Machine Learning and Pattern Recognition: final report

Claudio Macaluso, s317149

January 24, 2025

1 Preliminary hypothesis: a visual inspection of data

!! add something about unimodal/multimodal !!
!! add something about correlation?? (2-3 seems correlated) !!

All the analyses conducted in this section consist of graphical evaluations made by observing the scatter plots and histograms of the data (Figure 1), therefore, they are not measurable and quantitative assessments, but only qualitative results extracted 'by eye'.

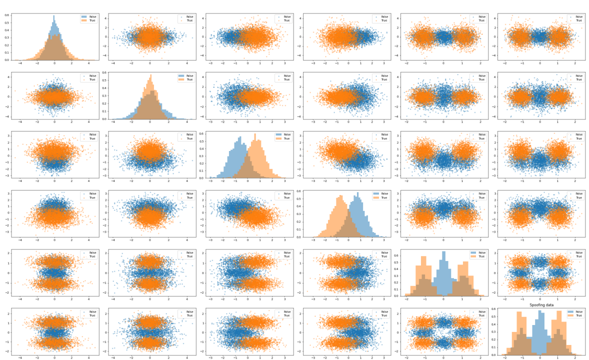

Figure 1: Spoofing dataset scatter matrix

Now we'll proceed with the analysis grouping features based on their common characteristics.

About the first two features, they are first of all unimodal: this can be useful if we use some model that assume a distribution that is inherintly unimodal (e.g. Gaussian models). Unfortunately, as we can see they show a significant overlap, and for these, their discriminative power can be less than other features. Despite this, these two features combined with the other ones can anyway contribute positively to the classification task. Another thing that have to be taken into account, is that the variances are different for the two classes and this can lead to a decrease in performance if we consider some model that assume the same variance for all the classes (e.g. Tied Covariance Gaussian Model).

Moving on the last two, they seem less overlapped compared to the previous ones, but in this case they're multimodal: multimodality can be challenging, specifically if we use some model that assume an underlying unimodal distribution (e.g. Gaussian Models). So an extrimely simple model in the specific case of these two features could perform poorly. So, taking into account mean and variance of the classes might not be significative anymore: mean doesn't represent any actual cluster of data points and, since variance represent spread around mean, we can't rely on the variance as a good parameter of spread of data neither. (review previous) For this two features, we can expect that models assuming unimodal distribution (e.g. Gaussian Models) perform poorly compared to others. Moreover, looking at the scatter plots, we can imagine that a quadratic and more complex separation surface can lead to better results than a linear one (e.g. Tied Covariance Gaussian Model, Logistic Regression or Support Vector Machine using standard kernel).

The third and the fourth features instead are unimodal and have a lower overlap, so they could probably perform well using models that assumes underlying unimodal distribution and be, on their own, the most discriminative features over all. Furthermore, since these features have similar class variance, it could be suitable to apply some models that make the assumption of same covariance among classes (e.g. Tied Covariance Gaussian Model).

Applying LDA moreover, we can notice that the found directions highly correspond to the directions of the third and the fourth features, since LDA tend to select linear combination of the features in a way that make classes more distinguishable in the projected data, confirming the hypothesis for which these two features lead to a more discriminative separation of classes.

```python
D, L, label_dict = load("project/data/trainData.txt")
W, _ = lda(D, L, m=1)
print(W)

# Output:
#[[-0.01063821]
# [ 0.0134172 ]
# [-0.96566604]
# [ 0.96774246]
# [ 0.0217633 ]
# [-0.03289391]]
```

Last thing about the third and fourth features, as we can see from scatter plots, the two are correlated, so using models assuming non correlated features (e.g. Naive Bayes Gaussian Models) can lead to a decrease in performance for these two features, so we will need to take care about this in the further analyses.

2 Dimensionality reduction techniques

Now, let's evaluate qualitatively with a visual inspection, as we've done in the previous section, the impact of dimensionality reduction in our dataset.

