Project Report

Author: Matteo Celia s316607@studenti.polito.it
Computer Engineering course (Artificial Intelligence and Data Analytics)
Polytechnic University of Turin

Contents

1 Introduction

The project task consists of a binary classification problem. The goal is to perform fingerprint spoofing detection, i.e. to identify genuine vs counterfeit fingerprint images. The dataset consists of labeled samples corresponding to the genuine (True, label 1) class and the fake (False, label 0) class. The samples are computed by a feature extractor that summarizes high-level characteristics of a fingerprint image. The data is 6-dimensional.

2 Visualization

In the following section is reported the analysis and visualization of the data.

Histogram plots of the dataset:

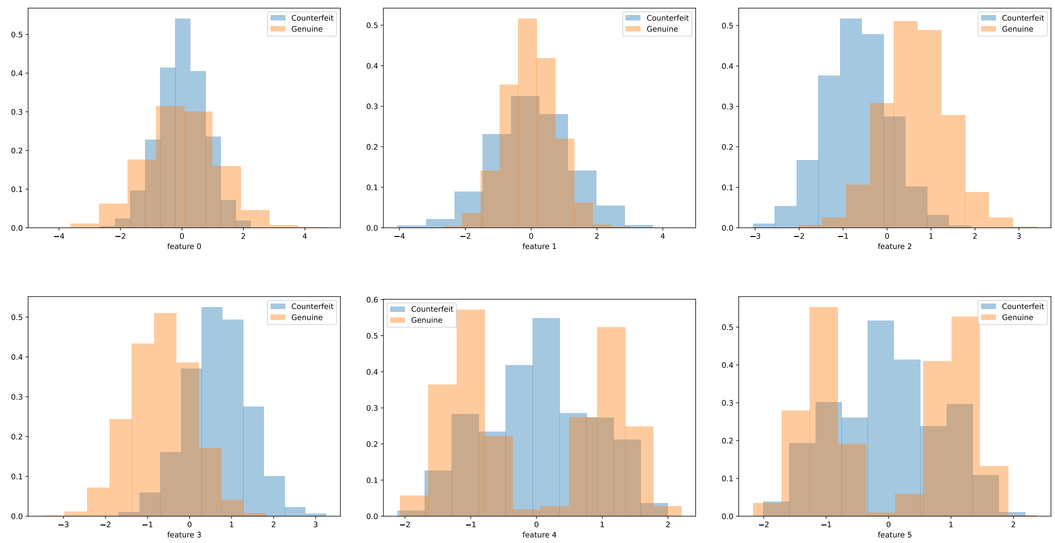

Figure 1: Histogram plots of the dataset features

For both the first two features a large class overlap is present in the region around the mean for the two classes. It can be seen that the first two features resemble gaussian distributions with (approximately) zero mean and different variance. For the first feature the variance is 0.569 for False class and 1.43 for True class. For the second feature the variance is 1.42 for False class and 0.578 for True class. It is clear that for both features there is just one mode or ”peak” (one for each class).

Analyzing the third and fourth features it is noticeable that the overlap is smaller than it was for the first two features and is related to the ”tails” of the gaussians for the two classes. The distributions for the two classes are gaussian shaped, with almost identical variance and different mean. Also in this case, for both features, one mode or ”peak” is present for each class.

Considering the last two features, there is a large overlap between the tails of the distribution of the False class (Counterfeit) and the distribution of the True class (Genuine). In this case two modes are present in both features for the True class and just one for the False class. It has been shown in the past that there is a difference in the characteristics that define male and female fingerprints. Even though the data is represented by a 6-dimensional vector that summarizes high-level characteristics of a fingerprint, it is reasonable to assume that the two distributions regarding the True class for these last two features might resemble the male and female distribution of fingerprint images. On the other hand counterfeit fingerprints don't show the same trend because since they are counterfeit, their values for these features are distant from the True class distribution.

Scatter plots of the dataset:

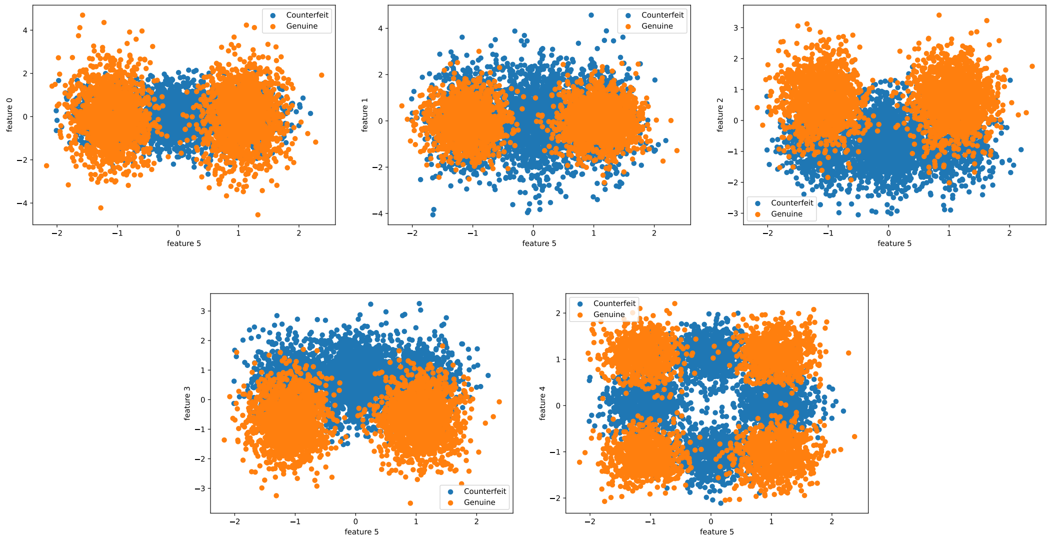

Figure 2: Pair-wise scatter plots between feature 5 and all other features

Regarding the pair-wise scatter plots, it can be seen that, when considering feature 5 and the other features (except feature 4), two clusters are present for the Genuine class and just one for the Counterfeit class. When considering pair-wise scatter plot between feature 4 and 5 instead, four clusters can be identified for each class. In general it seems that there is no linear relationship between the features.

3 Dimensionality Reduction

Dimensionality reduction is a technique used to reduce the number of input variables or features in a dataset. The primary goal of dimensionality reduction is to simplify the dataset while preserving its important characteristics and reducing noise, redundancy, and computational cost. In order to do that, a mapping is computed from the **n**-dimensional feature space to a **m**-dimesnional feature space with $\mathbf{m} << \mathbf{n}$.

The two most used methods for dimesnionality reduction are: Principal Component Analysis (PCA, unsupervised) and Linear Discriminant Analysis (LDA, supervised).

3.1 PCA

Principal Component Analysis (PCA) is an unsupervised (meaning it doesn't take into account the classes in the data) dimensionality reduction method which is used to compute a projection matrix $\mathbf{P}$ by minimizing the average reconstruction error:

$$\frac{1}{\mathbf{K}}\sum_{i=1}^{K}||\mathbf{x_i}-\hat{\mathbf{x}}_\mathbf{i}||^2$$

where $\mathbf{K}$ is the number of samples and $\hat{\mathbf{x}}_\mathbf{i} = \mathbf{Py}$, which is the reconstruction of the projected point $\mathbf{y}$ in the original space.

It can be shown that the optimal solution is then given by the matrix $\mathbf{P}$ whose columns are the m eigenvectors of $\frac{1}{\mathbf{K}}\sum_{i=1}^{K}\mathbf{x_i}\mathbf{x_i}^\intercal$ corresponding to the m largest eigenvalues.

In the end PCA can be interpreted as the linear mapping that preserves the directions with highest variance.

In the following are reported the histogram plots of the projected features for the 6 PCA directions starting from the the principal component (top-left) related to the largest variance.

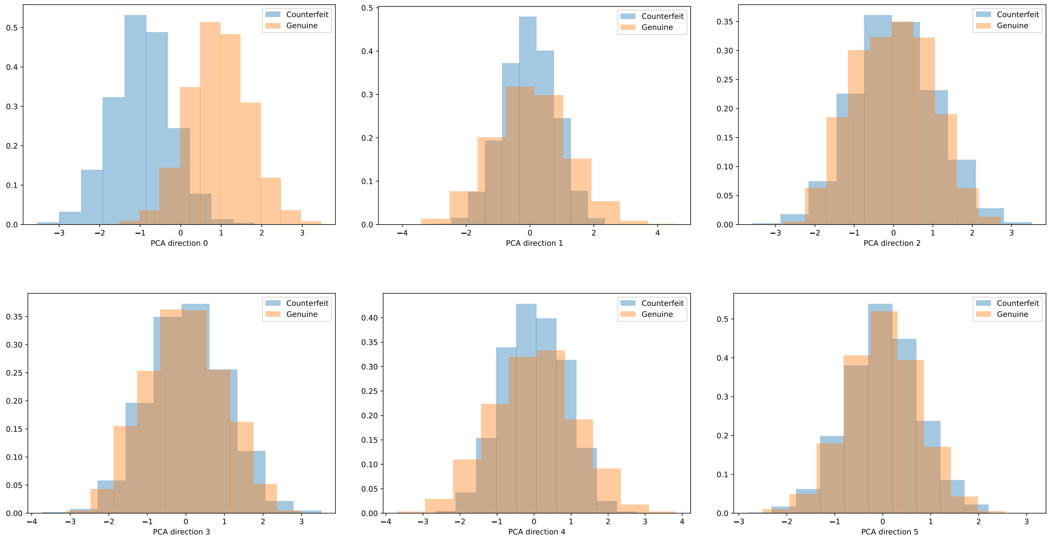

Figure 3: Histogram plots of the projected features for the 6 PCA directions

It's easy to see how the projection over the principal component is the one that shows less overlap between the two class distributions. Indeed as said before, this is the direction corresponding to the highest variance, hence it is reasonable to think that it will show less overlap. The overlap between the two class distributions is larger for the projections over the other PCA directions. From these histograms though, it's not easy to spot the different clusters inside each class.

PCA is an unsupervised method, thus it doesn't take into account the class of the samples when computing the directions. A better way to obtain a discriminant direction for the dataset could be using LDA.

3.2 LDA

Linear Discriminant Analysis (LDA) is a supervised dimensionality reduction method particularly useful when the data points of each class are scattered along the same directions of the class mean thus making it difficult to properly separate the classes.

Fisher Linear Discriminant Analysis aims at finding a direction that has a large separation between the classes and small spread inside each class. The spread is measured in terms of class (co)variance.

