Politecnico di Torino

Computer Engineering

Alexandro Buffa - s316999

Simone Giambrone - s317002

Machine Learning and Pattern Recognition

Language Classification

Academic Year 2022/23

Contents

Chapter 1

Introduction

1.1 Abstract

The goal of the application is to build a model that best fits the Language Detection task exploiting the most common Machine Learning tools. We will analyze and discuss their performance, explaining pros and cons.

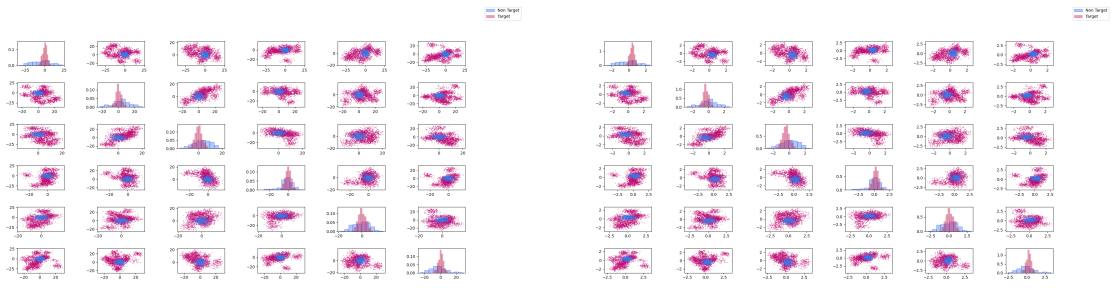
1.2 Problem Overview

The dataset consists of utterances belonging to different languages and are represented by means of language embeddings, i.e. low-dimensional representations of speech obtained by mapping audio segments to a duration-independent, low-dimensional manifold. The embeddings are 6-dimensional, continuous-valued vectors, belonging to either the target language (label 1) or the non-target language (label 0) class. The embedding components do not have a physical interpretation.

The training set consists of 2371 samples in total, 1971 sample for the non target class and 400 for the target class, whereas the test set contains 4403 samples, 3603 for non target class and 800 for target class. We can clearly see that the datasets are unbalanced, with both sets having significantly more non target samples.

1.3 Features Analysis

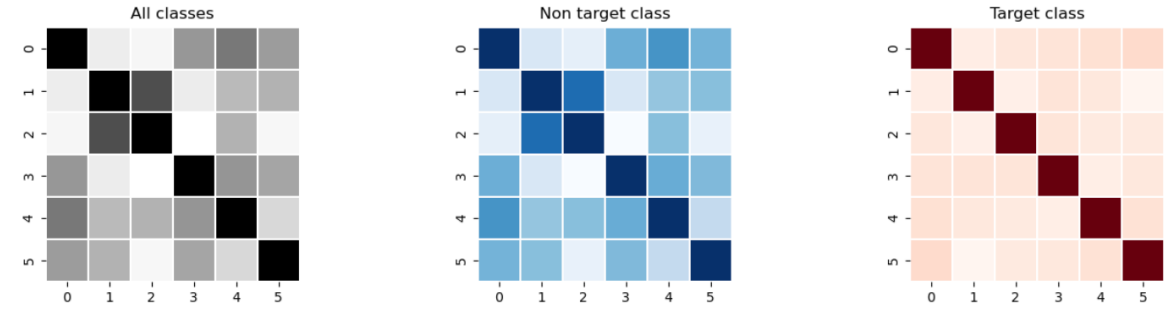
We represented the 6 features of the dataset with histograms and scatter plots, both on RAW and z-normalized data to better understand how they are related among each other.



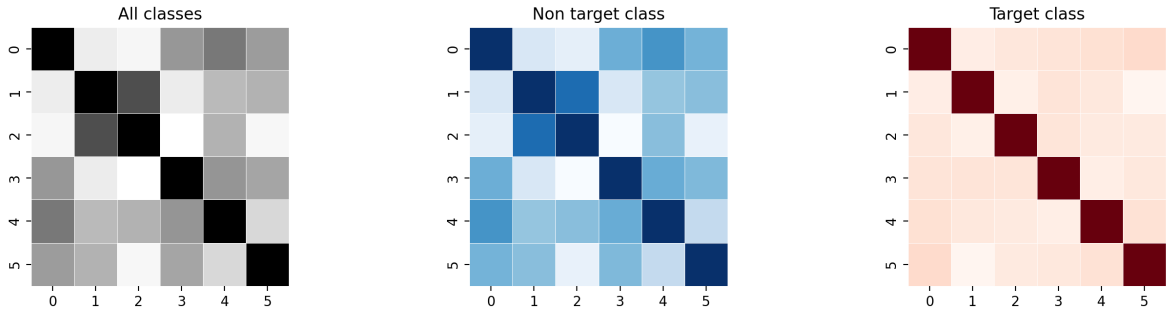
(a) (b)

Figure 1.1: (a) RAW features (b) Z-Normalized Features.

We also analyzed the Pearson Correlation among the different features using heatmaps, calculating it for the whole dataset in the first hand and then for the separated classes.



(a)



(b)

Figure 1.2: Pearson Correlation - (a) RAW features (b) Z-Normalized Features.

By analyzing the first heatmap, we can notice that the features $(1, 2)$ have a strong correlation, meaning that we may benefit by using PCA to reduce the dimensionality of our dataset, but since we have 6 features per sample, we risk to lose some important information by using it.

Chapter 2

Dimensionality Reduction

We analyzed the variance of our training set after using PCA as a Dimensionality Reduction technique to see if it's worth using it, both on the RAW dataset and after applying z-normalization.

