2 Lecture Outline

Reading: Chapter 3

This lecture covers

- Linear Basis Function Models
- Maximum Likelihood and Least Squares
- Regularized Least Squares

3.0 Introduction to Linear Models for Regression

Given a training data set of D-dimensional observations $\{\mathbf{x}_n\}$, where $n = 1, \ldots, N$, together with corresponding target values $\{t_n\}$, the goal in regression is to predict the value of t for a new value of $\mathbf{x}$. In the simplest approach, this can be done by directly constructing an appropriate function $y(\mathbf{x})$ whose values for new inputs $\mathbf{x}$ constitute the predictions for the corresponding values of t.

More generally, from a probabilistic perspective, we aim to model the predictive distribution $p(t|\mathbf{x})$ because this expresses our uncertainty about the value of t for each value of $\mathbf{x}$. From this conditional distribution we can make predictions of t, for any new value of $\mathbf{x}$ in such a way as to minimize the expected value of a suitably chosen loss function. For the squared loss, we have already seen the optimal solution (prediction) is given by the conditional expectation of t, $\mathbb{E}[t|\mathbf{x}]$.

Figure 1: Block diagrams for training and test stages

Although linear models have significant limitations as practical techniques for pattern recognition, particularly for problems involving input spaces of high dimensionality, they have nice analytic properties and form the foundation for more sophisticated models to be discussed in later chapters.

3.1 Linear Basis Function Models

The simplest linear model for regression is one that involves a linear combination of the input variables

$$y(\mathbf{x}, \mathbf{w}) \quad = \quad w_0 + w_1 x_1 + \cdots + w_D x_D \tag{1}$$

where $\mathbf{x} = (x_1, \ldots, x_D)$. This is often simply known as *linear regression*. This model is a linear function of the parameters $\{w_j\}$ and a linear function of the input variables $\{x_n\}$ and this imposes significant limitations on the model.

However, we can obtain a much more useful class of functions by taking linear combinations of a fixed set of nonlinear functions of the input variables, known as *basis functions*. Such models are linear function of

the parameters, which gives them simple analytical properties and yet can be nonlinear with respect to the input variables.

We therefore extend the class of models by considering linear combinations of fixed, nonlinear functions of the input variables, of the form

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) \tag{2}$$

where the $\phi_j(\mathbf{x})$ are known as *basis functions*. There are M parameters $w_0, \ldots, w_M$.

The *bias* parameter w_0 allows for any fixed offset in the data. It is convenient to define an additional dummy 'basis function' $\phi_0(\mathbf{x}) = 1$ so that we may compactly write

$$\begin{aligned} y(\mathbf{x}, \mathbf{w}) &= w_0\phi_0(\mathbf{x}) + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) \\ &= \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) \\ &= \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) \end{aligned} \tag{3}$$

where $\mathbf{w} = (w_0, \ldots, w_{M-1})^T$ and $\boldsymbol{\phi} = (\phi_0, \ldots, \phi_{M-1})^T$. By using nonlinear basis functions, we allow the function $y(\mathbf{x}, \mathbf{w})$ to be a nonlinear function of the input vector $\mathbf{x}$. However, (3) is a called linear model because it is a linear function in $\mathbf{w}$.

Example Basis Functions

Polynomial basis: In polynomial regression, we have $\phi_j(x) = x^j$ so the model, as seen in Section 1.1, is

$$y(x, \mathbf{w}) = \sum_{j=0}^{M-1} w_j x^j. \tag{4}$$

If $M = 2$, this model is known as *linear regression* otherwise it is known as *polynomial regression*.

Gaussian basis: Another choice for the basis function includes the 'Gaussian' function

$$\phi_j(x) = \exp\left\{-\frac{(x - \mu_j)^2}{2s^2}\right\} \tag{5}$$

where the μ_j govern the locations of the basis functions in input space (localization), and the parameter s governs their spatial scale. As an example, if the data point x is near μ_j, the $\phi_j(x) \approx 1$ and if x is far from μ_j, the $\phi_j(x) \approx 0$.

