# Machine learning and pattern Recognition Final project

PULSAR DETECTION

By

MUHAMMAD SARIB KHAN s298885

# AIM of this project

Analysis of htru2 dataset by using machine learning algorithms to portray their behavior on an unbalanced dataset. It is expected that features are related and distributed. i will be using various models and evaluate their performance, and ultimately find out which models are the better ones and how I precise results can be generated

# Dataset

The HTRU2 dataset describes a sample of pulsar candidates collected during the high time resolution universe survey. Pulsars, a rare kind of neutron star, are capable of producing radio emissions that are detectable on earth. Their rotation cause their emission beam to sweep across the sky and when our line of sight is crossed a detectable pattern of broadband radio emission is produced. Rapid rotation cause the pattern to repeat periodically. A slightly different pattern is produced by different pulsars which also varies slightly with each rotation. The candidate, which is here defined as potential signal detection is a result of averaging over many rotations of a pulsar. If no additional info is at hand, every candidate can be describing a real pulsar. In practice nearly all detections are caused by radio frequency interference(RFI) and noise, making legitimate signals hard to find. To facilitate rapid analysis, machine learning tools are employed to automate the process of labelling candidates, especially the classification systems as then the candidate data sets are treated as binary classification problems.

# Features of htru2

Eight continuous variables are describing the candidate. These are as follows:

- Mean of the integrated profile
- Standard deviation of the integrated profile
- Excess kurtosis of the integrated profile
- Skewness of the integrated profile
- Mean of the dm-snr curve
- Standard deviation of the dm-snr curve
- Excess kurtosis of the dm-snr curve
- Skewness of the dm-snr curve

# Features of htru2(cont'd)

- All of the 8 features have enough difference in means and variances, so z-normalization is applied to center every feature to its mean and scale it to the unit variance. It is to be considered that in the concerned data set, there are 16259 negative pulsar signals (causes by rfi/noise) and 1639 real pulsar signals making it a total of 17898 samples. The formula of the z-normalization is (u and sigma are column vector containing values of mean and standard deviation of 8 features):

$$x_i = \frac{x_i - \mu}{\sigma}$$

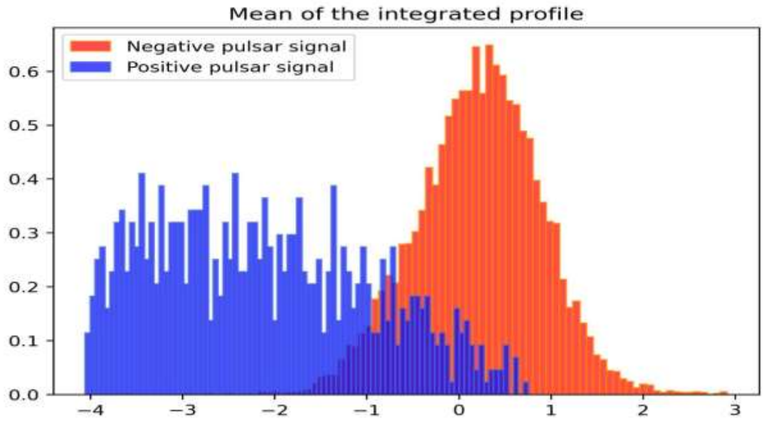
# Features of htru2(cont’d)

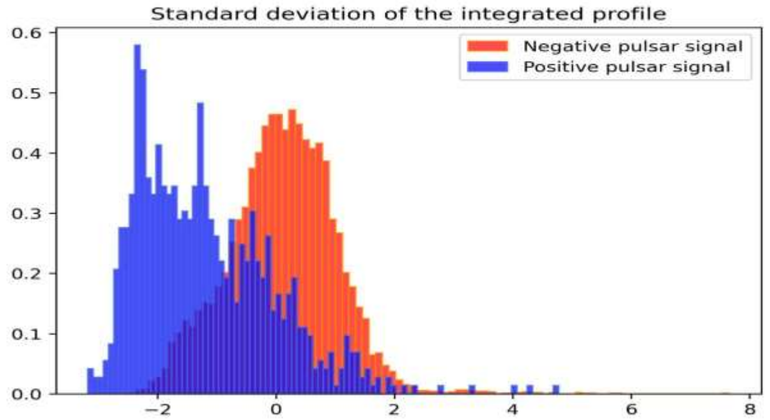
- We have a training data set comprised of 8929 samples and one test data set comprised of 8969 samples. For analysis of models only training set will be used and test set will be used at the end for validation. The plotting of data (2D Principal component analysis scatterplot):

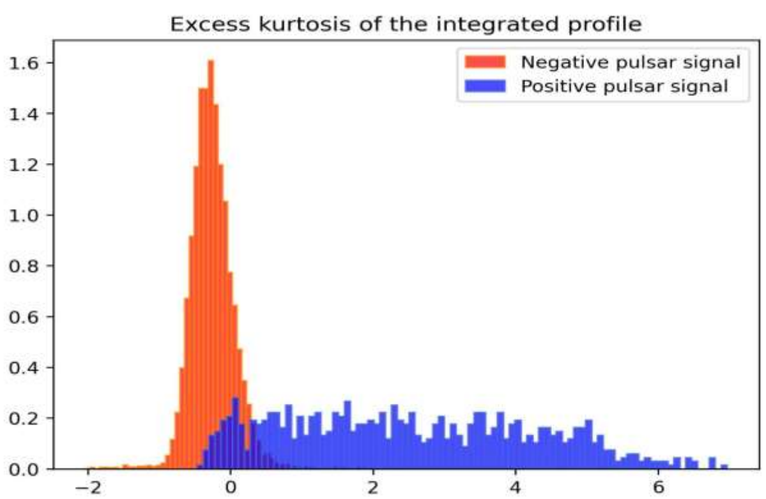
# Z-normalized histogram plots of features

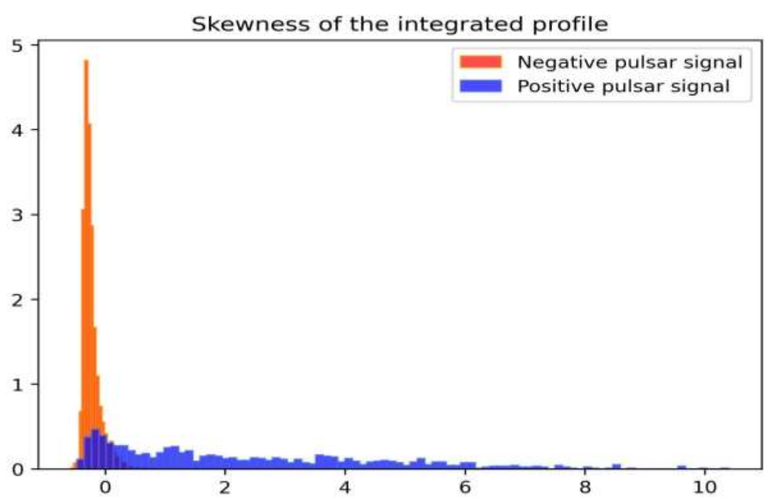
- It is to be mentioned that classifiers other than gaussian classifiers and non-regularized logistic regression will be affected by this scaling. It might even benefit pca which has its bias towards the highest variance.

