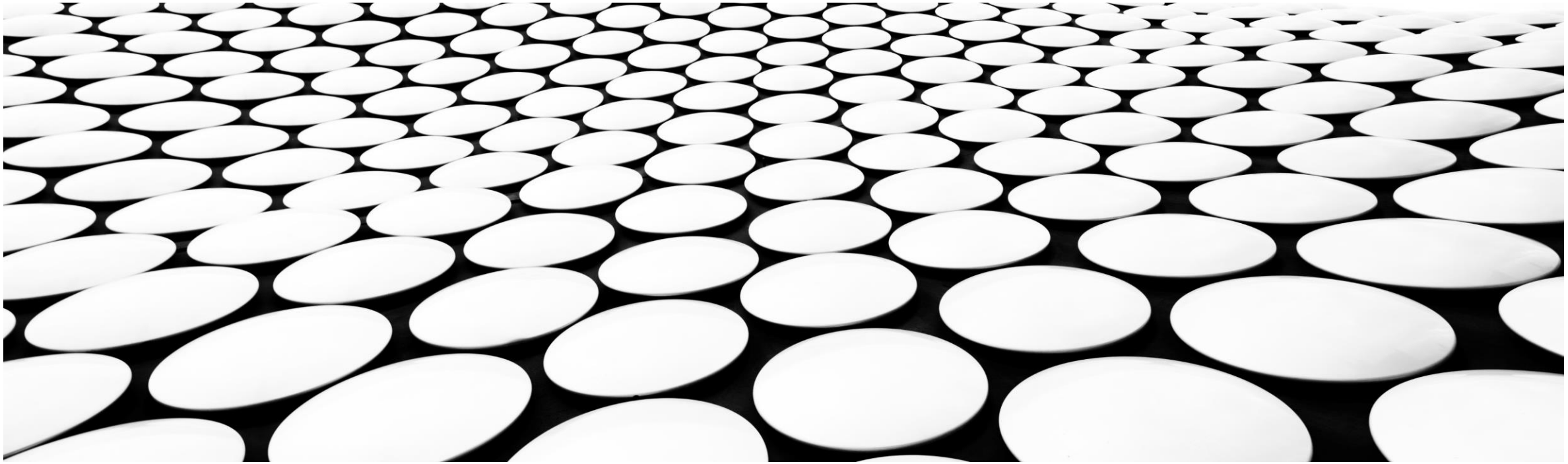
深度学习

邱怡轩



今天的主题

- 线性模型（续）
- PyTorch 基础

线性模型（续）

线性回归

- 给定模型 $Y = \beta' x + \varepsilon$
- 给定数据 $(x_i, Y_i),\ i = 1, \ldots, n$
- 估计 β

损失函数

- 常用最小二乘准则（即 MSE）
- $L(\beta) = n^{-1} \sum_{i=1}^{n} (Y_i - \beta' x_i)^2$
- 可以通过误差正态假设+极大似然准则导出

Logistic 回归

- 给定模型 $Y|x \sim Bernoulli(\rho(\beta' x))$
- $\rho(x) = e^x/(e^x + 1)$，即 Sigmoid 函数
- $\rho(\beta' x)$ 代表 Y 取1的概率

