18 Statistical Design and Analytical Strategies for Discovery of Disease-Specific Protein Patterns

Ziding Feng, Yutaka Yasui, Dale McLerran, Bao-Ling Adam, and John Semmes

CONTENTS

18.1 Introduction .......................................................... 368
18.2 Statistical Properties of Protein Expression Measurements .................................................. 369
18.3 Study Design .......................................................... 373
  18.3.1 The Issue of Overfitting ........................................... 373
  18.3.2 Use of Training and Test Data Sets and Sample Size Considerations ............................. 374
  18.3.3 Controlling for Potential Confounders by Frequency Matching ...................................... 376
  18.3.4 Replicated Assays ..................................................... 377
  18.3.5 Balance between Comparison Groups and Replicate over the Major Sources of Variance Components .......................................................... 377
18.4 Data Analysis .......................................................... 378
  18.4.1 Identifying Poor-Quality Spectra .................................................................................. 378
  18.4.2 Peak Identification and Alignment .............................................................................. 378
  18.4.3 Reduction of Dimensionality ...................................................................................... 380
  18.4.4 Classifier Construction ............................................................................................... 383
    18.4.4.1 Logistic Regression ............................................................................................ 383
    18.4.4.2 Boosting ............................................................................................................ 385
  18.4.5 Assessment of Prediction Error ..................................................................................... 387
18.5 Conclusion ........................................................................ 388
References .............................................................................. 388
18.1 INTRODUCTION

The rapid advances in proteome technology offer great opportunities for cancer researchers to find protein biomarkers or protein patterns for early detection of cancer. Since many types of cancer are curable if treated early and are incurable at later stages, early detection is an effective way for fighting against cancer.

Bioinformatic and statistical methods for genomic data, in particular the data from Affimatrix microarray or c-DNA spot array experiments, have matured over the past five years and have made contributions to biology and medicine. To make similar headway in proteomics, we need to understand two challenges in proteomic data analyses. First, the number of proteins, due to posttranslational modifications, is even larger than the number of genes. This high dimensionality leads to difficulties in identifying protein biomarkers or protein patterns truly diagnostic for cancer. Not only computational demand, but also the chance of false findings, is high because the dimension of the data is usually much larger than the number of samples under investigation. This is parallel to genomic data but at a larger scale. Second, in the Affimatrix microarray or c-DNA spot array data, the complementary double-helix structure of DNA greatly facilitates the fidelity and reproducibility of the expression data. Such an advantage does not apply to proteomics due to proteins’ three-dimensional structures. Measurement variation occurs not only on protein expression intensity but also on the protein mass quantification. Specifically, the same protein or peptide can appear at different mass values on different proteomics platforms and, worse, in different runs on the same instrument, due to the limitations of the instrument. This creates great difficulties in protein identifications and measurements for using protein patterns as a disease diagnostic tool.

Statistics has contributed greatly in agriculture, industry, technology, and biomedicine. Regardless of the settings, good statistical practice follows three ordered principles. First, we need to understand the data-generating process, the sources of variations, and systematic biases in the process. Second, based on these understandings, we need to design experiments that eliminate or reduce the biases and variations (noises), or at least enable their control and measurement so that rigorous analyses and inference could be made. Third, analysis methods must take into account the data-generating process and must be congruent to the experimental design. This means a good analysis is impossible without correct execution of the first two principles. It also emphasizes the importance of close interactions between laboratory/bench scientists and statisticians from the early experimental planning to the final reporting of findings.

The objective of this chapter is to provide operational guidelines to meet these three principles for biologists and biostatisticians in planning proteomics experiments and analyzing and interpreting their data.

The remaining part of this chapter is organized into three sections corresponding to the above three principles, plus a summary section. We use surface enhanced laser desorption/ionization–time of flight (SELDI-TOF) data as an example, but the principles apply to data from other proteomic platforms, such as matrix-assisted laser desorption/ionization–time of flight (MALDI-TOF). Our discussions focus on using
protein/peptide peaks to form disease classifiers without knowing protein identities. SELDI technology does not directly provide protein identification. Therefore, quantitative methods for peptide/protein identification are not discussed here. We also exclude the statistical design and analysis for formally validating a given disease classifier (e.g., a protein marker or a panel of protein markers/peaks with an explicit rule to combine them for diagnosis). The statistical experimental design and analysis for formal validation of a given classifier are very different from that for classifier construction and would require a whole chapter to discuss them. This chapter answers two questions: (1) how to plan an experiment for studying protein/peptide patterns for disease classification, and (2) after obtaining such data, how to construct a classifier.

18.2 STATISTICAL PROPERTIES OF PROTEIN EXPRESSION MEASUREMENTS

SELDI-TOF utilizes the ProteinChip Technology developed by Ciphergen Biosystems, Inc. Biological samples are first placed on the surface of wells on a protein chip. Proteins bind to the surface with certain chemical (e.g., hydrophobic, hydrophilic, ionic) or biochemical (e.g., antibody, enzyme, receptor) properties. Nonbinding proteins, salts, etc., are then washed way. Matrix, or energy-absorbing molecule (EAM), is then applied to the surface that will absorb the energy from a laser beam and transfer the energy to ionize proteins. The protein mass and its intensity are determined by the time-of-flight technology. Heavier proteins/peptides take a longer time to reach the detector, while lighter ones require less time. These different times to reach the detector are translated into different protein masses. The number of protein particles hitting the detector at a given time point is translated into a protein intensity measure.

