CHAPTER

Supervised Learning: The Epilogue

10

10.1 INTRODUCTION

This chapter is the last one related to supervised learning, and it is intended to serve three purposes. The first sections focus on the last stage of the design procedure of a classification system. In other words, we assume that an optimal classifier has been designed, based on a selected set of training feature vectors. Our goal now is to evaluate its performance with respect to the probability of classification error associated with the designed system. To this end, methodologies will be developed for the estimation of the classification error probability, using the available, hence finite, set of data. Once the estimated error is considered satisfactory, full evaluation of the system performance is carried out in the real environment for which the system has been designed, such as a hospital for a medical diagnosis system or a factory for an industrial production–oriented system.

It is important to note that the evaluation stage is not cut off from the previous stages of the design procedure. On the contrary, it is an integral part of the procedure. The evaluation of the system's performance will determine whether the designed system complies with the requirements imposed by the specific application and intended use of the system. If this is not the case, the designer may have to reconsider and redesign parts of the system. Furthermore, the misclassification probability can also be used as a performance index, in the feature selection stage, to choose the best features associated with a specific classifier.

The second goal of this chapter is to tie together the various design stages that have been considered separately, so far, in the context of a case study coming from medical ultrasound imaging. Our purpose is to help the reader to get a better feeling, via an example, on how a classification system is built by combining the various design stages. Techniques for feature generation, feature selection, classifier design and system evaluation will be mobilized in order to develop a realistic computer-aided diagnosis medical system to assist a doctor reaching a decision.

In the final sections of the chapter, we will move away from the fully supervised nature of the problem that we have considered so far in the book, and we will allow unlabeled data to enter the scene. As we will see, in certain cases, unlabeled

data can offer additional information that can help the designer in cases where the number of labeled data is limited. Semi-supervised learning is gaining in importance in recent years, and it is currently among the hottest research areas. The aim of this chapter is to introduce the reader to the semi-supervised learning basics and to indicate the possible performance improvement that unlabeled data may offer if used properly.

10.2 ERROR-COUNTING APPROACH

Let us consider an M-class classification task. Our objective is to *estimate* the classification error probability by testing the "correct/false" response of an independently designed classifier using a *finite set* of N test feature vectors. Let N_i be the vectors in each class, with $\sum_{i=1}^{M} N_i = N$ and P_i the corresponding error probability for class ω_i. *Assuming independence among the feature vectors*, the probability of k_i vectors from class ω_i being misclassified is given by the binomial distribution

$$\text{prob}\{k_i \textit{ misclassified}\} = \binom{N_i}{k_i} P_i^{k_i}(1-P_i)^{N_i-k_i} \tag{10.1}$$

In our case the probabilities P_i are not known. An estimate $\hat{P}_i$ results if we maximize (10.1) with respect to P_i. Differentiating and equating to zero result in our familiar estimate

$$\hat{P}_i = \frac{k_i}{N_i} \tag{10.2}$$

Thus, the total error probability estimate is given by

$$\hat{P} = \sum_{i=1}^{M} P(\omega_i)\frac{k_i}{N_i} \tag{10.3}$$

where $P(\omega_i)$ is the occurrence probability of class ω_i. We will now show that $\hat{P}$ is an unbiased estimate of the true error probability. Indeed, from the properties of the binomial distribution (Problem 10.1) we have

$$E[k_i] = N_i P_i \tag{10.4}$$

which leads to

$$E[\hat{P}] = \sum_{i=1}^{M} P(\omega_i)P_i \equiv P \tag{10.5}$$

that is, the true error probability. To compute the respective variance of the estimator, we recall from Problem 10.1 that

$$\sigma_{k_i}^2 = N_i P_i(1-P_i) \tag{10.6}$$

leading to

$$\sigma_{\hat{P}}^2 = \sum_{i=1}^{M} P^2(\omega_i) \frac{P_i(1-P_i)}{N_i} \tag{10.7}$$

Thus, the error probability estimator in (10.3), which results from simply counting the errors, is unbiased but only asymptotically consistent as $N_i \to \infty$. *Thus, if small data sets are used for testing the performance of a classifier, the resulting estimate may not be reliable.*

In [Guyo 98] the minimum size of the test data set, N, is derived in terms of the true error probability P of the already designed classifier. The goal is to estimate N so that to guarantee, with probability $1-a, 0 \leq a \leq 1$, that P does not exceed the estimated from the test set, $\hat{P}$, by an amount larger than $\epsilon(N,a)$, that is

$$\text{prob}\{P \geq \hat{P} + \epsilon(N,a)\} \leq a \tag{10.8}$$

Let $\epsilon(N,a)$ be expressed as a function of P, that is, $\epsilon(N,a) = \beta P$. An analytical solution for Eq. (10.8) with respect to N is not possible. However, after some approximations certain bounds can be derived. For our purposes, it suffices to consider a simplified formula, which is valid for typical values of a and β ($a = 0.05, \beta = 0.2$),

$$N \approx \frac{100}{P} \tag{10.9}$$

In words, if we want to guarantee, with a risk a of being wrong, that the error probability P will not exceed $\frac{\hat{P}}{1-\beta}$, then N must be of the order given in Eq. (10.9). For $P = 0.01, N = 10{,}000$ and for $P = 0.03, N = 3000$. Note that this result is independent of the number of classes. Furthermore, if the samples in the test data set are not independent, this number must be further increased. Such bounds are also of particular importance, if the objective is to determine the size N of the test data set that provides good confidence in the results, when comparing different classification systems with relatively small differences in their error probabilities.

Although the error-counting approach is by far the most popular one, other techniques have also been suggested in the literature. These techniques estimate the error probability by using smoother versions of the discriminant function(s) realized by the classifier. The error-counting approach can be thought of as an extreme case of a hard limiter, where a 1 or 0 is produced and counted, depending on the discriminant function's response, that is, whether it is false or true, respectively. See, for example, [Raud 91, Brag 04].

10.3 EXPLOITING THE FINITE SIZE OF THE DATA SET

The estimation of the classification error probability presupposes that one has decided upon the data set to which the error counting will be applied. This is

not a straightforward task. The set of samples that we have at our disposal is finite, and it has to be utilized for both training and testing. Can we use the same samples for training and testing? If not, what are the alternatives? Depending on the answer to the question, the following methods have been suggested:

- *Resubstitution Method:* The same data set is used, first for training and then for testing. One need not go into mathematical details in order to see that such a procedure is not very fair. Indeed, this is justified by the mathematical analysis. In [Fole 72] the performance of this method was analyzed using normal distributions. The analysis results show that this method provides an *optimistic* estimate of the true error probability. The amount of bias of the resubstitution estimate is a function of the ratio N/l, that is, the data set size and the dimension of the feature space. Furthermore, the variance of the estimate is inversely proportional to the data set size N. In words, *in order to obtain a reasonably good estimate, N as well as the ratio N/l must be large enough*. The results from the analysis and the related simulations show that N/l should be at least three and that an upper bound of the variance is $1/8N$. Of course, if this technique is to be used in practice, where the assumptions of the analysis are not valid, experience suggests that the suggested ratio must be even larger [Kana 74]. Once more, the larger the ratio N/l, the more comfortable one feels.

- *Holdout Method:* The available data set is divided into two subsets, one for training and one for testing. The major drawback of this technique is that it reduces the size for both the training and the testing data. Another problem is to decide how many of the N available data will be allocated to the training set and how many to the test set. This is an important issue. In Section 3.5.3 of Chapter 3, we saw that designing a classifier using a finite data set introduces an excess mean error and a variance around it, as different data sets, of the same size, are used for the design. Both of these quantities depend on the size of the training set. In [Raud 91], it is shown that the classification error probability of a classifier, designed using a finite training data set, N, is always higher than the corresponding asymptotic error probability ($N \to \infty$). This excess error decreases as N increases. On the other hand, in our discussion in the previous section we saw that the variance of the error counting depends on the size of the test set, and for small test data sets the estimates can be *unreliable*. Efforts made to optimize the respective sizes of the two sets have not yet led to practical results.

- *Leave-One-Out Method:* This method [Lach 68] alleviates the lack of independence between the training and test sets in the resubstitution method and at the same time frees itself from the dilemma associated with the holdout method. The training is performed using $N-1$ samples, and the test is carried out using the excluded sample. If this is misclassified, an error is counted. This is repeated N times, each time excluding a *different* sample. The total

number of errors leads to the estimation of the classification error probability. Thus, training is achieved using, basically, all samples, and at the same time independence between training and test sets is maintained. The major disadvantage of the technique is its high computational complexity. For certain types of classifiers (i.e., linear or quadratic) it turns out that a simple relation exists between the leave-one-out and the resubstitution method ([Fuku 90], Problem 10.2). Thus, in such cases the former estimate is obtained using the latter method with some computationally simple modifications.

The estimates resulting from the holdout and leave-one-out methods turn out to be very similar, for comparable sizes of the test and training sets. Furthermore, it can be shown (Problem 10.3, [Fuku 90]) that the holdout error estimate, for a Bayesian classifier, is an upper bound of the true Bayesian error. In contrast, the resubstitution error estimate is a lower bound of the Bayesian error, confirming our previous comment that it is an optimistic estimate. To gain further insight into these estimates and their relation, let us make the following definitions:

- P_e^N denotes the classification error probability for a classifier designed using a finite set of N training samples.
- $\bar{P}_e^N$ denotes the average $E[P_e^N]$ over all possible training sets of size N.
- P_e is the average asymptotic error as $N \to \infty$.

