MLPR - Gender Identification

Emanuele, Raimondo

Abstract

The goal of the project is the development of the best-fit model for the binary classification of gender from high-level features, representing face images. The analysis will start with the study of the database features and their distributions and if needed, the gaussianization of the features. Then some machine learning algorithms will be deployed and the results applied to the dataset will be analyzed. After developing some considerations, the best candidate models will be selected, and more techniques such as score calibration and score-level fusion will be applied to obtain the best possible classification of the dataset.

Contents

1 Introduction

1.1 Problem Overview

The dataset consists of face image embeddings. Each row represent a different image and contains 12 continuous-valued features, belonging to either the male (label 0) or the female (label 1) class. The features do not have a physical interpretation.

The samples belong to 3 different age groups. Each age group may be characterized by different distributions for the embeddings, but the age information is not available. Both the training set and test set are imbalanced:

	Male	Female
Training set	720	1680
Test set	4200	1800

Table 1.1: Training set and Test set samples

As a first observation, having such differences in the distributions of the Training set and the Test set might generate some discrepancies in the results and might lead to less accurate classifications.

2 Feature Analysis

For a first overview of the data distribution, the raw features have been represented.

We can note from Figure 2.1 that most of the raw features, are not well represented by Gaussian distribution, so we presume that a Gaussian model would perform poorly on this training set. For some of the features, we notice some trends that suggest the presence of at gaussian distribution with three "bells", overlapping, possibly signifying the distributions for the 3 different age groups present in the dataset. We see this trend especially in features 2, 5, 6 and 9. We can also note a strong overlap, especially in features number 2, 4, 6, 7; which can be predictive of these features not being very discriminative for classification, while feature 10 appears to be the most discriminative, with less overlap compared to the others.

No significant outliers were identified, but for completeness, a Z-normalized version of the features was represented, expecting very similar results to the Raw features during validation. Z-normalization produces a new set of data with zero mean and unit variance, by mapping each sample as following:

$$x_i' = \frac{x_i - \mu}{\sigma}, \forall i \in 1, ..., n$$

where x_i = value of the i-th sample, μ = mean vector and σ = standard deviation of the samples.

2.1 Gaussianization

Some of the features already follow a gaussian distribution while others appear to be more disorganized. Gaussianization of the features is believed to be useful for this dataset. In Figure 2.2 we can see that, after Gaussianization, the female class samples follow Gaussian distributions, while this is not properly true for male class samples.

2.2 Pearson correlation

To illustrate the correlation between different features, the absolute value of the Pearson Correlation coefficient is used:

$$\left| \frac{Cov(X,Y)}{\sqrt{Var(X)}\sqrt{Var(Y)}} \right|$$

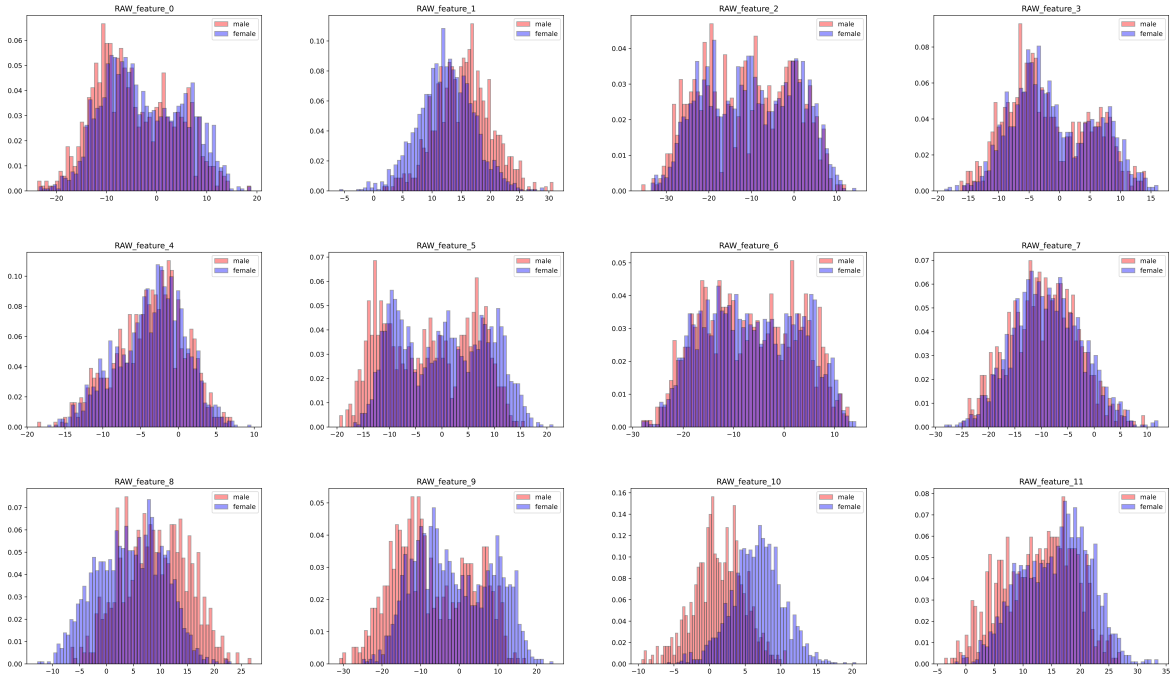


Figure 2.1: Raw features

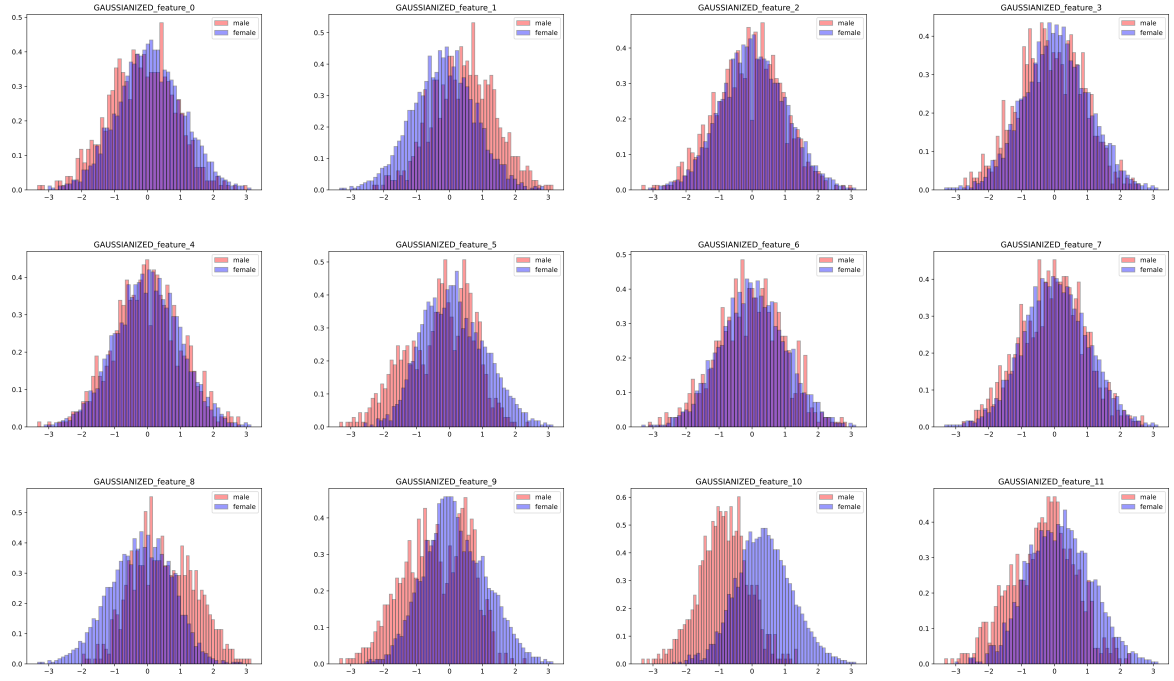
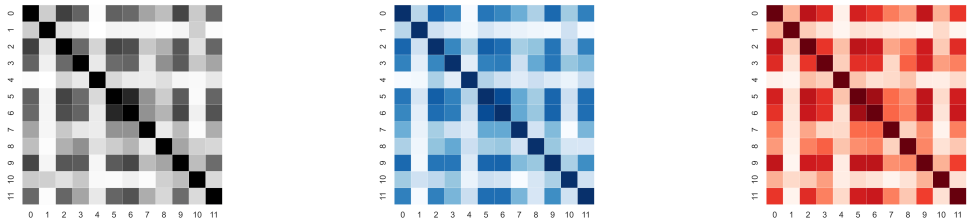


