COS 424: Interacting with Data

Lecturer: Rob Schapire
Scribe: Zia Khan

Lecture #14
April 3, 2007

Recall from previous lecture that in regression we are trying to predict a real value given our data. Specifically, in the New Jersey school district example discussed in class, we looked at a linear model $\hat{f}(x_i) = wx_i$ where we related x_i enrollment in district i and tried to predict the budget y_i for district i based on w the money spent per student. In general, we would like to handle multiple dimensions or fit a more complex model to the data.

1 Linear Regression

To generalize our original model we start with a vector $\mathbf{x} \in \mathbb{R}^n$ where n is the dimensionality of the data. We use the notation $x_i(j)$ to denote the jth component of the ith vector $\mathbf{x_i}$. We can write all of the components of

$$\mathbf{x_i} = \langle x_i(1), \ldots, x_i(n) \rangle$$

which are referred to as variables, predictors, features or attributes - depending on the data we are interested in modelling. In general, the linear model will look as follows

$$\hat{f}(\mathbf{x}) = w_1 x(1) + w_w x(2) + \ldots + w_n(n) = \mathbf{w} \cdot \mathbf{x}$$

where n is the dimensionality of the data. To fit our model we want to minimize

$$\sum_{i=1}^{m} \left(\hat{f}(x_i) - y_i \right)^2 = \sum_{i=1}^{m} (\mathbf{w} \cdot \mathbf{x_i} - y_i)^2$$

where m is the number of data points. The process of fitting this model is called *linear regression.*

If we look back at the New Jersey school district data, we might want to model w_0 the fixed cost for students as well as w_1 the cost per student. Consequently, we should fit the model

$$\hat{f}(x_i) = w_0 + w_1 x_i$$

to our data set where x_i is the enrollment in the district. To fit this model we use a trick where we replace each data point with a 2-dimensional vector $x_i \rightarrow \langle 1, x_i \rangle$. It is easy to see that when we add 1 as the first dimension of each vector $\mathbf{x_i}$, we obtain the model above.

When we fit the model to the data, we obtain $w_0 = -3,540,476$ for the fixed cost for students and $w_1 = 12,054$ for the cost per student. The fixed cost number makes little sense. We can fix this by allowing for larger variance in the cost for school districts and

we obtain $w_0 = 99,138$ for the fixed cost for students and $w_1 = 12,054$ for the cost per student. This illustrates one of the many ways we can adjust our data so that we obtain a model that is a better fit for our data.

We can take this further and consider a model where we account for the higher cost of special education students

$$\hat{f}(x_i) = w_0 + w_1 x_i(1) + w_2 x_i(2)$$

where $x_i(1)$ is the number of regular students and $x_i(2)$ is the number of special education students in district i. When we fit our model we prepend one to the vector $\langle x_i(1), x_i(2)\rangle \rightarrow \langle 1, x_i(1), x_i(2)\rangle$ to account for w_0 the fixed cost for students. With this higher dimensional model we get $w_0 = 154,192$ for the fixed cost for students and $w_1 = 8,495$ for the cost per regular student and a larger $w_2 = 35,288$ cost per special education student.

2 Linear Model

Now we consider the general problem of fitting a vector

$$\mathbf{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_n \end{pmatrix}$$

to data. To do so we define a matrix

$$\mathbf{X} = \begin{pmatrix} \mathbf{x_1} \\ \mathbf{x_2} \\ \vdots \\ \mathbf{x_m} \end{pmatrix}$$

that has m rows and n columns where m is the number of data points and n is the dimensionality of the data. Thus, the ith row of $\mathbf{X}$ is equal to the data vector $\mathbf{x_i}$. We also define a vector

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}$$

where y_i is the predicted value associated with the ith data vector $\mathbf{x_i}$.

With these definitions we can rephrase the minimization in terms of matrices and vectors

$$\mathbf{Xw} - \mathbf{y} = \begin{pmatrix} \mathbf{w} \cdot \mathbf{x_1} - y_1 \\ \vdots \\ \mathbf{w} \cdot \mathbf{x_m} - y_m \end{pmatrix}$$

The Euclidean length squared, or L_2-norm, of the vector on the right hand side is just our original objective for linear regression

$$\sum_{i=1}^{m} (\mathbf{w} \cdot \mathbf{x_i} - y_i)^2 = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \tag{1}$$

