Wine Quality Prediction

Author: Andrea Cavallo
Student ID: 287965
Course: Machine Learning and Pattern Recognition

July 6, 2021

Abstract

The report analyses the Wine Quality Prediction task, using a dataset from the UCI repository containing features about the Portuguese *Vinho Verde* wine. The goal is to predict whether the wine has a good or a bad quality. In the original dataset, qualities are ranked from 0 to 10, whereas for this task the quality has been transformed into a binary attribute (low quality if lower than 6, high quality if higher than 6). Several models are applied and their performances are compared. The effectiveness of the selected model is then evaluated on a test dataset.

1 Data visualization

First of all, some characteristics of the features contained in the training dataset are analyzed. The training set contains 1226 bad quality samples and 613 good quality samples, so classes are not balanced. The dataset contains 11 features, representing physical properties of the wines. Figure 1 shows the distributions of raw features. It can be observed that features do not have zero mean, so Z-normalization can be applied as a pre-processing step. Moreover, distributions do not generally have a Gaussian shape, which may be due to the presence of outliers.

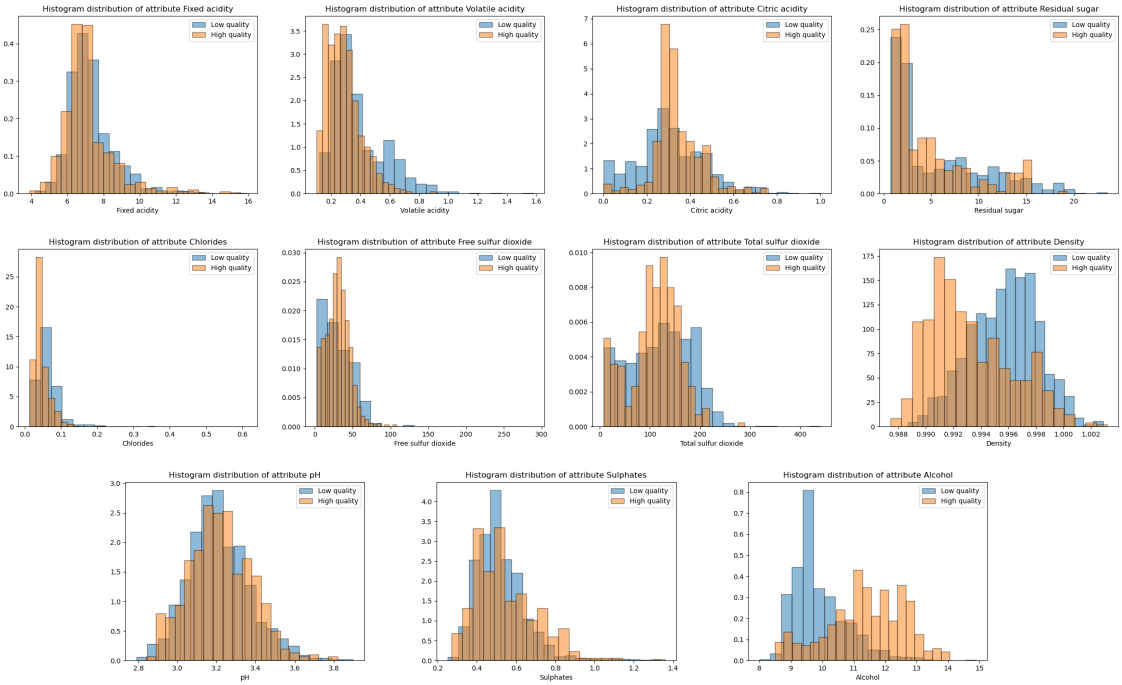


Figure 1: Distributions of raw features

To face this issue, features can be pre-processed with Gaussianization. Gaussianization consists in mapping the features to values with uniform distribution and then transforming them through the inverse of the Gaussian cumulative distribution function. First, the rank of a feature x over the training set is computed:

$$r(x) = \frac{\sum_{i=1}^{N} I[x < x_i] + 1}{N + 2}$$

where x_i is the value of the considered feature for the i-th sample. Then, the transformed feature is computed as

$$y = \Phi^{-1}(r(x))$$

where Φ is the inverse of the cumulative distribution function (percent point function). In Figure 2 it is possible to observe that Gaussianization provides a relevant difference in the distribution of some of the features.

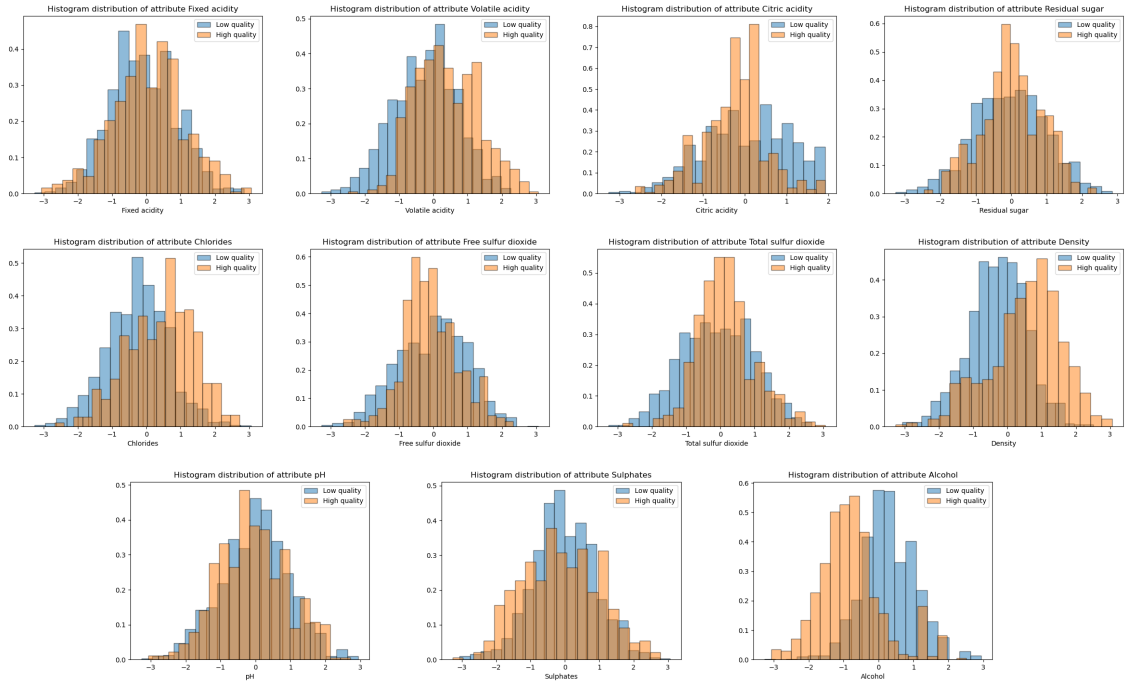
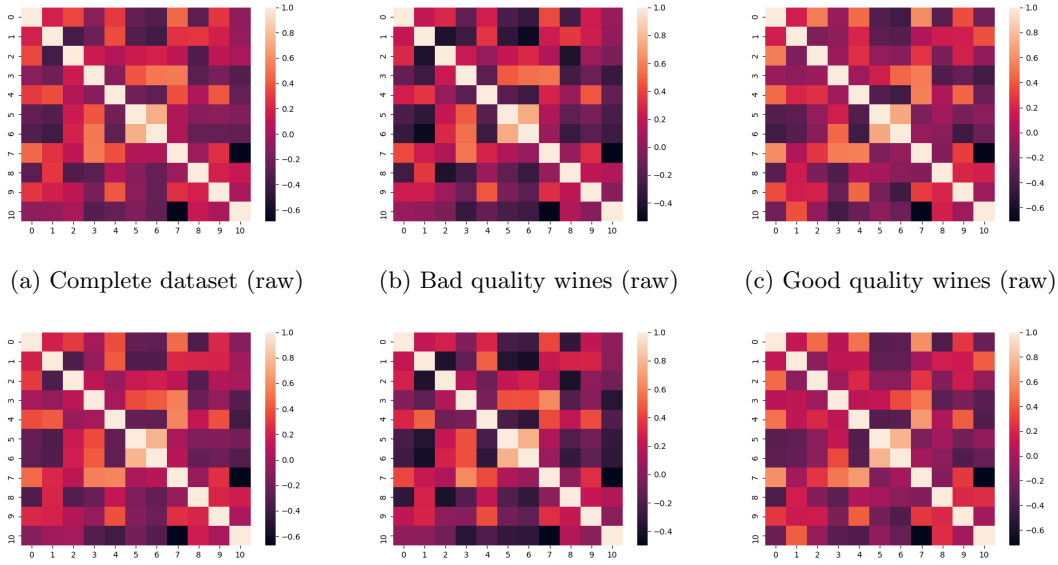


