

Машинное обучение

Лекция 5

Метрические методы. Мультиклассовая классификация.

Михаил Гуцин

mhushchyn@hse.ru

НИУ ВШЭ, 2022



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

На прошлой лекции

- ▶ Матрица ошибок (Confusion matrix)
- ▶ Доля правильных ответов (Accuracy)
- ▶ Точность (Precision)
- ▶ Полнота (Recall)
- ▶ F_1 -мера
- ▶ ROC кривая
- ▶ Precision-Recall кривая

План

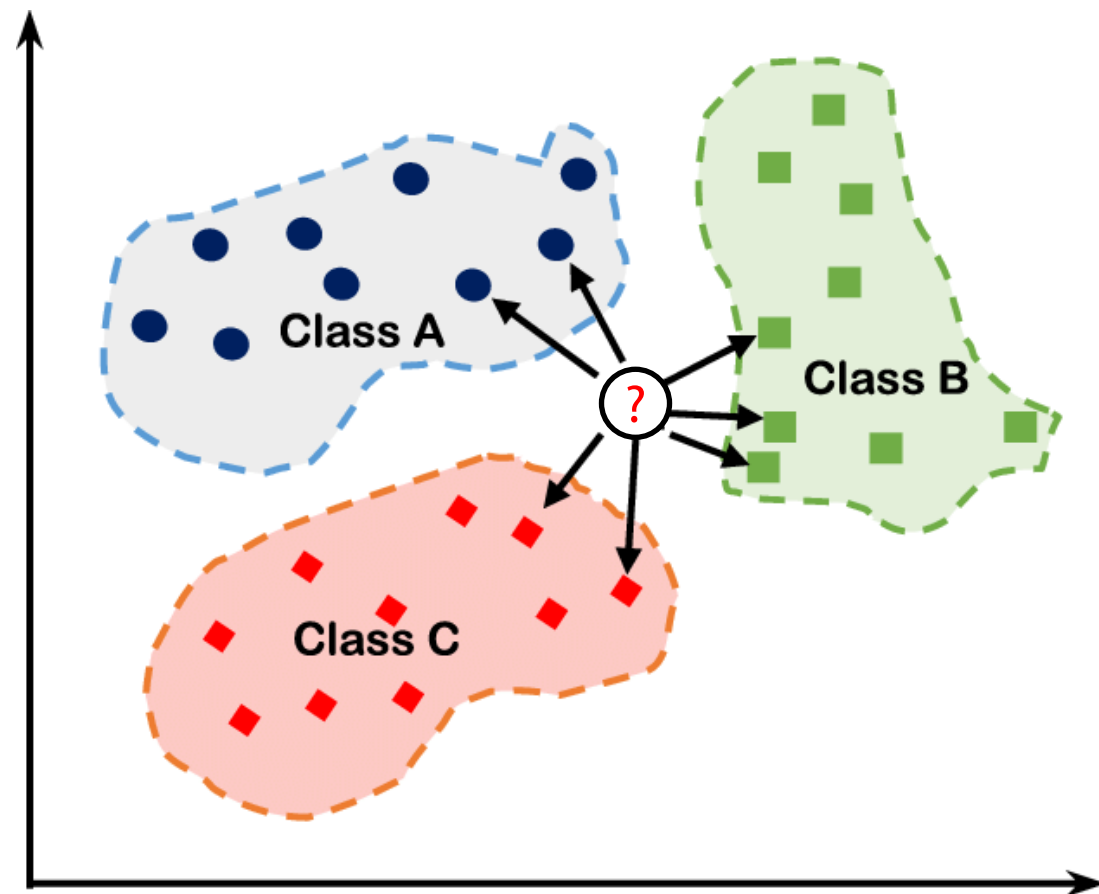
- ▶ Метрические методы
 - kNN для классификации
 - kNN для регрессии
- ▶ Мультиклассовая классификация
 - Сведение к бинарной классификации
 - Логистическая регрессия на K классов
 - Метрики качества
 - Классификация с пересекающимися классами

kNN для классификации



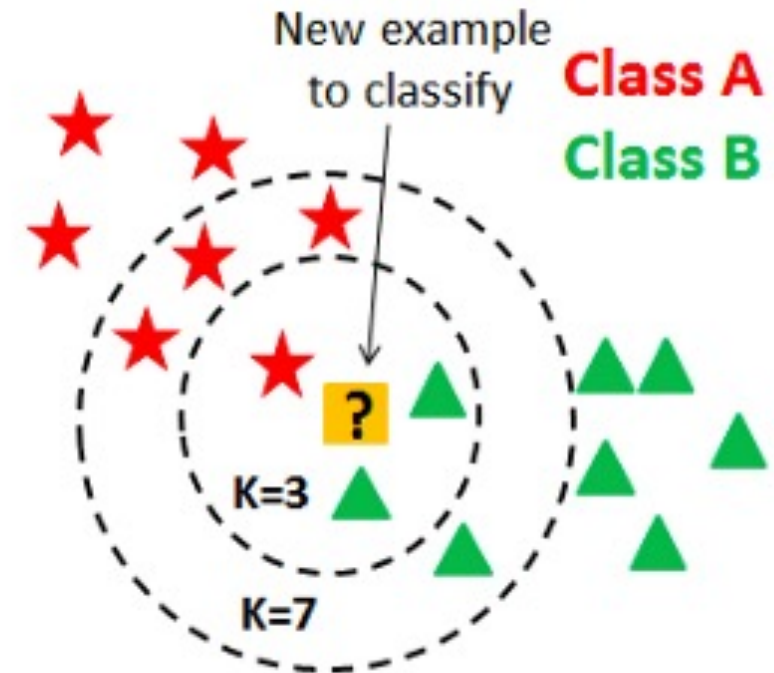
Метрические методы

Метрические методы в машинном обучении – это методы, которые используют **расстояния** между объектами



Задача

- ▶ Есть объекты **нескольких** классов
- ▶ Для каждого **нового** объекта нужно определить его класс
- ▶ Будем **сравнивать** новые объекты с уже известными



Алгоритм k Nearest Neighbors #1

- ▶ Пусть дан набор из n точек: $\{x_i, y_i\}_{i=1}^n$, где
 - $x_i = (x_{i1}, x_{i2}, \dots, x_{id})^T$ - вектор из d признаков объекта;
 - $y_i = \{0, 1, 2, \dots, m\}$ – метка класса объекта.
- ▶ Пусть дана функция расстояния между двумя объектами:

$$\rho(x_i, x_j): X \times X \rightarrow [0, \infty)$$

- симметричная и неотрицательная
- является мерой (не обязательно)

Алгоритм k Nearest Neighbors #2

- ▶ Запоминаем обучающую выборку $\{x_i, y_i\}_{i=1}^n$.
- ▶ Никаких весов для обучения в алгоритме нет, модель не строится.
- ▶ Для каждого нового объекта **u** сортируем объекты обучающей выборки по расстоянию:

$$\rho(u, x_u^{(1)}) \leq \rho(u, x_u^{(2)}) \leq \dots \leq \rho(u, x_u^{(k)})$$

– $x_u^{(i)}$ - i -й сосед объекта u .

