Politecnico di Torino

Machine Learning & Pattern Recognition

Project - Gender Identification 2022/2023

Arpino Vittorio – s301607

Abstract

The goal of this report is to analyze a dataset composed of various low-level images of males and females using various Machine Learning (ML) algorithms. Initially, an analysis of the dataset structure will be carried out, trying to understand its distribution, and then we will proceed to analyze the various classifiers in order to deliver a system that is able to classify as good as possible our samples considering the minimum cost.

Index

Application Information

The training dataset is composed of 2400 samples divided into 720 males and 1680 females, so it is very biased towards females who represent 70%. Each sample is composed of 12 features and represents a low-dimensional image obtained by mapping the images onto a common low-dimensional manifold. These embeddings have no physical interpretation, and the samples belong to 3 different age groups, each of which is defined by a different distribution for the embeddings, however age information is not available. The test dataset, on the other hand, is made up of 6000 samples of which 4200 males and 1800 females therefore does not correspond in class proportion to the training dataset but respects all the other information.

Dataset Analysis

Now we start to analyze the dataset with some plots, on starting from to see the distribution of our features over histograms (before we have centered our dataset):

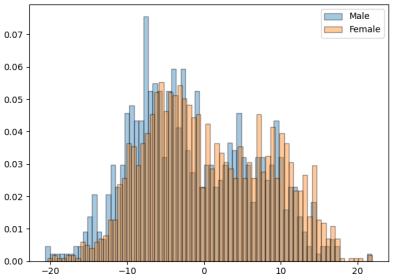

Figure 1. Feature Dataset 1

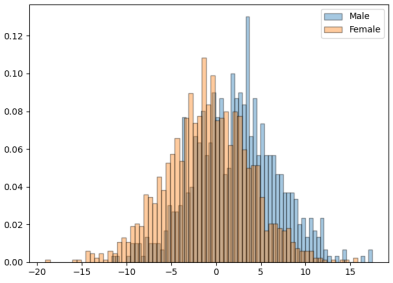

Figure 2. Feature Dataset 2

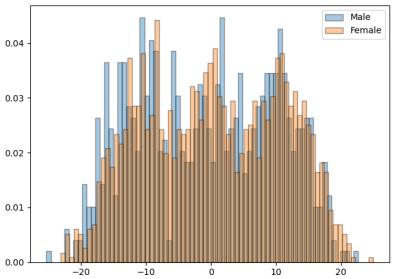

Figure 3. Feature Dataset 3

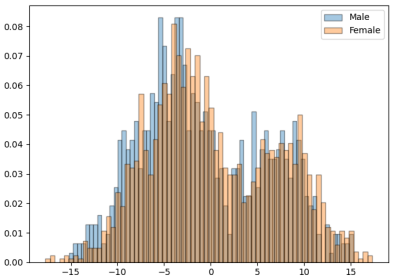

Figure 4. Feature Dataset 4

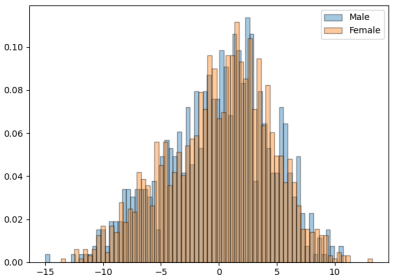

Figure 2. Feature Dataset 5

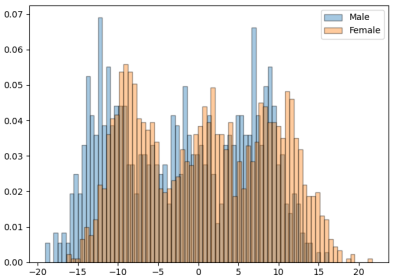

Figure 6. Feature Dataset 6

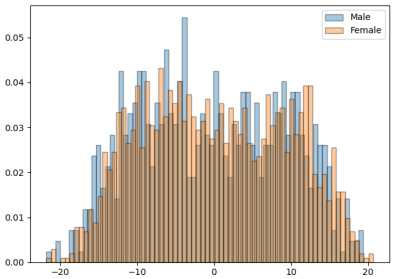

Figure 7. Feature Dataset 7

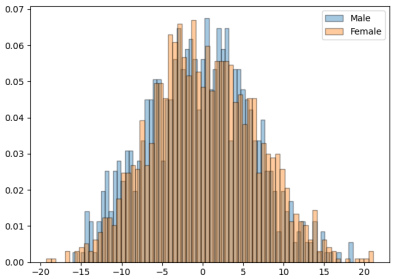

Figure 8. Feature Dataset 8

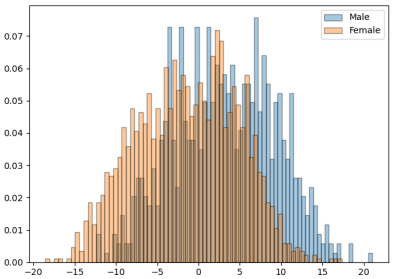

Figure 9. Feature Dataset 9

As we can see, there some histograms, such as figure 3-6-7, that remembers a gaussian with 3 components and it reflects the 3 groups of ages that we have talked before. Moreover, there are

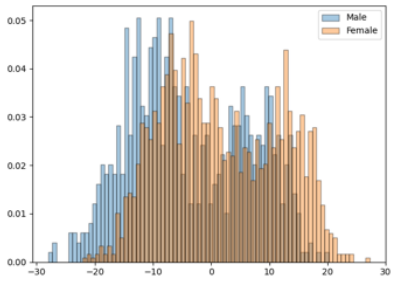

Figure 10. Feature Dataset 10

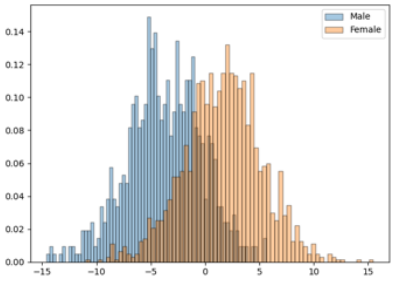

Figure 11. Feature Dataset 11

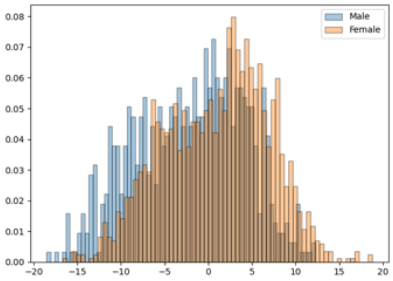

Figure 12. Feature Dataset 12

some histograms that remember the gaussian density such as figure 5-8-2. In general, the distribution for the single feature is the same for both the classes but there are some distributions that allow us to distinguish some class to another, for example in the figure 11 we have the most discriminant feature for our dataset.
A transformation that we can use is the Linear Discriminant Analysis (LDA), this preprocessing step allows us to understand if the features are linearly discriminable, so this means that if a linear and/or gaussian model may perform better than no-gaussian and/or no-linear model. LDA founds C-1 directions over to plot our features where C is the number of classes (in our case we have only 1 dimension because we trait a binary classification problem):

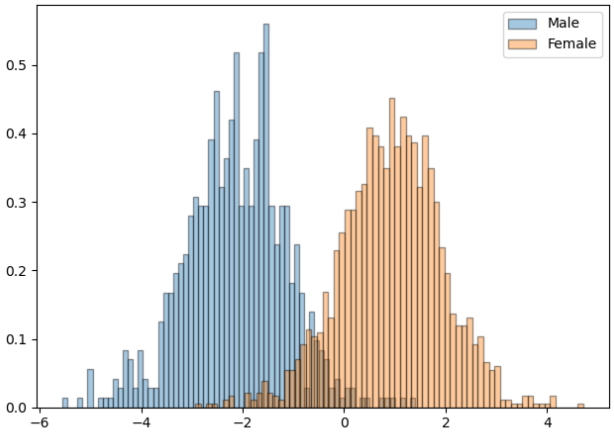

Figure 16. LDA plot for datasets feature

As we can see from this plot (Figure 16), classes are discriminable but there is a small region where we can make some mistakes. However, if we see the scatter plots for our dataset, the distributions are like gaussians one so this means that gaussians models can perform well (we have taken the most represent scatter plots and all the others are like these):

