Cazzola Michele s323270

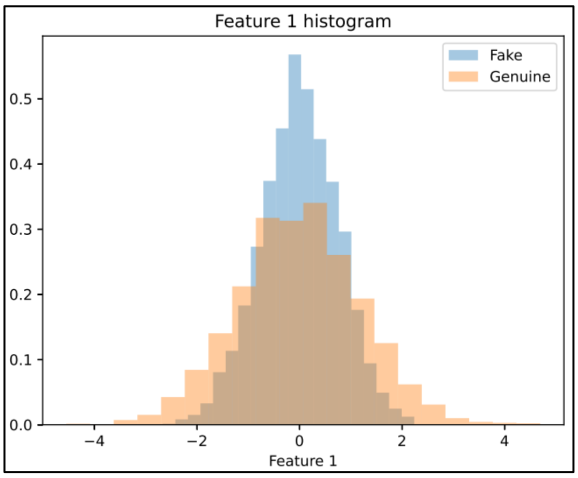
MACHINE LEARNING AND PATTERN RECOGNITION

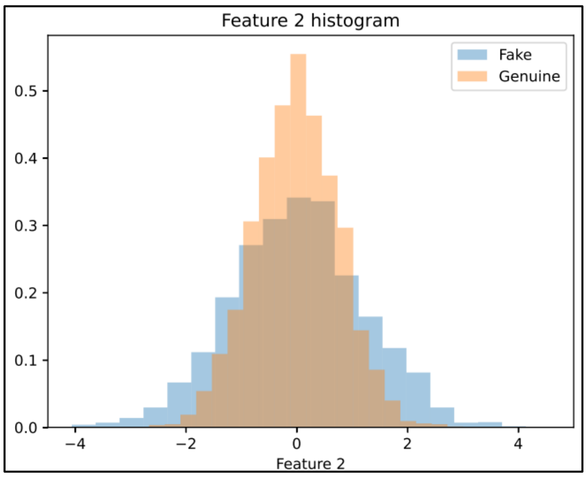
Project task

Laboratory 2 – Loading and visualization

1. For features 1-2, both classes exhibit a unimodal distribution with Normal shape, and they overlap in the central part of their domain:
    - the mean values are the same (approximately 0) for both features and their respective classes.
    - the variances are different:
        - approximately 0.6 for the *fake* class of the first feature and the *genuine* class of the second feature.
        - approximately 1.4 for the *genuine* class of the first feature and the *fake* class of the second feature.

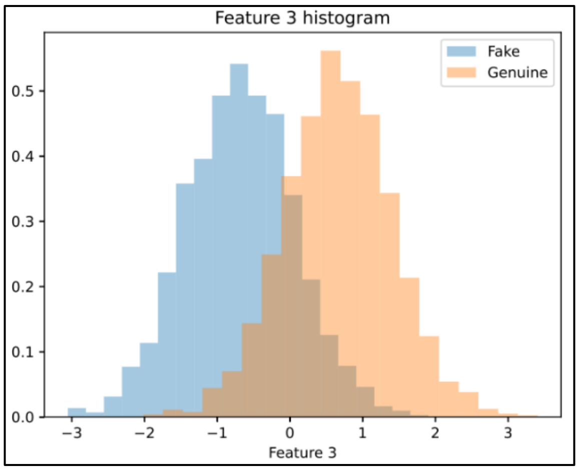
   We observe that, for each feature, the class with the lower variance exhibits the highest modal frequency (peak value).

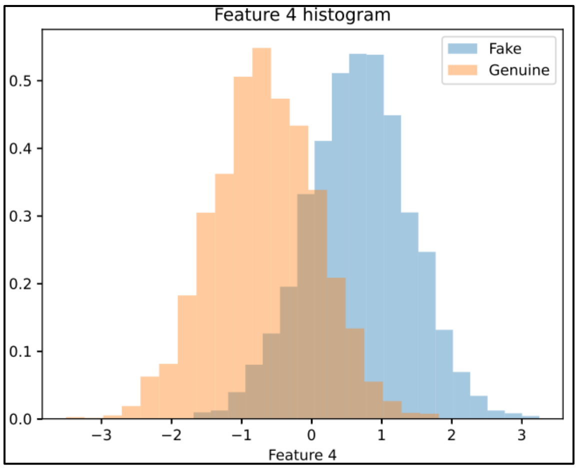




2. For features 3-4, both classes demonstrate a unimodal distribution with Normal shape, and they overlap on their respective sides; for each pair of classes within each feature:
    - the mean values are opposite but nearly equal in magnitude (between 0.6 and 0.7).
    - the variances are nearly equal (between 0.5 and 0.6).

   We observe that, for each feature, the classes display the similar modal frequencies.

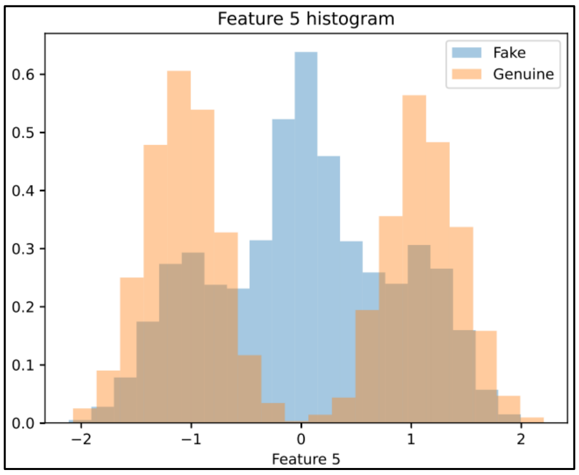


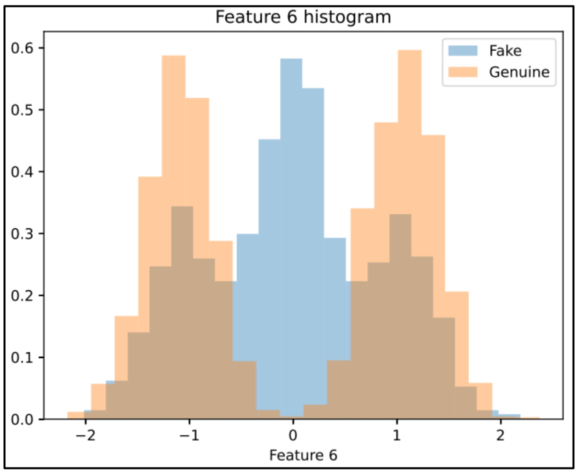


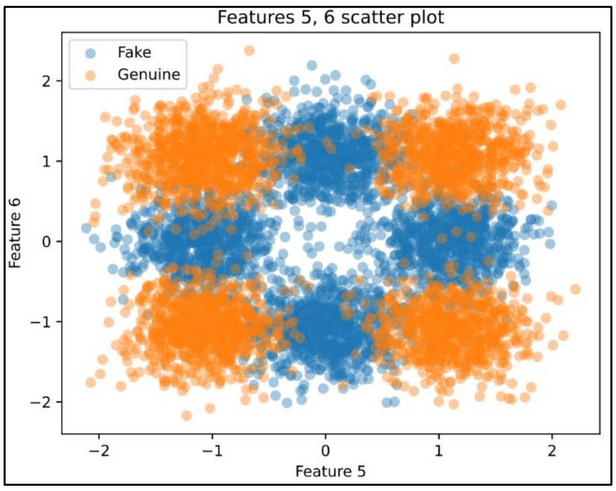
3. For features 5-6, the *fake* class displays a unimodal distribution, while the *genuine* class exhibits a bimodal distribution:
    - for the *fake* class, the modal values are opposite.
    - for the *genuine* class, the modal value is approximately zero.

   We observe that they overlap around the modal values of the *fake* class distribution, while the overlapping is minimal in the central part of the domain.

   Furthermore, the scatter plots highlight the presence of four clusters for each class.

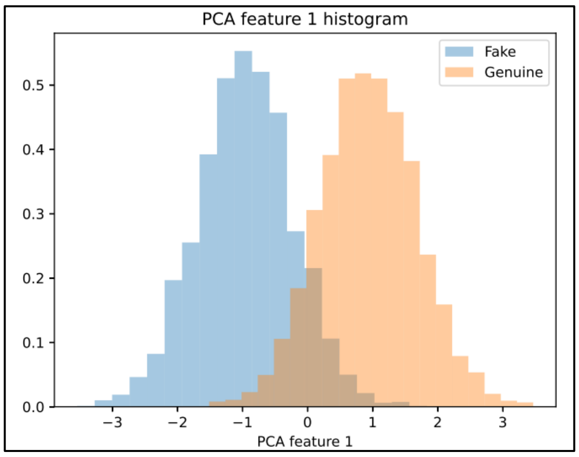


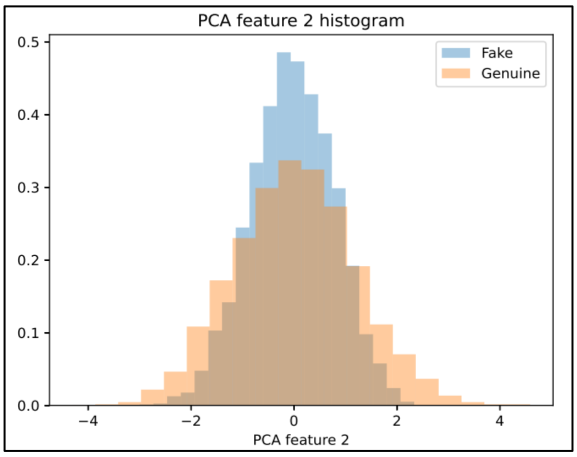


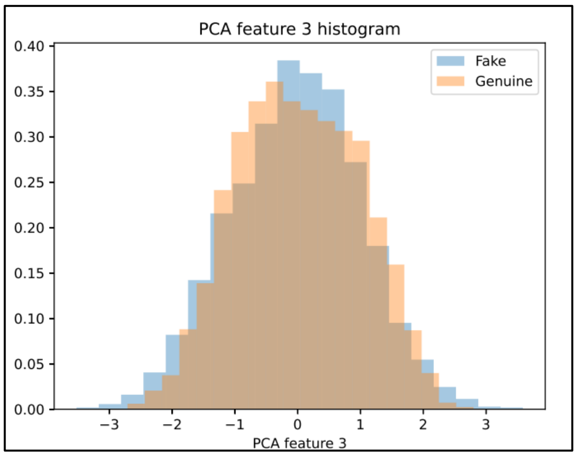


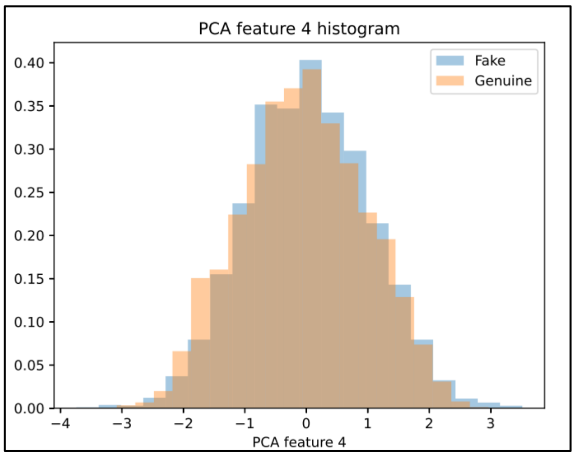
Laboratory 3 – Dimensionality reduction

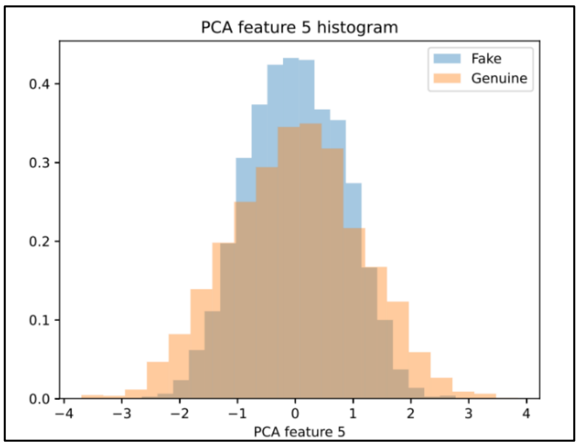
1. By applying PCA, we get these histograms of the six different projected features, in descending order of explained variance:

