Project - Biometric Identification 2022/2023

Luca Costa 290230

Michelangelo Caretto 310178

INTRODUCTION

The goal of the project is the development of a classifier able to verify the correctness of an identity claim based on biometric characteristics of a subject, in details through spoken utterance. The task is to compare an unknown utterance with those belonging to the claimed identity and check if they are similar or not.

The problem can be cast as a binary problem over pairs of utterances. Given a pair of utterances the classifier should decide if the unknown one and the claimed identity one correspond to the same speaker (label 1) or different speaker (label 1).

In this use case the utterances are represented in terms of speaker embeddings. The embeddings are 5-dimensional, continuous-valued vectors, and features have no specific physical interpretation.

INFORMATION ABOUT THE DATASET

Each sample of the dataset is composed of a pair of embeddings for a total dimension of 10.

The training dataset is composed of 3500 samples: 43% belong to label 1 and 57% to class 0.

The test dataset instead is composed of 7500 samples in total: 33% belong to label 1 and 67% to class 0.

Both datasets are slightly imbalanced, with the different speaker class (label 0) having slightly more samples. Pairs belong to either the male or female gender, which is not provided by the datasets.

DATASET FEATURES ANALYSIS

For the analysis of the dataset (which has been normalized first using z-score) is crucial to plot the distribution of the features over histograms. For each feature has been made a distinction between samples belonging to different classes.

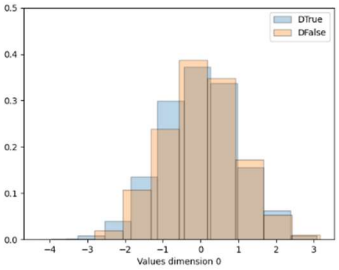


Figure 1

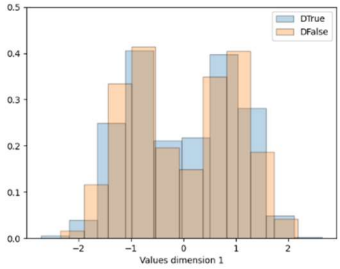


Figure 2

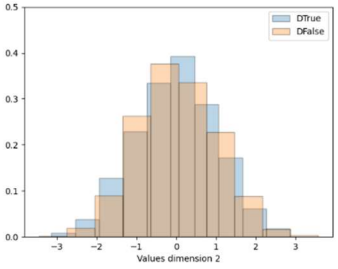


Figure 3

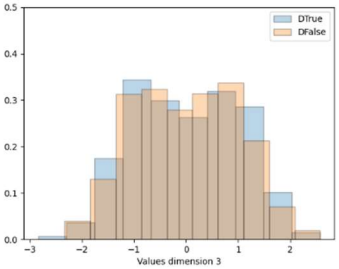


Figure 4

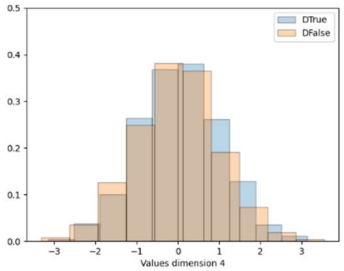


Figure 5

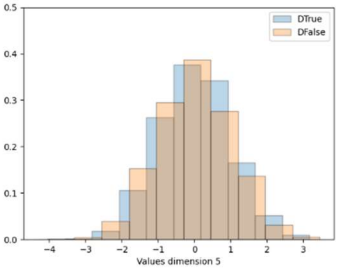


Figure 6

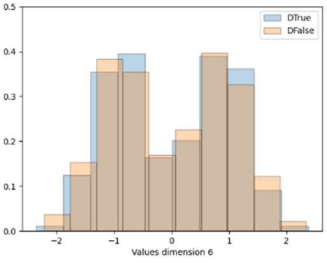


Figure 7

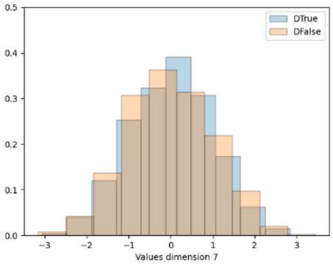


Figure 8

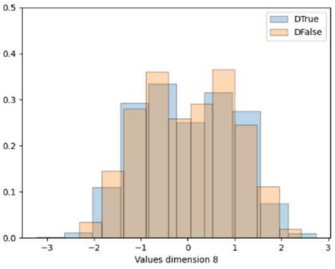


Figure 9

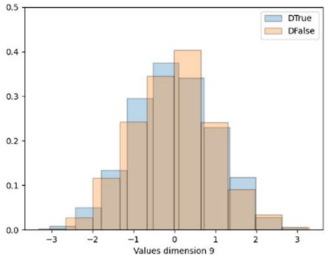


Figure 10

All feature distributions follow a gaussian one with an exception for feature 1, 3, 6, 8. These last reflect a gaussian with 2 components due to the presence of both genders, male and female.

To have a general view over our dataset it has been applied a PCA transformation with 2 dimensions on the left and a LDA transformation on the right. In the PCA graph is evident the presence of two clusters for both the class and in the LDA graph the distribution of the single feature of both classes overlap. These two clusters can be explained by the presence of spoken utterance belonging to the male and female gender. With these results is expected that the data cannot be well separated linearly so a non-linear model should perform better in the classification task.

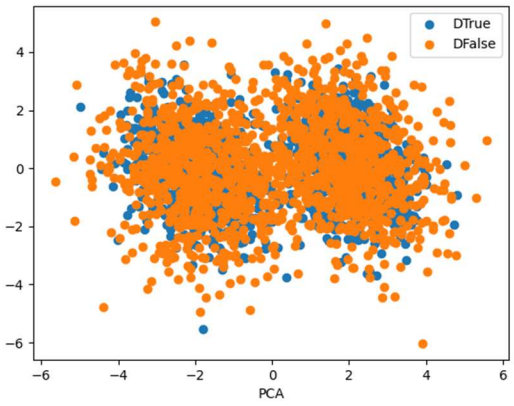


Figure 11 PCA

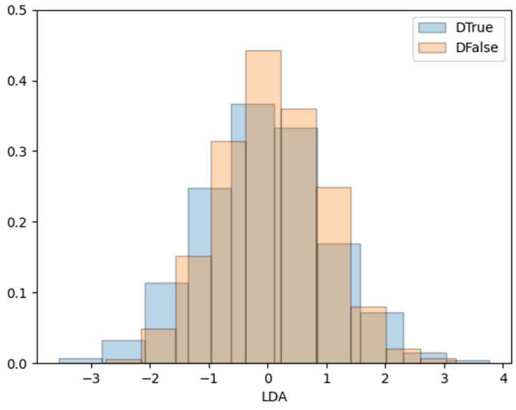


Figure 12 LDA

DATASET FEATURES CORRELATION ANALYSIS

The graphs below represent the correlation between each feature of the training dataset.

Each Heat Map shows the absolute value of the Pearson Correlation Coefficient. On the left it has been performed on the whole dataset and on the right we have the graph of the Explained Variance.

