PULSAR DETECTION

MACHINE LEARNING AND PATTERN RECOGNITION

MILAD BEIGI

Turin, Italy

Table of Contents

Introduction

Description

Pulsars are kind of Neutron stars and considerably interesting for scientific research. As this exceptional kind of star produces radio emissions detectable here on Earth, machine learning tools can be used to label pulsar candidates to facilitate rapid analysis automatically. Classification systems are widely implemented, considering the candidate data sets as **binary classification problems**.

Candidate information

Each candidate is described by eight **continuous variables** and a single class variable. The first four are simple statistics obtained from the integrated pulse profile (folded profile). This is an array of continuous variables that describe a longitude-resolved version of the signal that has been averaged in both time and frequency. The remaining four variables are similarly obtained from the DM-SNR curve.

1. Mean of the integrated profile.
2. The standard deviation of the integrated profile.
3. Excess kurtosis of the integrated profile.
4. The skewness of the integrated profile.
5. Mean of the DM-SNR curve.
6. The standard deviation of the DM-SNR curve.
7. Excess kurtosis of the DM-SNR curve.
8. The skewness of the DM-SNR curve.
9. Class

Classification task

Given the features above, our final task is to determine whether each of the samples is a Pulsar candidate or not. So, we are faced with a **binary classification problem,** and we need to build different classifiers, analyze them, and compare their performance and cost for various applications.

