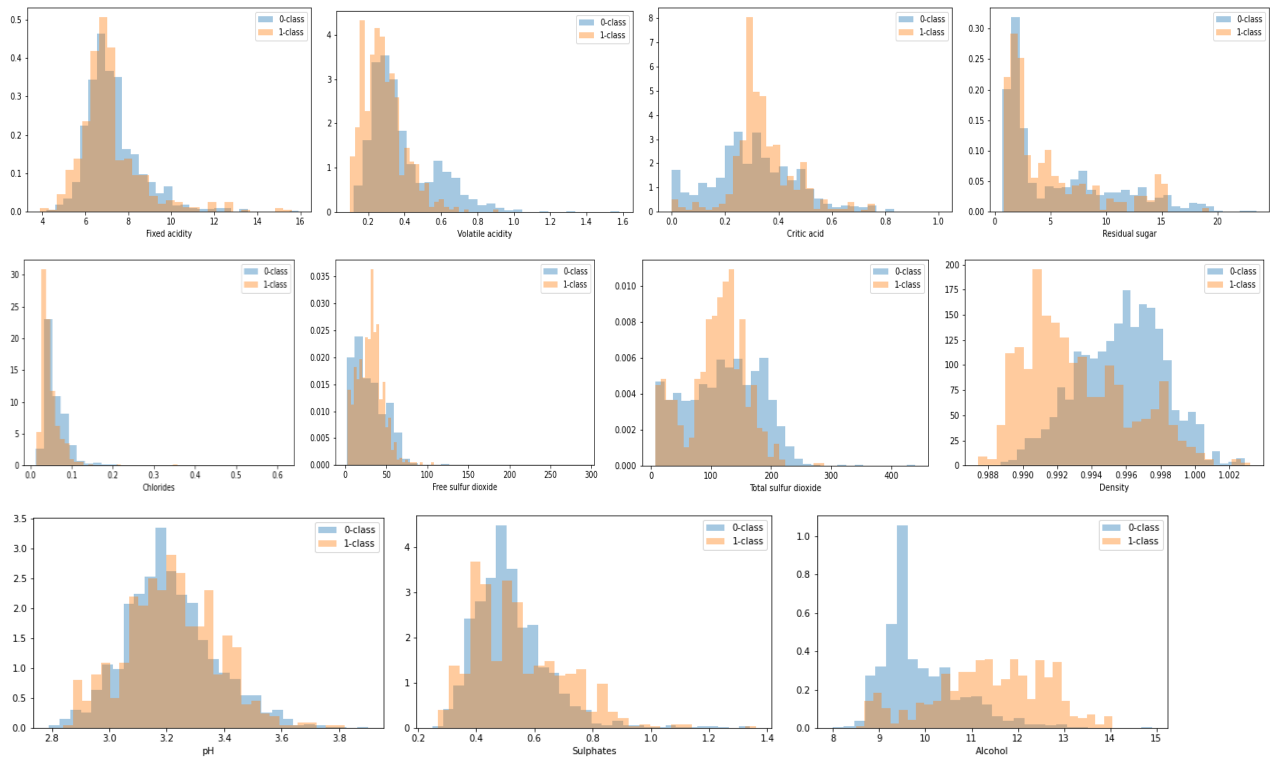
1. Introduction

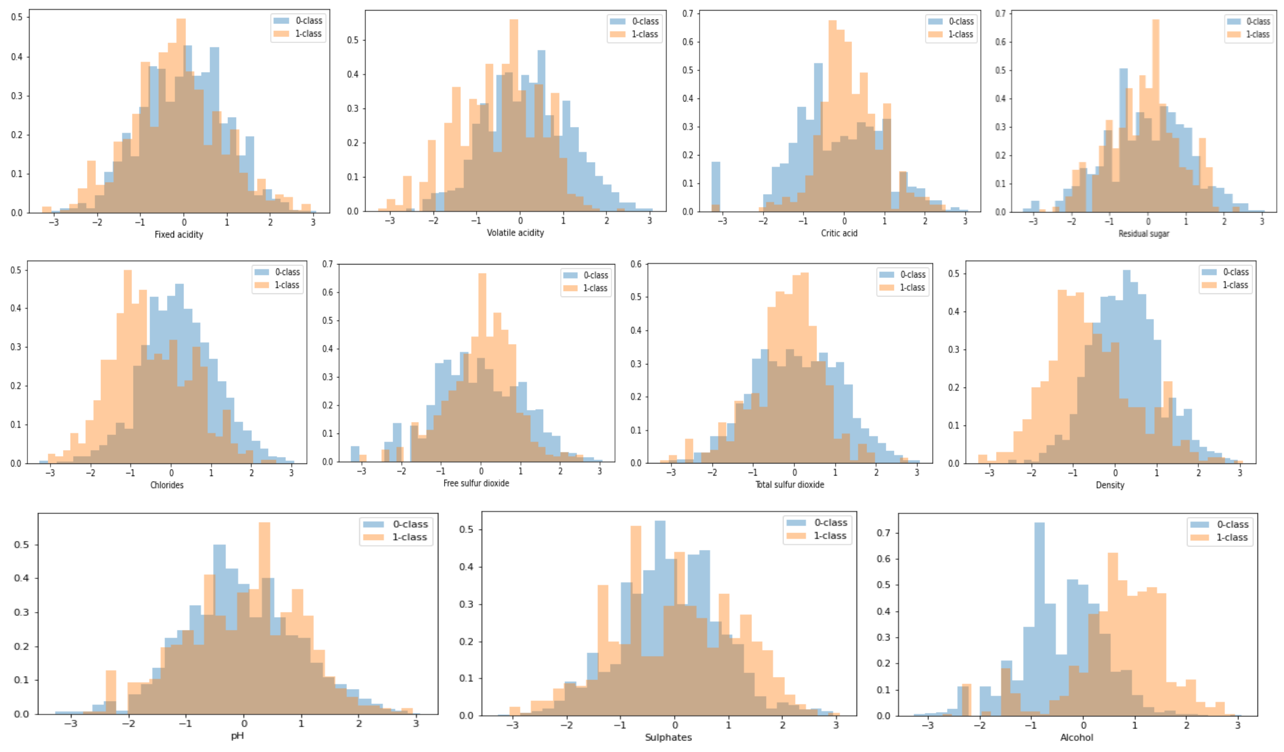
The ***Wine Quality*** dataset is a data set which describes sample related to red and white variants of the Portuguese *"Vinho Verde"* wine. The inputs include objective tests (e.g. *PH values*) and the output is based on sensory data (median of at least 3 evaluations made by wine experts). Each expert graded the wine quality between ***0 (very bad)*** and ***10 (very excellent)***.
These datasets can be viewed as classification or regression tasks. For the project, the classes have been binarized - Feature *12 is simply 0 (low quality, <= 5)* or *1 (high quality, >= 7).*
To simplify the problem, wines with quality 6 have been removed. Red and white wines have been merged into a single dataset.
The input variables are: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol
The training set contains ***1226*** low quality variants (identified by class ***0***) and ***613*** high quality variants (identified by class ***1***).
The evaluation set contains ***1158*** low quality variants (identified by class ***0***) and ***664*** high quality variants (identified by class ***1***).

2. Wine Quality Features

