COMP-551: Applied Machine Learning
Assignment #2
Mandana Samiei
Student ID: 260779555

PART 1

1- I saved train set as DS1_train.csv which is 2800*21 and test set as DS1.csv which is 1200*21.
The parameters of probabilistic LDA are:
Mu_0 and Mu_1 are vectors with dimension: (20,)
Covariance is a 20*20 matrix which I showed just the first 4 elements in figure 1.

2- As we can see, all the fitted parameters are somehow similar to the means and covariance matrix provided for data generation which is reasonable and they shouldn't have that much difference.

```
_____________________Learnt parameters of LDA_____________________

Mu0=
[ 1.28997251  1.23624334  1.220287    1.20549323  1.17859172  1.23446731
  1.28861815  1.30171476  1.22754846  1.15551633  1.21743501  1.29914179
  1.19170375  1.21234877  1.19232966  1.20720612  1.22832235  1.2199207
  1.2769214   1.15398153]
Mu1=
[ 2.11760562  2.10248629  2.03882353  2.07878407  2.13571022  2.05641243
  2.13520943  2.10619605  2.05072619  2.12577598  2.10814375  2.07910419
  2.08870026  2.10512069  2.1047091   2.15959987  2.04223794  2.09581034
  2.13266951  2.09630901]
Covariance=
[[ 7.57290218  5.16405155  5.73457257  4.81028735  5.56830721  5.77080431
   4.46908698  5.11471394  4.78214793  4.82314904  3.80654123  5.05369666
   6.66296165  5.71371123  5.6940697   5.59064788  5.57137141  5.34852186
   5.38669547  5.51391128]
 [ 5.16405155  6.53859891  4.99839611  4.07369319  5.16013664  5.23109095
   4.20358702  3.71945966  3.95024411  4.73970132  3.18978174  4.43666362
   5.54913113  4.87851608  5.11424538  4.87055395  5.34269769  4.78032461
   5.18065253  4.99438542]
 [ 5.73457257  4.99839611  6.91255897  4.53661383  5.54311395  6.16854408
   4.43025908  4.46433044  4.66276927  4.68190964  3.1465899   4.41160399
   5.88845616  4.90794948  5.72878544  5.59484788  5.96283375  4.60238744
   4.51090859  4.67992298]
 [ 4.81028735  4.07369319  4.53661383  5.50500894  5.03326997  4.19844468
   3.56457071  4.11637481  3.16259633  3.92893042  2.5951919   3.97667376
   5.51373458  4.52770093  4.42707917  4.76445123  4.40495306  4.29566218
   3.71968575  5.45468785]

P_C1=0.5
P_C2=0.5
```

Figure 1- LDA Parameters

The best accuracy, precision, recall and F1-measure achieved by the LDA classifier is:

```
_____________________Test Set Evaluation(LDA)_____________________

Accuracy: 95.33333333333%

Precision: 94.88448844888%

Recall: 95.83333333333%

F-measure: 95.3565505804%
```

