Project - Pulsars

Gabriele Cassetta s296284

Outline

HTRU2 is a data set which describes a sample of pulsar candidates collected during the High Time Resolution Universe Survey. Pulsars are a rare type of Neutron star that produce radio emission detectable here on Earth. As pulsars rotate, their emission beam sweeps across the sky. A potential signal detection known as a 'candidate', is averaged over many rotations of the pulsar. In the absence of additional info, each candidate could potentially describe a real pulsar. However in practice almost all detections are caused by radio frequency interference (RFI) and noise, making legitimate signals hard to find.

Machine learning tools are now being used to automatically label pulsar candidates to facilitate rapid analysis.

Features

The data set shared here contains 16,259 spurious examples caused by RFI/noise, and 1,639 real pulsar examples. Each candidate is described by 8 continuous variables.

1. Mean of the integrated profile.
2. Standard deviation of the integrated profile.
3. Excess kurtosis of the integrated profile.
4. Skewness of the integrated profile.
5. Mean of the DM-SNR curve.
6. Standard deviation of the DM-SNR curve.
7. Excess kurtosis of the DM-SNR curve.
8. Skewness of the DM-SNR curve.

Features

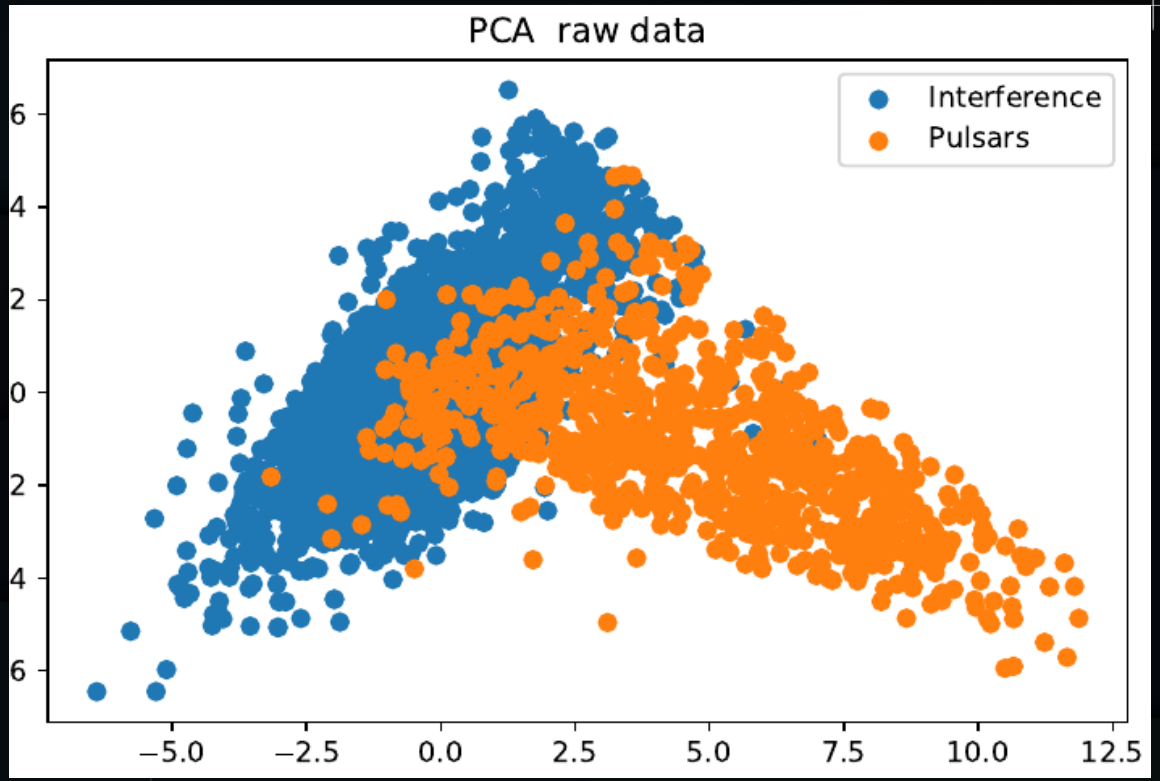
The 8 features come in different scales, having considerably different means and variances, so it is worth applying Z-normalization (centering every feature to its mean and scaling to unit variance).

μ: [110.8, 46.4, 0.4, 1.8, 12.6, 26.2, 8.3, 105.4]
σ: [25.8, 6.8, 1.0, 6.3, 29.5, 19.4, 4.4, 104.3]

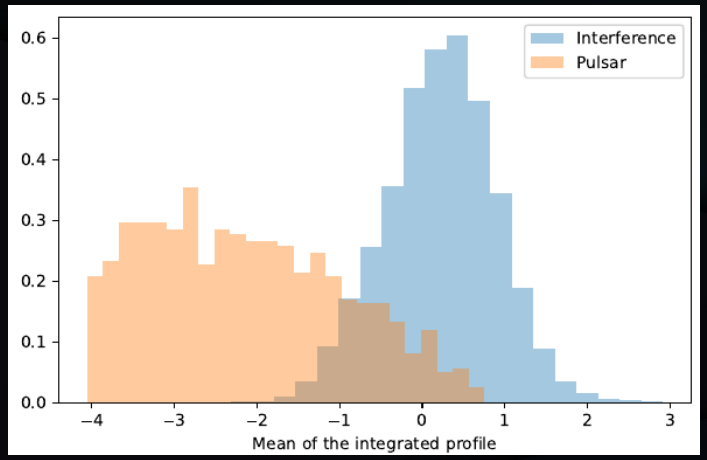
Gaussian classifiers as well as non-regularized logistic regression won’t be affected by this scaling, but other classifiers will, and PCA itself may benefit from it, being no longer biased in its search for directions with the highest variance.

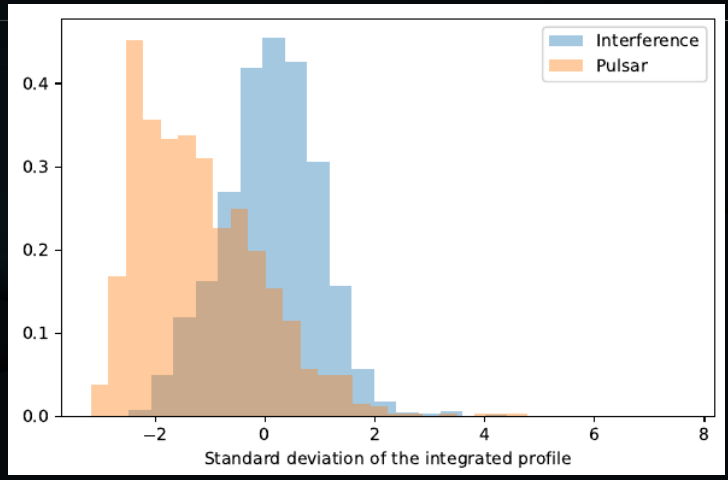
Features

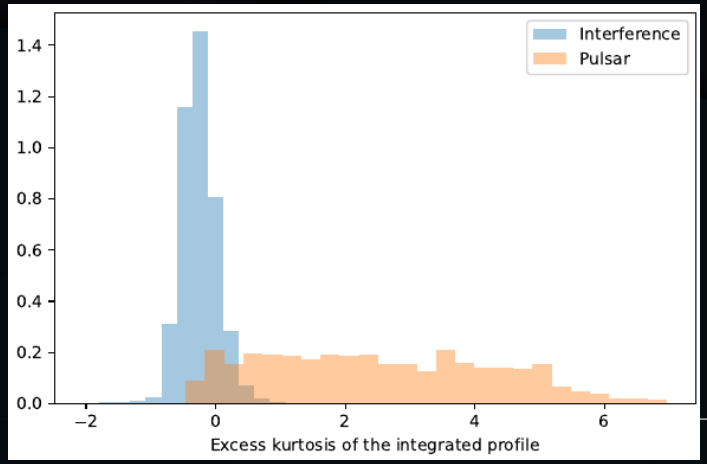
We shall now proceed to plot the data, both with a 2-dimensional PCA scatter and feature-by-feature histograms

