Discriminative Modeling vs Generative Modeling

http://freemind.pluskid.org/machine-learning/discriminative-modeling-vs-generative-modeling

posted on Free Mind on June 9, 2012
generated with pandoc on December 3, 2015
category: Machine Learning

tags: Generative Modeling, Learning Models

机器学习的算法或者模型的分类有很多种，其中有一种分法把模型分为 Discriminative Modeling（判别模型）和 Generative Modeling（生成模型）两种。为了写起来方便我们以下简称 DM 和 GM 。在一个基本的机器学习问题中，我们通常有输入 $x \in \mathcal{X}$ 和输出 $y \in \mathcal{Y}$ 两个部分，简单地来说，DM 关注于 x 和 y 的关系，或者说在给定某个 x 的情况下所对应的 y 应该满足的规律或分布；而 GM 则试图描述 x 和 y 的联合分布。两种视角各有各的优缺点，本文我们就试图来对两种建模方式做一些基本的了解。

我们不妨从具体的算法出发。记得我最初接触机器学习相关的东西的时候，做的是一个垃圾邮件分类的项目，当时我们在做的就是尝试各种不同的算法，其中有两种典型的算法 Naive Bayes 和 Logistic Regression 实际上就分别是 GM 和 DM 的两种最基本的算法。所以我们在这里也不妨用垃圾邮件分类这个场景来了解一下 DM 和 GM 。

垃圾邮件分类如今可算是一个非常古老的问题了，从 Gmail 的自动垃圾过滤来看，目前的 state-of-the-art 已经可以非常成熟了：precision 上来说，我从 2005 年到 2012 年这段时间内被误判为垃圾邮件的好邮件大概一共四、五封吧（当然可能还会有一些我压根就没有看到的被误杀掉的邮件）；recall 上则稍微逊色一点，偶尔会在收件箱里出现 spam 。当然 recall 比 precision 略差也情有可原：

- precision 和 recall 之间是一个 trade-off ，对于垃圾邮件分类来说，将正常邮件标记为垃圾邮件和将垃圾邮件认为是正常邮件相比，前者是严重得多的错误，所以一般会倾向于调高 precision 。
- spammer 在不断地进化，针对 anti-spam 工具设计新的 spam 。

但总的来说因该也算是机器学习在实际中应用最成熟的领域之一了。当然好的应用算法、数据、工程各个方面都是非常重要的，这里我们只讲一些基本模型，毕竟不能跑题了呀！ ^_^ 一般机器学习算法在处理文本文档的时候多用 Vector Space Model (VSM) 来把一个文档转化为一个定长的向量进行处理。VSM 需要先构建一个词典，用 $f^1, \ldots, f^m$ 表示词典中的 m 个单词（这里 f 是 feature 的缩写），我们用 $x_1, \ldots, x_n$ 表示 n 个训练数据（文档）。

VSM 将每个文档转化为一个 m 维的向量，每个维度对应一个单词（feature）。最简单的 VSM 只判断单词是否在文档中出现，例如，我们用 $f^j(x_i)$ 表示文档 x_i 所对应的向量的第 j 维，那么

$$f^j(x_i) = \begin{cases} 1 & f^j \in x_i \\ 0 & \text{otherwise} \end{cases}$$

这种模型有一个好处是它产生出来的向量是 binary 的：每个维度上只有 0 或者 1 出现，因此处理起来可能会更简单一些。更复杂一点的 VSM 可能会将 $f^j(x_i)$ 设为单词 f^j 在文档 x_i 中出现的次数或者 TF-IDF 值。当然我们也可以跳出词典的限制，此时 f^j 并不一定对应某个单词，而表示任意一个 feature，例如，当 x_i 的发件人在我的通讯录里的时候，$f^j(x_i) = 1$，否则等于 0。

Naive Bayes 这个算法可以从它的名字来进行解读。首先它是根据 Bayes 公式来进行分类的。令 X 和 Y 分别代表 “文档” 和 “是否是 spam” 的随机变量（$Y = 1$ 表示是 spam）。这里我们对 $P(Y)$ 和 $P(X|Y)$ 这两个概率模型进行建模，从而可以算出

$$P(X, Y) = P(X|Y)P(Y)$$

在对文档 x 行分类的时候我们分别计算

$$P(Y = 1|X = x) = \frac{P(Y = 1, X = x)}{P(X = x)} = \frac{P(X = x|Y = 1)P(Y = 1)}{P(X = x)}$$
$$P(Y = 0|X = x) = \frac{P(Y = 0, X = x)}{P(X = x)} = \frac{P(X = x|Y = 0)P(Y = 0)}{P(X = x)}$$

然后比较两个概率值，简单的办法是看哪个大就分为哪一类，复杂一点的情况可以在两个概率值相差不大的时候输出 “不知道” `-.-bb`。注意上式中红色部分两个式子是一样的，由于我们只是要比较两个值的相对大小，因此一样的部分可以直接省略，从而我们其实只是在比较 $P(Y = 1, X = x)$ 和 $P(Y = 0, X = x)$ 这两个概率值，这是可以通过模型直接计算出来的。

