Name: Samarth Manjunath

UTA ID: 1001522809

Subject: Data mining

Note: I have used up my two late days for submission.

Problem-1

Part-a

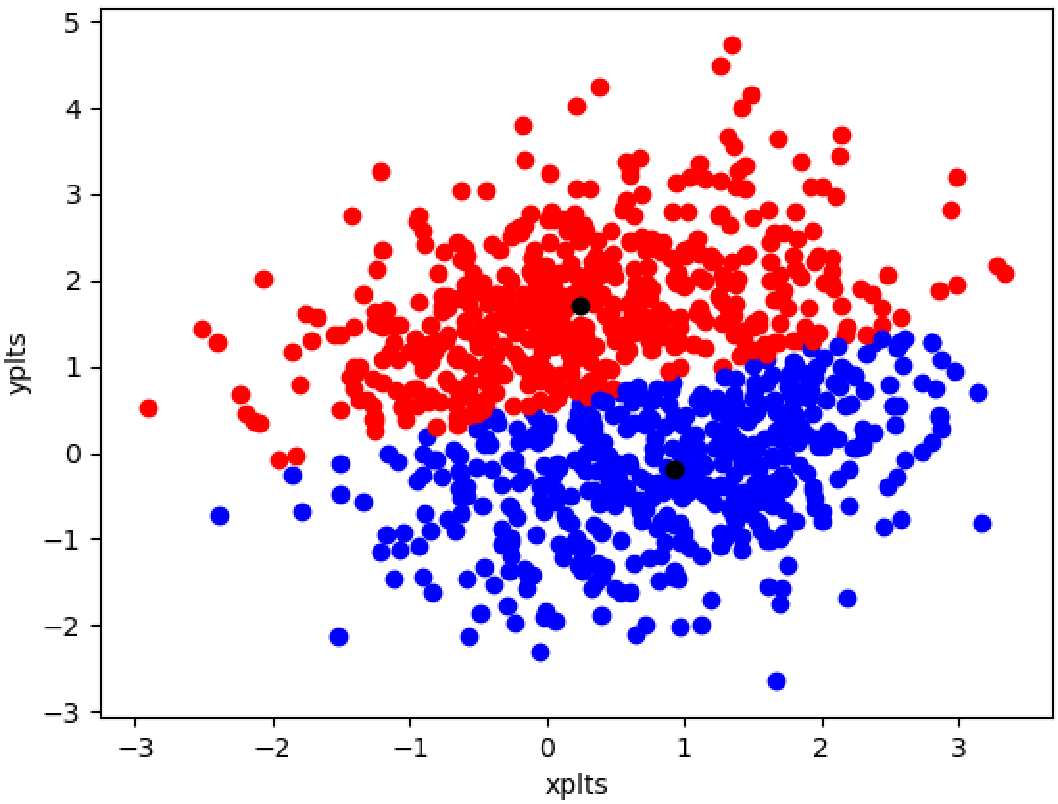
A generic K means implementation

1. The problem was to solve the clustering issue where the data points had to be assigned to its nearest neighbors and this has been done successfully through the submitted code
2. The code first takes the user input for number of clusters and the centroids using which the output is calculated.
3. First the Eucledian distances are calculated between every point and the centroids provided and thus the least distance is recorded.
4. The least distance's source nodes are also recorded and thus the centroids are updated based on this data.
5. This is repeated for many iterations until the stopping condition which has been provided in the requirements.
6. Scatter plot of the proceedings are recorded and shown at the end.