It turns out that the holdout and leave-one-out methods (for statistically independent samples) provide an *unbiased* estimate of $\bar{P}_e^N$. In contrast, the resubstitution method provides a biased (underestimated) estimate of $\bar{P}_e^N$. Figure 10.1 shows the trend of a typical plot of $\bar{P}_e^N$ and the average (over all possible sets of size N) resubstitution error as functions of N [Fole 72, Raud 91]. It is readily observed that as the data size N increases, both curves tend to approach the asymptotic P_e.

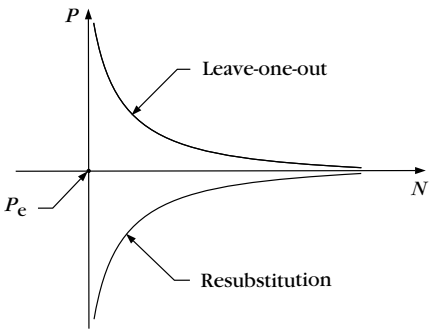


FIGURE 10.1

Plots indicating the general trend of the average resubstitution and leave-one-out error probabilities as functions of the number of training points.

A number of variations and combinations of these basic schemes have also been suggested in the literature. For example, a variation of the leave-one-out method is to leave $k > 1$, instead of one, samples out. The design and test process is repeated for all distinct choices of k samples. References [Kana 74, Raud 91] are two good examples of works discussing various aspects of the topic.

In [Leis 98] a method called *cross-validation with active pattern selection* is proposed, with the goal of reducing the high computational burden required by the leave-one-out method. It is suggested not to leave out (one at a time) all N feature vectors, but only $k < N$. To this end the "good" points of the data set (expected to contribute a 0 to the error) are not tested. Only the k "worst" points are considered. The choice between "good" and "bad" is based on the respective values of the cost function after an initial training. This method exploits the fact that the outputs of the classifier, trained according to the least squares cost function, approximate posterior probabilities, as discussed in Chapter 3. Thus, those feature vectors whose outputs have a large deviation from the desired value (for the true class) are expected to be the ones that contribute to the classification error.

Another set of techniques have been developed around the *bootstrap method* [Efro 79, Hand 86, Jain 87]. A major incentive for the development of these techniques is the variance of the leave-one-out method estimate for small data sets [Efro 83]. According to the "bootstrap" philosophy, new data sets are artificially generated. This is a way to overcome the limited number of available data and create more data in order to better assess the statistical properties of an estimator. Let X be the set of the available data of size N. A *bootstrap design sample set of size* N, X^*, is formed by *random* sampling with *replacement* of the set X. Replacement means that when a sample, say $\boldsymbol{x}_i$, is "copied" to the set X^*, it is not removed from X but is reconsidered in the next sampling. A number of variants have been built upon the bootstrap method. A straightforward one is to design the classifier using a bootstrap sample set and count the errors using the samples from X that do not appear in this bootstrap sample set. This is repeated for different bootstrap sample sets. The error rate estimate, e_0, is computed by counting all the errors and dividing the sum by the total number of test samples used. However, in [Raud 91] it is pointed out that the bootstrap techniques improve on the leave-one-out method only when the classification error is large.

Another direction is to combine estimates from different estimators. For example, in the so-called 0.632 estimator ([Efro 83]), the error estimate is taken as a *convex* combination of the resubstitution error, e_{res}, and the bootstrap error e_0,

$$e_{0.632} = 0.368 e_{res} + 0.632 e_0$$

It has been reported that the 0.632 estimator is particularly effective in cases of small size data sets [Brag 04]. An extension of the 0.632 rule is discussed in [Sima 06] where convex combinations of different estimators are considered and the combining weights are computed via an optimization process.

Confusion Matrix, Recall and Precision

In evaluating the performance of a classification system, the probability of error is sometimes not the only quantity that assesses its performance sufficiently. Let us take for example, an M-class classification task. An important issue is to know whether there are classes that exhibit a higher tendency for confusion. The *confusion matrix* $A = [A(i,j)]$ is defined so that its element $A(i,j)$ is the number of data points whose true class label was i and were classified to class j. From A, one can directly extract the *recall* and *precision* values for each class, along with the overall accuracy:

- *Recall* (R_i). R_i is the percentage of data points with true class label i, which were correctly classified in that class. For example, for a two-class problem, the recall of the first class is calculated as $R_1 = \frac{A(1,1)}{A(1,1)+A(1,2)}$.
- *Precision* (P_i). P_i is the percentage of data points classified as class i, whose true class label is indeed i. Therefore, for the first class in a two-class problem, $P_1 = \frac{A(1,1)}{A(1,1)+A(2,1)}$.
- *Overall Accuracy* (Ac). The overall accuracy, Ac, is the percentage of data that has been correctly classified. Given an M-class problem, Ac is computed from the confusion matrix according to the equation $Ac = \frac{1}{N}\sum_{i=1}^{M} A(i,i)$, where N is the total number of points in the test set.

Take as an example a two-class problem where the test set consists of 130 points from class ω_1 and 150 points from class ω_2. The designed classifier classifies 110 points from ω_1 correctly and 20 points to class ω_2. Also, it classifies 120 points from class ω_2 correctly and 30 points to class ω_1. The confusion matrix for this case is

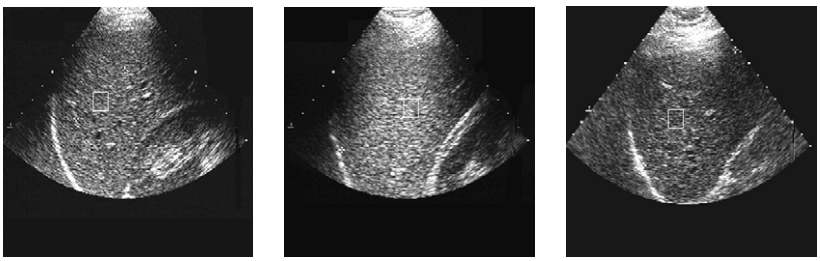
$$A = \begin{bmatrix} 110 & 20 \\ 30 & 120 \end{bmatrix}$$

The recall for the first class is $R_1 = \frac{110}{130}$ and the precision $P_1 = \frac{110}{140}$. The respective values for the second class are similarly computed. The accuracy is $Ac = \frac{110+120}{130+150}$.

10.4 A CASE STUDY FROM MEDICAL IMAGING

Our goal in this section is to demonstrate the various design stages, discussed in the previous chapters, via a case study borrowed from a real application. It will not come as a surprise to say that focusing on a single example cannot cover all possible design approaches that are followed in practice. However, our aim is to provide a flavor for the newcomer. After all, "perfection is the enemy of the good."

Our chosen application comes from the medical imaging discipline. Our task is to develop a pattern recognition system for the diagnosis of certain liver diseases. Specifically, the system will be presented with *ultrasound* images of the



(a) (b) (c)

FIGURE 10.2

Ultrasound images corresponding to (a) normal liver, (b) liver with fatty infiltration, and (c) liver with cirrhosis. The square shows the image area on which the analysis was carried out.

liver, and it must be able to recognize *normal* from *abnormal* cases. Abnormal cases correspond to two types of liver diseases, namely, *cirrhosis* and *fatty liver infiltration*. For each case, two different gratings must be recognized, depending on the degree of the disease development [Cavo 97]. Figure 10.2 shows three examples of ultrasound images corresponding to (a) a normal liver, (b) an abnormal liver with fatty infiltration, and (c) an abnormal liver with cirrhosis. It is readily realized that the visual differences between the images are not great. This makes the clinical diagnosis and the diagnostic accuracy very much dependent on the skill of the doctor. Thus, the development of a pattern recognition system can assist the doctor in assessing the case and, together with other clinical findings, reduce the need for invasive techniques (biopsy).

The first stage in the design process involves the close cooperation of the system designer with the specialist, that is, the doctor, in order to establish a "common language" and have the designer *understand* the task and define, in common with the doctor, the goals and requirements of the pattern recognition system. Besides the acceptable error rate, other performance issues come into play, such as complexity, computational time, and cost of the system. The next stage involves various image processing steps, such as image enhancement, in order to assist the system by presenting it only useful information as much as possible. Then things are ripe to begin with the design of the pattern recognition system.

Figure 10.3 outlines the task. There are five possible classes. The pattern recognition system can be designed either around a single classifier, which assigns an unknown image directly to one of the five classes, or around a number of classifiers built on a tree structure philosophy. The latter approach was adopted here. Figure 10.4 illustrates the procedure. A separate classifier was used at each node, and each of them performs a two-class decision. At the first node, the respective classifier decides between normal and abnormal cases. At the second node, images, classified as abnormal, are tested and classified in either the cirrhosis or the fatty liver infiltration class, and so on. The advantage of such a procedure is that we break the problem into a number of simpler ones. It must be stressed, however, that in

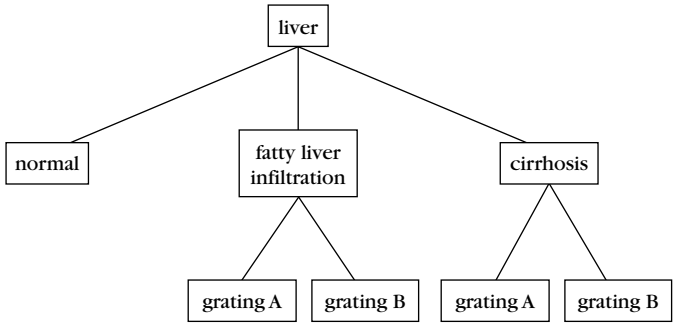


FIGURE 10.3
The classification task.

