Think Stats： 用Python数据分析

前言

本书介绍了跟数据分析相关的工具。参照平时我分析数据时的流程，组织本书内容：

* 数据导入与清理：无论数据格式怎样，通常都需要花时间和精力来读入数据，清洗和改变格式，check that everything made it through the translation process intact.
* 单变量探研：通常， 我会先集中研究单个变量，确定这些变量背后的意义，观察这些值的分布，最终确定合适的汇总性统计。
* 逐对探研：为了探测出变量之间可能的关联，我会观察相关表格和单点图，计算相关性和线性拟合。
* 多元分析： 如果发现明显的关联关系，我会使用多回归来添加控制变量，并审查更复杂的关联关系。
* 评估和假设性测试：在做统计结果报告时，回答这些问题是很重要的：效果有多大？如果再重新测量的话，会有多大的可复再性？那些明显的效果改变的可能性有多大？
* 数据可视化：在数据探研过程中，数据可视化可以很好地帮助我们找到可能的关联和效果。最终，结果为了经受住考验，数据可视化是一个有效的方式来跟人交流结果。

本书采用了程序模式（而非数学模式），相对于数学模式，程序模式有如下好处：

* 用Python代码，而不是数据公式来表达想法。Python代码更有可读性，同时，由于Python可直接执行，读者可下载执行并修改它。
* 每一章都包含了练习题，读者可下载下来体验并借以加固对学习的理解。一般来说，你写代码时，你就是在通过代码来表达理解；而调试代码时， 你又是在纠正理解。
* 一些练习题包含了试验来测试统计学特性。例如，你可以通过生成随机样本并计算他们和的方式来探研CLT。最终的可视化结果演示了为什么CLT是可行的，什么时候又失效。
* 用数学方式不容易理解一些想法时，使用模拟会更容易理解。例如，我们可以运行随机模拟来近似地估算出P值，这样可以加固对P值的理解。
* 由于本书使用了Python这个通用的编程语言，读者可以从几乎任何数据源导入数据。这些数据不再局限于那些已经清理且为特定统计工具而格式化好的数据。

本书以工程方式展开。在我的课上，学生们会参与到一个学期长度的工程中来，这个工程需要学生来提出统计方面的问题，寻找相关数据来解决这个问题，解决过程中使用学到的技术。

为了演示我解决统计分析问题的思路，贯穿本书所有章节的有一个案例分析。这个案例使用了下面两个来源的数据：

* 美国家庭增长调查（NSFG），这个调查是由美国疾病控制和预防中心实施的，旨在收集这些数据：美国家庭生活、婚姻与离婚、怀孕、不孕情况、避孕情况和男女性健康。
* BRFSS（）， 这个调查由执行， 旨在。

其它的例子使用的数据有IRS、美国统计局和Boston Marathon。

本书的第二版包含了第一版中的所有章节，其中不少章节是重写的，新写了些关于回归、时间序列分析、生存分析和分析方法。第一版本时没有使用Pandas,SciPy或StatsModels，这样所有的这些材料都是新的。

* 1. 怎么写本书

人们在写一新教科书时， 通常会先读一大堆旧的书。 这样的结果，大多数书都有差不多的内容，又是几乎一样的顺序。

我并没有这样做。实际上， 在写本书时，我几乎没有用到打印版的材料，有这样的几个原因：

* 我的目标是用新的方式来处理这些材料，所以也没多大的兴趣在已经存在的方式上。
* 由于我想通过一种免费的方式发布此书，我也想确保此书中任何部分都不受版权的限制。
* 本书的不少读者并不能方便地从图书馆借到书，所以我尽力使用网上可得到的材料作为引用。
* 一些旧媒体的支持者认为过多地使用电子材料是懒惰和不可靠的。在懒惰方面，他们是对的，不过在第二个问题上，我认为是错误的，所以我想测试下我的理论。

本书写作过程中， 我使用最多的是Wikipedia.一般来说，Wiki上读到的统计材料是很好的（尽管写作的过程中我还是做了些修改）。本书里也包含进了不少Wiki页面引用，我也鼓励大家来看这些Wiki材料.大多数情况下，Wiki页面的内容填补了我描述的空白。词汇和标记也保持着跟Wiki的一致，除非我有足够的理由来修改。其它一些有用的材料是Wolfram MatchWorld和Reddit统计论坛里的内容。

* 1. 使用代码

本书使用的代码和数据都可以在<https://github.com/AllenDowney/ThinkStats2>里找到，Git是一个版本控制软件，使用它可以跟踪项目中的文件。Git控制的文件集合称为仓库。GitHub是提供了Git仓库和方便的操作界面。

我仓库的GitHub主页里提供了多种方式来使用代码：

* 通过Fork方式来创建一个Copy。
* 直接Clone我的的仓库，不过，这样你不能再修改代码后上传。
* 不使用Git情况下，可直接从GitHub页面里下载Zip包。