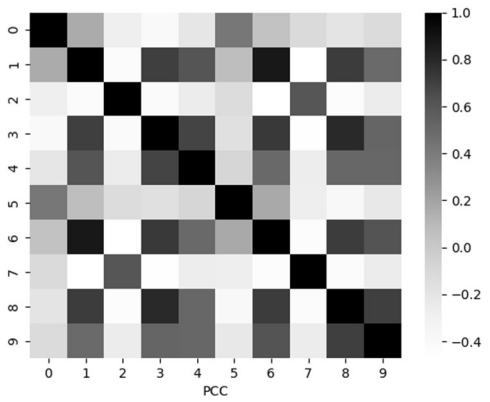


Figure 13 Pearson Correlation Coefficient on all dataset

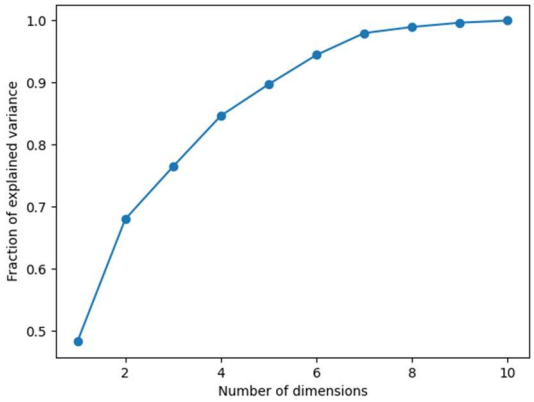
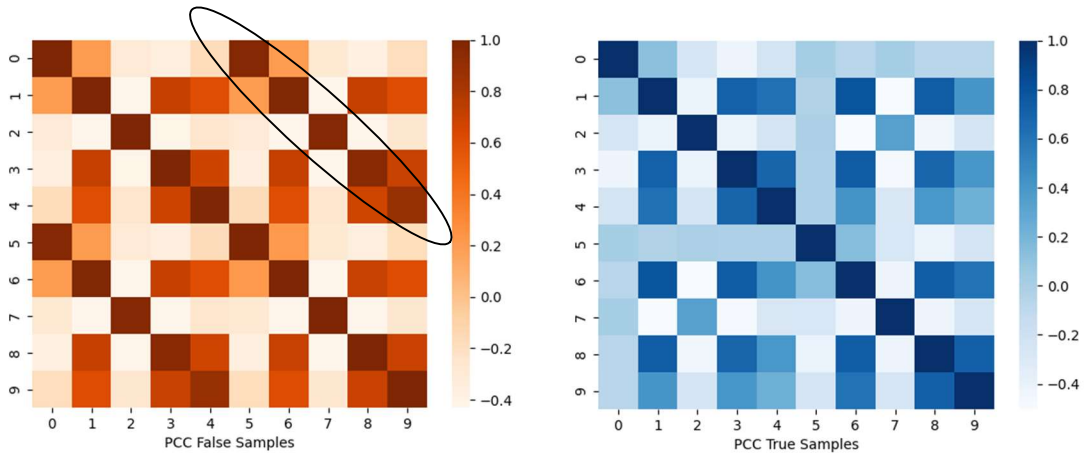
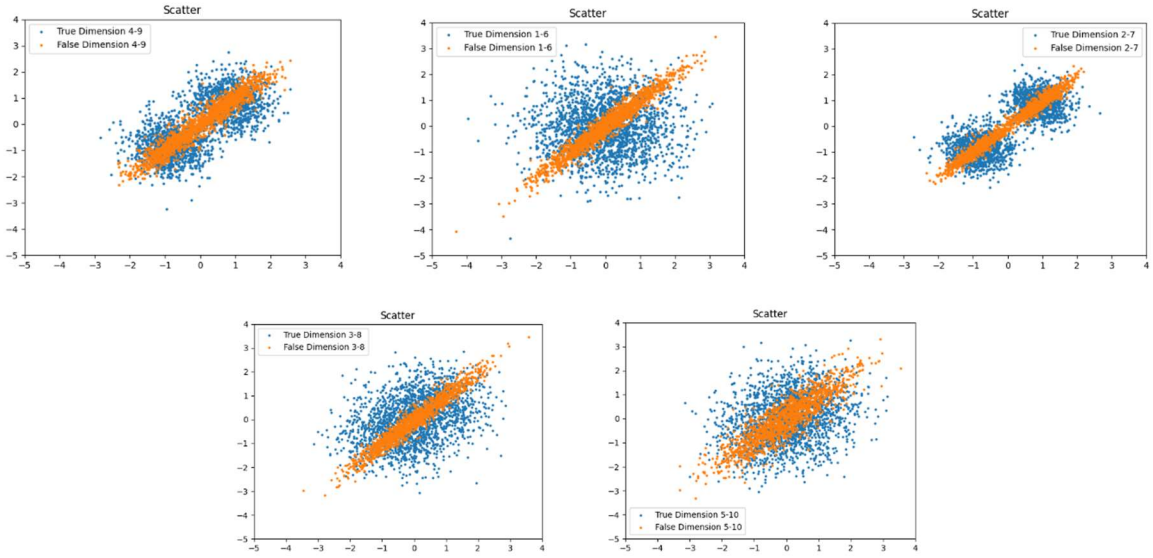


Figure 14 Explained Variance

Looking at figure 13 features 2-7 and 4-9 are the highest correlated. For what concern the others one there is a slight correlation such as also no correlation between some of them. Figure 14 tells us that removing two dimensions will keep very high the variance of the dataset so without losing a lot of information which is crucial during the training phase of our classifier.

Below there are the heatmap of the Pearson Correlation Coefficient of the dataset features belonging respectfully on the left to the label 0 (False) and on the right to the label 1 (True). They are pretty similar with exception for features (1-6), (2-7), (3-8), (4-9), (5-10) which are more correlated in the False samples compared to the True samples. The higher correlation can be seen clearly in the scatter plot immediately below the heatmaps.





With this data we can deduce that an MVG classifier should perform better.

TRAINING PROTOCOL

For the training phase it has been used the K-fold cross validation with 5 folds.

The target application working point given by the project is defined by the triple
($\tilde{\pi}$= 0.1, Cf n = 1, Cf p = 1).
During this phase two others application working points have been used: ($\tilde{\pi}$ = 0.5, Cf n = 1, Cf p = 1) and ($\tilde{\pi}$ = 0.9, Cf n = 1, Cf p = 1).

The performance of the different classifiers is measured in terms of Detection Cost Function (DCF). The classifier with the best performance will be the one with the lowest DCF.

GAUSSIAN CLASSIFIER

In this section the classifier has been trained with several gaussian models: MVG (Multi Variate Gaussian), NB (Naïve Bayes), TC (Tied Covariance) and TNB (Tied Naïve Bayes). The first table shows the performance of the dataset in terms of minimum DCF without pre-processing technique for dimensionality reduction. The following ones show the performances when a pre-process technique has been applied. If applied the type of pre-process technique is described at the top of the table.

	RAW			Z-Score		
Model	π_T=0.1	π_T=0.5	π_T=0.9	π_T=0.1	π_T=0.5	π_T=0.9
MVG	0. 268	**0. 084**	0. 152	0. 276	**0. 084**	0. 152
NB	0.998	0.960	0.996	0.999	0.960	0.996
TC	1.000	0.937	0.990	0.999	0.937	0.990
TNB	0.996	0.963	0.999	0.998	0.963	0.999

As expected from the previous consideration in the Dataset Feature Analysis, since the Tied Covariance model uses a linear separation surface for the classification task it performs badly. The same is for the Tied Naïve Bayes model.

Naïve Bayes applied on the dataset without pre-processing performs bad. This model assumes the independence between the features by diagonalizing the covariance matrix, but we have seen from the heatmaps a strong correlation among some attributes especially in the samples belonging to the False label (label 0).

MVG is the one performing better among all because it uses a quadratic separation rule and the features' distribution fit well into a gaussian.

	RAW PCA 8			Z-Score PCA 8		
Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
MVG	0. 274	**0. 084**	0. 159	0. 326	**0. 104**	0. 193
NB	0.998	0.946	0.992	1.000	0.946	0.992

TC	1.000	0.919	0.981	1.000	0.919	0.981
TNB	0.990	0.953	0.998	0.992	0.953	0.998

	RAW PCA 9			Z-Score PCA 9		
Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
MVG	0. 275	**0. 084**	0. 153	0. 332	**0. 103**	0. 183
NB	0.998	0.946	0.992	1.000	0.946	0.992
TC	1.000	0.919	0.981	1.000	0.919	0.981
TNB	0.990	0.953	0.998	0.992	0.953	0.998

We have applied PCA as pre-processing method for dimensionality reduciton on all the gaussian classifiers. We performed the classification task reducing by a maximum of 2 the number of the database dimension and it ends up that working with 8 directions has the best result.

The MVG classifier keeps its good performances.