接下来看两个模型，首先是 $P(Y)$，由于 Y 只取 0 和 1 两个值，因此用一个 Bernoulli Distribution 即可描述，该模型只有一个参数 $p = P(Y = 1)$。而 $P(X|Y)$ 则分别是两个模型 $p_0(X) = P(X|Y = 0)$ 和 $p_1(X) = P(X|Y = 1)$，每一个模型又是所有特征的联合概率分布，例如：

$$p_0(X) = p_0(f^1 = f^1(X), \ldots, f^m = f^m(X))$$

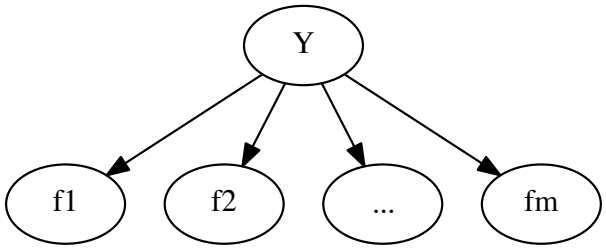
这里 “Naive” 出现了，由于 m 一般比较大，这么多特征的联合概率分布建模和处理起来都非常费力，于是这里做了一个非常 Naive 的假设，就是所有的特征之间是相互独立的（更确切地说，是相对于 Y 条件独立

的)，根据独立性的定义，这里的联合概率可以写为每个特征各自的概率的乘积：

$$p_0(X) = \prod_{j=1}^{m} p_0(f^j = f^j(X)) \tag{1}$$

用概率图模型可以表示成图 1 中的样子。

Figure 1: .....



注意这是非常强的一个假设（否则也不会被命了 “Naive” 这样的名字吧？:D)，例如如果每个 feature 表示单词的话，就要求一封邮件中各个单词出现与否是相互独立的，这明显是不符合实际情况的，不过 Naive Bayes 模型视乎在垃圾邮件分类问题中还是效果挺不错的。这样一来，对一个联合概率分布进行建模就简化为对每个 feature f^j 进行建模。最简单的情况，如果特征是 binary 的话，我们仍然可以和 Y 一样用 Bernoulli Distribution 。如果用了 TF-IDF 之类的连续特征的话，则可以用高斯分布之类的进行建模。在本文中我们只考虑 binary feature 的情况。

接下来我们来看 Logistic Regression ，这个算法直接对 $P(Y|X)$ 进行建模，两类的模型如下：

$$P(Y=1|X) = \frac{\exp(\beta^0 + \sum_{j=1}^{m} \beta^j f^j(X))}{1 + \exp(\beta^0 + \sum_{j=1}^{m} \beta^j f^j(X))}$$
$$P(Y=0|X) = \frac{1}{1 + \exp(\beta^0 + \sum_{j=1}^{m} \beta^j f^j(X))}$$

很容易验证这两项相加等于 1 ，并且

$$\log\left(\frac{P(Y=1|X)}{P(Y=0|X)}\right) = \beta^0 + \sum_{j=1}^{m} \beta^j f^j(X)$$

换句话说，分类边界实际上是一个线性函数，这也是 Logistic Regression 模型设计的 motivation 。这里就可以先做一个简单的对比：Naive

Bayes 需要同时对 X 和 Y 进行建模，得到联合分布 $P(X,Y)$ ，因此是生成模型，由于 X 是个比较复杂的东西，建模起来很痛苦，于是 Naive Bayes 不得不做了很强的假设，从此一辈子戴上了 “Naive” 的帽子。而 Logistic Regression 作为一个典型的判别模型，则直接对 $P(Y|X)$ 进行建模，而不关心 $P(X)$ ，从而免去了许多麻烦和纠结，也不需要做那么大的取舍。

不过话虽如此，但是 Logistic Regression 也做了另一个很强的假设就是 linear model，接下来我们将看到实际上 Logistic Regression 是和 Naive Bayes 同流合污的。我们暂时回到 Naive Bayes ，由于它是 GM ，对 $P(X,Y)$ 进行了建模，那我们来看看把它转化为一个 DM 会怎么样呢？也就是说，在我们已有的 $P(X,Y)$ 模型下，去求 $P(Y|X)$ ，或者，为了和 Logistic Regression 那里进行对比，我们直接去求

$$\begin{aligned}\log\left(\frac{P(Y=1|X)}{P(Y=0|X)}\right) &= \log\left(\frac{P(X|Y=1)P(Y=1)}{P(X|Y=0)P(Y=0)}\right) \\ &= \log p_1(X) + \log p - \log p_0(X) - \log(1-p)\end{aligned} \tag{2}$$

