Issue 15, 2024

https://doi.org/10.23939/sisn2024.15.322

УДК 004.852

МАТЕМАТИЧНА МОДЕЛЬ ЛОГІСТИЧНОЇ РЕГРЕСІЇ ДЛЯ БІНАРНОЇ КЛАСИФІКАЦІЇ. Ч. 2. ПРОЦЕСИ ПІДГОТОВКИ, НАВЧАННЯ І ТЕСТУВАННЯ ДАНИХ

Петро Кравець[1], Володимир Пасічник[2], Микола Проданюк[3]

Національний університет "Львівська політехніка",
кафедра інформаційних систем та мереж, Львів, Україна,
[1] E-mail: Petro.O.Kravets@lpnu.ua, ORCID: 0000-0001-8569-423X;
[2] E-mail: Volodymyr.V.Pasichnyk@lpnu.ua, ORCID: 0000-0002-5231-6395;
[3] E-mail: Mykola.M.Prodaniuk@lpnu.ua, ORCID: 0000-0001-9544-3792

У цій статті розглянуто теоретичні аспекти логістичної регресії для бінарної класифікації даних, включаючи процеси підготовки даних, навчання, тестування та показники оцінювання моделей.

Сформульовано вимоги до вхідних наборів даних, описано способи кодування категоріальних даних, визначено та обґрунтовано способи масштабування вхідних ознак.

Розроблено схему навчання логістичної регресії методом градієнтного спуску для мінімізації функції втрат відповідним налаштуванням ваг ознак призначеної для класифікації вибірки об'єктів. Визначено особливості побудови рекурентних методів класичного та стохастичного градієнтного спуску. Описано вимоги до організації вибірки даних для моделі багатоетапного навчання з метою уникнення перенавчання або недонавчання логістичної регресії.

Наведено схему тестування навченої логістичної регресії та описано основні метрики якості бінарної класифікації. Відмічено вплив висоти порогу класифікації на ефективність логістичної регресії.

За результатами роботи намічено напрями перспективних досліджень логістичної регресії.

Ключові слова: математична модель, логістична регресія, бінарна класифікація, аналіз даних, машинне навчання, сигмоїдна функція, логарифмічна функція втрат, градієнтний спуск, поріг класифікації, метрики якості класифікації.

Постановка проблеми

Логістична регресія є одним з найпоширеніших методів у машинному навчанні для бінарної класифікації даних у багатьох галузях, включаючи медицину, фінанси та інформаційні технології [1–5]. Її популярність обумовлена простотою реалізації, інтерпретованістю результатів, здатністю до ефективної обробки навіть складних наборів даних. Проте, для досягнення найкращих результатів у роботі з логістичною регресією необхідно добре розуміти її методику та особливості.

У цій роботі досліджуються процеси навчання і тестування логістичної регресії для бінарної класифікації даних з метою глибшого розуміння принципів, які лежать в основі цього методу машинного навчання. Незважаючи на те, що логістична регресія є відносно простою моделлю, вона є потужним інструментом у сфері машинного навчання і її правильне використання може забезпечити необхідну точність та ефективність процесу аналізу даних.

Ця публікація є другою частиною статті про застосування логістичної регресії для бінарної класифікації даних. У першій частині виконано огляд літературних джерел, розглянуто математичні основи логістичної регресії, поняття логістичної функції та логарифмічної функції втрат.

У цій статті запропоновано детальне вивчення процесу навчання та тестування логістичної регресії для бінарної класифікації даних. Розглянуто не лише технічні аспекти цього методу, але й акцентовано увагу на розумінні фундаментальних принципів, що лежать в його основі.

Для ефективного застосування логістичної регресії необхідно провести належну підготовку вхідних даних, яка включає у себе кодування категоріальних та масштабування числових ознак призначених для класифікації об’єктів.

У роботі розглянуто важливі математичні аспекти навчання за допомогою методу градієнтного спуску, що є ключовим етапом у побудові моделі. Важливу роль в цьому процесі відіграє правильне формулювання функції втрат та вибір оптимальних параметрів моделі.

Розроблена схема тестування дозволить перевірити здатність моделі до узагальнення даних. Огляд різних метрик якості навчання, таких як точність, чутливість, специфічність, F-міра та AUC ROC, допоможуть об’єктивно оцінити та зрозуміти ефективність роботи моделі.

Не менш важливим аспектом є аналіз впливу висоти порогу на ефективність класифікації. Правильний вибір порогу класифікації дозволить оптимізувати роботу моделі. Встановлення оптимального порогу є ключовим етапом у роботі з логістичною регресією. Розуміння цього ефекту допоможе вдосконалити процес класифікації та забезпечити оптимальну продуктивність моделі.

Загалом стаття покликана розкрити основи та надати глибоке і цілісне розуміння принципів логістичної регресії як потужного інструменту аналізу даних.

Аналіз останніх досліджень та публікацій

Сучасні публікації з логістичної регресії роблять значний внесок в науку та практику, допомагаючи розвивати нові підходи до аналізу даних і прийняття рішень.

У частині 1. РЕГРЕСІЙНІ МОДЕЛІ УЗАГАЛЬНЕННЯ ДАНИХ нашої попередньої статті ми вже провели детальний аналіз останніх досліджень і публікацій, виконали огляд літературних джерел, розглянули математичні основи логістичної регресії, поняття логістичної функції та логарифмічної функції втрат.

У роботах зарубіжних авторів [4–6] вказується на те що метод логістичної регресії є зручним інструментом аналізу даних та прийняття рішень, який застосовується у багатьох галузях для вирішення різноманітних завдань, що потребують моделювання імовірностей подій, аналізу, прогнозування або класифікації.

Однак, за винятком деяких монографічних досліджень [3, 4, 6, 7], висвітлення математичних основ методу логістичної регресії у загальному масиві публікацій зі статистичного аналізу та машинного навчання є розрізненим, фрагментарним та проблемно-орієнтованим. Обмеження на обсяг статті у періодичних друкованих виданнях не дозволяє авторам цілісно висвітлити проблему класифікації методом логістичної регресії.

Електронні публікації [10, 16,18] теж фрагментарно описують проблему логістичної регресії. В публікаціях [19–23], більша увага звернена на практичну реалізацію конкретних методів та алгоритмів для вирішення окремих задач для заданої предметній області, що дещо відволікає від загального розуміння проблеми.

У цій статті ми систематизуємо теоретичні аспекти логістичної регресії для бінарної класифікації даних, включаючи процеси підготовки даних, навчання, тестування та показники оцінювання моделей.

Формулювання цілі статті

Метою цієї роботи є математичне обґрунтування моделі навчання та тестування даних методом логістичної регресії.

Для досягнення мети необхідно:

- Проаналізувати метод градієнтного спуску для навчання логістичної регресії.
- Розробити схеми навчання та тестування вибірки даних методом логістичної регресії.
- Описати метрики якості бінарної класифікації методом логістичної регресії.
- Окреслити нові, перспективні напрямки дослідження та застосування логістичної регресії.

Виклад основного матеріалу

Підготовка вхідних даних

Призначені для класифікації вхідні дані задаються таблицею $(X_i, y_i)_{i=1}^n$, рядки (X_i, y_i) якої є результатом окремого спостереження, n – обсяг вибірки, $X_i \in R^{m+1}$ – набір ознак об'єкта, m – кількість ознак об'єкта, $y_i \in \{0,1\}$ – актуальна мітка класу цього об'єкта.

Для ефективної та точної роботи регресійної моделі потрібна певна підготовка вхідних даних для вибору найбільш інформативних ознак об'єктів.

По-перше, щоб модель була адекватною і точною, досліджувана вибірка повинна бути репрезентативною, тобто повинна містити дані про характеристики об'єктів, що відображають їхні різноманітні, спільні та відмінні ознаки.

По-друге, потрібно перетворити категоріальні дані у числові скалярні чи векторні значення.

По-третє, необхідно вибрати найбільш важливі ознаки, які мають найвищу кореляцію з цільовою змінною. При цьому необхідно уникати взаємної кореляції або подібності ознак між собою, оскільки це може призвести до нестабільних оцінок параметрів та зменшення точності моделі.

