

支持向量机（上）

Machine Learning Engineer

机器学习工程师

讲师：Ivan

目录

CONTENTS

01

二分类线性可分支持向量机

02

二分类线性不可分支持向量机

03

多分类支持向量机

04

SVM 工具包介绍



01 二分类线性可分支持向量机

1.1

线性模型

1.2

最大间隔分类器

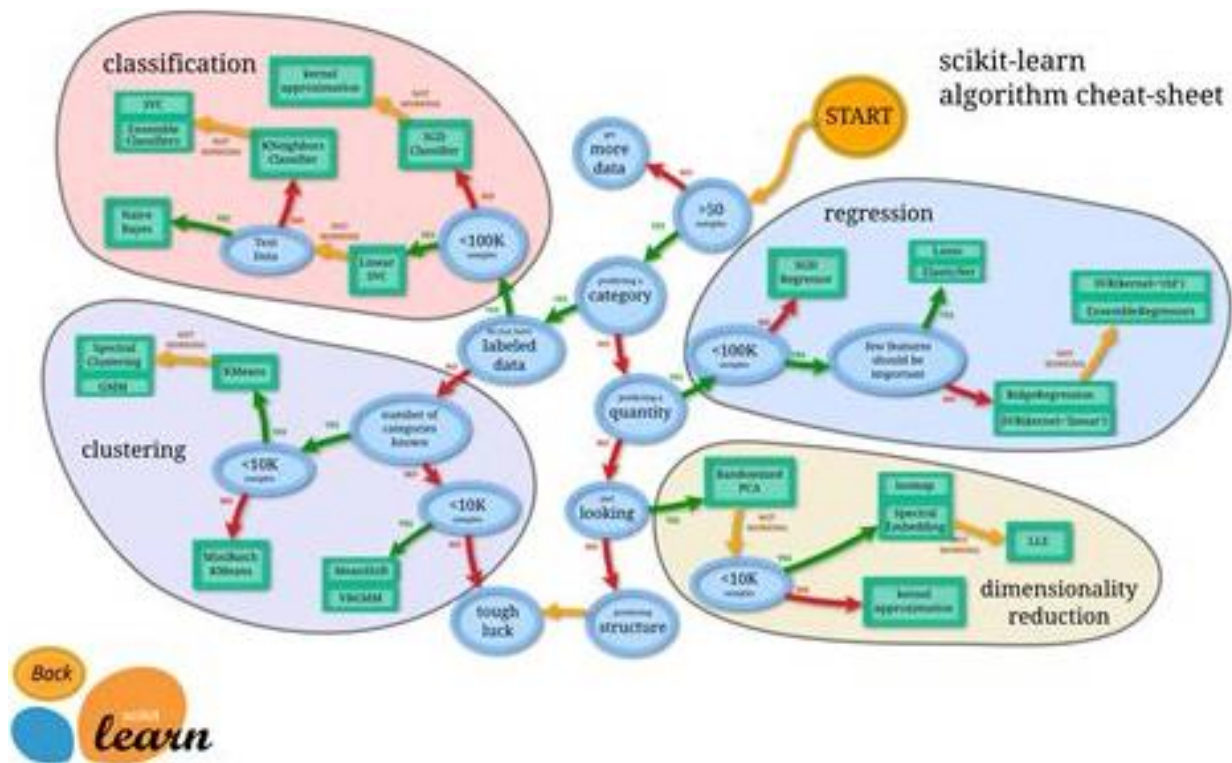
1.3

与逻辑回归的对比

什么是支持向量机(Support Vector Machine, SVM)?

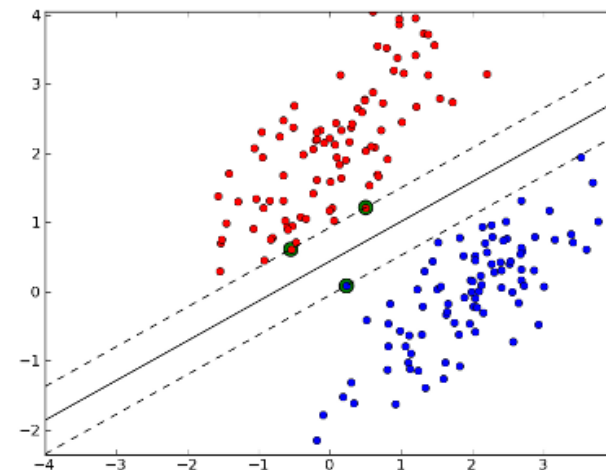
机器学习算法分类的几个指标：

1. 分类问题？回归问题？
2. 有监督？无监督？
3. 线性模型？非线性模型？
4. 特征离散？特征连续？
5. 凸优化？非凸优化？

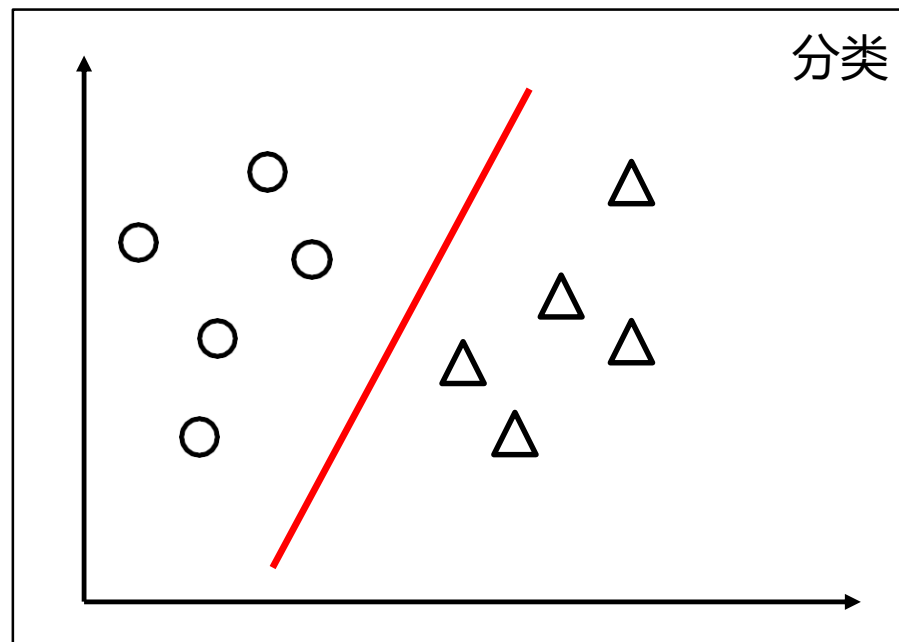


什么是支持向量机(Support Vector Machine, SVM)?

- **基本形式**：有监督二分类线性分类模型
- **扩展形式**：
 1. 有监督二分类非线性分类模型
 2. 有监督多分类（线性/非线性）分类模型
 3. 有监督线性回归模型（Support Vector Regression, SVR）
 4. 基于核函数的SVM/SVR



