

# Занятие 9. Базовые методы классификации

---

Гирдюк Дмитрий Викторович

8 ноября 2025

СПбГУ, ПМ-ПУ, ДФС

# Задача классификации i

- Постановка задачи обучения с учителем (supervised learning): необходимо предсказать значение целевой переменной  $y \in Y$  объекта по набору его признаков  $x \in X$
- Среда описывается совместным распределением  $f_{X,Y}(x, y)$ , а выборкой из нее является набор пар  $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^N$
- Результирующая модель возвращает значение  $y$  по признакам  $x$ :  $y = h(x; \theta)$
- Классификация:
  - $Y = \{0, 1\}$  – бинарная (binary)
  - $Y = \{1, 2, \dots, K\}$  – многоклассовая (multiclass)
  - $Y = \{0, 1\}^K$  – многозначная (multi-label)

## Задача классификации ii

- Если представить, что мы знаем апостериорное распределение  $P(Y = y|x)$ , то не составит труда задать правило  $h(x)$  для предсказания:

$$h(x) = \operatorname{argmax}_{y \in Y} P(Y = y|x)$$

- Распределения неизвестны  $\implies$  можно оценить с помощью статистических методов

# Метод максимума правдоподобия

- Формула Байеса

$$P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})}$$

где  $P(\mathcal{D}|\theta)$  называется правдоподобием

- Если данные в обучающей выборке независимы и одинаково распределены (аббревиатура iid), то

$$P(\mathcal{D}|\theta) = \prod_{i=1}^N p(y^{(i)}|x^{(i)}, \theta)$$

- Простая идея: пусть  $\theta$  доставляет максимум правдоподобию, т.е.

$$\theta_{\text{mle}} = \arg \max_{\theta} P(\mathcal{D}|\theta)$$

- Но на практике чаще работают с отрицательным логарифмом правдоподобия – минимизацией отрицательной суммы логарифмов условных вероятностей

## Логистическая регрессия

---

# Логистическая регрессия i

- Логистическая регрессия появляется из желания смоделировать апостериорное распределение линейной функцией, гарантировав, что все вероятности суммируются в единицу и лежат в интервале  $[0; 1]$
- Это возможно при использовании logit-преобразования:

$$\log \frac{P(Y = 1|X = \mathbf{x})}{P(Y = K|X = \mathbf{x})} = \boldsymbol{\theta}_1^T \mathbf{x},$$

$$\log \frac{P(Y = 2|X = \mathbf{x})}{P(Y = K|X = \mathbf{x})} = \boldsymbol{\theta}_2^T \mathbf{x},$$

...

$$\log \frac{P(Y = K - 1|X = \mathbf{x})}{P(Y = K|X = \mathbf{x})} = \boldsymbol{\theta}_{K-1}^T \mathbf{x}.$$

- Замечание: в данной секции вектор  $\mathbf{x}$  дополнен единицей

## Логистическая регрессия ii

- Так как сумма вероятностей должна быть равна единице, легко вывести функцию вероятности (softmax):

$$P(Y = k|X = \mathbf{x}) = \frac{\exp(\boldsymbol{\theta}_k^T \mathbf{x})}{1 + \sum_{j=1}^{K-1} \exp(\boldsymbol{\theta}_j^T \mathbf{x})}, \quad k = 1, \dots, K - 1,$$

$$P(Y = K|X = \mathbf{x}) = \frac{1}{1 + \sum_{j=1}^{K-1} \exp(\boldsymbol{\theta}_j^T \mathbf{x})}$$

- Таким образом, распределение  $P(Y = y|X = \mathbf{x}) = p_y(\mathbf{x}; \boldsymbol{\theta})$  параметризовано  $\boldsymbol{\theta} = [\boldsymbol{\theta}_1^T, \dots, \boldsymbol{\theta}_{K-1}^T]$
- Вопрос: как выглядит отрицательный логарифм правдоподобия?