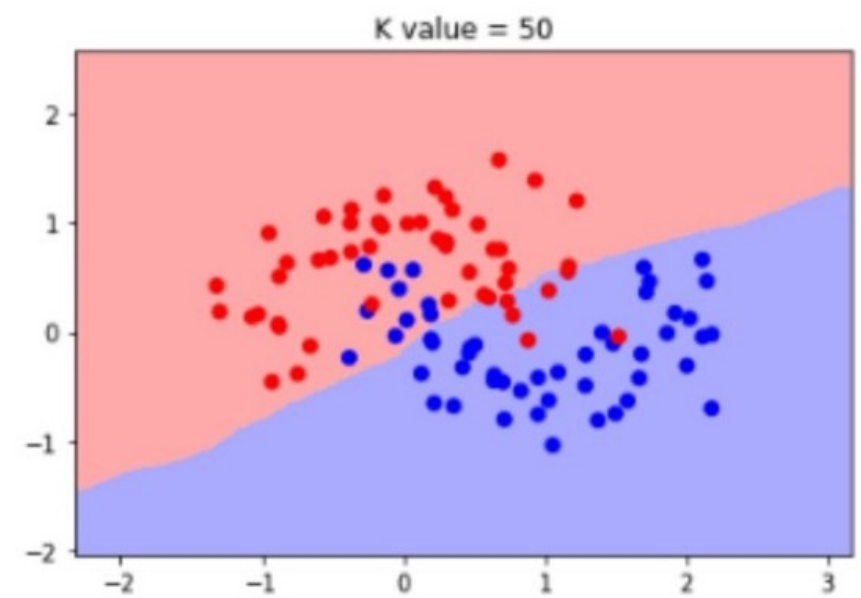
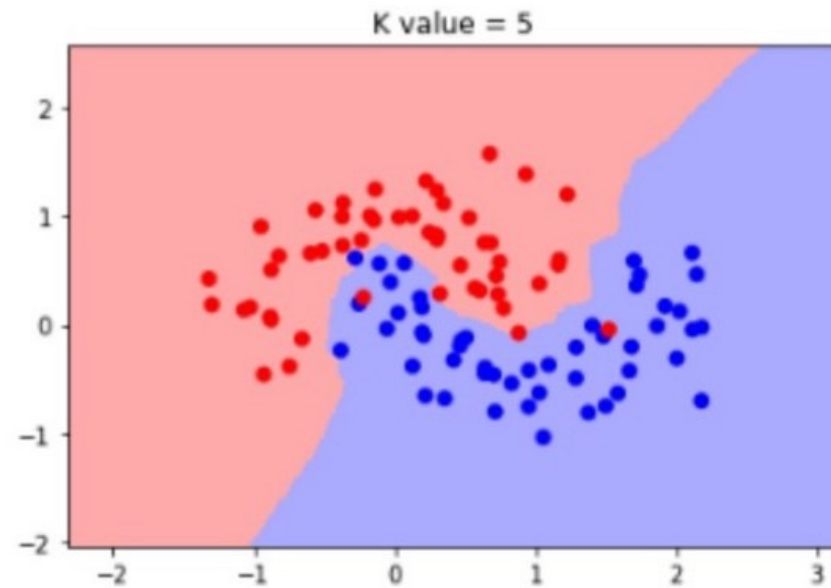
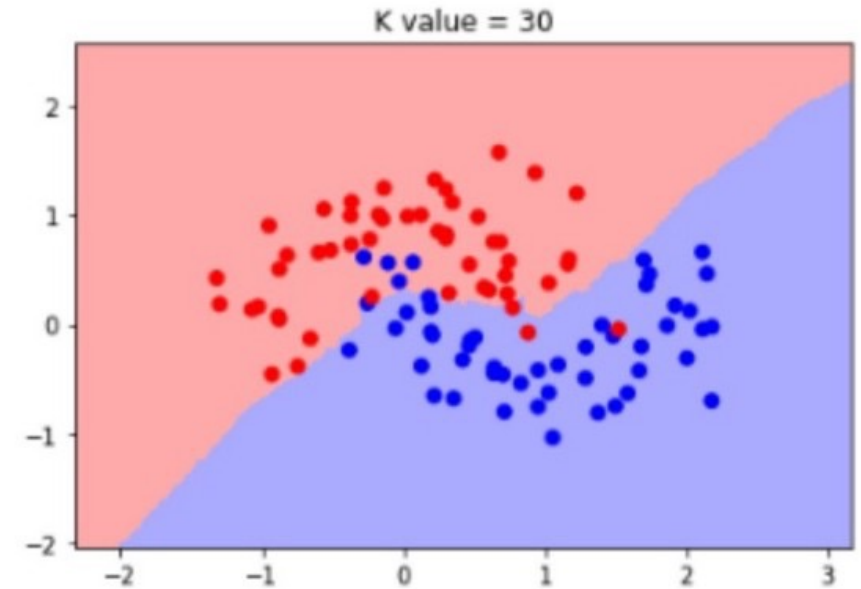
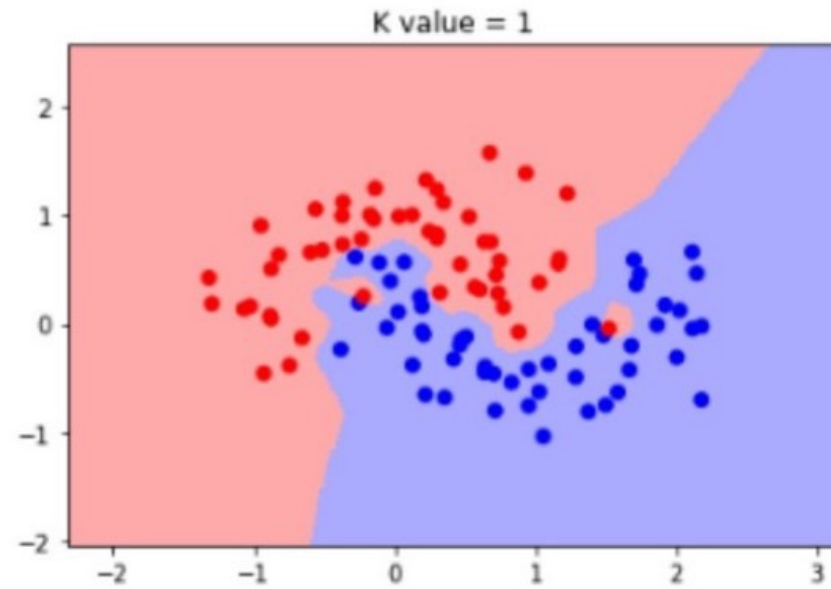
Алгоритм k Nearest Neighbors #3

- ▶ Алгоритм относит объект **u** к тому классу, представителей которого окажется больше всего среди **k** его ближайших соседей:

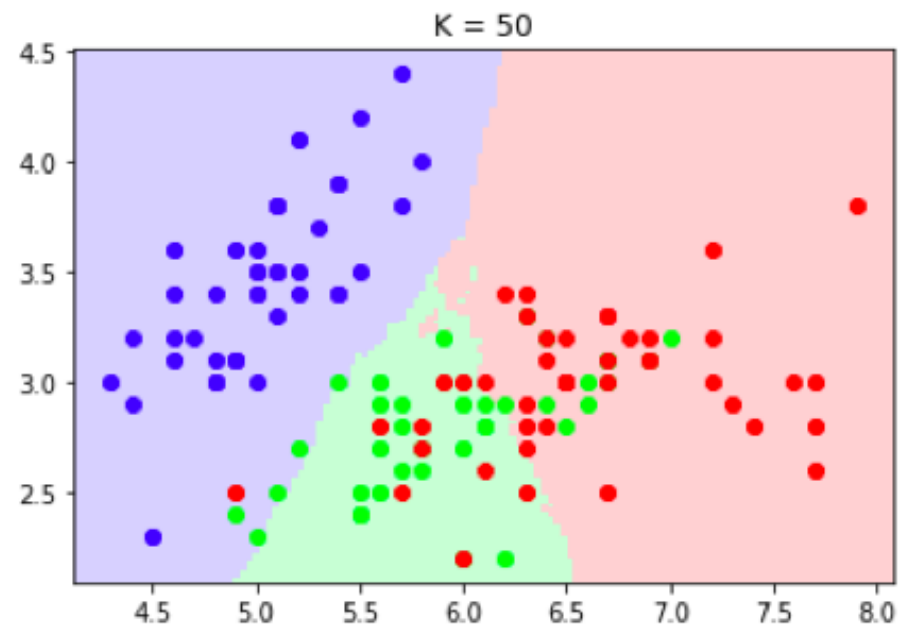
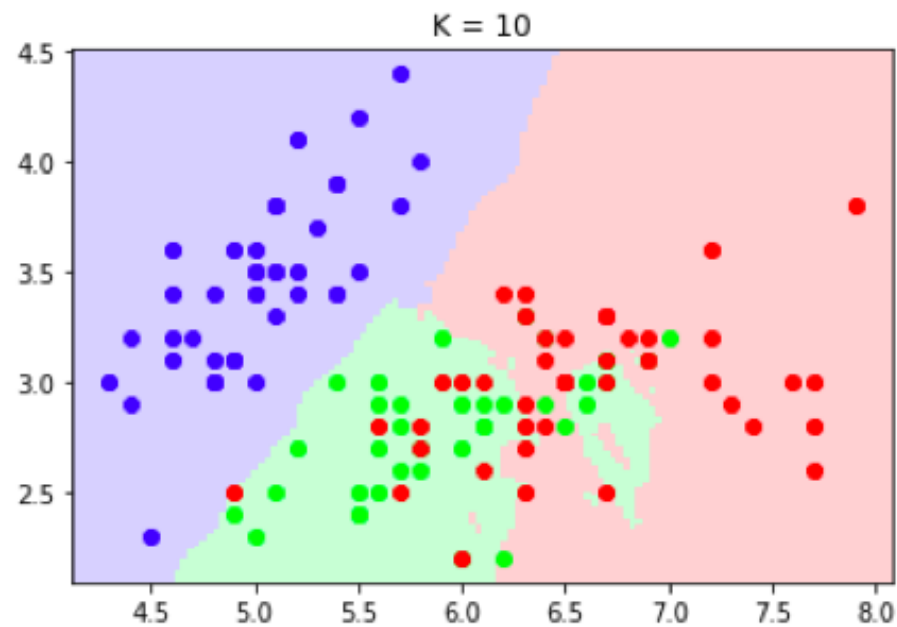
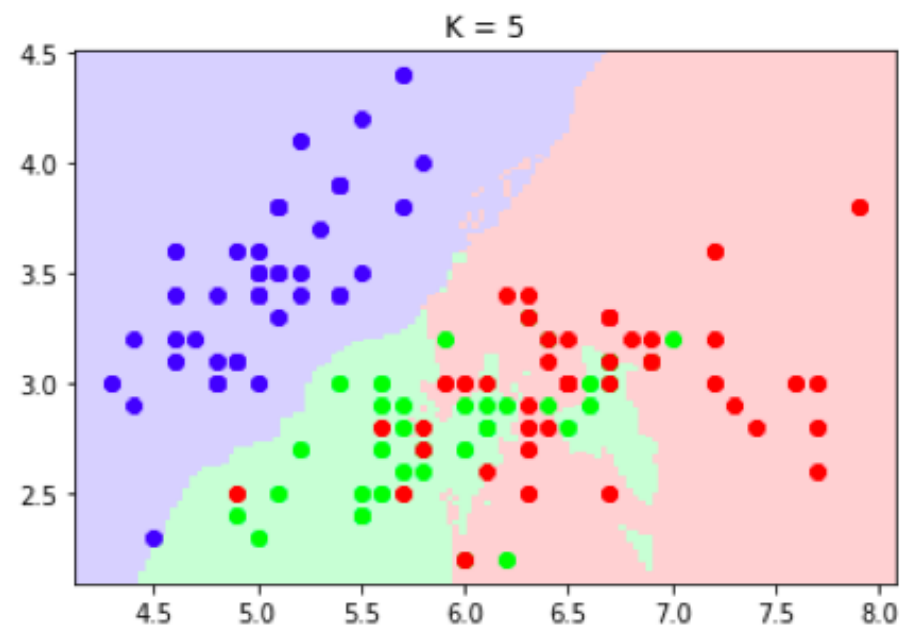
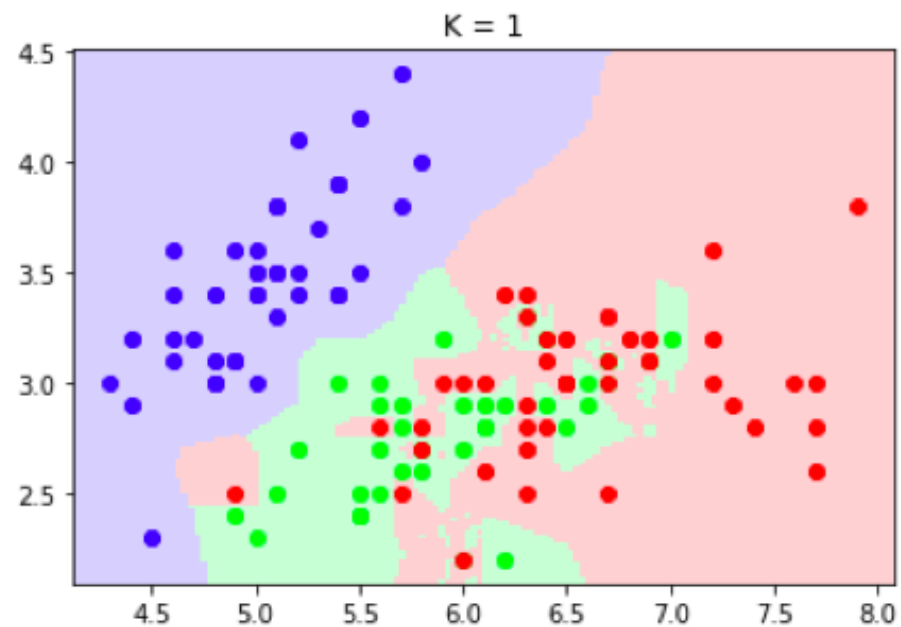
$$\hat{y}(u) = \arg \max_c \sum_{j=1}^k [y_u^{(j)} = c]$$

- $c = \{0, 1, 2, \dots, m\}$ – метка класса
- $y_u^{(j)}$ – метка j -го соседа объекта **u**.

Пример



Пример



Вопросы

- ▶ Почему все соседи вносят одинаковый вклад в прогноз?
- ▶ Как сделать вклад более близких соседей весомее?

Модификация

- ▶ Алгоритм относит объект **u** к тому классу, представителей которого окажется больше всего среди **k** его ближайших соседей:

$$\hat{y}(u) = \arg \max_c \sum_{j=1}^k w_j [y_u^{(j)} = c]$$

- $c = \{0, 1, 2, \dots, m\}$ – метка класса,
- $y_u^{(j)}$ – метка j -го соседа объекта **u**,
- w_j – некоторый вес соседа.

Примеры весов

- ▶ Чем ближе сосед, тем больше вклад:

