察异辨花: 简单得线性和非线性分类

Introduction to linear and nonlinear methods for classification

王斐, Michael

SenseTime Edu

Math, Economics, Philosophy (UCD, Nottingham, CUHK)

https://github.com/Michael-yunfei/MDLforBeginners

2020 年 7 月 23 日

本节内容

1. 课程结构和进程简介
2. 数据阵 (Dataframe) 简介
3. 两个大脑论和人工智能的思维方式
4. 分类和线性分类
5. 损失函数 (Loss Function)
6. 梯度下降法 (Gradient Descent)
7. 支持向量机 (Support Vector Machine)
8. 非线性分类介绍 (SVM, kernels)

课程结构

第二章是我们高中教材机器学习的入门章节，本章可以说是整本教材最重要的章节，原因有:

- 第一次开始介绍机器学习的数理模型
- 第一次系统介绍分类 (人工智能领域 70% 的问题是涉及分类)
- 透过本章了解机器学习的整个思路和流程

辅导讲义的组成部分:

- 课前预习 C1
- 课堂讲义 C2 (讲义偏重数学严谨度，课堂偏重理解)
- 课后练习 C3 (包括实验)

认真完成以上三个内容可以得到 A, 只完成 C1-C2 可以得 B。

相关考核

考核机制的设计如下:

- 课后练习 C3 - 10%
- 期中考试 - 20 %
- 期末考试 - 30 %
- 案例报告 - 40 % （会讲故事 + 编程且调用相关程序)

具体的考试形式和报告内容要求，后面会有具体的指示。

课程结构和进程

写给辅导老师们的前言: 我们整个高中教材大体可以分为三大块:

- 图片 (分类，识别和聚类): Convolutional Neural Network
- 语音 (识别，分类): Sequence Model (Recurrent NN)
- 文本 (特征提取和分析): Sequence Model (RNN)

三个方向都涉及到深度学习模型 (神经网络模型), 给学生介绍的时候，重点放在:

- 想象力 的构建上 (后面我会在神经网络模型中介绍，很多巧妙的设计数学概念并不复杂，更多是需要想象力)
- 强化学生的人工智能思维
- 强化学生的数据感觉和数据处理能力

课程结构和进程

由于神经网络模型背后的数学框架较为复杂，深入得讲解并不符合高中的教学目标，所以:

- 线性 (感知器 perceptron) 和非线性 (支持向量机 support vector machines) 分类就成为整个高中的教学重点
  - 数学模型要讲透彻: 理解每一个环节的公式和算法原理
  - 机器学习框架下的编程要逐步解释: 学生必须可以自主完成感知器的算法编写
  - 有条件得可以引导进行支持向量机的算法编写
  - 课时安排: (1 + 2 + 6 = 9), 如果课堂讲解是 3 个小时，总课时就需要 27 个小时。(12 个小时是必须的)
- 对于神经网络模型的讲解:
  - 可视化分析 + 案例演示
  - 课外体验 + 参观 (很多模型训练时间长，需要算力大)

第二章的重要性

我要在反复强调下第二章的重要性，你如果可以很好得理解第二章的模型，其它得才好举一反三，而且有助于我们学习神经网络模型。

- 数学不好，没问题，不要担心；
- 会算加减乘除就可以学懂，要对自己有信心；
- 当然最重要的是要有耐心！（听课 + 练习 + 实验 =12 个小时)

课前预习题目浅析

课前预习第一部分是生活案例题，从生活中找一个分类的例子，然后描述下你是如何分类的 (不超过 60 字)，描述必须包含下面三个部分:

- 分类的主体对象是什么？(比如，汽车，花草，等等)
- 分类的依据和标准是什么？
- 分类后如何衡量分类得好坏？

课前预习题目浅析

分类简单讲就是：寻找差异 (看起来不一样) 比如下面这种图,

课前预习作文题目浅析

因为事物的属性可以是很繁杂的，比如自然界中光是树木的科目分类就达到几万种 (如果考虑不同植被的话，这个科目的量级会进一步上升, 吴征镒)。所以我们在对这些属性进行界定时，最简单有效的方式就是对其数字化 (或者说有数字去测量和统计)，比如下面这个案例。

属种	代码	花瓣长度	花瓣宽度
山鸢尾	s1	1.4	0.2
山鸢尾	s2	1.7	0.4
变色鸢尾	ve1	3.9	1.4
变色鸢尾	ve2	4.9	1.5
维吉尼亚鸢尾	vig1	6.9	2.3
维吉尼亚鸢尾	vig2	6.1	1.9

数据阵定义

因为我们在机器学习领域经常接触数据阵，所以我们先来熟悉一下数据阵的概念和相关术语。

定义

我们把由数字和对其的描述 (有其根据) 组成的信息形态，称为数据; 我们将统一在一个样本下的数列，称为数据组(数据串); 有多个数据组构成的一系列数据，称为数据阵。

姓名	身高	性别	成绩
Michael	176	1	89
Ani	163	0	93
Marina	161	1	82
Emmet	168	1	76
Lucy	165	0	67

数和数据的差别还是很大的!

数据阵的常用术语

因为我们几乎任何一个模型最初的切入点都是数据阵，所以很有必要统一下专业术语。

数据阵(dataframe)

属性(n)=7

标注结果

样本

m=15

样本	交易日期	房屋年龄	地铁距离(米)	便利店数量	维度	经度	房屋价格(万台币/坪)	房屋价格(万元/平方米)
1	2012.917	32	84.87882	10	24.98298	121.54024	37.9	2.720
2	2012.917	19.5	306.5947	9	24.98034	121.53951	42.2	3.028
3	2013.583	13.3	561.9845	5	24.98746	121.54391	47.3	3.394
4	2013.500	13.3	561.9845	5	24.98746	121.54391	54.8	3.932
5	2012.833	5	390.5684	5	24.97937	121.54245	43.1	3.093
6	2012.667	7.1	2175.03	3	24.96305	121.51254	32.1	2.303
7	2012.667	34.5	623.4731	7	24.97933	121.53642	40.3	2.892
8	2013.417	20.3	287.6025	6	24.98042	121.54228	46.7	3.351
9	2013.500	31.7	5512.038	1	24.95095	121.48458	18.8	1.349
10	2013.417	17.9	1783.18	3	24.96731	121.51486	22.1	1.586
11	2013.083	34.8	405.2134	1	24.97349	121.53372	41.4	2.971
12	2013.333	6.3	90.45606	9	24.97433	121.5431	58.1	4.169
13	2012.917	13	492.2313	5	24.96515	121.53737	39.3	2.820
14	2012.667	20.4	2469.645	4	24.96108	121.51046	23.8	1.708
15	2013.500	13.2	1164.838	4	24.99156	121.53406	34.3	2.461

X

Y

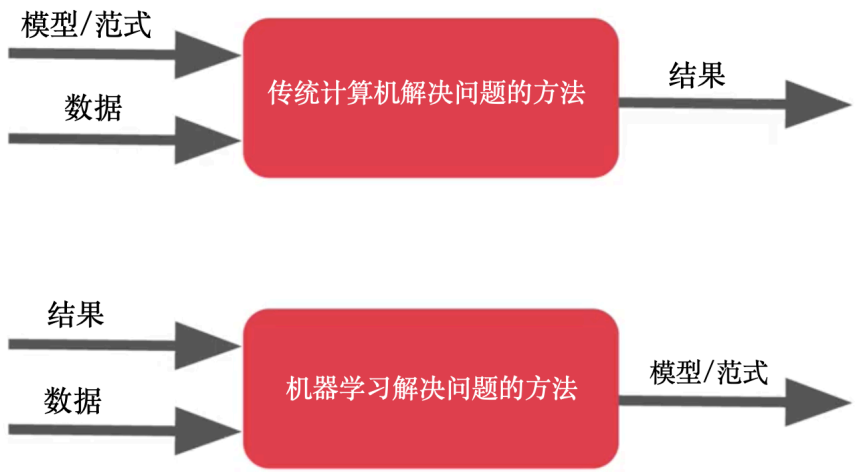
X = m x n = 15 x 7 的矩阵

Y=15 x 1 向量

人工智能思维方式

这里我想给大家做一下思维转换，面向未来:

- 传统解决问题方案: 观察世界，抽象问题，解决问题
- 人工智能解决问题方案: 电脑观察世界，人抽象问题，两个大脑共同解决问题。



假定自然界运行有规律的话，那么对这些规律事件的记录和存储全部交给计算机，然后再由机器学习、深度学习模型对这些规律进行提取，从而进行预测。

人工智能思维方式

现在我们就通过分类问题 来理解人工智能的思维方式！

什么是分类?

定义

分类指的是根据事物不同的属性，对某个或一些物体进行划分的过程。在没有任何属性的情况下，我们不能进行分类。在有属性的情况下，我们可以依据属性，进行分类，比如黑猫和白猫。

从哲学的角度上讲，存在的便是有属性的在。绝对的存在 (即无属性的纯存在) 在中文哲学的概念里指的是 '道' (老子里有一句名言: 道生一，一生二，三生万物)，在西方哲学里指的是 '上帝'。黑格尔有一句名言: 绝对得有即是绝对得无 (纯有等同于纯无，中文: 白马非马)

- 人 - 属性: 男女，身高，体重，肤色...
- 花 - 属性: 颜色，开花季节...
- 书本 - 重量，颜色，科目

什么是线性分类

例

在预习题目中，我们要求同学们求解下列的方程组:

$$
\begin{aligned}
56w_1 + 8w_2 &= 36 \\
32w_1 + 4w_2 + w_3 &= 20.5 \\
48w_1 + 3w_2 + w_3 &= 29.8
\end{aligned}
$$

$$
\begin{bmatrix} 56 & 8 & 0 \\ 32 & 4 & 1 \\ 48 & 3 & 1 \end{bmatrix}
\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}
=
\begin{bmatrix} 36 \\ 20.5 \\ 29.8 \end{bmatrix}
$$

稍加计算便可以得出:

$$
w_1 = 0.6; \quad w_2 = 0.3; \quad w_3 = 0.1
$$

这个题目是老师专门设计过的，其背后的场景为下面的表格。

什么是线性分类

姓名	学习时间(小时)	练习题目数量	考前睡眠(1-正常休息;0-熬夜)	成绩
Michael	56	8	0	36
Ani	32	4	1	20.5
Marina	48	3	1	29.8
Emmet	60	9	0	38.7
Lucy	59	10	1	38.5

例

$$X \cdot w = Y$$

$$\begin{bmatrix} 56 & 8 & 0 \\ 32 & 4 & 1 \\ 48 & 3 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 36 \\ 20.5 \\ 29.8 \end{bmatrix}$$

什么是线性分类

如果假定学习的各种属性 (学习时间、练习题目数量，等) 与结果 (成绩) 存在简单的线性关系的话，解方程可以告诉我们:

- $w_1 = 0.6$, 即学习时间对成绩的影响比重约为 60%;
- $w_2 = 0.3$, 即练习题目的数量对成绩的影响比重约 30%;
- $w_3 = 0.1$, 即考前睡眠对成绩的影响比重约 10%。

