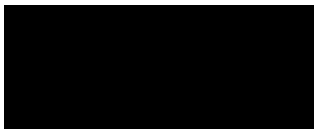


Linear regression

Mauricio A. Álvarez

Machine Learning and Adaptive Intelligence
The University of Sheffield



Contents

Review of vector/matrix notation and linear algebra

A regression model

Linear regression

Gradient descent

Stochastic Gradient Descent

Regularisation

Scalar and vectors

- q A **scalar** is just a numeric value like 0.9 or 18:7.
- q Scalars are usually denoted as lower case letters like x or a .
- q A **vector** is an ordered list of scalar values. Sometimes we refer to these scalar values of the vector as *attributes* or *entries* of the vector.
- q Vectors are usually denoted by bold lowercase letters like \mathbf{x} or \mathbf{y} .

Vectors

- q A vector can appear sometimes written as a row vector, e.g.

$$\mathbf{x} = [x_1; x_2; x_3; x_4; x_5]$$

Or as a column vector

$$\mathbf{x} = \begin{bmatrix} 2 \\ 6 \\ 6 \\ 6 \\ 4 \end{bmatrix} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix}$$

- q In this module, ALL vectors will be column vectors by default. So, when you see a vector, e.g. $\mathbf{x}; \mathbf{y}; \mathbf{z}$ always think this vector has a column-wise shape.

Matrices

- q A **matrix** is a rectangular array of scalars arranged in rows and columns.
- q Matrices are usually denoted by bold uppercase letters, e.g. **X** or **Y**.
- q The following matrix has three rows and two columns

$$\mathbf{X} = \begin{matrix} & \begin{matrix} 2 & 3 \end{matrix} \\ \begin{matrix} 4 & 5 \end{matrix} & \begin{matrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{matrix} \end{matrix}$$

- q The entries in the matrix above are of the form x_{ij} , where the first subindex i indicates the row of the element and the second subindex j indicates the column.

Matrix transpose

- q Let \mathbf{X} be a matrix with elements x_{ij} .
- q The transpose of a matrix \mathbf{X} is a new matrix $\mathbf{X}^>$ with elements x_{ji} .

$$\mathbf{X} = \begin{matrix} & \begin{matrix} 2 \\ 4 \end{matrix} & \begin{matrix} 5 \\ 7 \end{matrix} & \begin{matrix} 3 \\ 5 \end{matrix} \\ \begin{matrix} 4 \\ 3 \end{matrix} & \begin{matrix} 1 \\ 6 \end{matrix} & \begin{matrix} 6 \\ 9 \end{matrix} & \begin{matrix} 5 \\ 8 \end{matrix} \end{matrix}; \quad \mathbf{X}^> = \begin{matrix} & \begin{matrix} 4 \\ 5 \end{matrix} & \begin{matrix} 2 \\ 7 \end{matrix} & \begin{matrix} 3 \\ 1 \end{matrix} \\ \begin{matrix} 1 \\ 6 \end{matrix} & \begin{matrix} 6 \\ 9 \end{matrix} & \begin{matrix} 5 \\ 8 \end{matrix} & \begin{matrix} 5 \\ 8 \end{matrix} \end{matrix}$$

Matrix multiplication

- q Let **A** be a matrix with entries a_{ik} of dimensions $p \times q$.
- q Let **B** be a matrix with entries b_{kj} of dimensions $t \times s$.
- q Matrix multiplication of the form **AB** is only possible if $q = t$.
- q If this is the case, the matrix **C** = **AB** has dimensions $p \times s$ with entries

$$c_{ij} = \sum_k a_{ik} b_{kj}.$$

Transpose of a product

- q Let \mathbf{w} be a vector of dimensions d . 1. Let \mathbf{X} be a matrix with dimensions $n \times d$.

- q The transpose of the product \mathbf{Xw} , $(\mathbf{Xw})^T$ is

$$(\mathbf{Xw})^T = \mathbf{w}^T \mathbf{X}^T:$$

- q We can apply this result to a product of several matrices

$$\begin{aligned}(\mathbf{ABCD})^T &= ((\mathbf{AB})(\mathbf{CD}))^T \\&= (\mathbf{CD})^T (\mathbf{AB})^T \\&= \mathbf{D}^T \mathbf{C}^T \mathbf{B}^T \mathbf{A}^T:\end{aligned}$$

From a scalar operation to a vector operation

- q It is usually desirable to transform a scalar operation into a vector operation.
- q When coding scalar operations, we require making use of loops, which can be expensive.
- q In contrast, vector operations are handled efficiently by low-level routines already included in modules like numpy.

