Курс машинного обучения ЛНМО

Фадеев А.В.

зима - весна 2020

1 Что такое машинное обучение и data science?

Человека всегда интересовали вопросы о причинно-следственных связях между явлениями. Что влияет на завтрашнюю погоду? Каким законам подчиняются квантовые частицы? По каким закономерностям изменяются курсы валют? Или даже более приземлённые вопросы вроде: "какого цвета галстук надеть на завтрашний ужин?". Все такие вопросы можно записать в общем виде на математическим языке:

$$f(X) = y \tag{1}$$

Здесь присутствуют три важные компоненты: f, X, y. Поговорим о них подробнее.

Наверняка вы множество раз встречали такую запись, например, в таком виде:

$$y = 2x^2 + 1$$

Это уравнение нам говорит, что некоторая величина y зависит от некоторого x указанным выше образом. Имея в распоряжении такое уравнение, нам ничего не стоит для каждого x однозначно ответить на вопрос, какой же y ему соответствует. Т.е. из трёх компонент уравнения (1) здесь есть явно заданная функция f, а y можно найти для любого x:

$$f,\ X \longrightarrow y$$

Вы скорее всего встречались с задачей *обращения* функции, когда по известным y и f необходимо найти X:

$$f,\ y \longrightarrow X$$

Такая задача обычно сложнее первой и требует дополнительный шаг по поиску *обратной функции* (или её аналога):

$$f, y \longrightarrow f^{-1},\ y \longrightarrow X$$

А теперь рассмотрим самый интересный случай, когда нам неизвестен вид функции f, но мы откуда-то знаем, что определённым X соответствуют конкретные значения y:

$$X,\ y \longrightarrow f$$

Такая ситуация обычно возникает в ходе экспериментов, когда с помощью измерительных приборов измеряется отклик системы (y) на значение некоторых параметров (X). Кроме того, все перечисленные в первом абзаце вопросы можно сформулировать в виде такой задачи.

Предположим, например, что мы хотим научиться предсказывать погоду на завтрашний день. Наша мечта - найти такую функцию f, которая бы возвращала все интересующие нас параметры завтрашней погоды для любой точки Земли и любого момента времени, например, такие:

1. Температура ($T \in \mathbb{R}$)
2. Скорость ветра ($v \in \mathbb{R}$)
3. Направление ветра ($w \in \mathbb{R}^3$)
4. Влажность ($h \in \mathbb{R}$)
5. Давление ($p \in \mathbb{R}$)
6. Наличие осадков ($r \in \{True, False\}$)

Т.е. если бы кто-то нам сообщил аналитический вид функции:

$$f(latitude, longitude, time) = (T, v, w, h, p, r),$$

мы были бы очень счастливы!

Конечно, никто не знает как выглядит функция f, однако за многолетнюю историю человечества накопилось большое количество наблюдений за природой (*данных*) в виде X и y. Поэтому возникает идея исследовать эти данные в надежде, что нам удастся догадаться до истинной закономерности, которую придумала Природа.
Конечно, какую бы закономерность в данных мы бы ни увидели, мы не сможем доказать, что именно эта (и никакая другая!) закономерность является *истинной*. Мы можем лишь надеяться, что получится найти такую функцию f, которая бы *согласовывалась* со всеми наблюдениями (в том числе и теми, которые будут в будущем).
Конечно, и на этом пути есть ряд существенных трудностей, например, таких:

- Все данные X и y, которые мы будем пытаться объяснить с помощью f, не являются абсолютно точными. Обычно они получены человеком/измерительным прибором, т.е. всегда присутствует некоторая погрешность измерения и, может быть, есть даже совершенно некорректные данные, полученные по невнимательности/из-за технических сбоев приборов/скачков напряжения и т. д.

- Вообще говоря, мы не знаем даже от каких параметров X на самом деле зависит предсказываемая величина y. Так, в примере выше, поиск функции f в предложенном виде уже допускает весьма смелое предположение, что все 6 параметров погоды можно однозначно посчитать для заданных координат и времени суток

(что, очевидно, не так: погода в моем дворе в 12:00 всегда разная!). Разумно было бы учесть ещё множество вводных параметров:

$$f(latitude, longitude, time, T_{day-1}, T_{day-2}, etc...) = (T, v, w, h, p, r),$$

Какие именно это должны быть параметры, сколько их, нам, конечно же, заранее неизвестно. Вполне может быть, что между данными, которыми мы располагаем, вообще нет никакой взаимосвязи.