Part-b

Testing the K means for k = 2 and initial centers c1 = (10, 10) and c2 = (-10,-10)

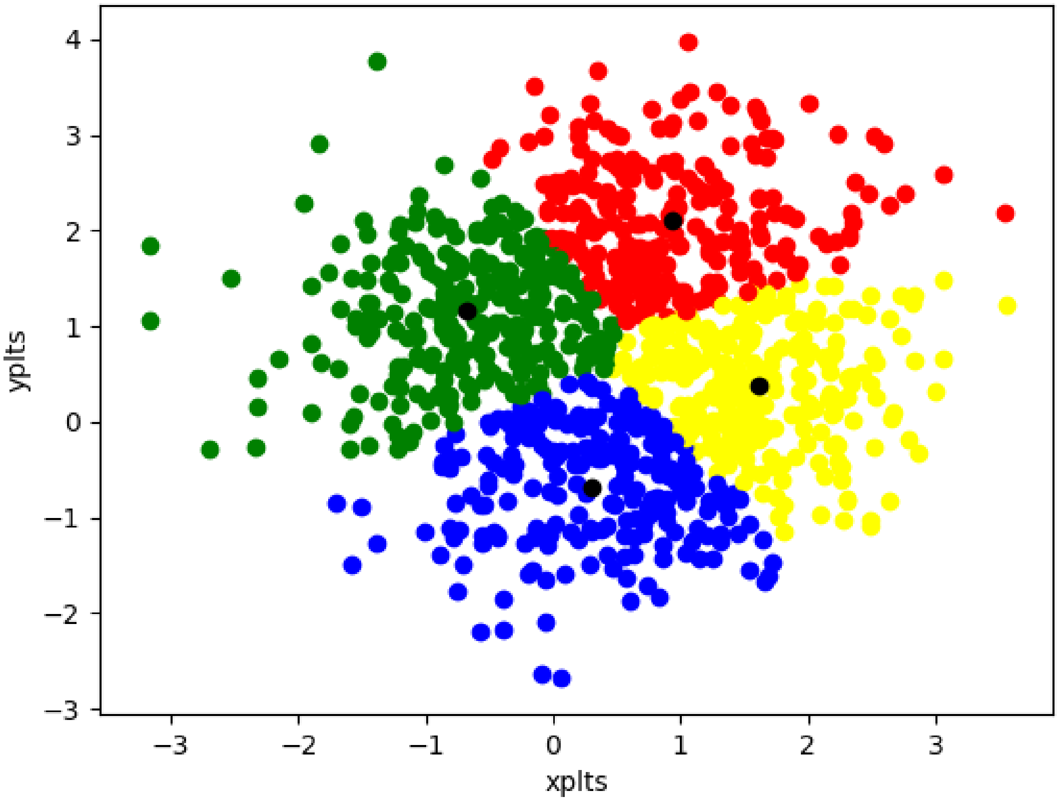
1. Run the code by going to the file's path name in command prompt by py Problem_1.py
2. Please enter the number of clusters into which the clustering needs to be done: 2
3. Please enter the cluster1 : 10 10
4. Please enter the cluster2 : -10 -10
5. Center values post kmeans implementation [[0.23935925 1.70153147]
 [0.92033563 -0.19862205]]
6. number of iterations: 20



Above is the`Scatter plot of k=2

Part-c

7. Testing the K means for k = 4 and initial centers c1 = (10, 10) , c2 = (-10,-10), c3=(10,-10) and c4=(-10,10)
8. Center values post kmeans implementation [[0.9283048   2.11431003] [0.30277435 -0.67989235] [1.61597555 0.38354475] [-0.6747311   1.15566032]]
9. number of iterations: 27



Above is the Scatter plot of k=4

Problem-2

Non-parameteric density estimation

Method to execute the code:

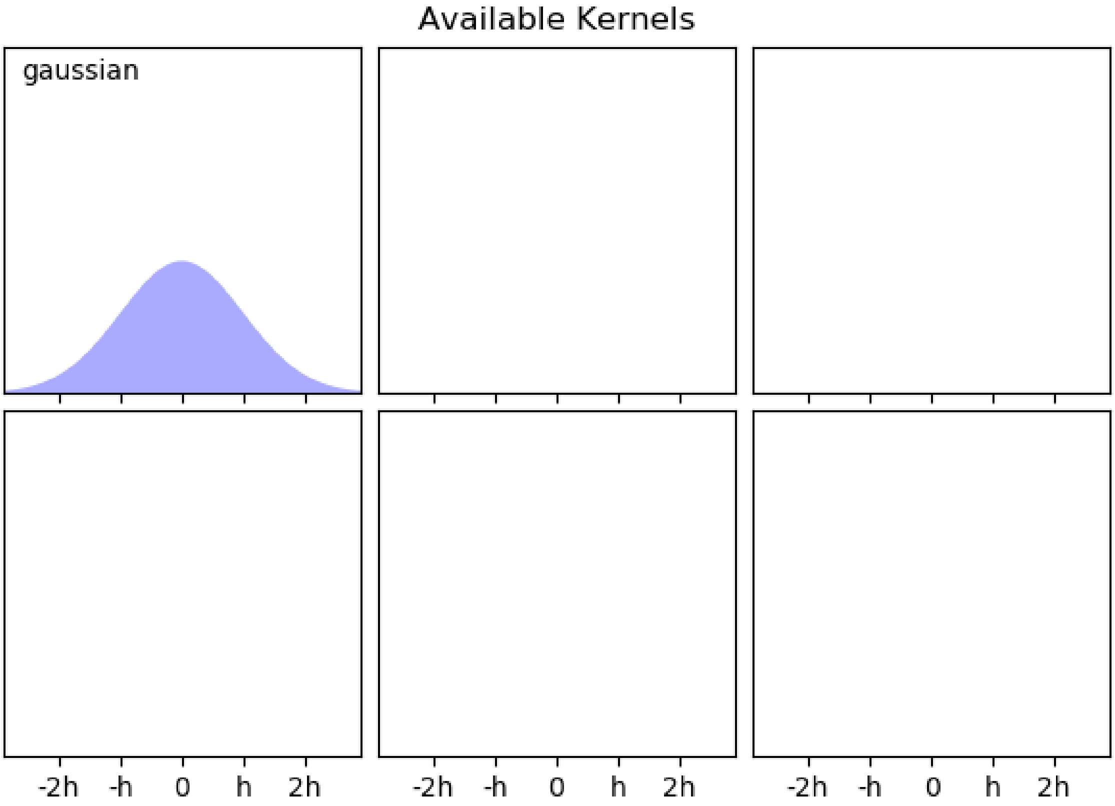
Go to the code’s path

Type py Problem_2.py

Graphs will be generated which are shown below.

