Глава
7

Предобработка данных

Как решаются конкретные задачи? Кодирование входов-выходов. Виды нормировки. Линейная предобработка входов. Понижение размерности и отбор наиболее значимых входов.

> 🕮 Вытапливай воск, но сохраняй мед.
> **Козьма Прутков**
>
> 🕮 Мелочи не играют решающей роли. Они решают все.
> **Х.Маккей.** ***Как уцелеть среди акул***

Необходимые этапы нейросетевого анализа

Теперь, после знакомства с базовыми принципами нейросетевой обработки, можно приступать к практическим применениям полученных знаний для решения конкретных задач. Первое, с чем сталкивается пользователь любого нейропакета - это необходимость подготовки данных для нейросети. До сих пор мы не касались этого, вообще говоря, непростого вопроса, молчаливо предполагая, что данные для обучения уже имеются и представлены в виде, доступном для нейросети. На практике же именно предобработка данных может стать наиболее трудоемким элементом нейросетевого анализа. Причем, знание основных принципов и приемов предобработки данных не менее, а может быть даже более важно, чем знание собственно нейросетевых алгоритмов. Последние как правило, уже "зашиты" в различных нейроэмуляторах, доступных на рынке. Сам же процесс решения прикладных задач, в том числе и подготовка данных, целиком ложится на плечи пользователя. Данная глава призвана заполнить этот пробел в описании *технологии* нейросетевого анализа.

Для начала выпишем с небольшими комментариями всю технологическую цепочку, т.е. необходимые этапы нейросетевого анализа[1]:

- **Кодирование входов-выходов**: нейросети могут работать только с числами.
- **Нормировка данных**: результаты нейроанализа не должны зависеть от выбора единиц измерения.
- **Предобработка данных**: удаление очевидных регулярностей из данных облегчает нейросети выявление нетривиальных закономерностей.
- **Обучение нескольких нейросетей** с различной архитектурой: результат обучения зависит как от размеров сети, так и от ее начальной конфигурации.

[1] В предположении, что задача определена и база данных для обучения нейросети уже собрана

- **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 перобработка 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 распоряжения даные.

Рассмотрим произвольную компоненту нормированных (предобработанных) данных: $\tilde{x}_i$. Среднее количество информации, приносимой каждым примером $\tilde{x}_i^{\alpha}$, 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 $\tilde{x}_i^{\alpha}$ равномерно распределены в единичном интервале, информация такой переменной максимальна.

Общий принцип предобработки 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 $\{c_1, \ldots, c_n\}$. Но в первом случае эти классы упорядочены - их можно *ранжировать*: $c_1 \succ c_2 \succ \ldots \succ c_n$, 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).

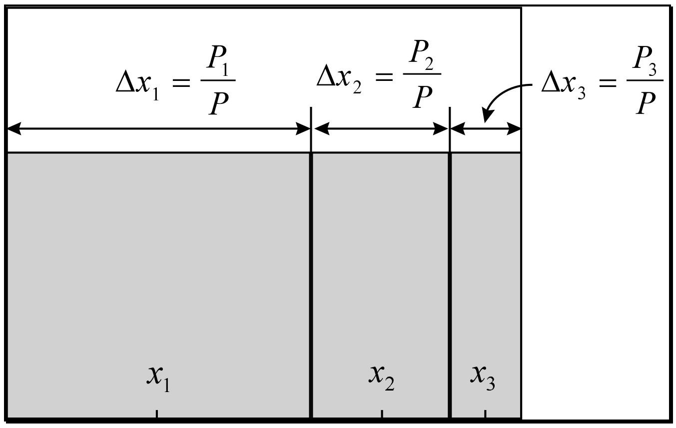

Рисунок 1. Илюстрация способа кодирования кардинальных переменных с учетом количества примеров каждой категории.

Кодирование категориальных переменных

В принципе, категориальные переменные также можно закодировать описанным выше способом, пронумеровав их произвольным образом. Однако, такое навязывание несуществующей упорядоченности только затруднит решение задачи. Оптимальное кодирование не должно искажать структуры соотношений между классами. Если классы не упорядоченны, такова же должна быть и схема кодирования.

Наиболее естественной выглядит и чаще всего используется на практике двоичное кодирование типа "$n \to n$", когда имена n категорий кодируются значениями n бинарных нейронов, причем первая категория кодируется как $(1,0,0,\ldots,0)$, вторая, соответственно - $(0,1,0,\ldots,0)$ и т.д. вплоть до n-ной: $(0,0,0,\ldots,1)$. (Можно использовать биполярную кодировку, в которой нули заменяются на "-1".) Легко убедиться, что в такой симметричной кодировке расстояния между всеми векторами-категориями равны.

Такое кодирование, однако, неоптимально в случае, когда классы представлены существенно различающимся числом примеров. В этом случае, функция распределения значений переменной крайне неоднородна, что существенно снижает информативность этой переменной. Тогда имеет смысл использовать более компактный, но симметричный код $n \to m$, когда имена n классов кодируются m-битным двоичным кодом. Причем, в новой кодировке активность кодирующих нейронов должна быть равномерна: иметь приблизительно одинаковое среднее по примерам значение активации. Это гарантирует одинаковую значимость весов, соответствующих различным нейронам.[2]

В качестве примера рассмотрим ситуацию, когда один из четырех классов (например, класс c_1) некой категориальной переменной представлен гораздо большим числом примеров, чем остальные: $P_1 \gg P_2 \sim P_3 \sim P_4$. Простое кодирование $n \to n$ привело бы к тому, что первый нейрон активировался бы гораздо чаще остальных. Соответственно, веса оставшихся нейронов имели бы меньше возможностей для обучения. Этой ситуации можно избежать, закодировав четыре класса двумя бинарными нейронами следующим образом:

[2] Íå âàäàâàÿñü â ïîäðîáíîñòè, îòìåòèì, ÷òî èíñòðóìåíòîì òàêîãî êîäèðîâàíèÿ ìîãóò ñëóæèòü *öèêëè÷åñêèå* êîäû, â êîòîðûõ êàæäîå êîäîâîå ñëîâî âñòðå÷àåòñÿ âìåñòå ñî âñåìè ñäâèãàìè öèêëè÷åñêèìè ñäâèãàìè.

