ECE 8110: INTRODUCTION TO MACHINE LEARNING AND PATTERN RECOGNITION

HOMEWORK # 2

ML VS. BAYESIAN ESTIMATION

Vira Oleksyuk

04/18/2014

1. **Generate two 2D GRVs with a mean of GRV1(1,1) and GRV2(-1,-1) respectively. Plot the theoretical probability of error for an ML classifier as a function of the prior probabilities and the covariance matrices. Since there are a number of degrees of freedom, determine the best way to visualize the results.**

I generated two 2D Gaussian random variables GRV1 and GRV2 and plotted their probability distributions and support regions on Fig. 1. Third graph on Fig. 1 shows generated data points and corresponding support regions of their probabilities.

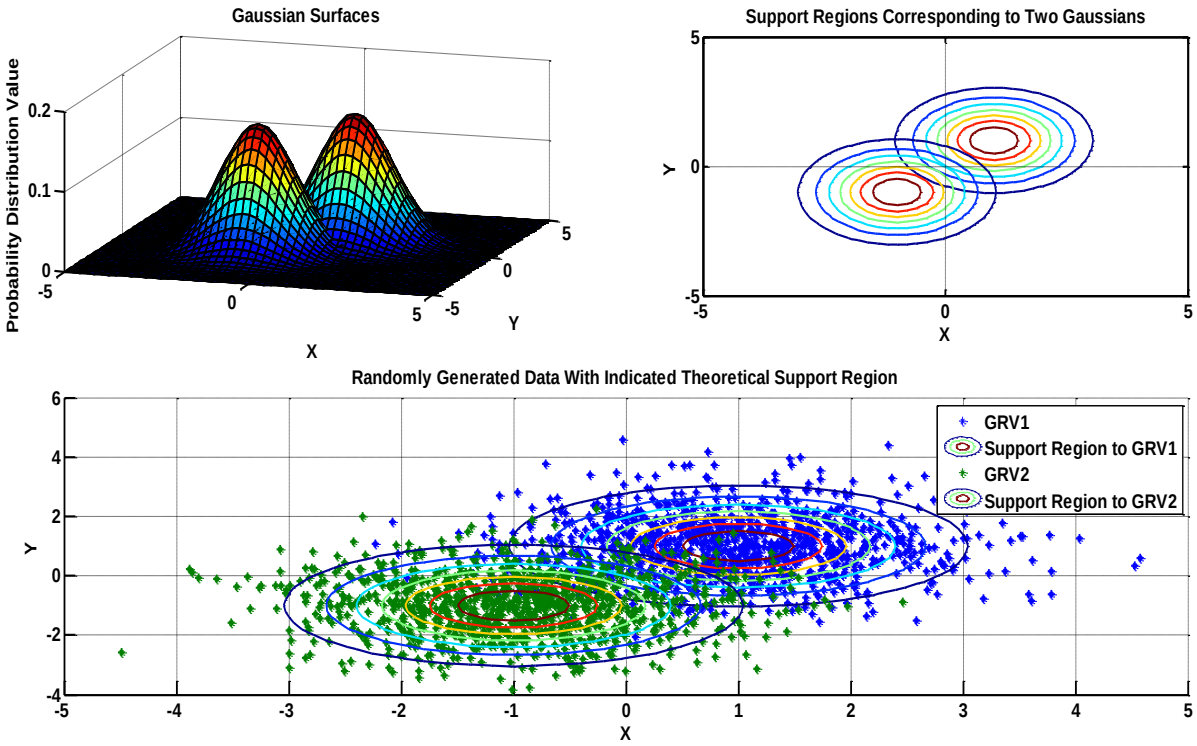


Figure 1. Graphical representation of GRV1 and GRV2 with their support regions

In this problem we have to investigate how theoretical probability of error of ML classifier changes with change in priors and covariance matrices.

Assuming equal identity covariance matrices for both GRVs, we can say that the decision surface will be a straight line. It will be on the middle distance between two mean of those GRV. This conclusion we can clearly see on the plot of decision surfaces (Fig.1). We also can calculate it using following equations [2].

$$g = x'W_i x + w_i' x + w_{i0}, \quad (1)$$

$$W_i = -\frac{\Sigma_i^{-1}}{2}, \quad (2)$$

$$w_i = \mu_i \sum_i^{-1}, \tag{3}$$

$$w_{i0} = -\frac{\mu_i' \sum_i^{-1} \mu_i}{2} - \frac{\log(det \sum_i)}{2} + \log(P_i) \tag{4}$$

So the decision surfaces can be easily calculated using those four equations. I calculated those for each GRV and then equate g1=g2. I had to specified prior probabilities of GRV1 and GRV2, so I varies them as P1=0.001, P2=1-P1; P1=0.1; P1=0.5; P1=0.9; P1=0.999. Results of those calculations presented on Fig.2.

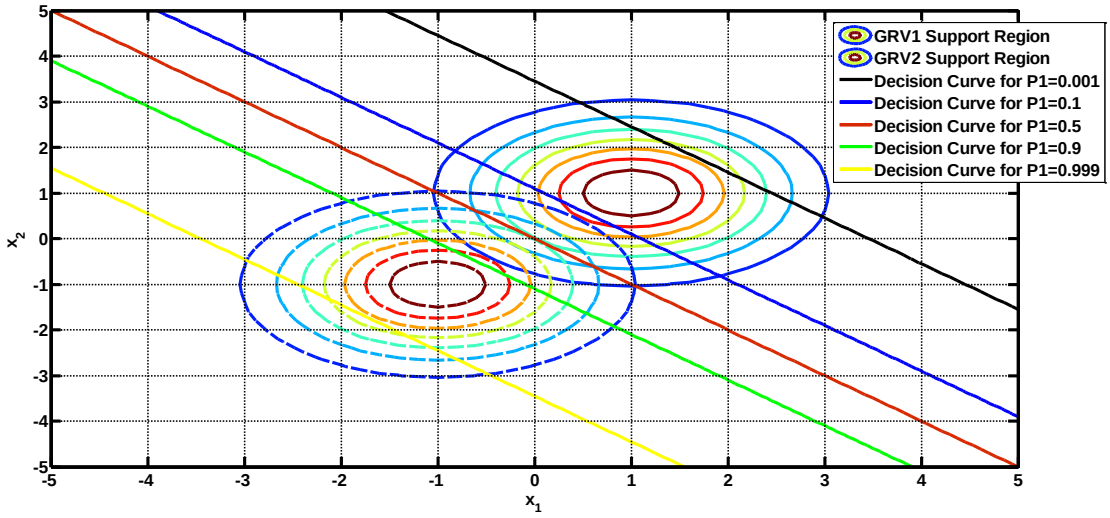


Figure 2. Decision curves corresponding to different prior probabilities of GRV1 and GRV2

One can see form the Fig. 2 decision line goes right in between of two GRV decision surfaces. When prior probabilities are not equal, decision line is shifting closer to one or another GRV. If prior probability of GRV is almost zero (0.001), it will have almost no influence on the decision curve, and vice versa.