We can minimize this objective by multiplying out the norm

$$\min_{\mathbf{w}} (\mathbf{X}\mathbf{w} - \mathbf{y})^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) = \min_{\mathbf{w}} \mathbf{w}^\top \mathbf{X}^\top \mathbf{X}\mathbf{w} - 2\mathbf{w}^\top \mathbf{X}^\top \mathbf{y} + \mathbf{y}^\top \mathbf{y}$$

and computing the gradient

$$\bigtriangledown_{\mathbf{w}} = 2\mathbf{X}^\top \mathbf{X}\mathbf{w} - 2\mathbf{X}^\top \mathbf{y} = 0$$

and setting it equal to zero. Now we can solve for $\mathbf{w}$ by computing the inverse

$$\begin{aligned} \left(\mathbf{X}^\top \mathbf{X}\right) \mathbf{w} &= \mathbf{X}^\top \mathbf{y} \\ \mathbf{w} &= \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y} \end{aligned}$$

The n by m matrix $\left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top$ is referred to as the pseudo-inverse. Of course, there is no guarantee that the pseudo-inverse will exist. It only exists when $\left(\mathbf{X}^\top \mathbf{X}\right)^{-1}$ is non-singular, and in this case, the solution $\mathbf{w}$ is unique. However, even when $\mathbf{X}^\top \mathbf{X}$ is singular, there are techniques for computing the minimum of equation (1).

It is not all that limiting to use just a linear model. Many problems can be reduced to the linear case. In this lecture we considered two: the polynomial model and linear splines.

2.1 Fitting Polynomials

We can use linear models to fit polynomial models. In Figure 1, we fit a cubic model

$$\hat{f}(x) = w_0 + w_1 x + w_2 x + w_3 x^3$$

by mapping each data point to a vector $x_i \rightarrow \langle 1, x_i, x_i^2, x_i^3 \rangle$ and fitting a linear model as described above. If the data is in $n > 1$ dimensions and we want to fit a polynomial we use a similar mapping. For instance, for $n = 2$ we can fit a quadratic

$$\hat{f}(x_1, x_2) = a + bx_1 + cx_2 + dx_1 x_2 + ex_1^2 + fx_2^2$$

using the mapping $\langle x_1, x_2 \rangle \rightarrow \langle 1, x_1, x_2, x_1 x_2, x_1^2, x_2^2 \rangle$. In general, if we start with n dimensions and want to fit a degree k polynomial we will need to map our data up to $O(n^k)$ dimensions. As with SVMs we can solve the problem of explicitly mapping data vectors in higher dimensions by using the Kernel trick where we replace an inner product

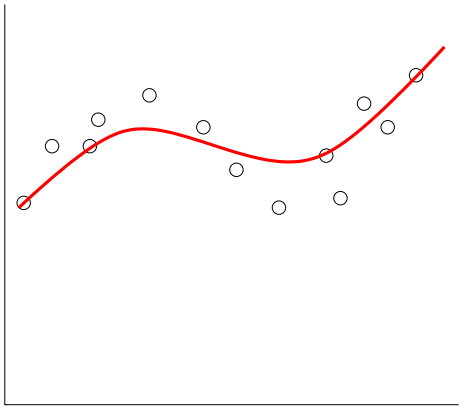

Figure 1: We can use linear regression to do polynomial regression. To fit a cubic polynomial to each data point x_i we map the data point to a vector $x_i \to \langle 1, x_i, x_i^2, x_i^3 \rangle$.

$\mathbf{x_i} \cdot \mathbf{x_j} = K(\mathbf{x_i}, \mathbf{x_j})$ with a kernel function. For the polynomial model we would use the kernel $K(\mathbf{x_i}, \mathbf{x_j}) = (1 + \mathbf{x_i} \cdot \mathbf{x_j})^k$. To obtain a procedure called *kernel regression,* our original objective must be rewritten in terms of inner products $\min \sum_{i=1}^{m} (\mathbf{w} \cdot \mathbf{x_i} - y_i)^2$. Of course, we face the same problem in kernel regression as we do in SVMs. As we increase the dimensionality of the data points we require more data.

2.2 Linear Splines

Linear splines offer another example of how to reduce a problem to linear regression. Here, we are trying to build a piece-wise linear function to fit the data (see Figure 2). For our discussion, we assume the knots, k_1 and k_2 in the example in Figure 2, are known in advance. We can imagine splitting the data points up into three groups using the given knots $k_1$and k_2 and solve three separate regression problems, but this does not assure continuity.

