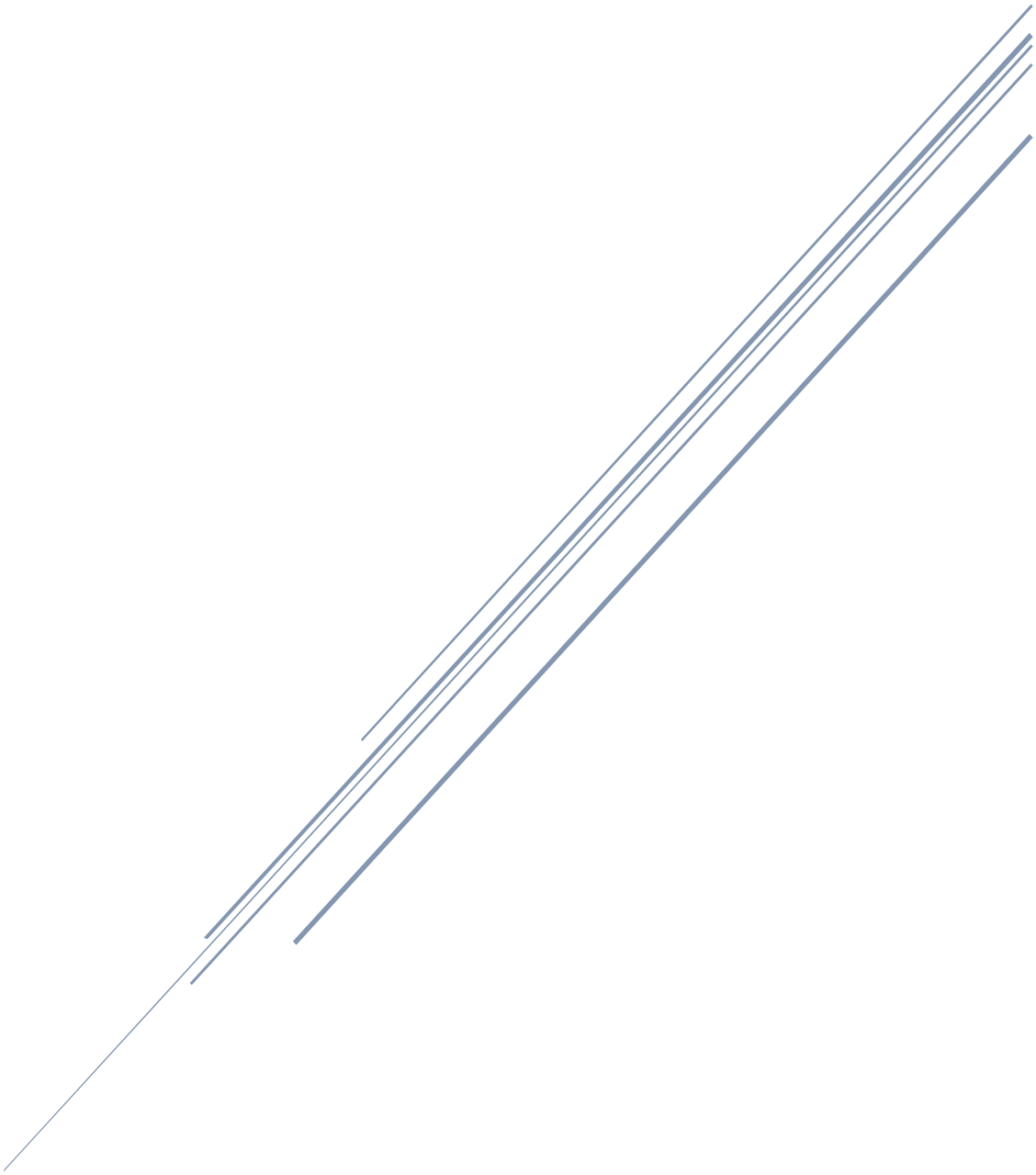
DATA ANALYSIS ALGORITHM

Assignment-2



Mohaab Hassan
29626889

PART-A (Document Clustering)

Question 1 [EM for Document Clustering]

1.Derive Expectation and Maximization steps of the hard-EM algorithm for Document Clustering

Date:
Page:

Derivation for Expectation-Maximization steps for hard clustering:

We begin by assuming our model parameters,

- $\{d_1, d_2, d_3, \ldots d_N\}$ is our collection of documents
- Ψ is probability vector of size K where $\sum_{K=1}^{K} \Psi_K = 1$
- μ_K denotes word proportion i.e number of words belonging to K^{th} face of die where $\sum_{w \in A} \mu_{K,w} = 1$

where A is all words in dictionary

Z → Latent variable

The probability of observed documents is

$$p(d_1, d_2, d_3 \ldots d_n) = \prod_{n=1}^{N} p(d_n)$$

$$= \prod_{n=1}^{N} \sum_{K=1}^{K} p(z_n, K=1, d_n)$$

$$= \prod_{n=1}^{N} \sum_{K=1}^{K} \left(\Psi_K \prod_{w \in A} \mu_{K,w}^{c(w,d_n)} \right)$$

where $c(w, d_n)$ denotes the count of words in a document.

