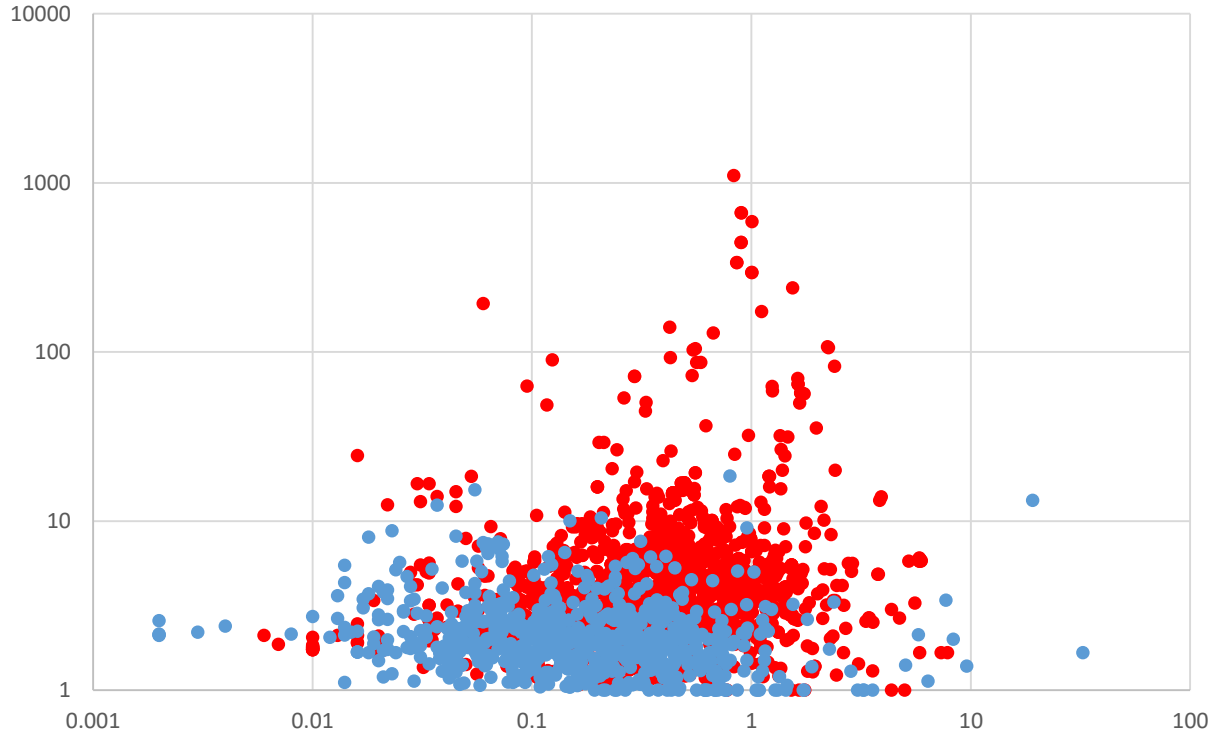
Задача классификации

- Ввод (Input): $\mathbf{x}$
- Вывод (Output): $\boldsymbol{y}$
- Целевая зависимость (Target function): $\boldsymbol{f}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$
- Данные (Data): $(\mathbf{x_1}, \boldsymbol{y_1}), (\mathbf{x_2}, \boldsymbol{y_2}), \ldots, (\mathbf{x}_N, \boldsymbol{y}_N)$
- Гипотеза (Hypothesis): $\boldsymbol{h}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$

Y – множество классов



Гипотеза компактности

«Гипотеза компактности — в задачах классификации предположение о том, что схожие объекты гораздо чаще лежат в одном классе, чем в разных; или, другими словами, что классы образуют компактно локализованные подмножества в пространстве объектов.»

