斯坦福 ML 公开课笔记 6

本篇笔记针对斯坦福 ML 公开课的第 6 个视频，主要内容包括朴素贝叶斯的多项式事件模型、神经网络、支持向量机。

朴素贝叶斯多项式事件模型

在上篇笔记中，那个最基本的 NB 模型也被成为多元伯努利事件模型（Multivariate Bernoulli Event Model，以下简称 NB-MBEM）。该模型有多种扩展，一种是在上一篇笔记中已经提到的每个分量的多值化，即将 $p(x_i|y)$由伯努利分布扩展到多项式分布；还有一种在上一篇笔记中也已经提到，即将连续变量值离散化。本文将要介绍一种与多元伯努利事件模型有较大区别的 NB 模型，即多项式事件模型（Multinomial Event Model，以下简称 NB-MEM）。

首先，NB-MEM 改变了特征向量的表示方法。在 NB-MBEM 中，特征向量的每个分量代表词典中该 index 上的词语是否在文本中出现过，其取值范围为{0,1}，特征向量的长度为词典的大小。而在 NB-MEM 中，特征向量中的每个分量的值是文本中处于该分量的位置的词语在词典中的索引，其取值范围是{1,2,…,|V|},|V|是词典的大小，特征向量的长度为相应样例文本中词语的数目。

举例来说，在 NB-MBEM 中，一篇文档的特征向量可能如下所示：

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{4321} \\ \vdots \\ x_{50000} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} a \\ aardvark \\ aardwolf \\ \vdots \\ buy \\ \vdots \\ zygmurgy \end{matrix} \tag{1}$$

其在 NB-MEM 中的向量表示则如下所示：

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{21} \\ \vdots \\ x_{123} \end{bmatrix} = \begin{bmatrix} 43000 \\ 35000 \\ 15000 \\ \vdots \\ 1 \\ \vdots \\ 16000 \end{bmatrix} \begin{matrix} the \\ product \\ is \\ \vdots \\ a \\ \vdots \\ it \end{matrix} \tag{2}$$

在 NB-MEM 中，假设文本的生成过程如下：

- 确定文本的类别，比如是否为垃圾文本、是财经类还是教育类等；
- 遍历文本的各个位置，以相同的多项式分布生成各个词语，生成词语时相互独立。

由上面的生成过程可知，NB-MEM 假设文本类别服从多项式分布或伯努利分布，而词典中所有的词语服从多项式分布。生成过程还可如下解释，即先在类别所服从的多项式分布中选取类别，然后遍历整个文本，在词语所服从的多项式分布中选取词语，放到文本中相应的位置上。

于是，NB-MEM 的参数如下所示：

$$\varphi_y = p(y) \tag{3}$$

$$\varphi_{k|y=1} = \mathrm{p}(x_j = \mathrm{k}|\mathrm{y} = 1) \tag{4}$$

$$\varphi_{k|y=0} = \mathrm{p}(x_j = \mathrm{k}|\mathrm{y} = 0) \tag{5}$$

于是，我们可以得到参数在训练集上的极大似然估计：

$$\begin{aligned}\ell(\varphi_y, \varphi_{k|y=1}, \varphi_{k|y=0}) &= \prod_{i=1}^{m} p(x^{(i)}, y^{(i)}) \\ &= \prod_{i=1}^{m}\left(\prod_{j=1}^{n_i} p\left(x_j^{(i)}\middle| y; \varphi_{k|y=1}, \varphi_{k|y=0}\right)\right) p(y^{(i)}; \varphi_y)\end{aligned} \tag{6}$$

