Метрические алгоритмы классификации. Метод ближайших соседей

Преподаватель: Рыжиков А.С.

НИУ ВШЭ, Москва, 2018

Содержание

1. Метод ближайших соседей
   - Гипотеза компактности
   - Метод ближайших соседей
   - Окно Парсена и потенциальные функции
2. Модификации
   - Поиск потенциальных функций
   - Отбор эталонов и отсеивание выбросов
   - Методы поиска ближайших соседей
   - Другие модификации scikit-learn
3. Пакет scikit-learn

Гипотеза компактности

Задача классификации

Дано: X - объекты, Y - ответы (метки классов), $X^l = (x_i, y_y)_{i=1}^l$ - объекты обучающей выборки
Найти: $f(x)$ для $Y \approx f(X)$

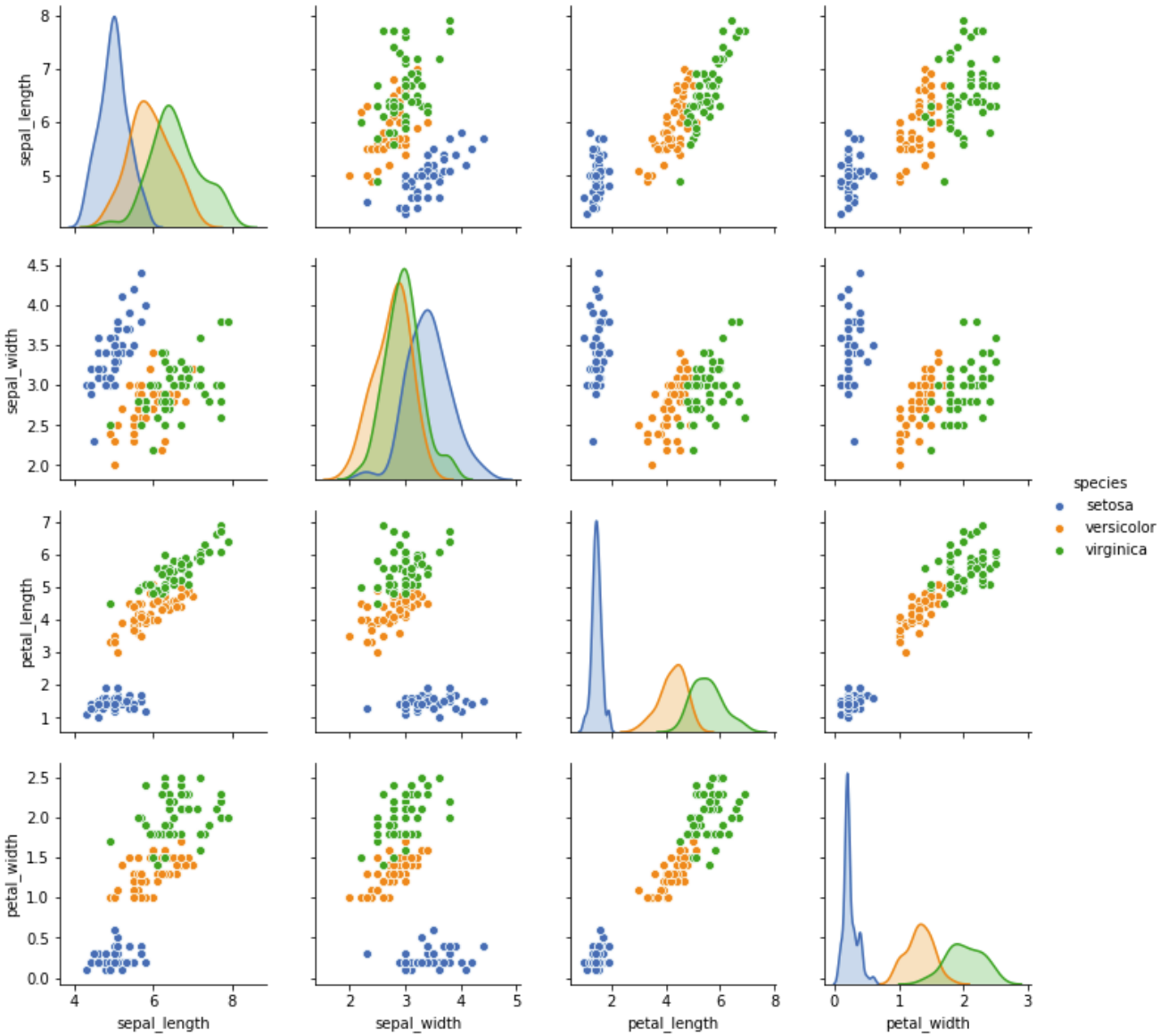
Гипотеза компактности

Идея: объекты одного класса более похожи друг на друга

Мера сходства

Вопрос: а что значит "похожи"?
Ответ: объекты x_i, x_j похожи, если расстояние $\rho(x_i, x_j) \to 0$

Гипотеза компактности. Иллюстрация



Примеры расстояний

- $\rho_{minkowski,p}(x_i, x_j) = \left(\Sigma_{f=1}^{F}(x_i^f - x_j^f)^p\right)^{1/p}$, где F - количество признаков (измерений) для объектов
- расстояние *Махаланобиса* $\rho_\Sigma(x_i, x_j) = \sqrt{(x_i - x_j)^T \Sigma (x_i - x_j)}$, где Σ - симметричная положительно-определённая матрица
- расстояние *Левенштейна* между строковыми объектами - редакторское расстояние. Пример: $\rho(hello, hell) = 1, \rho(hello, ehllo) = 1$
- расстояния на других категориальных признаках

Расстояния на категориальных признаках. Примеры

- $\rho(x_i, x_j) = int(x_i != x_j)$ - индикатор совпадения
