EECE 5644
Aleksei Krotov

Q1.

1. Without knowing priors or conditionals in a binary classification problem, the probability of error, following a decision (choice) c_i is:
$$p_{err}^{(0)} = p(c_0 \mid w_1) + p(c_1 \mid w_0) = p(c_0)\big(1 - p(w_0)\big) + \big(1 - p(c_0)\big)p(w_0).$$
At any $p(c_0)$ other than 0.5, $p_{err}^{(0)}$ would depend on the priors, but at $p(c_0) = p(c_1) = 0.5$ the error would not depend on the priors and will be equal to 0.5, or 50%, as well. The decision therefore would be choosing any of the two options with equal probability, aka "coin toss".
2. When priors are available, the decision rule would maximize a prior, or equivalently minimizing probability of error: $c = \arg\max(p(w_i)) = \arg\min(p_{err}(w_i))$, where $p_{err}^{(1)} = p(c_0 \mid w_1) + p(c_1 \mid w_0) = \|\| \text{suppose } p(w_1) \geq p(w_2) \Rightarrow p(c_1) = 1 \|\| = 1 \cdot p(w_2)$. Since $p(w_2) \leq p(w_1)$, $p(w_2) \leq 0.5$, therefore $p_{err}^{(1)} \leq 0.5 \leq p_{err}^{(0)}$
3. When both priors and conditional distributions are available, the decision rule would maximizing posterior distribution of the state of the nature conditioned on the observation, or minimizing the respective error: $c = \arg\max \dfrac{p(x \mid w_i)p(w_i)}{p(x)}$. The probability of error is then $p_{err}^{(2)} = \int \min\big[p(w_0 \mid x), p(w_1 \mid x)\big]p(x)dx = \int \min\big[p(x \mid w_0)p(w_0), p(x \mid w_1)p(w_1)\big]dx$. Therefore, this error is bound as following: $p_{err}^{(2)} \leq \min\big[p(w_0), p(w_1)\big] \leq p_{err}^{(1)}$.

Q2A.

1. For a binary decision rule in the form of a likelihood-ratio test, the BDR would be as follows:

$$\begin{cases} L = 1, & \dfrac{p(x \mid L=1)\,p(L=1)}{p(x \mid L=0)\,p(L=0)} \geq 1 \\ L = 0, & \text{otherwise} \end{cases}$$

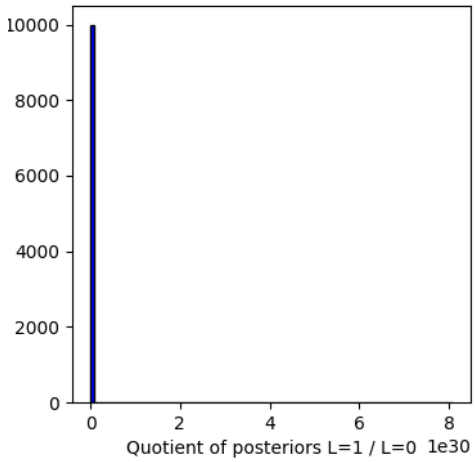
Therefore, the classification could be specified with respect to the ratio of the posterior conditional probabilities, as follows:

$$\begin{cases} L = 1, & \dfrac{p(x \mid L=1)}{p(x \mid L=0)} \geq \dfrac{p(L=0)}{p(L=1)} \\ L = 0, & \text{otherwise} \end{cases}$$, and $\gamma = \dfrac{p(L=0)}{p(L=1)} = \dfrac{0.35}{0.65} \approx 0.58$

For this and the following coding problems, the HW1.py script was used and multiple images were generated, stored in the following Git repository:
The minimum achievable error could then be computed by integrating the expression (Q1.3) using the given priors and conditionals, across all four dimensions of the feature space. Due to computation/accuracy tradeoff, absolute and relative integration tolerances were increased to 1.49e-3 and 1.49e-1 respectively, and the maximum sub-division of an interval for adaptive

curvature was decreased to 20 segments. The computed probability of error was $p(err) = 1.9795 \approx 1.98\%$

2. A binary classifier based on the above decision rule was implemented and evaluated for a wide range of thresholds γ. However, as the threshold in a ML/Bayes decision rule is varied, a ratio of the posterior PDFs may cover wide range of values, potentially complicating selection of an appropriate step size for the threshold. These values were found to range from 1e0 to 1e31, confirming the above presumption.



