Logistic regression

Procheta Sen

UNIVERSITY OF LIVERPOOL

Probabilistic vs "ordinary" classifier

- **"Ordinary" classifier** is a function f that assigns to an input object $\overline{X}$ a predicted class c from a fixed set of classes $\{c_1, c_2, \ldots, c_k\}$, i.e.

$$c = f(\overline{X}).$$

- **Probabilistic classifier** is a ___conditional distribution $P(C\,|\,\overline{X})$___. For an input object $\overline{X}$ it gives probabilities $p_1, p_2, \ldots, p_k$, where

$$p_i = P(c_i \mid \overline{X})$$

and $p_1 + p_2 + \ldots + p_k = 1.$

Two types of models: discriminative vs generative

Discriminative

- Assume that the conditional distribution $P(C \mid X)$ (i.e. the probabilistic classifier) has specific form $P_\theta(C \mid X)$ depending on some parameters $\theta = (\theta_1, \ldots, \theta_k)$
- Use training data set to **find** / **learn** parameters $\theta_1, \ldots, \theta_k$ such that the resulting distribution is "best possible" among all distributions of the assumed form

Generative

- Assume that data come from specific distribution $P_\theta(X, C)$ depending on some parameters $\theta = (\theta_1, \ldots, \theta_k)$
- Use training data set to **find** / **learn** parameters $\theta_1, \ldots, \theta_k$ such that the resulting distribution is "best possible" among all distributions of the assumed form
- Use $P_\theta(X, C)$ to classify new objects

Probabilistic classifiers

Generative

- Naive Bayes

$$P(H \mid E) = \frac{P(E, H)}{P(E)} = \frac{P(E \mid H)P(H)}{P(E)}$$

…

Discriminative

- **Logistic regression** (today!)
- Multilayer perceptrons (neural networks)

…

Setup

- We consider the binary classification problems with classes $\{-1, +1\}$
- We want to build a probabilistic classifier that outputs the probability of a particular training instance $\overline{X}$ being positive ($y = +1$) or negative ($y = -1$)

Logistic regression: main idea

- Define a separating hyperplane H **parameterised** by the feature weights $\overline{W} = (w_1, \ldots, w_d)$ and a bias parameter b, i.e.

$$H = \left\{ b + \sum_{i=1}^{d} w_i x_i = 0 \mid x_1, \ldots, x_d \right\}$$

- **In perceptron**, to classify an input object $\overline{X} = (x_1, \ldots, x_d)$, we used only the sign of

$$b + \overline{W}^T \overline{X} = b + \sum_{i=1}^{d} w_i x_i,$$

which tells in which of the two half-spaces created by the hyperplane the point is located.