极大化似然估计函数，可以得到各个参数的极大似然估计：

$$\varphi_{k|y=1} = \frac{\sum_{\mathrm{i}=1}^{m}\sum_{j=1}^{n_i} I\{x_j^{(i)} = k \wedge y^{(i)} = 1\}}{\sum_{i=1}^{m} I\{y^{(i)} = 1\} n_i} \tag{7}$$

$$\varphi_{k|y=0} = \frac{\sum_{\mathrm{i}=1}^{m}\sum_{j=1}^{n_i} I\{x_j^{(i)} = k \wedge y^{(i)} = 0\}}{\sum_{i=1}^{m} I\{y^{(i)} = 0\} n_i} \tag{8}$$

$$\varphi_y = \frac{\sum_{i=1}^{m} I\{y^{(i)} = 1\}}{m} \tag{9}$$

在$\varphi_{k|y=1}$和$\varphi_{k|y=0}$上使用 Laplace 平滑，得到公式如下：

$$\varphi_{k|y=1} = \frac{\sum_{\mathrm{i}=1}^{m}\sum_{j=1}^{n_i} I\left\{x_j^{(i)} = k \wedge y^{(i)} = 1\right\} + 1}{\sum_{i=1}^{m} I\{y^{(i)} = 1\} n_i + |\mathrm{V}|} \tag{10}$$

$$\varphi_{k|y=0} = \frac{\sum_{\mathrm{i}=1}^{m}\sum_{j=1}^{n_i} I\left\{x_j^{(i)} = k \wedge y^{(i)} = 0\right\} + 1}{\sum_{i=1}^{m} I\{y^{(i)} = 0\} n_i + |V|} \tag{11}$$

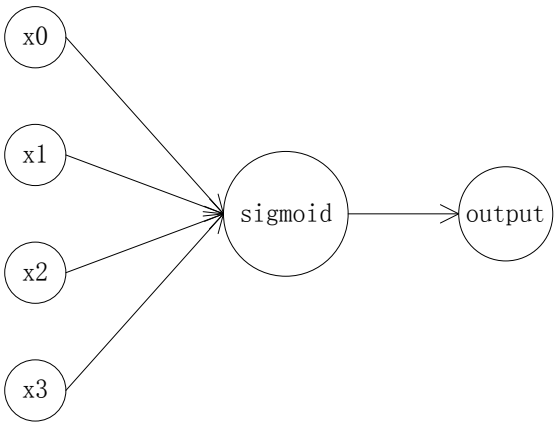
其中，|V|为词典的大小。

神经网络

之前介绍的无论是感知器算法还是逻辑斯蒂回归还是刚才所介绍的朴素贝叶斯模型（朴素贝叶斯算法是前置假设是多项式分布的多项式模型，所以也属于

逻辑斯蒂回归模型），其最终结果反映在数据上都是一条直线或一个超平面，但如果数据并不是线性可分的话，这些模型的性能会变差。针对该问题，涌现出很多对非线性可分数据进行分类的算法，神经网络（neural network）是其中最早出现的一种。

