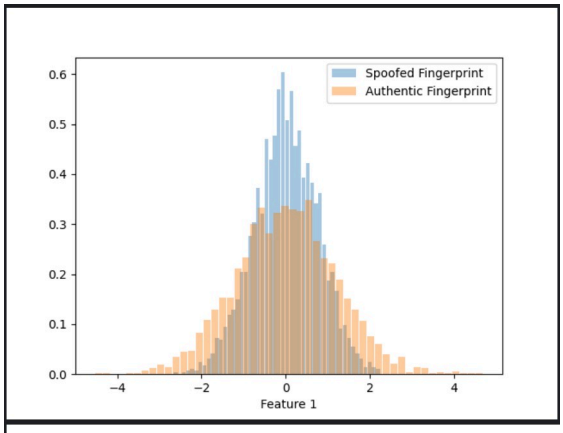
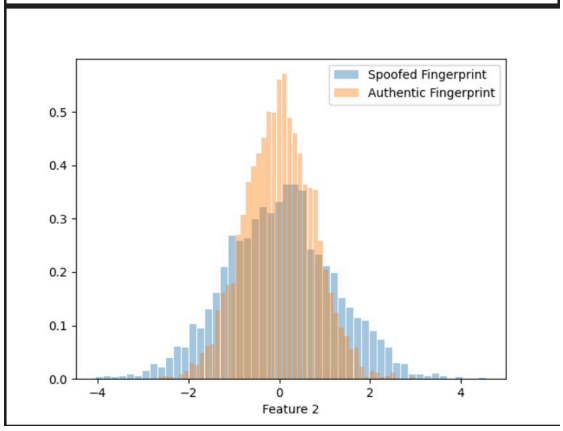
MACHINE LEARNING AND PATTERN RECOGNITION 2023/2024 COURSE: PROJECT REPORT

GREGORIO NICORA, s310820

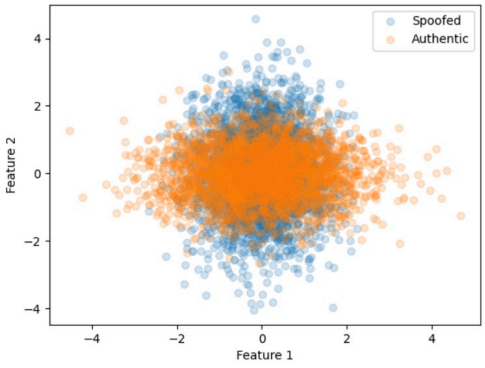
LAB 02

1. As reported in the graphs below, we can notice that the two classes overlap over feature 1 and feature 2. Though, we can notice that while plotting the two classes over feature 1 the Spoofed class is more focused around the 0 value and the variance is lower with respect to the Authentic class, resulting in a higher peak in 0 of the Spoofed class.
   Plotting the two classes over feature 2, this behaviour is reversed: Authentic class is more concentrated, with a peak, in 0 and has a lower variance with respect to the Spoofed class.

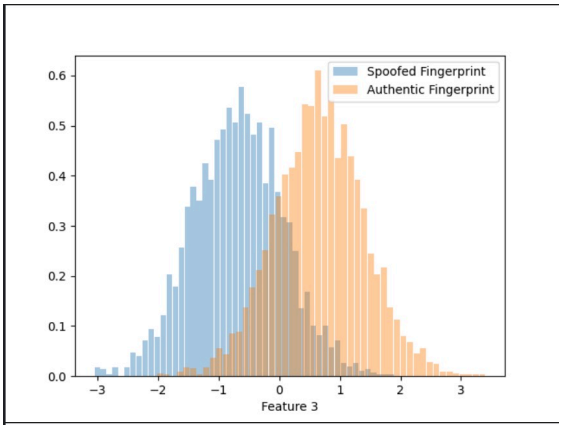


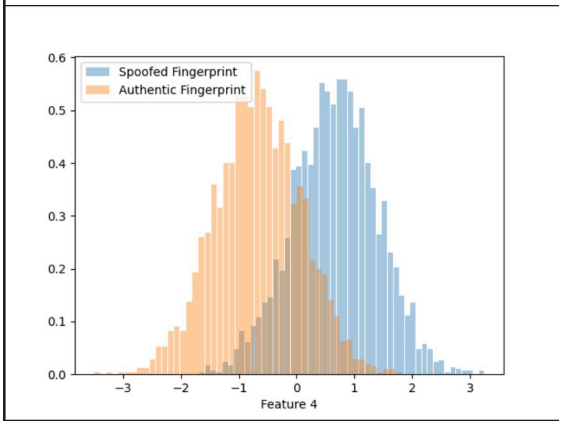


   This behaviour can be confirmed also by the scatter plot in which we plot all the samples belonging to the two classes and we can notice that the Spoofed class is more spread over feature 2 while the Authentic class is more spread over feature 1.

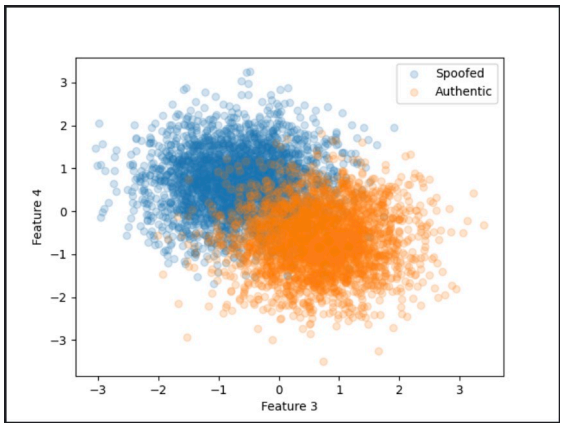


2. As we can notice from the graphs below, along feature 3 the two classes are more distinguishable, although still being overlapped around 0. We can also see that the two distributions are overall pretty similar to each other over feature 3, despite being shifted from 0 in opposite directions. The peak of the Spoofed class can be observed around -1 while the one of the Authentic around +1.
   Almost the same behaviour, but mirrored along the x-axis, can be noticed if we plot the two distributions over feature 4.

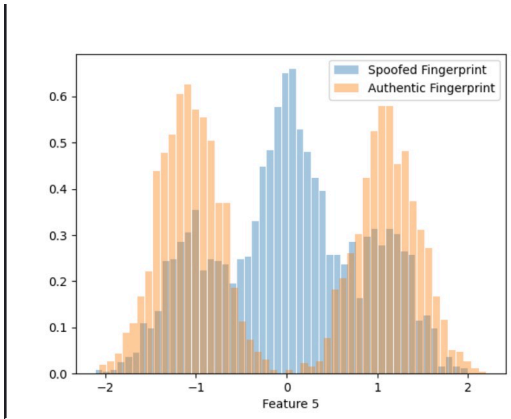


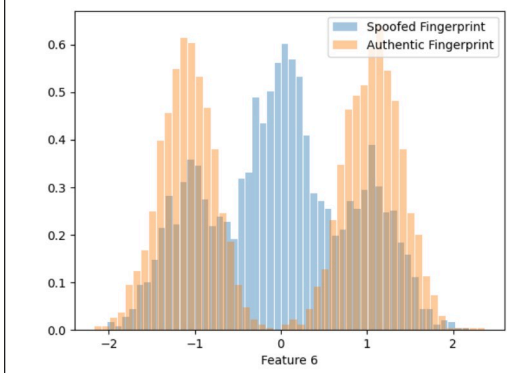


From the scatter plot over feature 3 vs feature 4, the two classes distributions can be distingushed from each other although still overlapping over the principal diagonal of the plot.

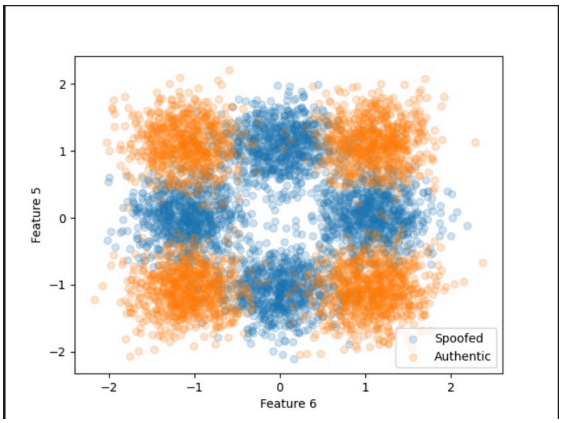


3. As can be seen blow, over feature 5 and feature 6 the two classes present almost identical distributions. Two peaks in around -1 and +1 can be clearly noticed for the Authentic class while three peaks can be seen around -1, 0 and +1. The two distributions overlap around -1 and +1 where they both have peaks, while around 0 we can notice almost only the presence of the Spoofed class distribution.



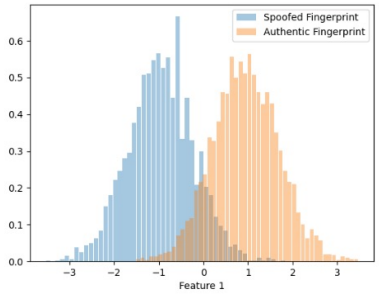


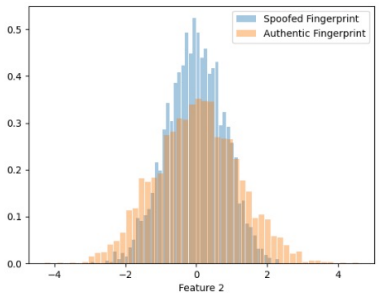
The behaviour described above is confirmed by the scatter plot of the two classes over the two features. In particular, it can be noticed that there are zones (4 clusters) around the values in which the histograms have peaks with only orange samples and others (other 4 clusters) also along the values where the histograms have peaks with only blue ones. Plotting an histogram over a particular feature make this samples "sum up" their presence over that particular direction.