所有的代码都是用Python2和Python3写的，不需要再做转换。

写本书的过程中，我使用了Anaconda这个工具（包含了pandas、NumPy、SciPy、StatsModels和matplotlib）。Anaconda可方便安装，默认情况下，安装时，是用户级的，而非系统级的，这样你不再需要管理员权限。它都支持Python2和Python3。

本书中使用的Python包，虽然是通用的，但并没包含在标准的Python安装文件里，在一些环境中也不好安装。如果你安装这些包有问题时， 我强烈推荐你使用Anaconda或其它一些已经包含这些工具的安装包。

在clone了Git仓库后（或解压了Zip包后），你将看到ThinkStats2这个文件夹下有一个叫nsfg.py的文件。运行此文件，它会读一个数据文件，执行些测试，打印些消息，如“All tests passed”。如果执行中有什么错，很有可能意味着还有些Python包需要安装。

大多数练习是使用Python脚本的，不过有些也使用IPython notebook。如果以前没有使用过IPython的话，可参考<http://ipython.org/ipython-doc/stable/notebook/notebook.html>文档。

写本书时一个假定：读者对Python和OOP已经熟悉，对pandas、NumPy和SciPy没有要求。如果你对这三个已经很熟悉了，可以跳过其中的一些章节。

数学方面，我也假定读者知道些基本的数学常识，如对数和求和。期间我会提到一些微积分的概念，不过你不需要微积分有了解。

如果你没有学习过统计学，我认为此书是一个不错的开始。

1. 探研数据分析

本书的命题：武装了可用方法的数据可以解答问题并帮助在不确定情况下做决定。

作为一个例子，列举一个案例分析。妻子和我在等待第一个孩子时，听说一般第一个孩子会延迟一段时间出生，从统计上来说，对否？这个案例就是这个问题引起的。

如果搜索下的话，会发现不少这方面的讨论。一些人认为这个说法是对的，一些人认为不靠谱，也有一些有完全相反的说法：第一个孩子一般会早生。

一些讨论中，人们会提供些数据来证明他们的观点。我发现下面这些比较典型：

“我两个朋友最近刚生了第一个孩子，他们都是有两周的延迟。”

“我第一个孩子比预产期晚了两周，现在我认为第二个孩子要早出生两周了！！”

“我不认为这个说法是对的，因为我姐姐是妈妈的第一个孩子，她还早出生了，不少表兄妹都这样。”

这样的报道称为轶事证明，由于他们都是基于些未经公开发布和个人的数据。一般来说， 这样的轶事也无可厚非，所以我也不想指责有这样说法的人。

不过， 我们还是想更有说服力和可靠的答案。这些轶事证明大多是失败的，因为：

* 观察的数据量太小:对第一个孩子来说，如果怀孕周期长的话，the difference is probably small compared to natural variation。
* 选择性偏见： 参与讨论的人们可能会因为他们的第一个孩子出生而感兴趣。这样情况下，选择的数据会把结果带偏。
* 确认的偏见：相信这个说法的人会贡献证据来证明，反对的人会提供反对的证据。
* 不准确：轶事通常是个人讲述的， 经常会记错、表述错误并不准确地重复等。

这样，我们怎么才能做的更好呢？

* 1. 统计学方式

为了纠正轶事的不限制，我们使用统计工具，包括：

* 数据收集： 我们将使用全美国范围收集的数据，这些数据当时就是为了有效地统计美国人口的。
* 描述性统计：我们将生成统计结果， 它将简明地给出数据摘要，使用（evaluate）不同的方法来可视化数据。
* 探研式数据分析:我们将找寻模式，区别和其它的特性（针对我们感兴趣的问题）。同时，我们将校验数据的一致性并找到不足。
* 估算：我们将使用样式数据来评估出些整体特征。
* 假设测试：在发现有明显效果的地方（如两组有不同时）,我们会评估这个效果是不是偶然造成的。

经过上面步骤后，我们可以得出一个可信且极有可能是正确的结论。

* 1. NSFG

从1973年开始，美国的CDC开始执行NSFG调查，旨在收集“这些方面的信息：家庭生活、结婚和离婚、怀孕及不孕情况、避孕使用、男女性健康”。收到的数据为了制定健康服务部署和健康教育计划。

我们将使用这些数据来调查是否第一个孩子出生会晚些和其它问题。为了有效地理解这些数据，我们需要理解此调查的构思。

NSFG是一个跨Sectional的调研，也即在时间的某个点上本调研取一个组织的快照。跟跨Sectional相对的是纵向的调研，这种调研方式会在一定时间内重复地观察一个组织。

NSFG已经进行了7次了，它每一次执行称为一个Cycle。我们将使用Cycle6的数据，这个数据是从2002年1月到2003年3月。

调研的目标是针对人口情况得出统计结论，NSFG调研的目标人口是美国15到44年龄段的人。理想情况下，调查应该收集人口中每一个成员的数据，不过这个很不现实。相反，我们从一个称为样本的人口子集里收集数据。参与调查的人口称为调查对象。

