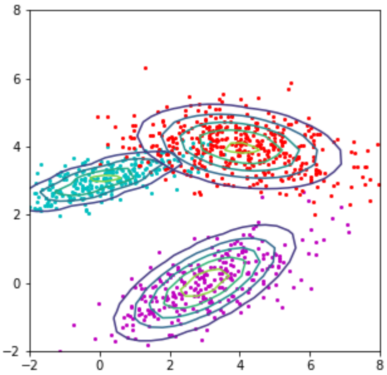
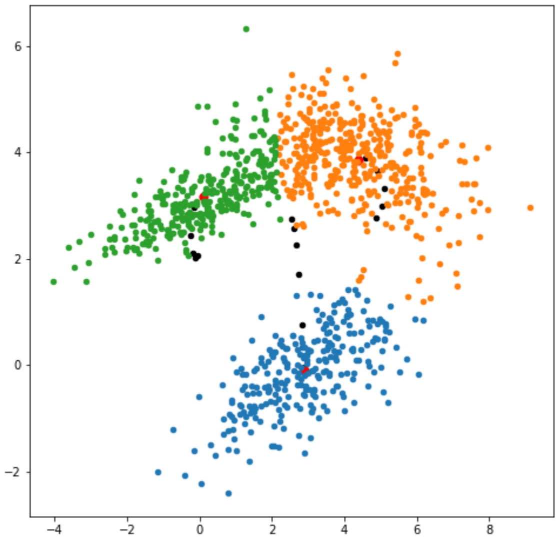
Foundations of Machine Learning Assignment

Abhilash Pulickal Scaria
aps1n21@soton.ac.uk
Student Id:33124639

1) K-Means Clustering



(a) Data from mixed Gaussian density



(b) K-Means Clustering

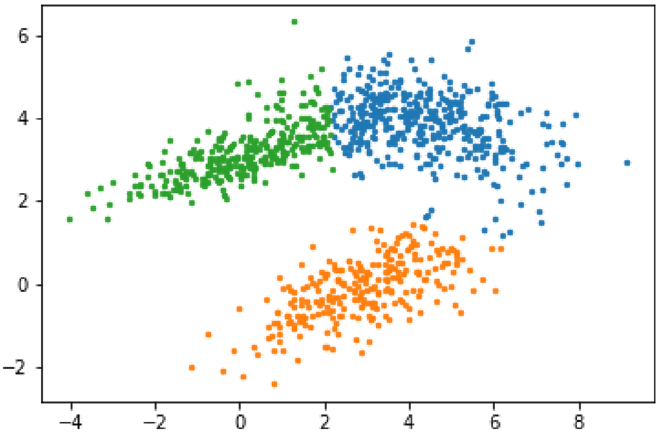
Figure 1

1) After sampling data from a mixture of Gaussian density using the code snippet provided we implement the K-Means Clustering algorithm. The observations obtained are given above.

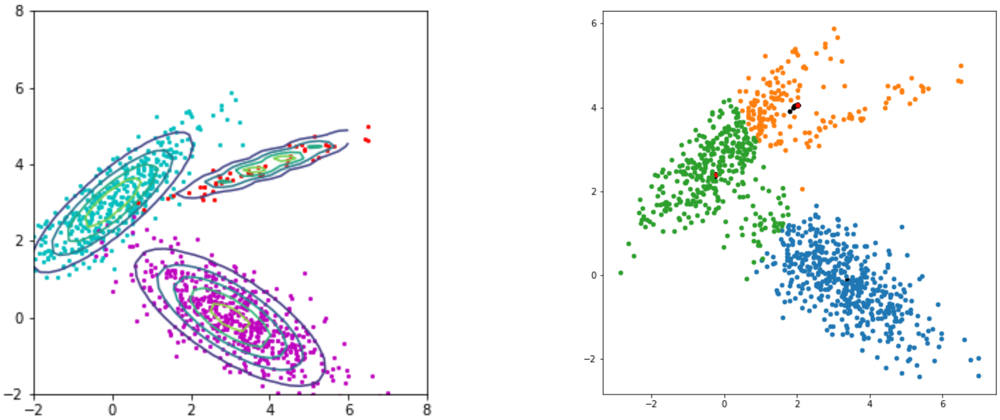
2) Figure 1(a) shows contours on the probability density we have used . Comparing this with figure 1(b), which is the output produced by the K-Means clustering we can see that K-Means has done a decent job at clustering the given data. We can observe the 3 centroids(red color) and the black points which shows the convergence of cluster centres.

