Многоклассовая классификация. Нелинейные модели классификации

Елена Кантонистова

ВШЭ, 2023

МНОГОКЛАССОВАЯ КЛАССИФИКАЦИЯ

ПОДХОД ONE-VS-ALL

Решаем задачу классификации на K классов.

- Обучим K бинарных классификаторов $b_1(x), \dots, b_K(x)$, каждый из которых решает задачу: *принадлежит объект x к классу k_i или не принадлежит?*

Например, линейные классификаторы будут иметь вид

$$b_k(x) = sign((w_k, x) + w_{0k})$$

ПОДХОД ONE-VS-ALL

Решаем задачу классификации на K классов.

- Обучим K бинарных классификаторов $b_1(x), \dots, b_K(x)$, каждый из которых решает задачу: *принадлежит объект x к классу k_i или не принадлежит?*

Например, линейные классификаторы будут иметь вид

$$b_k(x) = sign((w_k, x) + w_{0k})$$

- Тогда в качестве итогового предсказания будем выдавать класс самого уверенного классификатора:

$$a(x) = \underset{k \in \{1,\dots,K\}}{arg\max}((w_k, x) + w_{0k})$$

ПОДХОД ONE-VS-ALL

Решаем задачу классификации на K классов.

- Обучим K бинарных классификаторов $b_1(x), \dots, b_k(x)$, каждый из которых решает задачу: *принадлежит объект x к классу k_i или не принадлежит?*

Например, линейные классификаторы будут иметь вид

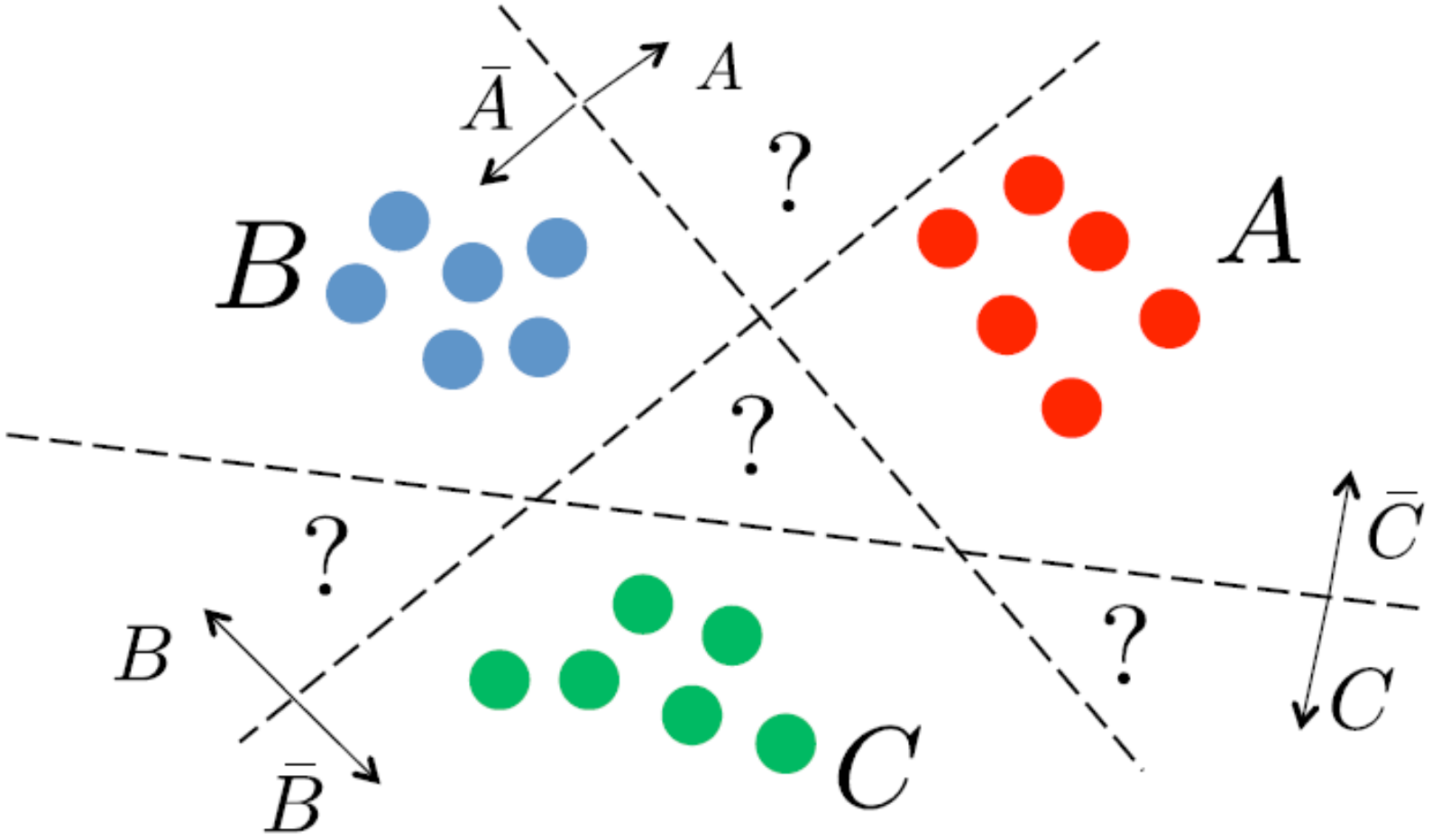
$$b_k(x) = sign((w_k, x) + w_{0k})$$

- Тогда в качестве итогового предсказания будем выдавать класс самого уверенного классификатора:

$$a(x) = \underset{k \in \{1,\dots,K\}}{arg\max}((w_k, x) + w_{0k})$$

- Предсказания классификаторов могут иметь разные масштабы, поэтому сравнивать их некорректно.

ПОДХОД ONE-VS-ALL



ПОДХОД ALL-VS-ALL

- Для каждой пары классов i и j обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс i или j

(если всего K классов, то получим C_K^2 классификаторов).

Каждый такой классификатор будем обучать только на объектах классов i и j.

ПОДХОД ALL-VS-ALL

- Для каждой пары классов i и j обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс i или j

(если всего K классов, то получим C_K^2 классификаторов).

Каждый такой классификатор будем обучать только на объектах классов i и j.