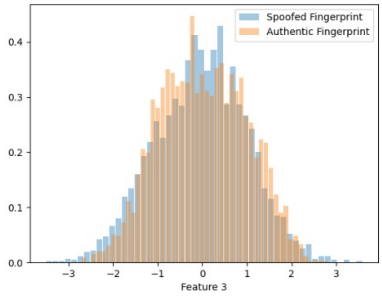


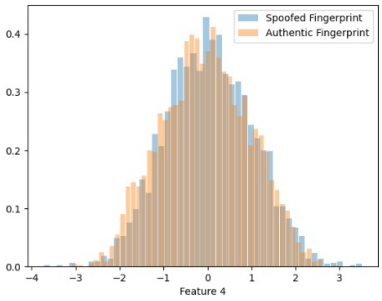
LAB 03: PCA and LDA

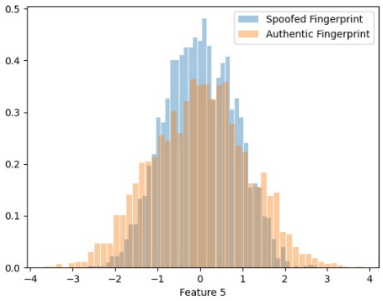
Principal Component Analysis (PCA) is an unsupervised dimensionality reduction technique that aims at finding the directions of maximum variance in a dataset. The goal is to find a lower-dimensional subspace to project the original data onto, while preserving as much variance as possible.

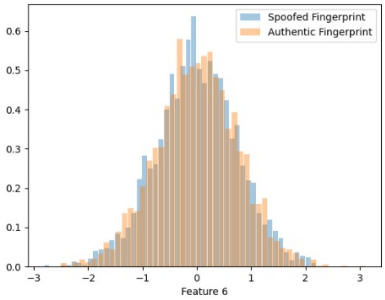








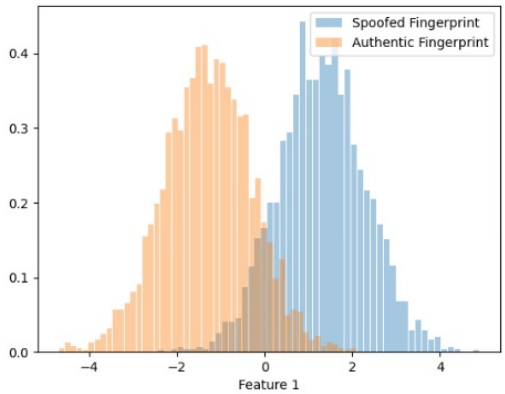




As we can see from the histograms, the first principal component captures most of the variance and it separates decently the two classes. The other principal components overlap, so they are not very useful for classification.

Linear Discriminant Analysis (LDA), on the other hand, is a supervised dimensionality reduction technique that aims at finding the directions that maximize the separation between dataset classes.

Only one linear discriminant is found because the number of classes is 2. The linear discriminant separates the two classes partially. It is comparable to the first principal component of PCA.



Both cases simplify the dataset as much that the well-distinctive cluster of the last two features is lost.
So, the preprocessing techniques could be useful to speed up the computation but they could also lose important information. In very simple dataset as this one, with only 6 dimensions, presumably the best choice is to use the original dataset.

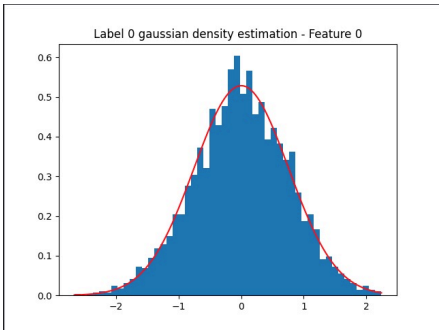
We further investigate in into the classification task taking advantage of the LDA model as classifier. Only applying LDA as a classifier would result in an error rate of 9.3 %.

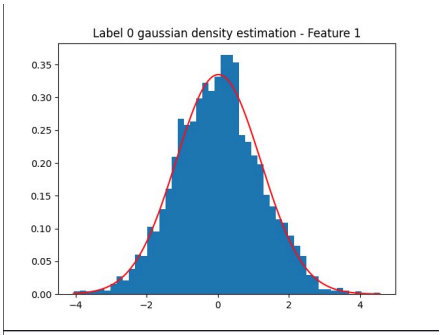
As a result, we want to try applying also PCA before using LDA as classificator model:

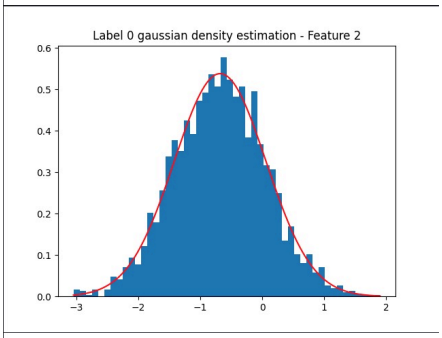
MODEL	error rate
pca 1	9.35%
pca 2	9.25%
pca 3	9.25%
pca 4	9.25%
pca 5	9.3%
pca 6	9.3%

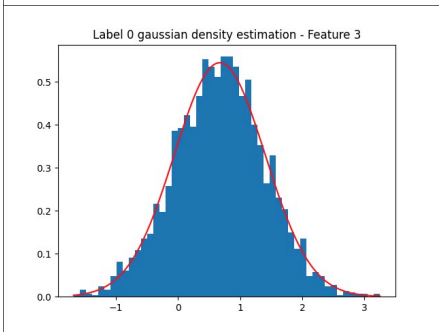
LAB 04: Multivariate Gaussian Density

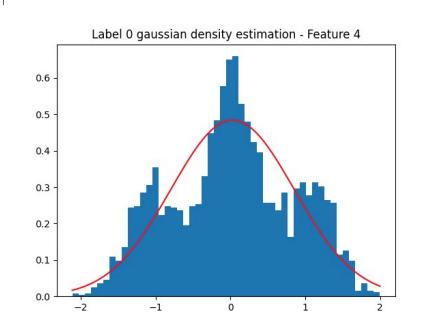
In order to understand the potential power of Multivariate Gaussian Density we try to take a glance at the correspondence between the features distribution of our dataset and the Gaussian density function.

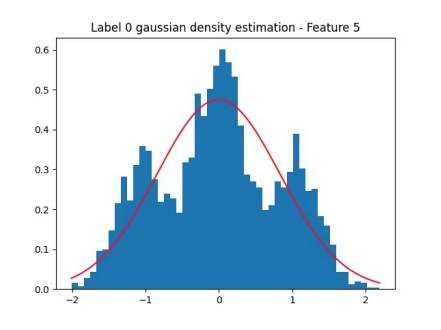




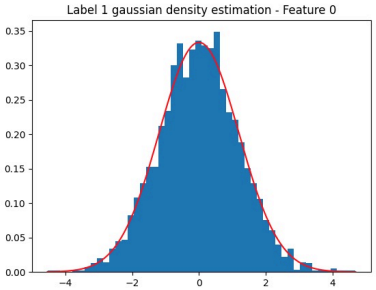


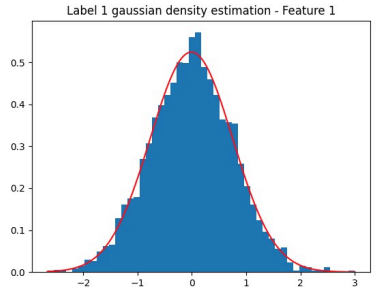


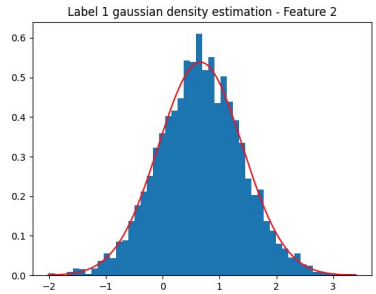


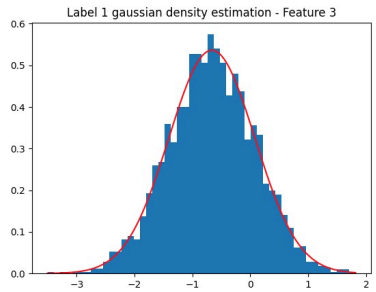


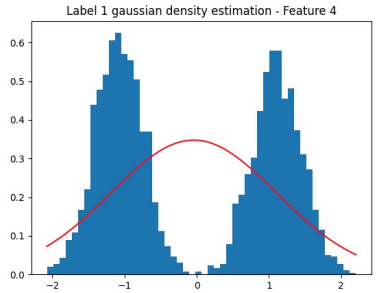
As we can see from the graphs above, in class 0 samples, features 0 to 3 could be modeled with a good grade of approximation by a unimodal Gaussian density function. Features 4 and 5, on the other hand, would result in a much less precise approximation.

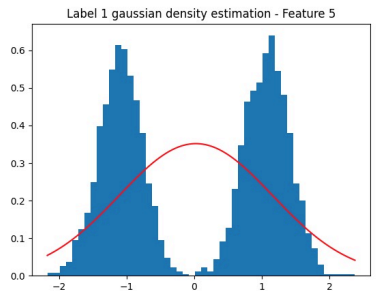

Label 1 gaussian density estimation - Feature 0


Label 1 gaussian density estimation - Feature 1


Label 1 gaussian density estimation - Feature 2


Label 1 gaussian density estimation - Feature 3


Label 1 gaussian density estimation - Feature 4


Label 1 gaussian density estimation - Feature 5

Taking into consideration class 1 samples, we can get to similar conclusions: being features 0 to 3 approximable with a good grade of precision by a unimodal Gaussian density distribution while features 4 and 5 having similar mean to what their unimodal Gaussian density function approximation would be but a completely different distribution along respective axis.

