

# Linear regression

Mauricio A. Álvarez

Machine Learning and Adaptive Intelligence  
The University of Sheffield



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

- A **scalar** is just a numeric value like 0.9 or  $-18.7$ .
- Scalars are usually denoted as lower case letters like  $x$  or  $a$ .
- 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.
- Vectors are usually denoted by bold lowercase letters like  $\mathbf{x}$  or  $\mathbf{y}$ .

# Vectors

- 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

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

- 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

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

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{bmatrix}$$

- 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

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

$$\mathbf{X} = \begin{bmatrix} 4.1 & -5.6 \\ -2.6 & 7.9 \\ 3.5 & 1.8 \end{bmatrix}, \quad \mathbf{X}^\top = \begin{bmatrix} 4.1 & -2.6 & 3.5 \\ -5.6 & 7.9 & 1.8 \end{bmatrix}$$

# Matrix multiplication

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

- Let  $\mathbf{w}$  be a vector of dimensions  $d \times 1$ . Let  $\mathbf{X}$  be a matrix with dimensions  $n \times d$ .
- The transpose of the product  $\mathbf{Xw}$ ,  $(\mathbf{Xw})^\top$  is

$$(\mathbf{Xw})^\top = \mathbf{w}^\top \mathbf{X}^\top.$$

*How to prove that? (Linear Algebra)*

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

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



# From a scalar operation to a vector operation

- It is usually desirable to transform a scalar operation into a vector operation.
- When coding scalar operations, we require making use of loops, which can be expensive.
- 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

$$\begin{aligned}
 i=1: & \left( y_1 - \sum_{j=1}^d x_{1j} w_j \right)^2 \\
 & \vdots \\
 & \left( y_n - \sum_{j=1}^d x_{nj} w_j \right)^2
 \end{aligned}
 \rightarrow
 \begin{aligned}
 & \sum_{i=1}^n (y_i - \sum_{j=1}^d x_{ij} w_j)^2 \\
 & \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} - \begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,d} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{pmatrix}
 \end{aligned}$$

$\begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,d} \end{pmatrix} = \begin{matrix} 1 \times d \\ d \times d \end{matrix}$

$y_1 - x_1 w$   
 $\rightarrow y_2 - x_2 w$   
 $\vdots$   
 $y_n - x_n w$

$\rightarrow Y - wX$

$\rightarrow \sum_{j=1}^d x_{nj} w_j$

# Answer (I)

- The sum above can be written as

$$\begin{aligned}\sum_{i=1}^n (y_i - \sum_{j=1}^d x_{ij} w_j)^2 &= (y_1 - \sum_{j=1}^d x_{1j} w_j)(y_1 - \sum_{j=1}^d x_{1j} w_j) \\ &\quad + \dots \\ &\quad + (y_n - \sum_{j=1}^d x_{nj} w_j)(y_n - \sum_{j=1}^d x_{nj} w_j).\end{aligned}$$

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

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

## Answer (II)

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

$$\begin{aligned}\mathbf{v}^\top \mathbf{v} &= \left[ (y_1 - \sum_{j=1}^d x_{1j} w_j) \quad \cdots \quad (y_n - \sum_{j=1}^d x_{nj} w_j) \right] \begin{bmatrix} (y_1 - \sum_{j=1}^d x_{1j} w_j) \\ \vdots \\ (y_n - \sum_{j=1}^d x_{nj} w_j) \end{bmatrix} \\ &= \sum_{i=1}^n (y_i - \sum_{j=1}^d x_{ij} w_j)^2.\end{aligned}$$

- 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  $\mathbf{x}_i^\top$ .
- The internal sums in the entries of  $\mathbf{v}$  can then be written as

$$\sum_{j=1}^d x_{ij} w_j = \mathbf{x}_i^\top \mathbf{w} = \begin{bmatrix} x_{i1} & x_{i2} & \cdots & x_{id} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}$$

- We can now write  $\mathbf{v}$  as

$$\mathbf{v} = \begin{bmatrix} y_1 - \mathbf{x}_1^\top \mathbf{w} \\ \vdots \\ y_n - \mathbf{x}_n^\top \mathbf{w} \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} \mathbf{x}_1^\top \mathbf{w} \\ \vdots \\ \mathbf{x}_n^\top \mathbf{w} \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix} \mathbf{w}$$