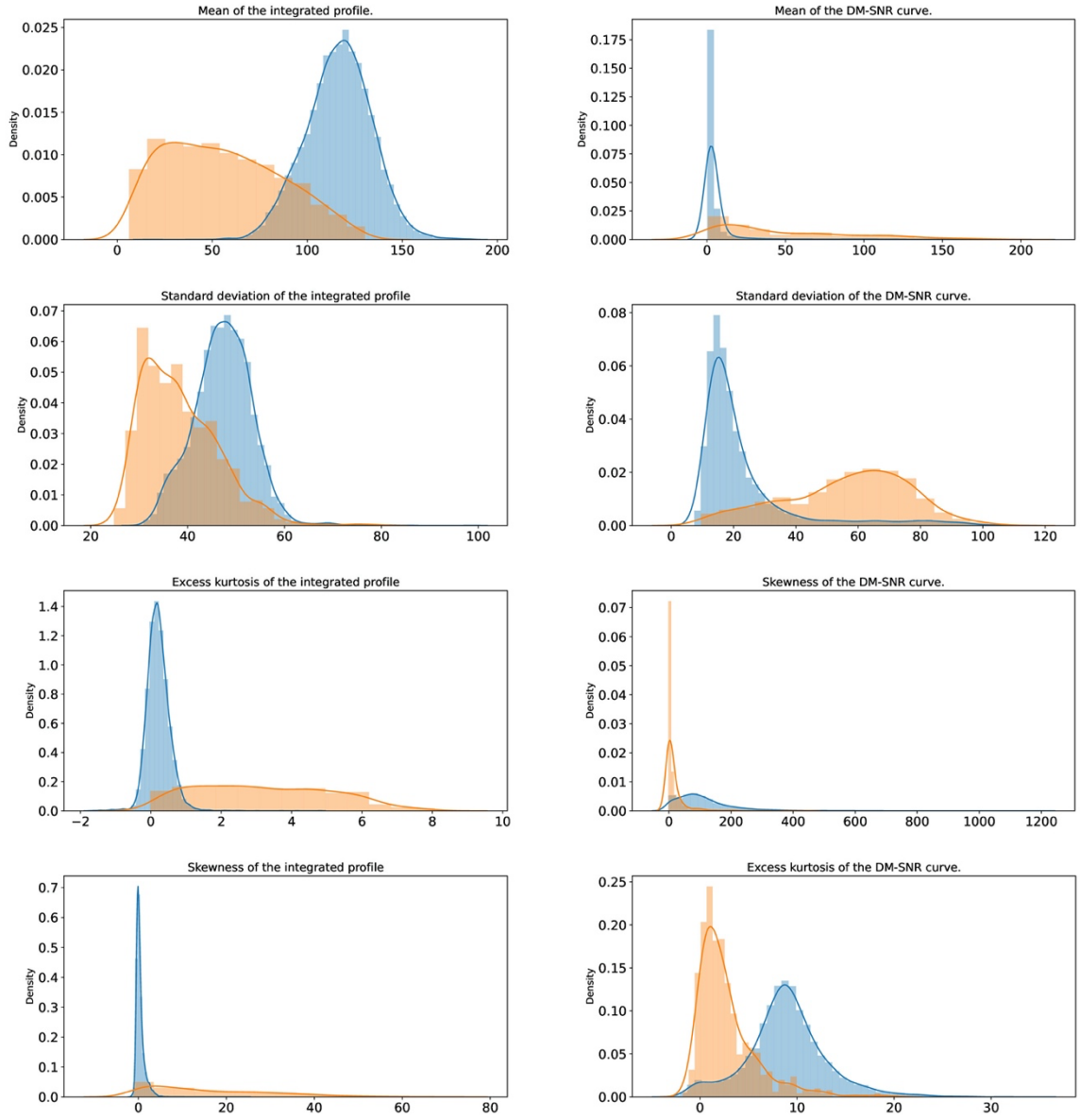
Feature and class analysis

Class distribution

By looking at the distribution of each class, we can realize that the Dataset is **highly imbalanced**. There are 8108 non-pulsars and 821 pulsars in the training set with total number of 8929 samples. The dataset might need rebalancing during the model-building process.

Feature distribution

Now we turn our attention to the histogram of features. Plots for the integrated profile are presented on the left side, and the DM-SNR curve plots are on the right. The orange color represents the non-pulsars and the blue color pulsars.



Apart from the mean and standard deviation of the integrated profile, all other features are highly right-skewed. Overall, the dataset is not normally distributed.

The SD of the integrated profile appears normal but has a tail on the right side, and only the mean of the Integrated Profile has a heavy tail on the left side and is left-skewed.

The mean of the DM-SNR curve, Standard deviation of the DM-SNR curve, and Skewness of the integrated profile seem to have more outliers than other features. This analysis suggests that Gaussianized features might produce better results. 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.

Correlation Analysis

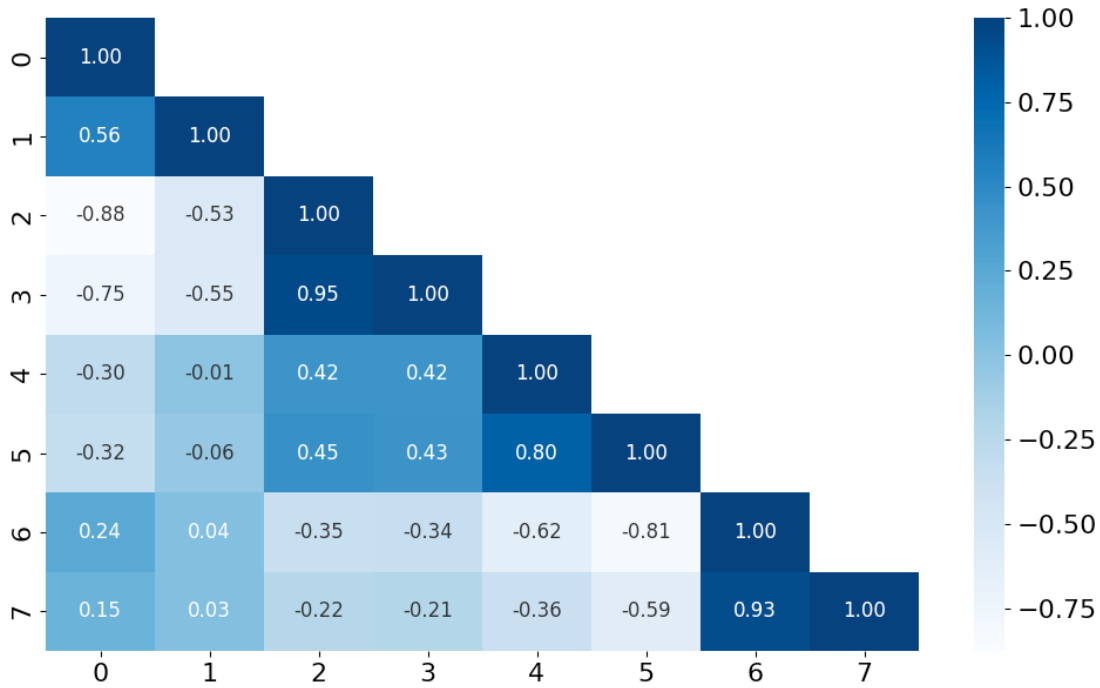
The correlation heatmap taken from the absolute value of the Pearson correlation coefficient describes positive and negative correlations between features. (Whole dataset)

Highly positively correlated:

- The skewness of the integrated profile and Excess kurtosis of the DM-SNR curve
- The skewness of the DM-SNR curve and Excess kurtosis of the DM-SNR curve
- The Mean of the DM-SNR curve and Standard Deviation of the DM-SNR curve

Highly negatively correlated:

- The Mean of the integrated profile and Excess kurtosis of the integrated profile
- The Mean of the integrated profile and Skewness of the integrated profile
- The Excess kurtosis of the DM-SNR curve and Standard Deviation of the DM-SNR curve



We can use PCA to map data to 6 or 5 uncorrelated features to reduce the amount of computation and the number of parameters to estimate. But removing 2 or 3 features, given the limited amount of data we have, might also decrease the accuracy of our models.

Classification

Applications and Cross-validation

Before analyzing different classifiers, we need to select applications and the cross-validation method for our analysis. As we are not dealing with a large dataset, re-training the models

doesn't have a considerable cost. K-fold **cross-validation** would use additional data and give us more accurate results than a single split.

Our main application will be a uniform prior one:

$$(\pi, Cfp, Cfn) = (0.5\,, 1, 1)$$

We will also consider unbalanced applications in which the prior is biased towards one of the two classes:

$$(\pi, Cfp, Cfn) = (0.1, 1, 1)\ ,\ \ (\pi, Cfp, Cfn) = (0.9, 1, 1)$$

At first, we measure performance in terms of normalized minimum detection costs. The cost we would pay if we made optimal decisions for the validation set. We will consider how to choose an optimal threshold in the second stage.

MVG Classifiers

We can expect the naive Bayes assumption to poorly fit the dataset as we have seen the degree of correlation. Tying covariances, on the other hand, might produce better results since the classifier may inaccurately estimate the covariance of pulsars because of class imbalance.

MVG Classifiers – minDCF on the validation set

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features - no PCA	
Full-Cov	**0.142**	0.281	0.690
Diag-Cov	0.192	0.314	0.726
Tied Full-Cov	0.113	**0.221**	**0.575**
Tied Diag-Cov	0.161	0.264	0.594
		Gaussianized features - no PCA	
Full-Cov	0.159	0.245	0.741
Diag-Cov	0.152	0.276	0.620
Tied Full-Cov	**0.131**	**0.238**	**0.532**
Tied Diag-Cov	0.163	0.290	0.623
		Gaussianized features - PCA (m=7)	
Full-Cov	0.155	**0.240**	0.729
Diag-Cov	0.164	0.251	0.659
Tied Full-Cov	**0.133**	0.245	**0.529**
Tied Diag-Cov	0.139	0.251	0.564
		Gaussianized features - PCA (m=6)	
Full-Cov	0.154	0.241	0.706

As the result shows, the Tied Full-Cov model performs best with raw features. Gaussianization and PCA are not effective and degrade the results in general. However,

gaussianized features with PCA perform slightly better than gaussianized features without PCA. This stops after m=6 PCA direction or doesn't improve the results further.

We can also see how gaussianization yielded significantly better results, with respect to raw data, under the naive Bayes assumptions, since the transformation seems to have decreased feature correlation by slightly whitening the covariance matrices.

Overall, the best model is currently the MVG model with Tied Full-Cov matrices with raw features.

Logistic Regression

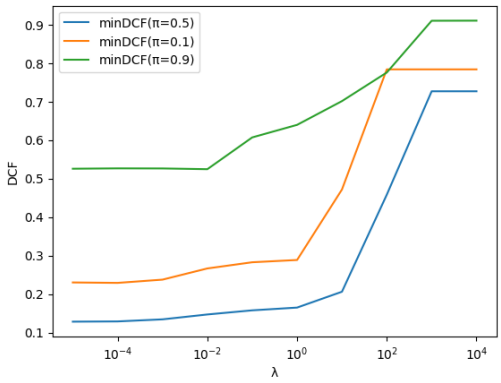
Now, we will look at discriminative approaches. Because PCA didn't positively affect our gaussian models, we only consider using the full features. We compare the results with and without Gaussianization. Given the good results obtained by tied-covariance gaussian models, we can expect other linear classifiers to work well.

At first, we analyze the regularized (linear) Logistic Regression. Since our dataset is highly imbalanced, we need to re-balance the costs of the different classes.