# Histograms

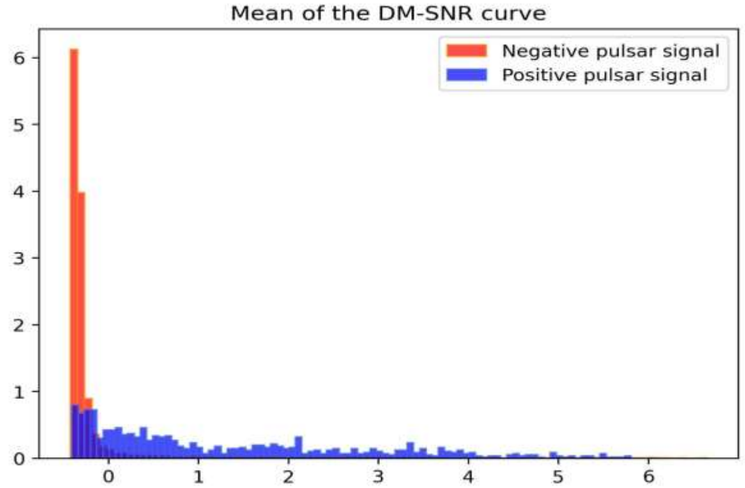

Mean of the integrated profile
Negative pulsar signal
Positive pulsar signal

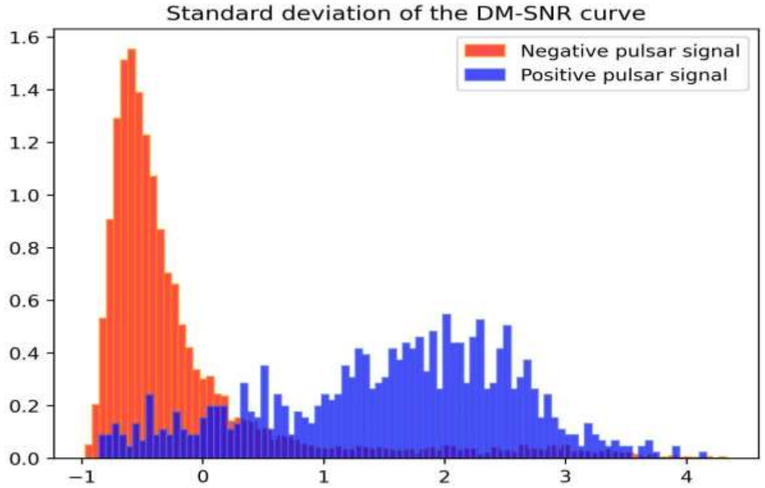

Standard deviation of the integrated profile
Negative pulsar signal
Positive pulsar signal

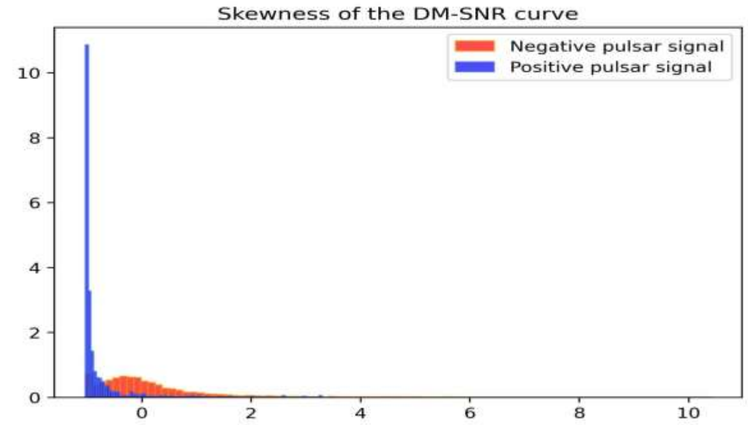

Excess kurtosis of the integrated profile
Negative pulsar signal
Positive pulsar signal

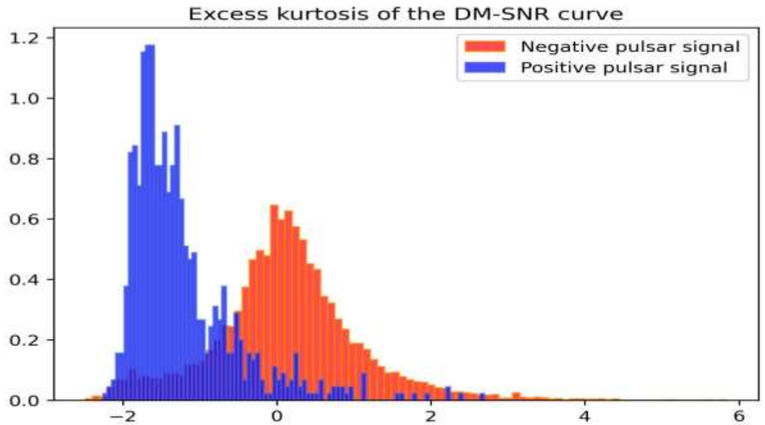

Skewness of the integrated profile
Negative pulsar signal
Positive pulsar signal

# Histograms


Mean of the DM-SNR curve
Negative pulsar signal
Positive pulsar signal


Standard deviation of the DM-SNR curve
Negative pulsar signal
Positive pulsar signal


Skewness of the DM-SNR curve
Negative pulsar signal
Positive pulsar signal


Excess kurtosis of the DM-SNR curve
Negative pulsar signal
Positive pulsar signal

# Histograms evaluation

- We can infer from the plots that the features are very decently distributed and very separable. it can be argued that there are some outliers but they are not very evident or downright outliers. Because of this, the decision was to opt out of applying gaussianization or any other preprocessing techniques.

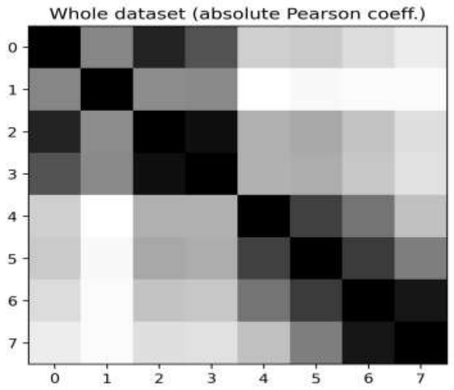
# Pearson correlation

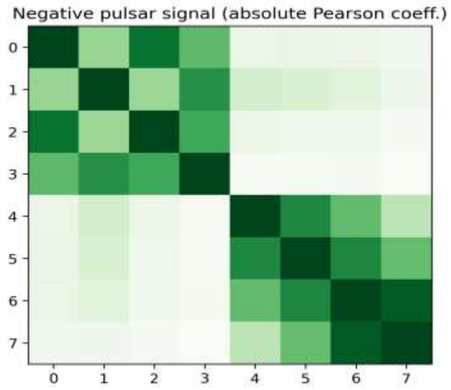
- The next step is to plot heatmaps regarding correlation. The absolute value of pearson correlation coefficient will be used for this. By doing this, we can understand as to how to apply pca on the concerned dataset.