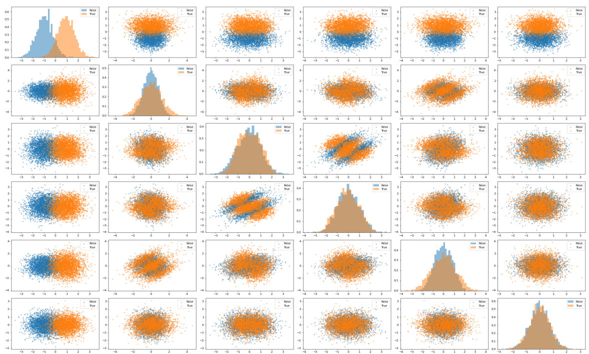

Figure 2: Spoofing dataset scatter matrix projected according to PCA directions

As we expected, since PCA is a non supervised dimensionality reduction technique (i.e. class agnostic), the found directions seem to not separate well, at least in some of the features, the classes to which the data belong. From a first analysis to the scatter plots of the PCA projected data, we can see that, except for the first component, the others seem to show high overlap between classes. Calculating the means and variances of classes in the projected space, we can see that the first component is the only one with a consistent difference in the mean of the classes. For the variances of the classes instead, we can notice that second and fifth features seem those with the biggest difference, so, disregarding the fact that visually they appear to have a high degree of overlapping, they can still be useful for the classification task. A possible benefit of PCA projection in this case could be that, from the histograms, we can notice that all the new features have unimodal distribution, so some models could improve their performance due to these new characteristics of the projected data. This last one is just a coincidence, since PCA objective isn't inherintly searching for a projection space in which features are unimodal, but still can be taken into account as an improvement in the new features space. Finally, as we expect, we can notice, calculating Pearson correlation matrix, that features are decorrelated in this new space (considering whole dataset, without class division), since projecting using PCA decorrelate features.

Clearly, a better separation can be achieved by applying LDA dimensionality reduction technique, that, unlike the previous one, is a supervised technique that aims to find the direction in which to project data for which the separation between classes is maximized.

Setting up a simple classification rule for 1-dimensional data that uses the mean of the means of the classes

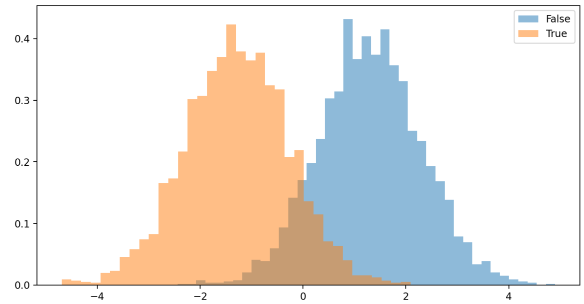

Figure 3: Spoofing dataset scatter matrix projected according to LDA directions

as a threshold to classify each sample, we can begin to check the impact of dimensionality reduction techniques on classification (Table 1), and use error rate to check the correctness of the model.

Table 1: Table showing the error rate using a decreasing number Principal Components and classifying with a simple LDA, mean based classifier

Dimensions	**Error Rate**	**Explained Variance**
any	9.30%	-
6	9.30%	100.00%
5	9.25%	90.68%
4	9.40%	74.29%
3	9.25%	57.69%
2	**9.05**%	40.94%
1	9.25%	23.99%

With this configuration, we can conclude that, looking just at error rate metrics as an indicator of the goodness of the dimensionality reduction hyperparameter m used in the context of this simple mean based classifier, the optimal value is 2. Using this configuration, retained variance is equal to 40.94%, and it is enough to use this 2-dimensional representation to capture the most significant patterns in the data while discarding less relevant variability.

3 Models evaluation

Now we can begin to analyse various models and their performance for our specific dataset. For each model, we'll write first of all, what do we expect, and then evaluate their effective performance using some chosen metrics.

3.1 Gaussian Models

3.1.1 Early analysis

Gaussian models are a group of probabilistic generative models that try to estimate the probability distribution of data starting from a representative dataset provided at training time. In particular, they assume that data are distributed following a Gaussian distribution, using as parameters the class-specific mean and variance. Decision boundaries are formed by comparing the likelihood of data points under these found Gaussian distributions, combined with prior probabilities to apply Bayes' theorem to find scores used in classification.

The strong assumption of Gaussian distributed data is a simplification used to deal with a well known distribution probability density function and all of its properties: due to this approximation, we expect these model to perform well in cases in which the underlying real data distribution is actually near to the Gaussian one. We can try to visualize the goodness of the hypothesis plotting, above data histograms, the Gaussian distribution obtained by using the mean and the variance of each feature for each class.

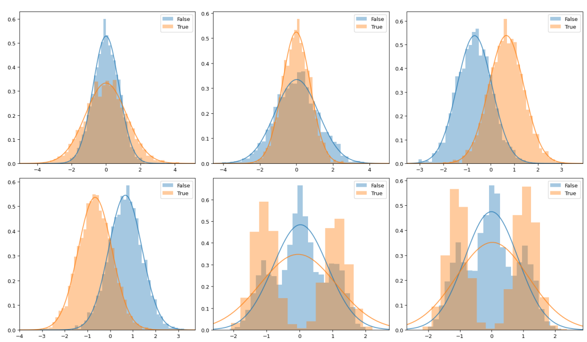

Figure 4: Per class features histograms of the spoofing dataset and their respective gaussian distribution, found using the mean and the variance of the per class features data

From the derived plot (Figure 4), we can see that for most of the features, at least visually, the Gaussian assumption is largely satisified, with the exception of the last two features: in fact, as we said in the data visual inspection section, those feature are multimodal and bad fits with models that assume an unimodal probability distribution (i.e. Gaussian models).

