Доповідь

Кравець Ольга та Кравець Назар

Група ПМО-41

“Лінійна регресія”

“Машинне навчання, як політика - всі про неї говорять, одиниці розуміють, а займаються лише експерти. Спробуємо якось виправити цю ситуацію.”

Машинне навчання - це розділ штучного інтелекту.

Мета машинного навчання - передбачити результат за вхідними даними.

Чим різноманітніші вхідні дані, тим простіше машині знайти закономірності і тим точніший результат.

Карта світу машинного навчання

Класифікувати алгоритми можна десятком способів. Ми показуємо один з методів, який здається найзручнішим для поясювання. Треба розуміти, що не буває так, щоб задачу розв'язував лише один метод.

Ось що машини сьогодні вміють, а що не під силу навіть найбільш навченим.

Машина може	Машина не може
Передбачати	Створювати нове
Запам'ятовувати	Різко порозумнішати
Відтворювати	Вийти за рамки завдання
Вибирати найкраще	Вбити всіх людей

Розпочнемо з базового огляду. Сьогодні в машинному навчанні є лише чотири основні напрями.

Основні типи машинного навчання

Вступ. Класичне навчання

Перші алгоритми прийшли до нас ще в 1950-х роках з чистої статистики. Вони розв'язували формальні задачі - шукали закономірності в числах, оцінювали близькість точок в просторі і вираховували напрямки.

Сьогодні на класичних алгоритмах тримається добра половина інтернету. Коли ви зустрічаєте блок «Рекомендовані статті» на сайті, або банк блокує всі ваші гроші на картці після першої ж покупки кави за кордоном - це майже завжди справа рук одного з цих алгоритмів.

Зрозуміло, що великі корпорації люблять розв'язувати всі проблеми нейромережами. Це спричинено тим, що зайві 2% точності для них легко конвертуються в додаткові 2 мільярди прибутку. Решті ж варто включати голову. Коли задача розв'язується класичними методами, то дешевше реалізувати скільки-небудь корисну для бізнесу систему саме за їх допомогою, а вже потім думати про якесь покращення. А якщо ви не розв'язали задачу, то не розв'язати її на 2% краще вам не особливо допоможе.

При всій своїй популярності, класичні алгоритми настільки прості, що їх легко пояснити навіть дитині. Сьогодні вони як основи арифметики - застосовуються постійно, але дехто все одно став їх забувати.

Класичне навчання

Навчання з вчителем

Класичне навчання люблять ділити на дві категорії — з вчителем і без. Часто можна зустріти їх англійські назви — Supervised і Unsupervised Learning.

У першому випадку у машини є якийсь учитель, який говорить їй як правильно. Розповідає, що на цій картинці кішка, а на цій собака. Тобто вчитель вже заздалегідь розділив всі дані на кішок і собак, а машина навчається на конкретних прикладах.

У навчанні без учителя, машині просто вивалюють купу фотографій тварин на стіл і кажуть «розберися, хто тут на кого схожий». Дані не

розмічені, у машини немає вчителя, і вона намагається сама знайти будь-які закономірності. Про ці методи ми й поговоримо.

Очевидно, що з учителем машина навчиться швидше і точніше, тому в задачах його використовують набагато частіше. Ці задачі діляться на два типи: класифікація - передбачення категорії об'єкта, і регресія - передбачення місця на числовій прямій.

Класифікація

«Поділяє об'єкти за заздалегідь відомою ознакою. Шкарпетки за кольорами, документи за мовами, музику за жанрами»

Сьогодні використовують для:

Спам-фільтри

Визначення мови

Пошук схожих документів

Аналіз тональності

Розпізнавання рукописних букв і цифр

Визначення підозрілих транзакцій

Класифікація речей - найпопулярніша задача у всьому машинному навчанні. Машина в ній як дитина, яка навчається розкладати іграшки: роботів в один ящик, танки в інший. Опа, а якщо це робот-танк? Що ж, час розплакатися і випасти в помилку.

Для класифікації завжди потрібен учитель - розмічені дані з ознаками і категоріями, які машина буде вчитися визначати за цими ознаками. Далі класифікувати можна що завгодно: користувачів за інтересами - так роблять алгоритмічні стрічки, статті за мовами і тематиками - важливо для пошукових систем, музику за жанрами - згадайте плейлисти, навіть листи у вашій поштовій скриньці.

