REPORT: BANKNOTE AUTHENTICATION

Giuseppe Salvi, s287583
26/01/2022

1. DESCRIPTION OF THE PROBLEM AND DATASET ANALYSIS

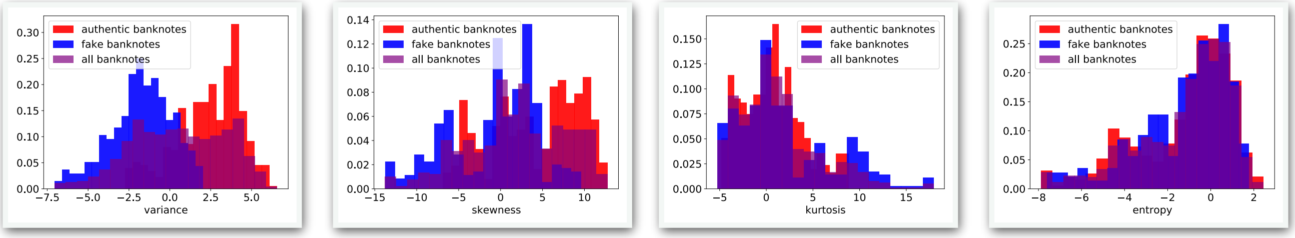
The task requires verifying whether a banknote is authentic or not. The dataset is taken from the UCI Machine Learning repository and contains 675 samples for the training dataset (Train.txt) and 697 for the testing dataset (Test.txt). There is also a second version of the dataset that corresponds to the original data degraded by noise (TrainH.txt, TestH.txt).

There are 5 attributes: 4 are the features and 1 is the target attribute, which contains the value 0 for authentic banknotes, 1 for fake banknotes. The features are continuous and correspond to variance, skewness, kurtosis, and entropy.

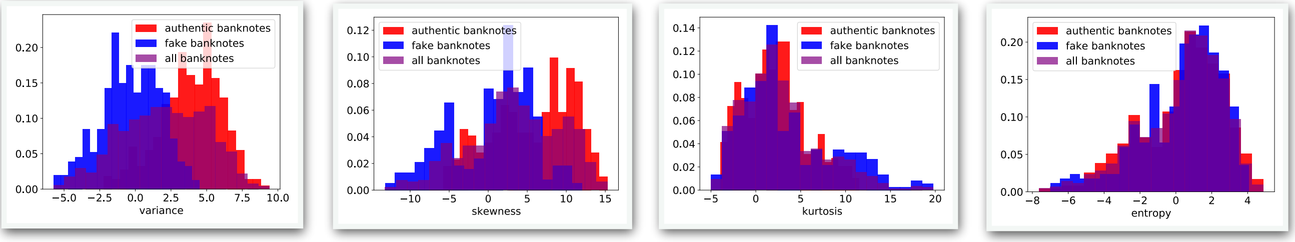
In the training set, 374 samples correspond to authentic banknotes, while 301 correspond to fake ones. In the evaluation set, we have 388 authentic and 309 fake. The same happens for the degraded version of the dataset.

So the dataset contains a balanced ratio of both classes (55,4% - 44,6 % for train dataset, 55,7 % - 44,3 % for test dataset). Given the small amount of data, a k-fold cross-validation technique could be a good idea to estimate more reliable parameters for our models.

Histograms of features of Train dataset



Histograms of features of Train dataset degraded by noise



We can see that some features have a Gaussian-like distribution more than others and the two classes are separable; this is true, for example, for the variance feature.

1.1 CORRELATION ANALYSIS

By looking at the heat map showing the Pearson correlation coefficient (Figure1 and Figure 2), we can see that there is no particular correlation between the features. The highest value is between kurtosis and entropy (feature 2 and feature 3) and has a value of 0.33 for the original case, and 0.313 for the dataset degraded with noise. This, and also the fact that we have few features, suggests that dimensionality reduction techniques, such as PCA, could not be beneficial for this task. For this reason, I decide not to use them.

