线性回归的学习研究

Yuan

2019 年 3 月 22 日

摘要

本文就机器学习中常用的线性回归方法中经典的最小二乘法和用于处理大规模数据问题在线学习算法进行论述。

1 问题定义

线性回归是指给定数据集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_m, y_m)\}$，其中 $\boldsymbol{x}_i$ 是维数为 d 的向量，即 $\boldsymbol{x}_i = (x_{i1}; x_{i2}; \cdots; x_{id})$，$y_i \in \mathbb{R}$。线性回归试图学得 $f(\boldsymbol{x}_i) = \boldsymbol{w}^\top \boldsymbol{x}_i + b$，使得 $f(\boldsymbol{x}_i)$ 接近 y_i，$\boldsymbol{w}$ 及 b 为参数，$\boldsymbol{w}^\top = (w_1, w_2, \cdots, w_d)$。

2 学习算法

2.1 最小二乘法

最小二乘法是解决线性回归问题的经典方法，它通过最小化误差的平方和寻找数据的最佳函数匹配。为了更清晰的数学表示，这里将原始问题中的一些量进行合并和改写。首先将 $\boldsymbol{w}$ 和 b 合并成一个新的向量：

$$\hat{\boldsymbol{w}} = \begin{pmatrix} \boldsymbol{w} \\ b \end{pmatrix}$$

同时把数据集 D 表示为一个 $m \times (d+1)$ 大小的矩阵 $\boldsymbol{X}$，其中每一行对应一个实例，该行 d 个元素对应实例的 d 个属性值，最后一个元素恒为 1：

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1 \\ x_{21} & x_{22} & \cdots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{md} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{x_1}^\top & 1 \\ \boldsymbol{x_2}^\top & 1 \\ \vdots & \vdots \\ \boldsymbol{x_m}^\top & 1 \end{pmatrix}$$

同时令：

$$\boldsymbol{y}^\top = (y_1, y_2, \cdots, y_m)$$

最小二乘法采用平方损失，即：

$$J(\hat{\boldsymbol{w}}) = \|\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y}\|^2$$

当 $\|\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y}\|^2$ 关于 $\hat{\boldsymbol{w}}$ 的导数[1]为 $\mathbf{0}$ 时，损失达到最小。若用矩阵的形式改写这个式子则有：

$$\|\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y}\|^2 = (\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y})^\top (\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y})$$

得到其关于 $\hat{\boldsymbol{w}}$ 的导数为：

$$\frac{dJ(\hat{\boldsymbol{w}})}{d\hat{\boldsymbol{w}}} = 2\boldsymbol{X}^\top (\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y}) = \mathbf{0}$$

整理得：

$$\hat{\boldsymbol{w}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

通过矩阵计算就能得到要学得的参数 $\hat{\boldsymbol{w}}$，但如果输入矩阵 $\boldsymbol{X}$ 中存在线性相关或者近似线性相关的列，那么输入矩阵 $\boldsymbol{X}$ 就会变成或者近似变成奇异矩阵（singular matrix）。这是一种病态矩阵，矩阵中任何一个元素发生一点变动，整个矩阵的行列式的值和逆矩阵都会发生巨大变化。这将导致最小二乘法对观测数据的随机误差极为敏感，进而使得最后的线性模型产生非常大的方差，这个在数学上称为多重共线性（multicollinearity）。

[1]此处求导后的结果为向量，即梯度。

2.2 在线学习算法

最小二乘法通过计算最后的参数矩阵学得参数，当样本数目非常多的时候，其计算成本会非常的大。故当数据集特别大的时候，这种算法就不再适用，就有一种称为序列学习算法（squential learning），也叫作在线算法（online algorithms）的算法得到了广泛的应用。

序列学习算法就是将数据的样本点，看成在时间上有序的，即数据集合不是一次性得到全部的数据，而是一个样本一个样本采集得到的。那么，在训练算法的时候，也就可以一个样本一个样本的输入的训练算法的中，利用新的样本不断的改进模型，进而达到学习的目的。随机梯度下降法常应用于这类迭代计算的最优化问题中。随机梯度下降法基于梯度下降法，所以先介绍梯度下降法。

梯度下降法 由前面可以知道，线性回归的损失函数为:

$$J(\hat{\boldsymbol{w}}) = \|\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y}\|^2$$

其关于参数 $\hat{\boldsymbol{w}}$ 梯度为:

$$\nabla J(\hat{\boldsymbol{w}}) = 2\boldsymbol{X}^\top(\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y})$$

梯度下降法就是:

$$\hat{\boldsymbol{w}}_k = \hat{\boldsymbol{w}}_{k-1} - \eta\nabla J(\hat{\boldsymbol{w}}_{\boldsymbol{k-1}})$$

式中 η 为学习率，控制下降的速度，参数 k 为迭代轮数。通过不停迭代，求得 $\hat{\boldsymbol{w}}_k$。

随机梯度下降法 不同于上述的梯度下降法，每次迭代都使用了全部的数据集，该算法每次迭代仅仅使用一个样本。该样本从样本集合中随机选取，故名为随机梯度下降法:

$$\hat{\boldsymbol{w}}_k = \hat{\boldsymbol{w}}_{k-1} - \eta\nabla \bar{J}(\hat{\boldsymbol{w}}_{\boldsymbol{k-1}})$$

$$\nabla \bar{J}(\hat{\boldsymbol{w}}) = 2\bar{\boldsymbol{x}}_r(\bar{\boldsymbol{x}}_r^\top\hat{\boldsymbol{w}} - y_r)$$

式中 η 同样为学习率，参数 k 为迭代轮数。$\bar{J}(\hat{\boldsymbol{w}})$ 为单样本情况下的损失函数的梯度。由于每次更新只用一个样本更新 $\hat{\boldsymbol{w}}$，故之前梯度中的矩阵 $\boldsymbol{X}$

退化为一个行向量，向量 $\boldsymbol{y}$ 退化为标量。定义集合 $N = \{1, 2, \cdots, m\}$，m 为数据集 D 的样本个数，r 从 N 从中随机选择。$\bar{\boldsymbol{x}}_r$ 为从矩阵 $\boldsymbol{X}$ 中选出的第 r 行的行向量的转置，即 $\bar{\boldsymbol{x}}_r = \boldsymbol{X}_{r,*}^{\top}$。同样地，$y_r$ 也为向量 $\boldsymbol{y}$ 的第 r 行的标量。

3 总结

线性回归是一种形式简洁，用途广泛的模型。本文论述了该模型的两种学习算法：最小二乘法及随机梯度下降法。最小二乘法利用最小化平方损失来优化参数，最终可以归结为一系列矩阵运算从而求得模型参数。随机梯度下降法使用最小二乘法提出的损失函数的梯度进行迭代式的数值运算，同时还使用随机选择样本对总体样本进行近似，降低了运算规模，适合大规模数据下的模型学习。