COMP4131: Data Modelling and Analysis

Lecture 6: Linear Regression and Logistic Regression

Daokun Zhang

University of Nottingham Ningbo China

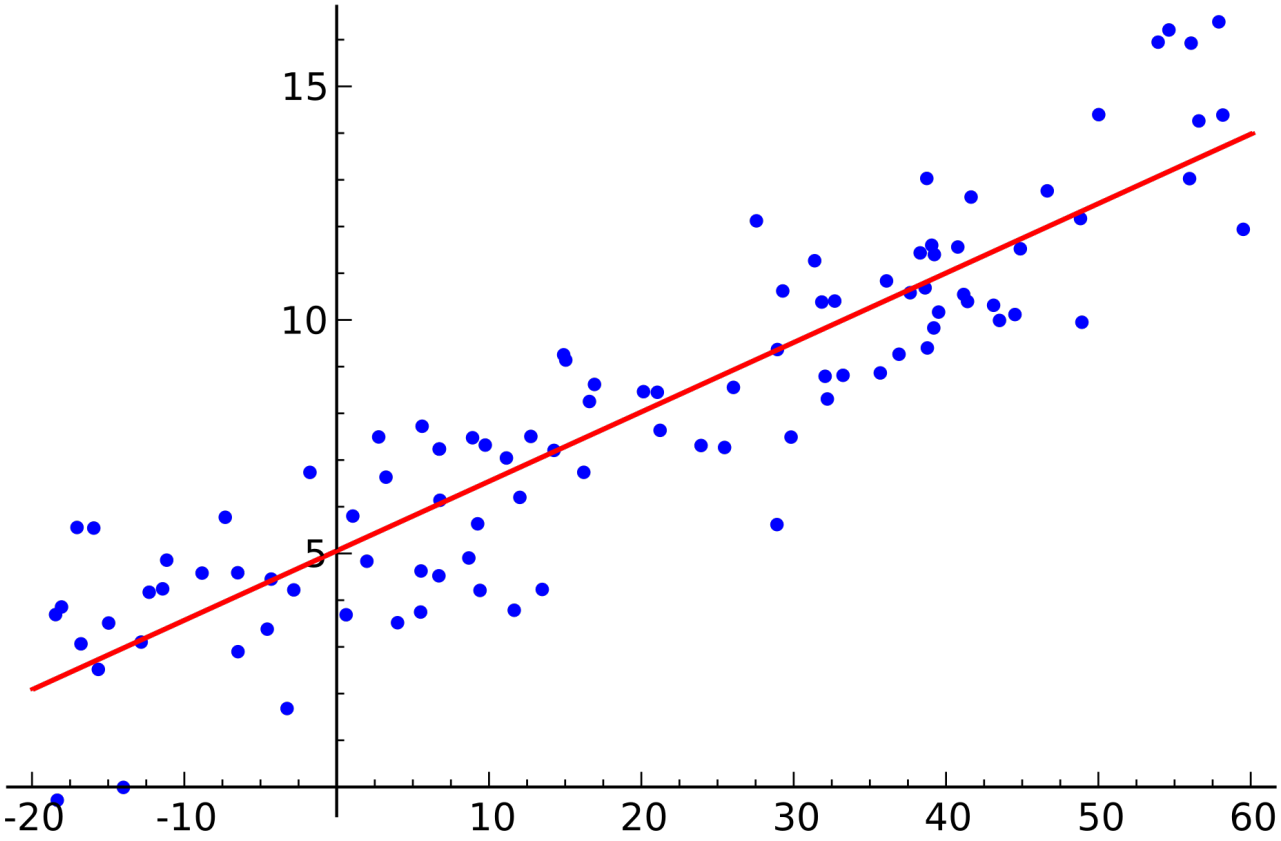
daokun.zhang@nottingham.edu.cn

March 24, 2025

Overview

Linear Regression

A Simple Example of Linear Regression



$$y = kx + b$$

Linear Regression with Multiple Input Variables

Given the input sample $\mathbf{x} = (x_1, x_2, \cdots, x_D)^{\mathrm{T}}$ with D feature dimensions, we can predict the target value $y \in \mathbb{R}$ as

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D,$$

where $\mathbf{w} = (w_0, w_1, \cdots, w_D)^{\mathrm{T}}$ are the parameters.

Here $y(\mathbf{x}, \mathbf{w})$ is

- a linear function of the parameters $w_0, w_1, \cdots, w_D$, and
- a linear function of the input variables $x_1, x_2, \cdots, x_D$.