Date: . .
Page:

The posterior probability of a document belonging to a cluster is given by

$$p(z_{n,K}=1 \mid H\theta_n) \Rightarrow \gamma(z_n, K)$$

where $\theta = (H_1, H_2, \cdots H_K)$

$\gamma(z_n, K)$ is given by

$$= \frac{\text{number of words in a document} \times \text{prob of cluster}}{\text{number of words in all clusters}}$$

$$= \frac{\psi_K \prod_{w \in W} H_{k,w}^{c(w,d_n)}}{\prod_{n=1}^{N} \sum_{K=1}^{K} \left(\psi_k \prod_{w \in A} H_{K,w}^{c(w,d_n)} \right)}$$

We then calculate cluster probability which is given by

$$\psi_K = \frac{N_K}{N} \quad \text{where} \quad N_K = \sum_{N=1}^{N} \gamma(z_n, K)$$

Word proportion for each cluster,

$$H_{k,w} = \frac{\sum_{N=1}^{N} \gamma(z_{n,k}) \cdot c(w, d_n)}{\sum_{w \in A} \sum_{N=1}^{N} \gamma(z_{n,K}) \cdot c(w', d_n)}$$

Date: . .
Page:

We now derive the steps for hard max based on our parameters,

- Choose an initial parameter

$$\theta^{old} = (\pi^{old}, \mu_1^{old}, \dots \mu_K^{old})$$

while convergence is not met :

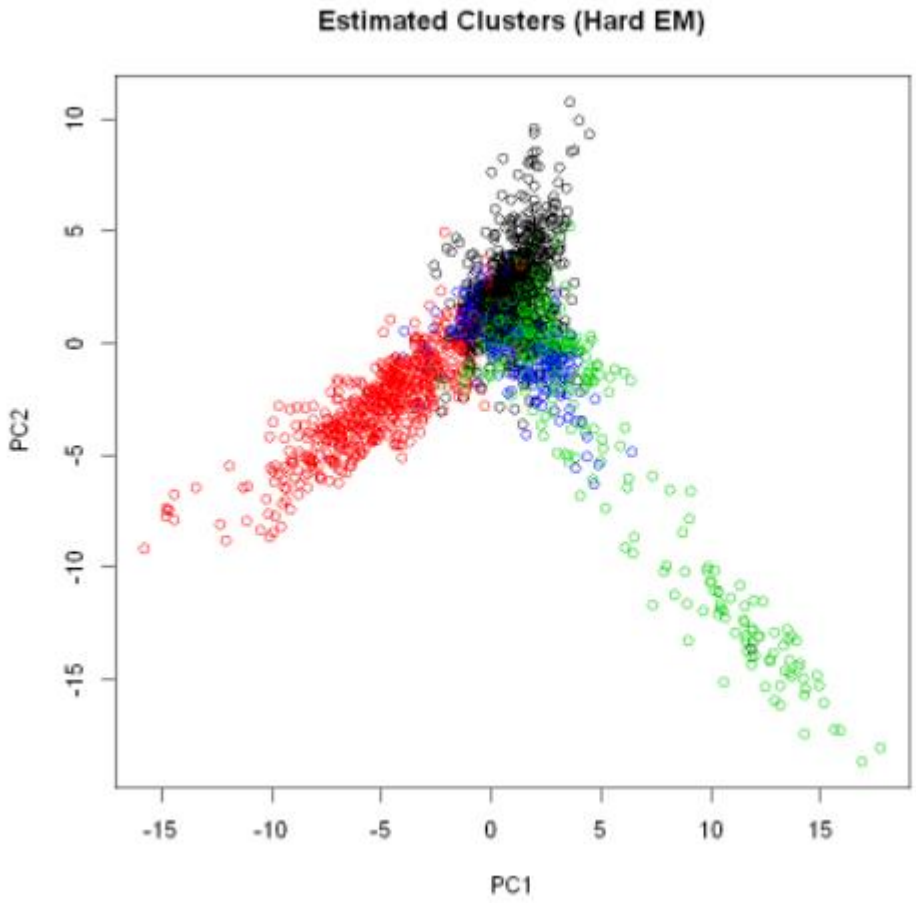
E step : set $z\ \forall_n, \forall_K$: $\underset{z}{\text{argmax}}\ \gamma(z_{n,K})$