*Y - wX.*

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

## Answer (IV)

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

□ Finally

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

# Two common types of products

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

*inner*

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

*$m \times 1 \cdot m \times 1 \rightarrow 1 \times 1$*

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

*outer*

$$\mathbf{x} \mathbf{y}^T = \begin{bmatrix} x_1 y_1 & \cdots & x_1 y_p \\ x_2 y_1 & \cdots & x_2 y_p \\ \vdots & \vdots & \vdots \\ x_m y_1 & \cdots & x_m y_p \end{bmatrix}$$

*$m \times 1 \cdot 1 \times p = m \times p$*

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

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

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

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



# 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  $\mathbf{w}$

$$\frac{df(\mathbf{w})}{d\mathbf{w}} = \begin{bmatrix} \frac{\partial f(\mathbf{w})}{\partial w_1} \\ \vdots \\ \frac{\partial f(\mathbf{w})}{\partial w_d} \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} = \mathbf{x}.$$

- Some useful identities when differentiating with respect to a vector

$f(\mathbf{w})$	$\frac{df(\mathbf{w})}{d\mathbf{w}}$
$\mathbf{w}^\top \mathbf{x}$	$\mathbf{x}$
$\mathbf{x}^\top \mathbf{w}$	$\mathbf{x}$
$\mathbf{w}^\top \mathbf{w}$	$2\mathbf{w}$
$\mathbf{w}^\top \mathbf{C}\mathbf{w}$	$2\mathbf{C}\mathbf{w}$

# Identity matrix and the inverse of a matrix

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

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

- 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

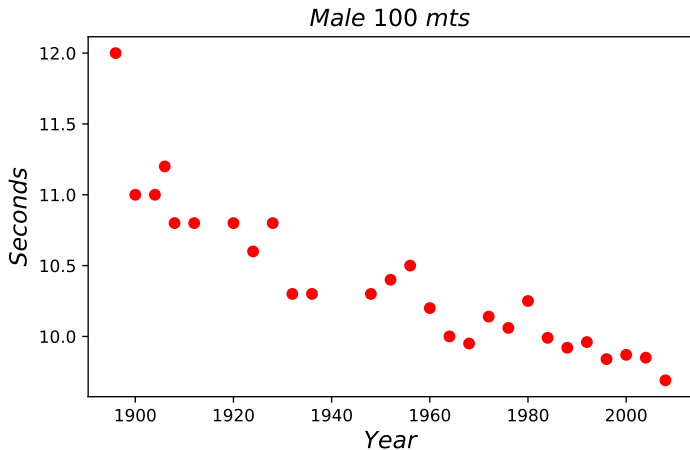
Regularisation

# Olympic 100m Data



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

# Dataset



# Model

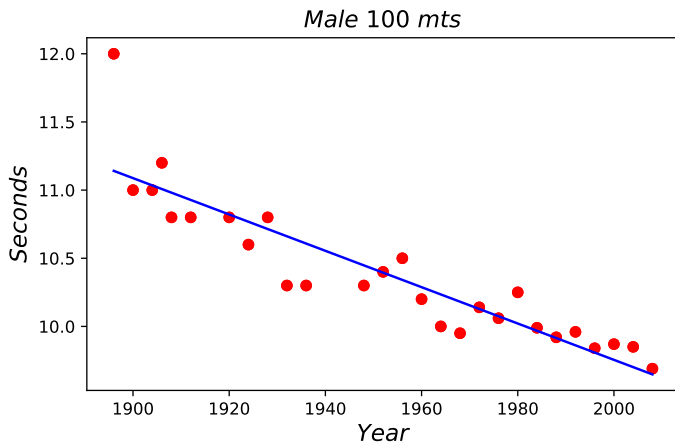
- We will use a linear model  $f(x, \mathbf{w})$  to predict  $y$ , where  $y$  is the time in seconds and  $x$  the year of the competition.
- The linear model is given as

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

where  $w_0$  is the intercept and  $w_1$  is the slope.

- 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

- 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.

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

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^\top \mathbf{x}.$$

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

- Notice that  $x_0 = 1$ .

# Parenthesis: Gaussian pdf

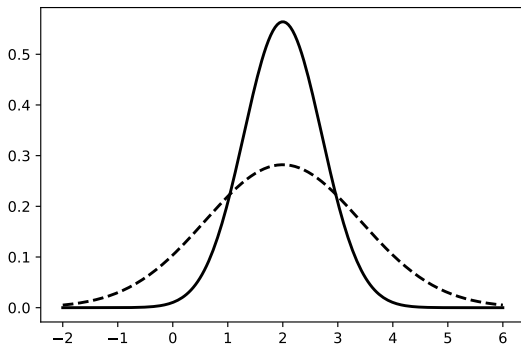
- The Gaussian pdf has the form

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

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

# 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)

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

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

noises

where  $\epsilon \sim \mathcal{N}(0, \sigma^2)$ .