Linear Basis Function Models

We can extend the linear regression model to the linear basis function models by considering linear combinations of fixed nonlinear functions of the input variables, of the form

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

where $\phi_j(\mathbf{x})$ are the basis functions, and M is the number of parameters.

To be more concise, we define an additional dummy 'basis function' $\phi_0(\mathbf{x}) = 1$ for the *bias* parameter w_0 so that

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

where $\mathbf{w} = (w_0, \cdots, w_{M-1})^{\mathrm{T}}$ and $\boldsymbol{\phi}(\mathbf{x}) = [\phi_0(\mathbf{x}), \cdots, \phi_{M-1}(\mathbf{x})]^{\mathrm{T}}$.

Examples of Basis Functions

- Feature Pre-processing: impute/remove missing values, feature normalization
- Feature Selection
- Dimension Reduction
- Identity Basis Function: $\phi(\mathbf{x}) = \mathbf{x}$
- Polynomial Basis Function: $\phi_j(x) = x^j$
- Gaussian Basis Function: $\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$
- Sigmoidal Basis Function: $\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$ with $\sigma(a) = \frac{1}{1+\exp(-a)}$
- Fourier Basis Function
- ...

Maximum Likelihood and Least Squares

Now consider a data set of inputs $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ with corresponding target values $\mathbf{t} = \{t_1, \cdots, t_N\}$, to find the best parameter $\mathbf{w}$, we need to minimize the sum-of-squares error function:

$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - y(\mathbf{x}, \mathbf{w})\}^2$$
$$= \frac{1}{2} \sum_{n=1}^{N} \left\{t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}\left(\mathbf{x}_n\right)\right\}^2 .$$

Question: Why the square error is used to design the objective function?

Maximum Likelihood and Least Squares

We assume that the target variable t is given by a deterministic function $y(\mathbf{x}, \mathbf{w})$ with additive Gaussian noise so that

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

where ϵ is a zero mean Gaussian random variable with precision (inverse variance) β. Thus we can write

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

where

$$\mathcal{N}\left(x|\mu, \sigma^2\right) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}.$$

Maximum Likelihood and Least Squares

For the given data set $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and $\mathbf{t} = \{t_1, \cdots, t_N\}$, the joint likelihood function can be expressed as

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

Taking the logarithm of the likelihood function, and making use of the standard form for the univariate Gaussian, we have

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

Maximum Likelihood and Least Squares

The gradient of the sum-of-squares error function $E_D(\mathbf{w})$ takes the form

$$\nabla E_D(\mathbf{w}) = -\sum_{n=1}^{N} \left\{ t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n) \right\} \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}}.$$

Setting this gradient to zero gives

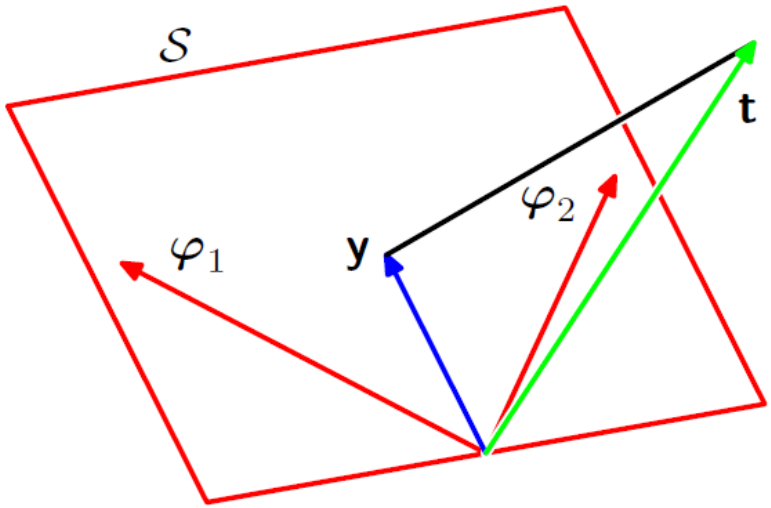
$$0 = -\sum_{n=1}^{N} t_n \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} + \mathbf{w}^{\mathrm{T}} \left(\sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}_n) \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} \right).$$

Solving for $\mathbf{w}$ we obtain

$$\mathbf{w}^* = \left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t},$$

which are known as the *normal equations* for the least squares problem. Here $\boldsymbol{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1), \cdots, \boldsymbol{\phi}(\mathbf{x}_N)]^{\mathrm{T}}$ is an $N \times M$ matrix.

