Урок 10

Нейронные сети

10.1. Однослойная нейронная сеть

В этом уроке речь пойдет о нейронных сетях. Нейронная сеть — это универсальная модель, решающая широкий класс задач регрессии и классификации.

10.1.1. Нейронная сеть и задача регрессии

Пусть $\{(x_i, y_i)\}_{i=1}^{\ell}$ — обучающая выборка в задаче регрессии, $x_i \in \mathbb{R}^d$ — признаковое описание i-го объекта выборки, $y_i \in \mathbb{R}^1$ — значение целевой зависимой переменной на i-ом объекте выборки. Требуется построить поверхность $a(x)$, аппроксимирующую неизвестную целевую зависимость.

Задача оценки риска в страховании и финансах — пример задачи регрессии. Признаками в данной задаче являются время до страхового случая x_1 и цена страховки x_2. Требуется оценить ожидаемый риск y. Поверхность $a(x)$, которая аппроксимирует исторический риск y в точках x, представлена на следующем графике.

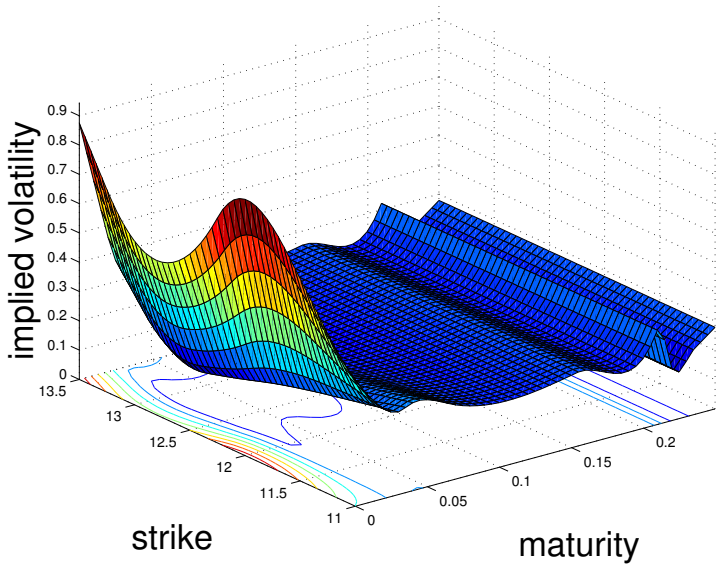


Рис. 10.1: Аппроксимирующая поверхность в задаче оценки риска.

Другие примеры задач регрессии $a(x) \mapsto y$:

- x — вектор исторических цен потребления электроэнергии, y — цена электроэнергии в следующий час,
- x — векторизованный снимок поверхности земли со спутника, y — объем зеленых насаждений;
- x — история продаж товара, y — уровень потребительского спроса.

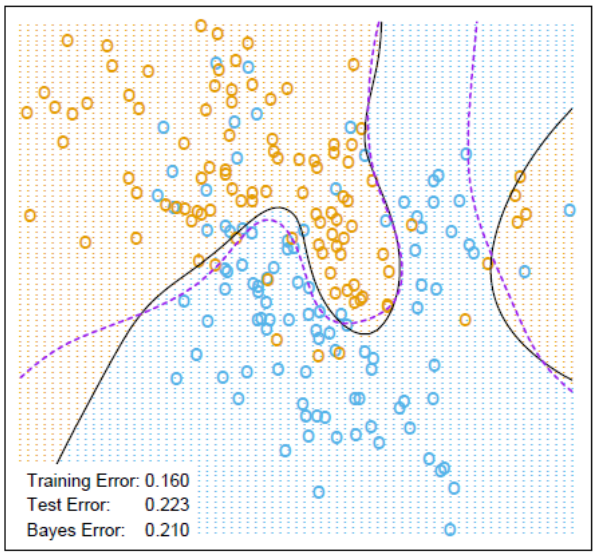


Рис. 10.2: Задача классификации с двумя признаками x_1 и x_2 (отложены по осям). На графике изображена разделяющая поверхность, которая отделяет объекты синего класса от объектов оранжевого класса.

10.1.2. Нейронная сеть и задача классификации

В задачах классификации целевая переменная y принимает конечное множество значений и называется меткой класса. Пусть $\{(x_i, y_i)\}_{i=0}^{\ell}$ — обучающая выборка в задаче бинарной классификации, $x_i \in \mathbb{R}^d$ и $y_i \in \{1, -1\}$ — признаковое описание и метка класса для i-го объекта соответственно. Требуется построить разделяющую поверхность

$$a(x) \mapsto y \in \{-1, 1\},$$

которая отделяет объекты одного класса от объектов другого класса таким образом, что, если $a(x) > 0$, то объект x принадлежит классу $y = +1$, и, если $a(x) \leq 0$, то x принадлежит классу $y = -1$.
Другие примеры задач классификации $a(x) \mapsto y$:

- x — временной ряд акселерометра мобильного телефона, y — вид физической активности;
- x — страница документа, y — релевантность этого документа по поисковому запросу;
- x — нотная запись в музыкальном произведении, y — это следующая нота, которая предполагается к проигрыванию.

10.1.3. Нейронная сеть как универсальная модель

Нейронная сеть считается универсальной моделью, так как она способна аппроксимировать любые поверхности. Соответствующую теорему сформулировал в 1957 году Андрей Николаевич Колмогоров.