- 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  $\epsilon_i$ .
- For a fixed  $\mathbf{x}$  and a fixed  $\mathbf{w}$ ,  $f(\mathbf{x}, \mathbf{w})$  is a constant, then

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

where  $\epsilon$  is a continuous RV.

- What is the pdf for  $y$ ? (we are adding a constant to a Gaussian RV)
  - $E\{y\} = E\{\text{constant} + \epsilon\} = \text{constant}$
  - $\text{var}\{y\} = \text{var}\{\text{constant}\} + \text{var}\{\epsilon\} = \sigma^2$ .

## Gaussian regression model (II)

- This means that

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

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

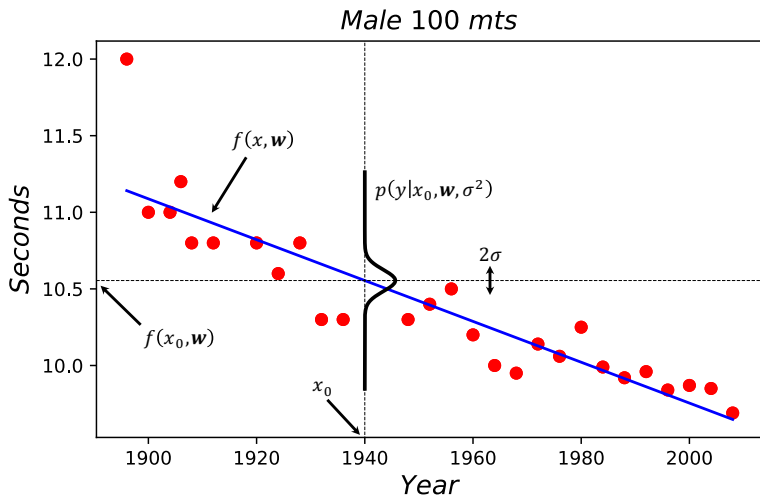
$$y \sim \mathcal{N}(f(\mathbf{x}, \mathbf{w}), \sigma^2).$$

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

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

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

# Gaussian regression model (III)



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

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

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

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

$$p(y_n | \mathbf{x}_n, \mathbf{w}, \sigma^2) = \mathcal{N}(y_n | f(\mathbf{x}_n, \mathbf{w}), \sigma^2) = \mathcal{N}(y_n | \mathbf{w}^\top \mathbf{x}_n, \sigma^2).$$

- 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

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

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

- The expression above can then be written as

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



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

- When we look at a Gaussian pdf, like

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

we assume that both  $\mu$  and  $\sigma^2$  are given. In this case, the pdf follows all the properties we reviewed before.

- The same is true for

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

- Given  $\mathbf{w}^\top \mathbf{x}_n$  and  $\sigma^2$ , then each  $p(y_n|\mathbf{x}_n, \mathbf{w}, \sigma^2)$  is a pdf.
- 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  $\mathbf{w}^\top$  (therefore I don’t know what is  $\mathbf{w}^\top \mathbf{x}_n$ )”
  - “I don’t know what is  $\sigma^2$ ”.

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

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

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

receives the name of a *likelihood function*.

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

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

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

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

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

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

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

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

- 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

- If we multiply  $LL(\mathbf{w}, \sigma^2)$  by minus one, we get

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

- The ML criterion for this model has a close connection with the sum-of-squared errors used in non-probabilistic formulations of linear regression.
- Maximising the log-likelihood function is equivalent to minimising the sum-of-squares errors.
- Notice that the log is a monotonic function, meaning that if we find  $\mathbf{w}, \sigma^2$  that maximise  $g(\mathbf{w}, \sigma^2)$ , those will also maximise  $\log(g(\mathbf{w}, \sigma^2))$ .

