跟我学人工智能

刘森

2018 年 1 月 29 日

你好，世界hello, world

1 机器学习算法

不同的算法，本身没有好坏之分，有的只是，根据不同的场景选择合适的算法。

线性回归和Logistic回归，虽然听起来都叫作“回归”，但其实两者却是做不一样的事情：一个是做连续数据的预测，一个是做离散数据的预测；一个是真正做回归的，一个是做分类的，它们两个【用途】是完全不一样的。【如何推导出来？】线性回归是用高斯分布的方式推导出来，Logistic回归既然是做分类，就用Bnody分布，两点分布来推导出来。两者大的工具都是【最大似然估计】。在线性回归里面，要讨论一个东西：【最小二乘法的本质是什么】。或者说，为什么有最小二乘法呢？有没有最小三乘法呢？有没有最小四乘法呢？在【线性回归】和【Logistic回归】中强调两个工具：【梯度下降算法】和【极大似然估计】。

1.1 线性回归

高斯分布
极大似然估计MLE
最小二乘法的本质

1.1.1 什么是线性回归

线性回归$y = ax + b$

考虑多个变量情形，例如两个变量，$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$ ，可以写成如下形式：

$$h_\theta(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$$

其中，θ展开后，呈现如下形式：

$$\begin{pmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{pmatrix}$$

其中，x展开后，呈现如下形式：

$$\begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}$$

上式中的1就表示x_0，而相应的θ_0表示截距，是比较难以直接解释的。

再把上面的式子拿过来，

$$h_\theta(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$$

在$h_\theta(x)$中，x看起来是【自变量】，但事实上是【样本】，所以x是已知的，而θ是未知的，我们要通过某一种办法来求解出θ，这个就是【线性回归要解决的问题】。

第05课《回归》00:10:30

目前讲的问题是【what】，即什么是线性回归。过一会儿，会讲【how】，用什么样的工具去求，如何去求的问题。

1.1.2 使用极大似然估计解释最小二乘

第05课《回归》00:20:00

使用极大似然估计解释最小二乘

$$y^{(i)} = \theta^T x Y(i) + \epsilon^{(i)}$$

the $\epsilon^{(i)}$ are distributed IID (independently and identically distributed) according to a Gaussian distribution (also called a Normal distribution) with mean *zero* and some variance σ^2.

误差$\epsilon^{(i)}(1 \leq i \leq m)$是独立同分布的，服从均值为0，方差为某定值$\sigma^2$的【高斯分布】。原因：【中心极限定理】，可以查阅一下“中心极限定理的意义”。

似然函数第05课《回归》00:21:21

首先，两边是相等的：

$$y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)}$$

其中，$x^{(i)}$表示第i个【样本】，$\theta^T x^{(i)}$表示第i个样本的【预测值】，$y^{(i)}$表示第i个样本的【真实值】，而$\epsilon^{(i)}$表示第i个样本的误差。

根据【中心极限定理】，$\epsilon^{(i)}$应该是呈现一个高斯分布的形态。

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(\epsilon^{(i)})^2}{2\sigma^2})$$

另外，$\epsilon^{(i)} = y^{(i)} - \theta^T x^{(i)}$，此时将$\epsilon^{(i)}$代入上式：

$$p(y^i|x^i;\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2})$$

如此一来，上式当中就没有误差ϵ了，因此只要指定了x和θ，就可以认为是一个y的分布。换句话讲，y其实服从的是【均值是$\theta^T x$，方差是某一个σ的高斯分布（正态分布）】。

那么，用什么可以估计这个θ呢？答：【最大似然估计】。

在上面的公式中，i只是表示第i个样本，假设一共有m个样本，那么，【m个样本的似然估计】就可以表示为：

$$L(\theta) = \prod_{i=1}^{m} p(y^i|x^i;\theta) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2})$$

如此一来，怎么求θ呢？直接对【似然函数】取对数，然后再想办法。

高斯的对数似然与最小二乘

$$
\begin{aligned}
l(\theta) &= \log L(\theta) \\
&= \log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}) \\
&= \sum_{i=1}^{m} \log \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}) \\
&= m \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2} \cdot \frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \theta^T x^{(i)})^2
\end{aligned} \tag{1}
$$

现在，其实是通过【最大似然估计】加上【高斯分布】来得到了【最小二乘法】目标函数。换句话说，这就是解释的“为什么会有最小二乘法”这个概念。

$$
J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \tag{2}
$$

1.1.3 θ的解析式的求解过程

θ的解析式的求解过程第05课《回归》00:29:52

将M个N维样本组成矩阵X：（1）X的每一行对应一个样本，共M个样本（measurements）；（2）X的每一列对应样本的一个维度，共N维（regressors）（3）还有额外的一维常数项$x_0^{(i)}$，全为1。

目标函数：

$$
J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 = \frac{1}{2} (X\theta - y)^T (X\theta - y) \tag{3}
$$