Example

Write the following scalar operation into a vector/matrix form

$$\sum_{i=1}^n \sum_{j=1}^d (y_i - x_{ij} w_j)^2 :$$

Answer (I)

q The sum above can be written as

$$\sum_{i=1}^n \left(y_i - \sum_{j=1}^d x_{ij} w_j \right)^2 = \left(y_1 - \sum_{j=1}^d x_{1j} w_j \right) \left(y_1 - \sum_{j=1}^d x_{1j} w_j \right) + \left(y_n - \sum_{j=1}^d x_{nj} w_j \right) \left(y_n - \sum_{j=1}^d x_{nj} w_j \right) :$$

q Let us define a vector \mathbf{v} of dimensions $n + 1$ with entries given as

$$\left(y_i - \sum_{j=1}^d x_{ij} w_j \right) :$$

Answer (II)

- q The product of vectors $\mathbf{v}^T \mathbf{v}$ gives the same result than the required sum,

$$\begin{aligned} \mathbf{v}^T \mathbf{v} &= \sum_{i=1}^n \left(y_i \sum_{j=1}^d x_{ij} w_j \right)^2 \\ &= \sum_{i=1}^n \sum_{j=1}^d (y_i x_{ij} w_j)^2 \end{aligned}$$

- q How do we express the elements in \mathbf{v} with vectors and matrices?

Answer (III)

- For a fixed i , $x_{i1}; \dots; x_{id}$ can be grouped into a vector $\mathbf{x}_i^>$.
- The internal sums in the entries of \mathbf{v} can then be written as

$$\sum_{j=1}^d x_{ij} w_j = \mathbf{x}_i^> \mathbf{w} = \begin{matrix} & \begin{matrix} 2 & 3 \\ w_1 & w_2 \\ \vdots & \vdots \\ w_d \end{matrix} \\ x_{i1} & x_{i2} & \dots & x_{id} \end{matrix}$$

- We can now write \mathbf{v} as

$$\mathbf{v} = \begin{matrix} \begin{matrix} 2 \\ 6 \\ 4 \end{matrix} y_1 & \mathbf{x}_1^> \mathbf{w} & \begin{matrix} 3 \\ 7 \\ 5 \end{matrix} \\ \vdots & & \vdots \\ y_n & \mathbf{x}_n^> \mathbf{w} & \end{matrix} = \begin{matrix} \begin{matrix} 2 \\ 6 \\ 4 \end{matrix} y_1 & \begin{matrix} 2 \\ 6 \\ 4 \end{matrix} \mathbf{x}_1^> \mathbf{w} & \begin{matrix} 3 \\ 7 \\ 5 \end{matrix} \\ \vdots & & \vdots \\ y_n & \mathbf{x}_n^> \mathbf{w} & \end{matrix} = \begin{matrix} \begin{matrix} 2 \\ 6 \\ 4 \end{matrix} y_1 & \begin{matrix} 2 \\ 6 \\ 4 \end{matrix} \mathbf{x}_1^> & \begin{matrix} 3 \\ 7 \\ 5 \end{matrix} \\ \vdots & & \vdots \\ y_n & \mathbf{x}_n^> & \end{matrix} \mathbf{w}$$

- We can group the scalars $y_1; \dots; y_n$ into a vector \mathbf{y} .
- We can group the row vectors $\mathbf{x}_1^>; \dots; \mathbf{x}_n^>$ into a matrix \mathbf{X} .

Answer (IV)

q It means that $\mathbf{v} = \mathbf{y} - \mathbf{X}\mathbf{w}$.

q Finally

$$\sum_{i=1}^n (y_i - \sum_{j=1}^d x_{ij} w_j)^2 = \mathbf{v}^T \mathbf{v} = (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w}) :$$

Two common types of products

- q **Inner product.** The inner product between two vectors results in a scalar.
- q Let \mathbf{x} and \mathbf{y} be vectors of dimension $m - 1$. The inner product is given as

$$\mathbf{x}^T \mathbf{y} = \sum_{i=1}^m x_i y_i$$

- q **Outer product.** The outer product between two vectors results in a matrix.
- q Let \mathbf{x} be a vector of dimension $m - 1$ and \mathbf{y} a vector of dimension $p - 1$. The outer product is given as

$$\mathbf{xy}^T = \begin{bmatrix} x_1 y_1 & x_1 y_2 & \dots & x_1 y_p \\ x_2 y_1 & x_2 y_2 & \dots & x_2 y_p \\ \vdots & \vdots & \ddots & \vdots \\ x_m y_1 & x_m y_2 & \dots & x_m y_p \end{bmatrix}$$

