Fingerprint spoofing detection

Giovanni Bordero s313010 Francesco Dente s313355

July 16, 2023

1 Introduction

The goal of the project is the development of a classifier to detect whether a fingerprint image represents an authentic fingerprint or a spoofed fingerprint, i.e. a fingerprint image obtained by cloning, possibly with different methods, the fingerprint of an individual. The fingerprint images are represented by means of embeddings, i.e. low-dimensional representations of images obtained by mapping images to a low-dimensional manifold (typically few hundred dimensions).The datasets are imbalanced, with the spoofed fingerprint class having significantly more samples, the training set contains 2325 samples (1525 spoofed, 800 authentic). The spoofed fingerprint samples belong to one of 6 possible sub-classes, corresponding to different spoofing approaches, but the actual spoofing method (sub-class label) is not available. The target application considers an application where prior probabilities of authentic and spoofed classes are the same, but labeling a spoofed fingerprint as authentic has a larger cost due to the security vulnerabilities that such errors would create.
The target application working point is therefore defined by the triplet $(\pi_T = 0.5, C_{fn} = 1, C_{fp} = 10)$

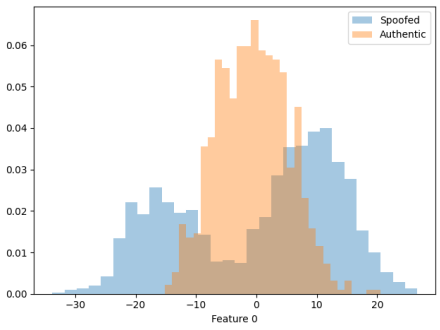
1.1 Source-code

The main ML functions developed during the laboratories, and used for the project are in *utils.py*. The other files were used for writing scripts in order to make all the needed analysis.

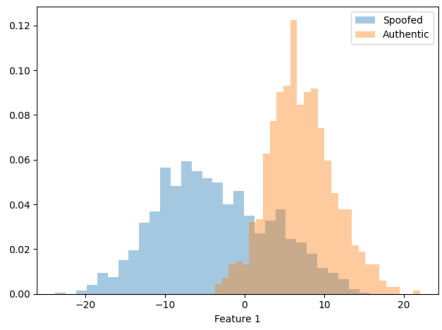
2 Features analysis

2.1 Data visualization

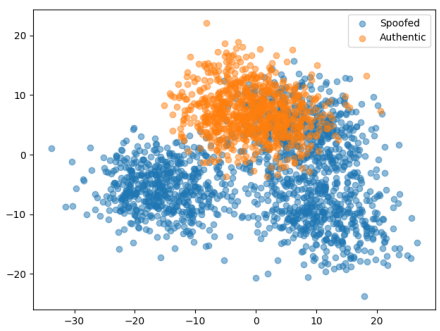
Assuming that PCA allows us to preserve as much as possible relationships between different samples, after projecting our samples on the 2 principal components we can visualize our data. Then, if we exploit also LDA, we can plot the histogram for our samples projected on the most discriminant LDA direction.



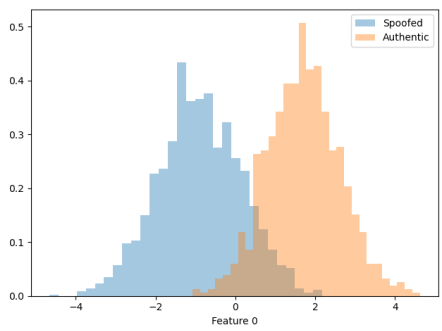
(a) Histogram for feature 0 after PCA-2



(b) Histogram for feature 1 after PCA-2



(c) 2D scatter plot of features after PCA-2

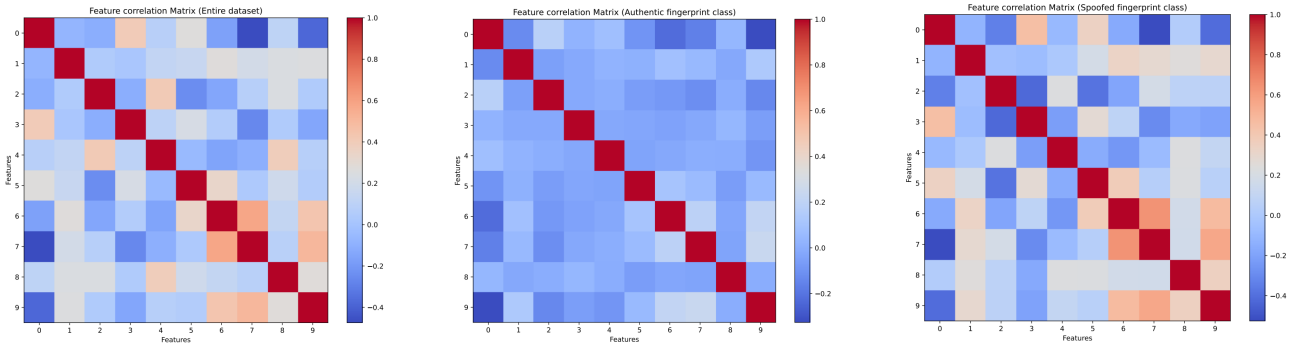


(d) Histogram for LDA direction

We can see that the *spoofed* class is characterized by multiple clusters, thus it could be difficult to model it using Gaussian densities, on the contrary a Gaussian distribution appears to be suitable to model the *authentic* class.
Looking at the histogram obtained after projection on the main LDA direction, we may think that a linear classifier could be able to discriminate classes to some degree, but, given the distribution of the features we saw in the scatter plots, we expect non-Gaussian and/or non-linear models (e.g. MVG) to be significantly better for the task.

2.2 Features correlation analysis

We can then proceed and analyze the correlation between features in the dataset (Pearson correlation coefficient), using heatmaps to visualize them. We can see that for authentic class features are weakly correlated,

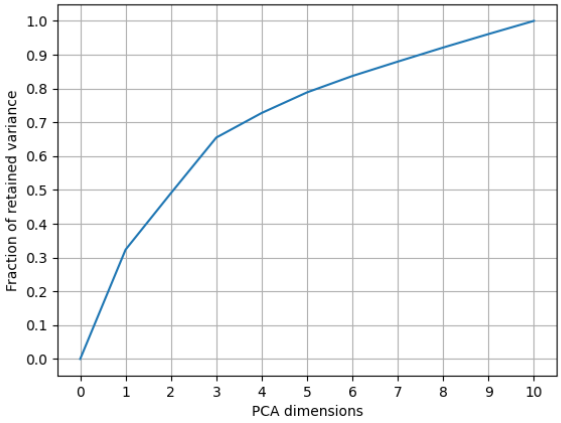


(a) Heatmap for the entire dataset (b) Heatmap for authentic class (c) Heatmap for spoofed class

while for spoofed, we have larger correlations (especially between features 6 and 7). Thus, we could use PCA to remove correlations between features regarding the spoofed class, but we must be aware that we could throw away some useful information regarding the authentic class (in which there are not correlations between features, thus we may think that they all carry discriminant information).

2.3 PCA-explained variance

To decide which values for PCA could be worth considering in our analysis, we further look at the plot of Fraction of explained variance VS PCA dimensions We can see that with PCA-9 we explain about 96% of the



dataset variance, 92% with PCA-8, 88% with PCA-7. We therefore decide to start by considering *NO-PCA, PCA-9, PCA-8, PCA-7* in our analysis.

3 Training Protocol

We adopt a K-Fold cross validation with K = 5, we will measure models performance in terms of minimum DCF. The best way to evaluate performance would be using actual DCF after calibration for each model, but it would complicate things much more, so we will use minDCF and assume that eventually the scores can be calibrated well so that actual DCF is close to min DCF. The working point of our target application is $(\pi_t, C_{fn}, C_{fp}) = (0.5, 1, 10)$, that we will summarize in an effective prior $\tilde{\pi} = 0.091$(working point = $(0.091, 1, 1)$). We will train models to optimize our target application, but performance of models on two alternative applications will also be analyzed.
Alternative working points:

- $(0.5, 1, 5) = (0.167, 1, 1)$
- $(0.5, 1, 20) = (0.0476, 1, 1)$ We chose to analyze two alternative applications with different costs of letting a spoofed fingerprint in.

