Machine Learning
and
Pattern Recognition

Wine Quality Detection

Krzysztof Kleist

Table of contents

Introduction

Two datasets were created, using red and white wine samples. 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).The two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine. For the project purpose the wines were divided into two groups: good – class label 1., bad – class label 0. Wines with grade below 6. were considered as bad, above 6 – good. Wines with grade equal to 6 were removed from the dataset.

The original task requires determining the optimal classification model taught throughout the semester.

We keep the original train/evaluation split.

Wine Quality Dataset Features

Train set contains 1839 samples of wine. Test set contains 1822 samples of wine. Each wine sample in the dataset 11 attributes regarding:

1. fixed acidity
2. volatile acidity
3. citric acid
4. residual sugar
5. chlorides
6. free sulphur dioxide
7. total sulphur dioxide
8. density
9. pH
10. sulphates
11. alcohol

Features are all continuous in all samples of both datasets (train and test set).

The classes are ordered and not balanced (e.g. there are many more normal wines than excellent or poor ones). Outlier detection algorithms could be used to detect the few excellent or poor wines but this is out of the scope of our project.

Building a Classifier for the Dataset

Below there are visualised raw data features for initial observation of feature distribution. The features are lined according to their order.

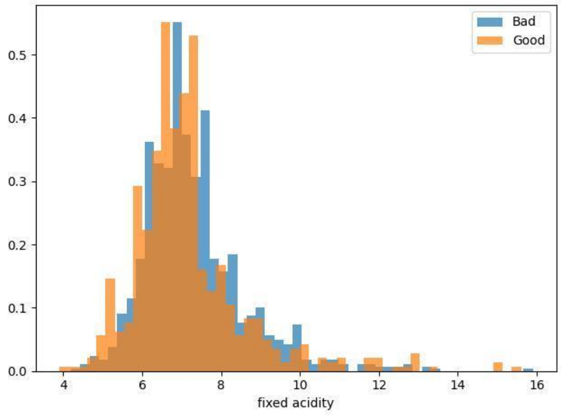

Figure 1. fixed acidity

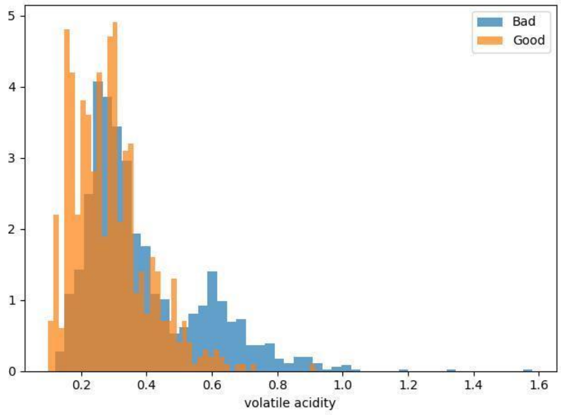

Figure 2. volatile acidity

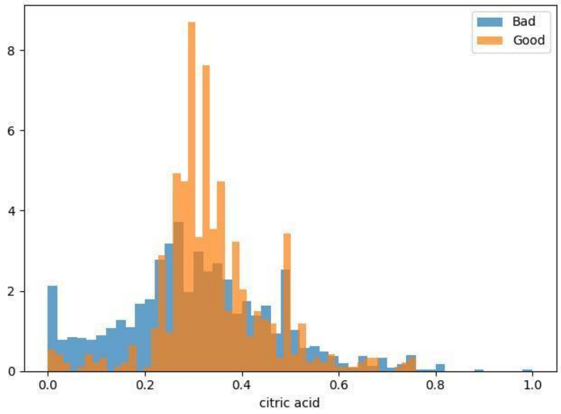

Figure 3. citric acid

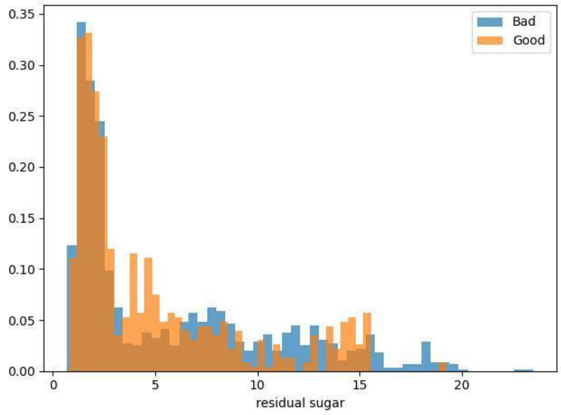

Figure 4. residual sugar

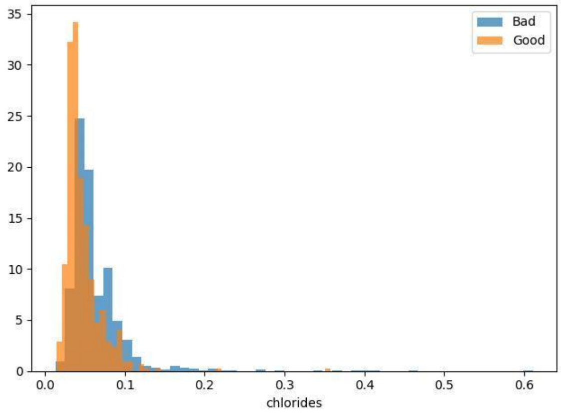

Figure 5. chlorides

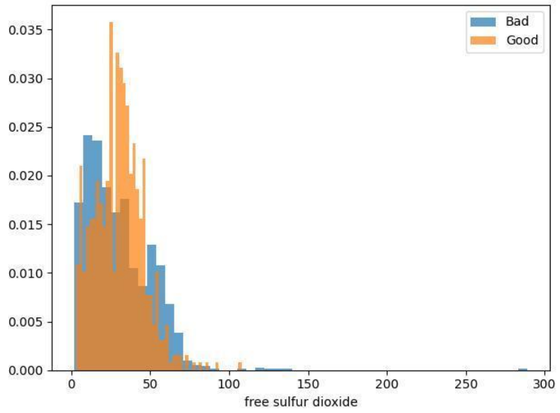

Figure 6. free sulphur dioxide

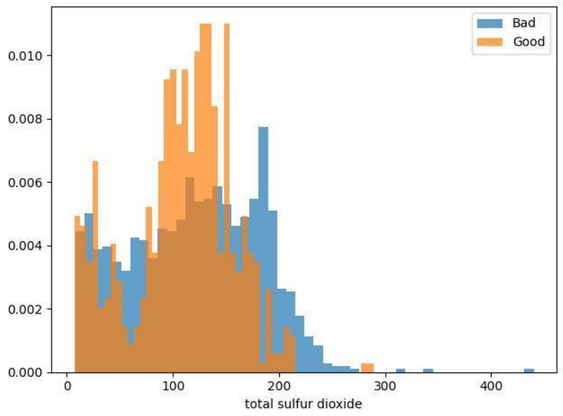

Figure 7. total sulphur dioxide

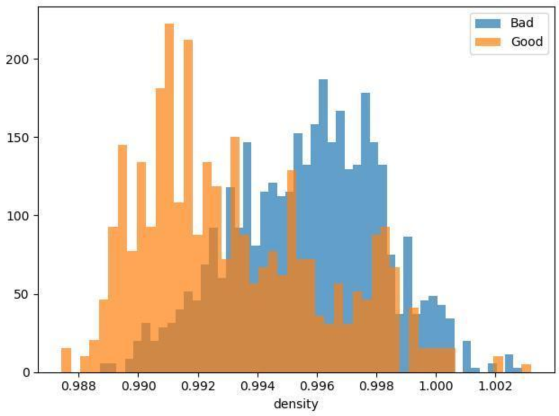

Figure 8. density

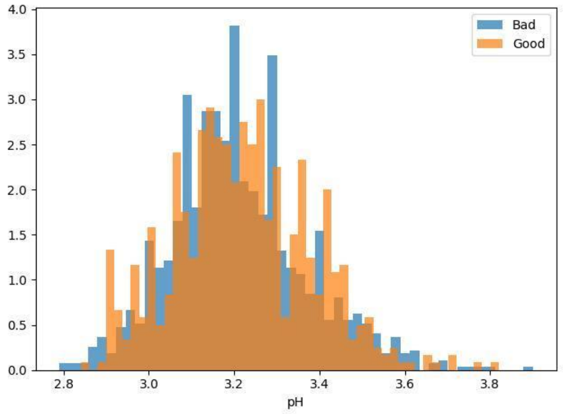

Figure 9. pH

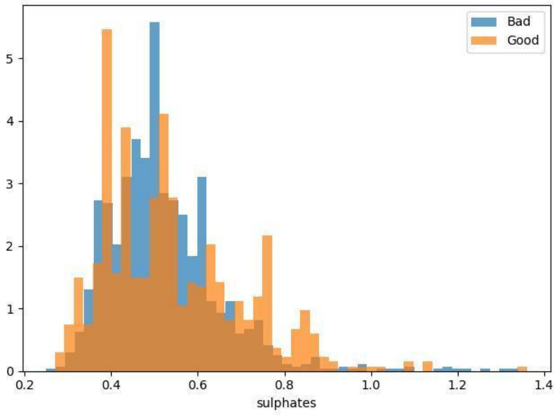

Figure 10. sulphates

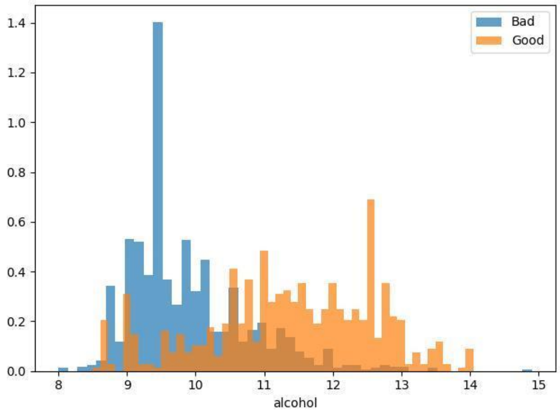

Figure 11. alcohol

Due to the presence of outliers, we expect classification approaches may produce suboptimal results (especially gaussian based methods). Looking at the features distribution in features such as fixed acidity, residual sugar, free sulphur dioxide and pH we can see that data has very low variance and data doesn't allow for discriminative observations.

We therefore further pre-process data by "Gaussianizing" the features. Depending on the classifier used and error rates received Dimensionality reduction methods such as Principle Component Analysis or Linear Discriminant Analysis has been used. Depending on the case they might be used together or not used at all.

Gaussianization is a procedure that allows mapping a set of features to values whose empirical cumulative distribution function is well approximated by a Gaussian c.d.f.

The processing consists in mapping the features to a uniform distribution and then transforming the mapped features through the inverse of Gaussian cumulative distribution function.

Let x be the feature we want to transform:

$$r(x) = \frac{\sum_{i=1}^{N} \mathbb{I}\,[x_i < x] + 1}{N + 2},$$

x_i is the value of the considered feature for the i-th training sample. We add 1 to the numerator and 2 to the denominator to avoid numerical issues in the next stage (i.e. we assume the existence of a feature smaller than all the others and a feature larger than all the others).