По-четверте, у вибірці бажано прибрати шум у вигляді даних з несуттєвими ознаками та аномальними значеннями неважливих для моделі ознак.

По-п'яте, потрібно провести нормування ознак об'єктів, приведення їх числових значень до одного масштабу, щоб запобігти їх непропорційному впливу на результати навчання та неправильній інтерпретації вагових коефіцієнтів.

Кодування категоріальних даних

Категоріальні дані – це нечислові дані, які можна розділити на групи або категорії на основі їхніх властивостей, характеристик або значень. Такими категоріями можуть бути кольори, марки автомобілів, види продуктів, статуси (наприклад, –студент", "викладач", "співробітник") та інші категорії, які не мають числового зображення. В аналізі даних такі категорії часто використовуються для класифікації або групування даних з метою отримання висновків або вироблення прогнозів.

Для категоріальних даних характерні операції над множинами і не застосовуються числові операції, усереднення значень, операції порівняння і впорядкування (крім лексикографічного сортування). Наприклад, не можна знайти середнє значення категоріальних даних, значеннями яких є власні назви. Категоріальні дані характеризуються їх модою – значенням, що трапляється найчастіше у сукупності спостереження, наприклад, популярним іменем у групі студентів чи поширеною маркою автомобіля.

Для комп'ютерного моделювання категоріальні дані потрібно перетворити у числові, оскільки більшість алгоритмів машинного навчання працюють саме з числовими значеннями. Для цього можна застосовувати методи шкалювання чи кодування категорій, де кожна категорія отримує унікальне числове зображення. Наприклад, для кодування кольорів можна використати RGB-схему, де відтінки кольорів Red, Green, Blue позначаються числами від 0 до 255.

Розповсюдженим способом перетворення категорій у числову форму є унітарне кодування (One-Hot Encoding), для якого дозволеними комбінаціями бітів є ті, що містять лише один біт, рівний 1, а інші біти дорівнюють 0. Для побудови такого коду у стовпці значень категоріальної ознаки x, наприклад, $(d,a,b,c,a,d,c,b)^{\mathrm{T}}$ вибираються унікальні категорії a,b,c,d і за ними формується $n=4$ (для цього прикладу) додаткових стовпців. Входження ознаки з первинного категоріального стовпця у відповідний додатковий стовпчик відмічається 1. Наприклад:

№	Ознака x	Категорії			
		a	*b*	*c*	*d*
1	*d*	0	0	0	1
2	*a*	1	0	0	0
3	*b*	0	1	0	0
4	*c*	0	0	1	0
5	*a*	1	0	0	0
6	*d*	0	0	0	1
7	*c*	0	0	1	0
8	*b*	0	1	0	0

Ознака x є загальною назвою множини категорій $\{a,b,c,d\}$, наприклад, “температура». Значеннями a,b,c,d можуть бути семантично означені атрибути x, які інтерпретуються як категорії, наприклад, “холодно”, “тепло”, “гаряче”, “дуже гаряче”.

Для багатопараметричної вибірки необхідно зробити подібні кодування для кожного стовпчика ознак. Сформовані двійкові вектори можуть бути використані як числові значення категоріальних ознак x.

Якщо ознаки мають велику кількість унікальних значень, то кодування One-Hot призведе до значного зростання обсягу даних. Крім цього, цей спосіб призводить до лінійної залежності (мультиколінеарності) для багатофакторних моделей.

Кодування One-Hot використовується за незначної кількості ознак та малої кількості унікальних категорій. Інакше рекомендується використовувати спосіб Impact Encoding, який ще називають Target Encoding, Mean Encoding або Likelihood Encoding.

Спосіб кодування Impact Encoding полягає у такому. Нехай x є категоріальною змінною зі значеннями $(a,b,b,a,c,a)^{\mathrm{T}}$, кожному з яких відповідає певне цільове числове позначення, наприклад, таке, що є актуальною міткою класу, яку модель використовує для свого навчання. Дані групуються за категоріями і для кожної категорії розраховується середнє значення цільової змінної. Кодування категорій полягає в тому, що вони замінюються середніми значеннями цільової змінної, наприклад:

№	Ознака x	Цільове значення	Згруповані категорії	Цільове значення	Середнє цільове значення	Код категорії
1	*a*	1	*a*	1	2/3	0.667
2	*b*	0	*a*	1		0.5
3	*b*	1	*a*	0		0.5
4	*a*	1	*b*	0	1/2	0.667
5	*c*	1	*b*	1		1
6	*a*	0	*c*	1	1	0.667

Наявність різних цільових значень для однакових категорій може бути пов’язане з тим, що вони задаються для комбінації ознаки x з іншими ознаками багатофакторної моделі (тут не показано) або є наслідком неточності вхідних даних.

У кодуванні Impact Encoding, кожна категорія буде представлена своїм власним числовим кодом, який відображає статистичний вплив цієї категорії на цільову змінну. Однозначність такого кодування залежить від специфіки вхідних даних.

Існує ряд варіантів Impact-кодування. Наприклад, від середнього цільового значення кожної категорії можна відняти середнє цільове значення по усій вибірці, або відняти середнє наперед заданої базової категорії. Це дозволить уникнути випадковості в обчисленнях та може допомогти моделі краще інтерпретувати взаємозв'язки між категоріями.

Спосіб Impact Encoding є чутливим до випадкових аномалій або шумів у вхідних даних, що може призвести до неправильного кодування категорій. Також Impact Encoding може призвести до перенавчання і втрати інформації, якщо кількість унікальних значень категоріальної ознаки дуже велика, а деякі категорії мають малу кількість спостережень, оскільки середні значення можуть бути нестабільними та надто впливовими, особливо при невеликих обсягах даних.

Крім наведених способів, для кодування категоріальних ознак використовують способи Label Encoding, Hashing Trick, які забезпечують однозначне відображення кожної ознаки в унікальне число, та інші [6]. Кожен з цих способів має свої переваги та недоліки. Вибір конкретного способу залежить від конкретної задачі та особливостей набору даних.

Масштабування ознак

Коли вхідні ознаки дуже відрізняються діапазонами значень, зазвичай їх масштабують, щоб вони мали порівнювані діапазони. Можна стандартизувати вхідні значення, центруючи їх, щоб отримати нульове середнє значення при стандартному одиничному середньоквадратичному відхиленні:

$$x'_j = \frac{x_j - \mu_j}{\delta_j},\ j = 1..m,$$

де $\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{i,j}$ – середнє значення ознаки x_j для n спостережень у вхідному наборі даних, $\delta_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_{i,j} - \mu_j\right)^2}$ – стандартне середньоквадратичне відхилення значень ознак x_j у вхідному наборі даних.

Крім того, можна нормалізувати значення вхідних ознак так, щоб їхні значення знаходились у діапазоні від 0 до 1:

$$x'_j = \frac{x_j - \min\limits_i(x_{i,j})}{\max\limits_i(x_{i,j}) - \min\limits_i(x_{i,j})},\ j = 1..m.$$

Найкращим, але обчислювально затратним, вважається робастне масштабування відповідно до квартильного діапазону від 25 до 75. Для цього обчислюється медіана ознаки, яка віднімається від кожного значення ознаки і результат ділиться на міжквартильний діапазон *IRQ*:

$$x'_j = \frac{x_j - median(x_j)}{IRQ},\ j = 1..m.$$

Міжквартильний діапазон IRQ визначається як різниця між такими значеннями варіаційного ряду (ранжованої або впорядкованої вибірки), що відповідно 75% і 25% значень елементів вибірки будуть менші або рівні цим значенням. Наприклад, нехай $x_1 \le x_2 \le ... \le x_{\#25\%} \le ... \le x_{\#75\%} \le ... \le x_n$.

Елементи $x_1, x_2, ..., x_{\#25\%}$ складають 25%, а елементи $x_1, x_2, ..., x_{\#25\%}, ..., x_{\#75\%}$ – 75% від обсягу вибірки. Тоді $IRQ = x_{\#75\%} - x_{\#25\%}$.