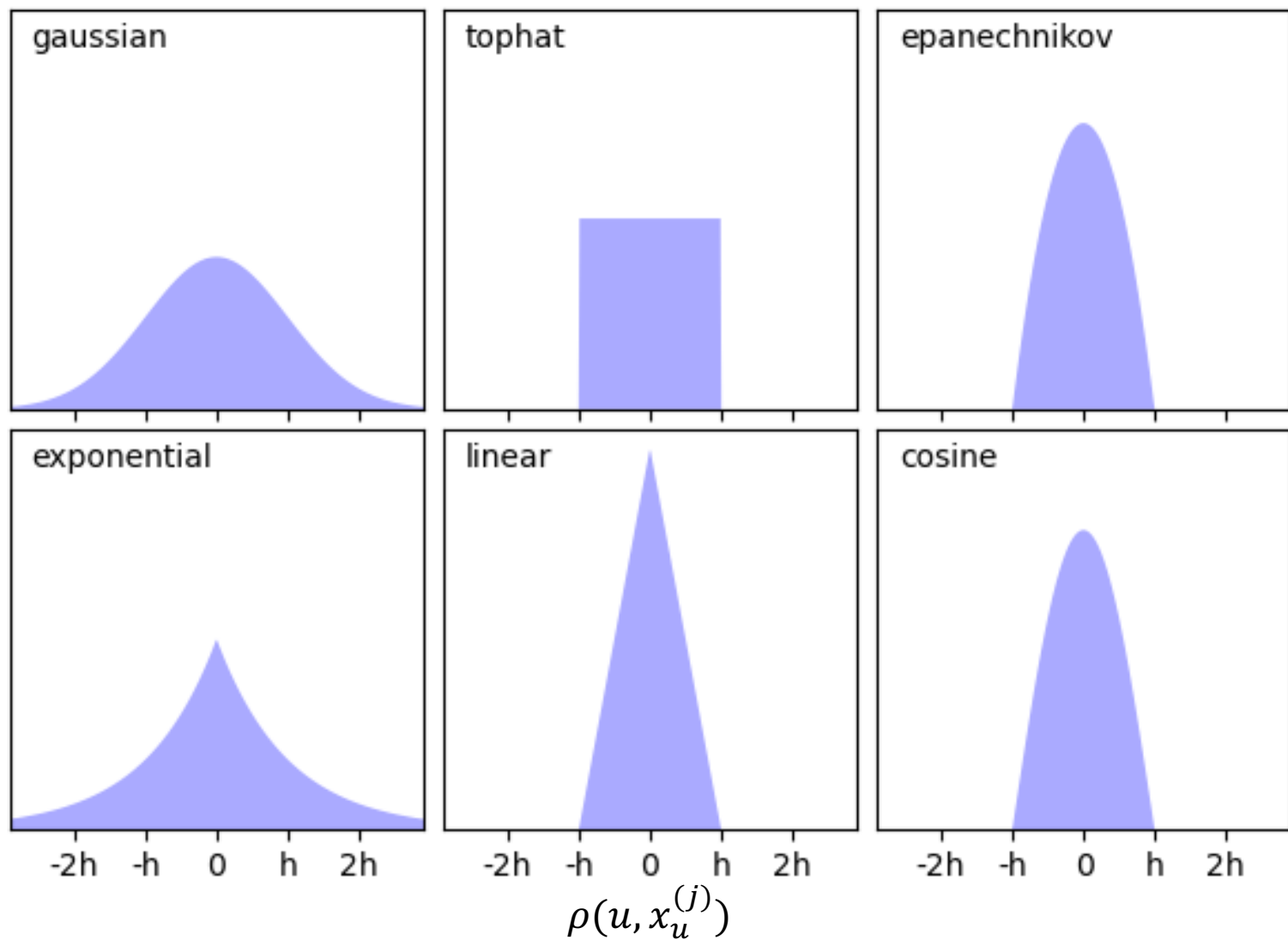
$$w_j = \frac{1}{\rho(u, x_u^{(j)})}$$

- ▶ Метод парзеновского окна:

$$w_j = K\left(\frac{\rho(u, x_u^{(j)})}{h}\right)$$

- h - ширина окна (гиперпараметр)
- K - ядро (некоторая функция близости двух объектов)

Примеры ядер



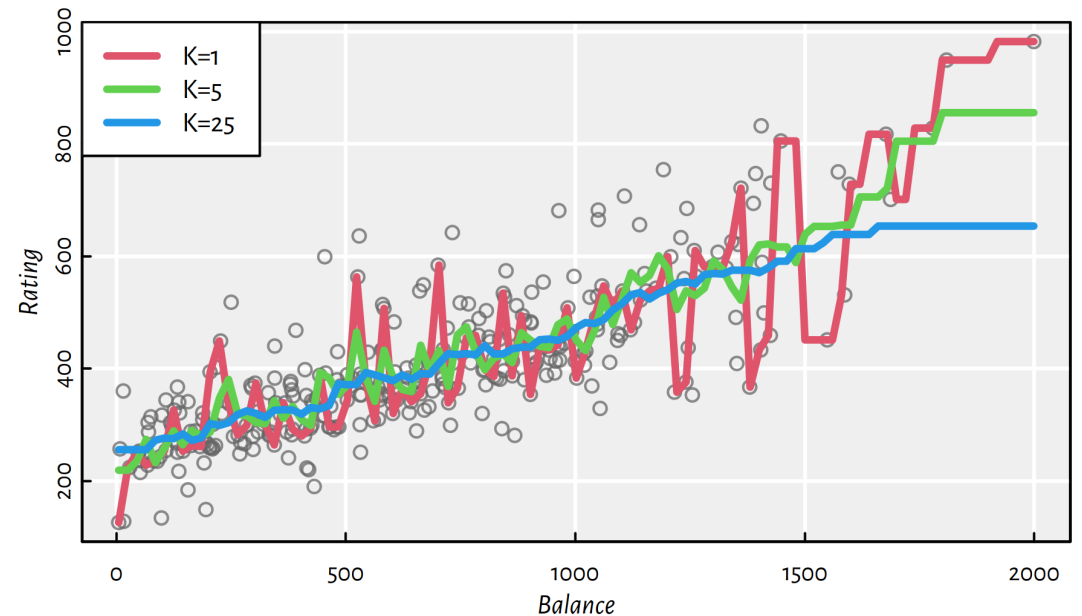
- Gaussian kernel (kernel = 'gaussian')
 $K(x; h) \propto \exp(-\frac{x^2}{2h^2})$
- Tophat kernel (kernel = 'tophat')
 $K(x; h) \propto 1$ if $x < h$
- Epanechnikov kernel (kernel = 'epanechnikov')
 $K(x; h) \propto 1 - \frac{x^2}{h^2}$
- Exponential kernel (kernel = 'exponential')
 $K(x; h) \propto \exp(-x/h)$
- Linear kernel (kernel = 'linear')
 $K(x; h) \propto 1 - x/h$ if $x < h$
- Cosine kernel (kernel = 'cosine')
 $K(x; h) \propto \cos(\frac{\pi x}{2h})$ if $x < h$

kNN для регрессии



Задача

- ▶ Есть объекты (X)
- ▶ Нужно предсказать некоторую величину (y)
- ▶ Функция, которая описывает зависимость y от X - **модель регрессии**



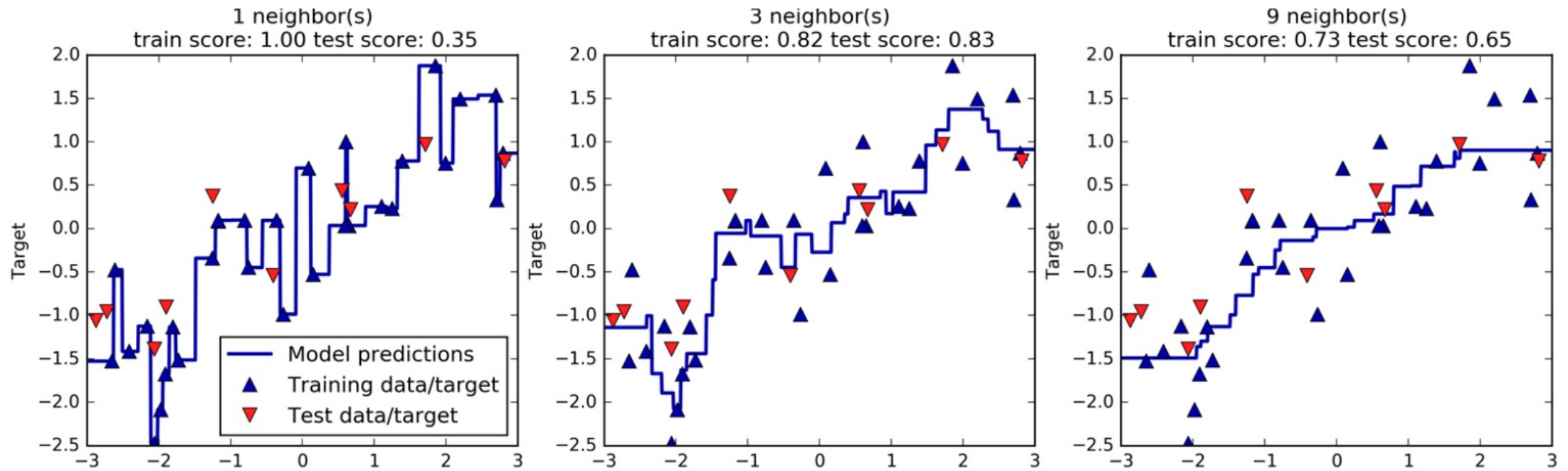
Алгоритм

- ▶ Для нового объекта **u** находим такое число **c**, что:

$$\hat{y}(u) = \arg \min_{c \in R} \sum_{j=1}^k \mathbf{w}_j \left(y_u^{(j)} - c \right)^2$$

- c – 'усредненное' значение целевой переменной по соседям,
- $y_u^{(j)}$ – значение целевой переменной j -го соседа объекта **u**,
- \mathbf{w}_j – некоторый вес соседа.