Differentiating a function in a vector/matrix form (I)

- q We will see cases in which a function $f(\mathbf{w})$ depends on some parameters grouped in a vector \mathbf{w} .
- q We would like to find the vector of parameters \mathbf{w} that maximise $f(\mathbf{w})$.
- q For example, suppose $f(\mathbf{w})$ is defined as

$$f(\mathbf{w}) = \sum_{i=1}^d w_i x_i$$

- q We can group the scalars $x_1; \dots; x_d$ into \mathbf{x} . Likewise for \mathbf{w} .
- q According to what we saw before, we can write $f(\mathbf{w})$ as $f(\mathbf{w}) = \mathbf{x}^T \mathbf{w}$.

Differentiating a function in a vector/matrix form (II)

- For a fixed \mathbf{x} , we are interested in computing the gradient of $f(\mathbf{w})$ with respect to \mathbf{w}

$$\frac{df(\mathbf{w})}{d\mathbf{w}} = \begin{matrix} \frac{\partial f(\mathbf{w})}{\partial w_1} & \frac{\partial f(\mathbf{w})}{\partial w_2} & \dots & \frac{\partial f(\mathbf{w})}{\partial w_d} \end{matrix} = \begin{matrix} x_1 & x_2 & \dots & x_d \end{matrix} = \mathbf{x}$$

- Some useful identities when differentiating with respect to a vector

$f(\mathbf{w})$	$\frac{df(\mathbf{w})}{d\mathbf{w}}$
$\mathbf{w}^T \mathbf{x}$	\mathbf{x}
$\mathbf{x}^T \mathbf{w}$	\mathbf{x}
$\mathbf{w}^T \mathbf{w}$	$2\mathbf{w}$
$\mathbf{w}^T \mathbf{Cw}$	$2\mathbf{Cw}$

Identity matrix and the inverse of a matrix

- q The identity matrix of size N is a square matrix with ones on the main diagonal and zeros elsewhere, e.g.,

$$\mathbf{I}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- q The inverse matrix of a matrix \mathbf{A} of dimensions $d \times d$, denoted as \mathbf{A}^{-1} , satisfies

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_d$$

Contents

Review of vector/matrix notation and linear algebra

A regression model

Linear regression

Gradient descent

Stochastic Gradient Descent

Regularisation

Olympic 100m Data

Image from Wikimedia Commons <http://bit.ly/191adDC>.

Dataset

Model

- q We will use a linear model $f(x; \mathbf{w})$ to predict y , where y is the time in seconds and x the year of the competition.

- q The linear model is given as

$$f(x; \mathbf{w}) = w_0 + w_1 x;$$

where w_0 is the intercept and w_1 is the slope.

- q We use \mathbf{w} to refer both to w_0 and w_1 .

Data and model

Contents

Review of vector/matrix notation and linear algebra

A regression model

Linear regression

Gradient descent

Stochastic Gradient Descent

Regularisation

Linear model

- q A simple model for regression consists in using a linear combination of the attributes to predict the output

$$f(\mathbf{x}; \mathbf{w}) = w_0 + w_1 x_1 + \dots + w_D x_D;$$

where $w_0; w_1; \dots; w_D$ are the parameters of the regression model.

- q The term w_0 is the bias term or intercept, e.g. $f(\mathbf{0}; \mathbf{w}) = w_0$.
- q The expression above can be written in a vectorial form

$$f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^T \mathbf{x};$$

where we have defined $\mathbf{w} = [w_0; w_1; \dots; w_D]^T$ and $\mathbf{x} = [1; x_1; \dots; x_D]^T$.

- q Notice that $x_0 = 1$.

Parenthesis: Gaussian pdf

- q The Gaussian pdf has the form

$$p(y) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left(-\frac{(y - \mu)^2}{2\sigma^2} \right) ;$$

- q A Gaussian pdf requires two parameters μ and σ^2 , the mean and the variance of the RV Y .
- q We denote the Gaussian pdf as $p(y; \mu, \sigma^2) = N(y; \mu, \sigma^2)$ or $y \sim N(\mu, \sigma^2)$.

Parenthesis: Gaussian pdf

The mean of the three Gaussians is $\mu = 2$ and the variances are $\sigma^2 = 0.5$ (solid), and $\sigma^2 = 2$ (dashed).

Gaussian regression model (I)

- q We use a Gaussian regression model to relate the inputs and outputs

$$y = f(\mathbf{x}; \mathbf{w}) + \epsilon;$$

where $\epsilon \sim N(0; \sigma^2)$.

