requires\_grad=True 以跟踪梯度

another\_tensor = torch.tensor([1.0])：创建另一个一维张量 [1.0]，默认 requires\_grad=False（不跟踪梯度）。

由于 tensor\_requires\_grad 需要梯度，PyTorch 会构建计算图以跟踪此操作。

result.backward() 计算 result 关于所有 requires\_grad=True 的输入张量的梯度。 对于函数 \(y = x + c\)（其中 c 是常数），导数 \(\frac{dy}{dx} = 1\)。 因此，tensor\_requires\_grad 的梯度为 tensor([1.])。

深度学习中的损失函数（Loss Function）是一个衡量预测结果与真实结果之间差异的函数 ，也称为误差函数。它通过计算模型的预测值与真实值之间的不一致程度，来评估模型的性能。

损失函数（Loss Function）是什么？损失函数是深度学习中用于衡量模型预测结果与真实结果之间差异的函数。

损失函数通过计算一个数值，来表示模型预测的准确性或误差大小。

为什么需要损失函数？在训练过程中，模型的目标是通过调整其参数来最小化损失函数的值，从而提高预测的准确性。

损失函数能量化模型预测与真实结果之间的差异。

损失函数（Loss Function）的值直接反映了模型预测结果与真实标签之间的差异程度。具体关系如下：

1. 核心关系

损失值越大 → 模型预测与真实结果的差异越大 → 模型性能越差。

损失值越小 → 模型预测与真实结果越接近 → 模型性能越好。

异常情况说明

损失值突然增大：可能由于学习率过大、数据中存在噪声或模型过拟合。

损失值长期不变：可能是学习率过小或模型陷入局部最优。

选择合适的损失函数（如 MSE、交叉熵）需根据具体任务类型（回归或分类）决定。

优化器在训练过程中更新神经网络的参数，以减少损失函数的值。

PyTorch 提供了多种优化器，例如 SGD、Adam 等。

训练模型

训练模型是机器学习和深度学习中的核心过程，旨在通过大量数据学习模型参数，以便模型能够对新的、未见过的数据做出准确的预测。

训练模型通常包括以下几个步骤：

数据准备：

收集和处理数据，包括清洗、标准化和归一化。

将数据分为训练集、验证集和测试集。

定义模型：

选择模型架构，例如决策树、神经网络等。

初始化模型参数（权重和偏置）。

选择损失函数：

根据任务类型（如分类、回归）选择合适的损失函数。

选择优化器：

选择一个优化算法，如SGD、Adam等，来更新模型参数。

前向传播：

在每次迭代中，将输入数据通过模型传递，计算预测输出。

计算损失：

使用损失函数评估预测输出与真实标签之间的差异。

反向传播：

利用自动求导计算损失相对于模型参数的梯度。

参数更新：

根据计算出的梯度和优化器的策略更新模型参数。

迭代优化：

重复步骤5-8，直到模型在验证集上的性能不再提升或达到预定的迭代次数。

评估和测试：

使用测试集评估模型的最终性能，确保模型没有过拟合。

模型调优：

根据模型在测试集上的表现进行调参，如改变学习率、增加正则化等。

部署模型：

Dataset/DataLoader：PyTorch 数据加载工具

os：文件路径操作