We then compute the transformed feature as:

$$y = \Phi^{-1}\big(r(x)\big),$$

where ϕ is the inverse of the cumulative distribution function (percent point function, p.p.f) of the standard normal distribution. Gaussianized feature plots are presented below.

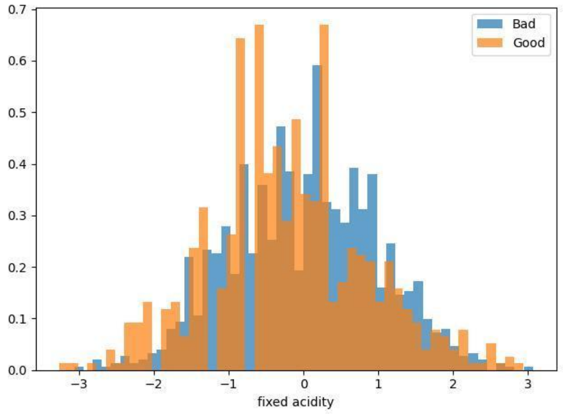

Figure 12. fixed acidity

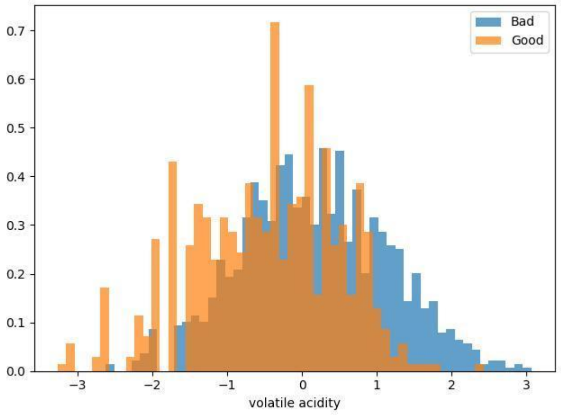

Figure 13. volatile acidity

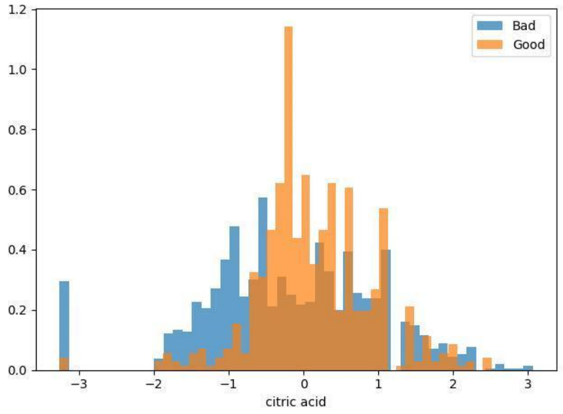

Figure 14. citric acid

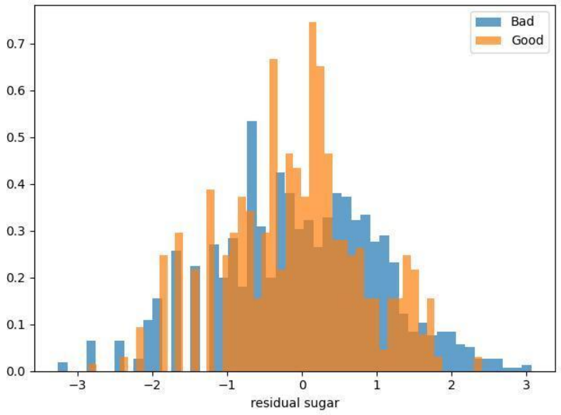

Figure 15. residual sugar

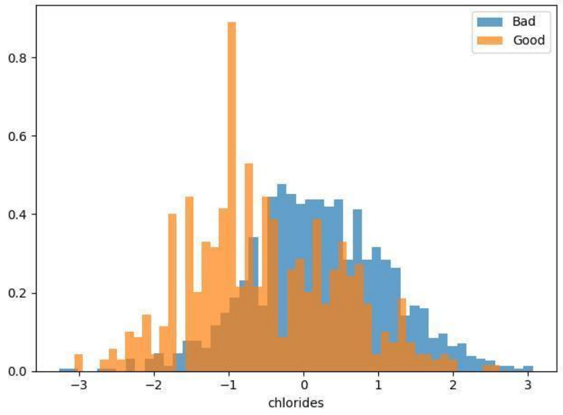

Figure 16. chlorides

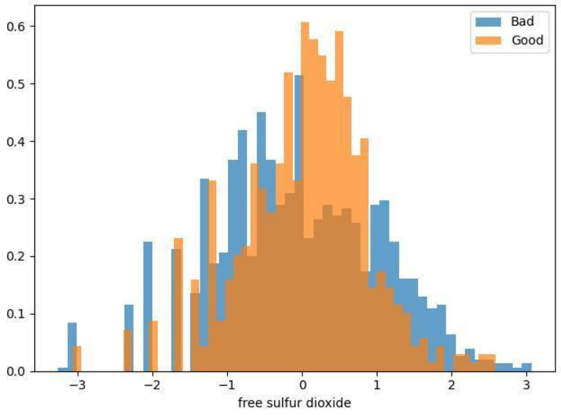

Figure 17. free sulphur dioxide

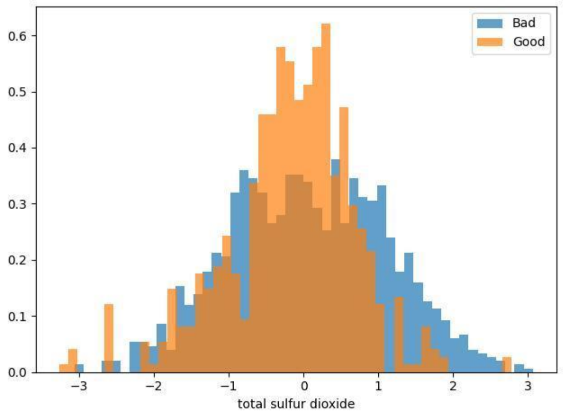

Figure 18. total sulphur dioxide

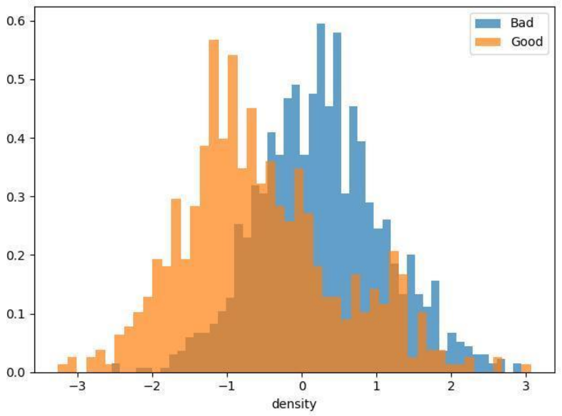

Figure 19. density

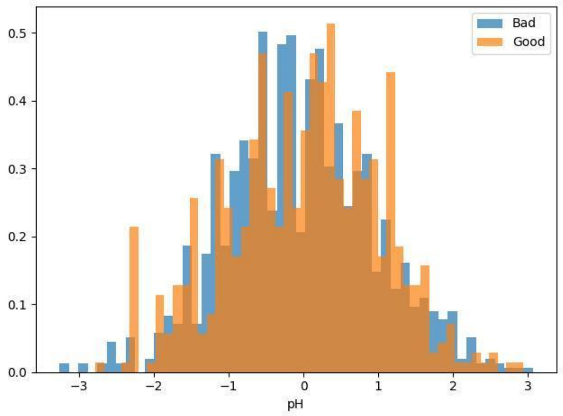

Figure 20. pH

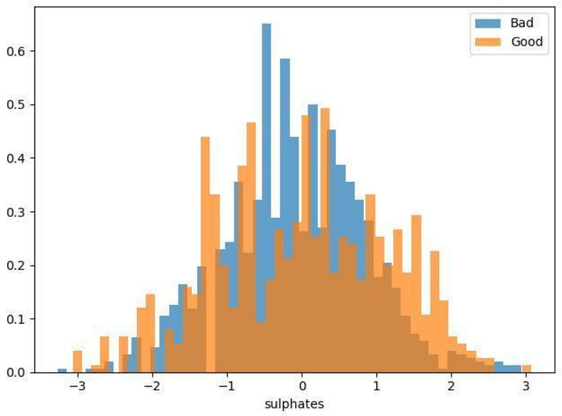

Figure 21. sulphates

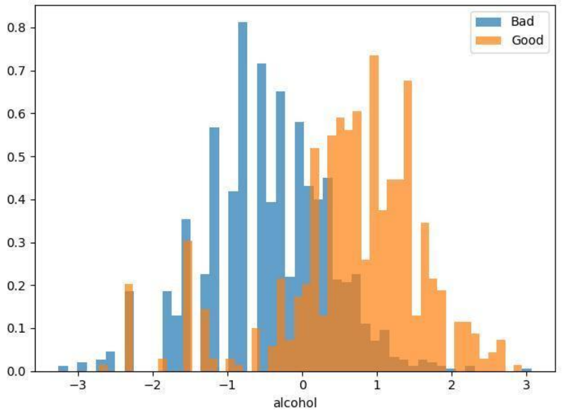

Figure 22. alcohol

Dimensionality Reduction

Dimensionality reduction are techniques for reducing the number of input variables in training data. When dealing with high dimensional data, it is often useful to reduce the dimensionality by projecting the data to a lower dimensional subspace which captures the "essence" of the data.

Principal component analysis (PCA) is an unsupervised dimensionality reduction technique which simplifies the complexity in high-dimensional data while retaining trends and patterns. It does this by transforming the data into fewer dimensions, which act as summaries of features.

For dataset mean $\bar{x}$ PCA subspace is computed from eigenvectors of the empirical covariance matrix:

$$\frac{1}{K}\sum_{i}(x_i - \bar{x})(x_i - \bar{x})^T$$

To keep only the m directions with highest variance we keep only the first "m" transformed directions.

While trying different classification methods in the upcoming parts we have done cross validations with validation sets to get the optimal value of m.

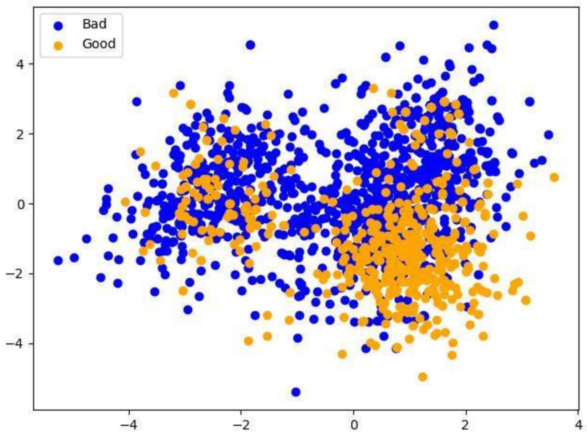

Figure 23. PCA for m=2

Classification of the Dataset

We start classification with Gaussian classifiers. We are going to implement all Gaussian classifiers shown in the labs and lectures. To understand which model is most promising, and to assess the effects of using PCA, we can adopt two methodologies, we can split the training dataset into development (for model training) and validation subsets (single-fold in the following) or we can employ K-Fold cross-validation.