此外，我们还可以对式 (1) 进行一下简单的变形：

$$\begin{aligned}p_0(X) &= \exp\log p_0(X) \\ &= \exp\left(\sum_{j=1}^{m}\log p_0(f^j = f^j(x))\right) \\ &= \exp\left(\sum_{j=1}^{m}\left(I_{f^j(x)=1}\log p_0(f^j=1) + (1-I_{f^j(x)=1})\log p_0(f^j=0)\right)\right) \\ &= \exp\left(\sum_{j=1}^{m}\left(f^j(x)\log p_0(f^j=1) + (1-f^j(x))\log p_0(f^j=0)\right)\right) \\ &= \exp\left(\sum_{j=1}^{m}\log p_0(f^j=0) + \sum_{j=1}^{m} f^j(x)\log\frac{p_0(f^j=1)}{p_0(f^j=0)}\right)\end{aligned}$$

这里 $I_{f^j(x)=1}$ 是 indicator function ，当下标中的条件满足时等于一，否则等于零。由于 f^j 本身就是 binary feature ，所以在倒数第二个等式中我们用了 $f^j(x) = I_{f^j(x)=1}$ 来进行化简。对 p_1 可以做类似的变形，于是式 (2) 可以进一步化为

$$\begin{aligned}\log\left(\frac{P(Y=1|X)}{P(Y=0|X)}\right) = &\sum_{j=1}^{m}\log p_1(f^j=0) + \sum_{j=1}^{m} f^j(X)\log\frac{p_1(f^j=1)}{p_1(f^j=0)} \\ &+ \log p - \log(1-p) \\ &- \sum_{j=1}^{m}\log p_0(f^j=0) - \sum_{j=1}^{m} f^j(X)\log\frac{p_0(f^j=1)}{p_0(f^j=0)}\end{aligned}$$

式子看起来有点复杂，不过如果我们把所有的常数项合并之后记为 β^0 ，而把 $f^j(X)$ 的系数合并之后记为 β^j 的话，就可以看到这个模型和 Logistic Regression 是一模一样的。像 Naive Bayes 和 Logistic Regression 这样的通常被称为 Generative-Discriminative Pair ，类似的还有 Hidden Markov Model 和 Linear-chain Conditional Random Field 等。

一般来说，DM 比 GM 看起来更诱人一些，特别是我们的目标就是分类的时候，DM 直接建模 $P(Y|X)$ 进行分类，而 GM 则先建模 $P(X,Y)$ 然后再通过该模型计算 $P(Y|X)$ 进行分类。

首先 $P(X,Y)$ 的估计本身会很困难，需要的计算量和训练数据量都会非常巨大，像 Naive Bayes 那样为了使得模型复杂度被控制在可以处理的范围内就不得不做了非常强的假设。值得注意的是，虽然 Naive Bayes 的模型从某种程度上来说是相互对应的，但是两者实际上是差别很大的算法，比如如果 X 严重违背了独立性假设的话，Naive Bayes 的性能有可能会受到严重影响，而 Logistic Regression 则不太会受到 X 上的相关性之类的限制。比如，在自然语言处理中分析句子的语法树的时候，为了提高性能通常会使用一些冗余度和相关性非常高的特征，这个时候如果使用独立性假设很强的 GM 来处理，效果就不好，但如果不做很强的独立性假设，模型的复杂度又会指数级别增长，很快就无法处理了。不过用对等的 DM Conditional Random Field 来做的话，就可以不用 care 特征之间的关联与否，而有效地利用 rich feature set 。

其次 GM 实际上分两步来解决问题，多余的迂回步骤有可能会积累更多的误差。关于这个问题 Vapnik 有一句比较有名的话就是说的：

> *One should solve the problem directly and never solve a more general problem as an intermediate step.*
>
> —V.N. Vapnik. Stastical Learning Theory. John Wiley & Sons, 1998.

Andrew Ng 和 Michael Jordan 在 2001 年的一篇 NIPS 论文 [Ng and Jordan, 2001] 中分析过，对于一对 generative-discriminative pair 来说，DM 有比 GM 更低的 asymptote error。不过 GM 也并不是一无是处，在同一篇论文中，他们也给出了另一个结论，就是 GM 能比对应的 DM 更快地达到 asymptote error ，也就是说虽然最终 DM 会 outperform GM ，但是在 DM 还未达到它的 asymptote error 之前，GM 可能先达到了自己的 asymptote error，从而在一段时间内反而 outperform DM。

虽然 asymptote error 之类的可能在实际中并不一定很有参考价值，不过 GM 在实用上也还有另一个优点：那就是处理 latent variable 或者 partially observed data 以及 partially labeled data 之类的：因为它同时对 X 和 Y 都建立了概率模型，所以如果 X 中有出现没有观察到的量的时候，就可以很自然地使用 EM 框架 来进行处理。极端情况下，在完全没有 label 信息的情况下，GM 仍然是可以工作的——unsupervised learning 可以看成是 GM 的一种。

总而言之，DM 和 GM 之间的争论也不是一天两天的事情了，虽然好

像看起来 DM 更靠谱一些，但是 GM 也有它自己的优势，并且在具体的实际问题中 GM 和 DM 哪个效果更好也并不是能一下子 “看” 出来的。所以大概两者之间的争斗还会不断持续下去吧——或者说称为是和平共处？:P

References

[Ng and Jordan, 2001] Ng, A. Y. and Jordan, M. I. (2001). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *NIPS*, pages 841–848.