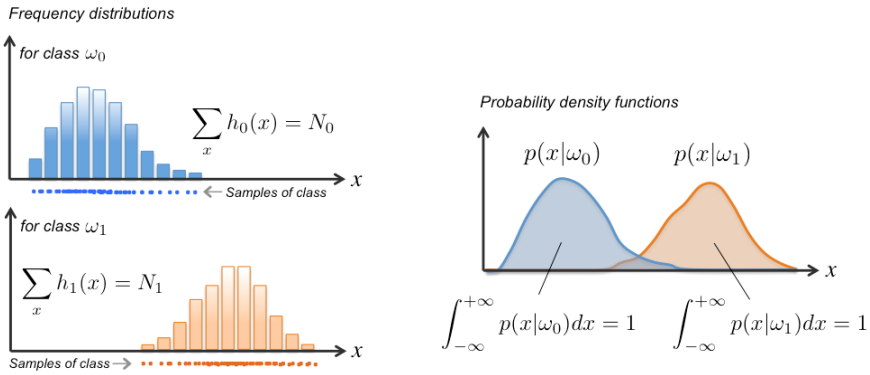

Fig. 5.19: A good class distribution for feature x and two classes ω_0 and ω_1.

5.6 Feature selection

Which features are relevant? or which features should be extracted? Such questions arise because there is a huge number of features that can be extracted and unfortunately, we don't know which or which of them are really necessary. First, we should not forget the reason why we extract features... so at least we could answer the question: why are they really necessary? As we explained in the introduction of this Chapter (see Section 5.1), our task is to recognize or detect our *objects of interest*, and we need to differentiate them from the *background*. For example in X-ray images of aluminum castings, we can have several *potential defects* that were detected using some segmentation approaches (see Section 4.5). As the segmentation is far from perfect, the potential defects consist of not only 'defects', but 'regular structures' as well. Our object of interest in this example is the defects, whereas the background corresponds to the regular structure of the aluminum casting. From the X-ray images, we can extract features that describe the potential defects (*e.g.*, area, width, height, location, contrast, statistical textures, etc.). In order to recognize the defects, we have to analyze the extracted features of the available potential defects and select those features that are able to properly separate the defects from the regular structures. In this example, we could expect a good separability of both classes by selecting the contrast (see Section 5.3.2) because it gives a measure of the difference in the gray value between the segmented region and its neighborhood.

5.6.1 Basics

In general, if we have two classes (ω_1 for 'object of interest' and ω_0 for 'background') and we want to analyze the performance of extracted feature x, *e.g.*, contrast, we can investigate the frequency distribution for each class as illustrated in

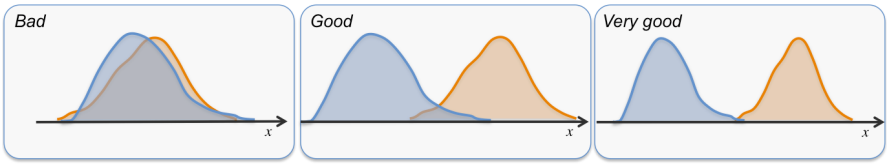

Fig. 5.20: Class distribution for there different features. It is clear that the best separability is achieved by the last features.

histograms of Fig. 5.19. In this case, for frequency distribution of class ω_k we only take into account the samples that belong to the k-th class. In this *supervised* approach, the label d_i of i-th sample must be available, for $i = 1 \ldots N$ for N samples. That means, someone, for example an expert, must annotate the label of each sample of the dataset. Thus, if the i-th sample belongs to class ω_k, then $d_i = k$. For N samples we will have a vector $\mathbf{d}$ with N elements.

The available data should be representative enough, that means on the one hand that N_k, the number of samples of class ω_k, must be large enough, and on the other hand, for each class, the samples of the dataset must include the full range of variations that exist in the class itself. In our example, if x is the contrast of potential defects, we compute the frequency distribution of class ω_1 and the frequency distribution of class ω_0 by considering only the samples of 'defects' and 'regular structures' respectively. In addition, we can estimate the probability density functions from each frequency distribution known as $p(x|\omega_k)$, *i.e.*, the probability of x given class ω_k. As we can see in Fig. 5.19, feature x is able to properly separate both classes because it takes low values for class ω_0 and high values for class ω_1, however, there is some degree of overlapping.

