Cálculo de probabilidade por simulação

As probabilidades são calculadas a partir de um modelo de probabilidades (fórmula matemática). Uma outra maneira de resolver o problema é realizar o experimento aleatório muitas vezes, e observar a proporção de vezes que o evento ocorre. Dessa maneira estamos calculando a probabilidade por simulação.

```
import numpy as np
import time
```

Simulação 1 - Sorteio de 2 dados

Cálculo analítico

Seja o evento A a ocorrência dos números 3 e 6 no lançamento de dois dados. Podemos calcular a probabilidade intuitivamente contando as possibilidades de ocorrer os números 3 e 6 e o total de possibilidades para dois dados.

Se representarmos o lançamento de dois dados por um par de números (d1, d2), onde d1 é o resultado do dado 1 e d2 o resultado do dado 2, temos a seguinte tabela de possibilidades: $\{(3,6),(6,3)\}$.

$$P[A]=\frac{2}{36}=0,0556$$

Simulação Interativa

A função Dado simula interativamente o lançamento de dois dados e o calculo da probabilidade de observarmos os valores 3 e 6. A função np.random.randint(menor, maior) sorteia valores inteiros uniformemente distribuídos entre menor e maior-1.

```
# Simulacao iterativa
def Dado(n):
    deuCerto = 0                          # Inicia quatidade de vezes
que o evento ocorreu
    for i in range(n):                    # Executa n vezes o sorteio
dos dados
        d1 = np.random.randint(1, 7)    # sorteia dado 1 (número
inteiro aleatório entre 1 e 6)
        d2 = np.random.randint(1, 7)    # sorteia dado 2 (número
inteiro aleatório entre 1 e 6)
        if ((d1 == 3) & (d2 == 6)) | ((d1 == 6) & (d2 == 3)):  # Testa
se o evento ocorreu
            deuCerto = deuCerto + 1     # Incrementa quatidade de
vezes que o evento ocorreu
    return deuCerto/n
```

Os comandos abaixos simulam 50000 lançamentos de dois dados. Imprime a proporção de vezes que o par (3,6) foi observado (probabilidade simulada). Imprime o valor previsto pela teoria (probabilidade teórica). Imprime o tempo de simulação (em segundos).

```
t1 = time.perf_counter()
probS = Dado(50000)
t2 = time.perf_counter()
print('Probabilidade simulada:  {:.4f}'.format(probS))
print('Probabilidade teórica: {:.4f}'.format(2/36))
print('Tempo de simulação: {:.4f}'.format(t2-t1))
```

```
Probabilidade simulada:  0.0548
Probabilidade teórica: 0.0556
Tempo de simulação: 0.1967
```

Simulação 2 - Moeda

Simulação interativa

Lançar uma moeda até sair a primeira cara. Calcular a probabilidade do número de lançamentos necessários ser par.

Simular um lançamento de uma moeda:

- considerar 0 = cara
- considerar 1 = coroa
- utilizar np.ramdom.randint(0,2) para sortear

Simular o evento (quantidade de vezes que a moeda foi lançada até sair cara):

- implementar um laço que execurta até ser sorteado cara
- contar a quantidade de sorteios executados até sair cara

Testar se o evento ocorreu

- testar se a quantidade de sorteios (k) executados é par
- (k % 2) == 0 testa se K é par (resto da divisão por 2 igual a 0)

```
# Simulacao interativa
def Moeda(n):
    # Inicia quatidade de vezes que o evento ocorreu
    deuCerto = 0
    for i in range(n):  # Executa n vezes o sorteio da moeda
        # Sortear da moeda até sair cara
        # Contar quantas vezes sorteou
        # Se a quantidade de sorteios for par, incrementa deuCerto
(quantidade de vezes que o evento ocorreu)
        # Substituir o comando a seguir pelo algoritmo
        k = 0
        moeda = 0
        while(moeda != 1):
```

```
            k = k+1
            moeda = np.random.randint(0, 2)
        if (k % 2 == 0):
            deuCerto = deuCerto + 1

    return deuCerto/n
```

Os comandos abaixos simulam 50000 o lançamento de uma moeda até sair a primeira cara. Imprime a proporção de vezes que foram necessários um número par de lançamentos (probabilidade simulada). Imprime o valor previsto pela teoria (probabilidade teórica). Imprime o tempo de simulação (em segundos).

```
import time
t1 = time.perf_counter()
probS = Moeda(100000)
t2 = time.perf_counter()
print('Probabilidade simulada:  {:.4f}'.format(probS))
print('Probabilidade teórica: {:.4f}'.format(1/3))
print('Tempo de simulação: {:.4f}'.format(t2-t1))
```

```
Probabilidade simulada:  0.3312
Probabilidade teórica: 0.3333
Tempo de simulação: 0.4215
```

Entrega

- Completar o código da função Moeda
- Imprimir o netebook para pdf
- Fazer upload do pdf no AVA