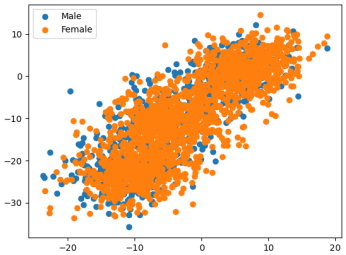

Figure 17. Scatter Plot Feature 0 and Feature 2

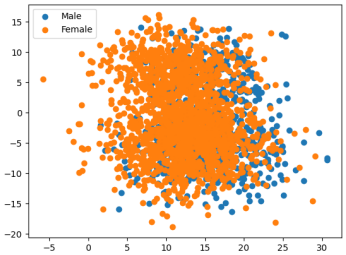

Figure 18. Scatter Plot Feature 1 and Feature 3

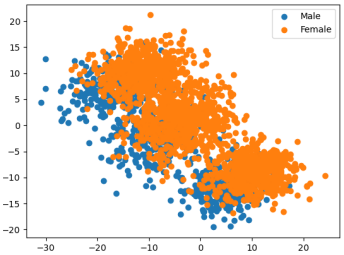

Figure 19. Scatter plot Feature 9 and Feature 5

It seems to have gaussian distributions with more components (cluster or subsets) and this idea is related to the fact that the dataset is composed by three groups of age, so this explains why we see that.

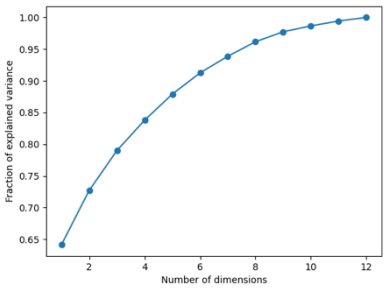

Figure 20. Explained Variance

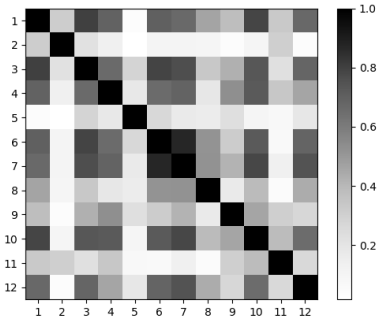

Figure 21. Heatmap for all dataset

Now, turn the attention over the correlation analysis for our dataset therefore we will analyze the Pearson Correlation for our features. We start to consider the heatmap inside the figure 21, as we can see there are some features strongly correlated, such as 1-10 and 3-6, and other features uncorrelated, such as 2-5. Most of the features are slightly correlated among each other (the correlation coefficient is near to 0.4/0.6), so this suggests that we may have benefit if we map data from 12- dimensional to 10-dimensional space in order to reduce the number of parameters to estimate for a model. The figure 20 explains us that if we remove 2 dimensions (so we pass from 12 to 10), we will maintain the 97% of the variance so we can remove 1/2 dimensions maintaining a lot of information.

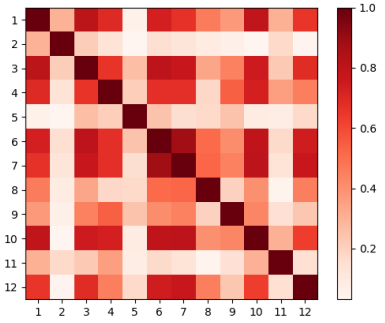

Figure 22. Heatmap for female class

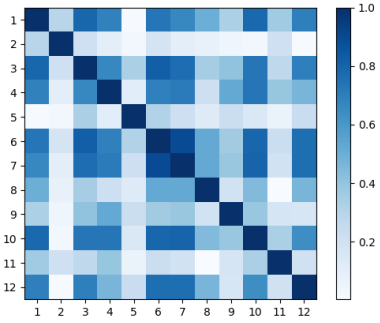

Figure 23. Heatmap for male class

Now, turn the attention over the heatmaps for a single class. As we can see, these two heatmaps are very similar so this means that the Multivariate Gaussian Model (MVG) and Tied Gaussian Model (TMVG) have similar performance and that if we consider the Naïve Bayes model the performance will be worst because there is moderately correlation between the data (this regards also the Tied Model with Naïve assumption).

Validation Methodology

To thoroughly evaluate various models and select the most suitable one, we will employ different preprocessing techniques and implement a K-Fold cross-validation approach with K=5. We have chosen this methodology because it allows us to have a larger amount of data for both training and validation purposes. It's important to note that our dataset is highly unbalanced, meaning that the number of instances in different classes is significantly imbalanced. The primary application we are focusing on is denoted as ($\tilde{\pi}$, C_{fn}, C_{fp}) = (0.5, 1, 1). However, we also intend to consider other types of applications, specifically ($\tilde{\pi}$, C_{fn}, C_{fp}) = (0.1, 1, 1) and ($\tilde{\pi}$, C_{fn}, C_{fp}) = (0.9, 1, 1).
To measure the performance of our models, we will utilize a metric called normalized minimum detection costs. This metric considers the costs associated with different types of errors and provides a comprehensive evaluation of the model's effectiveness in detecting the target class.
By employing these rigorous evaluation techniques and considering various preprocessing methods, we aim to identify the best-performing model that can effectively handle the challenges posed by the imbalanced dataset and deliver accurate results across different application scenarios.

Classifiers Performances

Gaussian Models

Now we will consider the Gaussian Classifiers and we try to use different pre-processes techniques, like Z-Score and PCA, to determine if they are effective; and we will try a combination of them.
In the following, we report the results for different scenarios:

	RAW			Z-Score		
Model	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**
MVG	0.121	0.308	0.347	0.121	0.308	0.347
NB	0.463	0.784	0.789	0.463	0.784	0.789
TMVG	0.114	0.302	0.336	0.114	0.302	0.336
TNB	0.462	0.781	0.8	0.462	0.781	0.8

Table 1. Gaussian Table for Raw and Z-Score results

	RAW + PCA (12)			Z-Score + PCA (12)		
Model	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**
MVG	0.121	0.308	0.347	0.121	0.308	0.347
NB	0.126	0.323	0.349	0.122	0.32	0.343
TMVG	0.114	0.302	0.336	0.114	0.302	0.336
TNB	0.122	0.308	0.353	0.119	0.288	0.333

Table 2. Gaussian Table for Raw and Z-Score with Pca (12)

As we can see from these tables, the results are in linear with our expectation. The Z-Score tries to transform the data by simply to center and divide by the standard deviation, so the results are like what we see with the RAW features. The Naïve Bayes assumption over the features is not so strong

(as we expected) due to fact that, as we have seen in the heatmaps, there is a moderately correlation among the features. In order to avoid this problem, we have tried to project our samples in a space where we maintain the highest variance and they are the directions that we have found with PCA (this corresponds to diagonalize the covariance matrix). The result over PCA (12) gives us better result over models that use the naïve bayes assumption (remember for these types of models is important that features are uncorrelated so the covariance matrix should be diagonal as possible and PCA gives us orthogonal directions where to plot our data, given that orthogonality means independence, our projected data will be independent).
Another important fact is that some models preserve the performances, and this is due to fact that PCA preserves the principal discriminant information in the leading directions, so MVG and TMVG do not lose the information and the performances are the same.
MVG and TMVG have the same performance, in general, because the MVG assumes that we have two different covariance matrices for the classes, but, given that the covariance matrixes of two classes are similar, the TMVG performs in the same way. Remember that a generic decision rule of Gaussian Model is described by:

$$s(x) = x^T A x + x^T b + c$$

where:

- $A = -\frac{1}{2}(\Lambda_1 - \Lambda_0)$ where Λ_x represents the precision matrix for the class x
- $b = (\Lambda_1 \mu_1 - \Lambda_0 \mu_0)$ where μ_x represents the mean for the given class x
- $c = -\frac{1}{2}(\mu_1^T \Lambda_1 \mu_1 - \mu_0^T \Lambda_0 \mu_0) + \frac{1}{2}(\log|\Lambda_1| - \log|\Lambda_0|)$

but if the covariance matrixes are equals (means also that the precision matrixes are equal because $\Lambda = \Sigma^T$), the decision rule will be $s(x) = x^T b + c$. This explains why the performance of MVG and Tied are very similar.

