【猫猫的机器学习-1】从初中数学开始的Linear Regression（完整推导版）

文章封面的抹布来自一个令我印象深刻的数学课发生的事，正好是讲函数的，写文章时莫名联想到了。不过这是玩梗罢了，它并没有真扔到我脸上（只是桌上）大家一笑而过就好。本来想画棕黄色的抹布，但是实在太嗯了就改蓝色了XD

引言

等中高考结束后我也差不多期末考了，这段时间算是思绪很乱，学习效率很低下，在迫不得已的焦虑下整了本 *Introduction to machine learning*《机器学习导论》，对这玩意有点兴趣和冲动后就去看了很多有关的东西，我觉得我还是更适合从实际出发吧，所以就决定开这个系列。
正如标题所写，这个新系列，我希望做的是在保证质量的前提下尽可能让内容更好理解。我还是一如既往的那个不擅长理科的Makiror，自己也有些恶补缺失基础和一步步学习的经验，所以你可以比较不用担心我的Blog会出现令人费解的高深内容，或是没必要的，故弄玄虚的花里胡哨。

> 注意：这可能会是一个**有时效性**的系列，人工智能这个领域的变化实在是太快了，我没法保证它的内容一直不过时。

文章没有假设读者门槛，如果有什么看不懂的地方顺道补上就好了，第一篇不会难的。不过不过，如果你有一点点的数学基础就更好了。
本文更偏向理论，文章内示例使用Haskell编程语言编写，不熟悉语法的看看文档多写几下就好了，这么一来，你还能顺便多用一个函数式编程语言，这算是两个愿望一次实现啦（

基础概念

因为是第一篇，考虑到不知道机器学习的读者，在正文开始之前我们先来了解一些机器学习有关的基础概念。

什么是机器学习？

机器学习（Machine Learning）是一门关注如何让计算机通过数据学习和自动改进的领域，它已经在各个行业和领域产生了巨大的影响。随着计算能力的提升和大数据的兴起，机器学习技术变得越来越重要，被广泛应用于自动驾驶、语音识别、推荐系统、金融预测等众多领域。
在传统的计算机程序中，我们需要明确地表示规则和逻辑来完成特定任务，例如图像识别、语音识别或自然语言处理。但是在面对复杂的、高度变化的现实世界问题时，这样使得我们的任务变得非常困难甚至不可行，而机器学习就是让计算机从大量的数据中进行学习，并通过统计学方法和一些算法来自动发现数据中的模式和规律。它的目标是构建能够从数据中进行学习、泛化并做出预测或决策的模型，这些模型可以根据已有的数据进行训练和优化，然后利用学到的知识来对新的、未见过的数据进行预测和决策。

学习算法

学习算法是机器学习的核心部分，根据输入数据和所需的任务，自动调整模型的参数或规则，使得模型能够从数据中进行学习和预测。

监督学习是机器学习中的一种算法类型，本篇的『线性回归』就是一种监督学习算法。它是一种通过使用带有标签的训练数据来构建预测模型的算法，模型从已知输入和相应的输出之间的关系中学习，然后根据这个关系对新的输入进行预测。与之对应的无监督模型的数据是没有标签的，算法从数据中发现类别并学习。