- $\rho(x_i, x_j) = int(x_i != x_j) + int(x_i == x_j)p(x_j)$ - сглаженный индикатор совпадения
- $\rho(x_i, x_j) = \frac{int(x_i != x_j)}{p(x_j)p(x_j)}$

Метод ближайших соседей

Для задачи бинарной классификации

$f(x, X_{train}) = sign(\Sigma_{x_i \in X_K(x)} w(x_i, x) y_i))$, где $X_K(x)$ - K ближайших соседей для нового объекта x

Для многоклассовой классификации

$f(x, X_{train}) = argmax_{y \in \mathbb{Y}}(\Sigma_{x_i \in X_K(x)} w(x_i, x)[y_i = y])) = argmax_{y \in \mathbb{Y}} \Gamma_y(x)$, где $X_K(x)$ - K ближайших соседей для нового объекта x

Метод ближайшего соседа

$w(x,x_i) = [i = 1]$

Преимущества:

- простота реализации
- интерпретируемость, вывод на основе прецедентов (case-base reasoning, CBR)
- скорость обучения и применения (нужно считать 1 ближайшего)

Недостатки:

- неустойчивость к погрешностям (шум, выбросы)
- отсутствие настраиваемых параметров
- низкое качество классификации
- приходится хранить всю выборку целиком

Метод K ближайших соседей

$w(x,x_i) = [i \leq K]$

Преимущества:

- менее чувствителен к шуму
- появился параметр K

Недостатки:

- работает чуть дольше (вместо поиска одного ближайшего соседа K)
- необходимость в поиске оптимального K
- неоднозначность при классификации - $\Gamma_{y1}(x) = \Gamma_{y2}(x)$
- разные соседи учитываются с одним весом

Метод K взвешенных ближайших соседей

$$w(x,x_i) = [i \leq K]w_i$$

Преимущества:

- однозначность классификации

Недостатки:

- каким должно быть w_i?