Figure 2- LDA Best Fit Performance

3- I tried 100 different values of K from 1 to 200 scaled by 2, here are just 10 values of k with their corresponding performance evaluation.

```
__ K = 1  ___ Confusion matrix = [[326, 294], [274, 306]]

Accuracy: 52.6666666667%
Precision: 52.5806451613%
Recall: 54.3333333333%
F1_measure: 53.4426229508%

__ K = 3  ___ Confusion matrix = [[322, 293], [278, 307]]

Accuracy: 52.4166666667%
Precision: 52.3577235772%
Recall: 53.6666666667%
F1_measure: 53.0041152263%

__ K = 5  ___ Confusion matrix = [[334, 295], [266, 305]]

Accuracy: 53.25%
Precision: 53.1001589825%
Recall: 55.6666666667%
F1_measure: 54.3531326282%

__ K = 7  ___ Confusion matrix = [[332, 290], [268, 310]]

Accuracy: 53.5%
Precision: 53.3762057878%
Recall: 55.3333333333%
F1_measure: 54.3371522095%

__ K = 9  ___ Confusion matrix = [[334, 288], [266, 312]]

Accuracy: 53.8333333333%
Precision: 53.6977491961%
Recall: 55.6666666667%
F1_measure: 54.6644844517%
```

```
__ K = 11  ___ Confusion matrix = [[342, 286], [258, 314]]

Accuracy: 54.6666666667%
Precision: 54.4585987261%
Recall: 57.0%
F1_measure: 55.7003257329%

__ K = 13  ___ Confusion matrix = [[336, 290], [264, 310]]

Accuracy: 53.8333333333%
Precision: 53.6741214058%
Recall: 56.0%
F1_measure: 54.8123980424%

__ K = 15  ___ Confusion matrix = [[333, 285], [267, 315]]

Accuracy: 54.0%
Precision: 53.8834951456%
Recall: 55.5%
F1_measure: 54.6798029557%


__ K = 17  ___ Confusion matrix = [[339, 289], [261, 311]]

Accuracy: 54.1666666667%
Precision: 53.9808917197%
Recall: 56.5%
F1_measure: 55.2117263844%

__ K = 19  ___ Confusion matrix = [[337, 283], [263, 317]]

Accuracy: 54.5%
Precision: 54.3548387097%
Recall: 56.1666666667%
F1_measure: 55.2459016393%
```

Figure 3- KNN Performance Evaluation over different values of K

Afterwards, I visualized accuracy, precision, recall and F1 measure over different values of K.