In feature selection, we have to decide just which features (extracted from our potential objects of interest) are relevant to the classification. By analyzing each extracted feature, three general scenarios are possible (see Fig. 5.20): a bad, a good and a very good separability. In the first scenario, the confusion between both classes is so high that it is impossible to separate the classes satisfactorily, *i.e.*, a classifier cannot distinguish either of the classes. In the second scenario, a good separation is possible with some overlapping of the classes, *i.e.*, a classifier will not recognize both classes perfectly, however, in many cases this scenario can be acceptable. In the third scenario, the separability is very good, and a classifier could identify both classes in approximately 100% of the cases. If all extracted features are in the first scenario, there is no classifier that can separate both classes, *i.e.*, new features are required. On the other hand, if we have a feature of the third scenario, the recognition can be easily performed by thresholding. In this case, no sophisticated classifiers are required. Unfortunately, the third scenario seldom occurs and we have to deal with some degree of overlapping.

In order to overcome the overlapping problem, more than one feature can be selected, however, the same three scenarios are also possible (see Fig. 5.21 for two features).

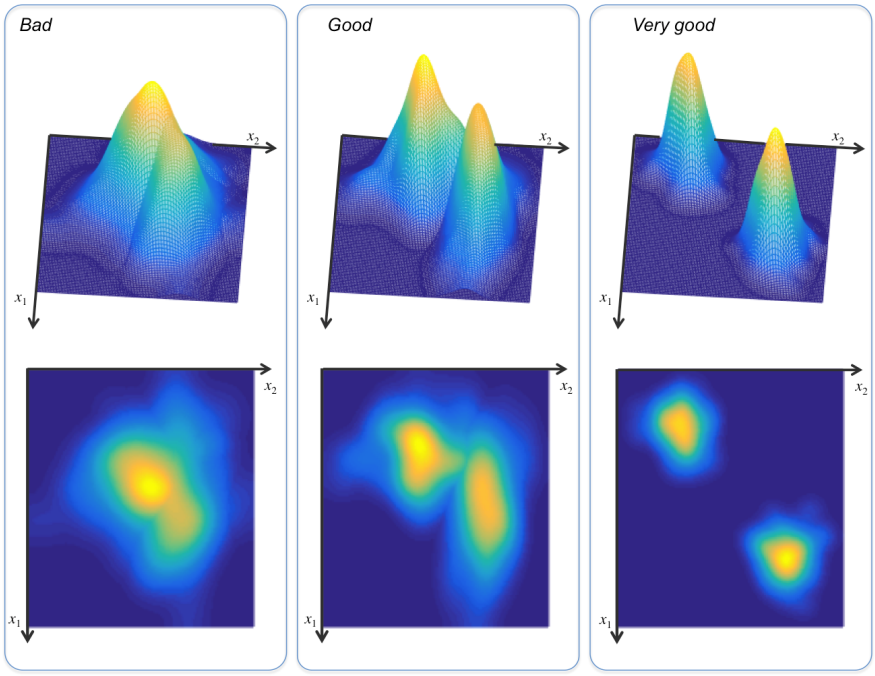

Fig. 5.21: Class distribution for three different pairs of features (x_1, x_2). As in Fig. 5.20, it is clear that the best separability is achieved by the last feature. The figure shows two types of visualization of the feature space of two features: a 3D representation and a top view using a colormap. A third type of visualization for this data is available in Fig. 5.23.

In this Section, we will review some known techniques that can be used in feature selection. The reasons why feature selection is necessary are as follows:

1. It is possible that some extracted features are not *discriminative* enough, *i.e.*, there is no information in these features for separating the classes. An example of this case is illustrated in the first scenario in Fig. 5.20. This may occur for example when we consider the mean gray value (5.16) of potential defects when detecting defects in welds. The (absolute) gray value of some defects can be very similar to the gray value of some regions of the background. In this example, we need rather a relative gray value such as a contrast (5.21).
2. Some extracted features with good separability could be redundant, *i.e.*, they are somehow correlated. An example of this case is shown in the third scenario of Fig.5.21 because x_1 are highly correlated with x_2. In this example, the separability by using (x_1, x_2) is very similar to the separability by using x_1 only. This may occur for example when we use two contrasts (5.21) to discriminate defects from background, maybe one contrast is enough and the second one does not increase the separability at all because it is redundant.

