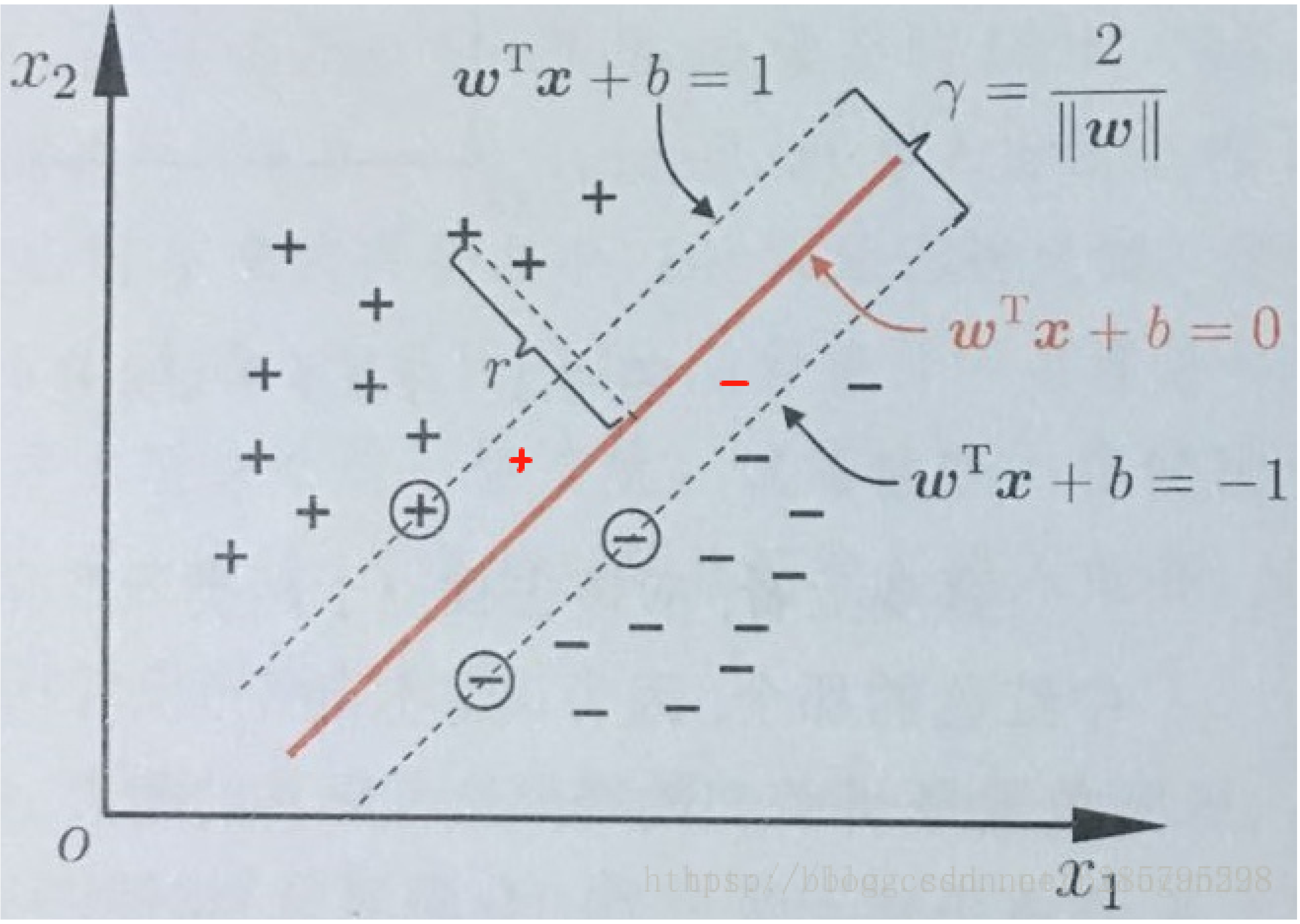
支持向量机（SVM）从入门到放弃再到掌握 - jhoojhooablido - CSDN博客