- q It assumes that each output y_i that we observe can be explained as the prediction of an underlying model, $f(\mathbf{x}_i; \mathbf{w})$ plus a noise term ϵ_i .
- q For a fixed \mathbf{x} and a fixed \mathbf{w} , $f(\mathbf{x}; \mathbf{w})$ is a constant, then

$$y = \text{constant} + \epsilon;$$

where ϵ is a continuous RV.

- q What is the pdf for y ? (we are adding a constant to a Gaussian RV)
 - $Efyg = Ef\text{constant} + g = \text{constant}$
 - $\text{var}fyg = \text{var}f\text{constant}g + \text{var}f g = \sigma^2$.

Gaussian regression model (II)

- q This means that

$$y \sim N(\text{constant}; \sigma^2);$$

where we said constant was $f(\mathbf{x}; \mathbf{w})$, this is,

$$y \sim N(f(\mathbf{x}; \mathbf{w}); \sigma^2);$$

- q Because we assumed that \mathbf{x} and \mathbf{w} are given, we can also write

$$p(y|\mathbf{x}; \mathbf{w}; \sigma^2) = N(y|f(\mathbf{x}; \mathbf{w}); \sigma^2);$$

- q If we knew the value for \mathbf{w} , once we have a new \mathbf{x} , we can predict the output as $f(\mathbf{x}; \mathbf{w})$.
- q σ^2 tells us the noise variance.

Gaussian regression model (III)

How do we estimate \mathbf{w} ? (I)

- q We start with a training dataset $(\mathbf{x}_1; y_1); \dots; (\mathbf{x}_N; y_N)$.
- q We assume that the random variables $Y_1; \dots; Y_N$ are *independent*,

$$p(y_1; \dots; y_N | \mathbf{x}_1; \dots; \mathbf{x}_N) = p(y_1 | \mathbf{x}_1) \prod_{n=1}^N p(y_n | \mathbf{x}_n):$$

- q We also assume that the RVs $Y_1; \dots; Y_N$ follow an *identical* distribution, Gaussian in this case

$$p(y_n | \mathbf{x}_n; \mathbf{w}; \sigma^2) = N(y_n | f(\mathbf{x}_n; \mathbf{w}); \sigma^2) = N(y_n | \mathbf{w}^T \mathbf{x}_n; \sigma^2):$$

- q Both assumptions go by the name of the *iid* assumption, *independent and identically distributed*.

How do we estimate \mathbf{w} ? (II)

- Putting both assumptions together, we get

$$p(\mathbf{y}|\mathbf{X};\mathbf{w};^2) = \prod_{n=1}^N p(y_n|\mathbf{x}_n;\mathbf{w};^2) = \prod_{n=1}^N N(y_n|\mathbf{w}^T \mathbf{x}_n;^2);$$

where $\mathbf{y} = [y_1; \dots; y_N]^T \in \mathbb{R}^N$ and $\mathbf{X} = [\mathbf{x}_1; \dots; \mathbf{x}_N]^T \in \mathbb{R}^{N \times (D+1)}$.

- The expression above can then be written as

$$\begin{aligned} p(\mathbf{y}|\mathbf{X};\mathbf{w};^2) &= \prod_{n=1}^N N(y_n|\mathbf{w}^T \mathbf{x}_n;^2); \\ &= \prod_{n=1}^N \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y_n - \mathbf{w}^T \mathbf{x}_n)^2}{2}\right); \\ &= \frac{1}{(2\pi)^{\frac{N}{2}}} \exp\left(-\frac{1}{2} \sum_{n=1}^N (y_n - \mathbf{w}^T \mathbf{x}_n)^2\right); \end{aligned}$$

How do we estimate \mathbf{w} ? (III)

- q When we look at a Gaussian pdf, like

$$p(y) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y - \mu)^2}{2\sigma^2}\right);$$

we assume that both μ and σ^2 are given. In this case, the pdf follows all the properties we reviewed before.

- q The same is true for

$$p(\mathbf{y}|\mathbf{X}; \mathbf{w}; \sigma^2) = \prod_{n=1}^N p(y_n|\mathbf{x}_n; \mathbf{w}; \sigma^2) = \prod_{n=1}^N \mathcal{N}(y_n|\mathbf{w}^T \mathbf{x}_n; \sigma^2):$$

- q Given $\mathbf{w}^T \mathbf{x}_n$ and σ^2 , then each $p(y_n|\mathbf{x}_n; \mathbf{w}; \sigma^2)$ is a pdf.
- q A different approach would be to say: I have some data for $\{y_n\}_{n=1}^N$ and $\{\mathbf{x}_n\}_{n=1}^N$ but
 - "I don't know what is \mathbf{w}^T (therefore I don't know what is $\mathbf{w}^T \mathbf{x}_n$)"
 - "I don't know what is σ^2 ".