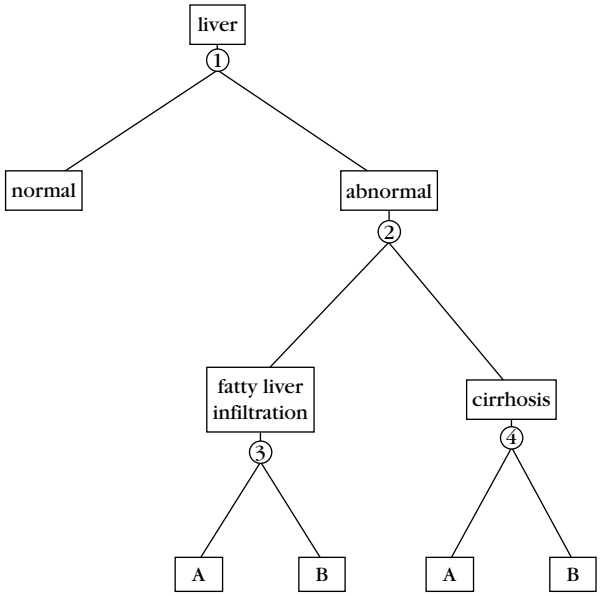


FIGURE 10.4
A tree-structured hierarchy of classifiers.

other applications such a procedure may not be applicable. For the design of the classification system, 150 ultrasound liver images were obtained from a medical center. Fifty of them correspond to normal cases, 55 of them to patients suffering from cirrhosis, and 45 of them to patients suffering from fatty liver infiltration. Three classifiers were adopted for comparison, namely the least squares linear classifier, the minimum Euclidean distance classifier, and the kNN for different values of k. Each time, the same type of classifier was used for all nodes. *From the discussions*

with the specialists, we concluded that what was of interest here was the texture of the respective images. The methods described in Section 7.2.1 of Chapter 7 were used, and a total of 38 features were generated for each of the images. This is a large number, and a feature selection procedure was "mobilized" to reduce this number. Let us first concentrate on the first node classification task and the LS linear classifier.

- For each of the 38 features the t-test was applied, and only 19 of them passed the test at a significance level of 0.001. The latter is chosen so that "enough" features pass the test. Taking into account the size of the problem, enough was considered to be around 15 for our problem. However, 19 is still a large number, and a further reduction was considered necessary. For example, 19 is of the same order as 50 (the number of normal patterns), which would lead to poor generalization.
- The 19 features were considered in pairs, in triples, up to groups of seven, in all possible combinations. For each combination, the optimal LS classifier was designed and each time the corresponding classification error rate was estimated, using the leave-one-out method. It turned out that taking the features in groups larger than two did not improve the error rate significantly. Thus, it was decided that $l = 2$ was satisfactory and that the best combination consisted of the *kurtosis* and the ASM. The percentage of correct classification with this combination was 92.5%.

For the design of the linear classifier of "node 2" the same procedure was followed, using, of course, only the images originating from abnormal cases. Of the 38 originally produced features, only 15 passed the t-test. The optimal combination of features turned out to be the mean, the variance, and the correlation. It may be worth pointing out that the variance was rejected from the t-test during the design of the "node 1" classifier. The percentage of correct classification for node 2 was 90.1%. The optimal combination for the "node 3" LS classifier was the variance, entropy, the sum entropy, and the difference entropy corresponding to a correct classification rate of 92.2%. Finally, the optimization procedure for the "node 4" classifier resulted in the mean value, the ASM, and the contrast with a correct classification rate estimate of 83.8%.

Having completed the design with the LS linear classifiers, the same procedure was followed for the Euclidean minimum distance classifier and the kNN classifier. However, in both of these cases the resulting error rate estimates were always higher than the ones obtained with the LS classifier. Thus, the latter one was finally adopted.

Once more, it must be stated that this case study does not and cannot represent the wealth of classification tasks encountered in practice, each with its own specific requirements. We could state, with a touch of exaggeration, of course, that each classification task is like a human being. *Each one has its own personality*! For example, the dimension of our problem was such that it was computationally feasible, with today's technology to follow the procedure described. The feature

selection, classifier design, and classification error stages were combined to compute the best combination. This was also a motivation for choosing the specific case study, that is, to demonstrate that the various stages in the design of a classification system are not independent but they can be closely interdependent. However, this may not be possible for a large number of tasks, as, for example, the case of a large multilayer neural network in a high-dimensional feature space. Then the feature selection stage cannot be easily integrated with that of classifier design, and techniques such as those presented in Chapter 5 must be employed. Ideally, what one should aim at is to have a procedure to design the classifiers by minimizing the error probability directly (not the LS, etc.), and at the same time this procedure should be computationally simple (!) to allow also for a search for the optimal feature combination. However, this "utopia" is still quite distant.

10.5 SEMI-SUPERVISED LEARNING

All the methods that we have considered in the book so far have relied on using a set of *labeled* data for the training of an adopted model structure (classifier). The final goal was, always, to design a "machine," which, after the training phase, can predict *reliably* the labels of unseen points. In other words, the scope was to develop a *general rule* based on the *inductive inference* rationale. In such a perspective, the *generalization* performance of the designed classifier was a key issue that has "haunted" every design methodology.

In this section, we are going to relax the design procedure from both "pillars" on which all our methods were so far built; (a) the labeled data set used for the training and (b) our concern about the generalization performance of the developed classifier. Initially, unlabeled data will be brought into the game, and we will investigate the possibility of whether this extra information, in conjunction with the labeled data, can offer performance improvement. Moving on, in a later stage, we will consider cases where the classifier design is not focused on predicting the labels of "future" unseen data points. In contrast, the optimization of a loss function will entirely rely on best serving the needs of a *given set* of unlabeled data, which are at the designer's disposal "now", that is, at the time of the design. The latter concept of designing a classifier is known as *transductive inference* to contrast it to the inductive inference mentioned earlier.

Designing classifiers by exploiting information that resides in both labeled and unlabeled data springs from a fact of life; that is, in many real applications collecting unlabeled data is much easier than the task of labeling them. In a number of cases, the task of labeling is time consuming, and it requires annotation by an expert. Bioinformatics is a field in which unlabeled data is abundant, yet only a relatively small percentage is labeled, as, for example in protein classification tasks. Text classification is another area where unlabeled data is fairly easy to collect while the labeling task requires the involvement of an expert. Annotating music is also a very

demanding task, which, in addition, involves a high degree of subjectivity, as, for example, in deciding the genre of a music piece. On the other hand, it is very easy to obtain unlabeled data.

Figures 10.5 and 10.6, inspired by [Sind 06], present two simple examples raising expectations that performance may be boosted by exploiting additional information that resides in an unlabeled data set. In Figure 10.5a we are given two labeled points and one, denoted by "?", whose class is unknown. Based on this limited information, one will readily think that the most sensible decision is to classify the unknown point to the "*" class. In Figure 10.5b, in addition to the previous three points, a set of unlabeled points is shown. Having this more complete picture, one will definitely be tempted to reconsider the previous decision. In this case,

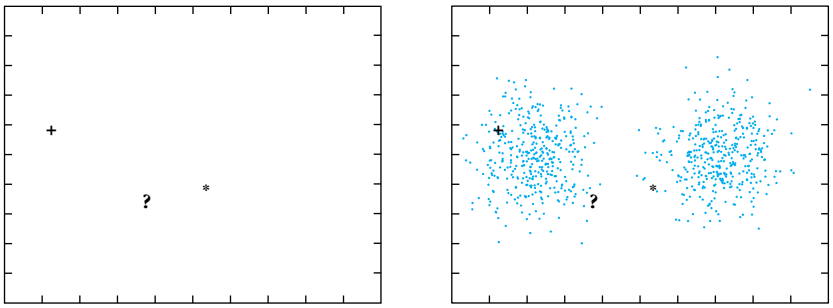


FIGURE 10.5

(a) The unknown point, denoted by "?", is classified in the same class as point "∗". (b) The setup after a number of unlabeled data have been provided, which leads us to reconsider our previous classification decision.