给定训练样本集$D=(x_1,y_1),(x_2,y_2),...,(x_m,y_m))),y_i\in\{-1,1\}$, 线性分类器基于训练样本D
多。



但我们可以直观感受到，这根红色线代表的超平面抗“扰动”性最好。这个超平面离直线两边的数

在这里，这个超平面可以用函数$f(x)=w^Tx+b$表示。当$f(x)$等于0的时候，x便是位于超平面上的点，而$f(x)$大于0的点对应 y=1 的数据点，$f(x)$小于0的点对应y=-1的点。

为什么是$y_i\in\{-1,1\}$,换句话说，y只能是-1，和1吗？不能是y=-100 表示反例，y=2000表示正例吗？当然可以。`y 只是一个label ，标注为{-1，+1}不过为了描述方便` 。

若y=0表示反例，y=300表示正例，只不过分正类的标准变为$(y-150) * f(x)>0$

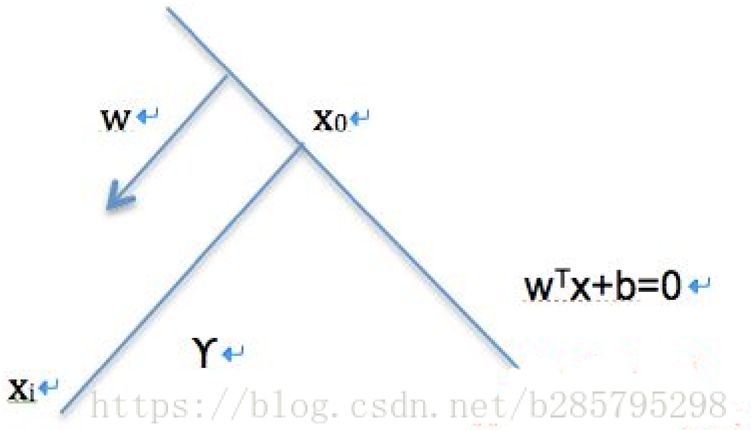
不妨令：

$$\begin{cases} w^Tx_i+b\geqslant+1, y_i=+1; \\ w^Tx_i+b\leqslant-1, y_i=-1 \end{cases}$$

$w^T x_i + b \leq -1, y_i = -1$

为什么可以这么令呢？我们知道，所谓的 支持向量 ，就是使得上式等号成立，即最靠近两条虚边身
就 更加支持 “样本的分类”了。为什么要这么令呢？还是为了计算方便。接着往下看，你一定能

我们可以计算得到空间中任意样本点x到超平面的距离为：$r = \frac{|w^T x + b|}{//w//}$。为什么呢？



如图所示，有：$x = x_0 + r\frac{w}{//w//}$(简单平面几何)
又有：$w^T x_0 + b = 0$，代入上式，求得：$r = \frac{|w^T x + b|}{//w//}$。

因为$y_i \in \{-1, 1\}$,，两个异类支持向量到超平面的距离之和（也称为“间隔”）可表示为：$r =$

很显然， 我们要找到符合这样一个条件的超平面来分开两类数据 ：
这个超平面离两类样本都足够远，也就是使得“间隔”最大。即最终确定的参数w和b,使得r最大

$$\max_{w,b} \frac{2}{//w//}$$
$$s.t. y_i(w^T x_i + b) \geq 1, \quad i = 1, 2, ..., m$$

这等价于

$$\min_{w,b} \frac{1}{2} //w//^2$$
$$s.t. y_i(w^T x_i + b) \geq 1, \quad i = 1, 2, ..., m$$

由此我们得到了SVM的 基本型 。

##凸优化
我们可以看到，上面的基本型目标函数是二次的，约束条件是线性的，这是一个 凸二次规划 问题。
高效。

啥是凸？ 什么是凸优化 ？

凸优化说的是这么一回事情，

$X \subset R^n$ 为一凸集，$f: X \to R$ 为一凸函数，凸优化就是要找出一点 $x^* \in X$,使得任意 $x \in$
可以想象成给我一个凸函数，我要去找到最低点。当然凸优化是一个很大很厉害的领域，在这
解，就好，有兴趣的朋友可以参考凸优化的概念或者Stephen Boyd & Lieven Vandenberg

为啥叫 二次规划 问题呢?

