Gender Detection Analysis

Alessandro Mandrile

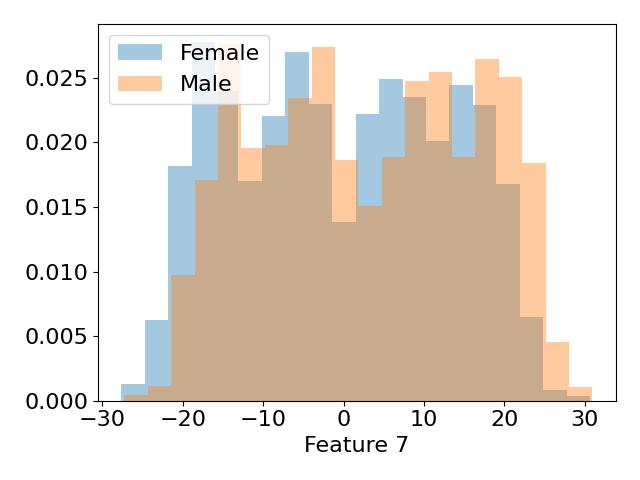
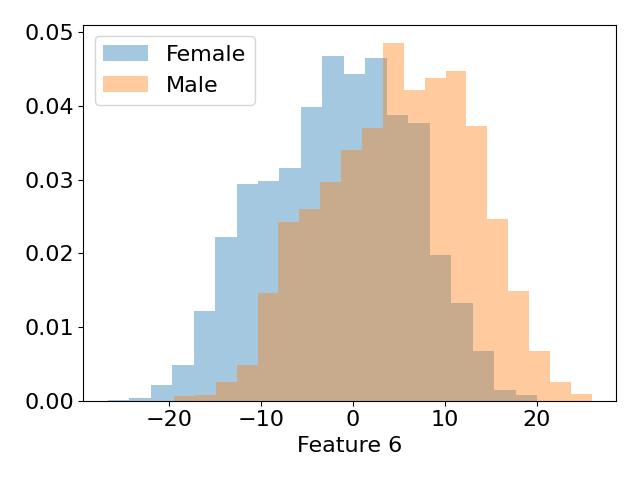
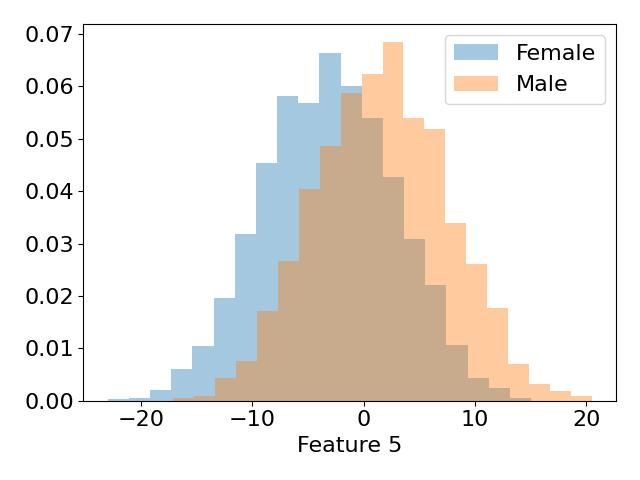
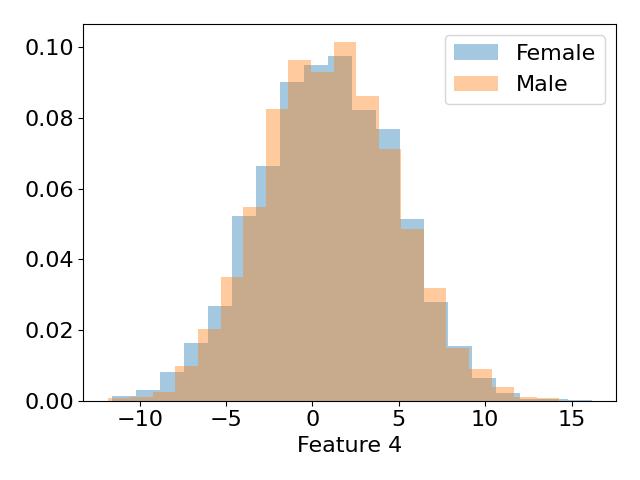
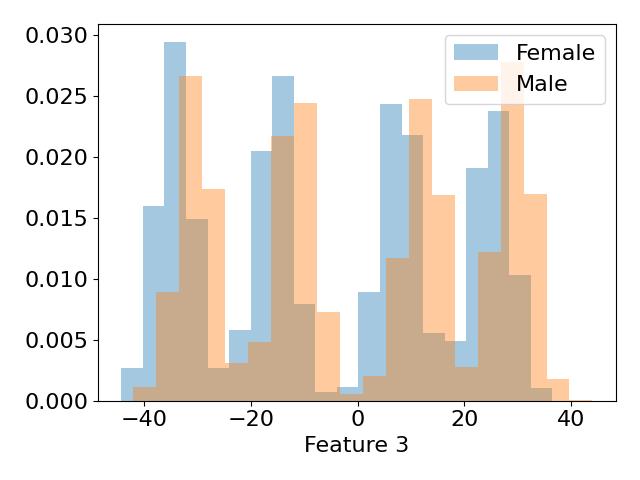
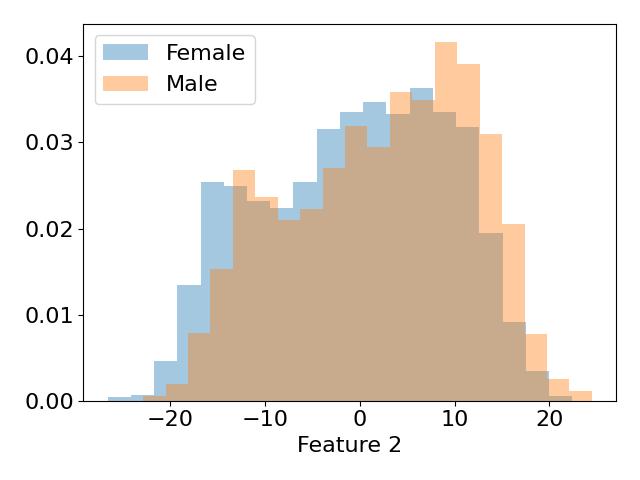
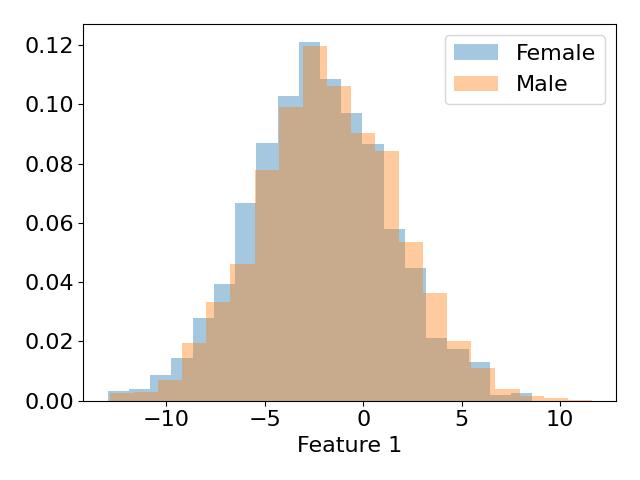
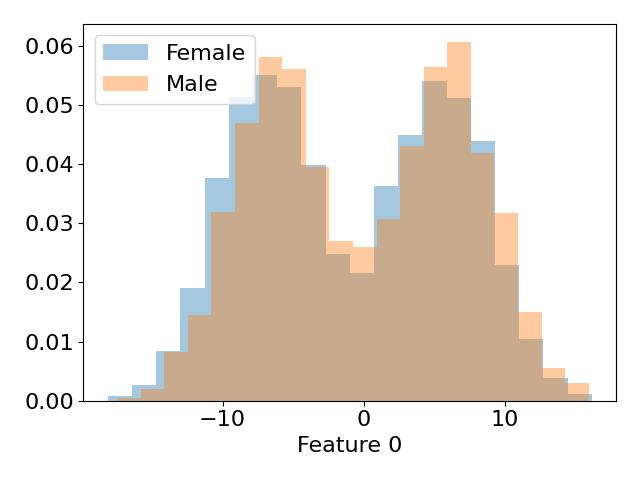
* Introduction