3) The result obtained from the K-Means implementation is similar to the result obtained when using sklearn K-Means algorithm. In both methods the centroids obtained are same.

Figure 2: K-Means sklearn



4) K-Means algorithm doesnot take into account cluster size or density, it only considers the euclidean distance between cluster centroid and datapoints. So when we come across non spherical clusters or clusters with varying sizes K-Means fails. K-Means algorithm is guaranteed to converge but not always to global optimum. As its cost function is not convex, initialization of weights directly affects its results. The figure 3 shows failure of K-Means. One of the cluster is smaller(lesser number of datapoints) compared to others, but since K-Means only considers Euclidean distance between centroid and datapoints it clusters it wrongly.



(a) Data from mixed Gaussian density

(b) K-Means Clustering

Figure 3

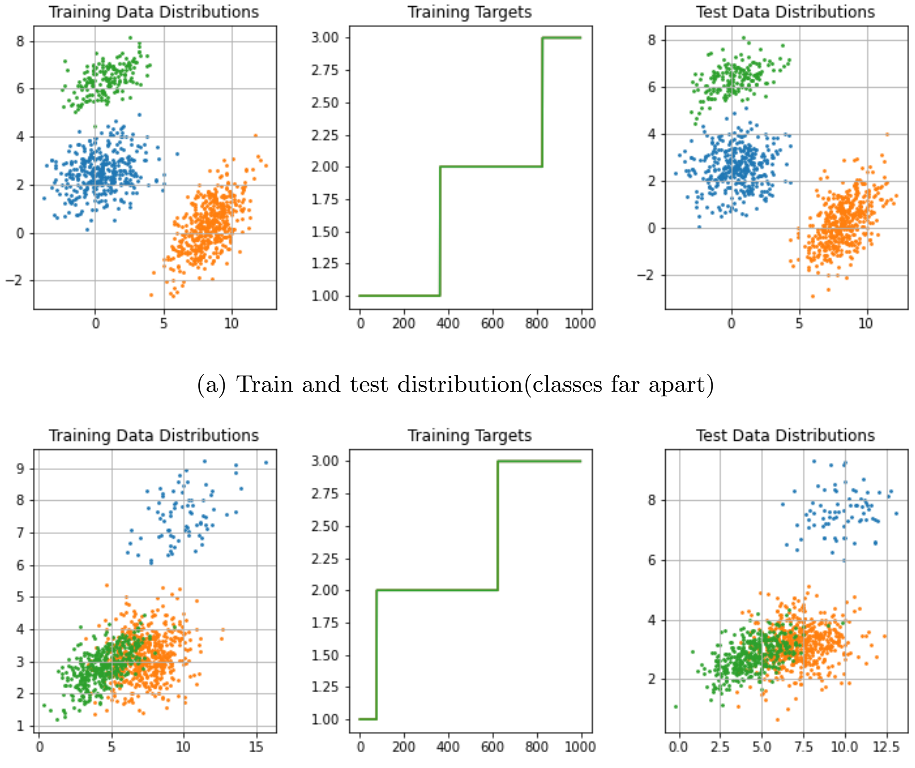
5) The seeds dataset from UCI repository is taken and clustered using K-Means. Seeds dataset is a dataset containing measurement of geometrical properties of kernels belonging to three different varieties of wheat. Table 1 shows the result of K-Means clustering. Labels(1,2,3) denote the actual target value and Cluster(0,1,2) denotes K-Means clustering output. From table we can see that the algorithm misclassified 17 datapoints out of 210. So the K-Means algorithm has 91.9% clustering accuracy on this dataset.

Table 1: K-Means Clustering on seeds dataset

Label / Cluster	0	1	2
1	62	6	2
2	5	0	65
3	4	66	0

2) Multi-Layer Perceptron

1) Using the code snippet provided we create two classification problems, one in which classes are easy to seperate and another in which classes overlap. The problems created are given in figure 4.

