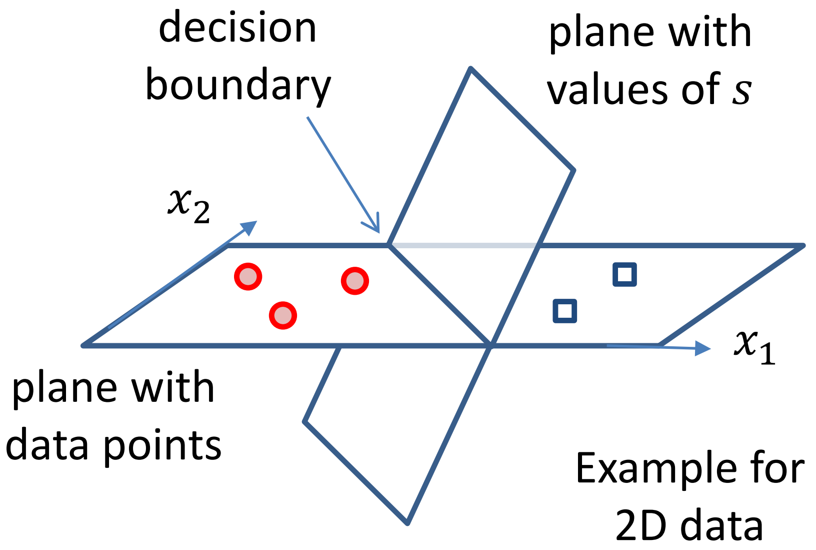
Perceptron is a linear binary classifier

Perceptron is a binary classifier:

Predict class A if $s \geq 0$
Predict class B if $s < 0$
where $s = \sum_{i=0}^{m} x_i w_i$

Perceptron is a <u>linear classifier</u>: s is a linear function of inputs, and the decision boundary is linear

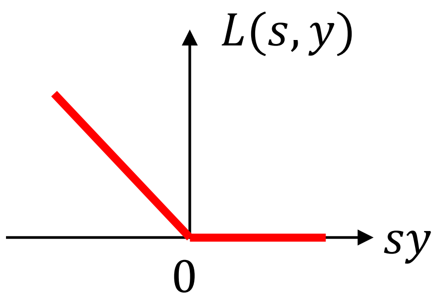


Loss function for perceptron

- "Training": finds weights to minimise some loss. Which?
- Our task is binary classification. Let's arbitrarily encode one class as $+1$ and the other as -1. So each training example is now $\{\boldsymbol{x}, y\}$, where y is either $+1$ or -1
- Recall that, in a perceptron, $s = \sum_{i=0}^{m} x_i w_i$, and the sign of s determines the predicted class: $+1$ if $s > 0$, and -1 if $s < 0$
- Consider a single training example. If y and s have same sign then the example is classified correctly. If y and s have different signs, the example is misclassified

Loss function for perceptron

- Consider a single training example. If y and s have the same sign then the example is classified correctly. If y and s have different signs, the example is misclassified
- The perceptron uses a loss function in which there is no penalty for correctly classified examples, while the penalty (loss) is equal to s for misclassified examples*
- Formally:
  - $L(s,y) = 0$ if both s, y have the same sign
  - $L(s,y) = |s|$ if both s, y have different signs



- This can be re-written as $L(s,y) = \max(0, -sy)$

* This is similar, but not identical to another widely used loss function called ***hinge loss***

Perceptron training algorithm

Choose initial guess $\boldsymbol{w}^{(0)}$, $k = 0$

For i from 1 to T (epochs)

For j from 1 to N (training examples)

Consider example $\{\boldsymbol{x}_j, y_j\}$

<u>Update</u>*: $\boldsymbol{w}^{(k++)} = \boldsymbol{w}^{(k)} - \eta \nabla L(\boldsymbol{w}^{(k)})$

$$L(\boldsymbol{w}) = \max(0, -sy)$$

$$s = \sum_{i=0}^{m} x_i w_i$$

η is learning rate

*There is no derivative when $s = 0$, but this case is handled explicitly in the algorithm, see next slides

Perceptron training algorithm

When classified correctly, weights are unchanged

When misclassified: $\boldsymbol{w}^{(k+1)} = -\eta(\pm\boldsymbol{x})$
($\eta > 0$ is called *learning rate)*

<u>If $y = 1$, but $s < 0$</u>
$w_i \leftarrow w_i + \eta x_i$
$w_0 \leftarrow w_0 + \eta$

<u>If $y = -1$, but $s \geq 0$</u>
$w_i \leftarrow w_i - \eta x_i$
$w_0 \leftarrow w_0 - \eta$

<u>*Convergence Theorem*</u>: if the training data is linearly separable, the algorithm is guaranteed to converge to a solution. That is, there exist a finite K such that $L(\boldsymbol{w}^K) = 0$