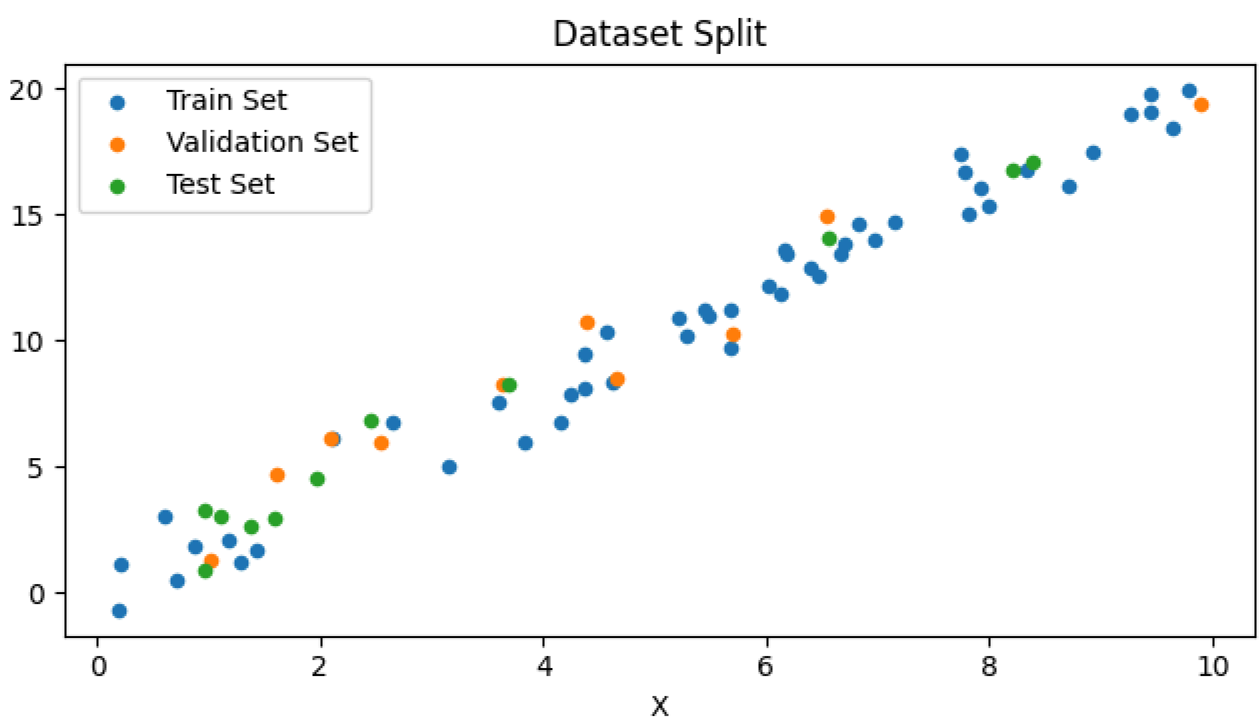
训练集、测试集和验证集

前面提到的“数据”什么的，虽然都这么说，但具体来说在机器学习中的数据也是有划分并区别的，我们有必要先笼统地了解一下。

训练集（Train Set）是用于训练机器学习模型的数据集，在监督学习中，训练集的内容包括样本和对应的输出（也就是我们前面所说的标签），而在无监督模型中，训练集没有对应的输出，仅包含样本。

评估训练后模型的泛化能力（即对未知对应值的数据的预测或者分类能力）、性能等指标的，与训练集相互独立的数据集叫做测试集（Test Set），测试集的样本不包括对应的输出，我们使用测试集来获得对模型性能的评估。

模型训练好后，我们**可以**（这意味着它不是必需的）使用验证集（Validation set）来评估模型的表现，这个概念可能和测试集容易搞混，测试集用于最后评估模型，而验证集用于学习过程中的 *超参数* 调整和选择，验证集的数据是从训练集中划分出来的。



这张散点图是训练集、测试集、验证集的分布示例，其中样本的数量是 训练集 > 测试集 > 验证集。

> 没找到顺手的画图工具，图是用Python的matplotlib写的。

在机器学习算法中，模型的受到多个参数的影响，而我们需要一种可以给我们自己人为控制的参数，这种参数就被叫

做超参数，在训练之前设置的（而不是训练产生的），它存在是为了我们在训练过程中能做些自定义的调整。超参数可以调整模型的复杂度、控制模型的训练过程以及影响模型的泛化能力。

简单线性回归

我们简单了解了一下机器学习是什么，接下来正文开始。

回归分析简介

回归分析是一种统计学方法，用于研究变量之间的关系，并建立一个数学模型来描述和预测因变量与自变量之间的关联，进而用于预测、评估关联性等。这个拟合的数学模型是函数，根据数据特征会再细分成不同类型的函数（例如线性函数、非线性函数）。在回归分析中，我们使用给定的数据来估计回归模型的参数，拟合出回归模型以达到我们的目的。

本文的线性回归是回归分析的一种，说白了就是回归系数要求线性，假设自变量和因变量存在线性关系，我们可以通过一些算法来拟合出这个线性回归模型。

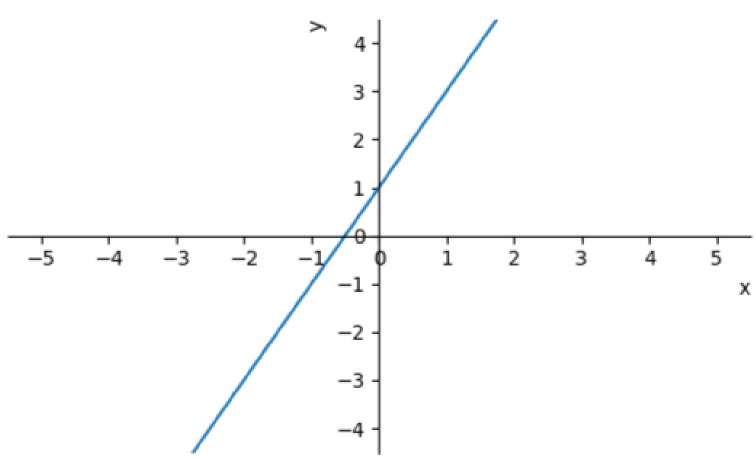
模型表达式

所谓简单线性回归（Simple linear regression）就是一元线性回归，只有一个自变量的情况叫做简单回归，我们初中时都学过最简单的一次函数，形如方程：