«В математическом анализе *компактными* называются ограниченные замкнутые множества. *Гипотеза компактности* не имеет ничего общего с этим понятием и должна пониматься в «более бытовом» смысле этого слова.»

machinelearning.ru

Метрические классификаторы

Метрический классификатор
Lazy learning

$$h(\mathrm{x}; D) = \arg\max_{y \in \mathrm{Y}} \sum_{\mathrm{x}_i \in D} \underbrace{[y_i = y] w(\mathrm{x}_i, \mathrm{x})}_{\Gamma_y(\mathrm{x})}$$

$w(\mathrm{x}_i, \mathrm{x})$ — вес соседа x_i

$\Gamma_y(\mathrm{x})$ – близость x к классу *y*

kNN – k ближайших соседей

$$w(\mathrm{x}_i, \mathrm{x}) = 1, \text{если } \mathrm{x}_i - \text{один из k ближайших соседей}$$

$$w(\mathrm{x}_i, \mathrm{x}) = 1, \text{если } \rho(\mathrm{x}_\mathrm{i}, \mathrm{x}) < \mathrm{R} \text{ (Radius Neighbors)}$$

Оценка качества – “Leave One Out”:

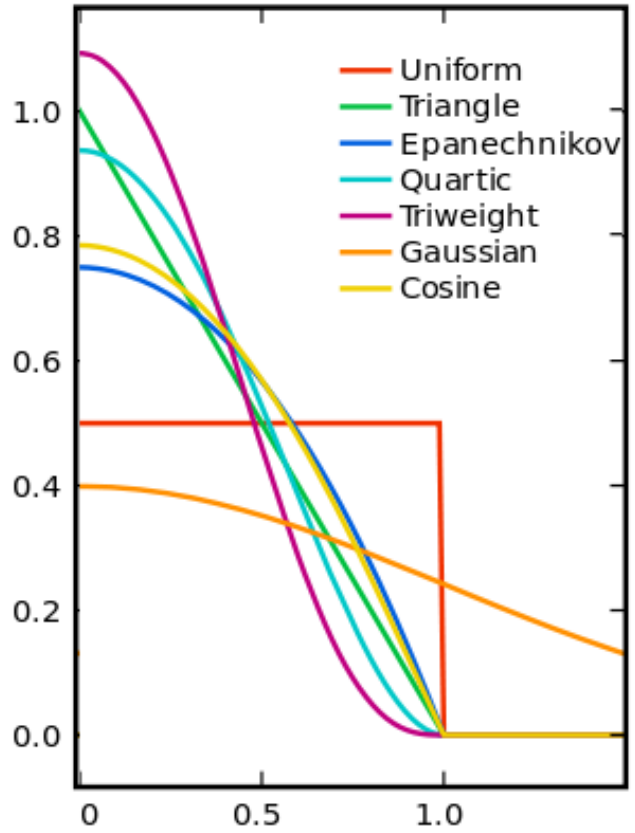
$$\mathrm{LOO}(k, D) = \sum_{\mathrm{x}_i \in D} h_k(\mathrm{x}_i; D \backslash \mathrm{x}_i) \neq y_i$$

WkNN – k взвешенных ближайших соседей

Варианты *w*:

$$w_i = \frac{r - \rho(\mathrm{x}, \mathrm{x}_i)}{r}$$

$$w_i = q^{-\rho(\mathrm{x}, \mathrm{x}_i)}$$



Метод окна Парзена (Parzen window):

$$w(\mathrm{x}, \mathrm{x}_i) = K\left(\frac{\rho(\mathrm{x}, \mathrm{x}_i)}{r}\right)$$

Фиксированная ширина

$$w(\mathrm{x}, \mathrm{x}_i) = K\left(\frac{\rho(\mathrm{x}, \mathrm{x}_i)}{\rho(\mathrm{x}, \mathrm{x}_j)}\right)$$, где x_j- (k+1)-й сосед

Переменная ширина

Метод потенциальных функций

$$h(\mathrm{x}, \mathrm{D}) = \arg\max_{y \in \mathrm{Y}} \sum_{\mathrm{x}_i \in D} [y_i = y] \gamma_i K\left(\frac{\rho(\mathrm{x}, \mathrm{x_i})}{r_i}\right)$$

γ_i — веса объектов (заряд)

r_i - радиус действия

Инициализация: $\gamma_i = 0$

Если $h(\mathrm{x}_i) \neq y_i \quad \rightarrow \quad \gamma_i = \gamma_i + 1$

В самом начале можно выбрать случайно или по наибольшему классу.