## Логистическая регрессия iii

- Отрицательный логарифм правдоподобия в таком случае имеет вид

$$\begin{aligned} NLL(\boldsymbol{\theta}) &= -\log \prod_{i=1}^N \prod_{k=1}^K p_k(\mathbf{x}^{(i)}; \boldsymbol{\theta})^{[y^{(i)}=k]} = \\ &= -\sum_{i=1}^N \sum_{k=1}^K [y^{(i)} = k] \log(p_k(\mathbf{x}^{(i)}; \boldsymbol{\theta})) = \\ &= -\sum_{i=1}^N \log p_{y^{(i)}}(\mathbf{x}^{(i)}; \boldsymbol{\theta}) \end{aligned}$$

## Логистическая регрессия для бинарного случая i

- В случае бинарной классификации нам достаточно одной функции для задания распределения. Для удобства заменим класс 2 на класс 0. Тогда

$$p(\mathbf{x}; \boldsymbol{\theta}) = P(Y = 1 | X = \mathbf{x}) = \frac{\exp(\boldsymbol{\theta}^T \mathbf{x})}{1 + \exp(\boldsymbol{\theta}^T \mathbf{x})} = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

где функция  $\sigma(x) = \frac{1}{1+\exp(-x)}$  называется *сигмоидой*

- Отрицательный логарифм правдоподобия может быть записан как

$$\begin{aligned} NLL(\boldsymbol{\theta}) &= - \sum_{i=1}^N \left[ y^{(i)} \log p(\mathbf{x}^{(i)}; \boldsymbol{\theta}) + (1 - y^{(i)}) \log \left( 1 - p(\mathbf{x}^{(i)}; \boldsymbol{\theta}) \right) \right] \\ &= - \sum_{i=1}^N \left[ y^{(i)} \boldsymbol{\theta}^T \mathbf{x}^{(i)} - \log \left( 1 + \exp(\boldsymbol{\theta}^T \mathbf{x}^{(i)}) \right) \right] \end{aligned}$$

## Поиск решения i

- Производная  $NLL$  по вектору параметров  $\theta$

$$\begin{aligned}\frac{\partial NLL}{\partial \theta} &= - \sum_{i=1}^N \left[ y^{(i)} \mathbf{x}^{(i)} - \frac{1}{1 + \exp(\theta^T \mathbf{x}^{(i)})} \exp(\theta^T \mathbf{x}^{(i)}) \mathbf{x}^{(i)} \right] = \\ &= \sum_{i=1}^N \mathbf{x}^{(i)} \left[ \sigma(\theta^T \mathbf{x}^{(i)}) - y^{(i)} \right]\end{aligned}$$

- Гессиан можете посчитать в свободное время – он положительно определенный, следовательно функция  $NLL$  выпуклая. Если свободного времени нет, то обратитесь к литературе [1] (пункт 10.2.3.4)

## Поиск решения ii

- Ну а дальше различные численные методы оптимизации. Вариации градиентного спуска, итеративно перевзвешенный метод наименьших квадратов (IRLS), методы второго порядка (например, метод Ньютона-Рафсона). Хотя вычисление гессиана достаточно трудозатратно, потому чаще используют квази-Ньютоновские методы. Например, BFGS, LBFGS

## Логистическая регрессия в scikit-learn i

- Логистическая регрессия представлена в линейных моделях scikit-learn – LogisticRegression [2]
- Поддерживает те же регуляризации (параметр `penalty`), что и обычная линейная регрессия:  $L_1$ ,  $L_2$  и ElasticNet. Параметр  $C$  отвечает за степень регуляризации
- Есть возможность прокинуть весовые коэффициенты (параметр `class_weight`), присутствует также балансировка при дисбалансе наблюдений в тренировочном множестве
- Масса алгоритмов оптимизации для нахождения решения: выбор зависит от способа регуляризации. По умолчанию, LBFGS
- Как обычно, не забываем фиксировать `random_state`