$$y = kx + b(k \neq 0)$$

其中 x 是自变量，y 是因变量。k 是斜率（slope），决定了自变量 x 对因变量 y 的影响程度，一个常数项。b是截距（intercept），一个常数项。考虑一个简单的一次函数表达式和它的图像：

$$y = 2x + 1$$



Graph of y = 2x + 1

从图像中我们可以看到一次函数的图像呈直线，也就是自变量 x 和因变量 y 呈线性关系。其中当自变量 x 为0时，因变量 y 的值等于截距 b。一元线性回归模型也是如此，自变量和因变量存在线性关系，所以只要有这样的一个函数（即斜率和截距已知），我们就可以通过自变量预测因变量。在线性回归中，截距 b 的意义是基准值，它表示在自变量为0时因变量的基准值，不过它的意义在实际背景下才能做对应的解释，有些时候自变量为0意味着没有意义（例如用于表示物理量的时候）

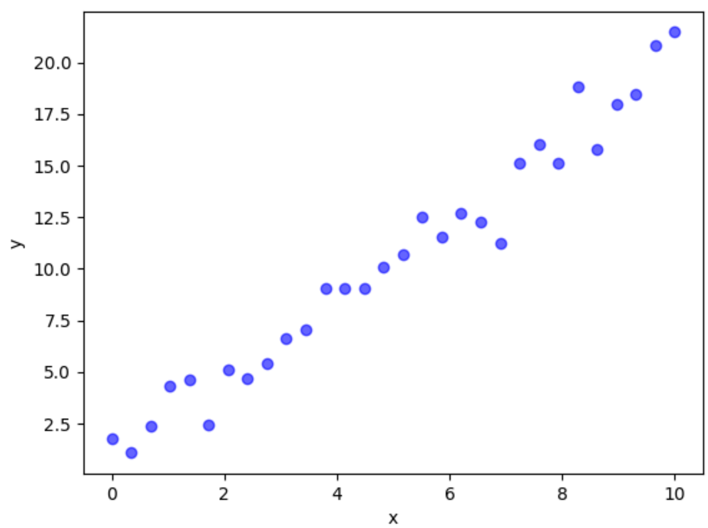
一元线性回归是线性回归中最简单的形式，最直接的前提是自变量和因变量存在线性关系，且每个观测值不会受到其他观测值影响，在后面我们拟合这条直线时还有一些前提会细讲。总而言之，我们只需要拟合出这条直线即可进行预测和推断，而我们接下来将探讨如何找到最佳拟合的斜率和截距。

最小二乘法

我们为算法提供一堆二元组作为训练样本（训练集是它们全部的集合的叫法），一个自变量 x 对应一个因变量 y，所以可以这样表示它们。

$$\langle x_0, y_0\rangle, \langle x_1, y_1\rangle ..., \langle x_n, y_n\rangle$$

但是样本长什么样都有可能，尤其是在现实应用中，在分析 n 组样本时它们很可能只是大致呈线性分布，而无法找到一条直线能经过所有点，也就是这个方程 $y = kx + b$ 无确定解。



例如在这张例图中，30个样本大致呈线性分布，但很显然它们不是一条直线，没有一个确定的规律可以使我们根据它，确定一个自变量对应的值。但是我们可以尽可能求出一个近似解。

想象一条我们猜测的直线 l 在这个例图上，它与这些点整体的分布方向大致一致（不一定经过所有点），然后再从每个点竖直方向向 l 做线段，这个线段的长度也就是猜测的因变量和实际的因变量的误差，我们要考虑的只是如何猜测这条直线的问题。