Теорема (А. Н. Колмогоров, 1957) *Каждая непрерывная функция $a(x)$, заданная на единичном кубе d-мерного пространства, представима в виде*

$$a(x) = \sum_{i=1}^{2d+1} \sigma_i \left(\sum_{j=1}^{d} f_{ij}(x_j) \right),$$

где $x = [x_1, \ldots, x_d]^T$ — вектор описания объекта, функции $\sigma_i(\cdot)$ и $f_{ij}(\cdot)$ являются непрерывными, а f_{ij} не зависят от выбора a.

Согласно формулировке теоремы, функция $a(x)$ определена только на единичном кубе, а, следовательно, все элементы выборки должны лежать в нем. Это не является проблемой: всегда можно отмасштабировать признаки так, чтобы на обучающей выборке каждый признак принимал значения из отрезка $[0, 1]$.

Также следует отметить, что в теореме не указан конкретный вид функции σ_i и f_{ij}. Проблема конструирования нейронной сети остается сложной задачей до сих пор.

10.1.4. Однослойная нейронная сеть как единственный нейрон

Однослойная нейронная сеть, или нейрон, определяется выражением:

$$a(x, w) = \sigma(w^T x) = \sigma\left(\sum_{j=1}^{d} w_j^{(1)} x_j + w_0^{(1)}\right),$$

где σ — функция активации, w — вектор параметров (весов), x — вектор описания объекта. Функция активации должна быть непрерывной, монотонной и, желательно, дифференцируемой функцией.

Следует обратить внимание, что для удобства $x \in \mathbb{R}^{d+1}$ дополнен постоянным на всех объектах признаком $x_0 = 1$. Соответствующий ему вклад в скалярное произведение $w^T x$ равен w_0.

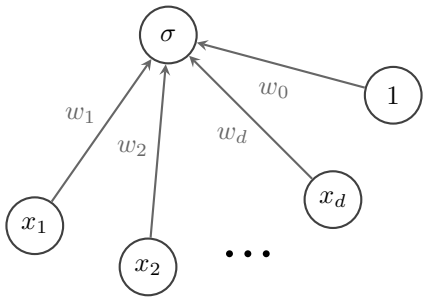


Рис. 10.3: Нейрон можно изобразить в виде вершины графа: он характеризуется своей функцией активации, имеет один выход и множество входов.

10.1.5. Функции активации: нейронная сеть как линейная модель

При решении задачи линейной регрессии в качестве функции активации можно выбрать тождественную функцию $\sigma = id$, тогда на выходе нейрона будет:

$$a(x, w) = w^T x,$$

то есть нейрон будет представлять собой линейный регрессор.

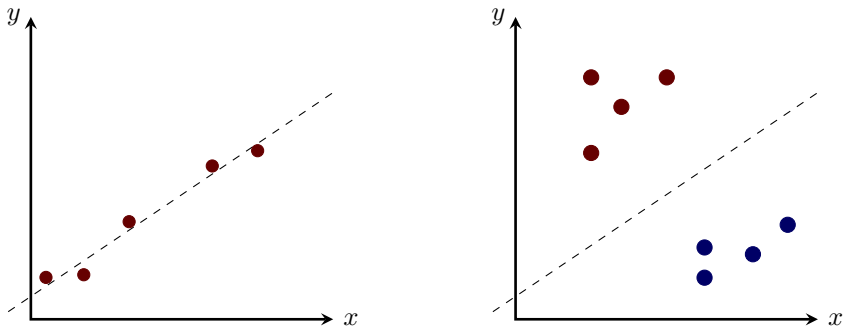


Рис. 10.4: Нейрон является линейным регрессором (график слева) или линейным классификатором (справа), если выбрать в качестве функции активации тождественную функцию или sign соответственно.

При решении задачи линейной классификации в качестве функции активации следует использовать пороговую функцию $\sigma = \text{sign}$. В таком случае нейрон:

$$a(x, w) = \text{sign}(w^T x)$$

является линейным классификатором.

10.1.6. Функции активации: модель логистической регрессии

Однослойная нейронная сеть является моделью логистической регрессии, если в качестве функции активации выбрана сигмоидная функция активации σ:

$$a(x, w) = \sigma(w^T x) = \frac{1}{1 + \exp(-w^T x)}.$$

Эта функция определяет вероятность принадлежности некоторого заданного объекта x к классу $y = 1$ при заданных параметрах w:

$$\sigma(w^T x) = P(y = 1|w, x).$$

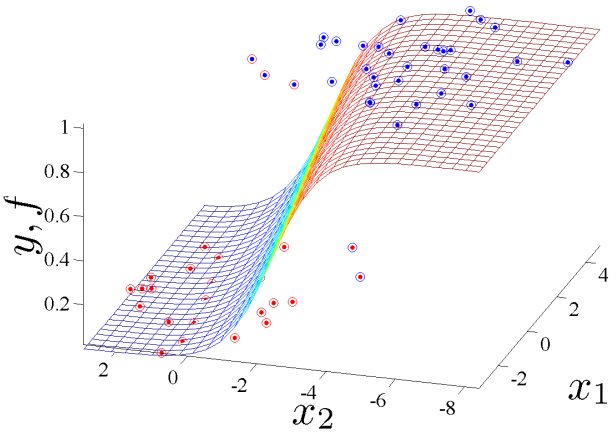


Рис. 10.5: Аппроксимирующая поверхность на графике построена с использованием сигмоидной функции активации.

