Predictive Modeling

Department of Mathematical Sciences
Qiuying Sha, Professor

Michigan Tech

Chapter 13. Nonlinear Classification Models

The **previous chapter** described models that were intrinsically **linear**

- The structure of the model would produce **linear class boundaries** unless nonlinear functions of the predictors were manually specified.

This chapter deals with some intrinsically **nonlinear models**. We will cover the following models:

- Nonlinear Discriminant Analysis
- Neural Networks
- Flexible Discriminant Analysis
- Support Vector Machines
- *K*-Nearest Neighbors
- Naive Bayes

13.1 Nonlinear Discriminant Analysis

In LDA, we assume that **the variance is the same** for all classes, resulting the linear boundaries for classification.

In this section, we **modify this assumption** to handle data that are best separated by nonlinear structures.

These methods include:

- Quadratic Discriminant Analysis (QDA),
- Regularized Discriminant Analysis (RDA), and
- Mixture Discriminant Analysis (MDA).

Quadratic and Regularized Discriminant Analysis

Quadratic Discriminant Analysis (QDA)

LDA assumes that the predictors in each class shared a common covariance structure, resulting the class boundaries were linear functions of the predictors.

In QDA, this assumption is relaxed so that a class-specific covariance structure can be accommodated.

- The decision boundaries now become quadratically curvilinear in the predictor space.
- Improve model performance for many problems.

This generalization is that the data requirements become more stringent.

- Since class-specific covariance matrices are utilized, the inverse of the matrices must exist.
  - This means that the number of predictors must be less than the number of cases within each class.
  - Also, the predictors within each class must not have pathological levels of collinearity.
  - Additionally, if the majority of the predictors in the data are indicators for discrete categories, QDA will only to able to model these as linear functions, thus limiting the effectiveness of the model.

LDA and QDA each minimize the total probability of misclassification assuming that the data can truly be separated by hyperplanes or quadratic surfaces.

Reality may be, however, that the data are best separated by structures somewhere between linear and quadratic class boundaries.

Regularized Discriminant Analysis (RDA)

RDA, proposed by Friedman (1989), is one way to bridge the separating surfaces between LDA and QDA.

Friedman proposed the following covariance matrix:

$$\tilde{\Sigma}_l(\lambda) = \lambda\Sigma_l + (1-\lambda)\Sigma$$

where Σ_l is the covariance matrix of the l^{th} class and Σ is the pooled covariance matrix across all classes.

The tuning parameter, λ, enables the method to flex the covariance matrix between LDA (when $\lambda = 0$) and QDA (when $\lambda = 1$).

If a model is tuned over λ, a data-driven approach can be used to choose between linear or quadratic boundaries as well as boundaries that fall between the two.

RDA makes another generalization of the data: the pooled covariance matrix can be allowed to morph from its observed value to one where the predictors are assumed to be independent:

$$\Sigma(\gamma) = \gamma\Sigma + (1-\gamma)\sigma^2 I$$

where σ^2 is the common variance of all predictors and I is the identity matrix.

Tuning an RDA model over λ and γ enables the training set data to decide the most appropriate assumptions for the model.

Mixture Discriminant Analysis (MDA)

MDA was developed by Hastie and Tibshirani (1996) as an extension of LDA.

- LDA assumes a distribution of the predictor data such that the class-specific means are different.

- MDA generalizes LDA in a different manner;

  - It allows each class to be represented by *multiple* multivariate normal distributions. These distributions can have different means but, like LDA, the covariance structures are assumed to be the same.

Figure 13.1 presents this idea with a single predictor.

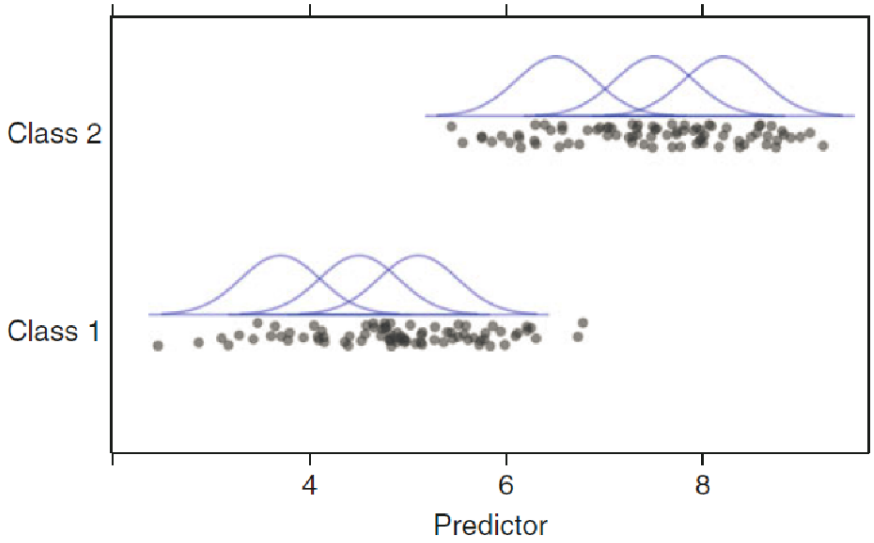


Fig. 13.1: For a single predictor, three distinct subclasses are determined within each class using mixture discriminant analysis

- Here each class is represented by three normal distributions with different means and common variances.

- The modeler would specify how many different distributions should be used and the MDA model would determine their optimal locations in the predictor space.

How are the distributions aggregated so that a class prediction can be calculated?

- MDA modifies $Pr[X|Y = C_l]$.

- The class-specific distributions are combined into a single multivariate normal distribution by creating a per-class mixture.

- Suppose D_{lk} is the discriminant function for the k^{th} subclass in the l^{th} class, the overall discriminant function for the l^{th} class would be proportional to

$$D_\ell(x) \propto \sum_{k=1}^{L_\ell} \phi_{\ell k} D_{\ell k}(x),$$

where L_l is the number of distributions being used for the i^{th} class and the ϕ_{lk} are the mixing proportions that are estimated during training.

- This overall discriminant function can then produce class probabilities and predictions.

