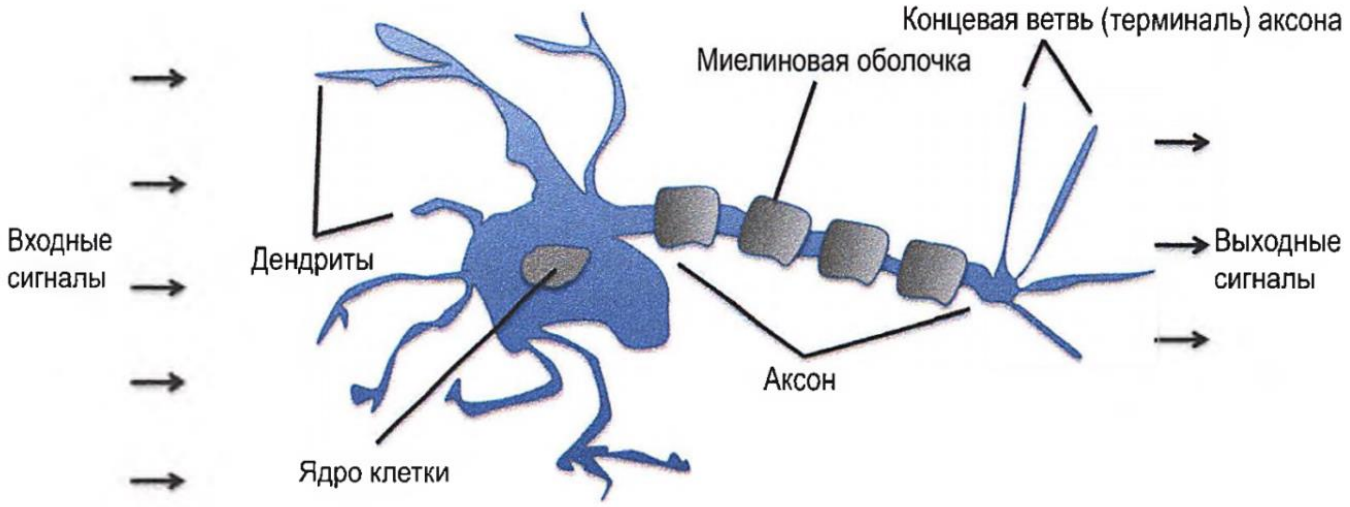
Краткая история машинного обучения

Нейроны – это взаимосвязанные нервные клетки головного мозга, которые участвуют в обработке и передаче химических и электрических сигналов.



Стараясь понять, каким образом работает биологический мозг, с целью разработки искусственного интеллекта *Уоррен Маккалок* и *Уолтер Питтс* в 1943 г. впервые опубликовали концепцию упрощенной клетки головного мозга, так называемого нейрона Маккалока-Питтса (нейрон MCP). Маккалок и Питтс описали такую нервную клетку в виде простого логического элемента с бинарными выходами. Множественные входные сигналы поступают в дендриты, затем интегрируются в клеточное тело, и если накопленный сигнал превышает определенный порог, то генерируется выходной сигнал, который аксоном передается дальше.

Всего несколько лет спустя Фрэнк Розенблатт представил научному сообществу концепцию правила обучения персептрона, опираясь на модель нейрона MCP. Вместе с правилом персептрона Розенблатт предложил алгоритм, который автоматически обучался оптимальным *весовым коэффициентам*, которые затем перемножались с входными признаками для принятия решения о том, активировать нейрон или нет. В контексте обучения с учителем и задачи классификации такой алгоритм можно использовать для распознавания принадлежности образца к тому или иному классу.

Более формально эту проблему можно изложить как задачу бинарной классификации, где для простоты мы обозначим наши два класса как 1 и -1. Затем можно определить передаточную функцию, или *функцию активации* $\varphi(z)$, где z – это так называемый *чистый вход* – линейная комбинация входного вектора x и соответствующего весового вектора ω.