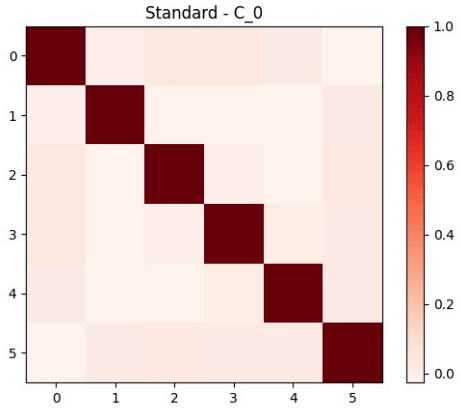
LAB 05: Multivariate Gaussian Model

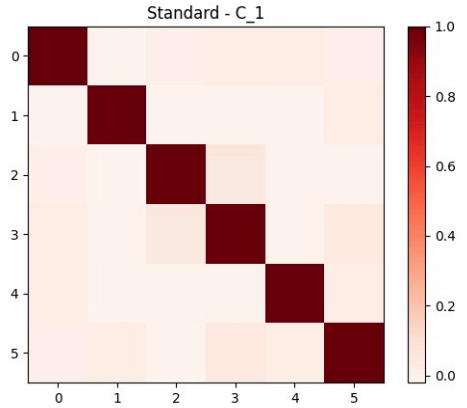
We proceed on evaluating the goodness of different variants of the Multivariate Gaussian models: standard, naive-Bayes and tied.

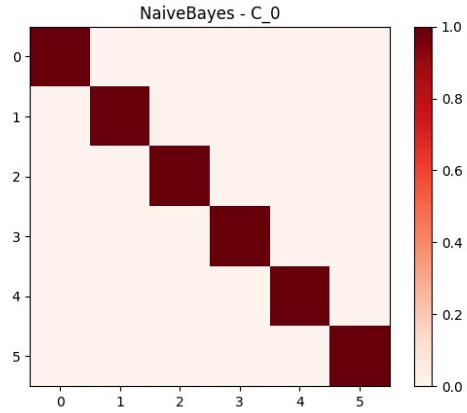
MVG assumes that features can be jointly modeled by Gaussian distributions.
As previously seen in LAB 04, we do not expect this model to perform that well because of the distribution of the samples along feature 4 and 5 not following Gaussian distributions pattern.

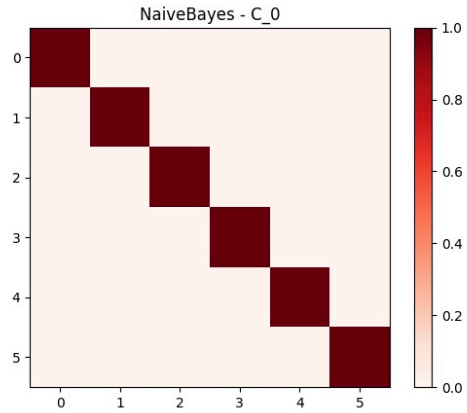
MODEL TYPE	ERROR RATE
STANDARD MVG	7 %
NAIVE-BAYES MVG	7.2 %
TIED MVG	9.3 %

Error rates show that the tied version of MVG performs worse thant the others. This is caused by the covariance matrix changing very much throughout the different samples' features. This result in not being able to approximate the classes distributions using one single covariance matrix.









Another approach to understand why standard and naive Bayes MVG models give results that are so close can be found in analyzing the Pearson correlation coefficient between features for each class. As it can be viewed above, features show very low correlation with each other in both classes, so the Naive Bayes assumption can hold pretty good and make its model perform very similarly to the standard one.

Starting from lab 04, where we plotted unimodal Gaussian distributions over each feature of class samples, we can easily discover why, though working significantly better than the tied version, both standard and naive Bayes models still struggle to model the dataset distribution with a high grade of accuracy. IN fact, we saw that features 4 and 5, do not have a distribution that can be modeled seamlessly with a Gaussian distribution. Thus, applying feature selection and discarding the last two features of each data sample could lead to better results.

MODEL TYPE	ERROR RATE
STANDARD MVG	7.95 %
NAIVE-BAYES MVG	7.65 %
TIED MVG	9.5 %

For all the models, the performance has worsen a bit with respect to the 'full features models' so, we can conclude that, despite having distribution that are not well approximated by Gaussian density function, features 4 and 5 still give some knowledge to our models.

As a further analysis wwe take into consideration models using only features respectively features 0 and 1 and features 2 and 3

MODEL TYPE	ERROR RATE
STANDARD MVG	36.5 %
NAIVE-BAYES MVG	36.3 %
TIED MVG	49.45 %

MODEL TYPE	ERROR RATE
STANDARD MVG	9.45 %
NAIVE-BAYES MVG	9.45 %
TIED MVG	9.4 %

The performance of the first analysis(first two features of the dataset) is pretty disastrous, this can be motivated by the fact that classes plotted on those two features aren't separable at all, as can be also seen from the relative scatter plot of lab 02.
On the other hand, the second analysis (features 2 and 3) gives pretty good results, similar to the full dataset models. Also this case can be motivated by the fact that the two classes means along these features are very different, resulting in more separable classes as can be seen in the relative scatter plot of lab 02.

MODEL TYPE	PCA1	PCA2	PCA3	PCA4	PCA5	PCA6
STANDARD MVG	9.25 %	8.8 %	8.8 %	8.05 %	7.1 %	7 %

MODEL TYPE	PCA1	PCA2	PCA3	PCA4	PCA5	PCA6
NAIVE-BAYES MVG	9.25 %	8.85 %	9 %	8.85 %	8.75 %	8.9 %
TIED MVG	9.35 %	9.25 %	9.25 %	9.25 %	9.3 %	9.3 %

In conclusion, analyzing MVG variants results over the dataset on which we applied PCA (with various dimensions) we can see that applying PCA before this model does not bring any significant value. The only model that performs more or less the same throughout all the different grades of PCA is the tied one which had already a poor behaviour with the full dataset.

LAB 07: Model Evaluation

During our analysis we're gonna take advantage of some metrics to evaluate the goodness of our classification models. In particular, we're gonna use Detection Cost Function (DCF) that is a metric which combines the error rate of a model and the costs we assigned to misclassification errors. We're also gonna evaluate minimum DCF which is the DCF value while selecting the optimal threshold (not theoretical).

The working points we are evaluating are:
• (π_T, C_{FP}, C_{FN}) = (0.5, 1, 1)
• (π_T, C_{FP}, C_{FN}) = (0.1, 1, 1)
• (π_T, C_{FP}, C_{FN}) = (0.9, 1, 1)
• (π_T, C_{FP}, C_{FN}) = (0.5, 1, 9)
• (π_T, C_{FP}, C_{FN}) = (0.5, 9, 1)

For working point = (0.5, 1, 1) results are:

MODEL TYPE	METRIC	NO PCA	PCA1	PCA2	PCA3	PCA4	PCA5	PCA6
STANDARD MVG	Error DCF minDCF	7% 0.14 0.130	9.25% 0.177 0.177	8.8% 0.181 0.172	8.8% 0.178 0.173	8.05% 0.162 0.154	7.1% 0.145 0.132	7% 0.146 0.130
NAIVE-BAYES MVG	Error DCF minDCF	7.2% 0.144 0.131	9.25% 0.177 0.177	8.85% 0.175 0.172	9% 0.180 0.176	8.85% 0.180 0.174	8.75% 0.181 0.173	8.9% 0.178 0.173
TIED MVG	Error DCF	9.3% 0.186	9.35% 0.177	9.25% 0.179	9.25% 0.189	9.25% 0.183	9.3% 0.188	9.3% 0.185

MODEL TYPE	METRIC	NO PCA	PCA1	PCA2	PCA3	PCA4	PCA5	PCA6
	minDCF	0.181	0.177	0.179	0.183	0.182	0.181	0.181

For working point = (0.1, 1, 1) results are:

MODEL TYPE	METRIC	NO PCA	PCA1	PCA2	PCA3	PCA4	PCA5	PCA6
STANDARD MVG	Error DCF minDCF	7% 0.305 0.263	9.25% 0.407 0.369	8.8% 0.387 0.353	8.8% 0.162 0.154	8.05% 0.363 0.300	7.1% 0.308 0.271	7% 0.306 0.260
NAIVE-BAYES MVG	Error DCF minDCF	7.2% 0.302 0.257	9.25% 0.407 0.369	8.85% 0.387 0.356	9% 0.180 0.174	8.85% 0.395 0.362	8.75% 0.403 0.354	8.9% 0.398 0.351
TIED MVG	Error DCF minDCF	9.3% 0.406 0.363	9.35% 0.400 0.369	9.25% 0.400 0.363	9.25% 0.183 0.182	9.25% 0.415 0.361	9.3% 0.412 0.365	9.3% 0.411 0.363

For working point = (0.9, 1, 1) results are:

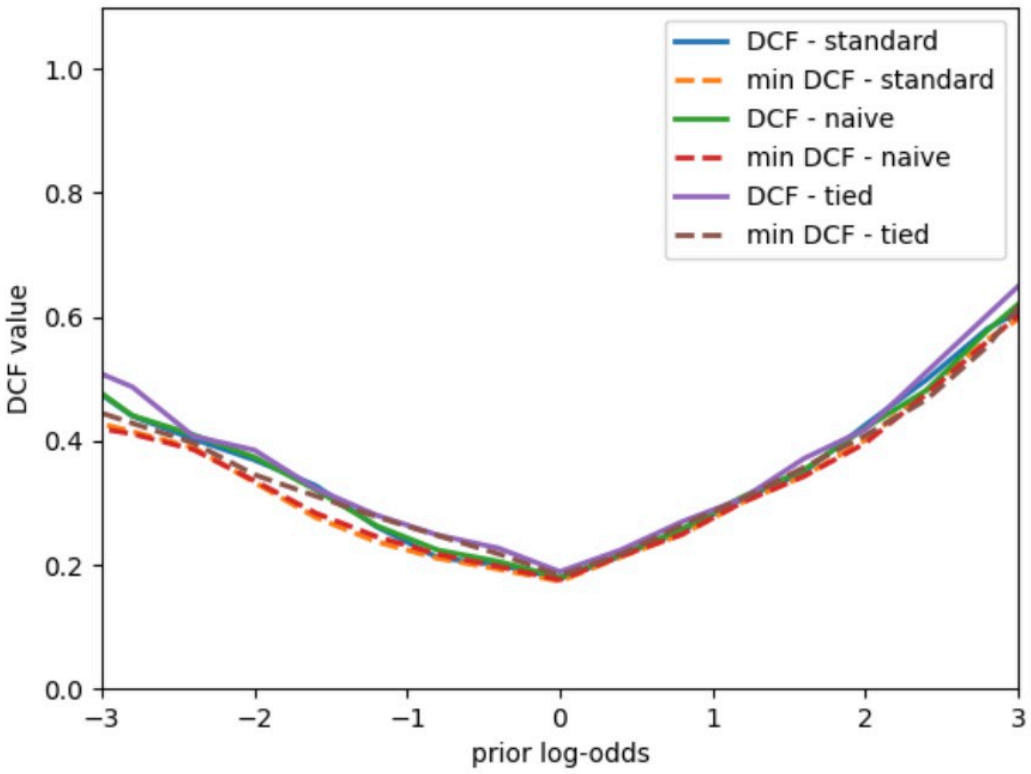
MODEL TYPE	METRIC	NO PCA	PCA1	PCA2	PCA3	PCA4	PCA5	PCA6
STANDARD MVG	Error DCF minDCF	7% 0.400 0.342	9.25% 0.467 0.434	8.8% 0.450 0.438	8.8% 0.444 0.415	8.05% 0.444 0.415	7.1% 0.397 0.351	7% 0.384 0.382
NAIVE-BAYES MVG	Error DCF minDCF	7.2% 0.389 0.351	9.25% 0.467 0.434	8.85% 0.441 0.432	9% 0.466 0.430	8.85% 0.466 0.430	8.75% 0.468 0.435	8.9% 0.458 0.437
TIED MVG	Error DCF minDCF	9.3% 0.463 0.442	9.35% 0.472 0.434	9.25% 0.462 0.435	9.25% 0.450 0.444	9.25% 0.450 0.444	9.3% 0.308 0.271	9.3% 0.455 0.442

Working points (0.5, 1, 9) and (0.5, 9, 1) result in two effective priors already tested in the previous tables so won't be explicitly reported.

For the target application (0.5, 1, 1), the best results in terms of minDCF is given by the Standard MVG Model. It can be seen that actDCF actually differs a little bit from the minDCF but I wouldn't say that the difference could be covered by a calibration model being the difference so small.

For the given applications, (0.1, 1, 1) and (0.9, 1, 1) are the two working points in which there is a slight miscalibration (difference between actDCF and minDCF).

The best configuration for the π_tilde = 0.1 is given by the Standard MVG Model with PCA = 3. Below a comparison of the three MVG variants given PCA = 3 over the target application π_tilde = 0.1

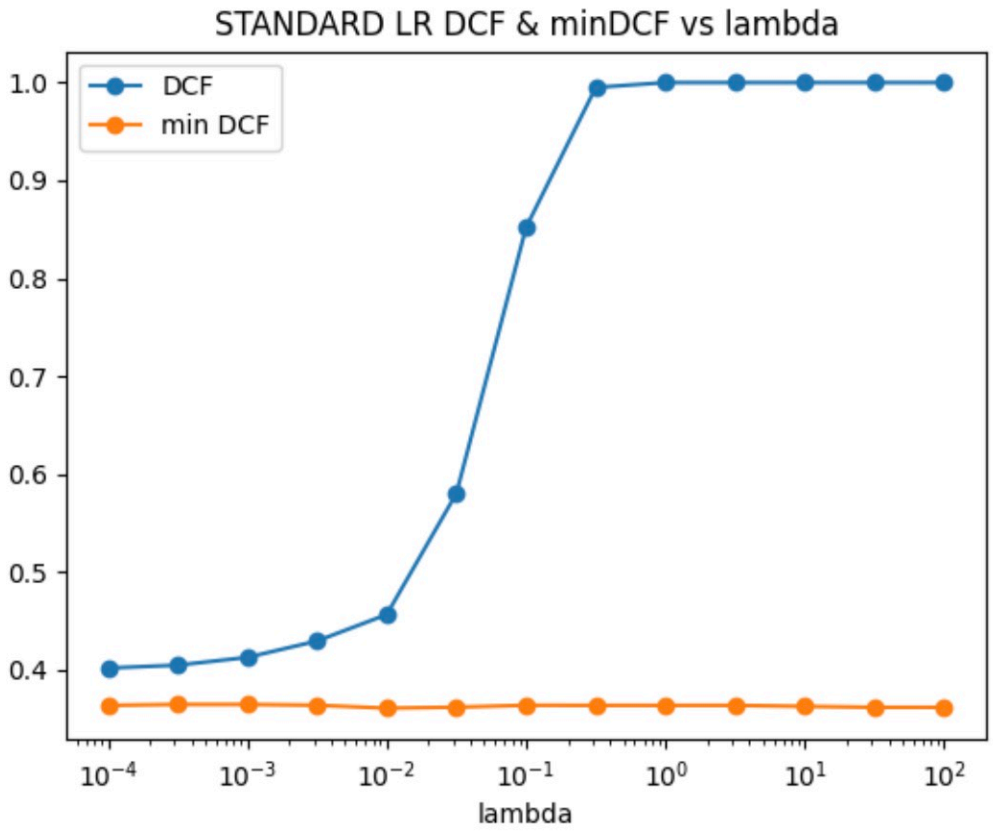


To conclude, MVG gives scores that can be considered overall always well-calibrated, as reported by the graph above.

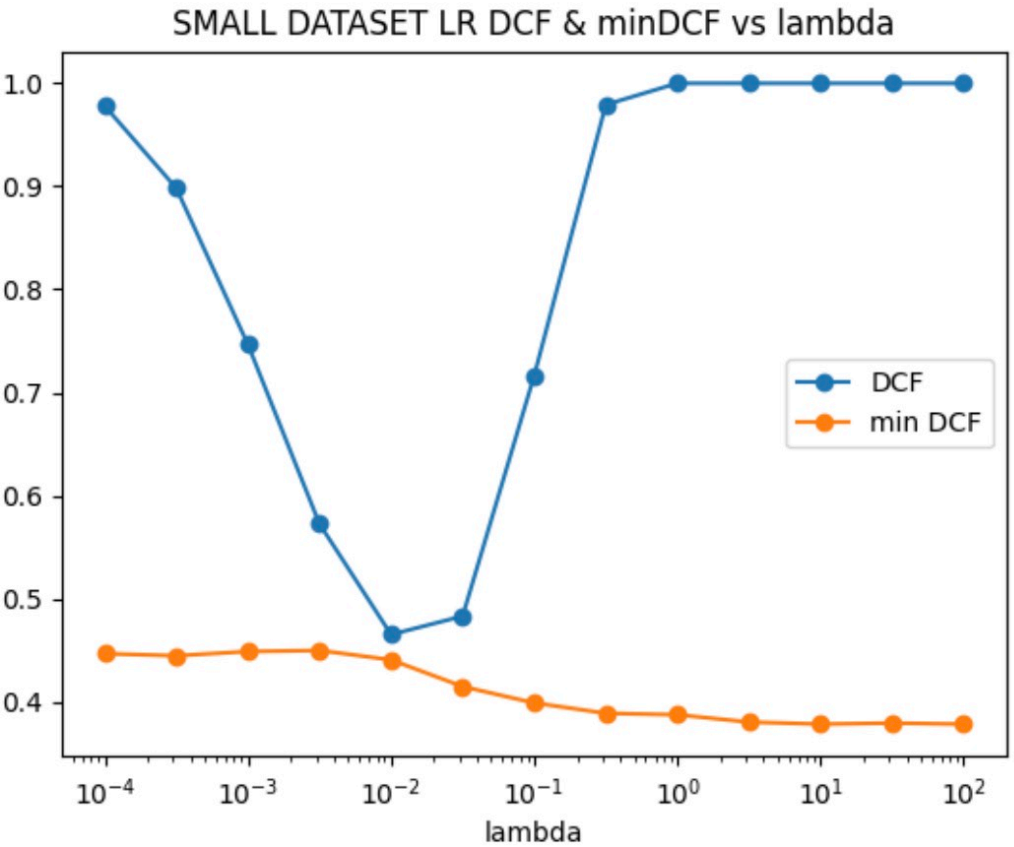
LAB 08: Logistic Regression

Logistic Regression (LR) is a classification model that aims at estimating a separation surface parameters for, in this case, a binary problem.
We're gonna explore the performance of both linear and quadratic LR models.