通常情况下，跨Sectional的调研应该是representative，意味着：每一个人口对象都有平等的机会参与。这个目标也很达成，不过， 还是要尽可能地接近。

NSFG并不是representative，相反它估计设计成oversampled（采样过密）。调研的设计者募集了三组（西班牙裔、非洲裔和十几岁的青年人），比例要其比整个美国人口比例高， 以便能推导出足够的统计结论。

当然，采样过于密集的一个问题是，用这个统计结果可能不容易得到更为普遍的结论。后续我们会再讨论这个不足。

在处理这些数据时， 很有必要熟悉下codebook，这个文档描述了NSFG调研的设计，调研的问题和受访的反馈。相当材料请见http://www.cdc.gov/nchs/nsfg/nsfg\_cycle6.htm

* 1. 导入数据

代码和使用的数据可以从<https://github.com/AllenDowney/ThinkStats2>得到。

下载后， 可以找到这个文件ThinkStats2/code/nsfg.py。运行一下，它会读一个数据文件，执行一些测试，并打印下这样的Message：“All tests passed”。

我们看下，都做了什么。NSFG Cysle6的怀孕情况数据在一个叫2002FemPreg.dat.gz的文件中。它是一个gzip压缩的文本文件，有固定宽度的列。文件中每行是一个记录，它含有怀孕情况的数据。

文件的格式记录在2002FemPreg.dct中， 这是一个Stata字典文件。Stata是一个统计软件系统，本语义环境中的字典文件是一个列表，每一项都有变量名字、变量类型和索引（这个索引会告诉我们从哪一行中找到每一个变量）

1. 数据分布

2.1 直方图显示

描述一个变量的一个最好方式是汇报数据集中数值和每一个数据出现的次数。这个描述平常称为变量的分析。

最常用的是直方图，这是一个表示每一个值出现频次的图。本语境中，频次意味着每一个值出现的次数。

Python中， 一个有效地计算频次的方式是使用字典。对于一个给定的值序列t:

Hist = {}

for x in t:

hist[x] = hist.get(x,0) +1

最终的结果是一个字典实例，它收集了从值到频次的映射。另外，也可以使用collections包中的Counter类。

From collections import Counter

counter = Counter(t)

结果是一个Counter实例，它是字典的子类。

另一个的实现方式是使用pandas提供的方法value\_counts,前面的章节里我们已经看到这个方法的使用了。不过， 本书中， 我创建了一个类Hist，它表达直方图的数据，并有方法来操作这些数据。

2.2 表达直方图

Hist的构造方法可以接受一个序列，或字典或pandas的Series，或另一个Hist。你可以这样创建一个Hist：

>>> import thinkstats2

>>> hist = thinkstats2.Hist([1,2,2,3,5])

>>> hist

Hist({1:1,2:2,3:1,5:1})

Hist对象有一个Freq方法， 它接受一个值返回对应的频次值：

>>>hist.Freq(2)

2

方括号操作符也可以表达一样的含义：

>>>hist[2]

2

如果用一个没有包含的值，返回的频次将是0.

>>> hist.Freq(4)

0

values方法返回一个没有排序的list：

>>> hist.Values()

[1,5,3,2]

为了顺序地遍历这些值，可以使用Python内置的sorted方法：

For val in sorted(hist.Values()):

print(val, hist.Freq(val))

或者使用Items方法来遍历值与频次对：

for val, freq in hist.Items():

print(val, freq)

2.3 绘制直方图

本书的例子中， 我写了一个叫thinkplot.py的包，它提供了绘制Hists或thinkstats2.py中定义的其它对象的方法。 这个thinkplot包， 使用了pyplot，pyplot是matplotlib包的一部分。

可以使用下面的代码来绘制hist：

>>> import thinkplot

>>> thinkplot.Hist(hist)

>>> thinkplot.Show(xlabel=’value’, ylabel=’f’)

