Recent developments in classification:  
Tree-based methods and support vector machines

A simple reference: An Introduction to Statistical Learning with Applications in R by G. James, D. Witten, T. Hastie, and R. Tibshirani (2013), Springer.

Tree-based methods for regression and classification: involve stratifying or segmenting the predictor space into a number of simple regions.

**Regression tree:** (for prediction of $y$ using $X$)

**Building a regression tree:**

1. Divide the predictor space, i.e. the set of possible values for predictors $X_1, X_2, \ldots, X_p$, into $J$ distinct and non-overlapping regions, say, $R_1, \ldots, R_J$.

2. For every observation that falls in the region $R_j$, we make the same prediction, which is simply the mean of the data in $R_j$.

In practice, for simplicity, we use high-dimensional rectangles to do the division. The goal is to find $R_1, \ldots, R_J$ that minimize the residual sum of squares, i.e.

$$
\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,
$$

where $\hat{y}_{R_j}$ is the mean response for the training observations in $R_j$. This is not feasible in practice so that one uses a top-down, greedy approach known as recursive binary splitting. It is greedy because at each step of the tree building process the best split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step.

First, we select the predictor $X_j$ and the cut-point $s$ such that the splitting of the predictor space into the regions $\{X|X_j < s\}$ and $\{X|X_j \geq s\}$ achieves minimum RSS. Defining

$$
R_1(j,s) = \{X|X_j < s\}, \quad R_2(j,s) = \{X|X_j \geq s\},
$$

we seek the value of $j$ and $s$ that minimize the equation

$$
\sum_{i: \mathbf{x}_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: \mathbf{x}_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2.
$$
This can be done easily provided that \( p \) is not too large.

Second, \textit{tree pruning}: Conceptually, we can consider every subtree of a large tree \( T_0 \) and use cross-validation to find a subtree that has the lowest test error. This is, however, hard to implement. One possible solution is to use the idea of \textit{cost complexity pruning}, which is also known as \textit{weakest link pruning}. Basically, we consider a sequence of subtrees indexed by a nonnegative running parameter \( \alpha \). For a given value of \( \alpha \), there corresponds a subtree \( T \subset T_0 \) such that

\[
|T| \sum_{m=1}^{[T]} \sum_{i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T|
\]

is as small as possible, where \(|T|\) denotes the number of terminal nodes of the tree \( T \). In practice, \( \alpha \) can be obtained via cross-validation.

\textbf{Classification tree}: Similar to regression tree except that it is used to predict a qualitative response. Instead of RSS, the classification error rate is used in the building process. We assign a data point to a category using the \textit{most commonly occurring class} of the region to which the data point belongs. Let \( p_{mk} \) be the proportion of the training data in the \( m \)th region that are from the \( k \)th class. Then,

\[
E = 1 - \max_k (\hat{p}_{mk}),
\]

where \( \hat{p}_{mk} \) is the proportion of the training observations in the \( m \)th region that are from the \( k \)th category. This error, however, does not work satisfactorily in practice. Instead, two alternative measures are used:

1. Gini index:

\[
G = \sum_{k=1}^{K} \hat{p}_{mk} (1 - \hat{p}_{mk})
\]

which is a measure of total variance across the \( K \) categories. Note that \( G \) assume a small value if all of the \( \hat{p}_{mk} \)'s are close to zero or one. Therefore, the Gini index is referred to as a measure of node purity, a small value indicates that a node contains predominantly observations from a single category.

2. Cross-entropy:

\[
D = -\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk}).
\]

Note that \( 0 \leq \hat{p}_{mk} \leq 1 \), it follows that \( 0 \leq -\hat{p}_{mk} \log(\hat{p}_{mk}) \). One can show that \( D \) will assume a value near zero if the \( \hat{p}_{mk} \)'s are all near zero or one. [Similar to Gini index as a measure of purity.]

\textbf{Improvements over tree-based methods}:
1. Bagging: (Bootstrap aggregation). Take bootstrap samples from the training data set, then take the average of the prediction of each bootstrap sample. This is called bagging. For classification, take the majority vote of the bootstrap samples.

2. Random forests: Use methods to de-correlate the trees in bagging. Specifically, in building a tree, each split only considers a random sample of $m$ explanatory variables as split candidate from the $p$ predictors. ($m \leq p$. When $m = p$, random forests reduce to bagging.)

