

理解Gamma分布、Beta分布与Dirichlet分布



Eureka

600 人赞同了该文章

栏文章汇总

文章结构如下：

1：神奇的Gamma函数

- 1.1 Gamma函数
- 1.2 Gamma函数可视化
- 1.3 从二项分布到Gamma函数

2：Beta分布

- 2.1 认识Beta分布
- 2.2 可视化百变星君Beta分布
- 2.3 Beta-Binomial共轭
- 2.4 Beta分布的应用

3：Dirichlet分布

- 3.1 认识Dirichlet分布
- 3.2 可视化Dirichlet分布
- 3.3 Dirichlet-Multinomial共轭
- 3.4 Beta/Dirichlet 分布的一个性质

4：参考文献

| 注：本文主要参考了 Rickjin (靳志辉)大神的《LDA数学八卦》前两章内容。

1. 神奇的Gamma函数

1.1 Gamma函数

Gamma函数如下：

$$\Gamma(\alpha) = \int_0^{\infty} t^{\alpha-1} e^{-t} dt, \alpha > 0$$

很奇怪，但可以形象理解为用一个伽马刀，对 α 动了一刀，于是指数为 $\alpha - 1$ ，动完刀需要扶着梯子 ($-t$) 才能走下来（记忆，摘自QUETAL博客）。

通过分布积分可以得到如下性质：

$$\begin{aligned}\Gamma(\alpha + 1) &= \int_0^{\infty} t^{\alpha} e^{-t} dt = - \int_0^{\infty} t^{\alpha} d(e^{-t}) = - \left[t^{\alpha} e^{-t} \Big|_0^{\infty} - \alpha \int_0^{\infty} e^{-t} t^{\alpha-1} dt \right] \\ &= \alpha \Gamma(\alpha)\end{aligned}$$

易证明有如下性质：

$$\Gamma(n+1) = n!, \Gamma(1) = 1, \Gamma\left(\frac{1}{2}\right) = \sqrt{\pi}$$

其中还有几个重要的等式，这里就不证明了，有兴趣的可以查找相关资料：

$$\int_0^{\infty} x^{p-1} e^{-\alpha x} dx = \alpha^{-p} \Gamma(p)$$

$$\int_0^{\infty} x^{-(p+1)} e^{-\alpha x^{-1}} dx = \alpha^{-p} \Gamma(p)$$

$$\int_0^{\infty} x^{p-1} e^{-\alpha x^2} dx = \frac{1}{2} \alpha^{-\frac{p}{2}} \Gamma\left(\frac{p}{2}\right)$$

$$\int_0^{\infty} x^{-(p+1)} e^{-\alpha x^2} dx = \alpha^{-\frac{p}{2}} \Gamma\left(\frac{p}{2}\right)$$

1.2 Gamma函数可视化

```
import numpy as np
from scipy.special import gamma
import matplotlib.pyplot as plt

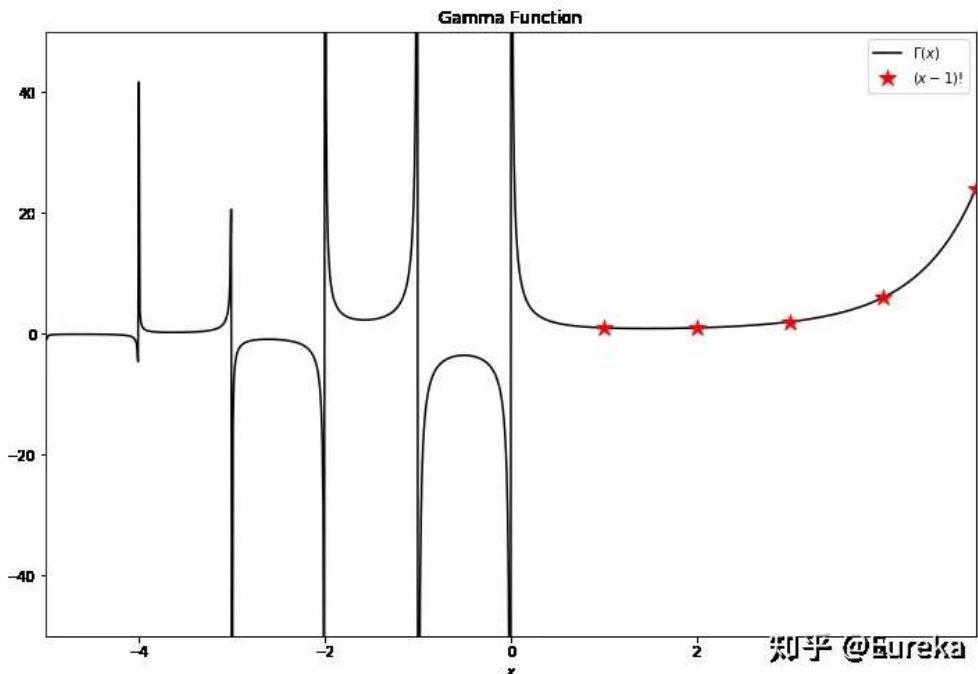
fig = plt.figure(figsize=(12,8))
# The Gamma function
x = np.linspace(-5, 5, 1000)
plt.plot(x, gamma(x), ls='-', c='k', label='$\Gamma(x)$')

# (x-1)! for x = 1, 2, ..., 6
x2 = np.linspace(1, 6, 6)
y = np.array([1, 1, 2, 6, 24, 120])
pylab.plot(x2, y, marker='*', markersize=12, markeredgecolor='r',
            markerfacecolor='r', ls=' ', c='r', label='$(x-1)!$')
```

```

plt.title('Gamma Function')
plt.ylim(-50,50)
plt.xlim(-5, 5)
plt.xlabel('$x$')
plt.legend()
plt.show()

```



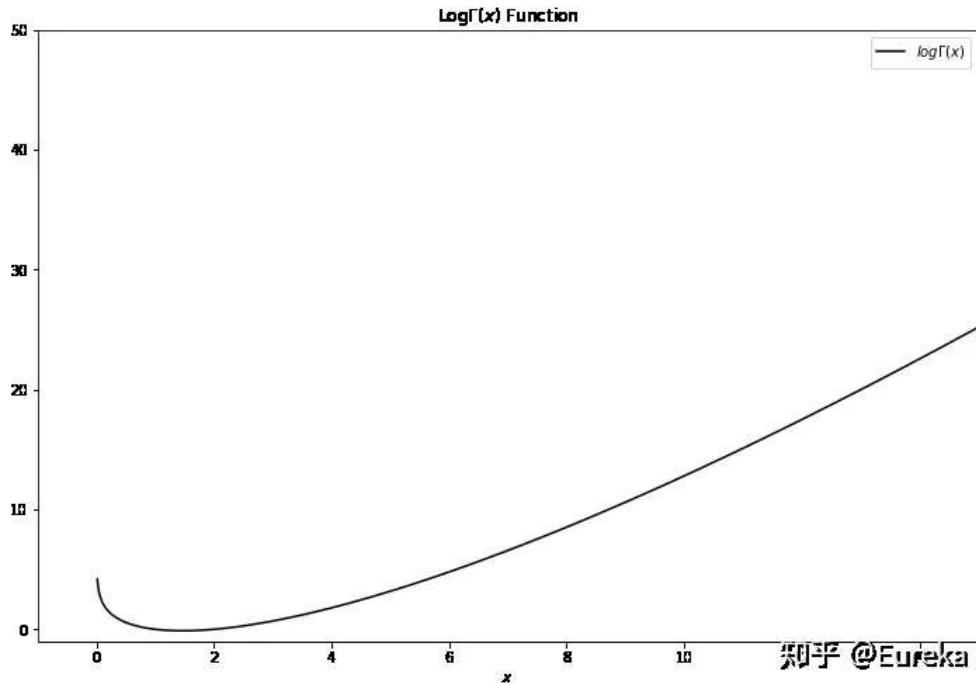
从图可以看出Gamma函数可凸函数，不仅如此， $\log\Gamma(x)$ 也是凸函数。

```

fig = plt.figure(figsize=(12,8))
# The Gamma function
x = np.linspace(0, 15, 1000)
plt.plot(x, np.log(gamma(x)), ls='-', c='k', label='$\log\Gamma(x)$')

plt.title('Log$\Gamma(x)$ Function')
plt.ylim(-1,50)
plt.xlim(-1, 15)
plt.xlabel('$x$')
plt.legend()
plt.show()

```



如下函数被称为Digamma函数:

$$\Psi = \frac{d \log\Gamma(x)}{dx}$$

Digamma函数具有如下性质:

$$\Psi(x+1) = \Psi(x) + \frac{1}{x}$$

1.3 从二项分布到Gamma函数

对Gamma函数做个变形，可以得到如下式子：

$$\int_0^\infty \frac{t^{\alpha-1} e^{-t} dt}{\Gamma(\alpha)} = 1$$

取积分中的函数作为概率密度，就得到一个简单的Gamma分布的密度函数：

$$Gamma(t|\alpha) = \frac{t^{\alpha-1} e^{-t}}{\Gamma(\alpha)}$$

如果做一个变换 $t = \beta x$ ，就得到Gamma分布的更一般形式：

$$Gamma(x|\alpha, \beta) = \frac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}$$