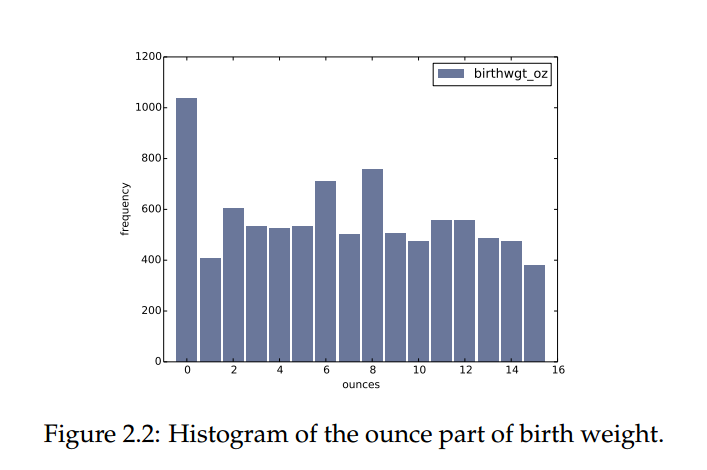
2.4 NSFG变量

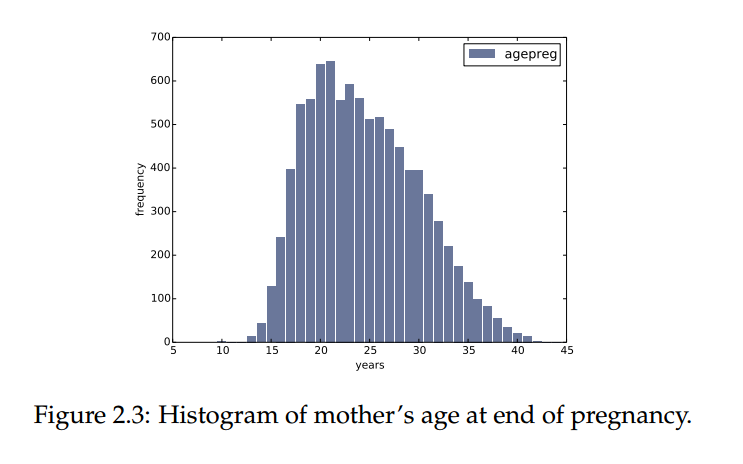
现在回到NSFG的数据。本章使用的代码在first.py文件中。

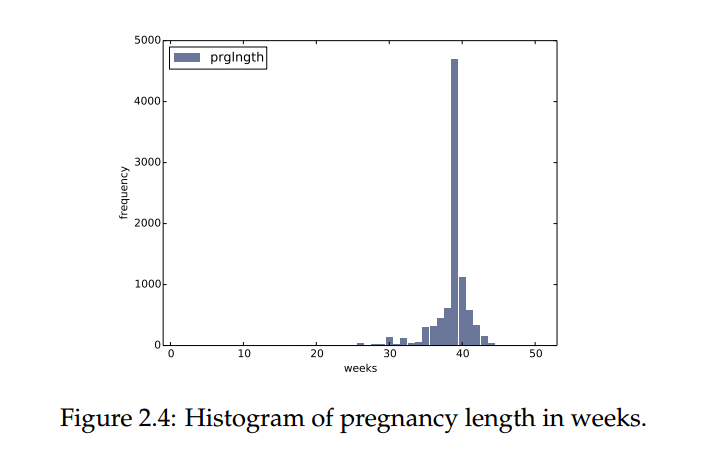
当开始处理一个新的数据集时，建议先整体过一遍这些数据，一个好方法是先看其直方图数据。

1.6节中， 我们从centiyears数值转成agepreg，将birthwgt\_lb和birthwgt\_oz合并成一个单一的数值，totalwgt\_lb.本节中，我使用这些变量来描述直方图数据的特性。

我们开始时先读取这些数据，选活着出生的数据：







preg = nsfg.ReadFemPreg()

live = preg[preg.outcome == 1]

方括号中的表达式是一个布尔型的Series，它从DataFrame中选择一系列行并返回一个新的DataFrame.接下来，我将生成并绘制活着婴儿的birthwgt\_lb数据：

Hist = thinkstats2.Hist(live.birthwgt\_lb, label=’birthwgt\_lb’)

thinkplot.Hist(hist)

thinkplot.Show(xlabel=’p’, ylabel=’f’)

传到Hist的参数类型是pandas的Series对象时， 那些nan值将直接丢弃。label是一个字符串，它将出现在图表的legend。

图2.1显示了结果。出现最多的值，称之为mode，是7磅。整体的分布是一个钟型，这也是正态分布，或者叫高斯分布。不过跟真正的正态分布有所不同的是，这个分布是非对称的。它左边的尾巴从右边的尾巴伸的远些。

图2.2显示了birthwgt\_oz分布。理论上，我们期望这个分布是均匀的，也就是说，所有的数值都有相同的频次。而实事上，数值0比其它值出现的多些，1和15少了些，可能是由于受调者四舍五入地处理了出生重量数值。

图2.3显示了agepreg数据分布， 即母亲在婴儿出生时的年龄。Mode值是21岁。这个分布也基本上是钟型分布。不过，这次右边的尾巴比左边的尾巴长了些。大多数母亲在她们20来岁，少量的在30岁。

图2.4显示了prglngth分布，即怀孕的周数。出现最多的周数是39。左边的尾巴比右边的要长，也就是说早产儿比晚产的多些。晚产的很少超过43周的，晚产再多时， 医生就会介入了。

2.5 极端值情况

查看直方图时， 很容易视别出最常见的数值和分布的形状，不过极端值不容易发现。

极端值通常是测量和汇报时出错了， 或者在无误的情况下反映了很少出现的情况。

Hist提供了Largest和Smallest方法， 这些方法接受一个整型的n，返回n个最大或最小的数值：

for weeks, freq in hist.Smallest(10):

print(weeks, freq)