Раніше всі спам-фільтри працювали на алгоритмі Баєса. Машина вважала скільки разів слово «кредит» зустрічається в спамі, а скільки разів на

нормальних листах. Множила ці дві ймовірності за формулою Баєса і складала результати всіх слів.

Пізніше спамери навчилися обходити фільтр Байеса, просто вставляючи в кінець листа багато слів з «хорошими» рейтингами. Метод отримав іронічну назву Отруєння Баєса, а фільтрувати спам стали іншими алгоритмами. Але метод назавжди залишився в підручниках як найпростіший, красивий і один з перших практично корисних.

Регресія

«Намалюй лінію уздовж моїх точок. Саме так, це машинне навчання»

Регресія – це метод визначення зв'язку між однією змінною (y) та іншими змінними (x). У статистиці лінійна регресія — це підхід до моделювання лінійного відношення між y та x.

Сьогодні використовують для:

Прогноз вартості цінних паперів

Аналіз попиту, обсягу продажів

Медичні діагнози

Будь-які залежності числа від часу

Регресія - та ж класифікація, тільки замість категорії ми передбачаємо число. Вартість автомобіля з його пробігом, кількість корків щодо часу доби, обсяг попиту на товар від зростання компанії і.т.д. На регресію ідеально лягають будь-які задачі, де є залежність від часу.

Регресію дуже люблять фінансисти і аналітики, вона вбудована навіть в Excel. Всередині все працює, знову ж таки, банально: машина тупо намагається намалювати лінію, яка в середньому відображає залежність. Правда, на відміну від людини з фломастером і вайтбордом, робить вона це з точністю - обчислюючи середню відстань до кожної точки і намагаючись всім догодити.

Коли регресія малює пряму лінію, її називають лінійною, коли криву - поліноміальною. Це два основні типи регресії, далі вже починаються

рідкоземельні методи. Але є й таке як **Логістична Регресія**, яка насправді не регресія, а метод класифікації, від чого у всіх постійно плутанина. Не робіть так.

Схожість регресії і класифікації підтверджується ще й тим, що багато хто з класифікаторів, після невеликого тюнінгу, перетворюються в регресорів. Наприклад, ми можемо не просто дивитися до якого класу належить об'єкт, а запам'ятовувати, наскільки він близький - і ось, у нас регресія.

Приклад

Припустимо, ми хочемо купити діамант. У нас є кільце, яке належало моєї бабусі. Його оправа містить діамант масою 1,35 карата і я хочу знати скільки він буде коштувати. Я беру з собою блокнот і ручку і йду в ювелірний магазин. Там я записую всі ціни на діаманти, які є у вітрині, а також їх масу в каратах. Починаючи з першого діаманта, який має масу 1,01 карата і коштує 7366$.

Я йду далі і роблю те ж саме для інших діамантів в магазині.

Стовпці з даними діамантів:

Carats	price
1.01	7,366
.49	985
.31	544
1.51	9,140
.37	493
.73	3,011
1.53	11,413
.56	1,814
.41	876
74	

Зверніть увагу, що наш список містить два стовпці. Кожен стовпець має свій атрибут - масу в каратах і ціну - і кожен рядок є окремою точкою даних, що представляє один діамант.

Фактично ми створили невеликий набір даних у формі таблиці. Зверніть увагу, що він відповідає нашим критеріям якості.

Дані є релевантними (міра відповідності отримуваного результату бажаному): маса безумовно пов'язана з ціною.

Вони точні: ми перевірили ціни, які записали.

Вони пов'язані: всі стовпці і рядки заповнені.

І, як ми побачимо далі, цих даних достатньо, щоб дати відповідь на наше питання.

Постановка точного питання

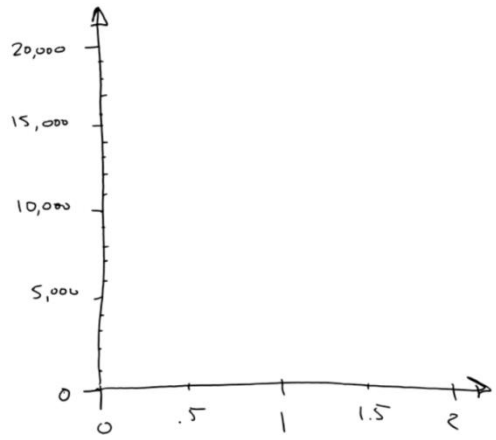
Тепер давайте точно сформулюємо наше запитання: "Скільки буде коштувати діамант масою 1,35 карата?"