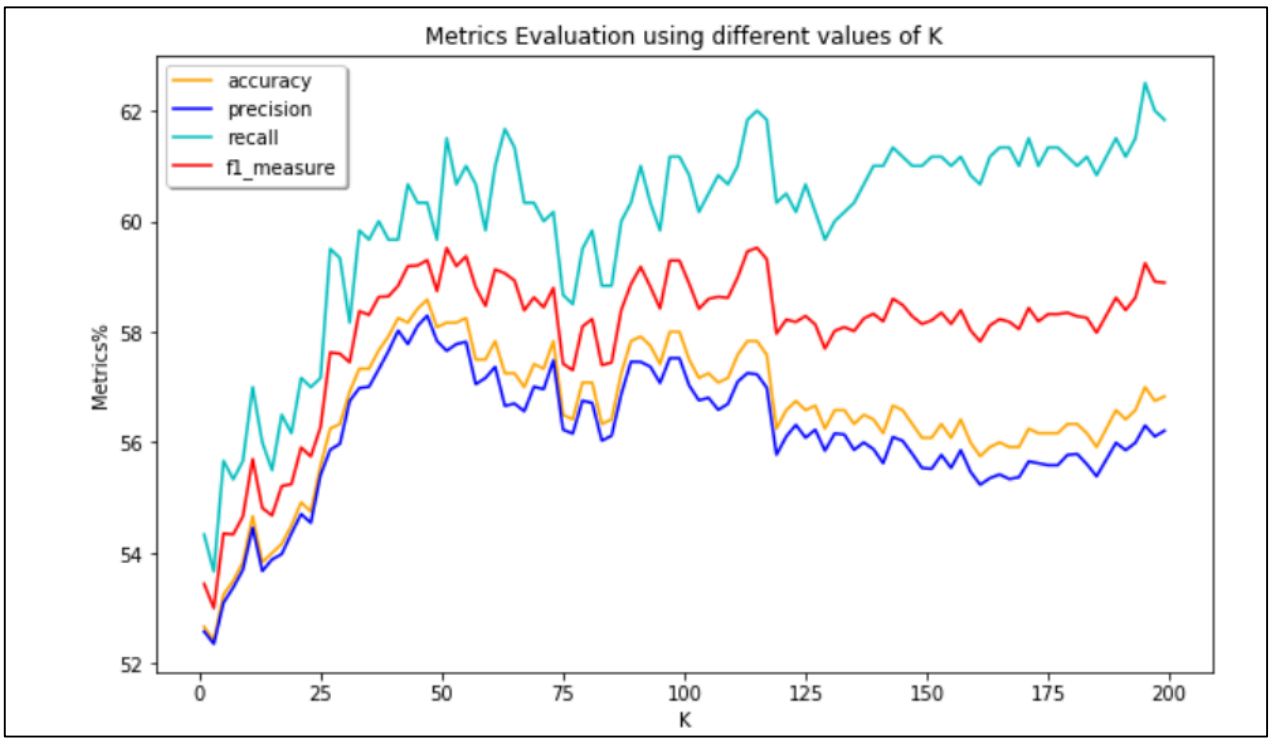


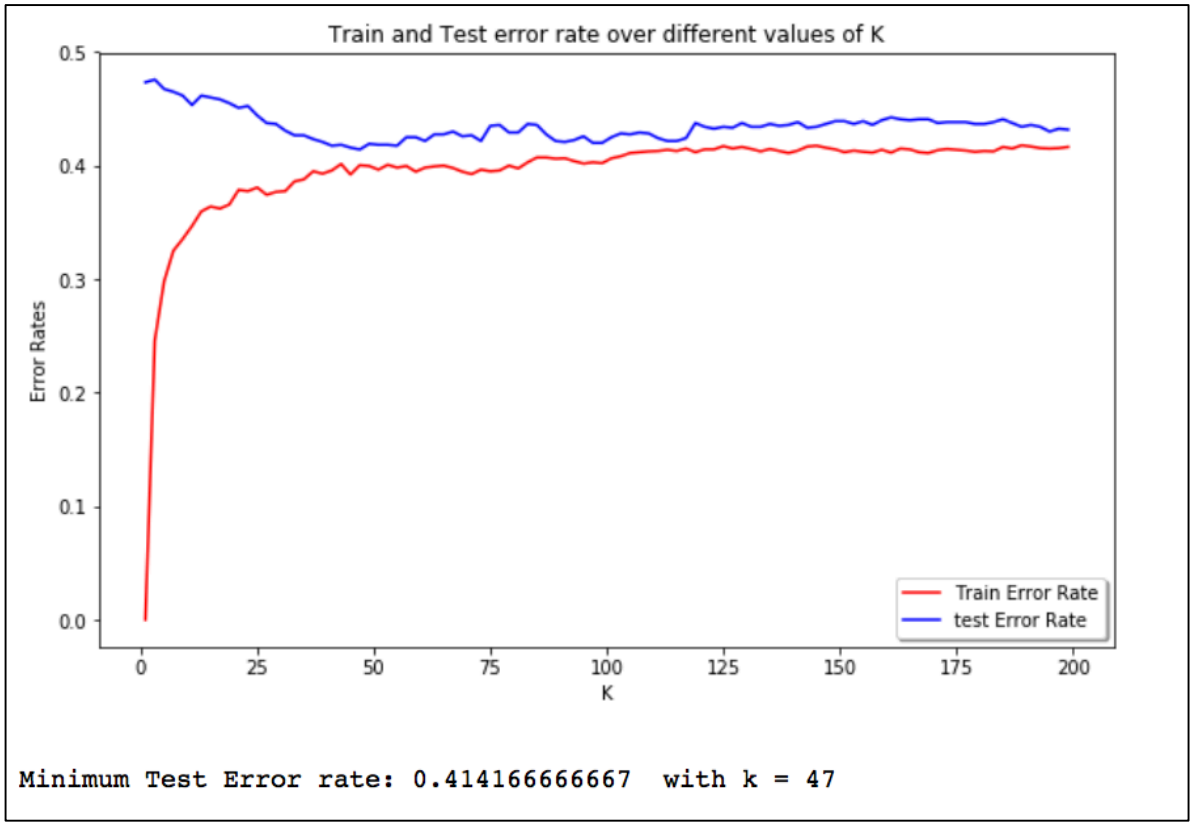
Figure 4- Evaluation Metrics over different values of K

```
Maximum Accuracy: 58.5833333333  with k = 47

Maximum Precision: 58.2930756844  with k = 47

Maximum Recall: 62.5  with k = 195

Maximum F1 Measure: 59.52  with k = 115
```