Если количество классов равно K:

$$y = [y^1, \ldots, y^K]^T,$$

то необходимо использовать сеть из K нейронов, каждый из которых вычисляет вероятность принадлежности к своему классу. В качестве функции активации используется softmax:

$$\sigma = \text{softmax}(w_1^T x, \ldots, w_K^T x) = \frac{\exp(w_k^T x)}{\sum_{i=1}^K \exp(w_i^T x)}.$$

10.1.7. Функции активации: другие примеры

Гиперболический тангенс в качестве функции активации:

$$\tanh(x) = \frac{\exp(2x) - 1}{\exp(2x) + 1}$$

представляет собой дифференцируемую альтернативу пороговой функции $\text{sign}(\cdot)$.

Существует еще более мягкий вариант: функция активации

$$\text{softsign}(x) = \frac{x}{1 + |x|},$$

как и $\tanh(x)$ стремится к ± 1, но в отличие от него, сходится к этим значениям не так быстро.

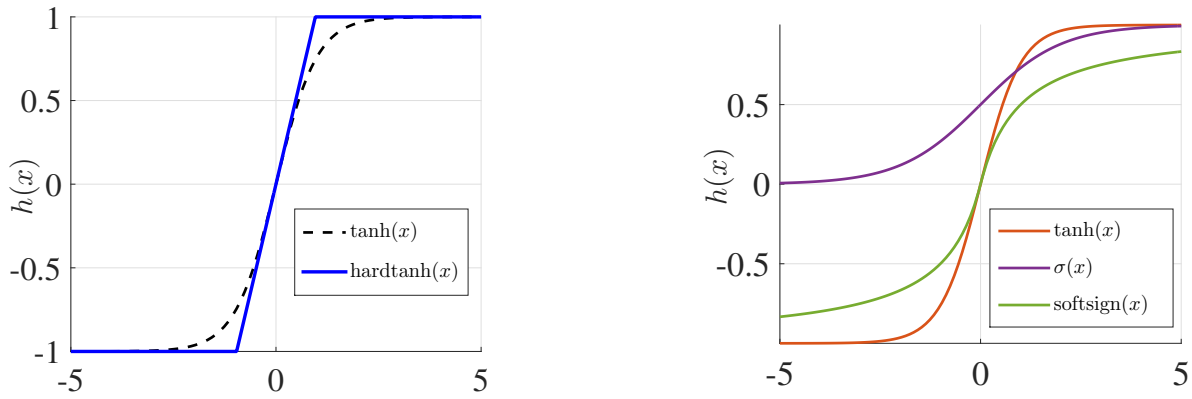


Рис. 10.6: Графики различных функций активации

В сетях глубокого обучения, в более сложных нейронных сетях, используются функция-выпрямитель Rectified linear unit (ReLu):

$$\text{ReLu}(f) = \max(0, f).$$

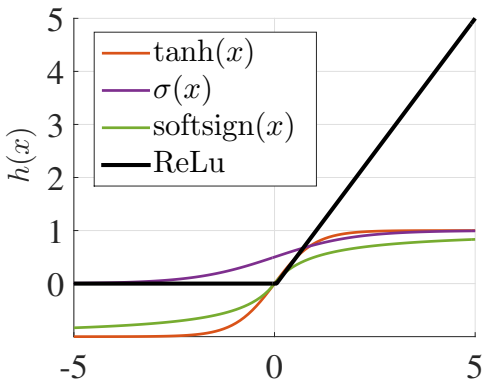


Рис. 10.7: Графики функций активации: $\text{ReLu}(f)$ и ее дифференцируемого приближения.

Эту функция ведет себя также, как работает диод в радиоэлектрических цепях. Данная функция не дифференцируема, но имеет следующее дифференцируемое приближение:

$$\ln(1 + \exp(f))$$

10.2. Многослойная нейронная сеть. Функция ошибки

В этом уроке будет рассказано про двухслойную нейронную сеть и про различные функции ошибок.

10.2.1. Пределы применимости однослойных сетей

Однослойные сети применимы только для линейно разделимых выборок.

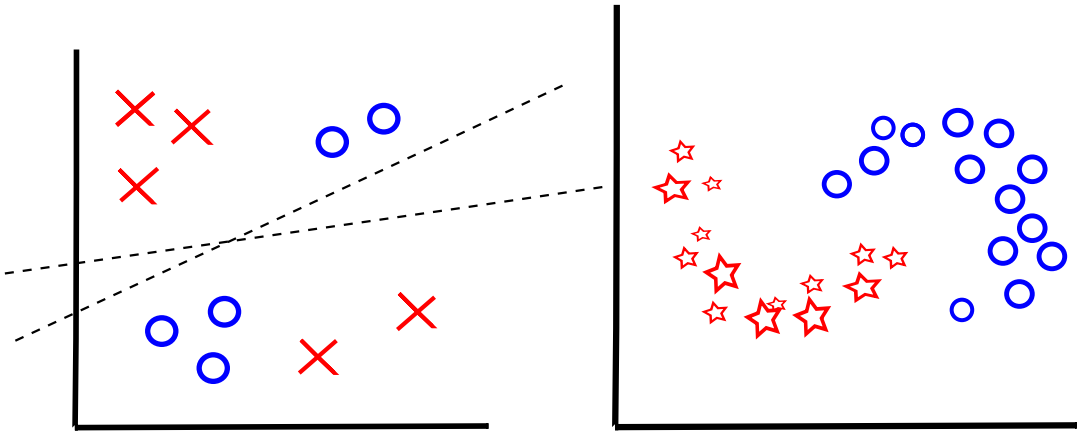


Рис. 10.8: Два примера линейно неразделимых выборок.