У нашому списку немає діаманта масою 1,35 карата, тому необхідно використовувати наявні дані для отримання відповіді на це питання.

Побудова існуючих даних

Перше, що необхідно зробити, це намалювати горизонтальну числову лінію, яку називають віссю. На ній буде відображатися маса. Діапазон мас - від 0 до 2, тому ми намалюємо лінію, що охоплює цей діапазон, і помістимо на неї позначки для кожних 0,5 карата.

Потім ми намалюємо вертикальну вісь, на якій запишемо ціни, і з'єднаємо її з горизонтальною віссю маси. Одиницею виміру для цієї осі буде долар США. Тепер у нас є набір осей координат.

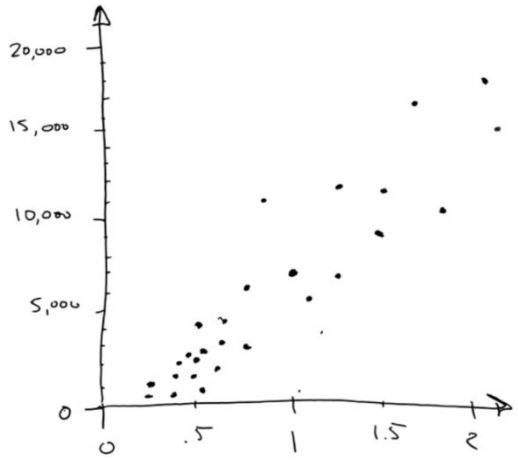


Осі маси і ціни

Тепер ми перетворимо ці дані в точкову діаграму. Це відмінний спосіб візуалізації числових наборів даних.

Беремо першу точку даних і проводимо вертикальну лінію від позначки 1,01 карата. Потім проводимо горизонтальну лінію від позначки 7366$. Там, де ці лінії перетинаються, малюємо точку. Ця точка і представляє наш перший діамант.

Тепер ми пройдемо по всім діамантів з нашого списку і зробимо те ж саме. В результаті ми отримаємо таку картину: безліч точок, кожна з яких відповідає одному діаманту.

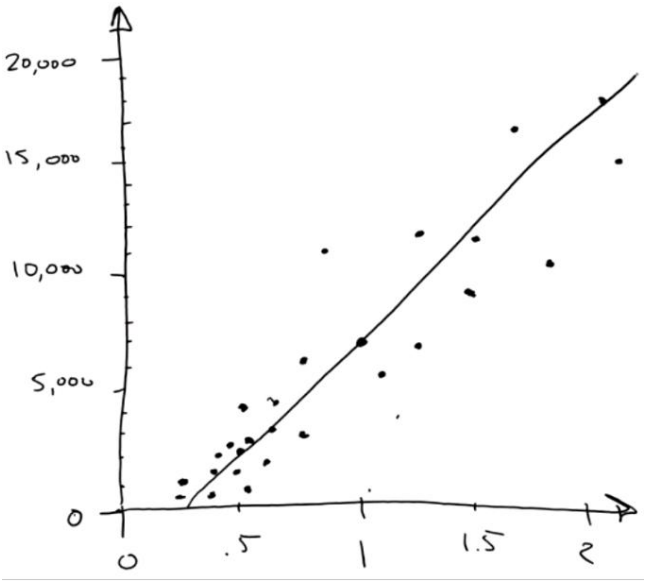


Точкова діаграма

Побудова моделі на основі точок даних

Тепер, якщо поглянути на точки під невеликим кутом, то весь цей набір виглядає як товста і нечітка лінія. Можна скористатися маркером і провести через набір точок пряму лінію.

Намалювавши лінію, ми створили модель. Щоб краще зрозуміти, уявіть собі, що реальний світ зображений у спрощеній формі, як комікс. Комікс не відображає всієї дійсності: лінія не проходить через всі крапки даних. Але це корисне спрощення.