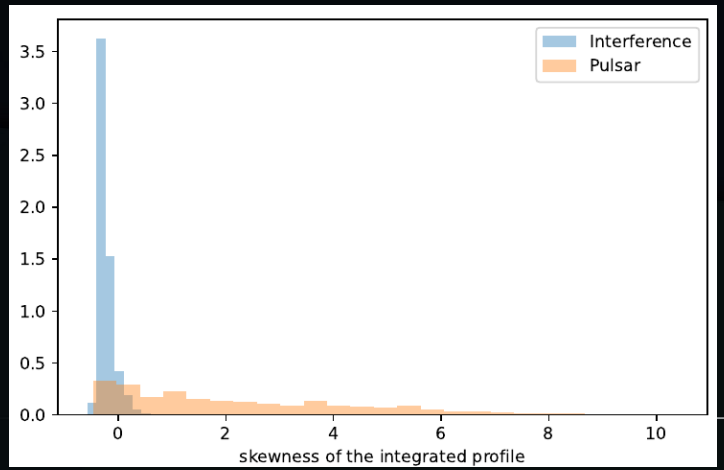


Raw features

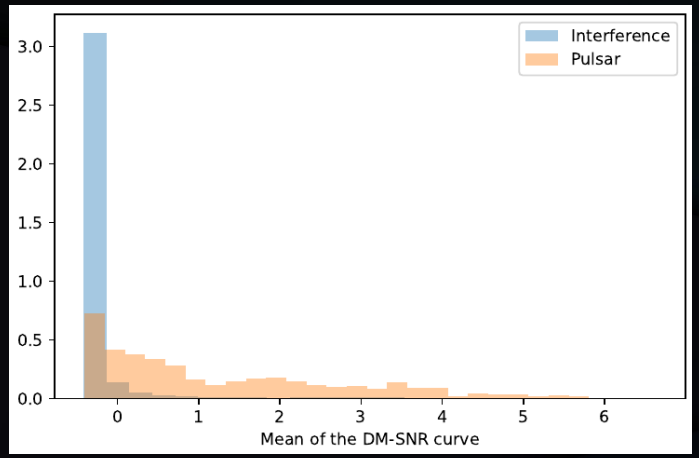


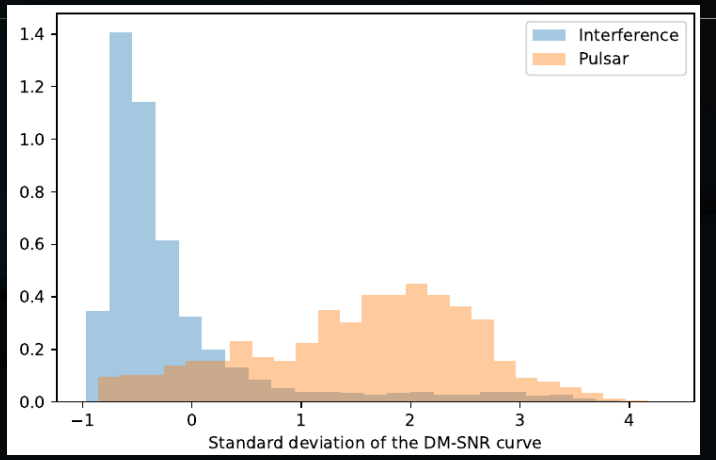


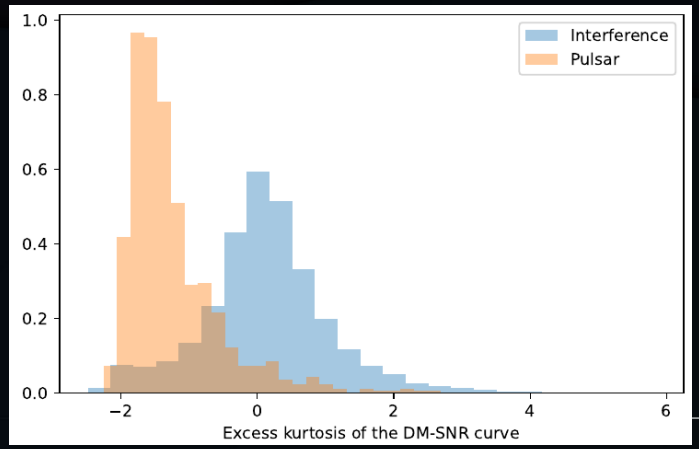


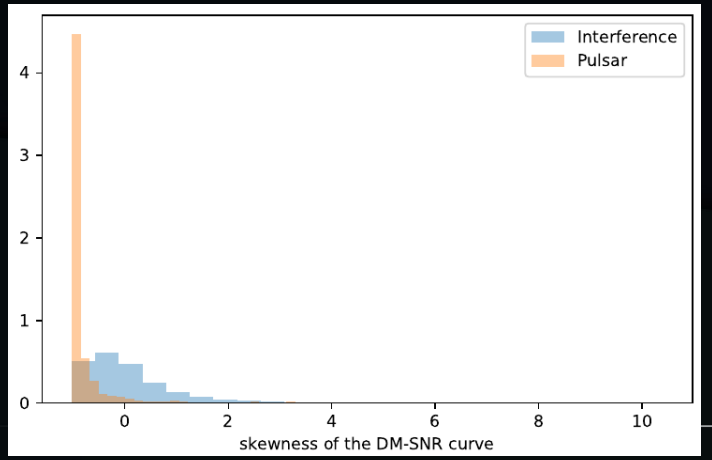


Raw features







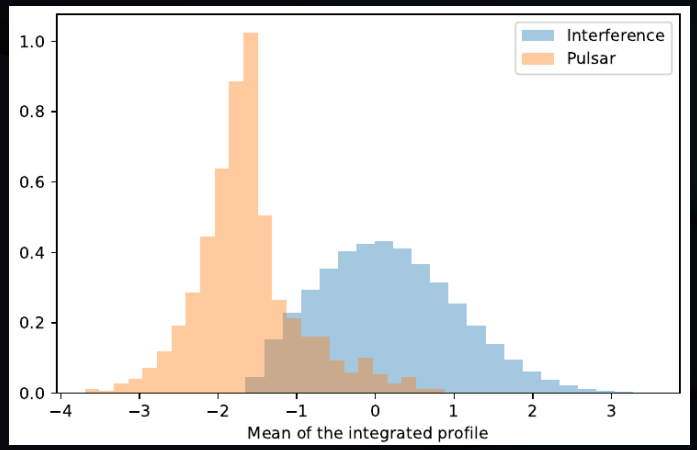


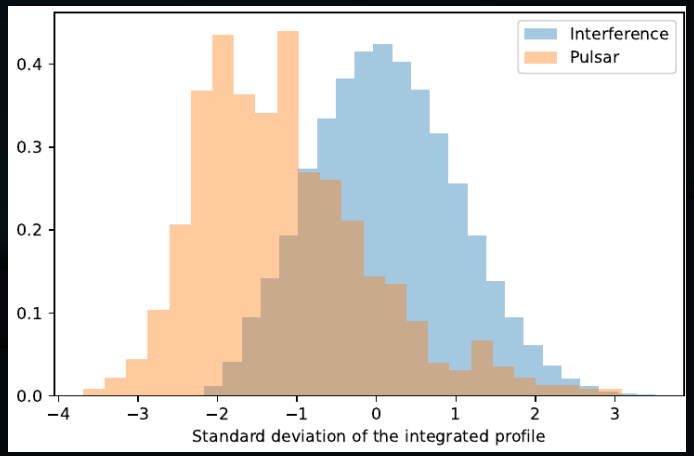
Features

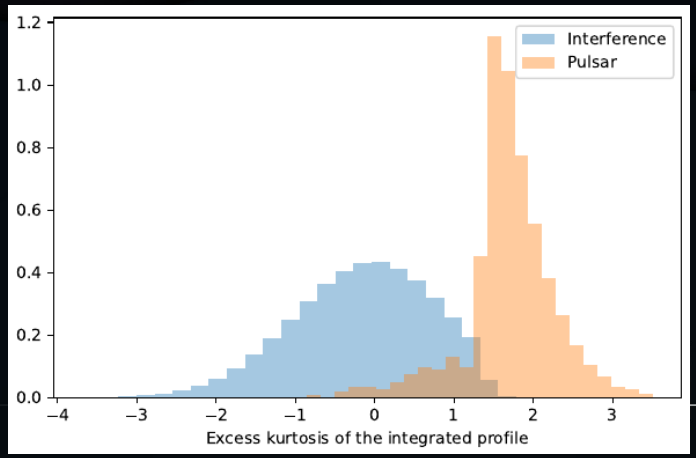
The histograms show that although easily separable, the 8 features have slightly irregular distributions and are affected by some outliers, although not particularly relevant. We may apply gaussianization and see whether gaussian classifiers perform better this way. The transformed data may better fit the assumptions of these classifiers but it may also get corrupted and give a poorer representation of the training data.

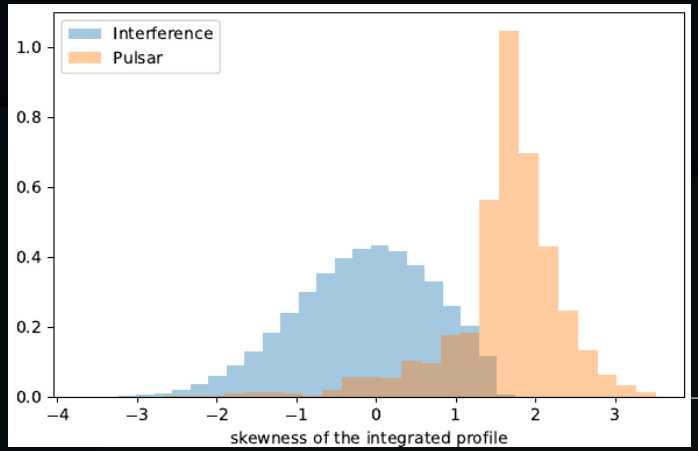
The transformation, of both the training (DTR) and the validation set (DTE), consists in ranking each sample in DTR, dividing by the number of samples in DTR, and then applying the percent point function. It’s important that the ranking be computed in the DTR only, since estimating anything through the DTE would bias the results.