For this project we will employ K-Fold cross-validation as it has major advantages compared to single-fold. Single-fold approach is definitely faster but has less data for validation and model training and in the case of our dataset it might be insufficient. When it comes to K-Fold cross-validation more data is available for training and validation and also the training is done on the whole dataset. In this project we will k fold with k=5.Since We tried many values with cross validation after a number it doesn't make great differences in the evaluation.

We are interested in choosing the best possible approach. Therefore we measure performance in terms of minimum detection costs (minDCF). Min DCF measures the cost we would pay if we made optimal decisions for the test set using the recognizer scores.

For the MVG classifiers we acquire results shown in the table below (as said before the data is split using K-Fold cross-validation with K=5).

	Raw features				Gaussianized features		
	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$		$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
				No PCA			
Full-Cov	0.412	0.831	0.922		0.419	0.842	0.971
Tied Full-Cov	**0.348**	0.853	0.778		**0.392**	0.825	0.931
Diag-Cov	0.454	0.852	0.922		0.494	0.862	0.985
Tied Diag-Cov	0.416	0.881	0.977		0.477	0.869	0.981
				PCA (m = 8)			
Full-Cov	**0.361**	0.835	0.922		0.416	0.872	0.910
Tied Full-Cov	0.378	0.849	0.868		**0.414**	0.864	0.912
Diag-Cov	0.408	0.925	0.937		0.508	0.943	0.987
Tied Diag-Cov	0.400	0.856	0.894		0.427	0.881	0.923
				PCA (m = 9)			
Full-Cov	0.358	0.847	0.908		0.414	0.846	0.914
Tied Full-Cov	**0.348**	0.851	0.764		**0.395**	0.821	0.854
Diag-Cov	0.414	0.912	0.914		0.502	0.901	0.986
Tied Diag-Cov	0.373	0.874	0.790		0.403	0.847	0.869

The Tied Full-Cov model obtains best results with or without PCA or gaussianization. These results are for K-fold protocol k=5. We don’t use a single fold approach in our project.

PCA is not effective for Tied Full-Cov models. Only on raw features with m=8 it is effective for Full-Cov but considering all other cases we can't say PCA is effective for Full-Cov models. It even decreases performance in many cases. For Diag-Cov models on the other hand despite not being able to perform better than the Tied Full-Cov with PCA m=9 makes a good effect on the Diag-Cov models(but further dimensionality reduction with PCA decreases accuracy). Diagonal-Cov performs in general worse.

Even though it provides performance improvement, Gaussianization doesn’t provide significant improvement over raw features.

When we try evaluation with different K values the results don’t change remarkably, suggesting that the amount of data is enough for validation and model training (this may differ for other classification models).

Overall, the best candidate is currently MVG with Tied Full-Cov without PCA and with PCA m=9 even though they both give similar results in these cases we can't say they will perform the same all the time.

Bayes Error Plots and ROC curves for the best results of MVG classifiers of different types (highlighted in the table) are shown below.

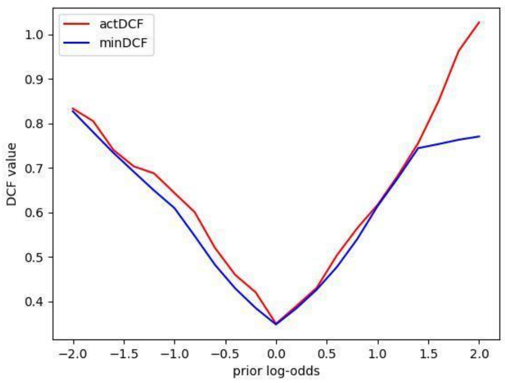

Figure 24. Tied Full-Cov, raw data, no PCA , π̃=0.5

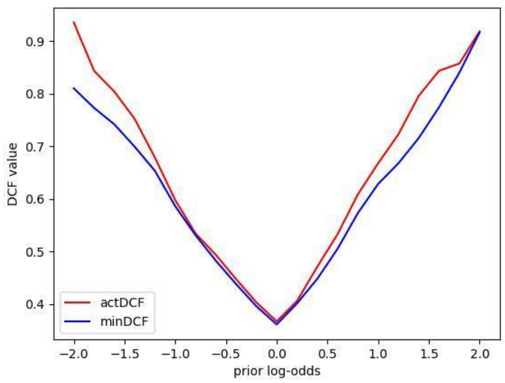

Figure 25. Tied Full-Cov, gaussianized, no PCA , π̃=0.5

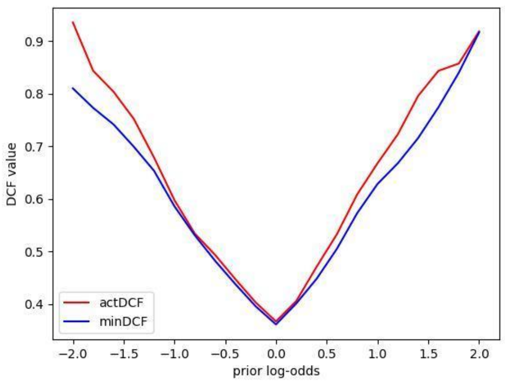

Figure 26. Full-Cov, raw data, PCA m=8, π̃=0.5

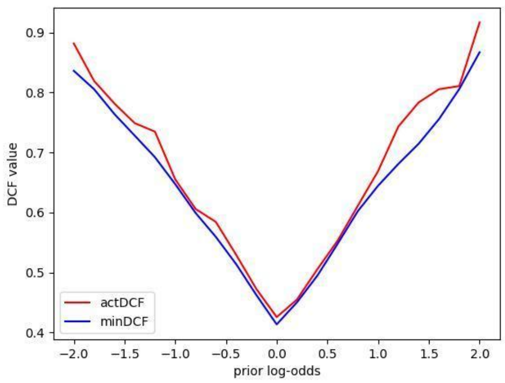

Figure 27. Tied Full-Cov, gaussianized, PCA m=8, π̃=0.5

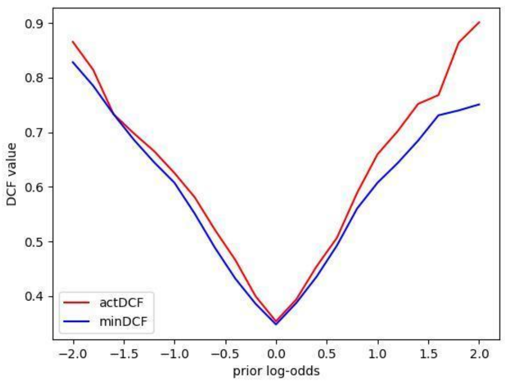

Figure 28. Tied Full-Cov, raw data, PCA m=9, π̃=0.5

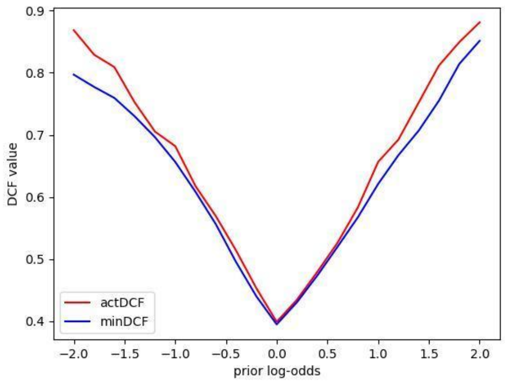

Figure 29. Tied Full-Cov, gaussianized, PCA m=9, π̃=0.5

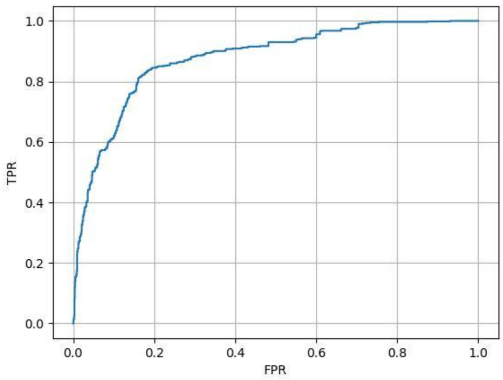

Figure 30. Tied Full-Cov, raw data, no PCA , π̃=0.5

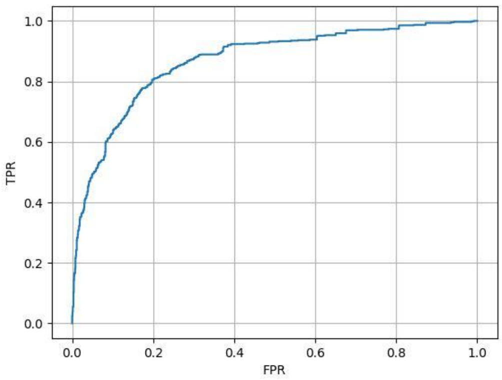

Figure 31. Tied Full-Cov, gaussianized, no PCA , π̃=0.5

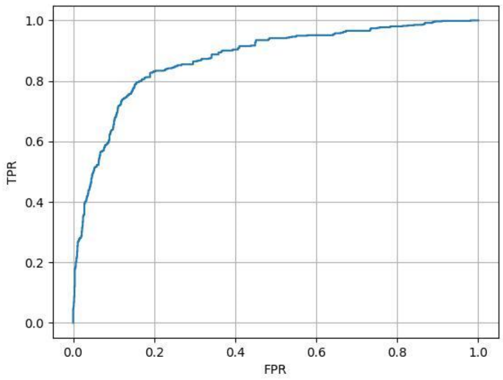

Figure 32. Full-Cov, raw data, PCA m=8, π̃=0.5

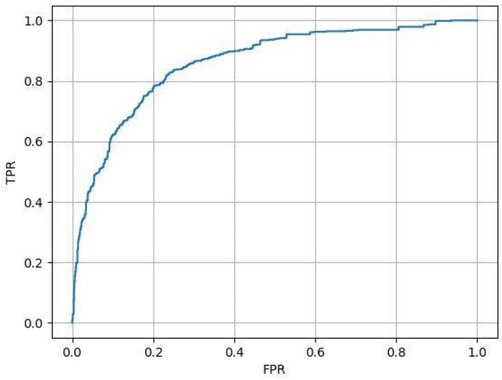

Figure 33. Tied Full-Cov, gaussianized, PCA m=8, π̃=0.5

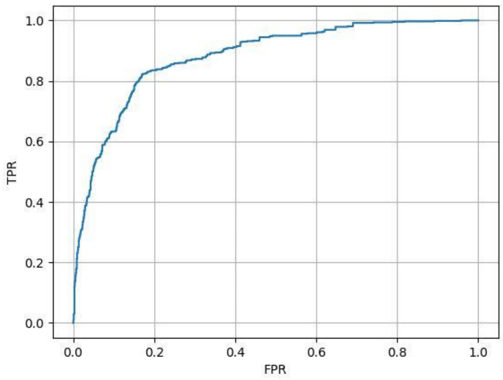

Figure 34. Tied Full-Cov, raw data, PCA m=9, π̃=0.5

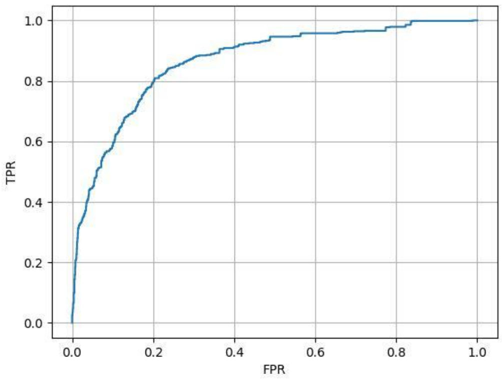

Figure 35. Tied Full-Cov, gaussianized, PCA m=9, π̃=0.5

We continue our approaches with discriminative models. Given the limited effectiveness of PCA or even some cases having negative effects and decreasing performance we only consider using a whole set of features. In this case we compare results with and without gaussianization.

