Introduction and Regression

Advanced Methods for Data Analysis (36-402/36-608)

Spring 2014

1 Course logistics

- Instructor: Ryan Tibshirani, TAs: Robert Lunde, Sonia Tardova
- See course website: `http://www.stat.cmu.edu/~ryantibs/advmethods/` for syllabus, office hours, course calendar, etc.
- 12 homeworks, 1 take-home exam, 1 in-class exam, 1 take-home final
- Class outline: 55 minutes lecture, 5 minutes break, 20 minutes working through R examples

2 Regression and prediction

2.1 Point prediction

- In the most simple *regression* setup we begin with $Y \in \mathbb{R}$, a real-valued random variable
- What is the optimal point prediction for Y? Depends on how we measure error. If we consider *mean squared error*,
$$\mathrm{MSE}(r) = \mathbb{E}[(Y - r)^2],$$
then the optimal prediction is $r = \mathbb{E}(Y)$
- In practice how would we make this point prediction? We estimate the expected value: given sample values $y_1, y_2, \ldots y_n$, we take
$$\hat{r} = \frac{1}{n} \sum_{i=1}^{n} y_i.$$
- Example: suppose I'm taking a particular cholesterol medication, predict my improvement in blood cholesterol level

2.2 Regression function

- Point prediction has limited applicability; more often we want to predict Y based on another variable X, called the *predictor*, *covariate*, or *input*. In this context Y is now often called the *response*, *outcome*, or *output*
- We make our prediction for Y a function of X, denoted $r(X)$; e.g., what will be the amount of improvement in blood cholesterol level as a function of the dose of a particular medication X?

- Again sticking with mean squared error,

$$\mathrm{MSE}(r) = \mathbb{E}\big[\big(Y - r(X)\big)^2\big] = \mathbb{E}\Big[\mathbb{E}\big[\big(Y - r(X)\big)^2 | X\big]\Big],$$

we can see that the optimal function is $r(X) = \mathbb{E}(Y|X)$, or $r(x) = \mathbb{E}(Y|X = x)$. We call this the *regression function*, it's what we use to predict Y from X

- We're going to assume that we can write

$$Y = r(X) + \varepsilon,$$

where ε is a random variable, called the *error term*, that has mean zero and is independent of X. Check: what happens when we take expectation conditional on $X = x$ on both sides?

- Disclaimer: this model assumes nothing about causality! Only a predictive relationship. And even once we are clear about this, assuming that the error ε is independent of X is not always realistic ... why?

2.3 How to estimate the regression function?

- From samples $(x_1, y_1), (x_2, y_2,), \ldots (x_n, y_n)$, we could estimate the regression function via

$$\hat{r}(x) = \frac{1}{n_x} \sum_{i:x_i = x} y_i, \tag{1}$$

where $n_x = |\{i : x_i = x\}|$. But this really only works when X takes discrete values; otherwise, at any x, we'd typically have only one or zero x_i values. Back to cholesterol example: what happens if we didn't observe any person in our study who took exactly $x = 31.5$ mg weekly of the drug?

- One way you already know to estimate the regression function, if you're willing to use a linear relationship: *linear regression*. We find the best fitting linear function, i.e., we find α, β to minimize

$$\mathrm{MSE}(\alpha, \beta) = \mathbb{E}[(Y - \alpha - \beta X)^2].$$

Taking derivatives, it is not hard to see that

$$\beta = \frac{\mathrm{Cov}(X, Y)}{\mathrm{Var}(X)}, \quad \alpha = \mathbb{E}(Y) - \beta \mathbb{E}(X)$$

- In sample, we can do one of two things: first, just plug in sample estimates to get

$$\hat{\beta} = \frac{\sum_{i=1}^n (x_i - \bar{x}) y_i}{\sum_{i=1}^n (x_i - \bar{x})^2}, \quad \hat{\alpha} = \bar{y} - \hat{\beta}\bar{x},$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$ and similarly for $\bar{y}$; second, find $\hat{\alpha}, \hat{\beta}$ to minimize

$$\sum_{i=1}^n (y_i - \alpha - \beta x_i)^2,$$

which ends up yielding the same estimates as above

- Let's look at this slightly differently: to make a prediction at an arbitrary point x, we use