Gaussianized features

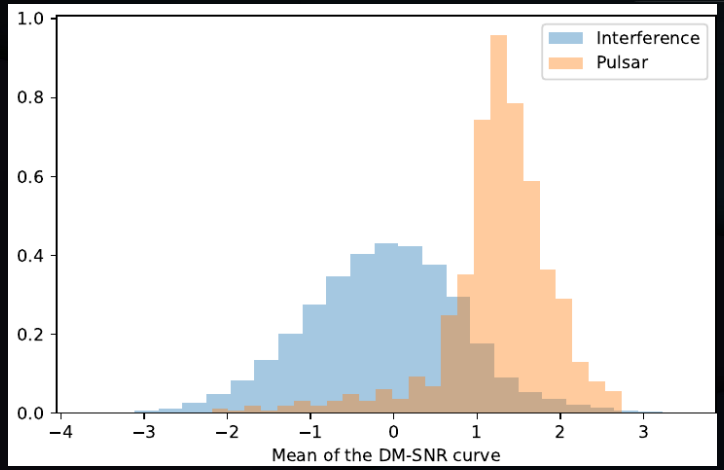


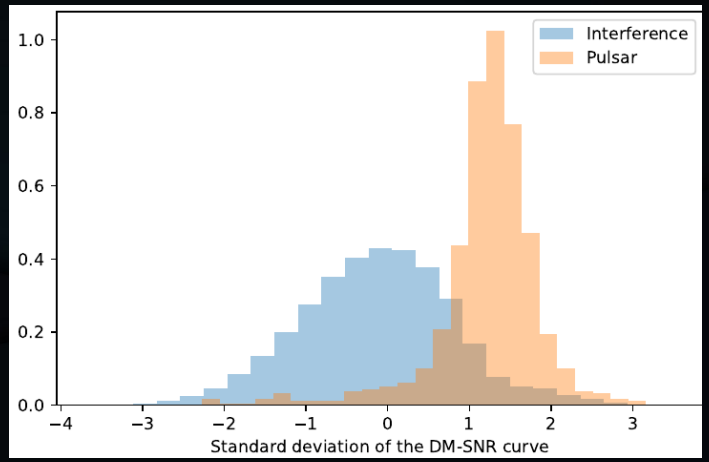


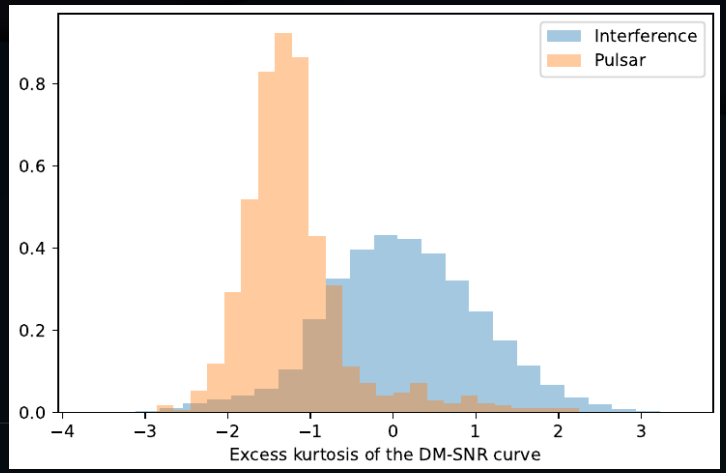


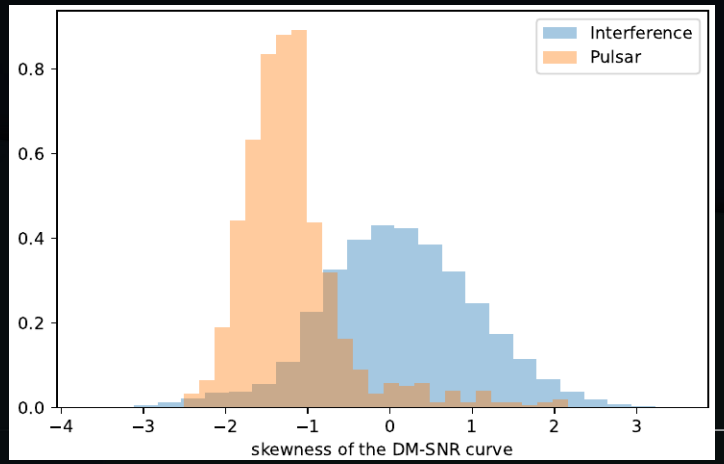


Gaussianized features



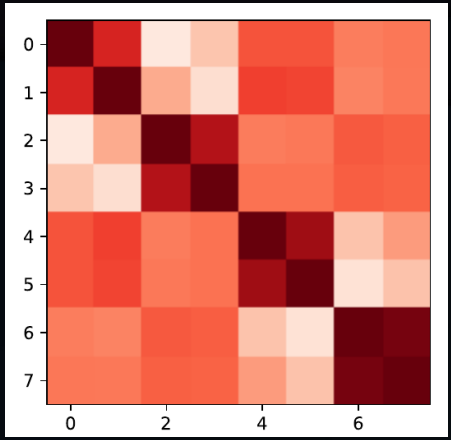




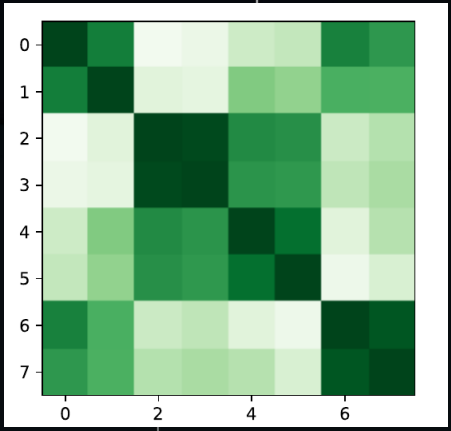


Features

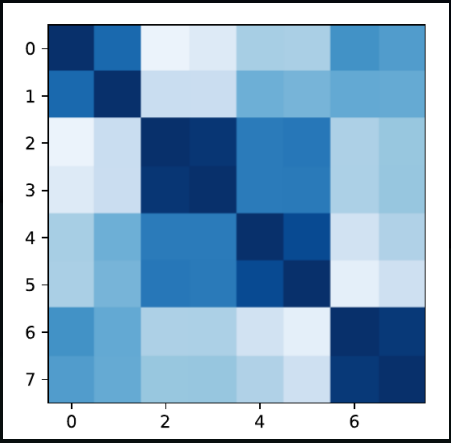
A correlation analysis through the use of Pearson’s coefficient and heatmaps reveals that 1 couple of features (6-7) is strongly correlated for both classes, whereas other couples (0-1, 2-3, 4-5) show different degrees of correlation. We should be safe projecting data into a 7-dimensional space, but we can try with fewer dimensions too and see whether performance stays stable or not.



Interference class



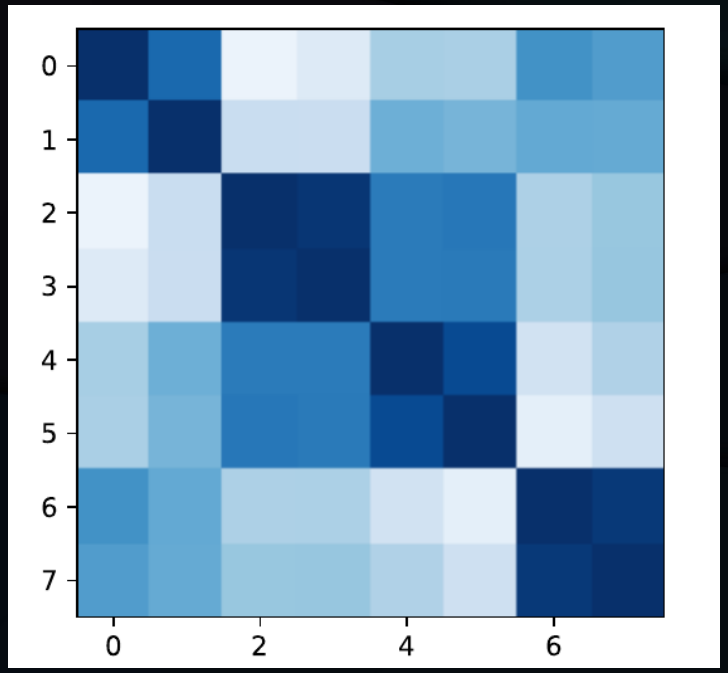
Pulsar class



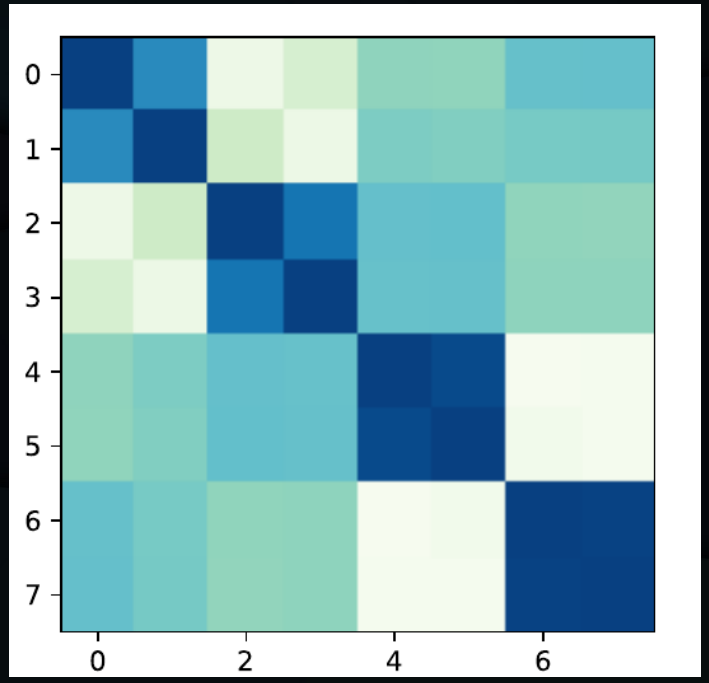
Whole dataset

Features

We can also appreciate the change that gaussianization made correlation-wise. Feature correlation seems to have slightly decreased.



Raw data



Gaussianized data

Gaussian classifiers

We start by considering gaussian classifiers.

With this degree of correlation, we can expect the naive Bayes assumption to poorly fit the dataset. Tying covariances may on the other hand prove useful, since the classifier may inaccurately estimate the covariance of pulsars because of class imbalance. Whether the two classes could be generated by the same noise around their mean, though, is hard to tell, and would require reasonings that are beyond the scope of the project.