Logistic regression: main idea

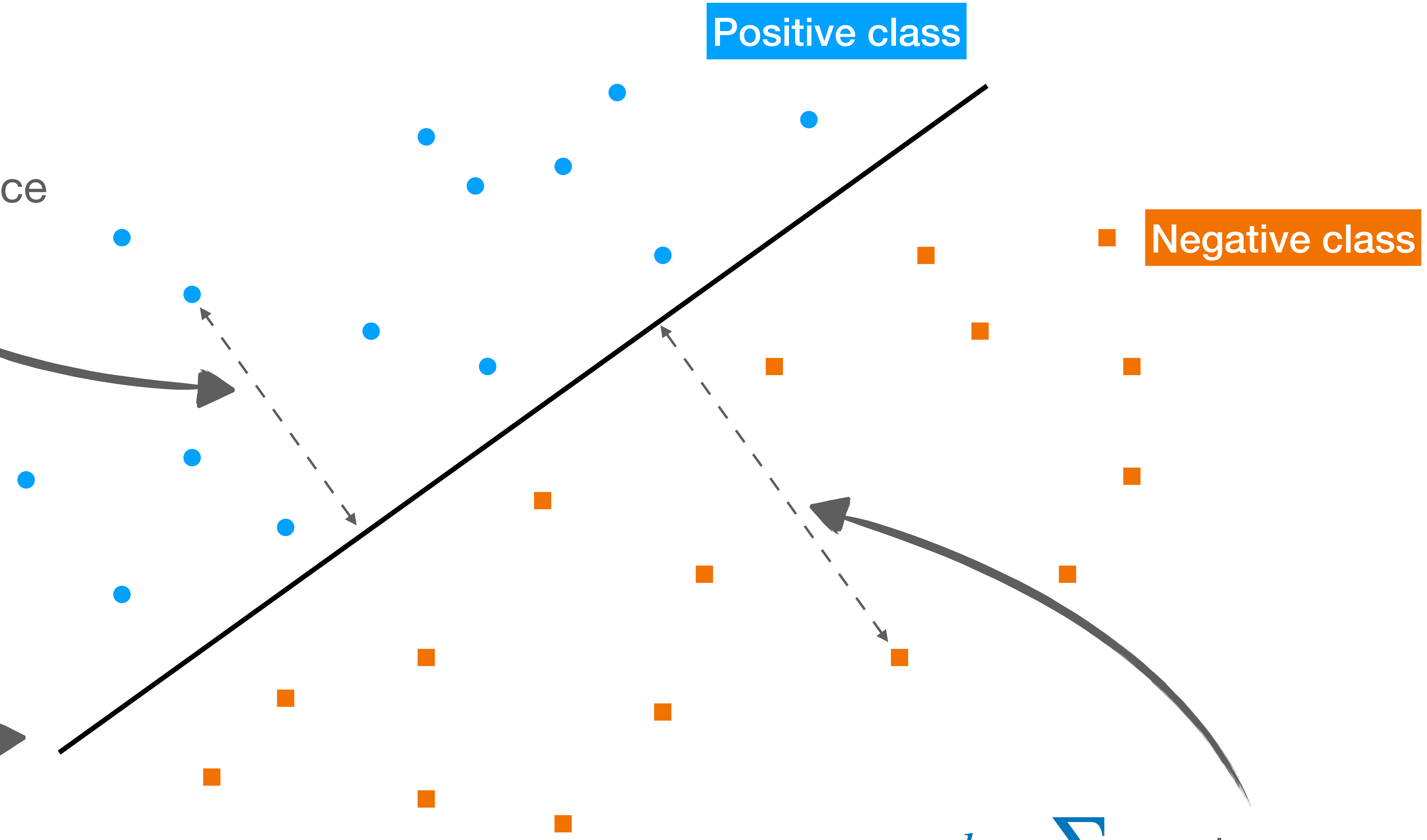
- However, the actual value of $b + \overline{W}^T \overline{X}$ conveys extra useful information: it is proportional to the distance from point $\overline{X}$ to the hyperplane H

- **Logistic regression** uses
    - sign of $b + \overline{W}^T \overline{X}$ to **classify object** $\overline{X}$ and
    - $|b + \overline{W}^T \overline{X}|$ to **quantify our confidence** in this classification: the larger the value, the further $\overline{X}$ from the separating hyperplane H