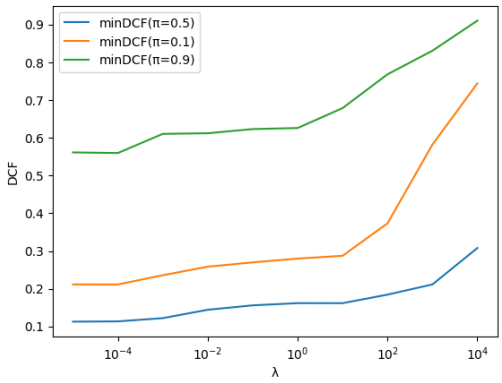
$$J(\boldsymbol{w},b)=\frac{\lambda}{2}\|\boldsymbol{w}\|^2+\frac{\pi_T}{n_T}\sum_{i=1|c_i=1}^{n}\log\left(1+e^{-z_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)}\right)+\frac{1-\pi_T}{n_F}\sum_{i=1|c_i=0}^{n}\log\left(1+e^{-z_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)}\right)$$

We need to choose the best value for λ , so we compare the model with different values of it.

Gaussianized features



Raw features



Regularization provides no benefit. The best results are obtained with small values of λ. Gaussianization seems to produce worse results when increasing the values of λ. We start training the linear logistic regression model with small values of λ and different priors π_T to see the effects on the other applications.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features	
MVG (Full-Cov)	0.142	0.281	0.690
MVG (Tied Full-Cov)	0.113	0.221	0.575
		Raw features	
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.115	0.214	0.531
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.1}$)	0.112	0.212	0.541
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.9}$)	0.121	0.216	0.518
		Gaussianized features	
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.125	0.236	0.521
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.1}$)	0.128	0.227	0.536
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.9}$)	0.127	0.240	0.517

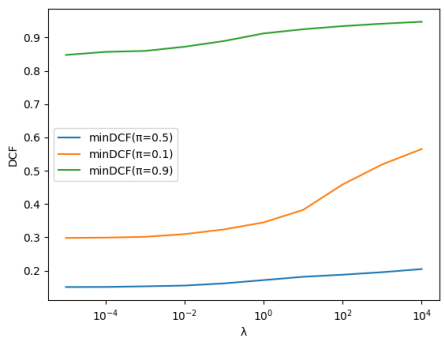
Overall, the Logistic Regression model performs slightly better than the MVG models. Using different values for π_T does not improve the Logistic Regression models for the other two applications. Gaussianization, like on MVG, yields slightly worse results than raw data. Apparently, this transformation slightly corrupts the integrity of the distribution.

Since MVG corresponds to quadratic separation rules, we repeat the analysis for Quadratic Logistic Regression, but we don't expect better results than Linear Logistic Regression.

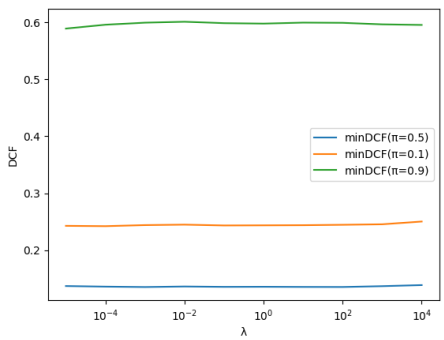
Quadratic Logistic Regression

We need to choose the best value for λ so we compare the model with different values of it.

Gaussianized features



Raw features



Results are similar as for the linear case. Regularization and Gaussianization are not helpful. Again, we consider training using a different prior π_T to see the effects on the other applications. We restrict the analysis to models with small regularization.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features	
MVG (Full-Cov)	0.142	0.281	0.690
MVG (Tied Full-Cov)	0.131	0.238	0.532
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.115	0.214	0.531
		Raw features	
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.137	0.242	0.589
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.1}$)	0.142	0.246	0.606
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.9}$)	0.138	0.262	0.549
		Gaussianized features	
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.150	0.297	0.848
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.1}$)	0.151	0.273	0.882
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.9}$)	0.155	0.339	0.853

So far, the best model is Linear Logistic Regression. Logistic Regression with a quadratic kernel didn't improve the results of MVG classifiers. This suggests that classes are linearly separable by linear decision rules.

Now, we turn our attention to SVMs and to Gaussian Mixture Models. We start with analyzing linear SVM, we expect the results to be quite good as logistic regression, if not better.

Linear SVM

For linear SVM, we need to tune the hyper-parameter C. We start with a model that does not balance the two classes. We consider the SVM formulation that the bias term is simulated through feature expansion, and is regularized.