4 Gaussian Classifier

According to what we saw during Features Analysis, the two classes have different covariance matrices, and appear not to be easily linearly separable. Therefore, we expect a Tied covariance model to perform poorly, while Naive Bayes and MVG could be more suitable for our task.

- **MVG:** overall, since the proportion $\frac{N_c}{D}$ (N_c = samples of class c, D = dimensionality of samples), is not particularly low for each of the two classes, we expect that MVG may provide good estimates of the two covariance matrices, thus it appears to be the most suitable approach.
- **Naive Bayes:** If the feature independence assumption is met, then Naive Bayes can help to obtain a more reliable estimate of the covariance matrices and avoid overfitting, since it needs to estimate $O(D)$ independent elements for each one. PCA may help in improving Naive Bayes performance. This could be useful for the minority class (authentic fingerprint), of which we have fewer samples.

We test anyway all the three approaches, along with PCA:

MVG classifier — min DCF

PCA	**MinDCF**($\tilde{\pi} = 0.091$)	**MinDCF**($\tilde{\pi} = 0.048$)	**MinDCF**($\tilde{\pi} = 0.167$)
-	0.332	0.444	0.266
9	0.331	0.447	0.268
8	0.337	**0.424**	**0.266**
7	**0.330**	0.434	0.267
6	0.335	0.440	0.268

7 PCA is the most effective, even if 9 and no-PCA are very close.

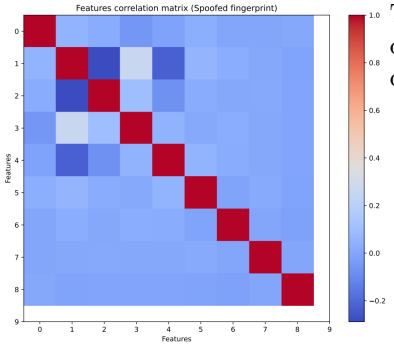
Tied MVG classifier — min DCF

PCA	**MinDCF**($\tilde{\pi} = 0.091$)	**MinDCF**($\tilde{\pi} = 0.048$)	**MinDCF**($\tilde{\pi} = 0.167$)
-	0.491	**0.556**	0.406
9	**0.480**	0.572	**0.394**
8	0.493	0.578	0.400
7	0.484	0.582	0.396
6	0.491	0.582	0.399

The linear model doesn't provide good results and even the PCA does not bring improvements.

Naive Gaussian classifier — min DCF

PCA	MinDCF($\tilde{\pi} = 0.091$)	MinDCF($\tilde{\pi} = 0.048$)	MinDCF($\tilde{\pi} = 0.167$)
-	0.467	0.598	0.353
9	**0.392**	0.489	0.304
8	0.393	**0.489**	0.303
7	0.398	0.490	**0.296**
6	0.400	0.498	0.301



The naive assumption is more effective after PCA, the improvement could be due to the removal of a correlation between features (especially in the spoofed class). This can be seen in the heatmap of the class spoofed, after PCA-9

Figure 3: Spoofed class heatmap after PCA-9

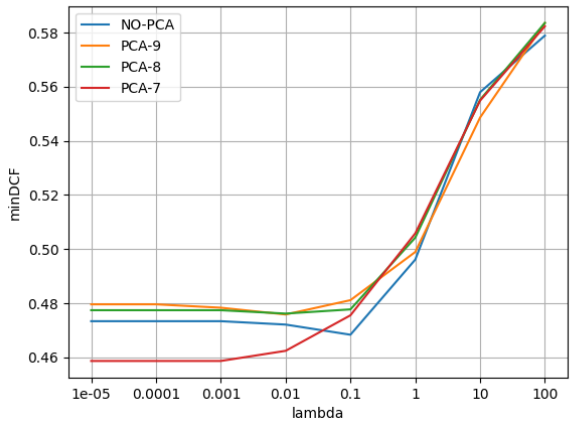
4.1 Conclusions

We conclude that MVG is the most appropriate model for our target application because even if the training set is unbalanced, with the authentic class having fewer samples than the spoofed class, there are still enough samples from both classes to obtain a reliable estimate of the covariance matrix.
We also see that dimensionality reduction techniques bring slight improvements to our MVG model, so we conclude that the best choice would be **PCA-7 + MVG**. Since we saw that PCA-7 gave some improvements, we also tried PCA-6, which gave worse results, so we stopped using PCA there.

5 Logistic Regression classifier

5.1 Linear Logistic Regression

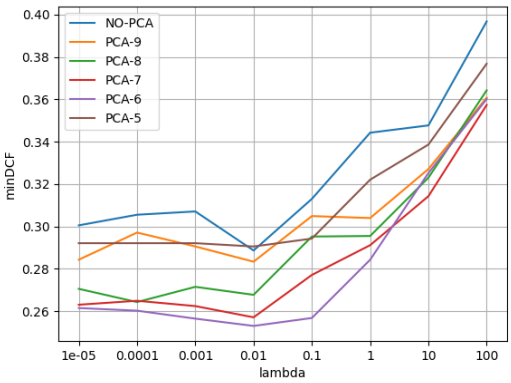
Based on what we saw during feature analysis and results obtained with Gaussian classifiers, we expect the linear model to perform poorly, similar to what we saw with the Tied Covariance model, which also uses a linear decision rule for classification. We train a prior-weighted Linear-Log-Reg model using the $\tilde{\pi}$ of our target application.

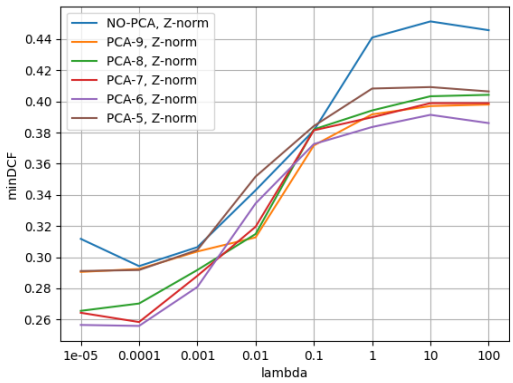


We can see that PCA-7, like in Gaussian models, slightly improves the results, but overall the minDCF values are worse than those obtained with non-linear models such as MVG. We therefore need to resort to quadratic logistic regression models.

5.2 Quadratic Logistic Regression classifier

Since the dimensionality of our feature space is not significantly large, we can afford to project our samples into an expanded feature space (applied after PCA) in order to obtain quadratic separation surfaces in the original space. We analyze the behavior of our model with different values of m for PCA, and with/without z-normalization as a preprocessing technique (applied after PCA). We train a prior-weighted Q-Log-Reg model using the $\tilde{\pi} \approx 0.091$ of our target application.





We can see that results are significantly better, even better than those obtained with MVG. This is consistent with the fact the the spoofed class appeared to be hard to approximate with a gaussian density. Since we saw that PCA was improving the performance, we analyzed our model for PCA up to PCA-5, where performance stopped improving and started decreasing.
We see also that z-norm does not bring any improvement to our model, therefore we can assess that the best model is **Q-Log-Reg + PCA-6** with $\lambda = 0.1$. Even if $\lambda = 0.01$ provides a slightly better performance, we choose an higher value of λ ($\lambda = 0.1$), that may generalize better on unseen data (less risk of overfitting).

Q-Log-Reg (PCA-6, $\lambda = 0.1$) — min DCF

Embedded prior	**MinDCF**($\tilde{\pi} = 0.0909$)	**MinDCF**($\tilde{\pi} = 0.0476$)	**MinDCF**($\tilde{\pi} = 0.167$)
0.091	0.257	0.349	0.207

We can also analyze the behaviour of our best model when embedding a prior different from the one of our target application.