梯度：

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta (\frac{1}{2} (X\theta - y)^T (X\theta - y)) \\
&= \nabla_\theta (\frac{1}{2} (\theta^T X^T - y^T)(X\theta - y)) \\
&= \nabla_\theta (\frac{1}{2} (\theta^T X^T X\theta - \theta^T X^T y - y^T X\theta + y^T y)) \\
&= \frac{1}{2} (2X^T X\theta - X^T y - (y^T X)^T) \\
&= X^T X\theta - X^T y
\end{aligned} \tag{4}
$$

在上式中，求驻点，令$X^TX\theta - X^Ty = 0$。
所以θ取值如下：

$$\theta = (X^TX)^{-1} \cdot X^Ty \tag{5}$$

1.1.4 最小二乘法意义下的参数最优解

参数的解析解

$$\theta = (X^TX)^{-1}X^Ty \tag{6}$$

若X^TX不可逆或防止过拟合，增加λ扰动

$$\theta = (X^TX + \lambda I)^{-1}X^Ty \tag{7}$$

第05课《回归》00:38:36
“简单”方法记忆结论

$$\begin{aligned} & X\theta = y \\ & \Rightarrow X^TX\theta = X^Ty \\ & \Rightarrow \theta = (X^TX)^{-1}X^Ty \end{aligned} \tag{8}$$

1.1.5 梯度下降算法Gradient Descent

第05课《回归》01:14:49
事实上，通过解析解的方式$\theta = (X^TX)^{-1}X^Ty$来求解θ是没有问题的，真的是对的；但是，我们往往在机器学习中，习惯在这个地方引出“梯度下降算法”，并且也习惯使用“梯度下降算法”来求解θ。我们这里还是求目标函数$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$的梯度，延着负梯度方向不停的下降，降到某一个值，降不下去了，我们就可以知道，得到了一个局部的极小值，这个局部的极小值点，或许就是我们想要的θ。
初始化θ（随机初始化）
沿着负梯度方向迭代，更新后的θ使$J(\theta)$更小

$$\theta_j = \theta_j - \alpha \cdot \frac{\partial J(\theta)}{\partial \theta} \tag{9}$$

其中，α表示学习率、步长。这个步长α事实上是有办法可以指定比较优的，后续再讲怎么选α会更优。

1.1.6 梯度方向

梯度方向第05课《回归》01:15:37

$$\begin{aligned}\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{1}{2}(h_\theta(x)-y)^2 \\ &= 2\cdot\frac{1}{2}(h_\theta(x)-y)\cdot\frac{\partial}{\partial\theta_j}(h_\theta(x)-y) \\ &= (h_\theta(x)-y)\cdot\frac{\partial}{\partial\theta_j}(\sum_{i=0}^{n}\theta_i x_i - y) \\ &= (h_\theta(x)-y)x_j\end{aligned} \tag{10}$$

这里是求偏导，一共有n个特征，当前求解的是第j个特征的偏导。得到这个梯度之后，我们就可以延着负梯度的方向下降下去就可以了。

这里有个需要注意的地方，我们求出来的是梯度，而我们要用的是**负梯度**，而**负梯度**应该是$-(h_\theta(x)-y)x_j$，因此可以调换一下$h_\theta(x)$和y的位置，写成这种形式$(y-h_\theta(x))x_j$。

1.1.7 批量梯度下降算法

Repeat until convergence(会聚; 集收敛)

$$\theta_j := \theta_j + \alpha\sum_{i=1}^{m}(y^{(i)}-h_\theta(x^{(i)}))x_j^{(i)} \tag{11}$$

gradient descent. Note that, while gradient descent can be susceptible to local minima in general, the optimization problem we have posed here for linear regression has only one global, and no other local, optima; thus gradient always converges (assuming the learning rate α is not too large) to the global minimum. Indeed, J is a convex(凸形; 凸的) quadratic(二次的;) function.

1.1.8 随机梯度下降算法

for $i=1$ to m

$$\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)} \tag{12}$$

This algorithm is called **stochastic gradient descent** (also **incremental gradient descent**). Whereas batch gradient descent has to scan through the entire training set before taking a single step – a costly operation if m is large – stochastic gradient descent can start making progress right away, and continues to make progress with each example it looks at. Often, stochastic gradient descent gets θ "close" to the minimum much faster than batch gradient descent. (Note however that it may never "converge" to the minimum, and the parameters θ will keep oscillating(振荡;（使）摆动) around the minimum of $J(\theta)$; but in practice most of the values near the minimum will be reasonably good approximations to the true minimum.) For these reasons, particularly when the training set is large, stochastic gradient descent is often preferred over batch gradient descent.

