К-ближайших соседей

ЛЕКЦИЯ 2

Цель занятия

- Алгоритм KNN (K-Nearest Neighbors)
- Как измерять расстояния
- Как оценить качество работы модели
- Общий алгоритм построения ML моделей

Как мы обучаемся?

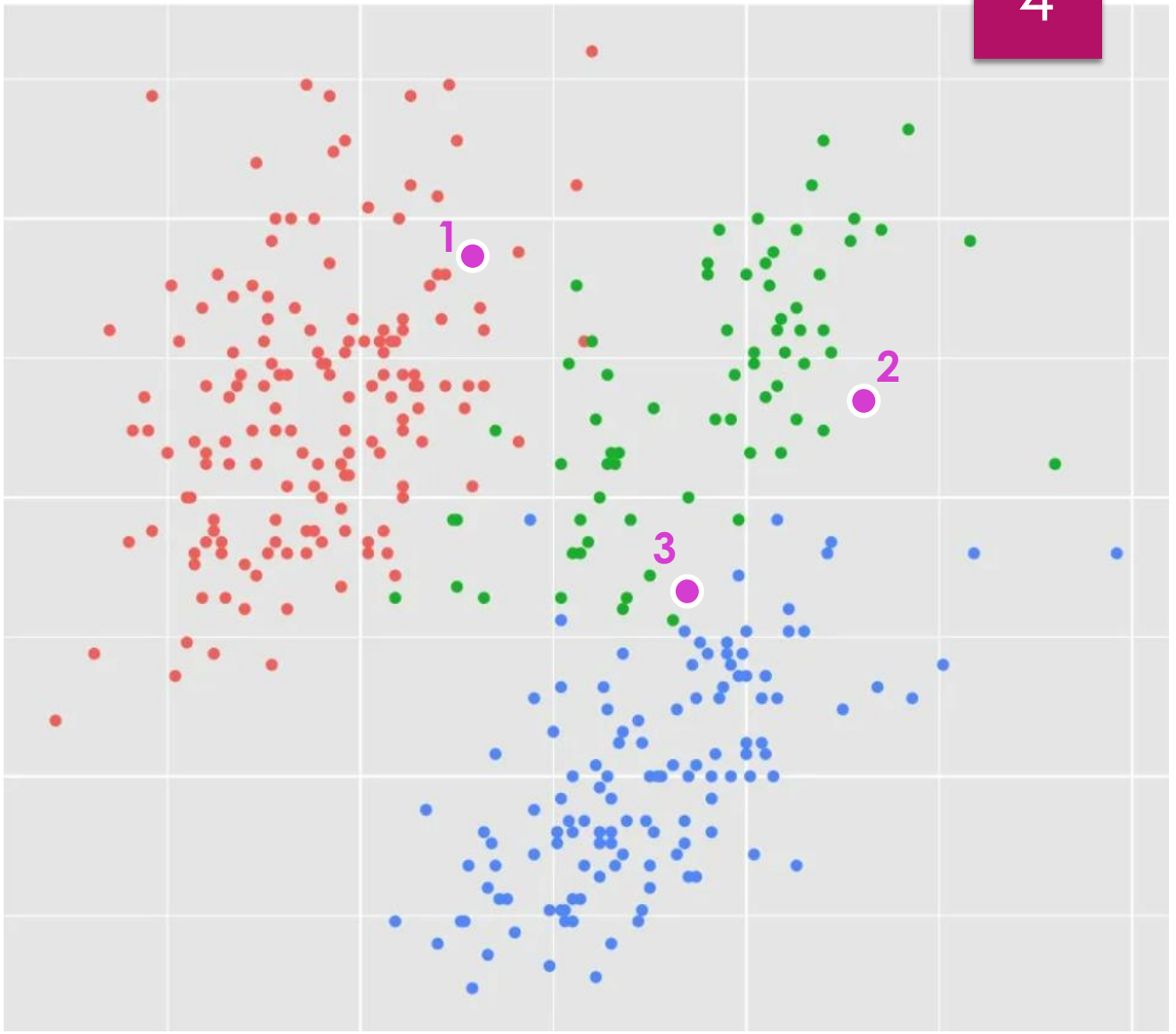
Запоминаем примеры

Сравниваем новый объект с тем, что уже знаем

Формируем ответ на основе наиболее похожих примеров

Гипотеза компактности

- Объекты, принадлежащие одному классу, будут иметь схожие признаки.