10.2.2. Двухслойная нейронная сеть

Двухслойная нейронная сеть представляет собой линейную комбинацию из D нейронов (однослойных нейронных сетей):

$$a(x,w) = \sigma^{(2)}\left(\sum_{i=1}^{D} w_i^{(2)} \cdot \sigma^{(1)}\left(\sum_{j=1}^{d} w_{ji}^{(1)} x_j + w_{0i}^{(1)}\right) + w_0^{(2)}\right).$$

Это выражение можно переписать в векторных обозначениях:

$$a(x,w) = \sigma^{(2)}\left(w^{T^{(2)}}\sigma^{(1)}\left([w_1^{T^{(1)}}x, \ldots, w_D^{T^{(1)}}x]\right)\right).$$

Вектор w параметров нейронной сети получается соединением всех параметров нейронной сети на всех слоях:

$$w = \{W^{(1)}, w^{(2)}\},$$

где:

$$W^{(1)} = [w_0^{(1)}, w_1^{(1)}, \ldots, w_d^{(1)}]^T \in \mathbb{R}^{(d+1)\times D},$$

$$w_i^{(1)} = [w_{0i}^{(1)}, w_{1i}^{(1)}, \ldots, w_{di}^{(1)}]^T \in \mathbb{R}^{d+1}, \qquad w^{(2)} = [w_0^{(2)}, w_1^{(2)}, \ldots, w_D^{(2)}]^T \in \mathbb{R}^{D+1}.$$

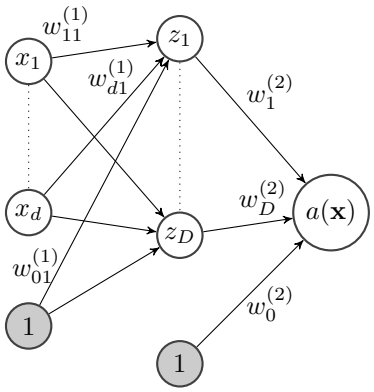


Рис. 10.9: Изображение двухслойной нейронной сети с помощью направленного графа.

Продолжая по аналогии, можно получить трехслойную и многослойную нейронные сети.

10.2.3. Разделяющая способность многослойной нейронной сети

Теорема (Универсальная теорема аппроксимации, K. Hornik, 1991) *Для любой непрерывной функции $f(x)$ найдется нейронная сеть $a(x)$ с линейным выходом $a(x, W) = \sigma^{(M)}(x)$, где*

$$f^{(k)}(x) = w_0^{(k)} + W^{(k)} z^{(k-1)}(x), \qquad z^{(k)}(x) = \sigma^{(k)}(f^{(k-1)}(x)),$$

аппроксимирующая $f(x)$ с заданной точностью.

Теорема выполняется для различных функций активации, в частности для функции сигмоид и функции гиперболический тангенс. Чтобы получить требуемую аппроксимацию необходимо определить оптимальные значения параметров w^*.

10.2.4. Разделяющая поверхность: проблема переобучения

Качество аппроксимации целевой переменной y функцией $a(x, w)$ зависит от выбора параметров w.

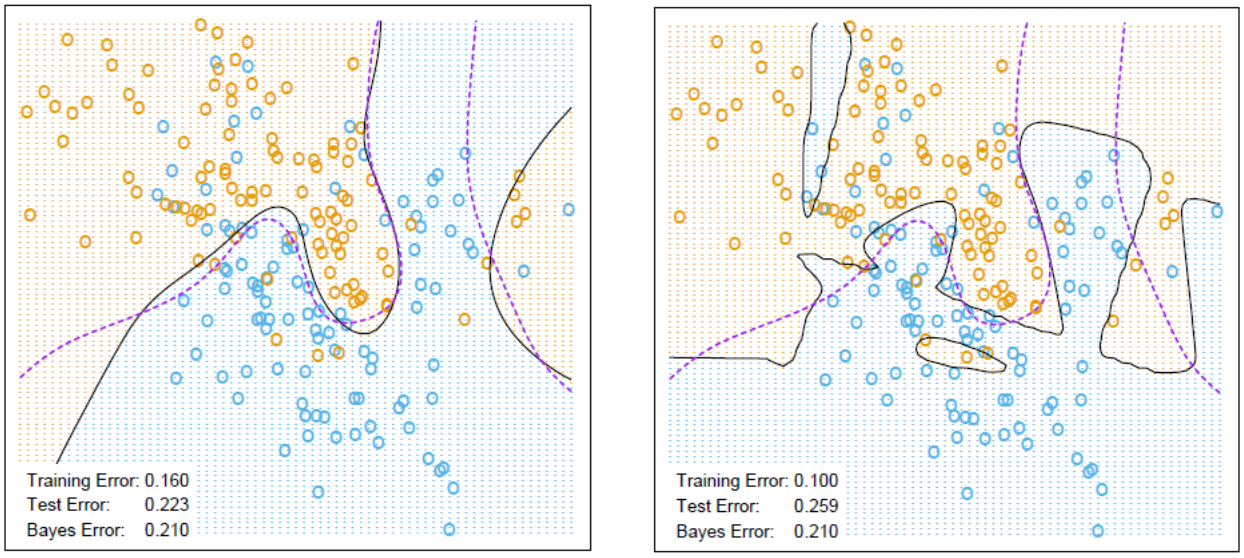


Рис. 10.10: Разделяющая поверхность нейронной сети в случае оптимальных и неоптимальных значений параметров.

