Métodos de Estatística Aplicada com Python

Aula 4

Carlos Góes[1]

[1]Pós-Graduação em Ciência de Dados
Instituto de Educação Superior de Brasília

2017

Sumário

Sumário

Médias enganam

Introdução

- Médias são estatísticas que sumarizam um número N de observações.
- Mais formalmente, médias são MEDIDAS DE TENDÊNCIA CENTRAL de uma variável aleatória.
- Elas são definidas da seguinte maneira:

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + \ldots + x_N}{N} \tag{1}$$

- Mas o que isso realmente quer dizer?

Médias enganam

Introdução

- Vamos supor que nós utilizemos um dado não viciado de seis lados para obter números aleatórios
- Com base nesses resultados, podemos calcular a média para cada horizonte de N repetições:

$$\begin{bmatrix} \bar{x_1} \\ \bar{x_2} \\ \vdots \\ \bar{x_N} \end{bmatrix}_{Nx1} = \begin{bmatrix} \frac{1}{1}\sum_{i=1}^{1} x_i \\ \frac{1}{2}\sum_{i=1}^{2} x_i \\ \vdots \\ \frac{1}{N}\sum_{i=1}^{N} x_i \end{bmatrix}_{Nx1} \quad (2)$$

Médias enganam

Introdução

- Primeiro, importamos os pacotes necessário.

```
import random
import numpy as np
import matplotlib.pyplot as plt
```

- Depois, criamos uma função para medir a média acumulada:

```
def media_cum(tamanho_amostra):
    amostra = [random.randint(1,6) for
    i in range(tamanho_amostra)]
    numerador = np.cumsum(amostra)
    denominador = [i+1 for i in range(tamanho_amostra)]
    return numerador / denominador
```

- Usamos essa função para criar três simulações:

```
tamanho = 500

sim1 = media_cum(tamanho)
sim2 = media_cum(tamanho)
sim3 = media_cum(tamanho)
```

Médias enganam

Introdução

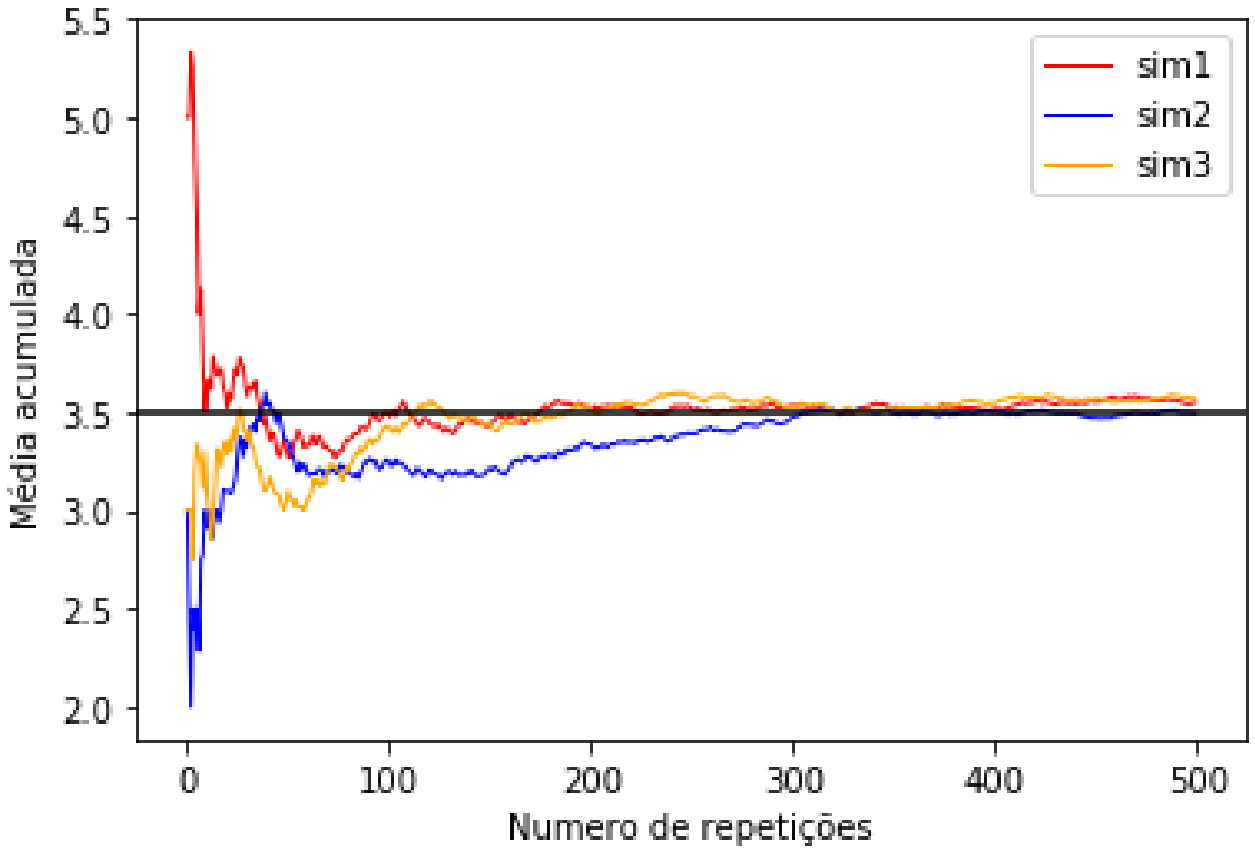
- Finalmente, vamos plotar os resultados.

```
x = range(tamanho)

plt.plot(x, sim1, color='red', linewidth=1, label="sim1")
plt.plot(x, sim2, color='blue', linewidth=1, label="sim2")
plt.plot(x, sim3, color='orange', linewidth=1, label="sim3
plt.axhline(3.5, color='black')
plt.legend(loc="upper right")
plt.xlabel("Numero de repetições")
plt.ylabel("Média acumulada")
plt.show()
```