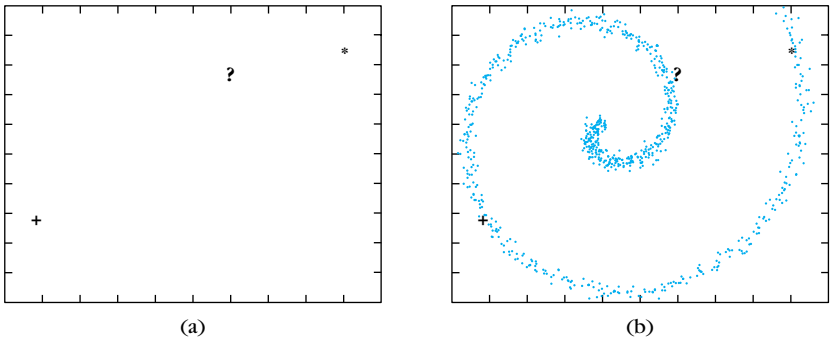


FIGURE 10.6

(a) The unknown point, denoted by "?", is classified in the same class as point "∗". (b) The setup after a number of unlabeled data have been provided. The latter forces us, again, to reconsider our previous classification decision.

the extra information unveiled by the unlabeled data, and used by our perceptive mechanism, is the *clustered structure* of the data set. Figure 10.6 provides us with a slightly different viewpoint. Once more, we are given the three points, and the same decision as before is drawn (Figure 10.6a). In Figure 10.6b, the unlabeled data reveal a *manifold structure* on which the points reside (see also Section 6.6). The extra piece of information, which is disclosed to us now, is that the unknown point is closer to the "+" than to the "∗" point, if the geodesic, instead of the Euclidean, distance is used. Of course, in both cases, reconsideration of our initial decision is justified only under the following assumptions:

- *Cluster assumption*: If two points are in the same cluster, they are likely to originate from the same class.
- *Manifold assumption*: If the marginal probability distribution, $p(\boldsymbol{x})$, is supported on a manifold, then points lying close to each other on the manifold are more likely to be in the same class. Another way to express this is that the conditional probability, $P(y|\boldsymbol{x})$, of the class label y, is a smooth function of $\boldsymbol{x}$, with respect to the underlying structure of the manifold.

Figure 10.5 illustrates the cluster assumption, and Figure 10.6 the manifold one.

Both assumptions can be seen as particular instances of a more general assumption that covers both classification and regression:

- *Semi-supervised assumption*: If two points are close in a high-density region, then their corresponding outputs should have close values.

In other words, closeness between points is not a decisive factor, if, considered by itself. It has to be considered in the context of the underlying data distribution. This is apparent from the previous two figures. According to the semi-supervised smoothness assumption, if two points are close and linked by a path through a high-density area, they are likely to give closely located outputs. On the other hand, if the path that links them goes through a low-density region, there is no need for the corresponding outputs to be close ([Chap 06a, p. 5]).

Although semi-supervised learning has attracted a lot of interest recently, it is not new as a problem. Semi-supervised learning has been addressed in the statistics community as early as the mid-1960s, for example, [Scud 65]. Transductive learning was introduced by Vapnik and Chervonenkis in the mid-1970s, [Vapn 74]. Over the years, a large number of approaches and algorithms have been proposed, and it is beyond the scope of the present section to cover the area in depth and in its entire breadth. Our goal is to present some of the basic directions that are currently popular and at the same time are based on methods previously addressed in this book.

10.5.1 Generative Models

Generative models are perhaps the oldest semi-supervised methods and they have been used in statistics for many years. The essence behind these methods is to

model the class-conditional densities $p(\boldsymbol{x}|y)$, using information provided by both labeled and unlabeled data. Once such a model is available, one can compute the marginal distribution $p(\boldsymbol{x})$

$$p(\boldsymbol{x}) = \sum_{y} P(y)p(\boldsymbol{x}|y) \tag{10.10}$$

the joint distribution

$$p(y, \boldsymbol{x}) = P(y)p(\boldsymbol{x}|y) \tag{10.11}$$

and finally the quantity that is required by the Bayesian classifier

$$P(y|\boldsymbol{x}) = \frac{P(y)p(\boldsymbol{x}|y)}{\sum_{y} P(y)p(\boldsymbol{x}|y)} \tag{10.12}$$

The class label is an integer, $y \in \{1, 2, \ldots, M\}$, where M is the number of classes. If $P(y)$ is not known then an estimate of it is used. The above formulas are familiar to us from Chapter 2, but they are repeated here for the sake of completeness.

In the sequel, we adopt a parametric model for the class conditional densities, that is, $p(\boldsymbol{x}|\, y; \boldsymbol{\theta})$. Let also $P_y,\ y = 1, 2, \ldots, M$, denote the respective estimates of the class priors $P(y)$. Assume that we are given two types of data sets:

- *Unlabeled data*: This data set consists of N_u samples $\boldsymbol{x}_i \in \mathcal{R}^l,\ i = 1, 2, \ldots, N_u$, which are assumed to be independently and identically distributed random vectors drawn from the marginal distribution $p(\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{P})$, which is also parameterized in terms of $\boldsymbol{\theta}$, and $\boldsymbol{P} = [P_1, P_2, \ldots, P_M]^T$. The corresponding set is denoted by D_u.
- *Labeled data*: We assume that N_l samples are randomly and independently generated and they are *subsequently* labeled by an expert. Let N_y of them be associated with class $y = 1, 2, \ldots, M$, where $N_l = \sum_y N_y$. We adopt the notation $\boldsymbol{z}_{iy},\ i = 1, 2, \ldots, N_y,\ y = 1, 2, \ldots, M$, to represent the ith sample assigned in the yth class. The set of labeled samples is denoted as $D_l = \{\boldsymbol{z}_{iy}, i = 1, 2, \ldots, N_y,\ y = 1, 2, \ldots, M\}$. This type of labeling data matches best a number of practical applications. For example, in medical imaging an expert is given a set of images, which have been previously produced, and labels them accordingly. Other "mechanisms" of generating labeled data are also possible, by adopting different assumptions, see [Redn 84, Shas 94, Mill 97, Mill 03].

Our task now is to estimate the set of the unknown parameters, that is, $\boldsymbol{\Theta} \equiv [\boldsymbol{\theta}^T, \boldsymbol{P}^T]^T$ in the mixture model (see Section 2.5.5 of Chapter 2.)

$$p(\boldsymbol{x}; \boldsymbol{\Theta}) = \sum_{y=1}^{M} P_y p(\boldsymbol{x}|y; \boldsymbol{\theta}) \tag{10.13}$$

using the observations in D_u and D_l. For simplicity, in the previous mixture model we have assumed one mixture component per class. This can be relaxed,

for example, [Mill 97]. If only D_u was available, then the task would reduce to the mixture modeling task with hidden class (mixture) labels, as discussed in Section 2.5.5.

It is known from statistics and it is readily deduced from the definition of the log-likelihood in Section 2.5, that if the set of observations is the union of two independent sets then the log-likelihood is the sum of the log-likelihoods of the respective sets. In our case the following are valid (see also [Redn 84]):

$$D_u: \quad L_u(\boldsymbol{\Theta}) = \sum_{i=1}^{N_u} \ln p(\boldsymbol{x}_i; \boldsymbol{\Theta}) = \sum_{i=1}^{N_u} \ln \sum_{y=1}^{M} P_y p(\boldsymbol{x}_i|y; \boldsymbol{\theta}) \tag{10.14}$$

$$\begin{aligned} D_l: \quad L_l(\boldsymbol{\Theta}) &= \sum_{y=1}^{M} \sum_{i=1}^{N_y} \ln p(y, \boldsymbol{z}_{iy}; \boldsymbol{\Theta}) + \ln \frac{N_l!}{N_1! N_2! \ldots N_M!} \\ &= \sum_{y=1}^{M} \sum_{i=1}^{N_y} \ln \left(P_y p(\boldsymbol{z}_{iy}|y; \boldsymbol{\theta}) \right) + \ln \frac{N_l!}{N_1! N_2! \ldots N_M!} \end{aligned} \tag{10.15}$$

Note that in the case of labeled data the "full" observations of the joint events $(y, \boldsymbol{z}_{iy})$ are made available to us. The second term in the log-likelihood function results from the generalized Bernoulli theorem [Papo 02, p.110]. This is a consequence of the way labeled samples were assumed to occur. Basically, we are given N_l random samples and after the labeling N_1 of them are assigned to class $y = 1$, N_2 of them to class $y = 2$ and so on. However, this term is independent of $\boldsymbol{\theta}$ and $\boldsymbol{P}$ and, in practice, is neglected. The unknown set of parameters $\boldsymbol{\theta}$, $\boldsymbol{P}$ can now be obtained by maximizing the sum $L_u(\boldsymbol{\Theta}) + L_l(\boldsymbol{\Theta})$ with respect to $\boldsymbol{\theta}$ and $\boldsymbol{P}$. Due to the nature of $L_u(\boldsymbol{\Theta})$ optimization has to be carried out in the framework discussed in Section 2.5.5. The EM algorithm is the most popular alternative toward this end.