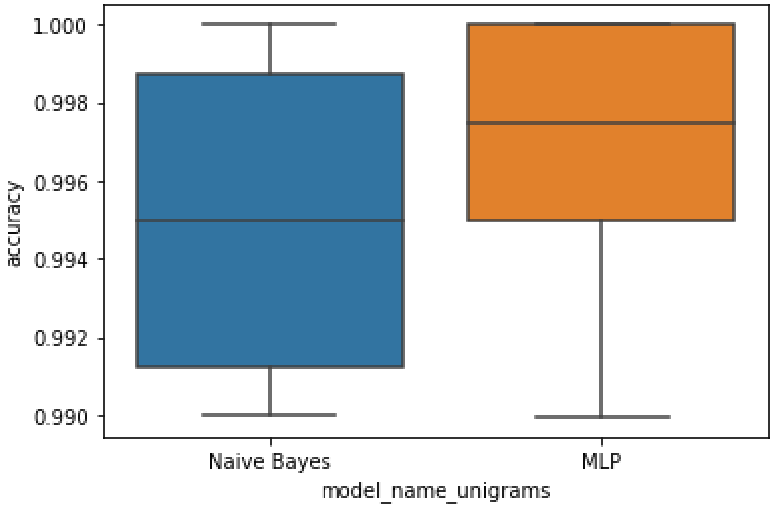


(a) Train and test distribution(classes far apart)

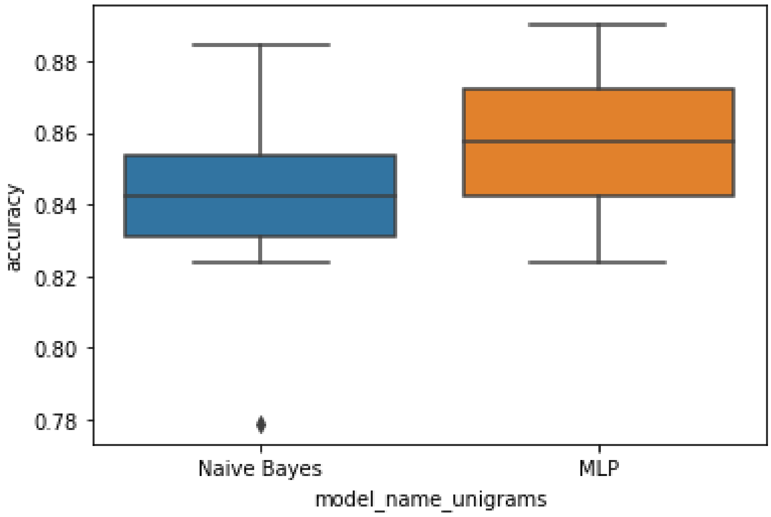
(b) Train and test distribution(classes overlap)

Figure 4

2) We use MLP and Naive bayes classifier on the two datasets. After performing ten-fold cross validation and visualizing the results on two boxplots(figure 5) we can observe that both models perform really well on the easier classification problem(figure 4(a)), but has lower accuracy for the classification problem in which classes overlap(figure 4(b)). MLP has better median accuracy in both these problems, 99.7% for figure 4(a) dataset and 86% for figure 4(b) dataset compared to Naive bayes which has 99.5% accuracy for figure 4(a) dataset and 84% for figure 4(b) dataset.



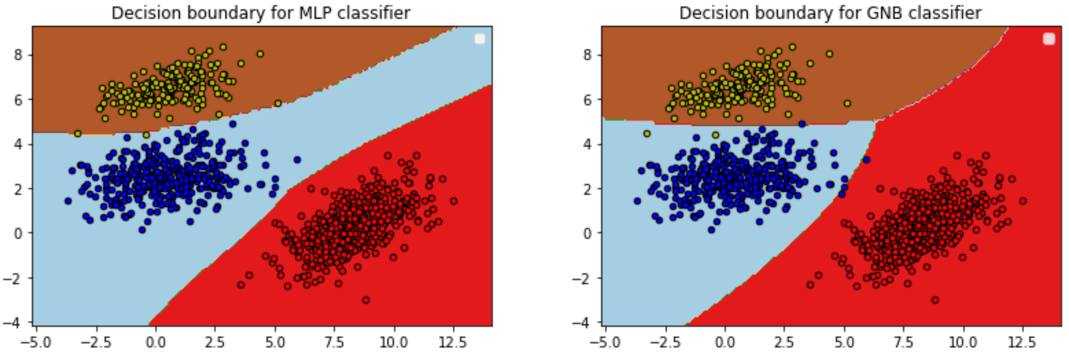
(a) Boxplot obtained by ten-fold crossvalidation for classification of data(fig 4(a))



(b) Boxplot obtained by ten-fold crossvalidation for classification of data(fig 4(b))

Figure 5

3) MLPs are global approximators and can be trained to implement any given nonlinear input-output mapping[Anke Meyer-Baese, Volker Schmid, in Pattern Recognition and Signal Analysis in Medical Imaging (Second Edition), 2014]. Naive Bayes on the other hand is a fast classification technique but has two fundamental assumptions the first is complete independence of features and second is that attribute should follow a normal distribution, which is not always true. The class boundaries obtained for the classification problem(figure 4(a) classes far apart) from MLP and Naive bayes are given in figure 6.