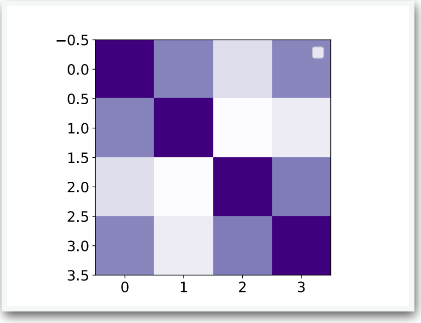


Figure 1: Correlation matrix heat-map Training dataset

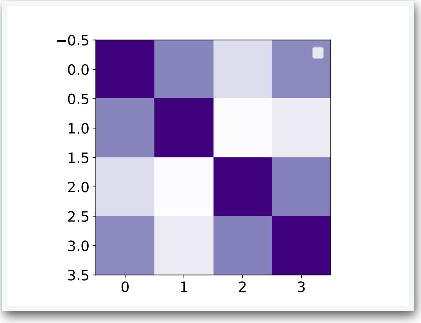


Figure 2: Correlation matrix heat-map Training dataset degraded with noise

	0	1	2	3
0	1.0 (1.0)	.28 (.27)	-.38 (-.36)	.26 (.24)
1	.28 (.27)	1.0 (1.0)	-.79 (-.77)	-.55 (-.52)
2	-.38 (-.36)	-.79 (-.77)	1.0 (1.0)	.33 (.31)
3	.26 (.24)	-.55 (-.52)	.33 (.31)	1.0 (1.0)

Figure 3: Correlation matrix: train, (trainH)

2. CLASSIFIERS

Given the small size of the training dataset, I decided to adopt the k-fold cross-validation methodology (with k = 5), as well as the single-fold one (with a 2/3 split), to see if we have an improvement.

The main application is a uniform prior one $(\tilde{\pi}, C_{fp}, C_{fn}) = (0.5, 1, 1)$, and I considered also two unbalanced applications $(\tilde{\pi}, C_{fp}, C_{fn}) = (0.1, 1, 1)$, $(\tilde{\pi}, C_{fp}, C_{fn}) = (0.9, 1, 1)$, with bias towards authentic or fake banknotes.

To choose the most promising approach, I measure performance in terms of normalized minimum detection cost, which measures the cost we would pay if we made optimal decisions for the test set (validation set in our case) using the recognizer scores.

2.1 GAUSSIAN CLASSIFIERS

I start analyzing the results for the normal dataset. With the single fold protocol, I obtain minDCF = 0 both for the Full Covariance model and the Tied Covariance model, for all three applications. This suggests that our model is theoretically perfect, or more probably we have too little data to perform validation and our dataset is overall very simple.

For this reason, it's important to base our considerations more on the result of the k-fold approach, since at least we use all the data at our disposal for training and validation. However, I continue to bring the results on the single-fold approach to compare them at the end in the experiments section and see if my decision will turn out to be correct or not.

The k-fold results confirm that the best models are the Full Covariance one and the Tied Covariance one, even though the first one performs slightly better.

Train	**Single Fold**			**K-Fold K=5**		
minDCF	**$\tilde{\pi}$ = 0.5**	**$\tilde{\pi}$ = 0.1**	**$\tilde{\pi}$ = 0.9**	**$\tilde{\pi}$ = 0.5**	**$\tilde{\pi}$ = 0.1**	**$\tilde{\pi}$ = 0.9**
Full	0.0000	0.0000	0.0000	0.0033	0.0033	0.0080
Diagonal	0.1994	0.6156	0.2403	0.2686	0.6808	0.3449
Tied	0.0000	0.0000	0.0000	0.0246	0.0299	0.0535

The best model is the Full Covariance one, also in the degraded dataset.
In this case, we can see a significant worsening in the results for the unbalanced applications, especially the one with $\tilde{\pi}$ = 0.1. In comparison with the results on the normal dataset, the best model is worse and more distant from the perfect result; this is reasonable since the dataset presents more noise. The results of k-fold are worse than the single-fold ones, this confirms that probably the amount of data for validation and model training is not enough for the single-fold setup.