Médias enganam

Introdução

Desenho da máquina:



Médias enganam

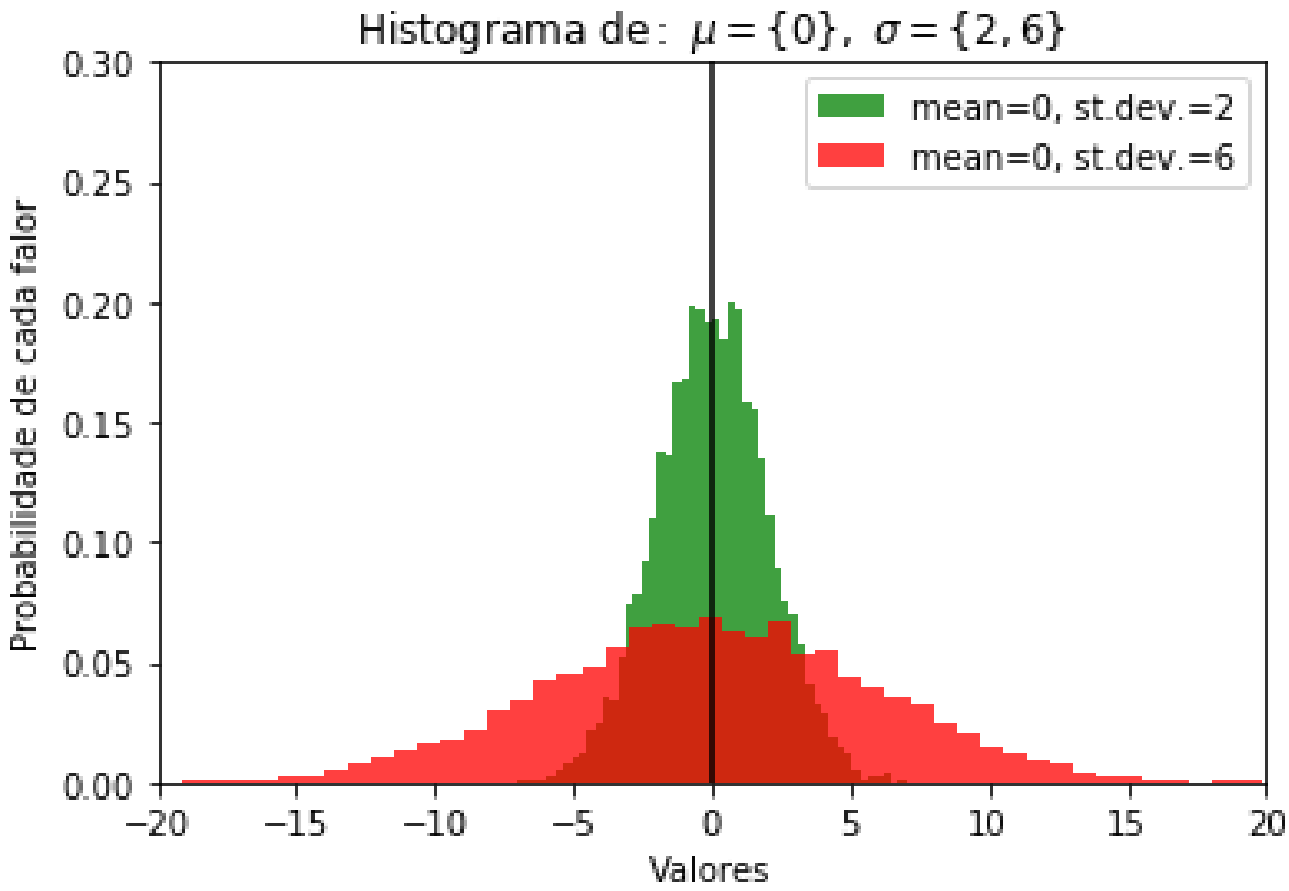
Variabilidade ao redor da média

- O Romário ($1,67m$) e o Shaquille O'Neal ($2,16m$) são em média bem altos: $\bar{x} = 1,915m$
- Obviamente, isso não quer dizer muita coisa, nem muda o fato de o Romário ser baixinho.
- Vamos comparar duas distribuições de média zero, mas com níveis distintos de variabilidade ao redor da média...

Médias enganam

Introdução

Qual a diferença entre elas?



Médias enganam

Variabilidade ao redor da média

- Todas as medidas de centralidade da distribuição (média, mediana e moda) indicariam a mesma coisa.
- Mas essas distribuições são claramente diferentes.
- Cerca de 20% das observações em verde estão ao redor de zero, enquanto apenas cerca de 6% estão ao redor de zero na distribuição vermelha.
- Por isso, é preciso ir além das medidas de tendência central...

Sumário

Medidas de Dispersão

Variância

- Se as médias, modas e medianas das duas distribuições são iguais, como podemos descrever as diferenças entre elas?
- Por outra medida, que chamamos de variância.

Medidas de Dispersão

Variância

- Variância é a média do quadrado do desvio entre cada observação e a média da distribuição.