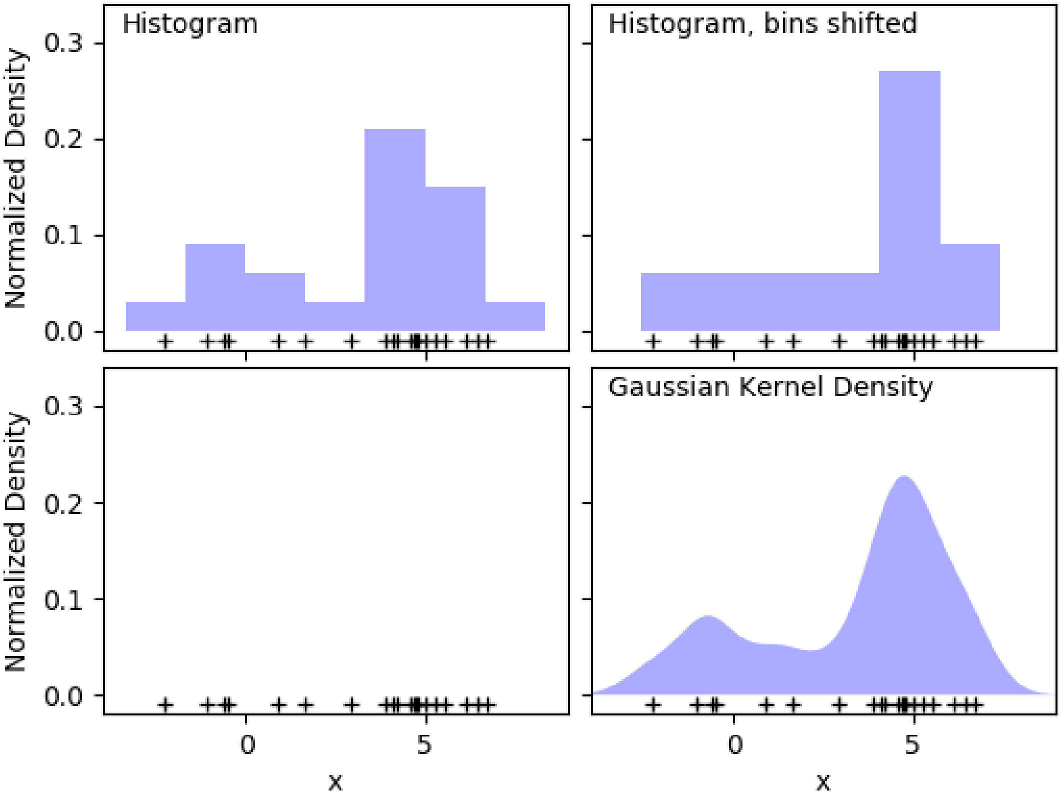
Part-a

1. Function performs kernel density for the provided bandwidth and the generated gaussian random variables for 1D, 2D.



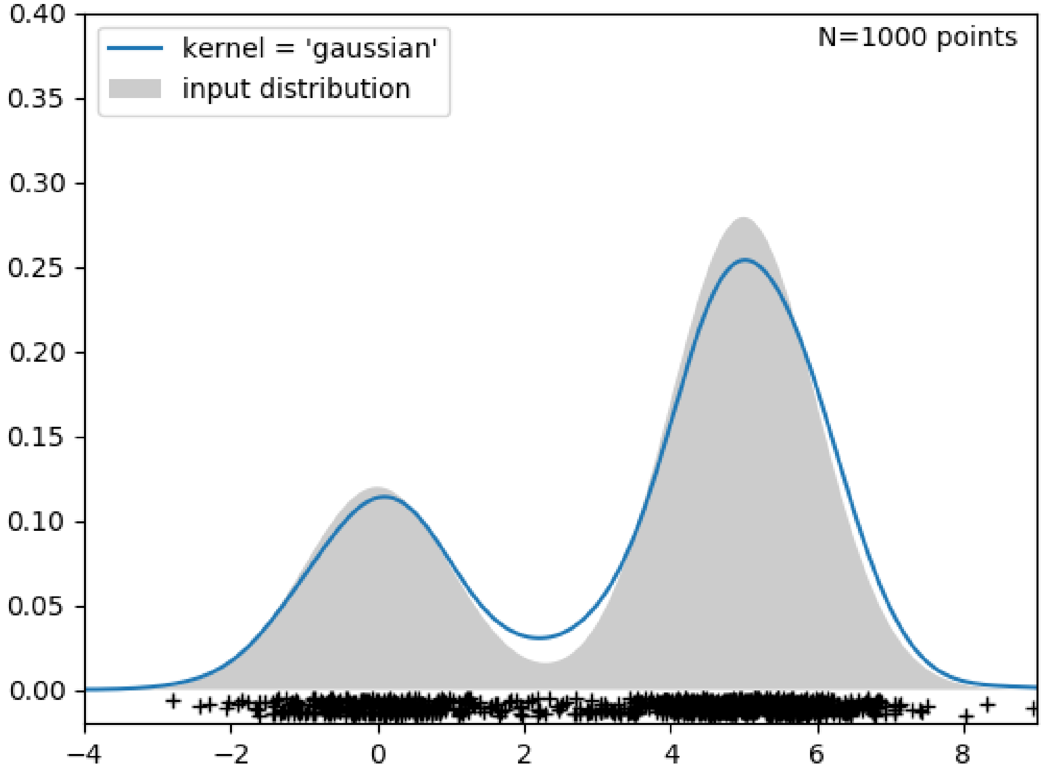
Part-b

2. To generate N=1000 gaussian random variables, change the code for N=1000 and mu=5 and sigma=1 and h={.1,1,5,10}



Part-c

1. To generate N=1000 gaussian random variables, change the code for N=1000 and mu=5 and sigma=1 and another random variable mu=0, sigma=0.2 and h={.1,1,5,10}



Part-d

1. To generate 2 sets of 2D Gaussian random data with N1=500 and N2=500 using the given parameters where h={.1,1,5,10}, change the N value in the code.