In order to get a feeling on how the presence of labeled data affects the results of the EM algorithm, when compared with the case where only unlabeled data are used, let us consider the example of Section 2.5.5, where the conditional densities were assumed to be Gaussians, that is,

$$p(\boldsymbol{x}|y; \boldsymbol{\theta}) = \frac{1}{\left(2\pi\sigma_y^2\right)^{l/2}} \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{\mu}_y\|^2}{2\sigma_y^2} \right) \tag{10.16}$$

where $\boldsymbol{\theta} = [\boldsymbol{\mu}_1^T, \ldots, \boldsymbol{\mu}_M^T, \sigma_1^2, \ldots, \sigma_M^2]^T$. The E-step now becomes:

E-step:

$$\begin{aligned} Q(\boldsymbol{\Theta}; \boldsymbol{\Theta}(t)) = &\sum_{i=1}^{N_u} \sum_{y=1}^{M} P(y|\boldsymbol{x}_i; \boldsymbol{\Theta}(t)) \ln \left(p(\boldsymbol{x}_i|y; \boldsymbol{\mu}_y, \sigma_y^2) P_y \right) \\ &+ \sum_{i=1}^{N_l} \ln \left(p(\boldsymbol{z}_{iy}|y; \boldsymbol{\mu}_y, \sigma_y^2) P_y \right) \end{aligned} \tag{10.17}$$

In words, the expectation operation is required for the unlabeled samples only, since for the rest the corresponding labels are known. Using similar steps as in Problem 2.31 and considering both log-likelihood terms, recursions (2.98), (2.99) and (2.100) are modified as follows:

M-step:

$$\boldsymbol{\mu}_y(t+1) = \frac{\sum_{i=1}^{N_u} P(y|\boldsymbol{x}_i; \boldsymbol{\Theta}(t))\boldsymbol{x}_i + \sum_{i=1}^{N_y} \boldsymbol{z}_{iy}}{\sum_{i=1}^{N_u} P(y|\boldsymbol{x}_i; \boldsymbol{\Theta}(t)) + N_y} \tag{10.18}$$

$$\sigma_y^2(t+1) = \frac{\sum_{i=1}^{N_u} P(y|\boldsymbol{x}_i; \boldsymbol{\Theta}(t))\|\boldsymbol{x}_i - \boldsymbol{\mu}_y(t+1)\|^2}{l\left(\sum_{i=1}^{N_u} P(y|\boldsymbol{x}_i; \boldsymbol{\Theta}(t)) + N_y\right)}$$

$$+ \frac{\sum_{i=1}^{N_y} \|\boldsymbol{z}_{iy} - \boldsymbol{\mu}_y(t+1)\|^2}{l\left(\sum_{i=1}^{N_u} P(y|\boldsymbol{x}_i; \boldsymbol{\Theta}(t)) + N_y\right)} \tag{10.19}$$

$$P_y(t+1) = \frac{1}{N_u + N_l}\left(\sum_{i=1}^{N_u} P(y|\boldsymbol{x}_i; \boldsymbol{\Theta}(t)) + N_y\right) \tag{10.20}$$

Remarks

- Provided that the adopted mixture model for the marginal density is correct, the use of unlabeled data is guaranteed to improve performance, for example, [Cast 96]. However, if this is not the case, and the adopted model does not match the characteristics of the true distribution that generates the data, incorporating unlabeled data may actually degrade the performance. This is a very important issue, since in practice it may not be an easy task to have good knowledge about the exact nature of the underlying distribution. This claim has been supported by a number of researchers, and a theoretical justification is provided in [Cohe 04].
- Looking at Eqs. (10.14) and (10.15), we observe that if $N_u >> N_l$, which is usually the case in practice, the unlabeled data term is the dominant one. This is also clear by inspecting the recursion in (10.18)–(10.20). To overcome this, an appropriate weighting of the two log-likelihood terms may be used, for example, [Cord 02, Niga 00]. Another problem associated with the EM algorithm is, as we already know, that it can be trapped in a local maximum. This can also be a source for performance degradation when using unlabeled data. This is treated in [Niga 00].

10.5.2 Graph-Based Methods

In any classification task, the ultimate goal is to predict the class label given the observation $\boldsymbol{x}$. In the generative modeling, the philosophy is to model the "generation" mechanism of the data and also to adopt a model for $p(\boldsymbol{x}|y)$, which then implies

all the required information, that is, $p(\boldsymbol{x}), p(y, \boldsymbol{x}), P(y|\boldsymbol{x})$. However, throughout this book, the majority of the methods we dealt with were developed on a different rationale. If all we need is to infer the class labels, let us model the required information *directly*. As Vapnik stated:

> *When solving a given problem, try to avoid solving a more general one as an intermediate step.*

For example, if the densities underlying the classes are Gaussians with the same covariance matrix, one need not bother to estimate the covariance parameters; exploiting the fact that the optimum discriminant function is linear can be sufficient to design a good classifier [Vapn 99]. Such techniques are known as *diagnostic or discriminative* methods. Linear classifiers, backpropagation neural networks, and support vector machines are typical examples that fall under the diagnostic design methodology. In all these methods, the marginal probability density, $p(\boldsymbol{x})$, was not considered explicitly for the estimation of the corresponding optimal parameters. The obvious question that now arises is whether and how such techniques can benefit from the existence of unlabeled data. The latter "express" themselves via $p(\boldsymbol{x})$. On the other hand, the marginal probability density does not enter into the discriminative models explicitly. The way out comes through *penalization*, where one forces the solution to respect certain *general characteristics* of $p(\boldsymbol{x})$. Typical such characteristics, which have been exploited in semi-supervised learning, are: (a) the clustering structure that may underlie the data distribution and (b) the manifold geometry on which the data might lie. This information can be embedded in the form of *regularization* in the optimization of a loss function associated with the classification task (see Section 4.19).

Graph methods fall under the diagnostic design approach, and a number of techniques have been proposed to exploit classification-related information that resides in the data distribution. In order to present the basic rationale behind graph methods, we will focus on a technique that builds around the manifold assumption. This technique also fits nicely with a number of concepts discussed in previous chapters in the book.

As we have already seen in Section 6.7.2, graph methods start with the construction of an undirected graph $G(V, E)$. Each node, v_i, of the graph corresponds to a data point, $\boldsymbol{x}_i$, and the edges connecting nodes, e.g., v_i, v_j, are weighted by a weight $W(i, j)$ that quantifies *similarity* between the corresponding points, $\boldsymbol{x}_i$, $\boldsymbol{x}_j$. There was discussed how these weight values can be used to provide information related to the local structure of the underlying manifold—that is, *the intrinsic geometry* associated with $p(\boldsymbol{x})$.

Assume that we are given a set of N_l labeled points $\boldsymbol{x}_i$, $i = 1, 2 \ldots, N_l$, and a set of N_u unlabeled points $\boldsymbol{x}_i$, $i = N_l + 1, \ldots, N_l + N_u$. Our kickoff point is Eq. (4.79)

$$\sum_{i=1}^{N_l} \mathcal{L}\left(g(\boldsymbol{x}_i), y_i\right) + ||g||_H^2 \tag{10.21}$$

where H is used explicitly to denote that the norm of the regularizer is taken in the RKHS space, and we have assumed $\Omega(\cdot)$ to be a square one. The loss function is considered over the labeled data only. In [Belk 04, Sind 06], it is suggested to adding an extra regularization term that *reflects the intrinsic structure* of $p(\boldsymbol{x})$. Using some differential geometry arguments, which are not of interest to us here, it turns out that a quantity that approximately reflects the underlying manifold structure is related to the Laplacian matrix of the graph (see also Section 6.7.2). The proposed in [Belk 04] optimization task is:

$$\arg\min_{g\in H} \frac{1}{N_l}\sum_{i=1}^{N_l}\mathcal{L}\left(g(\boldsymbol{x}_i), y_i\right) + \gamma_H ||g||_H^2 +$$

$$\frac{\gamma_I}{(N_l+N_u)^2}\sum_{i,j=1}^{N_l+N_u}\left(g(\boldsymbol{x}_i)-g(\boldsymbol{x}_j)\right)^2 W(i,j) \tag{10.22}$$

Observe that two normalizing constants are present in the denominators and account for the number of points contributing to each of the two data terms. The parameters γ_H, γ_I control the relative significance of the two terms in the objective function. Also note that in the last term all points, labeled as well as unlabeled, are considered. For those who do not have "theoretical anxieties," it suffices to understand that the last term in the cost accounts for the local geometry structure. If two points are far apart, the respective $W(i,j)$ is small so their contribution to the cost is negligible. On the other hand, if the points are closely located, $W(i,j)$ is large, and these points have an important "say" in the optimization process. This means that the demand (through the minimization task) of nearby points to be mapped to similar values (i.e., $g(\boldsymbol{x}_i) - g(\boldsymbol{x}_j)$ to be small) will be seriously taken into account. This is basically a smoothness constraint, which is in line with the essence of the manifold assumption stated before. Using similar arguments as in Section 6.7.2, we end up with the following optimization task

$$\arg\min_{g\in H} \frac{1}{N_l}\sum_{i=1}^{N_l}\mathcal{L}\left(g(\boldsymbol{x}_i), y_i\right) + \gamma_H ||g||_H^2 + \frac{\gamma_I}{(N_l+N_u)^2}\boldsymbol{g}^T L\boldsymbol{g} \tag{10.23}$$

where $\boldsymbol{g} = [g(\boldsymbol{x}_1), g(\boldsymbol{x}_2), \ldots, g(\boldsymbol{x}_{N_l+N_u})]^T$. Recall that the Laplacian matrix is defined as

$$L = D - W$$

where D is the diagonal matrix with elements $D_{ii} = \sum_j^{N_l+N_u} W(i,j)$ and $W = [W(i,j)]$, $i,j = 1,2,\ldots,N_l+N_u$. A most welcome characteristic of this procedure is that the Representer Theorem, discussed in Section 4.19.1, is still valid and the minimizer of (10.23) admits an expansion

$$g(\boldsymbol{x}) = \sum_{j=1}^{N_l+N_u} a_j K(\boldsymbol{x}, \boldsymbol{x}_j) \tag{10.24}$$

where $K(\cdot,\cdot)$ is the adopted kernel function. Observe that the summation is taken over labeled as well as unlabeled points.