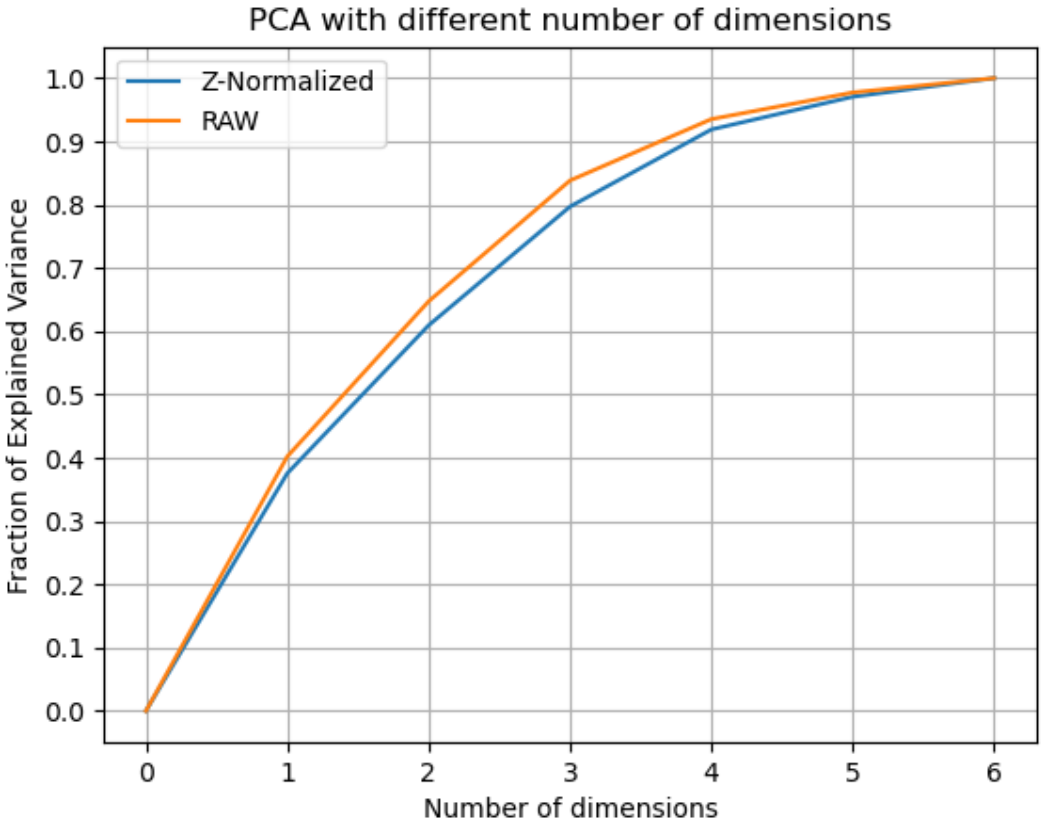


Figure 2.1: Fraction of explained variance after PCA.

From the plot above, first of all we can notice that PCA performs better on the RAW dataset rather than on the z-normalized one. On closer inspection we can see that we lose around 3% of variance after reducing our dimensionality by one dimension.

Chapter 3

Classification and Validation

3.1 Introduction

To write this report, we collected results from various classification models, listed below:

- Generative Models - Linear and Quadratic Classifiers
  - Multivariate Gaussian Classifier (**MVG**)
  - MVG + Diagonal Covariance
  - MVG + Tied Covariance
  - MVG + Diagonal Covariance with Tied Covariance
- Logistic Regression
  - Binary Logistic Regression
  - Quadratic Logistic Regression
  - Prior Weighted Logistic Regression
- Support Vector Machine (**SVM**)
  - Linear SVM
  - Polynomial SVM of degree 2 and 3
  - SVM with RBF kernel
- Gaussian Mixture Models (**GMM**)
  - Basic GMM
  - GMM + Diagonal Covariance
  - GMM + Tied Covariance

Before moving on to the analysis, we have to specify some basic guidelines that we will follow throughout this phase.

- All of the following results have been calculated after performing a K-Fold Stratified Cross Validation on the dataset with K = 5, ensuring that the ratio betweeen the two classes is kept in every fold.

- Inside each cell of the following tables, we have reported the **minDCF** and its average on both our application working points, we will call this metric C_{prim}.
  We do not care about **actDCF** in this phase, we will consider it after choosing the most promising model(s).
- **minDCF** has been computed with $C_{fn} = C_{fp} = 1$, as per project specification. We will consider different working points in our analysis, in particular we will focus on three different ones:
  - The ones needed for our application:

$$(\pi, C_{fn}, C_{fp}) = (0.5, 1, 1)$$

$$(\pi, C_{fn}, C_{fp}) = (0.1, 1, 1)$$

  - A third one:

$$(\pi, C_{fn}, C_{fp}) = (0.9, 1, 1)$$

- All the tables in this report will follow the following color scheme to represent results:
  - ***This Color*** will be used to represent the best result we obtained. Each table will feature two (or more in case of equal top results): one for the minDCF(actDCF in the last phase) and one for the C_{prim} metric.
  - ***This Color*** will be used to represent the second best result we obtained. Each table will feature two (or more in case of equal top results): one for the minDCF(actDCF in the last phase) and one for the C_{prim} metric.
  - ***This Color*** will be used to represent the third best result we obtained. Each table will feature two (or more in case of equal top results): one for the minDCF(actDCF in the last phase) and one for the C_{prim} metric.
  - This Color will be used to represent the worst 10% of the obtained results. Each table will feature a bottom 10% for the minDCF(actDCF in the last phase) and one for the C_{prim} metric.
  - This Color will be used to represent the best 10% of the obtained results. Each table will feature a top 10% for the minDCF(actDCF in the last phase) and one for the C_{prim} metric.

3.2 Multivariate Gaussian Classifier

We start considering Gaussian classifiers (MVG classifier, MVG classifier with Naive Bayes assumption, MVG classifier with Tied Covariance, and the combination of the last two).
This generative model assumes gaussian distributed data:

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

In particular, the Tied Covariance assumes that the classes share the same covariance matrix and the Naive Bayes assumes that the features are indipendently distributed, impliying that covariance matrices are diagonal.

3.2.1 Expectations

From the feature analysis we can grasp a gaussian distribution for every feature, so we can imagine a good classification via MVG. Moreover, as we observed through the Pearson Correlation Heatmap, some features, like $(1, 2)$, are highly correlated, so we can expect that the Naive assumption will not improve at all our model.

3.2.2 Results

MVG		**RAW**			
		$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$	C_{prim}
no PCA	**Naive**	0,541	0,137	0,182	0,339
	Standard	0,497	*0,125*	0,189	***0,311***
	Tied	1	0,957	0,957	0,979
	Tied Naive	1	0,708	0,708	0,854
PCA $m = 5$	**Naive**	0,499	*0,125*	0,184	***0,312***
	Standard	0,517	***0,122***	0,187	0,320
	Tied	1	0,955	0,955	0,978
	Tied Naive	1	0,956	0,956	0,978
		Z-Norm			
		$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$	C_{prim}
no PCA	**Naive**	0,541	0,137	0,182	0,339
	Standard	0,497	*0,125*	0,189	***0,311***
	Tied	1	0,957	0,957	0,979
	Tied Naive	1	0,708	0,708	0,854
PCA $m = 5$	**Naive**	0,573	0,156	0,263	0,365
	Standard	0,517	0,132	0,187	0,325
	Tied	1	0,957	0,957	0,979
	Tied Naive	1	0,963	0,963	0,982

3.2.3 Considerations

Even though we expected better results overall, we can quite surely affirm that, as we expected, Naive assumption only worsen our results, being at its worst when combined with the Tied assumption.

3.3 Logistic Regression

We will now consider Logistic Regression, a model that, rather than modeling the distribution of the samples $X|C$, models the class posterior distribution $C|X$. We'll consider both the weighted and unweighted versions of the model, to see which one performs better. We'll also use both the linear and quadratic versions of the model, which allow us to find different separation rules, linear in the first case, quadratic in the second one.

3.3.1 Expectations

By analyzing our dataset shape, we can expect linear Logistic Regression to perform way worse than the quadratic one. We will perform the analysis in subsequent steps, considering each time the most performing and promising solution(s) to advance to the following step(s).

