59 LOGISTIC REGRESSION

In this chapter, we describe a popular *discriminative* approach for classification problems known as logistic regression. Assuming binary classification with labels $\boldsymbol{\gamma} \in \{\pm 1\}$ and features $\boldsymbol{h} \in \mathbb{R}^M$, we explained earlier in expression (28.85) that the optimal Bayes classifier for predicting $\boldsymbol{\gamma}$ is given by

$$\boldsymbol{\gamma}^\bullet = \begin{cases} +1, & \text{when } \widehat{\gamma}_{\rm LR} = \text{logit}(h) \geq 0 \\ -1, & \text{otherwise} \end{cases} \tag{59.1}$$

where $\boldsymbol{\gamma}^\bullet$ minimizes the probability of error, and logit(h) denotes the logit function defined in terms of the true conditional probabilities:

$$\text{logit}(h) = \ln\left(\frac{\mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h)}{\mathbb{P}(\boldsymbol{\gamma} = -1|\boldsymbol{h} = h)}\right) \tag{59.2}$$

It was shown in Sec. 28.4 that this expression for logit(h) or, equivalently, $\widehat{\gamma}_{\rm LR}$, followed from minimizing the logistic risk, namely,

$$\widehat{\gamma}_{\rm LR} = \underset{\widehat{\boldsymbol{\gamma}}=c(\boldsymbol{h})}{\text{argmin}} \left\{\mathbb{E} \ln\left(1 + e^{-\boldsymbol{\gamma}\widehat{\boldsymbol{\gamma}}}\right)\right\} \tag{59.3}$$

where the expectation is over the joint distribution of $(\boldsymbol{\gamma}, \boldsymbol{h})$. Expression (59.1) shows that we can deduce the optimal Bayes solution by examining the sign of logit(h) or $\widehat{\gamma}_{\rm LR}$. Unfortunately, this is not possible in general because, once again, the conditional probability $\mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h)$, which is needed to evaluate the logit, is not known beforehand. The logistic regression approach of this chapter addresses this challenge by restricting $c(h)$ to an *affine* function of the feature data, i.e., by assuming

$$c(h) = h^{\mathsf T} w - \theta \tag{59.4}$$

for some scalar offset θ and vector parameter $w \in \mathbb{R}^M$ to be determined.

59.1 LOGISTIC MODEL

Under the affine model (59.4), and referring to the earlier expressions (28.82a)–(28.82b), the logistic formulation models the conditional probabilities in the form of logistic functions as follows:

$$\mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h) = \frac{1}{1 + e^{-(h^{\mathsf{T}} w - \theta)}} \tag{59.5a}$$

$$\mathbb{P}(\boldsymbol{\gamma} = -1|\boldsymbol{h} = h) = \frac{1}{1 + e^{+(h^{\mathsf{T}} w - \theta)}} \tag{59.5b}$$

These expressions correspond to composing the sigmoid functions $1/(1+e^{-z})$ and $1/(1+e^{z})$ with the affine function $h^{\mathsf{T}}w - \theta$. The above relations explain why the logistic solution is a *discriminative* approach. This is because it models directly the conditional probabilities rather than the joint distribution for $(\boldsymbol{\gamma}, \boldsymbol{h})$, as was the case with generative approaches. We can group the above two relations into a single expression and write

$$\begin{aligned} \mathbb{P}(\boldsymbol{\gamma} = \gamma|\boldsymbol{h} = h) &= \frac{1}{1 + e^{-\gamma(h^{\mathsf{T}} w - \theta)}}, \quad \text{since } \gamma \in \{\pm 1\} \\ &= \frac{1}{1 + e^{-\gamma\widehat{\gamma}}}, \quad \text{where } \widehat{\gamma} \overset{\Delta}{=} h^{\mathsf{T}} w - \theta \end{aligned} \tag{59.6}$$

Figure 59.1 illustrates the behavior of logistic functions of the form $1/(1+e^{-z})$ and $1/(1+e^{z})$. Note that these functions return values between 0 and 1 (as befits a true probability measure).

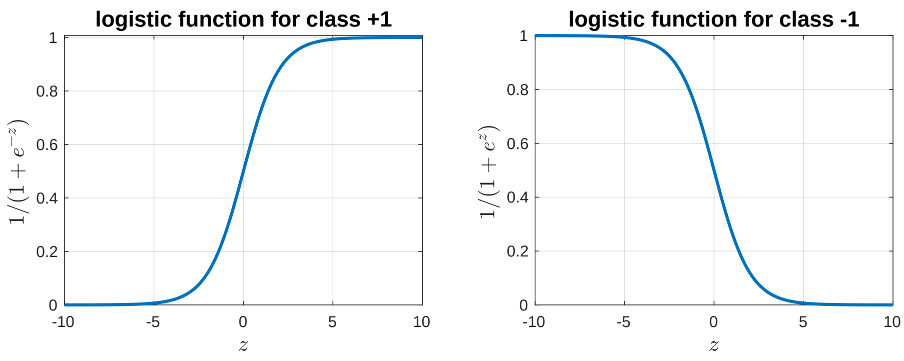

Figure 59.1 Typical behavior of logistic functions for two classes. The figure shows plots of the functions $1/(1+e^{-z})$ (*left*) and $1/(1+e^{z})$ (*right*) assumed to correspond to classes $+1$ and -1, respectively.

Using construction (59.4), problem (59.3) is transformed into

$$(w^o, \theta^o) \;=\; \underset{(w,\theta)}{\text{argmin}} \left\{ \mathbb{E} \ln\left(1 + e^{-\boldsymbol{\gamma}(\boldsymbol{h}^{\mathsf{T}} w - \theta)}\right) \right\} \tag{59.7}$$

Comparing with (59.5a)–(59.5b), we find that this objective is attempting to maximize (59.5a) on average when $\boldsymbol{\gamma} = +1$ and is attempting to minimize it when $\boldsymbol{\gamma} = -1$. Once (w^o, θ^o) are determined, the prediction for the label variable is given by (where we are dropping the subscript LR from $\widehat{\gamma}_{\text{LR}}$ and writing $\widehat{\gamma}$

because, in this chapter, all predictions will be based on the logistic formulation):

$$\widehat{\boldsymbol{\gamma}} = \boldsymbol{h}^{\mathsf{T}} w^o - \theta^o \tag{59.8}$$

The sign of $\widehat{\gamma}$ is used to decide on the label for h:

$$\begin{cases} \text{if } h^{\mathsf{T}} w^o - \theta^o \geq 0, \text{ assign } h \text{ to class } +1 \\ \text{if } h^{\mathsf{T}} w^o - \theta^o < 0, \text{ assign } h \text{ to class } -1 \end{cases} \tag{59.9}$$

This conclusion can also be deduced by comparing $\mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h)$ to 1/2, as is required by the optimal Bayes solution:

$$\begin{aligned} &\mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h)\frac{1}{1 + e^{-(h^{\mathsf{T}} w^o - \theta^o)}} \geq 1/2 \\ &\Longleftrightarrow h^{\mathsf{T}} w^o - \theta^o \geq 0 \\ &\Longleftrightarrow \gamma^o(h) = +1, \quad (\textbf{class assigned to } h\ ) \end{aligned}$$

where we are denoting the class variable assigned to h under model (w^o, θ^o) by γ^o. We therefore arrive at the schematics shown in Fig. 59.2. The figure shows a collection of feature vectors and a hyperplane whose normal direction is w^o. All points on the hyperplane satisfy the relation $h^{\mathsf{T}} w^o - \theta^o = 0$, while points on both sides of the hyperplane satisfy $h^{\mathsf{T}} w^o - \theta^o < 0$ for one side and $h^{\mathsf{T}} w^o - \theta^o > 0$ for the other. The logistic regression solution decides on the label for a feature vector h by verifying on which side h falls relative to the separating hyperplane.

Since the distribution of the data $(\boldsymbol{\gamma}, \boldsymbol{h})$ is not known, we will proceed to solve problem (59.7) by transforming it into an *empirical risk minimization* problem and then applying any of the stochastic approximation methods we described in earlier chapters at some great length. Although we will focus in the body of the chapter on the binary classification problem, we explain in the comments at the end of the chapter how logistic regression can be extended to multi-class classification problems — see, e.g., expression (59.85) and also Prob. 59.14.

Remark 59.1. **(Comparing with linear discriminant analysis)** Comparing (59.5a) with expression (56.5) for linear discriminant analysis (LDA), we find that logistic regression assumes directly that the conditional probability has a logistic form, which is parameterized in terms of the unknown vector w; there is no Gaussian assumption on the feature data. In contrast, LDA assumes that the feature distribution is Gaussian according to (56.3) and arrives at expression (56.5), which is parameterized in terms of the moment quantities $\{m_r, \Sigma\}$. It was seen before in (56.7) that model (56.5) in the Gaussian case leads to a logit expression that is *linear* in the feature space.

■

59.2 LOGISTIC EMPIRICAL RISK

We assume the availability of N training samples $\{\gamma(n), h_n\}$ where $\gamma(n) \in \{\pm 1\}$ is the label associated with the $n-$th feature vector, $h_n \in \mathbb{R}^M$. Using this data,

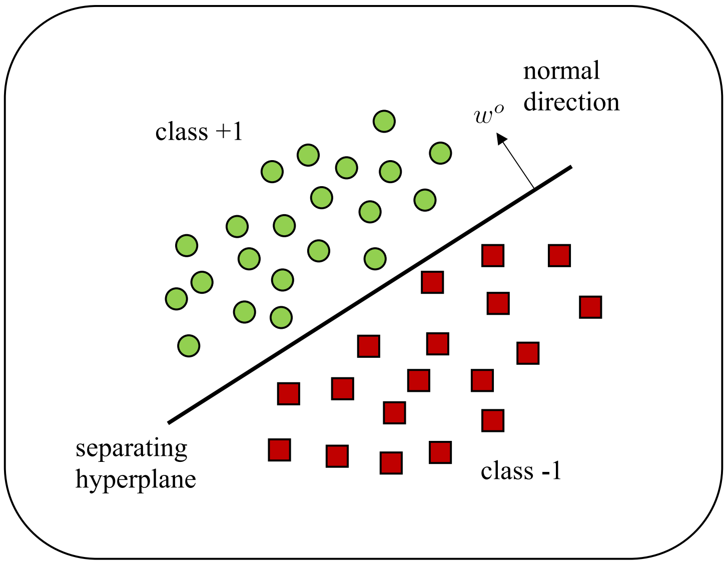

Figure 59.2 Classification of feature vectors into two classes: data with nonnegative logit values, $\widehat{\gamma} = h^{\mathsf{T}} w^o - \theta^o \geq 0$, are assigned to one class and data with negative logit values, $\widehat{\gamma} = h^{\mathsf{T}} w^o - \theta^o < 0$, are assigned to another class. The unknown vector w^o defines the direction that is normal to the separating hyperplane.

we replace the stochastic optimization problem (59.7) by the empirical risk minimization problem

$$(w^\star, \theta^\star) \overset{\Delta}{=} \underset{w \in \mathbb{R}^M, \theta \in \mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{N} \sum_{n=0}^{N-1} \ln\left(1 + e^{-\gamma(n)(h_n^{\mathsf{T}} w - \theta)}\right) \right\} \tag{59.10}$$

where the optimizer is now denoted by $(w^\star, \theta^\star)$. This problem can be solved by a variety of stochastic optimization methods, already discussed in previous chapters, such as stochastic gradient algorithms and variations thereof including mini-batch implementations, ADAM, or accelerated momentum. It is sufficient to illustrate the solution by considering one method. We focus on stochastic gradient implementations, with and without regularization, which rely on instantaneous gradient approximations. The sampling of the data in the stochastic implementation can also be done with or without replacement.

Before describing the recursive algorithm, we clarify that we can motivate the same empirical risk problem (59.10) from a purely maximum-likelihood perspective. Indeed, assume the training data $\{\boldsymbol{\gamma}(n), \boldsymbol{h}_n\}$ are independent and identically distributed observations arising from some joint distribution, $f_{\boldsymbol{\gamma},\boldsymbol{h}}(\gamma, h)$. Under the assumed logistic model (59.6), the log-likelihood function of the labels given the features is given by the product expression:

$$
\begin{aligned}
\ell(w) &\stackrel{\Delta}{=} \ln\left\{\prod_{n=0}^{N-1}\mathbb{P}\Big(\boldsymbol{\gamma}(n)=\gamma(n)\mid \boldsymbol{h}_n=h_n;w\Big)\right\}\\
&= \sum_{n=0}^{N-1}\ln\left(\frac{1}{1+e^{-\gamma(n)(h_n^{\mathsf T}w-\theta)}}\right) \qquad (59.11)
\end{aligned}
$$

so that maximizing $\ell(w)$ over w leads to the empirical risk minimization problem (59.10).

In practice, the optimization problem (59.10) is modified to incorporate regularization for reasons already explained in Chapter 51, such as reducing ill-conditioning, reducing the possibility of overfitting, and endowing $w^\star$ with desirable properties such as having a small norm or sparse structure. For generality, we will consider logistic regression under elastic-net regularization and replace (59.10) by

$$
\begin{aligned}
&(w^\star,\theta^\star) \stackrel{\Delta}{=} \qquad (59.12)\\
&\underset{w\in\mathbb{R}^M,\theta\in\mathbb{R}}{\operatorname{argmin}}\left\{P(w)\stackrel{\Delta}{=}\alpha\|w\|_1+\rho\|w\|^2 \;+\; \frac{1}{N}\sum_{n=0}^{N-1}\ln\Big(1+e^{-\gamma(n)(h_n^{\mathsf T}w-\theta)}\Big)\right\}
\end{aligned}
$$

where α and ρ are nonnegative scalars, with one of them or both being set to zero depending on whether we desire to enforce ℓ_1-regularization alone ($\rho=0$), ℓ_2-regularization alone ($\alpha=0$), or no regularization at all ($\alpha=\rho=0$). Using the result of Example 16.13, we list a stochastic proximal algorithm for solving (59.12) in (59.14), where the notation $\mathbb{T}_\beta(x)$ refers to the soft-thresholding function defined by (11.18), namely,

$$
\mathbb{T}_\beta(x) \stackrel{\Delta}{=} \begin{cases} x-\beta, & \text{if } x\geq\beta\\ 0, & \text{if } -\beta<x<\beta\\ x+\beta, & \text{if } x\leq-\beta\end{cases} \qquad (59.13)
$$

When x is vector-valued, the operation $\mathbb{T}_\beta(x)$ is applied to the individual entries of x and the result is a vector of soft-thresholded values.

Stochastic proximal logistic regression for minimizing (59.12)

```
given dataset $\{\gamma(m), h_m\}_{m=0}^{N-1}$ or streaming data $(\gamma(n), h_n)$;
start from arbitrary initial conditions, $\{\boldsymbol{w}_{-1}, \boldsymbol{\theta}(-1)\}$.
repeat until convergence over $n \geq 0$ :
    select at random or receive a sample $(\boldsymbol{\gamma}(n), \boldsymbol{h}_n)$ at iteration $n$
    $\widehat{\boldsymbol{\gamma}}(n) = \boldsymbol{h}_n^{\mathsf{T}} \boldsymbol{w}_{n-1} - \boldsymbol{\theta}(n-1)$
    $\boldsymbol{\theta}(n) = \boldsymbol{\theta}(n-1) - \dfrac{\mu \boldsymbol{\gamma}(n)}{1 + e^{\boldsymbol{\gamma}(n)\widehat{\boldsymbol{\gamma}}(n)}}$
    $\boldsymbol{z}_n = (1 - 2\mu\rho)\boldsymbol{w}_{n-1} + \dfrac{\mu \boldsymbol{\gamma}(n)\boldsymbol{h}_n}{1 + e^{\boldsymbol{\gamma}(n)\widehat{\boldsymbol{\gamma}}(n)}}$
    $\boldsymbol{w}_n = \mathbb{T}_{\mu\alpha}(\boldsymbol{z}_n)$
end
return $(w^\star, \theta^\star) \leftarrow (\boldsymbol{w}_n, \boldsymbol{\theta}(n))$;
classify a feature $h$ by using the sign of $\widehat{\gamma} = h^{\mathsf{T}} w^\star - \theta^\star$ as in (59.9)
```
(59.14)

When $\alpha = 0$ and only ℓ_2-regularization is present, the above listing reduces to the stochastic gradient logistic regression algorithm:

$$\left\{\begin{array}{l} \widehat{\boldsymbol{\gamma}}(n) = \boldsymbol{h}_n^{\mathsf{T}} \boldsymbol{w}_{n-1} - \boldsymbol{\theta}(n-1) \\ \boldsymbol{\theta}(n) = \boldsymbol{\theta}(n-1) - \dfrac{\mu \boldsymbol{\gamma}(n)}{1 + e^{\boldsymbol{\gamma}(n)\widehat{\boldsymbol{\gamma}}(n)}} \\ \boldsymbol{w}_n = (1 - 2\mu\rho)\boldsymbol{w}_{n-1} + \dfrac{\mu \boldsymbol{\gamma}(n)\boldsymbol{h}_n}{1 + e^{\boldsymbol{\gamma}(n)\widehat{\boldsymbol{\gamma}}(n)}}, \quad n \geq 0 \end{array}\right. \tag{59.15}$$

We can simplify the notation by extending the feature and weight vectors as follows:

$$h \leftarrow \left[\begin{array}{c} 1 \\ \hline h \end{array}\right], \quad w \leftarrow \left[\begin{array}{c} -\theta \\ \hline w \end{array}\right] \tag{59.16}$$

so that (59.15) can be rewritten more compactly in the following manner where the offset parameter is now implicit:

$$\left\{\begin{array}{l} \widehat{\boldsymbol{\gamma}}(n) = \boldsymbol{h}_n^{\mathsf{T}} \boldsymbol{w}_{n-1} \\ \boldsymbol{w}_n = A\boldsymbol{w}_{n-1} + \left(\dfrac{\mu \boldsymbol{\gamma}(n)}{1 + e^{\boldsymbol{\gamma}(n)\widehat{\boldsymbol{\gamma}}(n)}}\right)\boldsymbol{h}_n, \quad n \geq 0 \end{array}\right. \tag{59.17}$$

where the diagonal matrix A depends on the regularization parameter:

$$A \stackrel{\Delta}{=} \left[\begin{array}{cc} 1 & \\ & (1 - 2\mu\rho)I_M \end{array}\right] \tag{59.18}$$

When a mini-batch of size B is used, the above recursion is replaced by

$$\begin{cases} \text{select } B \text{ data samples } \{\boldsymbol{\gamma}(b), \boldsymbol{h}_b\} \text{ at random} \\ \widehat{\gamma}(b) = \boldsymbol{h}_b^{\mathsf{T}} \boldsymbol{w}_{n-1}, \quad b = 0, 1, \ldots, B-1 \\ \boldsymbol{w}_n = A\boldsymbol{w}_{n-1} + \displaystyle\sum_{b=0}^{B-1} \left(\frac{\mu\, \boldsymbol{\gamma}(b) \boldsymbol{h}_b}{1 + e^{\boldsymbol{\gamma}(b)\widehat{\boldsymbol{\gamma}}(b)}} \right) \boldsymbol{h}_b, \quad n \geq 0 \end{cases} \tag{59.19}$$

Example 59.1 **(Binary classification using logistic regression)** In Fig. 59.3 we show a collection of 150 feature points $h_n \in \mathbb{R}^2$ whose classes $\gamma(n) \in \{\pm 1\}$ are known beforehand. The data arises from the dimensionally reduced iris dataset from Example 57.3. We denoted the two-dimensional reduced feature vectors in that example by the notation h'_n, but will revert to the notation h_n here. We employ the two classes shown in the bottom plot of Fig. 57.5 and denote them by $\gamma(n) \in \{\pm 1\}$.