- 给定数据 $(x_i, Y_i),\ i = 1, \ldots, n$
- 估计 β

损失函数

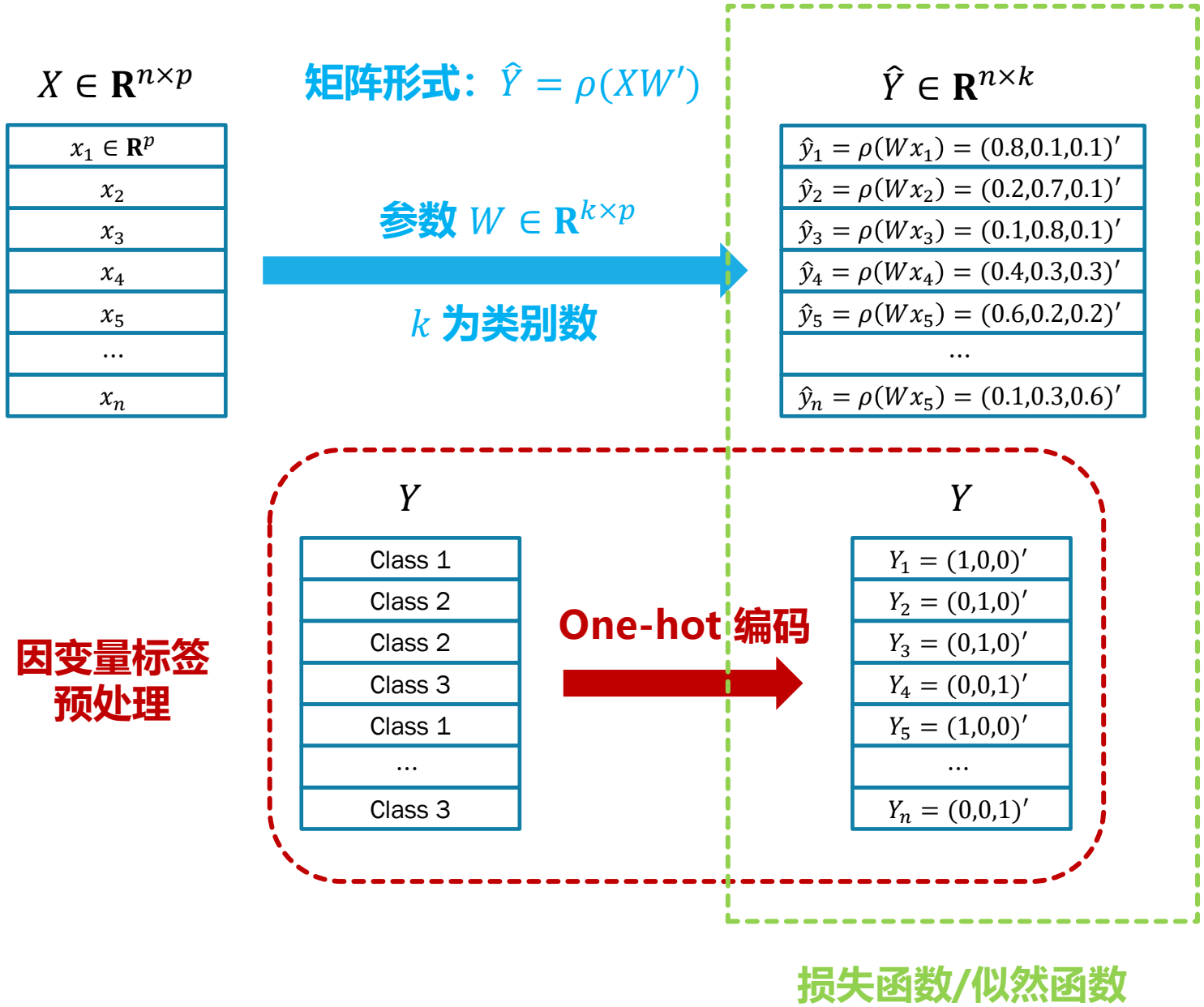
- 利用极大似然准则，Bernoulli 分布的对数密度函数为
  $l(p;y) = y\log p + (1-y)\log(1-p)$
- 其中 $p = \rho(\beta' x)$
- 似然函数 $L(\beta) = n^{-1}\sum_{i=1}^{n} l(\rho(\beta' x_i); Y_i)$

线性多分类

- 给定模型 $Y|x \sim Multinom(\rho(Wx))$
- $x \in \mathbf{R}^p,\ Y \in \{0,1\}^k,\ Wx \in \mathbf{R}^k$
- 每一个 Y 只有一个元素是1，其余为0
- $\rho(\cdot)$ 是把 Wx 映射到 $(0,1)^k$ 上的一个给定的函数，且得到的向量元素和为1
- $\rho(Wx)$ 代表 Y 取各类别的概率

