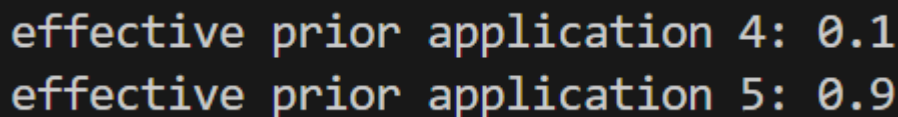


LAB07 REPORT

- (0.5, 1.0, 1.0), i.e., uniform prior and costs
- (0.9, 1.0, 1.0), i.e., the prior probability of a genuine sample is higher (in our application, most users are legit)
- (0.1, 1.0, 1.0), i.e., the prior probability of a fake sample is higher (in our application, most users are impostors)
- (0.5, 1.0, 9.0), i.e., the prior is uniform (same probability of a legit and fake sample), but the cost of accepting a fake image is larger (granting access to an impostor has a higher cost than labeling as impostor a legit user - we aim for strong security)
- (0.5, 9.0, 1.0), i.e., the prior is uniform (same probability of a legit and fake sample), but the cost of rejecting a legit image is larger (granting access to an impostor has a lower cost than labeling a legit user as impostor - we aim for ease of use for legit users)

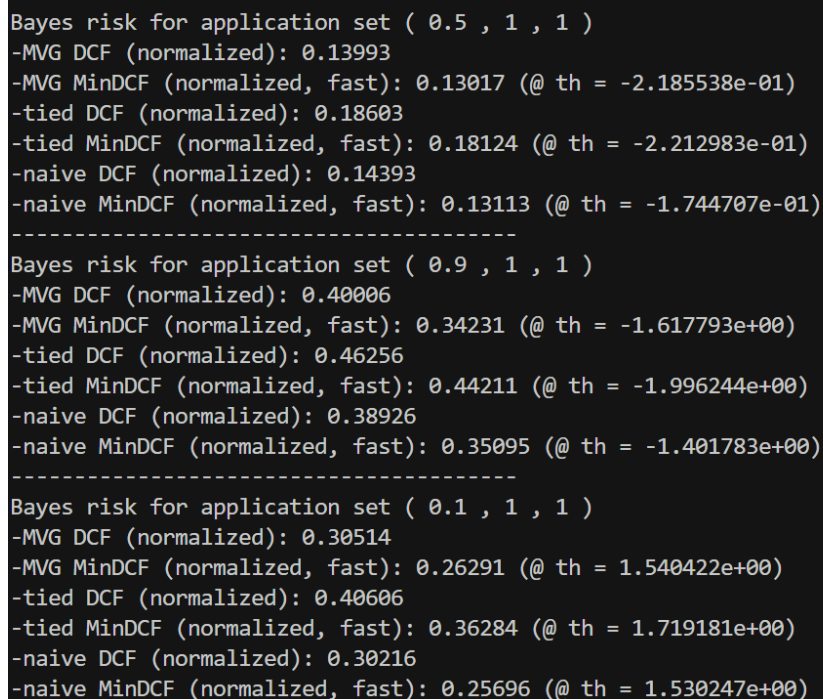
First of all, we are given these 5 different applications and we are asked to represent the last 2 applications in terms of effective prior. The effective prior is gonna “absorb” the cost for the misclassification



```
effective prior application 4: 0.1
effective prior application 5: 0.9
```

As we can see, the application 4 had a great cost for a false positive, this is reflected by a prior that is more keen towards the negative class! Same reasoning in reverse for the application 5.

We now focus on the first 3 applications, no PCA first



```
-----
Bayes risk for application set ( 0.5 , 1 , 1 )
-MVG DCF (normalized): 0.13993
-MVG MinDCF (normalized, fast): 0.13017 (@ th = -2.185538e-01)
-tied DCF (normalized): 0.18603
-tied MinDCF (normalized, fast): 0.18124 (@ th = -2.212983e-01)
-naive DCF (normalized): 0.14393
-naive MinDCF (normalized, fast): 0.13113 (@ th = -1.744707e-01)
-----
Bayes risk for application set ( 0.9 , 1 , 1 )
-MVG DCF (normalized): 0.40006
-MVG MinDCF (normalized, fast): 0.34231 (@ th = -1.617793e+00)
-tied DCF (normalized): 0.46256
-tied MinDCF (normalized, fast): 0.44211 (@ th = -1.996244e+00)
-naive DCF (normalized): 0.38926
-naive MinDCF (normalized, fast): 0.35095 (@ th = -1.401783e+00)
-----
Bayes risk for application set ( 0.1 , 1 , 1 )
-MVG DCF (normalized): 0.30514
-MVG MinDCF (normalized, fast): 0.26291 (@ th = 1.540422e+00)
-tied DCF (normalized): 0.40606
-tied MinDCF (normalized, fast): 0.36284 (@ th = 1.719181e+00)
-naive DCF (normalized): 0.30216
-naive MinDCF (normalized, fast): 0.25696 (@ th = 1.530247e+00)
```