据了解（其实就是知道），目标函数和约束条件都为 变量的线性函数 ，叫做----- 线性规划问题 。
目标函数为 变量的二次函数 和约束条件为 变量的线性函数 ，叫做----- 二次规划问题 。
目标函数和约束条件都为 非线性函数 ，叫做----- 非线性规划问题 。

#对偶问题

对于
$\min_{w,b} \frac{1}{2} // w //^2$
$s.t. y_i(w^T x_i + b) \geq 1$， $i = 1, 2, ..., m$
为了后面的描述方便，记这个式子为 （1）式。

使用 **拉格朗日乘子法** 可以得到其“对偶问题”。
这是拉格朗日对偶性，即，通过给每一个约束条件加上一个拉格朗日乘子。然后定义出拉格朗日函
过一个目标函数包含约束条件，便可以清楚解释问题 。

比如对 （1）式 每一个约束（共有m个约束，$y_i(w^T x_i + b) \geq 1$），添加拉格朗日乘子 $\alpha_i \geq 0$,则整

$$L(w, b, \alpha) = \frac{1}{2} // w //^2 + \sum_{i=1}^{m} \alpha^i (1 - y_i(w^T x_i + b))$$

为什么使用这样的拉格朗日乘子，又为何这样构建？这实际上是因为我们的目标函数是不等式约束
一个约束 $\alpha_i \geq 0$。最终我们便通过KKT条件来产生原问题的对偶问题。
同样的，将上面这个式子记为 （2）式 。

可以看到，由于 $\alpha_i \geq 0$, 这样，但凡有约束条件之一不满足，如 $y_k(w^T x_k + b) < 1$），

$L(w, b, \alpha) = \infty$。只有约束条件均满足的时候，

$L(w, b, \alpha)$有最优值，为 $L(w, b, \alpha) = \frac{1}{2} // w //^2$

所以优化 $\frac{1}{2} // w //^2$ 等价于优化 $L(w, b, \alpha)$当然，要满足约束条件 $\alpha_i \geq 0$。

于是，我们的目标函数可以表示为:

$\min_{w,b} \max_{\alpha \geq 0} L(w, b, \alpha)$

满足一定条件下，等价于（注意，这个满足一定条件，是指满足KKT条件）

$\max_{\alpha\geq 0}\min_{w,b} L(w,b,\alpha)$

后者把最小和最大的位置交换，这样使得运算方便起来。

##KKT条件
什么是KKT条件？其实在这之前，本文有稍微有提到过。在这里正式介绍一下。

KKT条件 是一个线性规划问题能 有最优解的 充分和必要条件。

一般地，一个最优化数学模型可以表示成如下形式：

$$min f(x)$$
$$s.t. h_i(x) = 0, i = 1, 2, ..., p$$
$$g_j(x) \leq 0, j = 1, 2, ..., q$$
$$x \in X \in R^n$$

$h(x)$ 是等式约束。
$g(x)$ 是不等式约束。
p, q 表示约束的数量。

而这个最优化数学模型的最优解x^*须满足的条件，即KKT条件为：

$$h_i(x^*) = 0, i = 1, 2, ..., p, 和\ g_j(x^*) \leq 0, j = 1, 2, ..., q$$
$$\bigtriangledown f(x^*) + \sum_{i=1}^{p} \lambda_i \bigtriangledown h_i(x^*) + \sum_{j=1}^{q} \mu_k \bigtriangledown g_k(x^*) = 0$$
$$\lambda_i \neq 0, \mu_k \geq 0, \mu_k g_k(x^*) = 0$$

于是我们的整个问题转化为

$L(w, b, \alpha)$ 对w, b求最小

再对α 求最大。

对于第一步，先令$L(w, b, \alpha)$对w, b求偏导为0，可得：
$$w = \sum_{i=1}^{m} \alpha_i y_i x_i,$$

$$0 = \sum_{i=1}^{m} \alpha_i y_i.$$

将此两个式子带入 （2）式 消去w, b。便得到了 （1）式 的对偶问题。

$$\max_{\alpha\geq 0} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$s.t. \sum_{i=1}^{m} \alpha_i y_i = 0,$$