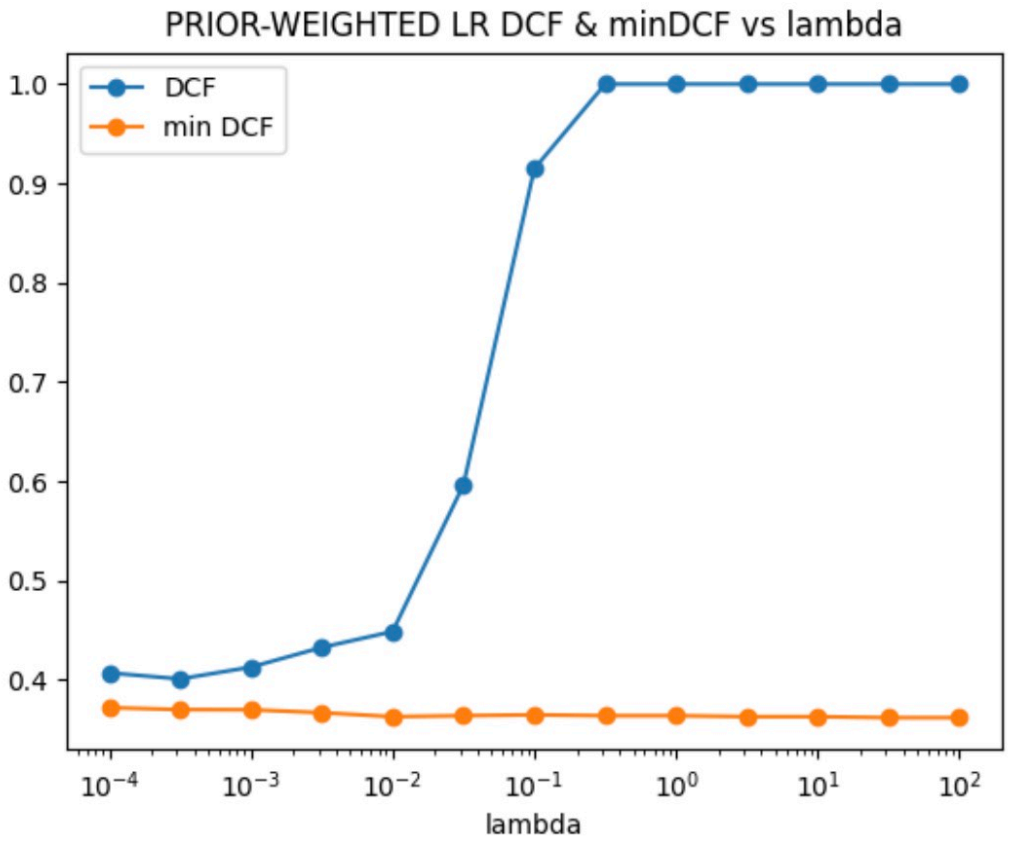
First of all we're gonna explore the behaviour of this model when varying its hyperparameter λ (regularization term) for the primary application $\pi_T = 0.1$ through computing actDCF and minDCF.



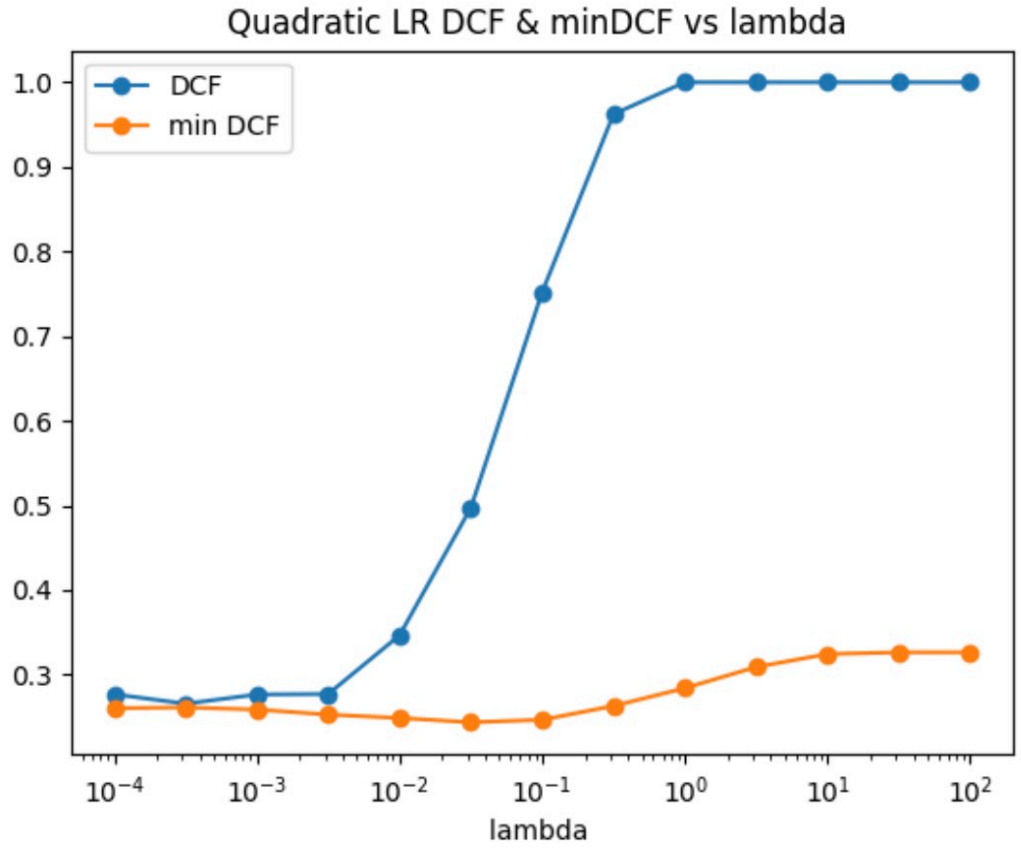
We also want to look at the behaviour using 1/100 of the original dataset size.



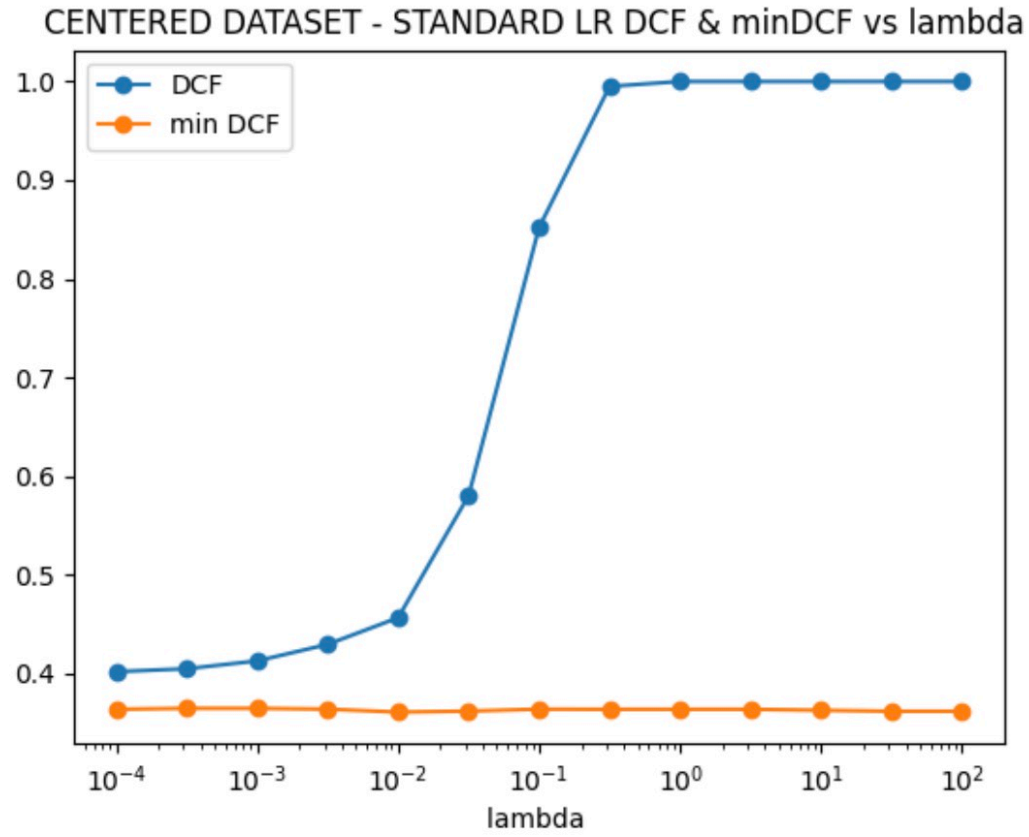
We can do the same analysis also on the Prior-Weighted version of the Logistic Regression Model.



We also want to investigate the behaviour of the Quadratic Logistic Regression model, a model the exploits feature expansion to provide a quadratic separation surface instead of a linear one.



Finally, the project also required to analyze the performance of the Standard Logistic Regression Model over a centered dataset.



We can easily see that no significant change, with respect to the uncentered dataset with the same model, has occured.

These graphs show that the regularization term is useful only when using a small dataset.

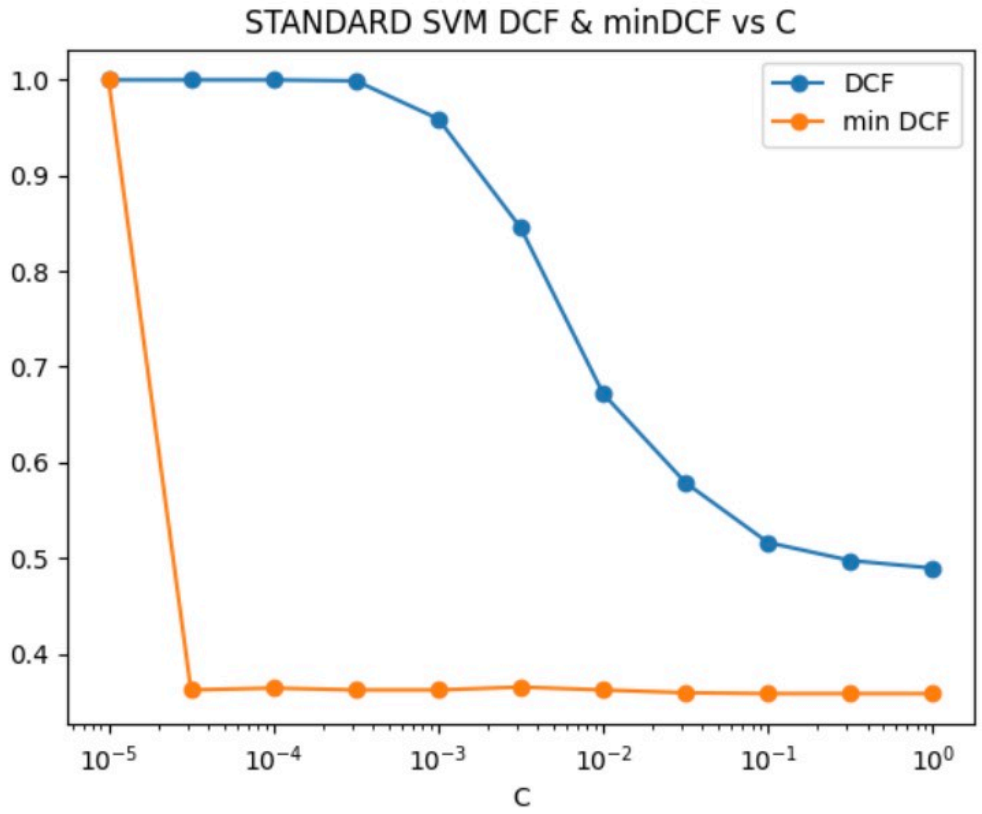
In conclusion, we can easily see that, a part from few cases, every variant of this model provides miscalibrated scores (large difference between actDCF and minDCF). For thi reason, we're gonna evaluate the best model in this section using the minDCF metric which gives us an idea of what the model could achieve if its scores would be calibrated.