- В качестве итогового предсказания выдадим класс, который предсказало наибольшее число алгоритмов:

$$a(x) = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}} \sum_{i=1}^{K} \sum_{j \neq i} [a_{ij}(x) = k]$$

ПОДХОД ALL-VS-ALL

- Для каждой пары классов i и j обучим бинарный классификатор $a_{ij}(x)$, который будет предсказывать класс i или j

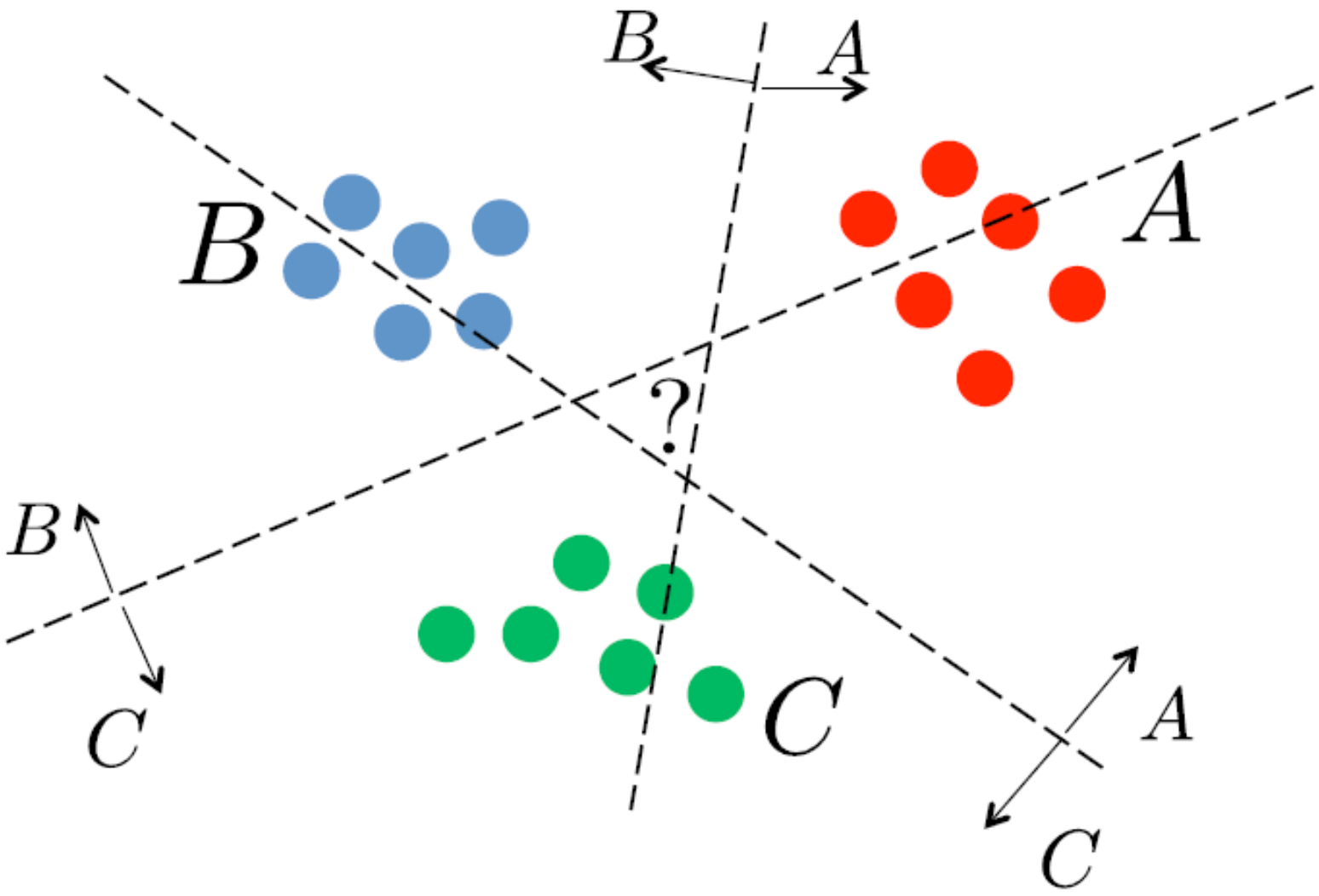
(если всего K классов, то получим **C_K^2 классификаторов**).

Каждый такой классификатор будем обучать только на объектах классов i и j.

- В качестве итогового предсказания выдадим класс, который предсказало наибольшее число алгоритмов:

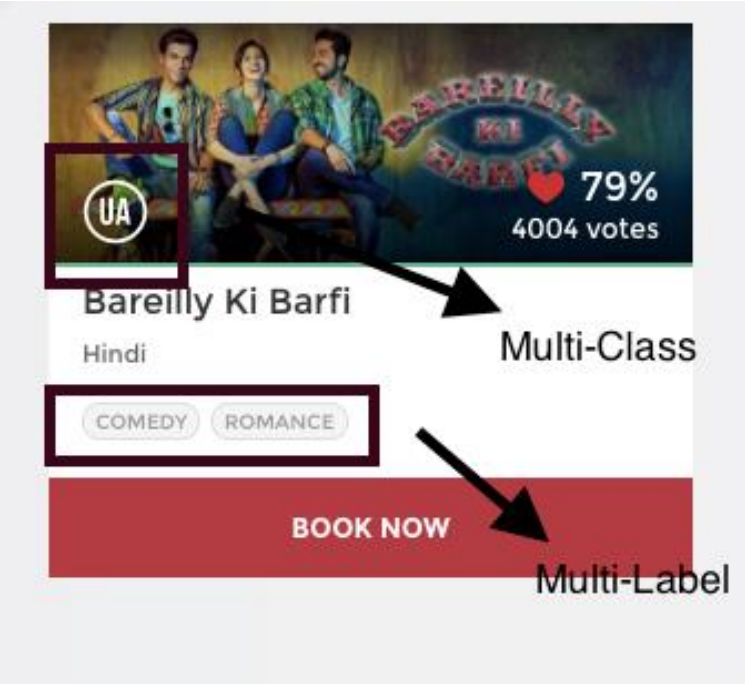
$$a(x) = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}} \sum_{i=1}^{K} \sum_{j \neq i} [a_{ij}(x) = k]$$

ПОДХОД ALL-VS-ALL



MULTICLASS AND MULTI-LABEL CLASSIFICATION

- Если каждый объект может принадлежать только одному классу, то решаем задачу multiclass классификации
- Если каждый объект может принадлежать нескольким классам (задача классификации с пересекающимися классами), то решаем задачу multi-label классификации.



Метрики

В случае, если классов больше, чем два, также можно построить матрицу ошибок.

Например, в задаче с тремя классами может получиться следующая матрица (определяем, с каким животным имеем дело, с кошкой, рыбой или курицей):

		True/Actual		
		Cat	Fish	Hen
Predicted	Cat	4	6	3
	Fish	1	2	0
	Hen	1	2	6

Для вычисления точности и полноты в этом случае существует несколько подходов:

- Микроусреднение (micro-average)
- Макроусреднение (macro-average)
- Взвешенное усреднение (weighted-average)

Метрики: macro-average

Макроусреднение (macro-average)

В этом подходе мы вычисляем значение выбранной метрики для каждой бинарной ситуации (кошка/не кошка, рыба/не рыба, курица/не курица), а затем усредняем полученные числа.

Например, посчитаем точность и полноту для ситуации кошка/не кошка:

		True/Actual		
		Cat	Fish	Hen
Predicted	Cat	4	6	3
	Fish	1	2	0
	Hen	1	2	6

- $precision(cat) = \frac{TP}{TP + FP} = \frac{4}{4 + 6 + 3} = \frac{4}{13}$

То есть false positive - это все объекты, которые модель ошибочно назвала кошкой (их 6+3)

- $recall(cat) = \frac{TP}{TP + FN} = \frac{4}{4 + 1 + 1}$

Здесь false negative - это все кошки, которых модель не нашла (кошки, названные моделью не кошками).

Тогда macro-average

$$precision = \frac{precision(cat) + precision(fish) + precision(hen)}{3}$$

Метрики: weighted average

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐔)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐔)	1	2	6

Взвешенное усреднение (weighted-average)

В этом подходе мы усредняем посчитанные для каждого класса метрики с весами, пропорциональными количеству объектов класса.