Пример



Функции расстояния

- ▶ Метрика Минковского:

$$\rho(a, b) = \left(\sum_{i=1}^d |a_i - b_i|^p \right)^{\frac{1}{p}}$$

- ▶ $p = 2$ – Евклидова метрика,
- ▶ $p = 1$ – Манхэттенское расстояние,
- ▶ $p = \infty$ – метрика Чебышева (наибольшее по координатное расстояние),

Функции расстояния

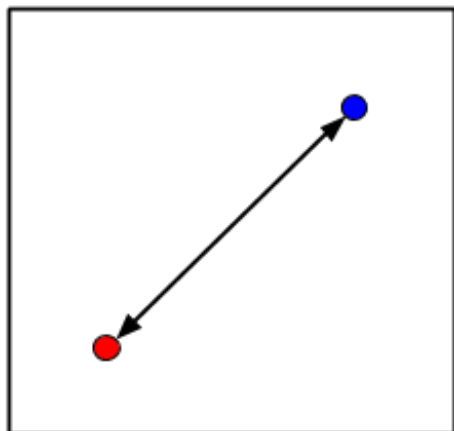
- ▶ Косинусное расстояние:

$$\rho(a, b) = \arccos \frac{a^T b}{\|a\| \|b\|}$$

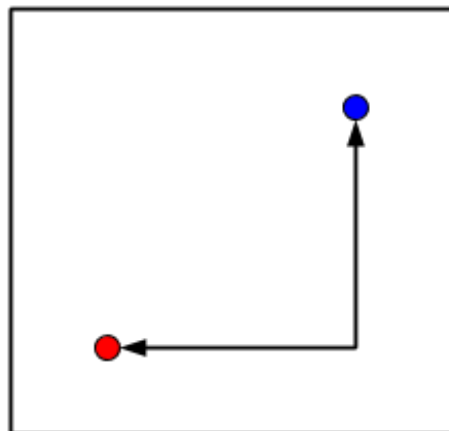
- ▶ Угол между векторами a и b .

Функции расстояния

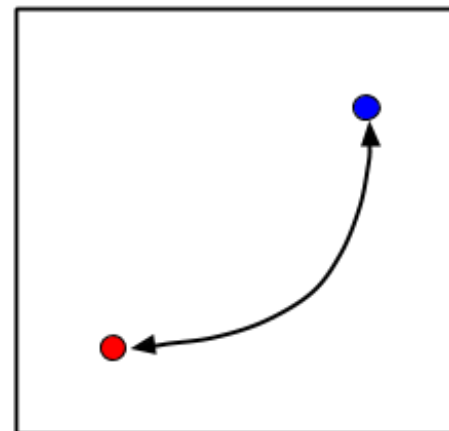
Euclidean



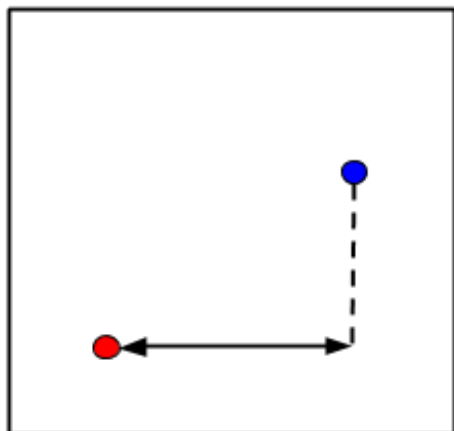
Manhattan



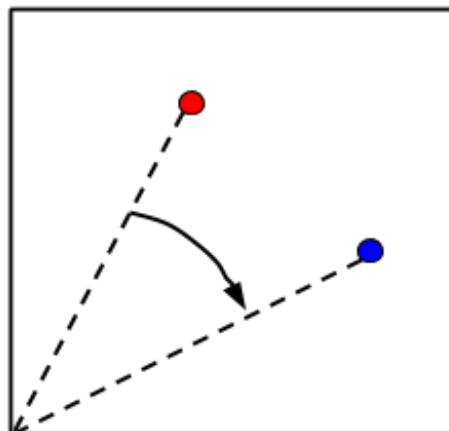
Minkowski



Chebychev



Cosine Similarity



Hamming



Основные проблемы kNN

Все проблемы kNN из-за функций расстояния:

$$\rho^2(a, b) = (a_1 - b_1)^2 + (a_2 - b_2)^2 + \dots + (a_d - b_d)^2$$

- ▶ Все признаки должны быть одного масштаба
- ▶ Шумовые признаки снижают качество алгоритма
- ▶ Плохо работает на больших размерностях данных