在没有明确**批量梯度下降**和**随机梯度下降**哪个更优的情况下，优先选择“随机梯度下降”。

在求解θ的时候，不一定非要是最好的，它只能说是**堪用**的，能够work的，可用的就行了。有的时候，不要追求完美，完美往往是达不到的，只要能够过得去，还可以，就行了。

1.1.9 折中：mini-batch

第05课《回归》01:22:37

如果不是每拿到一个样本即更改梯度，而是若干个样本的平均梯度作为更新方向，则是mini-batch梯度下降算法。

机器学习(Machine learning)，第一，要会理论，它是一个能够走多远的基础；第二，是要会写代码，它是保证当前的工作能够持续下去。“渔”和“鱼”都得要。

机器学习中，很多时间都花在了**如何建模型、如何调参、如何选特征、如何优化模型**这些事情上，写代码并没有那么的困难。

1.1.10 权值的设置

高斯核函数第05课《回归》01:45:26

我们希望用“线性回归”求得模型的“残差”服从高斯分布（正态分布）；如果不服从高斯分布，我们就去重新选“特征”。如果不服从高斯分布，说明有些“特征”没有被考虑进来。

1.2 逻辑回归：分类问题的首选算法

第05课《回归》02:00:30

1.2.1 Logistic/sigmoid函数

$$g(z) = \frac{1}{1+e^{-z}} \tag{13}$$

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}} \tag{14}$$

$$\begin{aligned} g'(x) &= (\frac{1}{1+e^{-x}})' = \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\ &= \frac{1}{1+e^{-x}} \cdot (1 - \frac{1}{1+e^{-x}}) \\ &= g(x) \cdot (1 - g(x)) \end{aligned} \tag{15}$$

1.2.2 Logistic回归参数估计

假定

$$\begin{aligned} P(y=1|x;\theta) &= h_\theta(x) \\ P(y=0|x;\theta) &= 1 - h_\theta(x) \end{aligned} \tag{16}$$

上面的两个式子，可以用一个式子来表示：

$$p(y|x;\theta) = (h_\theta(x))^y (1-h_\theta(x))^{1-y} \tag{17}$$

那么似然函数$L(\theta)$这样求：

$$\begin{aligned} L(\theta) &= p(\vec{y}|X;\theta) \\ &= \prod_{i=1}^{m} p(y^{(i)}|x^{(i)};\theta) \\ &= \prod_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}} (1-h_\theta(x^{(i)}))^{1-y^{(i)}} \end{aligned} \tag{18}$$

1.2.3 对数似然函数

第05课《回归》02:01:23

$$\begin{aligned} l(\theta) &= \log L(\theta) \\ &= \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)})) \end{aligned} \tag{19}$$

$$\begin{aligned} \frac{\partial}{\partial\theta_j} l(\theta) &= (y\frac{1}{g(\theta^T x)} - (1-y)\frac{1}{1-g(\theta^T x)})\frac{\partial}{\partial\theta_j} g(\theta^T x) \\ &= (y\frac{1}{g(\theta^T x)} - (1-y)\frac{1}{1-g(\theta^T x)}) g(\theta^T x)(1-g(\theta^T x))\frac{\partial}{\partial\theta_j}\theta^T x \\ &= (y(1-g(\theta^T x)) - (1-y)g(\theta^T x))x_j \\ &= (y - h_\theta(x))x_j \end{aligned} \tag{20}$$

1.2.4 参数的迭代

第05课《回归》02:01:55

Logistic回归参数的学习规则

$$\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)} \tag{21}$$

比较上面的结果和“线性回归”的结论的差别：它们具有相同的形式，Logistic回归是用于做分类的，属于广义线性回归。

1.2.5 对数线性模型

一个事件的**几率**odds，是指该**事件发生的概率**与**事件不发生的概率**的比值。

对数几率：$\log it$函数

$$\begin{aligned} P(y=1|x;\theta) &= h_\theta(x) \\ P(y=0|x;\theta) &= 1-h_\theta(x) \end{aligned} \tag{22}$$

$$\begin{aligned} \log it(p) &= \log \frac{p}{1-p} \\ &= \log \frac{h_\theta(x)}{1-h_\theta(x)} \\ &= \log \frac{\frac{1}{1+e^{-\theta^T x}}}{\frac{e^{-\theta^T x}}{1+e^{-\theta^T x}}} \\ &= \theta^T x \end{aligned} \tag{23}$$

说一个事儿：如果有一个x_1和x_2，正常去求x_1占多大比例的方法是

$$\frac{x_1}{x_1+x_2} \tag{24}$$

如果不想这么算，而是想做一个指数的比例，那么x_1和x_2就变成了e^{x_1}和e^{x_2}，再求比例就是

$$\frac{e^{x_1}}{e^{x_1}+e^{x_2}} = \frac{1}{1+e^{x_2-x_1}} \tag{25}$$

如果令$x_2-x_1=-z$，就得到：

$$\frac{1}{1+e^{-z}} \tag{26}$$

这样就得到了sigmoid函数。

1.3 工具

梯度下降算法
极大似然估计

1.4 Softmax

1.5 聚类

按道理来说，“算法”和“模型”是两个完全不同的概念，算法一般用

于表示解决特定的数学问题，而模型则侧重于解决实际的问题，但在不同场景下，两者也会经常混用。

“聚类”是无监督的机器学习算法，而“线性回归、Logistic回归、Softmax、SVM、随机森林”都是有监督的机器学习算法。

“聚类”并不像“有监督分类”那样有y值，而只有$m \times n$维的向量数据，其中m表示m个样本，n表示有n个特征。“聚类”并不依赖于y值，而是依赖于$m \times n$维的数据，根据其内部之间的相似性，来做聚类。