1.1 线性模型

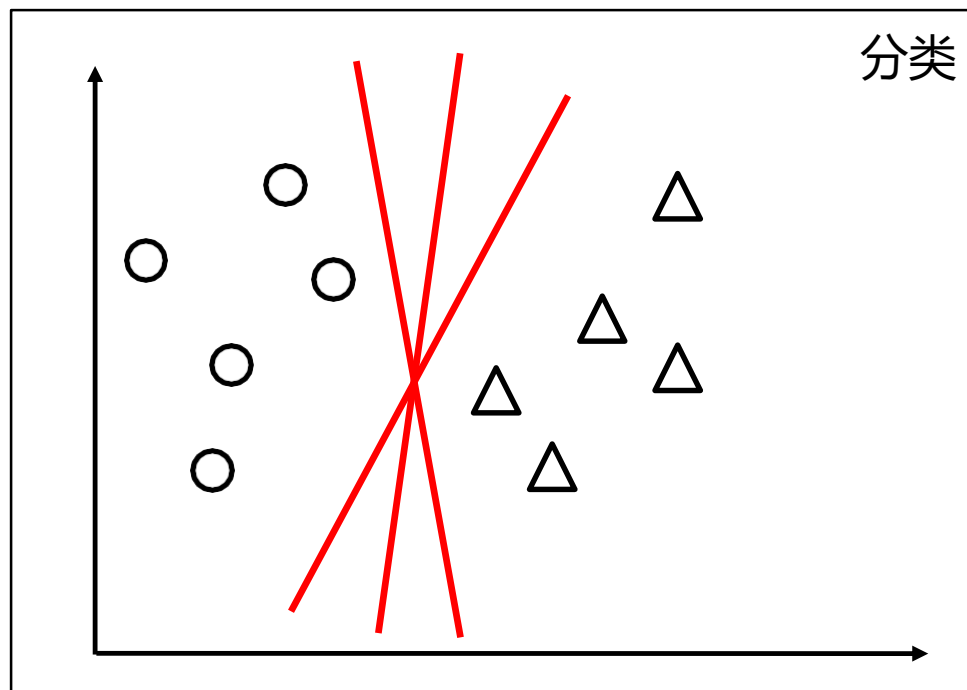


线性模型(linear model)：特征的线性组合进行分类

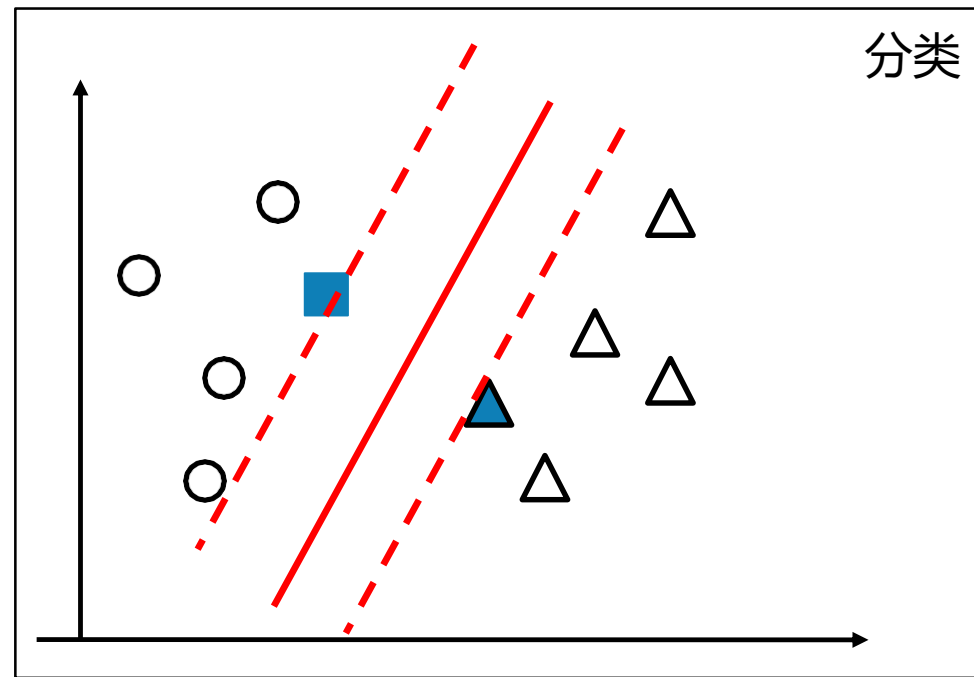
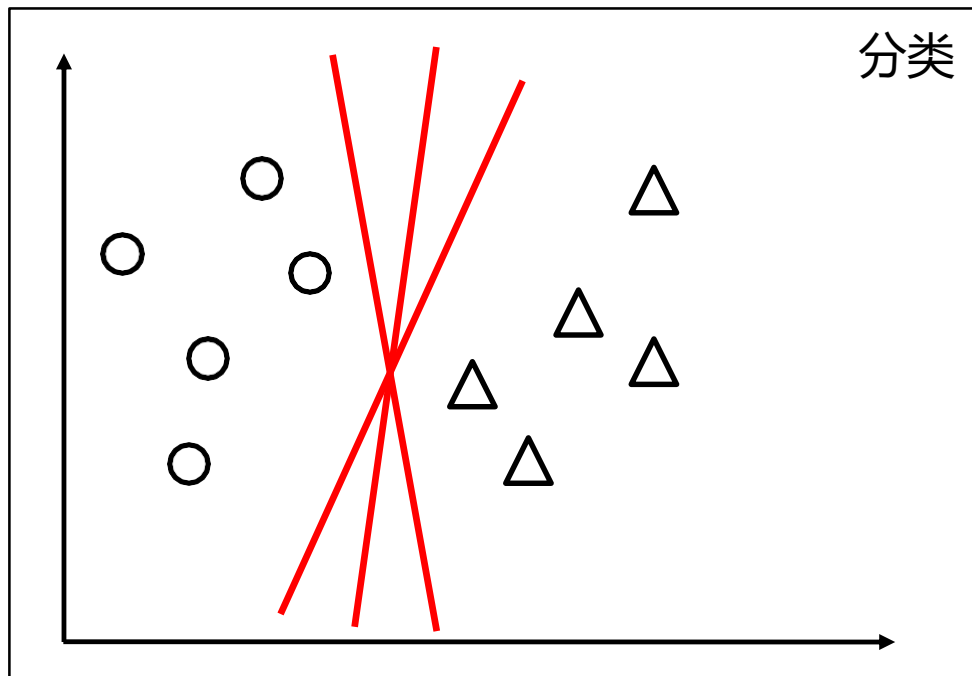
模型表达式： $f(x) = w_1x_1 + w_2x_2 + \cdots + w_dx_d + b = w^Tx + b$

例如：逻辑回归 (Logistic Regression)

1.2 最大间隔分类器



数据线性可分：如何从多个线性分类其中进行选择？



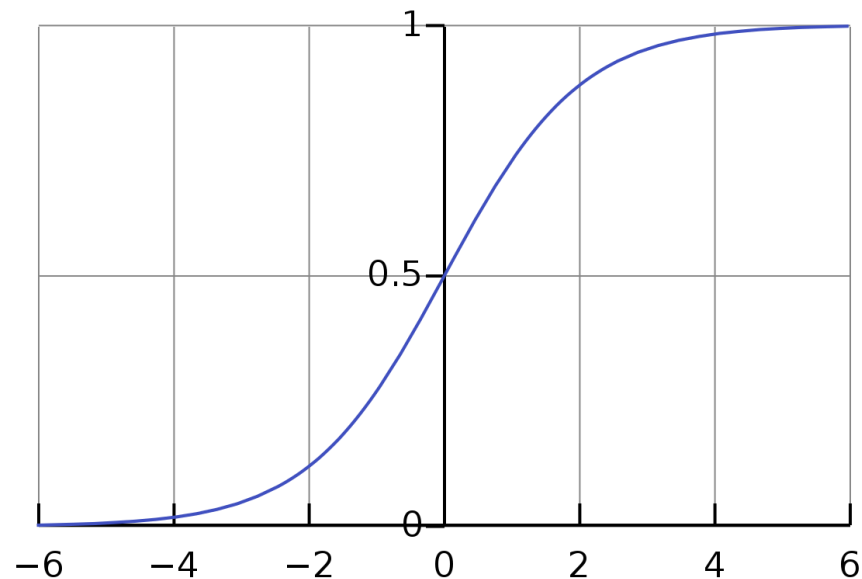
核心想法：最大间隔分类器

- 离数据最远的线性分类器最安全
- 离数据最远的线性分类器最容易泛化
- SVM是线性模型中一种
- 虚线上的点被称为支持向量

回顾逻辑回归：

预测函数：

$$y = \frac{1}{1 + e^{-w^T x}}$$



- 输出实际上是概率 $p(y = 1|x)$
- $w^T x \rightarrow \infty, p(y = 1|x) \rightarrow 1$; $w^T x$ 越大，输出越靠近1
- $w^T x \rightarrow -\infty, p(y = 1|x) \rightarrow 0$; $w^T x$ 越小，输出越靠近0

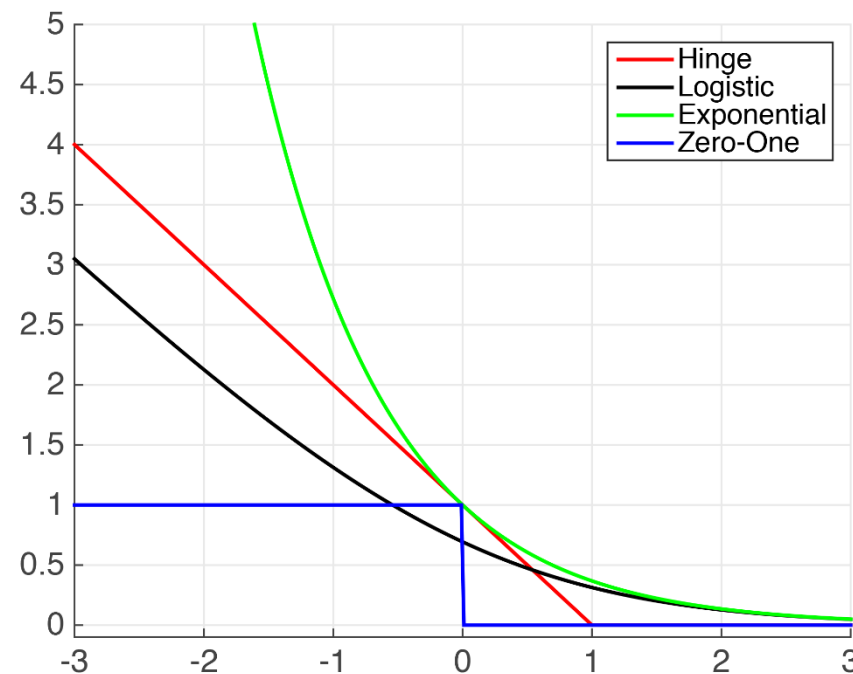
1.3 与逻辑回归的对比

回顾逻辑回归：

损失函数：

$$L(x, y) = -y \log p(y = 1|x) - (1 - y) \log(1 - p(y = 1|x))$$

- 关于模型参数 w ，这是一个凸函数
- 如果将 $y w^T x$ 看作自变量，损失函数：
- 称为Logistic 损失函数
- 等价于 最大似然函数/交叉熵损失函数



