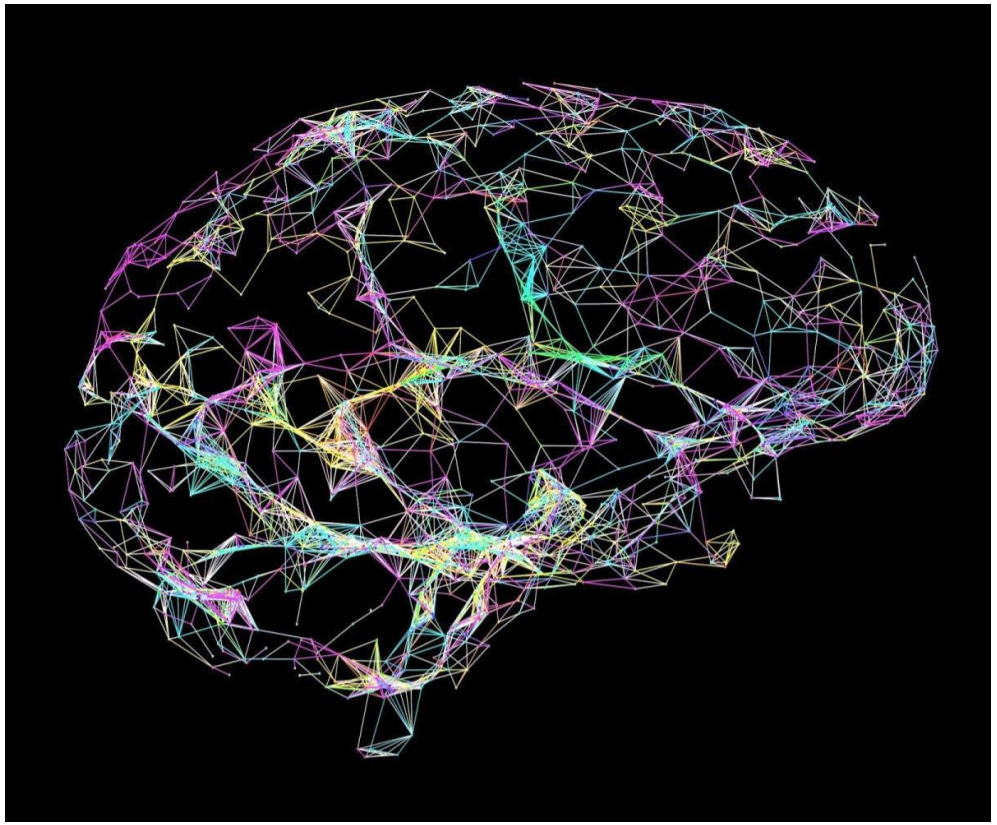
Градиентный спуск в нейронных сетях.

ТРИБУНСКАЯ ДАРЬЯ, ИС-1.1-23.

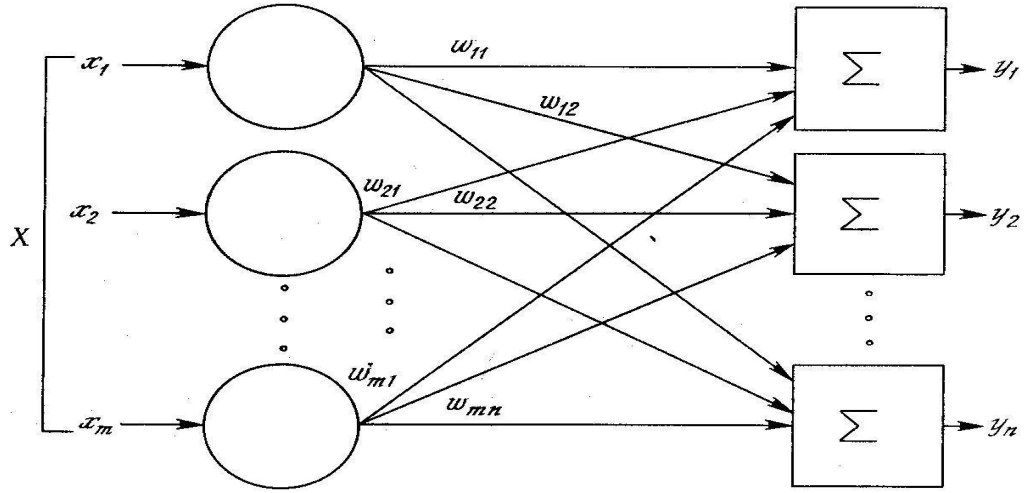
Что такое нейронных сети?