一个$m \times n$维的数据，经过某种聚类算法之后，这m个样本聚类到k个簇当中，这样就把一个$m \times n$维矩阵转换成了一个$m \times k$维矩阵，这本质上就是一个聚类的过程。我们发现，这个数据是从一个$m \times n$维矩阵转换成了一个$m \times k$维矩阵，这本质上又是一个降维的过程。所以“聚类”这个词和“降维”这个词，两者本质上是一样的。

1.5.1 本次目标

掌握**K-means聚类**的思路和使用条件

了解**层次聚类**的思路和方法

理解**密度聚类**并能够应用于实践：（1）DBSCAN；（2）密度最大值聚类

掌握**谱聚类**的算法：考虑谱聚类和**PCA**的关系。

1.5.2 聚类的定义

聚类就是对大量未知标注的数据集，按数据的内在相似性将数据集划分为多个类别，使类别内的数据相似度较大而类别间的数据相似度较小。

而这种**内在相似性**，通常用**相似度**或**距离**来度量。往往，**距离**求出来之后，将距离的-1次方作为**相似度**。

1.5.3 相似度/距离计算方法总结

第10课聚类00:11:18

闵可夫斯基距离Minkowski

$$dist(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}} \tag{27}$$

在上式中，当$p = 2$时，就是标准的“欧氏距离”；当$p = 1$时，就是“曼哈顿距离”；当$p = \infty$时，就相当于取$|x_i - y_i|$的最大距离，就称为“切比雪夫距离”。

杰卡德相似系数（Jaccard），从集合的角度来理解

$$J(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|} \tag{28}$$

余弦相似度（cosine similarity），从角度来理解

$$\cos(\theta) = \frac{a^T b}{|a| \cdot |b|} \tag{29}$$

Pearson相关系数（Pearson CorrelationCoefficient）是用来衡量两个数据集合是否在一条线上面，它用来衡量定距变量间的线性关系。

$$\begin{aligned} \rho_{XY} &= \frac{cov(X,Y)}{\sigma_X \sigma_Y} \\ &= \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \\ &= \frac{\sum_{i=1}^{n}(X_i-\mu_X)(Y_i-\mu_Y)}{\sqrt{\sum_{i=1}^{n}(X_i-\mu_X)^2}\sqrt{(\sum_{i=1}^{n}Y_i-\mu_Y)^2}} \end{aligned} \tag{30}$$

假设$A = (A_1, A_2, ...A_n)$，有n个数，我们可以求它的平均值$\bar{A}$，那么$\frac{1}{n}(A_i - \bar{A})^2$就可以认为是随机变量A的均方差。同时，有$B = (B_1, B_2, ...B_n)$，也有$n$个数，它的均方差也可以表示为$\frac{1}{n}(B_i - \bar{B})^2$。也可以用$\frac{1}{n}(A_i - \bar{A})(B_i - \bar{B})$来度量$A$和$B$之间的协方差。这个$A$和$B$之间协方差，也可以除以它的标准差，就有了Pearson相关系数。

相对熵（K-L距离）

$$D(p \parallel q) = \sum_x p(x) \log \frac{p(x)}{q(x)} = E_{p(x)} \log \frac{p(x)}{q(x)} \tag{31}$$

Hellinger距离

$$D_\alpha(p \parallel q) = \frac{2}{1-a^2}\left(1 - \int p(x)^{\frac{1+\alpha}{2}} q(x)^{\frac{1-\alpha}{2}} dx\right) \tag{32}$$

注意：不要拘泥于某种相似性的计算方式，在实践当中，有些场景，适合用欧氏距离，如果是推荐系统，就习惯用杰卡德系数，如果是文本的相似度，就可能用余弦相似度，这都是有可能的。

我们要说明的就是，在“聚类”算法当中，可以任意挑选一个“相似度计算方法”，就能够算出样本i和样本j的相似性S_{ij}；如果有m个样本，就形成一个$m \times m$的相似度方阵，后面就对$m \times m$的方阵使用各种手段来去做聚类。至于说用哪一个“相似度”算法，则是一个相对独立的事情。

1.5.4 Hellinger distance

第10课聚类00:20:50

1.5.5 余弦相似度与Pearson相似系数

第10课聚类00:24:50

n维向量x和y的夹角记作θ，根据余弦定理，其余弦值为：

$$\cos(\theta) = \frac{x^T y}{|x| \cdot |y|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \tag{33}$$

这两个向量的Pearson相关系数是：

$$\begin{aligned} \rho_{XY} &= \frac{cov(X,Y)}{\sigma_X \sigma_Y} \\ &= \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \\ &= \frac{\sum_{i=1}^{n}(X_i-\mu_X)(Y_i-\mu_Y)}{\sqrt{\sum_{i=1}^{n}(X_i-\mu_X)^2}\sqrt{(\sum_{i=1}^{n} Y_i-\mu_Y)^2}} \end{aligned} \tag{34}$$

如果此时的μ_X和μ_Y都为0，则此时的Pearson相关系数恰好是“余弦相似度”。

相关系数即将x、y坐标向量各自平移到原点后的夹角余弦!

