Классификация. Метод ближайших соседей (kNN). Машины опорных векторов (SVM)

Сергей Лисицын

lisitsyn.s.o@gmail.com

22 марта 2011 г.

Задача классификации
classification

- $\sim$ задача распознавания образов
- Имеется некоторое пространство объектов
- Объекты разделены каким-либо образом на классы – непересекающиеся множества схожих объектов
- Известны классы некоторых из объектов (прецеденты)
- Решение задачи – алгоритм, позволяющий определить класс для любого элемента пространства объектов на основе имеющихся прецедентов
- В некоторых случаях необходимо определение вероятностей классов

Общая задача классификации

Пусть имеется некоторое множество объектов $\mathcal{X}$, конечное множество классов $\mathcal{C}$ и определено отображение

$$f : \mathcal{X} \to \mathcal{C},$$

причём известно, что некоторым элементам $X \subset \mathcal{X}$ соответствуют некоторые классы из множества $\mathcal{C}$. Множество

$$\{X, f(X)\} = \{(x_i, f(x_i))\}_{i=1}^{N}$$

называется обучающей выборкой.

Задача классификации состоит в нахождении функции $\hat{f}$, аппроксимирующей f на всех элементах $\mathcal{X}$, которая позволит любой объект из $\mathcal{X}$ отнести к некоторому классу из $\mathcal{C}$.

Для чего это нужно?

- Распознавание изображений
- Поддержка решений в банковской сфере
- Дифференциальная диагностика
- Классификация документов
- Анализ геостатистической информации
- Другие задачи, требующие «узнавания объектов»

Качество классификации

- На каждом объекте x из обучающей выборки $(X, f(X))$ можно вычислить функцию потерь построенного алгоритма $\hat{f}$

$$L_{\hat{f}}(x) = [\hat{f}(x) \neq f(x)]$$

- Средняя сумма ошибок на всей обучающей выборке $x_1, \ldots, x_N$ – эмпирический риск – даёт понятие о качестве построенного алгоритма

$$\hat{R}(\hat{f}) = \frac{1}{N} \sum_{i=1}^{N} L_{\hat{f}}(x_i)$$

- Вполне ясно, что лучше этот риск минимизировать
- В вычислительной теории обучения эта мысль формализована в принцип минимизации эмпирического риска

Принцип минимизации эмпирического риска
Empirical Risk Minimization (ERM) principle

- Функционал эмпирического риска $\hat{R}(\hat{f})$ должен минимизироваться, чтобы обеспечить максимально возможно высокое качество на любых данных
- Минимизация риска встречается даже в статистике
  - $\hat{R}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2 \to \min$ – метод наименьших квадратов
  - $\hat{R}(\hat{p}) = -\frac{1}{N}\sum_{i=1}^{N}\ln \hat{p}(x_i) \to \min$ – метод максимального правдоподобия Фишера
- Однако, безусловный минимум риска почти никогда не даст правильного алгоритма классификации, он слишком приспособлен к обучающим данным

Обобщающая способность алгоритма классификации
Generalization ability

- Хороший алгоритм классификации не только допускает мало ошибок на обучающей выборке, но и мало ошибается на любых других данных
- Обобщающей способностью алгоритма называется его способность допускать мало ошибок на тестовых данных после обучения
- Теоретические оценки обобщающей способности чрезвычайно завышены и пока не могут дать ответа о применимости конкретных алгоритмов обучения к конкретным данным
- На практике оценить обобщающую способность можно с помощью скользящего контроля по обучающей выборке

Скользящий контроль
Cross validation (CV)

- Процедура скользящего контроля позволяет оценить насколько хорошо алгоритм обучения обобщает данные. Суммарная ошибка

$$CV(\mu, \mathcal{X}) = \sum_{\mathcal{X}_1, \mathcal{X}_2 : \mathcal{X}_1 \cup \mathcal{X}_2 = \mathcal{X}} Q\Big(\underbrace{\mu(\mathcal{X}_1)}_{\substack{\text{алгоритм,} \\ \text{построенный на} \\ \text{основе выборки } \mathcal{X}_1}}, \mathcal{X}_2 \Big)$$

- Контроль по q блокам (q-fold CV) разбивает обучающую выборку на q непересекающихся подмножеств
- Минимум функции ошибки на скользящем контроле важен для настройки параметров алгоритмов и вообще оценки качества

Переобучение и недообучение
Overfitting, underfitting

- Алгоритм вполне может не допускать ошибок на обучающей выборке вообще, но при этом суммарная ошибка скользящего контроля и его результаты на других данных катастрофически плохи. Такое явление называется переобучением (overfitting)
- С другой стороны, алгоритм может одинаково плохо работать как на обучающей выборке, так и на реальных данных – в этом случае алгоритм недообучается (underfitting)
- Для хорошего качества алгоритма необходимо избегать этих крайностей

Принципы MLE и MAP

maximum likelihood estimation (MLE), maximum a posteriori (MAP)

- Метод максимального правдоподобия (MLE): условная вероятность данных при гипотезе максимизируется

$$\hat{h} = \arg\max_h P(\mathcal{D}|h)$$

- Принцип максимальной апостериорной вероятности (MAP): условная вероятность гипотезы при данных максимизируется

$$\hat{h} = \arg\max_h P(h|\mathcal{D})$$

Принцип максимальной апостериорной вероятности для классификации

MAP classification

- В случае классификации гипотеза – принадлежность классифицируемого объекта некоторому классу
- Оптимальным классификатором с точки зрения MAP и ERM является

$$\hat{f}(x) = \arg\max_{c \in \mathcal{C}} P(c|x)$$

- Саму функцию вероятности $P(c|x)$ найти по обучающей выборке не представляется возможным, приходится находить её каким-то иным образом