M step : set θ^{new} based on above equations

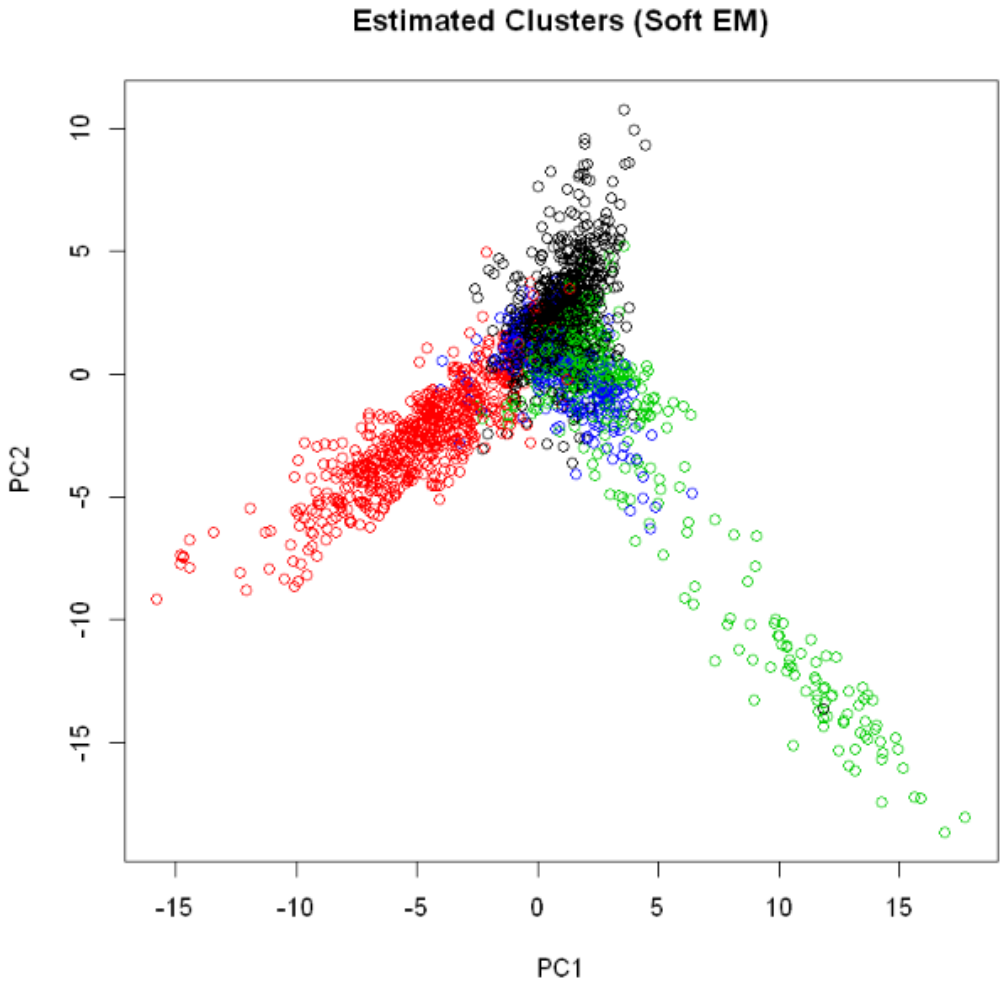
$\theta^{old} \leftarrow \theta^{new}$

2. Perform a PCA on the clustering that you get based on the hard-EM and soft-EM(Number of Cluster K=4)

PCA on Clusters using Hard EM and number of cluster =4



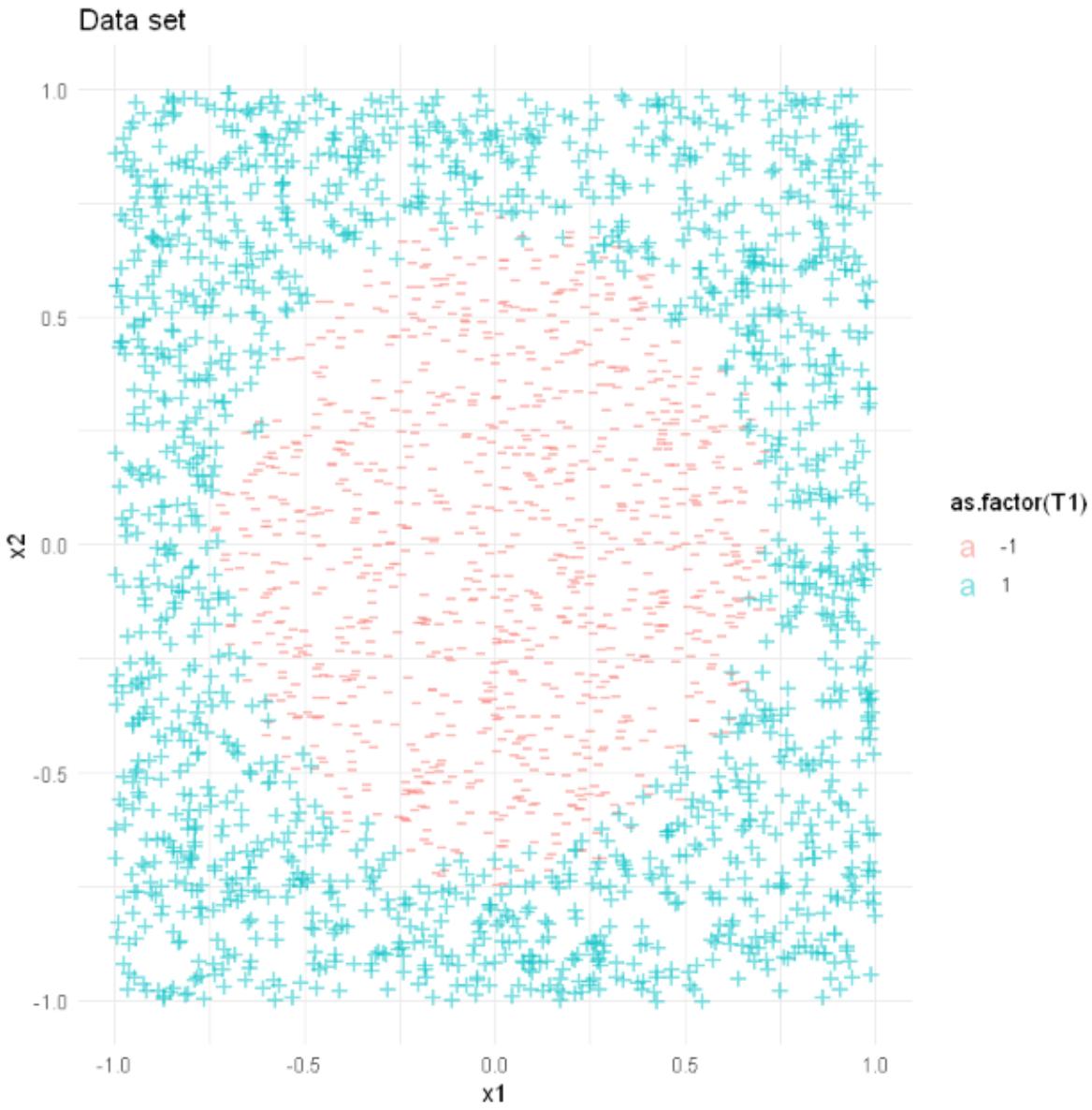
PCA on Clusters using Soft EM and number of cluster =4



