# Машинное обучение

## Лекция 12:
## Метод k ближайших соседей

Докладчик: Артем Лебедевич

# Что рассмотрим сегодня?

- Постановка задачи
- Алгоритм поиска ближайших соседей
- Переобучение
- Выбор метрики
- Взвешенный поиск соседей

# Метрические методы

Сперва пару слов про метрические методы.

- «скажи мне, кто твой друг, и я скажу, кто ты»;
- алгоритмы почти **не имеют фазы обучения**; вместо этого они запоминают всю обучающую выборку, а на этапе предсказания просто ищут объекты, похожие на целевой;
- метрические модели **являются непараметрическими**, потому что они не делают явных допущений о глобальных законах, которым подчиняются данные (например, у линейной регрессии – подчинение нормальному распределению, а у линейных классификаторов – существование линейной разделяющей плоскости);
- из прошлых пунктов ясно: алгоритм неприменим при большом количестве данных;

# Постановка задачи

# Постановка задачи

Пусть задана выборка $\mathbb{D} = (X|y)_{i=1}^{n}$, где $X \subseteq \mathbb{X} = \mathbb{R}^{n\times m}$, $y \subseteq \mathbb{Y} = \{1, \dots, L\}$, то есть

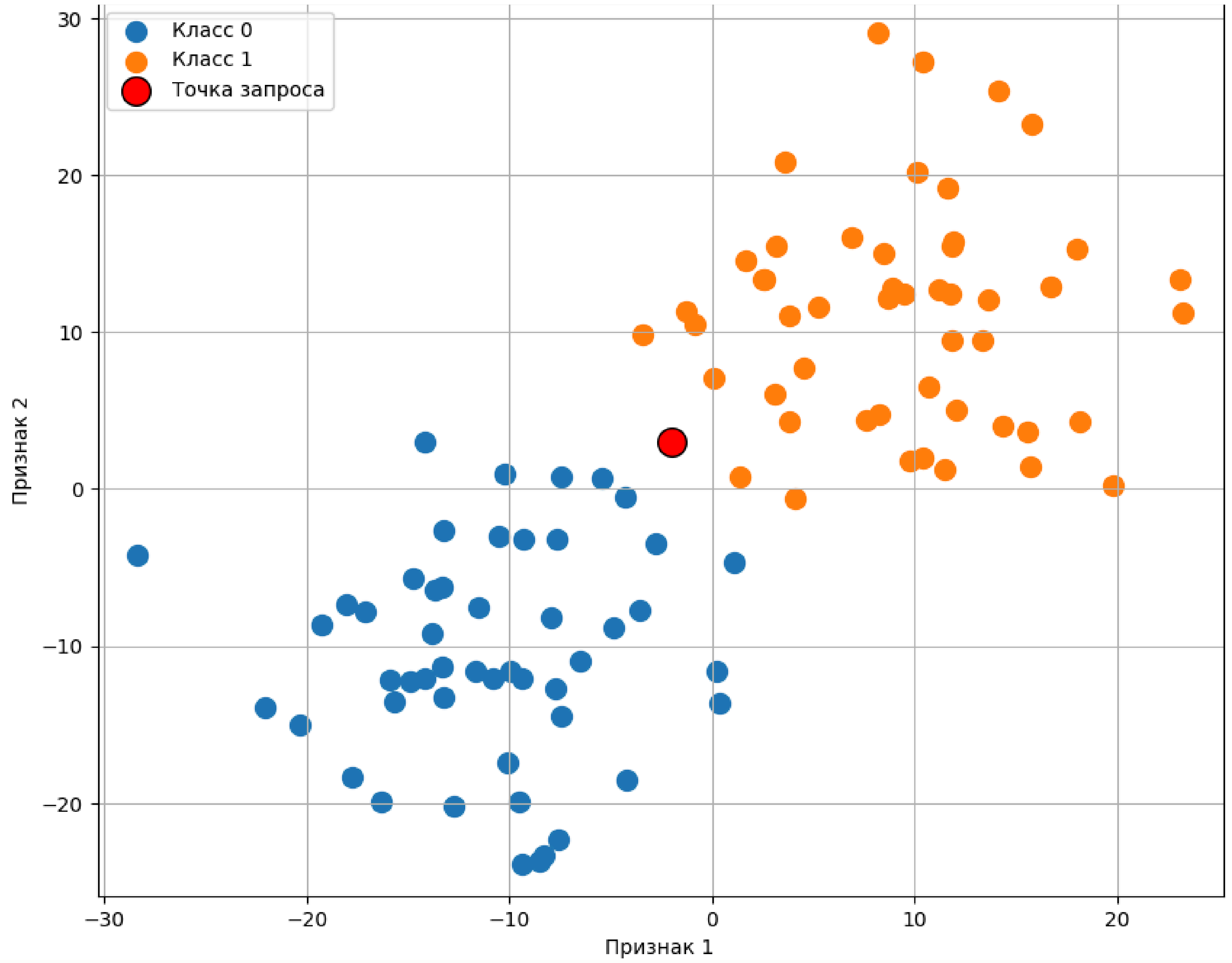
$$\mathbb{D} = \begin{pmatrix} x_{11} & \dots & x_{1m} & y_1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \dots & x_{nm} & y_n \end{pmatrix}.$$

Таким образом, решается задача классификации на $L$ классов.