- Нейросеть — это тип машинного обучения, при котором компьютерная программа имитирует работу человеческого мозга. Подобно тому, как нейроны в мозге передают сигналы друг другу, в нейросети информацией обмениваются вычислительные элементы.

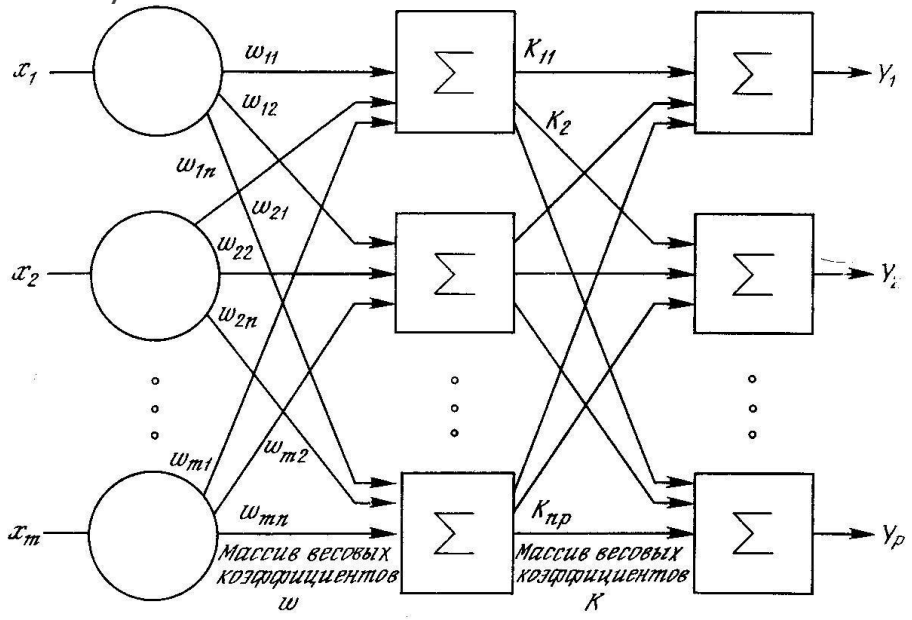


История нейросети.

- Идею нейронных сетей впервые предложили исследователи из Чикагского университета Уоррен Маккалоу и Уолтер Питтс в 1944 году. Первую обучаемую нейросеть в 1957 году продемонстрировал психолог Корнеллского университета Фрэнк Розенблатт. Она была примитивной



- В 1980-х годах, когда появились более мощные компьютеры для вычислений, исследователи смогли разработать нейросети с двумя и тремя уровнями обучения.



История нейросети.

- Однако возрождение интереса к нейронным сетям и революция в глубоком обучении произошли лишь в последние годы благодаря индустрии компьютерных игр.
  Современные игры требуют сложных вычислений для обработки большого числа операций. В итоге производители начали выпускать графические процессоры (GPU), которые объединяют тысячи относительно простых вычислительных ядер на одном чипе.
  Современные графические процессоры позволили развивать «глубокое обучение» — повышать глубину слоев нейросети. Именно благодаря ему появились самообучаемые нейросети, которые не требуют настройки, а самостоятельно обрабатывают входящую информацию.

Типы нейросети.