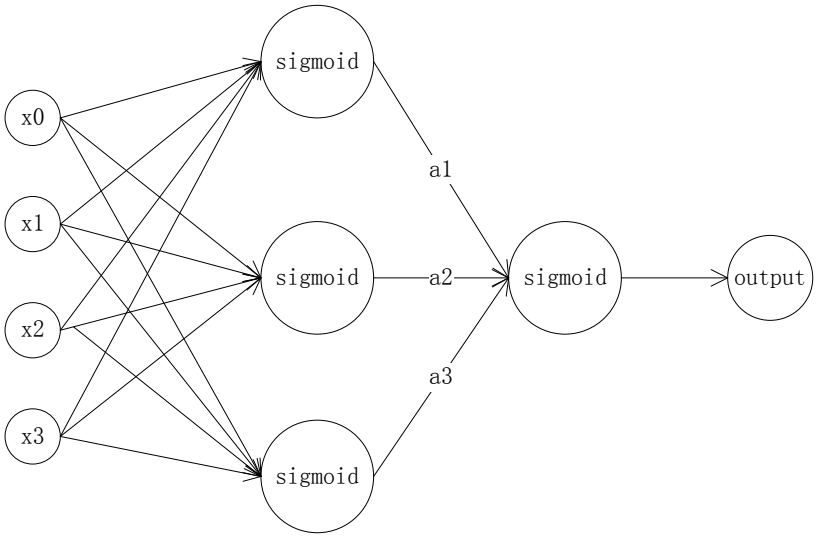
对于逻辑斯蒂回归模型，可以将其表示为下图所示：



其中，x_i 是输入的特征向量的各个分量，sigmoid 是计算单元，output 是函数输出。sigmoid 计算单元有参数θ，其函数形式为：

$$g(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)} \tag{12}$$

而神经网络则是将这样的计算单元组合起来，如下图所示：



其中，a_1，a_2，a_3 是中间单元的输出。可以看到，该图所示的神经网络有四个参数，分别为四个 sigmoid 单元的参数。这些参数之间的关系如下式所述：

$$a_i = g(\theta_i^T x) \quad i = 1,2,3$$

$$h_\theta(x) = g(\theta_4^T \vec{a}) \quad \vec{a} = [a_1 \quad a_2 \quad a_3]^T$$

学习这些参数需要用到成本函数比如：

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(y^{(i)} - h_\theta(x^{(i)}))^2 \quad (13)$$

这是第一次视频里就提到的二次成本函数（quadratic cost function），可以使用梯度下降方法最小化成本函数来求得参数，在神经网络中的梯度下降算法有一个专门的名称叫做反向传播算法（back propagation）。

在上面那个神经网络的样例图中，与输入直接相连的称为隐藏层（hidden layer），与输出直接相连的称为输出层（output layer）。神经网络算法的一大特点就在于不知道隐藏层计算的东西的意义，另一个特点在于神经网络有比较多的局部最优值，可以通过多次随机设定初始值然后运行梯度下降算法获得最优值。

接着，展示了两个神经网络实现的应用的视频。一个是 Hammerton 数字识别应用，对手写的数字进行识别，该应用的作者是 Yann LeCun[1]，他以字符识别与卷积神经网络而著名[2]。另外一个应用则是 NETtalk 神经网络，使用神经网络来阅读文本，作者是 Terry J. Sejnowski[3]。

支持向量机之函数间隔与几何间隔

要理解支持向量机（Support Vector Machine），必须先了解函数间隔（functional margin）与几何间隔（geometric margin）。以下假设数据集是线性可分的。

首先变换一下符号，类别 y 可取值由{0,1}变为{-1,1}，假设函数 g 为：

$$g(z) = \begin{cases} -1 & z < 0 \\ 1 & z \geq 0 \end{cases} \quad (14)$$

而目标函数 h 也由：

$$h_\theta(x) = g(\theta^T x) \quad (15)$$

变为：

$$h_{w,b}(x) = g(w^T x + b) \quad (16)$$

其中，公式 15 中$x, \theta \epsilon \mathbb{R}^{n+1}$，且 x_0=1。而在公式 16 中，$x, w \epsilon \mathbb{R}^n$，b 取代了公式 15 中 x_0 的作用。

[1] http://yann.lecun.com/

[2] http://en.wikipedia.org/wiki/Yann_LeCun

[3] http://en.wikipedia.org/wiki/Terry_Sejnowski

由公式 16，我们得知，w,b 可以唯一的确定一个超平面。

一个点$(x^{(i)}, y^{(i)})$到由 w,b 决定的超平面的函数间隔是：

$$\hat{\ell}^{(i)} = y^{(i)}(w^T x^{(i)} + b) \tag{17}$$

超平面与整个训练集合的函数间距是：

$$\hat{\ell} = \min_i \hat{\ell}^{(i)} \tag{18}$$

公式 17 还有一个性质，即对于正确分类的数据点，函数间隔不小于 0。

函数间隔的问题在于只要成倍增大 w,b，就能是函数间隔变大。为了解决这个问题，就有了几何间隔的定义，几何间隔定义如下：

$$\max_{w,b} \ell \qquad s.t.\ y^{(i)}\left(w^T x^{(i)} + b\right) \geq \ell \quad 且 ||w|| = 1$$

即在||w||=1 条件下函数间隔最小值。

几何间隔与函数间隔的意义在于为根据训练集合得到的模型增添了一个指标，使得模型不仅保证分类结果的正确性，更要进一步保证分类结果的确定性。