3. Boosting: It does not use bootstrap samples. Instead trees are grown sequentially: each tree is grown using information from previously grown trees.

**Boosting for regression trees**

1. Set $\hat{f}(\mathbf{x}) = 0$ and $r_i = y_i$ for all $i$ in the training data set.

2. For $b = 1, \ldots, B$, repeat:
   
   (a) Fit a tree $\hat{f}^b$ with $d$ splits ($d + 1$ terminal nodes) to the training data $(\mathbf{X}, \mathbf{r})$
   
   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:
   
   $$\hat{f}(\mathbf{x}) \leftarrow \hat{f}(\mathbf{x}) + \lambda \hat{f}^b(\mathbf{x}).$$
   
   (c) Update the residuals,
   
   $$r_i \leftarrow r_i - \lambda \hat{f}^b(\mathbf{x}_i).$$

3. Output the boosted model
   
   $$\hat{f}(\mathbf{x}) = \sum_{b=1}^B \lambda \hat{f}^b(\mathbf{x}),$$

   where $\lambda$ is a shrinkage parameter. Typically, a small $\lambda$ is used, e.g. 0.01 or 0.001.

**Support vector machines**

1. Hyperplane: In a $p$-dimensional space $\mathbf{X} = (X_1, \ldots, X_p)'$, a hyperplane is simply a linear function, say
   
   $$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0.$$

2. Classification using a separating hyperplane: linear boundary

   For a given observation $\mathbf{x} = (x_1, \ldots, x_p)'$, classify $\mathbf{x}$ to category 1, denoted by $y = 1$ if $\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p > 0$. Classify $\mathbf{x}$ to category 2, denoted by $y = -1$ if $\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p < 0$. In other word, perform classification (for observations not on the hyperplane) based on which side of the hyperplane the data points lie.
The hyperplane is called the *separating hyperplane*. Note that for each data point \( \mathbf{x}_i = (x_{1i}, x_{2i}, \ldots, x_{pi})' \), the classification rule satisfies that
\[
y_i(\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi}) > 0.
\]

In application, if a separating hyperplane exists, then there are infinitely many separating hyperplanes provide the same classification.

3. The maximum margin classifier

Choose the hyperplane that has the maximum margin. [margin: the perpendicular distance to the hyperplane]

Justification: The distance of a data point to the separating hyperplane matters. We are in general more confident to classify those data points that are far away from the separating hyperplane.

Mathematically, the classification problem then becomes (where \( M \) denotes the margin)
\[
\text{maximize}_{\beta_0, \beta_1, \ldots, \beta_p} M
\]
\[
\text{subject to }
\sum_{j=1}^{p} \beta_j^2 = 1
\]
\[
y_i(\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi}) \geq M, \quad i = 1, \ldots, n.
\]

Discussions

(a) The maximum margin classifier is in effect determined by those data points that are close to the separating hyperplane. More specifically, those points are on the margin. Because they are vectors in \( \mathbb{R}^p \) so that they are referred to as the *supporting vectors*.

(b) Non-separable case: See the support vector classifiers next.

4. Support vector classifiers

Basic idea: Prefer a classifier with (a) greater robustness to individual data points, and (b) better classification for most of the training observations. Mathematically, we have
\[
\text{maximize}_{\beta_0, \beta_1, \ldots, \beta_p, \epsilon_1, \ldots, \epsilon_n} M,
\]
\[
\text{subject to }
\sum_{j=1}^{p} \beta_j^2 = 1
\]
\[
y_i(\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi}) \geq M(1 - \epsilon_i)
\]
\[
\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C
\]
where $C$ is a nonnegative tuning parameter. The $\epsilon_i$ are the slack variables that allow individual observations to be on the wrong side of the margin or the hyperplane. [The latter occurs when $\epsilon_i > 1$.

In practice, $C$ is commonly determined by the method of cross-validation. Observations that lie directly on the margin or on the wrong side of the margin for their class are known as support vectors. They affect the support vector classifier.

5. Support vector machines (SVM)

Consider the case of non-linear decision boundaries where a linear hyperplane fails to perform separation. What is next?