Мы хотим определить, к какому классу относится рассматриваемый объект. Принцип ближайших соседей заключается в том, чтобы классифицировать целевой объект, исходя из того, какие классы у объектов, которые максимально похожи на него.

# Постановка задачи

Интуитивно степень похожести объектов в $\mathbb{R}^2$ хочется различать по расположению в пространстве и расстоянию между ними. Эту идею и использует метод $k$ ближайших соседей.



# Алгоритм поиска ближайших соседей

# Алгоритм

Пусть в пространстве объектов $\mathbb{X}$ задана метрика $\rho : \mathbb{X} \times \mathbb{X} \to [0; +\infty)$. И пусть мы хотим классифицировать некоторый объект $x' \in \mathbb{R}^m$. Для этого мы ищем $k$ наиболее близких к нему объектов $x^{(1)}, \ldots, x^{(k)}$ в смысле метрики $\rho$ в обучающей выборке $X_{\text{train}} \subset X$. То есть эти $k$ объектов должны удовлетворять следующему условию: если обозначить $X_k = \{x^{(1)}, \ldots, x^{(k)}\}$, то

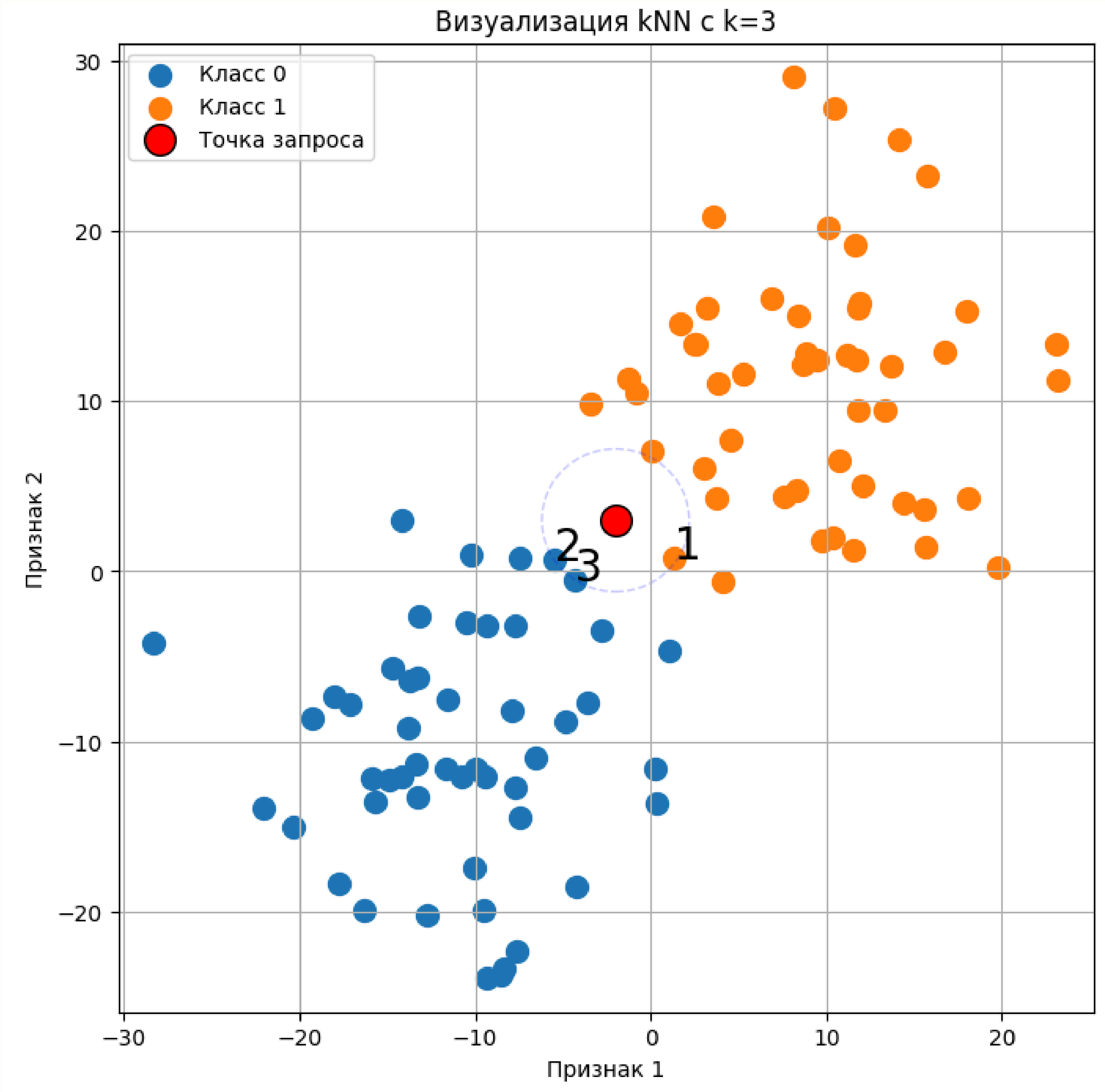
$$\forall x \in X_k, \forall \tilde{x} \in X \setminus X_k \quad \rho(x', x) \leqslant \rho(x', \tilde{x}).$$