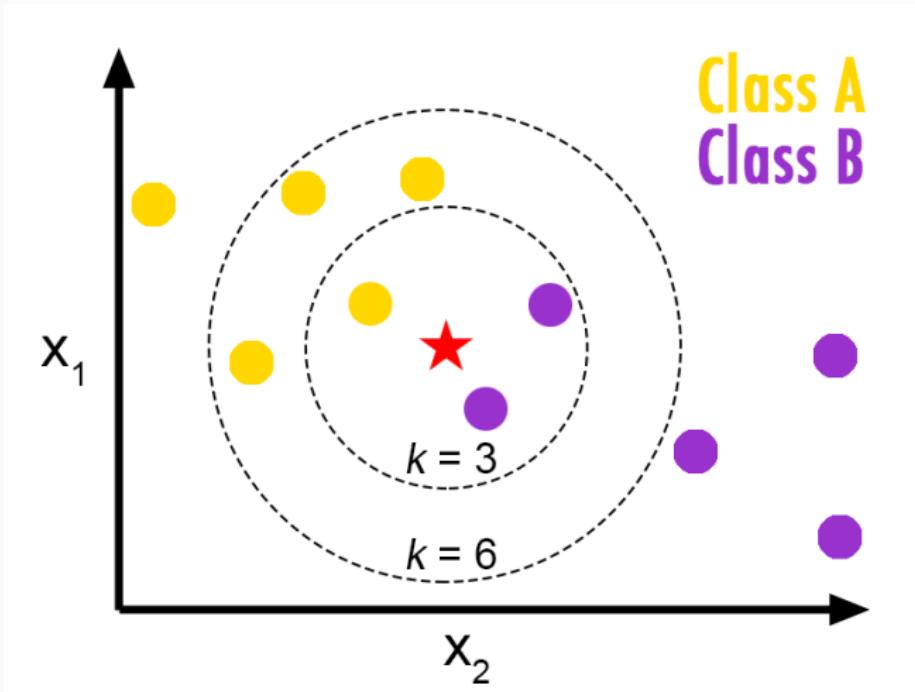
## Метод $k$ -ближайших соседей

---

# Метод $k$ -ближайших соседей

- Метод  $k$ -ближайших соседей ( $k$ -nearest neighbors, kNN) – относительно простой метрический алгоритм для задач классификации, основанный на оценке схожести некоторого наблюдения/объекта/сэмпла и классифицированных ранее его соседей
- Классифицируемое наблюдение относится к классу, преобладающему среди  $k$  ближайших соседей наблюдения
- Близость определяется некоторой фиксированной метрикой (например, евклидовой)
- Основное предположение заключается в том, что близкие наблюдения (в смысле значения метрики) принадлежат одному классу (так называемая «гипотеза компактности»)

### Пример [3]



## Формальное определение i

- Имеем размеченную обучающую выборку  
 $X = \{\boldsymbol{x}^{(i)}\}_{i=1}^N, Y = \{y^{(i)}\}_{i=1}^N, \boldsymbol{x}^{(i)} \in \mathbb{R}^m, y^{(i)} \in \mathbb{N}$
- Выберем некоторую метрику  $\rho(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$
- Отсортируем для некоторого нового наблюдения  $\hat{\boldsymbol{x}}$  объекты обучающей выборки  $\boldsymbol{X}$ :

$$\rho(\hat{\boldsymbol{x}}, \boldsymbol{x}^{(o_1)}) \leq \rho(\hat{\boldsymbol{x}}, \boldsymbol{x}^{(o_2)}) \leq \dots \leq \rho(\hat{\boldsymbol{x}}, \boldsymbol{x}^{(o_n)})$$

- Тогда метод ближайших соседей формально записывается в виде

$$\hat{y} = \arg \max_{y \in Y} \sum_{i=1}^N \left[ y^{(o_i)} = y \right] \omega(i, \hat{\boldsymbol{x}}),$$

где  $\omega(i, \hat{\boldsymbol{x}})$  есть весовая функция, которая оценивает степень важности  $o_i$ -го наблюдения для классификации  $\hat{\boldsymbol{x}}$

