Jeffrey Kerley
Jakcqc

This question is to demonstrate that: 1) when the selected model is poor, the
maximum-likelihood classifier does not produce
satisfactory results; and 2) proper transformation of the data can compensate for poor models.
The dataset1 used for this
question is divided into training2
and test3 data, with each one consisting of 3 classes in a 2D-feature space.
a) Assuming Gaussian distribution for all three class conditional densities, and with unknown means and covariances,
compute the maximum likelihood estimates for each class using the training data.

> trainClass1Mean
[,1]
[1,] 0.2991254
[2,] -0.2062300
> trainClass1Sigma
[,1] [,2]
[1,] 1.18558231 -0.01828165
[2,] -0.01828165 0.68462313
> trainClass2Mean
[,1]
[1,] 0.3416033
[2,] 0.2564200
> trainClass2Sigma
[,1] [,2]
[1,] 2.7647359 0.1659947
[2,] 0.1659947 3.7044193
> trainClass3Mean
[,1]
[1,] 0.4470942
[2,] 0.4128826
> trainClass3Sigma
[,1] [,2]
[1,] 5.1824081 -0.7028792
[2,] -0.7028792 6.7559457

We assume MLE and get our estimates to be expected mean and covariance matrix for each training class.

b) Ignore the priors, i.e. assume 1/3 for all three classes, and redo parts a) and b) of MiniProject 2, Question 1, and use

those same Matlab functions in part c) below. Use the means and variances from part a) above.
We use these functions for question c.

c) Classify the test data and compute the test error using confusion matrix.
We run each testclass data into our discrim function and use a piecewise process to decide if class1 or class2 or class3.

```
> confMat[1,] = getConf(testClass1)
> confMat[2,] = getConf(testClass2)
> confMat[3,] = getConf(testClass3)
> confMat
        [,1]      [,2]      [,3]
[1,] 0.8260870 0.1739130 0.0000000
[2,] 0.1827957 0.6236559 0.1935484
[3,] 0.0000000 0.1333333 0.8666667
```

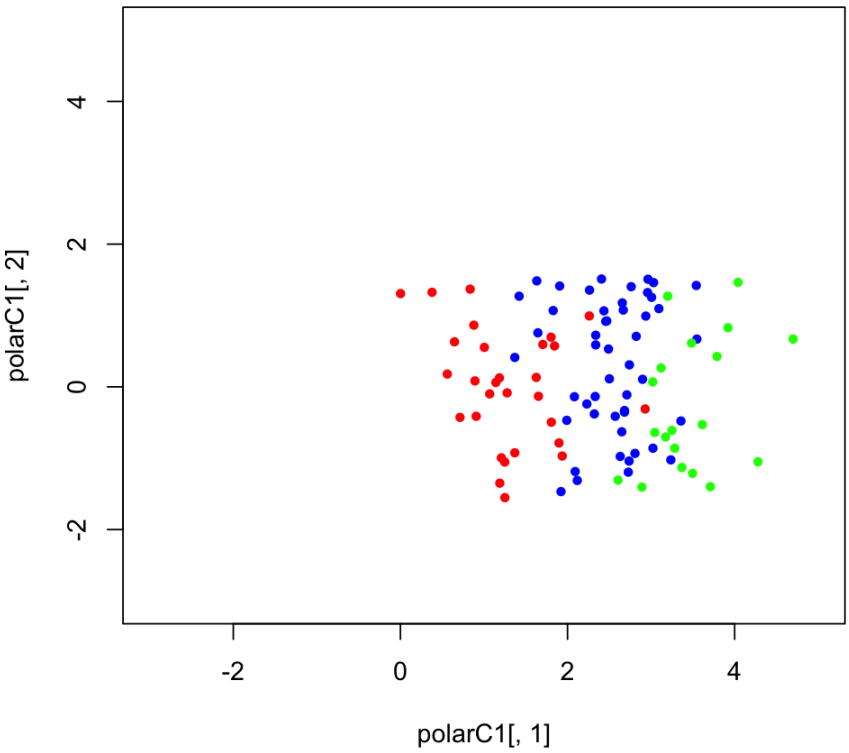
Above is our confusion matrix and we can see the results of the model and the test data set.