在活着出生的怀孕长度记录中， 10个最小值是[0,4,9,13,18,19,20,21,22]。小于10周的数值肯定是出错了。很可能是outcome数值没能正确地编码。数值大于30周的一般是可法的。10到30周的数据不好确定。一些数据是错误的， 一些数据代表了早产未成熟的婴儿。

另一个极端值是最大值：

weeks count

43 148

44 46

45 10

46 1

47 1

48 7

50 2

怀孕超过42周时，大多数医生会建议引产，所以更大些的数据就有些出奇了。特别是，50周的怀孕时长，在医学上太不正常了。

处理这些极端性的方式，大多情况下视具体的业务情况而定。也看你想具体做出什么样的分析来。

本例中， 最初始的问题是，第一个孩子是否会出生早些。当人们提问出这个问题时，他们通常更关注足月的怀孕情况，所以本次分析中， 我将集中在那些怀孕时长超过27周的数据。

2.6 第一个婴儿

现在我们可以对比第一个婴儿和其它婴儿的怀孕时间长短了。用birthord把DataFrame里的活着出生婴儿区分开，并计算他们的直方图数据。

firsts=live[live.birthord == 1]

others=live[live.birthord!=1]

first\_hist = thinkstats2.Hist(firsts.prglngth)

other\_hist = thinkstats2.Hist(others.prglngth)

随后，使用相同的坐标画出直方图：

width = 0.45

thinkplot.PrePlot(2)

thinkplot.Hist(first\_hist,align=’right’, width = width)

thinkplot.Hist(other\_hist,align=’left’, width=width)

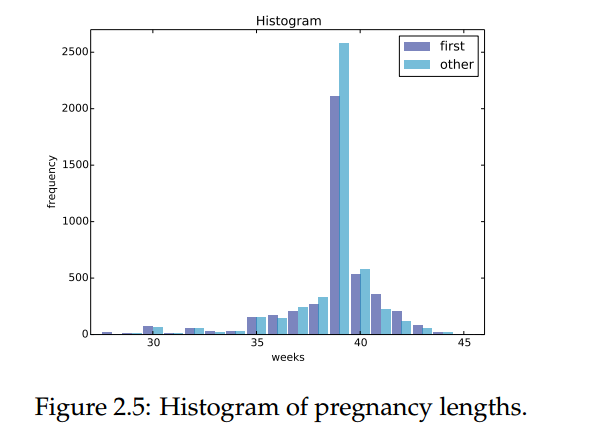
thinkplot.Show(xlabel=’weeks’, ylabel=’frequency’)

thinkplot.PrePlot接受直方图的个数，它使用这些信息来选择不一样的颜色。

Thinkplot.Hist通常使用align=’center’，这样每一个bar相对于数据来说在中心位置。本图中，我使用align=’right’和align=’left’将bar放到数的两侧。

通过width=0.45,设置两个bar的总宽度为0.9，在每个pair间保留点空间。

最终，调整坐标来只显示27到46周的数据。图2.5显示了结果。



直方图经常使用，因为它将最经常出现的数据很直观地显示出来。不过，比较两组数据的分布情况时，直方图不是最好的选择。本例中， 由于第一个孩子的数据少于其它的，所以some of the apparent differences in the histograms are due to sample sizes。下一章中， 我们使用概率聚集函数来解决此问题。

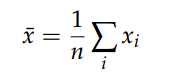
2.7 总结下分布图

直方数据是样本数据分布的全量描述，也就是说，给定一个直方数据，我们可以重现出样本数据（虽然不是完全复合顺序）

如果分布情况的细节很重要的话，很有必要来展现出直方图来。不过， 通常情况下，我们使用下面的几个特性来归纳分布情况：

* 向心趋势：是不是这些数据倾向于聚在一个特定点上？
* 聚集情况：是否有多于一个的cluster？
* spread情况：有多少个变量？
* 是不是数据会很陡地从中心点降下来？
* 极值情况：是否有极值远离于mode。

为了回答这些问题的统计称之为归纳统计。最常使用的统计技术是平均值。

平均值的定义如下：

通常情况下，mean和average是通用的， 不过，还是有此区分的：

* Mean是上面公式计算出来的。
* Average只是想表达中心化趋势的一个方式。

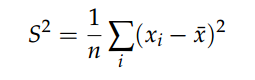
通常情况下，均值是描述一组数据的好方法。例如，苹果大都是一样的个头时，我买了6个，总重量是3磅，我们可以很自然地说每个苹果大概有半磅重。

不过， 南瓜就会多样性多了。假设我种了一些菜，一天，我收获三个decorative南瓜，有一个是一磅，两个是三磅重，还有一个Atlantic南瓜重591磅。均值是100磅，如果我告诉你“我种的南瓜average重量是100磅”的话， 很容易误导。本例中， 由于没有一个100磅左右的南瓜，这样的average就没什么意义了。