To find theoretical probability of error for an ML classifier I will have to integrate probability distribution functions in regions, which are behind the decision surface for each GRV. It is easy to do if decision region is linear, and is not so easy if it is not. Table 1 presents results for theoretical error calculation. We can see how error will for each of the new prior pair.

Table 1. Theoretical Error of ML Classifier (identity covariance matrices)

Case	P1=0.001 P2=0.999	P1=0.1 P2=0.9	P1=0.5 P2=0.5	P1=0.9 P2=0.1	P1=0.999 P2=0.001
Theoretical Error	0.0014	0.0430	0.0786	0.0430	0.0014

Next, I was trying to vary covariance matrixes for GRV. I was able to find decision surfaces for those,

which are not lines but curves. As I mentioned before, it is easy to calculate integral of 2d Gaussian surface along straight line to find theoretical error. It is not easy to do so for integration along curves. I was not sure how to complete this task.

It was shown in HW1 how covariance matrix changes causes support region transformation. I will try to visualize how decision curves changing with change of covariances and priors. I will use four figures to visualize it. Means will be constant to both GRV and equal to [1 1] and [-1 -1].

a) Let $\sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and $\sigma_2 = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}$. This case will show dependency on magnitude changes.

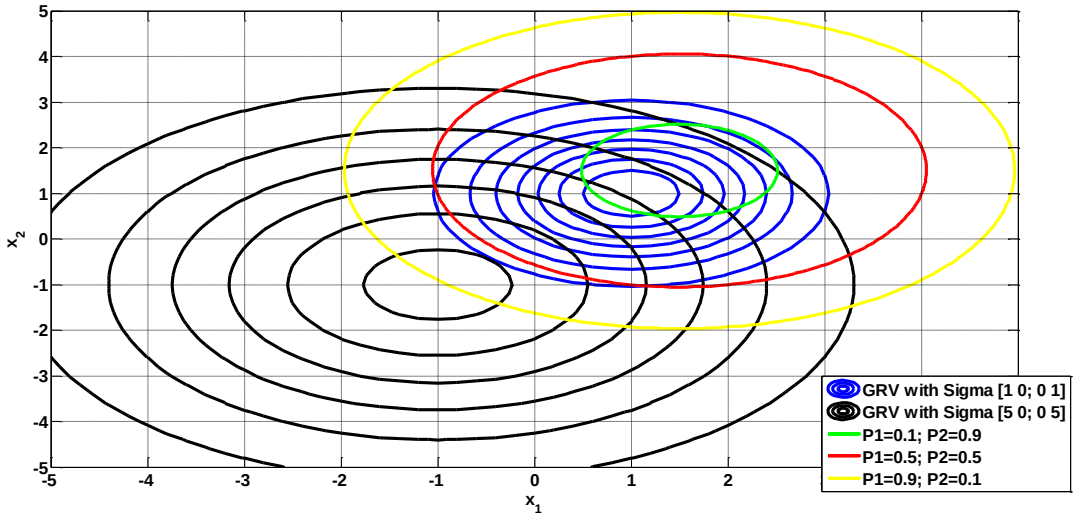


Figure 3. Decision curves for covariances [1 0;0 1] and [5 0;0 5] and varying priors

b) Let $\sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and $\sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix}$. This case will show dependency on horizontal stretch of one of the GRV.

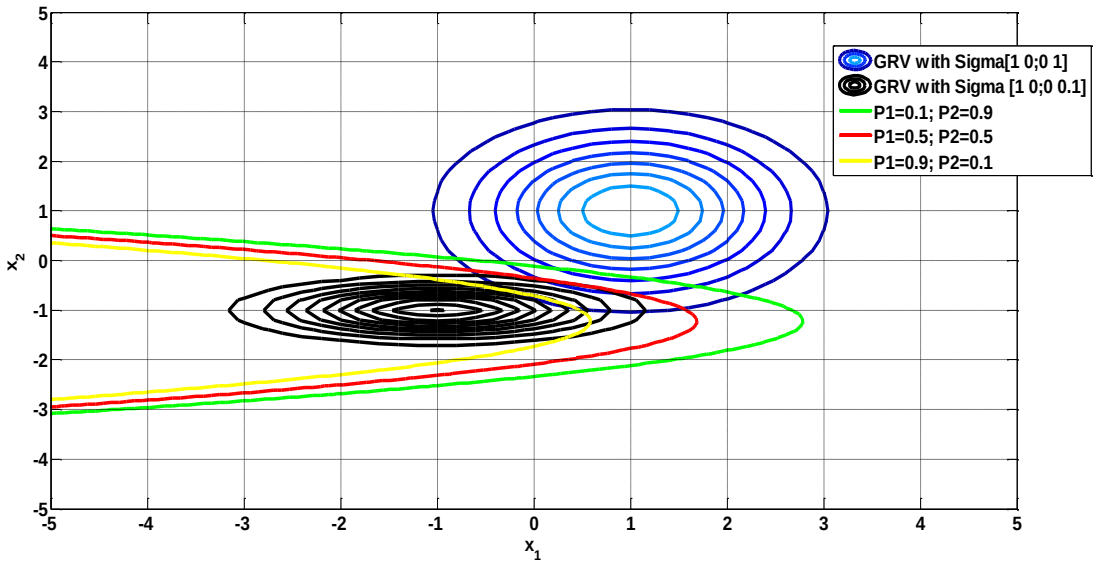


Figure 4. Decision curves for covariances [1 0;0 1] and [1 0;0 0.1] and varying priors

c) Let $\sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and $\sigma_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 3 \end{bmatrix}$. This case will show dependency on vertical stretch of one of the GRV.

