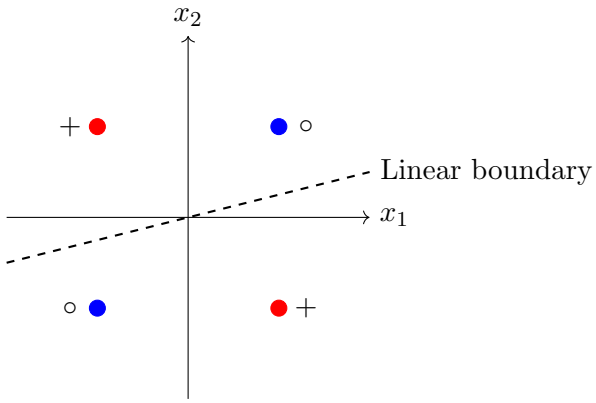
1 Basic Computations

It's good to know how vectors and matrix vector products and derivatives work.

- $\begin{bmatrix}1\\2\end{bmatrix} + \begin{bmatrix}3\\2\end{bmatrix} =$
- $\begin{bmatrix}1\\2\\3\end{bmatrix} + \begin{bmatrix}2\\3\\1\end{bmatrix} + \begin{bmatrix}4\\2\\0\end{bmatrix} =$
- $\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix} \cdot \begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} =$
- $\frac{\partial}{\partial x}(2 + 4x^2 + e^x) =$
- For a vector 🙂 $\in \mathbb{R}^{n\times 1}$, $\nabla[🙂^\top(e^{\mathbf{x}})] =$

2 Linear Neural Networks

In their seminal 1969 book *Perceptrons*, MIT researchers Minsky and Papert showed that linear classifiers could **not** perfectly classify data according to the XOR operation. Naturally, they claimed, a model that learns a line should not be able to discriminate data which are nonlinearly correlated.

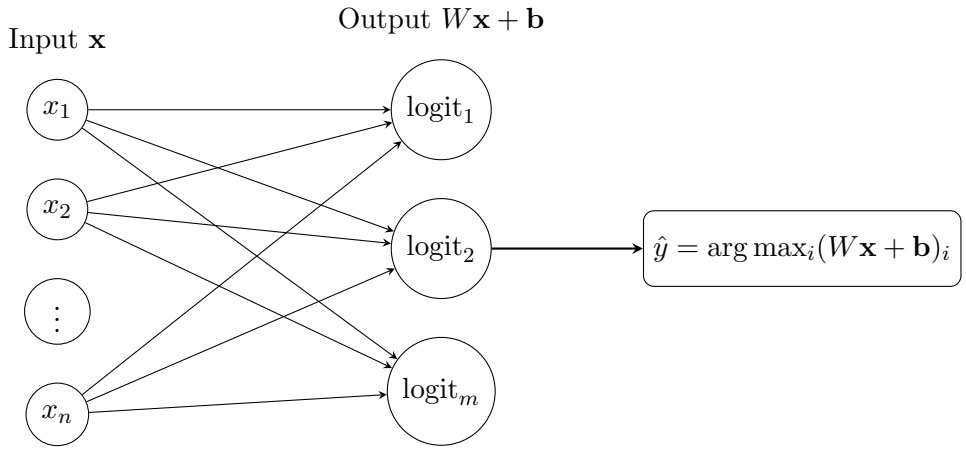


XOR data distribution centered at the origin.

Later work rectified this by imposing a nonlinear **activation function** through an intermediary **hidden layer**, allowing for nonlinear decision boundaries. In this problem, we will explore first why a linear classifier learns linear decision boundaries and second, why a hidden layer without nonlinearity is a redundancy.

(a) A linear classifier $f_\theta(\mathbf{x})$ is a single layer neural network which takes a datapoint $\mathbf{x} \in \mathbb{R}^{n\times 1}$ and outputs a prediction $\hat{y} \in \mathcal{C}$, where $\mathcal{C} = \mathbb{R}^m$ is the set of classes and $y \in \mathcal{C}$ is the true classification of $\mathbf{x}$. *Hint: n is the number of input features, m is the number of output classes*

$$\hat{y} = f_\theta(\mathbf{x}) = \arg\max_{i\in\mathcal{C}}(W\mathbf{x} + \mathbf{b})_i \quad W \in \mathbb{R}^{m\times n}, \mathbf{b} \in \mathbb{R}^{m\times 1}$$



In other words, we predict the class for which the associated entry in $W\mathbf{x} + \mathbf{b}$, which we call the **logits** of this network, is greatest. Geometrically, we can think of this as partitioning the input space into m regions, where any two regions are separated by a set of inputs for which the logits associated with each region are equal, i.e. the **decision boundary**.

Show that each decision boundary is a hyperplane and, using this, informally explain why a linear classifier would not be able to perfectly classify a nonlinear data distribution.

Hint: A hyperplane in a Euclidean vector space is defined as a set $\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{a}^T\mathbf{x} = b\}$ for some $\mathbf{a} \in \mathbb{R}^n, b \in \mathbb{R}$.

(b) Show that adding more layers to this network is redundant.

3 Perceptron

You want to predict if Saathvik will like the recommended anime. You get two recommendations from respected otaku, Chloe and Diana, to read a script you have and rate it on a scale of 1 to 4. The critics are not perfect; here are five data points including the critics' scores and the performance of the anime:

#	Anime Name	C	D	Like?
1	Dragon Ball Z	1	1	-
2	Death Note	3	2	+
3	Sword Art Online	2	4	+
4	One Piece	3	4	+
5	Hunter x Hunter	2	3	-

(a) First, you would like to examine the linear separability of the data. Plot the data on a 2D plane; label favorable anime with '+' and non-favorable anime with '-', and determine if the data are linearly separable.

(b) Now you decide to use a perceptron to classify your data. Suppose you directly use the scores given above as features, together with a bias feature. That is $f_0 = 1$, $f_1 =$ score given by Chloe, and $f_2 =$ score given by Diana.

Run one pass through the data with the perceptron algorithm, filling out the table below. Go through the data points in order, e.g., using data point #1 at step 1.

Step	Weights	Score	Correct?
1	[-1, 0, 0]	$-1 \cdot 1 + 0 \cdot 1 + 0 \cdot 1 = -1$	Yes
2			
3			
4			
5			

Final weights:

(c) Have weights been learned that separate the data?

(d) More generally, irrespective of the training data, you want to know if your features are powerful enough to allow you to handle a range of scenarios. Circle the scenarios for which a perceptron using the features above can indeed perfectly classify anime that are favorable according to the given rules:

(a) Your reviewers are awesome: if the total of their scores is more than 8, then the anime will definitely be favorable, and otherwise it won't be.

(b) Your reviewers are horror critics. Your anime will be favorable if and only if each reviewer gives either a score of 2 or a score of 3.

(c) Your reviewers have weird but different tastes. Your anime will be profitable if and only if both reviewers agree.

(Credits to CS 188 Fall 2024 for this question.)