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_m \end{pmatrix}, \quad \omega = \begin{pmatrix} \omega_1 \\ \omega_2 \\ \cdots \\ \omega_m \end{pmatrix}$$

$$z = (\omega_1 x_1 + \cdots + \omega_m x_m) = \sum_{j=1}^{m} x_j \omega_j = \omega^T x$$

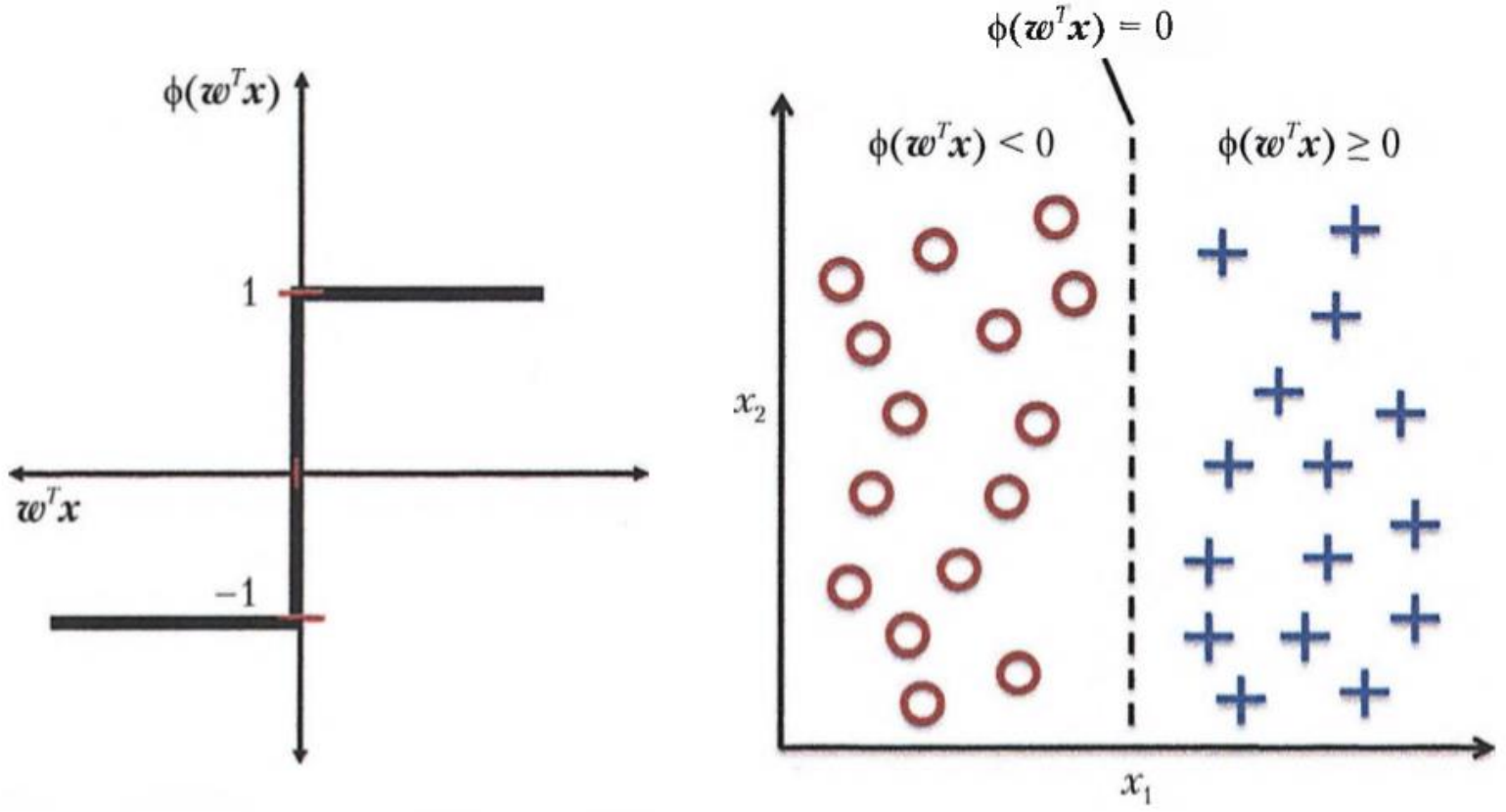
Если выход из $\varphi(z)$, превышает заданный *порог* θ ($\varphi(z) > \theta$), то мы распознаем класс 1, в противном случае – класс -1. В алгоритме персептрона функция активации $\varphi(z)$ – это простая единичная ступенчатая функция, которую иногда также называют *ступенчатой функцией Хевисайда* (или функцией единичного скачка).