In Section 4.19.1, it was demonstrated how use of the Representer Theorem can facilitate the way the optimal solution is sought. This is also true here. Take as an example the case where the loss function is the least squares one that is, $\mathcal{L}\left(g(\boldsymbol{x}_i), y_i\right) = (y_i - g(\boldsymbol{x}_i))^2$. Then it is easy to show (Problem 10.4) that the coefficients in the expansion (10.24) are given by

$$[a_1, a_2, \ldots a_{N_l+N_u}]^T \equiv \boldsymbol{a} = (J\mathcal{K} + \gamma_H N_l I + \frac{\gamma_I N_l}{(N_l+N_u)^2} L\mathcal{K})^{-1}\boldsymbol{y} \tag{10.25}$$

where I is the identity matrix, $\boldsymbol{y} = [y_1, y_2, \ldots, y_{N_l}, 0, \ldots, 0]^T$, J the $(N_l + N_u) \times (N_l + N_u)$ diagonal matrix with N_l entries as 1 and the rest 0, i.e., $J = diag(1, 1, \ldots, 1, 0, \ldots, 0)$ and $\mathcal{K} = [K(i,j)]$ is the $(N_l+N_u)\times(N_l+N_u)$ Gram matrix. Combining (10.25) and (10.24) results in the optimum classifier, employing labeled as well as unlabeled data, given by

$$g(\boldsymbol{x}) = \boldsymbol{y}^T(J\mathcal{K} + \gamma_H N_l I + \frac{\gamma_I N_l}{(N_l+N_u)^2} L\mathcal{K})^{-1}\boldsymbol{p} \tag{10.26}$$

where $\boldsymbol{p} = [K(\boldsymbol{x}, \boldsymbol{x}_1), \ldots, K(\boldsymbol{x}, \boldsymbol{x}_{N_l+N_u})]^T$ (see also Section 4.19.1). The resulting minimizer is known as the *Laplacian regularized kernel least squares* (LRKLS) solution and it can be seen as a generalization of the kernel ridge regressor given in (4.100). Indeed, if $\gamma_I = 0$, unlabeled data do not enter into the game and the last term in the parenthesis becomes zero. Then for $\mathcal{C} = \gamma_H N_l$ we obtain the kernel ridge regressor form (Eq. (4.110)). Note that Eq. (10.26) is the result of a scientific evolution process that spans a period spreading over three centuries. It was Gauss in the nineteenth century who solved for the first term in the parenthesis. The second term was added in the mid-1960s, due to the introduction of the notion of regularized optimization ([Tiho 63, Ivan 62, Phil 62]). To our knowledge, ridge regression was introduced in statistics in [Hoer 70]. The kernelized version was developed in mid-1990s following the work of Vapnik and his coworkers, and the Laplacian "edition" was added in the beginning of this century!

We can now summarize the basic steps in computing the LRKLS algorithm:

Laplacian regularized kernel least squares classifier

- Construct a graph using both labeled and unlabeled points. Choose weights $W(i,j)$ as described in Section 6.7.2.
- Choose a kernel function $K(\cdot,\cdot)$ and compute the Gram matrix $\mathcal{K}(i,j)$.
- Compute the Laplacian matrix $L = D - W$.
- Choose $\gamma_H,\ \gamma_I$.
- Compute $a_1, a_2, \ldots, a_{N_l+N_u}$ from Eq. (10.25).

Given an unknown $\boldsymbol{x}$, compute $g(\boldsymbol{x}) = \sum_{j=1}^{N_l+N_u} a_j K(\boldsymbol{x}, \boldsymbol{x}_j)$. For the two-class classification case the class label, $y \in [+1, -1]$, is obtained by $y = \text{sign}\{g(\boldsymbol{x})\}$.

By changing the cost function and/or the regularization term different algorithms result with different performance trade-offs, for example, [Wu 07, Dela 05, Zhou 04]. Another direction that has been followed within the graph theory framework is what is called *label propagation*, for example, [Zhu 02]. Given a graph, nodes corresponding to the labeled points are assigned their respective class label (e.g., ± 1 for the two-class case) and the unlabeled ones are labeled with a zero. Labels are then propagated in an iterative way through the data set along high-density areas defined by the unlabeled data, until convergence. In [Szum 02] label propagation is achieved by considering Markov random walks on the graph. An interesting point is that the previously discussed two directions to semi-supervised learning, which build on graph theoretic arguments, turn out to be equivalent or, at least, very similar, see, for example, [Beng 06]. Once more, the world is small!

10.5.3 Transductive Support Vector Machines

According to the inductive inference philosophy, one starts from the particular knowledge (training set using labeled data) and then the general rule (the classifier or regressor) is derived, which is subsequently used to predict the labels of *specific* points comprising the test set. In other words, one follows a path

$$\text{particular} \longrightarrow \text{general} \longrightarrow \text{particular}$$

Vapnik and Chervonenkis, pushing the frontiers, questioned whether this is indeed the best path to follow in practice. In cases where the training data set is limited in size, deriving a good general rule becomes a hard task. For such cases, they proposed the *transductive inference* approach, where one follows a "direct" path

$$\text{particular} \longrightarrow \text{particular}$$

In such a way, one may be able to exploit information residing in a *given* test set and obtain improved results. Transductive learning is a special type of semi-supervised learning and the goal is to predict the labels of the points in a *specific* test set by embedding the points of the set, explicitly, in the optimization task. From this perspective, label propagation techniques, discussed before, are also transductive in nature. For a more theoretical treatment of transductive learning, the reader is referred to, for example, [Vapn 06, Derb 03, Joac 02].

In the framework of support vectors machines (see Section 3.7), and for the two class problem, transductive learning is cast as follows. Given the set $D_l = \{\boldsymbol{x}_i, i = 1, 2, \ldots, N_l\}$ of labeled points and the set $D_u = \{\boldsymbol{x}_i, i = N_l + 1, \ldots, N_l + N_u\}$ compute the labels $y_{N_l+1}, \ldots, y_{N_l+N_u}$ of the points in D_u so that the hyperplane that separates the two classes, by taking into consideration *both* labeled and unlabeled

points, has maximum margin. The corresponding optimization tasks for the two versions (hard margin and soft margin) become:

Hard margin TSVM

$$\text{minimize} \quad J(y_{N_l+1},\ldots,y_{N_l+N_u},\boldsymbol{w},w_0)=\frac{1}{2}||\boldsymbol{w}||^2 \tag{10.27}$$

$$\text{subject to} \quad y_i(\boldsymbol{w}^T\boldsymbol{x}_i+w_0)\geq 1,\ i=1,2,\ldots,N_l \tag{10.28}$$

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i+w_0)\geq 1,\ i=N_l+1,\ldots,N_l+N_u \tag{10.29}$$

$$y_i\in\{+1,-1\},\ i=N_l+1,\ldots,N_l+N_u \tag{10.30}$$

Soft margin TSVM

$$\text{minimize} \quad J(y_{N_l+1},\ldots,y_{N_l+N_u},\boldsymbol{w},w_0,\boldsymbol{\xi})=\frac{1}{2}||\boldsymbol{w}||^2+$$

$$C_l\sum_{i=1}^{N_l}\xi_i+C_u\sum_{i=N_l+1}^{N_l+N_u}\xi_i \tag{10.31}$$

$$\text{subject to} \quad y_i(\boldsymbol{w}^T\boldsymbol{x}_i+w_0)\geq 1-\xi_i,\ i=1,2,\ldots,N_l \tag{10.32}$$

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i+w_0)\geq 1-\xi_i,\ i=N_l+1,\ldots,N_l+N_u \tag{10.33}$$

$$y_i\in\{+1,-1\},\ i=N_l+1,\ldots,N_l+N_u \tag{10.34}$$

$$\xi_i\geq 0,\ i=1,2,\ldots,N_l+N_u \tag{10.35}$$

where C_l, C_u are user-defined parameters that control the importance of the respective terms in the cost. In [Joac 99, Chap 05] an extra constraint is used that forces the solution to assign unlabeled data to the two classes in roughly the same proportion as that of the labeled ones.

Figure 10.7 shows a simplified example, for the hard margin case, illustrating that the optimal hyperplane, which results if only labeled examples are used (SVM), is different from the one obtained when both labeled and unlabeled data are employed (TSVM). Performing labeling (of the unlabeled samples), so that the margin between the resulting classes is maximized, pushes the decision hyperplane in sparse regions and it is in line with the *clustering* assumption stated in the beginning of this section.