	RAW + PCA (11)			Z-Score + PCA (11)		
Model	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**
MVG	0.122	0.31	0.347	0.124	0.313	0.35
NB	0.134	0.314	0.367	0.126	0.317	0.337
TMVG	0.123	0.303	0.352	0.121	0.299	0.355
TNB	0.124	0.304	0.369	0.125	0.29	0.356

Table 3.Gaussian Table for Raw and Z-Score with Pca (11)

	RAW + PCA (10)			Z-Score + PCA (10)		
Model	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**
MVG	0.169	0.423	0.479	0.192	0.41	0.538
NB	0.172	0.434	0.479	0.186	0.418	0.543
TMVG	0.169	0.394	0.47	0.186	0.427	0.532
TNB	0.173	0.383	0.465	0.182	0.404	0.535

Table 4. Gaussian Table for Raw and Z-Score with Pca (10)

Moreover, if we try to reduce the dimensionality to 11, we maintain a good fraction for the explained variance and so the performances are in linear with a dimension space of 12. If we try to reduce further the space, we lose information, and the results get worst. Given the results, we

have decided to continue to use Z-Scored and Raw data, but we will not consider further dimension reduction such as PCA (10) and PCA (11) because we have not any benefit.

Discriminative Models

Now we will focus on the discriminative models such as Logistic Regression Model (LR) and Quadratic Logistic Regression Model (QLR).
These types of models try to directly estimate the posterior distribution using the sigmoid function and, in order to compute the score, we will use the formulation:

$$s(x) = w^T x + b$$

where:

- "w^T" represents the orthogonal vector respect to the hyperplane that we have defined
- "b" represents the bias term as the how much we need to move from the hyperplane in horizontal

We need to estimate as parameter "w" and "b" in order to define our decision rule, so our objective will be:

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

where:

- "z" is the vector that has elements -1 or 1 depend for sample x (for example, in position "i" we will have 1 if the x_i belong to class 1 or -1 if it belongs to class 0)
- "n_x" represents the number of samples inside the class x
- "π_T" represents the prior that helps us to generalize the model

Moreover, the LR does not make any assumption over the distribution of the data but simply find a hyperplane that maximizes the posterior probability.
An important thing is that we add to objective function a regularization term (in order to avoid the problem when the classes are linearly separable), so we will estimate with training the best value for λ. The interval chosen to find the best value is from 10^{-5} to 10^{5} in a logarithmic fashion with initially setting $\pi_T = 0.5$ (given that's our main task).

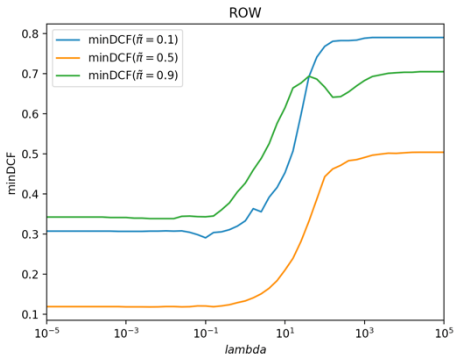

Figure 24. Plot LR with $\pi_T = 0.5$ and RAW features

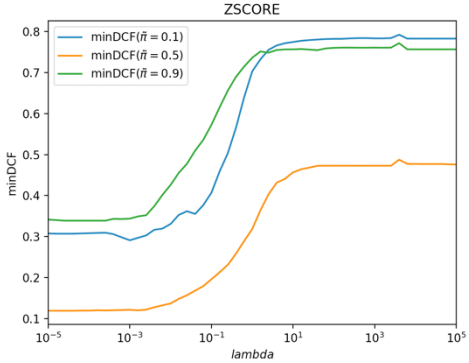

Figure 25. Plot LR with $\pi_T = 0.5$ and Z-Score features

As we see from these plots, the minimum is reached by setting λ=0, so we will consider it as the best value for hyper parameter λ. We do not report the plot and the results for PCA (12) because it's very similar to Raw one.
In order to analyze the performance, we will consider the following table:

	RAW			Z-Score		
Model	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**
LR (π_T =0.5)	0.119	0.308	0.343	0.119	0.308	0.343
LR (π_T =0.9)	0.116	0.338	0.321	0.116	0.338	0.321
LR (π_T =0.1)	0.121	0.29	0.37	0.121	0.29	0.37

Table 5. LR Table for Raw and Z-Score with $\lambda = 0$

When the value of λ (lambda) is too large, we observe that the model struggles to correctly classify the samples. This occurs because increasing λ leads to excessive "generalization," where the model becomes overly simplistic. As a result, it may fail to capture the complexities and nuances of the data, leading to poor classification performance.
On the other hand, for small values of λ, the model tends to achieve a good separation on the training data and can generalize well to unseen data. In this case, the regularization term has little to no benefit in improving the model's performance. Therefore, it is more advantageous to report the results of the non-regularized version of the model.
To strike a balance between overfitting and underfitting, a suitable value of λ could be chosen and can be in the order of 10^{-4}. This value minimizes overfitting while still delivering comparable performance to other small λ values. By selecting this value, we can mitigate the risk of overfitting the model to the training data and improve its ability to generalize to new, unseen data.
Since Z-Score is a linear transformation, the performances obtained on the raw data and the Z-Scored data are the same. Using different values for π_T does not improve a lot the model, as we can see there is a slightly improvements if we try to use π_T =0.9 but it's caused by the fact that the dataset is imbalanced to the class 1 (female class).
Now we will focus on QLR version of the model and as we did for the LR, we will consider the same interval for λ. As we have done previously, we will consider just the plots of Raw data and Z-Score data (because the result with PCA (12) are similar).
Differently from the LR, the QLR model has worse performance (as we expected) given that the scatter plots, histograms and LDA analysis demonstrate that quadratic models can perform worse than linear ones because the distribution for the data is not separable well with quadratic rules but with linear decision rules.

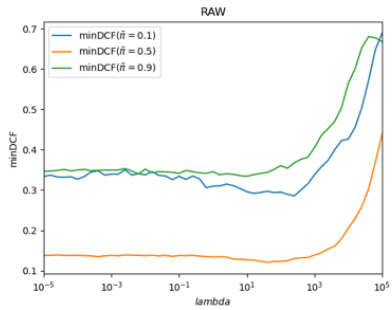

Figure 26. Plot QLR with $\pi_T = 0.5$ and RAW features

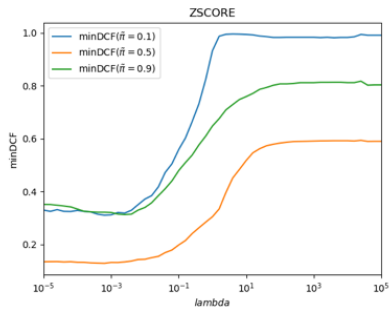

Figure 27. Plot QLR with $\pi_T = 0.5$ and Z-Score features

Differently, from the linear one case the results are different from Raw features and Z-Scored because we have done a feature expansion operation that compute the dot product inside another embedding space, so the model is sensitive to transformation of data. The value of λ chosen will be 10^2 for the Raw Features while 0 for the Z-Score features (also here we can do the same consideration of LR about the value λ).
As we have anticipated, the QLR model has worse performance than the LR (as we expected):

	RAW (λ = $\mathbf{10^2}$)			Z-Score (λ = 0)		
Model	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**
QLR (πt =0.5)	0.123	0.295	0.36	0.133	0.326	0.351
QLR (πt =0.9)	0.132	0.377	0.357	0.145	0.354	0.369
QLR (πt =0.1)	0.123	0.282	0.411	0.141	0.352	0.366

Table 6. QLR Table for Raw and Z-Score with λ = 0 and λ = 10^2

No Probabilistic Models

Now we consider the no probabilistic models like Support Vector Machine models (SVMs), and they are called in this way because the score that they produce has not a probabilistic interpretation. We will start to consider the linear SVM and, as we did before, we try to analyze the model and the assumption.
As for the LR, the SVM model tries to find a hyperplane that divides in the best way the classes but, differently from the LR model, it tries to find the hyperplane that maximizes the margin, and we call it as "maximum margin hyperplane". As we did for Discriminative models, we will consider the prior-weighted version of the model so the objective (or dual problem SVM) that we want to maximize will be:

$$\max_{\alpha} \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{H} \boldsymbol{\alpha} \quad \text{subject to} \quad 0 \leq \alpha_i \leq C_i,\ i = 1, \ldots, n$$

where:

- $C_i = C\frac{\pi_T}{\pi_T^{emp}}$ or $C_i = C\frac{\pi_F}{\pi_F^{emp}}$ depends on if the sample "i" is male or female
- π_T^{emp} and π_F^{emp} represent the empirical prior evaluated over the training dataset

Even here we have a hyper-parameter to tune "C" and we will consider an interval 10^{-5} to 10^{5} in a logarithmic fashion. In general, we expect, as also confirmed previously, that a linear decision rule will be better than a quadratic one in order to separate the classes.