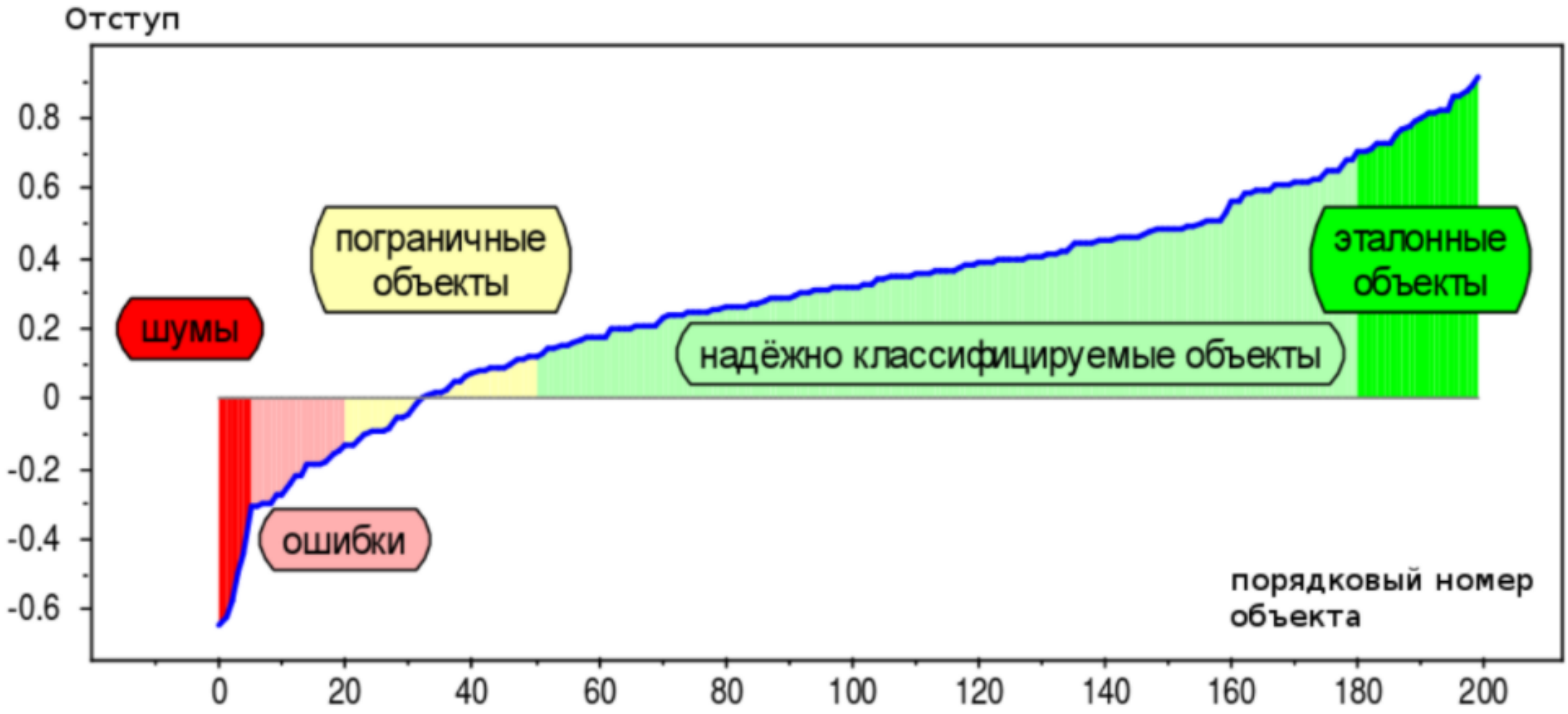
Prototype selection (Отбор эталонов)

$$f(\mathrm{x}): X \to Y$$

$$h(\mathrm{x}) = \arg\max_{y \in Y} \Gamma_y(\mathrm{x})$$

Margin (отступ): $\mathrm{M}(\mathrm{x}_i) = \Gamma_{y_i}(\mathrm{x}_i) - \max_{y \in Y \setminus y_i} \Gamma_y(\mathrm{x}_i)$

Objects by margin



Prototype selection (Отбор эталонов)

$$h(\mathrm{x};\Omega) = \arg\max_{y \in Y} \sum_{\mathrm{x}_i \in \Omega} [y_i = y]\, w(\mathrm{x}_i, \mathrm{x})$$