В случае оптимальных значений параметров, как это видно по графику (левый), разделяющая поверхность отнесла несколько объектов не к тому классу, к которому они принадлежат. Тем не менее именно эту разделяющую поверхность следует считать оптимальной.

Действительно, в случае неоптимальных значений параметров, веса нейросети подстроены таким образом, что текущая выборка классифицируется идеально: каждый объект принадлежит области своего класса. Однако при изменении состава выборки разделение уже будет неверным. Такая нейросеть, которая корректно аппроксимирует обучающую выборку, но плохо справляется с контрольной, называется переобученной.

10.2.5. Функции ошибки нейронной сети

Пусть $Q(w)$ — функция ошибки, которая зависит как от конфигурации w нейронной сети, так и от ее состава. Задача нахождения оптимальных параметров w^*, то есть таких, при которых ошибка нейронной сети минимальна, имеет вид:

$$w^* = \operatorname{argmin}_w Q(w).$$

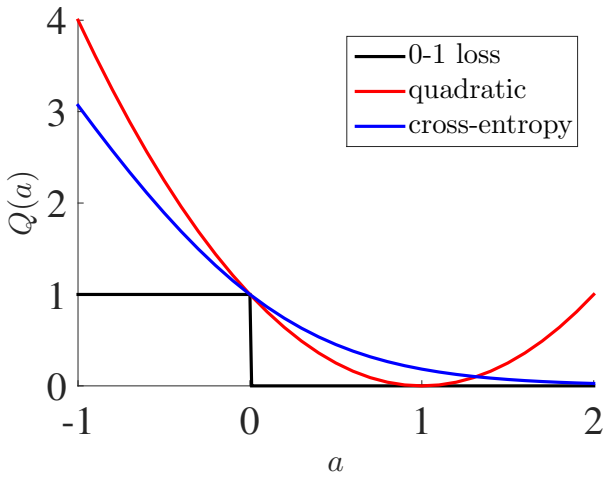


Рис. 10.11: Графики функций потерь

Выбор функции ошибки зависит от специфики решаемой задачи:

- В задаче регрессии (при решении различных прикладных задач) в качестве функции ошибки часто используется функция «галочка»:
$$Q(w) = \sum_{i=1}^{\ell} |a(x_i, w) - y_i|.$$
Но, к сожалению, эта функция не является дифференцируемой.

- Дифференцируемой функцией ошибки, используемой в задачах регрессии, является, например, квадратичная функция ошибки:
$$Q(w) = \sum_{i=1}^{\ell} (a(x_i, w) - y_i)^2.$$

- В задачах классификации используется функция ошибки «0-1 loss», которая равна числу несовпадений между восстановленными метками классов и фактическими:
$$Q(w) = \sum_{i=1}^{\ell} [\operatorname{sign} a(x_i, w) = y_i].$$

- Дифференцируемая функция ошибки для задачи классификации — кросс-энтропия, функция наибольшего правдоподобия в задаче логистической регрессии:
$$Q(w) = -\sum_{i=1}^{\ell} \left(y_i \ln a(x_i, w) + (1 - y_i) \ln(1 - a(x_i, w))\right).$$

10.3. Оптимизация параметров нейронной сети

10.3.1. Задача оптимизации

Задача оптимизации для нахождения оптимальных значений параметров:

$$w^* = \operatorname{argmin}_w Q(w),$$

где Q — функция ошибки, которая зависит от выборки, структуры нейросети (числа слоёв, нейронов и видов функций активаций) и значения вектора параметров w.

На следующем графике изображено характерное поведение функции ошибки при изменении значений двух параметров.