Q-Log-Reg (PCA-6, $\lambda = 0.1$) — min DCF

Embedded prior	**MinDCF**($\tilde{\pi} = 0.091$)	**MinDCF**($\tilde{\pi} = 0.048$)	**MinDCF**($\tilde{\pi} = 0.167$)
0.048	0.260	0.358	0.209
0.167	0.271	0.362	0.203
0.2	0.274	0.373	0.203
0.344 (Dataset)	0.283	0.382	0.208

We can see that the cases with a prior matched to the target application give the best performance, except for the working point (0.048, 1, 1), which gives the best results when using a prior equal to 0.091. Besides this, we can finally conclude that using a matched prior to train our prior-weighted model seems to be the best option for our target application.

5.3 Conclusions

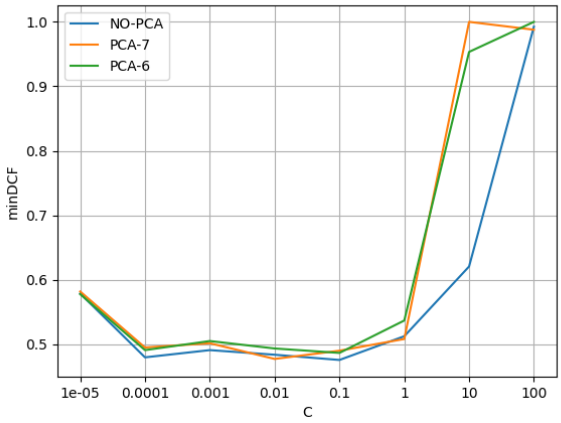
According to what we said, we conclude that the best choice for our Logistic Regression model is a **Quadratic Logistic Regression** model with **PCA-6** and $\lambda = 0.1$, prior-weighted with a prior $\pi_T = 0.091$.

6 Support Vector Machine

We now see SVM models, both linear and non-linear, in our implementation of SVMs we will be regularizing also the bias term b, the hyperparameter k which controls the regularization of the bias term will be set to 1.

6.1 Linear SVM

We take a quick look at linear SVM models, even though we expect the results to be poor. We analyze the behavior of the model for different values of PCA, after rebalancing the classes according to our target application. We look only at PCA-7 and PCA-6, which gave the best results with the previous models.



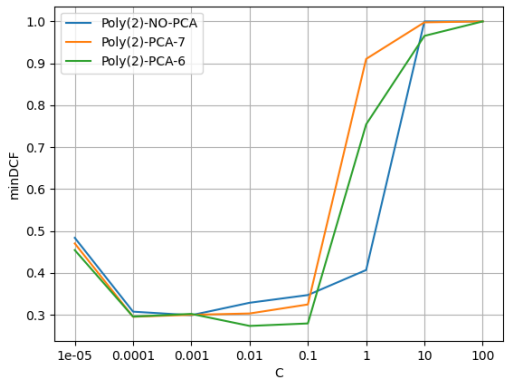
As expected, we have poor results for our task using linear SVM, similarly to what we saw using linear logistic regression, thus we don't further analyze the use of pre-processing techniques such as z-norm.

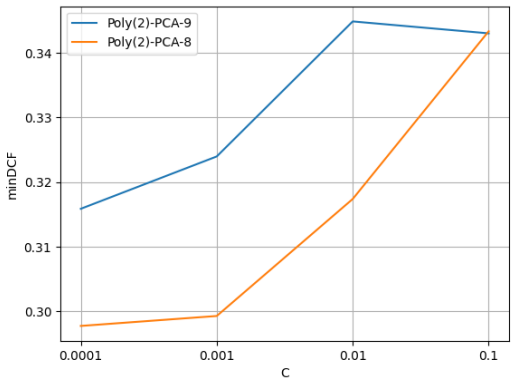
6.2 Kernel SVM

We thus need to resort to kernel SVMs, that allow us to perform classification in an expanded feature space. In this way, we are computing a linear separation surface in the expanded space, which corresponds to a non-linear separation surface in the original space. In contrast with Quadratic logistic regression, we can train a SVM in a large (even infinite) dimensional Hilbert space, without requiring explicitly to compute the mapping, thus we can afford high-dimensional expansions that may lead to better performances.

6.2.1 Kernel Poly

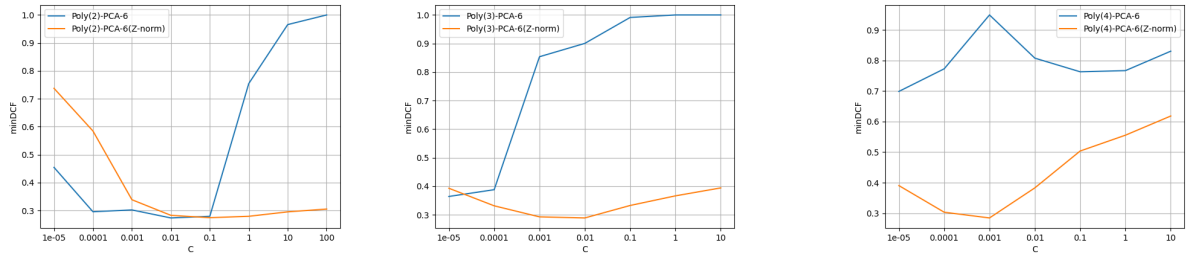
We now analyze the behaviour of the model with different polynomial kernels, we also re-balance classes because, as we have seen in Q-LogReg, this may lead to improvements for our model.
We fix the constant term c to 1, and we firstly analyze polynomial kernel with degree $d = 2$ looking at models without PCA, and with PCA-7/PCA-6 (those who gave best results for respectively MVG and Q-LogReg).





We can see that kernel poly with degree 2 significantly improves the result with respect with linear SVM, but it's slightly worse than Q-LogReg. We complete our analysis by looking at the other 2 values of PCA (PCA-9 and PCA-8) trying only C values around the supposed best value.
Eventually we can assess that the best results are obtained using PCA-6, like in Q-LogReg.

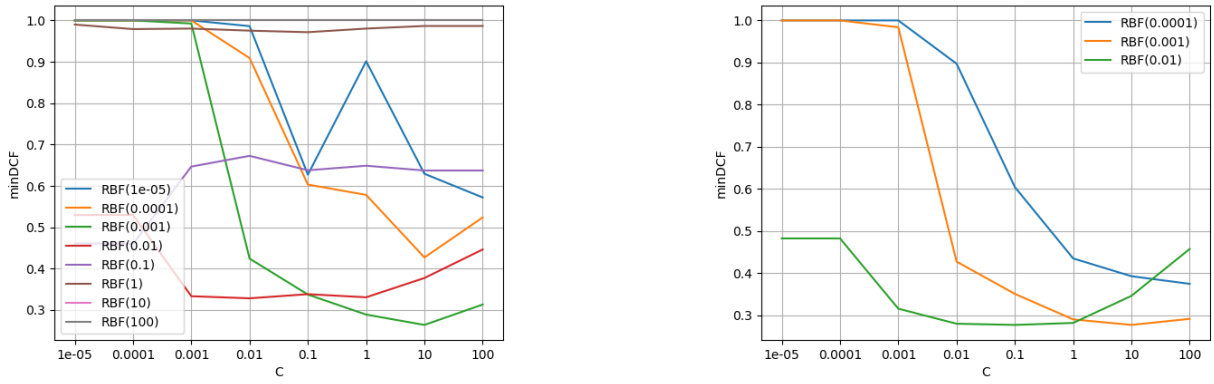
We now see the effect of z-normalization on our model, we try also degree 3 and 4 polynomial kernels, analyzing only PCA-6.



Z-norm improves our model when using polynomial kernel with degree 3 and 4, obtaining almost the same performance as with polynomial kernel with degree 2.
In conclusion, the best model regarding poly kernel is **Poly(2) with C=0.01 + PCA-6 without z-normalization**(minDCF=0.273) Even if models with d=3 and d=4 + z-normalization have similar performance to d=2, we choose a polynomial kernel with degree 2, that has less risk of overfitting our data.