Histograms of the ***Wine Quality*** dataset features (training set). Features are sorted by their order, from left to right, top to bottom. Orange histograms are referring to high quality variants, blue histograms to low quality variants.

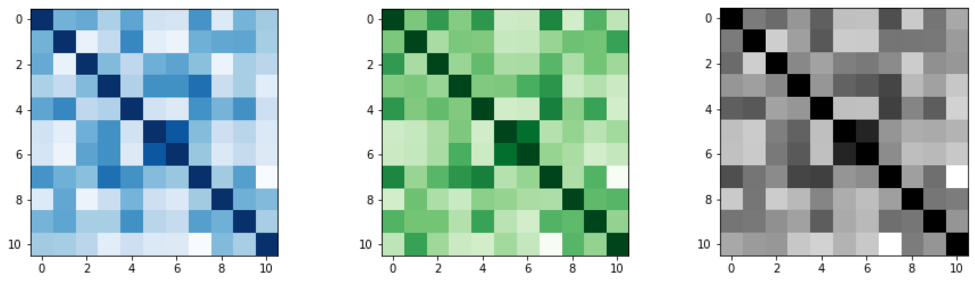


A first analysis of the training data shows that, in few cases, the raw features have strange distributions, characterized by the presence of outliers. Because of these outliers, we expect that classification approaches may produce not-optimal results for Gaussian-based methods for example, but since to this behaviour is detected only for few features, it could also be not if those features are not much influent on the classification. Indeed, we now plot data with a pre-processing approach which can help us to “*Gaussianize*” the dataset.
This method is based on the *rank* computation of the training set features and on the computation of the ***p.p.f function (percent point function)***. Obviously, the transformation is applied both to training and evaluation samples.



Histograms of the ***Wine Quality*** dataset features (training set) after Gaussianization. Features are sorted by their order, from left to right, top to bottom. Orange histograms are referring to high quality samples, blue histograms to low quality samples.