$c_1 = (0,0), c_2 = (1,0), c_3 = (0,1), c_4 = (1,1)$, 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.

Линейная нормировка оптимальна, когда значения переменной x_i плотно заполняют определенный интервал. Но подобный "прямолинейный" подход применим далеко не всегда. Так, если в данных имеются относительно редкие выбросы, намного превышающие типичный разброс, именно эти выбросы определят согласно предыдущей формуле масштаб нормировки. Это приведет к тому, что основная масса значений нормированной переменной $\widetilde{x}_i$ сосредоточится вблизи нуля: $|\widetilde{x}_i| << 1$.

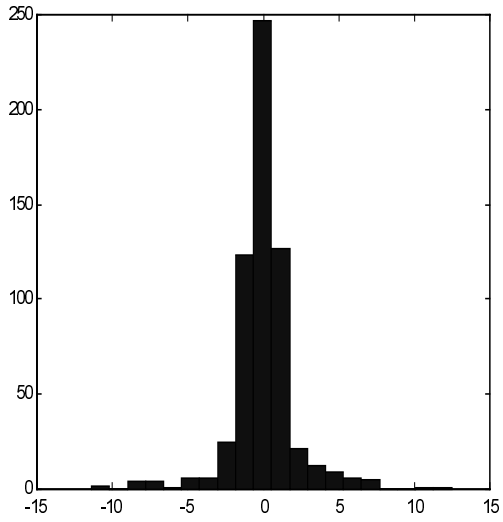

Рисунок 2. Гистограмма значений переменной при наличии редких, но больших по амплитуде отклонений от среднего

Гораздо надежнее, поэтому, ориентироваться при нормировке не на экстремальные значения, а на типичные, т.е. статистические характеристики данных, такие как среднее и дисперсия[3]:

$$\widetilde{x}_i = \frac{x_i - \bar{x}_i}{\sigma_i}, \qquad \bar{x}_i \equiv \frac{1}{P}\sum_{\alpha=1}^{P} x_i^{\alpha}, \quad \sigma_i^2 \equiv \frac{1}{P-1}\sum_{\alpha=1}^{P}\left(x_i^{\alpha} - \bar{x}_i\right)^2 .$$

В этом случае основная масса данных будет иметь единичный масштаб, т.е. типичные значения всех переменных будут сравнимы (см. Рисунок 2).

Однако, теперь нормированные величины не принадлежат гарантированно единичному интервалу, более того, максимальный разброс значений $\widetilde{x}_i$ заранее не известен. Для входных данных это может быть и не важно, но выходные переменные будут использоваться в качестве эталонов для выходных нейронов. В случае, если выходные нейроны - сигмоидные, они могут принимать значения лишь в единичном диапазоне. Чтобы установить соответствие между обучающей выборкой и нейросетью в этом случае необходимо ограничить диапазон изменения переменных.

Линейное преобразование, как мы убедились, неспособно отнормировать основную массу данных и одновременно ограничить диапазон возможных значений этих данных. Естественный

[3] Â âûðàæåíèè äëÿ äèñïåðñèè çíàìåíàòåëü ðàâåí $(P\text{-}1)$, à íå P èç-çà ôëóêòóàöèé *îöåíêè* ñðåäíåãî $\bar{x}_i$

выход из этой ситуации - использовать для предобработки данных функцию активации тех же нейронов. Например, нелинейное преобразование

$$\widetilde{x}_i = f\left(\frac{x_i - \bar{x}_i}{\sigma_i}\right), \qquad f(a) = \frac{1}{1+e^{-a}}$$

нормирует основную массу данных одновременно гарантируя, что $\widetilde{x}_i \in [0, 1]$ (см. Рисунок 3).

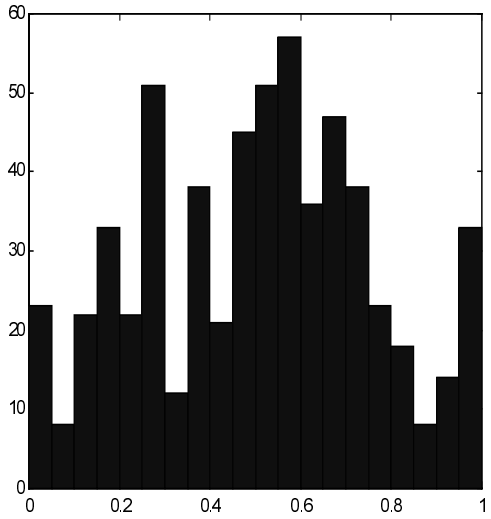

Рисунок 3. Нелинейная нормировка, использующая логистическую функцию активации $f(a) = \left(1+e^{-a}\right)^{-1}$

Как видно из приведенного выше рисунка, распределение значений после такого нелинейного преобразования гораздо ближе к равномерному.

До сих пор мы старались максимизировать энтропию каждого входа (выхода) по отдельности. Но, вообще говоря, можно добиться гораздо большего максимизируя их *совместную* энтропию. Рассмотрим эту технику на примере совместной нормировки входов, подразумевая, что с таким же успехом ее можно применять и для выходов а также для всей совокупности входов-выходов.

Совместная нормировка: *выбеливание* входов

Если два входа статистически не независимы, то их совместная энтропия меньше суммы индивидуальных энтропий: $H\left(\widetilde{x}_i \widetilde{x}_j\right) \le H\left(\widetilde{x}_i\right) + H\left(\widetilde{x}_i\right)$. Поэтому добившись статистической независимости входов мы, тем самым, повысим информационную насыщенность входной информации. Это, однако, потребует более сложной процедуры совместной нормировки входов.