其中 α 称为shape parameter，主要决定了分布曲线的形状，而 β 称为rate parameter或 inverse scale parameter ($\frac{1}{\beta}$ scale parameter)，主要决定曲线有多陡。

```

import numpy as np
from scipy.stats import gamma
from matplotlib import pyplot as plt

alpha_values = [1, 2, 3, 3, 3]
beta_values = [0.5, 0.5, 0.5, 1, 2]
color = ['b', 'r', 'g', 'y', 'm']
x = np.linspace(1E-6, 10, 1000)

fig, ax = plt.subplots(figsize=(12, 8))

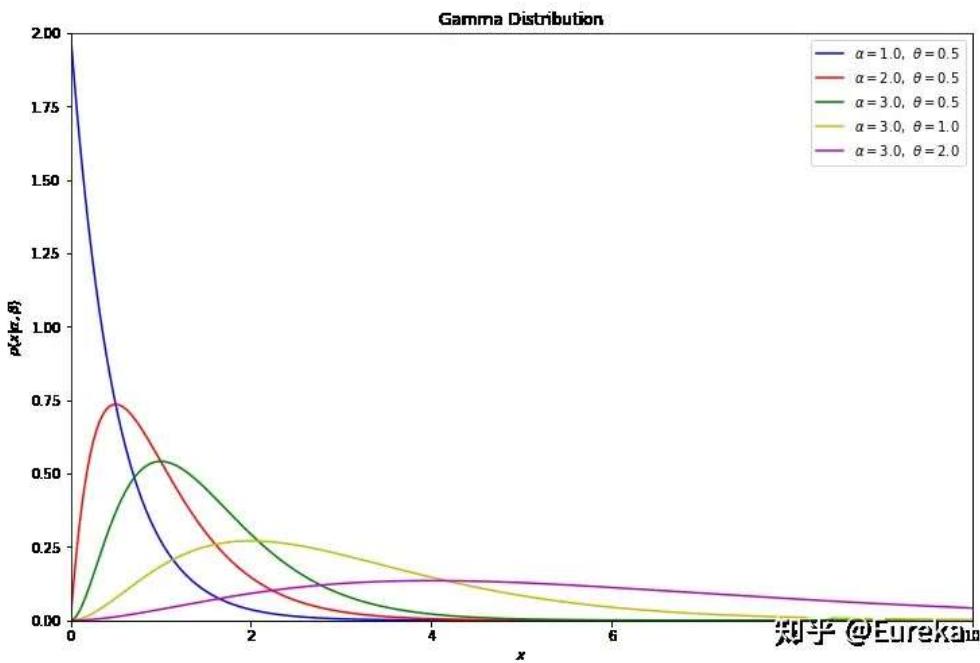
for k, t, c in zip(alpha_values, beta_values, color):
    dist = gamma(k, 0, t)
    plt.plot(x, dist.pdf(x), c=c, label=r'$\alpha=%1f, \theta=%1f$' % (k, t))

plt.xlim(0, 10)
plt.ylim(0, 2)

plt.xlabel('$x$')
plt.ylabel(r'$p(x|\alpha,\beta)$')
plt.title('Gamma Distribution')

plt.legend(loc=0)
plt.show()

```



我们可以发现Gamma分布的概率密度和Poisson分布在数学上的形式具有高度的一致性。参数 λ 的Poisson分布，概率为：

$$Poisson(X=k|\lambda) = e^{-\lambda} \frac{\lambda^k}{k!}$$

而在Gamma分布的密度函数中取 $\alpha = k + 1, \beta = 1$ ，可以得到：

$$Gamma(x|\alpha = k + 1) = \frac{x^k e^{-x}}{\Gamma(k + 1)} = \frac{x^k e^{-x}}{k!}$$

可以看到这两个分布在数学形式上是一致的，只是Poisson分布是离散的，Gamma分布是连续的，可以直观认为，Gamma分布是Poisson分布在正实数集上连续化版本。

我们在概率论与数理统计的课程中都学过， $Poisson(\lambda)$ 分布可以看成是二项分布 $B(n, p)$ 在 $np = \lambda, n \rightarrow \infty$ 条件下的极限分布：

$$B(k; n, p) = C_n^k p^k (1-p)^{n-k} \xrightarrow{np=\lambda, n \rightarrow \infty} Poisson(X = k | np = \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

二项分布也满足下面一个奇妙的等式：

$$P(x \leq K) = \frac{n!}{k!(n-k-1)!} \int_p^1 t^k (1-t)^{n-k-1} dt$$

这个分布式反应二项分布和 β 分布的关系，证明后面再讲。

我们在右等式做个变换 $t = \frac{x}{n}$

$$\begin{aligned} P(x \leq K) &= \frac{n!}{k!(n-k-1)!} \int_p^1 t^k (1-t)^{n-k-1} dt \\ &= \frac{n!}{k!(n-k-1)!} \int_{np}^n \left(\frac{x}{n}\right)^k \left(1 - \frac{x}{n}\right)^{n-k-1} d\frac{x}{n} \\ &= \frac{(n-1)!}{k!(n-k-1)!} \int_{np}^n \left(\frac{x}{n}\right)^k \left(1 - \frac{x}{n}\right)^{n-k-1} dx \\ &= \int_{np}^n \binom{n-1}{k} \left(\frac{x}{n}\right)^k \left(1 - \frac{x}{n}\right)^{n-k-1} dx \\ &= \int_{np}^n Binomial(Y = k | n-1, \frac{x}{n}) dx \end{aligned}$$

上式左侧是二项分布 $B(n, p)$ ，而右侧为无穷多个二项分布 $B(n-1, \frac{x}{n})$ 的积分求和，所以可以写为

$$Binomial(X \leq k | n, p) = \int_{np}^n Binomial(Y = k | n-1, \frac{x}{n}) dx$$

对两边在条件 $np = \lambda, n \rightarrow \infty$ 条件下取极限，则左边有 $B(n, p) \rightarrow Poisson(\lambda)$ ，而右边有 $B(n-1, \frac{x}{n}) \rightarrow Poisson(x)$ ，所以得到：

$$Poisson(X \leq k | \lambda) = \int_{\lambda}^{\infty} Poisson(Y = k | x) dx$$

把Poisson分布展开，于是得到：

$$Poisson(X \leq k | \lambda) = \int_{\lambda}^{\infty} \frac{x^k e^{-x}}{k!} dx$$

此为Poisson-Gamma duality.

我们对上式两边取极限 $\lambda \rightarrow 0$ ，左边是Poisson至多发生 k 事件的概率， $\lambda \rightarrow 0$ 的时候就不可能有事件再发生了，故 $P(X \leq k) = 1$ ，于是：

$$1 = \lim_{\lambda \rightarrow 0} \int_{\lambda}^{\infty} \frac{x^k e^{-x}}{k!} dx = \int_0^{\infty} \frac{x^k e^{-x}}{k!} dx$$

该积分式子说明 $\frac{x^k e^{-x}}{k!}$ 在实数集上是一个概率分布函数，而这个函数恰好就是Gamma分布。我们继续把上式右边中的 $k!$ 移到左边，于是得到：

$$k! = \int_0^{\infty} x^k e^{-x} dx$$

于是我们得到了将 $k!$ 表示为积分的方法。

我们将 $Poisson(X \leq k|\lambda) = \int_{\lambda}^{\infty} \frac{x^k e^{-x}}{k!} dx$ 进行变换下：

$$Poisson(X \leq k|\lambda) + \int_0^{\lambda} \frac{x^k e^{-x}}{k!} dx = 1$$

我们可以看到，Poisson分布的概率密度累积函数和Gamma分布的概率密度累积函数有互补的关系。

做个小结：我们从二项分布的等式出发，同时利用二项分布的极限是Poisson分布，推导出了Gamma分布，同时把 $k!$ 表示成积分形式了。

2. Beta分布

2.1 认识Beta分布

我们将由几个问题来得引出几个分布：

问题一：

- 1: $X_1, X_2, \dots, X_n \stackrel{iid}{\sim} Uniform(0, 1)$
- 2: 把这个 n 个随机变量排序后得到顺序统计量 $X_{(1)}, X_{(2)}, \dots, X_{(n)}$
- 3: 问 $X_{(k)}$ 是什么分布

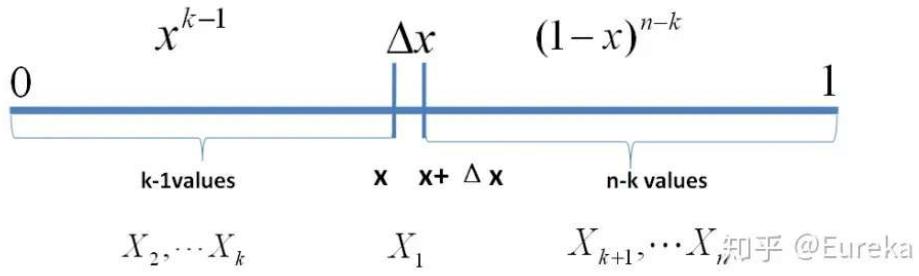
首先我们尝试计算 $X_{(k)}$ 落在一个区间 $[x, x + \Delta x]$ 的概率，也就是如下概率值：

$$P(x \leq X_{(k)} \leq x + \Delta x) = ?$$

我们可以把 $[0, 1]$ 分成三段 $[0, x], [x, x + \Delta x], (x + \Delta x, 1]$ 。

我们考虑第一种情形：假设 n 个数中只有一个落在区间 $[x, x + \Delta x]$ 内，则这个区间内的数 $X_{(k)}$ 是第 k 大的，则 $[0, x]$ 中应该有 $k - 1$ 个数， $(x + \Delta x, 1]$ 中有 $n - k$ 个数，我们将此描述为事件 E ：

$$\begin{aligned} E &= \{X_1 \in [x, x + \Delta x], \\ &\quad X_i \in [0, x] (i = 2, \dots, k) \\ &\quad X_j \in (x + \Delta x, 1] (j = k + 1, \dots, n)\} \end{aligned}$$



则有：

$$\begin{aligned} P(E) &= \prod_{i=1}^n P(X_i) \\ &= x^{k-1}(1-x-\Delta x)^{n-k}\Delta x \\ &= x^{k-1}(1-x)^{n-k}\Delta x + o(\Delta x) \end{aligned}$$

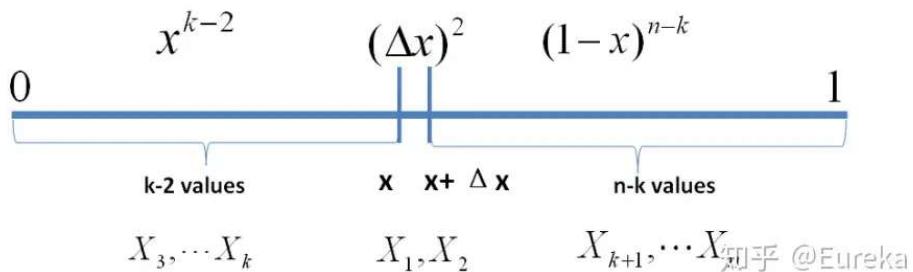
$o(\Delta x)$ 是 Δx 的高阶无穷小。显然 n 个数落在 $[x, x + \Delta x]$ 区间有 n 种取法，余下 $n - 1$ 个数中有 $k - 1$ 个数落在 $[0, x]$ 中有 $\binom{n-1}{k-1}$ 种组合，所以和事件 E 等价的事件一共有 $n\binom{n-1}{k-1}$ 个。