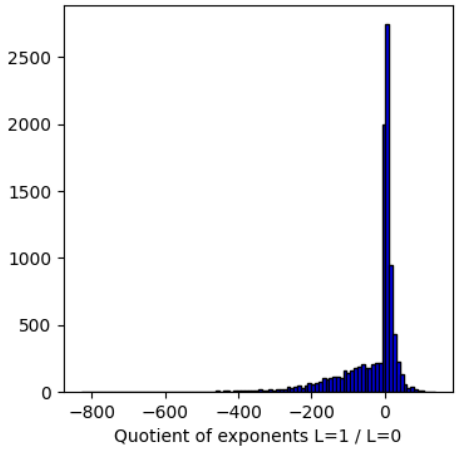
To alleviate that, and to simplify repetitive PDF computations, the ratio of the posterior PDFs was reduced, removing the constant scaling factor, and converted into a logarithmic expression – an equivalent transform given d/dx(log(x)) > 0 for any x>0:

$$\frac{p(x \mid L=1)}{p(x \mid L=0)} = \left(\frac{|\Sigma_0|}{|\Sigma_1|}\right)^{1/2} \cdot \exp\left[\frac{1}{2}\left((\mathbf{x}-\mu_1)^T \Sigma_1^{-1}(\mathbf{x}-\mu_1) - (\mathbf{x}-\mu_0)^T \Sigma_0^{-1}(\mathbf{x}-\mu_0)\right)\right] = \left(\frac{|\Sigma_0|}{|\Sigma_1|}\right)^{1/2} \cdot \exp(\frac{1}{2} \cdot f(\mathbf{x})),$$

$$\ln \frac{p(x \mid L=1)}{p(x \mid L=0)} = \frac{1}{2} \ln\left(\frac{|\Sigma_0|}{|\Sigma_1|}\right) + \frac{1}{2} f(\mathbf{x}),$$

$$\begin{cases} L=1, & \ln \dfrac{p(x \mid L=1)}{p(x \mid L=0)} > 2\ln\left(\dfrac{p(w_0)}{p(w_1)}\right) - \ln\left(\dfrac{|\Sigma_0|}{|\Sigma_1|}\right) \\ L=0, & \textit{otherwise} \end{cases}$$

The ratio of PDFs was thus reduced to a difference of two quadratic forms $f(\mathbf{x})$. That difference, being an exponent, spans a much smaller range of values as shown in the following histogram.

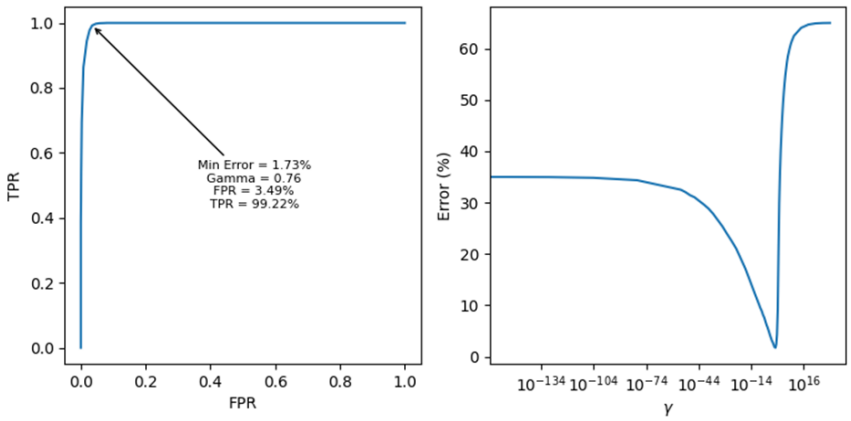


The original threshold γ is hence related to the equivalent threshold $\gamma_{\log}$ for the reduced difference of the two quadratic forms in the following way:

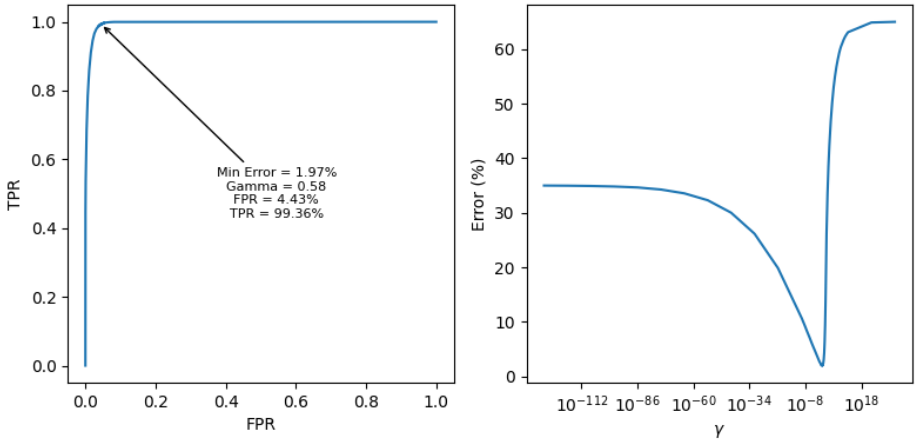
$$\gamma^2 = \frac{|\Sigma_1|}{|\Sigma_0|} \cdot \exp(\gamma_{\log}) \ .$$