Медіана визначає величину, розміщену посередині ранжованої вибірки:

$$median(x) = \begin{cases} x_{n\ div\ 2+1}, & n = 2k+1, \\ \dfrac{x_{n\ div\ 2} + x_{n\ div\ 2+1}}{2}, & n = 2k. \end{cases}$$

Стандартизація здебільшого використовується тоді, коли дані відповідають нормальному розподілу. Наприклад, стандартизація даних здійснюється у задачах кластеризації, в алгоритмах вимірювання відстані між об'єктами. Нормалізація часто використовується у випадку, коли дані не відповідають нормальному розподілу, наприклад, у нейронних мережах, опрацюванні зображень тощо. Робастне масштабування не чутливе до викидів у даних і використовується для розподілів, для яких не існує математичного сподівання і дисперсії.

Наявність вхідних даних із порівнянним діапазоном корисна для аналізу впливу значень ознак на прогнози і допомагає прискорити градієнтний спуск [7].

Навчання логістичної регресії методом градієнтного спуску

Навчання логістичної регресії здійснюється на тренувальній вибірці даних. Перед навчанням ваг логістичної регресії вибірка ділиться на дві частини – тренувальний (навчальний) та валідаційний (тестовий) набори з відповідним відношенням обсягів 7:3 або 8:2.

Процес навчання полягає у визначенні таких значень ваг $W^{\mathrm{T}} = (w_0, w_1, ..., w_m)$ ознак $X_i = (1, x_{i,1}, x_{i,2}, ..., x_{i,m})^{\mathrm{T}} \in R^{m+1}$, $i = 1..n$ у моделі логістичної регресії, які мінімізують логарифмічну функцію втрат:

$$F(W) = -\log L(W) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \cdot \log\sigma\left(W^{\mathrm{T}} X_i\right) + (1 - y_i) \cdot \log\left(1 - \sigma\left(W^{\mathrm{T}} X_i\right)\right)\right] \to \min_W, \qquad (1)$$

де $y_i \in \{0,1\}$ – фактична мітка класу для ознак X_i; $W^{\mathrm{T}} X_i$ – згортка ваг з ознаками; $\sigma\left(W^{\mathrm{T}} X_i\right)$ – сигмоїдна функція.

Враховуючи гладкість та унімодальність функції втрат, для пошуку оптимальних значень вагових коефіцієнтів зручно використати метод градієнтного спуску.

Градієнтний метод використовується для оптимізації функцій, тобто для визначення значення аргументу, для якого функція приймає мінімальне або максимальне значення. У задачах мінімізації функцій використовується метод градієнтного спуску, а в задачах максимізації – градієнтного підйому. Цей метод був запропонований О.-Л. Коші (A.-L. Cauchy) у 1847 р.

У теперішній час градієнтний метод використовується для розв'язування оптимізаційних задач теорії управління, економіки, машинного і глибинного навчання та інших галузей [8].

Крім класичного градієнтного методу, для оптимізації функції правдоподібності у моделях машинного навчання використовують подібні методи, наприклад, Ньютона-Рафсона (Newton-Raphson), адаптивного градієнтного спуску (Adaptive Gradient Descent), вагового покрокового вибору (Weighted Stepwise Selection) та інші, які забезпечують швидку збіжність та ефективність оптимізації параметрів моделі [9, 10].

Градієнт є векторною величиною, яка складається з часткових похідних функції по кожному регульованому параметру. Градієнт функції визначає напрямок її найшвидшого зростання у кожній точці простору параметрів, а модуль (евклідова норма вектора) – величину кроку такого зростання. Для мінімізації функції втрат необхідно використати антиградієнт – градієнт зі знаком мінус.

Градієнт логарифмічної функції втрат по вагових змінних $W^{\mathrm{T}} = (w_0, w_1, ..., w_m)$ має такий вигляд:

$$\nabla F(W) = \left(\frac{\partial F(W)}{\partial w_0}, \frac{\partial F(W)}{\partial w_1}, ..., \frac{\partial F(W)}{\partial w_m} \right).$$

Схема навчання логістичної регресії подана на рис. 1.

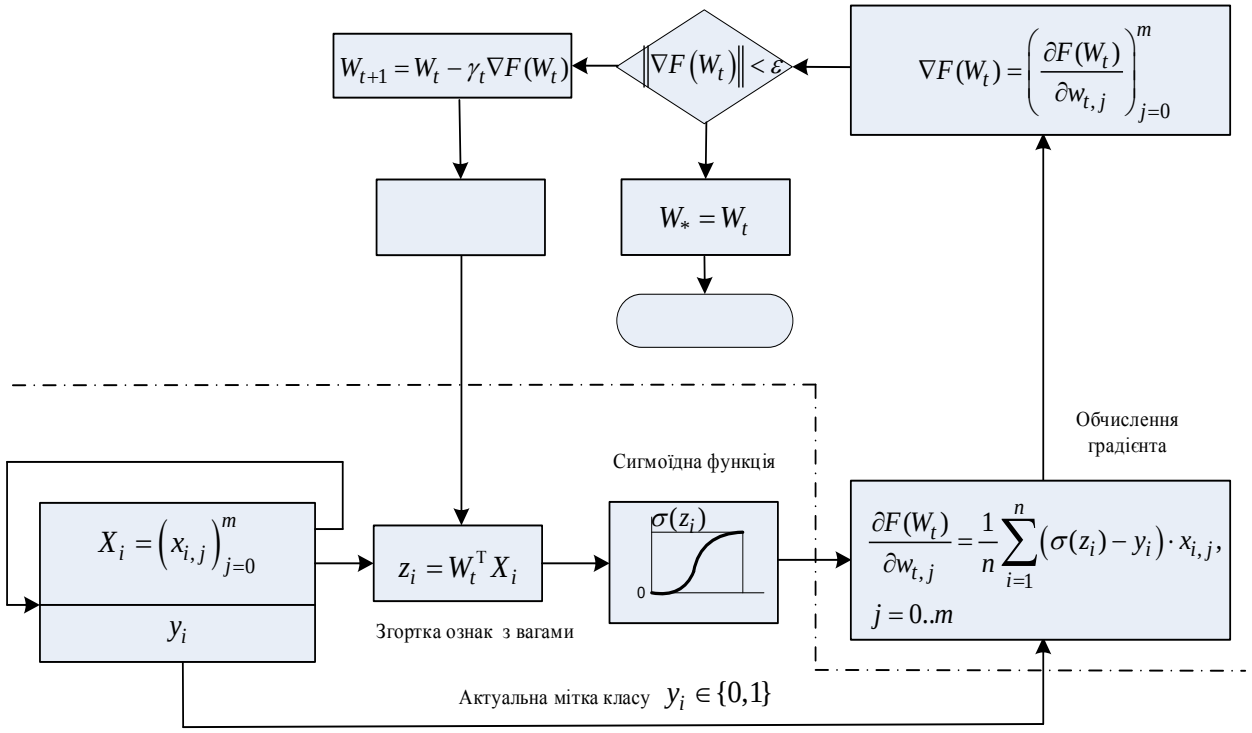

Рис. 1. Схема навчання логістичної регресії

Дані у вигляді значення ознак X_i, $i = 1..n$ об'єкта з його істинною міткою класу $y_i \in \{0,1\}$ послідовно читаються із навчальної вибірки обсягом n записів. Для кожного елемента даних обчислюється сигмоїдна функція

$$\sigma(z_i) = \frac{1}{1 + e^{-z_i}}$$

на основі згортки $z_i = W_t^{\mathrm{T}} X_i$ ознак об'єкта з фіксованим для усієї вибірки вектором ваг W_t. Початкове значення W_0 вектора ваг задається.

На кожній ітерації навчання $t = 0, 1, 2...$ обчислюється градієнт $\nabla F(W_t)$ функції логарифмічних втрат $F(W_t)$ на основі всієї вибірки даних, або міні-пакету даних, який є частиною загальної вибірки, на що вказує знак суми у формулі часткової похідної. Компоненти $\frac{\partial F(W_t)}{\partial w_{t,j}}$ градієнта обчислюються на основі значень вхідних ознак $x_{i,j}$ ($i = 1..n$, $j = 1..m$), сигмоїдної функції $\sigma(z_i)$ та актуальної (фактичної) мітки класу $y_i \in \{0,1\}$.