这即解释了为何文档间求距离使用夹角余弦——因为这一物理量表征了文档去均值化后随机向量间相关系数。

1.5.6 聚类的基本思想

第10课聚类00:27:32

给定一个有N个对象的数据集，构造数据的k个簇，$k \leq n$。满足下列条件：

（1）每一个簇至少包含一个对象

（2）每一个对象属于且仅属于一个簇

（3）将满足上述条件的k个簇称作一个合理划分。

基本思想：对于给定的类别数目k，首先给出初始划分，通过迭代改变**样本**和**簇**的隶属关系，使得每一次改进之后的划分方案都较前一次好。

1.5.7 K-means算法

K-means算法，也称为k-平均或k-均值，是一种广泛使用的聚类算法，或者成为其他聚类算法的基础。

假定输入样本为$S = x_1, x_2, ...x_m$，则算法步骤为：

（1）选择初始的k个类别中心$\mu_1, \mu_2, ...\mu_k$

（2）对于每个样本x_i，将其标记为距离类别中心最近的类别，即：

$$label_i = \arg\min_{1 \le j \le k} \parallel x_i - \mu_j \parallel \tag{35}$$

（3）将每个类别中心更新为隶属该类别的所有样本的均值

$$\mu_j = \frac{1}{|c_j|} \sum_{i \in c_j} x_j \tag{36}$$

（4）重复最后两步，直到类别中心的变化小于某阈值。

中止条件：迭代次数、簇中心变化率、最小平方误差MSE(Minimum Squared Error)

很好的一个问题就是：（1）K-means算法中的k如何选择？（2）k个中心如何初始化？

对于第一个问题，一般是使用两种方式。第一种，有的时候，可以通过“先验的知识”来确定的，比如说，抛骰子只能有6种可能的数值。第二种，就是交叉验证，不断的尝试k的大小，来看看最小平方误差是否会减小。当k没有更好的办法选择时，只能够通过相互交叉验证的方式帮助我们做。

第二个问题解答第10课聚类00:39:00

对于第二个问题，一种是随机的给定初始值。

1.5.8 K-means的公式化解释

第10课聚类00:55:32

记K个簇中心为$\mu_1, \mu_2, ...\mu_k$，每个簇的样本数目为$N_1, N_2, ..., N_k$

使用平方误差作为目标函数：

$$J(\mu_1,\mu_2,...\mu_k)=\frac{1}{2}\sum_{j=1}^{K}\sum_{i=1}^{N_j}(x_i-\mu_j)^2 \tag{37}$$

该函数为关于$\mu_1,\mu_2,...\mu_k$的凸函数，其驻点为：

$$\begin{aligned}\frac{\partial J}{\partial \mu_j}&=\sum_{x_i\in\{J\}}(x_i-\mu_j)=0\\ &\Rightarrow \mu_j=\frac{1}{N_j}\sum_{x_i\in\{J\}}x_i\end{aligned} \tag{38}$$

k-均值的聚类结果，一定是类“圆”的。

1.5.9 K-means聚类方法总结

第10课聚类01:01:32

优点：

（1）是解决聚类问题的一种经典算法，简单，快速

（2）对处理大数据集，该算法保持可伸缩性和高效率

（3）当簇近似为高斯分布时，它的效果较好

缺点：

（1）在簇的平均值可被定义的情况下才能使用，可能不适用于某些应用

（2）必须事先给出k（要生成的簇的数目），而且对初值敏感，对于不同的初始值，可能会导致不同结果

（3）不适合于发现非凸形状的簇或者大小差别很大的簇

（4）对噪声和孤立点数据敏感

可作为其他聚类方法的基础算法，如谱聚类。

1.5.10 对K-means的思考：K-Mediods聚类(K中值距离)

第10课聚类01:02:40

K-Means将簇中所有点的均值作为新质心，若簇中含有异常点，将导致均值偏离严重。以一维数据为例：

（1）数据1、2、3、4、100的均值为22，显然距离“大多数”数据1、2、3、4比较远

（2）改成求数组的中位数3，在该实例中更为稳妥。

（3）这种聚类方式即K-Mediods聚类（K中值距离）

初值的选择，对聚类结果有影响吗？如何避免？

1.5.11 轮廓系数(Silhouette)