3.3.2 Results

Logistic Regression	$\pi = 0.1$			
	RAW		**Z-Norm**	
λ	**Linear**	**Quadratic**	**Linear**	**Quadratic**
10	1	0.394	1	0.614
10^{-1}	1	0.397	1	0.466
10^{-2}	1	0.415	1	0.419
10^{-3}	1	0.422	1	0.405
10^{-4}	1	0.422	1	0.412
10^{-5}	1	0.422	1	0.422
10^{2}	1	0.388	1	0.705
10^{3}	1	0.4	1	0.916
	$\pi = 0.5$			
10	0.49	0.098	0.492	0.189
10^{-1}	0.503	0.096	0.491	0.142
10^{-2}	0.504	*0.095*	0.5	0.116
10^{-3}	0.504	*0.095*	0.504	0.098
10^{-4}	0.504	*0.095*	0.504	*0.095*
10^{-5}	0.504	*0.095*	0.504	***0.094***
10^{2}	0.486	0.097	0.501	0.221
10^{3}	0.487	0.101	0.584	0.242
	$\pi = 0.9$			
10	0.537	0.137	0.539	0.305
10^{-1}	0.544	0.135	0.536	0.251
10^{-2}	0.545	0.136	0.543	0.224
10^{-3}	0.545	0.137	0.544	0.142
10^{-4}	0.545	0.137	0.545	0.129
10^{-5}	0.545	0.137	0.545	0.137
10^{2}	0.517	0.139	0.542	0.335
10^{3}	0.518	0.139	0.619	0.368
	C_{prim}			
10	0.7450	*0.2460*	0.7460	0.4015
10^{-1}	0.7515	***0.2465***	0.7455	0.3040
10^{-2}	0.7520	0.2550	0.7500	0.2675
10^{-3}	0.7520	0.2585	0.7520	0.2515
10^{-4}	0.7520	0.2585	0.7520	0.2535
10^{-5}	0.7520	0.2585	0.7520	0.2580
10^{2}	0.7430	***0.2425***	0.7505	0.4630
10^{3}	0.7435	0.2505	0.7920	0.5790

During this first analysis, we focused on highlighting the difference between Linear and Quadratic Logistic Regression, and as we said before, the linear model performs way worse than the Quadratic one. In this analysis we also focused on analyzing the difference between the RAW dataset and the Z-Normalized one, noticing that the RAW dataset performs more regularly in terms of C_{prim}, even if the lowest minDCF is obtained using the Z-Normalized dataset. In the next analysis we will analyze the impact of PCA on the outcomes. In the next step we will consider only the better values of λ according to our C_{prim} metric, in this case $\{10^{-1}, 10, 10^{2}\}$ and just the quadratic version of the model, which is the best one for our application.

Logistic Regression		$\pi = 0.1$	
	λ	**RAW**	**Z-Norm**
No PCA	10	0.397	0.614
	10^{-1}	0.398	0.466
	10^{2}	0.393	0.705
PCA $(m = 5)$	10	0.414	0.624
	10^{-1}	0.372	0.475
	10^{2}	0.512	0.708
		$\pi = 0.5$	
No PCA	10	*0.098*	0.189
	10^{-1}	*0.096*	0.142
	10^{2}	*0.099*	0.221
PCA $(m = 5)$	10	0.115	0.19
	10^{-1}	0.105	0.139
	10^{2}	0.131	0.223
		$\pi = 0.9$	
No PCA	10	0.144	0.305
	10^{-1}	0.14	0.251
	10^{2}	0.141	0.335
PCA $(m = 5)$	10	0.221	0.304
	10^{-1}	0.155	0.255
	10^{2}	0.243	0.336
		C_{prim}	
		RAW	**Z-Norm**
No PCA	10	0.2475	0.4015
	10^{-1}	*0.247*	0.304
	10^{2}	*0.246*	0.463
PCA $(m = 5)$	10	0.2645	0.407
	10^{-1}	*0.2385*	0.307
	10^{2}	0.3215	0.4655

From this second step we can see that PCA neither improves or worsens our results that much. The best results is actually obtained by using it and the values of λ that correspond to the top 3 results are the same three as before. We can notice that Z-norm doesn't produce good results so we will not use it in the next step, in which we will implement a prior-weighted version of the model to see if our results can still be improved. We will once again consider the same three values for λ and we will try both before and after applying PCA to our dataset.

Logistic Regression		**Unweighted**		
		$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
	λ			
No PCA	10	0.098	0.397	0.144
	10^{-1}	***0.096***	0.398	0.14
	10^2	0.099	0.393	0.141
PCA $(m = 5)$	10	0.115	0.414	0.221
	10^{-1}	0.105	0.372	0.155
	10^2	0.131	0.512	0.243

			Weighted	
			$\pi = 0.5$	
		$\pi_t = 0.5$	$\pi_t = 0.1$	$\pi_t = 0.9$
No PCA	10	0.101	***0.097***	0.098
	10^{-1}	0.098	***0.096***	0.104
	10^2	0.101	0.099	0.102
PCA $(m = 5)$	10	0.109	0.119	0.122
	10^{-1}	0.101	0.107	0.104
	10^2	0.134	0.132	0.154
			$\pi = 0.1$	
		$\pi_t = 0.5$	$\pi_t = 0.1$	$\pi_t = 0.9$
No PCA	10	0.392	0.407	0.417
	10^{-1}	0.397	0.4	0.415
	10^2	0.395	0.393	0.394
PCA $(m = 5)$	10	0.418	0.421	0.481
	10^{-1}	0.373	0.372	0.372
	10^2	0.522	0.519	0.59
			$\pi = 0.9$	
		$\pi_t = 0.5$	$\pi_t = 0.1$	$\pi_t = 0.9$
No PCA	10	0.135	0.151	0.13
	10^{-1}	0.128	0.147	0.133
	10^2	0.138	0.141	0.142
PCA $(m = 5)$	10	0.21	0.223	0.207
	10^{-1}	0.151	0.165	0.172
	10^2	0.234	0.238	0.221

			C_{prim}		
		Unweighted		**Weighted**	
			$\pi_t = 0.5$	$\pi_t = 0.1$	$\pi_t = 0.9$
No PCA	10	0.2475	0.2465	0.252	0.2575
	10^{-1}	0.247	0.2475	0.248	0.2595
	10^2	0.246	0.248	0.246	0.248
PCA $(m = 5)$	10	0.2645	0.2635	0.27	0.3015
	10^{-1}	***0.2385***	***0.237***	0.2395	***0.238***
	10^2	0.3215	0.328	0.3255	0.372

3.3.3 Considerations

Analyzing the C_{prim} metric we can see that there are many results which are really close among each other, the best ones in terms of raw numbers are obtained by using PCA $(m = 5)$,

$\lambda = 10^{-1}$ and either an unweighted version of the model or a weighted version with an effective prior $\pi_t = 0.9$.

3.4 Support Vector Machine

3.4.1 Introduction

For our analysis, we decided to implement different versions of SVM, in particular:

1. Linear SVM : Obtained by solving the primal problem:

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

   where $H_{ij} = z_i z_j \; k(x_i, x_j), \qquad k(x_1, x_2) = x_1^T x_2$

2. Polynomial SVM : Obtained by solving the dual problem. To solve it we need to use a Kernel Function that allows us to compute dot products in an expanded space. The kernel we used is the polynomial kernel of degree 2 and 3, which is:

$$k(x_1, x_2) = (x_1^T x_2 + 1)^d$$

3. RBF SVM : Obtained by solving the same aforementioned dual problem, but this time using the Gaussian Radial Basis Kernel Function, defined as:

$$k(x_1, x_2) = e^{-\gamma||x_1 - x_2||^2}$$

3.4.2 Expectations

By analyzing both previous results and the shape of our dataset, we can safely assume that Linear SVM will perform poorly in our analysis. Polynomial SVM of degree 2 should be able to separate our dataset really well. RBF SVM should also be able to perform good classification on our dataset, based on its shape.
Following the same approach used for Logistic Regression, we will divide the analysis in subsequents steps, considering each time the best results obtained in the previous step(s) and trying to improve them.

