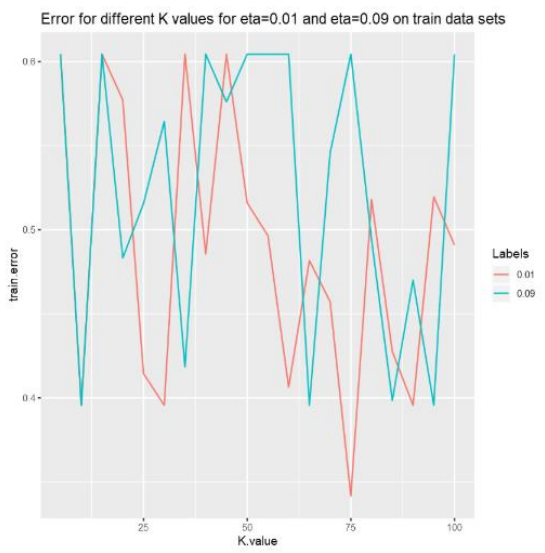



*Fig 3b. Error for 0.09 learning rate*

**Question III.**

For each combination of K (i.e, number of units in the hidden layer) in {5, 10, 15, …, 100} and μ (learning rate) in {0.01, 0.09}, run the 3-layer Neural Network given to you in Activity **5.1** and record testing error for each of them (40 models will be developed, based on all possible combinations). Plot the error for μ 0.01 and 0.09 vs K (one line for μ 0.01 and another line for μ 0.09 in a plot) and attach it to your PDF report. Based on this plot, find the best combination of K and μ and the corresponding model, then plot the test data while the points are coloured with their estimated class labels using the best model that you have selected; attach the plot to your PDF report.

**Inference:** For the test dataset the best model for K and Mu is 75 and 0.01 respectively as it has the least error. The epoch value is set at 100 as it converges at this value. Since it is a non-linear decision boundary it would match with the data and similar with the dataset.



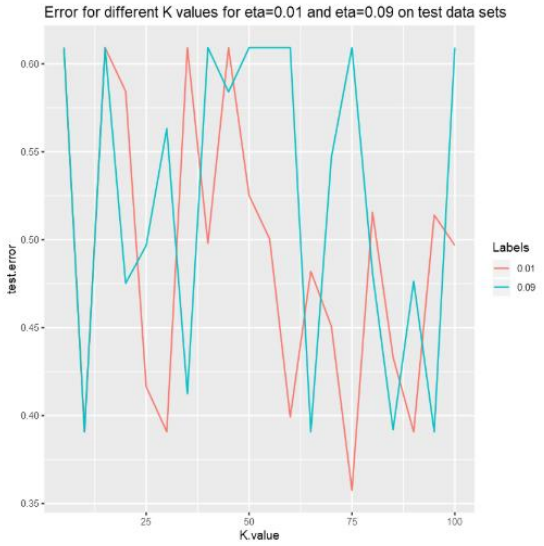




*Fig.4a Training error for eta 0.01 and 0.09*

*Fig.4b Testing error for eta 0.01 and 0.09*

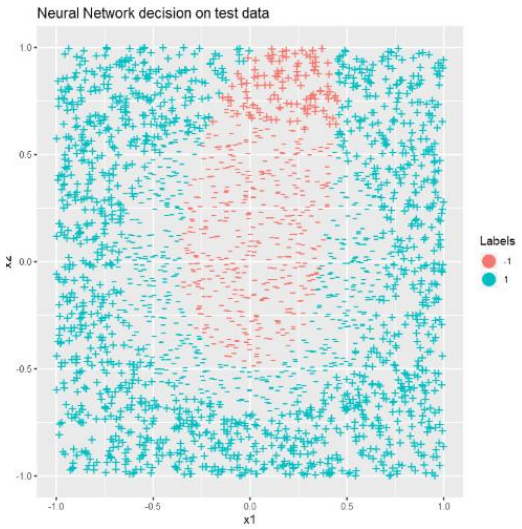



*Fig 4c. Neural network decision on best model*

**Question IV.**

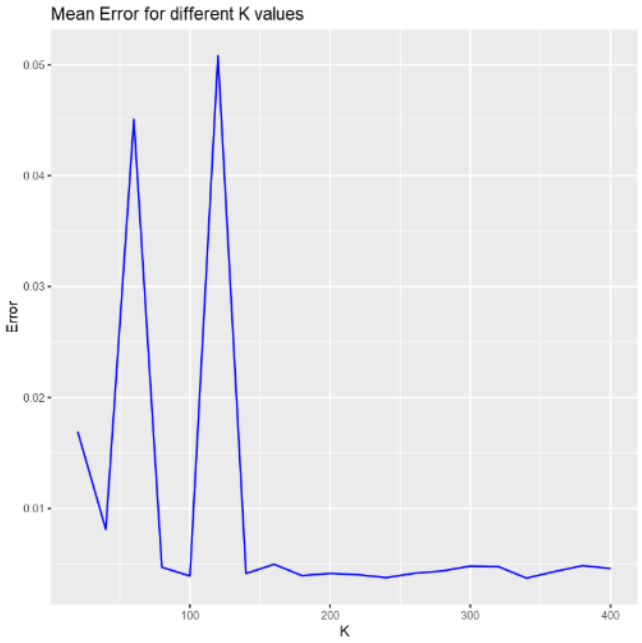
In your PDF report, explain the reason(s) responsible for such difference between perceptron and a 3-layer NN by comparing the plots you generated in Steps II and III.