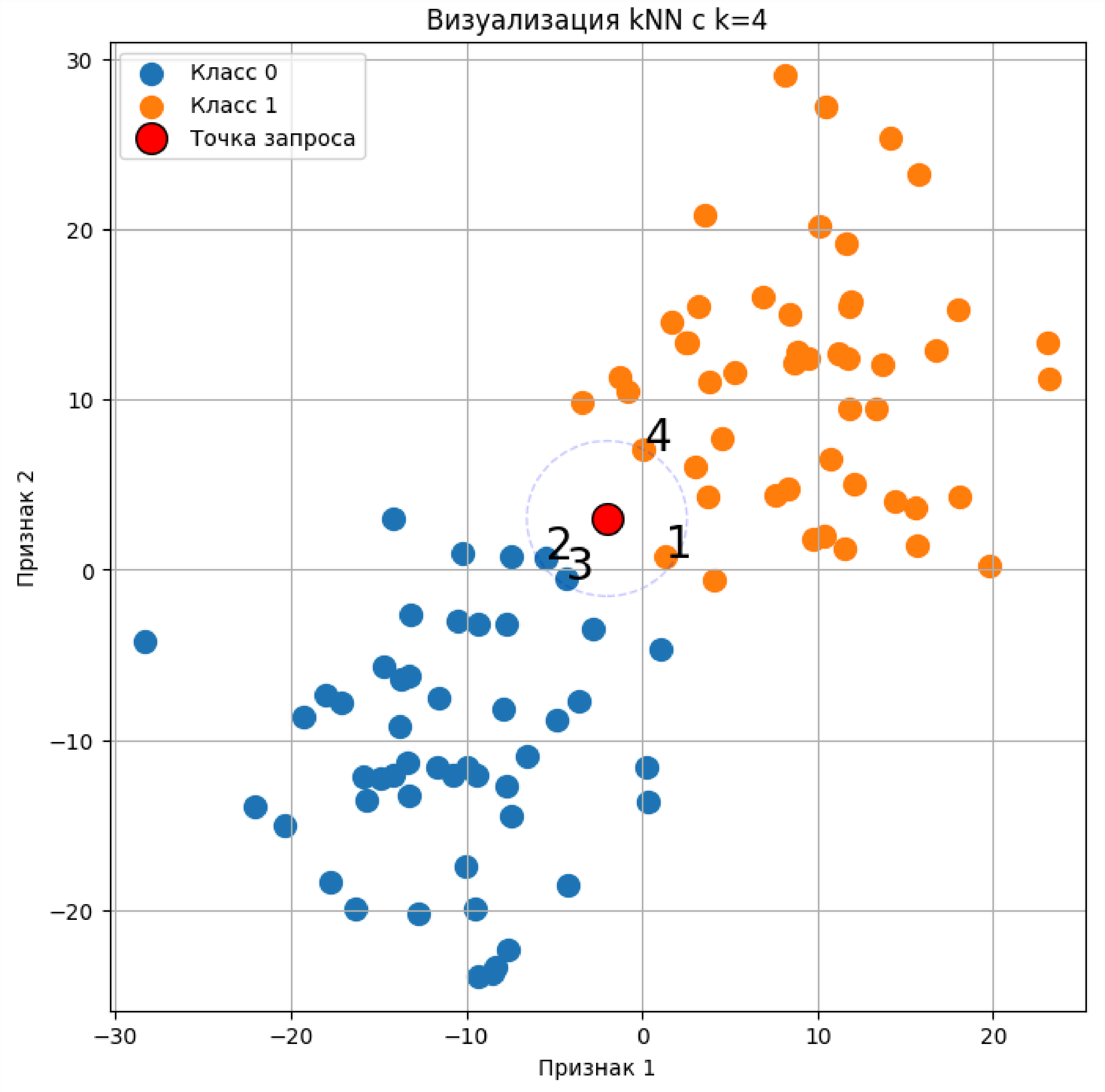
Обозначим для объектов $x^{(j)}$ истинные метки классов через $y^{(j)}$, $j = 1, \ldots, k$. Тогда класс объекта $x'$ формально определяется как