2.8 方差

如果没有一个数能概述南瓜的重量，我们可以使用两个数来表达的好一些：平均值和方差。

方差是一个概述性统计工具，旨在分布的波动或展开情况。 方差的定义如下：



方差的平方根称为标准偏差。

如果以前有过方差经验的话， 你可能见过使用（n-1）而不是n的公式。这种n-1的计算方式另有可用之处， 后续第8章再回过头来看这个话题。

Pandas的数据结构中提供了相应方法来计算均值、方差和标准偏差。

mean = live.prglngth.mean()

var = live.prglngth.var()

std = live.prglngth.std()

对所有的活着出生的孩子来说，平均怀孕时间是38.6周，标准偏差是2.7周，这也就是意味着，有一个两三周的偏差是很正常的。

方差是7.3周，这个不好理解。方差在一些计算中很有用，不过不算一个很好的统计工具。

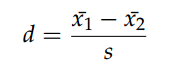
2.9 有效大小

有效大小是一个概括性的统计量， 用来描述有效性的大小。 例如，为了描述两组之间的差别，一个明显可选的是平均数的差别。

头胎孕期的平均长度是38.601， 非头胎时， 这个平均长度是38.523.再者相差0.078周，也就是13个小时。相对于通常的孕期时长，这个值的百分比是0.2%。

我们假定这个评估是准确的，这个差别没有实际的影响。实事上， 如果没有观察到如此大的数据量，也不会有人注意到这个差别。

另一个表达有效性大小的方式是对比组间的可变性。Cohen距离公式来表达这个统计量，定义如下：



这里的1和 2是两组数据的平均数, s是“池化的标准差”。下面是计算Cohen距离的代码：

def CohenEffectSize(g1, g2):

diff = g1.mean() – g2.mean()

v1 = g1.var()

v2 = g2.var()

n1, n2 = len(g1), len(g2)

p\_var = (n1 \* v1 + n2 \* v2) / (n1 + n2)

d = diff / math.sqrt(p\_var)

return d

上面的例子中，平均数的差值是0.029标准deviations， 这个值不大。作为对比，男女身高的差值是1.7标准deviations.

2.10 报告结果

我们已经看到了多个方式来描述第一个婴儿与其它婴儿孕期的长度（如果有一个的话）差。我们怎么汇总这些结果呢？

最终的回答依赖于谁在提出这个问题。科学家可能对实际的差值感兴趣，不论具体值有多大。医生可能只关心临床表现出来的差异，也即最终影响医疗决定的差异。孕妇可能更关心跟她相关的数据，如早产或晚产的可能性。

怎么汇总这些结果也取决于你的目标。如果你想表达这些效果的重要性，你可能会选用那些汇总性统计值。如果你想打消除病人的忧虑，你可能会选这样的统计量，以把这些差值放到具体的语境中。

不过，当然，你的决定也需要得到专业ethics的指导。说服性很有必要，你也需要设计统计报告和可视化以清楚地讲述故事。不过，你也应该尽力地使你的报告诚实可信，坦率地陈述出不确定和局限性。

1. 统计mass函数

本章中的代码在probability.py文件上。

3.1 Pmfs

另一种表达分布的方式是使用PMF，它会关联起每一个值及其出现的频次。频次除以样本个数n就是Probability。为了从频次得到其Probability， 我们对其除以n， 这个过程我们称之为归一化。

对于给定的Hist,我们可以创建一个字典，这个字典里保存着值到其出现纪律的隐射。

n = hist.Total((()

d = {}

for x,freq in hist.Items():

d[x] = freq / n

或者我们使用thinkstats2中提供的Pmf类。跟Hist一样，Pmf构造方法可以接收一个list， pandas序列,字典，Hist或另一个Pmf对象。下面是使用简单list的例子：

>>> import thinkstats2

>>> pmf = thinkstats2.Pmf([1,2,2,3,5])

>>> pmf

Pmf({1:0.2, 2:0.4, 3:0.2, 5:0.2})

Pmf做了统一化， 所以几率总和是1.