Лінія лінійної регресії

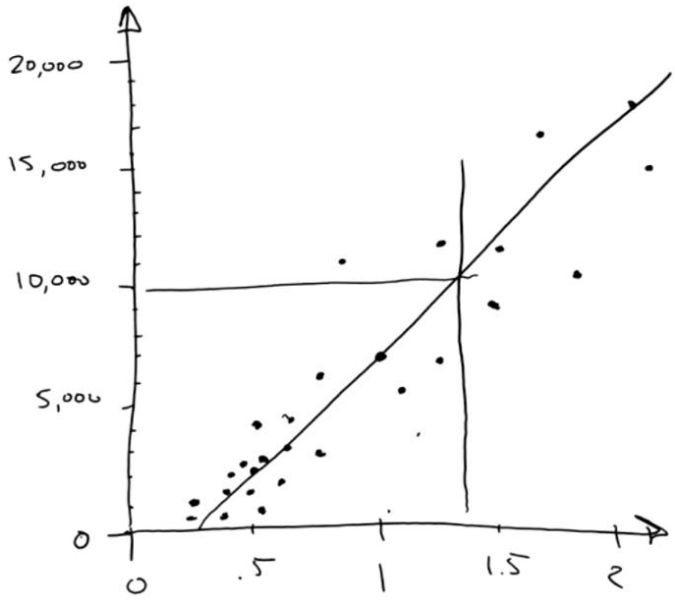
Той факт, що лінія не проходить через всі крапки, є нормальним. Фахівці з обробки даних пояснюють це так: існує модель (в нашому випадку - лінія), і кожна точка в моделі піддається впливу деякого шуму або відхилення, пов'язаного з нею. Є базова функціональна зв'язок, а є мінливий реальний світ, який додає шум і невизначеність.

Так як ми намагаємося відповісти на питання “Скільки?” - це називається регресією. І оскільки ми використовуємо пряму лінію, це - лінійна регресія.

Використання моделі для пошуку відповіді

Тепер у нас є модель і ми можемо поставити їй наше запитання: "Скільки буде коштувати діамант масою 1,35 карата?"

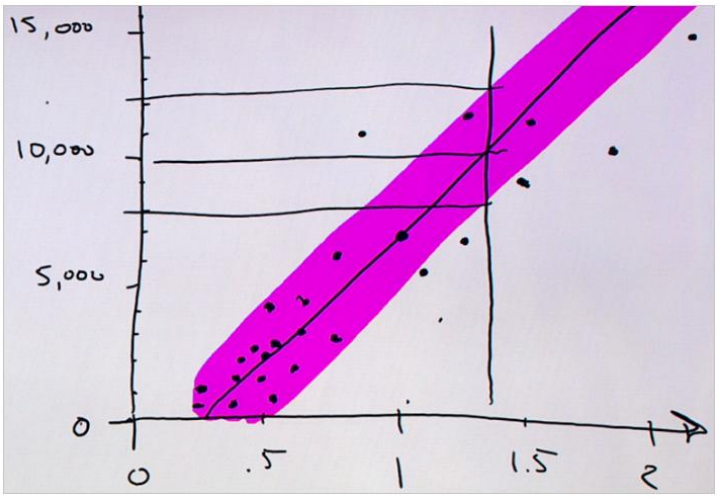
Для відповіді на питання ми проводимо вертикальну лінію від позначки 1,35 карата. Від точки її перетину з лінією моделі проводимо горизонтальну лінію до осі цін. Вона приводить нас до позначки 10 000. Ось так! Це і є відповідь: діамант масою 1,35 карата коштує близько 10 000 доларів США.



Пошук відповіді за допомогою моделі

Створення довірчого інтервалу

Було б логічним перевірити, наскільки точний цей прогноз. Корисно знати, чи буде діамант масою 1,35 карата коштувати рівно 10 000$, а може бути набагато більше або менше. Щоб це визначити, давайте намалюємо навколо лінії регресії так званий конверт, який буде включати в себе більшість точок. Цей конверт - наш довірчий інтервал. Ми можемо бути впевнені, що ціни потраплять в зазначений діапазон, адже це справедливо для більшості цін в минулому. Можна провести ще дві горизонтальні лінії від точок перетину лінії, проведеної від позначки 1,35 карата, з верхньою і нижньою межами даного конверта.