Figure 2.2: Gaussianized features

in order to represent the heatmaps in Figure 2.2.

No relevant correlation between features was found (except for a moderate correlation between features 5 and 6), so it is plausible to assume that PCA (Principal component analysis) will not be relevantly beneficial through the classifications. For completeness, this assumption will be tested.



(a) All classes *(b) Male class* *(c) Female class*

Figure 2.3: Heat-maps

3 Feature classification and validation

In the various classifiers that will be tested, the following choices are made:

- 3 main applications will be verified, a balanced application and two unbalanced ones:
  - $(\pi = 0,5; C_{fn} = 1; C_{fp} = 1)$
  - $(\pi = 0,1; C_{fn} = 1; C_{fp} = 1)$
  - $(\pi = 0,9; C_{fn} = 1; C_{fp} = 1)$
- Classifiers will be tested employing K-Fold cross validation with K = 5. This decision is given from the fact that the dataset contains visibly less samples from the Male class in training set, compared to the Female class samples.
- Classifiers will be run firstly without applying PCA, and then circumstantially applying PCA with m = 11, 10 (m: number of dimensions). Given the previous considerations, we expect a loss of performance as m decreases.
- For initial phase, performance of the classifiers will be measured with the calculation of the Normalized Minimum Detection Cost Function (**minDCF**), in order to select the most convenient approach in terms of "cost to pay" if optimal decisions were taken using the recognizer scores.

3.1 Multivariate Gaussian Classifiers

We are going to focus our attention on Multivariate Gaussian Classifiers (MVG), in particular, those assuming Gaussian distribution of data with the following covariance matrixes:

- Full Covariance Gaussian
- Diagonal Covariance
- Tied Full Covariance
- Tied Diagonal Covariance Gaussian

These are generative models assuming Gaussian distributed data, given the class, as follows:

$$(X|C = c) \sim N(\mu_c, \Sigma_c)$$

Tied MVG assumes that each class is similarly distributed, so, each class has its own mean μ_c as the other MVG models, but same covariance matrix. The diagonal models assume that there are no or minimal correlation between features, so covariance matrices are diagonal matrices.

Since heatmaps in Figure 2.3 have shown moderate correlation spread between features, we expect Diagonal covariance models to perform badly, but since there is no highly correlated features we don't expect significant benefit from PCA.
Moreover, we can notice that heatmaps of Male and Female class are very similar, so we can expect the Tied Model to perform well.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw			Gaussianized			Z-Norm		
	no PCA								
Full Cov	0.113	0.297	0.35	0.139	0.361	0.354	0.113	0.297	0.35
Diag Covariance	0.463	0.771	0.777	0.453	0.756	0.752	0.463	0.771	0.777
Tied Full Cov	0.109	0.299	0.342	0.128	0.35	0.343	0.109	0.299	0.342
Tied Diag Covariance	0.457	0.77	0.781	0.454	0.764	0.766	0.457	0.77	0.781
	PCA (m=11)								
Full Cov	0.117	0.31	0.353	0.142	0.363	0.362	0.119	0.315	0.356
Diag Covariance	0.127	0.318	0.371	0.136	0.398	0.347	0.123	0.298	0.345
Tied Full Cov	0.117	0.294	0.357	0.131	0.358	0.353	0.117	0.299	0.358
Tied Diag Covariance	0.12	0.298	0.364	0.132	0.345	0.359	0.124	0.285	0.349
	PCA (m=10)								
Full Cov	0.161	0.399	0.492	0.188	0.405	0.506	0.184	0.415	0.538
Diag Covariance	0.168	0.443	0.477	0.182	0.41	0.498	0.183	0.407	0.546
Tied Full Cov	0.162	0.389	0.478	0.183	0.428	0.49	0.182	0.429	0.534
Tied Diag Covariance	0.167	0.401	0.473	0.182	0.44	0.507	0.178	0.381	0.544

Table 3.1: MVG Classifiers - minDCF- blue: local best, red: absolute best

As we expected we have best performance in Tied Full covariance models. This suggests that models which exploit linear separation rules, like Linear Logistic Regression, would perform well.
As we expected, Diagonal models are not effective, for the aforementioned reason.
Gaussianization and Z-normalization preprocessing are not very effective. For completeness, in future classifications we will continue testing on gaussianized and znormalized features but the main interest is on the analyses conducted with the raw features.
PCA is not useful, but we can notice that it increases performances in Diagonal Model.

3.2 Logistic Regression Classifiers

Moving on to discriminative models, we will now analyze the results obtained with Linear Logistic Regression and Quadratic Logistic regression.

Since the classes are not at all naturally balanced, they are re-balance for the costs of the different classes minimizing for:

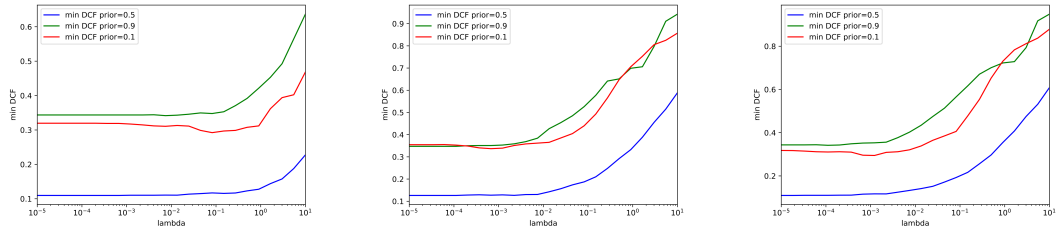
$$J(w,b) = \frac{\lambda}{2}\|w\|^2 + \frac{\pi_T}{T}\sum_{i=1|c_i=1}^{n} \log\left(1 + e^{-z_i\left(w^T x_i + b\right)}\right) + \frac{1-\pi_T}{n_F}\sum_{i=1|c_i=0}^{n} \log\left(1 + e^{-z_i\left(w^T x_i + b\right)}\right) \quad (1)$$

Where (w, b) are the model parameters. For both linear and quadratic logistic regression models $\pi_T = 0.5$ is set for selecting the most appropriate value of the hyperparameter λ.

3.2.1 Linear Logistic Regression

As we already said, since the Tied MVG introduces linear separation rules and has provided good results, it is expected that Logistic Regression will perform well. Furthermore, we do not expect that with 'Gaussianization' or 'Z-score Normalization' we will obtain significantly better results as Logistic Regression does not require specific assumptions on the data distribution.

