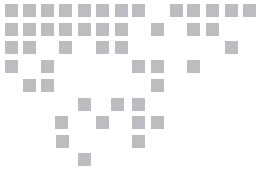

Chapter 2 第 2 章

实时监督式机器学习

2.1 什么是监督式机器学习

监督式机器学习旨在利用训练集数据，建立因变量和自变量之间的函数映射关系。如果用 X 代表自变量，Y 代表因变量，f 代表映射函数，b 代表映射函数的参数，那么监督式机器学习的任务就是找到恰当的函数 f 和参数 ，让下面的映射尽量符合要求：

$$y=f(x;b,e)$$

这里 e 为实际情况中的随机扰动项。

下面就来具体看看在监督式机器学习中，因变量、自变量和预测函数的含义。

（1）因变量

因变量是我们试图通过机器学习模型预测的变量，在实际应用中它往往无法在预测之时就能观测到。例如在实时股价波动方向预测中，未来股价的走向就是一个因变量，只有等待时间流逝之后才能得知。我们进行预测时只能根据当前已有的历史交易数据、基本面信息等进行判断。这个时候就需要利用已有的变量对这一因变量进行预测。对于不同的应用场景，因变量可以是金额、收入等连续变量，也可以是性别、状态等离散变量，还可以是三围、经纬度等多维变量。

（2）自变量

自变量是我们在预测时就已经获得的，可以用于因变量预测的数据。例如在实时股价走势预测的例子中，历史走势、历史成交量等数据都是在进行预测的时候就能够获得的数据，通过经验，我们还知道历史走势和未来走势可能具有一定的函数关系，于是历史走势就成了预测未来走势的自变量。在实际运用中，自变量往往具有实时、廉价的特点。我们

通过当前已有的、可以廉价获取的自变量数据，来预测更难观测到、更为珍贵的因变量数据，其实也是一种低买高卖的投资。同样，自变量可以是连续、离散、多维的数据，甚至是图片、文字等多媒体数据的集合。

（3）预测函数

预测函数是我们进行监督式机器学习的核心。理想的预测函数能够按照需求，将因变量映射到自变量空间。例如在实时股价走势预测这一应用场景中，我们可以采用线性函数等多种函数将历史数据映射到未来走势函数的空间中去。预测函数往往多种多样，其既可能是线性、树状、网格状函数，也可能是 Murmur Hash 等非线性二进制处理函数，还有可能是前面提到的这些函数的组合和叠加。

那么，什么样的机器学习模型才是适用于实际生产场景的呢？笔者根据自己的工作经验总结了以下几点。

- **低成本模型：**对于腾讯、谷歌、阿里巴巴等航母型企业以外的用户，推荐采用尽量低成本的模型。这里的低成本体现在两个方面：软件包尽量用现成的，且对软硬件的要求要尽量低。
- **模型易于解释：**在实际应用中，我们往往需要对模型产生的结果进行解释和排错。如果模型太过复杂，难以排错，那么势必会影响到实际应用。
- **模型易于修改：**建立的机器学习模型往往需要对未发生的事情进行预测。这个时候需要将人为判断放入模型中，这就要求机器学习模型应该能够很容易地带入人工设置的参数。

虽然近十年出现了成千上万的机器学习方法，其中不乏发表于顶级刊物上的名家大作，但是如果按照上面三条逐个进行检查，那么完全符合要求的机器学习方法也就所剩不多了。

例如不少读者都听说过深度学习这一众所周知的方法，可是仔细研究就会发现，深度学习的训练往往需要 GPU 硬件的支持，而且深度神经网络由于其过于复杂，几乎无法进行解释排错。所以深度神经网络的应用往往也局限在图像、自然语言处理等特殊场景下，对于风险分析等重要的应用场合完全无法涉足。我们在本书末尾将介绍深度学习的平台选择，供有兴趣的读者参考。

笔者之一的彭河森，其博士论文是以高维非线性机器学习为主题，但是经过多年的观察总结，发现还是线性模型和朴素贝叶斯模型在实际应用中最能满足上面这三点。因此本书也将着重讲述线性模型在实时机器学习中的应用。