- For this model, the number of distributions per class is the tuning parameter for the model (they need not be equal per class).

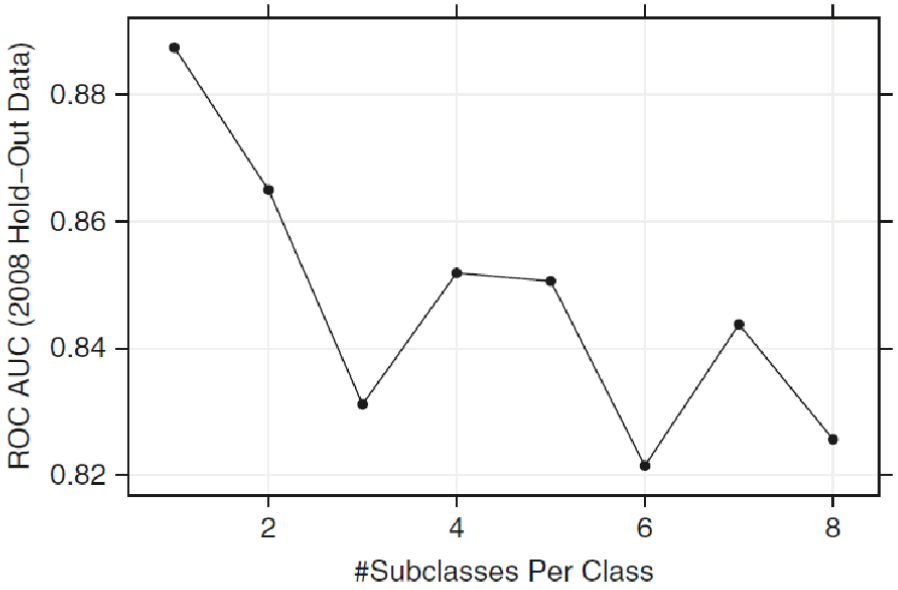


Fig. 13.2: The tuning parameter profile for the MDA model for the grants data. The optimal number of subclasses is 1, which is identical to performing LDA

13.2 Neural Networks

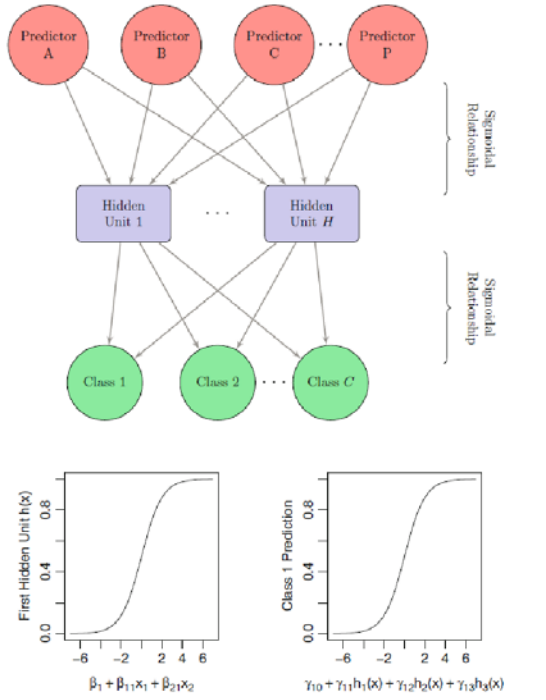


Fig. 13.3: A diagram of a neural network for classification with a single hidden layer. The hidden units are linear combinations of the predictors that have been transformed by a sigmoidal function. The output is also modeled by a sigmoidal function

- Same as partial least squares discriminant analysis, for NNet, the *C* classes can be encoded into *C* binary columns of dummy variables and then used as the outcomes for the model.

- Although the previous discussion on NNet for regression used a single response, the model can easily handle multiple outputs for both regression and classification.

Michigan Tech

- The *softmax* transformation is used to ensure that the outputs of the neural network are "probability-like" (they do add up to one),

$$f_{i\ell}^{*}(x) = \frac{e^{f_{i\ell}(x)}}{\sum_{l} e^{f_{il}(x)}},$$

where $f_{il}(x)$ is the model prediction of the l^{th} class and the i^{th} sample.

- The class with the largest predicted value would be used to classify the sample.
- What should the neural network optimize to find appropriate parameter estimates?
  - Minimize the sum of the squared errors

$$\sum_{l=1}^{C}\sum_{i=1}^{n}(y_{il} - f_{il}^{*}(x))^2$$

where y_{il} is the 0/1 indicator for class l

- Maximize the log likelihood of the Bernoulli distribution

$$\sum_{l=1}^{C}\sum_{i=1}^{n} y_{il} \ln f_{il}^{*}(x)$$

It is also called *entropy* or *cross-entropy*.

The likelihood has more theoretical validity than the squared error approach,

Like their regression counterparts, neural networks for classification have a significant potential for over-fitting.

When optimizing the sums of squares error or entropy, weight decay reduces the size of the parameter estimates.

- Minimize

$$\sum_{l=1}^{C}\sum_{i=1}^{n}(y_{il}-f_{il}^{*}(x))^2+\lambda\sum_{k=1}^{H}\sum_{j=0}^{P}\beta_{jk}^2+\lambda\sum_{l=1}^{C}\sum_{k=0}^{H}\gamma_{lk}^2$$

- Maximize

$$\sum_{l=1}^{C}\sum_{i=1}^{n}y_{il}\ln f_{il}^{*}(x)-\lambda\sum_{k=1}^{H}\sum_{j=0}^{P}\beta_{jk}^2-\lambda\sum_{l=1}^{C}\sum_{k=0}^{H}\gamma_{lk}^2$$

Also, as previously discussed, model averaging helps reduce over-fitting.

- The class probability estimates $f_{il}^{*}(x)$ would be averaged across networks and these average values would be used to classify samples.

