Homework 2

Hector Corrada Bravo

February, 2010

Pen-and-paper

1. We have seen many instances where *training error*, that is error on the training set, under-estimates test set error, an estimate of *expected prediction error*. For linear regression models we can show mathematically that, on average, training set error under-estimates test set error.

Suppose you have training set (x_i, y_i) for $i = 1, \ldots, N$ drawn at random from a population. Assume the following relationship holds:

$$Y = X\beta + \epsilon$$

with $\mathrm{E}(\epsilon) = 0$ and $\mathrm{var}(\epsilon) = \sigma^2$. Assume also that you have a test set $(\tilde{x}_i, \tilde{y}_i)$ for $i = 1, \ldots, N$ drawn from the same population as the training set.

Define training set error for estimate β as $R_{tr}(\beta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - x_i'\beta)^2$, and define test set error as $R_{te} = \frac{1}{N}\sum_{i=1}^{N}(\tilde{y}_i - \tilde{x}_i'\beta)^2$.

Show that, on average, the training-set error is less than or equal to the test-set error for the least squares estimate $\hat{\beta}^{\mathrm{ls}}$, estimated from the training set. That is, show

$$\mathrm{E}R_{tr}(\hat{\beta}^{\mathrm{ls}}) \leq \mathrm{E}R_{te}(\hat{\beta}^{\mathrm{ls}})$$

Hint: You will find it useful to write $R_{te}(\hat{\beta}^{\mathrm{ls}})$ in terms of $R_{te}(\beta^*)$ where β^* is the least squares estimate from the test set.

Data analysis

Explore some of the regression methods we have seen so far using a dataset of ozone-level measurements (dataset `ozone` in the `ElemStatLearn` package). Use polynomials of degrees 1,2,3,4,5 to fit ozone concentration as a function of daily

maximum temperature. For each of these polynomials use linear regression, ridge regression, and the lasso. Also try k-nearest neighbors.

Report on the relationship of expected prediction error (e.g. 5-fold cross validated residual sum of squares) and model complexity (polynomial degree and complexity parameters λ and k).

A couple of notes:

1. Use function `poly` to get the polynomial basis.
2. Use the `knnReg` function here `http://www.biostat.jhsph.edu/~hcorrada/PracticalML/src/knn.R` to do k-nearest neighbors.

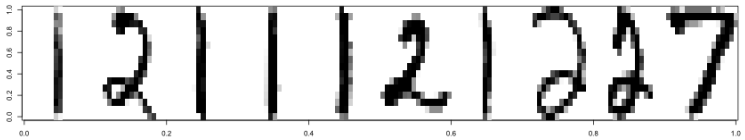
Make some predictions

The contest for today is hand-written digit recognition. You will train a classifier on images of hand-written ones (1), twos (2) and sevens (7). Your goal is to build a classifier that predicts the digit represented by new images.

Follow these instructions in R to get your training images:

```
> library(ElemStatLearn)
> data(zip.train)
> ?zip.train
>
load(url("http://www.biostat.jhsph.edu/~hcorrada/PracticalML/Data/zip_indices.rda"))
> train.set <- zip.train[train.inds,]
```

This will give you a matrix of size 400 by 257. The first column indicate which digit is represented in each row, the remaining 256 columns are gray-scale values of a 16 by 16 image. The first ten rows in the training set are the following images:



A couple of notes:

1. There is a nice function `zip2image` in the `ElemStatLearn` package you can use to see your data

2. We will test your classifier on a subset of ones, twos and sevens in the `zip.test` dataset.

3. The company who's paying for the prizes this time, really likes to see handwritten ones, so mis-classifying a one as any other digit is twice as costly (think of it as making two classification errors instead of one).

You will hand-in an R file, along with any other data required to run it, that takes new images and makes predictions. It should contain a function called `digit_predict` that takes a matrix of images in the same form as your training set and returns a vector of predictions. Something like:

```
digit_predict <- function(zips)
  {
    # do whatever you need to make predictions
    preds <- ...

    return(preds)
  }
```

First prize goes to whoever makes the fewest weighted mis-classifications (remember that misclassifying ones as another digit counts as two mistakes). Second prize goes to however best estimates their 3-by–3 *confusion matrix*, i.e. the rate at which your classifier misclassifies each digit as another digit, e.g. the rate at which ones are misclassified as twos, etc.

Handing in

This homework is due on Monday March 8. The pen-and-paper section is due at the beginning of class (1:30pm) along with writeups of the analysis and prediction sections. Please send the code you used for the analysis and prediction sections along with your digit-recognition classifier code to hcorrada@jhsph.edu with subject [Practical ML HW 2].