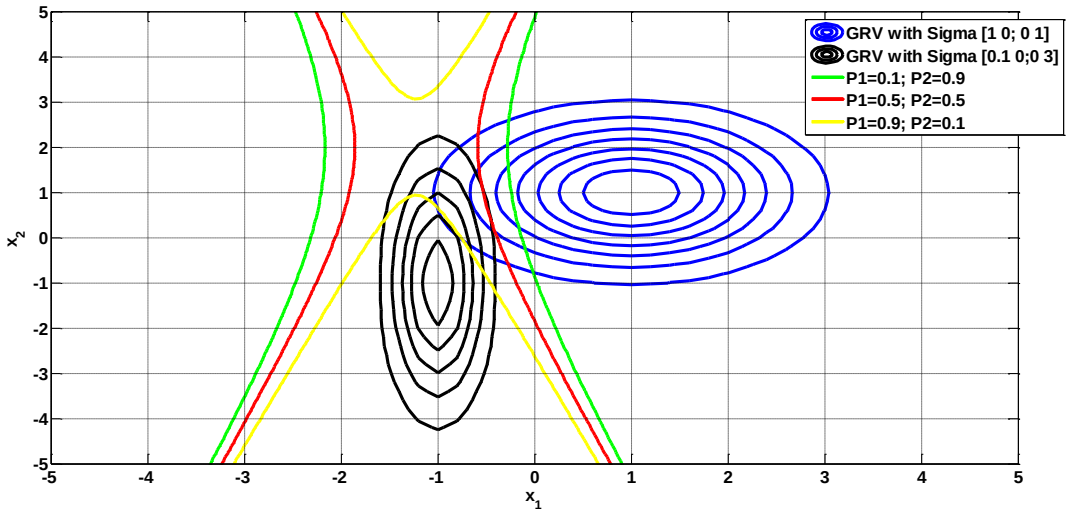


Figure 5. Decision curves for covariance matrices [1 0;0 1] and [0.1 0;0 3] and varying priors

d) Let $\sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and $\sigma_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$. This case will show dependency on diagonal stretch of one of the GRV.

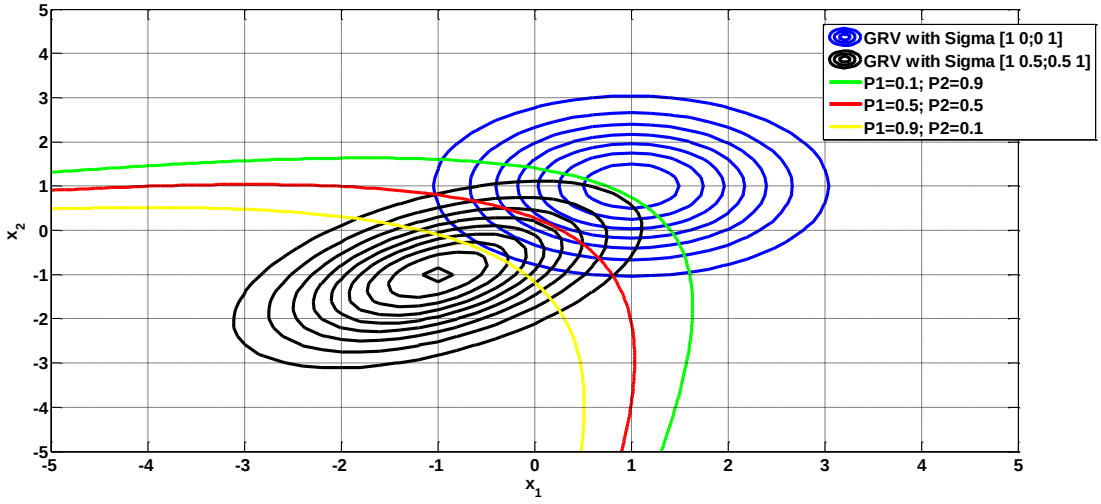


Figure 6. Decision curves for covariance matrices [1 0;0 1] and [1 0.5;0.5 1] and varying priors

As we can see form these figures, decision curves try to align with covariance matrices of GRVs, so error rate form classification will be minimal. It is very difficult to show numerical results for this problem.

2. Compare the theoretical results in (1) to those obtained when you construct an ML classifier by generating [100, 1,000, 10,000, 100,000] random variables for each class. Estimate the means and variances from the data. Only consider the equal priors case for this example, and focus on a small representative set of covariance matrices.

Table 2. Error for ML classifier

P1	P2	100 Samples	1 000 Samples	10 000 Samples	100 000 Samples	Theoretical
0.001	0.999	0.00040	0.0019	0.0018	0.0018	0.0019
0.1	0.9	0.02950	0.0503	0.0435	0.0457	0.0455
0.5	0.5	0.07500	0.0788	0.0799	0.0794	0.0793
0.9	0.1	0.04650	0.0468	0.0454	0.0454	0.0455
0.999	0.001	0.00036	0.0019	0.0018	0.0018	0.0019

Table 3 shows how error for mean and covariance estimation decreases with increase of samples number. I used two mean value and identity covariances. Figure 7 shows those results graphically.

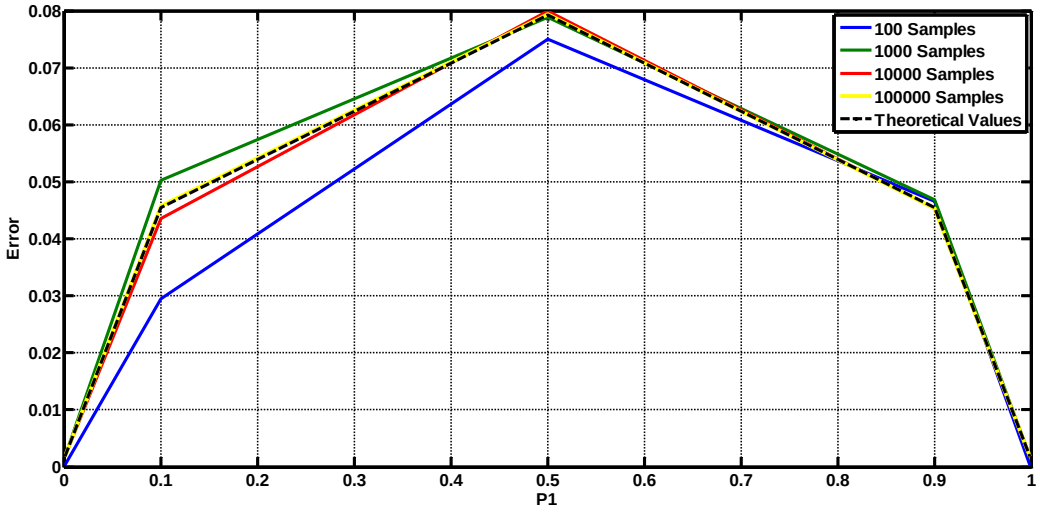


Figure 7. Dependency of error on number of experimental samples

Table 3. Approximation of means and variances for data.