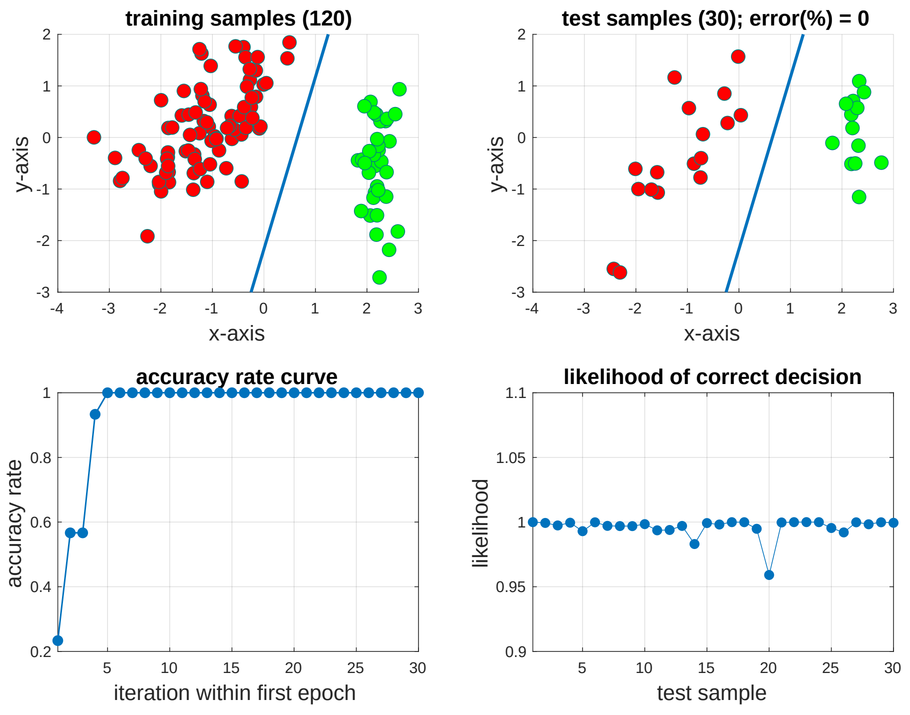

Figure 59.3 The plots in the first row show the 120 data samples used for training (*left*) and the 30 data samples using for testing (*right*). The separation line is obtained by running the stochastic gradient logistic algorithm (59.15). The algorithm performs five passes over the training data using $\mu = 0.5$. The rightmost plot in the second row shows the likelihood values for each of the test feature vectors, i.e., the probability that its classification is correct using (59.6).

The feature vectors are extended according to (59.16). We split the data into two sets: 120 points (80%) are selected randomly and used for training, while the remaining 30 points (20%) are used for testing. These two sets are exclusive of each other and no data point from one set appears in the other set. We use the 120 samples to train a logistic classifier using the stochastic gradient recursion (59.15) without regularization ($\rho = 0$) and with step-size $\mu = 0.5$. We also employ random reshuffling. Specifically, we run 5 epochs with the data reshuffled at the start of each epoch, and the algorithm runs over the reshuffled data starting from the weight iterate obtained at the end of the previous pass. The lines in the figure show the resulting separating curve. At the end of the training phase, we assess the empirical error rate of the classifier on the test data and find that it leads to zero errors. We also show in the figure the likelihood values for each of the test feature vectors, i.e., the probability that its classification is correct, by using expression (59.6) with w replaced by $w^\star$. In the bottom row on the left we show the accuracy rate curve over the first 30 iterations of the *first epoch* in order to illustrate how the error rate decreases over time. For each of these initial iterations, the weight iterate w_n is used to classify the data and the resulting accuracy rate is plotted. It is seen that, for this example, the error rate quickly drops to zero (or the accuracy rate quickly reaches 100%).

59.3 MULTICLASS CLASSIFICATION

Although the formulation of the logistic regression problem has focused so far on the binary case, it can nevertheless be extended to multiclass classification problems by following similar arguments. This is explained in the comments at the end of the chapter — see, e.g., expression (59.85) and also Prob. 59.14, which deal with the situation when h_n can belong to one of multiple classes.

A second useful way to solve multiclass classification problems in general is to use binary classifiers as building blocks. The logistic regression classifier is particularly useful for that purpose because, unlike other learning algorithms to be described in future chapters, it provides a level of confidence in its classification decision by means of expression (59.6), namely,

$$\left\{\begin{array}{l}\textbf{confidence level that } \gamma \textbf{ is the correct label for } h \textbf{ is assessed by}:\\ \mathbb{P}(\boldsymbol{\gamma} = \gamma | \boldsymbol{h} = h) = \dfrac{1}{1+e^{-\gamma\widehat{\gamma}}}, \quad \widehat{\gamma} = h^{\mathsf{T}} w^\star - \theta^\star\end{array}\right. \tag{59.20}$$

We describe next two popular techniques that take advantage of this property, known as the *one-versus-all* (OvA) and the *one-versus-one* (OvO) strategies.

59.3.1 OvA Strategy

Assume there are a total of R classes indexed by $r \in \{1, 2, \ldots, R\}$, and that we are given a collection of training points $\{r(n), h_n\}$, where $r(n)$ now denotes the class attached to feature h_n. Given a test feature vector h, we would like identify which class it belongs to. The OvA approach works by designing R separate binary classifiers, one for each class r. For the first classifier corresponding to

$r = 1$, all training points are redefined by setting:

$$\gamma(n) = \begin{cases} +1, & \text{if } h_n \in \text{class } r = 1 \\ -1, & \text{otherwise} \end{cases} \tag{59.21}$$

That is, data points belonging to class $r = 1$ are treated as belonging to the binary class $\gamma = +1$, while all remaining data points are treated as belonging to the binary class $\gamma = -1$. This construction is illustrated in Fig. 59.4. A logistic classifier is then trained and weight and offset parameters are obtained at the end of this first training denoted by $(w_1^\star, \theta_1^\star)$. The hyperplane separates features belonging to class $r = 1$ from all other features. This description already highlights one of the inconveniences of the OvA approach: during training, the binary classifier encounters many more misclassified data (those belonging to all other classes) than positive decisions.

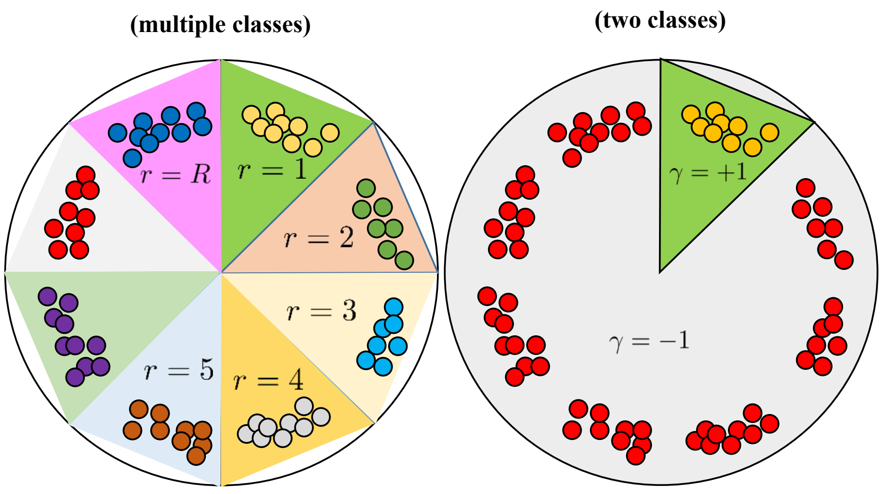

Figure 59.4 In the one-versus-all (OvA) strategy, a collection of R binary classifiers are designed for a multiclass classification problem involving R cases.

We repeat the procedure for the second class, $r = 2$. All training points are redefined by setting:

$$\gamma(n) = \begin{cases} +1, & \text{if } h_n \in \text{class } r = 2 \\ -1, & \text{otherwise} \end{cases} \tag{59.22}$$

A logistic classifier is trained and results in parameters $(w_2^\star, \theta_2^\star)$. This hyperplane separates features belonging to class $r = 2$ from all other features. We continue in this manner until a total of R separating hyperplanes are determined:

$$\Big\{(w_1^\star, \theta_1^\star),\ (w_2^\star, \theta_2^\star)\ \ldots,\ (w_R^\star, \theta_R^\star)\Big\} \tag{59.23}$$

According to this notation, hyperplane $(w_r^\star, \theta_r^\star)$ separates the features belonging to class r from all other features.

During testing, when the classification machine receives a feature vector, h, and wishes to classify it, the procedure is as follows. Each logistic classifier in (59.23) generates a classification decision for h (i.e., decides whether it belongs to class $+1$ or class -1) along with a likelihood measure, which measures the level of confidence of the classifier in its decision. The confidence level for each classifier $w_r^\star$ is obtained from expression (59.20):

$$\mathbb{P}(\boldsymbol{r} = r \mid \boldsymbol{h} = h; w_r^\star) \;=\; \frac{1}{1 + e^{-(h^\mathsf{T} w_r^\star - \theta_r^\star)}} \tag{59.24}$$

We then set the final classification class for h by selecting the classifier from the set (59.23) with the largest confidence level:

$$r^\star(h) \;\overset{\Delta}{=}\; \underset{1 \leq r \leq R}{\text{argmax}} \left\{ \frac{1}{1 + e^{-(h^\mathsf{T} w_r^\star - \theta_r^\star)}} \right\} \tag{59.25}$$

Alternatively, note that if a classifier $w_a^\star$ has a higher confidence than classifier $w_b^\star$, then

$$\frac{1}{1 + e^{-(h^\mathsf{T} w_a^\star - \theta_a^\star)}} > \frac{1}{1 + e^{-(h^\mathsf{T} w_b^\star - \theta_b^\star)}} \tag{59.26}$$

which is equivalent to

$$h^\mathsf{T} w_a^\star - \theta_a^\star > h^\mathsf{T} w_b^\star - \theta_b^\star \tag{59.27}$$

This suggests that we can also select the class label by solving instead

$$r^\star(h) \;\overset{\Delta}{=}\; \underset{1 \leq r \leq R}{\text{argmax}} \left\{ h^\mathsf{T} w_r^\star - \theta_r^\star \right\} \tag{59.28}$$

Example 59.2 (**Applying OvA to the iris dataset**) We consider the dimensionally reduced iris dataset from the top plot of Fig. 57.5. There are three classes denoted by $r \in \{0, 1, 2\}$ corresponding to the Setosa ($r = 0$), Versicolor ($r = 1$), and Virginica ($r = 2$) flower types. There are also a total of $N = 150$ samples. The plots in Fig. 59.5 show all data samples, along with the groupings that result from considering samples from one class against the combined samples from the other two classes.

The feature vectors are extended according to (59.16). A collection of 120 samples are selected for training while the remaining 30 samples are used for testing. We use the 120 samples to train a logistic classifier using the stochastic gradient recursion (59.15) with $\mu = 0.01$ and $\rho = 0.1$. Five passes of the algorithm with random reshuffling are applied to the data resulting in (where, for completeness, we are highlighting the offset and weight vector parameters):

$$\begin{bmatrix} -\theta_{0,12}^\star \\ \\ w_{0,12}^\star \end{bmatrix} = \left[\begin{array}{c} -0.4813 \\ \hline 1.1256 \\ -0.3421 \end{array}\right], \quad \text{(to separate class 0 from classes (1,2))} \tag{59.29a}$$

and

$$\begin{bmatrix} -\theta_{1,02}^\star \\ \\ w_{1,02}^\star \end{bmatrix} = \left[\begin{array}{c} -0.4134 \\ \hline -0.1498 \\ 0.5381 \end{array}\right], \quad \text{(to separate class 1 from classes (0,2))} \tag{59.29b}$$

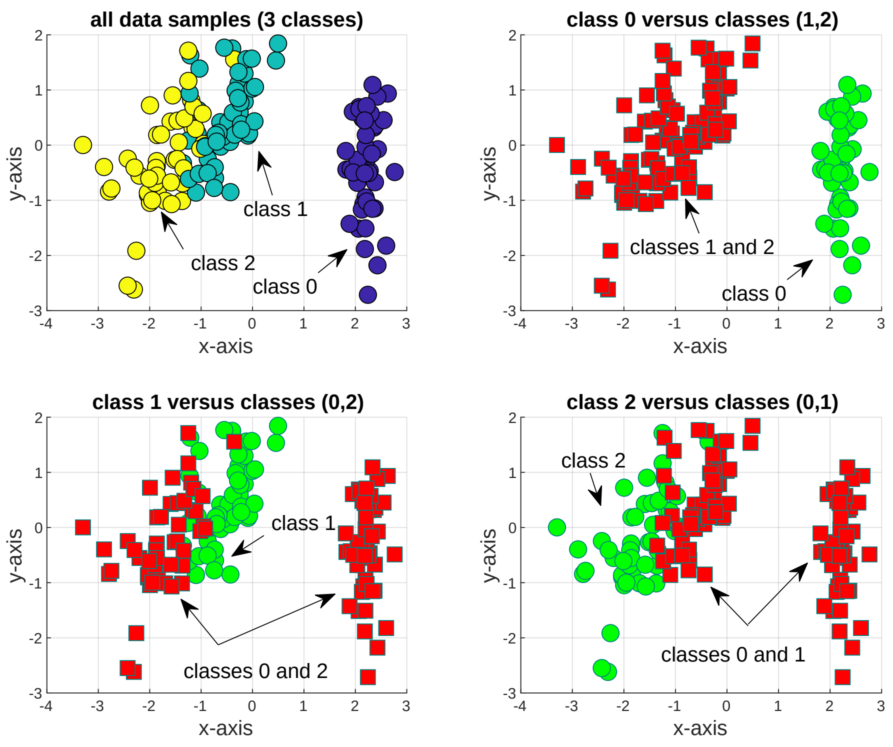

Figure 59.5 The top plot (*left*) shows all data samples from the three classes $r = 0, 1, 2$. The other plots show the groupings that result from considering samples from one class against the combined samples from the other two classes.

and

$$\begin{bmatrix} -\theta^{\star}_{2,01} \\ \\ w^{\star}_{2,01} \end{bmatrix} = \left[\begin{array}{c} -0.4931 \\ \hline -0.8530 \\ -0.2329 \end{array}\right], \quad \text{(to separate class 2 from classes (0,1))} \qquad (59.29c)$$

Figure 59.6 shows the training data and the test data. It also shows the resulting separating lines. It is clear from the plot on the right in the top row of the figure that it is not possible to separate class $r = 1$ from the combined classes $r \in \{0, 2\}$ by means of a linear classifier. The same is true, albeit to a lesser extent, for separating class $r = 2$ from the combined classes $r \in \{0, 1\}$. The empirical error rates obtained over the training data in each of the three cases shown in the figure are 0% for separating $r = 0$ from $r \in \{1, 2\}$, 26.67% for separating $r = 1$ from $r \in \{0, 2\}$, and 13.33% for separating $r = 2$ from $r \in \{0, 1\}$.

Next, for each test vector h, we use expression (59.24) to determine the likelihood that it belongs to class $r \in \{0, 1, 2\}$. The bottom plot in Fig. 59.7 shows the *largest* likelihood value for each test sample. The top plot on the right shows the predicted labels over the test data. The samples that are misclassified are marked in this plot by red. It is observed that five samples are misclassified resulting in an empirical error rate of 16.67% over the test data (or 5 errors out of 30 samples).

