WINE QUALITY DETECTION REPORT

D'ASCENZI GIULIA S287720, DE GIROLAMO PATRIZIO S292497

Report for the "Machine Learning and Pattern Recognition" course

Academic year 2020-2021

1 INTRODUCTION

The dataset used for the purpose of this analysis is related to red and white variants of the Portuguese "Vinho Verde" wine and is taken from the UCI repository.

The dataset was collected to predict human wine taste preferences. As a matter of fact, it associates the physiochemical characteristics of the analyzed wine to sensory evaluations made by experts.

The original dataset consists of 10 classes (quality 1 to 10), but for this project the dataset has been binarized, collecting all wines with low quality (lower than 6) into class 0 and good quality (greater than 6) into class 0. Wines with quality 6 have been discarded to simplify the task.

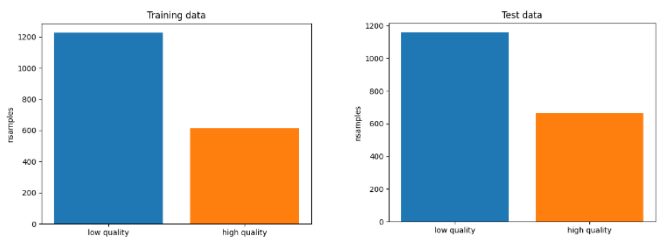
The dataset contains both red and white wines that were originally separated, whereas in this analysis they have been merged.

1.1 *Classes Balance*

The dataset has been divided into a training set and a test set. The training set contains 613 samples belonging to the "high quality" class and 1226 belonging to the "low quality" class. On the other hand, the evaluation set contains 664 samples of class "high quality" and 1158 samples of class "low quality". Therefore, the classes are partially balanced.

The following graphs show the comparison between the number of samples of the two classes in both training and test data set. It emerges that both the number of samples and the ratio between the two classes in the different datasets is similar.

Figure 1:



1.2 *Attributes Information*

The input variables have 11 continuous features based on physiochemical tests: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol.

The output variable is a discrete value representing the quality of the wine sample (0 low quality/1 high quality)

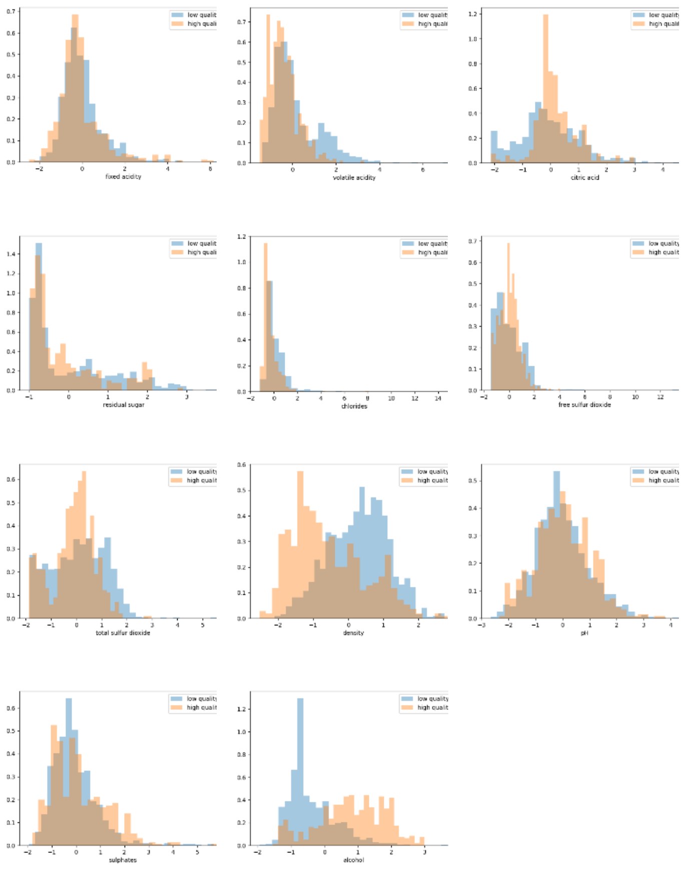
1.3 *Preprocessing and feature analysis*

The features of the dataset refer to different types of variables and have therefore different measuring scales.

Thus, in order to compare similarities between features, the dataset has been z-normalized through centering and scaling to unit variance the features.

The histograms below show the distributions of the training dataset features. Orange histograms refer to high quality wines, whereas blue histograms to low quality wines.

Figure 2:



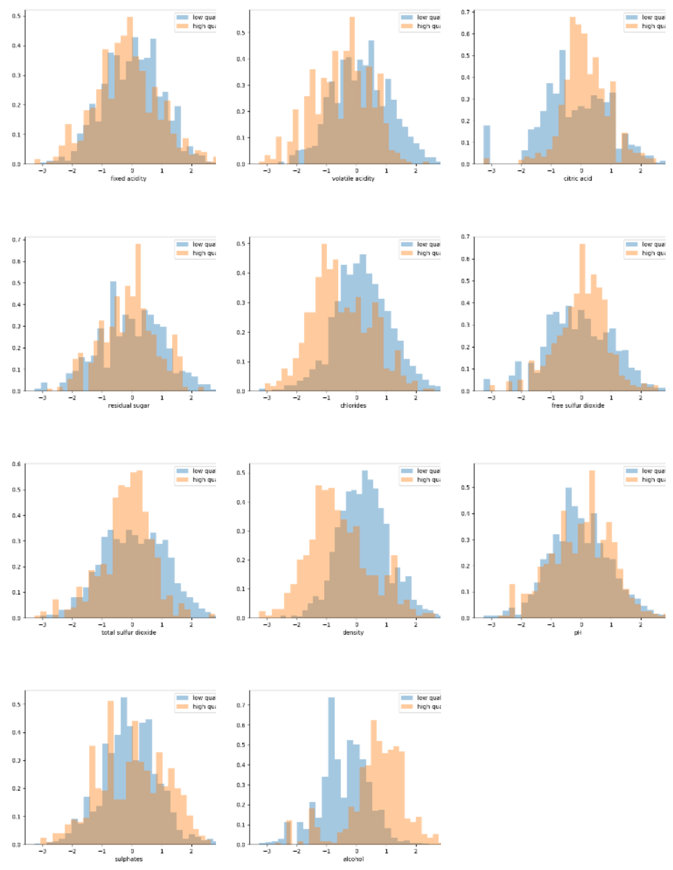
The analysis of the training data reveals that most of the features have an irregular distribution.