TrainH	Single Fold			K-Fold K=5		
minDCF	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**
Full	0.0233	0.1458	0.0233	0.0541	0.2167	0.0668
Diagonal	0.2384	0.6042	0.3953	0.2963	0.6993	0.4657
Tied	0.0259	0.1458	0.0465	0.0568	0.2474	0.0668

Overall, the best candidate is currently the MVG model with Full Covariance matrices for both datasets and all three applications.

2.2 LOGISTIC REGRESSION

The Logistic Regression Model performs slightly worse than the MVG with Full Covariance matrices. Small values of lambda lead to the best results in both datasets. The main application has the best results, also in this case.

Train	Single Fold			K-Fold K=5		
minDCF	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**
λ = 0	0.0000	0.0000	0.0000	0.0133	0.0133	0.0732
λ = 1e-6	0.0000	0.0000	0.0000	0.0133	0.0133	0.0508
λ = 1e-3	0.0000	0.0000	0.0000	0.0200	0.0266	0.0428
λ = 1	0.0233	0.1708	0.0233	0.0535	0.3462	0.0535

TrainH	Single Fold			K-Fold K=5		
minDCF	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**
λ = 0	0.0363	0.1146	0.0465	0.0567	0.2458	0.0963
λ = 1e-6	0.0363	0.1146	0.0465	0.0567	0.2458	0.0963
λ = 1e-3	0.0441	0.1250	0.0465	0.0567	0.2458	0.0941
λ = 1	0.0492	0.2020	0.1325	0.0894	0.3462	0.2117

2.3 SVM

I start with analyzing linear SVM. I considered different set of values for the parameters K and C: K = 1, 10 ; C = 0.1, 1, 10.

For the normal dataset, the best results are achieved with K = 10 and C = 10 and are in line with the best results for the Logistic Regression model.
The same happens for the degraded dataset with different settings of the hyper-parameters: K = 10, C = 0.1. Also for the other applications, the effects are comparable with the results of LR, for both the datasets.

The Polynomial Kernel SVM with d=2 (Quadratic kernel) improves the performances in both datasets; in particular, in the degraded version, it becomes the best model overall, with minDCF = 0.0367. This is achieved with parameters K=1, C=0.1, c=2. It's interesting the fact that, for the first time, the parameters that lead to the optimal results for the balanced application create a harmful (minDCF >1) model for the second unbalanced application ($\tilde{\pi}$ = 0.9) and almost for the other one ($\tilde{\pi}$ = 0.1). This happens only in the normal dataset version.

In the normal dataset, the RBF kernel SVM with parameters K = 1, C = 10, gamma = g = 1 obtains minDCF = 0.0033, which is the same value of MVG with Full Covariance matrices, and it is the best so far.
In the degraded dataset, it is not as good as the previous SVM models.

Train	Single Fold			K-Fold K=5		
minDCF	**$\tilde{\pi}$ = 0.5**	**$\tilde{\pi}$ = 0.1**	**$\tilde{\pi}$ = 0.9**	**$\tilde{\pi}$ = 0.5**	**$\tilde{\pi}$ = 0.1**	**$\tilde{\pi}$ = 0.9**
Linear SVM K=10, C=10	0.0000	0.0000	0.0000	0.0199	0.0199	0.0963
Quadratic Kernel K=10, C=1 d=2, c=0	0.0000	0.0000	0.0000	0.0120	0.9809	1.1588
RBF Kernel K=1, C=10 g=1	0.0104	0.0104	0.0000	0.0033	0.2300	0.1444