There are 8 wells per protein chip, 96 chips per manufacture batch. Assays may be run on different days and/or different machines. Therefore, the sources of variations are: well-to-well variation, chip-to-chip variation, batch-to-batch variation, day-to-day variation, machine-to-machine variation, and sample-to-sample variation. Only the last variation associated with a certain disease phenotype is the signal of scientific interest. All the others are noise that we want to reduce so that the signal-to-noise ratio is increased.

Figure 18.1 shows two spectra from the same serum sample. A visual inspection suggests that the protein mass/charge points are very consistent across spectra. However, the magnified segments reveal that the seemingly aligned peaks are not actually aligned. Typically, the magnitude of SELDI mass accuracy error is about 1,000–2,000 ppm (0.1–0.2% of mass per charge values). This misalignment makes assessment of protein intensity variation at a specified mass/charge point not very meaningful. Therefore, for the materials discussed in this section, we have performed peak identification and alignment of spectra. Baseline subtraction and normalization have been applied on these data at the laboratory and are not discussed here. A detailed discussion of our peak identification and alignment method is given in Section 18.4.
FIGURE 18.1 Two spectra produced from pooled serum sample and the magnified segment from 2950 to 2980 m/z.
In an experiment investigating the sources of biases and variations, a pooled serum sample was measured repeatedly using one SELDI machine. Six chips were taken from each of two batches and assays were run on five different dates, 2 to 4 chips per day. This yields 96 spectra (12 chips × 8 wells per chip).

Figure 18.2 depicts, for each well’s expression intensity, the $t$-statistic for departure from the grand mean over all chips and wells. It indicates that there are some systematic differences in intensity variability among wells, with wells 1 and 8 (the top and bottom wells on a chip) having more deviations from the grand mean intensity values. The systematic variation becomes smaller toward the middle wells. This could occur for many potential reasons; for example, if chips are not in a perfect...
perpendicular position in the machine. In planning a new experiment, either this systematic problem must be corrected technologically, or wells 1 and 8 should not be used for analyzing samples. It also suggests that samples should be assigned to wells at random to avoid systematic bias. Section 18.3.2 will discuss experimental design in depths.

Figure 18.3 shows histograms of the ratio of day-to-day variance to well-to-well variance, and the ratio of chip-to-chip variance to well-to-well variance, for all mass per charge points. It indicates that the chip-to-chip variance and the day-to-day variance are usually much smaller than the well-to-well variance: Note that “variance” refers to the degree of random measurement errors and should be distinguished from “bias,” the systematic measurement errors shown in Figure 18.2. This is a desirable property because the well is the smallest measurement unit and it is relatively easy to reduce the well-to-well variance by using multiple wells per sample and taking their mean.
18.3 STUDY DESIGN

18.3.1 The Issue of Overfitting

One central challenge in study design is the fact that the number of candidate protein peaks under examination for their diagnostic potential is usually substantially larger than the number of biological samples, even after peak identification and alignment that reduce the number of candidate protein peaks dramatically. In a classic regression problem, a model would fit the data extremely well if the degrees of freedom of the model are large enough, or fit the data exactly if the model has the degrees of freedom equal to the sample size; yet such a model does not approximate well the underlying mechanism that generated the data (e.g., proteins and their interactions that specify the observed disease status). This is because the large degrees of freedom tailor the model too finely to the features of observed data; the fitted model describes not only the systematic features of the underlying data-generating mechanism, but also the random features that are unique to the observed set of data. When such an overfitted model is used to predict a new observation, it fails. This phenomenon is called “overfitting.” This issue is so critical for high-dimensional genomic and proteomic data analyses and deserves more discussions below.

The degrees of freedom for a model are a statistical concept. It refers to the number of independent pieces of information used by the model. The more complex a model is, the more degrees of freedom it has. A set of $n$ independent observations has $n$ pieces of independent information. Let’s assume that we have a total of $n$ independent samples, some are from disease cases and the rest are from normal controls, and the biomarker measure is a continuous variable with distinct values for each sample. A simple model using the mean for each of the case and control groups has two degrees of freedom. A model that classifies a sample as “disease” if its value equals any of the values of the case samples and “normal” if its value equals any of the values of the control samples has $n$ degrees of freedom. It classifies the $n$ samples perfectly but it is useless for future prediction. This is an extreme example of overfitting because it does not uncover any underlying mechanism of biology and did not filter out any noise from the data. A good model would approximate the underlying mechanism of biology to a degree consistent to the amount of information in the data; that is, it is not too simple or too complex relative to the amount of information at hand. An overfitting occurs when a model is inappropriately too complex and flexible, relative to the amount of information available. The result of fitting such a complex flexible model to the finite data at hand is a model excellent for describing the data at hand but poor for approximating the underlying mechanism of biology and, consequently, poor for predicting disease classes of new observations.