Consequently, the classification (especially of Gaussian based methods) may produce sub-optimal results. We therefore further preprocessed our data by "Gaussianizing" the features.

The Gaussianization process allows mapping the features values to ones whose empirical cumulative distribution function is well approximated by a Gaussian c.d.f. To do so, the features have been mapped to a uniform distribution and then transformed through the inverse of Gaussian cumulative distribution function.

The histograms below show the distributions of the Gaussianized features.

Figure 3:



A correlation analysis of the Gaussianized features shows that feature 5 and 6 are strongly correlated.

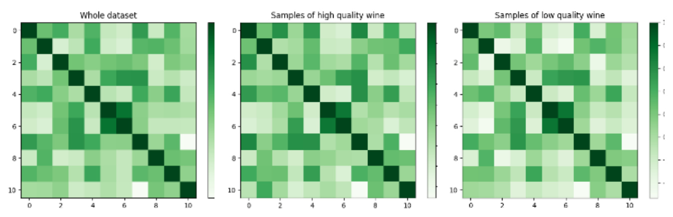
The heatmaps showing the Pearson correlation coefficient

$$\frac{\mathrm{Cov(X,Y)}}{\sqrt{\mathrm{Var(X)}}\sqrt{Var(Y)}}$$

can be found below.

From right to left: whole dataset, high quality features, low quality features.

Figure 4:



This suggests that the classification may benefit from using PCA to map data to 10 or 9 uncorrelated features to reduce the number of parameters to estimate.

2 CLASSIFICATION OF WINE QUALITY FEATURES

2.1 Methodologies used for validation

In order to understand which model is the most promising and to assess the effects of using PCA, both single fold-validation and K-Fold cross-validation have been adopted.

At first, a single fold approach has been used, since the training with this approach is faster. Indeed, single fold approach consists in splitting the training dataset into two subsets, one (66% of the original set) for development and the other one for validation. Subsequently, the K-Fold approach has been used to get more robust results. In this case, the training set are iteratively split into 5 folds, 4 used for training and 1 for validation, after being shuffled. The validation scores have been eventually put together and used to compute the performance metrics. In this way, there is more data available for training and validation.

This document focuses on the analysis of the balanced uniform prior application:

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

However, two other unbalanced applications have been considered:

$(\pi, C_{fp}, C_{fn}) = (0.9,1,1)$ and $(\pi, C_{fp}, C_{fn}) = (0.1,1,1)$

In the first part, the main aim of the analysis was to choose the most promising approach. Therefore, the performances have been measured in terms of normalized minimum detection costs, namely the costs that would be paid by making optimal decisions for the validation set through the use of recognizers scores.

2.2 MVG CLASSIFIERS

The first classifier analyzed is the multivariate gaussian classifier, which assumes that the data, given the class, can be described by a Gaussian distribution.

The following table shows the results in terms of minimum DCF of the studied Gaussian classifiers.

Table 1:

	Single Fold			5-Fold		
	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.9$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.9$	$\tilde{\pi} = 0.1$
Raw Features – no PCA						
Full - Cov	0.304	0.812	0.777	0.312	0.842	0.778
Diag -Cov	0.437	0.875	0.818	0.420	0.921	0.845
Tied Full-Cov	0.334	0.733	0.779	0.333	0.748	0.812
Tied Diag- Cov	0.412	0.901	0.832	0.402	0.932	0.866
Gaussianized Features – no PCA						
Full - Cov	0.270	0.740	0.807	0.306	0.790	0.784
Diag -Cov	0.456	0.848	0.860	0.448	0.914	0.834
Tied Full-Cov	0.348	0.812	0.867	0.354	0.884	0.803
Tied Diag- Cov	0.451	0.891	0.839	0.451	0.942	0.879
Gaussianized Features – PCA (m=10)						
Full - Cov	0.312	0.817	0.779	0.328	0.862	0.807
Diag -Cov	0.362	0.747	0.780	0.378	0.824	0.804
Tied Full-Cov	0.324	0.682	0.791	0.328	0.753	0.809
Tied Diag- Cov	0.335	0.656	0.822	0.334	0.782	0.822
Gaussianized Features – PCA (m=9)						
Full - Cov	0.300	0.817	0.783	0.319	0.814	0.800
Diag -Cov	0.349	0.764	0.810	0.368	0.814	0.804
Tied Full-Cov	0.331	0.677	0.786	0.327	0.752	0.816
Tied Diag- Cov	0.337	0.673	0.822	0.335	0.783	0.825

The diagonal covariance models (both full diagonal and tied diagonal) do not give good results if compared to the other models with or without Gaussianization and PCA. These models work under the naive-bayes-assumption, according to which the different components for each class are uncorrelated. Therefore, in this case, this assumption does not produce accurate results. Applying PCA slightly improves the performance, probably because the within-class correlation decreases by removing the low variances directions.

The tied full covariance model performs pretty good on the validation data, confirming the similarity between classes shown by the correlation analysis. It performs better than the diagonal models accounting the within-class correlations. In this case, PCA does not help in producing better results.

The best-performing model is the full covariance model, which is able to account for correlations. What is more, having enough data compared to the dimensionality of the samples, we succeed in obtaining robust results.

Gaussianization fails at significantly improving the results, although it still performs slightly better if compared to raw features.

Even if it does not help to increase much the performances, PCA can still be used to reduce the number of parameters and therefore to reduce the complexity of the model. Nevertheless, given the limited effectiveness, it will not be considered for the features analysis.

The results between single-fold and K-fold are consistent, suggesting that the amount of data is enough for validation and model training also when it comes to the single-fold set up.

None of the models produces accurate results for the unbalanced applications.

In summary, the best candidate is currently the MVG model with full covariance matrices. The chosen one is the one with Gaussianized features, since it shows slightly better results than the one with raw features, and the K-fold approach, even if it is a bit worse than the single fold result. Nonetheless, this approach provides more robust results.

Since the best-performing models are the full covariance and the tied full models (which respectively have a quadratic surface rule and a linear surface rule), the decision was to proceed by analyzing both the quadratic and the linear models.