Figure 2: Distributions of Gaussianized features

Then, correlation among features is analysed. In particular, Figure 3 shows the Pearson correlation coefficient among raw and Gaussianized features in the complete dataset and also for specific classes. It can be observed that some features appear to be correlated (although not strongly). Therefore, PCA can be tested as a way to retain only uncorrelated dimensions.



(a) Complete dataset (raw) (b) Bad quality wines (raw) (c) Good quality wines (raw)

(d) Complete dataset (Gauss (e) Bad quality wines (Gauss) (f) Good quality wines (Gauss)

Figure 3: Correlation of features

2 Classifiers

2.1 Assessing the most promising models

After an initial analysis of the features of the dataset, several classifiers are tested on the training data and their performance is evaluated in terms of minimum Detection Cost Function, *id est* the best achievable result in case of optimal threshold selection.
Initially, the models are trained and evaluated using 5-fold cross validation and also using a single split (training is faster, but there are less data for validation and model training). The application of interest is a uniform prior one: $(\tilde{\pi}, C_{fp}, C_{fn}) = (0.5, 1, 1)$.

2.1.1 Gaussian models

The first classifiers to be tested on the training dataset are different kinds of Gaussian classifiers. In particular, Table 1a reports the minimum detection cost function for a full-covariance model, a naive Bayes model, a tied full-covariance model and a tied naive Bayes model. The models are tested on raw and Gaussianized features, and dimensionality reduction is applied. The results are evaluated both using 5-fold cross validation and on a single split (80% of the dataset for training, 20% for evaluation).
It can be observed that the best performance is achieved by the full-covariance model, which can be explained by observing that the dataset is large enough to properly estimate an entire covariance matrix for each class, with no need of simplifying assumptions such as the ones that diagonal and tied models do. Additionally, the better performance of full-covariance Gaussian models can be justified by the fact that they generate quadratic separation surfaces, whereas tied-covariance models generate linear separation surfaces, which are less adaptable to the real distributions of data. Diagonal covariance models, on the other hand, provide the worst performance, which may indicate that the hypothesis of uncorrelated components does not hold in this case. Moreover, it can be noticed that Gaussianization is helpful (even though some models perform similarly also on raw features) and PCA = 10 provides an improvement with respect to the complete set of features.

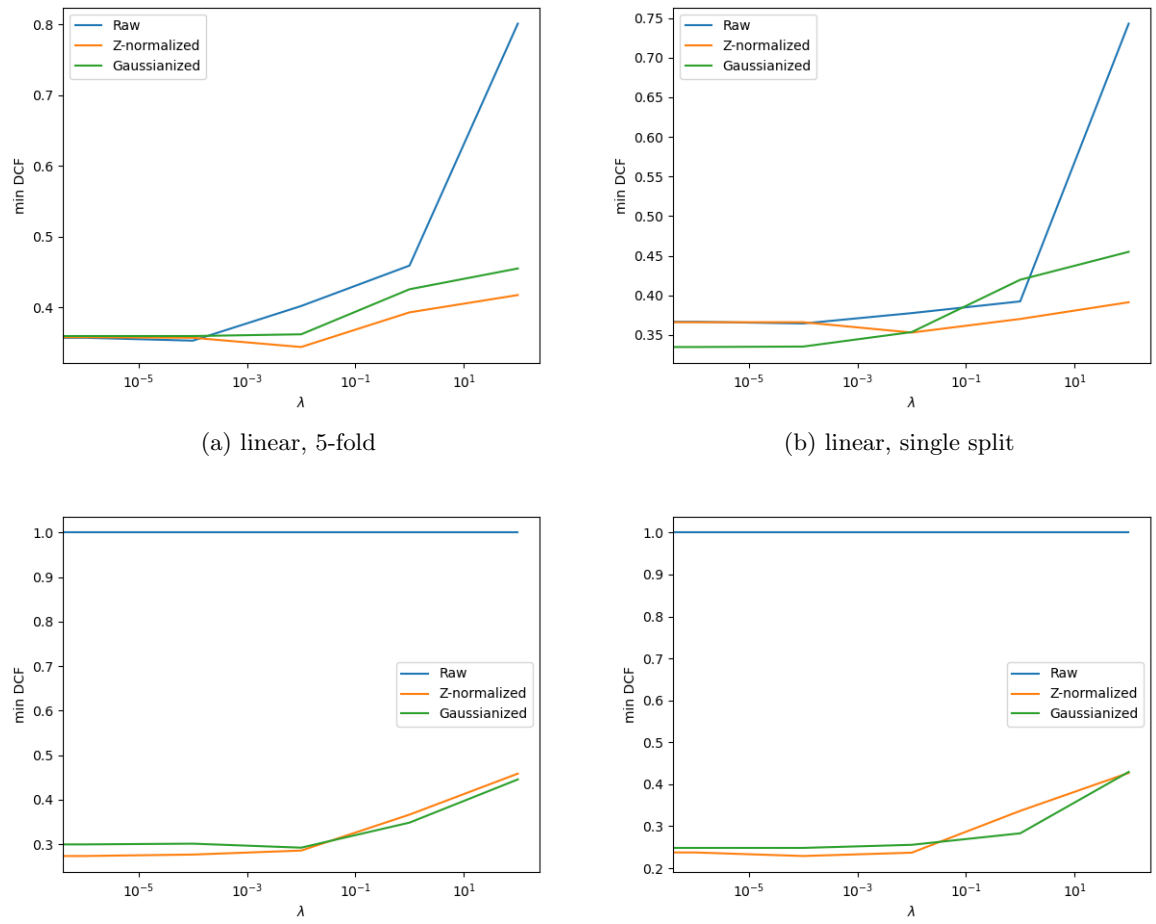
Model	**Single split** $\tilde{\pi} = 0.5$	**5-fold** $\tilde{\pi} = 0.5$
Gaussianized features - no PCA		
Full-Cov	0.255	0.301
Diag-Cov	0.460	0.439
Tied Full-Cov	0.344	0.345
Tied Diag-Cov	0.441	0.442
Gaussianized features - PCA = 10		
Full-Cov	**0.242**	**0.291**
Diag-Cov	0.388	0.400
Tied Full-Cov	0.359	0.350
Tied Diag-Cov	0.362	0.352
Gaussianized features - PCA = 9		
Full-Cov	0.256	0.303
Diag-Cov	0.392	0.405
Tied Full-Cov	0.361	0.354
Tied Diag-Cov	0.362	0.356
Raw features - no PCA		
Full-Cov	0.264	0.312
Diag-Cov	0.409	0.420
Tied Full-Cov	0.327	0.336
Tied Diag-Cov	0.391	0.403