考虑第二种情形：假设 n 个数中只有两个落在区间 $[x, x + \Delta x]$ 内：

$$\begin{aligned} E' = \{X_1, X_2 &\in [x, x + \Delta x], \\ X_i &\in [0, x) (i = 3, \dots, k) \\ X_j &\in (x + \Delta x, 1] (j = k + 1, \dots, n)\} \end{aligned}$$

则有：

$$\begin{aligned} P(E') &= \prod_{i=1}^n P(X_i) \\ &= x^{k-2}(1-x-\Delta x)^{n-k}(\Delta x)^2 \\ &= o(\Delta x) \end{aligned}$$



从以上分析可以得到：只要落在 $[x, x + \Delta x]$ 内的数字超过一个，则对应的事件的概率就是 $o(\Delta x)$ ，于是：

$$\begin{aligned} P(x \leq X_{(k)} \leq x + \Delta x) &= n \binom{n-1}{k-1} P(E) + o(\Delta x) \\ &= n \binom{n-1}{k-1} x^{k-1}(1-x)^{n-k}\Delta x + o(\Delta x) \end{aligned}$$

所以得到 $X_{(k)}$ 的概率密度函数是:

$$\begin{aligned}f(x) &= \lim_{\Delta x \rightarrow 0} \frac{P(x \leq X_{(k)} \leq x + \Delta x)}{\Delta x} \\&= n \binom{n-1}{k-1} x^{k-1} (1-x)^{n-k} \\&= \frac{n!}{(k-1)!(n-k)!} x^{k-1} (1-x)^{n-k}, x \in [0, 1]\end{aligned}$$

我们知道利用Gamma函数可以把很多数学概念从整数集合延拓到实数集合。

我们在上式中取 $\alpha = k, \beta = n - k + 1$, 于是得到:

$$f(x) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1}$$

这就是Beta分布了。

那么这个游戏里我们取Beta分布的峰值是胜率最大的。

2.2 可视化百变星君Beta分布

```
import numpy as np
from scipy.stats import beta
from matplotlib import pyplot as plt

alpha_values = [1/3, 2/3, 1, 2, 2, 4, 10, 20]
beta_values = [1, 2/3, 3, 1, 1, 6, 4, 30, 20]
colors = ['tab:blue', 'tab:orange', 'tab:green', 'tab:red', 'tab:purple',
          'tab:brown', 'tab:pink', 'tab:gray', 'tab:olive']
x = np.linspace(0, 1, 1002)[1:-1]

fig, ax = plt.subplots(figsize=(14, 9))

for a, b, c in zip(alpha_values, beta_values, colors):
    dist = beta(a, b)
    plt.plot(x, dist.pdf(x), c=c, label=r'$\alpha=%.\mathbf{1f}, \beta=%.\mathbf{1f}$' % (a, b))

plt.xlim(0, 1)
plt.ylim(0, 6)

plt.xlabel('$x$')
plt.ylabel(r'$p(x|\alpha, \beta)$')
plt.title('Beta Distribution')

ax.annotate('Beta(1/3,1)', xy=(0.014, 5), xytext=(0.04, 5.2),
            arrowprops=dict(facecolor='black', arrowstyle='-'))

ax.annotate('Beta(10,30)', xy=(0.276, 5), xytext=(0.3, 5.4),
            arrowprops=dict(facecolor='black', arrowstyle='-'))

ax.annotate('Beta(20,20)', xy=(0.5, 5), xytext=(0.52, 5.4),
            arrowprops=dict(facecolor='black', arrowstyle='-'))

ax.annotate('Beta(1,3)', xy=(0.06, 2.6), xytext=(0.07, 3.1),
            arrowprops=dict(facecolor='black', arrowstyle='-'))

ax.annotate('Beta(2,6)', xy=(0.256, 2.41), xytext=(0.2, 3.1),
            arrowprops=dict(facecolor='black', arrowstyle='-'))

ax.annotate('Beta(4,4)', xy=(0.53, 2.15), xytext=(0.45, 2.6),
```

```

        arrowprops=dict(facecolor='black', arrowstyle='-'))

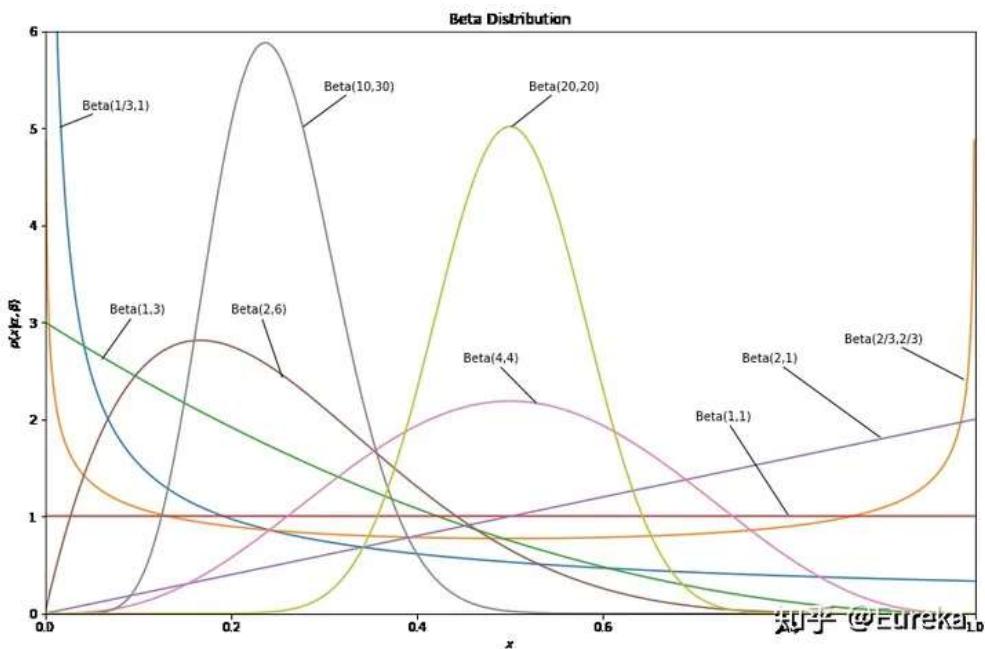
ax.annotate('Beta(1,1)', xy=(0.8, 1), xytext=(0.7, 2),
            arrowprops=dict(facecolor='black', arrowstyle='-'))

ax.annotate('Beta(2,1)', xy=(0.9, 1.8), xytext=(0.75, 2.6),
            arrowprops=dict(facecolor='black', arrowstyle='-'))

ax.annotate('Beta(2/3,2/3)', xy=(0.99, 2.4), xytext=(0.86, 2.8),
            arrowprops=dict(facecolor='black', arrowstyle='-'))

#plt.Legend(loc=0)
plt.show()

```



从图中可以看出，Beta分布可以是凹的、凸的、单调上升的、单调下降的；可以是曲线也可以是直线，而均匀分布也特殊的Beta分布。可以尝试改下参数，看看Beta分布的各种形态。

2.3 Beta-Binomial共轭

问题二：

- 1: $X_1, X_2, \dots, X_n \stackrel{iid}{\sim} \text{Uniform}(0, 1)$, 排序后对应的顺序统计量 $X_{(1)}, X_{(2)}, \dots, X_{(n)}$, 我们要猜测 $p = X_{(k)}$;
- 2: $Y_1, Y_2, \dots, Y_m \stackrel{iid}{\sim} \text{Uniform}(0, 1)$, Y_i 中有 m_1 个比 p 小, m_2 个比 p 大;
- 3: 问 $P(p|Y_1, Y_2, \dots, Y_m)$ 是什么分布

由于 $p = X_{(k)}$ 在 $X_{(1)}, X_{(2)}, \dots, X_{(n)}$ 中是第 k 大的，我们容易推得到 $p = X_{(k)}$ 在 $X_{(1)}, X_{(2)}, \dots, X_{(n)}, Y_1, Y_2, \dots, Y_m \stackrel{iid}{\sim} \text{Uniform}(0, 1)$ 这 $(m+n)$ 个独立随机变量中是第 $k+m_1$ 大的。可以按上一节的推导，此时 $p = X_{(k)}$ 的概率密度是 $\text{Beta}(p|k+m_1, n-k+1+m_2)$ 。

按贝叶斯推导的逻辑：

- 1): $p = X_{(k)}$ 是我们要猜测的参数，我们推导出 p 的分布是 $f(p) = \text{Beta}(p|k, n-k+1)$ ，称为 p 的先验分布。
- 2): 数据 Y_i 中有 m_1 个比 p 小, m_2 个比 p 大, Y_i 相当于做了 m 次贝努力实验，所以 m_1 服从二项分布 $B(m, p)$ 。
- 3): 在给定来自数据的提供的 (m_1, m_2) 知识后， p 的后验分布为 $f(p|m_1, m_2) = \text{Beta}(p|k+m_1, n-k+1+m_2)$

贝叶斯参数估计的基本过程是：

先验分布+数据知识=后验分布

因此可以得到：

$$\begin{aligned} & \text{Beta}(p|k, n-k+1) + \text{BinomCount}(m_1, m_2) \\ & = \text{Beta}(p|k+m_1, n-k+1+m_2) \end{aligned}$$

更一般的，对于非负实数 α, β ，我们有如下关系：

$$\text{Beta}(p|\alpha, \beta) + \text{BinomCount}(m_1, m_2) = \text{Beta}(p|\alpha+m_1, \beta+m_2)$$