3.4.3 Results

Linear SVM

SVM - Linear			RAW			C_{prim}
			$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$	
No PCA	$C = 1$	$K = 1$	1	0.84	0.914	0.920
		$K = 10$	1	0.846	0.888	0.923
	$C = 10$	$K = 1$	1	0.896	0.955	0.948
		$K = 10$	1	0.844	0.892	0.922
	$C = 10^{-2}$	$K = 1$	1	0.786	0.951	***0.893***
		$K = 10$	1	0.967	0.971	0.984
	$C = 10^{-4}$	$K = 1$	1	***0.482***	*0.513*	***0.741***
		$K = 10$	1	0.81	0.915	0.905
PCA $(m = 5)$	$C = 1$	$K = 1$	1	0.928	0.958	0.964
		$K = 10$	1	0.91	0.93	0.955
	$C = 10$	$K = 1$	1	0.929	0.953	0.965
		$K = 10$	1	0.957	0.975	0.979
	$C = 10^{-2}$	$K = 1$	1	0.832	0.873	0.916
		$K = 10$	1	0.872	0.968	0.936
	$C = 10^{-4}$	$K = 1$	1	***0.523***	0.551	*0.762*
		$K = 10$	1	0.942	0.98	0.971

In this step we focused on varying the C and K parameters to find the best combinations, but since the results are already pretty bad, we will not perform any subsequent steps using Linear SVM, because, as expected, it works really poorly and doesn't separate our dataset well.

Polynomial SVM

We will split the Polynomial SVM analysis into 2 steps, in the first one we will set $k = 1$ and $C = 10^{-1}$ and analyze the results as we vary c and d.

SVM - Polynomial ($k = 1$)	**RAW**						C_{prim}	
	π **= 0.1**		π **= 0.5**		π **= 0.9**			
	$d = 2$	$d = 3$	$d = 2$	$d = 3$	$d = 2$	$d = 3$	$d = 2$	$d = 3$
$c = 0$	0.434	1	***0.096***	0.587	0.189	0.684	***0.265***	0.794
$c = 1$	0.412	1	*0.097*	0.494	0.183	0.631	***0.255***	0.747
$c = 10$	0.422	1	***0.101***	0.45	0.166	0.565	*0.262*	0.725

We can immediately see that $d = 3$ doesn't work well at all on our dataset, so we will not use it in the next step of the analysis. In the other hand, $d = 2$ performs really well, which means that polynomials of degree 2 manage to separate our dataset in a good way. Now analyzing the values of c we can see that the parameter doesn't impact our results in a major way, so in the next step we will just consider the best value of c we got, which is $c = 1$ and calculate the impact of PCA and Z-normalization on our results.

SVM - Polynomial ($c = 1, d = 2$)	**RAW**		
	$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$
No PCA	0.412	*0.097*	0.183
PCA ($m = 5$)	0.391	***0.092***	0.159
	Z-Norm		
	$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$
No PCA	0.413	0.108	0.183
PCA ($m = 5$)	0.406	***0.105***	0.176

	C_{prim}	
	RAW	**Z-Norm**
No PCA	*0.255*	0.261
PCA ($m = 5$)	***0.242***	***0.256***

As we can see from the table, PCA actually improves our overall result, according to our C_{prim} metric, and it gives the best results when it's not paired with Z-Normalization. This is also the best C_{prim} we got up until now, considering all the analyzed models.

RBF SVM

SVM - RBF (NO PCA NO Z-Norm)			RAW			
			$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$	C_{prim}
$C = 1$	$\gamma = 1$	$K = 1$	0.746	0.661	0.999	0.704
		$K = 10$	0.737	0.622	0.998	0.680
	$\gamma = 10^-3$	$K = 1$	0.436	0.104	0.21	0.270
		$K = 10$	0.431	***0.1***	0.207	***0.266***
$C = 10$	$\gamma = 1$	$K = 1$	0.689	0.563	0.998	0.626
		$K = 10$	0.689	0.563	0.998	0.626
	$\gamma = 10^-3$	$K = 1$	0.405	***0.088***	0.182	***0.247***
		$K = 10$	0.406	*0.091*	0.177	*0.249*
$C = 10^{-2}$	$\gamma = 1$	$K = 1$	0.741	0.618	0.998	0.680
		$K = 10$	0.724	0.605	0.998	0.665
	$\gamma = 10^-3$	$K = 1$	0.545	0.147	0.202	0.346
		$K = 10$	0.516	0.132	0.244	0.324
$C = 10^{-4}$	$\gamma = 1$	$K = 1$	0.886	0.414	0.998	0.650
		$K = 10$	0.731	0.619	0.998	0.675
	$\gamma = 10^-3$	$K = 1$	1	0.998	0.998	0.999
		$K = 10$	0.887	0.409	0.578	0.648

Analyzing the results from this first step, we can see that the best results are obtained when $\gamma = 10^{-3}$, so we will use just this value for the parameter in the following steps. We also discard values $C < 1$ because they're not promising too. In the next step we'll also take into consideration PCA and Z-Normalization to see their impact on the analysis.

SVM - RBF ($\gamma = 10^{-3}$)			RAW			
			$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$	C_{prim}
No PCA	$C = 1$	$K = 1$	0.436	0.104	0.21	0.270
		$K = 10$	0.431	0.1	0.207	0.266
	$C = 10$	$K = 1$	0.405	***0.088***	0.182	*0.247*
		$K = 10$	0.406	***0.091***	0.177	***0.249***
PCA ($m = 5$)	$C = 1$	$K = 1$	0.42	0.101	0.197	0.261
		$K = 10$	0.416	0.1	0.193	0.258
	$C = 10$	$K = 1$	0.398	***0.088***	0.186	***0.243***
		$K = 10$	0.414	0.093	0.172	0.254
			Z-Norm			
			$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$	C_{prim}
No PCA	$C = 1$	$K = 1$	1	0.824	0.933	0.912
		$K = 10$	0.924	0.641	0.831	0.783
	$C = 10$	$K = 1$	0.906	0.253	0.35	0.580
		$K = 10$	0.945	0.735	0.813	0.840
PCA ($m = 5$)	$C = 1$	$K = 1$	0.924	0.682	0.929	0.803
		$K = 10$	0.905	0.714	0.872	0.810
	$C = 10$	$K = 1$	0.591	0.166	0.327	0.379
		$K = 10$	1	0.807	0.92	0.904

While our analysis doesn't benefit from Z-Normalization, the same thing can't be said about PCA. In fact, the best results are obtained after applying it to the dataset.