То есть weighted average

$$precision = \frac{6}{25} \cdot precision(cat) + \frac{10}{25} \cdot precision(fish) + \frac{9}{25} \cdot precision(hen)$$

так как всего 25 объектов, и из них 6 кошек, 10 рыб и 9 куриц.

Метрики: micro-average

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐓)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐓)	1	2	6

Микроусреднение (micro-average)

В этом подходе мы вычисляем значения TP, TN, FP, FN по всей матрице ошибок сразу, исходя из их определения. Затем по полученным числам вычисляем выбранные метрики.

$$precision = \frac{12}{12 + 13} = \frac{12}{25}$$

TP - это количество верно угаданных объектов положительного класса. В нашем случае $TP = 4 + 2 + 6 = 12$

FP - это суммарное количество false positive-предсказаний. Например, если cat предсказана как fish, то это false positive для fish. Таким образом, FP - это сумма всех неверных предсказаний, то есть
$FP = 6 + 3 + 1 + 0 + 1 + 2 = 13$

Метрики: micro-average

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐓)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐓)	1	2	6

Далее,

$$recall = \frac{TP}{TP + FN}$$

FN - это сумма false negative-предсказаний. Например, если cat предсказана как fish, то это false negative для cat. Таким образом, FN - это опять же сумма всех неверных предсказаний, то есть

$FN = 6 + 3 + 1 + 0 + 1 + 2 = 13$

- Получается, что в случае микроусреднения $precision = recall$
- И так как f1-score - это среднее гармоническое точности и полноты, то при микроусреднении

$precision = recall = f1$

МНОГОКЛАССОВАЯ ЛОГИСТИЧЕСКАЯ РЕГРЕССИЯ

- Бинарная лог.регрессия предсказывает вероятность класса 1:

$$(w, x) \to a(x) = \frac{1}{1 + \mathrm{e}^{-(w,x)}} = \frac{e^{(w,x)}}{1 + e^{(w,x)}}$$

- Предположим, у нас есть K линейных моделей, каждая из которых дает оценку принадлежности выбранному классу: $b_k(x) = (w_k, x)$.

- Преобразуем вектор предсказаний в вектор вероятностей (softmax-преобразование):

$$\boldsymbol{softmax(b_1, \dots, b_K) = \left(\frac{exp(b_1)}{\sum_{i=1}^{K} exp(b_i)}, \frac{exp(b_2)}{\sum_{i=1}^{K} exp(b_i)}, \dots, \frac{exp(b_K)}{\sum_{i=1}^{K} exp(b_i)}\right)}$$

Тогда вероятность класса k:

$$P(y = k|x, w) = \frac{\exp((w_k, x))}{\sum_{i=1}^{K} \exp((w_i, x))}$$

ОБУЧЕНИЕ ВЕСОВ МОДЕЛИ

$$a_j(x) = P(y = j|x, w) = \frac{\exp(b_j(x))}{\sum_{i=1}^{K} \exp(b_i(x))}$$

Обучение – по методу максимального правдоподобия (аналогично бинарной классификации):

$$\Pi = \prod_{i=1}^{n} a_1(x_i)^{[y_i=1]} \cdot a_2(x_i)^{[y_i=2]} \cdot \dots a_K(x_i)^{[y_i=K]} =$$

$$= \prod_{i=1}^{n} \prod_{j=1}^{K} a_j(x_i)^{[y_i=j]} \to \max_{w_1,\dots,w_K}$$

$$-\sum_{i=1}^{n} \sum_{j=1}^{K} [y_i = j]\, log P(y = j|x_i, w) \to \min_{w_1,\dots,w_K}$$

ОБУЧЕНИЕ ВЕСОВ МОДЕЛИ

$$a_j(x) = P(y = j|x, w) = \frac{\exp(b_j(x))}{\sum_{i=1}^{K} \exp(b_i(x))}$$

Обучение – по методу максимального правдоподобия (аналогично бинарной классификации):

$$\Pi = \prod_{i=1}^{n} a_1(x_i)^{[y_i=1]} \cdot a_2(x_i)^{[y_i=2]} \cdot \dots a_K(x_i)^{[y_i=K]} =$$

$$= \prod_{i=1}^{n} \prod_{j=1}^{K} a_j(x_i)^{[y_i=j]} \to \max_{w_1,\dots,w_K}$$

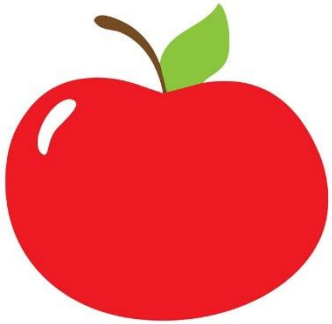
То есть в итоге обучаем одну модель (а не K моделей)

$$-\sum_{i=1}^{n} \sum_{j=1}^{K} [y_i = j] \, log P(y = j|x_i, w) \to \min_{w_1,\dots,w_K}$$

НАИВНЫЙ БАЙЕСОВСКИЙ КЛАССИФИКАТОР

НАИВНЫЙ БАЙЕСОВСКИЙ КЛАССИФИКАТОР

Наивный байесовский классификатор – это алгоритм классификации, основанный на теореме Байеса с допущением о независимости признаков.



Пример: фрукт может считаться яблоком, если:

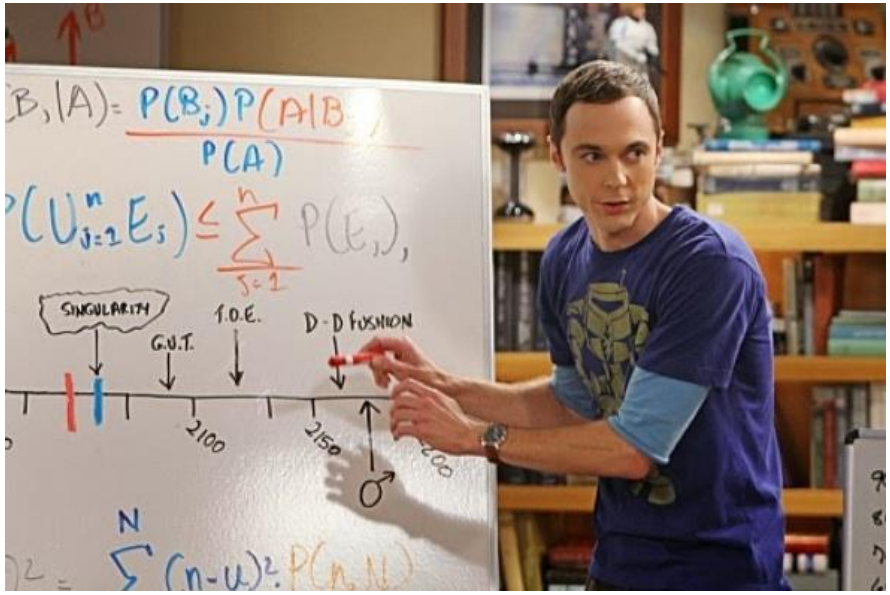
1) он красный
2) круглый
3) его диаметр составляет порядка 8 см

Предполагаем, что признаки вносят независимый вклад в вероятность того, что фрукт является яблоком.