The large number of protein peaks allows analysts to construct a complex model with degrees of freedom possibly near or equal to the number of samples, leading to overfitting. This overfitting and the resulting overly optimistic claim/interpretation could occur implicitly and the analysts or readers may not realize them.

van’t Veer et al.\textsuperscript{2} examined microarray data with 23,881 genes measured over 78 breast cancer cases: clinically, 44 cases were in the good prognosis group and 34 were in the poor prognosis group. The goal of their analysis was to use gene
expressions to predict prognosis. A microarray predictor was generated by the following process: approximately 5,000 genes were selected initially from 23,881 genes using a fold-change and \( p \) value criterion, of which 231 genes were selected if they had large absolute correlation with the disease class labels. Then, they used a "leave-one-out" cross-validation method to select an optimal subset of 70 genes, constructed a 70-dimensional centroid vector for the 44 good-prognosis cases, and used the correlation of each case with this centroid to choose a cutoff value to form a binary microarray predictor. This classifier was produced by the multistage modeling process described above from a large number of genes. Therefore, overfitting is a potential concern. Did it really overfit?

Tibshirani and Efron\(^3\) did an interesting analysis to address whether there is an overfitting. If one naively uses the 70-gene predictor as a covariate in a logistic regression model together with six other clinical covariates, the odds ratio (OR) of the disease associated with the microarray predictor is 60, while the largest OR from the six clinical predictors is 4.4. The OR of 60 indicates a remarkably strong association between the disease and the predictor. In contrast, Tibshirani and Efron\(^3\) used a "prevalidation" procedure; that is, they set aside 6 cases at a time when performing the above gene/cutoff point selection using the remaining 72 cases. A binary microarray predictor from the 72 cases was used to predict the 6 cases, and the predicted values for the 6 cases were saved as their binary microarray predictor. They repeated this process 12 additional times and obtained a microarray predictor for each of the 78 cases. When they used this predictor in the same logistic regression model, the OR for the microarray predictor dropped from 60 to 4.7, less than the largest clinical OR, 4.9. A more rigorous full cross-validation showed that adding the microarray predictor to the six clinical predictors in logistic regression models only decreased the prediction error from 29.5 to 28.2\%. van’t Veer et al. also performed some sort of cross-validation (but details were not provided in their paper) and the OR for the microarray predictor was 18. The large difference in the ORs for the microarray predictor suggests that an overfitting might have occurred.

The key message from this example is the importance of having an unbiased assessment of true future prediction error of a predictor. Future prediction error is defined as the prediction error of a predictor constructed from training samples on (a large set of) independent new samples from the same population. This can be assessed unbiasedly by separating samples into a training set and a test set at the design stage (Section 18.3.2), or by rigorously estimating it from all available data (Section IVE). The practice of not estimating the future prediction error by an independent test set or a careful statistical approach should be avoided.

### 18.3.2 Use of Training and Test Data Sets and Sample Size Considerations

Section 18.3.1 illustrates the importance of having an independent test data set, not involved in the construction of a predictor, to evaluate the true future prediction error. We recommend this as the first choice for the estimation of future prediction error when there are a large enough number of samples to split into training and test data sets. This should be an essential part in the planning of experiments. When
an independent test data set is not feasible, a careful and thorough cross-validation is necessary to guide the construction of a predictor and assess its performance. The cross-validation is the second choice because there are many ways of performing cross-validation, as seen from the above example, and analysts could use the approach that leads to seemingly “better” results, without realizing that an overfitting has occurred. This section discusses the setting when we have enough samples to split into training and test sets.

How large should each data set be? There is no formal simple rule to determine the size of a training set. This is because the training sample size depends on signal-to-noise ratios in the data and the complexity of a final predictor (the number of protein peaks to be combined and how they are combined). These are unknown before the analysis of data. Because of the high dimensionality and the potential complexity of predictors that could arise, our suggestion is to get as many samples as feasible, usually at least 100 samples for each disease class.

The prediction rule usually has its potential clinical application in mind. Therefore, the test sample size can be determined using a joint confidence region for sensitivity and specificity. Sensitivity is the true positive fraction, the proportion of the disease cases that are classified correctly by the prediction rule. Specificity is the true negative fraction, the proportion of the normal controls that are classified correctly by the prediction rule. Sensitivity and specificity are binomial proportions so the sample size calculation follows the standard one for binomial proportions with additional considerations of the two jointly. The null hypothesis is that either sensitivity or specificity of the rule is lower than a respective predefined unacceptable value. The sample size should be large enough such that if the classification performance is truly better than the specified unacceptable cutoffs, the joint confidence region of sensitivity and specificity will not include the specified unacceptable cutoffs. For example, if we hope a new test has sensitivity and specificity both at 92% and we want above 90% power to rule out the unacceptable sensitivity and specificity of 84% using a 95% joint confidence region, we need 250 diseased and 250 nondiseased samples. Details of this procedure are described in Section 8.2 of Pepe.4

The unacceptable sensitivity and specificity depend on the intended clinical application. For example, for a general population screening for ovarian cancer, due to the very low prevalence of the disease the specificity has to be very high, say above 99%. Otherwise, a lot of false positives will be generated for unnecessary work-up and worry. However, if it will be used for a high-risk population surveillance, for example, a prostate cancer screening for men with moderately elevated PSA levels, a test with a lower specificity can be applied to decide whether the subjects should be evaluated further by biopsy. In the latter situation, high sensitivity is more important than high specificity because the current practice sends all men with moderately elevated PSA levels to biopsy.