Метод окна Парсена

$w(x_i, x) = \mathcal{K}(\frac{\rho(x_i,x)}{h})$, где $\mathcal{K}$ - ядро, невозрастающее, положительное на [0,1]

Окно постоянной ширины h:

$$f(x, X_{train}, \mathbf{h}) = argmax_{y \in \mathbb{Y}}(\Sigma_{x_i \in X_K(x)} K(\frac{\rho(x_i, x)}{\mathbf{h}})[y_i = y]))$$

Окно переменной ширины:

$$f(x, X_{train}) = argmax_{y \in \mathbb{Y}}(\Sigma_{x_i \in X_K(x)} K(\frac{\rho(x_i, x)}{\rho(x_{K+1}, x)})[y_i = y]))$$

Метод потенциальных функций

$w(x_i, x) = \gamma(x_i)\mathcal{K}(\frac{\rho(x_i,x)}{h_i})$, где $\gamma(x_i) \leq 0$ - вес i'го соседа

Физическая аналогия

$\gamma(x_i) \leq 0$ - величина заряда в точке x_i

h_i - характерный радиус воздействия вокруг точки обучающей выборки x_i

$\mathcal{K}$ - потенциал

y_i - знак заряда (в случае бинарной классификации)

Алгоритм настройки весов объектов

Data: X^l - обучающая выборка
Result: $\gamma(x_i) : \forall x_i \in X^l$
инициализация: $\gamma(x_i) = \gamma_i = 0 : \forall i \in X^l$;
while *число ошибок на выборке* $Q(f, X^l) > \epsilon$ **do**
    выбрать объект $x_i \in X^l$;
    if $f(x_i) \neq y_i$ **then**
        $\gamma_i = \gamma_i + 1$
    end
end

Algorithm 1: Простой алгоритм настройки весов объектов

Преимущества и недостатки метода потенциальных функций

Преимущества:

- простота реализации
- не надо хранить выборку (потоковый алгоритм обучения)
- разреженность: все объекты с $\gamma_i = 0$ можно выкинуть

Недостатки:

- медленная сходимость
- результат обучения сильно зависит от порядка просмотров объектов
- слишком грубо настраиваются веса γ_i
- вообще не настраиваются параметры h_i
- неустойчивость к шуму. Переобучение