Довірчі інтервали

Тепер ми можемо щось сказати про наш довірчий інтервал. Можна з упевненістю говорити, що ціна діаманта 1,35 карата складе близько 10 000 дол. США, при цьому вона може бути не нижче 8 000 і не вище 12 000 долл. США.

Тобто:

Ми поставили запитання, на яке можна відповісти за допомогою даних.

Ми створили модель, використовуючи лінійну регресію.

Ми зробили прогноз, доповнений довірчим інтервалом.

При цьому ми не користувалися математичними формулами або комп'ютерами.

Якби у нас були додаткові відомості, такі як:

огранювання діаманта;

відтінки кольорів (наскільки близький колір діаманта до білого);

наявність в камені сторонніх включень;

то ми б мали додаткові стовпці. В цьому випадку математика вже буде корисною. Коли є більше двох стовпців, важко намалювати точки на папері. Математика дозволить відмінно вписати отриману лінію або площину в ваші дані.

Крім того, якби замість невеликого переліку діамантів у нас було б дві тисячі або два мільйони каменів, то набагато швидше цю роботу можна було б зробити за допомогою комп'ютера.

Сьогодні ми поговорили про те, як створити лінійну регресію, а також ми зробили прогноз, використовуючи дані.

Як будується модель

Лінійна регресійна модель має наступний вигляд:

$$y = \beta_0 + \beta_1 x_1 + ... + \beta_k x_k + u \qquad (2.1)$$

де y– залежна змінна;

$x_1, x_2, ..., x_n$– незалежні змінні;

u– випадкова похибка, розподіл якої в загальному випадку залежить від незалежних змінних, але математичне очікування якої рівне нулю.

Згідно з моделлю (2.1), математичне очікування залежної змінної є лінійною функцією незалежних змінних:

$$E(y) = \beta_0 + \beta_1 x_1 + ... + \beta_k x_k + u \qquad (2.2)$$

Вектор параметрів $\beta_0, \beta_1, ..., \beta_k$ є невідомим і задача лінійної регресії полягає в оцінці цих параметрів на основі деяких експериментальних значень y і $x_1, x_2, ..., x_n$. Тобто, для деяких n експериментів є відомі значення $\{y_i, y_{i1}, ..., y_{ip}\}_{i=1}^{n}$ незалежних змінних і відповідне їм значення залежної змінної.

Згідно з визначенням моделі для кожного експериментального випадку залежність між змінними визначається формулою:

$$y_i = \beta_0 + \beta_1 x_1 + ... + \beta_k x_k + u \qquad (2.3)$$

На основі цих даних потрібно оцінити значення параметрів $(\beta_0, \beta_1, ..., \beta_k)$, а також розподіл випадкової величини u. Зважаючи на характеристики досліджуваних змінних, можуть додаватися різні додаткові специфікації моделі і застосовуватися різні методи оцінки параметрів. Серед найпоширеніших специфікацій лінійних моделей є класична модель лінійної регресії та узагальнена модель лінійної регресії.

Згідно з класичною моделлю лінійної регресії додатково вводяться такі вимоги щодо специфікації моделі та відомих експериментальних даних:

$\forall i \neq jE(u_i u_j | x_i) = 0$ (відсутність кореляції залишків);

$\forall iE(u_j^2 | x_i) = \sigma^2$ (гомоскедастичність - незалежність дисперсії випадкових складових від номера спостереження).

Часто додається також умова нормальності випадкових відхилень, яка дозволяє провести значно ширший аналіз оцінок параметрів та їх значимості, хоча і не є обов'язковою для можливості використання наприклад методу найменших квадратів $(u_j | x_i) \sim N(0, \sigma^2)$.

Передбачимо, що незалежна змінна набула значень $x_1, x_2 ..., x_n$, внаслідок чого залежна змінна набула значень $y_1, y_2 ..., y_n$. У припущенні лінійної залежності отримуємо n рівностей.

$y_i = a_0 + a_1 x_i + \varepsilon_i, i = 1...n,$ (2.4)

де ε_i– незалежні і розподілені так само, як ε.