The sample size consideration for a test set discussed above is for the situation where we have a well-defined predictor for an intended clinical use and want to assess the predictor’s performance for the intended use. Often we just want to use the test set to check potential overfitting and overinterpretation. In this setting, the majority of the sample, say 70 to 85%, should be used for the training set, especially
when there is not a large number of samples to adequately split to the two sets. This is because at the training stage the construction of a predictor is the main objective and should use as many available samples as possible.

The test set should be kept by a person who is not involved in constructing the classifier. Only the final predictor should be tested on the test set. If an analyst modifies the predictor after seeing the test set performance, or chooses a predictor from multiple predictors based on their test set performance, the final test set performance is overestimated, sometimes extremely, because the test set is becoming the training set when it is used for selecting predictors.

There is another situation where an estimate of future prediction error from even a rigorous assessment differs from the prediction error in an independent test set. This happens when the test samples and the training samples are not from the same population. Differences in subject selection criteria, sample processing, and storage conditions could lead to this situation. The definition of “population” must, therefore, include subject selection, sample processing, and storage. Samples are from the same population if the subject selection criteria, the sample processing, and storage conditions are consistent.

When samples are collected from multiple laboratories, these conditions must be consistent across all laboratories. If samples are processed under different conditions in one of the labs, bias could occur and a predictor constructed from this lab’s data will not be consistent with data from the other labs because the samples are not from the same population. If samples are from multiple labs, one way to perform a cross-validation is to leave one lab’s data out, using the remaining data to construct a predictor that is then applied to the data from the lab left out to estimate the prediction error. Perform this in turn for all labs and then combine the prediction errors (“leave-one-lab-out cross-validation”). If all predictors are similar and performed similarly well, that means the data were collected in a consistent way or at least any inconsistency did not affect protein profiles. We could then combine all data to construct one predictor. Otherwise, it is necessary to identify which lab’s samples appear abnormal and potentially down-weight or discard their data. In a prospective study requiring sample collection from multiple sites, it is absolutely necessary to develop a formal protocol that details subject selection, sample processing, storage, and assay procedures, and all sites must follow the protocol closely. This is a general good practice for all multicenter studies, but it is particularly important when the measurements of interest are high-dimensional because small systematic bias could accumulate in multidimension and therefore has a big impact on future prediction error.

18.3.3 Controlling for Potential Confounders by Frequency Matching

In comparing control and cancer subjects with respect to protein profiles, it is advantageous to select controls such that they have the same distributions as cancer cases for all factors that might distort the associations of protein profiles with the disease, such as smoking status, age, race, family history of cancer, etc. Such factors are called potential confounders and they may be correlated with both disease status
and diagnostic protein markers. Imbalance between cases and controls with respect to confounders will introduce bias. Often an individual match is not necessary and a frequency matching will be adequate; e.g., two groups have the same percent of smokers, age distribution, racial mix, etc. If the individual matching is used, it would be advantageous to account for it in the analysis procedure (e.g., by using conditional logistic regression).

Potential confounders could also arise from sample collection, processing, storage, and assay procedures. Even when samples are collected, processed, and stored in the same way, if case and control samples are assayed at different times but are not consistent, potential confounding could occur. As implied by Figure 18.2, even the allocation of samples to the wells on ProteinChips could introduce bias. Because we can never know all confounding factors, the safest way is to randomly assign cancer and control samples to wells, chips, processors, and assay dates, balancing the number of control and cancer samples in each chip. Assay technicians should be blinded on disease status when performing the assays.

18.3.4 Replicated Assays

Each sample should be assayed at least twice, preferably three or more times repeatedly. As we will see in Section 18.4.1, poor spectra with extremely low signal-to-noise ratios do occur. With three measurements, we have information to judge whether an observed abnormal spectrum is a poor-quality spectrum due to instrument/assay, or a correct spectrum from a poor-quality sample. In addition, an average of three replicated intensity measurements would have a standard deviation that is $1/\sqrt{3}$ times (less than 58% of) the standard deviation of a single measurement. This is a simple statistical scheme to reduce the noise level in the intensity measurements. The exceptional setting where multiple measurements cannot be taken is when the assay cost is far more than the cost of obtaining specimen. This is not the case for SELDI and MALDI experiments.