1.3 与逻辑回归的对比

回顾逻辑回归：

正则项：

$$|w|_2^2$$

- 关于模型参数 w ，这也是一个凸函数
- L2 正则化
- 防止模型过拟合

1.3 与逻辑回归的对比

回顾逻辑回归：

优化目标函数：

$$\ell(x, y) = \frac{1}{n} \sum_{i=1}^n -y_i \log p(y_i = 1|x_i) - (1 - y_i) \log(1 - p(y_i = 1|x_i)) + \frac{\lambda}{2} |w|_2^2$$

- 模型要点：

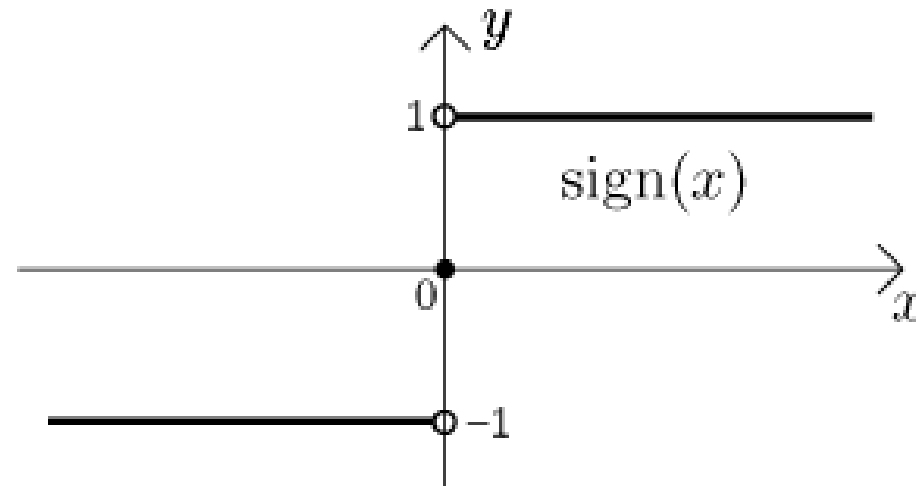
- 模型形式：线性模型
- 损失函数：最大似然/交叉熵/Logistic 损失函数
- 正则项：L2正则防止过拟合

1.3 与逻辑回归的对比

支持向量机：

预测函数：

$$y = \text{sgn}(w^T x)$$



- 输出不是概率，而是+1或者-1，代表二分类
- 预测输出和 $w^T x$ 的绝对值大小无关，只和 $w^T x$ 的符号有关
- $w^T x$ 几何意义：正比于 x 到平面的有向距离

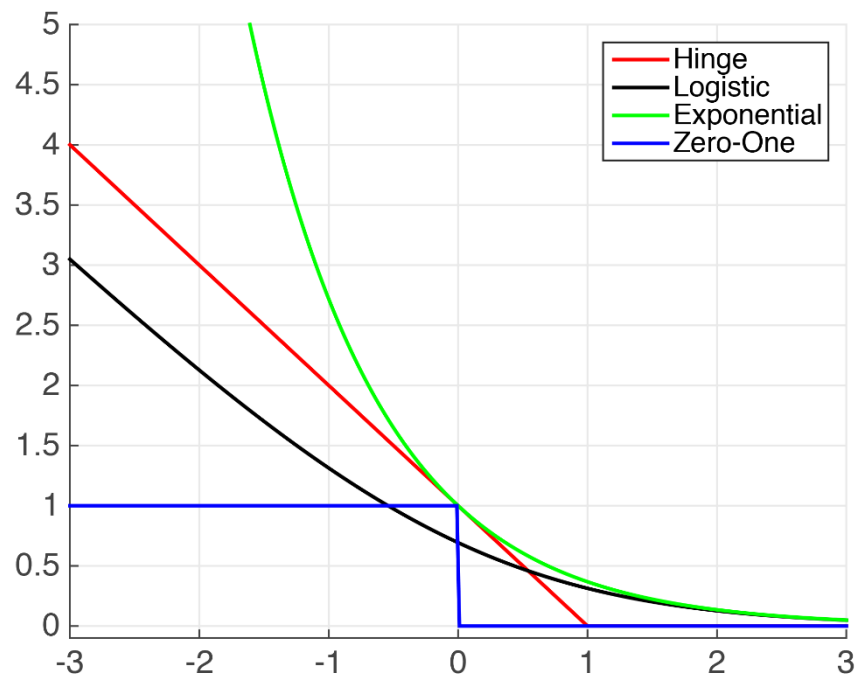
1.3 与逻辑回归的对比

支持向量机：

损失函数：

$$L(x, y) = \max\{0, 1 - yw^T x\}$$

- 关于模型参数 w ，这是一个凸函数
- 如果将 $yw^T x$ 看作自变量，损失函数:
- 称为Hinge (铰链) 损失函数
- 当 $yw^T x \rightarrow -\infty$ 时，近似于Logistic损失函数



1.3 与逻辑回归的对比

支持向量机：

正则项：

$$|w|_2^2$$

- 关于模型参数 w ，这是一个凸函数
- L2 正则化
- 防止模型过拟合
- 在SVM中， $1/|w|_2^2$ 对应于分类决策面的间隔

1.3 与逻辑回归的对比

支持向量机：

优化目标函数：

$$\ell(x, y) = \frac{1}{n} \sum_{i=1}^n \max\{0, 1 - y_i w^T x_i\} + \frac{\lambda}{2} |w|_2^2$$

• 模型要点：

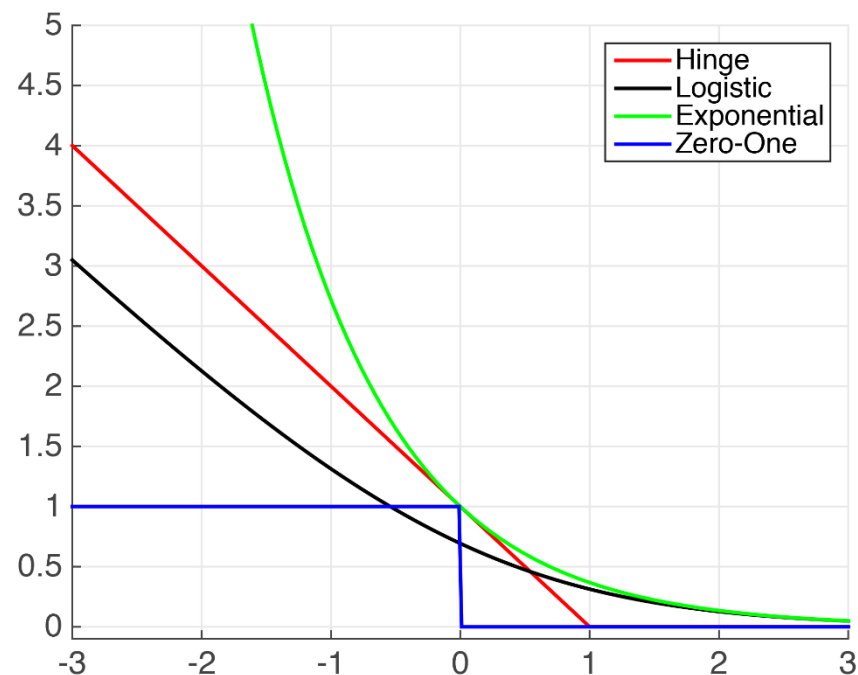
- 模型形式：线性模型
- 损失函数：Hinge 损失函数
- 正则项：L2正则防止过拟合，最大化分类间隔

1.3 与逻辑回归的对比

综合对比：

	支持向量机	逻辑回归
模型	二分类决策线性模型	二分类概率线性模型
正则化	L2 正则化	L2 正则化
损失函数	Hinge 损失函数	Logistic 损失函数
原始优化问题	凸优化，SMO算法	凸优化，梯度下降