The evaluation of the ROC curve was next performed via classification of the prepared dataset, iterated across $\gamma_{\log}$. Five ranges were determined based on minimum, percentile-2.5, percentile-25, percentile-75, percentile-97.5, and maximum values of the difference of the quadratic forms. Step size for $\gamma_{\log}$ was varied across these ranges, such the two outermost ranges were divided into six steps, the central range was divided into 30 steps, and the two transient ranges were divided into 18 steps.

The resulting ROC curve and the plot, relating error probability with the original threshold value are shown below. The minimum error was 1.73% attained at $\gamma = 0.76$, resulting in FPR = 3.49% and TPR = 99.22%, and the AUC is 0.9943. The error and threshold values deviate from the theoretically determined ones which is likely caused by a relatively small sample set across four dimensions.



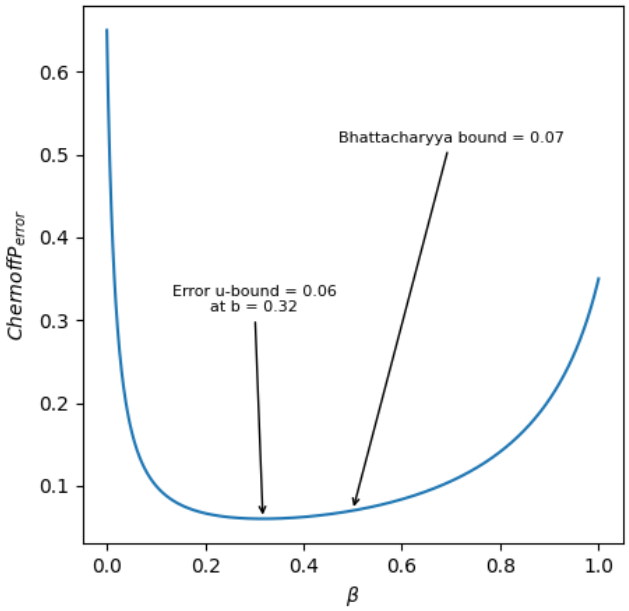
To test that, the same procedure was repeated for N = 1e6, and the estimates became closer to the theoretical ones: The minimum error was 1.97%, attained at $\gamma = 0.58$, resulting in FPR = 4.43% and TPR = 99.36%



Q2B.

The Chernoff and Bhattacharyya error bounds were next evaluated. The first one was determined as a numerical minimum over N=1000 samples of $P_{error}(\beta)$ calculated as

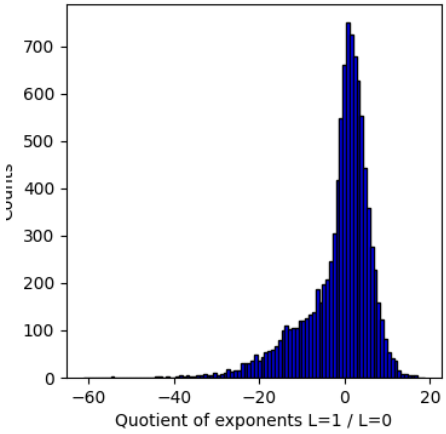
$Chernoff_{Error} = p^{\beta}(w_0)p^{1-\beta}(w_1)\cdot e^{-k(\beta)}$, where, using that both PDFs are Gaussian, $k(\beta) = \frac{\beta(1-\beta)}{2}(\mu_1-\mu_0)^T\left[\beta\Sigma_0+(1-\beta)\Sigma_1\right]^{-1}(\mu_1-\mu_0)+\frac{1}{2}\ln\frac{|\beta\Sigma_0+(1-\beta)\Sigma_1|}{|\Sigma_1|^{\beta}|\Sigma_2|^{1-\beta}}$. The obtained