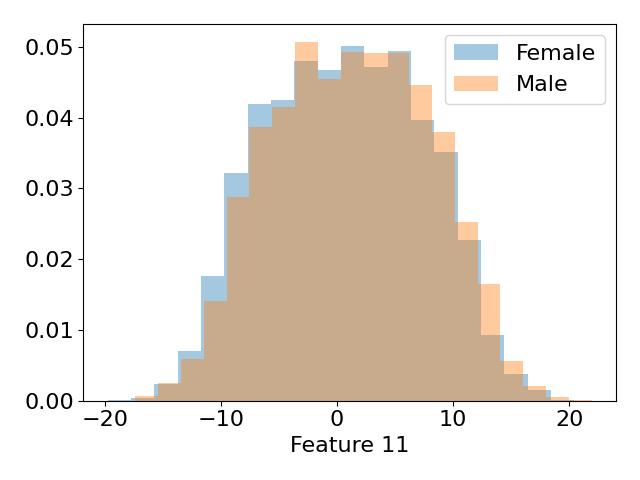
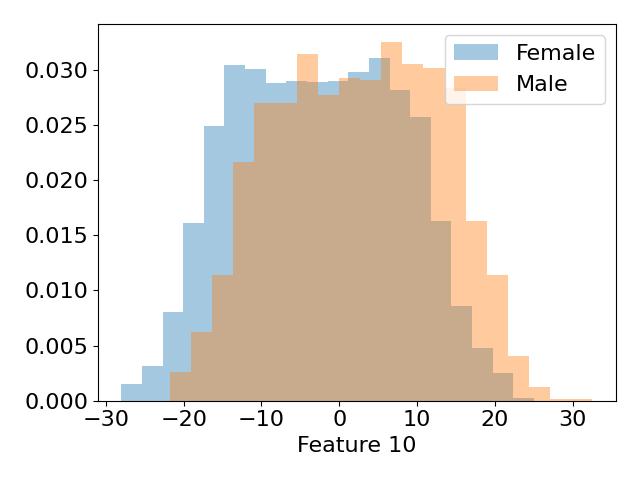
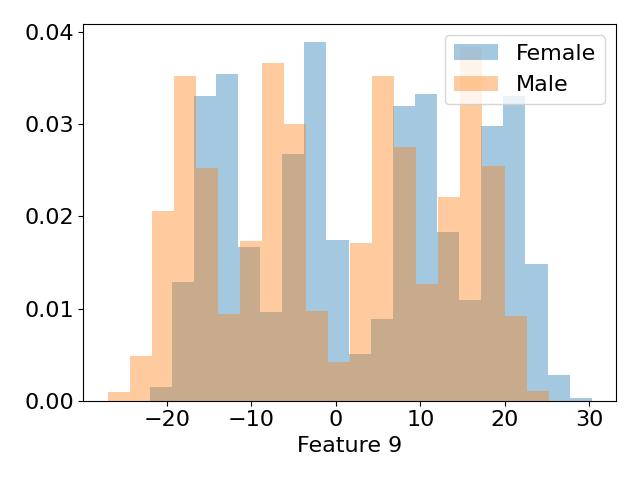
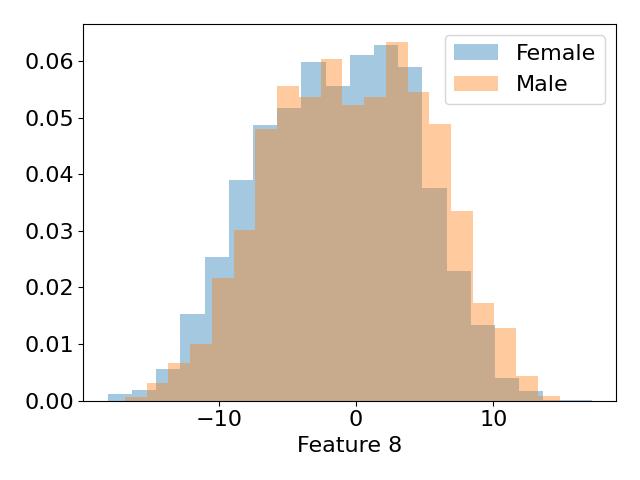
The dataset consists of synthetic speaker embeddings that represent the acoustic characteristics of a spoken utterance. A speaker embedding is a small-dimensional, fixed sized representation of an utterance. Features are continuous values that represent a point in the m-dimensional embedding space. The embeddings have already been computed and reduced to 12-dimensional feature space.