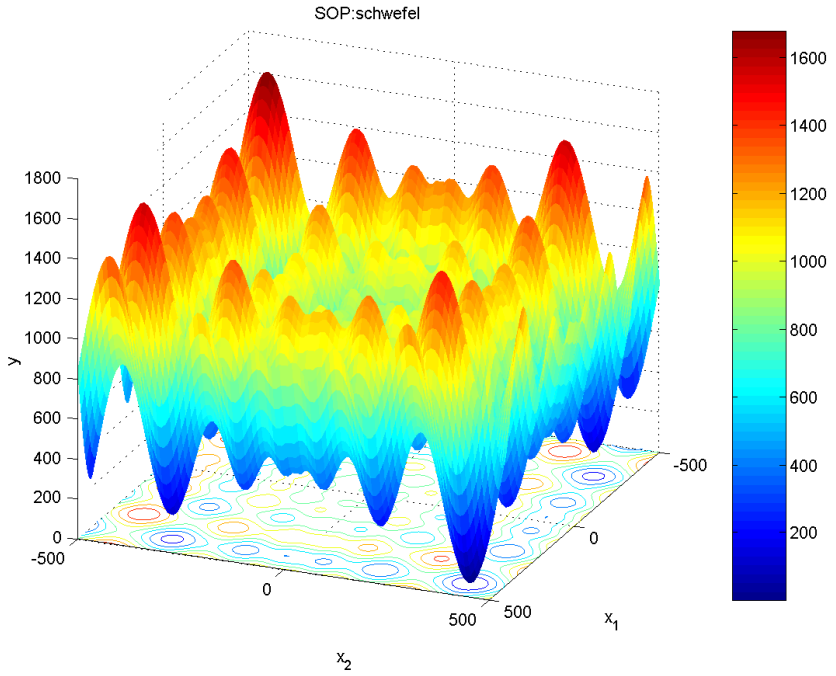


Рис. 10.12: Функция ошибки в зависимости от двух параметров.

Как отчетливо видно по графику, функция ошибки может иметь значительное число локальных минимумов. Задача является нетривиальной даже в пространстве малой размерности. При решении задачи оптимизации для нахождения оптимальных значений параметров есть два типа алгоритмов:

- Алгоритмы стохастической оптимизации:
  - Случайный перебор $w_1, w_2, \ldots$;
  - Генетический алгоритм оптимизации $w_1 \to w_2 \to \ldots$;
  - Моделируемый отжиг, значения w задаются по расписанию;
- Алгоритмы градиентного спуска.

Применение алгоритмов градиентного спуска к данной задаче будет обсуждаться далее.

10.3.2. Локально-квадратичная аппроксимация функции ошибки

Алгоритм локальной аппроксимации функции ошибки минимизирует не саму функцию ошибки, а функцию, которая ее аппроксимирует в окрестности точки w^*. Например, удобно аппроксимировать исходную функцию квадратичной функцией:

$$Q(w) \approx Q(w^*) + \frac{1}{2}(w - w^*)^T \mathbf{H}(w - w^*).$$

На каждой итерации этого алгоритма ищется минимум приближенной функции и окрестность сдвигается в сторону более оптимальных значений.

10.3.3. Градиентный спуск

Алгоритм градиентного спуска предполагает, что функция ошибки является дифференцируемой (например квадратичная или кросс-энтропийная) и можно вычислить ее значение $Q(w)$ и значение ее градиента $\nabla_w Q(w)$.

При использовании градиентного спуска следует помнить, что он может найти как точку глобального минимума, так и точку локального минимума. Градиентный спуск — итеративная процедура нахождения параметров:

$$w_{k+1} = w_k - \alpha \sum_{i=1}^{\ell} \nabla_w \varphi(w_k, x_i),$$

где $\varphi(w_k, x_i)$ — ошибка на одном объекте, α — размер шага, k — номер итерации. Итеративный процесс останавливается, когда стабилизируется значение функции ошибки.

10.3.4. Стохастический градиентный спуск

Стохастический градиентный спуск, вариант метода градиентного спуска, заключается в том, что градиент считается не на всех элементах выборки, а только на одном случайно выбранном объекте x_i:

$$w_{k+1} = w_k - \alpha \nabla_w \varphi(w_k, x_k).$$

Итерации также продолжаются до стабилизации функции ошибки.

10.3.5. Обратное распространение ошибки (backprop)

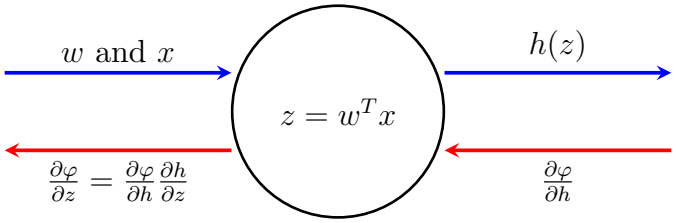
Очень популярный и быстрый алгоритм — это алгоритм обратного распространения ошибки backpropagation, или backprop. Для каждого нейрона сети $h(w^T x)$ и одного объекта выборки x вычисляется:

- Взвешенная сумму входных сигналов $z = w^T x$.
- Значение на выхода $h(z)$ (прямое распространение).
- Производная ошибки $\frac{\partial \varphi(z)}{\partial z}$ по значению нейрона и его параметрам. Эта производная зависит от значений последующих нейронов и называется обратным распространением.

По обратному распространению можно вычислить градиент функции ошибки по w:

$$\nabla_w \varphi(z) = \frac{\partial \varphi(z)}{\partial z} \nabla_w z = \frac{\partial \varphi(z)}{\partial z} x,$$

а затем подкорректировать веса аналогично тому, как это делалось в градиентных методах.



Существует так называемая проблема затухания градиента. При больших значениях входов нейронов $|x|$ значения производной функции активации $h'(x) \to 0$, например для сигмоидной функции $h(x) = \text{sigmoid}(x)$ или гиперболического тангенса $h(x) = \tanh(x)$. Следовательно, в данном случае градиент $\frac{\partial \varphi}{\partial z^{(s)}}$ не будет распространяться на предыдущие слои.