Geometry Interpretation of Least Squares



- consider $\mathbf{t} = \{t_1, \cdots, t_N\}$ is a vector in an N-dimensional space
- define $\varphi_j = \{\phi_j(\mathbf{x}_1), \cdots, \phi_j(\mathbf{x}_N)\}$ as the jth column vector of $\mathbf{\Phi}$
- $\{\varphi_0, \cdots, \varphi_{M-1}\}$ span a subspace $\mathcal{S}$ of dimensionality M, if $M < N$
- the least-squares solution corresponds to the choice of $\mathbf{y} = \{y(\mathbf{x}_1, w), \cdots, y(\mathbf{x}_N, w)\}$ that lies in subspace $\mathcal{S}$ and that is closest to $\mathbf{t}$, i,e, the projection of $\mathbf{t}$ onto $\mathcal{S}$

Regularized Least Squares

To avoid over-fitting, we consider adding a regularization term to an error function, so that the total error function to be minimized takes the form

$$E(\mathbf{w}) = E_D(\mathbf{w}) + \lambda E_W(\mathbf{w}),$$

where λ is the regularization coefficient, $E_D(\mathbf{w})$ is the data-dependent error, and $E_W(\mathbf{w})$ is the regularization term:

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\left(\mathbf{x}_n\right)\right\}^2,$$

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

Then the total error function becomes

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

Regularized Least Squares

The gradient of the new error function $E(\mathbf{w})$ takes the form

$$\nabla E(\mathbf{w}) = -\sum_{n=1}^{N} \left\{ t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n) \right\} \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} + \lambda \mathbf{w}.$$

Setting this gradient to zero gives

$$0 = -\sum_{n=1}^{N} t_n \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} + \mathbf{w}^{\mathrm{T}} \left(\sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}_n) \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} \right) + \lambda \mathbf{w}^{\mathrm{T}}.$$

Solving for $\mathbf{w}$ we obtain

$$\mathbf{w}^* = \left(\lambda \mathbf{I} + \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t},$$

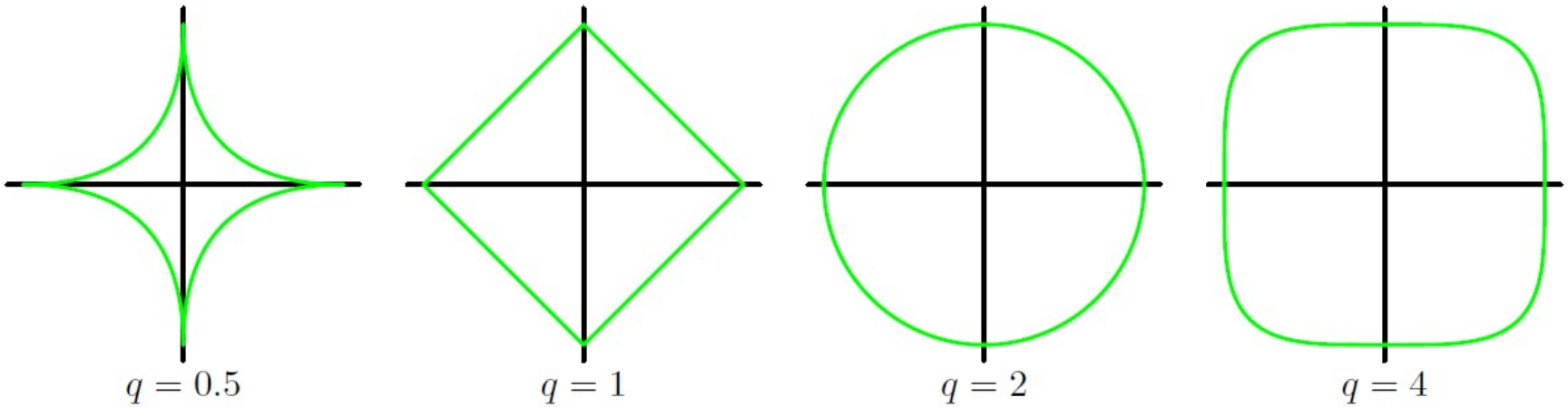
where $\mathbf{I}$ is the $M \times M$ identity matrix.

