# 学习笔记

## sklearn库的简单了解

做一下学习笔记。

网站为：[scikit-learn: machine learning in Python — scikit-learn 1.4.0 documentation](https://scikit-learn.org/stable/)。

从getting start部分进入，有scikit-learn的大致介绍。以下是一些简单的笔记总结。

sklearn是一个开源的机器学习库，支持监督学习和无监督学习，提供模型拟合、数据预处理、模型选择与评估的一些有用的工具。

### 内置的算法和模型

简单来说就是很多现成的estimators，每种数据都有自己适合的模型。

一般来说有两个输入：

1. 一个矩阵，行为样本数量，列由不同特征值组成
2. 真实输出y（无监督学习中不需要），通常为一维数组

### 对数据进行预处理

可以对数据进行预处理，对不同的特征施加变换

### 链接预处理器和estimator

Transfromer和estimator结合为一个pipeline，简而言之是把数据的预处理和预测器打包到一起，然后允许调用一些库，分割训练集和测试集。

**模型评估**

Sklearn提供模型评估的工具，尤其对于cross-validation这方面的评估。

#### 什么是交叉验证

一般的划分方法是简单地分为测试集和训练集，但是最终模型与参数的选取极大程度依赖划分方法。一言以蔽之，很难得到最好的模型与参数。

因此提出交叉验证。（这里姑且是学会了怎么用word打公式，后面会抽时间去学习latex）

1. LOOCV方法

N为数据集的数据数量，只选取一个数据作为测试集，n-1个数据用于训练，重复n次。为公式，计算量比较大。

优化后的

1. K-fold Cross Validation

LOOCV其实是一种特殊的k折验证方法，将数据分为k份，挑其中的一份作为测试集，重复k次。

公式是类似的，计算量更小。一般来说K太大不是好事，通常选择5或10。

1. Classification Problem

以上都是回归问题，这里是分类问题。公式如下。

### 自动参数搜索

自动寻找最佳参数组合，一般不会用自己的训练数据用来预处理，否则会降低模型的泛化能力，sklearn就提供了pipeline，而不需要我们单独调用estimator。

#### 超参数是什么

事先给定的参数，其它参数通过训练得出。分为**模型超参数和算法超参数。前者用于模型选择，后者影响学习的速度与质量。比如学习率和迭代次数就是一种超参数。**

## ****logistics函数相关****

这里我选择了吴恩达老师的课程。

### logistic回归

表示对y的预测，为一个概率。为一个线性回归的输出。

但我们希望输出是一个概率，使输出转化为的函数。

此时会套上sigmoid：且

这时候就把输出规定在（0，1）了。Z越大，输出越接近1。

### 损失函数，成本函数

一般的计算loss的方法（求平方）通常非凸，会得到多个局部最优解。

### 梯度下降

找到一个初始点，从初始点处向下下降，收敛到全局最优解

重复更新操作：, 表示学习率

不断更新参数，直到收敛到极小值。对于多个参数会同时进行这个操作，用于更新多个参数。在这里我们需要用到导数和偏导数的知识。

### 计算图

把复杂函数的计算过程转化为有序步骤的流程图，前向传播和反向传播的直观体现。本质上是链式法则的运用。

#### Logistic 回归中的梯度下降法

首先初始化参数。正向传播得到损失函数，再对损失函数求导，通过链式法则得到所需参数的偏导数。

对于多个样本，利用循环。第一个循环用于遍历所有样本，另一个循环用于遍历所有特征。但是此种方法对于大数据集较为低效。

这时候使用向量化高效化计算。

### 向量化

使用, 比for循环更快

本质上是充分利用CPU或GPU的并行化，更快地计算

#### 关于np.dot的查阅

它不会自动进行矩阵转置

1. 用于矩阵时，按矩阵乘法处理传入的两个二维数组
2. 用于两个一维数组时，计算点积，简而言之是计算出一个数
3. 传入的一维数组通常被视为列向量

使用numpy库的各种方法可以减少一些explicit for-loop，使用for loop前先看是否能用numpy。利用向量化同样可以解决logistic回归的问题。

#### 值得注意的点

我自己手算sigmoid导数的时候没有注意到可以写成一种很简便的形式。同时查阅了一下交叉熵的求导方法，很疑惑一个是怎么求导的，查阅资料得知一般看成求导。

为什么能直接看成ln求导呢？

选用其它常数作为底数的话，会导致我们计算微分进行反向传播时，出现额外的常数项，将计算复杂化，实际上这些常数的存在可能并不会对预测的准确性产生较大影响，但是将e作为底数时，对神经网络框架的兼容性应该是最简洁、较优秀的。

利用链式法则很容易得到的求偏导结果，获得线性回归的某个权重的参数。

#### Python中的广播

Sum函数中设置axis参数为0，对行操作使得矩阵的列相加。Axis为1时，对列操作水平求和。

Reshape函数中（a，b），a表示rows，b表示columns。

Python中的广播机制用于矩阵之间或者矩阵与数之间的运算操作，把原有的数据copy一下，使得两个矩阵能进行运算。

由于广播可能引起一些难以发现的bug，建议在定义或创建矩阵时明确其行列数。

需要的时候，使用assert检查，reshape修改，保证矩阵和向量是需要的维度。

#### 补充

在外面套个log

具有多个样本时，就会加上求和符号了。

最大似然估计：基于已知样本数据，寻找一组参数值，使得这些数据出现的概率最大。基于这个得到了成本函数。

MSE主要用来度量模型的预测值和真实值之间的差异程度。

查阅得知，R2分数通过计算模型预测值和真实值的差异与总方差之间的比例来确定的，R2一般不是一个绝对的评估指标，一般用于比较不同模型之间的相对性能。R2对线性回归模型直接适用，但是对于非线性模型，可能解释性会有所不同。

*SSR为模型预测值和实际观测值的差异的平方和，SST为实际观测值与观测值的平均值之间的差异的平方和，R2越接近1，拟合程度越好。*