A correlation analysis throw heatmaps of Gaussianized features shows that feature ***5*** and ***6***, are strongly correlated. The heatmaps show us the *Pearson* correlation coefficient of the various features in the dataset.
In these graphs we have ***grey – whole dataset***, ***blue – low quality samples***, ***green – high quality samples.***
This could mean that using ***PCA*** to map data to ***10*** (or 9) we may benefit in terms of classification approaches.



3. Classifying Wine Quality features

We start considering simple Gaussian classifiers. Though we employed Gaussianization over the whole dataset, the histograms show for each class a consistent deviation from the Gaussian assumptions. We consider gaussian models with covariance matrixes both diagonal and not. We adopt the ***single split*** methodology and the ***5-fold*** methodology to evaluate the effects of using *PCA*. We consider a single fold consists of ***80%*** of the training data for the parameter's estimation and ***20%*** for validation data, randomly sampled. Our main application will be uniform prior one:

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

But we will also consider unbalanced applications:

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

We measure for the moment performance using the ***normalized minimum detection costs*** that represent the best cost that we can obtain with the scores generated by our classifier.

MVG classifiers – min DCF on the validation set

Gaussianized features – no PCA						
Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
	Single fold			**5-fold**		
Full-Cov	**0.256**	0.769	0.689	**0.304**	0.788	0.790
Diag-Cov	0.456	0.831	0.861	0.449	0.833	0.914
Tied-Full-Cov	0.349	**0.743**	0.808	0.356	0.801	0.885
Gaussianized features — PCA (m = 10)						
Full-Cov	0.278	**0.755**	0.673	0.329	**0.770**	**0.754**
Diag-Cov	0.346	0.860	0.683	0.385	0.856	0.779
Tied-Full-Cov	0.343	0.795	**0.647**	0.339	0.833	0.767
Gaussianized features — PCA (m = 9)						
Full-Cov	**0.273**	0.779	**0.665**	**0.313**	**0.787**	**0.760**
Diag-Cov	0.325	0.853	0.714	0.374	0.859	0.801
Tied-Full-Cov	0.343	0.795	0.648	0.338	0.833	**0.760**
Raw features (no Gaussianization) — no PCA						
Full-Cov	0.264	0.702	0.802	0.312	0.780	0.843
Diag-Cov	0.409	0.822	0.941	0.419	0.847	0.922
Tied-Full-Cov	0.327	0.754	0.706	0.334	0.811	0.746

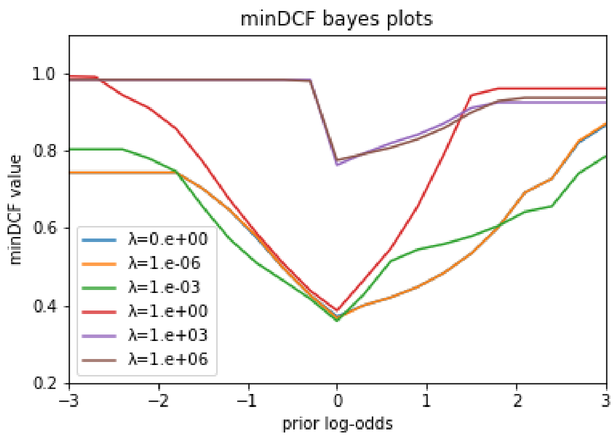
With ***Full-Covariance model*** for gaussianized features we can account for correlations, and it performs better than the other two methods both for *5-fold cross validation* and *single fold* method. With *PCA* we can reduce the number of features and we can simplify the estimation because *PCA* allows us to remove dimensions with small variance, but this in this way the method improve our estimation for both approach (*5-fold and single fold*) but only for $\tilde{\pi} = 0.5$ *and* 0.9, it means that our models based on the supposition that there are more low quality wines perform worse.

The ***Diagonal-covariance model*** for gaussianized features can simplify the estimation compared to the previous model, but due to the correlation for two features it is not the model that performs best for our classification. *PCA* diagonalizes the global covariance matrix, but this does not imply that the within-class covariance matrices of each class would be diagonal. This model, at the end, perform better if we apply *PCA* to the gaussianize dataset but its performance doesn't reach the performance that we can see if we apply a model with full covariance matrix.