Подготовка модели

- Дано: обучающая выборка

$$\boldsymbol{X} = (\boldsymbol{x_i}, \boldsymbol{y_i})_{\boldsymbol{i}}^{\boldsymbol{l}}$$

- Решаем задачу классификации

$$\mathbb{Y} = \{\boldsymbol{1}, \boldsymbol{2}, \ldots, \boldsymbol{K}\}$$

- Как обучаем?

  Запоминаем обучающую выборку $\boldsymbol{X}$

Применение модели

- Дано: новый объект **x**
- Работа модели:

1. Сортируем объекты обучающей выборки по расстоянию до нового объекта : $\boldsymbol{\rho}\left(\boldsymbol{x}, \boldsymbol{x}_{(1)}\right) \leq \boldsymbol{\rho}\left(\boldsymbol{x}, \boldsymbol{x}_{(2)}\right) \leq \ \ldots \leq \ \boldsymbol{\rho}\left(\boldsymbol{x}, \boldsymbol{x}_{(l)}\right)$

2. Выбираем k соседей с минимальным расстоянием:

$\boldsymbol{x}_{(1)}, \boldsymbol{x}_{(2)}, \ldots, \boldsymbol{x}_{(k)}$

3. Среди k соседей выбираем наиболее встречающийся класс:

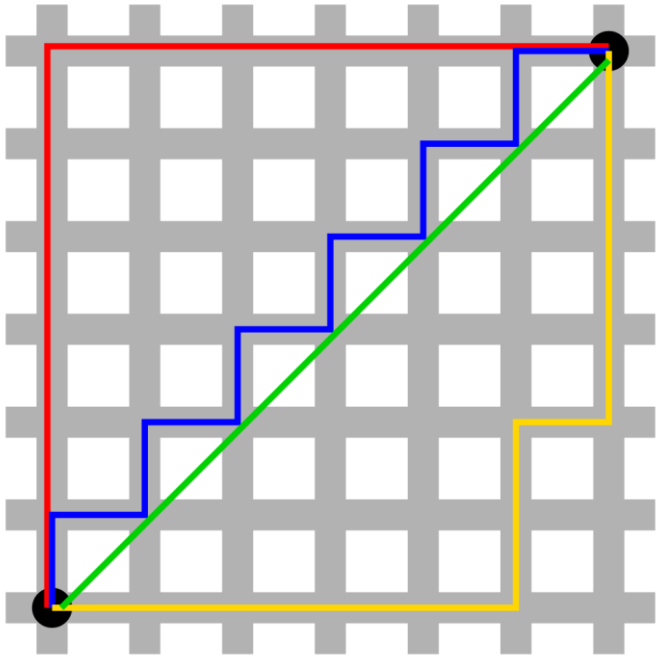
$$a(x) = argmax \sum_{i=0}^{k} \left[y_{(i)} = y\right]$$

Числовые признаки

Возраст	Пол (0-м, 1-ж)	вес	рост	Курит?	Глюкоза	Сердечно-сосудистое заболевание
27	0	81	179	1	2	1
43	1	58	168	0	0	0
32	0	78	182	0	1	0
…	…	…	…	…	…	…

Манхэттенская метрика

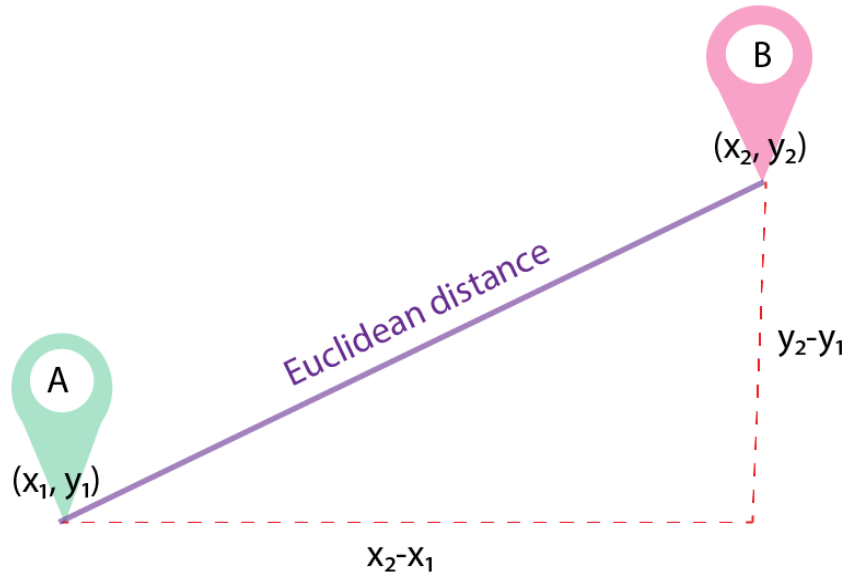
$$\rho(x, z) = \sum_{i=1}^{d} |x_i - z_i|$$