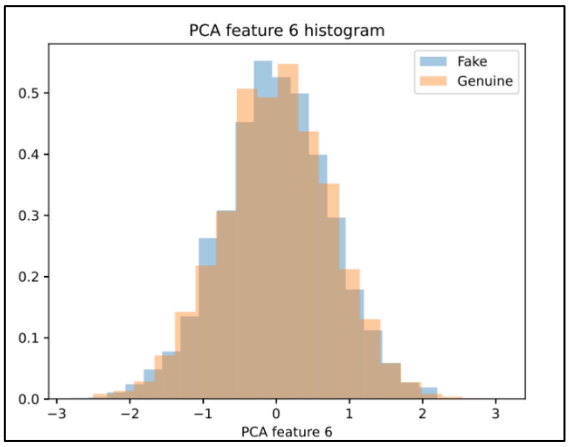






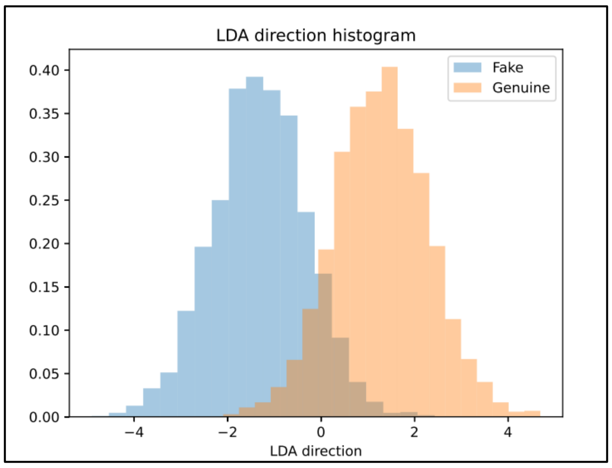






We observe that these dimensions still present histograms with a significant overlapping, except for the first one, where the class distribution are more distinguishable. Moreover, these plots show that, despite the strong overlapping, all the features have a Gaussian distribution, which could lead to an advantage in classification stage.
The scatterplots do not highlight the presence of clusters among reduced features, differently by the ones referred to the original ones.

2. By applying 1-dimensional LDA, we get the following histogram, which refers to the class distribution of the projected feature:

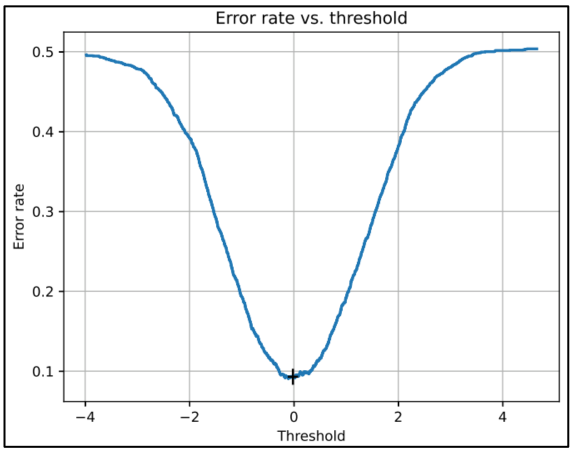


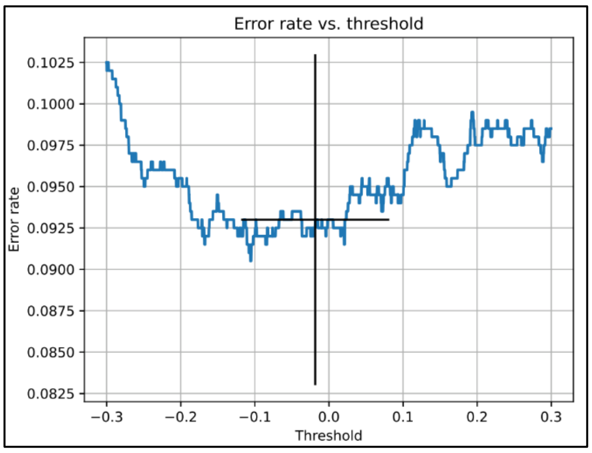
We can observe that a significant overlapping is still present, but:

- This overlapping is less than the one observed in the original features: it is comparable with the one obtained in the first PCA feature.
- both the class distributions are bell-shaped, as after applying PCA.

thus, the LDA direction is better than the original features and it is more suitable for utilization in a classification task.

3. To apply LDA as a classifier, the dataset is split randomly into training and validation set, with a dimension of two-thirds and one-third with respect to the original dataset, respectively. The threshold is computed as the mean of the mean values per class, working only on the training set: its value is about -0.02, as it is possible to notice in the plots at point four. The error rate obtained is 0.093 (9.3 %): the accuracy is greater than 90%, thus the classifier could be considered as sufficiently accurate, if we consider its simplicity.
4. We change the threshold in a wide range, delimited by the minimum and maximum value in the validation set with a step of about 10^{-4}; the goal is to find the value that minimizes the error rate, just by performing the prediction stage with different values of the threshold. The results are shown in the following plots:





The left plot presents a global view: we can observe that the error rate has a minimum around a point (denoted by the black cross), which represents the optimal value of the

threshold; it is close to the computed one at the previous point. In fact, by zooming in (right plot), we can notice that the minimum value for the error rate is about 0.09 (9 %) and it corresponds to a threshold value near to -0.1, while the computed one is about -0.02: this is coherent with the error rate obtained as the previous point (0.093), which is near to the optimal one.
Therefore, the accuracy can be improved by a little quantity, but the error rate computed at the previous point is higher than the minimum one by less than 1%.

5. Applying PCA as a preprocessing stage means to:
    - Select the number of PCA dimensions *m*, to keep after PCA transformation; in particular, the *m* dimensions chosen are the ones that explain the higher variance.
    - Apply 1-dimensional LDA on the dataset, projected over the *m* PCA features.
    - Use LDA as a classifier, comparing the results obtained for different values of *m*.

   This task has been implemented by iterating over valid (and meaningful) values of *m*, such as from 2 to 5 (included):
    - with *m* = 1 LDA would become irrelevant since it would not perform a dimensionality reduction.
    - with *m* = 6 PCA would become irrelevant since it would not perform a dimensionality reduction, but only a projection over other directions than the original ones.

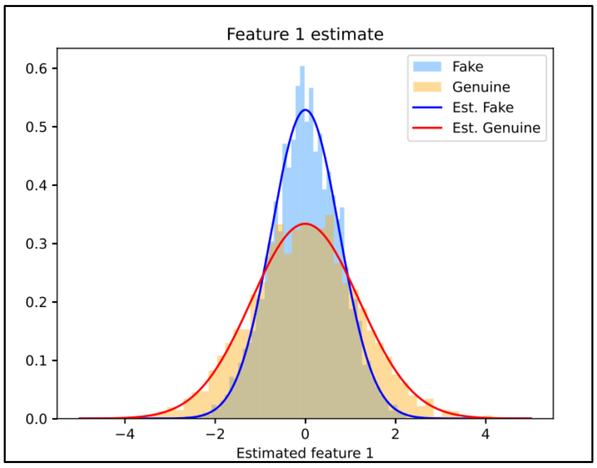
   The results obtained are the following:

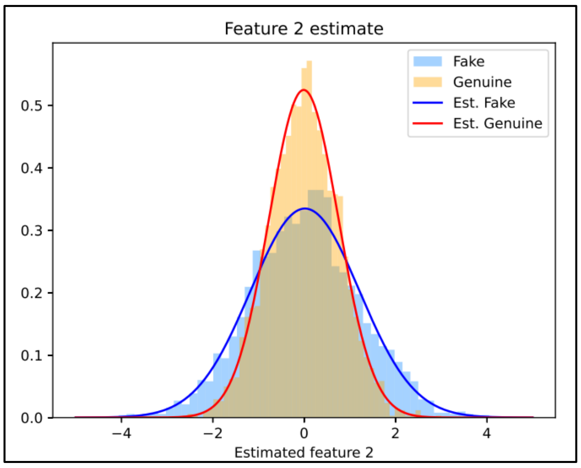
PCA dimensions (*m*)	Error rate	Error rate (%)
5	0.0930	9.30
4	0.0925	9.25
3	0.0925	9.25
2	0.0925	9.25

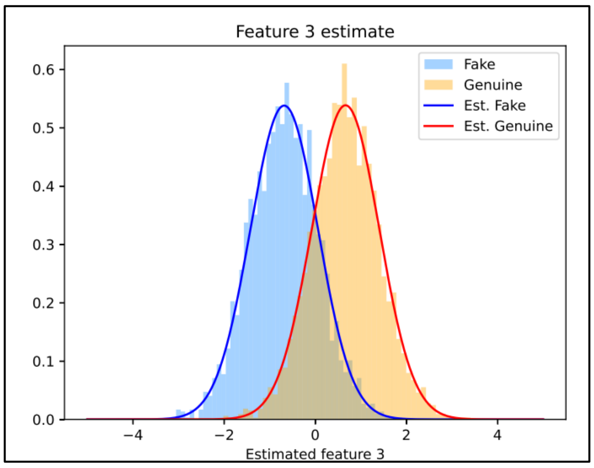
   Therefore, we can observe that PCA preprocessing is beneficial for LDA classification in all cases, with an accuracy improvement for each value of *m*, except for five.

