Machine learning and pattern recognition project

Giuseppe Scarso, s308807

1 Introduction

The assigned project task consists of a binary classification problem. The goal is to perform fingerprint spoofing detection, i.e. to identify genuine vs fake fingerprint images, the task is divided into some steps, in this document are reported all the steps represented by the laboratories.

2 Lab2

The main objective for this lab was to analyze the data coming from the dataset, which consists of labeled samples corresponding to the genuine (True, label 1) class and the fake (False, label 0) class. Once imported the file with all the samples the first step was to plot the histogram and the pair-wise scatter plots. Here there are all pictures:

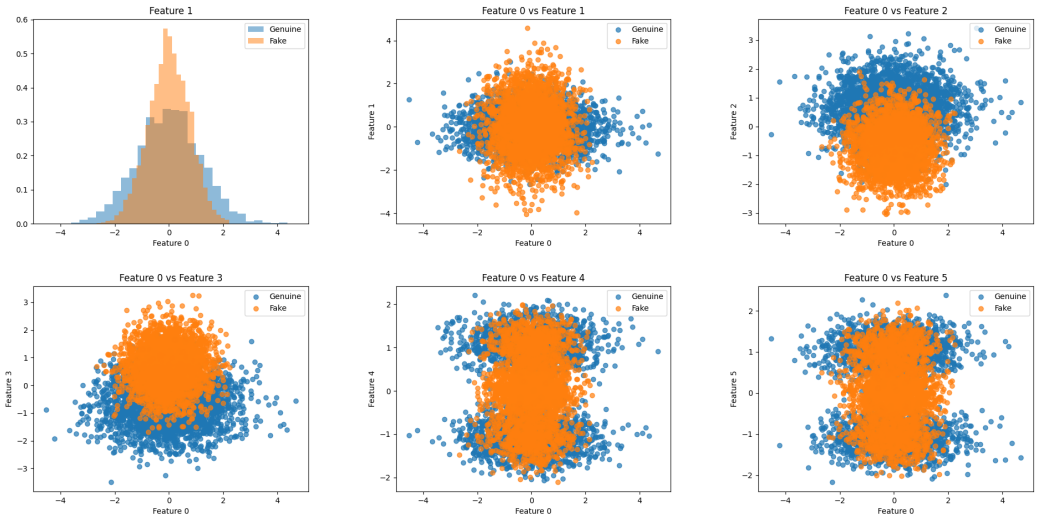

Figure 1: Histogram of the first feature (Feature 0) and the related scatter plots

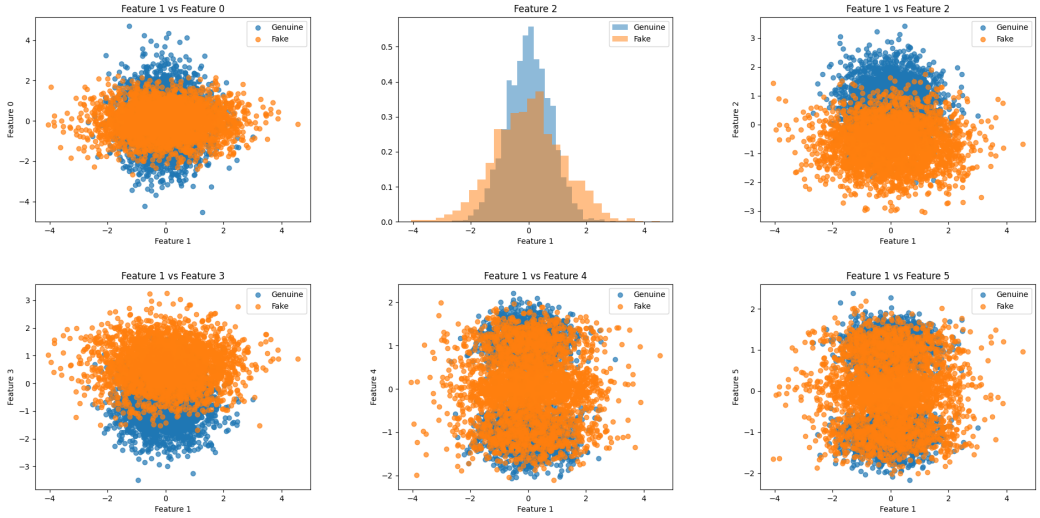

Figure 2: Histogram of the second feature (Feature 1) and the related scatter plots