Each row corresponds to a different speaker and contains 12 features followed by the gender label (1 for female, 0 for male). The features do not have any meaning. Speakers belong to four different age groups. The age information, however, is not available.

* Features

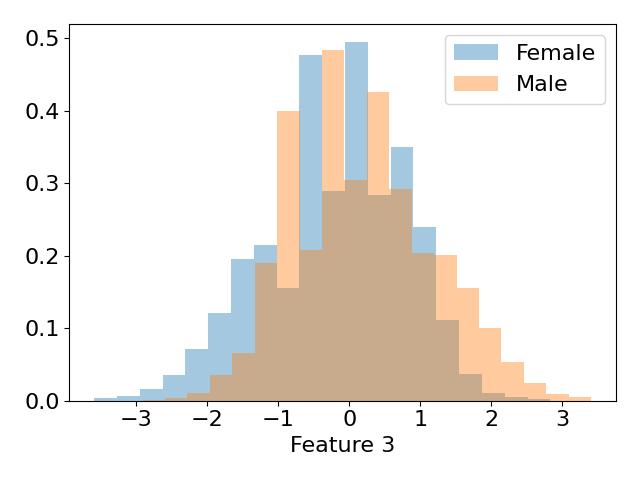
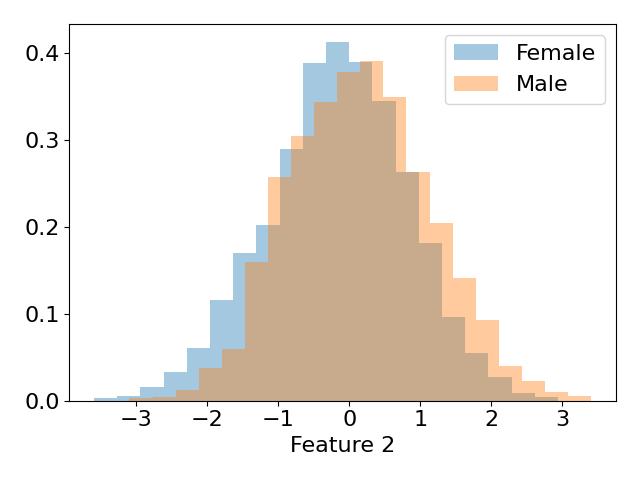
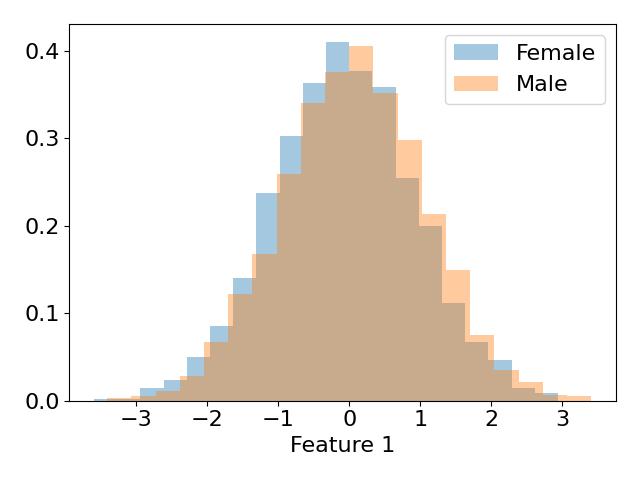
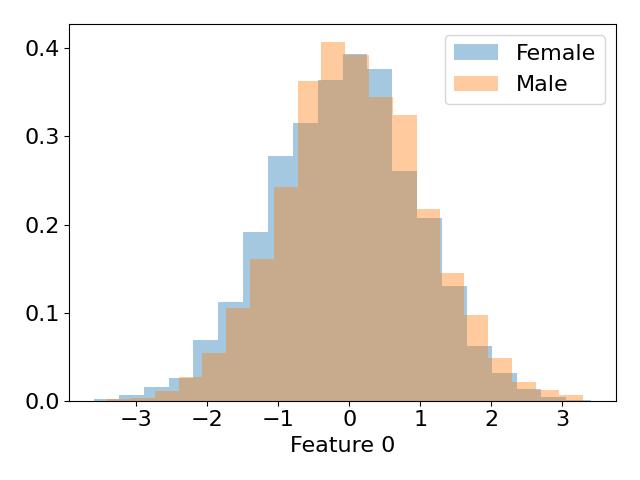
Histogram of the dataset features (training set). Features are sorted by their order, from left to right, top to bottom.