- Допустим, нам удалось найти такую функцию f, которая превосходно бы объяснила наши экспериментальные данные. Как мы можем быть уверены, что *в будущем*, т.е. на новых данных эта функция по-прежнему будет точно описывать закон? Неужели, её придется постоянно пересматривать по мере поступления новых данных?

На все эти вопросы (и многие другие) мы постепенно будем учиться отвечать.

Вероятно, читатель уже догадался, что нам предстоит работать в сфере некорректно поставленных задач, т.е. таких, для которых не существует строгого математического решения. Нам предстоит принимать важные решения исходя из эвристических соображений. Чистых математиков призываю не отступать, а, наоборот воодушевиться - Природа бросает нам очень серьёзный вызов!

Оказывается, подобные задачи можно очень успешно решать с помощью компьютера. Начнём с того, что все данные X и y обычно имеют оцифрованный вид и хранятся в памяти компьютеров. Кроме того, благодаря высокой скорости вычислений, появляется возможность быстро исследовать терабайты данных и выявлять их закономерности. Машинное обучение - это просто набор эффективных приемов, рекомендаций и алгоритмов, позволяющих с помощью компьютера находить решения задач, связанных в первую очередь, с установлением закономерностей между данными.

В заключение, чтобы ещё больше подогреть ваш интерес, отмечу, что с помощью методов машинного обучения можно решать такие странные на первый взгляд задачи (попробуйте самостоятельно придумать им интерпретацию!):

$$X \longrightarrow y, \tag{2}$$

$$X \longrightarrow f, \tag{3}$$

$$\emptyset \longrightarrow X. \tag{4}$$

2 Самый первый алгоритм: линейная регрессия

Линейная регрессия - это очень простой, но в то же время гибкий в опытных руках инструмент анализа данных. Часто его недооценивают, ошибочно полагая, что слово "линейный" в названии алгоритма существенно ограничивает его описательную способность.
На примере этого фундаментального алгоритма мы сформулируем основные концепции и термины машинного обучения, которые в дальнейшем нам будут постоянно встречаться.

Задача, которую предстоит решить:

Для заданного m-элементного множества $X \subset \mathbb{R}^n$ и вектора соответствующих значений $y \in \mathbb{R}^m$, требуется найти такую функцию $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, которая хорошо приближает исходные данные: $f(X_i) \approx y_i$, $i = \overline{1 \dots n}$, а также хорошо обобщается на новые данные.

В первую очередь, обратите внимание на **некорректность** данной задачи: понятие "хорошо" в предложенной формулировке не определяется (как и загадочная фраза про обобщение), кроме того, не уточнено, что имеется в виду под приближением. Однако с точки зрения здравого смысла более-менее понятно, в чём состоит задача - мы это как раз обсуждали в предыдущей главе: по данным X и y, найти разумную функцию f, которая бы их связывала.
Я умышленно начал с такой "полуматематической" формулировки, потому что чаще всего именно с такими формулировками вы будете встречаться на практике. Превращение такой формулировки во вполне корректную задачу - это важный шаг решения задачи и целое искусство.

Итак, предположим, что каждый элемент данных состоит из n чисел. Так, в соответствии с примером первой главы, каждый элемент из X - это вектор из трёх значений: (широта, долгота, время). В терминологии машинного обучения каждую компоненту таких векторов принято называть **признаком** (или атрибутом). Каждому вектору из X соответствует некоторое число y (эти значения часто называют **метками**) - в нашем примере пусть это будет температура воздуха. Таким образом, факт наличия у нас данных X и y можно обозначить фразой:

Даны m размеченных данных с n признаками.