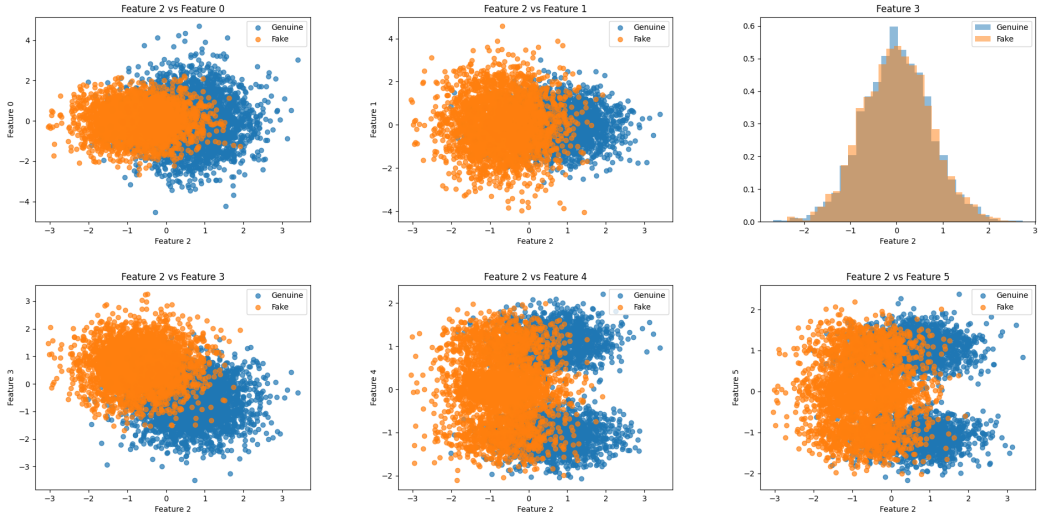

Figure 3: Histogram of the third feature (Feature 2) and the related scatter plots

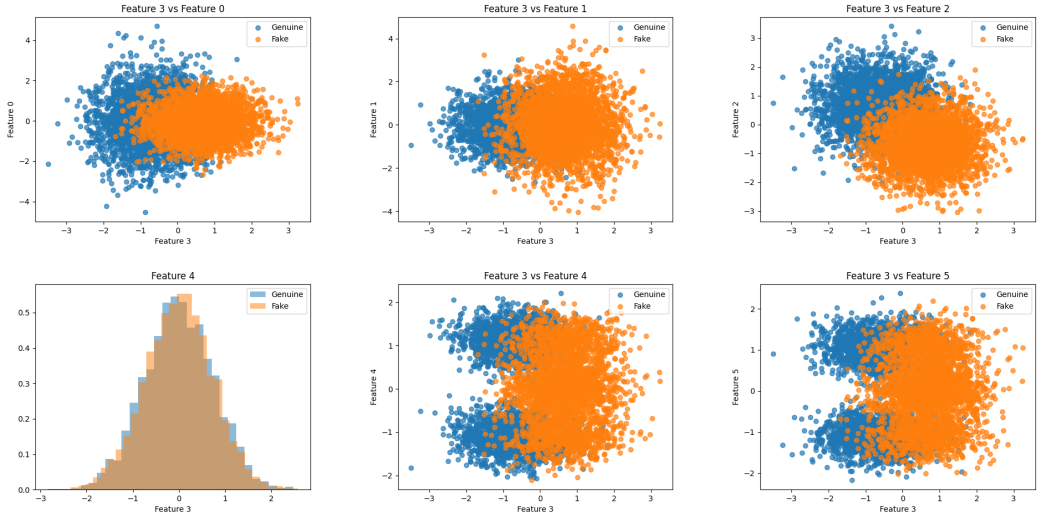

Figure 4: Histogram of the fourth feature (Feature 3) and the related scatter plots

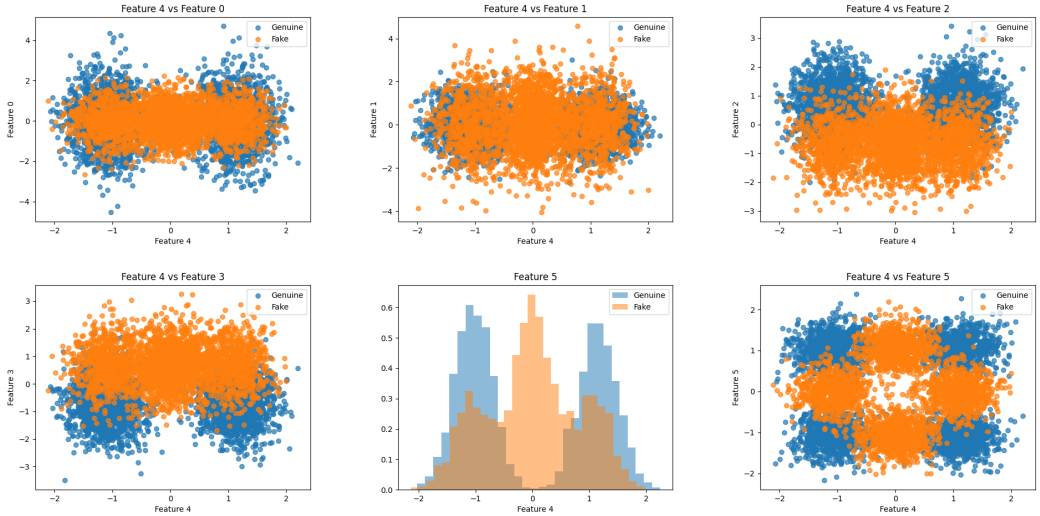

Figure 5: Histogram of the fifth feature (Feature 4) and the related scatter plots