2.1.1 “江湖门派”对预测模型的不同看法

具有深厚的技术功底只是能在工业界生存的一半因素，另外一半取决于门派站队是否正确。机器学习和统计中一直存在不同的门派，这些门派就像江湖中的各种高手一样，都有着自己的独门绝技，很多文献和教程往往只会注重一方面门派，而忽略了很多其他门派的贡献。尽信书不如无书，我们鼓励大家多阅读、多思考，在实际应用中形成自己的知识

架构体系。

大家可能注意到了，第 1 章对机器学习的定义其实很模糊，比如：里面的模型参数应该是随机的还是实际固定存在的？ 是真的随机还是只是我们没有观测到而已？这些我们并没有给出具体的回答。这是因为不同的门派对它的意义和看法都各不一样。下面就来看看都有哪些观点。

在统计和计算机理论领域，势力非常强大的贝叶斯学派认为参数是一个随机变量，因为我们每次对因变量、自变量进行观测之时，看到的都是一个不可观测到的随机变量产生作用的侧影。与此相对应的，稍微低调一点的频率学派则认为参数是客观存在的固定数值，只是因为随机扰动的存在导致我们无法进行精确观测。

大家不要小看上面看似微小的观点差别，这种基本世界观的差距，会直接影响一个企业的架构设计。对我们个人来说，最直接的影响就是不同门派的人往往会各自为政，互不往来。如果没弄清楚情况，频率学派的人去以贝叶斯学派为主的公司面试，往往会碰一鼻子灰，公司内部政治斗争往往也会按照这样的门派站队。所以及时认清派系也是我们核心技术人员生存的重要本领。

说个最直观的例子，LinkedIn 的一些机器学习高管是贝叶斯学派的忠实信徒，自然，LinkedIn 领导开发的几个项目也都是基于贝叶斯理论的模型，如果不是贝叶斯模型，那么先要将其贝叶斯化，才能得到首肯。而贝叶斯模型最擅长的当然是线性模型，所以致使 LinkedIn 几乎所有机器学习模型都是线性的。贝叶斯模型并没有什么不好，只是，如果倾全公司之力，只做贝叶斯模型，就限制了深度学习等新工具的加入，错过了进步的机会，会比较可惜。

2.1.2 工业界的学术门派

以笔者在工业界摸爬滚打多年积累的经验来看，工业界的门派主要涉及了以下三个方向。

1. “重造轮子”学派

“重造轮子（reinvent the wheel）”学派拥有非常强大的势力，“重造轮子”学派的人员往往都像是在世外山谷中修炼了千年的高人，其见地都会让人眼前一亮，他们大都具有独当一面的技术能力。通常实力越是强大的公司，越有可能存在“重复造轮子”的情况，比如，阿里巴巴、百度。

阿里巴巴有一个很有意思的例子。阿里巴巴某些业务组的技术能力非常出众，在 Hadoop 生态刚刚兴起的时候，其开发人员不满于 Hadoop 的运行效率，自己用 C 语言开发了一套自有分布式机器学习平台。后来 Spark 和 MLLib 的出现直接解决了 Hadoop 运行效率的问题，笔者也很好奇他们的那套系统后来怎么样了。

百度也存在“重造轮子”的情况，深度学习盛行的时候，百度发布了自己的深度学习

开源框架 PaddlePaddle，但两位笔者至今没有认识任何非百度的人士在任何场景中测试或应用了该框架。

2. “参数调教”学派

“参数调教”门派比较类似于高位于庙堂之上的士大夫，具有出神入化的理论功底，但是作品却很难接地气，微软、雅虎[⊖]都有这样的例子。

在第一次互联网泡沫达到鼎盛的时候，微软、雅虎等公司都成立了自己的研究部门，如微软研究院、雅虎研究院等。这些部门的研究人员不需要像学术界的同僚那样寻找资金项目支持，只需要专注发表文章。就和科举一样，发文章的竞争也是非常激烈的，为了得到最好的结果，往往需要进行大量的参数调教。从微软、雅虎等公司出身的研究员，一般都特别热衷于支持向量机（SVM）和深度神经网络（Deep Neural Network，DNN）两个方法，因为这两个方法特别适合“参数调教”，且往往能调出特别好看的结果，以利于文章的发表。当然过度的“参数调教”带来的后果就是模型过度拟合，纸面上非常惊艳的模型拿到实际应用中可能只会是一筹莫展。现在雅虎被 Verizon 低价收购，微软研究院现在也在向业务组转型。