LDA thus wants to maximize the between-class variability over within-class variability ratio for the transformed samples in order to find the direction $\mathbf{w}$:

$$\max_{\mathbf{w}} \frac{\mathbf{w}^\intercal \mathbf{S_B} \mathbf{w}}{\mathbf{w}^\intercal \mathbf{S_W} \mathbf{w}}$$

In the following are reported the histogram plot of the projected features for the LDA direction. The direction is only one because LDA allow computing $\mathbf{C} - 1$ directions, where $\mathbf{C}$ is the number of classes (2 in this case).

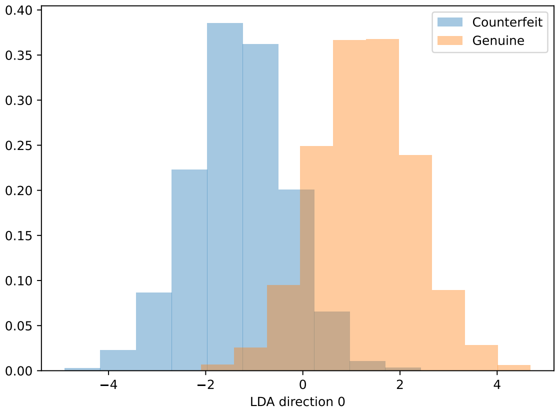

Figure 4: Histogram plot of the projected features over the LDA direction

It can be seen that the projection over the LDA direction results in low class overlap, lower than any class overlap present on the original features histograms 1.

The result, however, is comparable to the one obtained from the principal component direction in PCA (3), meaning the distribution of the samples can be easily described based on the variance of the data even without considering the class information.

3.2.1 LDA for classification

LDA was originally introduced to solve binary problems. Indeed, once we have estimated $\mathbf{w}$, we can project our test samples over $\mathbf{w}$, and assign the class according to whether the projected value (score) is larger or lower than a given threshold.

The dataset is split in two parts: model training data (that will be used to estimate the model parameters) and validation data (that will be used for evaluation and model selection). It's employed a simple random split that assigns $\frac{2}{3}$ of the samples to the model training partition and $\frac{1}{3}$ of the samples to the validation partition.

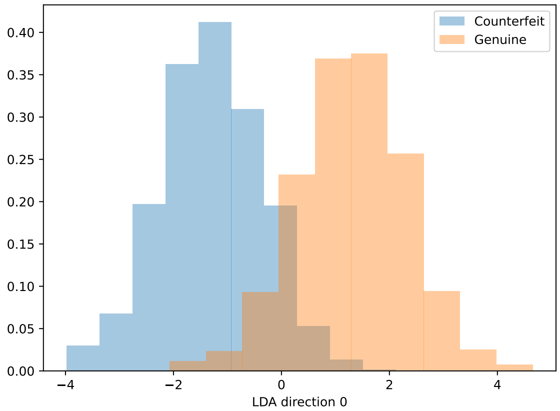

Figure 5: Histogram plot of the projected validation set over the LDA direction

In the following are reported the results of classification using LDA and employing as threshold the mean of the projected class means, computed on the training data. The samples are then classified as Genuine (1) if they are greater than or equal the threshold (in the LDA space) or Counterfeit (0) otherwise. For this specific threshold the error rate obtained is: 0.093.

Figure 6 shows the value of the error rate corresponding to different thresholds in the range [-1,1]. In particular, the minimum error rate achieved is: 0.092 at threshold: 0.0303 (the same error rate might be achieved also at different thresholds). It can be seen that as the threshold moves away from 0 in both directions, the error rate increase. That's because, as it can be seen from Figure 5, the projection of the validation set over the LDA direction produces a small overlap around 0 (on the x-axis).

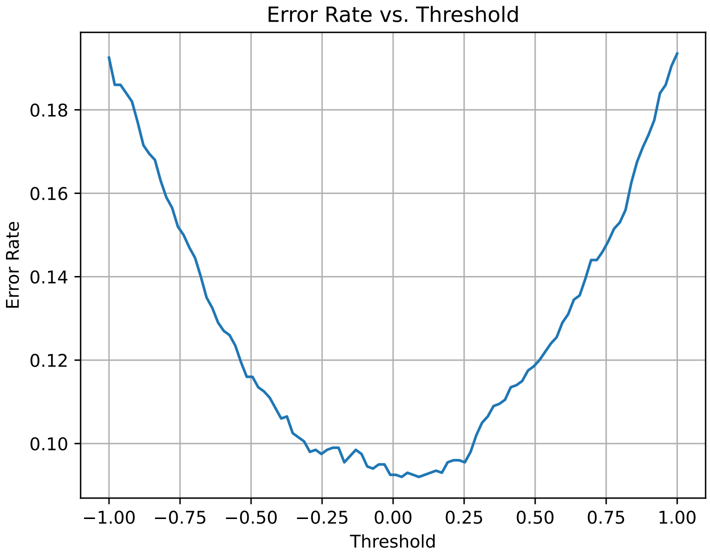

Figure 6: Error rate value with respect to the threshold used for classification

Now, before classifying the validation set, the features are preprocessed with PCA (at different values of m and estimated only on the training data). Results are reported in Table 5. Note: values of m like 1 and 6 are not considered because reducing to 1 dimension makes LDA irrelevant, whereas reducing to 6 dimensions does not change the LDA subspace.

PCA dimensions	m=2	m=3	m=4	m=5
Error rate	0.9075	0.0925	0.0925	0.093

Table 1: Error rate when pre-processing using different number PCA dimensions

At m=2 the error rate is so high because in this case the Genuine and Counterfeit class are ”inverted” so that values greater than the threshold (classified as Genuine) mostly belong to the Counterfeit class and vice versa. If the behavior of the classifier is changed so that values greater than the threshold are assigned to the Counterfeit class, the error rate for m = 2 becomes: 0.0925 (thus equal to the lowest value found for other values of m but using a lower number of dimensions).

Overall, when preprocessing with PCA, the error rate is slightly reduced from 0.093 to 0.0925 (so not that useful) and of course the computation effort is reduced too considering that the LDA direction will be computed on a smaller dimension dataset.

4 Gaussian models

Uni-variate Gaussian models were fitted to the different features of the different classes of the dataset. For each class, for each component of the feature vector of that class, the ML estimate have been computed (empirical mean and covariance matrix). In the following are reported the distribution density on top of the normalized histogram.

For the first four features, considering both classes, the gaussian densities provide a good fit. In the case of the last two features instead the actual data is not well described by the gaussian densities. This is more true for the Genuine class than for the Counterfeit class.

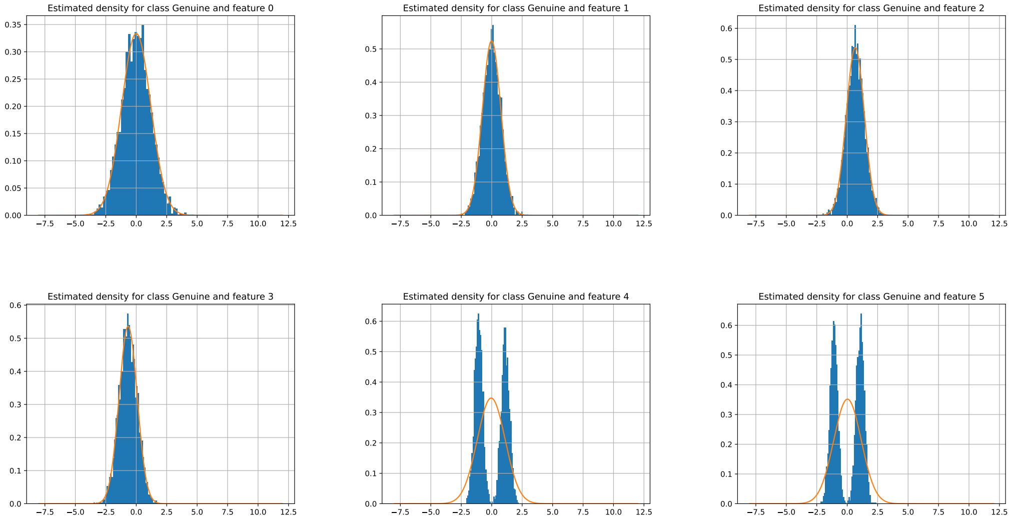

Figure 7: Gaussian distribution densities on top of the normalized histogram for the Genuine class

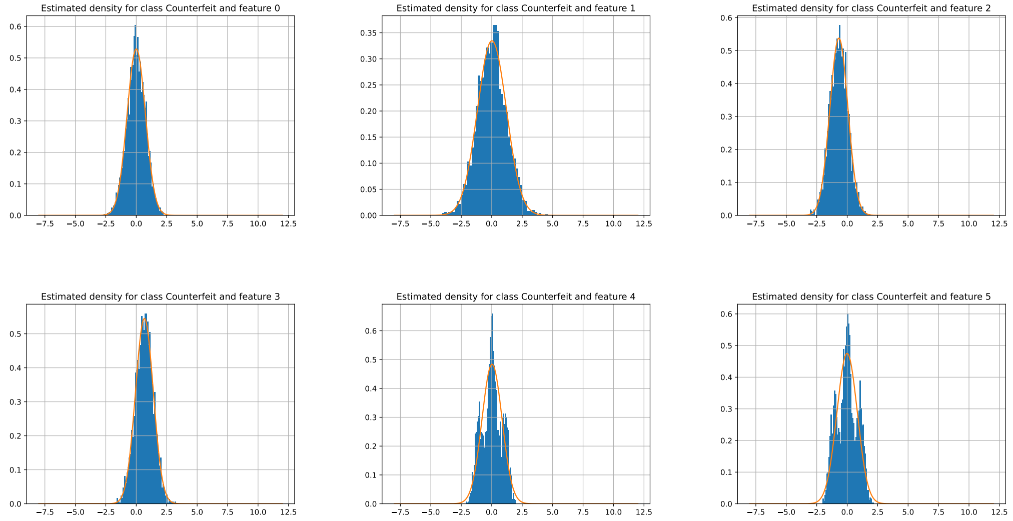

Figure 8: Gaussian distribution densities on top of the normalized histogram for the Counterfeit class

4.1 Classification

Now different gaussian models are employed for classification: MVG, MVG with Tied covariance matrix for all classes and Naive Bayes that considers diagonal class covariance matrix (assumes featues indipendence). The error rate for these models are reported in the following and compare with the result obtained using LDA as a classifier. The prior used for all these experiments is the same for both classes (0.5).

model	MVG	Tied	NB	LDA
Error rate	0.07	0.093	0.072	0.093

Table 2: Error rate for the gaussian models compared to LDA

It can be seen that the best performing model is MVG with full covariance matrix for each class. Moreover, the Tied model provides the same accuracy as the LDA model. Indeed, this can be explained considering that LDA assumes that all classes have the same within–class covariance. The Naive Bayes model performs almost as good as MVG but using only diagonal class covariance matrices. This can point to the fact that the features are not strongly correlated.