Після обчислення градієнта перевіряється умова точності навчання: $\left\|\nabla F\left(W_t\right)\right\| < \varepsilon$. Якщо потрібна точність не досягнута, то відбувається перерахунок вектора ваг за формулою градієнтного спуску. Далі все повторюється для нового значення вектора ваг до досягнення точності навчання. Якщо точність досягнута, то робота алгоритму начання завершується, а навчені ваги ознак вибірки об'єктів запам'ятовуються.

Градієнтний метод мінімізації функції втрат задається рекурентною залежністю:

$$W_{t+1} = W_t - \gamma_t \nabla F(W_t), \tag{2}$$

де W_t – значення вектора ваг на ітерації $t = 0,1,2,...$; γ_t – крок методу, монотонно спадна послідовність додатних величин.

Для обчислення градієнта необхідно знайти часткові похідні функції $F(W_t)$ по регульованих параметрах W_t .

Для визначення градієнта логарифмічної функції $F(W)$ втрат використаємо правило обчислення часткових похідних для складеного виразу (1).

Введемо позначення $L_i(W) = y_i \cdot \log p_i + (1-y_i)\cdot \log(1-p_i)$, де $p_i = \sigma\left(W^{\mathrm{T}} X_i\right)$. Тоді

$$F(W) = -\log L(W) = -\frac{1}{n}\sum_{i=1}^{n} L_i(W).$$

Похідна складеної функції:

$$-\frac{\partial L_i(W)}{\partial w_j} = -\frac{\partial L_i(W)}{\partial p_i}\cdot\frac{\partial p_i}{\partial z}\cdot\frac{\partial z}{\partial w_j}, \tag{3}$$

де $z = W^{\mathrm{T}} X_i$.

Тепер обчислення $\frac{\partial L_i(W)}{\partial w_j}$ зводиться до незалежного обчислення співмножників.

Похідна логарифмічної імовірності $L_i(W)$ відносно p дорівнює:

$$\frac{\partial L_i(W)}{\partial p_i} = \frac{y_i}{p_i} - \frac{1-y_i}{1-p_i}.$$

Враховуючи, що $\frac{\partial p_i}{\partial z} = p_i(1-p_i)$, де $p_i = \sigma(z)$, та $\frac{\partial z}{\partial w_j} = x_{i,j}$, де $z = W^{\mathrm{T}} X_i$, після підстановки значень часткових похідних у (3) отримаємо:

$$-\frac{\partial L_i(w)}{\partial w_j} = -\frac{\partial L_i(w)}{\partial p_i}\cdot\frac{\partial p_i}{\partial z}\cdot\frac{\partial z}{\partial w_j} = -\left(\frac{y_i}{p_i} - \frac{1-y_i}{1-p_i}\right)\cdot p_i(1-p_i)\cdot x_{i,j} =$$

$$= -\left(y_i(1-p_i) - (1-y_i)p_i\right)x_{i,j} = -(y_i - p_i)x_{i,j}.$$

Оскільки $p_i = \sigma(z) = \sigma\left(W^{\mathrm{T}} X_i\right)$, то останній вираз запишемо у такому вигляді:

$$-\frac{\partial L_i(W)}{\partial w_j} = (p_i - y_i)x_{i,j} = \left(\sigma\left(W^{\mathrm{T}} X_i\right) - y_i\right)x_{i,j}.$$

Отже, градієнт сумарної функції втрат $F(W)$ дорівнює:

$$\nabla F(W_t) = \left(\frac{1}{n}\sum_{i=1}^{n}\left(\sigma\left(W_t^{\mathrm{T}} X_i\right) - y_i\right)\cdot x_{i,j}\right)_{j=0}^{m},$$

де $x_{i,0} = 1$.

Звідси векторне рівняння градієнтного спуску (2) можна записати у конкретизованій компонентній формі:

$$w_{t+1,j} = w_{t,j} - \gamma_t \frac{\partial F(W_t)}{\partial w_{t,j}} = w_{t,j} - \frac{\gamma_t}{n} \sum_{i=1}^{n} \left(\sigma\left(W_t^{\mathrm{T}} X_i\right) - y_i \right) \cdot x_{i,j}, \ j = 0..m. \quad (4)$$

Градієнтний метод (4) забезпечує рух у сторону найбільшого зменшення функції втрат. Напрям зміни вагових параметрів на кожній ітерації буде перпендикулярним до ліній рівня функції втрат.

На швидкість збіжності рекурентного методу (4) значний вплив має параметр γ, який задає величину кроку, з яким будуть змінюватися вагові коефіцієнти. Якщо вибрати надто великий постійний крок навчання, то можна пропустити глобальний мінімум і алгоритм не зможе знайти розв'язок з необхідною точністю. Занадто малий крок навчання призведе до зростання необхідної кількості ітерацій. Тому, крок навчання змінюють у часі.

Поточне значення кроку γ_t обчислюють методом Барзилай-Борвейна (J. Barzilay, J. Borwein) [11]

$$\gamma_t = \frac{\left| \left(W_t - W_{t-1}\right)^{\mathrm{T}} \left(\nabla F\left(W_t\right) - \nabla F\left(W_{t-1}\right) \right) \right|}{\left\| \nabla F\left(W_t\right) - \nabla F\left(W_{t-1}\right) \right\|^2}$$

або знаходять з умови Вулфа (P. Wolfe) [12]

$$\gamma_t^* = \arg\min_{\gamma} F\left(W_t + \gamma \nabla F\left(W_t\right)\right).$$

Значення γ_t в умові Вулфа можна знайти наближеним лінійним пошуком.

Крім того, величина кроку γ_t відповідає умові Вулфа, якщо виконуються такі дві нерівності для варіанту мінімізації функції:

- нерівність Армійо (L. Armijo) [13]:

$$F\left(W_t - \gamma_t \nabla F\left(W_t\right)\right) \le F\left(W_t\right) - c_1 \gamma_t \left\| \nabla F\left(W_t\right) \right\|^2;$$

- умова кривизни:

$$\left(\nabla F\left(W_t\right)\right)^{\mathrm{T}} \nabla F\left(W_t - \gamma_t \nabla F\left(W_t\right)\right) \le c_2 \left\| \nabla F\left(W_t\right) \right\|^2,$$

де $0 < c_1 < c_2 < 1$.

Умови Вульфа (нерівність Армійо та умова кривизни) накладають обмеження відповідно на верхню та нижню величину кроку градієнтного методу.

Робота алгоритму градієнтного спуску завершується при досягненні точності пошуку глобального мінімуму функції втрат. Оскільки у точці мінімуму опуклої функції $F(W) = -\log L(W)$ складові градієнта дорівнюють нулю, то умову точності можна задати у вигляді $\left\| \nabla W_t \right\| < \varepsilon$, де $\left\| \cdot \right\|$ – евклідова норма, ε – потрібна точність розв'язку.

Стохастичний градієнтний спуск

Для обчислення градієнта на кожній ітерації (4) використовується сума по усій вибірці вхідних даних. Для досить великих вибірок це може призвести до значних затрат часу для пошуку оптимальних значень вагових параметрів. Тому, замість класичного градієнтного методу, у машинному навчанні частіше використовують стохастичний градієнтний метод, позбавлений необхідності обчислювати таку суму [14, 15]. Стохастичний градієнтний метод на кожній ітерації $t = 0,1,2,...$ оновлює ваги для випадково вибраного i-го елемента вибірки:

$$w_{t+1,j} = w_{t,j} - \gamma_t \frac{\partial F(W_t)}{\partial w_{t,j}} = w_{t,j} - \gamma_t \left(\sigma\left(W_t^{\mathrm{T}} X_i\right) - y_i \right) \cdot x_{i,j}, \ j = 0..m. \quad (5)$$

Для опуклої функції втрат збіжність стохастичного методу градієнтного спуску визначається такими умовами:

1) $\gamma_t \underset{t\to\infty}{\to} 0$ – параметр γ_t є додатною монотонно спадною величиною;

2) $\sum_{t=0}^{\infty} \gamma_t = \infty$ – ряд $\{\gamma_t\}_0^{\infty}$ є розбіжним;

3) $\sum_{t=0}^{\infty} \gamma_t^{\,2} < \infty$ – ряд $\{\gamma_t^{\,2}\}_0^{\infty}$ є збіжним.