Методы отбора эталонов

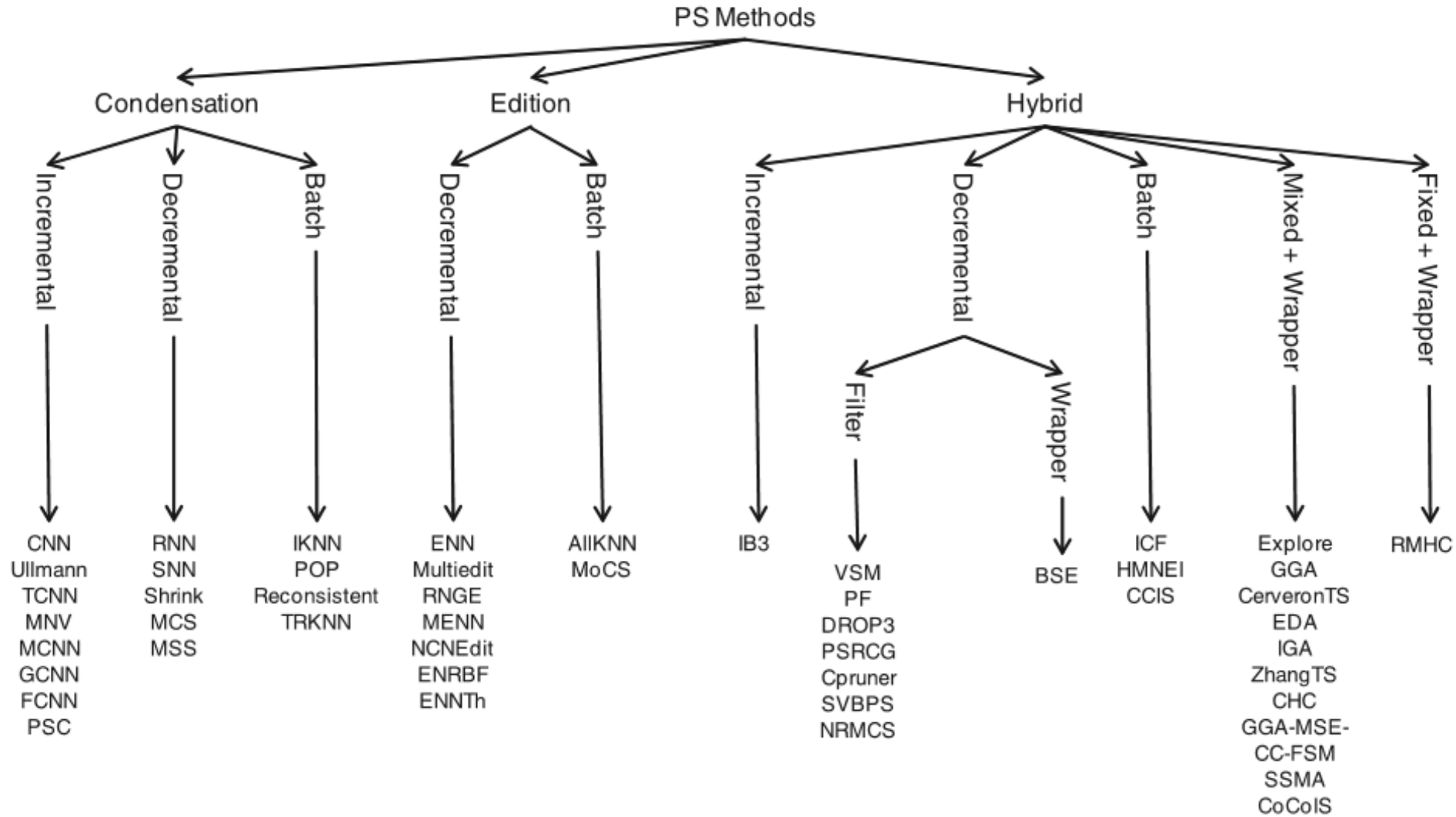
Направление поиска:

- Incremental
- Decremental
- Batch (Decremental)
- Mixed
- Fixed (Mixed)
- Replacement

Тип выбора:

- Condensation
- Edition
- Hybrid

Методы отбора эталонов



DROP5
(*Decremental Reduction Optimization Procedure)*

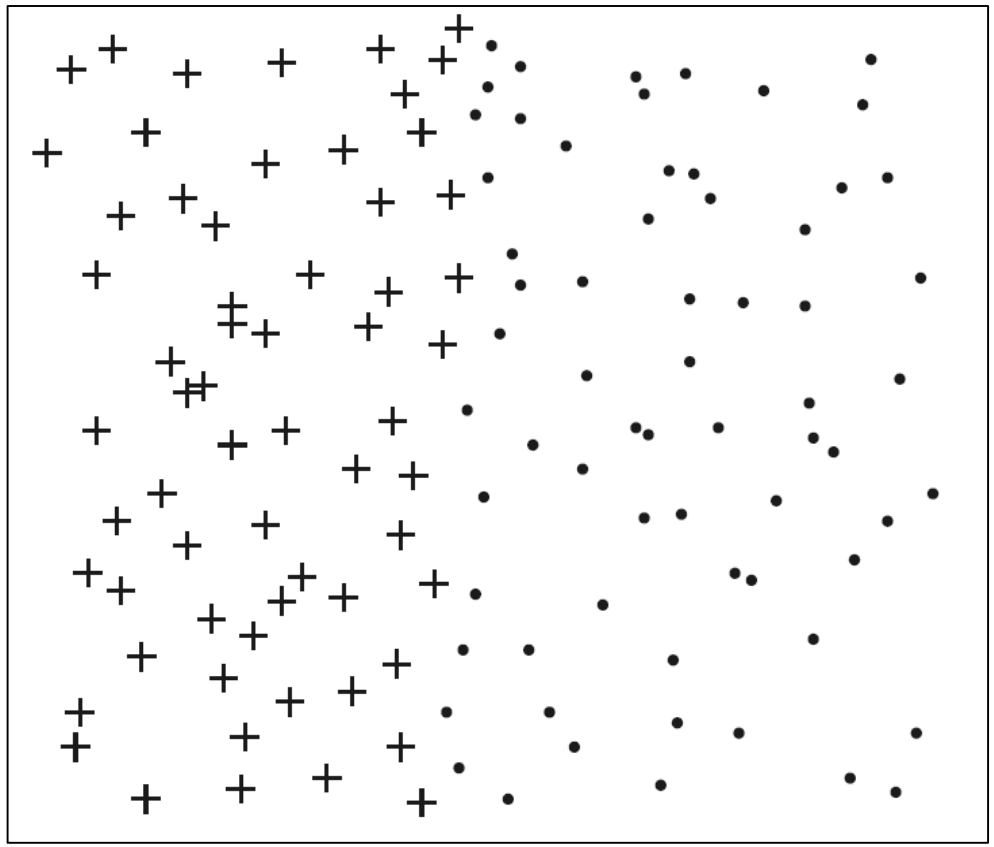
Начинаем с полного набора обучающих точек.