要点总结



1.1

SVM：间隔最大线性分类器

1.2

Hinge 损失函数 vs Logistic 损失函数

1.3

确定性模型 vs 概率性模型

1.4

线性SVM的优化目标函数



02

二分类线性不可分支持向量机

2.1

线性支持向量机的几何解释

2.2

松弛变量

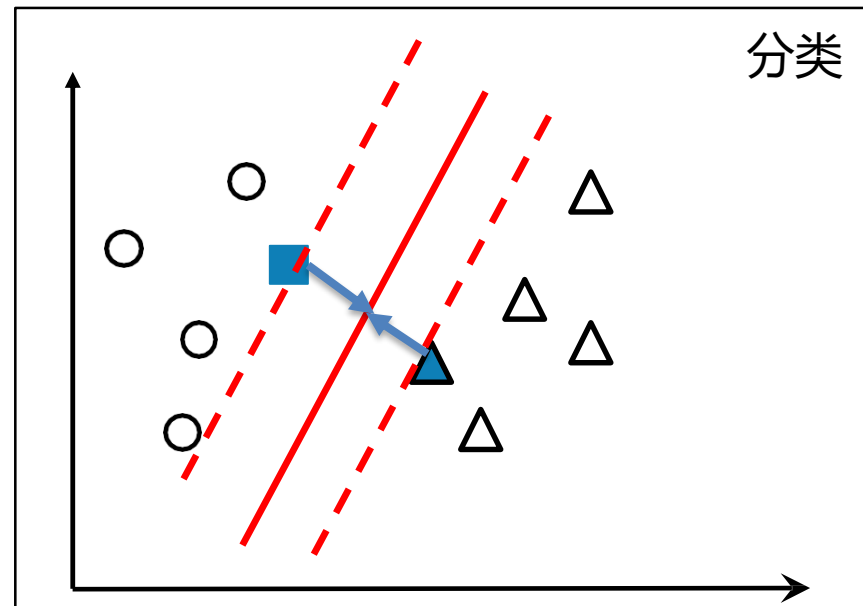
2.3

线性不可分支持向量机

如何刻画/描述数据到分类器的间隔？

x_i 到直线 $y = w^T x$ 的距离： $\frac{|w^T x_i|}{|w|_2}$

$y = w^T x$ 在数据集上的间隔： $\min_i \frac{|w^T x_i|}{|w|_2}$



注意到 $\frac{|w^T x_i|}{|w|_2}$ 关于 w 是齐次的，所以可以找到一个 w ，使得 $\min_i |w^T x_i| = 1$

线性支持向量机的优化目标函数

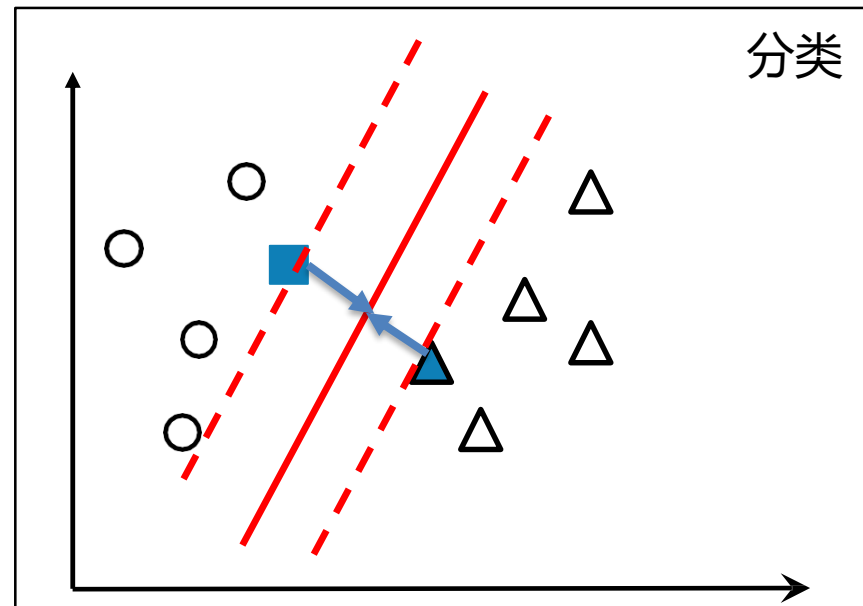
- 所有数据点被正确分类：

$$y_i w^T x_i \geq 1, \forall i \in [n]$$

- 最大化分类间隔：

$$\max \frac{1}{|w|_2} \Leftrightarrow \min |w|_2^2$$

思考一下为什么是这两个表达式？



线性支持向量机的优化目标函数

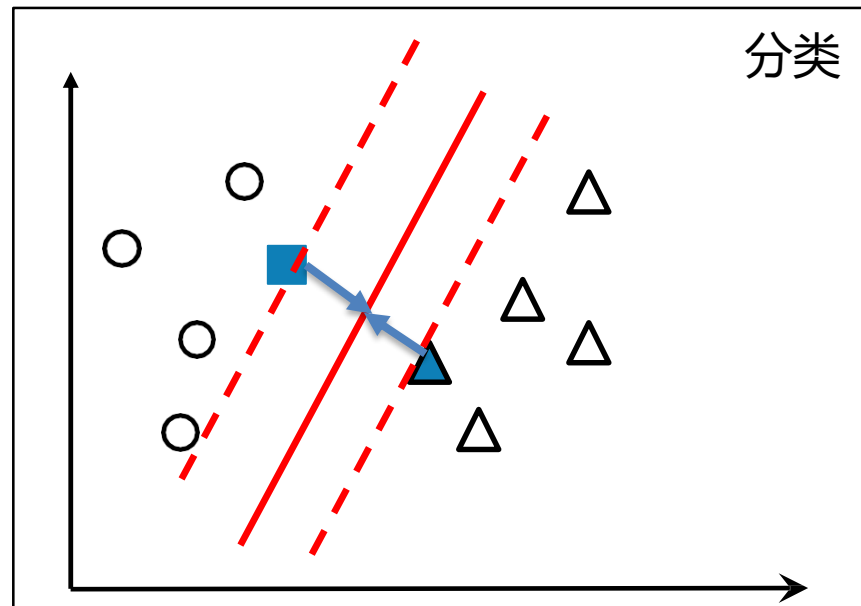
- 所有数据点被正确分类
- 最大化分类间隔

约束优化形式：

$$\begin{aligned} \min \quad & |w|_2^2 \\ \text{s.t.} \quad & y_i w^T x_i \geq 1, \forall i \in [n] \end{aligned}$$

非约束优化形式：

$$\min \ell(x, y) = \frac{1}{n} \sum_{i=1}^n \max\{0, 1 - y_i w^T x_i\} + \frac{\lambda}{2} |w|_2^2$$



线性支持向量机的优化目标函数

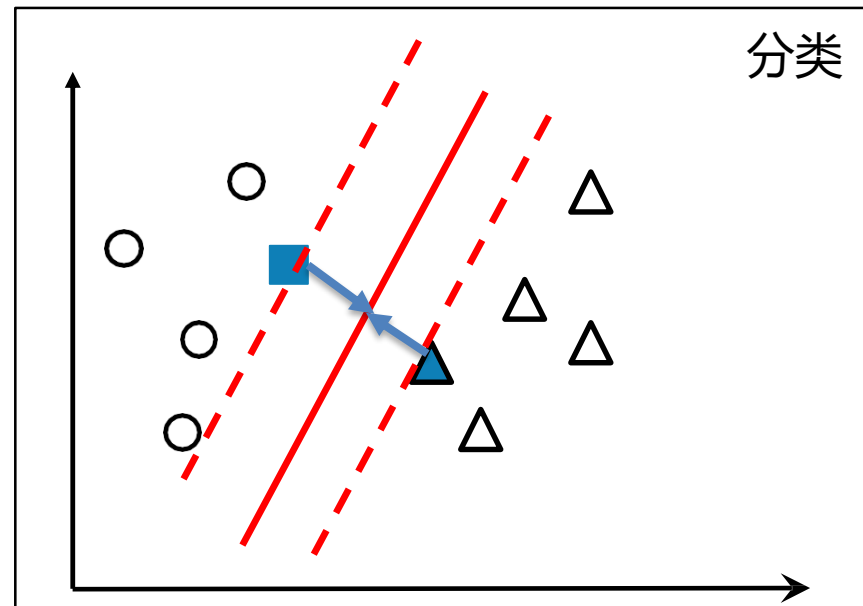
- 所有数据点被正确分类
- 最大化分类间隔

约束优化形式：

$$\begin{aligned} \min \quad & |w|_2^2 \\ \text{s.t.} \quad & y_i w^T x_i \geq 1, \forall i \in [n] \end{aligned}$$