Regularized Least Squares

A more general regularizer is sometimes used, for which the regularized error takes the form

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\left(\mathbf{x}_n\right)\right\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q,$$

where $q = 2$ corresponds to the *quadratic* regularizer, and the case of $q = 1$ know as the *lasso* regularizer.

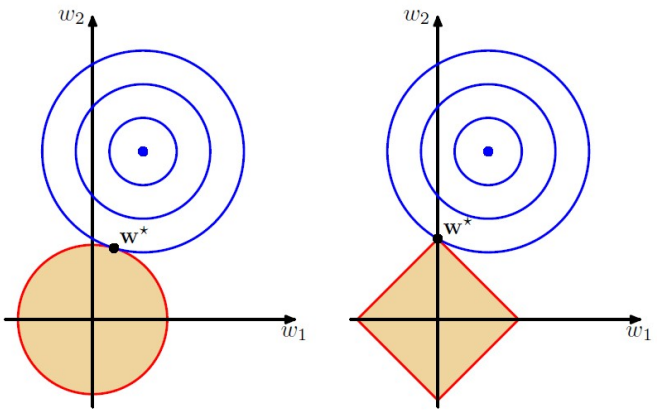


Contours of the regularization term for various values of the parameter q.

Regularized Least Squares

Minimizing the regularized error $E(\mathbf{w})$ is equivalent to

$$\min_{\mathbf{w}} \underbrace{\frac{1}{2}\sum_{n=1}^{N}\left\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\left(\mathbf{x}_n\right)\right\}^2}_{E_D(\mathbf{w})}, \; s.t. \; \sum_{j=1}^{M}|w_j|^q \leqslant \eta.$$



The contours of the unregularized error function (blue) along with the constraint region for the quadratic regularizer $q = 2$ on the left and the lasso regularizer $q = 1$ on the right, in which the optimum value for the parameter vector $\mathbf{w}$ is denoted by $\mathbf{w}^*$. The lasso gives a sparse solution in which $w_1^* = 0$.

The Bias-Variance Decomposition

Given the joint distribution $p(\mathbf{x}, t)$ of the input $\mathbf{x}$ and target value t, we can determine the specific estimate $y(\mathbf{x})$ of the value t for each input $\mathbf{x}$ (a mapping function from $\mathbf{x}$ to t), as the function that minimizes the expected squared loss

$$\mathbb{E}_{(\mathbf{x},t)}[L] = \iint \{y(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) \mathrm{d}\mathbf{x} \mathrm{d}t.$$

The optimal $y(\mathbf{x})$ should be

$$y^*(\mathbf{x}) = h(\mathbf{x}) = \int t p(t|\mathbf{x}) \mathrm{d}t = \mathbb{E}_t[t|\mathbf{x}].$$

For a given estimate function $y(\mathbf{x})$, the expected squared loss can be expressed as

$$\begin{aligned}\mathbb{E}_{(\mathbf{x},t)}[L] &= \iint \{y(\mathbf{x}) - h(\mathbf{x}) + h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) \mathrm{d}\mathbf{x} \mathrm{d}t \\ &= \int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x}) \mathrm{d}\mathbf{x} + \iint \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) \mathrm{d}\mathbf{x} \mathrm{d}t.\end{aligned}$$

The Bias-Variance Decomposition

In general, the prediction function $y(\mathbf{x})$ is learned from a given data set $\mathcal{D}$, the we use $y(\mathbf{x}; \mathcal{D})$ to denote $y(\mathbf{x})$, and write the squared difference between $y(\mathbf{x})$ and $h(\mathbf{x})$ as

$$\{y(\mathbf{x}; \mathcal{D}) - h(\mathbf{x})\}^2.$$

If we add and subtract the quantity $\mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})]$ inside the braces, and then expand, we obtain

$$\begin{aligned}
&\{y(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] + \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] - h(\mathbf{x})\}^2 \\
&= \{y(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})]\}^2 + \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] - h(\mathbf{x})\}^2 \\
&\quad + 2\{y(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})]\}\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] - h(\mathbf{x})\}.
\end{aligned}$$

We now take the expectation of this expression with respect to $\mathcal{D}$ and note that the final term will vanish, giving

The Bias-Variance Decomposition

$$\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D})-h(\mathbf{x})\}^2\right]$$
$$=\underbrace{\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]-h(\mathbf{x})\}^2}_{(\text{bias})^2}+\underbrace{\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D})-\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right]}_{\text{variance}}.$$