第10课聚类01:07:37

Silhouette系数是对聚类结果有效性的解释和验证，由Peter J. Rousseeuw于1986年提出。

计算样本i到同簇其他样本的平均距离a_i。a_i越小，说明样本i越应该被聚类到该簇。将a_i称为样本i的簇内不相似度。簇C中所有样本的a_i均值称为簇C的簇不相似度。

计算样本i到其他某簇C_j的所有样本的平均距离b_{ij}，称为样本i与簇C_j的不相似度。定义为样本i的簇间不相似度：$b_i = \min\{b_{i1}, b_{i2}, ...b_{iK}\}$。其中，$b_i$越大，说明样本$i$越不属于其他簇。

根据样本i的簇内不相似度a_i和簇间不相似度b_i，定义样本i的轮廓系数：

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

s_i接近于1，则说明样本i聚类合理；s_i接近于-1，则说明样本i更应该聚类到另外的簇；若s_i近似为0，则说明样本i在两个簇的边界上。

所有的样本的s_i的均值，称为聚类结果的轮廓系数，是该聚类是否合理、有效的度量。

1.5.12 层次聚类方法

第10课聚类01:19:19

层次聚类方法对给定的数据集进行层次的分解，直到某种条件满足为止。具体又分为：（1）凝聚的层次聚类：AGNES算法，（2）分裂的层次聚类：DIANA算法。

凝聚的层次聚类（AGNES算法），是一种自底向上的策略，首先将每个对象作为一个簇，然后合并这些原子簇为越来越大的簇，直到某个终结条件被满足。

分裂的层次聚类（DIANA算法），采用自顶向下的策略，它首先将所有对象置于一个簇中，然后逐渐细分为越来越小的簇，直到达到了某个终

结条件。

1.5.13 密度聚类方法

第10课聚类01:22:39

密度聚类算法的指导思想是，只要样本点的密度大于某阈值，则将该样本添加到最近的簇中。

这类算法能克服基于距离的算法只能发现“类圆形”的聚类的缺点，可发现任意形状的聚类，且对噪声数据不敏感。但计算密度单元的计算复杂度大，需要建立空间索引来降低计算量。

常用的两个算法：DBSCAN算法和密度最大值算法。

DBSCAN算法（Density-Based Spatial Clustering of Applications with Noise）是一个比较有代表性的基于密度的聚类算法。与划分和层次聚类方法不同，它将簇定义为密度相连的点的最大集合，能够把具有足够高密度的区域划分为簇，并可在有“噪声”的数据中发现任意形状的聚类。

密度最大值聚类是一种简洁优美的聚类算法，可以识别各种形状的类簇，并且参数很容易确定。

第10课聚类01:40:39

1.5.14 谱和谱聚类

第10课聚类01:57:06

方阵作为线性算子，它的所有特征值的全体统称方阵的谱。

（1）方阵的谱半径为最大的特征值（2）矩阵A的谱半径：A^TA的最大特征值

谱聚类是一种基于图论的聚类方法，通过对样本数据的拉谱拉斯矩阵的特征向量进行聚类，从而达到对样本数据聚类的目的。

2 支持向量机SVM

2.1 复习：对偶问题

第09课SVM 00:01:22 复习：对偶问题

一般优化问题的Lagrange乘子法

Lagrange函数

【lsieun】“仿射”和“线性变换”似乎是同一个意思。

第09课SVM 00:01:22 复习：Lagrange对偶函数(dual function)

概念：KKT条件

【lsieun】Lagrange函数的最大值，就等于“原函数”的最小值。这里主要是讲Lagrange乘子法作为一种工具，将原来求“最小值”的问题，转换为求“最大值”的问题。

第09课SVM 00:03:42 线性方程的最小二乘问题

此处是举例子，用于解释上面的“由最小值转换成求最大值的解决方法”。

第09课SVM 00:04:26 强对偶条件

若要对偶函数的最大值即为原问题的最小值，考察需要满足的条件

第09课SVM 00:05:37 强对偶KKT条件: Karush-Kuhn-Tucker

2.2 主要内容和目标

理解支持向量机SVM的原理和目标SVM核心的东西what.

掌握支持向量机的计算过程和算法步骤SVM核心的东西how.