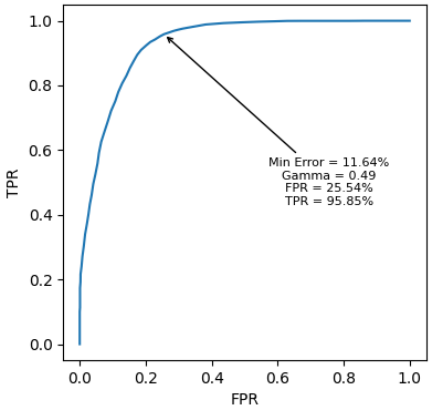


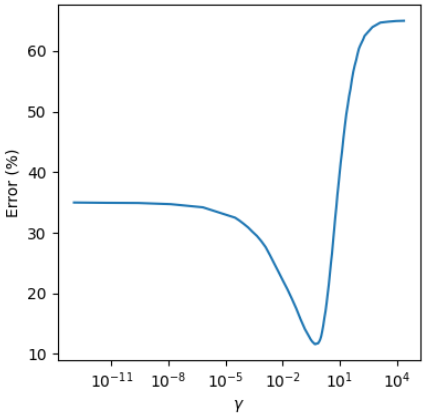
bound is showing below. It indicates the upper bound for the classification error at 6%, obtained at $\beta = 0.32$, as well as a softer Bhattacharyya bound at 7%, obtained at $\beta = 0.5$.

Q2C.

With the covariance matrices stripped of their non-diagonal elements, the classifier was implemented in a similar way. The reduced exponent difference of the two quadratic forms spanned a smaller range, so percentile-based tessellation of the range for $\gamma_{\log}$ differed, from that in Part A. Obtained empirical minimum error became 11.64% at $\gamma = 0.49$, FPR = 25.54%, TPR = 95.85%. Computational time of theoretical minimum error exceeded 20 minutes, and computation was aborted. The model mismatch, i.e. assumption of isotropic distribution, increased the error drastically, however the γ value, although noisy due to relatively small sample size, is still within 25% of the theoretical one for the known, different, and anisotropic covariances.



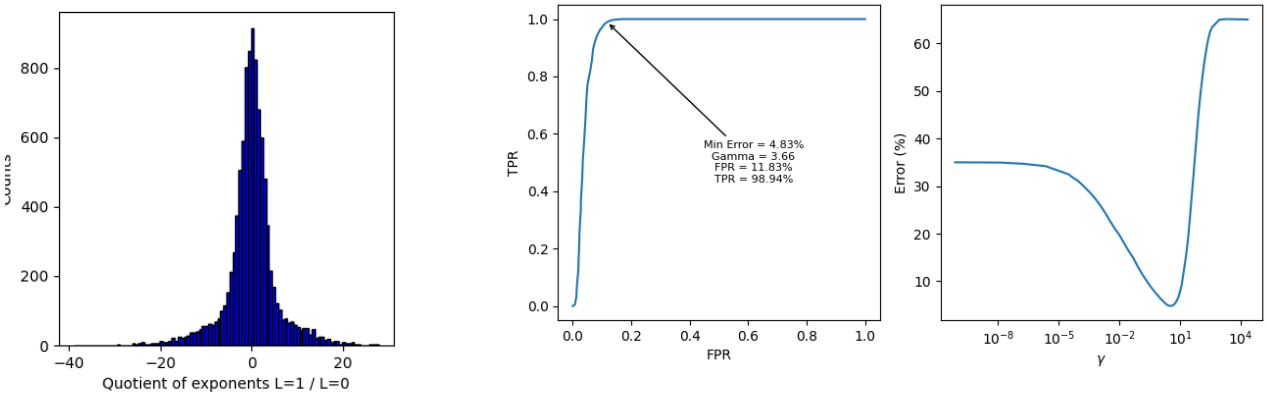




Q2D.

The assumption of common covariance results in a linear classification surface, being a hyper-plane. That plane is parallel to the direction of the largest eigenvalue of the common covariance matrix and is shifted from the middle of the line connecting the means by a factor determined by the priors ratio.

The range of the difference of the quadratic forms decreased, compared to full-covariance model, and the respective distribution became visually normal. The minimum error became 4.83% at γ =3.66, FPR = 71.26 and TPR = 98.58. Such performance of the classifier now became drastically worse again, the threshold value is still within 25% of the theoretical one, and the profile of the ROC curve changed to the S-shaped one.



The less accurately the distributions of classes in a dataset are estimated, the higher the classification error, and the more downward the ROC curve is pulled.

Q2E.

