Language Detection Task

Cristiano Serra 305960

Abstract. The aim of this report is to explore the performance of different classifiers on a binary classification task. The dataset used is the Language Detection. The purpose of this project is to analyze the dataset given, and to compare the various models introduced during the course.

1 Dataset Information

1.1 Dataset Description

The dataset contains a set of 6 different features, obtained by mapping audio segments to a duration-independent, low-dimensional manifold. The features do not have any physical interpretation. Each set of data has a label, either 1 or 0 (binary classification task), 1 if the language is the same as the target one (for example Italian) and 0 if not.

1.2 Features

As said before, the features do not have any physical interpretation. Given that fact we will refer to them as feature 1 to 6.

Input variables

1 – Feature 1

2 – Feature 2

3 – Feature 3

4 – Feature 4

5 – Feature 5

6 – Feature 6

Output variable

Labels 0 or 1

1.2 Feature Distribution

Before getting to model's analysis, let's look at the feature distribution. In this case histogram plots and a correlation heatmap have been used to show the raw data distribution and the correlation between different features, for the whole dataset and both for the data related to the label 0 and label 1.

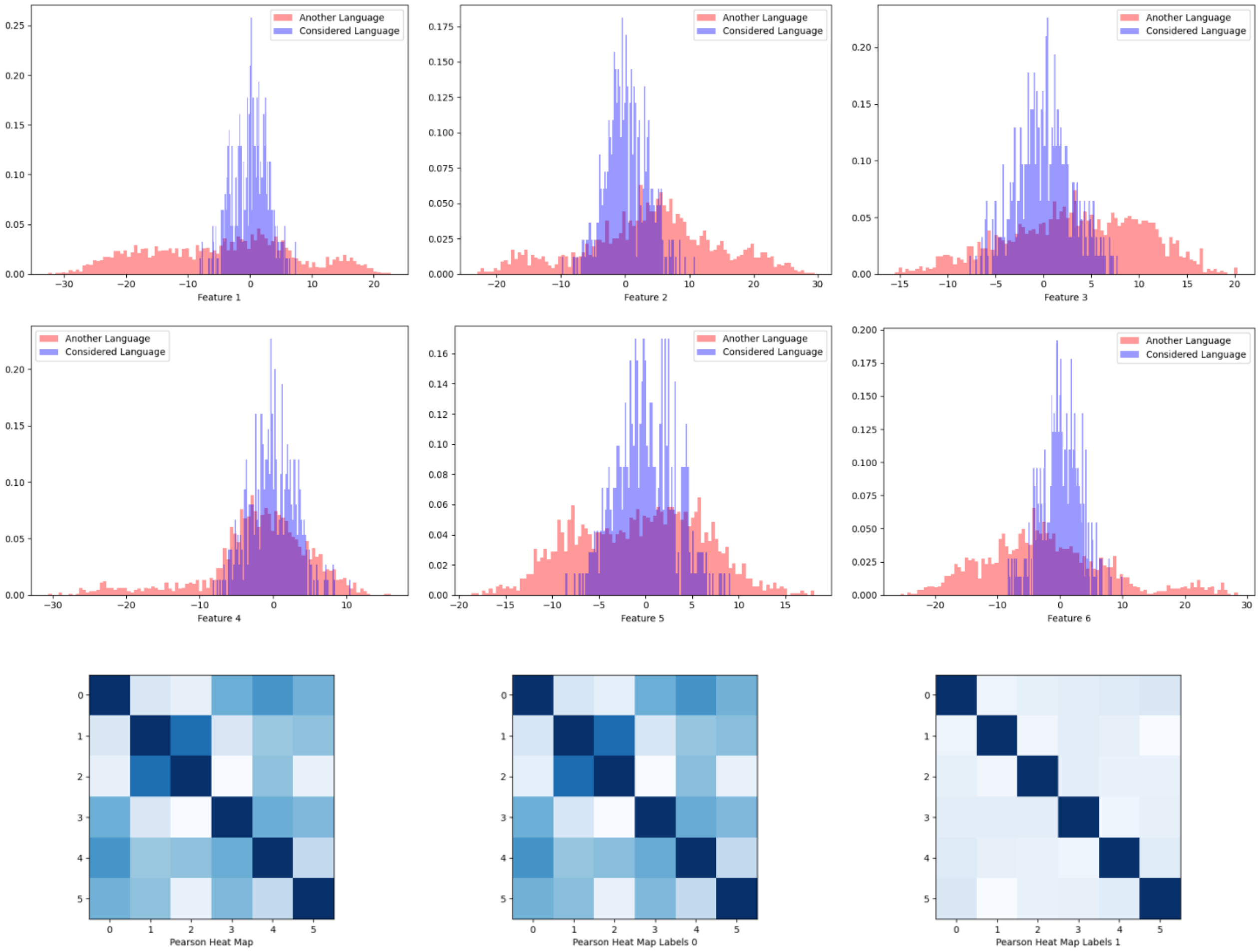

Figure 1. Raw feature distribution and correlation heat maps

1.3 Dataset Analysis and Preliminary assumptions

The training dataset consist of a total of 2371 samples.

The classes are not balanced, with class zero having 1971 samples and class zero having only 400 samples.

We will consider two main application, one with (π, Cfn, Cfp) = (0.5, 1, 1) and one with (π, Cfn, Cfp) = (0.1, 1, 1) .

As can be seen from the correlation heat maps, it seems that a couple of features are well correlated, in this regard trying to reduce the number of features using a PCA approach can be useful. Another thing that can be observed is that the class one heatmap is quite different from the class zero's one, while the latter is almost identical to the whole dataset, the former doesn't seem to have any significant correlation between different features.

Regarding the training phase, while using a single split approach would've improved computational time, using k fold leads to more robust estimation. So, in this project a k fold approach has been chosen, with K = 3, with a random shuffle before computing the folds.

To compare model normalized minimum detection (MinDCF) has been used.

2 Preliminary considerations and Data Pre-Processing

The first pre-processing step is to normalize the dataset, this has been done using Gaussianization. Where we use the training part of each fold to compute a ranking used to normalize the test part. The scope of Gaussianization is to map the dataset from its original domain into a Gaussian domain with mean = 0 and variance = 1, although depending on the original data, it can be slightly off.