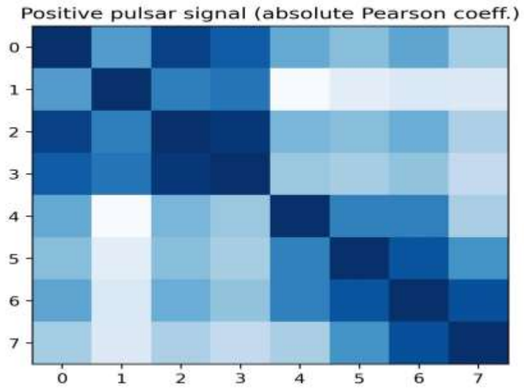
$$corr = \left| \frac{Cov(X,Y)}{\sqrt{Var(X)}\sqrt{Var(Y)}} \right|$$

# Heatmaps

we can see here that strongest correlations can be found among 1&3, 3&4 and 7& 8. Now by employing pca we can reduce dimensionality to map samples from a higher dimension(8th dimension here) to lower dimension. We will try different value of m to observe the behavior of the models. At the same time, the expectation is that information will be lost as we go down the dimensions.



# Methodologies of analysis

- Single split methodology is applied at first. Training set is split into two subset where one will be used for training and the other for validation. This makes it 66% training and 33% validation. Finally, we will get the classifier evaluated on the validation set.
- Using K-Fold cross validation methodology where the training set will be divided k-times into k-number of distinct sets. K-1 sets will be used for training and the remaining one is used for validation. By retraining model of over total training set we will get our required classifier. Here better and more robust results are expected since the dataset is highly imbalanced.

# Gaussian classifiers

Different values of m will be used to test pca. The number of dimensions is not too high, so the performance of pca is expected to drop as the value of m is decreased. Dimensionality reduction will only be applied on training data to protect the model from any bias.

Main uniform prior:

(π, Cfp, Cfn) = (0.5, 1, 1)

Imbalanced :

(π, Cfp, Cfn) = (0.1, 1, 1) & (π, Cfp, Cfn) = (0.9, 1, 1)

Here π is the bias towards the positive class wrt negative class (effective prior)

# Gaussian Classifiers (Cont'd)

- By considering gaussian classifiers like full covariance and naïve bayes(Diagonal covariance), we will start our analysis on our models. It is to remember that though every class its on mean $u_c$, the covariance matrix Σ is over the total dataset.
- We assume here that the data is following gaussian distribution:

$$Xi \mid Ci = c, \Theta) \sim (X \mid C = c, \Theta) \sim N(\mu c(\text{edit later}) , \Sigma c)$$

# Gaussian classifiers

- Naïve bayes is expected to give the most subpar results because it assumes that features are being independently distributed. Over here, features are badly correlated with no covariance which will eventually lead to off-diagonal elements of covariance matrix to be zero. But as shown in heat maps above, our features are correlated, the results are expected to be sub-optimal.
- Due to the ability of capturing correlations and large number of samples, the tied version of the full covariance should provide better results.

# Gaussian classifiers(Single split)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.197 | 0.322 | 0.747 |
| Diagonal-Covariance | 0.225 | 0.431 | 0.697 |
| Tied Full-Covariance | 0.186 | 0.296 | 0.613 |
| Tied Diagonal-Covariance | 0.192 | 0.327 | 0.640 |

No PCA

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.135 | 0.271 | 0.697 |
| Diagonal-Covariance | 0.208 | 0.474 | 0.655 |
| Tied Full-Covariance | 0.105 | 0.234 | 0.477 |
| Tied Diagonal-Covariance | 0.135 | 0.267 | 0.544 |

PCA m=7

# Gaussian classifiers(Single split)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.156 | 0.261 | 0.663 |
| Diagonal-Covariance | 0.217 | 0.477 | 0.673 |
| Tied Full-Covariance | 0.148 | 0.253 | 0.509 |
| Tied Diagonal-Covariance | 0.167 | 0.276 | 0.577 |

PCA m=6

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.155 | 0.244 | 0.739 |
| Diagonal-Covariance | 0.210 | 0.424 | 0.722 |
| Tied Full-Covariance | 0.155 | 0.247 | 0.498 |
| Tied Diagonal-Covariance | 0.171 | 0.288 | 0.627 |

PCA m=5

# Gaussian classifiers(Single split)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.176 | 0.296 | 0.865 |
| Diagonal-Covariance | 0.208 | 0.400 | 0.674 |
| Tied Full-Covariance | 0.154 | 0.250 | 0.492 |
| Tied Diagonal-Covariance | 0.171 | 0.291 | 0.614 |

PCA m=4

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.197 | 0.322 | 0.747 |
| Diagonal-Covariance | 0.225 | 0.431 | 0.697 |
| Tied Full-Covariance | 0.186 | 0.296 | 0.613 |
| Tied Diagonal-Covariance | 0.192 | 0.327 | 0.640 |

PCA m=3

# Gaussian classifiers(7-folds)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.141 | 0.286 | 0.672 |
| Diagonal-Covariance | 0.193 | 0.315 | 0.747 |
| Tied Full-Covariance | 0.112 | 0.224 | 0.573 |
| Tied Diagonal-Covariance | 0.161 | 0.267 | 0.579 |

No PCA

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.139 | 0.304 | 0.641 |
| Diagonal-Covariance | 0.214 | 0.506 | 0.724 |
| Tied Full-Covariance | 0.112 | 0.223 | 0.572 |
| Tied Diagonal-Covariance | 0.138 | 0.271 | 0.601 |

PCA m=7

# Gaussian classifiers(7-folds)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.152 | 0.289 | 0.650 |
| Diagonal-Covariance | 0.223 | 0.526 | 0.721 |
| Tied Full-Covariance | 0.140 | 0.259 | 0.580 |
| Tied Diagonal-Covariance | 0.164 | 0.298 | 0.589 |

PCA m=6

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.150 | 0.250 | 0.642 |
| Diagonal-Covariance | 0.220 | 0.454 | 0.733 |
| Tied Full-Covariance | 0.150 | 0.262 | 0.574 |
| Tied Diagonal-Covariance | 0.171 | 0.312 | 0.604 |

PCA m=5

# Gaussian classifiers(Single split)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.179 | 0.326 | 0.821 |
| Diagonal-Covariance | 0.208 | 0.446 | 0.690 |
| Tied Full-Covariance | 0.150 | 0.261 | 0.572 |
| Tied Diagonal-Covariance | 0.171 | 0.313 | 0.604 |

PCA m=4

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Full-Covariance | 0.197 | 0.353 | 0840 |
| Diagonal-Covariance | 0.225 | 0.474 | 0.658 |
| Tied Full-Covariance | 0.183 | 0.309 | 0.591 |
| Tied Diagonal-Covariance | 0.186 | 0.336 | 0.599 |

PCA m=3

# Gaussian classifiers (Cont'd)

- From the results, the assumptions made on naïve bayes and tied full covariance prove to be correct. Additionally, there is consistence between k- folds and single split as well, however the results from k-fold are expected to be more robust.
- Keeping m=7  and applying pca results are considerably consistent with results obtained on z-normalized data. Reducing will cause slump in performance.
- Linear separation rule of tied full-cov has outperformed the quadratic one given by full-cov model. It can be argued here (and only here on the basis of these results) that linear classifiers behave better.