We continue with regularised (linear) Logistic Regression. Since classes are not balanced. We re-balance the costs of different classes minimising:

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

We can compare the models by using different values of $\boldsymbol{\lambda}$ and π.The values are plotted below.

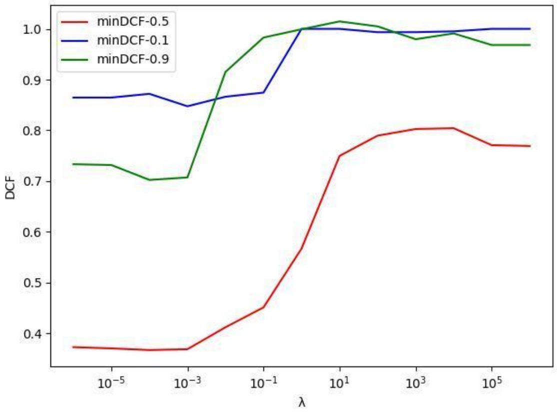

Figure 36. Raw data

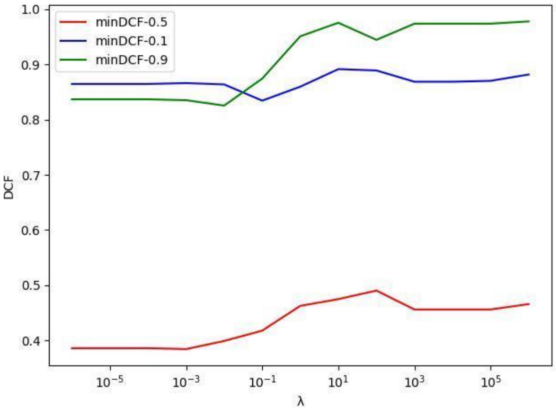

Figure 37. Gaussianized data

Our primary metric is the redline. Regularisation provides little to no benefit. Best results are obtained with small values of $\boldsymbol{\lambda}$. Gaussianization provides good performance especially when the $\boldsymbol{\lambda}$ values are high. We can also consider training using a different prior π_t to see the effects on other applications. We restrict the analysis to $\boldsymbol{\lambda}$=1e-4, and consider only K-fold protocol with k=5.

	Raw features				Gaussianized features		
	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$		$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
No PCA							
Log Reg (λ = 1e-4, π_T = 0.5)	0.367	0.872	0.702		**0.386**	0.865	0.837
(λ = 1e-4, π_T = 0.1)	**0.358**	0.852	0.747		0.387	0.812	0.946
(λ = 1e-4, π_T = 0.9)	0.374	0.870	0.679		0.409	0.927	0.741
PCA (m = 8)							
Log Reg (λ = 1e-4, π_T = 0.5)	0.381	0.885	0.746		0.418	0.897	0.807
(λ = 1e-4, π_T = 0.1)	**0.379**	0.847	0.834		**0.411**	0.855	0.949
(λ = 1e-4, π_T = 0.9)	0.400	0.894	0.680		0.421	0.931	0.737
PCA (m = 9)							
(λ = 1e-4, π_T = 0.5)	0.366	0.863	0.707		**0.387**	0.861	0.761
(λ = 1e-4, π_T = 0.1)	**0.354**	0.849	0.742		**0.387**	0.825	0.923
(λ = 1e-4, π_T = 0.9)	0.374	0.878	0.666		0.407	0.911	0.734

Within LR models Log Reg (λ = 1e-4, πT = 0.1) with Raw features and PCA m=9 performs the best. Gaussianization doesn’t provide any performance improvement in many cases, it has the opposite effect and decreases the performance. PCA m=8 has negative effects in all of the options, this tells us it does not provide enough reduction for functional classification. PCA m=9 on the other hand even though not great but it provides slight improvement in raw features and provides very slight deterioration in gaussianized features.

Overall until this point The Tied Full-Cov model obtains best results with or without PCA.

Logistic regression provides very small improvement over the MVG model with linear classification rules.

Using different values for π_T does not improve the Logistic regression models for other applications.

Bayes Error Plots and ROC curves for the best results of Logistic Regression classifiers of different types (highlighted in the table) are shown below.

