# KNN with KD-Tree

В этом задании вам предстоит реализовать алгоритм k-ближайших соседей (KNN) с использованием структуры данных KD-Tree. KD-Tree позволяет эффективно искать ближайших соседей, ускоряя процесс классификации.

## 1. KD-Tree [5 баллов]

**KD-Tree** (k-dimensional tree) — это структура данных для быстрого поиска ближайших точек в k-мерном пространстве. Алгоритм построения KD-Tree включает следующие шаги:

1. Выбрать ось разбиения, будем использовать стратегию максимальной вариативности – ось с наибольшим разбросом значений используется для разбиения.
2. Выбрать медиану значений точек вдоль выбранной оси и разделить множество на две части.
3. Рекурсивно построить левое и правое поддерево, повторяя процесс разбиения.

Поиск ближайших соседей в KD-Tree выполняется с использованием обхода дерева в глубину с отсечением заведомо неподходящих ветвей.

Более подробно можно найти на просторах интернета, например, вот [тут](https://medium.com/@katyayanivemula90/what-is-a-k-dimensional-tree-8265cc737d77).

Реализуйте класс KDTree, который содержит:

1. Метод \_\_init\_\_ должен принимать на вход набор точек X, по которому будет строиться дерево, а так же размер листов leaf\_size построенного дерева.
2. Метод query должен принимать на вход набор точек X, для каждой из которых необходимо найти k ближайших соседей.

Отдельно протестируйте вашу реализацию, тест должен содержать генерацию случайного X\_train небольшого размера (100-200 точек), но с разными размерностями и X\_test (30 точек), а затем сравнивать поиск ближайших через честный поиск и через KD-дерево.

## 2. KNNClassifier [2 балла]

Реализуйте KNNClassifier, используя KD-дерево. Класс должен содержать:

1. Метод \_\_init\_\_ принимает на вход количество соседей, по которым предсказывается класс, и размер листьев KD-дерева.
2. Метод fit должен по набору данных и меток строить классификатор.
3. Метод predict\_proba должен предсказывать вероятности классов для заданного набора данных основываясь на классах соседей
4. Метод predict, который использует predict\_proba и возвращает вектор с предсказанными классами

## 3. Processing [2 балла]

Для успешного обучения классификатора необходимо правильно подготовить данные и корректно оценить качество. Для этого реализуйте:

1. Функцию для разбиения данных на холдауты – train\_test\_split
2. Различные стратегии нормализации данных, можно использовать любые, но все должны поддерживать общий интерфейс – fit для сохранения статистик по данным, transform для преобразования данных по сохраненным статистикам и fit\_transform для объединения двух предыдущих методов
3. В качестве метрик будем использовать стандартные accuracy и f1\_score, реализуйте их вычисление. Каждая метрика принимает y\_pred и y\_true и возвращает число

## 4. Обучение [1 балл]

Обучите классификатор на наборах данных cancer.csv, поиск злокачественной опухоли, и spam.csv, детекция спам-сообщений. Для каждого набора данных необходимо:

1. Прочитать и подготовить набор данных
2. Обучить и подобрать оптимальный k
3. Построить график зависимости метрик от k (для удобства, можно на одном графике)
4. Сделать выводы