We tune the regularization hyperparameter λ, to find the best possible value in terms of minDCF. We perform tuning for $\pi_T = 0.5$ and no PCA



(a) Raw *(b) Gaussianized* *(c) Z-normed*

Figure 3.1: Weighted LR tuning

The tuning of the hyper-parameter λ shows that regularization is required and brings benefits especially to pre-processed features with Z-Score Normalization and Gaussianization, with best results obtained for small values of it. Moreover, for $\lambda = 1$, the results significantly starts to get worse, especially for Z-score Normalization and Gaussianization of the features. The scale normalization effects of Z-score and Gaussianization do not seem much relevant, and it is needed a smaller regularization term than Logistic Regression with RAW features. PCA provides almost the same results as the ones without it.

If the regularization parameter has a λ value that is too large, we observe that the model struggles to correctly classify the samples: this is because we reach excessive semplicistic model by selecting smaller values for $\mathbf{w}$. As a result, model fails to capture the complexities and nuances of the data, leading to poor classification performance. On the contrary, for small values of λ, the model tends to achieve a good separation by selecting higher values for $\mathbf{w}$, the result is a model that is performing on training data but generalizes worse to unseen data, in respect to a model with higher values of λ.

To find a balance between overfitting and underfitting, we choose $\lambda = 10^{-4}$ in order to mitigate the risk of overfitting the model to the training data and maximize its ability to generalize to new, unseen data.

We now turn the attention on the tabular values for different π_T in Table 3.2

We can conclude that MVG with Tied Covariance Matrix performs similarly. It is reasonable to affirm that Logistic Regression retrieve the same outcomes of the MVG model with linear classification rules. Moreover, we can see that $\pi_T = 0.1$ slightly increases performance in application $\pi = 0.1$, while this is not the case for $\pi_T = 0.9$ which does not bring improvements for the other application $\pi = 0.9$. This is because our training set is unbalanced: when we set $\pi_T = 0.1$ we associate a lower cost to mispredictions over male class which is the class that has lower samples and then it is more proned to errors; while we associate a $\pi_{female} = 1 - \pi_T = 0.9$ to female class which is characterized by more samples and then it's less prone to mispredictions.
Note that, since Z-Score is a linear transformation, the performance obtained on the raw data and the ZScored data are the same.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw			Gaussianized			Z-Norm		
	no PCA								
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.5$)	0.11	0.32	0.344	0.126	0.355	0.347	0.11	0.318	0.344
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.1$)	0.113	0.289	0.353	0.127	0.343	0.354	0.113	0.288	0.352
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.9$)	0.119	0.352	0.351	0.127	0.358	0.354	0.12	0.351	0.352
	PCA (m=11)								
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.5$)	0.114	0.31	0.357	0.128	0.36	0.358	0.115	0.313	0.355
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.1$)	0.121	0.289	0.361	0.132	0.354	0.369	0.119	0.299	0.362
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.9$)	0.122	0.346	0.365	0.126	0.364	0.359	0.119	0.339	0.361

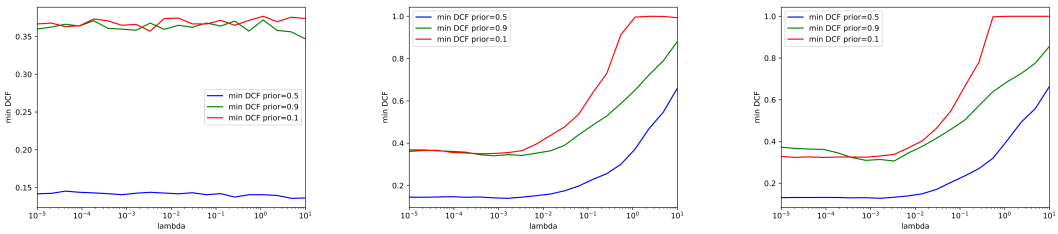
Table 3.2: Weighted Logistic Regression- minDCF

3.2.2 Quadratic Logistic Regression

We now implement quadratic logistic regression, which projects the data into a higher-dimensional 2D space (expanded feature space). In some cases, a linear separation rule might not be able to properly classify the data, quadratic logistic regression supplies to this. Quadratic LR performs a linear separation in this space which corresponds to a quadratic separation rule in the original space.

It is not expected for regularization to have a significant impact on the minDCF computation due to the higher complexity of the model.

We performed regularization hyperparameter λ tuning also for this model, with $\pi_T = 0.5$ and no PCA:



(a) Raw *(b) Gaussianized* *(c) Z-normed*

Figure 3.2: Quadratic LR tuning

Regularization is not so helpful for RAW features, but it is for pre-processed features with Z-Score Normalization and Gaussianization as in the previous model. For the same considerations of Weighted LR, we find a compromise between overfitting and underfitting in the value of $\lambda = 10^{-2}$ for all Raw, Gaussianized and Z-Score features.
Again, let's consider training with different values of π_T in Table 3.3.

This model gets worse results than the previous ones: as expected, quadratic separation rule seem to suffer from overfitting and are less performing than linear separation rules. Differently from the Weighted logistic Regression, here, Z-normalization produces different minDCF from raw data: this is because, in Quadratic Logistic Regression, we work in an expanded feeature space, so, also a linear transformation as Z-normalization changes the model.
Pre-processed dataset with PCA m = 11, has the same outcomes as the Logistic Regression. Also in this case, using different values of π_T slightly improves the classification performance when $\pi_T = 1$ for the corresponding application.

	$\pi=0.5$	$\pi=0.1$	$\pi=0.9$	$\pi=0.5$	$\pi=0.1$	$\pi=0.9$	$\pi=0.5$	$\pi=0.1$	$\pi=0.9$
	Raw			Gaussianized			Z-Norm		
	no PCA								
Quad LogReg($\lambda=10^{-2}$, $\pi_T=0.5$)	0.142	0.367	0.36	0.145	0.361	0.368	0.131	0.328	0.372
Quad LogReg($\lambda=10^{-2}$, $\pi_T=0.1$)	0.135	0.332	0.355	0.14	0.368	0.365	0.133	0.327	0.365
Quad LogReg($\lambda=10^{-2}$, $\pi_T=0.9$)	0.166	0.412	0.431	0.155	0.36	0.412	0.141	0.349	0.399
	PCA (m=11)								
Quad LogReg($\lambda=10^{-2}$, $\pi_T=0.5$)	0.114	0.31	0.357	0.128	0.36	0.358	0.115	0.313	0.355
Quad LogReg($\lambda=10^{-2}$, $\pi_T=0.1$)	0.121	0.289	0.361	0.132	0.354	0.369	0.119	0.299	0.362
Quad LogReg($\lambda=10^{-2}$, $\pi_T=0.9$)	0.122	0.346	0.365	0.126	0.364	0.359	0.119	0.339	0.361

Table 3.3: Quadratic Logistic Regression- minDCF

3.2.3 Considerations

We can conclude that classes are better separated with linear decision rules. The study for this application continues with the training of SVMs and to Gaussian-Mixture Models, and we start with the first one for which we will expect good result in its linear form. To confirm our previous assumptions, we continue with the pre-processing techniques analyzed in the previous models (Z-score Normalization and Gaussianization)

3.3 Support Vector Machines

To continue testing the classifiers, we will now test three different SVM models:

- Linear SVM
- Polynomial quadratic kernel SVM
- Radial Basis Function kernel SVM

The dual formulation of the SVM problem is a maximization problem w.r.t. α:

$$J^D(\boldsymbol{\alpha}) = -\frac{1}{2}\boldsymbol{\alpha}^T\boldsymbol{H}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathbf{1} \qquad \text{subject to} \quad 0 \leq \alpha_i \leq C, \forall i \in \{1,\dots,n\}, \sum_{i=1}^{n} \alpha_i z_i = 0$$

Where n is the number of training samples, C is a hyper-parameter, 1 is a n-dimensional vector of ones, z_i is the class label for the i-th sample encoded as

$$z_i = \begin{cases} +1, & \text{if } x_i \text{ belongs to class 1 (Female)} \\ -1, & \text{if } x_i \text{ belongs to class 0 (Male)} \end{cases}$$

and H is the matrix:

$$\boldsymbol{H}_{ij} = z_i z_j \boldsymbol{x}_i^T \boldsymbol{x}_j$$

Due to the class imbalance, the SVM implementation considered class balancing. Balancing is done by considering a different value of C for the different classes in the box constraint of the dual formulation:

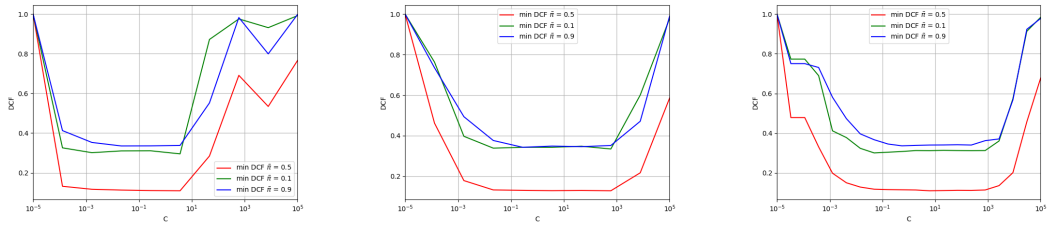
$$0 \leq \alpha_i \leq C_i, i = 1,\dots,n$$

where $C_i = C_F$ for samples in the Female class and $C_i = C_M$ for the ones in the Male class.
We choose $C_F = C\frac{\pi_T}{\pi_T^{emp}}$ and $C_M = C\frac{\pi_F}{\pi_F^{emp}}$, π_T^{emp} and π_F^{emp} being the empirical priors for the classes computed over the training set.

We employed 5-fold cross validation for choosing the best value of C in the 3 models proposed, without using class rebalancing.

3.3.1 Linear SVM

We frst select the hyperparameter C:



(a) Raw *(b) Gaussianized* *(c) Z-normed*

Figure 3.3: Linear SVM C tuning

Considering the graphs, the best value to select is $C = 10^{-1}$ as it provides the lowest values of DCF providing a stable margin, especially for the Raw features, that is our main focus, having obtained better results applying the classifiers for the raw features in the previous sections.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw			Gaussianized			Z-Norm		
	no PCA								
Linear SVM ($K = 1$, $C = 0.1$, $\pi_T = unb$)	0.113	0.315	0.337	0.132	0.336	0.351	0.117	0.303	0.351
Linear SVM ($K = 1$, $C = 0.1$, $\pi_T = 0.5$)	0.112	0.317	0.337	0.128	0.336	0.348	0.12	0.29	0.35

Table 3.4: Linear Support vector machine- minDCF

Linear SVM, as we can see from the table above, performs very similarly to Linear LogReg and MVG Tied covariance, which are our best candidates until now. This confirms the previously made assumptions that applications with linear decision rules work well in our dataset. Moreover, the results obtained with class balancing are very similar to the ones obtained in the unbalanced case, not bringing any important benefit, if not in the case of $\pi = 0.1$ where we have slightly better results. This might be due to the unbalance of the training dataset in having significantly more samples for the Female class.

3.3.2 Quadratic kernel SVM

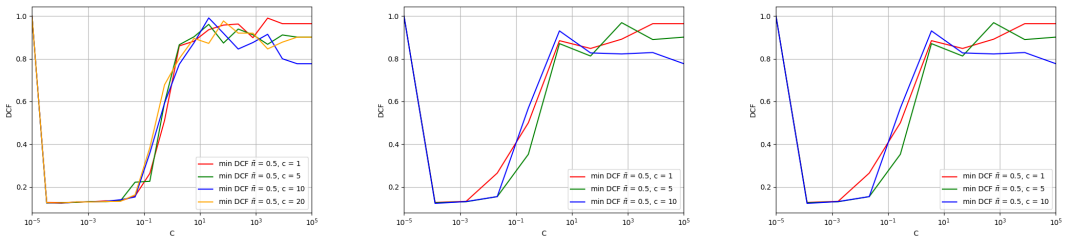
After linear SVM, we move on to the non-linear SVM models. The non-linearity can be obtained via an explicit expansion of the feature space into a higher dimensional space as in Quadratic Logistic Regression, which is very computationally expensive. A much better option is to exploit the dual formulation of the SVM problem that allows us to work with only the dot product in the expanded space. This is achieved by using the so called *kernel function*:

$$k(\hat{x}_i, \hat{x}_j) = \phi(\hat{x}_i)^T \phi(\hat{x_j})$$

In this first case we will make use of the polynomial kernel function, defined as:

$$k\left(\hat{x}_i, \hat{x}_j\right) = \left(\hat{x}_i^T \hat{x_j} + c\right)^d$$

Also in this case, the tuning of the hyper parameter C is needed. As opposed to the previous case, the C hyper-parameter is dependent on the parameter c and d (degree set to 2). Thus, a joint optimization of C and c is performed:



(a) Raw *(b) Gaussianized* *(c) Z-normed*

Figure 3.4: Quadratic Kernel SVM C and c tuning

The parameters selected are: $C = 10^{-3}$ and $c = 5$ that result in the lowest and most stable values for DCF, as we can evince looking at the graphs in figures **??**

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw			Gaussianized			Z-Norm		
	no PCA								
Quadratic SVM ($K = 1$, $C = 10^{-3}$, $\pi_T = unb$)	0.115	0.303	0.355	0.2	0.451	0.559	0.206	0.449	0.617
Quadratic SVM ($K = 1$, $C = 10^{-3}$, $\pi_T = 0.5$)	0.118	0.298	0.347	0.204	0.48	0.576	0.214	0.465	0.618

Table 3.5: Quadratic kernel Support vector machine- minDCF

From the table 3.5 we see that surprisingly, Quadratic SVM performs similarly to Linear SVM, for this it is believed that kernel functions might also perform adequately, and we expect to obtain similar results with the use of a SVM classifier with Radial Basis kernel.
In general, linear classification models are still preferred for this analyzed dataset due to their simplicity and similar performances to polinomial and kernel functions.
We already decided to focus our attention on the classification with raw features but until now chose to implement Gaussianization and znormalization for a more complete picture of the results. In the following models, since the gaussianization and znorm of features are not bringing any benefits at all, we will not consider them anymore.