$$\begin{aligned}\hat{r}(x) &= \hat{\alpha} + \hat{\beta}x \\ &= \hat{\alpha} + \frac{\sum_{i=1}^n (x_i - \bar{x}) y_i}{\sum_{i=1}^n (x_i - \bar{x})^2} x_i \\ &= \hat{\alpha} + \sum_{i=1}^n \frac{(x_i - \bar{x})x}{n s_x^2} \cdot y_i \end{aligned} \tag{2}$$

  where we abbreviated $s_x^2 = \sum_{i=1}^n (x_i - \bar{x})^2$

2.4 Linear smoothers

- From the above, we can see that the linear regression prediction (2) is of the form

$$\hat{r}(x) = \sum_{i=1}^n w(x, x_i) \cdot y_i, \tag{3}$$

  with $w(x, x_i) = (x_i - \bar{x})x/(ns_x^2)$. So is (1), with $w(x, x_i) = 1/n_x$ if $x_i = x$ and 0 otherwise, where recall $n_x = |\{i : x_i = x\}|$. In general, we call a regression function of the form (3) a *linear smoother* (note this is so-named because it is *linear in* y, but need not behave linearly as a function of x!)

- Note that the weight $w(x, x_i) = (x_i - \bar{x})x/(ns_x^2)$ does not actually depend on how far x_i is from x; this is not a problem if the true regression function is a straight line, but otherwise it is

- Another common, more flexible linear smoother is *k-nearest-neighbors regression*; for this we take

$$w(x, x_i) = \begin{cases} 1/k & \text{if } x_i \text{ is one of the } k \text{ nearest points to } x \\ 0 & \text{otherwise,} \end{cases}$$

  like an extension of (1). In other words

$$\hat{r}(x) = \frac{1}{k} \sum_{i \in N_k(x)} y_i, \tag{4}$$

  where $N_k(x)$ gives the k nearest neighbors of x

- To use this in practice, we're going to need to choose k. What are the tradeoffs at either end? For small k, we risk picking up noisy features of the sample that don't really have to do with the true regression function $r(x)$, called *overfitting*; for large k, we may miss important details, due to averaging over too many y_i values, called *underfitting*

2.5 Training and test errors

- How are we going to quantify this (overfitting vs underfitting)? Let's call $(x_1, y_1), \ldots (x_n, y_n)$, the sample of data that we used to fit $\hat{r}$, our *training sample*. What's wrong with looking at how well we do in fitting the training points themselves, i.e., the *expected training error*,

$$\mathbb{E}[\text{TrainErr}(\hat{r})] = \mathbb{E}\Big[\frac{1}{n} \sum_{i=1}^n \big(y_i - \hat{r}(x_i)\big)^2\Big]?$$

  This is way too optimistic, and doesn't have the right shape as a function of k!

- Suppose that we an independent *test sample* $(x'_1, y'_1), (x'_2, y'_2), \ldots (x'_m, y'_m)$ (following the same distribution as our training sample). We could then look at the *expected test error*,

$$\mathbb{E}[\text{TestErr}(\hat{r})] = \mathbb{E}\Big[\frac{1}{m}\sum_{i=1}^{m}\big(y'_i - \hat{r}(x'_i)\big)^2\Big].$$

  Note that the expectation here is taken over all that is random (both training and test samples). This really does capture what we want, and has the right behavior with k

- In short, underfitting leads to bad training error, but overfitting leads to good training error; both underfitting and overfitting lead to bad test errors (which is really we care about). Our goal is to find the right balance. We will see soon how to do this without access to test data. We will also see that underfitting and overfitting are related to two quantities called estimation bias and estimation variance (or just bias and variance)

2.6 Smoother linear smoothers

- Let's look back at the (3); recall that in k-nearest-neighbors regression, the weights $w(x, x_i)$ are either $1/k$ or 0, depending on x_i. This will often produce jagged looking fits. Why? Are these weights smooth over x?

- How about choosing

$$w(x, x_i) = \frac{\exp(-(x_i - x)^2/(2h))}{\sum_{j=1}^{n}\exp(-(x_j - x)^2/(2h))}?$$

  This is an example of *kernel regression*, using what's called a *Gaussian kernel*. The parameter h here is called the *bandwidth*, like k in k-nearest-neighbors regression, it controls the level of adaptivity/flexibility of the fit. One reason to prefer kernel regression is that it can produce smoother (less jagged) looks fits than k-nearest-neighbors