6.2.2 RBF Kernel

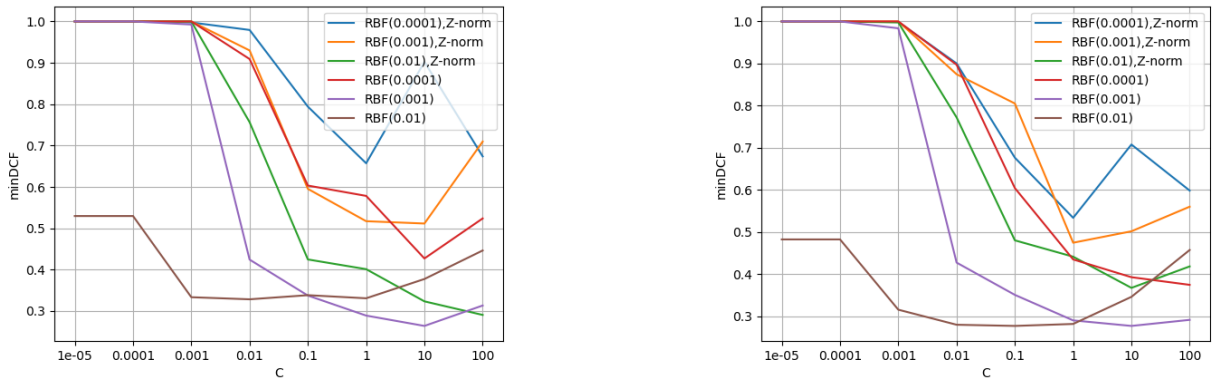
We now analyze the behaviour of the model using **RBF kernels**, we also re-balance classes because, as we have seen in Q-LogReg, this may lead to improvements for our model.



(a) NO-PCA

(b) PCA-6

We adopt a grid search approach for looking for the best hyper-parameters, the 3 more appealing models are those with $\gamma = [0.0001, 0.001, 0.01]$, so we further try to apply PCA-6 for these 3 models, without getting any particular improvement. We again try and see if z-normalization improves our 3 candidate models.



(a) NO-PCA

(b) PCA-6

Using z-normalization as pre-processing techniques only makes the model worse, thus we drop it.
In conclusion, the best model regarding RBF Kernel is with $\gamma = 0.001, C = 10$, NO-PCA (minDCF=0.264)

6.3 Conclusions

Up to now, we assumed that re-balancing classes according to our target application was the optimal choice, but now, after we selected our 2 candidate models, we can look also at their behaviour when trained with different priors, we may also look at models behaviour on our alternative applications.

Polynomial (degree 2) kernel - PCA-6, C=0.01

Embedded prior	**MinDCF**($\tilde{\pi} = 0.091$)	**MinDCF**($\tilde{\pi} = 0.048$)	**MinDCF**($\tilde{\pi} = 0.167$)
0.048	0.302	0.376	0.230
0.091	0.273	0.372	0.210
0.167	0.259	0.361	**0.200**
0.2	0.311	0.405	0.228
0.344 (Dataset)	0.270	0.362	0.212

RBF ($\gamma = 0.001$) kernel - NO-PCA, C=10

Embedded prior	**MinDCF**($\tilde{\pi} = 0.091$)	**MinDCF**($\tilde{\pi} = 0.048$)	**MinDCF**($\tilde{\pi} = 0.167$)
0.048	0.292	0.371	0.220
0.091	0.264	0.360	0.205
0.167	0.255	0.353	0.203
0.2	0.264	0.365	**0.202**
0.344 (Dataset)	0.276	0.375	0.206

According to these results, our assumptions was not totally correct, since for both models, the prior $\tilde{\pi} = 0.167$ gives better results for our target application, therefore the 2 best models are eventually:

- Polynomial (degree 2) kernel - PCA-6, C=0.01 - Embedded prior = 0.167
- RBF ($\gamma = 0.001$) kernel - NO-PCA, C=10 - Embedded prior = 0.167

7 Gaussian Mixture Models

Finally, we consider GMM classifiers. According to what we saw up to now, we expect that the best approach would be using different number of components for the two classes.

- The authentic class seems to be well described by Gaussian densities, thus we may think to use 1 or 2 guassian components to estimate its density.
- The spoofed class is composed by different sub-clusters (6 sub-classes of different spoofing techniques), thus we expect that using 4 or 8 components may give us the best results.

We first look at a Full-Covariance GMM model with NO-PCA, to see if our assumptions on the two classes are reliable

GMM-Full covariance, NO-PCA, with different components (G) for each class

Authentic	**Spoofed**	**MinDCF**($\tilde{\pi} = 0.091$)
G=1	G=2	0.290
G=1	G=4	0.264
G=1	G=8	0.273
G=1	G=16	0.293
G=2	G=2	0.306
G=2	G=4	0.283
G=2	G=8	0.285
G=2	G=16	0.302

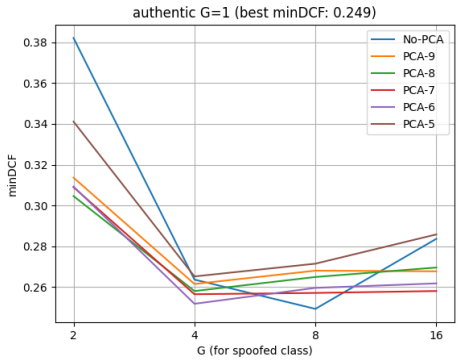
As we expected, GMM with authentic G=1 or 2, and spoofed G=4 or 8 gave us the best results.

7.1 GMM combinations

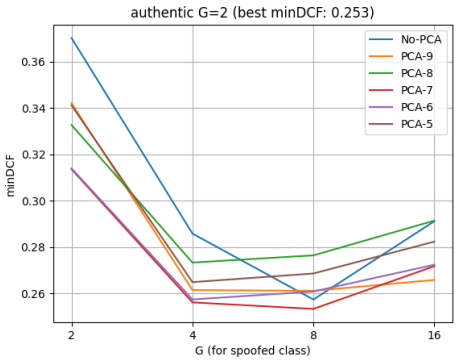
Using a full covariance matrix for each component may lead to not-reliable estimations as G grows, we may also try different combinations for both the authentic and spoofed densities, considering also Diagonal (D), Tied Full Covariance (T), and Tied Diagonal models (TD).

- **Authentic class**: we don't expect significant changes using approaches different from the full-covariance one. The samples we have are enough to reliably estimate a full covariance matrix (as we saw in MVG). Reducing the number of independent elements (thus using diagonal/tied approach) to estimate could be useful when G=2.
- **Spoofed class**: we expect to obtain some improvement trying different approaches, especially:
  - **Diagonal**: it may be useful to obtain more reliable estimates of the covariance matrix of each component, since, as we saw before, G=4 and G=8 are the best options.
  - **Tied Full Covariance** and **Tied Diagonal**: since clusters seem to have similar distributions, also the Tied covariance approach could be useful.

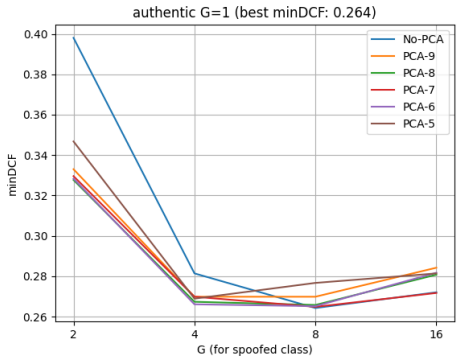
We try to adopt a grid search approach, analysing all the possible meaningful combinations for our task using also different PCAs, since the optimal dimensionality may depend on the model. We start using a Diagonal approach for the spoofed class, and varying the approach for the authentic class.