To verify this hypothesis the Pearson correlation matrices for the dataset classes are reported.

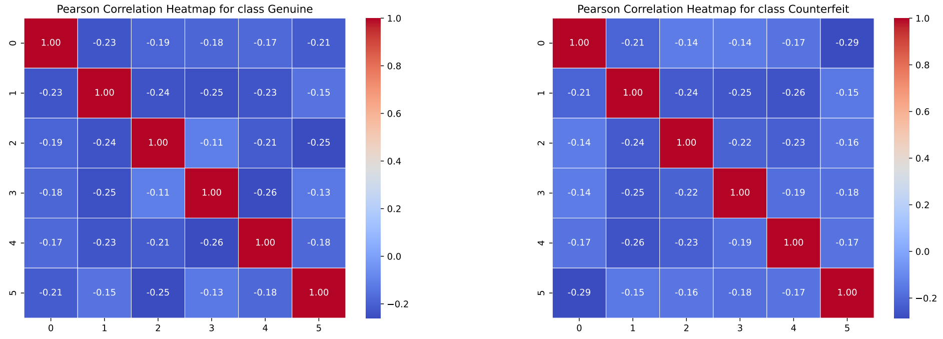

Figure 9: Pearson correlation matrices for the dataset classes

Indeed it can be seen that the covariance between features is relatively low thus making the Naive Bayes assumption holds good enough.

The Gaussian model assumes that features can be jointly modeled by Gaussian distributions. The goodness of the model is therefore strongly affected by the accuracy of this assumption. In 7 and 8 are reported the result of fitting a Gaussian density over each feature for each class which corresponds to the Naive Bayes model. As said before the gaussian assumption holds for the first four features of both classes while it doesn't hold that much for the last two.

In the following are reported the performance of the models only the first four features to check if this might be beneficial.

model	MVG	Tied	NB
Error rate	0.0795	0.095	0.0765

Table 3: Error rate for the gaussian models considering only the first 4 features

The results show a worsening in the performance of all three models as a consequence of removing the last two features. Despite the inaccuracy of the gaussian assumption for the last two features,it seems like the gaussian models are still able to extract some useful information to improve classification.

Earlier on, it was analyzed the distribution of features 1-2 and 3-4 (Note: in the plots the feature number goes from 0 to 5 but in the following i will refer to the first feature as 1, the second as 2 and so on) finding that for features 1 and 2 means are similar but variances are not, whereas for features 3 and 4 the two classes mainly differ for the feature mean, but show similar variance. Therefore it could be useful to analyze how these characteristics of the features distribution affect the performance of the different approaches. In the following are reported the results of the classification experiments considering first only the features 1-2 (jointly) and then only the features 3-4 (jointly).

model	MVG	Tied
ER features 1-2	0.365	0.0495
ER features 3-4	0.0945	0.094

Table 4: Error rate for the gaussian models considering features 1-2 (jointly) and then 3-4 (jointly)

Regarding the first set of features (1-2), the best performing model is the MVG while for the second set of features (3-4) the best performing model (even if only by a small margin) is the Tied model. These results can be explained by the fact that the Tied model considers the same (indeed tied) class covariance matrix for all the classes. Thus, because like said before the features 3-4 show a similar variance while features 1-2 don't, it is reasonable to obtain better performance for the Tied model when using only features 3-4 rather than 1-2. For this reason MVG performs better than Tied when using features 1-2, since it can better incorporate the different information from the two classes.

4.2 PCA preprocessing

In the following are reported the results of the classifation of the gaussian models when preprocessing with PCA.

PCA dimensions	m=2	m=3	m=4	m=5
MVG	0.088	0.088	0.0805	0.071
Tied	0.0925	0.0925	0.0925	0.093
NB	0.0885	0.09	0.0885	0.0875

Table 5: Error rate of gaussian models when preprocessing with PCA

Overall the performance doesn't seem to deteriorate too much after applying PCA, hence if there is a good tolerance on the error rate, it might be beneficial to work with less dimensional features. The best performing model is MVG with no PCA (0.7 error rate). It is worth nothing though that with PCA 5 MVG returns an error rate of 0.071 only slightly more than MCG with no PCA while at the same time reducing the number of dimensions to 5. Moreover, Naive Bayes with no PCA provides an error rate of 0.072, so still very low, while requiring to compute a lot less parameters (diagonal covariance matrix) than MVG with full covariance matrix.

4.3 Detection Cost Function Analysis

In this section the goal is to make optimal decisions when priors and costs are not uniform. Optimal Bayes decisions are decisions that minimize the expected Bayes cost, from the point of view of the recognizer.

First, the error rates for the gaussian models are reported with respect to different working points hence effective prior $\tilde{\pi}$.

eff prior $\tilde{\pi}$	0.1	0.5	0.9
MVG	0.1375	0.07	0.1395
Tied	0.168	0.093	0.1705
NB	0.136	0.072	0.142

Table 6: Error rate of gaussian models with different effective priors

Now, for each application are computed the optimal Bayes decisions for the validation set for the MVG models and its variants, with and without PCA. The actual and minimum DCF are computed for eaach applciation to compare the different models.

eff prior $\tilde{\pi}$	0.1(minDCF)	0.1(actDCF)	0.5(minDCF)	0.5(actDCF)	0.9(minDCF)	0.9(actDCF)
MVG(no PCA)	0.262	0.305	**0.13**	**0.14**	**0.342**	0.40
MVG(PCA 2)	0.352	0.388	0.173	0.176	0.438	0.443
MVG(PCA 3)	0.356	0.388	0.173	0.175	0.439	0.468
MVG(PCA 4)	0.301	0.353	0.153	0.160	0.415	0.460
MVG(PCA 5)	0.273	0.304	0.133	0.142	0.351	0.398
Tied (no PCA)	0.362	0.406	0.181	0.186	0.442	0.462
Tied (PCA 2)	0.363	0.396	0.178	0.185	0.435	0.478
Tied (PCA 3)	0.368	0.408	0.183	0.185	0.434	0.456
Tied (PCA 4)	0.360	0.403	0.182	0.185	0.444	0.461
Tied (PCA 5)	0.364	0.405	0.181	0.186	0.445	0.462
NB (no PCA)	**0.257**	**0.302**	0.131	0.144	0.351	**0.389**
NB (PCA 2)	0.356	0.386	0.171	0.176	0.432	0.442
NB (PCA 3)	0.364	0.395	0.174	0.18	0.434	0.459
NB (PCA 4)	0.361	0.397	0.171	0.177	0.431	0.463
NB (PCA 5)	0.354	0.393	0.173	0.175	0.434	0.466

Table 7: Actual and minimum DCF for guassian models and different applications

Across different application the best models are MVG and Naive Bayes both without applying PCA considering both actual DCF and minimum DCF.

With the exception of the application with effective prior equal to 0.5, the models are, overall, not well calibrated (e.g. for application with effective prior equal to 0.1 the best performing model has minDCF=0.257 and actDCF=0.302). This can be caused by the fact that the models are trained using a balanced dataset so that there is a bigger mis-match between the application effective prior and the empirical prior for applications with $\tilde{\pi}$=0.1 and $\tilde{\pi}$=0.9.

Considering now the application with $\tilde{\pi}$=0.1 (the main application), in the following are reported the Bayes error plots of the gaussian models with no PCA and PCA 5 which corresponds to the best results.