3. “拳打脚踢”学派

“拳打脚踢”门派更像是江湖中的丐帮，看似灰头土脸，但是非常接地气，得到的结果往往也是非常好的，这一派的带头人首推亚马逊。

笔者入职亚马逊的时候，亚马逊并没有专门的机器学习研究部门，当时，所有的研究员都分散在业务组，业务的快速驱动使得机器学习相关人员都适应了快速上线、抓主干的开发模式，这样的环境使得亚马逊的机器学习迭代非常快，而从来不会在参数调教上面浪费精力。

另外一方面，由于亚马逊非常讲究以客户为中心，出现机器学习模型失败的应用时，以 CEO 杰夫 · 贝索斯为首的领导团队会向下施压寻求解答。机器学习界有很多优秀的模型，如深度神经网络等复杂模型，它们虽然效果很好，但其解释和排错的难度也是非常大的。这样对解释、排错的需求使得线性模型、决策树、随机森林等简单模型在亚马逊得到了非常广泛的应用。为了方便这样的用途，亚马逊内部还开发了多项可视化工具，方便人工排错。

不同的公司具有不同文化，最后做出来的机器学习产品也是不同的。介绍上面这些例子，是希望大家在设计实时机器学习系统的时候能够将公司的文化环境、需求考虑在内，这样设计出来的系统才是好用的、受欢迎的系统。

2.1.3 实时机器学习实战的思路

本书的两位笔者都曾经供职于亚马逊和微软，横跨拳打脚踢和参数调教两大门派，对

⊖ 现在雅虎已经被 Verizon 收购，改名 Altaba，雅虎已经成为历史。

于实际的机器学习方法，我们是这么认为的。

不要重复造轮子：重复造轮子听起来是很酷。很多人喜欢吹毛求疵说 C 效率高，Java、Python 不行，碰到这种情况，笔者往往会反问，Fortran 更快，你为什么不用 Fortran 写？重复开发对开发人员的个人生活来说是非常不合适的。既然已经有了现成的工具，那么为什么不早点完成任务下班回家陪家人呢？现今主流开源软件的运行效率都还不错，只是在特殊情况下需要按照业务需求对软件运行环境进行配置，例如调节 Java Heap size、配置线程数量等。当然，这些操作听起来并不是那么酷，但是可以让你早点下班锻炼身体，为祖国健康工作五十年。

没有模型是完美的：教科书里的机器学习数据都假设用户在同一个宏观环境中进行操作，但是实际应用中并不是这样的。在实际应用中，任何模型的效果都会随着时间的流逝而衰减；很多曾经成立的模型假设，也会随着时间的推移变为无效，而新生态的出现需要我们不断去抓取新数据。这就要求我们不断地去更新模型，不断地审视自己的模型和假设，不断做出改进。

重视上下游生态：实时机器学习系统往往只是一个大组织中小小的一环。取得优秀的成绩固然很重要，但是实时机器学习中产生的数据、知识也可能是整个组织不可或缺的财富。采取一些简单但是易于解读的模型，往往有利于组织进行分析学习，从组织层面上达到新的高度。

2.2 怎样衡量监督式机器学习模型

本章前面对一个好的实时机器学习模型的衡量只提到了“优秀”“合适”这样的字眼，本节将会详细展开，讨论监督式实时机器学习模型的衡量标准。

在实际应用中，监督式实时机器学习效果的好坏可以分为统计量是否优秀和应用业绩是否优秀两个方面。下面将按照这两部分分别进行介绍。

在讨论技术细节之前，先进行一下符号的定义：

给定 n 组已知的自变量和因变量 $\{(Y_i, X_i)\}_{i=1}^{n}$ 作为测试数据集，对于任意 i，我们通过自变量 X_i 和模型 $f(X_i; b)$ 预测自变量的数值，得到对因变量的估计 $\hat{Y}_i = f(X_i; \hat{b})$。