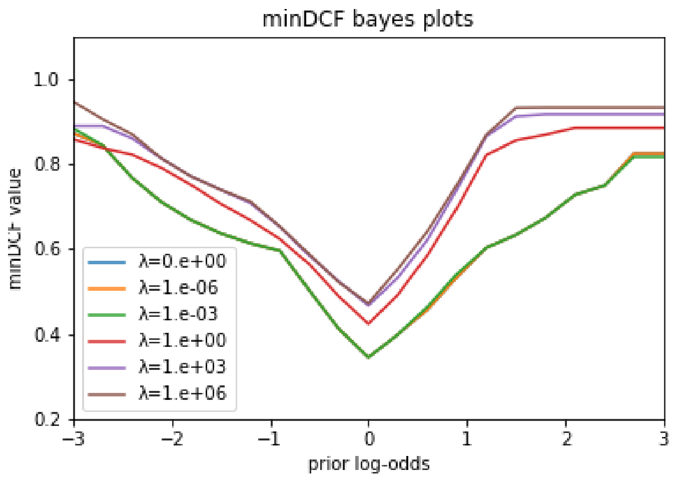
The ***Tied-Full-Covariance*** model can capture correlations because of the similarity of the covariance matrixes. But in this case the assumption does not improve the estimation because we have enough data for both classes, therefore estimating separately the matrixes provides a better model. Also, with this model *PCA* we can reduce the *minDCF* in relationship with a model without this but at the end this method does not allow us to perform better the estimation. Gaussianization significantly improves performance both for *single fold* approach and for the *5-fold* approach over raw features. Overall, the best candidate is currently the *MVG* model with *Full Covariance* matrixes, we choose the *5-fold* approach, because this approach can provide to us more reliability results, without *PCA.* At the end, we can also see that all the models are useless for imbalanced application.

We can now start considering regularized ***Logistic Regression*** and since our classes are almost imbalanced, so we can compare models with different values for λ also for re-balance costs for the classes and for no re-balance costs. The re-balancing of the costs for the classes can be done by minimizing the objective function with different values of $\boldsymbol{\pi_t}$. Below there are results with different values of $\boldsymbol{\pi_t}$ for the re-balancing.

***Linear Log-Reg*:** *minDCF* for $\lambda = \mathbf{0, 10^{-6},\ 10^{-3}, 1, 10^{3}, 10^{6}}$.

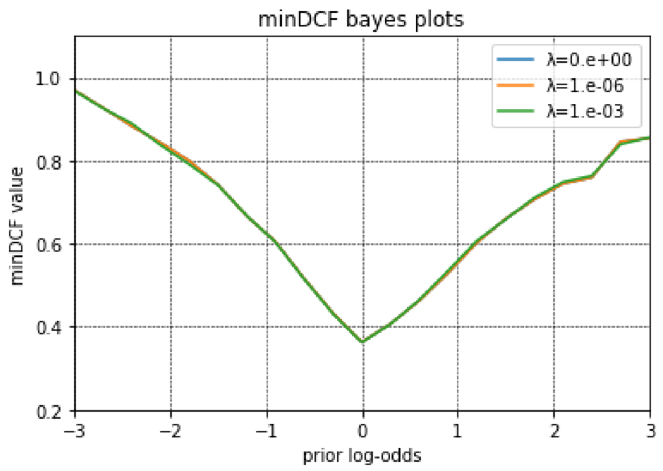


This graph represents ***minDCFs*** using the single fold approach correlated to ***raw features*** without *PCA*, ***which is the same dataset used for the MVG full-covariance model which, together with the analogous model for gaussianized features, gave us the best performance about the single fold approach***. We can see the trend of the *minDCF* in relation to *21* different values of $\tilde{\boldsymbol{\pi}}$ (from 0.1 to 0.9) on a logarithmic scale. We can see in this case that for raw features the better result is related to $\boldsymbol{\lambda} = \mathbf{10^{-3}}$ but the trend is that with decreasing lambda the result improves. At the end we can see that with large value for λ we obtain a solution that has small norm with large values for *minDCF* that cannot help us to well separate the classes.



This graph instead represents the ***minDCFs*** related to the ***Gaussianized*** dataset and calculated using also in this case the single fold approach without *PCA.* Gaussianization in this case allows us to obtain better results than the previous model based on raw features in particular for high lambda values. The trend of the curves is the same as also detected in the previous graph.

Finally, this graph is the graph associated with the representation of ***minDCFs*** using the ***5-fold*** approach on ***Gaussianized*** data (***which dataset is the one that previously gave us the best results for the MVG full-covariance model for the 5-fold approach***). In this case, in addition, we have rebalanced the cost of the individual classes by minimizing the weighted function objective values using different pit values. The results shown are relative to $\boldsymbol{\pi_t = 0.5}$ (we have also restricted our computation only for low values of lambda because higher values are useless for our computation due to the fact which we have high values for *minDCFs*).