A major difficulty associated with TSVM is that, in contrast to the convex nature of the standard SVM problem, the optimization is over the labels y_i, $i=N_l+1,\ldots,N_l+N_u$, which are integers in $\{+1,-1\}$ and it is an NP-hard task. To this end, a number of techniques have been suggested. For example, in [DeBi 04] the task is relaxed, and it is solved in the semidefinite programming framework. Algorithms based on coordinate descent searching have been proposed in [Joac 02, Demi 00, Fung 01]. A slight reformulation of the problem is proposed

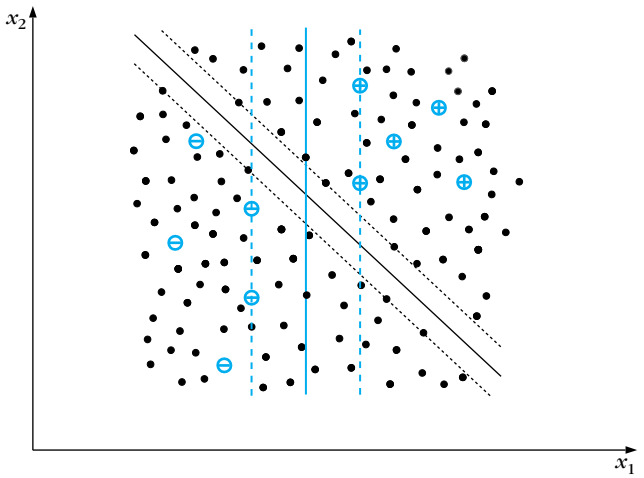


FIGURE 10.7

Red lines correspond to the SVM classifier when only labeled "−" and "+" points are available. The black lines result when the unlabeled data have been considered and have "pushed" the decision hyperplane to an area which is sparse in data.

in [Chap 06]. The constraints associated with the unlabeled data are removed and replaced by $|\boldsymbol{w}^T \boldsymbol{x}_i + w_0| \geq 1 - \xi_i, \ i = N_l + 1, \ldots, N_l + N_u$. Such constraints push the hyperplane away from the unlabeled data, since penalization occurs each time the absolute value becomes less than one. Hence, they are in line with the cluster assumption. This problem formulation has the advantage of removing the combinatorial nature of the problem; yet it remains nonconvex. Strictly speaking, the problem solved in [Chap 06] is not transductive in nature, since one does not try to assign labels to the unlabeled points. All one tries to do is to locate the hyperplane in sparse regions and do not "cut" clusters. This is the reason that such techniques are known as *semi-supervised SVM* or S^3VM (see, e.g., [Benn 98, Sind 06a]). At this point it is interesting to note that the borderline between transductive learning and semi-supervised learning that is inductive in nature is not clearly marked, and it is a topic of ongoing discussion. Take, for example, TSVM. After training, the resulting decision hyperplane can be used for induction to predict the label of an unseen point. We are not going to delve into such issues. A very interesting and quite enlightening discussion on the topic, in a "Platonic dialogue" style, is provided in [Chap 06a, Chapter 25].

Remarks

- Besides the previous semi-supervised methodologies that we have presented, a number of other techniques have been suggested. For example, self-training is simple in concept and a commonly used approach. A classifier is first trained with the labeled data. Then this is applied to the unlabeled data to perform label predictions. Based on a confidence criterion, those of the data that result

in confident predictions are added in the labeled training set. The classifier is retrained, and the process is repeated, for example, [Culp 07]. The procedure is similar with that used in the decision feedback equalization (DFE) ([Proa 89]) in communications for more than three decades, in the sense that the classifier uses its own predictions to train itself. Similar in concept is the *co-training* ([Mitc 99, Balc 06, Zhou 07] procedure. However, in this case, the feature set is split into a number of subsets, for example, two, and for each subset a separate classifier is trained using labeled data. The trained classifiers are then used to predict (in their respective feature subspace) labels for the unlabeled points. Each one of the classifiers passes to the other the most confident of the predictions, together with the respective points. The classifiers are then retrained using this new extra information. Splitting the feature set and training the classifiers in different subspaces provides a different complementary "view" of the data points. This reminds us of the classifier combination method where classifiers are trained in different subspaces (Section 4.21). As it was the case there, independence of the two sets is a required assumption.

- A major issue concerning semi-supervised techniques is whether and under what conditions one can obtain enhanced performance compared to training the classifier using labeled data only. A number of papers report that enhanced performance has been obtained. For example, in [Niga 00] a generative approach has been applied to the problem of text classification. It is reported that, using a number of 10,000 unlabeled articles, substantial improvement gains have been attained when the number of labeled documents is small. As the number of labeled data increases from a few tens to a few thousands, the classification accuracies (corresponding to semi-supervised and supervised training) start to converge. In [Chap 06a] a number of semi-supervised techniques were compared using eight benchmark data sets. Some general conclusions are: (a) One must not always expect to obtain improved performance when using unlabeled data. (b) Moreover, the choice of the type of the semi-supervised technique is a crucial issue. The algorithm should "match" the nature of the data set; algorithms that implement the clustering assumption (e.g., TSVM) must be used with data exhibiting a cluster structure, and algorithms that implement the manifold assumption (e.g., Laplacian LS) must be used with data residing on a manifold. So, prior to using a semi-supervised technique, one must have a good "feeling" about the data at hand. This point was also stressed in the remarks given previously, when dealing with the generative methods. It was stated that if the adopted model for the class conditional densities is not correct, then the performance, using unlabeled data, may degrade.

  Besides the cases already mentioned before in this section, the performance of semi-supervised techniques has also been tested in the context of other real applications. In [Kasa 04] transductive SVMs were used to recognize promoter sequences in genes. Their results show that TSVM achieve

enhanced performance compared to the (inductive) SVM. The news coming from [Krog 04], however, is not that encouraging. In the task of predicting functional properties of proteins, the SVM approach resulted in much better performance compared to TSVM. These results are in line with the comments made before; there is no guarantee for performance improvement when using semi-supervised techniques. Some more samples of applications of semi-supervised learning techniques include [Wang 03] for relevance feedback in image retrieval, [Kock 03] for mail classification, and [Blum 98] for Web mining.

- As it was stated in the beginning of this section, our goal was to present to the reader the main concepts behind semi-supervised learning and in particular to see how techniques that have been used in this book in earlier chapters can be extended to this case. A large number of algorithms and methods are around, and the area is, at the time of writing, the focus of an intense research effort by a number of serious groups worldwide. For deeper and broader information, the interested reader may consult, [Chap 06a, Zhu 07].
- Another type of classification framework was proposed (once more) by Vapnik [Chap 06a, Chapter 24]. It is proposed that an additional data set is used, which is not from the same distribution as the labeled data. In other words, the points of this data set do not belong to either of the classes of interest and are called the Universum. This data set is a form of data-dependent regularization and it encodes prior knowledge related to the problem at hand. The classifier must be trained so that the decision function to result in small values for the points in the Universum; that is, these points are forced to lie close to the decision surface. In [West 06] it is shown that different choices of Universa and loss functions result in some known types of regularizers. Early results reported in [West 06, Sinz 06] indicate that the obtained performance depends on the quality of Universum set. The choice of an appropriate Universum is still an open issue. The results obtained in [Sinz 06] suggest that must be carefully chosen and must contain invariant directions and to be positioned "in between" the two classes.

10.6 PROBLEMS

10.1 Let P be the probability that event A occurs. The probability that event A occurs k times in a sequence of N independent experiments is given by the binomial distribution

$$\binom{N}{k} P^k (1-P)^{N-k}$$

Show that $E[k] = NP$ and $\sigma_k^2 = NP(1-P)$.

10.2 In a two-class problem the classifier to be used is the minimum Euclidean distance one. Assume N_1 samples from class ω_1 and N_2 from class ω_2. Show that the leave-one-out method estimate can be obtained from the resubstitution method, if the distance of $\boldsymbol{x}$ from the class means $d_i(\boldsymbol{x}), i = 1, 2$, are modified as $d_i^{'}(\boldsymbol{x}) = (\frac{N_i}{N_i - 1})^2 d_i(\boldsymbol{x})$, if $\boldsymbol{x}$ belongs to class i. Furthermore, show that in this case the leave-one-out method always results in larger error estimates than the resubstitution method.

10.3 Show that for the Bayesian classifier, the estimate provided by the resubstitution method is a lower bound of the true error and that computed from the holdout method is an upper bound.

10.4 Show Eq. (10.25).

REFERENCES

[Balc 06] Balcan M. F., Blum A. "An augmented PAC model for semi-supervised learning," in *Semi-Supervised Learning* (Chapelle O., Schölkopf B., Zien A., eds.), MIT Press, 2006.

[Belk 04] Belkin V., Niyogi P., Sindhwani V. "Manifold regularization: A geometric framework for learning from examples," *Technical Report, TR:2004-06*, Department of Computer Science, University of Chicago, 2004.

[Benn 98] Bennett K., Demiriz A. "Semi-supervised support vector machines," in *Advances in Neural Information Processing Systems*, Vol. 12, 1998.

[Beng 06] Bengio Y., Delalleau O., Le Roux N. "Label propagation and quadratic criterion," in *Semi-Supervised Learning* (Chapelle O., Schölkopf B., Zien A., eds.), MIT Press, 2006.

[Blum 98] Blum A., Mitchell T. "Combining labeled and unlabeled data with co-training," *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pp. 92–100, 1998.

[Brag 04] Braga-Neto U., Dougherty E. "Bolstereol error estimation," *Pattern Recognition*, Vol. 37, pp. 1267–1281, 2004.

[Cast 96] Castelli V., Cover T. "The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter," *IEEE Transactions on Information Theory*, Vol. 42, pp. 2101–2117, 1996.

[Chap 05] Chapelle O., Zien A. "Semi-supervised classification by low density separation," *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, pp. 57–64, 2005.

[Chap 06] Chapelle O., Chi M., Zien A. "A continuation method for semi-supervised SVMs," *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA. 2006.

[Chap 06a] Chapelle O., Schölkopf B., Zien A. *Semi-Supervised Learning*, MIT Press, 2006.

[Cavo 97] Cavouras D., et al. "Computer image analysis of ultrasound images for discriminating and grating liver parenchyma disease employing a hierarchical decision tree scheme and the multilayer perceptron classifier," *Proceedings of Medical Informatics Europe '97*, pp. 517–521, 1997.

[Cohe 04] Cohen I., Cozman F.G., Cirelo M.C., Huang T.S. "Semi-supervised learning of classifiers: Theory, algorithms, and their application to human-computer interaction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26(12), pp. 1553–1567, 2004.

[Cord 02] Corduneanu A., Jaakola T. "Continuation methods for mixing heterogeneous sources," *Proceedings of 18th Annual Conference on Uncertainty in Artificial Intelligence* (Darwiche A., Friedman N., eds.), Alberta, Canada, Morgan Kaufmann, 2002.