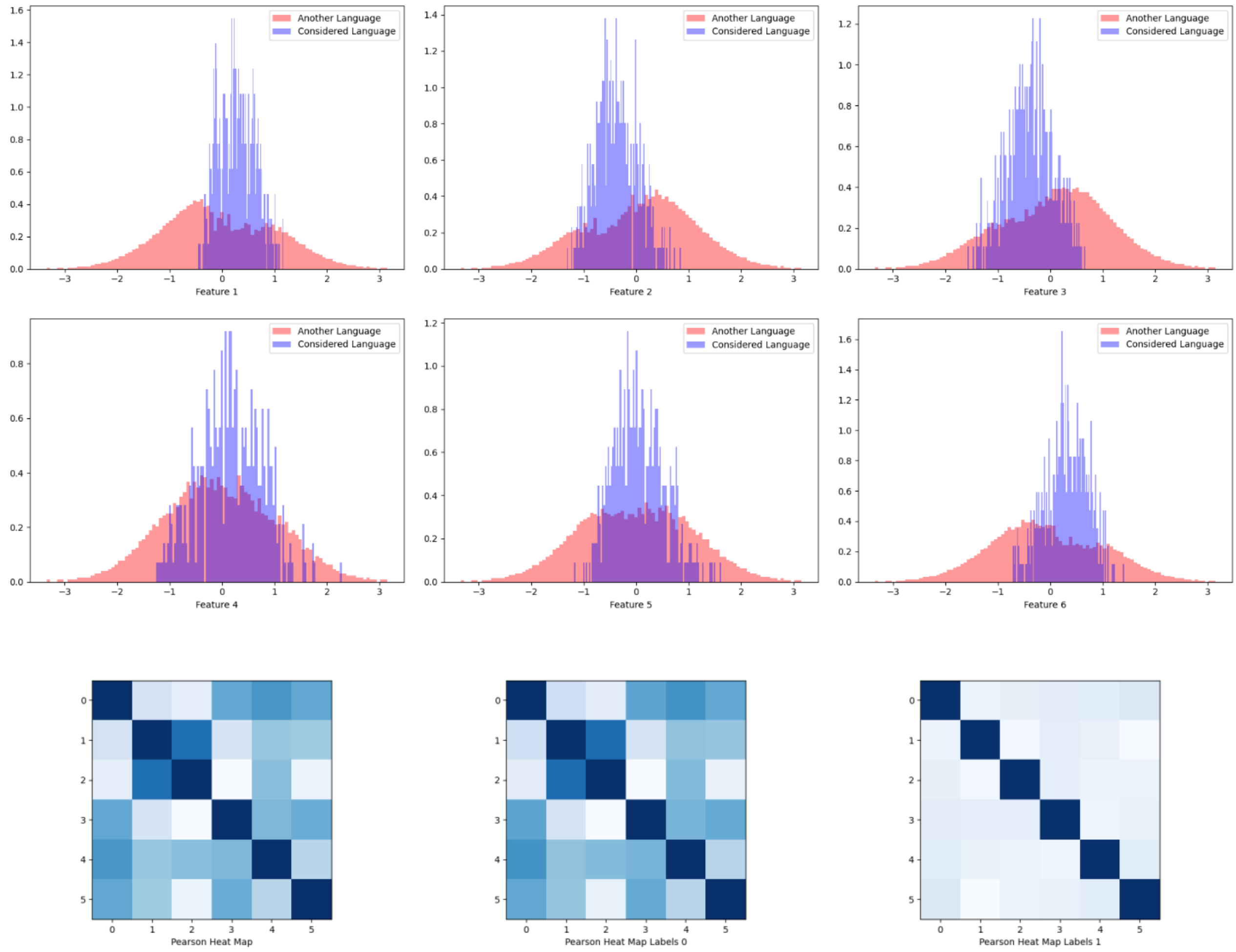

Figure 2. Normalized feature distribution and correlation heat maps

As can be seen, the data is basically the same, as the original dataset was already similar to a gaussian distribution.

In our tests, we will use both the raw data and the normalized data, to compare them.

3 Models training and Tuning

3.1 MVG Classifiers

We will first focus on generative models. The first model we will look at is the MVG classifier, with three different variants, Full, Diagonal and Tied.

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$
Raw Features – NO PCA		
Full-Cov	**0.131**	**0.476**
Diagonal-Cov	**0.126**	**0.476**
Tied-Cov	**0.508**	**1.0**
Raw Features – PCA = 5		
Full-Cov	**0.127**	**0.458**
Diagonal-Cov	**0.123**	**0.472**
Tied-Cov	**0.511**	**1.0**
Normalized Features – NO PCA		
Full-Cov	**0.291**	**0.693**
Diagonal-Cov	**0.276**	**0.674**
Tied-Cov	**0.746**	**1.0**
Normalized Features – PCA = 5		
Full-Cov	**0.297**	**0.696**
Diagonal-Cov	**0.278**	**0.671**
Tied-Cov	**0.684**	**1.0**

Table 1. MVG Classifiers Results with 3-fold on the validation dataset

From this table we can see that the scores evaluated with $\tilde{\pi} = 0.5$ performs much better than the other. Also, the tied assumption on our data is not correct, since in all cases the MVG with tied covariance matrixes performs much worse than the other models.

On the bright side the assumption made before about the dimensionality reduction is correct, since in almost all models, the results improve with lower dimensionality.

The normalization doesn’t seem to help in our case.

3.2 Linear Regression

We will now focus on discriminative models. We start by considering the regularized Linear Regression. To balance the costs of different classes, both applications were considered. The only hyperparameter in this case is λ, and, in that regard values between 10^{-5} and 10^{2} have been tried.

In the following figures, the $\tilde{\pi}$ refers to the prior used to compute the scores.

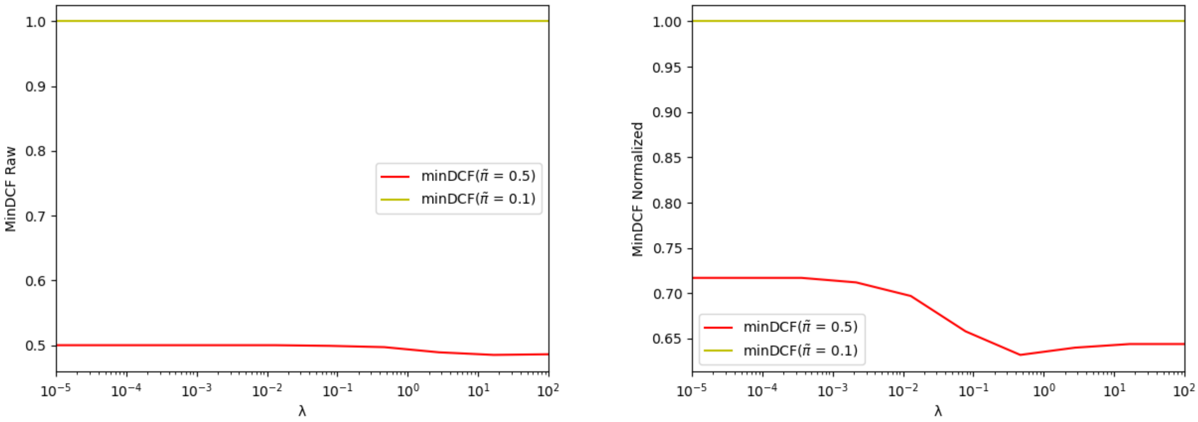

Figure 3. MinDCF for each λ and π_T = 0.5, Left: Raw Data, Right: Normalized Data

As can be seen above, no matter what the value of λ is, the MinDCF is always quite high compared to the previous models.