$\alpha_i \geq 0, i = 1, 2, ..., m$

类比来看，我们的目标函数没有$h(x) = 0$ 的等式约束。
于是，上面的过程，需要满足的KKT条件是

$$\begin{cases} \alpha_i \geq 0; \\ 1 - y_i(w^T x_i + b) \leq 0; \\ \alpha_i(1 - y_i(w^T x_i + b)) = 0. \end{cases}$$

我们看到，对于任意样本，总有$\alpha_i = 0$ 或者 $y_i(w^T x_i + b) = 1$.若$\alpha_i = 0$ ，则由$w = \sum_{i=1}^m \alpha_i y_i x$
时，必有$y_i(w^T x_i + b) = 1$ ，此时α_i对应的向量在最大间隔的边缘上（一开始示意图的虚线上）

接下来，怎么求α呢?
##SMO算法

##################################
一些延展:

我们可以在这里从损失函数的角度看一下LR与SVM的异同:

对于LR与SVM的异同我总结如下:

LR的详细介绍:https://blog.csdn.net/b285795298/article/details/88683987

- 相同点:

> LR和SVM都是**分类**算法
>
> LR和SVM都是**监督学习**算法。
>
> LR和SVM都是**判别模型**。
>
> 如果不考虑核函数，LR和SVM都是**线性分类**算法，也就是说他们的分类决策面都是线性的。
> 说明:LR也是可以用核函数的.但LR通常不采用核函数的方法.(**计算量太大**)

LR和SVM不同点:

1、LR采用log损失，SVM采用合页(hinge)损失。

逻辑回归的损失函数:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))\right]$$

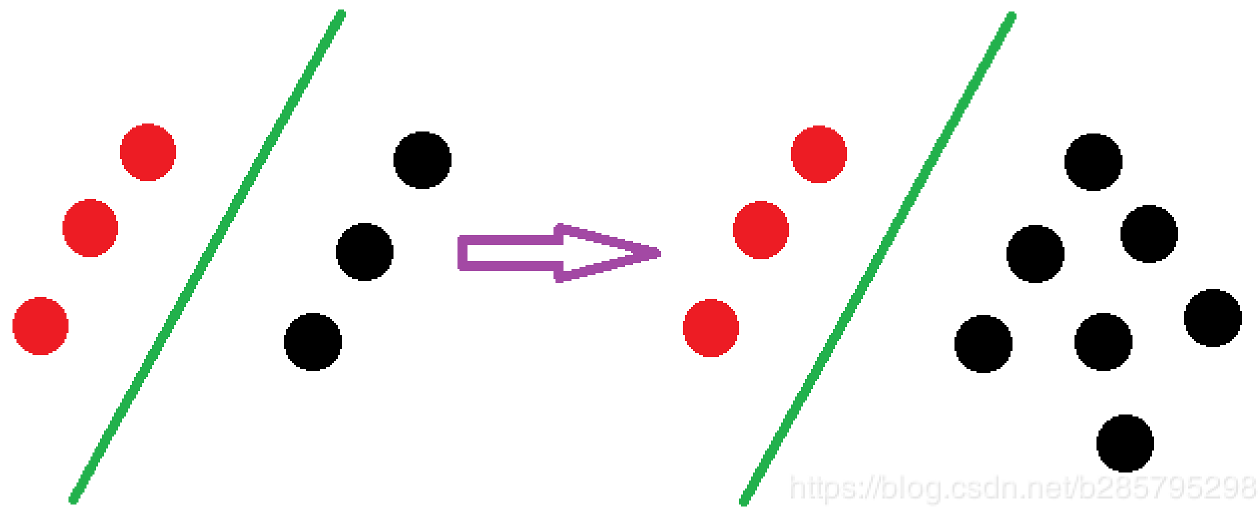
支持向量机的目标函数:

$$\mathcal{L}(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{n}\alpha_i\left(y_i(w^Tx_i+b)-1\right)$$