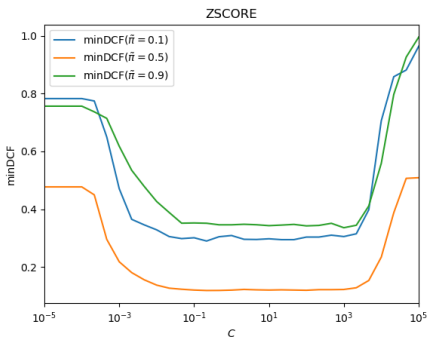

Figure 28. Plot SVM with π_T = 0.5 and Z-Score features

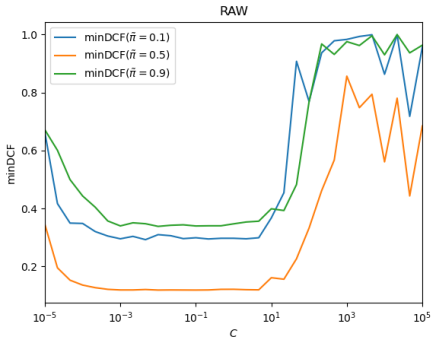

Figure 29. Plot SVM with π_T = 0.5 and RAW features

In general, C represents the margin, and it can be seen as a tradeoff between to have a low error in training or a large margin. If C goes to infinite, we try to have low error on training but, at same time, we do not generalize very well and we go in overfitting; differently if C goes to 0 means we do not care about the training data and we want a large margin. The choice of C is critical but, seeing the plots, we have decided to use C=10 to evaluate the performance, which optimizes different applications.

	RAW (C = 10)			Z-Score (C = 10)		
Model	**π = 0.5**	**π = 0.1**	**π = 0.9**	**π = 0.5**	**π = 0.1**	**π = 0.9**
SVM (π_T =0.5)	0.134	0.292	0.357	0.12	0.298	0.343
SVM (π_T =0.9)	0.121	0.338	0.359	0.115	0.343	0.335
SVM (π_T =0.1)	0.123	0.31	0.363	0.128	0.307	0.374

Table 7. SVM Table for Raw and Z-Score with C = 10

As we can see from the table, the results are good and like the LR (just for π_T =0.5 seems to be a little worse) and our assumption about the linear decision rule is reinforced. Surprisingly, the performance for π_T =0.1 are better than when π_T =0.5. Moreover, the performances for RAW and Z-Score features are different and the model receives a boost, if we use the Z-Score (this is due because with z-score we center the data and try to obtain a unit variance, so we will have benefit). As we can see from the obtained results, this model is one of the best to classify our sample so we will consider it (in particular, SVM (π_T =0.9)).

Now we will consider the SVMs models that explore the concept of the kernel function. In general, if we have a function that computes efficiently the dot product inside an expanded space, that we can call "k (x1, x2)", we can use it for training and scoring. This function will substitute the dot product inside the H matrix in the dual problem (remember the $H_{ij}=z_i z_j x_i^T x_j$) so we will obtain H= $z_i z_j k(x_i, x_j)$. It means that we want to find a maximum margin hyperplane inside the expanded space but that corresponds inside the original space as a quadratic/kernel rule. We will explore the polynomial SVM with a degree 2 (so it will be quadratic SVM) and a radial based version. Both the models depend on C and only the radial version depends on a parameter γ that we have tuned with value 10^{-1}, 10^{-2} and 10^{-3}.

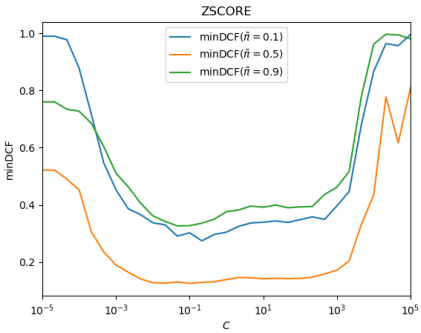

Figure 30. Plot PolSVM with π_T = 0.5 and Z-Score features

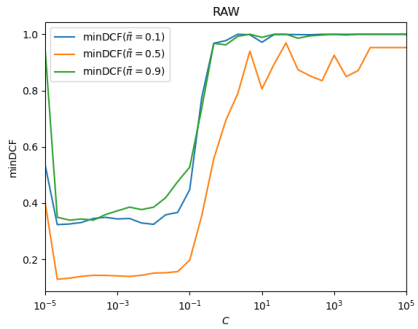

Figure 31. Plot PolSVM with π_T = 0.5 and Raw features

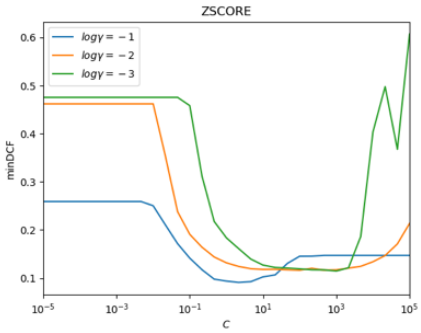

Figure 32. Plot RadialBasedSVM with π_T = 0.5 and Z-Score features

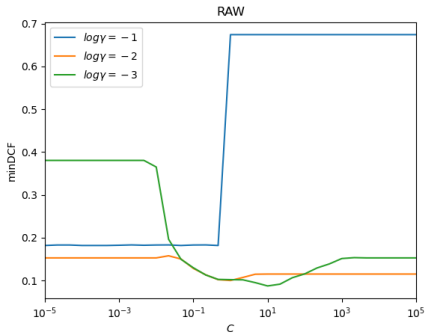

Figure 33. Plot RadialBasedSVM with π_T = 0.5 and Raw features

Also here, the choice of C is important and, based on the plots above, we have decided to choose C = 10^3 for Raw Features while C = 10^{-1} for Z-Score Feature. Instead, for the radial version based, the performances are good with γ = 0.001 and C = 10, in the case of Raw feature, and γ = 0.1 and C = 5, in the case of Z-Scored features.

	RAW (C = $\mathbf{10^3}$)			Z-Score (C = $\mathbf{10^{-1}}$)		
Model	**π = 0.5**	**π = 0.1**	**π = 0.9**	**π = 0.5**	**π = 0.1**	**π = 0.9**
QSVM (π_T =0.5)	0.141	0.341	0.368	0.126	0.302	0.327
QSVM (π_T =0.9)	0.153	0.369	0.35	0.135	0.382	0.298
QSVM (π_T =0.1)	0.159	0.379	0.449	0.148	0.33	0.44

Table 8. QSVM Table for Raw and Z-Score

	RAW (γ = 0.001, C = 10)			Z-Score (γ = 0.1, C = 5)		
Model	**π = 0.5**	**π = 0.1**	**π = 0.9**	**π = 0.5**	**π = 0.1**	**π = 0.9**
RBSVM (π_T =0.5)	0.134	0.292	0.357	0.12	0.298	0.343
RBSVM (π_T =0.9)	0.121	0.338	0.359	0.115	0.343	0.335
RBSVM (π_T =0.1)	0.123	0.31	0.363	0.128	0.307	0.374

Table 9. RBSVM Table for Raw and Z-Score

Quadratic SVM obtains worse performance than Quadratic LR, which are not good as the linear version of the two mentioned models. However, a Radial Basis kernel SVM can act as a good model respect to the Quadratic one and Quadratic LR (just the Z-Score feature for our main application).