Logistic regression: main idea

$b + \sum w_i x_i$ is

1) positive
2) proportional to the distance



$b + \sum w_i x_i$ is

1) negative
2) proportional to the distance

Hyperplane defined by $b + \sum w_i x_i = 0$

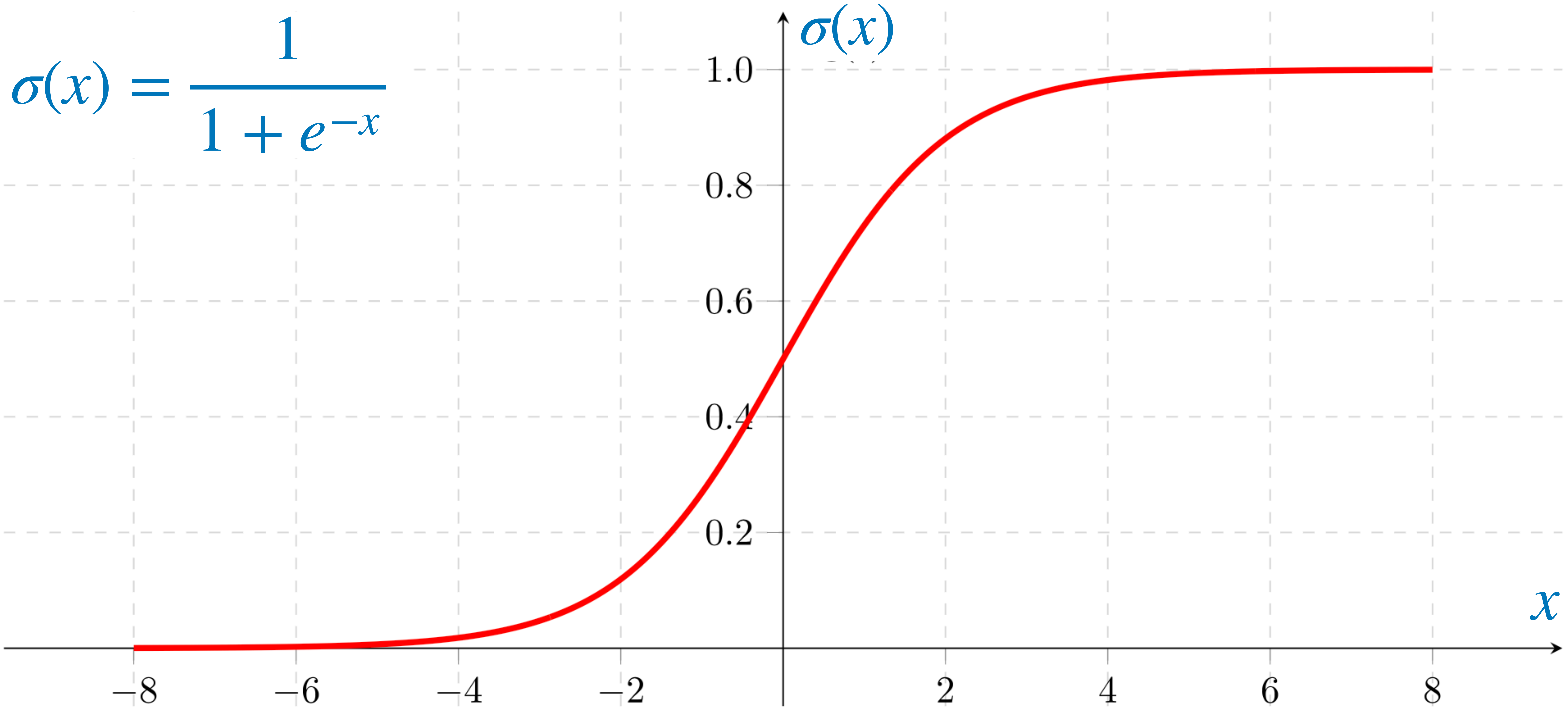
Logistic regression: main idea

- To interpret the confidence score $b + \overline{W}^T \overline{X} \in [-\infty, \infty]$ as probability we would like to transform it to a value in the interval $[0,1]$ so that $-\infty$ maps to 0 and ∞ maps to 1.

- A function that does this is the **logistic sigmoid function**.

Logistic sigmoid function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Logistic sigmoid function properties

- $\sigma(x) \in [0,1]$ for any $x \in [-\infty, \infty]$
- $1 - \sigma(x) = \sigma(-x)$
- $\dfrac{\partial \sigma}{\partial x} = \sigma(x) \cdot (1 - \sigma(x))$

Logistic regression: discriminative classifier

Discriminative classifier

- Assume that the conditional distribution $P(C \mid X)$ (i.e. the probabilistic classifier) has specific form $P_\theta(C \mid X)$ depending on some parameters $\theta = (\theta_1, \ldots, \theta_k)$
- Use training data set to **find** / **learn** parameters $\theta_1, \ldots, \theta_k$ such that the resulting distribution is "best possible" among all distributions of the assumed form