非约束优化形式：

$$\min \ell(x, y) = \frac{1}{n} \sum_{i=1}^n \max\{0, 1 - y_i w^T x_i\} + \frac{\lambda}{2} |w|_2^2$$



SVM的约束优化问题等价于非约束优化问题！

2.2 松弛变量

如何扩展到线性不可分？几何思想：

$$y_i w^T x_i \geq 1, \quad \forall i \in [n]$$

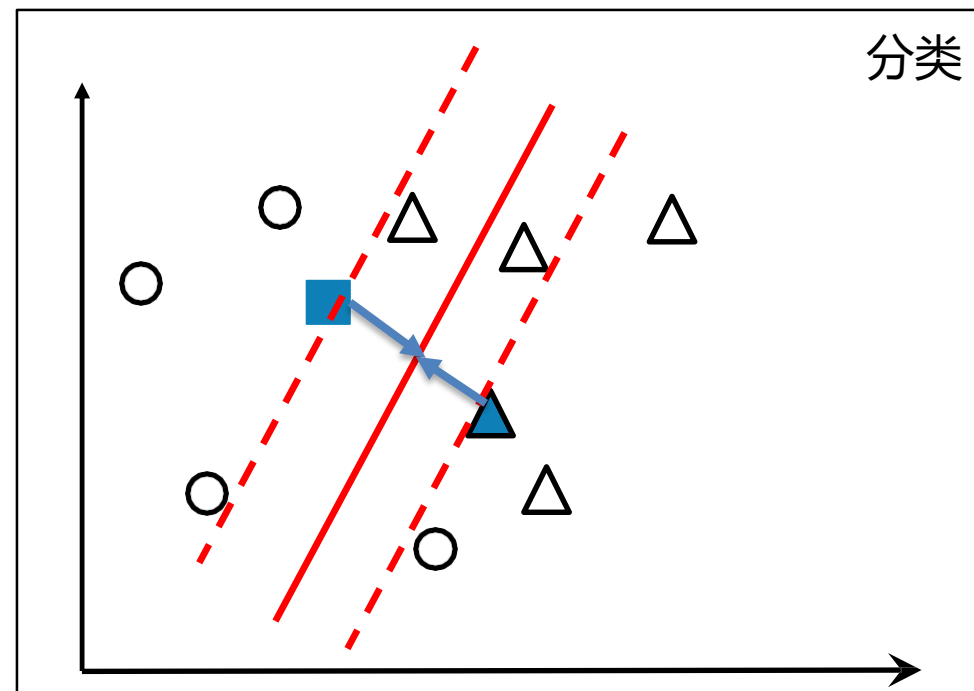
要求所有数据点到分类器的距离 > 1

引入松弛变量 ϵ_i ：

允许线性不可分的点到分类器距离 < 0 ：

$$y_i w^T x_i \geq 1 - \epsilon_i,$$

$$\epsilon_i \geq 0, \quad \forall i \in [n]$$



2.2 松弛变量

如何扩展到线性不可分？几何思想：

$$y_i w^T x_i \geq 1, \quad \forall i \in [n]$$

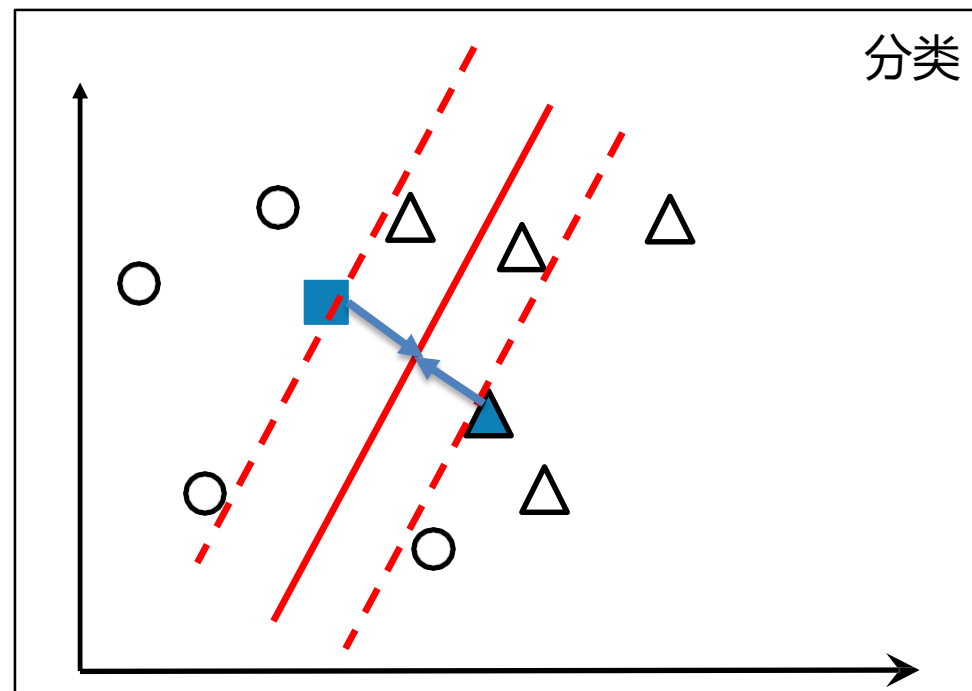
要求所有数据点到分类器的距离 > 1

引入松弛变量 ϵ_i ：

允许线性不可分的点到分类器距离 < 0 ：

$$y_i w^T x_i \geq 1 - \epsilon_i,$$

$$\epsilon_i \geq 0, \quad \forall i \in [n]$$



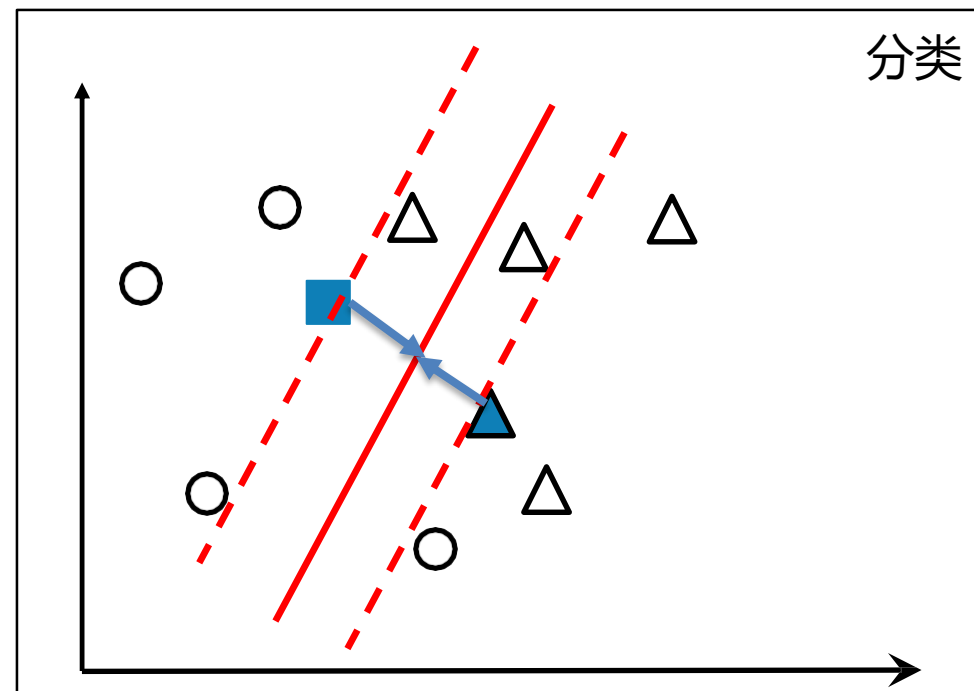
- 若 $\epsilon_i = 0$ ，那么 x_i 被正确分类，并且分类间隔 > 1
- 若 $0 < \epsilon_i \leq 1$ ，那么 x_i 被正确分类，分类间隔大于0，但是 < 1
- 若 $\epsilon_i > 1$ ，那么 x_i 被错误分类，分类间隔 < 0

约束优化形式：

$$\begin{aligned} \min \quad & \frac{\lambda}{2} |w|_2^2 + \frac{1}{n} \sum_i \epsilon_i \\ \text{s.t.} \quad & y_i w^T x_i \geq 1 - \epsilon_i, \forall i \in [n] \\ & \epsilon_i \geq 0 \end{aligned}$$