So far, we have considered a single input value $\mathbf{x}$. If we substitute this expansion back into the expected loss

$$\mathbb{E}_{\mathcal{D}}\left[\mathbb{E}_{(\mathbf{x},t)}[L]\right]=\mathbb{E}_{\mathcal{D}}\left[\int\{y(\mathbf{x};\mathcal{D})-h(\mathbf{x})\}^2p(\mathbf{x})\mathrm{d}\mathbf{x}+\iint\{h(\mathbf{x})-t\}^2p(\mathbf{x},t)\mathrm{d}\mathbf{x}\mathrm{d}t\right]$$
$$=\int\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D})-h(\mathbf{x})\}^2\right]p(\mathbf{x})\mathrm{d}\mathbf{x}+\iint\{h(\mathbf{x})-t\}^2p(\mathbf{x},t)\mathrm{d}\mathbf{x}\mathrm{d}t$$
$$=\int\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]-h(\mathbf{x})\}^2p(\mathbf{x})\mathrm{d}\mathbf{x}$$
$$+\int\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D})-\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right]p(\mathbf{x})\mathrm{d}\mathbf{x}+\iint\{h(\mathbf{x})-t\}^2p(\mathbf{x},t)\mathrm{d}\mathbf{x}\mathrm{d}t.$$

The Bias-Variance Decomposition

We obtain the following decomposition of the expected squared loss

$$\text{expected loss} = (\text{bias})^2 + \text{variance} + \text{noise},$$

where

$$(\text{bias})^2 = \int \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 p(\mathbf{x})\mathrm{d}\mathbf{x},$$

$$\text{variance} = \int \mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right] p(\mathbf{x})\mathrm{d}\mathbf{x},$$

$$\text{noise} = \iint \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t)\mathrm{d}\mathbf{x}\mathrm{d}t.$$

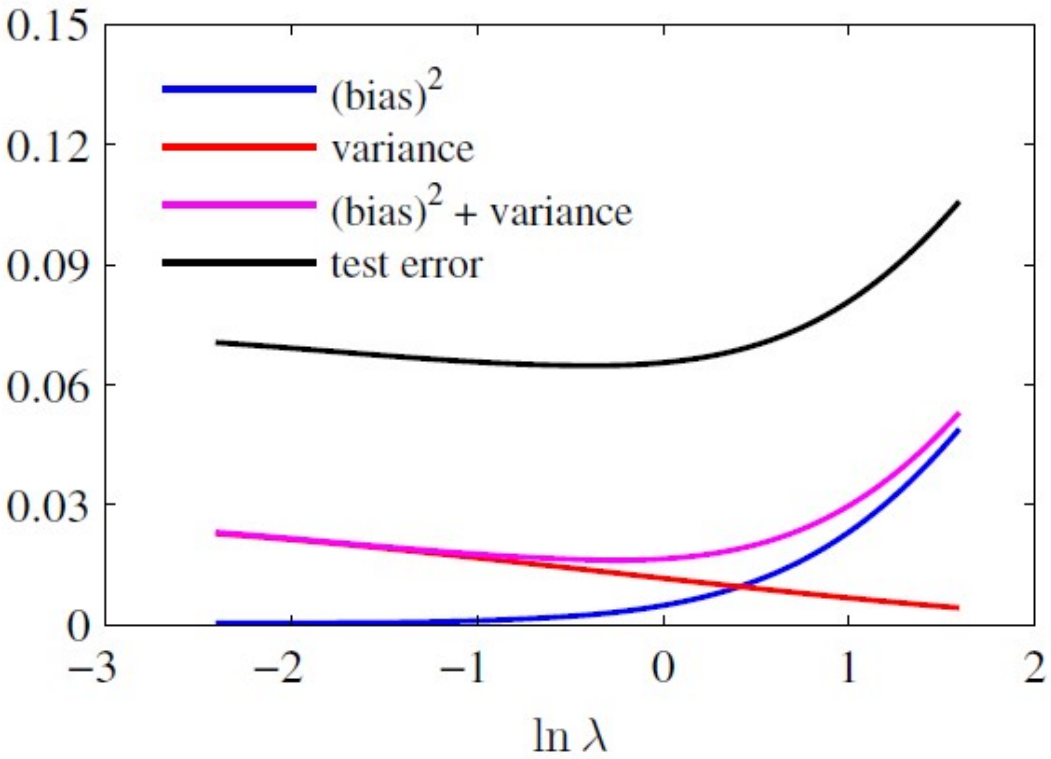
The Bias-Variance Decomposition

Given the regularized error

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\left(\mathbf{x}_n\right)\right\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q,$$

- large λ: restrict the model's freedom for fitting data, leading to **high bias and low variance**
- small λ: warrant more freedom to fit data, resulting in **low bias and high variance**

The Bias-Variance Decomposition



Plot of squared bias and variance, together with their sum, as well as the test error, for a linear regression model with the sinusoidal basis function and the quadratic regularizer. The minimum value of $(\text{bias})^2 + \text{variance}$ occurs around $\ln \lambda = 0.31$, which is close to the value that gives the minimum error on the test data.

