

Функционалы ошибки и поиск оптимальных параметров в задачах машинного обучения

Урок 5

На этой лекции вы найдете ответы на такие вопросы как:

- Что такое функция потерь
- Функции потерь регрессии и классификации
- Узнаем, как обучается модель
- Градиентный спуск и его модификации



Булгакова Татьяна

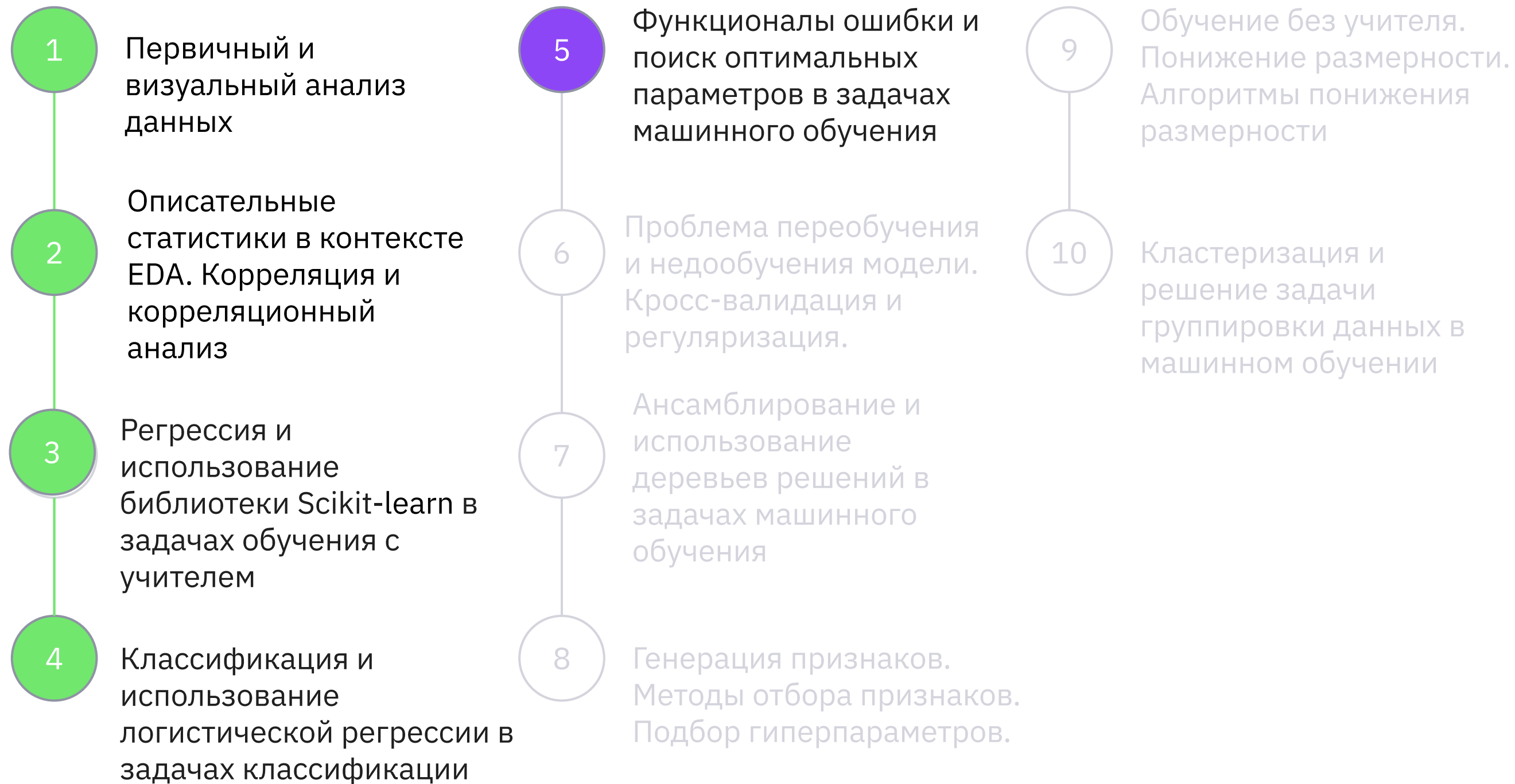
Преподаватель в GeekBrains, Нетология, Skillfactory

С 2010 года занимаюсь DataScience и NN. Фрилансер

- Участвовала в разработке программы по настройке оборудования для исследования пространственного слуха китообразных НИИ ИПЭЭ РАН
- Участвую в разработке рекомендательных систем по настройке нейростимуляторов для медицинских центров
- Работаю над курсом по нейронным сетям







План курса





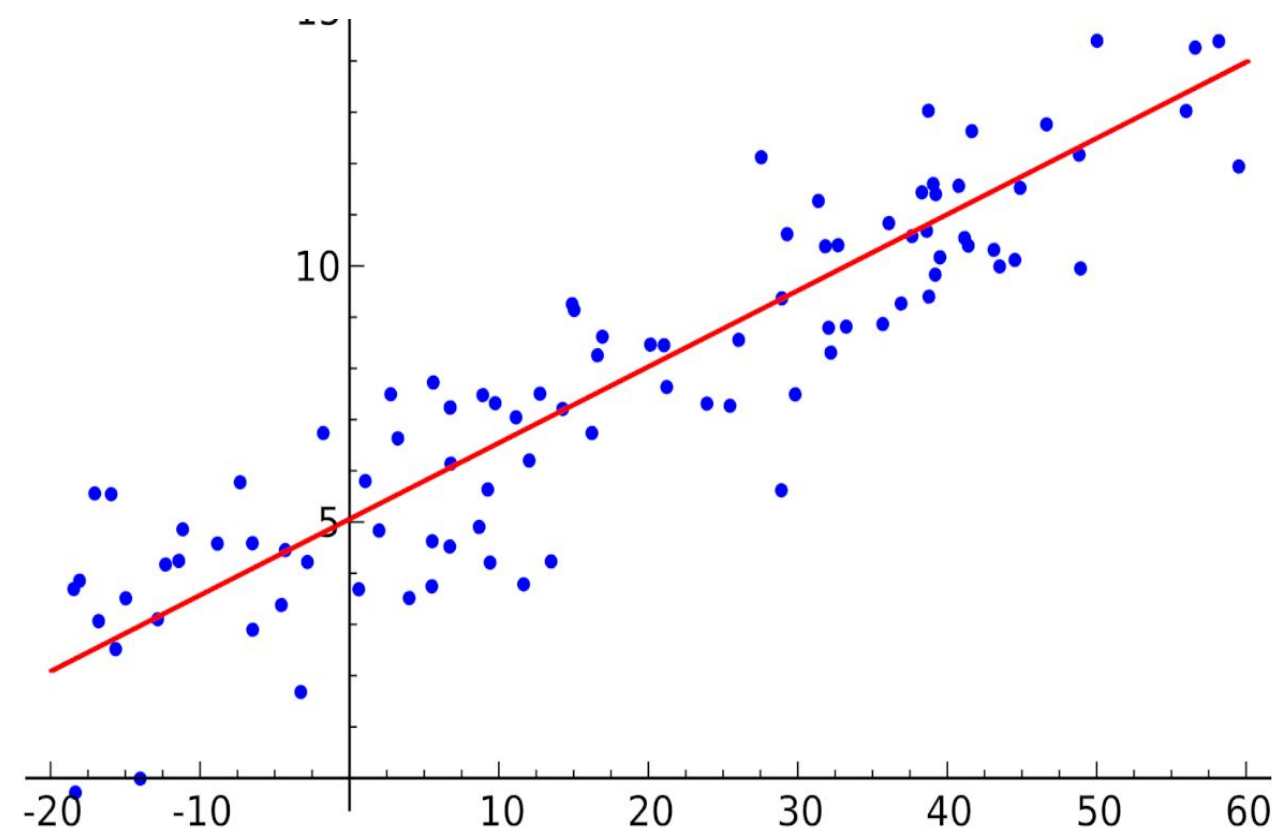
Что будет на уроке сегодня

-  Что такое функция потерь
-  Функции потерь регрессии и классификации
-  Узнаем, как обучается модель
-  Градиентный спуск и его модификации



Основные понятия

В процессе обучения модель получает на вход некоторый набор данных, и ее задача - извлечь из этого набора