**Inference :** Here Perceptron is a linear decision boundary , while the 3-layer NN is Neural network decision boundary which is non-linear decision boundary. This is the main reason for which the data will not match for the perceptron and it would match to the original data with the neural networks

## Question 3 [Self Taught Neural Network Learning, 30 Marks]

**Question III**

For each model in Step II, calculate and record the reconstruction error which is simply the average (over all data points while the model is fixed) of Euclidian distances between the input and output of the autoencoder (you can simply use "h2o.anomaly()" function). Plot these values where the x-axis is the number of units in the middle layer and the y-axis is the reconstruction error. Then, save and attach the plot to your PDF report. Explain your findings based on the plot in your PDF report.




| Error | K |
|---|---|
| 0.016905605 | 20 |
| 0.008122241 | 40 |
| 0.045066474 | 60 |
| 0.004698050 | 80 |
| 0.003916243 | 100 |
| 0.050804491 | 120 |
| 0.004133422 | 140 |
| 0.004968323 | 160 |
| 0.003941535 | 180 |
| 0.004147316 | 200 |
| 0.004025722 | 220 |
| 0.003762346 | 240 |
| 0.004153866 | 260 |
| 0.004364088 | 280 |
| 0.004806057 | 300 |
| 0.004759548 | 320 |
| 0.003716971 | 340 |
| 0.004307393 | 360 |
| 0.004845518 | 380 |
| 0.004585019 | 400 |

*Fig 5. Reconstruction Error*

**Inference :** we could see a decrease in the error rates as the K values increases with a considerable fluctuations and it remains the same with less number of fluctuations towards higher values of K, but it goes to convergence.

**Question IV**

IV. Build the 3-layer NN from Activity 5.1 or "h2o.deeplearning" function (make sure you set " autoencoder = FALSE") to build a classification model using all the original attributes from the training set and change the number of its neurons to: 20, 40, 60, 80, .., 400 like Step II. For each model, calculate and record the test error.


| Error_class | K |
|---|---|
| 0.334 | 20 |
| 0.577 | 40 |
| 0.560 | 60 |
| 0.607 | 80 |
| 0.538 | 100 |
| 0.528 | 120 |
| 0.543 | 140 |
| 0.552 | 160 |
| 0.513 | 180 |
| 0.516 | 200 |
| 0.523 | 220 |
| 0.475 | 240 |
| 0.434 | 260 |
| 0.517 | 280 |
| 0.481 | 300 |
| 0.460 | 320 |
| 0.469 | 340 |
| 0.359 | 360 |
| 0.463 | 380 |
| 0.457 | 400 |

*Fig 6. Classification Error*

**Inference :** we could see a decrease in the error rates as the classification as the K values increases with a considerable fluctuations.

**Question VI**

Plot the error rates for the 3-layer neural networks from Step IV and the augmented self-taught networks from Step V, while the x-axis is the number of extra features and y-axis is the classification error. Save and attach the plot to your PDF report. In your pdf, explain how the performance of the 3-layer neural networks and the augmented self-taught networks is different and why they are different or why they are not different, based on the plot.

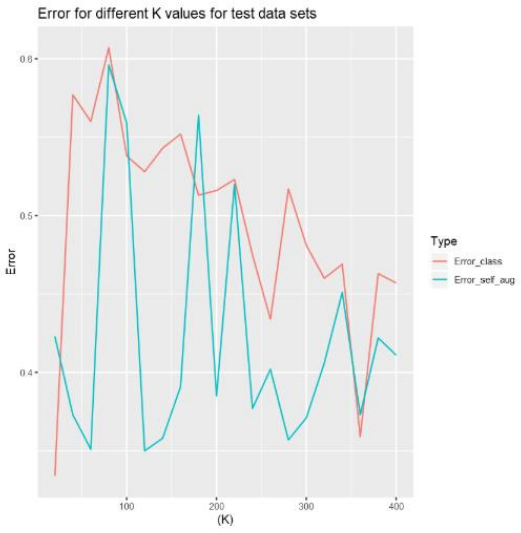



*Fig 7. Error of 3 layer NN and augmented self-taught networks*

**Inference :** As K values increases the error values decreases with huge fluctuations and comparing the 3-layer neural network and augmented self-taught network, we could say that the error rate for 3 layer neural network is higher than the augmented self-taught network and converges that is because augmented self-taught networks has more features (original + extra) which improvises the model and decreases the error.

**Note -** based on the different epoch values and initial weight values we would get different results. Above 3 question and analysis are made on one particular set of values. The values and results are subject to change on different runs of the program.