# Logistic regression

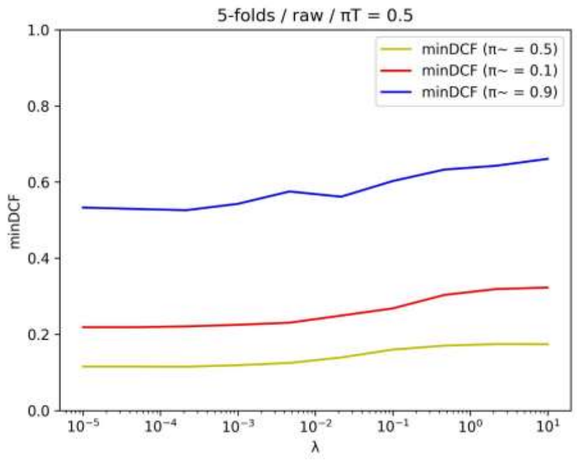
- A regularized version of objective function is used because classes are unbalanced:

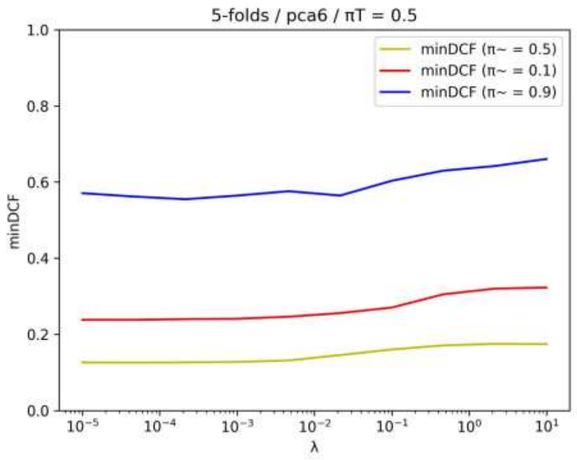
$$J(w,b) = \frac{\lambda}{2}\|w\|^2 + \frac{\pi_T}{n_T}\sum_{i=1|c_i=1}^{n}\log\left(1+e^{-z_i(w^T x_i+b)}\right) + \frac{1-\pi_T}{n_F}\sum_{i=1|c_i=0}^{n}\log\left(1+e^{-z_i(w^T x_i+b)}\right)$$

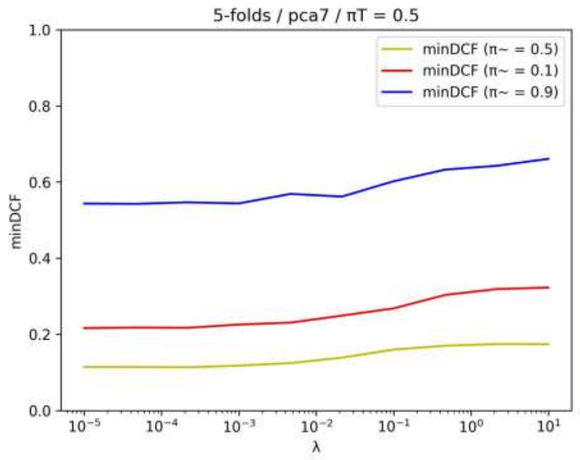
- We expect this linear classifier to work good since we obtained good results by tied-covariance gaussian models.
- Λ is our hyperparameter called regularization coefficient. Λ >> 0 means there is poor separation of classes and a small ||w||. If regularization coefficient is almost equal to zero, then the classes are well separated but generalization on unseen data is subpar
- Only considering our main application we will set πT to 0.5 and plot mindcf against regularization coefficient

# Logistic regression

- From the plots it can be observed that $10^{-5}$ can prove to be a decent value for regularization coefficient.
- It can also be observed that both pca and no pca, the results are not too different.



# Logistic regression(5-folds)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| LogReg(λ = $10^{-5}$ , πT = 0.5) | 0.126 | 0.238 | 0.571 |
| LogReg(λ = $10^{-5}$ , πT = 0.1) | 0.126 | 0.243 | 0.588 |
| LogReg(λ = $10^{-5}$ , πT = 0.9) | 0.134 | 0.241 | 0.520 |

No PCA

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| LogReg(λ = $10^{-5}$ , πT = 0.5) | 0.116 | 0.218 | 0.542 |
| LogReg(λ = $10^{-5}$ , πT = 0.1) | 0.111 | 0.211 | 0.562 |
| LogReg(λ = $10^{-5}$ , πT = 0.9) | 0.118 | 0.218 | 0.524 |

PCA m=7

# Logistic regression (5-folds)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| LogReg(λ = $10^{-5}$ , πT = 0.5) | 0.126 | 0.238 | 0.571 |
| LogReg(λ = $10^{-5}$ , πT = 0.1) | 0.126 | 0.243 | 0.588 |
| LogReg(λ = $10^{-5}$ , πT = 0.9) | 0.134 | 0.241 | 0.520 |

PCA m = 6

# Logistic regression

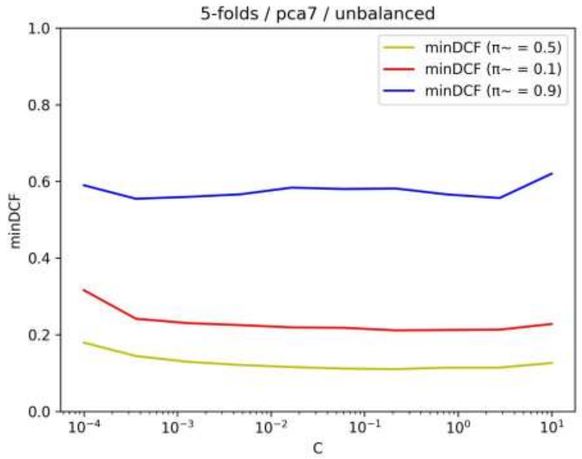
- We can see ultimately good performance from logistic regression. Linear models are continuing to perform better than quadratic ones.
- Pca with m = 7  performs the best
- Also, the classifier with $\pi T = 0.1$ has better performance than the others for the $\pi T=0.5$ of main application.  This is attributed to the class imbalance.