Классификация с помощью метода ближайших соседей

- Вся идея алгоритма состоит в аппроксимации апостериорной вероятности класса $P(c|x)$ через объекты обучающей выборки и расстояния до них
- Рассуждение по прецедентам (case-based reasoning) хорошо объясняет ответ, полученный алгоритмом
- Относится к классу алгоритмов ленивого обучения (lazy learning): обучение алгоритма сводится к «запоминанию» обучающей выборки
- Необходима метрика между всеми объектами, то есть пространство должно быть метрическим и не содержать категориальных и бинарных признаков

Метрические пространства

- Метрическое пространство – множество, с определённой на нём функцией $\rho(x, y)$ – метрикой
- Метрика должна быть неотрицательной, симметричной и для неё должно выполняться неравенство треугольника
- Примеры метрик:
  1. Метрика Минковского (при $p = 1$ – метрика Манхэттена (city-block), при $p = 2$ – евклидова)

$$\rho(x, y) = \left(\sum_i |x_i - y_i|^p \right)^{\frac{1}{p}}$$

  2. Расстояние Махаланобиса

$$\rho(x, y) = \sqrt{(x - y)^T S^{-1} (x - y)},$$

Метод k ближайших соседей
k Nearest Neighbors (kNN)

- Объекту x присваивается класс, характерный для большинства из k ближайших по метрике объектов:

$$\hat{f}(x) = \arg\max_{c \in C} \sum_{i=1}^{k} [f(y_i) = c],$$

где $\{y_1, y_2, ..y_k\} \subseteq T$ ближайшие к x объекты $\left(\sum_{i=1}^{k} \rho(x, y_i) \to \min\right)$, упорядоченные по возрастанию метрики: $\rho(x, y_1) \leqslant \rho(x, y_2) \leqslant \cdots \leqslant \rho(x, y_k)$

- Имеет один параметр k, иногда существенно влияющий на качество классификации

Метод k взвешенных ближайших соседей

Weighted k nearest neighbors

- В некоторых ситуациях максимум по классу может быть выражен неявно
- Введение весовой функции от ранга объекта $w(i)$ позволяет избежать такой ситуации

$$\hat{f}(x) = \arg\max_{c \in C} \sum_{i=1}^{k} [f(y_i) = c] w(i),$$

где $\{y_1, y_2, ..y_k\} \subseteq T$ ближайшие к x объекты $\left(\sum_{i=1}^{k} \rho(x, y_i) \to \min\right)$, упорядоченные по возрастанию метрики: $\rho(x, y_1) \leqslant \rho(x, y_2) \leqslant \cdots \leqslant \rho(x, y_k)$

- Такой алгоритм ещё устойчивее к шумам и более гибкий: настройке подлежат $w(i)$ и k

Метод парзеновских окон
Parzen kernel window classification

- Весовую функцию можно ввести не от ранга, а от расстояния до объекта

$$\hat{f}(x) = \arg\max_{c \in C} \sum_{e \in X} [f(e) = c] K\left(\frac{\rho(x, e)}{h}\right),$$

- В таком виде не нужно выбирать ближайшие объекты – дальние объекты сами будут иметь маленький вес
- Параметр h называется шириной парзеновского окна и может зависеть от объекта $h(e)$ (метод парзеновских окон переменной ширины)
- Функция ядра K является произвольной положительной невозрастающей функцией (ненулевой на $[0, +\infty)$)

Описание алгоритма

Вход: объект x, подлежащий классификации; множество пар $\{(t_i, f(t_i))\}_i = T$ обучающей выборки, параметр* k, метрика ρ (весовая функция $w(i)$)
Выход: класс, определённый для объекта x

1. найти метрики $\rho(x, t), t \in T$
2. отсортировать обучающую выборку T по убыванию метрики $\rho(x, t), t \in T$
3. выбрать первые k объектов отсортированной выборки
4. просуммировать весовую функцию (1 в случае kNN, $w(i)$ в случае wkNN, $K(\rho/h)$ в случае парзеновских окон) по всем объектам в соответствующие классу объекта элементы ассоциативного массива
5. выбрать из полученного ассоциатива ключ, которому соответствует максимальное значение — класс для объекта x

Для классификатора парзеновских окон шаги 2 и 3 не нужны.