Case	GRV 1		GRV 2	
	Mean	Variance	Mean	Variance
TRUE	[1 1]	1 0 0 1	[-1 -1]	1 0 0 1
100 samples	1.0512 0.8450	0.8150 0 0 0.9612	-1.0154 -0.9184	0.7285 0 0 0.8777
1000 samples	1.0006 0.9572	0.9465 0 0 1.0557	-1.0690 -1.0367	0.9997 0 0 0.9596
10000 samples	1.0093 0.9995	1.0058 0 0 0.9877	-0.9770 -0.9957	1.0054 0 0 0.9999
100000 samples	1.0023 1.0024	1.0004 0 0 1.0059	-0.9967 -1.0035	0.9975 0 0 1.0024

As we can see form the Table 3, the more data generates for GRV, the closer to true value estimated mean and variance get. 10000 and 100000 samples are a very good number of data points for accurate calculation. 100 samples will be not enough. That is why ML is not a good choice for experiments with small number of samples. It becomes biased.

3. Generate two 2D Gaussian GRVs with means of [-2,-5] and [3, 6] and covariance matrices of $\begin{pmatrix} 2.0 & 1.5 \\ 1.5 & 2.0 \end{pmatrix}$ **and** $\begin{pmatrix} 3.0 & -2.0 \\ -2.0 & 3.0 \end{pmatrix}$.

a. Construct an ML estimator and measure the error rate.

b. Convert each GRV to a Gaussian GRV with an identity covariance matrix by performing Principal Components Analysis (PCA).

c. Classify each data point by transforming it to the PCA space using a whitening transformation and computing the distance from the mean. Select the class assignment by choosing the class that has the smallest distance. This is essentially an ML classifier, but implemented in a slightly different way. Do your results match part (a)?

d. Examine the eigenvectors of each covariance matrix and relate those to the support region for each GRV. It is preferable to visualize this with a graph of the vectors overlaid on the support region.

In this problem, we will have to complete stepwise calculations. Figure 8 illustrates support regions for GRV1 and GRV2 with the corresponding decision surfaces.

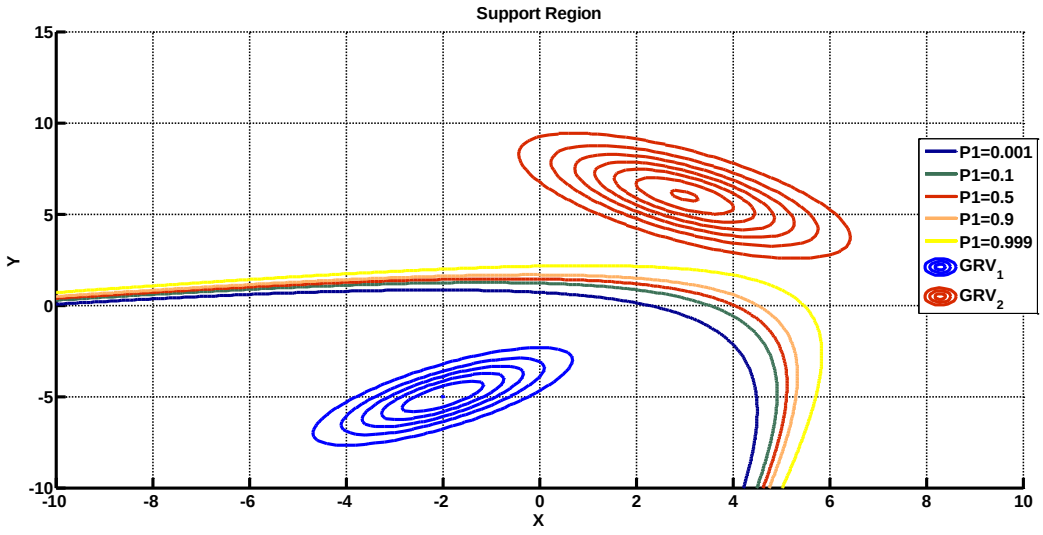


Figure 8. Support regions for GRV1 and GRV2 with the corresponding decision surfaces.

a) GRV1 and GRV2 are not overlapping, so the error rate will be very small if any. The calculated decision surface separates two distributions completely, and error was found as zero.

b) PCA analysis.

PCA method works with projection that best represents the data in a least-square sense. I will complete sequential steps [3] to provide PCA analysis of the generated data.

- Generate data as two dimensional Gaussian random variables with means and covariance matrixes as given (GRV1 and GRV2). Support regions of those GRVs shown on Figure 8. Assume we do not know their means and variances after GRVs were generated (as if it would be a real data set).
- Find the mean across each dimension. Each of two columns of GRV1 and GRV2 has calculated mean. I will subtract the mean from each of the data dimensions. This operation will drive the mean of each GRV to zero, which supported by following results.

$$\mathbf{m}_i = \frac{1}{n_i} \sum_{\mathbf{x} \in D_i} \mathbf{x}$$

$$\mathbf{S}_i = \sum_{\mathbf{x} \in D_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^t$$

%Mean for each GRVs
x0_1 = -1.9426 -4.9303
x0_2 = 3.0132 5.9676
%Mean for Adjusted to the mean GRV1 and GRV2
mean(GRV_adj_1)
-0.2238 -0.0078 (1.0e-14)
mean(GRV_adj_2)
-0.1946 -0.4816 (1.0e-14)

Figure 9 shows how GRV1 and GRV2 with zero mean distribute their values.

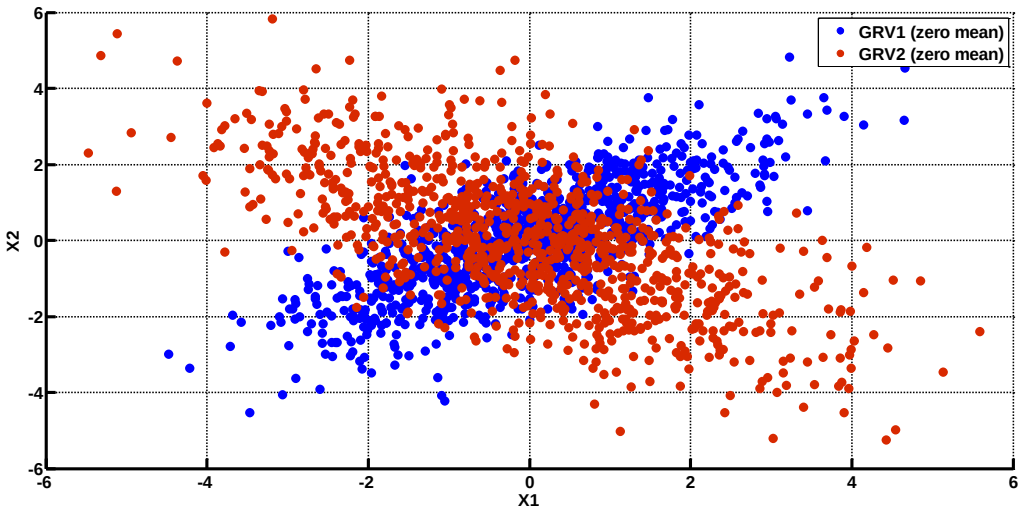


Figure 9. GRVs with zero mean

- Next, calculate the covariance matrices for each GRV using following equation:

$$cov(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \hat{X})(Y_i - \hat{Y})}{n-1}$$

In Matlab I calculated it using *cov(X,Y)* function and I received a 2x2 matrix. Results are following.

cov1 =		cov2 =	
1.9912	1.5087	3.0982	-2.0456
1.5087	1.9714	-2.0456	3.0924

Covariance1 and covariance2 are almost the same as the given in the problem values of covariance matrices.