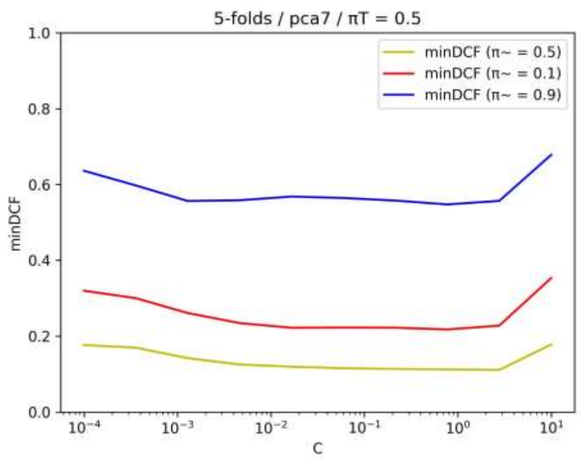
# Support vector machines

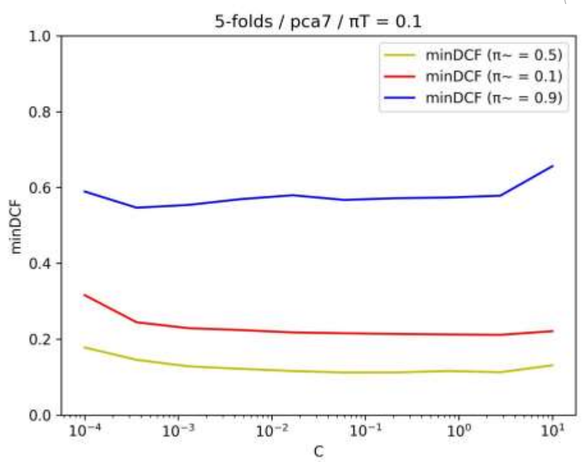
- Considering linear svm, tuning of hyperparameter c is required to balance the classes. Different values of c are considered. The constraint of dual formulation is:

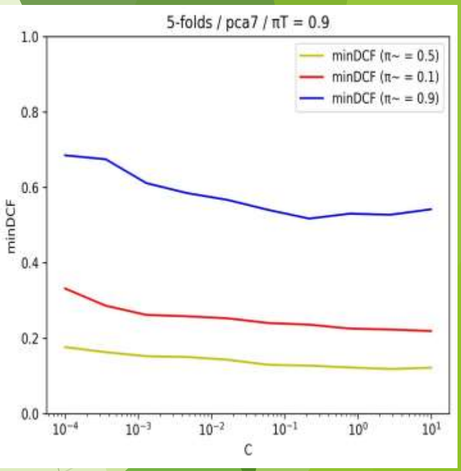
- Πem $C_T = C\frac{\pi_T}{\pi_T^{emp}}$al prior $0 \leq \alpha_i \leq C_i, \forall i \in \{1, ..., n\}$ ו trainir $C_F = C\frac{\pi_F}{\pi_F^{emp}}$
- In order to select our best value for c, mindcf graphs are plotted for both balanced and unbalanced svm.

# Linear Support vector machine plots



# Support vector machines

- We will be choosing our c as $10^{-2}$ for both applications since it is observed that the performance of two kinds of model are very similar in terms of mindcf
- Here we can see that pca m= does not made the results bad. They are the same (more or less).
- Only unbalanced application will be considered since rebalancing is not making a lot of difference.
- $\pi_T$=0.1 provided minutely better results.

# Linear SVM(5-folds)

No PCA

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Linear SVM(C = $10^2$ , unbalanced) | 0.117 | 0.223 | 0.580 |
| Linear SVM(C = $10^2$ , π = 0.5) | 0.120 | 0.225 | 0.564 |
| Linear SVM(C = $10^2$ , π =0.1) | 0.119 | 0.224 | 0.576 |
| Linear SVM(C = $10^2$ , π =0.9) | 0.145 | 0.253 | 0.558 |

PCA m= 7

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Linear SVM(C = $10^2$ , unbalanced) | 0.117 | 0.223 | 0.580 |
| Linear SVM(C = $10^2$ , π = 0.5) | 0.120 | 0.225 | 0.564 |
| Linear SVM(C = $10^2$ , π =0.1) | 0.119 | 0.224 | 0.576 |
| Linear SVM(C = $10^2$ , π =0.9) | 0.145 | 0.253 | 0.558 |

# Quadratic support vector machines

For quadratic svms, dual formulation is dependent on dot products:

$$H_{ij} = z_i z_j x^t_i x_j$$

No need for any explicit feature expansion since this would be enough to calculate the scalar products between test and training samples.

We can use a kernel function for training and scoring:

$$k(x_i , x_j ) = \Phi(x_i)^t \Phi(x^t{}_j )$$

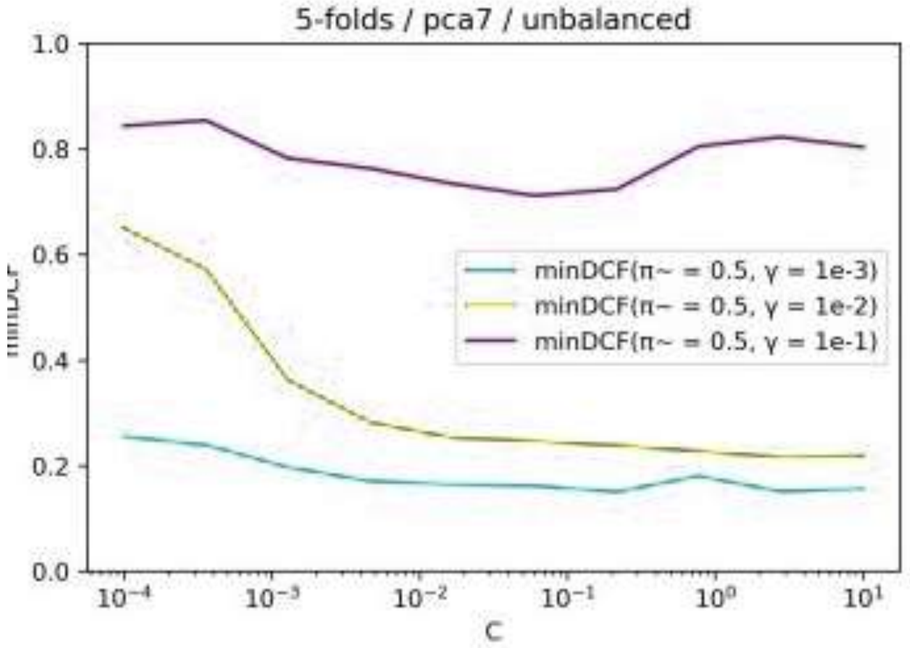
There are two types of different kernel functions we can utilize.

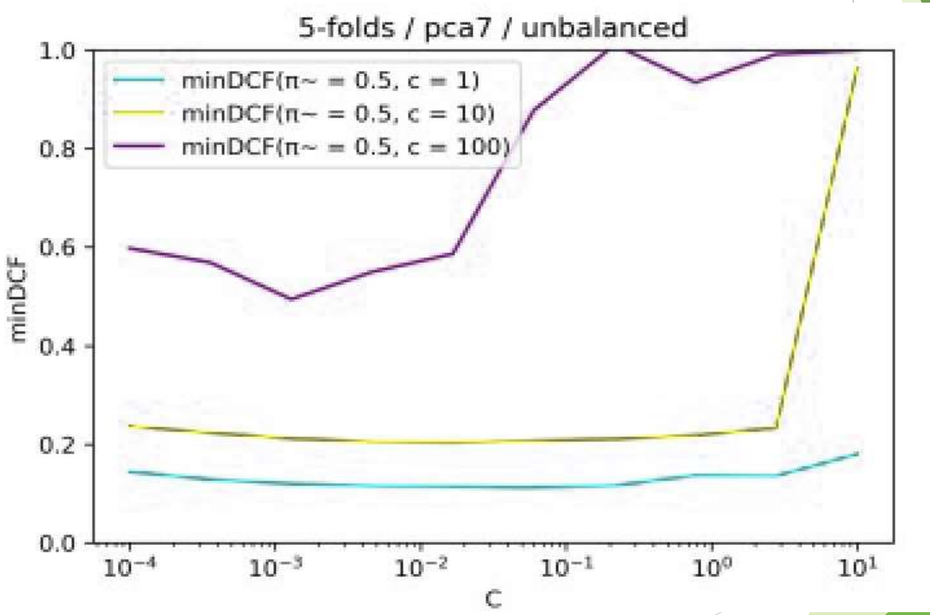
RBF (Radial basis function): $e^{-y||x_i - x_j||^2}$