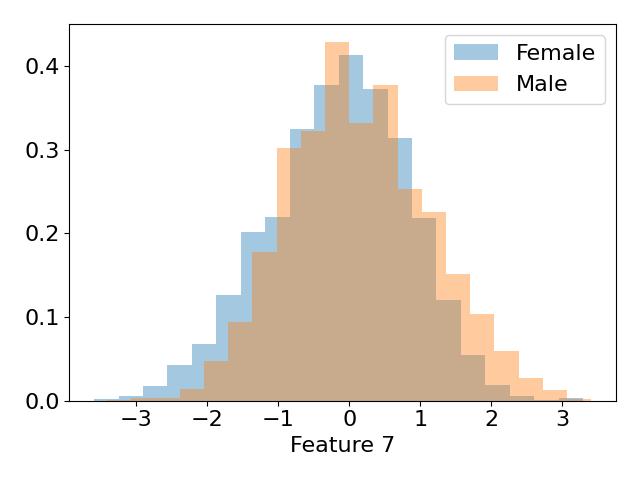
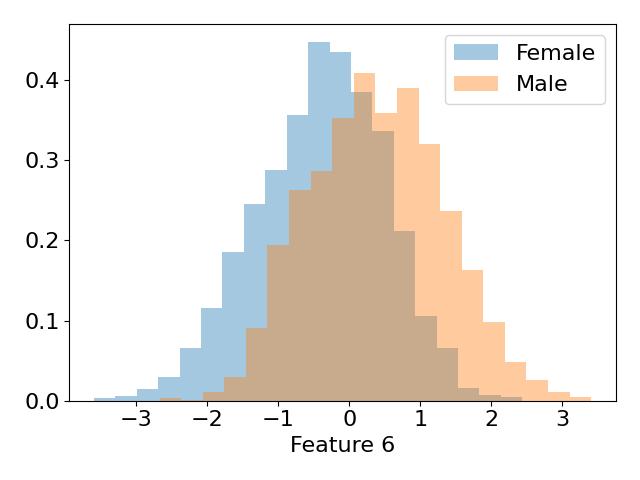
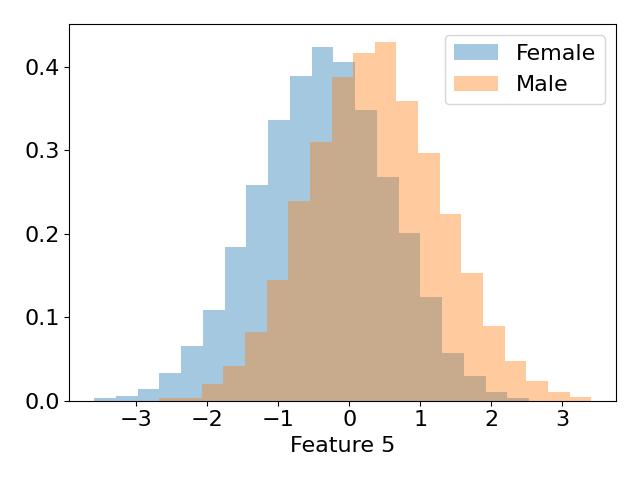
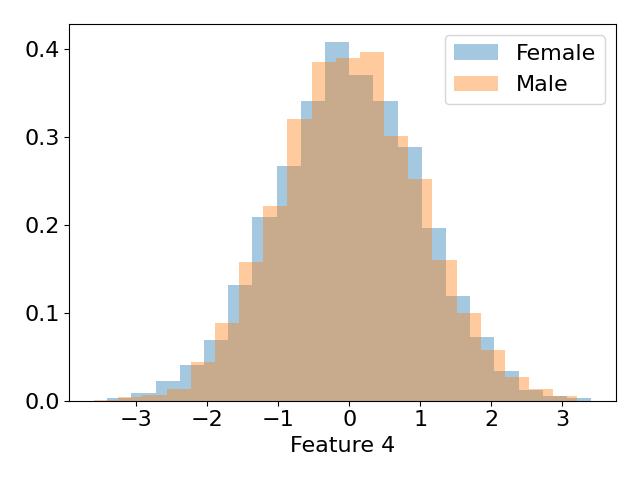


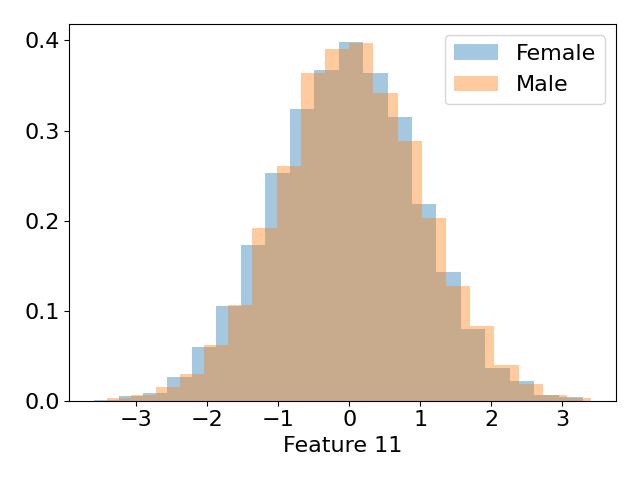
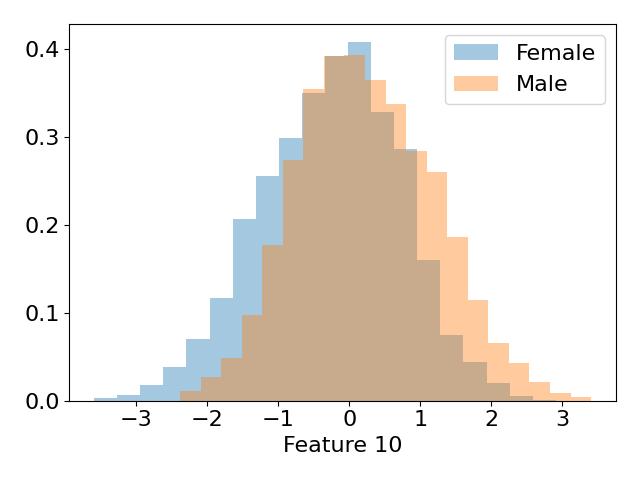
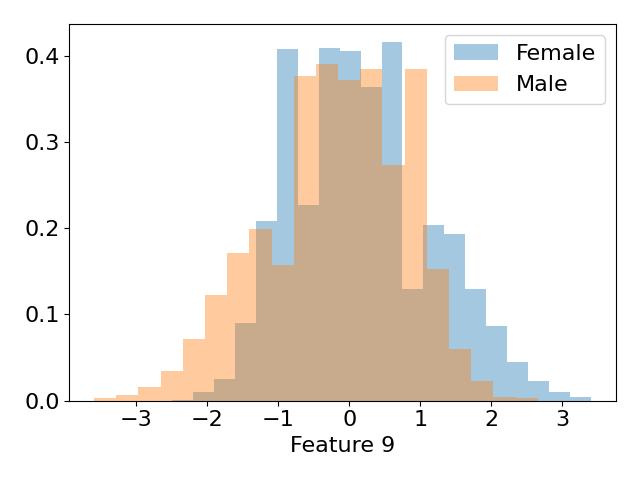
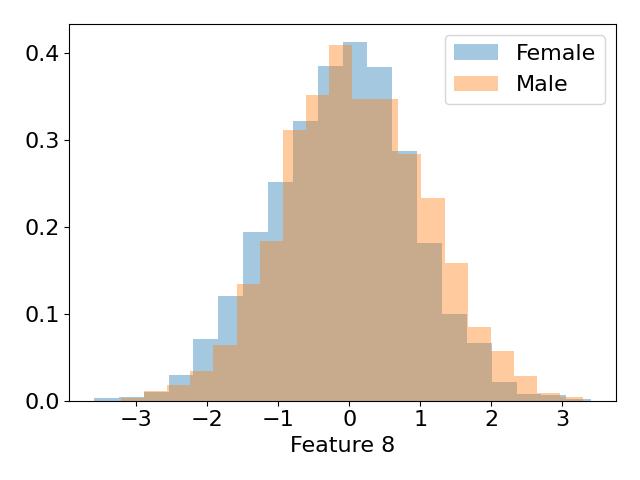


At first glance it’s easy to recognise the four different gaussians on features 3, 7, 9, but also on features 0 we can clearly recognise 2 spikes. Although on features 2,8,10 there are more peaks, they could be caused by imprecision in bins separation. Concerning features ranges we are lucky, as they are similar (10-30), but we will eventually apply a Z-Normalisation to them. We will see that while is never detrimental, in certain it increases a lot the performance of the classifier.