We restrict the analysis to models with balanced costs and consider only the ***K-fold approach.***

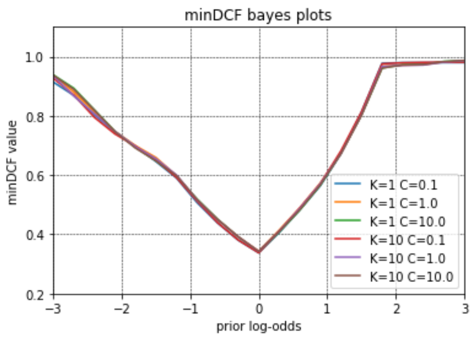
Gaussianized features — no PCA			
Type	$\boldsymbol{\tilde{\pi} = 0.5}$	$\boldsymbol{\tilde{\pi} = 0.1}$	$\boldsymbol{\tilde{\pi} = 0.9}$
MVG (Full-Cov)	**0.304**	**0.788**	**0.790**
MVG (Tied-Full-Cov)	0.356	0.801	0.885
Gaussianized features — no PCA			
	$\boldsymbol{\tilde{\pi} = 0.5}$	$\boldsymbol{\tilde{\pi} = 0.1}$	$\boldsymbol{\tilde{\pi} = 0.9}$
Log Reg ($\boldsymbol{\lambda = 10^{-6}, \pi_t = 0.5}$)	0.363	0.857	0.759
Log Reg ($\boldsymbol{\lambda = 10^{-6}, \pi_t = 0.1}$)	**0.340**	**0.781**	0.946
Log Reg ($\boldsymbol{\lambda = 10^{-6}, \pi_t = 0.9}$)	0.375	0.904	**0.717**
Log Reg ($\boldsymbol{\lambda = 10^{-6}}$, **unbalanced**)	0.357	0.820	0.845
Raw features — no PCA			
Log Reg ($\boldsymbol{\lambda = 10^{-6}, \pi_t = 0.5}$)	0.356	0.833	0.662
Log Reg ($\boldsymbol{\lambda = 10^{-6}, \pi_t = 0.1}$)	0.340	0.824	0.702
Log Reg ($\boldsymbol{\lambda = 10^{-6}, \pi_t = 0.9}$)	0.367	0.863	0.641
Log Reg ($\boldsymbol{\lambda = 10^{-6}}$, **unbalanced**)	0.346	0.828	0.658

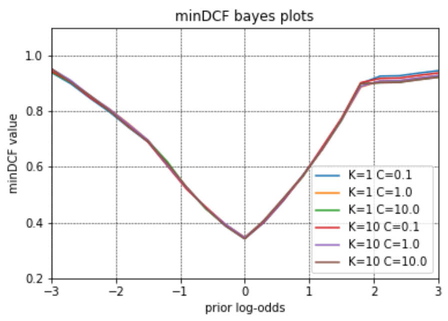
In conclusion, we can say that the *MVG* model with *full covariance* matrices performs better on our dataset considering the Gaussianized features. ***Linear logistic regression*** improves our performance only using a balanced costs for each class and only in the case where our application is not balanced (in particular when $\tilde{\pi} = 0.1$ and $\pi_t = 0.1$ or when $\tilde{\pi} = 0.9$ and $\pi_t = 0.9$). At the end, the best model remains the *MVG full covariance*, but therefore, we can think of using a quadratic logistic regression in relation to the separation rules of the multivariate model we have chosen. Instead, for the raw features we can see similar results but in this case for imbalanced applications with, $\pi_t = 0.9$ and $\tilde{\pi} = 0.9$ we have the best results observed since now.

Now, let's shift our attention to *SVMs* while obviously continuing to consider Gaussianized features without *PCA* and with a *5-fold approach*.

In our formulation of the *SVM* the regularization of the bias term, in general lead a sub-optimal decision in terms of separating margin. As ***K becomes larger***, the effects of regularizing ***b become weaker***. However, as ***K becomes larger***, the dual problem also becomes ***harder to solve*** (i.e., the algorithm may require many additional iterations). So, in this way higher values of ***K*** correspond to weaker regularization of the *SVM* bias term.

The choice of *C* instead does not look critical. We have selected different values only to compare different results.



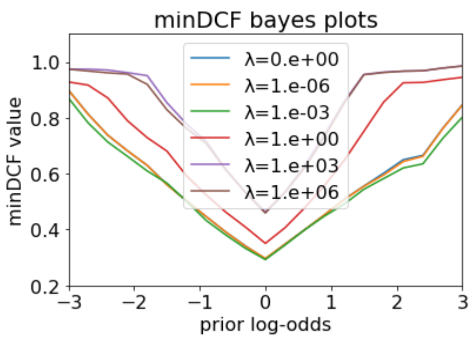


In the graphs above we can see on the formulations of the linear *SVM* for different parameters of *K* and *C*. Again, we resort to the *5-fold* validation approach for the dataset *Gaussianized* and without *PCA*. On the left we can see a model that balance the two classes in terms of value of *C*. We select $\boldsymbol{C_t} = \boldsymbol{C}\frac{\boldsymbol{\pi_t}}{\boldsymbol{\pi_t^{emp}}}$ and $\boldsymbol{C_f} = \boldsymbol{C}\frac{\boldsymbol{\pi_f}}{\boldsymbol{\pi_f^{emp}}}$ with $\pi_t = 0.5$ and with empirical prior that represent sample proportions. On the right the same model without balancing.
We can now compare in the table below our linear models in terms of *minDCF* for the *k-fold* approach without *PCA*.