Sigmoid basis: Another choice for the basis function include the 'sigmoidal' function

$$\phi_j(x) = \sigma\left(\frac{x - \mu_j}{s}\right) \tag{6}$$

where $\sigma(a)$ is the logistic sigmoid function defined by

$$\sigma(a) = \frac{1}{1 + \exp(-a)}. \tag{7}$$

Note that if the data point $x \approx \mu_j$, then $\phi_j(x) \approx 1/2$; if $x \gg \mu_j$, then $\phi_j(x) \approx 1$, and if $x \ll \mu_j$, then $\phi_j(x) \approx 0$. This is also a way to map a range of numbers to the interval $[0, 1]$ which can be thought of as a probability map.

3.1.1 Maximum Likelihood and Least Squares

In Chapter 1, we reformulated the polynomial regression problem in terms of the likelihood function

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}) \tag{8}$$

which expresses the uncertainty of the predicted t as normally-distributed, given the input $\mathbf{x}$ and coefficients $\mathbf{w}$. We assumed the target was given by an underlying deterministic function, $y(\mathbf{x}, \mathbf{w})$ with additive Gaussian noise

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \tag{9}$$

where $p(\epsilon|\beta) = \mathcal{N}(0, \beta^{-1})$ and thus

$$\mathbb{E}[t] = y(\mathbf{x}, \mathbf{w}). \tag{10}$$

Using the training data $\{\mathsf{x}, \mathsf{t}\}$ the likelihood function is expressed as

$$p(\mathsf{t}|\mathsf{x}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|y(\mathbf{x}_n, \mathbf{w}), \beta^{-1}) \tag{11}$$

and the log-likelihood function is given by

$$\ln p(\mathsf{t}|\mathsf{x}, \mathbf{w}, \beta) = \frac{-\beta}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n)\}^2 + \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi). \tag{12}$$

We showed that maximizing the log-likelihood function w.r.t. $\mathbf{w}$ was equivalent to minimizing the sum-of-squares error function (we didn't actually solve for $\mathbf{w}_{\mathrm{ML}}$). We also maximized the log-likelihood function w.r.t. β and found β_{ML}.

If we assume a squared loss function, then the optimal prediction of t for a new value of $\mathbf{x}$ is the conditional mean of the t:

$$\begin{aligned} \mathbb{E}[t|\mathbf{x}] &= \int t p(t|\mathbf{x}) dt \\ &= y(\mathbf{x}, \mathbf{w}). \end{aligned} \tag{13}$$

Least Squares Solution

We now will solve the linear regression problem via ML assuming a set of general basis functions $\boldsymbol{\phi}$ (not necessarily polynomial) and find $\mathbf{w}_{\mathrm{ML}}$.

Consider a data set of inputs $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ with corresponding target values $t_1, \ldots, t_N$. We group the target variables $\{t_n\}$ into a column vector t. Assuming the data points are i.i.d., we replace the underlying function $y(\mathbf{x}, \mathbf{w})$ with our model and obtain the likelihood function

$$p(\mathsf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \tag{14}$$

where $\boldsymbol{\phi}(\mathbf{x}_n) = [\phi_0(\mathbf{x}_n), \ldots, \phi_{M-1}(\mathbf{x}_n)]^T$.

Note that in supervised learning problems such as regression and classification, we are not seeking to model the distribution of the input variables. Thus $\mathbf{x}$ will always appear in the set of conditioning variables and so

we will drop the explicit $\mathbf{x}$ and write $p(\mathsf{t}|\mathbf{w},\beta)$. The log likelihood is given by

$$\begin{aligned}\ln p(\mathsf{t}|\mathbf{w},\beta) &= \sum_{n=1}^{N}\ln\mathcal{N}(t_n|\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n),\beta^{-1}) \\ &= \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) - \beta\frac{1}{2}\sum_{n=1}^{N}\{t_n-\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2 \\ &= \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) - \beta E_D(\mathbf{w}) \end{aligned} \tag{15}$$

where $E_D(\mathbf{w})$ is the sum-of-squares error function.