But, as we stated during the dimensionality reduction techniques analysis, we noticed that, applying PCA to data make them turn into unimodal distributions, so let's try it and see if the assumption is better satisified with projected data.

Indeed, as we can see in the plot (Figure 5) the features appear now all unimodal, and the class Gaussian curves seem to fit better in this case, even if, as we said, the price to pay is an higher degree of overlap of the features

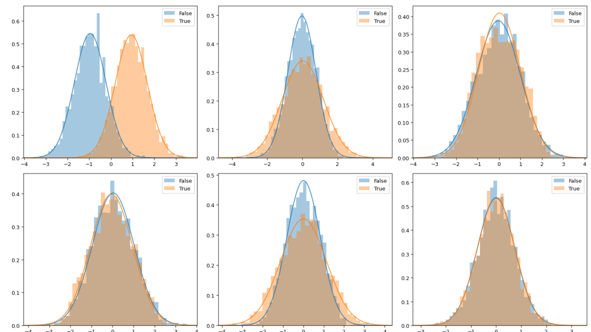

Figure 5: Per class features histograms of the PCA projected spoofing dataset and their respective gaussian distribution, found using the mean and the variance of the per class features data

overall: it's likely that, even if the Gaussian assumption is now satisfied, the PCA don't provide any relevant benefit, since, for the PCA projected features excluding the first one, classes are highly overlapped and, for this reason, quite not distinguishable each other. We will evaluate the consequences of all these reasoning quantitatively in further analyses to check if PCA is really convenient to apply or not and how it impact on Gaussian models.

!! think if adding same stuff about tied covariance is ok !!

3.1.2 Quantitative evaluation

Table 2: Table showing the error rate for Gaussian models

Model	**Error Rate**
Multivariate	**7.00%**
Naive	7.20%
Tied	9.30%
NaiveTied	9.30%
LDA	9.30%

As we can see from Table 2 by now, using uniform prior and uniform cost matrix, Multivariate Gaussian Models outperform the others. We can try to further analyse data and try to understand the reason of these results.

We can try to use some subset of the features to obtain error rate using the considered models, and, in light of the results, draw conclusions about data and models behaviour.

!! compare LDA with tied covariance model, since closely related !!

The results of the models trained with different subset of features (Table 3) shows that, as we analysed in the previous section, discarding features that badly fits Gaussian assumption lead to a slighltly worse, but still comparable to the results obtained with the full set of features: this is due probably to the fact that, while still providing some useful information, the last two features

Table 3: Table showing the error rate for Gaussian models, using a subset of the features

Model	Error Rate / Features subset		
	1-2	**3-4**	**1-4**
Multivariate	36.50%	9.45%	7.95%
Naive	**36.30%**	9.45%	**7.65%**
Tied	49.45%	**9.40%**	9.50%
NaiveTied	49.40%	**9.40%**	9.50%
LDA	49.45%	**9.40%**	9.50%

are probably the less determinant in the classification task using Gaussian model, since their distribution is far from a Gaussian one.

Moreover, we can notice that, while using first two features lead to bad results overall, Naive model outperform the others: infact, if we analyse the correlation matrix, we can notice that first two features are the less correlated pair of all. Since Naive Bayes hypothesis assume that features are not correlated, probably in this subset the hypothesis lead to a better understanding of underlying pattern, and not considering correlation allows to simplify the model in a way that help the classification task. For Tied models and LDA, we can observe that they perform poorly for this subset, probably due to its poor separability.

For the second and the third feature, we obtain overall good results: this confirm us, as we stated in the first section (**??**) and proved analysing LDA direction results, that these two features are probably the most discriminant and useful in the classification task. Here the tied assumption works better probably due to the separability of the features.