3.4.4 Final Considerations

After analyzing the three different types of SVM, we can say that Polynomial SVM gave us the best result ($\mathbf{C_{prim} = 0.242}$) with this combination of parameters:

$$(k = 1,\ c = 1,\ d = 2,\ \text{NO Z-Norm},\ \text{PCA}\ (m = 5))$$

Since the difference between the best Polynomial result and the best RBF result is negligible (0.001) we will consider both those models to perform score calibration and, subsequently, evaluation on the test set.

3.5 Gaussian Mixture Models

3.5.1 Introduction

The last model we will consider is GMM. It allows us to approximate generic distributions, so we expect better results with respect to the MVG classifier. We will need to tune an hyperparameter, which is the number of Gaussian Components, C.

3.5.2 Expectations

As we said beforehand, we expect this classifier to perform better than MVG because it can approximate generic distributions.

3.5.3 Results

GMM		**RAW**			**Z-Norm**			C_{prim}	
		$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$	$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$	**RAW**	**Z-Norm**
Diagonal GMM	1	0.134	0.53	0.179	0.138	0.53	0.178	0.332	0.334
	2	0.117	0.45	0.195	0.129	0.471	0.205	0.2835	0.3
	3	***0.11***	0.441	0.183	0.117	0.437	0.176	0.2755	0.277
	4	*0.102*	0.423	0.17	***0.098***	0.437	0.177	***0.2625***	*0.2675*
Standard GMM	1	0.123	0.506	0.2	0.14	0.511	0.191	0.3145	0.3255
	2	0.141	0.512	0.251	0.12	0.497	0.188	0.3265	0.3085
	3	0.146	0.478	0.228	0.118	0.504	0.215	0.312	0.311
	4	0.165	0.518	0.239	0.138	0.484	0.23	0.3415	0.311
Tied GMM	1	0.121	0.489	0.166	0.121	0.489	0.166	0.305	0.305
	2	0.115	0.429	0.204	0.119	0.478	0.155	***0.272***	0.2985
	3	0.121	0.463	0.2	0.118	0.462	0.201	0.292	0.29
	4	***0.11***	0.517	0.212	0.112	0.457	0.194	0.3135	0.2845

In this first analysis we focused on the analysis of the number of components of the GMM, to see which one performs better, we performed this analysis both on the RAW dataset and on the Z-Normalized one. We can see that increasing the number of components of the diagonal GMM, results keep getting better up until 2^4 components, so we will keep this in mind and try to increase it even more later to see if there's still room for improvement.
Standard GMM doesn't perform well at all, so we won't consider it in the following step(s). On the other hand, we will consider Tied GMM because it gives promising results up until now. The next step will be focused on analyzing the impact of PCA on our results, both with and without performing Z-Normalization on our dataset.

GMM		#	RAW		
			$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$
No PCA	**Diagonal GMM**	2	0.117	0.45	0.195
		3	0.11	0.441	0.183
		4	*0.102*	0.423	0.17
	Tied GMM	2	0.115	0.429	0.204
		3	0.121	0.463	0.2
PCA $(m = 5)$	**Diagonal GMM**	2	0.127	0.42	0.17
		3	0.125	0.498	0.19
		4	0.131	0.527	0.195
	Tied GMM	2	0.118	0.436	0.186
		3	0.118	0.476	0.186
			Z-Norm		
			$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$
No PCA	**Diagonal GMM**	2	0.129	0.471	0.205
		3	0.117	0.437	0.176
		4	*0.098*	0.437	0.177
	Tied GMM	2	0.119	0.478	0.155
		3	0.118	0.462	0.201
PCA $(m = 5)$	**Diagonal GMM**	2	0.119	0.493	0.225
		3	0.115	0.493	0.214
		4	0.116	0.48	0.251
	Tied GMM	2	0.117	0.495	0.174
		3	*0.108*	0.488	0.18
			C_{prim}		
			RAW	**Z-Norm**	
No PCA	**Diagonal GMM**	2	0.2835	0.3	
		3	0.2755	0.277	
		4	*0.2625*	*0.2675*	
	Tied GMM	2	*0.272*	0.2985	
		3	0.292	0.29	
PCA $(m = 5)$	**Diagonal GMM**	2	0.2735	0.306	
		3	0.3115	0.304	
		4	0.329	0.298	
	Tied GMM	2	0.277	0.306	
		3	0.297	0.298	

Analyzing our results, we can see that PCA doesn't bring any improvement whatsoever, both with and without the application of Z-Normalization. In this step we considered just the best models from before, which are Diagonal GMM with $2^2, 2^3, 2^4$ components and Tied GMM with $2^2, 2^3$ components.

The next step will focus on increasing the components of GMM to see if the results keep getting better. We'll also consider using PCA to see if it's worth using with more components. Also Z-normalization will still be used.

GMM		#	RAW		
			$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$
No PCA	**Diagonal GMM**	5	*0.105*	0.42	0.2
	Tied GMM	5	0.11	0.434	0.193
PCA $(m = 5)$	**Diagonal GMM**	5	0.127	0.502	0.218
	Tied GMM	5	0.119	0.447	0.164
			Z-Norm		
			$\pi = 0.1$	$\pi = 0.5$	$\pi = 0.9$
No PCA	**Diagonal GMM**	5	*0.109*	0.412	0.204
	Tied GMM	5	*0.104*	0.434	0.196
PCA $(m = 5)$	**Diagonal GMM**	5	0.11	0.45	0.208
	Tied GMM	5	*0.109*	0.435	0.23
			C_{prim}		
			RAW	**Z-Norm**	
No PCA	**Diagonal GMM**	5	*0.2625*	*0.2605*	
	Tied GMM	5	0.272	*0.269*	
PCA $(m = 5)$	**Diagonal GMM**	5	0.3145	0.28	
	Tied GMM	5	0.283	0.272	

Analyzing the results from this third step we can see that results kept improving even after increasing the number of components. We acheived the best result with a Diagonal GMM with 2^5 components, without applying PCA and with the Z-Normalized dataset.

3.6 Wrapping Up

We analyzed different models and the best ones that we found are the following:

Uncalibrated Results	$\pi = 0.1$		$\pi = 0.5$		$\pi = 0.9$		C_{prim}	
	minDCF	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**
MVG Standard (NO PCA NO Z-Norm)	0.4970	0.546	0.1250	*0.125*	0.1890	0.189	0.3110	*0.3357*
Quadratic Prior Weighted Logistic Regression (PCA $m = 5$, $\lambda = 10^{-1}, \pi_t = 0.5$) NO Z-Norm	0.370	0.400	0.106	*0.108*	0.152	0.164	0.2380	*0.2540*
Polynomial SVM (PCA $m = 5$, $d = 2$, c = 1, k = 1, C = 10^{-1}) NO Z-Norm	0.391	0.833	0.092	0.162	0.159	0.187	0.2415	0.4975
Diagonal GMM (2^5 components, NO PCA, Z-Norm)	0.109	*0.144*	0.412	0.423	0.204	0.316	0.2605	*0.2835*
RBF SVM (PCA $m = 5$, NO Z-Norm, $\gamma = 10^{-3}, C = 10, k = 1$)	0.398	0.818	0.088	0.154	0.186	0.415	0.2430	0.4860

In this table the color scheme refers just to the actDCF

We will now represent the Bayes Error Plots for our best models, to check if our results are already well calibrated or we need to perform score calibration on them.