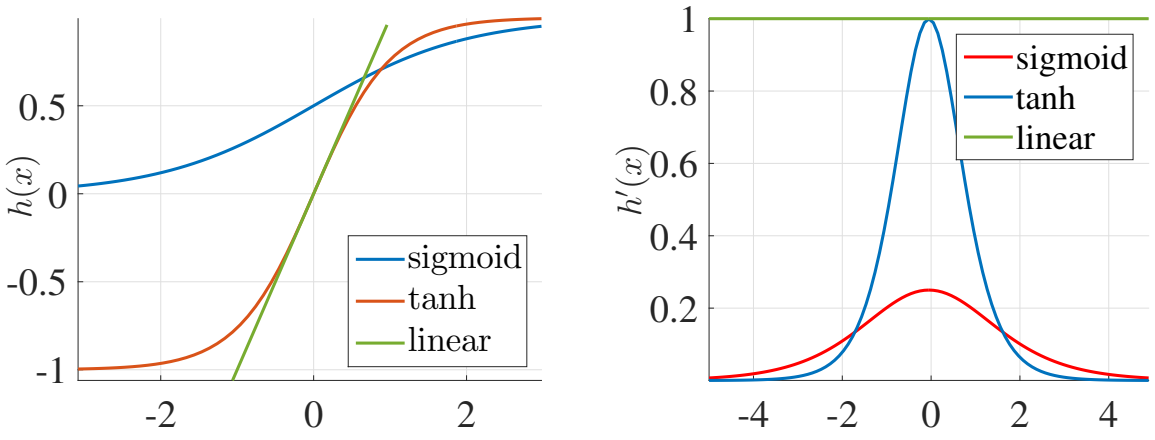


Рис. 10.13: Графики некоторых функций активаций и их производных.

Преимущества алгоритма обратного распространения ошибки:

- Градиент вычисляется за время, сравнимое с временем вычислением сети.
- Алгоритм подходит для многих дифференцируемых функций активации.
- Не обязательно использовать всю обучающую выборку.

Недостатки алгоритма обратного распространения ошибки:

- Возможна медленная сходимость к решению.
- Решение может быть локальным, а не глобальным минимумом.
- Возможно переобучение сети.

10.4. Регуляризация и прореживание нейронной сети

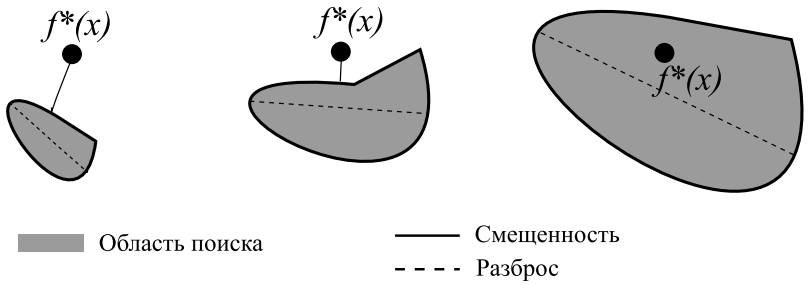
10.4.1. Регуляризация

Чтобы избежать переобучения, необходимо модифицировать задачу оптимизации: ввести штраф за большие значения весов:

$$w^* = \operatorname{argmin}_w \left[Q(w) + \tau \sum_{i,j,k} \left(w_{ij}^{(k)}\right)^2 \right],$$

где τ — коэффициент регуляризации, который контролирует жесткость ограничений параметров w. При этом, чем меньше τ, тем точнее функция $a(x)$ описывает выборку.

На следующем рисунке изображены характерные смещенность, разброс и область поиска решения в пространстве параметров в случае различных значений параметра регуляризации τ.



На рисунке слева изображен случай, когда коэффициент регуляризации имеет самое большое значение. Область поиска решения в пространстве параметров мала, но велика смещенность параметра от минимального значения функции ошибки.

Справа изображена противоположная ситуация: коэффициент τ мал, область поиска оптимальных параметров велика и смещенности нет. Но, возможно, нейронная сеть окажется переобученной. В центре рисунка изображен компромиссный вариант.

На следующем графике изображена зависимость значений параметров нейронной сети от τ.

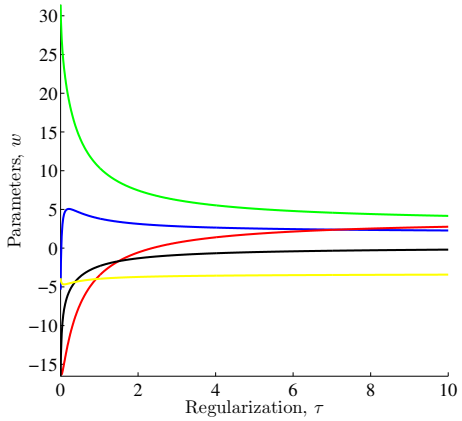


Рис. 10.14: Значения параметров сети в зависимости от τ.

По графику отчетливо видно, что регуляризация не снижает число параметров и не упрощает структуру сети. Более того, по мере увеличения τ параметры перестают изменяться.

10.4.2. Прореживание сети

Для снижения числа параметров предлагается исключить некоторые нейроны или связи. Принцип исключения следующий: если функция ошибки не изменяется, то нейронную сеть можно упрощать и дальше.