Поточну величину кроку градієнтного спуску на ітерації $t = 1, 2, \ldots$ можна задати монотонно спадною функцією $\gamma_t = \frac{\gamma_0}{t^{\alpha}}$. Потрібні початкові значення параметрів $\gamma_0 > 0$ та $\alpha > 0$, які забезпечать збіжність методу градієнтного спуску, можна визначити з аналітичних оцінок [16 – 18] або підібрати експериментально.

За такого підходу на кожній ітерації не гарантується переміщення у напрямку найбільшого зменшення функції, а лише забезпечується середньостатистичний напрямок вектора руху рекурентного методу (5) у сторону оптимального розв'язку. Хоча для цього може знадобитися більше пошукових кроків, ніж у класичному методі, але перерахунок ваг здійснюється значно скоріше.

На відміну від класичного градієнтного методу, стохастичний метод на окремих ітераціях може робити невдалі кроки, але у фіналі забезпечує приблизно такий самий оптимальний розв'язок.

Процес навчання з використанням міні-пакетів даних та епох

Для надійного розпізнавання класів даних потрібно повторити навчання логістичної регресії декілька разів. Кожен із таких етапів називається епохою, що означає один повний прохід через усі навчальні дані моделі. Кожна епоха складається з ітерацій, на яких модель використовується для передбачення вихідних значень, оцінки втрат та оновлення параметрів з використанням методу градієнтного спуску. Важливо правильно підібрати кількість таких епох, оскільки велика кількість епох може призвести до перенавчання (overfitting), тоді як замала – до недонавчання (underfitting).

За перенавчання модель логістичної регресії буде добре запам'ятовувати навчальні дані і погано узагальнюватиме їх для розпізнавання класів нових даних. За недонавчання модель логістичної регресії не зможе знайти жодної закономірності розподілу даних і точність передбачення класів буде низькою.

Зазвичай, перенавчання моделі призводить до значного зростання вагових коефіцієнтів, а недонавчання – до їх надмірного зменшення.

Підбір оптимальної кількості епох є важливою складовою успішного навчання моделі логістичної регресії і залежить від складності задачі, обсягу та характеристик даних.

Вхідна вибірка даних розділяється на міні-пакети для тренування моделі. Ці міні-пакети можуть бути розміщені у пам'яті або зчитуватися з диску у міру необхідності.

Навчальний алгоритм (наприклад, градієнтний спуск) застосовується до кожного міні-пакету окремо, тобто кожна ітерація навчального алгоритму працює з окремими міні-пакетами. Для класичного градієнтного спуску це означає обчислення градієнта функції втрат за кожний міні-пакет і відповідне оновлення ваг моделі [19 – 21].

Після завершення проходу через всі міні-пакети ваги моделі зберігаються, і це вважається завершенням однієї епохи навчання.

Після кожної епохи обчислюються показники якості навчання, такі як точність на навчальних та валідаційних даних.

Далі дані перемішуються або розділяються на нові міні-пакети для наступної епохи. Це допомагає уникнути перенавчання та поліпшити узагальнення моделі.

Процес повторюється з новими міні-пакетами або перемішаними даними і попередньо навченими вагами. Це може продовжуватися до досягнення виконання певного критерію зупинки, такого як вичерпання максимальної кількості епох або зменшення втрат моделі до певного рівня. Кількість епох можна встановити так, щоб метрики якості навчання на валідаційному наборі стабілізувалися або перестали покращуватися.

Надмірна кількість епох може призвести до перенавчання, а недостатня кількість – до недонавчання алгоритму.

Перенавчання виникає, коли модель добре адаптується до навчальних даних, але погано узагальнює їх для нових, раніше не опрацьованих даних.

При досить великій кількості епох модель може пристосуватися до навчальних даних настільки добре, що вона буде погіршувати свої результати на нових даних. Крім завеликої кількості епох, до перенавчання можуть призвести занадто складна модель навчання, недостатня кількість даних у вхідній вибірці, відсутність регуляризації функції втрат.

Перенавчання є небажаним, оскільки воно призводить до погіршення ефективності моделі на нових даних. Щоб уникнути перенавчання, важливо правильно підбирати параметри моделі, використовувати методи регуляризації та контролювати кількість епох навчання.

При недостатній кількості епох навчання може виникнути недонавчання моделі. Якщо кількість епох занадто мала, модель може не мати достатньої можливості вивчити загальні закономірності даних. Додатково це може статися з таких причин: перенавчання на попередніх епохах, яке не можна виправити малою кількістю нових епох; неадекватність даних досліджуваної системи.

Щоб уникнути недонавчання, важливо експериментувати з кількістю епох та складністю моделі, використовувати методи регуляризації для контролю складності моделі, а також перевіряти, чи правильно вибрані параметри моделі. Також можна використовувати методи валідації для оцінки ефективності моделі з різною кількістю епох.

Тестування логістичної регресії

Після навчання ваг здійснюється класифікація об’єктів тестової вибірки. Тестова або контрольна вибірка не повинна перекриватися з навчальною вибіркою. У якості тестової вибірки використовуються відомі і, як правило, перевірені на інших моделях дані.

Для тестової класифікації обчислюється згортка z_i ознак $\hat{X}_i$ кожного об’єкта з попередньо навченими вагами W_* для обчислення сигмоїдної функції $\sigma(z_i)$. Значення сигмоїдної функції порівнюється з порогом $\tau \in (0,1)$ для вироблення прогнозованої мітки класу $\hat{y}_i$ для i-го елемента тестової вибірки. Зазвичай поріг $\tau = 0.5$, але може визначатися з оптимізації мір якості навчання. Схема тестування зображена на рис. 2.

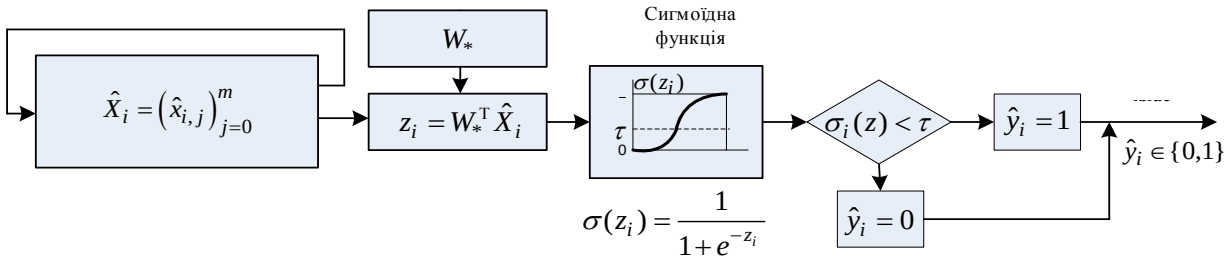

Рис. 2. Схема тестування навченої логістичної регресії

Метою тестування логістичної регресії є визначення її адекватності або класифікаційної спроможності, яка оцінюється відповідними мірами якості класифікації.

Міри якості бінарної класифікації

Міри якості класифікації обчислюються на основі матриці помилок (Confusion Matrix). Матриця помилок визначає кількості об'єктів реальних класів, спрогнозованих правильно або неправильно. Умовно клас з міткою 0 називають негативним, а клас 1 – позитивним. Для бінарної класифікації матриця помилок має вигляд:

Категорія		Прогнозований клас $\hat{y}$	
		позитивний, 1	негативний, 0
Реальний клас y	позитивний, 1	TP	FN
	негативний, 0	FP	TN

Елементи матриці помилок приймають такі значення:

- Істинно позитивний (TP, True Positive) – кількість об'єктів, що були класифіковані як позитивні (такі, що належать класу) і дійсно є такими;
- Істинно негативний (TN, True Negative) – кількість об'єктів, що були класифіковані як негативні (такі, що не належать класу) і дійсно є такими;
- Хибно позитивний (FP, False Positive) – кількість об'єктів, що були класифіковані як позитивні (такі, що належать класу), але фактично є негативними (не повинні належати цьому класу);
- Хибно негативний (FN, False Negative) – кількість об'єктів, що були класифіковані як негативні (такі, що не належать класу), але дійсно є позитивними (повинні належати цьому класу).

Елементи TP (True Positive) та TN (True Negative) вказують на правильну класифікацію, а елементи FP (False Positive) та FN (False Negative) – на помилкову класифікацію.