2.3 *LOGISTIC REGRESSION*

Up to now, the focus of the analysis was only on generative models, but now it turns on the discriminative models. The first one is the linear logistic regression.

2.3.1 *Linear Logistic Regression*

Linear logistic regression is a discriminative model and therefore models the class posterior distribution, rather than the distribution of the observed samples X|C.

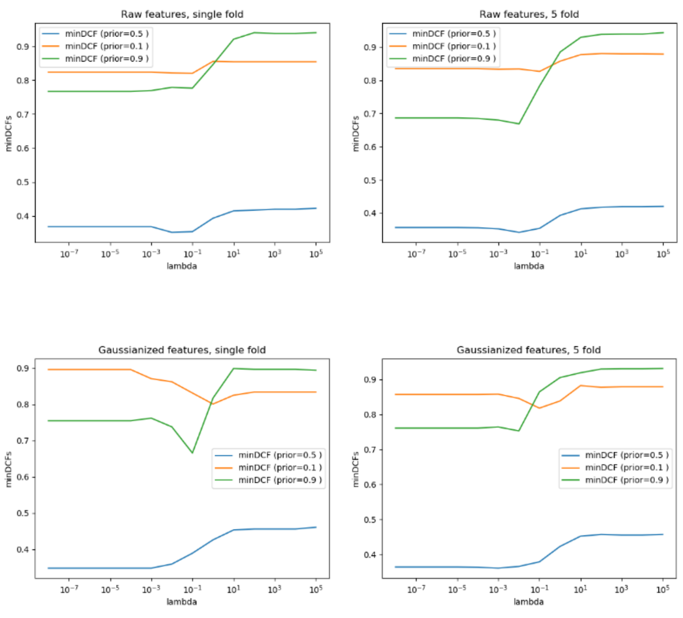
To do so, logistic regression looks for the linear hyperplanes that maximizes the likelihood of the training labels. This also corresponds to both the minimization of the average cross entropy between the empirical distribution of the data and the predicted labels distribution and to the minimization of the empirical risk of mis-classifying the data.

Since the classes of this dataset are not perfectly balanced and the logistic regression model automatically embeds the prior empirical probability, the original model has been modified to take in account the different weights of the cost loss due to the samples of each class, in order to reflect the prior of our applications.

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

The first step was the tuning of the hyperparameter λ. The following graphs show how the minDCF for the three chosen applications varies with respect to λ. (Left: single fold. Right: K-fold. Top: raw features. Bottom: Gaussianized features)

Figure 5:



Results with K-fold and single fold are similar. Consequently, the following analysis have been done using the K-fold approach, since the validation results are more reliable.

Regularization did not provide much benefit. Better results are achieved with values of λ less than 10^{-3}. In order to avoid overconfident models, that could be obtained with a too low λ value, but at the same time to get an optimal minDCF, the value chosen for λ was 10^{-3}.

Even in this case, the Gaussianization does not improve much the performance of the model. Indeed, logistic regression does not require assumptions on the data distribution.

In the following table are also reported the results achieved by using different prior π_T to see the effects on the other applications.

Table 2:

	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
Raw Features			
LR (λ= 10^{-3} , $\pi_T = 0.5$)	0.352	0.833	0.680
LR (λ= 10^{-3} , $\pi_T = 0.1$)	0.336	**0.818**	0.733
LR (λ= 10^{-3} , $\pi_T = 0.9$)	0.368	0.852	**0.653**
LR (λ= 10^{-3})	0.340	0.839	0.673
Gaussianized features			
LR (λ= 10^{-3} , $\pi_T = 0.5$)	0.360	0.858	0.764
LR (λ= 10^{-3} , $\pi_T = 0.1$)	0.340	**0.780**	0.933
LR (λ= 10^{-3} , $\pi_T = 0.9$)	0.376	0.898	**0.698**
LR (λ= 10^{-3})	0.359	0.829	0.836

Overall, the MVG model with full covariances performs better.

Furthermore, comparing these results with the best linear MVG classifier (Tied covariance model), the logistic regression performed slightly worse.

Since the full covariances MVG corresponds to quadratic separation rules, the analysis has been repeated with a quadratic logistic regression model.

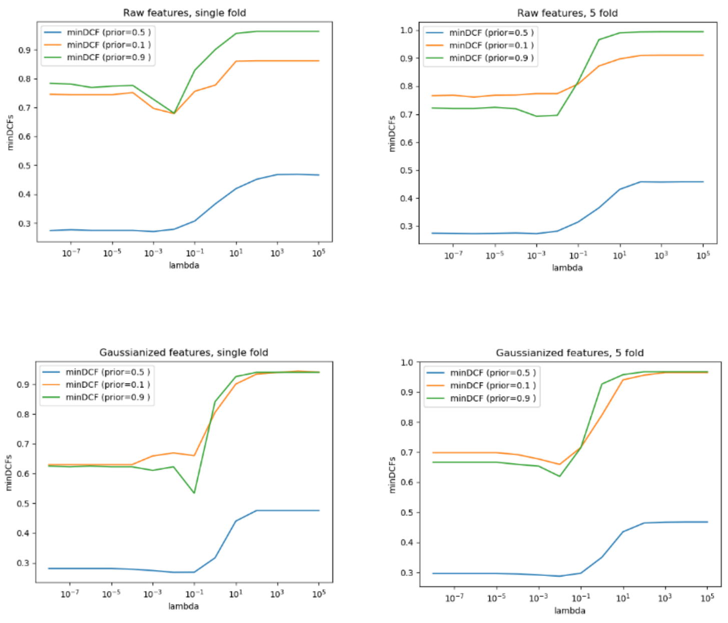
Class re-balancing helps for the unbalanced applications, whereas for the main application the best results are achieved with unbalanced classes, meaning that the class re-balancing was not necessary in this case.

2.3.2 Quadratic Logistic Regression

The following graphs show the minDCF for different values of λ.

(Top: raw features. Bottom: gaussianized features. Left: single fold. Right: K-fold)

Figure 6:



Also in this case, varying lambda does not produce improvement for lambda lower than 10^{-3}. Therefore, the regularized term has been set to 10^{-3} for the same reason as before.

In the following table are also reported the results achieved by using different prior π_T to see the effects on the other applications.

