CS142: Machine Learning Spring 2017

Lecture 2

Instructor: Pedro Felzenszwalb *Scribes: Dan Xiang, Tyler Dae Devlin*

Linear Regression, Basis Functions, Least Squares

Recall that the goal of supervised learning is to estimate a function $f : X \to Y$. It is often the case that Y is small, i.e. finite. In this case, we refer to the problem of estimating f as *classification* (e.g. classifying emails as spam / not spam).

This lecture will focus on examples where $X = \mathbb{R}^D$ and $Y = \mathbb{R}$. In this case, estimating f is called *regression* because the output set Y is infinite. An example of a regression problem might be estimating the fuel efficiency of a certain car at all of the car's possible speeds. In this case $X =$ the car's speed (mph) and $Y =$ the fuel efficiency (miles/gallon).

Let the *hypothesis space* H be a family of functions mapping $X \to Y$ and T be a training set $T = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_i \in \mathbb{R}^D$ and $y_i \in \mathbb{R}$ (in other words, each x_i is a D-dimensional vector whereas y_i is a real value). We shall select the function $g \in H$ that minimizes the squared error given by

$$E(g) = \frac{1}{2} \sum_{i=1}^{n} (g(x_i) - y_i)^2.$$

In words, the error function E looks at each training input x_i and computes the value of our estimate $g(x_i)$. It then looks at the difference between our estimate and the true value y_i and squares this error. After computing this squared-error for each of the n training examples, it sums them all together. We will see the purpose of the $\frac{1}{2}$ factor momentarily. To simplify matters, let H to be the set of linear functions

$$H = \left\{ h(x) : h(x) = w_0 + \sum_{i=1}^{D} w_i x_i \right\},$$

where $x = (x_1, \ldots, x_D) \in \mathbb{R}^D$. This definition of the hypothesis space H is inherently limiting. If our target function f deviates from linearity in a significant way, our estimate chosen from H will necessarily be a poor approximation.

With this in mind, let us expand H to be the set of functions that are linear combinations of a set of *basis functions*,

$$\phi_1, \ldots, \phi_M : \mathbb{R}^D \to \mathbb{R}.$$

Now every element $h \in H$ has the form

$$h(x) = w_0 + \sum_{i=1}^{M} w_i \phi_i(x),$$

where again $x = (x_1, \ldots, x_D) \in \mathbb{R}^D$.

Example. Define the following basis functions over the feature space $X = \mathbb{R}$:

$$\phi_1(x) = x,\ \phi_2(x) = x^2,\ \ldots,\ \phi_M(x) = x^M.$$

Then $h \in H$ is of the form

$$h(x) = w_0 + \sum_{k=1}^{M} w_k x^k,$$

which means that H is the set of all polynomials in x with degree no greater than M. □

Example. Define the basis functions

$$\phi_1(x) = \sin(x),\ \phi_2(x) = \cos(x),\ \phi_3(x) = \sin(2x),\ \phi_4(x) = \cos(2x)\ \ldots.$$

H, then, is the set of functions which are periodic and band-limited (bounded frequency). □

We define the feature mapping $\phi : \mathbb{R}^D \to \mathbb{R}^M$ by

$$\phi(x) = \begin{bmatrix} \phi_1(x) \\ \phi_2(x) \\ \vdots \\ \phi_M(x) \end{bmatrix}.$$

So H is the set of linear functions of $\phi(x)$. To summarize,

$$\mathbb{R}^D \xrightarrow{\phi} \mathbb{R}^M \xrightarrow{w} \mathbb{R},$$

where ϕ is a fixed nonlinear vector-valued function and w is a learned linear function. Then $h = w \circ \phi$.

Suppose we have some training data $T = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$ (note that each $x^{(i)} \in \mathbb{R}^D$) and H is the set of linear combinations of fixed basis functions $\phi_1, \ldots, \phi_M$. Then we can write every function in the hypothesis space H as a scalar (dot) product of the two vectors

$$h_w(x) = w \cdot \phi(x) = w^T \phi(x),$$

where $x = (x_1, \ldots, x_D) \in \mathbb{R}^D$, $\phi(x) = (\phi_1(x), \ldots, \phi_M(x)) \in \mathbb{R}^M$, and $w = (w_1, \ldots, w_M) \in \mathbb{R}^M$ (we drop w_0 since it is always possible add a constant basis function back in). Then h is not linear in x but it *is* linear in the vector of weights w. We can write down an error function

$$E(w) = \frac{1}{2}\sum_{i=1}^{n}(h_w(x^{(i)}) - y^{(i)})^2 = \frac{1}{2}\sum_{i=1}^{n}(w \cdot \phi(x^{(i)}) - y^{(i)})^2.$$

We want to find a weight vector w minimizing $E(w)$. So we search for the w such that

$$\nabla E(w) = 0$$

This is equivalent to the set of equations

$$\frac{\partial E}{\partial w_j} = \frac{1}{2}\sum_{i=1}^{n} 2(w \cdot \phi(x^{(i)}) - y^{(i)})\phi_j(x^{(i)}) = 0,$$

where $j \in \{1, \ldots, M\}$. Simplifying,

$$\begin{aligned}\sum_{i=1}^{n} y^{(i)}\phi_j(x^{(i)}) &= \sum_{i=1}^{n} w \cdot \phi(x^{(i)})\phi_j(x^{(i)}) \\ &= \sum_{i=1}^{n}\sum_{k=1}^{M} w_k\phi_k(x^{(i)})\phi_j(x^{(i)}).\end{aligned}$$

Swapping the order of summation, the above is equal to

$$= \sum_{k=1}^{M} w_k \sum_{i=1}^{n} \phi_k(x^{(i)})\phi_j(x^{(i)}).$$

Denoting

$$\begin{aligned}\mathcal{M}_{k,j} &\doteq \sum_{i=1}^{n} \phi_k(x^{(i)})\phi_j(x^{(i)}), \\ z_j &\doteq \sum_{i=1}^{n} y^{(i)}\phi_j(x^{(i)}),\end{aligned}$$

we have

$$\begin{aligned}\mathcal{M} &= \sum_{i=1}^{n} \phi(x^{(i)})\phi^T(x^{(i)}), \\ z &= \sum_{i=1}^{n} y^{(i)}\phi(x^{(i)}),\end{aligned}$$

where $\mathcal{M}$ is an $M \times M$ matrix and z is a $M \times 1$ vector. Thus, our minimization problem is equivalent to solving this linear system:

$$\boxed{\mathcal{M}w = z \iff \nabla E(w) = 0} \tag{1}$$

To solve this linear system in Matlab, use $w = \mathcal{M} \backslash z$.

If we have n data points and we try to fit using a polynomial of degree $M > n$, we will end up overfitting since there are too few data points for the complexity of the hypothesis space H.