- В зависимости от архитектуры нейросети делятся на типы:
  1. Прямого распространения — обрабатывают входные данные и сразу выдают результат. Чаще всего применяются для распознавания образов и текста, а также классификации данных;
  2. Рекуррентные — перенаправляют информацию туда и обратно по слоям, пока не получат конечный результат. Этот тип обычно используется для прогнозирования;
  3. Сверточные — обрабатывают каждый признак в отдельном слое. Такой тип применяется в классификации изображений, обработке языка и т.д.
- 4. Глубокие — имеют многослойную структуру, используются для решения сложных задач;
  5. Автоэнкодеры — обучаются извлекать наиболее важные признаки из входных данных, полезны для снижения размерности информации и улучшения качества изображений.
  Помимо основных типов встречаются десятки подтипов нейросетей. Например, модульные — это, по сути, совокупность нейросетей, которые работают независимо друг от друга, чтобы ускорить вычисления.

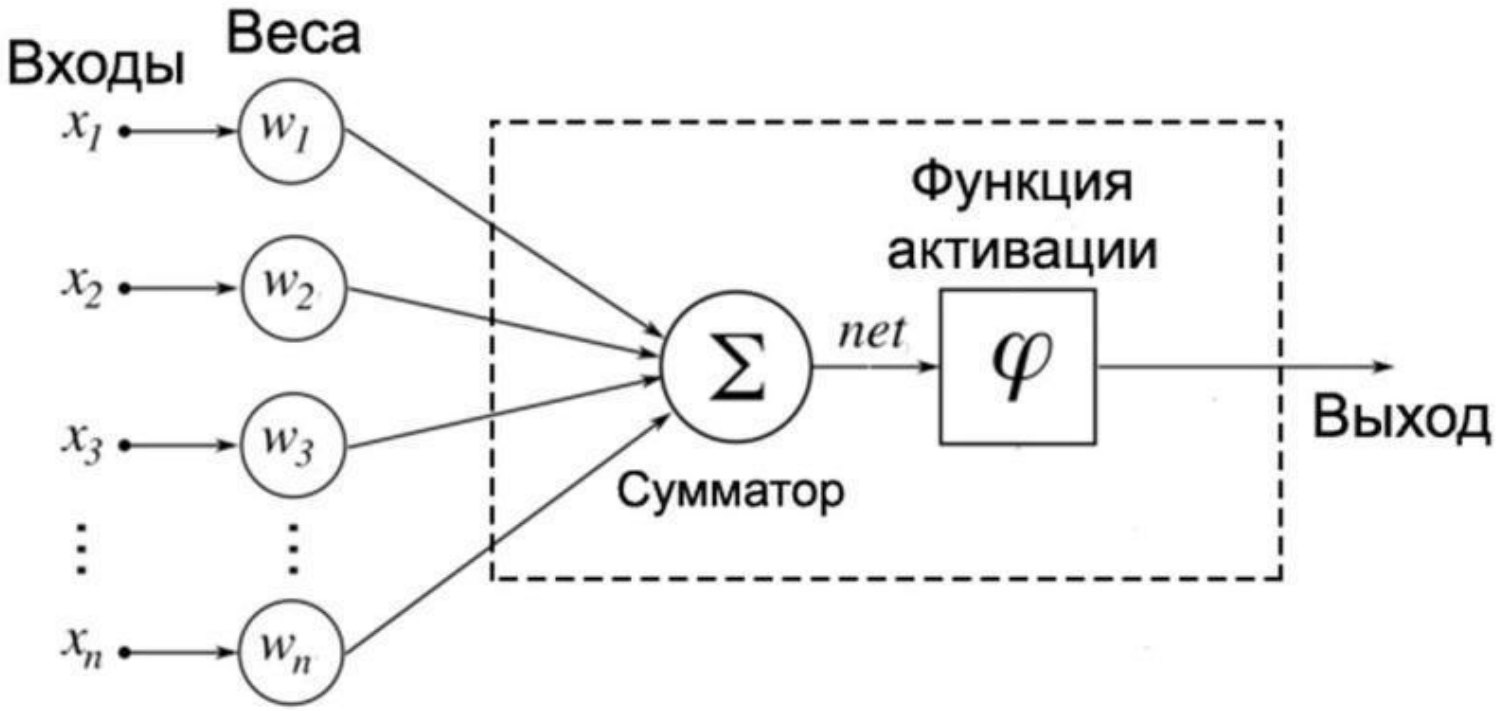
Типы нейросети.