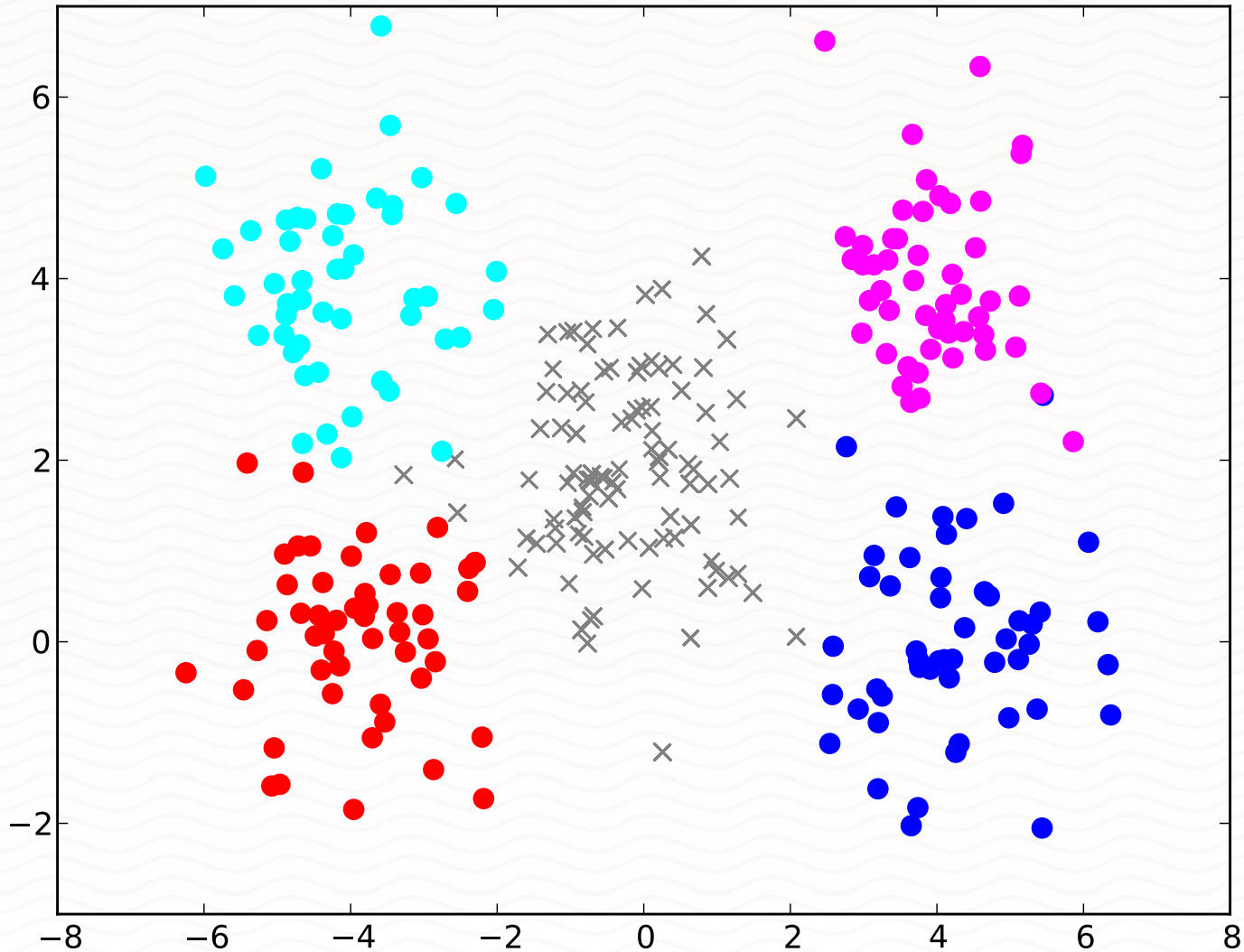
Реализация

Классификаторы weighted k Nearest Neighbors, Parzen window

```
1  def wkNN_classify(obj, rest, k, metric, w = lambda i: 1):
2      distances = \
3      map(lambda (x,c): (x,c,metric(obj,x)), rest)
4      sorted_distances = \
5      sorted(distances, key = lambda (x,c,m): m)[0:k]
6      votes = defaultdict(float)
7      for i, (x,c,m) in enumerate(sorted_distances):
8          votes[c] += w(i)
9      return max(votes,key=votes.get)
10
11 def parzen_classify(obj, rest, metric, K = lambda m: 1/m):
12     distances = \
13     map(lambda (x,c): (x,c,metric(obj,x)), rest)
14     votes = defaultdict(float)
15     for (x,c,m) in distances:
16         votes[c] += K(m)
17     return max(votes,key=votes.get)
```

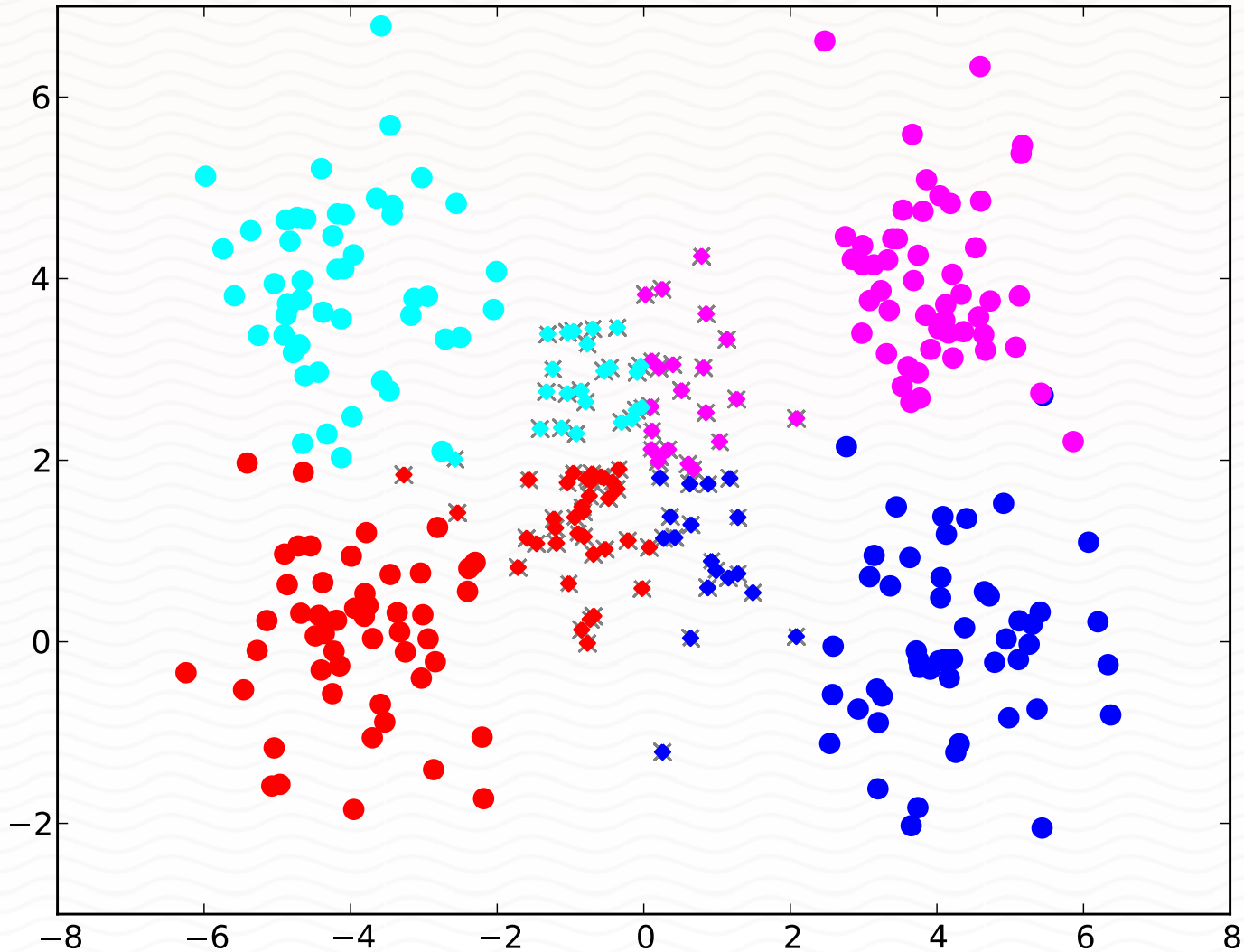
Пример классификации (абстрактный)