3. In order to simplify the testing stage, it is much better to extract a low number of features. In the training stage, we are allowed to investigate a huge number of features (in order to select some of them), however, in the testing stage it is recommended to use a reduced subset of these. Thus, the computational time of the testing stage will be significantly reduced.
4. In order to avoid the *curse of the dimensionality*, it is highly recommended to train a classifier with a low number of features. When we increase the number of selected features, the volume of our feature space increases exponentially. Thus, in order to be statistical significant we need to collect exponentially larger amounts of samples. This is not possible with a limited number of samples, for this reason the performance of the classifier tends to become reduced as the number of features increases [98].
5. Last but not least, in order to avoid *false correlations* some features should not have been extracted at all and must be filtered out in this step... just in case they were extracted. This is a very common mistake and it must be avoided before a classifier is trained. An example may be by trying to recognize a threat object (*e.g.*, a knife) in baggage screening using features that are not rotation invariant. Imagine that we extract all elliptical features (see Section 5.2.2) of potential knives. The orientation α of the fitted ellipse is extracted as well (5.8). It is possible, that in our training dataset the orientation of the potential knives is always very vertical, as in the series B0008 of $\mathbb{GDX}$ray (see Fig. 2.10). That means, the extracted feature α could have a distribution like scenario two or three of Fig. 5.20. The *separability* of this feature could lead to misinterpretation because we could think that we found an extraordinary good feature that can separate knives from background, however, we are saying that a knife must be always vertical if we want to recognize it! It is clear that the orientation should not have been extracted in order to avoid a false correlation. Another typical mistake occurs when considering the location (5.2) as feature in defect recognition. In our training data, it is possible that all defects are located in one part of the image, however, in real life they can be everywhere. Obviously, there is no algorithm that detects this error. When we design an automated system, we have to be very careful in order to select manually those features that could lead to false correlations. A guide to avoid this problem is suggested in Fig. 5.22.

Formally, the extracted features of a sample can be represented as a row vector $\mathbf{x}$ of m elements, where m is the number of extracted features. Thus, a sample can be viewed as a point $\mathbf{x} = [x_1 \dots x_m]$ in the feature space of m dimensions (see Fig. 5.19 for one dimension and Fig. 5.23 for two dimensions). The feature vector of all samples can be stored in matrix $\mathbf{X}$ of size $N \times m$, where N is the number of samples, *i.e.*, $N = \sum_k N_k$, and N_k is the number of samples of class ω_k. The j-th column of $\mathbf{X}$, called $\mathbf{x}_j$, consists of the values that take feature x_i in all samples. In addition, element x_{ij} means the feature x_j of i-th sample. The features are usually normalized as:

$$\tilde{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j} \tag{5.40}$$

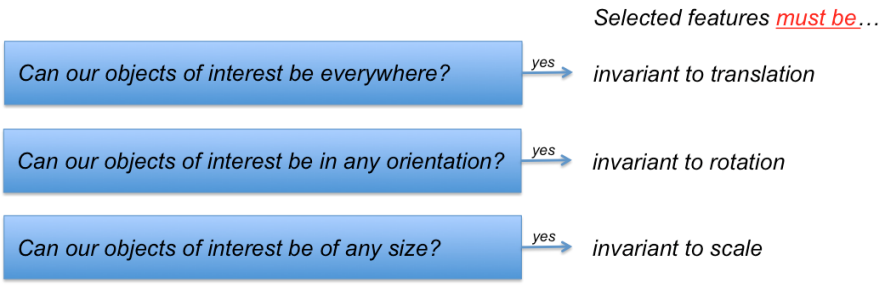

Fig. 5.22: In order to avoid false correlations we can follow these steps when extracting features. In these three cases, features that are extracted with 𝕏VIS Toolbox can be manually eliminated using commands Xnotranslation, Xnorotation and Xnoscale respectively.