以上式子实际上描述的就是**Beta-Binomial共轭**。共轭意思是先验和后验都服从同一个分布形式。这种形式不变，我们能够在先验分布中赋予参数很明确的物理意义，这个物理意义可以延伸到后验分布中进行解释，同时从先验变换到后验的过程中从数据中补充的知识也容易有物理解释。

(我感觉有共轭后计算更容易哈，因为形式都知道了，其他的就是凑参数了。还有另一个好处是：每当有新的观测数据，就把上次的后验概率作为先验概率，乘以新数据的likelihood，然后就得到新的后验概率，而不必用先验概率乘以所有数据的likelihood得到后验概率。)

从前面的过程中可以知道，Beta分布中的参数 α, β 也可以理解为物理计数，这两个参数经常被称为伪计数(pseudo-count)。基于以上逻辑，我们可以把 $\text{Beta}(p|\alpha, \beta)$ 写成下式来理解：

$$\text{Beta}(p|1, 1) + \text{BinomCount}(\alpha-1, \beta-1) = \text{Beta}(p|\alpha, \beta)$$

其中 $\text{Beta}(p|1, 1)$ 恰好的均匀分布 $Uniform(0, 1)$ 。

对于上式，可以从贝叶斯角度来理解。假设有一个不均匀的硬币抛出正面的概率是 p ，抛 m 次后得到正面和反面的次数分别为 m_1, m_2 次，那按传统概率学派的观点， p 的估计是 $\hat{p} = \frac{m_1}{m}$ 。而从贝叶斯学派的角度来看，开始对硬币的不均匀性一无所知，所以假设 $p \sim Uniform(0, 1)$ ，于是有了二项分布的计数 (m_1, m_2) 后，按照贝叶斯的公式计算 p 的后验分布：

$$\begin{aligned} P(p|m_1, m_2) &= \frac{P(p)P(m_1, m_2|p)}{P(m_1, m_2)} \\ &= \frac{1 \cdot P(m_1, m_2|p)}{\int_0^1 P(m_1, m_2|t)dt} \\ &= \frac{\binom{m}{m_1} p^{m_1} (1-p)^{m_2}}{\int_0^1 \binom{m}{m_1} t^{m_1} (1-t)^{m_2} dt} \\ &= \frac{p^{m_1} (1-p)^{m_2}}{\int_0^1 t^{m_1} (1-t)^{m_2} dt} \end{aligned}$$

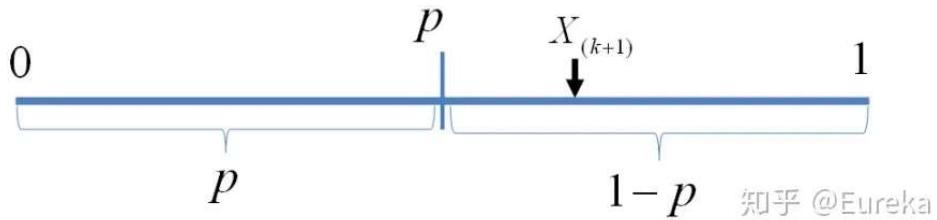
计算得到后验分布为正好是： $\text{Beta}(p|m_1+1, m_2+1)$

前面从二项分布推导Gamma分布的时候，使用了如下等式：

$$P(x \leq K) = \frac{n!}{k!(n-k-1)!} \int_p^1 t^k (1-t)^{n-k-1} dt$$

左边是二项分布的概率累积，右边是 $\text{Beta}(t|k+1, n-k)$ 分布的概率累积。现在我们来证明这个等式。

我们构造如下二项分布，取随机变量 $X_1, X_2, \dots, X_n \stackrel{iid}{\sim} Uniform(0, 1)$ ，一个成功的贝努利实验就是 X_i 。



我们可以得到：

$$P(C \leq k) = P(X_{(k+1)} > p)$$

此处 $P(X_{(k+1)})$ 是顺序统计量，为第 $k + 1$ 大的数。上述等式意思是：成功至多 k 次等于第 $k + 1$ 大的数必定失败（即失败至少 $n - k$ 次）。由于 $X_{(k+1)} \sim Beta(t|k+1, n-k)$ ，于是

$$\begin{aligned} P(C \leq k) &= P(X_{(k+1)} > p) \\ &= \int_p^1 Beta(t|k+1, n-k) dt \\ &= \frac{n!}{k!(n-k-1)!} \int_p^1 t^k (1-t)^{n-k-1} dt \end{aligned}$$

2.4 Beta分布的应用

1. 棒球击球率

那么我们简单说个Beta-Binomial共轭的应用。用一句话来说，beta分布可以看作一个概率的概率分布，当你不知道一个东西的具体概率是多少时，它可以给出了所有概率出现的可能性大小。

举一个简单的例子，熟悉棒球运动的都知道有一个指标就是棒球击球率(batting average)，就是用一个运动员击中的球数除以击球的总数，我们一般认为0.266是正常水平的击球率，而如果击球率高达0.3就被认为是非常优秀的。现在有一个棒球运动员，我们希望能够预测他在这一赛季中的棒球击球率是多少。传统的频率学派会直接计算棒球击球率，用击中的数除以击球数，但是如果这个棒球运动员只打了一次，而且还命中了，那么他就击球率就是100%了，这显然是不合理的，因为根据棒球的历史信息，我们知道这个击球率应该是0.215到0.36之间才对。对于这个问题，我们可以用一个二项分布表示（一系列成功或失败），一个最好的方法来表示这些经验（在统计中称为先验信息）就是用beta分布，这表示在我们没有看到这个运动员打球之前，我们就有了一个大概的范围。beta分布的定义域是 $(0, 1)$ 这就跟概率的范围是一样的。接下来我们将这些先验信息转换为beta分布的参数，我们知道一个击球率应该是平均0.27左右，而他的范围是0.21到0.35，那么根据这个信息，我们可以取 $\alpha = 81, \beta = 219$ 。（这样取值可以从Beta的均值和分布考虑）

```

import numpy as np
from scipy.stats import beta
from matplotlib import pyplot as plt

x = np.linspace(0, 1, 1002)[1:-1]

fig, ax = plt.subplots(figsize=(10, 6))

dist = beta(81, 219)
plt.plot(x, dist.pdf(x), c='b', label=r'$\alpha=%1f, \beta=%1f$' % (81, 219))

plt.xlim(0, .6)
plt.ylim(0, 16)

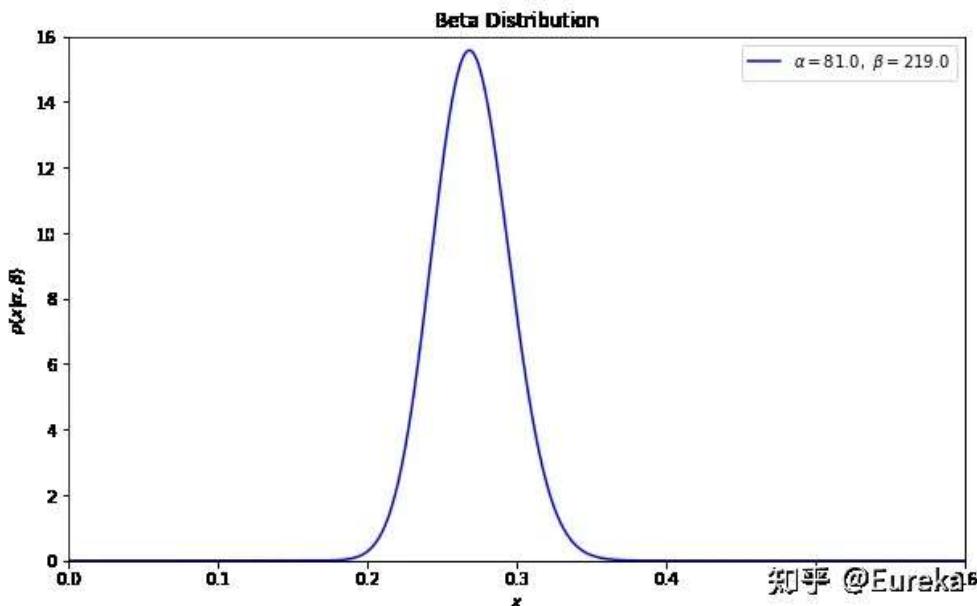
```

```

plt.xlabel('$x$')
plt.ylabel(r'$p(x|\alpha, \beta)$')
plt.title('Beta Distribution')

plt.legend(loc=0)
plt.show()

```



x 轴表示各个击球率的取值， x 对应的 y 值就是这个击球率所对应的概率。也就是说 beta 分布可以看作一个概率的概率分布。

那么有了先验信息后，现在我们考虑一个运动员只打一次球，那么他现在的数据就是“1中；1击”。这时候我们就可以更新我们的分布了，让这个曲线做一些移动去适应我们的新信息。因为 beta 分布与二项分布是共轭先验的 (Conjugate_prior)。那么后验是：