3.3.3 Radial Basis Function kernel SVM

We tune the hyper-parameter C jointly with the parameter γ :

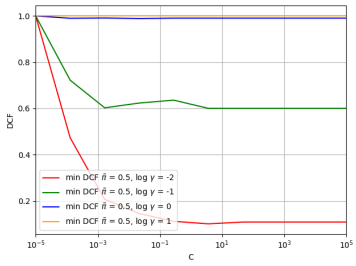


Figure 3.5: RBF C Tuning: No PCA

The parameters selected are $C = 10^1$ and $\gamma = 10^{-2}$

The minDCF values in the table above, for the raw features, suggest that the RBF kernel SVM works well and the values obtained are very similar to the Linear and quadratic SVM previously analyzed. For the testing of this classifier balancing for $\pi_T = 0.5$, for the hypermarameters chosen, yeld results very similar to class unbalancing. Overall, since RBF SVM yeld results that are slightly better than Linear SVM, we choose to consider it as a good candidate for our model.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw		
	no PCA		
RBF SVM ($K = 1$, $C = 10$, $\pi_T = unb$)	0.109	0.31	0.319
RBF SVM ($K = 1$, $C = 10$, $\pi_T = 0.5$)	0.109	0.31	0.319

Table 3.6: RBF kernel Support vector machine- minDCF

3.4 Gaussian Mixture Models

At last, we will consider Gaussian Mixture models (GMM), generative models based on the approximation of generic distributions by "fitting" MVG distributions and associating each sample to one of the "clusters" defined by each of the MVG components. These are obtained by iteratively increasing the number of Gaussian components by performing splits.
Using the 5-fold approach, we will take into consideration:

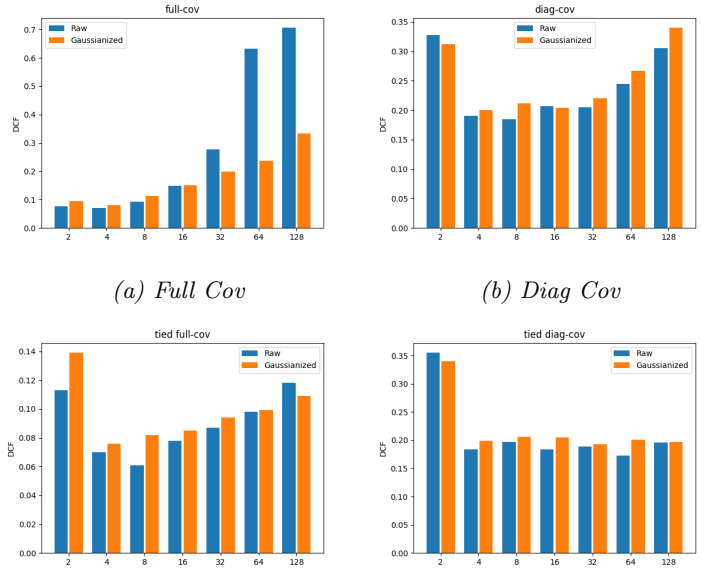
- Full covariance model
- Diagonal covariance model
- Tied Full covariance model
- Tied Diagonal covariance model

Since, until now, pre-processed data with Gaussianization and Z-norm haven't performed well, we will focus on RAW data features and keep Gaussianized data features just to confirm the expectations. As already done for SVM classifiers, e will also leave PCA since it's been bad performing in every previous model. For the Tied models, the covariance tying is done at class level, meaning that different classes (True and False) have different covariance matrices, while each of the MVG components inside a class share the same covariance matrix.

Components	2	4	8	16	32	64	128
				Raw			
Full-Cov	0.077	0.071	0.093	0.148	0.278	0.633	0.706
Diag-Cov	0.328	0.19	0.185	0.207	0.205	0.245	0.305
Tied Full-Cov	0.113	0.07	0.061	0.078	0.087	0.098	0.118
Tied Diag-Cov	0.355	0.184	0.197	0.184	0.189	0.173	0.196
				Gaussianized			
Full-Cov	0.095	0.08	0.113	0.15	0.199	0.238	0.333
Diag-Cov	0.312	0.2	0.212	0.204	0.22	0.267	0.34
Tied Full-Cov	0.139	0.076	0.082	0.085	0.094	0.099	0.109
Tied Diag-Cov	0.34	0.199	0.206	0.205	0.193	0.201	0.197

Table 3.7: minDCF for different numbers of components, trained on $\pi = 0.5$ with $pi_T = 0.5$

We can observe that the best result that it could be obtained is for full-covariance GMM, especially the one with 2, 4 and 8 components on RAW data features.
We believe that these results are due to the fact that the dataset contains samples for three different age groups that influence how certain features are distributed. In particular, some features present 3 different gaussian curves that overlap each other but still appear quite separated. For this, we choose to select as our best candidate GMM with 4 components instead of GMM with 8 components. We believe that GMM with 4 components is able in general to obtain better results because it is able to better divide



(a) Full Cov *(b) Diag Cov*

(c) Tied Full Cov *(d) Tied Diag Cov*

Figure 3.6: minDCF for different numbers of components

the 3 different age groups and classify the binary problem also according to these three age groups present.

Following this logic, GMM with 3 components should provide even more accurate results being able to capture the characteristics of the 3 age groups better, but usually GMM is trained with a number of clusters growing in power of 2, mainly for practical reasons and computational efficiency in the training, so we choose to not explore this possibility in the context of this report.

We report now the minDCF values for GMM with 4 components Tied Full Covariance for different applications, trined with $pi = 0.5$ on RAW data.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw		
GMM-4c-Tied Full Cov.($\pi_T = 0.5$)	0.07	0.272	0.224

Table 3.8: GMM-4c-Tied Full Cov

3.5 Comparisons and Score Calibration

3.5.1 Best Classifiers found

We compare now the models exploiting Bayes Error Plots. We selected the 4 best performing models trained on RAW data features:

We have discussed about **minDCF** metric and costs for the validation set using scores of the recognizer. The **minDCF** is an optimistic metric that assumes the selection of the optimal threshold for performing class assignment. However, the cost that we actually pay depends on the actual threshold used for performing class assignment, which is the theoretical threshold:

$$t = -\log \frac{\tilde{\pi}}{1-\tilde{\pi}}$$

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw	
GMM- 4 comp - Tied Full-Cov ($\pi_T = 0.5$)	0.07	0.221	0.206
RBF SVM ($K = 1$, $C = 10$, $\gamma = 10^{-2}$, $\pi_T = 0.5$)	0.109	0.31	0.319
MVG Tied Full Cov	0.109	0.299	0.342
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.5$)	0.11	0.32	0.344

Table 3.9: Best classifiers identified

Due to the possible miscalibration, the theoretical threshold usually does not correspond to the optimal one. For this reason, we introduce the **actualDCF** metric, allowing us to measure the precision of the actual decisions that will be made by the recognizer.
We decided to follow an approach to re-calibrate the scores, transforming them such that the theoretical threshold provides close to optimal values over a wide range of effective priors $\tilde{\pi}$.
For this approach, we need a transformation function f that maps the classifiers scores into well-calibrated scores $s_{cal} = f(s)$.
We will focus on the Discriminative score models approach, in fact our function f will be an affine function:

$$f(s) = \alpha s + \beta$$

$f(s)$ can thus be interpreted as the log-likelihood ratio for the two class hypotheses:

$$f(s) = \log \frac{f_{S|C}\left(s \mid H_T\right)}{f_{S|C}\left(s \mid H_F\right)} = \alpha s + \beta$$

The class posterior for $\tilde{\pi}$ corresponds to:

$$\log \frac{P(C = H_T \mid s)}{P(C = H_F \mid s)} = \alpha s + \beta + \log \frac{\tilde{\pi}}{1 - \tilde{\pi}}$$