(a) Gaussian

Model	**Single split** $\tilde{\pi} = 0.5$	**5-fold** $\tilde{\pi} = 0.5$
Gaussianized features - no PCA		
Log-Reg ($\lambda = 0$)	0.335	0.360
Quad-Log-Reg ($\lambda = 10^{-2}$)	0.256	0.292
Z-normalized features - no PCA		
Log-Reg ($\lambda = 10^{-2}$)	0.353	0.344
Quad-Log-Reg ($\lambda = 0$)	**0.237**	**0.273**
Z-normalized features - PCA = 10		
Log-Reg ($\lambda = 10^{-2}$)	0.353	0.345
Quad-Log-Reg ($\lambda = 0$)	0.243	0.282
Raw features - no PCA		
Log-Reg ($\lambda = 10^{-2}$)	0.364	0.353
Quad-Log-Reg ($\lambda = 0$)	1.000	1.000

(b) Logistic Regression

Table 1: Min DCF of Gaussian and Logistic Regression models



(a) linear, 5-fold

(b) linear, single split

(c) quadratic, 5-fold

(d) quadratic, single split

Figure 4: min DCF for different values of λ

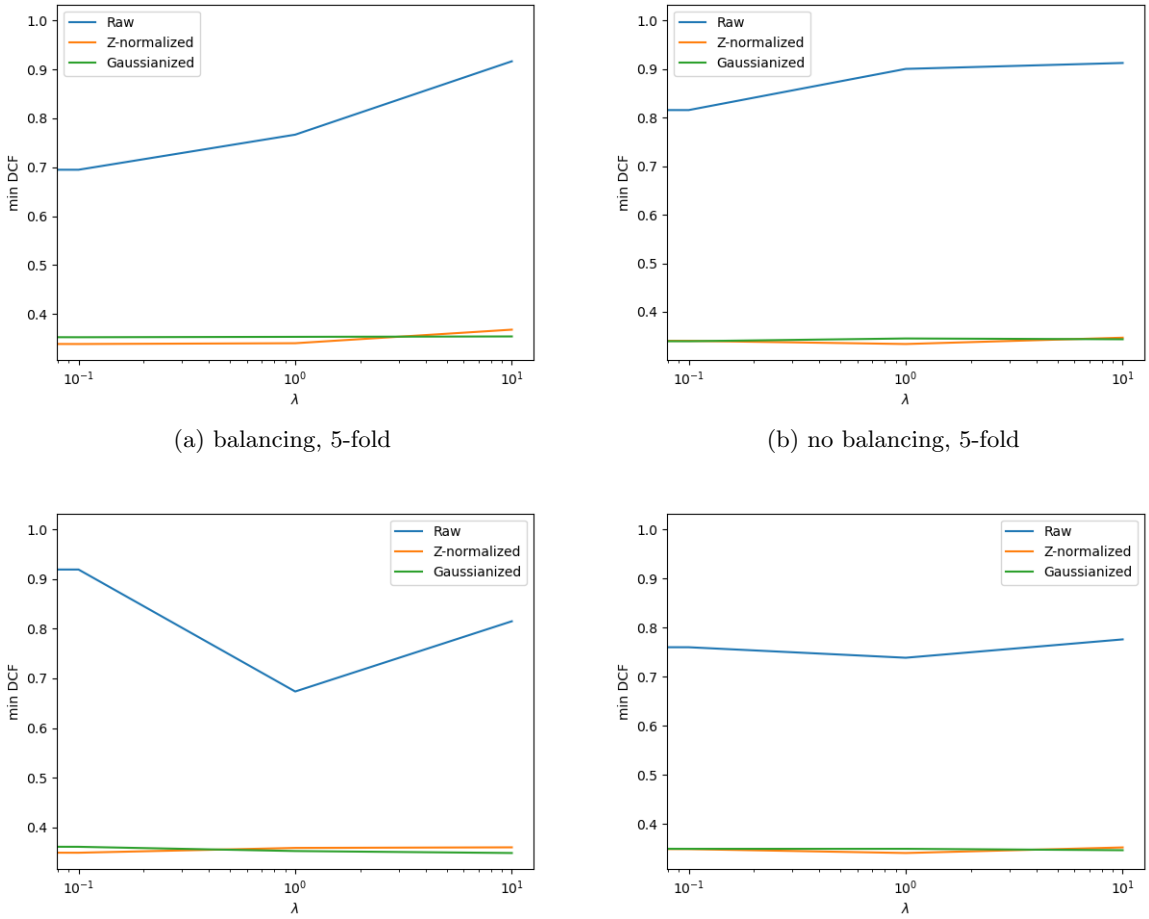
2.1.2 Logistic Regression

Then, logistic regression models are tested. Since classes are unbalanced, their costs are rebalanced. Thus, the loss function becomes:

$$J(w,b) = \frac{\lambda}{2}\|w\|^2 + \frac{\pi_T}{n_T}\sum_{i=1|c_i=1}^{n}\log(1+e^{-z_i(w^T x_i+b)}) + \frac{1-\pi_T}{n_F}\sum_{i=1|c_i=0}^{n}\log(1+e^{-z_i(w^T x_i+b)})$$