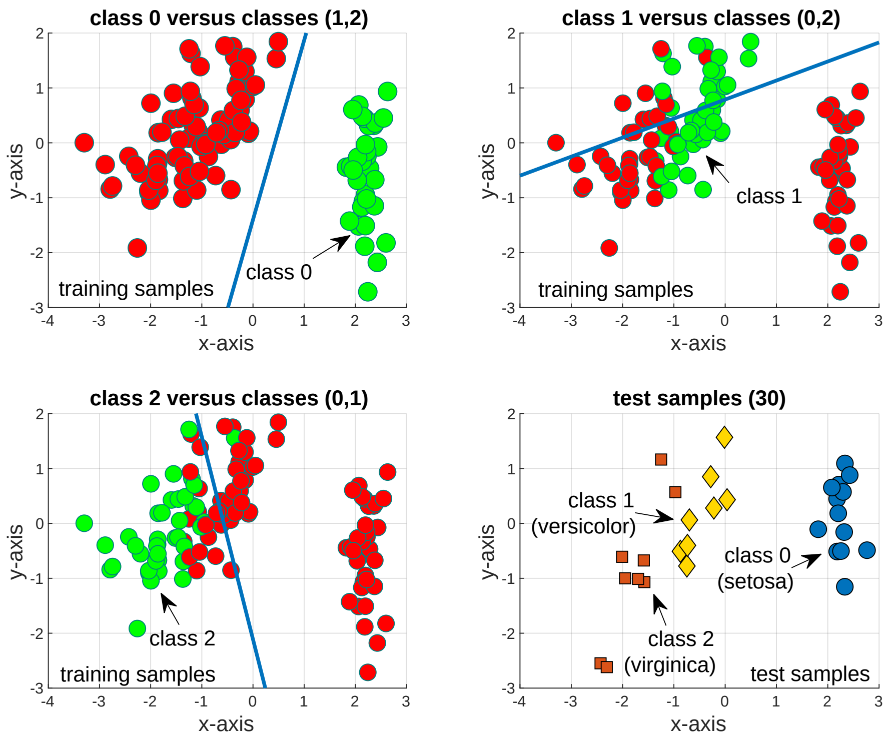

Figure 59.6 The right plot (*bottom*) shows the test data. The other plots show the logistic regression classifier that is obtained in each case. It is clear that it is possible to classify without errors the training data in the top plot (*left*) related to separating class $r = 0$ from $r \in \{1, 2\}$. The same is not true for the other two cases. In particular, it is not possible to separate class $r = 1$ from the combined classes $r \in \{0, 2\}$ by means of a linear classifier.

59.3.2 OvO Strategy

The second technique for multiclass classification is the *one-versus-one* (OvO) strategy. Starting with a total of R classes, there are $R(R-1)/2$ pairwise combinations of individual classes that are possible. For example, if $R = 3$ so that we have three classes, $r \in \{1, 2, 3\}$, then we can group the training data according to the following pairs of classes $(1, 2)$, $(1, 3)$, and $(2, 3)$. In the grouping $(1, 2)$, all data points belonging to classes $r = 1$ and $r = 2$ will be used to train a binary classifier to separate between these classes. In the second grouping $(1, 3)$, all data points belonging to classes $r = 1$ and $r = 3$ will be used to train a binary classifier to separate between these classes. And likewise for the data corresponding to the grouping $(2, 3)$ – see Fig. 59.8.

At the end of this training process, we end up with $R(R-1)/2$ classifiers, one for each pairing of classes. During testing, when the classification machine receives a new feature vector h and wishes to classify it, the procedure is as

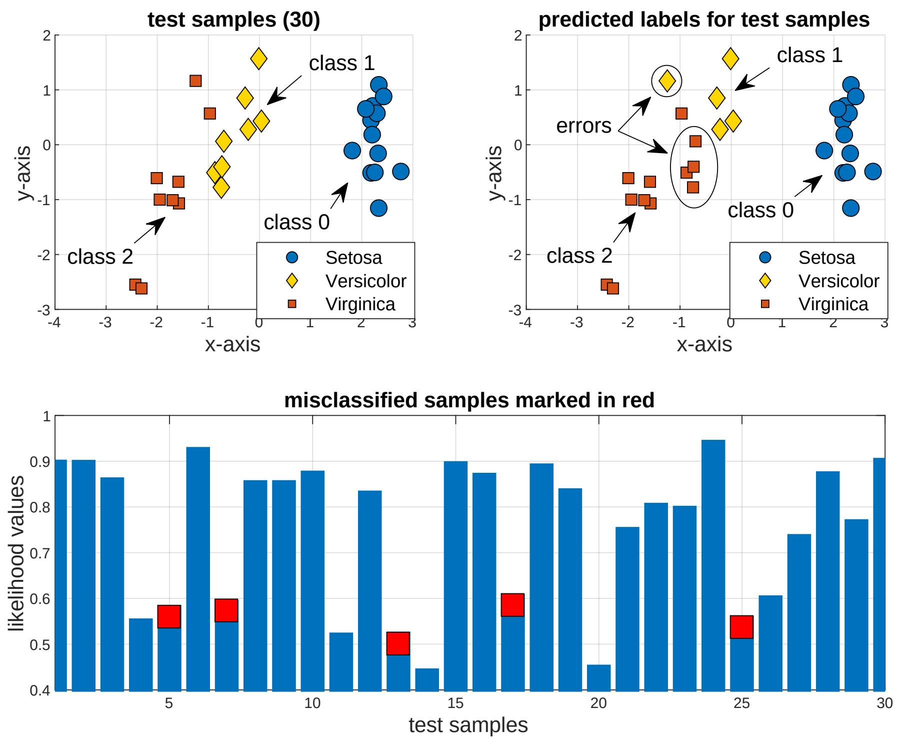

Figure 59.7 The bottom plot shows the *largest* likelihood value for each test sample. The top right plot shows the resulting predicted labels over the test data. It is observed that five test samples are misclassified.

follows. Each classifier generates a classification decision for h (whether it belongs to one of its classes or the other). For example, for the case $R = 3$ described above, assume that h belongs to class $r = 2$. Then, classifier $(1, 2)$ will decide that h belongs to class 2, classifier $(2, 3)$ will also decide that it belongs to class 2, while classifier $(1, 3)$ will issue some wrong decision. The final decision is to select the class that received the largest number of votes from the $R(R-1)/2$ classifiers. One inconvenience of this procedure is that some classes may receive an equal number of votes (which can, for example, be decided by randomly selecting one choice from among the possibilities).

Example 59.3 **(Applying OvO to the iris dataset)** We consider the same setting from Example 59.2 except that we now apply the OvO procedure. There are three classes denoted by $r \in \{0, 1, 2\}$ corresponding to the Setosa ($r = 0$), Versicolor ($r = 1$), and Virginica ($r = 2$) flower types. There are also a total of $N = 150$ samples: 120 of them are selected for training and the remaining 30 samples are used for testing. The plots in Fig. 59.9 show all training samples, along with the pairings that result from considering samples from one class against the samples from another class.

We again extend the feature vectors according to (59.16) and apply five passes of the

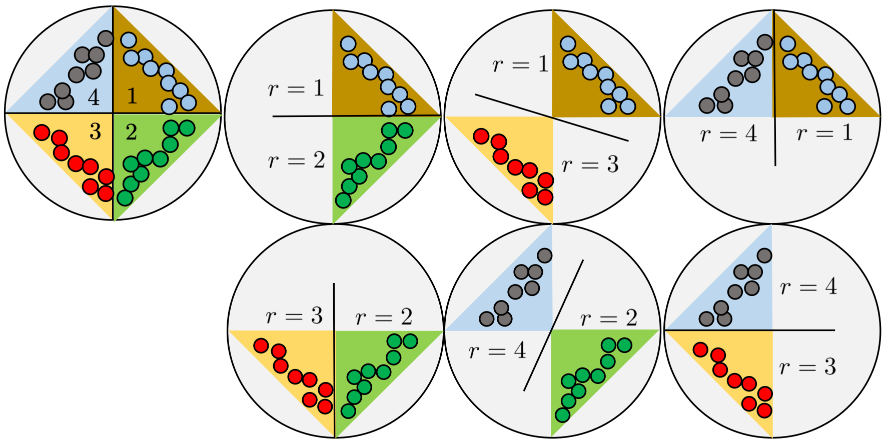

Figure 59.8 In the one-versus-one (OvO) strategy, a collection of $R(R-1)/2$ binary classifiers are designed for a multiclass classification problem involving R cases.

ℓ_2-regularized logistic regression algorithm (59.15) using $\mu = 0.01$ and $\rho = 0.1$. Using random reshuffling, the simulation leads to (where we are showing both the offset and the weight parameters for completeness):

$$\begin{bmatrix} -\theta_{01}^\star \\ \\ w_{01}^\star \end{bmatrix} = \begin{bmatrix} -0.2800 \\ \hline 0.9675 \\ -0.4310 \end{bmatrix}, \quad \text{(to separate class 0 from classe 1)} \tag{59.30a}$$

and

$$\begin{bmatrix} -\theta_{02}^\star \\ \\ w_{02}^\star \end{bmatrix} = \begin{bmatrix} -0.1095 \\ \hline 1.1166 \\ -0.0912 \end{bmatrix}, \quad \text{(to separate class 0 from class 2)} \tag{59.30b}$$

and

$$\begin{bmatrix} -\theta_{12}^\star \\ \\ w_{12}^\star \end{bmatrix} = \begin{bmatrix} 0.1993 \\ \hline 0.5390 \\ 0.3971 \end{bmatrix}, \quad \text{(to separate class 1 from class 2)} \tag{59.30c}$$

Figure 59.10 shows the training data and the test data. It also shows the resulting separating lines. It is clear from the plot on the right in the top row that classes $r = 1$ and $r = 2$ are not separable by a linear classifier. The middle plots in the figure show the original test data and the predicted labels. It is seen that there are 5 misclassifications (out of 30 test samples) resulting in an empirical error rate of 13.33%.

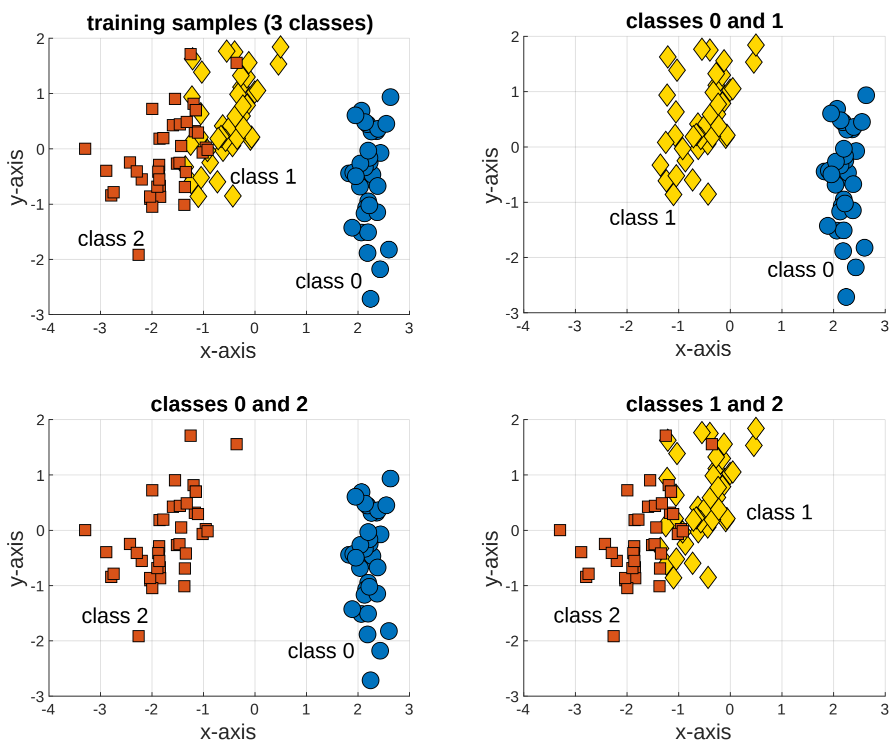

Figure 59.9 The top plot (*left*) shows all training samples from the three classes $r = 0, 1, 2$. The other plots show the pairings that result from considering samples from one class against samples from another class.

59.4 ACTIVE LEARNING

In this section and the next we discuss two important problems in learning where logistic regression plays a supporting role. The concepts discussed here can be applied to other supervised classification algorithms as well. We start with the problem of active learning.

The main objective of *active learning* is to endow a learning algorithm with the ability to select which training samples to use and in what order. The expectation is that by doing so, the classifier will be able to deliver improved performance with a *smaller number* of labeled samples. This is particularly helpful in applications where it is costly to collect labeled data.

59.4.1 Labeled Data

Assume we have a collection of N labeled data pairs $\{\gamma(n), h_n\}$. For simplicity, we assume two classes, $\gamma(n) \in \{\pm 1\}$, although the discussion can be easily extended to multiclass problems — see Example 59.5. Under active learning, the learner

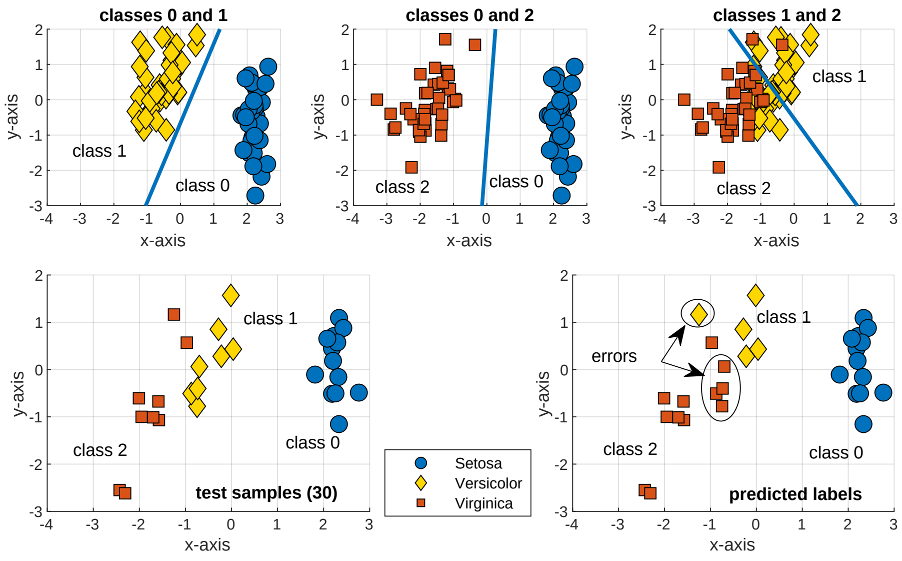

Figure 59.10 The top row shows the pairings of classes and the separating lines that result from logistic regression. It is clear that it is not possible to classify without errors the training data in the top rightmost plot related to separating class $r = 1$ from $r = 2$. The middle plots show the original test data and the predicted labels. It is seen that there are 5 misclassifications (out of 30 test samples).

selects initially a random subset of N_1 training samples, called the *seed*. We refer to this set by the letter $\mathcal{S}$, and denote the unused samples by $\mathcal{U}$. The learner uses the samples in $\mathcal{S}$ to train an initial classifier and arrive at its parameters $(w^\star, \theta^\star)$. This step can be carried out, for example, by using the stochastic gradient logistic regression algorithm (59.15) with or without regularization.

To continue, the classifier will now query the other set, $\mathcal{U}$, repeatedly to decide which of its samples to choose in order to to continue to update the classifier $(w^\star, \theta^\star)$. This procedure is known as *pool-based sampling*, which is one of the more popular schemes in active learning. The learner follows the following steps to carry out the query process:

(a) (**Compute confidence levels and uncertainties**) For each feature $h_n \in \mathcal{U}$ (i.e., for each sample in the pool of unused samples), the classifier evaluates the confidence level that it would have in assigning it to class $\gamma = +1$. We denote this confidence level by

$$p(n) \stackrel{\Delta}{=} \frac{1}{1 + e^{-(h_n^\mathsf{T} w^\star - \theta^\star)}}, \qquad \textbf{(confidence level)} \qquad (59.31)$$

Obviously, the confidence that the classifier has in assigning the same sample to the other class $\gamma = -1$ is $1 - p(n)$. We use the probabilities $\{p(n)\}$ to assess the level of uncertainty that we will have about the true label for h_n. The uncertainty is computed by using the following entropy measure for the

$n-$th sample:

$$\mathcal{H}(n) = -p(n)\log_2 p(n) - (1-p(n))\log_2(1-p(n)), \quad (\textbf{uncertainty}) \tag{59.32}$$

If the set $\mathcal{U}$ is very large, it may become computationally demanding to evaluate these uncertainties for all feature vectors in it. Alternatively, we can sample a random subset of $\mathcal{U}$ and only evaluate the confidence levels and entropy values for the features in this subset.

The following are two popular strategies to select the next sample from $\mathcal{U}$ (or its subset) for use in training. They are referred to as *uncertainty sampling* procedures:

(a1) (**least confidence**) One strategy is to select the sample h_{n^o} for which the learner is least confident about its class, i.e., the one for which $p(n)$ is closet to 0.5:

$$n^o = \underset{n\in\mathcal{U}}{\text{argmin}} \left\{|0.5 - p(n)|\right\} \tag{59.33}$$

Note that we are not selecting the sample with the smallest likelihood value, but rather the sample whose likelihood is closest to 0.5.

(a2) (**most uncertainty**) A related strategy is to select the sample h_{n^o} with the highest entropy value (i.e., the sample for which the algorithm is less certain about its class):

$$n^o = \underset{n\in\mathcal{U}}{\text{argmax}}\ \mathcal{H}(n) \tag{59.34}$$

Under binary classification, this criterion is similar to (59.33) because the entropy measure attains its maximum when $p(n) = 1/2$. For both cases of (a1) and (a2), and under mini-batch implementations, the classifier would query $\mathcal{U}$ to select B samples at once by choosing the B samples with the smallest confidence or largest entropy values.

(b) Once a new training sample $(\gamma(n^o), h_{n^o})$ has been selected, the learner updates its $(w^\star, \theta^\star)$ and repeats the procedure by seeking a new point from the unused set $\mathcal{U}$, updating the classifier, and so forth. Observe that under active learning, the classifier updates its parameters by repeatedly using data that it is least confident about (i.e., samples that are most challenging to classify correctly under the current parameter values).

59.4.2 Unlabeled Data

When all training samples are labeled, active learning helps attain higher accuracy levels with fewer training samples. However, active learning can also be applied to situations where there is a limited amount of labeled data.