Вместо того, чтобы использовать для нормировки индивидуальные дисперсии, будем рассматривать входные данные в совокупности. Мы хотим найти такое линейное преобразование, которое максимизировало бы их совместную энтропию. Для упрощения

задачи вместо более сложного условия статистической независимости потребуем, чтобы новые входы после такого преобразования были декоррелированы[4]. Для этого рассчитаем средний вектор и ковариационную матрицу данных по формулам:

$$\bar{\mathbf{x}} \equiv \frac{1}{P}\sum_{\alpha=1}^{P}\mathbf{x}^{\alpha}, \quad \Sigma_{ij}^{X} \equiv \frac{1}{P-1}\sum_{\alpha=1}^{P}\left(x_i^{\alpha}-\bar{x}_i\right)\left(x_j^{\alpha}-\bar{x}_j\right)$$

Затем найдем линейное преобразование, диагонализующее ковариационную матрицу. Соответствующая матрица составлена из столбцов - собственных векторов ковариационной матрицы:

$$\sum_j \Sigma_{ij}^{X} U_{jk} = \lambda_k U_{ik}$$

Легко убедиться, что линейное преобразование, называемое *выбеливанием* (whitening)

$$\tilde{x}_i = \left(x_k - \bar{x}_k\right) U_{ki} \Big/ \sqrt{\lambda_i}$$

превратит все входы в некоррелированные величины с нулевым средним и единичной дисперсией.

Если входные данные представляют собой многомерный эллипсоид, то графически выбеливание выглядит как растяжение этого эллипсоида по его главным осям (Рисунок 4).

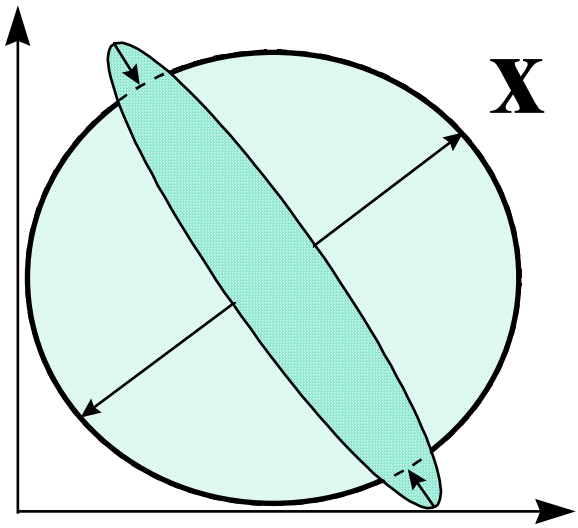

Рисунок 4. Выбеливание входной информации: повышение информативности входов за счет выравнивания функции распределения

Очевидно, такое преобразование увеличивает совместную энтропию входов, т.к. оно выравнивает распределение данных в обучающей выборке.

---

[4] Äåêîððåëèðîâàííûå âåëè÷èíû x_i èìåþò íóëåâûå êðîññ-êîððåëÿöèè: $\overline{\left(x_i-\bar{x}_i\right)\left(x_j-\bar{x}_j\right)}=0, \quad \forall i \neq j$

Понижение размерности входов

Сильной стороной нейроанализа является возможность получения предсказаний при минимуме априорных знаний. Поскольку заранее обычно неизвестно насколько полезны те или иные входные переменные для предсказания значений выходов, возникает соблазн увеличивать число входных параметров, в надежде на то, что сеть сама определит какие из них наиболее значимы. Однако, как это уже обсуждалось в Главе 3, сложность обучения персептронов быстро возрастает с ростом числа входов (а именно - как куб размерности входных данных $C \propto d^3$). Еще важнее, что с увеличением числа входов страдает и точность предсказаний, т.к. увеличение числа весов в сети снижает предсказательную способность последней (согласно предыдущим оценкам: $\varepsilon \geq \sqrt{d/P}$).

Таким образом, количество входов приходится довольно жестко лимитировать, и выбор наиболее информативных входных переменных представляет важный этап подготовки данных для обучения нейросетей. Глава 4 специально посвящена использованию для этой цели самих нейросетей, обучаемых без учителя. Не стоит, однако, пренебрегать и традиционными, более простыми и зачастую весьма эффективными методами линейной алгебры.

Один из наиболее простых и распространенных методов понижения размерности - использование главных компонент входных векторов. Этот метод позволяет не отбрасывая конкретные входы учитывать лишь наиболее значимые комбинации их значений.

Понижение размерности входов методом главных компонент

Собственные числа матрицы ковариаций λ_i, фигурировавшие в предыдущем разделе, являются квадратами дисперсий вдоль ее главных осей. Если между входами существует линейная зависимость, некоторые из этих собственных чисел стремятся к нулю. Таким образом, наличие малых λ_i свидетельствует о том, что реальная размерность входных данных объективно ниже, чем число входов. Можно задаться некоторым пороговым значением ε и ограничиться лишь теми главными компонентами, которые имеют $\lambda \geq \varepsilon \lambda_{\max}$. Тем самым, достигается понижение размерности входов, при минимальных потерях точности представления входной информации.

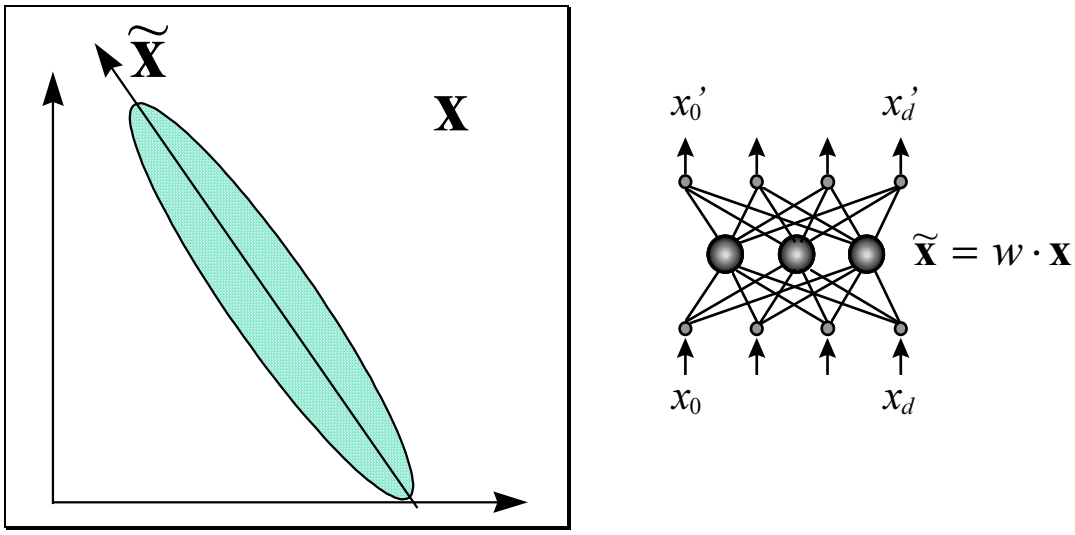

Рисунок 5. Понижение размерности входов методом главных компонент.