$$f(x') = \arg\max_{y \in \mathbb{Y}} \sum_{j=1}^{k} [y^{(j)} = y],$$

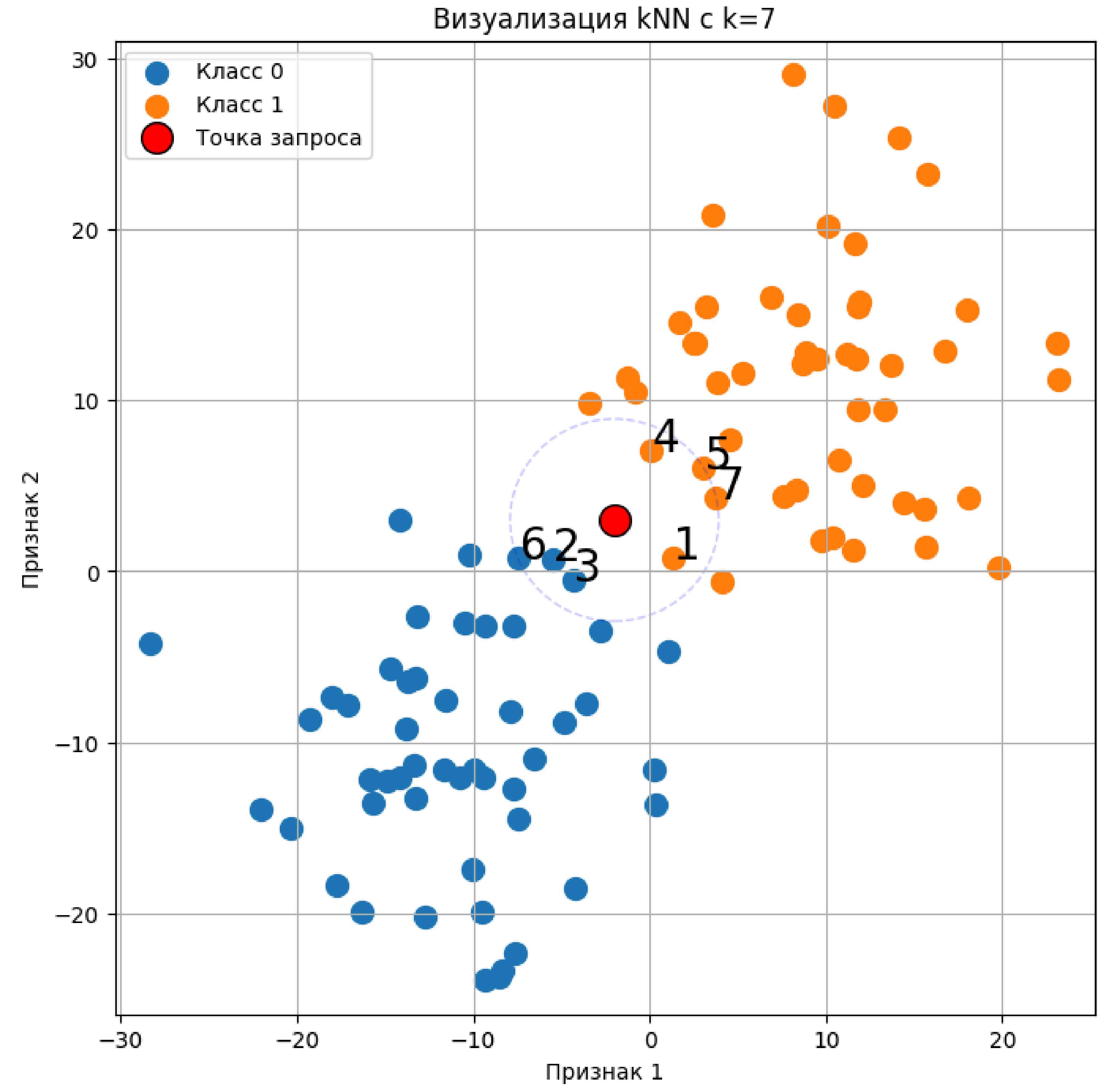
то есть для каждой метки класса $1, \ldots, L$ количество соседей для $x'$ с конкретной меткой $y$ считается путем суммирования всех индикаторов события $\{$метка соседа равна $y\}$.