The application to be tested is $\pi_T = \frac{1}{2}$. To estimate the hyperparameter λ, different values are tested, using cross-validation and a fixed evaluation set. Figure 4 shows the relationship between min DCF of different logistic regression models and different evaluation strategies and the hyperparameter λ. It can be observed that the choice of λ has a significant impact on the performance. Table 1b shows, then, the minimum Detection Cost Function for logistic regression models on differently pre-processed data, at the best value of λ between those tried. Both linear and quadratic linear regression models are tested. Quadratic logistic regression on raw features does not provide relevant results because of numerical errors (an overflow occurs during model training).
It can be observed that Gaussianization is not helpful in this case, because the models adapt well also to features whose distribution is not Gaussian. However, feature preprocessing (Z-normalization) is useful to avoid numerical errors and it also improves the min DCF. Quadratic models perform better, since their separation surfaces are more complex and can describe better the real distribution of data, without the risk of overfitting (data are enough to properly estimate all parameters). In fact, linear logistic regression models have a performance that is comparable to tied-covariance Gaussian models, since they both provide linear separation surfaces, whereas quadratic logistic regression performs similarly (but better) with respect to full-covariance Gaussian models. Moreover, it is possible to notice that PCA does not bring any relevant advantage, although it is not detrimental. This appears reasonable, since the samples of the dataset do not have many dimensions and the models can benefit from having all of them.

Therefore, PCA will not be used in the following part of the analysis.



(a) balancing, 5-fold

(b) no balancing, 5-fold

(c) balancing, single split

(d) no balancing, single split

Figure 5: linear SVM, min DCF for different values of C

2.1.3 SVM

The next models to be tested for the task are different flavors of SVMs. In particular, the first one is the linear SVM, even if the previous results have shown that linear models are likely to produce worse results with respect to more complex models.
Since classes are unbalanced, also class rebalancing is tested. In particular, two different values of C (boundary on α_i in the dual formulation of the SVM problem) are used for the two classes:

$$C_T = C\frac{\pi_T}{\pi_T^{emp}} \qquad C_F = C\frac{1-\pi_T}{\pi_F^{emp}}$$

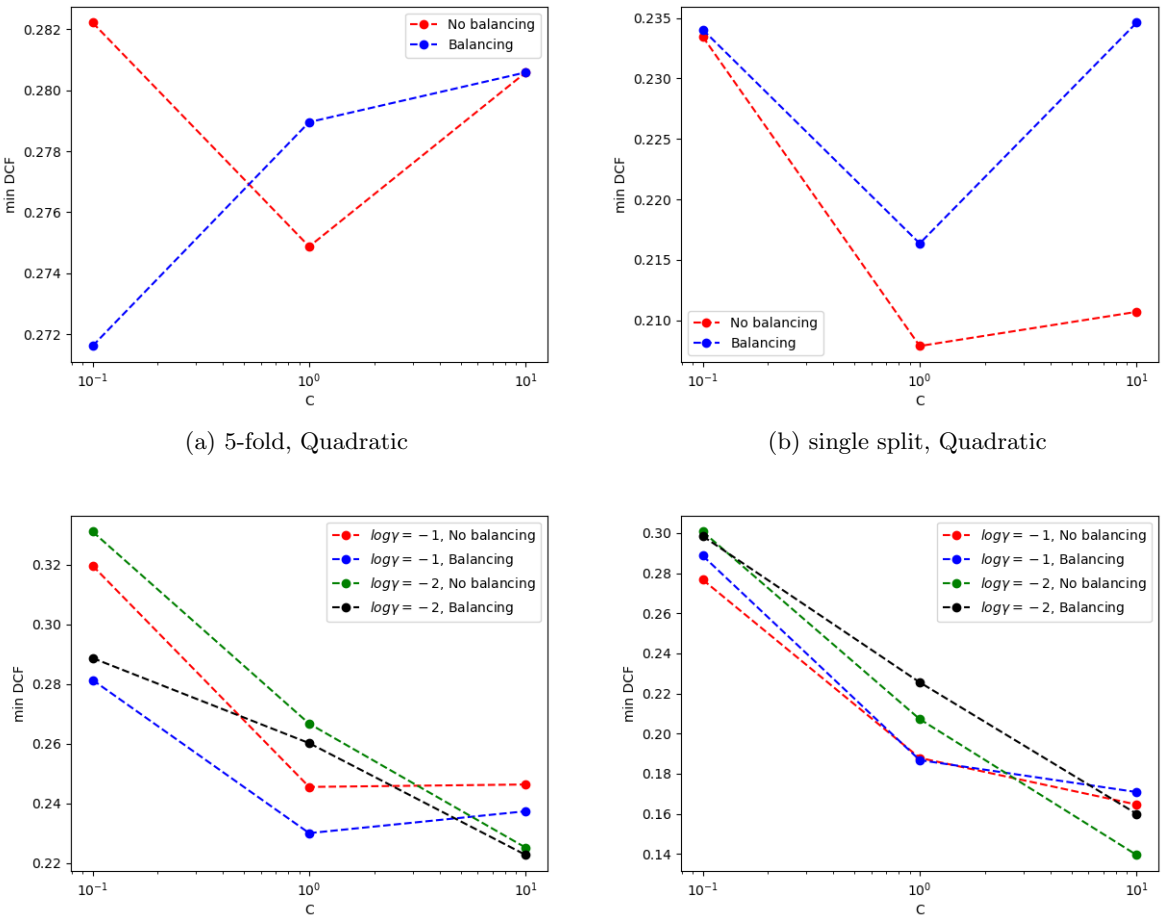
where π_T^{emp} and π_F^{emp} are the empirical priors for the two classes over the training set. Different values of the hyperparameter C are tested using 5-fold cross validation and a fixed evaluation set, and results can be seen in Figure 5. It can be noticed that the choice of C does not appear decisive with respect to the min DCF. Table 2 shows the results for the best choice of C on different data.
It can be observed that feature normalization is very helpful (performance on raw features is very poor), whereas Gaussianization does not bring any advantage. Therefore, for the following SVM models, only Z-normalized features will be used. Moreover, class rebalancing is not advantageous in this case. The performance is comparable to that of the previous models using linear separation surfaces (linear logistic regression and tied-covariance Gaussian models).