$$\rho(x_1, x_2) = |27 - 43| + |0 - 1| + |81 - 58| + |179 - 168| + |1 - 0| + |2 - 0| = 54$$

Евклидова метрика

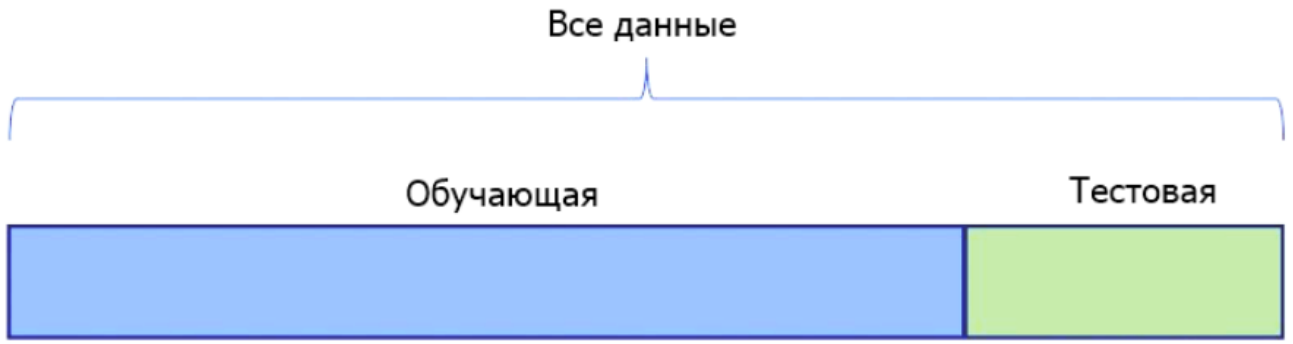
$$\rho(x, z) = \sqrt{\sum_{i=1}^{d} (x_i - z_i)^2}$$



$$\rho(x_1, x_2) = \sqrt{(27-43)^2+(0-1)^2+(81-58)^2+(179-168)^2+(1-0)^2+(2-0)^2} = 30{,}2$$