理解软间隔最大化的含义：（1）对线性不可分的数据给出（略有错误）的分割面；（2）线性可分的数据需要使用“软间隔”目标函数吗？

了解核函数的思想

了解SOM算法的过程

核函数：SVM本身是个线性分类器，那么在原始的SVM上，可以加上一些核函数，来构造一个非线性的分隔面，来更好的解决分类问题。

2.3 各种概念

第09课SVM 00:09:32 各种概念

SVM进行简单的化，可以分为三类：

第一类，线性可分支持向量机：（1）硬间隔最大化hard margin maximization；（2）硬间隔支持向量机

第二类，线性支持向量机：（1）软间隔最大化soft margin maximization；（2）软间隔支持向量机

第三类，非线性支持向量机：（1）核函数kernel function

从学习的角度来说，第一类（线性可分支持向量机）是最重要的，只要学会了第一类，稍微加一点东西就能变成第二类，对第二类稍微加一点东西就能变成第三类。[知识梯度lsieun]

在实际应用中，使用的较多的就是第二类和第三类了。

2.4 分隔超平面

第09课SVM 00:13:37 分隔超平面

[lsieun]什么是超平面？概率怎么理解，有时间查一查。之前查的时候，印象中是说，超过二维的平面，都叫超平面。

第09课SVM 00:13:37 分隔超平面的思考

如何定义两个集合的“最优”分割超平面？（1）找到集合“边界”上的若干点，以这些点为“基础”计算超平面的方向；以两个集合边界上的这些点的平均作为超平面的“截距”；（2）支持向量：support vector

若两个集合有部分相交，如何定义超平面，使得两个集合“尽量”分开？

第09课SVM 00:16:00 线性分类问题

假定一共有N个样本，最终可能只有n个样本参与到支持向量(support vector)里去了，一般而言，N是大于n的，甚至是N远大于n的，就比如说有10000个样本，有20个样本参与到支持向量里去了，相当于有9980都是零，只有20个非零，所以SVM是个稀疏的模型。在有些教材中，会特意将SVM放在“稀疏模型”中介绍。

第09课SVM 00:20:00 CNN 卷积神经网络也是个稀疏模型。在这一点上（都是稀疏模型），SVM和CNN是相似的。如果在面试中谈到了，或许是一个加分项呢，哈哈。。。

事实上，有很多很多条直线可以将两部分图形分开，但是谁是最优的直线呢？如果我们知道了“哪条直线是最优的”，我们又怎么把“目标函数”写出来呢？我们只有写出了“目标函数”，才能下一步去“如何优化它”。所以，首先要有目标函数。此处解决的是“有和无”（目标函数）的问题，之后才是“优化”的问题。

2.5 SVM的目标函数

第09课SVM 00:24:00 画图讲解“空间求距离”

在做SVM的时候，不要过多考虑究竟是多少维的问题，讲课的时候，画在纸上，用两维的数据是为了方便，其实它的算法/计算过程是一样的。推导的时候，是用二维来做，但真正做的时候，与N维是没有区别的。

sign函数：符号函数（一般用sign(x)表示）是很有用的一类函数，能够帮助我们在几何画板中实现一些直接实现有困难的构造。符号函数能够把函数的符号析离出来。在数学和计算机运算中，其功能是取某个数的符号（正或负）：当$x > 0$，$sign(x) = 1$;当$x = 0$，$sign(x) = 0$; 当$x < 0$，$sign(x) = -1$。

第09课SVM 00:32:00 很精彩的部分SVM的目标函数：求最小值的最大值，哈哈

2.6 求解SVM的目标函数

2.6.1 输入数据

第09课SVM 00:39:00 输入数据

假设给定一个特征空间上的训练数据集$T = \{(x_1, y_1), (x_2, y_2)...(x_N, y_N)\}$，其中，$x_i \in R^n$，$y_i \in \{+1, -1\}$，$i = 1, 2, ...N$。

x_i为第i个实例（若$n > 1$，则x_i为向量）。

y_i为x_i的类标记：（1）当$y_i = +1$时，称x_i为正例；（2）（1）当$y_i = -1$时，称x_i为负例。

(x_i, y_i)称为样本点。

[一个非常好的问题]为什么y_i的取值是+1和-1呢？

第09课SVM 00:44:00 在做logistic 回归的时候，y一个可以取1，一个可以取0。到了SVM里面呢，y一个取+1，一个取-1。为什么会这样呢？所有教科书中都不会讲为什么。事实上，logistic回归，也可以用+1和-1去推导，是可以做的。在SVM里面用+1和-1是为了方便，仅此而已，方便推导。方便在哪儿了呢？如果y等于+1或-1，那么$y_i \cdot f(x) = \frac{f(x)}{y_i}$，这样的话，就方便我们做推导。

[问题]SVM中实际当中用的多吗？

第09课SVM 00:45:00

[问题]SVM和Spark？SVM的分类效果，真的是好，一般而言，SVM的分类结果优于logistic回归、优于随机森林（RF），但是SVM的计算速度慢。比如，几千个，几万个样本，SVM不做优化的情况下，用到分钟级（时间）

能够把参数学出来，但是随机森林，秒级（时间）就能搞定。SVM往往是一个比较好的分类器。

[问题]SVM如何做多分类呢？

2.6.2 线性可分支持向量机

给定“线性可分训练集”，通过间隔最大化得到的分隔超平面为$y(x) = w^T\Phi(x) + b$，相应的分类决策函数$f(x) = sign(w^T\Phi(x) + b)$，该决策函数称为“线性可分支持向量机”。