The best result is given by the Quadratic Logistic Regression with $\lambda = 10^{-1}$.

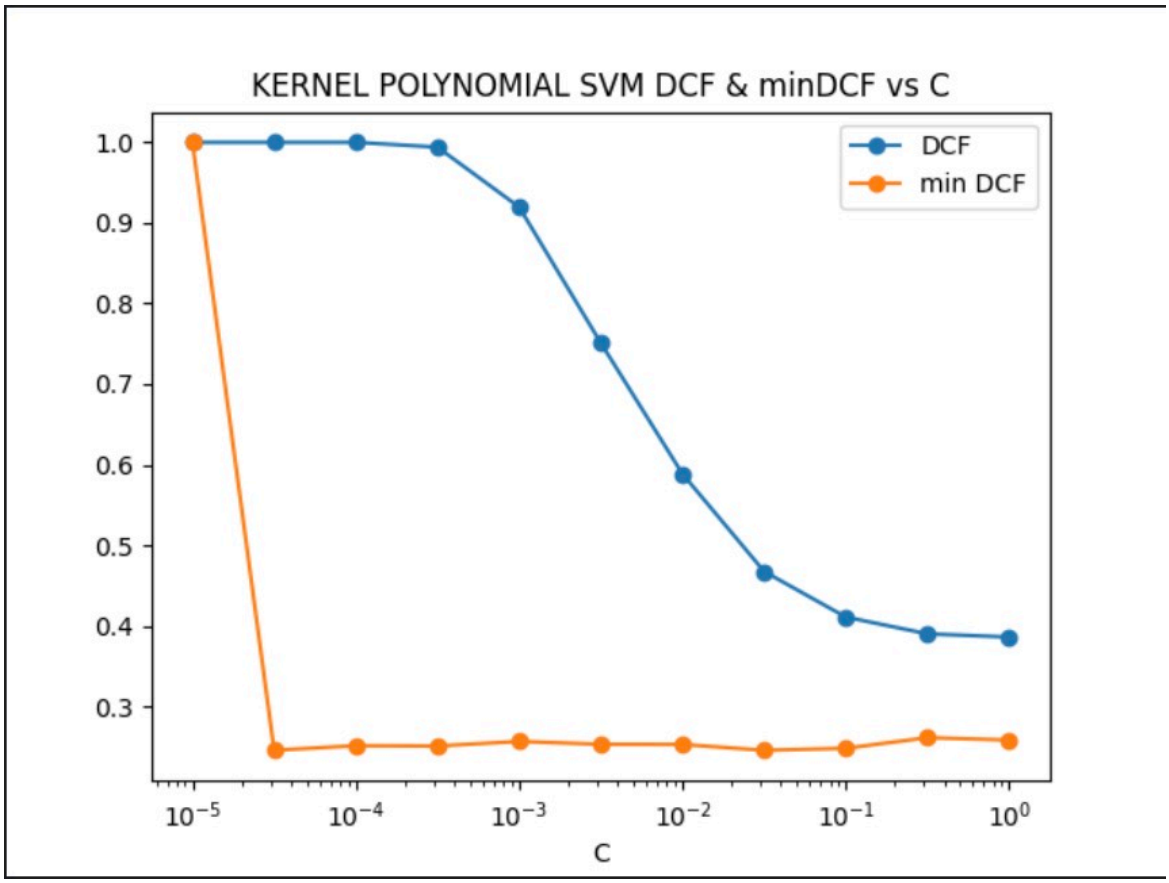
LAB 09: Support Vector Machines

SVM is a classification technique that aims at findi the hyperplane that best separates the dataset classes.
We want to analyze the performance of SVM and all its variants (linear and non-linear i.e. kernel polynomial and kernel RBF).

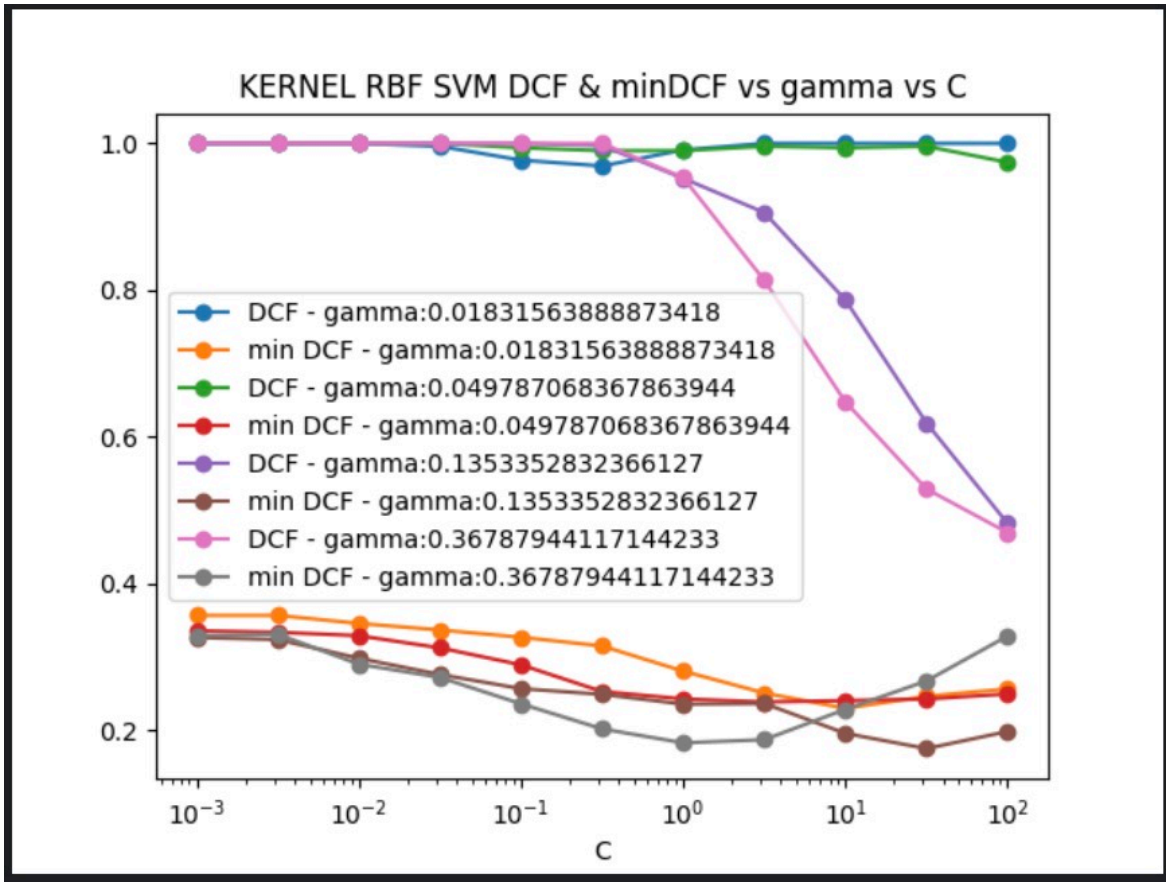
We want to investigate the difference performances of SVM models over the hyperparameters and the target application (0.1, 1, 1).



For the linear SVM model, C does not affect at all the minDCF values. On the other hand, it gives a better calibrated scores while increasing its value (less difference between actDCF and minDCF).



Kernel Polynomial SVM with d=2 has a similar behaviour to the standard SVM model, with scores that are miscalibrated but tend to be more calibrated with C increasing its value. on the other hand, minDCF is not affected by the different values of C.