Part B. Neural Network vs. Perceptron

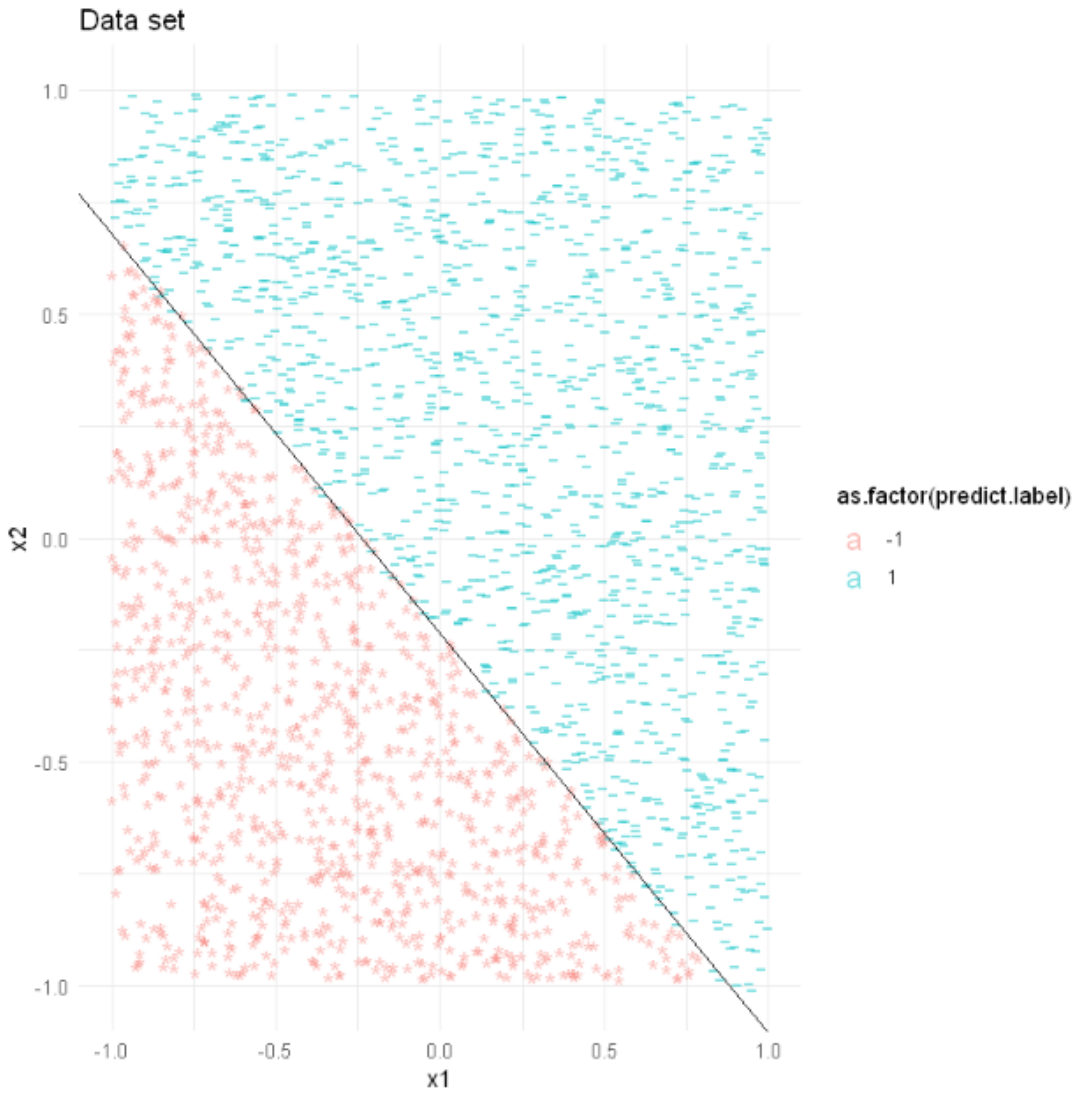
1.Plotting the training data

From the visualization we can see that data is non-linearly separable



2.Classification of Test labels from the predict values of Perceptron

Perceptron assumes the data to be linearly separable therefore we can see it divides the upper half to one class and the lower half to the other class establishing linear boundary. Giving us high error rate of 51.5% which fails to classify half of the labels