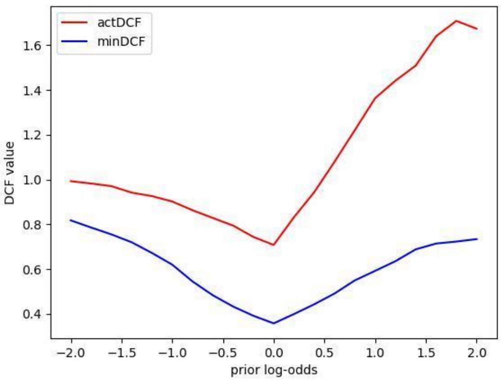

Figure 38. raw data, no PCA , π̃=0.5

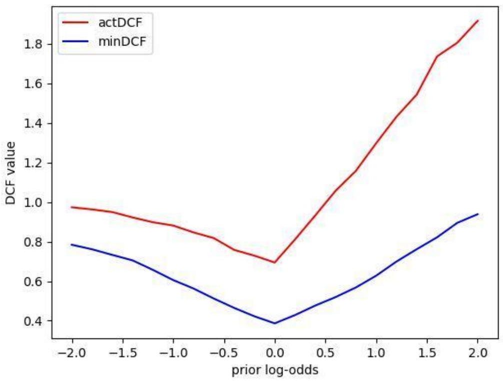

Figure 39. gaussianized, no PCA , π̃=0.5

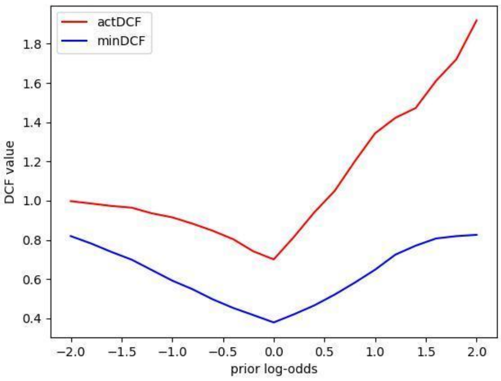

Figure 40. raw data, PCA m=8, π̃=0.5

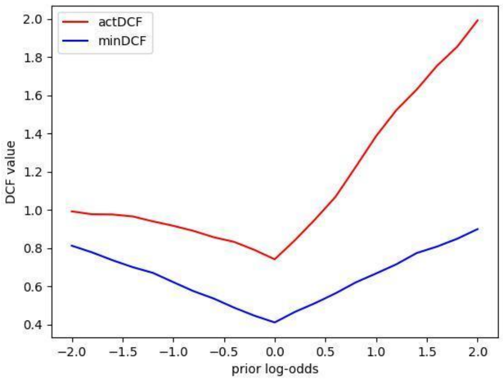

Figure 41. gaussianized, PCA m=8, π̃=0.5

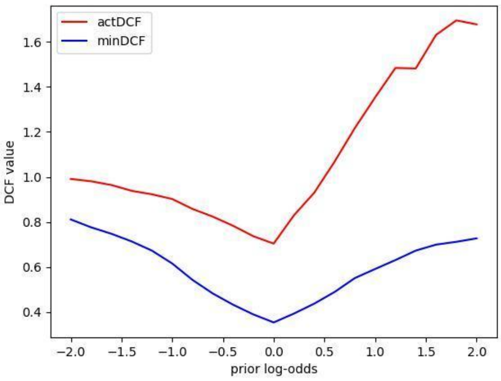

Figure 42. raw data, PCA m=9, π̃=0.5

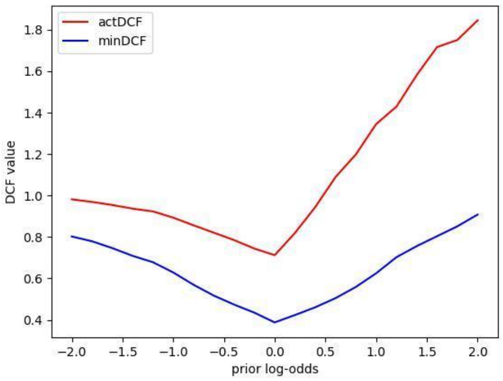

Figure 43. gaussianized, PCA m=9, π̃=0.5

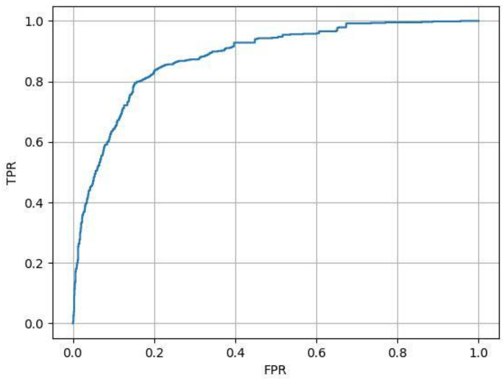

Figure 44. raw data, no PCA , π˜=0.5

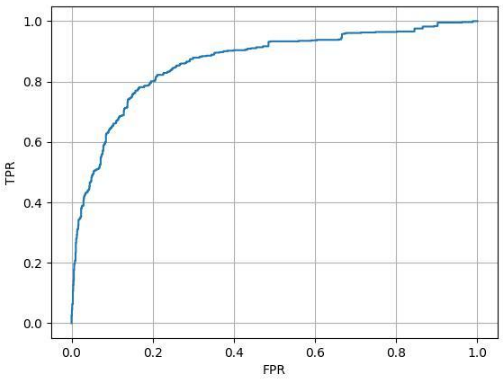

Figure 45. gaussianized, no PCA , π˜=0.5

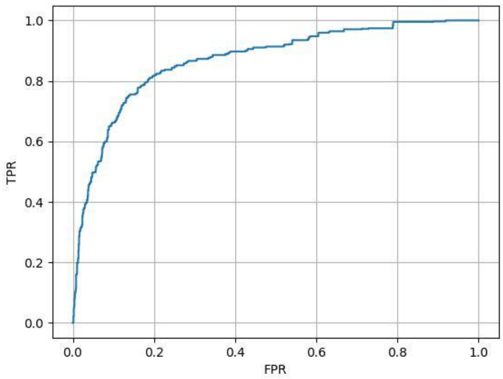

Figure 46. raw data, PCA m=8, π˜=0.5

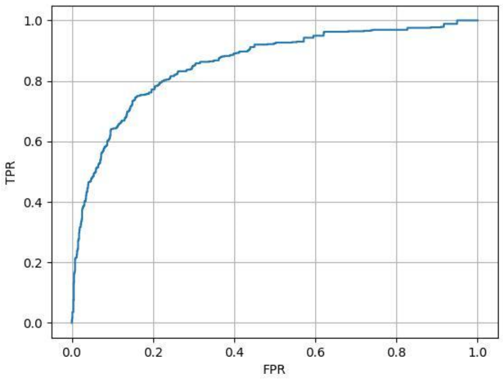

Figure 47. gaussianized, PCA m=8, π˜=0.5

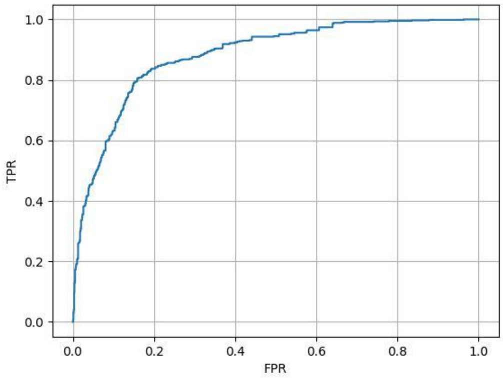

Figure 48. raw data, PCA m=9, π˜=0.5

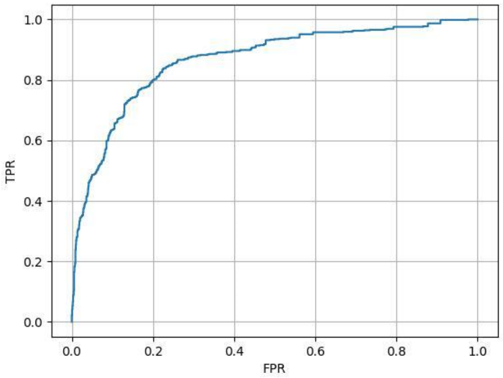

Figure 49. gaussianized, PCA m=9, π˜=0.5

When the priors are different minDCF doesn't change much but when they are different actDCF changes remarkably.

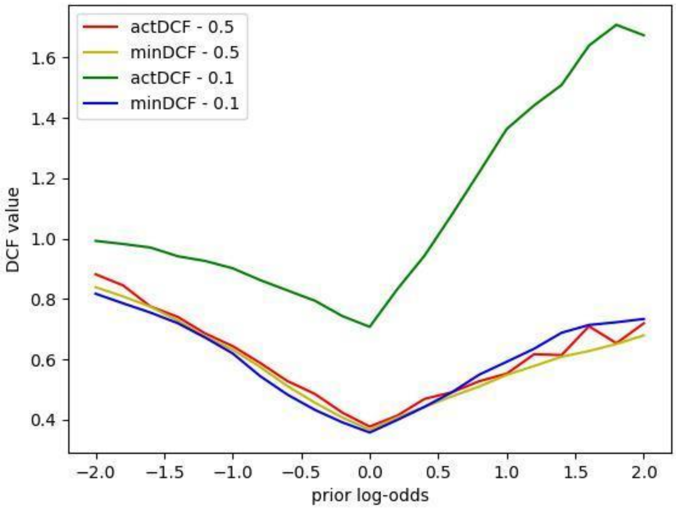

Figure 50. raw data, no PCA, priors comparison

We continue with Support Vector Machines. Again we resort to k-fold k=5. We consider the SVM formulation used in the lab (bias term is simulated through feature extension and is thus regularised). To rebalance the classes we use different C values for different classes.

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

subject to:

$$0 \leq \alpha_i \leq C_i, i \; = \; 1, \ldots \ldots, n$$

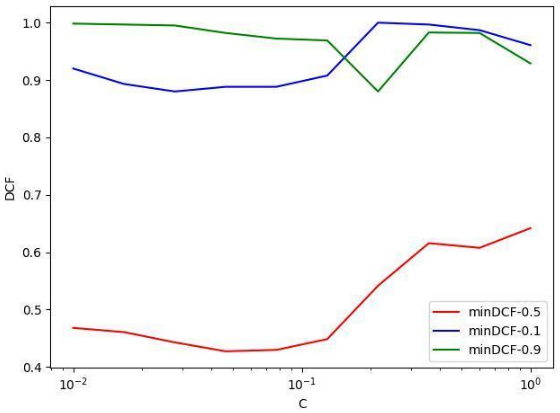

Figure 51. SVM linear, raw data

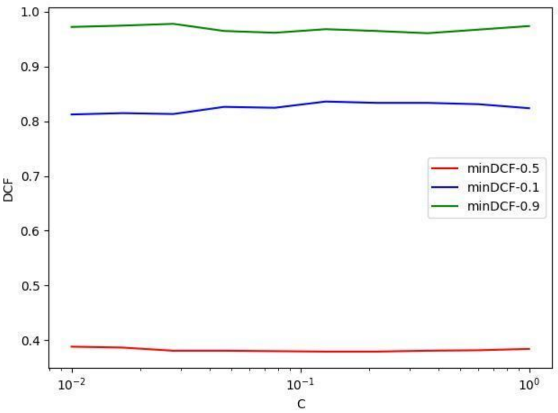

Figure 52. SVM linear, gaussianized data

Our primary metric is again the redline. We compare linear models in terms of minDCF. Linear SVM performs similarly to other linear as expected with a minDCF of 0.38 with gaussianized features.

We also consider two non-linear models on this dataset. The first will use a polynomial quadratic kernel (similar to the quadratic Logistic Regression model, again we expect similar results). The second will employ a Radial Basis Function kernel For the RBF kernel we also need to estimate the kernel width γ. We will use a grid search to jointly optimise C and γ. We will mainly focus on Gaussianized features since they perform much better with this model on this dataset. We will also plot non gaussianized versions to see the difference in results.

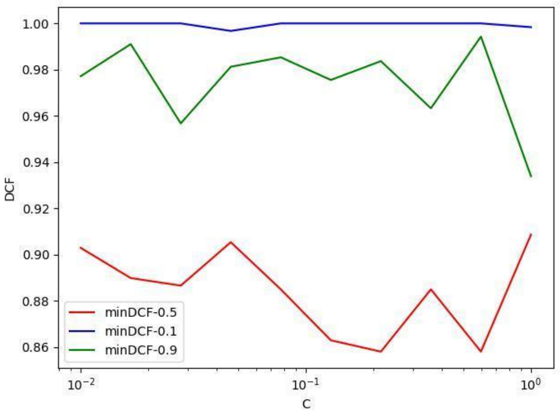

Figure 53. SVM polynomial, raw data

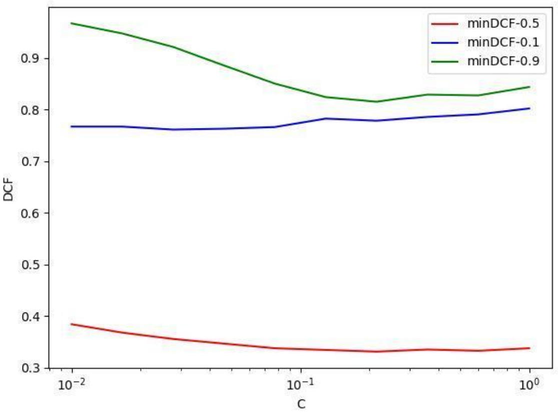

Figure 54. SVM polynomial, gaussianized data

Polynomial Quadratic kernel SVM: minDCF for different values of C, gaussianized features. Our primary metric is the red line

The choice of C does not significantly affect the performance. We again choose C=0.1, since it provides slightly more robust performance for the imbalanced applications. Compared to Full-Cov MVG Polynomial kernel with gaussianized data performs slightly better (Full-Cov minDCF: 0.35).

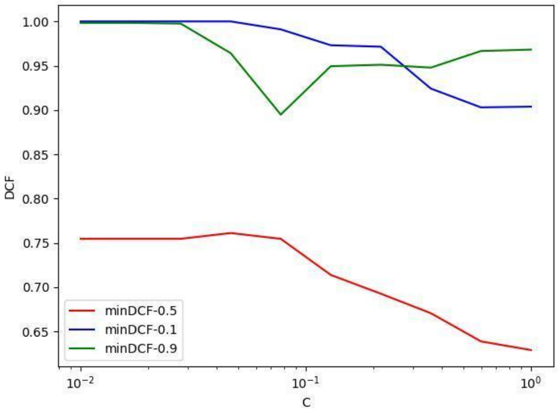

Figure 55. SVM RBF, raw data

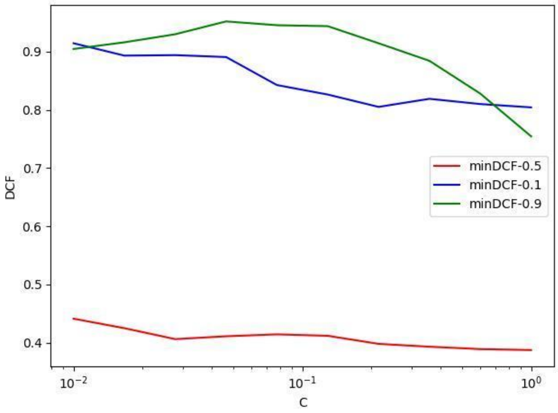

Figure 56. SVM RBF, gaussianized data

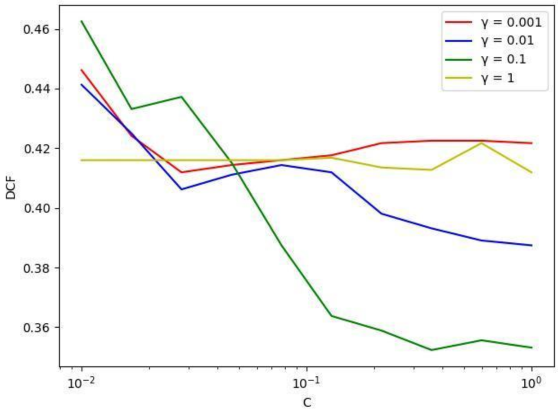

Figure 57. SVM RBF, gaussianized data, variant γ

RBF kernel SVM: min DCF ($\tilde{\pi} = 0.5$) for different values of C and Gaussianized features. Last graph with different γ values. All 3 graphs show that both γ and C affect the results. Both should be optimised jointly since optimal C depends on the chosen γ. Results above are results of k=5. Class re-balancing for the primary task (using the same values for C and as for the imbalanced model) provides very similar results. The RBF kernel SVM provides worse values than the previous models we used. It performs even worse than its gaussianized counterparts when raw features are used.