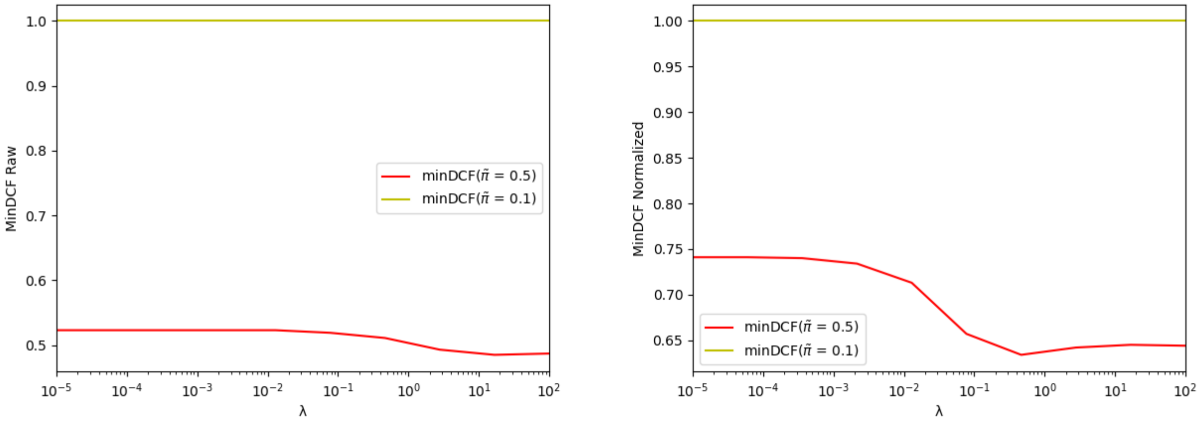

Figure 4. MinDCF for each λ and π_T = 0.1, Left: Raw Data, Right: Normalized Data

The results with the other application are almost identical.

In this case we can safely assume that our data does not exhibit a clear linear separation. This suggest that trying higher degree separations could potentially yield better results.

The effect of PCA in this case is basically zero in this model.

For example, taking the best value found, with λ equal to 1.67 * 10^{1} and π_T = 0.5.

Raw Features – NO PCA $\tilde{\pi} = 0.5$	**Raw Features – PCA = 5 $\tilde{\pi} = 0.5$**
0.485	**0.485**

3.3 Quadratic Regression

Given the assumptions made before, we turn our attention on Quadratic Regression, which allows us to find a quadratic separation in the data. For consistency's sake, the same values as before has been tested.

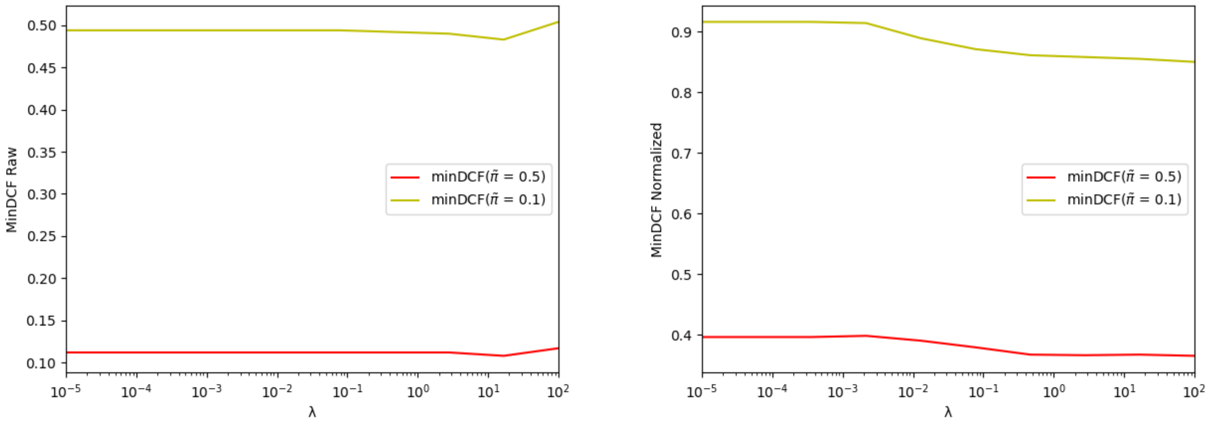

Figure 5. MinDCF for each λ and π_T = 0.5, Left: Raw Data, Right: Normalized Data

As can be seen, the quadratic regression works great with our dataset, having values as low as 0.108. In this model, as in the Linear Regression, the value of lambda has minimal impact on the overall results.

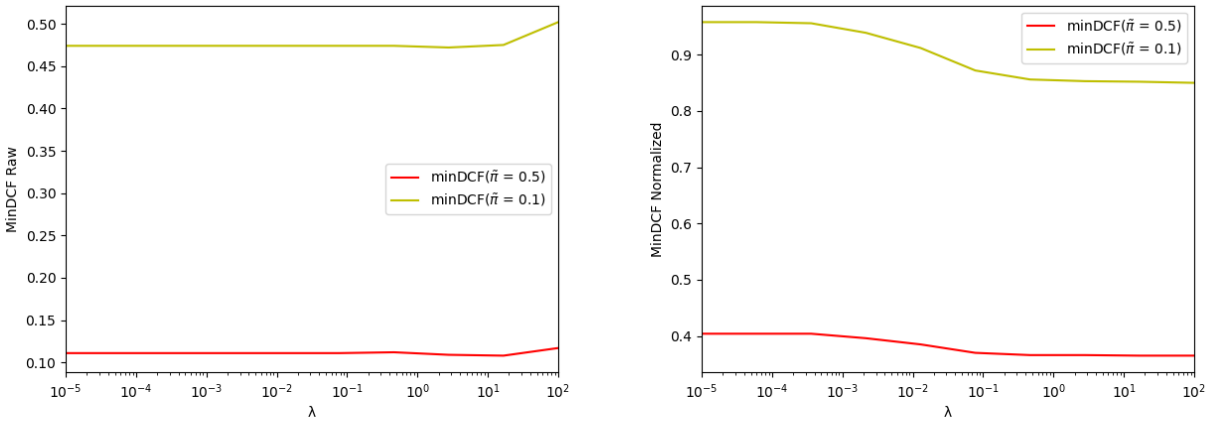

Figure 6. MinDCF for each λ and π_T = 0.1, Left: Raw Data, Right: Normalized Data

The results for the other application are basically the same, with the best value being the same as before.

The PCA in this case is detrimental, taking the best value found, with λ equal to 1.67 * 10^1 (π_T = 0.1).

Raw Features – NO PCA $\tilde{\pi} = 0.5$	**Raw Features – PCA = 5 $\tilde{\pi} = 0.5$**
0.108	**0.27**

3.4 Linear SVM

We will now turn our attention to Support Vector Machines, starting with the Linear Version. Similar to Linear Regression, the Linear SVM aims to find a linear separation in our dataset. Considering this approach, it is anticipated that the model's performance may not be optimal. Additionally, the Linear SVM has two hyperparameters, K and C. We have experimented with three different values for K [0.0, 1.0, 10.0], and ten values for C ranging from $[10^{-2}, 10^{0}]$.

The values shown in the following figures, refers to the best K found, in this case K = 1.0.