Polynomial: $(x^t_i x_j + c)^d$

Mindcf graphs were plotted again to select the value of concerned parameters(C, y, c)

# Quadratic SVm plots



# Quadratic SVm(5-folds)

No PCA

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| RBF SVM($C = 10^{-1}$ , $\gamma = 10^{-3}$ ) | 0.156 | 0.260 | 0.582 |
| Poly SVM($C = 10^{-3}$ , c = 1, d = 2) | 0.130 | 0.232 | 0.705 |

PCA m=7

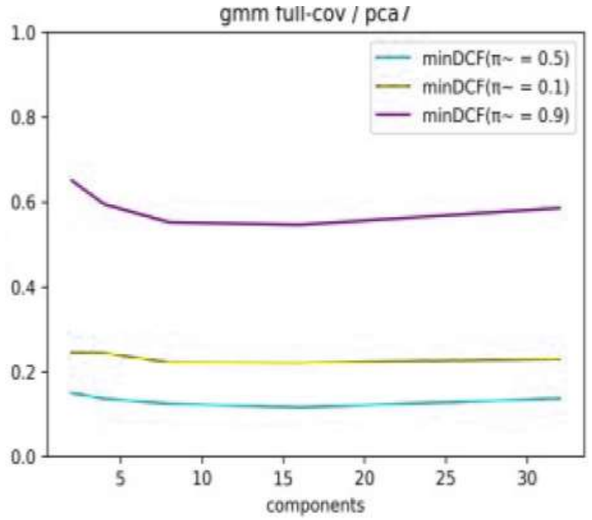
| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| RBF SVM($C = 10^{-1}$ , $\gamma = 10^{-3}$ ) | 0.156 | 0.260 | 0.582 |
| Poly SVM($C = 10^{-3}$ , c = 1, d = 2) | 0.130 | 0.232 | 0.705 |

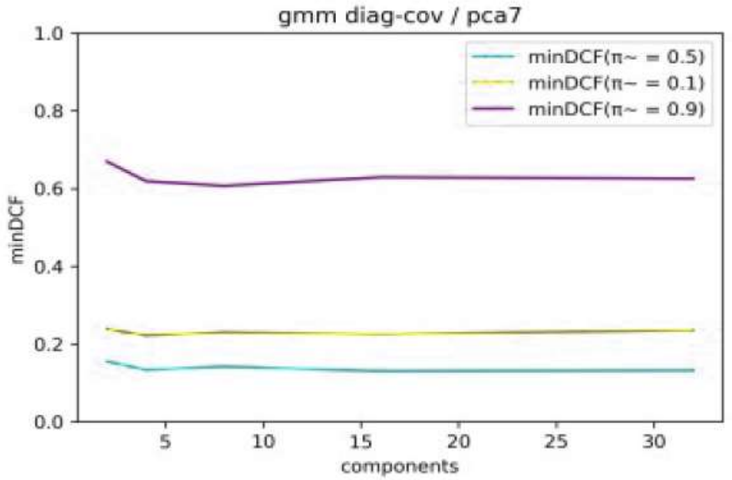
# Quadratic svm

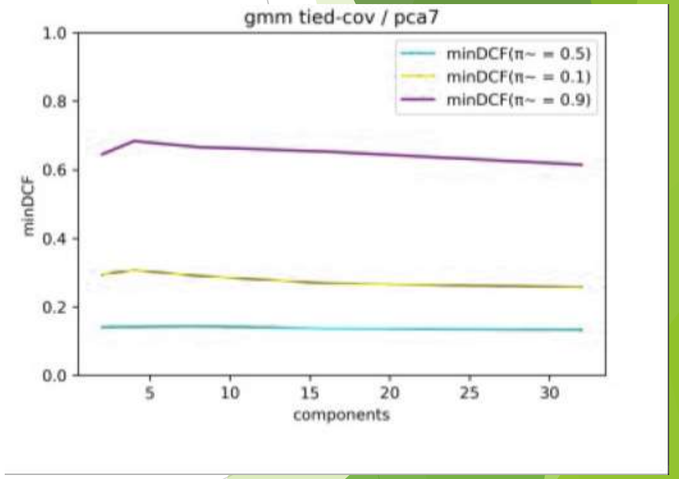
- Here we can see the quadratic classifer performed poorly than the linear ones. Henceforth, there is no good reason to consider these in calibration and evaluation as possible candidates.

# Gaussian mixture model

- The last model is gaussian mixture model. This model assumes that a mixture of finite number of gaussian distribution with unknown parameters leads to generation of each sample. The number of these distributions serves as a hyperparameter. Different types of gmm are plotted to find out correct number of components



# GMM

- GMM full cov with 8 components performs the best here. It will be taken into consideration as a possible candidates along with tied full-cov, linear svm and logistic regression.

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| GMM Full Cov (8 components) | 0.124 | 0.222 | 0.553 |
| GMM Diagonal Cov (16 components) | 0.130 | 0.225 | 0.630 |
| GMM Tied Cov (32 components) | 0.135 | 0.257 | 0.615 |

No PCA

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| GMM Full Cov (8 components) | 0.124 | 0.222 | 0.553 |
| GMM Diagonal Cov (16 components) | 0.130 | 0.225 | 0.630 |
| GMM Tied Cov (32 components) | 0.135 | 0.257 | 0.615 |

PCA m=7

# Scores calibration

The calibration is to be done of best models:

- Logistic regression ($\lambda = 10{-}5$ , $\pi T = 0.5$)
- Linear svm ($c=10^{-2}$, unbalanced)
- GMM full covariance (8 components)
- MVG tied covariance

So far, our consideration has been minDCF which is the empirical bayes cost we would have to pay if optimal decisions are made by using scores provided. But the real cost depends on the viability of the threshold employed to perform class assignment. A good threshold is needed  in order to compare log-likelihoods ratios. Optimal decision can also be made by recalibrating the scores such that optimal threshold turns into a theoretical threshold.