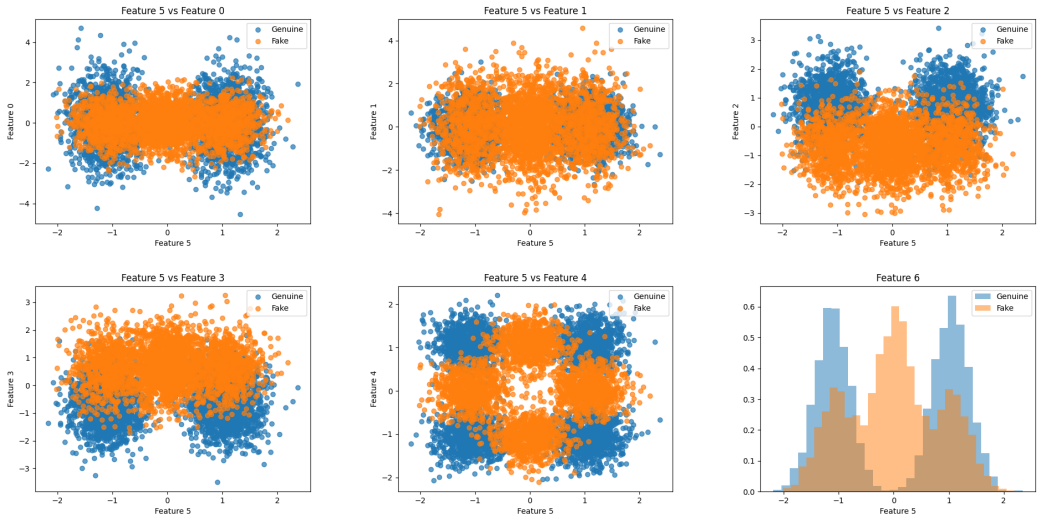

Figure 6: Histogram of the sixth feature (Feature 5) and the related scatter plots

We can compute alse the mean of each feature considering the two class, for the genuine and fake classes the mean of the 6 features is represented by the following vectors (one row for each feature):

```
Mean for genuine class:
[[ 5.44547838e-04]
 [-8.52437392e-03]
 [ 6.65237846e-01]
 [-6.64195349e-01]
 [-4.17251858e-02]
 [ 2.39384879e-02]]
```

```
Mean for fake class:
   [[ 0.00287744]
    [ 0.01869316]
    [-0.68094016]
    [ 0.6708362 ]
    [ 0.02795697]
    [-0.0058274 ]]
```

Feature 0 and 1 analysis

We can notice that the data appear to be zero-centered (in fact the mean of the two features is almost zero for both classes), and as shown in the histogram the two class overlaps, for the first feature the variance is higher for the genuine class resulting in a more spread out distribution, for the second feature it is the opposite, both histograms contain only one peak (Unimodal)

Feature 2 and 3 analysis

In this case the data is not exactly zero-centered and the distributions don't overlap completely, for the third and fourth feature the variance is almost the same and in both histogram there is one peak for the genuine class and another peak for the fake class

Feature 4 and 5 analysis

For the last features the situation is a bit different, they overlap a bit and the histograms present for both features two peaks for the genuine class and one for the fake class, looking the scatter plot we can see 4 cluster for both classes

3 Lab 3

First of all applying PCA to the data, the resulting histograms for the six feature are the following:

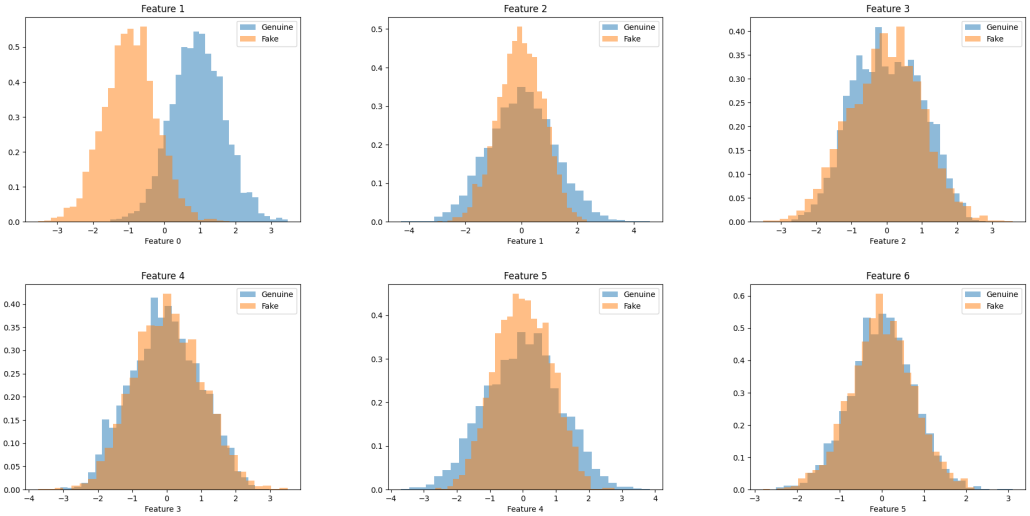

Figure 7: Histograms of the six features after PCA

Only for the first feature the two class does not overlap.
After computing LDA the result is the following:

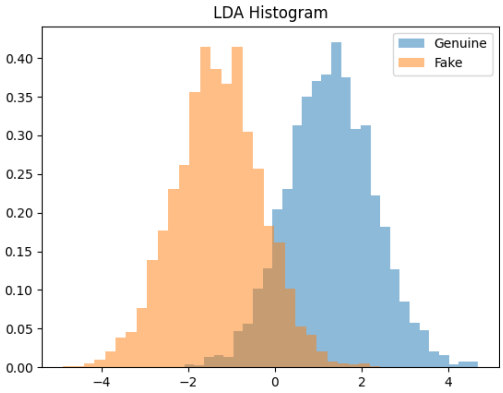

Figure 8: Histogram of the LDA direction

Apparently LDA, compared to the histograms generated in lab2, found a direction with little class overlap, in fact during lab 2 almost all the feature had a significant overlap between classes

Dividing the dataset in two parts, one for the training (2/3) and one for validation (1/3), we can calculate LDA and estimate a threshold for classification as the average of the mean of the two classes, in this case the threshold was: -0.0185 providing an error rate of 9.3% in the validation set

It is possible to adjust a bit the threshold using only the training data resulting in a more precise classifier also for the validation set, in fact with a threshold of -0.0119 the error rate is 9.25%

Finally, trying pre-processing the dataset with PCA, the best number of dimension is m = 2 with an error rate of 9.25% in the validation set

4 Lab 4

The objective of the laboratory is still to preform some preliminary qualitative analysis, computing first of all the ML estimates for each feature and class. Here there are all the values for mean(M) and variance(V)

	Genuine Class	Fake Class
Feature 1	M = 0.00054, C = 1.43023, LL = -4809	M = 0.00287, C = 0.56958, LL = -3401
Feature 2	M = -0.00852, C = 0.57827, LL = -3446	M = 0.01869, C = 1.42086, LL = -4767
Feature 3	M = 0.66523, C = 0.5489, LL = -3368	M = -0.68094, C = 0.54997, LL = -3348
Feature 4	M = -0.66419, C = 0.55334, LL = -3380	M = 0.6708, C = 0.53604, LL = -3310
Feature 5	M = -0.04172, C = 1.31776, LL = -4686	M = 0.02795, C = 0.6800, LL = -3666
Feature 6	M = 0.02393, C = 1.28702, LL = -4650	M = -0.0058, C = 0.70503, LL = -3720

Once knowing the ML estimates it is possible to compute the uni-variate Gaussian models for the different features, with the following results:

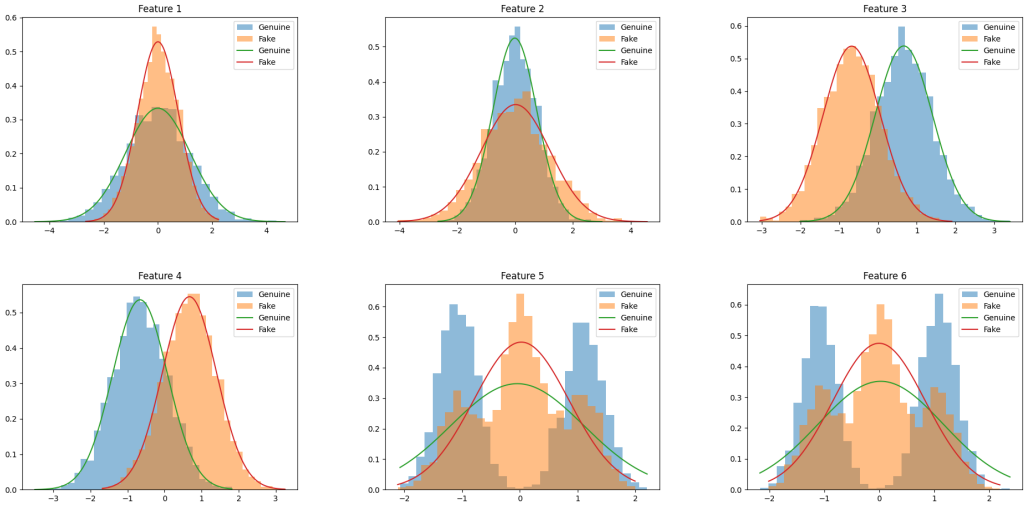

Figure 9: Histograms of the six features after fitting the uni-variate Gaussian models to the different features

We can notice that for the first 4 features the distribution fits very good the histogram, instead for the last 2 features the distributions don't fit the genuine class at all, because of the two peaks

5 Lab 5

The objective of this lab is to apply the Gaussian MVG model and its variant to the data.