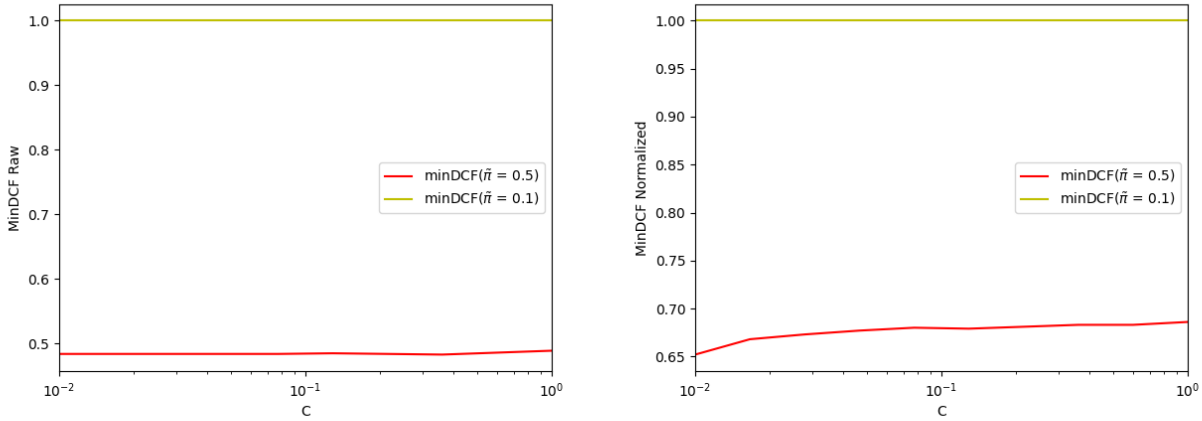

Figure 7. MinDCF for each C for linear SVM K = 1.0 π_T = 0.5, Left: Raw data, Right: Normalized data

As expected, the results are quite bad, this confirms the assumption made with Linear Regression.

We will now focus on the results with the other application, where the best K was 0.0 instead of 1.0, the results shown are still with K = 1.0 to compare the two applications.

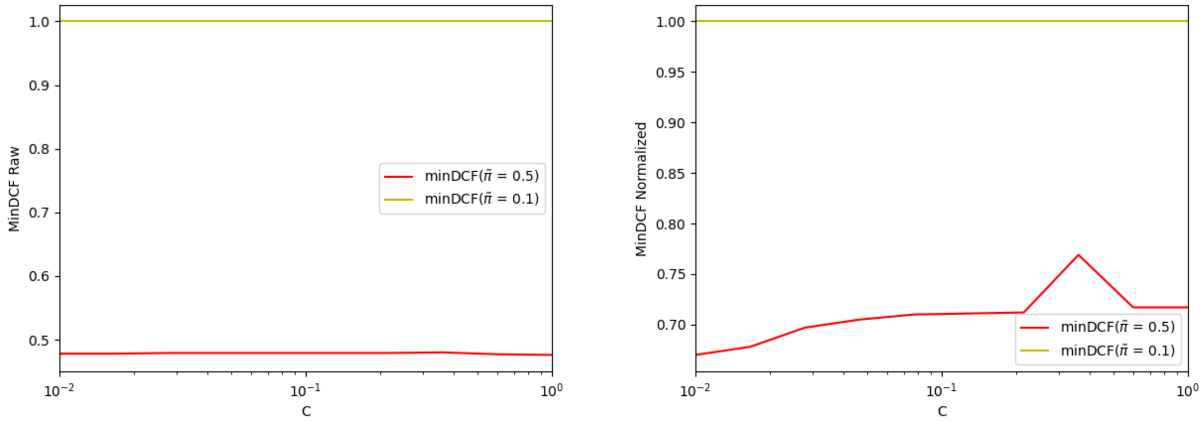

Figure 8. MinDCF for each C for linear SVM K = 1.0 π_T = 0.1, Left: Raw data, Right: Normalized data

Also in this case, the effect of PCA is almost negligible, taking the best combination from both applications, the resulting change is minimal and does not significantly impact the outcomes.

3.5 Polynomial SVM

In this model we introduce a Kernel Function defined as $k(x1, x2) = ({x_1}^T x_2 + c)^d$ which allows to use linear separation in an expanded space, corresponding to a nonlinear separation surface in the original one. The crucial hyperparameter in this model is d, that determines the complexity of the polynomial mapping. The other hyperparameter are K, C and c (the regularization parameter).

The polynomial SVM was experimented with using the same three values as the Linear SVM for the kernel parameter K. However, for the regularization parameter C, a range of five values spanning from $[10^{-2}, 10^{0}]$ was explored. As for the degree parameter d, three different values, namely [2, 3 4], were tested to avoid relying solely on a linear separation surface, which was deemed unsuitable for our dataset. Lastly, the regularization parameter c was varied within the range of [0.0, 1.0].

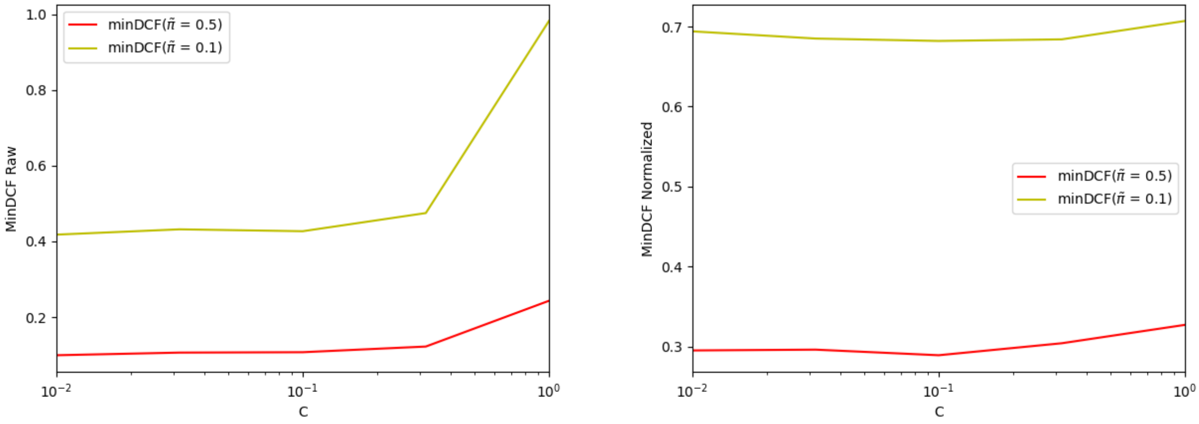

Figure 9. MinDCF for each C for Poly SVM and π_T = 0.5 d = 2 K = 10.0 c = 1.0, Left: Raw data, Right: Normalized data

The best value found with application π_T = 0.5 are with K = 10.0 d = 2 and c = 1.0.

For the sake of consistency, the results with application π_T = 0.1 are also shown with the same values, even if the best results have been found with K = 1.0, d = 2 and c = 0.0.