Logistic Regression

Logistic Regression for Binary Classification

Given the feature vector of an example $\mathbf{x}$, binary classification aims to determine its class label from the label candidate set $\{\mathcal{C}_1, \mathcal{C}_2\}$.

From probabilistic view, the class label corresponds the label $\mathcal{C}_k$ that maximize the conditional probability $p(\mathcal{C}_k|\mathbf{x})$.

Suppose we can model class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$, as well as the class priors $p(\mathcal{C}_k)$, the posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$ can be computed through Bayes' theorem.

$$\begin{aligned} p(\mathcal{C}_1|\mathbf{x}) &= \frac{p(\mathbf{x}|\mathcal{C}_1)\, p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)\, p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)\, p(\mathcal{C}_2)} \\ &= \frac{1}{1 + \exp(-a)} = \sigma(a), \end{aligned}$$

where we have defined $a = \ln \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$ and $\sigma(a)$ is the *sigmoid* function defined by $\sigma(a) = \frac{1}{1+\exp(-a)}$.

Logistic Regression for Binary Classification

We can use a linear basis function model to parameterize the log of probability ratio $a = \ln \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$, by setting $a = \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x})$ so that

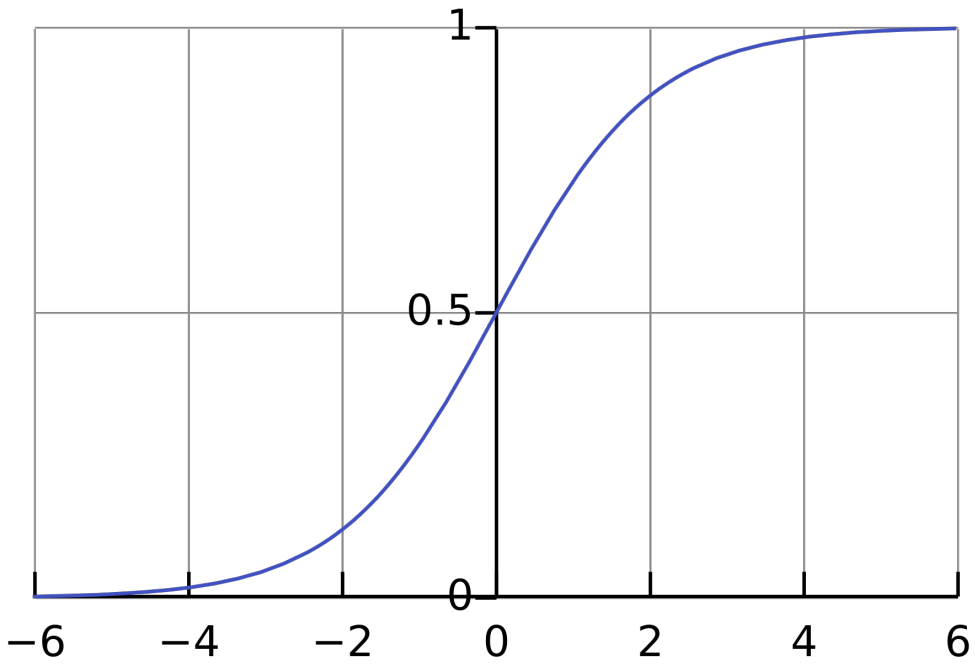
$$p(\mathcal{C}_1|\mathbf{x}) = \sigma(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x})), \text{ and } p(\mathcal{C}_2|\mathbf{x}) = 1 - p(\mathcal{C}_1|\mathbf{x}).$$

Now, we get the *logistic regression* model.

