《神经网络与深度学习》

线性分类模型

https://nndl.github.io/

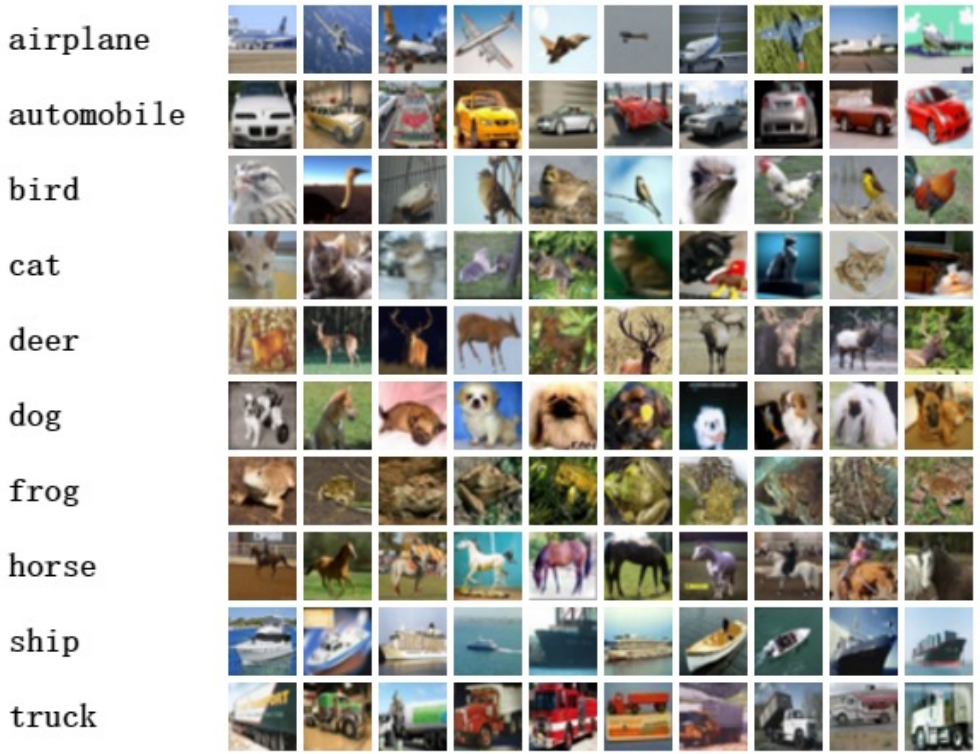
大纲

- 分类问题示例
- 线性分类模型
  - Logistic Regression
  - Softmax Regression
  - Perceptron
  - SVM

分类示例

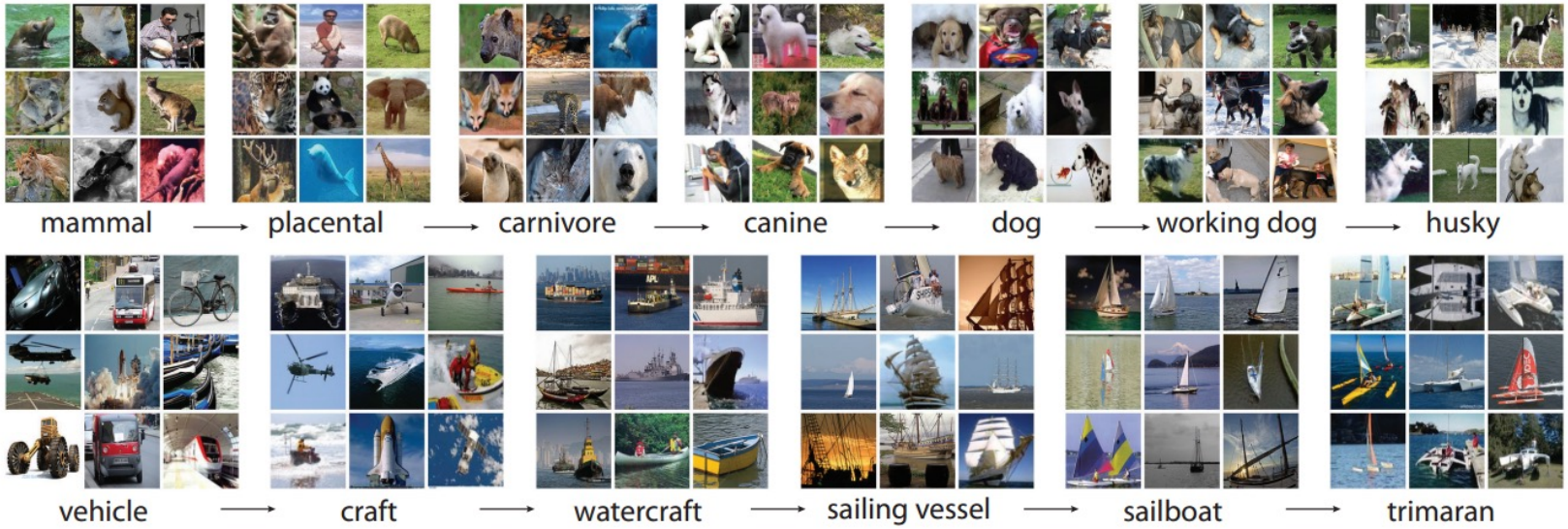
示例：图像分类 (Image Classification)

- 数据集：CIFAR-10
  - 60000张32x32色彩图像，共10类
  - 每类6000张图像



示例：图像分类

- 数据集： ImageNet
  - 14,197,122 images, 21841 synsets



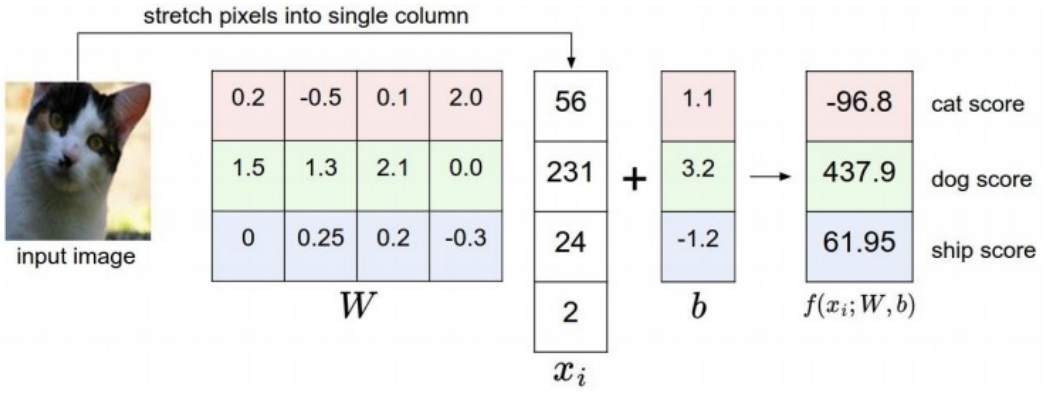
示例：图像分类

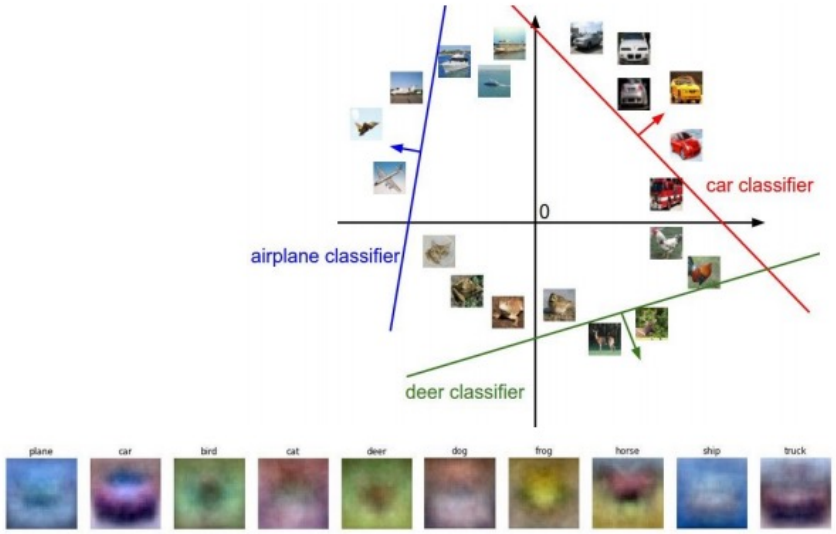
[32x32x3]
array of numbers 0...1
(3072 numbers total)