Алгоритм обчислення елементів матриці помилок подано на рис. 3. Він оснований на порівнянні актуальних значень міток класів y_i з мітками $\hat{y}_i$, отриманими після застосування методу логістичної регресії.

Елементи матриці помилок використовуються для обчислення мір якості класифікації та машинного навчання. Точне встановлення авторства таких мір є складним через їх широке використання у великій кількості досліджень та публікацій протягом тривалого часу, починаючи з середини 20 століття, коли статистичні та математичні методи стали інструментом аналізу даних та розпізнавання образів.

Серед декількох десятків мір класифікації найчастіше використовують такі [22]:

- Точність (Accuracy) – доля позитивних, тобто правильно спрогнозованих класів:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}.$$

Точність може давати помилкові значення для незбалансованих даних, коли кількість об'єктів одного класу значно переважає кількість об'єктів іншого класу [23].

- Влучність (точність Precision) або позитивна прогностична цінність (Positive Predictive Value, PPV) – доля істинно позитивних спрогнозованих значень (які в дійсності є позитивними), інакше – це імовірність того, що спрогнозований результат є позитивним:

$$PPV = \frac{TP}{TP + FP},$$

Показник PPV демонструє ступінь довіри до класифікатора, здатність класифікатори відрізняти один клас від іншого. Цей показник можна використовувати для незбалансованих вибірок даних.

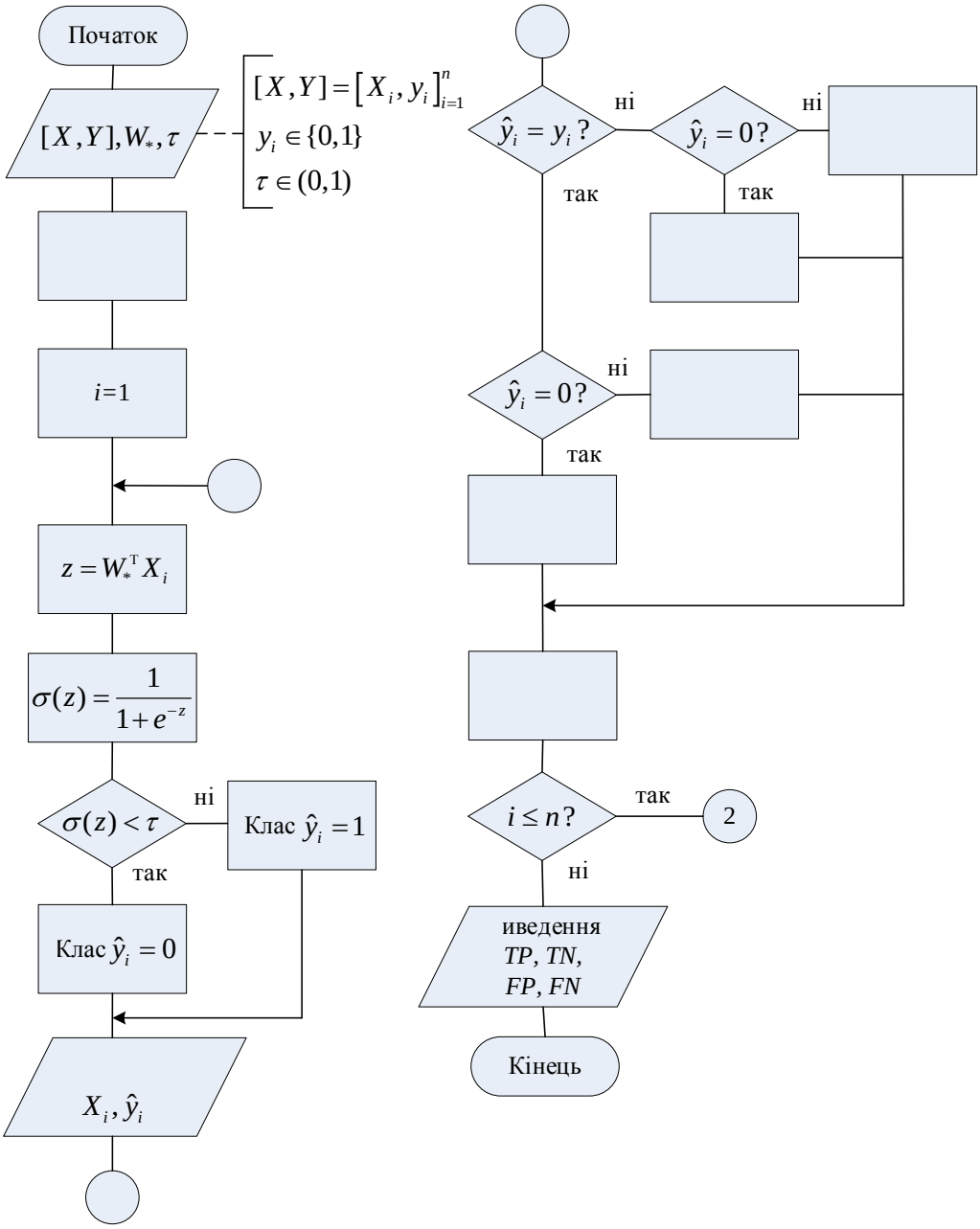

Рис. 3. Схема алгоритму класифікації та обчислення елементів матриці помилок

- Чутливість (Sensitivity), повнота (Recall) або істинно позитивний результат (True Positive Rate, TPR) – доля істинно позитивних відповідей класифікатора у загальній кількості фактично позитивних результатів або імовірність позитивного результату тесту за умови, що об'єкт справді позитивний:

$$TPR=\frac{TP}{TP+FN}.$$

Показник TPR демонструє спроможність класифікатора правильно визначати клас. Цей показник можна застосовувати для незбалансованих вибірок даних.

- Хибно позитивний результат (False Positive Rate, FPR) або коефіцієнт випадання (Fallout, FO) – доля хибно позитивних результатів у загальній кількості фактично негативних результатів:

$$FPR = \frac{FP}{FP + TN}.$$

Показник FPR визначає співвідношення між кількістю хибно позитивних результатів і загальною кількістю фактично негативних результатів.

- F_1-міра – середнє гармонійне метрик влучності та чутливості (повноти):

$$F_1 = 2 \cdot \frac{PPV \cdot TPR}{PPV + TPR}.$$

Міра F_1 належить інтервалу [0, 1]. Значення F_1 =1 вказує на ідеальні влучність та чутливість (повноту).

- Робоча характеристика ROC (Receiver Operating Characteristic) – крива помилок, що визначає результат роботи бінарного класифікатора. Будується у декартовій системі координат (x=FPR, y=TPR). Для побудови кривої обчислюються метрики FPR та TPR на основі елементів матриці помилок (рис. 3) для значень порогу $\tau \in [0,1]$ з кроком $\Delta\tau$. ROC-крива дозволяє абстрагуватись від конкретного значення порогу класифікації при порівнянні різних моделей. Ілюстративні приклади ROC-кривих подано на рис. 4.

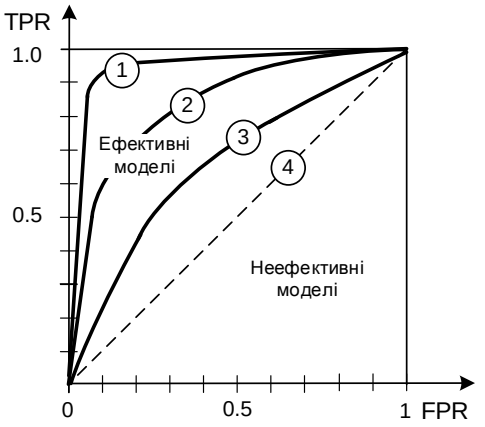

Рис. 4. Приклади ROC-кривих:
1 – відмінно; 2 – добре; 3 – задовільно;
4 – незадовільно

- Площа під кривою помилок AUC ROC (Area Under the Curve Receiver Operating Characteristic) – визначає інтегровану якість роботи алгоритму бінарної класифікації.