- Так же выделяются нейросети-художники, генеративные и самообучающиеся системы. Список расширяется, но каждый вид индивидуален и имеет свои параметры. Расскажу немного об этих трёх.
- Нейросеть-художник — это искусственный интеллект, способный создавать изображения после обучения на наборе данных модели. Используют генеративные алгоритмы.
  Генеративная нейросеть — используется при разработке новых данных: изображения, видео, музыка, тексты. Работают на базе глубокого обучения для создания уникальных результатов.
  Самообучающаяся нейросеть — способна обучаться на входных данных без ручной настройки параметров модели. Использует методы глубокого обучения, такие как обратное распространение ошибки, чтобы обновлять веса нейронов, улучшая качество.

Как работает нейросеть?

- Базовая нейронная сеть содержит три слоя искусственных нейронов:
1. Входной — обрабатывает информацию извне, анализирует или классифицирует ее и передает на следующий слой;
2. Скрытый (их может быть несколько) — анализирует выходные данные предыдущего слоя, обрабатывает их и передает на следующий;
3. Выходной — выдает окончательный результат после обработки всех данных.
Глубокие нейронные сети отличаются тем, что искусственные нейроны в них связаны друг с другом, а каждой такой связи присваивается определенный вес, который отражает ее значимость. Кроме того, связь между нейронами может быть «упреждающей». Это означает, что данные проходят через них только в одном направлении. Такое происходит, если значение «веса» соединения ниже заданного.
При обучении нейронной сети все ее «веса» изначально задаются случайными значениями. Обучающие данные подаются на нижний, или входной, слой. Затем они проходят через последующие слои, пока не достигают выходного. Во время обучения «веса» и пороговые значения постоянно корректируются до тех пор, пока данные обучения не будут постоянно давать одинаковые результаты.
Эти «веса» помогают определить важность той или иной переменной во входных данных. При прохождении каждого слоя входные данные умножаются на их «веса», а затем суммируются. Если получившееся значение выше заданного порога, то нейрон активируется и передает данные на следующий уровень.

Как работает нейросеть?



Как работает нейросеть?

- Попросим нейросеть ответить на вопрос, стоит ли заняться серфингом (варианты ответа «да» — 1 и «нет» — 0.
- Предположим, что на это решение (y-hat) влияют три фактора. Сформулируем их в виде вопросов:

  1. Хорошие ли волны? («да» — 1 и «нет» — 0);
  2. Свободен ли пляж? («да» — 1 и «нет» — 0);
  3. Фиксировались ли в последнее время нападения акул? («да» — 0 и «нет» — 1).

- Затем предположим следующее, используя входные данные:

  X1 = 1, так как волны есть;

  X2 = 0, так как на пляже нет толпы;

  X3 = 1, так как нападений акул не фиксировалось.