На следующем графике изображена характерная зависимость функции ошибки от числа удаленных параметров:

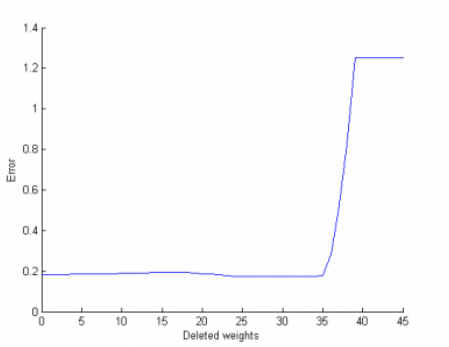


Рис. 10.15: Зависимость функции ошибки $Q(w)$ от числа удаленных связей.

По этому графику отчетливо видно, что функция ошибки начала изменяться, только когда значительное число связей было удалено. Связи (параметры) следует удалять исходя из следующих соображений. Параметр можно удалить, если:

- его значение близко к нулю.
- соответствующий сигнал обладает большой дисперсией, то есть реагирует на шум в данных.
- его удаление практически не меняет функцию ошибки.

Последний вариант следует рассмотреть подробнее: это так называемы «метод оптимального разрушения мозга» (optimal brain damage), или метод оптимального прореживания. Функцию ошибки можно разложить в ряд Тейлора в окрестности оптимума:

$$Q(w + \Delta w) = Q(w) + \mathbf{g}^T(w)\Delta w + \frac{1}{2}\Delta w^T \mathbf{H} \Delta w + o(\|w\|^3),$$

где $\mathbf{H}$ матрица Гессе, а $\mathbf{g}^T(w) = 0$, поскольку разложение идет в окрестности оптимума.

В методе оптимального прореживания необходимо исключить один параметр w_j и, возможно, подкорректировать остальные веса w таким образом, чтобы приближенное выражение для изменения функции ошибки:

$$\Delta Q(w, \Delta w) \approx \frac{1}{2}\Delta w^T \mathbf{H} \Delta w$$

было минимиальным. Исключению j-го параметра соответствует выполнение условия:

$$\Delta w_j + w_j = 0.$$

Таким образом, чтобы определить исключаемый признак, требуется решить задачу условной оптимизации:

$$j = \text{argmin}_j \left(\min_{\Delta w} \left(\Delta w^T \mathbf{H} \Delta w \middle| \Delta w_j + w_j = 0 \right) \right).$$

Можно показать, что для этого параметра w_j будем минимальным значение функции выпуклости:

$$L_j = \frac{w_j^2}{2\mathbf{H}^{-1}_{jj}}.$$

10.4.3. Построение нейронной сети

Нейронная сеть может работать в двух режимах:

- **Режим обучения:** В этом режиме задается структура нейронной сети и оптимизируются ее параметры.
- **Режим эксплуатации:** вычисление значений $a(x, w^*)$ при фиксированных значениях параметров.

Для того чтобы задать нейронную сеть, требуется указать число слоев, число нейронов в каждом слое, тип функции активации в каждом слое, тип функции ошибки. Также желательно, чтобы подготовленная выборка не содержала пропусков, а признаки были отнормированы.

10.4.4. Стабилизация параметров сети

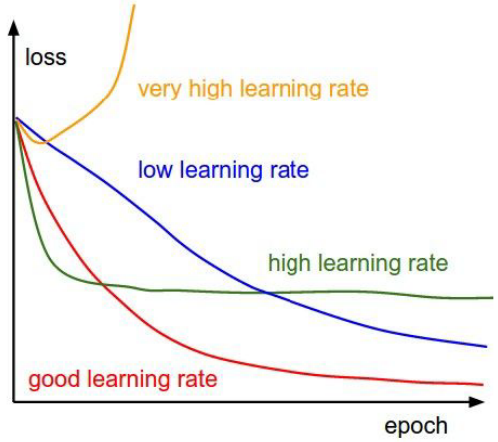


Рис. 10.16: Зависимость ошибки от номера итерации.

По скорости обучения можно судить о соответствии выборки и нейронной сети:

- Если нейронная сеть будет слишком сложна, то она быстро переобучится, как, например, показывает желтая линия.
- Если выборка будет сложна или сильно зашумлена для нейронной сети, то нейронная сеть будет обучаться медленно, как показывает синяя линия.
- Если выборка по сложности соответствует нейронной сети, то, скорее всего, мы увидим красную линию — хорошую скорость обучения.
- Важно также следить за разницей между значениями функции ошибки на обучении и на контроле. Эта разница не должна быть существенной. Если она велика, то это значит, что нейронная сеть переобучилась и следует изменить ее сложность.

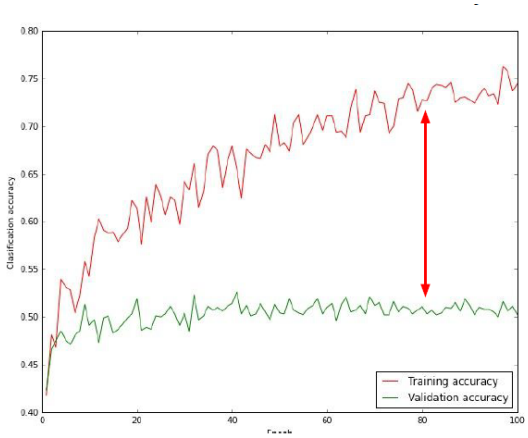


Рис. 10.17: Существенная разница между значениями ошибки на обучении и на контроле говорит о переобученности нейронной сети.