wkNN, $k = 4$, $w(i) = k - i$



Пример классификации (абстрактный)

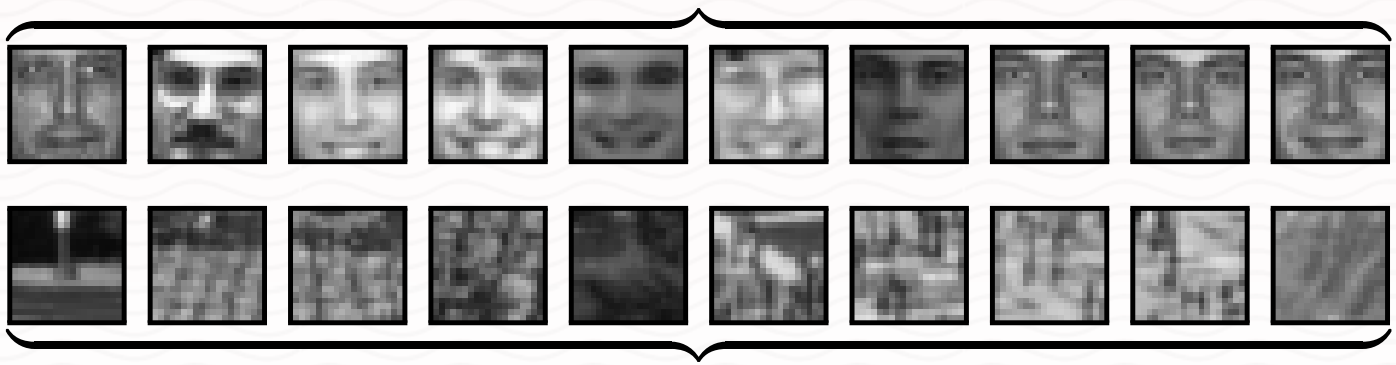
wkNN, $k = 4$, $w(i) = k - i$



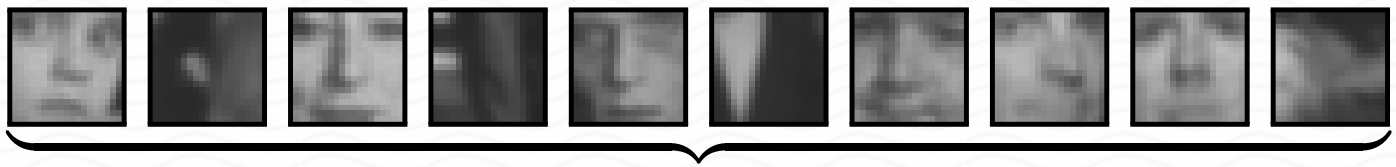
Пример классификации изображений

CBCL (Center For Biological and Computation Learning, MIT) Face Database #1

- Классификация изображений 19x19 на два класса
- Обучающая выборка (сокращённая):