To re-balance the classes we use a different value of C for the different classes:

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

Subject to

$$0 \leq \alpha_i \leq C_i \,, i = 1, \dots, n$$

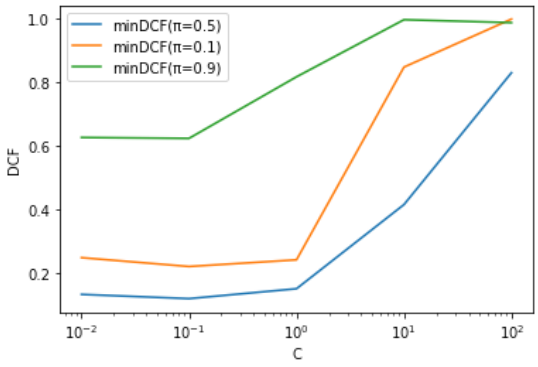
where $C_i = \; C_T$ for samples of class H_T and $C_i = C_F$ for samples of class H_F. We omitted the constraint related to the bias, since we are not explicitly modeling the bias term.

We select $C_T = \; C \; \frac{\pi_T}{\pi_T^{emp}}$ and $C_F = \; C \; \frac{\pi_F}{\pi_F^{emp}}$

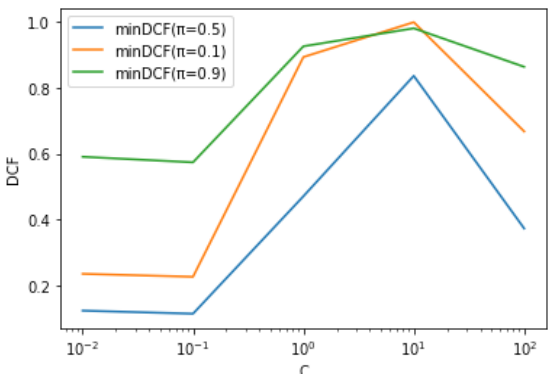
π_T^{emp} and π_F^{emp} are the empirical priors (i.e. sample proportions) for the two classes computed over the training set.

We only considered Raw features.

Without class re-balancing



Class re-balanced



Choice of C seems to be important specially for high values of C the model does not perform good at all. $C = 0.1$ provides the best results for the two models.

Now, we can compare linear models in terms of min DCF:

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features	
MVG (Tied Full-Cov)	0.131	0.238	0.532
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.115	0.214	0.531
Linear SVM ($\boldsymbol{C = 0.1}$)	**0.121**	0.222	0.625
Linear SVM ($\boldsymbol{C = 0.1,\ \pi_T = 0.5}$)	**0.115**	0.227	0.594

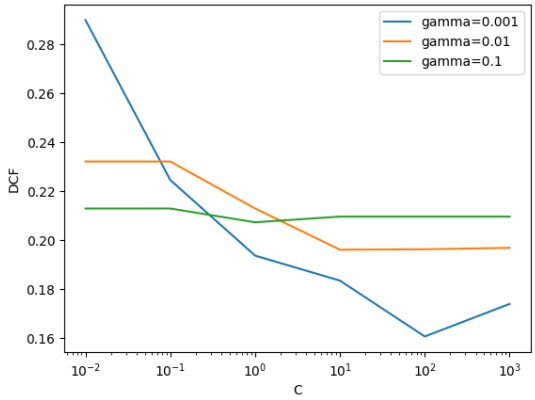
Linear SVM performs similarly to other linear approaches, as expected. Class re-balancing seems to have minimal improvements for our model. As we expected, the Linear SVM for the main application performs like Logistic Regression, but for other applications, Logistic Regression results are better.

Linear models perform better on this dataset, but we will consider a non-linear SVM formulation for the sake of analysis. We will use a Radial Basis Function kernel and skip the analysis of Quadratic SVM.

For the RBF kernel, we need to estimate the kernel parameter γ. We will use a grid search to jointly optimize C and γ. We will mainly focus on Raw features.

Kernel SVM

For kernel SVM, we need to estimate the hyper-parameter C and γ. For our main application ($\pi = 0.5$) *we* have the graph below for different values of C and γ.



The plot shows that both γ and C influence the results, and both should be optimized. Best results are obtained using γ = 0.001 and C = 100.