Gaussian classifiers

To assess which model performs better we will employ K-fold cross validation.
The value of K will be kept reasonably low, to 3, since a preliminary attempt to process the dataset faced expensive computations with support vector machines and gaussian mixture models. Although gaussian classifiers aren’t as expensive computation time-wise, the value of K must stay coherent between different models in order to compare them.

With K=3, at each iteration we will train the model by using 66% of the training data, previously shuffled so that DTR and DTE have similar distributions of data.

Gaussian classifiers

We shall cross-validate gaussian models with both raw and gaussianized data, as explained earlier.

PCA will be tested too with different values of m, from 7 down to 5, under which we expect performance to drop too drastically, since the number of dimensions is very modest compared to the number of samples.

Just like with gaussianization, it is important that dimensionality reduction be applied to each sample in DTE by using training data only, that is, in this case, by projecting over the m most variant directions found in DTR. Applying PCA on the whole dataset would bias the model, since unseen data would no longer be truly unseen.

Gaussian classifiers

We will consider both balanced and imbalanced applications.
Our main one will have uniform prior:
$(\pi, C_{fp}, C_{fn}) = (0.5, 1, 1)$

The imbalanced ones will be:
$(\pi, C_{fp}, C_{fn}) = (0.1, 1, 1)$
$(\pi, C_{fp}, C_{fn}) = (0.9, 1, 1)$

Where π is the effective prior, i.e. a bias towards the positive class in respect to the negative one. In looking for the most promising model, we will compute the minDCF, i.e. the empirical Bayes cost that we would pay if we made optimal decisions for the validation set.

Gaussian classifiers

Raw data – no PCA

	π=0.5	π=0.1	π=0.9
Full-Cov	0.142	0.285	0.661
Diag-Cov	0.193	0.315	0.736
Tied Full-Cov	0.112	0.224	0.574
Tied Diag-Cov	0.160	0.265	0.580

Gaussianized data – no PCA

	π=0.5	π=0.1	π=0.9
Full-Cov	0.153	0.241	0.696
Diag-Cov	0.154	0.278	0.603
Tied Full-Cov	0.133	0.233	0.534
Tied Diag-Cov	0.163	0.293	0.607

The Tied Full-Cov model on raw data yields the best results. Apparently the model had a hard time estimating class-specific covariance matrices, so tying covariances has proved useful.
We can also see how gaussianization yielded significantly better results, in respect to raw data, under the naive Bayes assumptions, since the transformation seems to have decreased feature correlation by slightly whitening the covariance matrices, as previously seen in heatmaps.

Gaussian classifiers

Raw data – m = 7

	π=0.5	π=0.1	π=0.9
Full-Cov	0.140	0.304	0.629
Diag-Cov	0.214	0.507	0.716
Tied Full-Cov	0.112	0.223	0.569
Tied Diag-Cov	0.140	0.272	0.594

Gaussianized data – m = 7

	π=0.5	π=0.1	π=0.9
Full-Cov	0.153	0.242	0.697
Diag-Cov	0.167	0.247	0.659
Tied Full-Cov	0.135	0.240	0.546
Tied Diag-Cov	0.137	0.255	0.568

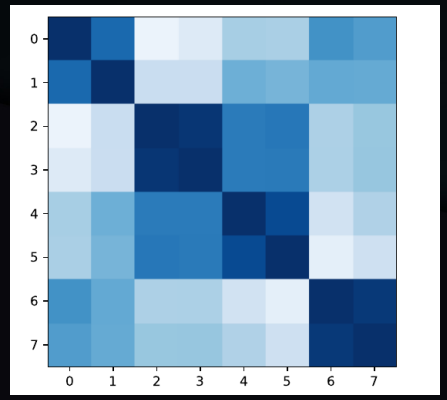
As expected, reducing dimensions down to 7 does not particularly degrade performance, likely because of the highly-correlated features 6 and 7.
Tied Diag-Cov seems to particularly benefit from PCA, possibly due to the new features having more similar class-specific diagonal covariances. Diag-Cov on the other hand performs worse, since PCA increases class-specific correlation by discarding the directions with low variance, although flattening it over the whole dataset. We can visualize the difference with heatmaps of the raw transformed space.

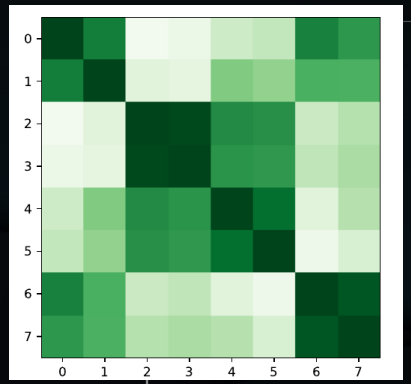
Gaussian classifiers

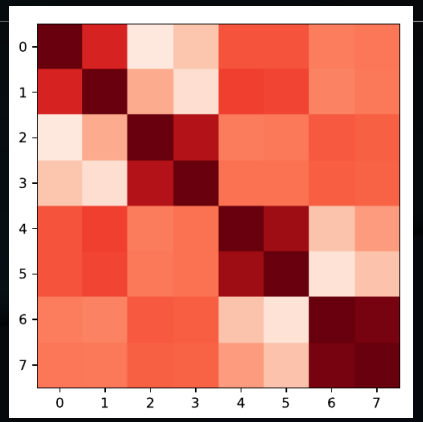
Raw data

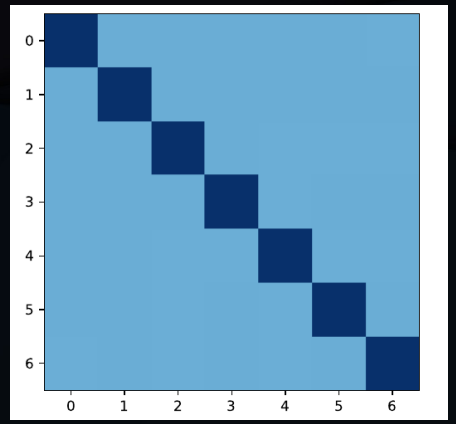
No PCA

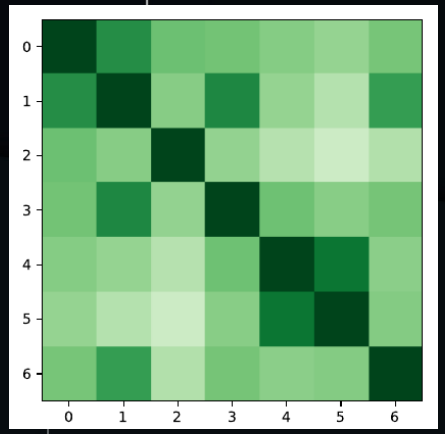
m = 7

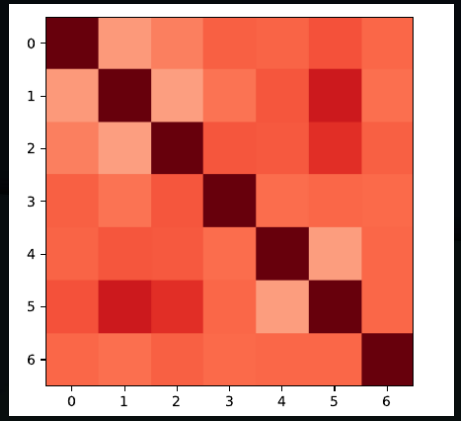












Whole dataset

Pulsar class

Interference class

Gaussian classifiers

Raw data – m = 6

	π=0.5	π=0.1	π=0.9
Full-Cov	0.152	0.292	0.638
Diag-Cov	0.222	0.531	0.728
Tied Full-Cov	0.138	0.254	0.577
Tied Diag-Cov	0.163	0.302	0.602

Raw data – m = 5

	π=0.5	π=0.1	π=0.9
Full-Cov	0.150	0.256	0.653
Diag-Cov	0.220	0.460	0.749
Tied Full-Cov	0.149	0.263	0.568
Tied Diag-Cov	0.171	0.313	0.595

As we can see, further reductions in the feature space worsens performance.
A 7-dimensional space did not worsen performance either, but since the features/samples ratio isn't too high, we shall keep every feature.

Our best model so far is Tied Full-Cov with no PCA and raw data, although it provides quite bad results for imbalanced applications (with a strong bias towards pulsars)

Logistic regression

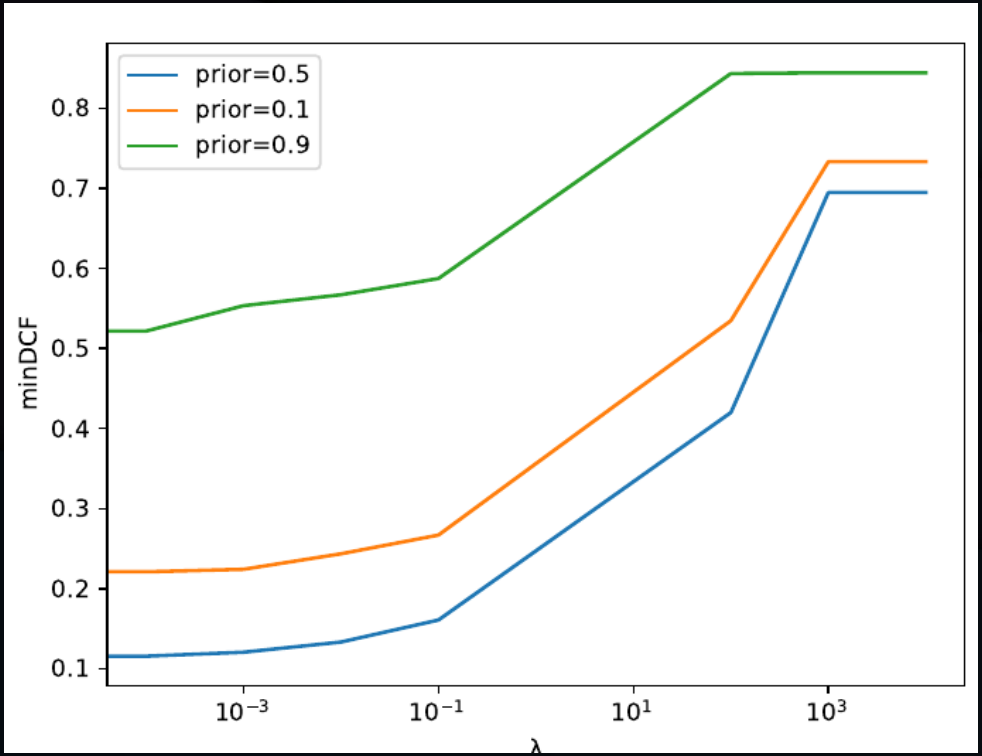
We'll now turn to discriminative approaches, in particular to regularized linear logistic regression. Given the good results obtained by tied-covariance gaussian models, we can expect other linear classifiers to work well.