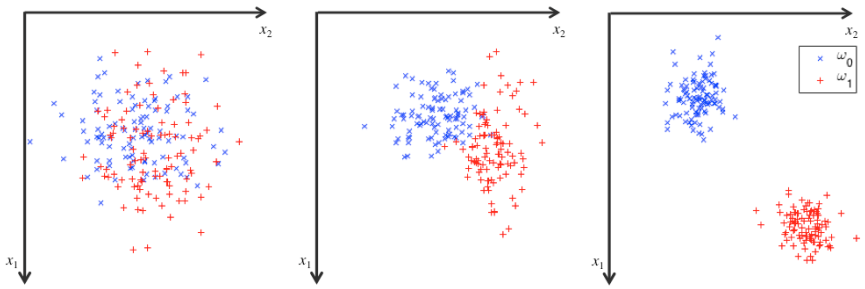

Fig. 5.23: In this visualization each sample is represented as a point in the feature space of two dimensions (x_1, x_2). The figure shows the visualization for the three examples of Fig. 5.21.

for $i = 1,...,N$ and $j = 1,...,m$, where μ_j and σ_j are the mean and standard deviation respectively of the $\mathbf{x}_j$. The normalized features have zero mean and a standard deviation equal to one[3].

A very good practice is to eliminate *i)* those features that are very constant, *i.e.*, $\sigma_j < \theta_1$, where θ_1 is some threshold, *e.g.*, 10^{-8}, and *ii)* those features that are very correlated, *i.e.*, if two of any extracted features ($\mathbf{x}_i$ and $\mathbf{x}_j$) are highly correlated (if $|\text{cov}(\mathbf{x}_i, \mathbf{x}_j)|/(\sigma_i \sigma_j) > \theta_2$) one of them is eliminated. We can set θ_2 to 0.99 for example. The feature 'cleaning' is implemented in function Xfclean of 𝕏VIS Toolbox.

The key idea of the feature selection is to select a subset of p features ($p \leq m$) that leads to the smallest classification error. The selected p features are arranged in a new row vector of p elements $\mathbf{z} = [z_1 \ldots z_p]$. The selected feature vector of all samples can be stored in matrix $\mathbf{Z}$ of size $N \times p$. This process is illustrated in Fig.

[3] In 𝕏VIS Toolbox, (5.40) is implemented in function Xfnorm

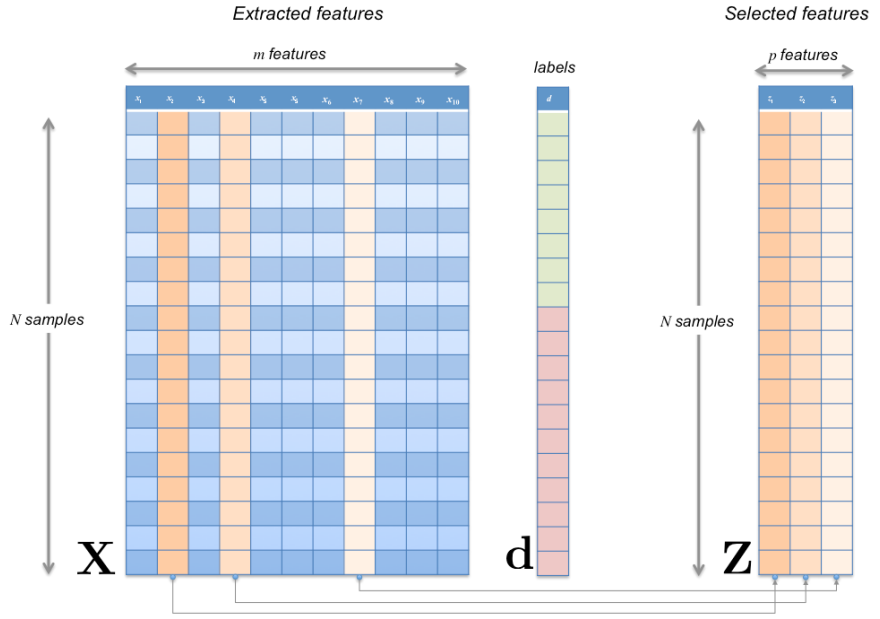

Fig. 5.24: Feature selection: there are m extracted features, from them p are selected. As we can see in the labels, the first samples belong to one class and the last one to another.