Interpreting scores as features, we have a quite similar expression as the log posterior ratio of the logistic regression model. In fact, if we let

$$\beta' = \beta + \log \frac{\tilde{\pi}}{1 - \tilde{\pi}}$$

we have the exact same model. This, can be employed as a model parameter over our training scores. To recover the calibrated score f(s) we will need then:

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

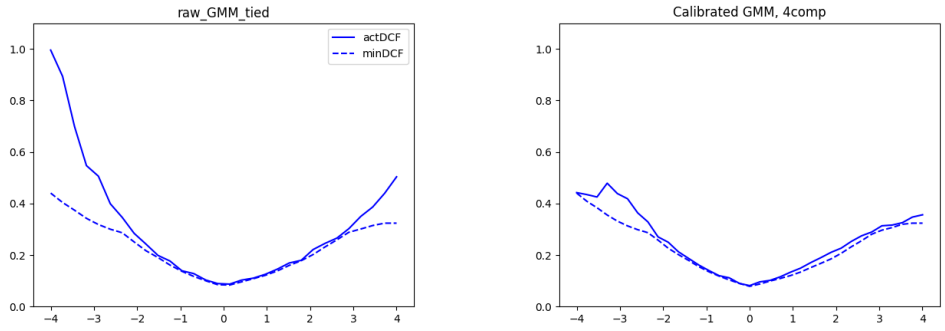
For the calibration, we don't estimate the calibration parameters over the whole set of scores, instead we partition the scores that we want to calibrate into 70% to be used to estimate the calibration parameters and 30% for actual calibration.

This is our best performing model. We can see that calibration was not very necessary, but slightly increases performance in application with $\pi = 0.1$

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
act_DCF_uncalibrated	0.076	0.25	0.22
act_DCF_calibrated	0.076	0.245	0.216
min_DCF	0.07	0.221	0.206

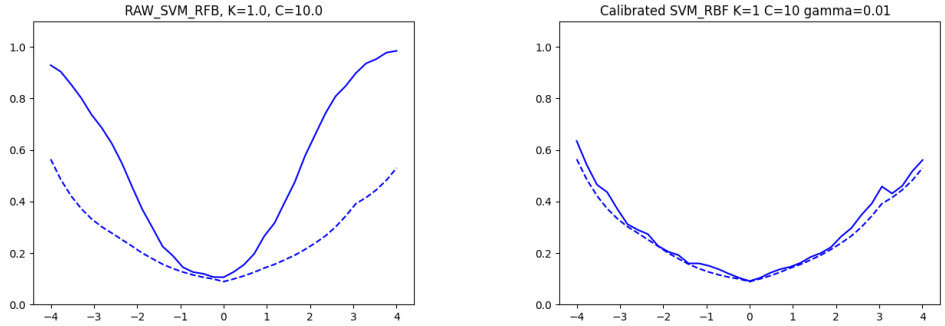
Table 3.10: act_DCF and min_DCF for GMM model

In RBF SVM, wee see that calibration increases perfomances over a much wider range of applications: in fact in $pi = 0.1$ and $pi = 0.9$ applications we have much better results. This is, as expected, because SVM returns non-probabilistic results that need calibration.



(a) Uncalibrated *(b) Calibrated*

Figure 3.7: GMM 4components- Tied Full Cov. - RAW - $\pi_T = 0.5$

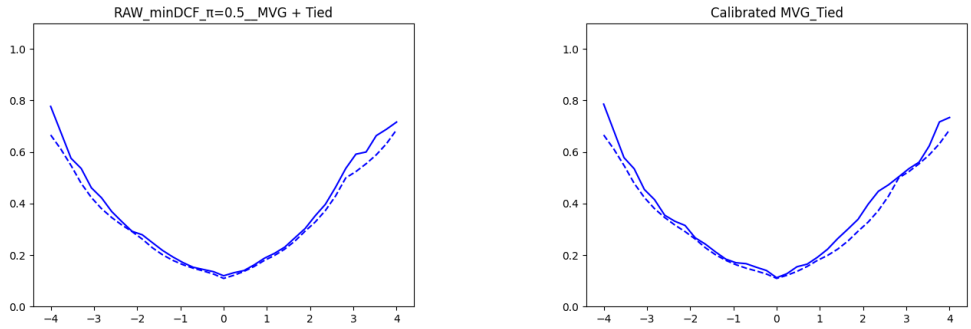


(a) Uncalibrated *(b) Calibrated*

Figure 3.8: Radial Basis SVM - C=10, K=1, $\gamma = 10^{-2}$- RAW - $\pi_T = 0.5$

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
act_DCF_uncalibrated	0.134	0.652	0.843
act_DCF_calibrated	0.109	0.325	0.333
min_DCF	0.109	0.31	0.319

Table 3.11: act_DCF and min_DCF for SVM model

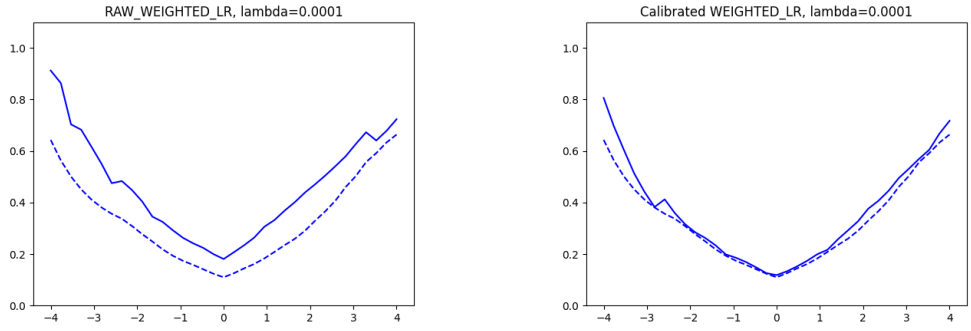


(a) Uncalibrated *(b) Calibrated*

Figure 3.9: MVG Tied Full Cov.- RAW - $\pi_T = 0.5$

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
act_DCF_uncalibrated	0.11	0.299	0.343
act_DCF_calibrated	0.109	0.299	0.362
min_DCF	0.109	0.299	0.342

Table 3.12: act_ DCF and min_ DCF for MVG model



(a) Uncalibrated *(b) Calibrated*

Figure 3.10: Weighted LR- $\lambda = 10^{-4}$- RAW - $\pi_T = 0.5$

Exactly as in GMM model, in this case MVG Tied doesn't need calibration.
In fact we can see a slightly loss in performance due to excessive calibration.
For Weighted Logistic Regression, calibration is needed.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
act_DCF_uncalibrated	0.23	0.457	0.382
act_DCF_calibrated	0.11	0.343	0.35
min_DCF	0.11	0.32	0.344

Table 3.13: act_ DCF and min_ DCF for W-LR model

4 Experimental Results

In this section we repeat all the conducted analyses on the evaluation set to study all the models previously taken into consideration, to confirm or discredit the results observed while applying the 5-fold cross-validation.
We will repeat the calculations in terms of minDCF as well as the hyperparameters tunings.

