

深度学习

输出层与代价函数

均方误差代价函数

公式

$$J = \frac{1}{2m} \sum_{i=1}^m ||y^{(i)} - h_{w,b}^{(i)}(x)||^2$$

适用范围

线性激活函数的输出。

优点

分类与回归问题均可用。

缺点

在某些情况下会出现残差衰减。

sigmoid输出层

公式

$$h = \frac{1}{1 + e^{-z}}$$

作用

作为输出层激活可用来完成二分类任务。

意义

将输入 z 变换为输出是正类的相对可能性

代价函数

通常配合交叉熵代价函数使用

sigmoid函数与几率的关系

如果我们用 h 表示正类($1-h$)表示负类, 则**几率(Odds)**的定义为 $h/(1-h)$ 。对几率取对数可以得到**对数几率(Logit odds)**为 $\ln(h/(1-h))$, 对数几率有很好的数学特性, 它是任意阶可导的凸函数。我们把sigmoid函数表达式带入对数几率式子可得:

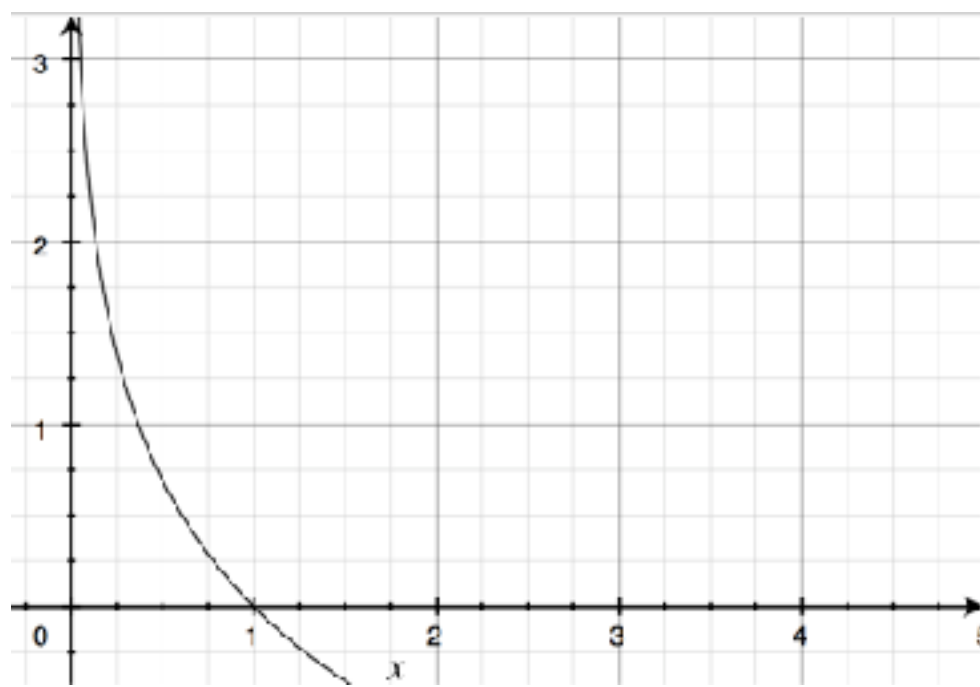
$$\ln \frac{h}{1-h} = z$$

可以得到: 训练完成的神经网络, 其输出层的输入表征一个几率, sigmoid函数相当于将几率转化为了结果为正类的概率。

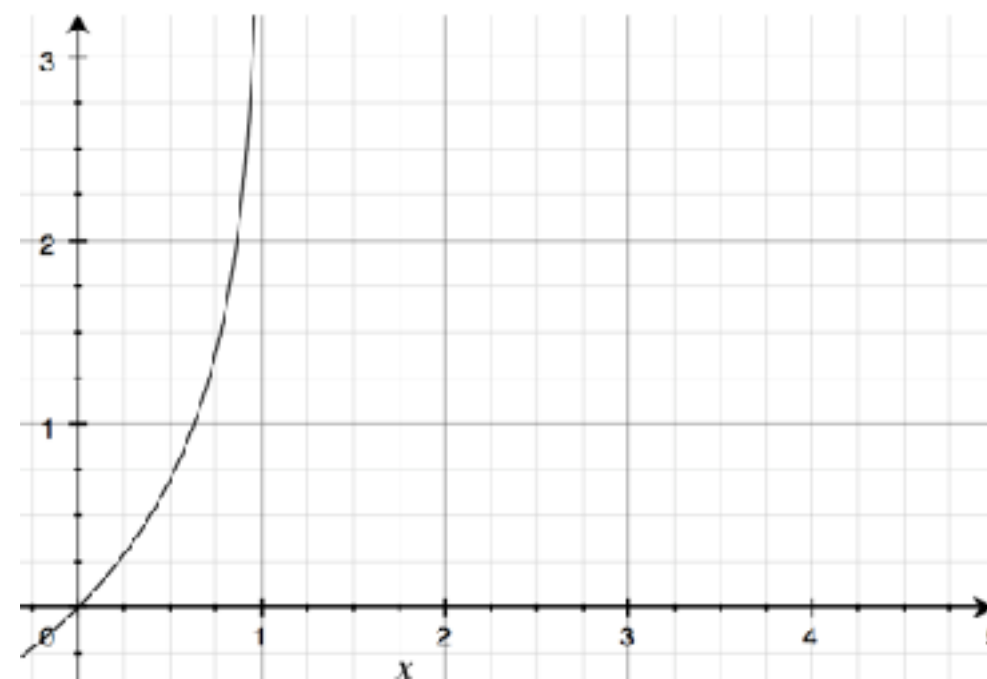
交叉熵代价函数

$$J = -\frac{1}{m} \sum_{i=1}^m [y_i \ln h(x_i) + (1 - y_i) \ln(1 - h(x_i))]$$

交叉熵可以用来比较两个概率的相似性，在机器学习中，常用来衡量预测值与标记值的误差。误差越大，代价越大。



当 $y=1$ 时， J 的图像。



当 $y=0$ ， J 的图像。

利用极大似然估计得到代价函数

若将标记 y 视为类后验概率估计, 则有:

$$p(y = 1|x) = h$$

$$p(y = 0|x) = 1 - h$$

对给定数据集对模型取对数似然(Log-likelihood)可得:

$$l = \sum_{i=1}^m \ln p(y_i | x_i; w, b)$$

根据类后验概率估计的式子即可以得到:

$$p(y_i | x_i; w, b) = y_i p(y = 1|x_i) + (1 - y_i) p(y = 0|x_i)$$

将上式带入对数似然函数, 并根据类后验概率估计可得:

$$l = \sum_{i=1}^m (y_i h(x_i) + (1 - y_i)(1 - h(x_i)))$$