$Beta(\alpha_0 + hits, \beta + misses)$ 。

其中 α_0 和 β_0 是一开始的参数，在这里是 81 和 219。所以在这一例子里， α 增加了 1 (击中了一次)。 β 没有增加 (没有漏球)。这就是我们的新的 beta 分布 $Beta(81 + 1, 219)$ 。

```

import numpy as np
from scipy.stats import beta
from matplotlib import pyplot as plt

x = np.linspace(0, 1, 1002)[1:-1]

fig, ax = plt.subplots(figsize=(10, 6))

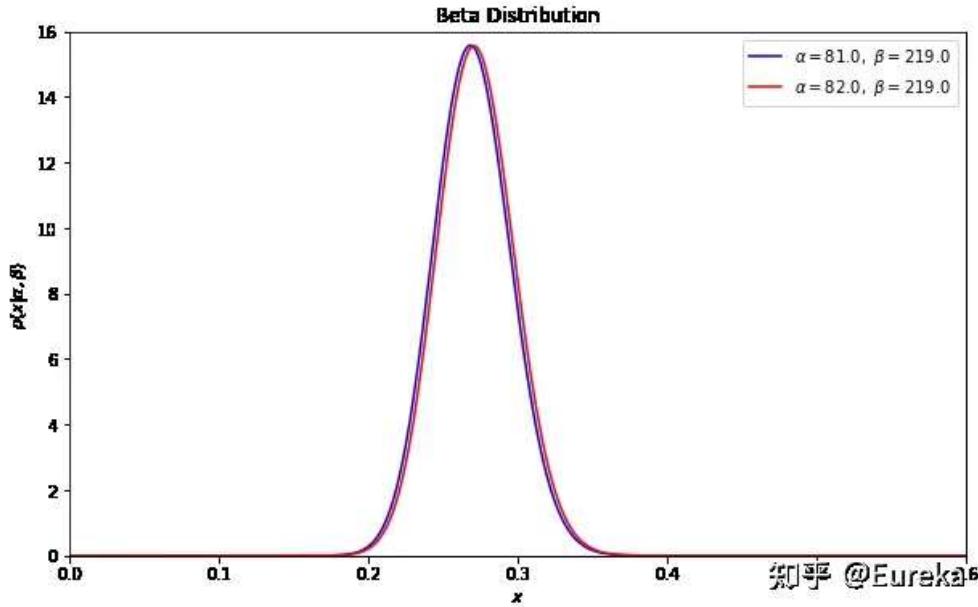
dist = beta(81, 219)
plt.plot(x, dist.pdf(x), c='b', label=r'$\alpha=%.1f, \beta=%.1f$' % (81, 219))

dist = beta(82, 219)
plt.plot(x, dist.pdf(x), c='r', label=r'$\alpha=%.1f, \beta=%.1f$' % (82, 219))

plt.xlim(0, .6)
plt.ylim(0, 16)

plt.xlabel('$x$')
plt.ylabel(r'$p(x|\alpha, \beta)$')
plt.title('Beta Distribution')
plt.legend(loc=0)
plt.show()

```



可以看到这个分布其实没多大变化，这是因为只打了1次球并不能说明什么问题。但是如果我们将到了更多的数据，假设一共打了300次，其中击中了100次，200次没击中，那么这一新分布就是： $Beta(81 + 100, 219 + 200)$

```

import numpy as np
from scipy.stats import beta
from matplotlib import pyplot as plt

x = np.linspace(0, 1, 1002)[1:-1]

fig, ax = plt.subplots(figsize=(10,6))

dist = beta(81, 219)
plt.plot(x, dist.pdf(x), c='b', label=r'$\alpha=%1f, \beta=%1f$' % (81, 219))

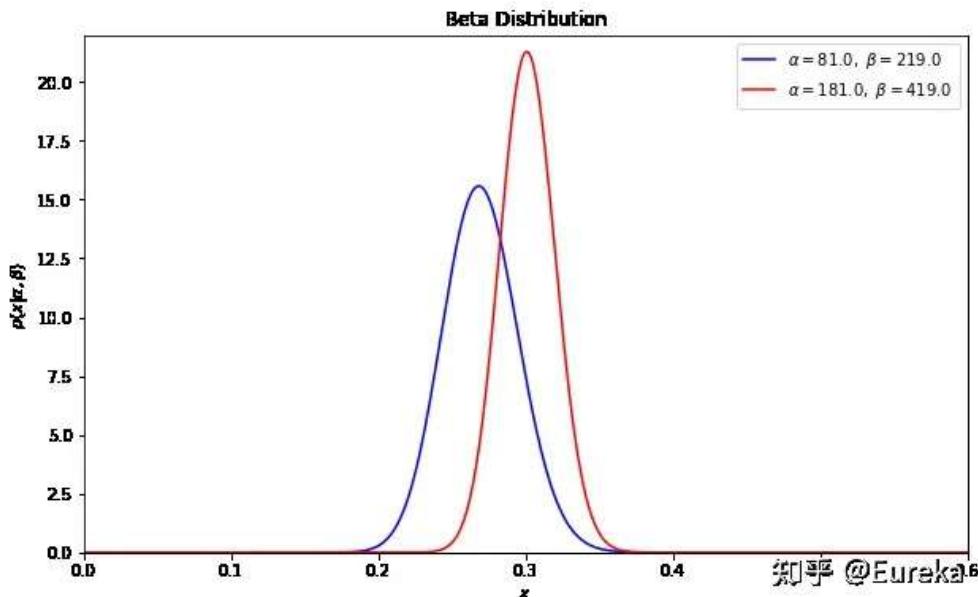
dist = beta(181, 419)
plt.plot(x, dist.pdf(x), c='r', label=r'$\alpha=%1f, \beta=%1f$' % (181, 419))

plt.xlim(0, .6)
plt.ylim(0, 22)

plt.xlabel('$x$')
plt.ylabel(r'$p(x|\alpha,\beta)$')
plt.title('Beta Distribution')

plt.legend(loc=0)
plt.show()

```



注意到这个曲线变得更加尖，并且平移到了一个右边的位置，表示比平均水平要高。

一个有趣的事情是，根据这个新的beta分布，我们可以得出他的数学期望为： $\frac{\alpha}{\alpha+\beta} = 0.303$ ，这一结果要比直接的估计要小 $\frac{100}{100+200} = 0.333$ 。你可能已经意识到，我们事实上就是在这个运动员在击球之前可以理解为他已经成功了81次，失败了219次这样一个先验信息。

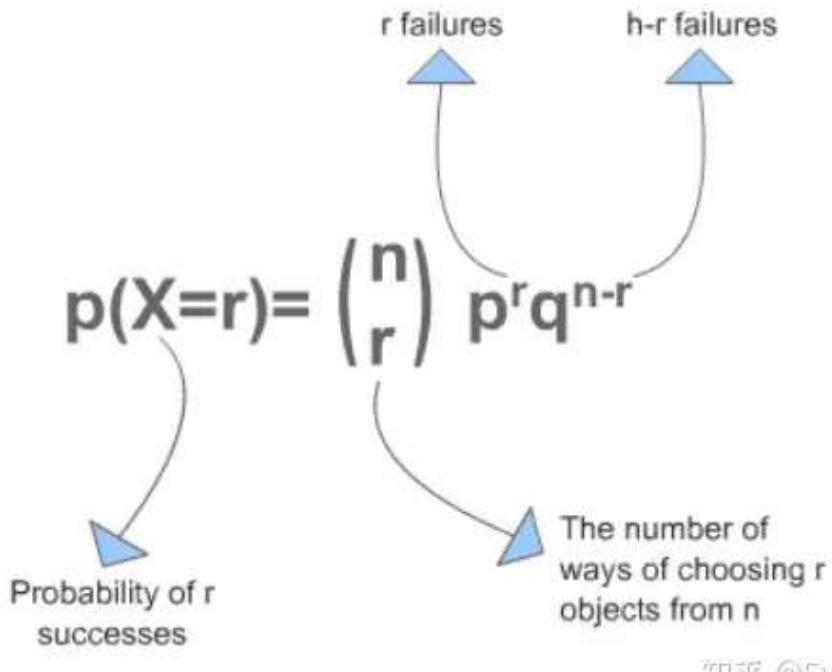
因此，对于一个我们不知道概率是什么，而又有一些合理的猜测时，beta分布能很好的作为一个表示概率的概率分布。

2. A/B测试

前面已大概了解了Beta分布的简单应用，我们再来深入了解下。

关于 **A/B** 测试，其实概念非常简单，简单来说，就是为同一个目标制定两个方案（比如两个页面），让一部分用户使用 **A** 方案，另一部分用户使用 **B** 方案，记录下用户的使用情况，看哪个方案更符合设计。

A 方案的转化率可以看作一个二项分布：



知乎 @Eureka

A方案的转化率分布就需要一个分布参数 p ，表示转化率的可能性。同样传统的频率学派会把实验总数中所有转化率的总数除以实验总数，得到这个 p 。以这个 p 为峰值获得一个类似高斯分布。贝叶斯学派不会假设 p 是固定不变的，他们会引入一个Beta分布作为二项分布的共轭先验，通过调整Beta分布参数，动态调整 p 的值。

通过前面我们已经知道 α, β 的含义， $\alpha + \beta$ 是实验的总次数， α 与 β 分别是实验不同结果的次数。下面我们看看 α, β 同比例增加会出现什么情况。

```

import numpy as np
from scipy.stats import beta
from matplotlib import pyplot as plt

x = np.linspace(0, 1, 1002)[1:-1]

fig, ax = plt.subplots(figsize=(10,6))

dist = beta(16, 84)
plt.plot(x, dist.pdf(x), c='b', label=r'$\alpha=%.1f, \beta=%.1f$' % (16, 84))

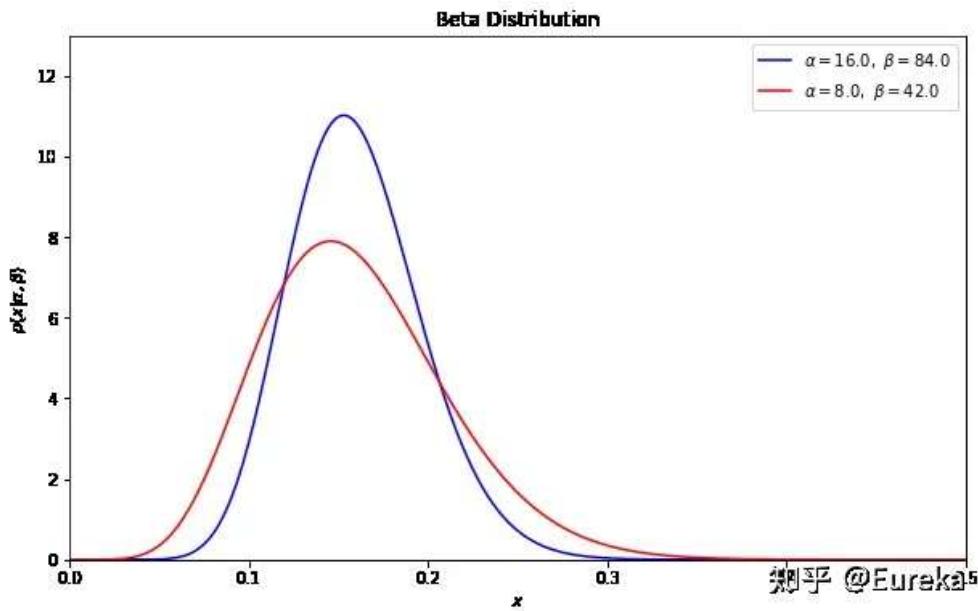
dist = beta(8, 42)
plt.plot(x, dist.pdf(x), c='r', label=r'$\alpha=%.1f, \beta=%.1f$' % (8, 42))

plt.xlim(0, .5)
plt.ylim(0, 13)

plt.xlabel('$x$')
plt.ylabel(r'$p(x|\alpha, \beta)$')
plt.title('Beta Distribution')

plt.legend(loc=0)
plt.show()

```



假设上图是一枚硬币抛100次有16次正面，和抛50次有8次正面的两个实验各自的Beta分布，可以看到虽然只是实验规模不同，但是分布密度图是不一样的：

第一：频率学派观点是应该猜测正面概率 $p=0.16$ ；**贝叶斯学派观点**是，以上两种情况的猜测到正面概率都小于0.16，因为**实验次数越少**，真实的正面和反面的差距就可能越大！

第二：实验次数越小，上面概率密度图应该越平缓（红线），因为少的实验次数不能增大决策信心。而蓝色的100次实验，明显有更大的信心猜测正面概率更接近0.16.