Удобно представлять себе данные X в виде матрицы (а y - в виде вектора-столбца):

$$X = \begin{pmatrix} X_1^1 & X_2^1 & \dots & X_n^1 \\ X_1^2 & X_2^2 & \dots & X_n^2 \\ \dots & \dots & \dots & \dots \\ X_1^m & X_2^m & \dots & X_n^m \end{pmatrix} = \begin{pmatrix} X^1 \\ X^2 \\ \dots \\ X^m \end{pmatrix} \sim \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{pmatrix} = y$$

Теперь, когда мы поняли, что из себя представляют данные, давайте подумаем, как придать смысл фразе "хорошо приблизить".
Результатом нашей работы должна быть функция f (определённая не только на конечном множестве векторов X!). Хорошее приближение - это (пока на интуитивном уровне) когда значения, которые выдаёт функция на исходных данных, достаточно "близки" к правильным меткам y. Конечно, некоторые значения могут оказаться "ближе" других, но нам бы хотелось иметь одно-единственное число, которое бы характеризовало "близость" в целом. Как этого добиться?
Вспомним, что нам нужно оценить близость между обычными числами (нашими прогнозами $f(X^i)$ и правильными ответами y_i). В качестве меры близости между двумя числами можно взять модуль их разницы:

$$|y_i - f(X^i)|.$$

Чем меньше это число, тем точнее наша функция f в точке X^i предсказывает истинное значение y_i.
Возьмем в качестве меры близости (обозначим её ρ_1) *для всех* данных простую сумму таких выражений:

$$\rho_1(y, f(X)) := |y_1 - f(X^1)| + |y_2 - f(X^2)| + \ldots + |y_m - f(X^m)| \tag{5}$$

Если такая сумма мала, то обязательно будут малы все слагаемые! То что нам нужно, за исключением одного момента: чем больше у нас будет данных, тем больше будет отклонение (мы всегда прибавляем неотрицательные числа). Поэтому вполне естественно усреднить такое выражение по всем данным:

$$MAE = \frac{1}{m} \cdot \rho_1(y, f(X)) \tag{6}$$

Здесь аббревиатура MAE расшифровывается как "Mean Absolute Error" (средняя абсолютная ошибка).

Вероятно, читатель распознал в значке ρ стандартное обозначение для *метрики* (аналога функции расстояния). Если немного изменить угол зрения и рассмотреть все прогнозы в виде одного m-мерного вектора $f(X) = (f(X^1), f(X^2), \ldots, f(X^m))$, то, в качестве меры близости разумно взять одну из возможных метрик в $\mathbb{R}^m$, например, выражение из уравнения (5) - неплохой вариант!
На практике однако чаще используют другую метрику - *евклидову*, которая является естественным обобщением теоремы Пифагора для многомерного пространства:

$$\rho_2(y, f(X)) := \sqrt{(y_1 - f(X^1))^2 + (y_2 - f(X^2))^2 + \ldots + (y_m - f(X^m))^2} \tag{7}$$

Аналогично определяется средняя квадратичная ошибка (Mean Squared Error), обратите внимание на то, что метрика в квадрате - это позволяет избавиться от корня:

$$MSE = \frac{1}{m} \cdot \rho_2^2(y, f(X)) \tag{8}$$

Еще один тривиальный, но важный момент, стоит его один раз проговорить: если функция f минимизирует ρ_i, то она же минимизирует и ρ_i^2, а также MSE (MAE), $i = 1, 2$ (сами значения минимумов могут быть различны, однако точки, в которых они достигаются, одни и те же, это следствие монотонности участвующих операций):

$$\underset{a}{argmin}(\rho_2(a, b)) = \underset{a}{argmin}(\rho_2^2(a, b)) = \underset{a}{argmin}(MSE(a, b)) \tag{9}$$

Функция ρ_2 (а, вернее, её квадрат ρ_2^2) является более предпочтительной по нескольким причинам. Во-первых, стоит отметить её *гладкость* (т.е. наличие непрерывной производной) - в дальнейшем нам это свойство пригодится. Во-вторых, такой вид меры близости является нетривиальным следствием предположения о том, что погрешности в данных подчиняются нормальному закону (а это зачастую вполне разумное предположение). Этот результат мы в нашем вводном курсе доказывать не будем, но о нём полезно знать.