(a) Class boundary for the classification problem(figure 4(a) classes far apart) from MLP classifier

(b) Class boundary for the classification problem(figure 4(a) classes far apart) from Naive Bayes classifier

Figure 6

For MLP classifier implemented to classify figure 4(b) classification problem we used the default number of hidden layers which is 100 and we obtain an accuracy of 86% . When we decrease the number of hidden layers to 5 we can see that MLP now has lower accuracy 81%. So we can observe that the simpler model with less hidden layers doesnot perform as well as the one with more hidden layers. In this classification problem increasing the hidden layers improved the performance, but increasing it above 100 doesnot improve the accuracy, it remains at 86%. Making the model complex is not guaranteed to increase its accuracy, in some cases it can result in overfitting also.

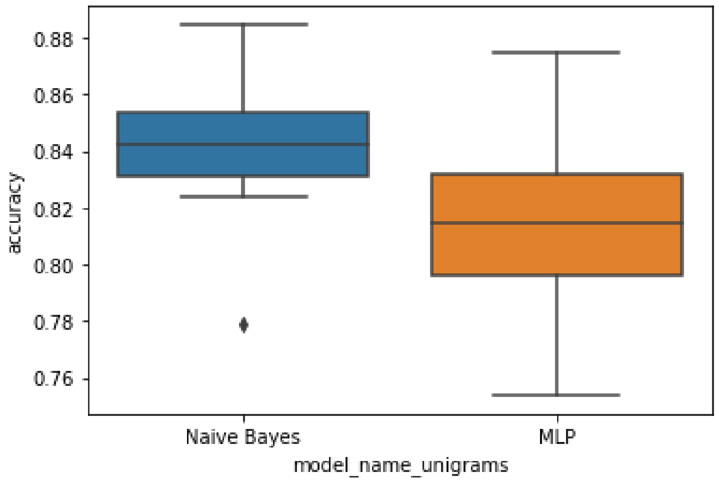
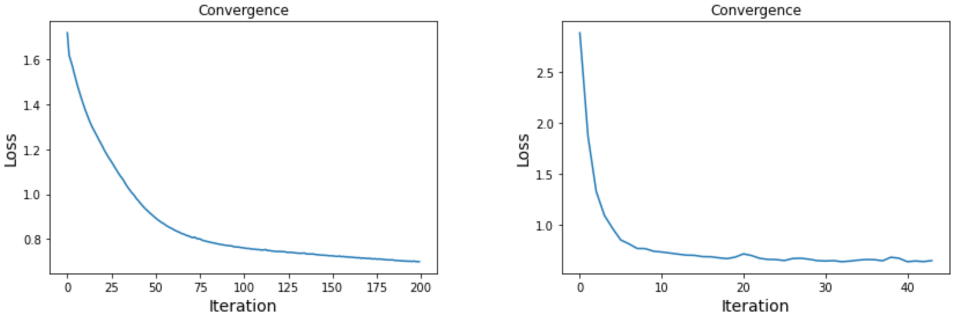


Figure 7: Boxplot obtained by ten-fold crossvalidation for classification of fig 4(b)(number of hidden layer = 5)

4) The initial learning rate is set to 0.001 by default. When we run it with this rate for the classification problem of figure 4(b) we can observe that even after 200 iterations(default number of iterations) the optimization algorithm has not reached convergence. After changing the learning rate to 0.1 we can see the optimization algorithm reaches convergence after 45 iterations.



(a) Convergence when learning rate set to 0.001(default)

(b) Convergence when learning rate set to 0.1

Figure 8

After setting random state to ensure reproducible results across multilple runs, we can observe that changing the regularization parameter(alpha) from the default value(0.0001) to 0.1 reduces the accuracy on test set from 84% to 80%. Changing alpha to 0.00001 also reduces the accuracy to 78.4%. So for this classification problem 0.0001 is a good value for alpha.