我们使用一个“帽子”来表示估计值，这样做是区分估计值和实际值，例如在直线 l 的函数图象中 x 对应的 y 是估计值，所以，我们可以设直线 l 的表达式为 $\hat{y} = kx + b$（还在读初中的读者应该很有亲切感吧，有就对了）。我们可以直接想到误差就是 $(y_i - \hat{y}_i)$，这个误差也叫残差（residual），然后我们求它的平方 $(y_i - \hat{y}_i)^2$，这么做最直接的原因是突出较大误差对整体误差的影响，同时避免正负误差的抵消，就都变成正数了。我们来整理一下：

我们用希腊字母 ε（epsilon）来表示残差

$$\varepsilon_i = y_i - \hat{y}_i$$

对于拟合线 l 的表达式 $\hat{y} = kx + b$，我们可以将 $kx + b$ 代入，并以此展开残差平方和公式。

$$\varepsilon_i^2 = y_i - (kx_i + b)$$

$$S = \sum_{i=1}^{n} \varepsilon_i^2 \quad = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - (kx_i + b))^2$$

我们要做的就是使 S 最小，我们先看求解斜率 k 的最优值，我们将残差平方和 S 对斜率 k 求导，并使导数等于0，即 $\frac{\partial S}{\partial k}=0$ ，则残差平方和取极小值。这边提一下，在写这一段的时候我有去看一下《人教A版高中数学(选修三册)》对一元线性回归的解法，在教材中不涉及求导，这里稍微有点点超纲了，不过，求导的方法更为通用，后面我们可以直接推广到多元线性回归，我们在简单线性回归的阶段就考虑好后面也更方便，是吧（心虚）

考虑到没有这方面知识的读者，这里简单解释一下求导是什么，导数（derivative）描述了函数在某一点处的变化率或斜率，想象出一个函数图像，求导就是要我们找出在函数某个点上的斜率。

现在你已经有对导数的基本概念了，我们先求解关于斜率 k 和截距 b 的导数。

$$\frac{\partial S}{\partial k}=\frac{\partial}{\partial k}\sum_{i=1}^{n}(y_i-(kx_i+b))^2$$

$$\frac{\partial S}{\partial b}=\frac{\partial}{\partial b}\sum_{i=1}^{n}(y_i-(kx_i+b))^2$$

在这里，我们将 S 视为一个复合函数，表示为 $f(g(k,b))$，内部函数的解析式为 $g(k,b)=y_i-(kx_i+b)$ ，然后我们通过链式法则分别对内部函数和外部函数进行求导，这样我们就能将导数计算拆分成多个简单的函数导数的乘积。我们先对内部函数 g 进行求导。对于 $y_i-(kx_i+b)$ 的 k 求导，其中 y_i 和 b 不含 k项，所以它们的导数为0，而我们只需要考虑 kx_i 项，所以我们对 k 求导的结果是 $-x_i$。我们再对这个内部函数的 b 进行求导，同理，因为截距 b 项是独立的，所以将其他项视为常数项，则对 b 的求导结果为-1。求导如下：

$$\frac{\partial}{\partial k}(y_i-(kx_i+b))=-x_i$$

$$\frac{\partial}{\partial b}(y_i-(kx_i+b))=-1$$

补充，∂ 是偏导数符号，表示函数在某个变量上的变化率，在我们的残差平方和函数中，y_i 是观测值，$\hat{y}_i$ 是预测的因变量值，斜率 k 和截距 b 在这里才是变量，我们在这里已经完成了对于斜率 k 和截距 b 的偏导数计算。

紧接着我们对外部函数进行求导，若内部函数为 $u=y_i-(kx_i+b)$，则外部函数为 $f(u)=u^2$，根据幂函数的导数公式：

若 $f(g)=x^n$，其中 n 是常数，则导数 $\frac{df}{dx}=nx^{n-1}$，根据这个公式，我们将外部函数带进去，即可进行求导。

$$\frac{df}{du}=\sum_{i=1}^{n}2u^{2-1}=\sum_{i=1}^{n}2u=\sum_{i=1}^{n}2(y_i-(kx_i+b))$$

我们先将对 S 关于斜率 k 求偏导的结果代入进去，继续推导并化简。

$$
\begin{aligned}
\frac{\partial S}{\partial k} &= \sum_{i=1}^{n} 2(y_i - (kx_i + b)) \cdot \frac{\partial}{\partial k} \sum_{i=1}^{n} (y_i - (kx_i + b)) \\
&= \sum_{i=1}^{n} 2(y_i - (kx_i + b)) \cdot (-x_i) \\
&= -2 \sum_{i=1}^{n} x_i (y_i - (kx_i + b))
\end{aligned}
$$