Addition of costs, associated with erroneous classification would result in a shift of the threshold γ, resulting in a following decision rule:

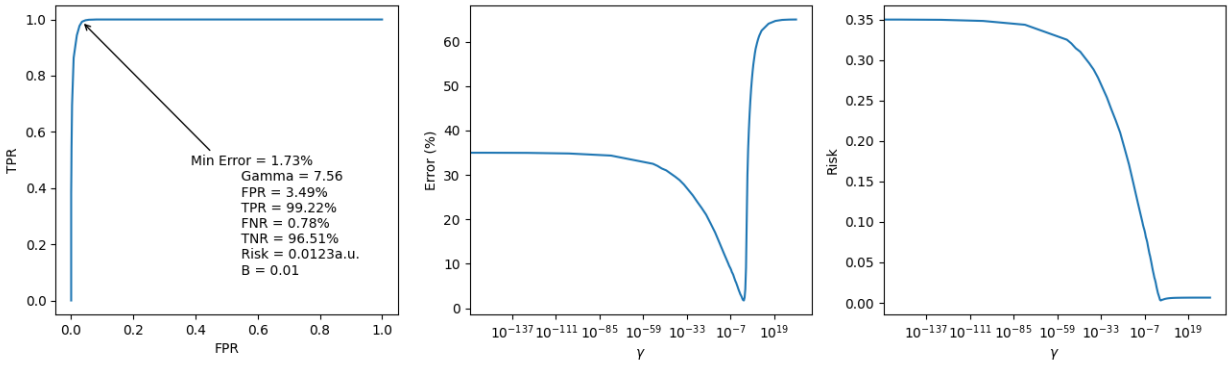
$$\begin{cases} L=1, & \ln\frac{p(x\mid L=1)}{p(x\mid L=0)} > 2\ln\left(\frac{p(w_0)}{p(w_1)}\right) - \ln\left(\frac{|\Sigma_0|}{|\Sigma_1|}\right) + \ln(r), \\ L=0, & \textit{otherwise} \end{cases}$$

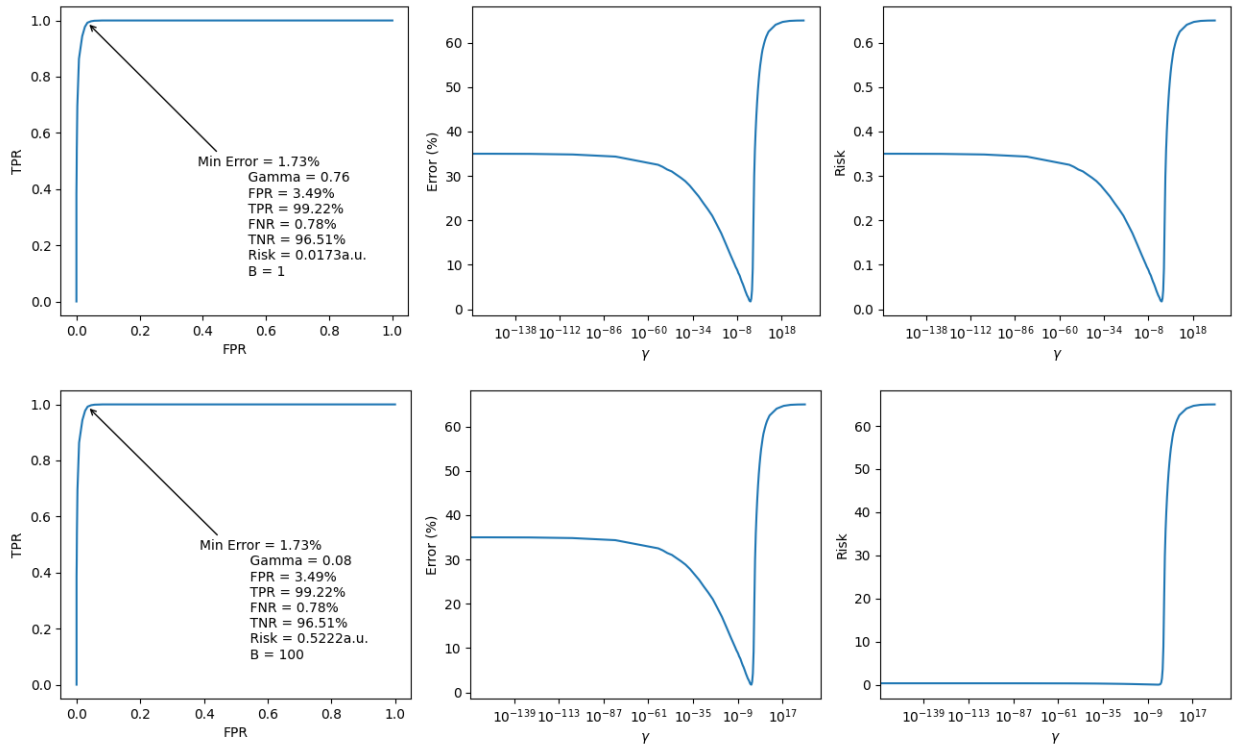
where $r = \frac{\lambda_{21}-\lambda_{11}}{\lambda_{12}-\lambda_{22}} = \frac{1}{B}$ and λ_{ij} are elements of the risk matrix R.