image parameters

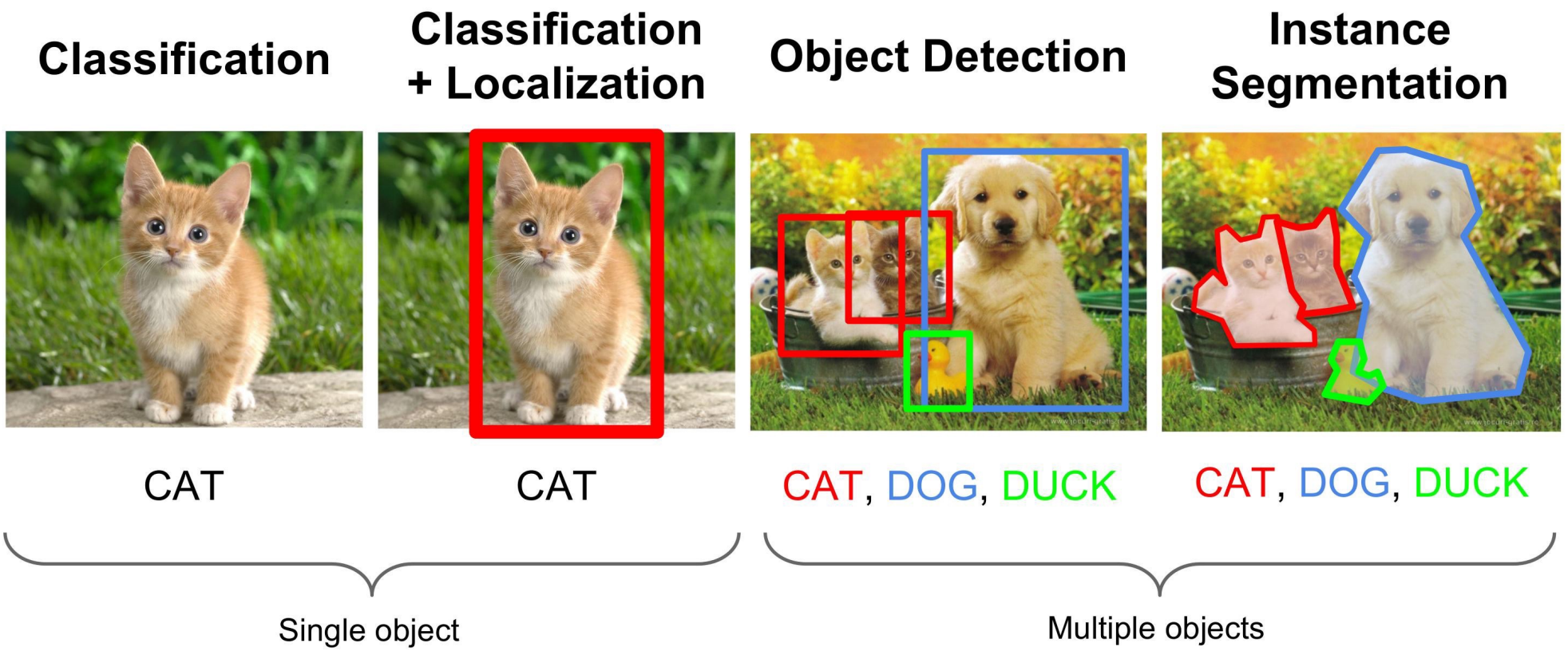
$f(x,W)$

10 numbers, indicating class scores



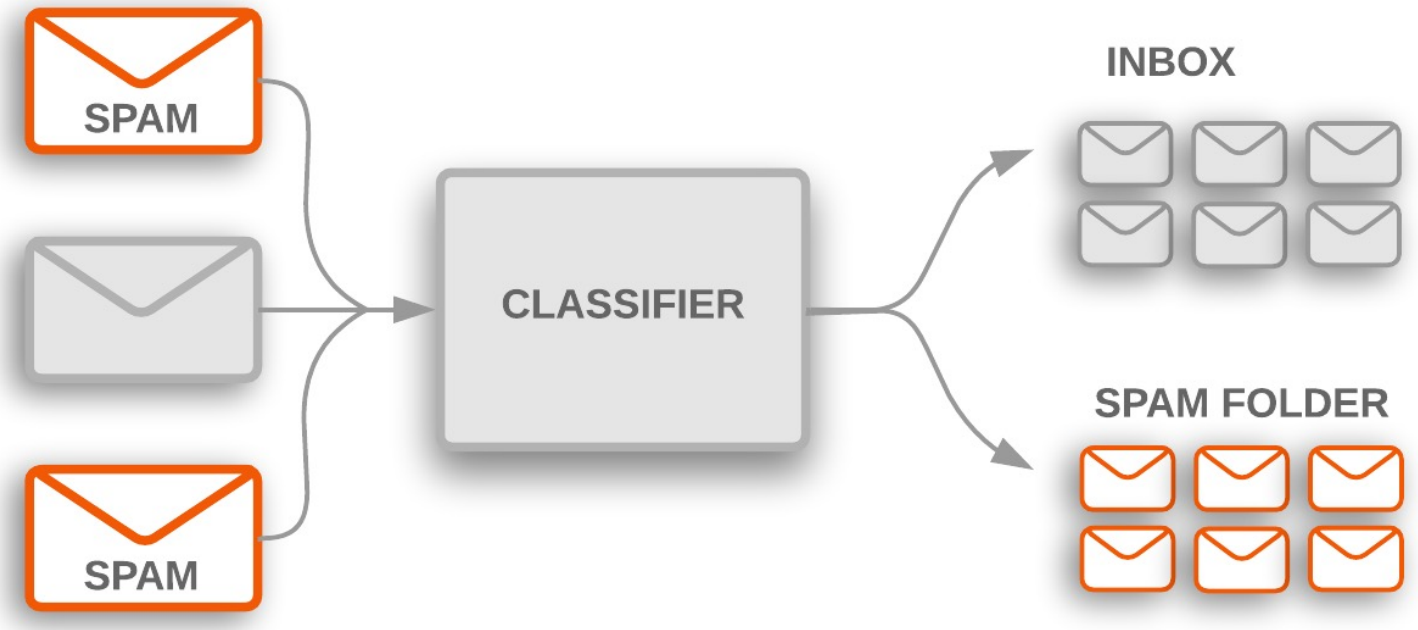


示例：图像分类、目标检测、实例分割



https://medium.com/zylapp/review-of-deep-learning-algorithms-for-object-detection-c1f3d437b852

示例：垃圾邮件过滤 (Spam Detection)