本节下面的所有内容都与讨论 Y_i 和 $\hat{Y}_i$ 的近似程度相关。

2.2.1 统计量的优秀

一个监督式机器学习模型若取得了优秀的统计量成绩，则代表着其预测或分类的误差较小，精确度上比较优秀。对于分类和回归预测这两个问题，我们将定义不同的统计量。这类统计量在现有机器学习软件包中往往具有完备的函数支持，例如 Scikit-learn 的 sklearn.metrics 模块中就含有数十种从统计量角度衡量模型优劣的函数。这里我们选取最常用的几种进行介绍。

1. 衡量回归预测的统计量

在回归、预测等场景中，因变量 Y 往往为连续变量。例如，我们可能会通过父母的身高预测子女成年后的身高，也可能通过社交舆情数据预测当日股票收盘时期的涨跌幅。这里的身高、涨跌幅都是连续变量，我们对其的预测值需要尽量接近真实观测值。为了达到这样的目的，常用的统计量有以下几种。

（1）均方误差

均方误差（Mean Square Error，MSE）是统计中最常见的误差衡量单位之一，其定义为：

$$\mathrm{MSE}=\frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i-Y_i)^2$$

在数学上，均方误差的估计可以追溯到正态分布方差的无偏估计。就算 Y_i 实际上不服从正态分布，均方误差仍然具有优良的统计性质。直观上来讲，我们希望通过机器学习模型所得预测的均方误差应尽量小。用 $E(\ )$ 代表对随机变量数学期望的计算，可以将其中一个观测的均方误差分解为两部分：

$$E(\hat{Y}-Y)^2=\mathrm{Var}(\hat{Y})+\mathrm{Bias}(\hat{Y})^2$$

这里的均方误差可以看作是 $E(\hat{Y}-Y)^2$ 的估计量，等式右边部分可以分为如下两部分来解读。

估计的方差 估计的方差（variance）刻画的是对因变量预测的变化程度。真实世界里，任何观测和度量都具有随机性，这样的随机性决定了我们对自变量的预测也具有客观存在的随机性。这样的随机性随着机器学习模型估计方法的不同可能会有所不同。

估计的系统性偏差 当我们的估计系统性地偏离真实数值的时候，系统性偏差 (bias) 就会被包含在均方误差中。在理论情况下，如果我们使用了无偏估计，系统性偏差为零，这时均方误差就只与方差有关。当然，在实际应用中，我们的模型或多或少都会有一定的系统性偏差，理想情况就比较难以达到了。

比较上面这两点的异同是所有数据科学家面试题目中的必考部分。为了便于大家理解，这里以图 2-1 作为例子进行对比。图 2-1 对比了具有完全相同均方误差的两组数据的估计值和真实值。图 2-1a 为无偏估计，但是估计方差较大；图 2-1b 的估计方差较小，但是估计有偏。当然，其实也是可以分别用方差和偏离程度来考量估计的优劣的。但是当我们具有多个统计量的时候，就往往需要通过实际情况进行取舍了。有的时候我们宁愿牺牲无偏估计，以换取估计的稳定性；有的时候我们又需要不顾一切地保证估计的无偏性。

（2）绝对误差中位数

在实际应用中我们往往会遇到极端值（outlier）。例如通过父母身高预测小孩身高的时候混入了姚明的身高，通过浏览行为预测网购金额的时候混入了王思聪的购买信息。这个时候由于极端数值的存在，均方误差的计算会大受影响，从而致使我们得到的模型评价的结论也并不贴近实际。