Восстановление пропущенных компонент данных

Главные компоненты оказываются удобным инструментом и для восстановления пропусков во входных данных. Действительно, метод главных компонент дает наилучшее линейное приближение входных данных меньшим числом компонент: $\tilde{\mathbf{x}} = w\mathbf{x}$ (Здесь мы, как и прежде, для учета постоянного члена включаем фиктивную нулевую компоненту входов, всегда равную единице - см. Рисунок 5, где справа показана нейросетевая интерпретация метода главных компонент. Таким образом, w - это матрица размерности $N \times (d+1)$). Восстановленные по N главным компонентам данные из обучающей выборки $\mathbf{x}'^{\alpha} = w^T \tilde{\mathbf{x}} = w^T w \mathbf{x}^{\alpha}$ имеют наименьшее среднеквадратичное отклонение от своих прототипов $\mathbf{x}^{\alpha}$. Иными словами, при отсутствии у входного вектора k компонент, наиболее вероятное положение этого вектора - на гиперплоскости первых $N = (d-k)$ главных компонент. Таким образом, для восстановленного вектора имеем: $\mathbf{x}'^{\alpha} = w^T \tilde{\mathbf{x}} = w^T w \mathbf{x}'^{\alpha}$, причем для известных компонент $\mathbf{x}'^{\alpha} = \mathbf{x}^{\alpha}$.

Пусть, например, у вектора $\mathbf{x}^{\alpha}$ неизвестна всего одна, k-я координата. Ее значение находится из оставшихся по формуле:

$$x'^{\alpha}_{k} = \left[w^T w\right]_{ki} x_i^{\alpha} \Big/ \left(1 - \left[w^T w\right]_{kk}\right),$$

где в числителе учитываются лишь известные компоненты входного вектора $\mathbf{x}^{\alpha}$.

В общем случае восстановить неизвестные компоненты (с индексами из множества K) можно с помощью следующей итеративной процедуры (см. Рисунок 6):

$$x_i^{n+1} = x_i^{\alpha}, \qquad i \notin K$$
$$x_i^{n+1} \to w_{ki} w_{kj} x_j^{n}, \quad i \in K$$

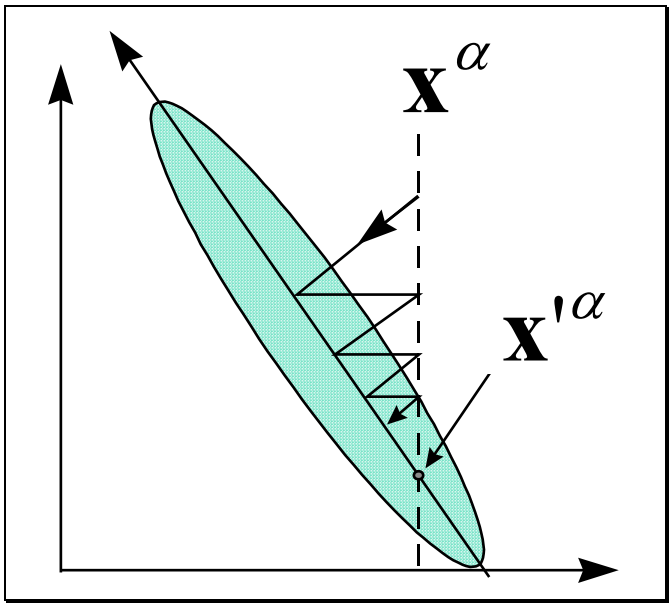

Рисунок 6. Восстановление пропущенных значения с помощью главных компонент. Пунктир - возможные значения исходного вектора с $k = 1$ неизвестными координатами. Наиболее вероятное его значение - на пересечении с $N = (d - k) = 2 - 1 = 1$ первыми главными компонентами

Понижение размерности входов с помощью нейросетей

Для более глубокой предобработки входов можно использовать все богатство алгоритмов самообучающихся нейросетей, о которых шла речь ранее. В частности, для оптимального понижения размерности входов можно воспользоваться методом *нелинейных главных компонент* (см. Рисунок 7).

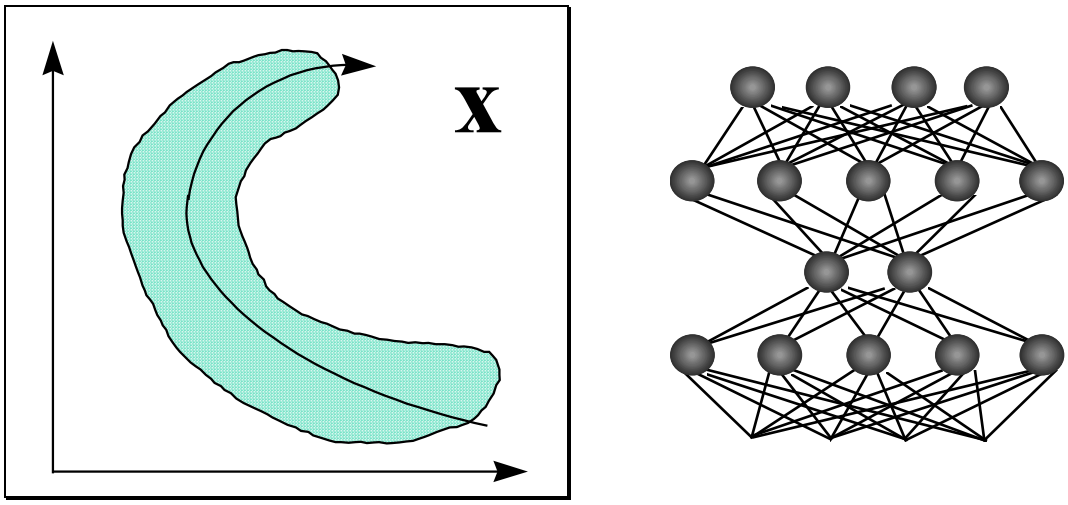

Рисунок 7. Понижение размерности входов методом нелинейных главных компонент

Такие сети с узким горлом также можно использовать для восстановления пропущенных значений - с помощью итерационной процедуры, обобщающей линейный вариант метода главных компонент (см. Рисунок 8).