ТЕОРЕМА БАЙЕСА

Теорема Байеса:

$$P(c|x) = \frac{P(x|c) \cdot P(c)}{P(x)}$$



- $P(c|x)$ - вероятность того, что объект со значением признака x принадлежит классу c.
- $P(c)$ – априорная вероятность класса c.
- $P(x|c)$ - вероятность того, что значение признака равно x при условии, что объект принадлежит классу c.
- $P(x)$ – априорная вероятность значения признака x.

ПРИМЕР РАБОТЫ БАЙЕСОВСКОГО АЛГОРИТМА

Пример: на основе данных о погодных условиях необходимо определить, состоится ли матч.

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

- Преобразуем набор данных в следующую таблицу:

Weather	No	Yes
Overcast	0	4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

ПРИМЕР РАБОТЫ БАЙЕСОВСКОГО АЛГОРИТМА

Решим задачу с помощью теоремы Байеса:

$$P(Yes|Sunny) = P(Sunny|Yes) \cdot P(Yes)/P(Sunny)$$

Таблица частот				
Weather	**No**	**Yes**		
Overcast	0	4	**=4/14**	**0.29**
Rainy	3	2	**=5/14**	**0.36**
Sunny	2	3	**=5/14**	**0.36**
Grand Total	5	9		
	=5/14	**=9/14**		
	0.36	**0.64**		

- $P(Sunny|Yes) = \frac{3}{9}, P(Sunny) = \frac{5}{14}, P(Yes) = \frac{9}{14}.$
- $P(Yes|Sunny) = \frac{3}{9} \cdot \frac{9}{14} : \frac{5}{14} = \frac{3}{5} = 0{,}6 \Rightarrow 60\%.$

БАЙЕСОВСКИЙ АЛГОРИТМ ДЛЯ КЛАССИФИКАЦИИ

Аналогичным образом с помощью наивного байесовского алгоритма можно прогнозировать несколько различных классов на основе множества признаков.

+ классификация быстрая и простая

+ в случае, если выполняется предположение о независимости, классификатор показывает очень высокое качество

- если в тестовых данных присутствует категория, не встречавшаяся в данных для обучения, модель присвоит ей нулевую вероятность