	Raw features				Gaussianized features		
	$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$		$\tilde{\pi} = 0.5$	$\tilde{\pi} = 0.1$	$\tilde{\pi} = 0.9$
No PCA							
SVM Linear (C = 0.1, K = 1.0)	**0.450**	0.896	0.980		**0.380**	0.831	0.963
SVM Poly (C = 0.1, K = 1.0, d = 2.0, c = 1.0)	**0.941**	1.000	0.971		**0.334**	0.777	0.831
SVM RBF (C = 0.1, K = 1.0, γ = 0.01)	**0.736**	0.982	0.911		**0.416**	0.825	0.953

So far , our best results have been achieved with the SVM Polynomial kernel with gaussianized features.

Bayes Error Plots and ROC curves for the best results of SVM classifiers of different types (highlighted in the table) are shown below.

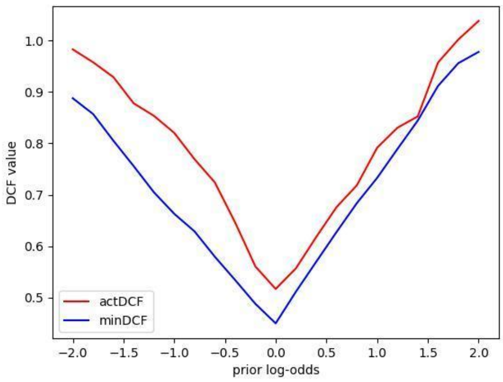

Figure 58. linear raw data

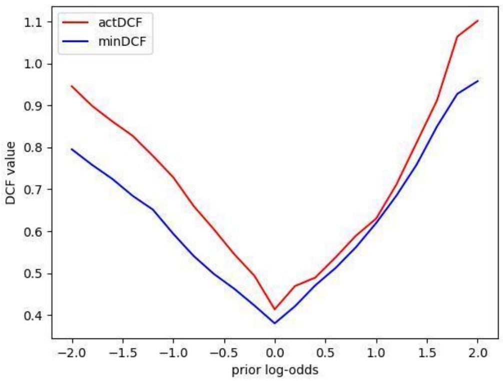

Figure 59. linear gaussianized

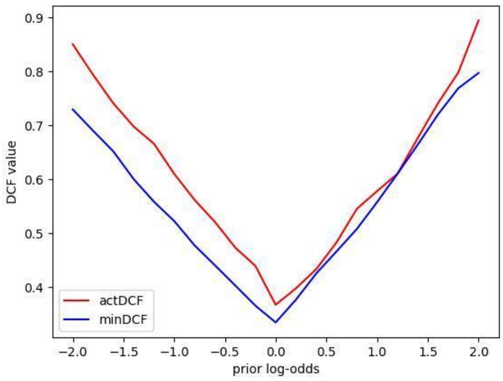

Figure 60. polynomial gaussianized

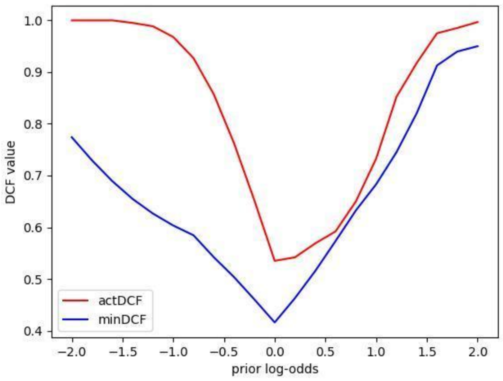

Figure 61. RBF gaussianized

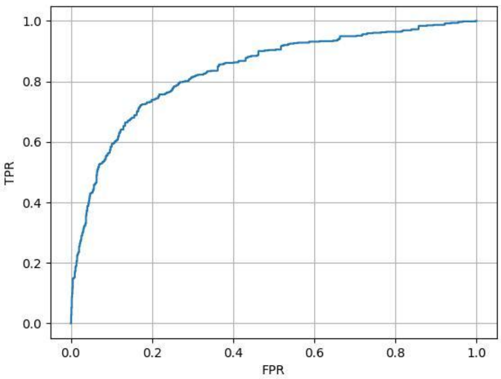

Figure 62. linear raw data

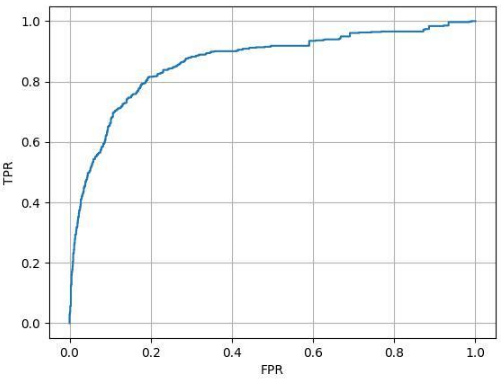

Figure 63. linear gaussianized

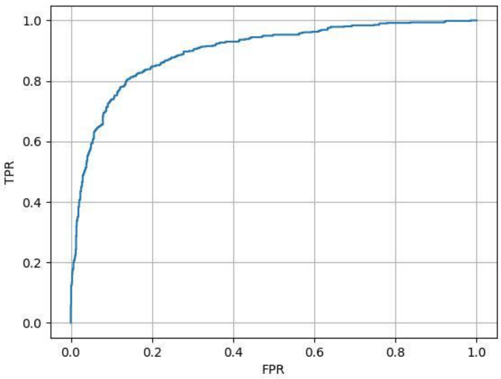

Figure 64. polynomial gaussianized

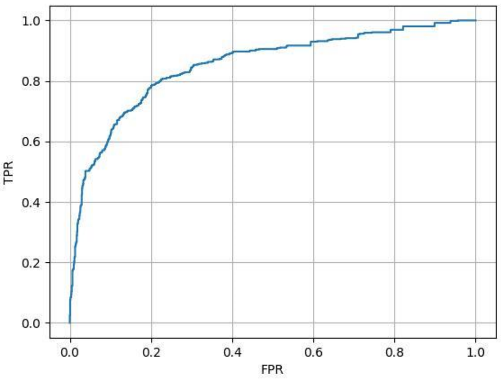

Figure 65. RBF gaussianized

The last model we consider is a generative approach based on training a GMM over the data of each class. We consider both full covariance and diagonal models, with and without covariance tying. For tied covariance models, tying takes place at class level (i.e. different classes have different covariance matrices). We use the K-fold protocol to select the number of Gaussians and to compare different models

Raw data

	Components:	1	2	4	8	16	32
Full-Cov	no PCA:	1.000	0.694	0.820	0.722	0.798	1.000
	PCA, m=8:	1.000	0.711	0.769	0.790	0.838	1.000
	PCA, m=9:	1.000	**0.694**	0.822	0.728	0.801	1.000
Tied Full-Cov	no PCA:	1.000	0.341	0.421	0.538	0.663	0.639
	PCA, m=8:	1.000	0.361	0.439	0.524	0.629	0.675
	PCA, m=9:	1.000	**0.337**	0.419	0.501	0.657	0.644
Diag-Cov	no PCA:	1.000	**0.423**	0.579	0.667	0.839	0.908
	PCA, m=8:	1.000	0.710	0.760	0.823	0.771	1.000
	PCA, m=9:	1.000	0.750	0.825	0.859	0.855	1.000
Tied Diag-Cov	no PCA:	1.000	**0.417**	0.460	0.483	0.522	0.582
	PCA, m=8:	1.000	0.431	0.432	0.426	0.469	0.499
	PCA, m=9:	1.000	0.424	0.424	0.427	0.493	0.551

Gaussianized data

	Components:	1	2	4	8	16	32
Full-Cov	no PCA:	1.000	0.567	0.651	0.936	0.856	0.754
	PCA, m=8:	1.000	**0.525**	0.770	0.875	0.891	0.768
	PCA, m=9:	1.000	0.532	0.923	0.896	0.864	0.678
Tied Full-Cov	no PCA:	1.000	0.443	0.504	0.515	0.523	0.573
	PCA, m=8:	1.000	0.451	**0.416**	0.506	0.521	0.516
	PCA, m=9:	1.000	0.456	0.426	0.501	0.555	0.570
Diag-Cov	no PCA:	1.000	0.478	**0.467**	0.577	0.825	0.816
	PCA, m=8:	1.000	0.554	0.668	0.780	0.975	0.985
	PCA, m=9:	1.000	0.547	0.753	0.775	0.955	0.989
Tied Diag-Cov	no PCA:	1.000	**0.493**	0.508	0.511	0.520	0.507
	PCA, m=8:	1.000	0.509	0.527	0.511	0.523	0.530
	PCA, m=9:	1.000	0.525	0.506	0.520	0.515	0.529

Bayes Error Plots and ROC curves for the best results of GMM classifiers of different types (highlighted in the table) are shown below.