第三：实验次数越大，上面概率密度图的均值更应该接近0.16，符合大数定律。

是不是相当合理！

```

from scipy.stats import beta
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns

#考虑50人有8人转化成功
people_in_branch = 50

#控制组与实验组
control, experiment = np.random.rand(2, people_in_branch)

#控制组转化率是16%
c_successes = sum(control < 0.16)

#实验组要比控制组好10%
e_successes = sum(experiment < 0.176)

c_failures = people_in_branch - c_successes
e_failures = people_in_branch - e_successes

#先验
prior_successes = 8
prior_failures = 42

fig, ax = plt.subplots(figsize=(12,8))

#控制组
c_alpha, c_beta = c_successes + prior_successes, c_failures + prior_failures

```

```

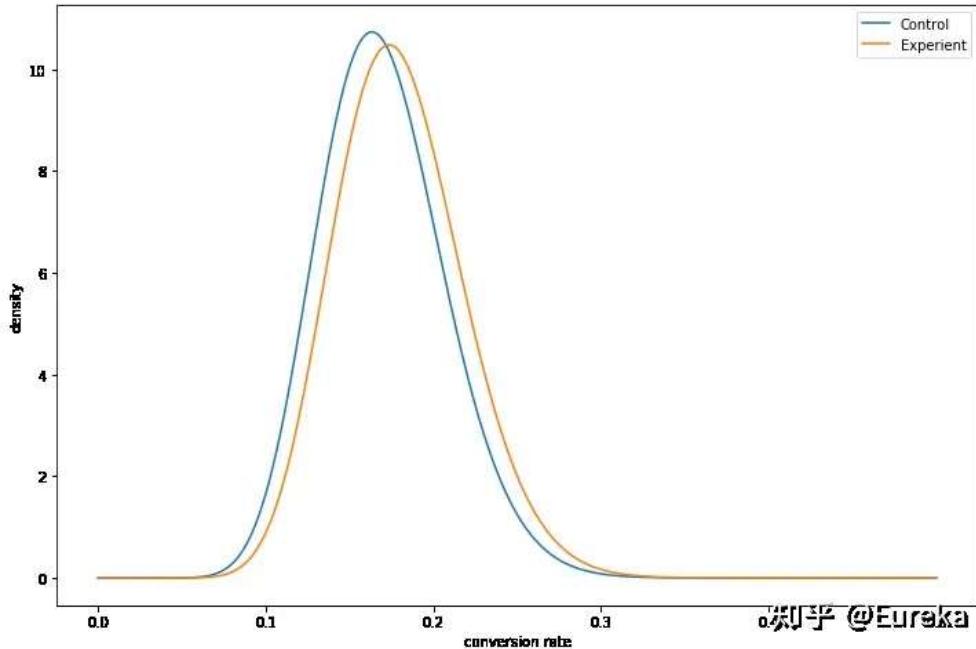
#实验组
e_alpha, e_beta = e_successes + prior_successes, e_failures + prior_failures

x = np.linspace(0., 0.5, 1000)

c_distribution = beta(c_alpha, c_beta)
e_distribution = beta(e_alpha, e_beta)

ax.plot(x, c_distribution.pdf(x),label='Control')
ax.plot(x, e_distribution.pdf(x),label='Experient')
plt.legend()
ax.set(xlabel='conversion rate', ylabel='density')
plt.show()

```



上图很多都重叠，我们无法说那个更好，下面我们增加数据量来得到更精确的信息：

```

#增加到4000人
more_people_in_branch = 4000

#控制组与实验组
control, experiment = np.random.rand(2, more_people_in_branch)

#加上已有的数据
c_successes += sum(control < 0.16)
e_successes += sum(experiment < 0.176)

c_failures += more_people_in_branch - sum(control < 0.16)
e_failures += more_people_in_branch - sum(experiment < 0.176)

fig, ax = plt.subplots(figsize=(12,8))

#控制组
c_alpha, c_beta = c_successes + prior_successes, c_failures + prior_failures
#实验组
e_alpha, e_beta = e_successes + prior_successes, e_failures + prior_failures

x = np.linspace(0., 0.5, 1000)

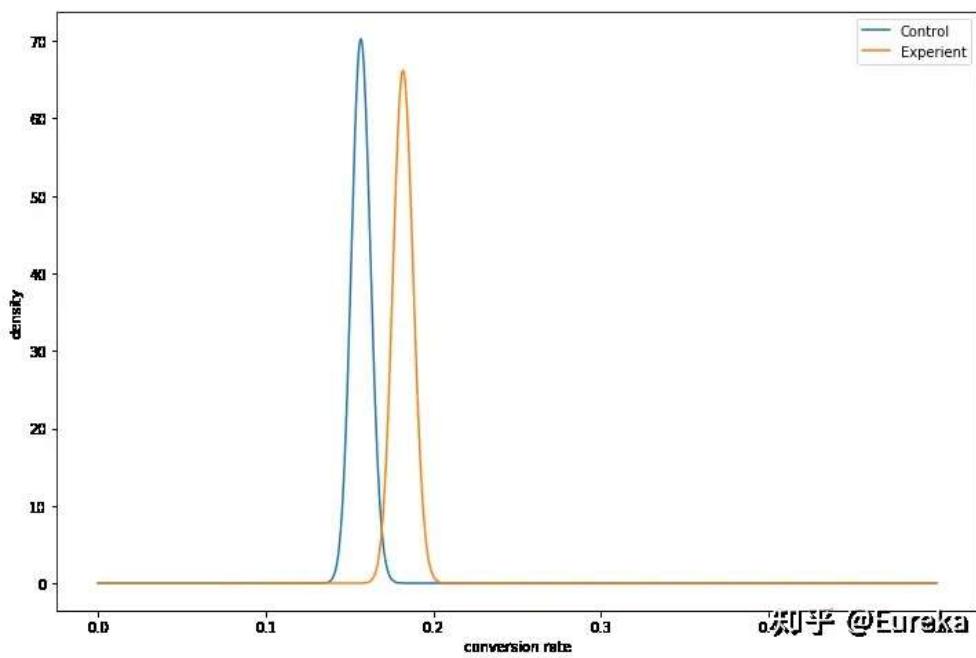
# Generate and plot the distributions!
c_distribution = beta(c_alpha, c_beta)
e_distribution = beta(e_alpha, e_beta)

```

```

ax.plot(x, c_distribution.pdf(x), label='Control')
ax.plot(x, e_distribution.pdf(x), label='Experient')
plt.legend()
ax.set(xlabel='conversion rate', ylabel='density')
plt.show()

```



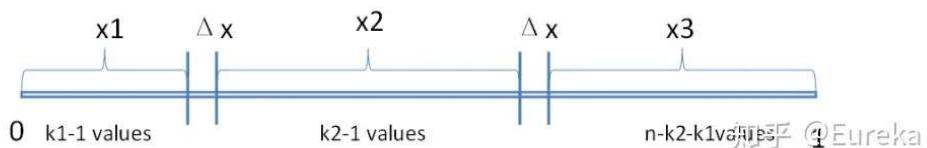
可以看出实验组要更好，又更高的转化率。

3: Dirichlet分布

3.1 认识Dirichlet分布

问题三：

1. $X_1, X_2, \dots, X_n \stackrel{iid}{\sim} Uniform(0, 1)$;
2. 排序后对应的顺序统计量 $X_{(1)}, X_{(2)}, \dots, X_{(n)}$;
3. 问 $(X_{(k_1)}, X_{(k_1+k_2)})$ 的联合分布是什么



同上面的推导：

$$\begin{aligned}
& P(X_{(k_1)} \in (x_1, x_1 + \Delta x), X_{(k_1+k_2)} \in (x_2, x_2 + \Delta x)) \\
&= n(n-1) \binom{n-2}{k_1-1, k_2-1} x_1^{k_1-1} x_2^{k_2-1} x_3^{n-k_1-k_2} (\Delta x)^2 \\
&= \frac{n!}{(k_1-1)!(k_2-1)!(n-k_1-k_2)!} x_1^{k_1-1} x_2^{k_2-1} x_3^{n-k_1-k_2} (\Delta x)^2
\end{aligned}$$

得到 $(X_{(k_1)}, X_{(k_1+k_2)})$ 的联合分布：

$$f(x_1, x_2, x_3) = \frac{n!}{(k_1-1)!(k_2-1)!(n-k_1-k_2)!} x_1^{k_1-1} x_2^{k_2-1} x_3^{n-k_1-k_2}$$

$$= \frac{\Gamma(n+1)}{\Gamma(k_1)\Gamma(k_2)\Gamma(n-k_1-k_2+1)} x_1^{k_1-1} x_2^{k_2-1} x_3^{n-k_1-k_2}$$