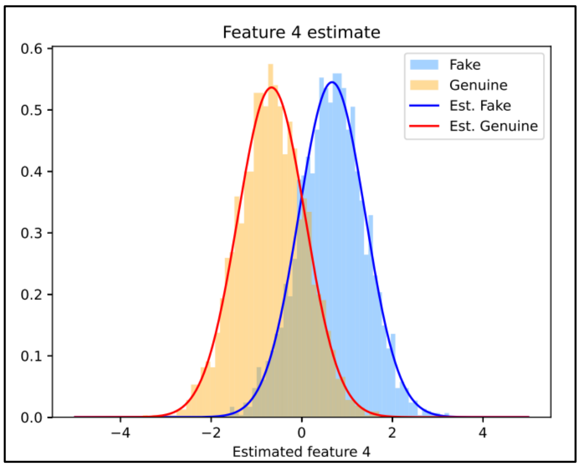
Laboratory 4 – Gaussian density estimation

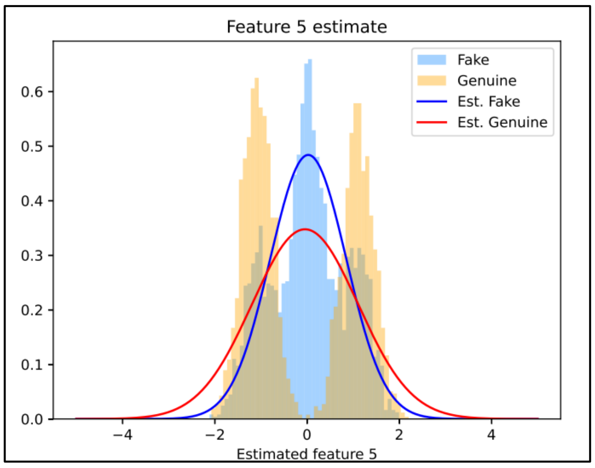
To perform Gaussian estimation, original dataset is split in two parts (by class), for each feature; then, the best Gaussian fitting is estimated by computing mean and variance with the maximum likelihood method and then by plotting the corresponding Gaussian distribution over the feature histogram. The results obtained are the following:

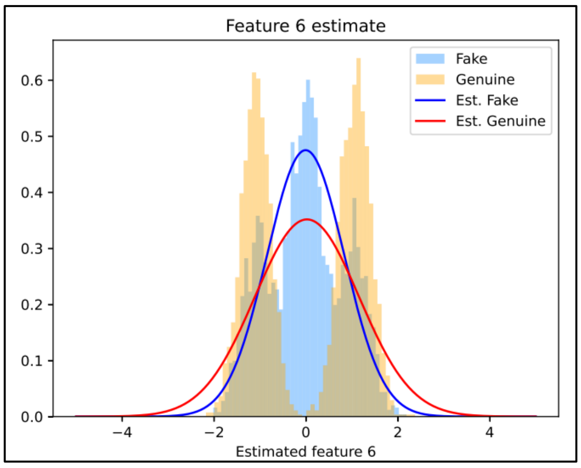












We can observe that a Gaussian fitting is good only for the first four features, while the other two ones have a different kind of distribution; therefore, this method results to be accurate only to model the first four features. The last two have the same mean value as the estimated Gaussian distribution, but they differ in shape from it.

Laboratory 5- Gaussian models

The application of MVG model on the dataset is performed by using the same splits as LDA, in three different variants (standard MVG, Tied MVG and Naïve Bayes MVG); the following error rates are obtained:

Classifier	Error rate (%)
Standard MVG	7.00
Tied MVG	9.30
Naïve Bayes MVG	7.20

As expected, we observe that:

- Tied MVG classifier has the same error rate as LDA classifier, since they are equivalent models.
- Naïve Bayes classifier is slightly worse than MVG, indeed per-class correlation matrices show a low correlation among dataset features: since covariances are far smaller than feature variances, covariance matrices are almost diagonal, meaning that they have non-diagonal elements, but they are negligible due to their value. Thus, a diagonal representation of per-class covariance matrices is a good approximation of actual ones.

Moreover, we can observe that dataset points seem to be not well-separable with linear surfaces, since LDA performs worse than other two classifiers.

As we can see in the previous plots, the Gaussian assumption holds only for both classes of features 1,2,3,4, whereas features 5,6 have per-class distributions that do not fit well with Gaussian estimation, thus a MVG classification performed only on them will surely lead to inaccurate results.

By applying the same classifiers on features 1-4 (that is discarding the worst-fitted two), we obtain the following results:

Classifier	Error rate (%)
Standard MVG	7.95
Tied MVG	9.50
Naïve Bayes MVG	7.65

Despite the last two features seem to be misleading for a MVG classifier, they carry some significant information on data distribution, since we observe an increase of error rates by removing them.

By repeating the classification only on features 1-2, we obtain the following results:

Classifier	Error rate (%)
Standard MVG	36.50
Tied MVG	49.45
Naïve Bayes MVG	36.30

We can observe the following:

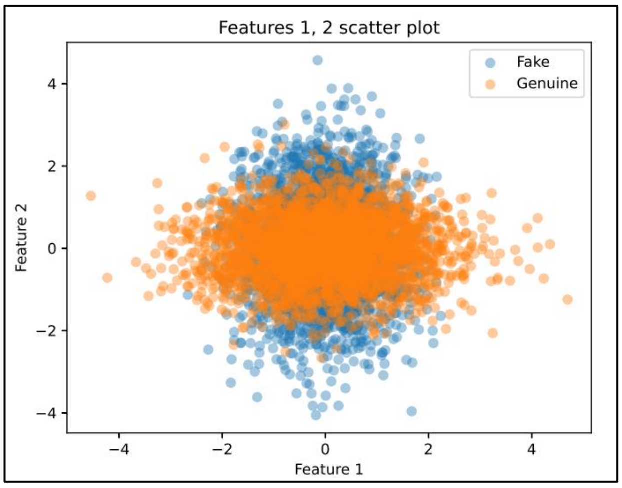
- Since features 1-2 classes are not well-separable (as shown in the scatter plot below), all MVG classifiers perform far worse than when applied on all the features.
- Tied MVG classifier performs worse than others since the two features have different per-class variances, thus per-class covariance matrices are surely different, and a tied covariance assumption is inaccurate.

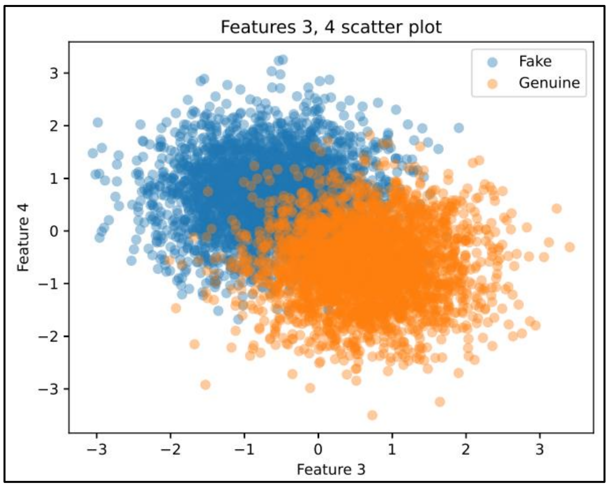
By repeating the classification on features 3-4, we obtain the following results:

Classifier	Error rate (%)
Standard MVG	9.45
Tied MVG	9.40
Naïve Bayes MVG	9.45

We can observe the following:

- Since features 3-4 classes are well separable (as shown in the scatter plot below), error rates do not increase significantly with respect to the model trained on all features.
- Since per-class variances are similar, tied covariance assumption works well; moreover, the simplification of the model is beneficial, indeed this variant performs slightly better than full-covariance MVG, avoiding the risk of overfitting.





Finally, applying the same model with a preprocessing stage, implemented as a PCA application on the whole dataset, we obtain the following results, in terms of percentual error rates:

PCA components	Standard MVG	Tied MVG	Naïve Bayes MVG
2	8.80	9.25	8.85
3	8.80	9.25	9.00
4	8.05	9.25	8.85
5	7.10	9.30	8.75

We can observe a general decrease of the accuracy for both Naïve Bayes and standard MVG classifier, due to the reduction of the information extracted from data. Tied MVG performs slightly better for a smaller number of components (less than 5): this is possible since all PCA features (except for the second and, partially, the fifth) have similar per-class covariances, thus the tied covariance assumption is slightly better in this case than in the case without PCA preprocessing.

In conclusion, the best overall Gaussian model (in terms of error rate) is the standard MVG classifier (without PCA), with an error rate of 7 % on the validation set.

Laboratory 7 – Model evaluation e Bayes decisions

Model evaluation is performed by computing DCF (both actual and minimum values) considering a set of given application priors, and their corresponding prior log-odds, after having performed label predictions basing on optimal Bayes decisions.

First, the analysis is focused on different application triplets:

- (0.5, 1.0, 1.0): uniform prior and costs
- (0.9, 1.0, 1.0): the prior probability of a genuine sample is higher
- (0.1, 1.0, 1.0): the prior probability of a fake sample is higher
- (0.5, 1.0, 9.0): the prior is uniform, but the cost of accepting a fake image is larger
- (0.5, 9.0, 1.0): the prior is uniform, but the cost of rejecting a legit image is larger

The corresponding effective prior are shown in the following table:

Prior	False negative cost	False positive cost	Effective prior
0.5	1.0	1.0	0.5
0.9	1.0	1.0	0.9
0.1	1.0	1.0	0.1
0.5	1.0	9.0	0.1
0.5	9.0	1.0	0.9

We can observe that different costs of misclassification are reflected into the effective prior: stronger security value, represented by a high false positive cost, corresponds to a lower prior probability of a genuine fingerprint. Indeed, we can also notice that some of these applications are equivalent, due to the reason just explained, thus we are going to analyze MVG classifier performance on three different applications, given only their effective priors (0.1, 0.5, 0.9), and considering unitary misclassification costs.

Optimal Bayes decisions are computed, in terms on both actual DCF and minimum DCF; the results are shown in the following table:

Effective prior	PCA dimensions	Full covariance		Tied covariance		Naïve Bayes	
		Min. DCF	Act. DCF	Min. DCF	Act. DCF	Min. DCF	Act. DCF
0.1	2	0.353	0.388	0.363	0.396	0.356	0.387
	3	0.356	0.388	0.368	0.408	0.365	0.395
	4	0.301	0.353	**0.361**	0.403	0.361	0.397
	5	0.274	0.304	0.365	0.405	0.354	0.393
	Not applied	**0.263**	0.305	0.363	0.406	**0.257**	0.302
0.5	2	0.173	0.176	**0.179**	0.185	0.171	0.177
	3	0.173	0.176	0.183	0.185	0.175	0.180
	4	0.154	0.161	0.182	0.185	0.172	0.177
	5	0.133	0.142	0.181	0.186	0.174	0.175
	Not applied	**0.130**	0.140	0.181	0.186	**0.131**	0.144
0.9	2	0.438	0.443	0.435	0.479	0.432	0.442
	3	0.439	0.468	**0.434**	0.457	0.434	0.459
	4	0.415	0.460	0.444	0.462	0.431	0.463

	5	0.351	0.398	0.445	0.463	0.431	0.466
	Not applied	**0.342**	0.400	0.442	0.463	**0.351**	0.389

Since scores are not necessarily well-calibrated, quality comparisons are performed basing on minimum DCF:

- Separately for each application, full covariance model is better than tied covariance model: in particular, this result is clear for $\pi = 0.5$ and $\pi = 0.9$; moreover, Naïve Bayes model is almost equivalent to full covariance, except for some cases of PCA preprocessing (with a high number of components) where full covariance performs better
- Separately for each variant:
  - Full covariance performs better without PCA preprocessing since some information is lost while reducing dimensionality.
  - Tied covariance performs better with PCA preprocessing (with different number of components, depending on the application): here, PCA helps finding features with similar covariances, so that tied model is more suitable.
  - Naïve Bayes variant performs better without PCA preprocessing, like full covariance model.
- Moreover, relative performance results are consistent for different application: in general, full covariance model performs better than tied covariance model and slightly better than Naïve Bayes one, independently from the application chosen. More details are explained in the previous points.