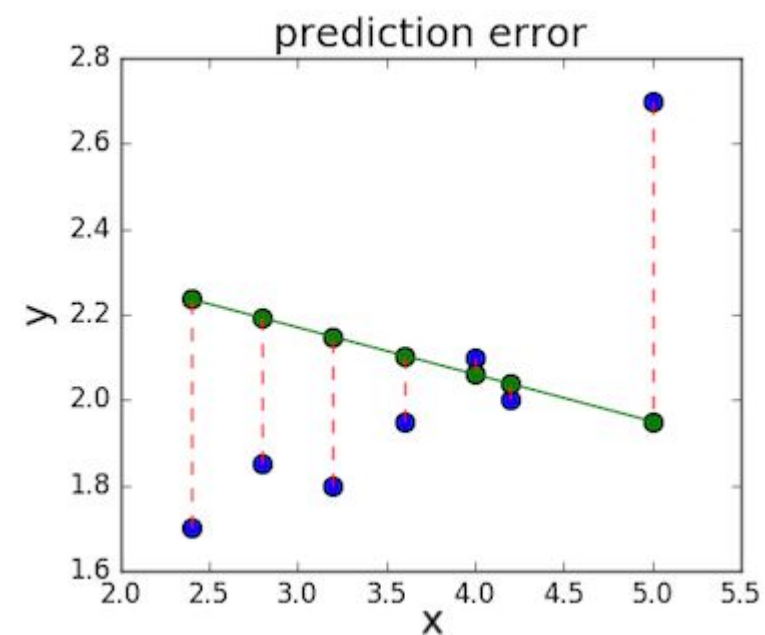


Линейная регрессия относится к задаче определения "наилучшего соответствия прямой линии" по набору точек данных и является простым предшественником нелинейных методов, используемых для обучения нейронных сетей.



Функция потерь

Функция потерь - это мера количества ошибок, допущенных линейной регрессией на наборе данных. Существует несколько функций потерь, каждая из которых рассчитывает расстояние между прогнозируемым и фактическим значениями $y(x)$





Функция потерь

Посмотрим на примере самой распространенной функции потерь - средней квадратичной ошибки (MSE)

$$\vec{y} = X\vec{w} + \epsilon$$

$$y_i = \sum_{j=0}^m w_j X_{ij} + \epsilon_i$$

Также есть определенные ограничения на модель, иначе это будет другой тип регрессии, а не линейная:

1. Математическое ожидание случайных ошибок равно нулю

$$\forall i : \mathbb{E} [\epsilon_i] = 0$$

2. Дисперсия случайных ошибок одинакова и конечна, это свойство называется гомоскедастичностью:

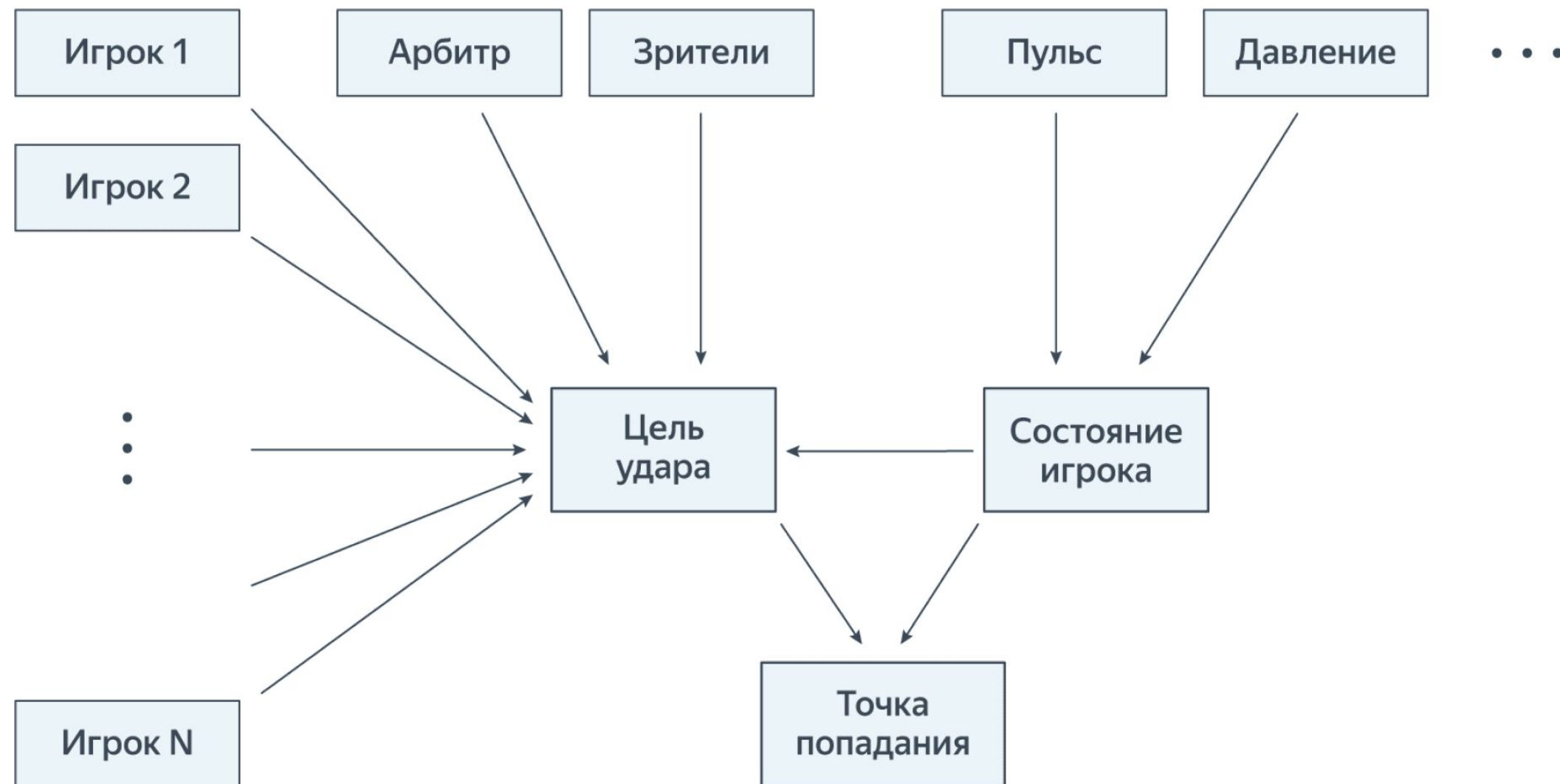
$$\forall i : \text{Var} (\epsilon_i) = \sigma^2$$

3. Случайные ошибки не связаны между собой:

$$\forall i \neq j : \text{Cov} (\epsilon_i, \epsilon_j) = 0.$$



Функция потерь





Функция потерь

$$\begin{aligned}
 p\left(\begin{matrix} \text{точка} \\ \text{попадания} \end{matrix}\right) &= \int p\left(\begin{matrix} \text{точка} & \text{цель} & \text{состояние} \\ \text{попадания} & \text{удара} & \text{игрока} \end{matrix}\right) d\left(\begin{matrix} \text{цель} \\ \text{удара} \end{matrix}\right) d\left(\begin{matrix} \text{состояние} \\ \text{игрока} \end{matrix}\right) = \\
 &= \int p\left(\begin{matrix} \text{точка} \\ \text{попадания} \end{matrix} \middle| \begin{matrix} \text{цель} & \text{состояние} \\ \text{удара} & \text{игрока} \end{matrix}\right) p\left(\begin{matrix} \text{цель} & \text{состояние} \\ \text{удара} & \text{игрока} \end{matrix}\right) d\left(\begin{matrix} \text{цель} \\ \text{удара} \end{matrix}\right) d\left(\begin{matrix} \text{состояние} \\ \text{игрока} \end{matrix}\right) = \\
 &= \int p\left(\begin{matrix} \text{точка} \\ \text{попадания} \end{matrix} \middle| \begin{matrix} \text{цель} & \text{состояние} \\ \text{удара} & \text{игрока} \end{matrix}\right) p\left(\begin{matrix} \text{цель} \\ \text{удара} \end{matrix} \middle| \begin{matrix} \text{состояние ...} \\ \text{игрок 1, игрок 2} \\ \text{арбитр, зрители} \end{matrix}\right) p\left(\begin{matrix} \text{состояние} \\ \text{игрока} \end{matrix} \middle| \begin{matrix} \text{пульс} \\ \text{давление} \end{matrix}\right) \times \\
 &\times \left[\prod_i p(\text{игрок } i) \right] p(\text{арбитр}) p(\text{зрители}) p(\text{пульс}) p(\text{давление}) \times \\
 &\times \left[\prod_i d(\text{игрок } i) \right] d(\text{арбитр}) d(\text{зрители}) d(\text{пульс}) d(\text{давление})
 \end{aligned}$$



Метод максимального правдоподобия

Процесс оценки параметров с помощью метода максимального правдоподобия включает следующие шаги:

1. Формулирование статистической модели, которая описывает вероятностное распределение наблюдаемых данных в зависимости от неизвестных параметров.
2. Запись функции правдоподобия, которая показывает вероятность получения наблюдаемых данных при заданных значениях параметров.
3. Максимизация функции правдоподобия путем нахождения таких значений параметров, при которых функция достигает своего максимального значения.
4. Вычисление оценок параметров на основе найденных значений, и их интерпретация.