Оценка обобщающей способности

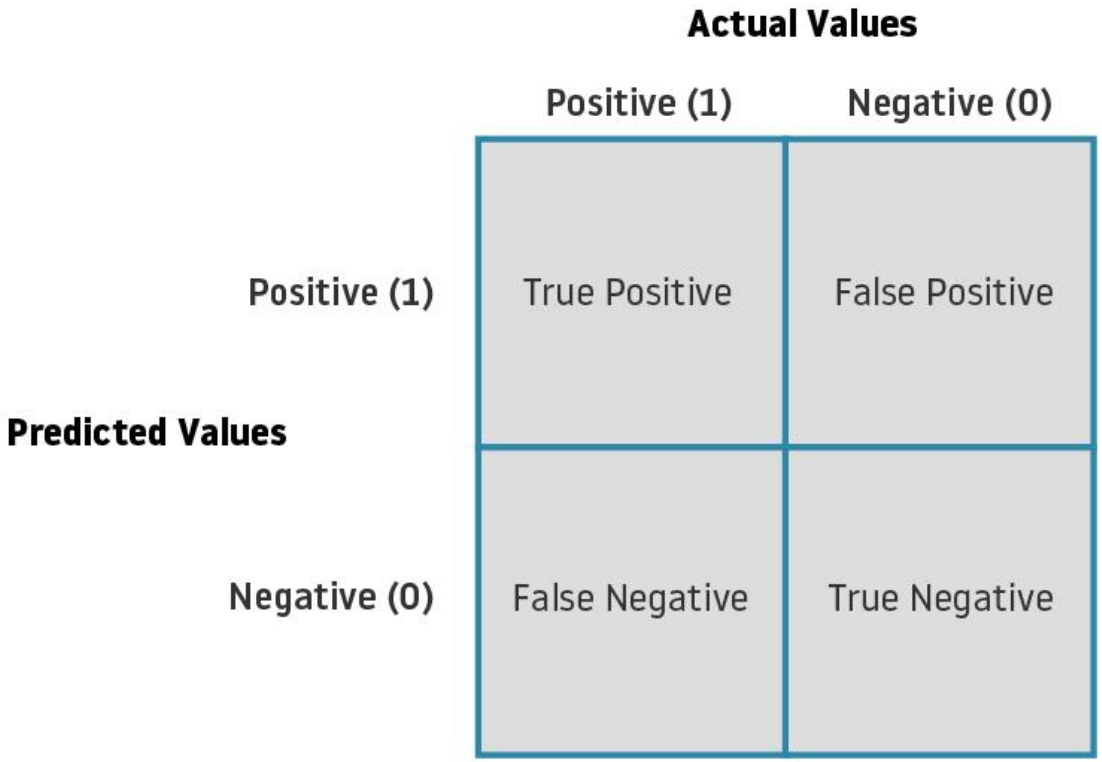
- Обучать и тестировать модель на одних и тех же данных – плохая идея.
- Нужно разбить выборку на обучающую и тестовую.



Метрики качества

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

В чём может быть проблема такой метрики?



Метрики качества

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

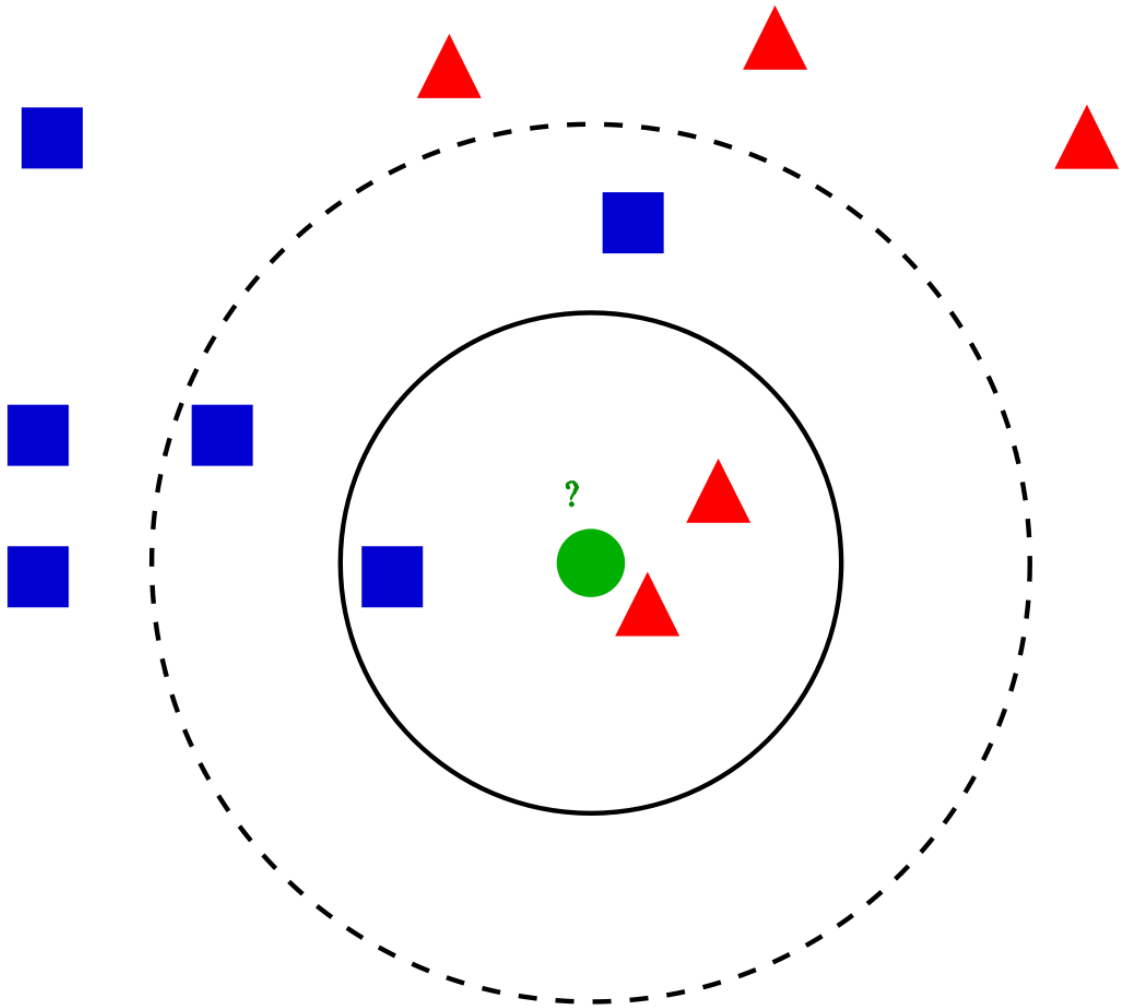
		Actual Values: Positive (1)	Actual Values: Negative (0)
Predicted Values	Positive (1)	True Positive	False Positive
	Negative (0)	False Negative	True Negative