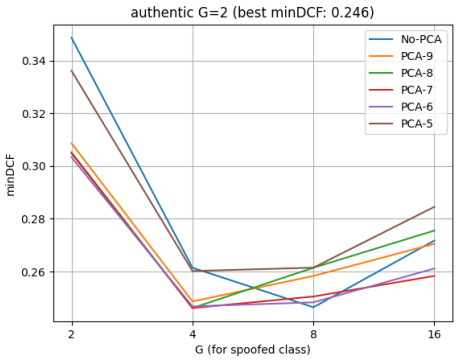
(a) D(Spoofed)-FC(authentic)



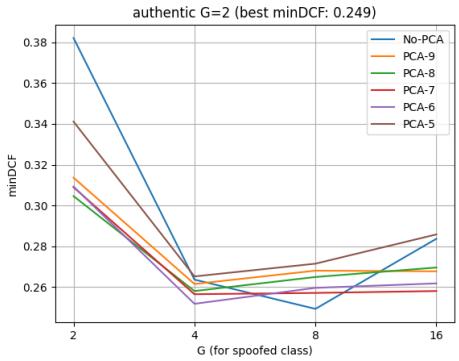
(b) D(Spoofed)-FC(authentic)



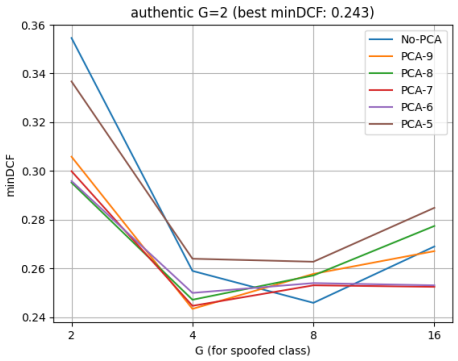
(c) D(Spoofed)-D(authentic)



(d) D(Spoofed)-D(authentic)



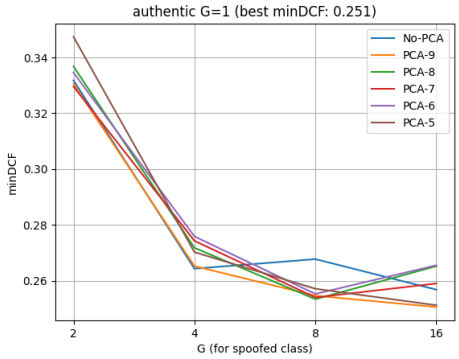
(e) D(Spoofed)-T(authentic)



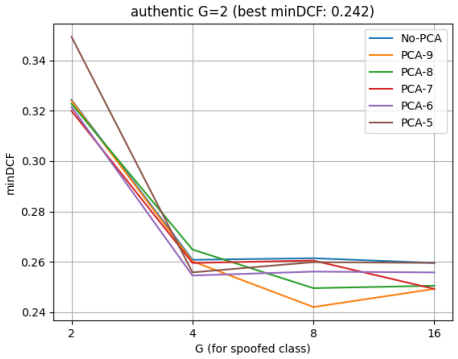
(f) D(Spoofed)-TD(authentic)

Figure 9

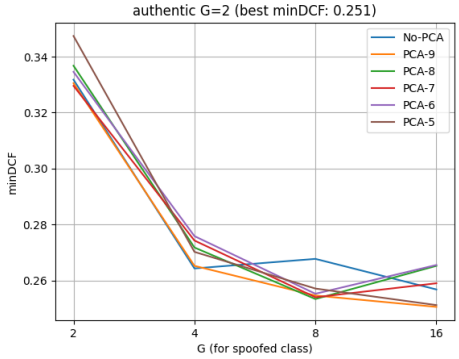
In line with our assumptions, the best performances are always together with G=4 or 8, and a **Diagonal approach** for the spoofed class performs better than the previous models we trained with the **Full covariance** approach. In addition to this, we also see that the diagonal approach makes the classification worse when used for the authentic class with G=1 (Figures 9c), on the other hand, when used together with G=2 (for the authentic class), it improves our model (Figure 9d) with respect to the full covariance approach. This is consistent with our previous assumptions regarding different approaches for the authentic class. **Tied-Diagonal**, **Tied** and **Diagonal** with G=2 and **Full covariance** with G=1 seem to be the best options for estimating the authentic class. Therefore, in the next analysis, we will consider only these 4 cases for the authentic class. We continue to use a **Tied Full Covariance** and a **Tied Diagonal** approach for the spoofed class, varying the approach for the authentic class.



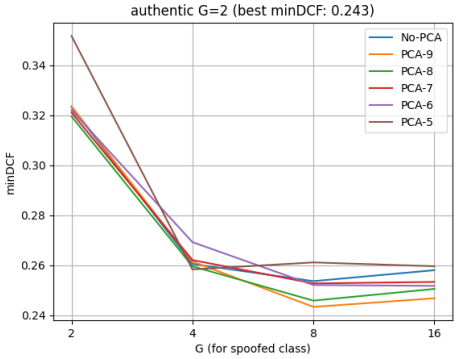
(a) T(Spoofed)-FC(authentic)



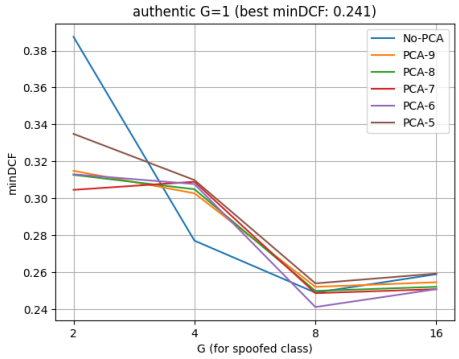
(b) T(Spoofed)-D(authentic)



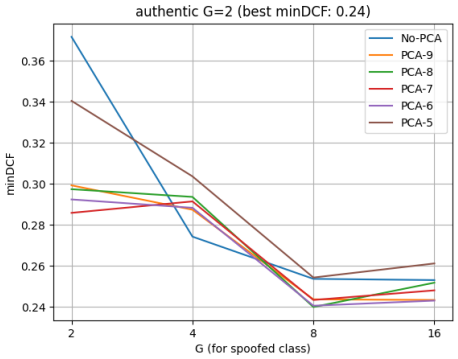
(c) T(Spoofed)-T(authentic)



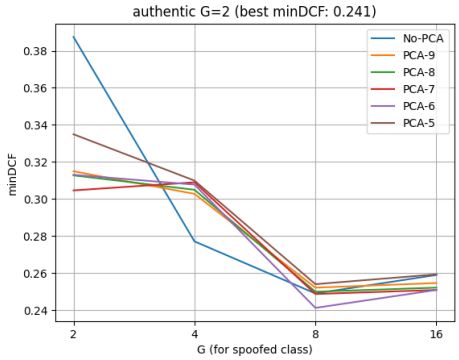
(d) T(Spoofed)-TD(authentic)



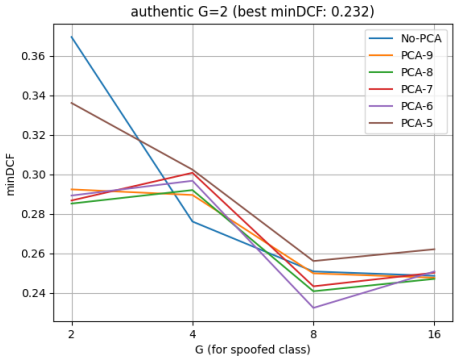
(e) TD(Spoofed)-FC(authentic)



(f) TD(Spoofed)-D(authentic)



(g) TD(Spoofed)-T(authentic)



(h) TD(Spoofed)-TD(authentic)

Figure 10

The best results are obtained by using the **Tied Diagonal** approach for both classes, G=8 for the spoofed class, G=2 for the authentic class. Tied Diagonal approach allows us to estimate only O(D) independent elements for the tied covariance matrix (same for each G component of the same class) of each class, using all class samples. In this way, with a value of G greater than 1, the model is able to generate more reliable estimates than when using different approaches such as Diagonal, Full Covariance, or Tied. We can also see that the assumption (tied assumption) that the sub-clusters of the spoofed class are similarly distributed is indeed met.