We will also apply Gaussianification to help models needing a gaussian features, but will see that it even lowers classifier performance, plus some classifiers, like GMM and RBF kernel Support vector machines, can exploit better those well separated features. Here the features of train set after applying guassianify:





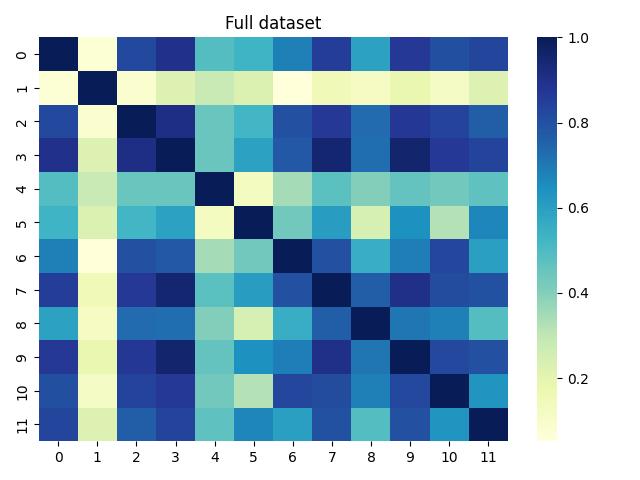
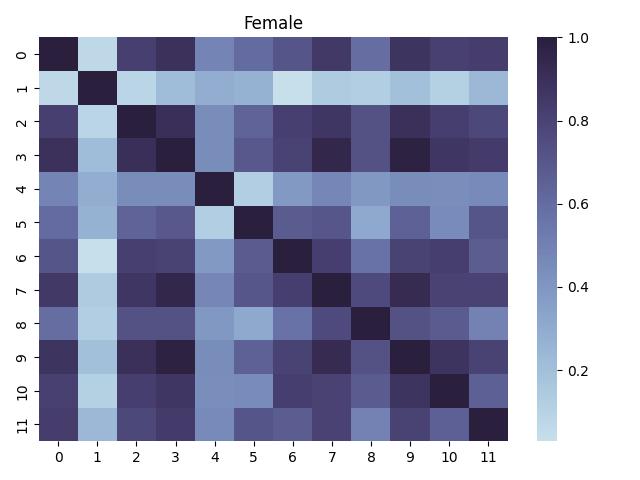
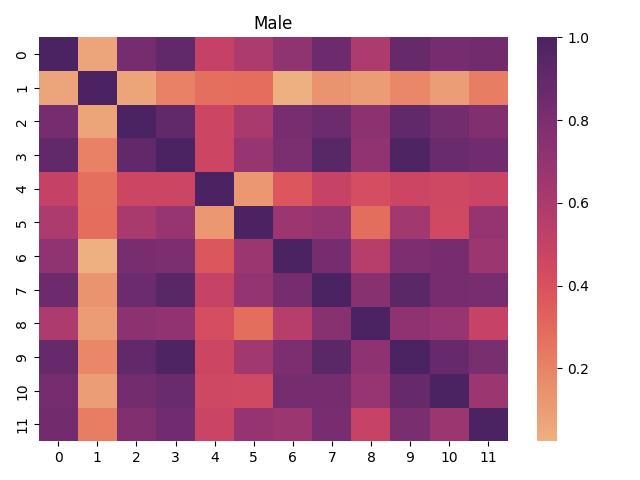


Note that, as expected, we lost the identity of multiple in-class clusters, but now features are more close to guassians.

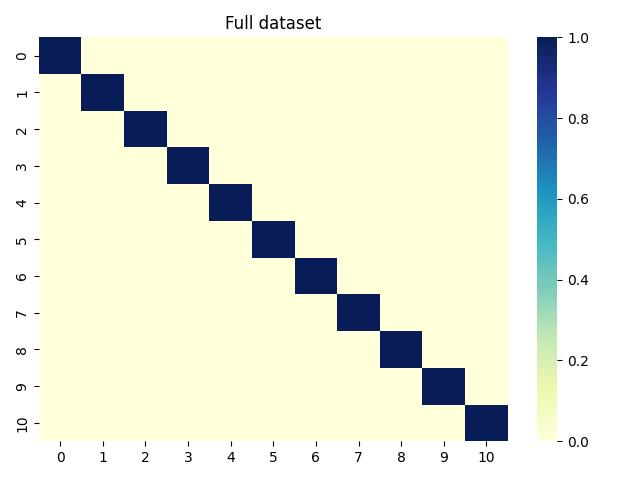
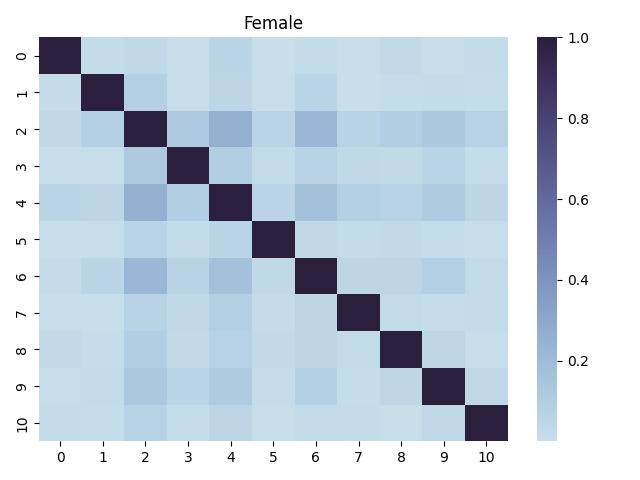
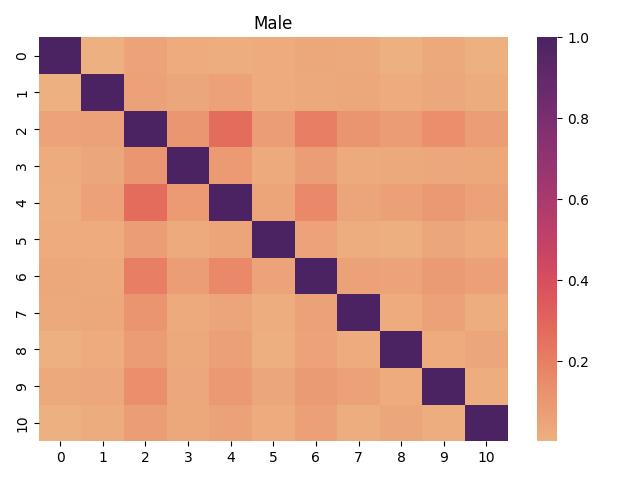
Final notes: Z-Normalisation and Gaussianify assumes that Train data and Test data will have similar distributions, since the transformations are computed on train data and applied on test.