Table 3:

	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
Raw Features			
Quad LR (λ= 10^{-3} , $\pi_T = 0.5$)	0.275	0.771	0.692
Quad LR (λ= 10^{-3} , $\pi_T = 0.1$)	0.275	**0.752**	0.703
Quad LR (λ= 10^{-3} , $\pi_T = 0.9$)	0.287	0.807	**0.642**
Quad LR (λ= 10^{-3})	**0.274**	0.769	0.686
Gaussianized features			
Quad LR (λ= 10^{-3} , $\pi_T = 0.5$)	0.291	0.676	0.653
Quad LR (λ= 10^{-3} , $\pi_T = 0.1$)	0.292	0.700	0.644
Quad LR (λ= 10^{-3} , $\pi_T = 0.9$)	0.307	0.747	**0.625**
Quad LR (λ= 10^{-3})	0.288	0.676	0.637

The results of the quadratic logistic regression outperforms the MVG classifier results.

Again, Gaussianization does not improve the results with respect to the raw features results. However, even if the results of the unbalanced applications are still pretty poor, the quadratic logistic regression with Gaussianized features is so far the model that gives the best results for those two applications.

For all the applications, the re-balancing does not improve the performances.

2.4 SVM

The analysis continues with the Support Vector Machine model.

The SVM classifier is a discriminative classifier that aims at finding a separation hyperplane between two classes that have the maximum margin, namely the maximum distance between the hyperplane and the closest points. In the case in which the classes are not linearly separable, the followed approach is the "soft margin" approach, that looks for the highest margin hyperplane in order to minimize the points that lie inside the margin or that are on the wrong side of the classification rule (therefore misclassified points).

Again, a re-balanced version of the model has been considered to reflect the effective priors of the chosen applications.

$$\max_{\alpha} \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{H}\boldsymbol{\alpha}$$

Subjected to:

$$0 \le \alpha_i \le C_i \; i = 1, \dots, n$$

$$C_i = C_T \; for \; sample \; of \; class \; H_T$$

$$C_i = C_F \; for \; sample \; of \; class \; H_F$$

$C_T = C \frac{\pi_T}{\pi_T^{emp}}$ and $C_F = C \frac{\pi_F}{\pi_F^{emp}}$ have been selected, where π_T^{emp} and π_F^{emp} are the empirical priors (i.e sample proportions) for the two classes computed over training set.

2.4.1 Linear SVM

The linear SVM model has as hyperparameter C that is used to define a trade-off between having a solution with a wider margin and a solution with a lower number of points laying inside the margin.

Higher values for C correspond to solutions with less permitted points inside the margin and a thin margin. These solutions have a higher risk of being overconfident and not performing well on unseen data. On the other hand, a lower value of C means allowing to have more points inside the margin, a wider margin that provides a less confident solution that generalizes better.

The following graphs show how minDCF varies depending on different values of the hyperparameter C.

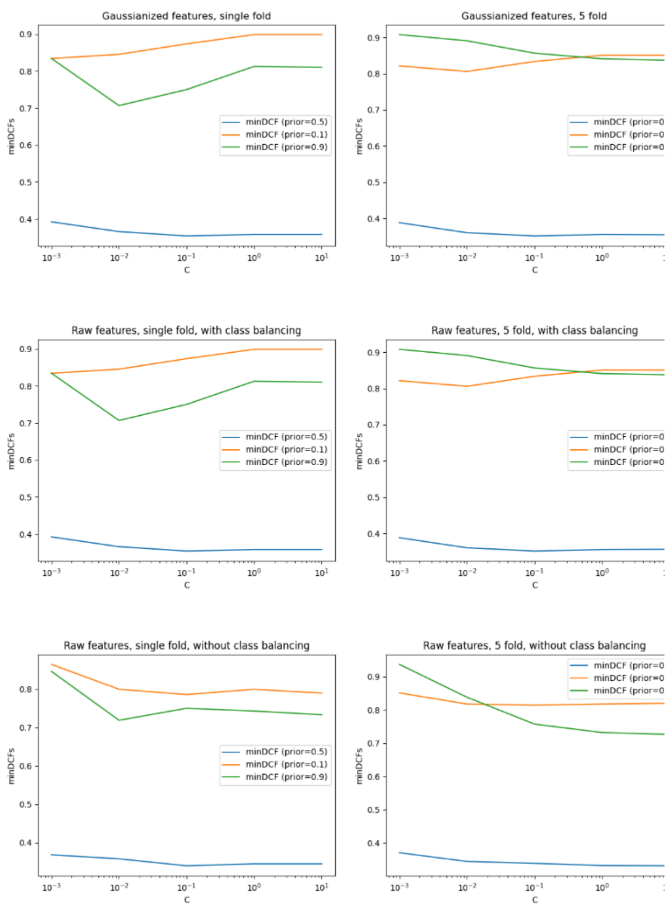
Right: single fold. Left: 5 fold.

First row: Gaussianized features, class balancing (pi_T= 0.5).

Second row: raw features, class balancing.

Third row: raw features, class without balancing.

Figure 7:



For the hyperparameter C the choice followed the same strategy used before: it has been set to 10^{-1}since it is the biggest value that assures good performances.

The graphs already show that re-balancing the model does not result in much higher performances for the target application of this analysis.

Furthermore, the Gaussianized preprocessing does not improve the performance, thus it has not been considered for the following analysis.

Table 4:

	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
Raw Features			
SVM (C=0.1 , $\pi_T = 0.5$)	0.339	0.849	0.668
SVM (C=0.1, $\pi_T = 0.1$)	0.892	0.995	0.996
SVM (C=0.1 , $\pi_T = 0.9$)	0.386	0.876	0.693
SVM (C=0.1)	0.338	0.817	0.761

Since the linear SVM is the last linear classifier analyzed, these results prove once again that the linear classifiers perform worse than quadratic classifiers in this dataset.

Table 5:

Comparison between the best linear models ($\tilde{\pi} = 0.5$)	**minDCF**
MVG Tied Full-Cov	*0.334*
SVM, C=0.1	0.338
Log reg, λ= 10^{-3}	0.340

All these three classifiers give similar results. Moreover, it c be noticed that the assumption of Gaussian distribution made by the tied full model does not worsen the performance if compared to models as SVM and LR, that do not make any assumption on the data distributions.