The first step for each variant is to calculate the ML estimates, then to compute the log likelihood and compare it to a threshold, since we assume a uniform prior the threshold is 0.

Analyzing the correlation matrix it is possible to notice that the features are not strong correlated since the values are very close to 0, this could affect in a positive way the Naive Bayes because it operates under the assumption that the features in the data are independent of each other.

To analyze if some feature (for example the last 2) affect in a bad way the classification, it's possible to try repeating the classification discarding some feature. As the next results show, discarding the last feature does not improve the performance of the classifier

Finally also PCA can be applied

5.1 MVG

The result obtained for this model is an error rate of 7%, apperntly this model performs better than the LDA classifier which had an error rate of 9.25%

- All feature: Error rate = 7%
- First 4 feature: Error rate = 8%

- First 2 feature: Error rate = 36.4%
- Only feature 3-4: Error rate = 9.4%
- PCA m = 1: Error rate = 9.2%
- PCA m = 2: Error rate = 8.8%
- PCA m = 3: Error rate = 8.8%
- PCA m = 4: Error rate = 8.1%
- PCA m = 5: Error rate = 7.1%

5.2 Tied Gaussian Model

Results for the Tied Gaussian Model

- All feature: Error rate = 9.3%
- First 4 feature: Error rate = 9.5%
- First 2 feature: Error rate = 49.5%
- Only feature 3-4: Error rate = 9.4%
- PCA m = 1: Error rate = 9.3%
- PCA m = 2: Error rate = 9.2%
- PCA m = 3: Error rate = 9.2%
- PCA m = 4: Error rate = 9.2%
- PCA m = 5: Error rate = 9.3%

5.3 Naive Bayes

Results for the Naive Bayes

- All feature: Error rate = 7.2%
- First 4 feature: Error rate = 7.6%
- First 2 feature: Error rate = 36.3%
- Only feature 3-4: Error rate = 9.4%
- PCA m = 1: Error rate = 9.2%
- PCA m = 2: Error rate = 8.8%
- PCA m = 3: Error rate = 9.0%
- PCA m = 4: Error rate = 8.8%
- PCA m = 5: Error rate = 8.8%

In conclusion we can say that PCA is not effective for the dataset considered, except for the MVG by reducing to 5 dimension but still the error rate is higher or the tied that improves a bit the accuracy, however the best result are achieved by the standard MVG model

6 Lab 6

Not requested for the project

7 Lab 7

Confusion matrices are an useful tool to better understand the performance of the model, considering the following set of priors, false negative costs and false positive costs it's possible to compute the effective priors obtaining these results:

- (0.5, 1.0, 1.0): 0.5
- (0.9, 1.0, 1.0): 0.9
- (0.1, 1.0, 1.0): 0.1
- (0.5, 1.0, 9.0): 0.1
- (0.5, 9.0, 1.0): 0.9

The last two application shows that it's enough to use a different prior to obtain the same effective prior, so let's consider only the first three applications, in general the calibration error does not become greater than 15% (of the minDCF) but some models are more well calibrated than others

Prior 0.5

- MVG: DCF = 0.140, minDCF = 0.130
- Tied: DCF = 0.186, minDCF = 0.181
- Naive Bayes: DCF = 0.144, minDCF = 0.131

The best model for this application according to the minDCF, considering also the various PCA, it's the standard MVG

Prior 0.9

- MVG: DCF = 0.4, minDCF = 0.342
- Tied: DCF = 0.463, minDCF = 0.442
- Naive Bayes: DCF = 0.389, minDCF = 0.351

The best model for this application according to the minDCF, considering also the various PCA, it's the standard MVG

Prior 0.1

- MVG: DCF = 0.305, minDCF = 0.263
- Tied: DCF = 0.406, minDCF = 0.363
- Naive Bayes: DCF = 0.302, minDCF = 0.257

The best model for this application according to the minDCF, considering also the various PCA, it's the standard Naive Bayes

Considering only the prior 0.1, applying PCA and comparing the different models, the best results are: for MVG m = 5 (minDCF = 0.274, actDCF = 0.304), for Tied m = 4 (minDCF = 0.361, actDCF = 0.403), for Naive m = 5 (minDCF = 0.354, actDCF = 0.393)

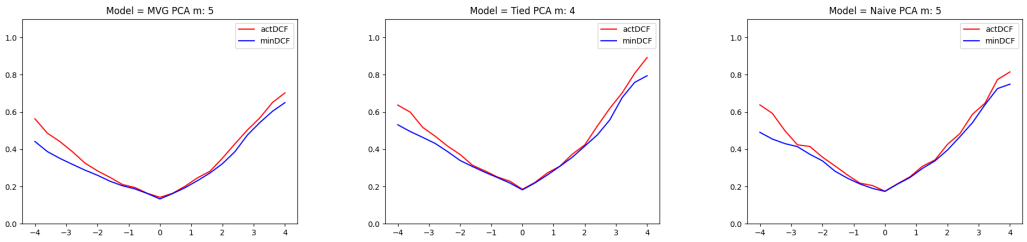

Figure 10: minDCF vs actDCF plot for best PCA for the models with prior 0.1

Models appear to be well calibrated since the calibration it's not high (about 11%)