Потрібно за значеннями пар (x_i, y_i) оцінити невідомі a_0, a_1. Як ми вже знаємо, кожне завдання оцінювання пов'язане з деяким критерієм якості. У теорії, що викладається нами, таким критерієм є критерій найменших квадратів:

$$\sum_{i=1}^{n} \varepsilon_i^2 - \min. \quad (2.5)$$

Запишемо цю суму інакше, так, щоб була помітна залежність від a_0, a_1:

$$\sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} [\overline{y}(x) - y_i]^2 = \sum_{i=1}^{n} [y_i - a_0 - a_1 x_1]^2 \qquad (2.6)$$

Тепер остаточно приходимо до такої задачі: знайти такі значення невідомих параметрів a_0, a_1щоб функція 2.11 набула найменшого значення:

$$Q(a_0 a_1) = \sum_{i=1}^{n} [y_i - a_0 - a_1 x_1]^2 \qquad (2.7)$$

Метод розв'язання цієї задачі відомий з курсу математичного аналізу чи вищої математики.

Знаходимо часткові похідні функції Q і прирівнюємо їх до нуля, внаслідок чого приходимо до системи лінійних рівнянь:

$$\begin{cases} \frac{\partial Q}{\partial a_0} = -2 \sum_{i=1}^{n} [y_i - a_0 - a_1 x_i] = 0, \\ \frac{\partial Q}{\partial a_1} = -2 \sum_{i=1}^{n} [y_i - a_0 - a_1 x_i]\, x_i = 0. \end{cases} \qquad (2.8)$$

Після очевидних перетворень отримуємо систему:

$$\begin{cases} n a_0 + a_1 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_{i,} \\ a_0 \sum_{i=1}^{n} x_i + a_i \sum_{i=1}^{n} x_i{}^2 = \sum_{i=1}^{n} x_i y_i \end{cases} \qquad (2.9)$$

(Усі перетворення є в книжці)

Позначимо вибіркові середні:

$$\overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \,, \overline{y} = \frac{1}{n} \sum_{i=1}^{n} y_i \,, \overline{xy} = \frac{1}{n} \sum_{i=1}^{n} x_i \, y_{i,} \overline{x}^2 = \frac{1}{n} \sum_{i=1}^{n} x_i{}^2 \,. \qquad (2.10)$$

У цих позначеннях після ділення кожного рівняння системи на n вона набуде вигляду:

$$\begin{cases} a_0 + a_1\overline{x} = \overline{y}, \\ a_x\overline{x} + a_1\overline{x}^2 = \overline{xv} \end{cases} \qquad (2.11)$$

а її рішення (шукані оцінки коефіцієнтів рівняння регресії) буде таким:

$$\begin{array}{l} \hat{a}_0 = \frac{\overline{x^2}\cdot\overline{y} - \overline{xy}\cdot\overline{x}}{\overline{x^2} - (\overline{x})^2} \\ \hat{a}_1 = \frac{\overline{xy} - \overline{y}\cdot\overline{x}}{\overline{x^2} - (\overline{x})^2} \end{array} \qquad (2.12)$$

Якщо ввести ще позначення $S_x^2 = \overline{x^2} - \overline{(x)}^2$ і перетворити вираз $\hat{a}_0$:

$$\hat{a}_0 = \frac{\overline{x^2}\cdot\overline{y} - \overline{xy}\cdot\overline{x} \pm \overline{y}\cdot(\overline{x})^2}{S_{..}^2} = \frac{S_x^2\cdot\overline{y} - \overline{x}(\overline{xy} - \overline{y}\cdot\overline{x})}{S_{..}^2} = \overline{y} - \hat{a}_1\cdot\overline{x}, \qquad (2.14)$$

то оцінка функції регресії набуде такого вигляду:

$$\tilde{\tilde{y}}(x) = \hat{a}_0 + \hat{a}_1 x = \overline{y} - \hat{a}_1\cdot\overline{x} + \hat{a}\cdot x = \overline{y} + \hat{a}_1(x - \overline{x}) \qquad (2.15)$$

Таким чином, отримали рівняння регресії, що є моделлю для опису даних.