Additionally, noticing that the preprocessing step of Gaussianizing the features is useless most of the time, we can conclude that the distribution of this dataset can be sufficiently approximated to a Gaussian distribution even if the best results so far are still gotten with quadratic logistic regression that does not need assumptions of this type.

Therefore, the analysis proceeds with the quadratic version of the SVM.

2.4.2 SVM with quadratic kernel

The first non-linear SVM model used is the polynomial quadratic kernel, that is similar to the quadratic Logistic Regression model and we therefore expect similar results.

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

The chose model has degree d = 2 (quadratic), whereas c is a hyper-parameter selected through cross-validation.

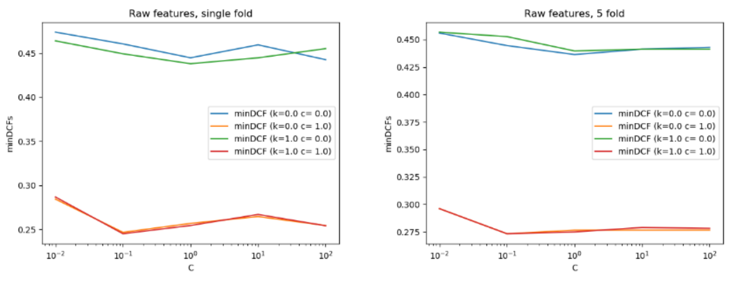
To add a regularized bias to the non- linear SVM version, a constant value K has been added to the kernel function.

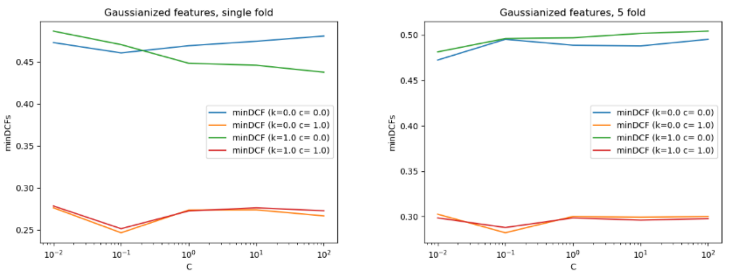
$$\hat{k}(x_1, x_2) = k(x_1, x_2) + K^2$$

In order to estimate the values of c and K that achieve the best results, different models have been trained by varying those parameters. In the following graphs are shown the minDCF obtained by the four best models found through this analysis, through the variation of the value of the hyper-parameter C.

Up: Raw features. Bottom: Gaussianized features. Left: single fold. Right: 5 fold.

Figure 8:





The best results are achieved with hyper-parameter K=0, c=1 and C=0.1.

Again, the K-fold results are consistent with the single fold results.

Table 6:

	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
Raw Features			
Quad SVM (K=0, c=1, C=0.1, π_T = 0.5)	**0.273**	0.798	0.691
Quad SVM (K=0, c=1, C=0.1, π_T = 0.1)	0.311	**0.760**	0.866
Quad SVM (K=0, c=1, C=0.1, π_T = 0.9)	0.325	0.959	**0.663**
Quad SVM (K=0, c=1, C=0.1)	0.277	0.774	0.723
Gaussianized features			
Quad SVM (K=0, c=1, C=0.1, π_T = 0.5)	**0.282**	0.730	0.627
Quad SVM (K=0, c=1, C=0.1, π_T = 0.1)	0.322	0.706	0.942
Quad SVM (K=0, c=1, C=0.1, π_T = 0.9)	0.308	0.853	0.613
Quad SVM (K=0, c=1, C=0.1)	0.296	0.674	0.718

These results confirm that quadratic surfaces are better at discriminating the classes of our dataset.

The best results are achieved with the balanced version of the model. Gaussianizing the features does not help to improve the performances.

Now the focus is on the RBF kernel.

2.4.3 SVM with RBF kernel

The Gaussian Radial Basis Function in another kernel function that can be used for non- linear SVM models.

The kernel function is

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

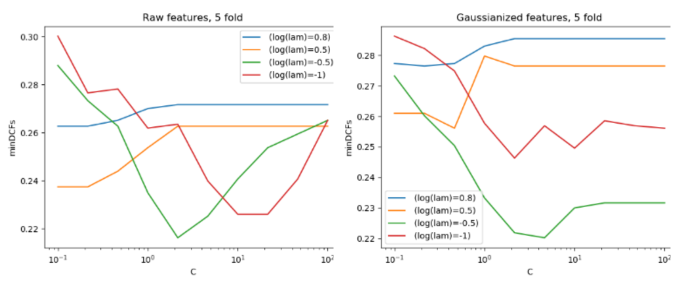
Where γ is the width of the kernel: small γ correspond to a wide kernel, while large γ to a narrow kernel.

The estimation of the kernel width γ has been done through a grid search in order to optimize C and γ.

In the graphs below are plotted the minDCFs of models trained with different γ and different C.

Left: raw features. Right: Gaussianized features.

Figure 9:



The plot shows that both γ and C influence the results. To jointly optimize both hyperparameters, the values chosen are $\gamma = 10^{-0.5}$ and $C=10^{0.1}$.

In the table are shown the results achieved through the model trained with those hyper-parameters, with and without re-balancing of the classes costs.

Table 7:

	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
Raw Features			
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$, $\pi_T = 0.5$)	**0.229**	0.573	0.610
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$, ,$\pi_T = 0.1$)	0.298	0.594	0.888
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$, $\pi_T = 0.9$)	0.261	0.764	**0.583**
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$)	0.242	0.570	0.774
Gaussianized features			
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$, $\pi_T = 0.5$)	0.230	0.523	0.615
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$, ,$\pi_T = 0.1$)	0.303	0.575	0.819
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$, $\pi_T = 0.9$)	0.242	0.692	0.606
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$)	0.232	0.539	0.656

The RBF results significantly outperforms our previous models for both the target application and the unbalanced ones.

Class re-balancing helps to increase the performances.

Therefore, the best model found so far for the target application is the class-balanced RBF SVM trained with raw features.