Понятие отступа

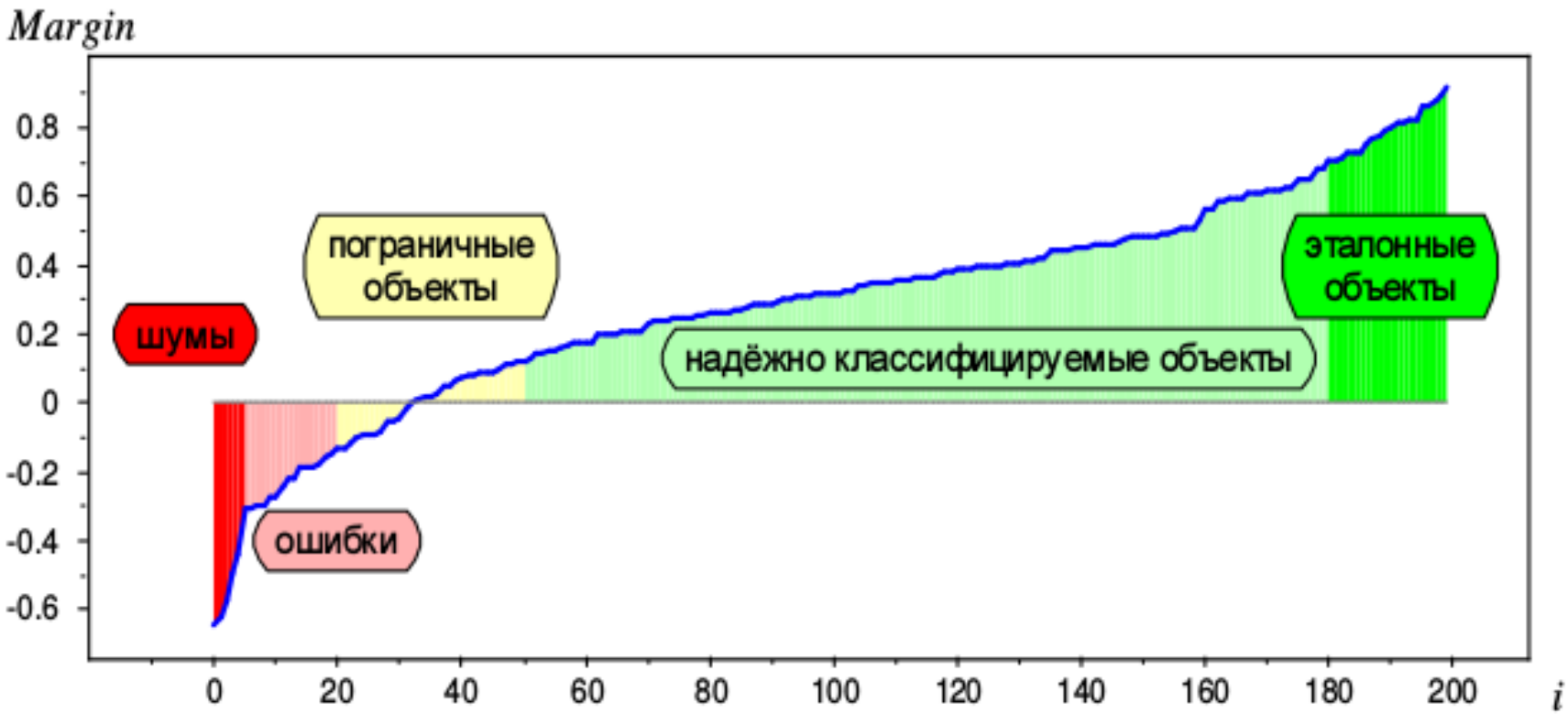
Вопрос

Как получить степень эталонности объекта для классификатора?

Отступ

$\Gamma_y(x) = \Sigma_{x_i \in x_K(x)} w(x_i, x)[y_i = y]$ - степень уверенности классификатора в принадлежности x к классу y
$M(x_i) = \Gamma_{y_i}(x_i) - max_{y \neq y_y} \Gamma_y(x_i)$ - Степень типичности объекта x_i для своего класса y_i (Margin, отступ)

Понятие отступа



Отбор эталонов

Задача: выбрать оптимальное подмножество эталонов $\Omega \subseteq X^l$
$f(x, \Omega) = argmax_{y \in \mathbb{Y}}(\Sigma_{x_i \in x_K(x)} w(x_i, x)[y_i = y]))$
Отличие от предыдущих решений: теперь K ближайших соседей ищется не по всей обучающей выборке, а только по необходимому и достаточному подмножеству эталонов Ω
Вопрос: как отбирать эталонные объекты?
Ответ: алгоритм STOLP

- Исключить выбросы и, возможно, пограничные объекты
- найти по одному эталону в каждом классе
- повторять, пока есть отрицательные отступы

Алгоритм STOLP

Data: X^l - обучающая выборка, δ, l_0
Result: Множество опорных объектов $\Omega \subseteq X^l$
forall $x_i \in X^l$ *проверить, является ли* x_i *выбросом* **do**
  if $M(x_i, X^l) < \delta$ **then**
    $X^l = X^l \backslash \{x_i\}$
  else
end
Инициализация: взять по одному эталону от каждого класса:
$\Omega = \{argmax_{x_i \in X^l_y} M(x_i, X^l) | y \in Y\}$
while $\Omega \neq X^l$ **do**
  выделить множество объектов с ошибкой $f(x, \Omega)$:
  $E = \{x_i \in X^l \backslash \Omega : M(x_i, \Omega) < 0\}$
  if $|E| < l_0$ **then**
    break;
  end
  Присоединить к Ω объект с наименьшим отступом:
  $x_i = argmin_{x \in E} M(x, \Omega), \Omega = \Omega \cup \{x_i\}$
end