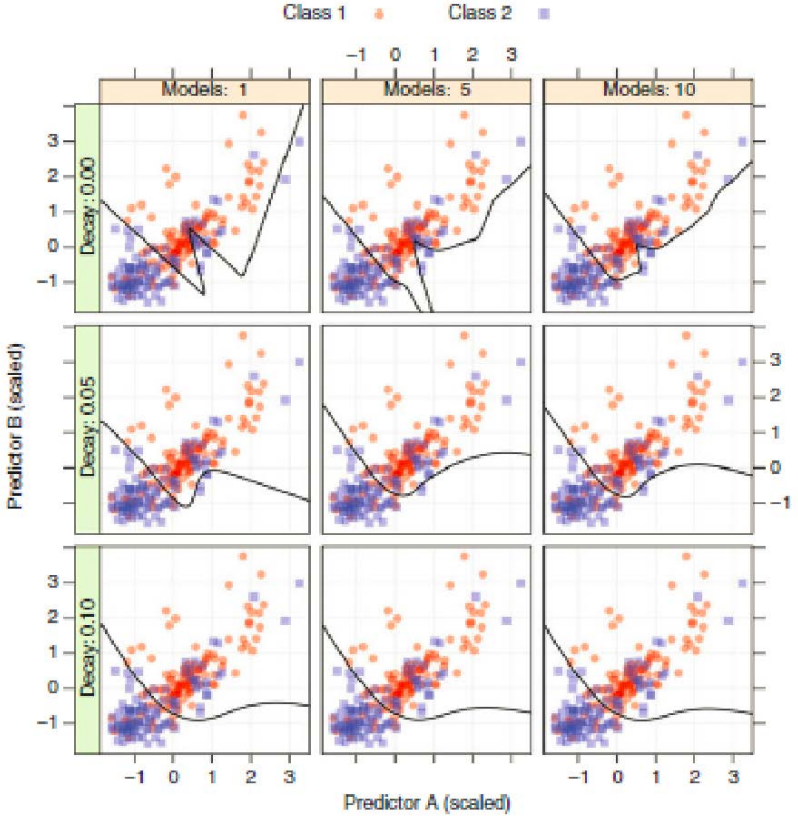


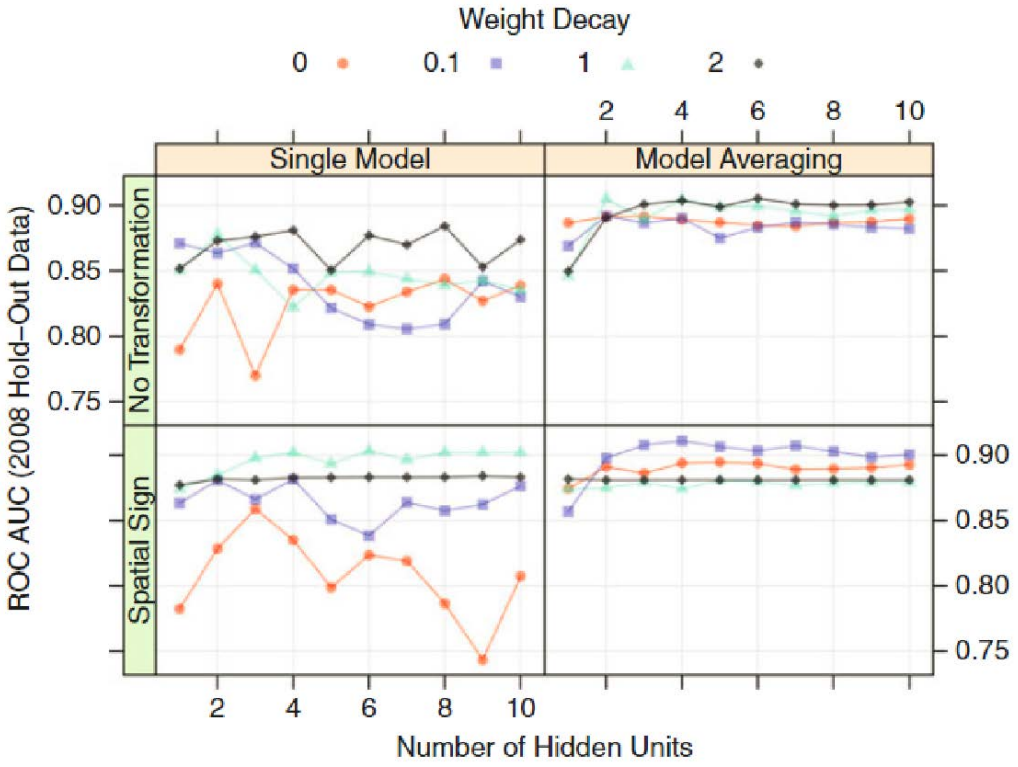
Fig. 13.4: Classification boundaries for neural networks with varying levels of smoothing and regularization. As weight decay and number of models increase, the boundaries become smoother

- This figure shows examples of models fit with different amounts of weight decay and model averaging.
- For these data, a single model with weight decay is probably the best choice since it is computationally least expensive.

Collinearity and non-informative predictors will have a comparable impact on model performance.

Several types of neural networks were fit to the grant data.

- First, single network models (i.e., no model averaging) were fit using entropy to estimate the model coefficients.
  - The models were tuned over the number of units in the hidden layer (ranging from 1 to 10), as well as the amount of weight decay (λ = 0, 0.1, 1, 2).
  - The best model used eight hidden units with λ = 2 and AUC = 0.884.
- Average of 10 networks
  - The best model had six hidden units with λ = 2 and AUC = 0.884.
- Various transformations of the data were evaluated, one in particular, the spatial sign transformation is used.



- When combined with a single network model, AUC=0.903.
- When model averaging was used, AUC=0.911.