虽然这个题目很简单的，但是其反应了机器学习的一般框架:

- 假定自然界中的事物发展是有规律的
- 这些规律被记录存储为数字和文本形式 (X, Y)
- 人类根据对问题的抽象设定相关的框架和模型 (例如，线性方程)
- 结合模型和数据，获得规律参数 w
- 拿到规律参数 w 后，再去预测

什么是线性分类

现在我们从鸢尾花数据阵中随意取三个样本，然后属性也取三个
$X = 3 \times 3, Y = 3 \times 1$, 然后解方程

属种	代码	花萼宽度 (a=1.1)	花瓣长度 (b=-3.7)	花瓣宽度 (c=9.)
山鸢尾	0	3	1.4	0.2
变色鸢尾	1	3.2	4.5	1.5
维吉尼亚鸢尾	2	2.7	5.1	1.9

我们可以理解大自然在不同的鸢尾花进化时，特意的分配了下面的规律参数:

- 花萼宽度 (1.1)
- 花瓣长度 (-3.7)
- 花瓣宽度 (9.4)

用这组规律参数去预测的话，准确率只有大约 25%，显示这种方法太过于简单粗暴。

什么是线性分类

定义

我们把可以表现为下列形式的方程，称为线性方程:

$$a_1 x_1 + a_2 x_2 + \cdots + a_n x_n + b = 0$$

其中 $x_1, x_2, \cdots, x_n$ 为变量, $a_1, a_2, \cdots, a_n$ 为系数。

线性方程本质是一种累计的过程，因为乘法也是加法，比如 $3 \times 5 = 5 + 5 + 5 = 15$。我们刚才计算的那几个例子都是线性的。

什么是线性分类

例

现实生活中，有些关系可以表现为线性关系，有些关系不可以表现为线性关系，比如，

$$a_1 x_1^2 + a_2 \sin(x_2) + a_3 e^{x_3} + b = 0$$

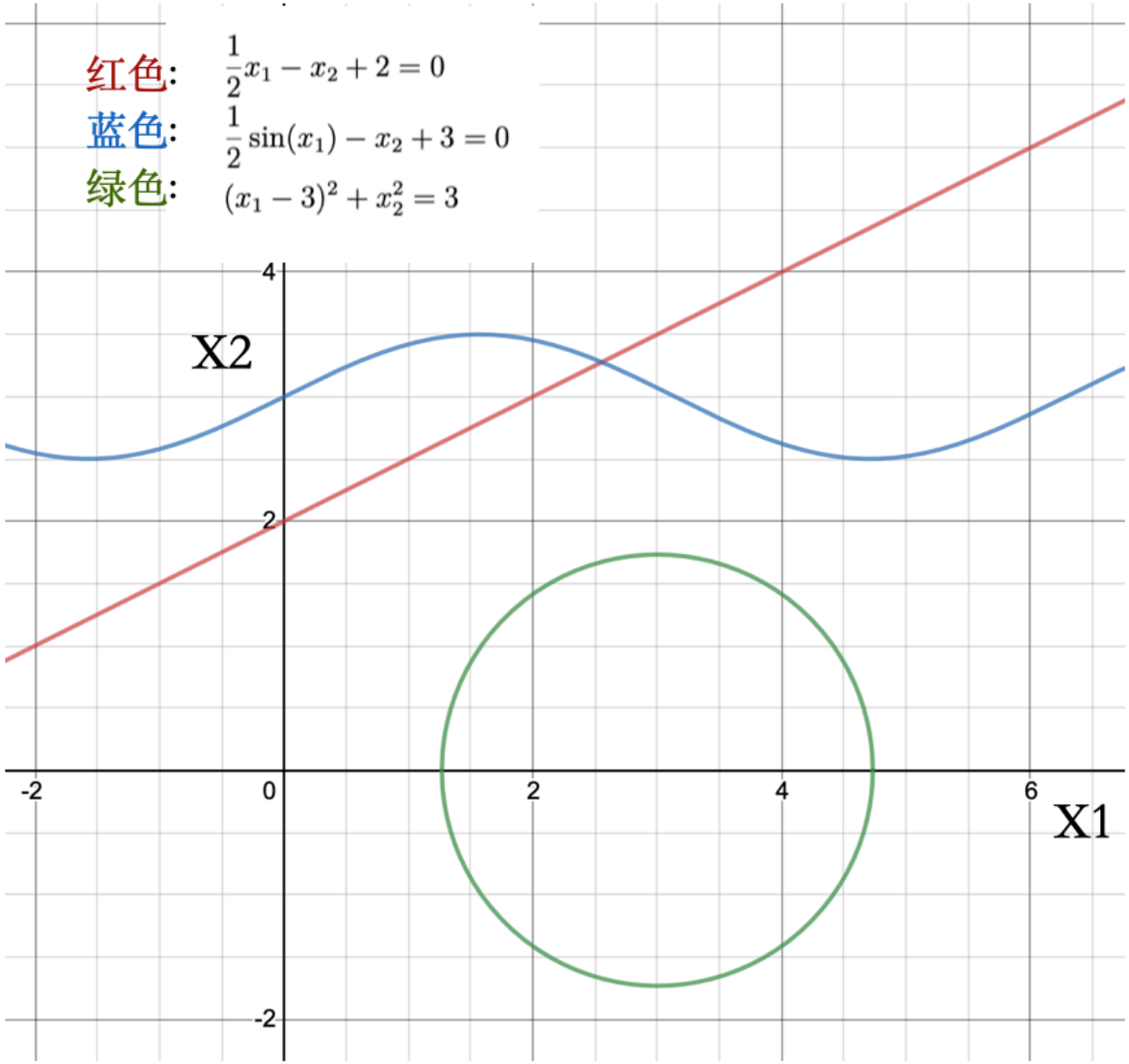
当 x_1, x_2, x_3 之间的关系为非线性是，我们可以对其进行线性转换，比如：

$$x_1' = x_1^2; \quad x_2' = \sin(x_2); \quad x_3' = e^{x_3}$$

那么新的关系就变成了线性：

$$a_1 x_1' + a_2 x_2' + a_3 x_3' = 0$$

什么是线性分类


红色: $\frac{1}{2}x_1 - x_2 + 2 = 0$
蓝色: $\frac{1}{2}\sin(x_1) - x_2 + 3 = 0$
绿色: $(x_1 - 3)^2 + x_2^2 = 3$
X2
X1
4
2
-2
0
2
4
6
-2

什么是线性分类: 矩阵和向量点积

定义

矩阵（Matrix）是一个按照长方阵列排列的复数或实数集合。对一个矩阵的简单描述可以为 $m \times n$ 的矩阵，即该矩阵有 m 行，n 列. $m \times n$ 也被称为矩阵的大小。

例

比如，我们有 3×3 的矩阵 A 和 2×3 的矩阵 B。

$$A = \begin{bmatrix} 1 & 3 & 4 \\ 2 & 5 & 8 \\ 3 & 9 & 7 \end{bmatrix}, \qquad B = \begin{bmatrix} 3.2 & 1.9 & 5.7 \\ 4.1 & 5.8 & 9.3 \end{bmatrix}$$

什么是线性分类: 矩阵和向量点积

定义

矩阵中单独的一行被成为行向量 (row vector), 单独的一列别成为列向量 (column vector)。

例

比如在上面的矩阵 A 中，我们有以下单独的向量:

$$A = \begin{bmatrix} 1 & 3 & 4 \\ 2 & 5 & 8 \\ 3 & 9 & 7 \end{bmatrix}$$

$$A_{r2} = [2\ 5\ 8], \qquad A_{c2} = \begin{bmatrix} 3 \\ 5 \\ 9 \end{bmatrix}$$

什么是线性分类: 向量点积

矩阵和向量的运算，是根植于方程中变量与系数 (规律参数) 的对应关系来的。我们这一小节中重点学习向量运算，下一章中开始学习矩阵运算。比如，下面的方程可以转为向量相乘:

$$2a + 3b + c = 3 \quad \Leftrightarrow \quad \begin{bmatrix} 2 & 3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = [3]$$

因为变量和系数，必须相对应，所以下面的运算一个是不合法的，另外两个是不合法的:

$$\begin{bmatrix} 2 & 3 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \qquad \begin{bmatrix} 2 & 3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \qquad \begin{bmatrix} 2 & 3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

点积运算时，向量维数 必须相同: 同位元素相乘后累加。(不要说向量的长度，长度是另外的概念)

什么是线性分类: 向量点积

定义

已知两个维数相同的向量 $w = [w_1, w_2, \cdots, w_n]$ 和 $X = [x_1, x_2, \cdots, x_n]$, 那么向量的相乘，也并被成为点积 (dot product) 定义为:

$$X \cdot a = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

$$w \cdot X = \begin{bmatrix} w_1 & w_2 & \cdots & w_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$= \sum_{i=1}^{n} x_i w_i = a_1 w_1 + a_2 w_2 + \cdots + a_n w_n$$

什么是线性分类: 向量点积几何性质

定义

当两个向量的点积运算结果为 0 时，我们称这两个向量正交，或者互为正交向量，数学表示为:

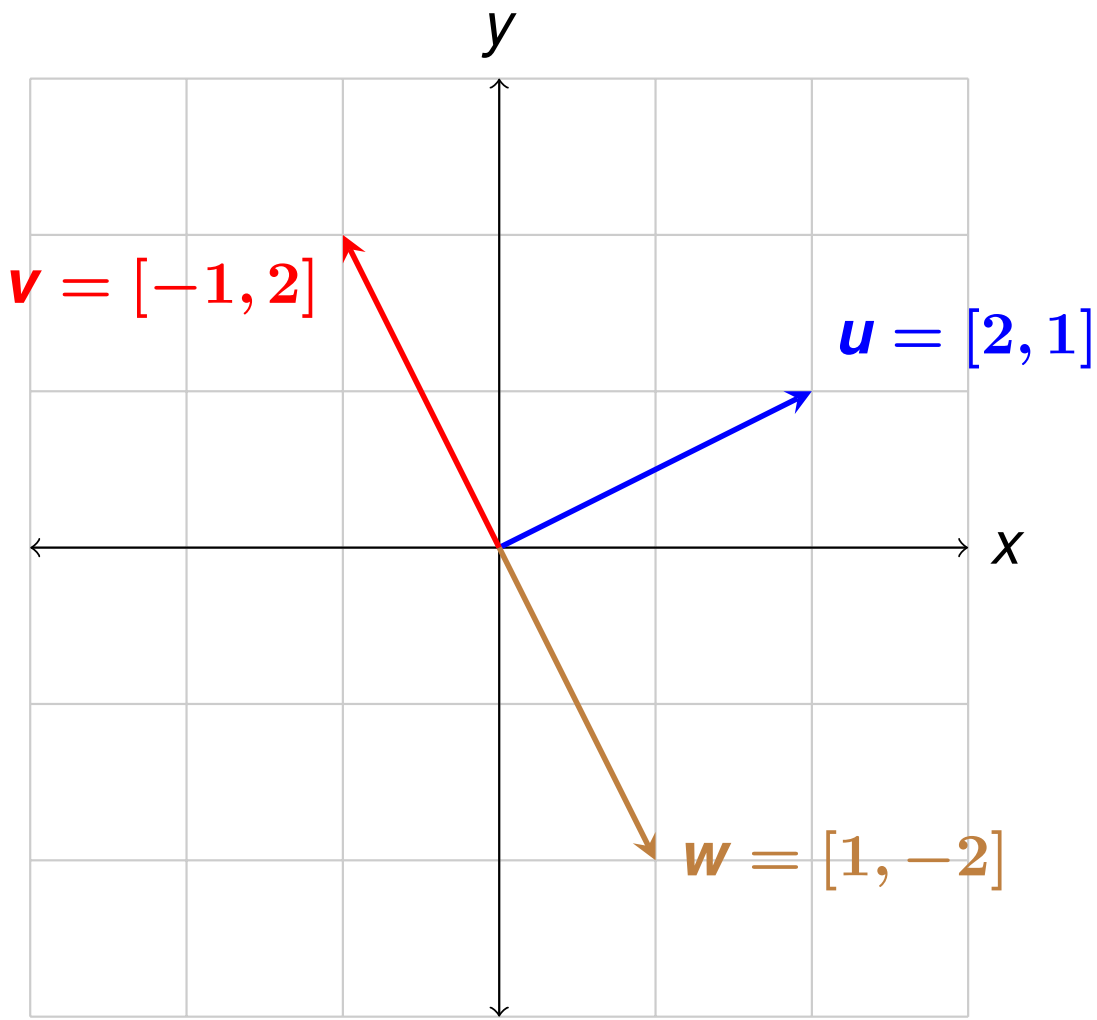
$$X \cdot w =< X, w >= 0 \quad \Leftrightarrow \quad \sum_{i=1}^{n} x_i w_i = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n = 0$$