Algorithm 2: STOLP

Алгоритм STOLP: преимущества и недостатки

Преимущества:

- устойчивость к выбросам
- меньший размер хранимых в памяти объектов
- большая скорость

Недостатки:

- жадность алгоритма (множество эталонов остаётся немножко избыточно)
- необходимость задавать параметр δ

Методы поиска ближайших соседей

Вопрос: какова алгоритмическая сложность поиска ближайшего соседа для выборки размера N размерности D (D признаков)?

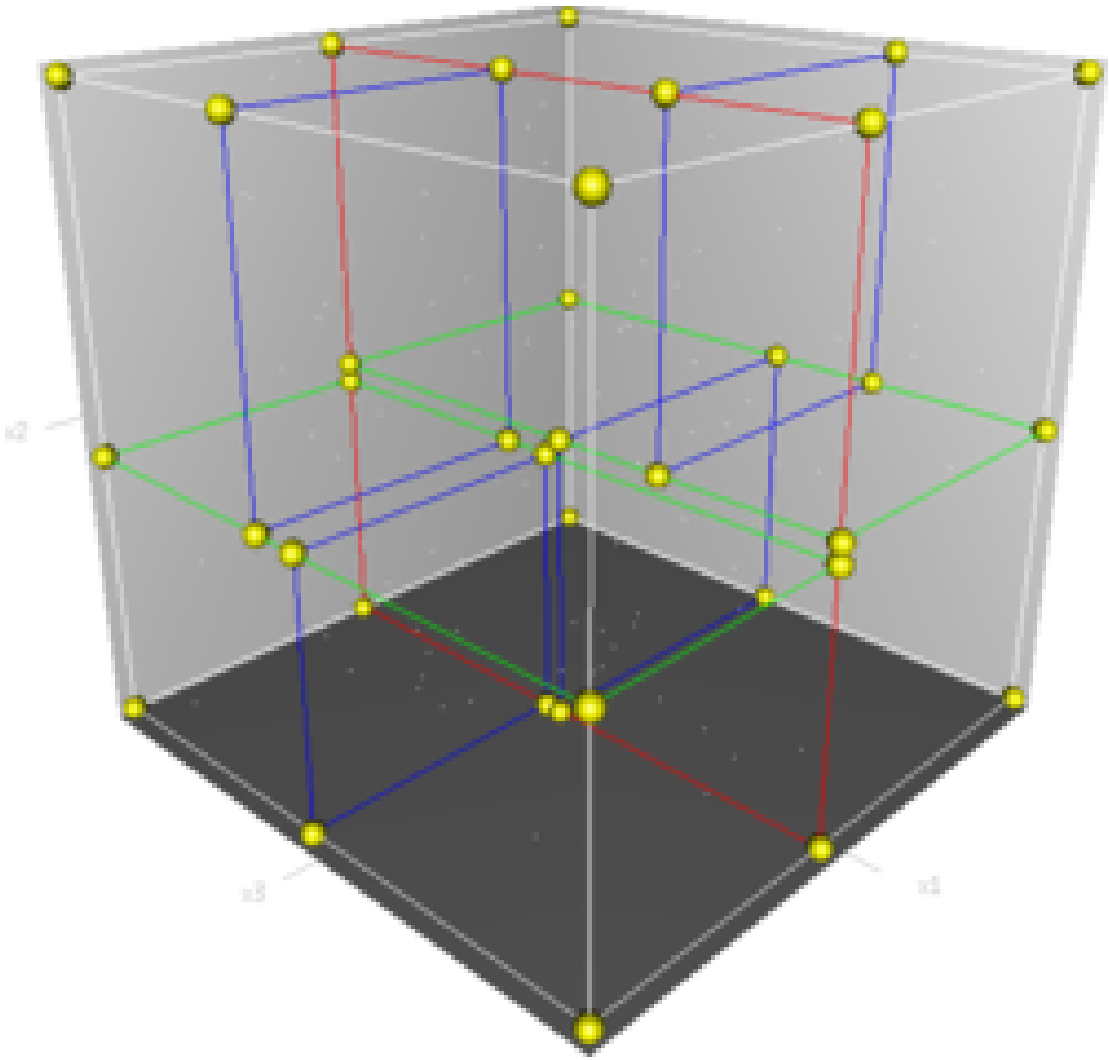
Методы поиска ближайших соседей

Вопрос: какова алгоритмическая сложность поиска ближайшего соседа для выборки размера N размерности D (D признаков)? **Ответ:** O(ND)