Scores are not explicitly calibrated: the relative mis-calibration loss is computed as the percentual loss of DCF value, divided by the value of minimum DCF:

$$100 \cdot \frac{DCF_{act} - DCF_{min}}{DCF_{min}}$$

Its values are shown in the following table:

Effective prior	PCA components	Full covariance	Tied covariance	Naïve Bayes
0.1	2	10.00	9.09	8.63
	3	8.86	10.89	8.36
	4	17.17	11.66	9.84
	5	11.07	11.03	10.87
	Not applied	16.06	11.91	17.59
0.5	2	1.68	3.45	3.48
	3	1.42	1.08	3.04
	4	4.69	1.60	3.09
	5	6.59	2.69	0.75
	Not applied	7.50	2.64	9.76
0.9	2	1.15	9.96	2.33
	3	6.56	5.14	5.70
	4	10.79	3.92	7.35

	5	13.33	3.91	7.37
	Not applied	16.87	4.63	10.92

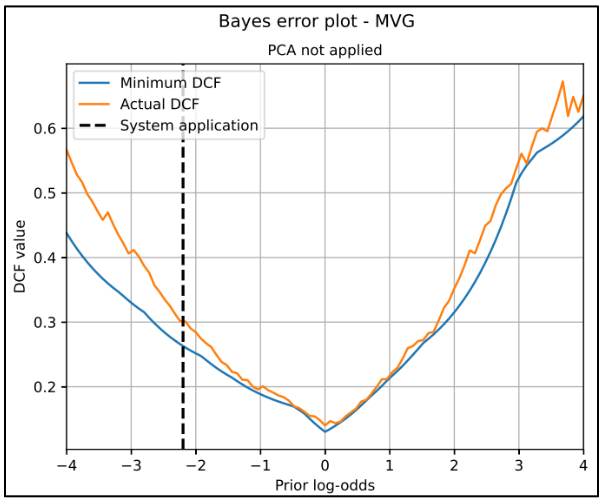
As we can see, tied covariance model is in general better calibrated than others, with full covariance that has a strong calibration loss for applications with unbalanced classes; moreover, we can notice that in general miscalibration increases with the quality of the model, in terms of minimum DCF: it is not a strict rule, but it is an observed trend in all models.

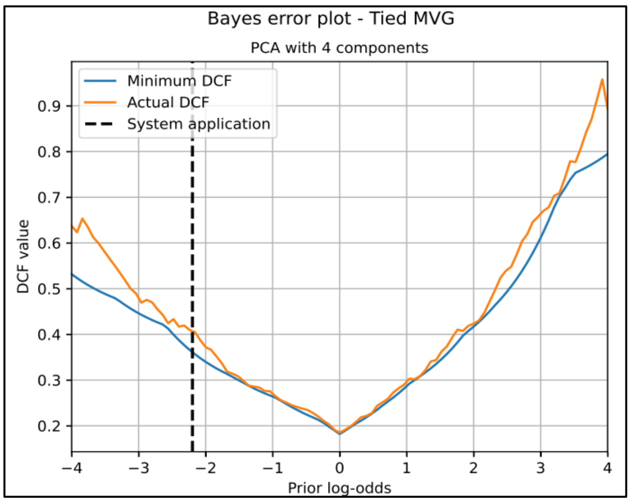
Finally, we can observe that unbalanced class priors are associated with a strong miscalibration, which is more evident with π = 0.1. Scores are not well-calibrated for each application: this is caused by lack of precision in Gaussian estimation related to data distribution, since MVG produces LLR scores, which represent logarithmic ratios between class conditional distributions.

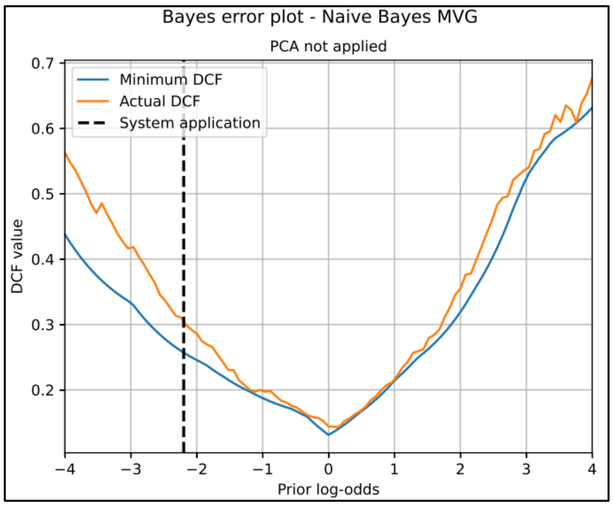
By considering only the first application (π = 0.1), the best results in terms of minimum DCF are the following:

- Full covariance: 0.263
- Tied covariance: 0.361.
- Naïve Bayes: 0.257

Bayes error plots are the following:







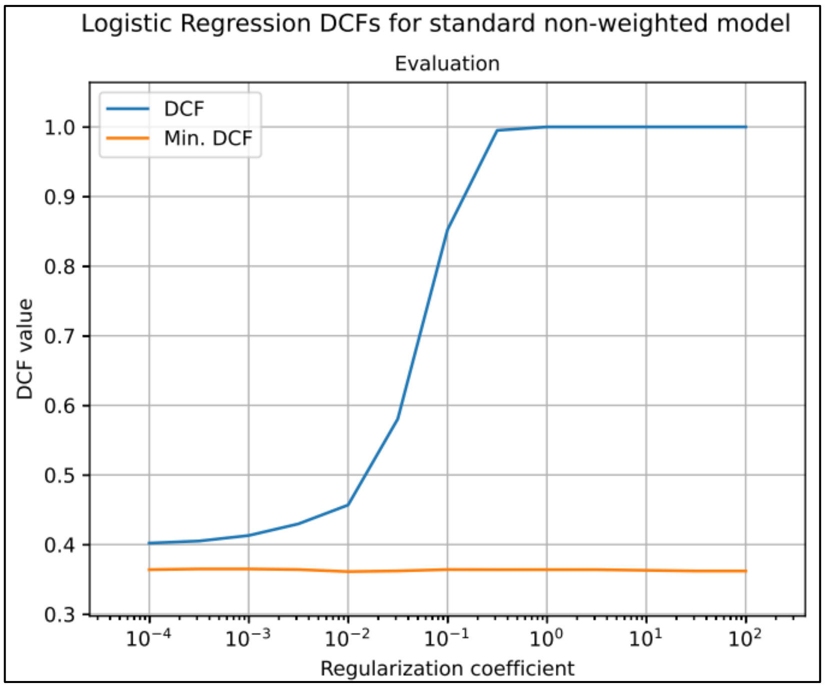
As we can see, Bayes error plots have the same shape in terms of minimum DCF, across the considered range, even if the minimum value is slightly different; the biggest difference is the more significant growth of minimum DCF for tied covariance model, while moving towards applications with unbalanced priors.

Scores are not well-calibrated on the full range of effective prior log-odds: as we can observe, there is a significantly increasing miscalibration moving away from the center of the range; this phenomenon involves our system application (highlighted in the plots), whose effective prior log-odds is:

$$\log \frac{\pi}{1-\pi} = -2.197$$

Lab 8 – Logistic regression

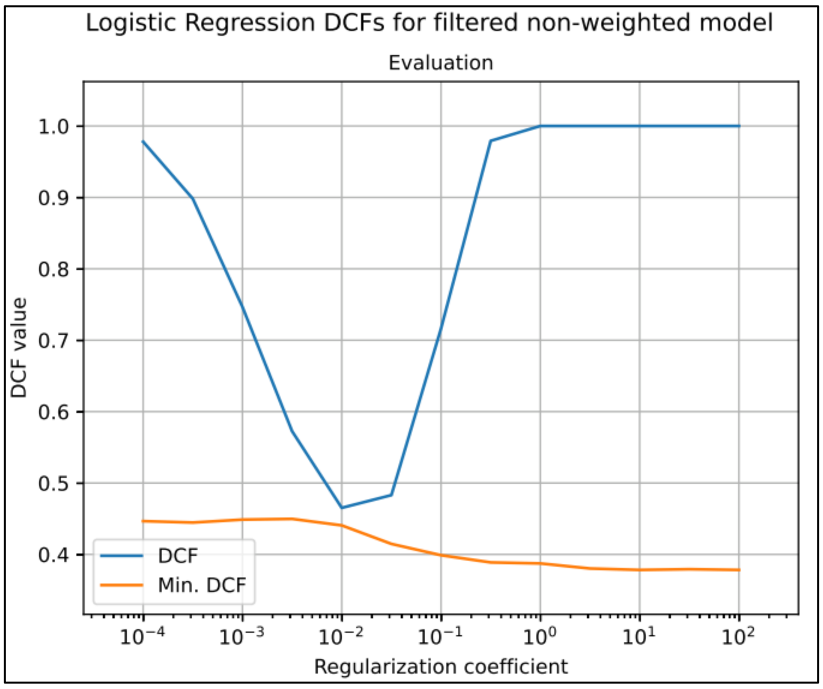
The first task is employed by using an unweighted logistic regression, using different values as regularization coefficient; by considering an application prior $\pi_T = 0.1$, minimum DCF and actual DCF are measured, and the following results are obtained:



As we can see:

- Minimum DCF is almost invariant to regularization, indeed its value is always included in the interval 0.35 – 0.4: the best one is 0.361, corresponding to $\lambda = 0.01$
- Actual DCF is strongly influenced by regularization: its value grows quickly, saturating to one for regularization coefficient values greater than 0.01; it is caused by the loss of probabilistic meaning of this metric when strong regularization is employed. Actual DCF corresponding to best minimum DCF is 0.457. Thus, miscalibration is worse for higher values of regularization, for the reason previously mentioned.
- No significant overfitting phenomena are observed, even with lower values of regularization coefficient, since actual DCF does not increase for smaller values of λ and minimum DCF is almost constant over the range of observed values.