Given the severe class imbalance, we rebalance the costs of the two classes, obtaining the objective function:

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

We consider our main application and thus set π_T to 0.5 and try different values of λ.

Logistic regression



Raw data

Regularization provides no benefit, so we will keep λ=0. Since logistic regression does not require specific assumptions on the data distribution, and given the poor results obtained for MVG, gaussianization is not of particular interest, but we may still try it out.

We may now try the model with different priors π_T and reduced spaces.

Logistic regression

Linear LR performs similarly to MVG Tied Full-Cov. Using different π_T only slightly improves the model for imbalanced applications. Reducing to 7 dimensions, once again, does not degrade performance.

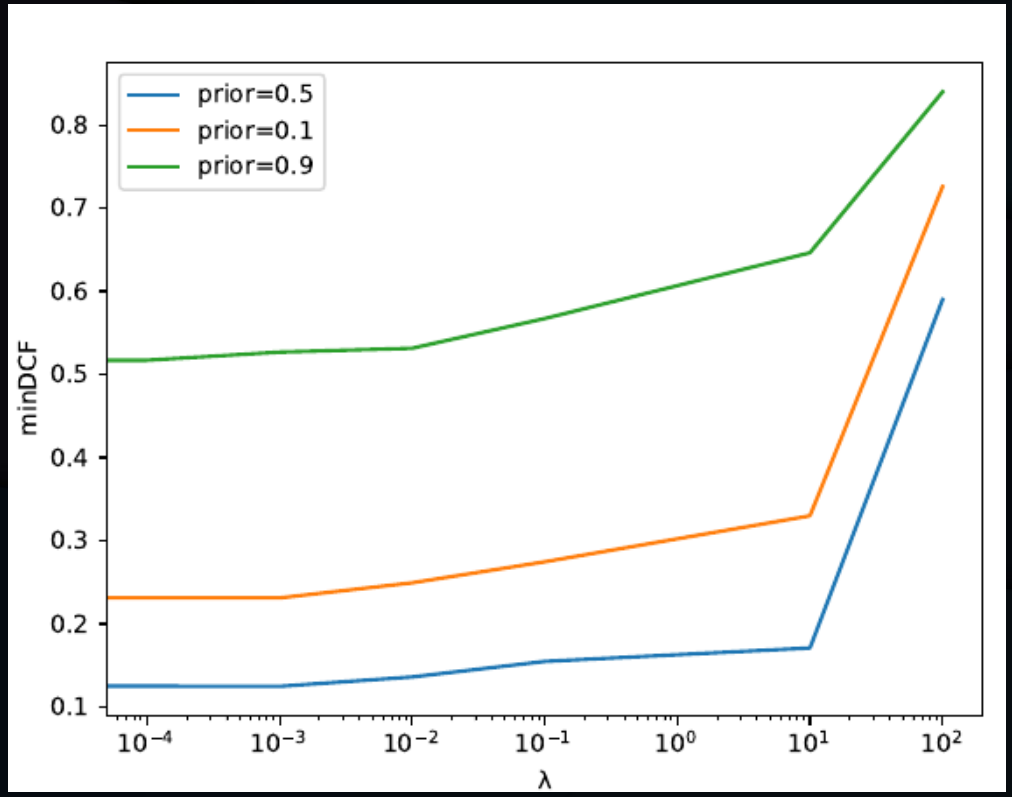
	$\overline{\pi}$=0.5	$\overline{\pi}$=0.1	$\overline{\pi}$=0.9
MVG Tied Full-Cov	0.112	0.224	0.574
LR π_T = 0.5 raw	0.116	0.219	0.519
LR π_T = 0.1 raw	0.115	0.216	0.541
LR π_T = 0.9 raw	0.120	0.223	0.517

No PCA

	$\overline{\pi}$=0.5	$\overline{\pi}$=0.1	$\overline{\pi}$=0.9
MVG Tied Full-Cov	0.112	0.223	0.569
LR π_T = 0.5 raw	0.116	0.219	0.539
LR π_T = 0.1 raw	0.117	0.216	0.550
LR π_T = 0.9 raw	0.117	0.216	0.528

PCA – m=7

Logistic regression



Gaussianized data

	$\overline{\pi}$=0.5	$\overline{\pi}$=0.1	$\overline{\pi}$=0.9
LR π_T = 0.5 raw	0.116	0.219	0.519
LR π_T = 0.5 gau	0.125	0.231	0.516
LR π_T = 0.1 gau	0.128	0.226	0.515
LR π_T = 0.9 gau	0.127	0.236	0.513

Gaussianization, just like on MVG, yields slightly worse results than raw data. Apparently the transformation slightly corrupts the integrity of the distribution. Given the poorer results obtained by quadratic gaussian classifiers, we may dismiss quadratic LR and turn to linear support vector machines.

Support Vector Machines

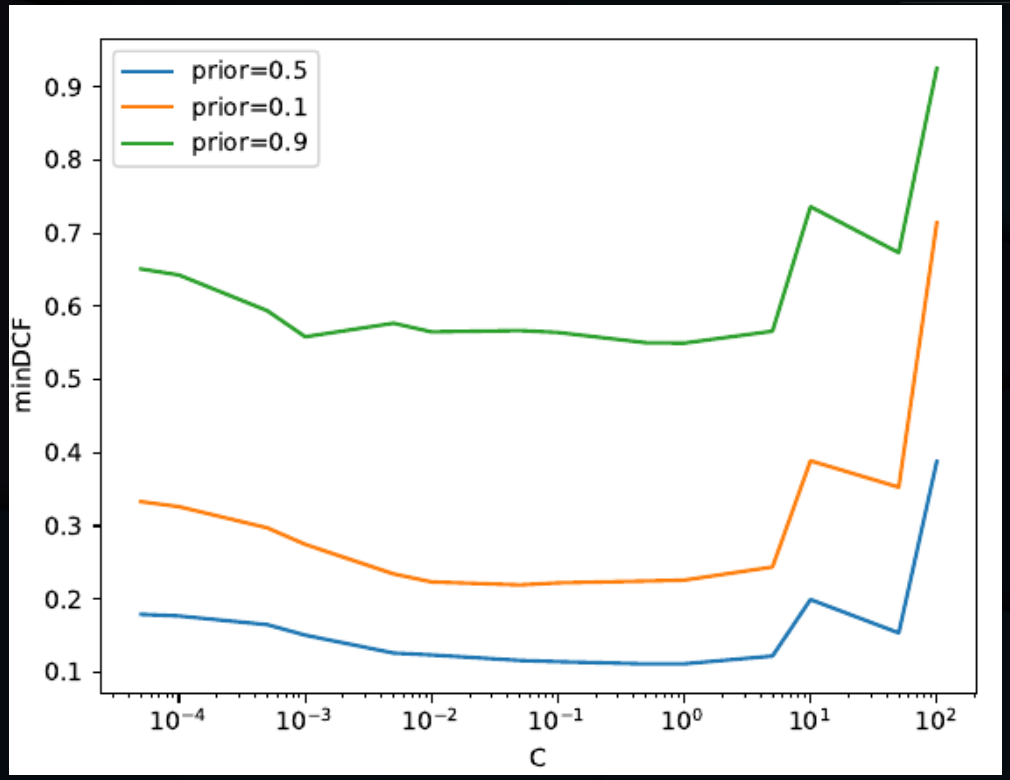
For linear SVM we need to tune the hyperparameter C through cross-validation. Given the severe class imbalancing, we can employ a different value of C for the two classes.