We denote the smaller set of labeled data by $\mathcal{S}$ and use it to train an initial classifier $(w^\star, \theta^\star)$ as before. The remaining unlabeled data are collected into the second set $\mathcal{U}$. For each of the feature vectors in $\mathcal{U}$, the active learner again uses expression (59.31) to evaluate how confident it will be about its classification. It then selects the feature vector h_{n^o} with the least confidence level, according to steps (a1) or (a2), and requests that its true label $\gamma(n^o)$ be provided by an *oracle.* The oracle is usually a human annotator. This situation provides an example of a design system involving a human-in-the-loop. Once $(h_{n^o}, \gamma(n^o))$ are known, the active learner uses this data point to update its classifier $(w^\star, \theta^\star)$, and the process repeats. Observe that in this implementation, the learner is only requesting labels for what it believes are the most informative feature vectors within the unlabeled data set. By doing so, the learner prioritizes the features and it becomes unnecessary to collect labels for all features in $\mathcal{U}$ at once, but only on demand.

Example 59.4 (**Active learning applied to a logistic regression model**) We consider the ℓ_2-regularized logistic regression algorithm (59.15) with the offset parameter set to zero, namely,

$$\boldsymbol{w}_n = (1-2\mu\rho)\boldsymbol{w}_{n-1} + \mu\frac{\boldsymbol{\gamma}(n)\boldsymbol{h}_n}{1+e^{\boldsymbol{\gamma}(n)\boldsymbol{h}_n^\mathsf{T}\boldsymbol{w}_{n-1}}}, \quad n \geq 0 \tag{59.35}$$

We generate $N = 2000$ random pairs of data $\{\gamma(n), h_n\}$ according to a logistic model. First, a random parameter $w^a \in \mathbb{R}^{10}$ is selected, and a random collection of feature vectors $\{h_n\}$ are generated with zero-mean unit-variance Gaussian entries. Then, for each h_n, the label $\gamma(n)$ is set to either $+1$ or -1 according to the following construction:

$$\gamma(n) = +1 \quad \text{if} \quad \left(\frac{1}{1+e^{-h_n^\mathsf{T}w^a}}\right) \geq 1/2, \quad \text{otherwise } \gamma(n) = -1 \tag{59.36}$$

A total of $K = 300$ epochs are run over the data, with the data randomly reshuffled prior to each run. We determine the value of the risk function $P(w)$ at the beginning of each epoch, denoted by $P(w_{-1}^k)$. This results in a learning curve showing how the risk value diminishes with the epoch index. We repeat the experiment 10 times and average the learning curves to obtain a smoother curve. The learning curves are plotted in normalized logarithmic scale, namely,

$$\ln\left(\frac{P(w_{-1}^k) - P(w^\star)}{\max_k\{P(w_{-1}^k) - P(w^\star)\}}\right) \tag{59.37}$$

where the minimizer $w^\star$ for $P(w)$ is "obtained" by applying a batch gradient-descent algorithm on the entire set of data points.

The simulation uses $\rho = 1$, $\mu = 0.0001$, and $M = 10$. We assume we know the labels for only 40 data points, while the labels for the remaining 1960 feature vectors will be requested on demand. We run algorithm (59.35) on the available labeled data points and obtain an initial classifier model, $\boldsymbol{w}_{40}$. Subsequently, we follow an active learning approach. We select 20 random samples from among the remaining 1960 samples. For each of the samples in this batch, we compute the confidence level $p(n)$ and retain the sample of least confidence, indexed by n^o according to (59.33). We request the label for this feature vector and use the data point $(\gamma(n^o), h_{n^o})$ to update w_{40} to w_{41}. We repeat the procedure 200 times. Figure 59.11 shows the learning curves for this construction, as well as the resulting weight for the classifier. It is seen that the learner is able to estimate well the entries of the classifier.

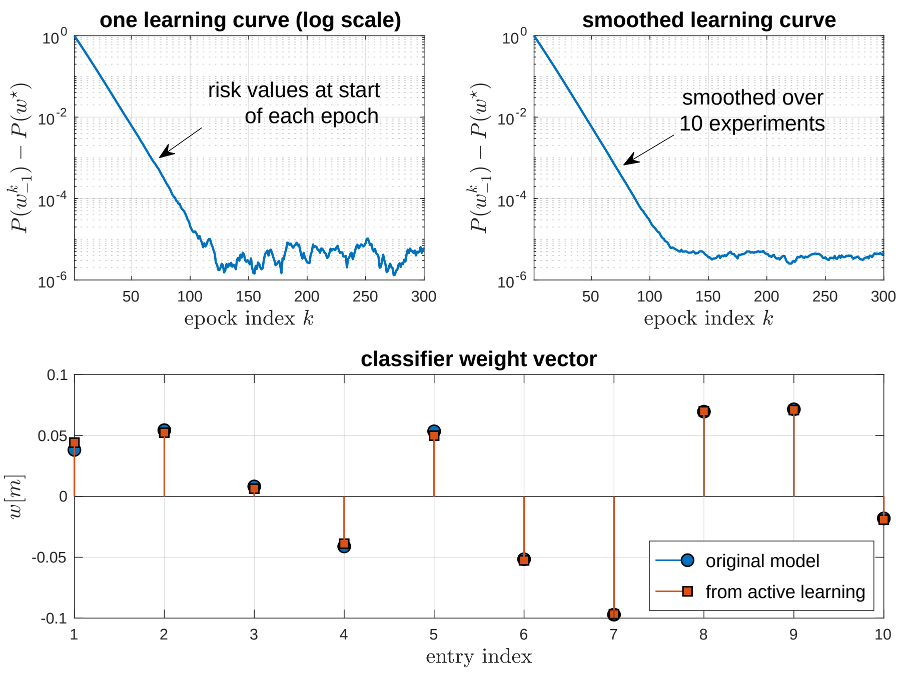

Figure 59.11 (*Top left*) A sample learning curve $P(w^k_{-1})$ relative to the minimum risk value $P(w^\star)$ in normalized logarithmic scale for the stochastic gradient implementation (59.35) under random reshuffling. (*Top right*) Smoothed learning curve obtained by averaging over 10 experiments. (*Bottom*) Actual logistic regression model w^a and the estimate for it obtained through active learning.

Example 59.5 (**Multiclass learning**) The description of the active learning procedure focused on the binary case. When there are multiple classes, say $r \in \{1, 2, \ldots, R\}$, we first apply the OvA procedure to design the binary classifiers $\{(w_1^\star, \theta_1^\star), \ldots, (w_R^\star, \theta_R^\star)\}$. Then, for each feature $h_n \in \mathcal{U}$, expression (59.24) provides the likelihood that it belongs to class r by classifier $w_r^\star$:

$$p'_r(n) \overset{\Delta}{=} \mathbb{P}(\boldsymbol{r} = r \mid \boldsymbol{h}_n = h_n; w_r^\star, \theta_r^\star) \;=\; \frac{1}{1 + e^{-(h_n^\mathsf{T} w_r^\star - \theta_r^\star)}} \tag{59.38}$$

We normalize these likelihoods to add up to one and transform them into a probability distribution as follows:

$$p_r(n) = \frac{p'_r(n)}{\sum_{\ell=1}^{R} p'_\ell(n)} \tag{59.39}$$

Using these values, we can assess the uncertainty about the class label for h_n by computing its entropy:

$$\mathcal{H}(n) = -\sum_{r=1}^{R} p_r(n) \log_2 p_r(n) \tag{59.40}$$

Then, we select the sample n^o as follows. For each sample h_n, we first determine which label appears to be the most likely for it, denoted by:

$$\widehat{r}(n) = \underset{1 \le r \le R}{\text{argmax}} \;\; p_r(n) \tag{59.41}$$

Let $\widehat{p}(n)$ denote the corresponding largest likelihood:

$$\widehat{p}(n) = p_{\widehat{r}(n)}(n) \tag{59.42}$$

Subsequently, we choose n^o by selecting the sample with the least confidence:

$$n^o = \underset{n \in \mathcal{U}}{\text{argmax}} \left\{ 1 - \widehat{p}(n) \right\} \tag{59.43}$$

Table 59.1 provides an example with 4 samples from $\mathcal{U}$ assuming $R = 3$ classes. From the entries in the last column we deduce that $n^o = 1$ for this example.

Table 59.1 Example showing the likelihood values for four samples from $\mathcal{U}$.

sample	$p_1(n)$	$p_2(n)$	$p_3(n)$	$\widehat{r}(n)$	$\widehat{p}(n)$	$1 - \widehat{p}(n)$
1	0.5	0.3	0.2	1	**0.5**	**0.5**
2	0.1	0.2	0.7	3	0.7	0.3
3	0.2	0.2	0.6	3	0.6	0.4
4	0.1	0.8	0.1	2	0.8	0.2

Example 59.6 (**Other classifier structures**) The description of the active learning procedure prior to the examples relied on the training of a binary logistic regression classifier, which allowed us to assess the confidence levels using expression (59.31). Active learning can be applied to other types of classifiers, which may not have an explicit expression for evaluating the confidence level associated with their decisions. Two examples of alternative policies that can be used to select the "least confident" sample include:

(a) Selecting the sample n^o from the unused set $\mathcal{U}$ (or a subset of it) that is closest to the separating hyperplane $(w^\star, \theta^\star)$. We explain in future expression (60.10) that the distance of a generic feature vector h to the hyperplane is given by

$$\text{distance from } h \text{ to hyperplane } (w^\star, \theta^\star) = |h^\mathsf{T} w^\star - \theta^\star| / \|w^\star\| \tag{59.44}$$

(b) For each $h_n \in \mathcal{U}$ (or in a subset of $\mathcal{U}$), we predict its label using $\text{sign}(h_n^\mathsf{T} w^\star - \theta^\star)$. We subsequently select a neighborhood of the closest K features to h from the seed set $\mathcal{S}$ and find out how many of them belong to the same class, say, N_h. Then, the ratio N_h/K is an approximation for the confidence level in assigning h to that class, from which we can estimate $p(n)$ (the confidence in assigning h to class $+1$) and continue from here as before.

59.5 DOMAIN ADAPTATION

We examine next the concept of *domain adaptation* where logistic regression again plays a useful supporting role. Domain adaptation deals with the situation where a learning algorithm is trained on data arising from a particular joint distribution (e.g., height and weight of female and male individuals from a certain geographic region $\mathbb{A}$), and then it is desired to adjust the classifier so that it can operate reliably on data arising from a perturbed distribution (such as height and weight measurements for female and male individuals from another geographic region, $\mathbb{B}$) without the need to carry out a new retraining stage. This is particularly useful in situations where collecting new training data maybe prohibitive. To formulate the domain adaptation problem, we need to distinguish between two domains: the source domain and the target domain.

59.5.1 Source Domain

Let $\boldsymbol{\gamma}$ and $\boldsymbol{h} \in \mathbb{R}^M$ refer to labels and feature vectors and consider a stochastic risk optimization problem of the general form:

$$w^o \overset{\Delta}{=} \underset{w\in\mathbb{R}^M}{\text{argmin}} \; \mathbb{E}\, Q(w;\boldsymbol{\gamma},\boldsymbol{h}) \tag{59.45}$$

where $Q(\cdot;\cdot)$ represents some loss function (such as the logistic loss) and the expectation is over the joint distribution of the data. We have used the notation $f_{\boldsymbol{\gamma},\boldsymbol{h}}(\gamma,h)$ before to refer to this distribution. For reasons that will become clear soon, we will instead use the notation $f_S(\gamma,h)$, with a subscript S, and refer to it as the *source* distribution.

We already know how to apply stochastic optimization procedures to minimize risk functions of the form (59.45). For instance, assume we have steaming data $\{\gamma(n),h_n\}$ arising from the distribution $f_S(\gamma,h)$. Then, we can iterate, for example, a stochastic gradient recursion to get successive estimates for w^o:

$$w_n = w_{n-1} - \mu\nabla_{w^\mathsf{T}} Q(w_{n-1};\gamma(n),h_n), \quad n\geq 0 \tag{59.46}$$

The same algorithm is useful for seeking the minimizer of an empirical risk approximation that solves instead:

$$w^\star \overset{\Delta}{=} \underset{w\in\mathbb{R}^M}{\text{argmin}} \left\{\frac{1}{N_S}\sum_{n=0}^{N_S-1} Q(w;\gamma(n),h_n)\right\} \tag{59.47}$$

assuming that we have access to a collection of N_S data realizations $\{\gamma(n),h_n\}$. We can also incorporate regularization into the empirical risk and solve instead

$$w^\star \overset{\Delta}{=} \underset{w\in\mathbb{R}^M}{\text{argmin}} \left\{\rho\|w\|^2 + \frac{1}{N_S}\sum_{n=0}^{N_S-1} Q(w;\gamma(n),h_n)\right\} \tag{59.48}$$

in which case we would apply the following stochastic gradient recursion:

$$w_n = (1-2\mu\rho)w_{n-1} - \mu\nabla_{w^\mathsf{T}} Q(w_{n-1};\gamma(n),h_n), \quad n\geq 0 \tag{59.49}$$

Once trained, the resulting classifier $w^\star$ is expected to perform well on test data arising from the *same* source distribution as the training data.

59.5.2 Target Domain

There are, however, important situations in practice where the test data arise from a *different* distribution than the training data. For example, it may be easy to collect feature data from some geographic region $\mathbb{A}$ and label the data according to whether an individual has one medical condition or another. And yet we are interested in employing the classifier to discriminate among features collected from another geographic region $\mathbb{B}$ where the distribution of the data follows a different pattern, and where it is either difficult or expensive to collect labels for the data. If we simply train the classifier using the available labeled data from region $\mathbb{A}$, and use it to classify the feature vectors from region $\mathbb{B}$ without proper adjustment, then the accuracy of the classification task will generally be low.

This situation gives rise to a scenario where a classifier is trained by data arising from a *source* distribution and needs to be tested on data arising from a different *target* distribution. Domain adaptation provides one solution to this problem; later in future Example 65.11 we will illustrate another solution method based on *transfer learning* in the context of neural networks. The terms "domain adaptation" and "transfer learning" are often used interchangeably even though the former refers to a narrower situation where the label and feature spaces are the same but only their probability distributions can change.

Domain adaptation can be motivated as follows. We denote the joint distribution for the data from the target domain by $f_T(\gamma, h)$, with a subscript T. We use Bayes rule and factor it as

$$f_T(h,\gamma) \;=\; f_T(h)\, f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h) \tag{59.50}$$

where the first component on the right-hand side is the distribution of the feature data under the target distribution, while the second component continues to be the conditional distribution of the label over the data. Observe that we are writing $f_T(h)$ instead of $f_{\boldsymbol{h}}(h)$ to emphasize that this data distribution is specific to the target domain. Likewise, we will write $f_S(h)$ to refer to the distribution of the feature data in the source domain. One of the common situations for domain adaptation is the case where the feature distribution is different between the source and target domains, i.e.,

$$f_S(h) \neq f_T(h) \tag{59.51}$$

The difference between these distributions needs to be small for better results. At the same time, it is assumed that the conditional distributions remain invariant across both domains:

$$f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h) \text{ is invariant in the source and target domains} \tag{59.52}$$

This case is referred to as domain adaptation under *covariate* shift.

Now, if we had access to labeled training data from the target distribution, we could consider minimizing directly the risk function over the target domain, which would be defined by

$$\begin{aligned} P_T(w) &\overset{\Delta}{=} \mathbb{E}_T\, Q(w;\boldsymbol{\gamma},\boldsymbol{h}) \\ &= \mathbb{E}_{\boldsymbol{h}}\Big\{\mathbb{E}_{\boldsymbol{\gamma}|\boldsymbol{h}}\, Q(w;\boldsymbol{\gamma},\boldsymbol{h})\Big\} \\ &= \int_{h\in\mathcal{H}} f_T(h)\Big\{\mathbb{E}_{\boldsymbol{\gamma}|\boldsymbol{h}}\, Q(w;\boldsymbol{\gamma},\boldsymbol{h})\Big\}dh \end{aligned} \tag{59.53}$$

where in the first line the expectation is over the target distribution $f_T(\gamma,h)$. Unfortunately, we only have access to *labeled* data from the *source* domain and not from the *target* domain. Therefore, we cannot employ an iterative procedure, such as a stochastic gradient algorithm, to minimize $P_T(w)$ because the loss values cannot be evaluated; only realizations for $\boldsymbol{h}$ are available from the target distribution but not for their labels $\boldsymbol{\gamma}$.

59.5.3 Training under Covariate Shift

The main question is whether it is possible to use the available *labeled* training data from the source domain to seek the minimizer of (59.53). The answer is in the affirmative. To show how this can be done, we first rework the risk expression (59.53) as follows:

$$\begin{aligned} P_T(w) &\overset{(a)}{=} \int_{h\in\mathcal{H}} \frac{f_S(h)}{f_S(h)} f_T(h)\Big\{\mathbb{E}_{\boldsymbol{\gamma}|\boldsymbol{h}}\, Q(w;\boldsymbol{\gamma},\boldsymbol{h})\Big\}dh \\ &= \int_{h\in\mathcal{H}} f_S(h)\left\{\mathbb{E}_{\boldsymbol{\gamma}|\boldsymbol{h}}\left(\frac{f_T(h)}{f_S(h)} Q(w;\boldsymbol{\gamma},\boldsymbol{h})\right)\right\}dh \\ &= \mathbb{E}_S\left\{\frac{f_T(h)}{f_S(h)}\, Q(w;\boldsymbol{\gamma},\boldsymbol{h})\right\} \end{aligned} \tag{59.54}$$

where in step (a) we multiplied and divided by the same quantity $f_S(h)$. In the last equality we used the fact that, under covariate shift, the conditional pdf of $\boldsymbol{\gamma}$ given $\boldsymbol{h}$ is the same for both source and target domains. The last expectation is over the joint distribution of the source domain. This derivation assumed that both the source and target distributions for the feature data share the same range space, written as $h \in \mathcal{H}$.