The implementation maintained the same procedure of finding of the dataset-based distribution of the logarithm-scaled LHS and iterating over the $\gamma_{\log}$, as in the previous parts. The risk factor was added to computation of the original γ:

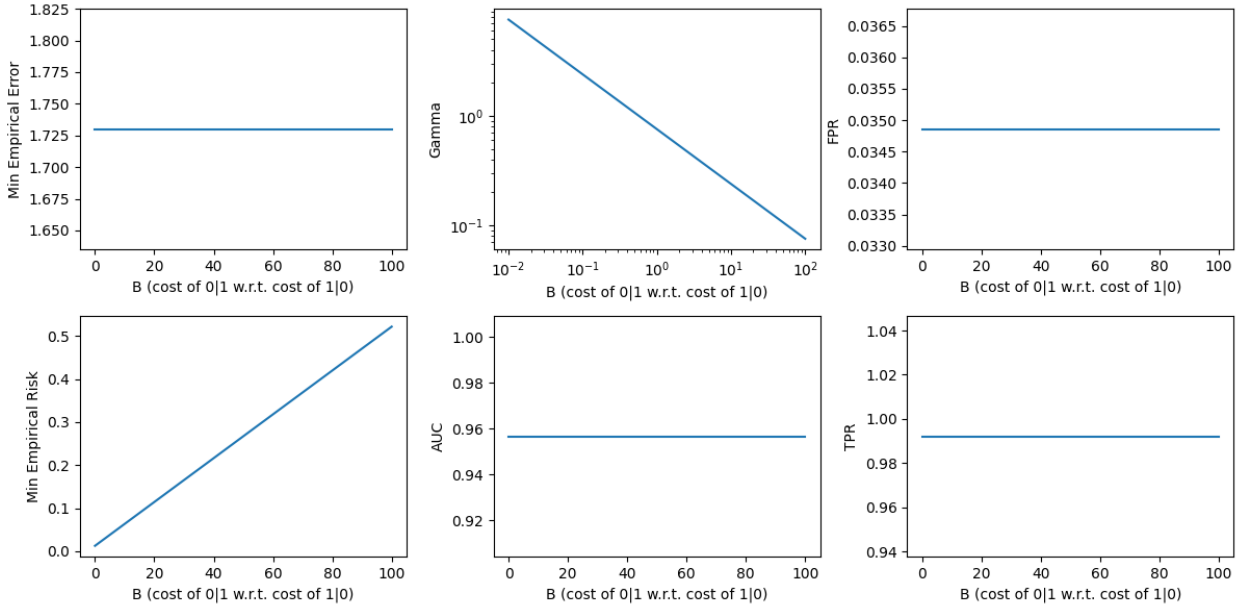
$$\ln\gamma^2\frac{\Sigma_1}{\Sigma_2} = \gamma_{\log} \rightarrow \gamma = \sqrt{\exp(\gamma_{\log})\cdot\frac{|\Sigma_0|}{|\Sigma_1|}\cdot\frac{1}{r}}$$

20 B values were examined in the range from 0.01 to 100. Examples of the ROC curve, the Error(gamma) curve, and the Risk(gamma) curve are shown below for B = {0.01, 1, 100}.





Finally, the last figure shows relation of classifier parameters with the B value.



Overall, non-equally weighting the costs of incorrect labeling maintained minimum achievable error, the ROC curve of the classifier and the operating point on the ROC curve minimizing orevall error. However, the threshold γ, defined with respect to the ratio of the posterior probabilities (Part 2A), decreased exponentially with increase of B, shifting the classification surface towards the class L=1 as the risk of incorrectly predicting that class increased. That can be seen also from the

shape of the risk vs. γ (right column) in the three rows of B = {0.01, 1, 100}. The minimum empirical risk scaled linearly with B.

Q3. Wine dataset.

The samples were assumed to be independent from each other, generally balanced across classes and features, distribution of features was assumed to be multivariate Gaussian, its parameters were estimated for each class from the sample set.