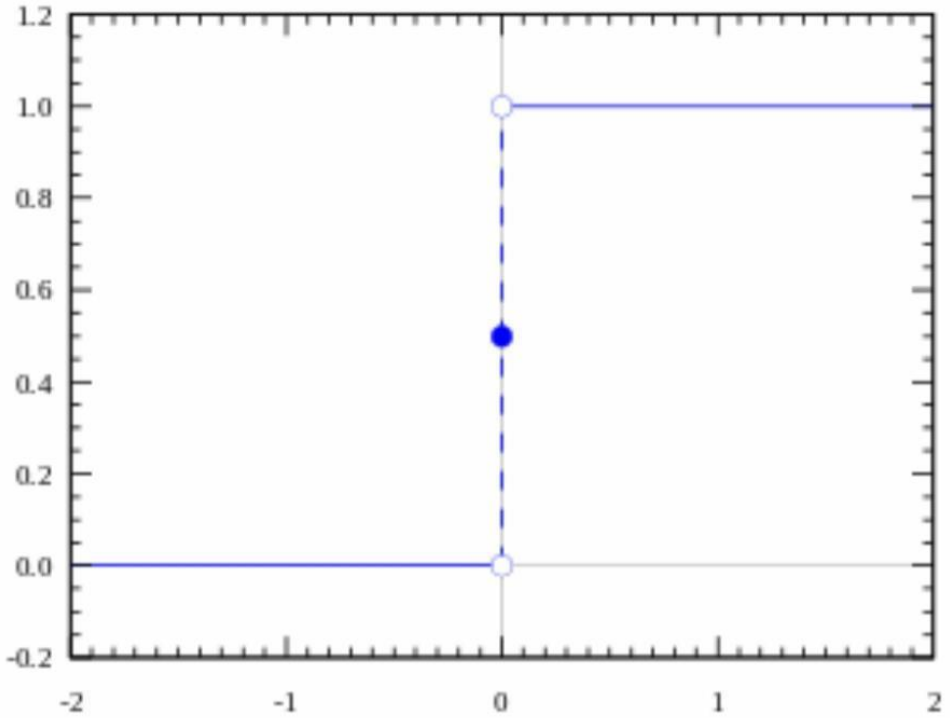
- Теперь нам нужно присвоить этим значениям некоторые «веса», чтобы определить их важность.

Как работает нейросеть?

- Расположим приоритеты следующим образом:
- W1 = 5, так как большие волны на этом пляже возникают нечасто;
  W2 = 2, так как вы не боитесь толпы;
  W3 = 4, так как вы боитесь акул.
  За пороговое значение мы примем 3, соответственно, значение смещения составит –3.
  Теперь можно начать подставлять значения в формулу, чтобы получить желаемый результат:
- Y-hat = (1*5) + (0*2) + (1*4) – 3 = 6.
- Поскольку полученное значение выше 3, то решение о серфинге будет положительным.

Нейрон и его активация.

- Нейрон — это функция, которая получает на вход числа (обычно много), преобразовывает и возвращает одно число (обычно от 0 до 1). Функция, из которой нейрон состоит, называется функцией активации, или передаточной функцией. Именно она отвечает за то, будет ли передан сигнал нейрона дальше (1 на выходе функции — если будет, 0 — если нет). В самом простом виде функция активации может быть пороговой (внимание на фотографию).
  В данном случае, если определенные значения на входе нейрона дали положительную сумму — нейрон активировался. Порог активации здесь выглядит как вертикальная палочка, простой переключатель вкл/выкл, а значит, нейрон может быть либо активен, либо нет — как лампочка.

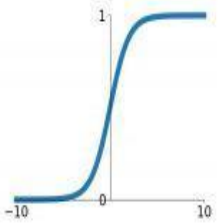


Нейрон и его активация.

- Есть и более сложные функции, которые могут изменять силу выходного сигнала, могут «чуть-чуть активировать» нейрон, или «активировать его посильнее»:
  Выбор функции активации нейронов влияет на то, какие задачи они могут решать, и их подбор — задача создателя нейросети. Если создатель понимает, какие функции для чего подходят, у него больше шансов сделать крутую нейросеть, которая решит его проблему.
  Главное: нейрон — это функция. Чем больше ее значение, тем важнее сигнал нейрона для всей сети.

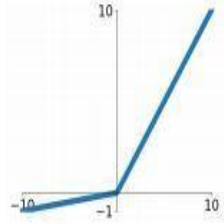
Sigmoid

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



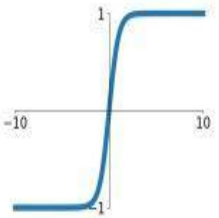
Leaky ReLU

$$\max(0.1x, x)$$



tanh

$$\tanh(x)$$

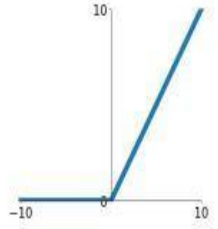


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

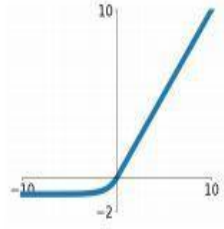
ReLU

$$\max(0, x)$$



ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Функция потерь.