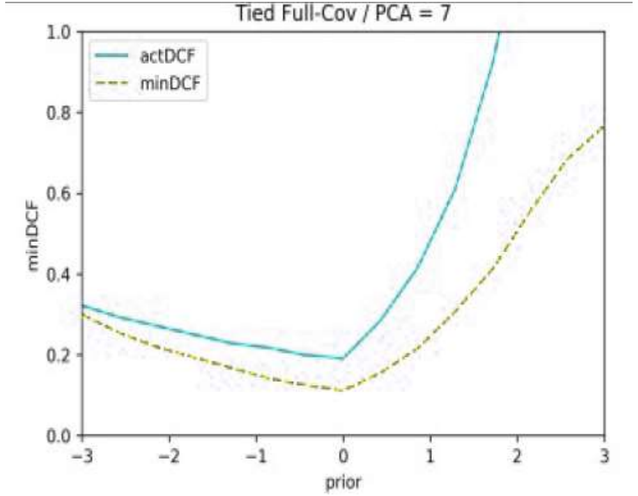
$$t = -\log \frac{\tilde{\pi}}{1-\tilde{\pi}}$$

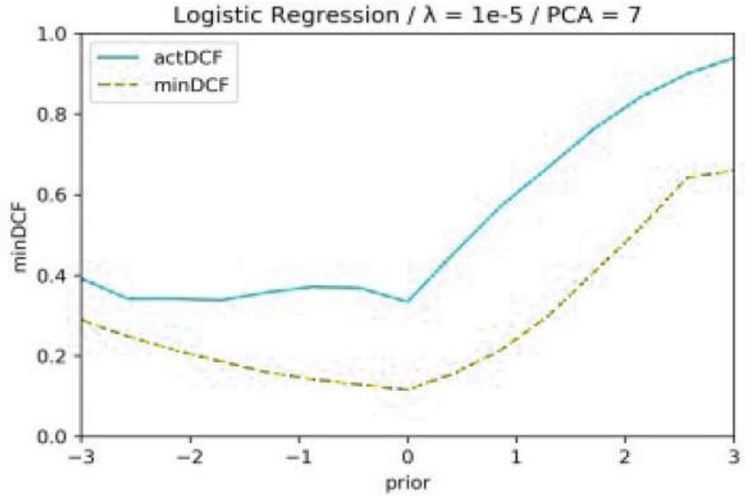
# SCORES Calibration

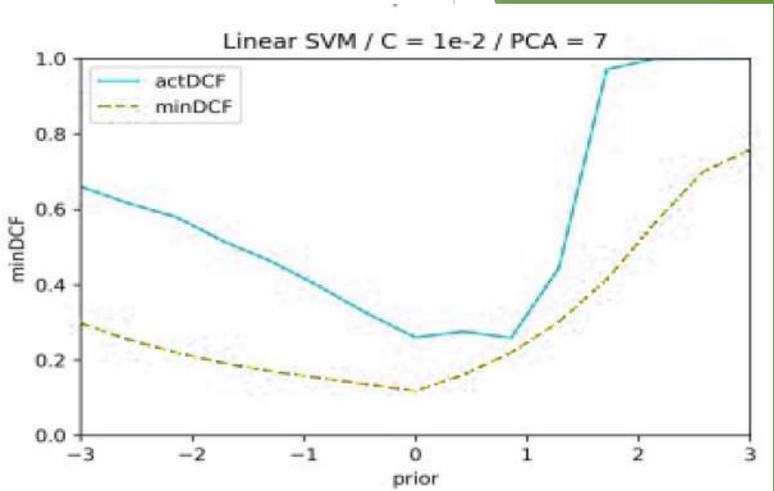
From the plots, it is evident that minDCF differs from actDCF by quite a lot. So we have to use a technique of score calibration in order to compute transformation function to map uncalibrated score into calibrated ones. $f(s) = \alpha s + \beta$ => Here we are assuming that the function is linear in s. For the value of $\pi$, we will select 0.5

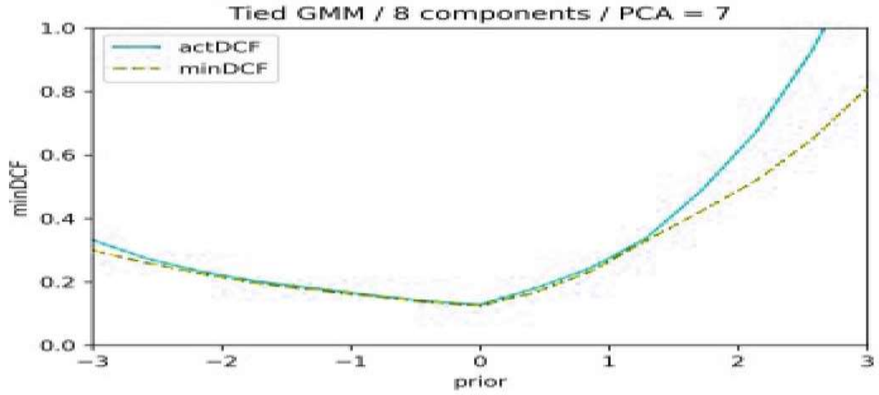
From the result graphs the mapping, as it can be seen, is successful over wide range of applications.