Pmf和Hist对象在很多方面很像，事实上， 他们都从父类继承了很多种方法。例如，Values和Items方法行为方式一致。最大的不同的是，Hist建立的是从值到个数的映射，而Pmf是从值到浮点型的映射。

使用Prob方法来查看特定值关联的几率

>>> pmf.Prob(2)

0.4

使用中括号表达式时， 效果一样。

你也可以通过增加值关联的几率方式来修改Pmf。

>>> pmf.Incr(2, 0.2)

>>> pmf.Prob(2)

0.6

或者也可以将几率值乘以因子：

>>> pmf.Mult(2, 0.5)

>>> pmf.Prob(2)

0.3

如果修改Pmf， 最终的结果可能不会再归一化，也即，最终的几率和将不再是1.为了检查此值， 你可以调用Total方法，它将返回几率的总和：

>>> pmf.Total()

0.9

可以调用Normalize方法来重新归一化：

>>> pmf.Normalize()

>>> pmf.Total()

1.0

Pmf对象提供了Copy方法，通过此方法， 你可以新建并修改一个拷贝，而不影响其原始值。

本节中使用的标记可能显得不一致， 不过这里有一个系统：我使用Pmf来表示类的名字，pmf表示类的实例，PMF表示数学统计中的概率Mass函数。

3.2 绘制PMF

Thinkplot提供了两种绘制Pmf的方法：

* 使用thinkplot.Hist可以将Pmf绘制成柱状图。当Pmf中数值不多时， 使用Pmf很有用。
* 使用thinkplot.Pmf可以将Pmf绘制成步状图。当Pmf中有大量数值且Pmf平滑时，使用此方法很有用。此方法两样适用于Hist对象。

另外， pyplot提供了一个称为hist的方法，此方法接收一个数据序列，计算其直方数据，并绘制出来。由于我使用了Hist对象，通常将不再使用pyplot.hist。

使用PMF而不是直方数据， 我们可以对比两组数据的分布， 而不至于被其样本数据的大小误导。使用此图， 可以看到， 第一个婴儿往往不会正点出生，通常会晚些（41周和42周）。

下面是生成图3.1的代码：

thinkplot.PrePlot(2,cols=2)

thinkplot.Hist(first\_pmf, align=’right’,width=width)

thinkplot.Hist(other\_pmf, align=’left’,width=width)

thinkplot.Config(xlabel=’weeks’, ylabel=’probability’, axis=[27,46,0.0.6]

thinkplot.PrePlot(2)

thinkplot.SubPlot(2)

thinkplot.Pmfs([first\_pmf, other\_pmf])

thinkplot.Show(xlabel=’weeks’, axis=[27,46,0,0.6])

PrePlot接收两个可选的参数rows和cols来生成风格图，此种情况下是两图占了一行。

位于左边的第一个图展示了使用thinkplot.Hist方法的Pmfs，我们前面已经看到。

第二次调用PrePlot方法会重新设置颜色的生成。随后SubPlot会调整成位于右边的第二个图上，展示了使用thinkplot.Pmfs方法的pmfs。我使用了axis选项来保证两个图在两样的横轴上， 这样的展示在你想对比两个图时很有用。

3.3 其它一些可视化

在想探索和视别模型与关联关系时， 使用直方图和PMFs很方便。 形成一定的想法后，一个好的提高就是设计一个展示方式， 尽可能地把你视别出来的模式清晰简洁地展示出来。

在NSFG数据中， 分布上最大的区别是在mode处。所以，我们对这些数据集中处理，并转化下来突现这些区别是很有意义的。

weeks = range(35,46)

diffs = []

for week in weeks:

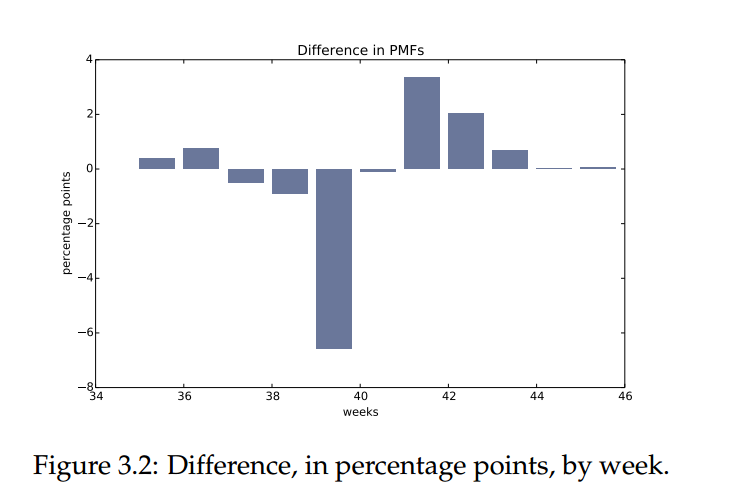
p1 = first\_pmf.Prob(week)

p2 = other\_pmf.Prob(week)

diff = 100 \* (p1 – p2)

diffs.append(diff)

thinkplot.Bar(weeks, diffs)



上面的代码中， weeks是周数的区间；diffs是两个PMFs百分比点的差。图3.2用柱状图展示了结果。这个图使模式更清楚：头胎较少在第39周出生， 反而更多地在第41周和42周出生。

现在， 我们暂且可以得到这样的结论。我们使用同样数据来展示一个明显的差别，随后选择一个合适的可视化图以将此差别明显化。我们不能确认这个效果是否真实，它可能是使用了一个随机调整的缘故。接下来我们将考虑这个问题。

