

- **Отбор оптимальных сетей:** тех, которые дадут наименьшую ошибку предсказания на неизвестных пока данных.
- **Оценка значимости предсказаний:** оценка ошибки предсказаний не менее важна, чем само предсказанное значение.

Если до сих пор мы ограничивали наше рассмотрение, в основном, последними этапами, связанными с обучением собственно нейросетей, то в этой главе мы сосредоточимся на первых этапах нейросетевого анализа - предобработке данных. Хотя переработка не связана непосредственно с нейросетями, она является одним из ключевых элементов этой информационной технологии. Успех обучения нейросети может решающим образом зависеть от того, в каком виде представлена информация для ее обучения.

В этой главе мы рассмотрим предобработку данных для обучения с учителем и постараемся, главным образом, выделить и проиллюстрировать на конкретных примерах *основной принцип* такой предобработки: увеличение информативности примеров для повышения эффективности обучения.

Кодирование входов-выходов

В отличие от обычных компьютеров, способных обрабатывать любую символьную информацию, нейросетевые алгоритмы работают только с числами, ибо их работа базируется на арифметических операциях умножения и сложения. Именно таким образом набор синаптических весов определяет ход обработки данных.

Между тем, не всякая входная или выходная переменная в исходном виде может иметь численное выражение. Соответственно, все такие переменные следует закодировать - перевести в численную форму, прежде чем начать собственно нейросетевую обработку. Рассмотрим, прежде всего основной руководящий принцип, общий для всех этапов предобработки данных.

Максимизация энтропии как цель предобработки

Допустим, что в результате перевода всех данных в числовую форму и последующей нормировки все входные и выходные переменные отображаются в единичном кубе. Задача нейросетевого моделирования - найти статистически достоверные зависимости между входными и выходными переменными. Единственным источником информации для статистического моделирования являются примеры из обучающей выборки. Чем больше бит информации принесет каждый пример - тем лучше используются имеющиеся в нашем распоряжении данные.

Рассмотрим произвольную компоненту нормированных (предобработанных) данных: \tilde{x}_i . Среднее количество информации, приносимой каждым примером \tilde{x}_i^α , равно энтропии распределения значений этой компоненты $H(\tilde{x}_i)$. Если эти значения сосредоточены в относительно небольшой области единичного интервала, информационное содержание такой компоненты мало. В пределе нулевой энтропии, когда все значения переменной совпадают, эта переменная не несет никакой информации. Напротив, если значения переменной \tilde{x}_i^α равномерно распределены в единичном интервале, информация такой переменной максимальна.

Общий принцип преобработки данных для обучения, таким образом, состоит в максимизации энтропии входов и выходов. Этим принципом следует руководствоваться и на этапе кодирования нечисловых переменных.

Типы нечисловых переменных

Можно выделить два основных типа нечисловых переменных: *упорядоченные* (называемые также *ординальными* - от англ. *order* - порядок) и *категориальные*. В обоих случаях переменная относится к одному из дискретного набора классов $\{c_1, \dots, c_n\}$. Но в первом случае эти классы упорядочены - их можно *ранжировать*: $c_1 \succ c_2 \succ \dots \succ c_n$, тогда как во втором такая упорядоченность отсутствует. В качестве примера упорядоченных переменных можно привести сравнительные категории: плохо - хорошо - отлично, или медленно - быстро. Категориальные переменные просто обозначают один из классов, являются *именами* категорий. Например, это могут быть имена людей или названия цветов: белый, синий, красный.

Кодирование ординальных переменных

Ординальные переменные более близки к числовой форме, т.к. числовой ряд также упорядочен. Соответственно, для кодирования таких переменных остается лишь поставить в соответствие номерам категорий такие числовые значения, которые сохраняли бы существующую упорядоченность. Естественно, при этом имеется большая свобода выбора - любая монотонная функция от номера класса порождает свой способ кодирования. Какая же из бесконечного многообразия монотонных функций - наилучшая?