We use the same idea as in regression analysis; we enlarge the feature space via using quadratic, cubic, and even higher-order polynomial functions of the predictors. For instance, we may consider

$$\text{maximize}_{\beta_0, \beta_{11}, \ldots, \beta_{p1}, \beta_{12}, \ldots, \beta_{p2}, \epsilon_1, \ldots, \epsilon_n} M,$$

subject to

$$\sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1$$

$$y_i \left( \beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ji} + \sum_{j=1}^{p} \beta_{j2} x_{j2}^2 \right) \geq M (1 - \epsilon_i)$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C$$

In practice, one often uses other functions in places of polynomial functions to perform classification.

SVM: For any pair of observations $x_i = (x_{1i}, \ldots, x_{pi})'$ and $x_j = (x_{1j}, \ldots, x_{pj})'$, define the inner product as

$$\langle x_i, x_j \rangle = \sum_{k=1}^{p} x_{ki} x_{kj}.$$  

Then, one can use some kernel functions to perform classification. For instance, a polynomial kernel is defined as

$$K(x_i, x_j) = \left( 1 + \sum_{k=1}^{p} x_{ki} x_{kj} \right)^d,$$

where $d$ is a positive integer referring to as the degree of the polynomial kernel. Another popular choice of nonlinear kernel is the radial kernel defined by

$$K(x_i, x_j) = \exp \left( -\gamma \sum_{j=1}^{p} (x_{ki} - x_{kj})^2 \right),$$

5
where $\gamma$ is a positive constant, e.g. $\gamma = 10^{-3}$, $10^{-2}$, or $10^{-1}$. In practice, one only needs to work on the inner product of the support vectors.

**Discussion**: The case of multiple categories can be handled by two possibilities: (a) one-verse-one classification and (b) one-versus-all classification.

**The ROC curve**: A graphical display to show the performance of classification. It stands for *receiver operating characteristics* in the communication theory. We use it to quantity the overall performance of a classifier. Typically, it displays the true positive rate versus the false positive rate in the training sample. That is, plotting power versus Type-I error or plotting $1-$ Type-II error versus Type-I error.

**R packages**: e1071 with command `svm` and ROCR with command `rocplot`.
1 Some useful web sites

- Dr. Chun-Houh Chen, Institute of Statistical Science, Academia Sinica, Taiwan:
  http://gap.stat.sinica.edu.tw/
  1. Generalized Association Plots (GAP)
  2. Sliced Inverse Regression (SIR)
  3. Multidimensional Scaling (MDS)

We shall use the data set m-barra-9003.txt to demonstrate the GAP.

- Multidimensional Scaling web page: (Maintained by Y-H Taguchi)
  http://www.granular.com/MDS/
  Contains useful information (e.g. introduction, softwares) about MDS.

- The book *Smoothing of Multivariate Data* by Jussi Klemelä. The R package denpro is for the visualization of multivariate density estimates.

2 Biplot

A biplot is a graphical representation of the information in an \( n \times p \) data matrix. It intends to describe two kinds of information contained in the data matrix. As usual, the information in the rows pertains to samples or sampling units and that in the columns pertains to variables. Obviously, one likes to approximate the information as accurate as possible. This leads to using the sample principal components.

Let \( X \) be an \( n \times p \) data matrix with \( j \)th row \( x_j \). Based on the discussion of principal component analysis, the best two-dimensional approximation of the data is

\[
x_j \approx \bar{x} + \hat{y}_j \hat{e}_1 + \hat{y}_j \hat{e}_2,
\]

where \( \hat{e}_1 \) and \( \hat{e}_2 \) are the first two eigenvectors of the sample covariance matrix \( S = \frac{1}{n-1} \sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})' \). The two eigenvectors determine a plane and the coordinates of the \( j \)th row are the pair of values of the principal components \( (\hat{y}_j, \hat{y}_j) \).

**Example.** Data on universities, Table 12.9. The command of biplot in R is `biplot`, which may apply to a principal component object.
3 Dependence in high-dimensional data: Copulas

Some references

• *An Introduction to Copulas* by Roger B. Nelsen (1998), Springer-Verlag.

• *Multivariate Models and Dependence Concepts* by Harry Joe, (1997), Chapman and Hall/CRC.