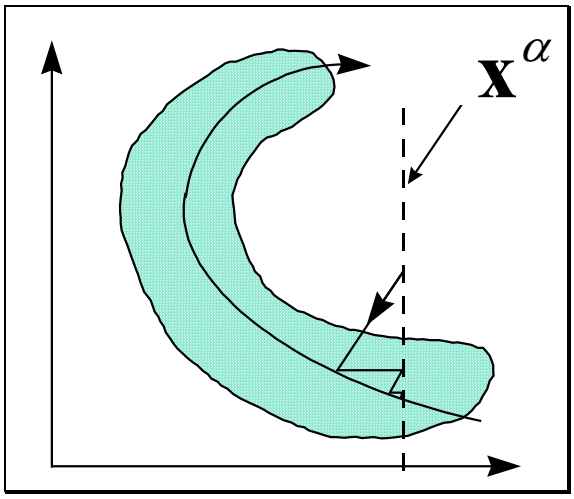

Рисунок 8. Восстановление пропущенных компонент данных с помощью нелинейных главных компонент

Однако, такую глубокую "предобработку" уже можно считать самостоятельной нейросетевой задачей. И мы не будем дале углубляться в этот вопрос.

Квантование входов

Более распространенный вид нейросетевой предобработки данных - *квантование входов*, использующее слой соревновательных нейронов (см. Рисунок 9).

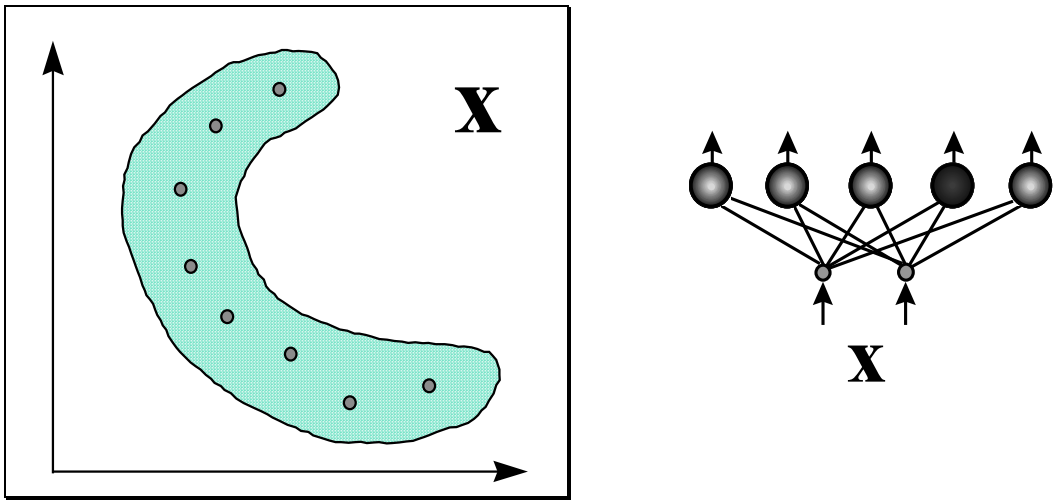

Рисунок 9. Понижение разнообразия входов методом квантования (кластеризации)

Нейрон-победитель является прототипом ближайших к нему входных векторов. Квантование входов обычно не сокращает, а наоборот, существенно увеличивает число входных переменных. Поэтому его используют в сочетании с простейшим линейным дискриминатором - однослойным персептороном. Получающаяся в итоге гибридная нейросеть, предложенная Нехт-Нильсеном в 1987 году, обучается послойно: сначала соревновательный слой кластеризует входы, затем выходным весам присваиваются значения выходной функции, соответствующие данному кластеру. Такие сети позволяют относительно быстро получать грубое - кусочно-постоянное - приближение аппроксимируемой функции (см. Рисунок 10).

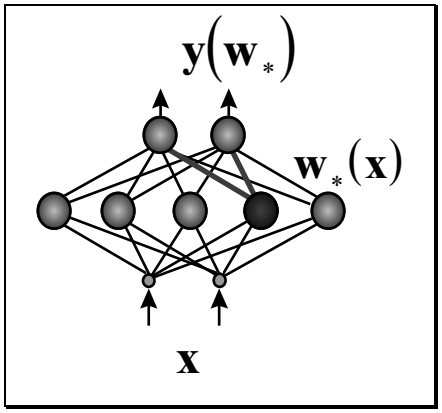

Рисунок 10. Гибридная сеть с соревновательным слоем, дающая кусочно-постоянное приближение функций

Особенно полезны кластеризующие сети для восстановления пропусков в массиве обучающих данных. Поскольку работа соревновательного слоя основана на сравнении расстояний между данными и прототипами, осутствие у входного вектора $\mathbf{x}^{\alpha}$ некоторых компонент не препятствует нахождению прототипа-победителя: сравнение ведется по оставшимся компонентам $i \notin K$:

$$\left|\mathbf{x}^{\alpha} - \mathbf{w}_k\right| = \sqrt{\sum_{i \notin K} \left(x_i^{\alpha} - w_{ki}\right)^2} .$$

При этом все прототипы $\mathbf{w}_k$ находятся в одинаковом положении. Рисунок 11 иллюстрирует эту ситуацию.

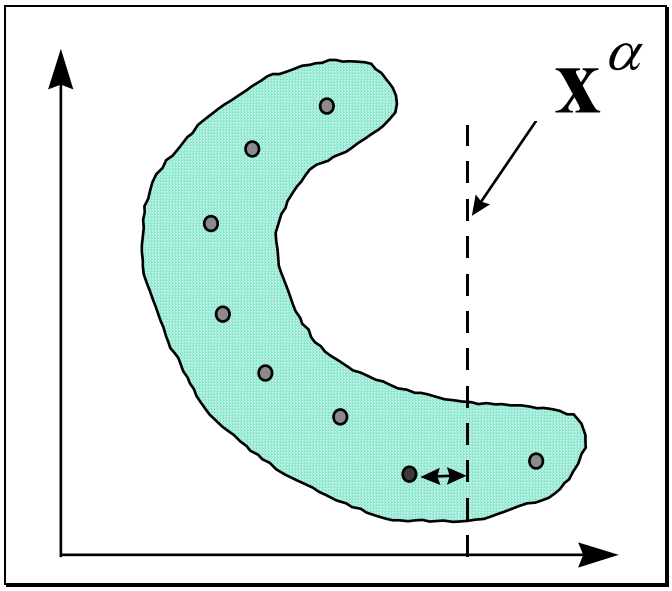

Рисунок 11. Наличие пропущенных компонент не препятствует нахождению ближайшего прототипа по оставшимся компонентам входного вектора $\mathbf{x}^{\alpha}$

Таким образом, слой квантующих входные данные нейронов нечувствителен к пропущенным компонентам, и может служить “защитным экраном” для минимизации последствий от наличия пропусков в обучающей базе данных.