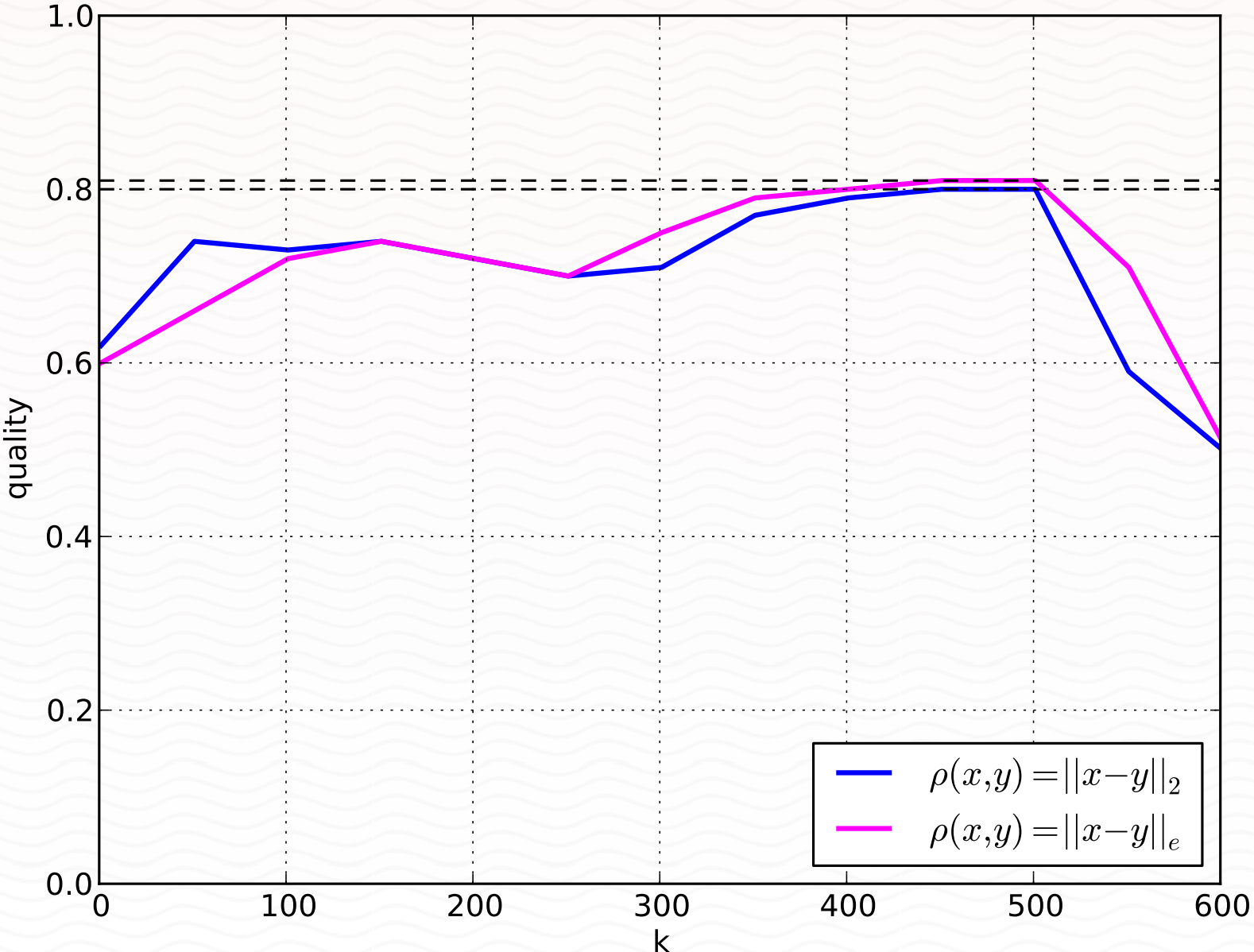


- Тестовая выборка (сокращённая):