上面这个分布其实就是3维形式的Dirichlet分布 $Dir(x_1, x_2, x_3 | k_1, k_2, n - k_1 - k_2 + 1)$ 。令 $\alpha_1 = k_1, \alpha_2 = k_2, \alpha_3 = n - k_1 - k_2 + 1$ ，于是概率密度函数：

$$f(x_1, x_2, x_3) = \frac{\Gamma(\alpha_1 + \alpha_2 + \alpha_3)}{\Gamma(\alpha_1)\Gamma(\alpha_2)\Gamma(\alpha_3)} x_1^{\alpha_1-1} x_2^{\alpha_2-1} x_3^{\alpha_3-1}$$

其中 $\vec{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$ 延拓到非负实数集合，以上概率分布也是良定义的。

3.2 可视化Dirichlet分布

Dirichlet概率密度函数定义：一个连续随机向量

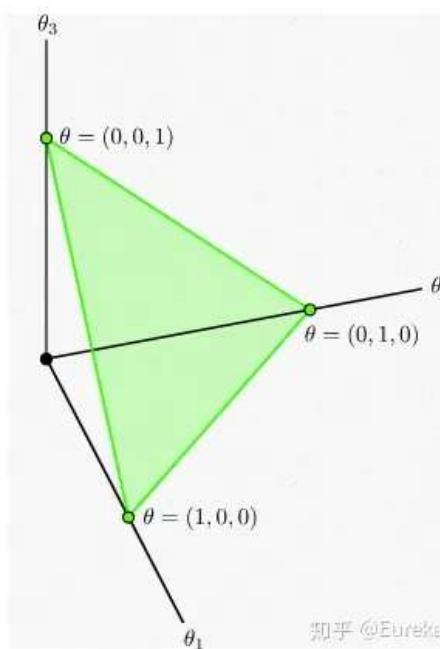
$\vec{X} = (x_1, \dots, x_k), (0 < x_i < 1, \sum_{i=1}^k x_i = 1)$ ，的 k 维Dirichlet分布，参数为 $\vec{\alpha} = (\alpha_1, \dots, \alpha_k), (\alpha_i > 0, i = 1, 2, \dots, k)$ ，它的概率密度函数如下：

$$Dir(\vec{\alpha}) \rightarrow P(\vec{X} | \vec{\alpha}) = \frac{\Gamma(\sum_{i=1}^k \alpha_i)}{\prod_{i=1}^k \Gamma(\alpha_i)} \prod_{j=1}^k x_j^{\alpha_j-1}$$

其中 $\vec{X}, \vec{\alpha}$ 维数是一样的。

由于Dirichlet分布描述的是多个定义于区间 $[0, 1]$ 的随机变量的概率分布，所以通常将其用作多项分布参数 θ_i 的概率分布。

下面考虑个三维的多项式分布，参数 θ_i 在如下平面上 (2-simplex)：



下面我们可视化Dirichlet概率密度函数，当 $\vec{\alpha}$ 给定时，看看 $Dir(\vec{\alpha})$ 在2-simplex上如何变化。

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.tri as tri
```

```

#生成等边三角形
corners = np.array([[0, 0], [1, 0], [0.5, 0.75**0.5]])
triangle = tri.Triangulation(corners[:, 0], corners[:, 1])

#每条边中点位置
midpoints = [(corners[(i + 1) % 3] + corners[(i + 2) % 3]) / 2.0 for i in range(3)]

def xy2bc(xy, tol=1.e-3):
    #将三角形顶点的笛卡尔坐标映射到重心坐标系
    s = [(corners[i] - midpoints[i]).dot(xy - midpoints[i]) / 0.75 for i in range(3)]
    return np.clip(s, tol, 1.0 - tol)

#有了重心坐标，可以计算Dirichlet概率密度函数的值
class Dirichlet(object):

    def __init__(self, alpha):
        from math import gamma
        from operator import mul
        from functools import reduce
        self._alpha = np.array(alpha)
        self._coef = gamma(np.sum(self._alpha)) / reduce(mul, [gamma(a) for a in self._alpha])

    def pdf(self, x):
        #返回概率密度函数值
        from operator import mul
        from functools import reduce
        return self._coef * reduce(mul, [xx ** (aa - 1) for (xx, aa) in zip(x, self._alpha)])

    def draw_pdf_contours(dist, nlevels=200, subdiv=8, **kwargs):
        import math

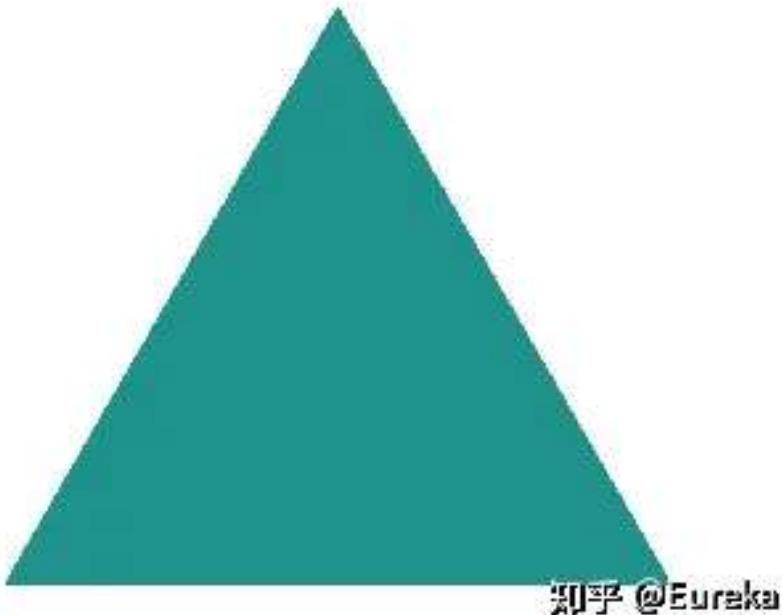
        #细分等边三角形网格
        refiner = tri.UniformTriRefiner(triangle)
        trimesh = refiner.refine_triangulation(subdiv=subdiv)
        pvals = [dist.pdf(xy2bc(xy)) for xy in zip(trimesh.x, trimesh.y)]

        plt.tricontourf(trimesh, pvals, nlevels, **kwargs)
        plt.axis('equal')
        plt.xlim(0, 1)
        plt.ylim(0, 0.75**0.5)
        plt.axis('off')

```

让我们看看几个对称的Dirichlet分布，第一个 $\alpha = [1, 1, 1]$ 产生一个均匀分布，所有点在 simplex 上概率相同。

```
draw_pdf_contours(Dirichlet([1, 1, 1]))
```



知乎 @Eureka

对于 $\alpha_i < 1$ ，分布集中在角落和simplex边界。（黄色概率最大，蓝色概率最小）

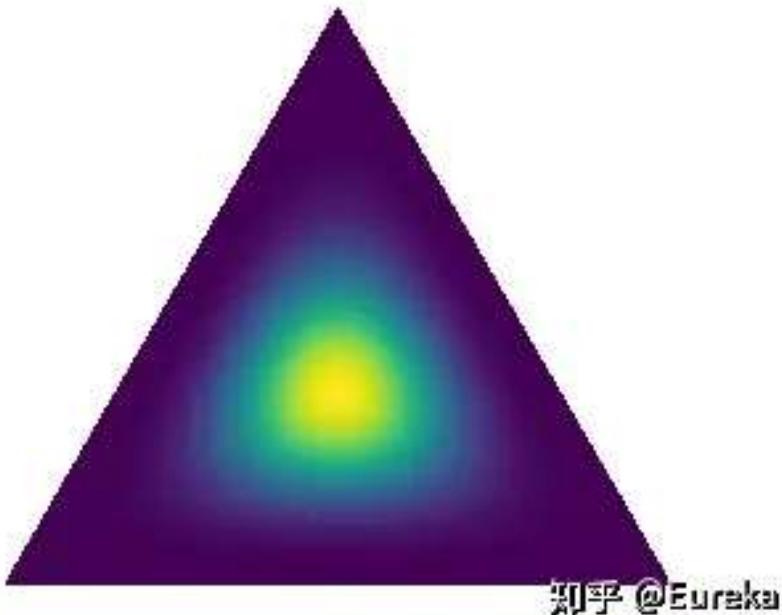
```
draw_pdf_contours(Dirichlet([0.999, 0.999, 0.999]))
```



知乎 @Eureka

对于 $\alpha_i > 1$ ，分布集中在simplex中心。

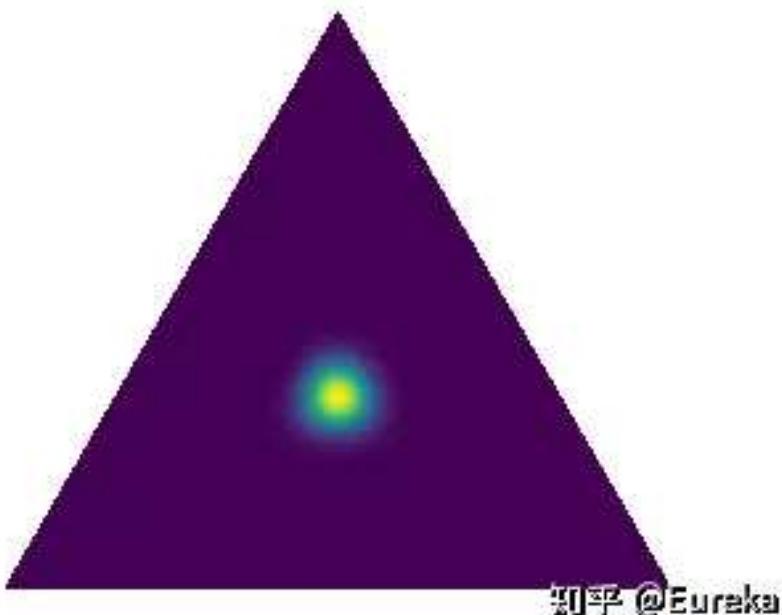
```
draw_pdf_contours(Dirichlet([5, 5, 5]))
```



知乎 @Eureka

随着 α_i 的增大，分布更加集中在simplex的中心