Logistic regression: model assumption

- For an object $\overline{X} = (x_1, \ldots, x_d)$, the probability that $\overline{X}$ belongs to the positive class is modelled as

$$P(y = +1 \mid \overline{X}) = \sigma(a) = \frac{1}{1 + e^{-a}},$$

where $a = b + \overline{W}^T \overline{X}$. Hence the probability that $\overline{X}$ belongs to the negative class is

$$P(y = -1 \mid \overline{X}) = 1 - P(y = +1 \mid \overline{X}) = 1 - \sigma(a) = \sigma(-a) = \frac{1}{1 + e^{a}}$$

It is convenient to write $P(y = t \mid \overline{X}) = \sigma(t \cdot a) = \dfrac{1}{1 + e^{-t \cdot a}}$, where $t \in \{-1, +1\}$

Logistic regression: choosing/fitting parameters

- Let $\mathscr{D} = \left\{(\overline{X_1}, y_1), (\overline{X_2}, y_2), \ldots, (\overline{X_n}, y_n)\right\}$ be the training data set
- Using the **maximum likelihood estimation method** we would like to find parameters $b, w_1, w_2, \ldots, w_d$ that maximise the likelihood function

$$\ell(b, w_1, w_2, \ldots, w_d, \mathscr{D}) = \prod_{i=1}^{n} \sigma\left(y_i(b + \overline{W}^T \overline{X_i})\right)$$

- This is equivalent to minimising $-\ell$ or minimising the negative log-likelihood function

$$-\ell\ell = -\log \ell = -\sum_{i=1}^{n} \log \sigma\left(y_i(b + \overline{W}^T \overline{X_i})\right)$$

Logistic regression: choosing/fitting parameters

$$-\ell\ell = -\log \ell = -\sum_{i=1}^{n} \log \sigma\left(y_i(b + \overline{W}^T \overline{X}_i)\right)$$

$$\nabla_{b,w_1,\ldots,w_d}(-\ell\ell) = -\nabla_{b,w_1,\ldots,w_d}\ell\ell$$

Denote $a_i = b + \overline{W}^T \overline{X}_i$

We need to compute $\dfrac{\partial \ell\ell}{\partial b}$ and $\dfrac{\partial \ell\ell}{\partial w_k}$ for every $k = 1, \ldots, d$.

Logistic regression: choosing/fitting parameters

$$\ell\ell = \log \ell = \sum_{i=1}^{n} \log \sigma \left(y_i (b + \overline{W}^T \overline{X}_i) \right) = \sum_{i=1}^{n} \log \sigma \left(y_i \cdot a_i \right)$$

$$\frac{\partial \ell\ell}{\partial b} =$$

Logistic sigmoid function properties

1) $\frac{\partial \sigma}{\partial x} = \sigma(x) \cdot (1 - \sigma(x))$

2) $1 - \sigma(x) = \sigma(-x)$

Logistic regression: choosing/fitting parameters

$$\ell\ell = \log \ell = \sum_{i=1}^{n} \log \sigma \left(y_i (b + \overline{W}^T \overline{X}_i) \right) = \sum_{i=1}^{n} \log \sigma \left(y_i \cdot a_i \right)$$

$$\frac{\partial \ell\ell}{\partial b} = \sum_{i=1}^{n} y_i \cdot \sigma\left(-y_i (b + \overline{W}^T \overline{X}_i) \right) = \sum_{i=1}^{n} y_i \cdot \sigma\left(-y_i \cdot a_i \right)$$