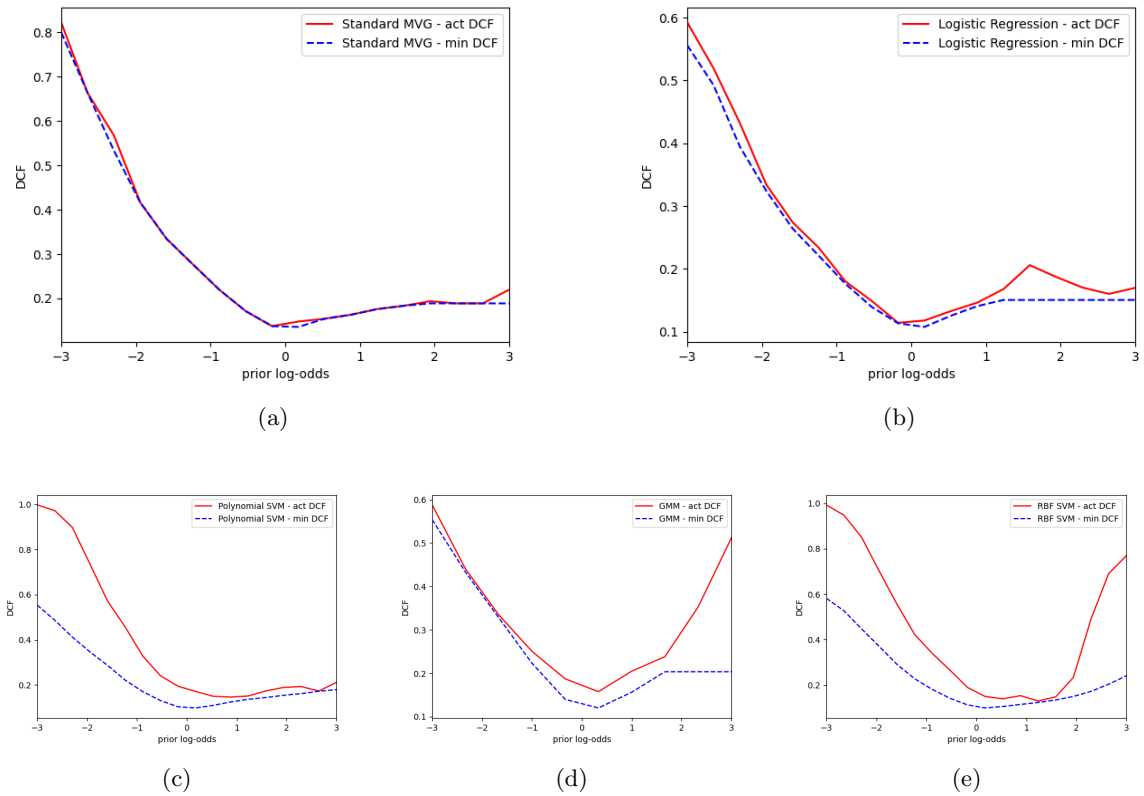


Figure 3.1: (a) MVG (b) LogReg (c) Polynomial SVM (d) GMM (e) RBF SVM.

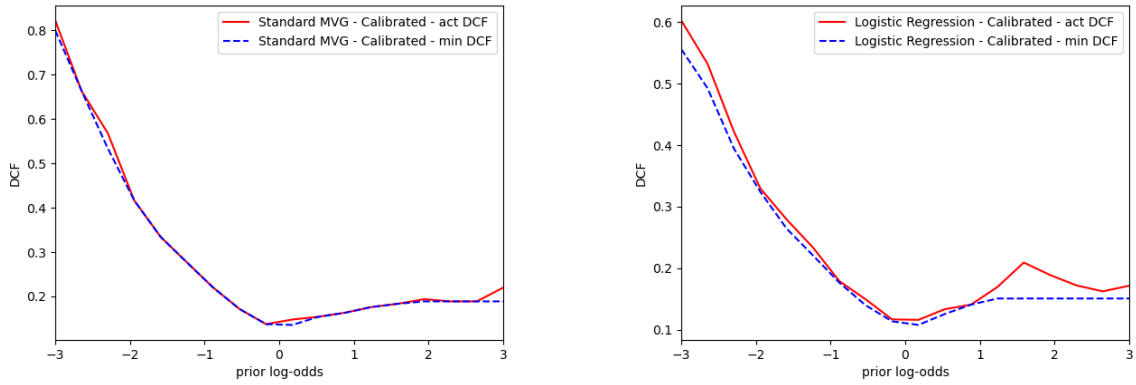
Looking at these Bayes Error Plots we can safely assume that the first two models, MVG and LogReg, don't need score calibration, because they're already well calibrated. In the other hand, Polynomial SVM, GMM and RBF SVM, may benefit from score calibration, which we will perform in the next section.

3.7 Score Calibration

3.7.1 Introduction

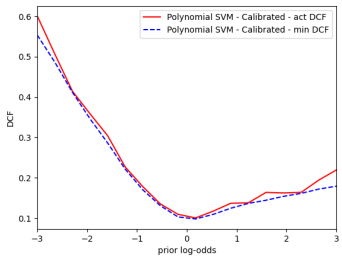
Since the minimum detection cost function ($minDCF$) we used so far to evaluate the models depends on a variable threshold, We now introduce a new metric, Actual DCF, which will use the theoretical threshold.
We will implement Score Calibration through the use of a Prior Weighted Logistic Regression model trained on the scores of our models, using as parameters: $\lambda = 10^{-4}, \pi_T = 0.5$, since we are looking for a linear function of the type $f(s) = \alpha s + \gamma - \log \frac{\pi_T}{1 - \pi_T}$

3.7.2 Calibration

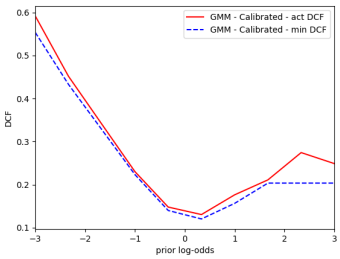


(a)

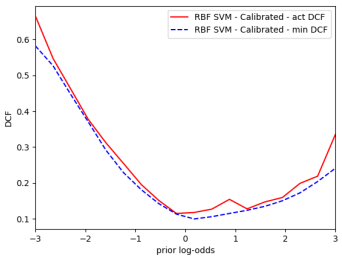
(b)



(c)



(d)



(e)

Figure 3.2: (a) MVG (b) LogReg (c) Polynomial SVM (d) GMM (e) RBF SVM.

Analyzing the plots after performing score calibration, we can see that, as we expected, MVG and LogReg didn't benefit at all from it. In the other hand, GMM, Polynomial SVM and RBF SVM improved a lot.