Gaussian Mixture Models

The last type of classifier that we will consider is the Gaussian Mixture Model. In general, this type of model tends to be used for density estimation problem and the assumption is that our data can be distributed as gaussian with one or more components. The previous results for gaussian models and the dataset analysis suggest that the models, that can be perform very well, will be GMM/Tied GMM for the same reason that we have explained before in the analysis of gaussian models. We will expect to obtain the best results for Gaussian Mixture Models with 2/4 components because, as we have seen during the dataset analysis, our training set has a distribution that remember a gaussian with 3 components/clusters and the numbers more towards to 3 are 2 and 4 (in general the Gaussian Mixture Models are trained with a number of clusters in a pow of 2 so we will not consider GMM with 3 components).
In this case the hyperparameter to tune is the number of components for each gaussian:

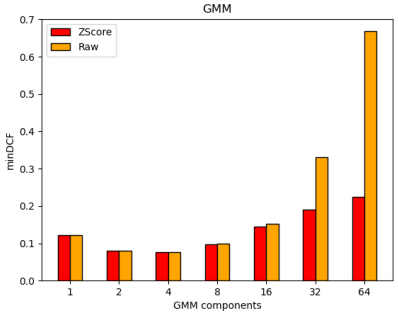

Figure 35. Plot GMM for Raw and Z-Score data

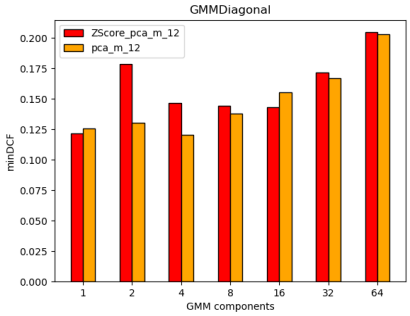

Figure 34. Plot GMMDiagonal for Raw and Z-Score data with Pca (12)

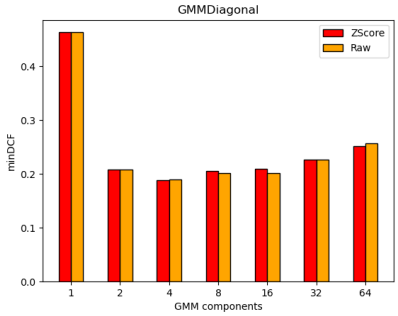

Figure 34. Plot GMMDiagonal for Raw and Z-Score data

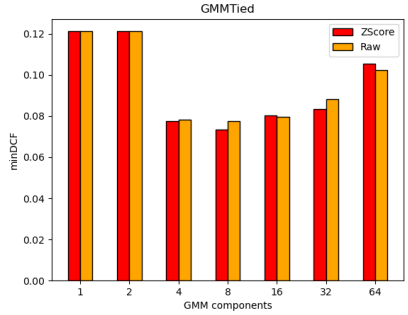

Figure 39. Plot GMM for Raw and Z-Score data

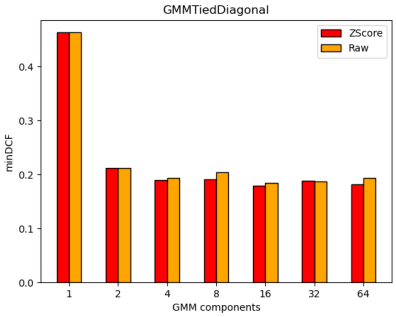

Figure 40. Plot GMMTiedDiagonal for Raw and Z-Score data

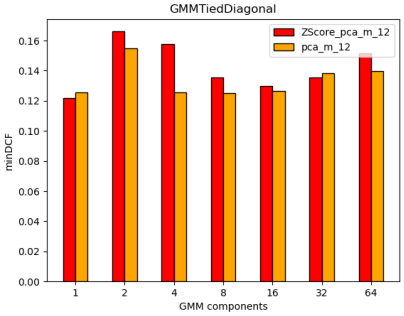

Figure 41. Plot GMMTiedDiagonal for Raw and Z-Score data with Pca (12)

We have considered also the PCA (12) preprocess step for models which have the assumption of diagonal covariance matrix (GMMDiagonal and GMMTiedDiagonal). As we expected, the two diagonal models do not produce good result because we have moderately correlation as we have seen in the heatmaps. The performances will be comparable to the other models, if we project our sample over PCA dimension space, in order to have a diagonal covariance matrix. We report in the following table, only the best models and we do not consider the gaussians with one component because they are already described above.

	RAW			Z-Score		
Model	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**
GMM (4 components)	**0.076**	0.229	0.205	**0.076**	0.229	0.205
GMMTied (8 components)	0.078	0.247	0.207	**0.073**	0.251	0.187
GMMTiedDiagonal (16 components)	0.184	0.486	0.47	0.179	0.525	0.472
GMMDiagonal (4 components)	0.189	0.46	0.482	0.188	0.46	0.481

Table 10. Gaussian Mixture Models best result for Raw and Z-Score

As we can see, the Gaussian Mixture models give us good results. The GMM (Full Covariance Version) trained on Z-Score is the best model (as we have analyzed before) with GMMTied with 8 components. It was reasonable, given that reflects the distribution of our data as we have seen in the scatter plots.

	RAW + PCA (12)		
Model	**$\pi = 0.5$**	**$\pi = 0.1$**	**$\pi = 0.9$**
GMMDiagonal (4 components)	0.12	0.321	0.321

Table 11. Gaussian Mixture Models best results for Raw with Pca(12)

The best GMM Tied Diagonal is the model with 1 component, so we do not report the result because it's a simple MVG with Tied Covariance. In the case of GMM Diagonal we report the result just the PCA (12) with raw features result because the best for Z-Score with PCA (12) is the model with 1 component.
The reason why the Tied assumption helps in this case is due, most likely, to the fact that the estimates of the two within class covariances (which each of them is shared among the Gaussian component of the classes) are more robust than the one obtained by not considering it.

The Full-Tied Covariance model with 8 components trained on Z-Scored data is very slightly better than the one with 4 components. However, we know from the application information that the samples belong to 3 different age groups, so we will choose 4 as number of components since it follows better how the data are naturally clustered.

The best classifiers at this point are, from each family:

Model	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
TMVG (Raw + PCA (12))	0.114	0.302	0.336
LR (π_T =0.9, λ=0; Z-Score)	0.116	0.338	0.321
RBSVM (π_T =0.9, C=5, γ=0.1; Z-Score)	0.115	0.343	0.335
GMM (4 components; Z-Score)	0.076	0.229	0.205
SVM (π_T =0.9, C=10; Z-Score)	0.115	0.343	0.335

Table 12. Best Classifier on Training

Since the gap between the GMM model and the other one is large, for all the three considered application, we considered it as the main candidate for the evaluation phase. However, we need to check if the scores of this model need to be calibrated or not.

Score Calibration

Until now, we have considered only the minimum detection costs as metric to evaluate different models, however, the issue is that the cost that we pay depends on a threshold. In order to understand also if the threshold is the theoretical one, we will use a metric known as actual DCF. The approach that we will employ is the one using Logistic Regression because it acts as posterior log-likelihood ratio so we will recover the calibrated score by simply subtracting for the theoretical threshold.
In order to estimate the parameters of calibration function, we will consider a K-Fold Approach (also because the number of samples that we have is limited).
The GMM Full-Tied Covariance model on Z-Scored data is the best promising one, so we will calibrate its scores.