两个向量正交时，在几何上表现为垂直 (perpendicular)，比如

$$u \cdot v = \begin{bmatrix} 2 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \end{bmatrix} = -2 + 2 = 0 \qquad z \cdot x = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = 0$$

什么是线性分类: 向量点积几何性质

两个向量正交时，在几何上表现为垂直 (perpendicular)，比如

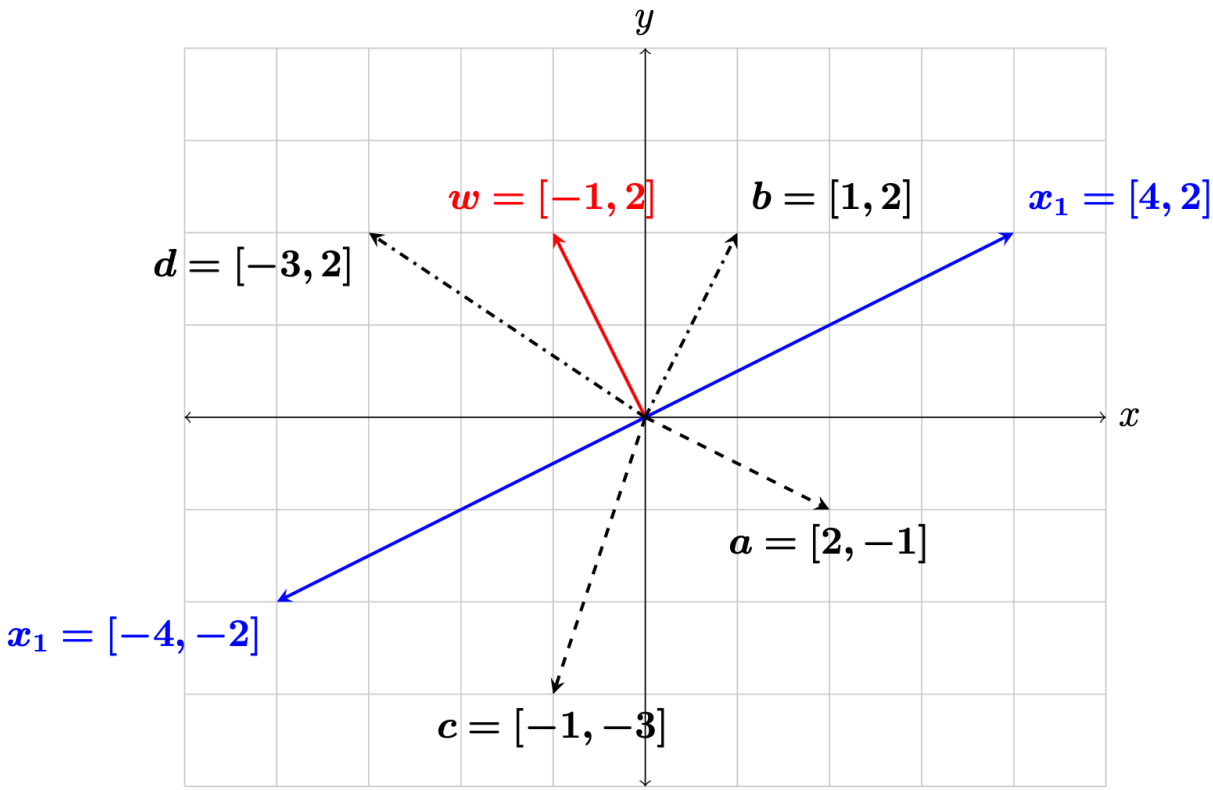


注意 $\boldsymbol{w} \cdot \boldsymbol{u} = 0$, 他们在几何上垂直，在代数上正交 (orthogonal)。

什么是线性分类: 向量点积几何性质

整个感知器分类和支持向量机，完全可以用下面这张图来概括。

与 w 点积	<0	$=0$	>0
	a	x_1	b
	c	x_2	d



感知器线性分类 (perceptron)

定义

假设自然界中可分类对象具有一定属性，且属性个数大于 1，当其属性的特征变量 (X) 与其属性的规律参数 (w) 存在下列线性关系时:

$$X \cdot w + b = 0; \quad \Leftrightarrow \quad x_1 w_1 + x_2 w_2 + \cdots + x_n w_n + b = 0$$

(其中 n 为分类对象的属性个数)，我们将其称为可线性分类。因为该公式的点积为零，所以我们将其转换为下列的格式:

$$\begin{bmatrix} X & 1 \end{bmatrix} \begin{bmatrix} w \\ b \end{bmatrix} = 0$$

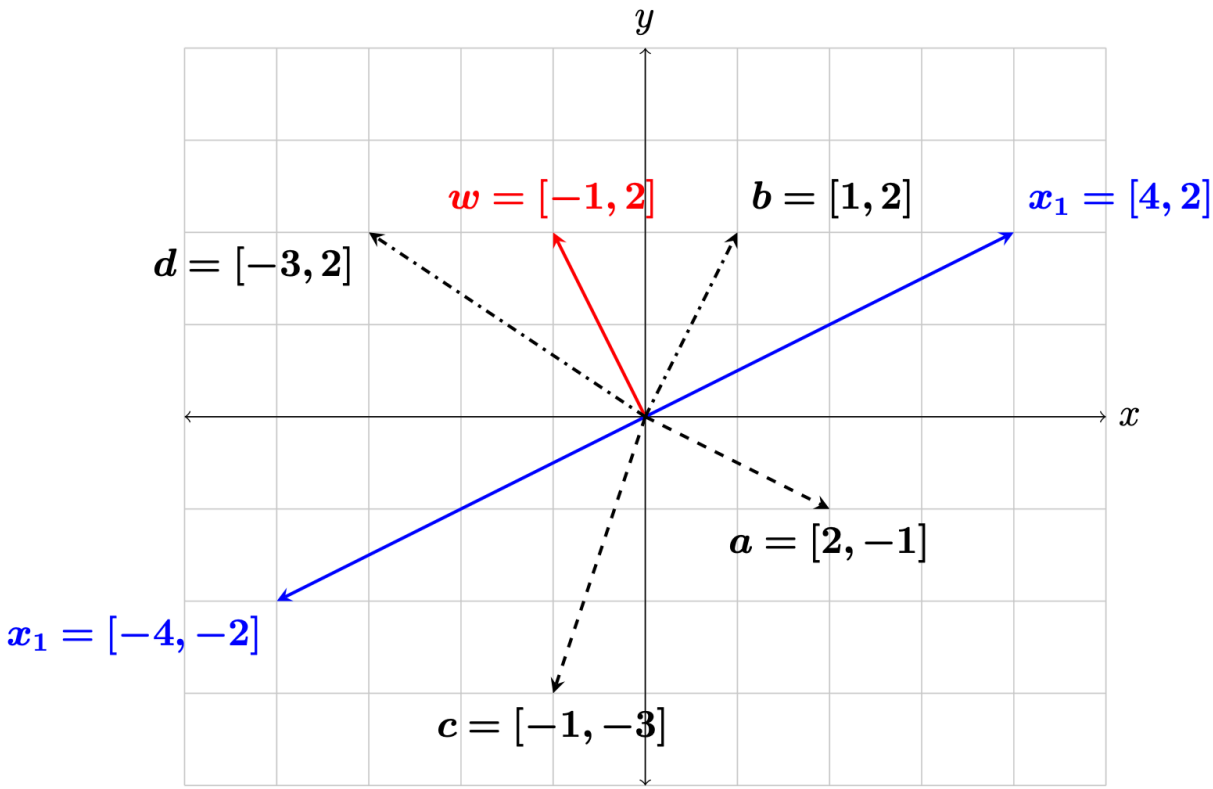
其中 b 为常数变量 (无属性变量), 我们将向量 w 和 x 组成的分界面以及正交矢量 (在二维中，就是一条线和其垂直线) 称为感知器 (percepton), 其中 w 就是我们需要计算的规律参数，而 X 则是由我们采集的数据组成的特征变量

感知器线性分类 (perceptron)

只能二分! 但是自然界很多问题二分也足够了:

- 黄猫 V.S. 非黄猫
- 蓝猫 V.S. 非蓝猫
- 重复二分等同于多类分法

与 w 点积	<0	$=0$	>0
	a	x_1	b
	c	x_2	d



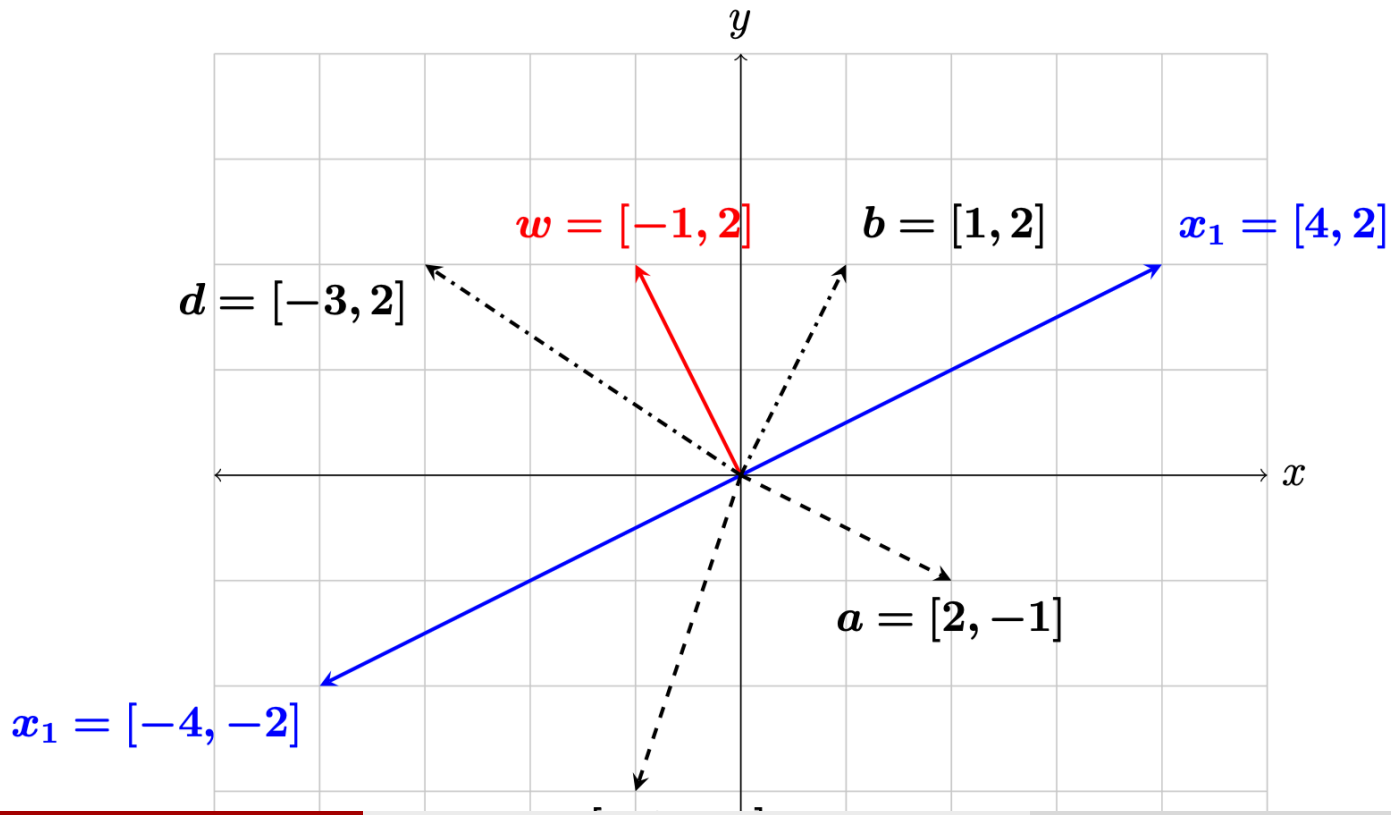
感知器线性分类 (perceptron)

感知器被定义为:

$$X \cdot w + b = 0$$

$$x_1 w_1 + x_2 w_2 + \cdots + x_n w_n + b = 0$$

也就是下图中的蓝色和红色的向量。



感知器线性分类 (perceptron) 过程

感知器分类

我们将向量 w 和 x 组成的分界面以及正交矢量 (在二维中，就是一条线和其垂直线) 称为感知器 (percepton), 其中 w 就是我们需要计算的规律参数，而 x 则是由我们采集的数据组成的特征变量。感知器分类一个二分过程，分类过程如下:

$$\hat{y} = \begin{cases} +1 & x \cdot w > 0 \\ -1 & x \cdot w < 0 \end{cases}$$

其中 $\hat{y}$ 是我们在进行分类是需要标注的结果。

损失函数

拥有了标注的数据，就有了对比依据[1]，就可以训练人工智能模型，从而可以调整模型参数，来提升预测准确度。而用来衡量准确度的方程就需要损失方程。

定义

假设数据阵标注过的数据准确可靠，在机器学习模型中，模型在根据学习到参数规律对特征变量 (统计的数据) 计算后输出的结果 $\hat{Y}$ 与数据阵[a]标注的数据结果 Y 差别的函数统称为损失函数 *(Loss Function)*。损失方程的一般形式为:

$$L(Y, \hat{Y}) : \rightarrow \mathbb{R}$$

[a](注意是 '数据阵'，所以是衡量的是模型在整个数据阵下的表现)

[1]对，没有对比就没有伤害。

感知器损失函数

定义

给定机器学习模型，对单个数据串的预测准确度的衡量的方程，被成为测量公式 (cost function)，例如，x_1 为一个数据串[a]，w 是我们的训练得出的规律参数，$\hat{y}_1$ 为我们预测的结果，y_1 为数据串标注的结果，那么我们的测量公式 (cost function) 为:

$$y_1 - \hat{y}_1 = y_1 - x_1 \cdot w$$

实际上，损失函数 (Loss function) 就是对测量公式的求和:

$$L = \sum_{i=1}^{m} y_i - \hat{y}_i$$

其中 L 为损失函数。

[a]即为 $1 \times n$ 的向量，重复记忆 $m \times n$, m 样本数，n 属性个数。

感知器损失函数

我们来详细解释下在感知分类中所定义的损失函数，以及为什么我们需要其最小化。再重复一下，y_i 为标注的结果，$\hat{y}_i$ 为模型预测的结果，我们看一下具体的计算:

$$-(y_i \times \hat{y}_i) = \begin{cases} -1 & \text{如果 } y_i = 1, \hat{y}_i = 1, \text{预测正确} \\ -1 & \text{如果 } y_i = -1, \hat{y}_i = -1, \text{预测正确} \\ 1 & \text{如果 } y_i = 1, \hat{y}_i - 1, \text{预测错误} \\ 1 & \text{如果 } y_i = -1, \hat{y}_i = 1, \text{预测错误} \end{cases}$$

那么由此得出的测量方程 (cost function) 为:

$$c_i = \max\{0, -(y_i \times \hat{y}_i)\} = \begin{cases} 0 & \text{如果 } y_i = 1, \hat{y}_i = 1, \text{预测正确} \\ 0 & \text{如果 } y_i = -1, \hat{y}_i = -1, \text{预测正确} \\ 1 & \text{如果 } y_i = 1, \hat{y}_i - 1, \text{预测错误} \\ 1 & \text{如果 } y_i = -1, \hat{y}_i = 1, \text{预测错误} \end{cases}$$

感知器损失函数

那么最终的损失函数为,

$$L = \sum_{i=1}^{m} c_i = \sum_{i=1}^{m} \max\{0, -(y_i \times \hat{y}_i)\} = \sum_{i=1}^{m} \mathbb{1}[\text{预测错误}]$$

因为损失函数统计的是预测错误的情况，所以我们希望最小化该损失函数，因为当预测错误的情况越少时，准确率越高，模型表现越好:

$$\begin{aligned}\min_{w} L = \sum_{i=1}^{m} c_i &= \sum_{i=1}^{m} \max\{0, -(y_i \times \hat{y}_i)\} \\ &= \sum_{i=1}^{m} \mathbb{1}[\text{预测错误}] \\ &= \sum_{i=1}^{m} \max\{0, -y_i \times (x_i \cdot w)\}\end{aligned}$$

理解梯度下降法 (Gradient Descent)

一旦确定了我们的损失函数后，我们希望最小化我们的函数，这样损失越小，准确率越高:

$$\begin{aligned}\min_{w} L &= \sum_{i=1}^{m} c_i = \sum_{i=1}^{m} \max\{0, -(y_i \times \hat{y}_i)\} \\ &= \sum_{i=1}^{m} \mathbb{1}[\text{预测错误}] \\ &= \sum_{i=1}^{m} \max\{0, -y_i \times (x_i \cdot w)\}\end{aligned}$$

- 优化问题需要借助导数的概念
- 天不生仲尼 (牛顿) 亘古如长夜

理解梯度下降法 (Gradient Descent)

定义

设有定义域和取值在实数域中的函数 $y = f(x)$。若 $f(x)$ 在点 x_0 的某个邻域内有定义，则当自变量 x 在 x_0 处取得增量 Δx 时，y 相应地取得增量 $\Delta y = f(x_0 + \Delta x) - f(x_0)$；当 Δy 与 Δx 之比在 $\Delta x \to 0$ 时极限存在，则称函数 $y = f(x)$ 在点 x_0 处可导，并称这个函数在 x_0 点处的导数, 记为 $f'(x_0)$, 即：

$$f'(x_0) = \lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}$$

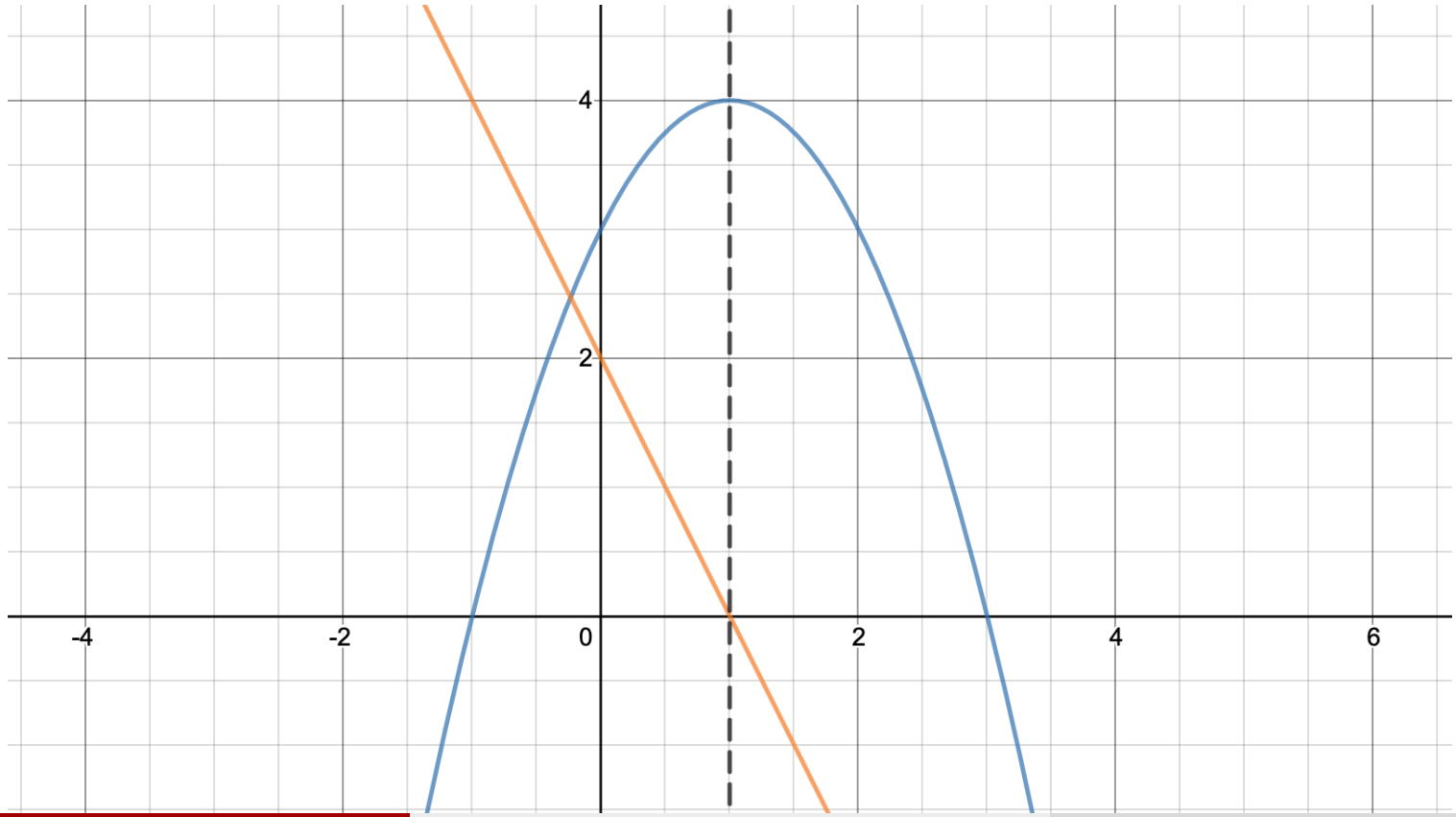
对以上定义，对于高中生来说，不需要太在意细节，我们需要重点了解其变化率的意涵。

理解梯度下降法 (Gradient Descent)