5.24 for $m = 10$ and $p = 3$. The p selected features are columns $s_1, s_2 \dots s_p$ of $\mathbf{X}$, that means column j of $\mathbf{Z}$ is equal to column s_j of $\mathbf{X}$, $\mathbf{z}_j = \mathbf{x}_{s_j}$, for $j = 1 \dots p$.

For a given set of selected features $\mathbf{s} = (s_1, s_2 \ \dots \ s_p)$ we need some measurement of *separability* that can be used to assess the performance of the selection, *i.e.*, for our three scenarios (see Fig. 5.20 and Fig. 5.21), this measurement should be low, high and very high respectively. We define the separability J as a function of $\mathbf{Z}$ (selected features) and $\mathbf{d}$ (labels of the samples). Since $\mathbf{Z}$ corresponds to the selected columns of $\mathbf{X}$ that are defined by $\mathbf{s}$, we can write the separability as $J(\mathbf{X}, \mathbf{s}, \mathbf{d})$.

The problem of feature selection can be stated as follows, given the extracted features for N samples ($\mathbf{X}$) and the labels of each sample ($\mathbf{d}$), find a set of features (indexed by $\mathbf{s} = (s_1, s_2 \ \dots \ s_p)$) that maximizes the separability ($J(\mathbf{X}, \mathbf{s}, \mathbf{d})$). This is an optimization problem

$$\hat{\mathbf{s}} = \underset{\mathbf{s} \subseteq \mathbf{Q}}{\operatorname{argmax}} J(\mathbf{X}, \mathbf{s}, \mathbf{d}), \qquad \text{s.t.} |\mathbf{s}| = p, \tag{5.41}$$

where $\mathbf{Q} = (1, 2, \dots m)$ is the set of all possible indices that can take $\mathbf{s}$.

There are many approaches that can be used to measure the separability. A very common one is based on Fisher criterion that ensures: *i)* a small intraclass variation and *ii)* a large interclass variation in the space of the selected features.

For the first condition, the intraclass-covariance (known also as between-class covariance matrix) is used:

$$\mathbf{C}_b = \sum_k p_k (\bar{\mathbf{z}}_k - \bar{\mathbf{z}})(\bar{\mathbf{z}}_k - \bar{\mathbf{z}})^\mathsf{T}, \tag{5.42}$$

where p_k denotes the a-priori probability of the k-th class, $\bar{\mathbf{z}}_k$ and $\bar{\mathbf{z}}$ are the mean value of the k-th class and the mean value of the selected features.

For the second condition, the interclass-covariance (known also as within-class covariance matrix) is used:

$$\mathbf{C}_w = \sum_{k=1}^{K} p_k \mathbf{C}_k, \tag{5.43}$$

where the covariance matrix of the k-th class is given by:

$$\mathbf{C}_k = \frac{1}{N_k - 1} \sum_{j=1}^{N_k} (\mathbf{z}_{kj} - \bar{\mathbf{z}}_k)(\mathbf{z}_{kj} - \bar{\mathbf{z}}_k)^\mathsf{T}, \tag{5.44}$$

with $\mathbf{z}_{kj}$ the j-th selected feature vector of the k-th class, N_k is the number of samples in the k-th class. Selection performance can be evaluated using the spur criterion for the selected features $\mathbf{z}$:

$$J = \text{spur}\left(\mathbf{C}_w^{-1} \mathbf{C}_b\right). \tag{5.45}$$

where 'spur' means the sum of the diagonal. The larger the objective function J, the higher the selection performance. For the examples of Fig. 5.23, this function takes the values 0.1, 2.1 and 27.8 respectively. The objective function defined in (5.45) can be used directly in (5.41).

Another approach that can be used to measure the separability is to compute the accuracy of a classifier with the selected features. In this approach, we divide $\mathbf{Z}$ in two subsets of samples: training and testing datasets. A classifier is designed using the training set, and afterwards is tested using the testing set. The separability J is defined as the accuracy evaluated on the testing set, *i.e.*, the ratio of samples that were correctly classified to the total number of samples[4].