Метод максимального правдоподобия

Итак, мы стремимся найти значения параметров w , при которых модель $p(y|x,w)$ будет наиболее соответствовать обучающим данным.

$$\hat{w}_{MLE} = \underset{w}{\operatorname{argmax}} p(y|X, w)$$

Величина $p(y|x,w)$ называется функцией правдоподобия (likelihood). Если мы считаем, что все объекты независимы, то функция правдоподобия распадается в произведение:

$$p(y|X, w) = p(y_1|x_1, w) \cdot \dots \cdot p(y_i|x_i, w)$$

Теперь мы применяем логарифм функции правдоподобия, так как умножение сложно, а сложение легко.

$$l(y|X, w) = \log p(y_1|x_1, w) + \dots + \log p(y_i|x_i, w)$$



Метод максимального правдоподобия

Как мы уже обсуждали

$$p(y_i|x_i, w) = p_\varepsilon(y - f_w(x_i))$$

$$l(y|X, w) = \sum_{i=1}^N \log p_\varepsilon(y_i - f_w(x_i))$$

Максимизация функции правдоподобия соответствует минимизации

$$\sum_{i=1}^N [-\log p_\varepsilon(y_i - f_w(x_i))]$$

Это выражение можно трактовать как функцию потерь. Таким образом, можно сказать, что подбор параметров модели вероятностей с использованием метода максимального правдоподобия является эквивалентом "инженерной" оптимизации функции потерь.



Метод максимального правдоподобия

Два важных аспекта: сходимость и эффективность.

Сходимость. При увеличении числа обучающих выборок к бесконечности, оценка максимального правдоподобия стремится к истинному значению параметра.

Эффективность. Оценка того, насколько близки мы к истинному параметру, осуществляется через ожидаемую среднеквадратичную ошибку, которая вычисляется как квадратичная разность между оценками и истинными значениями параметров. Для этого вычисляется математическое ожидание по m обучающим выборкам из данных, которые генерируют распределение.



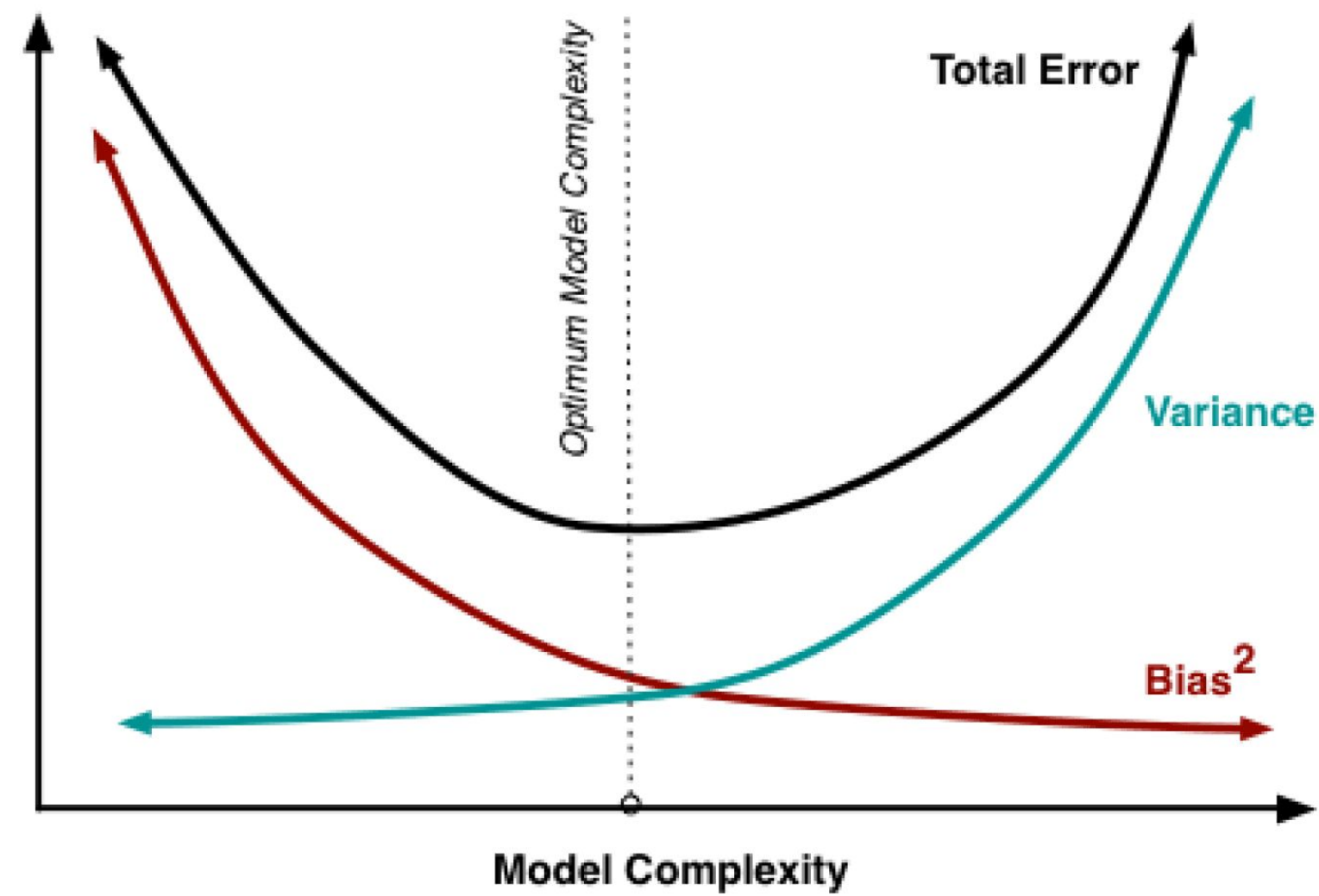
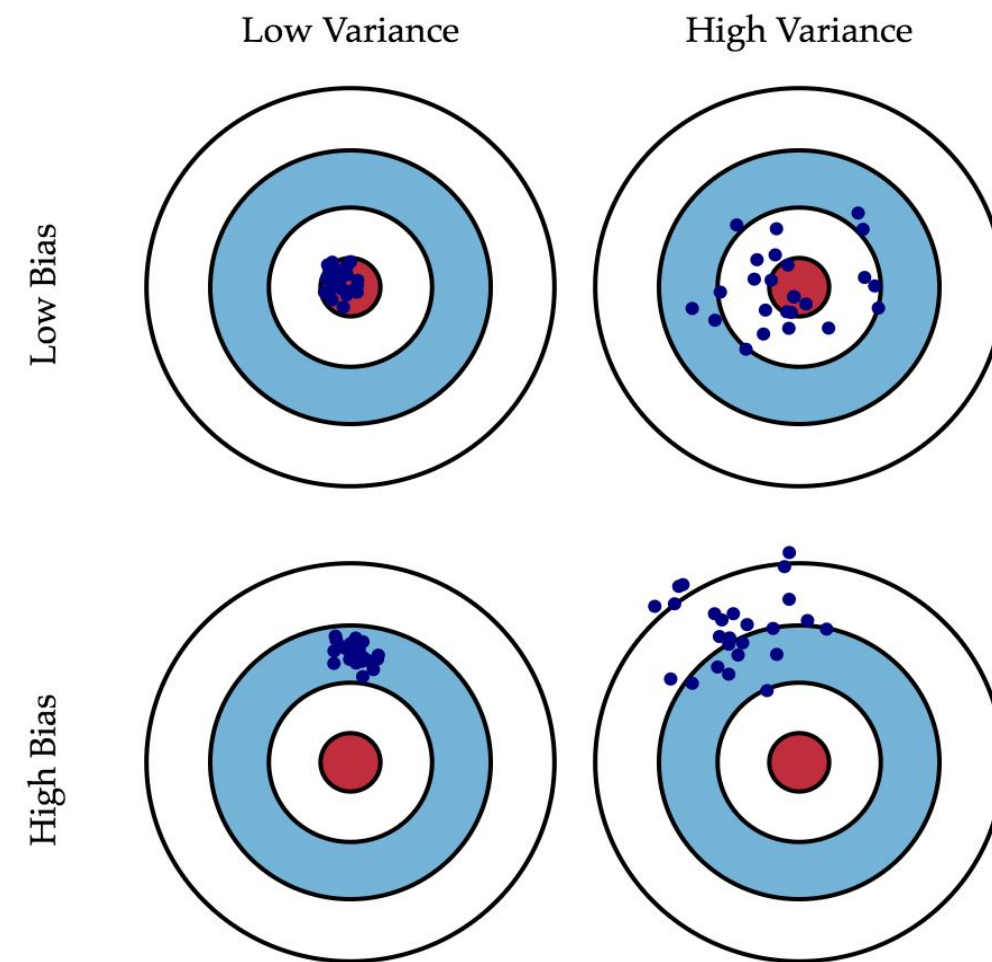
Разложение ошибки на смещение и разброс