Since logistic regression models with quadratic separation surfaces have provided good results, it is reasonable to expect a similar performance from quadratic kernel SVM models, as they both provide more complex rules for class separation with respect to linear models. In addition, also RBF kernel SVM

Model	Single split $\tilde{\pi} = 0.5$	5-fold $\tilde{\pi} = 0.5$
Gaussianized features - no PCA		
SVM ($C = 0.1$) - no rebalancing	0.339	0.350
SVM ($C = 0.1$) - rebalancing	0.352	0.360
Z-normalized features - no PCA		
SVM ($C = 1$) - no rebalancing	**0.334**	**0.341**
SVM ($C = 0.1$) - rebalancing	0.338	0.348
Raw features - no PCA		
SVM ($C = 0.1$) - no rebalancing	0.816	0.760
SVM ($C = 0.1$) - rebalancing	0.695	0.920

Table 2: Min DCF of different linear SVM models

are tested. To tune the different hyperparameters (C for the quadratic kernel SVM, C and γ for the RBF kernel SVM), different values are tested (joint optimization), and the results can be seen in Figure 6. It can be observed that the results for RBF are quite similar either using class rebalancing or not. The choice of the hyperparameters appears important in both cases.



(a) 5-fold, Quadratic

(b) single split, Quadratic

(c) 5-fold, RBF

(d) single split, RBF

Figure 6: kernel SVM, min DCF for different values of C

Table 3 shows, then, the results for the best choice of hyperparameters on Z-normalized data. It can be noticed that the quadratic kernel SVM performs very similarly to the quadratic logistic regression, as expected. However, the best result is achieved by the RBF kernel SVM, which significantly outperforms

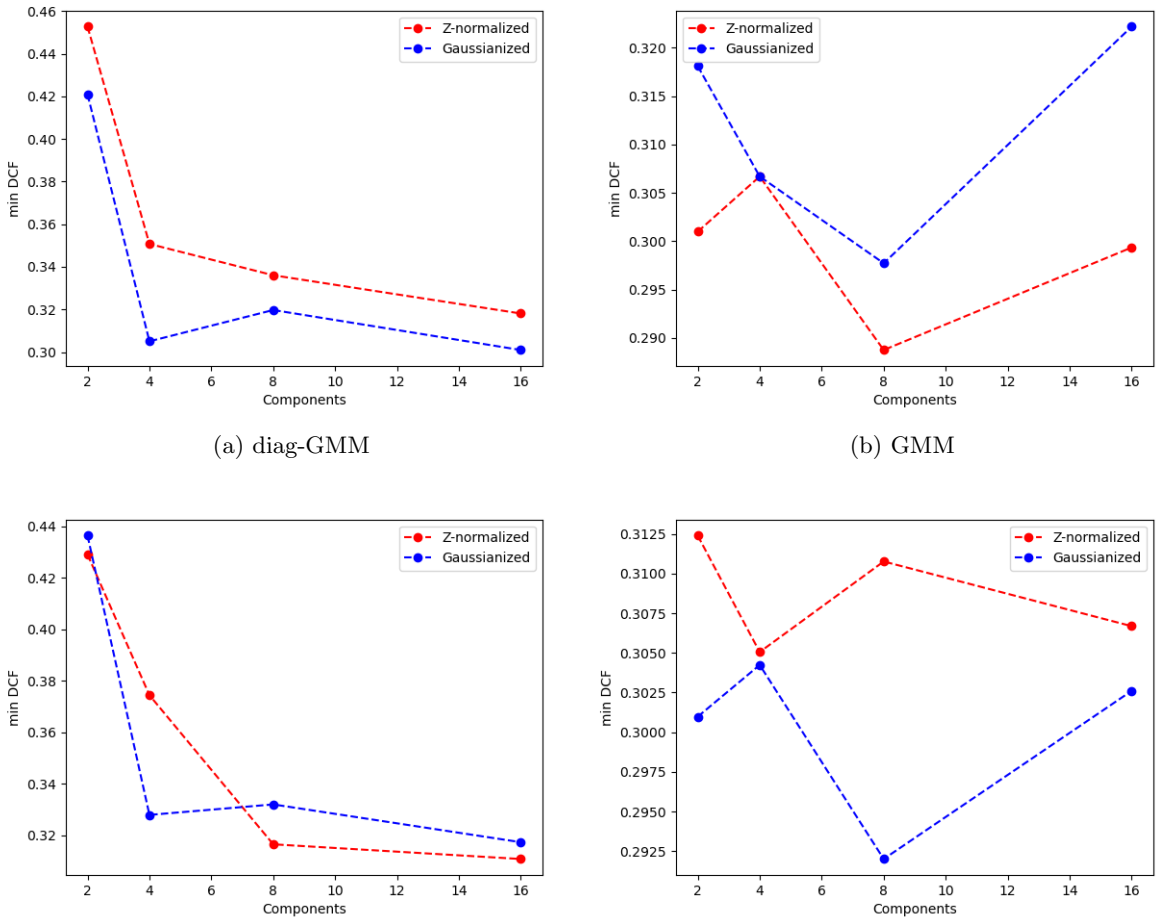
the other models. Moreover, it can be observed that class rebalancing is not significantly beneficial in this case, although it is not detrimental.

Model	**Single split** $\tilde{\pi} = 0.5$	**5-fold** $\tilde{\pi} = 0.5$
Z-normalized features - no PCA		
Quad-SVM ($C = 1$) - no rebalancing	0.208	0.275
Quad-SVM ($C = 0.1$) - rebalancing	0.234	0.272
RBF-SVM ($C = 10, log\gamma = -2$) - no rebalancing	**0.140**	0.225
RBF-SVM ($C = 10, log\gamma = -2$) - rebalancing	0.160	**0.223**

Table 3: Min DCF of different quadratic and RBF kernel SVM models

2.1.4 GMM

The final models to be tested are GMMs. The expectation is for them to perform better than standard Gaussian models, as they can approximate generic distributions. Different variants are analysed: standard GMM, GMM with diagonal covariance matrixes and tied-covariance GMM (where the covariance matrix is tied for each class, but different classes may have different covariance matrixes).
The optimal number of components is selected through cross-validation, as it allows a more detailed evaluation of the performance of the model with respect to a single split. Figure 7 shows the results for different numbers of components. For diagonal models, the increasing number of components seems to regularly decrease the min DCF, whereas for the other models the pattern is less regular.



(a) diag-GMM

(b) GMM

(c) tied-diag-GMM

(d) tied-GMM

Figure 7: min DCF for different GMM models

Model	5-fold $\tilde{\pi} = 0.5$
Z-normalized features - no PCA	
tied-GMM (4 components)	0.306
tied-diag-GMM (16 components)	0.311
GMM (8 components)	**0.289**
diag-GMM (16 components)	0.318
Gaussianized features - no PCA	
tied-GMM (8 components)	0.292
tied-diag-GMM (16 components)	0.317
GMM (8 components)	0.298
diag-GMM (16 components)	0.301

Table 4: Min DCF of different GMM models

Then, Table 4 shows the results for the best choice of components for each model. The analysis is performed both on Z-normalized features and on Gaussianized features. It can be observed that Gaussianization is not helpful. The model providing the best performance is the standard GMM, as there are enough data to properly tune all parameters. However, also the tied model using Gaussianized features has a very similar performance. Although it is likely for models with more components to provide better results due to their higher capability to approximate complex distributions, they were not tested, due to their computational complexity.