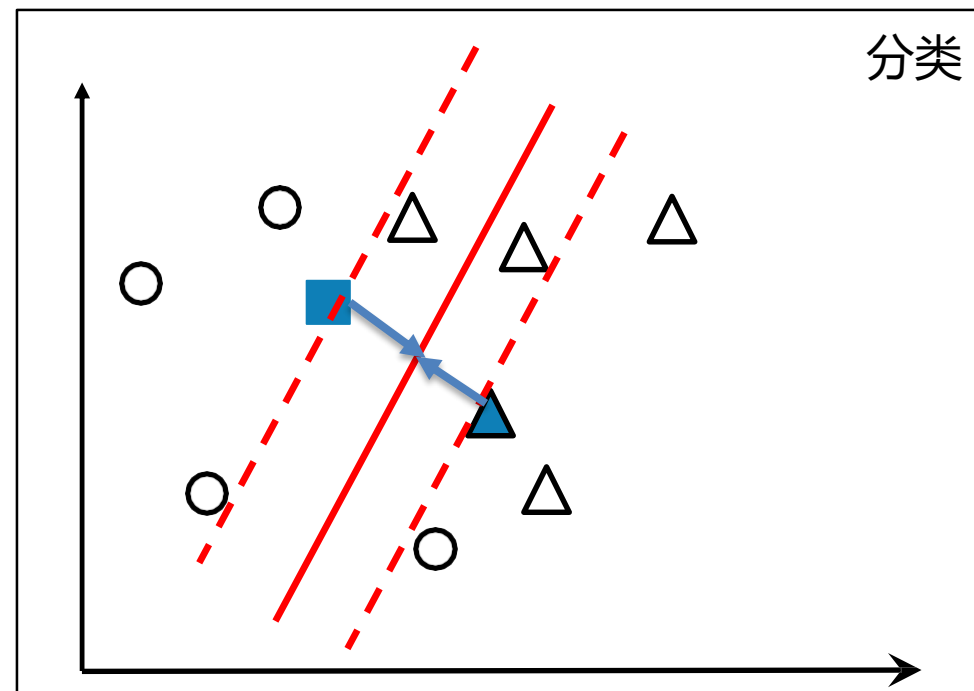
非约束优化形式：

$$\min \ell(x, y) = \frac{1}{n} \sum_{i=1}^n \max\{0, 1 - y_i w^T x_i\} + \frac{\lambda}{2} |w|_2^2$$

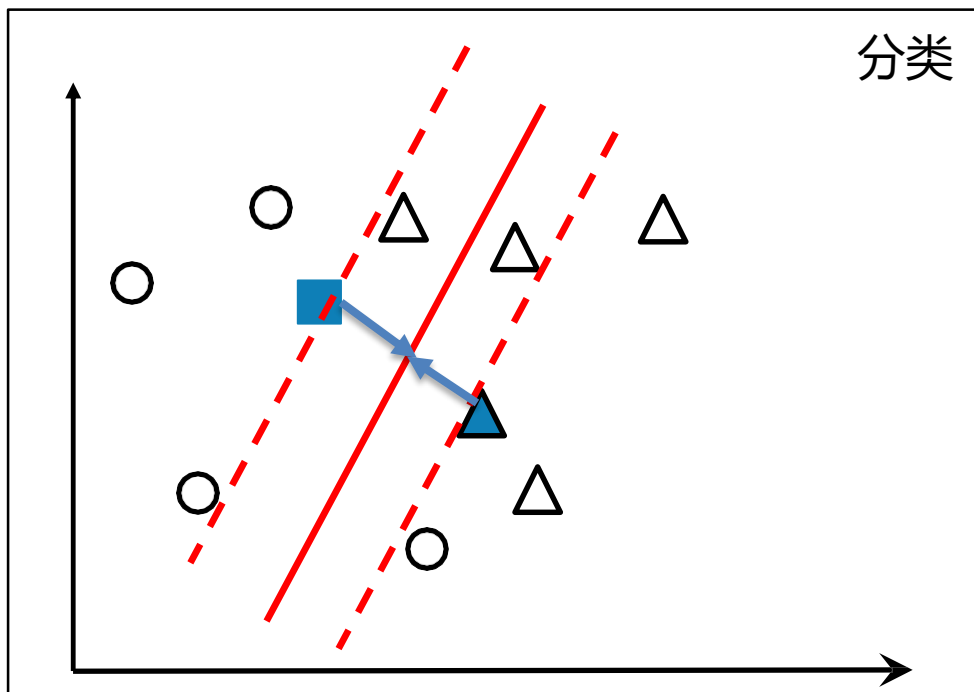


$$\min \frac{1}{n} \sum_{i=1}^n \max\{0, 1 - y_i w^T x_i\} + \frac{\lambda}{2} |w|_2^2$$

- 线性可分与线性不可分统一的形式
- SVM是数据自适应的
- 本质是一个凸优化问题，二次规划
- 可以转换为二次规划一般形式求解，也可以使用梯度下降法求解



要点总结



1.1

SVM的几何意义与解释

1.2

松弛变量的几何意义

1.3

SVM的优化问题，凸二次规划问题

1.4

SVM 的数据自适应性质



03

多分类支持向量机

3.1

One vs One 方法

3.2

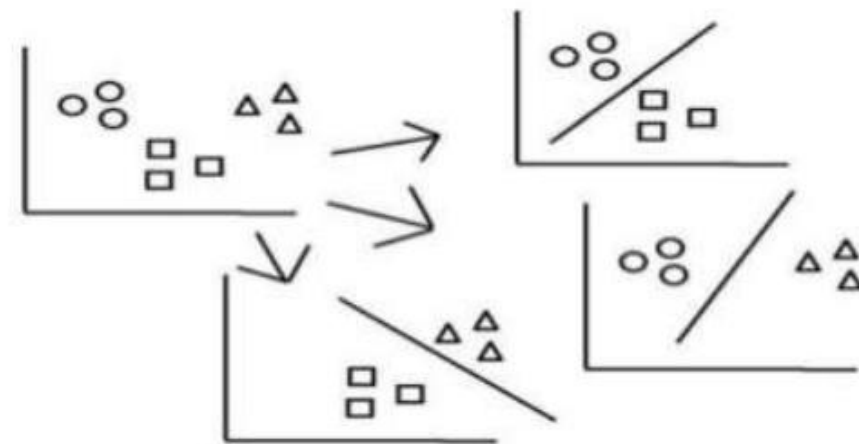
One vs All 方法

One vs One 方法

SVM 是二类分类器，如何扩展到k分类 ($k > 2$) ?

- One vs One – 即1对1
- 训练时刻：训练 $k(k-1)/2$ 个二分类器, $f_{i,j}(x)$
- $f_{i,j}(x)$ 预测 x 是第i类的可能性大于第j类的可能性
- 测试时刻：？

One-vs-One (OVO)

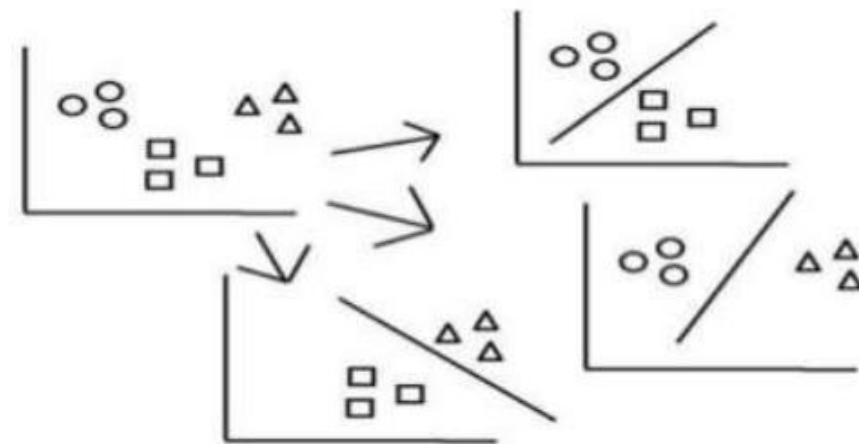


One vs One 方法

SVM 是二类分类器，如何扩展到k分类 ($k > 2$) ?