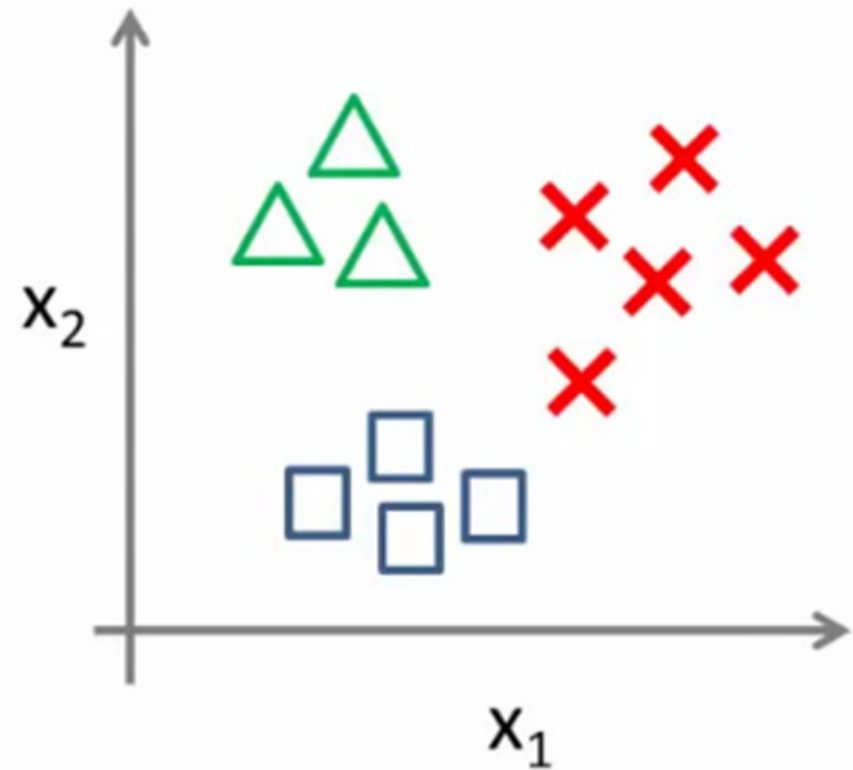
Многоклассовая классификация



Задача

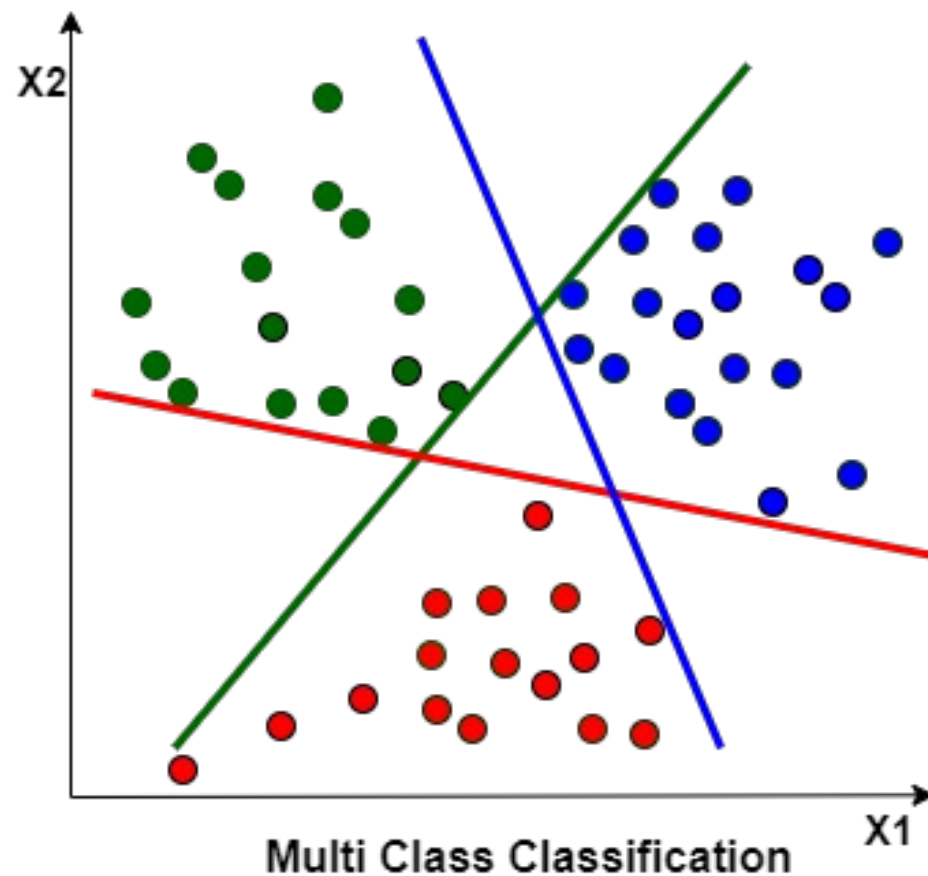
- ▶ Разделить объекты между **несколькими** классами.
- ▶ Многие классификаторы поддерживают несколько классов.
- ▶ Но не все ☹️
- ▶ Как **разложить** эту задачу на несколько задач бинарной классификации?

Multi-class classification:



Один против всех (one-vs-all)

- ▶ Пусть дано **K** классов
- ▶ Для **каждого** класса обучаем свой бинарный классификатор **отделять объекты этого класса от всех остальных**
- ▶ Всего обучаем **K** таких классификаторов



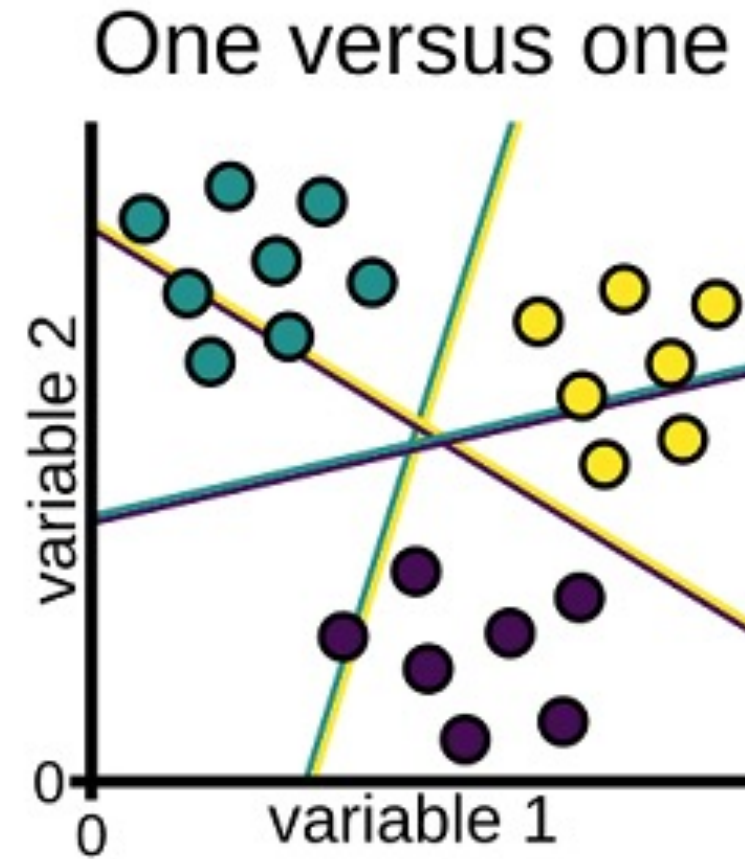
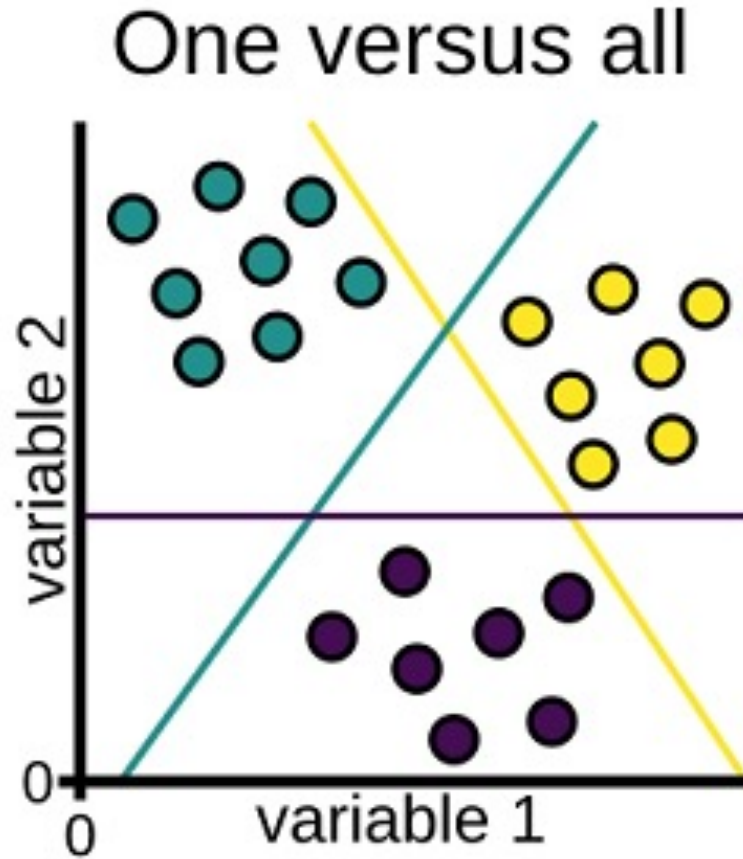
Один против всех (one-vs-all)

- ▶ Пусть есть K обученных классификаторов: $\hat{p}_1(x), \hat{p}_2(x), \dots, \hat{p}_K(x)$
- ▶ Пусть дан объект x_i , для которого делаем прогноз:
 - Получаем K прогнозов: $\hat{p}_1(x_i), \hat{p}_2(x_i), \dots, \hat{p}_K(x_i)$
 - Находим класс с максимальным прогнозом:

$$\hat{y}(x_i) = \arg \max_{k \in \{1, \dots, K\}} \hat{p}_k(x_i)$$

- Здесь $\hat{p}_k(x_i)$ – прогноз “вероятности” положительного (k -го) класса;
- $\hat{y}(x_i)$ - итоговый прогноз метки класса (одного из K)

Один против одного (one-vs-one)



Один против одного (one-vs-one)

- ▶ Для каждой пары классов i, j обучаем свой бинарный классификатор $\hat{y}_{ij}(x)$
- ▶ Пусть дан объект x_t , для которого делаем прогноз:

$$\hat{y}(x_t) = \arg \max_{k \in \{1, \dots, K\}} \sum_{i=1}^K \sum_{j \neq i} [\hat{y}_{ij}(x_t) = k]$$