- Calculate eigenvalues and eigenvectors of covariance matrices. Two cov matrices are square matrices, so the calculation is very easy. All eigenvectors are unit eigenvectors, which is automatically done in Matlab. Results of this step calculations presented following:

V1 =		V2 =	
0.7048	-0.7094	-0.7066	-0.7076
-0.7094	-0.7048	-0.7076	0.7066
D1 =		D2 =	
0.4725	0	1.0497	0
0	3.4900	0	5.1409

- Where V1 and V2 are eigenvectors of GRV1 and GRV2 accordingly. D1 and D2 – diagonal matrices of eigenvalues for GRV1 and GRV2. Graphically data sets with corresponding eigenvalues are presented in the Fig.10. As we can see from Fig. 10, eigenvectors are orthogonal to each other for each GRV.

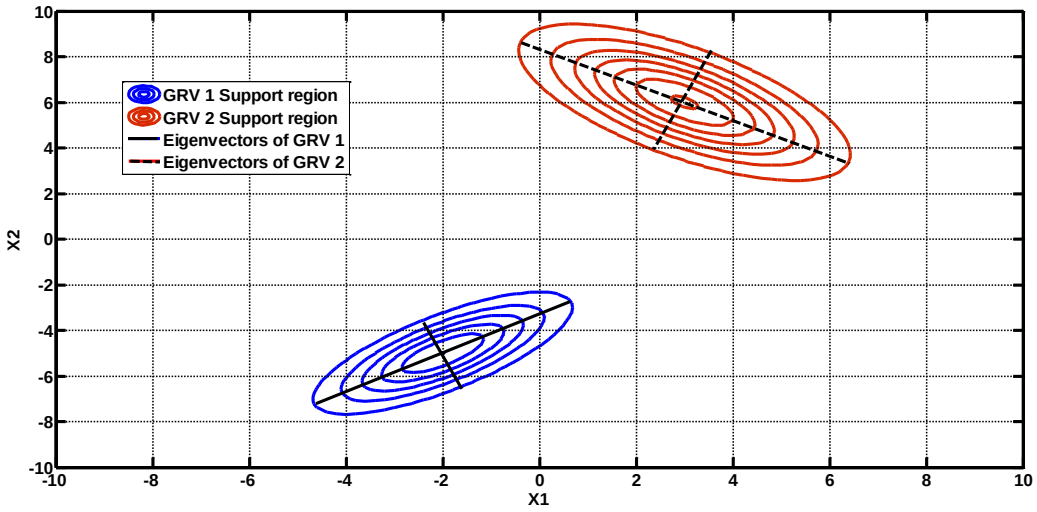


Figure 10. GRVs support regions and corresponding eigenvectors

- The decision calculated as following

$$\mathbf{w} = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$$

- As a next step, I will try to reduce number of the data dimensions. The eigenvector with greatest eigenvalue is the principle component of each data set. If data is multidimensional, we would have to rearrange eigenvectors from higher to lower eigenvalue, which gives us a notion of significance. If we need to decrease the dimensionality, we would neglect eigenvectors with lower eigenvalues. Two feature vectors are:

$$FeatureVector = (eigV1\ eigV2\ eigV3 \dots\ eigVn);$$

So we will neglect the smaller eigenvector, and use only the largest.

- Next, I will find the new data set. The chosen eigenvector will be transposed and multiplied on the original data.

$$FinalData = RowFeatureVector * RowDataAdjust$$

RowFeatureVectr is a eigenvectors matrix (transposed); *RowDataAdjust* is the mean adjusted and transposed original data set. Figures 11-14