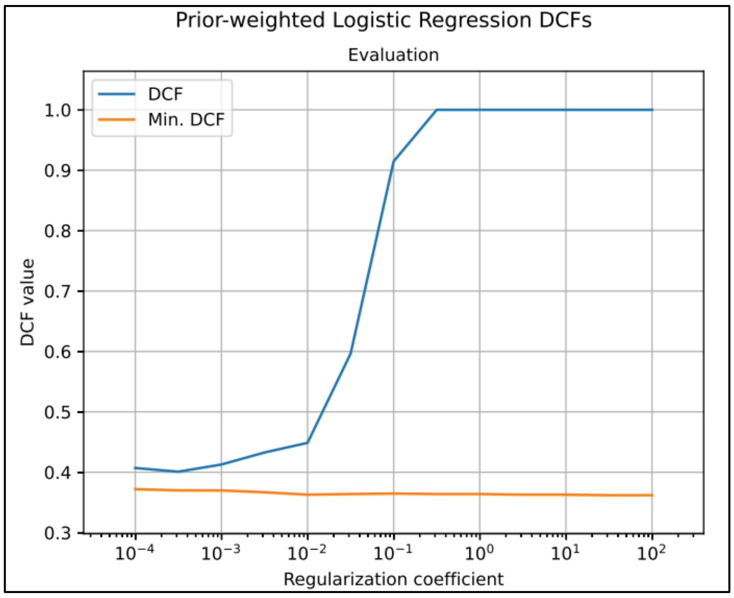
Since overfitting is more relevant for smaller datasets (given the number of parameters to estimate), the same task is performed on a filtered training dataset, with only fifty samples (instead of four thousand); the results are shown in the following plot:



We can observe that:

- Both actual and minimum DCF grow for lower values of λ: in particular, minimum DCF is an almost-decreasing curve, so that it stays constant for smaller values of λ, than it decreases until λ = 10 (0.378), remaining constant again until the end of the range. This is evidence of an overfitting phenomenon, which is mitigated only with strong regularization.
- As in the previous task, actual DCF loses its probabilistic meaning when strong regularization is used.

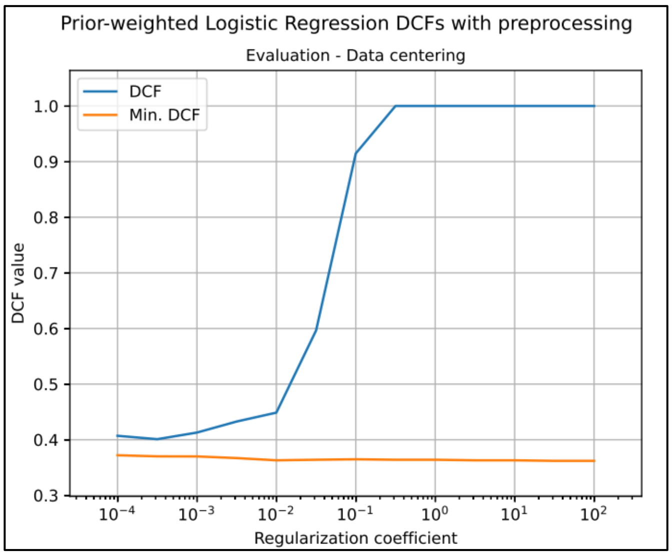
By using a prior-weighted variant, the information about the application prior is used both to build the model and to compute the scores in prediction step; the following results are obtained:



We can observe that:

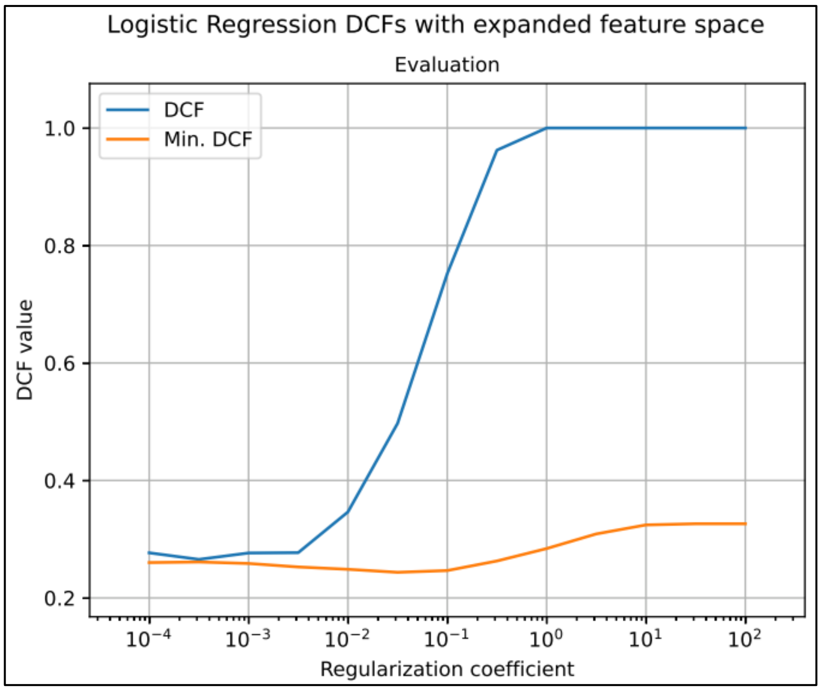
- Actual DCF grows due to loss of probabilistic interpretation, as in the previous tasks.
- Both actual and minimum DCF highlight a slight overfitting phenomenon, since they grow for low values of the regularization coefficient ($\lambda < 0.001$)
- The best result is obtained for minimum DCF equal to 0.362, which corresponds to $\lambda = 31.62$; since this configuration has a strong regularization, the actual DCF is 1.00, due to loss of probabilistic meaning
- Except for few, not significant differences, the results are like those obtained with an unweighted model.

Since the regularized LR is not invariant to transformations of data features, doing some preprocessing could lead to variations in the results; by centering (both training and validation) data with respect to the training set mean and applying the prior-weighted LR, the following results are obtained:



As we can see, data centering does not significantly affect the results, thus we can observe that data features were already centered with respect to the training set mean; in fact, the obtained plot is the same as the prior-weighted LR without data centering.

By expanding the feature space and applying a prior-weighted LR, it is possible to obtain quadratic boundary surfaces; the results are the following:



We can observe that:

- A phenomenon of slight overfitting has happened, since actual and minimum DCF increase for low values of λ.
- The best result is given by a minimum DCF of 0.244, corresponding to λ = 0.032 and an actual DCF of 0.497
- Data are better separable by a quadratic surface than a linear one since this model works better than a linear LR.

The following table summarizes the best results for MVG and LR classifiers:

Model	Minimum DCF	Actual DCF
MVG – Full covariance	0.263	0.305
MVG – Tied covariance	0.351	0.403
MVG – Naïve Bayes	0.257	0.302
LR – Unweighted	0.361	0.457
LR – Prior-weighted	0.362	1.000
LR – Expanded (unweighted)	**0.244**	0.497

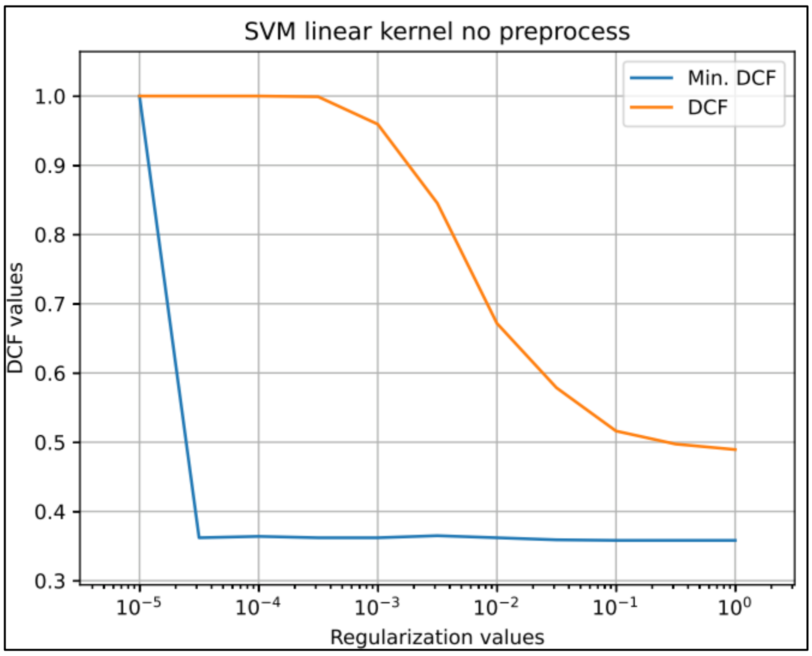
We can observe that:

- Models that find quadratic boundary surfaces perform better, in terms of minimum DCF.
- MVG models produce scores which are better calibrated than LR models.

thus, we can say that the dataset has features that are better separable by quadratic surfaces than linear ones, and it is coherent with the results obtained in the LR tasks. In conclusion, we observe that a quadratic logistic regression performs slightly better than a MVG model, even if its results are worse calibrated, having a higher actual DCF than Gaussian models.

Lab 9 – Support Vector Machine

The first task consists in a linear SVM, whose results are shown below:



Since regularization values correspond to parameter C, here regularization should be interpreted in the opposite way of logistic regression, indeed:

- For low values of C, that is strong regularization, the model underfits, with both actual and minimum DCF having large values.
- For high values of C, that is weak regularization, the model performs better: as in LR, overfitting phenomena is not observed, since dataset is large enough.

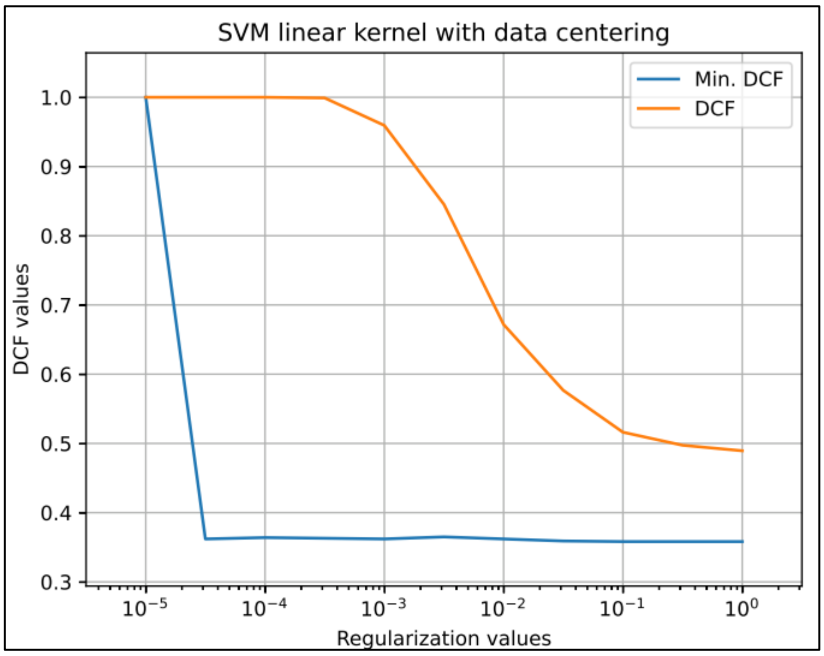
The best result, in terms of minimum DCF, is equal to 0.358, obtained for $C = 0.1$ and $K = 1$, and corresponding to an actual DCF of 0.516. In general, as in logistic regression, we can observe:

- Minimum DCF almost constant for all values of C.
- Actual DCF decreases for higher values of C, that is for weaker regularization.