- Есть функция, возрастающая при сильной ошибке предсказания. Чем больше ошибка, тем сильнее уменьшаются (штрафуются) веса активированных нейронов. Человеку нужно свести меру ошибки к минимуму, научить нейросеть делать верные предсказания о том, котик на картинке или булочка. Если функция потерь дает высокие значения при большой ошибке, мы просто ищем такие веса, при которых она дает минимальные значения.
  Кстати, функций потерь тоже существует много, они по-разному подходят для разных задач, но функции потерь объединяет то, что они возрастают вместе с мерой ошибки предсказания. Она вычисляется просто: из числа X (предсказанного) вычитаем число Y (то, что должно было предсказаться) — получаем число Err. Оно становится одним из аргументов функции потерь.

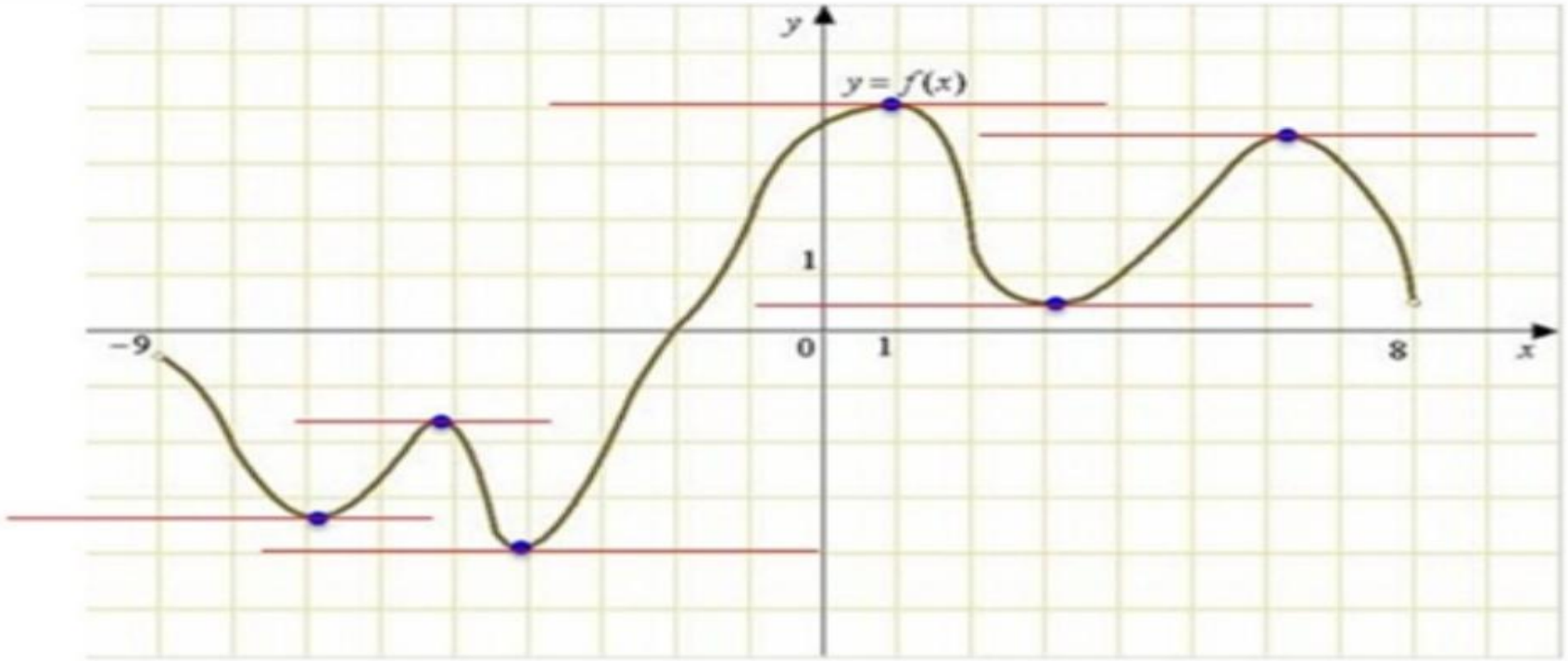
Схема обучения нейросети.