4.1 MVG classifiers

4.1.1 Results

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw			Gaussianized			Z-Norm		
	no PCA								
Full Cov	0.12	0.312	0.312	0.14	0.347	0.341	0.12	0.312	0.312
Diag Covariance	0.434	0.82	0.705	0.423	0.77	0.685	0.434	0.818	0.705
Tied Full Cov	0.116	0.301	0.308	0.134	0.338	0.318	0.116	0.301	0.308
Tied Diag Covariance	0.435	0.818	0.711	0.426	0.799	0.694	0.435	0.815	0.711
	PCA (m=11)								
Full Cov	0.123	0.322	0.333	0.138	0.353	0.338	0.126	0.325	0.332
Diag Covariance	0.13	0.356	0.324	0.14	0.36	0.319	0.123	0.323	0.339
Tied Full Cov	0.122	0.315	0.323	0.134	0.353	0.319	0.124	0.314	0.313
Tied Diag Covariance	0.124	0.327	0.32	0.136	0.35	0.346	0.12	0.31	0.339
	PCA (m=10)								
Full Cov	0.154	0.409	0.428	0.181	0.43	0.429	0.179	0.433	0.412
Diag Covariance	0.16	0.434	0.423	0.174	0.435	0.415	0.175	0.417	0.429
Tied Full Cov	0.15	0.403	0.419	0.17	0.421	0.4	0.172	0.41	0.405
Tied Diag Covariance	0.156	0.405	0.423	0.173	0.434	0.418	0.174	0.417	0.421

Table 4.1: MVG Classifiers - minDCF- test set

From the table above we see how the results using the evaluation set appear to be in line with the previously conducted analyses. The Full covariance and the Tied Full covariance MVG classifiers have very similar results between them, much more affine than the ones obtained while working on the training set only.
As we noticed before, PCA does not seem to be very useful and also the gaussianization and znormalization of the features does not bring benefits, once again. With these results we confirm that the best MVG classifiers have been identified correctly during the first phase of our study.

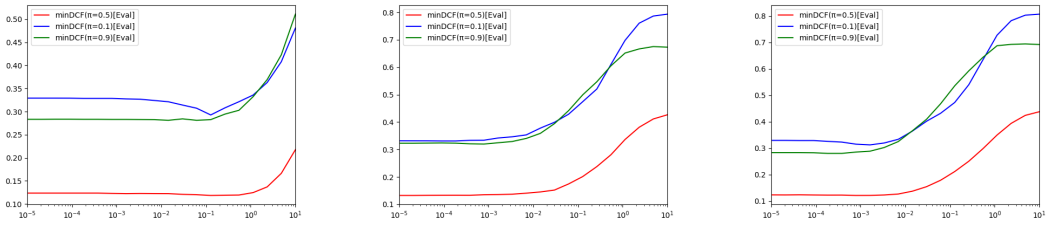
4.2 Logistic Regression classifiers

4.2.1 Linear logistic regression

We tune the hyperparameter λ:

The hyperparameter λ chosen is once again $\lambda = 10^{-4}$

Also in this case the results are aligned with the ones found before with the use of the training set, especially for the application point $\pi = 0.5$, while for the other two unbalanced application points we can see that they seem to have better results than the ones previously found. This could be because of the different distribution of the number of male samples and female samples, since in the training set we have significantly more female samples, while now there are more male samples, so the results for $\pi = 0.1$ and $\pi = 0.9$ application points might differ from the previous results for this reason.



(a) Raw *(b) Gaussianized* *(c) Z-normed*

Figure 4.1: Weighted LR tuning

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw			Gaussianized			Z-Norm		
	no PCA								
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.5$)	0.124	0.329	0.283	0.132	0.332	0.323	0.123	0.329	0.283
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.1$)	0.119	0.301	0.31	0.133	0.327	0.327	0.119	0.301	0.31
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.9$)	0.132	0.385	0.282	0.136	0.351	0.302	0.131	0.384	0.281
	PCA (m=11)								
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.5$)	0.128	0.339	0.311	0.135	0.348	0.331	0.127	0.343	0.292
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.1$)	0.124	0.308	0.323	0.135	0.341	0.34	0.123	0.31	0.316
LogReg($\lambda = 10^{-4}$, $\pi_T = 0.9$)	0.132	0.392	0.309	0.139	0.373	0.326	0.133	0.393	0.31

Table 4.2: Weighted Logistic Regression- minDCF

This will possibly be visible for all the classifiers that will be analyzed in this experimental phase.

Following the poor results obtained by quadratic Logistic regression, we choose to not implement it in the experimental phase since it was not a good candidate for our model and it was excluded.

4.3 Support Vector Machines

We will analyze all the SVM classification together to better understand their trends.

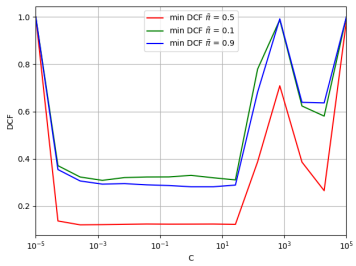


Figure 4.2: Linear SVM C tuning: raw features

Once again the choice of $C = 10^{-1}$ provides the more accurate results for Linear SVM

For the Quadratic kernel SVM, we can choose again $C = 10^{-3}$ and c=5 providing good results

For RBF SVM, the parameters selected are again $C = 10^{1}$ and $\gamma = 10^{-2}$

Considering these results, we see that quadratic SVM performs slightly better than the other SVM types, but the previously chosen best model from this table (RBF SVM) still performs well and it is

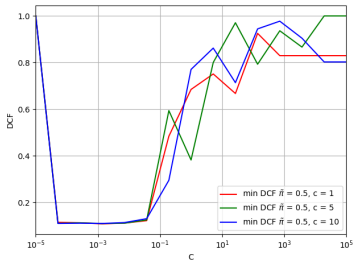


Figure 4.3: Quadratic kernel C and c Tuning: Raw features

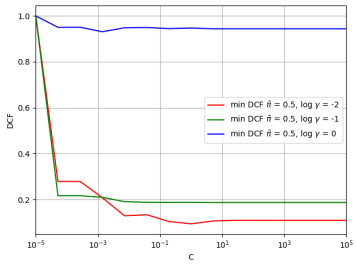


Figure 4.4: RBF C and γ Tuning: Raw features

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
Linear SVM ($K = 1$, $C = 0.1$, $\pi_T = unb$)	0.123	0.322	0.288
Linear SVM ($K = 1$, $C = 0.1$, $\pi_T = 0.5$)	0.119	0.308	0.294
Quadratic SVM ($K = 1$, $C = 10^{-3}$, $\pi_T = unb$)	0.109	0.287	0.287
Quadratic SVM ($K = 1$, $C = 10^{-3}$, $\pi_T = 0.5$)	0.11	0.288	0.316
RBF SVM ($K = 1$, $C = 10$, $\pi_T = unb$)	0.11	0.362	0.255
RBF SVM ($K = 1$, $C = 10$, $\pi_T = 0.5$)	0.11	0.362	0.255

Table 4.3: SVM- minDCF - raw features

confirmed as an adequate choice to be considered between the best classifiers.