```
draw_pdf_contours(Dirichlet([50, 50, 50]))
```

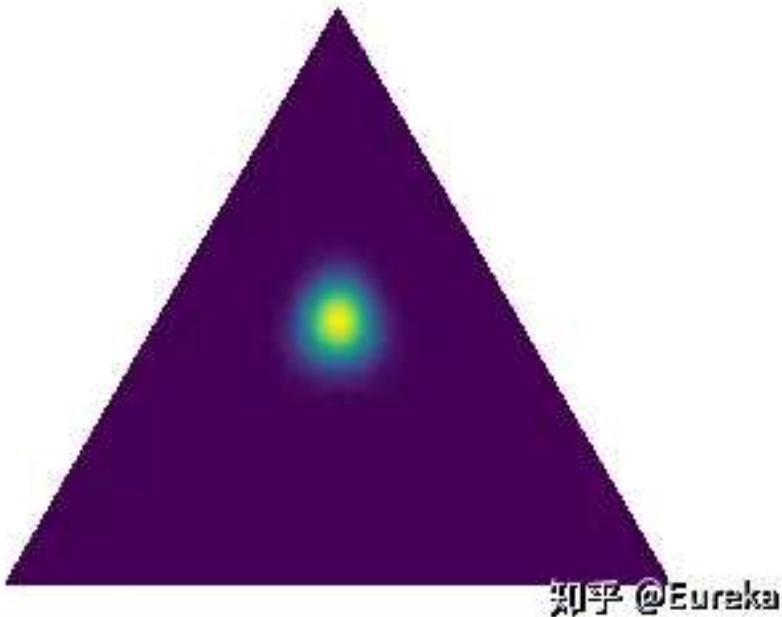


知乎 @Eureka

当 $\alpha_i \rightarrow \infty$ ，多项式分布的参数值将相等。

对于不对称的Dirichlet分布，如下 α_3 较其他值更大。

```
draw_pdf_contours(Dirichlet([30, 30, 50]))
```



知乎 @Eureka

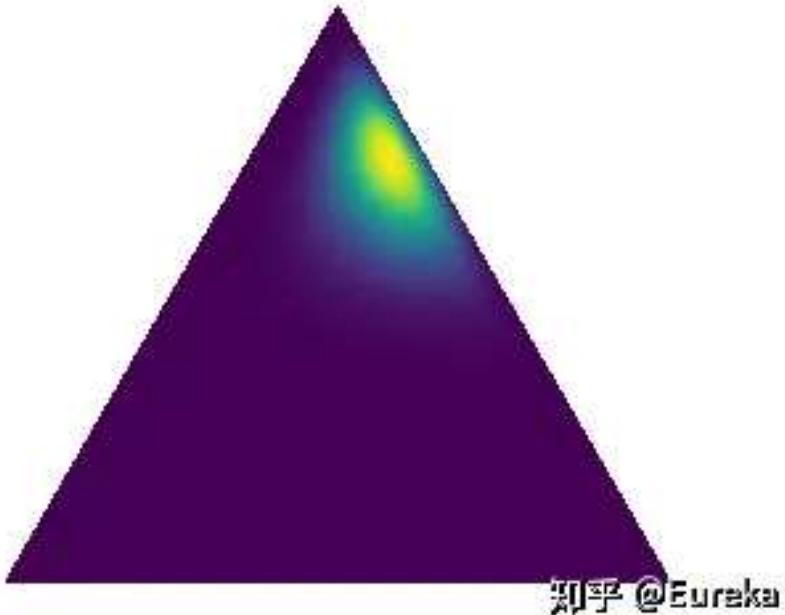
下面看看 α_i 的其他变化。

```
draw_pdf_contours(Dirichlet([1, 2, 3]))
```



知乎 @Eureka

```
draw_pdf_contours(Dirichlet([2, 5, 15]))
```



3.3 Dirichlet-Multinomial共轭

问题四：

1. $X_1, X_2, \dots, X_n \stackrel{iid}{\sim} Uniform(0, 1)$ ，排序后对应的顺序统计量 $X_{(1)}, X_{(2)}, \dots, X_{(n)}$ ；
2. 令 $p_1 = X_{(k_1)}, p_2 = X_{(k_1+k_2)}, p_3 = 1 - p_1 - p_2$ ，我们要猜测 $\vec{p} = (p_1, p_2, p_3)$ ；
3. $Y_1, Y_2, \dots, Y_m \stackrel{iid}{\sim} Uniform(0, 1)$ ， Y_i 中落到 $[0, p_1], [p_1, p_2], [p_2, 1]$ 三个区间的个数分别是 $m_1, m_2, m_3, m = m_1 + m_2 + m_3$ ；
4. 问后验分布 $P(\vec{p} | Y_1, Y_2, \dots, Y_m)$ 的分布是什么

为了方便我们记：

$$\vec{m} = (m_1, m_2, m_3), \vec{k} = (k_1, k_2, n - k_1 - k_2 + 1)$$

我们可以推导得到 p_1, p_2 在 $X_1, X_2, \dots, X_n, Y_1, Y_2, \dots, Y_m \stackrel{iid}{\sim} Uniform(0, 1)$ ，这 $m+n$ 个数中分别成为了第 $k_1+m_1, k_1+k_2+m_1+m_2$ 大的数。于是后验分布 $P(\vec{p} | Y_1, Y_2, \dots, Y_m)$ 应该是 $Dir(\vec{p} | k_1+m_1, k_2+m_2, n-k_1-k_2+1+m_3)$ ，即 $Dir(\vec{p} | \vec{k} + \vec{m})$ 。

按贝叶斯推导的逻辑：

- 1): 我们要猜测的参数 $\vec{p} = (p_1, p_2, p_3)$ ， p 的先验分布为 $Dir(\vec{p} | \vec{k})$ ；
- 2): 数据 Y_i 落到 $[0, p_1], [p_1, p_2], [p_2, 1]$ 三个区间的个数分别是 m_1, m_2, m_3 ，所以 $\vec{m} = (m_1, m_2, m_3)$ 服从多项分布 $Mult(\vec{m}, \vec{p})$ 。
- 3): 在给定来自数据提供的 \vec{m} 知识后， \vec{p} 的后验分布为 $Dir(\vec{p} | \vec{k} + \vec{m})$

以上贝叶斯分析过程的简单表述为：

$$Dir(\vec{p} | \vec{k}) + MultCount(\vec{m}) = Dir(\vec{p} | \vec{k} + \vec{m})$$

令 $\vec{k} = \vec{\alpha}$ ，把 $\vec{\alpha}$ 从整数集合延拓到实数集合，更一般的可以证明有如下关系：

$$Dir(\vec{p} | \vec{\alpha}) + MultCount(\vec{m}) = Dir(\vec{p} | \vec{\alpha} + \vec{m})$$

以上式子实际上描述的就是**Dirichlet-Multinomial共轭**，而我们从以上过程可以看到，Dirichlet分布中的参数 $\vec{\alpha}$ 都可以理解为物理计数。类似于Beta分布，我们也可以把 $Dir(\vec{p}|\vec{\alpha})$ 作如下分解：

$$Dir(\vec{p}|\vec{1}) + MultCount(\vec{m} - \vec{1}) = Dir(\vec{p}|\vec{\alpha})$$

此处 $\vec{1} = (1, 1, \dots, 1)$ 。

当然对于高维的Dirichlet与Multinomial也共轭。

Dirichlet分布是分布的原因：前面知道Dirichlet分布得到的向量各个分量的和是1，这个向量可以作为Multinomial分布的参数，所以我们说Dirichlet能够生成Multinomial分布，也就是分布的分布。Dirichlet分布和Multinomial分布式共轭的，所以Dirichlet和Multinomial这个组合总是经常被使用，Dirichlet分布在这里的角色就是分布的分布（Multinomial分布的分布）。在可视化部分也可以看到，Dirichlet不同的参数，Dirichlet分布得到的向量值不一样，可以对称，也可以不是对称的。

3.4 Beta/Dirichlet 分布的一个性质

如果 $p \sim Beta(t|\alpha, \beta)$ ，则：

$$\begin{aligned} E(p) &= \int_0^1 t * Beta(t|\alpha, \beta) dt \\ &= \int_0^1 t * \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1} dt \\ &= \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_0^1 t^\alpha (1-t)^{\beta-1} dt \end{aligned}$$

上式右边的积分对应到概率分布 $Beta(t|\alpha + 1, \beta)$ ，对于这个分布，我们有：

$$\int_0^1 \frac{\Gamma(\alpha + \beta + 1)}{\Gamma(\alpha + 1)\Gamma(\beta)} t^\alpha (1-t)^{\beta-1} dt = 1$$

把上式带入 $E(p)$ 的计算式，得到

$$\begin{aligned} E(p) &= \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \cdot \frac{\Gamma(\alpha + 1)\Gamma(\beta)}{\Gamma(\alpha + \beta + 1)} \\ &= \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha + \beta + 1)} \frac{\Gamma(\alpha + 1)}{\Gamma(\alpha)} \\ &= \frac{\alpha}{\alpha + \beta} \end{aligned}$$

这说明，对于Beta分布的随机变量，其均值可以用 $\frac{\alpha}{\alpha+\beta}$ 来估计。Dirichlet分布也有类似的结论，如果 $\vec{p} \sim Dir(\vec{t}|\vec{\alpha})$ ，同样可以证明：

$$E(\vec{p}) = \left(\frac{\alpha_1}{\sum_{i=1}^K \alpha_i}, \frac{\alpha_2}{\sum_{i=1}^K \alpha_i}, \dots, \frac{\alpha_K}{\sum_{i=1}^K \alpha_i}, \right)$$

4：参考文献

[LDA-math-神奇的Gamma函数 | 统计之都](#)

[Visualizing Dirichlet Distributions with Matplotlib](#)

[带你理解beta分布](#)

[用python做贝叶斯A/B测试 — 贝叶斯A/B测试入门 以及 “共轭先验” 是什么?](#)

编辑于 2018-06-12 19:24