- Задача подбора весов нейронной сети также называется тренировкой или задачей оптимизации. Веса нам нужны те, при которых функция потерь принимает минимальное значение. Когда мы знаем это значение, рассчитать нужные веса становится легко — все равно, что решить уравнение.
  А значит, желаемая схема обучения нейросети выглядит примерно так:
  Функция потерь принимает минимальное значение → находим соответствующие этому значению веса → ошибка минимальна → предсказание нейросети точно → вы великолепны.
  Только сначала нужно найти минимальное значение функции.

Вариант нахождения минимума.

- Вот на паре по математике нас Александра Алексеевна дает какой-нибудь многочлен и просит найти его минимум, что мы делам?
  Находим производную функции и приравниваем ее к нулю;
  Решаем получившееся уравнение, то есть находили, при каких значениях переменной производная равна нулю;
  Подставляем эти значения в функцию и проверяем, где получится минимальное значение.
  Напомним, что производная функции выражает скорость ее возрастания или убывания. Её геометрический смысл — показать угол, под которым в данной точке проходит касательная к функции. Если производную (а значит, угол касательной) приравнять к нулю, то мы найдем точки, в которых касательная параллельна оси абсцисс.
  Вот что это будут за точки: в них функция меняет направление. Такие точки обязательно становятся (хотя бы локальными) минимумами или максимумами функции. А вот как будут выглядеть касательные к этим точкам:

Точки, в которых касательная параллельна оси абсцисс.



Метод градиентного спуска.

- После этого страшного математического момента пора признаваться, что как раз вот так в нейронных сетях найти минимум функции потерь невозможно. У нее слишком много входных аргументов, и аналитические подходы (как показанный выше) не работают. Приходится применять специальный поисковой, пошаговый метод градиентного спуска
  Вот главное: нам до сих пор нужно найти минимальное значение функции потерь, это минимизирует ошибку в предсказании нейросети. Но мы сейчас выяснили, что по-школьному это сделать не получится.

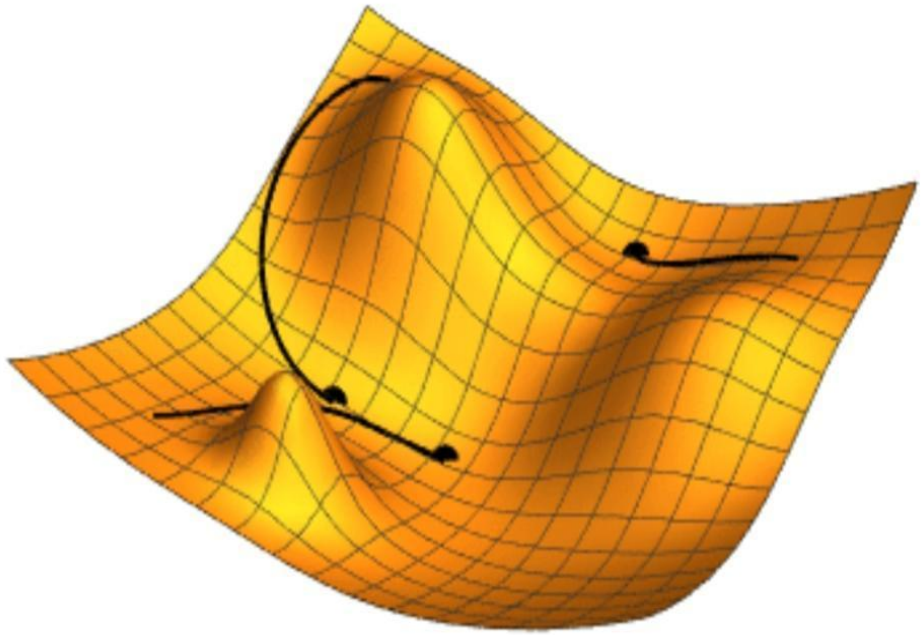
Что такое сам градиент?