Michigan Tech

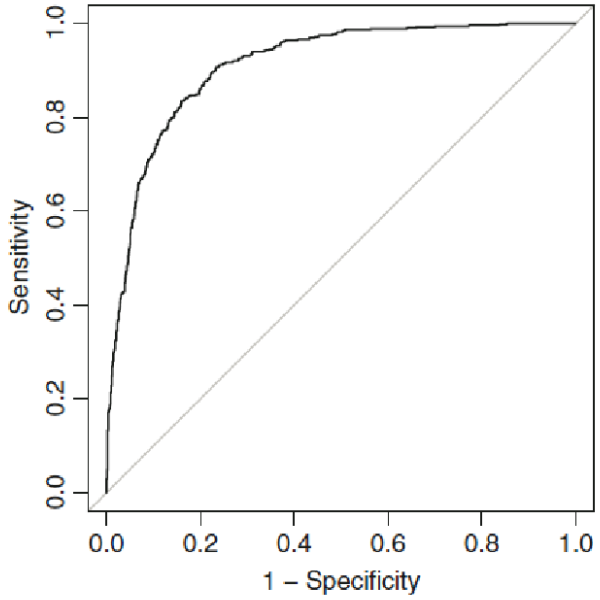


Fig. 13.5: *Top*: The models for grant success were tuned under four different conditions: with and without a transformation on the predictors and with and without model averaging. *Bottom*: The ROC curve for the 2008 holdout set when a model averaged network is used with the spatial sign transformation (area under the curve: 0.911)

13.3 Flexible Discriminant Analysis

Hastie et al. (1994) describe a process where, for C classes,

- a set of C linear regression models can be fit to binary class indicators and
- then the regression coefficients from these models can be post-processed to derive the discriminant coefficients (see Algorithm 13.1).

1 Create a new response matrix of binary dummy variable columns for each of the C classes

2 Create a multivariate regression model using any method that generates slopes and intercepts for predictors or functions of the predictors (e.g. linear regression, MARS, etc)

3 Post-process the model parameters using the optimal scoring technique

4 Use the adjusted regression coefficients as discriminant values

Algorithm 13.1: The flexible discriminant analysis algorithm for generalizing LDA model (Hastie et al. 1994)

This allows the idea of linear discriminant analysis to be extended in a number of ways.

- First, many of the models in Chaps. 6 and 7, such as the lasso, ridge regression, or MARS, can be extended to create discriminant variables.
  - For example, MARS can be used to create a set of hinge functions that result in discriminant functions that are nonlinear combinations of the original predictors.
  - As another example, the lasso can create discriminant functions with feature selection.

This conceptual framework is referred to as *flexible discriminant analysis* (FDA).

We illustrate the nonlinear nature of the flexible discriminant algorithm using MARS with the example data in Fig. 4.1 (p. 63).

- MARS has two tuning parameters: the number of retained terms and the degree of predictors involved in the hinge functions.

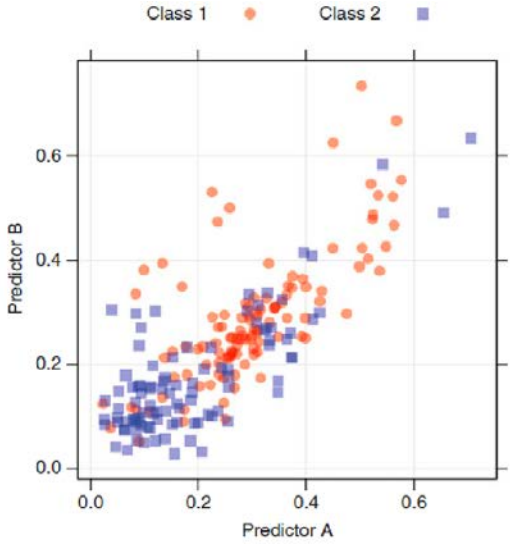
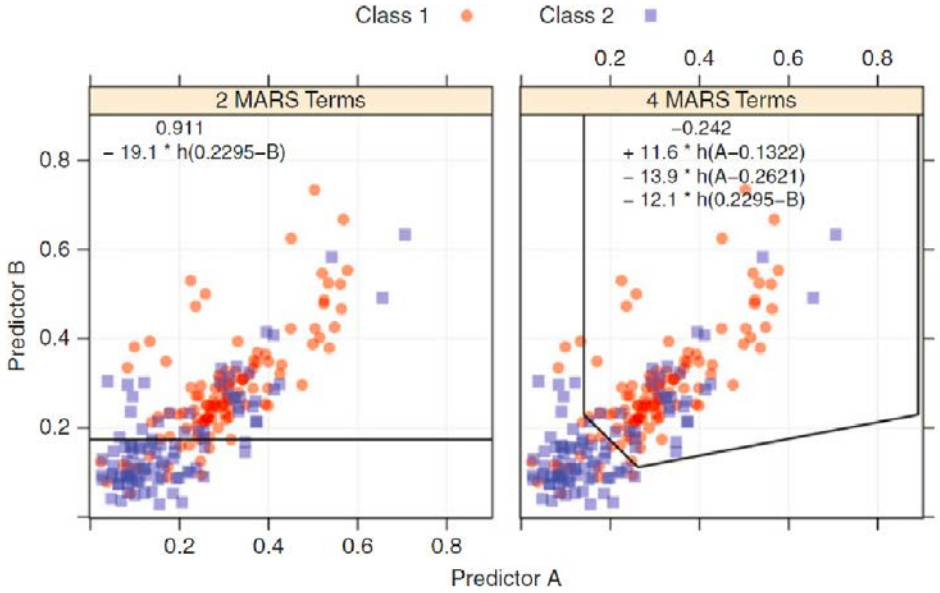


Fig. 4.1: An example of classification data that is used throughout the chapter

- If we use an additive model (i.e., a first-degree model), constrain the maximum number of retained terms to 2 and have a binary response of class membership, then discriminant function is

$$D(A, B) = 0.911 - 19.1 \times h(0.2295 - B)$$

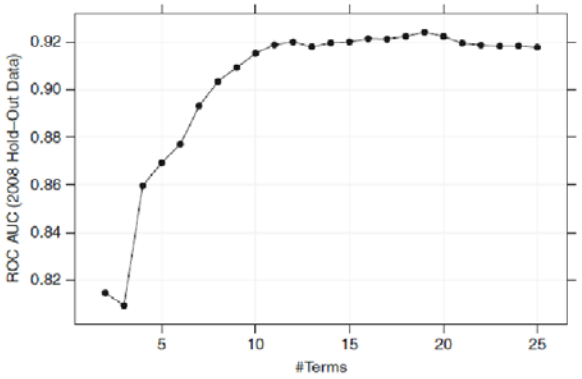


- If the maximum number of retained terms is relaxed to 4, then the discriminant equation is estimated to

$$\begin{aligned} D(A, B) = & -0.242 \\ & + 11.6 \times h(A - 0.1322) \\ & - 13.9 \times h(A - 0.2621) \\ & - 12.1 \times h(0.2295 - B). \end{aligned}$$

An FDA model was tuned and trained for the grant application model.