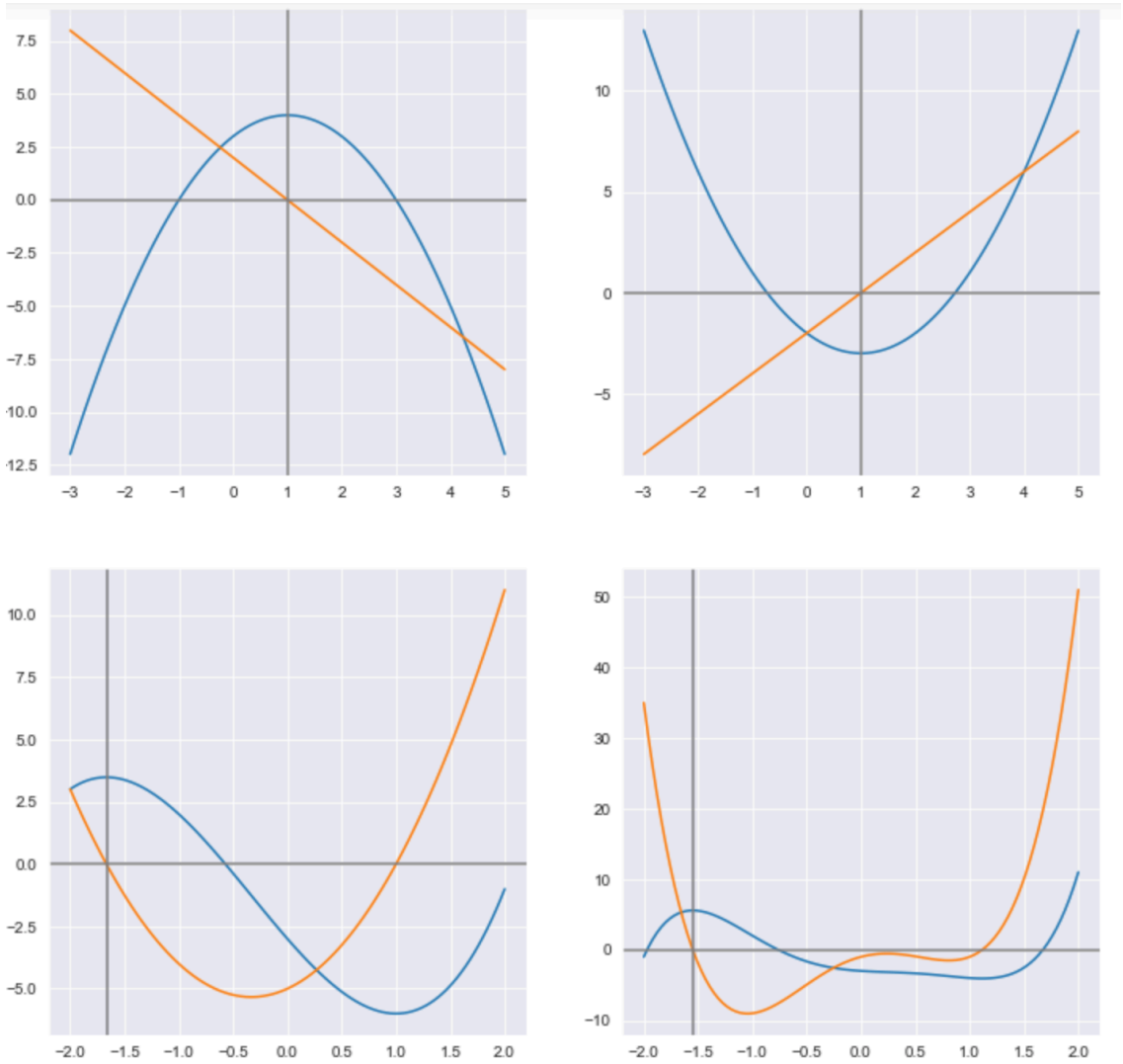
我们还是通过下面的例子来理解导数变化率的概念:

$$f(x) = -x^2 + 2x + 3$$

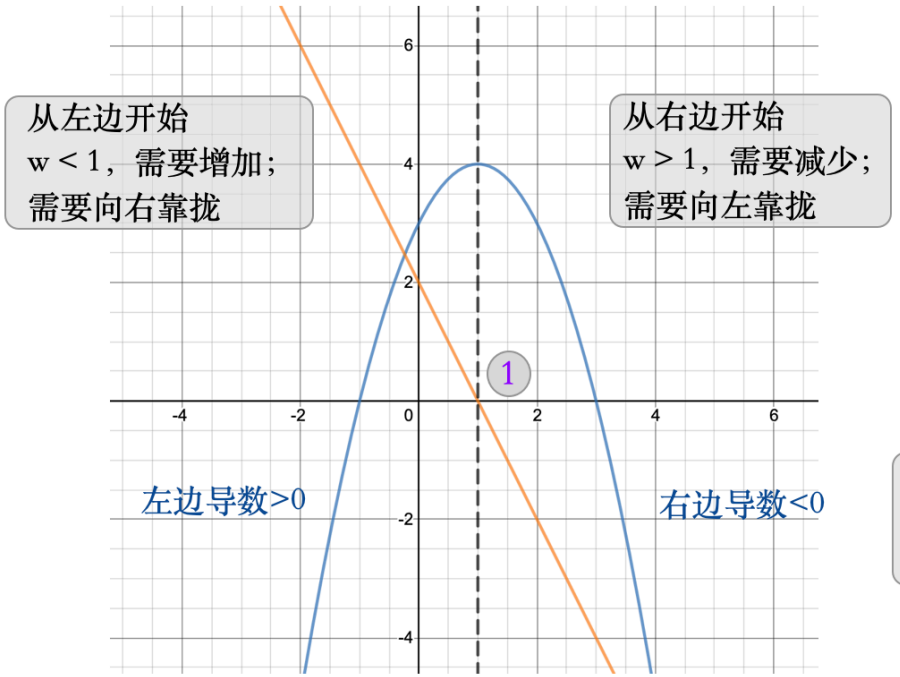
$$f'(x) = -2x + 2$$

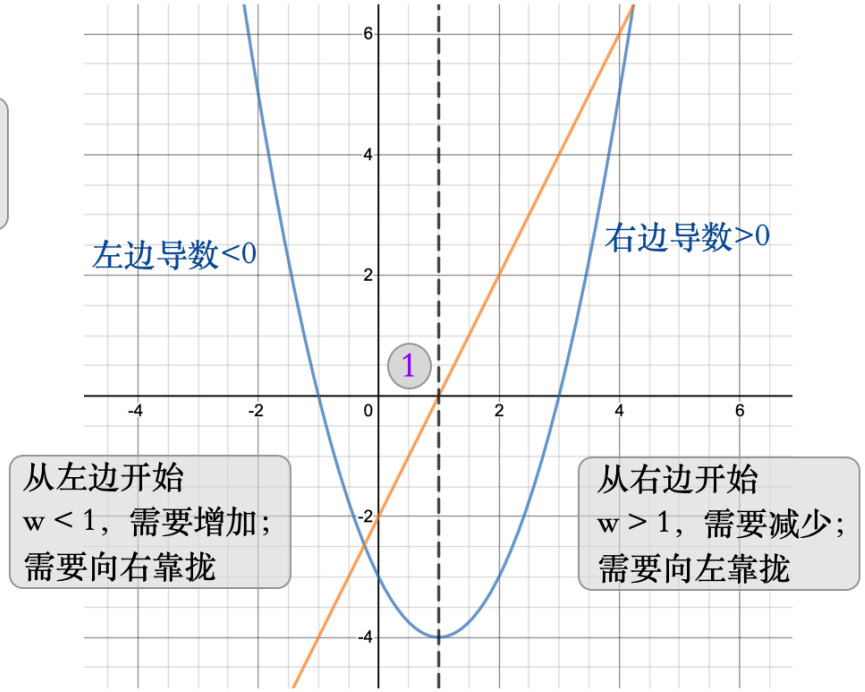


理解梯度下降法 (Gradient Descent)



理解梯度下降法 (Gradient Descent)





$w_{t+1} = w_t + \alpha f'(w)$

正值

w 增加，向右靠拢

$w_{t+1} = w_t + \alpha f'(w)$

负值

w 减少，向左靠拢

$w_{t+1} = w_t - \alpha f'(w)$

负值

w 增加，向右靠拢

$w_{t+1} = w_t - \alpha f'(w)$

正值

w 减少，向左靠拢

梯度下降法 (Gradient Descent)

定义

选定初始值 θ_0, 梯度下降法 (Gradient Descent) 通过下面的公式去逐步计算 $\theta_1, \theta_2, \cdots$, 直到达到我们满意的结果 (可以是计算的次数，也可以是计算的误差):

$$\theta_{t+1} = \theta_t - \alpha \nabla f(\theta_t)$$

其中，α 是为更新参数 (learning rate), $\nabla f(\theta_t)$ 为公式 $f(\cdot)$ 在 θ_t 时的导数值。因为在机器学习领域，$f(\cdot)$ 为我们所定义的损失函数 (Loss function), 所以我们希望最小化损失，因此是

$$\theta_{t+1} = \theta_t - \alpha \nabla f(\theta_t)$$

而不是

$$\theta_{t+1} = \theta_t + \alpha \nabla f(\theta_t)$$

梯度下降法 (Gradient Descent)

梯度下降法的一般公式为 (因为多数情况下是求最小值):

$$\theta_{t+1} = \theta_t - \alpha \nabla f(\theta_t)$$

注意更新参数 (learning rate) α 的选择对于我们找到期望的 θ 值有比较大的影响。一般可概括为:

- α 较小时，需要多次计算，θ 的更新较慢;
- α 较大时，计算次数下降，但是每次更新可能会错过期望值。

具体到 α 的选择时，我们后面在结合不同的案例时，还会进一步讲解。

感知器二元分类算法

在理解了什么是损失函数和梯度下降法后，我们来学习感知器分类的具体算法。这个算法的核心要点为:

- 假设分类问题可线性分类，即符合下列分类过程

$$\hat{y} = \begin{cases} +1 & x \cdot w + b > 0 \\ -1 & x \cdot w + b < 0 \end{cases}$$

  其中 $\hat{y}$ 为我们再分类过程中需要标注的结果，x 为 $1 \times n$ 的向量，表示分类对象有 n 个属性，w 长度为 $1 \times n$，表示为分类属性的规律参数。注意我们如果把点积写成矩阵的相乘的话应该是 xw^T。

- 我们设定的损失函数为

$$L = \sum_{i=1}^{m} c_i = \sum_{i=1}^{m} \max\{0, -y_i \times (x_i \cdot w + b)\}$$

感知器二元分类算法

- 损失函数的导数为:$\frac{\partial L}{\partial w} = \{0, -y_i \times x_i\}$, 等同于

$$\frac{\partial L}{\partial w} = -y_i \times x_i; \quad \text{当 } x_i \cdot w < 0 \text{ 时}$$
$$\frac{\partial L}{\partial w} = 0; \quad \text{当 } x_i \cdot w > 0 \text{ 时}$$

- 根据梯度下降法的公式 (求最小值), 我们对规律参数的更新公式为:

$$\begin{aligned} w_{t+1} &= w_t - \alpha \frac{\partial L}{\partial w} \\ &= w_t + \alpha(y_i \times x_i) \qquad (\text{当 } x_i \cdot w < 0) \end{aligned}$$

感知器二元分类算法

教材中的公式，假设我们的数据阵中有两个属性和一个自变量 b，那么我们的算法就变为:

- 假设分类问题可线性分类，即符合下列分类过程

$$\hat{y} = \begin{cases} +1 & x^{[1]}a_1 + x^{[2]}a_2 + b > 0 \\ -1 & x^{[1]}a_1 + x^{[2]}a_2 + b < 0 \end{cases}$$

- 我们设定的损失函数为

$$L = \sum_{i=1}^{m} c_i = \sum_{i=1}^{m} \max\{0, -y_i \times (x_i^{[1]}a_1 + x_i^{[2]}a_2 + b_i)\}$$

- 损失函数的导数为:$\frac{\partial L}{\partial a_1} = \{0, -y_i \times x_i^{[1]}\}$, 等同于

$$\frac{\partial L}{\partial w} = -y_i \times x_i^{[1]}; \quad \text{当 } x^{[1]}a_1 + x^{[2]}a_2 + b > 0 \text{ 时}$$

$$\frac{\partial L}{\partial w} = 0; \quad \text{当 } x^{[1]}a_1 + x^{[2]}a_2 + b < 0 \text{ 时}$$

感知器二元分类算法

- 根据梯度下降法的公式 (求最小值), 我们对规律参数的更新公式为:

$$
\begin{aligned}
a_{1,t+1} &= a_{1,t} - \alpha \frac{\partial L}{\partial a_1} \\
&= a_{1,t} + \alpha(y_i \times x_i^{[1]}) \qquad (\text{当 } x^{[1]}a_1 + x^{[2]}a_2 + b > 0 \text{ 时})
\end{aligned}
$$

- 相应得对于其它的参数的更新公式为:

$$
\begin{aligned}
a_{2,t+1} &= a_{2,t} + \alpha(y_i \times x_i^{[2]}) \\
b_{t+1} &= b_t + \alpha y_i
\end{aligned}
$$

感知器二元分类算法存在的问题

这里我们特别提醒同学们在使用感知线性分类模型时需要注意的几个问题:

- 当可分类对象存在线性分类规律时，该模型和算法给出的规律参数并不唯一，即存在多个集合解；
- 根据你数据阵的大小以及学习参数 (learning rate) 的选择不同，以上算法的运行时间会非常长
- 当可分类对象不存在线性分类时，该算法可能会陷入 '无限循环' 且仍无解的情况。

支持向量机 (Support Vector Machine)

支持向量机的发展:

- Vladimir Vapnik (1960s), 博士论文，心理学测量
- Vladimir Vapnik (1990s), AT&T (American Telephone and Telegraph Company)
- Cortes and Vapnik (1995) 发表文章，介绍了核方法 (kernel), 之后该模型便 '一发不可收'

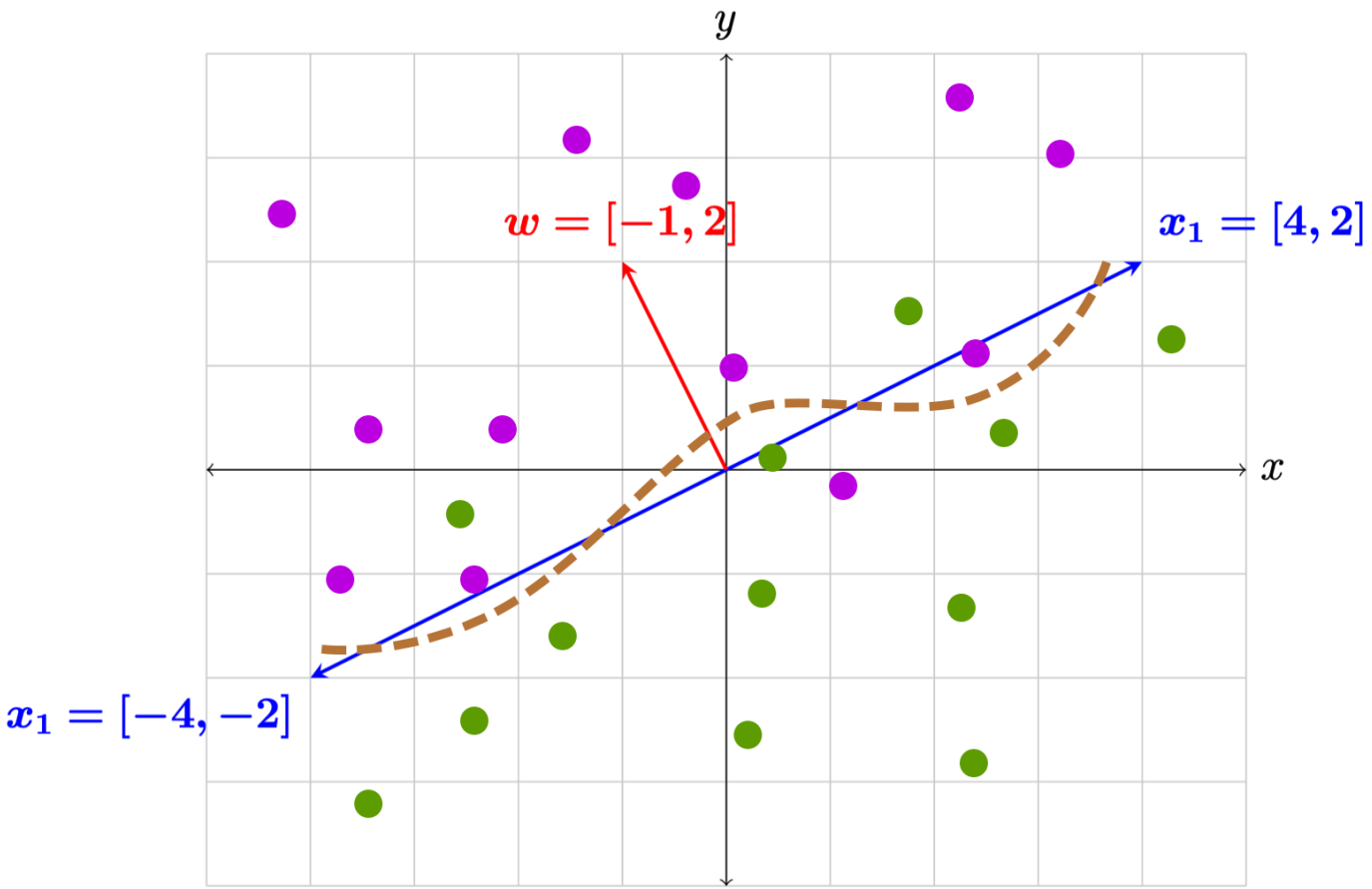
"[SVM] needs to be in the tool bag of every civilized person."
- Professor Dr. Patrick Winston, MIT

"Wow! This topic is totally devoid of any statistical content"
- -Dr. David Dickey, NCSU

支持向量机 (SVM)