How do we estimate \mathbf{w} ? (IV)

- q With y_n and \mathbf{x}_n given but with unknown values for \mathbf{w} and σ^2 , each $p(y_n|\mathbf{x}_n; \mathbf{w}; \sigma^2)$ is not a pdf anymore.
- q In that case, the function

$$p(\mathbf{y}|\mathbf{X}; \mathbf{w}; \sigma^2) = \prod_{n=1}^N N(y_n | \mathbf{w}^T \mathbf{x}_n; \sigma^2);$$

receives the name of a *likelihood function*.

- q We can think of a likelihood function as a function of the parameters \mathbf{w} and σ^2 ,

$$g(\mathbf{w}; \sigma^2) = p(\mathbf{y}|\mathbf{X}; \mathbf{w}; \sigma^2);$$

- q And subsequently, we can use *multivariate calculus* to find the values of $\mathbf{w}; \sigma^2$ that maximise $g(\mathbf{w}; \sigma^2)$.
- q In statistics, this is known as the *maximum-likelihood* (ML) criterion to estimate parameters.

How do we estimate \mathbf{w} ? (V)

- q Given $\mathbf{y}; \mathbf{X}$, we use the ML criterion to find the parameters \mathbf{w} and σ^2 that maximise

$$p(\mathbf{y}|\mathbf{X}; \mathbf{w}; \sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp \left(-\frac{1}{2\sigma^2} \sum_{n=1}^N (y_n - \mathbf{w}^T \mathbf{x}_n)^2 \right);$$

- q In practice, we prefer to maximise the log of the likelihood $p(\mathbf{y}|\mathbf{X}; \mathbf{w}; \sigma^2)$,

$$\begin{aligned} LL(\mathbf{w}; \sigma^2) &= \log p(\mathbf{y}|\mathbf{X}; \mathbf{w}; \sigma^2) \\ &= \frac{N}{2} \log(2\pi) - \frac{N}{2} \log \sigma^2 - \frac{1}{2\sigma^2} \sum_{n=1}^N (y_n - \mathbf{w}^T \mathbf{x}_n)^2; \end{aligned}$$

- q **Consistency of the ML criterion** If data was really generated according to the probability we specified, the correct parameters will be recovered in the limit as $N \rightarrow \infty$.

Connection with the sum of squared errors

- q If we multiply $LL(\mathbf{w}; \mathbf{y}, \mathbf{X})$ by minus one, we get

$$E(\mathbf{w}; \mathbf{y}, \mathbf{X}) = -\log p(\mathbf{y}|\mathbf{X}; \mathbf{w}) \propto \sum_{n=1}^N (y_n - \mathbf{w}^T \mathbf{x}_n)^2$$

- q The ML criterion for this model has a close connection with the sum-of-squared errors used in non-probabilistic formulations of linear regression.
- q Maximising the log-likelihood function is equivalent to minimising the sum-of-squares errors.
- q Notice that the log is a monotonic function, meaning that if we find \mathbf{w}^* that maximise $g(\mathbf{w})$, those will also maximise $\log(g(\mathbf{w}))$.

Normal equation (I)

- q Let us find an estimate for \mathbf{w} .
- q From what we saw before,

$$LL(\mathbf{w}; \sigma^2) = \frac{N}{2} \log(2\sigma^2) + \frac{N}{2} \log \sigma^2 + \frac{1}{2\sigma^2} \sum_{n=1}^N (y_n - \mathbf{w}^T \mathbf{x}_n)^2.$$

- q Using what we reviewed in the section on vector/matrix notation, it can be shown that this expression can be written in a vectorial form as

$$LL(\mathbf{w}; \sigma^2) = \frac{N}{2} \log(2\sigma^2) + \frac{N}{2} \log \sigma^2 + \frac{1}{2\sigma^2} (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

- q Let us focus on the term $(\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$,

$$(\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w}) = \mathbf{y}^T \mathbf{y} - \mathbf{w}^T \mathbf{X}^T \mathbf{y} - \mathbf{y}^T \mathbf{X}\mathbf{w} + \mathbf{w}^T \mathbf{X}^T \mathbf{X}\mathbf{w}$$

Normal equation (II)

- q We can find the \mathbf{w} that maximises $LL(\mathbf{w}; \mathbf{y})$ by taking the gradient $\frac{dLL(\mathbf{w}; \mathbf{y})}{d\mathbf{w}}$, equating to zero and solving for \mathbf{w} .