- First-degree MARS hinge functions were evaluated where the number of retained terms ranged from 2 to 25.
- Although the FDA model contained 19 terms, 14 unique predictors were used (of a possible 1,070).
- The discriminant function shown below can be additionally transformed to produce class probability estimates.



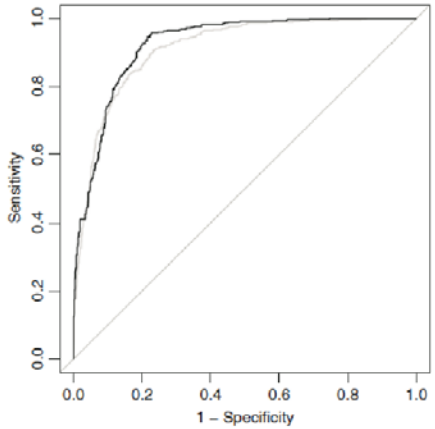


Fig. 13.7: *Top*: The parameter tuning profile for the FDA model. *Bottom*: The FDA ROC curve (area under the curve: 0.924) is shown in relation to the curve for the previous neural network model (in *grey*)

$$
\begin{aligned}
D(x) = 0.85 \\
&- 0.53 \times h(1 - \text{number of chief investigators}) \\
&+ 0.11 \times h(\text{number of successful grants by chief investigators} - 1) \\
&- 1.1 \times h(1 - \text{number of successful grants by chief investigators}) \\
&- 0.23 \times h(\text{number of unsuccessful grants by chief investigators} - 1) \\
&+ 1.4 \times h(1 - \text{number of unsuccessful grants by chief investigators}) \\
&+ 0.18 \times h(\text{number of unsuccessful grants by chief investigators} - 4) \\
&- 0.035 \times h(8 - \text{number of A journal papers by all investigators}) \\
&- 0.79 \times \text{sponsor code 24D} \\
&- 1 \times \text{sponsor code 59C} \\
&- 0.98 \times \text{sponsor code 62B} \\
&- 1.4 \times \text{sponsor code 6B} \\
&+ 1.2 \times \text{unknown sponsor} \\
&- 0.34 \times \text{contract value band B} \\
&- 1.5 \times \text{unknown contract value band} \\
&- 0.34 \times \text{grant category code 30B} \\
&+ 0.3 \times \text{submission day of Saturday} \\
&+ 0.022 \times h(54 - \text{numeric day of the year}) \\
&+ 0.076 \times h(\text{numeric day of the year} - 338).
\end{aligned}
$$

- Since many of the predictors in the FDA model are on different scales, it is difficult to use the discriminant function to uncover which variables have the most impact on the outcome.

- The same method of measuring variable importance described in Sect. 7.2 can be employed here.

13.4 Support Vector Machines

Consider the problem shown in the left panel of Fig. 13.9 where two variables are used to predict two classes of samples that are completely separable.

- There are a multitude (in fact an infinite) number of linear boundaries that perfectly classify these data.
- How would we choose an appropriate class boundary?

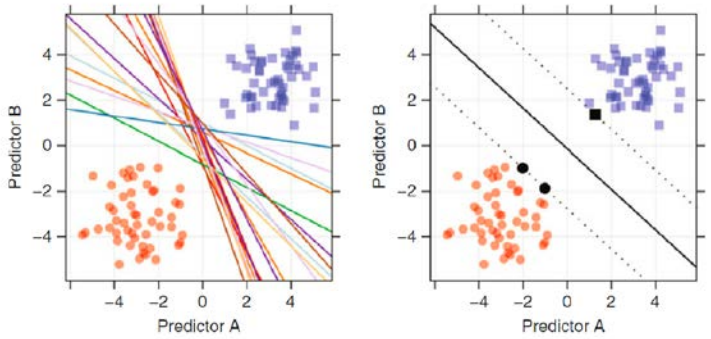


Fig. 13.9: *Left*: A data set with completely separable classes. An infinite number of linear class boundaries would produce zero errors. *Right*: The class boundary associated with the linear maximum margin classifier. The solid black points indicate the support vectors

- Many performance measures, such as accuracy, are insufficient since all the curves would be equivalent.
- What would a more appropriate metric be for judging the efficacy of a model?

- An alternate metric called the *margin*.
- Loosely speaking, the margin is the distance between the classification boundary and the closest training set point.
- In this example the three data points are equally closest to the classification boundary and are highlighted with solid black symbols.

Suppose we have a two-class problem and we code the class #1 samples with a value of 1 and the class #2 samples with −1.

Also, let the vectors $\mathbf{x}_i$ contain the predictor data for a training set sample.

The maximum margin classifier creates a decision value $D(\mathbf{x})$ that classifies samples such that if $D(\mathbf{x}) > 0$ we would predict a sample to be class #1, otherwise class #2.

For an unknown sample **u**, the decision equation can be written in a similar form as a linear discriminant function that is parameterized in terms of an intercept and slopes as

$$D(\mathbf{u}) = \beta_0 + \boldsymbol{\beta}'\mathbf{u}$$
$$= \beta_0 + \sum_{j=1}^{P} \beta_j u_j.$$

$$D(\mathbf{u}) = \beta_0 + \sum_{j=1}^{P} \beta_j u_j$$
$$= \beta_0 + \sum_{i=1}^{n} y_i \alpha_i \mathbf{x}_i'\mathbf{u}$$

In the completely separable case, the *α* parameters are exactly zero for all samples that are not on the margin.

What happens when the classes are not completely separable?