Figure 5- Maximum Accuracy, precision, recall and F1 measure

According to the above figure, K = 115 has the maximum F1-measure, so k = 115 is the best fit for classifying this dataset by using KNN. If we care about accuracy the best k is 47.
However, the best fit depends on the application and which criteria we care more about it. Also, this best K is different in each run and it is because of the data regeneration.

By the way, the average accuracy for DS1 by using KNN is almost 55% while LDA accuracy is almost 95%. From this result we can infer that the data is linearly separable and LDA (A linear model) works better than KNN (a non-linear model) which might be in opposite with our expectation about linear models. We can figure out a linear classifier can be more powerful that non-linear ones. Here, data points are almost linearly separable because we had 95% accuracy with LDA.
But, in this case our dataset has a lot of features so we are in a high dimension space, and in this situation KNN needs more data that LDA to be as precise as that. And it could be the reason that KNN doesn't works as well as LDA.
Also, I plotted the error rate for train set and test set according different values of k. Here is the diagram:




```
Minimum Test Error rate: 0.414166666667  with k = 47
```

Figure 6- Test set and train set error rate over different values of K

PART 2 – MIXURE OF GAUSSIANS

4-

In this question each class generated by a mixture of 3 Gaussians with weight $(0.1, 0.42, 0.48)$. I applied a random multinomial distribution to create these weights and according the amount of weight, I used the corresponding Gaussian distribution to generate data.
I saved train set as DS2_train.csv which is 2800*21 and test set as DS2.csv which is 1200*21.
C1: Positive class
C2: Negative class

5-

In this part, we were supposed to perform LDA and KNN classifiers on the new dataset DS2.
The parameters of LDA when dataset is a mixture of 3 Gaussians are as follows:

```
_____________________________ calculating LDA parameters_____________________________
Mu1 =
[ 1.0608464   1.09163096  1.07989652  1.08317114  1.08419221  1.08071179
  1.11562903  1.09991281  1.08939498  1.1219368   1.05892408  1.07471214
  1.09172776  1.10617532  1.09161919  1.0579607   1.11629799  1.0901453
  1.0691713   1.04199414]
Mu2 =
[ 1.20399794  1.17953112  1.26797618  1.22264068  1.18894772  1.21056311
  1.24639515  1.17444711  1.2419934   1.26446057  1.26089585  1.29385946
  1.23550993  1.28233183  1.24270364  1.20016577  1.2761073   1.19625909
  1.22953171  1.26386815]
```

```
Covariance =
[[ 7.77548148  5.32487907  4.54850448  4.9588283   4.35244909  5.61432289
   5.77008386  5.61767291  4.63323634  5.13407806  5.47727733  4.88457359
   4.98892467  5.98076695  5.28269245  5.69747984  5.29906863  5.51918578
   5.44435093  5.79045142]
 [ 5.32487907  7.15545331  4.82970017  5.1067955   4.96265668  5.97926601
   6.37530517  5.32264676  4.60859224  4.99715996  4.74365156  4.87711584
   4.82578735  5.95557748  5.51113136  5.91186601  5.13975868  5.03322439
   6.02341984  5.59351288]
 [ 4.54850448  4.82970017  6.64270751  5.08443529  4.69132687  4.75352801
   5.76387132  4.50427796  4.47216188  4.6001877   4.569965    4.73593736
   4.99654771  5.85905799  5.20653444  4.86423093  5.13772027  4.36640164
   5.56194133  5.64557202]
 [ 4.9588283   5.1067955   5.08443529  6.48374364  4.14529358  5.47434209
   6.03385394  5.26156884  4.64970405  5.25399015  4.75918074  4.62345585
   4.884777    5.34169081  5.1288448   5.31575308  5.04162868  5.10903926
   5.63235552  5.51423642]
 [ 4.35244909  4.96265668  4.69132687  4.14529358  5.61533632  4.66409169
   4.97497857  4.4265963   3.63564229  3.95252087  3.68602166  3.98090863
   4.21219629  4.7768733   5.12110848  4.8867014   4.24586703  4.44488858
   5.10510211  4.86393583]
 [ 5.61432289  5.97926601  4.75352801  5.47434209  4.66409169  7.82957436
   6.5836467   5.25087102  5.49040364  5.43274849  4.64214472  4.80288734
   5.30989165  6.31101423  5.58773916  5.92120014  5.47323418  5.38275699
   6.40853124  5.99832338]
 [ 5.77008386  6.37530517  5.76387132  6.03385394  4.97497857  6.5836467
   8.18611511  5.86217521  5.67584227  5.85830841  5.6271472   5.49875177
   5.46870059  6.75925521  5.82046568  6.20767677  5.66042868  5.55248253
   6.89274163  6.16332371]
 [ 5.61767291  5.32264676  4.50427796  5.26156884  4.4265963   5.25087102
   5.86217521  6.44696501  4.62291385  4.87205414  4.50769992  4.59693701
   4.23909998  5.35729878  5.15256701  5.31119569  4.79669794  5.64783165
   5.38875859  5.3690519 ]
```

```
P_C1 = 0.5
P_C2 = 0.5
```