- q Taking the gradient of each term in $LL(\mathbf{w}; \mathbf{y})$ wrt \mathbf{w} , we get

$$\frac{d}{d\mathbf{w}} \left[\frac{N}{2} \log(2\pi) \right] = 0; \quad \frac{d}{d\mathbf{w}} \left[\frac{N}{2} \log \sigma^2 \right] = 0; \quad \frac{d}{d\mathbf{w}} \left[\frac{1}{2\sigma^2} \mathbf{y}^T \mathbf{y} \right] = 0;$$

$$\frac{d}{d\mathbf{w}} \left[\frac{1}{2\sigma^2} \mathbf{w}^T \mathbf{X}^T \mathbf{y} \right] = \frac{1}{2\sigma^2} \mathbf{X}^T \mathbf{y};$$

$$\frac{d}{d\mathbf{w}} \left[\frac{1}{2\sigma^2} \mathbf{y}^T \mathbf{X} \mathbf{w} \right] = \frac{1}{2\sigma^2} \mathbf{X}^T \mathbf{y}$$

$$\frac{d}{d\mathbf{w}} \left[\frac{1}{2\sigma^2} \mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w} \right] = \frac{1}{2\sigma^2} 2 \mathbf{X}^T \mathbf{X} \mathbf{w}$$

Normal equation (III)

- Putting these terms together, we get

$$\begin{aligned}\frac{d}{d\mathbf{w}} LL(\mathbf{w}; \mathbf{y}) &= \frac{1}{2} \mathbf{X}^T \mathbf{y} + \frac{1}{2} \mathbf{X}^T \mathbf{y} - \frac{1}{2} 2 \mathbf{X}^T \mathbf{X} \mathbf{w} \\ &= \frac{1}{2} \mathbf{X}^T \mathbf{y} - \mathbf{X}^T \mathbf{X} \mathbf{w}\end{aligned}$$

- Now, equating to zero and solving for \mathbf{w} , we get

$$\begin{aligned}\frac{1}{2} \mathbf{X}^T \mathbf{y} - \mathbf{X}^T \mathbf{X} \mathbf{w} &= \mathbf{0} \\ \mathbf{X}^T \mathbf{X} \mathbf{w} &= \mathbf{X}^T \mathbf{y} \\ \mathbf{w} &= (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}\end{aligned}$$

- The expression for \mathbf{w} is known as the *normal equation*.
- The solution for \mathbf{w} exists if we can compute $(\mathbf{X}^T \mathbf{X})^{-1}$.
- The inverse can be computed as long as $\mathbf{X}^T \mathbf{X}$ is non-singular (e.g. determinant different from zero, or has full-rank).

Solving for β^2

- Following a similar procedure, it can be shown that the ML solution for β^2 is given as

$$\beta^2 = \frac{1}{N}(\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w});$$

Basis functions

- q The model that is linear in \mathbf{x} only allows linear relationships between \mathbf{x} and y .
- q We can extend the model to describe non-linear relationships between the inputs and the output by using basis functions, non-linear mappings from inputs to outputs.
- q However, we keep the linear relationship of y wrt \mathbf{w} for tractability.
- q The predictive model follows as $f(\mathbf{x}; \mathbf{w})$

$$f(\mathbf{x}; \mathbf{w}) = w_0 + \sum_{i=1}^M w_i \phi_i(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x});$$

where $\phi_i(\mathbf{x})$ are basis functions and we have $M + 1$ parameters for the vector \mathbf{w} and $\boldsymbol{\phi}(\mathbf{x}) = [1; \phi_1(\mathbf{x}); \dots; \phi_M(\mathbf{x})]^T$.

Examples of basis functions

Polynomial: $\phi_i(x) = x^i$. n

Exponential: $\phi_i(x) = \exp\left(\frac{(x - \mu_i)^2}{2\sigma_i^2}\right)$

Sigmoidal: $\phi_i(x) = \frac{x - \mu_i}{\sigma_i}$; $\sigma_i = 1 / (1 + \exp(-a_i))$.

Transforming the input using the basis functions

- q As an example, let us use polynomial basis functions to predict y , the time in seconds in the 100 mt Olympics competition.
- q For each x (year of the competition), we now compute the vector of polynomial basis functions

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ \vdots \\ x^M \end{bmatrix}$$

- q We have converted the unidimensional input feature x into a higher dimensional feature representation $\phi(x) \in \mathbb{R}^{M+1}$.