- Cortes and Vapnik (1995) develop extensions to the early maximum margin classifier to accommodate this situation.
- Their formulation puts a cost on the sum of the training set points that are on the boundary or on the wrong side of the boundary.
- When determining the estimates of the α values, the margin is penalized when data points are on the wrong side of the class boundary or inside the margin.
- The cost value would be a tuning parameter for the model and is the primary mechanism to control the complexity of the boundary.

Conversely, when the cost is relatively high (say a value of 16), the model can over-fit the data.

Using resampling to find appropriate estimates of these parameters tends to find a reasonable balance between under- and over-fitting.

The predictor equation is a function of only a subset of the training set points and these are referred to as the *support vectors.*

Since the prediction equation is *supported* solely by these data points, the maximum margin classifier is the usually called the *support vector machine*.

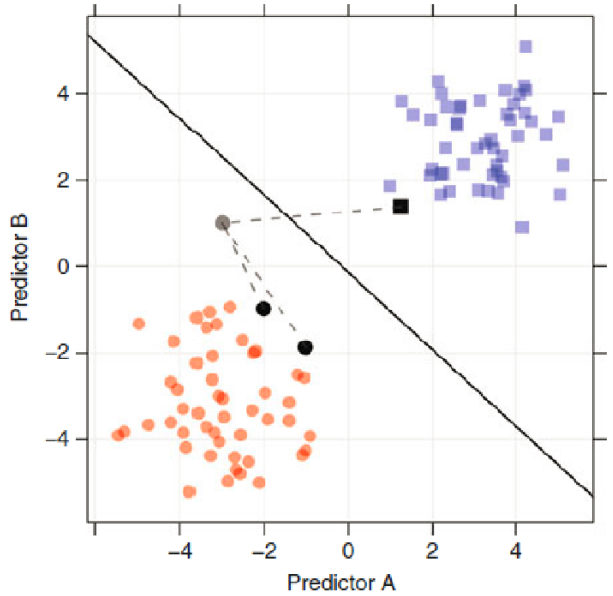


Fig. 13.10: Prediction of a new sample using a support vector machine. The final value of the decision equation is $D(u) = 0.583$. The grey lines indicate the distance of the new sample to the support vectors

- Thus far, we have considered linear classification boundaries for these models.
- Since the predictors enter into this equation in a linear manner, the decision boundary is correspondingly linear.
-

	True class	Dot product	y_i	α_i	Product
SV 1	Class 2	−2.4	−1	1.00	2.40
SV 2	Class 1	5.1	1	0.34	1.72
SV 3	Class 1	1.2	1	0.66	0.79

- Boser et al. (1992) extended the linear nature of the model to nonlinear classification boundaries by substituting the kernel function instead of the simple linear cross product:

$$D(\mathbf{u}) = \beta_0 + \sum_{i=1}^{n} y_i \alpha_i \mathbf{x}_i' \mathbf{u}$$
$$= \beta_0 + \sum_{i=1}^{n} y_i \alpha_i K(\mathbf{x}_i, \mathbf{u}),$$

where $K(\cdot, \cdot)$ is a *kernel function* of the two vectors.

  - For the linear case, the kernel function is the same inner product $\mathrm{x}'u$
  - Other nonlinear transformations can be applied, including:

$$\text{polynomial} = (scale\,(\mathbf{x}'\mathbf{u}) + 1)^{degree}$$
$$\text{radial basis function} = \exp(-\sigma\|\mathbf{x} - \mathbf{u}\|^2)$$
$$\text{hyperbolic tangent} = \tanh(scale\,(\mathbf{x}'\mathbf{u}) + 1).$$

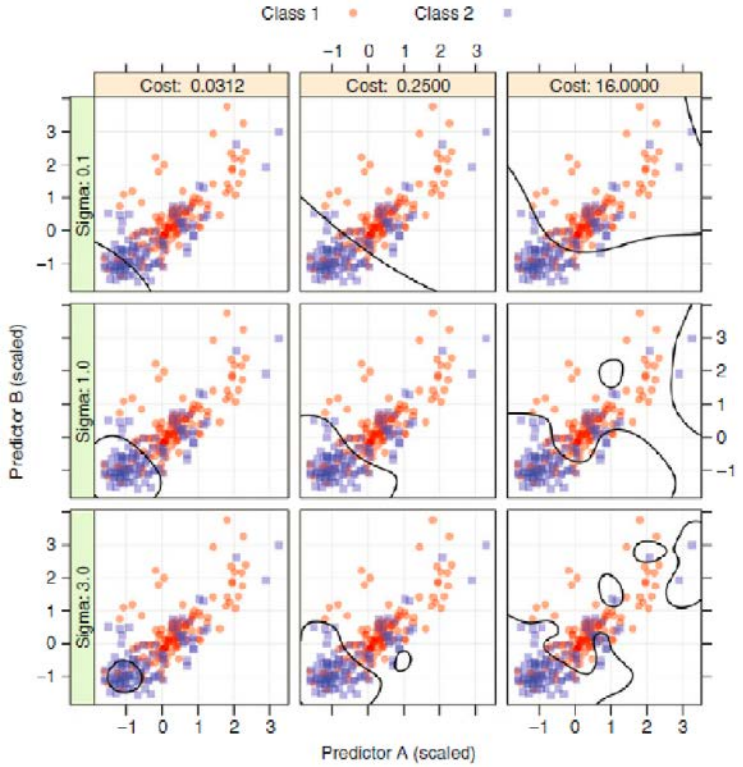


Fig. 13.11: Classification boundaries for nine radial basis function support vector machine models varied over the cost parameter and the kernel parameter (σ)

Figure 13.11 shows examples of the classification boundaries produced by several models using different combinations of the cost and tuning parameter values.

- When the cost value is low, the models clearly underfit the data.
- When the cost is relatively high (say a value of 16), the model can over-fit the data, especially if the kernel parameter has a large value.

Using resampling to find appropriate estimates of these parameters tends to find a reasonable balance between under- and over-fitting.