Figure 7 - LDA parameters in a mixture of Gaussians dataset

```
_____ calculating w and w0_____
W:
[  3.16706349e-02   3.57864853e-02   1.60490688e-02  -7.18858398e-05
  -1.76696721e-02   1.70332921e-02   5.91010070e-02   3.29540498e-02
  -4.98813340e-02  -1.68665720e-02  -5.36337471e-02  -5.03943613e-02
   6.65482547e-03  -3.74621726e-02   2.59354867e-02  -4.48244639e-02
   3.18229220e-02   1.52943747e-02   6.26852702e-03  -5.48815946e-02]
W0:
0.0560507987657
```

Figure 8- W and W0 of LDA classifier in a mixture of Gaussians dataset

```
_____ Train Evaluation (LDA) _____

Accuracy of train in LDA:
46.25%
```

Figure 9- Train set Accuracy Evaluation by using LDA

```
_____ classification Evaluation *** LDA *** _____

Accuracy: 47.0833333333%

Precision: 47.0288624788%

Recall: 46.1666666667%

F1-measure: 46.5937762826%
```

Figure 10- LDA Performance in a mixture of gaussians dataset

KNN classifier performance over different values of k when dataset is a mixture of 3 Gaussians:

```
__ K = 1  ___ Confusion matrix = [[324, 287], [276, 313]]

Accuracy: 53.0833333333%
Precision: 53.0278232406%
Recall: 54.0%
F1_measure: 53.5094962841%

__ K = 3  ___ Confusion matrix = [[336, 297], [264, 303]]

Accuracy: 53.25%
Precision: 53.0805687204%
Recall: 56.0%
F1_measure: 54.501216545%

__ K = 5  ___ Confusion matrix = [[328, 298], [272, 302]]

Accuracy: 52.5%
Precision: 52.3961661342%
Recall: 54.6666666667%
F1_measure: 53.5073409462%

__ K = 7  ___ Confusion matrix = [[329, 296], [271, 304]]

Accuracy: 52.75%
Precision: 52.64%
Recall: 54.8333333333%
F1_measure: 53.7142857143%

__ K = 9  ___ Confusion matrix = [[326, 292], [274, 308]]

Accuracy: 52.8333333333%
Precision: 52.7508090615%
Recall: 54.3333333333%
F1_measure: 53.5303776683%

__ K = 11  ___ Confusion matrix = [[334, 308], [266, 292]]

Accuracy: 52.1666666667%
Precision: 52.0249221184%
Recall: 55.6666666667%
F1_measure: 53.7842190016%

__ K = 15  ___ Confusion matrix = [[337, 300], [263, 300]]

Accuracy: 53.0833333333%
Precision: 52.9042386185%
Recall: 56.1666666667%
F1_measure: 54.4866612773%

__ K = 17  ___ Confusion matrix = [[328, 292], [272, 308]]

Accuracy: 53.0%
Precision: 52.9032258065%
Recall: 54.6666666667%
F1_measure: 53.7704918033%

__ K = 19  ___ Confusion matrix = [[324, 303], [276, 297]]

Accuracy: 51.75%
Precision: 51.6746411483%
Recall: 54.0%
F1_measure: 52.8117359413%

__ K = 21  ___ Confusion matrix = [[336, 299], [264, 301]]

Accuracy: 53.0833333333%
Precision: 52.9133858268%
Recall: 56.0%
F1_measure: 54.4129554656%

__ K = 23  ___ Confusion matrix = [[336, 302], [264, 298]]

Accuracy: 52.8333333333%
Precision: 52.6645768025%
Recall: 56.0%
F1_measure: 54.281098546%

__ K = 25  ___ Confusion matrix = [[341, 299], [259, 301]]

Accuracy: 53.5%
Precision: 53.28125%
Recall: 56.8333333333%
F1_measure: 55.0%
```

Figure 11- KNN performance over different values of K

```
Maximum Accuracy: 58.75  with k = 47

Maximum Precision: 59.3917710197  with k = 47

Maximum Recall: 55.3333333333  with k = 47

Maximum F1 Measure: 57.2907679034  with k = 47
```

Figure 12- Maximum Accuracy, precision, recall and F1 measure of KNN in a mixture of Gaussian dataset

According to figure 11, the maximum accuracy, precision, recall and F1-measure of KNN is for K= 47.