Для простоты мы можем перенести порог θ в левую часть равенства и определить вес с нулевым индексом как $\omega_0 = \theta$ и $x_0 = -1$:

$$z = (\omega_0 x_0 + \omega_1 x_1 + \cdots + \omega_m x_m) = \sum_{j=0}^{m} x_j \omega_j = \omega^T x$$

$$\varphi(z) = \begin{cases} 1, \text{если } z \geq 0 \\ -1, \text{иначе} \end{cases}$$

На рисунках ниже показано, как чистый вход функцией активации персептрона преобразуется в бинарный выход (-1 либо 1), и как это может использоваться для различения двух линейно разделимых классов:



Весь смысл идеи, лежащей в основе нейрона MCP и персептронной модели Розенблатта с порогом, состоит в том, чтобы использовать редукционистский подход для имитации работы отдельного нейрона головного мозга: он либо активируется, либо нет. Таким образом, первоначальное правило обучения персептрона Розенблатта, т.е. правило обновления весов в персептроне, было довольно простым и может быть резюмировано следующими шагами:

1. Инициализировать веса нулями либо малыми случайными числами.
2. Для каждого обучающего объекта $x^{(i)}$ выполнить следующие шаги:

    1) вычислить выходное значение $\hat{y}$

    2) обновить веса.

Здесь выходное значение – это метка класса, идентифицированная единичной ступенчатой функцией, которую мы определили ранее, при этом одновременное обновление каждого веса ω_j в весовом векторе ω можно более формально записать как

$$\omega_j = \omega_j + \Delta\omega_j$$

Значение $\Delta\omega_j$ которое используется для обновления веса ω_j, вычисляется правилом обучения персептрона:

$$\Delta\omega_j = \eta(y^{(i)} - \widehat{y^{(i)}})x_j^{(i)}$$

где η – это темп обучения (константа между 0.0 и 1.0), $y^{(i)}$ – истинная метка класса i-го обучающего объекта и $\widehat{y^{(i)}}$ – идентифицированная метка класса. Важно отметить, что все веса в весовом векторе обновляются одновременно, т.е. мы не вычисляем $\widehat{y^{(i)}}$повторно до тех пор, пока все веса $\Delta\omega_j$ не будут обновлены. Конкретно для двумерного набора данных мы запишем обновление следующим образом:

$$\Delta\omega_0 = \eta(y^{(i)} - \text{выход}^{(i)})$$

$$\Delta\omega_1 = \eta\left(y^{(i)} - \text{выход}^{(i)}\right)x_1^{(i)}$$

$$\Delta\omega_2 = \eta(y^{(i)} - \text{выход}^{(i)})x_2^{(i)}$$

Выполним простой мысленный эксперимент, чтобы проиллюстрировать, каким до прекрасного простым это обучающее правило на самом деле является. В двух сценариях, где персептрон правильно распознает метку класса, веса остаются неизменными:

$$\Delta\omega_j = \eta(-1 - -1)x_j^{(i)} = 0$$
$$\Delta\omega_j = \eta(1 - 1)x_j^{(i)} = 0$$

Однако в случае неправильного распознавания веса продвигаются в направлении соответственно положительного или отрицательного целевого класса:

$$\Delta\omega_j = \eta(1 - -1)x_j^{(i)} = \eta(2)\ x_j^{(i)}$$
$$\Delta\omega_j = \eta(-1 - 1)x_j^{(i)} = \eta(-2)\ x_j^{(i)}$$