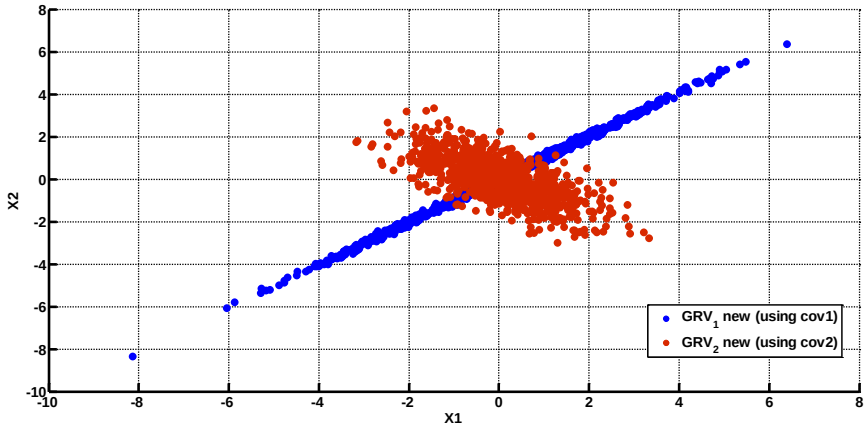


Figure 11. Whitening the data using cov1 and cov2 matrices

=>

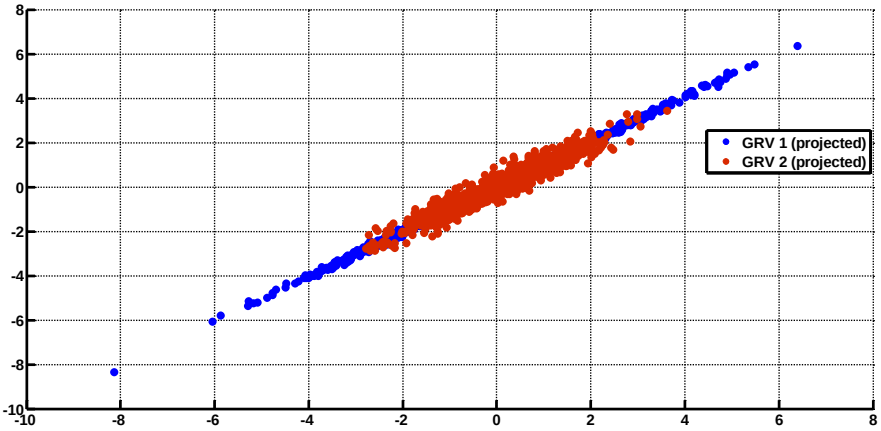


Figure 12. Whitening the data using only cov1 matrix

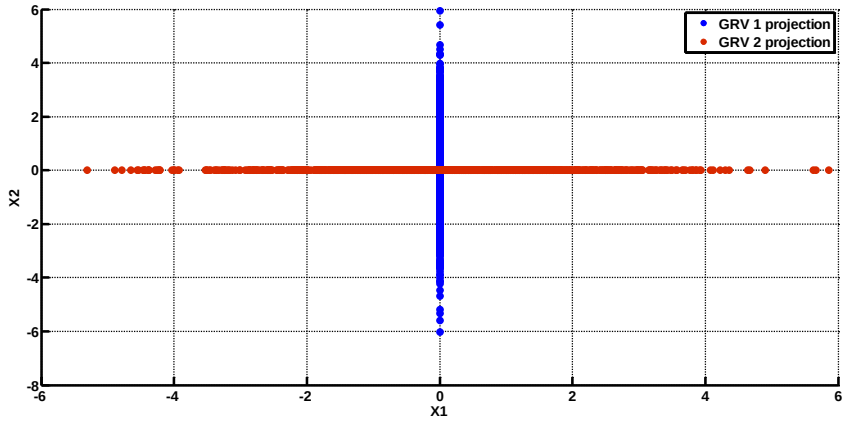


Figure 13. Manipulation with the data using cov1 and cov12 matrices

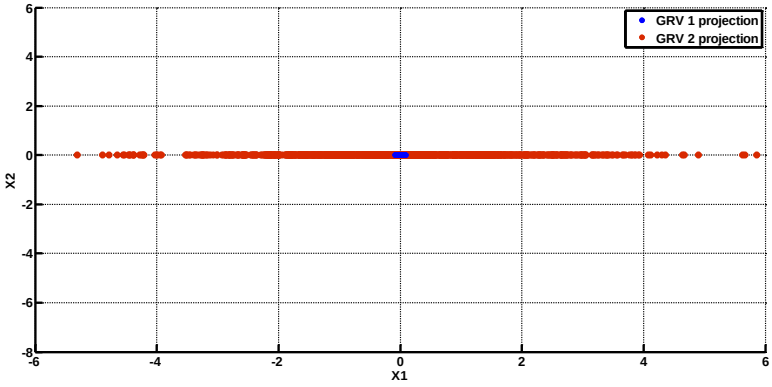
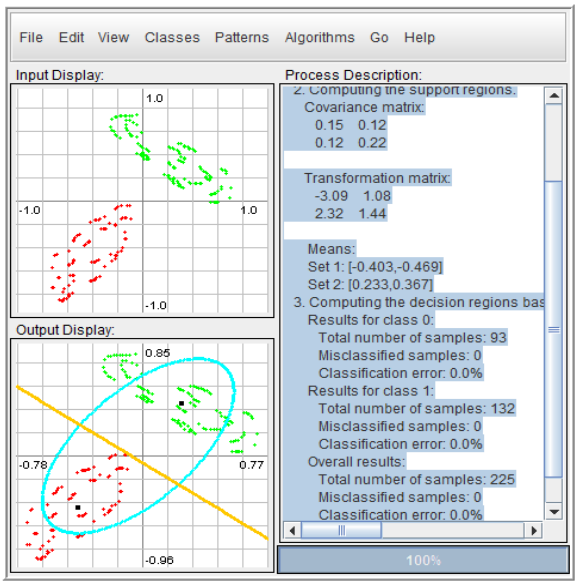


Figure 14. Final data projection for classification

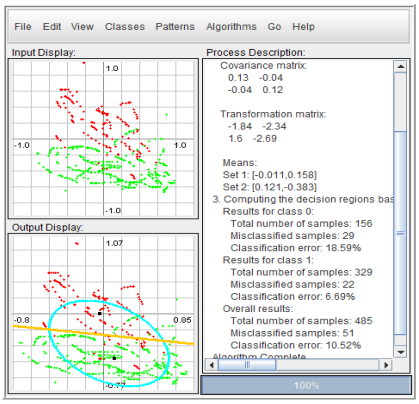
Figure 14 shows resulting projections of GRV1 and GRV2 on a line. The error of classification should be very small in this case. By calculation the error was found to be 2 samples misclassified form GRV1. Error 2/2000*100%=0.1%. This error is consistent with previous result (3.a.), which differentiated two classes without misclassification (for a 1000 data points set).

I also simulated several cases with the Java applet [4]. Figure 15 presents screen shots from those simulations. Fi.15.a shows very similar data sets to the given sets in the problem 3. The error of classification is zero as well.

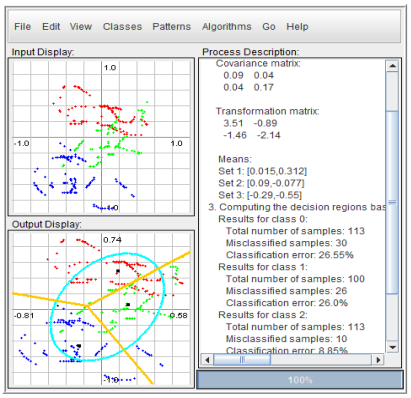
Fig.15b and 15c shows more complicated cases of classification with PCA.



a.



b.



c.

Figure 15. PCA algorithm simulation with the Java applet [4]

4. Following the example presented in the notes, assume you have a 1D GRV with mean and variance. Demonstrate estimation of the mean and variance using the theoretical results derived in class for Bayesian estimation. Compare this to a ML estimation. Show convergence as the number of data points is increased from 100 to 100K.

In this problem, we are working with 1D GRV, for which we do not know its mean and variance. We have to estimate its mean and variance using Bayesian and ML estimators, and compare results for both methods with increase of drawn data points from 100 to 100000. We are going to use following equations form the class notes to estimate mean and variance of a random variable:

$$\mu_n = \left(\frac{n\sigma_0^2}{n\sigma_0^2 + \sigma^2}\right)\hat{\mu}_n + \left(\frac{\sigma^2}{n\sigma_0^2 + \sigma^2}\right)\mu_0$$

$$\sigma_n^2 = \frac{\sigma_0^2\sigma^2}{n\sigma_0^2 + \sigma^2}$$

Where estimated mean form drawn samples

$$\hat{\boldsymbol{\mu}}_n = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k$$

Estimation of the mean and variance is based on the given data set, so one should expect dependency of the results on the amount of the drawn samples D. With number of samples approaching infinity, we should expect estimated values of mean and variance approach their true values. μ_0 and σ_0 are taken from the prior assumption about the data distribution ($N(\mu_0, \sigma_0) \sim N(1,1)$).

Figures 16-18 show histograms of samples' values drawn from the same random normal distribution. Number of samples vary from 5 to 100,000.

