paddle学习笔记之损失函数

损失函数的效果要结合模型目的来看

后续可能会有补充

此处先提回归和分类算法

回归和分类是机器学习中两种基本的预测方法，它们的本质区别在于输出变量的类型。回归问题输出的是连续的数值，分类问题输出的是有限的、离散的类别标签。两者都是监督学习的一部分，都依赖于带有标签的训练数据来学习模型。

回归的本质是对自变量因变量直接的关系进行模拟

回归的目的是预测数值型的目标值，本质是寻找自变量和因变量之间的关系，以便能够预测新的、未知的数据点的输出值

而对于分类是将输入数据分到对应的类别中

分类的目的是预测标签型的目标值，本质是根据输入数据的特征将其划分到预定义的类别中

（1）回归（Regression）的算法

主要用于预测数值型数据。

线性回归（Linear Regression）：这是最基本和常见的回归算法，它假设因变量和自变量之间存在线性关系，并通过最小化预测值和实际值之间的平方差来拟合数据。
多项式回归（Polynomial Regression）：当自变量和因变量之间的关系是线性的，可以使用多项式回归。它通过引入自变量的高次项来拟合数据，从而捕捉非线性关系。
决策树回归（Decision Tree Regression）：决策树回归是一种基于树结构的回归方法，它通过构建决策树来划分数据空间，并在每个叶节点上拟合一个简单的模型（如常数或线性模型）。决策树回归易于理解和解释，能够处理非线性关系，并且对特征选择不敏感。
随机森林回归（Random Forest Regression）：随机森林回归是一种集成学习方法，它通过构建多个决策树并将它们的预测结果组合起来来提高回归性能。随机森林回归能够处理高维数据和非线性关系，并且对噪声和异常值具有一定的鲁棒性。

（2）分类（Classification）的算法

主要用于发现类别规则并预测新数据的类别。

逻辑回归（Logistic Regression）：尽管名字中有“回归”，但实际上逻辑回归是一种分类算法，常用于二分类问题。它通过逻辑函数将线性回归的输出映射到(0,1)之间，得到样本点属于某一类别的概率。在回归问题中，有时也使用逻辑回归来处理因变量是二元的情况，此时可以将问题看作是对概率的回归。
支持向量机（SVM）：支持向量机是一种基于统计学习理论的分类算法。它通过寻找一个超平面来最大化不同类别之间的间隔，从而实现分类。SVM在高维空间和有限样本情况下表现出色，并且对于非线性问题也可以使用核函数进行扩展。
K最近邻（KNN）：K最近邻是一种基于实例的学习算法，它根据输入样本的K个最近邻样本的类别来确定输入样本的类别。KNN算法简单且无需训练阶段，但在处理大规模数据集时可能效率较低。
朴素贝叶斯分类器：朴素贝叶斯是一种基于贝叶斯定理的分类算法，它假设特征之间相互独立（即朴素假设）。尽管这个假设在实际应用中往往不成立，但朴素贝叶斯分类器在许多领域仍然表现出色，尤其是在文本分类和垃圾邮件过滤等方面。

回归问题的损失函数

平方损失

Huber损失

绝对值

其中最常用的是平方损失，然而其缺点是对于异常点会施以较大的惩罚，因而不够robust。如果有较多异常点，则绝对值损失表现较好，但绝对值损失的缺点是在y−f(x)=0处不连续可导，因而不容易优化。Huber损失是对二者的综合，当|y−f(x)|小于一个事先指定的值δ时，变为平方损失，大于δ时，则变成类似于绝对值损失，因此也是比较robust的损失函数。

分类问题的损失函数

0-1损失函数

logistic loss

hinge loss

指数损失

机器学习的损失函数

交叉熵损失函数

在用梯度下降法做参数更新的时候，模型学习的速度取决于两个值：一、**学习率**；二、**偏导值**。其中，学习率是我们需要设置的超参数，所以我们重点关注偏导值。从上面的式子中，我们发现，偏导值的大小取决于 x_i 和 $[\sigma(s)-y]$ ，我们重点关注后者，后者的大小值反映了我们模型的错误程度，该值越大，说明模型效果越差，但是该值越大同时也会使得偏导值越大，从而模型学习速度更快。所以，使用逻辑函数得到概率，并结合交叉熵当损失函数时，在模型效果差的时候学习速度比较快，在模型效果好的时候学习速度变慢。

优点

均方误差

分类问题中，使用sigmoid/softmx得到概率，配合MSE损失函数时，采用梯度下降法进行学习时，会出现模型一开始训练时，学习速率非常慢的情况（MSE损失函数）。

不足

分类常用Softmax函数

交叉熵函数

交叉熵损失函数的设计是基于最大似然思想：最大概率得到观察结果的假设是真的。

交叉熵的公式如下：

$$Loss = -[\sum_{k=1}^{N} t_k \log y_k + (1-t_k)\log(1-y_k)]$$

其中，loglog表示以*ee*为底数的自然对数。yk*y***k*代表模型输出，tk*t***k*代表各个标签。tk*t***k*中只有正确解的标签为1，其余均为0（one-hot表示）。

因此，交叉熵只计算对应着“正确解”标签的输出的自然对数。比如，假设正确标签的索引是“2”，与之对应的神经网络的输出是0.6，则交叉熵误差是−log0.6=0.51−log0.6=0.51；若“2”对应的输出是0.1，则交叉熵误差为−log0.1=2.30−log0.1=2.30。由此可见，交叉熵误差的值是由正确标签所对应的输出结果决定的。

自然对数的图形所示，当*xx*等于1时，*yy*为0；随着*xx*向0靠近，*yy*逐渐变小。因此，正确解标签对应的输出越大，交叉熵的值越接近0；当输出为1时，交叉熵误差为0。反之，如果正确解标签对应的输出越小，则交叉熵的值越大。