Comparing the quadratic models in term of minDCF on the target application:

Table 8:

Comparison between the bet quadratic models ($\tilde{\pi} = 0.5$)	**minDCF**
MVG Full - Cov	0.304
Quad LR ($\lambda = 10^{-3}$)	0.274
Quad SVM (K=0, c=1, C=0.1, $\pi_T = 0.5$)	0.273
RBF SVM ($\gamma = 10^{-0.5}$, $C=10^{0.1}$, $\pi_T = 0.5$)	0.229

The RBF is clearly the one that best performs on the target application.

The last model considered is the GMM.

2.5 GAUSSIAN MIXTURE MODEL

The Gaussian mixture model is a probabilistic model which assumes that all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters.

It allows to approximate any sufficiently regular distribution.

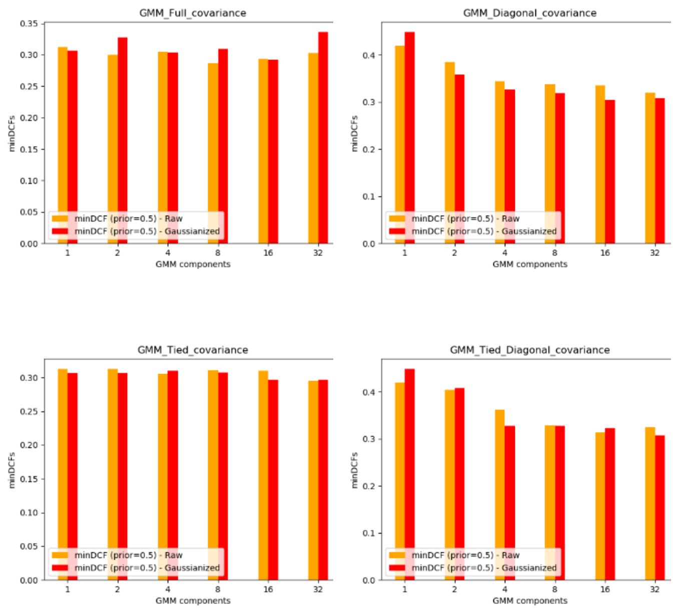
It can be used as a classification model by assuming that each class can be approximated by a GMM with a given number of sub-components.

Since this model benefits from having a large amount of training data, the approach used for the following analysis was the K-fold approach.

Moreover, different types of GMM have been tried (GMM with full covariances, GMM with diagonal covariances, GMM with tied covariances and GMM with tied diagonal covariances).

The graphs below show the performances in terms of minDCF obtained with raw features and Gaussianized features varying the number of components of each GMM.

Figure 10:



It can be seen that the model trained with Gaussianized features has overall similar performances to the one trained with raw features.

Also, the best results are achieved with GMM with full covariances and GMM with tied covariances.

In the table below are reported all the minDCF emerged from the analysis.

Table 9:

Number of components:	**2**	**4**	**8**	**8**	**16**	**32**
Raw Features						
Full Cov	0.312	0.300	0.305	**0.286**	0.292	0.302
Diag Full	0.420	0.384	0.344	0.337	0.335	0.319
Tied Full	0.312	0.312	0.305	0.310	0.309	**0.295**
Tied Diag	0.420	0.404	0.362	0.328	0.314	0.324
Gaussianized features						
Full Cov	0.306	0.327	0.303	0.309	0.292	0.336
Diag Full	0.448	0.358	0.327	0.318	0.305	0.309
Tied Full	0.306	0.306	0.309	0.307	0.296	**0.296**
Tied Diag	0.448	0.407	0.327	0.327	0.323	0.307

The results achieved for the one-component model are consistent with the ones of MVG models.

The best non-tied model is Full Covariance with 8 components trained with raw features. Instead, some degrees of overfitting can be seen in this model increasing more the number of components.

Furthermore, in this model the Gaussianization shows slightly worse performances. This can be explained since Gaussianization reduces the dynamic range of samples far from the data mean and thus reduces the separability of some clusters.

The best tied model is the Tied full model trained with raw features but also the one trained with Gaussianized features that has a similar performance.

From this first analysis of the studied classifiers, the two selected models are **SVM with RBF kernel (parameters $\gamma = 10^{-0.5}$, C=$10^{0.1}$, $\pi_T = 0.5$)** trained with **raw features** and the **SVM with quadratic kernel (parameters C=0.1, $\pi_T = 0.5$))** trained with **raw features**.

2.6 Analysis in terms of actual DCF

Up to now the analysis focused only on minimum DCF metrics.

Min DCF measures the cost that would be paid making optimal decisions for the validation set using the recognizer scores.

The actually paid cost, however, depends on the goodness of the decisions made using those scores that, in the binary case, corresponds to the goodness of the threshold used in practice to perform class assignment.

Therefore, now the analysis turns the attention to actual DCFs

If the scores are well calibrated, the optimal threshold that optimizes the Bayes risk is

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

The following table compares the values got as minDCF and actDCF for the three different applications using the two selected models.

Table 10:

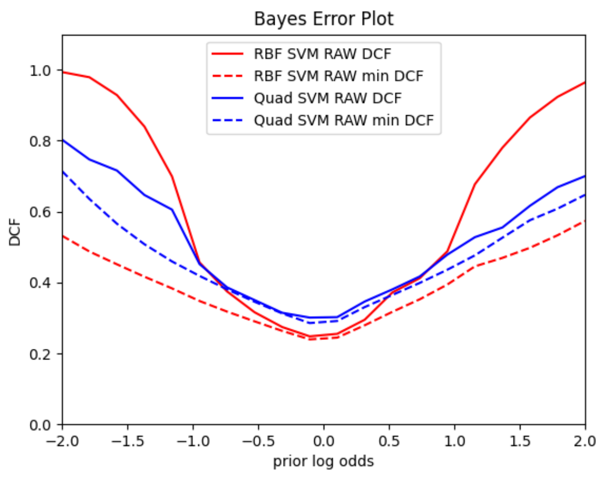
	$\tilde{\pi} = 0.5$		$\tilde{\pi} = 0.1$		$\tilde{\pi} = 0.9$	
	minDCF	**actDCF**	**minDCF**	**actDCF**	**minDCF**	**actDCF**
RBF SVM	**0.229**	**0.234**	0.573	0.993	0.610	0.982
Quad SVM	0.273	0.296	0.798	0.852	0.691	0.749

These results show that both models are quite calibrated for the target application of this report. In particular, with the RBF model the minDCF almost corresponds to the actual DCF, although both models lack a probabilistic interpretation.