2.1.5 Conclusion

Overall, it can be concluded that the most promising model is the RBF kernel SVM with hyperparameters $C = 10$, $\log \gamma = -2$, using rebalancing and Z-normalized features. However, also the logistic regression with quadratic separation surfaces ($\lambda = 0$, Z-normalized features, no PCA) and the GMM (8 components, Z-normalized features) provide good results. Therefore, all three will be taken into consideration in the following parts. Decisions are made considering the results of cross validation, since it is expected to provide more accurate results. However, the results achieved with a fixed evaluation set agree with those of cross validation in most cases.

2.2 Selecting the best threshold

After selecting the most promising models based on their mininum Detection Cost Functions, it is necessary to evaluate how they perform with the optimal theoretical threshold $t = -log\frac{\tilde{\pi}}{1-\tilde{\pi}}$. Table 5 shows how this affects the DCF using cross validation. In particular, the optimal estimated threshold is estimated separately for each iteration of the k-fold algorithm. It can be noticed that there is a loss of performance in all cases, although it is not huge. This is reasonable since the tested models are not probabilistic, and therefore they do not account for characteristics in the data that may lead to uncalibrated scores.

Model	5-fold $\tilde{\pi} = 0.5$			
	min DCF	act DCF ($t = -log\frac{\tilde{\pi}}{1-\tilde{\pi}}$)	act DCF (t^*)	calibrated
RBF-SVM	0.223	0.232	0.228	0.232
Quad-LR	0.273	0.285	0.276	0.290
GMM (8 components)	0.289	0.301	0.294	0.312

Table 5: min DCF vs actual DCF of selected models

To face this problem, two approaches are used. The first consists in estimating the threshold on the specific application by splitting the log-likelihood ratios provided by cross-validation in two parts. The first split is used to select the optimal threshold, and the second split is used for evaluation. The second alternative is to recalibrate scores using a linear logistic regression model. In particular, the scores s are calibrated with a linear function

$$f(s) = \alpha s + \beta - \log \frac{\tilde{\pi}}{1 - \tilde{\pi}}$$

where α and β are estimated through a linear logistic regression model and the theoretical threshold is then subtracted to retrieve the calibrated scores. The best value of λ is chosen by trying different ones during cross-validation and evaluating their performance. Results are shown in Table 5.
It can be observed that the optimal estimated threshold provides an improvement for all models, whereas the score calibration through logistic regression is not helpful (indeed, it makes results even worse).

2.3 Combining different classifiers

Another possibility that can be tested is to combine the outputs of different classifiers, so that the final score is the combination of scores of models that use different approaches, and therefore may be better in identifying some specific characteristics of the data. To achieve this, a linear model can be trained to combine the scores. In particular, the combined score can be expressed as

$$S = w^T s + c$$

where s is a vector containing the scores of the different models and w and c are estimated through the logistic regression model. This approach should also provide calibrated scores.

Model	**5-fold** $\tilde{\pi} = 0.5$	
	min DCF	act DCF ($t = -log\frac{\tilde{\pi}}{1-\tilde{\pi}}$)
Fusion SVM+LR	0.215	**0.222**
Fusion SVM+GMM	0.223	0.233
Fusion SVM+GMM+LR	**0.213**	0.225

Table 6: min DCF vs actual DCF of selected models and their fusion

Table 6 shows the results for different fusions of the selected models: SVM and LR, SVM and GMM and also all three models. Min DCFs are evaluated using 5-fold cross validation. It can be observed that the min DCF improves with respect to the single models, with the exception of the fusion SVM and GMM, that performs as the SVM alone. The actual DCF is slightly worse than the minimum DCF (around 5%), as it was for single models.

2.4 Final decision

From the previous analysis it is possible to observe that the performance of the calibrated RBF-SVM, the fusion of SVM and LR and the triple fusion of SVM, GMM and LR have very similar results in terms of min DCF.
Since it is reasonable to expect that a more complex model is able to better capture the different characteristics of the dataset, the final model proposed for the task is the most complex of the three stated above: the fusion of RBF kernel SVM ($C = 10$, $\log\gamma = -2$, rebalancing, Z-normalized features), quadratic logistic regression ($\lambda = 0$, Z-normalized features) and GMM (8 components, Z-normalized features), whose scores are combined through a linear model estimated with linear logistic regression ($\lambda = 0$).
All considerations are done on 5-fold cross validation results, since it is supposed to provide a more detailed analysis of the performance of the models on the complete dataset, whereas a single split may not be representative of the characteristics of the whole dataset.

3 Evaluation

In conclusion, after analysing the performances of different models on the training set and choosing the best one, the effectiveness of the decisions is evaluated by analysing the results of the previously tested models also on the test set. Hyperparameters were chosen according to the best choice of hyperparameters on the training set. Results are given in the form of minimum DCF.

3.1 Single models

Table 7a shows the results for Gaussian models, whose performances are comparable to the ones on the training set (especially those provided by k-fold cross validation approach, which turn out to be more

Model	all data	80% of data
	$\tilde{\pi} = 0.5$	
Gaussianized features - no PCA		
Full-Cov	0.336	0.315
Diag-Cov	0.377	0.382
Tied Full-Cov	0.320	0.324
Tied Diag-Cov	0.379	0.388
Gaussianized features - PCA = 10		
Full-Cov	0.320	**0.310**
Diag-Cov	0.332	0.332
Tied Full-Cov	0.313	0.318
Tied Diag-Cov	**0.310**	0.314
Gaussianized features - PCA = 9		
Full-Cov	0.326	0.328
Diag-Cov	0.338	0.344
Tied Full-Cov	0.322	0.324
Tied Diag-Cov	0.317	0.317
Raw features - no PCA		
Full-Cov	0.337	0.332
Diag-Cov	0.367	0.376
Tied Full-Cov	0.315	0.315
Tied Diag-Cov	0.369	0.370

(a) Gaussian

Model	all data	80% of data
	$\tilde{\pi} = 0.5$	
Gaussianized features - no PCA		
Log-Reg ($\lambda = 0$)	0.341	0.342
Quad-Log-Reg ($\lambda = 10^{-2}$)	0.284	0.278
Z-normalized features - no PCA		
Log-Reg ($\lambda = 10^{-2}$)	0.331	0.327
Quad-Log-Reg ($\lambda = 0$)	0.265	0.264
Z-normalized features - PCA = 10		
Log-Reg ($\lambda = 10^{-2}$)	0.329	0.327
Quad-Log-Reg ($\lambda = 0$)	**0.260**	**0.258**
Raw features - no PCA		
Log-Reg ($\lambda = 10^{-2}$)	0.344	0.342
Quad-Log-Reg ($\lambda = 0$)	1.000	1.000

(b) Logistic Regression

Table 7: Min DCF of Gaussian and Logistic Regression models

reliable), although in some cases the difference is relevant. The best model is the tied-diagonal covariance matrix model, which was performing significantly worse on the training set. The full-covariance model, which was the best on training data, performs better when trained only with 80% of the dataset. Also PCA provides a positive contribution in this case.