The features can be selected using several state–of–art algorithms reported in the literature. In the following, some selection algorithms are presented.

5.6.2 Exhaustive search

The selection of the features is performed by evaluating (5.41) for all possible combination of p features of $\mathbf{X}$. The combination that achieves the highest value for J is selected. This approach ensures that global maximum of J is attained, however, it requires $n = m!/(p!(m-p)!)$ evaluations of J. The number n can be prohibited for large m and p values. For instance, if we have $m = 100$ extracted features and

[4] Classifiers and accuracy estimation are covered in Chapter 6.

we want to select $p = 10$ features, then 1.73×10^{13} evaluations of J are required using exhaustive search. This function is implemented in command Xfexsearch of 𝕏VIS Toolbox.

5.6.3 Branch and bound

In Branch and bound, the global maximum of J is ensured also [188]. Given that J is a monotonically increasing function, *i.e.*, $J(\mathbf{z}_1) < J(\mathbf{z}_1,\mathbf{z}_2) < \ldots J(\mathbf{z}_1,\ldots\mathbf{z}_p)$, we can considerably reduce the number of evaluations of J. In branch and bound technique, we use a tree representation, where the root corresponds to the set of all features, and a node of the tree corresponds to a combination of features. The children's nodes are subsets of their parents. Nodes in the k-th level represent combinations of $m-k$ features. We starts by evaluating J at the main node ($k = 0$) with all features. This will be our *bound*, the current maximum. The key idea of the algorithm is to evaluate those children nodes that have a separability J higher than the bound. If that is the case, the we update the bound. Consequently, nodes whose separability J is lower than the bound will not be evaluated. This method is implemented in command Xfbb of 𝕏VIS Toolbox.

5.6.4 Sequential forward selection

This method selects the best single feature and then adds one feature at a time that, in combination with the selected features, maximizes the separability. The iteration is stopped once the selected subset reaches p features. This method requires $n = pm - p(p-1)/2$ evaluations. For instance, if we have $m = 100$ extracted features and we want to select $p = 10$ features, then 955 evaluations of J are required using SFS, this is a very low number in comparison with the number of evaluations required for exhaustive search. This method is implemented in command Xsfs of 𝕏VIS Toolbox.


Matlab Example 5.8: In this example, we extract intensity features of small cropped X-ray images (100×100 pixels) of salmon filets. The cropped images are in series N0002 of 𝔻𝕏ray. There are 100 cropped images with fishbones and 100 with no fishbones. The idea is to select those features that can be relevant for the separation between both classes 'fishbones' and 'background' (labels 1 and 0 respectively). Using the selected features, we could detect small regions with fishbones in an X-ray image of a salmon filet. In this series, the labels of the 200 cropped images are available. We initially extract several intensity features (more than 300). Afterwards, features that are not rotation invariant are eliminated because fishbones can be oriented in any direction. Additionally, high correlated or constant features are eliminated as well. We select 15 features using SFS and 3 from them using exhaus-

tive search. The computational time of the feature selection step is short because we are dealing with a small number of features and samples.

Listing 5.8 : Feature selection with SFS

```
% FeatureSelectionSFS.m
close all
gdx_dir      = Xgdxdir('N',2);                  % directory of series N0002 of GDX
opf.b        = Xfxbuild({'basicint','gabor',... % features to be extracted
                         'lbpri','haralick'}); % rotation invariant for LBP
[X0,Xn0]     = Xfxtractor(gdx_dir,'png',opf);   % feature extraction
[X,Xn]       = Xnorotation(X0,Xn0);             % only rotation invariant features
d            = Xloaddata('N',2,'labels.txt');   % labels
sc           = Xfclean(X);                      % delete constant and correlated features
figure
Xc           = X(:,sc);                         % sc = indices of selected features
Xcn          = Xn(sc,:);
opsfs.show   = 1;                               % display results
opsfs.p      = 15;                              % 15 features will be selected
s1           = Xsfs(Xc,d,opsfs);                % using SFS
Y1           = Xc(:,s1);                        % s1 = indices of selected features
Y1n          = Xcn(s1,:);
opexs.show   = 1;                               % display results
opexs.p      = 3;                               % 3 (from 15) features will be selected
s2           = Xfexsearch(Y1,d,opexs);          % using exhaustive search
Y2           = Y1(:,s2);
Y2n          = Y1n(s2,:);
figure
Xplotfeatures(Y2,d,Y2n)                         % plot of feature space
grid on; view(-25,30)
```