- Matematicamente, ela se define por:

$$Var(x) = \sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)^2 \tag{3}$$

$$Var(x) = \sigma^2 = \frac{(x_1 - \mu_x)^2 + \ldots + (x_N - \mu_x)^2}{N}$$

Medidas de Dispersão

Variância

- Intuição: variância é uma medida de dispersão ao redor da média.
- Ex: $x = \{1, 10, 25, 34\}$, o que significa que $\bar{x} = 15,7$
- Portanto:

$$\begin{aligned} \sigma^2 &= \frac{(1-15,7)^2 + (10-15,7)^2 + (25-15,7)^2 + (34-15,7)^2}{4} \\ \sigma^2 &= \frac{272,25 + 56,25 + 56,25 + 272,25}{4} \\ \sigma^2 &= 164.25 \end{aligned} \tag{4}$$

Medidas de Dispersão

Variância

- Podemos escrever uma função no python para calcular todos esses passos para a gente:

```
import numpy as np

def variancia(amostra):
    media = np.mean(amostra)
    var_vec = []
    for elemento in amostra:
        var_elemento = (elemento - media) ** 2
        var_vec.append(var_elemento)
    variancia = np.sum(var_vec) / len(amostra)
    return variancia

x = [1, 10, 25, 34]

variancia(x)
```

- Ou simplesmente usar o numpy:

```
np.var(x)
```

Medidas de Dispersão

Desvio Padrão

- Uma outra forma de expressar a variância é utilizar o desvio padrão, que é definido simplesmente como a raiz quadrada da variância:

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)^2} \tag{5}$$

- Uma das vantagens de usar o desvio padrão é que ele é mais intuitivo, porque as suas unidades são expressas na mesma unidade da variável aleatória

Medidas de Dispersão

Desvio Padrão

- Partindo da função anterior:

```
def variancia(amostra):
    media = np.mean(amostra)
    var_vec = []
    for elemento in amostra:
        var_elemento = (elemento - media) ** 2
        var_vec.append(var_elemento)
    variancia = np.sum(var_vec) / len(amostra)
    return variancia
```

- Podemos criar uma nova que calcula o desvio padrão:

```
def desvio_padrao(amostra):
    return variancia(amostra) ** (1/2)
```

Medidas de Dispersão

Desvio Padrão

- Desvio padrão e variância são medidas de dispersão ao redor da média.
- Por isso, quando uma nova observação é adicionada a uma distribuição, ela pode aumentar ou reduzir o desvio padrão.

Intuição

O quão mais próximos (distantes) os valores estiverem da média, menor (maior) vai ser o desvio padrão

Medidas de Dispersão

Desvio Padrão

Dica

Os números em si não importam, o que importa é qual a distância ao redor da média

- Considere as duas seguintes populações:
  - $x = \{10, 20, 30\}$
  - $y = \{110, 120, 130\}$
- Calculando o desvio padrão de x:
  - $\sigma_x^2 = 1/3[(10-20)^2 + (20-20)^2 + (30-20)^2] = 1/3[100 + 0 + 100] = 200/3 = 66,66...$
  - $\sigma_x = \sqrt{\sigma_x^2} = 8,16$
- Calculando o desvio padrão de y:
  - $\sigma_y^2 = 1/3[(110-120)^2 + (120-120)^2 + (130-120)^2] = 1/3[100 + 0 + 100] = 200/3 = 66,66...$
  - $\sigma_y = \sqrt{\sigma_y^2} = 8,16$

Medidas de Dispersão

Desvio Padrão

- Vamos testar no Python?

```
x = [10, 20, 30]
y = [110, 120, 130]

print(desvio_padrao(x),
desvio_padrao(y))
```

Medidas de Dispersão

Desvio Padrão

- Se nós somarmos um mesmo número a todos os elementos de um conjunto (movendo a média), o desvio padrão não é alterado...
- ...mas e se nós multiplicarmos todos os números por um fator?
- Vamos testar no Python?

```
x = [10, 20, 30]
y = [elemento * 2 for elemento in x]

print(desvio_padrao(x),
desvio_padrao(y))
```

Medidas de Dispersão

Desvio Padrão

- Em outras palavras, somar um mesmo número a todos os elementos não altera o desvio padrão:

$$\sigma_x = \sigma_{x+10} \tag{6}$$

- ...mas e se nós multiplicarmos todos os números por um fator?
- Vamos testar no Python?

```
x = [10, 20, 30]
y = [elemento * 2 for elemento in x]

print(desvio_padrao(x),
desvio_padrao(y))
```

Sumário

Médias enganam

Amostras e Populações

- Nos estamos interessados em estimar os parâmetros da população
- Mas colher informações sobre todo o conjunto da população é complexo, caro e muitas vezes inviável
- Por isso, queremos uma amostra representativa de onde podemos extrair estatísticas que nos informem sobre os parâmetros da população

Médias enganam

Amostras e Populações

- Ocorre que mesmo uma amostra representativa, naturalmente, não é exatamente idêntica à população...
- Vamos testar?

Médias enganam

Simulação para intuição

- Importe os pacotes

```
import numpy as np
import matplotlib.pyplot as plt
```

- Defina a média, desvio padrão, tamanho da população e da amostra

```
mu = 0
sigma = 10
pop_tam = 10000
amostra_tam = 500
```

- Crie uma população e selecione aleatoriamente uma amostra dessa população

```
pop = np.random.normal(mu, sigma, pop_tam)
amostra = np.random.choice(pop, size=amostra_tam)
```