Moreover:

- This result is comparable to the one obtained for MVG with tied covariance, indeed both approaches aim to find linear boundary surfaces
- As expected, SVM scores are mis-calibrated, since they do not have a probabilistic interpretation, in fact actual DCF for best linear SVM is higher than the one referred to the best MVG tied model

As done in logistic regression, a preprocessing operation (performed with data centering) does not affect results, even if SVM is not invariant to feature transformation, due to the regularization term; the following plot highlight what affirmed:

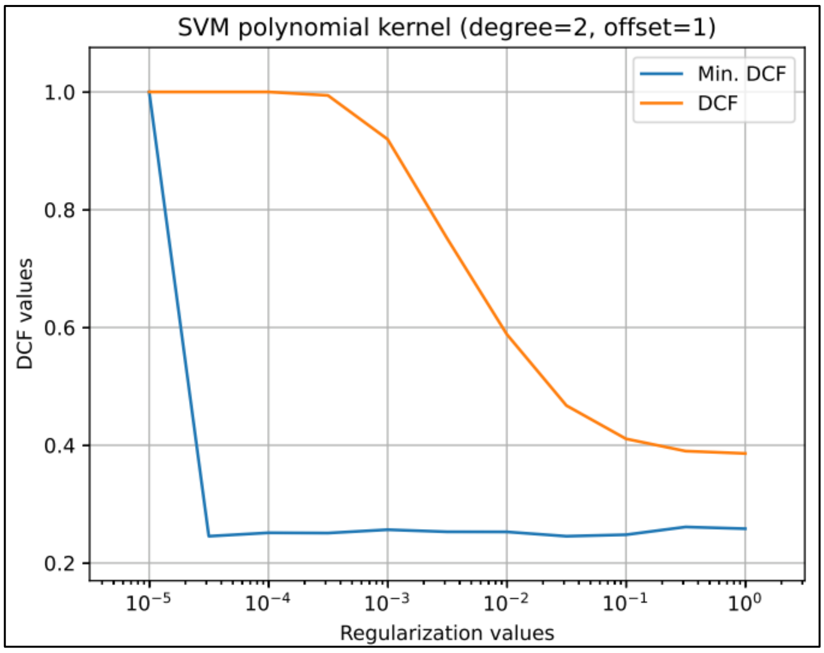


Thus, as for logistic regression, we can suppose data to be already centered with respect to training set mean.

A polynomial kernel is useful to find quadratic boundary surfaces, without expanding feature space; this task is performed by using the following kernel function:

$$k(\overline{x_1}, \overline{x_2}) = (\overline{x_1} \cdot \overline{x_2} + 1)^2$$

that is, a quadratic kernel with unitary offset term, which replaces the simple dot product used in the previous task to implement a linear kernel. The results are shown in the following plot:



We can observe that:

- The model is still involved in underfitting phenomenon for strong regularization, as in linear kernel SVM
- Both actual and minimum DCF have the same shape as the previous task, with respect to regularization values, despite being lower outside the underfitting region (C > 0.0001)
- The best result, in terms of minimum DCF, is 0.245, obtained with *C = 0.032 and K = 0*, and corresponding to an actual DCF of 0.467

The results obtained here are consistent to the previous ones:

- MVG models (full covariance and naïve Bayes variants) have a similar (slightly worse) minimum DCF, but a better calibration, since their score is probabilistic
- LR model with quadratic feature expansion performs slightly better, with a similar mis-calibration

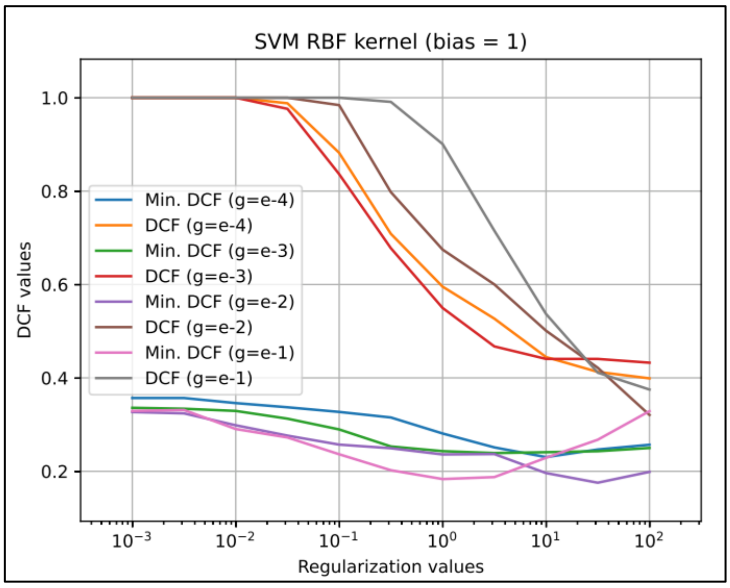
thus, the hypothesis of data to be well-separable by quadratic boundary surfaces but not by linear ones stays valid. The following table is a summary of the results of the best quadratic models up to here:

Model	Minimum DCF	Actual DCF
MVG – Full covariance	0.263	0.305
MVG – Naïve Bayes	0.257	0.302
LR – Expanded (unweighted)	**0.244**	0.497
SVM – Quadratic kernel	0.245	0.467

Another possible kernel function is the RBF (Radial Basis Function), which focuses on the relative distance among support vectors; its expression is the following:

$$k(\overline{x_1}, \overline{x_2}) = e^{-\gamma\|\overline{x_1} - \overline{x_2}\|^2}$$

where γ acts as an hyperparameter of the model; the following results are obtained by evaluating the model with unitary bias terms (*K = 1*), for different values of C and γ:



We can observe that:

- The model is still involved in underfitting issues, due to loss of probabilistic interpretation when strong regularization is performed
- In general, better results can be obtained with respect to previous kernels, in terms of minimum DCF

The results obtained are summarized in the following table, with only the best one shown, for each value of γ:

γ	K	C	Minimum DCF	Actual DCF
0.018	1	10.000	0.230	0.445
0.050	1	3.162	0.239	0.467
0.135	**1**	**31.623**	**0.175**	0.422
0.368	1	1.000	0.183	0.901

As we can see:

- For lower values of γ (wide kernels), the results are nearly equal to the ones obtained with quadratic LR and other SVM variants, both in terms of minimum and actual DCF
- For higher values of γ (narrow kernels), the results are better than others, in terms of minimum DCF, but they are affected by relevant mis-calibration

We can conclude that data features are better separable by surfaces obtained by an infinite dimensional mapping (RBF kernel), using narrow kernels, where support vectors influence only close points.

Lab10 – Gaussian Mixture Models

Each class is modeled separately, choosing the number of components from 1 to 32, but keeping the same kind of variant (full covariance or diagonal) for both classes at each step. The results, in terms of both minimum and actual DCF, are shown in the following table:

Variant	Components (False class)	Components (True class)	Minimum DCF	Actual DCF
Full	1	1	0.263	0.305
Full	1	2	0.265	0.305
Full	1	4	0.214	0.237
Full	1	8	0.185	0.196
Full	1	16	0.150	0.206
Full	1	32	0.185	0.227
Full	2	1	0.218	0.232
Full	2	2	0.216	0.234
Full	2	4	0.223	0.226
Full	2	8	0.186	0.213
Full	2	16	0.170	0.198
Full	2	32	0.186	0.227
Full	4	1	0.233	0.246
Full	4	2	0.232	0.236

Full	4	4	0.216	0.24
Full	4	8	0.189	0.206
Full	4	16	0.174	0.187
Full	4	32	0.187	0.238
Full	8	1	0.176	0.2
Full	8	2	0.181	0.191
Full	8	4	0.196	0.199
Full	8	8	0.179	0.193
Full	8	16	0.153	0.172
Full	8	32	0.175	0.19
Full	16	1	0.167	0.178
Full	16	2	0.166	0.175
Full	16	4	0.192	0.192
Full	16	8	0.175	0.205
Full	16	16	0.163	0.177
Full	16	32	0.175	0.195
Full	32	1	0.257	0.258
Full	32	2	0.256	0.26
Full	32	4	0.246	0.282
Full	32	8	0.219	0.225
Full	32	16	0.194	0.218
Full	32	32	0.234	0.25
Diagonal	1	1	0.257	0.302
Diagonal	1	2	0.261	0.305
Diagonal	1	4	0.210	0.226
Diagonal	1	8	0.205	0.221
Diagonal	1	16	0.143	0.181
Diagonal	1	32	0.137	0.197
Diagonal	2	1	0.245	0.272
Diagonal	2	2	0.249	0.267
Diagonal	2	4	0.199	0.21
Diagonal	2	8	0.204	0.209
Diagonal	2	16	0.154	0.173
Diagonal	2	32	0.144	0.181
Diagonal	4	1	0.145	0.15
Diagonal	4	2	0.154	0.161
Diagonal	4	4	0.148	0.169
Diagonal	4	8	0.140	0.152
Diagonal	4	16	0.137	0.169
Diagonal	4	32	0.141	0.168
Diagonal	8	1	0.173	0.176
Diagonal	8	2	0.176	0.187
Diagonal	8	4	0.158	0.184
Diagonal	8	8	0.146	0.181
Diagonal	8	16	0.132	0.149
Diagonal	**8**	**32**	**0.131**	0.152
Diagonal	16	1	0.207	0.222

Diagonal	16	2	0.204	0.222
Diagonal	16	4	0.196	0.205
Diagonal	16	8	0.198	0.214
Diagonal	16	16	0.162	0.177
Diagonal	16	32	0.164	0.18
Diagonal	32	1	0.175	0.196
Diagonal	32	2	0.175	0.197
Diagonal	32	4	0.193	0.198
Diagonal	32	8	0.193	0.199
Diagonal	32	16	0.185	0.207
Diagonal	32	32	0.177	0.199

We can observe:

- In general, results are better than MVG and logistic regression, whereas they are comparable with non-linear SVM ones: it is consistent to previous results, since data are better separable by non-linear boundary surfaces.
- The best result, in term of minimum DCF, is 0.131, corresponding to a diagonal GMM with 8 components for false class and 32 components for true class, and actual DCF is 0.152.
- As expected, score mis-calibration is lower than SVM and LR, since GMM produces LLR-like scores, which have probabilistic interpretation.
- The general trend of minimum DCF is increasing both for large and small values of the number of components, with a minimum point corresponding to the central part of the range, in particular with respect to the false class in the full covariance variant: this phenomenon indicates an overfitting issue, since the higher the number of components, the larger the number of parameters to estimate.
- For the same reason as the previous point, the diagonal covariance GMM performs in general better than the full covariance variant, achieving the best result for the GMM model; moreover, here the overfitting phenomenon is less relevant, due to the lower number of components to estimate.