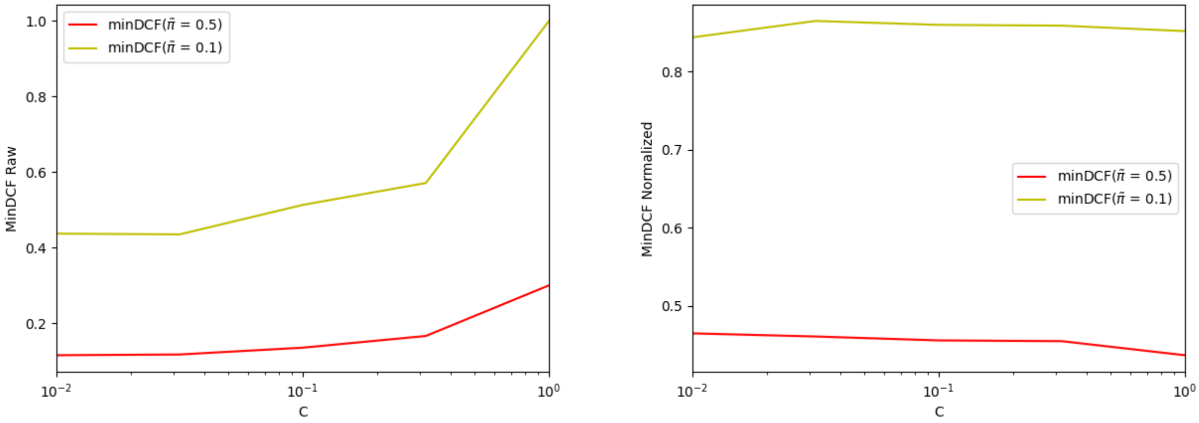

Figure 10. MinDCF for each C for Poly SVM and π_T = 0.1 d = 2 K = 10.0 c = 1.0, Left: Raw data, Right: Normalized data

As we can see from the pictures above, the results for both applications are quite similar, with the results with the raw data being almost identical.

In this regard, just to further confirm the assumption that our data is well suited for a quadratic separation surface, some results with d equal to 3 and 4 are also shown, both with PCA and without it.

To show the effectiveness of higher degree separation rules, the best values found for each of the three degrees will be shown, for both applications, with and without PCA.

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$
Raw Features – NO PCA		
SVM Poly (K = 10.0, C = 0.01, d = 2, c = 1.0, π_T = 0.5)	**0.099**	**0.418**
SVM Poly (K = 10.0, C ≅ 0.032, d = 3, c = 1.0, π_T = 0.5)	**0.339**	**1.0**
SVM Poly (K = 10.0, C = 0.1, d = 4, c = 1.0, π_T = 0.5)	**0.167**	**1.0**
SVM Poly (K = 1.0, C = 0.01, d = 2, c = 0.0, π_T = 0.1)	**0.115**	**0.537**
SVM Poly (K = 10.0, C = 0.01, d = 3, c = 1.0, π_T = 0.1)	**0.326**	**1.0**
SVM Poly (K = 0.0, C ≅ 0.032, d = 4, c = 1.0, π_T = 0.1)	**0.172**	**1.0**
Raw Features – PCA = 5		
SVM Poly (K = 1.0, C = 0.01, d = 2, c = 0.0, π_T = 0.5)	**0.094**	**0.464**
SVM Poly (K = 10.0, C ≅ 0.032, d = 3, c = 1.0, π_T = 0.5)	**0.365**	**1.0**
SVM Poly (K = 0.0, C ≅ 0.032, d = 4, c = 1.0, π_T = 0.5)	**0.22**	**1.0**
SVM Poly (K = 10.0, C ≅ 0.032, d = 2, c = 1.0, π_T = 0.1)	**0.113**	**0.437**
SVM Poly (K = 10.0, C = 0.01, d = 3, c = 1.0, π_T = 0.1)	**0.278**	**1.0**
SVM Poly (K = 10.0, C = 0.01, d = 4, c = 1.0, π_T = 0.1)	**0.172**	**0.994**

Table 2. SVM Poly results for different applications, degrees and PCA

From the table below, it’s possible to appreciate the effect of PCA, even if by few thousands. One intriguing observation worth noting is that while we expected quadratic separation rules (d = 2) to perform the best, it turns out that going from d = 3 to d = 4 leads to improved results. However, it is important to highlight that both d = 3 and d = 4 still exhibit inferior performance compared to the quadratic separation (d = 2). This kind of pattern is quite unusual.

The plots presented above (Figure 9 and 10) demonstrate that normalization has a negative impact. This consistent trend can be observed across all the models examined until now.

Additionally, it is important to note that for both the Polynomial and RBF kernels in SVM, a regularized bias term is required.

In our case the bias term is equal to $\varepsilon = K^2$ so in the case of the polynomial kernel, the final form will be $\widehat{K}(x_1, x_2) = K(x_1, x_2) + \varepsilon$.

3.6 Radial Basis Function SVM

This model differs from the last one by using a different kernel function, defined as $k(x_1, x_2) = \ e^{-\gamma||x_1 - x_2||^2}$. This kernel works by using the concept of similarity, so it operates in a potentially infinite dimensional space due to its nature. The model has three different hyperparameter, K, C and γ. The values tried for K and C are the same as in the last model, while three values of γ has been tried, namely $[10^{-3}, 10^{-2}, 10^{-1}]$.

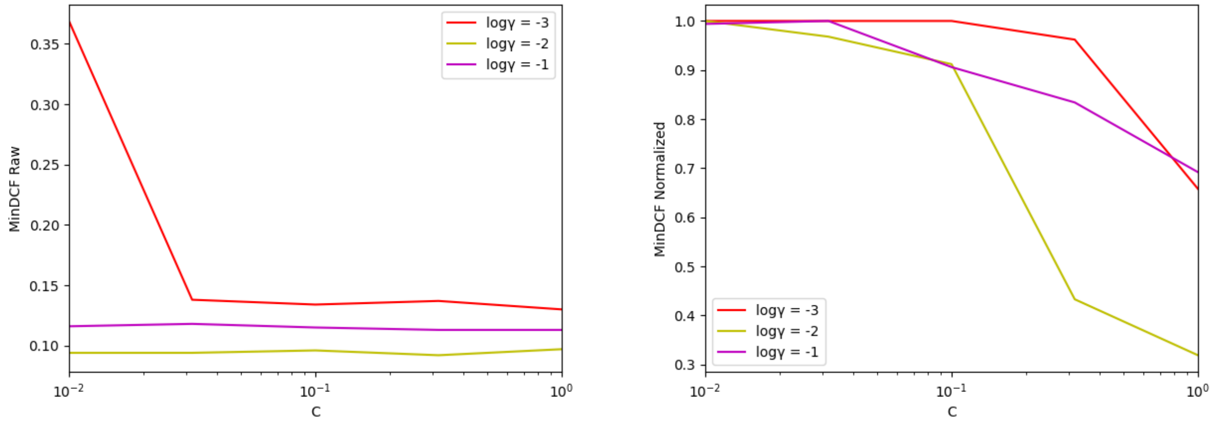

Figure 11. MinDCF for each C for SVM RBF and π_T = 0.5 K = 1.0, Left: Raw Data, Right: Normalized Data

As can be seen above, with normalized data, the results tend to worsen, however, it is worth noting that as the values of C increase, the results show improvement, suggesting that experimenting with higher values might yield better outcomes.

The plots have been done using the best K found, equal to 1.0.