[Culp 07] Culp M., Michailidis G. "An iterative algorithm for extending learners to a semi-supervised setting," *Proceedings of the Joint Statistical Meeting (JSM)*, Salt Lake, Utah, 2007.

[DeBi 04] DeBie T., Christianini N. "Convex methods for transduction ," in *Advances in Neural Information Processing Systems* (Thrun S., Saul L, Schölkopf B., eds.), pp. 73–80, MIT Press, 2004.

[Dela 05] Delalleau O., Bengio Y., Le Roux N. "Efficient non-parametric function induction in semi-supervised learning," *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics* (Cowell R.G., Ghahramani Z., eds.), pp. 96–103, Barbados, 2005.

[Demi 00] Demiriz A., Bennett K.P. "Optimization approaches to semi-supervised learning," *Applications and Algorithms of Complementarity* (Ferries M.C., Mangasarian O.L., Pang J.S., eds), pp. 121–141, Kluwer, Dordrecht, the Netherlands, 2000.

[Derb 03] Derbeko P., El-Yanif R., Meir R. "Error bounds for transductive learning via compression and clustering," in *Advances in Neural Information Processing Systems*, pp. 1085–1092, MIT Press, 2003.

[Efro 79] Efron B. "Bootstrap methods: Another look at the jackknife," *Annals of Statistics*, Vol. 7, pp. 1–26, 1979.

[Efro 83] Efron B. "Estimating the error rate of a prediction rule: Improvement on cross-validation," *Journal of the American Statistical Association*, Vol. 78, pp. 316–331, 1983.

[Fole 72] Foley D. "Consideration of sample and feature size," *IEEE Transactions on Information Theory*, Vol. 18(5), pp. 618–626, 1972.

[Fung 01] Fung G., Mangasarian O. "Semi-supervised support vector machines for unlabeled data classification," *Optimization Methods and Software*, Vol. 15, pp. 29–44, 2001.

[Fuku 90] Fukunaga K. *Introduction to Statistical Pattern Recognition*, 2nd ed., Academic Press, 1990.

[Guyo 98] Guyon I., Makhoul J., Schwartz R., Vapnik U. "What size test set gives good error rate estimates?" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20(1), pp. 52–64, 1998.

[Hand 86] Hand D.J. "Recent advances in error rate estimation," *Pattern Recognition Letters*, Vol. 5, pp. 335–346, 1986.

[Hoer 70] Hoerl A.E., Kennard R. "Ridge regression: biased estimate for nonorthogonal problems," *Technometrics*, Vol. 12, pp. 55–67, 1970.

[Ivan 62] Ivanov V.V. "On linear problems which are not well-posed," *Soviet Mathematical Docl.*, Vol. 3(4), pp. 981–983, 1962.

[Jain 87] Jain A.K., Dubes R.C., Chen C.C. "Bootstrap techniques for error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 9(9), pp. 628–636, 1987.

[Joac 02] Joachims T. *Learning to Classify Text Using Support Vector Machines*, Kluwer, Dordrecht, the Netherlands, 2002.

[Joac 99] Joachims T. "Transductive inference for text classification using support vector machines," *Proceedings of 16th International Conference on Machine Learning (ICML)*, (Bratko I., Dzeroski S., eds), pp. 200–209, 1999.

[Kana 74] Kanal L. "Patterns in Pattern Recognition," *IEEE Transactions on Information Theory*, Vol. 20(6), pp. 697–722, 1974.

[Kasa 04] Kasabov N., Pang S. "Transductive support vector machines and applications to bioinformatics for promoter recognition," *Neural Information Processing-Letters and Reviews*, Vol. 3(2), pp. 31–38, 2004.

[Kock 03] Kockelkorn M., Lüneburg A., Scheffer T. "Using transduction and multi-view learning to answer emails," *Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 266–277, 2003.

[Krog 04] Krogel M., Scheffer T. "Multirelational learning, text mining and semi-supervised learning for functional genomics," *Machine Learning*, Vol. 57(1/2), pp. 61–81, 2004.

[Lach 68] Lachenbruch P.A., Mickey R.M. "Estimation of error rates in discriminant analysis," *Technometrics*, Vol. 10, pp. 1–11, 1968.

[Leis 98] Leisch F., Jain L.C., Hornik K. "Cross-validation with active pattern selection for neural network classifiers," *IEEE Transactions on Neural Networks*, Vol. 9(1), pp. 35–41, 1998.

[Mill 97] Miller D.J., Uyar H. "A mixture of experts classifier with learning based on both labeled and unlabeled data," *Neural Information Processing Systems*, Vol. 9, pp. 571–577, 1997.

[Mill 03] Miller D.J., Browning J. "A mixture model and EM-algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25(11), pp. 1468–1483, 2003.

[Mitc 99] Mitchell T. "The role of unlabeled data in supervised learning," *Proceedings of the 6th International Colloquium on Cognitive Science*, San Sebastian, Spain, 1999.

[Niga 00] Nigam K., McCallum A.K., Thrun S., Mitchell T. "Text classification from labeled and unlabeled documents using EM," *Machine Learning*, Vol. 39, pp. 103–134, 2000.

[Papo 02] Papoulis A., Pillai S.U. *Probability, Random Variables, and Stochastic Processes*, 4th eds, McGraw-Hill, 2002.

[Phil 62] Phillips D.Z. "A technique for numerical solution of certain integral equation of the first kind," *Journal of Association of Computer Machinery (ACM)*, Vol. 9, pp. 84–96.

[Proa 89] Proakis J. *Digital Communications*, McGraw-Hill, 1989.

[Raud 91] Raudys S.J., Jain A.K. "Small size effects in statistical pattern recognition: Recommendations for practitioners," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13(3), pp. 252–264, 1991.

[Redn 84] Redner R.A., Walker H.M. "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review*, Vol. 26(2), pp. 195–239, 1984.

[Scud 65] Scudder H.J. "Probability of error of some adaptive pattern recognition machines," *IEEE Transactions on Information Theory*, Vol. 11, pp. 363–371, 1965.

[Shas 94] Shashahani B., Landgrebe D. "The effect of unlabeled samples in reducing the small sample size problem and mitigating the Hughes phenomenon," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 32, pp. 1087–1095, 1994.

[Sima 06] sima C, Dougherty E.R. "Optimal convex error estimators for classification," Pattery Recognist, Vol. 39(9), pp. 1763–1780, 2006.

[Sind 06a] Sindhawi V, Keerthi S., Chapelle O. "Deterministic annealing for semi-supervised kernel machines," *Proceedings of the 23nd nternational Conference on Machine Learning*, 2006.

[Sind 06] Sindhawi V., Belkin M., Niyogi P. "The geometric basis of semi-supervised learning," in *Semi-Supervised Learning* (Chapelle O., Schölkopf B., Zien A., eds.), MIT Press, 2006.

[Sinz 06] Sinz F.H., Chapelle O., Agarwal A., Schölkopf B. "An analysis of inference with the Universum," *Proceedings of the 20th Annual Conference on Neural Information Processing Systems (NIPS)*, MIT Press, Cambridge, Mass., USA, 2008.

[Szum 02] Szummer M., Jaakkola T. "Partially labeled classification with Markov random fields," in *Advances in Neural Information Processing Systems* (Dietterich T.G., Becker S., Ghahramani Z., eds.), MIT Press, 2002.

[Tiho 63] Tikhonov A.N. "On solving ill-posed problems and a method for regularization," *Doklady Akademii Nauk, USSR*, Vol. 153, pp. 501–504, 1963.

[Vapn 74] Vapnik V., Chervonenkis A.Y. *Theory of Pattern Recognition* (in Russian), Nauka, Moskow, 1974.

[Vapn 99] Vapnik V. *The Nature of Statistical Learning Theory*, Springer, 1999.

[Vapn 06] Vapnik V. "Transductive inference and semi-supevised learning," in *Semi-Supervised Learning* (Chapelle O., Schölkopf B., Zien A., eds.), MIT Press, 2006.

[Wang 03] Wang L., Chan K.L., Zhang Z. "Bootstrapping SVM active learning by incorporating unlabeled images for retrieval," *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 629–639, 2003.

[West 06] Weston J., Collobert R., Sinz F., Bottou L., Vapnik V. "Inference with the Universum," *Proceedings of the 23nd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

[Wu 07] Wu M., Schölkopf B. "Transductive classification via local learning regularization," *Proceedings 11th International conference on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 2007.

[Zhou 04] Zhou D., Bousquet O., Lal T.N., Weston J., Schölkopf "Learning with local and global consistency," in *Advances in Neural Information Processing Systems* (Thrun S., Saul L, Schölkopf B., eds.), pp. 321–328, MIT Press, 2004.

[Zhou 07] Zhou Z.H., Xu J.M. "On the relation between multi-instance learning and semi-supervised learning," *Proceedings of the 24th International Conference on Machine Learning*, Oregon State, 2007.

[Zhu 02] Zhu X., Ghahramani Z. "Learning from labeled and unlabeled data with label propagation," *Technical Report CMU-CALD-02-107*, Carnegie Mellon University, Pittsburgh, PA, 2002.

[Zhu 07] Zhu X. "Semi-supervised learning literature review," *Technical Report, TR 1530*, Computer Science Department, University of Wisconsin-Madison, 2007.