Table 7b shows the results for logistic regression models, that are more similar to the ones on the training set. PCA turns out to be more helpful than expected, but the improvement it brings is not huge. Quadratic logistic regression performs much better than linear logistic regression, as expected. Both for Gaussian and for logistic regression, models are trained using 80% of training data (as in the case of the single split) and also using all training data. It can be observed that, in general, training with only 80% of data does not make a big difference with respect to training with all training data. With logistic regression models, training on only 80% of the dataset provides even slightly better results.

Model	all data	80% of data
	$\tilde{\pi} = 0.5$	
Gaussianized features - no PCA		
SVM ($C = 0.1$) - no rebalancing	**0.309**	**0.311**
SVM ($C = 0.1$) - rebalancing	0.323	0.333
Z-normalized features - no PCA		
SVM ($C = 1$) - no rebalancing	0.317	0.320
SVM ($C = 0.1$) - rebalancing	0.332	0.324
Raw features - no PCA		
SVM ($C = 0.1$) - no rebalancing	0.740	0.669
SVM ($C = 0.1$) - rebalancing	0.623	0.835

Table 8: Min DCF of different linear SVM models

Table 8 shows the results for the linear SVM models. Differently from what happened on the training set, Gaussianization of the features is helpful, but also Z-normalization provides good results. Raw features perform very poorly, as expected. Training the models on 80% of the data or on the complete training dataset does not bring a huge difference in general. Class rebalancing is slightly detrimental in this case.

Model	$\tilde{\pi} = 0.5$
Z-normalized features - no PCA	
Quad-SVM ($C = 1$) - no rebalancing	0.267
Quad-SVM ($C = 0.1$) - rebalancing	0.278
RBF-SVM ($C = 10, log\gamma = -2$) - no rebalancing	**0.258**
RBF-SVM ($C = 10, log\gamma = -2$) - rebalancing	0.260

Table 9: Min DCF of different quadratic and RBF kernel SVM models

Table 9 shows the results for kernel SVM models (quadratic and RBF). As expected, both perform better than the linear SVM, and RBF kernel is better than quadratic kernel, although the difference among the two is not as relevant as it was on the training set. Quadratic kernel SVM performs similarly to quadratic logistic regression, as observed on the training set. The best result is achieved without rebalancing, but the difference when rebalancing is applied is not relevant.

Model	$\tilde{\pi} = 0.5$
Z-normalized features - no PCA	
tied-GMM (16 components)	0.316
tied-diag-GMM (16 components)	0.325
GMM (8 components)	0.306
diag-GMM (16 components)	0.329
Gaussianized features - no PCA	
tied-GMM (8 components)	**0.282**
tied-diag-GMM (16 components)	0.305
GMM (8 components)	0.328
diag-GMM (16 components)	0.320

Table 10: Min DCF of different GMM models

Table 10 shows the results for GMM models. The selected model (standard GMM, 8 components, Z-normalized features) is the best among the ones using Z-normalized features, but, contrarily to what was observed on the training set, the best performance is achieved with a tied-covariance model using Gaussianized features.

3.2 Model fusions and score calibration

Then, also score calibration and optimal threshold selection is evaluated. Table 11 shows the results on the three selected models. The optimal threshold t^* is the best threshold for the complete training set, and score calibration is performed by means of a linear logistic regression model trained on the complete training set using the optimal values for λ evaluated during the model tuning phase (*i.e.* score calibration on the training data). The DCF is calculated on the test set.

Model	$\tilde{\pi} = 0.5$			
	min DCF	act DCF ($t = -log\frac{\tilde{\pi}}{1-\tilde{\pi}}$)	act DCF (t^*)	calibrated
SVM	0.260	0.278	0.280	0.278
LR	0.265	0.286	0.293	0.273
GMM	0.306	0.341	0.325	0.315

Table 11: min DCF vs actual DCF of selected models and their fusion

It can be observed that the loss of performance due to uncalibrated scores is relevant, especially for the GMM. Optimal threshold selection is helpful for the GMM, but it makes results worse for the other two models. On the other hand, score calibration is beneficial for the GMM and the LR, and it is not detrimental for the SVM.

In conclusion, Table 12 shows the results for the different model fusions. Actual DCF is obtained by using the theoretical threshold, since model fusions are supposed to provide calibrated scores.
It can be observed that all the fusions are, in general, more performant with respect to the single models, both in terms of minimum DCF and actual DCF (the loss of performance is lower than the one observed for single models, so, model fusion provides also score calibration). Moreover, the fusion of all three models, that was selected as the final model in the previous section, achieves indeed the best result among the evaluated models.

Model	$\tilde{\pi} = 0.5$	
	min DCF	act DCF ($t = -log\frac{\tilde{\pi}}{1-\tilde{\pi}}$)
Fusion SVM+LR	0.252	0.262
Fusion SVM+GMM	0.260	0.267
Fusion SVM+GMM+LR	**0.250**	**0.258**

Table 12: min DCF vs actual DCF of selected models and their fusion

Overall, it can be concluded that the evaluation dataset has similar characteristics with respect to the training dataset, as the results achieved in the two cases are compatible. Nevertheless, the final performance of the fusion is worse than the one achieved on the training set, and some differences were observed also when dealing with single models.