On the other hand, for the unbalanced applications the scores are overall not well calibrated especially using the RBF classifier.

For a better visualization, these results have been plotted in a Bayes Error Plot, which shows the DCFs for different applications.

Figure 11:



It is clearly visible that the RBF outputs (quite-) calibrated scores only for the target application ($\tilde{\pi} = 0.5$), whereas changing prior log odds (and therefore the application) the curve of the minDCFs and the actualDCF depart from each other.

On the other hand, the quadratic SVM is well calibrated in a wider range of applications.

Therefore, for the target application for both classifiers, using the theoretical optimal threshold would give good results. Nonetheless, since this would not apply for the unbalanced applications, the solution followed was estimating a close-to-optimal threshold. The results are shown in the table below:

Table 11:

	minDCF	**actDCF**	**DCF with estimated t***
		$\tilde{\pi} = 0.5$	
RBF SVM	0.229	0.238	0.249
Quad SVM	0.289	0.315	0.314
		$\tilde{\pi} = 0.1$	
RBF SVM	0.576	**0.991**	**0.597**
Quad SVM	0.788	**0.859**	**0.898**
		$\tilde{\pi} = 0.9$	
RBF SVM	0.689	**0.988**	**0.732**
Quad SVM	0.697	**0.752**	**0.731**

This procedure looks quite accurate for the two unbalanced applications, since the results achieved with the estimated threshold are surely closer to the minDCF if compared to the DCF computed with the theoretical threshold.

Instead, the target application, being already quite calibrated, does not get a great improvement from this procedure.

Since this analysis focuses only on the target application, the chosen model is the **SVM with RBF kernel**, since it gives the best validation results and it also produces well calibrated scores.

3 EXPERIMENTAL RESULTS

The last step of this report assesses the quality of the chosen models on held-out data (the evaluation set).

Also, the performances are verified for unseen data.

For the experimental results the choice was to verify only the ones on the target application.

Again, the performances have been firstly evaluated by using the minimum DCF, then by using the actual DCF.

3.1 MVG Classifiers

Below are the Gaussian classifiers using either the single-fold protocol (i.e the model trained over 66% of the data) or the 5-fold cross validation protocol (i.e the final model that was re-trained using the whole dataset)

Table 12:

	66% Data	**100% data**
Raw Features – no PCA		
Full - Cov	**0.329**	**0.344**
Diag -Cov	0.379	0.473
Tied Full-Cov	**0.316**	**0.320**
Tied Diag- Cov	0.370	0.369
Gaussianized Features – no PCA		
Full - Cov	**0.325**	0.326
Diag -Cov	0.390	0.384
Tied Full-Cov	0.329	0.325
Tied Diag- Cov	0.396	0.387
Gaussianized Features – PCA (m=10)		
Full - Cov	0.334	0.330
Diag -Cov	0.332	0.331
Tied Full-Cov	0.323	0.326
Tied Diag- Cov	0.325	0.327
Gaussianized Features – PCA (m=9)		
Full - Cov	0.314	0.324
Diag -Cov	0.332	0.336
Tied Full-Cov	0.322	0.325
Tied Diag- Cov	0.325	0.324

The results are consistent with those achieved on validation set. Also in this case, the raw and Gaussianized features results are similar. The best model is confirmed to be the full-cov.

The results achieved with single fold and k-fold are very close and this means that for these models 66% of the data is enough to obtain good estimates of the model parameters.

PCA in most cases is not effective, but it is not detrimental for m=9.

Since the k-fold approach is more reliable and provides more robust results, it is therefore the one used for the models from now on.

3.2 LOGISTIC REGRESSION

3.2.1 Linear Logistic Regression

Below are shown the results achieved by considering the linear logistic regression models with estimated λ= 10^{-3}.

Table 13:

Raw Features	
LR (λ= 10^{-3} , $\pi_T = 0.5$)	**0.333**
LR (λ= 10^{-3} , $\pi_T = 0.1$)	**0.317**
LR (λ= 10^{-3} , $\pi_T = 0.9$)	0.350
LR (λ= 10^{-3})	0.340
Gaussianized features	
LR (λ= 10^{-3} , $\pi_T = 0.5$)	0.342
LR (λ= 10^{-3} , $\pi_T = 0.1$)	**0.310**
LR (λ= 10^{-3} , $\pi_T = 0.9$)	0.363
LR (λ= 10^{-3})	0.359

Again, the results are consistent with the expectations.

The best results are achieved with $\pi_T = 0.1$. However, the results with respect to the models using $\pi_T = 0.5$ and the unbalanced ones are not significant.

3.2.2 Quadratic Logistic Regression

The analysis is repeated with the quadratic logistic regression model.

Table 14:

Raw Features	
Quad LR (λ= 10^{-3} , $\pi_T = 0.5$)	0.253
Quad LR (λ= 10^{-3} , $\pi_T = 0.1$)	0.265
Quad LR (λ= 10^{-3} , $\pi_T = 0.9$)	0.258
Quad LR (λ= 10^{-3})	0.254
Gaussianized features	
Quad LR (λ= 10^{-3} , $\pi_T = 0.5$)	0.299
Quad LR (λ= 10^{-3} , $\pi_T = 0.1$)	0.293
Quad LR (λ= 10^{-3} , $\pi_T = 0.9$)	0.299
Quad LR (λ= 10^{-3})	0.288

As before, results on the evaluation set are consistent with those on the validation set.

The best results are now got with $\pi_T = 0.5$, but there is little difference.

3.3 SVM

3.3.1 Linear SVM

Below are reported also the results achieved with linear SVM:

Table 15:

Raw Features	
SVM (C=0.1 , $\pi_T = 0.5$)	0.334
SVM (C=0.1, $\pi_T = 0.1$)	0.643
SVM (C=0.1 , $\pi_T = 0.9$)	0.357
SVM (C=0.1)	0.312

The results are in line with the expectations. The choice of the hyperparameter has been effective also for the evaluation data.