3.4 类别大小驳论

在继续之前， 我想展示一个类型的计算（你可以使用Pmf来处理）， 我称这个例子为“类型大小驳论”。

在美国大学里， 学生和教职工的数量比大概是10比1.不过学生经常很惊讶地发现平均班级的大小要比10大。关于这个， 有两个原因：

* 学生每个学期通常要上四五个课，而教师只都一两个班。
* 享受小班授课学生的个数小， 而享受大班的个数多。

第一个因素比较明显，至少当被指出时是这样的，第二个更微妙。我们看一个例子。假定一个学校在一个学期提供65个班，这些班的大小分布如下：

|  |  |
| --- | --- |
| size | count |
| 5-9 | 8 |
| 10-14 | 8 |
| 15-19 | 14 |
| 20-24 | 4 |
| 25-29 | 6 |
| 30-34 | 12 |
| 35-39 | 8 |
| 40-44 | 3 |
| 45-49 | 2 |

如果你问Dean平均班级大小， 他将构造一个PMF，计算平均值，将汇报说班级的平均大小是23.7.下面是用到的代码：

d = {7:8, 12:8, 17:14, 22:4, 27:6, 32:12, 37:8, 42:3, 47:2}

pmf = thinkstats2.Pmf(d, label= ‘actual’)

print(‘mean’, pmf.Mean())

不过， 如果你调查下一组学生， 问他们班里有多少人，计算下平均值，你将会发现这个平均值将会更些。 我们来看下高了多少。

首先，我计算下学习观察到的分布值，这里跟每一个班大小相关联的概率使用了班级学生个数做了Biased操作。

Def BiasPmf(pmf, label):

new\_pmf = pmf.Copy(label = label)

For x,p in pmf.Items():

new\_pmf.Mult(x,x)

new\_pmf.Normalize()

return new\_pmf

对每一个班级大小的值x来说， 我们用它乘以频率（班级学生的个数）。结果是一个新的Pmf， 代表了biased分布。

现在我们可以画出实际和观察到的分布图：

biased\_pmf = BiasPmf(pmf, label = ‘observed’)

thinkplot.PrePlot(2)

thinkplot.Pmfs([pmf, biased\_pmf])

thinkplot.Show(xlabel=’class size’, ylabel=’PMF’)

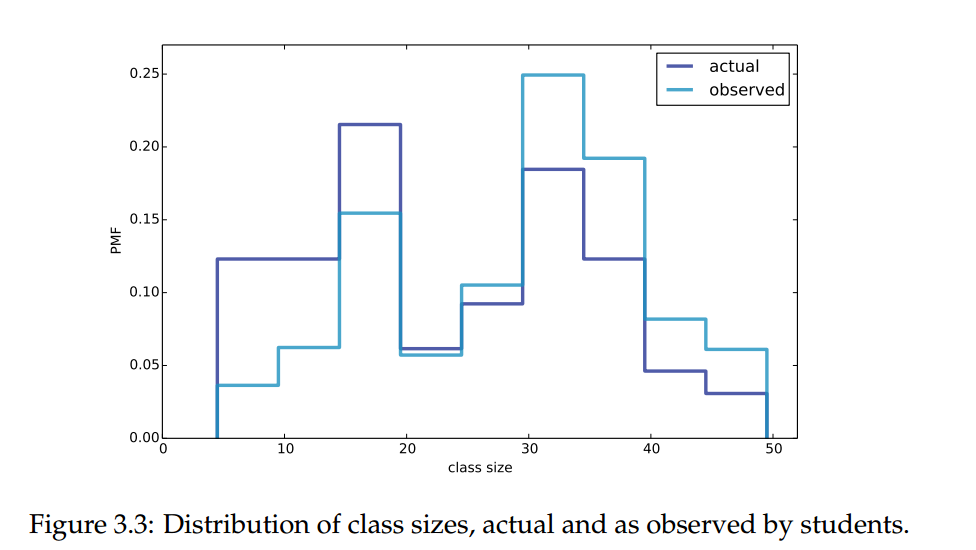


图3.3显示了结果。在这个Biased分布时，小班的个数少，大班的个数大。Biased分布的平均值是29.1，几乎是比实际值多了25%。

也可以方便地转化下这些数据。假定你想得现某一个学校的班级大小分布，不过从Dean里得不到可靠的数据。另一个可选的方式是， 选一组随机抽样的学生，并问他们班里有多少人。

最终的结果将是偏见的，跟我们前面看到的一样。不过， 你可以用它来评估最终的分布。下面是消除偏见的Pmf函数：

def UnbiasPmf(pmf, label):

new\_pmf = pmf.Copy(label=label)

for x,p in pmf.Items():

new\_pmf.Mult(x, 1.0/x)

new\_pmf.Normalize()

return new\_pmf

这个跟BiasPmf很像；唯一的区别是，这里对第一个概率值除以了x,而不乘以x。

3.5 DataFrame索引