* Multivariate Gaussian classifier

Now we will consider Guassian classifiers, but first it is better to check the absolute value of the Pearson correlation coefficient . Darker color implies larger value



Heat maps don’t depict a good picture. The number of correlated features is high, but after just a 11-PCA the scenario we have is this:



Although feature correlation has well decreased, results will show that PCA fails in dividing classes, causing also those classifier that benefit from it not to reach their competitor performance

In the process of choosing the best model and for optimizing hyperparameter, K-fold cross-validation has been used. Since the number of train samples is high, K=5 is chosen.

The main application will be a uniform prior one: (, Cfp, Cfn) = (0.5, 1, 1), but we will also consider unbalanced applications(, Cfp, Cfn) = (0.1, 1, 1) and (, Cfp, Cfn) = (0.9, 1, 1), even if both Train and Test dataset are perfectly balanced.

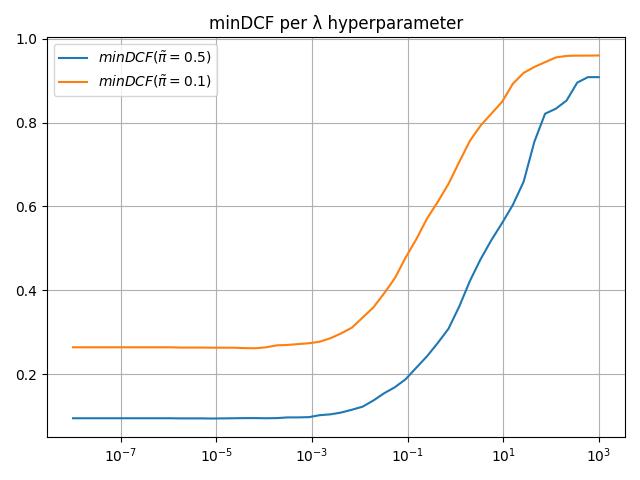
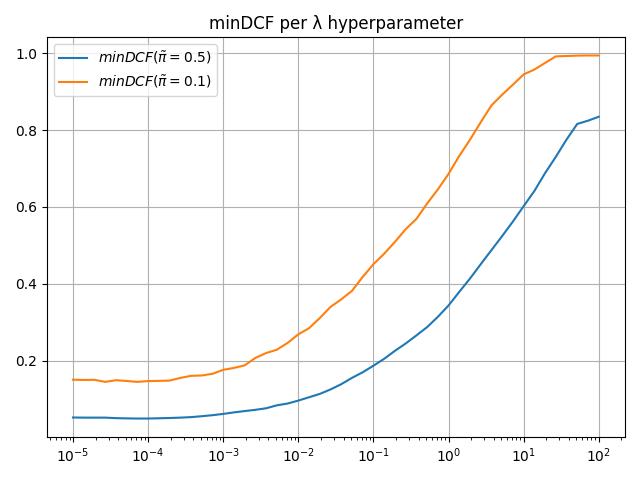
MVG classifier – min DCF on the validation set

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Raw features | | | |
| Full-Cov | 0.048 | 0.125 | 0.124 |
| Tied-Cov | 0.047 | 0.121 | 0.125 |
| Diag-Cov | 0.564 | 0.827 | 0.848 |
| Diag-Tied | 0.567 | 0.824 | 0.848 |
| Guassianized features | | | |
| Full-Cov | 0.059 | 0.187 | 0.174 |
| Tied-Cov | 0.060 | 0.180 | 0.166 |
| Diag-Cov | 0.541 | 0.810 | 0.825 |
| Diag-Tied | 0.540 | 0.805 | 0.816 |
| Z-normalised features | | | |
| Full-Cov | 0.048 | 0.125 | 0.124 |
| Tied-Cov | 0.047 | 0.121 | 0.125 |
| Diag-Cov | 0.564 | 0.827 | 0.848 |
| Diag-Tied | 0.567 | 0.824 | 0.848 |
| PCA(m=11) | | | |
| Full-Cov | 0.094 | 0.264 | 0.221 |
| Tied-Cov | 0.094 | 0.263 | 0.222 |
| Diag-Cov | 0.103 | 0.266 | 0.243 |
| Diag-Tied | 0.104 | 0.269 | 0.234 |

The table shows how for MVG raw and Normalised dataset behave the same, while PCA helps increasing performance of diagonal models (in green). On the other hand, full and tied covariance completely overwhelm the others on every dataset, never having a clear winner. By analysing the a sample couple of covariance matrix produced during the K-Fold protocol of the FullCov MVG we have seen that class T and class F matrices were very similar, so the FullCov model in this case is producing a close to linear separation rule. For this reason, for the next evaluations, we will consider the TiedCov MVG as representative of Gaussian Classifiers.

* Logistic regression model

We will now focus on logistic regression model, firstly by selecting an optimal λ for unregularized dataset. On this dataset there is no need to rebalance the Logistic Regression objective function. Here we can compare the effect of different λ on z-normalized (left) and guassianized (right) features.