The manually implemented classifier might be incorrectly configured (still could not find a bug), as it was verified with "sklearn.naive_bayes.GaussianNB" and they yielded different classification outcomes.

Using both red and white wines and assigning 0 and 1 to these categories, the manually set up classifier yielded error 68.8% while the "GaussianNR" yielded error 60.0% with the two following confusion matrices.

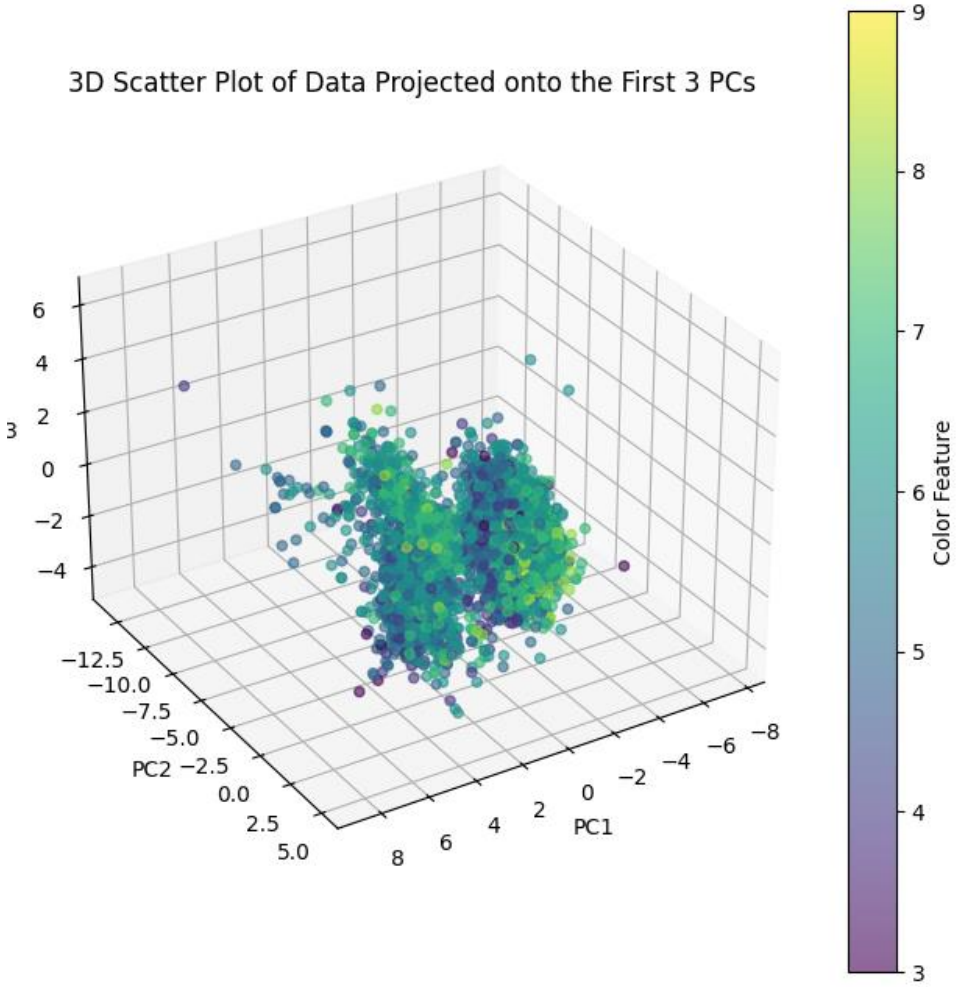
Table 1. Confusion matrix of manually set up, via multivariate_normal.pdf() classifier

			True Quality						
			3	4	5	6	7	8	9
			11	5	20	2873	3469	119	0
True Quality	3	30	4	1	0	20	5	0	0
	4	216	2	0	1	132	80	1	0
	5	2138	2	2	13	1370	726	25	0
	6	2836	3	2	3	1131	1643	54	0
	7	1079	0	0	3	186	865	25	0
	8	193	0	0	0	34	146	13	0
	9	5	0	0	0	0	4	1	0