- 数据 $(x_i, Y_i),\ i = 1, \ldots, n$
- 估计 W

线性多分类


$X \in \mathbf{R}^{n\times p}$
矩阵形式：$\hat{Y} = \rho(XW')$
$\hat{Y} \in \mathbf{R}^{n\times k}$
$x_1 \in \mathbf{R}^p$
x_2
x_3
x_4
x_5
…
x_n
参数 $W \in \mathbf{R}^{k\times p}$
k 为类别数
$\hat{y}_1 = \rho(Wx_1) = (0.8,0.1,0.1)'$
$\hat{y}_2 = \rho(Wx_2) = (0.2,0.7,0.1)'$
$\hat{y}_3 = \rho(Wx_3) = (0.1,0.8,0.1)'$
$\hat{y}_4 = \rho(Wx_4) = (0.4,0.3,0.3)'$
$\hat{y}_5 = \rho(Wx_5) = (0.6,0.2,0.2)'$
…
$\hat{y}_n = \rho(Wx_5) = (0.1,0.3,0.6)'$
因变量标签预处理
Y
Class 1
Class 2
Class 2
Class 3
Class 1
…
Class 3
One-hot 编码
Y
$Y_1 = (1,0,0)'$
$Y_2 = (0,1,0)'$
$Y_3 = (0,1,0)'$
$Y_4 = (0,0,1)'$
$Y_5 = (1,0,0)'$
…
$Y_n = (0,0,1)'$
损失函数/似然函数

Softmax 回归

- 一种常见的 ρ 的选择是 Softmax 函数

$$\text{softmax}(w_1, \ldots, w_k) = \frac{1}{Z}(e^{w_1}, \ldots, e^{w_k})$$

其中 $Z = \sum_{j=1}^{k} e^{w_j}$

- 此时的模型通常称为 Softmax 回归

损失函数

- 多项分布 $Multinom(p)$ 的对数密度函数为

$$l(p;y) = \sum_{j=1}^{k} y_j \log p_j$$

- 其中 $p = \rho(Wx) = (p_1, \ldots, p_k)$

- 似然函数 $L(\beta) = n^{-1} \sum_{i=1}^{n} l(\rho(Wx_i); Y_i)$

我们有办法

超越

极大似然准则吗?

KL 散度

- 我们先引入 KL 散度的概念
- 给定两个密度函数 p 和 q
- 定义 $\mathrm{KL}(q\|p) = \int q(z)\log\frac{q(z)}{p(z)}\mathrm{d}z$
- 还可以写成

$$\mathrm{KL}(q\|p) = \mathbf{E}_q \log q(Z) - \mathbf{E}_q \log p(Z)$$

KL 散度

- KL 散度的英文全称是
- Kullback–Leibler divergence

Solomon Kullback

Richard Leibler

KL 性质

- KL divergence 非负，$\mathrm{KL}(q\|p) \geq 0$
- 当 $q = p$ 时，$\mathrm{KL}(q\|p) = 0$
- KL 不是关于 p 和 q 对称的

- KL 可以用来衡量两个分布之间的区别

KL → MLE

- 现在让 $q = p^*$，表示数据的真实分布
- $p = p_\theta(x)$，表示模型的分布
- $\mathrm{KL}(p^* \| p_\theta) = \mathbf{E}_{p^*} \log p^*(x) - \mathbf{E}_{p^*} \log p_\theta(x)$

- 第一项与参数无关
- 第二项 $\mathbf{E}_{p^*} \log p_\theta(x) \approx n^{-1} \sum_{i=1}^{n} \log p_\theta(X_i)$
- 最小化 KL 等价于最大化似然函数！