The scalar $\alpha(h) = f_T(h)/f_S(h)$ in (59.54) is referred to as the *importance weight*. The key observation is that by scaling by $\alpha(h)$ we are able to transform the risk $P_T(w)$ from (59.53), which involves averaging over the target distribution, $f_T(h,\gamma)$, to an equivalent expression (59.54) that involves averaging over the source distribution, $f_S(h,\gamma)$. Motivated by expression (59.54), and using the labeled training data $\{\gamma(n), h_n)\}$ from the source domain, we introduce a regularized empirical risk of the form (say, an ℓ_2-regularized risk for illustration purposes):

$$P(w) \overset{\Delta}{=} \rho\|w\|^2 + \frac{1}{N_S}\sum_{n=0}^{N_S-1} \frac{f_T(h_n)}{f_S(h_n)}\, Q(w;\gamma(n),h_n) \tag{59.55}$$

where the ratio $f_T(h_n)/f_S(h_n)$ is evaluated at the source feature vector, h_n. We still cannot minimize this empirical risk because the pdfs, $f_T(h)$ and $f_S(h)$, are not known. All we have is a collection of N_S *labeled* feature vectors $\{h_0, \ldots, h_{N_S-1}\}$ from the source domain and a second collection of N_T *unlabeled* feature vectors $\{h_{N_S+1}, \ldots, h_{N_S+N_T-1}\}$ from the target domain. There are solution methods that rely on estimating the pdfs $f_S(h)$ and $f_T(h)$ from the data, which requires modeling these distributions explicitly, and then applying a stochastic gradient recursion to seek the minimizer of $P(w)$. We describe another solution that relies on the use of a logistic regression classifier.

To begin with, let us introduce a second label, denoted by σ, such that $\sigma = +1$ if feature h arises from the source domain and $\sigma = -1$ if h arises from the target domain. It can be easily verified that — see Prob. 59.16

$$\alpha(h) \overset{\Delta}{=} \frac{f_T(h)}{f_S(h)} = \frac{\mathbb{P}(\boldsymbol{\sigma} = +1)}{\mathbb{P}(\boldsymbol{\sigma} = -1)} \frac{\mathbb{P}(\boldsymbol{\sigma} = -1|h)}{\mathbb{P}(\boldsymbol{\sigma} = +1|h)} \tag{59.56}$$

where the first ratio, $\mathbb{P}(\boldsymbol{\sigma} = +1)/\mathbb{P}(\boldsymbol{\sigma} = -1)$, is a measure of the relative frequency of samples arising from the source and target domains. These probabilities can be estimated as follows:

$$\widehat{\mathbb{P}}(\boldsymbol{\sigma} = +1) = \frac{N_S}{N_S + N_T}, \quad \widehat{\mathbb{P}}(\boldsymbol{\sigma} = -1) = \frac{N_T}{N_S + N_T} \tag{59.57}$$

The rightmost ratio in (59.56) given by $\mathbb{P}(\boldsymbol{\sigma} = -1|h)/\mathbb{P}(\boldsymbol{\sigma} = +1|h)$, can be estimated by introducing a separate classifier to distinguish between features arising from one domain or the other. For example, we can train a logistic regression classifier and determine its parameters (w^x, θ^x) to separate between classes $\sigma = +1$ and $\sigma = -1$; it should be noted that these two classes may not generally be linearly separable (in which case we can use other classifier structures, such as kernel-based learning as discussed in a future chapter). We have available a total of $N_S + N_T$ feature vectors for this classification task, with N_S of them belonging to class $\sigma = +1$ and N_T of them belonging to class $\sigma = -1$. Let $\{h_n, \sigma(n)\}$ refer to all available feature vectors and their labels (source or target features). Then, we can learn (w^x, θ^x) by iterating the logistic regression algorithm:

$$\widehat{\sigma}(n) = h_n^{\mathsf{T}} w_{n-1}^x - \theta^x(n-1) \tag{59.58a}$$

$$\theta^x(n) = \theta^x(n-1) - \frac{\mu\sigma(n)}{1 + e^{\sigma(n)\widehat{\sigma}(n)}} \tag{59.58b}$$

$$w_n^x = (1 - 2\mu\rho) w_{n-1}^x + \frac{\mu\sigma(n) h_n}{1 + e^{\sigma(n)\widehat{\sigma}(n)}} \tag{59.58c}$$

Once this classifier is trained, we use its limiting parameters (w^x, θ^x) to estimate the probabilities that are needed in (59.56). Using expression (59.6), we can write

$$\mathbb{P}(\boldsymbol{\sigma} = +1|h) = \frac{1}{1 + e^{-(h^{\mathsf{T}} w^x - \theta^x)}} \tag{59.59}$$

$$\mathbb{P}(\boldsymbol{\sigma} = -1|h) = \frac{1}{1 + e^{(h^{\mathsf{T}} w^x - \theta^x)}} \tag{59.60}$$

Substituting into (59.56) we get

$$\begin{aligned}\alpha(h) &= \frac{f_T(h)}{f_S(h)}\\ &\approx \frac{N_S}{N_T}\,\frac{1+e^{-(h^\mathsf{T}w^x-\theta^x)}}{1+e^{(h^\mathsf{T}w^x-\theta^x)}}\\ &= \frac{N_S}{N_T}e^{-(h^\mathsf{T}w^x-\theta^x)}\\ &= \frac{N_S}{N_T}\,e^{-\widehat{\sigma}}\end{aligned} \tag{59.61}$$

where we are defining the predicted label:

$$\widehat{\sigma} \;\stackrel{\Delta}{=}\; h^\mathsf{T}w^x - \theta^x \tag{59.62}$$

A second method to estimate the ratio $\mathbb{P}(\boldsymbol{\sigma}=-1|h)/\mathbb{P}(\boldsymbol{\sigma}=+1|h)$ is to employ a $k-$nearest neighbor rule instead of the logistic regression classifier. In this case, a majority vote from the $k-$nearest neighbors to h from among all samples $\{h_n,\sigma(n)\}$ would determine its predicted label and allow us to estimate $\mathbb{P}(\boldsymbol{\sigma}=\sigma|h)$ by counting the number of votes for the label σ divided by k. In this case, the importance weight $\alpha(h)$ would be given by

$$\alpha(h) \approx \frac{N_S}{N_T}\,\frac{N_{-1}}{N_{+1}} \tag{59.63}$$

where N_{-1} is the number of neighbors from class -1 and N_{+1} is the number of neighbors from class $+1$ within the $k-$size neighborhood around h. Clearly, $N_{-1}+N_{+1}=k$.

Once the probabilities needed in (59.56) have been estimated, we can apply a stochastic gradient algorithm to minimize (59.55) using the labeled *source* data $\{\gamma(n),h_n\}$ as follows:

$$w_n = (1-2\mu\rho)w_{n-1} - \mu\alpha(n)\,\nabla_{w^\mathsf{T}}\,Q(w_{n-1};\gamma(n),h_n) \tag{59.64}$$

where $\alpha(n)$ is the scaling factor evaluated at the $n-$th feature vector from the source domain:

$$\alpha(n) \;\stackrel{\Delta}{=}\; \alpha(h_n) \;=\; \frac{N_S}{N_T}e^{-\widehat{\sigma}(n)},\quad \widehat{\sigma}(n)=h_n^\mathsf{T}w^x-\theta^x \tag{59.65}$$

In summary, we arrive at procedure (59.66) for solving the domain adaptation problem under covariate shift. This procedure is based on *instance-weighting* since each data point $(\gamma(n),h_n)$ from the source domain is weighted by the scalar $\alpha(n)$.

Domain adaptation algorithm for minimizing (59.55) under covariate shift

```
given:
  $N_S$ labeled data pairs from source domain: $\{\gamma(n), h_n\}, n = 0, 1, \ldots, N_S - 1$.
  $N_T$ unlabeled features from target domain: $\{h_{n+N_S}\}, n = 0, 1, \ldots, N_T - 1$.

logistic classifier
  • train a logistic classifier to distinguish between source features ($\sigma = +1$)
    and target features ($\sigma = -1$). Let $(w^x, \theta^x)$ denote the resulting classifier.
  • for each source vector $h_n$, determine $\alpha(n) = (N_S/N_T)e^{-(h_n^{\mathsf{T}} w^x - \theta^x)}$.

stochastic gradient algorithm
  • use the labeled source data $\{\gamma(n), h_n\}$ to train:
    $w_n = (1 - 2\mu\rho)w_{n-1} - \mu\alpha(n)\, \nabla_{w^{\mathsf{T}}} Q(w_{n-1}; \gamma(n), h_n)$
  • use the resulting classifier $w^\star$ to determine the labels
    for the target domain features $\{h_t\}, t = 0, 1, \ldots, N_T - 1$:
      $\gamma(t) = +1$, if $h_t^{\mathsf{T}} w^\star \geq 0$
      $\gamma(t) = -1$, if $h_t^{\mathsf{T}} w^\star < 0$
```

(59.66)

Example 59.7 (**Domain adaptation applied to logistic data**) We generate $N_S = 100$ random pairs of source data points $\{\gamma(n), h_n\}$ according to a logistic model. First, a random parameter model $w_S \in \mathbb{R}^2$ is selected, and a random collection of feature vectors $\{h_n\}$ are generated with zero-mean unit-variance Gaussian entries. Then, for each h_n, the label $\gamma(n)$ is set to either $+1$ or -1 according to the following construction:

$$\gamma(n) = +1 \quad \text{if} \quad \left(\frac{1}{1 + e^{-h_n^{\mathsf{T}} w_S}}\right) \geq 1/2, \quad \text{otherwise } \gamma(n) = -1 \tag{59.67}$$

We generate a second set of $N_T = 100$ random pairs of target data points $\{\gamma(n), h_n\}$ by using a similar logistic construction albeit with a different parameter model $w_T \in \mathbb{R}^2$.

A total of $K = 100$ epochs are run over the data, with the data randomly reshuffled prior to each run. Although the source and target features are not linearly separable in this example, we still train a logistic classifier w^x to separate between source and target data, as explained prior to the example. The result is (where, due to the extension, the top entry is the offset θ^x for the classifier):

$$\left[\begin{array}{c} -\theta^x \\ \hline w^x \end{array}\right] = \left[\begin{array}{c} 0.0049 \\ \hline -0.0089 \\ 0.0069 \end{array}\right] \tag{59.68}$$

We use the result to determine the scalars $\{\alpha(n)\}$ for the labeled source samples, and apply the logistic regression recursion to the source data. We again run $K = 100$ epochs and perform random reshuffling at the start of each run, leading to the estimate (no

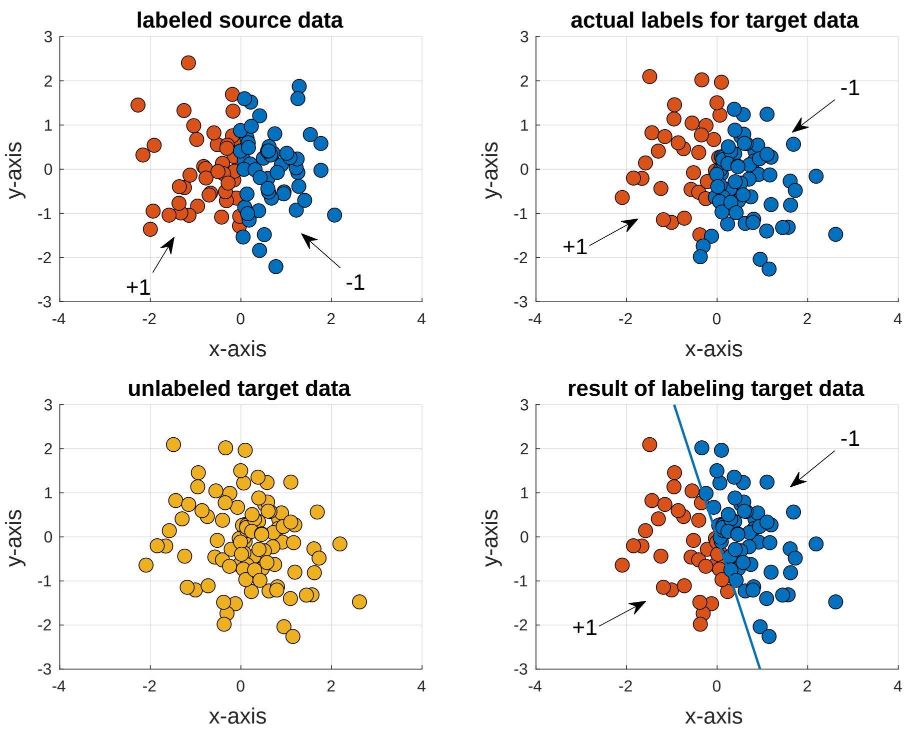

Figure 59.12 (*Top left*) Scatter diagram for the labeled source samples. (*Top right*) Scatter diagram for the labeled target samples. (*Bottom left*) Unlabeled target samples. (*Bottom right*) Predicted labels for the target samples.

offset was used in this case):

$$w^\star = \begin{bmatrix} -0.2108 \\ -0.0667 \end{bmatrix} \tag{59.69}$$

We use $w^\star$ to classify the target samples into classes $+1$ or -1. The simulation uses $\rho = 1$, $\mu = 0.0001$, and $M = 2$. Figure 59.12 shows the scatter diagrams for the labeled source and target data in the top row for comparison purposes. In the implementation, we are actually assuming that the labels for the target data are not known, as shown in the bottom row of the figure, and employ the above construction to predict their labels. For this example, the classification error is 19%.

It is not difficult to observe from repeating this experiment that the domain adaptation procedure can fail more often than desirable. This is because the source and target samples are not generally linearly separable, which results in poor estimates for $\{\alpha(n)\}$.

59.6 COMMENTARIES AND DISCUSSION

Logistic function, logit, and probit. In general, for any $x \in \mathbb{R}$, the logistic function $\sigma(x)$ is defined as the transformation

$$\sigma(x) \;\triangleq\; \frac{1}{1+e^{-x}}, \qquad \textbf{(logistic function)} \tag{59.70}$$

We showed a plot of this function in Fig. 59.1, where it is seen that $\sigma(x)$ satisfies the useful properties:

$$\sigma(0) = 1/2, \quad \lim_{x\to+\infty} \sigma(x) = 1, \quad \lim_{x\to-\infty} \sigma(x) = 0 \tag{59.71}$$

In other words, the function $\sigma(x)$ assumes increasing values in the range $(0,1)$ as x varies from $-\infty$ to $+\infty$. Accordingly, $\sigma(x)$ can be interpreted as a cumulative distribution of some underlying probability density function (pdf), which turns out to be the *logistic distribution*. To see this, let $\boldsymbol{x}$ denote a random variable with mean $\bar{x}$ and variance σ_x^2. Let $a^2 = 3\sigma_x^2/\pi^2$. Then, we say that $\boldsymbol{x}$ has a logistic distribution when its pdf has the form

$$f_{\boldsymbol{x}}(x) \;\triangleq\; \frac{1}{4a}\text{sech}^2\left(\frac{1}{2a}(x-\bar{x})\right), \qquad \textbf{(logistic pdf)} \tag{59.72}$$

where sech$(\cdot)$ refers to the hyperbolic secant function defined by

$$\text{sech}(t) \;\triangleq\; 2/(e^t + e^{-t}) \tag{59.73}$$

The cumulative distribution, which measures $\mathbb{P}(\boldsymbol{x} \leq x)$, is given by

$$F_{\boldsymbol{x}}(x) \;\triangleq\; \frac{1}{1+e^{-\frac{1}{a}(x-\bar{x})}} \tag{59.74}$$

so that $\sigma(x)$ corresponds to the cumulative distribution of a logistic pdf with mean zero and variance $\sigma_x^2 = \pi^2/3$.

Furthermore, for a generic $y \in (0,1)$, we define the logit function

$$g(y) \;\triangleq\; \text{logit}(y) \;\triangleq\; \ln\left(\frac{y}{1-y}\right), \qquad \textbf{(logit function)} \tag{59.75}$$

This function is closely related to the logistic function (59.70). Indeed, one function is the inverse of the other. That is,

$$g(y) = \ln\left(\frac{y}{1-y}\right) \iff y = \sigma(g(y)) \tag{59.76}$$

This observation provides one useful interpretation for the logistic regression model used in (59.6). Indeed, if we make the identifications:

$$y \leftarrow \mathbb{P}(\boldsymbol{\gamma} = \gamma|\boldsymbol{h} = h; w^o), \quad x \leftarrow \gamma h^\mathsf{T} w^o \tag{59.77}$$

then relation (59.6) is simply stating that $y = \sigma(x)$. Consequently, according to (59.76), it must hold that $x = \text{logit}(y)$.

The logistic function (59.70) was introduced by the Belgian mathematician **Pierre Verhulst (1804-1849)** in his studies of models for population growth. Verhulst's work was motivated by his observation that the rate of population growth should be dependent on the population size, which led him to propose a continuous-time differential equation in the paper by Verhulst (1845) of the following form:

$$\frac{dy(t)}{dt} = \alpha\, y(t)\left(1 - \frac{y(t)}{P}\right), \quad y(0) = P_o, \quad t > 0 \tag{59.78}$$

Here, the symbol $y(t)$ denotes the population size at time t, α denotes the growth rate, and P is the maximum population size. The solution that corresponds to the special case $P_o = 1/2$, $P = 1$, and $\alpha = 1$ leads to the logistic function:

$$y(t) = \frac{1}{1 + e^{-t}} \tag{59.79}$$

The designation "logit" for the function (59.75) was introduced by Berkson (1944,1951), whose work was motivated by the earlier introduction of the "probit" function by Gaddum (1933) and Bliss (1934a,b); the term "probit" was used in the references by Bliss (1934a,b). For any $y \in (0, 1)$, the probit function is defined as (compare with (59.75)):

$$g(y) \stackrel{\Delta}{=} \text{probit}(y) \stackrel{\Delta}{=} \Phi^{-1}(y), \qquad \textbf{(probit function)} \tag{59.80}$$

where $\Phi(x)$ denotes the cumulative function of the standard Gaussian distribution with zero mean and unit variance, i.e.,

$$\Phi(x) \stackrel{\Delta}{=} \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\tau^2/2} d\tau \tag{59.81}$$

and $\Phi^{-1}(y)$ is the inverse transformation. The probit is closely related to the $Q-$function for Gaussian distributions since one function is the inverse of the other:

$$g(y) = \text{probit}(y) \iff y = \Phi(g(y)) \tag{59.82}$$

For further details on the history of the logistic function and the logit and probit functions, the reader may refer to the treatment given by Cramer (2003).