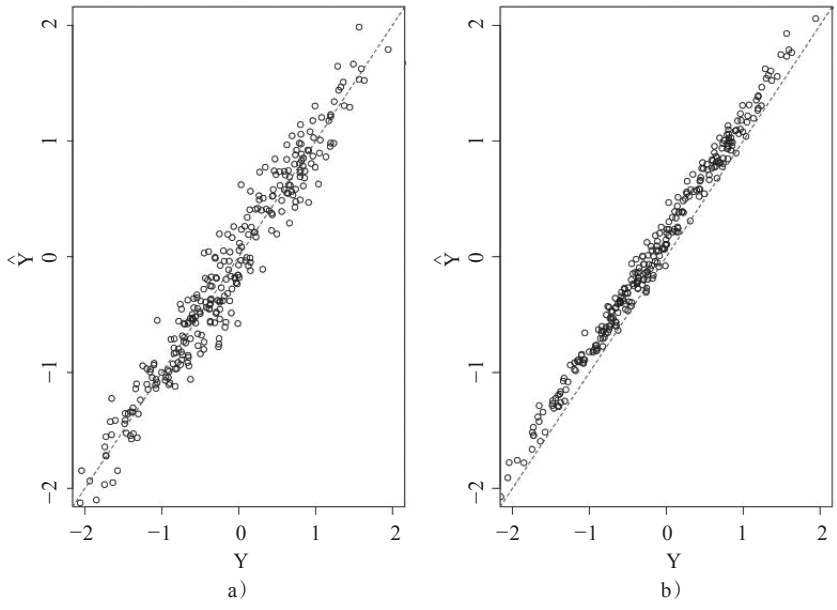

图 2-1 估计值和真实值的对比，两组数据具有相同的均方误差

为了解决这一问题，统计学家们引入了稳健统计量，提出了绝对误差中位数（MAE）的概念。绝对误差中位数的定义为：

$$\text{MAE} = \text{Median}(|\hat{Y}_i - Y_i|)$$

这里不再采用所有误差的均值，而是使用误差绝对值的中位数作为统计量，大大减少了极端观测对最终判断的影响。

图 2-2 中对比了存在极端值（见图 2-1a）和不存在极端值（见图 2-1b）的分布。图 2-1a 和图 2-1b 都有 300 个观测点，其中图 2-1a 具有 20 个随机选取的异常点。在不考虑极端观测的情况下，图 2-1a 和图 2-1b 的分布是完全相同的。如果使用均方误差进行效果衡量，那么图 2-1a 为 0.298，图 2-1b 为 0.043，图 2-1b 明显优于图 2-1a；如果用绝对误差中位数进行衡量，那么图 2-1a 为 0.159，图 2-1b 为 0.136，只是略微优于图 2-1a。

根据实际应用的经验，极端数值往往是客观存在的，因此，建议读者在进行评价的时候应尽量采用稳健统计量绝对误差中位数。

2. 衡量分类的统计量

在分类等任务中，因变量 Y 往往是离散变量。例如我们可能会通过用户的浏览行为预测点击具体页面的概率，这里最后得到的标签实际上是点击或不点击，是一个离散变量。也可能通过文字对话判断参与用户的性别，这里用户的性别往往也是离散变量。对于这样的分类问题，特别是分为两类的问题，我们往往会对实际标签和预测值进行分类，让其定

义为阳性（例如点击、男性）和阴性（例如不点击、女性），于是我们可以得到表 2-1 所示的内容。

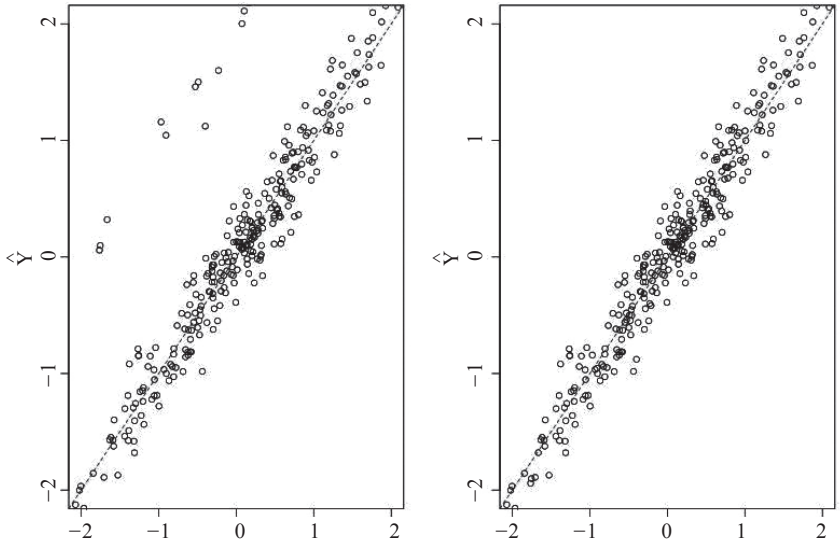

图 2-2 存在极端值（图 2-1a）和不存在极端值（图 2-1b）的统计量对比

表 2-1 预测标签和实际标签对比

	实际类别	
预测类别	阳性	阴性
阳性	真阳性	假阳性
阴性	假阴性	真阴性

统计学家根据表 2-1 定义了数十个统计量，本节将介绍最常见的两个统计量，即准确率和召回率。

（1）准确率 (precision)