Оцінки якості регресії

Найбільш типовими заходами якості в задачах регресії є

Середня квадратична помилка (англ. Mean Squared Error, MSE)

MSE застосовується в ситуаціях, коли нам треба підкреслити великі помилки і вибрати модель, яка дає менше помилок прогнозу. Грубі помилки стають помітнішими за рахунок того, що помилку прогнозу ми зводимо в квадрат. І модель, яка дає нам менше значення середньоквадратичної помилки, можна сказати, що що у цій моделі менше грубих помилок.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(a(x_i) - y_i)^2$$

Середня абсолютна помилка (англ. Mean Absolute Error, MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|a(x_i) - y_i|$$

Середньоквадратичний функціонал сильніше штрафує за великі відхилення в порівнянні з середньоабсолютним, і тому більш чутливий до викидів. При використанні будь-якого з цих двох функціоналів може бути корисно проаналізувати, які об'єкти вносять найбільший внесок у загальну помилку - не виключено, що на цих об'єктах була допущена помилка при обчисленні ознак або цільової величини.

Ефективне значення помилки підходить для порівняння двох моделей або для контролю якості під час навчання, але не дозволяє зробити висновків про те, на скільки добре дана модель вирішує завдання. Наприклад, MSE = 10 є дуже поганим показником, якщо цільова змінна приймає значення від 0 до 1, і дуже хорошим, якщо цільова змінна лежить в інтервалі (10000, 100000). У таких ситуаціях замість середньоквадратичної помилки корисно використовувати коефіцієнт детермінації - R^2

Коефіцієнт детермінації

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(a(x_i) - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Коефіцієнт детермінації вимірює частку дисперсії, внесеною моделлю, в загальній дисперсії цільової змінної. Фактично, дана міра якості - це нормована середньоквадратична помилка. Якщо вона близька до одиниці, то модель добре пояснює дані, якщо ж вона близька до нуля, то прогнози можна порівняти за якістю з сталою пророкуванням.

Середня абсолютна процентна помилка (англ. Mean Absolute Percentage Error, MAPE)

$$MAPE = 100\% \times \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - a(x_i)|}{|y_i|}$$

Це коефіцієнт, який не має розмірності, з дуже простою інтерпретацією. Його можна вимірювати в частках або відсотках. Якщо у вас вийшло, наприклад, що MAPE = 11.4%, то це говорить про те, що помилка склала 11,4% від фактичних значень. Основна проблема цієї помилки - нестабільність.

Корінь із середньої квадратичної помилки (англ. Root Mean Squared Error, RMSE)

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \overline{y_i})^2}$$

Приблизно така ж проблема, як і в MAPE: так як кожне відхилення зводиться в квадрат, будь невелике відхилення може значно вплинути на показник помилки. Варто зазначити, що існує також помилка MSE, з якої RMSE якраз і виходить шляхом вилучення кореня.

Симетрична MAPE (англ. Symmetric MAPE, SMAPE)

$$SMAPE = \frac{1}{n} \sum_{i=1}^{n} \frac{2 \times |y_i - a(x_i)|}{|y_i| + |a(x_i)|}$$

Середня абсолютна масштабована помилка (англ. Mean absolute scaled error, MASE)

$$MASE = \frac{\sum_{i=1}^{n} |Y_i - e_i|}{\frac{n}{n-1} \sum_{i=2}^{n} |Y_i - Y_{i-1}|}$$

MASE є дуже хорошим варіантом для розрахунку точності, так як сама помилка не залежить від масштабів даних і є симетричною: тобто позитивні і негативні відхилення від факту розглядаються в рівній мірі. *Зверніть увагу, що в MASE ми маємо справу з двома сумами: та, що в чисельнику, відповідає тестової вибірці, та, що в знаменнику - навчальної*. Друга фактично являє собою середню абсолютну помилку прогнозу. Вона ж відповідає середньому абсолютному відхиленню ряду в перших різницях. Ця величина, по суті, показує, наскільки навчальна вибірка передбачувана. Вона може бути дорівнює нулю тільки в тому випадку, коли всі значення в навчальній вибірці рівні один одному, що відповідає відсутності будь-яких змін в ряді даних, ситуації на практиці майже неможливою. Крім того, якщо ряд має тенденцію до зростання або зниження, його перші різниці будуть коливатися близько деякого фіксованого рівня. В результаті цього з різних рядах з різною структурою, знаменники будуть більш-менш порівнянними. Все це, звичайно ж, є очевидними плюсами MASE, так як дозволяє складати різні значення за різними рядах і отримувати незміщені оцінки.