Отбор наиболее значимых входов

До сих пор мы старались лишь представить имеющуюся входную информацию наилучшим - наиболее информативным - образом. Однако, рассмотренные выше методы предобработки входов никак не учитывали зависимость выходов от этих входов. Между тем, наша задача как раз и состоит в выборе входных переменных, наиболее значимых для предсказаний. Для такого более содержательного отбора входов нам потребуются методы, позволяющие оценивать *значимость* входов.

Линейная значимость входов

Легче всего оценить значимость входов в линейной модели, предполагающей линейную зависимость выходов от входов:

$$\left(y_j^{\alpha} - \bar{y}_j\right) = \sum_k w_{jk}\left(x_k^{\alpha} - \bar{x}_k\right)$$

Матрицу весов w_{jk} можно получить, например, обучением простейшего - однослойного персептрона с линейной функцией активации. Допустим теперь, что при определении выходов мы опускаем одну, для определенности - k-ю компоненту входов, заменяя ее средним значением этой переменной. Это приведет к огрублению модели, т.е. возрастании ошибки на величину:

$$\delta E = \frac{1}{P}\sum_{j\alpha}\left(y_j^\alpha - \hat{y}_j^\alpha\right)^2 = \frac{1}{P}\sum_{j\alpha}\left(w_{jk}\left(x_k^\alpha - \bar{x}_k\right)\right)^2 = \overline{\left(x_k^\alpha - \bar{x}_k\right)^2}\sum_j\left(w_{jk}\right)^2 = \sum_j\left(w_{jk}\right)^2 .$$

(Полагая, что данные нормированны на их дисперсию.) Таким образом, *значимость* k-го входа определяется суммой квадратов соответствующих ему весов.

Особенно просто определить значимость выбеленных входов. Для достаточно просто вычислить взаимную корреляцию входов и выходов:

$$\Sigma_{ij}^{XY} \equiv \frac{1}{P-1}\sum_{\alpha=1}^{P}\left(x_i^\alpha - \bar{x}_i\right)\left(y_j^\alpha - \bar{y}_j\right).$$

Действительно, при линейной зависимости между входами и выходами имеем: $\Sigma_{ij}^{XY} \equiv \frac{1}{P-1}\sum_{\alpha=1}^{P}\sum_k\left(x_i^\alpha - \bar{x}_i\right)w_{jk}\left(x_k^\alpha - \bar{x}_k\right) = \sum_k w_{jk}\Sigma_{ki}^X$.

Таким образом, в общем случае для получения матрицы весов требуется решить систему линейных уравнений. Но для предварительно выбеленных входов имеем: $\Sigma_{ki}^X = \delta_{ki}$, так что в этом случае матрица кросс-корреляций просто совпадает с матрицей весов обученного линейного персептрона: $\Sigma_{ij}^{XY} = w_{ji}$.

Резюмируя, значимость входов в предположении о приблизительно линейной зависимости между входными и выходными переменными для выбеленных входов пропорциональна норме столбцов матрицы кросс-корреляций: $\sum_i\left(\Sigma_{ij}^{XY}\right)^2$.

Не следует, однако, обольщаться существованием столь простого рецепта определения значимости входов. Линейная модель может быть легко построена и без привлечения нейросетей. Реальная сила нейроанализа как раз и состоит в возможности находить более сложные нелинейные зависимости. Более того, для облегчения собственно нелинейного анализа рекомендуется заранее освободиться от тривиальных линейных зависимостей - т.е. в качестве выходов при обучении подавать разность между выходными значениями и их линейным приближением. Это увеличит "разрешающую способность" нейросетевого моделирования (см. Рисунок 12).

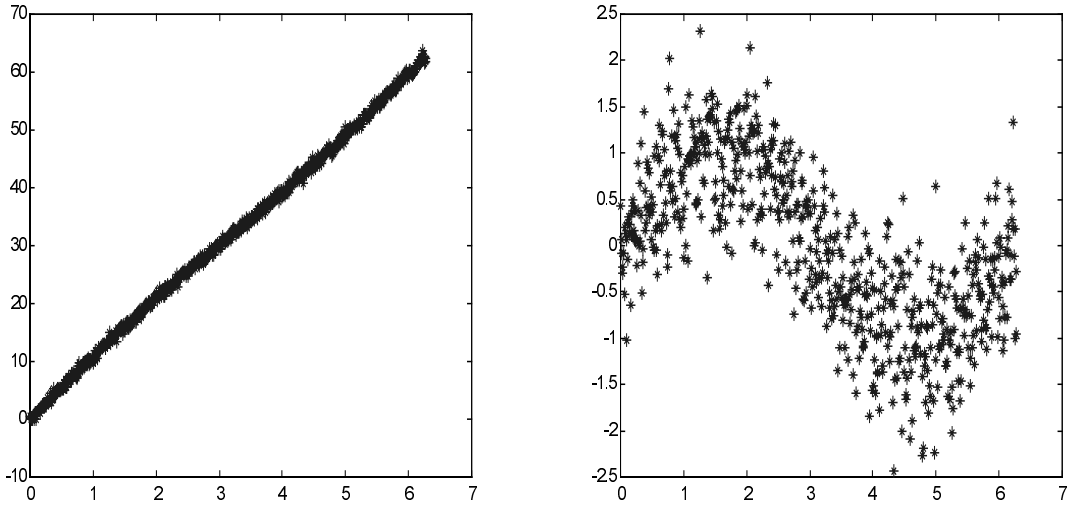

Рисунок 12. Выявление нелинейной составляющей функции $y = 10x + \sin(x) + 0.5\eta$ **после вычитания линейной зависимости** $y = 10x$ **. (Здесь** η **- гауссовый случайный шум)**

Для определения "нелинейной" значимости входов - после вычитания линейной составляющей, изложенный выше подход неприменим. Здесь надо привлекать более изощренные методики. К описанию одной из них, алгоритмам box-counting, мы и переходим.