7.2 Conclusions

We thus select as best model 2-TD(authentic), 8-TD(spoofed) with PCA-6. We may now see how it behaves on our other 2 alternative working points.

2-TD(authentic) 8-TD(spoofed), PCA-6

MinDCF($\tilde{\pi} = 0.091$)	**MinDCF**($\tilde{\pi} = 0.048$)	**MinDCF**($\tilde{\pi} = 0.167$)
0.232	0.337	0.172

8 Summary

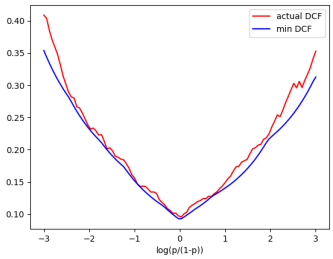
Summarizing, up to now our best models are:

	MinDCF($\tilde{\pi} = 0.091$)	**MinDCF**($\tilde{\pi} = 0.048$)	**MinDCF**($\tilde{\pi} = 0.167$)
Quadratic Log-Reg $\tilde{\pi} = 0.091, \lambda = 0.1$, PCA=6	0.257	0.349	0.207
RBF SVM $\tilde{\pi}$=0.167,γ=0.001,C=10,NO-PCA	0.255	0.353	0.203
GMM-PCA=6 Auth:2-TD Spoof:8-TD	0.232	0.337	**0.172**

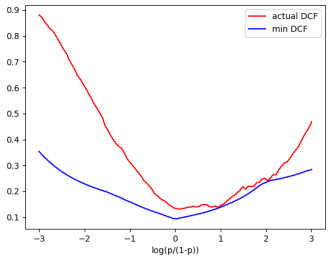
Poly(2) SVM (quadratic SVM), is very similar to Q-LogReg model, but slightly worse, thus we don't consider it.

9 Calibration

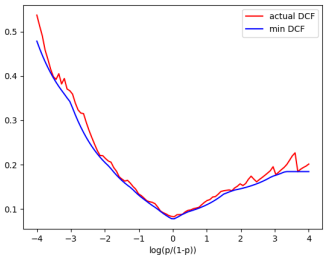
After choosing our best 3 models, we may consider calibration of them, since they may not provide well-calibrated scores (Q-LogReg and GMM), or they just don't provide scores with a probabilistic interpretation (RBF SVM). We expect that we will need to recalibrate the SVM scores, whereas the Q-LogReg and GMM scores may be already well-calibrated.



(a) Non calibrated Q-LogReg



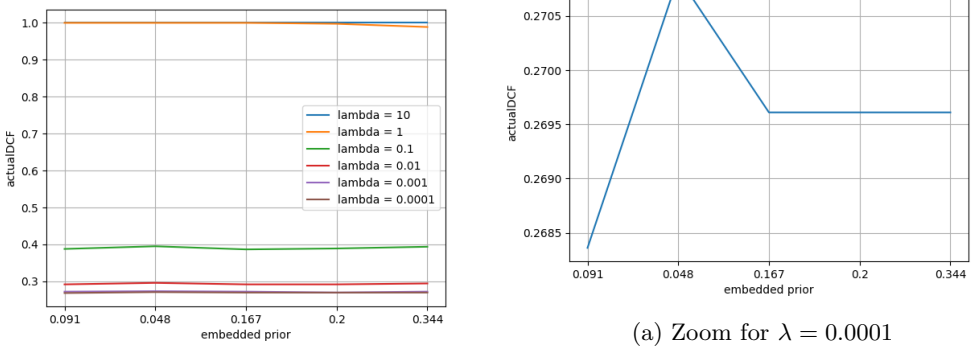
(b) Non calibrated RBF-SVM



(c) Non calibrated GMM

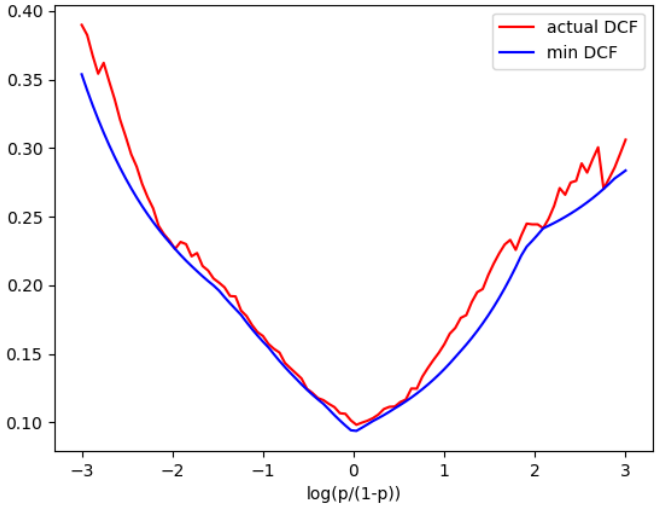
	MinDCFs($\tilde{\pi}$=0.091,0.048,0.167)	**actualDCFs($\tilde{\pi}$=0.091,0.048,0.167)**
Quadratic Log-Reg $\tilde{\pi} = 0.091, \lambda = 0.1$, PCA=6	0.257 0.349 0.167	0.275 0.408 0.210
RBF SVM $\tilde{\pi}$=0.167,γ=0.001,C=10,NO-PCA	0.255 0.353 0.203	0.695 0.88 0.485
GMM-PCA=6 Auth:2-TD Spoof:8-TD	0.232 0.337 0.172	0.246 0.359 0.176

We can see that Quadratic Log-Reg and GMM produce quite well calibrated scores, while SVM, as expected, is not well calibrated, so we use a prior-weighted logistic regression to calibrate the SVM scores.
So we do a cross-validation to find the best values for prior and λ of our prior-weighted linear LR calibration model.



(a) Zoom for $\lambda = 0.0001$

We can see that the best actualDCF is obtained with $\lambda = 0.0001$ and if we zoom further, we see that the best prior to use in training our LR is $\pi = 0.091$. We can now see how our new calibrated SVM model behaves:



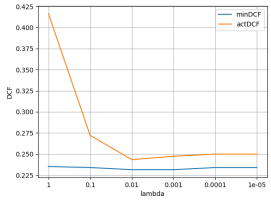
We can still observe some mis-calibration for some working points, but overall the calibration loss has been significantly reduced for the SVM model, also for our target application as we can see in the next table.

Calibrated-RBF(γ=0.001) NO-PCA, C=10

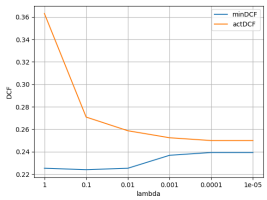
	minDCF	**actDCF**
$\tilde{\pi} = 0.091$	0.255	0.270
$\tilde{\pi} = 0.048$	0.353	0.390
$\tilde{\pi} = 0.167$	0.203	0.211

10 Fusion

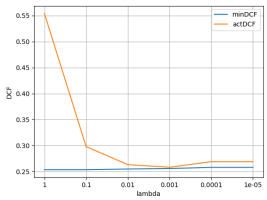
We can also consider the fusion of our best three system, employing again a prior-weighted Linear Log-Reg model, with prior $\pi_T = 0.091$ (according to previous cross-validation results for calibration). We perform cross-validation to select the best λ for our Log-Reg model.