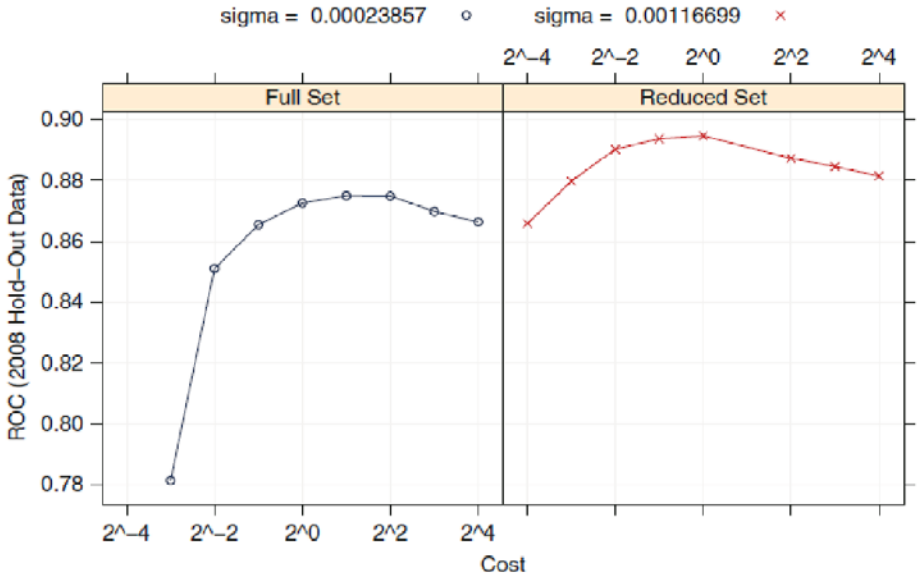


Fig. 13.12: Tuning parameter profile of the radial basis function SVM model for the grant data

We evaluated the radial basis function kernel to the grant data.

Also, both the full and reduced predictor sets were evaluated.

13.5 *K*-Nearest Neighbors

We first used the *K*-nearest neighbors (*K*NNs) model to perform imputation.

Then we used KNN in chapter 7 for regression.

Many of the ideas from *K*NN for regression directly apply to classification.

The classification methods discussed thus far search for linear or nonlinear boundaries that optimally separate the data.

These boundaries are then used to predict the classification of new samples.

*K*NN takes a different approach by using a sample's geographic neighborhood to predict the sample's classification.

- *K*NN for classification predicts a new sample using the *K*-closest samples from the training set.
- "Closeness" is determined by a distance metric, like Euclidean and Minkowski.
- The original measurement scales of the predictors affect the resulting distance calculations.
  - if predictors are on widely different scales, the distance value between samples will be biased towards predictors with larger scales. To allow each predictor to contribute equally to the distance calculation, we recommend centering and scaling all predictors prior to performing *K*NN.

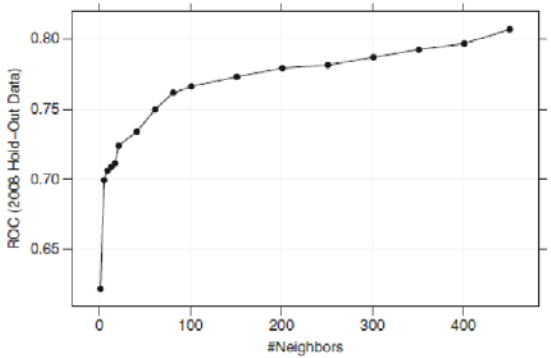
- Class probability estimates for the new sample are calculated as the proportion of training set neighbors in each class.
- The new sample's predicted class is the class with the highest probability estimate;
- If two or more classes are tied for the highest estimate, then the tie is broken at random or by looking ahead to the $K + 1$ closest neighbor.

K is the tuning parameter.

- Too few neighbors leads to highly localized fitting (i.e., over-fitting).
- Too many neighbors leads to boundaries that may not locate necessary separating structure in the data (under-fitting).
- Must use cross-validation or resampling approach for determining the optimal value of K.

Application to grant data:

- K was tuned between 1 and 451.
- The initial jump from 1 to 5 in predictive performance indicates that local geographic information is highly informative for categorizing samples.
- The steady incremental increase in predictive performance implies that neighborhoods of informative information for categorizing samples are quite large.
- This pattern is somewhat unusual for *K*NN. In most data sets, we are unlikely to use this many neighbors in the prediction.
- The optimal area under the ROC curve was 0.81, which occurred at $K = 451$.
- Figure 13.14 compares the *K*NN ROC profile with those of SVM and FDA.



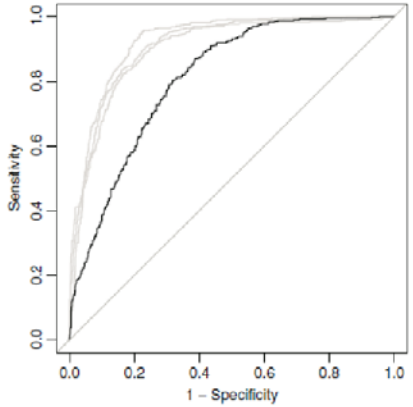


Fig. 13.14: *Top*: The parameter tuning profile for the *K*NN model. *Bottom*: The ROC curve for the test set data. The area under the curve was 0.81

13.6 Naive Bayes

Bayes' Rule answers the question "based on the predictors that we have observed, what is the probability that the outcome is class C_l?"

Mathematically,

- Let Y be the class variable and X represent the collection of predictor variables.
- Estimate $\Pr(Y = C_l \mid X)$

Bayes' Rule:

$$\Pr(Y = C_l \mid \mathrm{X}) = \frac{\Pr(Y = C_l)\Pr(X \mid Y = C_l)}{\Pr(X)}$$

- $\Pr(Y = C_l \mid X)$ is the *posterior probability* of the class.
- $\Pr(Y = C_l)$ is the *prior* probability of the outcome.

The naive Bayes model assumes that all of the predictors are independent of the others.

Under this assumption,

$$\Pr(\mathrm{X}) = \prod_{j=1}^{P} \Pr(X_j)$$

$$\Pr(\mathrm{X} \mid \mathrm{Y} = \mathrm{C}_l) = \prod_{j=1}^{P} \Pr(X_j \mid \mathrm{Y} = \mathrm{C}_l)$$

To estimate the individual probabilities,

- For continues predictors, we can assume normality, or using nonparametric kernel density estimators (Hardle et al. 2004) to estimate the probability densities.