## Формальное определение ii

- $\omega(i, \hat{x}) = [i = 1]$  – метод ближайшего соседа
- $\omega(i, \hat{x}) = [i \leq k]$  – метод k-ближайших соседей

## Выбор $k$

- Понятно, что при  $k = 1$  метод является неустойчивым к выбросам, а при  $k = n$  все новые наблюдения будут относиться к наиболее частотному классу
- На практике  $k$  выбирается либо на основе внешних свойств исследуемой области, либо путем кросс-валидации

## Типы наблюдений

- Наблюдения можно разделить на 3 типа: эталоны, неинформативные и выбросы
- Эталоны – самые информативные наблюдения, типичные представители своего класса
- Когда в некоторой области признакового пространства содержится большое количество эталонных наблюдений, многие из них становятся неинформативными: удалив их, это никоим образом не скажется на качестве классификации
- Под выбросами понимаются как наблюдения, достаточно далеко удаленные от всех остальных, так и те, что находятся в пределах большого числа наблюдений другого класса
- Чем меньше в обучающей выборке неинформативных наблюдений и выбросов, тем лучше качество классификации

# Масштабируемость

- Чем больше обучающая выборка, тем дольше происходит классификация
- Если в решаемой задаче необходимо последовательное дообучение, вычисление расстояния до всех наблюдений становится весьма неэффективным
- В таком случае, необходимы эффективные реализации поиска соседей на основе специфических структур данных/индексов (например, KD-деревья), или вовсе специальные схемы аппроксимации (например, Hierarchical Navigable Small Worlds, HNSW)

# Выбор метрики $i$

- Метрика должна достаточно адекватно отражать схожесть наблюдений в признаковом пространстве. Проблема состоит в том, что понятие "адекватно" сложно формализовать
- Числовые признаки практически всегда необходимо нормализовывать. Иначе вклад одних будет затмевать другие. Впрочем, некоторые признаки могут быть куда более значимыми, чем другие
- Проклятие размерности тоже никто не отменял. Если признаков много, то сумма отклонений между компонентами двух наблюдений приведет к тому, что большинство наблюдений будут равноудалены относительно друг друга (см. закон больших чисел). Зато можно брать произвольное  $k$ !

## Выбор метрики $\|\cdot\|$

- Отсюда следует, что либо признаки следует каким-либо образом отбирать, либо задавать им в метрике весовые коэффициенты. Или вовсе «обучать метрику» (см. metric learning) под признаковое пространство

# Обсуждение

- Метод  $k$ -ближайших соседей – отличное базовое решение
- kNN имеет всего 2 гиперпараметра, каждый из которых имеет принципиальное значение
- Обобщается на задачи регрессии: значение вычисляется как среднее значений по соседям

## $kNN$ в scikit-learn [4]

- $kNN$  реализован в scikit-learn: KNeighborsClassifier и KNeighborsRegressor
- Есть поддержка разреженных данных
- Кроме числа соседей  $k$  можно задавать следующее:
  - weights, веса наблюдений. Либо равнозначны (дефолтное), либо с учетом расстояния до соседей
  - algorithm. Способ поиска соседей: брутфорс, ball-дерево, KD-дерево, и автоматический подбор подходящего с учетом обучающей выборки (дефолтное)
  - leaf\_size. Максимальный размер листа в дереве, если выбрано ball/KD-дерево
  - metric и  $p$ . Метрику можно как реализовать самостоятельно, так и использовать из имеющегося: Минковского ( $p$  – ее параметр) и ее частные случаи (Чебышева и Манхэттенская)

# Использованные источники i

1. **Murphy K. P. Probabilistic Machine Learning: An introduction.**  
MIT Press, 2022. URL: probml.ai.
2. **LogisticRegression in scikit learn.** URL:  
[https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LogisticRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html).
3. **kNN illustration.** URL: <https://gist.github.com/wanibala84/e8c15faa69081dc0c68d79d5cdb12398>.
4. **kNN in scikit learn.** URL:  
<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>.