d) Bayesian estimates. The data has a simpler description when seen in polar coordinates. Use cart2pol() to transform all the data points to polar coordinates. Use scatter() to plot the transformed points. What you should find is that the transformed data looks Gaussian on the radius r and uniform on the angle θ. So, ignore the angle θ and classify the test data only on r as Follows.

```
> polarC1
           [,1]       [,2]
 [1,] 0.713282022 -0.42810917
 [2,] 1.189214879 -1.34944497
 [3,] 1.369130853 -0.92413116
 [4,] 1.208071079 -0.99457379
 [5,] 0.834742512  1.36915535
 [6,] 1.277322376 -0.08417367
 [7,] 2.260566111  0.99372567
 [8,] 1.142043735  0.06128119
 [9,] 1.844323955  0.57344527
[10,] 1.806013729 -0.49574033
[11,] 0.378640866  1.32521299
[12,] 1.004372298  0.55317673
[13,] 1.936380939 -0.96945851
[14,] 0.892051217  0.08327417
[15,] 1.625715559  0.13234016
[16,] 0.878716528  0.86416355
```

```
[17,] 1.651905213 -0.13215950
[18,] 1.068274689 -0.09846709
[19,] 1.185109152  0.12542254
[20,] 2.928882757 -0.31008211
[21,] 0.560938181  0.17915654
[22,] 1.704292197  0.59406457
[23,] 1.803273637  0.69630743
[24,] 0.906901426 -0.41297237
[25,] 0.001312031  1.30687257
[26,] 1.897733706 -0.78614338
[27,] 1.248846067 -1.05358694
[28,] 0.646928037  0.63106355
[29,] 1.250104213 -1.55272085
> polarC2
         [,1]       [,2]
 [1,] 2.898046  0.1056086
 [2,] 2.649178 -0.6295567
 [3,] 2.737124 -1.0399947
 [4,] 3.093340  1.0961609
 [5,] 1.368790  0.4120942
 [6,] 1.905928  1.4128363
 [7,] 2.958047  1.3213589
 [8,] 3.029938  1.4607204
 [9,] 2.435336  1.0648265
[10,] 2.681718 -0.3545747
[11,] 2.740272  0.3081445
[12,] 2.936505  0.9916815
[13,] 2.807240 -0.9326029
[14,] 2.457206  0.9180163
[15,] 2.330883 -0.1355567
[16,] 2.337077  0.7235154
[17,] 1.828157  1.0691346
[18,] 1.631998  1.4850355
[19,] 2.822212  0.7078000
[20,] 3.236273 -1.0234612
[21,] 2.091388 -1.1868227
[22,] 1.645037  0.7570678
[23,] 2.231282 -0.2402438
[24,] 2.500820  0.1125406
[25,] 1.990796 -0.4690345
[26,] 2.570898 -0.4125530
[27,] 2.709897 -0.1120615
[28,] 2.490258  0.5300891
[29,] 2.962888  1.5082828
```