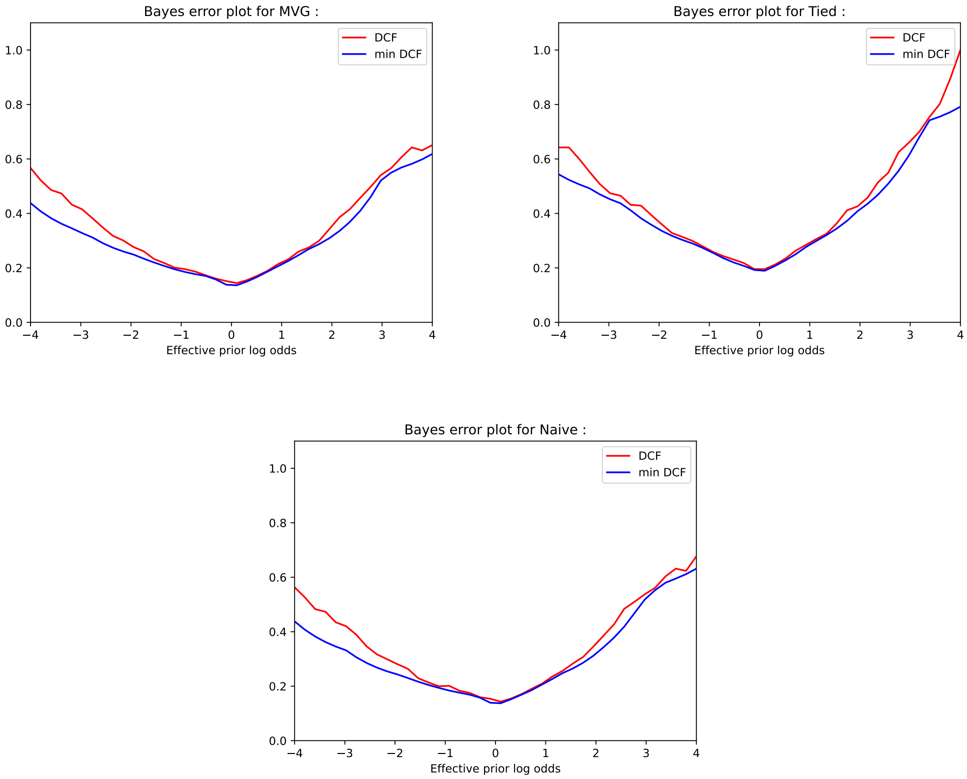

Figure 10: Bayes error plots for gaussian models with no PCA

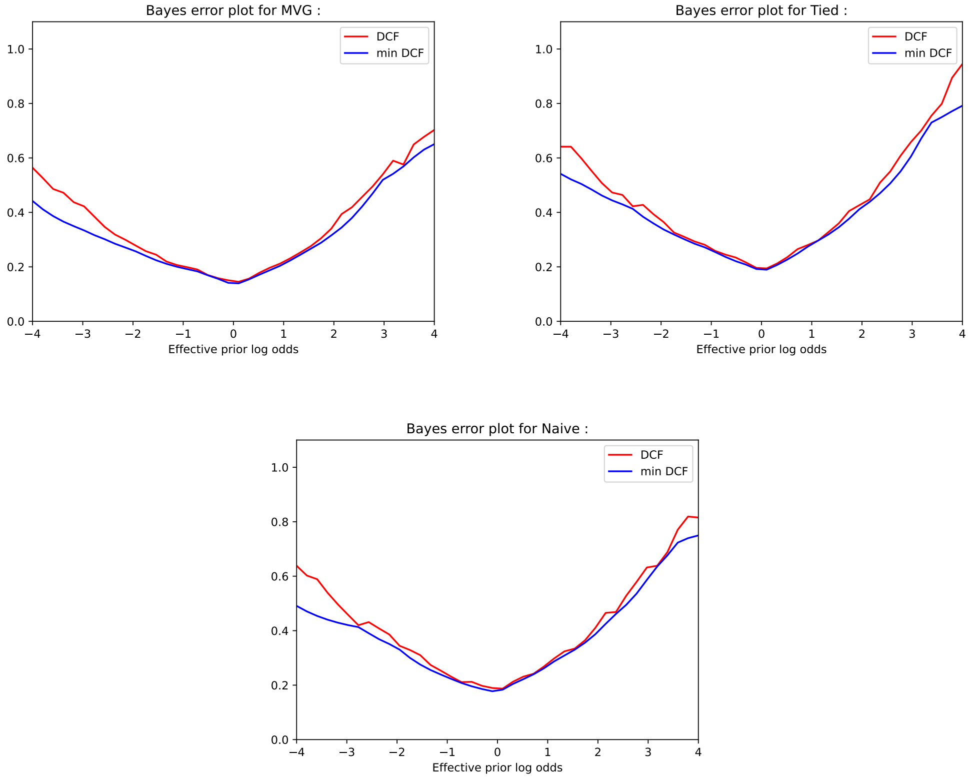

Figure 11: Bayes error plots for gaussian models with PCA 5

Overall, the minimum DCF gets larger when the effective prior moves away from the uniform prior ($\tilde{\pi}$=0.5). Around the value 0 for the prior log odds ($\log \frac{\tilde{\pi}}{1-\tilde{\pi}}$) the models seems good-enough calibrated while as the prior log odds moves away the models get less and less calibrated.

5 Logistic Regression

In this section the binary logistic regression model is going to be tested on the data.

In the following are reported the results and plots of the actual and minimum DCF of the different variants of logistic regression with respect to different value of λ (the parameter that controls the strength of the regularization). All the results are computed considering the application prior: $\tilde{\pi} = 0.1$.

It can be seen that for the linear LR model, larger values of λ degrade actual DCF since the regularized models tend to lose the probabilistic interpretation of the scores, hence the divergence between actual and min DCF.

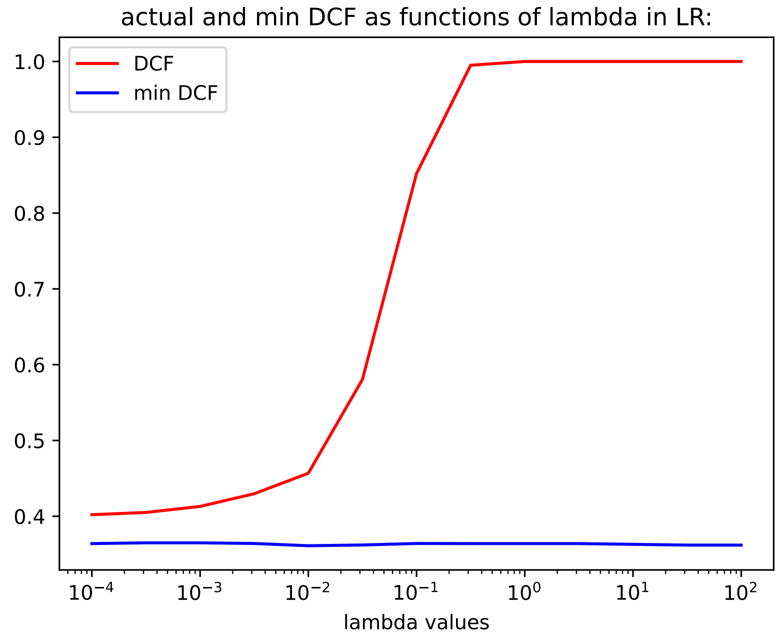

Figure 12: Actual and min DCF with respect to λ for linear LR

To better understand the role of regularization, it can be useful to analyze the results obtained if there where fewer training samples. Indeed the following plot shows the result for linear LR when keeping 1 out of 50 training samples. In this case, it can be seen that for certain λ values the actual DCF decreases until, at some point starts to increase again. Indeed higher values of the regularizer reduce overfitting, but may lead to underfitting and to scores that lose their probabilistic interpretation.

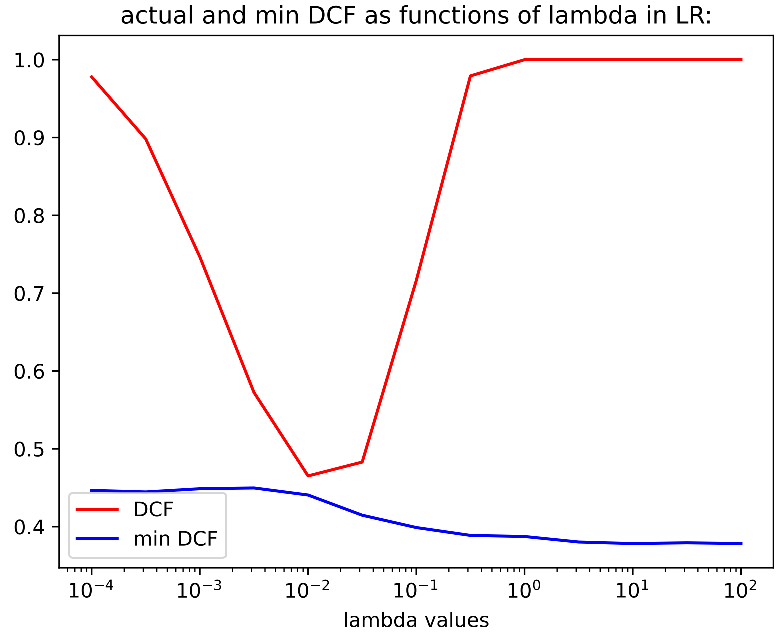

Figure 13: Actual and min DCF with respect to λ for linear LR with reduced training set

Considering now the full dataset again, the LR prior-weighted model is tested. It doesn't seem to be significant differences between the prior-weighted and non prior-weighted LR model. Moreover, the prior-weighted LR requires that the target prior is known when building the model.

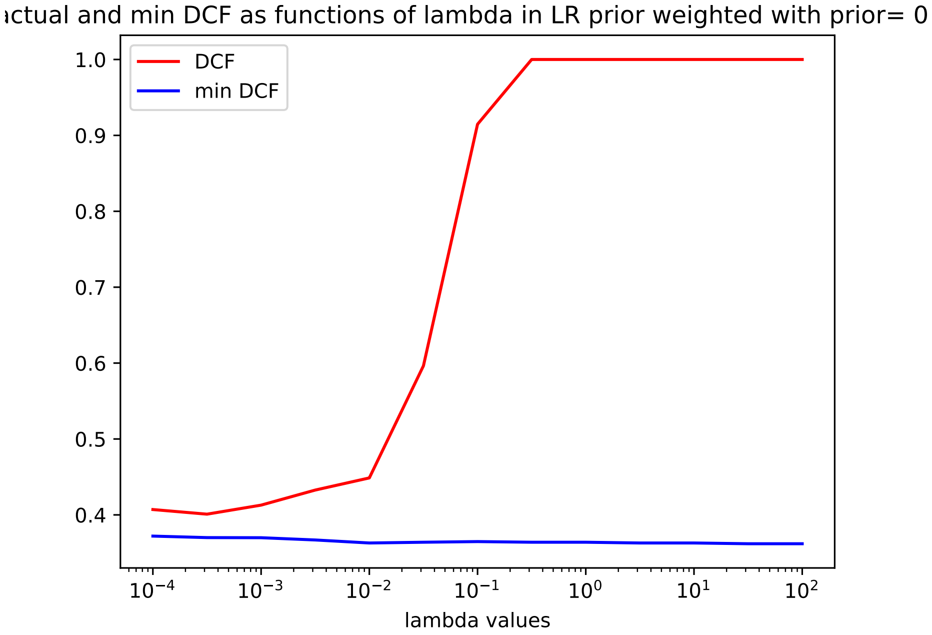

Figure 14: Actual and min DCF with respect to λ for linear prior-weighted LR

The analysis moves now to quadratic LR. The model shows a considerable improvement in both actual and min DCF. Also in this case, though, for higher values of λ there is a clear increase in actual DCF and just a slight increase in min DCF.

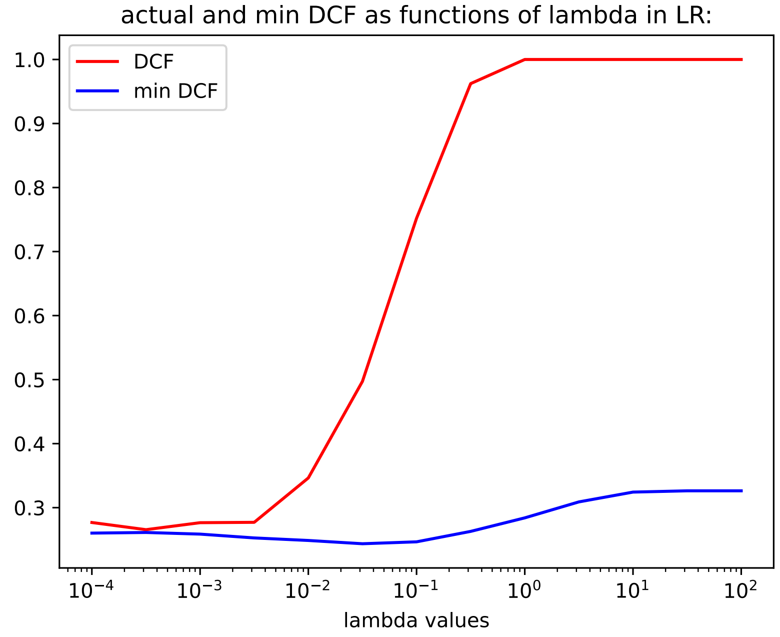

Figure 15: Actual and min DCF with respect to λ for quadratic LR

The non-regularized model is invariant to affine transformations of the data. However, once the regualrization term is introduced, affine trasnformations of the data can lead to different results. Is, thus reported the result of linear LR when centering the dataset with respect to the training set mean. The results are basically the same of the non-centered dataset as the original features were already almost standardized.