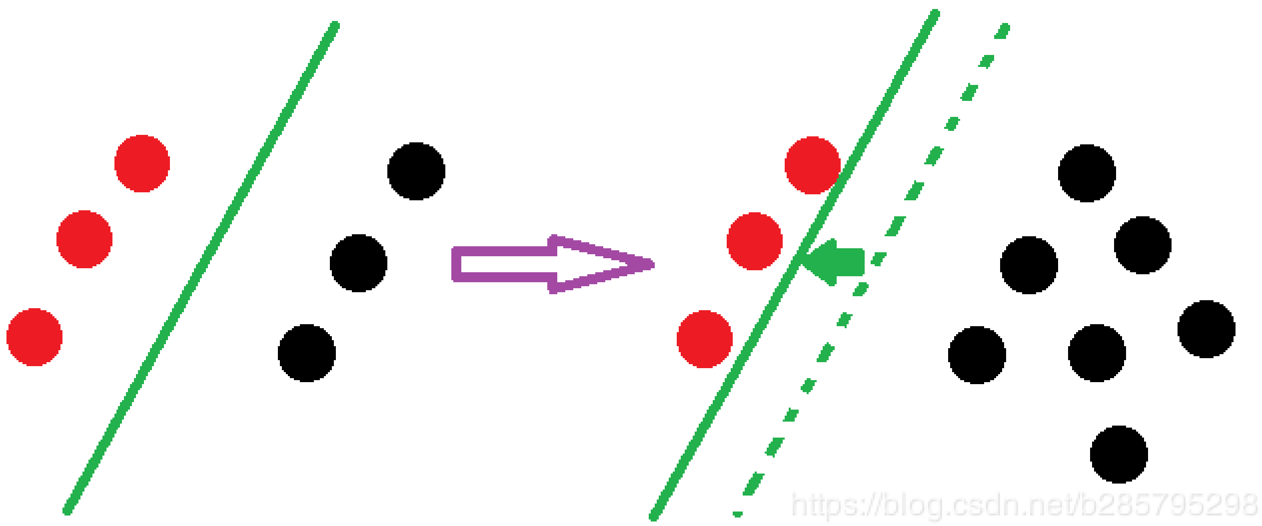
逻辑回归方法基于概率理论，假设样本为1的概率可以用sigmoid函数来表示，然后通过**极大似然**
。支持向量机基于几何**间隔最大化**原理，认为存在最大几何间隔的分类面为最优分类面.(有严格的

2、LR对异常值敏感，SVM对异常值不敏感(抗燥能力,SVM要强)(https://www.jianshu.com/p/
考虑全局（远离的点对边界线的确定也起作用，虽然作用会相对小一些）。LR模型找到的那个超
近中间分割线的那些点尽量远离，即只用到那些支持向量的样本。

支持向量机改变非支持向量样本并不会引起决策面的变化：



逻



LR则
先对数据做**balancing**。（引自http://www.zhihu.com/question/26768865/answer/340781

3、计算复杂度不同。对于海量数据，SVM的效率较低，LR效率比较高。 对于两者在feature和
考:https://blog.csdn.net/a244659184/article/details/81122521。该文章说明了:

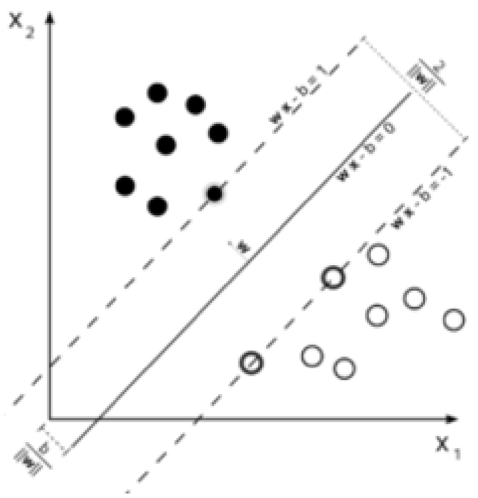
当样本较少，特征维数较低时，SVM和LR的运行时间均比较短，**SVM较短一些。准确率的话，L** **准确率赶超了LR。**SVM时间虽长，但在接收范围内。当数据量增长到20000时，特征维数增长到 **率却和LR相差无几。**(这其中主要原因是大量非支持向量参与计算,造成SVM的二次规划问题)