Médias enganam

Simulação para intuição

- Crie dois histogramas para comparar a amostra e a população:

```
hist2 = plt.figure()

barras = 50

plt.hist(pop, # a variável
         bins=barras, # número de barras
         normed=True, # valores em percentuais
         facecolor='green', # cor
         alpha=0.75, # transparência
         label="população") # rótulo

plt.hist(amostra, # a variável
         bins=barras, # número de barras
         normed=True, # valores em percentuais
         facecolor='brown', # cor
         alpha=0.75, # transparência
         label="amostra") # rótulo
```

Médias enganam

Simulação para intuição

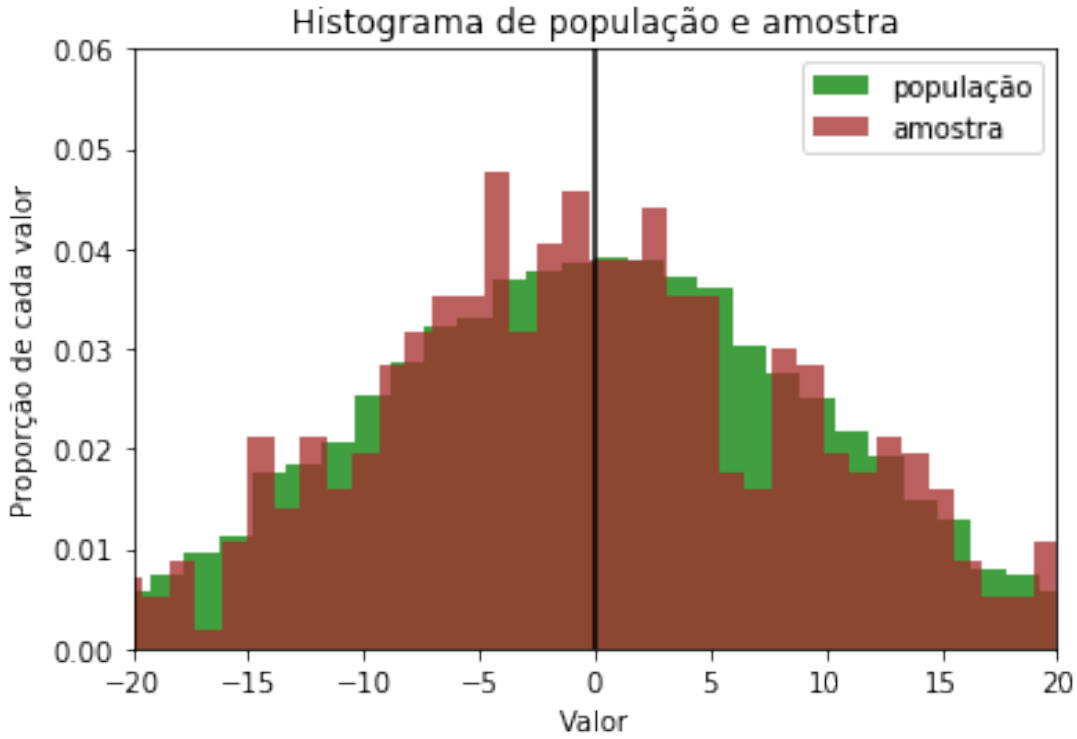
- Faça a formatação e imprima o gráfico:

```
plt.axvline(mu, color='black') # linha vertical

plt.legend(loc='upper right')
plt.xlabel('Valor')
plt.ylabel('Proporção de cada valor')
plt.title('Histograma de população e amostra')
plt.axis([-20, 20, 0, 0.06])  # limites dos eixos
plt.show()  # mostrar
```

Médias enganam

Simulação para intuição

- Impressão da máquina



Médias enganam

Erro Padrão da Média

- Se nossa amostra não é exatamente igual à população as estatísticas podem variar a depender da amostra selecionada
- Mesmo que a amostra não seja enviesada, existe uma imprecisão contida no próprio processo de amostragem
- Vamos entender isso?