8 Lab 8

Applying the Logistic Regression model to the data, starting from the standard non-weighted version, trying different values for the regularization term the result is the following:

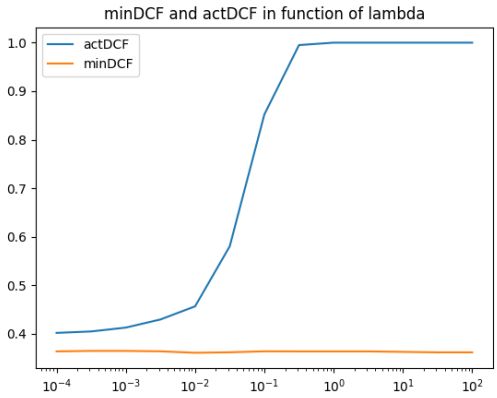

Figure 11: minDCF and actDCF vs λ

The more λ grow the more the actDCF goes away from the minDCF, so in this case it's not good to increase the regularization, regularization seems ineffective, and actually degrades actualDCF since the regularized models tend to lose the probabilistic interpretation of the scores.

If we consider less sample the regularization term is more effective, in fact there is a point in which the actDCF goes next to the minDCF:

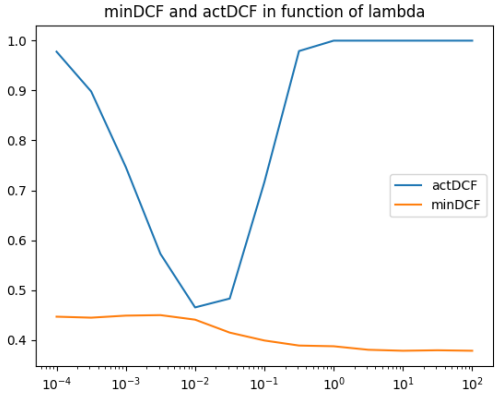

Figure 12: minDCF and actDCF vs λ with only 50 samples

Repeating the procedure with prior-weighted version and quadratic version the results are the following:

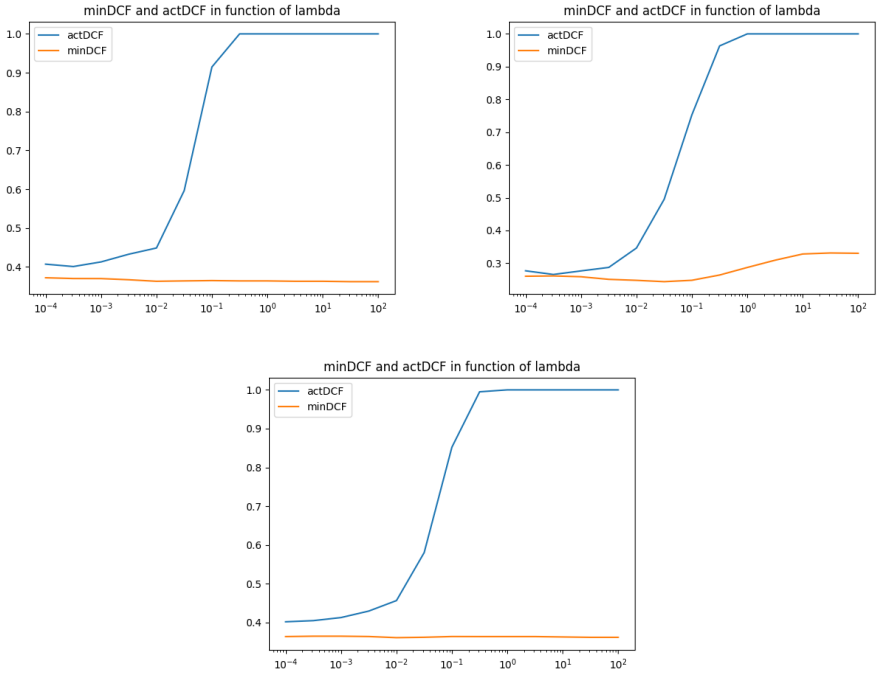

Figure 13: On the left the weighted, on the right the quadratic, on the bottom the linear version with the centered dataset

In any of these cases, regularization does not have a good impact, the model that minimize the minDCF is the quadratic LogReg with $\lambda = 0.0316$ and a minDCF of 0.243 even considering the other models studied in the previous labs.