4、对非线性问题的处理方式不同，LR主要靠特征构造，必须组合交叉特征，特征离散化。SVM **度不高)。**(由于可以利用核函数,。SVM则可以通过对偶求解高效处理。LR则在特征空间维度很高

5、SVM的损失函数就自带正则！！！（损失函数中的1/2||w||^2项），这就是为什么SVM是结 **项！！！**

$$\mathcal{L}(w,b,\alpha)=\boxed{\frac{1}{2}\|w\|^2}-\sum_{i=1}^{n}\alpha_i\left(y_i(w^Tx_i+b)-1\right)$$

关于正则化:
给定一个数据集，一旦完成Linear SVM的求解，所有数据点可以被归成两类
1）一类是落在对应分界平面外并被正确分类的点，比如落在正分界左侧的正样本或落在负分界在
2）第二类是落在gap里或被错误分类的点。



假设一个数据集已经被Linear SVM求解，那么往这个数据集里面增加或者删除更多的一类点**并不** 们在看看LR。

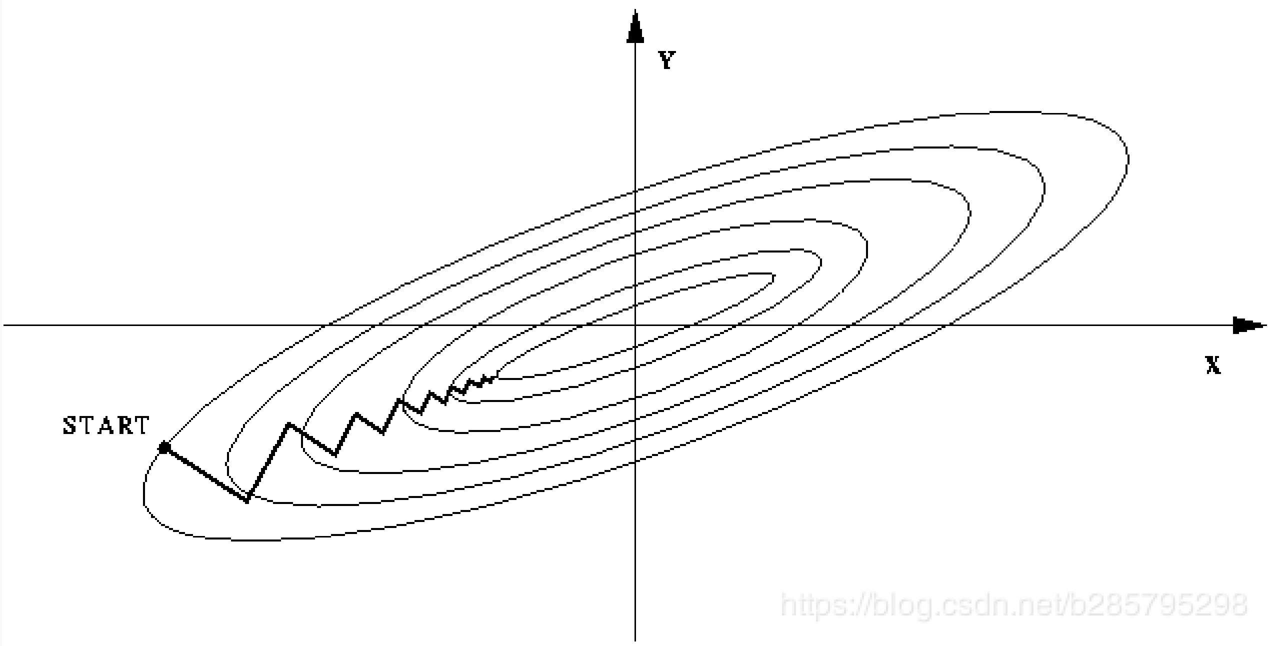
值得一提的是求解LR模型过程中，**每一个数据点对分类平面都是有影响的**，它的影响力远离它到 在实际应用中，如果数据维度很高，LR模型都会配合参数的L1 regularization。