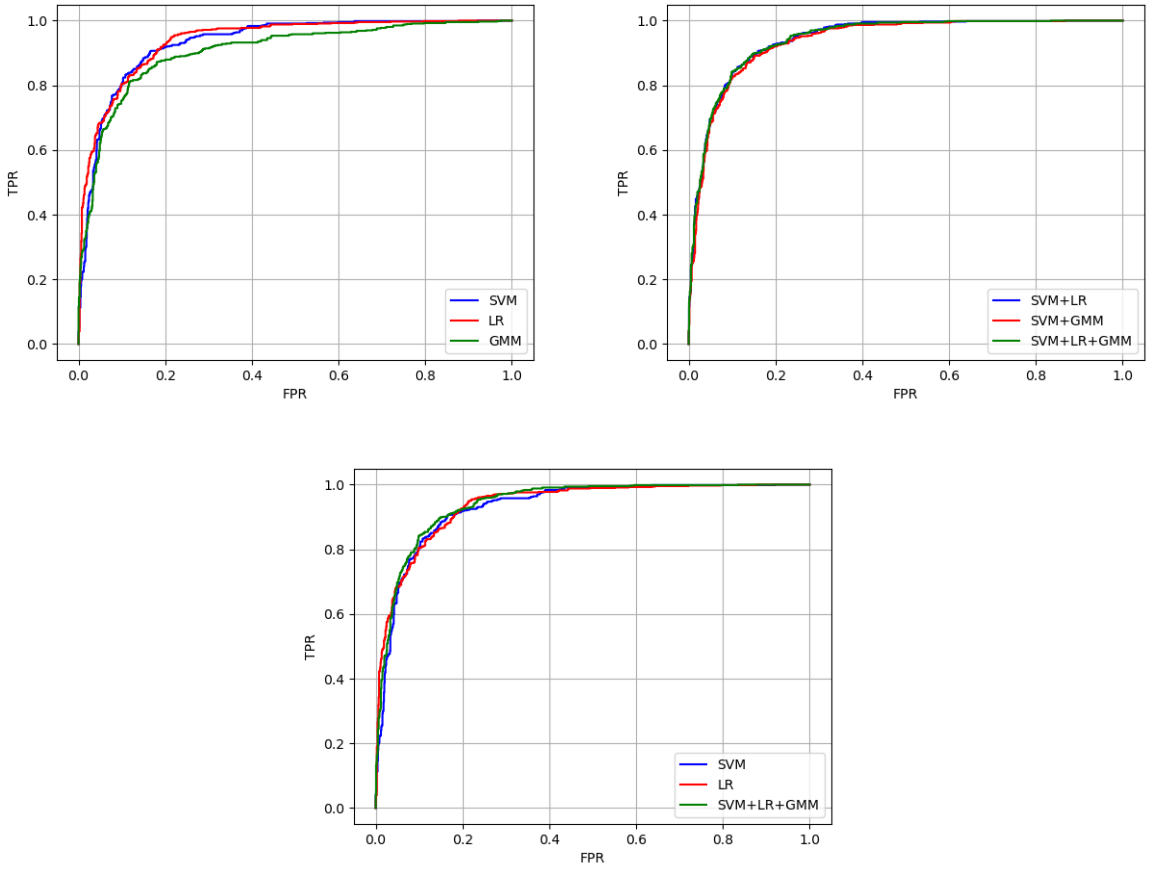


Figure 8: ROC plots

3.3 Different applications

In the end, it is possible to analyse the performance of the best models also for different applications.

Figure 8 shows the ROC plots for the three single models, the three model fusions and a comparison among single models and the best fusion. It can be observed that the GMM performs quite worse with respect to the other two single models, whereas the three fusions have very similar performances across all operating points. Moreover, the fusion achieves better results with respect to single models for the operating points of interest in the task, but there are operating points for which the single logistic regression performs better than the fusion.

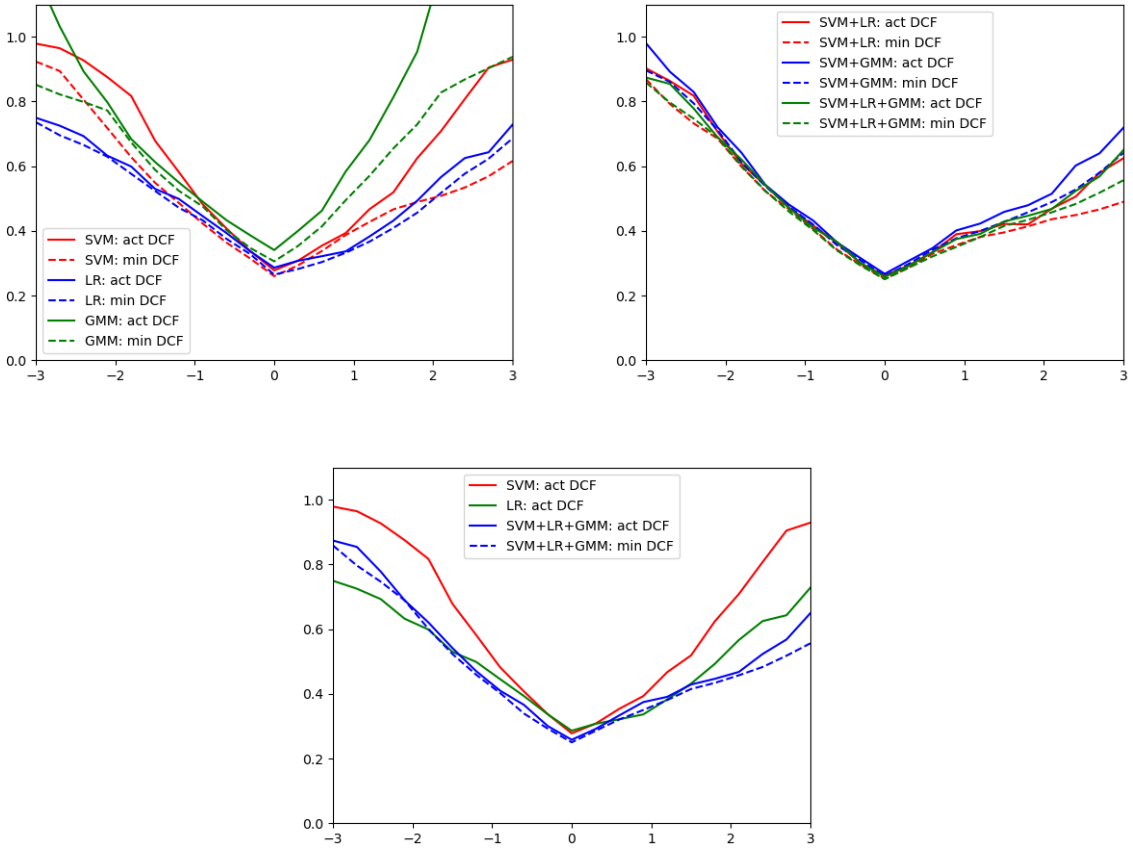


Figure 9: Bayes error plots

Figure 9 shows the Bayes error plots for the different models, analysing the min DCF and actual DCF for different values of the prior log-odds $\tilde{p} = \log(\frac{\tilde{\pi}}{1-\tilde{\pi}})$, which corresponds to the different applications. It can be observed that, among the single models, the GMM performs worse than the other two for many applications, and the performance loss due to uncalibrated scores is quite relevant for many applications (mostly for GMM and logistic regression). The three model fusions have comparable performances across all applications and their scores are quite well calibrated. Finally, the fusion of three models provides a better performance with respect to single models for the application of interest in the task, but, on some applications, the logistic regression model outperforms the fusion.