TrainH	Single Fold			K-Fold K=5		
minDCF	**$\tilde{\pi}$ = 0.5**	**$\tilde{\pi}$ = 0.1**	**$\tilde{\pi}$ = 0.9**	**$\tilde{\pi}$ = 0.5**	**$\tilde{\pi}$ = 0.1**	**$\tilde{\pi}$ = 0.9**
Linear SVM K=10 C=0.1	0.0337	0.1323	0.0543	0.0567	0.2458	0.0914
Quadratic Kernel K=1, C=0.1 d=2, c=0	0.0233	0.1354	0.0233	0.0367	0.2590	0.1898
RBF Kernel K=1, C=1 g=1	0.0000	0.5417	0.0000	0.0666	0.4186	0.3508

2.4 GMM

The last model I consider is GMM for classification, in all three variants: full, diagonal, and tied.
It becomes the best model for both datasets.

In particular, in the normal dataset, the best results are achieved by both the full and tied version, which gets minDCF=0 with the number of Gaussians M = 2. Also the diagonal version with M=16 is better than the previous models but slightly worse than other GMM versions.
Notably, also the unbalanced applications achieve great results, in line with the balanced one.

Train	Single Fold			K-Fold K=5		
minDCF	**$\tilde{\pi}$ = 0.5**	**$\tilde{\pi}$ = 0.1**	**$\tilde{\pi}$ = 0.9**	**$\tilde{\pi}$ = 0.5**	**$\tilde{\pi}$ = 0.1**	**$\tilde{\pi}$ = 0.9**
Full M=2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Diag. M=16	0.0078	0.0104	0.0078	0.0027	0.0066	0.0027
Tied M=2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

In the degraded dataset, the best version is the diagonal one, with M = 16, despite it is again very similar to the results obtained by the other versions. This establishes the new best model for this dataset with minDCF = 0.030.

TrainH	Single Fold			K-Fold K=5		
minDCF	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**
Full M=4	0.0104	0.0104	0.0155	0.0314	0.1054	0.0401
Diag. M=16	0.0155	0.1395	0.0155	0.0301	0.1403	0.0566
Tied M=4	0.0104	0.0104	0.0155	0.0314	0.1054	0.0401

2.5 CHOICE OF MODELS

Summing up, for the normal dataset I got the best results with the GMM classifier (M=2), with svm with RBF kernel (K=1, C=10, g=1), and with the MVG model with Full Covariance Matrices.

For the degraded dataset I got the best results with the GMM classifier (Full: M=4, Diagonal: M=16) and with the svm with Quadratic Kernel (K=1, C=0.1, c=0, d=2).

Since the performance of those models are all very similar I decide to choose them all, and continue the experiments with them.

3. EXPERIMENTS

Below are the results for our classifiers using either the single-fold protocol (i.e the model trained over 2/3 of the data selected for model training) or the 5-fold cross validation protocol (i.e. the final model that was re-trained using the whole dataset).
In the cells, I put in the parenthesis the results achieved during training for comparison.

I can say that the choice of basing parameter tuning on k-fold results more than on single fold results revealed to be the best one, since the results obtained through the single fold are more different to the training ones, the ones of the k-fold are on the contrary very close. This is even more true for the degraded dataset.

GMM and MVG have the best performance for the normal dataset, while for the degraded dataset the best model is confirmed to be the GMM Diagonal with M=16 Gaussians.
The SVM model is the one that presents the biggest differences between training and testing results, in particular for the unbalanced applications, where minDCF is different for several decimals.