Logistic sigmoid function properties

1) $\frac{\partial \sigma}{\partial x} = \sigma(x) \cdot (1 - \sigma(x))$

2) $1 - \sigma(x) = \sigma(-x)$

Logistic regression: choosing/fitting parameters

$$\ell\ell = \log \ell = \sum_{i=1}^{n} \log \sigma \left(y_i (b + \overline{W}^T \overline{X}_i) \right) = \sum_{i=1}^{n} \log \sigma \left(y_i \cdot a_i \right)$$

$$\frac{\partial \ell\ell}{\partial w_k} =$$

Logistic sigmoid function properties

1) $\dfrac{\partial \sigma}{\partial x} = \sigma(x) \cdot (1 - \sigma(x))$

2) $1 - \sigma(x) = \sigma(-x)$

Logistic regression: choosing/fitting parameters

$$\ell\ell = \log \ell = \sum_{i=1}^{n} \log \sigma\left(y_i(b + \overline{W}^T \overline{X}_i)\right) = \sum_{i=1}^{n} \log \sigma\left(y_i \cdot a_i\right)$$

$$\frac{\partial \ell\ell}{\partial w_k} = \sum_{i=1}^{n} y_i \cdot \sigma\left(-y_i(b + \overline{W}^T \overline{X}_i)\right) \cdot x_k^{(i)} = \sum_{i=1}^{n} y_i \cdot \sigma\left(-y_i \cdot a_i\right) \cdot x_k^{(i)},$$

where $\overline{X}_i = (x_1^{(i)}, x_2^{(i)}, \ldots, x_d^{(i)})$

Logistic regression: choosing/fitting parameters

$$\ell\ell = \log \ell = \sum_{i=1}^{n} \log \sigma\left(y_i(b + \overline{W}^T \overline{X}_i)\right) = \sum_{i=1}^{n} \log \sigma\left(y_i \cdot a_i\right)$$

Interpretation of $\sum_{i=1}^{n} y_i \cdot \sigma\left(-y_i \cdot a_i\right)$

- If $y_i = +1$, then $\sigma(-y_i \cdot a_i) = \sigma(-a_i) = 1 - \sigma(a_i) = P(y = -1 \mid \overline{X}_i)$
- If $y_i = -1$, then $\sigma(-y_i \cdot a_i) = \sigma(a_i) = P(y = +1 \mid \overline{X}_i)$
- Hence $\sigma(-y_i \cdot a_i)$ is the probability of misclassifying the training object X_i

$$\sum_{i=1}^{n} y_i \cdot \sigma\left(-y_i \cdot a_i\right) = \sum_{\overline{X}_i \in \mathcal{D}_+} P(y = -1 \mid \overline{X}_i) - \sum_{\overline{X}_i \in \mathcal{D}_-} P(y = +1 \mid \overline{X}_i)$$

Logistic regression: update rule

Gradient Descent method for finding local minimum of $f(\overline{Z}) = f(z_1, \ldots, z_d)$

1. Pick an initial point $\overline{Z}_0$

2. Iterate according to

$$\overline{Z}_{i+1} = \overline{Z}_i - \gamma_i \cdot \big((\nabla_{\overline{Z}} f)(\overline{Z}_i) \big)$$

where $\gamma_1, \gamma_2, \ldots,$ are step-sizes.

Logistic regression: update rule

minimize $-\ell\ell = -\log \ell = -\sum_{i=1}^{n} \log \sigma\left(y_i(b + \overline{W}^T \overline{X}_i)\right)$

$$\frac{\partial \ell\ell}{\partial b} = \sum_{i=1}^{n} y_i \cdot \sigma\big(-y_i \cdot a_i\big)$$

$$\frac{\partial \ell\ell}{\partial w_k} = \sum_{i=1}^{n} y_i \cdot \sigma\big(-y_i \cdot a_i\big) \cdot x_k^{(i)},\ k = 1,\ldots,d$$

Hence we have the following update rule

$$b \leftarrow b + \mu \cdot \sum_{i=1}^{n} y_i \cdot \sigma\big(-y_i \cdot a_i\big)$$

uses whole training set

$$\overline{W} \leftarrow \overline{W} + \mu \cdot \sum_{i=1}^{n} y_i \cdot \sigma\big(-y_i \cdot a_i\big)\overline{X}_i$$