Gaussianized features — no PCA			
Type	$\boldsymbol{\tilde{\pi} = 0.5}$	$\boldsymbol{\tilde{\pi} = 0.1}$	$\boldsymbol{\tilde{\pi} = 0.9}$
MVG (Full-Cov)	**0.304**	0.788	**0.790**
MVG (Tied-Full-Cov)	0.353	0.813	0.865
Log Reg ($\lambda = \boldsymbol{10^{-6}}$, **unbalanced**)	0.357	0.820	0.845
Log Reg ($\lambda = \boldsymbol{10^{-6}}$, $\boldsymbol{\pi_t = 0.5}$)	0.363	0.857	**0.759**
Linear SVM (K=1, C=0.1, unbalanced)	0.346	**0.787**	0.948
Linear SVM (K=1, C=0.1, $\boldsymbol{\pi_t = 0.5}$)	**0.338**	**0.763**	0.978

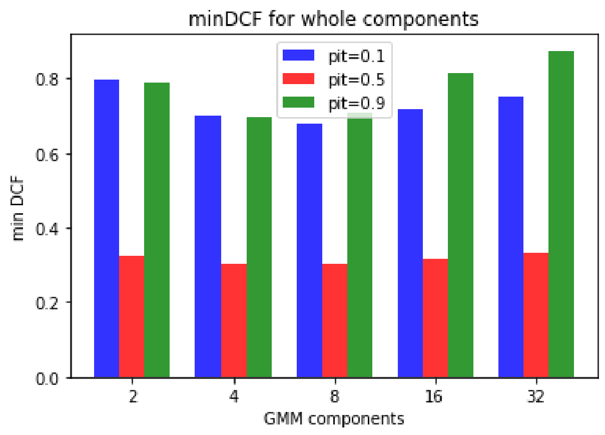
As we expected the *logistic regression* and the *linear SVM* have similar results for the computation of the *minDCF*. Another interesting thing is that with the balanced version of the logistic regression with $\pi_t = 0.5$, we do not have improvements in terms of result respect to the unbalanced version, contrary to what can be seen for *balanced linear SVM*. In any case, even for the linear formulation of the *SVM* there are no improvements compared to the formulation of a model *based on full-covariance MVG*. This thing may tell us that our data has an almost good gaussian distribution which is better than how much we can separate data linearly. In the next stage we will see how to perform the logistic regression with a quadratic separation rule to verify if this approach can help us or not.

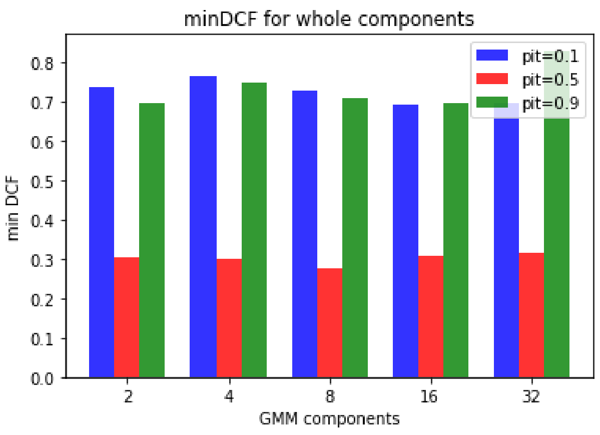
As we mentioned before we can consider training a model based on *quadratic logistic regression* to demonstrate the effectiveness of its in correlation to the same separation rule that we have for a *MVG full covariance model*. Therefore, we have applied the *quadratic logistic regression model* both with unbalanced costs for each class and both with balanced suppositions, in the table below you can see the results for the *k-fold approach* on the *Gaussianized* dataset with $\pi_t = 0.5$ (which gave us the bests results for this model).