3.7.3 Results

Calibrated Results	$\pi = 0.1$		$\pi = 0.5$		$\pi = 0.9$		C_{prim}	
	minDCF	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**
MVG Standard (NO PCA NO Z-Norm)	0.497	0.546	0.125	0.125	0.189	0.204	0.311	0.336
Quadratic Prior Weighted Logistic Regression (PCA $m = 5$, $\lambda = 10^{-1}, \pi_t = 0.5$) NO Z-Norm	0.373	0.393	0.101	***0.108***	0.152	0.167	0.237	***0.251***
Polynomial SVM (PCA $m = 5$, $d = 2$, c = 1, k = 1, C = 10^{-1}) NO Z-Norm	0.584	0.591	0.162	0.164	0.274	0.310	0.373	0.378
Diagonal GMM (2^5 components, NO PCA, Z-Norm)	0.412	0.437	0.109	***0.112***	0.203	0.255	0.261	***0.275***
RBF SVM (PCA $m = 5$, NO Z-Norm, $\gamma = 10^{-3}, C = 10, k = 1$)	0.427	0.436	0.097	***0.108***	0.165	0.188	0.262	*0.272*

In this table the color scheme refers just to the actDCF

As we expected, calibration had no impact on MVG and LogReg classifiers. The other three classifiers benefitted majorly from it. In the Evalutation phase we will consider all the five models nonetheless, because the evaluation set may differ slightly from the training one, so results may vary.

Chapter 4

Evaluation

4.1 Introduction

We will now train the best models that we got so far, which are:

- Standard MVG, without PCA and without Z-Normalization
- Quadratic Prior Weighted Logistic Regression, without Z-Normalization and with PCA
- Polynomial SVM, without Z-Normalization and with PCA
- Diagonal GMM, without PCA and with Z-Normalization
- RBF SVM, without Z-Normalization and with PCA

We will train them on our whole training dataset, instead of using K-Fold Cross Validation and we will evaluate the results we get on the Evaluation dataset we have.

4.1.1 Results

EVALUATION - UNCALIBRATED	π **= 0.1**		π **= 0.5**		π **= 0.9**		C_{prim}	
	minDCF	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**
MVG Standard (NO PCA NO Z-Norm)	0.522	0.522	0.153	*0.155*	0.193	0.198	0.338	*0.339*
Quadratic Prior Weighted Logistic Regression (PCA $m = 5$**,** $\lambda = 10^{-1}, \pi_t = 0.5$**) NO Z-Norm**	0.449	0.464	0.133	*0.141*	0.208	0.260	0.291	*0.303*
Polynomial SVM (PCA $m = 5$ **,** $d = 2$**, c = 1, k = 1, C =** 10^{-1}**) NO Z-Norm**	0.495	0.905	0.139	0.219	0.234	0.265	0.317	0.562
Diagonal GMM (2^5 **components, NO PCA, Z-Norm)**	0.465	0.471	0.153	*0.177*	0.283	0.605	0.309	*0.324*
RBF SVM (PCA $m = 5$**, NO Z-Norm,** $\gamma = 10^{-3}, C = 10, k = 1$**)**	0.522	0.910	0.175	0.221	0.477	0.690	0.349	0.566

Analyzing the most promising models from the previous part gave us these results. We will now plot BEPs and ROC curves for them to see if they need calibration or not, and in case perform it afterwards.

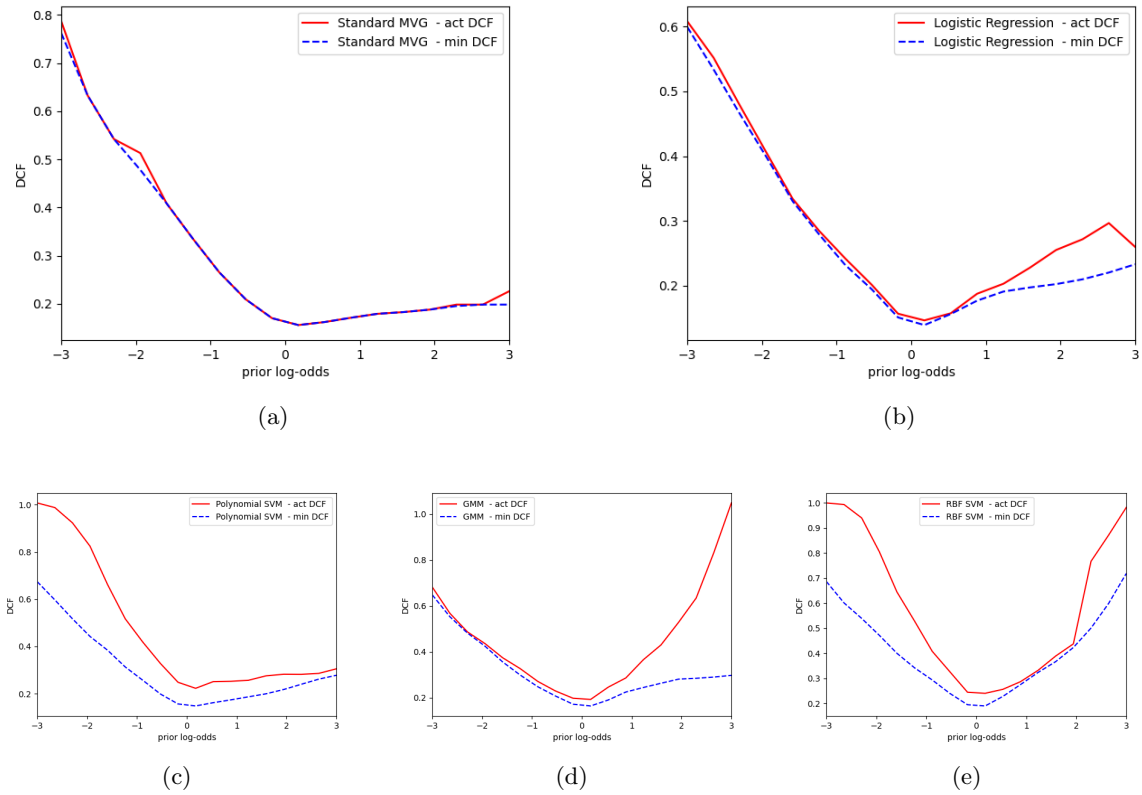


Figure 4.1: Uncalibrated (a) MVG (b) LogReg (c) Polynomial SVM (d) GMM (e) RBF SVM.

Analyzing the BEPs we can see that, like before, MVG and LogReg don't need calibration, while the other three models do, so the next step will be performing score calibration on our results.