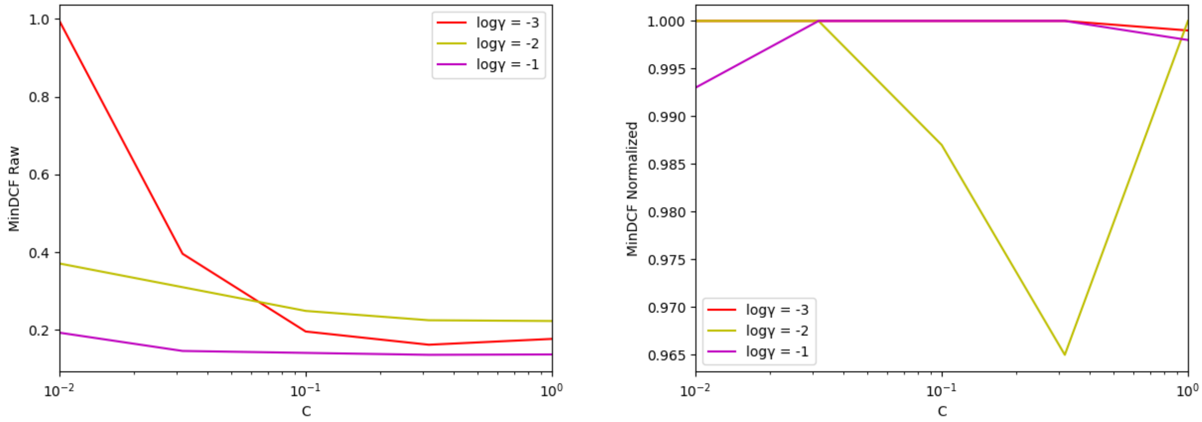

Figure 12. MinDCF for each C for SVM RBF and π_T = 0.1 K = 1.0, Left: Raw Data, Right: Normalized Data

Looking at the pictures above, it's possible to appreciate the difference between the application, while in the first one (π_T = 0.5) the best values were found with $\log \gamma = -2$, in the second one (π_T = 0.1) the same value was the worst of all three, with $\log \gamma = -1$ being the best.

As before, going up with the values of C, the model tends to yield better results.

The results of PCA, also in this case, are small, but they sometimes tend to improve the overall result, to show the effect of the latter, as usual, some results will be shown.

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$
Raw Features – NO PCA		
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -2$, π_T = 0.5)	**0.092**	**0.38**
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -2$, π_T = 0.1)	**0.225**	**0.454**
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -1$, π_T = 0.5)	**0.113**	**0.428**
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -1$, π_T = 0.1)	**0.136**	**0.429**
Raw Features – PCA = 5		
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -2$, π_T = 0.5)	**0.091**	**0.379**
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -2$, π_T = 0.1)	**0.224**	**0.448**
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -1$, π_T = 0.5)	**0.117**	**0.386**
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -1$, π_T = 0.1)	**0.154**	**0.402**

Table 3. SVM RBF results with and without PCA

Again, as said before, the change with the PCA applied, are negligible, a trend that we have seen among all the SVM models, this can be a result of the fact that the dimensionality is already quite low to begin with (only 6 features) or that the SVMs are already complex enough to capture the patterns in the original feature space without the need for dimensionality reduction.

3.7 Gaussian Mixture Models

The last model we consider is GMM, three different versions of the model has been tried, the full covariance (where each component has a unique covariance), the tied covariance (same covariance for all components) and diagonal covariance (different covariance for each component but diagonalized). Different numbers of components have been tried, namely [1, 2, 4, 8, 16, 32].

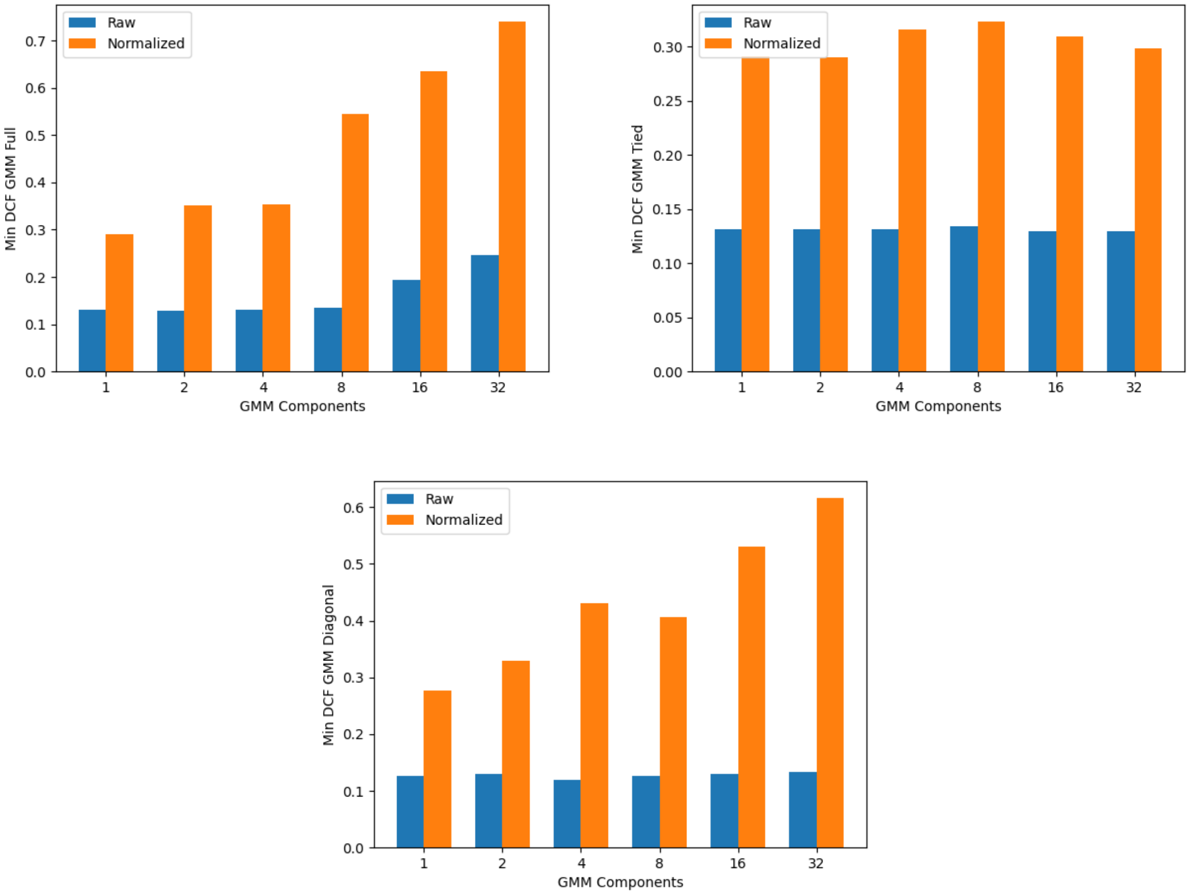

Figure 13. MinDCF by n of components NO PCA, Top-Left: Full, Top-Right: Tied, Bottom: Diagonal

From the picture above it’s possible to appreciate that the number of components doesn’t really change much the results, and going higher is detrimental, as can be seen from the Full Covariance’s results.

About the PCA, as in the other models, the change is almost negligible, however, in order to show the impact of it, some of the results will be shown as plotting them as done above will not effectively capture the observed change.