Нелинейная значимость входов. Box-counting алгоритмы

Алгортимы box-counting, как следует из самого их названия, основаны на подсчете чисел заполнения примерами P_i ячеек (boxes), на которые специально для этого разбивается пространство переменных $\mathbf{X} \otimes \mathbf{Y}$. Эти числа заполнения используются для оценки плотности вероятности распределения примеров по ячейкам. Набор вероятностей $p_i = P_i / P$ дает возможность рассчитать любую статистическую характеристику набора данных обучающей выборки.

Для определения значимости входов нам потребуется оценить предсказуемость выходов, обеспечиваемую данным набором входных переменных. Чем выше эта предсказуемость - тем лучше соответствующий набор входов. Таким образом, метод box-counting предоставляет в наше распоряжение технологию отбора наиболее значимых *признаков* для нейросетевого моделирования, технологию оптимизации входного *пространства признаков*.

Согласно общим положениям теории информации, мерой предсказуемости случайной величины X является ее энтропия, $H(X)$, определяемая как среднее значение ее логарифма. В методике box-counting энтропия приближенно оценивается по набору чисел заполнения ячеек, на которые разбивается интервал ее возможных значений: $H(X) = -\sum_i p_i^X \log p_i^X$. Качественно, энтропия есть логарифм эффективного числа заполненных ячеек $H(X) \cong \log N_X$ (см. Рисунок 13). Чем больше энтропия переменной, тем менее предсказуемо ее значение. Когда все значения примеров сосредоточены в одной ячейке - их энтропия равна нулю, т.к. положение данных определено (с данной степенью точности).

Равномерному заполнению ячеек соответствует максимальная энтропия - наибольший разброс возможных значений переменной.

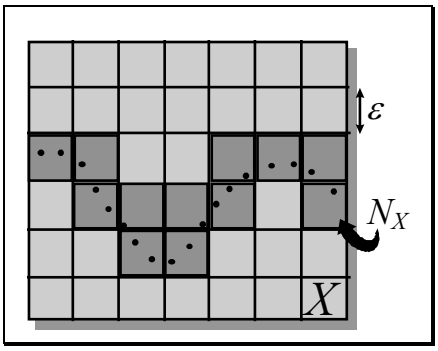

Рисунок 13. Смысл энтропии - эффективное число заполненных данными ячеек

Предсказуемость случайного вектора $\mathbf{Y}$, обеспечиваемое знанием другой случайной величины $\mathbf{X}$, дается кросс-энтропией:

$$I(\mathbf{Y},\mathbf{X}) = H(\mathbf{Y}) + H(\mathbf{X}) - H(\mathbf{X},\mathbf{Y}) = H(\mathbf{Y}) - H(\mathbf{Y}|\mathbf{X})$$

Качественно, кросс-энтропия равна логарифму отношения типичного разброса значений переменной $\mathbf{Y}$ к типичному разбросу этой переменной, но при известном значении переменной $\mathbf{X}$ (см. Рисунок 14):

$$I(\mathbf{Y},\mathbf{X}) = \log \frac{N_X N_Y}{N_{XY}} .$$

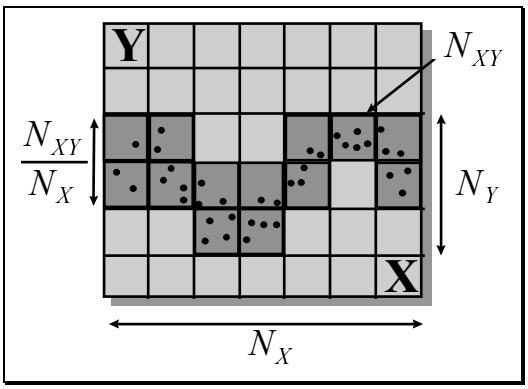

Рисунок 14. Иллюстрация к понятию кросс-энтропии: N_{XY} - полное число ячеек в объединенном пространстве $\mathbf{X} \otimes \mathbf{Y}$, N_X - число проекций ячеек на пространство $\mathbf{X}$, N_{XY}/N_X - характерный разброс по оси $\mathbf{Y}$ при фиксированном $\mathbf{X}$, N_Y - характерный разброс всех данных по оси $\mathbf{Y}$.

Чем больше кросс-энтропия, тем больше определенности вносит знание значения $\mathbf{X}$ в предсказание значения переменной $\mathbf{Y}$.

Описанный выше энтропийный анализ не использует никаких предположений о характере зависимости между входными и выходными переменными. Таким образом, данная методика дает наиболее общий рецепт определения значимости входов, позволяя также оценивать степень предсказуемости выходов.

В принципе, качество предсказаний и, соответственно, значимость входной информации определяется, в конечном итоге, в результате обучения нейросети, которая, к тому же, дает решение в явном виде. Однако, как мы знаем, обучение нейросети - довольно сложная вычислительная задача (требующая $\sim P^3$ операций). Между тем, существуют эффективные алгоритмы быстрого подсчета кросс-энтропии (с вычислительной сложностью $\sim P\log P$), намного более экономные, чем обучение нейросетей. Значение методики box-counting состоит в том, что не находя самого решения, она позволяет быстро предсказать качество этого прогноза. Поэтому эта методика может быть положена в основу предварительного отбора входной информации на этапе предобработки данных.

Формирование оптимального пространства признаков

В типичной ситуации набор выходных, прогнозируемых, переменных фиксирован, и требуется подобрать наилучшую комбинацию ограниченного числа входных величин. Оценка значимости входов позволяет построить процедуру систематического предварительного подбора входных переменных - до этапа обучения нейросети. Для иллюстрации опишем две возможные стратеги автоматического формирования признакого пространства.