Normal equations with a design matrix

- Given \mathbf{X} , we first compute a new design matrix

$$= \begin{pmatrix} 1 & \mathbf{x}_1^T \\ 1 & \mathbf{x}_2^T \\ \vdots & \vdots \\ 1 & \mathbf{x}_N^T \end{pmatrix} = \begin{pmatrix} 1 & x_{01} & x_{11} \\ 1 & x_{02} & x_{12} \\ \vdots & \vdots & \vdots \\ 1 & x_{0N} & x_{1N} \end{pmatrix} = \begin{pmatrix} 1 & x_{01} & x_{11} \\ 1 & x_{02} & x_{12} \\ \vdots & \vdots & \vdots \\ 1 & x_{0N} & x_{1N} \end{pmatrix}$$

- We now can use $(\mathbf{y}; 1)$ and write the Gaussian linear regression problem

$$p(\mathbf{y}|\mathbf{X}; \mathbf{w}; \sigma^2) = \prod_{n=1}^N \mathcal{N}(y_n | \mathbf{w}^T \mathbf{x}_n; \sigma^2);$$

where $\mathbf{x}_n = \begin{pmatrix} 1 \\ \mathbf{x}_n \end{pmatrix}$.

- Using the ML criterion, we arrive to the following normal equation

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y};$$

Olympic 100-mt data with $M = 5$

Alternative to find \mathbf{w}

- q For solving the normal equation, we need to invert $\mathbf{X}^T \mathbf{X}$.
- q This inversion has a computational complexity between $O((D + 1)^{2:4})$ to $O((D + 1)^3)$ (depending on the implementation).
- q The normal equation is linear regarding the number of instances in the training data, $O(N)$.
- q It can handle a large training set as long as it fits in memory.
- q Alternatively, we can use iterative optimisation in cases with a large number of features and too many instances to fit in memory.

Contents

Review of vector/matrix notation and linear algebra

A regression model

Linear regression

Gradient descent

Stochastic Gradient Descent

Regularisation

General problem

- q We are given a function $h(\mathbf{w})$, where $\mathbf{w} \in \mathbb{R}^p$.
- q Aim: to find a value for \mathbf{w} that minimises $h(\mathbf{w})$.
- q Use an iterative procedure

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha \mathbf{d}_k;$$

where \mathbf{d}_k is known as the search direction and it is such that

$$h(\mathbf{w}_{k+1}) < h(\mathbf{w}_k):$$

- q The parameter α is known as the **step size** or **learning rate**.

Gradient descent

- q Perhaps, the simplest algorithm for unconstrained optimisation.
- q It assumes that $\mathbf{d}_k = -\mathbf{g}_k$, where $\mathbf{g}_k = \mathbf{g}(\mathbf{w}_k)$.
- q Also known as **steepest descent**.
- q It can be written like

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mathbf{g}_k:$$

Step size

- q The main issue in gradient descent is how to set the step size.
- q If it is too small, convergence will be very slow. If it is too large, the method can fail to converge at all.

(a)

(b)

Figure: The function to optimise is $h(w_1; w_2) = 0.5(w_1^2 - w_2)^2 + 0.5(w_1 - 1)^2$. The minimum is at $(1; 1)$. In (a) $\alpha = 0.1$. In (b) $\alpha = 0.6$.

Alternatives to choose the step size

- q Line search methods (there are different alternatives).
- q Line search methods may use search directions other than the steepest descent direction.
- q Conjugate gradient (method of choice for quadratic objectives $g(\mathbf{w}) = \mathbf{w}^T \mathbf{A} \mathbf{w}$).
- q Use a *Newton* search direction.

Gradient descent for linear regression (I)

- q For simplicity, let us assume that the objective function $h(\mathbf{w})$ corresponds to the mean squared error

$$E(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^N (y_n - \mathbf{w}^T \mathbf{x}_n)^2:$$

- q We could also minimise the negative $LL(\mathbf{w})$ instead.
- q We write the update equation as

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \frac{d}{d\mathbf{w}} E(\mathbf{w}) \Big|_{\mathbf{w}=\mathbf{w}_k}:$$

Gradient descent for linear regression (II)

- Computing the gradient for $E(\mathbf{w})$, we get

$$\frac{d}{d\mathbf{w}} E(\mathbf{w}) = \frac{2}{N} \sum_{n=1}^N \mathbf{w}^T \mathbf{x}_n - y_n \quad \mathbf{x}_n = \frac{2}{N} \mathbf{X}^T (\mathbf{X} \mathbf{w} - \mathbf{y}) :$$

- The update equation follows as

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \frac{2}{N} \mathbf{X}^T (\mathbf{X} \mathbf{w}_k - \mathbf{y}) :$$

- The computation of the gradient involves using the whole dataset $(\mathbf{X}; \mathbf{y})$ at every step.
- For this reason, this algorithm is known as *batch gradient descent*.