18.3.5 Balance between Comparison Groups and Replicate over the Major Sources of Variance Components

The variance components analysis in Figure 18.3 could help us in deciding how to allocate control and cancer samples in the experimental design. The principle is to balance the control and cancer sample allocation on the major source of variance components. If the well-to-well variation is very small, not achieving balance between cancer and control samples on well positions is not critical. Figure 18.3 indicates, however, that this variance component (well-to-well) is indeed large. Therefore, balancing the well positions between two disease groups is important. In particular, allocating a specific type of samples (e.g., QC samples, disease samples, etc.) to a specific well position in each chip would be a bad idea.

Another use of variance component analysis is to guide where to place replicates. We often observe laboratory scientists taking duplicates in the same assay run for convenience, e.g., on the same chip. By doing it this way, however, they are likely to get consistent measurements on the same sample; this is not a good practice in
taking replicates. In fact, this approach leads to the largest variance of an average intensity of the replicate measurements because the replicates are taken at the well level and they could only reduce the well-to-well variance component. If the nesting structure is in the order of well, chip, assay day, and chip batch, for example, the average of replicate measurements would reach the smallest variance when replicates are placed at the chip batch level. The reason is that when we replicate at the chip batch level, we automatically replicate all factors nested within it, as long as we have a random assignment. If we replicate wells within chip, only wells are replicated but nothing above it is replicated. If, for practical reasons, we want to replicate at a level lower than the top, say at the chip level, it is then important to make sure that the levels with large variance components get replication. For example, from Figure 18.3, the well-to-well variance is the largest one. Replication at the well level, though not optimal, is not unreasonable if the logistic for a higher-level replication is very difficult.

18.4 DATA ANALYSIS

18.4.1 IDENTIFYING POOR-QUALITY SPECTRA

Sometimes it is easy to determine a poor-quality spectrum by visual inspection but often it is hard to decide (Figure 18.4). This is particularly true when they are at the borderline. We developed a logistic regression model to quantify the probability that a spectrum is of poor quality using three predictors:

1. Square root of mean square errors of intensity (poor quality spectra tend to have higher noise)
2. Autocorrelation (poor quality spectra tend to have low correlation between adjacent points)
3. Maximum intensity at a prechosen range of mass-per-charge values where almost all spectra showed a strong peak (poor-quality spectra tend to lack this strong peak)

The first two calculations are restricted to low molecular weight (2,000 to 3,170 Daltons) where high-intensity values are typically observed. Figure 18.5 shows the classification of “good” vs. “bad” spectra by this model. This approach is more objective and systematic than the subjective visual inspection and one can easily change the cutoff point to be more or less conservative.

18.4.2 PEAK IDENTIFICATION AND ALIGNMENT

As Figure 18.1 indicated, the error in mass accuracy poses a challenge in data analysis because the same protein or peptide could appear at slightly different mass-per-charge values across samples. Although many analysts use SELDI-TOF data without defining and aligning peaks for biomarker-profile identification, we believe these are essential procedures before using the data for the marker identification.
To align peaks, we first need to define them. Peaks can be identified by using the manufacturer’s internal algorithm. Our alternative simple definition calls a point a “peak” if it is the maximum intensity in its nearest \( \pm N \) mass per charge points. By trial and error with visual inspection, we found \( N = 10–20 \) as reasonable.

After peaks are identified by the simple rule, we look at every mass per charge point and a window around it: the window consists of all points with mass per charge values within \( \pm 0.2\% \) (or \( \pm 0.1\% \); this is equal to the mass accuracy of the instrument) of the point under inspection. We count the number of peaks in the window across all spectra, pick up the mass per charge point that has the largest number of peaks in its window, and use this point as the first aligned mass per charge point with an

**FIGURE 18.4** Triplicate spectra from same serum sample. The middle one indicates a bad spectrum.
aligned peak value of each spectrum set at its maximum intensity value within the window. Note that a spectrum without any peak in the window still gets an aligned peak value, but it would be relatively low. Now this first window is taken away from the mass-per-charge axis and we repeat the process (i.e., pick a mass per charge point that has the largest number of peaks in its window, use this point as the aligned mass per charge point, and take out the window from the remaining mass per charge axis) until no peak is left to be aligned in any spectrum. Figure 18.6 indicates four spectra before and after the peak alignment. See Yasui et al. \(^5\) for details of this procedure.

**18.4.3 Reduction of Dimensionality**

Now we are ready to construct a classifier. Depending on the peak identification and alignment methods, the number of potential predictors (aligned peaks) could still be on the order of a few dozen to a few thousands. If the number is big, it is hard to construct a classifier from the pool of potential predictors directly. One way to reduce the dimensionality is to filter out most of the peaks and only use the peaks with some promising features to be combined for a powerful classifier. In microarray analyses, fold changes, \(t\)-statistics or correlations between two disease groups are often used for filtering. However, as Pepe, et al. \(^6\) pointed out, it makes more sense...
to use sensitivity and specificity to rank the candidate genes for further investigation for disease diagnosis and early detection purposes. We have used a similar idea. Sensitivity and specificity are two measures and they change with the cutoff point. It would be easier to use a one-dimensional measure as a filtering criterion. Two sensible one-dimensional measures are partial area under the receiver operating characteristic (ROC) curve (PAUC) and Yuden distance (Figure 18.7). Yuden distance is defined by the sum of sensitivity and specificity minus 1. The range of Yuden distance is from 0 to 1, with 1 indicating zero classification error and 0 indicating no diagnostic capacity at all. Thus, filtering peaks by Yuden distance would give an equal weight to sensitivity and specificity.