Logistic regression. The technique is of broad appeal and has found applications in a wide range of fields, besides machine learning and pattern classification, such as in the life and social sciences. The driving force behind its appeal is that the independent variable $\boldsymbol{\gamma}$ is *not* continuous but *discrete* and can only assume a finite number of possibilities. For example, in the life sciences where logistic regression formulations are popular, it is customary for the variable $\boldsymbol{\gamma}$ to represent the state of a patient (e.g., whether the patient has a certain condition or not), while $\boldsymbol{h}$ collects measurements of biological variables. Likewise, in the social and political sciences, the variable $\boldsymbol{\gamma}$ may represent whether an individual leans towards one political affiliation or another based on observations of certain attributes.

The concept of logistic regression in statistical analysis was originally introduced by Cox (1958), who focused on the binary case — see also the texts by Cox (1969,2006). There is a strong connection with the probit regression concept introduced earlier by Bliss (1934a,b). In the logistic formulation, the conditional probability was modeled in (59.6) by the cumulative function of a logistic pdf, written here more compactly in terms of $\sigma(x) = 1/(1 + e^{-x})$ as

$$\mathbb{P}(\boldsymbol{\gamma} = +1 \mid \boldsymbol{h} = h; w^o, \theta^o) \;=\; \sigma(h^{\mathsf{T}} w^o - \theta^o) \tag{59.83}$$

In comparison, the probit model employs the cumulative distribution of the Gaussian distribution:

$$\mathbb{P}(\boldsymbol{\gamma} = +1 \mid \boldsymbol{h} = h; ; w^o, \theta^o) \;=\; \Phi(h^{\mathsf{T}} w^o - \theta^o) \tag{59.84}$$

where $\Phi(x)$ is defined by (59.81). The logit and probit models for binary classification lead generally to similar results, although the logit formulation is more popular — see, e.g., the discussion in Chapter 33.

Multinomial logistic regression. We focused mostly on the case of binary classification problems in the body of the chapter where $\boldsymbol{\gamma}$ assumes one of two possible values, $\boldsymbol{\gamma} \in \{\pm 1\}$. Variations of logistic regression that handle more than two states are of course possible; these formulations are referred to as *multinomial* or *multiclass* logistic regression problems, and also as *softmax regression problems* — see Prob. 59.14. In this case, it is assumed that there are R classes, labeled $r \in \{1, 2, \ldots, R\}$. Separate parameters (w_r, θ_r) are associated with each class. The conditional distribution of $\boldsymbol{r}$ given the feature $\boldsymbol{h}$ is modeled as the following *softmax* function:

$$\mathbb{P}(\boldsymbol{r} = r|\boldsymbol{h} = h) \;=\; e^{h^{\mathsf{T}} w_r - \theta_r} \left(\sum_{r'=1}^{R} e^{h^{\mathsf{T}} w_{r'} - \theta_{r'}} \right)^{-1}, \quad 1 \le r \le R \tag{59.85}$$

An application of multinomial logistic regression in the context of classification problems appears in Bohning (1992). More detailed treatments on logistic regression, along with examples of applications in several fields, can be found in the texts by Harrell (2001), Cramer (2003), Cox (2006), Hale, Yin, and Zhang (2008), Freedman (2009), Hilbe (2009), Bolstad (2010), Shi *et al.* (2010), and Hosmer and Lemeshow (2013). Further discussions in the context of machine learning, involving the consideration of regularized versions of logistic regression and other methods of solution, appear in several works, e.g., Figueiredo (2003), Ng (2004), Krishnapuram *et al.* (2005), Koh, Kim, and Boyd (2007), and in the texts by McCullagh and Nelder (1989), Bishop (2007), and Theodoridis and Koutroumbas (2008).

Multiclass classification problems. We have already encountered, and will continue to encounter, in our treatment several classification algorithms that can handle *multi-class* classification problems, such as naïve Bayes classifiers, $k-$nearest neighbor classifiers, self-organizing maps, decision trees, random forests, linear discriminant algorithms (LDA), and neural networks. There have also been works in the literature on extending the support vector machine (SVM) formulation of future Chapter 61 to multiclass problems — see, e.g., Vapnik (1998) and the articles by Joachims (1998), Platt (1998), Weston and Watkins (1999), Bredensteiner and Bennett (1999), Crammer and Singer (2001), and Lee, Lin, and Wahba (2004).

In Secs. 59.3.1 and 59.3.2 we discussed a different approach to multiclass classification, in the form of the OvO and OvA techniques, which rely on transforming the problem into a sequence of binary classification problems. While OvO involves training more classifiers than OvA ($O(R^2)$ vs. $O(R)$), it nevertheless uses less training data. It has been observed in the literature, based on extensive experimentation, that if the underlying binary classifiers are tuned well, then using the OvO or OvA strategies works rather well, even in comparison to more sophisticated multiclass classification techniques. The work by Rifkin and Klautau (2004) provides arguments in support of this performance, especially for the OvA method. The OvO and OvA strategies are intuitive and simple and perhaps, for this reason, they have been re-discovered multiple times. For further discussion, the reader may refer to the texts by Bishop (2007) and Hastie, Tibshirani, and Friedman (2009), as well as the articles by Hastie and Tibshirani (1998), Allwein, Shapire, and Singer (2000), Hsu and Lin (2002), Aly (2005), Garcia-Pedrajas and Ortiz-Boyer (2006), and Rocha and Goldenstein (2013).

A third method for transforming a multiclass classification problem into the solution of a sequence of binary classifiers is the *error-correcting output code* (ECOC) method, proposed in the works by Sejnowski and Rosenberg (1987) and Dietterich and Bakiri (1995) — see also the treatments in Allwein, Shapire, and Singer (2000), Hsu and Lin (2002), and Furnkranz (2002). The method relies on using ideas from coding theory to select between a collection of binary classifiers as follows. Assume we are faced with a multiclass classification problem involving R classes. Let B denote the number of base classifiers that we are going to use to attain multiclass classification. We introduce a coding matrix, denoted by C, of size $R \times B$ and whose entries are ± 1. Each row of C is associated with one class, $r = 1, 2 \ldots, R$, and the ± 1 entries on the $r-$th row of C constitute a "codeword" that we are using to represent that particular class. For example, assume $R = 4$ and $B = 6$. Then, one choice for the coding matrix C could be:

$$C = \left[\begin{array}{c|cccccc} r & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 \\ \hline 1 & +1 & -1 & +1 & +1 & -1 & -1 \\ 2 & +1 & +1 & -1 & +1 & +1 & +1 \\ 3 & -1 & -1 & -1 & -1 & +1 & +1 \\ 4 & -1 & -1 & +1 & -1 & -1 & -1 \end{array}\right] \tag{59.86}$$

We are labeling the columns by $\{b_\ell\}$ and these columns have a meaningful interpretation. For example, assume we are classifying images into four classes: cars, fruits, flowers, and airplanes. The value of b_1 could be indicating whether an image has wheels in it ($b_1 = +1$) or not ($b_1 = -1$). If you examine the values appearing in the b_1 column in the above example for C, we find that classes $r = 1$ and $r = 2$ have wheels in them while classes $r = 3$ and $r = 4$ do not. We say that the matrix C provides R codewords (rows); one for each class r. Moreover, each column of C (i.e., each base classifier) divides the training data into two groupings, regardless of their class r. For example, under column b_1, all training data that belong to classes $r = 1$ or $r = 2$ are assigned to class $\gamma = +1$, while the remaining training data that belong to classes $r = 3$ or $r = 4$ are assigned to class $\gamma = -1$. A binary classifier can then be trained on this grouping; this step results in a classifier with parameter vector $w_1^\star$. We repeat for column b_2. In this case, all training data that belong to class $r = 2$ are assigned to $\gamma = +1$, while the remaining training data that belong to classes $r = 1, 3, 4$ are assigned to class $\gamma = -1$. A binary classifier can then be trained on this grouping; this step results in a classifier with parameter vector $w_2^\star$. We repeat for columns $\{b_3, b_4, \ldots, b_6\}$. By the end of this training process, we end up with six trained binary classifiers, $\{w_\ell^\star\}$.

Next, during normal operation, when a new feature vector h is received, we employ the classifiers $\{w_\ell^\star\}$ to determine a codeword representation for h. For example, assume we find that the codeword corresponding to a particular h is

$$\text{codeword}(h) = \begin{bmatrix} +1 & -1 & -1 & +1 & -1 & -1 \end{bmatrix} \tag{59.87}$$

We then determine the "closest" row in C to this codeword, where closeness can be measured either in terms of the Euclidean norm or in terms of the Hamming distance; the Hamming distance between two vectors is the number of entries at which the vectors differ from each other:

$$\widehat{r} \overset{\Delta}{=} \underset{1\le r\le R}{\text{argmin}} \Big\{ \text{Hamming}\left(\text{codeword}(h), C(r,:)\right) \Big\} \tag{59.88}$$

For the above example, we find that $\widehat{r} = 1$ so that feature h is assigned to class $r = 1$. One of the weaknesses of this approach is that it disregards the correlations that may exist between different classes.

Active learning. We described some features of active learning in Sec. 59.4. Under this approach, the learner seeks to improve its performance by being proactive about which training samples to use. Active sampling can be employed for both cases of labeled and unlabeled data. In the latter case, it reduces the amount of labeling that needs to be provided by prioritizing samples for training. Although we illustrated the operation of active learning under the logistic regression classifier, the same methodology can be applied to other classifiers, including neural network structures, using other variations to identify the "least confident" samples as illustrated in Example 59.6 — see, e.g., Cortes and Vapnik (1995), Fujii *et al.* (1998), Tong and Koller (2000), Lindenbaum, Markovitch, and Rusakov (2004), and Settles (2010). The last reference provides a useful survey on active learning. Further useful reading includes the works by MacKay (1992), Cohn, Atlas, and Ladner (1994), Cohn, Ghahramani, and Jordan (1996), Dasgupta (2004), Baram, El-Yaniv, and Luz (2004), Schein and Ungar (2007), Dasgupta and Hsu (2008), and Dasgupta, Hsu, and Monteleoni (2008).

Domain adaptation. The weighted solution (59.66) for the domain adaptation problem was proposed by Bickel, Brueckner, and Scheffer (2007); in the same article they propose a second variant that determines $\{w^x, w^\star\}$ simultaneously by means of a Newton-type algorithm. The conclusion in (59.54) that weighting by the ratio of pdfs, $f_T(h)/f_S(h)$, helps transform expectation over the target distribution $f_T(h)$ to expectation over the source distribution $f_S(h)$ is due to Shimodaira (2000). A similar construction arises in the study of off-policy reinforcement learning algorithms — see Sec. 46.7. There are many other variations of domain adaptation, which differ by the manner by which they estimate the importance weight $f_T(h)/f_S(h)$. In the chapter we discussed two solutions; one based on training a logistic regressor to discriminate between source and target samples, and the other based on using the $k-$nearest neighbor rule. Other approaches that rely on parametric and non-parametric approaches to estimating the distributions $\{f_T(h), f_S(h)\}$ or their ratio are also possible. For example, following Shimodaira (2000), one could assume Gaussian forms for these distributions and estimate their sample means and covariances from the data:

$$\begin{cases} \widehat{\mu}_T = \dfrac{1}{N_T}\displaystyle\sum_{t=0}^{N_T-1} h_t, \quad \widehat{\mu}_S = \dfrac{1}{N_S}\displaystyle\sum_{s=0}^{N_S-1} h_s \\ \widehat{R}_T = \dfrac{1}{N_T-1}\displaystyle\sum_{t=0}^{N_T-1} (h_t-\widehat{\mu}_T)(h_t-\widehat{\mu}_T)^{\mathsf{T}} \\ \widehat{R}_S = \dfrac{1}{N_S-1}\displaystyle\sum_{s=0}^{N_S-1} (h_s-\widehat{\mu}_S)(h_s-\widehat{\mu}_S)^{\mathsf{T}} \\ f_T(h) \sim \mathcal{N}_{\boldsymbol{h}}(\widehat{\mu}_T, \widehat{R}_T), \quad f_S(h) \sim \mathcal{N}_{\boldsymbol{h}}(\widehat{\mu}_S, \widehat{R}_S) \end{cases} \tag{59.89}$$

Another approach to domain adaptation is based on the methodology of *optimal transport* — see, e.g., Courty *et al.* (2016,2017) and Redko, Habrard, and Sebban (2017). In this case, the conditional distributions $f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h)$ are allowed to be different over the source and target domains. We denote them by $f_T(\gamma|h)$ and $f_S(\gamma|h)$. One then seeks a mapping $t(h)$ operating on the source feature vectors such that, after the transformation, the distributions match each other:

$$f_S(\gamma|h) \;=\; f_T(\gamma|t(h)), \;\; \forall\, h \in \text{source domain} \tag{59.90}$$

In this approach, after the mapping $t(\cdot)$ is determined, one applies it to the source data and subsequently trains the classifier directly in the target domain using these transformed vectors. The main intuition is that after the transformation, the source data will behave similarly to the target data.

Good surveys on domain adaptation and transfer learning, including discussions on other approaches, are given by Weiss, Khoshgoftaar, and Wang (2016) and Kouw and Loog (2019). Useful performance results are given by Crammer, Kearns, and Wortman (2008), Mansour, Mohri, and Rostamizadeh (2009), Ben-David *et al.* (2010a,b) Cortes, Mansour, and Mohri (2010), and Germain *et al.* (2013).

PROBLEMS

59.1 Consider the logistic function $\sigma(x) = 1/(1+e^{-x})$. Verify that
(a) $\sigma(-x) = 1 - \sigma(x)$.
(b) $d\sigma/dx = \sigma(x)\sigma(-x)$.
(c) $d\sigma/dx = \sigma(x)(1-\sigma(x))$.
59.2 Consider the ℓ_2-regularized logistic risk

$$P(w) \;\stackrel{\Delta}{=}\; \rho\|w\|^2 \;+\; \mathbb{E}\left\{\ln\left(1+e^{-\boldsymbol{\gamma}\widehat{\boldsymbol{\gamma}}(w)}\right)\right\}, \quad \widehat{\boldsymbol{\gamma}}(w) = \boldsymbol{h}^{\mathsf{T}}w$$

and denote its minimizer by w^o. Prove that
(a) $\|w^o\| \leq \mathbb{E}\,\|\boldsymbol{h}\|/2\rho$.
(b) $\|w^o\|^2 \leq \text{Tr}(R_h)/4\rho^2$, where $\boldsymbol{h}$ is zero-mean and $R_h = \mathbb{E}\,\boldsymbol{h}\boldsymbol{h}^{\mathsf{T}}$.
59.3 Consider the logistic regression algorithm (59.15) without the offset parameter $\boldsymbol{\theta}$ (i.e., set it to zero). Introduce the auxiliary variable $\boldsymbol{d}(n) = +1$ if$\boldsymbol{\gamma}(n) = +1$ and $\boldsymbol{d}(n) = 0$ if$\boldsymbol{\gamma}(n) = -1$. Let further $\sigma(x) = 1/(1+e^{-x})$ denote the logistic function. Show that the logistic regression algorithm can be equivalently re-worked into the following form:

$$\begin{aligned}\boldsymbol{e}(n) &= \boldsymbol{d}(n) - \sigma(\boldsymbol{h}_n^{\mathsf{T}}\boldsymbol{w}_{n-1})\\ \boldsymbol{w}_n &= (1-2\mu\rho)\boldsymbol{w}_{n-1} + \mu\boldsymbol{h}_n\boldsymbol{e}(n)\end{aligned}$$

59.4 Consider the ℓ_2-regularized empirical logistic risk problem:

$$w^\star \;\stackrel{\Delta}{=}\; \underset{w\in\mathbb{R}^M}{\text{argmin}} \;\left\{\frac{1}{2}w^{\mathsf{T}}R_w^{-1}w \;+\; \frac{1}{N}\sum_{n=0}^{N-1}\ln\left(1+e^{-\gamma(n)h_n^{\mathsf{T}}w}\right)\right\}$$

where $R_w > 0$. Let $\sigma(z) = 1/(1+e^{-z})$. Show that $w^\star$ can be written in the form

$$w^\star = \frac{1}{N}\sum_{n=0}^{N-1}\lambda(n)\gamma(n)R_w h_n$$

where the coefficients $\{\lambda(n)\}$ are the derivatives of $\sigma(z)$ evaluated at $z = \gamma(n)\widehat{\gamma}(n)$, i.e.,

$$\lambda \;\stackrel{\Delta}{=}\; \left.\frac{d\sigma(z)}{dz}\right|_{z=\gamma\widehat{\gamma}}, \quad \widehat{\gamma} = h^{\mathsf{T}}w^\star$$

59.5 We continue with Prob. 59.4. Using model $w^\star$, show that the conditional probability of the label variable given the feature vector can be written in the form

$$\mathbb{P}(\boldsymbol{\gamma} = \gamma\,|\,h; w^\star) = \frac{1}{1+e^{-\gamma\widehat{\gamma}}}$$

for the following function of the feature vector h:

$$\widehat{\gamma}(h) \;\stackrel{\Delta}{=}\; \frac{1}{N}\sum_{n=0}^{N-1}\lambda(n)\gamma(n)h^{\mathsf{T}}R_w h_n$$

Conclude that the label γ that maximizes $\mathbb{P}(\boldsymbol{\gamma} = \gamma\,|\,h; w^\star)$ is the one that matches $\text{sign}(\widehat{\gamma}(h))$. *Remark.* For additional discussion on the material in this problem and the previous one, the reader may refer to the work by Jaakkola and Haussler (1999).
59.6 Consider N independent and identically-distributed observations $\{\boldsymbol{\gamma}(n), \boldsymbol{h}_n\}$. For each individual feature $\{\boldsymbol{\gamma}, \boldsymbol{h}\}$, the conditional probability of the label given the feature is modeled according to (59.6). Assume zero offsets in this problem. Verify that the log-likelihood function for the observations, denoted by $\ell(w)$, can be written in the form — compare with (59.11):

$$\ell(w) = \sum_{n=0}^{N-1}\left\{\left(\frac{1+\gamma(n)}{2}\right)\ln\left(\frac{1}{1+e^{-h_n^{\mathsf{T}}w}}\right) + \left(\frac{1-\gamma(n)}{2}\right)\ln\left(\frac{1}{1+e^{h_n^{\mathsf{T}}w}}\right)\right\}$$

Redefine the labels from $\{-1,+1\}$ to $\{0,1\}$ by using the transformation $\gamma \leftarrow (1+\gamma)/2$. Using the new labels, verify that the same log-likelihood function can be written in the following form (which is the negative of the so-called *cross-entropy* risk function encountered later in Sec. 65.7 in the context of neural networks):

$$\ell(w) = \sum_{n=0}^{N}\Big\{\gamma(n)\ln\mathbb{P}(\gamma(n)=1) + (1-\gamma(n))\ln\mathbb{P}(\gamma(n)=0)\Big\}$$

where $\mathbb{P}(\boldsymbol{\gamma} = 1) = 1/(1 + e^{-h^\mathsf{T} w})$ and $\mathbb{P}(\boldsymbol{\gamma} = 0) = 1/(1 + e^{h^\mathsf{T} w})$.