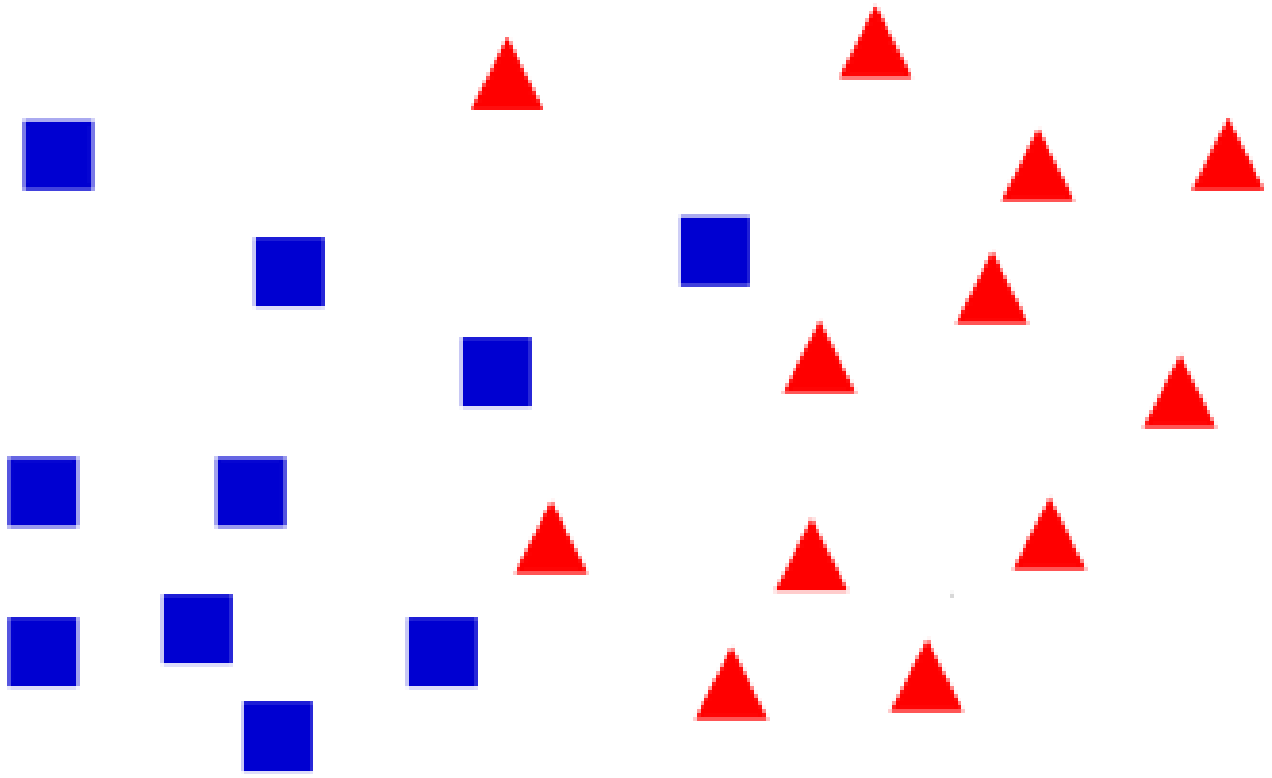
НАИВНЫЙ БАЙЕСОВСКИЙ АЛГОРИТМ

https://scikit-learn.org/stable/modules/naive_bayes.html

МЕТОД БЛИЖАЙШИХ СОСЕДЕЙ

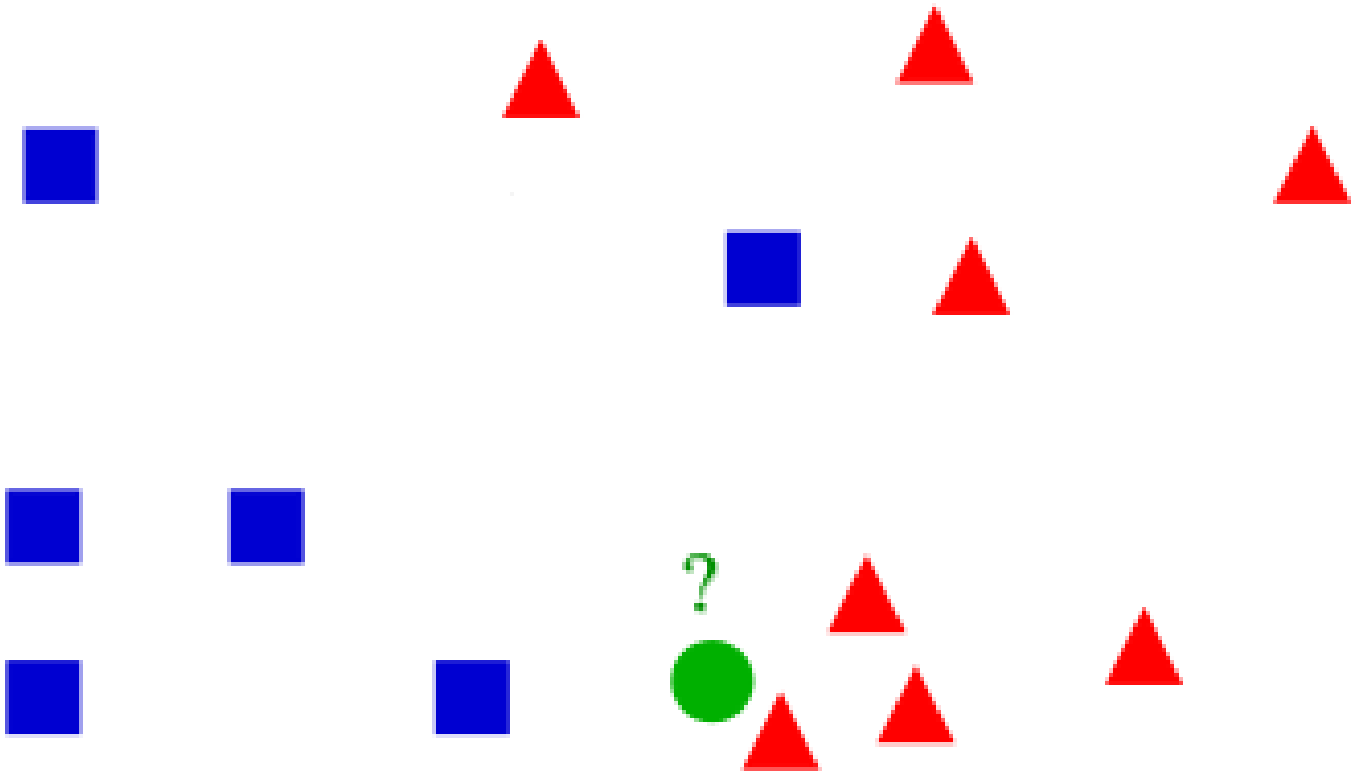
МЕТОД БЛИЖАЙШИХ СОСЕДЕЙ

Идея: схожие объекты находятся близко друг к другу в пространстве признаков.



МЕТОД БЛИЖАЙШИХ СОСЕДЕЙ

Как классифицировать новый объект?



МЕТОД БЛИЖАЙШИХ СОСЕДЕЙ

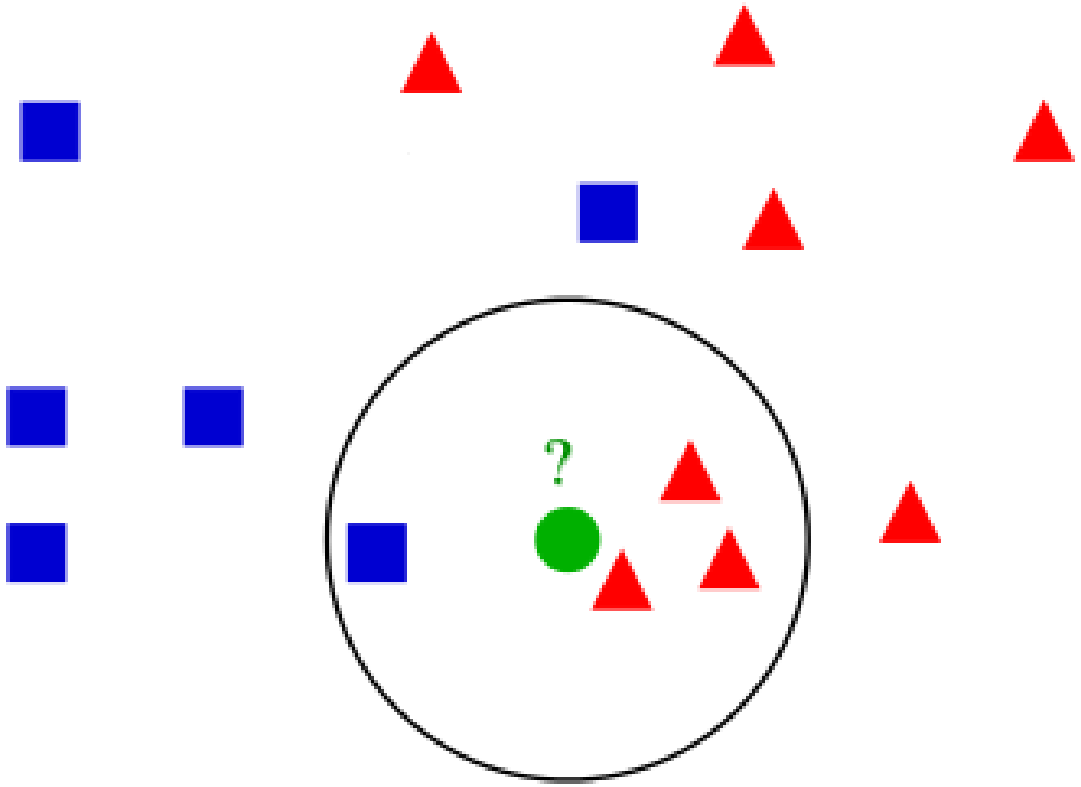
Чтобы классифицировать новый объект, нужно:

- Вычислить расстояние до каждого из объектов обучающей выборки.
- Выбрать k объектов обучающей выборки, расстояние до которых минимально.
- Класс классифицируемого объекта — это класс, наиболее часто встречающийся среди k ближайших соседей.

МЕТОД БЛИЖАЙШИХ СОСЕДЕЙ

Число ближайших соседей k – гиперпараметр метода.

Например, для $k = 4$ получим:



То есть объект будет отнесён к классу *треугольников*.

ФОРМАЛИЗАЦИЯ МЕТОДА

Пусть k – количество соседей. Для каждого объекта u возьмём k ближайших к нему объектов из тренировочной выборки:

$$x_{(1;u)}, x_{(2;u)}, \dots, x_{(k;u)}.$$

Тогда класс объекта u определяется следующим образом:

$$\boldsymbol{a(u) = \operatorname*{argmax}_{y \in Y} \sum_{i=1}^{k} [y(x_{(i;u)}) = y]}.$$

ФОРМАЛИЗАЦИЯ МЕТОДА

Пусть k – количество соседей. Для каждого объекта u возьмём k ближайших к нему объектов из тренировочной выборки:

$$x_{(1;u)}, x_{(2;u)}, \dots, x_{(k;u)}.$$

Тогда класс объекта u определяется следующим образом:

$$a(u) = \underset{y \in Y}{\operatorname{argmax}} \sum_{i=1}^{k} [y(x_{(i;u)}) = y].$$

Ближайшие объекты – это объекты, расстояние от которых до данного объекта наименьшее по некоторой метрике ρ.

ФОРМАЛИЗАЦИЯ МЕТОДА

Пусть k – количество соседей. Для каждого объекта u возьмём k ближайших к нему объектов из тренировочной выборки:

$$x_{(1;u)}, x_{(2;u)}, \dots, x_{(k;u)}.$$

Тогда класс объекта u определяется следующим образом:

$$a(u) = \underset{y \in Y}{\operatorname{argmax}} \sum_{i=1}^{k} [y(x_{(i;u)}) = y].$$

Ближайшие объекты – это объекты, расстояние от которых до данного объекта наименьшее по некоторой метрике ρ.

- В качестве метрики ρ как правило используют евклидово расстояние, но можно использовать и другие метрики.
- Перед использованием метода необходимо масштабировать данные, иначе признаки с большими числовыми значениями будут доминировать при вычислении расстояний.