Train	2/3 of Data (Single Fold)			All Data (K-Fold)		
minDCF	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**	**π̃ = 0.5**	**π̃ = 0.1**	**π̃ = 0.9**
GMM Full M=2	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0000)
SVM RBF K=1, C=10, g=1	0.0000 (0.0104)	0.1068 (0.0104)	0.7527 (0.0000)	0.0239 (0.0033)	0.0485 (0.2300)	0.0567 (0.1444)
MVG Full	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0033)	0.0000 (0.0033)	0.0000 (0.0080)

TrainH	2/3 of Data (Single Fold)			All Data (K-Fold)		
minDCF	π̃ = 0.5	π̃ = 0.1	π̃ = 0.9	π̃ = 0.5	π̃ = 0.1	π̃ = 0.9
GMM Full M=4	0.0406 (0.0104)	0.0620 (0.0104)	0.1038 (0.0155)	0.0309 (0.0314)	0.0879 (0.1054)	0.0386 (0.0401)
GMM Dia. M=16	0.0549 (0.0155)	0.0911 (0.1395)	0.1005 (0.0155)	0.0297 (0.0301)	0.1375 (0.1403)	0.0704 (0.0566)
SVM Quad. K=1,C=0.1, c=0, d=2	0.4010 (0.0233)	0.1359 (0.1354)	0.0567 (0.0233)	0.0386 (0.0367)	0.9320 (0.2590)	0.0670 (0.1898)

3.1 RECALIBRATION OF THE SCORES

During my analysis, I always considered minDCF as a metric to compare the classifiers and select the best one: min DCF measures the cost that we would pay if we made optimal decisions for the evaluation set using the recognizer scores.
However, the cost that we actually pay depends on the goodness of the decisions we make using those scores.

If we compare the results of minDCF with those of actual DCF, we can see if our scores are calibrated, or if we would need to recalibrate them, in order to have our models' performance in line with the minDCF results.

Since the verbosity, I report here only the results for our selected models (training in parenthesis as before).
These results suggest that in order to optimize our models, we would need to recalibrate the scores.
In particular, the situation is more critical for svm models, especially in unbalanced applications.

The best model is still GMM with M=2 for the normal dataset, and GMM Diagonal with M=16 for the degraded dataset, even in this case without score recalibration.

Train	2/3 of Data (Single Fold)			All Data (K-Fold)		
actDCF	π̃ = 0.5	π̃ = 0.1	π̃ = 0.9	π̃ = 0.5	π̃ = 0.1	π̃ = 0.9
MVG Full	0.0258 (0.0233)	0.0000 (0.0000)	0.0258 (0.0233)	0.0258 (0.1395)	0.0232 (0.6429)	0.0258 (1.7838)
GMM Full M=2	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0000)	0.0000 (0.0000)	0.0077 (0.0000)
SVM RBF K=1, C=10, g=1	0.4847 (0.7313)	0.8997 (8.9406)	1.4174 (8.6328)	0.0065 (0.0486)	0.9320 (2.0114)	1.0000 (1.6209)

TrainH	2/3 of Data (Single Fold)			All Data (K-Fold)		
actDCF	π̃ = 0.5	π̃ = 0.1	π̃ = 0.9	π̃ = 0.5	π̃ = 0.1	π̃ = 0.9
GMM Full M=4	0.0459 (0.0104)	0.1397 (0.1250)	0.1577 (0.0310)	0.0342 (0.0353)	0.1106 (0.1453)	0.0678 (0.0401)
GMM Dia. M=16	0.0569 (0.0572)	0.1203 (0.1635)	0.1998 (0.0233)	0.0342 (0.0353)	0.1602 (0.1693)	0.0833 (0.0700)
SVM Quad. K=1,C=0.1, c=0, d=2	0.3242 (0.0233)	0.7152 (0.9688)	1.7976 (0.1395)	0.0711 (0.0642)	2.0606 (1.7838)	0.2191 (5.7496)

3.3 CONCLUSIONS

In conclusion, the result of my analysis is that many different models showed great performance for this task, but the best model overall is GMM, both for the normal version of the dataset and the degraded one (GMM Full M=2, GMM Diagonal M=16 respectively). To improve the model, score recalibration should be done, in order to select as threshold the optimal one, as seen in the minDCF results.