Médias enganam

Erro Padrão da Média

```
import numpy as np
import matplotlib.pyplot as plt

mu = 0
sigma = 10
n_amostras = 10
amostra_tam = 500

pop = np.random.normal(mu, sigma, pop_tam)

amostra = np.matrix([[0 for x in range(n_amostras)]
for y in range(amostra_tam)])

medias = []
for i in range(N_amostras):
    s = np.random.choice(pop, size=amostra_tam)
    amostra[:,i] = np.transpose(np.matrix(s))
    media = s.mean()
    medias.append(media)
    print("Amostra " + str(i+1) + ", média: {}".format(media) )
```

Médias enganam

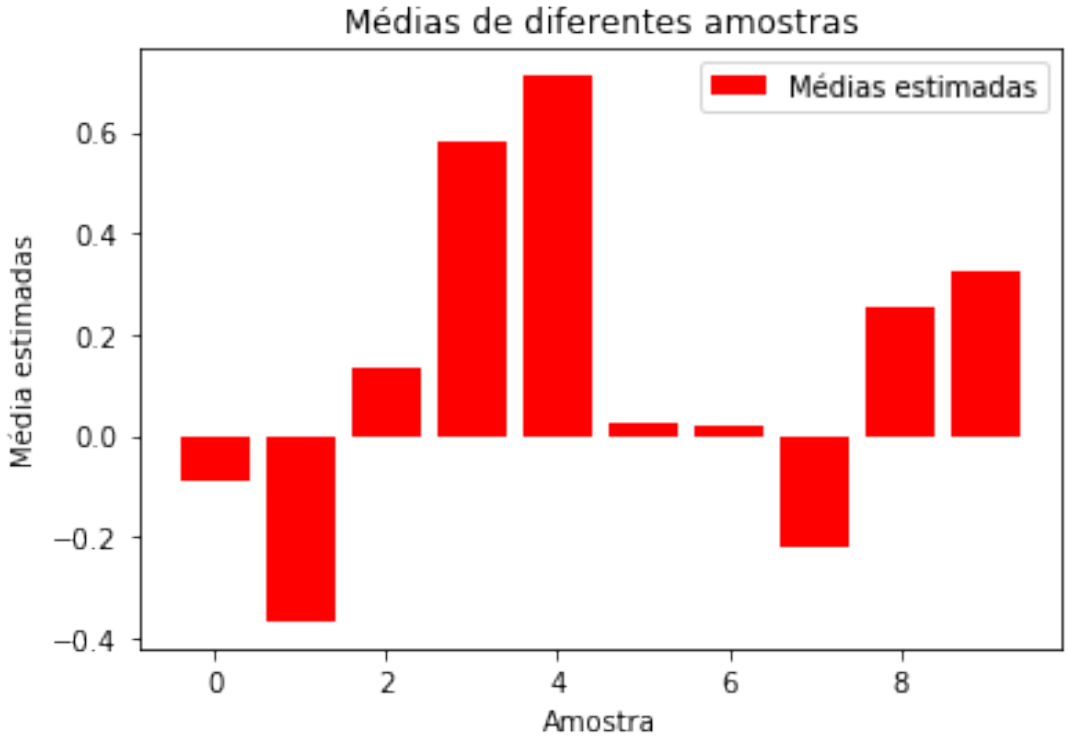
Erro Padrão da Média

```
barras = plt.figure()

plt.bar(range(n_amostras), means, color='red',
label='Médias estimadas')

plt.legend(loc=1)
plt.xlabel('Amostra')
plt.ylabel('Média estimadas')
plt.title('Médias de diferentes amostras')

plt.show()
```

Médias enganam

Erro Padrão da Média

- Impressão da máquina



Médias enganam

Erro Padrão da Média

- Existe uma incerteza própria do processo de amostragem
- Como saber o quão precisas são nossas médias?
- Estimando o "erro padrão"
- Matematicamente, o erro padrão da média se define como o desvio padrão da amostra dividido pelo tamanho da amostra:

$$s_\mu = \frac{s}{\sqrt{n}} \tag{7}$$

Intuição

Quanto maior o desvio padrão, maior o erro padrão. Quanto maior a amostra, menor o erro padrão.