ОБОБЩЕНИЯ

Как учесть расстояния до ближайших объектов?

- Можно задать веса $w_k = \frac{1}{k}$, где k – номер ближайшего соседа:

$$a(u) = \underset{y \in Y}{\operatorname{argmax}} \sum_{i=1}^{k} \frac{[y(x_{(i;u)}) = y]}{k}.$$

- Можно использовать более хитрые функции весов (метод Парзеновского окна):

$$a(u) = \underset{y \in Y}{\operatorname{argmax}} \sum_{i=1}^{k} K(\frac{\rho(u, x_i)}{h})[y(x_{(i;u)}) = y]$$

ЯДРА В МЕТОДЕ ПАРЗЕНОВСКОГО ОКНА

- $K(x) = \frac{1}{2}[|x| \leq 1]$ – прямоугольное ядро
- $K(x) = (1 - |x|) \cdot [|x| \leq 1]$ – треугольное ядро
- $K(x) = \frac{1}{\sqrt{2\pi}} \exp(-2x^2)$ - гауссовское ядро

На практике из-за простоты чаще всего используется прямоугольное ядро.

KNN В ЗАДАЧЕ РЕГРЕССИИ

- Простой вариант – усреднить целевые переменные у ближайших соседей

$$a(u) = \frac{1}{k}\sum_{i=1}^{k} y_i$$

- Можно использовать вариант с весами (формула Надарая-Ватсона):

$$a(u) = \frac{\sum_{i=1}^{k} K\left(\frac{\rho(u, x_i)}{h}\right) y_i}{\sum_{i=1}^{k} K(\frac{\rho(u, x_i)}{h})}$$

KNN: ИТОГИ

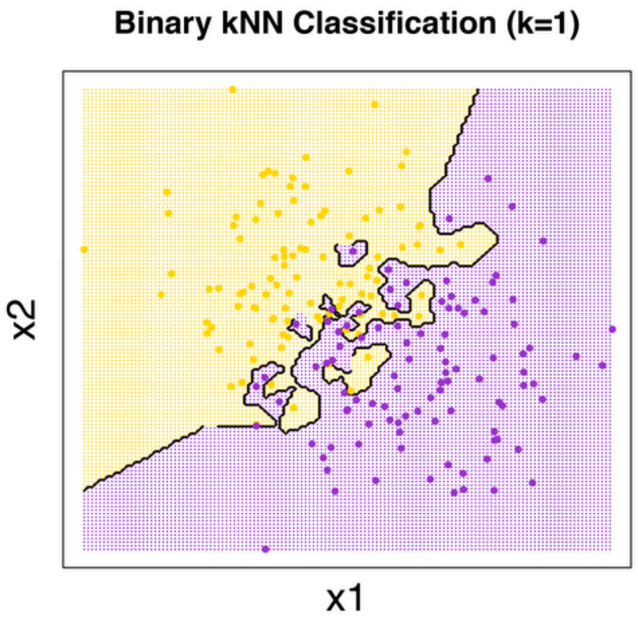
Преимущества:

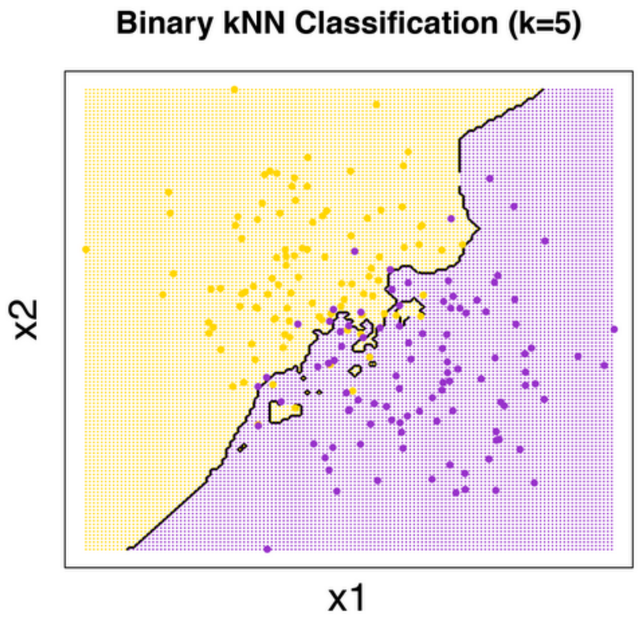
- Простой алгоритм
- Не делает никаких предположений о данных
- Иногда довольно хорошо работает
- Применяется и для классификации, и для регрессии

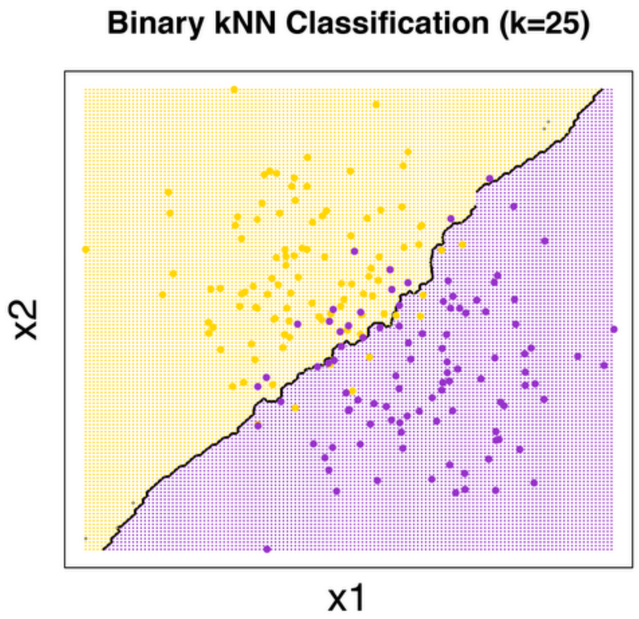
Недостатки:

- Требует больших затрат по времени и по памяти
- Чувствителен к масштабу данных
- Зависит от выбранной метрики (которую не всегда просто или даже невозможно подобрать под особенности задачи)

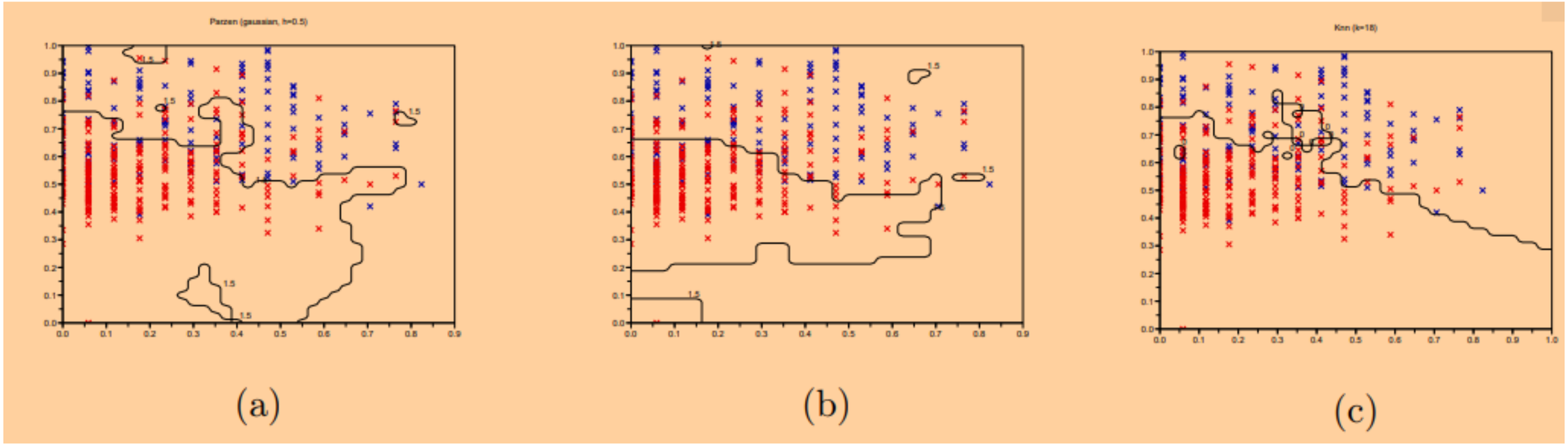
Влияние числа соседей в задаче классификации