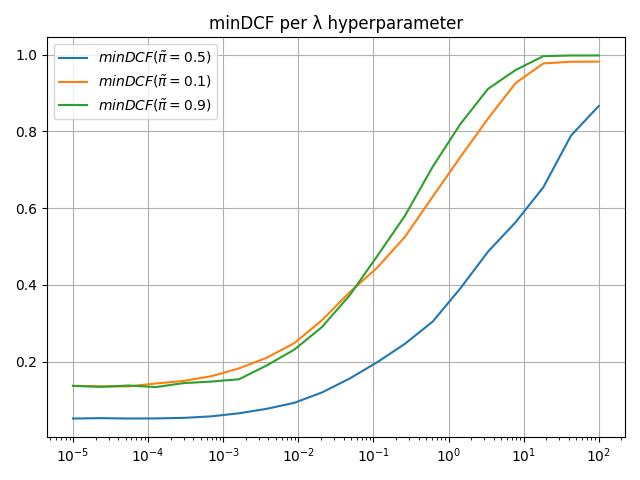


The two diagrams show that gaussianification doesn’t improve performance, instead it reduces those of unbalanced applications. In addition we can see that for λ lower than 10-4 we don’t get any improvement. We will not analyze anymore PCA as it failed on its preferred classifier (the diagonal), so we will go on as this dataset doesn’t require any Naive assumptions. In the next table we will analyze the effect on the chosen λ and a lower value.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Raw | | | |
| Log Reg (λ = 10−5, πT = 0.5) | 0.047 | 0.127 | 0.125 |
| Raw | | | |
| Log Reg (λ = 10−4, πT = 0.5) | 0.047 | 0.127 | 0.125 |
| Log Reg (λ = 10−4, πT = 0.1) | 0.046 | 0.135 | 0.128 |
| Log Reg (λ = 10−4, πT = 0.9) | 0.047 | 0.130 | 0.128 |
| Z-normalized | | | |
| Log Reg (λ = 10−4, πT = 0.5) | 0.046 | 0.136 | 0.129 |
| Guassianized | | | |
| Log Reg (λ = 10−4, πT = 0.5) | 0.055 | 0.162 | 0.160 |
| PCA (m=11) | | | |
| Log Reg (λ = 10−4, πT = 0.5) | 0.095 | 0.266 | 0.218 |

In red are highlighted the overall best values for a regularized dataset. We cans see that a lower λ doesn’t give a big boost, while if we go too low on it, we might over fit our data. Therefore, we select 10-4. In blue it is also highlighted the best score, obtained on the Z-normalized dataset, which is really close to that of Gaussian Classifiers, but pays of on unbalanced application. For this reason, for next comparison, we will take the raw dataset as representative for Logistic Regression. Again gaussianification and PCA didn’t bring any improvement, instead they lowered model performance and training with re-balanced model (in green) had detrimental effect.

Quadratic Log Reg: min DCF for different values of λ on the Z-Normalized dataset:



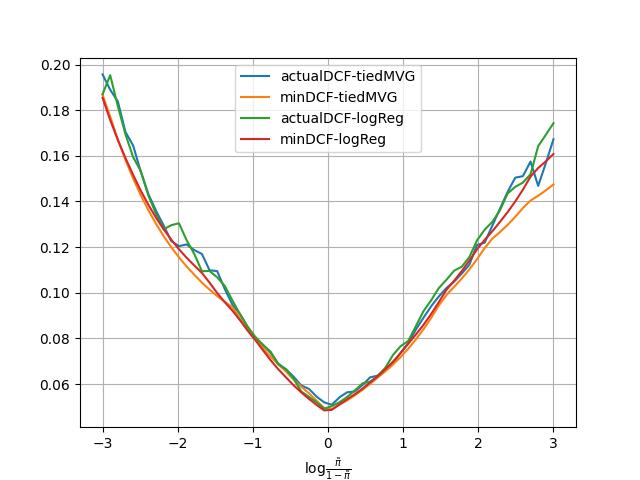
We only applied tuning validation on the z-normalized dataset because it proved to be more stable on models for which we don’t know optimal hyper-parameters. For this classifier the optimal λ appear to be 10-4. We analyzed though 5-fold cross-validation also the effect of re-balanced models for unbalanced applications.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Raw | | | |
| QuadLogReg (λ = 10−4, πT = 0.5) | 0.120 | 0.323 | 0.333 |
| Log Reg (λ = 10−4, πT = 0.5) | 0.047 | 0.127 | 0.125 |
| Z-normalized | | | |
| QuadLogReg (λ = 10−4, πT = 0.5) | 0.052 | 0.150 | 0.139 |
| QuadLogReg (λ = 10−4, πT = 0.1) | 0.055 | 0.154 | 0.145 |
| QuadLogReg (λ = 10−4, πT = 0.9) | 0.054 | 0.148 | 0.141 |
| MVG(FullCov) | 0.048 | 0.125 | 0.124 |
| Guassianized | | | |
| QuadLogReg (λ = 10−4, πT = 0.5) | 0.057 | 0.161 | 0.167 |
| Log Reg (λ = 10−4, πT = 0.5) | 0.055 | 0.162 | 0.160 |
| PCA (m=11) | | | |
| QuadLogReg (λ = 10−4, πT = 0.5) | 0.092 | 0.256 | 0.224 |
| Log Reg (λ = 10−4, πT = 0.5) | 0.095 | 0.266 | 0.218 |

The first thing we denote is that Z-normalized dataset had a positive impact on this classifier with respect to raw features, while re-balancing led to worse results even on unbalanced applications. Although gaussianification brought worse results, PCA appeared to be helpful for this model against its linear counterpart, but still this two pre-processing steps didn’t improve models results.

In conclusion the quadratic Logistic regression doesn’t seem to be a good model, with respect to the previous, but this could be because it seems that linear separation rules fits better for this dataset. In fact, linear Log Reg performs better than quadratic and even MVG with full covariance, which should be a quadratic model, resulted in being close to tied.

Here we can see a Bayes error plot with normalized minimum Detection Cost Function, to show how much and where the Logistic Regression model performs with respect to MVG.



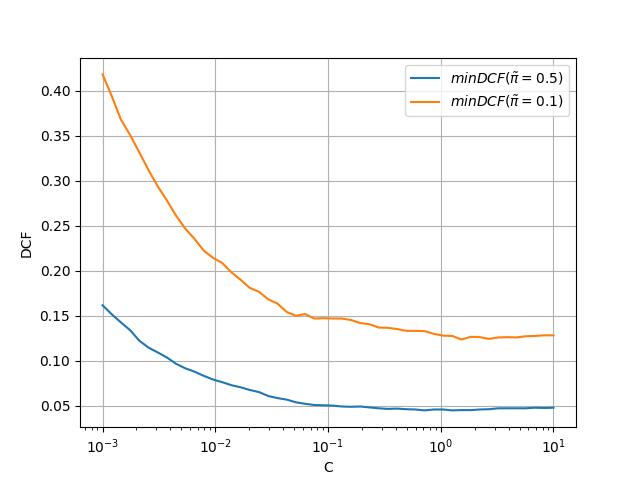
The two models behaves almost the same on minDCF metrics, with the Tied MVG slightly better on higher , and even on actual DCF they appear to be well calibrated, not obvious for the tied Mvg.