Нет такого постулата, из-за которого мы всегда обязаны выбирать ρ_2, а не ρ_1 (или ещё что-то более экзотичное)! И на самом деле бывают такие ситуации, о которых мы ещё поговорим, в которых ρ_1 оказывается более разумным выбором. Это как раз один из тех случаев, когда мы сами выбираем, как лучше формализовать задачу ("ууу... недостаточно строгости!" - покачают головой чистые математики).
Ещё немного терминологии: та функция, с помощью, которой мы оцениваем близость найденного нами решения к исходным данным (в нашем случае это MSE), называется **функцией потерь** (или Loss function). Под потерями имеется в виду та ошибка, которую мы допускаем, используя модель f.
Итак, задача приобретает вполне корректные очертания:

Даны m размеченных данных с n признаками. Требуется найти отображение между данными и признаками, минимизирующее функцию потерь MSE.

Осталось разобраться с моментом про "хорошо обобщается на новые данные". Как в момент построения модели f мы можем знать про новые данные, которыми мы сейчас не располагаем?
Оказывается, эта проблем решается на удивление просто: *давайте искать f, которая свяжет только часть данных из X и y, а её "качество обобщения" мы измерим с помощью функции потерь на оставшихся данных.*

Говоря математическим языком:

$$\begin{aligned} X &= X_{train} \cup X_{test}, \\ y &= y_{train} \cup y_{test} \end{aligned} \tag{10}$$

На этапе обучения модели f используем только X_{train} и y_{train}, а оцениваем качество обобщения модели на "новых данных" X_{test}:

$$MSE(f(X_{test}), y_{test}) \tag{11}$$

Остаётся открытым вопрос, как именно разбивать исходные данные на тренировочную и тестовую части. Обычно, на практике, разбиение производится случайным образом, с выделением около 70% данных для тренировочного множества. Конечно, этот рецепт подходит далеко не для всех ситуаций, однако есть универсальный совет, о котором всегда следует помнить: "структура" данных во множествах X, X_{train}, X_{test} должна быть однородной. Так, например, если существенная область значений одного из входящих в модель признаков сосредоточена в X_{test}, то, во время обучения на X_{train}, закономерности в этой области не будут найдены, что приведёт к плохой обобщаемости модели.

Итак, до настоящего времени мы обсуждали довольно-таки общие вещи, но словосочетание "линейная регрессия" ещё ни разу не встретилось (а ведь именно в честь неё называется эта глава!). Давайте, наконец, определим саму модель, т.е. тот принцип, по которому мы будем искать функцию f. По определению, будем называть **линейной регрессией** следующую "наивную" модель:

$$y_i \approx f_\alpha(X_1^i, X_2^i, \cdots, X_n^i) = \alpha_1 X_1^i + \alpha_2 X_2^i + \cdots + \alpha_n X_n^i + \alpha_{n+1} \tag{12}$$

Причем коэффициенты $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_{n+1})$ выбираются так, чтобы среднеквадратическая ошибка (разумеется, на некотором фиксированном тестовом множестве) была минимальной:

$$MSE(f_\alpha(X), y) \longrightarrow min \tag{13}$$

Давайте сделаем один важный "трюк": дополним матрицу X новым столбцом, состоящим из единиц. Разумеется, хотя новых данных у нас не появилось, однако у нас появился новый "признак" $X_{n+1}^i = 1$ для всех значений i. Это как раз тот признак, на который будет умножаться свободный коэффициент α_{n+1}. Т.е. без ограничения общности (с точностью до переобозначения $n+1$ на n) можно считать, что модель линейной регрессии выглядит так:

$$f_\alpha(X^i) = \alpha \cdot X^i, \quad \forall i = \overline{1 \ldots n} \tag{14}$$

Напомню, что здесь используется стандартное обозначение для **скалярного произведения**:

$$a \cdot b := a_1 b_1 + a_2 b_2 + \ldots + a_n b_n \tag{15}$$

Я назвал эту модель "наивной" не просто так: приближая значения y с помощью **линейной комбинации** признаков X^i мы делаем очень сильное предположение. Конечно же, редко когда выходные данные линейно зависят от входных, мир вокруг нас очень нелинеен! И что же, столько стараний напрасно? Мы вводили множество определений, давали рекомендации, выписывали формулы... И только ради того, чтобы в конце пути заявить, что это не работает?!
На самом деле это вовсе не "конец пути". Оказывается, имея в арсенале одну лишь линейную регрессию мы сможем обрабатывать данные, имеющие сколь угодно сложную нелинейную структуру!