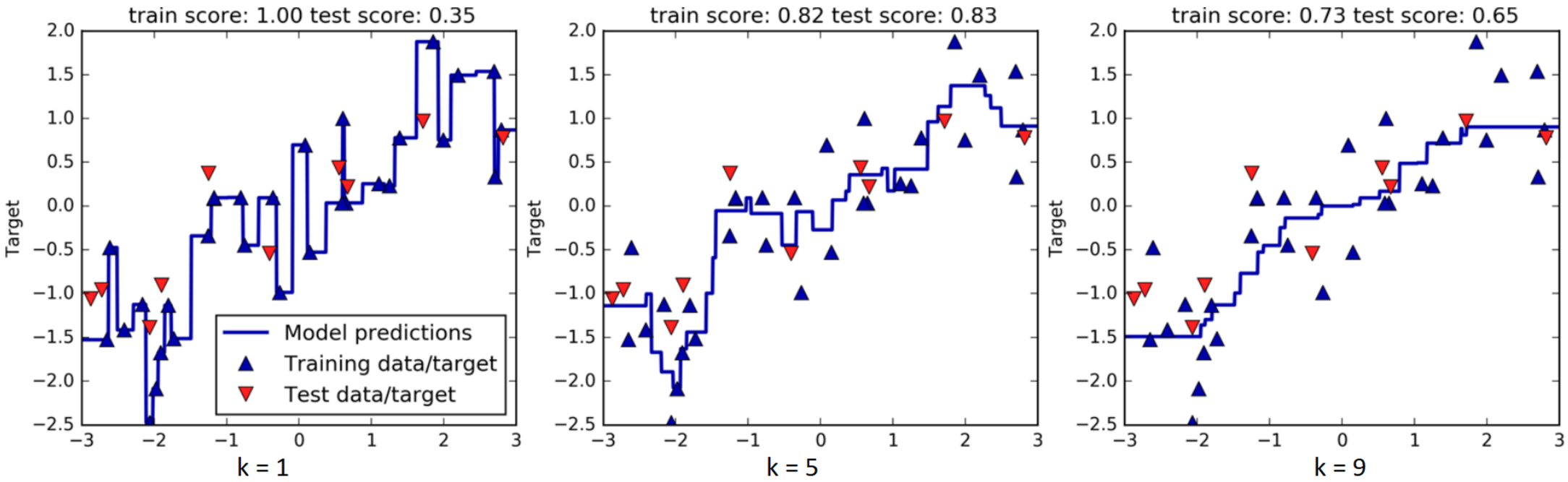




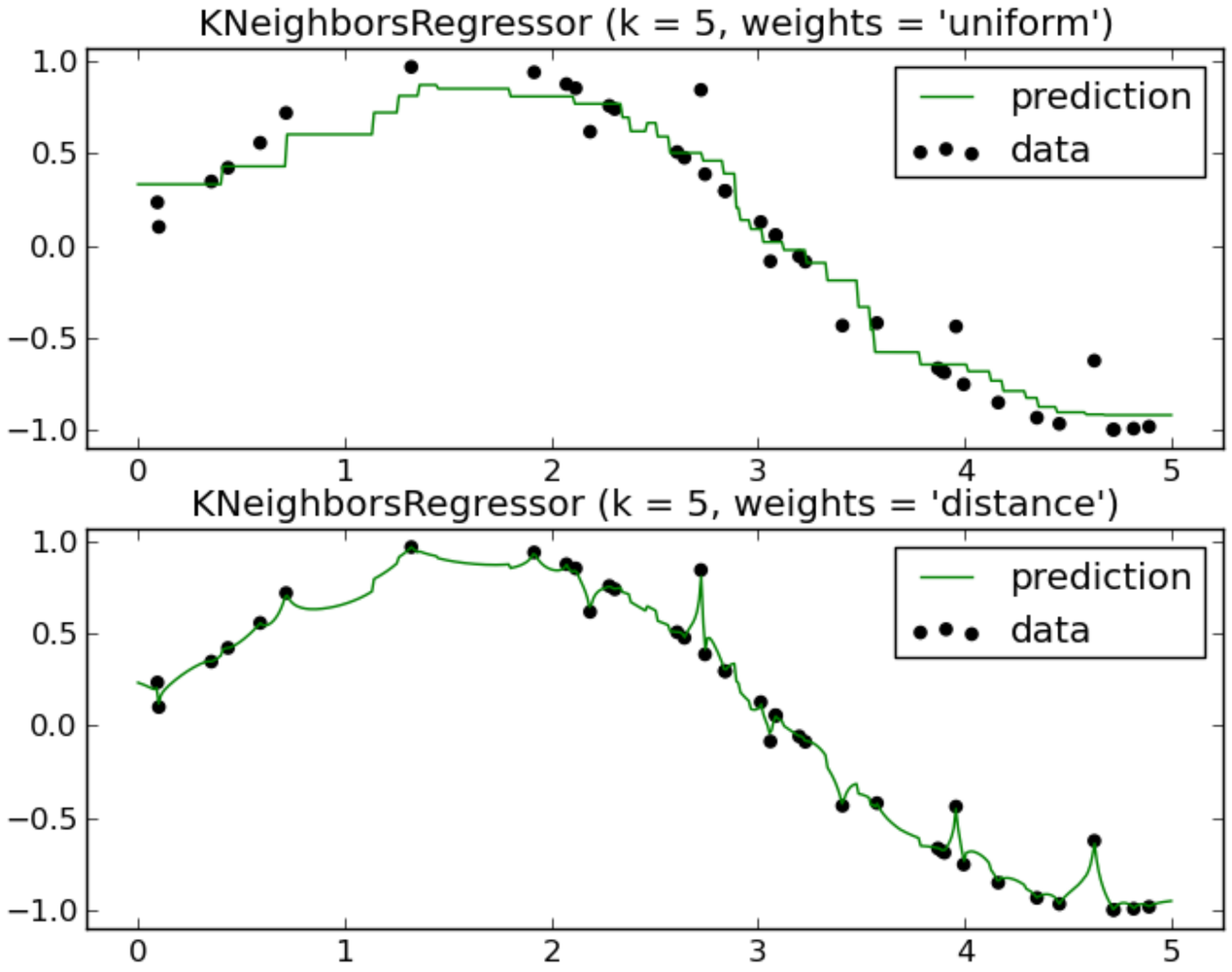
Влияние ядра в задаче классификации



Влияние числа соседей в задаче регрессии



Влияние вариантов взвешивания в регрессии



КАЛИБРОВКА ВЕРОЯТНОСТЕЙ

Калибровка вероятностей - приведение ответов алгоритма к значениям, близким к вероятностям объектов принадлежать конкретному классу.