- Если для функции многих переменных по очереди рассчитать частные производные для каждой из переменных (и записать эти производные в ряд, и взять в скобочки), получится вектор, называемый градиентом. Так как одна переменная соответствует одной координатной оси, каждый элемент вектора показывает скорость изменения функции вдоль своей оси, а все вместе они показывают направление, в котором быстрее всего возрастает функция в целом.
  Вот главное: градиент — это вектор, показывающий направление, в котором функция многих переменных быстрее всего возрастает или падает. «Градиент функции f» записывается так: ∇f (можно читать как «набла f»).

$$\nabla f = \left(\frac{\partial f}{\partial x} \; \frac{\partial f}{\partial y} \; \frac{\partial f}{\partial z}\right) = (2y^3, 6xy^2, 6z)$$

Градиентный спуск.

- Градиентный спуск — это эвристический алгоритм, который выбирает случайную точку, рассчитывает направление скорейшего убывания функции (пользуясь градиентом функции в данной точке), а затем пошагово рассчитывает новые значения функции, двигаясь в выбранную сторону. Если убывание значения функции становится слишком медленным, алгоритм останавливается и говорит, что нашел минимум.
  Стоит помнить, что говорим не о двумерных функциях. Представить, что происходит, можно по этой картинке:



Градиентный спуск.

- Градиентный спуск выбирает случайную точку, находит направление самого быстрого убывания функции и двигается до ближайшего минимума вдоль этого направления. Кстати, размер одного шага можно настроить, это бывает очень важно. Если на каком-то этапе разность между старой точкой (до шага) и новой снижается ниже предела, считается, что минимум найден, алгоритм завершен. Можно вообразить работу градиентного спуска как игру в «холодно-горячо» до тех пор, пока степень «потепления» не станет пренебрежительно малой.
  Как это работает:
  Градиентный спуск выбирает случайную точку старая_точка, принадлежащую функции
  Новая_точка рассчитывается как старая_точка минус (размер_шага * градиент_в_старой_точке)
  ЕСЛИ модуль (новая_точка минус старая_точка) < порогового_предела
  ТО старая_точка = минимум функции
  ИНАЧЕ — повторить алгоритм для новая_точка

Размер шага.

- Размер шага алгоритма определяет, насколько мы собираемся двигать точку на функции потерь, и этот параметр называется «скоростью обучения». Слегка запутывающее название, поскольку не всегда высокая скорость обучения гарантирует хороший результат. Скорее скорость обучения стоит воспринимать как ширину шагов, с которыми человек с завязанными глазами ищет, где «горячо» или «холодно». В некоторых случаях бывает так, что слишком широкие шаги вообще не позволяют достичь минимума, и машина бесконечно перешагивает через него, затем градиент «разворачивает» ее обратно, и алгоритм снова перескакивает через минимум. Маленькая скорость обучения хоть и придает точности, зато, конечно, увеличивает время на обучение нейросети.

Градиентый спуск. Вывод.

- Градиентный спуск ищет ближайшую к случайно выбранной точке впадину на графике функции. А поскольку в нейросетях функции очень сложные и локальных впадин-минимумов на них много, такой подход должен быть неэффективен в вопросах обучения нейросети и всегда натыкаться на локальные минимумы.
  Тем не менее градиентный спуск как метод обучения почему-то работает хорошо. В 2015 группа ученых из Курантовского института математических наук в Нью-Йорке нашла этому объяснение, показав, что большая часть локальных минимумов функций потерь, используемых в нейросетях, располагается близко к глобальному минимуму. Эта близость и позволяет натренированным при помощи градиентного спуска нейросетям справляться с задачами достаточно эффективно.

Спасибо за внимание!

- Источники:
- https://blog.sf.education/kak-rabotaet-neiroset/
- https://trends.rbc.ru/trends/industry/641157be9a7947d3401fa3e8
  https://sysblok.ru/knowhow/razbiraem-nejroseti-po-chastjam-kak-rabotaet-gradientnyj-spusk/