	$\boldsymbol{\pi = 0.5}$	$\boldsymbol{\pi = 0.1}$	$\boldsymbol{\pi = 0.9}$
		Raw features	
MVG (Tied Full-Cov)	0.131	0.238	0.532
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.115	0.214	0.531
Linear SVM ($\boldsymbol{C = 0.1}$)	0.121	0.222	0.625
RBF SVM ($\boldsymbol{C = 100,\ \gamma = 0.001}$)	0.160	0.291	0.777
RBF SVM ($\boldsymbol{C = 100,\ \gamma = 0.001,\ \pi_T = 0.5}$)	0.182	0.366	0.710

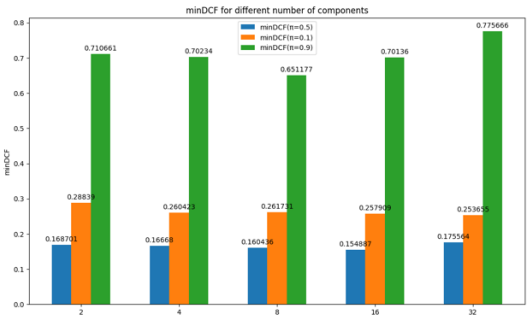
RBF kernel SVM provides worse results than Logistic Regression and MVG. Class re-balancing does not provide better results.

Gaussian Mixture Model (GMM)

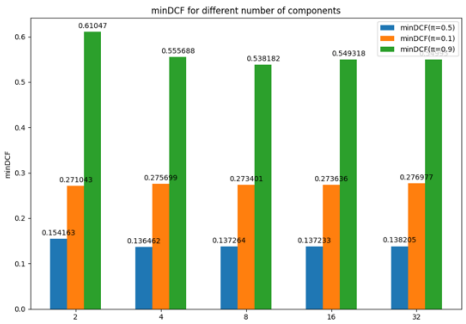
Our last model is a generative approach based on training a GMM over the data of each class. GMMs can approximate generic distributions, so we expect to obtain better results than with the Gaussian model. We consider both full covariance and diagonal, with and without covariance tying.

We use the K-fold protocol to select the number of Gaussians and to compare different models. We consider raw and gaussianized features.

Raw features



Gaussianized features



We faced very expensive computations, considerably more than SVM, so we needed to keep the number of components to a reasonably small value, by trying both naive and tied models. We had better results with 4, 16, and 32 GMM components.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features	
Full Cov, 16 Gau	0.154	0.257	0.701
Diag Cov, 16 Gau	0.158	0.249	0.785
Tied Full Cov, 32 Gau	0.148	0.255	0.699
Tied Diag Cov, 32 Gau	**0.145**	0.241	0.690
		Gaussianized features	
Full Cov, 4 Gau	0.136	0.275	0.555
Diag Cov, 16 Gau	**0.136**	0.275	0.543
Tied Full Cov, 32 Gau	0.143	0.284	0.544
Tied Diag Cov, 32 Gau	0.153	0.273	0.536

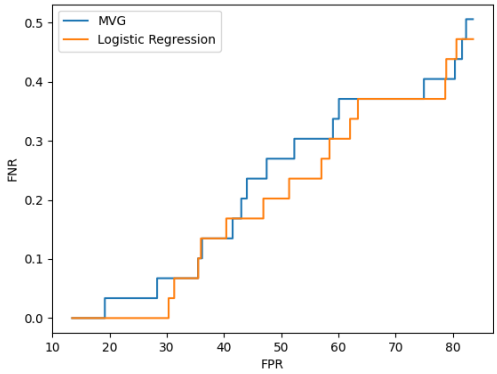
In the next section, we select two best models so far for further analysis.

Model Selection

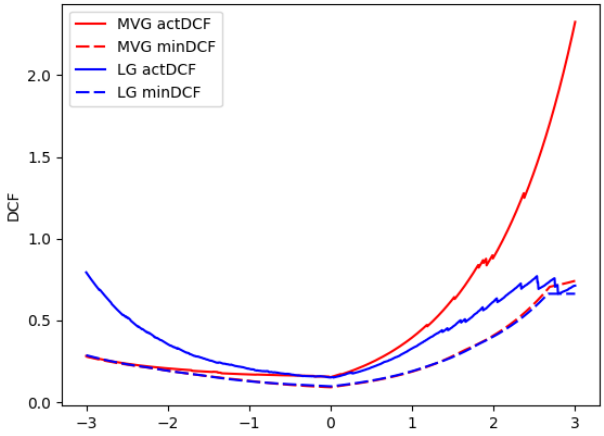
	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features - no PCA	
MVG (Tied Full-Cov)	0.113	0.221	0.575
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.115	0.214	0.531
Linear SVM ($\boldsymbol{C = 0.1,\ \pi_T = 0.5}$)	0.115	0.227	0.594

These are the models with the best performance on this dataset. We select the Logistic Regression model trained with balanced classes and the MVG with tied covariances as our main models for now, and we use a DET plot to compare these models.