准确率是指在被机器学习判断为阳性的观测中，真阳性观测所占的比例：

$$准确率=\frac{真阳性}{真阳性+假阳性}$$

准确率刻画的是喊“狼来了”的孩子有多少次喊狼来了的时候是正确的。例如，在实时股票走势预测的场景中，我们假设股价上涨是阳性观测，股价下跌是阴性观测。在通过机器学习模型对其进行分类预测时，准确率的定义就是被预测的走势中，被预测为会上涨的这些观测点中，实际上真正上涨的观测点所占的比例。

（2）召回率 (recall)

召回率是指在真实的阳性观测中，被判断为阳性的观测所占的比例：

$$召回率=\frac{真阳性}{真阳性+假阴性}$$

召回率刻画的是在所有狼来了的历史里面，有多少次牧羊小孩成功地发现了狼。例如，在实时股票走势预测的场景中，我们假设股价上涨是阳性观测，股价下跌是阴性观测。在通过机器学习模型对其进行分类预测时，召回率的定义就是，对于所有实际上涨的这些观测点中，被预测为可能会上涨的观测点所占的比例。

2.2.2 应用业绩的优秀

在回归预测的任务中，误差对业务产生的影响往往是不一样的。例如，想要通过建模预测航班售票的情况，若我们预测的乘客数量比实际超出太多，则可能会造成机场安排过多运力，造成浪费；但是当我们预测的乘客数量过少，又会造成超额售票，机场运力不足，这就会对乘客的体验造成影响。这个时候对机器学习模型优劣的判断就需要将不对称的收益考虑进去。

同样，在分类任务中，准确率和召回率是相互竞争的两个统计量。例如，我们如果奉行宁可错杀一百，不可放过一个的思想，将所有股价走势情况都预测为上涨，那么这样我们可以达到 100% 的召回率，但是准确率会变得很低。与此相对，若将所有观测都预测为下跌，这样我们可以达到 100% 的准确率，但是召回率又将变得非常低。所以，真正应用在实际之中时，我们往往需要对相互竞争的统计量进行权衡，选一个合适的中间点作为最终判断的准绳。

例如，在股价走势预测建模数据中，我们最后的评判标准可能是：

$$S=(假阳性*C_1+假阴性*C_2)/N$$

其中，N 为样本总量，C_1 为每起假阳性事件（将下跌预测为上涨）带来的损失，C_2 为每起假阴性事件（将上涨预测为下跌）带来的损失。而最后我们决策的准绳，可能是通过机器学习建模，使得上面的损失函数 S 尽量小。

2.3 实时线性分类器介绍

2.3.1 广义线性模型的定义

（广义）线性模型是机器学习发展几十年来理论和工具上最为完备的模型：不管是分类还是预测，线性模型都可以进行实时更新和预测；线性模型的解释性非常优秀，每个变量的回归系数都可以用于解释模型；最后，我们可以通过增减变量，修改特定的回归系数对模型进行人为加工。

继续前文的符号定义，假设回归因变量为 Y，自变量为 p 维向量 X。在线性模型中，我们企图获得 p 维参数向量，让我们可以通过 X 个个元素的现行组合得到 Y。它们的关系可以通过下面的函数来表示：

$$Y \sim F(.)$$
$$E(Y) = f(\eta)$$
$$\eta = X^{\mathrm{T}} b$$

其中，F（）为因变量 Y 的累计概率分布，E（）为数学期望的计算。我们可以从以下两个部分来解读这个模型。

（1）线性输入

$\eta=X^{\mathrm{T}}b$，每个自变量 X_i 对模型输出的贡献都是线性的，其贡献大小都由对应的 i 来决定。当 $b_i=0$ 时，自变量 X_i 不会影响最后的预测。这些线性输入的总和会直接影响最后因变量的取值。