An ROC curve is the functional curve of sensitivity against (1-specificity). It summarizes the operating characteristics of a predictor with continuous values. Each point on the curve corresponds to a cutoff point for a predictor (e.g., test positive for cancer if the predictor value is above the cutoff point, otherwise test negative). PAUC can be defined as the area under the ROC curve from specificity 1 to a predefined specificity, say 0.9. Choosing protein peaks that have larger PAUC values will preselect protein peaks that have high specificity. Similarly, we can define PAUC as the area under the ROC curve from the specificity corresponding to a predefined high sensitivity, say 0.9, to specificity 0 (i.e., sensitivity 1), and select protein peaks with large
PAUC values. This preselects protein peaks that have high sensitivity. We could predetermine the top percentiles for the two types of PAUC to be selected to control the number of protein peaks filtered out. Note that the multiple criteria (i.e., Yuden distance and the two types of PAUC) can be combined for filtering by taking the union of the selected peak set by each criterion. This is advantageous since we wish to keep
the peaks that are not necessarily discriminatory globally, but either highly specific or sensitive, to capture some smaller subgroups of cancer cases or controls.

## 18.4.4 Classifier Construction

There are many classifier construction algorithms: most of them are from either statistical science or computer science/machine learning. A good discussion of this topic can be found in a recent book by Hastie, et al. We describe here two algorithms that we found to have good performance and interpretations. They are logistic regression and boosting.

### 18.4.4.1 Logistic Regression

Logistic regression is a standard statistical regression algorithm to model a binary outcome \( Y \) (e.g., \( Y = 1 \) for cancer vs. \( Y = 0 \) for noncancer) using an array of candidate predictors (continuous, binary, categorical, or mixture of them). It is one of the most popular analysis methods in the field of epidemiology. Logistic regression assumes the following model:

\[
\text{Logit}(E[Y]) = b_0 + b_1X_1 + b_2X_2 + \cdots + b_MX_M, \quad Y \sim \text{Independent Bernoulli}(E[Y])
\]

where \( \text{Logit}(a) = \ln(a / (1 - a)) \), \( X_i \) is \( i \)th predictor (i.e., \( i \)th peak in the spectra), and \( b_i \) is the regression coefficient associated with the \( i \)th predictor. The value of \( b_i \) has an interpretation as the increase in log odds of cancer associated with one unit increase of \( X_i \).

For a binary predictor (i.e., having peak or not having peak at a specific mass per charge point), it is the log odds ratio of cancer for a patient with...
a peak at this mass-per-charge point compared to a patient without a peak there, holding all other factors constant. After the model is finalized, we can predict the probability of a spectrum in question being a cancer rather than noncancer using the formula: probability of cancer = \frac{1}{1 + \exp\{- (b_0 + b_1X_1 + b_2X_2 + \cdots + b_MX_M)\}}.

If the probability is larger than 0.5, the prediction is more indicative for cancer. Otherwise, it is more indicative for noncancer. Users could modify this cutoff point if sensitivity and specificity have different importance under their particular setting. Note that the intercept term \( b_0 \) depends on the ratio of cancer vs. noncancer cases in the data set: if the ratio changes from a training data set to a test data set, for example, caution must be exercised in applying a cutoff because the intercept term from the training set is not directly applicable to the test set. More discussions on logistic regression models can be found in Harrell. All major commercial statistical software packages, such as SAS, SPSS, STATA, BMPP, and S-Plus include logistic regression analysis capability.

Even with peak alignment and filtering, there may still be a few hundred peaks as potential predictors for logistic regression. We can use a forward variable-selection procedure: we first identify, among all \( X \)'s, the peak that has the best prediction power. That means that, among all logistic models with just one peak in the model, the model with this peak has the minimum misclassification error. After selecting the first peak for the model, we identify and add the best peak of all remaining peaks, given the first peak in the model. We continue the selection by adding the best peak, given the previously selected peaks in the model.

When shall we stop the selection? As we noted before, we could add more and more predictors and eventually overfit the data. We could use a cross-validation procedure to determine when to stop and minimize overfit. The idea of a tenfold cross-validation, for example, is as follows. We divide the data into ten equal parts randomly and only use nine parts to construct a classifier with the best \( p \) predictors. We then use the classifier to predict the remaining 10% of the data not used in classifier construction. This 10% data act like a validation data set. We continue this process with a fixed \( p \) for all ten parts. By the end, there is precisely one classification result for each sample that can gauge the performance of the classifier-building procedure with the fixed \( p \) on new independent data. After trying a set of different values for the number of predictors in the model \( p \), the optimal \( p \) could be determined by the one that gives the smallest cross-validation error. Alternatively, we can fix the significance level \( p \) of a newly added protein peak, instead of the number of peaks in the model, to determine the optimal stopping rule.