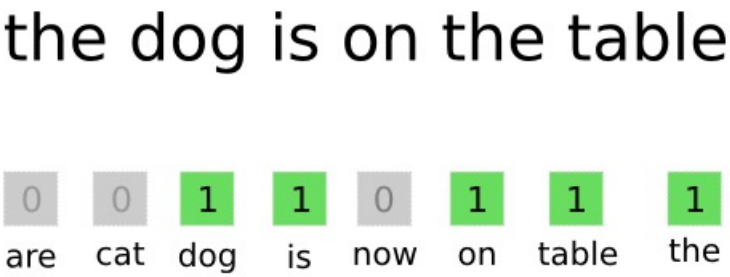


示例：文本分类 (Text Classification)

https://towardsdatascience.com/automated-text-classification-using-machine-learning-3df4f4f9570b

示例：文本分类

- 将样本x从文本形式转为向量形式
  - 词袋模型（Bag-of-Words，BoW）模型

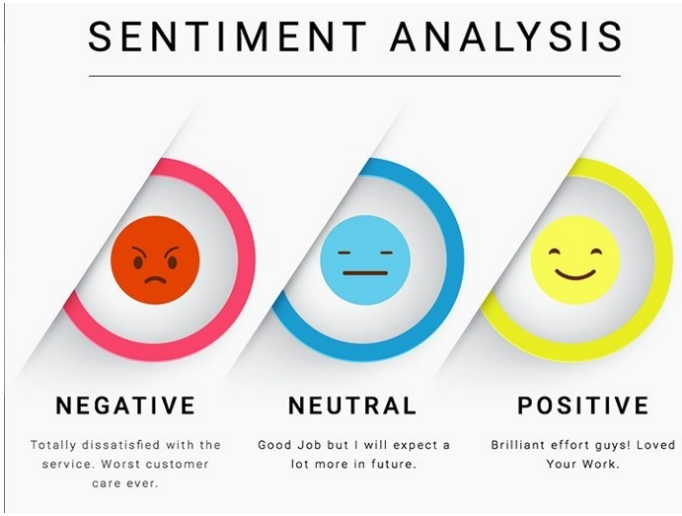


比如两个文本“我 喜欢 读书”和“我 讨厌 读书”中共有“我”、“喜欢”、“讨厌”、“读书”四个词，它们的BoW表示分别为

$$\mathbf{v}_1 = [1\ 1\ 0\ 1\ ]^{\mathrm{T}},$$

$$\mathbf{v}_2 = [1\ 0\ 1\ 1\ ]^{\mathrm{T}}.$$

示例：情感分类 (Sentiment Classification)



Review (X)	Rating (Y)
"This movie is fantastic! I really like it because it is so good!"	★★★★☆
"Not to my taste, will skip and watch another movie"	★★☆☆☆
"This movie really sucks! Can I get my money back please?"	★☆☆☆☆

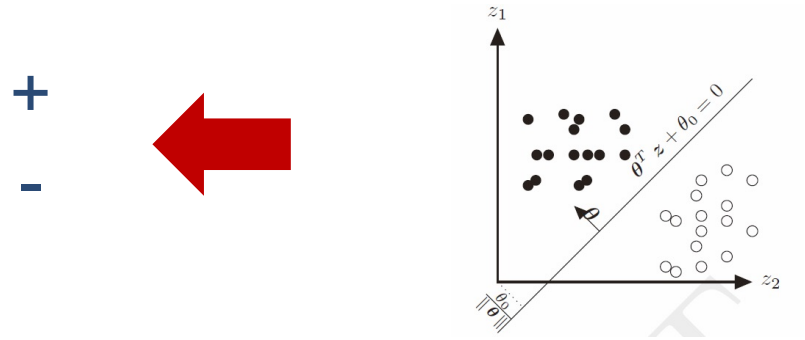
示例：情感分类

根据文本内容来判断文本的相应类别

D_1：“我喜欢读书”

D_2：“我讨厌读书”

	我	喜欢	讨厌	读书
D_1	1	1	0	1
D_2	1	0	1	1

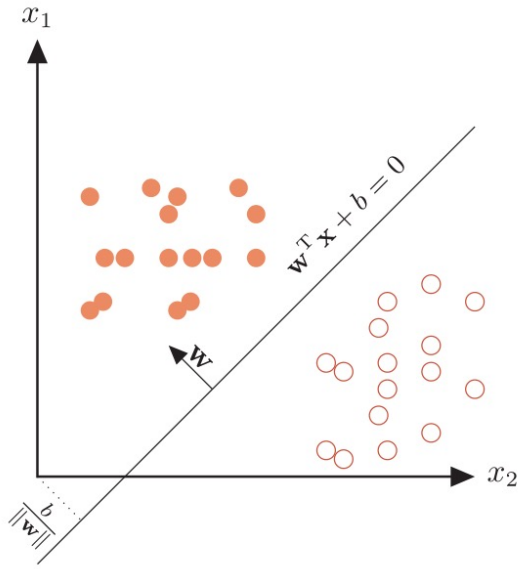


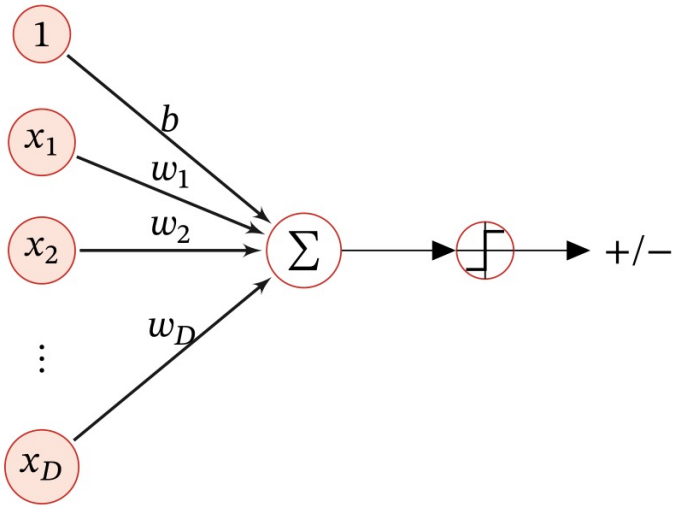
\+

\-

线性分类模型

线性模型


x_1
$\mathbf{w}^{\mathrm{T}}\mathbf{x}+b=0$
$\mathbf{w}$
x_2
$\frac{b}{\|\mathbf{w}\|}$


1
x_1
x_2
x_D
b
w_1
w_2
w_D
Σ
+/−

线性分类模型

- Logistic回归
- Softmax回归
- 感知器
- 支持向量机

Logistic Regression

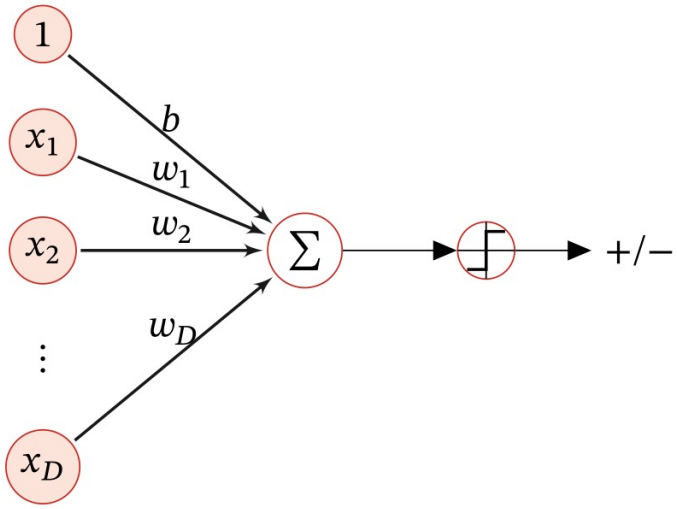
Logistic Regression

- 模型

$$g(f(\boldsymbol{x};\boldsymbol{w})) = \begin{cases} 1 & \text{if} \quad f(\boldsymbol{x};\boldsymbol{w}) > 0 \\ 0 & \text{if} \quad f(\boldsymbol{x};\boldsymbol{w}) < 0 \end{cases}$$