The output of this code is shown in Fig. 5.25. In this example, the features were extracted using commands Xfxtractor, and the features were selected using Xsfs and Xfexsearch of $\mathbb{X}$VIS Toolbox. In this experiment, only LBP feature were selected, *i.e.*, in testing stage it is not necessary to extract Gabor, Haralick and other basic intensity features. This result is very meaningful because the computational time is considerably reduced: from 0.32 s/image to 0.018 s/image (in a computer with 8GB RAM and a processor Intel Core i5 of 2.8 GHz, OS X 10.10.2) □

5.6.5 Sequential backward selection

This method selects all features and then eliminates one feature at a time that maximizes the separability. The iteration is stopped once the selected subset reaches p features. This method requires $n = (m-p+1)m-(m-p)(m-p+1)/2$ evaluations. For instance, if we have $m = 100$ extracted features and we want to select $p = 10$ features, then 5005 evaluations of J are required using SBS.

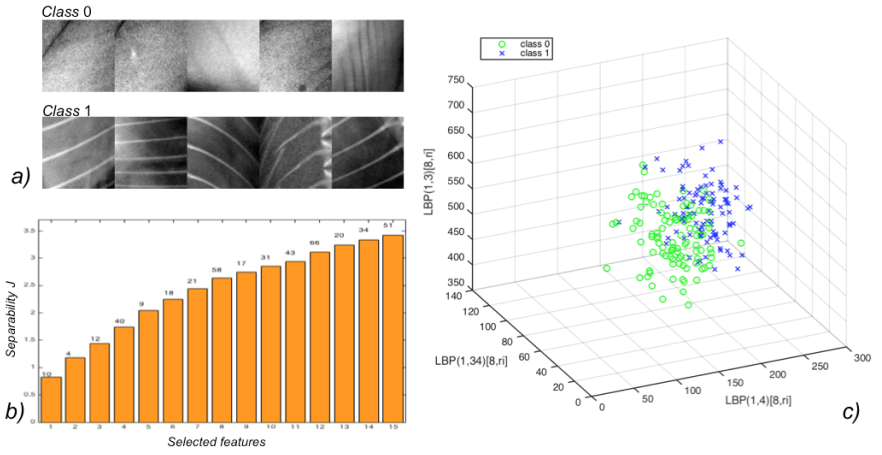

Fig. 5.25: Feature selection using SFS and exhaustive search: a) Some cropped X-ray images of both classes 'background' and 'fishbones'. In this example there are 200 cropped X-ray images, 100 from each class. There are $m = 241$ extracted features from N=200 samples, *i.e.*, the extracted features are stored in matrix $\mathbf{X}$ of size 200×241. b) Sequential forward selection. There are $p_1 = 15$ selected features and they correspond to columns 10, 4, 12, 40, 9 ... of $\mathbf{X}$, *i.e.*, the selected features are stored in matrix $\mathbf{Y}_1$ of size 200×15. c) Using exhaustive search, $p_2 = 3$ features are selected from $\mathbf{Y}_1$. The result is stored in matrix $\mathbf{Y}_2$ of 200×3 elements. The figure shows the feature space in 3D. The selected features are certain LBP features. We can see that the separability is 'good' and correspond to our second scenario. [→ Example 5.8]

5.6.6 Ranking by class separability criteria

Features are ranked using an independent evaluation criterion to assess the significance of every feature for separating two labeled groups. The absolute value two–sample t–Student test with pooled variance estimate is used as an evaluation criterion [142]. This method is implemented in command Xfrank of 𝕏VIS Toolbox.

5.6.7 Forward orthogonal search

In FOS, features are selected one at a time, by estimating the capability of each specified candidate feature subset to represent the overall features in the measurement feature space using a squared correlation function to measure the dependency between features [261]. This method is implemented in command Xfosmod of 𝕏VIS Toolbox.