Дисперсию ответов алгоритмов мы называем разбросом, а матожидание разности между истинным ответом и выданным алгоритмом - смещением (bias). Ошибка разбивается на три компоненты.

- ➡ Первая связана с шумом в данных
- ➡ Разброс характеризует разнообразие алгоритмов
- ➡ Смещение - способность модели алгоритмов адаптироваться к целевой зависимости.



Разложение ошибки на смещение и разброс





Функция потерь и метрика качества имеют важное различие:

- Суть функции потерь заключается в том, что мы сводим задачу построения модели к задаче оптимизации. Обычно требуется, чтобы функция потерь обладала хорошими свойствами, такими как дифференцируемость.
- Метрика качества, с другой стороны, является внешним и объективным критерием, который зависит не от параметров модели, а только от предсказанных меток.



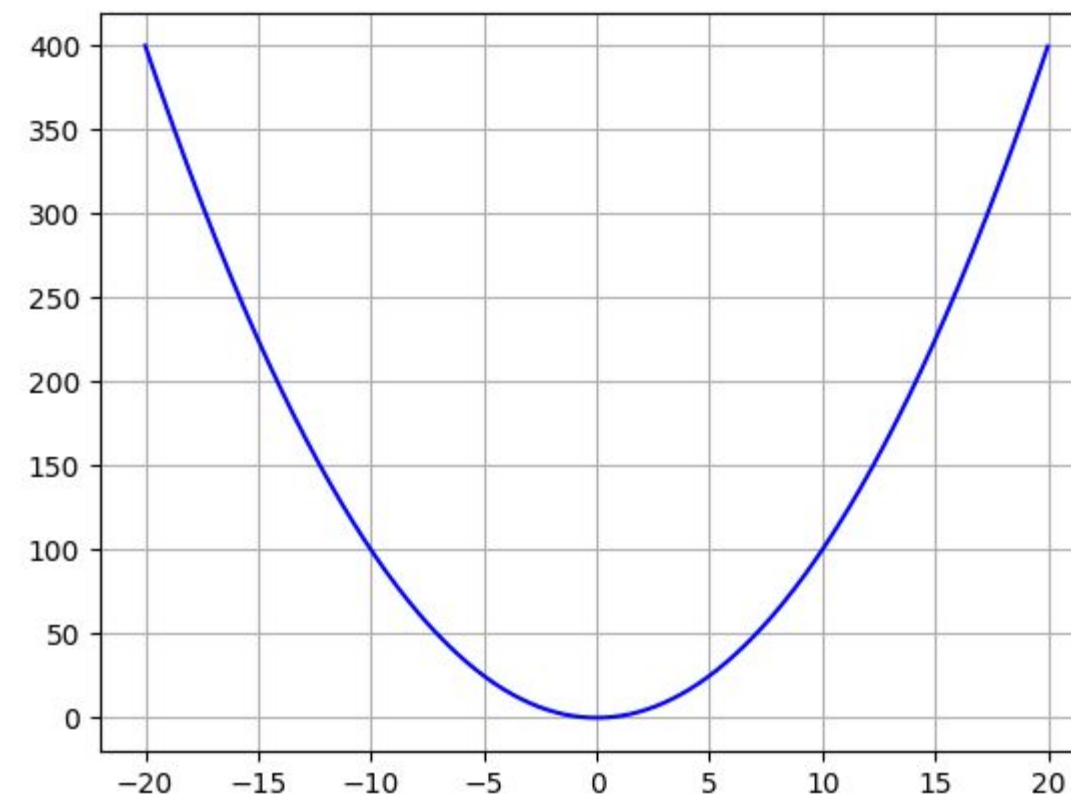
MSE

В некоторых случаях метрика может совпадать с функцией потерь. Например, в задаче регрессии MSE играет роль как функции потерь, так и метрики.

$$L(f, X, y) = \frac{1}{N} \sum_{i=1}^N (y_i - \langle x_i, w \rangle)^2$$

$$MSE(f, X, y) = \frac{1}{N} \|y - Xw\|_2^2$$

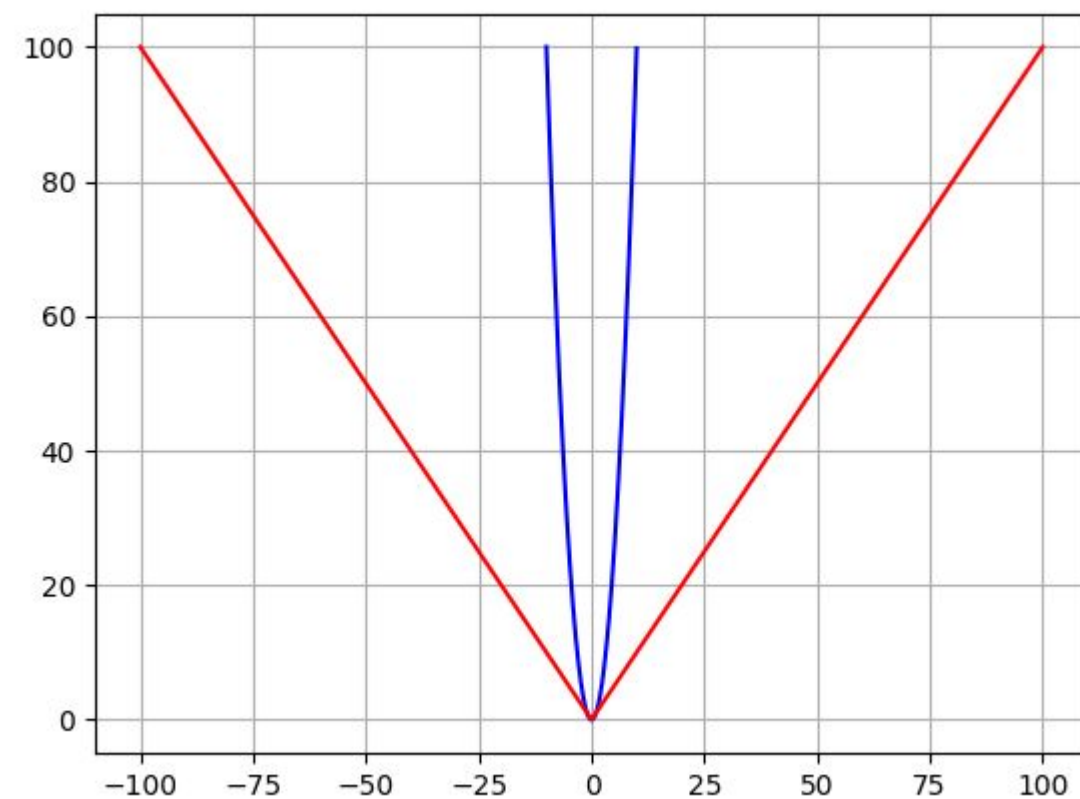
Данная функция потерь называется Mean Squared Error, MSE или среднеквадратическим отклонением. Разница с L^2 - нормой чисто косметическая, на алгоритм решения задачи она не влияет