The DET plot for false positive rate and false negative rate shows that the two models are almost identical for different thresholds, with Logistic Regression performing slightly better.



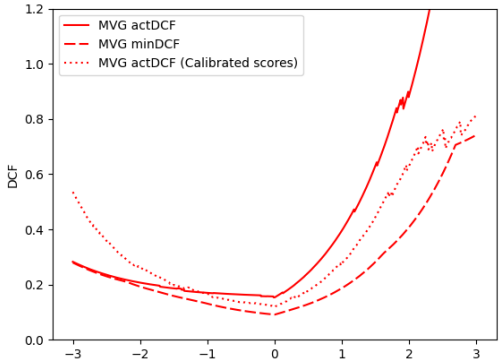
So far, we have used only minimum DCF metrics. Min DCF measures the cost we would pay if we made optimal decisions for the evaluation set using the recognizer scores. The cost we pay, however, depends on the goodness of the decisions we make using those scores. Hence, we turn our attention to actual DCFs. We use a Bayes error plot, which shows the DCFs for different applications.



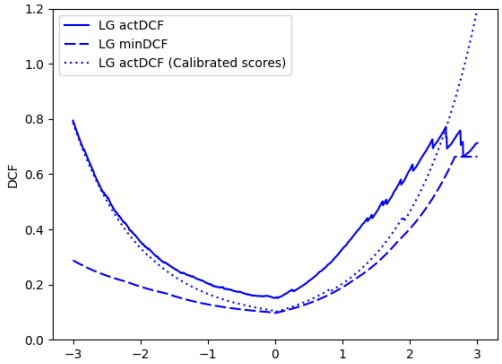
We can evaluate the actual DCF to assess how good the models would be if we used the theoretical threshold for each application. We can observe that the Logistic Regression provides scores that are almost calibrated. On the other hand, the calibration of MVG scores is quite poor. We can re-calibrate the scores, so the theoretical threshold provides close to optimal values. We use logistic regression to estimate the function that transforms our scores into well-calibrated scores.

We select the models trained with target application $\pi = 0.5$. We can check the Bayes error plots for the calibrated scores,

MVG with Calibration



LG with Calibration



As these graphs suggest, score calibration in Logistic Regression provides results that are closer to the minimum costs for a wide range of applications, but it seems MVG scores require re-estimation of a different threshold for different applications, as can be seen from the Bayes error plots.

We select the Logistic Regression model trained with balanced classes and the MVG with tied covariances as our final models.

Experimental Validation

We now need to assess the quality of our final models on held-out data (the evaluation set) We will verify the performance and analyze (some) of the choices we made to see how they affected performance for unseen data.

We start again from the Gaussian classifiers, and we evaluate systems in terms of minimum DCFs. The minimum DCF provides an optimistic estimate of the actual DCF. It's the cost we would have if we could select the optimal threshold for the evaluation set.

Below are the results for the multivariate gaussian classifier. We can see that the results are consistent with those obtained on the validation set. The Tied Full-Cov model performs best with raw features. Gaussianization and PCA are not effective and degrade the results in general.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features - no PCA	
Full-Cov	0.140	0.282	0.645
Diag-Cov	0.184	0.329	0.620
Tied Full-Cov	0.109	0.207	0.590
Tied Diag-Cov	0.151	0.262	0.543
		Gaussianized features - no PCA	
Full-Cov	0.149	0.239	0.620
Diag-Cov	0.152	0.286	0.574
Tied Full-Cov	**0.125**	**0.220**	**0.536**
Tied Diag-Cov	0.157	0.299	0.569
		Gaussianized features - PCA (m=7)	
Full-Cov	0.149	0.244	0.641
Diag-Cov	0.154	0.246	0.605
Tied Full-Cov	**0.125**	**0.228**	**0.524**
Tied Diag-Cov	0.128	0.239	0.522

Now, we check linear logistic regression models with the estimated $\boldsymbol{\lambda = 10^{-5}}$ value.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features	
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.107	0.198	0.541
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.1}$)	0.109	0.198	0.531
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.9}$)	0.115	0.202	0.509
		Gaussianized features	
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.121	0.215	0.487
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.1}$)	0.125	0.209	0.499
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.9}$)	0.122	0.213	0.479

Results are consistent with our expectations. the Logistic Regression model performs slightly better than the MVG models. Using different values for π_T does not improve the Logistic Regression models for the other two applications and Gaussianization does not provide better results.