- For categorical predictors, the probability distribution can be determined with the observed frequencies in the training set data.

  Notes: Laplacian correction can be used to avoid $\Pr(X_j|Y = C_l)$ by adding one observation in each group.

- To compute the overall conditional probability $\Pr(X \mid Y = C_l)$, each predictor is considered separately.

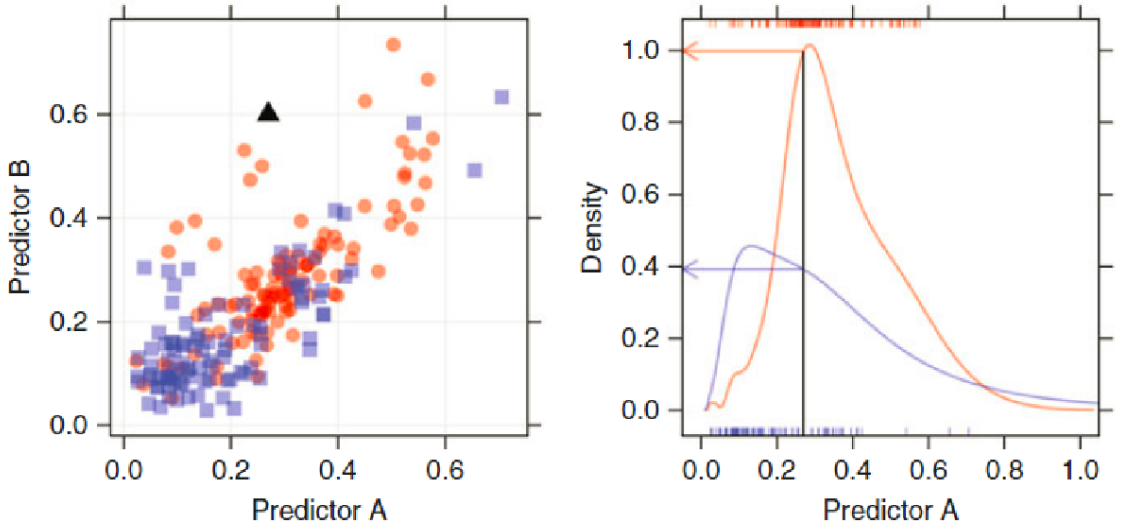


Fig. 13.15: *Left*: A plot of two class illustrative data where a new sample (the *solid triangle*) is being predicted. *Right*: Conditional density plots of predictor A created using a nonparametric density estimate. The value of predictor A for the new sample is shown by the *vertical black line*

Given such a severe and unrealistic assumption, why would one consider this model?

- First, the naive Bayes model can be computed quickly, even for large training sets.
- Secondly, despite such a strong assumption, the model performs competitively in many cases.

Class probabilities are created and the predicted class is the one associated with the largest class probability.

The meat of the model is the determination of the conditional and unconditional probabilities associated with the predictors.

For continuous predictors, one might choose simple distributional assumptions, such as normality.

The nonparametric densities (such as those shown in Fig. 13.16) can produce more flexible probability estimates.

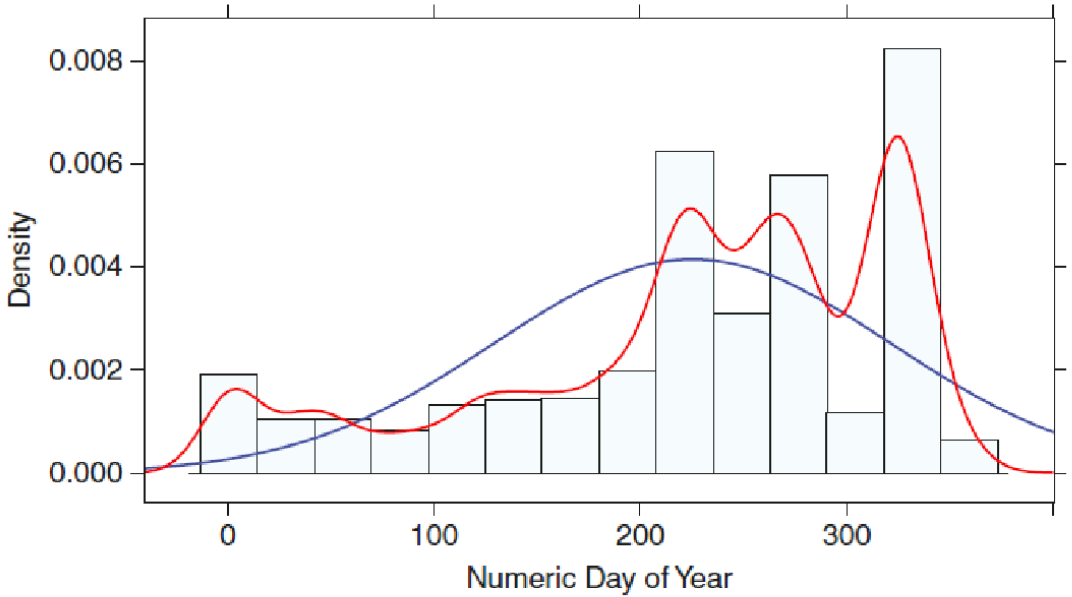


Fig. 13.16: Two approaches to estimating the density function $Pr[X]$ for the day of the year. The *blue line* is based on a normal distribution while the *red line* is generated using a nonparametric density estimator

For the grant data,

- Using a normal distribution for the continuous predictors, the area under the curve was estimated to be 0.78, a sensitivity of 58.8%, and a specificity of 79.6%.
- Using nonparametric estimation of the probability densities, the area under the ROC curve improves to 0.81, which corresponding increases in sensitivity (64.4%) and specificity (82.4%).
- The performance for this model is on par with *K*NNs, which is substantially below the results of the other models in this chapter.
- For this analysis, the reduced set of predictors was evaluated such that all predictors with less than 15 possible values were treated as discrete and their probabilities were calculated using their frequency distribution.