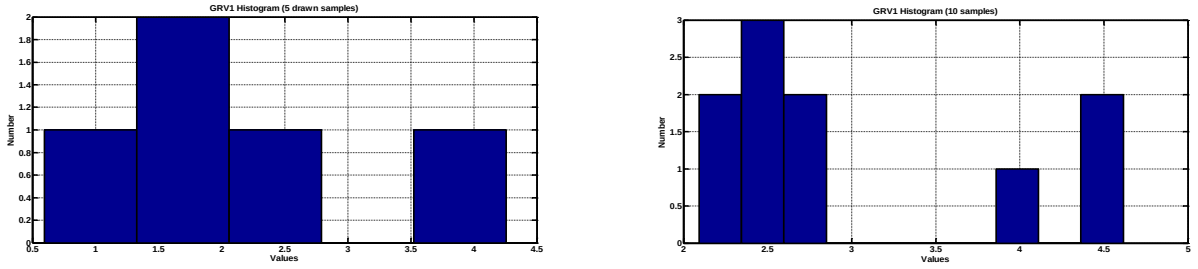


Figure 16. Histograms of the data distribution drawn from GRV (5 and 10 samples)

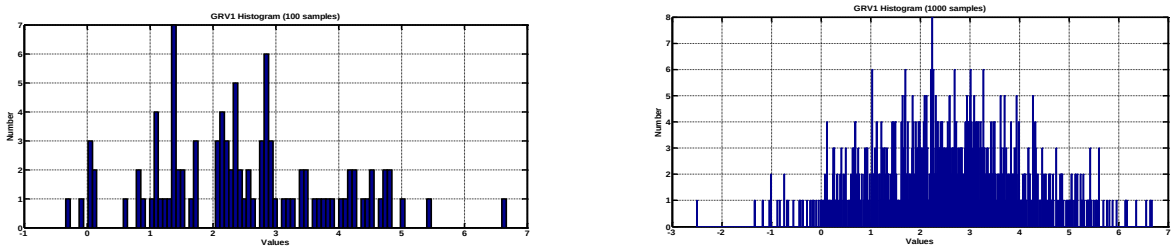


Figure 17. Histograms of the data distribution drawn from GRV (100 and 1000 samples)

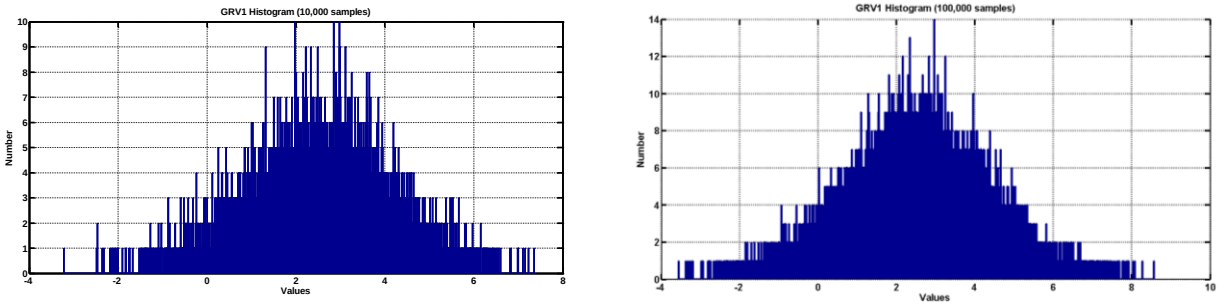


Figure 18. Histograms of the data distribution drawn from GRV (10,000 and 100,000 samples)

Figure 19 displays for which samples size the estimated mean value will converge to the true mean value in Bayesian estimation.

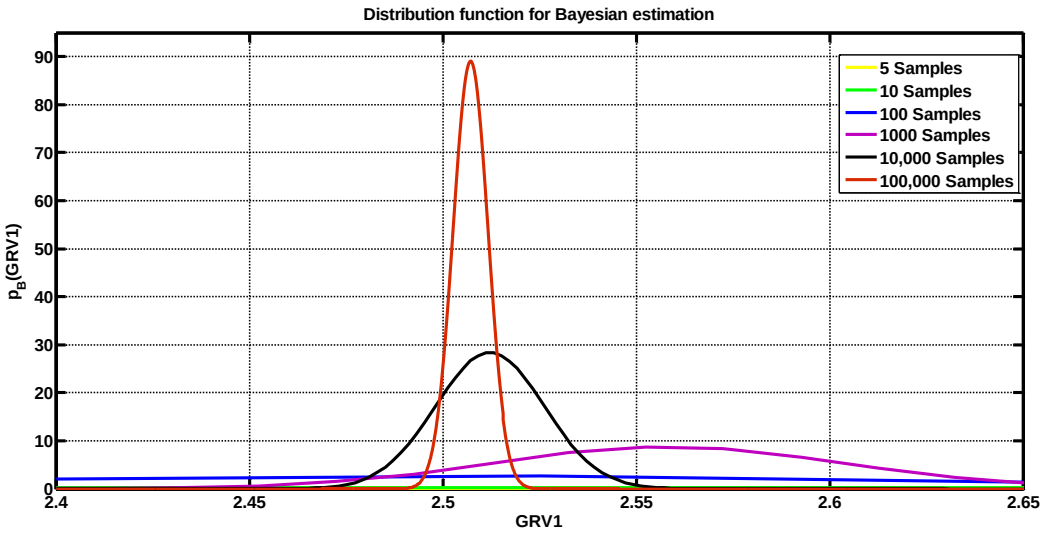


Figure 19. Bayesian Estimation of the Mean and Variance (for the range of 5 to 100,000 samples)

Figure 20 shows for which samples size the estimated mean value will converge to the true mean value in ML estimation. Numerical results are presented in Table 4.

Table 4. Approximation of means and variances with ML and Bayesian estimation (prior pdf assumed as *N(1,1)*).

Case	ML		Bayesian	
	Mean	Variance	Mean	Variance
TRUE	2.5	2	2.5	2
5 samples	3.1646	1.7935	2.5931	0.2640
10 samples	3.0201	2.1820	2.6583	0.1791
100 samples	2.6694	1.7479	2.6407	0.0172
1,000 samples	2.4595	2.0260	2.4566	0.0020
10,000 samples	2.5020	2.0390	2.5017	0.0002
100,000 samples	2.5003	1.9923	2.5003	0.0000

We can clearly see from Figure 19 and 20, Bayesian estimation peaks near the true value sooner that ML estimation does with increase of drawn samples. That is why for small number of samples Bayesian estimation generates better results. However, for large data sets both methods will estimate true values well. From Tab. 4 we can see that variance for Bayesian estimation converges to zero, while ML variances approaches true variance with N -> inf.