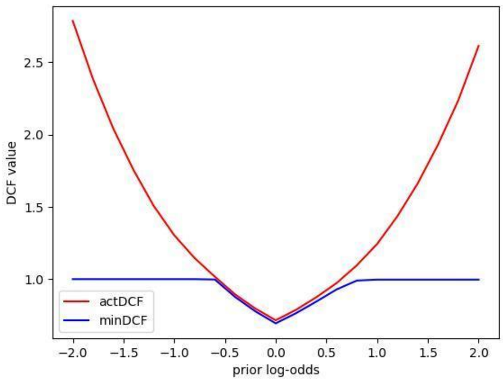

Figure 66. full cov, raw data, PCA m=9

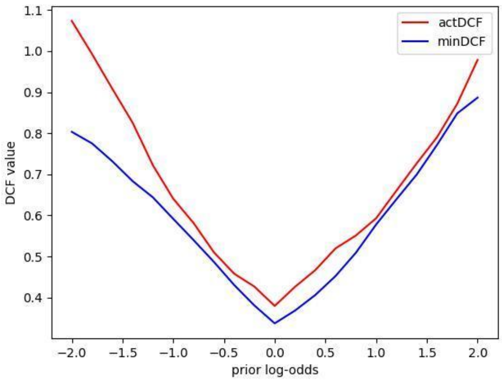

Figure 67. tied full cov, raw data, PCA m=9

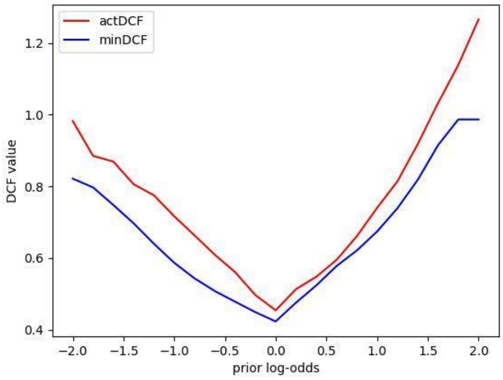

Figure 68. diag cov, raw data, no PCA

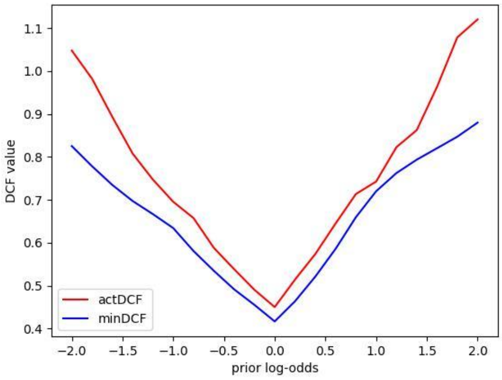

Figure 69.tied diag cov, raw data, no PCA

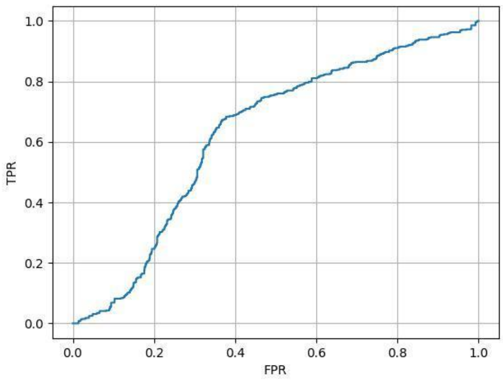

Figure 70. full cov, raw data, PCA m=9

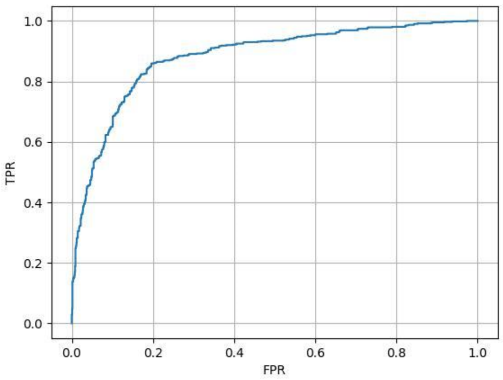

Figure 71. tied full cov, raw data, PCA m=9

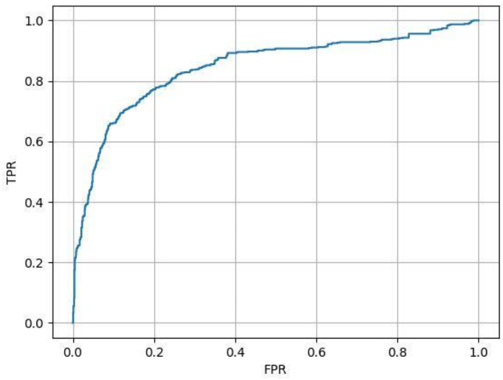

Figure 72. diag cov, raw data, no PCA

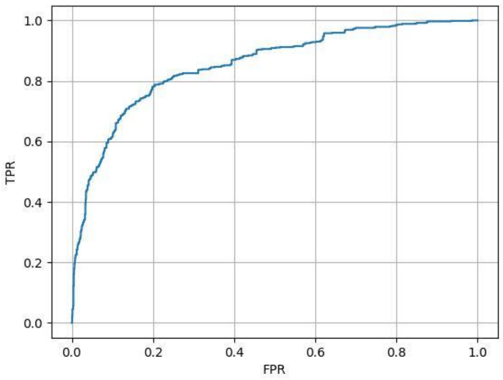

Figure 73. tied diag cov, raw data, no PCA

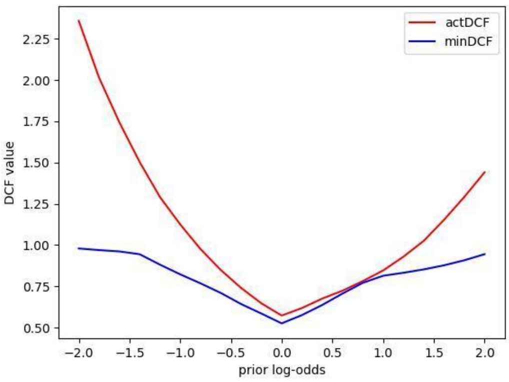

Figure 74. full cov, gaussianized, PCA m=9

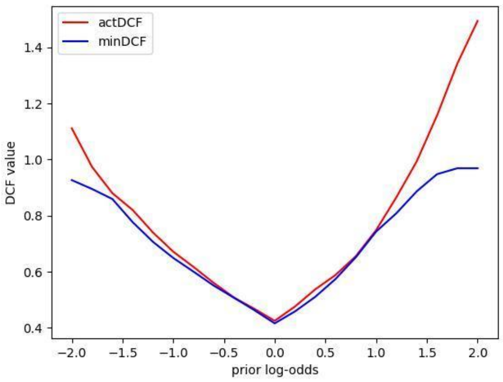

Figure 75. tied full cov, gaussianized, PCA m=8

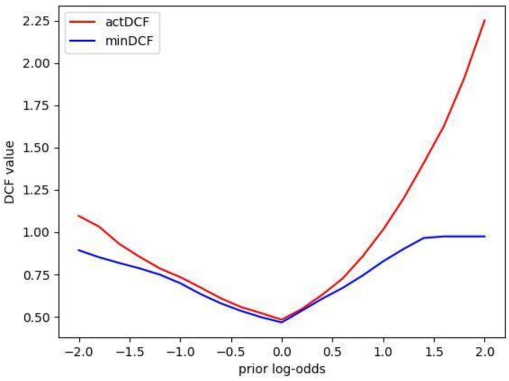

Figure 76. diag cov, gaussianized, no PCA

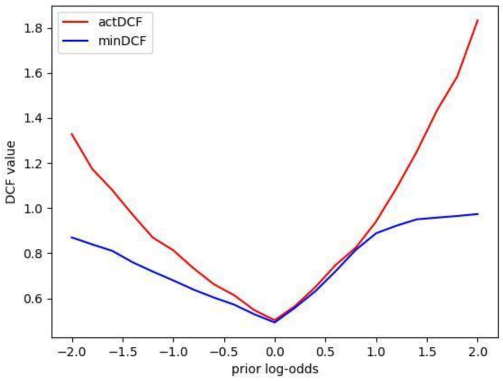

Figure 77. tied diag cov, gaussianized, no PCA

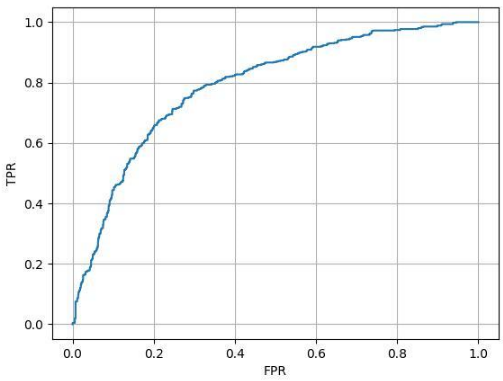

Figure 78. full cov, gaussianized, PCA m=8

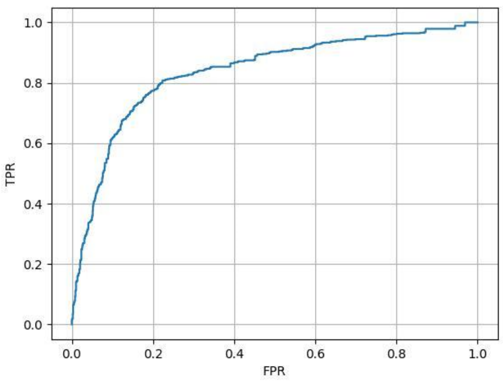

Figure 79. tied full cov, gaussianized, PCA m=8

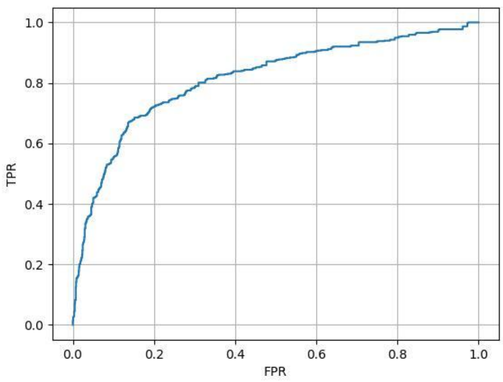

Figure 80. diag cov, gaussianized, no PCA

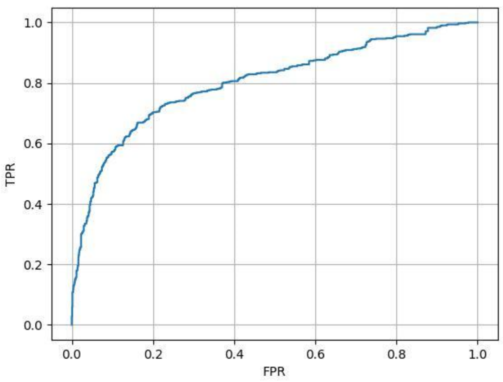

Figure 81. tied diag cov, gaussianized, no PCA

So far our best results have been achieved with SVM Polynomial kernel with gaussianized features. The models trained on gaussianized features show better performance. Further analysis of the dataset shows that, indeed, the data of each wine type presents several clusters that are well separated.

Gaussianization reduces the dynamic range of samples far from the data mean and thus reduces the separability of some clusters. The LBG iterations also split data along different directions, thus finding different 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.

Overall Tied models perform better. This can be explained by the presence of a large number of clusters that have similar within-class variability. With a limited number of components, non-tied models do not identify all the clusters. With lots of components, non-tied models need to estimate one covariance matrix for each cluster from a very small number of samples. Tied models allow for more robust estimates of the within-class covariance of the clusters, and thus we can employ a larger number of components and identify most of the class clusters. In this case, Gaussianization does not have a detrimental effect for models with a large number of components, and the best models are obtained using Raw data.

For evaluation and validation we select as a first candidate model the SVM Polynomial kernel with gaussianized features (SVM Poly (C=0.1, K=1.0, d=2.0, c=1.0)). As a second candidate we select GMM with Tied Multivariate Gaussian Classifier, raw data, PCA (m=9), $\tilde{\pi} = 0.5$. As third we select MVG with Tied Multivariate Gaussian Classifier, raw data, PCA (m=9), $\tilde{\pi} = 0.5$. As fourth and last we select Logistic Regression, raw data, PCA (m = 9), λ = 1e-4, $\pi_T = 0.5$, $\tilde{\pi} = 0.5$

Validation

Bayes Error Plots and ROC curves for the final classifiers are shown below.