Gradient descent and feature scaling

- q Always normalise the features if using gradient descent.
- q Gradient descent converges faster if all features have a similar scale.
- q If the attributes are in very different scales, it may take a long time to converge.

Contents

Review of vector/matrix notation and linear algebra

A regression model

Linear regression

Gradient descent

Stochastic Gradient Descent

Regularisation

Online learning and large datasets

- q Traditionally in machine learning, the gradient \mathbf{g}_k is computed using the whole dataset $D = \{ \mathbf{x}_n, y_n \}_{n=1}^N$.
- q There are settings, though, where only a subset of the data can be used.
- q **Online learning:** the instances (\mathbf{x}_n, y_n) appear one at a time.
- q **Large datasets:** computing the exact value for \mathbf{g}_k would be expensive, if not impossible.

Stochastic gradient descent (I)

- q In stochastic gradient descent (SGD), the gradient \mathbf{g}_k is computed using a subset of the instances available.
- q The word stochastic refers to the fact that the value for \mathbf{g}_k will depend on the subset of the instances chosen for computation.

Stochastic gradient descent (II)

- q In the stochastic setting, a better estimate can be found if the gradient is computed using

$$\mathbf{g}_k = \frac{1}{|S|} \sum_{i \in S} \mathbf{g}_{k,i},$$

where $S \subseteq D$, $|S|$ is the cardinality of S , and $\mathbf{g}_{k,i}$ is the gradient at iteration k computed using the instance (\mathbf{x}_i, y_i) .

- q This setting is called *mini-batch gradient descent*.

Step size in SGD

- Choosing the value of η is particularly important in SGD since there is no easy way to compute it.
- Usually the value of η will depend on the iteration k , η_k .
- It should follow the **Robbins-Monro** conditions

$$\sum_{k=1}^{\infty} \eta_k = \infty ; \quad \sum_{k=1}^{\infty} \eta_k^2 < \infty ;$$

- Various formulas for η_k can be used

$$\eta_k = \frac{1}{k} ; \quad \eta_k = \frac{1}{(\eta_0 + k)} ;$$

where η_0 slows down early iterations and $\eta_0 \in (0.5; 1]$.

Contents

Review of vector/matrix notation and linear algebra

A regression model

Linear regression

Gradient descent

Stochastic Gradient Descent

Regularisation

What is regularisation?

- q It refers to a technique used for preventing overfitting in a predictive model.
- q It consists in adding a term (a regulariser) to the objective function that encourages simpler solutions.
- q With regularisation, the objective function for linear regression would be

$$h(\mathbf{w}) = E(\mathbf{w}) + R(\mathbf{w});$$

where $R(\mathbf{w})$ is the regularisation term and λ the regularisation parameter.

- q In the expression for $h(\mathbf{w})$, we can use the negative $LL(\mathbf{w})$ instead of $E(\mathbf{w})$.
- q If $\lambda = 0$, we get $h(\mathbf{w}) = E(\mathbf{w})$.

Different types of regularisation

- q The objective function for linear regression would be

$$h(\mathbf{w}) = E(\mathbf{w}) + R(\mathbf{w});$$

where $R(\mathbf{w})$ follows as

$$R(\mathbf{w}) = \lambda \|\mathbf{w}\|_1 + (1 - \lambda) \frac{1}{2} \|\mathbf{w}\|_2^2;$$

where $\|\mathbf{w}\|_1 = \sum_{m=1}^p |w_m|$, and $\|\mathbf{w}\|_2^2 = \sum_{m=1}^p w_m^2$.

- q If $\lambda = 1$, we get ℓ_1 regularisation.
- q If $\lambda = 0$, we get ℓ_2 regularisation.
- q If $0 < \lambda < 1$, we get the elastic net regularisation.

Ridge regression or ℓ_2 regularisation

- q In ridge regression, $\lambda = 0$,

$$h(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^N (y_n - \mathbf{w}^T \mathbf{x}_n)^2 + \frac{\lambda}{2} \mathbf{w}^T \mathbf{w};$$

- q It can be shown that an optimal solution for \mathbf{w} is given as

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X} + \frac{\lambda}{2} \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y};$$

- q Notice that we can also use iterative procedure for optimising $h(\mathbf{w})$ either through batch gradient decent, SGD or mini-batch SGD.