Naïve Bayes has a significant improvement. One of the effects of PCA is to decorrelate features, so the model assumption (independence among the features) holds well.

Tied Covariance and Naïve Tied Covariance continue to have bad performances for the same reason explained above.

In all the results obtained the Z-Score normalization gives equal and slightly worst results.

DISCRIMINATIVE CLASSIFIER

In this part the dataset has been trained with discriminative classifiers. Two types of models have been implemented: the Logistic Regression model (LR) and the Support Vector Machine one (SVM). Along with them we have used also the Quadratic Logistic Regression (QLR) and the kernel Support Vector Machine applying the Polynomial and Radial Basis kernel function. The last two use a quadratic separation surface as classification rule. For all the models the prior-weighted version has been treated.

For what concern LR its goal is to find the best hyperplane that maximizes the posterior probability. This problem can be casted to the minimization of the objective function which has as input an hyperparameter called Regularization Term (λ).. The right value of λ has been found using cross-validation techniques using a range which goes from 10^{-5} to 10^{5} with a logarithmic step .

During the training phase the log-likelihood ratio has been performed using the class prior probabilities in the list: [(π_T = 0.1, π_F = 0.9), (π_T = 0.5, π_F = 0.5), (π_T = 0.9, π_F = 0.1)]. For each of them the Detection Cost Function has been performed using all the application working points listed in the "TRAINING PROTOCOL" section.

In the figure below we have plotted the value of DCFs as the Regularization term changes.

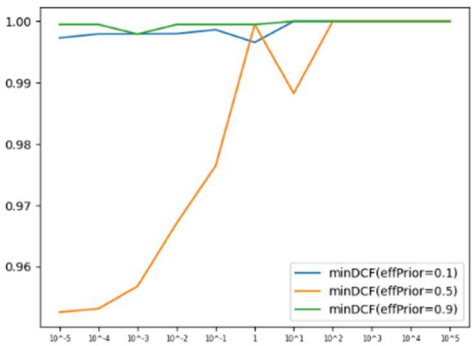


Figure 15 LR Zscore π_T=0.1

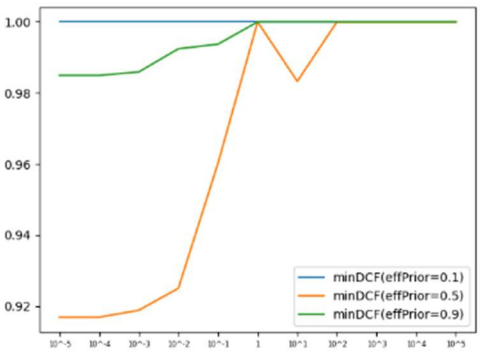


Figure 16 LR Zscore π_T=0.5

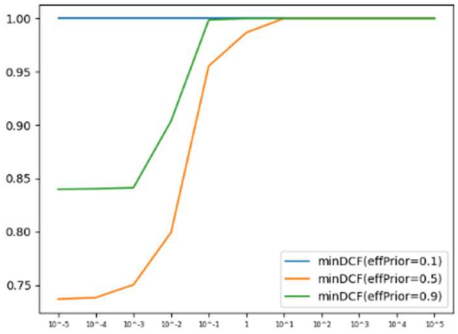


Figure 17 LR Zscore π_T=0.9

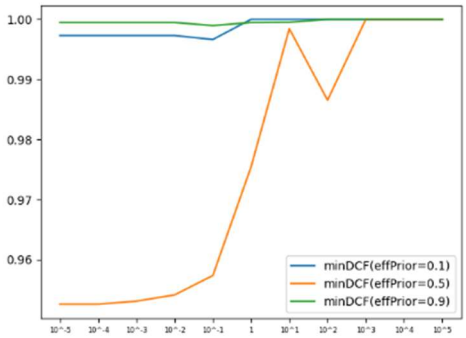


Figure 18 LR Raw π_T=0.1

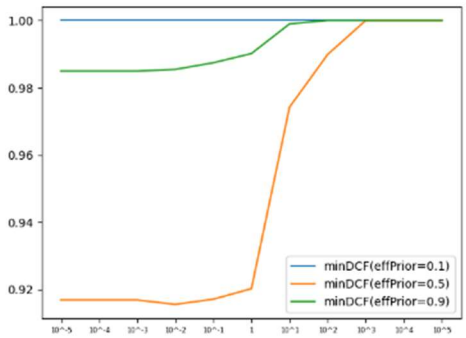


Figure 19 LR Raw π_T=0.5

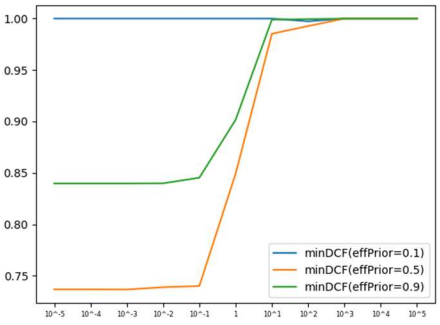


Figure 20 LR Raw π_T=0

The Raw dataset shows slightly better performance compared to the normalized one. In most the case the best value of DCF is given choosing the value 10^{-5} for the hyper parameter λ.

The table below shows the performance of the LR classifiers given the value chosen for the Regularization Term. As mentioned in the section above since LR classifiers use a linear separation surface to classify each sample so its performance is poor.

	RAW			Z-Score		
	$\lambda=10^{-5}$			**$\lambda=10^{-5}$**		
Model	$\tilde{\pi}$=0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
LR (π_T =0.1)	0.996	0.952	0.998	0.997	0.952	0.997
LR (π_T =0.5)	1.0	0.915	0.984	1.0	0.916	0.984
LR (π_T =0.9)	1.0	**0.736**	0.839	1.0	**0.736**	0.839

Now it's analyzed the quadratic version of the LR classifier, the QLR. For what concern the values of the hyperparameter λ have been made the same choice did in the training phase of the LR. Below the plots of the DCFs :