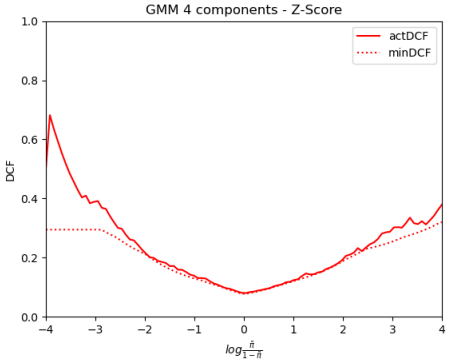

Figure 42. GMM Uncalibrated Model

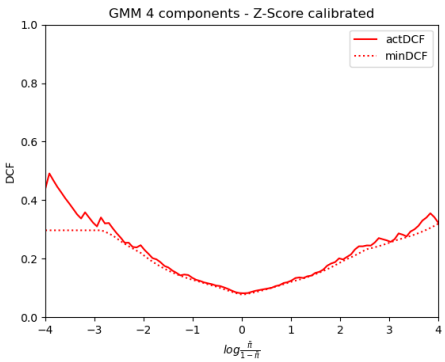

Figure 43. GMM Calibrated Model

The scores do not need calibration; indeed, the transformation seems to bring any particular benefit to the model. As we can see from the results below only the unbalanced application with $\tilde{\pi} = 0.9$ slightly improved.

minDCF	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
-	0.076	0.233	0.203

Table 13. Result minDCF for calibrated GMM

actDCF	π = 0.5	π = 0.1	π = 0.9
calibrated	0.081	0.236	0.211
uncalibrated	0.079	0.253	0.214

Table 14. Result actDCF for calibared and uncalibrated GMM

Since the performance between the uncalibrated scores and the calibrated ones do not differ too much, for simplicity we'll kept the uncalibrated model as a final classifier to deliver.
Moreover, we will consider the other model and we try to calibrate them:

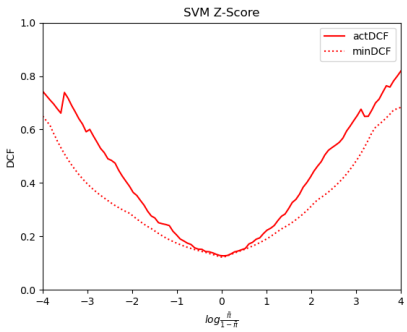

Figure 44. SVM Uncalibrated version

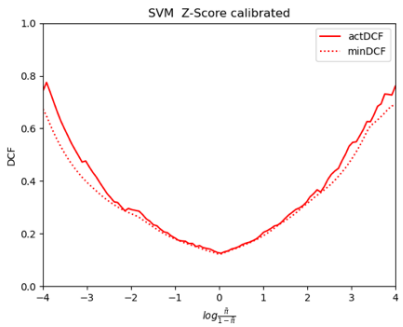

Figure 45. SVM Calibrated Version

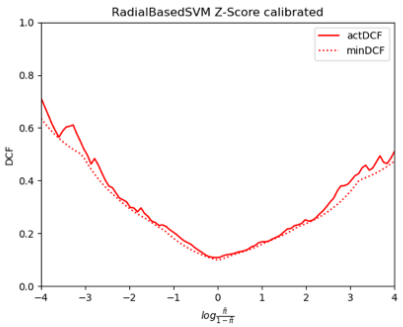

Figure 46. RBSVM Calibrated Version

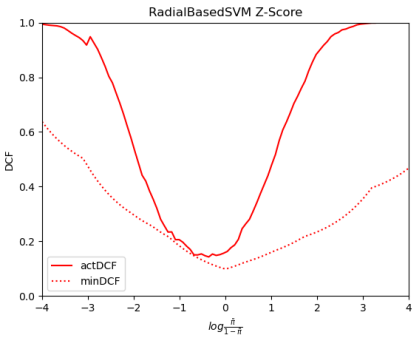

Figure 47. RBSVM Uncalibrated Version

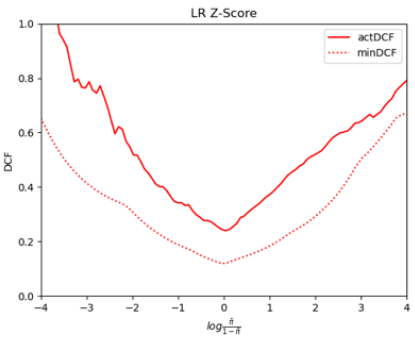

Figure 48. LR Uncalibrated Version

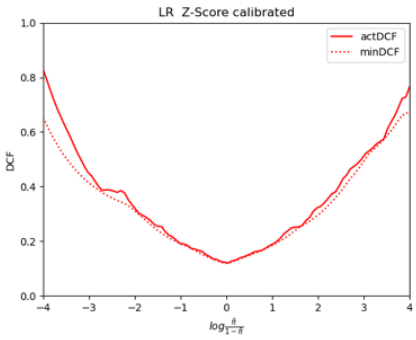

Figure 49. LR Calibrated Version

As we can see: LR, RBSVM and SVM have the score not calibrated and if we make calibration, we can obtain some improvements (the RBSVM has better performance on our main application):

minDCF	π = 0.5	π = 0.1	π = 0.9
SVM	0.116	0.345	0.338
RBSVM	0.098	0.321	0.252
LR	0.117	0.336	0.326

Table 15. Result for Calibrated Models

actDCF	Uncalibrated			Calibrated		
Model	π = 0.5	π = 0.1	π = 0.9	π = 0.5	π = 0.1	π = 0.9
SVM	0.234	0.364	0.7	0.117	0.366	0.358
RBSVM	0.159	0.652	0.922	0.107	0.328	0.257
LR	0.237	0.598	0.546	0.119	0.367	0.352

Table 16. Result in actDCF for calibrated models

Fusion

The method that we have used to make calibration allows to combine the different output for different models. We can also try to combine different classifiers in order to combine they different decisions and improve the performance. We try to combine as fusion the three best model that we have chosen: LR, SVM and GMM. The method that we can employ to make fusion is the same for the calibration, we will use the K-Fold Approach and try to combine the decisions with prior-weighted logistic regression.

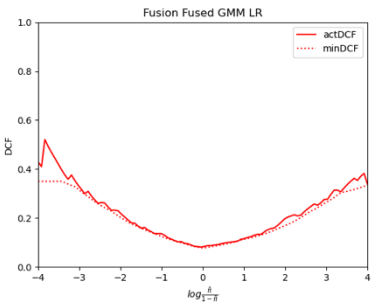

Figure 50. Fusion GMM + LR

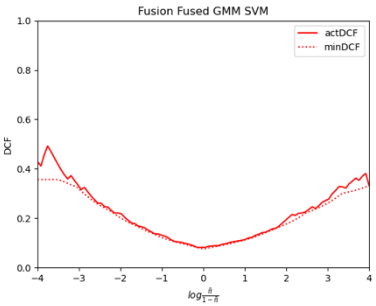

Figure 51. Fusion GMM + SVM

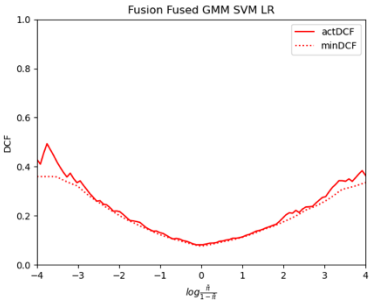

Figure 52. Fusion GMM + SVM + LR

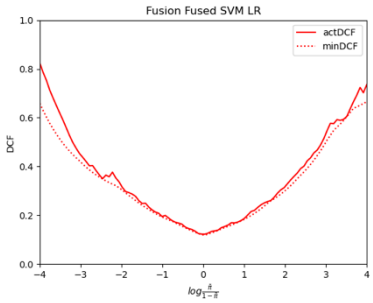

Figure 53. Fusion SVM + LR

As we can see from the plots, all the scores are already calibrated so we do not need add any further step in order to calibrate the models. Below we consider the results of these models in terms of minCDF:

Model	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
GMM + SVM + LR	0.076	0.221	0.19
GMM + LR	0.076	0.225	0.184
SVM + LR	0.117	0.326	0.331
GMM + SVM	0.075	0.224	0.192

Table 17. Result for Fused Models

As we can see from the table, the best model is GMM + SVM for our main application $\pi = 0.5$ but also GMM + SVM + LR and GMM + LR are towards. We can interpret these results as following: given the fact we have two types of linear decision boundaries LR and SVM (both trained with $\pi = 0.9$), they find the same hyperplane to separate the sample and so the fusion on these two models change little. Moreover, there are some improvements for $\pi = 0.1$ and $\pi = 0.9$, it seems the GMM model helps the other model in order to make less errors (in fact, SVM + LR does not provide good results on the other two types of applications). The final model that we choose as the best one for the training will be the **GMM Full Covariance with 4 components** and Z-Scored Preprocess (despite the performances are slightly better for the fusion GMM+SVM, we decided a simple solution).

Evaluation

Now we consider the quality of our choices on held-out data, and we will verify the performance. Differently from the training phase, we consider only the models that, during the previous phase, has obtained the best results.
Moreover, we do not consider the gaussian models because we incorporate them inside the Gaussian Mixture Models as a gaussian with 1 component (for example, a Full Covariance GMM is a simple gaussian model MVG).