启示

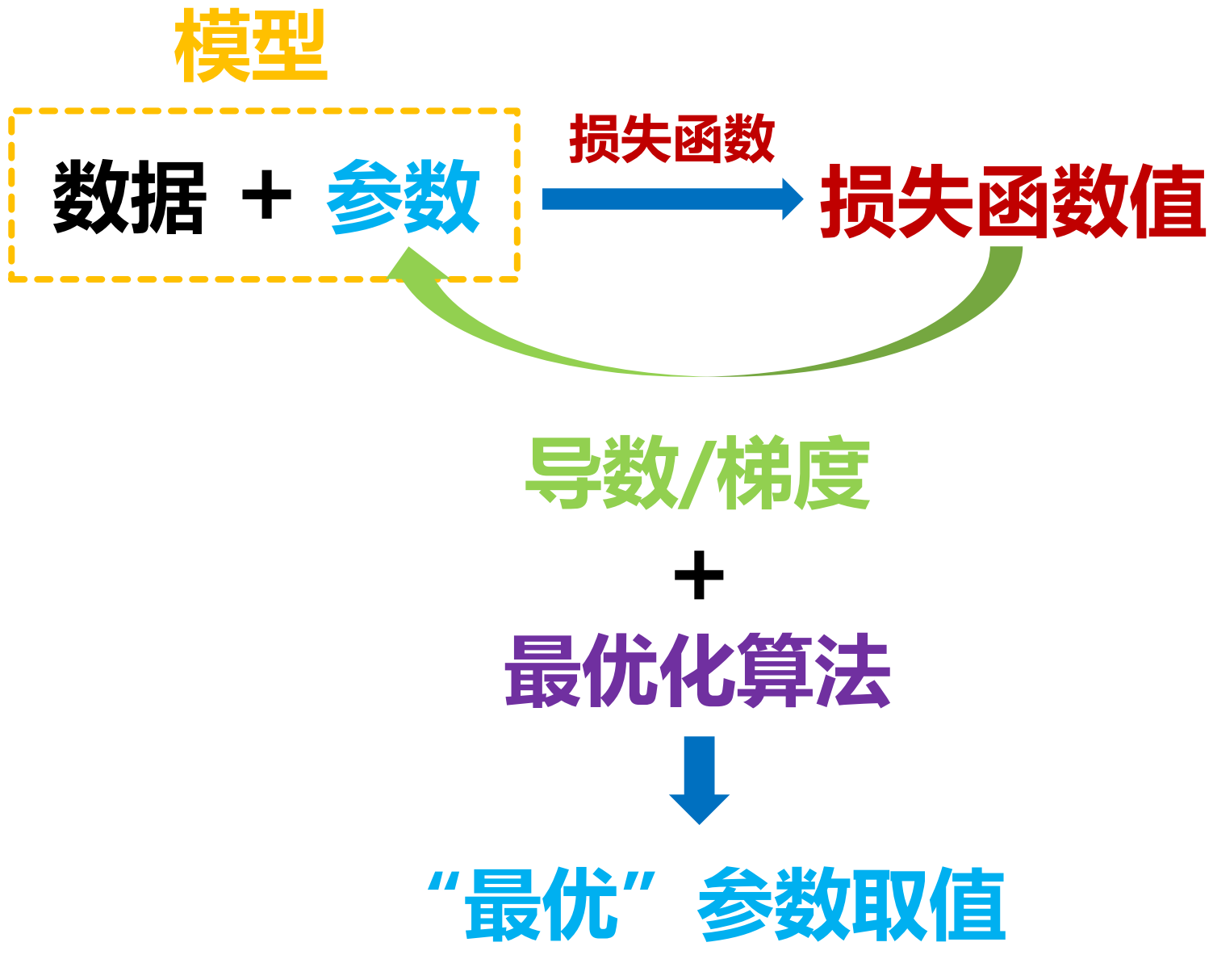
- 由于许多机器学习模型刻画的是数据的统计分布
- 所以对于这类问题，损失函数可以看作是两个分布之间的差别
- 除了 KL 散度之外，还有许多准则可以用来衡量分布之间的差异
- 启发了包括 GAN、WGAN 在内的众多流行的深度学习模型

小结

- 定义损失函数是许多机器学习模型的关键
- 极大似然准则是非常重要的方法
- 可以作为默认选项
- 还存在很多其他损失函数的定义方式
- 如 hinge loss，对应 SVM；check loss，对应分位数回归

PyTorch 基础

核心逻辑



PyTorch

- 参见 `lec3-pytorch.ipynb`

PyTorch 自动微分

- 参见 `lec3-autograd.ipynb`