Médias enganam

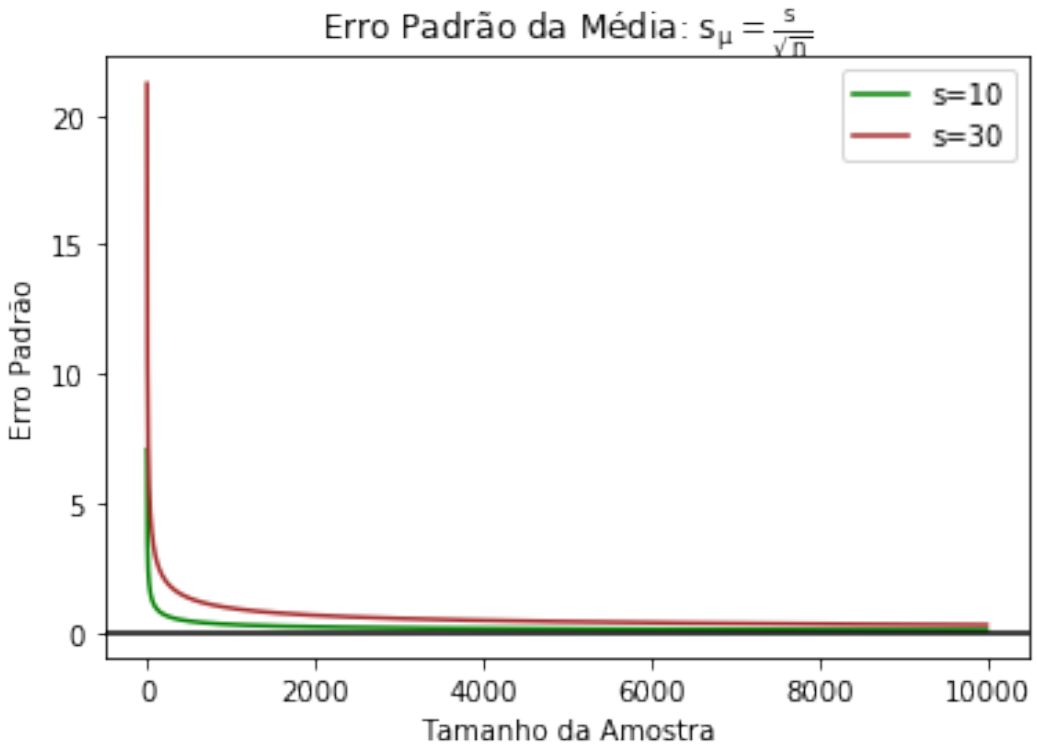
Erro Padrão da Média

```
sigma = 10
nn = np.linspace(2,10000,num=10000)

se1,se2 = [],[]
for n in nn:
    se1.append(sigma / np.sqrt(n))
    se2.append(3*sigma / np.sqrt(n))

seplot = plt.figure()

plt.plot(nn, se1, color='green', label='s=10')
plt.plot(nn, se2, color='brown', label='s=30')

plt.axhline(y=0, color='black')

plt.legend(loc=1)
plt.xlabel('Tamanho da Amostra')
plt.ylabel('Erro Padrão')
plt.title('Erro Padrão da Média: '
r'$\mathrm{ s_\mu = \frac{s}{\sqrt{n}}}$')
plt.show()
```

Médias enganam

Erro Padrão da Média

- Impressão da máquina



Médias enganam

Erro Padrão da Média

- Incluindo medidas incerteza em estimativas de amostra...

Médias enganam

Erro Padrão da Média

```
mu = 0
sigma = 10
n_amostras = 10
amostra_tam = 500

pop = np.random.normal(mu, sigma, pop_tam)

amostra = np.matrix([[0 for x in range(n_amostras)]
for y in range(amostra_tam)])

erros, medias = [], []
for i in range(n_amostras):
    s = np.random.choice(pop, size=amostra_tam)
    amostra[:,i] = np.transpose(np.matrix(s))
    media = s.mean()
    medias.append(media)
    erro = 1.96 * (sigma / np.sqrt(amostra_tam))
    erros.append(erro)
    print("Amostra " + str(i+1) + ", média: {:0.2f};
    erro-padrão: {:0.2f}".format(media, erro) )
```

Médias enganam

Erro Padrão da Média

```
barras = plt.figure()

plt.bar(range(n_amostras), medias, color='red',
label='Médias estimadas', yerr=erros)

plt.legend(loc=1)
plt.xlabel('Amostra')
plt.ylabel('Média estimadas')
plt.title('Médias de diferentes amostras')

plt.show()
```

Médias enganam

Erro Padrão da Média

- Impressão da máquina