我们的目标是最大化平均对数似然, 也就是最小化负的平均对数似然, 即:

$$\min(-\hat{l}) = -\frac{1}{m} \sum_{i=1}^m [y_i \ln h(x_i) + (1 - y_i) \ln(1 - h(x_i))] = J$$

交叉熵代价函数的梯度下降

我们以单样本为例，代价函数为：

$$J = -y \ln h(x) + (y - 1) \ln (1 - h(x))$$

为了方便推导，下面我们需要定义变量。

l 表示层索引， L 表示最后一层的索引。 j 、 k 均表示某一层神经元的索引。

N 表示神经网络每一层的神经元数量。 w 表示连接权重， b 表示偏置值。

$w_{jk}^{(l)}$ 表示第 l 层的第 j 个神经元与第 $(l - 1)$ 层的第 k 个神经元的连接权重。

$b_j^{(l)}$ 表示第 l 层的第 j 个神经元的偏置值。

$N^{(l)}$ 表示第 l 层神经元的数量。其中 $N^{(L)}$ 表示最后一层神经元的数量。

$z_j^{(l)} = \sum_{k=1}^{N^{(l-1)}} w_{jk}^{(l)} a_k^{(l-1)} + b_j^{(l)}$ 表示第 l 层第 j 个神经元的输入。

$a_j^l = \sigma(z_j^{(l)})$ 表示第 l 层第 j 个神经元的输出。其中 σ 表示 $sigmoid$ 激活函数。

输出层的残差表示为：

$$\delta^{(L)} = \frac{\partial J}{\partial z^{(L)}} = \frac{\partial J}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma(z^{(L)}) - y$$

输出层的连接权重的梯度：

$$\begin{aligned} \frac{\partial J}{\partial w_{jk}^{(L)}} &= \frac{\partial J}{\partial z_j^{(L)}} \frac{\partial z_j^{(L)}}{\partial w_{jk}^{(L)}} \\ &= (\sigma(z_j^{(L)}) - y) \cdot \frac{\partial}{\partial w_{jk}^{(L)}} \left(\sum_{K=1}^{N^{L-1}} (w_{jK}^L a_K^{(L-1)} + b_j^{(L)}) \right) \\ &= (\sigma(z_j^{(L)}) - y) a_k^{(L-1)} \end{aligned}$$