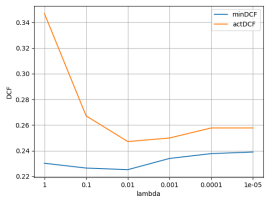
(a) GMM+SVM



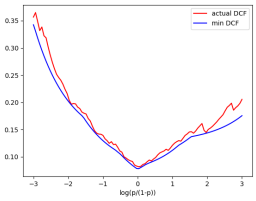
(b) QLR+GMM



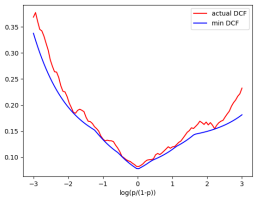
(c) QLR+SVM



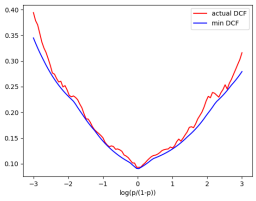
(d) QLR+GMM+SVM



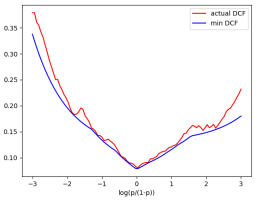
(e) $\lambda = 0.01$



(f) $\lambda = 0.01$



(g) $\lambda = 0.001$



(h) $\lambda = 0.01$

We have plots of actDCF and minDCF with respect to λ for a given fusion in the first row, the plots on the second row are the bayes error plot for the best λ of the corresponding fusion model.

Minimum and actual costs - Fusion of models, ($\tilde{\pi}$=0.091, LR-embedded prior=0.091)

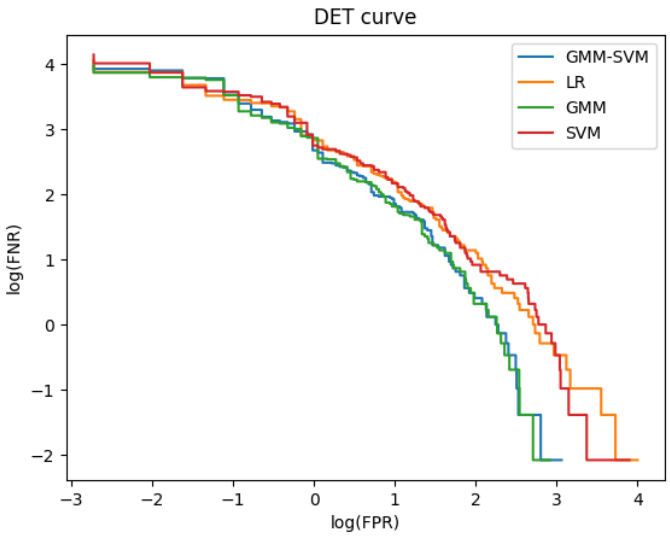
	minDCF	**actDCF**
GMM+SVM (λ=0.01)	0.231	0.243
QLR+GMM (λ=0.01)	0.225	0.259
QLR+SVM (λ=0.001)	0.256	0.258
QLR+GMM+SVM (λ=0.01)	0.225	0.247

QLR+GMM and QLR+GMM+SVM give great values for minDCF, but when computing the actualDCF we have a large loss due to miscalibration, so for these two cases the single model GMM still behaves better. However, we obtain slightly better results in terms of actDCF with **GMM+SVM**, so we choose **GMM+SVM** as the best model for our task. The main reason why actDCF is better is that the fusion model already provides well calibrated scores (since it is a linear prior weighted LR on samples with 2 features), on the other hand we didn't calibrate the GMM scores, if we had calibrated them we might have gotten similar results for actDCF with single GMM. Overall, we see that the most promising fusions are those that use **GMM**, we could argue that the fusion is useless, while it's only the **GMM** model that is responsible for the great performances.

GMM+SVM (Selected model)

	minDCF	**actDCF**
$\tilde{\pi} = 0.091$	0.231	0.243
$\tilde{\pi} = 0.048$	0.342	0.355
$\tilde{\pi} = 0.167$	0.176	0.182

We can further look at the DET plot for our 4 best models.



We can see that the 2 best models are GMM and GMM-SVM, with GMM being a bit better than the others.

11 Evaluation

After selecting and training on training set our best models (**GMM+SVM**), we now analyze how they behave on the evaluation data. According to our expectations, **GMM+SVM** is the best model for our target application,

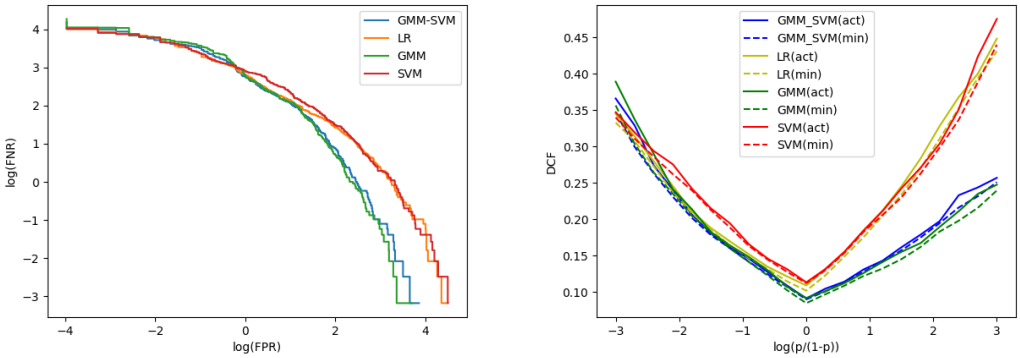
On evaluation/test set

	minDCF $\tilde{\pi}$**=0.091**	**minDCF** $\tilde{\pi}$**=0.048**	**minDCF** $\tilde{\pi}$**=0.0167**	**actDCF** $\tilde{\pi}$**=0.091**	**actDCF** $\tilde{\pi}$**=0.048**	**actDCF** $\tilde{\pi}$**=0.167**
Quadratic Log-Reg $\tilde{\pi} = 0.091, \lambda = 0.1$, PCA=6	0.261	0.333	0.194	0.266	0.345	0.196
RBF SVM (calibrated) $\tilde{\pi}$=0.167,γ=0.001,C=10,NO-PCA	0.278	0.340	0.222	0.281	0.346	0.225
GMM-PCA=6 Auth:2-TD Spoof:8-TD	0.254	0.356	0.189	0.277	0.389	0.190
GMM+SVM Fusion	0.252	0.346	**0.186**	0.264	0.365	**0.189**

however, **Q-LogReg** behaves better for one of the alternative working points (As we saw in bayes error plot, Q-LogReg was better-calibrated on a wider range of applications on the left side (where our working points are) than GMM+SVM). Except for SVM, all models present a cost due to mis-calibration error, we may think that, especially for GMM, calibration at training time could have been a great choice.

On validation set (taken from cross validation on training during our analysis)

	minDCF $\tilde{\pi}$=0.091	**minDCF $\tilde{\pi}$=0.048**	**minDCF $\tilde{\pi}$=0.0167**	**actDCF $\tilde{\pi}$=0.091**	**actDCF $\tilde{\pi}$=0.048**	**actDCF $\tilde{\pi}$=0.167**
Quadratic Log-Reg $\tilde{\pi} = 0.091, \lambda = 0.1$, PCA=6	0.257	0.349	0.167	0.275	0.408	0.210
RBF SVM (calibrated) $\tilde{\pi}$=0.167,γ=0.001,C=10,NO-PCA	0.255	0.353	0.203	0.270	0.390	0.211
GMM-PCA=6 Auth:2-TD Spoof:8-TD	0.232	0.337	0.172	0.246	0.359	**0.176**
GMM+SVM Fusion	0.231	0.342	**0.176**	0.243	0.355	0.182



(a) DET plot of 4 best models on evaluation (b) B.error plot of 4 best models on evaluation

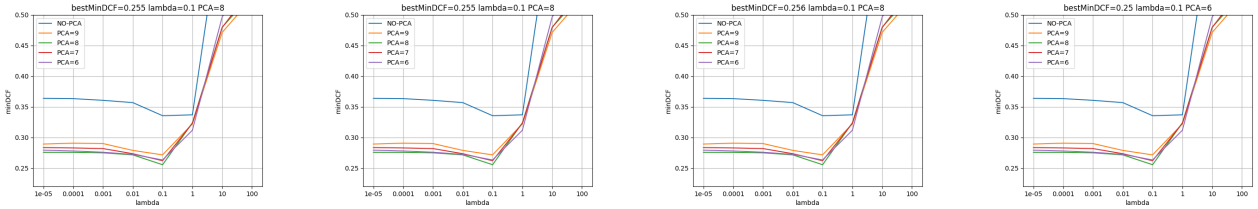
Again, as in the previous DET plot analysis, we see that the best models are GMM and GMM-SVM, with GMM being slightly better.
The bayes error plot confirms what we saw in the DET plot since SVM is worse than the other models for most of the applications, and LR behaves well for the left side, while is worse for the right side. We can see that overall our models are well-calibrated.