There is an important implication: if we use cross-validation only once for a prespecified number of predictors in the model, the cross-validation error is a good estimate of future prediction error. However, if we use cross-validation for model selection, such as described above, we still could overfit slightly. This is because, although such cross-validation discourages analysts from choosing too complicated models (too many predictors), the “optimal” number of predictors that minimizes the cross-validation error depends on the particular data at hand. If we draw another set of data from the same population, the “optimal” number may differ. Thus, selecting an optimal number of predictors itself is also overfitting but at a smaller order compared to that from a classifier with too many predictors.
18.4.4.2 **Boosting**

Boosting is an ingenious method developed in computer science during the 1990s\(^9-11\) for combining multiple “weak” classifiers into a powerful committee. There are variations of boosting algorithms. For a comprehensive discussion, see Hastie, et al.\(^7\) We will describe two popular boosting algorithms here: Discrete AdaBoost and Real AdaBoost. The applications of boosting for SELDI data were described in Qu\(^12\) for Discrete AdaBoost and Yasui\(^13\) for Real AdaBoost.

18.4.4.2.1 **Discrete AdaBoost**

Discrete AdaBoost uses a classifier \( f_m(x) \) that has value 1 (to indicate disease) or \(-1\) (to indicate nondisease) for a given protein peak indexed by \( m \). Suppose we use a simple “stump” classifier that classifies a sample as disease if the peak intensity at the \( m \)th protein peak above (or below) a certain level. This can be done by simply dividing the peak intensity into a number, say 50, of equally spaced segments and find the cutoff point that best differentiates disease and nondisease samples. If we are not comfortable about the quantitative measure of intensity, we can simply use \( X = 1 \) (peak) or \( X = 0 \) (no peak). We exam all peaks this way and choose the peak with a particular cutoff that gives the smallest misclassification error. This forms our first classifier \( f_m = 1 \). In this first classifier, we use an equal weight (weight \( = 1/N \)) for all \( N \) observations.

This classifier is usually a weak classifier. Now we update the weights by assigning larger weights to the observations that were misclassified by this classifier. For observation \( Y_i \) (\( Y_i = 1 \) for disease, \( Y_i = -1 \) for nondisease), the new weight is \( w_i = \text{previous weight} \times \exp\{c_m \times I[y_i f_m(x) = -1] \} \), where \( c_m = \log[(1 - \text{err}_m)/\text{err}_m] \), where \( \text{err}_m \) is the proportion of all \( N \) samples that are misclassified by the current classifier and \( I(.) \) is an indicator function and equals 1 if the statement in parenthesis is correct, 0 otherwise. Note that if the prediction is correct, then \( y_i f_m(x) = 1 \) and \( \exp\{c_m \times I[y_i f_m(x) = -1] \} = 1 \). The new weight is then equal to the old weight. If the prediction is incorrect, then \( \exp\{c_m \times I[y_i f_m(x) = -1] \} = \exp(c_m) \) is larger than 1 as long as \( c_m \) is positive; this always holds with a high probability if the weak classifier is better than flipping a coin. This is the first important feature of boosting: assign larger weights to those difficult to classify. Logistic regression uses the same weight for each observation.

We apply our favorite classifier algorithm again, the same way as described above, except now to the weighted data. We repeat this process \( M \) times and the final classifier is

\[
f(x) = \text{sign}\left[ \sum_{m=1}^{M} \text{coef}_m(x) \right]
\]

a weighted sum of all \( M \) classifiers: the classifier is \( f(x) = 1 \) if the sign of the sum is positive and \(-1\) otherwise.

We can think of this final classifier \( f(x) \) as a committee formed by \( M \) members (classifiers). Each member gives a vote 1 or \(-1\). A member that has better knowledge
(a smaller $\text{err}_m$ and therefore a larger $c_m$) carries larger weight in voting. This is the second important feature of boosting: a committee voting. This is analogous to $b$’s (regression coefficients) in logistic regression.

The third feature of boosting is that the same protein peak can be repeatedly selected as a committee member, likely with different values of $c_m$, while in logistic regression each protein peak can appear in the final predictor at most once.

How shall we choose the number of iteration $M$? We can use the cross-validation described above to determine $M$ that gives the smallest cross-validation error. If we let $M$ too big, we will eventually overfit the data.