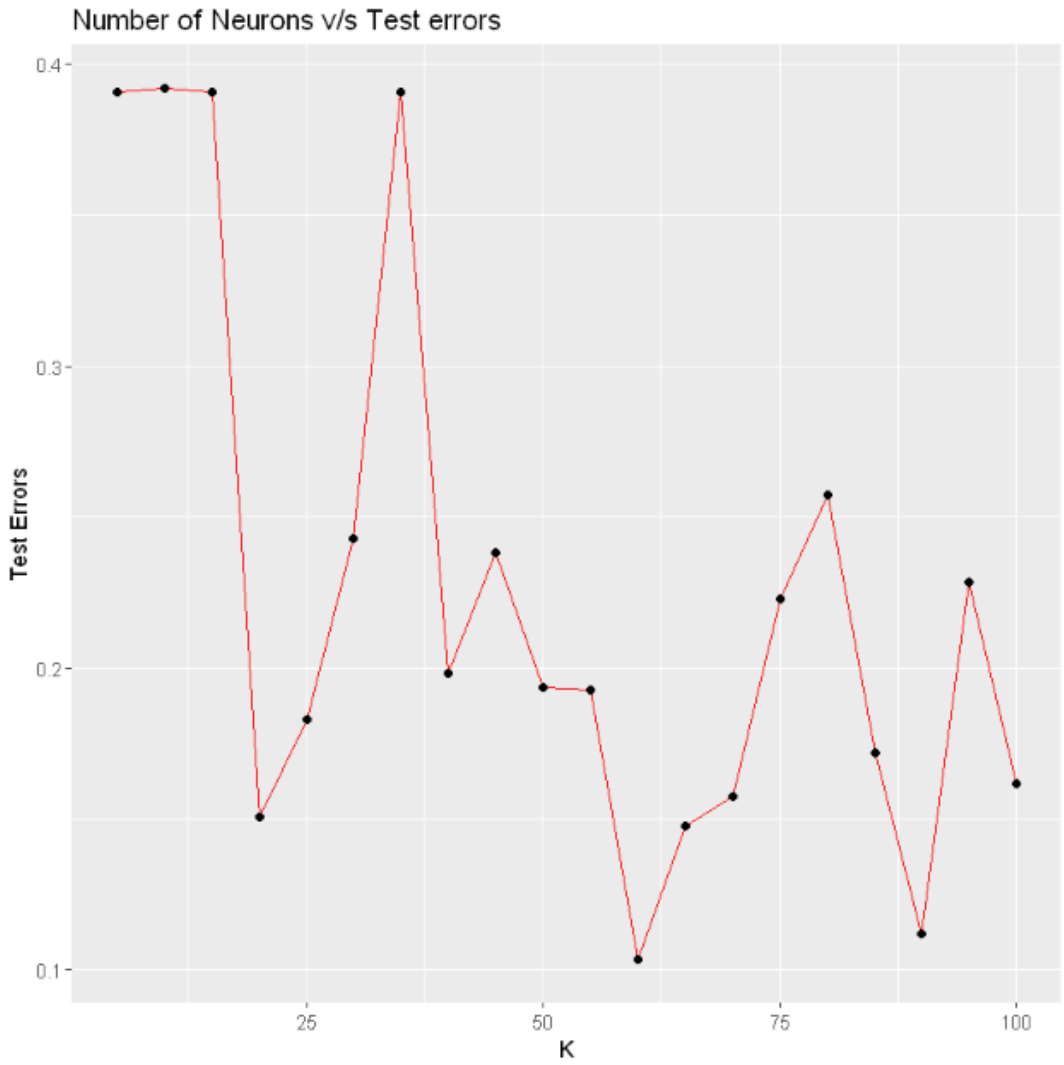


3.Errror Rate for different K values(Number of Neurons) using NN and Perceptron

Error Rate Of Perceptron =51.68%

Below table lists out the test error for varying number of neurons in the hidden layer. We can see that the test error is the least when number of neurons is 60 at 10.36%.