Gaussianized features — no PCA			
Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
MVG (Full-Cov)	**0.304**	0.788	**0.790**
MVG (Tied-Full-Cov)	0.353	0.813	0.865
LR ($\lambda = 10^{-6}$, $\pi_t = 0.5$)	0.340	0.781	0.946
LSVM (K=1, C=0.1, $\pi_t = 0.5$)	0.338	**0.763**	0.978
Quadratic LR ($\lambda = 10^{-6}$, unbalanced)	**0.294**	**0.693**	**0.661**
Quadratic LR ($\lambda = 10^{-6}$, $\pi_t = 0.5$)	0.296	0.698	0.663

As expected, a model of this type allowed us to have a better estimate regarding the calculation of the *minDCF* than all the other models described above. In this case it performs at its best even with respect to unbalanced applications with different $\tilde{\pi}$ compared to the standard one.
Now we turn our attention on *GMM models*. These types of models can approximate any distributions, so even if our data is enough gaussian distributed it can behave even better. In the graph below we can see the behaviour of *minDCF* for different applications both for gaussianized features on the left and for raw features in the right. In this case the best results that we have are for *raw features* ant this means that *GMM* model can approximate better all the features that we have seen before with the histograms of raw features that aren't gaussian distributed. For this method we can see that for the main applications ($\pi_t = 0.5$) the number of components that perform better are 8 because of the overfitting that we have detected with a much higher number of components due to the *minDCF* do not show us consistent results.

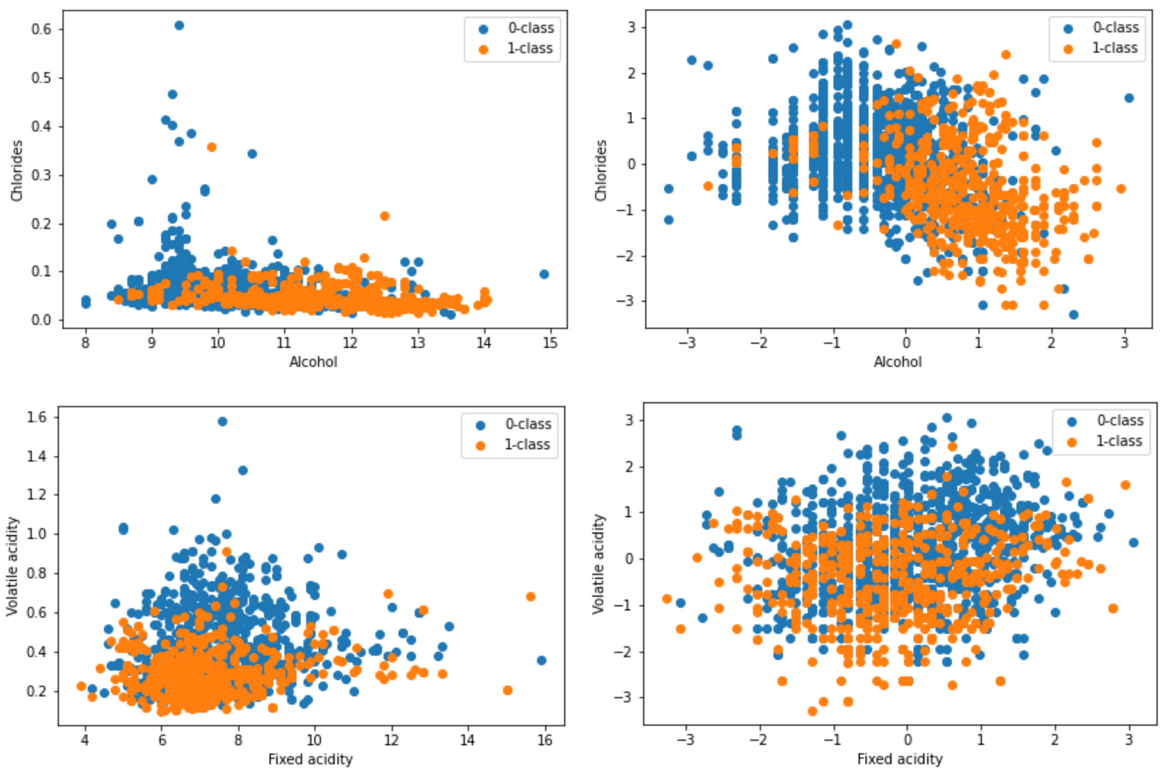




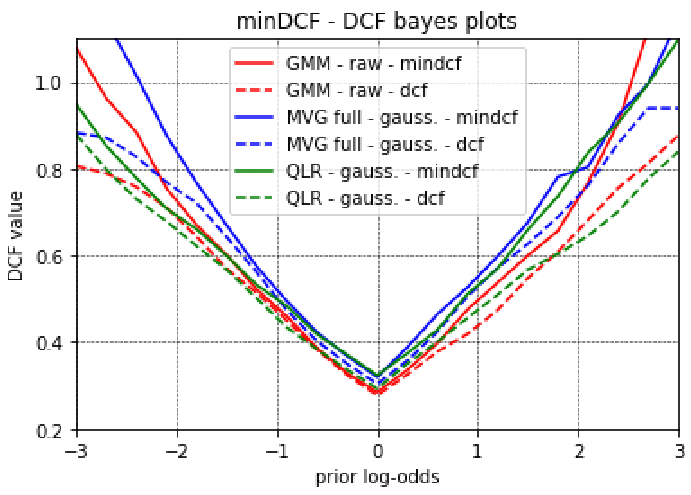
The table below show the results compared with the other results that we have collected previously.