We repeat the analysis for quadratic logistic regression, with the estimated $\boldsymbol{\lambda = 10^{-5}}$ value.

	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Raw features	
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	**0.137**	0.238	0.565
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.1}$)	0.140	0.246	0.600
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.9}$)	0.127	0.230	0.547
		Gaussianized features	
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	**0.151**	0.287	0.851
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.1}$)	0.149	0.273	0.872
QLog Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.9}$)	0.150	0.339	0.849

Like before, the results are consistent with our expectations. Quadratic Logistic Regression didn't improve the results of MVG and Linear logistic Regression classifiers.

Now, we consider linear and RBF kernel SVM using estimated values for C and $\boldsymbol{\gamma}$ as written in the tables below.

	$\boldsymbol{\pi = 0.5}$	$\boldsymbol{\pi = 0.1}$	$\boldsymbol{\pi = 0.9}$
		Raw features	
Linear SVM ($\boldsymbol{C = 0.1}$)	0.121	0.228	0.621
RBF SVM ($\boldsymbol{C = 100,\ \gamma = 0.001}$)	0.161	0.281	0.721

Again, the results are in line with what we expected. Linear SVM model provides better results than SVM with RBF kernel. Linear SVM for main application performs like Logistic Regression, but for other applications Logistic Regression results are better. We skip the analysis for GMM model as the model was not performing very well on training dataset.

Finally, we analyze the final system on the evaluation set in terms of actual DCF. As described before, actual DCFs are computed using the actual decisions we make for evaluation data.

Class predictions require that we compare scores with a threshold. If scores are calibrated, then we have shown that the optimal threshold depends only on the cost of errors and the class priors through $t = -log\frac{\pi}{1-\pi}$ where π is the effective prior for the H_T class.

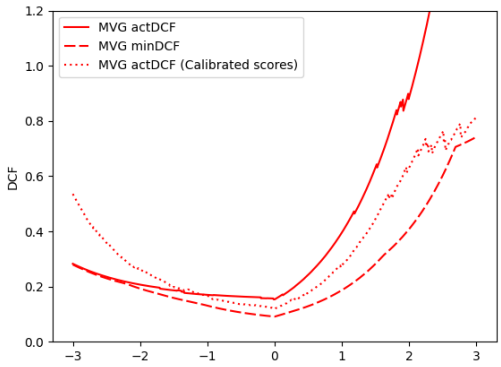
If scores are not calibrated, we can re-calibrate the scores, so the theoretical threshold provides close to optimal values.

For score calibration, we select the models trained with target application $\pi = 0.5$.

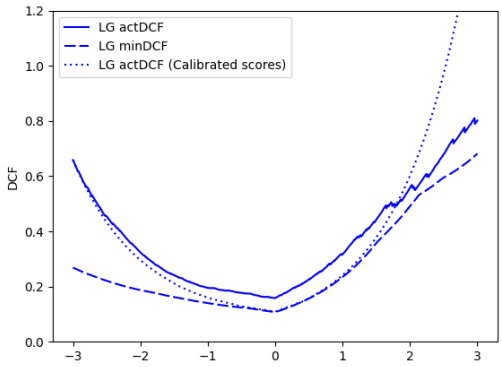
	$\pi = 0.5$	$\pi = 0.1$	$\pi = 0.9$
		Min DCF	
MVG Tied Full-Cov	0.109	0.207	0.590
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.107	0.198	0.541
		Actual DCF	
MVG (Tied Full-Cov)	0.168	0.638	0.536
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.157	0.371	0.592
		Actual DCF with calibrated scores	
MVG (Tied Full-Cov)	0.139	0.559	0.807
Log Reg ($\boldsymbol{\lambda = 10^{-5}, \pi_T = 0.5}$)	0.115	0.295	0.808

Score calibration, however, seems to require re-estimation of a different threshold for $\pi = 0.9$ as scores are not well-calibrated over a wide range of applications, as can be seen from the Bayes error plots:

MVG with Calibration



LG with Calibration



Conclusion

In conclusion, we can achieve a DCF cost of ≈ 0.1 for the primary application $\boldsymbol{\pi_T = 0.5}$ with Logistic Regression model trained with balanced classes.

However, the model is less effective for applications with imbalanced costs and prior proportions.

Overall, the similarity between validation and evaluation results suggests that the evaluation population is sufficiently like the training population. The choices we made on our training/validation sets proved effective also for the evaluation data.