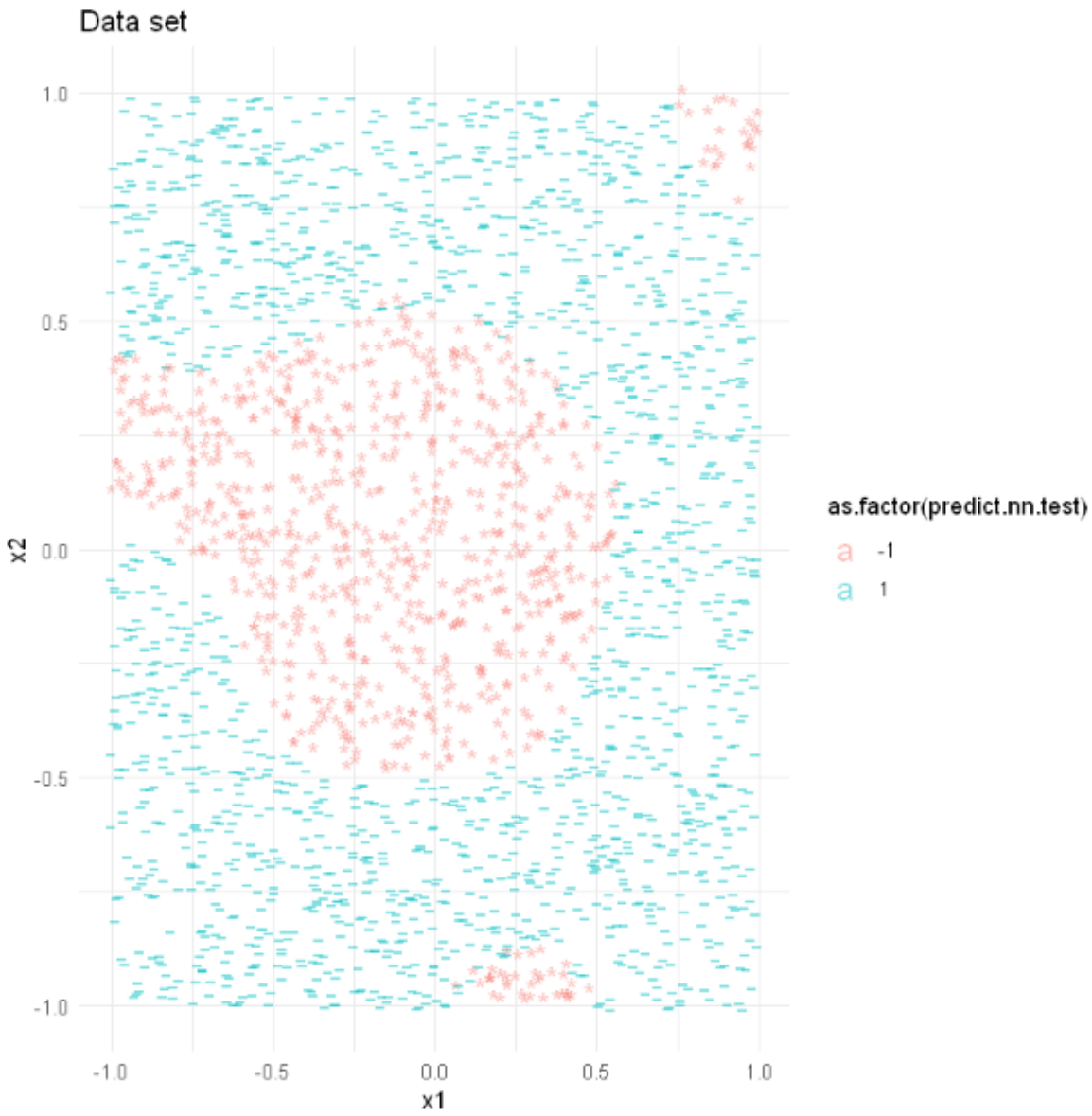
K	Test.Errors	Test Errors
5	0	0.3908
10	0	0.3920
15	0	0.3908
20	0	0.1508
25	0	0.1832
30	0	0.2428
35	0	0.3908
40	0	0.1984
45	0	0.2384
50	0	0.1936
55	0	0.1928
60	0	0.1036
65	0	0.1476
70	0	0.1576
75	0	0.2228
80	0	0.2572
85	0	0.1720
90	0	0.1120
95	0	0.2284
100	0	0.1616