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$
Raw Features – NO PCA		
GMM Full (n = 2)	**0.131**	**0.476**
GMM Full (n = 8)	**0.134**	**0.514**
GMM Tied (n = 2)	**0.131**	**0.476**
GMM Tied (n = 8)	**0.134**	**0.473**
GMM Diagonal (n = 2)	**0.129**	**0.457**
GMM Diagonal (n = 8)	**0.126**	**0.495**
Raw Features – PCA = 5		
GMM Full (n = 2)	**0.123**	**0.492**
GMM Full (n = 8)	**0.14**	**0.499**
GMM Tied (n = 2)	**0.127**	**0.461**
GMM Tied (n = 8)	**0.125**	**0.458**
GMM Diagonal (n = 2)	**0.129**	**0.458**
GMM Diagonal (n = 8)	**0.146**	**0.481**

Table 4. GMM Results with and without PCA

Only results with n equals 2 and 8 are shown for the reason said above. As can be seen, the effects of PCA are quite negligible, however, it improves the results almost across the whole board.

The Tied covariance model, unlike in the MVG, does surprisingly well.

4 Score Calibration

With score calibration we aim to recalibrate the scores coming out of a model where they are not well-calibrated. In this case the calibration was done with a K-Fold approach, with K = 3 and logistic regression to compute the calibration parameters. When using this kind of approach, it's possible for the MinDCF to change slightly, because we are calibrating the score of a fold using the α and β coming from the other part, so the computed parameters differ a little bit across the folds. However, the change is always minimal, but the ActDCF experiences more significant alterations as a result of the calibration process.

We expect that the calibrated scores provide a more similar ActDCF across the two applications, bringing them closer with respect to the non-calibrated ones.

To show the effectiveness of the process, some results will be shown, referring to the ActDCF. We also expect it to work particularly well on models not well-calibrated like the SVMs.

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	Difference
Raw Features – NO PCA NO Calibration			
SVM Linear (K = 1.0, C ≅ 0.36, π_T = 0.5)	**0.489**	**4.298**	**3.809**
SVM Poly (K = 10.0, C = 0.01, d = 2.0, c = 1.0, π_T = 0.5)	**0.158**	**0.438**	**0.28**
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -2$, π_T = 0.5)	**0.175**	**0.415**	**0.24**
GMM Full (n = 2)	**0.134**	**0.763**	**0.629**
GMM Tied (n = 2)	**0.144**	**0.956**	**0.812**
GMM Diagonal (n = 2)	**0.136**	**0.847**	**0.711**
Raw Features – NO PCA with Calibration			
SVM Linear (K = 1.0, C ≅ 0.36, π_T = 0.5)	**1.001**	**1.005**	**0.004**
SVM Poly (K = 10.0, C = 0.01, d = 2.0, c = 1.0, π_T = 0.5)	**0.153**	**0.433**	**0.28**
SVM RBF (K = 1.0, C ≅ 0.032, $\log \gamma = -2$, π_T = 0.5)	**0.192**	**0.412**	**0.22**
GMM Full (n = 2)	**0.207**	**1.105**	**0.898**
GMM Tied (n = 2)	**0.175**	**1.045**	**0.87**
GMM Diagonal (n = 2)	**0.178**	**1.08**	**0.902**

Table 5. Effect of calibration on different models in terms of ActDCF

As can be seen, the effects on the SVMs are good, decreasing the difference between the two applications. In other models like the GMM, however, calibration doesn't seem to provide a benefit.

5 Experimental Results

In this final part we will try all the different models on the test set, training on the whole train set. In order to maintain consistency with the previously applied methodology, when both normalization and PCA are employed, the latter is done after the normalization. There are two different lines of though in this regard, PCA before normalization or normalization before PCA, in this case, as said before, the latter has been chosen. Since there is no correct way, for computational (and time) constraints, the other methodology has not been tested, so it's not possible to know if it would've been better or worse, however, the main focus in this project was the consistency between the training and evaluation phase.

We expect to have similar results as in the training phase, with quadratic separation rule models to perform generally better.

In order to show the results and to keep the paper concise, the results will primarily focus on the values obtained with the previously mentioned hyperparameters (during the tuning phase). However, if a new set of hyperparameters surpasses the performance of the latter results, they will also be presented. This is because all previously attempted parameter sets have undergone evaluation in this aspect.

We will evaluate the models in term of MinDCF, as done during the training phase, this allows us to better compare the models when using the optimal threshold.

Model Type	$\widetilde{\pi} = 0.5$	$\widetilde{\pi} = 0.1$
Raw Features – NO PCA		
MVG Full-Cov	**0.135**	**0.516**
MVG Diagonal-Cov	**0.135**	**0.521**
MVG Tied-Cov	**0.596**	**1.0**
Linear Regression (λ = 1.67e+01, π_T = 0.5)	**0.586**	**1.0**
Linear Regression (λ = 1.67e+01, π_T = 0.1)	**0.587**	**1.0**
Quadratic Regression (λ = 1.67e+01, π_T = 0.5)	0.129	**0.559**
Quadratic Regression (λ = 1.67e+01, π_T = 0.1)	0.13	**0.558**
SVM Linear (K = 1.0, C = 0.01, π_T = 0.5)	**0.573**	**1.0**
SVM Linear (K = 1.0, C = 0.01, π_T = 0.1)	**0.769**	**1.0**
SVM Poly (K = 10.0, C = 0.01, d = 2, c = 1.0, π_T = 0.5)	0.109	**0.384**
SVM Poly (K = 10.0, C = 0.01, d = 2, c = 1.0, π_T = 0.1)	0.145	**0.441**
SVM RBF (K = 1.0, C ≅ 0.032, $\log\gamma = -2$, π_T = 0.5)	0.103	**0.42**
SVM RBF (K = 1.0, C ≅ 0.032, $\log\gamma = -2$, π_T = 0.1)	**0.246**	**0.441**
GMM Full (n = 2)	**0.131**	**0.495**
GMM Full (n = 8)	**0.132**	**0.485**
GMM Tied (n = 2)	**0.135**	**0.517**
GMM Tied (n = 8)	**0.13**	**0.499**
GMM Diagonal (n = 2)	**0.125**	**0.509**
GMM Diagonal (n = 8)	**0.144**	**0.535**

Table 6. Results on the test set in terms of MinDCF without PCA

The results are completely in line with what we expected, with the models with quadratic separation rules being among the best and the RBF SVM being the absolute best, as seen during the training phase.

We will now concentrate on results with PCA applied, as in the training phase, we expect them to be comparable with those obtained without it, with some increase in performance here and there.

Model Type	$\widetilde{\pi} = 0.5$	$\widetilde{\pi} = 0.1$
Raw Features – PCA = 5		
MVG Full-Cov	**0.131**	**0.519**
MVG Diagonal-Cov	**0.134**	**0.519**
MVG Tied-Cov	**0.588**	**1.0**
Linear Regression (λ = 1.67e+01, π_T = 0.5)	**0.587**	**1.0**
Linear Regression (λ = 1.67e+01, π_T = 0.1)	**0.587**	**1.0**
Quadratic Regression (λ = 1.67e+01, π_T = 0.5)	**0.252**	**0.877**
Quadratic Regression (λ = 1.67e+01, π_T = 0.1)	**0.253**	**0.862**
SVM Linear (K = 1.0, C = 0.01, π_T = 0.5)	**0.567**	**1.0**
SVM Linear (K = 1.0, C = 0.01, π_T = 0.1)	**0.627**	**1.0**
SVM Poly (K = 10.0, C = 0.01, d = 2, c = 1.0, π_T = 0.5)	**0.112**	**0.38**
SVM Poly (K = 10.0, C = 0.01, d = 2, c = 1.0, π_T = 0.1)	**0.127**	**0.404**
SVM RBF (K = 1.0, C ≅ 0.032, $\log\gamma = -2$, π_T = 0.5)	**0.105**	**0.41**
SVM RBF (K = 1.0, C ≅ 0.032, $\log\gamma = -2$, π_T = 0.1)	**0.236**	**0.447**

Table 7. Results on test set in terms of MinDCF with PCA Part 1

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$
Raw Features – PCA = 5		
GMM Full (n = 2)	0.125	0.482
GMM Full (n = 8)	0.135	0.456
GMM Tied (n = 2)	0.132	0.519
GMM Tied (n = 8)	0.134	0.51
GMM Diagonal (n = 2)	0.129	0.505
GMM Diagonal (n = 8)	0.135	0.53

Table 8. Results on test set in terms of MinDCF with PCA Part 2

The green results are the one that became better with PCA, while the red ones are those that worsened with it.