В соответствии с изложенным выше общим принципом, мы должны стремиться к тому, чтобы максимизировать энтропию закодированных данных. При использовании сигмоидных функций активации все выходные значения лежат в конечном интервале - обычно $[0, 1]$ или $[-1, 1]$. Из всех статистических функций распределения, определенных на конечном интервале, максимальной энтропией обладает равномерное распределение.

Применительно к данному случаю это подразумевает, что кодирование переменных числовыми значениями должно приводить, по возможности, к равномерному заполнению единичного интервала закодированными примерами. (Захватывая заодно и этап нормировки.) При таком способе "оцифровки" все примеры будут нести примерно одинаковую информационную нагрузку.

Исходя из этих соображений, можно предложить следующий практический рецепт кодирования ординальных переменных. Единичный отрезок разбивается на n отрезков - по числу классов - с длинами пропорциональными числу примеров каждого класса в обучающей выборке:

$\Delta x_k = P_k / P$, где P_k - число примеров класса k , а P , как обычно, общее число примеров. Центр каждого такого отрезка будет являться численным значением для соответствующего ординального класса (см. Рисунок 1).

$c_1 = (0,0), c_2 = (1,0), c_3 = (0,1), c_4 = (1,1)$, обеспечивающим равномерную "загрузку" кодирующих нейронов.

Отличие между входными и выходными переменными

В заключении данного раздела отметим одно существенное отличие способов кодирования входных и выходных переменных, вытекающее из определения градиента ошибки:

$\frac{\partial E}{\partial w_{ij}^{[n]}} = \delta_i^{[n]} x_j^{[n]}$. А именно, входы участвуют в обучении непосредственно, тогда как выходы -

лишь *опосредованно* - через ошибку верхнего слоя. Поэтому при кодировании категорий в качестве выходных нейронов можно использовать как логистическую функцию активации $f(a) = 1/(e^{-a} + 1)$, определенную на отрезке $[0, 1]$, так и ее антисимметричный аналог для отрезка $[-1, 1]$, например: $f(a) = \tanh(a)$. При этом кодировка выходных переменных из обучающей выборки будет либо $\{0, 1\}$, либо $\{-1, 1\}$. Выбор того или иного варианта никак не скажется на обучении.

В случае со входными переменными дело обстоит по-другому: обучение весов нижнего слоя сети определяется *непосредственно* значениями входов: на них умножаются невязки, зависящие от выходов. Между тем, если с точки зрения операции умножения значения ± 1 равноправны, между 0 и 1 имеется существенная асимметрия: нулевые значения не дают никакого вклада в градиент ошибки. Таким образом, выбор схемы кодирования входов влияет на процесс обучения. В силу логической равноправности обоих значений входов, более предпочтительной выглядит симметричная кодировка: $\{-1, 1\}$, сохраняющая это равноправие в процессе обучения.

Нормировка и предобработка данных

Как входами, так и выходами нейросети могут быть совершенно разнородные величины. Очевидно, что результаты нейросетевого моделирования не должны зависеть от единиц измерения этих величин. А именно, чтобы сеть трактовала их значения единообразно, все входные и выходные величины должны быть приведены к единому - единичному - масштабу. Кроме того, для повышения скорости и качества обучения полезно провести дополнительную предобработку данных, выравнивающую распределение значений еще до этапа обучения.

Индивидуальная нормировка данных

Приведение данных к единичному масштабу обеспечивается нормировкой каждой переменной на диапазон разброса ее значений. В простейшем варианте это - линейное преобразование:

$$\tilde{x}_i = \frac{x_i - x_{i,\min}}{x_{i,\max} - x_{i,\min}}$$

в единичный отрезок: $\tilde{x}_i \in [0, 1]$. Обобщение для отображения данных в интервал $[-1, 1]$, рекомендуемого для входных данных тривиально.