$$C_T = C \frac{\pi_T}{\pi_T^{emp}}$$

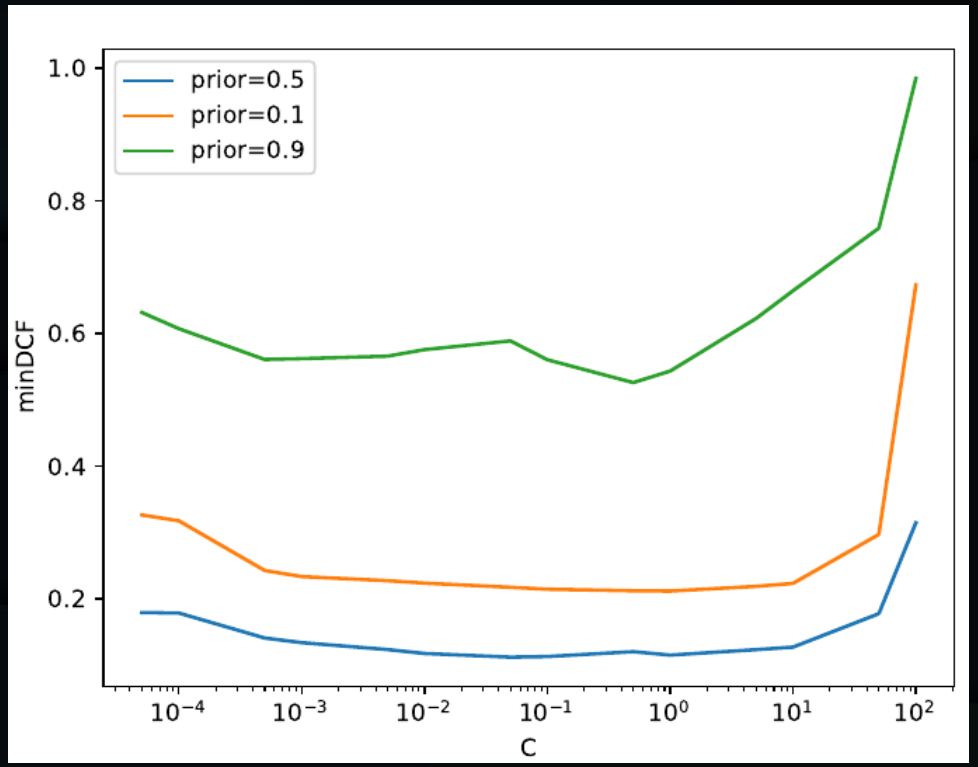
$$C_F = C \frac{\pi_F}{\pi_F^{emp}}$$

Where π^{emp} is the empirical prior, i.e. sample proportions in the training set.

Support Vector Machines



With class balancing



Without class balancing

Support Vector Machines

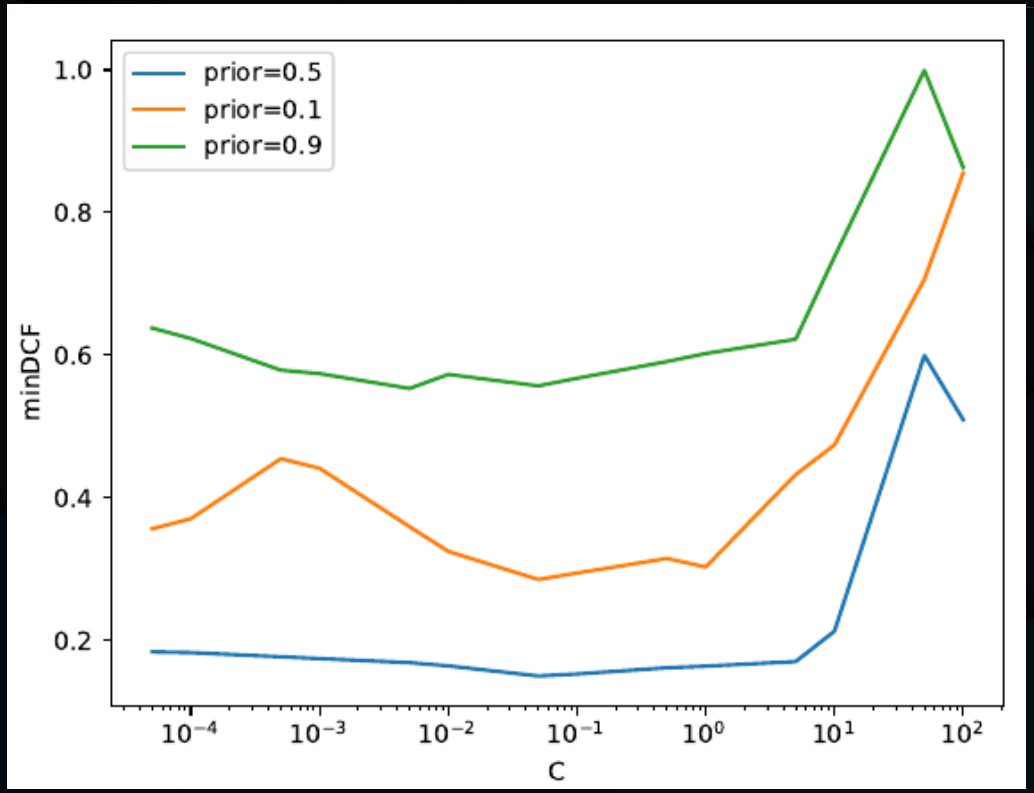
Class balancing doesn't seem crucial, whereas a good value for C seems to be 1.

	$\overline{\pi}$=0.5	$\overline{\pi}$=0.1	$\overline{\pi}$=0.9
MVG Tied Full-Cov	0.112	0.224	0.574
LR π_T = 0.5	0.116	0.219	0.519
SVM C=1 π_T=0.5	0.111	0.226	0.550
SVM C=1	0.115	0.212	0.543

As expected, linear SVM performs well, similarly to other linear classifiers. Class balancing seems to obtain overall similar results to standard SVM, so we will keep it.

We may now try to employ quadratic and RBF kernels, although poor results are to be expected given how linear classifiers seem to well fit the dataset.

Support Vector Machines

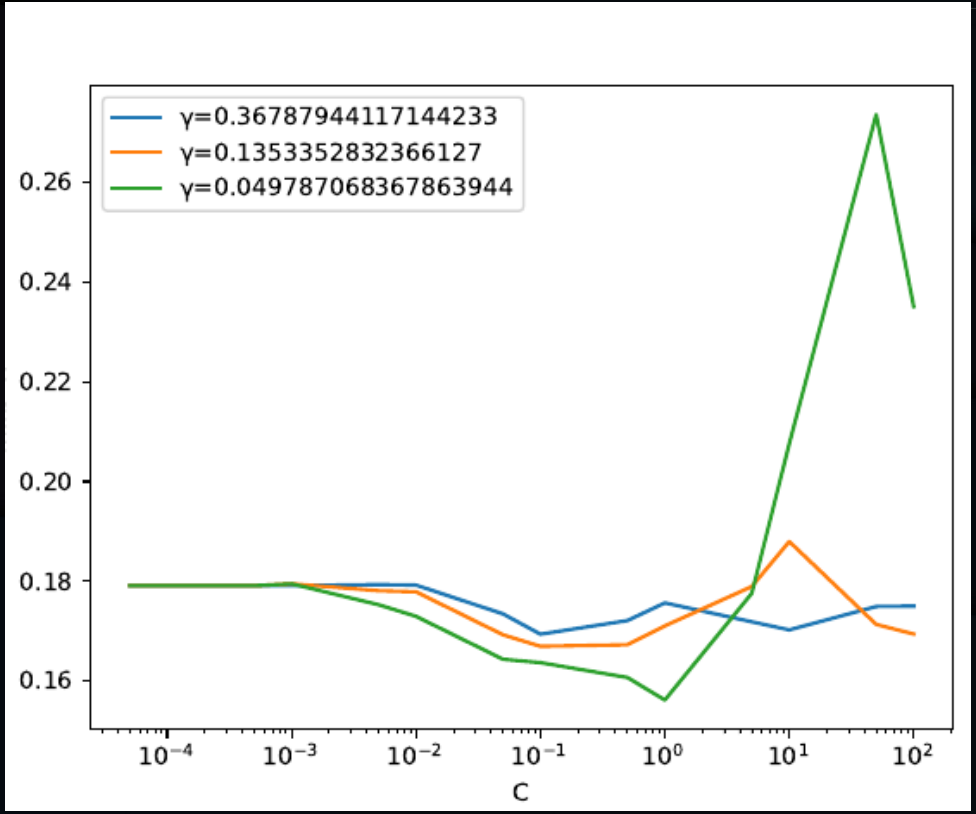


Quadratic SVM

	$\overline{\pi}$=0.5	$\overline{\pi}$=0.1	$\overline{\pi}$=0.9
MVG Tied Full-Cov	0.112	0.224	0.574
LR π_T = 0.5	0.116	0.219	0.519
SVM π_T=0.5	0.111	0.226	0.550
Q-SVM C=0.1 π_T=0.5	0.153	0.294	0.568

After choosing a good value of C, we can see how quadratic SVM yields poorer results than linear SVM, as expected.

Support Vector Machines



RBF-SVM

	$\widetilde{\pi}$=0.5	$\widetilde{\pi}$=0.1	$\widetilde{\pi}$=0.9
MVG Tied Full-Cov	0.112	0.224	0.574
LR π_T = 0.5	0.116	0.219	0.519
SVM π_T=0.5	0.111	0.226	0.550
RBF-SVM $\gamma=e^{-3}$ C=1 π_T=0.5	0.156	0.272	0.560

After picking the most promising values of C and γ, corresponding to the minimum of the green curve, we can see, as expected, that a RBF kernel obtains similar (sub-optimal) results as quadratic SVM.