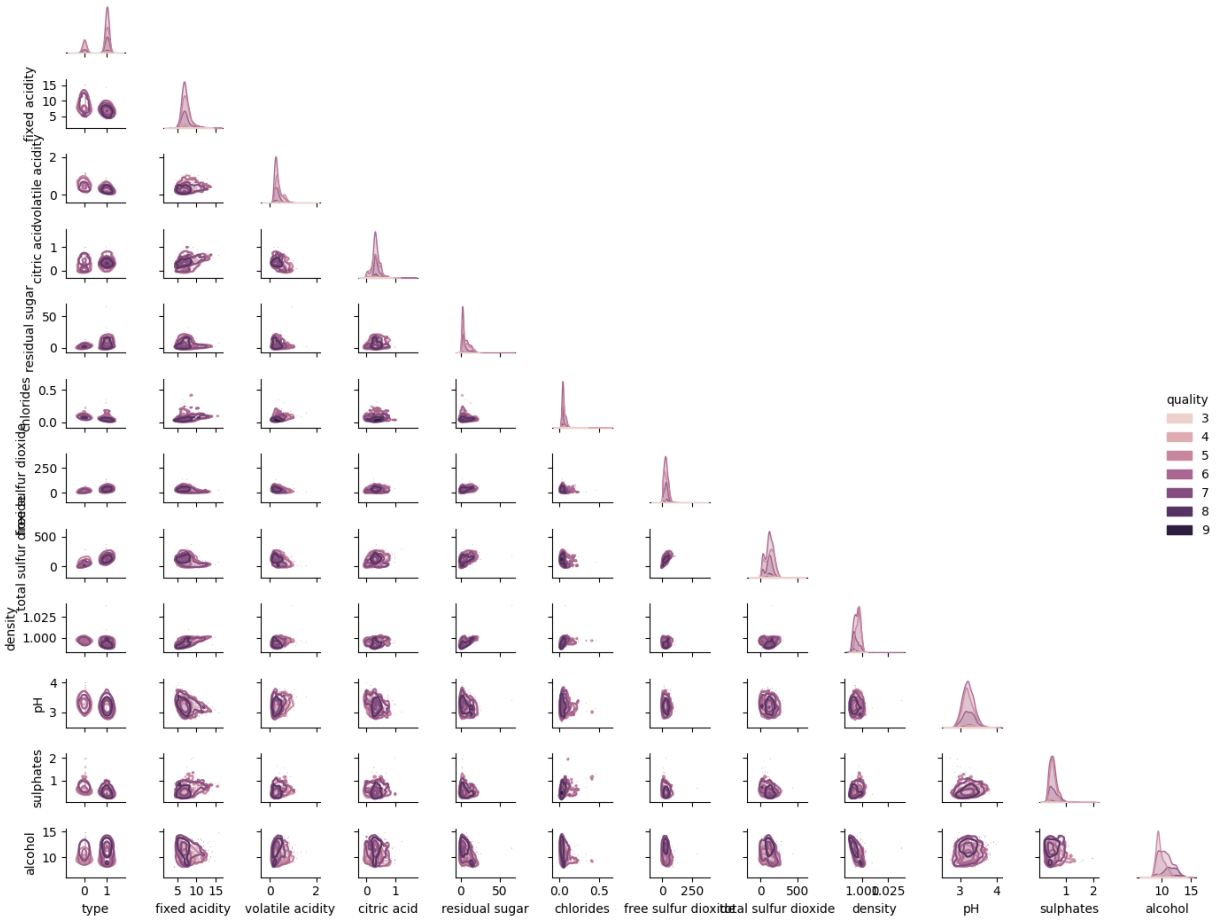
Table 2. Confusion matrix of sklearn.naive_bayes.GaussianNB

			True Quality						
			3	4	5	6	7	8	9
			59	159	1965	2489	1085	67	673
True Quality	3	30	7	4	8	7	2	0	2
	4	216	4	26	83	74	25	1	3
	5	2138	27	55	1045	834	133	2	42
	6	2836	16	60	715	1200	548	18	279
	7	1079	5	13	105	321	314	30	291
	8	193	0	1	9	52	63	16	52
	9	5	0	0	0	1	0	0	4

For the first visualization of the dataset, three PCs with largest VAF were used yielding the following distribution. The quality does not appear to cluster with respect to these three PCs, while there arose two clusters – likely corresponding to red and white wine types.



The following pairplot indicates that the sample set is unbalanced with respect to the classes, handling of the binary “type” feature may potentially affect classification performance (and it did for both classifiers, e.g. using numbers with a larger difference reduced error). It also suggests that Gaussian assumption of the within-class and between-feature distributions may not be appropriate for this dataset.



The same classifiers were trained and tested on only-red and only-white wine datasets. For the only-red-wine dataset, the classifiers showed smaller error, 53.0 and 43.5% respectively, and for the only-white-wine-dataset , the classifiers showed errors 69.3% and 65.0% respectively.

The human activity dataset has escaped my understanding and exceeded the ~20 debugging hours already spent on this homework, on a dark Sunday night.