Последовательное добавление наиболее значимых входов

Один из наиболее очевидных способов формирования пространства признаков с учетом реальной значимости входов - постепенный подбор наиболее значимых входов в качестве очередных признаков. В качестве первого признака выбирается вход с наибольшей индивидуальной значимостью:

$$. \; k_1 = \arg\max_k \left\{ I\left(\mathbf{Y}, X_k\right) \right\} .$$

Вторым признаком становится вход, обеспечивающий наибольшую предсказуемость в паре с уже выбранным:

$$k_2 = \arg\max_k \left\{ I\left(\mathbf{Y}, X_{k_1} X_k\right) \right\},$$

и так далее. На каждом следующем этапе добавляется вход, наиболее значимый в компании с выбранными ранее входами:

$$k_n = \arg\max_k \left\{ R_k^{(n)} = I\left(\mathbf{Y}, X_{k_1} \ldots X_{k_{n-1}} X_k\right) \right\} .$$

Такая процедура не гарантирует нахождения наилучшей комбинации входов, т.е. дает субоптимальный набор признаков, т.к. реально рассматривается лишь очень малая доля от полного числа комбинаций входов, и значимость каждого нового признака зависит от сделанного прежде выбора. Полный перебор, однако, практически неосуществим: выбор

оптимальной комбинации n входов при полном их числе d_X требует перебора $\binom{d_X}{n} \sim \frac{(d_X)^n}{n!}$ комбинаций.

Другим недостатком описанного выше подхода является необходимость подсчета кросс-энтропии в пространстве все более высокой размерности по мере увеличения числа отобранных признаков. Ниже описана процедура, свободная от этого недостатка, основанная на применении методики box-counting лишь в низкоразмерных пространствах (а именно - с размерностью $d_Y + 1$).

Формирование признакого пространства методом ортогонализации

Следующая систематическая процедура способна итеративно выделять наиболее значимые признаки, являющиеся линейными комбинациями входных переменных: $\widetilde{\mathbf{X}} = \mathbf{W} \cdot \mathbf{X}$ (подмножество входов является частным случаем линейной комбинации, т.е. формально можно найти лучшее решение, чем то, что доступно путем отбора наиболее значимых комбинаций входов).

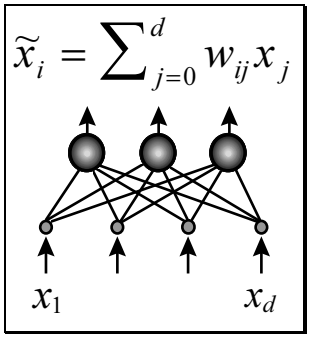

Рисунок 15. Выбор наиболее значимых линейных комбинаций входных переменных.

Для определения значимости каждой входной компоненты будем использовать каждый раз индивидуальную значимость этого входа[5]: $I(\mathbf{Y}, X_k)$.

Подсчитав индивидуальную значимость входов, находим направление в исходном входном пространстве, отвечающее наибольшей (нелинейной) чувствительности выходов к изменению входов. Это градиентное направление определит первый вектор весов, дающий первую компоненту пространства признаков:

$$w_{1k} = I(\mathbf{Y}, X_k).$$

Следующую компоненту будем искать аналогично первой, но уже в пространстве перпендикулярном выбранному направлению, для чего спроектируем все входные вектора в это пространство:

[5] Ýòî, êîíå÷íî, êîíêðåòíî. Âîîáùå ãîâîðÿ, â ïîëíûõàäàïòèâíûõ ñëåäîâàòåëüíûõ áû èñïîëüçîâàòü ñòðîãîå ïðåäåëåíèå çíà÷èìîñòè k-ãî âõîäà: $R_k = I(\mathbf{Y}, \mathbf{X}) - I(\mathbf{Y}, \mathbf{X} \setminus X_k)$, îäíàêî ýòî ïîäðàçóìåâàåò ïðîâåäåíèå box-counting â âûñîêîðàçìåðíîì ïðîñòðàíñòâå, ÷åãî ìû êàê ðàç è õîòèì èçáåæàòü.

$$\mathbf{X}^{(1)} = \mathbf{X} - \left(\mathbf{w}_1\mathbf{X}\right)\cdot\mathbf{w}_1 .$$

В этом пространстве можно опять подсчитать "градиент" предсказуемости, определив индивидуальную значимость спроектированных входов, и так далее. На каждом следующем этапе подсчитывается индивидуальная значимость $I\left(\mathbf{Y}, X_k^{(n)}\right)$ для проекции входов

$$\mathbf{X}^{(n)} = \mathbf{X} - \left(\mathbf{w}_1\mathbf{X}\right)\cdot\mathbf{w}_1 - \ldots - \left(\mathbf{w}_n\mathbf{X}\right)\cdot\mathbf{w}_n ,$$

что не требует повышения размерности box-counting анализа. Таким образом, описанная выше процедура позволяет формировать пространство признаков произвольной размерности - без потери точности.

Заключение

Конечно, описанными выше методиками не исчерпывается все разнообразие подходов к ключевой для нейро-анализа проблеме формирования пространства признаков. Мы не упомянули, в частности, генетические алгоритмы, которые в совокупностью с методикой box-counting являются весьма перспективным инструментом. Ничего не было сказано также о методике разделения независимых компонент (*blind signal separation*), расширяющей анализ главных компонент. Необъятного не объять. Главное, чтобы за деталями не затерялся основополагающий принцип предобработки данных: снижение существующей избыточности всеми возможными способами. Это повышает информативность примеров и, тем самым, качество нейропредсказаний.

Литература

Bishop C.M. (1995) *Neural Networks and Pattern Recognition*. Oxford Press.

Voss R.F. (1986) "Random Fractals, Characterization and Measurement", in *Scaling Phenomena in Disordered Systems*, R.Pynn and A.Skjeltorp, Eds., Plenum, NewYork.

Keller J.M., Chen S., and Crownover R.M. (1988) "Texture Description and Segmentation through Fractal Geometry". *Computer Vision, Graphics, and Image Processing*, **45**, 150-166.

Pineda F.J., and Sommerer J.C. (1994) "Estimating Generalized Dimensions and Choosing Time Delays: A Fast Algorithm", in *Time Series Prediction*, A.S.Weigend, N.A.Gershenfeld, Eds., Addison-Wesley, p.367-385.

Rissanen J. (1990) "Complexity of Models", in *Complexity, Entropy and the Physics of Information*, Ed. W.H.Zurek; Addison-Wesley, Redwood City, California, p. 117-125.