Calibration and Fusion

We start to consider the fused and calibrated models that we have presented above and try to understand their performances on the evaluation data. Remember that GMM model is the full covariance with 4 components, the LR model is the logistic regression trained with π_T=0.9 and SVM is the model trained with π_T =0.9. All these models have obtained these results by considering the Z-Score preprocessing step on the data. We have decided to not fuse the TMVG model because it's like a gaussian, so we have taken only one model for a family.

Model	π = 0.5	π = 0.1	π = 0.9
GMM + SVM + LR	0.06	0.153	0.149
GMM + LR	0.06	0.152	0.15
SVM + LR	0.122	0.335	0.281
GMM + SVM	0.06	0.152	0.149

Table 18. Results of fusion models on evaluation set

The results are good, it seems that GMM gives to the other two models a boost over the performance and reduce the cost. This explains why the performance are the same on the main application and are (more or less) the same also on the other two. Moreover, if we see the fusion for SVM+LR, the performances are poor differently from the others. We report now the bayes error plot, in order to understand if the models need calibration:

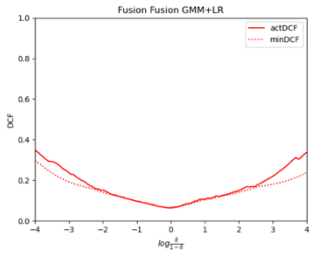

Figure 54. Fusion of GMM+LR

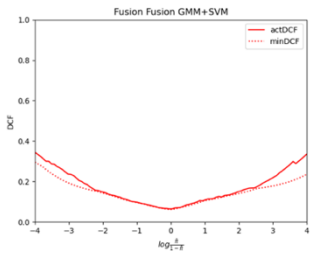

Figure 55. Fusion of GMM+SVM

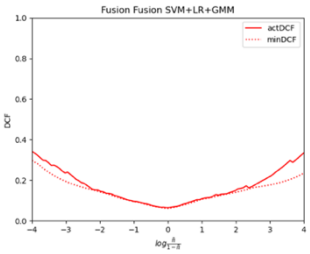

Figure 56. Fusion of GMM+SVM+LR

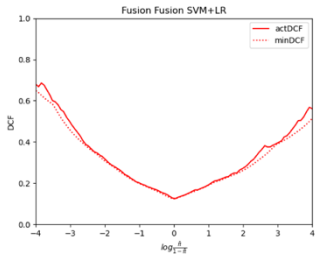

Figure 57. Fusion of LR+SVM

As we can see, the scores are well calibrated, so we do not act any calibration preprocess on the models:

	minDCF			actDCF		
Model	π = 0.5	π = 0.1	π = 0.9	π = 0.5	π = 0.1	π = 0.9
GMM + SVM + LR	0.06	0.153	0.149	0.064	0.154	0.16
GMM + LR	0.06	0.152	0.15	0.063	0.157	0.166
SVM + LR	0.122	0.335	0.281	0.123	0.34	0.3
GMM + SVM	0.06	0.152	0.149	0.064	0.154	0.161

Table 19. Results on Evaluation Set for fused models

We now also report the results for the single model with the best hyper parameters founded during the training phase:

	minDCF			actDCF		
Model	π = 0.5	π = 0.1	π = 0.9	π = 0.5	π = 0.1	π = 0.9
TMVG (Z-Score)	0.116	0.301	0.308	0.118	0.309	0.323
LR (π_T =0.9, λ=0; Z-Score)	0.123	0.339	0.279	0.216	0.648	0.51
RBSVM (π_T =0.9, C=5, γ=0.1; Z-Score)	0.098	0.332	0.196	0.149	0.569	0.897
GMM (4 components; Z-Score)	**0.06**	**0.152**	**0.157**	**0.062**	0.156	0.157
SVM (π_T =0.9, C=10; Z-Score)	0.13	0.351	0.287	0.209	0.363	0.688

Table 20. Results on evaluation set for best models

The performances are changed a lot but the best one remains the GMM with Full Covariance that has a large gap in performance in all the application. The TMVG confirms the performance, but RBSVM has better performance than when we have used in training while the other two models (LR and SVM) have worse performance than when we have used in the training phase. So now we will consider a det plot for GMM, SVM and RBSVM:

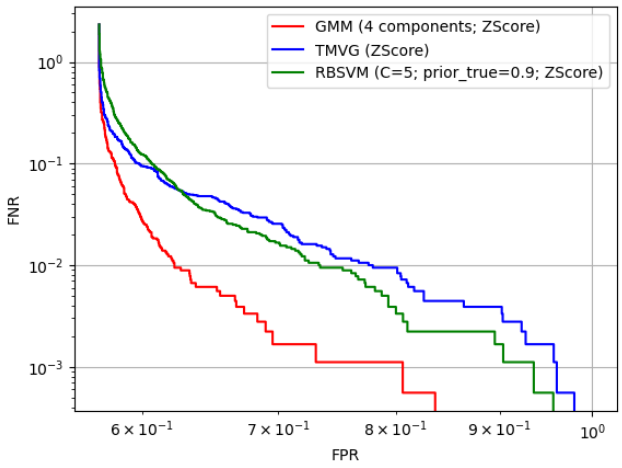

Figure 58. Det Plot for GMM, SVM and TMVG

Moreover, as we can see from the table, the gaussian models do not need any calibration and differently LR, RBSVM and SVM need to be calibrated. We will first consider the bayes error plot, in order to see the differences and then we see the same plot for calibrated scores:

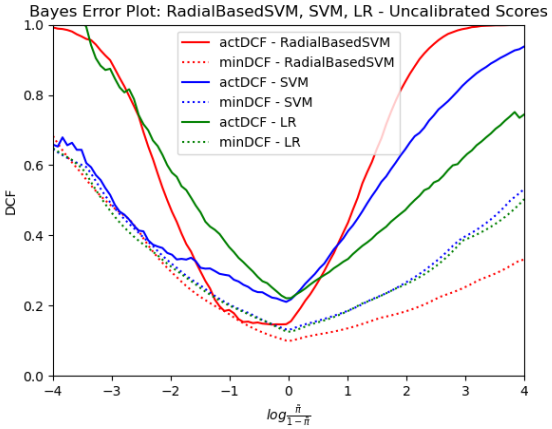

Figure 59. Uncalibrated Scores for RBSVM, SVM and LR

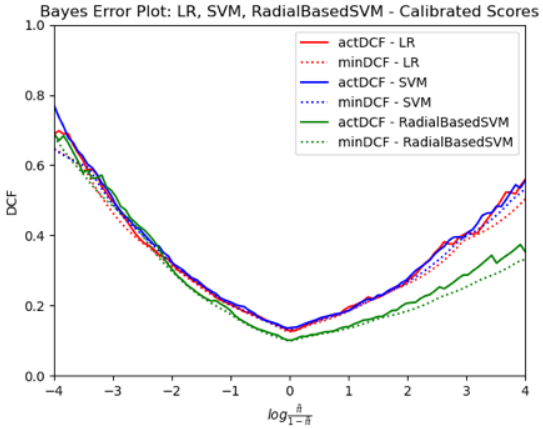

Figure 60. Calibrated Scores for RBSVM, SVM and LR

Uncalibrated Scores						
	minDCF			actDCF		
Model	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
LR (π_T =0.9, λ=0; Z-Score)	0.123	0.339	0.279	0.216	0.648	0.51
RBSVM (π_T =0.9, C=5, γ=0.1; Z-Score)	0.098	0.332	0.196	0.149	0.569	0.897
SVM (π_T =0.9, C=10; Z-Score)	0.13	0.351	0.287	0.209	0.363	0.688

Table 21. Results minDCF and actDCF for uncalibrated models

Calibrated Scores						
	minDCF			actDCF		
Model	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
LR (π_T =0.9, λ=0; Z-Score)	0.123	0.339	0.279	0.127	0.344	0.301
RBSVM (π_T =0.9, C=5, γ=0.1; Z-Score)	0.098	0.332	0.196	0.1	0.355	0.216
SVM (π_T =0.9, C=10; Z-Score)	0.13	0.351	0.287	0.133	0.363	0.311

Table 22. Results minDCF and actDCF for calibrated models

Now the scores are better calibrated as we can see from the plots and from the tables, but the best model remain GMM Full covariance with 4 components.
Below, we will present only the best model for each family of classifier (we exclude only the gaussian models because there is a correspondence between the GMM model and gaussian one). Moreover, in order to consider only the best model, we do not consider more quadratic classifiers such as QLR and Quadratic SVM because the quadratic models, as we have seen in training phase, are not the best types of models that we can use.