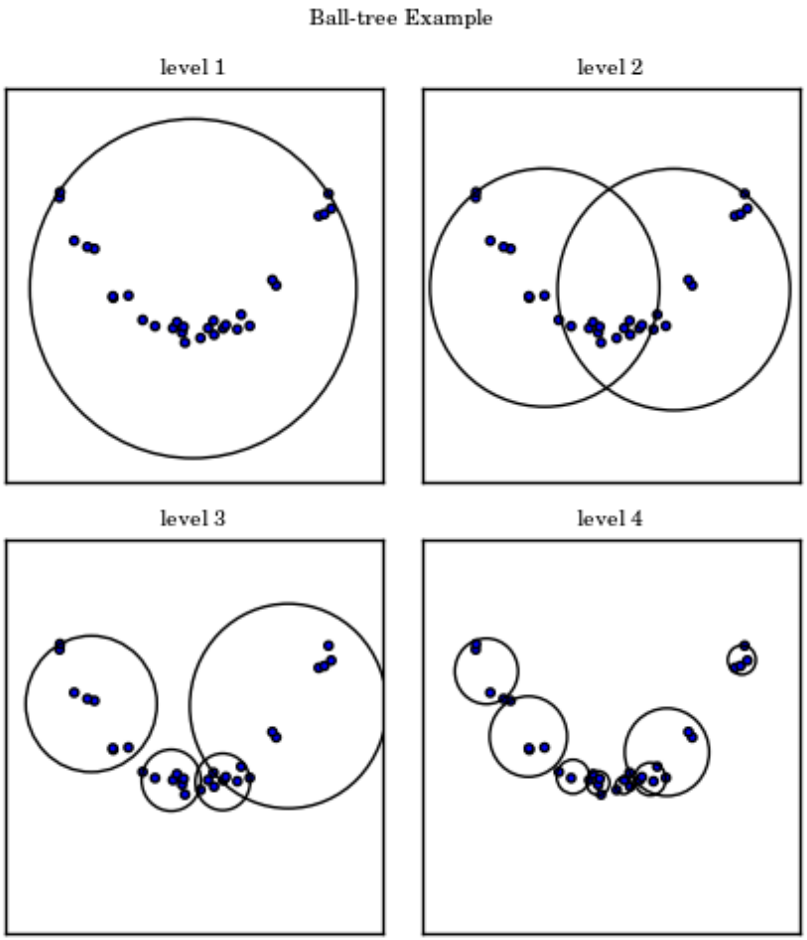
Методы поиска ближайших соседей

- bruteforce
- KDTree
- BallTree

KDTree



BallTree



Алгоритмическая сложность

- bruteforce - гарантированно O(DN)
- KDTree - O(Dlog(N)) при малом D, O(DN) иначе
- BallTree - O(Dlog(N)) гарантированно

Другие модификации scikit-learn

- RadiusNeighboursClassifier - не K ближайших соседей, а из некоторого радиуса
- NearestCentroidClassifier - не K ближайших соседей, а ближайший центроид

Пакет scikit-learn. Параметры

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html