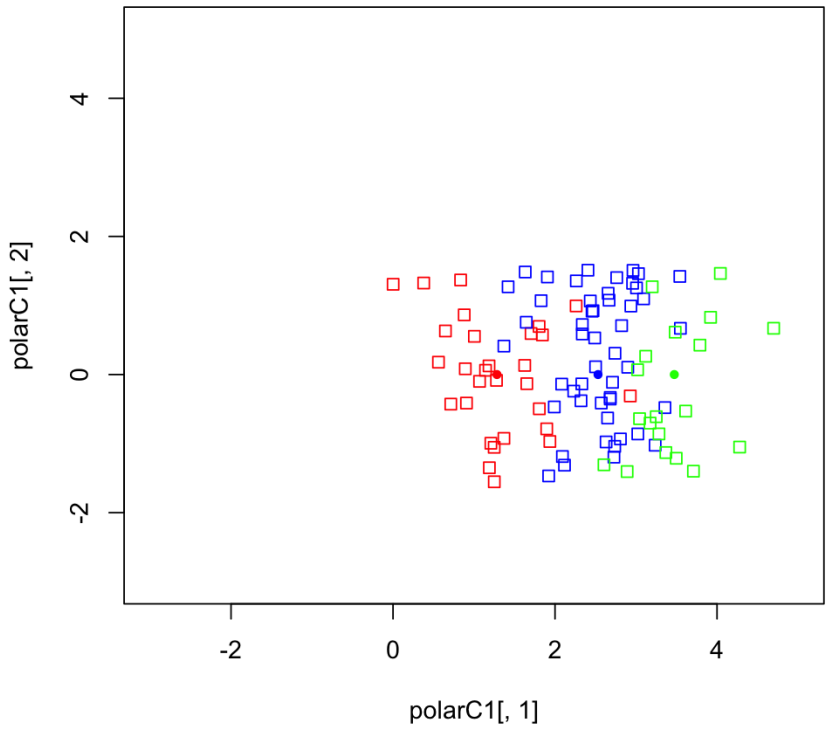
```
[30,] 2.655508  1.1765556
[31,] 3.357073 -0.4786482
[32,] 2.336647  0.5876228
[33,] 1.420580  1.2702438
[34,] 1.921651 -1.4679639
[35,] 2.761800  1.4034107
[36,] 2.630597 -0.9755083
[37,] 2.681978 -0.3317953
[38,] 2.406387  1.5110988
[39,] 3.547821  0.6693001
[40,] 3.006145  1.2541561
[41,] 2.469295  0.9238367
[42,] 3.022218 -0.8599162
[43,] 2.082429 -0.1384587
[44,] 2.320673 -0.3795662
[45,] 2.116697 -1.3119849
[46,] 2.727323 -1.1958016
[47,] 3.540950  1.4206286
[48,] 2.667096  1.0764588
[49,] 2.263148  1.3557090
> polarC3
         [,1]        [,2]
 [1,] 3.483644  0.61372903
 [2,] 2.604344 -1.30677570
 [3,] 3.119447  0.26470034
 [4,] 3.612758 -0.52924002
 [5,] 4.699120  0.66992189
 [6,] 3.199689  1.27102949
 [7,] 3.496054 -1.21222318
 [8,] 3.173591 -0.70297671
 [9,] 3.283341 -0.85950743
[10,] 3.019412  0.06912811
[11,] 4.279390 -1.05002710
[12,] 3.708787 -1.39917145
[13,] 2.890326 -1.40480811
[14,] 3.920634  0.82876915
[15,] 3.788955  0.42582150
[16,] 3.368356 -1.12945577
[17,] 3.249688 -0.61032261
[18,] 3.043890 -0.63905941
[19,] 4.041279  1.46410547
```



X = r, y = theta

The problem is now 1-D and again, if you inspect the data, Gaussian distribution is a more suitable pdf to describe all three classes. Assume then that each class has p(r|ωi) = Nμi,σ2 with μi unknown and variance σ2 = 0.23 for all three classes. Let the only prior knowledge about μi be p(μi) = Nμ0 = 0, σ20 = 110 and compute the Bayes estimates for μi and the posterior distribution p(μi |Di) of all three classes. Next compute p(r|ωi,Di) = ´p(r|μi) p(μi|Di)dμi and use this density estimate to
classify the test data and compute the test error using confusion matrix.

We see our bayes estimates for mean of r:
We get this through our formula for mu *sub n* that includes our mu sub 0, and var^2 sub 0.



Although I could not apply the rest, we can use the Reproducing density formula in the book to find p (mu | D). This is used with our bayes var2n and bayes mu sub n estimates substituted in a gaussian density function.