MAE

Средняя абсолютная ошибка (MAE) представляет собой среднее значение абсолютных отклонений между предсказаниями модели и фактическими значениями данных. Она вычисляется путем нахождения разницы между предсказаниями и истинными значениями, применения абсолютного значения к этой разнице и усреднения полученных значений по всему набору данных

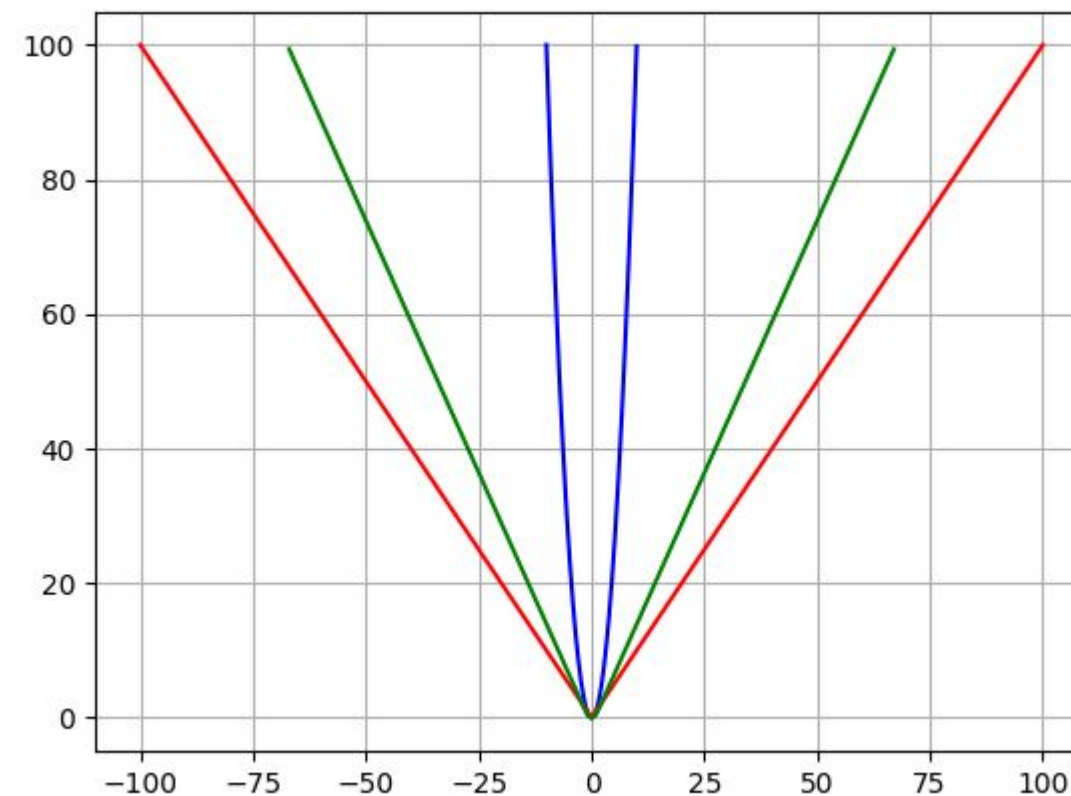




Функция Huber Loss

Предлагает компромиссное решение между MSE и MAE. Мы можем определить ее с помощью следующего кусочного уравнения:

В основном, это уравнение гласит: для значений потерь, меньших чем дельта, используйте MSE; для значений потерь, больших чем дельта, используйте MAE. Таким образом, эта функция эффективно объединяет лучшие стороны обеих функций потерь!





Binary Cross-Entropy

Задача функции потерь в классификации заключается в определении стоимости неточности предсказаний классификационной модели. Для выполнения бинарной классификации с использованием логистической регрессии, функция потерь может быть рассчитана следующим образом

$$LogLoss = -\frac{1}{l} \sum_{i=1}^l (y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i))$$

где l - размер выборки, $y_i = 0, 1$ - бинарная метка класса, заданная в примере, y .



Cross-Entropy

У нас стоит задача мультиклассовой классификации, где количество классов больше двух и они являются взаимоисключающими, то невозможно научиться классифицировать каждый класс отдельно и выбирать класс с максимальной вероятностью

Нам надо обобщить бинарную кросс-энтропию до категориальной, которая также называется просто кросс-энтропией. Для этого мы заменим несколько сигмoids (по одной для каждого класса) на softmax

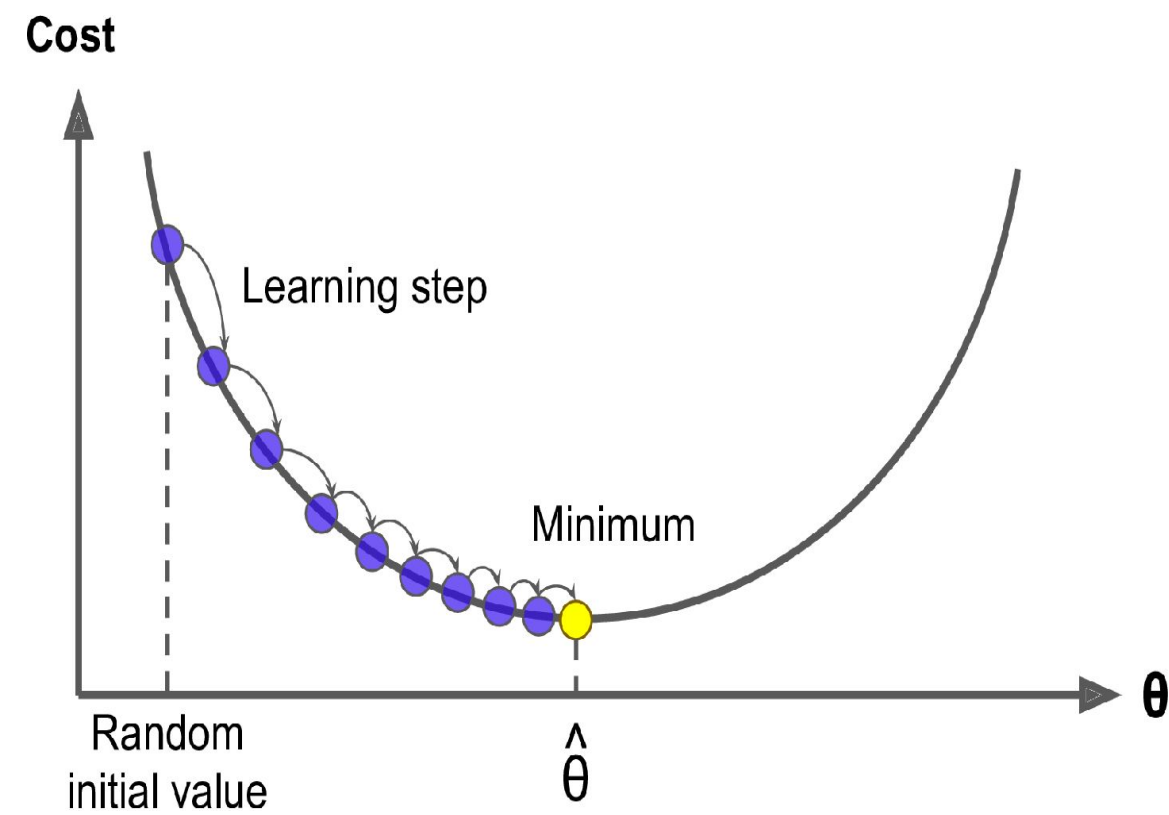
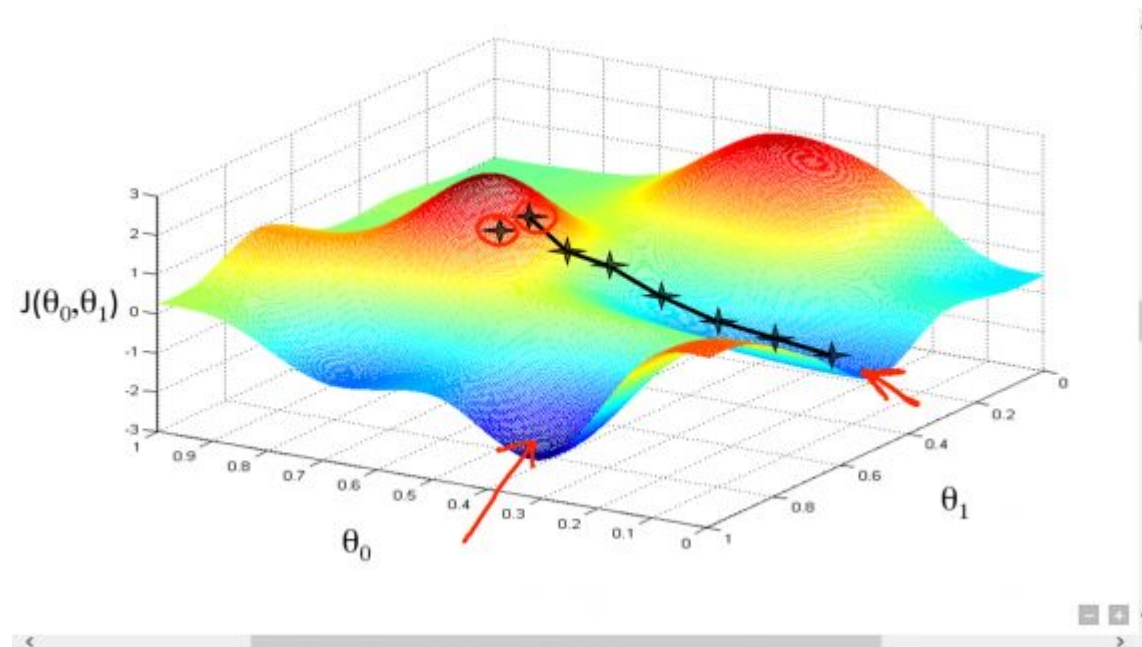
Теперь мы можем вычислять потери только для положительного класса, так как остальные выходы участвуют в функции софтмакс и через них проходит поток градиента. После обучения, при использовании модели, мы можем просто выбирать максимальное значение из всех вероятностей.