Важно следить за сбалансированностью классов!

Допустим, мы выявляем редкое заболевание.
Всего было 1000 пациентов, у 50 из них на самом деле есть заболевание.

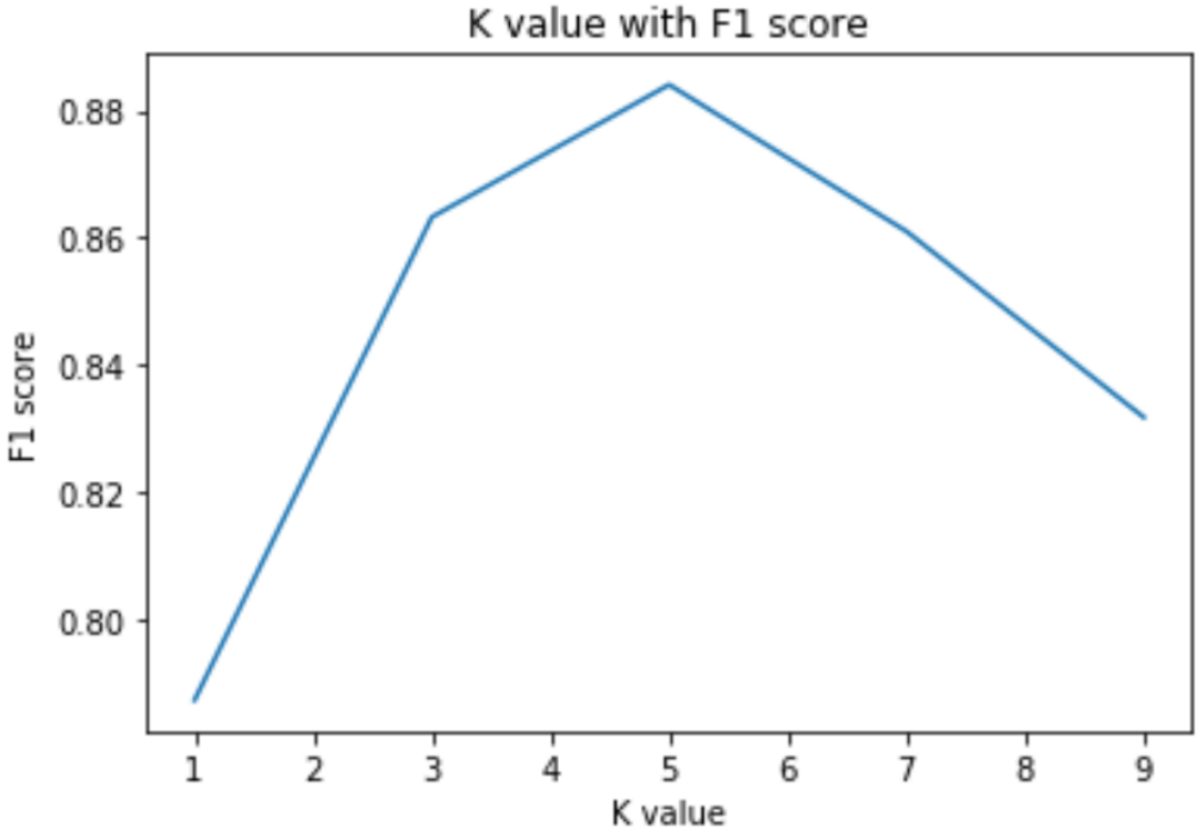
a(x)=0 – const

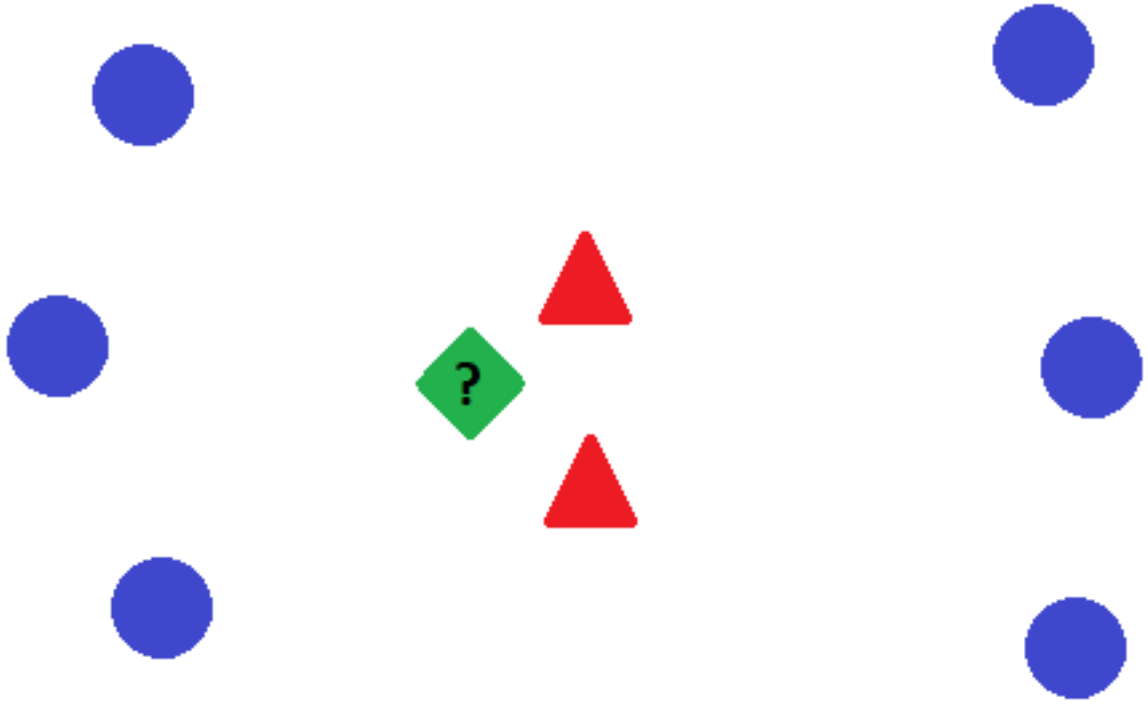
accuracy = 950/1000 = 0.95 (0.05 – доля ошибок)



Как выбрать k?

- Разделить выборку на train (80%) и test (20%)
- Обучить модель на тренировочной выборке
- Предсказать значения на тестовой выборке, посчитать метрику качества
- Выбрать K, показывающий наивысшее качество работы модели





Взвешенный KNN

$$a(x) = argmax \sum_{i=0}^{k} w_i [y_{(i)} = y]$$

Sklearn поддерживает следующие типы:

weights = “uniform”

weights = “distance”

$$w_i = \frac{1}{\rho(x, x_i)}$$

sklearn.neighbors.KNeighborsClassifier

class **sklearn.neighbors.KNeighborsClassifier**(*n_neighbors=5, *, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None*) [source]

Плюсы и минусы метода

- Очень простое обучение
- Мало гиперпараметров
- При большой выборке это может быть наилучшим вариантом

- Необходимость постоянно хранить в памяти всю обучающую выборку
- Время выполнения
- Другие методы обычно лучше