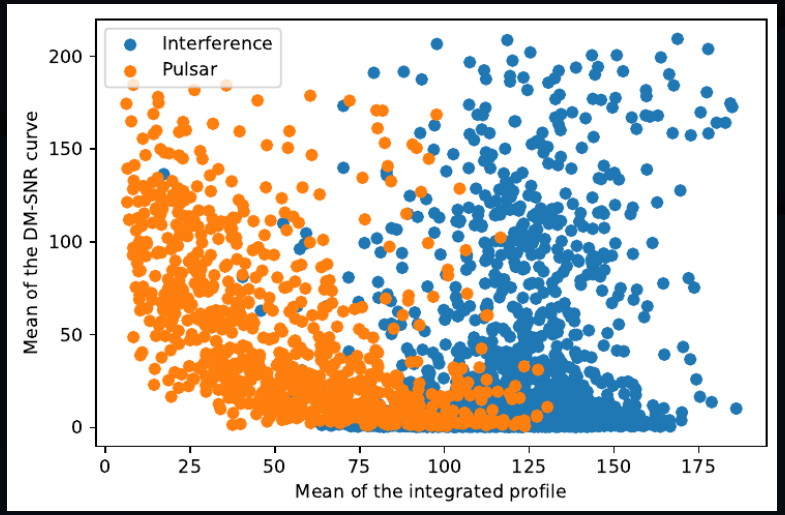
Gaussian Mixture Models

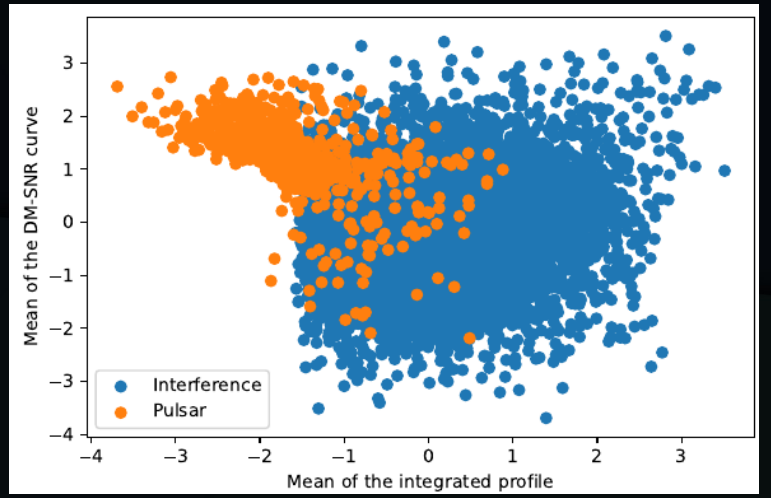
We shall now resort to Gaussian Mixture Models. A GMM can approximate generic distributions, so we can expect it to perform as well as MVG models, or even better.
Tied MVG gave the best results earlier on, but in GMM, unlike in MVG, tying takes place at in-class level, so we shouldn’t expect it to necessarily perform better.

A preliminary attempt to validate the model has faced very expensive computations, considerably more than SVM, so we shall keep the number of components to a reasonably small value, by trying both naive and tied models.

Gaussian Mixture Models

Gaussianization won't necessarily provide any benefit, given how it reduces the dynamic range of the features. The LGB algorithm finds different clusters at each iteration and splitting of the components, but since each component has its own gaussian, it may end up finding clusters that are not the actual ones of our dataset.





Scatters of features 1 and 5, raw on the left, gaussianized on the right

Gaussian Mixture Models

With a limited number of components, the full-covariance model is able to give an accurate estimate of each component's covariance matrix. With a bigger number of components, we may expect tied models to perform better, since they provide a more precise covariance estimate when the number of samples per component is low.

	4 Gau	8 Gau
Full cov	0.137	0.115
Tied full cov	0.142	0.155
Diag cov	0.198	0.160
Tied diag cov	0.177	0.158

Raw

	4 Gau	8 Gau
Full cov	0.141	0.141
Tied full cov	0.172	0.170
Diag cov	0.188	0.167
Tied diag cov	0.188	0.169

Gaussianized π_T=0.5

Score calibration

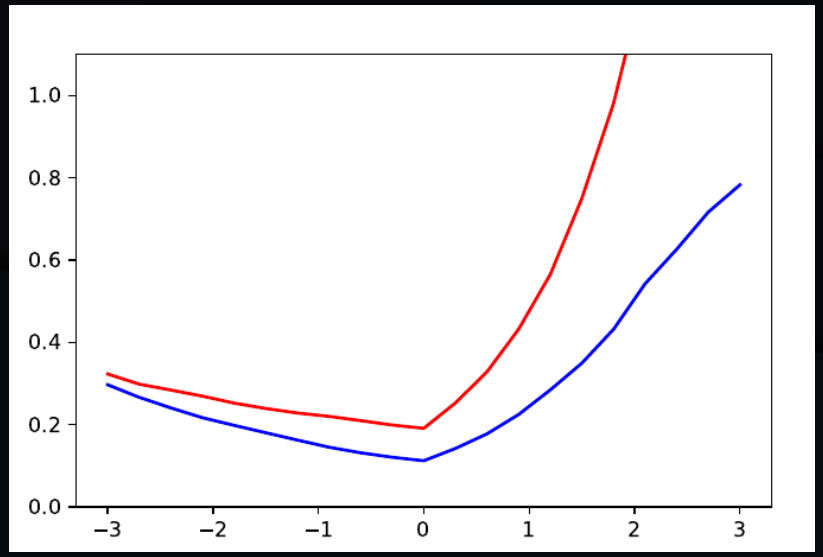
We have so far only considered the minDCF, i.e. the empirical Bayes cost that we would pay if we made optimal decisions using the scores provided by our recognizers. In the binary case, the optimal decision consists either in choosing a good threshold to compare log-likelihoods ratios with, or in recalibrating the scores so that the optimal threshold becomes the theoretical one:

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

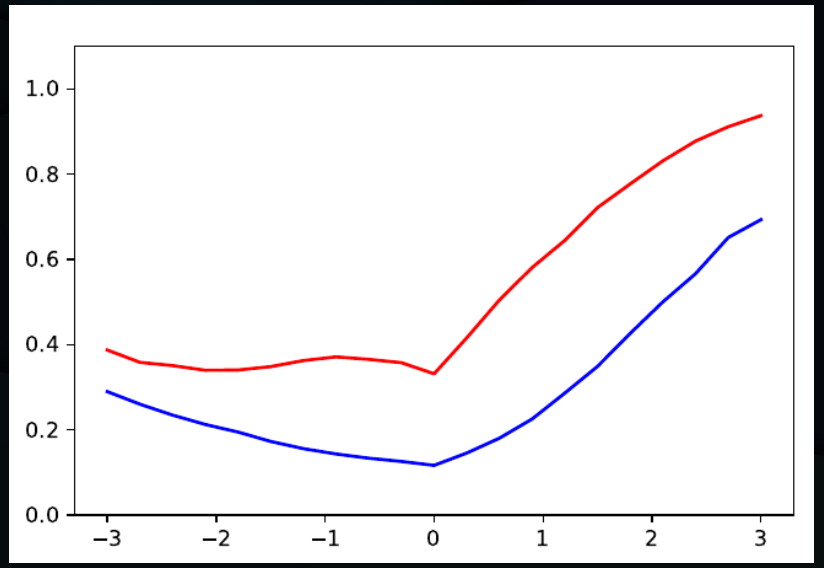
Let’s now consider the difference between the minDCF and the actual DCF for the best models we have tested so far.

Score calibration

Bayes error plots show a considerable distance between minDCF and actDCF for both MVG and LR. For unbalanced applications, the difference is even more remarkable.



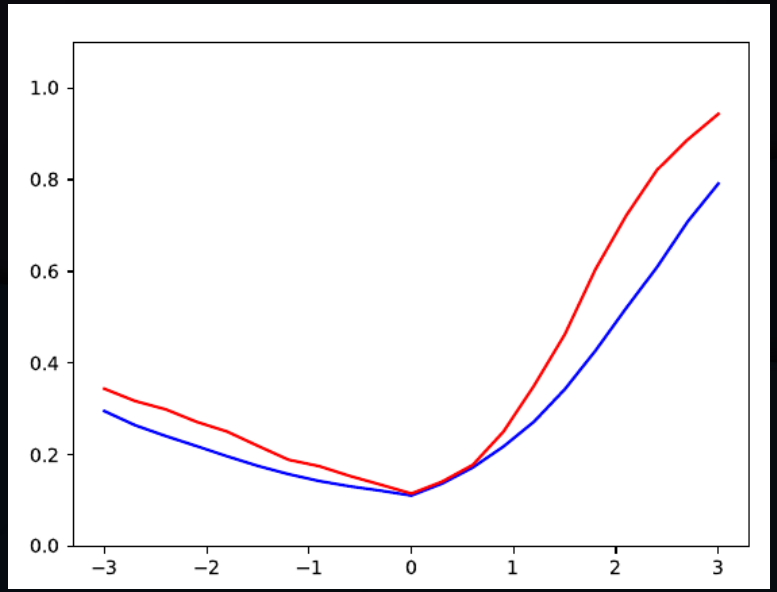
MVG Tied Full cov



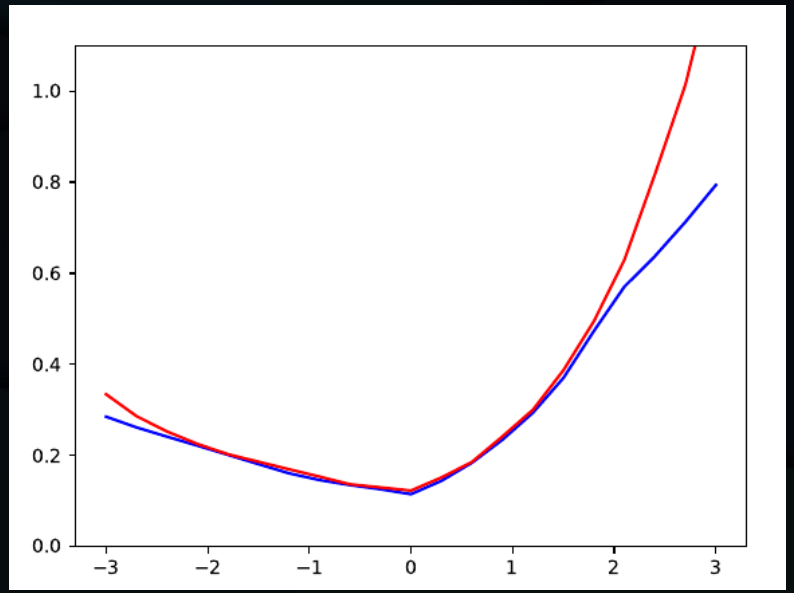
Linear LR λ=0

Score calibration

On the other hand SVM, although having no probabilistic interpretation, yields decently calibrated scores. GMM performs even better.