- ▶ Т.е. выбираем класс, за который наберется больше всего голосов

Логистическая регрессия на K классов



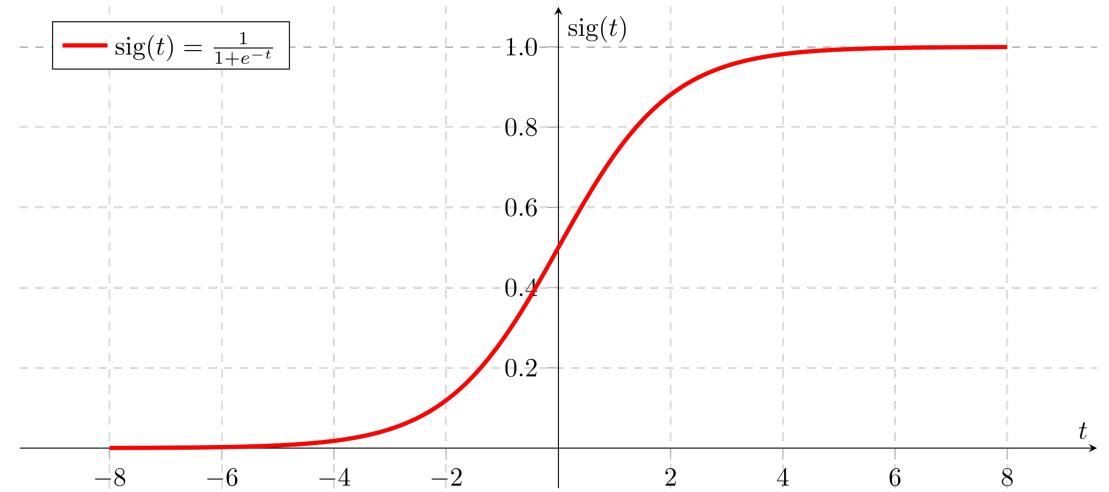
Логистическая регрессия на 2 класса

- ▶ Вероятность класса 1:

$$p(y = \mathbf{1}|x_i) = \sigma(x_i^T w) = \hat{y}_i$$

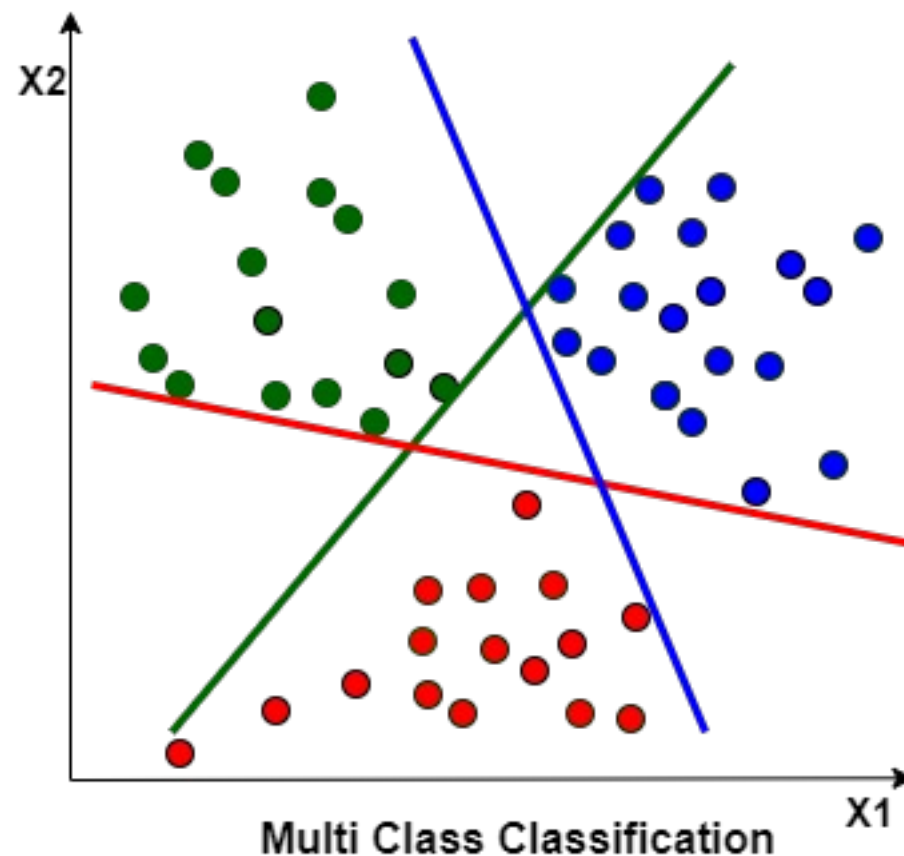
- ▶ Вероятность класса 0:

$$p(y = \mathbf{0}|x_i) = 1 - \sigma(x_i^T w)$$



Логистическая регрессия на K классов

- ▶ Строим **K** **один против всех** моделей:
 - Класс 1 против всех: $z_{i1} = x_i^T w_1$
 - Класс 2 против всех: $z_{i2} = x_i^T w_2$
 - Класс 3 против всех: $z_{i3} = x_i^T w_3$
 - Класс K против всех: $z_{iK} = x_i^T w_K$
- ▶ Получаем **K** векторов весов для обучения



Логистическая регрессия на K классов

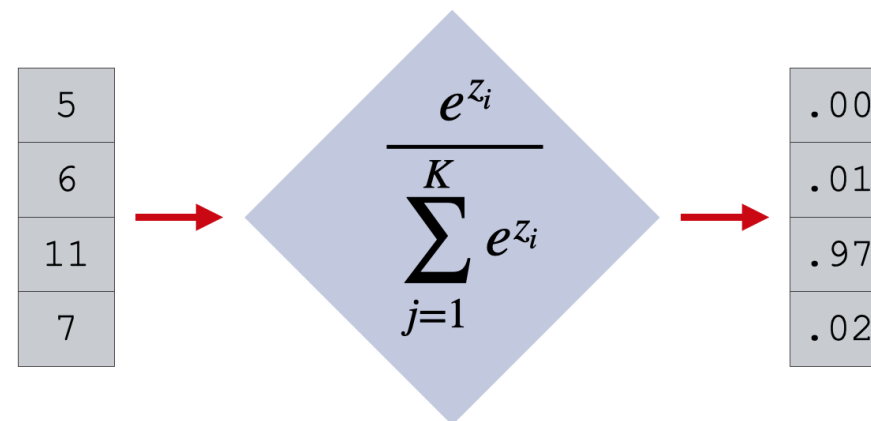
- SoftMax – многомерный вариант

СИГМОИДЫ:

$$\hat{y}_{i1} = p(y = \mathbf{1} | x_i) = \frac{e^{z_{i1}}}{\sum_{k=1}^K e^{z_{ik}}}$$

$$\hat{y}_{i2} = p(y = \mathbf{2} | x_i) = \frac{e^{z_{i2}}}{\sum_{k=1}^K e^{z_{ik}}}$$

$$\hat{y}_{iK} = p(y = \mathbf{K} | x_i) = \frac{e^{z_{iK}}}{\sum_{k=1}^K e^{z_{ik}}}$$



Логарифм правдоподобия для K классов

- ▶ Правдоподобие:

$$\text{Likelihood} = - \prod_{i=1}^n p(y = 1|x_i)^{[y_i=1]} p(y = 2|x_i)^{[y_i=2]} \dots p(y = K|x_i)^{[y_i=k]}$$

- ▶ Логарифм правдоподобия (функция потерь):

$$L = - \sum_{i=1}^n \sum_{k=1}^K [y_i = k] \log(\hat{y}_{ik}) \rightarrow \min_{w_1, \dots, w_K}$$

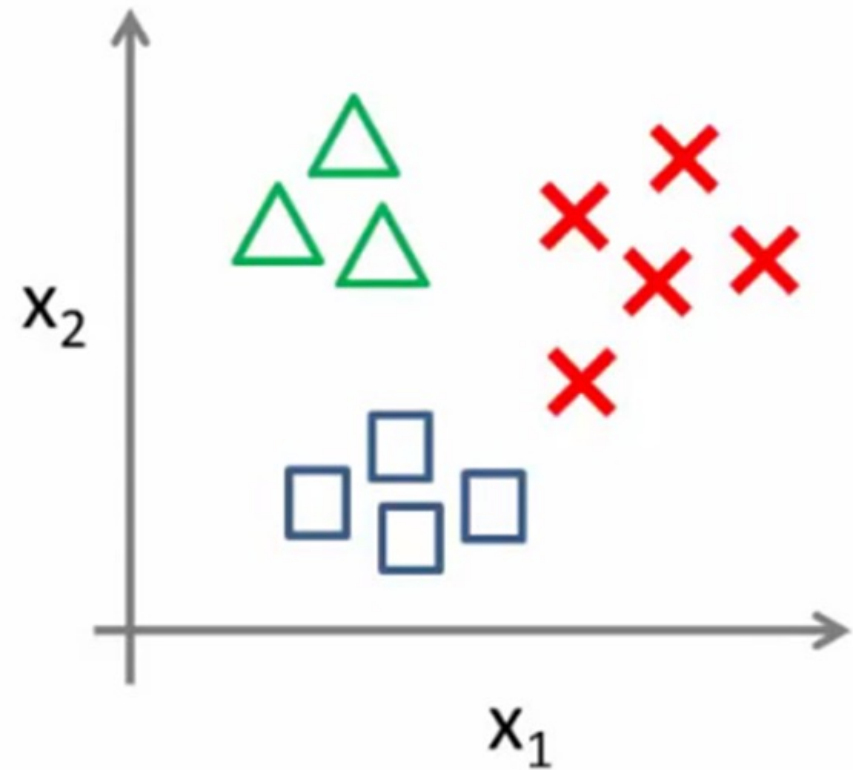
Метрики качества



Задача

- ▶ Мы знаем как считать метрики для **двух** классов.
- ▶ Что делать в случае K классов?