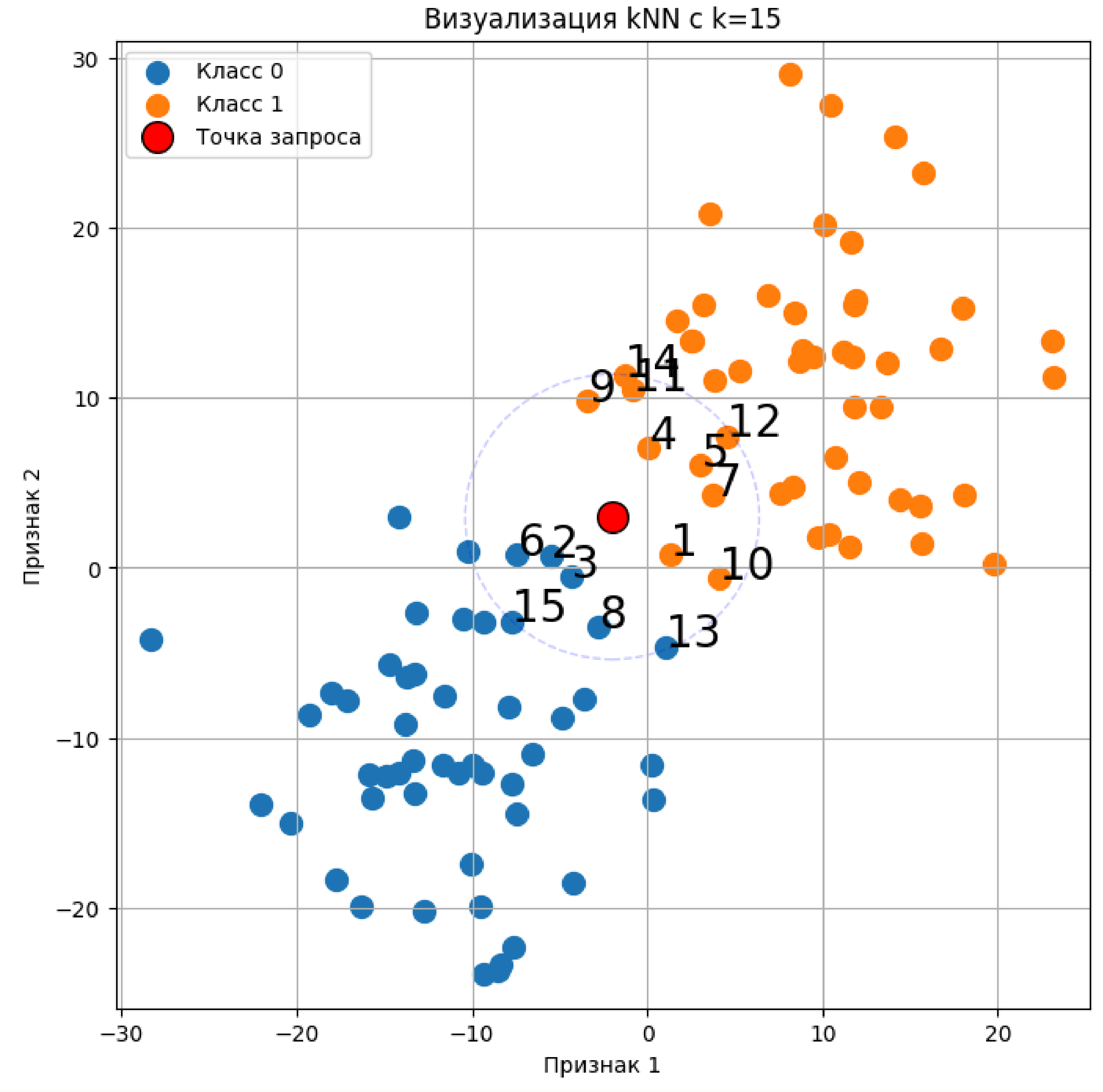
# Визуализация для 2 классов



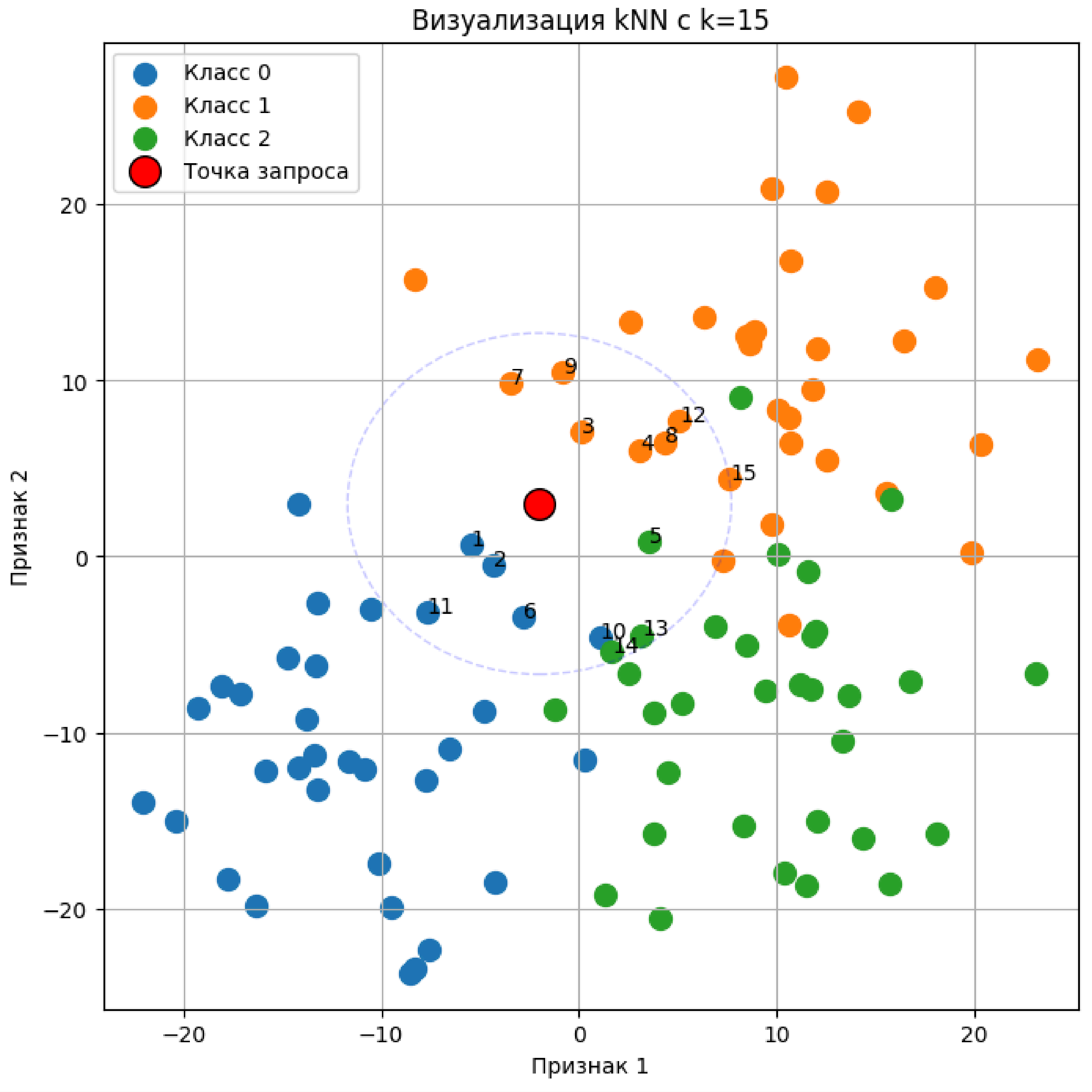



# Визуализация для 2 классов



# Визуализация для 3 классов



# Эвристика и вероятности

В качестве