同理，将对 S 关于截距 b 求偏导的结果代入进去，推导并化简。

$$
\begin{aligned}
\frac{\partial S}{\partial b} &= \sum_{i=1}^{n} 2(y_i - (kx_i + b)) \cdot \frac{\partial}{\partial b} (y_i - (kx_i + b)) \\
&= \sum_{i=1}^{n} 2(y_i - (kx_i + b)) \cdot (-1) \\
&= -2 \sum_{i=1}^{n} (y_i - (kx_i + b))
\end{aligned}
$$

我们先从截距 b 开始，将对 S 关于截距 b 的偏导数置为0，如下，移项和化简。

$$
\begin{aligned}
-2 \sum_{i=1}^{n} (y_i - (kx_i + b)) &= 0 \\
2 \sum_{i=1}^{n} (kx_i + b - y_i) &= 0 \\
2 \sum_{i=1}^{n} kx_i + 2 \sum_{i=1}^{n} b - 2 \sum_{i=1}^{n} y_i &= 0 \\
\sum_{i=1}^{n} kx_i + \sum_{i=1}^{n} b &= \sum_{i=1}^{n} y_i \\
\sum_{i=1}^{n} b &= \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} kx_i \\
b &= \frac{\sum_{i=1}^{n} y_i - \sum_{i=1}^{n} kx_i}{n}
\end{aligned}
$$

注意到，x 的平均值为 $\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$， y 的平均值为 $\bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}$，所以

$$
\begin{aligned}
b &= \frac{\sum_{i=1}^{n} y_i - \sum_{i=1}^{n} kx_i}{n} \\
b &= \bar{y} - k\bar{x}
\end{aligned}
$$

我们继续代入推斜率 k，先携带平均值简化后再代入。

$$2\sum_{i=1}^{n}(y_i-(kx_i+b))(-x_i)=0$$
$$(-2)\sum_{i=1}^{n}(x_iy_i-kx_i^2-bx_i)=0$$
$$2\sum_{i=1}^{n}(kx_i^2+bx_i-x_iy_i)=0$$
$$2\sum_{i=1}^{n}(kx_i^2+(\bar{y}-k\bar{x})x_i-x_iy_i)=0$$
$$2\sum_{i=1}^{n}(kx_i^2+\bar{y}x_i-k\bar{x}x_i-x_iy_i)=0$$
$$\sum_{i=1}^{n}kx_i^2+\sum_{i=1}^{n}\bar{y}x_i-\sum_{i=1}^{n}k\bar{x}x_i-\sum_{i=1}^{n}x_iy_i=0$$
$$\sum_{i=1}^{n}kx_i^2-\sum_{i=1}^{n}k\bar{x}x_i=\sum_{i=1}^{n}x_iy_i-\sum_{i=1}^{n}x_i\bar{y}$$
$$k(\sum_{i=1}^{n}x_i^2-\sum_{i=1}^{n}\bar{x}x_i)=\sum_{i=1}^{n}x_iy_i-\sum_{i=1}^{n}x_i\bar{y}$$
$$k=\frac{\sum_{i=1}^{n}x_iy_i-\sum_{i=1}^{n}x_i\bar{y}}{\sum_{i=1}^{n}x_i^2-\sum_{i=1}^{n}x_i\bar{x}}$$
$$k=\frac{\sum_{i=1}^{n}x_iy_i-\sum_{i=1}^{n}x_i\frac{\sum_{i=1}^{n}y_i}{n}}{\sum_{i=1}^{n}x_i^2-\sum_{i=1}^{n}x_i\frac{\sum_{i=1}^{n}x_i}{n}}$$
$$k=\frac{n\sum_{i=1}^{n}x_iy_i-\sum_{i=1}^{n}x_i\sum_{i=1}^{n}y_i}{n\sum_{i=1}^{n}x_i^2-(\sum_{i=1}^{n}x_i)^2}$$

所以，我们推导出了最佳的斜率 k 和最佳截距 b 的公式。

$$k=\frac{n\sum_{i=1}^{n}x_iy_i-\sum_{i=1}^{n}x_i\sum_{i=1}^{n}y_i}{n\sum_{i=1}^{n}x_i^2-(\sum_{i=1}^{n}x_i)^2}$$

$$b=\frac{\sum_{i=1}^{n}y_i-\sum_{i=1}^{n}kx_i}{n}$$

上述的内容是，为了更好读者理解我就自己推了一遍公式（是越推越起劲的这玩意），你没耐心的话只看推导结果即可，别被一大坨数学公式吓着，我们使用Haskell实现这样的算法是很简单的。

（未完成）