Більше значення AUC ROC вказує на кращий класифікатор. Практично значущими є класифікатори, для яких AUC ROC є більшим від 0.5 . Для цього можна використати таку експертну шкалу значень AUC ROC:

- відмінно: 0.9 – 1;
- дуже добре: 0.8 – 0.9;
- добре: 0.7 – 0.8;
- задовільно: 0.6 – 0.7;
- незадовільно: 0.5 – 0.6 .

Ідеальному класифікатору відповідає ROC-крива, яка проходить через точку (0, 1), площа під такою кривою дорівнює 1. Найгірший класифікатор визначається ROC-кривою, що проходить через точку (1, 0), площа під такою кривою дорівнює 0. Для класифікатора з рівноімовірним вгадуванням міток класів отримаємо близьку до прямої лінії ROC-криву, яка з'єднує точки (0, 0) та (1, 0), площа під такою лінією дорівнює 0.5. Значення AUC ROC, менші від 0.5, вказують на те, що модель працює гірше випадкового вгадування. Оцінка для таких моделей – погано або дуже погано.

Метрику AUC ROC вважають найбільш інформативним показником якості роботи класифікатора.

Практична класифікація

Після навчання і належного тестування модель логістичної регресії готова до практичного розпізнавання класів окремих об'єктів. Для визначення класу об'єкта на вхід сигмоїдної функції необхідно подати навчений вектор вагових коефіцієнтів W_* та вектор характеристичних ознак i -го об'єкта $\tilde{X}_i$.

Схема практичної класифікації методом логістичної регресії відповідає зображеній на рис. 2 схемі тестування, тільки замість тестової вибірки $\hat{X}$ на вхід подається робоча вибірка $\tilde{X}$ даних. Згортка $z_i = W_*^{\mathrm{T}} \tilde{X}_i$ є аргументом сигмоїдної функції $\sigma(z_i)$, значення якої порівнюється з порогом τ для визначення мітки прогнозованого класу $\hat{y}_i$.

Порогова функція

У схемах навчання, тестування та практичного застосування логістичної регресії використовується порогове значення $\tau \in [0,1]$, необхідне для розділення вибірки на класи. Поріг визначає імовірність компромісу між правильними і неправильними прогнозами належності об'єкта до одного із класів. Для цього вихідне значення сигмоїдної функції $\sigma(z_i)$ (прогнозована імовірність) порівнюється з пороговим значенням τ. Якщо прогнозована імовірність нижче порогу, то вважається, що об'єкт належить до класу $\hat{y}_i = 0$, інакше – до класу $\hat{y}_i = 1$.

Висота порогу впливає на значення елементів матриці помилок і, відповідно, на основні міри якості класифікації, такі як точність (Accuracy), чутливість (Sensitivity), специфічність (Specificity), F_1-показник та інші. Так, зміна порогу може призвести до зміни кількості правильно та неправильно класифікованих зразків і тим самим змінити значення точності (Accuracy) моделі. Також зміна порогу може впливати на кількість виявлених позитивних і негативних екземплярів і тим самим впливатиме на міри чутливості та специфічності. Ще зміна порогу може вплинути на точність (Precision) та чутливість, що призведе до зміни показника F_1.

За замовчуванням класифікатори працюють зі значенням порогу $\tau = 0.5$. Однак, щоб розділення на класи було найбільш адекватним, величина порогу повинна підбиратися у ході оптимізації мір якості бінарної класифікації. Наприклад, оптимальний поріг може визначатися максимальним значення показника F_1, отриманим для різних порогових значень з діапазону від 0 до 1. Налаштування порогу особливо важливе для незбалансованої вибірки, коли кількості позитивних та негативних класів значно відрізняються.

Висновки

У статті окреслено вимоги до підготовки вхідних даних для бінарної класифікації методом логістичної регресії, проведено математичне обгрунтування та наведено схему навчання логістичної регресії методом градієнтного спуску, який використовується для налаштування вагових коефіцієнтів ознак об'єктів з метою мінімізації логарифмічної функції втрат по усій вхідній вибірці. Застосування градієнтного методу обумовлене простотою його реалізації та добрими характеристиками збіжності для унімодальних функцій. Сформульовано вимоги до організації даних для багатоетапного машинного навчання для запобігання недонавчання або перенавчання моделі.

Наведено схему тестування логістичної регресії та описано основні міри ефективності навчання. Відмічено, що поріг навчання впливає на показники якості класифікації і його значення повинно визначатися розв'язуванням задачі оптимізації цих показників.

Подану у статті інформацію можна використати для моделювання і практичної побудови простих та ефективних систем аналізу і бінарної класифікації даних методом логістичної регресії.

Перспективними для дослідження є питання розширення функціональності моделі, робота з високорозмірними ознаками, нечіткими та недовизначеними даними, незбалансованими вибірками, врахування залежності між ознаками, мультикласове прогнозування, вивчення ефективності моделі в нових сферах застосування, покращення точності моделі методами регуляризації цільової функції, дослідження ефективності та змагальності алгоритмів прогнозування класів чи варіантів рішень на основі ігрових моделей.

Важливим також є практичне дослідження різних варіантів організації логістичної регресії методом комп'ютерного моделювання для визначення їх ефективності та меж застосування. Реалізація та аналіз методу комп'ютерного моделювання заслуговують окремого дослідження.

Список літератури

1. Басюк, Т. М., Литвин, В. В., Захарія, Л. М., & Кунанець, Н. Е. (2019). *Машинне навчання: навчальний посібник*. Львів: Видавництво "Новий Світ – 2000".

2. Christensen, R. (1997). *Log-Linear Models and Logistic Regression*. Springer. ISBN 10: 0387982477 / ISBN 13: 9780387982472.

3. Hosmer, D. W., & Lemeshow, S. (2000). *Applied Logistic Regression*. John Wiley & Sons Inc. DOI: https://doi.org/10.1002/0471722146.

4. Hilbe, J. M. (2009). *Logistic Regression Models (1st ed.)*. Chapman and Hall/CRC. DOI: https://doi.org/10.1201/9781420075779.

5. Cramer, J. S. (2003). The standard multinomial logit model. *In Logit Models from Economics and Other Fields, Chapter 7*. Cambridge: Cambridge University Press, 104–125. DOI: https://doi.org/10.1017/CBO9780511615412.008.

6. Leonard, T. (2020). *A course in categorical data analysis*. Taylor & Fransis.

7. Duboue, P. (2020). *The Art of Feature Engineering: Essentials for Machine Learning, 1st Edition*. Cambridge University Press.

8. Sun, T., Tang, K., & Li, D. (2022). Gradient Descent Learning With Floats. *IEEE Transactions on Cybernetics, 3* (52), 1763–1771. DOI: 10.1109/TCYB.2020.2997399.

9. Nocedal, J., & Wright, S. (2006). *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer. ISBN 9780387303031.

10. Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. Access mode: https://www.ruder.io/optimizing-gradient-descent/ .

11. Barzilai, J., & Borwein, J. M. (1988). Two-Point Step Size Gradient Methods. *IMA Journal of Numerical Analysis, 8,* 141–148. DOI: https://doi.org/10.1093/imanum/8.1.141.

12. Wolfe, P. (1969). Convergence Conditions for Ascent Methods. *SIAM Review*. *11* (2), 226–235. DOI: https://doi.org/10.1137/1011036. JSTOR 2028111.

13. Armijo, L. (1966). Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific J. Math*, *16* (1), 1–3. DOI: https://doi.org/10.2140/pjm.1966.16.1.

14. Yang, Z. (2022). Adaptive stochastic conjugate gradient for machine learning. *Expert Systems with Applications*, 206, 117719. DOI: https://doi.org/10.1016/j.eswa.2022.117719.

15. Wang, X., Yan, L., & Zhang, Q. (2021). Research on the Application of Gradient Descent Algorithm in Machine Learning. *International Conference on Computer Network, Electronic and Automation (ICCNEA), Xi'an, China*, 11–15. DOI: https://doi.org/10.1109/ICCNEA53019.2021.00014.

16. Fehrman, B., Gess, B., & Jentzen, A. (2020). Convergence Rates for the Stochastic Gradient Descent Method for Non-Convex Objective Functions. *Journal of Machine Learning Research, 21* (136), 1–48. Access mode: https://www.jmlr.org/papers/volume21/19-636/19-636.pdf.