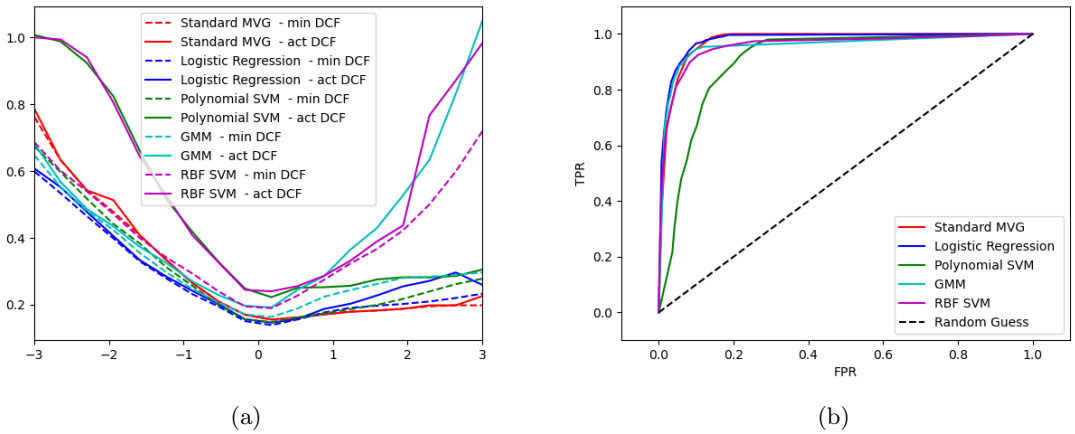


Figure 4.2: Uncalibrated (a) Comparison between BEP (b) Comparison between ROC curves

EVALUATION - CALIBRATED	$\pi = 0.1$		$\pi = 0.5$		$\pi = 0.9$		C_{prim}	
	minDCF	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**
MVG Standard (NO PCA NO Z-Norm)	0.522	0.522	0.154	***0.155***	0.193	0.198	0.338	0.339
Quadratic Prior Weighted Logistic Regression (PCA $m = 5$, $\lambda = 10^{-1}, \pi_t = 0.5$) NO Z-Norm	0.449	0.472	0.133	***0.134***	0.208	0.235	0.291	***0.303***
Polynomial SVM (PCA $m = 5$, $d = 2$, c = 1, k = 1, C = 10^{-1}) NO Z-Norm	0.495	0.495	0.139	*0.144*	0.234	0.243	0.317	***0.320***
Diagonal GMM (2^5 components, NO PCA, Z-Norm)	0.465	0.467	0.153	0.160	0.283	0.312	0.309	*0.314*
RBF SVM (PCA $m = 5$, NO Z-Norm, $\gamma = 10^{-3}, C = 10, k = 1$)	0.522	0.530	0.175	0.182	0.477	0.490	0.349	0.356

We performed calibration on all the five models and, as we expected, MVG and LogReg didn't benefit at all from it, while the other models did, especially the two SVM models.

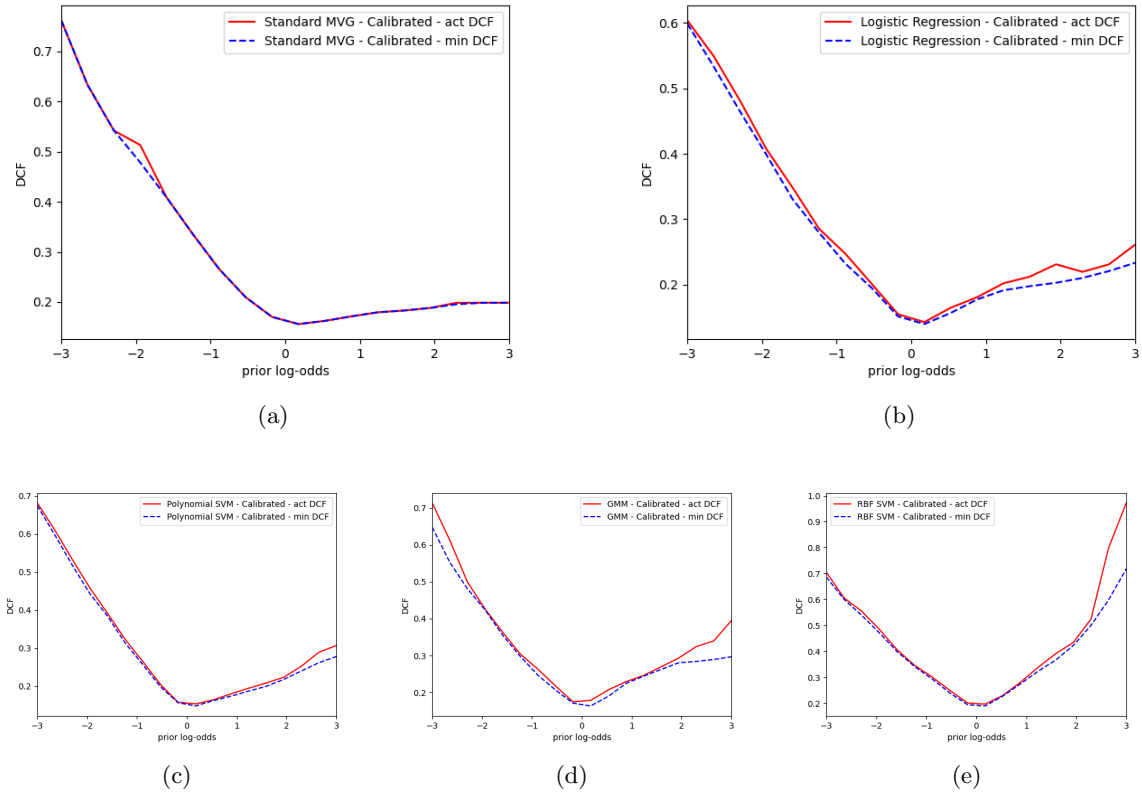
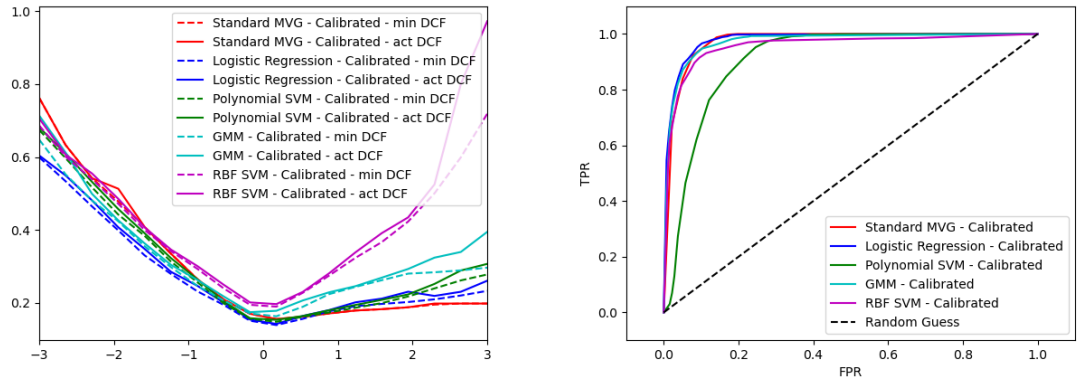


Figure 4.3: Calibrated (a) MVG (b) LogReg (c) Polynomial SVM (d) GMM (e) RBF SVM.



(a) (b)

Figure 4.4: Calibrated (a) Comparison between BEP (b) Comparison between ROC curves

4.2 Final Considerations

Talking about raw numbers, the model we found out to be the best is Prior Weighted Quadratic Logistic Regression, while the worst is Polynomial SVM, as we can also see from its ROC curve. The best four models we found are all really close in terms of actDCF and ROC curve, so they should be all taken into consideration when considering the average of the application points given to us (C_{prim}). In terms of single application point, we have to admit that, due to having an heavy unbalanced dataset, it's complicated to achieve good results when considering $\pi = 0.1$.