$g(x)$:决策函数
(decision function)

$f(x;w)$:判别函数
(discriminant function)



分类问题

- 将分类问题看作条件概率估计问题

  - 引入非线性函数g 来预测类别标签的条件概率$p(y=c|x)$。
  - 以二分类为例，

$$p(y=1|\mathbf{x})=g(f(\mathbf{x};\mathbf{w}))$$

  - 函数f：线性函数
  - 函数g：把线性函数的值域从实数区间“挤压”到了(0,1)之间，可以用来表示概率。

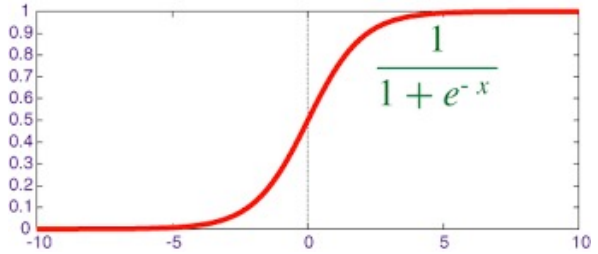
如何构建函数g?

Logistic函数与回归

Logistic函数

除了logisitc，还有什么函数$f: R \to (0,1)$?

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$



Logistic回归

$$p(y = 1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})$$

$$\triangleq \frac{1}{1 + \exp(-\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})}$$

学习准则

- 模型预测条件概率$p_\theta(y|\boldsymbol{x})$

$$p_\theta(y=1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^T\boldsymbol{x})$$

- 真实条件概率$p_r(y|\boldsymbol{x})$
  - 对于一个样本(x, y^*), 其真实条件概率为

$$p_r(y=1|\boldsymbol{x}) = y^*$$
$$p_r(y=0|\boldsymbol{x}) = 1 - y^*$$

如何衡量两个条件分布的差异?

熵（Entropy）

信息和数学的关系推荐阅读：吴军《数学之美》

- 在信息论中，熵用来衡量一个随机事件的不确定性。
  - 自信息（Self Information） $I(x) = -\log(p(x))$
  - 熵

$$
\begin{aligned}
H(X) &= \mathbb{E}_X[I(x)] \\
&= \mathbb{E}_X[-\log p(x)] \\
&= -\sum_{x \in X} p(x) \log p(x)
\end{aligned}
$$

假设一个随机变量 X 有三种可能值 x_1, x_2, x_3，不同概率分布对应的熵如下：

$p(x_1)$	$p(x_2)$	$p(x_3)$	熵
1	0	0	0
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{3}{2}\log 2$
$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\log 3$

  - 熵越高，则随机变量的信息越多；
  - 熵越低，则随机变量的信息越少 .

- 在对分布 q 的符号进行编码时，熵 H(q) 也是理论上最优的平均编码长度，这种编码方式称为熵编码（Entropy Encoding）

交叉熵（Cross Entropy）

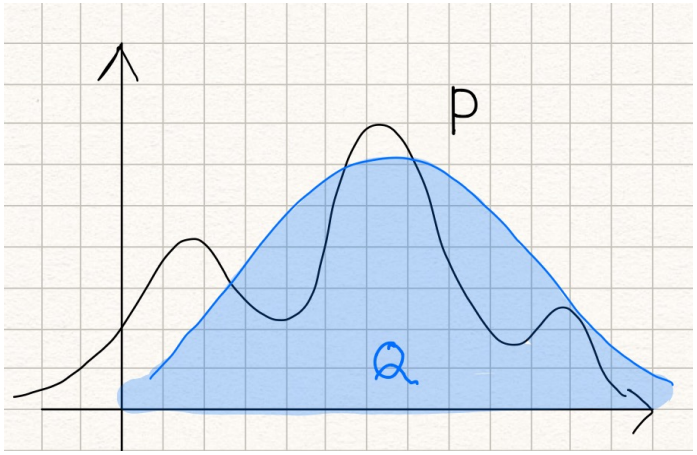
- 交叉熵是按照概率分布q的最优编码对真实分布为p的信息进行编码的长度。

$$H(p,q) = \mathbb{E}_p[-\log q(x)]$$
$$= -\sum_x p(x) \log q(x)$$

- 在给定 q 的情况下，如果p和 q 越接近，交叉熵越小；
- 如果p和 q 越远，交叉熵就越大.

KL散度（Kullback-Leibler Divergence）

- KL散度是用概率分布q来近似p时所造成的信息损失量。
  - KL散度是按照概率分布q的最优编码对真实分布为p的信息进行编码，其平均编码长度（即交叉熵）H(p,q)和p的最优平均编码长度（即熵）H(p)之间的差异。



$$\mathrm{KL}(p,q) = H(p,q) - H(p)$$

$$= \sum_x p(x) \log \frac{p(x)}{q(x)}$$

KL散度=交叉熵-真实分布的熵

$$\mathrm{D}_{KL}(p_r(y|x)||p_\theta(y|x)) = \sum_{y=0}^{1} p_r(y|x)\log\frac{p_r(y|x)}{p_\theta(y|x)}$$ KL散度

$$\propto -\sum_{y=0}^{1} p_r(y|x)\log p_\theta(y|x)$$ 交叉熵损失

y^*为x的真实标签

$$= -I(y^*=1)\log p_\theta(\mathrm{y}=1|x) - I(y^*=0)\log p_\theta(\mathrm{y}=0|x)$$

$$= -y^*\log p_\theta(\mathrm{y}=1|x) - (1-y^*)\log p_\theta(\mathrm{y}=0|x)$$

$$= -\log p_\theta(y^*|x)$$ 负对数似然

梯度下降

- 交叉熵损失函数，模型在训练集的风险函数为

$$\mathcal{R}(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)}\log\left(\sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-\sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(i)})\right)\right).$$

- 梯度为

$$\frac{\partial \mathcal{R}(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{x}^{(i)}\cdot\left(\sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(i)}) - y^{(i)}\right)\right)$$