Discriminative Models

We first consider the logistic regression models plotting λ for the same interval that we use during the training. Our best model, during the training was LR trained with π_t = 0.9 so we will consider the plot for this type of model (only Z-Scored because the results are similar with Raw features):

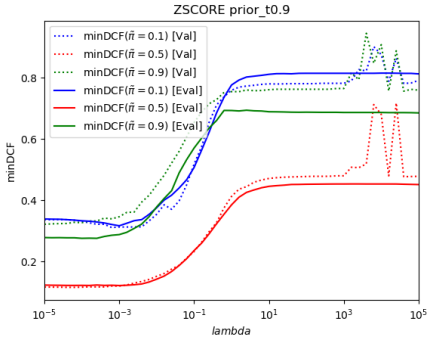

Figure 61. LR on evaluation set with π_t =0.9

The curves are overlapped until lambda is lower than 10^3 but the performances are similar a lot to what we have seen during the training, and this confirmed that the regularization does not provide any benefit. We do not report the results inside a table because the curves are very similar, and the results are a "copy" of the previous table. Instead, for the QLR we have seen that the performance was little worse than the linear version so we will not consider the evaluation for this type of model.

No Probabilistic Models

Now we consider the no probabilistic models such as SVM and Radial Based Version, given that the quadratic one does not give good result.
We consider for SVM only the data with Z-Score transformation and the results are in linear with our expectation:

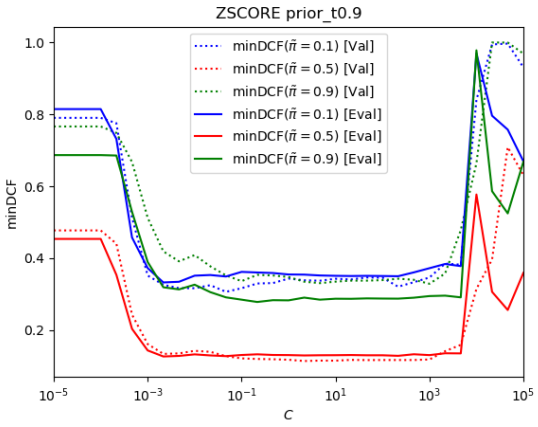

Figure 62. SVM on evaluation set with π_t =0.9

The best value chosen for C is the same, but we have little improvements on the performance as we have seen previously (we pass from 0.115 to 0.113).

Now we consider the Radial Based Version:

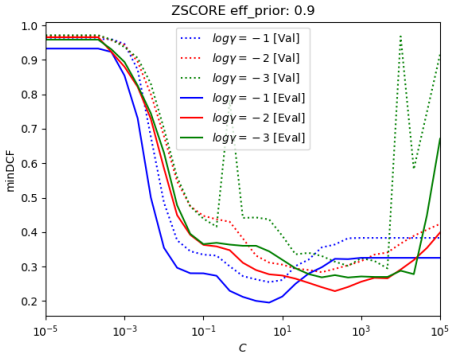

Figure 63. SVM on evaluation set with π_t=0.9

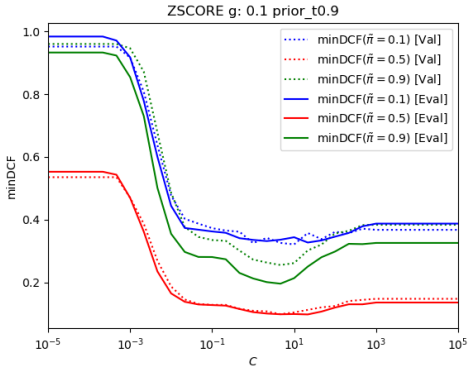

Figure 64. SVM on evaluation set with π_t=0.9

As we can see from these plots, the value for 0.1 is still the best for γ and the performances are the same also in this case by setting up γ value. During the training we have chosen C=5 but now, as we can see, 5 is not the best but a sub optimal solution with slight difference.

Gaussian Mixture Models

Now we consider the Gaussian Mixture Models and, to assess the result, we will consider only the best options for this type of model so: GMM full Covariance with Z-Score Feature, GMM tied covariance with Z-Score features, Diagonal GMM full covariance with PCA (12) + Z-Score and Diagonal Tied GMM one. We choose to plot only Z-Score features because the results with Raw feature are very similar:

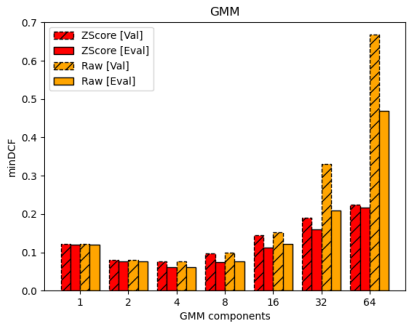

Figure 65. GMM on evaluation set

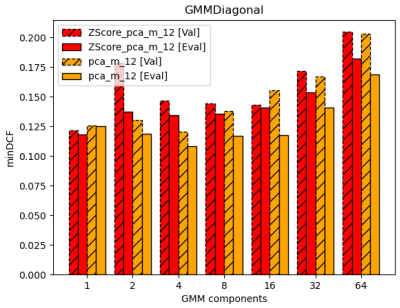

Figure 66. GMMDiagonal on evaluation set

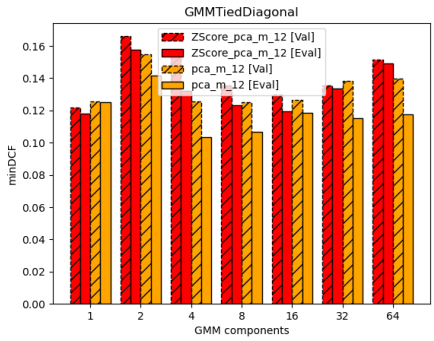

Figure 67. GMMTiedDiagonal on evaluation set

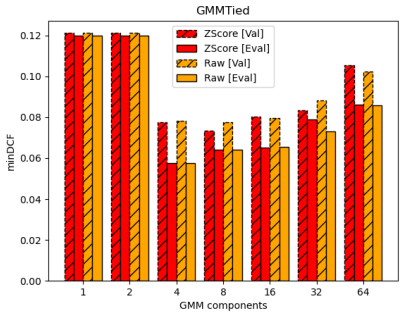

Figure 68. GMMTied on evaluation set

As we can see from the plots, we have little differences, and the best models are also in this case the GMM Tied and GMM with four components. We can evaluate the performances only of these two classifiers in order to understand if our choices was good during the training:

	RAW			Z-Score		
Model	**π = 0.5**	**π = 0.1**	**π = 0.9**	**π = 0.5**	**π = 0.1**	**π = 0.9**
GMM (4 components)	0.061	0.156	0.156	**0.06**	0.152	0.157
GMMTied (4 components)	0.058	0.178	0.16	**0.058**	0.178	0.16

Table 23. Results best GMM models

As we can see the chosen model is a sub-optimal solution, but the performances are less or more the same. The results are compatible with the results obtained using the fusion and the scores are already calibrated.

Conclusion

The strong similarity between the validation and evaluation sets clearly indicates that the evaluation population has a similar distribution as the training data. The choices and strategies used during the training phase have proven effective when applied to test data.
Using the Gaussian Mixture Model (GMM) full covariance model on z-score normalized data, with 4 components, has demonstrated its effectiveness in generating well-tuned scores in various application ambitions. In our main application ($\tilde{\pi}$ = 0.5), we achieved a remarkably low DCF (Detection Cost Function) cost of 0.06. Furthermore, for the sub-scenarios with $\tilde{\pi}$ values of 0.1 and 0.9, we obtained respectable DCF costs of 0.152 and 0.157, respectively.