- One vs One – 即1对1
- 训练时刻：训练 $k(k-1)/2$ 个二分类器, $f_{i,j}(x)$
- $f_{i,j}(x)$ 预测 x 是第i类的可能性大于第j类的可能性
- 使用全部的 $k(k-1)/2$ 个分类器
- 预测 x 为胜利次数最多的类别

One-vs-One (OVO)



One vs One 方法

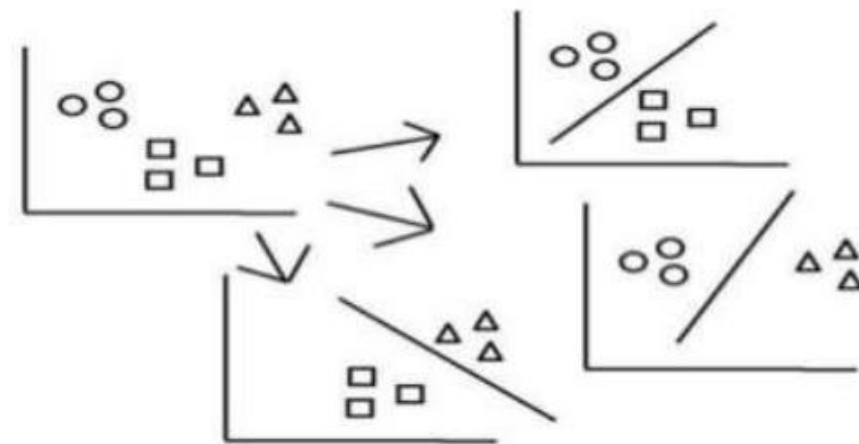
优点：

- 适用性广，LibSVM默认的实现方法
- 对于所有二类分类器都可使用，概率/非概率 分类器均可

缺点：

- 训练时刻计算复杂度高： $O(k^2)$
- 测试时刻计算复杂度高： $O(k^2)$

One-vs-One (OVO)

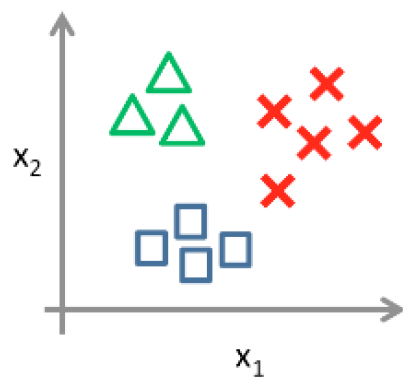





3.2 One vs All 方法

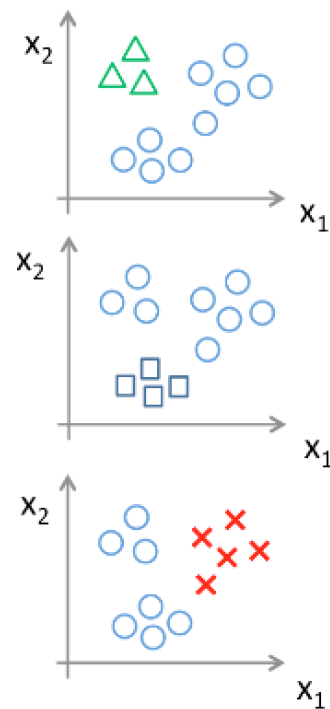
SVM 是二类分类器，如何扩展到k分类 ($k > 2$)？

- One vs All – 即1对多
- 训练时刻：训练k个二分类器, $f_i(x)$
- $f_i(x)$ 预测 x 属于第i类的分数
- 测试时刻：？

One-vs-all (one-vs-rest):



Class 1: 
Class 2: 
Class 3: 

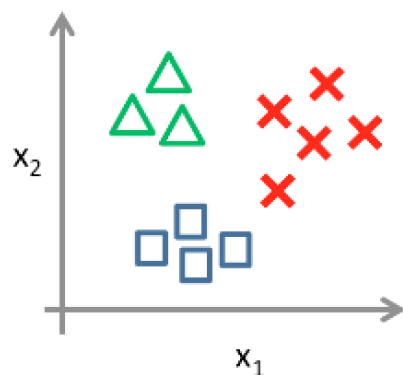





3.2 One vs All 方法

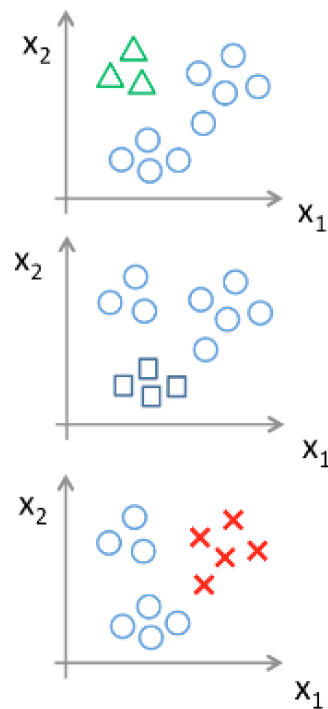
SVM 是二类分类器，如何扩展到k分类 ($k > 2$)？

- One vs All – 即1对多
- 训练时刻：训练k个二分类器, $f_i(x)$
- $f_i(x)$ 预测 x 属于第i类的分数
- 使用全部k个分类器
- 预测 x 为得分最高的类别

One-vs-all (one-vs-rest):



Class 1: 
Class 2: 
Class 3: 



3.2 One vs All 方法

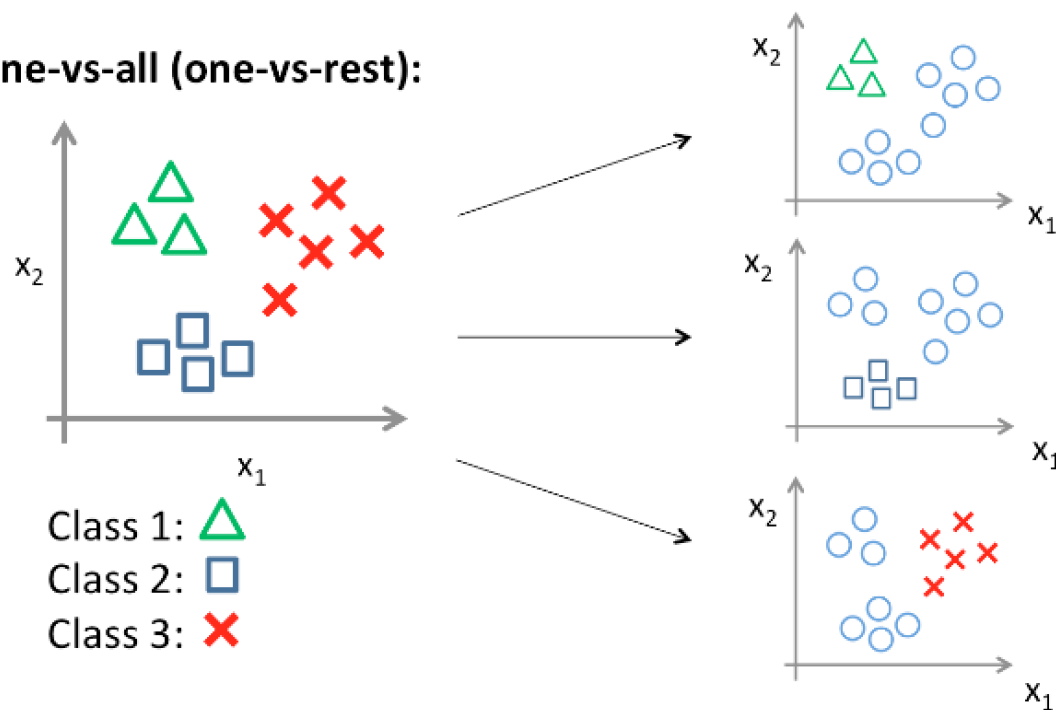
优点：

- 适用性有限，要求 $f_i(x)$ 表示 x 属于第 i 类的一个打分
- 多使用于概率分类器，例如逻辑回归

缺点：

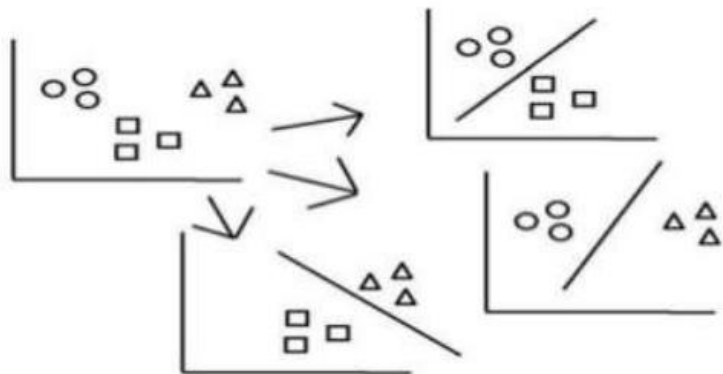
- 训练时刻计算复杂度低： $O(k)$
- 测试时刻计算复杂度低： $O(k)$

One-vs-all (one-vs-rest):



03 多分类支持向量机

One-vs-One (OVO)