Online vs Batch

Batch

- Uses the **entire training dataset** in every iteration to update the weight vector
- Popular optimisation algorithm for the batch learning of logistic regression is the Limited Memory BFGS (L-BFGS) algorithm
- Batch version is slow compared to the online version. But shows slightly improved accuracies in many cases

Online

- Uses only **one training object** in every iteration to update the weight vector
- The Stochastic Gradient Descent algorithm (SGD)
- SGD version can require multiple iterations over the dataset before it converges (if ever)
- SGD is a technique that is frequently used with large scale machine learning tasks (even when the objective function is non-convex)

Logistic regression online algorithm (Stochastic Gradient Descent)

`LogisticRegression`(Training data: $\{(\overline{X}_1, y_1), \ldots, (\overline{X}_n, y_n)\}$, Learning rate μ, `MaxIter`)

$w_i = 0$ `for all` $i = 1, \ldots, d$;

$b = 0$

for `iter = 1 … MaxIter` **do**

  for `i = 1 … n` **do**

    $a_i = b + \overline{W}^T \overline{X}_i$

    $w_s = w_s + \mu \cdot y_i \cdot \sigma(-y_i \cdot a_i) \cdot x_s^{(i)}$, `for all` $s = 1, \ldots, d$

    $b = b + \mu \cdot y_i \cdot \sigma(-y_i \cdot a_i)$

return $b, w_1, w_2, \ldots, w_d$

$$\overline{W} \leftarrow \overline{W} + \mu \cdot y_i \cdot \sigma\big(-y_i \cdot a_i\big)\overline{X}_i$$
$$b \leftarrow b + \mu \cdot y_i \cdot \sigma\big(-y_i \cdot a_i\big)$$

Logistic regression prediction

LogisticRegressionTest$(b, w_1, w_2, \ldots, w_d, \overline{X})$

$a = b + \overline{W}^T \overline{X}$

if $a > 0$ then

    predicted label = +1 #positive class

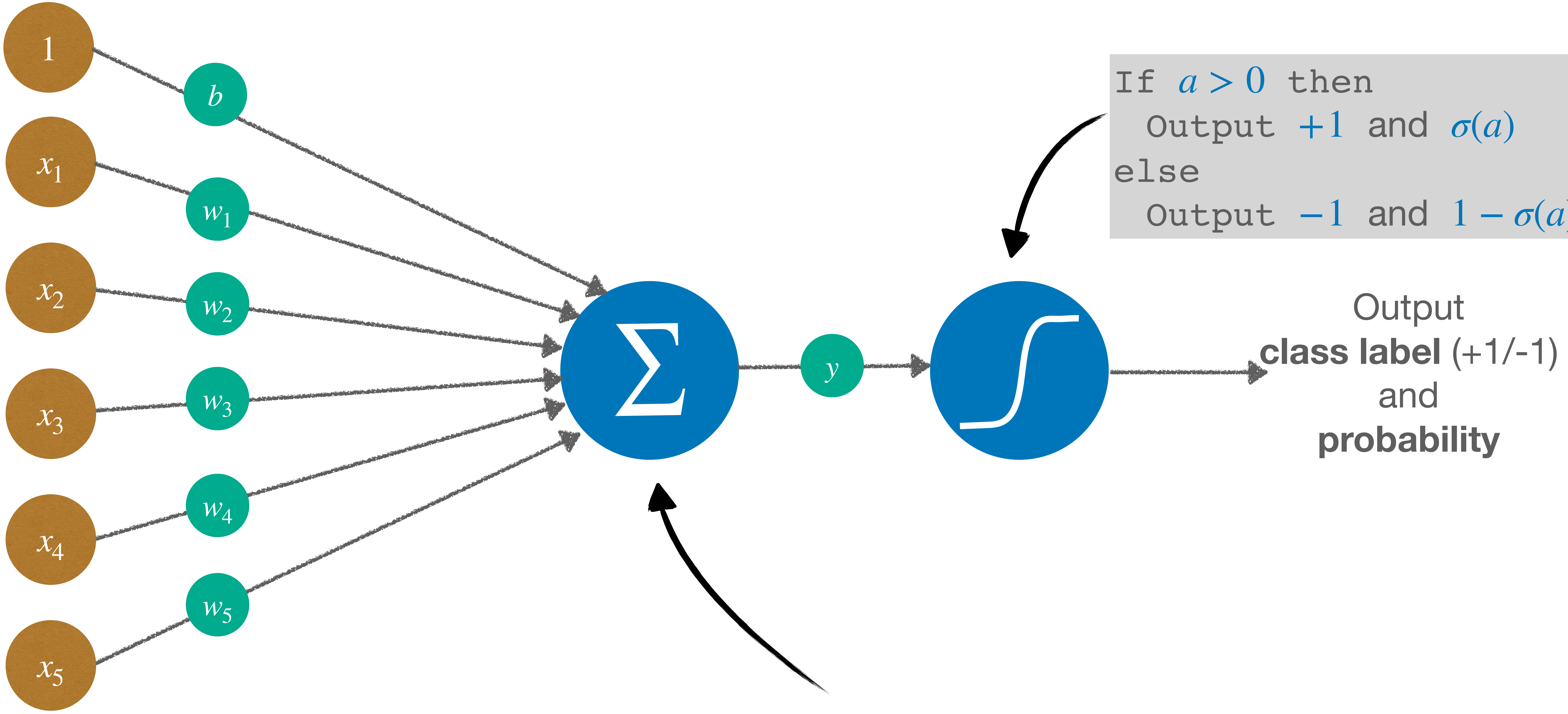
    probability that $\overline{X}$ belongs to the positive class = $\sigma(a)$ #confidence