Зачем это нужно?

- Вероятности гораздо проще интерпретировать
- Вероятности могут дать дополнительную информацию о результатах работы алгоритма

КАЛИБРОВКА ПЛАТТА

- Пусть есть два класса, $Y = \{+1, -1\}$

Задача: для классификатора $a(x)$, предсказывающего значения из отрезка $[0, 1]$, либо предсказывающего класс (+1 или -1), сделать калибровку, чтобы предсказания были вероятностями $p(y = +1|x)$.

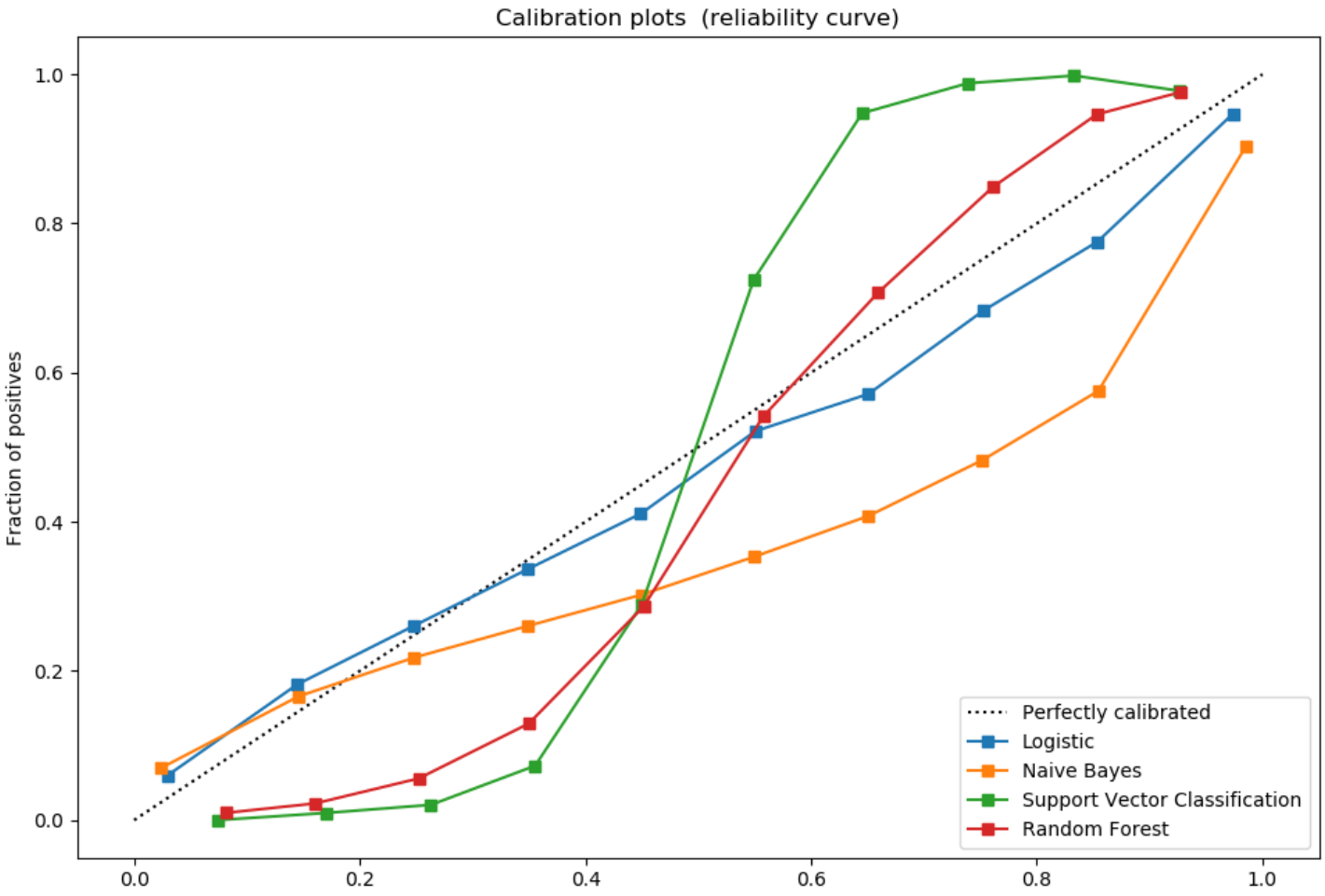
КАЛИБРОВКА ПЛАТТА

- Пусть есть два класса, $Y = \{+1, -1\}$

Задача: для классификатора $a(x)$, предсказывающего значения из отрезка $[0, 1]$, либо предсказывающего класс (+1 или -1), сделать калибровку, чтобы предсказания были вероятностями $p(y = +1|x)$.

Идея: обучаем логистическую регрессию на ответах классификатора $a(x)$.

ПРИМЕР ИЗ SKLEARN



КАЛИБРОВКА ПЛАТТА

- Пусть есть два класса, $Y = \{+1, -1\}$

Задача: для классификатора $a(x)$, предсказывающего значения из отрезка $[0, 1]$, либо предсказывающего класс (+1 или -1), сделать калибровку, чтобы предсказания были вероятностями $p(y = +1|x)$.

Идея: ***обучаем логистическую регрессию на ответах классификатора*** $\boldsymbol{a(x)}$.

КАЛИБРОВКА ПЛАТТА

- Пусть есть два класса, $Y = \{+1, -1\}$

Задача: для классификатора $a(x)$, предсказывающего значения из отрезка $[0, 1]$, либо предсказывающего класс (+1 или -1), сделать калибровку, чтобы предсказания были вероятностями $p(y = +1|x)$.

Идея: ***обучаем логистическую регрессию на ответах классификатора*** $\boldsymbol{a(x)}$.

- $\pi(x; \alpha; \beta) = \sigma(\alpha \cdot \boldsymbol{a(x)} + \beta) = \frac{1}{1+e^{-(\alpha \cdot a(x)+\beta)}}$

КАЛИБРОВКА ПЛАТТА

- Пусть есть два класса, $Y = \{+1, -1\}$

Задача: для классификатора $a(x)$, предсказывающего значения из отрезка $[0, 1]$, либо предсказывающего класс (+1 или -1), сделать калибровку, чтобы предсказания были вероятностями $p(y = +1|x)$.

Идея: ***обучаем логистическую регрессию на ответах классификатора*** $a(x)$.

- $\pi(x; \alpha; \beta) = \sigma(\alpha \cdot a(x) + \beta) = \frac{1}{1+e^{-(\alpha \cdot a(x)+\beta)}}$
- Находим α и β, минимизируя логистическую функцию потерь (*то есть обучаем логистическую регрессию*):

$$-\sum_{y_i=-1} \log\big(1 - \pi(x; \alpha; \beta)\big) - \sum_{y_i=+1} \log\big(\pi(x; \alpha; \beta)\big) \to \min_{\alpha,\beta}$$