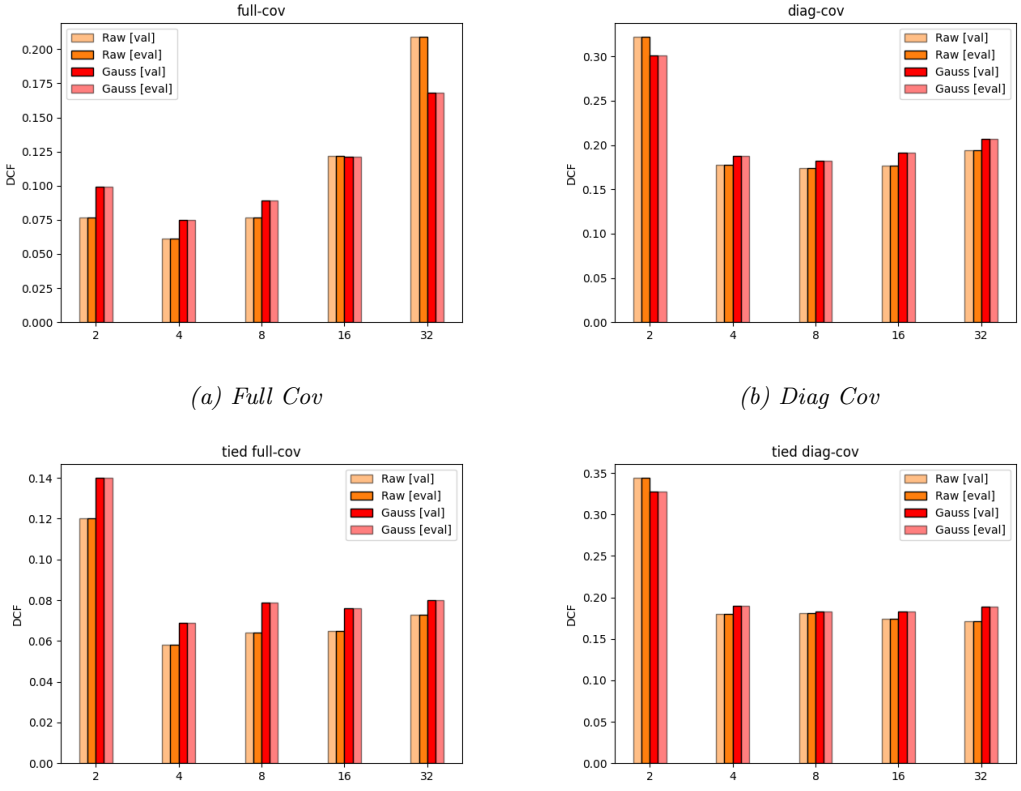
4.4 Gaussian Mixture Models

In these analyses we see similar trends to the ones we observed in the training phase. The previous best selected model, which is GMM tied full cov, is still the most performing ones and the order of performance is very similar to the ones observed in the previous phase.

Components	2	4	8	16	32
Full-Cov	0.077	0.061	0.077	0.122	0.209
Diag-Cov	0.322	0.178	0.174	0.177	0.194
Tied Full-Cov	0.12	0.058	0.064	0.065	0.073
Tied Diag-Cov	0.344	0.18	0.181	0.174	0.171

Table 4.4: minDCF for different numbers of components, $\pi = 0.5$

The tied full covariance GMM model for 4 splits still provides the best results and our choice for this as the best model for our classification is confirmed as it is proven to be the most effective.



(a) Full Cov

(b) Diag Cov

(c) Tied Full Cov

(d) Tied Diag Cov

Figure 4.5: minDCF for different numbers of components

4.5 Comparison and Score Calibration

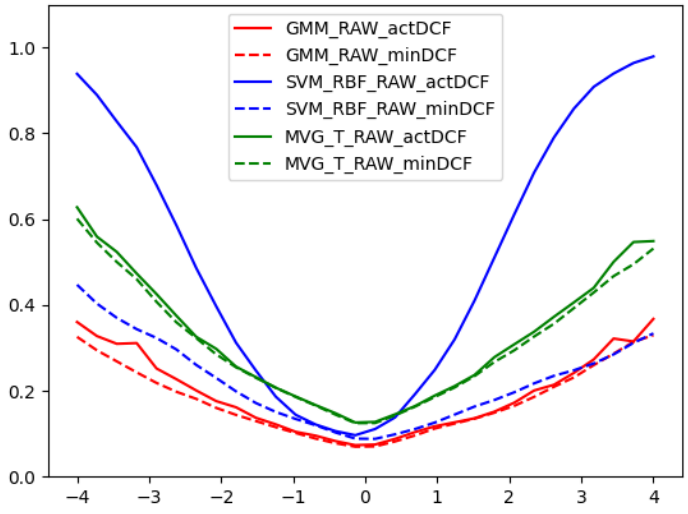
As in validation phase, we now compare the 3 best performing models trained on RAW data features, with $\pi_T = 0.5$:

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	Raw		
GMM- 4 comp - Tied Full-Cov ($\pi_T = 0.5$)	0.058	0.151	0.156
RBF SVM ($K = 1$, $C = 10$, $\gamma = 10^{-2}$, $\pi_T = 0.5$)	0.11	0.362	0.255
MVG Tied Full Cov ($\pi_T = 0.5$)	0.116	0.301	0.308

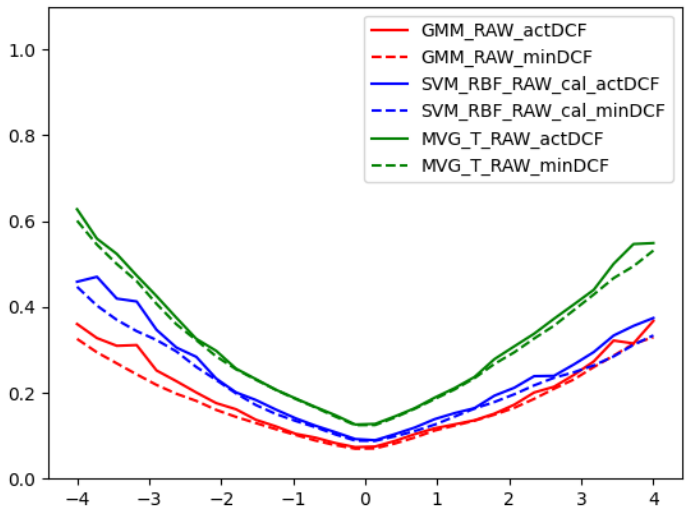
Table 4.5: Best classifiers in evaluation phase

As we plot the Bayes error curves for GMM-4comp-Tied and MVG-Tied we notice, exactly as happened for validation set, that the curves are already calibrated. So we decide to not apply further calibration. Instead, as expected, for RBF-SVM calibration is needed. We show a comparison in a single figure.

Exactly as in validation we see that GMM model outperformes the other models: this is assumed to be, again, because Mixture Gaussian Model better describes the three bells of samples belonging to three different age groups.



(a) Uncalibrated RBF-SVM



(b) Calibrated RBF-SVM

Figure 4.6: GMM-4comp-Tied vs MVG-Tied vs

	minDCF			actDCF		
	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
GMM- 4 comp - Tied Full-Cov ($\pi_T = 0.5$)	0.058	0.151	0.156	0.06	0.155	0.156
RBF SVM ($K = 1$, $C = 10$, $\gamma = 10^{-2}$, $\pi_T = 0.5$)	0.11	0.362	0.255	0.149	0.553	0.863
RBF SVM_calibrated($K = 1$, $C = 10$, $\gamma = 10^{-2}$, $\pi_T = 0.5$)	0.11	0.362	0.255	0.11	0.387	0.283
MVG Tied Full Cov ($\pi_T = 0.5$)	0.116	0.301	0.308	0.116	0.302	0.312

Table 4.6: act_DCF and min_DCF

5 Conclusion

Since the results of the training and the testing phase are consistent, we can assume that the distribution of the training and test dataset are similar. The best model that was selected during the training phase also was the best performing model in the evaluation, confirming our choice of selecting the GMM tied covariance with 4 components, raw features and no PCA. Especially considering our main application point ($\pi = 0.5$), we achieved a very low DCF of 0.058.