18.4.4.2.2 Real AdaBoost
Real AdaBoost has the following differences from the Discrete AdaBoost:

1. Instead of using a discrete classifier $f_m(x)$, Real AdaBoost uses a continuous classifier that produces a class-probability estimate $p_m$, the predicted probability of disease based on $m$th protein peak. A natural choice for constructing class-probability estimates is via logistic regression with a single covariate.
2. Calculates a quantitative classifier $f_m(x) = 0.5 \log[p_m(x)/(1 - p_m(x))]$
3. Updates the weight by: new weight = previous weight $\times \exp[-(y_if_m(x))]$, $i = 1, 2, \ldots N$, and renormalizes so that the sum of weights over $N$ samples equals to 1. The initial weight is $1/N$.
4. Repeat this process $M$ times and the final classifier is:

$$f(x) = \text{sign} \left[ \sum_{m=1}^{M} f_m(x) \right]$$

Note that in Real AdaBoost, $f_m(x)$ will be positive if $p_m(x) > 1/2$ and will increase as $p_m(x)$ increases. Therefore, in each iteration it assigns weights to each observation not only according to whether it is correctly classified but also the confidence of correct specification or the extent of misclassification. In each iteration, Discrete AdaBoost assigns one weight for all correctly classified and one weight for all incorrectly classified. Therefore, Real AdaBoost uses “confidence-rated” prediction in weights and we expect it will “learn faster” and have better predicting power. See Friedman et al.\textsuperscript{14} for discussion and a simulation experiment on this point.

18.4.4.2.3 Why Boosting?
Among many data mining algorithms, why do we prefer boosting? The key is the observation that it appears to be resistant to overfitting, yet it still has good predicting power. This consideration is important because any high-dimensional data provides ample opportunities to overfit and therefore it is the main trap we need to avoid. It is still not completely clear why boosting is resistant to overfitting. One possibility is due to its “slowness” in learning. It adds one predictor at a time and when a new predictor is added, boosting does not try to optimize all predictors in the model.
simultaneously. This reduces the variance of the model. Another explanation is that the impact of each weak classifier $f_m(x)$ reduces when iteration proceeds as the correctly classified observations with high confidence have the sum of $f_m(x)$ far away from zero and are less likely to change their signs by the new added weak classifier. Therefore, the bulk of the data are not affected much. That increases the stability of the model.

Another consideration is the ease of interpretability. Because each weak classifier is a very simple model, their weighted voting is easy to interpret. This is the advantage of boosting over other methods like neural network.

The logistic regression with cross-validation may be preferred in some instances over the boosting if their performance difference is not appreciable and the former uses considerably fewer number of peaks in the classifier. This is because further scientific investigations following the SELDI/MALDI-based biomarker exploration would study specific protein peaks in the classifier as potential biomarkers.

18.4.5 Assessment of Prediction Error

Assessment of prediction error serves two main purposes: selecting a classifier with the minimum future prediction error and estimating future prediction error of the classifier.

The number of iterations in boosting and the number of predictors to be included in logistic regression are called “tuning parameters.” They specify the complexity of the classifier. We can select the value of these tuning parameters such that the classifier has the smallest prediction error in cross-validation. The cross-validation error mimics the future prediction error. Therefore, we can use the estimated future prediction error to select our classifiers.

As described in Section 18.4.4.1, when the cross-validation procedure is used more than once, as we have to in classifier selection, the minimum cross-validation error among several classifiers will generally underestimate the future prediction error of the classifier in an independent new data set. Therefore, we want to have an assessment of the future prediction error after we finalized our classifier. This can be achieved by using bootstrap.

Bootstrap is a powerful yet general statistical strategy for making inference on quantities that is difficult otherwise. For good explanations of the bootstrap method, see Efron and Tibshirani.15 In our situation, we want to estimate a future prediction error for the final model. It goes like this:

We randomly draw, with replacement, $N$ observations from original $N$ observations to form a bootstrap sample. We repeat this $B$ times. We suggest $B$ to be at least 100.

For each bootstrap sample, we repeat our whole model selection process. That is where we start using disease status information, i.e., the reduction of dimensionality in Section 18.4.3. That means that, for each bootstrap sample, we will select candidate peaks, use cross-validation to determine the tuning parameters, and construct a classifier using a favorite algorithm. By the end, we will have $B$ bootstrap classifiers, one for each of $B$ bootstrap samples.

Use each of $B$ bootstrap classifiers to classify the observations in the original sample that are not selected in this particular bootstrap sample and compute the
prediction error (misclassification error, sensitivity, specificity, etc.). The average of B prediction errors is our estimate of future prediction error. We call this “validation prediction error.”

It is important to note that we have three types of prediction errors: training-set prediction error (the observed prediction error for the final model on the original training data), validation prediction error, and test-set prediction error, with increasing generalizability. The validation prediction error is the most you can extract from a training data set on future prediction error. It cannot completely replace the need for a test data set, but it could be a good compromise when it is not feasible to have a large enough test data set. At a minimum, analysts need to use cross-validation to select a classifier. Letting a model become too complex, either explicitly or implicitly, to achieve an apparent lowest training-set prediction error without cross-validation easily leads to overfitting.

18.5 CONCLUSION

Proteomics offers great hope as well as great challenges in biomedical research. There are, and will be, many statistical and bioinformatics algorithms to analyze such data. However, the following general principles will apply:

1. Need to understand sources of bias and variation of the data
2. Design experiments that eliminate or reduce bias and/or variance, and meet study objective
3. Make the analysis strategy consistent with the experimental design and the study objectives and also resistant to overfitting; rigorously assess future prediction errors

REFERENCES