Next, we use maximum likelihood to determine the parameters of the logistic regression model. To do this, we shall make use of the derivative of the sigmoid function, which can conveniently be expressed in terms of the sigmoid function

$$\frac{d\sigma}{da} = \sigma(1 - \sigma).$$

Logistic Regression for Binary Classification



Sigmoid Function $\sigma(a)$

Logistic Regression for Binary Classification

For a data set $\{\mathbf{x}_n, t_n\}$, where $t_n \in \{0, 1\}$, with $n = 1, \cdots, N$, the likelihood function can be written as

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} \{y_n(\mathbf{w})\}^{t_n} \{1 - y_n(\mathbf{w})\}^{1-t_n},$$

where $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$ and $y_n(\mathbf{w}) = p(\mathcal{C}_1|\mathbf{x}_n) = \sigma(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n))$. We can define an error function by taking the negative logarithm of the likelihood, which gives the *cross-entropy* error function in the form

$$\begin{aligned} E(\mathbf{w}) &= -\ln p(\mathbf{t}|\mathbf{w}) \\ &= -\sum_{n=1}^{N} \{t_n \ln y_n(\mathbf{w}) + [1 - t_n] \ln [1 - y_n(\mathbf{w})]\}. \end{aligned}$$

To avoid overfitting, we can also add a regularization term to the *cross-entropy* error function.

Logistic Regression for Binary Classification

Taking the gradient of the error function with respect to $\mathbf{w}$, we obtain

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} \{y_n(\mathbf{w}) - t_n\} \phi_n(\mathbf{x}),$$

we have made use of the equation where we have made use of the equation $\frac{d\sigma}{da} = \sigma(1-\sigma)$. We see that the he contribution to the gradient from data point n is given by the 'error' $y_n(\mathbf{w}) - t_n$ between the target value and the prediction of the model, times the basis function vector $\phi_n(\mathbf{x})$.

Iterative Reweighted Least Squares

In the case of the linear regression models, whose sum-of-squares error function is a quadratic function of $\mathbf{w}$, we can obtain the closed-form optimal solution of $\mathbf{w}$ by setting the gradient to zero.

For logistic regression, there is no longer a closed-form solution, due to the nonlinearity of the logistic sigmoid function.

However, the departure from a quadratic form is not substantial. To be precise, the error function is convex, and hence has a unique minimum.

Furthermore, the error function can be minimized by an efficient iterative technique based on the Newton-Raphson iterative optimization scheme:

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \mathbf{H}(\mathbf{w}^{\text{old}})^{-1}\nabla E(\mathbf{w}^{\text{old}}),$$

where $\mathbf{H}(\mathbf{w}^{\text{old}})$ is the Hessian matrix at $\mathbf{w} = \mathbf{w}^{\text{old}}$, whose elements comprise the second derivatives of $E(\mathbf{w})$ with respect to the components of $\mathbf{w}$.

Iterative Reweighted Least Squares

The gradient and Hessian matrix of *logistic regression*'s *cross-entropy* error function are given by

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} \{y_n(\mathbf{w}) - t_n\} \phi(\mathbf{x}_n) = \mathbf{\Phi}^{\mathrm{T}} \{\mathbf{y}(\mathbf{w}) - \mathbf{t}\},$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n(\mathbf{w}) \{1 - y_n(\mathbf{w})\} \phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^{\mathrm{T}} = \mathbf{\Phi}^{\mathrm{T}}\mathbf{R}(\mathbf{w})\mathbf{\Phi},$$

where $\mathbf{y}(\mathbf{w}) = [y_1(\mathbf{w}), \cdots, y_1(\mathbf{w})]^{\mathrm{T}}$, $\mathbf{\Phi} = [\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_N)]^{\mathrm{T}}$ is an $N \times M$ matrix, and $\mathbf{R}(\mathbf{w})$ is an $N \times N$ diagonal matrix with entries

$$R_{nn}(\mathbf{w}) = y_n(\mathbf{w}) \{1 - y_n(\mathbf{w})\}.$$

Iterative Reweighted Least Squares

The Newton-Raphson update formula for the *logistic regression* model then becomes

$$\begin{aligned}\mathbf{w}^{\text{new}} &= \mathbf{w}^{\text{old}} - \left\{\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}(\mathbf{w}^{\text{old}})\mathbf{\Phi}\right\}^{-1}\mathbf{\Phi}^{\mathrm{T}}\left\{\mathbf{y}(\mathbf{w}^{\text{old}}) - \mathbf{t}\right\} \\ &= \left\{\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}(\mathbf{w}^{\text{old}})\mathbf{\Phi}\right\}^{-1}\left\{\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}(\mathbf{w}^{\text{old}})\mathbf{\Phi}\mathbf{w}^{\text{old}} - \mathbf{\Phi}^{\mathrm{T}}\left\{\mathbf{y}(\mathbf{w}^{\text{old}}) - \mathbf{t}\right\}\right\} \\ &= \left\{\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}(\mathbf{w}^{\text{old}})\mathbf{\Phi}\right\}^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}(\mathbf{w}^{\text{old}})\mathbf{z}(\mathbf{w}^{\text{old}}),\end{aligned}$$