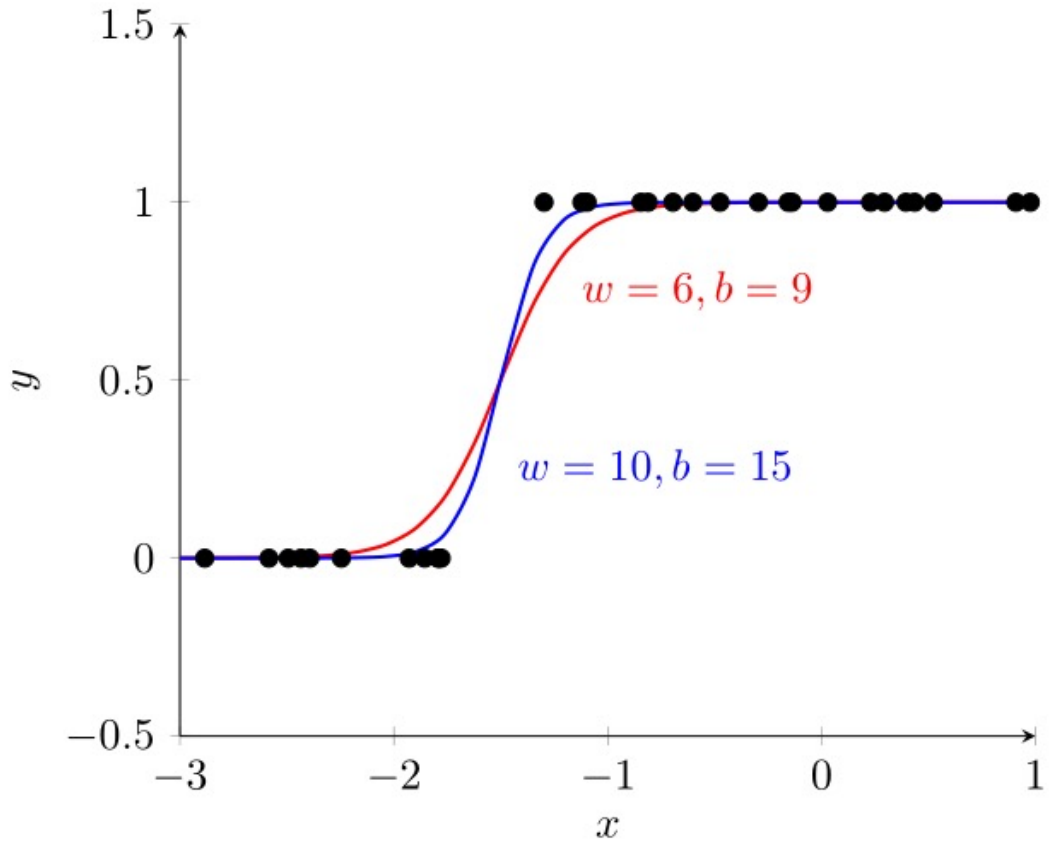
推导过程

$$\frac{\partial(\sigma(x))}{\partial x} = \sigma(x)(1-\sigma(x))$$

$$\mathcal{R}(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)}\log\left(\sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-\sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(i)})\right)\right).$$

$$\begin{aligned}\frac{\partial\mathcal{R}(\mathbf{w})}{\partial\mathbf{w}} &= -\frac{1}{N}\sum_{n=1}^{N}\left(y^{(n)}\frac{\hat{y}^{(n)}(1-\hat{y}^{(n)})}{\hat{y}^{(n)}}\mathbf{x}^{(n)} - (1-y^{(n)})\frac{\hat{y}^{(n)}(1-\hat{y}^{(n)})}{1-\hat{y}^{(n)}}\mathbf{x}^{(n)}\right)\\ &= -\frac{1}{N}\sum_{n=1}^{N}\left(y^{(n)}(1-\hat{y}^{(n)})\mathbf{x}^{(n)} - (1-y^{(n)})\hat{y}^{(n)}\mathbf{x}^{(n)}\right)\\ &= -\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}^{(n)}\left(y^{(n)}-\hat{y}^{(n)}\right).\end{aligned}$$

Logistic回归



Logistic回归和平方差损失

http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML17.html

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Step 2: Training data: (x^n, y^n), y^n: 1 for class 1, 0 for class 2

$$L(f) = \frac{1}{2}\sum_n \left(f_{w,b}(x^n) - y^n\right)^2$$

Step 3:

$$\frac{\partial\, (f_{w,b}(x)-y)^2}{\partial w_i} = 2\left(f_{w,b}(x) - y\right) \frac{\partial f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i}$$

$$= 2\left(f_{w,b}(x) - y\right) f_{w,b}(x)\left(1 - f_{w,b}(x)\right) x_i$$

$y^n = 1$ If $f_{w,b}(x^n) = 1$ (close to target) ➡ $\partial L/\partial w_i = 0$

If $f_{w,b}(x^n) = 0$ (far from target) ➡ $\partial L/\partial w_i = 0$

Logistic回归和平方差损失

http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML17.html

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Step 2: Training data: (x^n, y^n), y^n: 1 for class 1, 0 for class 2

$$L(f) = \frac{1}{2}\sum_n \left(f_{w,b}(x^n) - y^n\right)^2$$

Step 3:

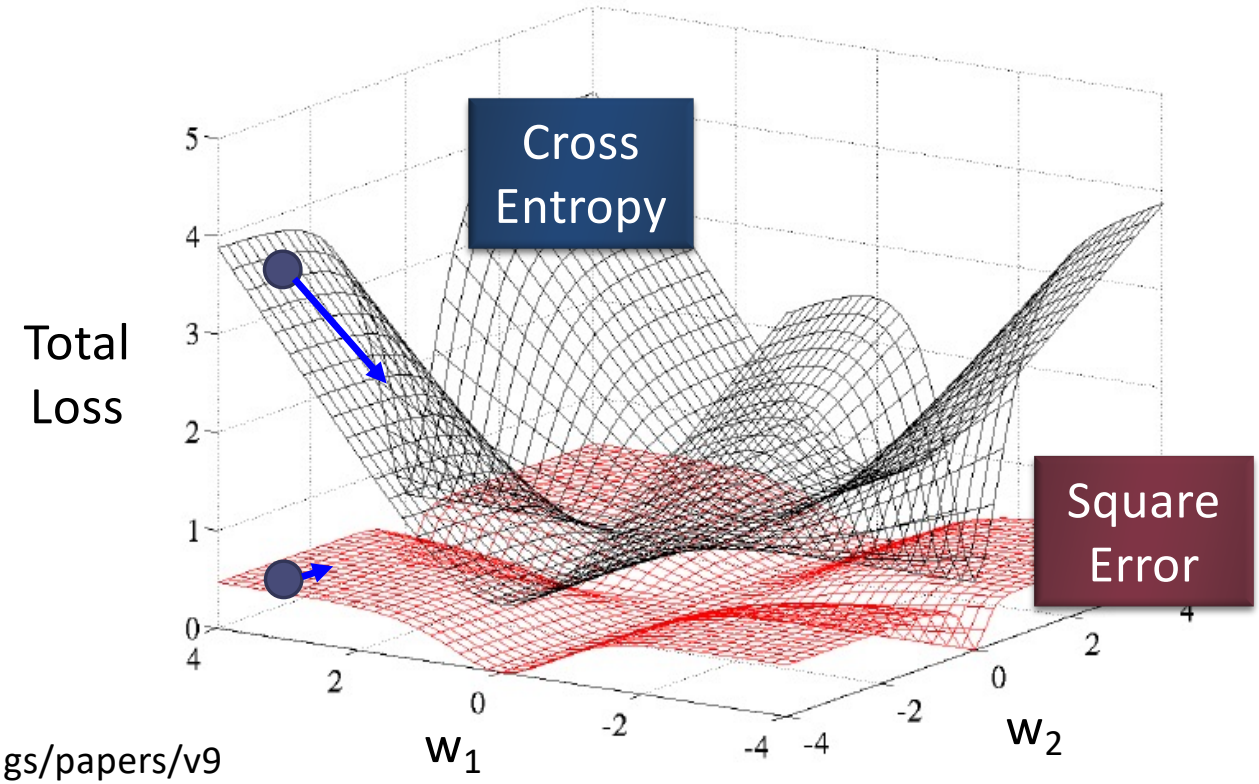
$$\frac{\partial\, (f_{w,b}(x)-y)^2}{\partial w_i} = 2\left(f_{w,b}(x) - y\right) \frac{\partial f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i}$$

$$= 2\left(f_{w,b}(x) - y\right) f_{w,b}(x)\left(1 - f_{w,b}(x)\right) x_i$$

$y^n = 0$ If $f_{w,b}(x^n) = 1$ (far from target) ➡ $\partial L/\partial w_i = 0$