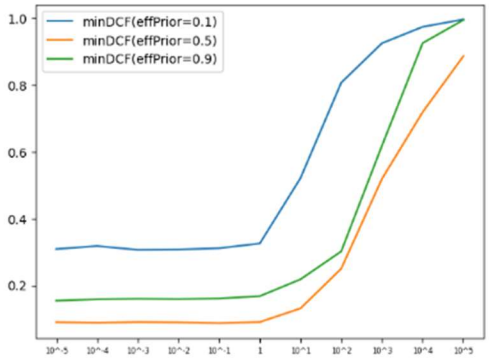


Figure 21 QLR Raw π_T=0.1

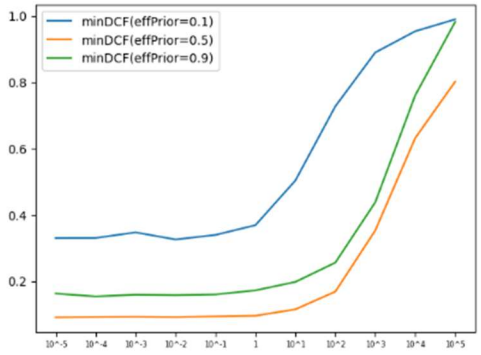


Figure 22 QLR Raw π_T=0.5

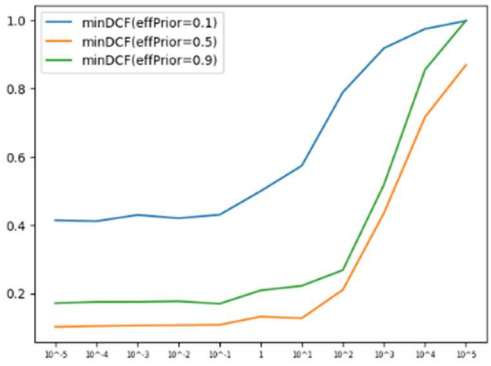


Figure 23 QLR Raw π_T=0.9

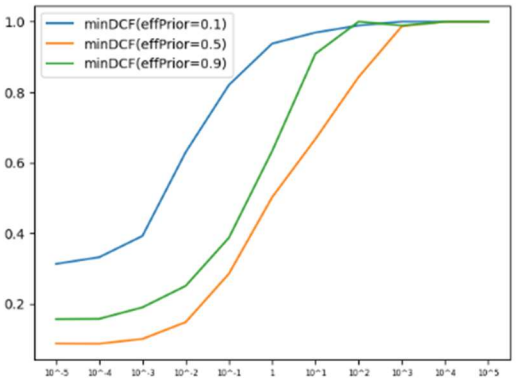


Figure 24 QLR Zscore π_T=0.1

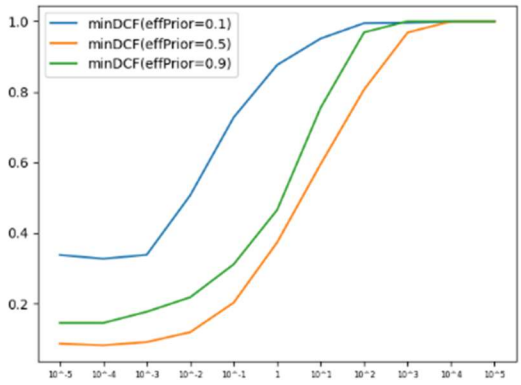


Figure 25 QLR Zscore π_T=0.5

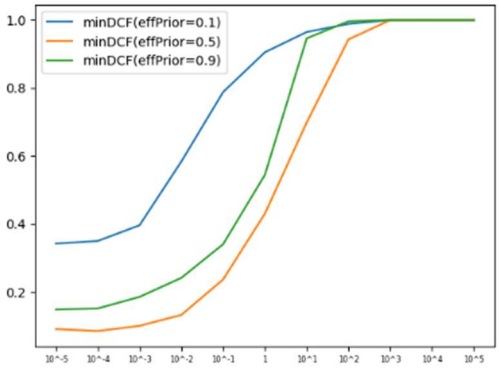


Figure 26 QLR Zscore π_T=0.9

As we already mentioned, since this model uses a quadratic separation surface it exhibits greater performance with respect to LR. The Raw Dataset gives out slightly better results compared to the normalized dataset. The best value of the hyper parameter λ is 10^{-5} in most of the cases. The chosen Regularization term's value minimizes overfitting while still delivering comparable performance to other values. By selecting this value, we can mitigate the risk of overfitting the table shows gives in number the results of the performance.

	RAW			Z-Score		
	λ = 10^-5			**λ = 10^-5**		
Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
QLR (π_T =0.1)	0.309	**0.090**	0.154	0.313	0.087	0.156
QLR (π_T =0.5)	0.330	0.091	0.163	0.337	**0.086**	0.145
QLR (π_T =0.9)	0.414	0.102	0.171	0.342	0.090	0.147

For further analysis we will consider π_T = 0.5 which give out the best result among all the configurations.

NON-PROBABILISTIC CLASSIFIERS

The classifiers treated in this section are called in this way because the scores generated do not have a probabilistic interpretation.

The first model analyzed is the Support Vector Machine (SVM). Like the Logistic Regression model the SVM aims to find a hyperplane that separates the classes. The difference is on the choice of the best hyperplane: with SVM the desired hyperplane is the one which maximizes the margin (e.g., maximizing the distance with respect to all points). As for the discriminative model, it has been analyzed the prior-weighted version. The hyperparameter to be tuned is denoted as C.

The Primal Objective function is given by :

$$L_P(w, b) = \frac{1}{2} \left|\left|w\right|\right|^2 + \ \mathrm{C} \ \sum_{i=1}^{n} [1 - z_i \ (\ w^T x_i + \mathrm{b})]$$

For class balancing, we can set $C_i = C_T$ for samples belonging to label 1 and $C_i = C_F$ for samples of label 0.

We can select $C_T \ = \ C \ \frac{\pi_T}{\pi_T^{emp}}$ and $C_F \ = \ C \ \frac{\pi_F}{\pi_F^{emp}}$

π_T^{emp} and π_F^{emp} are the empirical priors for the two classes computed over the training set.

All the following analysis are performed with class balancing.

The performance has been analyzed with values of C ranging from 10^{-5} to 10^{5} with a logarithmic step .

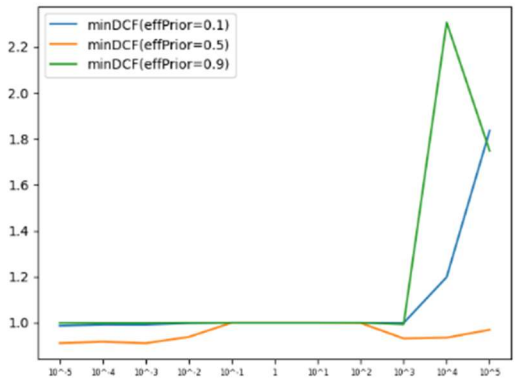


Figure 27 SVM Raw π_T=0.1

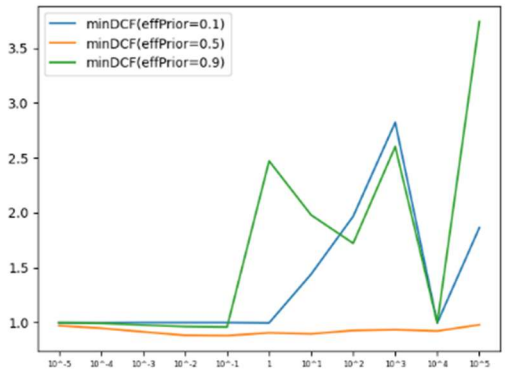


Figure 28 SVM Raw π_T=0.5

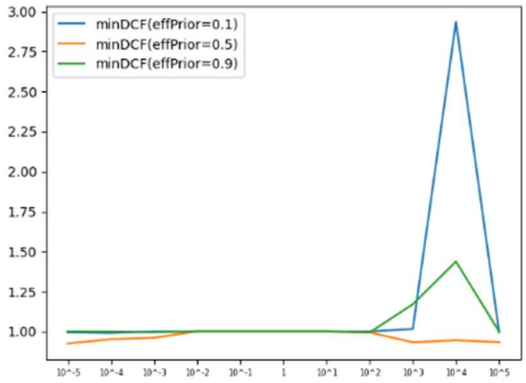


Figure 29 SVM Raw π_T=0.9

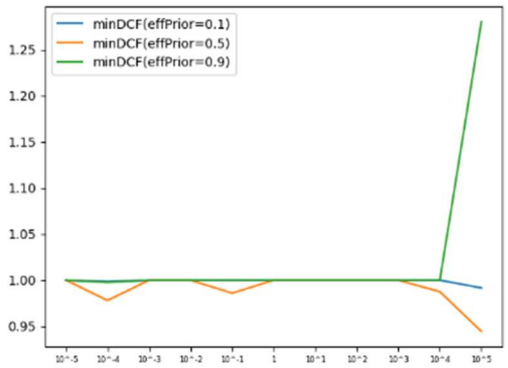


Figure 30 SVM Zscore π_T=0.1

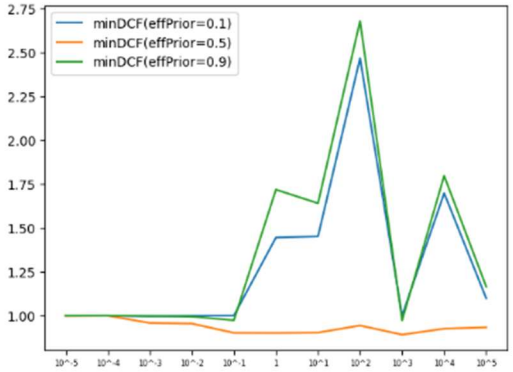


Figure 31 SVM Zscore π_T=0.5

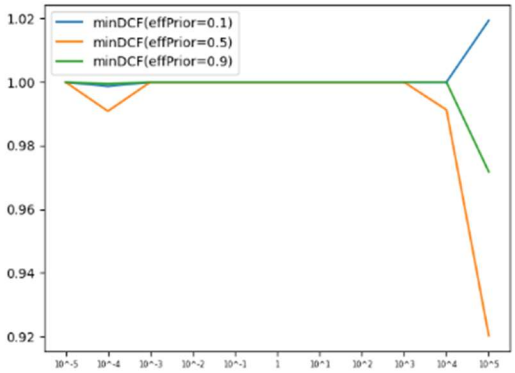


Figure 32 SVM Zscore π_T=0.9

Observing the plots its clear the bad performance of this linear model which is clearly not suitable to fit our data. The normalization of the dataset doesn't give improvements.