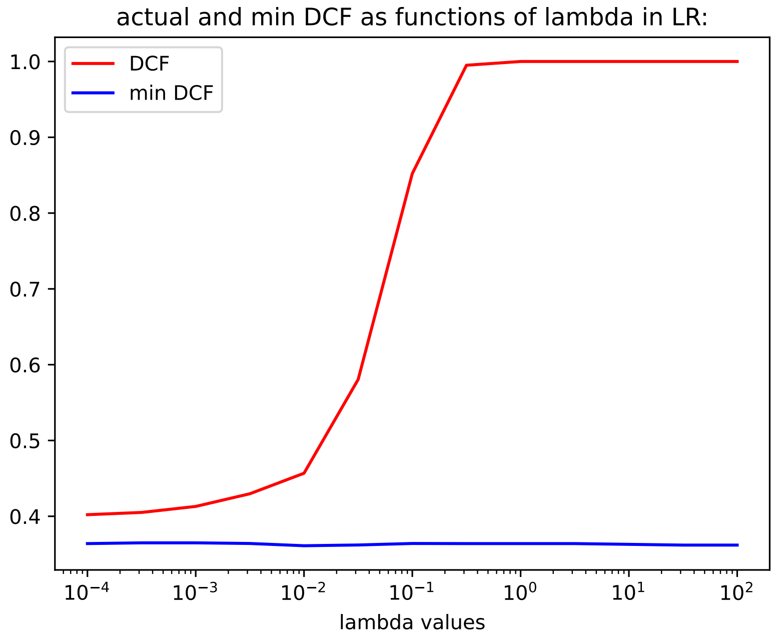

Figure 16: Actual and min DCF with respect to λ for linear LR with centered dataset

The best performing logistic regression model for the application prior $\tilde{\pi} = 0.1$ is the quadratic LR model with λ= 0.0316 with a min DCF of **0.244**.

Considering the Gaussian models analysed in the previous section, the best performing model in min DCF (and also actual DCF) is the Naive Bayes with no PCA. The Naive Bayes Gaussian classifier corresponds to a Multivariate Gaussian classifier with diagonal covariance matrices. Thus the decision function provided by the model is quadratic.

So both the two best models found so far provide a quadratic separation rule. The reason for this can be found in the fact that the relationship between the features and the target variable is non-linear as seen earlier. Moreover taking into account the covariance matrices of the features for each class can provide more accurate boundaries than linear models when different classes have distinct variance structures.

6 Support Vector Machines

In this section different variants of the SVM model are going to be tested on the data. The results are obtained using $\tilde{\pi} = 0.1$ as prior and fixing the value of the hyperparameter K to 1.

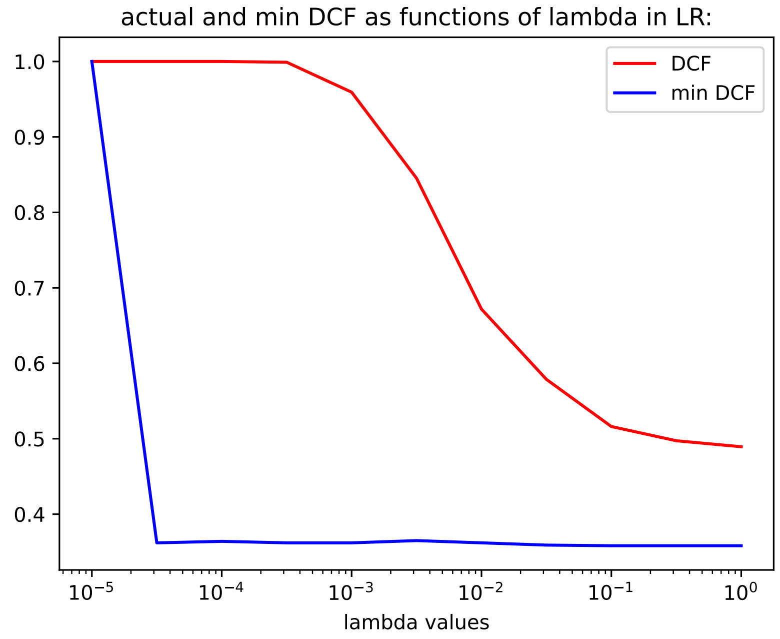

Figure 17: Actual and min DCF with respect to C for linear SVM

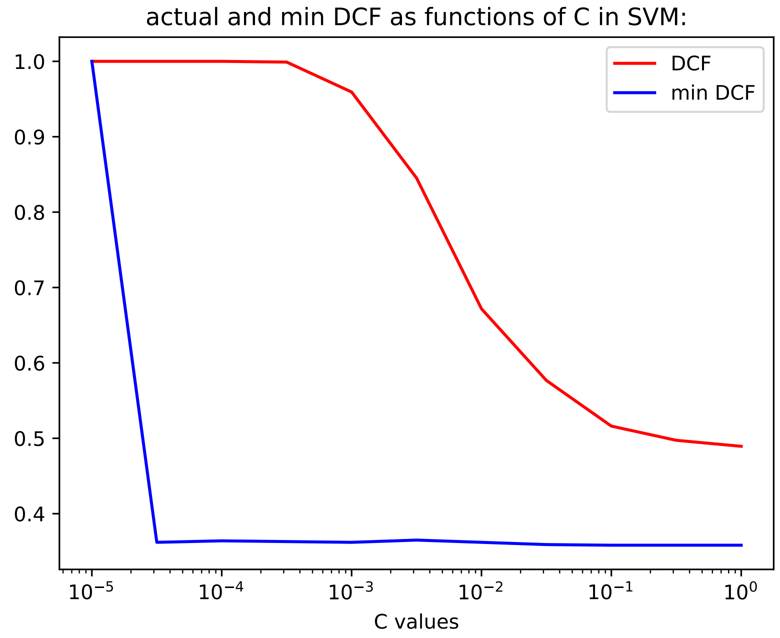

Figure 18: Actual and min DCF with respect to C for linear SVM with centered data

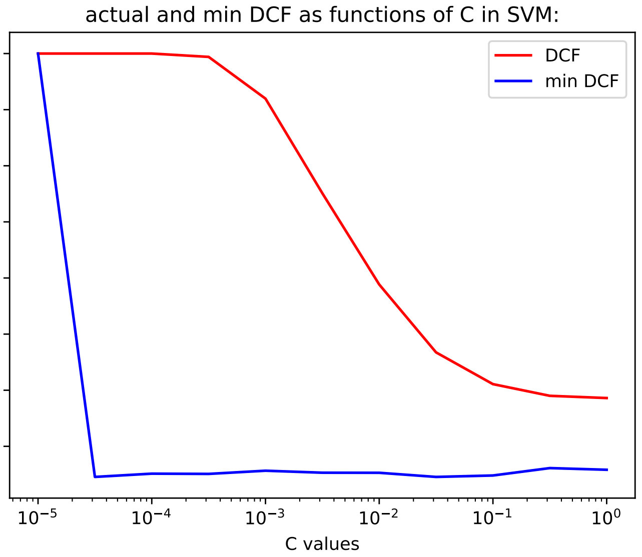

Figure 19: Actual and min DCF with respect to C for second order polynomial kernel SVM

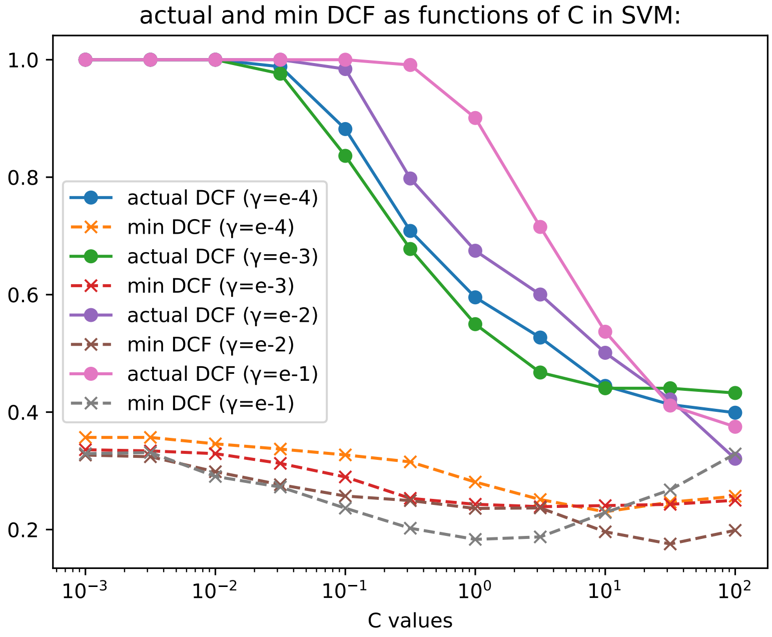

Figure 20: Actual and min DCF with respect to C and diferent values of γ for RBF kernel SVM

First the linear SVM model is analyzed 17. It can be seen that as the value of C grows (hence as the regularization gets weaker) the actual DCF gets gradually better while the min DCF remains constant after an initial quick decrease. This can be explained by the fact that too much regularization can cause underfitting thus leading to worse solutions so as the regularization decreases the actual DCF decreases. Moreover, the gap between between actual and min DCF is very marked especially for lower values of C thus showing poor calibrated scores (for the target application). Overall, the values of min DCF are not that promising but are in line with the results obtained for the other linear models analyzed so far (Tied gaussian and linear LR).

When considering centered data 18, no remarkable difference can be found.

Now the polynomial kernel SVM is considered 19 with hyperparameters c=1, d=2 (quadratic model) and, as before, different values of C. Polynomial kernel SVM with d=2 is equal to a quadratic model. Indeed the result for min DCF are similar to MVG and quadratic LR models. On the other hand, the actual DCF is much higher than the one for the before mentioned quadratic models thus showing a clear problem of score miss-calibration.

At last, the radial basis function (RBF) kernel SVM is analyzed considering a set of four values for the γ hyperparameter and different values of C as before. Also in this case, for greater values of C the actual DCF decreases. Nonetheless, the scores are not well calibrated also in this case. Overall this model is the one that provides the best min DCF value out of all SVM models configurations analyzed (and the best model so far in general) with **minDCF=0.175** for C=31.6 and $\gamma = e^{-2}$. The RBF kernel is particularly effective when the relationship between features and labels is non-linear. It can transform the data into a higher-dimensional space where a linear separation is possible. Indeed the RBF kernel is particularly effective when the relationship between features and labels is non-linear (as it was shown in this case) by transforming the data into a higher-dimensional space where a linear separation is possible. The problem of miss-calibration though, persists in this case too, showing a clear difference between actual and min DCF.

