首都师范大学

人工智能数学基础Python实践
第七章 概率论基础

张苗苗

信息工程学院

主要内容

- 在Python中实现以下操作：
  - 概率密度函数
  - 概率分布函数
  - 概率计算

- 所需的python库：
  - scipy中的stats模块

stats -- 统计模块

- Scipy中的stats模块包含了多种概率分布的随机变量，随机变量分为连续的和离散的两种。所有连续随机变量都是rv_continuous的派生类的对象，而所有离散随机变量都是rv_discrete的派生类的对象。

stats -- 统计模块

- stats提供了产生连续性分布的函数：
  - 均匀分布（uniform）
  - 正态分布（norm）
  - 贝塔分布（beta）
  - …
- 产生离散性分布的函数:
  - 伯努利分布（bernoulli）
  - 几何分布(geom)
  - 泊松分布（poisson）
  - …

使用时，调用分布的rvs即可

stats -- 统计模块

```
import scipy.stats as stats   # 导入stats模块

# 生成100个在[0,1]均匀分布的随机数
    x = stats.uniform.rvs(size=100)

# 生成20个服从[1,2]正态分布的随机数
    y = stats.norm.rvs(size=20,loc =1,scale=2)
```

stats -- 统计模块

stats连续型随机变量的公共方法：

名称	备注
rvs	产生服从指定分布的随机数
pdf	概率密度函数
cdf	累计分布函数
sf	残存函数（1-CDF）
ppf	分位点函数（CDF的逆）
isf	逆残存函数（sf的逆）
fit	对一组随机取样进行拟合，最大似然估计方法找出最适合取样数据的概率密度函数系数。

练习1-- 创建连续概率密度、分布函数和变量

在python中输出正态分布函数和分布图

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

```
from scipy.stats import norm
import numpy as np
import pylab as plt
X = norm()                    # 默认参数，loc=0，scale=1
Y = norm(loc=1.0,scale=2.0)   #loc 和 scale 参数，对应正态分布的期望和标准差
```

练习1-- 创建概率函数和分布图，以及变量

在python中输出正态分布函数和分布图

```
t = np.arange(-10,10,0.01)  # 范围
fig = plt.figure(figsize=(10,5))
fig.add_subplot(131)    # 子图1
plt.plot(t, X.pdf(t), label="$X$", color="red") # 概率密度
plt.plot(t, Y.pdf(t), "b--", label="$Y$")
plt.legend()
fig.add_subplot(132)  # 子图2
plt.plot(t, X.cdf(t), label="$X$", color="red") #概率分布
plt.plot(t, Y.cdf(t), "b--", label="$Y$")
plt.legend()
plt.show()
```

练习1-- 创建概率函数和分布图，以及变量

生成符合正态分布的变量

```
Y1 = Y.rvs(size=20000)  #变量
fig.add_subplot(133)
plt.hist(Y1)  #直方图
plt.show()

print(stats.norm.fit(Y1))  # 拟合系数
```

练习2-- 创建离散概率函数和分布图，变量

python中画出如下概率函数及分布函数图

X取值	0	1	2	3	4
对应概率	1/16	1/4	3/8	1/4	1/16

```
# 导入库，设置字体
from scipy import stats
import matplotlib.pyplot as plt
plt.rcParams["font.sans-serif"] = ["Microsoft YaHei"]
plt.rcParams['axes.unicode_minus'] = False
```

练习2-- 创建离散概率函数和分布图，以及变量

```
xk = [0,1,2, 3,4 ]    # 所有可能的取值
pk = [1/16, 1/4, 3/8, 1/4, 1/16]  # 各个取值的概率
#用rv_discrete 类自定义离散概率分布rvs
custome = stats.rv_discrete(values=(xk, pk))
#调用其rvs方法20次，获得符合概率的随机数:
custome1=custome.rvs(size=20)
```

练习2-- 创建离散概率函数和分布图，以及变量

```
## 画图
fig, (ax0, ax1) = plt.subplots(ncols=2, figsize=(10, 5))
#显示概率密度函数
ax0.set_title("概率函数")
ax0.plot(xk, pk, 'ro', ms=8, mec='r')
ax0.vlines(xk, 0, pk, colors='r', linestyles='-', lw=2)
```

练习2-- 创建离散概率函数和分布图，以及变量

python中画出如下概率函数及分布函数图

```
#显示"分布函数"
pk1=custome.cdf(xk)  # 分布函数
xmax=[1,2,3,4,5]
ax1.hlines(pk1,xk,xmax,linestyles='-',colors='b')
xk1=[1,2,3,4]
ymin=pk1[0:4]
ymax=pk1[1:5]
ax1.vlines(xk1,ymin , ymax, colors='b', linestyles='-', lw=2)
ax1.set_title("分布函数")
plt.show()
```

练习3-- 抛掷硬币的频率

在python中编程实现，抛10次硬币并计算正面朝上的次数和频率

```
#模拟扔10次硬币
import random
def coin_trail():
    heads = 0
    for i in range(10):
        if random.random()<=0.5:
            heads+=1
    return heads
```

```
#进行n次实验
def simulate(n):
    trails=[]
    for i in range(n):
        trails.append(coin_trail())
    return(sum(trails)/n)
```

首都师范大学

谢谢！