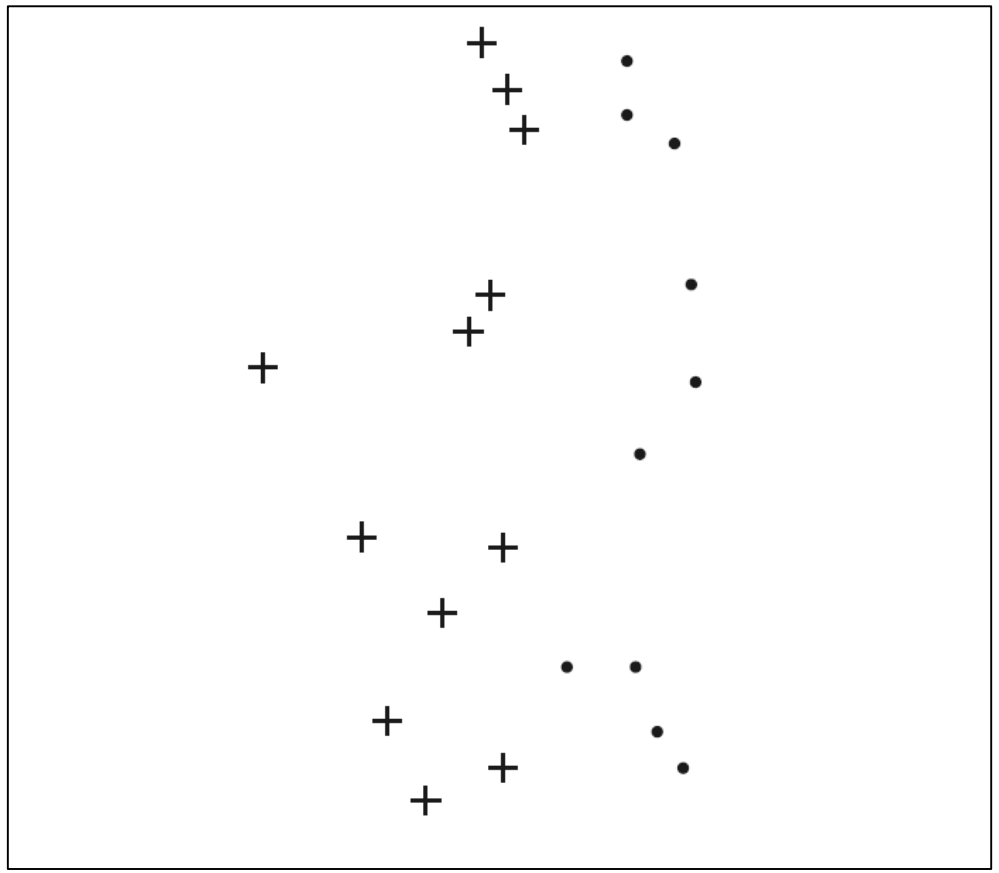
Отсортируем точки по расстоянию до ближайшей точки другого класса.

Идем от наименьшего расстояния к наибольшему (DROP5).

Удалить точку P, если среди точек из полного набора, у которых P была ближайшим соседом, правильно классифицированных останется столько же.

DROP5





Неравномерные признаки:

$$\rho(\mathrm{x}, \mathrm{x}_i) = \left(\sum_{j=1}^{n} w_j \left| x_j - x_{ij} \right|^p \right)^{\frac{1}{p}}$$

Метрика Минковского

Роль весов w:

1. Нормировка
2. Степень важности
3. Отбор

Добавление признаков

1. Выбираем один лучший признак k: $\rho_k(\mathrm{x}_i, \mathrm{x}_j) = |\mathrm{x}_{jk} - \mathrm{x}_{ik}|$
2. Есть расстояние ρ.
3. Добавляем признак *k'*:

$$\rho(\mathrm{x}_i, \mathrm{x}_j) = \rho(\mathrm{x}_i, \mathrm{x}_j) + w_{k'}\rho_{k'}(\mathrm{x}_i, \mathrm{x}_j)$$

4. Можно заменять признаки:

$$\rho(\mathrm{x}_i, \mathrm{x}_j) = \rho(\mathrm{x}_i, \mathrm{x}_j) - w_{k''}\rho_{k''}(\mathrm{x}_i, \mathrm{x}_j) + w_{k'}\rho_{k'}(\mathrm{x}_i, \mathrm{x}_j)$$

Будем добавлять, пока LOO уменьшается.