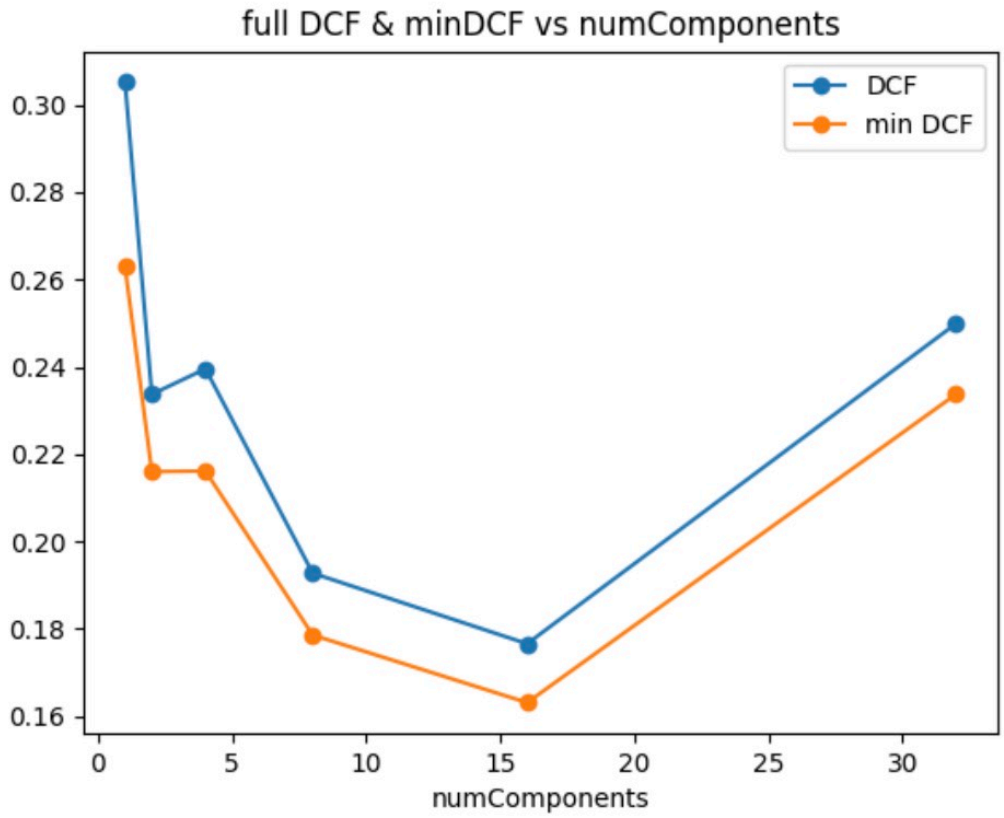
Analyzing the different models of RBF Kernel SVM we can observe a pretty similar behaviour of minDCF related to hyperparameter C. Different values of γ have a significant impact of the minDCF of the different models. Scores are still pretty miscalibrated (gap between actDCF and minDCF).

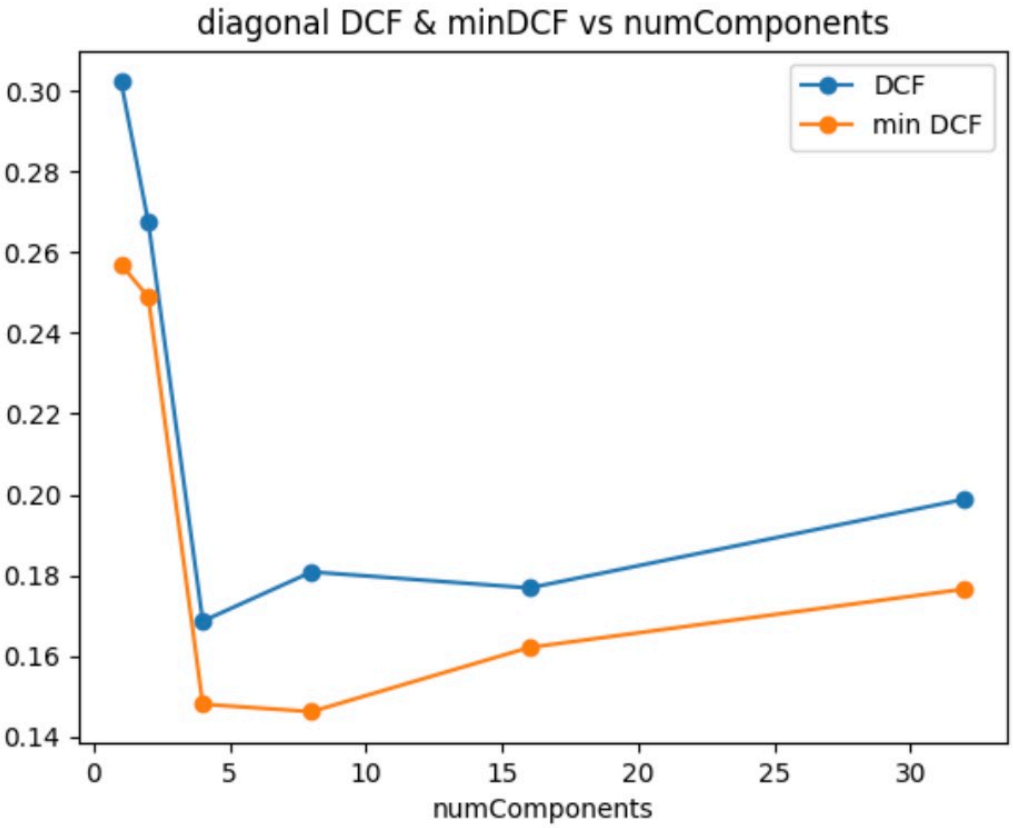
Overall, we can conclude that the best SVM model (decision based on the minDCF values) is represented by the RBF Kernel SVM with C~31.6 and γ~0.13.

LAB 10: Gaussian Mixture Models

Finally, we investigate the performance of Gaussian Mixture Models (GMM).

We want to compare the performances of the standard and the diagonal version of this model with respect to the number of components used.





As we can see the standard version perfroms well with 16 components but cannot reach the performance given by the diagonal version with 4 or 8 components (giving the loweat minDCF values

across the whole project). I decided to choose the 4 components version because it is less computationally expensive having almost the same performance of the 8 components version.

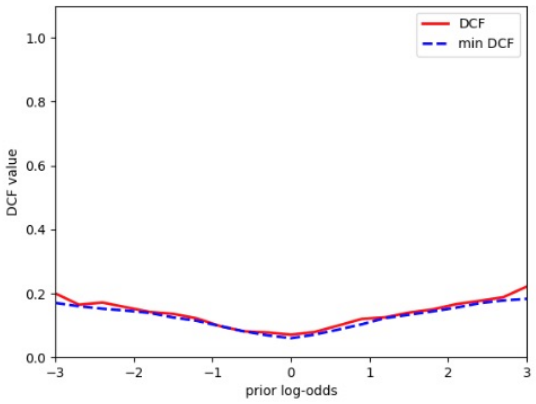
Overall we could've expected these very good results from the GMM since lab_02, where we plotted the dataset over its features. In fact, the point where the MVG model failed to reach good results (features 4 and 5) was pretty obviously well approximatable with a GMM variant!

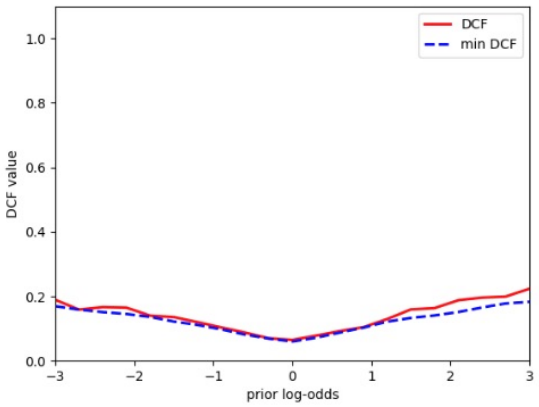
LAB 11: Calibration and Fusion

In this final lab we're going to analyze the effectiveness of score calibration and fusion models. In particular we're going to evaluate the performance of the best models of LR, SVM and GMM and decide which one performs best and how we can still get better overall results starting from these models.

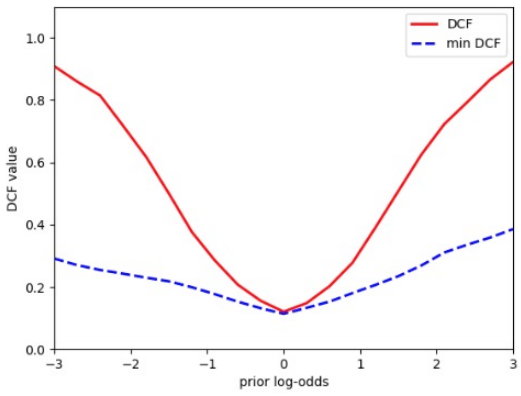
First of all we want to understand whether the best models need a score calibration process. We expect this to be necesssary for SVM and LR models as they tend to output miscalibrated scores.

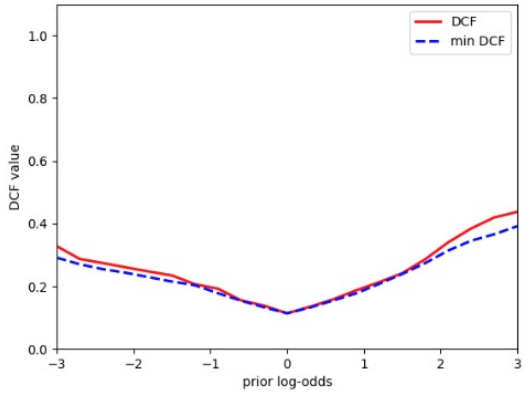
Starting from GMM we can see that the model already outputs calibrated scores over different applications but if we calibrate its scores we get slightly better results on the target application.





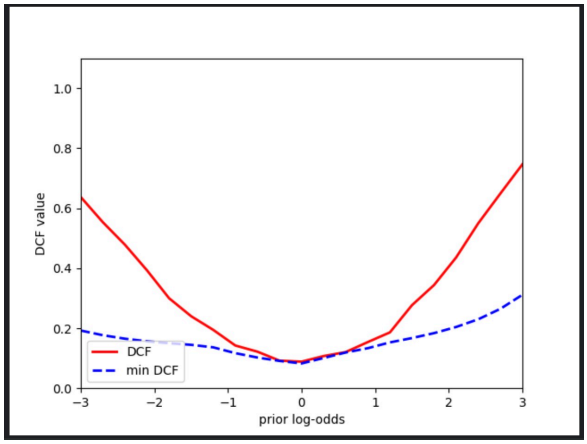
Then we go on with the best LR model:

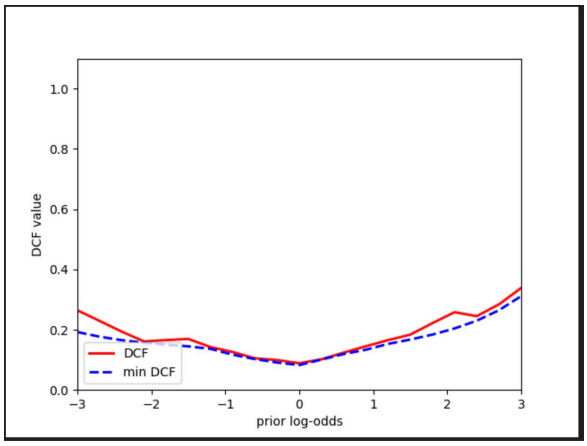




As we can see, here calibration has a significant effect on the actDCF value and the overall goodness of the output scores.