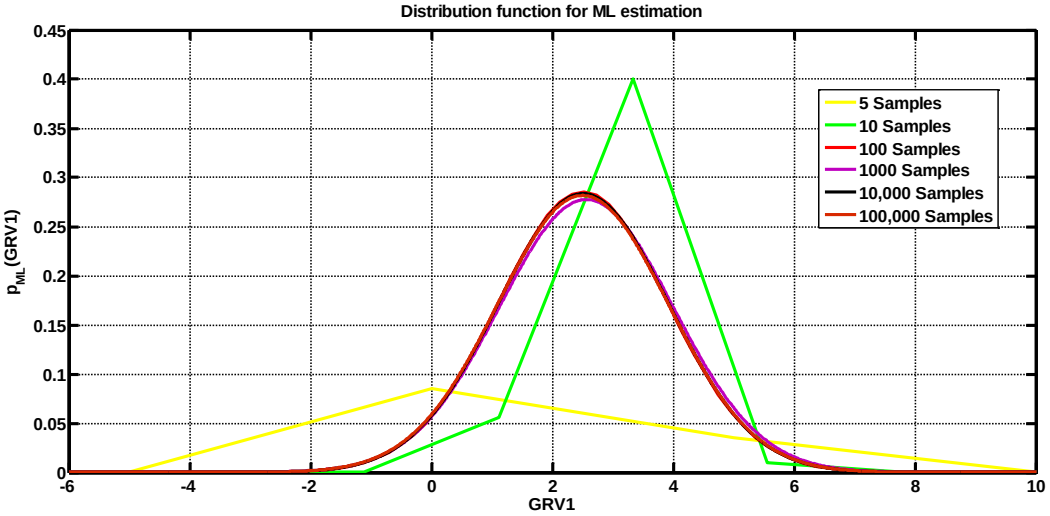


Figure 20. Bayesian Estimation of the Mean and Variance (for the range of 5 to 100,000 samples)

5. Consider the Bayesian estimation of the mean of a one-dimensional Gaussian distribution. Suppose you are given the prior for the mean as $p(\mu) \sim N(\mu_0, \sigma_0)$. Generate 1,000 1D GRVs for GRV[1,1]. Assume $\mu_0 = 0.0$ and $\sigma_0 = 1.0$ (assume the variance is known). Plot the

Bayesian estimate of $p(x|D)$ and μ as a function of the number of data points, n. Do this for 100, 1,000, 10,000 and 100,000 points. Explain your results.

This problem is very similar to the previous problem. Class conditional probability calculates as following:

$$p(x|D) = \int p(x|\mu)p(\mu|D)d\mu$$
$$= \int \frac{1}{\sqrt{2\pi}\sigma}\exp[-\frac{1}{2}(\frac{x-\mu}{\sigma})^2][\frac{1}{\sqrt{2\pi}\sigma_n}\exp[-\frac{1}{2}(\frac{\mu-\mu_n}{\sigma_n})^2]d\mu$$
$$= \frac{1}{2\pi\sigma\sigma_n}\exp[-\frac{1}{2}(\frac{x-\mu_n}{\sigma^2+\sigma_n^2})^2]f(\sigma,\sigma_n)$$

This follows

$$p(\mathrm{x}\,|\,D) \sim N(\mu_n, \sigma^2+\sigma_n^2)$$

We want to calculate the class-conditional density $p(x|D)$, with known parametric form($p(x|\mu) \sim N(\mu, \sigma^2)$ [2]. We also replace μ by μ_n and σ^2 by $\sigma^2+\sigma^2_n$. As a result, μ_n is treated as the true mean, and the known variance is increased to account for the additional uncertainty in x resulting from no knowing the actual mean. Figure 21 presents my results.

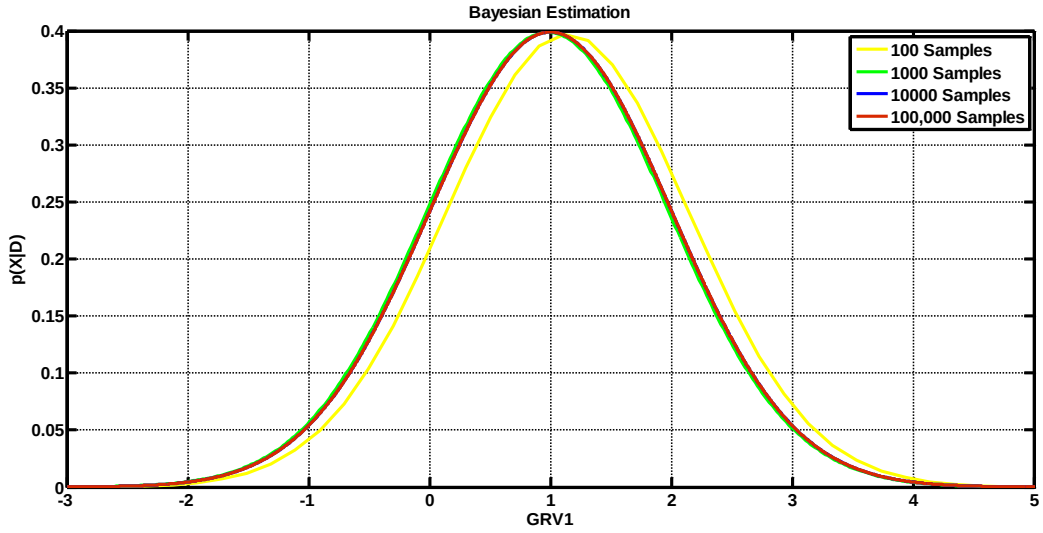


Figure 21. Class conditional density $p(x|D)$

Result for $p(x|D)$ is the desired class-conditional density along with prior probabilities $P(\omega j)$ produces the probabilistic information required to design the classifier[2]. In the other hand, ML only makes estimation for $\mu\hat{}$ and σ^2, and not estimates a distribution for p(x|D). Even for 100 samples, $p(x|D)$ is very close to true value.

Discussion

In this homework the goal was to understand the difference between maximum likelihood estimation and Bayesian estimation. ML estimation is not a very good method to use when the sample size is small because ML estimator becomes biased. ML estimation assumes priors as uniform, and it is not always true. Bayesian estimation improves on those issues and gives better result.

References

[1] C. M. Bishop, "Pattern Recognition and Machine Learning," New York : Springer, pp. , 2006.

[2] R.O. Duda, P.E. Hart, and D. G. Stork, "Pattern Classification," 2nd ed., pp. , New York : Wiley, 2001.

[3] Smith Lindsay I., A tutorial on Principal Components Analysis , web. http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

[4] Java Applet with PCA algorithm visualization http://www.isip.piconepress.com/projects/speech/software/demonstrations/applets/util/pattern_recognition/current/