$$CE(p_t) = -\log(p_t)$$



Градиентный спуск

Градиентный спуск — это метод поиска минимального значения функции потерь.





О градиенте

Помимо функции потерь, для метода градиентного спуска необходим градиент dJ/dw (производная функции потерь по одному весу, выполняемая для всех весов). dJ/dw зависит от выбора функции потерь. Наиболее распространенной функцией потерь является функция потерь среднеквадратичной ошибки.

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Производная этой функции относительно любого веса (эта формула показывает вычисление градиента для линейной регрессии):

$$\frac{\partial}{\partial \theta_j} \text{MSE}(\theta) = \frac{2}{m} \sum_{i=1}^m (\theta^T \cdot \mathbf{x}^{(i)} - y^{(i)}) x_j^{(i)}$$

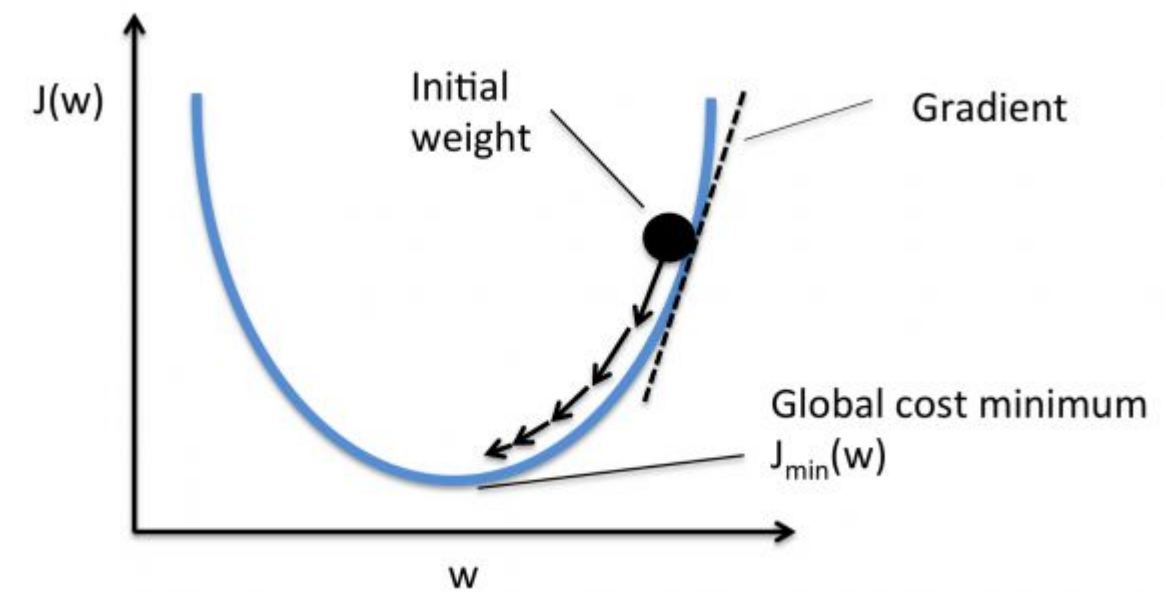


Коэффициент скорости обучения

Repeat until convergence {

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}





Градиентный спуск. Обобщение

После того как вы перебрали все примеры обучения, выполните следующие шаги:

1. Разделите аккумулятивные переменные весов и смещений на количество примеров обучения. Таким образом, вы получите средние градиенты для всех весов и средний градиент для смещения. Назовем их обновленными аккумуляторами (ОА).
2. Затем, используя следующую формулу, обновите все веса и смещение. Вместо $dJ / d\theta_j$ подставьте обновленные аккумуляторы для весов и смещения. То же самое сделайте для смещения.

Это только одна итерация градиентного спуска.

Повторите этот процесс снова и снова для заданного количества итераций. Это означает, что для первой итерации градиентного спуска вы перебираете все примеры обучения, вычисляете градиенты, затем обновляете веса и смещения. Затем повторяете это для заданного количества итераций градиентного спуска.



Стохастический градиентный спуск

Вместо того чтобы перебирать и использовать весь набор примеров обучения, мы применяем подход "используй только один". Здесь есть несколько моментов, которые следует отметить:

1. Необходимо перемешать набор примеров обучения перед каждым проходом в ГС, чтобы каждый раз перебирать их в случайном порядке. С каждой итерацией ГС необходимо перемешивать набор обучения и выбирать случайный пример обучения.
2. Поскольку каждый раз используется только один пример обучения, ваш путь к локальному минимуму будет неоптимальным.

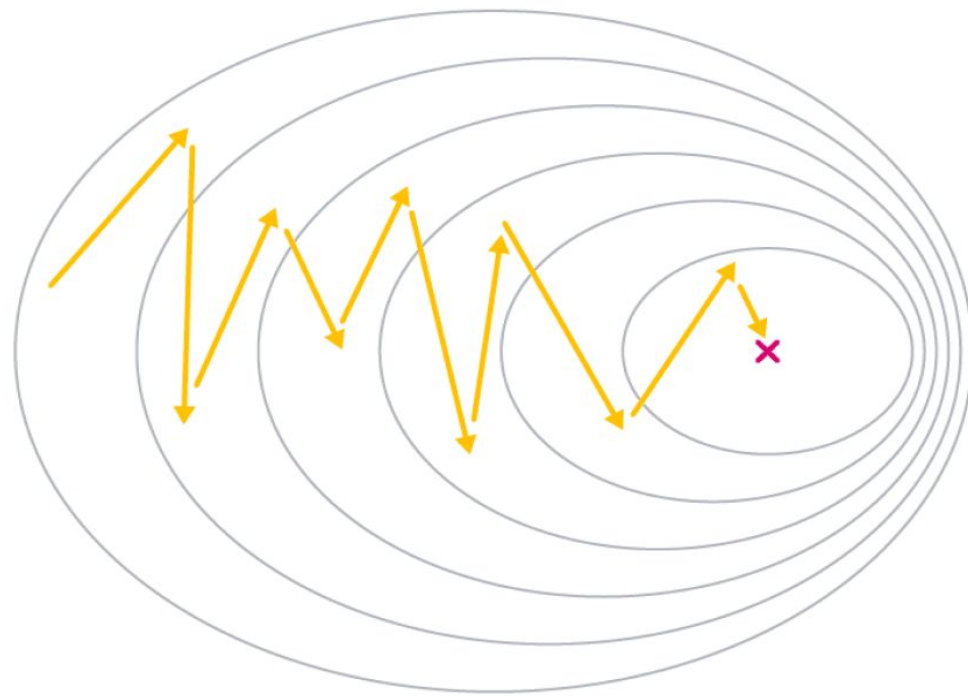
Еще одно преимущество этого метода - возможность работы с внешней памятью. Это связано с тем, что образец может быть настолько большим, что поместится только на жестком диске. В таких случаях стоит выбирать достаточно большой размер выборки. Доступ к данным с диска всегда медленнее, чем к данным из оперативной памяти, поэтому лучше получить сразу больше данных.