同理，输出层偏置的梯度：

$$\frac{\partial J}{\partial b_j^{(L)}} = \sigma(z_j^{(L)}) - y$$

softmax输出层

公式

$$h_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

作用

二分类或者多分类任务的输出层激活函数。

意义

将输入排序，并转换为概率表示。

代价函数

通常配合对数似然代价函数来表示代价。

对数似然代价函数

$$J = -\frac{1}{m} \sum_{i=1}^m \sum_{j=1}^{N^{(L)}} y_j^{(i)} \ln h_j^{(i)}(x)$$

对数似然函数的梯度下降

我们首先将代价函数转化为单样本的代价函数：

$$J = -\sum_{j=1}^{N^{(L)}} y_j \ln h_j(x)$$

输出层的残差表示为：

$$\delta_j^{(L)} = \frac{\partial J}{\partial z_j^{(L)}} = \frac{\partial J}{\partial a_j^{(L)}} \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} = -y_j \cdot \frac{1}{a_j^{(L)}} \cdot \frac{\partial J}{\partial z_j^{(L)}}$$

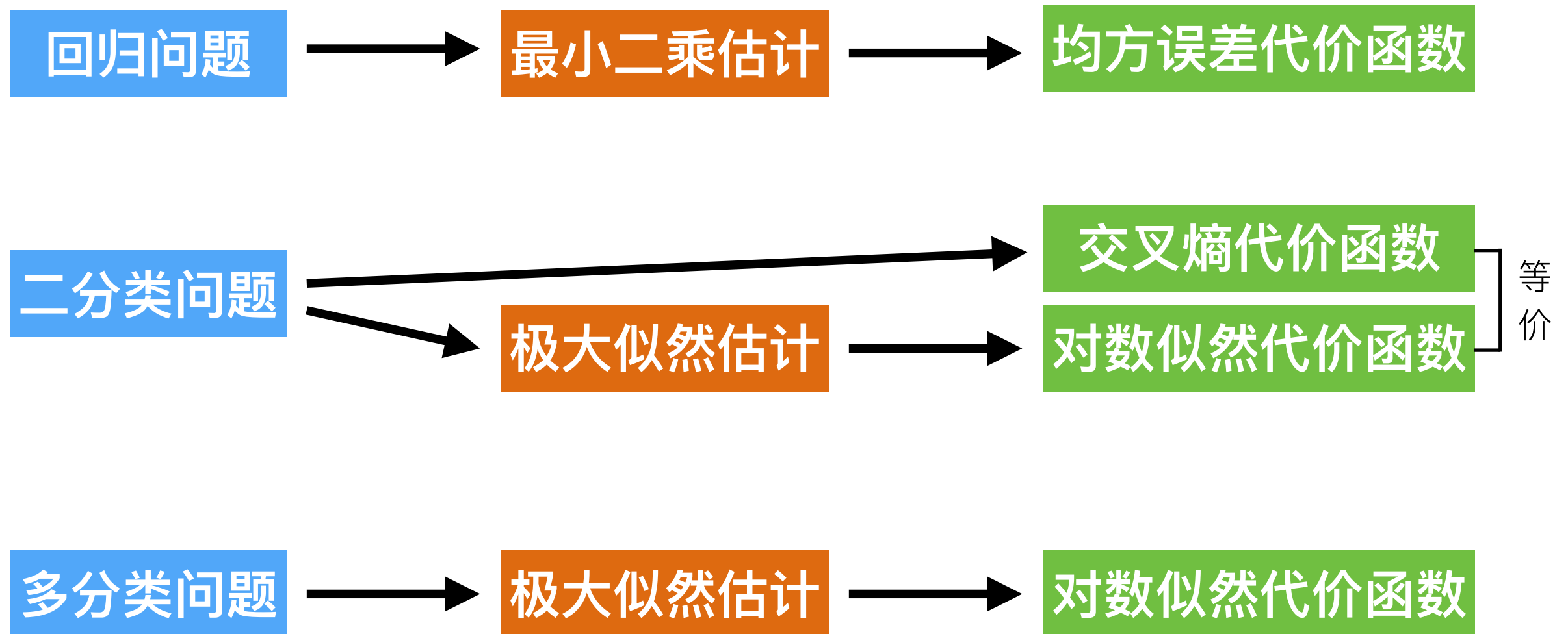
输出层连接权重的梯度为：

$$\frac{\partial J}{\partial w_{jk}^{(L)}} = \frac{\partial J}{\partial z_j^{(L)}} \frac{\partial z_j^{(L)}}{\partial w_{jk}^{(L)}} = \delta_j^{(L)} \cdot \frac{\partial}{\partial w_{jk}^{(L)}} \sum_{K=1}^{N^{L-1}} (w_{jK}^{(L)} a_K^{(L-1)} + b_j^{(L)}) = \delta_j^{(L)} \cdot a_k^{(L-1)}$$

