COMP 551 (Section 001) Assignment 2

"LINEAR CLASSIFICATION AND NEAREST NEIGHBOUR CLASSIFICATION"

Name: Rashik Habib

Date: 11th February 2018

Question 1: *Submitted the required dataset as "DS1_testset.txt" and "DS1_trainset.txt" within the directory "hwk2_datasets_corrected/". Please refer to the submitted code "A2Q1.py" for details on how the datasets were generated*.

Question 2: Using the dataset generated in Q1, and the maximum likelihood approach as seen in class, the probabilistic LDA model produced the following result:

	Accuracy	**Precision**	**Recall**	**F-measure**
LDA	0.953333	0.959459	0.94667	0.9530201

Table 1: Performance measures for the probabilistic LDA model (Single Gaussian)

These were considered to be very high-performance measures, reaching approximately 95%. For completeness, the coefficients learnt by the model are reported as follows:

$\mathbf{w}_{positive}$	$\mathbf{w}_{negative}$
-39.7988	-25.9462
23.6092	15.3675
15.7763	10.2598
9.20255	5.96875
27.0721	17.5862
11.8335	7.72348
-47.1107	-30.6447
66.3078	43.1878
80.7032	52.6095
-25.3232	-16.4748
35.8628	23.3195
33.5366	21.8462
-42.983	-28.0381
-35.9603	-23.4622
15.6989	10.2142
-35.8771	-23.3788
-81.6416	-53.1334
18.7579	12.26
2.4397	1.60822
13.4	8.78388

$\mathbf{w}_{0,\ positive}$	-46.1708
$\mathbf{w}_{0,\ negative}$	-20.0112

Table 2: Coefficients learnt by the probabilistic LDA model (Single Gaussian)

These coefficients correspond to the linear discriminants of each class, denoted as:

$$y_k(\boldsymbol{x}) = \boldsymbol{w}_k \boldsymbol{x}^T + w_{0,k}$$

The class discriminant leading to the higher value for the example was considered to decide which class the example would be classified as.

Question 3: The implemented model of KNN uses a Frobenius norm to determine the nearest neighbours. By varying the value of K from 5 to 50 in steps of 5, the following table for the performance measures was generated:

K	Accuracy	Precision	Recall	F-measure
5	0.543333	0.543478	0.54167	0.542571
10	0.560833	0.551049	0.65667	0.5992395
15	0.55	0.552817	0.52333	0.5376712
20	0.564167	0.56025	0.59667	0.5778854
25	0.5625	0.567325	0.52667	0.5462403
30	0.56	0.558065	0.57667	0.5672131
35	0.558333	0.563177	0.52	0.5407279
40	0.558333	0.558528	0.55667	0.557596
45	0.565833	0.572744	0.51833	0.544182
50	0.571667	0.574394	0.55333	0.5636672

Table 3: Performance measures for varying K in the K-NN model (Single Gaussian)

The table above was used to plot a graph for the performance measures as K varied. Although the recall and F-measure oscillated a bit, the precision and accuracy improved slightly with increasing the number of near neighbours. Despite that, it is quite clear that the non-linear KNN model remains far worse than the linear LDA model not only in terms of performance (by approximately 35-40% for each measure), but speed of computation as well. The best performance was achieved with 10 nearest neighbours (highlighted), considering the F-measure (59.9%) to be the critical performance indicator. The graph below shows the results of the table visually:

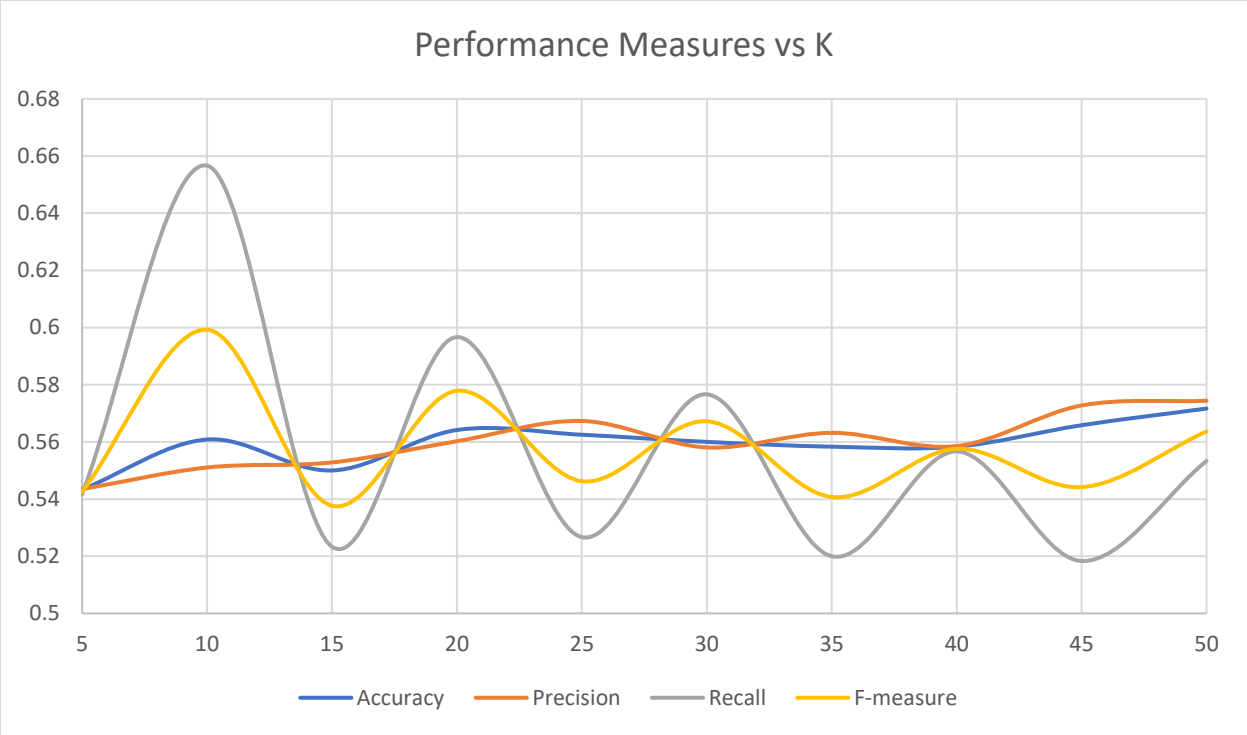

Figure 1: Performance measures for varying K in the K-NN model (Single Gaussian)

Question 4: *Submitted the required dataset as "DS2_testset.txt" and "DS2_trainset.txt" within the directory "hwk2_datasets_corrected/". The probability for choosing a particular distribution was used to divide the number of samples for each distribution. (For example, if the second Gaussian has 0.42 probability of generating a sample, then it will generate 0.42*2000 = 840 samples). Please refer to the submitted code "A2Q4.py" for details on how the datasets were generated*.

Question 5: The linear LDA model for the data generated by multiple Gaussians is as follows:

	Accuracy	**Precision**	**Recall**	**F-measure**
LDA	0.516667	0.515974	0.538333	0.526917

Table 4: Performance measures for the probabilistic LDA model (Multi Gaussian)

The performance can be seen to significantly drop when compare to the data generated by the single Gaussian. The KNN performance measures for the same dataset are as follows:

K	**Accuracy**	**Precision**	**Recall**	**F-measure**
5	0.516667	0.516722	0.515	0.51586
10	0.520417	0.518216	0.580833	0.547741
15	0.528333	0.528428	0.526667	0.527546

20	0.529167	0.526515	0.579167	0.551587
25	0.530417	0.529992	0.5375	0.533719
30	0.5225	0.52093	0.56	0.539759
35	0.52	0.519737	0.526667	0.523179
40	0.522083	0.520591	0.558333	0.538802
45	0.522917	0.522559	0.530833	0.526664
50	0.525833	0.524031	0.563333	0.542972

Table 5: Performance measures for the KNN model (Multi Gaussian)

A graph was plotted similar to Q3, but including the LDA F-measure performance, and analysed:

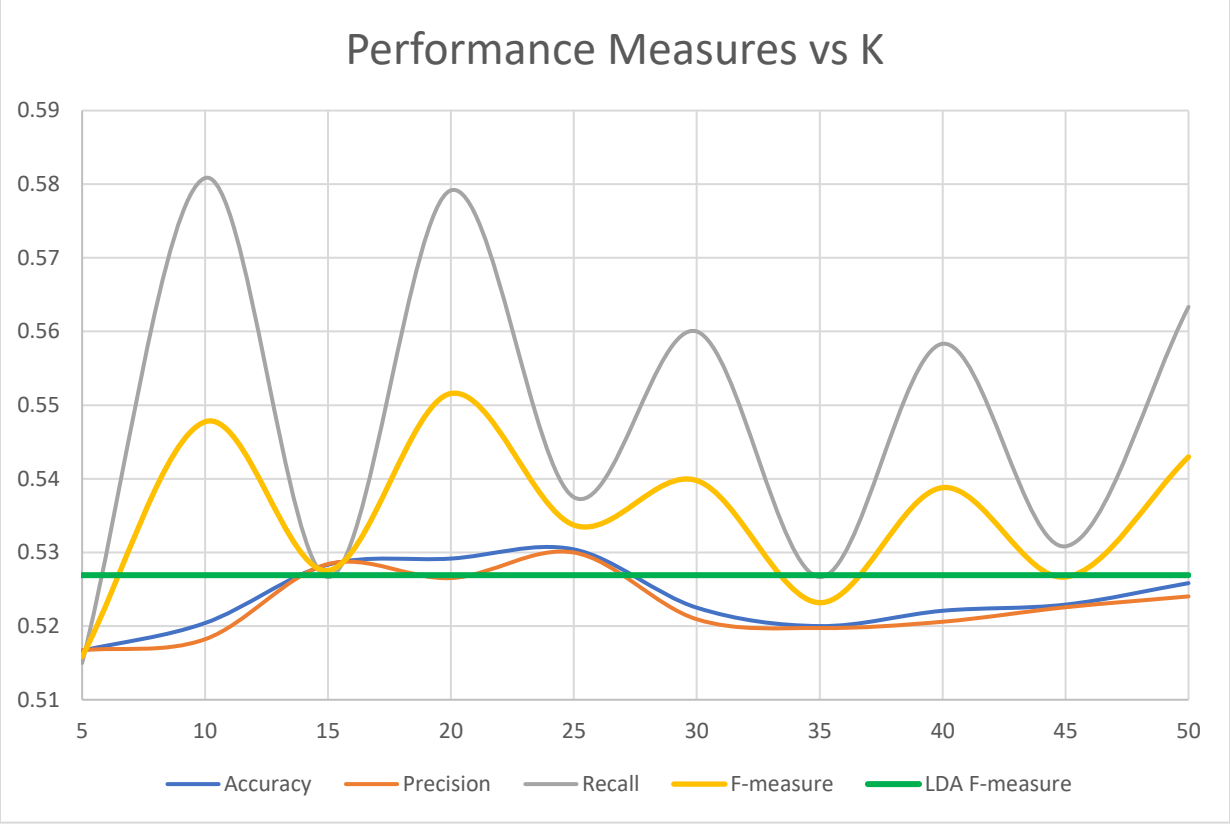

Figure 2: Performance measures for varying K in the K-NN model (Multi Gaussian)

The best (F-measure) performance for the KNN model is for 20 nearest neighbours.

It's very interesting to see that the performance for the non-linear KNN model does not decline significantly when we use the second dataset, whereas the linear LDA model is far worse than before. In fact, we see that the KNN model actually performs better than the LDA model for most values of K.

This is coherent with what we have seen in class. In the first case (single Gaussian), both the data tend to have similar distribution "shapes" and become linearly separable, leading to a significantly better performance by LDA. However, in the second case, when the data is

approximated to be from a single Gaussian when it actually is not, the decision boundary is poorly estimated by LDA. The general distribution "shape" does not necessarily make the data linearly separable and hence a non-linear decision boundary is required. Therefore, it is no surprise that the non-linear KNN model performs significantly better than the LDA.

Additionally, the LDA was still seen to be significantly faster in terms of computational speed. The KNN model's optimum number of neighbours would also heavily depend on the number of features of the data, as seen in class.

Question 6: To summarize in table format:

	DS1 (Single Gaussian)	**DS2 (Multi Gaussian)**
LDA	- Since the distribution was well estimated by the linear model, the linear decision boundary gave a significantly high (95%) performance measure. - A great classifier to use when good knowledge of the distribution is available (given that they have same covariance matrix)	- Since the distribution was <u>not</u> well estimated by the linear model, the linear decision boundary gave a significantly lower (53%) performance measure. A non-linear decision boundary was required to achieve a good decision boundary. - A bad classifier to use when good knowledge of the distribution is unavailable
KNN	- Considering the mean values were quite similar, at areas where the two classes 'met', the NN method was unlikely to produce correct classifications, as the NN was no longer a good measure. - Although varying the number of neighbours helps improve performance slightly, in the case where the data is linearly separable, this classifier is far worse (by 35-40%) and should be avoided.	- In a situation where the data is not linearly separable, and a non-linear decision boundary is required, the NN method proves to give better results than linear methods. - A suitable classifier to use when good knowledge of the distribution is unavailable

Table 6: Similarities and differences between LDA and KNN on DS1 and DS2