Поскольку стохастический градиент является лишь оценкой истинного градиента, SGD может быть достаточно шумным:

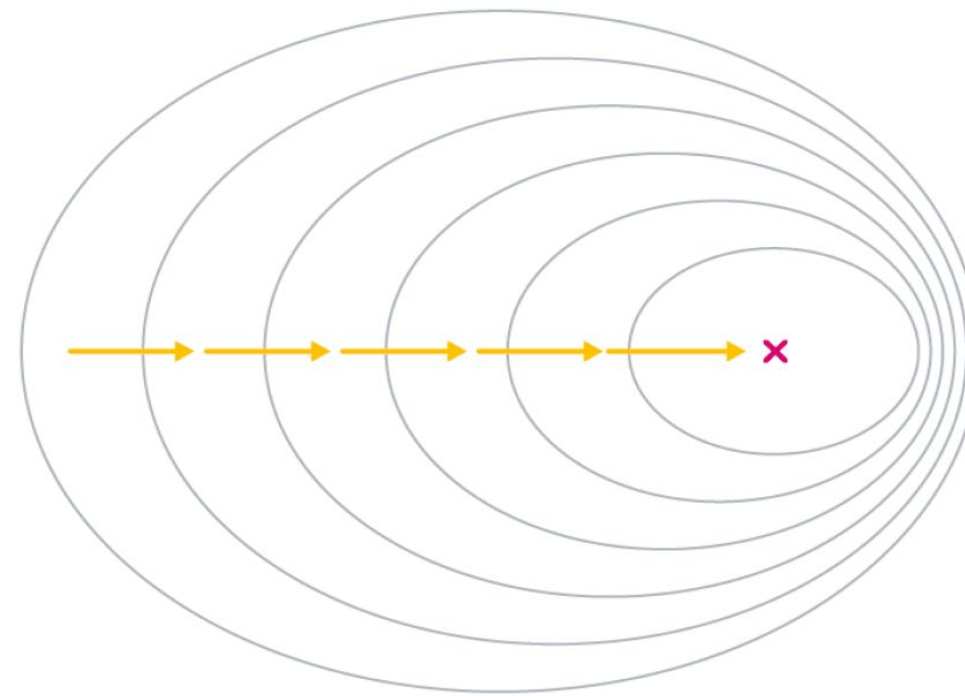


Стохастический градиентный спуск

Stochastic Gradient Descent

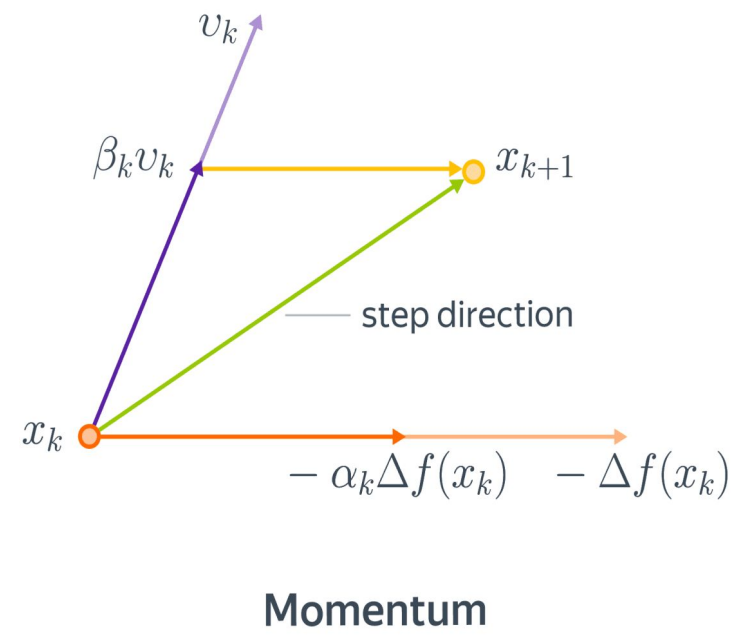


Gradient Descent

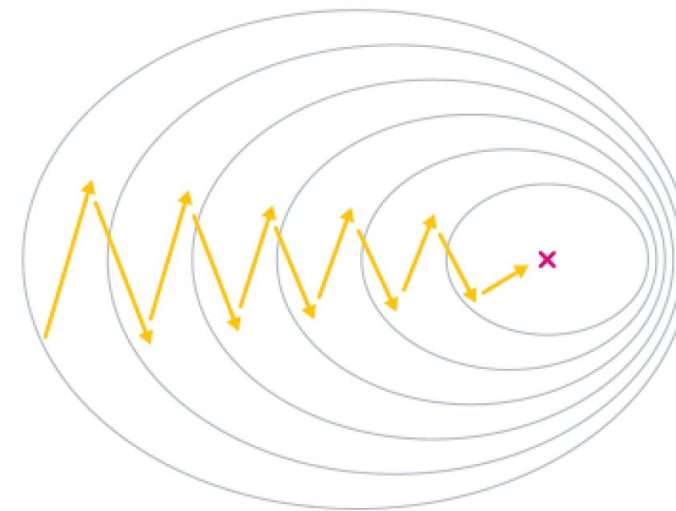




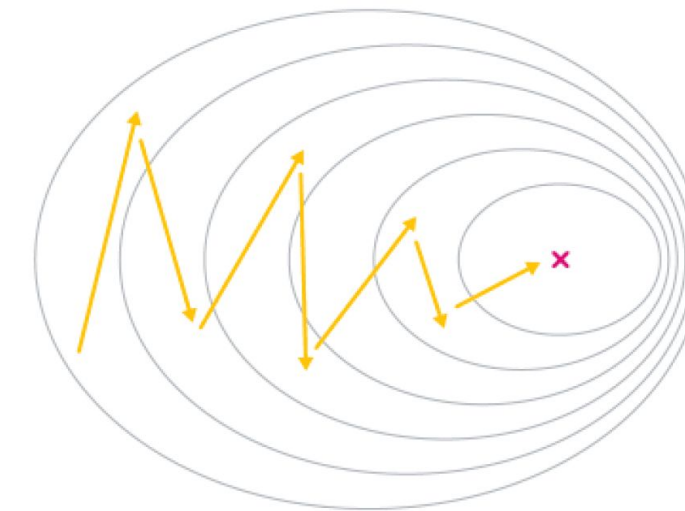
Метод инерции, momentum



SGD without momentum



SGD with momentum

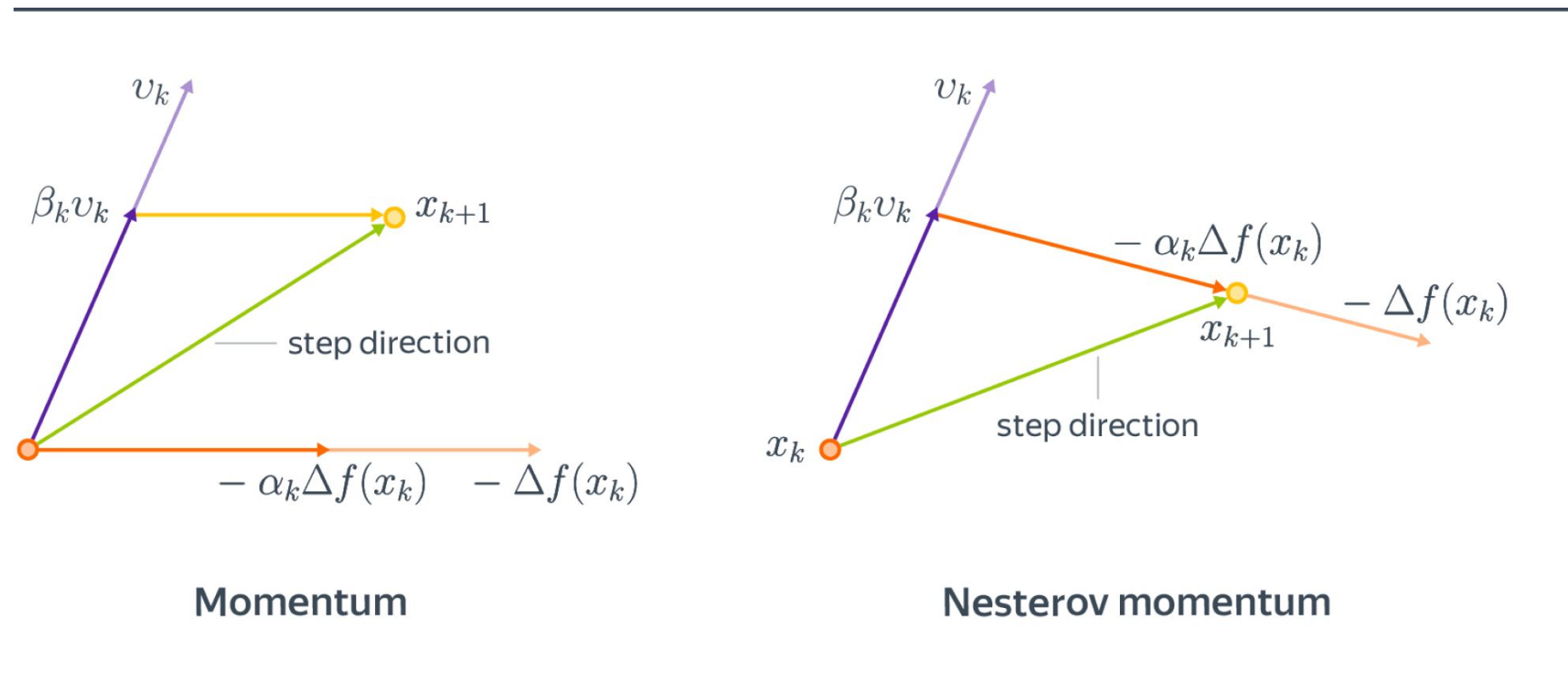


$$x_{k+1} = x_k - \alpha_k \nabla f(x_k) + \beta_k (x_k - x_{k-1}).$$



Accelerated Gradient Descent (Nesterov Momentum)

$$v_{k+1} = \beta_k v_k - \alpha_k \nabla f(x_k + \beta_k v_k) \quad x_{k+1} = x_k + v_{k+1}$$

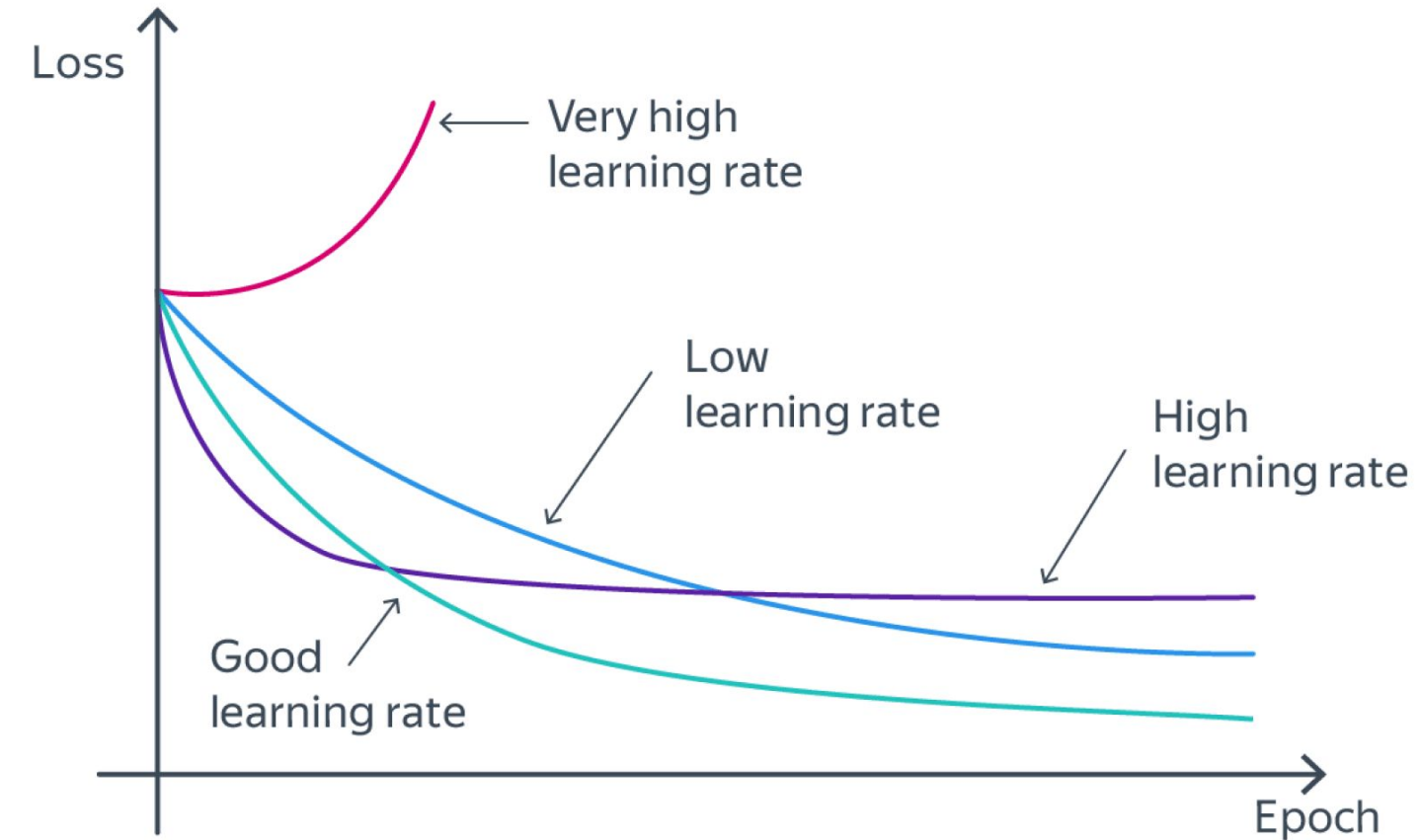




Adagrad (adaptive gradient)

$$G_{k+1} = \gamma G_k + (1 - \gamma)(\nabla f(x_k))^2$$

$$x_{k+1} = x_k - \frac{\alpha}{\sqrt{G_{k+1} + \varepsilon}} \nabla f(x_k).$$





Адам

Основная идея Adam заключается в комбинировании двух методов - адаптивного шага обучения и моментов.

1. Адаптивный шаг обучения:

Adam адаптивно регулирует скорость обучения для каждого параметра на основе истории градиентов. Это позволяет выбрать оптимальный шаг обучения для каждого параметра отдельно и избежать проблем со сходимостью на плоских участках функции потерь.

2. Моменты низкого порядка:

Adam использует два момента низкого порядка - первый момент (mean) и второй момент (variance) градиентов. Они используются для создания одномерной оценки среднего и дисперсии градиентов.



Выводы:

- Функция потерь представляет собой математическую функцию, которая измеряет разницу между предсказанными и фактическими значениями модели. Она позволяет оценить, насколько хорошо модель работает и какие параметры нужно изменить для улучшения ее результатов.
- Оптимизация, в свою очередь, градиентный спуск, которые позволяют найти минимум функции потерь и достичь наилучших результатов.
- Практическое применение функции потерь заключается в поиске оптимальных значений параметров модели, чтобы минимизировать функцию потерь. Существуют различные методы оптимизации, такие как модификации градиентного спуска. Например, в задачах классификации функция потерь может помочь модели правильно классифицировать объекты, а оптимизация может помочь найти оптимальные значения весов.
- Также функция потерь и оптимизация являются неотъемлемой частью обучения нейронных сетей. Они позволяют нейронной сети "учиться" на примерах и находить оптимальные значения весов, чтобы улучшить качество предсказаний.
- Функции потерь и оптимизации являются важными концепциями в машинном обучении. Они позволяют оценить и улучшить работу модели, а также найти оптимальные значения параметров. Знание и понимание этих концепций поможет вам стать более эффективным и успешным специалистом в области машинного обучения.



Спасибо за внимание