Недолік MASE в тому, що її важко інтерпретувати. Наприклад, MASE = 1.21 ні про що, по суті, не говорить. Це просто означає, що помилка прогнозу виявилася в 1.21 рази вище середнього абсолютного відхилення ряду в перших різницях, і нічого більше.

Крос-валідація

Хороший спосіб оцінки моделі передбачає застосування крос-валідації (ковзного контролю або перехресної перевірки).

В цьому випадку фіксується деяка множина розбиття вихідної вибірки на дві підвибірки: навчальну і контрольну. Для кожного розбиття виконується настройка алгоритму за навчальною підвибіркою, потім оцінюється його середня помилка на об'єктах контрольної підвибірки. Оцінкою змінного контролю називається середня по всім розбиттям величина помилки на контрольних підвибірках.

Відео

Надіємося, тепер ви маєте уявлення про один із найважливіших підходів до прогнозування в машинному навчанні та статистиці. Ми намагалися розповіли про головні принципи роботи лінійної регресії, підготовку даних для роботи моделі та деякі метрики для оцінювання ефективності роботи.

Лінійні моделі швидко навчаються і стійкі до шуму в даних, але в задачах з нелінійною залежністю вони слабкі та мало ефективні. Вони чутливі до викидів і за їх наявності можуть дуже помилятися в прогнозах.

```python
# Імпорт необхідних бібліотек
from time import time  # Імпорт функції для обробки часу (не використовується в цьому прикладі)
from sklearn.datasets import make_blobs  # Функція для генерації синтетичних даних
from sklearn.model_selection import train_test_split  # Функція для розділення даних на навчальний та тестовий набори
from sklearn.linear_model import LogisticRegression  # Клас для логістичної регресії
import matplotlib.pyplot as plt  # Бібліотека для візуалізації даних
import numpy as np  # Бібліотека для обробки числових операцій

# Встановлення випадкового насіння для відтворюваності результатів
np.random.seed(222)

# Генерація випадкових даних для X та Y
X = 2 * np.random.rand(100, 1)
Y = 9 * X + 2 + np.random.randn(100, 1)

# Функція для обчислення вартості (середньоквадратичної помилки) моделі
def compute_cost(X, Y, theta):
    m = len(Y)  # Кількість прикладів в навчальному наборі
    predictions = X.dot(theta)  # Обчислення прогнозних значень
    cost = (1 / (2 * m)) * np.sum(np.square(predictions - Y))  # Обчислення середньоквадратичної помилки
    return cost
```

```python
# Функція градієнтного спуску для навчання моделі
def gradient_descent(X, Y, theta, learning_rate, iterations):
    m = len(Y)  # Кількість прикладів в навчальному наборі
    cost_history = np.zeros(iterations)  # Масив для зберігання історії вартості
    for i in range(iterations):
        predictions = X.dot(theta)  # Обчислення прогнозних значень
        errors = np.subtract(predictions, Y)  # Обчислення відхилень прогнозів від реальних значень
        sum_delta = (learning_rate / m) * X.transpose().dot(errors)  # Обчислення змін для кожного параметра моделі
        theta -= sum_delta  # Оновлення параметрів моделі
        cost_history[i] = compute_cost(X, Y, theta)  # Зберігання вартості на поточній ітерації
    return theta, cost_history

# Додавання стовпця одиниць до матриці ознак для зручності обчислень
X_b = np.c_[np.ones((len(X), 1)), X]

# Ініціалізація випадкових параметрів моделі
theta = np.random.randn(2, 1)

# Визначення швидкості навчання та кількості ітерацій для градієнтного спуску
learning_rate = 0.01
iterations = 1000
```

```
# Виконання градієнтного спуску для навчання моделі та отримання оптимальних параметрів
theta, cost_history = gradient_descent(X_b, Y, theta, learning_rate, iterations)

# Візуалізація даних та результатів лінійної регресії
plt.scatter(X, Y)  # Додавання точок на графіку для відображення вихідних даних
plt.plot(X, X_b.dot(theta), color='red')  # Додавання лінії прогнозу на графіку
plt.xlabel('X')  # Підпис осі X
plt.ylabel('Y')  # Підпис осі Y
plt.title('Linear Regression with Gradient Descent')  # Заголовок графіку
plt.show()  # Відображення графіку
```