эвристики в свое время выяснили, что для классов можно также рассчитать вероятности классов

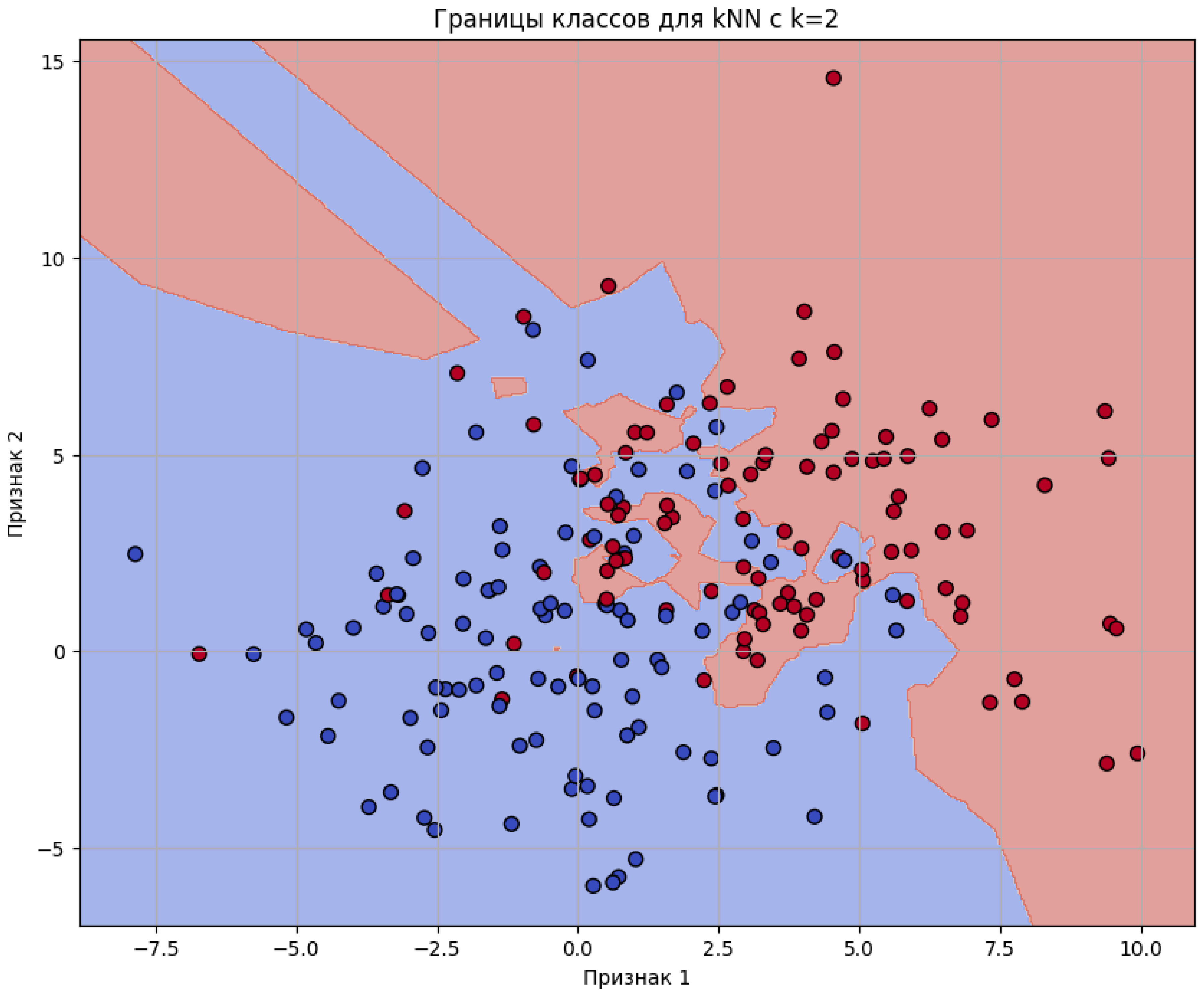
$$\mathbf{P}(y' = y) = \frac{\sum_{j=1}^{k} [y^{(j)} = y]}{k} \; \forall y \in \mathbb{Y}.$$