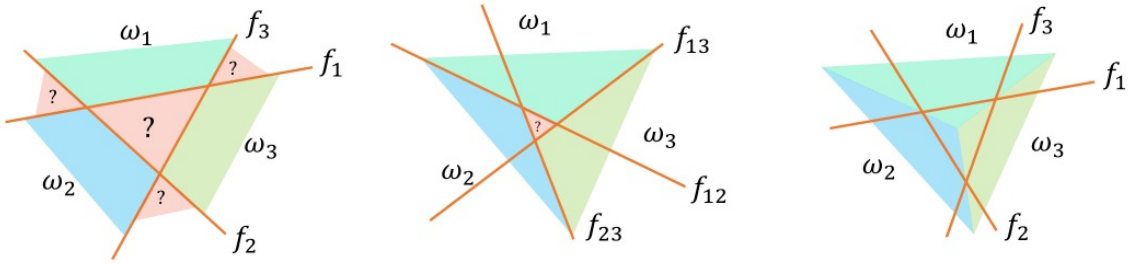
If $f_{w,b}(x^n) = 0$ (close to target) ➡ $\partial L/\partial w_i = 0$

交叉熵 v.s. 平方损失



http://jmlr.org/proceedings/papers/v9/glorot10a/glorot10a.pdf

多分类（Multi-class Classification）



(a) “一对其余”方式　　(b) “一对一”方式　　(c) “argmax”方式

“*argmax*”方式：这是一种改进的“一对其余”方式，共需要C个判别函数

$$f_c(\mathbf{x};\mathbf{w}_c)=\mathbf{w}_c^{\mathrm{T}}\mathbf{x}+b_c,\qquad c=[1,\cdots,C] \tag{3.10}$$

如果存在类别c,对于所有的其他类别$\tilde{c}(\tilde{c}\neq c)$都满足$f_c(\mathbf{x};\mathbf{w}_c)>f_{\tilde{c}}(\mathbf{x},\mathbf{w}_{\tilde{c}})$，那么$\mathbf{x}$属于类别$c$。即

$$y=\arg\max_{c=1}^{C} f_c(\mathbf{x};\mathbf{w}_c). \tag{3.11}$$

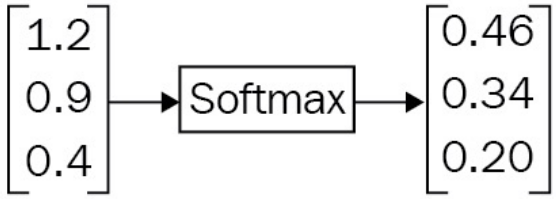
Softmax回归

- 多分类问题

$$y = \arg\max_{c=1}^{C} f_c(\boldsymbol{x}; \boldsymbol{w}_c)$$

- Softmax函数

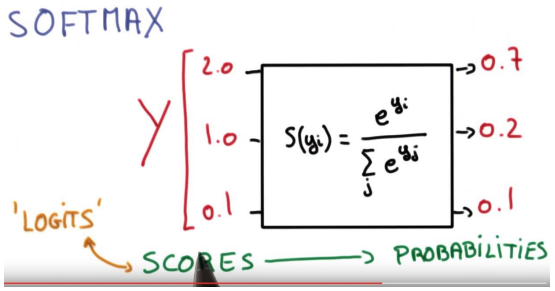
$$\text{softmax}(x_k) = \frac{\exp(x_k)}{\sum_{i=1}^{K} \exp(x_i)}$$



Softmax回归

- 利用softmax函数，目标类别y＝c的条件概率为：

$$P(y=c|\mathbf{x}) = \text{softmax}(\mathbf{w}_c^{\mathrm{T}}\mathbf{x}) = \frac{\exp(\mathbf{w}_c^{\top}\mathbf{x})}{\sum_{i=1}^{C}\exp(\mathbf{w}_i^{\top}\mathbf{x})}.$$



交叉熵损失

- KL散度

$$\mathrm{D}_{\mathrm{kl}}(p_r(y|x)||p_\theta(y|x)) = \sum_{y=1}^{C} p_r(y|x)\log\frac{p_r(y|x)}{p_\theta(y|x)}$$

$$\propto -\sum_{y=1}^{C} p_r(y|x)\log p_\theta(y|x)$$ 交叉熵损失

y^*为x的真实标签

$$= -\log p_\theta(y^*|x)$$ 负对数似然

参数学习

- 模型：Softmax回归
- 学习准则：交叉熵

$$\mathcal{R}(W) = -\frac{1}{N}\sum_{n=1}^{N}(\mathbf{y}^{(n)})^{\mathrm{T}}\log\hat{\mathbf{y}}^{(n)}$$

- 优化：梯度下降

$$\frac{\partial\mathcal{R}(W)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}^{(n)}\left(\mathbf{y}^{(n)}-\hat{\mathbf{y}}^{(n)}\right)^{\mathrm{T}}$$

交叉熵损失函数

- 负对数似然损失函数

$$\mathcal{L}(\mathbf{y}, f(\mathbf{x}, \theta)) = -\sum_{c=1}^{C} y_c \log f_c(\mathbf{x}, \theta)$$

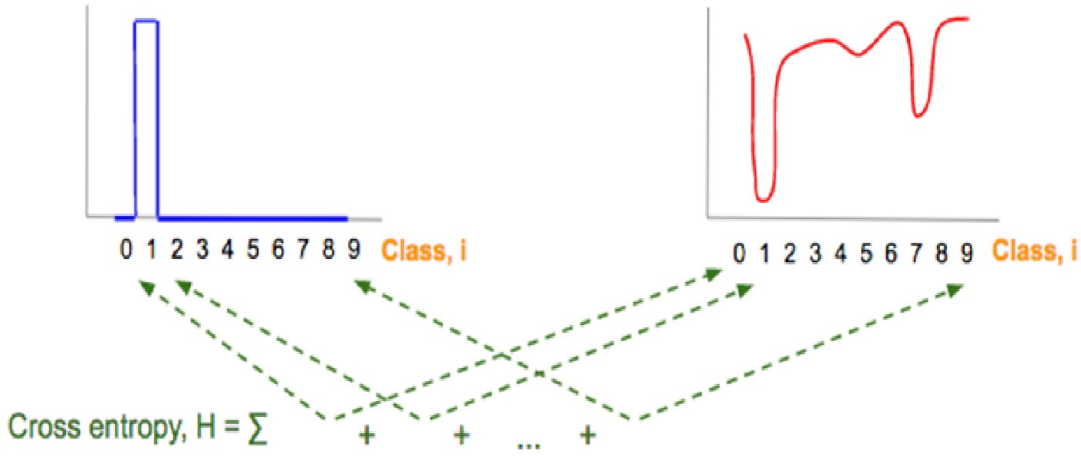
- 对于一个三类分类问题，类别为[0,0,1]，预测类别概率为[0.3,0.3,0.4]，则

$$\begin{aligned}\mathcal{L}(\theta) &= -\big(0 \times \log(0.3) + 0 \times \log(0.3) + 1 \times \log(0.4)\big) \\ &= -\log(0.4).\end{aligned}$$

交叉熵损失

$$-\sum_{y=1}^{C} p_r(y|x) \log p_\theta(y|x)$$

真实概率$p_r(y|x)$　　预测概率的负对数$-\log p_\theta(y|x)$



感知器

感知器

Psychological Review
Vol. 65, No. 6, 1958

THE PERCEPTRON: A PROBABILISTIC MODEL FOR INFORMATION STORAGE AND ORGANIZATION IN THE BRAIN[1]

F. ROSENBLATT

Cornell Aeronautical Laboratory

HAVING told you about the giant digital computer known as I.B.M. 704 and how it has been taught to play a fairly creditable game of chess, we'd like to tell you about an even more remarkable machine, the perceptron, which, as its name implies, is capable of what amounts to original thought. The first perceptron has yet to be built,