9 Lab 9

Linear

Training the linear model, trying several values for C the result is that for bigger value (weaker regularization) the actDCF goes closer to the minDCF

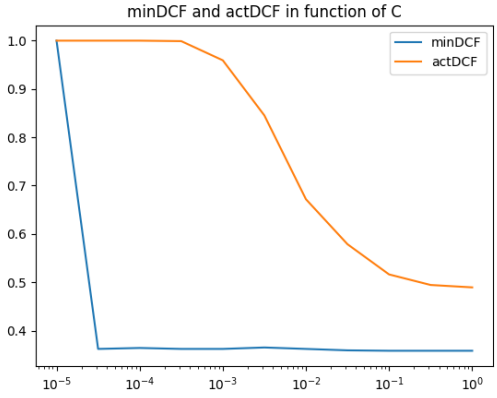

Figure 14: minDCF and actDCF vs C for linear SVM

The linear SVM achieve in the best case an error rate of 9.0%, which is more that logistic regression or MVG, also the minDCF (0.3582) is not as good as the other models

Polynomial

Doing the same for the polynomial version the result seems almost identical looking the plot

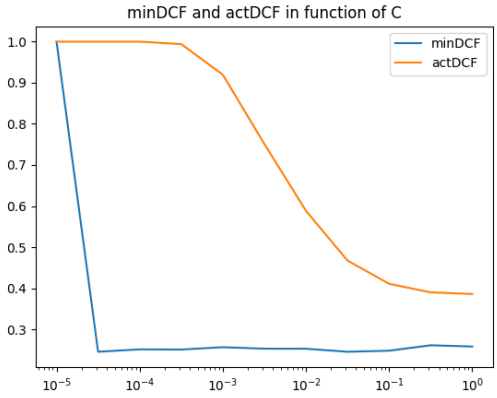

Figure 15: minDCF and actDCF vs C for polynomial kernel SVM

but in this case the error rate is similar to what obtained in the logistic regression (about 5.9%), even the minDCF is similar (0.2480) but the actDCF is 0.4109 which means that the model is not well calibrated

RBF

For RBF since we need to optimize two terms, a good method is to use a grid search considering different values for γ and C

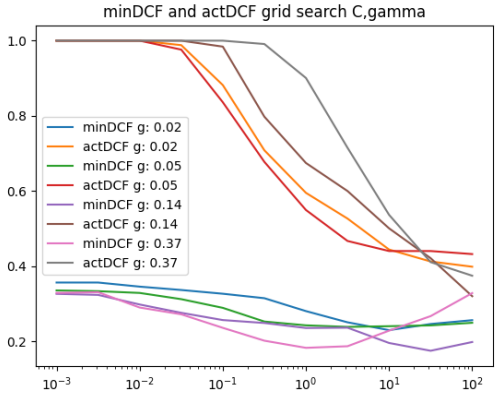

Figure 16: minDCF and actDCF vs γ and C

There are some combination of values that perform incredibly good, obtaining a 4.2% error rate and a minDCF of 0.187 even if the actDCF is a bit high compared to the minDCF (0.715)

10 Lab 10

The result obtained training the GMM models are very good, if we have to choose the model which gives us the best minDCF it is the diagonal one with 8 components achieving an error rate of 3.45%, a minDCF of 0.1463 and an actDCF of 0.1809

To summarize,the best model obtained until now (excluding MVG) are:

- Logistic Regression: Quadratic
- SVM: RBF
- GMM: Diagonal with 8 components

These are the Bayes error plot over a wide range of operating points (log-odds ranging from -4 to +4)

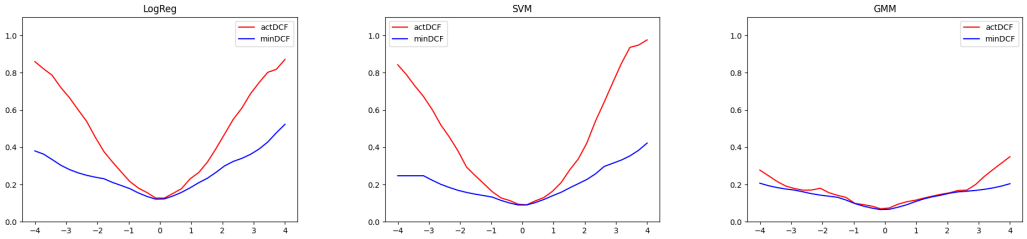

Figure 17: minDCF vs actDCF for various operating points

The only model that is well calibrated for most of the operating range is the GMM, the other two present an actDCF which is very far from the minDCF in several points.

11 Lab 11

Calibration

Now let's compute a calibration transformation of the scores for each model with the k-fold approach, using the validation set from the previous labs. For the prior 0.1 the results are the following:

Not calibrated

- LogReg: actDCF (0.1) = 0.495
- SVM: actDCF (0.1) = 0.422
- GMM: actDCF (0.1) = 0.181

Calibrated

- LogReg: actDCF (0.1) = 0.261
- SVM: actDCF (0.1) = 0.189
- GMM: actDCF (0.1) = 0.168