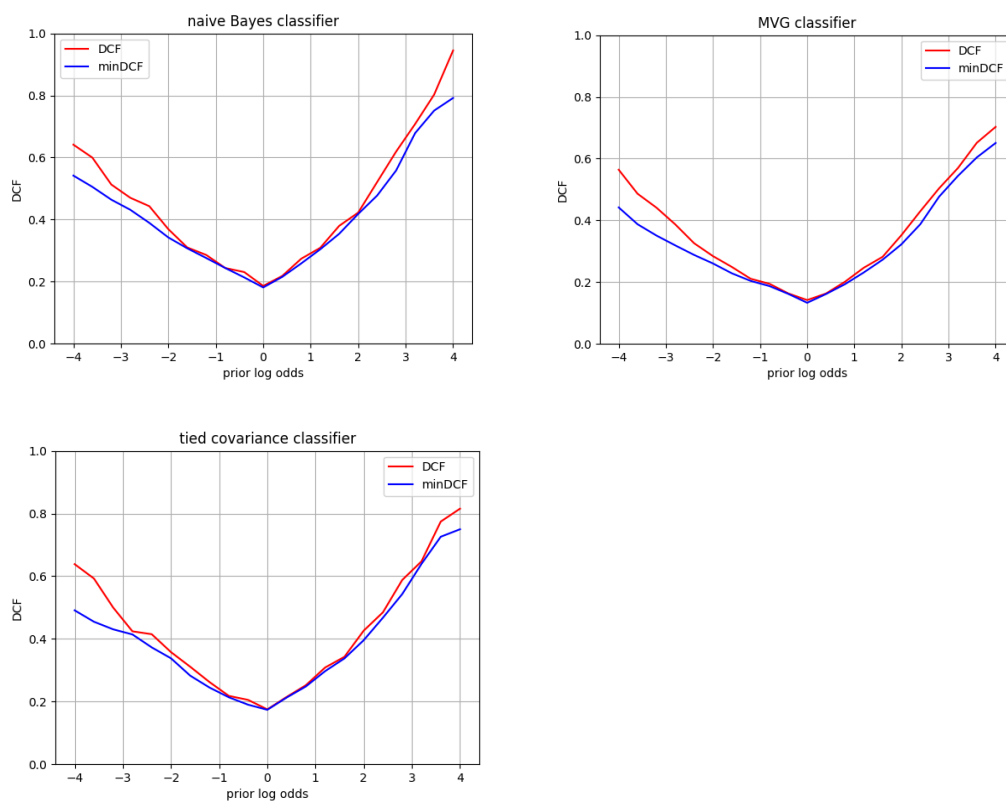
The minDCF does not vary much for the different models in an application. It is interesting to see how the MVG and Naive Bayes DCF are actually very similar, with some difference with the tied model. The Naive Bayes actually has the lowest minDCF. The actual DCF, although higher, shows similar behaviors. The best model in terms of DCF is the first one (the one with prior 0.5) this is probably due to the effective prior being 0.5, there is not much penalty in choosing a class over another.

The model shows not very good calibration, the DCF can be improved by choosing a different threshold.

Now we analyze the model behavior with PCA preprocessing (all the DCFs are visible in the python project, i did not have the space to insert the image).

The PCA actually helps reduce the model DCF.

The PCA setup that gives us the best result is the one with $m=5$, and these are the Bayes error plots.



We can see here graphically what we saw before: the minDCF varies for the different applications. We can see the MVG has the lowest minDCF, the model is well calibrated in the range $-2/2$ but could be improved in the $-4/2$ $2/4$ range. We could calibrate the score to improve the actDCF.

Nonetheless, the various classifiers show similar behavior as the prior of the application change, and the classifier does not show excessive calibration error (although in some applications they could benefit from a calibration).