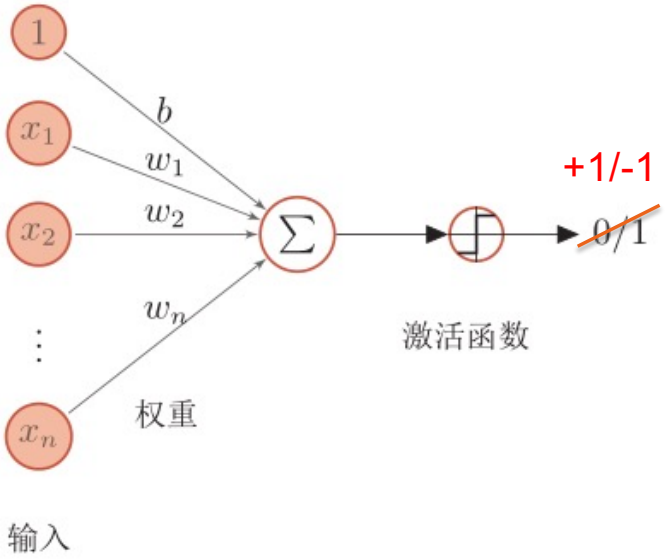
The New Yorker, December 6, 1958 P. 44

The IBM 704 computer

感知器

- 模拟生物神经元行为的机器，有与生物神经元相对应的部件，如权重（突触）、偏置（阈值）及激活函数（细胞体），输出为+1或-1。

$$\hat{y} = \begin{cases} +1 & \text{当} \mathbf{w}^{\mathrm{T}}\mathbf{x} > 0 \\ -1 & \text{当} \mathbf{w}^{\mathrm{T}}\mathbf{x} \leq 0 \end{cases},$$



感知器

- 学习算法
  - 一种错误驱动的在线学习算法：
  - 先初始化一个权重向量$\boldsymbol{w} \leftarrow 0$（通常是全零向量）；
  - 每次分错一个样本$(\boldsymbol{x}, y)$时，即

$$y\boldsymbol{w}^T\boldsymbol{x} < 0$$

  - 用这个样本来更新权重

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + y\boldsymbol{x}$$

  - 根据感知器的学习策略，可以反推出感知器的损失函数为

$$\mathcal{L}(\boldsymbol{w}; \boldsymbol{x}, y) = \max(0, -y\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})$$

感知器的学习过程

算法 3.1 两类感知器的参数学习算法

输入: 训练集 $\mathcal{D} = \{(\boldsymbol{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$，最大迭代次数 T

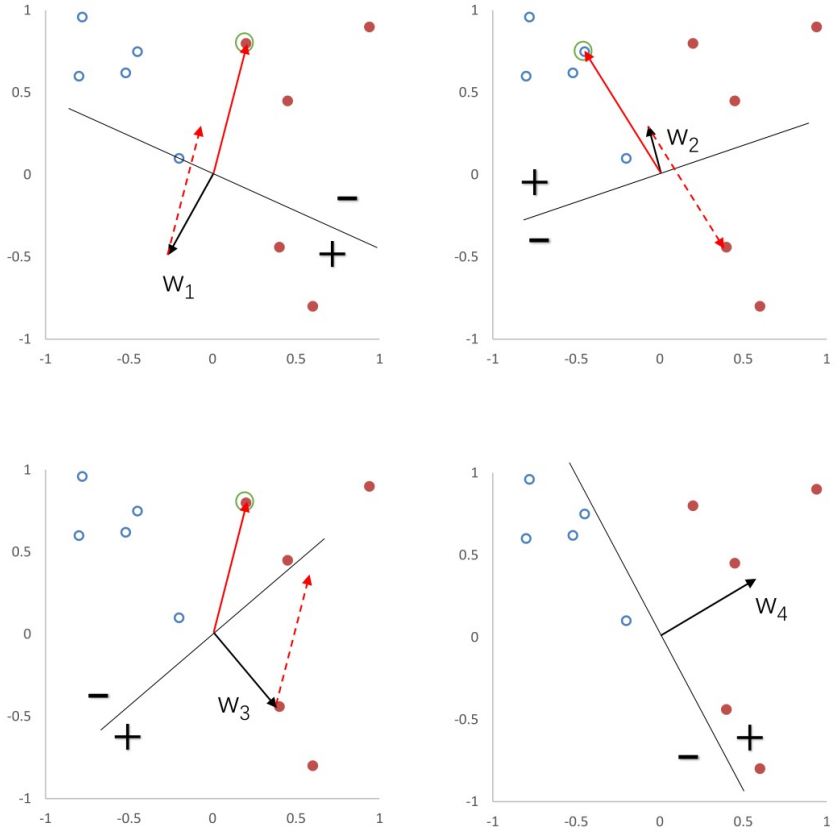
初始化：$\boldsymbol{w}_0 \leftarrow 0, k \leftarrow 0, t \leftarrow 0$；
repeat
  对训练集 $\mathcal{D}$ 中的样本随机排序；
  for $n = 1 \cdots N$ **do**
    选取一个样本 $(\boldsymbol{x}^{(n)}, y^{(n)})$；
    if $\boldsymbol{w}_k^{\mathrm{T}}(y^{(n)}\boldsymbol{x}^{(n)}) \leq 0$ **then**
      $\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k + y^{(n)}\boldsymbol{x}^{(n)}$；
      $k \leftarrow k + 1$；
    end
    $t \leftarrow t + 1$；
    if $t = T$ **then** break; // 达到最大迭代次数
  end
until $t = T$；
输出: $\boldsymbol{w}_k$

表示分错

对比Logistic回归的更新方式：

$$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + \alpha \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}^{(n)} \left(y^{(n)} - \hat{y}_{\boldsymbol{w}_t}^{(n)} \right)$$

感知器参数学习的更新过程



感知器的改进

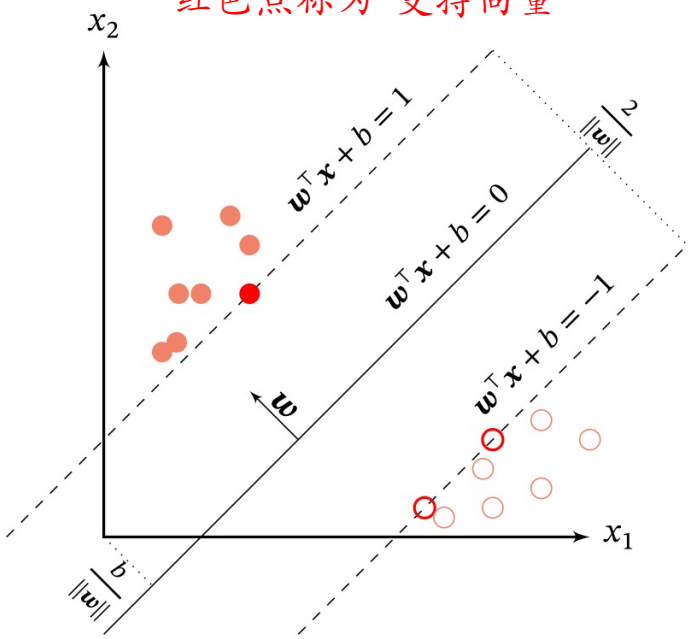
- 不足
  - 不能保证感知器的泛化能力；对样本的顺序比较敏感；如果训练集不是线性可分，永远不会收敛
- 改进
  - 参数平均感知器

$$\hat{y} = \text{sgn}\left(\frac{1}{T}\sum_{k=1}^{K} c_k(\boldsymbol{w}_k^{\top}\boldsymbol{x})\right)$$