59.7 Show that the log-likelihood function $\ell(w)$ in Prob. 59.6 is concave. In particular, verify that its Hessian matrix relative to w is nonpositive-definite.

59.8 Consider the same setting of Prob. 59.6 with $\gamma(n) \in \{\pm 1\}$ but assume now that we attach a Gaussian prior to the model $\boldsymbol{w}$, say, $\boldsymbol{w} \sim \mathcal{N}_{\boldsymbol{w}}(0, \sigma_w^2 I_M)$. Verify that the MAP estimator that maximizes the joint pdf of $\{\boldsymbol{w}, \boldsymbol{\gamma}(0), \ldots, \boldsymbol{\gamma}(N-1)\}$ given the feature vectors $\{h_n\}$ leads to the ℓ_2-regularized logistic regression solution.

59.9 Consider the same setting of Prob. 59.6 with $\gamma(n) \in \{\pm 1\}$ but assume now that we attach a Laplacian prior to the model $\boldsymbol{w}$. Specifically, the entries $\{\boldsymbol{w}_m\}$ of $\boldsymbol{w} \in \mathbb{R}^M$ are assumed to be independent of each other and follow a Laplace distribution with zero mean and variance σ_w^2:

$$f_{\boldsymbol{w}_m}(w_m) \;=\; \frac{1}{\sqrt{2}\,\sigma_w} \exp\Big\{-\sqrt{2}|w_m|/\sigma_w\Big\}$$

Verify that the MAP estimator that maximizes the joint pdf of $\{\boldsymbol{w}, \boldsymbol{\gamma}(1), \ldots, \boldsymbol{\gamma}(N-1)\}$ given the feature vectors $\{h_n\}$ leads to an ℓ_1-regularized logistic regression solution.

59.10 Let $\sigma(x) = 1/(1+e^{-x})$ refer to the logistic (or sigmoid) function. Consider the second log-likelihood function defined in Prob. 59.6 for labels $\{0, 1\}$, namely,

$$\ell(w) = \sum_{n=0}^{N} \Big\{\gamma(n) \ln \sigma(h_n^\mathsf{T} w) + (1 - \gamma(n)) \ln(1 - \sigma(h_n^\mathsf{T} w))\Big\}$$

Construct the data matrix H whose rows are $\{h_n^\mathsf{T}\}$, the column vector d whose entries are $\{\gamma(n)\}$, and the column vector $s(w)$ whose entries are $\{\sigma(h_n^\mathsf{T} w)\}$, i.e.,

$$H \;\triangleq\; \begin{bmatrix} h_0^\mathsf{T} \\ h_1^\mathsf{T} \\ \vdots \\ h_{N-1}^\mathsf{T} \end{bmatrix}, \quad d \;\triangleq\; \begin{bmatrix} \gamma(0) \\ \gamma(1) \\ \vdots \\ \gamma(N-1) \end{bmatrix}, \quad s(w) \;\triangleq\; \begin{bmatrix} \sigma(h_0^\mathsf{T} w) \\ \sigma(h_1^\mathsf{T} w) \\ \vdots \\ \sigma(h_{N-1}^\mathsf{T} w) \end{bmatrix}$$

Construct also the $N \times N$ diagonal matrix

$$D(w) \;\triangleq\; \text{diag}\Big\{\sigma(h_n^\mathsf{T} w)\Big(1 - \sigma(h_n^\mathsf{T} w)\Big)\Big\}$$

Verify that
(a) $\nabla_{w^\mathsf{T}} \ell(w) = H^\mathsf{T}(d - s(w))$.
(b) $\nabla_w^2 \ell(w) = -H^\mathsf{T} D(w) H$.

59.11 Continuing with Prob. 59.10, we wish to write down Newton recursion (12.197) for *maximizing* the log-likelihood function $\ell(w)$ using a unit-value step-size. Verify that the recursion in this case reduces to the following form over $m \geq 0$:

$$\begin{aligned} D_{m-1} \;&\triangleq\; D(w_{m-1}) \\ z_{m-1} \;&\triangleq\; H w_{m-1} + D_{m-1}^{-1}(d - s(w_{m-1})) \\ w_m \;&=\; (H^\mathsf{T} D_{m-1} H)^{-1} H^\mathsf{T} D_{m-1} z_{m-1} \end{aligned}$$

59.12 Conclude from Prob. 59.11 that the $m-$th iterate is the solution of the weighted least-squares problem

$$w^\star \;\triangleq\; \underset{w \in \mathbb{R}^M}{\text{argmin}} \;\; (z_{m-1} - Hw)^\mathsf{T} D_{m-1} (z_{m-1} - Hw)$$

How is this conclusion related to the iterative reweighted least-squares problem (50.167)?

59.13 Consider a binary classification problem where $\boldsymbol{\gamma} = \pm 1$ and the following risk functions:

$$\begin{aligned} P(c) &= \mathbb{E}\,(\boldsymbol{\gamma} - \widehat{\boldsymbol{\gamma}})^2, \quad &\text{(mean-square-error risk)} \\ P(c) &= \mathbb{E}\,\max\{0,\, 1 - \boldsymbol{\gamma}\widehat{\boldsymbol{\gamma}}\}, \quad &\text{(hinge risk)} \end{aligned}$$

where $\widehat{\boldsymbol{\gamma}} = c(\boldsymbol{h})$ denotes the prediction that is generated by the classifier, $c(\boldsymbol{h})$; we are not limiting the problem statement to linear classifiers. Show that the minima of the above risks over $\widehat{\boldsymbol{\gamma}}$ are given by:

$$\begin{aligned} &\widehat{\gamma} = 2\,\mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h) - 1, \quad &\text{(mean-square-error risk)} \\ &\widehat{\gamma} = \text{sign}\Big(\mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h) - 1/2\Big), \quad &\text{(hinge risk)} \end{aligned}$$

Derive in each case expressions for the confidence level $\mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h)$.

59.14 Refer to the multiclass logistic regression model (59.85) and assume zero offset parameters. Consider a collection of N independent data realizations $\{h_n, r(n)\}$, where $h_n \in \mathbb{R}^M$ is a feature vector and $r(n)$ is its class. Let W collect all models $\{w_r\}$ into its columns and introduce the notation

$$\sigma_{nr} \;\stackrel{\Delta}{=}\; \mathbb{P}(\boldsymbol{r} = r|h_n; W) = e^{h^{\mathsf{T}} w_r} \left(\sum_{r'=1}^{R} e^{h^{\mathsf{T}} w_{r'}} \right)^{-1}, \quad 1 \leq r \leq R$$

Introduce further the $R-$dimensional vectors:

$$\sigma_n \;\stackrel{\Delta}{=}\; \begin{bmatrix} \sigma_{n1} \\ \sigma_{n2} \\ \vdots \\ \sigma_{nR} \end{bmatrix}, \quad \gamma_n = \begin{bmatrix} \mathbb{I}[r(n) = 1] \\ \mathbb{I}[r(n) = 2] \\ \vdots \\ \mathbb{I}[r(n) = R] \end{bmatrix}$$

where $\mathbb{I}[x]$ is the indicator function assuming the value one when the statement x is true and zero otherwise.

(a) Argue that the log-likelihood function is given by

$$\ell(W) = \sum_{n=0}^{N-1} \sum_{r=1}^{R} \mathbb{I}[r(n) = r] h_n^{\mathsf{T}} w_r - \sum_{n=0}^{N-1} \ln \left(\sum_{r'=1}^{R} \exp\Big\{ h_n^{\mathsf{T}} w_r' \Big\} \right)$$

(b) For any model w_r, show that

$$\nabla_{w_r^{\mathsf{T}}}\, \ell(W) = \sum_{n=0}^{N-1} \Big(\mathbb{I}[r(n) = r] - \sigma_{nr} \Big) h_n$$

(c) Collect the column gradient vectors from part (b) into a matrix and conclude that

$$\nabla_W\, \ell(W) = \sum_{n=0}^{N-1} (\gamma_n - \sigma_n)^{\mathsf{T}} \otimes h_n$$

(d) Write down a gradient-ascent iteration for maximizing $\ell(W)$.

Remark. For a related discussion, the reader can refer to Murphy (2012).

59.15 Refer to the probit regression formulation (59.84) for binary classification. Formulate a maximum likelihood (ML) estimation problem for recovering the weight vector and use the ML formulation to motivate an ℓ_2-regularized stochastic gradient probit solution.

59.16 Establish relation (59.56). *Remark.* For additional motivation, see Bickel, Brueckner, and Scheffer (2007).

59.17 Refer to the Poison distribution (5.47). Argue that the canonical link function is given by $g(\mu) = \ln(\mu)$. Show that the empirical risk optimization problem (59.118) reduces to the following Poisson regression problem:

$$w^\star \;=\; \underset{w\in\mathbb{R}^M}{\text{argmax}} \left\{ \frac{1}{N}\sum_{n=0}^{N-1} \Big(\gamma(n)h_n^{\mathsf{T}}w - \exp\{h_n^{\mathsf{T}}w\}\Big) \right\}$$

59.A GENERALIZED LINEAR MODELS

Linear and logistic regression problems are special cases of the family of generalized linear models (GLMs). We revisit these two cases and introduce the generalization. For more details on such models, the reader may refer to the text by McCullagh and Nelder (1989). GLMs were introduced earlier in the work by Nelder and Wedderburn (1972) as a generalization of various regression models. The main intuition is that linear predictors are used to estimate transformations of the conditional mean.

Linear regression

Assume a random variable $\boldsymbol{\gamma} \in \mathbb{R}$ arises from a linear model of the form:

$$\boldsymbol{\gamma} = \boldsymbol{h}^{\mathsf{T}} w + \boldsymbol{v} \tag{59.91}$$

where $\boldsymbol{v}$ is a zero-mean Gaussian random variable that is independent of the random variable $\boldsymbol{h} \in \mathbb{R}^M$. Then, clearly, given $\boldsymbol{h}$, the conditional pdf of $\boldsymbol{\gamma}$ is Gaussian as well:

$$\boldsymbol{\gamma}|\boldsymbol{h} \;\sim \mathcal{N}_{\boldsymbol{\gamma}}(h^{\mathsf{T}}w, \sigma_v^2) \tag{59.92}$$

We denote the mean of this conditional distribution by $\mu = \mathbb{E}\,(\boldsymbol{\gamma}|\boldsymbol{h})$. In this case, μ depends linearly on the observation h through the model parameter $w \in \mathbb{R}^M$:

$$\mu = h^{\mathsf{T}} w \tag{59.93}$$

In linear regression, we estimate $\boldsymbol{\gamma}$ from h by using a similar structure for the *linear* predictor:

$$\widehat{\gamma} = h^{\mathsf{T}} w \tag{59.94}$$

The way we estimate the unknown w is by maximizing the log-likelihood function over a collection of N independent data pairs $\{\gamma(n), h_n\}$:

$$w^\star \;=\; \underset{w\in\mathbb{R}^M}{\text{argmax}} \;\ln\left\{ \prod_{n=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma_v^2}} \exp\Big\{-\frac{1}{2\sigma_v^2}(\gamma(n) - h_n^{\mathsf{T}}w)^2\Big\}\right\} \tag{59.95}$$

which in this case reduces to solving the least-squares problem:

$$w^\star \;=\; \underset{w\in\mathbb{R}^M}{\text{argmin}} \left\{ \frac{1}{N}\sum_{n=0}^{N-1} (\gamma(n) - h_n^{\mathsf{T}}w)^2 \right\} \tag{59.96}$$

Logistic regression

Consider next a situation where the random variable $\boldsymbol{\gamma}$ is binary-valued as in $\boldsymbol{\gamma} \in \{\pm 1\}$, where $\boldsymbol{\gamma}$ assumes its values according to a Bernoulli distribution:

$$\boldsymbol{\gamma}|\boldsymbol{h} \sim \text{Bernoulli}(p) \tag{59.97}$$

The success probability (i.e., the probability of getting $\gamma = +1$) is modeled by means of a logistic function:

$$p = \mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h) = \frac{1}{1 + e^{-h^{\mathsf{T}}w}} \tag{59.98}$$

The mean of the conditional distribution, $\mu = \mathbb{E}\,(\boldsymbol{\gamma}|\boldsymbol{h})$, is now given by

$$\mu = 2p - 1 \tag{59.99}$$

In logistic regression, we estimate $\boldsymbol{\gamma}$ from h by again using a *linear* predictor of the form

$$\widehat{\gamma} = h^{\mathsf{T}}w \tag{59.100}$$

In this case, the predictor does *not* have the same form as μ. However, they can be related to each other. Using (59.98) and the expressions for $\{\mu, \widehat{\gamma}\}$ it is easy to verify that

$$\widehat{\gamma} = \ln\Big(\frac{1+\mu}{1-\mu}\Big) \tag{59.101}$$

That is, $\widehat{\gamma}$ is obtained by means of some logarithmic transformation applied to the conditional mean. The way we estimate w is by maximizing the log-likelihood function over a collection of N independent data pairs $\{\gamma(n), h_n\}$:

$$w^\star \;=\; \underset{w\in\mathbb{R}^M}{\text{argmax}} \;\; \ln\left\{\prod_{n=0}^{N-1} \frac{1}{1+e^{-\gamma(n)h_n^{\mathsf{T}}w}}\right\} \tag{59.102}$$

which reduces to minimizing the logistic empirical risk:

$$w^\star = \underset{w\in\mathbb{R}^M}{\text{argmin}} \;\left\{\frac{1}{N}\sum_{n=0}^{N-1} \ln\left(1 + e^{-\gamma(n)h_n^{\mathsf{T}}w}\right)\right\} \tag{59.103}$$

Generalization

The previous two examples share some common elements:

(a) Each case assumes a particular *model for the conditional pdf* of the target variable $\boldsymbol{\gamma}$ given the observation $\boldsymbol{h}$, namely, for the distribution of $\boldsymbol{\gamma}|\boldsymbol{h}$. The nature of the variables can be different. For example, in one case, $\boldsymbol{\gamma}$ is real-valued but in the other case it is discrete and binary-valued.

(b) Each case assumes a *linear predictor* for the target variable in the form $\widehat{\boldsymbol{\gamma}} = \boldsymbol{h}^{\mathsf{T}}w$, for some model parameter w. This linear construction will be common for all generalized linear models (which explains the qualification "*linear*" in the name).

(c) The predictor $\widehat{\boldsymbol{\gamma}}$ is estimating some transformation of the conditional mean $\mu = \mathbb{E}\,(\boldsymbol{\gamma}|\boldsymbol{h})$. In the linear regression case, the predictor is estimating μ itself, while in the logistic regression case the predictor is estimating $\ln((1+\mu)/(1-\mu))$. We refer to the function that maps the mean to the predictor as the *link function* and denote it by the notation $\widehat{\gamma} = g(\mu)$. Thus, we have

$$g(\mu) = \mu, \qquad \textbf{(for linear regression)} \tag{59.104a}$$

$$g(\mu) = \ln\Big(\frac{1+\mu}{1-\mu}\Big), \qquad \textbf{(for logistic regression)} \tag{59.104b}$$

Clearly, under item (a), there are many possible choices for the conditional pdf model of $\boldsymbol{\gamma}|\boldsymbol{h}$. We will allow the model to belong to the family of exponential distributions. But first, we introduce *canonical exponential distributions*, which take the following form for scalar random variables $\boldsymbol{y} \in \mathbb{R}$:

$$f_{\boldsymbol{y}}(y) \;=\; \exp\left\{\frac{1}{d(\phi)}\Big(\theta y - b(\theta)\Big) + c(y,\phi)\right\} \tag{59.105}$$

where θ is a scalar parameter, and ϕ is the *dispersion* parameter. Several of the exponential distributions we considered in Chapter 5 can be written in this alternative form. Two examples are as follows.