Быстрый поиск соседей
k-d tree

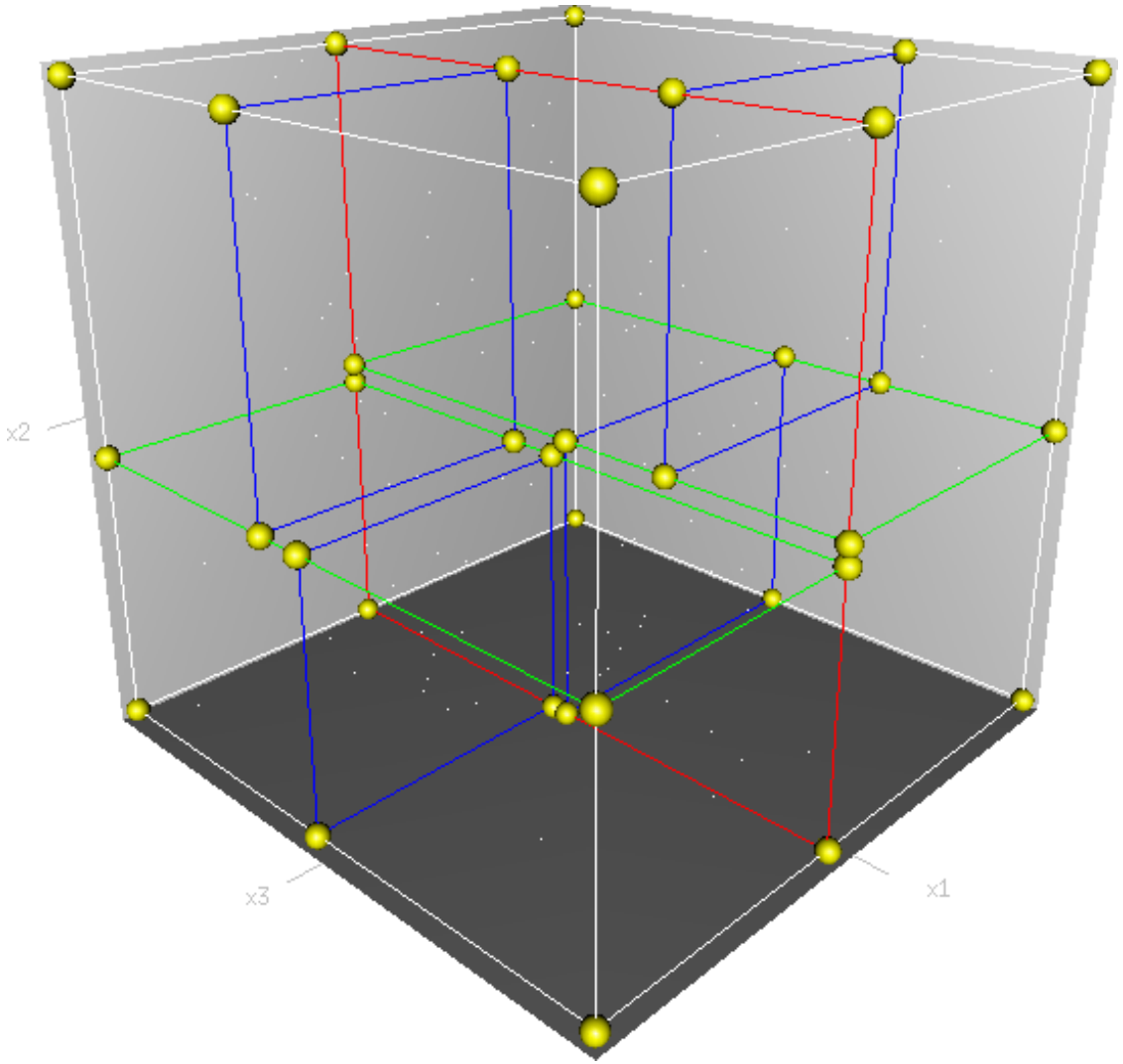
Разбиваем пространство гиперплоскостями, ортогональными одной из координатных осей, последовательно по медианным точкам.

Получаем участки с близким количеством точек в них (листы дерева).

Поиск соседей начинаем с точек листа, в котором находится точка, если соседей недостаточно – поднимаемся на узел выше.

Возможны ошибки, но их можно избежать, если следить, насколько близко точка лежит к границе листа.

На картинке слева – дерево глубины 3, делящее пространство на 8 листов (сначала по красной гиперплоскости, потом по зеленым гиперплоскостям, потом по синим).



Домашнее задание

Мягкий дедлайн (на полный балл) – в следующий четверг,

Жесткий дедлайн (на половинный балл) – еще через неделю.

Задачи можно решать на любом языке в любом окружении, но ОЧЕНЬ РЕКОМЕНДУЕТСЯ использовать Python и Jupyter Notebook.

На паре после лекции можно будет сдавать домашнее задание и задавать вопросы, если что-то не получается.

Ближайший мягкий дедлайн – 15 марта, жесткий – 22 марта.

Также есть дополнительные задачи на дополнительные баллы и более поздний дедлайн:

- Для того, чтобы сдавать дополнительные задания, нужно сделать все основные
- Больше 120-ти баллов набрать нельзя
- Для получения зачета нужно набрать половину баллов основных задач каждой домашки (можно за счет дополнительных :))