- 不足
  - 只能适用于二分类问题
- 改进
  - 广义感知器

$$\hat{\boldsymbol{y}} = \underset{\boldsymbol{y}\in \text{Gen}(\boldsymbol{x})}{\arg\max}\, \boldsymbol{w}^{\top}\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}),$$

支持向量机（Support Vector Machine，SVM）



支持向量机的目标是寻找一个超平面 $(\boldsymbol{w}^*, b^*)$ 使得 γ 最大，即

$$\max_{\boldsymbol{w},b} \quad \gamma$$

$$\text{s.t.} \quad \frac{y^{(n)}(\boldsymbol{w}^\top \boldsymbol{x}^{(n)}+b)}{\|\boldsymbol{w}\|} \geq \gamma, \forall n \in \{1,\cdots,N\}$$

$$\max_{\boldsymbol{w},b} \quad \frac{1}{\|\boldsymbol{w}\|^2}$$

$$\text{s.t.} \quad y^{(n)}(\boldsymbol{w}^\top \boldsymbol{x}^{(n)}+b) \geq 1, \forall n \in \{1,\cdots,N\}$$

数据集 $\mathcal{D}$ 中每个样本 $\boldsymbol{x}^{(n)}$ 到分割超平面的距离为：

$$\gamma^{(n)} = \frac{|\boldsymbol{w}^\top \boldsymbol{x}^{(n)}+b|}{\|\boldsymbol{w}\|} = \frac{y^{(n)}(\boldsymbol{w}^\top \boldsymbol{x}^{(n)}+b)}{\|\boldsymbol{w}\|}.$$

支持向量机的改进

- 不足
  - 如果训练集不是线性可分，永远不会收敛
- 改进
  - 放松约束条件

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n$$

$$\text{s.t.} \quad 1 - y^{(n)}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}^{(n)} + b) - \xi_n \le 0, \qquad \forall n \in \{1, \cdots, N\}$$

$$\xi_n \ge 0, \qquad \forall n \in \{1, \cdots, N\}$$

  - 将原始特征空间映射到高维

$$f(\boldsymbol{x}) = \mathrm{sgn}(\boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x} + b^*) = \mathrm{sgn}\left(\sum_{n=1}^{N}\lambda_n^* y^{(n)}(\boldsymbol{x}^{(n)})^{\mathrm{T}}\boldsymbol{x} + b^*\right).$$

$$\Longrightarrow \quad f(\boldsymbol{x}) = \mathrm{sgn}(\boldsymbol{w}^{*\mathrm{T}}\phi(\boldsymbol{x}) + b^*) = \mathrm{sgn}\left(\sum_{n=1}^{N}\lambda_n^* y^{(n)} k(\boldsymbol{x}^{(n)}, \boldsymbol{x}) + b^*\right),$$

核函数

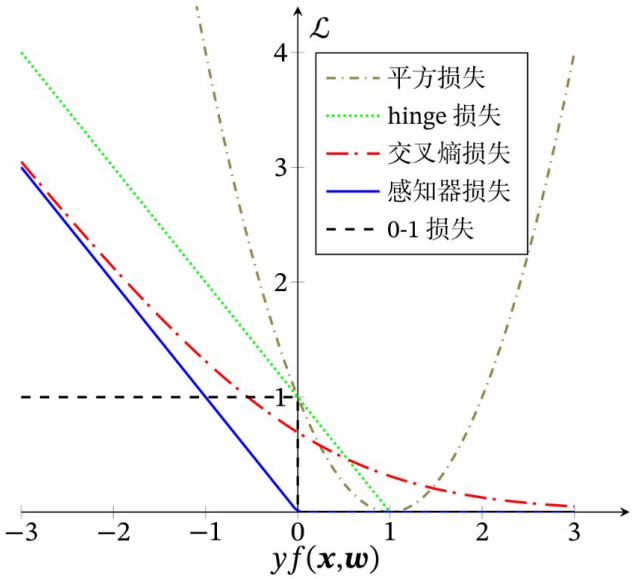
总结

线性分类模型小结

线性模型	激活函数	损失函数	优化方法
线性回归	-	$(y - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})^2$	最小二乘、梯度下降
Logistic 回归	$\sigma(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})$	$\boldsymbol{y} \log \sigma(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})$	梯度下降
Softmax 回归	$\mathrm{softmax}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{x})$	$\boldsymbol{y} \log \mathrm{softmax}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{x})$	梯度下降
感知器	$\mathrm{sgn}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})$	$\max(0, -y\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})$	随机梯度下降
支持向量机	$\mathrm{sgn}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})$	$\max(0, 1 - y\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x})$	二次规划、SMO 等

不同损失函数的对比

为了比较这些损失函数，我们统一定义类别标签 $y \in \{+1, -1\}$，并定义 $f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b$。这样对于样本 $(\mathbf{x}, y)$，若 $yf(\mathbf{x}; \mathbf{w}) > 0$，则分类正确，相反则分类错误。这样为了方便比较这些模型，我们可以将它们的损失函数都表述为定义在 $yf(\mathbf{x}; \mathbf{w})$ 上的函数。



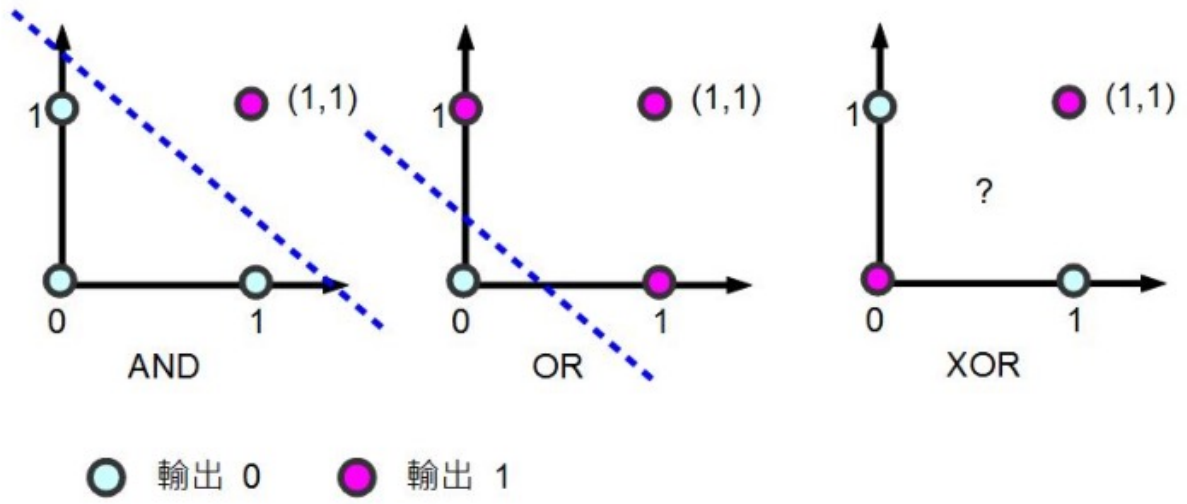
$$\mathcal{L}_{LR} = \log\left(1 + \exp\left(-yf(\mathbf{x}; \mathbf{w})\right)\right)$$

$$\mathcal{L}_{hinge} = \max\left(0, 1 - yf(\mathbf{x}; \mathbf{w})\right)$$

$$\mathcal{L}_{p} = \max\left(0, -yf(\mathbf{x}; \mathbf{w})\right)$$

$$\mathcal{L}_{squared} = \left(1 - yf(\mathbf{x}; \mathbf{w})\right)^2$$

XOR问题



特征

http://playground.tensorflow.org

如何处理非线性可分问题？