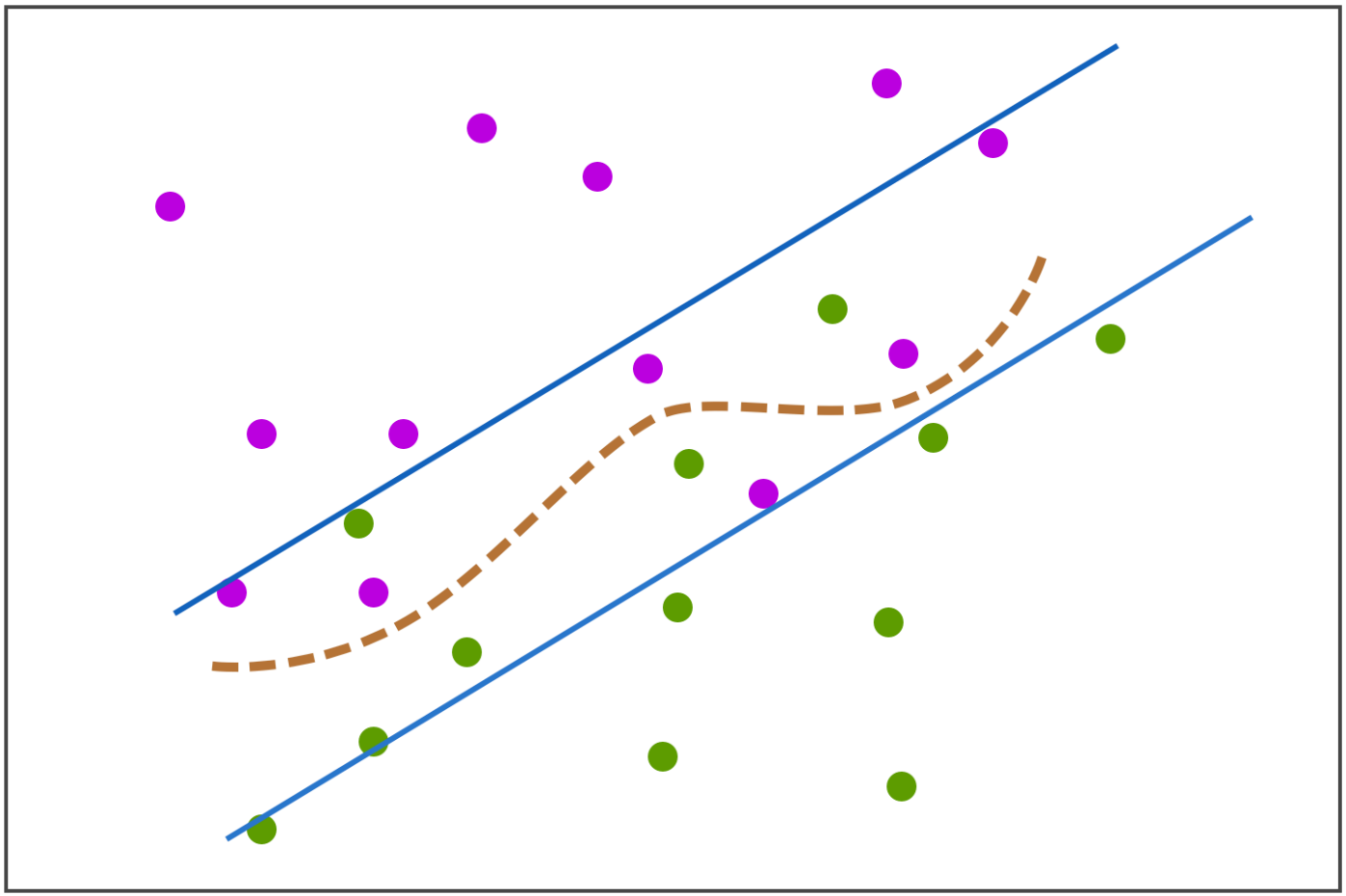
在学习感知分类器时，我们假定了直线分界面的存在，但是在现实生活中的很多例子，

- 要么该直线分界面不存在
- 要么该分界面并非完全线性



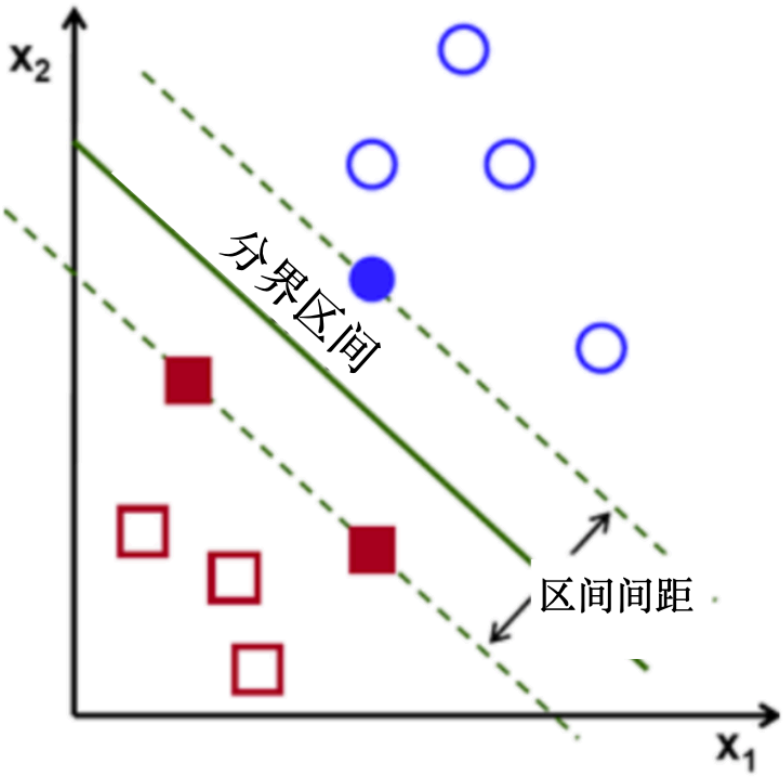
支持向量机 (SVM)

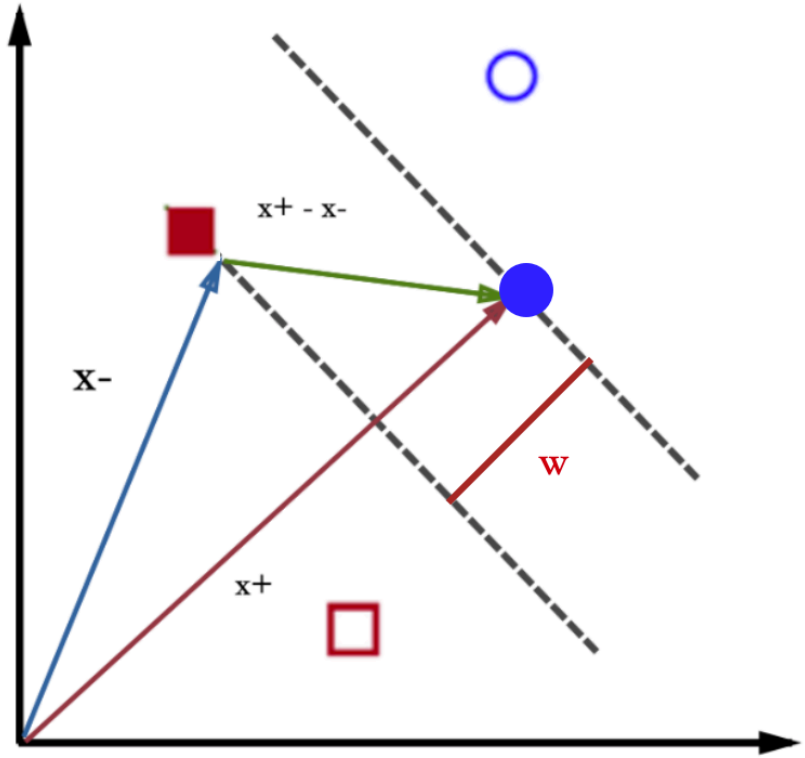
解决这个问题的方法是，将原来的直线分界面变成一个直线分界区间



支持向量机 (SVM)

我们想要分界区间的宽度越大越好，因为分界区间越宽，说明越容易分类。





支持向量机 (SVM): 测量分界面宽度

现在我们的问题就是如何测量 这个分界面的宽度。这里的测量指的是测量向量的长度。

定义

跟定长度为 n 的向量 x，其长度被定义为:

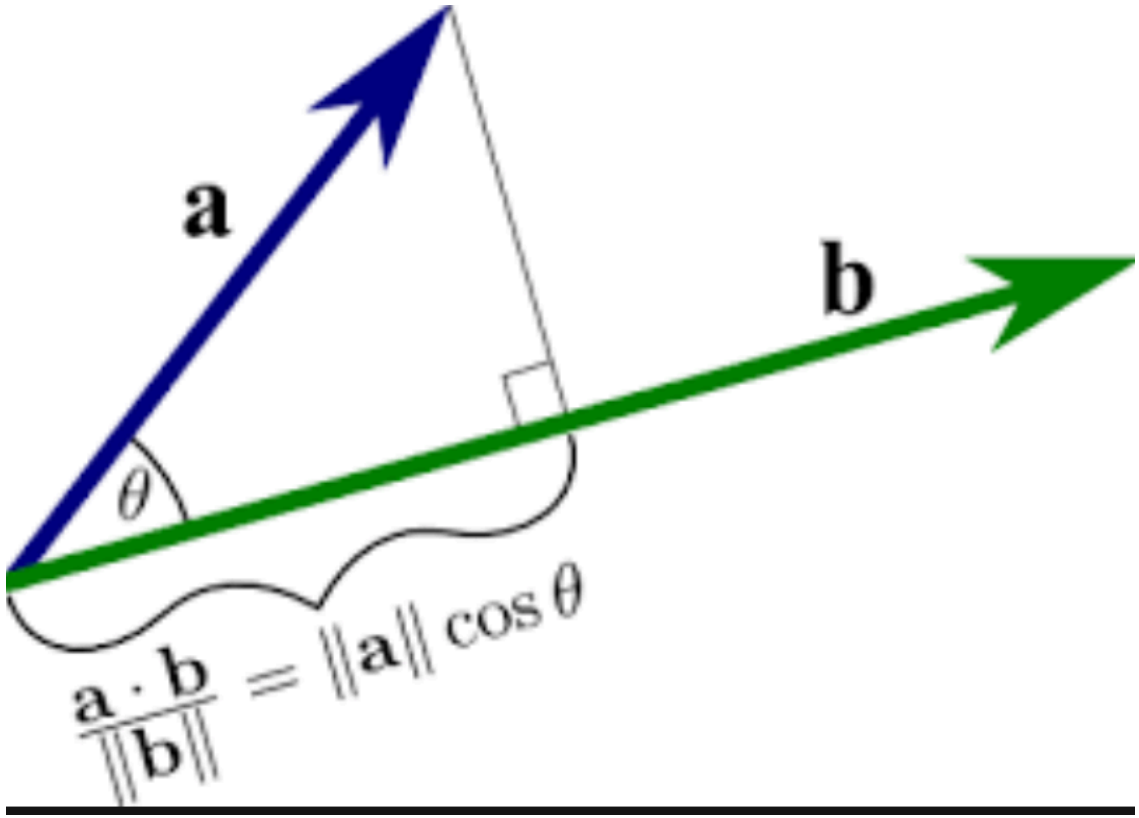
$$||x|| = \sqrt{< x, x >} = \sqrt{x \cdot x} = \sqrt{\sum_{i=1}^{n} x_i^2}$$

由此我们也可以得出下面的公式:

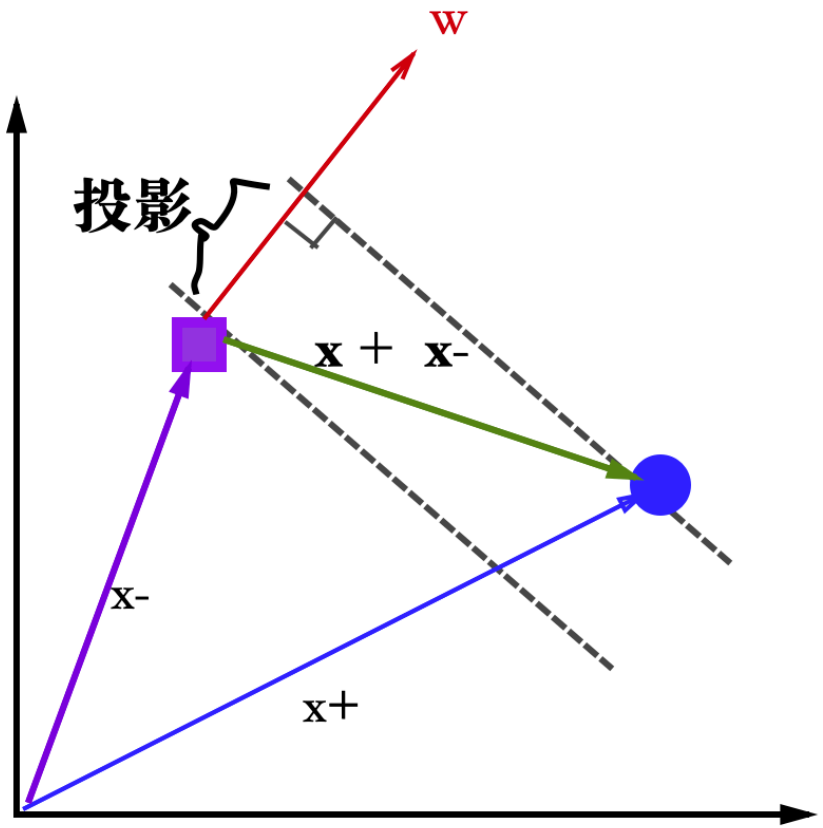
$$||x||^2 =< x, x >= x \cdot x = \sum_{i=1}^{n} x_i^2$$