In the table below are reported the values of the DCFs obtained with $C = 10^{-3}$

	RAW			Z-Score		
	C = 10^-3			C = 10^-3		
Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
SVM (π_T =0.1)	0.991	**0.911**	0.999	1.0	1.0	1.0
SVM (π_T =0.5)	1.0	0.915	0.977	0.996	**0.957**	0.998
SVM (π_T =0.9)	1.0	0.960	0.994	1.0	1.0	1.0

Given the data in the table we can confirm the bad performance of the model.

We now move on to the quadratic version of the SVM model. It is implemented exploiting dot-products inside the Dual Objective function which can be efficiently computed throw kernel functions. We explored two kernel functions: Polynomial and Radial Basis kernel functions.

Polynomial : $k(x_1, x_1) = (x_1^T x_2 + 1)^d$ with d=2

Radial Basis: $e^{-\gamma||x_1 - x_2||^2}$

For the C parameter we have employed the same range of values applied in the linear version. For what concern the quadratic version implemented with Radial Basis kernel function we uses the values $[10^{-1}, 10^{-2}, 10^{-3}]$ for the γ parameter.

Below the plot of the performance given by the Polynomial SVM. As always, we evaluated the performance for the dataset as it is and the normalized one.

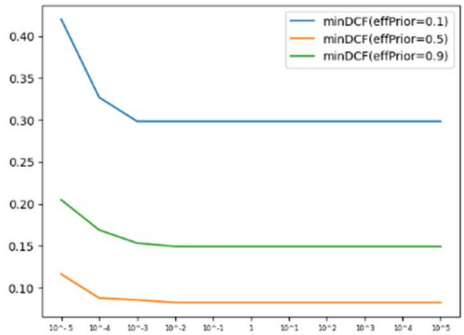


Figure 33 PolySVM Raw π_T=0.1

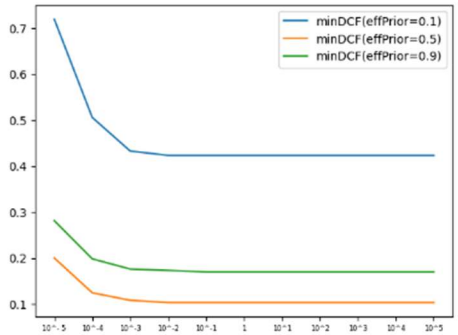


Figure 34 PolySVM Raw π_T=0.5

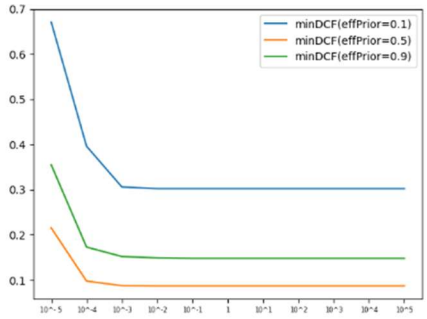


Figure 35 PolySVM Raw π_T=0.9

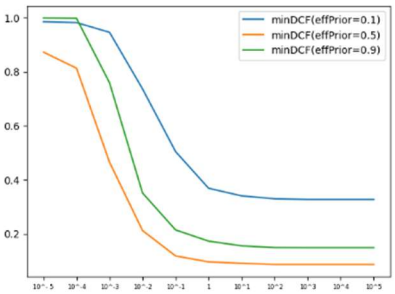


Figure 36 PolySVM Zscore π_T=0.1

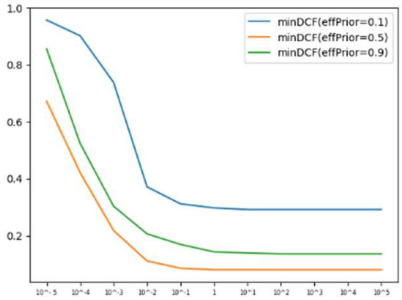


Figure 37 PolySVM Zscore π_T=0.5

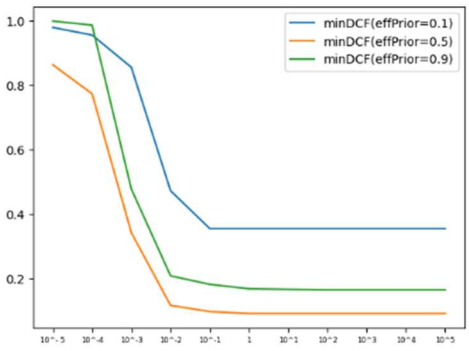


Figure 38 PolySVM Zscore π_T=0.9

The Polynomial SVM gives out better results compared to its linear version. By looking at the plots the best choice of C for both Raw version is given by 10^{-2} while for Zscore is 10^{2}.

	RAW			Z-Score		
	C = 10^-2			**C = 10^2**		
Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
QSVM (π_T =0.1)	0.302	0.087	0.148	0.327	0.086	0.148
QSVM (π_T =0.5)	0.298	**0.082**	0.149	0.291	**0.080**	0.136
QSVM (π_T =0.9)	0.423	0.103	0.170	0.354	0.090	0.164

The results are really good, as for the QLR model. This reinforces our assumption that a non-linear decision rule is appropriate for this dataset. Additionally, there isn't a notable difference in performance between the RAW and Z-Score features.

Based on the results obtained, it is evident that this model, particularly the QSVM with π_T = 0.5, is among the top-performing models for classifying our sample. Therefore, we will consider this model for further analysis.

Here we analyze the quadratic version of SVM with Radial Basis kernel function. We will take the decision on hyperparameter's tuning looking the two plots generated using the Raw and Normalized dataset. For both it has been used a class prior π_T=0.5

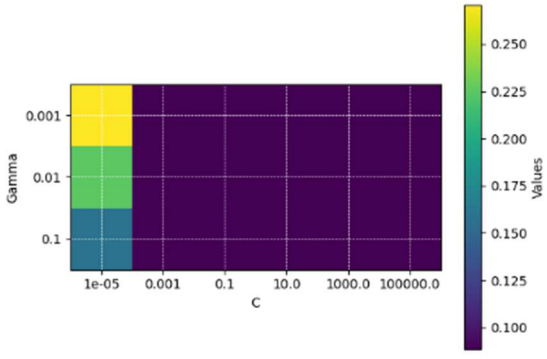


Figure 40 RBSVM Raw π_T=0.5

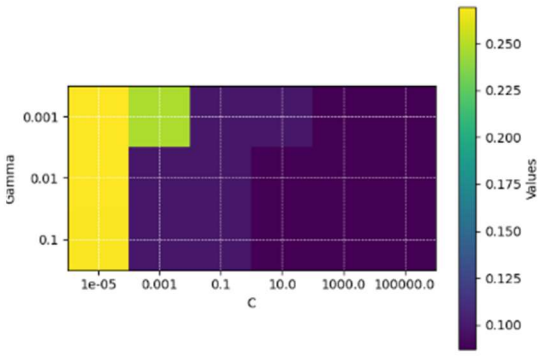


Figure 39 RBSVM Zscore π_T=0.5

For the Raw features it has been chosen the values $C = 10^{-3}\ and\ \gamma = 10^{-2}$

For the Zscore features it has been chosen the values $C = 10^{1}\ and\ \gamma = 10^{-2}$

	RAW			Z-Score		
	C = 10^-3	**γ=10^-2**		**C = 10^1**	**γ=10^-2**	
Model	π̃ =0.1	π̃ =0.5	π̃ =0.9	π̃ =0.1	π̃ =0.5	π̃ =0.9
RBSVM (πT =0.1)	0.385	0.105	0.204	0.358	0.120	0.217
RBSVM (πT =0.5)	0.415	**0.088**	0.204	0.468	**0.086**	0.217
RBSVM (πT =0.9)	0.428	0.105	0.162	0.468	0.116	0.176

The performance are comparable to the same model with Polynomial kernel function. It performs slightly worse.