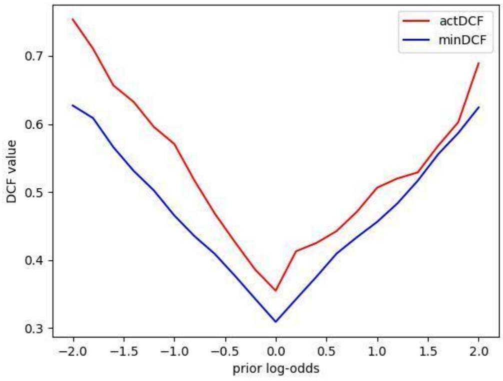

Figure 82. evaluation - SVM

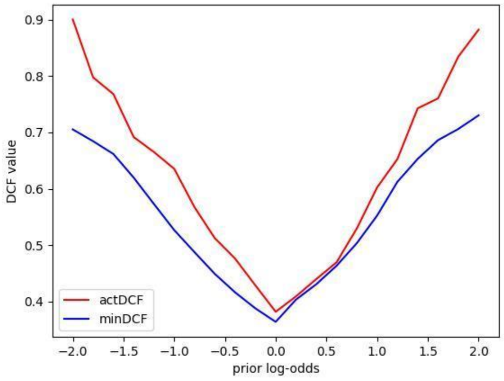

Figure 83. evaluation - GMM

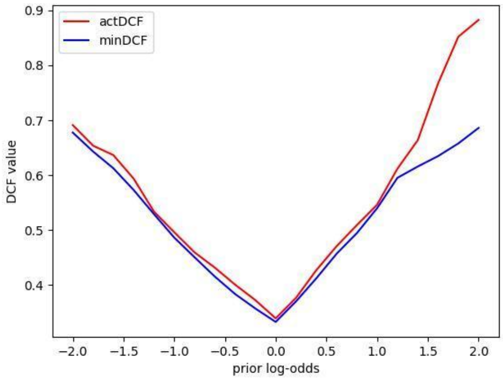

Figure 84. evaluation - MVG

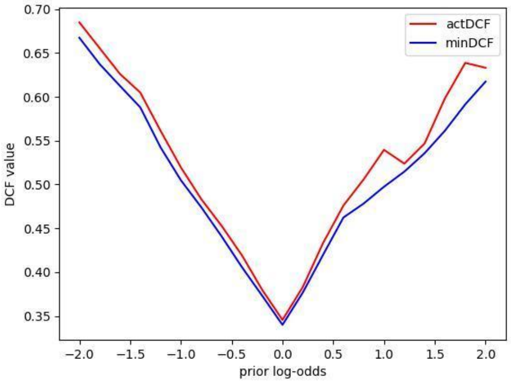

Figure 85. evaluation – logistic regression

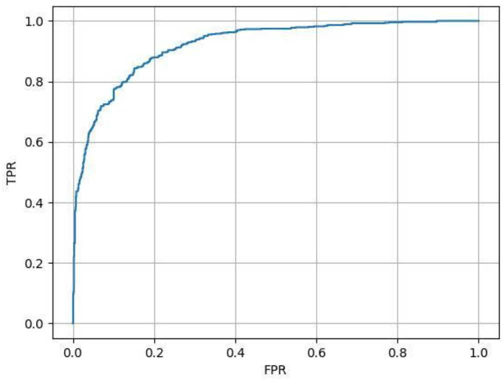

Figure 86. evaluation - SVM

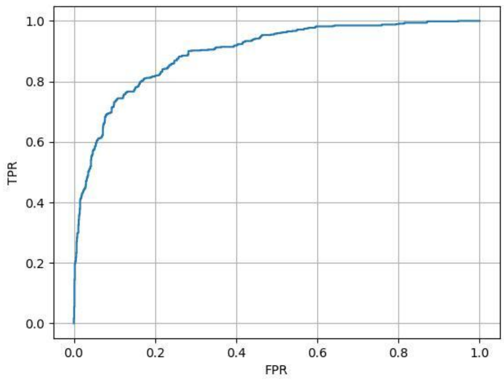

Figure 87. evaluation - GMM

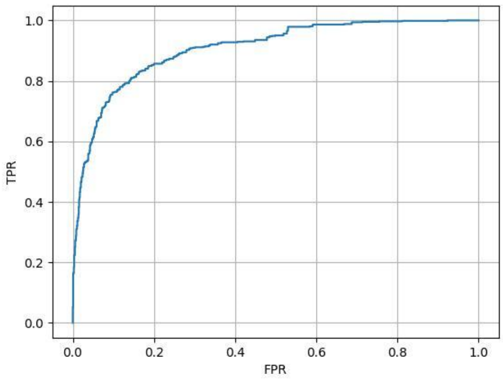

Figure 88. evaluation - MVG

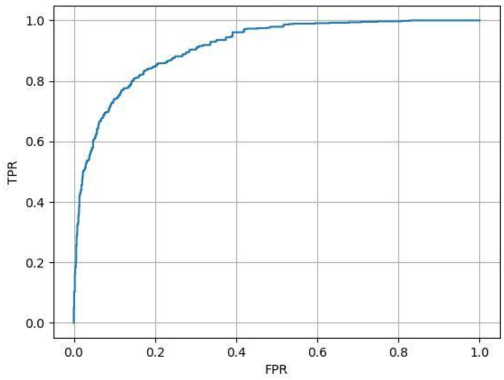

Figure 89. evaluation – logistic regression

Results of model evaluation for SVM:

- minDCF: 0.309
- actDCF: 0.355
- error rate: 15.09 %
- confusion matrix:

	Bad	Good
Bad	1066	183
Good	92	481

Results of model evaluation for GMM:

- minDCF: 0.364
- actDCF: 0.382
- error rate: 20.31 %
- confusion matrix:

	Bad	Good
Bad	885	97
Good	273	567

Results of model evaluation for MVG:

- minDCF: 0.333
- actDCF: 0.339
- error rate: 17.07 %
- confusion matrix:

	Bad	Good
Bad	957	110
Good	201	554

Results of model evaluation for Logistic Regression:

- minDCF: 0.340
- actDCF: 0.346
- error rate: 17.51 %
- confusion matrix:

	Bad	Good
Bad	948	109
Good	210	555

We can see that values attained with K-fold K=5 and evaluations set are consistent. Using the whole dataset improves the result values for all classification models. The results show that our choice of relying on K-fold protocol (K=5) was probably a good decision. The cross-validation values are close to the evaluation values for the K-fold protocol, which suggests that the training and evaluation populations, in this specific use case, have similar properties. We have to make decisions without knowing what is the optimal rule for the evaluation data, we need to map our scores to class labels, but we do not know the optimal score threshold. We want therefore to assess how good are the decisions that we are actually able to make using the recognizer scores .To evaluate the decision making capabilities of our systems, we consider actual DCFs (or just DCFs), which represent the (normalised) Bayes cost we would pay for our actual decisions.

Looking at the minDCF values calculated with test set the lowest value belongs to SVM Polynomial Kernel, gaussianized data, C = 0.1, d = 2.0, c = 1.0, K = 1.0, no PCA, $\tilde{\pi}$ = 0.5 it performs better compared to other models evaluated.

Looking at the actDCF values calculated with test set the lowest value belongs to Tied Multivariate Gaussian Classifier, raw data, PCA (m = 9), $\tilde{\pi}$ = 0.5.

Looking at the error rates SVM Polynomial Kernel, gaussianized data, C = 0.1, d = 2.0, c = 1.0, K = 1.0, no PCA, $\tilde{\pi}$ = 0.5 has the lowest error rate among all.Despite the model lacking a probabilistic interpretation.

At the end having decided on the best classifiers we can combine them computing scores for fusion. In that approach we need to compute fusion of scores acquired from different classifiers. In our case those classifiers will be GMM and SVM for the parameters chosen above.

Results of model evaluation for fusion between SVM and GMM:

- minDCF: 0.324
- actDCF: 0.332
- error rate: 16.25 %
- confusion matrix:

	Bad	Good
Bad	981	119
Good	177	545

Bayes Error Plot and ROC curve for the fused scores are shown below.

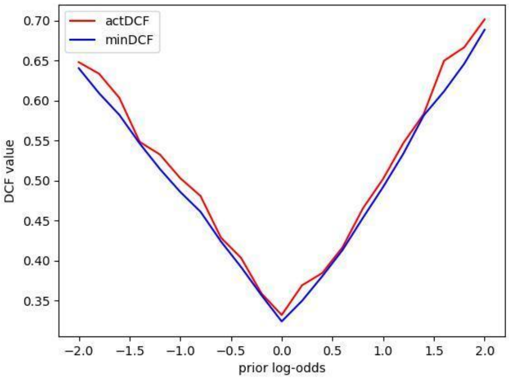

Figure 90. Bayes error plot for fused scores

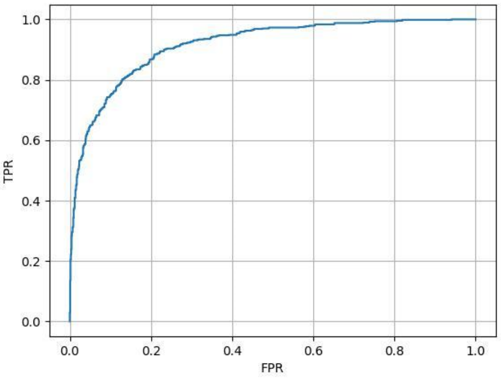

Figure 91. ROC plot for fused scores

The results of fused scores present really good and balanced results compared to individual classifiers both in terms of actual DCF and error rate.

Looking at the evaluation values done with the test set in general results are very similar but the model you should prefer depends on what you aim to achieve.

Conclusions

If we look at the actDCF values we obtained with evaluation (test) set lowest actDCF values belong to model Tied Multivariate Gaussian Classifier, raw data, PCA (m=9), $\tilde{\pi}$ = 0.5.

When we look at our other two evaluation methods the minDCF and error rate in both of those values SVM Polynomial Kernel, gaussianized data, C = 0.1, d = 2.0, c = 1.0, K = 1.0, no PCA, $\tilde{\pi}$ = 0.5 have performed the best.

When we again look at the values we got with K-Fold approach and evaluation set we can see that with k=5 we get good results and values we obtain are close to ones we obtain with validation (test) set but still when we test it with validation (test) set and use 100% of the training data all of our selected models perform better. The choices we made on our training / validation sets proved effective also for the evaluation data.

Finally on conclusion, even though Tied Multivariate Gaussian Classifier, raw data, PCA (m = 9), π tilde = 0.5 have performed best on actDCF with evaluation set considering the values we got with k-fold approach on the previous sections and with the validation results different than actDCF (minDCF, error rate) SVM Polynomial Kernel, gaussianized data, C = 0.1, d = 2.0, c = 1.0, K = 1.0, no PCA, $\tilde{\pi}$= 0.5 has performed better on many different occasions, especially it has a much lower error rate on the evaluation(test) set.

Thus our final decision for this dataset when considering single classifier is **SVM Polynomial Kernel, gaussianized data, C = 0.1, d = 2.0, c = 1.0, K = 1.0, no PCA, $\tilde{\pi}$ = 0.5.** Although taking fusion models into consideration (combination of GMM and SVM with parameters mentioned above) the results are more balanced in terms of different model evaluation classifiers and it could be a better fit for our dataset.