3.3.2 SVM with quadratic kernel

Table 16:

Raw Features	
Quad SVM (K=0, c=1, C=0.1, $\pi_T = 0.5$)	**0.270**
Quad SVM (K=0, c=1, C=0.1,$\pi_T = 0.1$)	0.305
Quad SVM (K=0, c=1, C=0.1, $\pi_T = 0.9$)	0.283
Quad SVM (K=0, c=1, C=0.1)	0.274
Gaussianized features	
Quad SVM (K=0, c=1, C=0.1, $\pi_T = 0.5$)	0.307
Quad SVM (K=0, c=1, C=0.1,$\pi_T = 0.1$)	0.300
Quad SVM (K=0, c=1, C=0.1, $\pi_T = 0.9$)	0.298
Quad SVM (K=0, c=1, C=0.1)	0.283

Again, the results follow the expectations.

The best results are achieved with the balanced version of the model.

3.3.3 SVM with RBF kernel

The focus is now on the SVM model with RBF kernel.

Table 17:

Raw Features	
RBF SVM ($\gamma = 10^{-0.5}$, C=$10^{0.1}$, $\pi_T = 0.5$)	0.258
RBF SVM ($\gamma = 10^{-0.5}$, C=$10^{0.1}$, ,$\pi_T = 0.1$)	0.387
RBF SVM ($\gamma = 10^{-0.5}$, C=$10^{0.1}$, $\pi_T = 0.9$)	0.268
RBF SVM ($\gamma = 10^{-0.5}$, C=$10^{0.1}$)	0.298
Gaussianized features	
RBF SVM ($\gamma = 10^{-0.5}$, C=$10^{0.1}$, $\pi_T = 0.5$)	0.268
RBF SVM ($\gamma = 10^{-0.5}$, C=$10^{0.1}$, ,$\pi_T = 0.1$)	0.350
RBF SVM ($\gamma = 10^{-0.5}$, C=$10^{0.1}$, $\pi_T = 0.9$)	0.262
RBF SVM ($\gamma = 10^{-0.5}$, C=$10^{0.1}$)	0.290

The validation results are again consistent with the evaluation results and the SVM model with RBF kernel continues being the best performing model.

3.4 GAUSSIAN MIXTURE MODELS

Eventually, the results of the GMM are analyzed and reported below:

Figure 12:

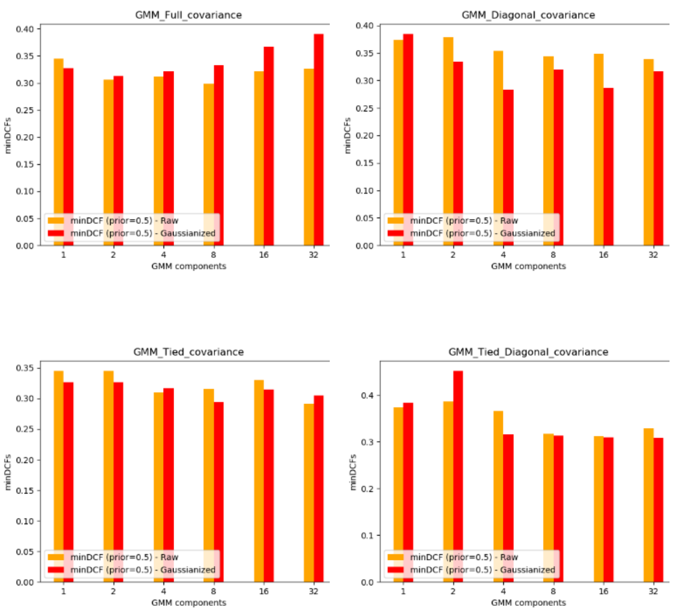


Table 18:

Number of components:	**2**	**4**	**8**	**8**	**16**	**32**
			Raw Features			
Full Cov	0.344	0.305	0.311	**0.298**	0.321	0.326
Diag Full	0.373	0.378	0.354	0.344	0.348	0.339
Tied Full	0.344	0.344	0.309	0.315	0.330	**0.291**
Tied Diag	0.373	0.386	0.366	0.317	0.312	0.328
			Gaussianized features			
Full Cov	0.326	0.313	0.321	0.332	0.367	0.390
Diag Full	0.384	0.334	0.282	0.320	0.286	0.316
Tied Full	0.326	0.326	0.316	0.294	0.314	**0.305**
Tied Diag	0.384	0.451	0.315	0.313	0.309	0.307

3.5 Analysis in terms of actual DCF

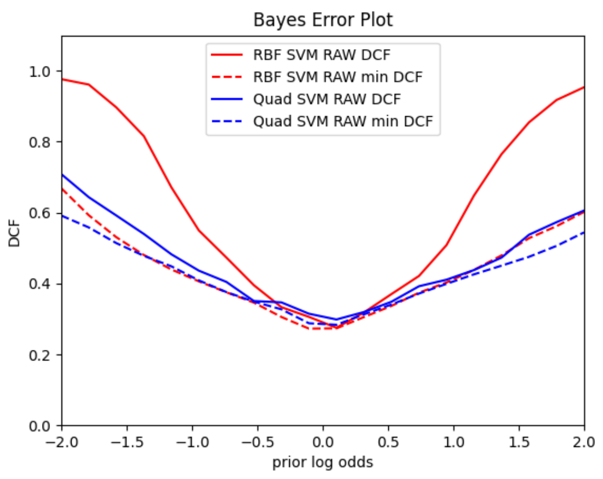
To assess the quality of the decisions that can be made using the recogniser scores, the actual DCF has been computed and compared to the minimum DCF that measures the potential capabilities of the system to produce good decisions.

Table 19:

	minDCF	**actDCF**
RBF SVM	0.258	0.286
Quad SVM	0.270	0.300

Below is also shown the Error Bayes Plot:

Figure 13:



Both the table and the bayes error plot show that for the target application the minimum DCF and the actual DCF are very close.

Score calibration does not require re-estimation of a different threshold for the target application, as the scores are already quite calibrated.

4 CONCLUSION

The wine quality dataset can be classified effectively using as model the SVM with RBF kernel that achieves a DCF cost of ≈0.3,

Furthermore, the Quadratic Logistic Regression and the SVM with quadratic kernel have good performance on this dataset.

However, the models are relatively ineffective for applications with unbalanced costs and prior proportions.

Overall, the similarity between validation and evaluation results suggests that the evaluation population is sufficiently similarto the training population.

The choices made on the training /validation sets proved effective also for the evaluation data.