GMM

The Gaussian Mixture Model (GMM) is a type of classifier commonly used for density estimation problems. The underlying assumption when using a GMM is that the data can be approximated as a mixture of one or more Gaussian distributions. In other words, the GMM assumes that the data is generated by multiple Gaussian components, each contributing to the overall distribution of the data. The expectation is that the best results will be achieved with Gaussian Mixture Models having 2 components. This choice is influenced by the dataset analysis, which indicates that the training data's distribution resembles a Gaussian distribution with approximately 2 components or clusters.

In this case, the hyperparameter to tune is the number of components for each Gaussian distribution within the mixture model.

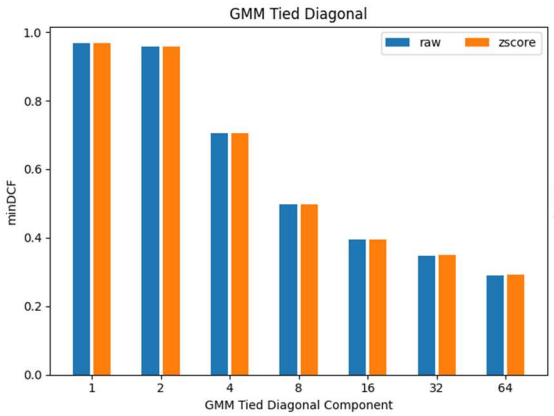


Figure 41

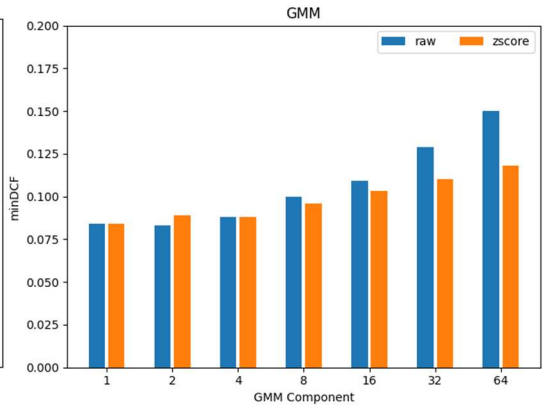


Figure 42

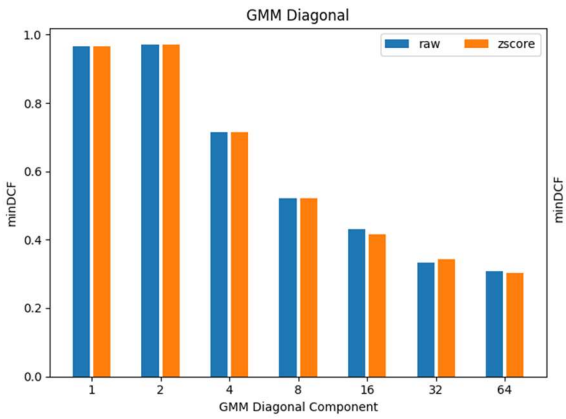


Figure 43

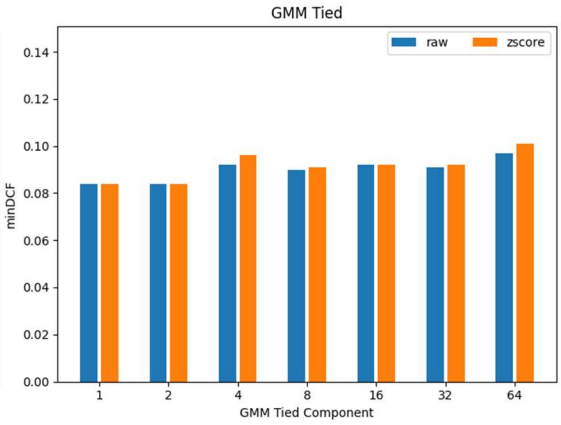


Figure 44

As we expected by analyze MVG results, the two diagonal models yield unsatisfactory results due to the moderate correlation observed in the heatmaps, while Tied Model in this case works well because the Tied assumption is between component of the same classes.

Here the list of the best GMM models:

	RAW			Z-Score		
Model	**$\tilde{\pi}$ =0.1**	**$\tilde{\pi}$ =0.5**	**$\tilde{\pi}$ =0.9**	**$\tilde{\pi}$ =0.1**	**$\tilde{\pi}$ =0.5**	**$\tilde{\pi}$ =0.9**
GMM (2 components)	0.31	0.084	0.167	0.309	0.089	0.168
GMM Tied (2 components)	0.266	0.084	0.15	0.266	0.084	0.15
GMM Diagonal (64 components)	0.896	0.308	0.488	0.901	0.302	0.516
GMM Tied Diagonal(64 components)	0.865	0.29	0.504	0.858	0.292	0.498

Now we are applying PCA as pre-processing that should provide better results in case of Diagonal assumption in the GMM, because of the decorrelation between features applied by it. In this case Z-score pre-processing is not applied since the previous results are similar.

Here the list of the best GMM+Pca models:

	RAW PCA 8		
Model	**$\tilde{\pi}$ =0.1**	**$\tilde{\pi}$ =0.5**	**$\tilde{\pi}$ =0.9**
GMM (1 Component)	0.384	0.102	0.2
GMM Tied (4 Component)	0.321	0.111	0.213
GMM Diagonal (1 Component	0.319	0.102	0.192
GMM Tied Diagonal(1 Component)	0.384	0.102	0.2

RAW PCA 9

Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
GMM (2 components)	0.349	0.094	0.187
GMM Tied (2 components)	0.349	0.094	0.187
GMM Diagonal (2 components)	0.349	0.094	0.187
GMM Tied Diagonal (2 components)	0.349	0.094	0.187

As expected the two models Gmm Diagonal and Gmm Tied Diagonal improved a lot, but they aren't interesting as the other models.

SCORE CALIBRATION

The best three performing models are:

Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
MVG Zscore	0. 268	**0. 084**	0. 152
QSVM Zscore (π_T =0.5,C=10**2)	0.291	**0.080**	0.136
QLR Zscore (π_T =0.5, λ =10**-5)	0.337	**0.086**	0.145

Overall we can see that QSVM (SVM with Polynomial kernel function) is the one performing better among the top 3 trained classifiers treated in this report.

Up until now, our focus has solely been on assessing different models using the minimum detection cost as our metric. However, a significant challenge lies in the fact that the cost we incur depends on a specific threshold. To gain a more comprehensive understanding and ascertain if the chosen threshold is theoretically optimal, we will introduce a metric known as the actual Detection Cost Function (DCF).

Our chosen approach for this evaluation involves utilizing Logistic Regression, as it serves as a posterior log-likelihood ratio. We will derive the calibrated score by straightforwardly subtracting the theoretical threshold.

To estimate the parameters of the calibration function, we will adopt a Singl -Fold Approach.

Among the various models, we'll calibrate the best three models.