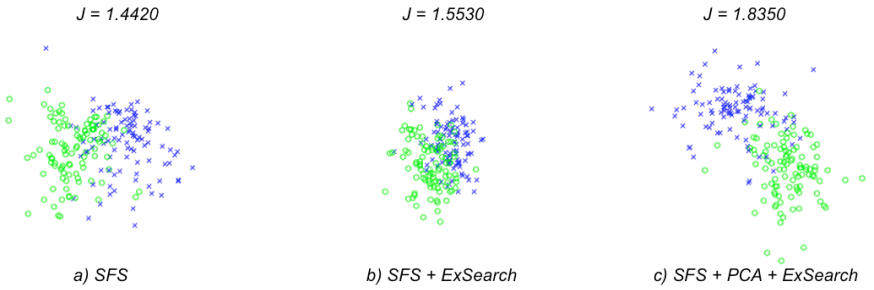

Fig. 5.26: Separability of three different feature selection methods for fishbone detection (see example 5.8 for details). Each visualization is a 3D plot, the axes and the grid are not represented for the sake of simplicity. a) The best three features after SFS. It corresponds to the first three columns of variable `Y1` that has 15 columns. b) The best three features (from 15 features selected by SFS) after exhaustive search. It correspond to variable `Y2`. This plot is the same of Fig. 5.25c. c) The three principal components of `Y1` is computed (`Ypca = Xpca(Y1,3);`). A new set of features including `Y1` and `Ypca` is defined (`Ynew = [Y1 Ypca];`). The plot shows the best three features of `Ynew` that are selected using exhaustive search. The last selection included one principal component. We observe how the separability J after Fisher criterion (5.45) is increased in each step.

5.6.8 Least square estimation

In LSE, features are selected one at a time, evaluating the capacity of the select feature subsets to reproduce sample projections on principal axis using Principal Component Analysis (PCA) [139]. This method is implemented in command Xlsef of 𝕏VIS Toolbox.

5.6.9 Combination with principal components

The first p principal components of the large set of features $\mathbf{X}$ (or a pre-selected subset of features using one of the mentioned approaches) are appended as new columns (features) of $\mathbf{X}$. Thus, we have a new set of features $\mathbf{X}_{\text{new}} = [\mathbf{X} \quad \text{pca}(\mathbf{X}, p)]$. Afterwards, a feature selection algorithm (like SFS or exhaustive search) is computed on $\mathbf{X}_{\text{new}}$. As result, the selected features can be some original features and some principal components [44]. An example is shown in Fig. 5.26. In this example, this method achieved the best separability with only three features, however, it is worth mentioning that using this method the computational time is increased significantly in the testing stage. The reason is not because we have to compute the PCA transformation, but because we have to extract all features required by PCA.

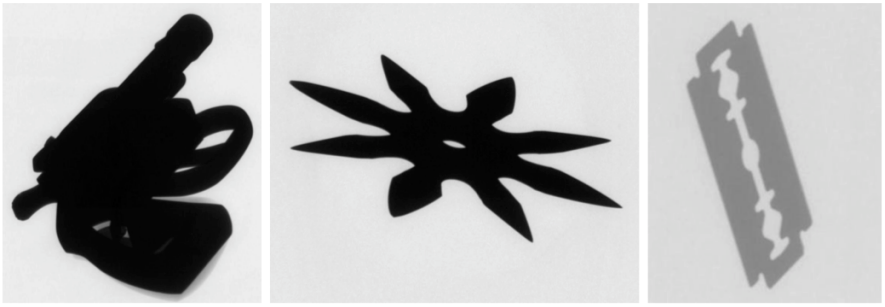

Fig. 5.27: Some objects used in example of Section 5.7: a handgun, a shuriken and a razor blade, from GDXray series B0049, B0050, and B0051 respectively.

5.6.10 Feature selection based in mutual information

In mRMR, the features are selected based on two criteria: minimal redundancy in order to remove redundant variables; and maximal relevance in order to select the relevant features that are able to separate the classes [201]. This method is implemented in command XfmRMR of XVIS Toolbox.