要说有什么本质区别，那就是两个模型对数据和参数的敏感程度不同，Linear SVM比较依赖**pen** **做L1 regularization的系数。但是由于他们或多或少都是线性分类器，所以实际上对低维度数据** **加稳定，为什么呢?**
因为Linear SVM在计算margin有多“宽”的时候是依赖数据表达上的距离测度(可以理解为度量

好（badly scaled，这种情况在高维数据尤为显著），所求得的所谓Large margin就没有意义。
全避免。所以使用Linear SVM之前一般都需要先对数据做normalization，(这里的normalizat
balancing)而求解LR（without regularization）时则不需要或者结果不敏感。

同时会有：**feature scaling**会使得gradient descent的收敛更好。

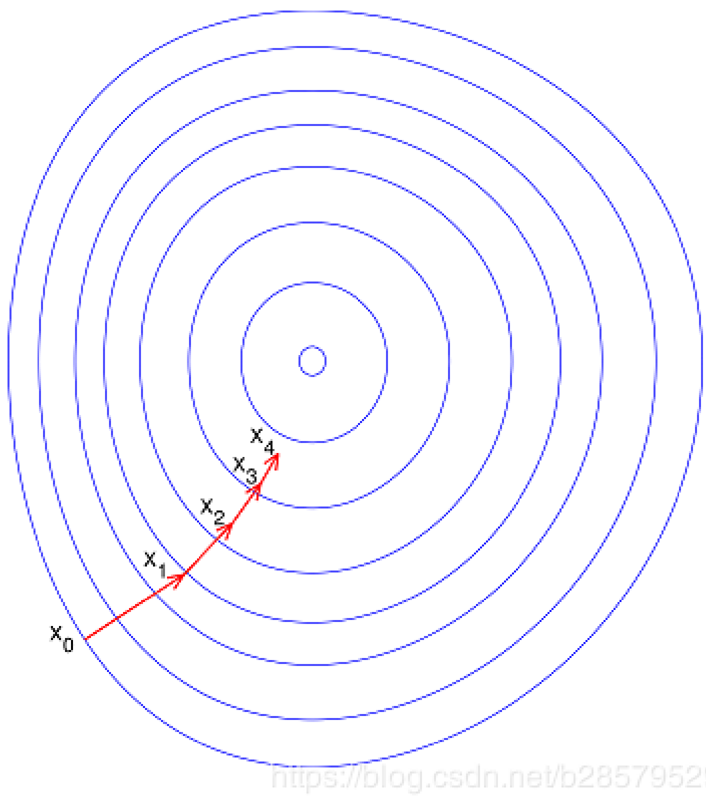
如果不归一化，各维特征的跨度差距很大，目标函数就会是“扁”的：



（图中椭圆表示目标函数的等高线，两个坐标轴代表两个特征）

这样feature scaling之后，在进行梯度下降的时候，梯度的方向就会偏离最小值的方向，走很多

如果归一化了，那么目标函数就“圆”了：



每一步梯度的方向都基本指向最小值，

https://www.zhihu.com/question/37129350)

向您推荐我的其他文章:

支持向量机（SVM）从入门到放弃再到掌握 :https://blog.csdn.net/b285795298/article/detai

逻辑回归(Logistic Regression-LR)从入门到放弃再到掌握 :https://blog.csdn.net/b285795298

决策树（decesion Tree）从入门到放弃再到掌握 :https://blog.csdn.net/b285795298/article/

随机森林(Random Forest) 从入门到放弃再到掌握:https://blog.csdn.net/b285795298/article/

##

参考博文:
机器学习中的线性代数之矩阵求导 https://blog.csdn.net/u010976453/article/details/543812
周志华老师的《机器学习》
参考:: https://www.jianshu.com/p/e8dca5613da6
https://www.jianshu.com/p/1a41a1567b87
https://www.cnblogs.com/bentuwuying/p/6616761.html
https://blog.csdn.net/zwqjoy/article/details/82312783