Finally, we take into consideration the best SVM model:



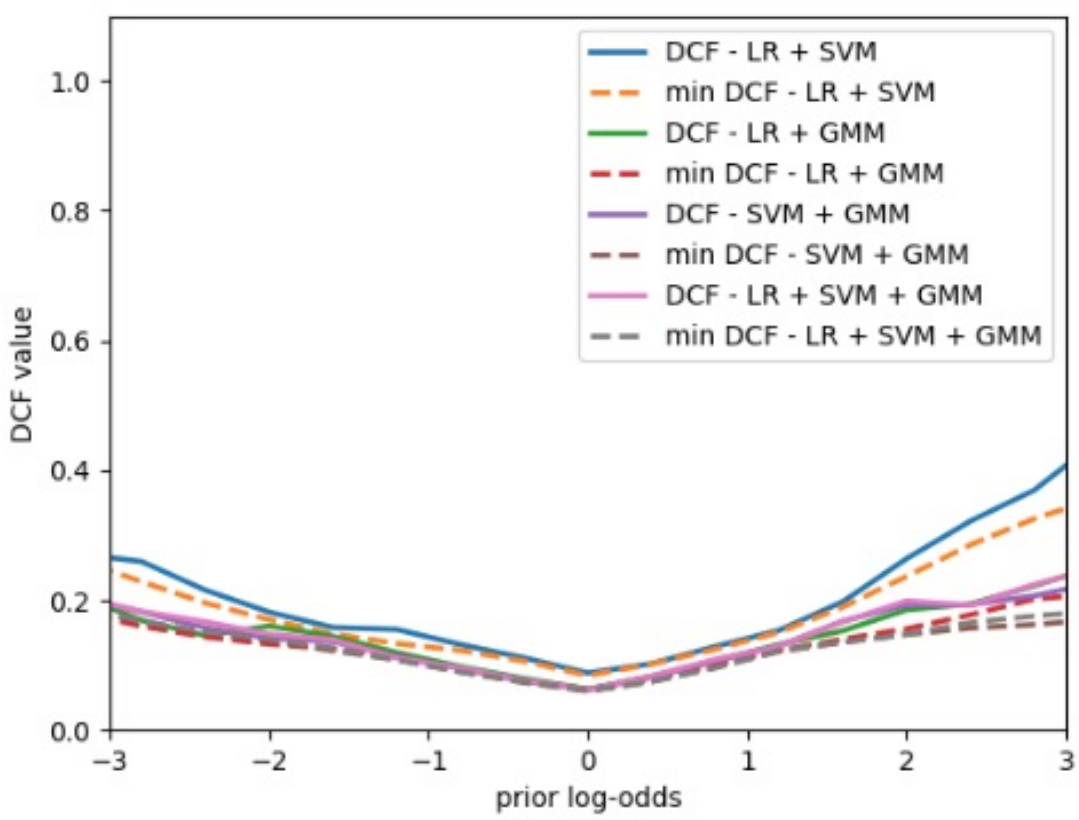


Also in this case, score calibration gives us a significant improvement over actDCF metric.

We take our analysis further into model fusion. In fact, we can train a logistic regression model that takes as input the output scores of our best models and tries to combine them in order to get better results at inference time.

I decided to take into consideration all the possible combinations of the previous models so: [LR + SVM], [LR + GMM], [SVM + GMM] and [LR + SVM + GMM].

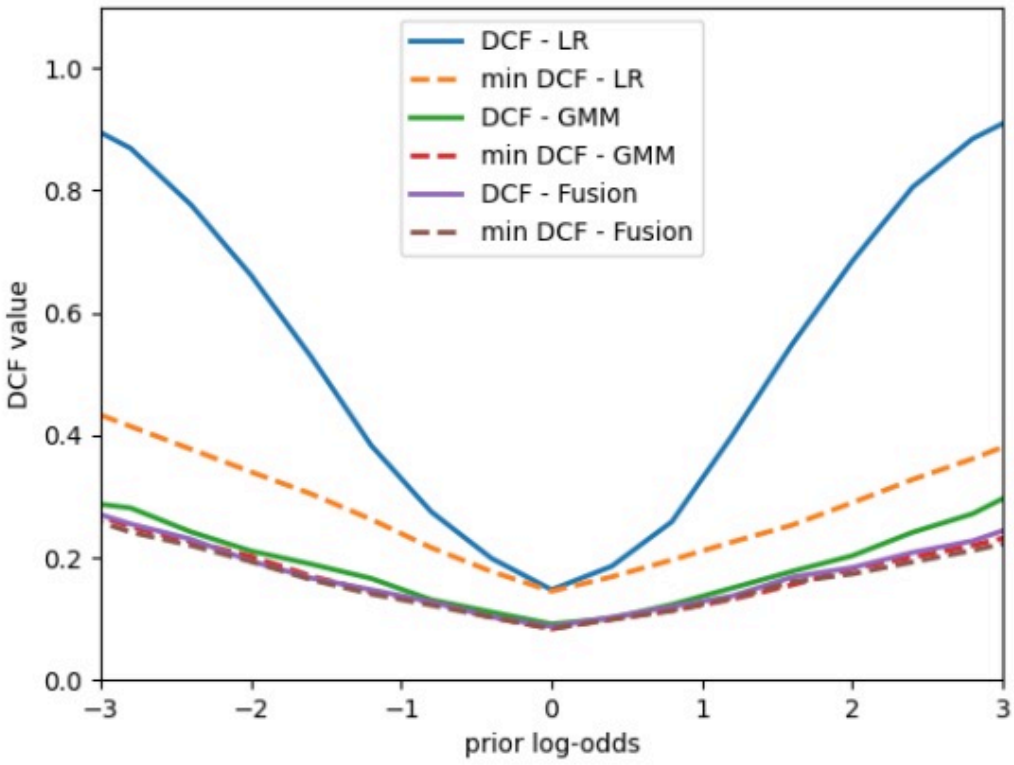
Below you can see the results and how they compare with each other:



Overall we can see that all fusion models perform quite well but the one with better results in my

analysis is the fusion of [LR + GMM].

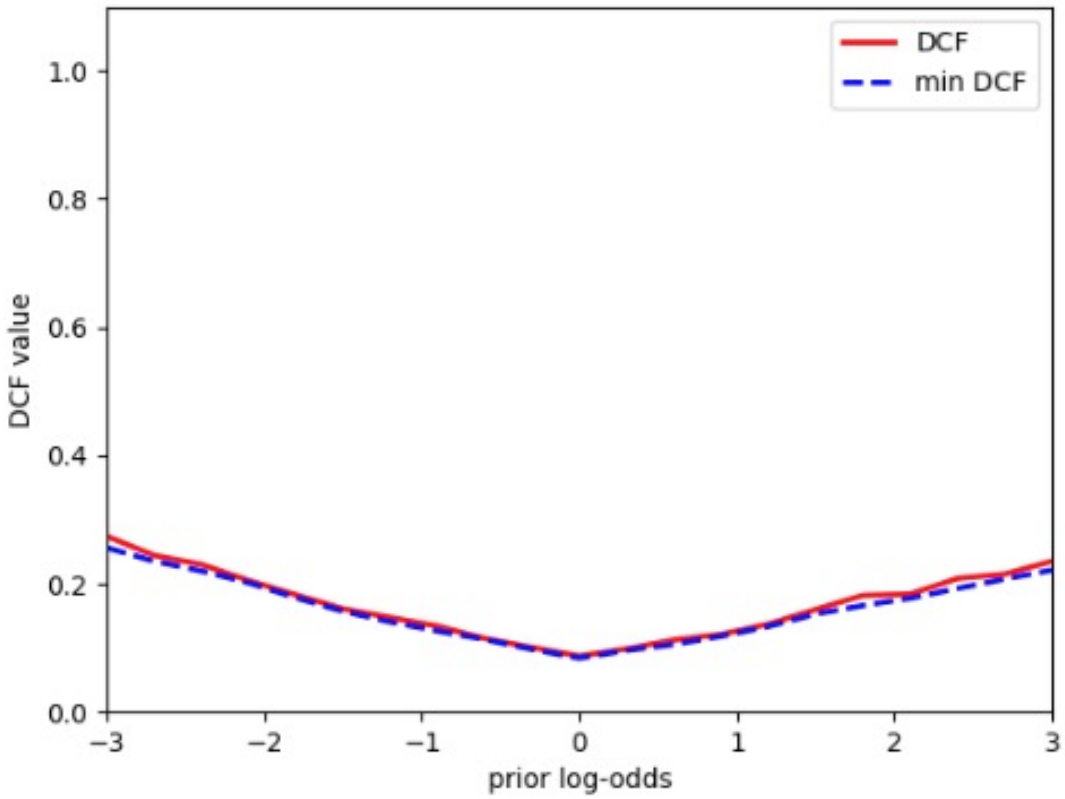
Comparing the fusion model picked with its composing models (LR, GMM) we get the graph below:



In conclusion, I decided to pick the fusion model [LR + GMM] as my final best model.

CONCLUSION

Overall, I decided to pick the fusion model of the best Logistic Regression model and the best Gaussian Mixture Model.
Applying this model to the *evalData.txt* dataset we get the following results:



We can concude that the model has been well calibrated and gets pretty good results also over the evaluation dataset.