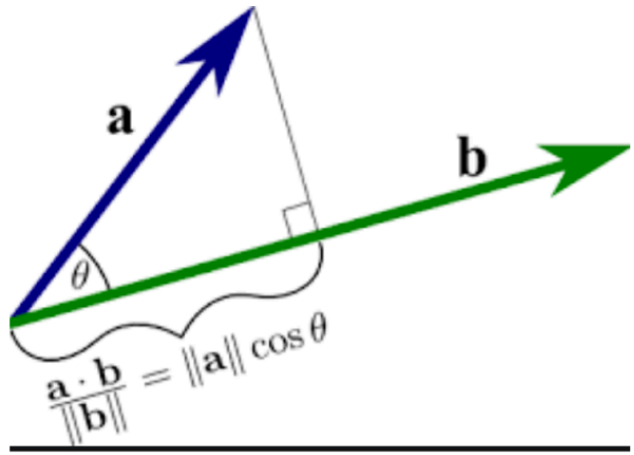
点积的投影概念

$$a \cdot b = ||a||||b|| \cos\theta; \quad a_{proj} \perp b = \frac{a \cdot b}{||b||} = ||a|| \cos\theta$$



支持向量机 (SVM): 测量分界面宽度





$$\text{分界区间宽度} = (x_+ - x_{-1}) \cdot \frac{w}{||w||}$$

$$a \cdot b = ||a||||b|| \cos\theta; \quad a_{proj} \perp b = \frac{a \cdot b}{||b||} = ||a|| \cos\theta$$

支持向量机 (SVM) 模型

支持向量机的整体思路与感知分类器十分类似，只是从一条线变成了一个分界区间。理解了点积投影的概念之后，我们就来构造支持向量机 (SVM) 模型:

- 仍然假设可分类对象为线性分类，且分类过程为二分过程，其中规律参数 w 为我们需要计算的数值，而 x 则是由我们采集的数据组成特征变量，分类过程表示为下:

$$\hat{y} = \begin{cases} +1 & x \cdot w + b > 0 \\ -1 & x \cdot w + b < 0 \end{cases}$$

其中 $\hat{y}$ 为我们在进行分类过程中需要标注的结果。因为 $\hat{y}$ 和 $x \cdot w$ 的正负值相同，所以我们可以将上面的分类过程表达为:

$$y(w \cdot x + b) \geq 1$$

支持向量机 (SVM) 模型

- 当 x_i 满足下列公式时:

$$y_i(w \cdot x_i + b) = 1; \quad y_i(w \cdot x_i + b) - 1 = 0; \quad \forall x_i \text{ 在分界区间内}$$

我们将 x_i 组成的分界区间和正交 (垂直) 向量 w 称为支持向量。

- 假设我我们有 x_+ 和 x_- 分别在分界区间的两条边界或者边界外上，那么分界区间的宽度可以由以下公式测量得出，

$$\text{分界区间宽度} = (x_+ - x_-) \cdot \frac{w}{||w||}$$

- 另外根据我们的分类过程的设定，我们有以下条件:

$$\begin{aligned} 1(w \cdot x_+ + b) = 1; \quad &\Rightarrow \quad x_+ \cdot w = 1 - b \\ -1(w \cdot x_- + b) = 1; \quad &\Rightarrow \quad -w \cdot x_- = 1 + b \end{aligned}$$

上面两个公式左右相加可以得到:

$$(x_+ - x_-) \cdot w = 2$$

支持向量机 (SVM) 模型

- 将公式

$$(x_+ - x_{-1}) \cdot w = 2$$

代入到下列分界区间宽度公式，可得

$$\begin{aligned} \text{分界区间宽度} &= (x_+ - x_-) \cdot \frac{w}{||w||} \\ &= \frac{2}{||w||} \end{aligned}$$

- 我们想要尽可能的扩大我们的分界区间宽度，

$$\max \text{ 分界区间宽度} = \frac{2}{||w||}; \quad \Leftrightarrow \quad \min \frac{1}{2}||w||^2$$

支持向量机 (SVM) 模型

那么，我们的优化问题就可以表示为:

$$\min_{w} \ \frac{1}{2}||w||^2$$
$$\text{s.t.} \ \ y_i(w \cdot x_i + b) \geq 1$$

因为之前的模型我们假定了这个直线分界面的存在，但是具体到实际应用中，这条假定的直线并不存在，因此我们在我们的模型中添加一定的容错率 ξ, 那么我们的优化问题就变成了:

$$\min_{w,\xi} \frac{1}{2}||w||^2 + C\sum_{i}^{m} \xi_i$$
$$\text{s.t.} \ \ y_i(w \cdot x_i + b) \geq 1 - \xi_i$$

对该问题的求解需要使用 Lagrange 方程，这里我们就不做详细的解释了。以上就是支持向量机的线性分类过程。

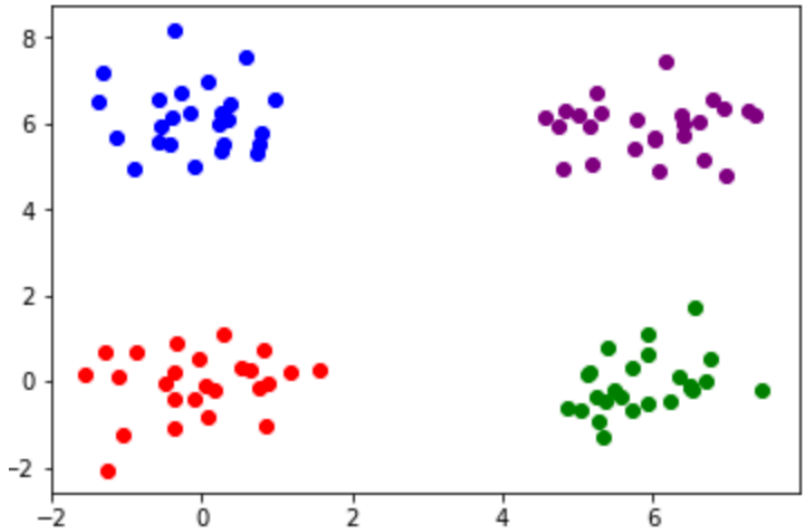
支持向量机: 核方法

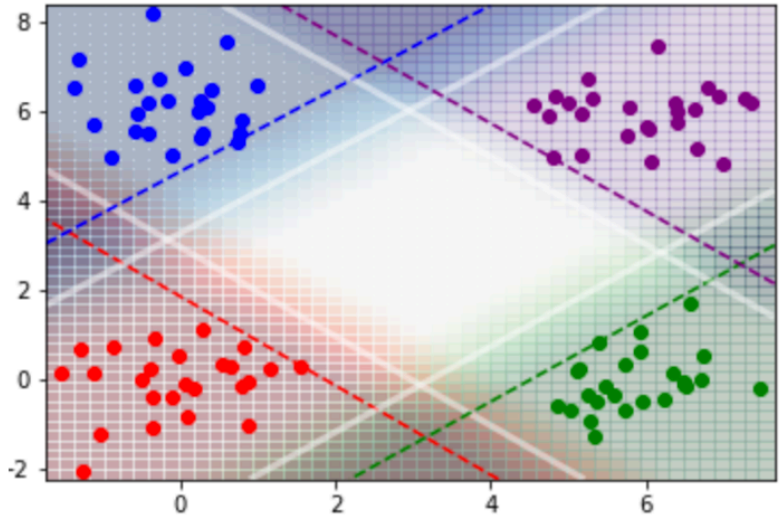
如果想要使用支持向量机 (SVM) 进行分线性分类，我们需要对特征变量 X 进行非线性转换，常用的非线性转换包括:

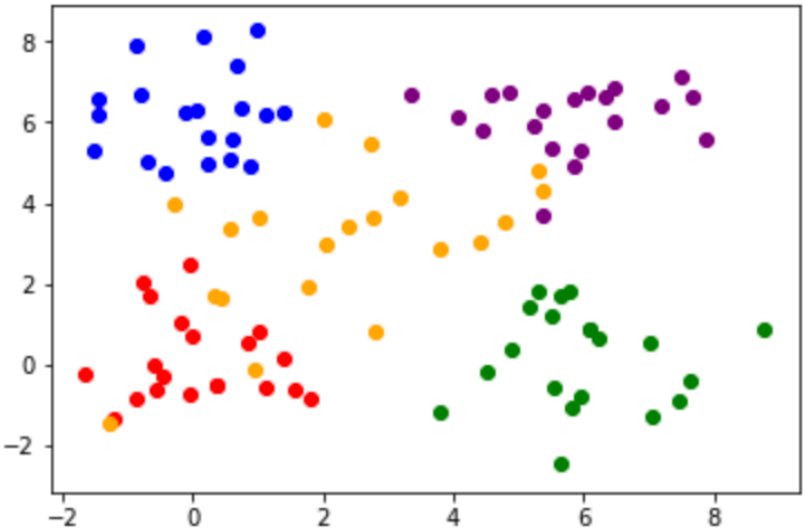
- 多项式转换: $K(x, x') = (1 + < x, x' >)^d$
- 指数转换: $K(x, x') = \exp(-\gamma||x - x'||^2)$
- 神经网络转换: $K(x, x') = \tanh(\kappa_1 < x, x' > +\kappa_2)$

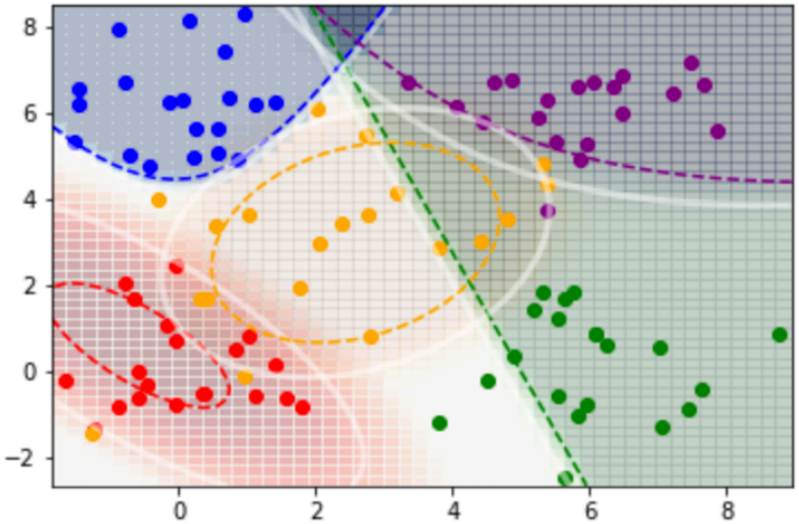
对这些 (kernles) 方法的转换，我们并不做要求，同学们需要了解下，这些核方法，因为后续我们还会见到这些方程。

支持向量机: 核方法









再见

Thank you！请完成课后习题 C3！

Reference I

MIT OpenCourserWare:
https://www.youtube.com/watch?v=_PwhiWxHK8o&t=50s
Stanford Online:
https://www.youtube.com/watch?v=8NYoQiRANpg&t=335s
Notes written by myself:
https://github.com/Michael-yunfei/CS229/blob/master/Notes/Support_Vector_Machines.pdf