7 Gaussian Mixture Models

In this section the GMM models are going to be tested on the validation considering the usual application prior $\tilde{\pi} = 0.1$ and up to 32 components for each class.

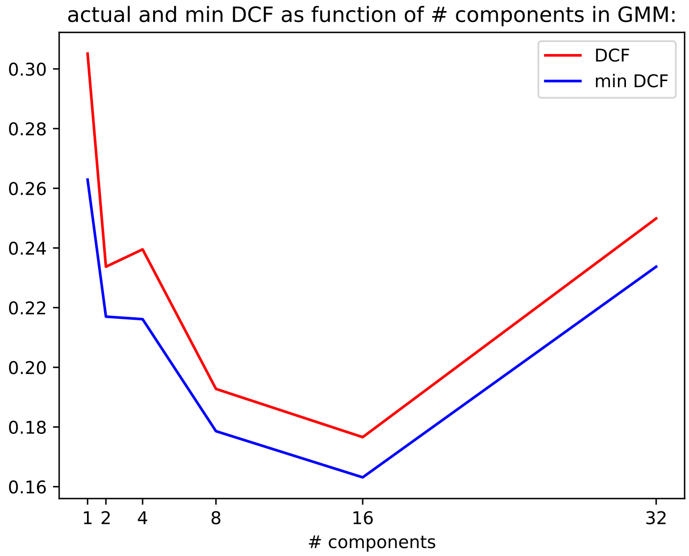

Figure 21: Actual and min DCF with respect to number of components for GMM

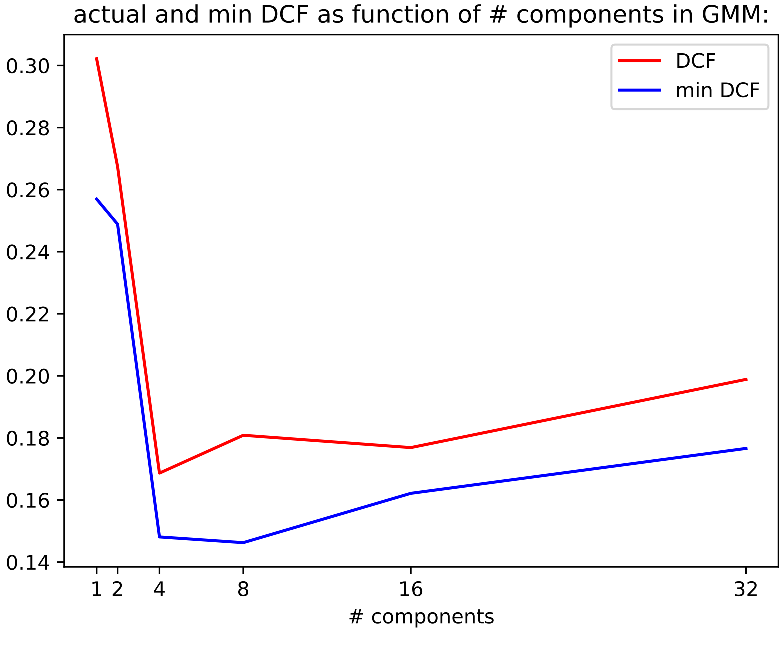

Figure 22: Actual and min DCF with respect to number of components for GMM with Diagonal covariance matrix for each component

It can be seen that both variants have worse performance as the number of components increases (worse for Full covariance GMM) leading to an overly complex model hence to overfitting. Overall the Diagonal GMM reaches not only a lower value of **min DCF = 0.146** and actual DCF =0.18 but it does so requiring a lower number of components (only 8 while the Full GMM requires 16 to reach its minimum). Overall the model seems quite well calibrated.

The fact that the diagonal variant has better results can also be seen in the fact that the features are kind of uncorrelated with each other within each component. Moreover, even though it might seems surprising that the diagonal model reaches its minimum (among the considered application and number of components analyzed) it can be understand by the fact that it requires far less parameters to optimize which makes it able to reach a good value of min DCF earlier than the Full GMM.

Now different combinations of number of components for the two classes in the Diagonal GMM are analyzed. The most relevant ones are the following (the first number represents the number of components for class Counterfeit and the second one represents the number of components for class Genuine. The considered prior is again the target prior):

models	minDCF	actDCF
GMM-Diag(8-32)	**0.131**	0.151
GMM-Diag(8-16)	0.132	**0.148**

Table 8: Actual and minimum DCF for Diagonal GMM with different components for each class

Both the reported models perform better than the previous one. In particular Diagonal GMM(8-32) has the best minimum DCF and Diagonal GMM(8-16) has the best actual DCF. It is clear that having more components for the Genuine class is improving the results, which could stem from a a difference in their characteristics.

The selected model for GMM is going to be the Diagonal GMM(8-16) since it has almost the same minDCF as GMM(8-32) but it has a better actual DCF and also less components for the Genuine class.

The assumption of Diagonal GMM models is that features are uncorrelated within each Gaussian component. This assumption seems to hold in this case, which can be found looking at the scatter plots.

8 Models comparison

In this section the best models from each category (LR, SVM and GMM) are going to be compared on different applications. First the best performing models for the application $\tilde{\pi}$ = 0.1 are shown in 9. Afterwards are shown the Bayes error plots of the models on different applications.

models	minDCF	actDCF
LR(quad)	0.244	0.497
SVM(RBF)	0.175	0.421
GMM(Diag,8-16)	**0.132**	**0.148**

Table 9: Comparison on min and actDCF for the best models from each approach

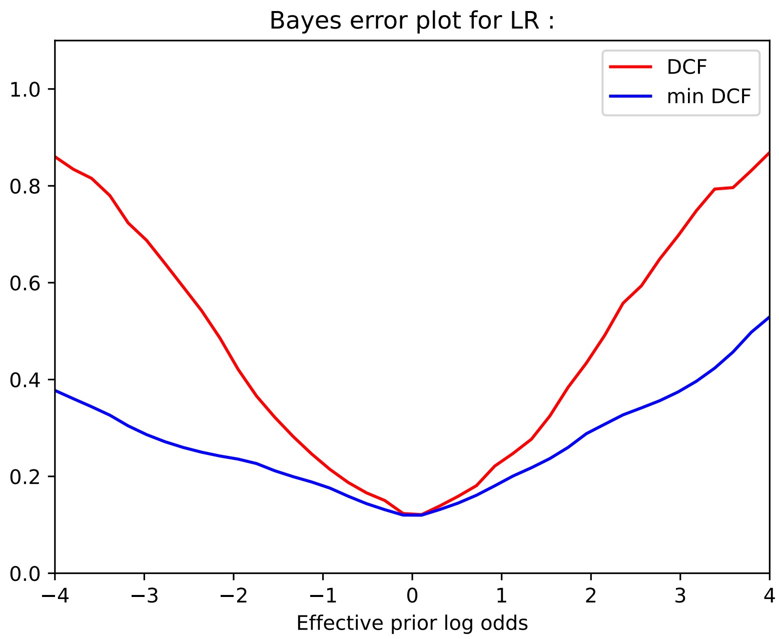

Figure 23: Bayes error plot for the best Logistic Regression model (quadratic)

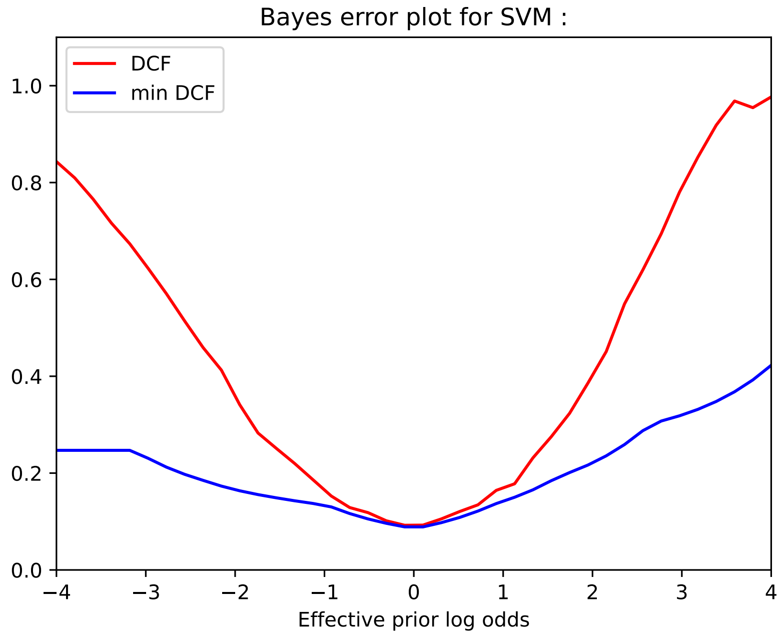

Figure 24: Bayes error plot for the best SVM model (RBF)

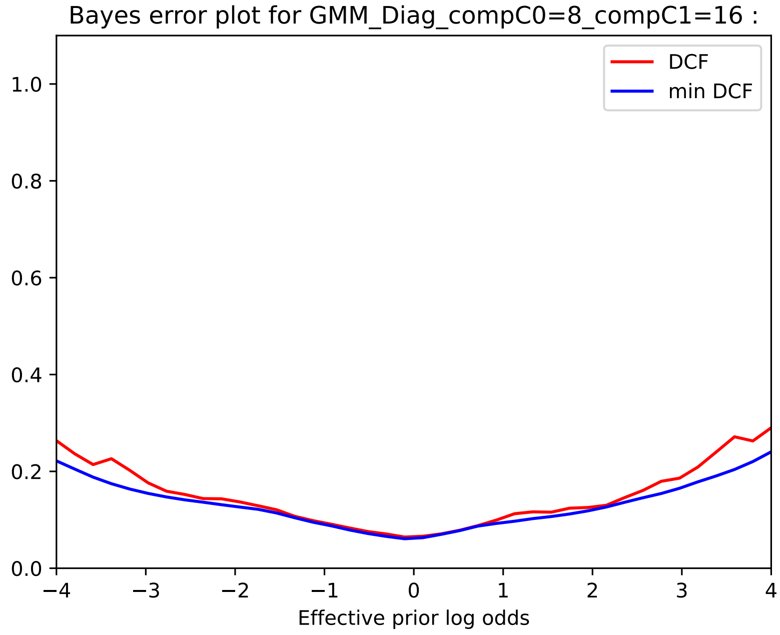

Figure 25: Bayes error plot for the best GMM(8-16) model (diagonal covariance)

It can be seen that, in terms of minimum DCF, the results are consistent with the relative ranking of the systems. Regarding the actual DCF, the SVM performs poorly and similarly to the LR model for some applications, thus showing a clear miscalibration. The GMM model instead, looks very well calibrated on most applications. This of course, makes sense considering that GMM scores are probabilistic considering that it's a generative model, hence leading to better calibrated scores. It doesn't seem like there are models that are harmful for some applciations, even though the SVM model has an actual DCF which is very close to 1 for a few operating points.