# Normal equation (I)

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

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

- 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

$$LL(\mathbf{w}, \sigma^2) = -\frac{N}{2} \log(2\pi) - \frac{N}{2} \log \sigma^2 - \frac{1}{2\sigma^2} (\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w})$$

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

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

## Normal equation (II)

- We can find the  $\mathbf{w}$  that maximises  $LL(\mathbf{w}, \sigma^2)$  by taking the gradient  $\frac{dLL(\mathbf{w}, \sigma^2)}{d\mathbf{w}}$ , equating to zero and solving for  $\mathbf{w}$ .
- Taking the gradient of each term in  $LL(\mathbf{w}, \sigma^2)$  wrt  $\mathbf{w}$ , we get

$$\begin{aligned}\frac{d}{d\mathbf{w}} \left[ -\frac{N}{2} \log(2\pi) \right] &= 0, & \frac{d}{d\mathbf{w}} \left[ -\frac{N}{2} \log \sigma^2 \right] &= 0, & \frac{d}{d\mathbf{w}} \left[ -\frac{1}{2\sigma^2} \mathbf{y}^\top \mathbf{y} \right] &= 0, \\ \frac{d}{d\mathbf{w}} \left[ \frac{1}{2\sigma^2} \mathbf{w}^\top \mathbf{X}^\top \mathbf{y} \right] &= \frac{1}{2\sigma^2} \mathbf{X}^\top \mathbf{y}, \\ \frac{d}{d\mathbf{w}} \left[ \frac{1}{2\sigma^2} \mathbf{y}^\top \mathbf{X} \mathbf{w} \right] &= \frac{1}{2\sigma^2} \mathbf{X}^\top \mathbf{y} \\ \frac{d}{d\mathbf{w}} \left[ -\frac{1}{2\sigma^2} \mathbf{w}^\top \mathbf{X}^\top \mathbf{X} \mathbf{w} \right] &= -\frac{1}{2\sigma^2} 2\mathbf{X}^\top \mathbf{X} \mathbf{w}\end{aligned}$$

## Normal equation (III)

- Putting these terms together, we get

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

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

$$\begin{aligned}\frac{1}{\sigma^2} \mathbf{X}^\top \mathbf{y} - \frac{1}{\sigma^2} \mathbf{X}^\top \mathbf{X} \mathbf{w} &= \mathbf{0} \\ \mathbf{X}^\top \mathbf{X} \mathbf{w} &= \mathbf{X}^\top \mathbf{y} \\ \mathbf{w}_* &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.\end{aligned}$$

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

# Solving for $\sigma_*^2$

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

$$\sigma_*^2 = \frac{1}{N}(\mathbf{y} - \mathbf{X}\mathbf{w}_*)^\top (\mathbf{y} - \mathbf{X}\mathbf{w}_*).$$



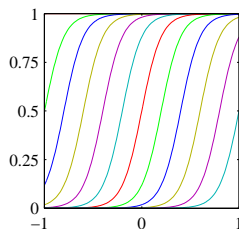
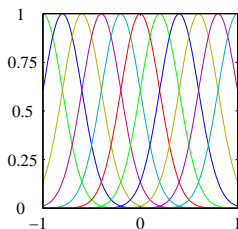
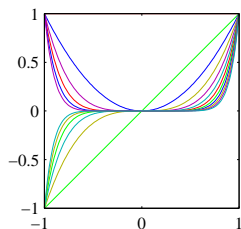
# Basis functions

- The model that is linear in  $\mathbf{x}$  only allows linear relationships between  $\mathbf{x}$  and  $y$ .
- 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.
- However, we keep the linear relationship of  $y$  wrt  $\mathbf{w}$  for tractability.
- The predictive model follows as  $f(\mathbf{x}, \mathbf{w})$

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

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

# Examples of basis functions



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

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

Sigmoidal:  $\phi_i(x) = \sigma\left(\frac{x-\mu_i}{s}\right)$ ,  $\sigma(a) = 1/(1 + \exp(-a))$ .

# Transforming the input using the basis functions

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

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

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

# Normal equations with a design matrix

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

$$\Phi = \begin{bmatrix} \phi(\mathbf{x}_1)^\top \\ \phi(\mathbf{x}_2)^\top \\ \vdots \\ \phi(\mathbf{x}_N)^\top \end{bmatrix} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_M(\mathbf{x}_2) \\ \vdots & \vdots & \cdots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{bmatrix}$$