Gaussian case. For Gaussian random variables, we showed in (5.12) that

$$f_{\boldsymbol{y}}(y) = \exp\left\{\frac{1}{\sigma^2}\Big(\mu y - \frac{\mu^2}{2}\Big) - \frac{1}{2}\Big(\ln(2\pi\sigma^2) + \frac{1}{\sigma^2}y^2\Big)\right\} \tag{59.106}$$

We can therefore make the identifications

$$\theta = \mu \tag{59.107a}$$
$$\phi = \sigma^2 \tag{59.107b}$$
$$d(\phi) = \phi \tag{59.107c}$$
$$b(\theta) = \mu^2/2 \tag{59.107d}$$
$$c(y,\phi) = -\frac{1}{2}\Big(\ln(2\pi\phi) + \frac{1}{\phi}y^2\Big) \tag{59.107e}$$

Observe that the mean of the distribution is represented by the parameter θ.

Bernoulli case. For Bernoulli random variables assuming values $\{0,1\}$, we showed in (5.19) that

$$f_{\boldsymbol{y}}(y) = \exp\left\{y\ln\left(\frac{p}{1-p}\right) \;+\; \ln(1-p)\right\} \tag{59.108}$$

and we can make the identifications

$$\theta = \ln\Big(\frac{p}{1-p}\Big) \tag{59.109a}$$
$$\phi = 1 \tag{59.109b}$$
$$d(\phi) = 1 \tag{59.109c}$$
$$b(\theta) = -\ln(1-p) \tag{59.109d}$$
$$c(y,\phi) = 0 \tag{59.109e}$$

Observe that the mean of the distribution is related to the parameter θ since $\mathbb{E}\,\boldsymbol{y} = p$ and, therefore,

$$\theta = \ln\Big(\frac{\mu}{1-\mu}\Big) \tag{59.110}$$

Canonical exponential case. For distributions of the general form (59.105), it is customary to select the parameter θ to play the role of the predictor, which ends up defining a *canonical* choice for the link function. Let us illustrate this construction by reconsidering the logistic regression problem albeit with the classes set to $\{0,1\}$ for illustration purposes. We therefore have that the conditional pdf of $\boldsymbol{\gamma}$ given $\boldsymbol{h}$ is described by the Bernoulli distribution:

$$\boldsymbol{\gamma}|\boldsymbol{h} \;\sim\; \text{Bernoulli}(p) \tag{59.111}$$

where the success probability is modeled as

$$p = \mathbb{P}(\boldsymbol{\gamma} = +1|\boldsymbol{h} = h) = \frac{1}{1 + e^{-h^{\mathsf{T}} w}} \tag{59.112}$$

The mean of the conditional distribution, $\mu = \mathbb{E}(\boldsymbol{\gamma}|\boldsymbol{h})$, is now given by

$$\mu = p \tag{59.113}$$

and it is easy to verify that

$$\widehat{\gamma} = \ln\left(\frac{\mu}{1-\mu}\right) \tag{59.114}$$

which is the same conclusion that would have resulted from setting $\widehat{\gamma} = \theta$ directly in view of (59.110).

Thus, we will assume canonical exponential distributions of the form (59.105) for the conditional distribution $\boldsymbol{\gamma}|\boldsymbol{h}$, namely,

$$f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h) \propto \exp\left\{\frac{1}{d(\phi)}\Big(\theta\gamma - b(\theta)\Big)\right\} \tag{59.115}$$

and use the linear predictor $\widehat{\gamma} = h^{\mathsf{T}} w$ to replace θ, i.e., we parameterize θ in the linear form $\theta = h^{\mathsf{T}} w$. This step implicitly defines a link function that maps $\mu = \mathbb{E}(\boldsymbol{\gamma}|\boldsymbol{h})$ to $\widehat{\gamma}$. The qualification "generalized" in GLM refers to the use of the more general exponential distribution (59.115) in modeling the conditional pdf $\boldsymbol{\gamma}|\boldsymbol{h}$, while the qualification "linear" in GLM refers to the linear model used for $\theta = h^{\mathsf{T}} w$. In this way, the conditional pdf takes the form:

$$f_{\boldsymbol{\gamma}|\boldsymbol{h}}(\gamma|h) \propto \exp\left\{\frac{1}{d(\phi)}\Big(\gamma h^{\mathsf{T}} w - b(h^{\mathsf{T}} w)\Big)\right\} \tag{59.116}$$

Once w is estimated, we end up with an approximation for the conditional pdf of $\boldsymbol{\gamma}|\boldsymbol{h}$, from which inference about $\boldsymbol{\gamma}$ can be performed. The way we estimate w is by maximizing the log-likelihood function over a collection of N independent realizations $\{\gamma(n), h_n\}$:

$$w^\star = \underset{w \in \mathbb{R}^M}{\text{argmax}} \;\; \ln\left\{\prod_{n=0}^{N-1} \exp\left(\frac{1}{d(\phi)}\Big(\gamma h^{\mathsf{T}} w - b(h^{\mathsf{T}} w)\Big)\right)\right\} \tag{59.117}$$

which reduces to maximizing the following empirical risk — see Prob. 59.17:

$$w^\star = \underset{w \in \mathbb{R}^M}{\text{argmax}} \left\{\frac{1}{N}\sum_{n=0}^{N-1}\Big(\gamma(n) h_n^{\mathsf{T}} w - b(h_n^{\mathsf{T}} w)\Big)\right\} \tag{59.118}$$

By doing so, and assuming ergodicity, we are in effect seeking predictions $\widehat{\gamma} = h^{\mathsf{T}} w$ that solve the Bayesian inference problem

$$w^o = \underset{w \in \mathbb{R}^M}{\text{argmin}} \left\{\mathbb{E}\left(b(\widehat{\boldsymbol{\gamma}}) - \boldsymbol{\gamma}\widehat{\boldsymbol{\gamma}}\right)\right\}, \quad \text{s.t. } \; \widehat{\boldsymbol{\gamma}} = \boldsymbol{h}^{\mathsf{T}} w \tag{59.119}$$

REFERENCES

Allwein, E., R. Shapire, and Y. Singer (2000), “Reducing multiclass to binary: A unifying approach for margin classifiers,” *J. Machine Learning Research*, vol. 1, pp. 113–141.

Aly, M. (2005), “Survey of multiclass classification methods,” *Neural Networks*, pp. 1–9.

Baram, Y., R. El-Yaniv, and K. Luz (2004), “Online choice of active learning algorithms,” *J. Machine Learning Research*, vol. 5, pp. 255–291.

Ben-David, S., J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan (2010a), “A theory of learning from different domains,” *Machine Learning*, vol. 79, nos. 1–2, pp. 151–175.

Ben-David, S., T. Luu, T. Lu, and D. Pál (2010b), “Impossibility theorems for domain adaptation,” *Proc. Artificial Intelligence and Statistics Conference* (AISTATS), pp. 129–136, Sardinia, Italy.

Berkson, J. (1944), “Application of the logistic function to bio-assay,” *J. Amer. Stat. Assoc.*, vol. 39, no. 227, pp. 357–365.

Berkson, J. (1951), “Why I prefer logits to probits?,” *Biometrics*, vol. 7, pp. 327–339.

Bickel, S., M. Brueckner, and T. Scheffer (2007), “Discriminative learning for differing training and test distributions,” in *Proc. International Conference on Machine Learning* (ICML), pp. 81–88, Corvallis, OR.

Bishop, C. (2007), *Pattern Recognition and Machine Learning*, Springer, NY.

Bliss, C. I. (1934a), “The method of probits,” *Science*, vol. 79, pp. 38–39.

Bliss, C. I. (1934b), “The method of probits,” *Science*, vol. 79, pp. 409–410.

Bohning, D. (1992), “Multinomial logistic regression algorithm,” *Annals Inst. Stat. Math.*, vol. 44, pp. 197–200.

Bolstad, W. M. (2010), *Understanding Computational Bayesian Statistics*, Wiley, NY.

Bredensteiner, E. J. and K. P. Bennett (1999), “Multicategory classification by support vector machines,” *Computational Optimization and Applications*, vol. 12, pp. 53–79.

Cohn, D., L. Atlas, and R. Ladner (1994), “Improving generalization with active learning,” *Machine Learning*, vol. 15, no. , pp. 201–221.

Cohn, D., Z. Ghahramani, and M. I. Jordan (1996), “Active learning with statistical models,” *J. Artificial Intelligence Research*, vol. 4, pp. 129–145, 1996.

Cortes, C., Y. Mansour, and M. Mohri (2010),“Learning bounds for importance weighting,” *Proc. Advances Neural Information Processing Systems* (NIPS), pp. 442–450, Vancouver, Canada.

Cortes, C. and V. N. Vapnik (1995), “Support-vector networks,” *Machine Learning*, vol. 20, pp. 273–297.

Courty, N., R. Flamary, A. Habrard, and A. Rakotomamonjy (2017), “Joint distribution optimal transportation for domain adaptation,’ *Proc. Advances in Neural Information Processing Systems* (NIPS), pp. 3730–3739, Long Beach, CA.

Courty, N., R. Flamary, D. Tuia, and A. Rakotomamonjy (2016), “Optimal transport for domain adaptation,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 39, no. 9, pp. 1853–1865.

Cox, D. R. (1958), “The regression analysis of binary sequences (with discussion),” *J. Roy. Stat. Soc. B*, vol. 20, pp. 215–242.

Cox, D. R. (1969), *Analysis of Binary Data*, Chapman and Hall, London.

Cox, D. R. (2006), *Principles of Statistical Inference*, Cambridge University Press.

Cramer, J. S. (2003), *Logit Models from Economics and Other Fields*, Cambridge University Press.

Crammer, K., M. Kearns, and J. Wortman (2008), “Learning from multiple sources,” *J. Machine Learning Research*, vol. 9, pp. 1757–1774.

Crammer, K. and Y. Singer (2001), “On the algorithmic implementation of multiclass kernel-based vector machines,” *J. Machine Learning Research*, vol. 2, pp. 265–292.

Dasgupta, S. (2004), “Analysis of a greedy active learning strategy.” in *Proc. Advances Neural Information Processing Systems* (NIPS), pp. 337–344, Vancouver, Canada.

Dasgupta, S., and D. Hsu (2008), "Hierarchical sampling for active learning," in *Proc. International Conference on Machine Learning* (ICML), pp. 208–215, Helsinki, Finland.

Dasgupta, S., D. Hsu, and C. Monteleoni (2008), "A general agnostic active learning algorithm," in *Proc. Advances in Neural Information Processing Systems* (NIPS), pp. 353–360, Vancouver, Canada.

Dietterich, T. G. and G. Bakiri (1995), "Solving multiclass learning problems via error correcting output codes," *J. Artificial Intelligence Research*, vol. 2, no. 263–286.

Figueiredo, M. (2003), "Adaptive sparseness for supervised learning," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1150–1159.

Freedman, D. A. (2009), *Statistical Models: Theory and Practice*, Cambridge University Press.

Fujii, A. T. Tokunaga, K. Inui, and H. Tanaka (1998), "Selective sampling for example based word sense disambiguation," *Computational Linguistics,* vol. 24, no. 4, pp. 573–597.

Furnkranz, J. (2002), "Round robin classification," *J. Machine Learning Research*, vol. 2, pp. 721–747.

Gaddum, J. H. (1933), "Reports on biological standard III. Methods of biological assay depending on a quantal response," *Medical Research Council*, Special Report Series of the Medical Research Council, no. 183, London.

Garcia-Pedrajas, N. and D. Ortiz-Boyer (2006), "Improving multi-class pattern recognition by the combination of two strategies," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 6, pp. 1001–1006.

Germain, P., A. Habrard, F. Laviolette, and E. Morvant (2013), "A PAC-Bayesian approach for domain adaptation with specialization to linear classifiers," *Proc. International Conference on Machine Learning* (ICML), pp. 738–746, Atlanta, GA.

Hale, E. T., M. Yin, and Y. Zhang (2008), "Fixed-point continuation for ℓ_1-minimization: Methodology and convergence," *SIAM J. Optim.*, vol. 19, pp. 1107–1130.

Harrell, F. E. (2001), *Regression Modeling Strategies*, Springer, NY.

Hastie, T. and R. Tibshirani (1998), "Classification by pairwise coupling," *The Annals of Statistics*, vol. 26, no. 2, pp. 451–471.

Hastie, T., R. Tibshirani, and J. Friedman (2009), *The Elements of Statistical Learning*, 2nd edition, Springer, NY.

Hilbe, J. M. (2009), *Logistic Regression Models*, Chapman and Hall.

Hosmer, D. W. and S. Lemeshow (2013), *Applied Logistic Regression*, 3rd edition, Wiley, NY.

Hsu, C.-W. and C.-J. Lin (2002), "A comparison of methods for multiclass support vector machines," *IEEE Trans. Neural Networks*, vol. 13, no. 2, pp. 415–425.

Jaakkola, T. and D. Haussler (1999), "Exploiting generative models in discriminative classifiers," *Proc. Advances Neural Information Processing Systems* (NIPS), pp. 1–7, Denver, CO.

Joachims, T. (1998), "Making large-scale support vector machine learning practical," in *Advances in Kernel Methods – Support Vector Learning*, B. Scholkopf, C. Burges, and A. Smola, *Eds.*, MIT Press, Cambridge, MA.

Koh, K., S. Kim, and S. Boyd (2007), "An interior-point method for large scale ℓ_1 regularized logistic regression, *J. Machine Learning Research*, vol. 8, pp. 1519–1555.

Kouw, W. M. and M. Loog (2019), "An introduction to domain adaptation and transfer learning," *available online at arXiv:1812.11806v2.*

Krishnapuram, B., L. Carin, M. Figueiredo, and A. Hartemink (2005), "Sparse multinomial logistic regression: Fast algorithms and generalization bounds," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 957–968.

Lee, Y., Y. Lin, and G. Wahba (2004), "Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data," *J. Amer. Stat. Assoc.*, vol. 99, no. 465, pp. 67–81.

Lindenbaum, M., S. Markovitch, and D. Rusakov (2004), "Selective sampling for nearest neighbor classifiers," *Machine Learning*, vol. 54, no. 2, pp. 125–152.

MacKay, D. J. C. (1992), “Information-based objective functions for active data selection,” *Neural Computation*, vol. 4, no. 4, pp. 590–604.

Mansour, Y., M. Mohri, and A. Rostamizadeh (2009), “Domain adaptation: Learning bounds and algorithms,” *Proc. Conference on Learning Theory* (COLT), pp. 19–30, Montreal, Canada.

McCullagh, P. and J. A. Nelder (1989), *Generalized Linear Models*, 2nd edition, Chapman & Hall, London.

Murphy, K. P. (2012), *Machine Learning: A Probabilistic Perspective*, MIT Press, Cambridge, MA.

Nelder, J. and R. Wedderburn (1972), “Generalized linear models,” *J. Royal Statistical Society*, Series A (General), vol. 135, np. 3, pp. 370–384.

Ng, A. Y. (2004), “Feature selection, ℓ_1 vs. ℓ_2 regularization, and rotational invariance,” *Proc. Int. Conf. Machine Learning* (ICML), pp. 78–86, Banff, Alberta, Canada, 78-86.

Platt, J. C. (1998), “Fast training of support vector machines using sequential minimal optimization,” In In *Advances in Kernel Methods – Support Vector Learning*, B. Scholkopf, C. Burges, and A. Smola, *Eds.*, MIT Press, Cambridge, MA.

Redko, I., A. Habrard, and M. Sebban (2017), “Theoretical analysis of domain adaptation with optimal transport,” *Proc. Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 737–753, Skopje, Macedonia.

Rifkin, R. and A. Klautau (2004), “In defense of one-vs-all classification,” *J. Machine Learning Research*, vol. 5, pp. 101–141.

Rocha, A. and S. K. Goldenstein (2013), “Multiclass from binary: Expanding One-vs-All, One-vs-One and ECOC-based approaches,” *IEEE Trans. Neural Networks and Learning Systems*, vol. 25, no. 2, pp. 289–302.

Schein, A. I. and L. H. Ungar (2007), “Active learning for logistic regression: An evaluation,” *Machine Learning*, vol. 68, no. 3, pp. 235–265.

Sejnowski, T. J., and C. R. Rosenberg (1987), “Parallel networks that learn to pronounce english text,” *J. Complex Systems*, vol. 1, no. 1, pp. 145–168.

Settles, B. (2010), “Active learning literature survey,” *Computer Sciences Technical Report 1648*, University of Wisconsin–Madison.

Shi, J., W. Yin, S. Osher, and P. Sajda (2010),“A fast hybrid algorithm for large scale ℓ_1-regularized logistic regression,” *J. Machine Learning Research*, vol. 11, pp. 713–741.

Shimodaira, H. (2000), “Improving predictive inference under covariate shift by weighting the log-likelihood function,” *J. Statistical Planning and Inference*, vol. 90, no. 2, pp. 227–244.

Theodoridis, S. and K. Koutroumbas (2008), *Pattern Recognition*, 4th edition, Academic Press.

Tong, S. and D. Koller (2000), “Support vector machine active learning with applications to text classification,” *Proc. International Conference on Machine Learning* (ICML), pp. 999–1006.

Vapnik, V. N. (1998), *Statistical Learning Theory*, Wiley, NY.

Verhulst, P. F. (1845), “Recherches mathématiques sur la loi dáccroissement de la population,” *Nouveaux Mémoires de l’Académie Royale des Sciences et Belles-Lettres de Bruxelles*, vol. 18, pp. 1–42.

Weiss, K., T. M. Khoshgoftaar, and D. Wang (2016), “A survey of transfer learning,” *J. Big Data*, vol. 3, no. 1, https://doi.org/10.1186/s40537-016-0043-6

Weston, J. and C. Watkins (1999), “Support vector machines for multiclass pattern recognition,” *Proc. European Symposium on Artificial Neural Networks*, pp. 219–224, Bruges, Belgium.