9 Calibration, Fusion and Evaluation

9.1 Calibration

In this section the effect of score calibration on the selected models are going to be analysed. The prior weighted logistic regression model is employed to learn a transformation which leads to better calibrated scores (the prior used is the application prior $\tilde{\pi} = 0.1$). K-fold is used to compute and evaluate the calibration transformation.

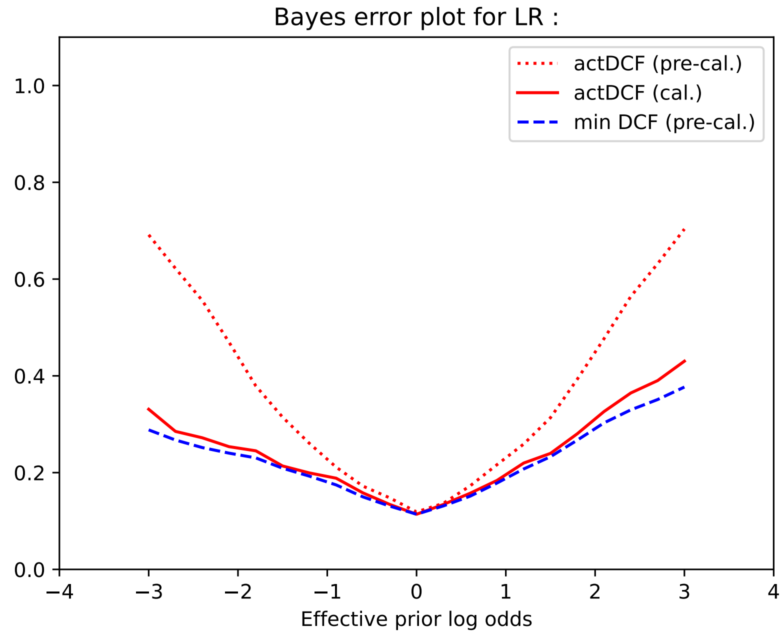

Figure 26: Bayes error plot for the calibrated selected LR model

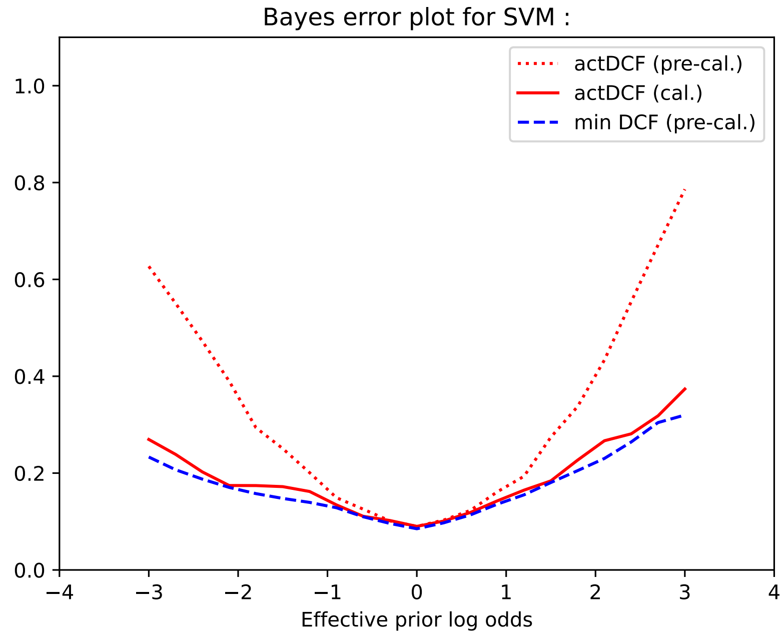

Figure 27: Bayes error plot for the calibrated selected SVM model

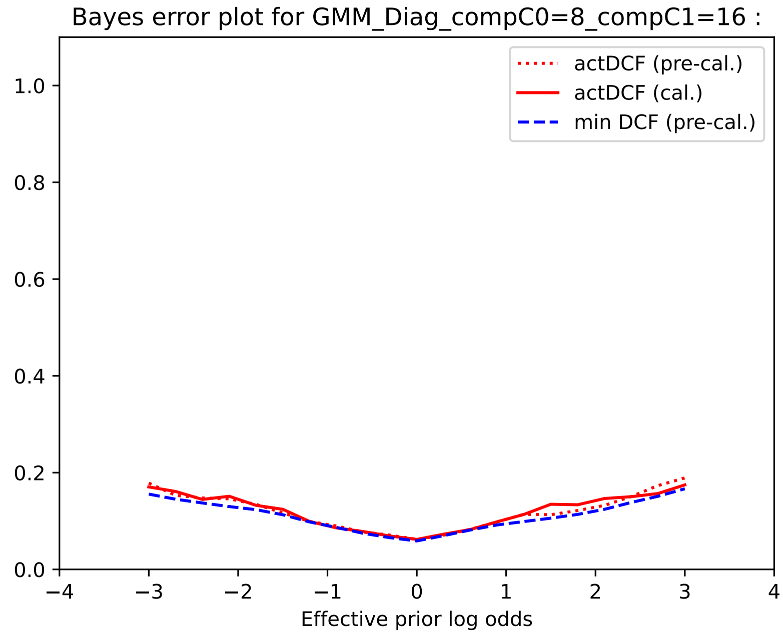

Figure 28: Bayes error plot for the calibrated selected GMM model

It can be seen that for all the selected models, the mis-calibration has vastly decreased over different ranges of applications thus leading to lower actual DCF values (maybe the least noticeable is the GMM which was already quite well calibrated).

9.2 Fusion

In this section the score-level fusion is going to be applied to the model previously selected for each approach.

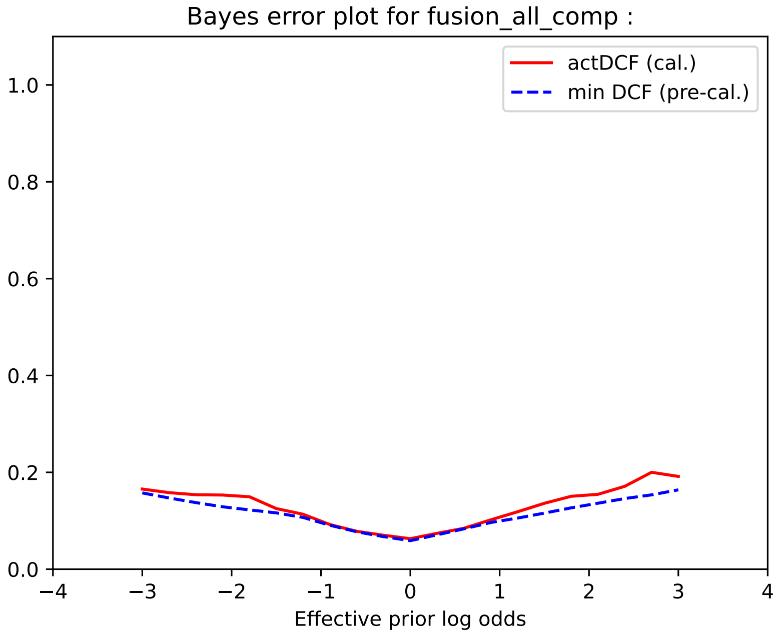

Figure 29: Bayes error plot for the calibrated fusion scores of the best model for each approach (LR,SVM,GMM)

The fusion not really improving the performance of the best single system. The actual DCF after calibration is, for the target application, **0.165**, while the actual DCF of the best system, Diagonal GMM(8-16) , is **0.148**. Nonetheless, the fused scores seems well calibrated, especially in the considered target application $\tilde{\pi} = 0.1$.

9.3 Evaluation

The ”delivered” system chosen to be used for application data is the Diagonal GMM with 8 components for class Counterfeit and 16 for class Genuine.

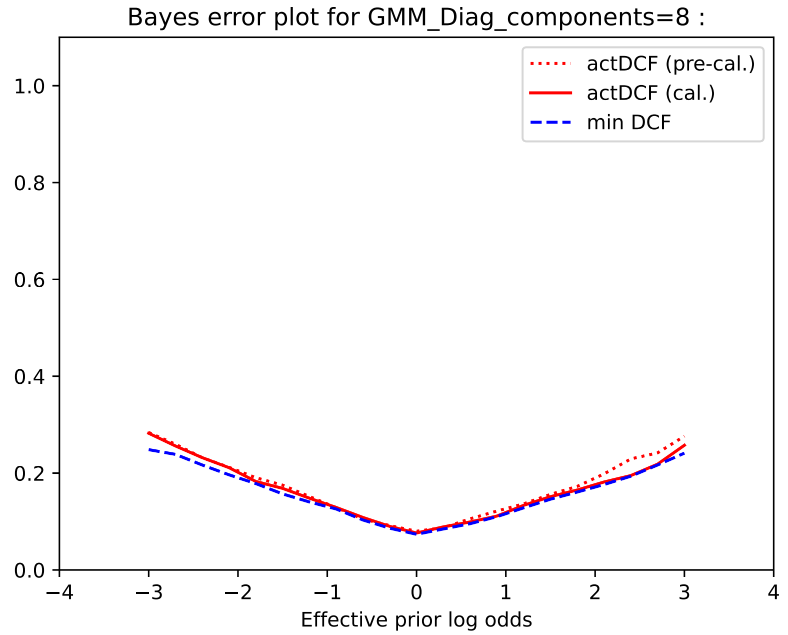

Figure 30: Bayes error plot for the calibrated Diagonal GMM(8-16) (delivered system) on the evaluation data

The scores seem very well calibrated along different ranges of applications including the target one.