Here the bayes error plot:

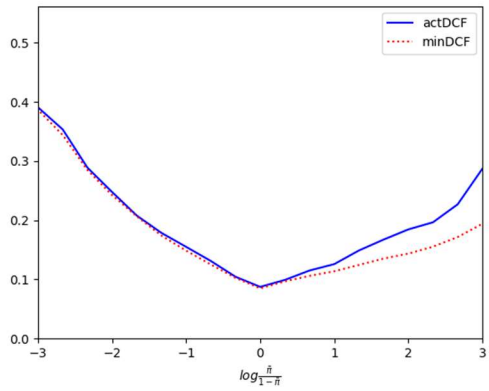


Figure 45 Uncalibrated MVG Zscore

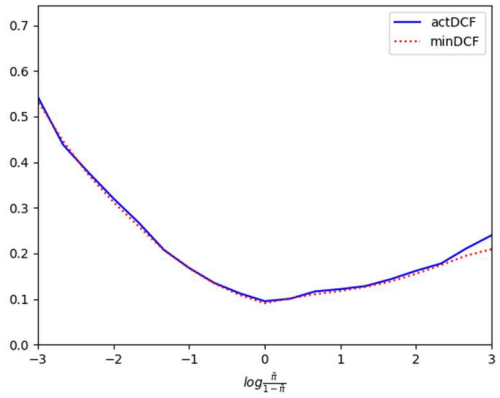


Figure 46 Uncalibrated QLR Zscore (π_T =0.5)

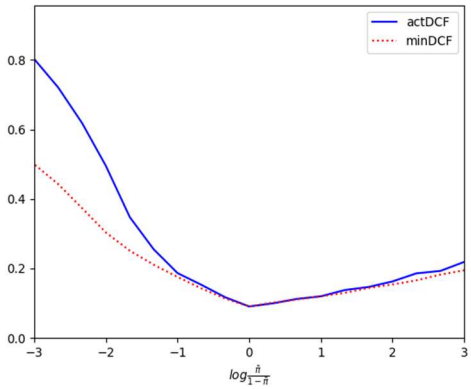


Figure 47 Uncalibrated QSVM Zscore (π_T =0.5)

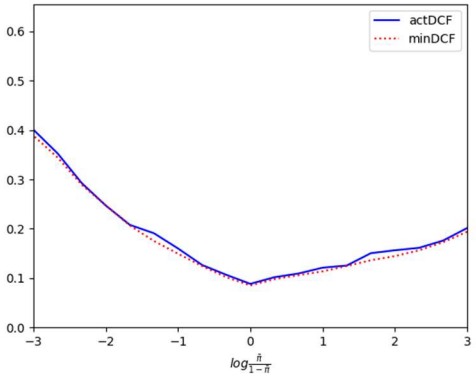


Figure 48 Calibrated MVG Zscore

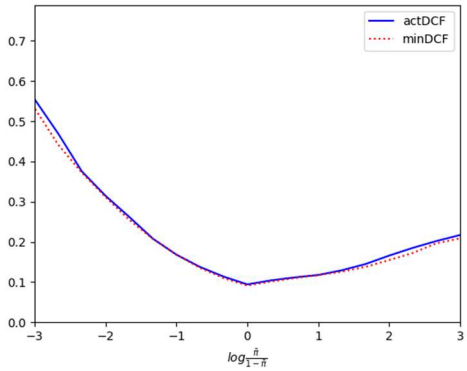


Figure 49 Calibrated QLR Zscore (π_T =0.5)

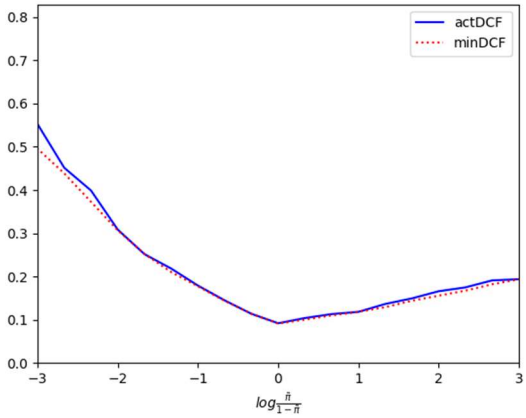


Figure 50 Calibrated QSVM Zscore (π_T =0.5)

The calibration has effect to the MVG and QSVM models, while for the QLR doesn't bring benefit. The calibration has no significant effect on the minDCF. Here is the table to prove it.

Calibrated

	actDCF			minDCF		
Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
MVG Zscore	0.276	0.088	0.088	0.269	0.084	0.084
QSVM Zscore (π_T =0.5)	0.361	0.091	0.091	0.348	0.091	0.091
QLR Zscore (π_T =0.5)	0.347	0.094	0.094	0.349	0.090	0.090

Uncalibrated

	actDCF			minDCF		
Model	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
MVG Zscore	0.277	0.087	0.087	0.271	0.084	0.084
QSVM Zscore (π_T =0.5)	0.562	0.090	0.090	0.344	0.090	0.090
QLR Zscore (π_T =0.5)	0.351	0.095	0.095	0.350	0.091	0.091

As there isn't a significant distinction in performance between the uncalibrated scores and the calibrated ones, we have decided to retain the uncalibrated model as our final classifier for simplicity.

Fusion

Here we explore the possibility of combining different classifiers which can enhance the overall performance. This approach fuses the outputs of our top three selected models: QLR, QSVM and MVG. We will utilize the K-Fold Approach to combine these decisions using prior-weighted logistic regression. We explored all the combinations given by the three classifiers.

Below the Bayes Error plots of the fused models

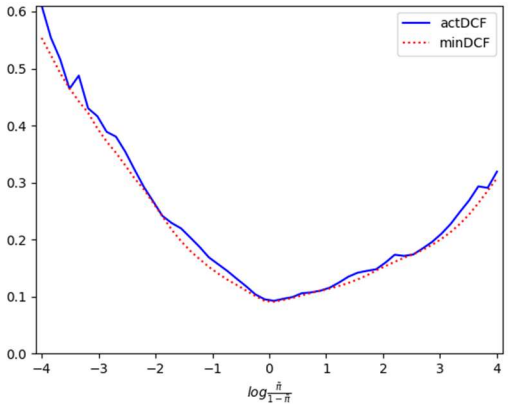


Figure 51 MVG_QLR_QSVM_Zscore

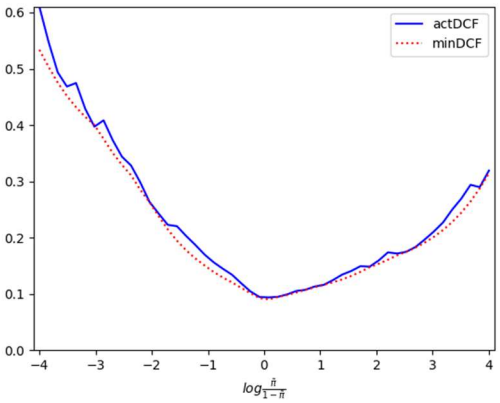


Figure 52 MVG_QLR_Zscore

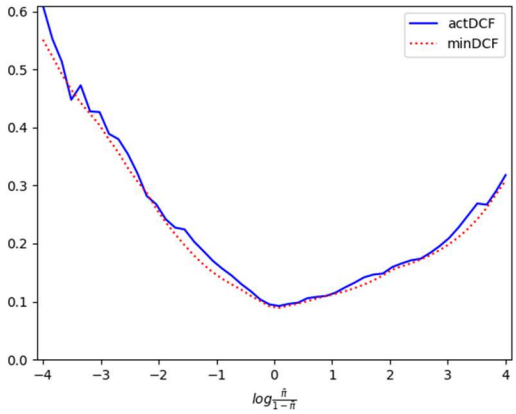


Figure 53 MVG_QSVM_Zscore

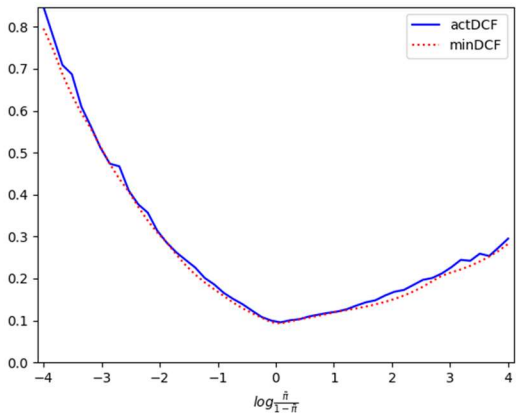


Figure 54 QLR_QSVM_Zscore

As indicated by the plots, all the scores are already calibrated, eliminating the need for further calibration steps for the models.