# Переобучение

# Проблема переобучения

Несмотря на то что формально фаза обучения отсутствует, алгоритм может легко переобучиться. Например, при задании $k = 1$ или $k = 2$ границы классов оказываются довольно сложными. Происходит это из-за того, что параметрами алгоритма можно считать всю обучающую выборку, довольно большую по размеру. Из-за этого алгоритму легко подстроиться под конкретные данные.

# Визуализация переобучения




Здесь цветные области – это то, что предсказала модель, а цвета точек – реальные метки классов.

# Выбор метрики

# Метрика - гиперпараметр

В данной модели $k$ – это гиперпараметр. Какой еще есть гиперпараметр? Вспомним, что мы не ограничивались использованием конкретной метрики $\rho$. Следовательно, ее можно задавать самостоятельно. Наиболее часто используемыми являются

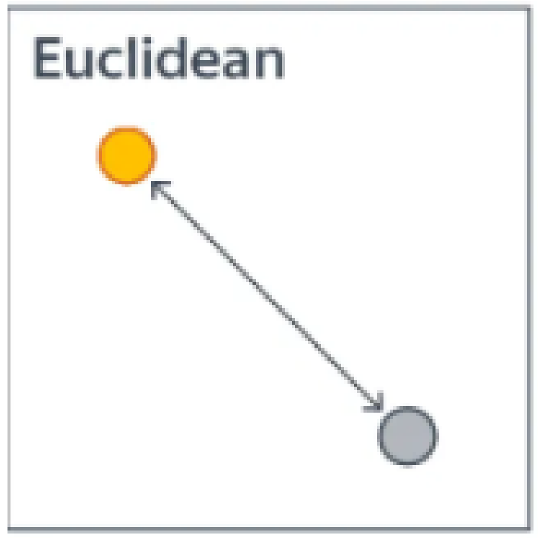
- евклидова метрика (самая частая)

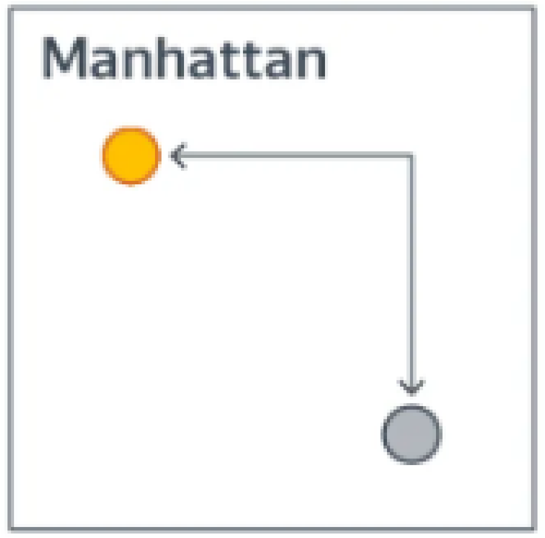
$$\rho(x, y) = \sqrt{\sum (x - y)^2};$$

- манхэттенская метрика

$$\rho(x, y) = \sum |x - y|,$$

  она часто используется в высокоразмерных пространствах из-за лучшей устойчивости к выбросам;