要点总结

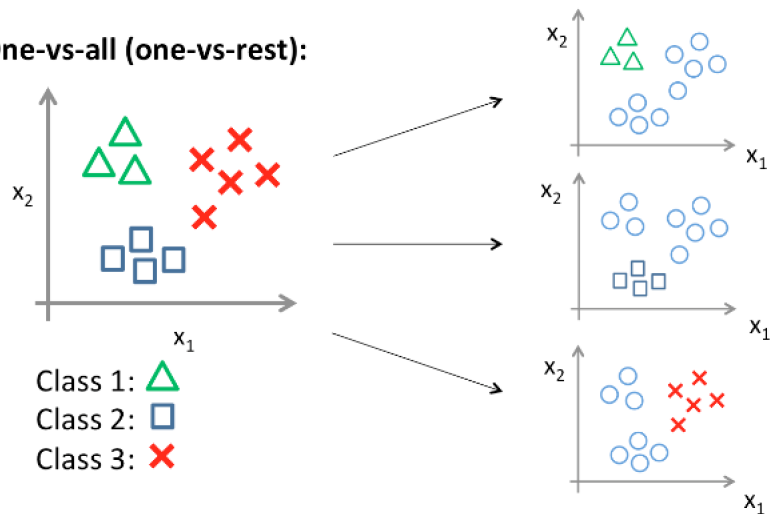
3.1

One vs One 方法 (SVM)

3.2

One vs All 方法 (LR)

One-vs-all (one-vs-rest):





04 SVM 工具包介绍

4.1

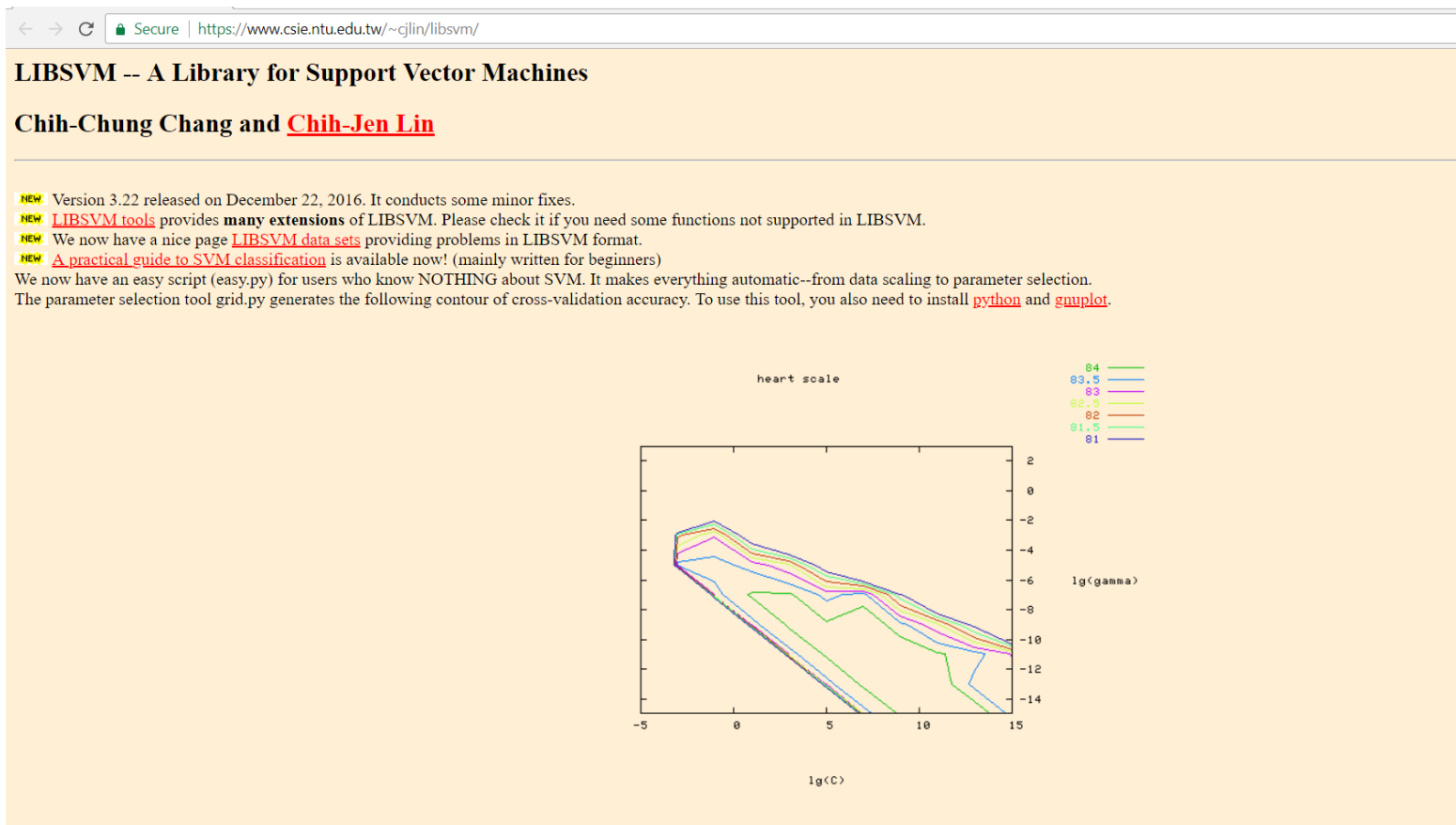
LibSVM

4.2

SVMLight

4.3

Scikit-learn



- 支持多种编程语言，提供命令行使用接口
- 支持线性/非线性SVM，支持分类和回归
- 目前最流行并且高效的SVM开源实现



- C/C++实现
- 支持基于SVM的结构预测以及半监督SVM
- 支持基于SVM的排序学习算法

Overview

SVM^{light} is an implementation of Support Vector Machines (SVMs) in C. The main features of the program are the following:

- fast optimization algorithm
 - working set selection based on steepest feasible descent
 - "shrinking" heuristic
 - caching of kernel evaluations
 - use of folding in the linear case
- solves classification and regression problems. For multivariate and structured outputs use [SVM^{struct}](#).
- solves ranking problems (e. g. learning retrieval functions in [STRIVER](#) search engine).
- computes XiAlpha-estimates of the error rate, the precision, and the recall
- efficiently computes Leave-One-Out estimates of the error rate, the precision, and the recall
- includes algorithm for approximately training large transductive SVMs (TSVMs) (see also [Spectral Graph Transducer](#))
- can train SVMs with cost models and example dependent costs
- allows restarts from specified vector of dual variables
- handles many thousands of support vectors
- handles several hundred-thousands of training examples
- supports standard kernel functions and lets you define your own
- uses sparse vector representation

The screenshot shows the Scikit-learn website's SVM module page. The browser address bar displays 'scikit-learn.org/stable/modules/svm.html'. The page features a navigation bar with links for Home, Installation, Documentation, and Examples, along with a search bar. On the left, a sidebar contains navigation links for previous, next, and up versions, as well as a list of topics under '1.4. Support Vector Machines'. The main content area is titled '1.4. Support Vector Machines' and includes a description of SVMs, their advantages and disadvantages, and a section for '1.4.1. Classification'.

scikit-learn

Home Installation Documentation Examples

Google Custom Search Search

Previous 1.3. Kernel f... Next 1.5. Stochastic... Up 1. Supervised...

scikit-learn v0.19.1
Other versions

Please cite us if you use the software.

1.4. Support Vector Machines

1.4.1. Classification

- 1.4.1.1. Multi-class classification
- 1.4.1.2. Scores and probabilities
- 1.4.1.3. Unbalanced problems

1.4.2. Regression

1.4.3. Density estimation, novelty detection

1.4.4. Complexity

1.4.5. Tips on Practical Use

1.4.6. Kernel functions

- 1.4.6.1. Custom Kernels
 - 1.4.6.1.1. Using Python functions as kernels
 - 1.4.6.1.2. Using the Gram matrix
 - 1.4.6.1.3. Parameters of the RBF Kernel

1.4.7. Mathematical formulation

- 1.4.7.1. SVC
- 1.4.7.2. NuSVC
- 1.4.7.3. SVR

1.4.8. Implementation details

1.4. Support Vector Machines

Support vector machines (SVMs) are a set of supervised learning methods used for [classification](#), [regression](#) and [outliers detection](#).

The advantages of support vector machines are:

- Effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Versatile: different [Kernel functions](#) can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The disadvantages of support vector machines include:

- If the number of features is much greater than the number of samples, avoid over-fitting in choosing [Kernel functions](#) and regularization term is crucial.
- SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation (see [Scores and probabilities](#), below).

The support vector machines in scikit-learn support both dense (`numpy.ndarray` and convertible to that by `numpy.asarray`) and sparse (any `scipy.sparse`) sample vectors as input. However, to use an SVM to make predictions for sparse data, it must have been fit on such data. For optimal performance, use C-ordered `numpy.ndarray` (dense) or `scipy.sparse.csr_matrix` (sparse) with `dtype=float64`.

1.4.1. Classification

`SVC`, `NuSVC` and `LinearSVC` are classes capable of performing multi-class classification on a dataset.

- Python实现，轻量级，接口简单，适合科研使用
- 支持线性/非线性，分类/回归 SVM



THANK YOU !

Machine Learning Engineer
机器学习工程师微专业