12 Post-evaluation analysis

We now look at our previous model choiches (models, hyperparameters), to assess whether we obtained optimal, close to optimal or sub-optimal results. We won't analyze linear classifiers (linear LR, Tied Gaussian), since we saw they are not suited for the task, neither Full covariance and Diagonal Gaussian, since they can be seen as specific cases of our GMMs (1(FC)-1(FC) and 1(D)-1(D)). We will use minimum DCF as metric.

12.1 Q-LogReg models

We start from Q-Log-Reg models, we analyze their behaviour on the evaluation data varying λ, using different priors to train the model, and with different values for PCA.

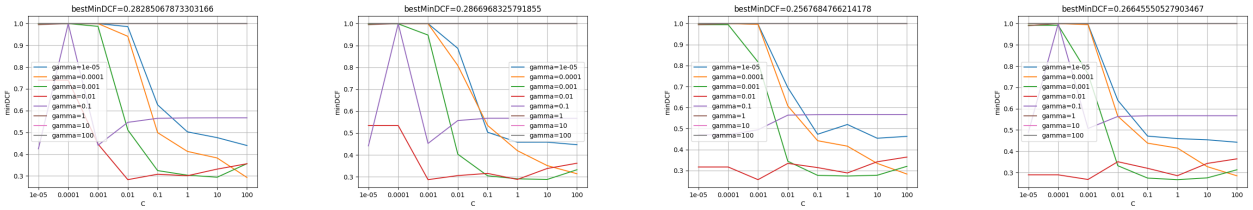


(a) π_T=0.048 (b) π_T=0.091 (c) π_T=0.167 (d) π_T=0.344(Dataset)

We can see that the best model is with $\lambda = 0.1$, PCA-6, and $\pi_T = 0.344$ with a minDCF=0.25 (vs 0.261 of our chosen model), thus our model is **sub-optimal**. We see that, in contrast with what we expected, embedding a prior matched with our application prior was not the correct decision.

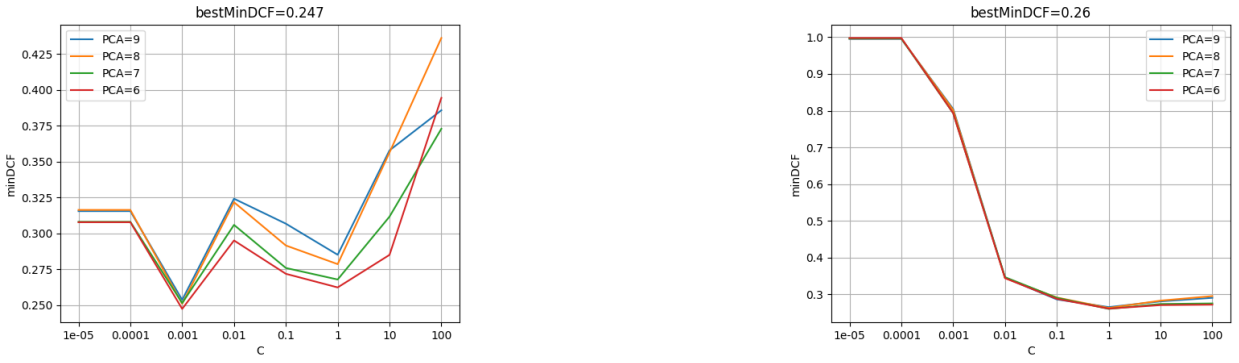
12.2 SVM models

We skip the analysis of quadratic SVM (Poly(2)), since it's similar to Q-LogReg, and expensive to analyze. We thus analyze our chosen model *RBF (γ=0.001), C=10, NO-PCA, Embedded prior=0.167*.
We start by varying γ, C, and the embedded prior used for rebalancing classes, without PCA.



(a) π_T=0.048 (b) π_T=0.091 (c) π_T=0.167 (d) π_T=0.2

The best combination is with γ = 0.01, C=0.001 and π_T=0.167, with minDCF=0.257, with respect with our chosen configuration that provided minDCF=0.278. Im general both γ = 0.01 and γ = 0.001 give good results. We further apply PCA using the two values of γ to see if we can improve the model.



(a) γ=0.01 (b) γ=0.001

We see that the best choice for our evaluation set was with γ = 0.01, C=0.001 and PCA-6 (0.247 vs 0.278), thus we made a **sub-optimal decision**

12.3 GMM models

We chose 2-TD+8-TD with PCA-6 as best model, now we analyze the alternatives, we will directly report the best configuration for each model combination that was considered during model selection.

	PCA	**minDCF$\tilde{\pi}$=0.091** on evaluation data	**minDCF $\tilde{\pi}$=0.091** on validation data (k-fold on training)
2(FC) - 4 (FC)	6	0.251	0.263
2(FC) - 8 (D)	7	0.244	0.253
2(FC) - 8 (T)	7	0.247	0.260
1(FC) - 16 (TD)	-	0.245	0.259
2(D) - 4 (FC)	-	0.242	0.266
2(D) - 16 (D)	7	0.241	0.258
2(D) - 8 (T)	-	0.244	0.261
2(D) - 16 (TD)	-	0.240	0.253
2(TD) - 4 (FC)	-	0.244	0.276
2(TD) - 8 (D)	8	0.246	0.257
2(TD) - 8 (T)	-	0.242	0.254
2(TD) - 16 (TD)	-	0.242	0.249

Again, we assess that we made a **sub-optimal choice**, the assumption of the combination TD-TD was quite correct, but we reduced too much the number of features (PCA-6 instead of NO-PCA), and we chose the wrong number of components for the spoofed class (8 instead of 16). We could also observe that the models with

Our chosen model

	PCA	**minDCF$\tilde{\pi}$=0.091** on evaluation data	**minDCF $\tilde{\pi}$=0.091** on validation data
2(TD) - 8 (TD)	6	0.254	0.232

FC for the authentic class lead to lower performances, probably because, since we have fewer samples of the authentic class, they tend to overfit. On the contrary, we see that the diagonal assumption for the authentic class, along with the use of 2 components would have been the best option.

12.4 Fusion

We can also see how our best fusion would have been affected if we used the optimal configurations for our models.

	minDCF$\tilde{\pi}$=0.091 on evaluation data	**actDCF $\tilde{\pi}$=0.091** on evalutation data
GMM+SVM (sub-optimal models)	0.252	0.264
GMM+SVM (optimal models)	0.240	0.267

We used $\lambda = 0.001$ for the fusion of the optimal models (instead of the $\lambda = 0.01$ used for the fusion of our chosen sub-optimal models). The fusion with optimal models has a lower minDCF, however it has a great mis-calibration error on our target application, bringing it to behave almost like our previously chosen fusion model.

12.5 Summary

We report in the following table the difference for what regards minDCF between optimal and sub-optimal (our choices) configurations

Results on evaluation set

	minDCF our choice	**minDCF** optimal model
QLog-Reg	0.261	0.250
SVM	0.278	0.247
GMM	0.254	0.240
GMM+SVM	0.252	0.240

We can see that we did not make particularly bad decisions regarding Q-LogReg, GMM and GMM+SVM, moreover, even with optimal configurations we can see that **GMM** and **GMM+SVM** would have been the better choices (as we estimated during our analysis). On the other hand, we made bad decisions regarding SVM, which is probably due to the fact that we did not do a deep enough analysis on the different PCA dimensions. It's also clear that the goodness of the GMM+SVM model is most likely due to the GMM scores alone.