https://education.yandex.ru/handbook/ml/article/metricheskiye-metody

# Метрика - гиперпараметр

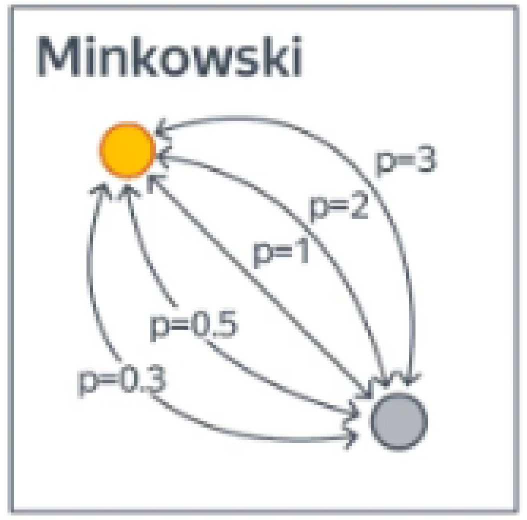
- метрика Минковского (обобщение евклидовой и манхэттенской)

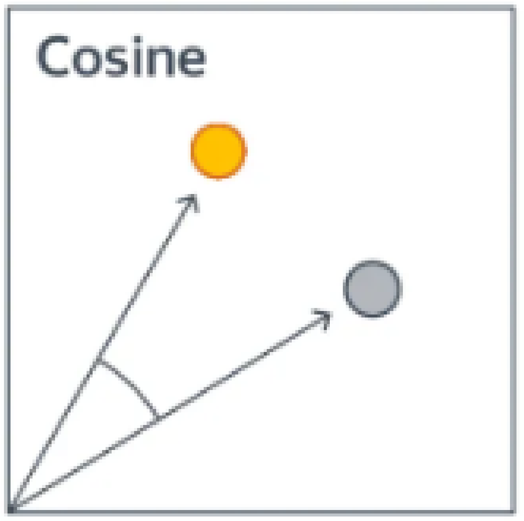
$$\rho(x, y) = \left(\sum |x - y|^p\right)^{1/p};$$

- косинусное расстояние

$$\rho(x, y) = 1 - \frac{(x, y)}{\|x\| \cdot \|y\|},$$

она применяется в задачах, связанных с текстами, поскольку там не важна норма векторного признака.




https://education.yandex.ru/handbook/ml/article/metricheskiye-metody

# Взвешенный поиск соседей

# Варианты взвешивания

Пусть мы также хотим учитывать расстояния до ближайших соседей. Есть два способа это сделать:

1. назначить индикаторам веса, которые тем больше, чем ближе объект к целевому

$$f(x') = \arg\max_{y \in \mathbb{Y}} \sum_{j=1}^{k} w_j \cdot [y^{(j)} = y],$$

где $w_i$ – это затухающий весовой коэффициент; чаще всего берутся следующие веса

$$w_j = \frac{1}{\rho(x', x^{(j)})}, \text{ или } w_j = \frac{k+1-j}{k}, \text{ или } w_j = q^j, \ 0 < q < 1;$$

2. представить веса как некоторую функцию от расстояния – ядерная функция (kernel function); с помощью функции ядра также можно решать задачи регрессии через kNN, но мы не будем рассматривать этот подход, так как он используется крайне редко.

# Плюсы и минусы kNN

Рассмотрим плюсы и минусы kNN. Плюсы:

- непараметрический, то есть не делает явных предположений о распределении данных;
- легко интерпретируемый
- достаточно точный, хотя чаще всего и уступает ансамблевым методам;

Минусы:

- неэффективный по памяти, поскольку нужно хранить всю обучающую выборку, следовательно, вычислительно дорогой;
- чувствителен к масштабу данных и неинформативным признакам;
- из-за отсутствия фазы обучения в настоящее время этот алгоритм почти не применяется.

# Применение метода

Несмотря на все минусы, у данного метода есть немало применений в прикладных задачах:

- **Рекомендательные системы**. Если посмотреть на саму формулировку задачи «предложить пользователю что-то похожее на то, что он любит», то KNN прямо напрашивается в качестве решения. Несмотря на то что сейчас часто используются более совершенные алгоритмы, метод ближайших соседей всё равно применяется в качестве хорошего бейзлайна.
- **Поиск семантически похожих документов**. Если векторные представления близки друг к другу, то темы документов схожи.
- **Поиск аномалий и выбросов**. Из-за того что алгоритм запоминает обучающую выборку полностью, ему легко посмотреть, насколько целевой объект похож на все данные, которые он видел.
- **Задача кредитного скоринга**. Рейтинги двух людей, у которых примерно одинаковая зарплата, схожие должности и кредитные истории, не должны сильно отличаться, поэтому KNN отлично подходит для решения такой задачи.

https://education.yandex.ru/handbook/ml/article/metricheskiye-metody