同理可得输出偏置的梯度：

$$\frac{\partial J}{\partial b_j^{(L)}} = \delta_j^{(L)}$$

代价函数适用范围



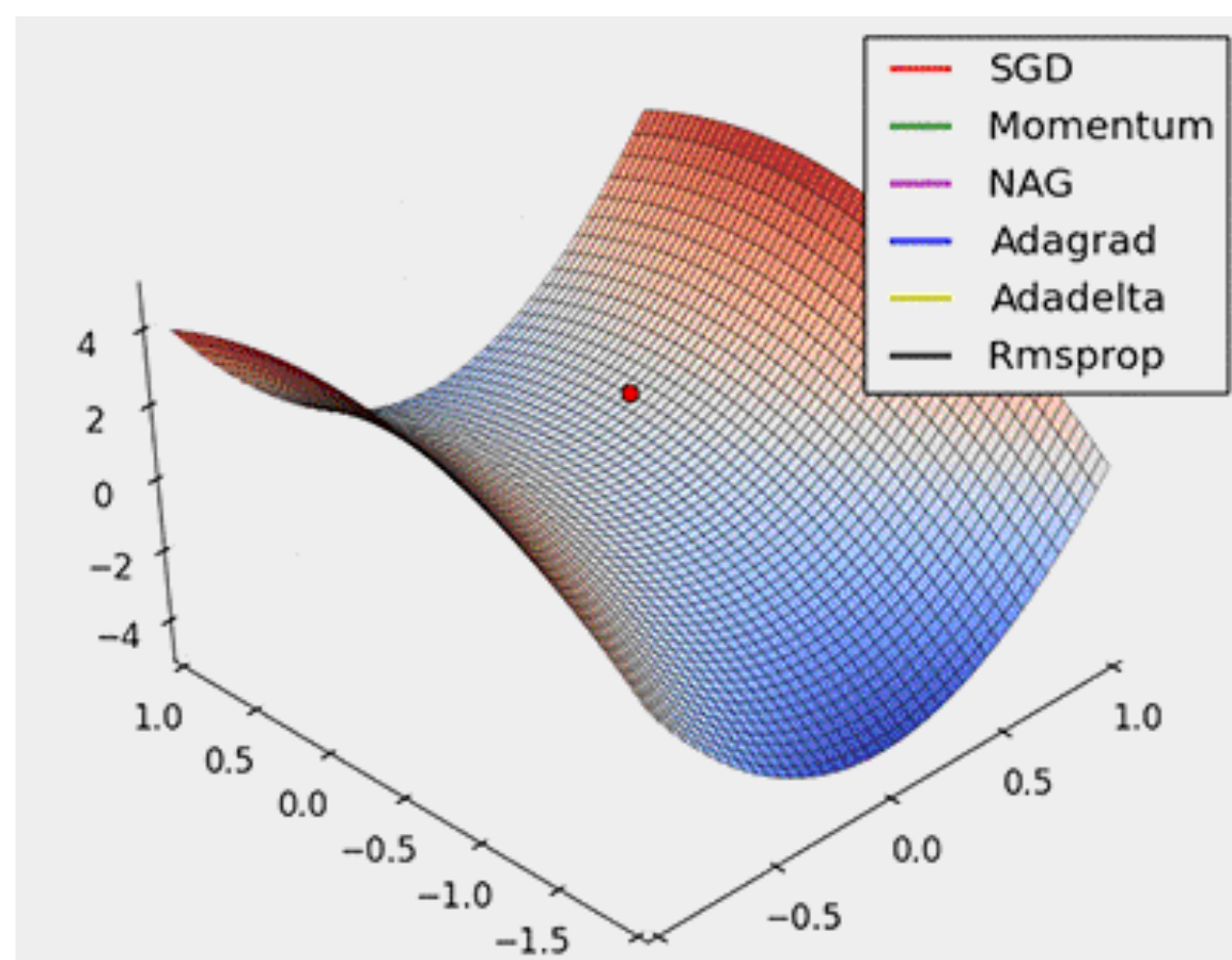
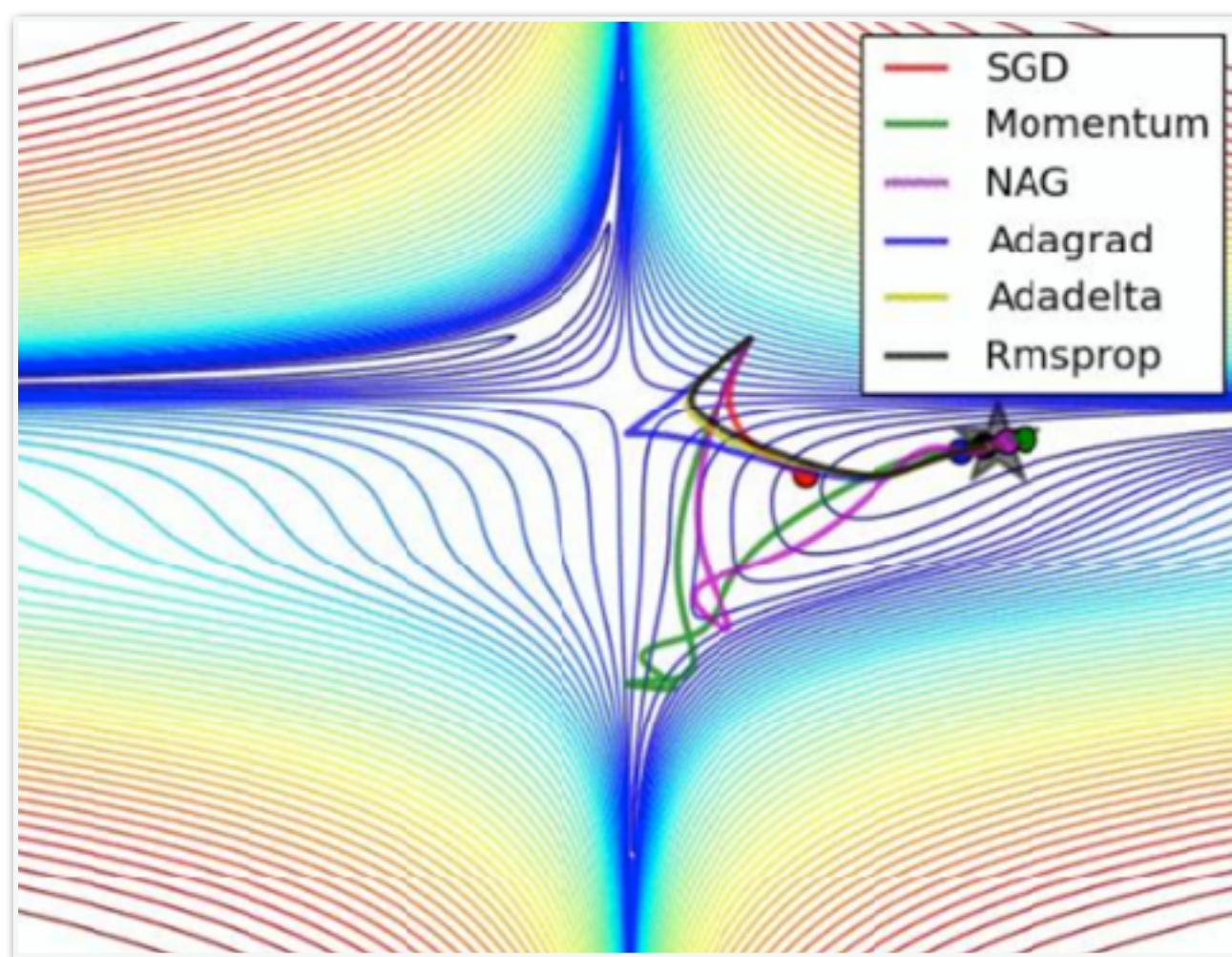
ANN的训练步骤

mini-batch SGD

- **LOOP**

1. 在数据集中取batch-size大小的样本
2. 在网络中前向传播，得带loss
3. 反向计算梯度
4. 使用梯度更新参数

了解多种优化器



小节

- 均方误差代价函数可以作为回归任务的代价函数，例如输出层使用线性激活函数。其用于分类任务可用，但不太合适。
- **sigmoid**函数在输出层的作用是将几率转化为概率。
- **softmax**函数在输出层的作用是将多个输出归一化，并转化为概率。输入越大，概率越大。
- 当遇到二分类任务时，可以选择**sigmoid**函数或者**softmax**函数作为输出层的激活函数。
- 当解决多分类任务时，选择**softmax**函数作为输出层激活函数。
- 输出层使用**sigmoid**激活函数时，输出层有且仅有一个神经元。其最适合的代价函数是交叉熵代价函数。
- 输出层使用**softmax**激活函数时，最适合的代价函数是对数似然代价函数。
- 交叉熵代价函数与对数似然代价函数形式上一致。但来源不同。

THANKS