As we as the number of neurons increases the error rate decreases ,the minimum test error is recorded at K=60

4.Classification Of Test Data using Neural Network

Considering K =60 as the optimum value we predict the test labels using Neural Network and on plotting the predicted labels we can see the NN predicts most of the labels right, It captures the non-linearity in the data.



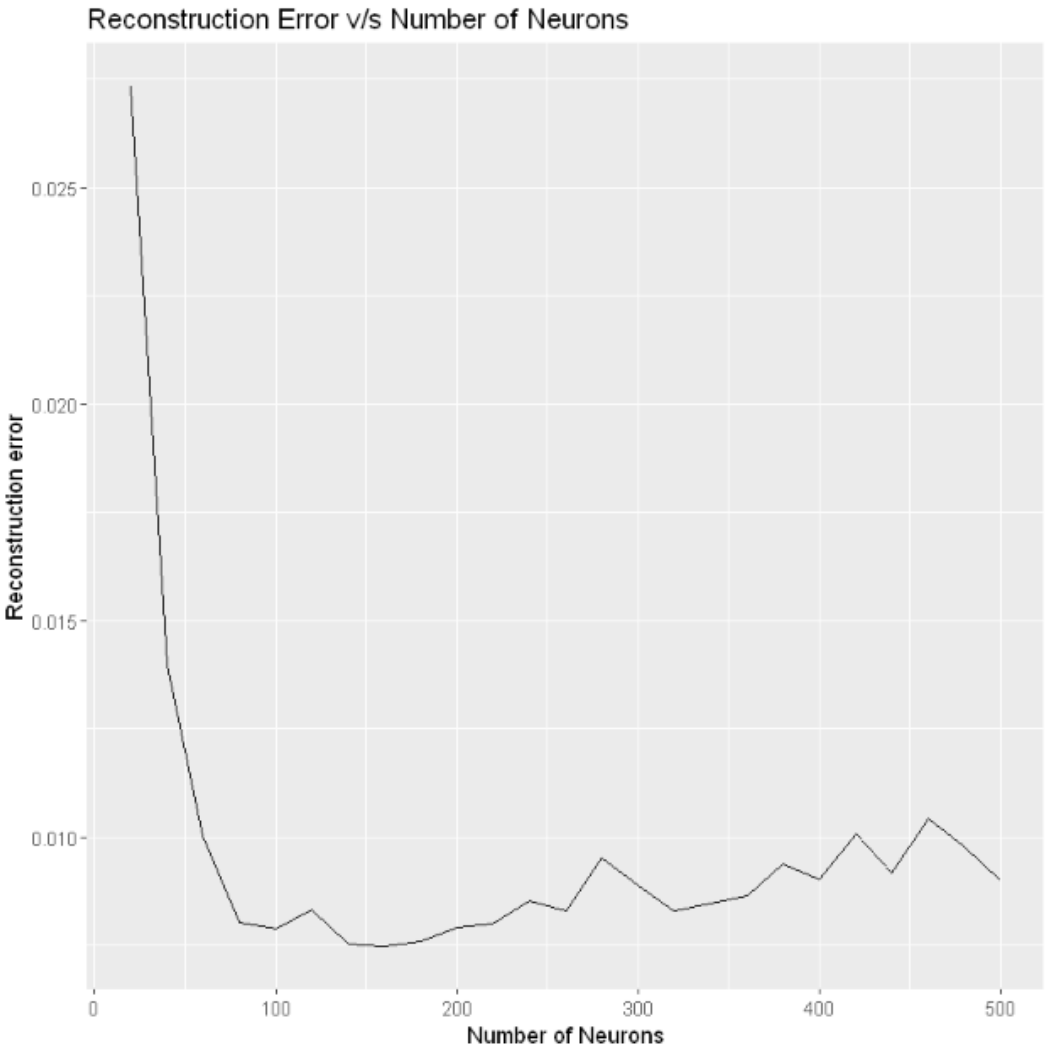
5.Difference between NN and perceptron

We can tell that the neural network performs way better than the perceptron as the data is not linearly inseparable and since perceptron works on linearly separable data, in this case it just divides the data into a linear boundary where everything below it is assigned to a label and the rest is assigned to another label pretty much making it a random guess.

However, the neural network captures the non-linearity of the data. This is because the activation function used in a Tanh function which is non-linear and thus it performs much better when compared to the perceptron.