Setting the gradient to zero we have

$$\begin{aligned}\mathbf{0}^T &= \nabla\ln p(\mathsf{t}|\mathbf{w},\beta) \\ &= \beta\sum_{n=1}^{N}\{t_n-\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}\boldsymbol{\phi}(\mathbf{x}_n)^T \\ &= \sum_{n=1}^{N}t_n\boldsymbol{\phi}(\mathbf{x}_n)^T - \mathbf{w}^T\left(\sum_{n=1}^{N}\boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\phi}(\mathbf{x}_n)^T\right). \end{aligned} \tag{16}$$

We denote the $N\times M$ *design matrix* (or data/observation matrix)

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \dots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \dots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \dots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix} \tag{17}$$

so that (16) is written compactly as

$$\mathsf{t}^T\boldsymbol{\Phi} - \mathbf{w}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi} = \mathbf{0}^T \tag{18}$$

or

$$\boldsymbol{\Phi}^T\mathsf{t} - \boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{w} = \mathbf{0}. \tag{19}$$

We easily obtain the ML solution as

$$\begin{aligned}\mathbf{w}_{\text{ML}} &= \left(\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\mathsf{t} \\ &= \boldsymbol{\Phi}^{\dagger}\mathsf{t} \end{aligned} \tag{20}$$

which are known as the *normal equations* for the LS problem. The quantity $\boldsymbol{\Phi}^{\dagger} = \left(\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T$ is the *Moore-Penrose pseudo-inverse* of the matrix $\boldsymbol{\Phi}$. Thus given a new input $\mathbf{x}'$, the predicted target is $\mathbf{w}_{\text{ML}}^T\boldsymbol{\phi}(\mathbf{x}')$.

The ML solution for β (computed in Chapter 1) is

$$\frac{1}{\beta_{\text{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\left\{t_n-\mathbf{w}_{\text{ML}}^T\boldsymbol{\phi}(\mathbf{x}_n)\right\}^2. \tag{21}$$

Recap: The predictive distribution is given by

$$p(t|\mathbf{x},\mathbf{w}_{\text{ML}},\beta_{\text{ML}}) = \mathcal{N}(t|\mathbf{w}_{\text{ML}}^T\boldsymbol{\phi}(\mathbf{x}),\beta_{\text{ML}}^{-1}) \tag{22}$$

and the point estimate is given by

$$\hat{t} = \mathbb{E}[t|\mathbf{x}] = \mathbf{w}_{\text{ML}}^T\boldsymbol{\phi}(\mathbf{x}). \tag{23}$$

3.1.4 Regularized Least Squares

In Section 1.1, we introduced the idea of adding a regularization term to an error function in order to control over-fitting so that the total error function to be minimized takes the form

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w}) \tag{24}$$

where λ is the regularization coefficient that controls the relative importance of the regularization term $E_W(\mathbf{w})$. One of the simplest forms of regularizer is

$$E_W(\mathbf{w}) \quad = \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}. \tag{25}$$

Combining this with our usual sum-of-squares error function, the total error function becomes

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{1}{2}\lambda\mathbf{w}^T\mathbf{w}. \tag{26}$$

This particular choice of regularizer is known as *weight decay*. Setting the gradient of the error function with respect to $\mathbf{w}$ to zero, we obtain[1]

$$\mathsf{t}^T\boldsymbol{\Phi} - \mathbf{w}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi} - \lambda\mathbf{w}^T \quad = \quad \mathbf{0}^T \tag{27}$$

or

$$\boldsymbol{\Phi}^T\mathsf{t} - \boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{w} - \lambda\mathbf{w} \quad = \quad \mathbf{0}. \tag{28}$$

We easily obtain the ML solution as

$$\mathbf{w}_{\mathrm{ML}} \quad = \quad \left(\lambda\mathbf{I} + \boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\mathsf{t} \tag{29}$$

Regularization allows complex models to be trained on data sets of limited size without severe over-fitting, essentially by limiting the effective model complexity. However, the problem of determining the optimal model complexity is then shifted from one of finding the appropriate number of basis functions to one of determining a suitable value of λ.

[1] $\frac{\partial}{\partial\mathbf{w}^T}\mathbf{w}^T\mathbf{w} = 2\mathbf{w}^T$