The best model is the GMM

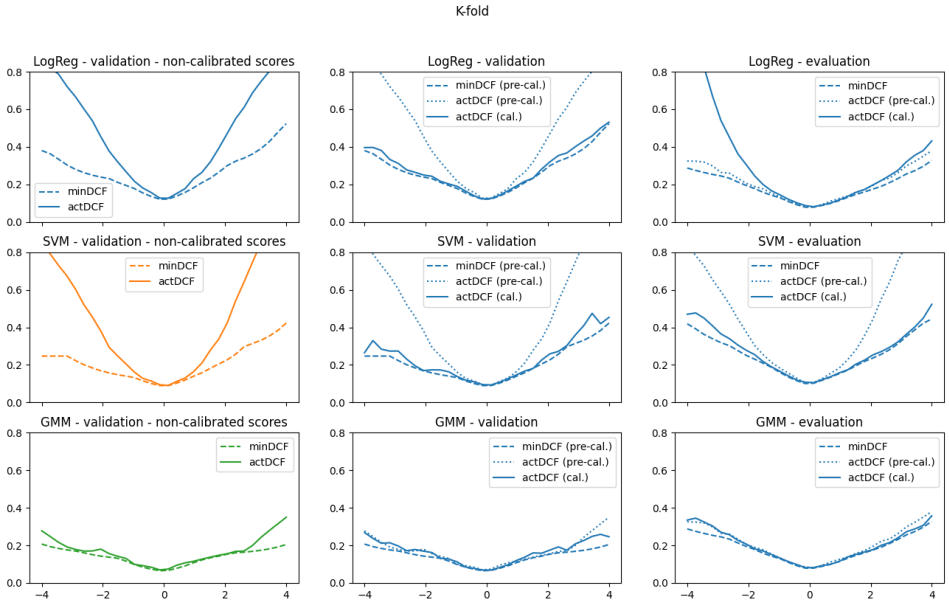

Figure 18: Comparison between the three models

Although the single models performed very good, the best results (iterating also over different priors) are achieved by the fusion of the models, in fact it obtains an actDCF of 0.1566 and minDCF of 0.156 (very well calibrated) on the validation data

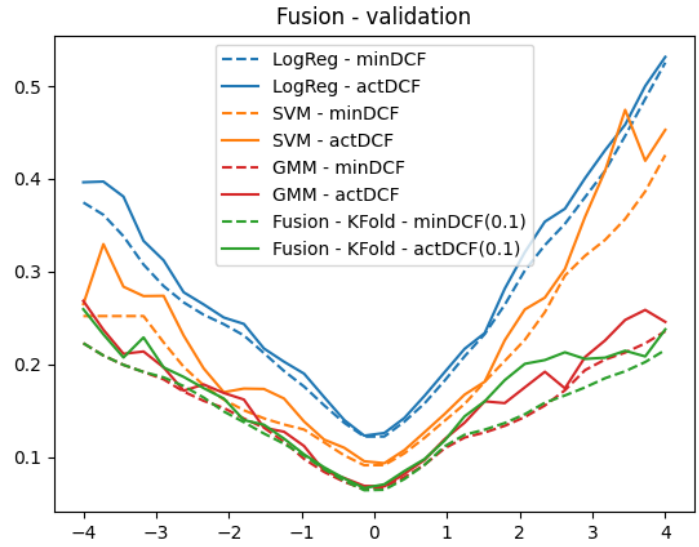

Figure 19: Fusion validation

Evaluation

	minDCF	actDCF no cal	actDCF cal	Error rate
LogReg	0.203	0.221	0.337	3.9%
SVM	0.262	0.403	0.289	5%
GMM	0.203	0.221	0.221	3.8%

The evaluation for the fusion model, considering the calibration , achieve:
minDCF = 0.198 - actDCF = 0.207 - Error rate = 3.8%

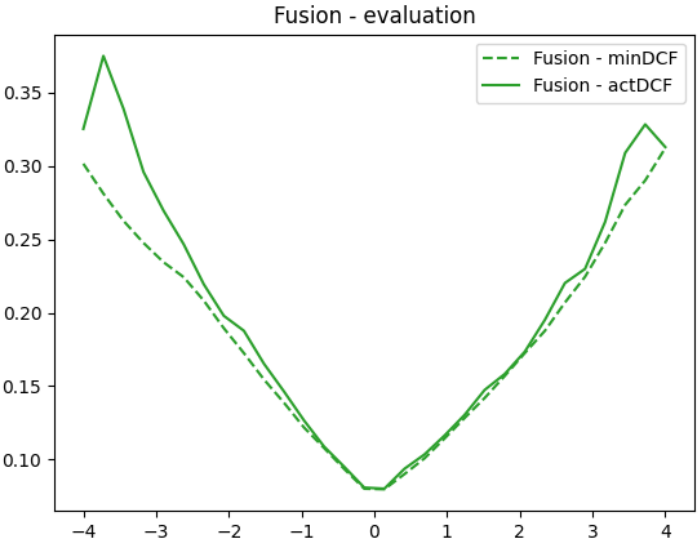

Figure 20: Fusion evaluation

In conclusion we can say that the GMM model and the fusion achieve the best results for this task.

Since the result are similar it may be convenient to use only the GMM instead of the fusion in order to reduce the computational cost and increase the speed.