Finally, application seems to handle better high , but since it’s of just some cents we will not consider it as a problem.

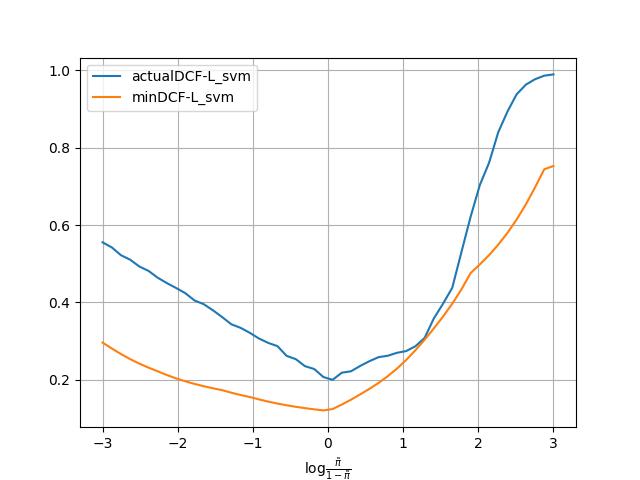
* Support Vector Machine

The main hyper-parameter to tune for a linear kernel support vector machine is the C upper limit for αi values.

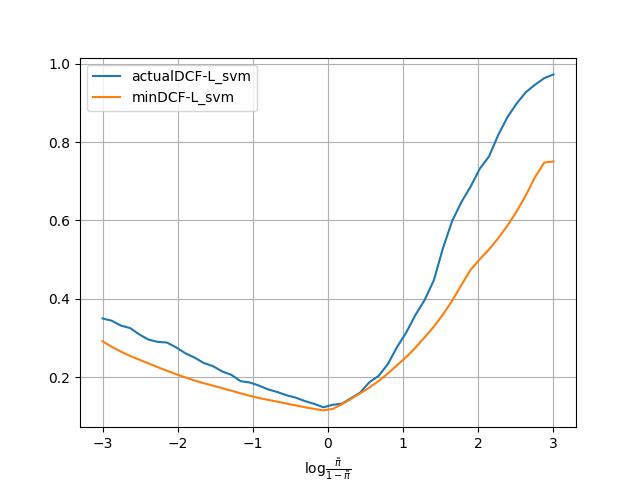
Since for high values of C the Dual problem to minimize becomes computationally too intensive, we looked for the lower value giving an almost good minDCF. Having already found models with a minDCF of 0.111 we chose the first value under 0.12, so C=2, but knowing that one the other hyper-parameters are tuned, we can still increase a bit C at cost of time. The graph below has been obtained with K-Fold (K=4) for C from 10-3 to 101.



Here an error Bayes plot for linear SVM, C = 2 and unregularized C proportions (meaning πT= πemp). MinDCF is like logistic regression and Tied MVG, but actual DCF is much worse. All evaluation still done on the training set, through K-Mean cross validation.



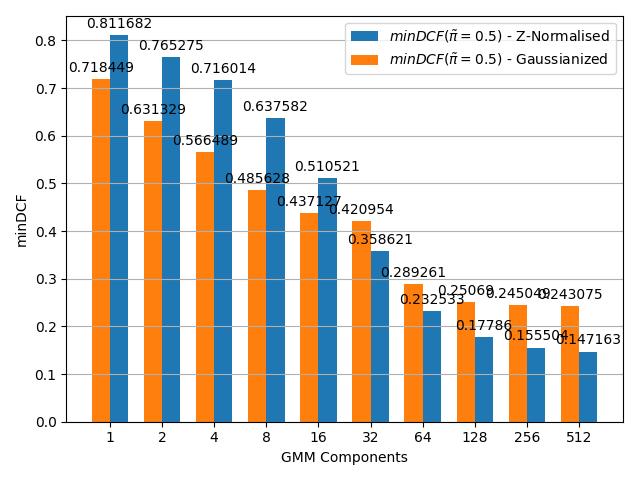
We will now rebalance the Dual problem C bounds and plot the actual DCF but keep K at 1 since tuning also K takes too much time.



We will skip non linear kernel for the Support vector machine, since we already saw the best results are obtained for linear separation surfaces.

* Gaussian Mixture Model

The last generative model is GMM, for which we will only see the Tied Covariance version, following what seen for MVG. Unfortunately, the EM algorithm we have is bugged (log-likelihood ratios increase at some EM iterations), so decisions taken by this model are sub-optimal. Anyway, a GMM trained with the actual algorithm got minDCF close the best linear model for G greater than 256. We will show the results, but not use the model.

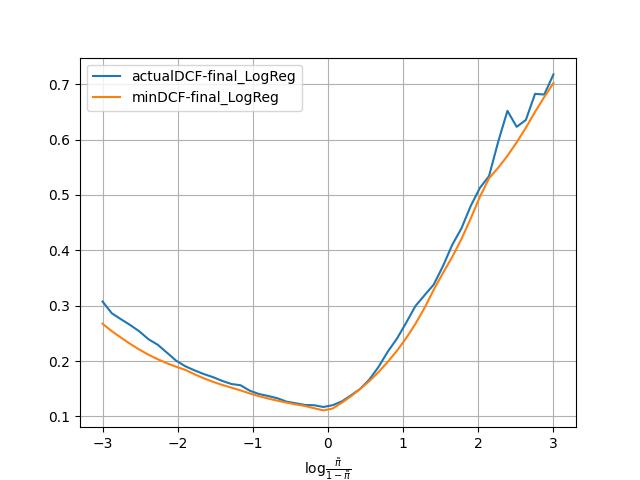


We will choose Logistic Regression because, even though has small differences from the other model, it is already well calibrated and ready to use.

So, our last proposal is Logistic Regression with λ = 10−4 and balanced class. While still keeping an eye on Tied MVG. A possible second solution could also have been a fusion of the two models, maybe with a score calibration

* Experimental result

We applied the chosen model on the test data, in raw, and here the results. As expected the DCF are good.



The actual DCF is 0.114 on the Test set, showing that HRTU2 dataset can be classified by linear classification rules.