Multi-class classification:



Микро-усреднение

- ▶ Пусть дано K классов
- ▶ Рассмотрим K один против всех задач
- ▶ Для каждой задачи считаем TP_k, FP_k, FN_k, TN_k
- ▶ Усредняем эти характеристики по всем классам:

$$\overline{TP} = \frac{1}{K} \sum_{k=1}^K TP_k$$

- ▶ Используем их для подсчета метрик качества:

$$Precision = \frac{\overline{TP}}{\overline{TP} + \overline{FP}}$$

Макро-усреднение

- ▶ Пусть дано K классов
- ▶ Рассмотрим K один против всех задач
- ▶ Для каждой задачи считаем TP_k, FP_k, FN_k, TN_k
- ▶ Используем их для подсчета метрик качества:

$$Precision_k = \frac{TP_k}{TP_k + FP_k}$$

- ▶ Усредняем эти метрики качества по всем классам:

$$\overline{Precision} = \frac{1}{K} \sum_{k=1}^K Precision_k$$










Пересекающиеся классы



Классификация с пересекающимися классами

Multi-Class

Multi-Label

C = 3	Samples			Samples		
  						
	Labels (t)			Labels (t)		
	[0 0 1]	[1 0 0]	[0 1 0]	[1 0 1]	[0 1 0]	[1 1 1]

Независимая классификация (one-vs-all)

- ▶ Пусть есть K обученных классификаторов: $\hat{p}_1(x), \hat{p}_2(x), \dots, \hat{p}_K(x)$
- ▶ Пусть дан объект x_i , для которого делаем прогноз:
 - Получаем K прогнозов: $\hat{p}_1(x_i), \hat{p}_2(x_i), \dots, \hat{p}_K(x_i)$
 - Тогда вектор прогнозов:

$$\hat{y}(x_i) = \begin{pmatrix} [\hat{p}_1(x_i) > \tau] \\ [\hat{p}_2(x_i) > \tau] \\ \dots \\ [\hat{p}_K(x_i) > \tau] \end{pmatrix}$$

- Можно взять порог $\tau = 0.5$
- Здесь $\hat{p}_k(x_i)$ – прогноз “вероятности” положительного (k -го) класса;
- $\hat{y}(x_i)$ - итоговый прогноз метки класса (несколько из K)

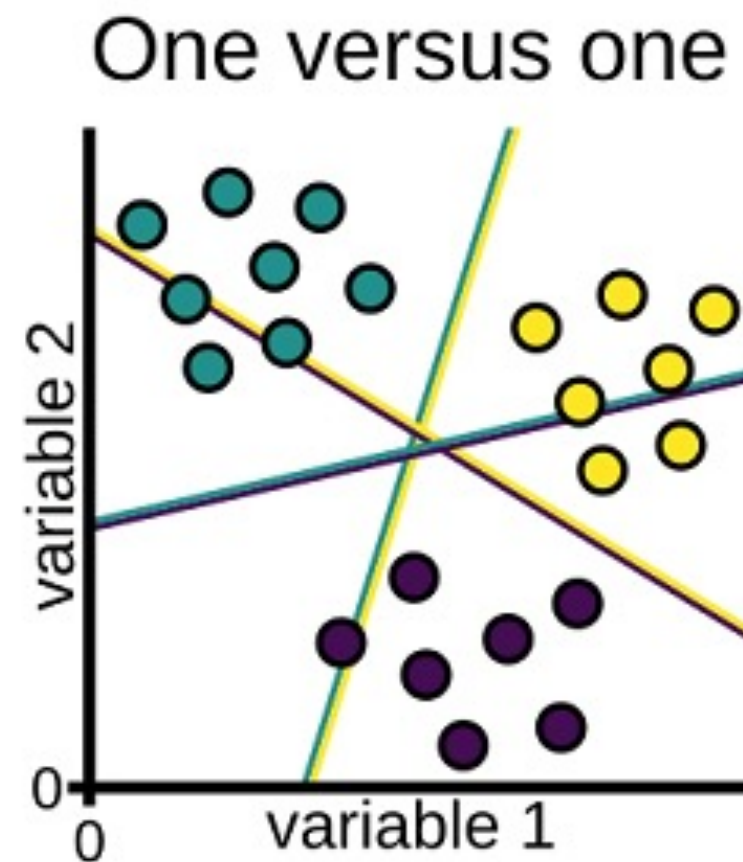
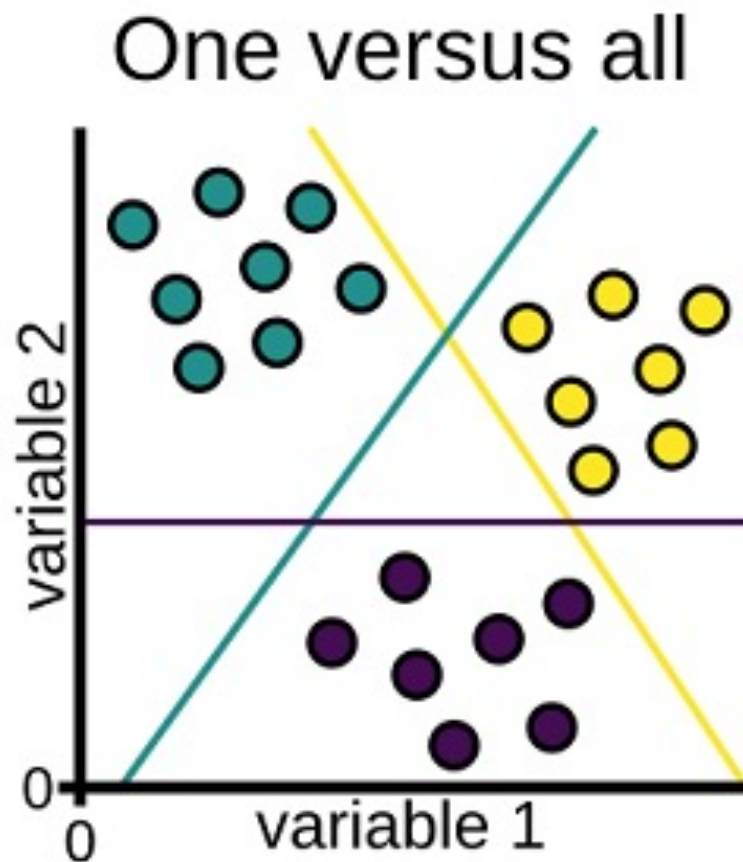
Независимая классификация

- ▶ Самое простое решение задачи **multi-label classification**
- ▶ Не учитывает связи между классами

Заключение



Резюме



Вопросы

- ▶ Опишите алгоритм k ближайших соседей для задач регрессии и классификации.
- ▶ Каковы проблемы использования метода k ближайших соседей на практике?
- ▶ Запишите формулы для следующих функций расстояния: расстояние Минковского, евклидово расстояние, манхэттэнское расстояние, косинусное расстояние.
- ▶ В чём заключается подход с независимой классификацией в задаче классификации с пересекающимися классами (multilabel classification)?
- ▶ Что такое микро- и марко-усреднение при оценивании качества многоклассовой классификации?