Gaussianized features — no PCA			
Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
MVG (Full-Cov)	**0.304**	0.788	0.790
Quadratic LR ($\lambda = 10^{-6}$, unbalanced)	**0.294**	**0.693**	**0.661**
Quadratic LR ($\lambda = 10^{-6}$, $\pi_t = 0.5$)	0.296	0.698	**0.663**
Full-Cov, 8 Gauss	0.305	**0.678**	0.709
RAW features – no PCA			
Full-Cov, 8 Gauss	**0.278**	0.730	0.710

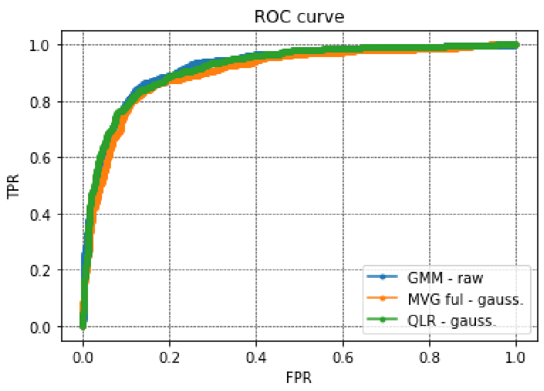
As we said before the GMM is more helpful without the supposition of the gaussianization in our case because we can see how even if our data has strange distribution (like the histograms show) with a pre-processing approach we reduce the separability of some clusters. Since each component has its own Gaussian, the model ends up finding clusters that do not correspond to the natural clusters of the dataset. The risk is that with the gaussianization the components may include data that should not be included. In the graphs below we can see the scatter associated to *Gaussianize features* (on the right) versus *Raw features* (on the left).

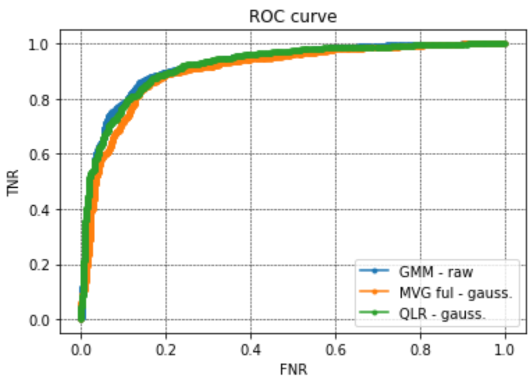


As we mentioned before the final best model for our data can be considered the ***GMM model with 8 components***. In the graph on the right we can see the behaviour of *minDCF* also for the other two model that we have seen that perform better (even with the *Gaussianization assumption*).



As we can observe with the ROC curves for the graph that show us the correlation between *FPR* and *TPR* the models that reach the highest *TPR* with the lowest *FPR* are preferable because it means that there are less errors on deciding that a class is *1*. Our graph reaches high *TPR* values with low *FPR* values, so our models are similarly good. Apparently the best one is *GMM*, but the difference is minimal. The same considerations are true for the *negative rates*.





MinDCF for the evaluation dataset

Gaussianized features — no PCA			
Type	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
MVG (Full-Cov)	**0.368**	**0.702**	1
Quadratic LR ($\lambda = 10^{-6}$, unbalanced)	**0.311**	**0.658**	**0.879**
RAW features – no PCA			
Full-Cov, 8 Gauss	**0.289**	0.744	**0.694**

4. Evaluation

After finding the best candidate models to evaluate, we can apply them to the evaluation data and obtain their minDCF.
This shows that our best decisions overall is the GMM model with 8 components, for both our main application and also the $\tilde{\pi} = 0.9$ one. The obtained minDCFs are compliant with our validation, so we can say that our model is good enough.

5. Conclusions

Concluding, our approach which is based on three different models on two different datasets (pre-processing or not) shows us that there are a lot of similarity between validation and evaluation set. This suggests to us that the datasets are similar and that our decisions are also true for both. We could improve our results using an approach based on kernel SVM that probably gave us similar results with quadratic logistic regression, but the models that we have obtained are sufficiently good. In any case the results obtained are not consistent for unbalanced applications.

Alberto Castrignanò s281689
Stefano Rainò s282435