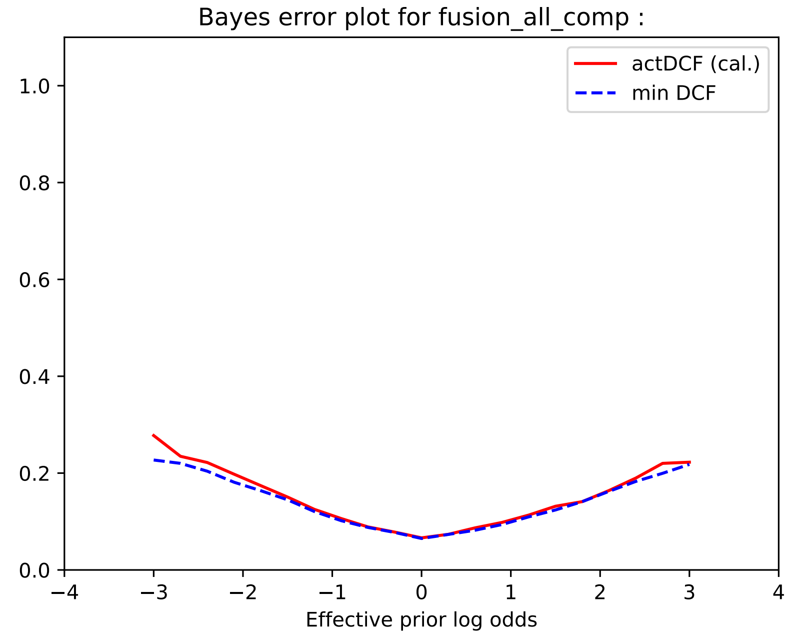

Figure 31: Bayes error plot for the calibrated fusion scores of the best model for each approach (LR,SVM,GMM) on evaluation data

The fusion of scores doesn't perform that well on evaluation data for the target application $\tilde{\pi} = 0.1$, providing an actual DCF of $\mathbf{0.203}$.

Now, we look at how, the best models from the other approach (LR,SVM), perform on evaluation data.

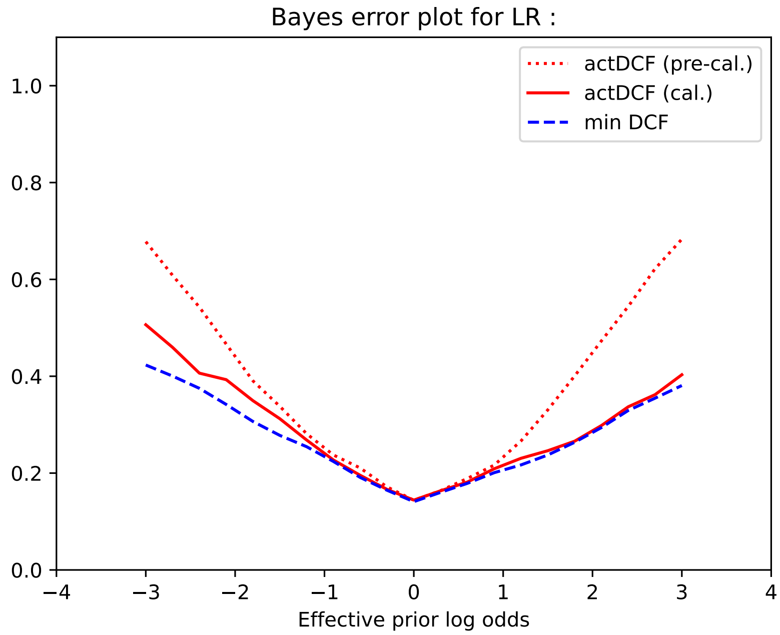

Figure 32: Bayes error plot for the calibrated Quadratic LR on the evaluation data

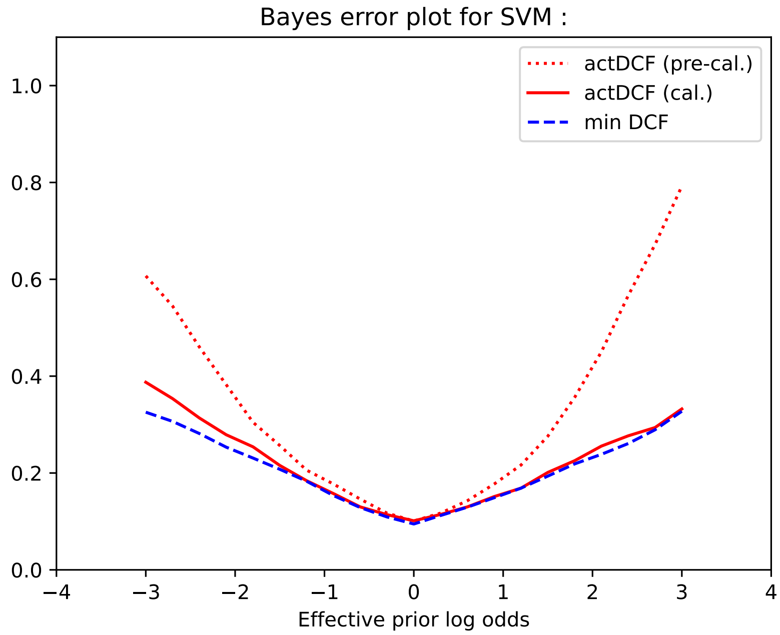

Figure 33: Bayes error plot for the calibrated RBF SVM on the evaluation data

Overall, it seems like the calibration strategy was effective for all the approaches and for a lot of different applications.

Now, different variants of the selected model (GMM(8-16) diagonal) are going to be tested on the evaluation data to analyze wheter the training strategy was effective (only the most intresting results are reported).

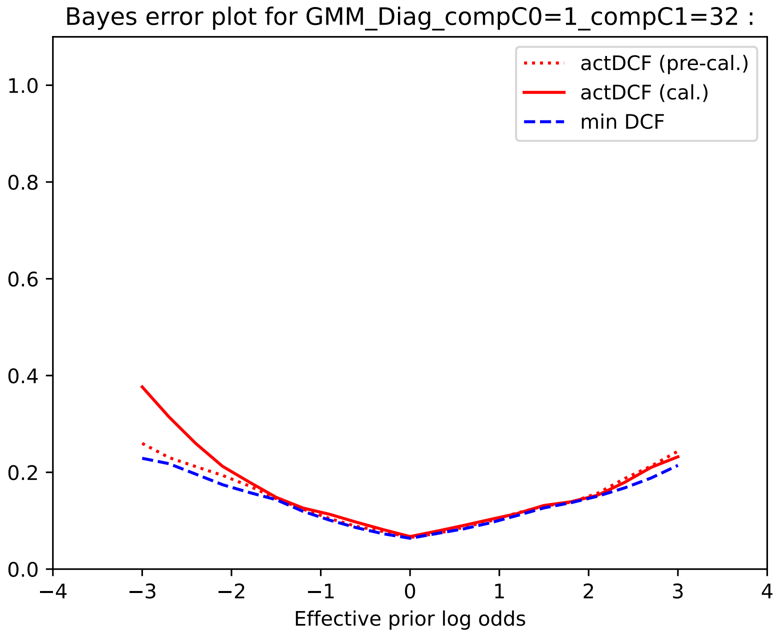

Figure 34: Bayes error plot for the calibrated Diagonal GMM(1-32) on the evaluation data

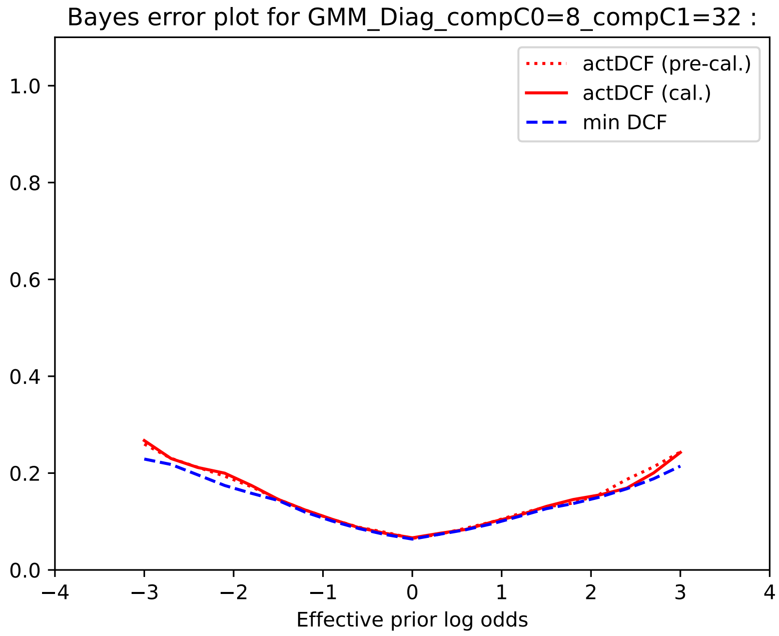

Figure 35: Bayes error plot for the calibrated Diagonal GMM(8-32) on the evaluation data

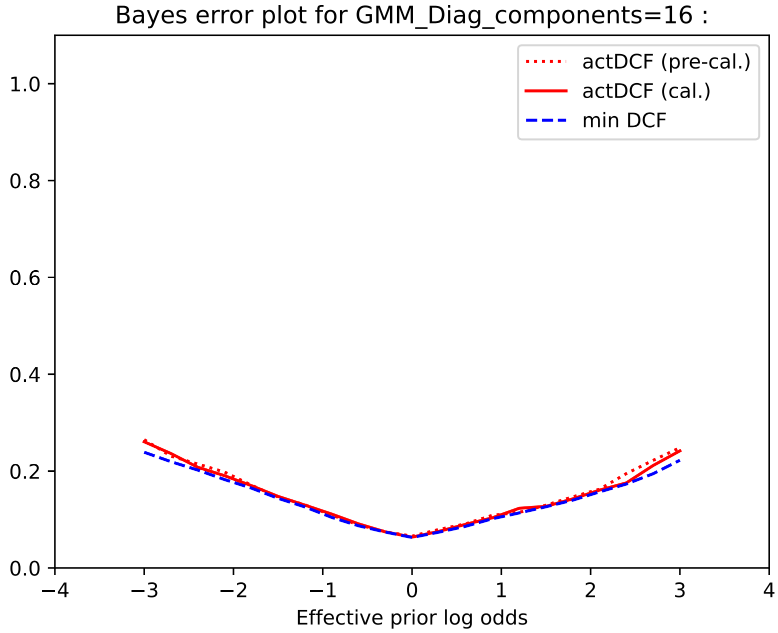

Figure 36: Bayes error plot for the calibrated Diagonal GMM(16-16) on the evaluation data

The reported models are those that perform the best among all the analysed variants.

models	minDCF	actDCF
GMM-Diag-8-16	0.134	0.139
GMM-Diag-8-32	0.134	0.138
GMM-Diag-1-32	0.134	0.143
GMM-Diag-16-16	0.136	0.137

Table 10: Actual and minimum DCF for guassian models and different applications

It can be seen that the best performing model is the Diagonal GMM with 16 components for both classes. The scores look well calibrated for all the systems. Thus, the chosen model (Diagonal GMM(8-16), was not the optimal one even though it's still not that far from the best.

Overall, the training choices seem to have been reasonable, leading to a good enough model even though there could have been better options.