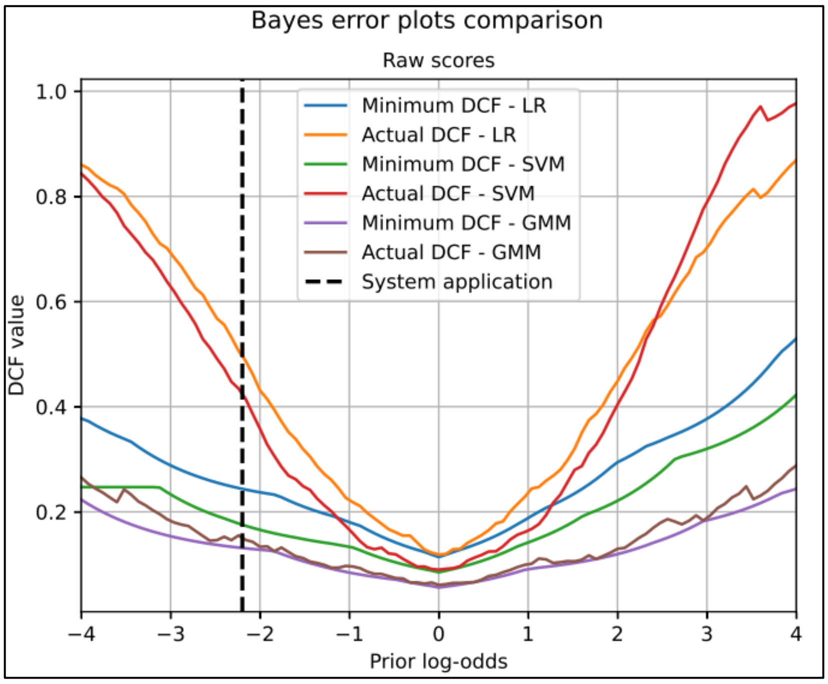
Thus, for the target application $\pi_T = 0.1$, the best results (one setup for each model shown) are the following:

Model	Setup	Minimum DCF	Actual DCF
Logistic regression	Variant: quadratic $\lambda = 0.032$	0.244	0.497

SVM	Kernel: RBF
	C = 31.623
	K = 1
	$\gamma = 0.135$
		0.175	0.422

GMM	**Variant: diagonal** **Components (false): 8** **Components (true) 32**	**0.131**	0.152

By considering all possible applications and comparing these three models, we obtain the following plot:



thus:

- In terms of minimum DCF, the relative ranking of the models is preserved for each possible application, that is the best model is GMM, followed by SVM and then by logistic regression, independently by the application chosen
- In terms of actual DCF, GMM has the lowest one for the full range of applications analyzed, since it is both the best performing and the best calibrated one; both SVM and logistic regression have poor calibrated scores, with the former having a larger actual DCF than the latter one, for applications that are strongly unbalanced to the true class (larger value of prior log-odds)
- Since there are no applications for which a model has an actual DCF greater than one, then none of the models are harmful for some applications; in fact, this means that each model has a DCF (not normalized) smaller than the dummy risk, so each of the models is better (for each application) than using the best dummy system.

Lab11 – Score calibration and fusion, system evaluation

Calibration and fusion

Each of the three selected models is calibrated using a K-fold cross validation approach, with *K = 5*. The calibrator is a prior-weighted logistic regression model, with a training prior chosen using cross validation; predictions basing on calibrated scores are performed using the application prior $\pi_T = 0.1$.

For each model, each calibrator is trained, used and validated (in terms of actual DCF), with a training prior spanning from 0.01 to 0.99, with a step of 0.01.

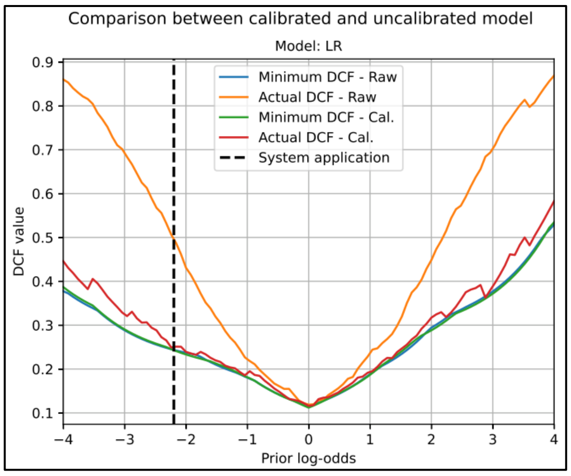
Then, models scores are stacked to form a unique score matrix, used in a score fusion task, performed with the same K-fold cross validation setup.

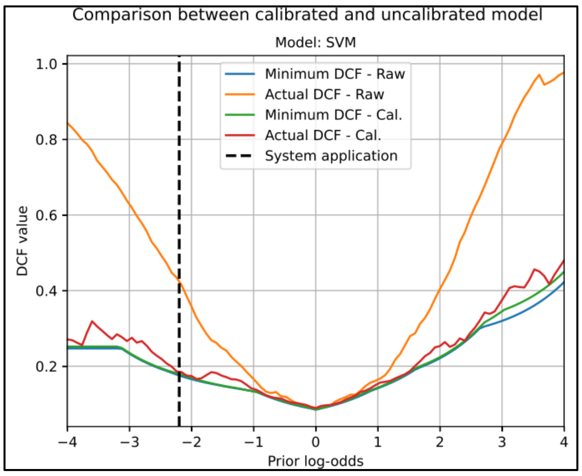
The following table sums up the best calibration setup found for each of the selected models, and their fusion:

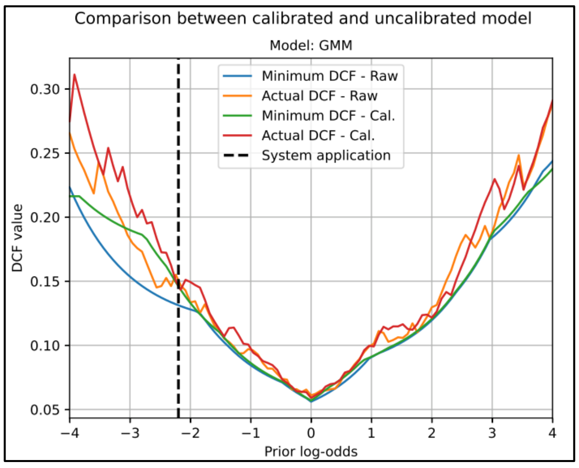
Model	Minimum DCF	Actual DCF	Training prior
Logistic regression	0.243	0.246	0.89
SVM	0.179	0.179	0.21
GMM	0.146	0.147	0.99
Fusion	0.119	**0.120**	**0.15**

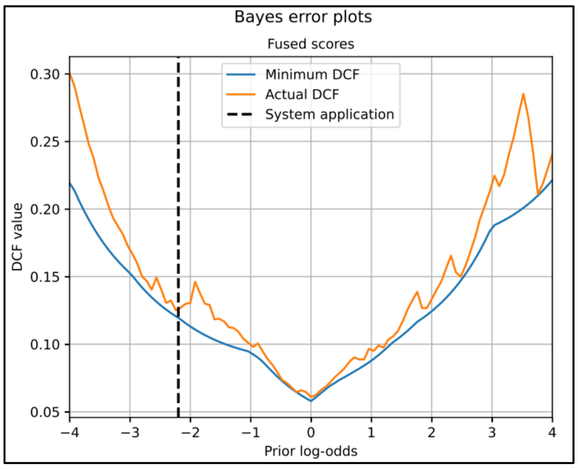
Since fusion has the best (lowest) calibrated actual DCF score, then it is chosen as final, delivered system.

By analyzing the performance of these models on different applications, the following plots are obtained:









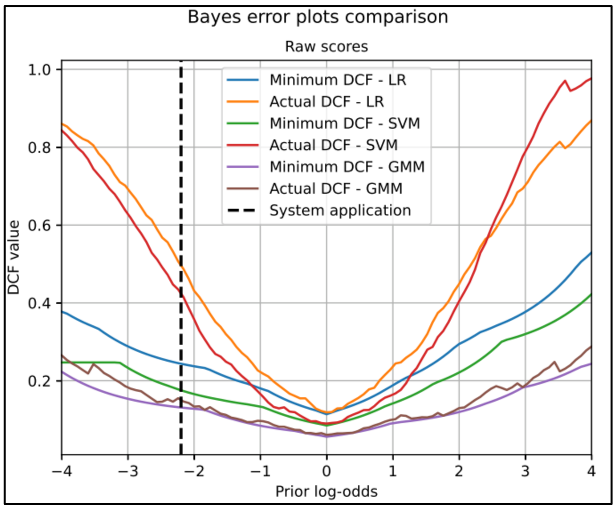
As we can see:

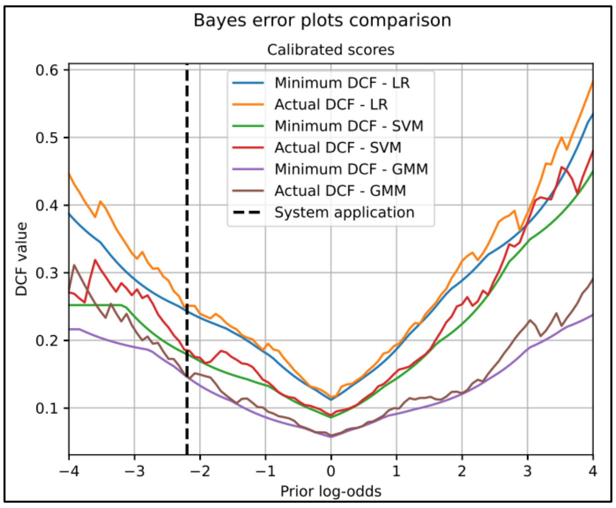
- Calibration works well for both SVM and logistic regression on the full range of applications analyzed, in particular on the target one
- Calibration does not affect significantly GMM mis-calibration, but the value of minimum DCF becomes significantly larger for some applications, as the corresponding actual DCF: this

could happen because the calibration transformation is affine on each fold, but not on the whole dataset

- Fused scores are well-calibrated only for some ranges of applications, including the target one: fusion process improves actual DCF with respect to single system, it means that chosen models extract different useful information from data, which are relevant to perform a good classification if combined (fused) together

Here follows a comparison among models before and after calibration:





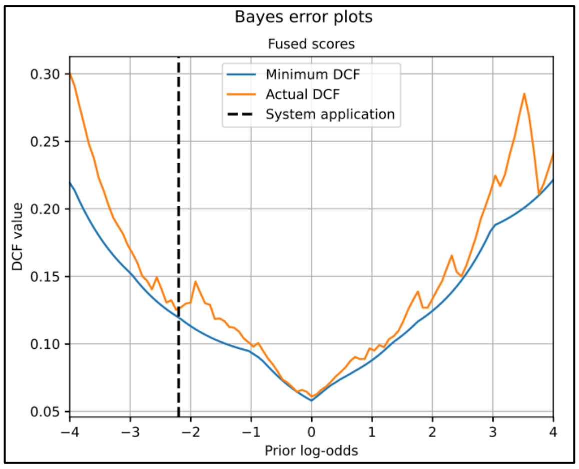
We can observe the improvements carried by calibration process over the full range of applications, for SVM and logistic regression; however, GMM results less calibrated than before, for applications with strongly unbalanced priors: when the false class is highly prevailing (low application prior), it performs worse than SVM.