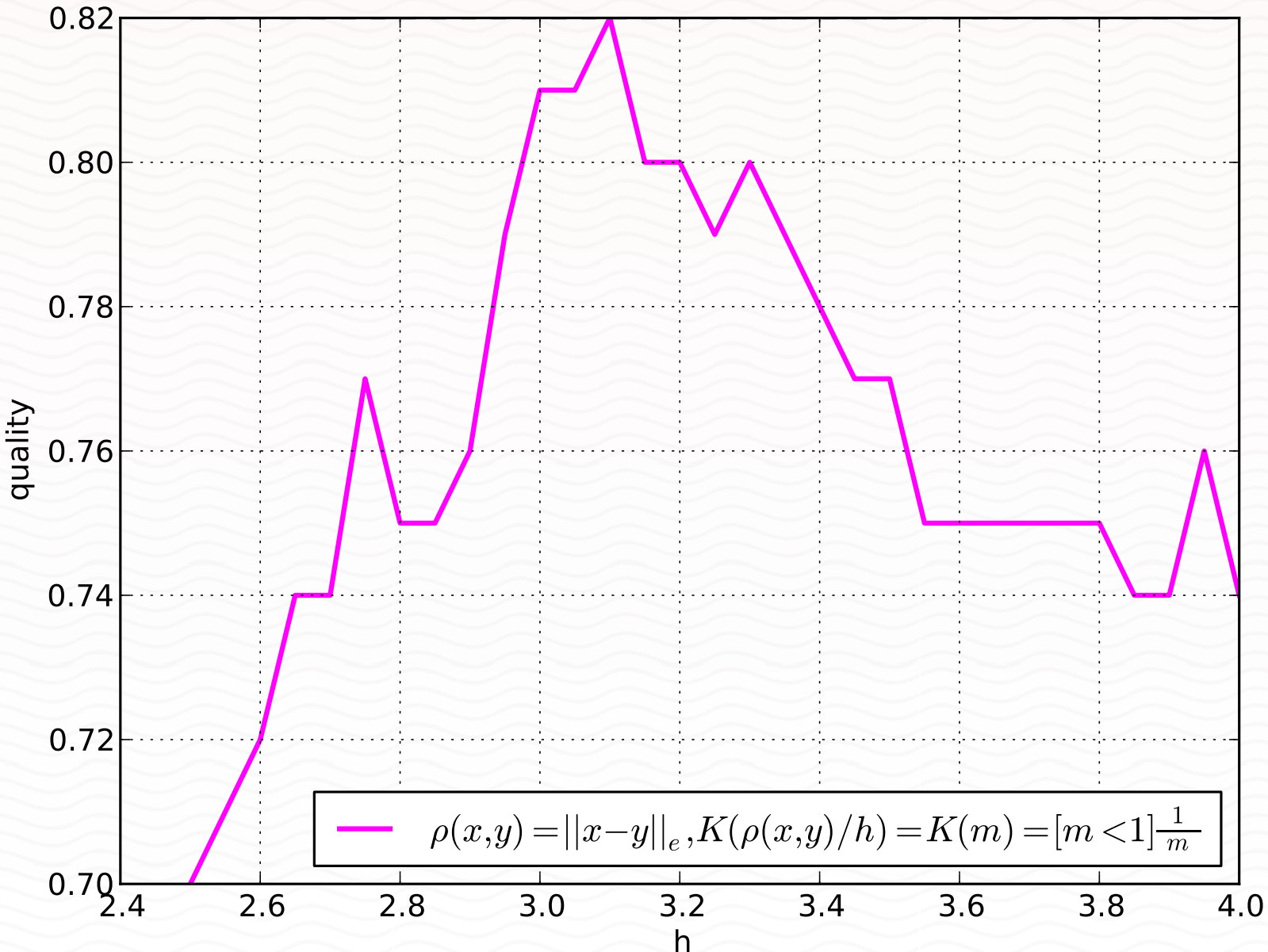
Классификация изображений: kNN

CBCL (Center For Biological and Computation Learning, MIT) Face Database #1



Классификация изображений: парзеновские окна

CBCL (Center For Biological and Computation Learning, MIT) Face Database #1



Способы повышения эффективности классификации

- Выбор более подходящей метрики
- Прореживание выборки (sampling)
- Фильтрация шумов
- Выбор эталонных объектов
- Понижение размерности данных
- Использование эффективных структур для хранения данных (kdTree, BallTree)

Эффективность и вычислительная сложность

- В целом, при адекватном выборе метрики и параметров методы ближайших соседей классифицируют объекты достаточно точно
- Вычислительная сложность алгоритма складывается из сортировки всей обучающей выборки $O(n \ln n)$ и обработки k ближайших объектов за $O(k)$, однако большую часть времени занимает нахождение расстояний до объектов
- Классификация каждого нового объекта требовательна не только по времени, но и по памяти
- Как обработка, так и хранение объектов легко распределяется на несколько вычислительных узлов

What to know?

- Классификация: общий смысл и математическая формулировка задачи
- Принцип минимизации эмпирического риска (ERM), оптимальный с точки зрения ERM классификатор
- Аппроксимация апостериорной вероятности ближайшими соседями
- Методы ближайшего соседа, k ближайших соседей, k взвешенных ближайших соседей, парзеновских окон
- Пример классификации изображений ближайшими соседями

Машины опорных векторов
Support vector machines (SVM)

- Впервые представлен в начале 90-х, но основу метода составляют методы обобщённого портрета Вапника, разработанные ещё в 70-х
- Целое семейство методов решающее задачу классификации и регрессии
- Использует бинарный классификатор, разбивающий выборку на две части (и кое-что ещё)
- Многоклассовые классификаторы строятся как композиции бинарных
- Идея состоит в том, что положение гиперплоскости определяется положениями граничных точек – опорных векторов
- Рассмотрим по мере усложнения: линейный SVM, регуляризованный SVM, ядровой SVM

Линейный SVM

- Линейный алгоритм, классифицирующий некоторые объекты в евклидовом пространстве $\mathbb{R}^n = \mathcal{X}$ на классы $\mathcal{C} = \{-1, +1\}$, определяется следующим образом:

$$f(x) = \begin{cases} -1, & \langle w, x \rangle < w_0 \\ +1, & \langle w, x \rangle \geq w_0 \end{cases}$$

  где $\langle \cdot, \cdot \rangle$ – скалярное произведение.
- Задача обучения алгоритма сводится к нахождению вектора $w = \sum_i \alpha_i f(x_i) x_i$ и параметра w_0
- Такой классификатор хорошо подходит для линейно разделимых выборок (пересечения линейных оболочек объектов классов пустые)

Линейный SVM: поиск параметров

- Из принципа структурной минимизации риска Вапником показано, что необходимо выбирать плоскость с наибольшим отступом (maximum margin):

$$\frac{2}{\langle w, w \rangle} \to \max$$

- Нахождение такой плоскости – решение задачи оптимизации

$$\frac{1}{2}\|w\|^2 \to \min$$

при условии

$$\forall i \quad f(x_i)(\langle w, x_i \rangle + w_0) \geqslant 1$$

- Линейная разделимость очень редко наблюдается на реальных данных

Некорректность задачи в случае плохой разделимости: регуляризованной SVM

- Противоречие: разделить объекты надо – разделить объекты невозможно
- Регуляризация задачи возможно позволит решить задачу с наименьшими «потерями»
- В 1995 году Вапник и Кортез предложили решать следующую задачу минимизации:

$$\frac{1}{2}\|w\|^2 + C\sum_i \xi_i \to \min,$$

где ξ_i – отступ объекта x_i от границы

Задача оптимизации: поиск положения разделяющей гиперплоскости

- Минимизация регуляризованного функционала

$$\frac{1}{2}\|w\|^2 + C\sum_i \xi_i \to \min$$

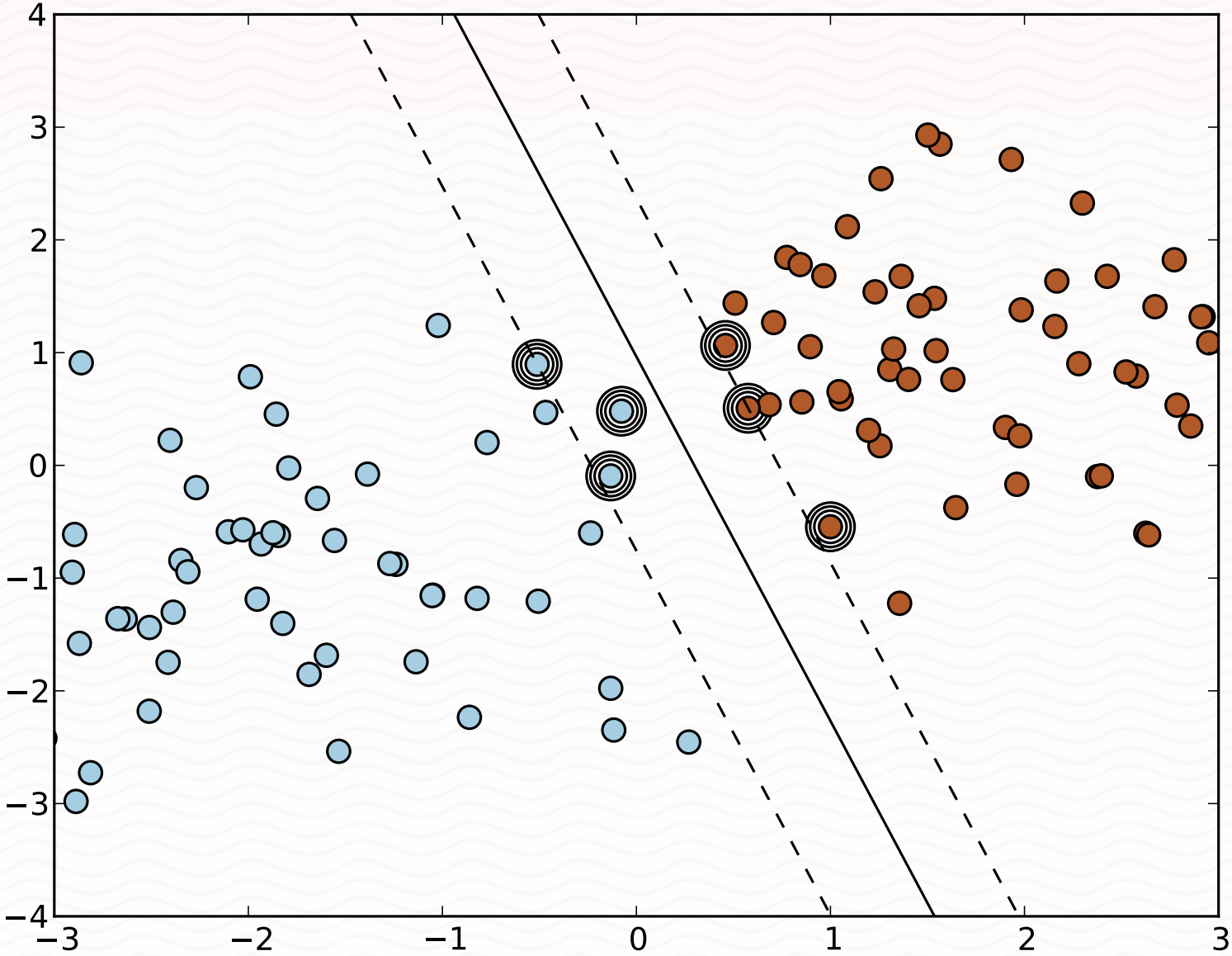
при условии

$$\forall i \quad f(x_i)(\langle w, x_i\rangle + w_0) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

- Положение гиперплоскости определяется только объектами на границе – опорными векторами (для них $\alpha_i \neq 0$)

$$w = \sum_i \alpha_i f(x_i) x_i$$

$$w_0 = -\frac{\max_{i, f(x_i)=-1}\langle w, x_i\rangle + \min_{i, f(x_i)=1}\langle w, x_i\rangle}{2}$$


4
3
2
1
0
−1
−2
−3
−4
−3
−2
−1
0
1
2
3

Sequential Minimal Optimization (SMO)

- Джон Платт из Microsoft Research в 1998 году разработал эффективный алгоритм поиска минимума функционала Лагранжа
- Задача квадратичного программирования разбивается на меньшие, решение которых ищется аналитически
- Алгоритм является итеративным и выполняет последовательные приближения по компонентами α_i
- В результате скорость обучения классификации возрастает вплоть до тысячи раз (sic!)

Некорректность задачи в случае неразделимости: спрямление пространства

- Разделяющая гиперплоскость для многих выборок совершенно бесполезна, даже регуляризация не позволяет найти «хорошую» гиперплоскость
- Любое пространство можно преобразовать в такое, в котором вся выборка линейно разделима – проблема лишь в том, как его найти
- Отображение $\psi : \mathcal{X} \to \mathcal{H}$, повышающее размерность пространства объектов, может разделить выборку
- Вместо преобразования пространство достаточно изменить скалярное произведение (kernel trick):