As expected, the change between the two is almost negligible in all cases, with the biggest difference in the order of the hundreds at most.

We now turn our attention on the results with normalization, we expect all the models to work worse than the raw features, as seen on the training phase. To keep the report concise, only the results with no PCA applied will be shown, however, this will provide a good overview of the effect of normalization.

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$
Gaussianized Features – NO PCA		
MVG Full-Cov	0.139	0.54
MVG Diagonal-Cov	0.152	0.586
MVG Tied-Cov	0.649	1.0
Linear Regression (λ = 1.67e+01, π_T = 0.5)	0.59	1.0
Linear Regression (λ = 1.67e+01, π_T = 0.1)	0.59	1.0
Quadratic Regression (λ = 1.67e+01, π_T = 0.5)	0.397	0.997
Quadratic Regression (λ = 1.67e+01, π_T = 0.1)	0.397	0.997
SVM Linear (K = 1.0, C = 0.01, π_T = 0.5)	0.586	1.0
SVM Linear (K = 1.0, C = 0.01, π_T = 0.1)	0.886	1.0
SVM Poly (K = 10.0, C = 0.01, d = 2, c = 1.0, π_T = 0.5)	0.15	0.435
SVM Poly (K = 10.0, C = 0.01, d = 2, c = 1.0, π_T = 0.1)	0.139	0.469
SVM RBF (K = 1.0, C ≅ 0.032, $\log\gamma = -2$, π_T = 0.5)	0.915	1.0
SVM RBF (K = 1.0, C ≅ 0.032, $\log\gamma = -2$, π_T = 0.1)	1.0	1.0
GMM Full (n = 2)	0.14	0.548
GMM Full (n = 8)	0.137	0.501
GMM Tied (n = 2)	0.139	0.54
GMM Tied (n = 8)	0.139	0.531
GMM Diagonal (n = 2)	0.147	0.542
GMM Diagonal (n = 8)	0.139	0.52

Table 9. Results on the test set in terms of MinDCF without PCA and normalization

Again, as expected, the results are worse in almost every case, however, the difference between raw data and normalized ones is not as high as in the training phase. This can be the consequence of the way the data were normalized in the first case. With a k-fold approach where every fold is normalized with the ranking from the other fold, and with the very few samples in the training set, the computation of the rankings can be quite imprecise. In this regard, using a higher value of K could've helped (however this is also true in general).

Just to further show the difference between the training results and the testing ones obtained with normalization, a plot of SVM Poly in both cases will be shown. There is no particular reason on the choice of the latter one, it's just for comparison purposes.

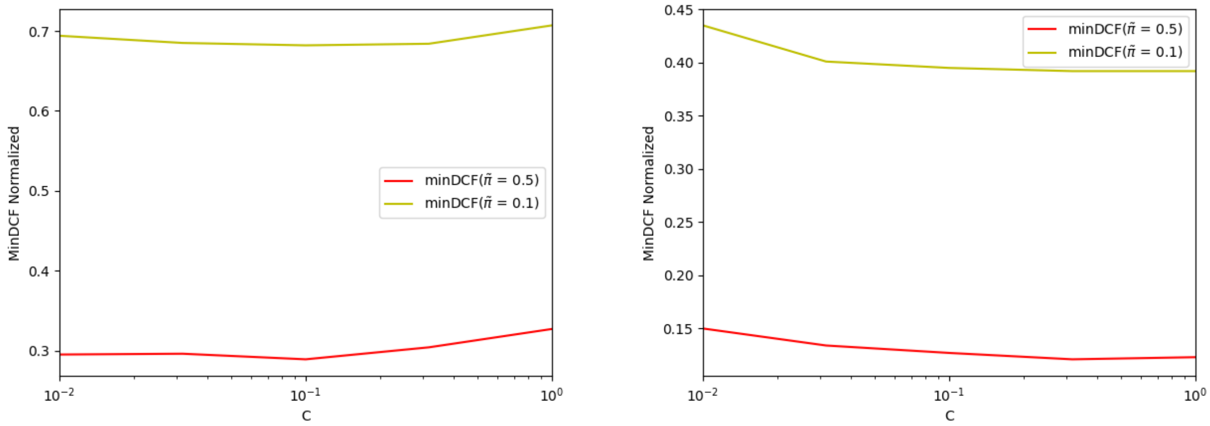

Figure 14. MinDCF for each C for SVM Poly and π_T = 0.5 K = 10.0 d = 2 c = 1.0, Left: Training Phase, Right: Evaluation Phase

As we can see the trend is almost the same, but the values are much smaller.

6 Conclusions

Finally, we can assess how good our models are in terms of ActDCF, while MinDCF allows us to evaluate a model when the best possible threshold is found, we also want to know how good our predictions are on the chosen applications. To show it, only the best few models, based on the assumptions made until now, will be shown.

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$
Raw Features – NO PCA NO Recalibration		
SVM Poly (K = 10.0, C = 0.1, d = 2, c = 1.0, π_T = 0.5)	**0.14**	**0.488**
SVM RBF (K = 0.0, C = 0.01, $\log\gamma = -2$, π_T = 0.5)	**0.122**	**0.506**
GMM Full (n = 2)	**0.134**	**0.842**
Quadratic Regression (λ = 1.67e+01, π_T = 0.5)	**0.132**	**1.092**

Table 10. Best results on the test set in terms of ActDCF

From this table we can see that the results are in line with what we've seen so far, the only thing to keep in mind is that while some of these models provide good results with $\tilde{\pi} = 0.5$ they don't when evaluating with the other $\tilde{\pi}$. In this case the SVM provides more balanced scores.

In this case rebalancing the score can improve the performance.

Model Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$
Raw Features – NO PCA with Recalibration		
SVM Poly (K = 10.0, C = 0.1, d = 2, c = 1.0, π_T = 0.5)	**0.156**	**0.458**
SVM RBF (K = 0.0, C = 0.01, $\log \gamma = -2$, π_T = 0.5)	**0.648**	**0.701**
GMM Full (n = 2)	**0.215**	**1.1**

Table 11. Best results on the test set in terms of ActDCF

Finally, we can conclude that with recalibration it’s possible to narrow the gap between different applications and that with this dataset, the models tried work pretty well, especially those which can find more complex patterns in the data.