其中，$\Phi(x)$是某个确定的特征空间转换函数，它的作用是将x映射到（更高的）维度。最简单直接的：$\Phi(x) = x$。

稍后会看到，求解分离超平面问题可以等价为求解相应的凸二次规划问题。

2.6.3 整理符号

分割平面：$y(x) = w^T\Phi(x) + b$

训练集：$x_1, x_2, \ldots, x_n$

目标值：$y_1, y_2, \ldots, y_n$

新数据的分类：$sign(y(x))$

[问题] 第09课SVM 00:48:00 分隔超平面，哪边是+1，哪边是-1，跟w方向有关系吗？是的。如果位于分隔超平面的法向量的正向，就是+1；位于法向量的负向，就是-1。

2.6.4 推导目标函数

第09课SVM 00:49:00 推导目标函数

2.6.5 最大间隔分离超平面

第09课SVM 00:52:00 最大间隔分离超平面

不要忘记了，虽然目标函数是我们的优化目标，但其实是想通过目标函数求解出其中w和b。

2.6.6 函数间隔和几何间隔

第09课SVM 00:54:00 函数间隔和几何间隔

2.6.7 建立新目标函数

第09课SVM 00:58:00 建立新目标函数

2.7 拉格朗日乘子法

第09课SVM 01:15:00 拉格朗日乘子法

3 卷积神经网络

00:46:23

4 Tensorflow

4.1 安装Tensorflow

conda install tensorflow
pip install tensorflow

5 数学知识

1.LATEX控制序列的概念（类似于函数）
控制序列可以是作为命令：以“\”开头，参数：必须参数和可选参数。

5.0.1 Probability Density Functions

Probability Density Functions https://onlinecourses.science.psu.edu/stat414/node/97

6 Vocabulary

generalized linear model (GLM)

7 论算法之间的区别

各自想法的出处
自己的总结

8 面试

9 其它知识

9.0.2 数据处理——One-Hot Encoding

原文地址：http://blog.csdn.net/google19890102/article/details/44039761

在实际的机器学习的应用任务中，**特征**有时候并不总是连续值，有可能是一些分类值，如性别可分为“male”和“female”。在机器学习任务中，对于这样的特征，通常我们需要对其进行特征数字化，如下面的例子：有如下三个特征属性：

性别：[”male”，”female”]

地区：[”Europe”，”US”，”Asia”]

浏览器：[”Firefox”，”Chrome”，”Safari”，”Internet Explorer”]

不合适的方法。对于某一个样本，如[”male”，”US”，”Internet Explorer”]，我们需要将这个分类值的特征数字化，最直接的方法，我们可以采用序列化的方式：[0,1,3]。但是这样的特征处理并不能直接放入机器学习算法中。

合适的方法。对于上述的问题，性别的属性是二维的，同理，地区是三维的，浏览器则是四维的，这样，我们可以采用One-Hot编码的方式对上述的样本“[”male”，”US”，”Internet Explorer”]”编码，“male”则对应着[1，0]，同理“US”对应着[0，1，0]，“Internet Explorer”对应着[0,0,0,1]。则完整的特征数字化的结果为：[1,0,0,1,0,0,0,0,1]。这样导致的一个结果就是数据会变得非常的稀疏。

实际的Python代码如下：

```
from sklearn import preprocessing
enc = preprocessing.OneHotEncoder()
enc.fit([[0,0,3],[1,1,0],[0,2,1],[1,0,2]])
array = enc.transform([[0,1,3]]).toarray()
print array
```

结果：[[1. 0. 0. 1. 0. 0. 0. 0. 1.]]

9.0.3 什么是张量(tensor)？

什么是张量(tensor)？ https://www.zhihu.com/question/20695804

张量的数学与物理意义是什么，张量的特性与优势是什么？https://www.zhihu.com/question/368149

怎么通俗地理解张量? https://www.zhihu.com/question/23720923

What is a tensor? https://www.quora.com/What-is-a-tensor

小学课本上画杨桃的故事每个人都听过，一个杨桃在不同角度看，就会呈现不同的样子。有些物理量也是一样的，它在不同的角度看就会有不同的数值。比如对于一个矢量，你的基底变化了，矢量的表示也会变化。但是矢量的长度永远不变。杨桃还是那个杨桃，物理量也还是那个物理量，但是一旦你换了个角度看，杨桃的形状就变了，物理量的数值也就变了。那么如果一个物理系统没有一个更好的观察方向，或者说我们需要频繁的变换我们的视角的时候，应该怎么把握一个胡乱变化的东西呢？你要记住，杨桃和物理量本身都是不变的，变的只是它在你眼中的形象。于是张量就出现了，它将视角变换时候的变换关系作为张量的定义，看似在乱七八糟变，实际上只有满足这样的变换关系，它才是不变的！研究一个看似乱七八糟变，实际上不变的东西，就是张量分析。

作者：大野喵渣链接：https://www.zhihu.com/question/36814916/answer/69248640
来源：知乎著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。