taking the form of a set of normal equations for a weighted least-squares problem, where $\mathbf{z}(\mathbf{w}^{\text{old}})$ is an N-dimensional vector with elements

$$\mathbf{z} = \mathbf{\Phi}\mathbf{w}^{\text{old}} - \mathbf{R}(\mathbf{w}^{\text{old}})^{-1}\left\{\mathbf{y}(\mathbf{w}^{\text{old}}) - \mathbf{t}\right\}.$$

Logistic Regression for Multi-Class Classification

For multi-class classification with label candidate set $\{\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_K\}$, the posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$ are given by a softmax transformation of linear functions of the feature variables

$$p\left(\mathcal{C}_k|\mathbf{x}\right) = y_k = \frac{\exp\left(a_k\right)}{\sum_{j=1}^{K}\exp\left(a_j\right)},$$

where the 'activations' a_k are given by

$$a_k = \mathbf{w}_k^{\mathrm{T}}\phi(\mathbf{x}).$$

We consider the use of maximum likelihood to determine the parameters $\{\mathbf{w}_k\}$ of this model. To do this, we will require the derivatives of y_k with respect to all of the activations a_j. These are given by

$$\frac{\partial y_k}{\partial a_j} = y_k\left(I_{kj} - y_j\right),$$

where I_{kj} are the elements of the identity matrix, with $I_{kj} = 1$ for $k = j$ and $I_{kj} = 0$ for $k \neq j$.

Logistic Regression for Multi-Class Classification

Next we write down the likelihood function. This is most easily done using the 1-of-K coding scheme in which the target vector $\mathbf{t}_n$ for a feature vector $\phi(\mathbf{x}_n)$ belonging to class $\mathcal{C}_k$ is a binary vector with all elements zero except for element k, which equals one:

$$\mathbf{t}_n = (0, \cdots, 1, \cdots, 0)^{\mathrm{T}}.$$
$$\uparrow$$
$$k$$

The likelihood function is then given by

$$p(\mathbf{T}|\mathbf{w}_1, \ldots, \mathbf{w}_K) = \prod_{n=1}^{N} \prod_{k=1}^{K} p(\mathcal{C}_k|\mathbf{x}_n)^{t_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}^{t_{nk}},$$

where $y_{nk} = p(\mathcal{C}_k|\mathbf{x}_n)$, $\mathbf{T} = [\mathbf{t}_1, \cdots, \mathbf{t}_N]^{\mathrm{T}}$ is an $N \times M$ matrix target variables with elements t_{nk}.

Logistic Regression for Multi-Class Classification

Taking the negative logarithm then gives of the likelihood function then gives

$$E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = -\ln p(\mathbf{T}|\mathbf{w}_1, \ldots, \mathbf{w}_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_{nk},$$

which is known as the *cross-entropy* error function for the multiclass classification problem.

We now take the gradient of the error function with respect to one of the parameter vectors $\mathbf{w}_j$:

$$\nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = \sum_{n=1}^{N} (y_{nj} - t_{nj})\, \phi(\mathbf{x}_n).$$

Logistic Regression for Multi-Class Classification

We again appeal to the Newton-Raphson update to obtain the corresponding Iterative Reweighted Least Squares algorithm for the multiclass problem. This requires evaluation of the Hessian matrix that comprises blocks of size $M \times M$ in which block j, k is given by

$$\nabla_{\mathbf{w}_k} \nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = -\sum_{n=1}^{N} y_{nk} \left(I_{kj} - y_{nj} \right) \boldsymbol{\phi}(\mathbf{x}_n) \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}}.$$

As with the two-class problem, the Hessian matrix for the multiclass logistic regression model is positive definite and so the error function again has a unique minimum.

References

Christopher M. Bishop. **Pattern Recognition and Machine Learning**. New York: Springer; 2006 Aug 17.

- Linear Regression: Section 3.1-3.2
- Logistic Regression: Section 4.3

The End