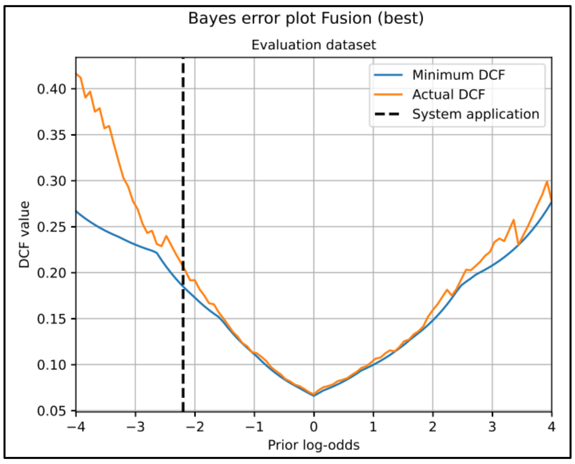
Evaluation

By applying the delivered system (fusion) on evaluation dataset, the following result is obtained:

Model	Minimum DCF	Actual DCF
Fusion	0.185	0.205

thus, the performance results to be worse, probably due to differences into data features in the evaluation set, involving one between classification or calibration part, or both. System performance across different applications is the following (left plot on validation, for comparison):





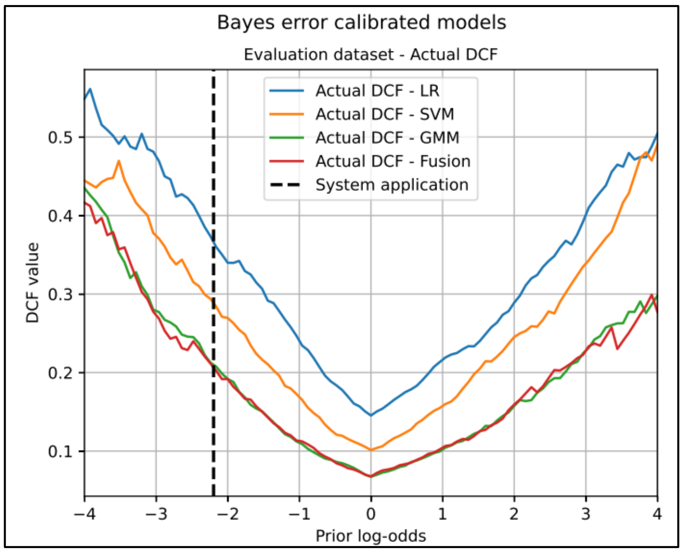
As we can see, scores are in general better calibrated on evaluation than validation set, but mis-calibration is higher on evaluation set, both for target application and for those corresponding to low effective priors.

By performing the same process on other systems, the following results are obtained:

Model	Minimum DCF	Actual DCF
Logistic regression	0.351	0.367
SVM	0.262	0.285
GMM	0.184	0.213

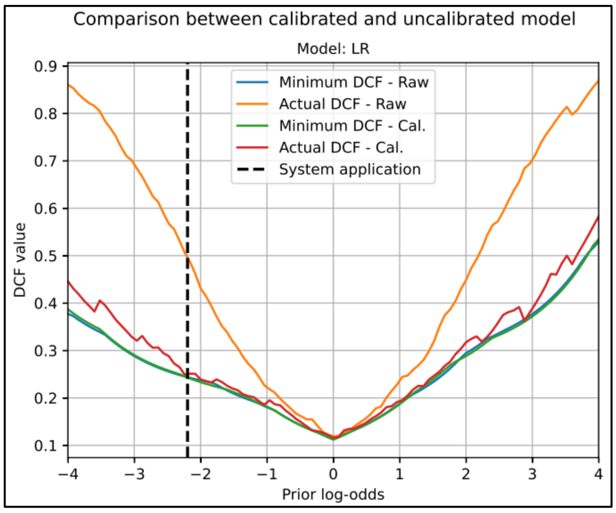
We can observe that all these models, as the fusion, are involved in a growth of actual DCF; however, as the fusion, they are still well-calibrated, despite a slight worsening with respect to the measure performed on validation set.

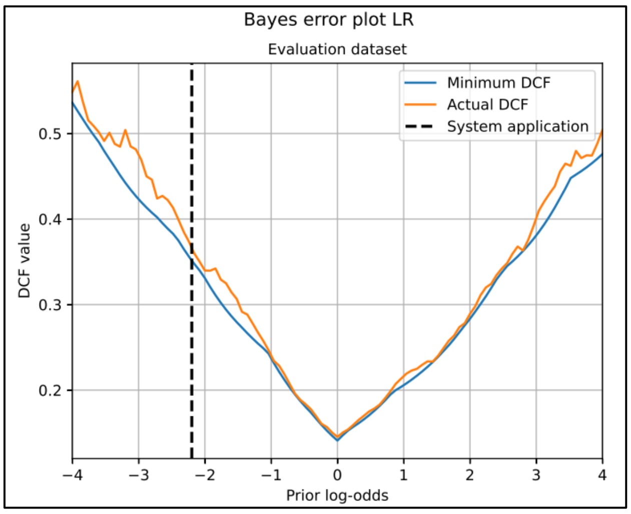
The following plot compares actual DCF of all the systems, for the evaluation set:

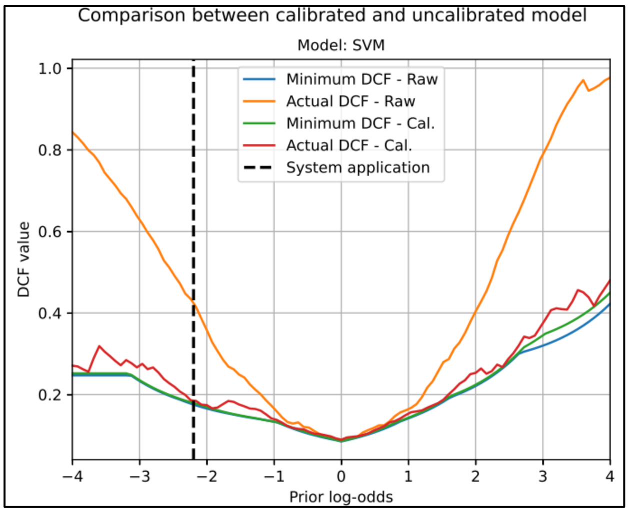


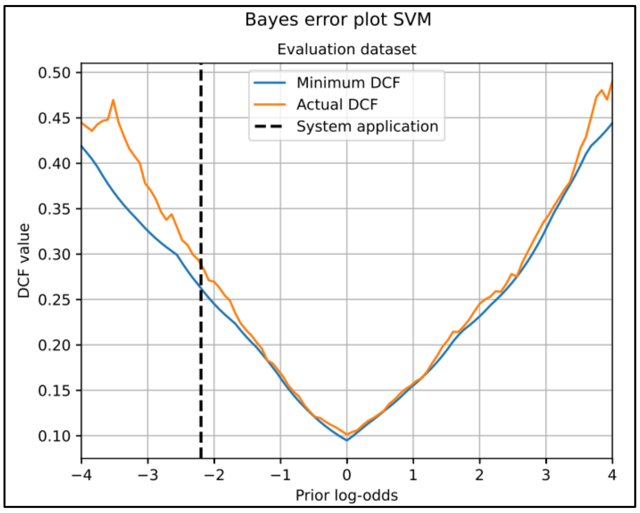
We can observe that, for the target application, relative rankings are the same as observed on the validation set, with fusion as the best system, followed by GMM (nearly equal as fusion with respect to actual DCF, on the full range of applications), SVM and logistic regression, in this order. However, for low application priors, fusion, GMM and SVM have almost equivalent performances. Moreover, since fusion in still the best model, we can affirm that the previously taken choice has been effective.

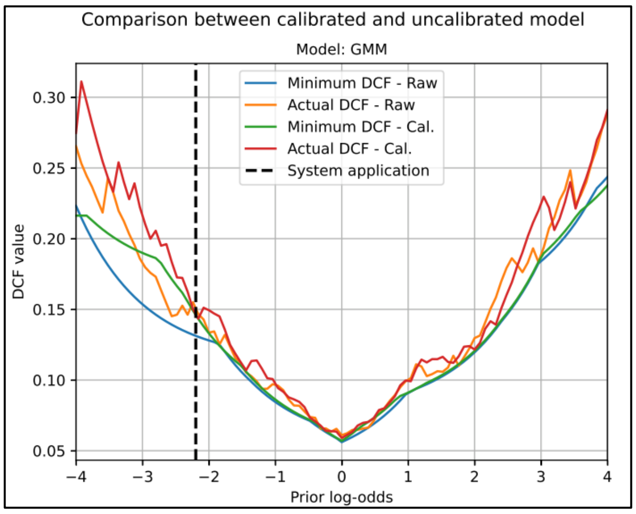
The following pairs of plots show how single systems behave (in terms of actual and minimum DCF) as application changes (left plots on validation set, for comparison):

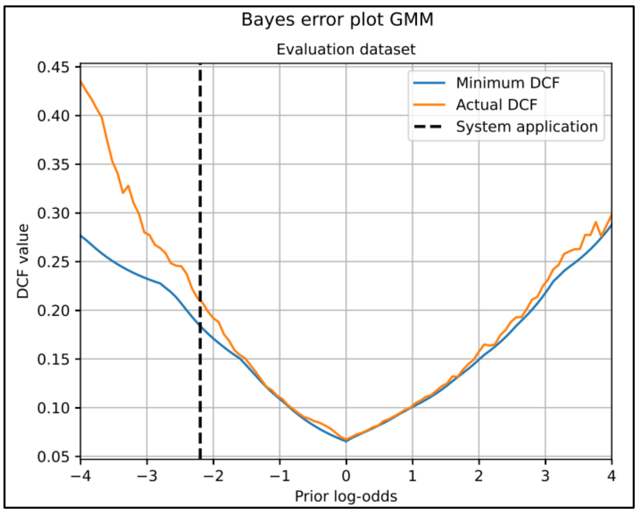












We can observe that, given the application prior, calibration is effective only for logistic regression, while other systems have more mis-calibrated scores corresponding to target prior log-odds; by considering the full range of applications, the scores are well-calibrated, except for the ones with low priors in SVM and GMM.

By applying a single approach (that is, logistic regression), in all its variants and setups previously employed (part *Lab8 - Logistic regression*), on the evaluation set, the following results are obtained (setup with filtered dataset is avoided, since it was used only to prove strong overfitting issues):

Model	λ	Minimum DCF	Actual DCF
Unweighted LR	0.03	0.505	0.616
Prior-weighted LR	0.32	0.506	1.000
Prior-weighted LR (data centering)	0.32	0.506	1.000
Quadratic LR (unweighted)	**0.01**	**0.347**	0.377

We can observe that:

- For each setup, the results obtained (in terms of minimum DCF, since no calibration is performed here) are worse than those obtained on validation set.
- Logistic regression on expanded feature space is still the best performing setup; moreover, it is the setup with the least worsening of performance.
- Unweighted and prior-weighted setups are here almost equivalent, in terms of minimum DCF, as on the validation split
- Data centering does still not affect classification, probably since evaluation data are already centered with respect to training set mean, as the validation data

In conclusion, limitely to logistic regression, there are not setups that are more promising than the chosen one (quadratic, unweighted), thus this choice has been optimal both for validation and evaluation data, by supposing that calibration transformation would be sufficiently effective.