Table 4: Table showing the error rate for Gaussian models, using a decreasing number Principal Components

Model	Error Rate / PCA					
	m=6	**m=5**	**m=4**	**m=3**	**m=2**	**m=1**
Multivariate	**7.00%**	7.10%	8.05%	8.80%	8.80%	9.25%
Naive	8.90%	8.75%	8.85%	9.00%	8.85%	9.25%
Tied	9.30%	9.30%	9.25%	9.25%	9.25%	9.35%
NaiveTied	9.20%	9.20%	9.15%	9.10%	9.35%	9.35%
LDA	9.30%	9.30%	9.25%	9.25%	9.25%	9.35%

Finally, using PCA to preprocess data, we can see from the Table 4 that, for Multivariate Gaussian Model, its performance degrades as the components taken into account decrease. Naive Bayes Model shows consistent performance across all PCA components, but still don't perform better than MVG. Tied covariance model exhibit minimal sensitivity to the reduction in dimensions, maintaining error rates around 9.25-9.35%. Their performance does not improve significantly even with all features.

From this results we can conclude that PCA captures the principal directions of variance in the data, but it does not necessarily align these directions with the features most useful for classification. This can explain why reducing the dimensionality through PCA degrades performance for the Multivariate model, which can effectively leverage the original feature structure to obtain optimal performance. As suggested in the provided text, we consider 5 different application for our analysis: ($\pi = 0.5, C_{fn} = 1.0, C_{fp} = 1.0$), ($\pi =, C_{fn} = 1.0, C_{fp} = 1.0$), ($\pi = 0.1, C_{fn} = 1.0, C_{fp} = 1.0$), ($\pi = 0.5, C_{fn} = 1.0, C_{fp} = 9.0$) and ($\pi = 0.5, C_{fn} = 9.0, C_{fp} = 1.0$). In the analysis of the five applications provided, the effective prior $\tilde{\pi}$ was computed using the formula:

$$\tilde{\pi} = \frac{\pi_T C_{fn}}{\pi_T C_{fn} + (1 - \pi_T) C_{fp}}$$

Notably, the last two applications are equivalent to other two in terms of effective prior. This implies that, while the raw priors and costs differ, their impact on decision-making thresholds is identical, so we can discard them.

Table 5: Table showing the DCF and minDCF using Gaussian models for the effective prior $\tilde{\pi} = 0.5$, using a decreasing number Principal Components

	DCF / minDCF / miscal. loss (PCA)						
	any	**m=6**	**m=5**	**m=4**	**m=3**	**m=2**	**m=1**
MV	0.1399	0.1399	0.1419	0.1609	0.1759	0.1760	0.1850
	0.1302	0.1302	0.1331	0.1537	0.1734	0.1731	0.1769
	0.0098	0.0098	0.0088	0.0072	0.0025	0.0029	0.0081
N	0.1439	0.1780	0.1750	0.1770	0.1799	0.1770	0.1850
	0.1311	0.1727	0.1737	0.1717	0.1746	0.1710	0.1769
	0.0128	0.0053	0.0013	0.0053	0.0053	0.0060	0.0081
T	0.1860	0.1860	0.1860	0.1850	0.1850	0.1850	0.1870
	0.1812	0.1812	0.1812	0.1821	0.1830	0.1789	0.1769
	0.0048	0.0048	0.0049	0.0029	0.0020	0.0062	0.0101
NT	0.1860	0.1840	0.1840	0.1830	0.1820	0.1870	0.1870
	0.1803	0.1800	0.1800	0.1789	0.1790	0.1769	0.1769
	0.0057	0.0041	0.0040	0.0041	0.0031	0.0101	0.0101

Table 6: Table showing the DCF and minDCF using Gaussian models for the effective prior $\tilde{\pi} = 0.9$, using a decreasing number Principal Components

	DCF / minDCF / miscal. loss (PCA)						
	any	**m=6**	**m=5**	**m=4**	**m=3**	**m=2**	**m=1**
MV	0.4001	0.4001	0.3980	0.4598	0.4680	0.4434	0.4775
	0.3423	0.3423	0.3512	0.4150	0.4392	0.4384	0.4342
	0.0577	0.0577	0.0468	0.0448	0.0288	0.0050	0.0433
N	0.3893	0.4512	0.4660	0.4630	0.4591	0.4424	0.4775
	0.3510	0.4359	0.4340	0.4313	0.4343	0.4323	0.4342
	0.0383	0.0153	0.0320	0.0317	0.0248	0.0101	0.0433
T	0.4626	0.4626	0.4626	0.4615	0.4565	0.4785	0.4806
	0.4421	0.4421	0.4451	0.4441	0.4342	0.4352	0.4342
	0.0204	0.0204	0.0174	0.0174	0.0223	0.0433	0.0464
NT	0.4716	0.4696	0.4696	0.4696	0.4706	0.4796	0.4806
	0.4434	0.4300	0.4300	0.4361	0.4361	0.4312	0.4342
	0.0282	0.0396	0.0396	0.0336	0.0346	0.0484	0.0464