（2）可预计的输出

给定 η 时，因变量的取值由连接函数 f 和 Y 的分布 F 来决定。我们常用的 f 和 e 有以下三种情况。

- 当 $f(\eta)=\eta$，且 F（）为正态分布的累计概率分布时，模型等于对正态分布的连续变量进行线性预测。
- 当 $f(\eta)=1/(1+\exp(\eta))$，且 F（）为二项分布累计概率分布的时候，模型等于逻辑回归模型，可用于对男女、好恶等类别进行分类预测。
- 当 $f(\eta)=\exp(\eta)$，且 F（）为泊松分布累计概率分布的时候，模型等于泊松模型，可用于对订票人数、车辆通过数量等数据进行预测。

综上所述，众多数据模型都是可以通过线性模型的特殊情况进行建模预测的。

2.3.2 训练线性模型

给定已知的样本 $\{(X_i, Y_i)\}_{i=1}^{n}$，假设现在需要通过模型训练得到线性模型参数 b，那么我们往往会定义目标函数 L，通过随机梯度下降的方法求得 b，使得 L 尽量小：

$$L=\frac{1}{n}\sum_{i=1}^{n}\left[f(x_i^{\mathrm{T}}b)-Y_i\right]^2+\lambda_1|b|_1+\lambda_2|b|_2$$

其中，λ_1 和 λ_2 是预先设置好的非负参数，$|\cdot|_1$ 为计算 L_1 的范数，$|\cdot|_2$ 为计算 L_2 的范数。

上面的目标函数可以分为如下两部分来理解。

- 预测误差：目标函数 L 第一项预测误差，我们训练一个模型当然是希望其得到的误差应尽量小。
- 惩罚函数（penalty function）：目标函数 L 中第二、三项的存在是为了防止所得模型的过度拟合，加入 L_1 惩罚函数还可以进行变量优先选择。

这里的参数 λ_1 和 λ_2 都是实现选择的参数，可以通过多次比较不同的模型来获取最有效的组合。

现在对线性模型的拟合工作已经在主流机器学习软件工具中完全自动化，在 Scikit-learn 中，对线性回归模型的拟合主要采用 sklearn.linear_model.SGDRegressor，对于分类问题，主要采用 sklearn.linear_model.SGDClaffier。

2.3.3 冷启动问题

机器学习应用中，其实收集数据才是最昂贵的一部分。若没有数据，那么一切模型都将是空中楼阁。对于新企业或新项目，没有数据进行模型训练，那么怎么样才能有最初始的模型呢？没有数据就有没模型，但是如果没有模型，往往也会难以收集到数据。怎么样才能解决这个鸡生蛋、蛋生鸡的问题呢？这个问题可能会因为不同的组织而有不同的答案，这里主要总结如下两个方案。

1. 借用其他相关数据

如果无法获得当前组织的机器学习数据进行建模，那么其中一个办法是从其他来源获取类似的数据，建立暂时能用的模型。等到产品成熟了，收集到足够多的数据以后再开发自身专有的模型。

例如，某初创业公司需要对小说影评的正负评价进行分类。但苦于暂时没有现成的数据，因此借用了相关网站，如豆瓣、知乎等帖子的内容，作为训练数据；又因为没有评价正负标签，该公司将豆瓣评分、知乎投票数量进行转化，获得了模型的正负标签。

2. 人工参与

在遇到建模冷启动问题的时候，该模型的使用人数往往并不高，如果对延迟的要求不高，完全可以通过人工标记的方法来解决。

例如，国内某家已经上市的门户视频网站，成立多年以来，分类、标记、推荐等业务都是通过人工完成的，且取得了尚佳的结果。如今该网站上市之后拥有了雄厚的资金实力，聘请了顶尖的机器学习专家进行视频的标签标记和推荐。此时通过多年的努力该网站已经积累了大量的标签数据，建模的效果也相当好。

另外一方面，处理冷启动问题的时候，我们也可以将人工意见写入模型之中，使其自动化运行。例如对于股价走势预测模型，我们可以通过人工经验，对历史走势、成交量等因子进行人工打分，将人工打分的结果放入现行模型中，进行前期应用。

当然，所有人工参与的方式都离不开严格的监督流程。本书的第 9 章会介绍通过 Elasticsearch 对数据进行可视化分析和质量监控的方法。