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

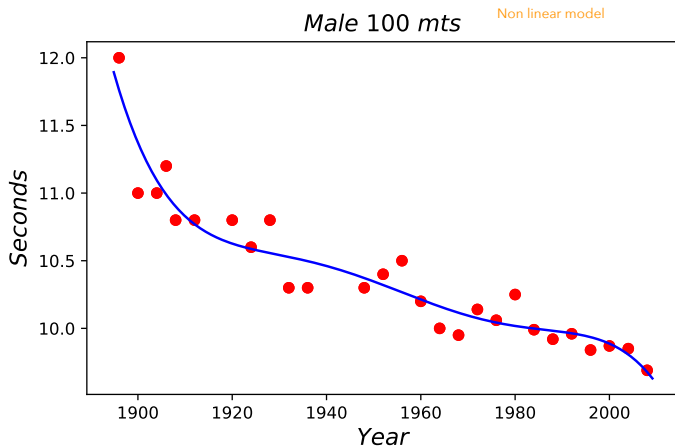
$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2) = \prod_{n=1}^N \mathcal{N}(y_n | \mathbf{w}^\top \phi_n, \sigma^2),$$

where  $\phi_n = \phi(\mathbf{x}_n)$ .

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

$$\mathbf{w}_* = (\Phi^\top \Phi)^{-1} \Phi^\top \mathbf{y}.$$

# Olympic 100-mt data with $M = 5$



# Alternative to find $\mathbf{w}$

different ways of finding  $\mathbf{w}$

- ❑ For solving the normal equation, we need to invert  $\mathbf{X}^T \mathbf{X}$ .
- ❑ This inversion has a computational complexity between  $\mathcal{O}((D+1)^{2.4})$  to  $\mathcal{O}((D+1)^3)$  (depending on the implementation).
- ❑ The normal equation is linear regarding the number of instances in the training data,  $\mathcal{O}(N)$ .
- ❑ It can handle a large training set as long as it fits in memory.
- ❑ 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

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

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \eta \mathbf{d}_k,$$

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

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

- The parameter  $\eta$  is known as the **step size** or **learning rate**.



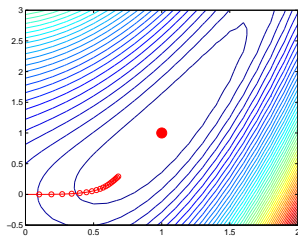
# Gradient descent

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

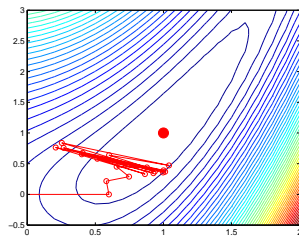
$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \mathbf{g}_k.$$

# Step size

- The main issue in gradient descent is how to set the step size.
- 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)  $\eta = 0.1$ . In (b)  $\eta = 0.6$ .

# Alternatives to choose the step size $\eta$

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

# Gradient descent for linear regression (I)

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

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

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

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \left. \frac{d}{d\mathbf{w}} E(\mathbf{w}) \right|_{\mathbf{w}=\mathbf{w}_k}.$$

# Gradient descent for linear regression (II)

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

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

- The update equation follows as

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

- 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

- Always normalise the features if using gradient descent.
- Gradient descent converges faster if all features have a similar scale.
- 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

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



# Stochastic gradient descent (I)

- In stochastic gradient descent (SGD), the gradient  $\mathbf{g}_k$  is computed using a subset of the instances available.
- The word update when new instances added **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)

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

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

# Step size in SGD

- Choosing the value of  $\eta$  is particularly important in SGD since there is no easy way to compute it.
- Usually the value of  $\eta$  will depend on the iteration  $k$ ,  $\eta_k$ .
- 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  $\eta_k$  can be used

$$\eta_k = \frac{1}{k}, \quad \eta_k = \frac{1}{(\tau_0 + k)^\kappa},$$

where  $\tau_0$  slows down early iterations and  $\kappa \in (0.5, 1]$ .

# Contents

Review of vector/matrix notation and linear algebra

A regression model

Linear regression

Gradient descent

Stochastic Gradient Descent

Regularisation

# What is regularisation?

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

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

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

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

# Different types of regularisation

- The objective function for linear regression would be

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

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

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

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

lasso regularisation

- If  $\alpha = 1$ , we get  $\ell_1$  regularisation.
- If  $\alpha = 0$ , we get  $\ell_2$  regularisation. Rich regression
- If  $0 < \alpha < 1$ , we get the elastic net regularisation.

# Ridge regression or $\ell_2$ regularisation

- In ridge regression,  $\alpha = 0$ ,

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

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

$$\mathbf{w}_* = \left( \mathbf{X}^\top \mathbf{X} + \frac{\lambda N}{2} \mathbf{I} \right)^{-1} \mathbf{X}^\top \mathbf{y}.$$

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