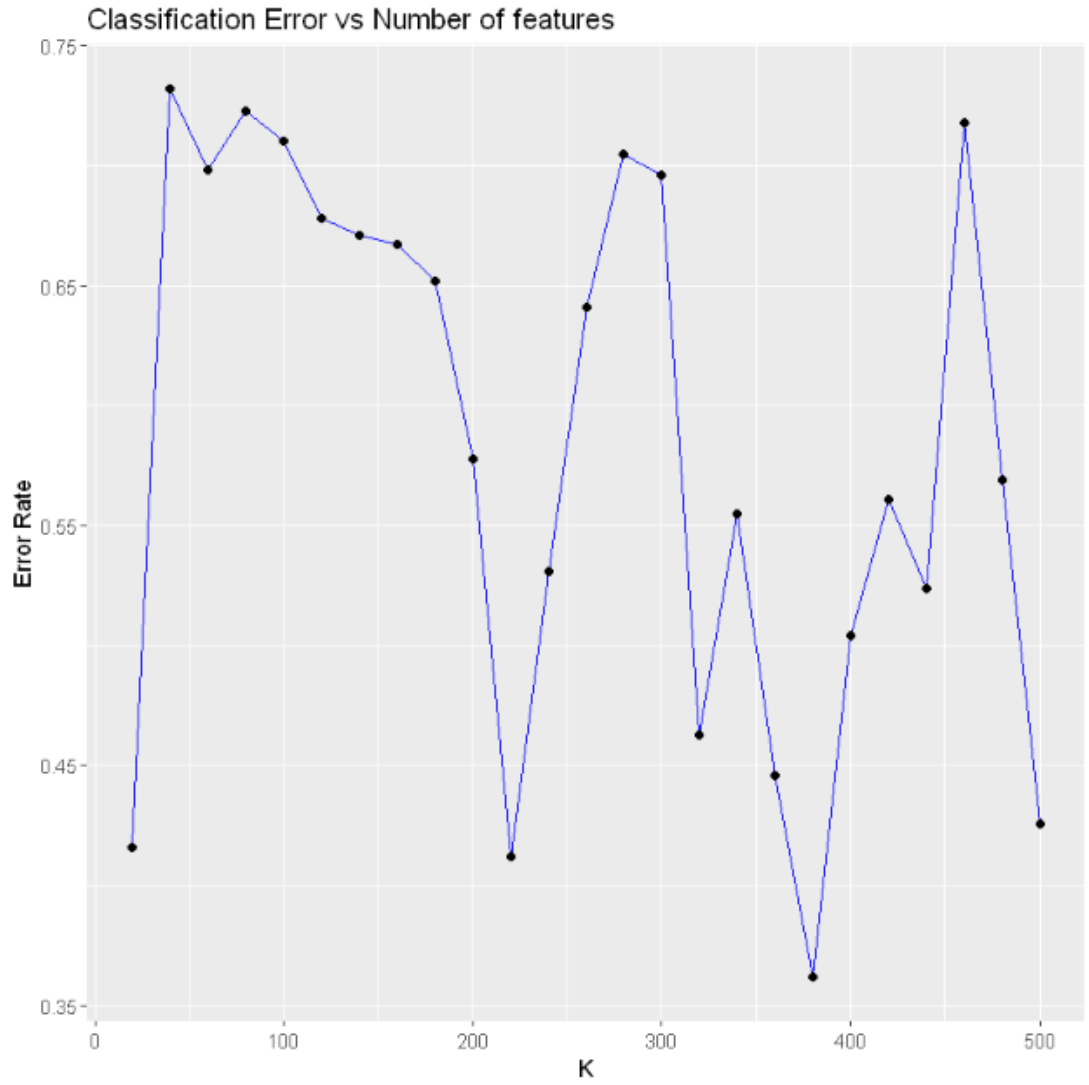
Part C. Self-Taught Learning

1.Plot these values where the x-axis is the number of units in the middle layer and the y-axis is the reconstruction error



From the above graph ,it is seen that the reconstruction error decreases initially with increase in number of neurons. The test error drops when the number of neurons 150 after which it spikes around 275 and then we get some random fluctuations.

2.The error rates for the 3-layer neural networks while the x-axis is the number of extra features and y-axis is the classification error.



From the graph we can make out that the error rate is minimum around 375 followed by just somewhere around 225.

3.Report the optimum number(s) of units in the middle layer of the autoencoder in terms of the reconstruction and misclassification errors.

The optimum number of units in the middle layer can be somewhere around 225 as that is where we can get a minimum classification and reconstruction error. This works basically like a trade-off point or a sweet spot between the two errors as at other points like near 400 neurons, classification error is very low but the reconstruction error is high and around 100 neurons, it is the converse. Thus we can say that 225 is optimum number of neurons in the hidden layer.