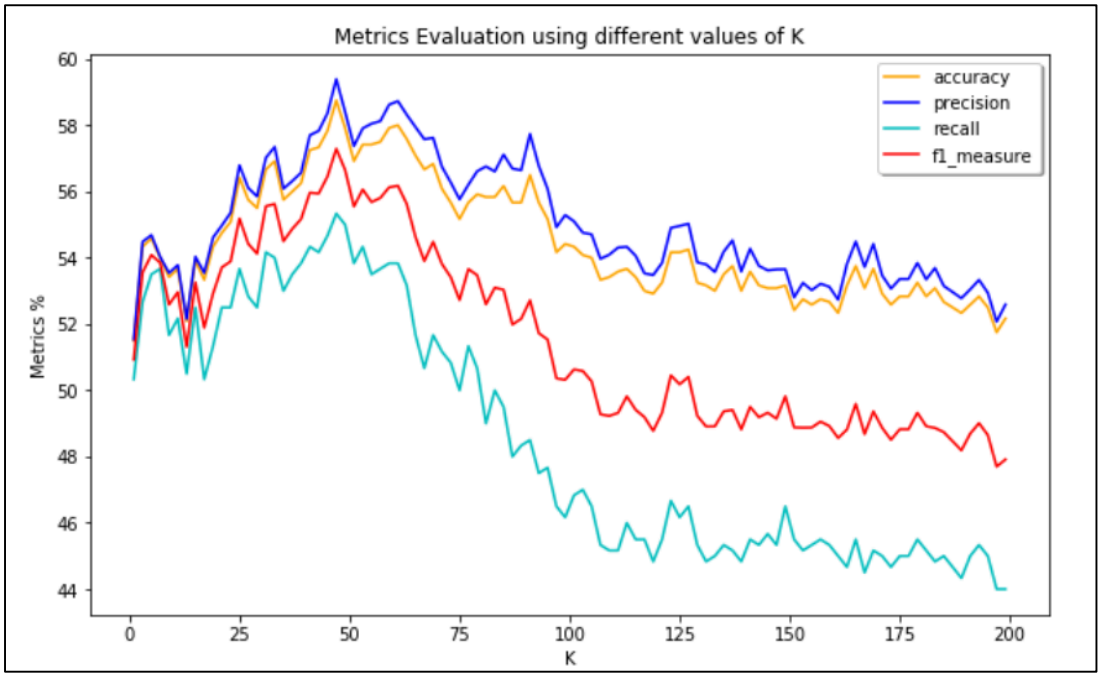


Figure 13- Figure 4- Evaluation Metrics over different values of K in a mixture of Gaussian dataset

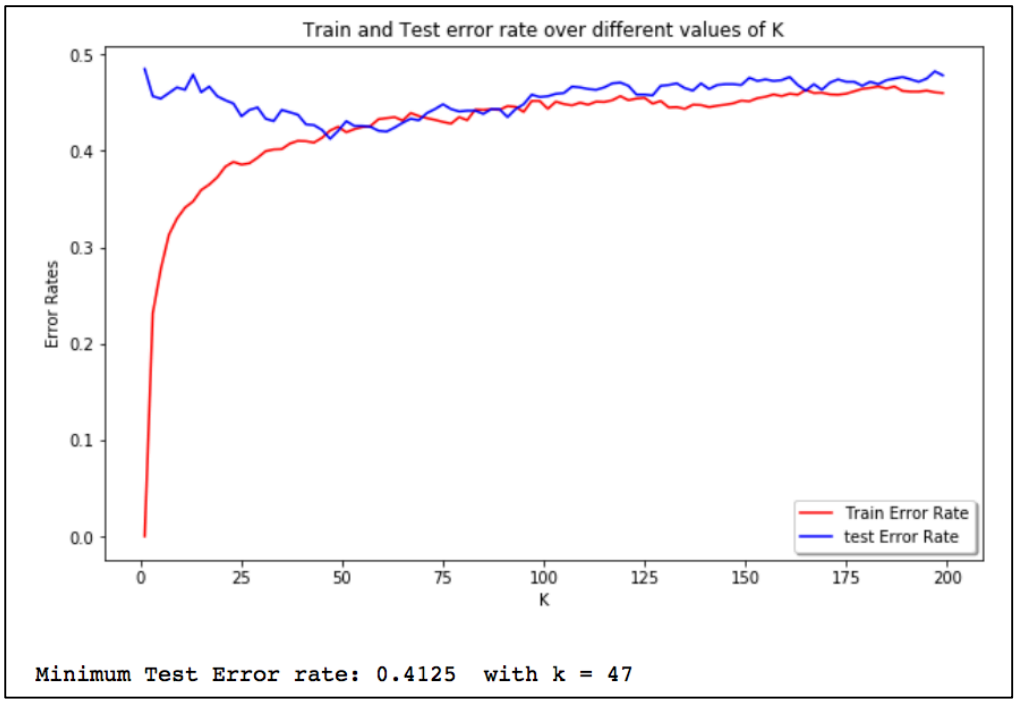


Figure 14- Test set and train set error rate over different values of K in a mixture of Gaussian dataset

6- LDA classifier accuracy for DS1 is almost 95% while for DS2 is nearly 47%. According to this information we can see that LDA has a better performance for DS1 rather than DS2, since in DS1 each class generated by just one Gaussian distribution while in DS2 each class is a mixture of 3 Gaussian, so in case of DS2 the estimations of mu_1 and mu_2 are not as precise as DS1's.
KNN average accuracy for DS1 and DS2 is almost 58%. So, there is no significant difference in KNN performance. In both datasets DS1 and DS2, KNN isn't work well and the reason is related to the curse of dimensionality and the fact that we need more data points in higher dimensions.
From these observations we can infer that linear model works very well on DS1 and then not that much worse than a non-linear classifier on DS2.