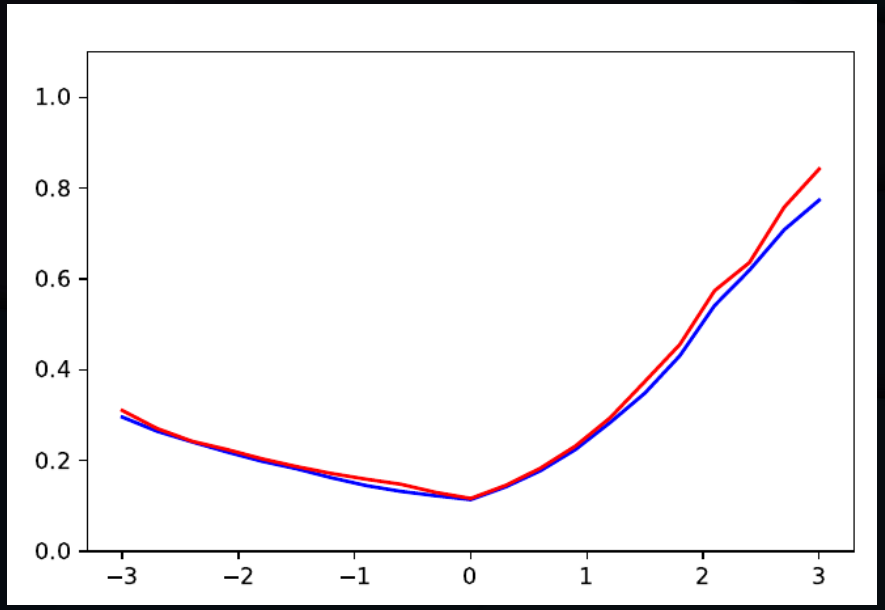
Linear SVM C=1 with balancing



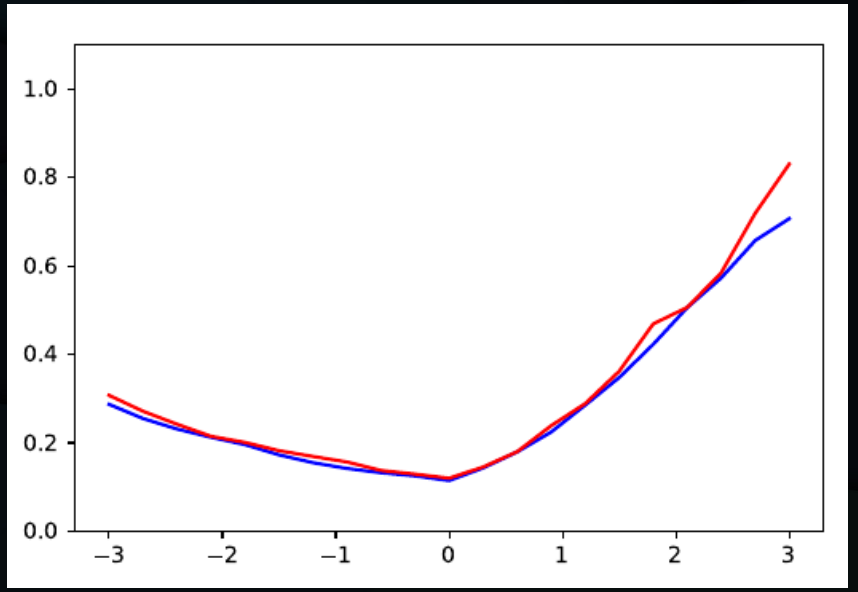
GMM Full Cov 8 Gau

Score calibration

After applying Kfold and logistic regression to the scores, we get much better calibration for all applications.

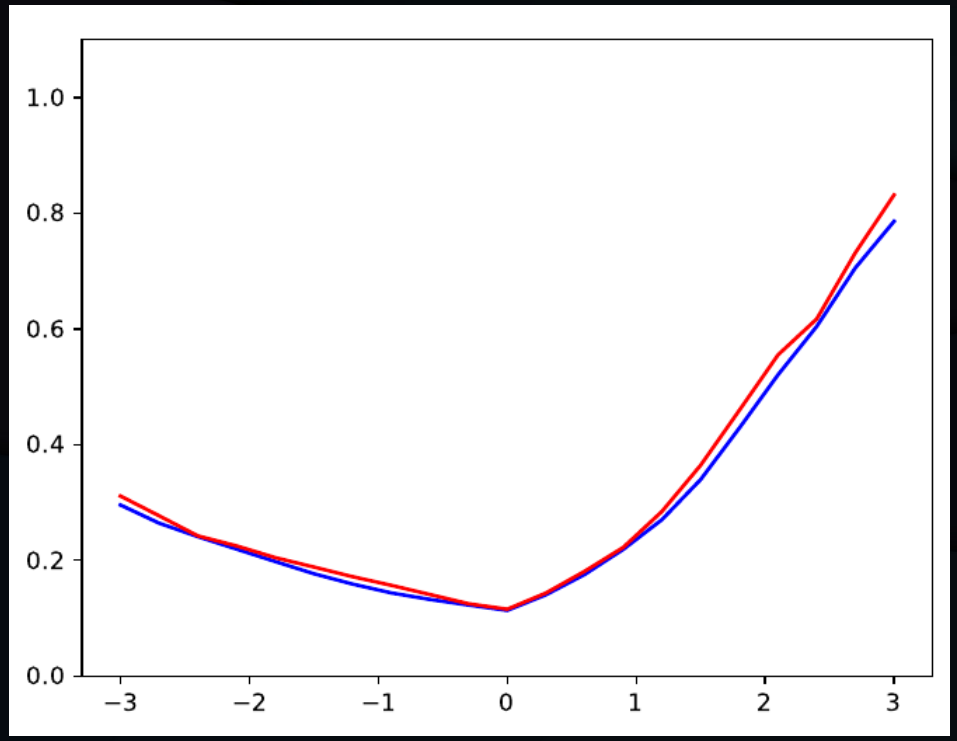


MVG Tied Full cov

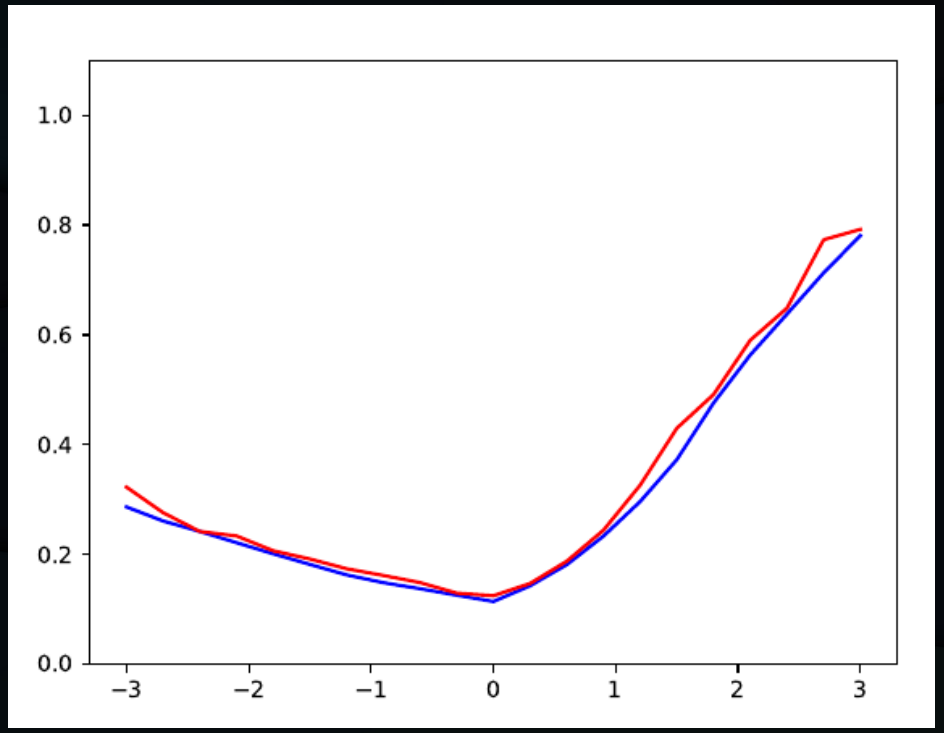


Linear LR λ=0

Score calibration



Linear SVM C=1 with balancing



GMM Full Cov 8 Gau

Experimental results

We now focus on trying out the in-so-far best classifiers on the test dataset. We will yet again consider minDCF measures in order to verify whether the proposed solution can indeed achieve the best accuracy.

We can expect the new minDCF measures to be slightly different than the previous ones, since the distribution of the data is not exactly the same in the training and test set. If the distribution is similar enough and our assumptions were correct, though, the difference should be very small.

Experimental results

	π=0.5	π=0.1	π=0.9
MVG Tied Full Cov	0.109	0.207	0.590
Linear LR λ=0	0.107	0.198	0.542
SVM C=0 π_T=0.5	0.110	0.198	0.540
GMM Full Cov 8Gau	0.110	0.218	0.586

Test data yielded very similar results to the training set, if not slightly better. Linear models as well as gaussian mixture ones proved to fit this dataset.

End