17. Shapiro, A., & Wardi, Y. Convergence analysis of gradient descent stochastic algorithms. *Journal of Optim Theory Appl.*, 91, 439–454 (1996). DOI: https://doi.org/10.1007/BF02190104.

18. Li, X. & Orabona, F. (2019). On the Convergence of Stochastic Gradient Descent with Adaptive Stepsizes. *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics, in Proceedings of Machine Learning Research*, 89, 983–992. Access mode: https://proceedings.mlr.press/v89/li19c.html.

19. Khirirat, S., Feyzmahdavian, H. R., & Johansson, M. (2017). Mini-batch gradient descent: Faster convergence under data sparsity. *IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, VIC, Australia*, 2880–2887. DOI: https://doi.org/10.1109/CDC.2017.8264077.

20. Qi, H., Wang, F., & Wang, H. (2023) Statistical Analysis of Fixed Mini-Batch Gradient Descent Estimator. *Journal of Computational and Graphical Statistics, 32*(4), 1348–1360, DOI: 10.1080/10618600.2023.2204130.

21. Li, M., Zhang, Y., Chen, Y., & Smola, A.Y. (2014). Efficient Mini-batch Training for Stochastic Optimization. *KDD'14, August*, 24–27, New York, NY, USA. DOI: http://dx.doi.org/ 10.1145/2623330.2623612.

22. Hossin, M., & Sulaiman, M.N. (2015). A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process 5*(2), 1–11. DOI: https://doi.org/10.5121/ijdkp.2015.5201.

23. Fernández, A., García, S., Galar, M., Prati, R.C., Krawczyk, B., & Herrera, F. (2018). *Learning from Imbalanced Data Sets (1-st ed.)*. Springer.

References

1. Basyuk, T. M., Lytvyn, V. V., Zakharia, L. M., & Kunanets, N. E. (2019). *Machine learning: a study guide (in Ukrainian)*. Lviv: "Novyy Svit – 2000" Publishing House.

2. Christensen, R. (1997). *Log-Linear Models and Logistic Regression*. Springer. ISBN 10: 0387982477 / ISBN 13: 9780387982472.

3. Hosmer, D. W., & Lemeshow, S. (2000). *Applied Logistic Regression*. John Wiley & Sons Inc. DOI: https://doi.org/10.1002/0471722146.

4. Hilbe, J. M. (2009). *Logistic Regression Models (1st ed.)*. Chapman and Hall/CRC. DOI: https://doi.org/10.1201/9781420075779.

5. Cramer, J. S. (2003). The standard multinomial logit model. *In Logit Models from Economics and Other Fields, Chapter 7*. Cambridge: Cambridge University Press, 104–125. DOI: https://doi.org/10.1017/CBO9780511615412.008.

6. Leonard, T. (2020). *A course in categorical data analysis*. Taylor & Fransis.

7. Duboue, P. (2020). *The Art of Feature Engineering: Essentials for Machine Learning, 1st Edition*. Cambridge University Press.

8. Sun, T., Tang, K., & Li, D. (2022). Gradient Descent Learning With Floats. *IEEE Transactions on Cybernetics, 3* (52), 1763–1771. DOI: 10.1109/TCYB.2020.2997399.

9. Nocedal, J., & Wright, S. (2006). *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer. ISBN 9780387303031.

10. Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. Access mode: https://www.ruder.io/optimizing-gradient-descent/ .

11. Barzilai, J., & Borwein, J. M. (1988). Two-Point Step Size Gradient Methods. *IMA Journal of Numerical Analysis, 8*, 141–148. DOI: https://doi.org/10.1093/imanum/8.1.141.

12. Wolfe, P. (1969). Convergence Conditions for Ascent Methods. *SIAM Review. 11* (2), 226–235. DOI: https://doi.org/10.1137/1011036. JSTOR 2028111.

13. Armijo, L. (1966). Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific J. Math, 16* (1), 1–3. DOI: https://doi.org/10.2140/pjm.1966.16.1.

14. Yang, Z. (2022). Adaptive stochastic conjugate gradient for machine learning. *Expert Systems with Applications*, 206, 117719. DOI: https://doi.org/10.1016/j.eswa.2022.117719.

15. Wang, X., Yan, L., & Zhang, Q. (2021). Research on the Application of Gradient Descent Algorithm in Machine Learning. *International Conference on Computer Network, Electronic and Automation (ICCNEA), Xi'an, China*, 11–15. DOI: https://doi.org/10.1109/ICCNEA53019.2021.00014.

16. Fehrman, B., Gess, B., & Jentzen, A. (2020). Convergence Rates for the Stochastic Gradient Descent Method for Non-Convex Objective Functions. *Journal of Machine Learning Research, 21* (136), 1–48. Access mode: https://www.jmlr.org/papers/volume21/19-636/19-636.pdf.

17. Shapiro, A., & Wardi, Y. Convergence analysis of gradient descent stochastic algorithms. *Journal of Optim Theory Appl.*, 91, 439–454 (1996). DOI: https://doi.org/10.1007/BF02190104.

18. Li, X. & Orabona, F. (2019). On the Convergence of Stochastic Gradient Descent with Adaptive Stepsizes. *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics, in Proceedings of Machine Learning Research*, 89, 983–992. Access mode: https://proceedings.mlr.press/v89/li19c.html.

19. Khirirat, S., Feyzmahdavian, H. R., & Johansson, M. (2017). Mini-batch gradient descent: Faster convergence under data sparsity. *IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, VIC, Australia*, 2880–2887. DOI: https://doi.org/10.1109/CDC.2017.8264077.

20. Qi, H., Wang, F., & Wang, H. (2023) Statistical Analysis of Fixed Mini-Batch Gradient Descent Estimator. *Journal of Computational and Graphical Statistics*, *32*(4), 1348–1360, DOI: 10.1080/10618600.2023.2204130.

21. Li, M., Zhang, Y., Chen, Y., & Smola, A.Y. (2014). Efficient Mini-batch Training for Stochastic Optimization. *KDD'14, August*, 24–27, New York, NY, USA. DOI: http://dx.doi.org/ 10.1145/2623330.2623612.

22. Hossin, M., & Sulaiman, M.N. (2015). A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process 5*(2), 1–11. DOI: https://doi.org/10.5121/ijdkp.2015.5201.

23. Fernández, A., García, S., Galar, M., Prati, R.C., Krawczyk, B., & Herrera, F. (2018). *Learning from Imbalanced Data Sets (1-st ed.)*. Springer.

MATHEMATICAL MODEL OF LOGISTIC REGRESSION FOR BINARY CLASSIFICATION. PART 2. DATA PREPARATION, LEARNING AND TESTING PROCESSES

Petro Kravets[1], Volodymyr Pasichnyk[2], Mykola Prodaniuk[3]

Lviv Polytechnic National University,
Information Systems and Networks Department, Lviv, Ukraine,
[1] E-mail: Petro.O.Kravets@lpnu.ua, ORCID: 0000-0001-8569-423X;
[2] E-mail: Volodymyr.V.Pasichnyk@lpnu.ua, ORCID: 0000-0002-5231-6395;
[3] E-mail: Mykola.M.Prodaniuk@lpnu.ua, ORCID: 0000-0001-9544-3792

Abstract. This article reviews the theoretical aspects of logistic regression for binary data classification, including data preparation processes, training, testing, and model evaluation metrics.

Requirements for input data sets are formulated, methods of coding categorical data are described, methods of scaling input features are defined and substantiated.

A scheme for learning logistic regression using the gradient descent method has been developed to minimize the loss function by the appropriate adjustment of the weights of the features of the sample of objects intended for classification. Features of the construction of recurrent methods of classical and stochastic gradient descent are determined. The requirements for the organization of the data sample for the multi-stage learning model in order to avoid overtraining or undertraining of logistic regression are described.

The scheme of testing the trained logistic regression is given and the main quality metrics of binary classification are described. The influence of the height of the classification threshold on the efficiency of logistic regression was noted.

According to the results of the work, the directions of perspective research of logistic regression are outlined.

Keywords: mathematical model, logistic regression, binary classification, data analysis, machine learning, sigmoid function, logarithmic loss function, gradient descent, classification threshold, classification quality metrics.