If we want the curve to be continuous, we can reduce the problem to straight linear regression by noticing that any linear spline can be a simple linear combination of basis functions. In the example in Figure 3, given knots k_1 and k_2 we can build up a piece-wise linear function step by step. We first start with a linear function

$$\hat{f}(x) = a + bx$$

which we use to account for points before the first knot k_1 (green line in Figure 3). Next, we add second line, the blue line in Figure 3, that is zero up until the first knot k_1

$$\hat{f}(x) = a + bx + c(x - k_1)_+$$

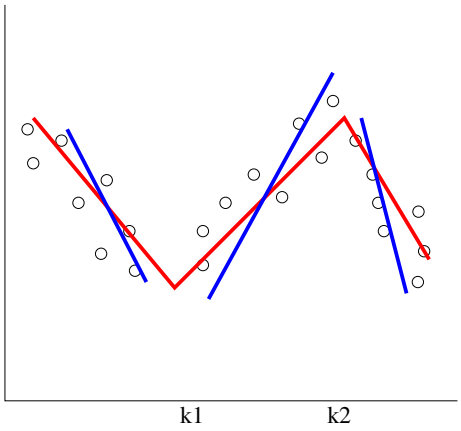

Figure 2: The aim of linear splines is to fit a piece-wise linear function such as the one shown in red. We can imagine splitting the data points up into three groups using the given knots $k_1$and k_2 and solve three separate regression problems. This, however, does not assure continuity.

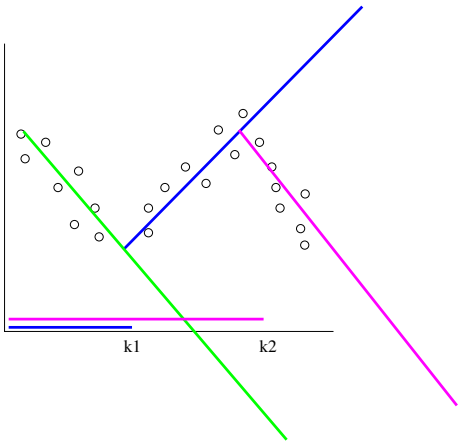

Figure 3: Any linear spline can be a simple combination of linear basis functions. To fit linear splines given the knots $k_1$and k_2 we map each point x_i we map the data point to a vector $x_i \rightarrow \langle 1, x, (x - k_1)_+, (x - k_2)_+ \rangle$ and solve the standard linear regression problem.

where

$$(z)_+ = \begin{cases} z & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

Last, we add a third line, the purple line in Figure 3, that is zero up until the second knot k_2

$$\hat{f}(x) = a + bx + c(x - k_1)_+ + d(x - k_2)_+$$

The coefficients in the function can be found by mapping each data point to a vector $x \to \langle 1, x, (x - k_1)_+, (x - k_2)_+ \rangle$ and solving the standard linear regression problem. This approach can generalize to higher order splines. It can incorporate additional constraints and generalize to higher dimensions.

3 Overfitting

3.1 Feature Selection

As we increase the dimensionality of our data, we run into the problem of overfitting. It is at least as bad in regression as it is in classification. The aim of *feature selection* is to find a subset of features/dimensions/variables that are best in some sense for our data. One can try to enumerate all 2^n features but for a large number of features this can be computationally expensive. Often one tries a greedy search method where features are added one-by-one to improve the objective or features are removed one-by-one to improve the objective.

3.2 Regularization

Another approach is to blame overfitting on weights that are too large. With large weight vectors it is easier to overfit our data. The idea is to constrain weight vectors in some way. *Regularization* refers to any time we add a penalty term that involves the weight vector. *Shrinkage* refers to any technique where we are shrinking the weights in some way.

One example of a regularization technique is *ridge regression*. Here we optimize the objective

$$\min_{\mathbf{w}} \sum_{i=1}^{m} (\mathbf{w} \cdot \mathbf{x_i} - y_i)^2 + \lambda \|\mathbf{w}\|_2^2$$

where the smoothing constant λ is chosen in advance. Larger values of λ tend to smooth curves by penalizing larger weight values through the L_2 norm of the weight vector $\|\mathbf{w}\|$. Finding a weight vector $\mathbf{w}$ for ridge regression is not hard. We can find a closed for solution in which we modify the pseudo-inverse

$$\mathbf{w} = \left(\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}\right)^{-1} \mathbf{X}^\top \mathbf{y}$$

where $\mathbf{I}$ is an n by n identity matrix.