$$\langle x, y \rangle = K(x, y) = \psi(x)\psi(y)$$

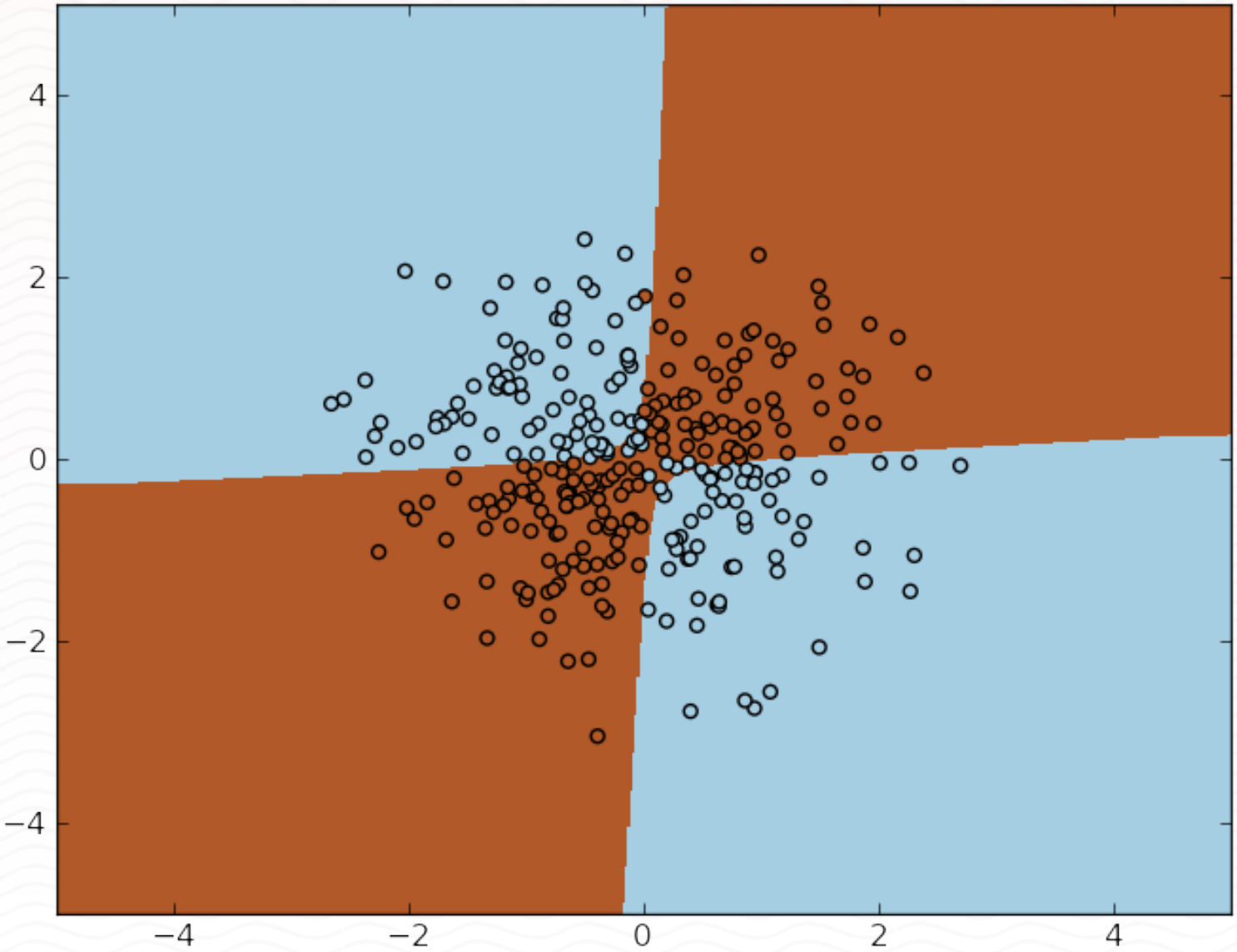
- С помощью такого преобразования можно работать даже в бесконечномерных пространствах

Выбор ядра спрямления

- Ядро – функция, удовлетворяющее теореме Мерсера
- Примеры ядер:
  1. Полиномиальное ядро $K_{\alpha,c,d}(x,y) = (\alpha x \cdot y + c)^d$
  2. Гауссовское (RBF) ядро $K_\beta(x,y) = \exp(-\beta \|x-y\|^2)$
  3. Сигмоидное ядро $K_{\alpha,c}(x,y) = \text{th}(\alpha x \cdot y + c)$
  4. Ядро Коши $K_\sigma(x,y) = \dfrac{1}{1 + \frac{\|x-y\|^2}{\sigma}}$
  5. $K(x,y) = \sum_i \min(x_i, y_i)$
- Универсальных строгих правил для выбора ядра не существует
- Для подавляющего числа задач предпочтительнее RBF, но в некоторых случаях полиномиальное ядро лучше
- Ядро может быть построено с помощью композиции нескольких ядер:
  - Неотрицательная линейная комбинация ядер – тоже ядро
  - Произведение ядер также ядро

Нелинейный SVM: выборка XOR

C=10000, RBF kernel, $\nu = 0.35$



Многоклассовая классификация с помощью SVM

- Ранее рассматривалась только бинарная классификация, $\mathcal{C} = \{-1, +1\}$, однако в реальном анализе данных гораздо чаще встречается многоклассовая классификация
- OVA (one versus all, один против всех) – строится столько же классификаторов, сколько классов, каждый из которых противопоставляет бинарно один класс всем остальным
- При классификации объекта выбирается тот классификатор, в котором объект классифицируется «увереннее», с максимальным отступом от гиперплоскости

Эффективность алгоритма

- Построенный алгоритм не требует никаких вычислений, связанных с объектами обучающей выборки
- Классификация выполняется достаточно быстро
- Само построение алгоритма классификации выполняется за существенное время и сильно зависит от реализации
- SVM – один из самых точных классификаторов
- На практике желательна нормировка входных данных
- Поиск наилучших значений параметров производится с помощью кросс-валидации или других процедур

What to know?

- Линейный SVM
- Регуляризация SVM при плохой линейной разделимости
- Kernel trick
- Sequential Minimal Optimization
- Мультиклассовая классификация с помощью SVM

Источники

1. Hastie T. et al, "The Elements of Statistical Learning: Data Mining, Inference and Prediction"
2. Tom Mitchell "Machine learning"
3. Золотых Н.Ю., «Машинное обучение»
4. Воронцов К.В., «Метрические алгоритмы классификации»
5. Воронцов К.В., «Линейные алгоритмы классификации»
6. Burges C., "A Tutorial on Support Vector Machines for Pattern Recognition"
7. Moore A., "Support Vector Machines"
8. Hsu C.-W. et al, "A Practical Guide to Support Vector Classification"
9. Platt J. "Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines"
10. Cristianini N. "Support Vector and Kernel Machines"