# Plot (uncalibrated classifier)

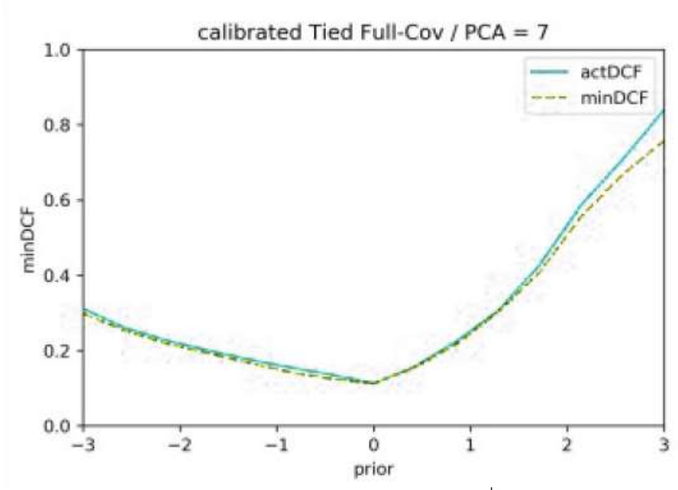


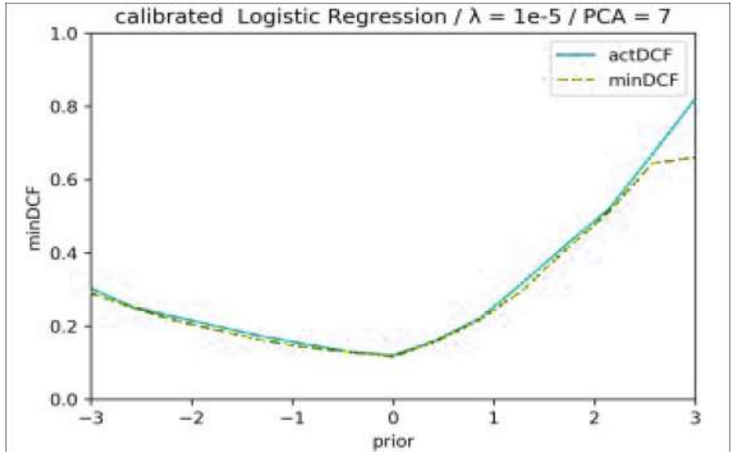



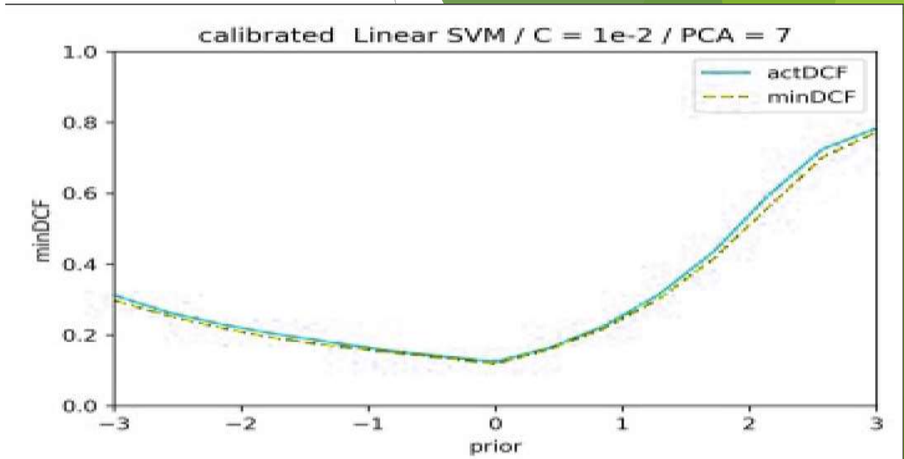



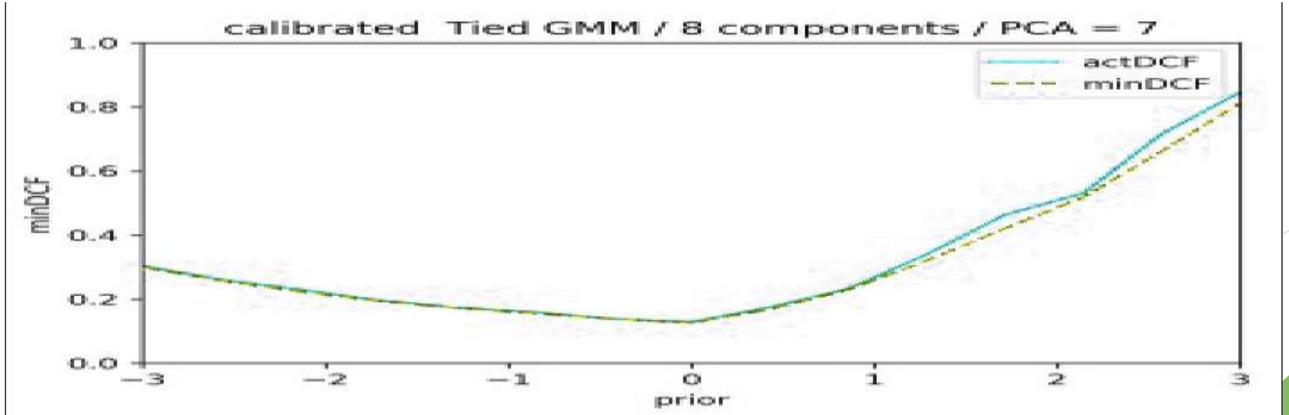



# Plot (Calibrated Classifiers)



# SCORE CALIBRATION(NO PCA)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Tied Full-Covariance | 0.112 | 0.224 | 0.573 |
| LogisticReg($10^{-5}$, 0.5) | 0.126 | 0.238 | 0.571 |
| SVM(C = $10^{-2}$ ) | 0.117 | 0.223 | 0.580 |
| GMM Full Covariance(8) | 0.124 | 0.222 | 0.553 |

minDCF

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Tied Full-Covariance | 0.113 | 0.228 | 0.616 |
| LogisticReg($10^{-5}$, 0.5) | 0.121 | 0.225 | 0.541 |
| SVM(C = $10^{-2}$ ) | 0.124 | 0.232 | 0.589 |
| GMM Full Covariance(8) | 0.128 | 0.235 | 0.579 |

Actual DCF

# SCORE CALIBRATION(PCA m=7)

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Tied Full-Covariance | 0.112 | 0.223 | 0.572 |
| LogisticReg($10^{-5}$, 0.5) | 0.116 | 0.218 | 0.542 |
| SVM(C = $10^{-2}$ ) | 0.117 | 0.223 | 0.580 |
| GMM Full Covariance(8) | 0.124 | 0.222 | 0.553 |

minDCF

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Tied Full-Covariance | 0.114 | 0.228 | 0.616 |
| LogisticReg($10^{-5}$, 0.5) | 0.115 | 0.225 | 0.533 |
| SVM(C = $10^{-2}$ ) | 0.126 | 0.233 | 0.588 |
| GMM Full Covariance(8) | 0.127 | 0.231 | 0.583 |

Actual DCF

# Evaluation

- Finally, we will analyze the performances of selected models over the test set. Pca m= proved to be a good strategy so only this approach will be considered.

- We can see the results are consistent with the analysis done over training set. Also it can be seen that logistic regression is the best model. To compare models, an roc curve can be drawn and the best model will have the highest area under curve.

minDCF

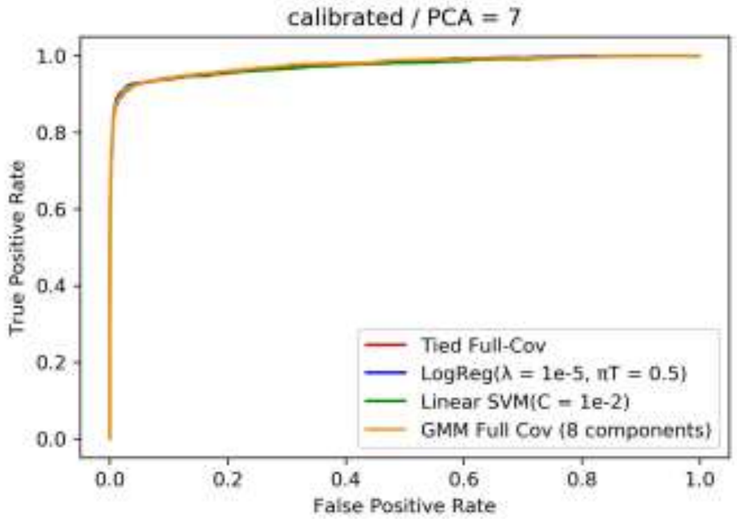
| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Tied Full-Covariance | 0.112 | 0.215 | 0.580 |
| LogisticReg($10^{-5}$ , 0.5) | 0.109 | 0.210 | 0.551 |
| SVM(C = $10^{-2}$ ) | 0.116 | 0.217 | 0.580 |
| GMM Full Covariance( 8) | 0.113 | 0.218 | 0.518 |

actDCF

| π | 0.5 | 0.1 | 0.9 |
|---|---|---|---|
| Tied Full-Covariance | 0.114 | 0.224 | 0.600 |
| LogisticReg($10^{-5}$ , 0.5) | 0.113 | 0.223 | 0.564 |
| SVM(C = $10^{-2}$ ) | 0.120 | 0.228 | 0.588 |
| GMM Full Covariance( 8) | 0.115 | 0.220 | 0.539 |

# ROC

- The curve looks nearly identical for all classifiers. The high slope on the left implies that models can correctly classify among true and false pulsar.



# Final thoughts

- Linear classifers have performed visibly better than quadratic ones. The performance over evaluation set was consistent with the performance over training set. This validates the choices that were made during the analysis