Applied Data Mining (UN3106)
Spring 2018
http://stat.columbia.edu/∼porbanz/UN3106S18.html

Peter Orbanz
porbanz@stat.columbia.edu
Phyllis Wan
pw2348@columbia.edu

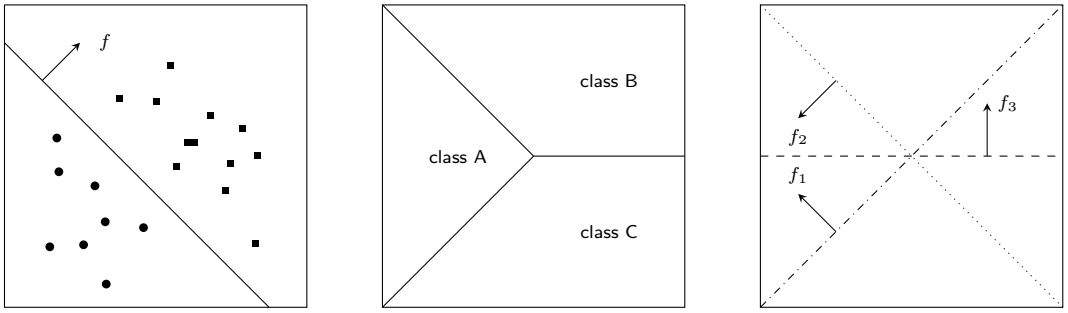
Homework 2

Due: 20 March 2018

Homework submission: We will collect your homework **at the beginning of class** on the due date. If you cannot attend class that day, you can leave your solution in Phyllis Wan's postbox in the Department of Statistics, 10th floor SSW, at any time before then.

We do not accept homework submitted late. There will be no exceptions.

Problem 1 (Trees)



(i) (ii) (iii)

1. Figure (i) shows a linear classifier f on a linearly separable training data set. Is it possible for a tree classifier to achieve the same training error as f?
2. Suppose the linear classifier f is also the unique Bayes-optimal classifier. Is it possible for a tree classifier to achieve the same risk as f?
3. The tree classifiers as we have discussed them so far (and as they are used in the two previous questions) represent axis-parallel planes, by making decisions of the form "is $x_j \geq c$?" We can, however, combine other classifiers using a decision tree: Given two-class classifiers $f_1, f_2, \ldots$, for example, which each output values ± 1, we can replace the decision by "is $f_i(\mathbf{x}) = +1$". Consider the three classes depicted in Figure (ii). The class boundaries are not axis-parallel. Use the three classifiers f_1, f_2, f_3 depicted in Figure (iii). Construct (=draw) a decision tree that combines the three into a three-class classifier g that classifies every point in Figure (ii) correctly.

Problem 2 (10-fold Cross Validation)

Consider a two-class classification problem with zero-one loss and training data set $\mathcal{X}_{\text{train}} = \{(\tilde{\mathbf{x}}_1, \tilde{y}_1), \ldots, (\tilde{\mathbf{x}}_n, \tilde{y}_n)\}$, where the class labels $\tilde{y}_i \in \{0, 1\}$. Give a test data point $\mathbf{x}$, recall that the k-nearest neighbor classifier calculate $\hat{y}$, the predicted class of $\mathbf{x}$, as follows:

- Find the k points in $\{\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n\}$ that are closest to $\mathbf{x}$ (in terms of Euclidean distance in $\mathbb{R}^d$).
- Predict $\hat{y}$ to be the majority class to which the k closest points belong.

We denote the above prediction to be

$$\hat{y} = f(\mathbf{x}; k, \mathcal{X}_{\text{train}}).$$

Question: Given data set $\mathcal{X} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, describe a step-by-step 10-fold cross validation procedure to choose an optimal value for the parameter k out of the values $\{1, 3, 5, 7, 9\}$. You can use the notation f defined above. Remember to address the following issues:

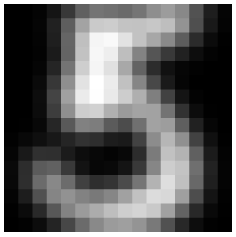
1. What are the 10 folds?
2. For each fold, what do you use as the training data and what do you use as the *validation* data?
3. What quantity do you compare for $k \in \{1, 3, 5, 7, 9\}$?
4. How do you determine which k is optimal?

Extra question: Nearest neighbor methods often work surprisingly well. Can you think of a reason why they may nonetheless be an inconvenient choice for an application running, for example, on a phone or a digital camera?

Problem 3 (Cross validating a nearest neighbor classifier)

A nearest neighbor classifier requires a parameter (the number k of neighbors used to classify). We will use cross validation to select the value of k for a specific type of data, handwritten digits.

Data. Download the digit data set from the course website. The zip archive contains two files: Both files are text files. The file `uspsdata.txt` contains a matrix with one data point (= vector of length 256) per row. The 256-vector in each row represents a 16×16 image of a handwritten number. The file `uspscl.txt` contains the corresponding class labels. The data contains two classes—the digits 5 and 6—so the class labels are stored as -1 and $+1$, respectively. The image on the right shows the first row, re-arranged as a 16×16 matrix and plotted as a gray scale image.



1. Read in the data into `R` from the dataset. It more convenient to model categorical data with the class `factor` in `R`. Use the function `as.factor` to transform class labels to `factor` class.
2. Plot the first four images using the following function. Note that the input `x` should be a numerical vector of length 256.

```
image.print <- function(x)
{
  x.matrix <- matrix(x,16,16,byrow=F)
  x.matrix.rotated <- t(apply(x.matrix, 1, rev))
  image(x.matrix.rotated, axes = FALSE, col = grey(seq(0, 1, length = 256)))
}
```

   The original image was scanned bottom up from the right. So here we first transform `x` into a 16×16 matrix, and then rotate and transpose the data matrix.

3. Randomly split the data into:
   - Training data (60% of the entire data set).
   - Test data (20%).
   - Validation data (20%).
4. The `R` function `knn` in library `class` implements the k-nearest neighbor classifier. We will only work with two classes and odd values of k, so you do not have to implement tie-breaking.

- Train a 1-nearest neighbor classifier using the training data and predict the labels of the images in the testing data. What is the test error (the empirical error rate on the test set)?
- Plot the misclassified images.

5. Select k as follows:
   - For $k \in \{1, 3, 5, 7, 9, 11, 13\}$, train the k-nn classifier on the training data and classify the images in the test set. Compute the test error for each k.
   - Which value of k should you choose? Why?
   - Finally, compute the error rate of the classifier for the optimal value of k on the validation set.