else

    predicted label = -1 #negative class

    probability that $\overline{X}$ belongs to the negative class = $1 - \sigma(a)$ #confidence

Logistic regression: neuron interpretation




```
If a > 0 then
  Output +1 and σ(a)
else
  Output −1 and 1 − σ(a)
```

$$a = b + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5$$

L2 regularisation

- Let us denote by $L(\mathscr{D}, \overline{W})$ the Loss of classifying a dataset $\mathscr{D}$ using a model represented by a weight vector $\overline{W}$
- We would like to impose L2 regularisation on $\overline{W}$.
- The overall objective to minimise can then be written as follows

$$J(\mathscr{D}, \overline{W}) = L(\mathscr{D}, \overline{W}) + \lambda\,||\,\overline{W}\,||_2^2 = L(\mathscr{D}, \overline{W}) + \lambda \sum_{i=1}^{d} w_i^2.$$

Here λ is called the **regularisation coefficient** and is usually set via **cross-validation.**

- The gradient of the overall objective simply becomes the addition of the loss-gradient and the scaled weight vector $\overline{W}$.

$$\nabla_{\overline{W}} J(\mathscr{D}, \overline{W}) = \nabla_{\overline{W}} L(\mathscr{D}, \overline{W}) + 2\lambda\overline{W}$$

L2 regularisation in logistic regression

L2 regularised logistic regression update rule for training object $(\overline{X}, y)$ with $a = b + \overline{W}^T X$

No regularisation: $\overline{W} \leftarrow \overline{W} + \mu \cdot y \cdot \sigma\left(-y \cdot a\right) \cdot \overline{X}$

With regularisation:

$$\overline{W} \leftarrow \overline{W} - \mu \cdot \left(-y \cdot \sigma\left(-y \cdot a\right) \cdot \overline{X} + 2\lambda\overline{W}\right)$$

$$= \overline{W} + \mu \cdot y \cdot \sigma\left(-y \cdot a\right) \cdot \overline{X} - 2\mu\lambda\overline{W}$$

$$= (1 - 2\mu\lambda)\overline{W} + \mu \cdot y \cdot \sigma\left(-y \cdot a\right) \cdot \overline{X}$$