As we can see from tables Table 5, Table 6 and Table 7, comparing minimum DCF of different models for different PCA configuration, we can see that the best model performance are obtained for the first two application

Table 7: Table showing the DCF and minDCF using Gaussian models for the effective prior $\tilde{\pi} = 0.1$, using a decreasing number Principal Components

	DCF / minDCF / miscal. loss (PCA)						
	any	**m=6**	**m=5**	**m=4**	**m=3**	**m=2**	**m=1**
MV	0.3051	0.3051	0.3041	0.3529	0.3879	0.3879	0.3970
	0.2629	0.2629	0.2738	0.3012	0.3563	0.3526	0.3686
	0.0422	0.0422	0.0303	0.0517	0.0316	0.0353	0.0283
N	0.3022	0.3920	0.3930	0.3970	0.3950	0.3869	0.3970
	0.2570	0.3535	0.3545	0.3614	0.3645	0.3562	0.3686
	0.0452	0.0385	0.0385	0.0356	0.0305	0.0307	0.0283
T	0.4061	0.4061	0.4051	0.4031	0.4082	0.3960	0.4021
	0.3628	0.3628	0.3648	0.3610	0.3681	0.3630	0.3686
	0.0432	0.0432	0.0402	0.0421	0.0401	0.0330	0.0334
NT	0.4080	0.4131	0.4112	0.4122	0.4112	0.3910	0.4021
	0.3631	0.3611	0.3611	0.3641	0.3610	0.3590	0.3686
	0.0449	0.0520	0.0500	0.0480	0.0502	0.0320	0.0334

by Multivariate Gaussian model without PCA and with PCA using 6 Principal Components, while for the last application the best value for minimum DCF is obtained by Naive Gaussian model without any preprocessing technique. Comparing min DCF values we can conclude that the first application is the most promising one, since values vary little from each other and the lowest value found for this application is even the lowest overall. The models performance for this application are also comparable each other. Overall, the performance of the first application are the best from the point of view of minDCF: values vary little from each other, and the best model have the smallest minDCF of all. For the other two, values for minimum DCF vary more than the first one, and they are relatively higher than the first one, with the third one having slighltly lower values than the third one.

Comparing DCFs values, we notice that for the first application the models calibration appear to be comparable between the different models, with Naive Bayes obtaining the best result using PCA with 5 Principal Components. For the second and the third one, the calibration loss appear to be comparable among models, with a single exception for the second application in which the only configuration that outperform the others is for the Multivariate Gaussian model trained using PCA with 2 Principal Components. For the third application, the order of magnitude of the calibration loss is higher than the second one: since the application that we're going to use will be this, we'll probably need to calibrate scores to obtain optimal performance from the models.

From the Bayes error plots at Figure 6 we can notice that the minimum DCF for the Multivariate Gaussian and Naive Bayes models remained relatively consistent across different applications, indicating robust performance. The rankings of these models were also consistent, with these two models typically outperforming Tied and Tied Naive Gaussian models in terms of minimum DCF. All models exhibited good calibration, as evidenced by the low miscalibration loss values. However, Tied Covariance Naive Bayes model had the lowest miscalibration loss, which indicates that it aligns well with the given application's effective prior, even though its overall performance (minDCF) was not the best.

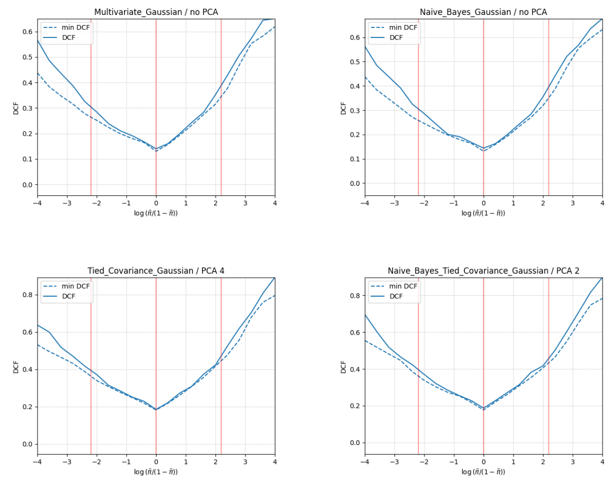

Figure 6: Bayes error plots of most promising setup of PCA for all the considered models

The Naive Bayes Gaussian model consistently performed the best, followed closely by the Multivariate Gaussian model for this application. These rankings were stable across applications, with Multivariate Gaussian Model and Naive Bayes Gaussian model performance beying comparable.

3.2 Logistic Regression

3.2.1 Early analysis

3.2.2 Quantitative evaluation