Below, we will analyze the results of these models in terms of the actDCF and minDCF.

	actDCF			minDCF		
	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
Model						
MVG + QSVM	0.280	**0.090**	0.090	0.286	**0.086**	0.086
QSVM + QLR	0.355	0.092	0.092	0.336	0.090	0.090
QLR + MVG	0.296	**0.090**	0.090	0.284	0.088	0.088
QSVM +QLR+ MVG	0.291	**0.090**	0.090	0.287	0.088	0.088

As we can see by comparing the table with the previous results , fusion has a significant improvement for the application prior $\tilde{\pi}$ =0.9.

The overall best result is obtained combining Q-SVM and MVG, so because of that we'll choose this model as final one, that is the most balance between all application prior(0.1,0.5,0.9).

Evaluation

Now, we shift our attention to evaluating the effectiveness of our selections on a test dataset, thoroughly assessing their performance. In this phase, we narrow our consideration to only those models that demonstrated the most promising results in the previous phase which are the top 3 classifier and their fusion.

Fused Evaluation

We start to consider fused and uncalibrated models (because we have seen that there aren't big improvements in calibration) and try to understand their performance on test data.

Remember that we are using MVG
Full Diagonal, QSVM and QLR trained both with $\pi T=0.5$.

All these models have obtained these results by considering the Z-Score preprocessing step on the data.

	actDCF			minDCF		
	$\tilde{\pi}=0.1$	$\tilde{\pi}=0.5$	$\tilde{\pi}=0.9$	$\tilde{\pi}=0.1$	$\tilde{\pi}=0.5$	$\tilde{\pi}=0.9$
Model						
MVG + QSVM	0.363	0.103	0.103	0.289	0.089	0.089
QSVM + QLR	0.361	0.103	0.103	0.316	0.090	0.090
QLR + MVG	0.353	0.104	0.104	0.291	0.087	0.087
QSVM +QLR+ MVG	0.353	0.104	0.104	0.291	0.087	0.087

The outcomes are positive: it appears that all the four models have similar results compared to the results of the previous table. This data shows that the tuning of the hyperparameter during the validation phase was optimal because we have obtained almost the same performance.

To gain further insights, we will now present the Bayes error plot, aiming to assess whether calibration is necessary for these models.

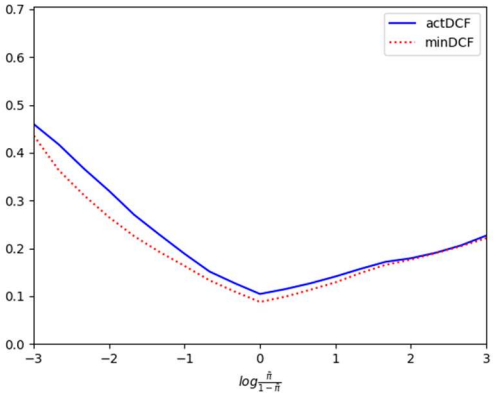


Figure 55 MVG_QLR_QSVM_Zscore

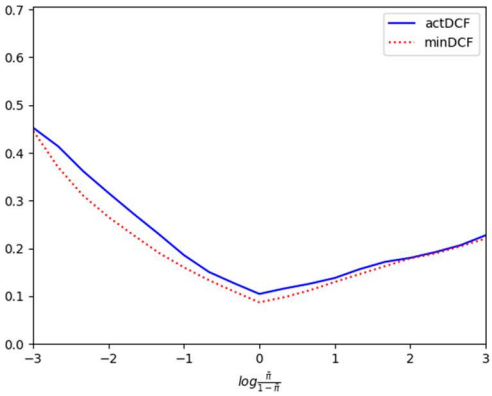


Figure 56 MVG_QLR_Zscore

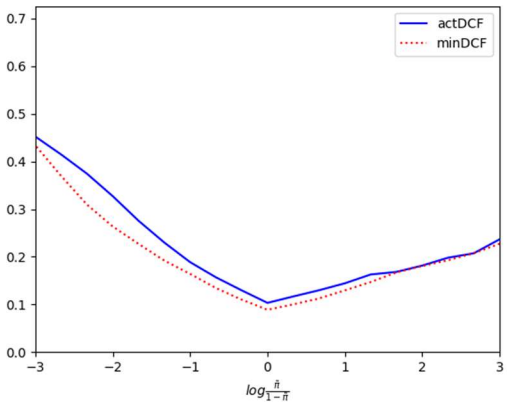


Figure 57 MVG_QSVM_Zscore

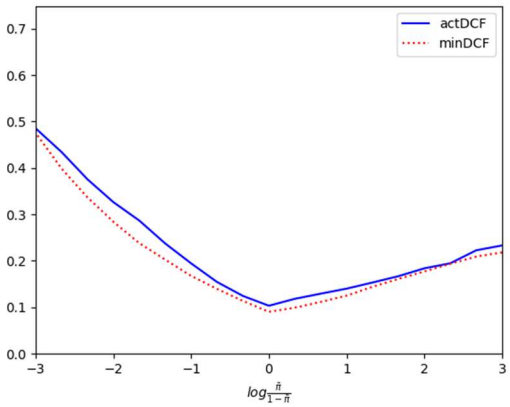


Figure 58 QLR_QSVM_Zscore

As we can see the score are well calibrated so there is no need to pre-process it.

Below the performance of the single classifiers over the evaluation set.

	actDCF			minDCF		
	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9	$\tilde{\pi}$ =0.1	$\tilde{\pi}$ =0.5	$\tilde{\pi}$ =0.9
Model						
MVG Zscore	0.288	0.094	0.094	0.287	0.091	0.091
QSVM Zscore (π_T =0.5)	0.434	0.099	0.099	0.383	0.093	0.093
QLR Zscore (π_T =0.5)	0.317	0.091	0.091	0.318	0.090	0.090

We can observe that the delivered system achieves very close performance with respect to what we would have obtained using better hyper-parameters and model configurations.

As we can see from the graph below there is no strict need to calibrate the scores.

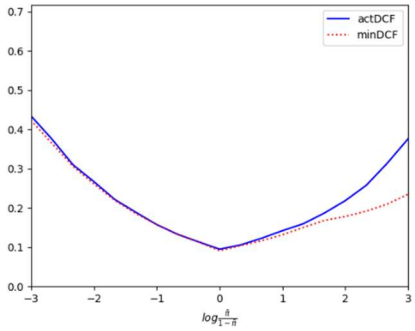


Figure 59 MVG

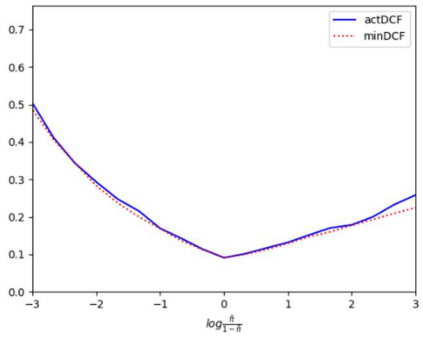


Figure 60 QLR

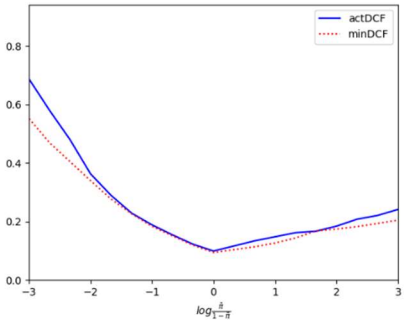


Figure 61 QSVM

CONCLUSION

Again the fused model performs slightly worse than single models.

The striking resemblance between the validation and evaluation sets strongly suggests that the evaluation dataset and training data have similar distribution. This highlights the effectiveness of the choices and strategies employed during the training phase when applied to the test data.