Чтобы получить более глубокое интуитивное понимание мультипликативного коэффициента $x_j^{(i)}$, обратимся к еще одному простому примеру, где:

$$y^{(i)} = 1, \widehat{y^{(i)}} = -1, \eta = 1$$

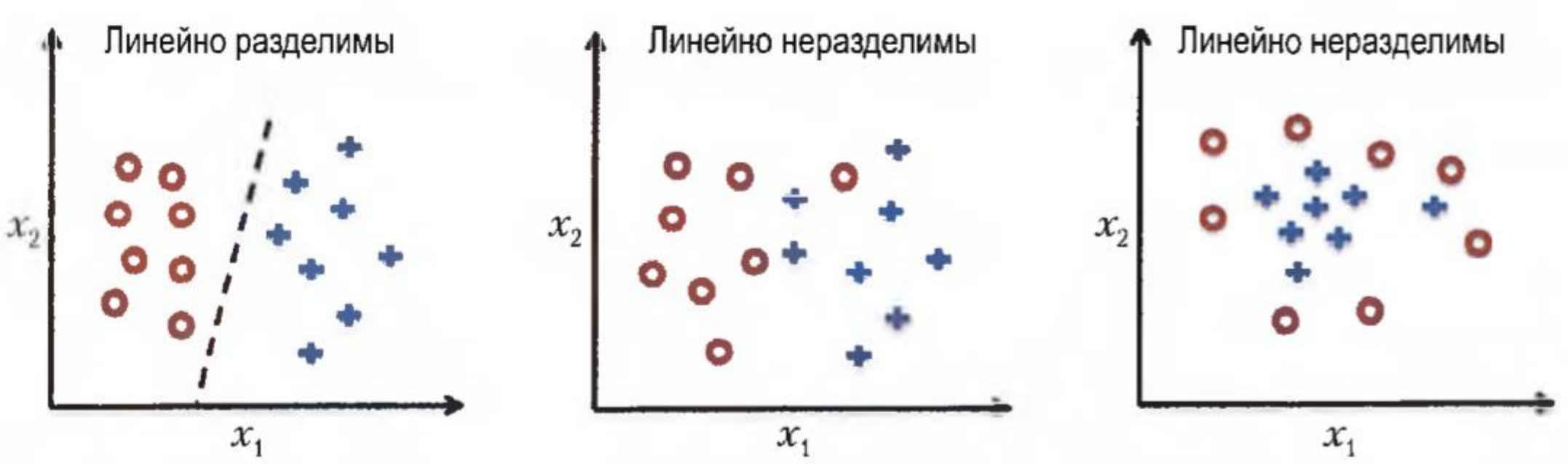
Пусть $x_j^{(i)} = 0.5$, и мы ошибочно классифицируем этот объект как -1. В этом случае мы увеличиваем соответствующий вес на 1, благодаря чему чистый вход будет более положительным при следующей встрече с этим объектом и, таким образом, с большей вероятностью будет выше порога единичной ступенчатой функции для классификации образца как 1:

$$\Delta\omega_j^{(i)} = (1 - -1)0.5 = 1$$

Обновление веса пропорционально значению $x_j^{(i)}$. Например, если имеется еще один образец $x_j^{(i)} = 2$, который правильно классифицируется как -1, то мы отодвинем границу решения на еще большее расстояние, чтобы в следующий раз классифицировать этот образец правильно:

$$\Delta\omega_j^{(i)} = (1 - -1)2 = 4$$

Важно отметить, что сходимость персептрона, т.е. достижение устойчивого состояния, гарантируется, только если эти два класса линейно разделимы и темп обучения достаточно небольшой. Если эти два класса не могут быть разделены линейной границей решения, то мы можем установить максимальное число проходов по обучающему набору данных (*эпох*) и/или порог на допустимое число случаев ошибочной классификации – иначе персептрон никогда не прекратит обновлять веса.



Резюме

Приведенный ниже рисунок иллюстрирует то, каким образом персептрон получает входы из объекта x и смешивает их с весами ω для вычисления чистого входа. Чистый вход затем передается функции активации (в данном случае единичная ступенчатая функция), которая генерирует бинарный выход -1 или +1 – распознанную метку класса объекта. Во время фазы обучения этот выход используется для вычисления ошибки распознавания и для обновления весов.